

```

4024      ;
4025      .PAGE
4026      .TITLE 'DISPLAY HANDLER -- 10-30-78 -- DISPLC'
4027      ;
4028      ; HANDLER DEPENDENT EQUATES
4029      ;
4030      007D      CLRCOD      =      $7D      ; CLEAR SCREEN ATASCI CODE
4031      009E      CNTL1      =      $9F      ; POKEY KEY CODE FOR ^1
4032      ;
4033      0068      FRMADR      =      SAVADR
4034      0066      TOADR      =      MLTTMP
4035      ;
4036      .PAGE
4037      ;
4038      ;
4039      ;*=EDITRV
4040      ;
4041      ; SCREEN EDITOR HANDLER ENTRY POINT
4042      ;
4043      E400      FB F3      EDITOR:      .WORD EOPEN-1
4044      E402      33 F6      .WORD RETUR1-1      ; (CLOSE)
4045      E404      3D F6      .WORD EGETCH-1
4046      E406      A3 F6      .WORD EOUTCH-1
4047      E408      33 F6      .WORD RETUR1-1      ; (STATUS)
4048      E40A      3C F6      .WORD NOFUNC-1      ; (SPECIAL)
4049      E40C      4C E4 F3      JMP PWRONA
4050      E40F      00      .BYTE 0      ; ROM FILLER BYTE
4051      ;
4052      ;*=SCRENV
4053      ;
4054      ; DISPLAY HANDLER ENTRY POINT
4055      ;
4056      E410      F5 F3      DISPLA:      .WORD DOPEN-1
4057      E412      33 F6      .WORD RETUR1-1      ; (CLOSE)
4058      E414      92 F5      .WORD GETCH-1
4059      E416      B6 F5      .WORD DUTCH-1
4060      E418      33 F6      .WORD RETUR1-1      ; (STATUS)
4061      E41A      FB FC      .WORD DRAW-1      ; (SPECIAL)
4062      E41C      4C E4 F3      JMP PWRONA

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 2

```

4063 E41F 00 .BYTE 0 :ROM FILLER BYTE
4064 ;
4065 *=KEYBDV
4066 ;
4067 ;
4068 ; KEYBOARD HANDLER ENTRY POINT
4069 ;
4070 E420 33 F6 KBDHND: .WORD RETUR1-1
4071 E422 33 F6 .WORD RETUR1-1 ; (CLOSE)
4072 E424 E1 F6 .WORD KGETCH-1
4073 E426 3C F6 .WORD NOFUNC-1 ; (DUTCH)
4074 E428 33 F6 .WORD RETUR1-1 ; (STATUS)
4075 E42A 3C F6 .WORD NOFUNC-1 ; (SPECIAL)
4076 E42C 4C E4 F3 JMP PWRONA
4077 E42F 00 .BYTE 0 ; ROM FILLER BYTE
4078 ;
4079 ;
4080 ; INTERRUPT VECTOR TABLE ENTRY
4081 *=VCTABL-INTABS+VKEYBD
4082 E488 BE FF .WORD PIRQ5 ; KEYBOARD IRQ INTERRUPT VECTOR
4083 ;
4084 *=KBDORG
4085 ;
4086 F3E4 A9 FF PWRONA: LDA #$FF
4087 F3E6 8D FC 02 STA CH
4088 F3E9 AD E6 02 LDA MEMTOP+1
4089 F3EC 29 F0 AND #$F0 ; INSURE 4K PAGE BOUNDARY
4090 F3EE 85 6A STA RAMTOP
4091 F3F0 A9 40 LDA #$40 ; DEFAULT-TO UPPER CASE ALPHA AT PWRON
4092 F3F2 8D BE 02 STA SHFLOK
4093 F3F5 60 RTS ; POWER ON COMPLETED
4094 .PAGE
4095 ;
4096 ;
4097 ; BEGIN DISPLAY HANDLER OPEN PROCESSING
4098 ;
4099 F3F6 A5 2B DOPEN: LDA ICAX2Z ; GET AUX 2 BYTE
4100 F3F8 29 0F AND #$F
4101 F3FA D0 08 BNE OPNCOM ; IF MODE ZERO. CLEAR ICAX1Z

```

```

4102 F3FC A5 2A      EOPEN:    LDA    ICAX1Z      ; CLEAR "CLR INHIBIT" AND "MXD MODE" BITS
4103 F3FE 29 0F      AND      #$F
4104 F400 85 2A      STA    ICAX1Z
4105 F402 A9 00      LDA    110
4106 F404 85 57      OPNCOM:    STA    DINDEX
4107 F406 A9 E0      LDA    #$E0      ; INITIALIZE GLOBAL VBLANK RAM
4108 F408 8D F4 02    STA    CHBAS
4109 F40B A9 02      LDA    #2
4110 F40D 8D F3 02    STA    CHACT
4111 F410 8D 2F 02    STA    SDMCTL      ; TURN OFF DMA NEXT VBLANK
4112 F413 A9 01      LDA    #SUCCE
4113 F415 85 4C      STA    DSTAT      ; CLEAR STATUS
4114 F417 A9 C0      LDA    #$C0      ; DO IRQEN
4115 F419 05 10      ORA    POKMSK
4116 F41B 85 10      STA    POKMSK
4117 F41D 8D 0E D2    STA    IRQEN
4118 F420 A9 00      LDA    #0
4119 F422 8D 93 02    STA    TINDEX      ; TEXT INDEX MUST ALWAYS BE 0
4120 F425 85 64      STA    ADRESS
4121 F427 85 7B      STA    SWPFLG
4122 F429 8D F0 02    STA    CRSINH      ; TURN CURSOR ON AT OPEN
4123 F42C A0 0E      LDY    #14      ; CLEAR TAB STOPS
4124 F42E A9 01      LDA    #1      ; INIT TAB STOPS TO EVERY 8 CHARACTERS
4125 F430 99 A3 02    CLRTBS:    STA    TABMAP,Y
4126 F433 88          DEY
4127 F434 10 FA      BPL    CLRTBS
4128 F436 A2 04      LDX    #4      ; LOAD COLOR REGISTERS
4129 F438 BD C1 FE      DOPEN8:    LDA    COLRTB,X
4130 F43B 9D C4 02    STA    COLOR0,X
4131 F43E CA          DEX
4132 F43F 10 F7      BPL    DOPENB
4133 F441 A4 6A      LDY    RAMTOP      ; DO TXTMSC=$2C40 (IF MEMTOP=3000)
4134 F443 88          DEY
4135 F444 8C 95 02    STY    TXTMSC+1
4136 F447 A9 60      LDA    #$60
4137 F449 8D 94 02    STA    TXTMSC
4138 F44C A6 57      LDX    DINDEX
4139 F44E BD 69 FE      LDA    ANCONV,X      ; CONVERT IT TO ANTIC CODE
4140 F451 D0 04      BNE    DOPENA      ; IF ZERO, IT IS ILLEGAL

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 4

```

4141 F453 A9 91      OPNERR:  LDA  #BADMOD      ; SET ERROR STATUS
4142 F455 85 4C      STA  DSTAT
4143 F457 85 51      DOPENA:  STA  HOLD1
4144 F459 A5 6A      LDA  RAMTOP      ; SET UP AN INDIRECT POINTER
4145 F45B 85 65      STA  ADDRESS+1
4146 F45D BC 45 FE      LDY  ALOCAT,X      ; ALLOCATE N BLOCKS OF 40 BYTES
4147 F460 A9 28      DOPEN1:  LDA  #40
4148 F462 20 21 F9      JSR  DBSUB
4149 F465 88          DEY
4150 F466 D0 F8      BNE  DOPEN1
4151 F468 AD 6F 02      LDA  GPRIOR      ; CLEAR GTIA MODES
4152 F46B 29 3F      AND  11:3F
4153 F46D 85 67      STA  OPNTMP+1
4154 F46F A8          TAY
4155 F470 E0 08      CPX  #8      ; TEST IF 320X1
4156 F472 90 17      BCC  NOTE
4157 F474 BA          TXA      ; GET 2 LOW BITS
4158 F475 6A          ROR  A
4159 F476 6A          ROR  A
4160 F477 6A          ROR  A
4161 F478 29 C0      AND  #$C0      ; NOW 2 TOP BITS
4162 F47A 05 67      ORA  OPNTMP+1
4163 F47C A8          TAY
4164 F47D A9 10      LDA  #16      ; SUBTRACT 16 MORE FOR PAGE BOUNDARY
4165 F47F 20 21 F9      JSR  DBSUB
4166 F482 E0 0B      CPX  #11      ; TEST MODE 11
4167 F484 D0 05      BNE  NOT8      ; IF MODE = 11
4168 F486 A9 06      LDA  #6      ; PUT GTIA LUM VALUE INTO BACKGROUND REGISTER
4169 F488 8D C8 02      STA  COLOR4
4170 F48B 8C 6F 02      NOTE:    STY  GPRIOR      ; STORE NEW PRIORITY
4171 F48E A5 64      LDA  ADDRESS      ; SAVE MEMORY SCAN COUNTER ADDRESS
4172 F490 85 58      STA  SAVMSC
4173 F492 A5 65      LDA  ADDRESS+1
4174 F494 85 59      STA  SAVMSC+1
4175 F496 AD 0B D4      VBWAIT:  LDA  VCOUNT      ; WAIT FOR NEXT VBLANK BEFORE MESSING
4176 F499 C9 7A      CMP  #$7A      ; WITH THE DISPLAY LIST
4177 F49B D0 F9      BNE  VBWAIT
4178 F49D 20 1F F9      JSR  DBDEC      ; START PUTTING DISPLAY LIST RIGHT UNDER RAM
4179 F4A0 BD 75 FE      LDA  PAGETB,X      ; TEST IF DISPLAY LIST WILL BE IN TROUBLE

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 5

```

4180 F4A3 F0 06      BEQ  NOMOD      ; OF CROSSING A 256 BYTE PAGE BOUNDARY
4181 F4A5 A9 FF      LDA  #$FF      ; IF SO, DROP DOWN A PAGE
4182 F4A7 85 64      STA  ADDRESS
4183 F4A9 C6 65      DEC  ADDRESS+1
4184 F4AB A5 64      NOMOD: LDA  ADDRESS      ; SAVE END OF DISPLAY LIST FOR LATER
4185 F4AD 85 68      STA  SAVADR
4186 F4AF A5 65      LDA  ADDRESS+1
4187 F4B1 85 69      STA  SAVADR+1
4188 F4B3 20 13 F9    JSR  DBDDEC      ; (DOUBLE BYTE DOUBLE DECREMENT)
4189 F4B6 A9 41      LDA  #$41      ; (ANTIC) WAIT FOR VBLANK AND JMP TO TOP
4190 F4B8 20 17 F9    JSR  STORE
4191 F4BB 86 66      STX  OPNTMP
4192 F4BD A9 18      LDA  #24      ; INITIALIZE BOTSCR
4193 F4BF 8D BF 02    STA  BOTSCR
4194 F4C2 A5 57      LDA  DINDEXT      ; DISALLOW MIXED MODE IF MODE.GE.9
4195 F4C4 C9 09      CMP  #9
4196 F4C6 B0 2D      BCS  NOTMXD
4197 F4C8 A5 2A      LDA  ICAX1Z      ; TEST MIXED MODE
4198 F4CA 29 10      AND  #$10
4199 F4CC F0 27      BEQ  NOTMXD
4200 F4CE A9 04      LDA  #4
4201 F4D0 8D BF 02    STA  BOTSCR
4202 F4D3 A2 02      DOPEN2: LDX  #2      ; ADD 4 LINES OF TEXT AT BOTTOM OF SCREEN
4203 F4D5 A9 02      LDA  #2
4204 F4D7 20 17 F9    JSR  STORE
4205 F4DA CA         DEX
4206 F4DB 10 F8      BPL  DOPEN2
4207 F4DD A4 6A      LDY  RAMTOP      ;RELOAD MSC FOR TEXT
4208 F4DF 88         DEY
4209 F4E0 98         TYA
4210 F4E1 20 17 F9    JSR  STORE
4211 F4E4 A9 60      LDA  #$60
4212 F4E6 20 17 F9    JSR  STORE
4213 F4E9 A9 42      LDA  #$42
4214 F4EB 20 17 F9    JSR  STORE
4215 F4EE 18         CLC
4216 F4EF A9 0C      LDA  #MXDMDE-NUMDLE ; POINT X AT MIXED MODE TABLE
4217 F4F1 65 66      ADC  OPNTMP
4218 F4F3 85 66      STA  OPNTMP

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 6

```

4219 F4F5 A4 66      NOTMXD:  LDY  OPNTMP
4220 F4F7 BE 51 FE      LDX  NUMDLE,Y      ; GET NUMBER OF DISPLAY LIST ENTRIES
4221 F4FA A5 51      DOPEN3:  LDA  HOLD1      ; STORE N DLE'S
4222 F4FC 20 17 F9      JSR  STORE
4223 F4FF CA          DEX
4224 F500 D0 F8      BNE  DOPEN3
4225 F502 A5 57      LDA  DINDEXT      ; DO THE MESSY 320X1 PROBLEM
4226 F504 C9 08      CMP  #8
4227 F506 90 1C      BCC  DOPEN5
4228 F508 A2 5D      LDX  #93      ; GET REMAINING NUMBER OF DLE'S
4229 F50A A5 6A      LDA  RAMTOP      ; RELOAD MEMORY SCAN COUNTER
4230 F50C 38          SEC
4231 F50D E9 10      SBC  #$10
4232 F50F 20 17 F9      JSR  STORE
4233 F512 A9 00      LDA  #0
4234 F514 20 17 F9      JSR  STORE
4235 F517 A9 4F      LDA  #$4F      ; (ANTIC) RELOAD MSC CODE
4236 F519 20 17 F9      JSR  STORE
4237 F51C A5 51      DOPEN4:  LDA  HOLD1      ; DO REMAINING DLE'S
4238 F51E 20 17 F9      JSR  STORE
4239 F521 CA          DEX
4240 F522 D0 F8      BNE  DOPEN4
4241 F524 A5 59      DOPEN5:  LDA  SAVMSC+1      ; POLISH OFF DISPLAY LIST
4242 F526 20 17 F9      JSR  STORE
4243 F529 A5 58      LDA  SAVMSC
4244 F52B 20 17 F9      JSR  STORE
4245 F52E A5 51      LDA  HOLD1
4246 F530 09 40      ORA  #$40
4247 F532 20 17 F9      JSR  STORE
4248 F535 A9 70      LDA  #$70      ; 24 BLANK LINES
4249 F537 20 17 F9      JSR  STORE
4250 F53A A9 70      LDA  #$70
4251 F53C 20 17 F9      JSR  STORE
4252 F53F A5 64      LDA  ADDRESS      ; SAVE DISPLAY LIST ADDRESS
4253 F541 8D 30 02      STA  SDLSTL
4254 F544 A5 65      LDA  ADDRESS+1
4255 F546 8D 31 02      STA  SDLSTL+1
4256 F549 A9 70      LDA  #$70      ; ADD LAST BLANK LINE ENTRY
4257 F54B 20 17 F9      JSR  STORE      ; POSITION ADDRESS=SDLSTL-1

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 7

```

4258 F54E A5 64          LDA  ADDRESS      ; STORE NEW MEMTOP
4259 F550 8D E5 02       STA  MEMTOP
4260 F553 A5 65          LDA  ADDRESS+1
4261 F555 8D E6 02       STA  MEMTOP+1
4262 F558 A5 68          LDA  SAVADR
4263 F55A 85 64          STA  ADDRESS
4264 F55C A5 69          LDA  SAVADR+1
4265 F55E 85 65          STA  ADDRESS+1
4266 F560 AD 31 02       LDA  SDLSTL+1
4267 F563 20 17 F9       JSR  STORE
4268 F566 AD 30 02       LDA  SDLSTL
4269 F569 20 17 F9       JSR  STORE
4270 F56C A5 4C          LDA  DSTAT        ; IF ERROR OCCURRED ON ALLOCATION, OPEN THE EDITOR
4271 F56E 10 07         BPL  DOPEN9
4272 F570 48             PHA                ; SAVE STATUS
4273 F571 20 FC F3       JSR  EOPEN        ; OPEN THE EDITOR
4274 F574 68             PLA                ; RESTORE STATUS
4275 F575 AB            TAY                ; AND RETURN IT TO CIO
4276 F576 60            RTS
4277 F577 A5 2A          DOPEN9: LDA  ICAX1Z      ; TEST CLEAR INHIBIT BIT
4278 F579 29 20          AND  #$20
4279 F57B D0 0B          BNE  DOPEN7
4280 F57D 20 B9 F7       JSR  CLRSCR        ; CLEAR SCREEN
4281 F580 8D 90 02       STA  TXTROW        ; AND HOME TEXT CURSOR (AC IS ZERO)
4282 F583 A5 52          LDA  LMARGN
4283 F585 8D 91 02       STA  TXTCOL
4284 F588 A9 22          DOPEN7: LDA  #$22      ; EVERYTHING ELSE IS SET UP
4285 F58A 0D 2F 02       ORA  SDMCTL        ; SO TURN ON DMACTL
4286 F58D 8D 2F 02       STA  SDMCTL
4287 F590 4C 21 F6       JMP  RETUR2
4288                      ;
4289                      ;
4290 F593 20 96 FA          GETCH: JSR  RANGE      ; GETCH DOES INCRSR, GETPLT DOESN'T
4291 F596 20 A2 F5       JSR  GETPLT
4292 F599 20 32 FB       JSR  INATAC        ; CONVERT INTERNAL CODE TO ATASCII
4293 F59C 20 D4 F9       JSR  INCRSB
4294 F59F 4C 34 F6       JMP  RETUR1
4295 F5A2 20 47 F9       GETPLT: JSR  CONVRT    ; CONVERT ROW/COLUMN TO ADDRESS
4296 F5A5 B1 64          LDA  (ADDRESS),Y

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 8

```

4297 F5A7 2D A0 02      AND    DMASK
4298 F5AA 46 6F      SHIFTD:  LSR    SHFAMT      ; SHIFT DATA DOWN TO LOW BITS
4299 F5AC B0 03      BCS    SHIFT1
4300 F5AE 4A      LSR    A
4301 F5AF 10 F9      BPL    SHIFTD      ; (UNCONDITIONAL)
4302 F5B1 8D FA 02      SHIFTI: STA    CHAR
4303 F5B4 C9 00      CMP    #0      ; RESTORE FLAGS ALSO
4304 F5B6 60      RTS
4305      ;
4306      ;
4307      ;
4308 F5B7 8D FB 02      DUTCH:  STA    ATACHR
4309 F5BA 20 96 FA      JSR    RANGE
4310      ;
4311 F5BD AD FB 02      OUTCHA: LDA    ATACHR      ; TEST FOR CLEAR SCREEN
4312 F5C0 C9 7D      CMP    #CLRCOD
4313 F5C2 D0 06      BNE    OUTCHE
4314 F5C4 20 B9 F7      JSR    CLRSCR
4315 F5C7 4C 21 F6      JMP    RETUR2
4316 F5CA AD FB 02      OUTCHE: LDA    ATACHR      ; TEST FOR CARRIAGE RETURN
4317 F5CD C9 9B      CMP    #CR
4318 F5CF D0 06      BNE    OUTCHB
4319 F5D1 20 30 FA      JSR    DOCRWS      ; DO CR
4320 F5D4 4C 21 F6      JMP    RETUR2
4321 F5D7 20 ED F5      OUTCHB: JSR    OUTPLT
4322 F5DA 20 D8 F9      JSR    INCRSR
4323 F5DD 4C 21 F6      JMP    RETUR2
4324      ;
4325      ;
4326 F5E0 AD FF 02      OUTPLT: LDA    SSFLAG      ; *****LOOP HERE IF START/STOP FLAG IS NON-0
4327 F5E3 D0 FB      BNE    OUTPLT
4328 F5E5 A2 02      LDX    #2
4329 F5E7 B5 54      CRLOOP: LDA    ROWCRS,X      ; SAVE CURSOR LOCATION FOR DRAW LINE TO DRAW
4330 F5E9 95 5A      STA    OLDROW,X
4331 F5EB CA      DEX
4332 F5EC 10 F9      BPL    CRLOOP
4333 F5EE AD FB 02      LDA    ATACHR      ; CONVERT ATASCII(ATACHR) TO INTERNAL(CHAR)
4334 F5F1 A8      TAY      ; SAVE ATACHR
4335 F5F2 2A      ROL    A

```


ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 9

```

4336 F5F3 2A          ROL    A
4337 F5F4 2A          ROL    A
4338 F5F5 2A          ROL    A
4339 F5F6 29 03       AND    #3
4340 F5F8 AA          TAX
4341 F5F9 98          TYA
4342 F5FA 29 9F       AND    #$9F
4343 F5FC 1D F6 FE     ORA    ATAIN,X
4344 F5FF 8D FA 02     OUTCH2: STA  CHAR
4345 F602 20 47 F9     JSR    CONVRT
4346 F605 AD FA 02     LDA    CHAR
4347 F608 46 6F       SHIFTU: LSR    SHFAMT
4348 F60A B0 04       BCS    SHIFT2
4349 F60C 0A          ASL    A
4350 F60D 4C 08 F6     JMP    SHIFTU
4351 F610 2D A0 02     SHIFT2: AND    DMASK
4352 F613 85 50       STA    TMPCHR
4353 F615 AD A0 02     LDA    DMASK
4354 F618 49 FF       FOR    #$FF
4355 F61A 31 64       AND    (ADDRESS),Y
4356 F61C 05 50       ORA    TMPCHR
4357 F61E 91 64       STA    (ADDRESS),Y
4358 F620 60          RTS
4359
4360
4361 F621 20 A2 F5     ;
RETUR2: JSR    GETPLT
4362 F624 85 5D       STA    OLDCHR
4363 F626 A6 57       LDX    DINDE
4364 F628 D0 0A       BNE    RETUR1
4365 F62A AE F0 02     LDX    CRSINH
4366 F62D D0 05       BNE    RETUR1
4367 F62F 49 B0       EOR    #$80
4368 F631 20 FF F5     JSR    OUTCH2
4369 F634 A4 4C       RETUR1: LDY    DSTAT
4370 F636 A9 01       LDA    *SUCCES
4371 F638 85 4C       STA    DSTAT
4372 F63A AD FB 02     LDA    ATACHR
4373 F63D 60          NOFUNC: RTS
4374

```

```

4375 ;
4376 ;
4377 ; END OF DISPLAY HANDLER
4378 ;
4379 .PAGE
4380 ;
4381 ;
4382 ;
4383 ;
4384 F63E 20 83 FC EGETCH: JSR SWAP
4385 F641 20 88 FA JSR ERANGE
4386 F644 AS 6B LDA BUFCNT ; ANYTHING IN THE BUFFER?
4387 F646 D0 34 BNE EGETC3 ; YES
4388 F648 AS 54 LDA ROWCRS ; NO, SO SAVE BUFFER START ADDRESS
4389 F64A 85 6C STA BUFSTR
4390 F64C AS 55 LDA COLCRS
4391 F64E 85 6D STA BUFSTR+1
4392 F650 20 E2 F6 EGETC1: JSR KGETCH ; LET'S FILL OUR BUFFER
4393 F653 84 4C STY DSTAT ; SAVE KEYBOARD STATUS
4394 F655 AD FB 02 LDA ATACHR ; TEST FOR CR
4395 F658 C9 9B CMP #CR
4396 F65A F0 12 BEQ EGETC2
4397 F65C 20 AD F6 JSR DOSS ; NO, SO PRINT IT
4398 F65F 20 B3 FC JSR SWAP ; JSR DOSS DID SWAP SO SWAP BACK
4399 F662 A5 63 LDA LOGCOL ; BEEP IF NEARING LOGICAL COL 120
4400 F664 C9 71 CMP #113
4401 F666 D0 03 BNE EGETC6
4402 F668 20 0A F9 JSR BELL
4403 F66B 4C 50 F6 EGETC6: JMP EGETC1
4404 F66E 20 E4 FA EGETC2: JSR OFFCRS ; GET BUFFER COUNT
4405 F671 20 00 FC JSR DOBUFC
4406 F674 AS 6C LDA BUFSTR ; RETURN A CHARACTER
4407 F676 85 54 STA ROWCRS
4408 F678 AS 6D LDA BUFSTR+1
4409 F67A 85 55 STA COLCRS
4410 F67C A5 6B EGETC3: LDA BUFCNT
4411 F67E F0 11 BEQ EGETC5
4412 F680 C6 6B EGETC7: DEC BUFCNT ;AND RETURN TILL BUFCNT=0
4413 F682 F0 0D BEQ EGETC5

```

```

4414 F684 A5 4C          LDA  DSTAT          ; IF ERR, LOOP ON EGETC7 UNTIL BUFR IS EMPTY
4415 F686 30 F8          BMI  EGETC7
4416 F688 20 93 F5        JSR  GETCH
4417 F68B 8D FB 02        STA  ATACHR
4418 F68E 4C B3 FC        JMP  SWAP          ; AND RETURN WITHOUT TURNING CURSOR BACK ON
4419 F691 20 30 FA        EGETC5: JSR  DOCRWS          ; DO REAL CARRIAGE RETURN
4420 F694 A9 9B          LDA  #CR          ; AND RETURN EOL
4421 F696 8D FB 02        STA  ATACHR
4422 F699 20 21 F6        JSR  RETUR2          ; TURN ON CURSOR THEN SWAP
4423 F69C 84 4C          STY  DSTAT          ; SAVE KEYBOARD STATUS
4424 F69E 4C B3 FC        JMP  SWAP          ; AND RETURN THROUGH RETUR1
4425                      ;
4426 F6A1 6C 64 00        JSRIND: JMP  (ADRESS)          ; JSR TO THIS CAUSES JSR INDIRECT
4427                      ;
4428 F6A4 8D FB 02        EOUTCH: STA  ATACHR          ; SAVE ATASCII VALUE .
4429 F6A7 20 B3 FC        JSR  SWAP
4430 F6AA 20 88 FA        JSR  ERANGE
4431 F6AD 20 E4 FA        DOSS:   JSR  OFFCRS          ; TURN OFF CURSOR
4432 F6B0 20 8D FC        JSR  TSTCTL          ; TEST FOR CONTROL CHARACTERS (Z=1 IF CTL)
4433 F6B3 F0 09          BEQ  EOUTC5
4434 F6B5 0E A2 02        EOUTC6: ASL  ESCFLG          ; ESCFLG ONLY WORKS ONCE
4435 F6B8 20 CA F5        JSR  OUTCHE
4436 F6BB 4C B3 FC        ERETN:  JMP  SWAP          ; AND RETURN THROUGH RETUR1
4437 F6BE AD FE 02        EOUTC5: LDA  DSPFLG          ; DO DSPFLG AND ESCFLG
4438 F6C1 0D A2 02        ORA  ESCFLG
4439 F6C4 D0 EF          BNE  EOUTC6          ; IF NON-0 DISPLAY RATHER THAN EXECUTE IT
4440 F6C6 0E A2 02        ASL  ESCFLG
4441 F6C9 E8            INX          ; PROCESS CONTROL CHARACTERS
4442 F6CA BD C6 FE        LDA  CNTRL$X          ; GET DISPLACEMENT INTO ROUTINE
4443 F6CD 85 64          STA  ADRESS
4444 F6CF BD C7 FE        LDA  CNTRL$+I,X          ; GET HIGH BYTE
4445 F6D2 85 65          STA  ADRESS+1
4446 F6D4 20 A1 F6        JSR  JSRIND          ; DO COMPUTED JSR
4447 F6D7 20 21 F6        JSR  RETUR2          ; DO CURSOR
4448 F6DA 4C B3 FC        JMP  SWAP          ; ALL DONE SO RETURN THROUGH RETUR1
4449                      ;
4450                      ;
4451                      ;
4452                      ;

```

```

4453      ;      END SCREEN EDITOR.
4454      ;
4455      ;
4456      ;      BEGIN KEYBOARD HANDLER
4457      ;
4458      ;
4459      ;
4460      ;
4461  F6DD  A9 FF      KGETC2:  LDA  #$FF
4462  F6DF  8D FC 02    STA  CH
4463  F6E2  A5 2A      KGETCH:  LDA  ICAX1Z      ; TEST LSB OF AUX1 FOR SPECIAL EDITOR READ MO
4464  F6E4  4A        LSR  A
4465  F6E5  B0 62      BCS  GETOUT
4466  F6E7  A9 B0      LDA  #BRKABT
4467  F6E9  A6 11      LDX  BRKKEY      ; TEST BREAK
4468  F6EB  F0 58      BEQ  K7          ; IF BREAK PUT BRKABT IN DSTAT AND CR IN ATA
4469  F6ED  AD FC 02    LDA  CH
4470  F6F0  C9 FF      CMP  B$FF
4471  F6F2  F0 EE      BEQ  KGETCH
4472  F6F4  85 7C      STA  HOLDCH      ; SAVE CH FOR SHIFT LOCK PROC
4473  F6F6  A2 FF      LDX  #$FF      ; "CLEAR" CH
4474  F6FB  8E FC 02    STX  CH
4475  F6FB  20 D8 FC    JSR  CLICK      ; DO KEYBOARD AUDIO FEEDBACK (A IS OK)
4476  F6FE  AA        KGETC3:  TAX          ; DO ASCCON
4477  F6FF  E0 C0      CPX  #$C0      ; TEST FOR CTL & SHIFT TOGETHER
4478  F701  90 02      BCC  ASCCOI
4479  F703  A2 03      LDX  #3          ; BAD CODE
4480  F705  BD FE FE    ASCCOI:  LDA  ATASCI,X
4481  F708  BD FB 02    STA  ATACHR      ; DONE
4482  F70B  C9 80      CMP  #$80      ; DO NULLS
4483  F70D  F0 CE      BEQ  KGETC2
4484  F70F  C9 81      CMP  #$81      ; CHECK ATARI KEY
4485  F711  D0 0B      BNE  KGETCI
4486  F713  AD B6 02    LDA  INVFLG
4487  F716  49 80      FOR  #$80
4488  F718  8D B6 02    STA  INVFLG
4489  F71B  4C DD F6    JMP  KGETC2      ; DONT RETURN A VALUE
4490  F71E  C9 82      KGETCI:  CMP  #$82      ; CAPS/LOWER
4491  F720  D0 07      BNE  K1

```

```

4492 F722 A9 00          LDA    #0          ; CLEAR SHFLOK
4493 F724 8D BE 02       STA    SHFLOK
4494 F727 F0 B4          BEQ    KGETC2
4495 F729 C9 83          K1:    CMP    #$83      ; SHIFT CAPS/LOWER
4496 F72B D0 07          BNE    K2
4497 F72D A9 40          LDA    #$40
4498 F72F 8D BE 02       STA    SHFLOK      ; SHIFT BIT
4499 F732 D0 A9          BNE    KGETC2
4500 F734 C9 84          K2:    CMP    #$84      ; CNTL CAPS/LOWER
4501 F736 D0 07          BNE    K3
4502 F738 A9 80          LDA    #$80      ; CNTL BIT
4503 F73A 8D BE 02       STA    SHFLOK
4504 F73D D0 9E          BNE    KGETC2
4505 F73F C9 85          K3:    CMP    #$85      ; DO EOF
4506 F741 D0 0A          BNE    K6
4507 F743 A9 88          LDA    #EOFERR
4508 F745 85 4C          K7:    STA    DSTAT
4509 F747 85 11          STA    BRKKEY      ; RESTORE BREAK
4510 F749 A9 9B          GETOUT: LDA    #CR      ; PUT CR IN ATACHR
4511 F74B D0 26          BNE    K8      ; (UNCONDITIONAL)
4512 F74D A5 7C          K6:    LDA    HOLDCH    ; PROCESS SHIFT LOCKS
4513 F74F C9 40          CMP    #$40      ; REGULAR SHIFT AND CONTROL TAKE PRECEDENCE
4514 F751 B0 15          BCS    K5      ; OVER LOCK
4515 F753 AD FB 02       LDA    ATACHR      ; TEST FOR ALPHA
4516 F756 C9 61          CMP    #$61      ; LOWER CASE A
4517 F758 90 0E          BCC    K5      ; NOT ALPHA IF LT
4518 F75A C9 7B          CMP    #$7B      ; LOWER CASE Z+1
4519 F75C B0 0A          BCS    K5      ; NOT ALPHA IF GE
4520 F75E AD BE 02       LDA    SHFLOK      ; DO SHIFT/CONTROL LOCK
4521 F761 F0 05          BEQ    K5      ; IF NO LOCK DONT RE-DO IT
4522 F763 05 7C          ORA    HOLDCH
4523 F765 4C FE F6       JMP    KGETC3      ; DO RETRY
4524 F768 20 8D FC       K5:    JSR    TSTCTL    ; DONT INVERT MSB OF CONTROL CHARACTERS
4525 F76B F0 09          BEQ    K4
4526 F76D AD FB 02       LDA    ATACHR
4527 F770 4D B6 02       FOR    INVFLG
4528 F773 8D FB 02       K8:    STA    ATACHR
4529 F776 4C 34 F6       K4:    JMP    RETUR1    ; ALL DONE
4530                      ;

```

```

4531 ;
4532 .PAGE
4533 ;
4534 ;
4535 ; CONTROL CHARACTER PROCESSORS
4536 ;
4537 F779 A9 80 ESCAPE: LDA #$80 ; SET ESCAPE FLAG
4538 F77B SD A2 02 STA ESCFLG
4539 F77E 60 RTS
4540 F77F C6 54 CRSRUP: DEC ROWCRS
4541 F781 10 06 BPL COMRET
4542 F783 AE BF 02 LDX BOTSCR ; WRAPAROUND
4543 F786 CA DEX
4544 F787 86 54 UPDNCM: STX ROWCRS
4545 F789 4C 5C FC COMRET: JMP STRBEG ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4546 F78C E6 54 CRSRDN: INC ROWCRS
4547 F78E A5 54 LDA ROWCRS
4548 F790 CD BF 02 CMP BOTSCR
4549 F793 90 F4 BCC COMRET
4550 F795 A2 00 LDX #0
4551 F797 F0 EE BEQ UPDNCM ; (UNCONDITIONAL)
4552 F799 C6 55 CRSRLF: DEC COLCRS
4553 F79B A5 55 LDA COLCRS
4554 F79D 30 04 BMI CRSRLI ; (IF LMARGN=0. THIS ELIMINATES PROBLEM CASE)
4555 F79F C5 52 CMP LMARGN
4556 F7A1 80 04 BCS COMRE1
4557 F7A3 A5 53 CRSRLI: LDA RMARGN
4558 F7A5 85 55 LFRTCM: STA COLCRS
4559 F7A7 4C DD FB COMRE1: JMP DOLCOL ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4560 F7AA E6 55 CRSRRT: INC COLCRS
4561 F7AC A5 55 LDA COLCRS
4562 F7AE C5 53 CMP RMARGN
4563 F7B0 90 F5 BCC COMRE1
4564 F7B2 F0 F3 BEQ COMRE1 ; (CAUSE BLE)
4565 F7B4 A5 52 LDA LMARGN
4566 F7B6 4C A5 F7 JMP LFRTCM ; UNCONDITIONAL TO COMMON STORE
4567 F7B9 20 F3 FC CLRSCR: JSR PUTMSC
4568 F7BC A0 00 LDY #0
4569 F7BE 98 TYA ; PUT 0 IN THE AC

```

```

4570 F7BF 91 64          CLRSC2:  STA  (ADRESS),Y  ; (AC IS ZERO)
4571 F7C1 C8            INY
4572 F7C2 D0 FB        BNE  CLRSC2
4573 F7C4 E6 65        INC  ADRESS+1
4574 F7C6 A6 65        LDX  ADRESS+1
4575 F7CS E4 6A        CPX  RAMTOP
4576 F7CA 90 F3        BCC  CLRSC2
4577 F7CC A9 FF        LDA  #$FF          ; CLEAN UP LOGICAL LINE BIT MAP
4578 F7CE 99 B2 02      CLRSC3:  STA  LOGMAP,Y    ; (Y IS ZERO AFTER CLRSC2 LOOP)
4579 F7D1 CS           INY
4580 F7D2 C0 04        CPY  #4
4581 F7D4 90 F8        BCC  CLRSC3
4582 F7D6 20 E4 FC      HOME:    JSR  COLCR      ; PLACE COLCRS AT LEFT EDGE
4583 F7D9 85 63        STA  LOGCOL
4584 F7DB 85 6D        STA  BUFSTR+1
4585 F7DD A9 00        LDA  #0
4586 F7DF 85 54        STA  ROWCRS
4587 F7E1 85 56        STA  COLCRS+1
4588 F7E3 85 6C        STA  BUFSTR
4589 F7E5 60           RTS
4590                   ;
4591 F7E6 A5 63        BS:       LDA  LOGCOL      ; BACKSPACE
4592 F7ES C5 52        CMP  LMARGN
4593 F7EA F0 21        BEQ  BS1
4594 F7EC A5 55        BSA:     LDA  COLCRS      ; LEFT EDGE?
4595 F7EE C5 52        CMP  LMARGN
4596 F7F0 D0 03        BNE  BS3          ; NO
4597 F7F2 20 73 FC      JSR  DELTIM          ; YES. SEE IF LINE SHOULD BE DELETED
4598 F7F5 20 99 F7      BS3:    JSR  CRSRLF
4599 F7F8 A5 55        LDA  COLCRS
4600 F7FA C5 53        CMP  RMARGN
4601 F7FC D0 07        BNE  BS2
4602 F7FE A5 54        LDA  ROWCRS
4603 800 F0 03        BEQ  BS2
4604 F802 20 7F F7      JSR  CRSRUP
4605 FB05 A9 20        BS2:     LDA  #$20      ; MAKE BACKSPACE DESTRUCTIVE
4606 F807 8D FB 02      STA  ATACHR
4607 FB0A 20 E0 F5      JSR  OUTPLT
4608 FB0D 4C DD FB      BSI:    JMP  DOLCOL    ; AND RETURN

```

```

4609 F810 20 AA F7      TAB:      JSR      CRSRRT      ; BEGIN SEARCH
4610 F813 A5 55          LDA      COLCRS      ; TEST FOR NEW LINE
4611 F815 C5 52          CMP      LMARGN
4612 F817 D0 0A          BNE      TAB1          ; NO
4613 F819 20 34 FA      JSR      DOCR      ; DO CARRIAGE RETURN
4614 FB1C 20 20 FB      JSR      LOGGET      ; CHECK IF END OF LOGICAL LINE
4615 F81F 90 02          BCC      TAB1          ; NO, CONTINUE
4616 F821 B0 07          BCS      TAB2          ; (UNCONDITIONAL)
4617 F823 A5 63      TAB1:      LDA      LOGCOL      ; CHECK FOR TAB STOP
4618 F825 20 25 FB      JSR      BITGET
4619 FB28 90 E6          BCC      TAB          ; NO, SO KEEP LOOKING
4620 F82A 4C DD FB      TAB2:      JMP      DOLCOL      ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4621 F82D A5 63      SETTAB:     LDA      LOGCOL
4622 F82F 4C 06 FB      JMP      BITSET      ; SET BIT IN MAP AND RETURN
4623 F832 A5 63      CLR TAB:     LDA      LOGCOL
4624 F834 4C 12 FB      JMP      BITCLR      ; CLEAR
4625 F837 20 9D FC      INSCHR:    JSR      PHACRS
4626 F83A 20 A2 F5      JSR      GETPLT      ; GET CHARACTER UNDER CURSOR
4627 F83D 85 7D          STA      INSDAT
4628 F83F A9 00          LDA      #0
4629 F841 8D BB 02      STA      SCRFLG
4630 F844 20 FF F5      INSCH4:    JSR      OUTCH2      ; STORE DATA
4631 F847 A5 63          LDA      LOGCOL      ; SAVE LOGCOL: IF AFTER INCRSA LOGCOL IS
4632 F849 48          PHA          ; < THAN IT IS NOW, END LOOP
4633 F84A 20 DC F9      JSR      INCRSA      ; SPECIAL INCRSR ENTRY POINT
4634 F84D 68          PLA
4635 F84E C5 63          CMP      LOGCOL
4636 F850 B0 0C          BCS      INSCH3      ; QUIT
4637 F852 A5 7D      INSCHI:     LDA      INSDAT      ; KEEP GOING
4638 F854 48          PHA
4639 F855 20 A2 F5      JSR      GETPLT
4640 F858 85 7D          STA      INSDAT
4641 F85A 68          PLA
4642 F85B 4C 44 F8      JMP      INSCH4
4643 F85E 20 A8 FC      INSCH3:    JSR      PLACRS
4644 FB61 CE BB 02      INSCH6:     DEC      SCRFLG
4645 F864 30 04          BMI      INSCH5      ; IF SCROLL OCCURRED
4646 F866 C6 54          DEC      ROWCRS      ; MOVE CURSOR UP
4647 F868 D0 F7          BNE      INSCH6      ; (UNCOND) CONTINUE UNTIL SCRFLG IS MINUS

```



```

4648 F86A 4C DD FB      INSCH5:  JMP  DOLCOL      ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4649                      ;
4650                      ;
4651 F86D 20 9D FC      DELCHR:  JSR  PHACRS
4652 F870 20 47 F9      DELCHI:  JSR  CONVRT      ; GET DATA TO THE RIGHT OF THE CURSOR
4653 F873 A5 64          LDA  ADDRESS
4654 F875 85 68          STA  SAVADR      ;SAVE ADDRESS TO KNOW WHERE TO PUT DATA
4655 F877 A5 65          LDA  ADDRESS+1
4656 F879 85 69          STA  SAVADR+1
4657 F87B A5 63          LDA  LOGCOL
4658 F87D 48            PHA
4659 F87E 20 D4 F9      JSR  INCRSB      ; PUT CURSOR OVER NEXT CHARACTER
4660 F881 68            PLA
4661 F882 C5 63          CMP  LOGCOL      ;TEST NEW LOGCOL AGAINST OLD LOGCOL
4662 F884 B0 10          BCS  DELCH2      ; IF OLD. GE. NEW THEN QUIT
4663 F886 A5 54          LDA  ROWCRS      ; IS ROW OFF SCREEN?
4664 F888 CD BF 02      CMP  BOTSCR
4665 F88B B0 09          BCS  DELCH2      ; YES, SO QUIT
4666 F88D 20 A2 F5      JSR  GETPLT      ; GET DATA UNDER CURSOR
4667 F890 A0 00          LDY  #0
4668 F892 91 68          STA  (SAVADR),Y  ; PUT IT IN PREVIOUS POSITION
4669 F894 F0 DA          BEQ  DELCHI      ; AND LOOP (UNCONDITIONAL)
4670 F896 A0 00          DELCH2:  LDY  #0
4671 F898 98            TYA
4672 F899 91 68          STA  (SAVADR),Y  CLEAR THE LAST POSITION
4673 F89B 20 68 FC      JSR  DELTIA      ; TRY TO DELETE A LINE
4674 F89E 20 A8 FC      JSR  PLACRS
4675 FBA1 4C DD FB      JMP  DOLCOL      ; AND RETURN
4676 F8A4 38            INSLIN:  SEC      ; NORMAL INSLIN PUTS "1" INTO BIT MAP
4677 F8A5 20 7B FB      INSLIA:  JSR  EXTEND      ; ENTRY POINT FOR C=0
4678 FSA8 A5 52          LDA  LMARGN      ; DO CARRIAGE RETURN (NO LF)
4679 FBAA 85 55          STA  COLCRS
4680 FBAC 20 47 F9      JSR  CONVRT      ; GET ADDRESS
4681 FBAF A5 64          LDA  ADDRESS      ; SET UP TO=40+FROM (FROM = CURSOR)
4682 FSB1 85 68          STA  FRMADR
4683 F8B3 18            CLC
4684 FBB4 69 28          ADC  #40
4685 FBB6 85 66          STA  TOADR
4686 F8BS A5 65          LDA  ADDRESS+1

```

```

4687 FBBA 85 69          STA  FRMADR+1
4688 FBBC 69 00          ADC  #0
4689 FBBE 85 67          STA  TOADR+1
4690 FSC0 A6 54          LDX  ROWCRS      ;SET UP LOOP COUNTER
4691 FSC2 E0 17          CPX  #23
4692 FBC4 F0 0B          BEQ  INSLI2
4693 FBC6 20 4E FB      INSLI1: JSR  MOVLIN
4694 FSC9 E8              INX
4695 F8CA E0 17          CPX  #23
4696 FBCC D0 F8          BNE  INSLI1
4697 FBCE 20 9B FB      INSLI2: JSR  CLRLIN      ; CLEAR CURRENT LINE
4698 FBD1 4C DD FB          JMP  DOLCOL      ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4699 FBD4 20 DD FB      DELLIN: JSR  DOLCOL      ; GET BEGINNING OF LOG LINE (HOLDD
4700 F8D7 A4 51      DELLIA: LDY  HOLD1      ; SQUEEZE BIT MAP
4701 FSD9 84 54          STY  ROWCRS      ; PUT CURSOR THERE
4702 FBDB A4 54      DELLIB: LDY  ROWCRS
4703 FBDD 98      DELLII: TYA
4704 F8DE 38          SEC
4705 FBDF 20 23 FB      JSR  L02GET      ; GET NEXT BIT
4706 FSE2 08          PHP
4707 FBE3 98          TYA
4708 FBE4 18          CLC
4709 FBE5 69 78          ADC  #120
4710 FSE7 28          PLP
4711 FeE8 20 04 FB      JSR  BITPUT      ; WRITE IT OVER PRESENT BIT
4712 FBEB C8          INY
4713 FBEC C0 18          CPY  #24
4714 FREE D0 ED          BNE  DELLI1      ; LOOP
4715 FBFO AD B4 02      LDA  LOGMAP+2      ; SET LSB
4716 FSF3 09 01          ORA  #1
4717 FBF5 BD B4 02      STA  LOGMAP+2
4718 F8F8 A5 52      DELLI2: LDA  LMARGN      ; DELETE LINE OF DATA USING PART OF SCROLL
4719 FBFA 85 55          STA  COLCRS      ; CR NO LF
4720 FBFC 20 47 F9      JSR  CONVRT
4721 FBFF 20 B7 FB      JSR  SCROLL
4722 F902 20 20 FB      JSR  LOGGET      ; TEST NEXT LINE FOR CONTINUATION
4723          ; IS IT A NEW LOG LINE?
4724 F905 90 D4          BCC  DELLIB      ; NO SO DELETE ANOTHER
4725 F907 4C DD FB      JMP  DOLCOL      ; YES SO DOLCOL AND RETURN

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 19

```

4726 F90A A0 20      BELL:      LDY      #$20
4727 F90C 20 D8 FC    BELL1:      JSR      CLICK
4728 F90F 88          DEY
4729 F910 10 FA          BPL      BELL1
4730 F912 60          RTS
4731          .PAGE
4732          ;
4733          ;
4734          ; ROUTINES
4735          ;
4736          ;
4737          ; DOUBLE BYTE DECREMENT OF      INDIRECT POINTER
4738          ; INCLUDING DB      SUBTRACT AND      DB DOUBLE DECREMENT
4739          ;
4740 F913 A9 02      DBDDEC:      LDA      #2
4741 F915 D0 0A          BNE      DBSUB      ; (UNCONDITIONAL)
4742          ;
4743          ; STORE DATA INDIRECT AND DECREMENT POINTER
4744          ; (PLACED HERE      TO SAVE JMP DBDEC AFTER STORE)
4745 F917 A4 4C      STORE:      LDY      DSTAT      ; RETURN ON ERROR
4746 F919 30 2B          BMI      STOK
4747 F91B A0 00          LDY      #0
4748 F91D 91 64      STORE1:      STA      (ADDRESS),Y
4749          JMP      DBDEC      ; DECREMENT AND RETURN
4750          ;
4751 F91F A9 01      DBDEC:      LDA      #1
4752 F921 8D 9E 02    DBSUB:      STA      SUBTMP
4753 F924 A5 4C          LDA      DSTAT      ; RETURN ON ERROR
4754 F926 30 1E          BMI      STOK
4755 F928 A5 64          LDA      ADDRESS
4756 F92A 38          SEC
4757 F92B ED 9E 02    SBC      SUBTMP
4758 F92E 85 64          STA      ADDRESS
4759 F930 B0 02          BCS      DBSUB1
4760 F932 C6 65          DEC      ADDRESS+1
4761 F934 A5 0F      DBSUB1:      LDA      APPMHI+1      ; MAKE SURE NOTHING EVER OVERWRITES APPMHI
4762 F936 C5 65          CMP      ADDRESS+1
4763 F938 90 0C          BCC      STOK      ; OK
4764 F93A D0 06          BNE      STRERR      ; ERROR

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 20

```

4765 F93C A5 0E          LDA  APPMHI
4766 F93E C5 64          CMP  ADDRESS
4767 F940 90 04          BCC  STROK
4768 F942 A9 93          STRERR: LDA  #SCRMEM      ; SHOW MEM TOO SMALL FOR SCREEN ERROR
4769 F944 85 4C          STA  DSTAT
4770 F946 60          STROK:  RTS
4771                      ;
4772                      ;
4773                      ;
4774                      ; CONVERT ROW/COLUMN CURSOR INTO REAL ADDRESS (FROM SAVMSC ON UP)
4775
4776 F947 A5 54          CONVRT: LDA  ROWCRS      ; SAVE CURSOR
4777 F949 48          PHA
4778 F94A A5 55          LDA  COLCRS
4779 F94C 48          PHA
4780 F94D A5 56          LDA  COLCRS+1
4781 F94F 48          PHA
4782 F950 20 F3 FC      JSR  PUTMSC
4783 F953 A5 54          LDA  ROWCRS      ; PUT 10#ROWCRS INTO MLTTMP
4784 F955 85 66          STA  MLTTMP
4785 F957 A9 00          LDA  #0
4786 F959 85 67          STA  MLTTMP+1
4787 F95B A5 66          LDA  MLTTMP      ; QUICK X8
4788 F95D 0A          ASL  A
4789 F95E 26 67          ROL  MLTTMP+1
4790 F960 85 51          STA  HOLD1      ; (SAVE 2X VALUE)
4791 F962 A4 67          LDY  MLTTMP+1    ; ""
4792 F964 8C 9F 02      STY  HOLD2      ; ""
4793 F967 0A          ASL  A
4794 F968 26 67          ROL  MLTTMP+1
4795 F96A 0A          ASL  A
4796 F96B 26 67          ROL  MLTTMP+1
4797 F96D 18          CLC                      ; ADD IN 2X
4798 F96E 65 51          ADC  HOLD1
4799 F970 85 66          STA  MLTTMP
4800 F972 A5 67          LDA  MLTTMP+1
4801 F974 6D 9F 02      ADC  HOLD2
4802 F977 85 67          STA  MLTTMP+1
4803 F979 A6 57          LDX  DINDEX      ; NOW SHIFT MLTTMP LEFT DHLIN TIMES TO FINIS

```

```

4804 F97B BC 81 FE          LDY    DHLINE,X      ; MULTIPLY
4805 F97E 88                CONVRI:  DEY          ; LOOP N TIMES
4806 F97F 30 07              BMI     CONVR2
4807 F981 06 66              ASL     MLTTMP
4808 F983 26 67              ROL     MLTTMP+1
4809 F985 4C 7E F9          JMP     CONVRI
4810 F988 BC A5 FE          CONVR2:  LDY    DIV2TB,X    ; NOW DIVIDE HCRSR TO ACCOUNT FOR PARTIAL BYT
4811 F98B A5 55              LDA     COLCRS
4812 F98D A2 07              LDX     #7            ; TRICKY
4813 F98F 88                CONVR3:  DEY
4814 F990 30 0A              BMI     CONVR4
4815 F992 CA                DEX
4816 F993 46 56              LSR     COLCRS+1
4817 F995 6A                ROR     A
4818 F996 6E A1 02          ROR     TEMPLBT      ; SAVE LOW BITS FOR MASK
4819 F999 4C 8F F9          JMP     CONVR3
4820 F99C C8                CONVR4:  INY          ; SO Y IS ZERO UPON RETURN FROM THIS ROUTINE
4821 F99D 18                CLC
4822 F99E 65 66              ADC     MLTTMP      ; ADD SHIFTED COLCRS TO MLTTMP
4823 F9A0 85 66              STA     MLTTMP
4824 F9A2 90 02              BCC     CONVR5
4825 F9A4 E6 67              INC     MLTTMP+1
4826 F9A6 38                CONVR5:  SEC          ; * TRICKY
4827 F9A7 6E A1 02          CONVR6:  ROR     TEMPLBT      ; SLIDE A "1" UP AGAINST LOW BITS (CONTINUE T
4828 F9AA 18                CLC
4829 F9AB CA                DEX          ; AND FINISH SHIFT SO LOW BITS ARE
4830 F9AC 10 F9              BPL     CONVR6      ; RIGHT JUSTIFIED.
4831 F9AE AE A1 02          LDX     TEMPLBT      ; TEMPLBT IS NOW THE INDEX INTO DMASKTB
4832 F9B1 A5 66              LDA     MLTTMP      ; PREPARE FOR RETURN
4833 F9B3 18                CLC
4834 F9B4 65 64              ADC     ADRESS
4835 F9B6 85 64              STA     ADRESS
4836 F9B8 85 5E              STA     OLDADR      ; REMEMBER THIS ADDRESS FOR CURSOR
4837 F9BA A5 67              LDA     MLTTMP+1
4838 F9BC 65 65              ADC     ADRESS+1
4839 F9BE 85 65              STA     ADRESS+1
4840 F9C0 85 5F              STA     OLDADR+1
4841 F9C2 BD B1 FE          LDA     DMASKT,X
4842 F9C5 8D A0 02          STA     DMASK

```

```

4843 F9C8 85 6F          STA  SHFAMT
4844 F9CA 68             PLA
4845 F9CB 85 56          STA  COLCRS+1
4846 F9CD 68             PLA
4847 F9CE 85 55          STA  COLCRS
4848 F9D0 68             PLA
4849 F9D1 85 54          STA  ROWCRS
4850 F9D3 60             RTS
4851                      ;
4852                      ;
4853                      ; INCREMENT CURSOR AND DETECT BOTH END OF LINE AND END      OF SCREEN
4854                      ;
4855 F9D4 A9 00          INCRSB: LDA  It0          ; NON-EXTEND ENTRY POINT
4856 F9D6 F0 02          BEQ  INCRSC
4857 F9D8 A9 9B          INCRSR: LDA  #498          ; SPECIAL CASE ELIMINATOR
4858 F9DA 85 7D          INCRSC: STA  INSDAT
4859 F9DC E6 63          INCRSA: INC  LOGCOL          ; (INSCHR ENTRY POINT)
4860 F9DE E6 55          INC  COLCRS
4861 F9E0 D0 02          BNE  INCRS2          ; DO HIGH BYTE
4862 F9E2 E6 56          INC  COLCRS+1
4863 F9E4 A5 55          INCRS2: LDA  COLCRS          ; TEST END OF LINE
4864 F9E6 A6 57          LDX  DINDEX
4865 F9EB DD BD FE      CMP  COLUMN,X          ; TEST TABLED VALUE FOR ALL    SCREEN MODES
4866 F9EB F0 0B          BEQ  INC2A          ; DO CR IF EQUAL
4867 F9ED E0 00          CPX  #0          ; MODE 0?
4868 F9EF D0 06          BNE  INCRS3          ; IF NOT, JUST RETURN
4869 F9F1 C5 53          CMP  RMARGN          ; TEST AGAINST RMARGN
4870 F9F3 F0 02          BEQ  INCRS3          ; EQUAL IS OK
4871 F9F5 B0 01          BCS  INC2A          ; IF GREATER THAN DO CR
4872 F9F7 60          INCRS3: RTS
4873 F9F8 E0 08          INC2A: CPX  #8          ; CHECK MODE
4874 F9FA 90 04          BCC  DOCR1          ; NOT 320X1 SO DO IT
4875 F9FC A5 56          LDA  COLCRS+1          ; TEST MSD
4876 F9FE F0 F7          BEQ  INCRS3          ; ONLY AT 64 SO DON'T DO IT
4877 FA00 A5 57          DOCR1: LDA  DINDEX          ; DON'T MESS WITH LOGMAP IF NO MODE ZERO
4878 FA02 D0 30          BNE  DOCR
4879 FA04 A5 63          LDA  LOGCOL          ; TEST LINE OVERRUN
4880 FA06 C9 51          CMP  #81
4881 FA08 90 0A          BCC  DOCRIB          ; IF LESS THAN 81 IT IS DEFINITELY NOT LINE 3

```

```

4882 FA0A A5 7D LDA INSDAT
4883 FA0C F0 26 BEQ DOCR ; ONLY DO LOG LINE OVERFLOW IF INSDAT <>0
4884 FADE 20 30 FA JSR DOCRWS ; LOG LINE OVERFLOW IS SPECIAL CASE
4885 FA11 4C 77 FA JMP INCRSI ; RETURN
4886 FA14 20 34 FA DOCRIB: JSR DOCR ; GET IT OVER WITH
4887 FA17 A5 54 LDA ROWCRS
4888 FA19 18 CLC ; TEST LOGICAL LINE BIT MAP
4889 FA1A 69 78 ADC #120
4890 FA1C 20 25 FB JSR BITGET
4891 FA1F 90 08 BCC DOCRIA ; DON'T EXTEND IF OVERRUN IS INTO MIDDLE OF L
4892 FA21 A5 7D LDA INSDAT ; DON'T EXTEND IF INSDAT IS ZERO
4893 FA23 F0 04 BEQ DOCRIA ; (INSCHR SPECIAL CASE)
4894 FA25 18 CLC ; INSERT "0" INTO BIT MAP
4895 FA26 20 A5 FB JSR INSLIA
4896 FA29 4C DD FB DOCRIA: JMP DOLCOL ; CONVERT ROW AND COL TO LOGCOL AND RETURN
4897 FA2C A9 00 NOSCR1: LDA #0 ; DOCR WITHOUT SCROLL
4898 FA2E F0 02 BEQ NOSCR1 ; (UNCONDITIONAL)
4899 FA30 A9 9B DOCRWS: LDA #$98 ; DOCR WITH SCROLLING (NORMAL MODE)
4900 FA32 85 7D NOSCR1: STA INSDAT
4901 FA34 20 E4 FC DOCR: JSR COLCR ; PLACE COLCRS AT LEFT EDGE
4902 FA37 A9 00 LDA #0
4903 FA39 85 56 STA COLCRS+1
4904 FA3B E6 54 INC ROWCRS
4905 FA3D A6 57 DOCR2: LDX DINDEX
4906 FA3F A0 18 LDY #24 ; SET UP SCROLL LOOP COUNTER
4907 FA41 24 78 BIT SWPFLG
4908 FA43 10 05 BPL DOCR2A ; BRANCH IF NORMAL
4909 FA45 A0 04 LDY #4
4910 FA47 98 TYA
4911 FA48 D0 03 BNE DOCR2B ; (UNCONDITIONAL)
4912 FA4A BD 99 FE DOCR2A: LDA NOROWS,X ; GET NO OF ROWS
4913 FA4D C5 54 DOCR2B: CMP ROWCRS
4914 FA4F D0 26 BNE INCRSI
4915 FA51 8C 9D 02 STY HOLDS
4916 FA54 8A TXA ; DON'T SCROLL IF MODE <> 0
4917 FA55 D0 20 BNE INCRS1
4918 FA57 A5 7D LDA INSDAT ; OR IF INSDAT = 0
4919 FA59 F0 1C BEQ INCRS1
4920 LDA INSDAT ; IF INSDAT <> $9B THEN ROLL IN A 0

```

```

4921 FA5B C9 9B          CMP    #$9B          ; TO EXTEND BOTTOM LOGICAL LINE
4922 FA5D 38            SEC
4923 FA5E F0 01          BEQ    DOCR4B
4924 FA60 18            CLC
4925 FA61 20 AC FB      DOCR4B: JSR    SCROLL      ; LOOP BACK TO HERE IF >1 SCROLLS
4926 FA64 EE BB 02      INC    SCRFLG
4927 FA67 C6 6C          DEC    BUFSTR      ; ROWS MOVE UP SO BUFSTR SHOULD T00
4928 FA69 CE 9D 02      DEC    HOLDS
4929 FA6C AD B2 02      LDA    LOGMAP
4930 FA6F 38            SEC                  ; FOR PARTIAL LINES, ROLL IN A "1"
4931 FA70 10 EF          BPL    DOCR4B      ; AGAIN IF PARTIAL LOGICAL LINE
4932 FA72 AD 9D 02      LDA    HOLDS      ; PLACE CURSOR AT NEW LINE NEAR THE BOTTOM
4933 FA75 85 54          STA    ROWCRS
4934 FA77 4C DD FB      INCRS1: JMP    DOLCOL      ; COLVERT ROW AND COL TO LOGCOL AND RETURN
4935                      ;
4936                      ;
4937                      ; SUBEND: SUBTRACT ENDPT      FROM ROWAC OR COLAC. (X=0 OR 2)
4938                      ;
4939 FA7A 38            SUBEND: SEC
4940 FA7B 85 70          LDA    ROWAC,X
4941 FA7D E5 74          SBC    ENDPT
4942 FA7F 95 70          STA    ROWAC,X
4943 FA81 B5 71          LDA    ROWAC+1,X
4944 FA83 E5 75          SBC    ENDPT+1
4945 FA85 95 71          STA    ROWAC+1,X
4946 FA87 60            RTS
4947                      ;
4948                      ;
4949                      ; RANGE: DO CURS R RANGE TEST. IF ERROR POP STACK TWICE AND JMP RETURN
4950                      ; (ERANGE IS EDITOR ENTRY POINT AND TEST IF EDITOR IS OPEN.
4951                      ; IF IT ISNT IT OPENS THE EDITOR AND CONTINUES)
4952                      ;
4953 FA88 AD BF 02      ERANGE: LDA    BOTSCR      ; IF BOTSCR=4
4954 FABB C9 04          CMP    #4
4955 FABD F0 07          BEQ    RANGE      ; THEN IT IS IN MIXED MODE AND OK
4956 FABF A5 57          LDA    DINDEXT      ; IF MODE = 0
4957 FA91 F0 03          BEQ    RANGE      ; THEN IT IS IN EDITOR MODE AND OK
4958 FA93 20 FC F3      JSR    EOPEN      ; IF NOT, OPEN EDITOR
4959 FA96 A9 27      RANGE: LDA    #39      ; *** RANGE CHECK RMARGN *** SET UP AC

```



```

4960 FA98 C5 53          CMP    RMARGN      ; *** RANGE CHECK RMARGN *** COMPARE
4961 FA9A B0 02          BCS    RANGE3      ; *** RANGE CHECK RMARGN *** BRANCH GE
4962 FA9C 85 53          STA    RMARGN      ; *** RANGE CHECK RMARGN *** BAD SO STORE
4963 FA9E A6 57          RANGE3: LDX    DINDEX
4964 FAA0 BD 99 FE          LDA    NOROWS,X  ; CHECK ROWS
4965 FAA3 C5 54          CMP    ROWCRS
4966 FAA5 90 2A          BCC    RNGERR      ; (ERROR IF TABLE. GE.ROWCRS)
4967 FAA7 F0 28          BEQ    RNGERR
4968 FAA9 E0 08          CPX    #8          ; CHECK FOR 320X1
4969 FAAB D0 0A          BNE    RANGE1      ; SPECIAL CASE IT
4970 FAAD A5 56          LDA    COLCRS+1
4971 FAAF F0 13          BEQ    RNGOK      ; IF HIGH BYTE IS 0, COL IS OK
4972 FAB1 C9 01          CMP    #1
4973 FAB3 D0 1C          BNE    RNGERR      ; IF >1, BAD
4974 FAB5 F0 04          BEQ    RANGE2      ; IF 1, GO CHECK LOW BYTE
4975 FAB7 A5 56          RANGE1: LDA    COLCRS+1 ; FOR OTHERS, NON-ZERO HIGH BYTE IS BAD
4976 FAB9 D0 16          BNE    RNGERR
4977 FABB BD 8D FE          RANGE2: LDA    COLUMN,X ; CHECK LOW BYTE
4978 FABE C5 55          CMP    COLCRS
4979 FAC0 90 0F          BCC    RNGERR
4980 FAC2 F0 0D          BEQ    RNGERR
4981 FAC4 A9 01          RNGOK:  LDA    #SUCCES ; SET STATUS OK
4982 FACE 85 4C          STA    DSTAT
4983 FAC8 A9 80          LDA    #BRKABT      ; PREPARE BREAK ABORT STATUS
4984 FACA A6 11          LDX    BRKKEY      ; CHECK BREAK KEY FLAG
4985 FACC 85 11          STA    BRKKEY      ; 'CLEAR' BREAK
4986 FACE F0 06          BEQ    RNGER2      ; IF BREAK, QUIT IMMEDIATELY AND RETURN TO CIO
4987 FAD0 60             RTS
4988 FAD1 20 D6 F7          RNGERR: JSR    HOME ; ON RANGE ERROR, BRING CURSOR BACK
4989 FAD4 A9 8D          LDA    #CRSROR      ; SHOW CURSOR OVERRANGE ERROR
4990 FAD6 85 4C          RNGER2: STA    DSTAT
4991 FADS 68             RNGERI: PLA          ; RESTORE STACK (THIS ROUTINE IS ALWAYS 1 LEV
4992 FAD9 68             PLA          ; AWAY FROM RETURN TO CIO)
4993 FADA A5 7B          LDA    SWPFLG      ; IF SWAPPED, SWAP BACK
4994 FADC 10 03          BPL    RETUR3
4995 FADE 20 B9 FC          JSR    SWAPA      ; AND DONT DO RETUR1
4996 FAE1 4C 34 F6          RETUR3: JMP    RETUR1 ; RETURN TO CIO
4997                      ;
4998                      ;

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 26

```

4999      ;
5000      ; OFFCRS: RESTOR  E OLD DATA  UNDER CURSOR SO IT CAN BE MOVED
5001      ;
5002  FAE4  A0 00      OFFCRS:      LDY      #0
5003  FAE6  A5 5D      LDA      OLDCHR
5004  FAE8  91 5E      STA      (OLDADR),Y
5005  FAEA  60          RTS
5006      ;
5007      ;
5008      ;
5009      ; BITMAP ROUTINES:
5010      ;
5011      ; BITCON: PUT MASK IN BITMSK AND INDEX IN X
5012      ; BITPUT: PUT CARRY INTO BITMAP
5013      ; BITROL: ROL CARRY INTO BOTTOM OF BITMAP (SCROLL)
5014      ; BITSET: SET PROPER BIT
5015      ; BITCLR: CLEAR PROPER BIT
5016      ; BITGET: RETURN CARRY SET IF BIT IS THERE
5017      ; LOGGET: DO BITGET FOR LOGMAP INSTEAD OF TABMAP
5018      ;
5019  FAEB  48          BITCON:      PHA
5020  FAEC  29 07      AND      117
5021  FAEE  AA          TAX      ;GET MASK
5022  FAEF  BD B9 FE      LDA      MASKTB,X
5023  FAF2  85 6E      STA      BITMSK
5024  FAF4  68          PLA          ; PROCESS INDEX
5025  FAF5  4A          LSR      A
5026  FAF6  4A          LSR      A
5027  FAF7  4A          LSR      A
5028  FAFB  AA          TAX
5029  FAF9  60          RTS
5030
5031
5032  FAFA  2E B4 02      BITROL:      ROL      LOGMAP+2
5033  FAFD  2E B3 02      ROL      LOGMAP+1
5034  FB00  2E B2 02      ROL      LOGMAP
5035  FB03  60          RTS
5036      ;
5037      ;

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 27

```

5038 FB04 90 0C      BITPUT:      BCC      BITCLR      ; AND RETURN
5039                ; OTHERWISE FALL THROUGH TO BITSET AND RETURN
5040 FB06 20 EB FA      BITSET:      JSR      BITCON
5041 FB09 BD A3 02      LDA      TABMAP,X
5042 FB0C 05 6E      ORA      BITMSK
5043 FB0E 9D A3 02      STA      TABMAP,X
5044 FB11 60      RTS
5045                ;
5046 FB12 20 EB FA      BITCLR:      JSR      BITCON
5047 FB15 AS 6E      LDA      BITMSK
5048 FB17 49 FF      FOR      #$FF
5049 FB19 3D A3 02      AND      TABMAP,X
5050 FB1C 9D A3 02      STA      TABMAP,X
5051 FB1F 60      RTS
5052                ;
5053 FB20 A5 54      LOGGET:      LDA      ROWCRS
5054 FB22 18      LO1GET:      CLC
5055 FB23 69 78      LO2GET:      ADC      #120
5056 FB25 20 EB FA      BITGET:      JSR      BITCON
5057 FB28 18      CLC
5058 FB29 BD A3 02      LDA      TABMAP,X
5059 FB2C 25 6E      AND      BITMSK
5060 FB2E F0 01      BEQ      BITGE1
5061 F830 38      SEC
5062 FB31 60      BITGE1:      RTS
5063                ;
5064                ;
5065                ;
5066                ;
5067                ; INATAC: INTERNAL (CHAR) TO ATASCII (ATACHR) CONVERSION
5068                ;
5069 FB32 AD FA 02      INATAC:      LDA      CHAR
5070 FB35 A4 57      LDY      DINDEX      ; IF GRAPHICS MODES
5071 FB37 C0 03      CPY      #3
5072 FB39 B0 0F      BCS      INATAI      ; THEN DON'T CHANGE CHAR
5073 FB3B 2A      ROL      A
5074 FB3C 2A      ROL      A
5075 FB3D 2A      ROL      A
5076 FB3E 2A      ROL      A

```

```

5077 FB3F 29 03          AND    #3
5078 FB41 AA           TAX
5079 FB42 AD FA 02      LDA    CHAR
5080 FB45 29 9F        AND    #$9F
5081 FB47 1D FA FE      ORA    INTATA,X
5082 FB4A 8D FB 02      INATA1: STA  ATACHR
5083 FB4D 60           RTS
5084                   ;
5085                   ;
5086                   ;
5087                   ; MOVLIN: MOVE 40 BYTES AT FRMADR TO TOADR SAVING OLD TOADR
5088                   ;      DATA IN THE LINBUF. THEN MAKE NEXT FRMADR
5089                   ;      BE AT LINBUF FOR NEXT TRANSFER & TOADR=TOADR+40
5090                   ;
5091 FB4E A9 02      MOVLIN:  LDA    #LINBUF/256 ; SET UP ADRESS=LINBUF=3247
5092 FB50 85 65      STA    ADRESS+1
5093 FB52 A9 47      LDA    #LINBUF.AND.$FF
5094 FB54 85 64      STA    ADRESS
5095 FB56 A0 27      LDY    #39
5096 FB58 B1 66      MOVLII1: LDA    (TOADR),Y ; SAVE TO DATA
5097 FB5A 85 50      STA    TMPCHR
5098 FB5C B1 68      LDA    (FRMADR),Y ; STORE DATA
5099 FB5E 91 66      STA    (TOADR),Y
5100 FB60 A5 50      LDA    TMPCHR
5101 FB62 91 64      STA    (ADRESS),Y
5102 FB64 88        DEY
5103 FB65 10 F1      BPL    MOVLII1
5104 FB67 A5 65      LDA    ADRESS+1 ; SET UP FRMADR=LAST LINE
5105 FB69 85 69      STA    FRMADR+1
5106 FB6B A5 64      LDA    ADRESS
5107 FB6D 85 68      STA    FRMADR
5108 FB6F 18        CLC                ; ADD 40 TO TOADR
5109 F870 A5 66      LDA    TOADR
5110 FB72 69 28      ADC    #40
5111 FB74 85 66      STA    TOADR
5112 F876 90 02      BCC    MOVLII2
5113 FB78 E6 67      INC    TOADR+1
5114 FB7A 60      MOVLII2:  RTS
5115

```

```

5116 ;
5117 ;
5118 ; EXTEND: EXTEND BIT MAP FROM ROWCRS (EXTEND LOGICAL LINE)
5119 ;
5120 FD7B 08 EXTEND: PHP ; SAVE CARRY
5121 FB7C A0 17 LDY #23
5122 FB7E 98 EXTENT: TYA
5123 F87F 20 22 FB JSR LOIGET
5124 FB82 08 PHP
5125 FB83 98 TYA
5126 FBB4 18 CLC
5127 FB85 69 79 ADC #121
5128 FB87 28 PLP
5129 FB88 20 04 FB JSR BITPUT
5130 FBBB 88 EXTEN3: DEY
5131 FBBC 30 04 BMI EXTEN4
5132 FBBE C4 54 CPY ROWCRS
5133 FB90 B0 EC BCS EXTEN1
5134 FB92 A5 54 EXTEN4: LDA ROWCRS
5135 FB94 1B CLC
5136 FB95 69 78 ADC #120
5137 FB97 2B PLP
5138 FB98 4C 04 FB JMP BITPUT ; STORE NEW LINE'S BIT AND RETURN
5139 ;
5140 ;
5141 ;
5142 ; CLRLIN: CLEAR LINE CURSOR IS ON
5143 ;
5144 FB9B A5 52 CLRLIN: LDA LMARGN
5145 FB9D 85 55 STA COLCRS
5146 FB9F 20 47 F9 JSR CONVRT
5147 FBA2 A0 27 LDY 1139
5148 FBA4 A9 00 LDA #0
5149 FBA6 91 64 CLRLI1: STA (ADRESS),Y
5150 FBAB BB DEY
5151 FBA9 10 FB BPL CLRLI1
5152 FBAB 60 RTS
5153 ;
5154 ;

```

```

5155      ;
5156      ;
5157      ; SCROLL: SCROLL SCREEN
5158      ;
5159      FBAC 20 FA FA      SCROLL: JSR BITROL      ; ROLL IN CARRY
5160      FBAF A5 58      LDA SAVMSC      ; SET UP WORKING REGISTERS
5161      FBB1 85 64      STA ADDRESS
5162      FBB3 A5 59      LDA SAVMSC+1
5163      FBB5 85 65      STA ADDRESS+1
5164      FBB7 A0 28      SCROL1: LDY #40      ; LOOP
5165      FBB9 B1 64      LDA (ADDRESS),Y
5166      FBBB A6 6A      LDY RAMTOP      ; TEST FOR LAST LINE
5167      FBBD CA      DEX
5168      FBBE E4 65      CPX ADDRESS+1
5169      FBC0 D0 08      BNE SCROL2
5170      FBC2 A2 D7      LDX #$D7
5171      FBC4 E4 64      CPX ADDRESS
5172      FBC6 B0 02      BCS SCROL2
5173      FBCB A9 00      LDA #0      ; YES SO STORE ZERO DATA FOR THIS ENTIRE LINE
5174      FBCA A0 00      SCROL2: LDY #0
5175      FBCC 91 64      STA (ADDRESS),Y
5176      FBCE E6 64      INC ADDRESS
5177      FBD0 D0 E5      BNE SCROLL
5178      FBD2 E6 65      INC ADDRESS+1
5179      FBD4 A5 65      LDA ADDRESS+1
5180      FBD6 C5 6A      CMP RAMTOP
5181      FBD8 D0 DD      BNE SCROLL
5182      FBDA 4C DD FB      JMP DOLCOL      ; AND RETURN
5183      ;
5184      ;
5185      ; DOLCOL: DO LOGICAL COLUMN FROM BITMAP AND COLCRS
5186      ;
5187      FBDD A9 00      DOLCOL: LDA #0      ; START WITH ZERO
5188      FBDF 85 63      STA LOGCOL
5189      FBE1 A5 54      LDA ROWCRS
5190      FBE3 85 51      STA HOLD1
5191      FBE5 A5 51      DOLCO1: LDA HOLD1      ; ADD IN ROW COMPONENT
5192      FBE7 20 22 FB      JSR LOIGET
5193      FBEA B0 0C      BCS DOLCO2      ; FOUND BEGINNING OF LINE

```

```

5194 FBEC A5 63          LDA LOGCOL          ; ADD 40 AND LOOK BACK ONE
5195 FREE 18            CLC
5196 FBEF 69 28          ADC #40
5197 FBF1 85 63          STA LOGCOL
5198 FBF3 C6 51          DEC HOLD1          ; UP ONE LINE
5199 FBF5 4C E5 FB       JMP DOLCO1
5200 FBFB 18            DOLC02: CLC          ; ADD IN COLCRS
5201 FBF9 A5 63          LDA LOGCOL
5202 FBFB 65 55          ADC COLCRS
5203 FBFD 85 63          STA LOGCOL
5204 FBFF 60            RTS
5205                    ;
5206                    ;
5207                    ;
5208                    ; DOBUFC: COMPUTE BUFFER COUNT AS THE NUMBER OF BYTES FROM
5209                    ; BUFSTR TO END OF LOGICAL LINE WITH TRAILING SPACES REMOVED
5210                    ;
5211 FC00 20 9D FC       DOBUFC: JSR PHACRS
5212 FC03 A5 63          LDA LOGCOL
5213 FC05 48            PHA
5214 FC06 A5 6C          LDA BUFSTR          ; START
5215 FC08 85 54          STA ROWCRS
5216 FC0A A5 6D          LDA BUFSTR+1
5217 FC0C 85 55          STA COLCRS
5218 FC0E A9 01          LDA #1
5219 FC10 85 6B          STA BUFCNT
5220 FC12 A2 17          DOBUF1: LDX #23      ; NORMAL
5221 FC14 A5 7B          LDA SWPFLG          ; IF SWAPPED, ROW 3 IS THE LAST LINE ON SCREEN
5222 FC16 10 02          BPL DOB1
5223 FC18 A2 03          LDX #3
5224 FC1A E4 54          DOB1: CPX ROWCRS      ; TEST IF CRSR IS AT LAST SCREEN POSITION
5225 FC1C D0 0B          BNE DOBU1A
5226 FC1E A5 55          LDA COLCRS
5227 FC20 C5 53          CMP RMARGN
5228 FC22 D0 05          BNE DOBU1A
5229 FC24 E6 68          INC BUFCNT          ; YES, SO FAKE INCRSR TO AVOID SCROLLING
5230 FC26 4C 39 FC       JMP DOBUF2
5231 FC29 20 D4 F9       DOBU1A: JSR INCRSB
5232 FC2C E6 6B          INC BUFCNT

```

```

5233 FC2E A5 63          LDA LOGCOL
5234 FC30 C5 52          CMP LMARGN
5235 FC32 D0 DE          BNE DOBUF1      ; NOT YET EOL
5236 FC34 C6 54          DEC ROWCRS      :BACK UP ONE INCRSR
5237 FC36 20 99 F7       JSR CRSRLF
5238 FC39 20 A2 F5       DOBUF2: JSR GETPLT      ; TEST CURRENT COLUMN FOR NON-ZERO DATA
5239 FC3C D0 17          BNE DOBUF4      ; QUIT IF NON-ZERO
5240 FC3E C6 6B          DEC BUFCNT      ; DECREMENT COUNTER
5241 FC40 A5 63          LDA LOGCOL      ; BEGINNING OF LOGICAL LINE YET?
5242 FC42 C5 52          CMP LMARGN
5243 FC44 F0 0F          BEQ DOBUF4      ; YES, SO QUIT
5244 FC46 20 99 F7       JSR CRSRLF      ; BACK UP CURSOR
5245 FC49 A5 55          LDA COLCRS      ; IF LOGCOL=RMARGN, GO UP 1 ROW
5246 FC4B C5 53          CMP RMARGN
5247 FC4D D0 02          BNE DOBUF3
5248 FC4F C6 54          DEC ROWCRS
5249 FC51 A5 6B          DOBUF3: LDA BUFCNT
5250 FC53 D0 E4          BNE DOBUF2      ; LOOP UNLESS BUFCNT JUST WENT TO ZERO
5251 FC55 68             DOBUF4: PLA
5252 FC56 85 63          STA LOGCOL
5253 FC58 20 A8 FC       JSR PLACRS
5254 FC5B 60             RTS
5255                      ;
5256                      ;
5257                      ;
5258                      ;
5259                      ; STRBEG: MOVE   BUFSTR TO   BEGINNING OF LOGICAL LINE.
5260                      ;
5261 FC5C 20 DD FB       STRBEG: JSR DOLCOL      ; USE DOLCOL TO POINT HOLD1 AT BOL
5262 FC5F A5 51          LDA HOLD1
5263 FC61 85 6C          STA BUFSTR
5264 FC63 A5 52          LDA LMARGN
5265 FC65 85 6D          STA BUFSTR+1
5266 FC67 60             RTS
5267                      ;
5268                      ;
5269                      ;
5270                      ;
5271                      ;

```



```

5272      ; DELTIM: TIME TO DELETE A LINE IF IT IS EMPTY AND AN EXTENSION
5273      ;
5274 FC68 A5 63      DELTIA:   LDA   LOGCOL      ; IF LOGCOL<?LMAR GN
5275 FC6A C5 52      CMP   LMARGN      ; THEN DONT MOVE UP ONE
5276 FC6C D0 02      BNE   DELTIB      ; LINE BEFORE TESTING DELTIM
5277 FC6E C6 54      DEC   ROWCRS
5278 FC70 20 DD FB      DELTIB:   JSR   DOLCOL
5279 FC73 A5 63      DELTIM:   LDA   LDGCOL      ; TEST FOR EXTENSION
5280 FC75 C5 52      CMP   LMARGN
5281 FC77 F0 13      BEQ   DELT13      ; NO
5282 FC79 20 47 F9      JSR   CONVRT
5283 FC7C A5 53      LDA   RMARGN      ; SET UP COUNT
5284 FC7E 38          SEC
5285 FC7F E5 52      SBC   LMARGN
5286 FC81 A8          TAY
5287 FC82 B1 64      DELTI1:   LDA   (ADDRESS),Y
5288 FC84 D0 06      BNE   DELTI3      ; FOUND A NON-0 SO QUIT AND RETURN
5289 FC86 88          DEY
5290 FC87 10 F9      BPL   DELTI1
5291 FC89 4C DB F8      DELTI2:   JMP   DELLIB      ; DELETE A LINE AND RETURN
5292 FCBC 60          DELTI3:   RTS
5293      ;
5294      ;
5295      ;
5296      ; TSTCTL: SEARC  H CNTRLs TABLE TO SEE IF ATACHR IS A CNTL CHAR
5297      ;
5298 FC8D A2 2D      TSTCTL:   LDX   #45          ; PREPARE TO SEARCH TABLE
5299 FCSF BD C6 FE      TSTCTI:  LDA   CNTRLs,X
5300 FC92 CD FB 02      CMP   ATACHR
5301 FC95 F0 05      BEQ   TSTCT2
5302 FC97 CA          DEX
5303 FC98 CA          DEX
5304 FC99 CA          DEX
5305 FC9A 10 F3      BPL   TSTCT1
5306 FC9C 60          TSTCT2:  RTS
5307      ;
5308      ;
5309      ;
5310      ; PUSH ROWCRS, COLCRS AND COLCRS+1

```

```

5311 ;
5312 FC9D A2 02 PHACRS: LDX #2
5313 FC9F B5 54 PHACR1: LDA ROWCRS,X
5314 FCA1 9D B8 02 STA TMPROW,X
5315 FCA4 CA DEX
5316 FCA5 10 FS BPL PHACR1
5317 FCA7 60 RTS
5318 ;
5319 ;
5320 : PULL COLCRS+I, COLCRS AND ROWCRS
5321 ;
5322 FCAB A2 02 PLACRS: LDX #2
5323 FCAA BD B8 02 PLACR1: LDA TMPROW,X
5324 FCAD 95 54 STA ROWCRS,X
5325 FCAF CA DEX
5326 FCB0 10 FS BPL PLACR1
5327 FCB2 60 RTS
5328 ;
5329 ;
5330 ;
5331 ; SWAP: IF MI XED MODE. SWAP TEXT CURSORS WITH REGULAR CURSORS
5332 ;
5333 FCB3 20 B9 FC SWAP: JSR SWAPA ; THIS ENTRY POINT DOES RETUR1
5334 FCB6 4C 34 F6 JMP RETUR1
5335 FC89 AD BF 02 SWAPA: LDA BOTSCR
5336 FCBC C9 IS CMP #24
5337 FCBE F0 17 BEQ SWAPS
5338 FCC0 A2 0B LDX #11
5339 FCC2 B5 54 SWAP1: LDA ROWCRS,X
5340 FCC4 48 PHA
5341 FCC5 BD 90 02 LDA TXTROW,X
5342 FCCB 95 54 STA ROWCRS,X
5343 FCCA 68 PLA
5344 FCCB 9D 90 02 STA TXTROW,X
5345 FCCE CA DEX
5346 FCCF 10 F1 BPL SWAP1
5347 FCDI A5 7B LDA SWPFLG
5348 FCD3 49 FF FOR #$FF
5349 FCD5 85 7B STA SWPFLG

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 35

```

5350 FCD7 60      SWAP3:      RTS
5351              ;
5352              ;
5353              ; CLICK:      MAKE  CLICK THROUGH KEYBOARD SPEAKER
5354              ;
5355 FCDB  A2 7F    CLICK:      LDX   #$7F
5356 FCDA  8E 1F D0 CLICK1:    STX   CONSOL
5357 FCDD  8E 0A D4      STX   WSYNC
5358 FCE0  CA      DEX
5359 FCE1  10 F7    BPL   CLICK1
5360 FCE3  60      RTS
5361              ;
5362              ;
5363      • 5363    ; COLCR:    PUTS  EITHER O OR LMARGN INTO COLCRS BASED ON MODE AND SWPFLG
5364              ;
5365 FCE4  A9 00    COLCR:      LDA   #0
5366 FCE6  A6 7B      LDX   SWPFLG
5367 FCEB  D0 04      BNE   COLCR1
5368 FCEA  A6 57      LDX   DINDEX
5369 FCEC  D0 02      BNE   COLCR2
5370 FCEE  A5 52    COLCR1:    LDA   LMARGN
5371 FCF0  85 55    COLCR2:    STA   COLCRS
5372 FCF2  60      RTS
5373              ;
5374              ;
5375      ; PUTMSC: PUT SAVMSC INTO ADRESS
5376              ;
5377 FCF3  A5 58    PUTMSC:     LDA   SAVMSC      ; SET UP ADDRESS
5378 FCF5  85 64      STA   ADRESS
5379 FCF7  A5 59      LDA   SAVMSC+1
5380 FCF9  85 65      STA   ADRESS+1
5381 FCFB  60      RTS
5382              ;
5383              .PAGE
5384              ;
5385              ;
5386      ; DRAW - DRAW A LINE FROM OLDROW, OLD COL TO NEWROW, NEWCOL
5387      ; (THE AL MILLER METHOD FROM BASKETBALL)
5388 FCFC  A2 00    DRAW:      LDX   #0

```

```

5389 FCFE A5 22          LDA  ICCOMZ      ; TEST COMMAND: $11=DRAW $12=FILL
5390 FD00 C9 11          CMP   #$11
5391 FD02 F0 08          BEQ   DRAWA
5392 FD04 C9 12          CMP   #$12      ; TEST FILL
5393 FD06 F0 03          BEQ   DRAWS      ; YES
5394 FD0B A0 84          LDY   #NVALID    ; NO. SO RETURN INVALID COMMAND
5395 FD0A 60             RTS
5396 FD0B E8             INX
5397 FD0C 8E 87 02      DRAWS:
DRAWA: STX  FILFLG
5398 FD0F A5 54          LDA  ROWCRS      ; PUT CURSOR INTO NEWROW.NEWCOL
5399 FD11 85 60          STA  NEWROW
5400 FD13 A5 55          LDA  COLCRS
5401 FD15 85 61          STA  NEWCOL
5402 FD17 A5 56          LDA  COLCRS+1
5403 FD19 85 62          STA  NEWCOL+1
5404 FD1B A9 01          LDA  #1
5405 FD1D 85 79          STA  ROWINC      ; SET UP INITIAL DIRECTIONS
5406 FD1F 85 7A          STA  COLINC
5407 FD21 38            SEC
5408 FD22 A5 60          LDA  NEWROW      ; DETERMINE DELTA ROW
5409 FD24 E5 5A          SBC  OLDROW
5410 FD26 85 76          STA  DELTAR
5411 FD28 B0 0D          BCS  DRAW1      ; DO DIRECTION AND ABSOLUTE VALUE
5412 FD2A A9 FF          LDA  #$FF      ;BORROW WAS ATTEMPTED
5413 FD2C 85 79          STA  ROWINC      ; SET DIRECTION=DOWN
5414 FD2E A5 76          LDA  DELTAR
5415 FD30 49 FF          FOR  #$FF      ; DELTAR = :DELTAR.'
5416 FD32 18            CLC
5417 FD33 69 01          ADC  #1
5418 FD35 85 76          STA  DELTAR
5419 FD37 38            SEC
5420 FD38 AS 61          DRAW1: LDA  NEWCOL      ; NOW DELTA COLUMN
5421 FD3A E5 5B          SBC  OLDCOL
5422 FD3C 85 77          STA  DELTAC
5423 FD3E A5 62          LDA  NEWCOL+1    ; TWO-BYTE QUANTITY
5424 FD40 E5 5C          SBC  OLDCOL+1
5425 FD42 85 78          STA  DELTAC+1
5426 FD44 B0 16          BCS  DRAW2      ; DIRECTION AND ABSOLUTE VALUE
5427 FD46 A9 FF          LDA  #$FF      ; BORROW WAS ATTEMPTED

```

```

5428 FD48 85 7A          STA COLINC          ; SET DIRECTION = LEFT
5429 FD4A A5 77          LDA DELTAC
5430 FD4C 49 FF          FOR #$FF          ; DELTAC = 1DELTAC!
5431 FD4E 85 77          STA DELTAC
5432 FD50 A5 78          LDA DELTAC+1
5433 FD52 49 FF          FOR #$FF
5434 FD54 85 78          STA DELTAC+1
5435 FD56 E6 77          INC DELTAC          ; ADD ONE FOR TWOS COMPLEMENT
5436 FD58 D0 02          BNE DRAW2
5437 FD5A E6 78          INC DELTAC+1
5438 FD5C A2 02          LDX #2          ; ZERO RAM FOR DRAW LOOP
5439 FD5E A0 00          LDY #0
5440 FD60 84 73          STY COLAC+1
5441 FD62 98          DRAW3A: TYA
5442 FD63 95 70          STA ROWAC,X
5443 FD65 B5 5A          LDA OLDROW>X
5444 FD67 95 54          STA ROWCRS,X
5445 FD69 CA          DEX
5446 FD6A 10 F6          BPL DRAW3A
5447 FD6C A5 77          LDA DELTAC          ; FIND LARGER ONE (ROW OR COL)
5448          ;          STA COUNTR          ; (PREPARE COUNTR AND ENDPT)
5449          ;          STA ENDPT
5450 FD6E E8          INX          ; MAKE X 0
5451 FD6F A8          TAY
5452 FD70 A5 78          LDA DELTAC+1
5453 FD72 85 7F          STA COUNTR+1
5454 FD74 85 75          STA ENDPT+1
5455 FD76 D0 0B          BNE DRAWS          ; AUTOMATICALLY LARGER IF MSD>0
5456 FD78 A5 77          LDA DELTAC
5457 FD7A CS 76          CMP DELTAR          ; LOW COL >LOW ROW'?
5458 FD7C B0 0S          BCS DRAWS          ; YES
5459 FD7E A5 76          LDA DELTAR
5460 FD80 A2 02          LDX #2
5461 FD82 A8          TAY
5462 FD83 98          DRAWS: TYA          ; PUT IN INITIAL CONDITIONS
5463 FD84 85 7E          STA COUNTR
5464 FD86 85 74          STA ENDPT
5465 FD88 48          PHA          ; SAVE AC
5466 FD89 A5 75          LDA ENDPT+1          ; PUT LSB OF HIGH BYTE

```

```

5467 FD8B 4A          LSR  A          ; INTO CARRY
5468 FD8C 68          PLA          ; RESTORE AC
5469 FD8D 6A          ROR  A          ; ROR THE 9 BIT ACUMULATOR
5470 FD8E 95 70        STA  ROWAC,X
5471 FD90 A5 7E        DRAW4A: LDA  COUNTR      ; TEST ZERO
5472 FD92 05 7F        ORA  COUNTR+1
5473 FD94 D0 03        BNE  DRAW11      ; IF COUNTER IS ZERO, LEAVE DRAW
5474 FD96 4C 42 FE      JMP  DRAW10
5475 FD99 18          DRAW11: CLC          ; ADD ROW TO ROWAC (PLOT LOOP)
5476 FD9A A5 70        LDA  ROWAC
5477 FD9C 65 76        ADC  DELTAR
5478 FD9E 85 70        STA  ROWAC
5479 FDA0 90 02        BCC  DRAW5
5480 FDA2 E6 71        INC  ROWAC+1
5481 FDA4 A5 71        DRAW5: LDA  ROWAC+1      ; COMPARE ROW TO ENDPOINT
5482 FDA6 C5 75        CMP  ENDPT+1      ; IF HIGH BYTE OF ROW IS .LT. HIGH
5483 FDAB 90 14        BCC  DRAW6      ; BYTE OF ENDPT. BLT TO COLUMN
5484 FDAA D0 06        BNE  DRAW5A
5485 FDAC A5 70        LDA  ROWAC
5486 FDAE C5 74        CMP  ENDPT      ; LOW BYTE
5487 FDB0 90 0C        BCC  DRAW6      ; ALSO BLT
5488 FDB2 18          DRAW5A: CLC          ; GE SO MOVE POINT
5489 FDB3 A5 54        LDA  ROWCRS
5490 FDB5 65 79        ADC  ROWINC
5491 FDB7 85 54        STA  ROWCRS
5492 FDB9 A2 00        LDX  #0          ; AND SUBTRACT ENDPT FROM ROWAC
5493 FDBB 20 7A FA      JSR  SUBEND
5494 FDBE 18          DRAW6: CLC          ; DO SAME FOR COLUMN (DOUBLE BYTE ADD)
5495 FDBF A5 72        LDA  COLAC      ; ADD
5496 FDC1 65 77        ADC  DELTAC
5497 FDC3 85 72        STA  COLAC
5498 FDC5 A5 73        LDA  COLAC+1
5499 FDC7 65 78        ADC  DELTAC+1
5500 FDC9 85 73        STA  COLAC+1
5501 FDCB C5 75        CMP  ENDPT+1      ; COMPARE HIGH BYTE
5502 FDCD 90 27        BCC  DRAWS
5503 FDCF D0 06        BNE  DRAW6A
5504 FDD1 A5 72        LDA  COLAC      ; COMPARE LOW BYTE
5505 FDD3 C5 74        CMP  ENDPT

```

```

5506 FDD5 90 1F          BCC  DRAWS
5507 FDD7 24 7A          DRAW6A: BIT  COLINC
5508 FDD9 10 10          BPL  DRAW6B
5509 Fddb C6 55          DEC  COLCRS      ; DO DOUBLE BYTE DECREMENT
5510 FDDD A5 55          LDA  COLCRS
5511 FDDF C9 FF          CMP  #$FF
5512 FDE1 D0 0E          BNE  DRAW7
5513 FDE3 A5 56          LDA  COLCRS+1
5514 FDE5 F0 0A          BEQ  DRAW7      ; DON'T DEC IF ZERO
5515 FDE7 C6 56          DEC  COLCRS+1
5516 FDE9 10 06          BPL  DRAW7      ; (UNCONDITIONAL)
5517 FDEB E6 55          DRAW6B: INC  COLCRS      ; DO DOUBLE BYTE INCREMENT
5518 FDED D0 02          BNE  DRAW7
5519 FDEF E6 56          INC  COLCRS+1
5520 FDF1 A2 02          DRAW7:  LDX  #2      ; AND SUBTRACT ENDPT FROM COLAC
5521 FDF3 20 7A FA        JSR  SUBEND
5522 FDF6 20 96 FA        DRAW8:  JSR  RANGE
5523 FDF9 20 E0 F5        JSR  OUTPLT      ; PLOT POINT
5524 FDFC AD B7 02        LDA  FILFLG      ; TEST RIGHT FILL
5525 FDFF F0 2F          BEQ  DRAWS
5526 FE01 20 9D FC        JSR  PHACRS
5527 FE04 AD FB 02        LDA  ATACHR
5528 FE07 8D BC 02        STA  HOLD4
5529 FE0A A5 54          DRAW8A:  LDA  ROWCRS      ; SAVE ROW IN CASE OF CR
5530 FE0C 48             PHA
5531 FE0D 20 DC F9        JSR  INCRSA      ; POSITION CURSOR ONE PAST DOT
5532 FE10 68             PLA      ; RESTORE ROWCRS
5533 FE11 85 54          STA  ROWCRS
5534 FE13 20 96 FA        DRAW8C:  JSR  RANGE
5535 FE16 20 A2 F5        JSR  GETPLT      ; GET DATA
5536 FE19 D0 0C          BNE  DRAWSB      ; STOP IF NON-ZERO DATA IS ENCOUNTERED
5537 FE1B AD FD 02        LDA  FILDAT      ; FILL DATA
5538 FETE 8D FB 02        STA  ATACHR
5539 FE21 20 E0 F5        JSR  OUTPLT      ; DRAW IT
5540 FE24 4C 0A FE        JMP  DRAWSA      ; LOOP
5541 FE27 AD BC 02        DRAW8B:  LDA  HOLD4
5542 FE2A SD FB 02        STA  ATACHR
5543 FE2D 20 AB FC        JSR  PLACRS
5544 FE30 36             DRAWS:  SEC      ; DO DOUBLE BYTE SUBTRACT

```

```

5545 FE31 A5 7E          LDA   COUNTR
5546 FE33 E9 01          SBC   #1
5547 FE35 85 7E          STA   COUNTR
5548 FE37 A5 7F          LDA   COUNTR+1
5549 FE39 E9 00          SBC   #0
5550 FE3B 85 7F          STA   COUNTR+1
5551 FE3D 30 03          BMI   DRAW10
5552 FE3F 4C 90 FD          JMP   DRAW4A
5553 FE42 4C 34 F6      DRAW10: JMP   RETUR1
5554                   .PAGE
5555                   ;
5556                   ;
5557                   ;   TABLES
5558                   ;
5559                   ;
5560                   ;   MEMORY ALLOCATION
5561                   ;
5562 FE45 18 10 0A 0A      ALOCAT: .BYTE 24,16,10,10,16,28,52,100,196,196,196,196
5563 FE49 10 1C 34 64
5564 FE4D C4 C4 C4 C4
5565                   ;
5566                   ;
5567                   ;   NUMBER OF DISPLAY LIST ENTRIES
5568                   ;
5569 FE51 17 17 0B 17      NUMDLE: .BYTE 23,23,11,23,47,47,95,95,97,97,97,97
5570 FE55 2F 2F 5F 5F
5571 FE59 61 61 61 61
5572 FE5D 13 13 09 13      MXDMDE: .BYTE 19,19,9,19,39,39,79,79,65,65,65,65 ; (EXT OF NUMDLE)
5573 FE61 27 27 4F 4F
5574 FE65 41 41 41 41
5575                   ;
5576                   ;
5577                   ; ANTIC CODE FROM INTERNAL MODE CONVERSION TABLE
5578                   ;
5579                   ;   INTERNAL      ANTIC CODE      DESCRIPTION
5580                   ;   0              2              40X2X8  CHARACTERS
5581                   ;   1              6              20X5X8   ""
5582                   ;   2              7              20X5X16  ""
5583                   ;   3              8              40X4X8  GRAPHICS

```


ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 41

```
5584 ; 4 9 80X2X4 GRAPHICS
5585 ; 5 A 80X4X4 ""
5586 ; 6 B 160X2X2 ""
5587 ; 7 D 160X4X2 ""
5588 ; 8 F 320X2XI ""
5589 ; 9 SAME AS 8 BUT GTIA 'LUM' MODE
5590 ; 10 SAME AS 8 BUT GTIA 'COL/LUM REGISTER' MODE
5591 ; 11 SAME AS 8 BUT GTIA 'COLOR' MODE
5592 ;
5593 FE69 02 06 07 08 ANCONV: .BYTE 2,6,7,8,9,$A,$B,$D,$F,$F,$F,$F ;ZEROS FOR RANGE TEST IN
5594 FE6D 09 0A 0B 0D
5595 FE71 0F 0F 0F 0F
5596 ;
5597 ;
5598 ; PAGE TABLE TELLS WHICH DISPLAY LISTS ARE IN DANGER OF
5599 ; CROSSING A 256 BYTE PAGE BOUNDARY
5600 ;
5601 FE75 00 00 00 00 PAGETB: .BYTE 0101010101010111111111
5602 FE79 00 00 00 01
5603 FE7D 01 01 01 01
5604 ;
5605 ;
5606 ; THIS IS THE NUMBER OF LEFT SHIFTS NEEDED TO MULTIPLY
5607 ; COLCRS BY 10,20, OR 40. (ROWCRS*10)/(2**DHLINE)
5608 ;
5609 FE81 02 01 01 00 DHLINE: .BYTE 2,1,1,0,0,1,1,2,2,2,2,2
5610 FE85 00 01 01 02
5611 FE89 02 02 02 02
5612 ;
5613 ;
5614 ; COLUMN: NUMBER OF COLUMNS
5615 ;
5616 FEED 28 14 14 28 COLUMN: .BYTE 40,20,20,40,80,80,160,160,64,80,80,80 ; MODE 8 IS SPECIAL
5617 FE91 50 50 A0 A0
5618 FE95 40 50 50 50
5619 ;
5620 ;
5621 ;
5622 NOROWS: NUMBER OF ROWS
```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 42

```

5623 ;
5624 FE99 18 18 0C 18 NOROWS: .BYTE 24,24,12,24,48,48,96,96,192,192,192,192
5625 FE9D 30 30 60 60
5626 FEAT C0 C0 C0 C0
5627 ;
5628 ;
5629 ;
5630 ;
5631 ; DIV2TB: HOW MANY RIGHT SHIFTS FOR HCRSR FOR PARTIAL BYTE MODES
5632 ;
5633 FEA5 00 00 00 02 DIV2TB: .BYTE 0,0,0,2,3,2,3,2,3,1,1,1
5634 FEA9 03 02 03 02
5635 FEAD 03 01 01 01
5636 ;
5637 ;
5638 ; DMASKT: DISPLAY MASK TABLE
5639 ;
5640 FEB1 00 FF F0 0F DMASKT: .BYTE $00, $FF, $F0, $0F
5641 FEB5 C0 30 0C 03 .BYTE $C0,$30,$0C,$03
5642 ;
5643 ; MASKTB: BIT MASK. (ALSO PART OF DMASKTB! DO NOT SEPARATE)
5644 ;
5645 FEB9 80 40 20 10 MASKTB: .BYTE $80, $40, $20, $10, $08, $04, $02, $01
5646 FEED 08 04 02 01
5647 ;
5648 ;
5649 ;
5650 ;
5651 FEC1 28 CA 94 46 COLRTB: .BYTE $28, $CA, $94, $46, $00
5652 FEC5 00
5653 ;
5654 ;
5655 ;
5656 ;
5657 ; CNTRLS: CONTROL CODES AND THEIR DISPLACEMENTS INTO THE
5658 ; CONTROL CHARACTER PROCESSORS
5659 ;
5660 FEC6 1B CNTRLS: .BYTE $1B
5661 FEC7 79 F7 .WORD ESCAPE

```

ERR	LINE	ADR.	R1	R2	R3	R4	DISPLAY HANDLER - 10-30-78 - DISPLC
	5662	FEC9	1C				.BYTE 31C
	5663	FECA	7F	F7			.WORD CRSRUP
	5664	FECC	1D				.BYTE 31D
	5665	FECD	8C	F7			.WORD CRSRDN
	5666	FECF	1E				.BYTE s1E
	5667	FED0	99	F7			.WORD CRSRLF
	5668	FED2	1F				.BYTE \$1F
	5669	FED3	AA	F7			.WORD CRSRRT
	5670	FED5	7D				.BYTE 37D
	5671	FED6	B9	F7			.WORD CLRSCR
	5672	FEDS	7E				.BYTE \$7E
	5673	FED9	E6	F7			.WORD BS
	5674	FEDB	7F				.BYTE 37F
	5675	FEDC	10	FS			.WORD TAB
	5676	FEDE	98				.BYTE \$98
	5677	FEDF	30	FA			.WORD DOCRWS
	5678	FEE1	9C				.BYTE \$9C
	5679	FEE2	D4	F8			.WORD DELLIN
	5680	FEE4	9D				.BYTE \$9D
	5681	FEE5	A4	F8			.WORD INSLIN
	5682	FEE7	9E				.BYTE 49E
	5683	FEES	32	FS			.WORD CLRTAB
	5684	FEEA	9F				RYTE \$9F
	5685	FEEB	2D	F8			.WORD SETTAB
	5686	FEED	FD				.BYTE \$FD
	5687	EEEE	0A	F9			.WORD BELL
	5688	FEF0	FE				.BYTE \$FE
	5689	FEF1	6D	FS			.WORD DELCHR
	5690	FEF3	FF				.BYTE \$FF
	5691	FEF4	37	F8			.WORD INSCHR
	5692						;
	5693						;
	5694						;
	5695						;
	5696						;
	5697						; ATAIN
	5698						: ATASCI TO INTERNAL TABLE
	5699	FEF6	40	00	20	60	ATAINT:
	5700						.BYTE \$40,\$00,\$20,\$60
							;

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 44

```

5701 ;
5702 ; INTATA : INTERNAL TO ATASCI TABLE
5703 ;
5704 FEFA 20 40 00 60 INTATA: .BYTE $20,$40,$00,$60
5705 ;
5706 ;
5707 ATASCI : ATASC II CONVERSION TABLE
5708 ;
5709 FEFE 6C 6A 3B 80 ATASCI: .BYTE $6C, $6A, $3B, $80, $80, $68, $28, $2A ; LOWER CASE
5710 FF02 80 6B 2B 2A
5711 FF06 6F 80 70 75 .BYTE $6F, $80, $70, $75, $9B, 369, $2D, $3D
5712 FF0A 9B 69 2D 3D
5713 FF0E
5714 FF0E 76 80 63 80 .BYTE $76, $80, $63, $80, $80, $62, $78, $7A
5715 FF12 80 62 78 7A
5716 FF16 34 80 33 36 .BYTE $34, $80, $33, $36, $1B, $35, $32, $31
5717 FF1A 1B 35 32 31
5718 FF1E
5719 FF1E 2C 20 2E 6E .BYTE $2C, $20, $2E, $6E, $80, $6D, $2F, $81
5720 FF22 80 6D 2F 81
5721 FF26 72 80 65 79 .BYTE $72, $80, $65, $79, $7F, $74, $77, $71
5722 FF2A 7F 74 77 71
5723 FF2E
5724 FF2E 39 80 30 37 .BYTE $39, $80, $30, $37, $7E, $3B, $3C, $3E
5725 FF32 7E 38 3C 3E
5726 FF36 66 68 64 80 .BYTE $66, $68, $64, $80, $82,$67, $73, $61
5727 FF3A 82 67 73 61
5728 FF3E
5729 FF3E
5730 FF3E 4C 4A 3A 80 .BYTE $4C, $4A, $3A, $80, #80, $4B, $5C, $5E ; UPPER CASE
5731 FF42 80 4B 5C 5E
5732 FF46 4F 80 50 55 .BYTE $4F, $80, $50, $55, $9B, $49, $5F, $7C
5733 FF4A 9B 49 5F 7C
5734 FF4E
5735 FF4E 56 80 43 80 .BYTE $56, $80, $43, $80, $80, $42, $58, $5A
5736 FF52 80 42 58 5A
5737 FF56 24 80 23 26 .BYTE $24, $80, $23, $26, $1B, $25, $22, $21
5738 FF5A 1B 25 22 21
5739 FF5E

```

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 45

5740	FF5E	5B 20 5D 4E		.BYTE \$5B, \$20, \$5D, \$4E, \$B0, \$4D, \$3F, \$81
5741	FF62	80 4D 3F 81		
5742	FF66	52 80 45 59		.BYTE \$52, \$80, \$45, \$59, \$9F, \$54, \$57, \$51
5743	FF6A	9F 54 57 51		
5744	FF6E			
5745	FF6E	28 80 29 27		.BYTE \$28, \$80, \$29, \$27, \$9C, \$40, \$7D, \$9D
5746	FF72	9C 40 7D 9D		
5747	FF76	46 48 44 80		.BYTE \$46, \$48, \$44, \$80, \$83, \$47, \$53, \$41
5748	FF7A	83 47 53 41		
5749	FF7E			
5750	FF7E			
5751	FF7E	0C 00 7B 80		.BYTE \$0C, \$0A, \$7B, \$80, \$80, \$0B, \$1E, \$1F ; CONTROL
5752	FF82	80 0B 1E 1F		
5753	FF86	0F B0 10 15		.BYTE \$0F, \$B0, \$10, \$15, \$9B, \$09, \$1C, \$1D
5754	FFSA	9B 09 1C 1D		
5755	FFBE			
5756	FFBE	16 80 03 80		.BYTE \$16, \$80, \$03, \$80, \$B0, \$02, \$18, \$1A
5757	FF92	80 02 18 1A		
5758	FF96	80 80 85 80		.BYTE \$80, \$80, \$85, \$80, \$1B, \$80, \$FD, \$80
5759	FF9A	1B 80 FD 80		
5760	FF9E			
5761	FF9E	00 20 60 0E		.BYTE \$00, \$20, \$60, \$0E, \$80, \$0D, \$80, \$81
5762	FFA2	80 0D 80 81		
5763	FFA6	12 80 05 19		.BYTE \$12, \$80, \$05, \$19, \$9E, \$14, \$.17, \$11
5764	FFAA	9E 14 17 11		
5765	FFAE			
5766	FFAE	80 80 80 80		.BYTE \$80, \$80, \$80, \$80, \$FE, \$80, \$7D, \$FF
5767	FFB2	FE 80 7D FF		
5768	FFB6	06 08 04 80		.BYTE \$06, \$08, \$04, \$B0, \$84, \$07, \$13, \$01
5769	FFBA	84 07 13 01		
5770				
5771				
5772				
5773				
5774				
5775	FFBE	AD 09 D2	PIRQ5:	LDA KBCODE
5776	FFC1	CD F2 02		CMP CHI ; TEST AGAINST LAST KEY PRESSED
5777	FFC4	D0 05		BNE PIR03 ;IF NOT, GO PROCESS KEY
5778	FFC6	AD F1 02		LDA KEYDEL ;F KEY DELAY BYTE > 0

ERR LINE ADR. R1 R2 R3 R4

DISPLAY HANDLER - 10-30-78 - DISPLC

Page 46

```

5779 FFC9 D0 20          BNE   PIRQ4      ; IGNORE KEY AS BOUNCE
5780 FFCB AD 09 D2      PIRQ3: LDA   KBCODE    ; RESTORE AC
5781 FFCE C9 9F          CMP   #CNTL1     ; TEST CONTROL 1 (SSFLAG)
5782 FFD0 D0 0A          BNE   PIRQ1
5783 FFD2 AD FF 02      LDA   SSFLAG
5784 FFD5 49 FF          FOR   #$FF
5785 FFD7 8D FF 02      STA   SSFLAG
5786 FFDA B0 0F          BCS   PIRQ4      ; (UNCONDITIONAL) MAKE ^1 INVISIBLE
5787 FFDC 8D FC 02      PIRQ1: STA   CH
5788 FFDF 8D F2 02      STA   CH1
5789 FFE2 A9 03          LDA   #3
5790 FFE4 8D F1 02      STA   KEYDEL     ; INITIALIZE KEY DELAY FOR DEBOUNCE
5791 FFE7 A9 00          LDA   #0        ; CLEAR COLOR SHIFT BYTE
5792 FFE9 85 4D          STA   ATRACT
5793 FFEB A9 30          PIRQ4: LDA   #$30
5794 FFED 8D 2B 02      STA   SRTIMR
5795 FFF0 68          PIRQ2: PLA
5796 FFF1 40          RTI
5797
5798
5799 FFF2 FF FF FF FF    .BYTE $FF, $FF, $FF, $FF, $FF, $FF
5800 FFF6 FF FF
5801
5802 FFFB          CRNTPC   =*
5803          *= $14
5804 0014 00          KBDSPR: .BYTE $FFFB-CRNTPC : ^GDISPLC IS TOO LONG
5805 0015          .END
ASSEMBLY ERRORS = 0

```