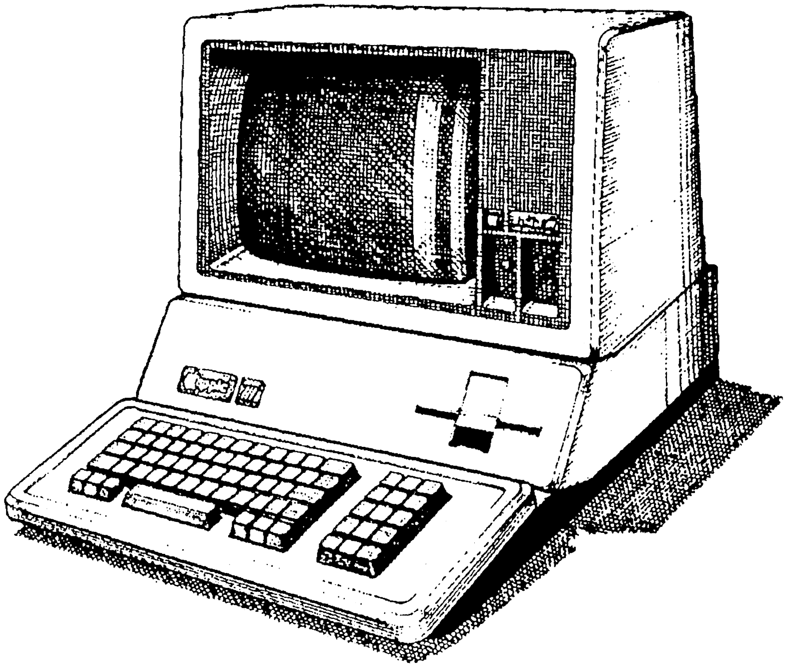
Apple /// Computer Technical Information

Apple ///
RS-232 SERIAL DRIVER
Version 1.30
Source Code Listing



FORMATTED LISTING

```
; ######################################################################################################
; #   PROJECT  :  Apple /// SOS RS-232 Driver 1.30 (6502 Assembly Source Code)
; #   FILE NAME:  RS232.text
; ######################################################################################################

                .TITLE       "SOS RS232 DRIVER"

;
;               SOS RS232 DRIVER
;
;               (C)  APPLE COMPUTER 1981, 1982, 1983
;
;               Jim Trezzo     1/07/83
;
;               V 1.01  9/11/81 Clear NO_OUTPUT flag during reset
;               V 1.02  4/23/82 Don't drop DTR during reset and wait for
;                       DLYCNT before close.
;               V 1.30  1/07/83 Add comment field
;

DEVTYPE         .EQU         63                        ;I/O CHAR DEV
SUBTYPE         .EQU         01                        ;DEV SUBTYPE
MANID           .EQU         01                        ;MANUFACTURER ID-APPLE
RELEASE         .EQU         1300                      ;RELEASE NUMBER-BCD FORMAT

;
;  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          $010                      ;  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
$010            .ENDM

;
;       INCREMENT WORD MACRO
;

                .MACRO       INW
                INC          %1
                BNE          $210
                INC          %1+1
$210            .ENDM

;
;       INCREMENT ADDRESS MACRO
;
;               INCREMENTS 3 BYTE ADDRESS
;

                .MACRO       INCADR
                INC          %1
                BNE          $310
                INC          %1+1
                BNE          $310                      ;Bank overflow ?
                SEC                                    ;Yes
                ROR          %1+1
                INC          %1+1+1400                 ;Increment X byte
$310            .ENDM

                .MACRO       SET_1MHZ

                LDA          E_REG
                ORA          #BITON7                   ;Set 1 MHZ mode
                STA          E_REG
                .ENDM

                .MACRO       SET_2MHZ

                LDA          E_REG
                AND          #07F                      ;Set 2 MHZ mode
                STA          E_REG
                .ENDM

                .PROC        RS232

;
;       SOS GLOBAL DATA AND SUBROUTINES
;
ALLOCSIR        .EQU         1913                      ;SOS interrupt allocation manager
DEALCSIR        .EQU         1916                      ;SOS interrupt deallocation manager
SYSERR          .EQU         1928                      ;SOS error return
;
;       SOS Error Codes
;
XREQCODE        .EQU         20                        ;Invalid request code
XCTLCODE        .EQU         21                        ;Invalid control/status code
XCTLPARM        .EQU         22                        ;Invalid control/status param
XNOTOPEN        .EQU         23                        ;Device not open
XNOTAVIL        .EQU         24                        ;Device not available
XNORESRC        .EQU         25                        ;Resource not available
XBADOP          .EQU         26                        ;Invalid operation for device
;
;       HARDWARE I/O ADDRESSES
;
ACIADATA        .EQU         0C0F0                     ;ACIA DATA REGISTER
```

```
ACIASTAT        .EQU         0C0F1                     ;ACIA STATUS REGISTER
ACIACMD         .EQU         0C0F2                     ;ACIA COMMAND REGISTER
ACIACTL         .EQU         0C0F3                     ;ACIA CONTROL REGISTER
E_REG           .EQU         0FFDF                     ;ENVIRONMENT REGISTER
B_REG           .EQU         0FFEF                     ;BANK REGISTER
;
;       GENERAL EQUATES
;
TRUE            .EQU         80
FALSE           .EQU         00
BITON0          .EQU         01
BITON1          .EQU         02
BITON2          .EQU         04
BITON3          .EQU         08
BITON4          .EQU         10
BITON6          .EQU         40
BITON7          .EQU         80
ASC_LF          .EQU         0A
ASC_FF          .EQU         0C
ASC_CR          .EQU         0D
;
                .PAGE

                .WORD        0FFFF
                .WORD        73.
                .ASCII       "(C) Apple Computer 1981, 1982, 1983.  "
                .ASCII       "Built-in Serial Port RS-232 Driver."

;---------------
;
;       DEVICE INFORMATION BLOCK
;
;---------------
;       DEVICE HEADER BLOCK
;---------------

IDBLK           .WORD        0000                      ;LINK TO NEXT DEVICE HANDLER
                .WORD        RS_MAIN                   ;ENTRY POINT ADDRESS
                .BYTE        6                         ;LENGTH OF DRIVER NAME
                .ASCII       ".RS232         "         ;DRIVER NAME
                .BYTE        80,00,00                  ;DEV NUM, DEV SLOT, DEV UNIT
                .BYTE        DEVTYPE                   ;DEVICE TYPE
                .BYTE        SUBTYPE                   ;DEV SUBTYPE
                .BYTE        00                        ;FUTURE USE
                .WORD        0000                      ;BLOCK COUNT-NOT USED
                .WORD        MANID                     ;MANUFACTURER ID
                .WORD        RELEASE                   ;RELEASE NUMBER-BCD

;---------------
;       DEVICE CONFIGURATION BLOCK
;---------------

CNFGBLK         .WORD        12.                       ;CONFIGURATION BLOCK LENGTH

DCB             .BYTE        06                        ;BAUD RATE - 300
                .BYTE        22                        ;Data format
;CTL - Hi nybble
;CMD - Lo nybble
                .BYTE        00                        ;Carriage return delay
                .BYTE        00                        ;Line feed delay
                .BYTE        00                        ;Form feed delay
                .BYTE        00                        ;00 - no protocol
;80 - XON/XOFF
;40 - ENQ/ACK
                .BYTE        13                        ;Character to use as XOFF (or ENQ)
                .BYTE        11                        ;Character to use as XON (or ACK)
                .BYTE        223.                      ;Buffer level which triggers XOFF
                .BYTE        132.                      ;Buffer level which triggers XON
                .BYTE        80.                       ;Character count for ENQ/ACK
                .BYTE        00                        ;Hardware handshake support

DCB_LN          .EQU         *-DCB

                .ASCII       "(C) Apple Computer Inc. 1983"

                .PAGE
;
;   SOS Device Handler Interface
;

SOSINT          .EQU         0C0
REQCODE         .EQU         SOSINT+0                  ;SOS request code
BUFFER          .EQU         SOSINT+2                  ;Buffer pointer
REQCNT          .EQU         SOSINT+4                  ;Requested count/Byte count
CTLSTAT         .EQU         SOSINT+2                  ;Control/status code
CSLIST          .EQU         SOSINT+3                  ;Control/status list pointer
RETPTR          .EQU         SOSINT+8                  ;Returned count pointer


;
```

```
;  Zero Page Storage
;

ZPGSAVE         .EQU         SOSINT+0A                 ;Saved zero page storage

ZPGTEMP         .EQU         ZPGSAVE+00                ;Temporary zero page storage

OPRODPTR        .EQU         0E1                       ;Producer pointer
ICSMRPTR        .EQU         0E2                       ;Consumer pointer
RETCNT          .EQU         0E3                       ;Returned byte count word


;
;   Private Variable Storage
;

SIRADDR         .WORD        SIRTABLE
SIRTABLE        .BYTE        1,0                       ;ACIA resource
                .WORD        ACIAMIH
MIHBANK         .BYTE        0
SIRCOUNT        .EQU         *-SIRTABLE

OPENFLG         .BYTE        FALSE                     ;Device open flag
IS_NEWLINE      .BYTE        FALSE                     ;Bit 7 (1=new line mode)
NEWLINE         .BYTE        00                        ;Newline character
IN_PROG         .BYTE        00                        ;Bit 7 (1=XOFF in progress)
;Bit 6 (1=XOFF needs to be sent)
SEND_XON        .BYTE        00                        ;Bit 7 (1=XON needs to be sent)
RTS_FALSE       .BYTE        00                        ;Bit 7 (1=RTS false)
NO_OUTPUT       .BYTE        00                        ;Bit 7 (1=suspend output)
DLYCNT          .BYTE        00                        ;Delay count for MIH
BRK_CNT         .BYTE        00                        ;Interval count for Break signal
CHAR_OUT        .BYTE        00                        ;Output character count for ENQ/ACK
IN_PROG1        .BYTE        00                        ;Bit 7 (1=ENQ in progress)

;
;       Device control parameters
;

CNTL_PARAM      .BYTE        15.                       ;List length

BAUD            .BYTE        00                        ;BAUD RATE
DFORMAT         .BYTE        00                        ;Data format
;CTL - Hi nybble
;CMD - Lo nybble
```

```
CRDELAY         .BYTE        00                        ;Carriage return delay
LFDELAY         .BYTE        00                        ;Line feed delay
FFDELAY         .BYTE        00                        ;Form feed delay
PROTOCOL        .BYTE        00                        ;00 - none
;80 - XON/XOFF
;40 - ENQ/ACK
CTLCHR1         .BYTE        00                        ;Character to use as XOFF (or ENQ)
CTLCHR2         .BYTE        00                        ;Character to use as XON (or ACK)
MAXBUF          .BYTE        00                        ;Buffer level which triggers XOFF
;       (or RTS false)
MINBUF          .BYTE        00                        ;Buffer level which triggers XON
;       (or RTS true)
CHARCNT         .BYTE        00                        ;Character count for ENQ/ACK
HDW_HSHAKE      .BYTE        00                        ;Hardware handshake
;       Bit 7 (1=enabled)
RD_IMMEDIATE    .BYTE        00                        ;Bit 7 (1=read immediate mode)
STAT_REG        .BYTE        00                        ;Status reg - saved from last interrupt
STAT_LATCH      .BYTE        00                        ;Latched status bits - cleared by reset
;       or status request-1
;Bit 0 (1=parity error)
;Bit 1 (1=framing error)
;Bit 2 (1=overrun)
;Bit 5 (1=DCD went false)
;Bit 6 (1=DSR went false)
;Bit 7 (1=input character lost)

CNTL_LN         .EQU         *-CNTL_PARAM

;
;       Data Buffers
;

OBUFCNT         .BYTE        0                         ;Local output buffer byte count
OSPRODPTR       .BYTE        0                         ;Producer buffer pointer
OCSMRPTR        .BYTE        0                         ;Consumer buffer pointer
OLOCBUF         .BLOCK       0100,0                    ;Local output buffer

IBUFCNT         .BYTE        0                         ;Local input buffer byte count
ISCSMRPTR       .BYTE        0                         ;Input consumer pointer
IPRODPTR        .BYTE        0                         ;Input producer pointer
ILOCBUF         .BLOCK       0100,0                    ;Local input buffer

                .PAGE
;--------------------------------------------------------------
;
```

```
;       RS232 DRIVER - MAIN ENTRY POINT
;
;--------------------------------------------------------------

RS_MAIN         .EQU         *
                SWITCH       REQCODE,8,RS_REQSW


BADREQ          LDA          #XREQCODE                 ;Invalid request code
                JSR          SYSERR


NOTOPEN         LDA          #XNOTOPEN                 ;Device not open
                JSR          SYSERR


RS_REQSW        .EQU         *                         ;RS232 driver request switch
                .WORD        RS_READ-1
                .WORD        RS_WRITE-1
                .WORD        RS_STAT-1
                .WORD        RS_CNTL-1
                .WORD        BADREQ-1
                .WORD        BADREQ-1
                .WORD        RS_OPEN-1
                .WORD        RS_CLOSE-1
                .WORD        RS_INIT-1

                .PAGE
;--------------------------------------------------------------
;
;       RS232 Driver -- Initialization Request
;
;--------------------------------------------------------------

RS_INIT         .EQU         *

                LDA          #FALSE
                STA          OPENFLG                   ;Set serial port to not open
                CLC                                    ;Insure carry clear for load program
                RTS

                .PAGE
;----------------------------------------------------------------------
;
;       RS232 Driver -- Open Request
```

```
;
;----------------------------------------------------------------------

RS_OPEN         .EQU         *
                BIT          OPENFLG                   ;Serial Port open?
                BPL          $010                      ;  No
                LDA          #XNOTAVIL
                JSR          SYSERR

$010            LDA          B_REG
                AND          #0F
                STA          MIHBANK                   ;Set interrupt handler bank
                LDA          #SIRCOUNT
                LDX          SIRADDR
                LDY          SIRADDR+1
                JSR          ALLOCSIR                  ;Allocate the ACIA
                BCS          $020

                LDX          #DCB_LN-1                 ;Copy Device Configuration Block
;  into device control parameters

$015            LDA          DCB,X
                STA          CNTL_PARAM+1,X
                DEX
                BPL          $015

                LDA          #0
                STA          IS_NEWLINE                ;Set newline mode to False
                STA          NEWLINE                   ;Clear newline character
                STA          RD_IMMEDIATE              ;Read immediate mode off
                STA          IN_PROG                   ;XOFF in progress flag off
                STA          SEND_XON                  ;Send XON flag off

                JSR          CNTL00                    ;Set up ACIA
                LDA          #TRUE                     ;  and clear STAT_REG, STAT_LATCH,
;  RTS_FALSE, NO_OUTPUT, CHAR_OUT,
;  IN_PROG1, DLYCNT and BRK_CNT
                STA          OPENFLG                   ;Set serial port open
                RTS

$020            LDA          #XNORESRC
                JSR          SYSERR
                .PAGE
;----------------------------------------------------------------------
;
```

```
;       RS232 Driver -- Close Request
;
;----------------------------------------------------------------------

RS_CLOSE        .EQU         *
                ASL          OPENFLG                   ;Serial Port open?
                BCS          $05                       ;  Yes
                JMP          NOTOPEN

$05             LDA          OBUFCNT                   ;Wait for write completion
                ORA          DLYCNT                    ; and delay complete
                BNE          $05

                PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system
                SET_1MHZ
                LDA          ACIACMD
                AND          #0F0                      ;Disable Rcv/Xmit Interrupt
                STA          ACIACMD                   ;DTR off, RTS off
                LDA          ACIASTAT                  ;Clear any prior interrupt
                PLP                                    ;Restore interrupt status

                LDA          #SIRCOUNT
                LDX          SIRADDR
                LDY          SIRADDR+1
                JSR          DEALCSIR                  ;Deallocate the ACIA
                RTS

                .PAGE
;-----------------------------------------------------------------------
;
;       RS232 Driver -- Read Request
;
;-----------------------------------------------------------------------

RS_READ         .EQU         *
                BIT          OPENFLG                   ;Serial Port open?
                BMI          $05
                JMP          NOTOPEN

$05             LDA          ISCSMRPTR                 ;Get CSMRPTR from driver storage
                STA          ICSMRPTR                  ;Put in temporary zero page
                LDY          #00                       ;Prevent offset
                STY          RETCNT                    ;Zero return count
                STY          RETCNT+1

                LDA          #0FF                      ;One's complement count
                EOR          REQCNT
                STA          REQCNT
                LDA          #0FF
                EOR          REQCNT+1
                STA          REQCNT+1

$010            INC          REQCNT                    ;Increment count
                BNE          $015                      ;Is count zero ?
                INC          REQCNT+1
                BEQ          $099                      ;Yes, terminate

$015            LDA          IBUFCNT                   ;Is input buffer empty ?
                BNE          $020                      ;No, continue
                BIT          RD_IMMEDIATE              ;Is read immediate mode set ?
                BPL          $015                      ;No, loop until character received
                BMI          $099                      ;Yes, terminate

$020            LDY          #0
                LDX          ICSMRPTR
                LDA          ILOCBUF,X                 ;Get char from local input buffer
                STA          (BUFFER),Y                ;Send to user buffer
                PHA                                    ;Save character on stack
                INCADR       BUFFER                    ;Increment addr - user buffer pointer
                INC          ICSMRPTR
                DEC          IBUFCNT
                INW          RETCNT

                LDA          MINBUF                    ;Check if below min buffer level
                CMP          IBUFCNT                   ;       (IBUFCNT < MINBUF ?)
                BCC          $025                      ;No, continue

                BIT          IN_PROG                   ;Yes, XOFF in progress ?
                BMI          $022                      ;Yes, send XON

                BIT          RTS_FALSE                 ;Is RTS false ?
                BPL          $025                      ;No, continue

                PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system
                SET_1MHZ                               ;Yes, set 1 MHZ mode
                LDA          ACIACMD                   ;Set RTS true and
                AND          #0F2                      ;  enable xmit interrupt
                ORA          #05
                STA          ACIACMD                   ;Set to [xxxx01x1]
                LDA          #0
                STA          RTS_FALSE                 ;Clear RTS_FALSE
                SET_2MHZ                               ;Set 2 MHZ mode
                PLP                                    ;Restore interrupt status
                JMP          $025

$022            LDA          #80                       ;Send XON
                STA          SEND_XON                  ;Set flag
                JSR          PRIME_OUT                 ;Prime output routine

$025            PLA                                    ;Retrieve character from stack
                BIT          IS_NEWLINE                ;Is newline mode set ?
                BPL          $010                      ;No, get next char
                CMP          NEWLINE                   ;Yes, is char terminator ?
                BEQ          $099                      ;If yes, terminate
                JMP          $010                      ;No, get next char

$099            LDY          #0
                LDA          ICSMRPTR                  ;Terminate
                STA          ISCSMRPTR                 ;Save pointer

                LDA          RETCNT                    ;Get count of returned bytes
                STA          (RETPTR),Y                ;Send to user
                LDA          RETCNT+1
                INY
                STA          (RETPTR),Y

                RTS                                    ;Return to user

                .PAGE
;-----------------------------------------------------------------------
;
;       RS232 Driver -- Write Request
;
;-----------------------------------------------------------------------

RS_WRITE        .EQU         *
                BIT          OPENFLG                   ;Serial Port open?
                BMI          $05
                JMP          NOTOPEN


$05             LDA          OSPRODPTR                 ;Get PRODPTR from driver storage
                STA          OPRODPTR                  ;Put in temporary zero page

                LDA          #0FF                      ;One's complement count
                EOR          REQCNT
                STA          REQCNT
                LDA          #0FF
                EOR          REQCNT+1
                STA          REQCNT+1

$010            INC          REQCNT                    ;Increment count
                BNE          $030                      ;Is count zero ?
                INC          REQCNT+1
                BNE          $030                      ;No

                JSR          PRIME_OUT                 ;Prime consumer
                LDA          OPRODPTR                  ;Save producer pointer in driver
                STA          OSPRODPTR
                RTS                                    ;Return to user

$030            LDX          OBUFCNT                   ;Is local output buffer full ?
                INX
                BNE          $040                      ;No

                JSR          PRIME_OUT                 ;Local buffer is full, prime consumer
                JMP          $030

$040            LDY          #00
                LDA          (BUFFER),Y                ;Get character from user buffer
                INCADR       BUFFER                    ;Increment addr - user buffer ptr

                LDX          OPRODPTR                  ;Get producer pointer
                STA          OLOCBUF,X                 ;Store character in local buffer
                INC          OPRODPTR                  ;Advance local buffer
                INC          OBUFCNT                   ;Advance count
                BNE          $010                      ;Branch always taken


                .PAGE
;----------------------------------------------------------------------
;
;       RS232 Driver -- Status Request
;
;----------------------------------------------------------------------

RS_STAT         .EQU         *
                BIT          OPENFLG                   ;Serial Port open?
                BMI          $05
                JMP          NOTOPEN
$05             SWITCH       CTLSTAT,3,STATSW


BADCTL          LDA          #XCTLCODE                 ;Invalid control code
                JSR          SYSERR


STATSW          .WORD        STAT00-1
                .WORD        STAT01-1
                .WORD        STAT02-1
                .WORD        STAT03-1

STAT00          RTS                                    ;0 -- NOP


STAT01          LDY          #0                        ;1 -- Retrieve device control
;  parameters (including RD_IMMEDIATE
;     STAT_REG and STAT_LATCH)
                LDA          (CSLIST),Y
                CMP          CNTL_PARAM                ;Check for room in status list
                BCS          $01                       ;  >= OK

                LDA          #XCTLPARM                 ;  < NG
                JSR          SYSERR

$01             LDY          #CNTL_LN-1
                PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system

$05             LDA          CNTL_PARAM,Y
                STA          (CSLIST),Y
                DEY
                BPL          $05

                INY
                STY          STAT_LATCH                ;Clear status latch bits
                PLP                                    ;Restore interrupt status
                RTS


STAT02          LDY          #0                        ;2 -- Get newline character

                LDA          IS_NEWLINE
                STA          (CSLIST),Y
                INY
                LDA          NEWLINE
                STA          (CSLIST),Y

                RTS

STAT03          LDY          #0                        ;3 -- Retrieve driver buffer info

                LDA          #0FF                      ;Output buffer size
                JSR          CNTOUT
                LDA          OBUFCNT                   ;Number of chars in output buffer
                JSR          CNTOUT
                LDA          #0FF                      ;Input buffer size
                JSR          CNTOUT
                LDA          IBUFCNT                   ;Number of chars in input buffer
                JSR          CNTOUT

                RTS

CNTOUT          STA          (CSLIST),Y
                INY
                LDA          #0                        ; high byte (0)
                STA          (CSLIST),Y
                INY

                RTS

                .PAGE
;----------------------------------------------------------------------
;
;       RS232 Driver -- Control Request
;
;----------------------------------------------------------------------

RS_CNTL         .EQU         *
                BIT          OPENFLG                   ;Serial Port open?
                BMI          $05                       ;  Ok
                JMP          NOTOPEN
$05             SWITCH       CTLSTAT,3,CNTLSW
                JMP          BADCTL


CNTLSW          .WORD        CNTL00-1
                .WORD        CNTL01-1
                .WORD        CNTL02-1
                .WORD        CNTL03-1


CNTL00          .EQU         *                         ;0 -- Reset device

                BIT          IN_PROG                   ;XOFF in progress ?
                BPL          $020                      ;No, continue

                LDA          #80                       ;Yes, send XON
                STA          SEND_XON                  ;Set flag
                JSR          PRIME_OUT                 ;Prime output routine

$015            BIT          SEND_XON                  ;Wait until XON gets out
                BMI          $015

$020            PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system
                LDA          BAUD                      ;Validate data rate
                AND          #00F
                STA          BAUD

                SET_1MHZ

                LDA          #0
                STA          IBUFCNT                   ;Zero Input Buffer count
                STA          OBUFCNT                   ;Zero Output Buffer count
                STA          DLYCNT                    ;Zero delay count
                STA          BRK_CNT                   ;Zero interval count

                STA          OSPRODPTR                 ;Zero pointers
                STA          OCSMRPTR
                STA          ISCSMRPTR
                STA          IPRODPTR

                STA          RTS_FALSE                 ;Clear RTS false flag
                STA          NO_OUTPUT                 ;Clear suspend output flag
                STA          CHAR_OUT                  ;Zero output character count
                STA          IN_PROG1                  ;ENQ in progress flag off
                STA          STAT_LATCH                ;Clear status latch bits
                LDA          ACIASTAT
                STA          STAT_REG                  ;Save status reg

                LDA          DFORMAT                   ;Validate data format
                AND          #0E0
                ORA          #BITON4                   ;Set receiver clock source to internal
                ORA          BAUD
                LDX          #03
                CPX          BAUD                      ;If data rate  is 110 baud
                BNE          $025

                ORA          #BITON7                   ;  force two stop bits

$025            STA          ACIACTL                   ;Set up ACIA control register
                LDA          DFORMAT
                ASL          A
                ASL          A
                ASL          A
                ASL          A
                AND          #0E0
                ORA          #09                       ;Xmit disabled, Rcv enabled
;DTR and RTS on
                STA          ACIACMD                   ;Set up ACIA command register

                PLP                                    ;Restore interrupt status
                RTS


CNTL01          LDY          #0                        ;1 -- Load device control parameters
;  except STAT_REG and STAT_LATCH
                LDA          (CSLIST),Y
                CMP          CNTL_PARAM                ;Check length of control list
                BEQ          $01                       ;  = OK

                LDA          #XCTLPARM                 ;   NG
                JSR          SYSERR

$01             LDY          #CNTL_LN-3

$05             LDA          (CSLIST),Y
                STA          CNTL_PARAM,Y
                DEY
                BPL          $05

                JSR          CNTL00                    ;Set up ACIA

                RTS

CNTL02          .EQU         *                         ;2 -- Set New Line Character

```

```
                LDY          #0
                LDA          (CSLIST),Y
                STA          IS_NEWLINE
                INY
                LDA          (CSLIST),Y
                STA          NEWLINE

                RTS

CNTL03          .EQU         *                         ;3 -- Transmit Break

$05             LDA          OBUFCNT                   ;Wait for write completion
                BNE          $05

                TAY
                LDA          (CSLIST),Y                ;Get number of break intervals
                BMI          $050                      ;Too large, return
                BEQ          $050                      ;Zero, return
                CMP          #101.                     ;Check if > 100 (23.3 sec)
                BCS          $050                      ;Too large, return

                STA          BRK_CNT                   ;Save interval count
                PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system
                SET_1MHZ                               ;Set 1 MHz mode
                LDA          ACIACMD                   ;Transmit Break
                ORA          #0C
                STA          ACIACMD                   ;Set to [xxxx11xx]
                LDA          #0
                STA          RTS_FALSE                 ;Clear RTS false
                PLP                                    ;Restore interrupt status

$010            LDY          #181.                     ;This double loop takes 233 ms
                LDX          #0                        ;  in 1 MHz mode
$015            DEX
                BNE          $015
                DEY
                BNE          $015

                DEC          BRK_CNT                   ;Loop for interval count
                BNE          $010

                JSR          PRIME_OUT                 ;Prime output routine

$050            RTS
```

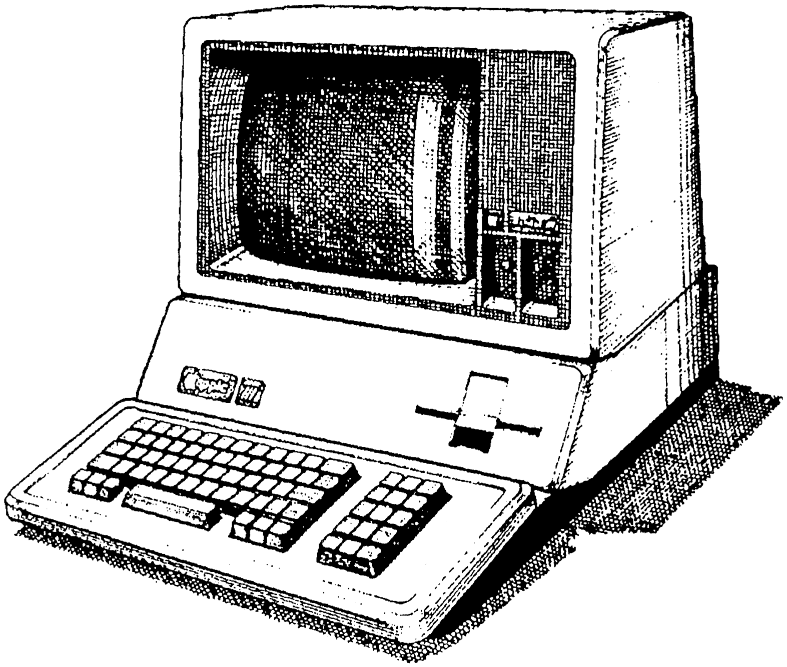
```

                .PAGE
;----------------------------------------------------------------------
;
;       ACIA MASTER INTERRUPT HANDLER
;
;----------------------------------------------------------------------

ACIAMIH         .EQU         *

                STY          STAT_REG                  ;Save current status reg

                TYA
                AND          #BITON3                   ;Check receiver data reg full
                BEQ          $010                      ;No, continue

                TYA                                    ;Input interrupt
                AND          #67
                ORA          STAT_LATCH
                STA          STAT_LATCH                ;Latch status bits
                JMP          RS_IN

$010            TYA                                    ;Treat as output interrupt
                AND          #60
                ORA          STAT_LATCH
                STA          STAT_LATCH                ;Latch status bits
                JMP          RS_OUT

RS_IN           .EQU         *                         ;Receive next character

                SET_1MHZ
                LDX          ACIADATA                  ;Read character
                SET_2MHZ
                TXA

                BIT          PROTOCOL                  ;Is XON/XOFF protocol mode set?
                BPL          $016                      ;No, continue

                CMP          CTLCHR1                   ;Yes, check for XOFF
                BNE          $010                      ;No

                LDA          #TRUE                     ;Yes, suspend output
                STA          NO_OUTPUT
                JMP          RS_OUT

```

```
$010            CMP          CTLCHR2                   ;Check for XON
                BNE          $015                      ;No

                LDA          #FALSE                    ;Yes, resume output
                STA          NO_OUTPUT
                BEQ          RS_OUT                    ;Always taken

$015            LDX          IBUFCNT                   ;Check if max buffer level exceeded
                CPX          MAXBUF                    ;       (IBUFCNT >= MAXBUF ?)
                BCC          $020                      ;No, continue

                BIT          IN_PROG                   ;Yes, check if XOFF in progress
                BMI          $020                      ;Yes, continue

                LDX          #BITON6                   ;No, set XOFF needs to be sent
                STX          IN_PROG
                BNE          $020                      ;Branch always taken

$016            BVC          $017                      ;Is ENQ/ACK protocol mode set?

                CMP          CTLCHR2                   ;Yes, check for ACK
                BNE          $020                      ;No, continue

                LDA          CHARCNT                   ;Yes, reset output char count
                STA          CHAR_OUT
                LDA          #0
                STA          IN_PROG1                  ;Clear ENQ in progress
                BEQ          RS_OUT                    ;Always taken

$017            BIT          HDW_HSHAKE                ;Is Hardware handshake enabled?
                BPL          $020                      ;No, continue

                LDX          IBUFCNT                   ;Check if max buffer level exceeded
                CPX          MAXBUF                    ;       (IBUFCNT >= MAXBUF ?)
                BCC          $020                      ;No, continue

                LDX          BRK_CNT                   ;Check for Break in progress
                BNE          $020                      ;Yes, continue (can't change RTS)

                PHA                                    ;No, save character on stack
                LDA          #BITON7
                STA          RTS_FALSE
                SET_1MHZ
                LDA          ACIACMD                   ;Set RTS to false
                AND          #0F3                      ;  Xmit interrupt will be disabled
                STA          ACIACMD                   ;    ACIA set to [xxxx00xx]
                SET_2MHZ
                PLA                                    ;Retrieve character from stack

$020            LDX          IBUFCNT                   ;Is buffer full ?
                INX
                BNE          $025                      ;No, continue

                LDA          #BITON7                   ;Yes, latch char lost bit
                ORA          STAT_LATCH
                STA          STAT_LATCH
                BMI          RS_OUT                    ;Always taken

$025            LDX          IPRODPTR                  ;Address in local buffer to store data
                STA          ILOCBUF,X                 ;Store char in local input buffer
                INC          IBUFCNT
                INC          IPRODPTR

RS_OUT          .EQU         *                         ;Output next character

                LDA          BRK_CNT                   ;Check for Break in progress
                BEQ          $001                      ;No, continue
                JMP          RETURN                    ;Yes, return

$001            BIT          HDW_HSHAKE                ;Hardware handshake mode enabled ?
                BPL          $003                      ;No, continue

                BIT          RTS_FALSE                 ;Yes, check for RTS false
                BPL          $002                      ;RTS true, continue
                JMP          RETURN                    ;RTS false, return

$002            LDA          STAT_REG                  ;Check DSR and DCD status
                AND          #60
                BEQ          $003                      ;DSR and DCD true, continue

                LDA          DLYCNT                    ;DSR or DCD false, disable xmit int
                BNE          $011                      ;  unless delay in progress
                JMP          D_XMIT

$003            LDA          #BITON4                   ;Check xmit data reg empty
                BIT          STAT_REG
                BNE          $004                      ;Reg empty, continue

                JMP          E_XMIT                    ;Reg not empty, enable xmit interrupt

```

```
$004    BIT     IN_PROG         ;XOFF need to be sent ?
        BVC     $005            ;No, continue

        LDA     #BITON7         ;Yes, set XOFF in progress
        STA     IN_PROG
        LDA     CTLCHR1         ;Send XOFF
        JMP     $020

$005    BIT     SEND_XON        ;XON need to be sent ?
        BPL     $010            ;No, continue

        LDA     #0              ;Yes, clear flags
        STA     SEND_XON
        STA     IN_PROG
        LDA     CTLCHR2         ;Send XON
        JMP     $020

$010    LDA     DLYCNT          ;Any transmit delay in progress ?
        BEQ     $015            ;No

$011    DEC     DLYCNT          ;Yes, decrement count
        JMP     E_XMIT

$015    LDX     OBUFCNT         ;Is local output buffer count zero ?
        BEQ     D_XMIT          ;Yes, disable xmit interrupt and return

        BIT     NO_OUTPUT       ;Is output suspended ?
        BMI     D_XMIT          ;Yes, disable xmit interrupt and return

        BIT     PROTOCOL        ;Is, ENQ/ACK protocol mode set?
        BVC     $018            ;No, continue

        LDA     CHAR_OUT        ;Yes, check output char count
        BNE     $016            ;Count not yet exhausted, send char

        BIT     IN_PROG1        ;Check for ENQ in progress
        BMI     D_XMIT          ;Yes, disable xmit interrupt and return

        LDA     #BITON7         ;No, set ENQ in progress
        STA     IN_PROG1
        LDA     CTLCHR1         ;Send ENQ
        JMP     $020

$016    DEC     CHAR_OUT        ;Decrement output char count
$018    LDX     OCSMRPTR        ;No, get consumer pointer
```

```
                LDA          OLOCBUF,X                 ;Get character from buffer
                DEC          OBUFCNT
                INC          OCSMRPTR

$020            TAX
                SET_1MHZ
                STX          ACIADATA                  ;Send character

                CPX          #ASC_CR                   ;Check for any delay
                BNE          $022
                LDA          CRDELAY
                JMP          $024

$022            CPX          #ASC_LF
                BNE          $023
                LDA          LFDELAY
                JMP          $024

$023            CPX          #ASC_FF
                BNE          E_XMIT
                LDA          FFDELAY

$024            STA          DLYCNT

E_XMIT          SET_1MHZ
                LDA          ACIACMD                   ;Enable transmit interrupt
                AND          #0F2
                ORA          #05                       ;Set to [xxxx01x1]
                STA          ACIACMD

                RTS                                    ;Return to user

D_XMIT          SET_1MHZ
                LDA          ACIACMD                   ;Disable transmit interrupt
                AND          #0F2
                ORA          #09                       ;Set to [xxxx10x1]
                STA          ACIACMD
RETURN          RTS                                    ;Return to user

PRIME_OUT       .EQU         *                         ;Called by Read, Write and Control
;  request routines

                PHP                                    ;Save interrupt status
                SEI                                    ;Disable interrupt system
                BIT          RTS_FALSE
                BMI          $010                       ;Return if RTS false
                JSR          E_XMIT                     ;Enable transmit interrupt
                SET_2MHZ
$010            PLP                                     ;Restore interrupt status

                RTS                                     ;Return

                .END

```

```
; ######################################################################################################
; #   END OF FILE:  RS232.text
; #   LINES      :  1036
; #   CHARACTERS :  51017
; #   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