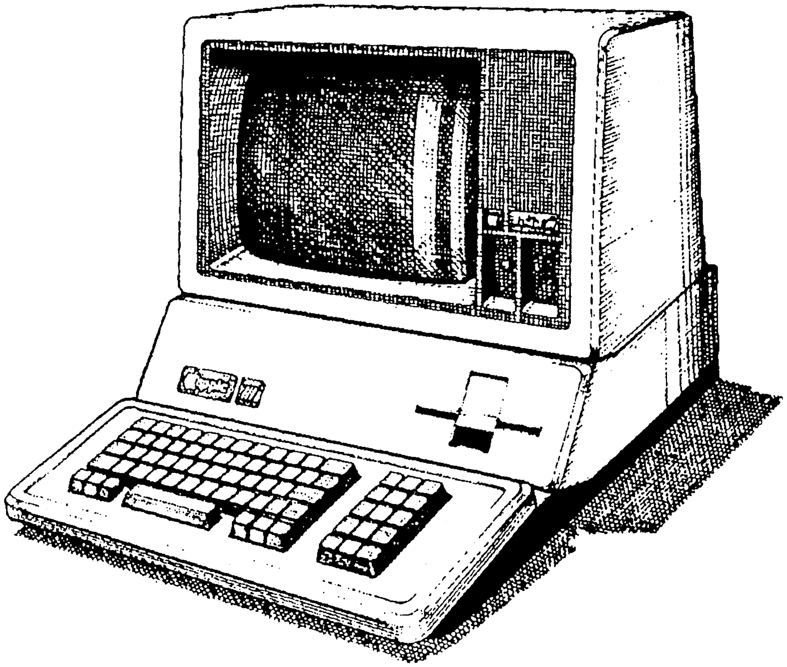
Apple /// Computer Technical Information

# Apple ///
# CONSOLE DRIVER
# Version 1.31
# Source Code Listing



```
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONSOLE.TEXT
; ####################################################################################################

                .TITLE       "SOS Console Driver"
                .NOPATCHLIST
                .NOMACROLIST

;--------------------------------------------------------------------
;
;               SOS Console Driver
;
;               Copyright (C) 1983 by Apple Computer Inc.
;               All Rights Reserved
;
;               Previous Copyright (C) 1980, 1981
;
;
;       Revisions:
;
;       1.00    14-Nov-80       Initial Release
;
;       1.12    23-Sep-81
;           Bug fixes:
;               Download 1-8 characters.
;               Download entire character set.
;               Include saved screen state in console state table.
;               Adjust all pointers for proper extended addressing.
;               Fix SYNC to monitor positive edge of vertical blanking.
;               Delete extraneous data returned by status calls 12, 13, & 14.
;               Fix erase option of character and line delete.
;           Extensions:
;               Add video toggle on control-5.
;               Add dump & restore contents of viewport.
;               Change keyboard transform table to include alpha-lock data.
;               Retain cursor on SYNC.
;
;       1.30    11-Jan-83
;           Bug fixes:
;               Wait for pending download on close.
;               Fix branch in 40 column horizontal shift right.
;               Fix cursor in dump & restore contents of viewport.
;               Disable interrupts while setting events and screen mode.
;           Extensions:
;               Turn on video iff buffer is empty.
;               Set bit 7 on control characters read from screen
;                   (applies to char copy and screen read status).
;               Don't dump viewport when displaying control characters.
;               Add status request 9, read screen with normal/inverse flag.
;       1.31    17-Mar-83
;               Fix VERIFY to eleminate noise when setting screen switches.
;
;-----------------------------------------------------------------------

DEVTYPE         .EQU         61
SUBTYPE         .EQU         01
APPLE           .EQU         0001
RELEASE         .EQU         1310
                .PAGE
;-----------------------------------------------------------------------
;
;  The macro SWITCH performs an N way branch based on a switch index.  The
;  maximum value of the switch index is 127 with bounds checking provided
;  as an option.  The macro uses the A and Y registers and alters the C,
;  Z, and N flags of the status register, but the X register is unchanged.
;
;               SWITCH  [index], [bounds], adrs_table, [*]
;
;       index   This is the variable that is to be used as the switch index.
;               If omitted, the value in the accumulator is used.
;
;      bounds   This is the maximum allowable value for index.  If index
;               exceeds this value, the carry bit will be set and execution
;               will continue following the macro.  If bounds is omitted,
;               no bounds checking will be performed.
;
;  adrs_table   This is a table of addresses (low byte first) used by the
;               switch.  The first entry corresponds to index zero.
;
;           *   If an asterisk is supplied as the fourth parameter, the
;               macro will push the switch address but will not exit to
;               it; execution will continue following the macro.  The
;               program may then load registers or set the status before
;               exiting to the switch address.
;
;-----------------------------------------------------------------------
;
                .MACRO       SWITCH
                .IF          "%1" <> ""                ;If PARM1 is present,
                LDA          %1                        ;  Load A with switch index
                .ENDC
                .IF          "%2" <> ""                ;If PARM2 is present,
                CMP          #%2+1                     ;  Perform bounds checking
                BCS          $3579                     ;  on switch index
                .ENDC
                ASL          A
                TAY
                LDA          %3+1,Y                    ;Get switch address from table
                PHA                                    ;  and push onto stack
                LDA          %3,Y
                PHA
                .IF          "%4" <> "*"               ;If PARM4 is omitted,
                RTS                                    ;  Exit to code
                .ENDC                                  ;Otherwise, drop through
                .IF          "%2" <> ""
$3579
                .ENDC
                .ENDM

                .INCLUDE     :CONS.DAT1.TEXT
                .INCLUDE     :CONS.DAT2.TEXT
                .INCLUDE     :CONS.DAT3.TEXT
                .INCLUDE     :CONS.MAIN.TEXT
                .INCLUDE     :CONS.READ.TEXT
                .INCLUDE     :CONS.WRIT.TEXT
                .INCLUDE     :CONS.FCTN.TEXT
                .INCLUDE     :CONS.STAT.TEXT
                .INCLUDE     :CONS.CNTL.TEXT
                .INCLUDE     :CONS.DNLD.TEXT
                .INCLUDE     :CONS.MISC.TEXT
                .INCLUDE     :CONS.UTL1.TEXT
                .INCLUDE     :CONS.UTL2.TEXT

                .END


; ####################################################################################################
; #   END OF FILE:  CONSOLE.TEXT
; #   LINES      :  121
; #   CHARACTERS :  6057
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.DAT1.TEXT
; ####################################################################################################

                .PROC        CONSOLE
                .WORD        0FFFF
                .WORD        59.
                .ASCII       "Console Driver -- "
                .ASCII       "Copyright (C) 1983 by Apple Computer Inc."
;----------------------------------------------------------------------
;
;  Device Handler Identification Block
;
;----------------------------------------------------------------------
;
IDBLK           .WORD        0000                      ;Link to next device handler
                .WORD        CNSLDH                    ;Entry point address
                .BYTE        8                         ;Length of device name
                .ASCII       ".CONSOLE       "
                .BYTE        80,00,00                  ;Device, Slot & Unit numbers
                .BYTE        DEVTYPE
                .BYTE        SUBTYPE
                .BYTE        00
                .WORD        0000
                .WORD        APPLE
                .WORD        RELEASE
                .WORD        00                        ;No configuration block
                .PAGE
;----------------------------------------------------------------------
;
;  Global Data:
;
;    SUSPFLSH:  Suspend and Flush Output Flags
;       7 => Suspend Output
;       6 => Flush Output
;
;    SCRNMODE:  Current Screen Mode
;       7 => Off / On
;       6 => Text / Graphics
;       2 => Page 1 / Page 2
;       1 => 40 Col / 80 Col
;       0 => B & W / Color
;
```

```
;
;  State Flags:
;
;    HMODE:  Hardware Mode
;       7 => 40 Col / 80 Col
;       1 => 40 Col / 80 Col
;       0 => B & W / Color
;
;    SMODE:  Software Mode
;       5 => Normal / Inverse
;       4 => Disable / Enable Cursor
;       3 => Disable / Enable Scroll
;       2 => Disable / Enable Auto Carriage Return
;       1 => Disable / Enable Auto Line Feed
;       0 => Disable / Enable Auto Advance
;
;
;  Permanant Zero Page Data:
;
;    BASE1, BASE2:  Screen Memory Pointers
;       The base pointers point to the beginning of the current line.  In
;       40 column mode, BASE1 points to the ASCII data while BASE2 points
;       to the color information.  In 80 column mode, BASE1 points to col-
;       umn 0 of the viewport while BASE2 points to column 1.
;
;
;  Temporary Zero Page Data:
;
;    WORK1, WORK2:
;       These pointers are used in conjunction with BASE1 and BASE2 for
;       scrolling, shifting, etc.
;
;    COUNT:
;       Number of bytes read or written.
;
;    ONEBYTE:
;       Boolean flag for single byte read requests.
;
;    BLANK:
;       Holds an ASCII space in the current video mode (normal or inverse)
;       for use in clearing the viewport.
;
;    TEMPX:
;       Temporary storage for X.
;
```

```
;    FLAGS:
;       Miscellaneous flags for use by SCROLL, SHIFT, SCRNDUMP, etc.
;
;    TEMP1, TEMP2, TEMP3, TEMP4:
;       General temporary storage for use by SCROLL, SHIFT, SCRNDUMP, etc.

; ####################################################################################################
; #   END OF FILE:  CONS.DAT1.TEXT
; #   LINES      :  89
; #   CHARACTERS :  3512
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.DAT2.TEXT
; ####################################################################################################

                .PAGE
;
;  SOS Global Data & Subroutines
;
SUSPFLSH        .EQU         1902                      ;Suspend & Flush flags
SCRNMODE        .EQU         1906                      ;Current Screen Mode
ALLOCSIR        .EQU         1913
DEALCSIR        .EQU         1916
QUEEVENT        .EQU         191F
SYSERR          .EQU         1928
;
;  SOS Error Codes
;
XREQCODE        .EQU         20                        ;Invalid request code
XCTLCODE        .EQU         21                        ;Invalid controlstatus code
XCTLPARM        .EQU         22                        ;Invalid controlstatus parm
XNOTOPEN        .EQU         23                        ;Device not open
XNOTAVIL        .EQU         24                        ;Device not available
XNORESRC        .EQU         25                        ;Unable to obtain resource
;
;  Hardware I/O Addresses
;
KAPORT          .EQU         0C000
KBPORT          .EQU         0C008
KYBDSTRB        .EQU         0C010
KYBDCLR         .EQU         01                        ;Clear keyboard interrupt flag
KYBDDSBL        .EQU         01                        ;Disable keyboard interrupts
KYBDENBL        .EQU         81                        ;Enable keyboard interrupts
BELL            .EQU         0C040
VMODE0          .EQU         0C050                     ;Video mode switches
VMODE1          .EQU         0C052
VMODE2          .EQU         0C054
VMODE3          .EQU         0C056
SCRLDSBL        .EQU         0C0D8                     ;Disable graphics scroll
DNLDDSBL        .EQU         0C0DA                     ;Disable character download
DNLDENBL        .EQU         0C0DB                     ;Enable character download
VBLCLR          .EQU         18                        ;Clear both VBL interrupt flags
VBLDSBL         .EQU         18                        ;Disable both VBL interrupts
VBLENBL         .EQU         90                        ;Enable VBL interrupt on CB2
```

```
E_REG           .EQU         0FFDF                     ;Environment register
E_IORB          .EQU         0FFE0                     ;6522 input/output register B
E_PCR           .EQU         0FFEC                     ;6522 peripheral control register
E_IFR           .EQU         0FFED                     ;6522 interrupt flag register
E_IER           .EQU         0FFEE                     ;6522 interrupt mask register
B_REG           .EQU         0FFEF                     ;Bank register
;
;  ASCII Equates and Special Keys
;
ASC_NUL         .EQU         00                        ;Null
ASC_SOH         .EQU         01                        ;Start of Header
ASC_STX         .EQU         02                        ;Start of Text
ASC_ETX         .EQU         03                        ;End of Text
ASC_ENQ         .EQU         05                        ;Enquiry
ASC_ACK         .EQU         06                        ;Acknowledgement
ASC_BS          .EQU         08                        ;Backspace
ASC_HT          .EQU         09                        ;Horizontal Tab
ASC_LF          .EQU         0A                        ;Line Feed
ASC_VT          .EQU         0B                        ;Vertical Tab
ASC_FF          .EQU         0C                        ;Form Feed
ASC_CR          .EQU         0D                        ;Carriage Return
ASC_NAK         .EQU         15                        ;Negative Acknowledge
ASC_CAN         .EQU         18                        ;Cancel
ASC_ESC         .EQU         1B                        ;Escape
ASC_FS          .EQU         1C                        ;File Separator
ASC_GS          .EQU         1D                        ;Group Separator
ASC_US          .EQU         1F                        ;Unit Separator
ASC_SP          .EQU         20                        ;Space
LARROW          .EQU         ASC_BS                    ;Left Arrow
RARROW          .EQU         ASC_NAK                   ;Right Arrow
UARROW          .EQU         ASC_VT                    ;Up Arrow
DARROW          .EQU         ASC_LF                    ;Down Arrow
;
;  Miscellaneous Equates
;
TRUE            .EQU         80
FALSE           .EQU         00
BITON0          .EQU         01
BITON2          .EQU         04
BITON3          .EQU         08
BITON4          .EQU         10
BITON5          .EQU         20
BITON6          .EQU         40
BITON7          .EQU         80
BITOFF0         .EQU         0FE
```

```
BITOFF4         .EQU         0EF
BITOFF5         .EQU         0DF
BITOFF7         .EQU         07F
BUFMAX          .EQU         80                        ;Maximum buffer size
TEXTCSA         .EQU         0C00                      ;Text character set address
                .PAGE
;-----------------------------------------------------------------------
;
;   SOS Device Handler Interface
;
;-----------------------------------------------------------------------
;
SOSINT          .EQU         0C0
REQCODE         .EQU         SOSINT+0                  ;SOS request code
BUFFPTR         .EQU         SOSINT+2                  ;Buffer pointer
REQCNT          .EQU         SOSINT+4                  ;Requested count
RTNCNT          .EQU         SOSINT+8                  ;Returned count
SCCODE          .EQU         SOSINT+2                  ;Status / Control code
SCLIST          .EQU         SOSINT+3                  ;Status / Control list
;
;
;-----------------------------------------------------------------------
;
;  Zero Page Data (preserved) and Zero Page Save Area
;
;-----------------------------------------------------------------------
;
ZPDATA          .EQU         SOSINT+10.
BASEPTRS        .EQU         ZPDATA+0                  ;Screen memory base pointers
BASE1           .EQU         BASEPTRS+0                ;  even col. / text bytes
BASE2           .EQU         BASEPTRS+2                ;  odd col. / color bytes
ZPLENGTH        .EQU         4
;
ZPSAVE          .BLOCK       ZPLENGTH
;
;
;-----------------------------------------------------------------------
;
;  Zero Page Data (temporary)
;
;-----------------------------------------------------------------------
;
WORKPTRS        .EQU         ZPDATA+ZPLENGTH
WORK1           .EQU         WORKPTRS+0
WORK2           .EQU         WORKPTRS+2
```

```
COUNT           .EQU         WORKPTRS+4                ;Current I/O count
ONEBYTE         .EQU         COUNT+2                   ;One byte console read flag
BLANK           .EQU         ONEBYTE+1
TEMPX           .EQU         BLANK+1
FLAGS           .EQU         TEMPX+1
TEMP1           .EQU         FLAGS+1
TEMP2           .EQU         TEMP1+1
TEMP3           .EQU         TEMP2+1
TEMP4           .EQU         TEMP3+1

; ####################################################################################################
; #   END OF FILE:  CONS.DAT2.TEXT
; #   LINES      :  138
; #   CHARACTERS :  7595
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.DAT3.TEXT
; ####################################################################################################

                .PAGE
;-----------------------------------------------------------------------
;
;  Console State Table
;
;-----------------------------------------------------------------------
;
CONSTTBL        .EQU         *                         ;Console state table
;
ANYKYEVNT       .BLOCK       5                         ;Any Key Event parameters
ATTNEVNT        .BLOCK       5                         ;Attention Event parameters
ATTNCHAR        .BYTE        0                         ;Attention character
;
DFLTTBL         .EQU         *                         ;This block initialized from default values
;
KYBDMODE        .BYTE        0                         ;Console/Keyboard mode flag
NEWLINE         .BYTE        0                         ;New Line flag
NEWLNCHR        .BYTE        0                         ;New Line character
NOWAIT          .BYTE        0                         ;No Wait flag
ECHO            .BYTE        0                         ;Screen Echo flag
CHCPYFLG        .BYTE        0                         ;Character Copy flag
CHCPYCHR        .EQU         ASC_NAK                   ;Character Copy character
CHDELFLG        .BYTE        0                         ;Character Delete flag
CHDELCHR        .EQU         ASC_BS                    ;Character Delete character
LNDELFLG        .BYTE        0                         ;Line Delete flag
LNDELCHR        .EQU         ASC_CAN                   ;Line Delete character
ESCAPE          .BYTE        0                         ;Escape Mode flag
;
SCRSTTBL        .EQU         *                         ;Screen state table
;
HMODE           .BYTE        0                         ;Hardware mode
SMODE           .BYTE        0                         ;Software mode
TPX             .BYTE        0                         ;Text position
TPY             .BYTE        0
VPL             .BYTE        0                         ;Viewport
VPR             .BYTE        79.
VPT             .BYTE        0
VPB             .BYTE        23.
TCF             .BYTE        0F                        ;Text color
```

```
TCB             .BYTE        00
;
SCRSTLEN        .EQU         *-SCRSTTBL
;
DFLTLEN         .EQU         *-DFLTTBL
;
SCRSTSAV        .BLOCK       SCRSTLEN                  ;Saved screen state table
;
CONSTLEN        .EQU         *-CONSTTBL
                .PAGE
;----------------------------------------------------------------------
;
;  Default Values for State Table
;
;----------------------------------------------------------------------
;
DFLTVAL         .BYTE        FALSE                     ;Console / Keyboard flag
                .BYTE        FALSE                     ;Newline flag
                .BYTE        ASC_CR                    ;Newline character
                .BYTE        FALSE                     ;Nowait flag
                .BYTE        TRUE                      ;Screen echo flag
                .BYTE        TRUE                      ;Character copy flag
                .BYTE        TRUE                      ;Character delete flags
                .BYTE        TRUE                      ;Line delete flags
                .BYTE        TRUE                      ;Escape mode flags
                .BYTE        02                        ;Hardware mode
                .BYTE        0D                        ;Software mode
                .BYTE        0.                        ;Cursor position
                .BYTE        0.
                .BYTE        0.                        ;Viewport
                .BYTE        79.
                .BYTE        0.
                .BYTE        23.
                .BYTE        0F                        ;Text colors
                .BYTE        00
                .PAGE
;-----------------------------------------------------------------------
;
;   Private Variable Storage
;
;-----------------------------------------------------------------------
;
KYBDBUFS        .EQU         01500                     ;Type ahead buffers
KABUF           .EQU         KYBDBUFS
KBBUF           .EQU         KYBDBUFS+BUFMAX
```

```
XFORMTBL        .EQU         01700                     ;Keyboard transform table
;
KADATA          .BYTE        0                         ;Temp Storage
KBDATA          .BYTE        0                         ;  for Interrupt Processing
;
KEYCNT          .BYTE        0                         ;Buffered keystroke count
BUFSIZ          .BYTE        0                         ;Current buffer size
BUFHEAD         .BYTE        0                         ;Index of first character
BUFTAIL         .BYTE        0                         ;Index of last character
;
OPENFLG         .BYTE        0                         ;Device open flag
READING         .BYTE        0                         ;Read in progress flag
DSPLYCTL        .BYTE        0                         ;Display control characters
;
SMFLAGS         .EQU         *
SMINV           .BYTE        0                         ;Inverse video
SMCURSOR        .BYTE        0                         ;Cursor enabled
SMSCROLL        .BYTE        0                         ;Scroll flag
SMAUTOCR        .BYTE        0                         ;Auto CR
SMAUTOLF        .BYTE        0                         ;Auto LF
SMAUTOADV       .BYTE        0                         ;Auto advance
;
VPHMAX          .BYTE        79.                       ;viewport maximum horizontal index
VPVMAX          .BYTE        23.                       ;viewport maximum vertical index
TCOLOR          .BYTE        0F0                       ;text fg/bg color byte
;
CTLINDX         .BYTE        0                         ;function buffer index
CTLBUFF         .BLOCK       8                         ;control function buffer
CTLQUOTA        .BYTE        0                         ;parameter quota
;
DNLDFLG         .BYTE        00                        ;Bit 7=Active, Bit 6=Request
DNLDCEL         .BYTE        00                        ;Current download cell number
DNLDCHR         .BYTE        00                        ;Current download ASCII code
DNLDIMG         .WORD        0000                      ;Pointer to character image
                .PAGE
;-----------------------------------------------------------------------
;
;  Addresses used as subroutine parameters and SIR request tables
;
;-----------------------------------------------------------------------
;
ANYKYPARM       .WORD        ANYKYEVNT
ATTNPARM        .WORD        ATTNEVNT
;
KYBDSADR        .WORD        KYBDSTBL
```

```
KYBDSTBL        .BYTE        2,0                       ;Keyboard interrupt
                .WORD        KYBDMIH
KYBDBANK        .BYTE        0
KYBDSSIZ        .EQU         *-KYBDSTBL
;
DNLDSADR        .WORD        DNLDSTBL
DNLDSTBL        .BYTE        5,0,0,0,0                 ;VBL positive
                .BYTE        6,0                       ;VBL negative
                .WORD        DNLDINT
DNLDBANK        .BYTE        0
                .BYTE        10,0,0,0,0                ;Character download / Graphics scroll
DNLDSSIZ        .EQU         *-DNLDSTBL
;
SYNCSADR        .WORD        SYNCSTBL
SYNCSTBL        .BYTE        5,0,0,0,0                 ;VBL positive
SYNCSSIZ        .EQU         *-SYNCSTBL
;
;
;---------------------------------------
;
;   Base Calculator Address Tables
;
;---------------------------------------
;
BASL            .BYTE        000,080,000,080
                .BYTE        000,080,000,080
                .BYTE        028,0A8,028,0A8
                .BYTE        028,0A8,028,0A8
                .BYTE        050,0D0,050,0D0
                .BYTE        050,0D0,050,0D0
BASH            .BYTE        004,004,005,005
                .BYTE        006,006,007,007
                .BYTE        004,004,005,005
                .BYTE        006,006,007,007
                .BYTE        004,004,005,005
                .BYTE        006,006,007,007
                .PAGE
;----------------------------------------------------------------------
;
;  Escape Command and Escape Operator Tables
;
;----------------------------------------------------------------------
;
ESCCMD          .BYTE        "B"                       ;Viewport bottom right
                .BYTE        "T"                       ;Viewport top left
                .BYTE        "V"                       ;Clear Viewport
                .BYTE        "S"                       ;Clear Screen
                .BYTE        "P"                       ;Clear to End of Page
                .BYTE        "L"                       ;Clear to End of Line
                .BYTE        "H"                       ;Home Cursor
                .BYTE        ASC_BS                    ;Move left
                .BYTE        ASC_NAK                   ;Move right
                .BYTE        ASC_VT                    ;Move up
                .BYTE        ASC_LF                    ;Move down
ECMDCNT         .EQU         *-ESCCMD
;
ESCOP           .BYTE        ASC_ETX
                .BYTE        ASC_STX
                .BYTE        ASC_SOH
                .BYTE        ASC_FS
                .BYTE        ASC_GS
                .BYTE        ASC_US
                .BYTE        ASC_FF
                .BYTE        ASC_BS
                .BYTE        ASC_HT
                .BYTE        ASC_VT
                .BYTE        ASC_LF


; ######################################################################################################
; #   END OF FILE:  CONS.DAT3.TEXT
; #   LINES      :  197
; #   CHARACTERS :  10599
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ######################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.MAIN.TEXT
; ####################################################################################################

                .PAGE
;-----------------------------------------------------------------------
;
;  Console Device Handler
;
;  This is the device handler's entry point.  It sets the extended
;  addressing bytes to zero and moves in the permanant zero page
;  data, then switches to the appropriate request handler.  If the
;  request handler modifies the permanant zero page data, it must
;  call ZPOUT before it exits to SOS.
;
;-----------------------------------------------------------------------
;
CNSLDH          .EQU         *
                LDX          #0FF-ZPDATA
                LDY          #00
                TYA
$010            STA          1400+ZPDATA,X             ;Set extend bytes to zero
                CPX          #ZPLENGTH
                BCS          $020
                LDA          ZPSAVE,X
                STA          ZPDATA,X                  ;Set up zero page data
                TYA
$020            DEX
                BPL          $010
;
                SWITCH       REQCODE,8,CREQSW
;
;
CBADREQ         LDA          #XREQCODE                 ;Invalid request code
                JSR          SYSERR
;
CNOTOPEN        LDA          #XNOTOPEN                 ;Console is not open
                JSR          SYSERR
;
CREQSW          .WORD        CNSLREAD-1
                .WORD        CNSLWRIT-1
                .WORD        CNSLSTAT-1
                .WORD        CNSLCNTL-1
                .WORD        CBADREQ-1
                .WORD        CBADREQ-1
                .WORD        CNSLOPEN-1
                .WORD        CNSLCLOS-1
                .WORD        CNSLINIT-1
                .PAGE
;-----------------------------------------------------------------------
;
;  Keyboard Interrupt Handler
;
;-----------------------------------------------------------------------
;
KYBDMIH         .EQU         *
;
;  Read keyboard data and clear interrupt
;
                LDX          #KYBDCLR
                LDA          KAPORT                    ;Read data port
                BMI          $010
                STX          E_IFR                     ;No data ready -- clear
                RTS                                    ;  interrupt and exit
$010            AND          #BITOFF7
                STA          KADATA
                LDA          KBPORT                    ;Read status port
                EOR          #3C
                STA          KBDATA
                STX          E_IFR                     ;Clear interrupt
                STX          KYBDSTRB                  ;  and keyboard strobe
                BMI          KIHSPCL
                LDA          KADATA
                CMP          #ASC_CR
                BNE          KIHXFORM
                LDA          KBDATA
                AND          #BITON2                   ;Transform CR iff
                BNE          KIHXFORM                  ;  CTRL is held down
                JMP          KIHA1KY
;
;  Special key
;    Check for console control commands
;    Do not transform character code
;
KIHSPCL         AND          #36                       ;Isolate A1, A2, CTRL, & SHIFT
                CMP          #BITON2
                BEQ          $050
$010            JMP          KIHA1KY                   ;Not a console control command
```

```
$050            LDA          KADATA
                CMP          #"5"                      ;Toggle video?
                BCC          $010
                BNE          $060
                LDA          SCRNMODE
                EOR          #BITON7
                STA          SCRNMODE
                RTS
$060            CMP          #"6"                      ;Flush input buffer?
                BNE          $070
                LDA          #00
                STA          KEYCNT
                STA          BUFHEAD
                STA          BUFTAIL
                RTS
$070            CMP          #"7"                      ;Suspend screen output?
                BNE          $080
                LDA          SUSPFLSH
                EOR          #BITON7
                STA          SUSPFLSH
                RTS
$080            CMP          #"8"                      ;Display control characters?
                BNE          $090
                LDA          DSPLYCTL
                EOR          #BITON7
                STA          DSPLYCTL
                RTS
$090            CMP          #"9"                      ;Flush screen output?
                BNE          KIHA1KY
                LDA          SUSPFLSH
                AND          #BITOFF7
                EOR          #BITON6
                STA          SUSPFLSH
                RTS
;
;  Standard key
;    Transform character code
;    Check for alpha lock
;
KIHXFORM        LDA          KADATA
$010            CMP          #7B                       ;Convert ASCII code to
                BCC          $030                      ;  transform table index
                CMP          #7E
                BCC          $020
                EOR          #0C0
                BNE          $040
$020            AND          #5F
$030            ORA          #0C0
$040            TAX
                LDA          KBDATA                    ;Get control & shift keys
                LSR          A
                AND          #03
                ORA          XFORMTBL,X                ;OR in key number
                TAX
                LDA          XFORMTBL,X                ;Need to test alpha lock?
                BPL          $050
                LDA          KBDATA                    ;Check alpha lock key
                AND          #BITON3
                BEQ          $050
                TXA
                ORA          #BITON0                   ;Force shift key on
                TAX
$050            LDA          XFORMTBL,X                ;Get key code
                AND          #BITOFF7
                STA          KADATA
;
;  Set bit 7 according to Apple 1 key
;
KIHA1KY         LDA          KBDATA
                AND          #BITON4
                BEQ          KIHCKEV
                LDA          KADATA
                ORA          #BITON7
                STA          KADATA
;
;  Check for Any Key and Attention events
;
KIHCKEV         BIT          READING                   ;If reading,
                BMI          $010                      ;  ignore Any Key event
                LDA          ANYKYEVNT                 ;Check Any Key Event
                BEQ          $010
                LDX          ANYKYPARM
                LDY          ANYKYPARM+1
                JSR          QUEEVENT                  ;Queue the event
                LDA          #FALSE
                STA          ANYKYEVNT                 ;Disable Any Key event
                BEQ          $020
$010            LDA          ATTNEVNT                  ;Check Attention Event
                BEQ          KIHBFCH
                LDA          KADATA
                CMP          ATTNCHAR
                BNE          KIHBFCH
                LDX          ATTNPARM
                LDY          ATTNPARM+1
                JSR          QUEEVENT                  ;Queue the event
                LDA          #FALSE
                STA          ATTNEVNT                  ;Disable Attention event
$020            STA          READING                   ;Terminate any read in progress
                STA          KEYCNT                    ;Flush the input buffer
                STA          BUFHEAD
                STA          BUFTAIL
                STA          SUSPFLSH                  ;Clear suspend & flush flags
;
;  Buffer the character
;
KIHBFCH         LDX          BUFSIZ                    ;Buffering enabled?
                BEQ          $030
                DEX
                CPX          KEYCNT                    ;Any room in buffer?
                BCS          $010
                BIT          BELL                      ;Buffer overflow
                BCC          $030
$010            INC          KEYCNT                    ;Bump the key count
                LDX          BUFTAIL
                LDA          KADATA
                STA          KABUF,X                   ;Buffer the keystroke
                LDA          KBDATA
                STA          KBBUF,X
                INX
                CPX          BUFSIZ                    ;Bump buffer tail pointer
                BCC          $020
                LDX          #0
$020            STX          BUFTAIL
$030            RTS
                .PAGE
;-----------------------------------------------------------------------
;
;  Subroutine GETKEY
;
;  This subroutine gets the next keystroke from the type ahead buffer.
;  On entry, the interrupt system must be enabled but the keyboard
;  interrupt must be masked.  On exit, if carry is clear, A contains
;  the keyboard A port data and X contains the keyboard B port data;
;  Y is undefined.  If carry is set, no data is returned; either the
;  buffer was empty and the NOWAIT flag is true, or the read was
;  terminated by the interrupt handler.
;
;-----------------------------------------------------------------------
;
GETKEY          .EQU         *
                LDA          KEYCNT                    ;Anything in the buffer?
                BNE          $030                      ;  Yes
                PHP
                SEI
                LDA          SCRNMODE
                ORA          #BITON7
                STA          SCRNMODE                  ;Turn on video
                LDA          E_REG
                ORA          #BITON5
                STA          E_REG
                PLP
                BIT          NOWAIT                    ;Check the NOWAIT flag
                BPL          $010
                ASL          READING                   ;Clear the READING flag,
                RTS                                    ;  set carry, and exit
;
$010            LDX          BUFSIZ                    ;Preserve buffer size in X
                LDA          #1                        ;Set buffer size to 1
                STA          BUFSIZ
                LDA          #KYBDENBL                 ;Unmask the keyboard
                STA          E_IER
                BIT          ESCAPE                    ;In ESCAPE mode?
                BVC          $020                      ;  No
                CLC
                LDA          TPX                       ;Preserve current cursor and
                LDY          HMODE                     ;  replace it with plus sign
                BPL          $015
                LSR          A
                BCC          $015
                TAY
                LDA          (BASE2),Y
                PHA
                AND          #BITON7
                ORA          #2B
                STA          (BASE2),Y
                BCS          $020
$015            TAY
                LDA          (BASE1),Y
                PHA
                AND          #BITON7
                ORA          #2B
                STA          (BASE1),Y
$020            LDA          KEYCNT                    ;Wait for a keystroke
                BEQ          $020
                BVC          $026                      ;Not in ESCAPE mode
                PLA                                    ;Restore original cursor
                BCC          $024
                STA          (BASE2),Y
                BCS          $026
$024            STA          (BASE1),Y
$026            LDA          #KYBDDSBL                 ;Mask the keyboard
                STA          E_IER
                STX          BUFSIZ                    ;Restore the buffer size
                SEC
                BIT          READING                   ;Check the reading flag
                BPL          $060                      ;  Exit with carry set
;
$030            LDY          BUFHEAD                   ;Get buffer index of keystroke
                DEC          KEYCNT
                BNE          $040                      ;If KEYCNT = 0
                LDA          #0
                STA          BUFHEAD                   ;  then BUFHEAD := BUFTAIL := 0
                STA          BUFTAIL
                BEQ          $050
$040            INC          BUFHEAD                   ;  else BUFHEAD := BUFHEAD + 1
                LDA          BUFHEAD
                CMP          BUFSIZ                    ;If BUFHEAD >= BUFSIZ
                BCC          $050
                LDA          #0                        ;  then BUFHEAD := 0
                STA          BUFHEAD
$050            LDA          KABUF,Y                   ;Load the A and B port data
                LDX          KBBUF,Y
                CLC
$060            RTS
                .PAGE
;-----------------------------------------------------------------------
;
;  Subroutine SCRNECHO
;
;  This subroutine writes a single character to the screen.  On entry,
;  the character must be in A.  On exit, all registers are undefined.
;
;-----------------------------------------------------------------------
SCRNECHO        .EQU         *
                BIT          ECHO                      ;Screen Echo enabled?
                BPL          $010
                PHA
                JSR          CURSOR                    ;Remove cursor
                PLA
                JSR          PRINT                     ;Print the character
                JSR          CURSOR                    ;Restore cursor
$010            RTS
;
;
;-----------------------------------------------------------------------
;
;  Subroutine BACKSP
;
;  This subroutine performs the screen backspace when the console
;  deletes an input character.  On entry, the input buffer pointer
;  must point to the character to be deleted and the overflow flag
;  must be set to indicate that the character should be erased, or
;  clear to indicate that it should be left on the screen.  On exit,
;  all registers are undefined.
;
;-----------------------------------------------------------------------
BACKSP          .EQU         *
                LDA          ECHO
                BPL          $020                      ;Screen Echo not enabled
                LDY          #0
                LDA          (BUFFPTR),Y               ;Printable character?
                CMP          #ASC_SP
                BCC          $020
                PHP                                    ;Save overflow flag
                JSR          CURSOR                    ;Remove cursor
                LDA          #ASC_BS
                JSR          PRINT                     ;Backspace
                PLP
                BVC          $010                      ;Don't erase
                LDA          #ASC_SP
                JSR          PRINT                     ;Erase the character
                LDA          #ASC_BS
                JSR          PRINT
$010            JSR          CURSOR                    ;Restore cursor
$020            RTS

; ####################################################################################################
; #   END OF FILE:  CONS.MAIN.TEXT
; #   LINES      :  349
; #   CHARACTERS :  17347
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.READ.TEXT
; ####################################################################################################
```

```
                .PAGE
;-----------------------------------------------------------------------
;
;  Console Read Request
;
;    Parameters:
;       BUFFPTR:  Pointer to caller's data buffer
;       REQCNT:  Requested read count
;       RTNCNT:  Pointer to actual read count
;
;    Zero Page Temporary Storage
;       COUNT:  Number of bytes read
;       ONEBYTE:  TRUE if REQCNT = 1
;
;  If the ECHO or ESCAPE functions are enabled, this segment will call
;  PRINT to display a character or perform a screen control function.
;
;-----------------------------------------------------------------------
;
CNSLREAD        .EQU         *
;
;  Initialize read variables
;
                BIT          OPENFLG
                BMI          $010
                JMP          CNOTOPEN
$010            BIT          KYBDMODE                  ;Keyboard mode?
                BMI          $030
                LDA          SMCURSOR                  ;Save cursor status
                PHA
                BMI          $020
                LDA          #ASC_ENQ                  ;Turn on cursor
                JSR          SCRNECHO
$020            LDA          #FALSE
                STA          ONEBYTE                   ;Clear one byte read flag
                LDA          REQCNT+1
                BNE          $040
                LDA          REQCNT
                CMP          #1
                BNE          $040
                ROR          ONEBYTE                   ;Set one byte read flag
                BMI          $040
;
$030            LDA          REQCNT                    ;Make requested count even
                AND          #BITOFF0
                STA          REQCNT
;
$040            LDA          ESCAPE
                AND          #BITON7
                STA          ESCAPE                    ;Clear escape pending
                LDA          #0
                STA          COUNT
                STA          COUNT+1                   ;Zero bytes read count
                PHP
                SEI
                STA          SUSPFLSH                  ;Clear suspend & flush flags
                LDA          #KYBDDSBL
                STA          E_IER                     ;Mask the keyboard
                LDA          #TRUE
                STA          READING                   ;Set the READING flag
                PLP
;
;  Main read loop
;
CNSLLOOP        LDA          COUNT                     ;If COUNT >= REQCNT
                CMP          REQCNT                    ;  then goto CNSLEXIT
                LDA          COUNT+1
                SBC          REQCNT+1
                BCC          $020
$010            JMP          CNSLEXIT
;
$020            JSR          GETKEY                    ;Get next keystroke
                BCS          $010
                BIT          KYBDMODE                  ;Console or Keyboard mode?
                BPL          TSTESCAPE
;
;  Keyboard mode read
;
KYBDRDY         PHA                                    ;Save ASCII byte
                LDY          #0
                STA          (BUFFPTR),Y               ;Store data byte in buffer
                INY
                TXA
                STA          (BUFFPTR),Y               ;Store status byte in buffer
                LDA          #02
                JMP          BUMPCNT                   ;Go update COUNT and BUFFPTR
;
;  Console mode read
;
TSTESCAPE       BIT          ECHO                      ;Test for Escape Mode
                BPL          TSTCHDEL
                BIT          ESCAPE
                BPL          TSTCHDEL
                BVC          $040                      ;Escape not pending
                LDY          #ECMDCNT-1
                CMP          #"a"
                BCC          $010
                CMP          #"{"
                BCS          $010
                AND          #BITOFF5                  ;Upshift lower case alpha
$010            CMP          ESCCMD,Y                  ;Search for escape command
                BEQ          $020
                DEY
                BPL          $010
                ASL          ESCAPE                    ;Not found -- clear pending flag
                BCS          $030
$020            LDA          ESCOP,Y                   ;Get screen control character
                JSR          SCRNECHO
$030            JMP          CNSLLOOP
;
$040            CMP          #ASC_ESC                  ;Is this an ESC?
                BNE          TSTCHDEL
                ROR          ESCAPE                    ;Set escape pending
                BMI          $030
;
TSTCHDEL        BIT          ONEBYTE                   ;Test for character delete
                BMI          TSTLNDEL
                BIT          CHDELFLG
                BPL          TSTLNDEL
                CMP          #CHDELCHR
                BNE          TSTLNDEL
                LDA          COUNT                     ;Anything to delete?
                ORA          COUNT+1
                BEQ          $030
                LDA          COUNT
                BNE          $010
                DEC          COUNT+1                   ;Decrement current read count
$010            DEC          COUNT
                LDA          BUFFPTR
                BNE          $020
                DEC          BUFFPTR+1                 ;Decrement buffer pointer
$020            DEC          BUFFPTR
                JSR          BACKSP                    ;Backspace
$030            JMP          CNSLLOOP
;
TSTLNDEL        BIT          ONEBYTE                   ;Test for line delete
                BMI          TSTCHCPY
                BIT          LNDELFLG
                BPL          TSTCHCPY
                CMP          #LNDELCHR
                BNE          TSTCHCPY
                LDA          ECHO
                BPL          $050
                BVC          $040
;
$010            LDA          COUNT                     ;Anything to delete?
                ORA          COUNT+1
                BEQ          $060
                LDA          COUNT
                BNE          $020
                DEC          COUNT+1                   ;Decrement current read count
$020            DEC          COUNT
                LDA          BUFFPTR
                BNE          $030
                DEC          BUFFPTR+1                 ;Decrement buffer pointer
$030            DEC          BUFFPTR
                BIT          LNDELFLG
                JSR          BACKSP                    ;Backspace
                JMP          $010
;
$040            LDA          #"\"
                JSR          SCRNECHO                  ;Write "\ CR LF"
                LDA          #ASC_CR
                JSR          SCRNECHO
                LDA          #ASC_LF
                JSR          SCRNECHO
$050            SEC
                LDA          BUFFPTR                   ;Reset buffer pointer
                SBC          COUNT
                STA          BUFFPTR
                LDA          BUFFPTR+1
                SBC          COUNT+1
                STA          BUFFPTR+1
                LDA          #0                        ;Reset current read count
                STA          COUNT
                STA          COUNT+1
$060            JMP          CNSLLOOP
;
TSTCHCPY        BIT          ECHO                      ;Test for character copy
                BPL          CNSLRDY
                BIT          CHCPYFLG
                BPL          CNSLRDY
                CMP          #CHCPYCHR
                BNE          CNSLRDY
                JSR          SCRNPICK                  ;Copy character from screen
                ASL          A
                CMP          #40
                ROR          A
                EOR          #BITON7
;
CNSLRDY         PHA                                    ;Save character for new line test
                LDY          #0
                STA          (BUFFPTR),Y               ;Store character in buffer
;
                BIT          ECHO                      ;Echo enabled?
                BPL          $020
                BVS          $010
                CMP          #20                       ;Check for control character
                BCC          $020
$010            JSR          SCRNECHO
$020            LDA          #01
;
BUMPCNT         PHA
                CLC
                ADC          COUNT                     ;Update current read count
                STA          COUNT
                BCC          $010
                INC          COUNT+1
$010            PLA
                CLC
                ADC          BUFFPTR                   ;Update buffer pointer
                STA          BUFFPTR
                BCC          TSTNEWLN
                INC          BUFFPTR+1
                LDA          BUFFPTR+1
                CMP          #0FF
                BCC          TSTNEWLN
                SBC          #080                      ;Wrap buffer at FF page
                STA          BUFFPTR+1
                INC          1400+BUFFPTR+1
;
TSTNEWLN        PLA                                   ;Test for New Line
                BIT          NEWLINE
                BPL          $010
                CMP          NEWLNCHR
                BEQ          CNSLEXIT
$010            JMP          CNSLLOOP
;
CNSLEXIT        ASL          READING                  ;Clear the READING flag
                LDA          #KYBDENBL
                STA          E_IER                    ;Unmask the keyboard
                LDY          #0
                LDA          COUNT                    ;Return the actual byte count
                STA          (RTNCNT),Y
                INY
                LDA          COUNT+1
                STA          (RTNCNT),Y
                BIT          KYBDMODE
                BMI          $020
                PLA
                BMI          $010
                LDA          #ASC_ACK                 ;Turn off cursor
                JSR          SCRNECHO
$010            JSR          ZPOUT
$020            RTS

; ####################################################################################################
; #   END OF FILE:  CONS.READ.TEXT
; #   LINES      :  245
; #   CHARACTERS :  11718
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.WRIT.TEXT
; ####################################################################################################
```

```
                .PAGE
;-----------------------------------------------------------------------
;
;  Console Write Request
;
;    Parameters:
;       BUFFPTR:  Pointer to caller's data buffer
;       REQCNT:  Number of bytes to write
;
;    Zero Page Temporary Storage
;       COUNT:  Number of bytes written
;
;       Additional zero page data may be used to perform screen
;       control functions.
;
;-----------------------------------------------------------------------
;
CNSLWRIT        .EQU         *
                BIT          OPENFLG
                BMI          $010
                JMP          CNOTOPEN
$010            JSR          CURSOR                    ;Remove Cursor
                LDA          #0
                STA          COUNT                     ;Zero COUNT
                STA          COUNT+1
;
$020            LDA          COUNT                     ;Check for end of buffer
                CMP          REQCNT
                LDA          COUNT+1
                SBC          REQCNT+1
                BCS          $060                      ;Go Exit
$030            BIT          SUSPFLSH                  ;Check suspend and flush flags
                BMI          $030                      ;  Suspend
                BVS          $050                      ;  Flush
                LDY          #0
                LDA          (BUFFPTR),Y               ;Get next byte
                JSR          PRINT                     ;Print the byte
                INC          BUFFPTR
                BNE          $040                      ;Bump pointer
                INC          BUFFPTR+1
                BNE          $040
                LDA          #80
                STA          BUFFPTR+1                 ;Process buffer wrap around
                INC          1400+BUFFPTR+1
$040            INC          COUNT
                BNE          $020                      ;Bump bytes read count
                INC          COUNT+1
                JMP          $020
;
$050            LDA          #00
                STA          CTLINDX                   ;Clear any pending cntl function
;
$060            JSR          CURSOR                    ;Restore cursor
                JMP          ZPOUT                     ;Save Zero Page data and exit
                .PAGE
;-----------------------------------------------------------------------
;
;  Subroutine PRINT
;
;  This routine processes a single byte of output.  Characters are
;  printed by calling DISPLAY.  Screen control functions are processed
;  by accumulating any required parameters in CTLBUFF then switching
;  to the appropriate screen control routine.
;
;    Parameters:
;       A:  The byte to process
;
;    Exit:
;       A, X, Y:  Undefined
;
;-----------------------------------------------------------------------
;
PRINT           .EQU         *
                LDY          CTLINDX                   ;Get control function index
                BNE          $010
                ORA          DSPLYCTL
                CMP          #ASC_SP                   ;Display or control?
                BCS          DISPLAY
                TAX
                LDA          QUOTATBL,X                ;Get function quota
                BEQ          $020
                STA          CTLQUOTA
                TXA
$010            STA          CTLBUFF,Y                 ;Save function character
```

```
                INY
                STY          CTLINDX                   ;Update buffer index
                CPY          CTLQUOTA                  ;See if quota filled
                BCC          $020
                LDY          #0
                STY          CTLINDX                   ;Zero buffer index
                SWITCH       CTLBUFF,,CTLSWTBL,*
$020            RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine DISPLAY
;
;  This routine displays a single character.  If auto advance is
;  enabled, it calls CF_HT to advance the cursor.
;
;    Parameters:
;       A:  The character to be displayed
;
;    Exit:
;       A, X, Y:  Undefined
;
;----------------------------------------------------------------------
;
DISPLAY         .EQU         *
                ORA          #80                       ;set hi-bit
                EOR          SMINV                     ;set normal or inverse
                PHA                                    ;(for safe keeping)
                BIT          HMODE                     ;80 column text?
                BPL          $010
                LDA          TPX
                LSR          A                         ;80 col: X=TPX/2
                TAY                                    ;carry bit clear?
                BCC          $020                      ;yes: use page 1
                PLA
                STA          (BASE2),Y                 ;80 col page two
                BCS          $030
$010            LDY          TPX                       ;40 col: X=TPX
                LDA          TCOLOR
                STA          (BASE2),Y                 ;set color byte
$020            PLA
                STA          (BASE1),Y                 ;80 col page one
$030            BIT          SMAUTOADV                 ;if auto advance,
                BPL          $040
                JMP          CF_HT                     ;advance cursor
```

```
$040            RTS
                .PAGE
;----------------------------------------------------------------------
;
;   Control Function Quota and Switch Tables
;
;----------------------------------------------------------------------
QUOTATBL        .EQU         *                         ;The Control Function Quota Table
                .BLOCK       1,0                       ;  contains the total number of
                .BLOCK       15.,1                     ;  bytes required by the function,
                .BLOCK       1,2                       ;  including the function character
                .BLOCK       2,1                       ;  itself.  A zero indicates that
                .BLOCK       2,2                       ;  the function is unimplemented.
                .BLOCK       1,2
                .BLOCK       1,1
                .BLOCK       3,2
                .BLOCK       1,3
                .BLOCK       1,0
                .BLOCK       4,1
CTLSWTBL        .EQU         *
                .WORD        CF_NUL-1                  ;00  no-op
                .WORD        CF_SOH-1                  ;01  Save Environment & Release Viewport
                .WORD        CF_STX-1                  ;02  Set Viewport Upper Left
                .WORD        CF_ETX-1                  ;03  Set Viewport Lower Right
                .WORD        CF_EOT-1                  ;04  Restore Environment
                .WORD        CF_ENQ-1                  ;05  Cursor On
                .WORD        CF_ACK-1                  ;06  Cursor Off
                .WORD        CF_BEL-1                  ;07  Audible signal
                .WORD        CF_BS-1                   ;08  Backspace
                .WORD        CF_HT-1                   ;09  Forward Space
                .WORD        CF_LF-1                   ;0A  Line Feed
                .WORD        CF_VT-1                   ;0B  Reverse Line Feed
                .WORD        CF_FF-1                   ;0C  Home Cursor
                .WORD        CF_CR-1                   ;0D  Carriage Return
                .WORD        CF_SO-1                   ;0E  Screen Off
                .WORD        CF_SI-1                   ;0F  Screen On
                .WORD        CF_DLE-1                  ;10  Set Text Mode
                .WORD        CF_DC1-1                  ;11  Normal Video
                .WORD        CF_DC2-1                  ;12  Inverse Video
                .WORD        CF_DC3-1                  ;13  Foreground Color
                .WORD        CF_DC4-1                  ;14  Background Color
                .WORD        CF_NAK-1                  ;15  Set Text Options
                .WORD        CF_SYN-1                  ;16  Sync on VBL
                .WORD        CF_ETB-1                  ;17  Horizontal Shift
                .WORD        CF_CAN-1                  ;18  Go to X
```

```
                        .WORD        CF_EM-1                   ;19  Go to Y
                        .WORD        CF_SUB-1                  ;1A  Go to X,Y
                        .WORD        CF_ESC-1                  ;1B  No-op
                        .WORD        CF_FS-1                   ;1C  Clear Screen
                        .WORD        CF_GS-1                   ;1D  Clear to End of Screen
                        .WORD        CF_RS-1                   ;1E  Clear Line
                        .WORD        CF_US-1                   ;1F  Clear to End of Line


; ####################################################################################################
; #   END OF FILE:  CONS.WRIT.TEXT
; #   LINES      :  182
; #   CHARACTERS :  10016
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.FCTN.TEXT
; ####################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;  Screen Control Functions
;
;  These routines perform all screen control functions.
;
;    Parameters:
;       All parameters are accumulated in CTLBUFF
;
;    Exit:
;       A, X, Y:  Undefined
;
;----------------------------------------------------------------------
;
CF_NUL          .EQU         *
CF_ESC          .EQU         *
                RTS                                     ;NO-OP
;
CF_SOH          .EQU         *                          ;Save & Release Viewport
                LDY          #SCRSTLEN
$010            LDA          SCRSTTBL-1,Y
                STA          SCRSTSAV-1,Y
                DEY
                BNE          $010
                CLC
                LDA          TPX
                ADC          VPL
                STA          TPX                        ;retain X posn
                LDA          TPY
                ADC          VPT
                STA          TPY                        ;retain y posn
                LDA          #0
                STA          VPL                        ;zero left margin
                STA          VPT                        ;zero top margin
                LDA          #0FF
                STA          VPR                        ;Let VERIFY set the right edge
                STA          VPB                        ;  and bottom margin
                JMP          VERIFY
```

```
;
CF_STX          .EQU         *                          ;SET VIEWPORT UPPER LEFT
                CLC
                LDA          VPL
                ADC          TPX                        ;at cursor posn
                STA          VPL                        ;set left margin
                LDA          VPT
                ADC          TPY
                STA          VPT                        ;set top margin
                LDA          #0
                STA          TPX                        ;reset cursor X
                STA          TPY                        ;  and cursor Y
                JMP          VERIFY                     ;and verify
;
CF_ETX          .EQU         *                          ;SET VIEWPORT LOWER RIGHT
                CLC
                LDA          TPX
                ADC          VPL
                STA          VPR                        ;set left margin
                LDA          TPY
                ADC          VPT                        ;& bottom margin
                STA          VPB
                JMP          VERIFY                     ;and verify
;
CF_EOT          .EQU         *                          ;RESTORE VIEWPORT
                LDY          #SCRSTLEN
$010            LDA          SCRSTSAV-1,Y
                STA          SCRSTTBL-1,Y
                DEY
                BNE          $010
                JMP          VERIFY
;
CF_ENQ          .EQU         *                          ;ENABLE CURSOR
                LDA          SMODE
                ORA          #BITON4
                STA          SMODE
                LDA          #TRUE
                STA          SMCURSOR
                RTS
;
CF_ACK          .EQU         *                          ;DISABLE CURSOR
                LDA          SMODE
                AND          #BITOFF4
                STA          SMODE
                LDA          #FALSE
                STA          SMCURSOR
                RTS
;
CF_BEL          .EQU         *                         ;Sound Bell
                BIT          BELL
                RTS
;
CF_BS           .EQU         *                         ;BACKSPACE
                DEC          TPX
                BPL          $020
                BIT          SMAUTOCR                  ;BS at left:
                BPL          $010
                LDA          VPHMAX                    ;Wrap to right
                STA          TPX                       ;  edge of viewport
                JMP          CF_VT
$010            INC          TPX
$020            RTS
;
CF_HT           .EQU         *                         ;ADVANCE
                LDA          TPX
                CMP          VPHMAX
                BCS          $010                      ;at edge?
                INC          TPX                       ;no: advance
                RTS
$010            BIT          SMAUTOCR                  ;auto CR on?
                BPL          CF_EXIT
                LDA          #0                        ;yes: wrap to
                STA          TPX                       ;left margin
                JMP          CF_LF                     ;& line feed
;
CF_LF           .EQU         *                         ;LINE FEED
                LDA          TPY
                CMP          VPVMAX
                BCS          $010                      ;at edge?
                INC          TPY                       ;no: move down
                JMP          TBASCAL                   ;calc base address
$010            BIT          SMSCROLL                  ;auto scroll?
                BPL          CF_EXIT
                LDA          #00
                JMP          SCROLL                    ;yes: go to it
;
CF_VT           .EQU         *                         ;REVERSE LINE FEED
                LDA          TPY
                BEQ          $010                      ;at top?
                DEC          TPY                       ;no: do it
                JMP          TBASCAL                   ;calc base address
$010            BIT          SMSCROLL                  ;auto scroll?
                BPL          CF_EXIT
                LDA          #80
                JMP          SCROLL
;
CF_FF           .EQU         *                         ;FORM FEED
                LDA          #0
                STA          TPX                       ;reset TPX
                STA          TPY                       ;  and TPY
                JMP          TBASCAL                   ;calc base address
;
CF_CR           .EQU         *                         ;CARRIAGE RETURN
                LDA          #0
                STA          TPX                       ;reset TPX
                BIT          SMAUTOLF                  ;auto LF set?
                BPL          CF_EXIT
                JMP          CF_LF                     ;yes: go to it
;
CF_SO           .EQU         *                         ;SCREEN OFF
                LDA          #FALSE
                STA          SCRNMODE
                JMP          VERIFY
;
CF_SI           .EQU         *                         ;SCREEN ON
                LDA          #TRUE
                STA          SCRNMODE
                JMP          VERIFY
;
CF_DLE          .EQU         *                         ;SET HARDWARE MODE
                LDA          CTLBUFF+1
                STA          HMODE
                JMP          VERIFY
;
CF_DC1          .EQU         *                         ;NORMAL VIDEO
                LDA          SMODE
                AND          #BITOFF5                  ;reset INVERSE bit
                STA          SMODE
                LDA          #FALSE
                STA          SMINV
CF_EXIT         RTS
;
CF_DC2          .EQU         *                         ;INVERSE VIDEO
                LDA          SMODE
                ORA          #BITON5                   ;set INVERSE bit
                STA          SMODE
                LDA          #TRUE
                STA          SMINV
                RTS
;
CF_DC3          .EQU         *                         ;FOREGROUND COLOR
                LDA          CTLBUFF+1
                STA          TCF
                JMP          VERIFY                    ;set TCOLOR
;
CF_DC4          .EQU         *                         ;BACKGROUND COLOR
                LDA          CTLBUFF+1
                STA          TCB
                JMP          VERIFY                    ;set TCOLOR
;
CF_NAK          .EQU         *                         ;SET SOFTWARE MODE
                LDA          CTLBUFF+1
                AND          #0F
                STA          CTLBUFF+1
                LDA          SMODE                     ;Save bits 7-4
                AND          #0F0
                ORA          CTLBUFF+1
                STA          SMODE
                JMP          VERIFY
;
CF_SYN          .EQU         *                         ;SYNCHRONIZE WITH VBL
                LDA          #SYNCSSIZ
                LDX          SYNCSADR
                LDY          SYNCSADR+1
                JSR          ALLOCSIR                  ;Allocate CB2 for VBL
                BCS          CF_EXIT
                JSR          CURSOR                    ;Restore cursor while waiting
                PHP
                SEI
                LDA          E_PCR
                AND          #1F
                ORA          #60                       ;Set up CB2 to monitor
                STA          E_PCR                     ;  VBL positive edge
                LDA          #08
                STA          E_IER
                STA          E_IFR
                PLP
$010            BIT          E_IFR                     ;Wait for VBL edge
                BEQ          $010
                JSR          CURSOR                    ;Remove cursor
                LDA          #SYNCSSIZ
                LDX          SYNCSADR
                LDY          SYNCSADR+1
                JMP          DEALCSIR                  ;Release CB2 resource
;
CF_ETB          .EQU         *                         ;HORIZONTAL SCROLL
                LDA          CTLBUFF+1
$010            JMP          SHIFT
;
CF_CAN          .EQU         *                         ;Go To X
                LDA          CTLBUFF+1
                CMP          VPHMAX                    ;out of range?
                BCC          $010
                LDA          VPHMAX                    ;Set to right margin
$010            STA          TPX
                RTS
;
CF_EM           .EQU         *                         ;Go To Y
                LDA          CTLBUFF+1
                CMP          VPVMAX                    ;out of range?
                BCC          $010
                LDA          VPVMAX                    ;Set to top
$010            STA          TPY
                JMP          TBASCAL                   ;get base address
;
CF_SUB          .EQU         *                         ;Go To X, Y
                JSR          CF_CAN
                LDA          CTLBUFF+2
                STA          CTLBUFF+1
                JMP          CF_EM
;
CF_FS           .EQU         *                         ;CLEAR SCREEN
                JSR          CF_FF
                JMP          CLREOS
;
CF_GS           .EQU         *                         ;CLEAR TO EOS
                JMP          CLREOS
;
CF_RS           .EQU         *                         ;CLEAR LINE
                LDA          #0
                STA          TPX
                JMP          CLREOL
;
CF_US           .EQU         *                         ;CLEAR TO EOL
                JMP          CLREOL
```

```
; ####################################################################################################
; #   END OF FILE:  CONS.FCTN.TEXT
; #   LINES      :  265
; #   CHARACTERS :  12878
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ######################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.STAT.TEXT
; ######################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;  Console Status Request
;
;    Parameters:
;       SCCODE:  Status / Control code
;       SCLIST:  Pointer to caller's status / control list
;
;       Before switching to the appropriate request handling code,
;       Y is set to zero.
;
;----------------------------------------------------------------------
;
CNSLSTAT        .EQU         *
                BIT          OPENFLG                   ;Is the Console open?
                BMI          $010
                JMP          CNOTOPEN
$010            SWITCH       SCCODE,18.,CSTATSW,*
                BCS          CBADCTL
                LDY          #0
                RTS
;
CBADCTL         LDA          #XCTLCODE                 ;Invalid control code
                JSR          SYSERR
;
CSTATSW         .WORD        CSTAT00-1
                .WORD        CSTAT01-1
                .WORD        CSTAT02-1
                .WORD        CSTAT03-1
                .WORD        CSTAT04-1
                .WORD        CSTAT05-1
                .WORD        CSTAT06-1
                .WORD        CBADCTL-1
                .WORD        CSTAT08-1
                .WORD        CSTAT09-1
                .WORD        CSTAT10-1
                .WORD        CSTAT11-1
                .WORD        CSTAT12-1
                .WORD        CSTAT13-1
                .WORD        CSTAT14-1
                .WORD        CSTAT15-1
                .WORD        CSTAT16-1
                .WORD        CSTAT17-1
                .WORD        CSTAT18-1
;
CSTAT00         RTS                                  ;0 -- NOP
;
CSTAT01         .EQU         *                       ;1 -- Console Status Table
                LDA          (SCLIST),Y
                CMP          #CONSTLEN
                BCS          $010
                LDA          #XCTLPARM
                JSR          SYSERR
$010            LDA          #CONSTLEN
                STA          (SCLIST),Y
                TAY
$020            LDA          CONSTTBL-1,Y
                STA          (SCLIST),Y
                DEY
                BNE          $020
                RTS
;
CSTAT02         .EQU         *                       ;2 -- New Line
                LDA          NEWLINE
                STA          (SCLIST),Y
                INY
                LDA          NEWLNCHR
                STA          (SCLIST),Y
                RTS
;
CSTAT03         .EQU         *                       ;3 -- Console / Keyboard mode
                LDA          KYBDMODE
                STA          (SCLIST),Y
                RTS
;
CSTAT04         .EQU         *                       ;4 -- Buffer Size
                LDA          BUFSIZ
                STA          (SCLIST),Y
                RTS
;
CSTAT05         .EQU         *                       ;5 -- Current Key Count
                LDA          KEYCNT
                STA          (SCLIST),Y
                RTS
;
CSTAT06         LDY          #5                         ;6 -- Attention Event
$010            LDA          ATTNEVNT,Y
                STA          (SCLIST),Y
                DEY
                BPL          $010
                RTS
;
CSTAT08         LDY          #4                         ;8 -- Any Key Event
$010            LDA          ANYKYEVNT,Y
                STA          (SCLIST),Y
                DEY
                BPL          $010
                RTS
;
CSTAT09         .EQU         *                          ;09 -- Read Screen with norm/inv
                JSR          SCRNPICK
                EOR          #BITON7
                EOR          SMCURSOR
                LDY          #0
                STA          (SCLIST),Y
                RTS
;
CSTAT10         .EQU         *                          ;10 -- No Wait Input
                LDA          NOWAIT
                STA          (SCLIST),Y
                RTS
;
CSTAT11         .EQU         *                          ;11 -- Screen Echo
                LDA          ECHO
                STA          (SCLIST),Y
                RTS
;
CSTAT12         .EQU         *                          ;12 -- Character Copy
                LDA          CHCPYFLG
                STA          (SCLIST),Y
                RTS
;
CSTAT13         .EQU         *                          ;13 -- Character Delete
                LDA          CHDELFLG
                STA          (SCLIST),Y
                RTS
;
CSTAT14         .EQU         *                          ;14 -- Line Delete
```

```
                LDA          LNDELFLG
                STA          (SCLIST),Y
                RTS
;
CSTAT15         .EQU         *                         ;15 -- Escape Functions
                LDA          ESCAPE
                STA          (SCLIST),Y
                RTS
;
CSTAT16         .EQU         *                         ;16 -- Cursor Position
                LDA          TPX
                STA          (SCLIST),Y
                INY
                LDA          TPY
                STA          (SCLIST),Y
                RTS
;
CSTAT17         .EQU         *                         ;17 -- Pick Character
                JSR          SCRNPICK
                ASL          A
                CMP          #40
                ROR          A
                EOR          #BITON7
                LDY          #0
                STA          (SCLIST),Y
                RTS
;
CSTAT18         .EQU         *                         ;18 -- Screen Dump
                LDA          HMODE
                STA          (SCLIST),Y
                INY
                LDA          VPHMAX
                STA          (SCLIST),Y
                INY
                LDA          VPVMAX
                STA          (SCLIST),Y
                LDA          #00
                BIT          DSPLYCTL
                BMI          $010
                JMP          SCRNDUMP
$010            STA          (SCLIST),Y                ;If control characters are being
                DEY                                    ;  displayed, dump a null viewport
                STA          (SCLIST),Y
                RTS

```

```
; ####################################################################################################
; #   END OF FILE:  CONS.STAT.TEXT
; #   LINES      :  174
; #   CHARACTERS :  7340
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.CNTL.TEXT
; ####################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;  Console Control Request
;
;    Parameters:
;       SCCODE:  Status / Control code
;       SCLIST:  Pointer to caller's status / control list
;
;       Before switching to the appropriate request handler, Y is
;       set to zero and A is loaded with the first byte of the list.
;
;----------------------------------------------------------------------
;
CNSLCNTL        .EQU         *
                BIT          OPENFLG                   ;Console open?
                BPL          $010
                SWITCH       SCCODE,18.,CCNTLSW,*
                BCS          $020
                LDY          #00
                LDA          (SCLIST),Y
                RTS
;
$010            JMP          CNOTOPEN
;
$020            JMP          CBADCTL
;
CCNTLSW         .WORD        CCNTL00-1
                .WORD        CCNTL01-1
                .WORD        CCNTL02-1
                .WORD        CCNTL03-1
                .WORD        CCNTL04-1
                .WORD        CCNTL05-1
                .WORD        CCNTL06-1
                .WORD        CBADCTL-1
                .WORD        CCNTL08-1
                .WORD        CBADCTL-1
                .WORD        CCNTL10-1
                .WORD        CCNTL11-1
                .WORD        CCNTL12-1
                .WORD        CCNTL13-1
                .WORD        CCNTL14-1
                .WORD        CCNTL15-1
                .WORD        LOADSET-1
                .WORD        LOAD8-1
                .WORD        CCNTL18-1
;
CCNTL00         LDA          E_IER                     ;0 -- Reset
                PHA                                    ;Save current interrupt state
                LDA          #KYBDDSBL                 ;  and mask off interrupts
                STA          E_IER
                LDA          #BUFMAX
                STA          BUFSIZ                    ;Set buffer size to maximum
                LDA          #00
                STA          KEYCNT                    ;Flush buffer
                STA          BUFHEAD
                STA          BUFTAIL
                STA          READING                   ;No read in progress
                STA          ANYKYEVNT                 ;Disable any key event
                STA          ATTNEVNT                  ;Disable attention event
                STA          CTLINDX                   ;Abort control function in progress
                STA          DSPLYCTL                  ;Clear display control char. flag
                STA          SUSPFLSH                  ;Clear suspend & flush output flags
                JSR          CURSOR                    ;Remove cursor
                LDX          #DFLTLEN
$010            LDA          DFLTVAL-1,X               ;Copy configuration block
                STA          DFLTTBL-1,X
                DEX
                BNE          $010
                JSR          CF_SOH                    ;Save screen state & verify
                JSR          CURSOR                    ;Restore the cursor
                JSR          ZPOUT                     ;Save screen zero page
                PLA
                AND          #KYBDENBL                 ;Restore previous interrupt state
                ORA          #BITON7
                STA          E_IER
                RTS
;
CCNTL01         .EQU         *                         ;1 -- Console Status Table
                CMP          #CONSTLEN
                BEQ          $010
                LDA          #XCTLPARM
                JSR          SYSERR
$010            JSR          CURSOR
```

```
                LDY          #CONSTLEN
$020            LDA          (SCLIST),Y
                DEY
                STA          CONSTTBL,Y
                BNE          $020
                JSR          VERIFY
                JSR          CURSOR
                JSR          ZPOUT
                RTS
;
CCNTL02         .EQU         *                          ;2 -- New Line
                AND          #BITON7
                STA          NEWLINE
                INY
                LDA          (SCLIST),Y
                STA          NEWLNCHR
                RTS
;
CCNTL03         .EQU         *                          ;3 -- Console / Keyboard mode
                AND          #BITON7
                STA          KYBDMODE
                RTS
;
CCNTL04         .EQU         *                          ;4 -- Buffer Size
                CMP          #BUFMAX+1
                BCC          $010
                LDA          #XCTLPARM
                JSR          SYSERR
$010            LDX          #KYBDDSBL
                STX          E_IER
                STY          KEYCNT
                STY          BUFHEAD
                STY          BUFTAIL
                STA          BUFSIZ
                LDX          #KYBDENBL
                STX          E_IER
                RTS
;
CCNTL05         LDA          E_IER                      ;5 -- Flush Buffer
                PHA
                LDA          #KYBDDSBL
                STA          E_IER
                STY          KEYCNT
                STY          BUFHEAD
                STY          BUFTAIL
                PLA
                AND          #KYBDENBL
                ORA          #BITON7
                STA          E_IER
                RTS
;
CCNTL06         PHP                                  ;6 -- Attention Event
                SEI
                LDY          #5
$010            LDA          (SCLIST),Y
                STA          ATTNEVNT,Y
                DEY
                BPL          $010
                PLP
                RTS
;
CCNTL08         PHP                                  ;8 -- Any Key Event
                SEI
                LDY          #4
$010            LDA          (SCLIST),Y
                STA          ANYKYEVNT,Y
                DEY
                BPL          $010
                PLP
                RTS
;
CCNTL10         .EQU         *                       ;10 -- No Wait Input
                AND          #BITON7
                STA          NOWAIT
                RTS
;
CCNTL11         .EQU         *                       ;11 -- Screen Echo
                AND          #BITON7+BITON6
                STA          ECHO
                RTS
;
CCNTL12         .EQU         *                       ;12 -- Character Copy
                AND          #BITON7
                STA          CHCPYFLG
                RTS
;
CCNTL13         .EQU         *                       ;13 -- Character Delete
                AND          #BITON7+BITON6
                STA          CHDELFLG
                RTS
```

```
;
CCNTL14         .EQU         *                           ;14 -- Line Delete
                AND          #BITON7+BITON6
                STA          LNDELFLG
                RTS
;
CCNTL15         .EQU         *                           ;15 -- Escape Functions
                AND          #BITON7
                STA          ESCAPE
                RTS
;
CCNTL18         .EQU         *                           ;18 -- Restore contents of viewport
                BIT          DSPLYCTL
                BMI          $020
                INY
                EOR          HMODE
                BMI          $010
                LDA          (SCLIST),Y
                CMP          VPHMAX
                BNE          $010
                INY
                LDA          (SCLIST),Y
                CMP          VPVMAX
                BNE          $030
                LDA          #80
                JMP          SCRNDUMP
;
$010            LDA          (SCLIST),Y
                INY
                ORA          (SCLIST),Y
                BNE          $030
$020            RTS
;
$030            LDA          #XCTLPARM
                JSR          SYSERR


; ####################################################################################################
; #   END OF FILE:  CONS.CNTL.TEXT
; #   LINES      :  210
; #   CHARACTERS :  9290
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.DNLD.TEXT
; ####################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine LOADCHR
;
;  This subroutine is called to load an ASCII code and a character
;  image into one of the character download cells in the text pages.
;
;  LOADCHR requires four bytes of zero page storage for pointers. In
;  order to make it callable from either a device handler or an
;  interrupt processor, all zero page references are indexed by X.
;  On entry, the X register must contain the zero page offset to the
;  character image pointer.  The two bytes following the image
;  pointer are used to address the download locations in the text
;  page.
;
;     Input Parameters:
;        DNLDCEL  -- character download cell number: [0,7]
;        DNLDCHR  -- ASCII character code: [0,7F]
;        X reg    -- zero page offset to pointers
;                    (0,X) image pointer set by caller
;                    (2,X) download cell pointer set by LOADCHR
;
;  On exit, DNLDCEL, DNLDCHR, and X will be unchanged.  The image
;  pointer will have been incremented by eight.  A and Y are destroyed.
;
;----------------------------------------------------------------------
;
DIMGPTR         .EQU         00                        ;Zero page pointer to image
DCELPTR         .EQU         02                        ;Zero page pointer to cell
;
LOADCHR         .EQU         *
                LDY          #00                       ;Use Y for row counter
$010            LDA          DNLDCEL                   ;Set up cell pointer
                AND          #03                       ;  for ASCII code
                ORA          DCPTRL,Y
                STA          DCELPTR,X
                LDA          DNLDCEL
                LSR          A
                LSR          A
                CPY          #04
                ROL          A
                ORA          #08
                STA          DCELPTR+1,X
                LDA          DNLDCHR                   ;Store ASCII code into
                STA          (DCELPTR,X)               ;  download cell
                LDA          DCELPTR+1,X               ;Fix cell pointer
                EOR          #0C                       ;  for character image
                STA          DCELPTR+1,X
                LDA          (DIMGPTR,X)               ;Store character image
                STA          (DCELPTR,X)               ;  into download cell
                INC          DIMGPTR,X                 ;Increment the image pointer
                BNE          $020
                INC          DIMGPTR+1,X
$020            INY                                    ;Increment the row number
                CPY          #08
                BCC          $010                      ;Not done yet
                RTS
;
DCPTRL          .EQU         *                         ;Table of download cell addresses
                .BYTE        078,07C,0F8,0FC
                .BYTE        078,07C,0F8,0FC
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine DNLDINT
;
;  This subroutine processes the VBL interrupt that signals the
;  completion of a character download cycle.  If the REQUEST bit of
;  DNLDFLG is set, another block of eight characters will be
;  downloaded; otherwise, the CB1 and CB2 resources will be
;  released and the ACTIVE bit will be cleared.  DNLDINT assumes
;  that the X register points to a four byte area on the zero page
;  that can be used for LOADCHR.
;
;----------------------------------------------------------------------
;
DNLDINT         .EQU         *
                BIT          DNLDDSBL                  ;Disable download
                LDA          #VBLDSBL
                STA          E_IER                     ;Mask VBL interrupts
                BIT          DNLDFLG                   ;Test REQUEST bit
                BVC          $030
                CLI                                    ;Enable interrupts
                LDA          #07
                STA          DNLDCEL                   ;Start with cell 7
                LDA          DNLDIMG
                STA          DIMGPTR,X                 ;Set up IMAGE pointer
                LDA          DNLDIMG+1
                STA          DIMGPTR+1,X
$010            JSR          LOADCHR                   ;Load one character image
                INC          DNLDCHR                   ;Bump character code
                BPL          $020
                ASL          DNLDFLG                   ;Clear REQUEST bit
$020            DEC          DNLDCEL                   ;Bump cell number
                BPL          $010                      ;More to do
                LDA          DIMGPTR,X
                STA          DNLDIMG                   ;Save IMAGE pointer
                LDA          DIMGPTR+1,X
                STA          DNLDIMG+1
                JMP          DNLD_GO                   ;Enable downloading
;
$030            ASL          DNLDFLG                   ;Clear ACTIVE bit
                LDA          #DNLDSSIZ
                LDX          DNLDSADR
                LDY          DNLDSADR+1
                JSR          DEALCSIR                  ;Deallocate SIRs
                RTS
                .PAGE
;---------------------------------------------------------------------
;
;  Subroutine GETSIRS
;
;  This subroutine allocates SIRs 5 & 6 and initializes them to
;  monitor VBL for character downloading.  If the SIRs can not be
;  allocated, it sets an error code and returns directly to the
;  dispatcher.
;
;---------------------------------------------------------------------
;
GETSIRS         .EQU         *
                BIT          DNLDFLG                   ;Wait for any previous
                BMI          GETSIRS                   ;  request to finish
                LDA          #DNLDSSIZ
                LDX          DNLDSADR
                LDY          DNLDSADR+1
                JSR          ALLOCSIR
                BCS          $010
                PHP
                SEI
                LDA          E_PCR                     ;Set CB1 to monitor VBL
                AND          #0F                       ;  negative edge and
                ORA          #60                       ;  CB2 to monitor
                STA          E_PCR                     ;  positive edge
                LDA          #VBLDSBL
                STA          E_IER
                PLP
                RTS
;
$010            PLA                                    ;  pull caller's
                PLA                                    ;  address, and
                LDA          #XNORESRC                 ;  return to dispatcher
                JSR          SYSERR                    ;  with an error
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine LOADSET
;
;  This subroutine is called to initiate downloading of the entire
;  text screen character set.  LOADSET calls GETSIRS to set up the
;  6522 to monitor VBL and interrupt on the negative edge.  It then
;  copys the character set to the screen's local data area, sets the
;  request bit, and enables the VBL interrupt.  The VBL interrupt
;  processor, DNLDINT, will complete the actual downloading.
;
;    Parameters:
;       SCLIST:  Pointer to caller's 1024 byte character set
;
;    Zero Page Temporary Storage:
;       WORK1:  Pointer to system's character set
;
;----------------------------------------------------------------------
;
LOADSET         .EQU         *
                JSR          GETSIRS
                LDA          #TEXTCSA%100
                STA          WORK1
                STA          DNLDIMG
                LDA          #TEXTCSA/100
                STA          WORK1+1
                STA          DNLDIMG+1
                LDA          #ASC_NUL
                STA          DNLDCHR
                LDX          #4                        ;Set X to move 4 pages
                LDY          #0                        ;Set Y to move full page
                LDA          SCLIST+1
                CMP          #0FB
                BCC          $010
                SBC          #080                      ;Adjust address to avoid
                STA          SCLIST+1                  ;  bank wrap around
                INC          1400+SCLIST+1
$010            LDA          (SCLIST),Y                ;Copy character set to
                STA          (WORK1),Y                 ;  text char set buffer
                INY
                BNE          $010
                INC          SCLIST+1
                INC          WORK1+1
                DEX
                BNE          $010
                LDA          #0C0                      ;Set download active
                STA          DNLDFLG                   ;  and request flags
                LDA          #VBLENBL
                STA          E_IER                     ;Enable interrupts in VBL neg
                RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine LOAD8
;
;  This subroutine is called to download up to eight text character
;  images.  LOAD8 calls GETSIRS to set up the 6522 to monitor VBL
;  and interrupt on the negative edge.  It then loads the character
;  images into the screen's download cells and enables downloading
;  and the VBL interrupt.  The download operation is completed by
;  the interrupt processor DNLDINT.
;
;    Parameters:
;       SCLIST:  Pointer to caller's character sets
;
;    Zero Page Temporary Data:
;       COUNT:  Number of characters to download
;       WORK1:  Pointer to character image for LOADCHR
;       WORK2:  Work area for LOADCHR
;
;----------------------------------------------------------------------
;
LOAD8           .EQU         *
                CMP          #01                       ;Check download count
                BCS          $010
                RTS
;
$010            CMP         #09
                BCC         $020
                LDA         #XCTLPARM                 ;Too many
                JSR         SYSERR
;
$020            STA         COUNT
                JSR         GETSIRS
;
                INC         SCLIST                    ;Bump list address
                BNE         $030                      ;  to first character
                INC         SCLIST+1
;
$030            LDA         #08
                STA         DNLDCEL
;
$040            LDY         #00
                LDA         (SCLIST),Y                ;Get character code
                STA         DNLDCHR
                INC         SCLIST                    ;Bump list address
                BNE         $050                      ;  to character image
                INC         SCLIST+1
;
$050            LDA         #03
                STA         WORK1+1
                LDA         DNLDCHR
                ASL         A
                ASL         A
                ROL         WORK1+1                   ;Set up address of character
                ASL         A                         ;  image in C00 to FFF space
                ROL         WORK1+1
                STA         WORK1
;
                LDY         #07
$060            LDA         (SCLIST),Y                ;Copy character image
                STA         (WORK1),Y                 ;  to C00 image space
                DEY
                BPL         $060
;
                DEC         DNLDCEL
                LDX         #WORK1
                JSR         LOADCHR                   ;Download this character
;
                LDA         DNLDCEL
                CMP          COUNT
                BCS          $050                      ;Do same character again
                LDA          #08
                ADC          SCLIST                    ;Bump list address
                STA          SCLIST                    ;  to next character
                BCC          $070
                INC          SCLIST+1
$070            DEC          COUNT
                BNE          $040
;
                LDA          #080                      ;Set download active
                STA          DNLDFLG
DNLD_GO         BIT          DNLDENBL
                LDA          #VBLCLR
                STA          E_IFR                     ;Clear both VBL flags
$080            BIT          E_IORB                    ;Check composite blanking
                BVC          $090
                BIT          E_IFR                     ;Check VBL flags
                BEQ          $080
$090            STA          E_IFR                     ;Clear VBL flags
                LDA          #VBLENBL                  ;Enable VBL interrupt
                STA          E_IER
                RTS


; ####################################################################################################
; #   END OF FILE:  CONS.DNLD.TEXT
; #   LINES      :  288
; #   CHARACTERS :  14806
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.MISC.TEXT
; ####################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;  Console Open Request
;
;----------------------------------------------------------------------
;
CNSLOPEN        .EQU         *
                BIT          OPENFLG                   ;Console open?
                BPL          $010                      ;  No
                LDA          #XNOTAVIL
                JSR          SYSERR
;
$010            LDA          #KYBDSSIZ                 ;Allocate the keyboard interrupt
                LDX          KYBDSADR
                LDY          KYBDSADR+1
                JSR          ALLOCSIR
                BCS          $020
                LDA          #TRUE
                STA          OPENFLG                   ;Set console open
                JSR          CCNTL00                   ;Reset console parameters
                PHP
                SEI
                LDA          E_PCR
                AND          #0F1
                ORA          #002                      ;Set up keyboard interrupt
                STA          E_PCR
                PLP
                LDA          #KYBDCLR
                STA          E_IFR                     ;Clear keyboard flag
                BIT          KYBDSTRB                  ;Clear the keyboard strobe
                LDA          #KYBDENBL
                STA          E_IER                     ;Unmask keyboard interrupts
                CLC
                RTS
;
$020            LDA          #XNORESRC                 ;Couldn't get keyboard resource
                JSR          SYSERR
                .PAGE
```

```
;----------------------------------------------------------------------
;
;  Console Close Request
;
;----------------------------------------------------------------------
;
CNSLCLOS        .EQU         *
                ASL          OPENFLG                   ;Console open?
                BCS          $010                      ;  Yes
                JMP          CNOTOPEN
;
$010            BIT          DNLDFLG                   ;Wait for pending download
                BMI          $010
                LDA          #KYBDDSBL
                STA          E_IER                     ;Mask keyboard interrupts
                BIT          KYBDSTRB                  ;Clear the keyboard strobe
                LDA          #KYBDSSIZ
                LDX          KYBDSADR
                LDY          KYBDSADR+1
                JSR          DEALCSIR                  ;Deallocate the keyboard interrupt
                RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Console Initialization Request
;
;----------------------------------------------------------------------
;
CNSLINIT        .EQU         *
                LDA          #FALSE
                STA          OPENFLG
                LDA          B_REG                     ;Set bank register for
                STA          KYBDBANK                  ;  keyboard and download
                STA          DNLDBANK                  ;  interrupt handlers
                LDA          #TEXTCSA%100              ;Set up character download call
                STA          SCLIST
                LDA          #TEXTCSA/100
                STA          SCLIST+1
                LDA          #00
                STA          1400+SCLIST+1
                JSR          LOADSET
                CLC
                RTS

```

```
; ####################################################################################################
; #   END OF FILE:  CONS.MISC.TEXT
; #   LINES      :  83
; #   CHARACTERS :  4017
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.UTL1.TEXT
; ####################################################################################################

                .PAGE
;----------------------------------------------------------------------
;
;   Subroutine VERIFY
;
;  This subroutine checks the screen's hardware mode, software mode,
;  and viewport parameters for self consistency.  It also sets the
;  screen switches and the following internal variables:
;       HMODE, SMINV, SMCURSOR, SMSCROLL, SMAUTOCR, SMAUTOLF,
;       SMAUTOADV, VPHMAX, VPVMAX, and TCOLOR
;
;    Parameters:  none
;
;    Exit:
;       A, X, Y:  Undefined
;
;----------------------------------------------------------------------
;
VERIFY          .EQU         *
                LDA          HMODE                      ;Validate HMODE
                AND          #03                        ;  and set 80 column
                ASL          A                          ;  flag in bit 7
                CMP          #04
                BCC          $010
                LDA          #04
$010            ROR          A
                STA          HMODE
                LDY          SMODE                      ;Preserve SMODE
                LDA          #00
                LDX          #5
$020            STA          SMFLAGS,X
                LSR          SMODE                      ;Set SMODE flags
                ROR          SMFLAGS,X
                DEX
                BPL          $020
                STY          SMODE
                LDA          #79.
                BIT          HMODE                      ;Screen width := If 80 column,
                BMI          $100                       ;  then 79.
                LDA     #39.                    ;  else 39.
$100            CMP     VPR                     ;VPR <= Screen width
                BCS     $110
                STA     VPR
$110            LDA     VPR
                CMP     VPL                     ;VPL <= VPR
                BCS     $120
                STA     VPL
$120            SEC
                LDA     VPR                     ;VPHMAX :=
                SBC     VPL                     ;  VPR - VPL
                STA     VPHMAX
                CMP     TPX                     ;TPX <= VPHMAX
                BCS     $200
                STA     TPX
$200            LDA     #23.
                CMP     VPB                     ;VPB <= Screen height
                BCS     $210
                STA     VPB
$210            LDA     VPB
                CMP     VPT                     ;VPT <= VPB
                BCS     $220
                STA     VPT
$220            SEC
                LDA     VPB                     ;VPVMAX :=
                SBC     VPT                     ;  VPB - VPT
                STA     VPVMAX
                CMP     TPY                     ;TPY <= VPVMAX
                BCS     $300
                STA     TPY
$300            LDA     TCB
                AND     #0F                     ;TCB=TCB MOD 16
                STA     TCB
                LDA     TCF
                AND     #0F                     ;TCF=TCF MOD 16
                STA     TCF
                ASL     A
                ASL     A
                ASL     A                       ;SET TCOLOR :=
                ASL     A
                ORA     TCB                     ;TCF * 16 + TCB
                STA     TCOLOR
                PHP
                SEI
                LDA     SCRNMODE                ;Check screen mode
                ASL          A
                BMI          $500                      ;  Graphics
                LDA          E_REG
                ORA          #BITON5
                BCS          $400
                AND          #BITOFF5
$400            STA          E_REG
                LDA          HMODE
                AND          #03
                BCC          $410
                ORA          #BITON7
$410            STA          SCRNMODE                  ;Set screen mode
                LSR          A
                AND          #01
                TAY
                LDA          #00
                ROL          A
                TAX
                LDA          VMODE0,X                  ;B&W / Color
                LDA          VMODE1,Y                  ;40 / 80 Column
                BIT          VMODE2                    ;Page 1 always
                BIT          VMODE3                    ;Text of course
$500            PLP
                JSR          TBASCAL                   ;New base addr.
                RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine CURSOR
;
;  This subroutine displays or removes the cursor by inverting the
;  character at the current cursor position.
;
;    Parameters:  none
;
;    Exit:
;       A, X, Y:  Undefined
;
;----------------------------------------------------------------------
;
CURSOR          .EQU         *
                BIT          SMCURSOR                  ;is cursor enabled?
                BPL          $020                      ;if not, exit
                LDA          TPX
                BIT          HMODE
                BPL          $010                      ;40 col: X=TPX
                LSR          A                         ;80 col: X=TPX/2
                BCC          $010
                TAY
                LDA          (BASE2),Y                 ;get character
                EOR          #80                       ;and invert it
                STA          (BASE2),Y                 ;put it back
                RTS
$010            TAY
                LDA          (BASE1),Y                 ;get character
                EOR          #80                       ;and invert it
                STA          (BASE1),Y                 ;put it back
$020            RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Single Character Screen Read   (Console character copy)
;
;  This subroutine returns the character at the current cursor position.
;
;    Parameters:  none
;
;    Exit:
;       A:  character
;       X, Y:  Undefined
;
;
;----------------------------------------------------------------------
;
SCRNPICK        .EQU         *
                LDA          TPX
                TAY
                BIT          HMODE
                BPL          $010                      ;40 Col -- Y := TPX
                LSR          A                         ;80 Col -- Y := TPX/2
                TAY
                BCC          $010
                LDA          (BASE2),Y                 ;Read odd text page
                BCS          $020
$010            LDA          (BASE1),Y                 ;Read even text page
$020            RTS
                .PAGE
;----------------------------------------------------------------------
;
;  Subroutine TBASCAL -- Text Base Address Calculator
```

```
;
;  This subroutine sets the base address registers, BASE1 and BASE2,
;  to point to the current line in screen memory.  BASE1 always points
;  to column 0 of the current viewport while BASE2 points to column 1.
;
;  Entry TBASCAL:
;    Parameters:  none
;
;  Entry TBASCAL1:
;    Parameters:
;       X:  Absolute screen line number
;
;  Exit (either entry point):
;       A:  Undefined
;       X:  Absolute screen line number
;       Y:  Unchanged
;
;----------------------------------------------------------------------
;
TBASCAL         .EQU         *
                CLC
                LDA          TPY                        ;vertical position
                ADC          VPT                        ; + viewport top
                TAX
TBASCAL1        .EQU         *
                CLC
                LDA          VPL                        ;viewport left:
                BIT          HMODE
                BPL          $010                       ;if 80 column mode,
                LSR          A                          ;then divide by two
$010            PHP
                ADC          BASL,X                     ;base address (LO)
                STA          BASE1                      ;same for both pages
                STA          BASE2
                LDA          BASH,X                     ;base address (HI)
                PLP
                BCC          $020
                DEC          BASE1                      ;Odd window adjustment
                EOR          #0C
$020            STA          BASE1+1                    ;even page address
                EOR          #0C
                STA          BASE2+1                    ;odd page address
                RTS
                .PAGE
;----------------------------------------------------------------------
```

```
;
;  Subroutine CLREOL -- Clear to End of Line
;
;  This subroutine clears the current line from the current cursor
;  position to the end of the line.  The starting position may be
;  passed in Y using the CLREOL1 entry point.  The text base address
;  pointers, BASE1 and BASE2, must point to the current line.
;
;  Entry CLREOL:
;    Parameters:  none
;
;  Entry CLREOL1:
;    Parameters:
;       Y:  Starting horizontal position
;
;  Zero Page Temporary Storage:
;       BLANK, TEMPX
;
;  Exit (either entry point):
;       A, Y:  Undefined
;       X:  Preserved
;
;----------------------------------------------------------------------
;
CLREOL          .EQU         *
                LDY          TPX                       ;horizontal position
CLREOL1         .EQU         *
                LDA          #80+ASC_SP                ;Set up a blank
                EOR          SMINV
                STA          BLANK
                BIT          HMODE
                BPL          $200
                TYA
                BNE          $150
;
;  80 column clear full line
;
                LDA          VPHMAX                    ;Start at right edge
                LSR          A
                TAY
                LDA          BLANK                     ;Load the blank
                BCC          $110
$100            STA          (BASE2),Y                 ;Clear odd column
$110            STA          (BASE1),Y                 ;  then even column
                DEY
                BPL          $100                      ;Repeat to BOL
                RTS
;
;  80 column clear to end of line
;
$150            STX          TEMPX                     ;Save X
                CLC
                SBC          VPHMAX                    ;Calculate negative number
                TAX                                    ;  of bytes to blank
                TYA
                LSR          A
                TAY
                LDA          BLANK                     ;Load the blank
                BCS          $170
$160            STA          (BASE1),Y
                INX
                BPL          $180
$170            STA          (BASE2),Y
                INY
                INX
                BMI          $160
$180            LDX          TEMPX                     ;Restore X
                RTS
;
;  40 column clear to end of line
;
$200            LDA          BLANK
                STA          (BASE1),Y
                LDA          TCOLOR
                STA          (BASE2),Y
                CPY          VPHMAX
                INY
                BCC          $200
                RTS

```

```
; ####################################################################################################
; #   END OF FILE:  CONS.UTL1.TEXT
; #   LINES      :  299
; #   CHARACTERS :  13475
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
; ####################################################################################################
; #   PROJECT  :  Apple /// SOS Console Driver 1.31 (6502 Assembly Source Code)
; #   FILE NAME:  CONS.UTL2.TEXT
; ####################################################################################################

                  .PAGE
;----------------------------------------------------------------------
;
;  Subroutine CLREOS -- Clear to End of Screen
;
;  This subroutine clears the screen from the current cursor position
;  to the end of the viewport.  The CLREOS1 entry allows the line number
;  to be passed in X and the starting column number in Y.
;
;  Entry CLREOS:
;    Parameters:  none
;
;  Entry CLREOS1:
;    Parameters:
;       X:  Starting absolute line number
;       Y:  Starting column number
;
;  Exit:
;       A, X, Y:  Undefined
;
;----------------------------------------------------------------------
;
CLREOS          .EQU         *
                CLC
                LDA          TPY
                ADC          VPT
                TAX                                   ;Get starting line number
                LDY          TPX                      ;Get starting cursor position
CLREOS1         .EQU         *
$010            JSR          TBASCAL1
                JSR          CLREOL1                  ;Clear this line
                LDY          #0                       ;Reset starting column
                CPX          VPB
                INX
                BCC          $010
                JMP          TBASCAL                  ;Reset base address
                .PAGE
;-----------------------------------------------------------------------
;
```

```
;  Scroll Text Viewport
;
;  This subroutine scrolls the text within the viewport up or down by
;  one line.  On entry, A must contain an UP/DOWN flag ( $00 => UP,
;  $80 => DOWN ).
;
;  Parameters:
;       A:  Up / Down flag
;
;  Zero Page Temporary Storage:
;       WORK1, WOR2:  Screen pointers
;       FLAGS:  Bit 7 -- even / odd flag for scroll loop
;               Bit 6 -- up / down flag
;       TEMP1:  Starting Y index for scroll loop
;
;  Subroutines called:
;       TBASCAL1, CLREOL1
;
;  Exit:
;       A, X, Y:  Undefined
;
;-----------------------------------------------------------------------
;
SCROLL          .EQU         *
                STA          FLAGS                     ;Save UP/DOWN flag
                SEC
                LDA          VPHMAX
                BIT          HMODE
                BPL          $010
                LSR          A
$010            STA          TEMP1                     ;Get starting loop index
                ROR          FLAGS                     ;Save even/odd flag
                LDX          VPT
                BIT          FLAGS
                BVC          $020
                LDX          VPB
$020            JSR          TBASCAL1                  ;Get starting base addresses
;
$030            BIT          FLAGS
                BVC          $040
                CPX          VPT                       ;Scroll down
                BEQ          $080                      ;  All done
                DEX                                    ;  Go up one line
                BPL          $050
$040            CPX          VPB                       ;Scroll up
```

```
                BEQ          $080                      ;  All done
                INX                                    ;  Go down one line
;
$050            LDA          BASE1
                LDY          BASE1+1                   ;Copy source address
                STA          WORK1                     ;  to destination address
                STY          WORK1+1
                LDA          BASE2
                LDY          BASE2+1
                STA          WORK2
                STY          WORK2+1
                JSR          TBASCAL1                  ;Get next source address
                LDY          TEMP1
                BIT          FLAGS
                BPL          $070
$060            LDA          (BASE2),Y                 ;Scroll this line
                STA          (WORK2),Y                 ;  move odd column
$070            LDA          (BASE1),Y                 ;  move even column
                STA          (WORK1),Y
                DEY
                BPL          $060
                BMI          $030
;
$080            LDY          #0
                JSR          CLREOL1                   ;Clear last line
                JMP          TBASCAL
                .PAGE
;-----------------------------------------------------------------------
;
;   Horizontal Shift
;
;  This subroutine shifts the text within the viewport left or right.
;  On entry, A must contain an eight bit signed shift offset, negative
;  for left shifts and positive for right shifts.
;
;  Parameters:
;       A:  Signed shift offset
;
;  Zero Page Temporary Storage:
;       BLANK, TEMPX
;       WORK1, WORK2:  Screen pointers
;       FLAGS:  Bit 7 -- right / left flag
;               Bit 6 -- odd / even flag for shift right
;       TEMP1:  Positive shift offset
;       TEMP2:  Absolute shift column
;       TEMP3:  shift right -- starting shift index
;               shift left -- shift count
;       TEMP4:  shift right -- starting clear index
;               shift left -- column for clear
;
;  Subroutines Called:
;       CLREOS1, CLREOL1
;
;  Exit:
;       A, X, Y:  Undefined
;
;-----------------------------------------------------------------------
;
SHIFT           .EQU         *
                TAY
                BEQ          $020                      ;Nothing to do
                AND          #BITON7
                STA          FLAGS                     ;Set right / left flag
                TYA
                CMP          #80
                BCC          $010
                EOR          #0FF                      ;Make shift count positive
$010            ADC          #00
                STA          TEMP1                     ;Absolute shift offset
                ADC          VPL                       ;Absolute column number
                STA          TEMP2                     ;  for base address
                LDX          VPT
                SEC
                LDA          VPHMAX
                SBC          TEMP1
                BCS          $030
                LDY          #00                       ;Shift entire viewport
                JSR          CLREOS1
$020            RTS
$030            BIT          FLAGS
                BMI          $060
;
                SEC                                    ;Set up for shift right
                BIT          HMODE
                BPL          $040
                LSR          A
$040            STA          TEMP3                     ;Set starting index for shifting
                LDA          #BITON6
                BCS          $050
                LDA          #00
```

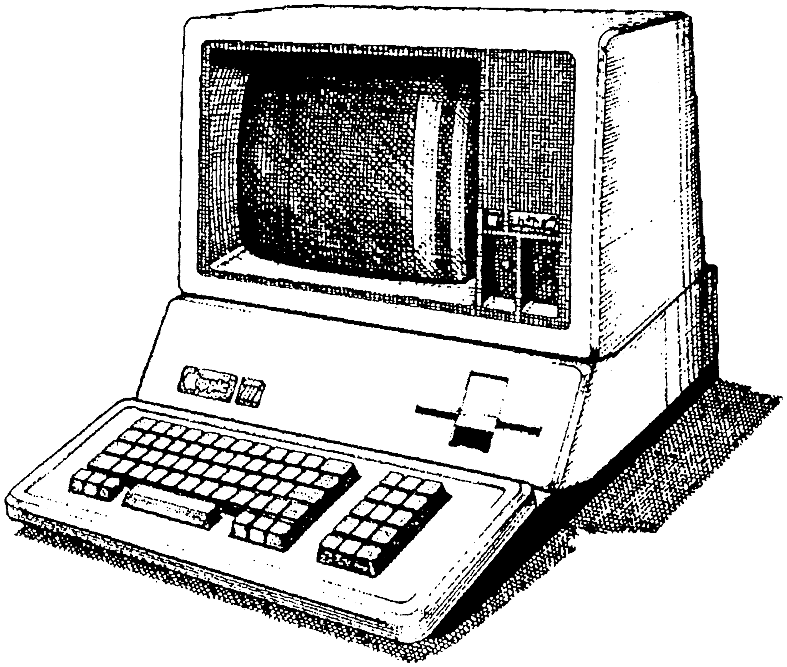
```
$050            ORA          FLAGS                    ;Set odd / even flag
                STA          FLAGS
                LDY          TEMP1
                DEY
                STY          TEMP4                    ;Set index for clearing
                LDA          #80+ASC_SP
                EOR          SMINV
                STA          BLANK                    ;Set up space character
                JMP          SHIFT1
;
$060            TAY                                   ;Set up for shift left
                BIT          HMODE
                BMI          $070
                SEC
                ROL          A
$070            STA          TEMP3                    ;Set count for shifting
                INY
                STY          TEMP4                    ;Set index for clearing
;
SHIFT1          JSR          TBASCAL1                 ;Get base address
                CLC
                LDA          TEMP2
                BIT          HMODE
                BPL          $010
                LSR          A
$010            PHP
                ADC          BASL,X
                STA          WORK1                    ;Get shifted base address
                STA          WORK2
                LDA          BASH,X
                PLP
                BCC          $020
                DEC          WORK1
                EOR          #0C
$020            STA          WORK1+1
                EOR          #0C
                STA          WORK2+1
                BIT          FLAGS
                BMI          SHFTLF
;
SHFTRT          LDY          TEMP3                    ;Shift this line right
                BVC          $020
$010            LDA          (BASE2),Y
                STA          (WORK2),Y
$020            LDA          (BASE1),Y
```

```
                STA          (WORK1),Y
                DEY
                BPL          $010
                LDA          TEMP4                     ;Clear beginning of line
                BIT          HMODE
                BPL          $050
                LSR          A
                TAY
                LDA          BLANK
                BCC          $040
$030            STA          (BASE2),Y
$040            STA          (BASE1),Y
                DEY
                BPL          $030
                BMI          SHIFT2
$050            TAY
$060            LDA          BLANK
                STA          (BASE1),Y
                LDA          TCOLOR
                STA          (BASE2),Y
                DEY
                BPL          $060
;
SHIFT2          CPX          VPB
                INX                                    ;Go to next line
                BCC          SHIFT1
                JMP          TBASCAL
;
SHFTLF          LDY          #00                       ;Shift this line left
                STX          TEMPX
                LDX          TEMP3                     ;Get shift count
$010            LDA          (WORK1),Y
                STA          (BASE1),Y
                DEX
                BMI          $020
                LDA          (WORK2),Y
                STA          (BASE2),Y
                INY
                DEX
                BPL          $010
$020            LDX          TEMPX
                LDY          TEMP4
                JSR          CLREOL1
                JMP          SHIFT2
                .PAGE
```

```
;-----------------------------------------------------------------------
;
;  Dump and Restore Contents of Viewport
;
;  This subroutine will dump or restore the contents of the viewport to
;  or from the caller's buffer.  On entry,  A must contain a dump/restore
;  flag.  ($00 => Dump  $80 => Restore)
;
;  Parameters:
;       A:  Dump / Restore flag
;
;  Zero Page Temporary Storage:
;       WORK1, WORK2:  Extended pointers to caller's buffer
;       FLAGS:  Bit 7 -- odd / even move count flag
;               Bit 6 -- dump / restore flag
;       TEMP1:  Starting move index
;       TEMP2:  Move count
;
;  Exit:
;       A, X, Y:  Undefined
;
;-----------------------------------------------------------------------
;
SCRNDUMP        .EQU         *
                STA          FLAGS
                JSR          CURSOR                    ;Turn cursor off
                LDA          VPHMAX
                STA          TEMP2
                INC          TEMP2
                BIT          HMODE
                BMI          $010
                ASL          TEMP2
                ASL          A
$010            LSR          A
                STA          TEMP1
                ROR          FLAGS
                CLC
                LDA          SCLIST
                ADC          #03                       ;Set work pointers to
                STA          WORK1                     ;  to caller's buffer
                LDA          SCLIST+1
                ADC          #00
                CMP          #0F0
                LDX          1401+SCLIST
                BCC          $020
                SBC          #80                       ;Adjust extended address
                INX
$020            STA          WORK1+1
                STX          1401+WORK1
                LDA          TEMP2
                LSR          A
                ADC          WORK1
                STA          WORK2
                LDA          WORK1+1
                ADC          #00
                STA          WORK2+1
                STX          1401+WORK2
;
;  Copy the contents of the window
;
                LDX          VPT
$100            JSR          TBASCAL1
                LDY          TEMP1
                BIT          FLAGS
                BVS          $120
;
                BPL          $115
$110            LDA          (BASE2),Y
                STA          (WORK2),Y
$115            LDA          (BASE1),Y
                STA          (WORK1),Y
                DEY
                BPL          $110
                BMI          $140
;
$120            BPL          $135
$130            LDA          (WORK2),Y
                STA          (BASE2),Y
$135            LDA          (WORK1),Y
                STA          (BASE1),Y
                DEY
                BPL          $130
;
$140            CLC
                LDA          WORK1
                ADC          TEMP2
                STA          WORK1
                BCC          $150
                INC          WORK1+1
$150            CLC
```

```
                LDA          WORK2
                ADC          TEMP2
                STA          WORK2
                BCC          $160
                INC          WORK2+1
$160            CPX          VPB
                INX
                BCC          $100
;
                JSR          TBASCAL
                JSR          CURSOR                    ;Restore cursor
                RTS
                .PAGE
;----------------------------------------------------------------------
;
;  ZPOUT
;
;  This subroutine saves the driver's zero page data.
;
;----------------------------------------------------------------------
;
ZPOUT           LDX          #ZPLENGTH-1               ;Zero Page save area length
$010            LDA          ZPDATA,X
                STA          ZPSAVE,X
                DEX
                BPL          $010
                RTS


; ####################################################################################################
; #   END OF FILE:  CONS.UTL2.TEXT
; #   LINES      :  382
; #   CHARACTERS :  16487
; #   Formatter  :  Assembly Language Reformatter 1.0.2 (07 January 1998)
; #   Author     :  David T. Craig -- 71533.606@compuserve.com -- Santa Fe, New Mexico USA
; ####################################################################################################
```



# The End