```
                        ttl     "tp I/065 Rom routines       %s                    Page %d"

                ;********** I/O Part variables **********

                ;       6502 I/O part DOS65 and bootstrap
                ;         Written by E.J.M. Visschedijk
                ;Copyright (c) 1986 Kim gebruikersclub Nederland

                        pag     80

                ;Variables and definitions

        0708    TOUTT   equ     1800            ;Time out for screen off in seconds

                ;Memory map definitions

        E000    FDCPIA  equ     $E000           ;Pia address on fdc card
        E004    FDCFDC  equ     $E004           ;FDC address on fdc card
        E100    IOVIA1  equ     $E100           ;Via 1 address on cpu card
        E110    IOVIA2  equ     $E110           ;Via 2 address on cpu card
        E130    IOACIA  equ     $E130           ;Acia address on cpu card
        E140    IOCRTC  equ     $E140           ;CRTC address on vdu card
        E400    BTBLCKS equ     $E400           ;Temporary blocks for bootstrap
        E700    IO_VARS equ     $E700           ;Variable area for IO65
        E800    VDDRAM  equ     $E800           ;Start of 2Kbyte VDU-ram
        F000    IO_BEG  equ     $F000           ;Begin of IO65
        FFF0    DTRAM   equ     $FFF0           ;DAT ram address for virtual disk

                ;Temporary variables on zeropage

        0000    PTA     equ     $00
        0002    PTB     equ     $02

                ;Bootstrap variables

        00FB    SECC    equ     $FB             Current tsl
        00FC    RPOIN   equ     $FC             Input data pointer
        00FE    ERCNT   equ     $FE             Floppy errors read
        00FF    ERC1    equ     $FF             Floppy dens/restore

        E700            org     IO_VARS         ;Start variable aria

E700            SAVSTACKP res   1               ;Software stackpointer
E701            STATTOG res     1               ;Kind of statusline
E702            CRPX    res     1               ;Print var. for cursor X position
E703            CRPY    res     1               ;Print var. for cursor Y position
E704            CURPX   res     1               ;Current cursor X position
E705            CURPY   res     1               ;Current cursor Y position
E706            CURP    res     1               ;Direct cursor addressing flag
E707            SCURPX  res     1               ;CURPX save address
E708            SCURPY  res     1               ;CURPY save address
E709            XTMP    res     1
E70A            XPSOLD  res     1
E70B            YPSOLD  res     1
E70C            OUTCHR  res     1
E70D            MAXSTL  res     1
E70E            HSFLG   res     1               ;^Q ^S flag
E70F            DEVCHOIC res    1               ;$00=select output, $01=select input
E710            DEVMODOUT res   1               ;Output device mode bits
E711            DEVMODINP res   1               ;Input device mode bits
E712            SDEVMODOUT res  1               ;DEVMODOUT saveaddress
E713            SDEVMODINP res  1               ;DEVMODINP saveaddress
E714            SDVOUT  res     1
E715            SDVINP  res     1
E716            OUTRET  res     1               ;Default output device bits
E717            INPRET  res     1               ;Default input device bits
E718            CLSEC   res     1               ;Variable for clock update
E719            FUNENA  res     1               ;Flag for next char. is function key
E71A            FNCEN   res     1               ;Disable function keys =(FF)
E71B            VIAVRA  res     1               ;Vialoop time out variable
E71C            GRFLG   res     1               ;Grafics flag
E71D            PNTXLSAVE res   1               ;Saveaddress for PNTXL
E71E            PNTXHSAVE res   1               ;Saveaddress for PNTXH
E71F            PNTX    res     1               ;Program in ram, self modifying
E720            PNTXL   res     1
E721            PNTXH   res     2               ;Also one byte for RTS
E723            DMPRAM  res     1               ;Program in ram, self modifying
E724            DUMPL   res     1
```

```
E725            DUMPH   res     2               ;Also one byte for RTS
E727            LDALET  res     1               ;Program in ram, self modifying
E728            LETADRL res     1
E729            LETADRH res     2               ;Also one byte for RTS
E72B            FRWR    res     1               ;Program in ram, self modifying
E72C            FRWRL   res     1
E72D            FRWRH   res     2               ;Also one byte for RTS
E72F            FRRE    res     1               ;Program in ram, self modifying
E730            FRREL   res     1
E731            FRREH   res     2               ;Also one byte for RTS
E733            MONESC  res     1               ;'Escape' sequence variable for monitor
E734            ACCTL   res     1               ;Variable for acia control register
E735            ACCMD   res     1               ;Variable for acia command register
E736            TIMDAT  res     1               ;Switch for print time or date
E737            SEC1.20 res     1               ;Count 1/20 of seconds
E738            TOUTF   res     1
E739            TREMP   res     1               ;Transmit register empty flag
E73A            DECI1   res     1               ;Variables for calculate decimal
E73B            DECI2   res     1
E73C            DECI3   res     1
E73D            LNRL    res     1               ;Linenumber in a file, lowbyte
E73E            LNRH    res     1               ;Linenumber in a file, highbyte
E73F            COMP    res     2               ;Compare endaddress in EEOS
E741            ESCFLG  res     1               ;Escapeflag
E742            STATFG  res     1               ;Select local statusline flag
E743            INVERS  res     1               ;Normal or invers video
E744            INVERSS res     1               ;INVERS saveaddress
E745            INVST   res     1               ;Default invers for statusline
E746            CHKCRT  res     1               ;Check crt controller

        E750            org     IO_VARS+$50
E750            NDVRS

                ;       See IODEFS for meanings of interrupts

E750            INTV1   res     2               interrupt 1 vector
E752            INTV2   res     2               interrupt 2 vector
E754            INTV3   res     2               interrupt 3 vector
E756            INTV4   res     2               interrupt 4 vector
E758            INTV5   res     2               interrupt 5 vector
E75A            INTV6   res     2               interrupt 6 vector
E75C            INTV7   res     2               interrupt 7 vector
E75E            INTV8   res     2               interrupt 8 vector
E760            INTV9   res     2               interrupt 9 vector
E762            INTV10  res     2               interrupt 10 vector
E764            INTV11  res     2               interrupt 11 vector
E766            INTV12  res     2               interrupt 12 vector
E768            INTV13  res     2               interrupt 13 vector
E76A            INTV14  res     2               interrupt 14 vector
E76C            INTV15  res     2               interrupt 15 vector
E76E            INTV16  res     2               interrupt 16 vector

E770            SVAINT  res     1               ;Accu save in interrupt routine
E771            COUD    res     1               ;Old cursor mode saveaddress
E772            TOUTIL  res     1
E773            TOUTIH  res     1
E774            TOUTL   res     1               ;Time out for screen off
E775            TOUTH   res     1
E776            WRBEG   res     2               ;Begin address mem. as output
E778            WREND   res     2               ;End address mem. as output
E77A            REBEG   res     2               ;Begin address mem. as input
E77C            REEND   res     2               ;End address mem. as input

E77E            DATUPD  res     1               ;Date updated flag
E77F            DAY     res     1               ;Date variables
E780            MONTH   res     1               ;
E781            YEAR    res     1               ;
                ;
E782            HOURS   res     1               ;Time variables
E783            MINUTES res     1
E784            SECONDS res     1
                ;
E785            DVO4VEC res     2               ;Output device 4 vector, don't change
E787            DVI4VEC res     2               ;Input device 4 vector, don't change
                ;
                ;*** I/O device vectors, free for the user ***
                ;
E789            DVO6VEC res     2               ;Output device 6 vector
```

Peter de Bu

```
E78B            DVI6VEC res     2               ;Input device 6 vector
E78D            DVO7VEC res     2               ;Output device 7 vector
E78F            DVI7VEC res     2               ;Input device 7 vector
E791            DVO8VEC res     2               ;Output device 8 vector
E793            DVI8VEC res     2               ;Input device 8 vector
                ;
                ;*** Software stack aria ***
                ;
E795            ASAVESTACK res  8               ;Accustack
E79D            XSAVESTACK res  8               ;Xreg stack
E7A5            YSAVESTACK res  8               ;Yreg stack
                ;
E7AD            START   res     2               ;Pointer to start of videoram
                ;
E7AF            UNRINT  res     2               ;Unrecognized interrupt vector
E7B1            NMIVECTOR res   2
E7B3            BRKVECTOR res   2
E7B5            IRQVECTOR res   2
                ;

                ;*** Buffer aria ***

E7B7            KEYPNT  res     1               ;Pointer from keyboardbuffer
E7B8            KEYBUF  res     40              ;Keyboardbuffer, 40 char.
          0028  MAXBUF  equ     40              ;Number of characters in buffer

                ;FDC COMMANDS

          0009  REST    equ     $09             ;12 MS
          0019  SEEKT   equ     $19             ;12ms for canon and others

          0080  RDSKT   equ     $80

          E400          org     BTBLCKS
                ;
E400            SYSB    res     256             ;Boot system blok
E500            TSLB    res     256             ;Boot tsl blok

                ;*** Addresses on floppydisk controller ***

                ;*** Pia addresses ***

          E000  PIA     equ     FDCPIA
          E000  PAD     equ     PIA             ;Data and datadirection A
          E001  PAC     equ     PIA+$01         ;Command register
          E002  PBD     equ     PIA+$02         ;Data and datadirection B
          E003  PBC     equ     PIA+$03         ;Command register

                ;*** Fdc addresses ***

          E004  FDC     equ     FDCFDC
          E004  CMR     equ     FDC             ;Command register
          E004  STR     equ     FDC             ;Status register
          E005  TKR     equ     FDC+$01         ;Track register
          E006  SCR     equ     FDC+$02         ;Sector register
          E007  DTR     equ     FDC+$03         ;Data register

                ;*** Via 1 addresses, A=keyboard, B=Centronics printer data ***

          E100  VIAA    equ     IOVIA1
          E100  VAPBD   equ     VIAA            ;Port B data
          E101  VAPAD   equ     VIAA+$01        ;Port A data
          E102  VAPBDD  equ     VIAA+$02        ;Port B data direction
          E103  VAPADD  equ     VIAA+$03        ;Port A data direction
          E104  VATACL  equ     VIAA+$04        ;T1, latch-low, counter low
          E105  VATACH  equ     VIAA+$05        ;T1, counter high
          E106  VATALL  equ     VIAA+$06        ;T1, latch low
          E107  VATALH  equ     VIAA+$07        ;T1, latch high
          E108  VATBCL  equ     VIAA+$08        ;T2, latch low, counter low
          E109  VATBCH  equ     VIAA+$09        ;T2, counter high
          E10A  VASR    equ     VIAA+$0A        ;Shift register
          E10B  VAACR   equ     VIAA+$0B        ;Auxiliary control register
          E10C  VAPCR   equ     VIAA+$0C        ;Peripheral control register
          E10D  VAIFR   equ     VIAA+$0D        ;Interrupt flag register
          E10E  VAIER   equ     VIAA+$0E        ;Interrupt enable register
          E10F  VAADN   equ     VIAA+$0F        ;Port A data, no handshake

                ;*** Via 2 addresses, A=Viacom, B3,4=Centronics printer ***
```

```
E110  VIAB    equ     IOVIA2
E110  VBPBD   equ     VIAB            ;Port B data
E111  VBPAD   equ     VIAB+$01        ;Port A data
E112  VBPBDD  equ     VIAB+$02        ;Port B data direction
E113  VBPADD  equ     VIAB+$03        ;Port A data direction
E114  VBTACL  equ     VIAB+$04        ;T1, latch-low, counter low
E115  VBTACH  equ     VIAB+$05        ;T1, counter high
E116  VBTALL  equ     VIAB+$06        ;T1, latch low
E117  VBTALH  equ     VIAB+$07        ;T1, latch high
E118  VBTBCL  equ     VIAB+$08        ;T2, latch low, counter low
E119  VBTBCH  equ     VIAB+$09        ;T2, counter high
E11A  VBSR    equ     VIAB+$0A        ;Shift register
E11B  VBACR   equ     VIAB+$0B        ;Auxiliary control register
E11C  VBPCR   equ     VIAB+$0C        ;Peripheral control register
E11D  VBIFR   equ     VIAB+$0D        ;Interrupt flag register
E11E  VBIER   equ     VIAB+$0E        ;Interrupt enable register
E11F  VBADN   equ     VIAB+$0F        ;Port A data, no handshake

      ;*** Acia addresses ***

E130  ACIA    equ     IOACIA
E130  RECREG  equ     ACIA            ;Receiver register
E130  TRAREG  equ     ACIA            ;Transmitter register
E131  ACIASR  equ     ACIA+$01        ;Status register
E132  ACICMD  equ     ACIA+$02        ;Command register
E133  ACICTL  equ     ACIA+$03        ;Control register

      ;*** Crtc addresses ***

E140  CRTC    equ     IOCRTC
E140  CRTCAR  equ     CRTC            ;Address register
E141  CRTCRF  equ     CRTC+$01        ;Register file

      ;*** Dat ram address for virtual disk ***

FFF0  DATRAM  equ     DTRAM           ;DATRAM for VDISK

      ;*** Macro definitions ***

      TIME    macro   EENH,MAX
              LDX     EENH
              INX
              STX     EENH
              CPX     #MAX
              BNE     INTND
              STA     EENH
              endm

      CASE1   macro   VOORW1,BESTEM1
              CMP     #VOORW1
              BNE     1.F
              JMP     BESTEM1
      1
              endm

      CASE2   macro   VOORW2,BESTEM2
              CMP     #VOORW2
              BNE     BESTEM2
              endm

      CASE3   macro   VOORW3,BESTEM3
              CMP     #VOORW3
              BEQ     BESTEM3
              endm

              cont    RESINIT.MAC
      ;********** RESINIT **********

F000          org     IO_BEG

      ;*** JUMP TABEL ***

F000 4C DDF6  I065    JMP     OUT             To active outputs, A,X,Y restored
F003 4C 99F6          JMP     INP             To active inputs, X,Y restored
F006 4C F6F6          JMP     OUTX            To outputs pointed by X,Y
F009 4C A7F6          JMP     INPX            To inputs pointed by X
F00C 4C B4F0          JMP     INITS           Initialize pointed device
```

```
F00F 4C B8F3          JMP   ISLINE        Initialization of statusline right part
F012 4C EDF3          JMP   FILLNM        Put linenumber on statusline
F015 4C 24F4          JMP   FILFNM        Put filename on statusline
F018 4C 66F4          JMP   CSLPRT        Clear right part of statusline
F01B 4C 90F4          JMP   CWHSLN        Clear whole statusline
F01E 4C A1F4          JMP   MAKSLN        Make own statusline
F021 4C BBF4          JMP   NORSTAT       Reset normal statusline
F024 4C 67F9          JMP   POSIT         Position the cursor to X,Y
F027 4C 5FF9          JMP   UNPOS         Get old cursor position back
F02A 4C 16FB          JMP   SET65         Set CRTC to 6545, vertical blank active
F02D 6C B1E7  ENEMI   JMP   [NMIVECTOR]
F030 6C B5E7  IERQU   JMP   [IRQVECTOR]


              ;*** Main Reset routine ***

F033 D8       RESET   CLD                 First clear decimal
F034 78               SEI                 No interrupts

F035 A2 0F            LDX   #$0F          Initialize DATRAM
F037 8A       1       TXA
F038 49 0F            EOR   #$0F
F03A 9D F0FF          STA   DATRAM,X
F03D CA               DEX
F03E 10 F7            BPL   1.B

F040 A9 00            LDA   #$00          Reset all variables to $00
F042 A2 51            LDX   #NDVRS+1&255  X is the loopcounter
F044 9D FFE6  2       STA   IO_VARS-1,X   Reset variable
F047 CA               DEX
F048 D0 FA            BNE   2.B           Ready?

F04A CA               DEX
F04B 9A               TXS                 Initiate the stackpointer

F04C A0 00            LDY   #$00
F04E B9 62F1  3       LDA   VART1,Y       Get offset
F051 F0 0A            BEQ   4.F           Branch if end
F053 AA               TAX
F054 B9 ADF1          LDA   VART2,Y       Get data
F057 9D 00E7          STA   IO_VARS,X     Set variable to initial value
F05A C8               INY
F05B D0 F1            BNE   3.B

F05D 20 D3F0  4       JSR   KBINIT        Keyboard init.
F060 20 2CF1          JSR   VDUINIT       Crtc init.
F063 20 2CFB          JSR   FORMFEED      Clear the screen

              ;Timer initialisation, software clock

F066 AD 0BE1          LDA   VAACR         Get old value
F069 09 40            ORA   #%01000000    Free running mode for timer 1
F06B 8D 0BE1          STA   VAACR
F06E A9 C3            LDA   #$C3          Interrupt every 1/20 second
F070 A0 50            LDY   #$50
F072 8C 04E1          STY   VATACL
F075 8D 05E1          STA   VATACH
F078 A9 C0            LDA   #$C0          Interrupts from timer 1
F07A 8D 0EE1          STA   VAIER

F07D 58               CLI                 Interrupts allowed now

F07E A2 01            LDX   #$01          Only on screen
F080 20 9DF7          JSR   PRTEXT        Print message
F083 0D492F4F20       fcc   '\rI/O 65  2.01\n\r'
F088 3635202032
F08D 2E30310A0D
F092 1B69202048       fcc   $1B,'i  HaViSoft  ',$1B,'n\n\r',0
F097 6156695361
F09C 667420201B
F0A1 6E0A0D00
F0A5 20 99F6  1       JSR   INP
F0A8 29 5F            AND   #%01011111    Accept lower and upper case
F0AA C9 42            CMP   #'B
F0AC D0 F7            BNE   1.B
F0AE 20 BFFD          JSR   BCNTR         Execute bootstrap
F0B1 4C A5F0          JMP   1.B
```

```
                ;*** By X selectable device initialization ***

F0B4 8A         INITS   TXA
F0B5 F0 1B              BEQ     2.F             Not correct input
F0B7 30 0C              BMI     1.F             Branch for input devices
F0B9 C9 09              CMP     #9
F0BB B0 15              BCS     2.F             Not correct input
F0BD 20 91F6            JSR     XTODEV
F0C0 A2 0F              LDX     #$0F            Offset for table
F0C2 4C 3DF7    3       JMP     STATHAND        Execute initialization routine
F0C5 29 7F      1       AND     #$7F
F0C7 C9 09              CMP     #9
F0C9 B0 07              BCS     2.F             Not correct input
F0CB 20 91F6            JSR     XTODEV
F0CE A2 17              LDX     #$17            Offset for table
F0D0 D0 F0              BNE     3.B
F0D2 60         2       RTS

                ;*** Keyboard initialisation subroutine ***

F0D3 A9 00      KBINIT  LDA     #$00
F0D5 8D B7E7            STA     KEYPNT
F0D8 8D 03E1            STA     VAPADD          Port A is input
F0DB AD 0CE1            LDA     VAPCR           Get old value
F0DE 09 0E              ORA     #%00001110      Set CA2 high
F0E0 29 FE              AND     #%11111110      Neg. edge CA1
F0E2 8D 0CE1            STA     VAPCR
F0E5 A9 82              LDA     #$82            Enable interrupts from CA1 only
F0E7 8D 0EE1            STA     VAIER
F0EA AD 0BE1            LDA     VAACR
F0ED 09 01              ORA     #%00000001      Latch mode port A
F0EF 8D 0BE1            STA     VAACR
F0F2 18         DUMRTS  CLC
F0F3 60                 RTS

                ;*** Centronics parallel printer initialisation subroutine ***

F0F4 A9 FF      PRINIT  LDA     #$FF
F0F6 8D 02E1            STA     VAPBDD          All outputs on port B
F0F9 AD 0CE1            LDA     VAPCR           Get old value
F0FC 09 A0              ORA     #%10100000      CB2=write handshake puls
F0FE 29 AF              AND     #%10101111      CB1=Neg. edge triggered
F100 8D 0CE1            STA     VAPCR
F103 AD 12E1            LDA     VBPBDD          Get old port B data direction register
F106 29 E7              AND     #%11100111      B3 and B4 inputs for 'select' and 'paper out'
F108 8D 12E1            STA     VBPBDD          Set data direction
F10B A9 00              LDA     #$00
F10D 8D 00E1            STA     VAPBD           Try to send $00 to device to get handshakepuls
F110 A0 C0              LDY     #$C0            Wait a while
F112 A2 FF      1       LDX     #$FF
F114 AD 0DE1    2       LDA     VAIFR           Read inter. flag register
F117 29 10              AND     #$10            Test only CB1 flag
F119 18                 CLC
F11A D0 0F              BNE     3.F             Then handshake received
F11C CA                 DEX                     Wait and
F11D D0 F5              BNE     2.B             Try again
F11F 88                 DEY                     Wait and
F120 D0 F0              BNE     1.B             Try again

                ;No handshake received, device not present!!!

F122 AD 10E7            LDA     DEVMODOUT       Get output device select bits
F125 29 BF              AND     #%10111111      Disable centronics par. output
F127 8D 10E7            STA     DEVMODOUT       Set bits
F12A 38                 SEC                     Error exit
F12B 60         3       RTS

                ;*** Initialise the CRT controller ***

F12C A2 0F      VDUINIT LDX     #15             16 registers to fill
F12E 8A         1       TXA
F12F 8D 40E1            STA     CRTCAR          Select register
F132 BD 41F1            LDA     CTB,X           Get value from table
F135 8D 41E1            STA     CRTCRF          Put value in register
F138 CA                 DEX                     Next register
F139 10 F3              BPL     1.B             Ready?
F13B E8                 INX
F13C 8E 71E7            STX     COUD
```

```
F13F 18              CLC
F140 60              RTS

                ;*** CRTC register values ***

F141 7E         CTB  fcc   $7E          HOR TOT (N-1)
F142 50              fcc   $50          HOR DISP
F143 5F              fcc   $5F          HOR SYNC POS
F144 88              fcc   $88          H/V SYNC WIDTH
F145 1E              fcc   $1E          VERT TOT (N-1)
F146 05              fcc   $05          VERT TOT ADJ
F147 19              fcc   $19          VER DISP
F148 1C              fcc   $1C          VERT SYNC POS
F149 80              fcc   $80          MODE CONTROL
F14A 09              fcc   $09          SCANLINES (N-1)
F14B 00              fcc   $00          CUR START
F14C 09              fcc   $09          CUR END
F14D 00              fcc   $00          START HI
F14E 00              fcc   $00          START LO
F14F 00              fcc   $00          CUR HI
F150 00              fcc   $00          CUR LO

                ;*** Acia initialisation ***

F151 AD 35E7    RSINIT LDA   ACCMD      Get value for commandregister
F154 8D 31E1         STA   ACIASR       First reset acia
F157 8D 32E1         STA   ACICMD       Set value in commandregister
F15A AD 34E7         LDA   ACCTL        Get value for controlregister
F15D 8D 33E1         STA   ACICTL       Set value in controlregister
F160 18              CLC
F161 60              RTS

                ;Offsettabel for variables

F162 10         VART1 fcc   DEVMODOUT
F163 11               fcc   DEVMODINP
F164 16               fcc   OUTRET
F165 17               fcc   INPRET
F166 45               fcc   INVST
F167 33               fcc   MONESC
F168 01               fcc   STATTOG
F169 72               fcc   TOUTIL
F16A 73               fcc   TOUTIH
F16B 74               fcc   TOUTL
F16C 75               fcc   TOUTH
F16D 38               fcc   TOUTF
F16E 89               fcc   DVO6VEC
F16F 8A               fcc   DVO6VEC+1
F170 8B               fcc   DVI6VEC
F171 8C               fcc   DVI6VEC+1
F172 8D               fcc   DVO7VEC
F173 8E               fcc   DVO7VEC+1
F174 8F               fcc   DVI7VEC
F175 90               fcc   DVI7VEC+1
F176 91               fcc   DVO8VEC
F177 92               fcc   DVO8VEC+1
F178 93               fcc   DVI8VEC
F179 94               fcc   DVI8VEC+1
F17A 23               fcc   DMPRAM
F17B 26               fcc   DMPRAM+3       RTS
F17C 27               fcc   LDALET
F17D 2A               fcc   LDALET+3       RTS
F17E 2B               fcc   FRWR
F17F 2E               fcc   FRWR+3         RTS
F180 2F               fcc   FRRE
F181 32               fcc   FRRE+3         RTS
F182 B5               fcc   IR0VECTOR
F183 B6               fcc   IR0VECTOR+1
F184 B1               fcc   NMIVECTOR
F185 B2               fcc   NMIVECTOR+1
F186 34               fcc   ACCTL
F187 35               fcc   ACCMD
F188 AD               fcc   START
F189 AE               fcc   START+1
F18A AF               fcc   UNRINT         Vector to unrec. interrupt
F18B B0               fcc   UNRINT+1
F18C 50               fcc   INTV1          Interruptvector 1
F18D 51               fcc   INTV1+1
```

```
F18E 52                 fcc     INTV2           Interruptvector 2
F18F 53                 fcc     INTV2+1
F190 54                 fcc     INTV3           Interruptvector 3
F191 55                 fcc     INTV3+1
F192 56                 fcc     INTV4           Interruptvector 4
F193 57                 fcc     INTV4+1
F194 58                 fcc     INTV5           Interruptvector 5
F195 59                 fcc     INTV5+1
F196 5A                 fcc     INTV6           Interruptvector 6
F197 5B                 fcc     INTV6+1
F198 5C                 fcc     INTV7           Interruptvector 7
F199 5D                 fcc     INTV7+1
F19A 5E                 fcc     INTV8           Interruptvector 8
F19B 5F                 fcc     INTV8+1
F19C 60                 fcc     INTV9           Interruptvector 9
F19D 61                 fcc     INTV9+1
F19E 62                 fcc     INTV10          Interruptvector 10
F19F 63                 fcc     INTV10+1
F1A0 64                 fcc     INTV11          Interruptvector 11
F1A1 65                 fcc     INTV11+1
F1A2 66                 fcc     INTV12          Interruptvector 12
F1A3 67                 fcc     INTV12+1
F1A4 68                 fcc     INTV13          Interruptvector 13
F1A5 69                 fcc     INTV13+1
F1A6 6A                 fcc     INTV14          Interruptvector 14
F1A7 6B                 fcc     INTV14+1
F1A8 6C                 fcc     INTV15          Interruptvector 15
F1A9 6D                 fcc     INTV15+1
F1AA 6E                 fcc     INTV16          Interruptvector 16
F1AB 6F                 fcc     INTV16+1
F1AC 00                 fcc     0               End of tabel

                ;Datatabel for variables

F1AD 80         VART2   fcc     $80             DEVMODOUT
F1AE 80                 fcc     $80             DEVMODINP
F1AF 80                 fcc     $80             OUTRET
F1B0 80                 fcc     $80             INPRET
F1B1 80                 fcc     $80             INVST
F1B2 1E                 fcc     $1E             MONESC
F1B3 82                 fcc     $82             STATTOG
F1B4 09                 fcc     (TOUTT&255)+1   TOUTIL
F1B5 08                 fcc     (TOUTT>>8)+1    TOUTIH
F1B6 09                 fcc     (TOUTT&255)+1   TOUTL
F1B7 08                 fcc     (TOUTT>>8)+1    TOUTH
F1B8 FF                 fcc     $FF             TOUTF
F1B9 1EF8               fdb     RESODEV         Default to reset output
F1BB 28F8               fdb     RESIDEV         Default to reset input
F1BD 1EF8               fdb     RESODEV         Default to reset output
F1BF 28F8               fdb     RESIDEV         Default to reset input
F1C1 1EF8               fdb     RESODEV         Default to reset output
F1C3 28F8               fdb     RESIDEV         Default to reset input
F1C5 8D                 fcc     $8D             STA DMPRAM
F1C6 60                 fcc     $60             RTS
F1C7 AD                 fcc     $AD             LDA
F1C8 60                 fcc     $60             RTS
F1C9 8D                 fcc     $8D             STA FRWR
F1CA 60                 fcc     $60             RTS
F1CB AD                 fcc     $AD             LDA FRRE
F1CC 60                 fcc     $60             RTS
F1CD 2FF2               fdb     INT             IRQ vector
F1CF 70F3               fdb     UNNMI           NMI vector
F1D1 BA                 fcc     $BA             ACCTL
F1D2 05                 fcc     $05             ACCMD
F1D3 00E8               fdb     VDURAM          START
F1D5 57F3               fdb     UNINT           Interrupt ignored vector
F1D7 83F2               fdb     INT1-1
F1D9 F1F0               fdb     INT2-1
F1DB F1F0               fdb     INT3-1
F1DD F1F0               fdb     INT4-1
F1DF F1F0               fdb     INT5-1
F1E1 16F3               fdb     INT6-1
F1E3 F1F0               fdb     INT7-1
F1E5 F1F0               fdb     INT8-1
F1E7 F1F0               fdb     INT9-1
F1E9 F1F0               fdb     INT10-1
F1EB F1F0               fdb     INT11-1
F1ED F1F0               fdb     INT12-1
```

```
F1EF F1F0             fdb     INT13-1
F1F1 F1F0             fdb     INT14-1
F1F3 27F3             fdb     INT15-1
F1F5 76F2             fdb     INT16-1

                ;*** Initialization for via I/O routines ***

F1F7 A9 EF      VOINIT LDA    #WBYTEF&255
F1F9 8D 85E7          STA     DVO4VEC
F1FC A9 F7            LDA     #WBYTEF>>8
F1FE 8D 86E7          STA     DVO4VEC+1
F201 A9 FF            LDA     #$FF            All outputs
F203 D0 0C            BNE     1.F

F205 A9 0F      VIINIT LDA    #RBYTE&255
F207 8D 87E7          STA     DVI4VEC
F20A A9 F8            LDA     #RBYTE>>8
F20C 8D 88E7          STA     DVI4VEC+1
F20F A9 00            LDA     #$00            All inputs
F211 8D 13E1    1     STA     VBPADD          Set data direction
F214 A9 F8            LDA     #$F8
F216 8D 1CE1          STA     VBPCR           Full auto handshake
F219 18               CLC
F21A 60               RTS

F21B A2 00      MDINIT LDX    #0
F21D BD 76E7    1     LDA     WRBEG,X         Copy beginaddress into selfmod. program
F220 9D 2CE7          STA     FRWR+1,X
F223 BD 77E7          LDA     WRBEG+1,X
F226 9D 2DE7          STA     FRWR+2,X
F229 18               CLC
F22A 60               RTS

F22B A2 04      MIINIT LDX    #4
F22D D0 EE            BNE     1.B

                      cont    INTERRUPT.MAC
                ;********** INTERRUPT **********

                ;*** IRQ interrupt routine ***

F22F D8         INT   CLD
F230 8D 70E7          STA     SVAINT
F233 68               PLA
F234 48               PHA                     Get old processor status
F235 29 10            AND     #%00010000
F237 F0 04            BEQ     3.F             Branch if not a software interrupt
F239 A2 0F            LDX     #$0F            Table offset for software interrupt
F23B D0 3B            BNE     INTFUN

F23D AD 70E7    3     LDA     SVAINT
F240 48               PHA
F241 8A               TXA
F242 48               PHA                     Save all registers
F243 98               TYA
F244 48               PHA
F245 AD 0DE1          LDA     VAIFR           Interrupt from VIAA?
F248 30 11            BMI     VIAAINT
F24A AD 1DE1          LDA     VBIFR           Interrupt from VIAB?
F24D 30 13            BMI     VIABINT         Branch if yes
F24F 2C 31E1    INTA  BIT     ACIASR          Interrupt from acia?
F252 10 04            BPL     1.F
F254 A2 0E            LDX     #$0E            Table offset for acia interrupt
F256 D0 17            BNE     2.F
F258 6C AFE7    1     JMP     [UNRINT]        Jump for interrupt ignored

F25B 2D 0EE1    VIAAINT AND   VAIER
F25E A2 FF            LDX     #$FF            Table offset
F260 D0 05            BNE     INTHAN
F262 2D 1EE1    VIABINT AND   VBIER
F265 A2 06            LDX     #$06            Table offset
F267 0A         INTHAN ASLA
F268 F0 08            BEQ     INTEX
F26A 18               CLC
F26B 0A         INTLOP ASLA
F26C E8               INX
F26D 90 FC            BCC     INTLOP          Branch if not this interrupt
F26F 20 78F2    2     JSR     INTFUN          Execute interrupt routine
```

```
                        ;               Return address for executed interrupt routine
F272 68         INTEX   PLA
F273 A8                 TAY
F274 68                 PLA
F275 AA                 TAX                     Restore registers
F276 68                 PLA
F277 40         DUMRTI  RTI

F278 8A         INTFUN  TXA
F279 0A                 ASLA
F27A AA                 TAX
F27B BD 51E7            LDA     INTV1+1,X       Get high byte
F27E 48                 PHA
F27F BD 50E7            LDA     INTV1,X         Get low byte
F282 48                 PHA
F283 60                 RTS                     Execute interrupt routine

                        ;Clock update interrupt routine

F284 2C 04E1    INTTIM  BIT     VATACL          Reset timer interrupt flag
F287 A9 00              LDA     #$00            Update clock
                        TIME    SEC1.20,$14
F289 AE 37E7            LDX     SEC1.20
F28C E8                 INX
F28D 8E 37E7            STX     SEC1.20
F290 E0 14              CPX     #$14
F292 D0 76              BNE     INTND
F294 8D 37E7            STA     SEC1.20
F297 CE 1BE7            DEC     VIAVRA          Timer for vialoop
F29A 2C 38E7            BIT     TOUTF
F29D 10 0D              BPL     1.F
F29F CE 74E7            DEC     TOUTL           Timeout for screen off decrements
F2A2 D0 08              BNE     1.F
F2A4 CE 75E7            DEC     TOUTH
F2A7 D0 03              BNE     1.F
F2A9 8D 38E7            STA     TOUTF           Set flag
                1       TIME    SECONDS,$3C
F2AC AE 84E7            LDX     SECONDS
F2AF E8                 INX
F2B0 8E 84E7            STX     SECONDS
F2B3 E0 3C              CPX     #$3C
F2B5 D0 53              BNE     INTND
F2B7 8D 84E7            STA     SECONDS
                        TIME    MINUTES,$3C
F2BA AE 83E7            LDX     MINUTES
F2BD E8                 INX
F2BE 8E 83E7            STX     MINUTES
F2C1 E0 3C              CPX     #$3C
F2C3 D0 45              BNE     INTND
F2C5 8D 83E7            STA     MINUTES
                        TIME    HOURS,$18
F2C8 AE 82E7            LDX     HOURS
F2CB E8                 INX
F2CC 8E 82E7            STX     HOURS
F2CF E0 18              CPX     #$18
F2D1 D0 37              BNE     INTND
F2D3 8D 82E7            STA     HOURS

                        ;End of day now

F2D6 AE 80E7            LDX     MONTH           What month is it?
F2D9 E0 02              CPX     #2
F2DB D0 0B              BNE     NOCHE

F2DD AD 81E7            LDA     YEAR            Something strange with february
F2E0 29 03              AND     #%00000011      Check for divisible by 4
F2E2 D0 04              BNE     NOCHE           Then not divisible
F2E4 A9 1D              LDA     #29             In those years febr. has 29 days
F2E6 D0 03              BNE     FEBCH           Continue elsewhere

F2E8 BD 0AF3    NOCHE   LDA     MONTAB-1,X      What day is it?
F2EB CD 7FE7    FEBCH   CMP     DAY             Check end of month
F2EE D0 12              BNE     DTUPDC          Then not on end of month

                        ;End of month now

F2F0 A9 00              LDA     #0
F2F2 8D 7FE7            STA     DAY             First day 0
```

```
F2F5 E8              INX             Next month
F2F6 E0 0D           CPX   #13       Only 12 months
F2F8 D0 05           BNE   DTUPDB     Then not next year

                ;End of year now

F2FA A2 01           LDX   #1        Start with month 1
F2FC EE 81E7         INC   YEAR
F2FF 8E 80E7 DTUPDB  STX   MONTH     Set month

F302 EE 7FE7 DTUPDC  INC   DAY       Next day
F305 A9 FF           LDA   #$FF
F307 8D 7EE7         STA   DATUPD    Set flag for date updated

F30A 60      INTND   RTS

F30B 1F      MONTAB  fcc   31        January
F30C 1C              fcc   28        February
F30D 1F              fcc   31        March
F30E 1E              fcc   30        April
F30F 1F              fcc   31        May
F310 1E              fcc   30        June
F311 1F              fcc   31        July
F312 1F              fcc   31        August
F313 1E              fcc   30        September
F314 1F              fcc   31        October
F315 1E              fcc   30        November
F316 1F              fcc   31        December

                ;*** Interrupt from keyboard ***

F317 AD 01E1 KEYINT  LDA   VAPAD     Read key from via
F31A AE B7E7 KEYINTA LDX   KEYPNT    Keyboardbuffer pointer
F31D E0 28           CPX   #MAXBUF   Max. buffer
F31F F0 06           BEQ   BUFVOL

F321 9D B8E7 KEYINTB STA   KEYBUF,X  Put character in buffer
F324 EE B7E7         INC   KEYPNT
F327 60      BUFVOL  RTS

                ;*** Acia interrupt routine ***

F328 AD 31E1 ACINT   LDA   ACIASR    Get statusregister
F32B 29 08           AND   #$08      Receiver interrupt?
F32D F0 1B           BEQ   TRMINT    Or transmitter interrupt?

                ;Receiver register full

F32F AD 30E1         LDA   RECREG    Read acia receiverregister
F332 C9 13           CMP   #$13      ^S stop for handshake?
F334 D0 06           BNE   3.F
F336 A9 FF           LDA   #$FF      Set that handshake flag
F338 8D 0EE7 2       STA   HSFLG
F33B 60              RTS

F33C C9 11   3       CMP   #$11      ^Q continue for handshake
F33E D0 04           BNE   4.F       It was a normal character
F340 A9 00           LDA   #$00      Reset handshake flag and
F342 F0 F4           BEQ   2.B       continue elsewhere
F344 AD 30E1 4       LDA   RECREG    Read character from acia
F347 4C 1AF3         JMP   KEYINTA   Continue as a keyboard input

                ;Transmit register empty

F34A AD 31E1 TRMINT  LDA   ACIASR    Get the statusregister
F34D 29 10           AND   #$10      Was it a transmitter interrupt?
F34F F0 05           BEQ   1.F       Not a transmitter interrupt
F351 A9 00           LDA   #$00
F353 8D 39E7         STA   TREMP     Reset transmitterreg. empty flag
F356 60      1       RTS


                ;It was an unrecognized interrupt!!!!

F357 A2 00   UNINT   LDX   #$00      Select active output device
F359 20 9DF7         JSR   PRTEXT    Print errormessage
F35C 070A0D4952      fcc   7,'\n\rIRQ ignored\n\r',0
F361 5120696 76E
```

```
F366 6F7265640A
F36B 0D00
F36D 4C 72F2            JMP     INTEX

                ;It was an NMI!!!!

F370 A2 00      UNNMI   LDX     #$00            Select active output device
F372 20 9DF7            JSR     PRTEXT          Print message
F375 070A0D4E4D         fcc     7,'\n\rNMI ignored\n\r',0
F37A 492069676E
F37F 6F7265640A
F384 0D00
F386 40                 RTI

                        cont    STATUSLN.MAC
                ;*********** STATUSLN **********

                ;Statusline routines

F387 20 67F9    PSSEL   JSR     POSIT           First posit the cursor
F38A 38         SELSTL  SEC                     Select local status line
F38B 6E 42E7            ROR     STATFG
F38E AD 42E7            LDA     STATFG
F391 C9 80              CMP     #$80
F393 D0 0C              BNE     1.F

F395 AE 43E7    SETINV  LDX     INVERS
F398 8E 44E7            STX     INVERSS
F39B AE 45E7            LDX     INVST
F39E 8E 43E7            STX     INVERS
F3A1 60         1       RTS

F3A2 18         UNSELS  CLC                     Unselect local status line
F3A3 2E 42E7            ROL     STATFG
F3A6 AD 42E7            LDA     STATFG
F3A9 D0 F6              BNE     1.B

F3AB AE 44E7    PRSEND  LDX     INVERSS         Reset the invers video mode
F3AE 8E 43E7            STX     INVERS
F3B1 60                 RTS

F3B2 20 A2F3    UNSLPS  JSR     UNSELS
F3B5 4C 5FF9            JMP     UNPOS

                ;       Initialize part of statusline

F3B8 08         ISLINE  PHP
F3B9 78                 SEI
F3BA 48                 PHA
F3BB 8A                 TXA
F3BC 48                 PHA
F3BD 98                 TYA
F3BE 48                 PHA
F3BF A2 32              LDX     #50
F3C1 A0 19              LDY     #25
F3C3 20 87F3            JSR     PSSEL
F3C6 A9 20              LDA     #32             Clear right part of statusline
F3C8 8D 0DE7            STA     MAXSTL
F3CB 20 85F4            JSR     CLSPRT4
F3CE 20 9DF7            JSR     PRTEXT
F3D1 1432194C6E         fcc     20,50,25,'Ln:       Fn:',0
F3D6 3A20202020
F3DB 2020202046
F3E0 6E3A00
F3E3 20 B2F3            JSR     UNSLPS
F3E6 68                 PLA
F3E7 A8                 TAY
F3E8 68                 PLA
F3E9 AA                 TAX
F3EA 68                 PLA
F3EB 28                 PLP
F3EC 60                 RTS

                ;*** Set linenumber in statusline ***

F3ED 8D 3EE7    FILLNM  STA     LNRH
F3F0 8C 3DE7            STY     LNRL
F3F3 8A                 TXA
```

```
F3F4 48              PHA
F3F5 08              PHP
F3F6 78              SEI
F3F7 A2 36           LDX   #54
F3F9 A0 19           LDY   #25
F3FB 20 87F3         JSR   PSSEL
F3FE 20 9DF7         JSR   PRTEXT         Print text
F401 2020202020      fcc   '     '        Delete old characters
F406 20
F407 14361900        fcc   20,54,25,0     Reposition cursor
F40B AD 3EE7         LDA   LNRH           Get highbyte linenumber
F40E AE 3DE7         LDX   LNRL           Get lowbyte linenumber
F411 20 02FF         JSR   CONVER         Convert 2 bytes hex to 3 bytes dec.
F414 20 41FF         JSR   PRIDEC         Print decimal numbers, only to screen
F417 20 B2F3         JSR   UNSLPS
F41A 28              PLP
F41B 68              PLA
F41C AA              TAX
F41D AD 3EE7         LDA   LNRH
F420 AC 3DE7         LDY   LNRL
F423 60              RTS

                     ;*** Put filename (pointed by A,Y) in statusline ***

F424 8C 20E7  FILFNM STY   PNTXL          Lowbyte pointer
F427 8D 21E7         STA   PNTXH          Highbyte pointer
F42A 08              PHP                  Save processorstatus
F42B 78              SEI
F42C 20 5CF6         JSR   SAVEREGS       Save all registers
F42F 20 CCFF         JSR   ADPNTX         Setup a selfmodifying program
F432 A9 0E           LDA   #14
F434 A2 41           LDX   #65
F436 A0 19    FILFL5 LDY   #25
F438 8D 0DE7  FILFL0 STA   MAXSTL
F43B 20 87F3         JSR   PSSEL
F43E 20 1FE7  FILFL1 JSR   PNTX           Get character
F441 F0 11           BEQ   FILFL2         Stop on $00
F443 C9 0D           CMP   #$0D
F445 F0 0D           BEQ   FILFL2         Stop on CR
F447 20 DDF6         JSR   OUT            Set on statusline
F44A 20 B1FF         JSR   PNTXINC        Increase pointers
F44D CE 0DE7         DEC   MAXSTL         Maximum characters for this statusline
F450 D0 EC           BNE   FILFL1         Ready?
F452 F0 0A           BEQ   FILFL4         Continue elsewhere
F454 A9 20    FILFL2 LDA   #'             Space in A
F456 20 DDF6  FILFL3 JSR   OUT            Fill rest with spaces
F459 CE 0DE7         DEC   MAXSTL
F45C D0 F8           BNE   FILFL3         Ready?
F45E 20 B2F3  FILFL4 JSR   UNSLPS
F461 20 78F6         JSR   RESTOREGS      Restore all registers
F464 28              PLP                  Also processorstatus
F465 60              RTS

                     ;*** Clear part of statusline ***

F466 08       CSLPRT PHP
F467 78              SEI
F468 20 5CF6         JSR   SAVEREGS
F46B 20 73F4         JSR   CLSPRT1
F46E 20 78F6         JSR   RESTOREGS
F471 28              PLP
F472 60              RTS

F473 A9 20    CLSPRT1 LDA  #32
F475 A2 30            LDX  #48             X position on statusline
F477 A0 19    CLSPRT3 LDY  #25
F479 8D 0DE7  CLSPRT2 STA  MAXSTL
F47C 20 87F3          JSR  PSSEL
F47F 20 85F4          JSR  CLSPRT4
F482 4C B2F3          JMP  UNSLPS

F485 A9 20    CLSPRT4 LDA  #' '
F487 20 DDF6          JSR  OUT
F48A CE 0DE7          DEC  MAXSTL
F48D D0 F6            BNE  CLSPRT4
F48F 60               RTS

                     ;*** Clear whole statusline ***
```

```
F490 08         CWHSLN  PHP
F491 78                 SEI
F492 20 5CF6            JSR     SAVEREGS
F495 A9 50              LDA     #80
F497 A2 01              LDX     #1              X,Y position
F499 20 77F4            JSR     CLSPRT3
F49C 20 78F6            JSR     RESTOREGS
F49F 28                 PLP
F4A0 60                 RTS

                ;*** Print user defined (pointed by A,Y) statusline ***

F4A1 8C 20E7    MAKSLN  STY     PNTXL           Lowbyte pointer
F4A4 8D 21E7            STA     PNTXH           Highbyte pointer
F4A7 08                 PHP                     Save processorstatus
F4A8 78                 SEI
F4A9 20 5CF6            JSR     SAVEREGS        Save all registers
F4AC 20 CCFF            JSR     ADPNTX          Setup a selfmodifying program
F4AF A9 83              LDA     #$83
F4B1 8D 01E7            STA     STATTO6
F4B4 A9 50              LDA     #80
F4B6 A2 01              LDX     #1              X,Y position
F4B8 4C 36F4            JMP     FILFL5          Continue elsewhere

F4BB 08         NORSTAT PHP
F4BC 78                 SEI
F4BD 48                 PHA
F4BE A9 82              LDA     #$82
F4C0 8D 01E7            STA     STATTO6
F4C3 A9 02              LDA     #$02
F4C5 20 CBF4            JSR     PRSTAT
F4C8 68                 PLA
F4C9 28                 PLP
F4CA 60                 RTS

                ;*** Print a statusline pointed by A ***

F4CB A8         PRSTAT  TAY                     Set flags
F4CC F0 09              BEQ     OUTPSTAT        Output device statusline
F4CE C9 01              CMP     #$01
F4D0 F0 46              BEQ     INPSTAT         Input device statusline
F4D2 C9 02              CMP     #$02
F4D4 F0 7F              BEQ     HLPSTAT         Normal help statusline
F4D6 60                 RTS

                ;*** Statusline for outputdevice selection ***

F4D7 A9 80      OUTPSTAT LDA    #$80
F4D9 8D 01E7            STA     STATTO6
F4DC 20 90F4            JSR     CWHSLN          Clear the line
F4DF A2 06              LDX     #6
F4E1 A0 19              LDY     #25
F4E3 20 87F3            JSR     PSSEL
F4E6 20 9DF7            JSR     PRTEXT          Print text
F4E9 4F75747075         fcc     'Output devices ',0
F4EE 7420646576
F4F3 6963657320
F4F8 00
F4F9 AD 10E7            LDA     DEVMODOUT       What are the active devices?
F4FC A8         ODSTA   TAY                     Keep them in Y reg.
F4FD A2 31              LDX     #'1             Start with device 1
F4FF A9 20      PRST    LDA     #' 
F501 20 DDF6            JSR     OUT             Print a space
F504 8A                 TXA                     Get device number
F505 20 DDF6            JSR     OUT             Print device number
F508 98                 TYA                     Active devices in A
F509 18                 CLC
F50A 0A                 ASLa                    Shift most significant device in carry
F50B A8                 TAY                     Keep rest of devices in Y
F50C 20 3FF5            JSR     PRONOFF         Print 'on' (C=1), 'off' (C=0)
F50F E8                 INX                     Get next device number
F510 E0 39              CPX     #'9             Device 9 doesn't exist
F512 D0 EB              BNE     PRST            Continue for rest of the devices
F514 20 A2F3            JSR     UNSELS
F517 60                 RTS

                ;*** Statusline for inputdevice selection ***
```

```
F518 A9 81      INPSTAT LDA     #$81
F51A 8D 01E7            STA     STATTOG
F51D 20 90F4            JSR     CWHSLN          Clear the line
F520 A2 07              LDX     #7
F522 A0 19              LDY     #25
F524 20 87F3            JSR     PSSEL
F527 20 9DF7            JSR     PRTEXT          Print text
F52A 496E707574         fcc     'Input devices ',0
F52F 2064657669
F534 6365732000
F539 AD 11E7            LDA     DEVMODINP       What are the active devices
F53C 4C FCF4            JMP     ODSTA           Continue elsewhere

                ;*** Print 'on' or 'off' ***

F53F 90 0A      PRONOFF BCC     PROFF           if C=0 then 'off'
F541 20 9DF7            JSR     PRTEXT          Print text
F544 3D6F6E2020         fcc     '=on  ',0
F549 00
F54A 60                 RTS
F54B 20 9DF7    PROFF   JSR     PRTEXT          Print text
F54E 3D6F666620         fcc     '=off ',0
F553 00
F554 60                 RTS

                ;*** Print normal statusline ***

F555 20 40F6    HLPSTAT JSR     SVPXY           Save X,Y cursorpointers
F558 20 90F4            JSR     CWHSLN          Clear the line
F55B A2 02              LDX     #2
F55D A0 19              LDY     #25
F55F 20 87F3            JSR     PSSEL
F562 20 9DF7            JSR     PRTEXT          Print text
F565 54696D6520         fcc     'Time ',0
F56A 00
F56B 20 83F5            JSR     PRTIM           Print the time
F56E 20 B5F5            JSR     PRDAT           Print the date
F571 20 12F6            JSR     ST2UD           Print rest of the line
F574 20 A2F3            JSR     UNSELS
F577 4C B3FD            JMP     SF1ENDD         Reset rest of registers etc.

F57A A9 2D      PRSTRP  LDA     #'-
F57C D0 02              BNE     1.F
F57E A9 3A      PRTIMB  LDA     #':
F580 4C DDF6    1       JMP     OUT             Print a '-' or a ':'

                ;*** Print the time ***

F583 20 8AF3    PRTIM   JSR     SELSTL
F586 20 02FB            JSR     VERBLANK        Wait for vertical blank
F589 20 9DF7            JSR     PRTEXT
F58C 14071900           fcc     20,7,25,0       Set pointer to Time position
F590 AD 82E7            LDA     HOURS
F593 20 8EFF            JSR     PRHD            Print the hours
F596 20 7EF5            JSR     PRTIMB          Print a ':'
F599 AD 83E7            LDA     MINUTES
F59C 20 ABF5            JSR     PRTIMA          Print the minutes
F59F 20 7EF5            JSR     PRTIMB          Print a ':'
F5A2 AD 84E7            LDA     SECONDS         Print the seconds
F5A5 20 ABF5            JSR     PRTIMA
F5A8 4C A2F3            JMP     UNSELS

                ;*** Print a hex byte as two decimal nibbles ***

F5AB 20 A2FF    PRTIMA  JSR     HEXDE           Convert hex to decimal
F5AE 20 76FF            JSR     PRNIBL          Print a nibble
F5B1 98                 TYA
F5B2 4C 76FF            JMP     PRNIBL          Print second nibble

                ;*** Print the date ***

F5B5 20 8AF3    PRDAT   JSR     SELSTL
F5B8 20 9DF7            JSR     PRTEXT          Print text
F5BB 1411194461         fcc     20,17,25,'Date ',0
F5C0 74652000
F5C4 AD 7FE7            LDA     DAY             Get day
F5C7 20 ABF5            JSR     PRTIMA          Print day
```

```
F5CA 20 7AF5            JSR     PRSTRP          Print a '-'
F5CD AD 80E7            LDA     MONTH           Get month
F5D0 20 ABF5            JSR     PRTIMA          Print month
F5D3 20 7AF5            JSR     PRSTRP          Print a'-'
F5D6 AD 81E7            LDA     YEAR            Get year
F5D9 20 ABF5            JSR     PRTIMA          Print year
F5DC 4C A2F3            JMP     UNSELS

                ;*** Update the clock and the date on the statusline ***

F5DF AD 01E7    CLCKUP  LDA     STATTOG         Get kind of statusline
F5E2 C9 82              CMP     #$82
F5E4 D0 2B              BNE     2.F             Only in this mode
F5E6 AE 84E7            LDX     SECONDS         Get seconds
F5E9 EC 18E7            CPX     CLSEC           Only once per second an update
F5EC F0 23              BEQ     2.F             Then already done this second
F5EE AC 05E7            LDY     CURPY           Get cursor Y position
F5F1 C0 18              CPY     #24             Compare with statusline position
F5F3 F0 1C              BEQ     2.F             Don't update when you're in this line
F5F5 8E 18E7            STX     CLSEC           Keep this second
F5F8 AE 04E7            LDX     CURPX           Get cursor X position
F5FB 20 5CF6            JSR     SAVEREGS        Save all registers
F5FE 20 83F5            JSR     PRTIM           Print the time
F601 2C 7EE7            BIT     DATUPD          Also update date? (Only once a day)
F604 10 08              BPL     1.F             Than no update
F606 20 B5F5            JSR     PRDAT           Update date
F609 A9 00              LDA     #$00
F60B 8D 7EE7            STA     DATUPD          Reset the flag
F60E 4C B3FD    1       JMP     SF1ENDD         Reset registers etc.
F611 60         2       RTS

                ;*** Update normal statusline ***

F612 20 8AF3    ST2UD   JSR     SELSTL
F615 20 9DF7            JSR     PRTEXT          Print text
F618 142019436F         fcc     20,32,25,'Col: ',0
F61D 6C3A2000
F621 AE 02E7            LDX     CRPX            Get old curs X pos. (current is on statusline)
F624 E8                 INX                     Because first column is 1
F625 8A                 TXA
F626 20 8EFF            JSR     PRHD            Print it

F629 20 9DF7            JSR     PRTEXT          Print text
F62C 142919526F         fcc     20,41,25,'Row: ',0
F631 773A2000
F635 AE 03E7            LDX     CRPY            Get old curs Y pos. (current is on statusline)
F638 E8                 INX                     Because first row is 1
F639 8A                 TXA
F63A 20 8EFF            JSR     PRHD            Print it
F63D 4C A2F3            JMP     UNSELS

                ;*** Save cursor pointers ***

F640 AE 04E7    SVPXY   LDX     CURPX           Get X pointer
F643 AC 05E7            LDY     CURPY           Get Y pointer
F646 4C 5CF6            JMP     SAVEREGS        Save them

                ;*** Print the old statusline ***

F649 AD 01E7    STATOUD LDA     STATTOG         What was the old mode?
F64C 29 7F              AND     #$7F            MSB is the on/off bit
F64E 20 CBF4            JSR     PRSTAT          Print the line
F651 AD 01E7            LDA     STATTOG         Was line on or off?
F654 10 03              BPL     STATO           Than line was off
F656 4C 2AFD            JMP     STATAAN         Put line on
F659 4C 35FD    STATO   JMP     STATUIT         Turn line off

                        cont    IO_ROUTS.MAC
                ;************ IO_ROUTS *************

                ;*** Save A, X and Y registers on a software stack ***

F65C 48         SAVEREGS PHA                    First save the accu
F65D 8A                 TXA                     Save X in the accu
F65E AE 00E7            LDX     SAVSTACKP       Get the software stackpointer
F661 9D 9DE7            STA     XSAVESTACK,X    Save X on xstack
F664 68                 PLA                     Get old accu
F665 9D 95E7            STA     ASAVESTACK,X    Save A on accustack
```

```
F668 98              TYA                     Get Y in accu
F669 9D A5E7         STA    YSAVESTACK,X     Save Y on ystack
F66C EE 00E7         INC    SAVSTACKP        Increase the stackpointer
F66F AD 00E7         LDA    SAVSTACKP
F672 29 07           AND    #$07
F674 8D 00E7         STA    SAVSTACKP
F677 60              RTS

                ;*** Restore A,X and Y registers from software stack ***

F678 CE 00E7    RESTOREGS DEC  SAVSTACKP       Decrease the stackpointer
F67B AD 00E7         LDA    SAVSTACKP
F67E 29 07           AND    #$07
F680 8D 00E7         STA    SAVSTACKP
F683 AA              TAX                     Get the stackpointer
F684 BC A5E7         LDY    YSAVESTACK,X     Restore Y
F687 BD 95E7         LDA    ASAVESTACK,X     Get A and
F68A 48              PHA                     save it
F68B BD 9DE7         LDA    XSAVESTACK,X     Get X
F68E AA              TAX                     Restore X
F68F 68              PLA                     Restore A
F690 60              RTS

                ;       Convert value in X to devicebit in accu

F691 38         XTODEV SEC
F692 A9 00             LDA    #$00
F694 6A         1      RORA
F695 CA                DEX
F696 D0 FC             BNE    1.B
F698 60                RTS

                ;*** Input from active input device ***

F699 20 5CF6    INP    JSR    SAVEREGS
F69C A2 00             LDX    #0
F69E 20 A7F6           JSR    INPX
F6A1 48                PHA
F6A2 20 78F6           JSR    RESTOREGS
F6A5 68                PLA
F6A6 60                RTS

F6A7 AD 11E7    INPX   LDA    DEVMODINP
F6AA 8D 15E7           STA    SDVINP           Save current active input devicebit
F6AD 8A                TXA
F6AE F0 0C             BEQ    1.F
F6B0 C9 09             CMP    #9
F6B2 F0 23             BEQ    2.F              Get evt. as.
F6B4 B0 06             BCS    1.F              Branch if no special device selected
F6B6 20 91F6           JSR    XTODEV           Convert X to devicebit in accu
F6B9 8D 11E7           STA    DEVMODINP
F6BC A2 07      1      LDX    #$07             Offset for table
F6BE AD 11E7           LDA    DEVMODINP        Which device is active?
F6C1 D0 03             BNE    INPA
F6C3 20 28F8           JSR    RESIDEV          At least 1 device must be active
F6C6 18         INPA   CLC
F6C7 0A         INPB   ASLa                    If C=1 then device is active
F6C8 E8                INX
F6C9 90 FC             BCC    INPB             Look for active device
F6CB 20 4EF7           JSR    FUNTAB           Get character from active device
F6CE AA         3      TAX
F6CF AD 15E7           LDA    SDVINP           Get old active inputdevice
F6D2 8D 11E7           STA    DEVMODINP
F6D5 8A                TXA
F6D6 60                RTS

F6D7 20 1BF9    2      JSR    GETEVTAS         Get key if depressed
F6DA 4C CEF6           JMP    3.B

                ;*** Output to active output device ***

F6DD 48         OUT    PHA
F6DE 20 5CF6           JSR    SAVEREGS
F6E1 2C 42E7           BIT    STATFG
F6E4 10 07             BPL    1.F
F6E6 68                PLA                     If local statusline selected
F6E7 20 59FB           JSR    PRVDU
F6EA 4C 78F6           JMP    RESTOREGS
```

```
F6ED 68        1       PLA
F6EE A2 00             LDX     #0
F6F0 20 F6F6           JSR     OUTX
F6F3 4C 78F6           JMP     RESTOREGS

F6F6 8D 0CE7   OUTX    STA     OUTCHR          Save character to print
F6F9 AD 10E7           LDA     DEVMODOUT       Save output devicebits
F6FC 8D 14E7           STA     SDVOUT
F6FF 8A                TXA
F700 F0 27             BEQ     3.F             No special output device selected
F702 C9 09             CMP     #9
F704 F0 0B             BEQ     2.F             Branch if on statusline
F706 B0 21             BCS     3.F             No special output device selected
F708 20 91F6           JSR     XTODEV          X to devicebit in accu
F70B 8D 10E7           STA     DEVMODOUT
F70E 4C 29F7           JMP     3.F             Perform output
F711 C0 81     2       CPY     #$81
F713 B0 14             BCS     3.F             Branch if position not correct
F715 68                PLA
F716 98                TYA
F717 AA                TAX                     X position
F718 A0 19             LDY     #25             Y position
F71A 20 83F9           JSR     PNTCUR
F71D 20 95F3           JSR     SETINV          Invers for statusline
F720 AD 0CE7           LDA     OUTCHR
F723 20 59FB           JSR     PRVDU           Print the character
F726 4C ABF3           JMP     PRSEND          Reset invers

F729 A2 FF     3       LDX     #$FF            Table offset
F72B AD 10E7           LDA     DEVMODOUT       Which device is active?
F72E D0 03             BNE     4.F
F730 20 1EF8           JSR     RESODEV         At least 1 device must be active
F733 20 3DF7   4       JSR     STATHAND        Output character with table handler routine
F736 AD 14E7           LDA     SDVOUT
F739 8D 10E7           STA     DEVMODOUT       Restore old output devicebits
F73C 60                RTS

               ;*** Table handler ***

F73D 18        STATHAND CLC
F73E 0A        1       ASLa                    If C=1 then active
F73F E8                INX                     Increase loopcounter
F740 90 FC             BCC     1.B             Next bit
F742 48                PHA                     Save current devices and counter
F743 8A                TXA
F744 48                PHA
F745 20 4EF7           JSR     FUNTAB          Execute routine from table

               ;This is the returnaddress after execution
               ;of a routine called by FUNTAB

F748 68                PLA
F749 AA                TAX                     Restore current devices and counter
F74A 68                PLA
F74B D0 F0             BNE     STATHAND        There is also another device active
F74D 60                RTS

               ;*** Execute a routine from the table ***

F74E 8A        FUNTAB  TXA
F74F 0A                ASLa                    Calculate table offset
F750 AA                TAX                     Offset in X
F751 BD 5EF7           LDA     TABEL+$01,X     Get highbyte
F754 48                PHA                     Push on stack
F755 BD 5DF7           LDA     TABEL,X         Get lowbyte
F758 48                PHA                     Push on stack
F759 AD 0CE7           LDA     OUTCHR          Get character to print for output routines
F75C 60                RTS                     Call the routine from the table

F75D 58FB      TABEL   fdb     DVO1-$01        Output devices
F75F 8BF8              fdb     DVO2-$01
F761 4BF9              fdb     DVO3-$01
F763 EBF7              fdb     DVO4-$01
F765 45F8              fdb     DVO5-$01
F767 79F8              fdb     DVO6-$01
F769 7FF8              fdb     DVO7-$01
F76B 85F8              fdb     DVO8-$01
```

```
F76D ABF8               fdb     DVI1-$01        Input devices
F76F F1F0               fdb     DVI2-$01
F771 44F9               fdb     DVI3-$01
F773 0BF8               fdb     DVI4-$01
F775 5EF8               fdb     DVI5-$01
F777 7CF8               fdb     DVI6-$01
F779 82F8               fdb     DVI7-$01
F77B 88F8               fdb     DVI8-$01

F77D F1F0               fdb     IT1-$01         Init. output devices
F77F F3F0               fdb     IT2-$01
F781 50F1               fdb     IT3-$01
F783 F6F1               fdb     IT4-$01
F785 1AF2               fdb     IT5-$01
F787 F1F0               fdb     IT6-$01
F789 F1F0               fdb     IT7-$01
F78B F1F0               fdb     IT8-$01

F78D D2F0               fdb     IP1-$01         Init. input devices
F78F F1F0               fdb     IP2-$01
F791 50F1               fdb     IP3-$01
F793 04F2               fdb     IP4-$01
F795 2AF2               fdb     IP5-$01
F797 F1F0               fdb     IP6-$01
F799 F1F0               fdb     IP7-$01
F79B F1F0               fdb     IP8-$01

F79D 2C 42E7    PRTEXT  BIT     STATFG
F7A0 30 0B              BMI     3.F             Branch if local st.line selected
F7A2 8E 09E7            STX     XTMP
F7A5 E0 09              CPX     #9
F7A7 D0 04              BNE     3.F
F7A9 88                 DEY
F7AA 8C 04E7            STY     CURPX
F7AD 68         3       PLA                     Get returnaddres
F7AE 8D 28E7            STA     LETADRL         Set pointer lowbyte
F7B1 68                 PLA
F7B2 8D 29E7            STA     LETADRH         Set pointer highbyte

F7B5 EE 28E7    2       INC     LETADRL         Increase low byte pointer
F7B8 D0 03              BNE     PRTEXTB
F7BA EE 29E7            INC     LETADRH         If necessary also increase highbyte

F7BD 20 27E7    PRTEXTB JSR     LDALET          Get character
F7C0 F0 21              BEQ     3.F             End of text?
F7C2 2C 06E7            BIT     CURP            Test for cursor addressing active
F7C5 D0 04              BNE     1.F             Branch if active
F7C7 C9 0C              CMP     #$0C            Formfeed not allowed
F7C9 F0 EA              BEQ     2.B
F7CB 2C 42E7    1       BIT     STATFG
F7CE 10 06              BPL     8.F             Branch if no local st.line selected
F7D0 20 DDF6            JSR     OUT
F7D3 4C B5F7            JMP     2.B
F7D6 AE 09E7    8       LDX     XTMP
F7D9 AC 04E7            LDY     CURPX
F7DC C8                 INY
F7DD 20 F6F6            JSR     OUTX            Output character to devices
F7E0 4C B5F7            JMP     2.B             Back in loop
F7E3 AD 29E7    3       LDA     LETADRH         Restore returnaddress highbyte
F7E6 48                 PHA
F7E7 AD 28E7            LDA     LETADRL         Restore returnaddress lowbyte
F7EA 48                 PHA
F7EB 60                 RTS                     That was all

                ;*** Via communication routines called 'viacom' ***

F7EC 6C 85E7    VIAOUT  JMP     [DVO4VEC]       Output via a vector

F7EF 8D 11E1    WBYTEF  STA     VBPAD           Write first byte without handshake

                ;Change the vector for rest of the bytes

F7F2 A9 FD              LDA     #WBYTE&255
F7F4 8D 85E7            STA     DVO4VEC
F7F7 A9 F7              LDA     #WBYTE>>8
F7F9 8D 86E7            STA     DVO4VEC+$01
F7FC 60                 RTS
```

```
                ;Write rest of the bytes with handshake

F7FD 48         WBYTE   PHA                     First save A
F7FE 20 32F8            JSR     VIALOOP         Wait for the handshake
F801 D0 04              BNE     WBYTEND         Handshake received?

                ;No handshake received

F803 68                 PLA                     Restore A
F804 4C 1EF8            JMP     RESODEV         Reset Output devices

                ;Handshake received

F807 68         WBYTEND PLA
F808 8D 11E1            STA     VBPAD           Put character in dataregister
F80B 60                 RTS

F80C 6C 87E7    VIAINP  JMP     [DVI4VEC]       Input via a vector

F80F 20 32F8    RBYTE   JSR     VIALOOP         Wait for handshake
F812 D0 06              BNE     RBYTEND         Handshake received?

                ;No handshake received

F814 20 28F8            JSR     RESIDEV         Reset Input devices
F817 A9 0D              LDA     #$0D
F819 60                 RTS

                ;Handshake received

F81A AD 11E1    RBYTEND LDA     VBPAD           Get character from dataregister
F81D 60                 RTS

                ;Reset the I/O devices

F81E AD 16E7    RESODEV LDA     OUTRET          Get default output device
F821 8D 10E7            STA     DEVMODOUT       Set output device
F824 8D 14E7            STA     SDVOUT
F827 60                 RTS

F828 AD 17E7    RESIDEV LDA     INPRET          Get default input device
F82B 8D 11E7            STA     DEVMODINP       Set input device
F82E 8D 15E7            STA     SDVINP
F831 60                 RTS

F832 A9 0A      VIALOOP LDA     #$0A            Set to 10 seconds
F834 8D 1BE7            STA     VIAVRA
F837 AD 1DE1    VIAL    LDA     VBIFR           Read via interrupt flag register
F83A 29 02              AND     #$02            Check only CA1 flag
F83C D0 07              BNE     1.F             Branch if handshake received?
F83E 2C 1BE7            BIT     VIAVRA
F841 10 F4              BPL     VIAL            Branch if time not out
F843 A9 00              LDA     #$00            Error exit, Z=0
F845 60         1       RTS                     Normal exit Z=1

                ;*** Memory as I/O device routines ***

F846 20 2BE7    MEMOUT  JSR     FRWR            Write character in memory
F849 AD 79E7            LDA     WREND+1
F84C CD 2DE7            CMP     FRWRH           Test for end with highbyte
F84F D0 0B              BNE     1.F             Then not yet on end
F851 AD 78E7            LDA     WREND           Else also
F854 CD 2CE7            CMP     FRWRL           test for end with lowbyte
F857 D0 03              BNE     1.F             Then not on end yet
F859 4C 1EF8            JMP     RESODEV         Reset the Output devices
F85C 4C BAFF    1       JMP     FRWRINC         Increase the pointers

F85F 20 2FE7    MEMINP  JSR     FRRE            Read character from memory
F862 48                 PHA                     Save it
F863 AD 7DE7            LDA     REEND+1
F866 CD 31E7            CMP     FRREH           Test for end with highbyte
F869 D0 0B              BNE     1.F             Then not yet on end
F86B AD 7CE7            LDA     REEND           Else also
F86E CD 30E7            CMP     FRREL           test for end with lowbyte
F871 D0 03              BNE     1.F             Then not on end yet
F873 20 28F8            JSR     RESIDEV         Reset the Input devices
F876 68         1       PLA
```

```
F877 4C C3FF          JMP     FRREINC         Increase pointers

                ;*** Indirect jumps to user defined I/O devices ***

F87A 6C 89E7    FREEO1  JMP     [DVO6VEC]       Output device 6
F87D 6C 8BE7    FREEI1  JMP     [DVI6VEC]       Input device 6
F880 6C 8DE7    FREEO2  JMP     [DVO7VEC]       Output device 7
F883 6C 8FE7    FREEI2  JMP     [DVI7VEC]       Input device 7
F886 6C 91E7    FREEO3  JMP     [DVO8VEC]       Output device 8
F889 6C 93E7    FREEI3  JMP     [DVI8VEC]       Input device 8

                ;*** Centronics printer output routine ***

F88C 48         PROUT   PHA                     Save the character
F88D AD 10E1    PROUTA  LDA     VBPBD           Check SELECT
F890 29 08              AND     #$08            Printer selected?
F892 F0 F9              BEQ     PROUTA          No, then wait
F894 AD 10E1            LDA     VBPBD           Check PE
F897 29 10              AND     #$10            Paper empty?
F899 D0 F2              BNE     PROUTA          Yes, then wait
F89B AD 0DE1    WACHTACK LDA    VAIFR           Wait for acknowledge
F89E 29 10              AND     #$10            Check only CB1 flag
F8A0 F0 F9              BEQ     WACHTACK        Wait until set
F8A2 68                 PLA                     Restore the character to print
F8A3 8D 00E1            STA     VAPBD           Send character and reset CB1 and CB2 flags
F8A6 60         PRITEND RTS

F8A7 A9 FF      2       LDA     #$FF            Set flag
F8A9 8D 19E7            STA     FUNENA          Then next character is function key

                ;*** Get a char. from the keyboardbuffer, check for functions ***

F8AC 20 CBF8    GETASKEY JSR    GETKEY          Get a key from buffer
F8AF 2C 19E7            BIT     FUNENA          Test for function flag
F8B2 10 0B              BPL     1.F
F8B4 A2 00              LDX     #$00            It was a function key
F8B6 8E 19E7            STX     FUNENA          Reset the flag
F8B9 20 FDFC            JSR     FUNTOETSEN      Perform functionkey routine
F8BC 4C ACF8            JMP     GETASKEY        Continue after function execution
F8BF 2C 1AE7    1       BIT     FNCEN           Are function keys allowed?
F8C2 10 01              BPL     3.F             Branch if allowed
F8C4 60                 RTS                     Return with character in accu
F8C5 CD 33E7    3       CMP     MONESC          Was it the entry to a functionkey?
F8C8 F0 DD              BEQ     2.B             Check normal/functionentry
F8CA 60                 RTS                     It was a normal key

                ;*** Get a character from keyboardbuffer ***

F8CB AD 01E7    GETKEY  LDA     STATTOG         Check statusline mode
                CASE2   $82,GTKY                Do I have to update the statusline?
F8CE C9 82              CMP     #$82
F8D0 D0 19              BNE     GTKY
F8D2 AC 05E7            LDY     CURPY           Get the Y cursorpointer
F8D5 C0 18              CPY     #24             In statusline yet?
F8D7 F0 12              BEQ     GTKY            Then don't update statusline now
F8D9 AE 04E7            LDX     CURPX           Get also X cursorpointer
F8DC 8E 02E7            STX     CRPX            Copy pointers to new variables
F8DF 8C 03E7            STY     CRPY            Because pointers change in next calls
F8E2 20 5CF6            JSR     SAVEREGS        Save old position
F8E5 20 12F6            JSR     ST2UD           Update the statusline now
F8E8 20 B3FD            JSR     SF1ENDD         Restore pointers etc.
F8EB 20 20FB    GTKY    JSR     CURSOUD         Get old cursor mode
F8EE 20 DFF5    GTKYA   JSR     CLCKUP          Update the time every second
F8F1 20 21F9            JSR     TOFFSC          Screen off if time out
F8F4 AC B7E7            LDY     KEYPNT          Get keyboardbuffer pointer
F8F7 F0 F5              BEQ     GTKYA           Buffer empty, so wait
F8F9 20 2FF9            JSR     TONSC           Turn screen on
F8FC 20 1CFB            JSR     CURSUIT         No cursor
F8FF AD B8E7    GTKYB   LDA     KEYBUF          Get character from buffer
F902 48                 PHA                     Save it
F903 08                 PHP                     Save status
F904 78                 SEI                     No interrupts now
F905 A2 01              LDX     #$01            Last position possible
F907 EC B7E7    GTKYD   CPX     KEYPNT          What is current position
F90A F0 09              BEQ     GTKYC           That was the end
F90C BD B8E7            LDA     KEYBUF,X        Get first character
F90F 9D B7E7            STA     KEYBUF-$01,X    Shift one position to the left
F912 E8                 INX                     New position
```

```
F913 D0 F2              BNE     GTKYD           Jump always
F915 CE B7E7    GTKYC   DEC     KEYPNT          Current position
F918 28                 PLP                     Interrupts are allowed now
F919 68                 PLA                     Restore character
F91A 60                 RTS

                ;*** Get character from buffer if there is one ***

F91B AD B7E7    GETEVTAS LDA    KEYPNT          Get pointer
F91E D0 DF              BNE     GTKYB           Continue elsewhere if there was one
F920 60                 RTS                     Buffer empty, return with A=00

                ;*** Screen off if time out ***

F921 AD 38E7    TOFFSC  LDA     TOUTF
F924 D0 08              BNE     1.F             Branch if time not out yet
F926 A2 01      2       LDX     #1              Accu was 0
F928 8E 40E1            STX     CRTCAR
F92B 8D 41E1            STA     CRTCRF          Turn off screen
F92E 60         1       RTS

                ;*** Turn screen on after depressing key ***

F92F 08         TONSC   PHP
F930 78                 SEI
F931 A2 01              LDX     #1
F933 BD 72E7    3       LDA     TOUTIL,X        Re-initialize the timer
F936 9D 74E7            STA     TOUTL,X
F939 CA                 DEX
F93A 10 F7              BPL     3.B
F93C 8E 38E7            STX     TOUTF           Set flag off
F93F 28                 PLP
F940 AD 42F1            LDA     CTB+1           Get register value
F943 D0 E1              BNE     2.B

                ;*** RS232 input routine ***

F945 AD B7E7    RS232I  LDA     KEYPNT          Get pointer in keyboard
F948 F0 FB              BEQ     RS232I          Buffer empty, so wait
F94A D0 B3              BNE     GTKYB           Continue elsewhere

                ;*** RS232 output routine with ^S and ^Q handshake ***

F94C 2C 39E7    RS232O  BIT     TREMP           Wait for transmit register empty
F94F 30 FB              BMI     RS232O          Then not empty yet
F951 2C 0EE7    RSO1    BIT     HSFLG           Stop for handshake?
F954 30 FB              BMI     RSO1            Wait until control Q is received
F956 8D 30E1            STA     TRAREG          Transmit character
F959 A9 FF              LDA     #$FF
F95B 8D 39E7            STA     TREMP           Set transmit reg. full flag
F95E 60                 RTS

                cont    VDU_ROUTS.MAC
                ;************ VDU_ROUTS *************

F95F AE 0AE7    UNPOS   LDX     XPSOLD          Place cursor to position before
F962 E8                 INX
F963 AC 0BE7            LDY     YPSOLD          the POSIT call
F966 C8                 INY
F967 48         POSIT   PHA                     Place cursor on X,Y position
F968 AD 04E7            LDA     CURPX
F96B 8D 0AE7            STA     XPSOLD
F96E AD 05E7            LDA     CURPY
F971 8D 0BE7            STA     YPSOLD
F974 A9 80      1       LDA     #$80
F976 8D 06E7            STA     CURP
F979 8A                 TXA
F97A 20 59FB            JSR     PRVDU
F97D 98                 TYA
F97E 20 59FB            JSR     PRVDU
F981 68                 PLA
F982 60                 RTS

F983 48         PNTCUR  PHA
F984 4C 74F9            JMP     1.B

F987 29 EF      INRNG   AND     #VDURAM+$700)>8 Maximum
F989 09 08              ORA     #$08            This bit is always set
```

```
F98B 60              RTS

                ;*** Calculate DUMPH and DUMPL from CURPX and CURPY ***

F98C 18         CNVXYD  CLC                     Prepare addition
F98D AD ADE7            LDA     START           Get lowbyte START (= position 0,0)
F990 6D 04E7            ADC     CURPX           Add X position
F993 8D 24E7            STA     DUMPL           Temporary result in DUMPL
F996 AD AEE7            LDA     START+1         Get highbyte START
F999 69 00              ADC     #0              Add the carry, if there is one
F99B 8D 25E7            STA     DUMPH           Temporary result in DUMPH

                ;Add number of lines * 80 addresses

F99E AC 05E7            LDY     CURPY           What is the current line?
F9A1 18                 CLC
F9A2 F0 11      3       BEQ     2.F             Jump if on line 0
F9A4 AD 24E7            LDA     DUMPL           For every line add 80
F9A7 69 50              ADC     #80             80 characters per line
F9A9 8D 24E7            STA     DUMPL           Result is DUMPL
F9AC 90 04              BCC     1.F             Add carry in DUMPH
F9AE 18                 CLC
F9AF EE 25E7            INC     DUMPH           Increase DUMPH after carry
F9B2 88         1       DEY                     To next line
F9B3 90 ED              BCC     3.B             Branch always
F9B5 AD 25E7    2       LDA     DUMPH           Get DUMPH
F9B8 20 87F9            JSR     INRNG
F9BB 8D 25E7            STA     DUMPH           Result is DUMPH
F9BE 60                 RTS

                ;*** Calculate DUMPH and DUMPL, then convert ***
                ;*** DUMPL/H to cursorregister values in CRT ***

F9BF 20 8CF9    CNVCLH  JSR     CNVXYD          First calculate DUMPL/H
                ;Next routine follows automatically

                ;*** Convert DUMPL/H to cursorregister values in CRT ***

F9C2 A0 0F      DLHCLH  LDY     #$0F            Number of register in CRT
F9C4 8C 40E1            STY     CRTCAR          Select cur lo
F9C7 AD 24E7            LDA     DUMPL           Get DUMPL value
F9CA 8D 41E1            STA     CRTCRF          Dumpl in cur lo
F9CD CD ADE7            CMP     START           16 bit substr.
F9D0 AD 25E7            LDA     DUMPH           Then highbyte
F9D3 AA                 TAX
F9D4 ED AEE7            SBC     START+1         Highbyte START
F9D7 90 04              BCC     1.F             If START>DUMP then normal exit
F9D9 8A                 TXA
F9DA 29 07              AND     #$07
F9DC 24                 fcc     $24             Skip next instruction
F9DD 8A         1       TXA                     Get DUMPH for normal exit
F9DE 29 0F              AND     #$0F            Strip 5 most significant bits
F9E0 88                 DEY
F9E1 8C 40E1            STY     CRTCAR          Select cur hi
F9E4 8D 41E1            STA     CRTCRF          DUMPH in cur hi
F9E7 60                 RTS

                ;*** Scroll in opposit direction (down) ***

F9E8 38         SCRLDWN SEC
F9E9 A5 00      1       LDA     PTA
F9EB 48                 PHA
F9EC A5 01              LDA     PTA+1
F9EE 48                 PHA
F9EF A5 02              LDA     PTB
F9F1 48                 PHA
F9F2 A5 03              LDA     PTB+1
F9F4 48                 PHA
F9F5 90 43              BCC     2.F
F9F7 AD ADE7            LDA     START           Calculate new START address
F9FA AA                 TAX                     Save old START lowbyte
F9FB E9 50              SBC     #80             (old START-80=new START) address
F9FD 8D ADE7            STA     START
FA00 8D 24E7            STA     DUMPL           This value also in DUMPL, writepos. is 0,0 !!
FA03 48                 PHA                     Save START low for CRT register
FA04 AD AEE7            LDA     START+1         Get highbyte
FA07 A8                 TAY                     Save old START highbyte
FA08 E9 00              SBC     #0              Substract the carry
```

```
FA0A 20 87F9          JSR    INRNG
FA0D 8D AEE7          STA    START+1        Result is new START highbyte
FA10 8D 25E7          STA    DUMPH          And also new DUMP highbyte
FA13 29 07            AND    #$07           Strip for crtc
FA15 48               PHA                   Save START high for CRT register

                ;Calculate statusline position: START+$0780

FA16 18               CLC
FA17 8A               TXA                   Get old START lowbyte
FA18 69 80            ADC    #$80           Calculate lowbyte
FA1A 85 00            STA    PTA            Result in zeropage pointer
FA1C 98               TYA                   Get old START highbyte
FA1D 69 07            ADC    #$07           Calculate highbyte
FA1F 20 87F9          JSR    INRNG
FA22 85 01            STA    PTA+1          Result in zeropage pointer

                ;Calculate last line position: START+$0730

FA24 18               CLC
FA25 8A               TXA                   Get old START lowbyte
FA26 69 30            ADC    #$30           Calculate lowbyte
FA28 85 02            STA    PTB            Result in zeropage pointer
FA2A 98               TYA                   Get old START highbyte
FA2B 69 07            ADC    #$07           Calculate highbyte
FA2D 20 87F9          JSR    INRNG
FA30 85 03            STA    PTB+1          Result in zeropage pointer

FA32 A0 00            LDY    #0
FA34 4C 77FA          JMP    CPSTLL         Continue elsewhere

                ;*** Scroll the screen up one line ***

FA37 18         SCROLL CLC
FA38 90 AF             BCC    1.B
FA3A AD ADE7    2      LDA    START          Calculate new START address
FA3D AA                TAX                   Save old START address lowbyte
FA3E 69 50             ADC    #80            (old START+80=new START) address
FA40 8D ADE7           STA    START
FA43 48                PHA                   Save START low for CRT register
FA44 AD AEE7           LDA    START+1        Get high byte
FA47 A8                TAY                   Save old START highbyte
FA48 69 00             ADC    #0             Add the carry if there was one
FA4A 20 87F9           JSR    INRNG
FA4D 8D AEE7           STA    START+1        Result is new START highbyte
FA50 29 07             AND    #$07           Strip for crtc
FA52 48                PHA                   Save START high for CRT register

                ;Calculate last line position: START+$0780

FA53 18               CLC
FA54 8A               TXA                   Get old START lowbyte
FA55 69 80            ADC    #$80           Calculate lowbyte
FA57 85 00            STA    PTA            Result in zeropage pointer
FA59 8D 24E7          STA    DUMPL          Also in new DUMPL
FA5C 98               TYA                   Get old START highbyte
FA5D 69 07            ADC    #$07           Calculate highbyte
FA5F 20 87F9          JSR    INRNG
FA62 85 01            STA    PTA+1          Result in zeropage pointer
FA64 8D 25E7          STA    DUMPH          Also in DUMPH, writepos. is 0,24!!!

                ;Calculate statusline position :START+$07D0

FA67 18               CLC
FA68 8A               TXA                   Get old START lowbyte
FA69 69 D0            ADC    #$D0           Calculate lowbyte
FA6B 85 02            STA    PTB            Result in zeropage pointer
FA6D 98               TYA                   Get old START highbyte
FA6E 69 07            ADC    #$07           Calculate highbyte
FA70 20 87F9          JSR    INRNG
FA73 85 03            STA    PTB+1          Result in zeropage pointer

FA75 A0 17            LDY    #23
FA77 A2 00      CPSTLL LDX    #0             Set X,Y cursorpointer
FA79 8E 04E7           STX    CURPX
FA7C 8C 05E7           STY    CURPY

                ;Copy calculated values in CRT registers
```

```
FA7F A0 00              LDY     #0
FA81 A2 0C              LDX     #$0C            Number of START HI register
FA83 20 02FB            JSR     VERBLANK        Wait for vertical blank if CRT=6545
FA86 8E 40E1            STX     CRTCAR          Select START HI register
FA89 68                 PLA                     Get value
FA8A 8D 41E1            STA     CRTCRF          First set START HI
FA8D E8                 INX                     Number of next register
FA8E 8E 40E1            STX     CRTCAR          Select START LO register
FA91 68                 PLA                     Get value
FA92 8D 41E1            STA     CRTCRF          Then set START LO

                ;Copy statusline into last line on screen

FA95 A2 50              LDX     #$50            Only 80 characters
FA97 B1 00      MVR     LDA     [PTA],Y         Get statusline character
FA99 91 02              STA     [PTB],Y
FA9B A9 20              LDA     #' '            Get space character
FA9D 91 00              STA     [PTA],Y         Fill old statusline with spaces
FA9F E6 00              INC     PTA             Increase statuslinepointer lowbyte
FAA1 F0 16              BEQ     HGA             If zero then also highbyte
FAA3 E6 02      HGC     INC     PTB             Increase lastlinepointer lowbyte
FAA5 F0 1D              BEQ     HGB             If zero also highbyte
FAA7 CA         HGD     DEX                     Ready?
FAA8 D0 ED              BNE     MVR             If not all characters done then continue

FAAA 68                 PLA
FAAB 85 03              STA     PTB+1
FAAD 68                 PLA
FAAE 85 02              STA     PTB
FAB0 68                 PLA
FAB1 85 01              STA     PTA+1
FAB3 68                 PLA
FAB4 85 00              STA     PTA
FAB6 4C C2F9            JMP     DLHCLH          Calculate new cursoraddress and ready!!!

FAB9 E6 01      HGA     INC     PTA+1           Increase also highbyte
FABB A5 01              LDA     PTA+1
FABD 20 87F9            JSR     INRNG
FAC0 85 01              STA     PTA+1           Store new value
FAC2 D0 DF              BNE     HGC             Branch always
FAC4 E6 03      HGB     INC     PTB+1           Increase also highbyte
FAC6 A5 03              LDA     PTB+1
FAC8 20 87F9            JSR     INRNG
FACB 85 03              STA     PTB+1           Store new value
FACD D0 D8              BNE     HGD             Branch always

                ;*** Place cursor on next position, eventually ***
                ;*** new line or scroll.                      ***

FACF EE 04E7    CURVER  INC     CURPX           First increase X pointer
FAD2 AE 04E7            LDX     CURPX           Get X pointer
FAD5 E0 50              CPX     #80             Was it the last position
FAD7 D0 12              BNE     UITCRV          If not then normal next position
FAD9 A2 00              LDX     #$00            Else goto first position on next line
FADB 8E 04E7            STX     CURPX           Store 00 in pointer
FADE EE 05E7            INC     CURPY           Increase then also Y position
FAE1 AC 05E7            LDY     CURPY           Check if it was the last line
FAE4 C0 18              CPY     #24             Only 24 lines on 1 screen
FAE6 D0 03              BNE     UITCRV          Normal next position now
FAE8 4C 37FA            JMP     SCROLL          It was the last line so scroll!!!

                ;*** Calculate new DUMP and set cursor ***

FAEB EE 24E7    UITCRV  INC     DUMPL           Increase DUMP address
FAEE D0 0F              BNE     UITCRVA         If zero then also highbyte
FAF0 EE 25E7            INC     DUMPH           Increase highbyte
FAF3 AD 25E7            LDA     DUMPH           Get highbyte
FAF6 C9 F0              CMP     #VDURAM+$0800>>8 Watch out for maximum
FAF8 D0 05              BNE     UITCRVA         Jump if not over maximum
FAFA A9 E8              LDA     #VDURAM>>8      Reset DUMPH
FAFC 8D 25E7            STA     DUMPH           Put value in register
FAFF 4C C2F9    UITCRVA JMP     DLHCLH          Set cursor and ready

                ;*** Wait for a vertical blank, only for 6545 ***

FB02            VERBLANK
FB02 2C 46E7            BIT     CHKCRT          $00 for 6845
```

```
FB05 F0 0E            BEQ     1,F
FB07 AD 40E1          LDA     CRTCAR          Get statusregister
FB0A 29 20            AND     #$20            Check only vertical blank bit
FB0C D0 F4            BNE     VERBLANK        Wait until it is zero
FB0E AD 40E1  VRBLK   LDA     CRTCAR          Get statusregister
FB11 29 20            AND     #$20            Check vert. blank bit again
FB13 F0 F9            BEQ     VRBLK           Wait until it is set
FB15 60       1       RTS                     Now you have the whole vert. blank time

                      ;*** Set flag to 6545 crt controller ***

FB16 A9 FF    SET65   LDA     #$FF
FB18 8D 46E7          STA     CHKCRT
FB1B 60               RTS

                      ;*** Set no cursor ***

FB1C A9 20    CURSUIT LDA     #$20            Value for no cursor
FB1E D0 03            BNE     CURSZET         Continue elsewhere

                      ;*** Restore old cursor mode ***

FB20 AD 71E7  CURSOUD LDA     COUD            Get old cursor mode

                      ;*** Send cursor mode to CRT ***

FB23 A2 0A    CURSZET LDX     #$0A            Get number of register
FB25 8E 40E1          STX     CRTCAR          Select CUR START register
FB28 8D 41E1          STA     CRTCRF          Place value in register
FB2B 60               RTS

                      ;*** Clear screen subroutine ***

FB2C A9 20    FORMFEED LDA    #'              Get space character
FB2E A2 00            LDX     #$00            Reset counter
FB30 9D 00E8  FFBACK  STA     VDURAM,X        Place spaces direct in videoram
FB33 9D 00E9          STA     VDURAM+$0100,X
FB36 9D 00EA          STA     VDURAM+$0200,X
FB39 9D 00EB          STA     VDURAM+$0300,X
FB3C 9D 00EC          STA     VDURAM+$0400,X
FB3F 9D 00ED          STA     VDURAM+$0500,X
FB42 9D 00EE          STA     VDURAM+$0600,X
FB45 9D 00EF          STA     VDURAM+$0700,X
FB48 CA               DEX                     Decrease counter for next 256 bytes
FB49 D0 E5            BNE     FFBACK          Ready?
FB4B 20 49F6          JSR     STATOUD         Restore statusline, it was destroyed

                      ;*** Cursor to home position ***

FB4E A9 00    CURSHOME LDA    #$00            Reset X,Y cursor pointer
FB50 8D 04E7          STA     CURPX
FB53 8D 05E7          STA     CURPY
FB56 4C BFF9          JMP     CNVCLH          Calculate DUMP and place cursor

                      ;*** Print a character only on the screen ***
                      ;*** Test for direct cursor addressing    ***
                      ;*** Place cursor on X,Y position         ***

FB59 48       PRVDU   PHA
FB5A 20 5CF6          JSR     SAVEREGS
FB5D 68               PLA
FB5E 20 64FB          JSR     PRVDU1
FB61 4C 78F6          JMP     RESTOREGS

FB64 AA       PRVDU1  TAX                     Save character in X
FB65 AD 06E7          LDA     CURP            Check direct cursor adressing flag
FB68 F0 39            BEQ     PRINTKAR        Normal character
FB6A C9 FF            CMP     #$FF            If CURP=FF then error took place
FB6C F0 20            BEQ     UITPOINT        To error exit
FB6E 2C 06E7          BIT     CURP            Test for pointers
FB71 30 02            BMI     XPOINTER        It was a X pointer
FB73 70 13            BVS     YPOINTER        It was a Y pointer

FB75 CA       XPOINTER DEX                    Value minus 1
FB76 8A               TXA
FB77 C9 50            CMP     #80             Max X pointer is 80
FB79 30 06            BMI     XOKE
FB7B A9 FF            LDA     #$FF            Error X was to large
```

```
FB7D 8D 06E7          STA    CURP          Set error flag
FB80 60               RTS                  X-error exit

FB81 8D 07E7  XOKE    STA    SCURPX        Save X pointer
FB84 4E 06E7          LSR    CURP          Set flag to Y pointer
FB87 60               RTS                  Get next pointer

FB88 CA       YPOINTER DEX                 Value minus 1
FB89 8A               TXA
FB8A C9 19            CMP    #25           Max Y pointer is 25
FB8C 30 06            BMI    YOKE

              ;Normal and error exit

FB8E A9 00    UITPOINT LDA   #0            Error Y was to large
FB90 8D 06E7          STA    CURP          Reset curp
FB93 60               RTS                  Y-error exit

              ;Both pointers correct

FB94 8D 05E7  YOKE    STA    CURPY         Set Y pointer
FB97 AD 07E7          LDA    SCURPX        Get X pointer
FB9A 8D 04E7          STA    CURPX         Set X pointer
FB9D 20 BFF9          JSR    CNVCLH        Calculate DUMP and place cursor
FBA0 4C 8EFB          JMP    UITPOINT      Reset curp

              ;*** Compute the character and print it ***

FBA3 8A       PRINTKAR TXA                 Restore character in A
FBA4 C9 20            CMP    #$20          Check for control characters
FBA6 90 3B            BCC    CONTROL       Branch if control
FBA8 2C 41E7          BIT    ESCFLG        Is the escape flag set?
FBAB 10 26            BPL    PRTKRA        Branch if not set
FBAD C9 69            CMP    #'i           Invers video?
FBAF D0 0B            BNE    PRTKRB        Branch if not invers
FBB1 A9 80            LDA    #$80          Get value
FBB3 8D 43E7  PRTKD   STA    INVERS        Set invers
FBB6 A9 00    PRTKRG  LDA    #$00          Reset escape flag
FBB8 8D 41E7          STA    ESCFLG
FBBB 60               RTS

FBBC C9 6E    PRTKRB  CMP    #'n           Check for reset invers video
FBBE D0 04            BNE    PRTKRE        Branch if not
FBC0 A9 00            LDA    #0            Reset invers video
FBC2 F0 EF            BEQ    PRTKD         Branch always

FBC4 C9 46    PRTKRE  CMP    #'F           Check for set grafics
FBC6 F0 06            BEQ    PRTKRI        Branch if yes
FBC8 C9 47            CMP    #'G           Reset grafics
FBCA D0 EA            BNE    PRTKRG        Branch if not
FBCC A9 00            LDA    #0            Reset grafics
FBCE 8D 1CE7  PRTKRI  STA    GRFLG
FBD1 F0 E3            BEQ    PRTKRG        Branch always

FBD3 2C 1CE7  PRTKRA  BIT    GRFLG         Grafics enabled?
FBD6 F0 02            BEQ    PRTKRH        Branch if disabled
FBD8 29 1F            AND    #$1F          Make characters <$32
FBDA 0D 43E7  PRTKRH  ORA    INVERS        Eventually invers video
FBDD 20 23E7          JSR    DMPRAM
FBE0 4C CFFA          JMP    CURVER        Place cursor on next position

              ;*** Compute the control characters ***

FBE3 C9 07    CONTROL CMP    #$07
FBE5 D0 10            BNE    1.F

              ;*** Ring the bell ***

FBE7 AD 1CE1  PIEPER  LDA    VBPCR
FBEA 29 DF            AND    #%11011111    PCR 5 :=0
FBEC 09 C0            ORA    #%11000000    PCR 7 and 6 :=1
FBEE 8D 1CE1          STA    VBPCR         CB2 low
FBF1 09 E0            ORA    #%11100000    PCR 7,6,5 :=1
FBF3 8D 1CE1          STA    VBPCR         CB2 High
FBF6 60               RTS

FBF7 C9 08    1       CMP    #$08
FBF9 D0 20            BNE    2.F
```

```
                        ;*** Cursor 1 column back

FBFB CE 04E7    BCKSPCA DEC     CURPX           Decrease X pointer
FBFE AE 04E7            LDX     CURPX
FC01 E0 FF              CPX     #$FF            One line up?
FC03 D0 13              BNE     BCKSPCC         Branch if not begin of line
FC05 AC 05E7            LDY     CURPY           Is it allowed another line up?
FC08 D0 06              BNE     BCKSPCB         Branch if allowed
FC0A 20 E8F9            JSR     SCRLDWN         Otherwise scroll down
FC0D 4C 13FC            JMP     BCKSPCD         Continue elsewhere
FC10 CE 05E7    BCKSPCB DEC     CURPY           One line up
FC13 A2 4F      BCKSPCD LDX     #79             Get pointer to last column
FC15 8E 04E7            STX     CURPX           Place cursor in last column
FC18 4C BFF9    BCKSPCC JMP     CNVCLH          Calculate DUMP, place cursor

FC1B C9 09      2       CMP     #$09
FC1D D0 0B              BNE     3.F

                        ;*** Normal tab ***

FC1F 20 CFFA    TAB     JSR     CURVER          One place to the right
FC22 AD 04E7            LDA     CURPX           Check position
FC25 29 07              AND     #$07            Tabs are on every 8th position
FC27 D0 F6              BNE     TAB             Ready?
FC29 60                 RTS

FC2A C9 0A      3       CMP     #$0A
FC2C D0 16              BNE     4.F

                        ;*** Linefeed ***

FC2E AE 04E7    LINEFEED LDX    CURPX           Get current X position
FC31 8E 07E7            STX     SCURPX          Save it
FC34 EE 05E7            INC     CURPY           To next line
FC37 AC 05E7            LDY     CURPY           Get next line number
FC3A C0 18              CPY     #24             Not to high number?
FC3C D0 23              BNE     VTUIT           No
FC3E 20 37FA            JSR     SCROLL          Scroll now
FC41 4C 5BFC            JMP     VTTA            Continue elsewhere

FC44 C9 0B      4       CMP     #$0B
FC46 D0 1C              BNE     5.F

                        ;*** One line up ***

FC48 AE 04E7    VERTTAB LDX     CURPX           Get current X position
FC4B 8E 07E7            STX     SCURPX          Save it
FC4E CE 05E7            DEC     CURPY           To previous line
FC51 AC 05E7            LDY     CURPY           Get that linenumber
FC54 C0 FF              CPY     #$FF            Not to low number?
FC56 D0 09              BNE     VTUIT           No
FC58 20 E8F9            JSR     SCRLDWN         So scroll down now
FC5B AE 07E7    VTTA    LDX     SCURPX          Get save X pointer
FC5E 8E 04E7    VTTB    STX     CURPX           Restore X
FC61 4C BFF9    VTUIT   JMP     CNVCLH          Calculate DUMP and place cursor

FC64 C9 0C      5       CMP     #$0C
FC66 D0 03              BNE     6.F
FC68 4C 2CFB            JMP     FORMFEED        Clear screen

FC6B C9 0D      6       CMP     #$0D
FC6D D0 04              BNE     7.F

                        ;*** Carriage return ***

FC6F A2 00      CARRET  LDX     #$00            X position is 0
FC71 F0 EB              BEQ     VTTB            Branch always

FC73 C9 19      7       CMP     #$19
FC75 D0 4F              BNE     8.F

                        ;*** Erase to end of screen ***

FC77 A5 00      EEOS    LDA     PTA
FC79 48                 PHA
FC7A A5 01              LDA     PTA+1
FC7C 48                 PHA
```

```
FC7D AD 24E7            LDA     DUMPL
FC80 85 00              STA     PTA
FC82 AD 25E7            LDA     DUMPH
FC85 85 01              STA     PTA+1

                ;Calculate end of screen address: START+$077F

FC87 18                 CLC
FC88 AD ADE7            LDA     START           Get START lowbyte
FC8B 69 7F              ADC     #$7F
FC8D 8D 3FE7            STA     COMP            Result lowbyte
FC90 AD AEE7            LDA     START+1         Get START highbyte
FC93 69 07              ADC     #$07
FC95 20 87F9            JSR     INRNG
FC98 8D 40E7            STA     COMP+1          Result highbyte

FC9B A0 00              LDY     #0
FC9D A9 20      EOSLPA  LDA     #' '            Get space character
FC9F 91 00      EOSLP   STA     [PTA],Y         Clear one place

                ;Check if ready

FCA1 A6 00              LDX     PTA             Low byte of pointer
FCA3 EC 3FE7            CPX     COMP            Equal to compare address
FCA6 D0 0E              BNE     INCRM           If not, continue
FCA8 A6 01              LDX     PTA+1           Else check if highbytes
FCAA EC 40E7            CPX     COMP+1          Equal
FCAD D0 07              BNE     INCRM           If not continue
FCAF 68                 PLA
FCB0 85 01              STA     PTA+1
FCB2 68                 PLA
FCB3 85 00              STA     PTA
FCB5 60                 RTS

FCB6 E6 00      INCRM   INC     PTA             Increase the pointer
FCB8 D0 E5              BNE     EOSLP
FCBA E6 01              INC     PTA+1
FCBC A5 01              LDA     PTA+1
FCBE 20 87F9            JSR     INRNG
FCC1 85 01              STA     PTA+1
FCC3 4C 9DFC            JMP     EOSLPA          Continue in loop

FCC6 C9 1A      8       CMP     #$1A
FCC8 D0 18              BNE     9.F

                ;*** Erase to end of line ***

FCCA 20 40F6    EEOL    JSR     SVPXY           Save current X,Y cursorpointer
FCCD A9 20      EEOLA   LDA     #' '            Get a space character
FCCF 20 23E7            JSR     DMPRAM
FCD2 AE 04E7            LDX     CURPX           Get X position
FCD5 E0 4F              CPX     #$4F            In last column?
FCD7 F0 06              BEQ     EEOLB           Branch if in last col.
FCD9 20 CFFA            JSR     CURVER          Cursor to next position
FCDC 4C CDFC            JMP     EEOLA           Continue in loop
FCDF 4C B3FD    EEOLB   JMP     SF1ENDD         Restore X,Y pointer etc.

FCE2 C9 1C      9       CMP     #$1C
FCE4 D0 03              BNE     10.F
FCE6 4C 4EFB            JMP     CURSHOME        Cursor to home position

FCE9 C9 14      10      CMP     #$14
FCEB D0 06              BNE     11.F

                ;*** Control T is start for X,Y pointer ***

FCED A2 80      CURPNT  LDX     #$80            Set bit for get X pointer
FCEF 8E 06E7            STX     CURP
FCF2 60                 RTS

FCF3 C9 1B      11      CMP     #$1B
FCF5 D0 05              BNE     12.F

                ;*** Escape character ***

FCF7 A9 FF      ESCAPE  LDA     #$FF            Set escape flag
FCF9 8D 41E7            STA     ESCFLG
FCFC 60         12      RTS
```

```
                        cont    FUNKEYS.MAC
                ;********** FUNKEYS **********

                ;*** Functionkey routines ***

FCFD            FUNTOETSEN
FCFD 29 DF              AND     #%11011111
FCFF C9 49              CMP     #$49            I?
FD01 F0 3D              BEQ     ICNTR           Input devices
FD03 C9 4F      FTO     CMP     #$4F            O?
FD05 F0 35              BEQ     OCNTR           Output devices
FD07 C9 53      FTS     CMP     #$53            S?
FD09 F0 01              BEQ     SCNTR           Toggle statusline on/off
FD0B 60         FTEND   RTS                     Not allowed

                ;*** Toggle statusline on/off ***

FD0C AD 01E7    SCNTR   LDA     STATTOG         Get statusline mode
FD0F 30 10              BMI     STATU           Was on, switch off
FD11 09 80      STATA   ORA     #$80            Was off, switch on
FD13 8D 01E7            STA     STATTOG
FD16 A9 19              LDA     #25             25 lines
FD18 A2 06      PRTAL   LDX     #$06            Number of register in CRT
FD1A 8E 40E1            STX     CRTCAR          Select register
FD1D 8D 41E1            STA     CRTCRF          Set in register
FD20 60                 RTS

FD21 29 7F      STATU   AND     #$7F
FD23 8D 01E7            STA     STATTOG         MSB is on/off bit
FD26 A9 18              LDA     #24             Select 24 lines
FD28 D0 EE              BNE     PRTAL           Continue elsewhere

                ;*** Switch statusline on ***

FD2A AD 01E7    STATAAN LDA     STATTOG
FD2D 29 7F              AND     #$7F            First off
FD2F 8D 01E7    STATN   STA     STATTOG
FD32 4C 0CFD            JMP     SCNTR           Then toggle

                ;*** Switch statusline off ***

FD35 AD 01E7    STATUIT LDA     STATTOG
FD38 09 80              ORA     #$80            First on
FD3A D0 F3              BNE     STATN           Then toggle

                ;*** Change Out- and Inputdevices, function O and I ***

FD3C A9 00      OCNTR   LDA     #$00            Output = 00
FD3E F0 02              BEQ     SF1A            Skip next instruction
FD40 A9 01      ICNTR   LDA     #$01            Input = 01
FD42 8D 0FE7    SF1A    STA     DEVCHOIC
FD45 AA                 TAX                     Choice in X
FD46 BD 10E7            LDA     DEVMODOUT,X     Save old mode
FD49 9D 12E7            STA     SDEVMODOUT,X
FD4C AD 01E7            LDA     STATTOG         Get old statusline mode
FD4F 48                 PHA                     Save it
FD50 20 40F6            JSR     SVPXY           Save X,Y cursorpointer
FD53 20 35FD            JSR     STATUIT         Statusline off
FD56 AD 0FE7    SF1AA   LDA     DEVCHOIC        Get choice
FD59 20 CBF4            JSR     PRSTAT          Print statusline 0 or 1
FD5C 20 2AFD            JSR     STATAAN         Line on
FD5F 20 CBF8    SF1B    JSR     GETKEY          Get a key
FD62 C9 0D              CMP     #$0D            Was it car. ret?
FD64 F0 26              BEQ     SF1END          If yes exit
FD66 38                 SEC
FD67 E9 30              SBC     #$30            Check if it was a number from 1...8
FD69 30 F4              BMI     SF1B            Was smaller ASCII then 30
FD6B F0 F2              BEQ     SF1B            0 not allowed
FD6D C9 09              CMP     #$09
FD6F B0 EE              BCS     SF1B            8 is the maximum
FD71 AA                 TAX                     Put number in X
FD72 A9 00              LDA     #$00
FD74 38                 SEC
FD75 6A         SF1C    RORa                    Toggle the bit from changed device
FD76 CA                 DEX                     Loopcounter depends of typed number
FD77 D0 FC              BNE     SF1C            Not reached yet
FD79 49 FF              EOR     #$FF            Invert all bits, pointed device bit is 0
```

```
FD7B AE 0FE7           LDX    DEVCHOIC        Get choice I/O
FD7E 5D 10E7           EOR    DEVMODOUT,X     On devices are 0 now
FD81 49 FF             EOR    #$FF            On devices are 1 now
FD83 9D 10E7           STA    DEVMODOUT,X     Save new device select bits
FD86 9D 14E7           STA    SDVOUT,X
FD89 4C 56FD           JMP    SF1AA           Back in loop
FD8C AE 0FE7  SF1END   LDX    DEVCHOIC        Get I/O
FD8F BD 10E7           LDA    DEVMODOUT,X     Get new dev. sel. bits
FD92 5D 12E7           EOR    SDEVMODOUT,X    Find only the changed bits
FD95 3D 10E7           AND    DEVMODOUT,X     Find only the ON switched bits
FD98 F0 0F             BEQ    SF1ENDC         Branch if no devices switched on
FD9A A8                TAY                    Save bits in Y
FD9B 8A                TXA                    Get I/O selection in A
FD9C D0 05             BNE    SF1ENDA         Branch if input selected
FD9E 98                TYA                    Restore dev. sel. bits
FD9F A2 0F             LDX    #$0F            Offset for output devices
FDA1 D0 03             BNE    SF1ENDB         Continue elsewhere

FDA3 98       SF1ENDA  TYA                    Restore dev. sel. bits
FDA4 A2 17             LDX    #$17            Offset for input devices

FDA6 20 3DF7  SF1ENDB  JSR    STATHAND        Initialise the switched on devices
FDA9 20 35FD  SF1ENDC  JSR    STATUIT         Statusline off
FDAC 68                PLA                    Get old statusline mode
FDAD 8D 01E7  SF1ENDCS STA    STATTOG         Restore it
FDB0 20 49F6           JSR    STATOUD         Restore statusline
FDB3 20 79F6  SF1ENDD  JSR    RESTOREGS       Restore all registers
FDB6 8E 04E7           STX    CURPX           Restore X,Y cursor pointers
FDB9 8C 05E7           STY    CURPY
FDBC 4C BFF9           JMP    CNVCLH          Calculate DUMP and place cursor

                ;*** Initialise I/O devices ***

                       cont   BOOTSTRAP.MAC
                ;=====================================================================
                ; file         BOOTSTRAP.MAC
                ;
                ; program      mon65 2.01
                ;
                ; contains     dos65 bootstrap
                ;
                ; by           ad brouwer and erwin visschedijk
                ;
                ; date           -apr-84
                ;              06-apr-86       for mon65 2.01 havisoft
                ;=====================================================================

FDBF 08       BCNTR    PHP
FDC0 78                SEI
FDC1 20 D0FE           JSR    INITHW          Init hardware
FDC4 B0 46             BCS    BOER
FDC6 A2 00             LDX    #0              System block
FDC8 A9 E4             LDA    #SYSB>>8
FDCA 86 FC             STX    RPOIN
FDCC 85 FD             STA    RPOIN+1
FDCE A0 01             LDY    #1
FDD0 20 2EFE           JSR    READSECT        Read system
FDD3 B0 37             BCS    BOER            Error
FDD5 AE 30E4           LDX    SYSB+$30
FDD8 AC 31E4           LDY    SYSB+$31
FDDB F0 2F             BEQ    BOER
FDDD E6 FD             INC    RPOIN+1
FDDF 20 2EFE           JSR    READSECT        TSL
FDE2 B0 28             BCS    BOER
FDE4 A9 20             LDA    #$20
FDE6 85 FB             STA    SECC
FDE8 AD 10E5           LDA    TSLB+$10        Start address
FDEB 85 FC             STA    RPOIN
FDED AD 11E5           LDA    TSLB+$11
FDF0 85 FD             STA    RPOIN+1
FDF2 F0 18             BEQ    BOER
FDF4 A6 FB    BO10     LDX    SECC            TS pair
FDF6 BD 00E5           LDA    TSLB,X
FDF9 BC 01E5           LDY    TSLB+1,X
FDFC F0 2C             BEQ    BO20            End
FDFE AA                TAX
FDFF 20 2EFE           JSR    READSECT
FE02 B0 08             BCS    BOER
```

```
FE04 E6 FD              INC     RPOIN+1         256 bytes
FE06 E6 FB              INC     SECC
FE08 E6 FB              INC     SECC
FE0A D0 E8              BNE     BO10
FE0C A9 3F      BOER    LDA     #$3F            Drive off
FE0E 8D 02E0            STA     PBD
FE11 28         8       PLP
FE12 A2 00              LDX     #0
FE14 20 9DF7            JSR     PRTEXT
FE17 424F4F542F         FCC     'BOOT/READ ERROR\n\r',0
FE1C 5245414420
FE21 4552524F52
FE26 0A0D00
FE29 60                 RTS
FE2A 28         BO20    PLP
FE2B 6C 12E5            JMP     [TSLB+$12]      Start dos65

FE2E 98         READSECT TYA
FE2F 88                 DEY
FE30 F0 19              BEQ     3.F
FE32 CC 23E4            CPY     SYSB+$23
FE35 90 0B              BCC     4.F
FE37 AD 20E4            LDA     SYSB+$20
FE3A 0A                 ASLA
FE3B 90 05              BCC     4.F
FE3D 98                 TYA
FE3E ED 23E4            SBC     SYSB+$23
FE41 A8                 TAY
FE42 B9 00E4    4       LDA     SYSB,Y
FE45 90 04              BCC     3.F
FE47 18                 CLC
FE48 6D 23E4            ADC     SYSB+$23
FE4B A8         3       TAY

                ;Floppy read sector

FE4C 8C 06E0    FREAD   STY     SCR
FE4F AD 02E0            LDA     PBD
FE52 09 08              ORA     #$08
FE54 88                 DEY
FE55 F0 07              BEQ     2.
FE57 CC 23E4            CPY     SYSB+$23
FE5A 90 02              BCC     2.
FE5C 29 F7              AND     #^$08
FE5E 8D 02E0    2       STA     PBD
FE61 A0 64              LDY     #100            Test ready drive
FE63 2C 04E0    RDY2    BIT     STR
FE66 10 0A              BPL     RDY1
FE68 A9 28              LDA     #40
FE6A 20 F5FE            JSR     BTWAIT
FE6D 88                 DEY
FE6E D0 F3              BNE     RDY2
FE70 F0 43              BEQ     RS7
FE72 A9 05      RDY1    LDA     #5              Set error count
FE74 85 FE              STA     ERCNT
FE76 85 FF              STA     ERC1
FE78 D0 13              BNE     RS1
FE7A 29 10      RS20    AND     #$10
FE7C F0 12              BEQ     RS2
FE7E 46 FF              LSR     ERC1
FE80 B0 03              BCS     RS11
FE82 20 F1FE    RS10    JSR     RESTORE
FE85 AD 02E0    RS11    LDA     PBD             Switch density
FE88 49 20              EOR     #$20
FE8A 8D 02E0            STA     PBD
FE8D 20 B7FE    RS1     JSR     TSEEK
FE90 A9 80      RS2     LDA     #RDSKT          Lees sector
FE92 8D 04E0            STA     CMR
FE95 20 CCFE            JSR     CM9
FE98 A0 00              LDY     #0
FE9A F0 06              BEQ     RS4
FE9C AD 07E0    RS3     LDA     DTR             Haal op
FE9F 91 FC              STA     [RPOIN],Y
FEA1 C8                 INY
FEA2 2C 02E0    RS4     BIT     PBD
FEA5 30 F5              BMI     RS3             DRQ?
FEA7 50 F9              BVC     RS4             IRQ?
FEA9 AD 04E0            LDA     STR
```

```
FEAC 29 1C          AND   #$1C
FEAE 18             CLC
FEAF F0 05          BEQ   9.            OK
FEB1 C6 FE          DEC   ERCNT
FEB3 D0 C5          BNE   RS20
FEB5 38       RS7   SEC                 Error
FEB6 60       9     RTS

FEB7 EC 05E0  TSEEK CPX   TKR           Seek track
FEBA F0 10          BEQ   CM9
FEBC 8E 07E0        STX   DTR
FEBF A9 19          LDA   #SEEKT
FEC1 8D 04E0  COMST STA   CMR
FEC4 20 CCFE        JSR   CM9
FEC7 2C 02E0  CM1   BIT   PBD           Till ready
FECA 50 FB          BVC   CM1
FECC 20 CFFE  CM9   JSR   CM19
FECF 60       CM19  RTS

FED0 A9 D0    INITHW LDA    #$D0        Init hardware
FED2 8D 04E0         STA    CMR
FED5 A2 04           LDX    #4
FED7 8E 03E0         STX    PBC
FEDA A9 0E           LDA    #$E
FEDC 8D 02E0         STA    PBD
FEDF A9 00           LDA    #0
FEE1 8D 03E0         STA    PBC
FEE4 A9 3F           LDA    #$3F
FEE6 8D 02E0         STA    PBD
FEE9 8E 03E0         STX    PBC
FEEC A9 FF           LDA    #$FF
FEEE 20 F5FE         JSR    BTWAIT
FEF1 A9 09    RESTORE LDA   #REST
FEF3 D0 CC           BNE    COMST

FEF5 48       BTWAIT PHA
FEF6 A9 FF           LDA    #$FF
FEF8 E9 01    WAI1   SBC    #1
FEFA B0 FC           BCS    WAI1
FEFC 68              PLA
FEFD E9 00           SBC    #0
FEFF B0 F4           BCS    BTWAIT
FF01 60              RTS

                     cont   COMMON.MAC
                ;********** COMMON **********

                ;*** Convert from 2 hex bytes to 3 decimal bytes ***

FF02 8E 3AE7  CONVER STX    DECI1       Lowbyte
FF05 8D 3BE7         STA    DECI2       Highbyte
FF08 A9 00           LDA    #0
FF0A 8D 3CE7         STA    DECI3
FF0D AA              TAX
FF0E F8              SED
FF0F F0 1B           BEQ    HDF         Branch always
FF11 CE 3BE7  HDA    DEC    DECI2
FF14 8A              TXA
FF15 69 56           ADC    #$56        Add 256
FF17 AA              TAX
FF18 68              PLA
FF19 69 02           ADC    #$02
FF1B 90 0F           BCC    HDF
FF1D EE 3CE7         INC    DECI3
FF20 D0 0A           BNE    HDF
FF22 CE 3AE7  HDB    DEC    DECI1
FF25 8A              TXA
FF26 69 01           ADC    #1
FF28 AA              TAX
FF29 68              PLA
FF2A 69 00           ADC    #0
FF2C 48       HDF    PHA
FF2D 18              CLC
FF2E AD 3AE7         LDA    DECI1
FF31 D0 EF           BNE    HDB
FF33 AD 3BE7         LDA    DECI2       End test
FF36 D0 D9           BNE    HDA
FF38 D8              CLD
```

```
FF39 68              PLA
FF3A 8D 3BE7         STA     DECI2
FF3D 8E 3AE7         STX     DECI1
FF40 60              RTS

                     ;*** Print decimal numbers ***

FF41 AD 3CE7   PRIDEC LDA     DECI3
FF44 D0 0B           BNE     CONOE          Branch if no leading zero
FF46 AD 3BE7         LDA     DECI2
FF49 D0 15           BNE     CONOF          Branch if no leading zero
FF4B AD 3AE7         LDA     DECI1
FF4E 4C 66FF         JMP     NHXOUT         Print least sign. byte
FF51 20 66FF   CONOE JSR     NHXOUT         Print byte 3
FF54 AD 3BE7         LDA     DECI2
FF57 20 6AFF         JSR     PRBYT          Print byte 2
FF5A AD 3AE7   CONOG LDA     DECI1
FF5D 4C 6AFF         JMP     PRBYT          Print byte 1
FF60 20 66FF   CONOF JSR     NHXOUT
FF63 4C 5AFF         JMP     CONOG

                     ;*** Print a byte ***

FF66 C9 10     NHXOUT CMP    #$10           Smaller than 10 (decimal mode)
FF68 90 0C           BCC     PRNIBL
FF6A 48        PRBYT PHA                    Save the byte
FF6B 4A              LSRa
FF6C 4A              LSRa                   First take the highest nibble
FF6D 4A              LSRa
FF6E 4A              LSRa
FF6F 20 7EFF         JSR     NIBASC         Nibble to ASCII conversion
FF72 20 DDF6         JSR     OUT            Print the ASCII value
FF75 68              PLA                    Restore the byte
FF76 29 0F     PRNIBL AND    #$0F           Now take lower byte
FF78 20 7EFF         JSR     NIBASC         Nibble to ASCII conversion
FF7B 4C DDF6         JMP     OUT            Print byte

FF7E C9 0A     NIBASC CMP    #$0A           Number or A...F
FF80 18              CLC
FF81 30 02           BMI     NA             Branch if number
FF83 69 07           ADC     #$07
FF85 69 30     NA    ADC     #$30           Calculate ASCII value
FF87 60              RTS

                     ;*** Print space followed by a number ***

FF88 A9 20     PRHEDE LDA    #'             Get a space character
FF8A 20 DDF6         JSR     OUT            First print a space
FF8D 8A              TXA                    Number was in X

                     ;*** Print a number ***

FF8E 20 A2FF   PRHD  JSR     HEXDE          Calculate decimal
FF91 F0 07           BEQ     PRHDB          <10 then first a space
FF93 20 76FF         JSR     PRNIBL         Print nibble on screen
FF96 98        PRHDA TYA                    Rest is in Y
FF97 4C 76FF         JMP     PRNIBL         Print also this nibble
FF9A A9 20     PRHDB LDA     #'             Get a space character
FF9C 20 DDF6         JSR     OUT            Print it
FF9F 4C 96FF         JMP     PRHDA

                     ;*** Convert hex. in A to dec. in X and Y ***

FFA2 A2 00     HEXDE LDX     #$00           X is most sign. decade
FFA4 18              CLC
FFA5 C9 0A     HEXDEA CMP    #$0A           Result larger then 10?
FFA7 90 05           BCC     HDEND          Branch if not larger
FFA9 E9 0A           SBC     #$0A           Substract 10 every time
FFAB E8              INX                    After every substraction incr. X
FFAC D0 F7           BNE     HEXDEA         Branch always
FFAE A8        HDEND TAY                    Least significant decade in Y
FFAF 8A              TXA                    Most sign. decade in X and A
FFB0 60              RTS


                     ;*** Increase X pointers ***

FFB1 EE 20E7   PNTXINC INC   PNTXL          Increase lowbyte
```

```
FFB4 D0 03              BNE     PNTXINCA
FFB6 EE 21E7            INC     PNTXH           Increase highbyte
FFB9 60         PNTXINCA RTS

                ;*** Increase FRWR pointers ***

FFBA EE 2CE7    FRWRINC INC     FRWRL           Increase lowbyte
FFBD D0 03              BNE     FRWRINCA
FFBF EE 2DE7            INC     FRWRH
FFC2 60         FRWRINCA RTS                    Increase Highbyte

                ;*** Increase FRRE pointers ***

FFC3 EE 30E7    FRREINC INC     FRREL           Increase lowbyte
FFC6 D0 03              BNE     FRREINCA
FFC8 EE 31E7            INC     FRREH           Increase Highbyte
FFCB 60         FRREINCA RTS

                ;*** Make selfmodifying routines ***

FFCC A9 AD      ADPNTX  LDA     #$AD            LDA ABS
FFCE D0 06              BNE     STPNTX
FFD0 A9 BD      BDPNTX  LDA     #$BD            LDA ABS,X
FFD2 D0 02              BNE     STPNTX
FFD4 A9 8D      D8PNTX  LDA     #$8D            STA ABS
FFD6 8D 1FE7    STPNTX  STA     PNTX            In program with pointer X
FFD9 A9 60              LDA     #$60            RTS
FFDB 8D 22E7            STA     PNTX+$03
FFDE 60                 RTS

                        LIB     IODEFS
                ;Output device definitions

        FB59    DVO1    equ     PRVDU
        F89C    DVO2    equ     PROUT
        F94C    DVO3    equ     RS232O
        F7EC    DVO4    equ     VIAOUT
        F846    DVO5    equ     MEMOUT
        F87A    DVO6    equ     FREEO1
        F880    DVO7    equ     FREEO2
        F886    DVO8    equ     FREEO3

                ;Input device definitions

        F8AC    DVI1    equ     GETASKEY
        F0F2    DVI2    equ     DUMRTS          Not used
        F945    DVI3    equ     RS232I
        F80C    DVI4    equ     VIAINP
        F85F    DVI5    equ     MEMINP
        F87D    DVI6    equ     FREEI1
        F883    DVI7    equ     FREEI2
        F889    DVI8    equ     FREEI3

                ;Output device initialization definitions

        F0F2    IT1     equ     DUMRTS          Not used
        F0F4    IT2     equ     PRINIT
        F151    IT3     equ     RSINIT
        F1F7    IT4     equ     VOINIT
        F21B    IT5     equ     MOINIT
        F0F2    IT6     equ     DUMRTS          Not used
        F0F2    IT7     equ     DUMRTS          Not used
        F0F2    IT8     equ     DUMRTS          Not used

                ;Input device initialization definitions

        F0D3    IP1     equ     KBINIT
        F0F2    IP2     equ     DUMRTS          Not used
        F151    IP3     equ     RSINIT
        F205    IP4     equ     VIINIT
        F22B    IP5     equ     MIINIT
        F0F2    IP6     equ     DUMRTS          Not used
        F0F2    IP7     equ     DUMRTS          Not used
        F0F2    IP8     equ     DUMRTS          Not used

                ;Interrupt vectors definitions

        F2B4    INT1    equ     INTTIM          T1  via A, Clock interrupt
```

```
F0F2  INT2    equ     DUMRTS          T2  via A
F0F2  INT3    equ     DUMRTS          CB1 via A
F0F2  INT4    equ     DUMRTS          CB2 via A
F0F2  INT5    equ     DUMRTS          SR  via A
F317  INT6    equ     KEYINT          CA1 via A, Keyboard interrupt
F0F2  INT7    equ     DUMRTS          CA2 via A
F0F2  INT8    equ     DUMRTS          T1  via B
F0F2  INT9    equ     DUMRTS          T2  via B
F0F2  INT10   equ     DUMRTS          CB1 via B
F0F2  INT11   equ     DUMRTS          CB2 via B
F0F2  INT12   equ     DUMRTS          SR  via B
F0F2  INT13   equ     DUMRTS          CA1 via B
F0F2  INT14   equ     DUMRTS          CA2 via B
F328  INT15   equ     ACINT           Acia interrupt
F277  INT16   equ     DUMRTI          Software interrupt

      ;*** System vectors ***

FFFA          org     $FFFA

FFFA 2DF0             fdb     ENEMI
FFFC 33F0             fdb     RESET
FFFE 30F0             fdb     IERQU


F000          end     I065
```

label table

ACCMD	E735	ACCTL	E734	ACIA	E130	ACIASR	E131	ACICMD	E132
ACICTL	E133	ACINT	F328	ADPNTX	FFCC	ASAVESTAC	E795	BCKSPCA	FBFB
BCKSPCB	FC10	BCKSPCC	FC18	BCKSPCD	FC13	BCNTR	FDBF	BDPNTX	FFD0
BO10	FDF4	BO20	FE2A	BOER	FE0C	BRKVECTOR	E7B3	BTBLCKS	E400
BTWAIT	FEF5	BUFVOL	F327	CARRET	FC6F	CHKCRT	E746	CLCKUP	F5DF
CLSEC	E718	CLSPRT1	F473	CLSPRT2	F479	CLSPRT3	F477	CLSPRT4	F485
CM1	FEC7	CM19	FECF	CM9	FECC	CMR	E004	CNVCLH	F9BF
CNVXYD	F98C	COMP	E73F	COMST	FEC1	CONOE	FF51	CONOF	FF60
CONOG	FF5A	CONTROL	FBE3	CONVER	FF02	COUD	E771	CPSTLL	FA77
CRPX	E702	CRPY	E703	CRTC	E140	CRTCAR	E140	CRTCRF	E141
CSLPRT	F466	CTB	F141	CURP	E706	CURPNT	FCED	CURPX	E704
CURPY	E705	CURSHOME	FB4E	CURSOUD	FB20	CURSUIT	FB1C	CURSZET	FB23
CURVER	FACF	CWHSLN	F490	D8PNTX	FFD4	DATRAM	FFF0	DATUPD	E77E
DAY	E77F	DECI1	E73A	DECI2	E73B	DECI3	E73C	DEVCHOIC	E70F
DEVMODINP	E711	DEVMODOUT	E710	DLHCLH	F9C2	DMPRAM	E723	DTR	E007
DTRAM	FFF0	DTUPDB	F2FF	DTUPDC	F302	DUMPH	E725	DUMPL	E724
DUMRTI	F277	DUMRTS	F0F2	DVI1	F8AC	DVI2	F0F2	DVI3	F945
DVI4	F80C	DVI4VEC	E787	DVI5	F85F	DVI6	F87D	DVI6VEC	E78B
DVI7	F883	DVI7VEC	E78F	DVI8	F889	DVI8VEC	E793	DVO1	FB59
DVO2	F88C	DVO3	F94C	DVO4	F7EC	DVO4VEC	E785	DVO5	F846
DVO6	F87A	DVO6VEC	E789	DVO7	F880	DVO7VEC	E78D	DVO8	F886
DVO8VEC	E791	EEOL	FCCA	EEOLA	FCCD	EEOLB	FCDF	EEOS	FC77
ENEMI	F02D	EOSLP	FC9F	EOSLPA	FC9D	ERC1	00FF	ERCNT	00FE
ESCAPE	FCF7	ESCFLG	E741	FDC	E004	FDCFDC	E004	FDCPIA	E000
FEBCH	F2EB	FFBACK	FB30	FILFL0	F438	FILFL1	F43E	FILFL2	F454
FILFL3	F456	FILFL4	F45E	FILFL5	F436	FILFNM	F424	FILLNM	F3ED
FNCEN	E71A	FORMFEED	FB2C	FREAD	FE4C	FREEI1	F87D	FREEI2	F883
FREEI3	F889	FREEO1	F87A	FREEO2	F880	FREEO3	F886	FRRE	E72F
FRREH	E731	FRREINC	FFC3	FRREINCA	FFCB	FRREL	E730	FRWR	E72B
FRWRH	E72D	FRWRINC	FFBA	FRWRINCA	FFC2	FRWRL	E72C	FTEND	FD0B
FTO	FD03	FTS	FD07	FUNENA	E719	FUNTAB	F74E	FUNTOETSE	FCFD
GETASKEY	F8AC	GETEVTAS	F91B	GETKEY	F8CB	GRFLG	E71C	GTKY	F8EB
GTKYA	F8EE	GTKYB	F8FF	GTKYC	F915	GTKYD	F907	HDA	FF11
HDB	FF22	HDEND	FFAE	HDF	FF2C	HEXDE	FFA2	HEXDEA	FFA5
HGA	FAB9	HGB	FAC4	HGC	FAA3	HGD	FAA7	HLPSTAT	F555
HOURS	E782	HSFLG	E70E	ICNTR	FD40	IERQU	F030	INCRM	FCB6
INITHW	FED0	INITS	F0B4	INP	F699	INPA	F6C6	INPB	F6C7
INPRET	E717	INPSTAT	F518	INPX	F6A7	INRNG	F987	INT	F22F
INT1	F284	INT10	F0F2	INT11	F0F2	INT12	F0F2	INT13	F0F2
INT14	F0F2	INT15	F328	INT16	F277	INT2	F0F2	INT3	F0F2
INT4	F0F2	INT5	F0F2	INT6	F317	INT7	F0F2	INT8	F0F2
INT9	F0F2	INTA	F24F	INTEX	F272	INTFUN	F27B	INTHAN	F267
INTLOP	F26B	INTND	F30A	INTTIM	F284	INTV1	E750	INTV10	E762
INTV11	E764	INTV12	E766	INTV13	E768	INTV14	E76A	INTV15	E76C
INTV16	E76E	INTV2	E752	INTV3	E754	INTV4	E756	INTV5	E758
INTV6	E75A	INTV7	E75C	INTV8	E75E	INTV9	E760	INVERS	E743
INVERSS	E744	INVST	E745	IO65	F000	IOACIA	E130	IOCRTC	E140
IOVIA1	E100	IOVIA2	E110	IO_BEG	F000	IO_VARS	E700	IP1	F0D3
IP2	F0F2	IP3	F151	IP4	F205	IP5	F22B	IP6	F0F2
IP7	F0F2	IP8	F0F2	IRQVECTOR	E7B5	ISLINE	F3B8	IT1	F0F2
IT2	F0F4	IT3	F151	IT4	F1F7	IT5	F21B	IT6	F0F2
IT7	F0F2	IT8	F0F2	KBINIT	F0D3	KEYBUF	E7B8	KEYINT	F317
KEYINTA	F31A	KEYINTB	F321	KEYPNT	E7B7	LDALET	E727	LETADRH	E729
LETADRL	E728	LINEFEED	FC2E	LNRH	E73E	LNRL	E73D	MAKSLN	F4A1
MAXBUF	0028	MAXSTL	E70D	MEMINP	F85F	MEMOUT	F846	MIINIT	F22B
MINUTES	E783	MOINIT	F21B	MONESC	E733	MONTAB	F30B	MONTH	E780
MVR	FA97	NA	FF85	NDVRS	E750	NHXOUT	FF66	NIBASC	FF7E
NMIVECTOR	E7B1	NOCHE	F2E8	NORSTAT	F4BB	OCNTR	FD3C	ODSTA	F4FC
OUT	F6DD	OUTCHR	E70C	OUTPSTAT	F4D7	OUTRET	E716	OUTX	F6F6
PAC	E001	PAD	E000	PBC	E003	PBD	E002	PIA	E000
PIEPER	FBE7	PNTCUR	F983	PNTX	E71F	PNTXH	E721	PNTXHSAVE	E71E
PNTXINC	FFB1	PNTXINCA	FFB9	PNTXL	E720	PNTXLSAVE	E71D	POSIT	F967
PRBYT	FF6A	PRDAT	F5B5	PRHD	FF8E	PRHDA	FF96	PRHDB	FF9A
PRHEDE	FF88	PRIDEC	FF41	PRINIT	F0F4	PRINTKAR	FBA3	PRITEND	F8A6
PRNIBL	FF76	PROFF	F54B	PRONOFF	F53F	PROUT	F88C	PROUTA	F88D
PRSEND	F3AB	PRST	F4FF	PRSTAT	F4CB	PRSTRP	F57A	PRTAL	FD18
PRTEXT	F79D	PRTEXTB	F7BD	PRTIM	F583	PRTIMA	F5AB	PRTIMB	F57E
PRTKD	FBB3	PRTKRA	FBD3	PRTKRB	FBBC	PRTKRE	FBC4	PRTKRG	FBB6
PRTKRH	FBDA	PRTKRI	FBCE	PRVDU	FB59	PRVDU1	FB64	PSSEL	F387
PTA	0000	PTB	0002	RBYTE	F80F	RBYTEND	F81A	RDSKT	0080
RDY1	FE72	RDY2	FE63	READSECT	FE2E	REBEG	E77A	RECREG	E130
REEND	E77C	RESET	F033	RESIDEV	F828	RESODEV	F81E	REST	0009
RESTORE	FEF1	RESTOREGS	F67B	RPOIN	00FC	RS1	FE8D	RS10	FE82
RS11	FE85	RS2	FE90	RS20	FE7A	RS232I	F945	RS232O	F94C
RS3	FE9C	RS4	FEA2	RS7	FEB5	RSINIT	F151	RSO1	F951
SAVEREGS	F65C	SAVSTACKP	E700	SCNTR	FD0C	SCR	E006	SCRLDWN	F9E8

SCROLL	FA37	SCURPX	E707	SCURPY	E708	SDEVMODIN	E713	SDEVMODOU	E712
SDVINP	E715	SDVOUT	E714	SEC1.20	E737	SECC	00FB	SECONDS	E784
SEEKT	0019	SELSTL	F38A	SET65	FB16	SETINV	F395	SF1A	FD42
SF1AA	FD56	SF1B	FD5F	SF1C	FD75	SF1END	FD8C	SF1ENDA	FDA3
SF1ENDB	FDA6	SF1ENDC	FDA9	SF1ENDCS	FDAD	SF1ENDD	FDB3	ST2UD	F612
START	E7AD	STATA	FD11	STATAAN	FD2A	STATFG	E742	STATHAND	F73D
STATN	FD2F	STATO	F659	STATOUD	F649	STATTOG	E701	STATU	FD21
STATUIT	FD35	STPNTX	FFD6	STR	E004	SVAINT	E770	SVPXY	F640
SYSB	E400	TAB	FC1F	TABEL	F75D	TIMDAT	E736	TKR	E005
TOFFSC	F921	TONSC	F92F	TOUTF	E738	TOUTH	E775	TOUTIH	E773
TOUTIL	E772	TOUTL	E774	TOUTT	0708	TRAREG	E130	TREMP	E739
TRMINT	F34A	TSEEK	FEB7	TSLB	E500	UITCRV	FAEB	UITCRVA	FAFF
UITPOINT	FB8E	UNINT	F357	UNNMI	F370	UNPOS	F95F	UNRINT	E7AF
UNSELS	F3A2	UNSLPS	F3B2	VAACR	E10B	VAADN	E10F	VAIER	E10E
VAIFR	E10D	VAPAD	E101	VAPADD	E103	VAPBD	E100	VAPBDD	E102
VAPCR	E10C	VART1	F162	VART2	F1AD	VASR	E10A	VATACH	E105
VATACL	E104	VATALH	E107	VATALL	E106	VATBCH	E109	VATBCL	E108
VBACR	E11B	VBADN	E11F	VBIER	E11E	VBIFR	E11D	VBPAD	E111
VBPADD	E113	VBPBD	E110	VBPBDD	E112	VBPCR	E11C	VBSR	E11A
VBTACH	E115	VBTACL	E114	VBTALH	E117	VBTALL	E116	VBTBCH	E119
VBTBCL	E118	VDUINIT	F12C	VDURAM	E800	VERBLANK	FB02	VERTTAB	FC48
VIAA	E100	VIAAINT	F25B	VIAB	E110	VIABINT	F262	VIAINP	F80C
VIAL	F837	VIALOOP	F832	VIAOUT	F7EC	VIAVRA	E71B	VIINIT	F205
VOINIT	F1F7	VRBLK	FB0E	VTTA	FC5B	VTTB	FC5E	VTUIT	FC61
WACHTACK	F89B	WAI1	FEF8	WBYTE	F7FD	WBYTEF	F7EF	WBYTEND	F807
WRBEG	E776	WREND	E778	XOKE	FB81	XPOINTER	FB75	XPSOLD	E70A
XSAVESTAC	E79D	XTMP	E709	XTODEV	F691	YEAR	E781	YOKE	FB94
YPOINTER	FB8B	YPSOLD	E70B	YSAVESTAC	E7A5				

Errors detected: 0