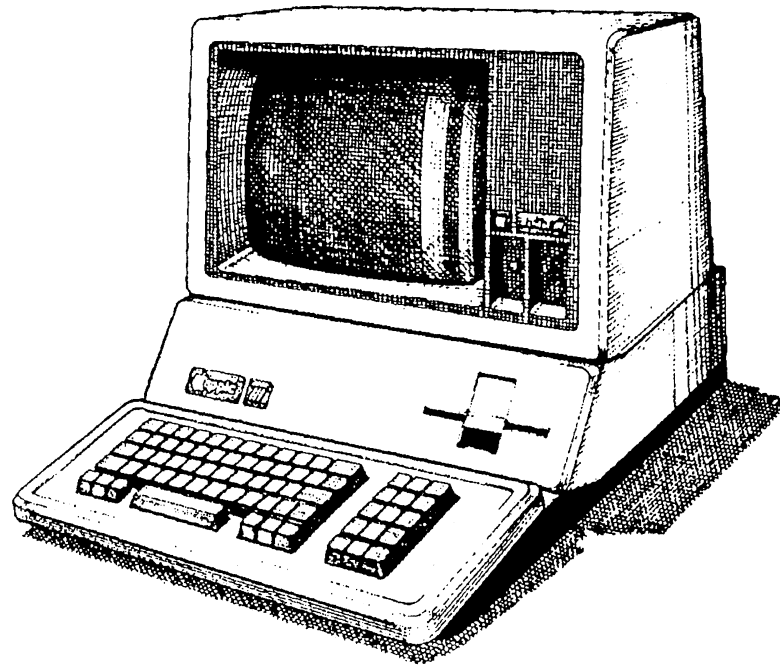




Apple /// Computer Technical Information

**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
; #####

000001          .TITLE          "SOS RS232 DRIVER"
000002
000003 ;
000004 ;           SOS RS232 DRIVER
000005 ;
000006 ;           (C)  APPLE COMPUTER 1981, 1982, 1983
000007 ;
000008 ;           Jim Trezzo      1/07/83
000009 ;
000010 ;           V 1.01  9/11/81 Clear NO_OUTPUT flag during reset
000011 ;           V 1.02  4/23/82 Don't drop DTR during reset and wait for
000012 ;                   DLYCNT before close.
000013 ;           V 1.30  1/07/83 Add comment field
000014 ;
000015
000016 DEVTYPE      .EQU          63                ;I/O CHAR DEV
000017 SUBTYPE      .EQU          01                ;DEV SUBTYPE
000018 MANID         .EQU          01                ;MANUFACTURER ID-APPLE
000019 RELEASE      .EQU          1300              ;RELEASE NUMBER-BCD FORMAT
000020
000021 ;
000022 ;   The macro SWITCH performs an N way branch based on a switch index.  The
000023 ;   maximum value of the switch index is 127 with bounds checking provided
000024 ;   as an option.  The macro uses the A and Y registers and alters the C,
000025 ;   Z, and N flags of the status register, but the X register is unchanged.
000026 ;
000027 ;           SWITCH  [index], [bounds], adrs_table, [*]
000028 ;
000029 ;           index   This is the variable that is to be used as the switch index.
000030 ;                   If omitted, the value in the accumulator is used.
000031 ;
000032 ;           bounds  This is the maximum allowable value for index.  If index
000033 ;                   exceeds this value, the carry bit will be set and execution
000034 ;                   will continue following the macro.  If bounds is omitted,
000035 ;                   no bounds checking will be performed.
000036 ;
000037 ;   adrs_table    This is a table of addresses (low byte first) used by the

```

```

000038 ;           switch.  The first entry corresponds to index zero.
000039 ;
000040 ;           *   If an asterisk is supplied as the fourth parameter, the
000041 ;           macro will push the switch address but will not exit to
000042 ;           it; execution will continue following the macro.  The
000043 ;           program may then load registers or set the status before
000044 ;           exiting to the switch address.
000045 ;
000046
000047           .MACRO           SWITCH
000048           .IF             "%1" <> ""                ;If PARM1 is present,
000049           LDA             %1                        ; Load A with switch index
000050           .ENDC
000051           .IF             "%2" <> ""                ;If PARM2 is present,
000052           CMP             #2+1                      ; Perform bounds checking
000053           BCS             $010                      ; on switch index
000054           .ENDC
000055           ASL             A
000056           TAY
000057           LDA             %3+1,Y                    ;Get switch address from table
000058           PHA             ; and push onto stack
000059           LDA             %3,Y
000060           PHA
000061           .IF             "%4" <> "*"              ;If PARM4 is omitted,
000062           RTS             ; Exit to code
000063           .ENDC
000064 $010           .ENDM
000065
000066 ;
000067 ;           INCREMENT WORD MACRO
000068 ;
000069
000070           .MACRO           INW
000071           INC             %1
000072           BNE             $210
000073           INC             %1+1
000074 $210           .ENDM
000075
000076 ;
000077 ;           INCREMENT ADDRESS MACRO
000078 ;
000079 ;           INCREMENTS 3 BYTE ADDRESS
000080 ;
000081
000082           .MACRO           INCADR

```

```

000083         INC             %1
000084         BNE             $310
000085         INC             %1+1
000086         BNE             $310             ;Bank overflow ?
000087         SEC                     ;Yes
000088         ROR             %1+1
000089         INC             %1+1+1400       ;Increment X byte
000090 $310         .ENDM
000091
000092         .MACRO           SET_1MHZ
000093
000094         LDA             E_REG
000095         ORA             #BITON7         ;Set 1 MHZ mode
000096         STA             E_REG
000097         .ENDM
000098
000099         .MACRO           SET_2MHZ
000100
000101         LDA             E_REG
000102         AND             #07F           ;Set 2 MHZ mode
000103         STA             E_REG
000104         .ENDM
000105
000106         .PROC            RS232
000107
000108 ;
000109 ;           SOS GLOBAL DATA AND SUBROUTINES
000110 ;
000111 ALLOCSIR     .EQU        1913           ;SOS interrupt allocation manager
000112 DEALCSIR    .EQU        1916           ;SOS interrupt deallocation manager
000113 SYSERR      .EQU        1928           ;SOS error return
000114 ;
000115 ;           SOS Error Codes
000116 ;
000117 XREQCODE    .EQU        20             ;Invalid request code
000118 XCTLCODE    .EQU        21             ;Invalid control/status code
000119 XCTLPARM    .EQU        22             ;Invalid control/status param
000120 XNOTOPEN    .EQU        23             ;Device not open
000121 XNOTAVIL    .EQU        24             ;Device not available
000122 XNORESRC    .EQU        25             ;Resource not available
000123 XBADOP      .EQU        26             ;Invalid operation for device
000124 ;
000125 ;           HARDWARE I/O ADDRESSES
000126 ;
000127 ACIADATA     .EQU        0C0F0         ;ACIA DATA REGISTER

```

```

000128 ACIASTAT      .EQU      0C0F1      ;ACIA STATUS REGISTER
000129 ACIACMD      .EQU      0C0F2      ;ACIA COMMAND REGISTER
000130 ACIACTL      .EQU      0C0F3      ;ACIA CONTROL REGISTER
000131 E_REG      .EQU      0FFDF      ;ENVIRONMENT REGISTER
000132 B_REG      .EQU      0FFEF      ;BANK REGISTER
000133 ;
000134 ;          GENERAL EQUATES
000135 ;
000136 TRUE      .EQU      80
000137 FALSE     .EQU      00
000138 BITON0     .EQU      01
000139 BITON1     .EQU      02
000140 BITON2     .EQU      04
000141 BITON3     .EQU      08
000142 BITON4     .EQU      10
000143 BITON6     .EQU      40
000144 BITON7     .EQU      80
000145 ASC_LF     .EQU      0A
000146 ASC_FF     .EQU      0C
000147 ASC_CR     .EQU      0D
000148 ;
000149          .PAGE
000150
000151          .WORD      0FFFF
000152          .WORD      73.
000153          .ASCII    "(C) Apple Computer 1981, 1982, 1983.  "
000154          .ASCII    "Built-in Serial Port RS-232 Driver."
000155
000156 ;-----
000157 ;
000158 ;          DEVICE INFORMATION BLOCK
000159 ;
000160 ;-----
000161 ;          DEVICE HEADER BLOCK
000162 ;-----
000163
000164 IDBLK      .WORD      0000      ;LINK TO NEXT DEVICE HANDLER
000165          .WORD      RS_MAIN     ;ENTRY POINT ADDRESS
000166          .BYTE      6          ;LENGTH OF DRIVER NAME
000167          .ASCII    ".RS232      " ;DRIVER NAME
000168          .BYTE      80,00,00    ;DEV NUM, DEV SLOT, DEV UNIT
000169          .BYTE      DEVTYP      ;DEVICE TYPE
000170          .BYTE      SUBTYP      ;DEV SUBTYPE
000171          .BYTE      00          ;FUTURE USE
000172          .WORD      0000      ;BLOCK COUNT-NOT USED

```

```

000173          .WORD      MANID          ;MANUFACTURER ID
000174          .WORD      RELEASE        ;RELEASE NUMBER-BCD
000175
000176 ;-----
000177 ;          DEVICE CONFIGURATION BLOCK
000178 ;-----
000179
000180 CNFGBLK      .WORD      12.          ;CONFIGURATION BLOCK LENGTH
000181
000182 DCB          .BYTE      06          ;BAUD RATE - 300
000183          .BYTE      22          ;Data format
000184 ;CTL - Hi nybble
000185 ;CMD - Lo nybble
000186          .BYTE      00          ;Carriage return delay
000187          .BYTE      00          ;Line feed delay
000188          .BYTE      00          ;Form feed delay
000189          .BYTE      00          ;00 - no protocol
000190 ;80 - XON/XOFF
000191 ;40 - ENQ/ACK
000192          .BYTE      13          ;Character to use as XOFF (or ENQ)
000193          .BYTE      11          ;Character to use as XON (or ACK)
000194          .BYTE      223.        ;Buffer level which triggers XOFF
000195          .BYTE      132.        ;Buffer level which triggers XON
000196          .BYTE      80.         ;Character count for ENQ/ACK
000197          .BYTE      00          ;Hardware handshake support
000198
000199 DCB_LN       .EQU        *-DCB
000200
000201          .ASCII      "(C) Apple Computer Inc. 1983"
000202
000203          .PAGE
000204 ;
000205 ;          SOS Device Handler Interface
000206 ;
000207
000208 SOSINT       .EQU        0C0
000209 REQCODE      .EQU        SOSINT+0    ;SOS request code
000210 BUFFER      .EQU        SOSINT+2    ;Buffer pointer
000211 REQCNT      .EQU        SOSINT+4    ;Requested count/Byte count
000212 CTLSTAT     .EQU        SOSINT+2    ;Control/status code
000213 CSLIST      .EQU        SOSINT+3    ;Control/status list pointer
000214 RETPTR      .EQU        SOSINT+8    ;Returned count pointer
000215
000216
000217 ;

```

```

000218 ; Zero Page Storage
000219 ;
000220
000221 ZPGSAVE      .EQU      SOSINT+0A      ;Saved zero page storage
000222
000223 ZPGTEMP      .EQU      ZPGSAVE+00      ;Temporary zero page storage
000224
000225 OPRODPTR     .EQU      0E1              ;Producer pointer
000226 ICSMRPTR     .EQU      0E2              ;Consumer pointer
000227 RETCNT      .EQU      0E3              ;Returned byte count word
000228
000229
000230 ;
000231 ; Private Variable Storage
000232 ;
000233
000234 SIRADDR      .WORD      SIRTABLE
000235 SIRTABLE     .BYTE      1,0              ;ACIA resource
000236             .WORD      ACIAMIH
000237 MIHBANK      .BYTE      0
000238 SIRCOUNT   .EQU      *-SIRTABLE
000239
000240 OPENFLG      .BYTE      FALSE           ;Device open flag
000241 IS_NEWLINE   .BYTE      FALSE           ;Bit 7 (1=new line mode)
000242 NEWLINE      .BYTE      00              ;Newline character
000243 IN_PROG     .BYTE      00              ;Bit 7 (1=XOFF in progress)
000244 ;Bit 6 (1=XOFF needs to be sent)
000245 SEND_XON    .BYTE      00              ;Bit 7 (1=XON needs to be sent)
000246 RTS_FALSE   .BYTE      00              ;Bit 7 (1=RTS false)
000247 NO_OUTPUT    .BYTE      00              ;Bit 7 (1=suspend output)
000248 DLYCNT      .BYTE      00              ;Delay count for MIH
000249 BRK_CNT      .BYTE      00              ;Interval count for Break signal
000250 CHAR_OUT     .BYTE      00              ;Output character count for ENQ/ACK
000251 IN_PROG1    .BYTE      00              ;Bit 7 (1=ENQ in progress)
000252
000253 ;
000254 ; Device control parameters
000255 ;
000256
000257 CNTL_PARAM   .BYTE      15.             ;List length
000258
000259 BAUD         .BYTE      00              ;BAUD RATE
000260 DFORMAT      .BYTE      00              ;Data format
000261 ;CTL - Hi nybble
000262 ;CMD - Lo nybble

```

```

000263 CRDELAY          .BYTE      00          ;Carriage return delay
000264 LFDELAY          .BYTE      00          ;Line feed delay
000265 FFDELAY          .BYTE      00          ;Form feed delay
000266 PROTOCOL         .BYTE      00          ;00 - none
000267 ;80 - XON/XOFF
000268 ;40 - ENQ/ACK
000269 CTLCHR1             .BYTE      00          ;Character to use as XOFF (or ENQ)
000270 CTLCHR2             .BYTE      00          ;Character to use as XON (or ACK)
000271 MAXBUF             .BYTE      00          ;Buffer level which triggers XOFF
000272 ;                (or RTS false)
000273 MINBUF             .BYTE      00          ;Buffer level which triggers XON
000274 ;                (or RTS true)
000275 CHARCNT             .BYTE      00          ;Character count for ENQ/ACK
000276 HDW_HSHAKE         .BYTE      00          ;Hardware handshake
000277 ;                Bit 7 (1=enabled)
000278 RD_IMMEDIATE        .BYTE      00          ;Bit 7 (1=read immediate mode)
000279 STAT_REG            .BYTE      00          ;Status reg - saved from last interrupt
000280 STAT_LATCH          .BYTE      00          ;Latched status bits - cleared by reset
000281 ;                or status request-1
000282 ;Bit 0 (1=parity error)
000283 ;Bit 1 (1=framing error)
000284 ;Bit 2 (1=overrun)
000285 ;Bit 5 (1=DCD went false)
000286 ;Bit 6 (1=DSR went false)
000287 ;Bit 7 (1=input character lost)
000288
000289 CNTL_LN             .EQU      *-CNTL_PARAM
000290
000291 ;
000292 ;                Data Buffers
000293 ;
000294
000295 OBUFCNT             .BYTE      0          ;Local output buffer byte count
000296 OSRODPTR            .BYTE      0          ;Producer buffer pointer
000297 OCSMRPTR            .BYTE      0          ;Consumer buffer pointer
000298 OLOCBUF             .BLOCK     0100,0      ;Local output buffer
000299
000300 IBUFCNT             .BYTE      0          ;Local input buffer byte count
000301 ISCSMRPTR           .BYTE      0          ;Input consumer pointer
000302 IPRODPTR            .BYTE      0          ;Input producer pointer
000303 ILOCBUF             .BLOCK     0100,0      ;Local input buffer
000304
000305 .PAGE
000306 ;-----
000307 ;

```



```

000308 ;      RS232 DRIVER - MAIN ENTRY POINT
000309 ;
000310 ;-----
000311
000312 RS_MAIN      .EQU      *
000313              SWITCH    REQCODE,8,RS_REQSW
000314
000315
000316 BADREQ      LDA      #XREQCODE      ;Invalid request code
000317              JSR      SYSERR
000318
000319
000320 NOTOPEN     LDA      #XNOTOPEN      ;Device not open
000321              JSR      SYSERR
000322
000323
000324 RS_REQSW     .EQU      *              ;RS232 driver request switch
000325              .WORD    RS_READ-1
000326              .WORD    RS_WRITE-1
000327              .WORD    RS_STAT-1
000328              .WORD    RS_CNTL-1
000329              .WORD    BADREQ-1
000330              .WORD    BADREQ-1
000331              .WORD    RS_OPEN-1
000332              .WORD    RS_CLOSE-1
000333              .WORD    RS_INIT-1
000334
000335              .PAGE
000336 ;-----
000337 ;
000338 ;      RS232 Driver -- Initialization Request
000339 ;
000340 ;-----
000341
000342 RS_INIT     .EQU      *
000343
000344              LDA      #FALSE
000345              STA      OPENFLG      ;Set serial port to not open
000346              CLC              ;Insure carry clear for load program
000347              RTS
000348
000349              .PAGE
000350 ;-----
000351 ;
000352 ;      RS232 Driver -- Open Request

```

```

000353 ;
000354 ;-----
000355
000356 RS_OPEN      .EQU      *
000357              BIT        OPENFLG      ;Serial Port open?
000358              BPL        $010         ; No
000359              LDA        #XNOTAVIL
000360              JSR        SYSERR
000361
000362 $010          LDA        B_REG
000363              AND        #0F
000364              STA        MIHBANK      ;Set interrupt handler bank
000365              LDA        #SIRCOUNT
000366              LDX        SIRADDR
000367              LDY        SIRADDR+1
000368              JSR        ALLOCSIR     ;Allocate the ACIA
000369              BCS        $020
000370
000371              LDX        #DCB_LN-1   ;Copy Device Configuration Block
000372 ; into device control parameters
000373
000374 $015          LDA        DCB,X
000375              STA        CNTL_PARAM+1,X
000376              DEX
000377              BPL        $015
000378
000379              LDA        #0
000380              STA        IS_NEWLINE   ;Set newline mode to False
000381              STA        NEWLINE     ;Clear newline character
000382              STA        RD_IMMEDIATE ;Read immediate mode off
000383              STA        IN_PROG     ;XOFF in progress flag off
000384              STA        SEND_XON   ;Send XON flag off
000385
000386              JSR        CNTL00      ;Set up ACIA
000387              LDA        #TRUE      ; and clear STAT_REG, STAT_LATCH,
000388 ; RTS_FALSE, NO_OUTPUT, CHAR_OUT,
000389 ; IN_PROG1, DLYCNT and BRK_CNT
000390              STA        OPENFLG    ;Set serial port open
000391              RTS
000392
000393 $020          LDA        #XNORESRC
000394              JSR        SYSERR
000395              .PAGE
000396 ;-----
000397 ;

```

```

000398 ;          RS232 Driver -- Close Request
000399 ;
000400 ;-----
000401
000402 RS_CLOSE      .EQU          *
000403              ASL            OPENFLG          ;Serial Port open?
000404              BCS            $05              ; Yes
000405              JMP            NOTOPEN
000406
000407 $05          LDA            OBUFCNT          ;Wait for write completion
000408              ORA            DLYCNT          ; and delay complete
000409              BNE            $05
000410
000411              PHP            ;Save interrupt status
000412              SEI            ;Disable interrupt system
000413              SET_1MHZ
000414              LDA            ACIACMD
000415              AND            #0F0            ;Disable Rcv/Xmit Interrupt
000416              STA            ACIACMD          ;DTR off, RTS off
000417              LDA            ACIASTAT        ;Clear any prior interrupt
000418              PLP            ;Restore interrupt status
000419
000420              LDA            #SIRCOUNT
000421              LDX            SIRADDR
000422              LDY            SIRADDR+1
000423              JSR            DEALCSIR         ;Deallocate the ACIA
000424              RTS
000425
000426              .PAGE
000427 ;-----
000428 ;
000429 ;          RS232 Driver -- Read Request
000430 ;
000431 ;-----
000432
000433 RS_READ        .EQU          *
000434              BIT            OPENFLG          ;Serial Port open?
000435              BMI            $05
000436              JMP            NOTOPEN
000437
000438 $05          LDA            ISCSMRPTR        ;Get CSMRPTR from driver storage
000439              STA            ICSMRPTR        ;Put in temporary zero page
000440              LDY            #00            ;Prevent offset
000441              STY            RETCNT          ;Zero return count
000442              STY            RETCNT+1

```

```

000443
000444         LDA         #0FF             ;One's complement count
000445         EOR         REQCNT
000446         STA         REQCNT
000447         LDA         #0FF
000448         EOR         REQCNT+1
000449         STA         REQCNT+1
000450
000451  $010        INC         REQCNT             ;Increment count
000452         BNE         $015             ;Is count zero ?
000453         INC         REQCNT+1
000454         BEQ         $099             ;Yes, terminate
000455
000456  $015        LDA         IBUFCNT         ;Is input buffer empty ?
000457         BNE         $020             ;No, continue
000458         BIT         RD_IMMEDIATE       ;Is read immediate mode set ?
000459         BPL         $015             ;No, loop until character received
000460         BMI         $099             ;Yes, terminate
000461
000462  $020        LDY         #0
000463         LDX         ICSMRPTR
000464         LDA         ILOCBUF,X         ;Get char from local input buffer
000465         STA         (BUFFER),Y       ;Send to user buffer
000466         PHA                     ;Save character on stack
000467         INCADR        BUFFER         ;Increment addr - user buffer pointer
000468         INC         ICSMRPTR
000469         DEC         IBUFCNT
000470         INW         RETCNT
000471
000472         LDA         MINBUF             ;Check if below min buffer level
000473         CMP         IBUFCNT         ;      (IBUFCNT < MINBUF ?)
000474         BCC         $025             ;No, continue
000475
000476         BIT         IN_PROG           ;Yes, XOFF in progress ?
000477         BMI         $022             ;Yes, send XON
000478
000479         BIT         RTS_FALSE        ;Is RTS false ?
000480         BPL         $025             ;No, continue
000481
000482         PHP                     ;Save interrupt status
000483         SEI                     ;Disable interrupt system
000484         SET_1MHZ      ;Yes, set 1 MHZ mode
000485         LDA         ACIACMD         ;Set RTS true and
000486         AND         #0F2             ; enable xmit interrupt
000487         ORA         #05

```

```

000488          STA          ACIACMD          ;Set to [xxxx01x1]
000489          LDA          #0
000490          STA          RTS_FALSE        ;Clear RTS_FALSE
000491          SET_2MHZ
000492          PLP
000493          JMP          $025             ;Restore interrupt status
000494
000495 $022          LDA          #80          ;Send XON
000496          STA          SEND_XON         ;Set flag
000497          JSR          PRIME_OUT        ;Prime output routine
000498
000499 $025          PLA
000500          BIT          IS_NEWLINE        ;Is newline mode set ?
000501          BPL          $010            ;No, get next char
000502          CMP          NEWLINE         ;Yes, is char terminator ?
000503          BEQ          $099            ;If yes, terminate
000504          JMP          $010            ;No, get next char
000505
000506 $099          LDY          #0
000507          LDA          ICSPTR          ;Terminate
000508          STA          ICSPTR          ;Save pointer
000509
000510          LDA          RETCNT           ;Get count of returned bytes
000511          STA          (RETPTR),Y      ;Send to user
000512          LDA          RETCNT+1
000513          INY
000514          STA          (RETPTR),Y
000515
000516          RTS
000517
000518          .PAGE
000519 ;-----
000520 ;
000521 ;          RS232 Driver -- Write Request
000522 ;
000523 ;-----
000524
000525 RS_WRITE      .EQU          *
000526          BIT          OPENFLG          ;Serial Port open?
000527          BMI          $05
000528          JMP          NOTOPEN
000529
000530
000531 $05          LDA          OSPPTR          ;Get PRODPTR from driver storage
000532          STA          OPTR            ;Put in temporary zero page

```

```

000533
000534          LDA          #0FF          ;One's complement count
000535          EOR          REQCNT
000536          STA          REQCNT
000537          LDA          #0FF
000538          EOR          REQCNT+1
000539          STA          REQCNT+1
000540
000541  $010          INC          REQCNT          ;Increment count
000542          BNE          $030          ;Is count zero ?
000543          INC          REQCNT+1
000544          BNE          $030          ;No
000545
000546          JSR          PRIME_OUT        ;Prime consumer
000547          LDA          OPRODPTR        ;Save producer pointer in driver
000548          STA          OSPRODPTR
000549          RTS
000550          ;Return to user
000551  $030          LDX          OBUFCNT        ;Is local output buffer full ?
000552          INX
000553          BNE          $040          ;No
000554
000555          JSR          PRIME_OUT        ;Local buffer is full, prime consumer
000556          JMP          $030
000557
000558  $040          LDY          #00
000559          LDA          (BUFFER),Y        ;Get character from user buffer
000560          INCADR        BUFFER        ;Increment addr - user buffer ptr
000561
000562          LDX          OPRODPTR        ;Get producer pointer
000563          STA          OLOCBUF,X        ;Store character in local buffer
000564          INC          OPRODPTR        ;Advance local buffer
000565          INC          OBUFCNT        ;Advance count
000566          BNE          $010          ;Branch always taken
000567
000568
000569          .PAGE
000570  ;-----
000571  ;
000572  ;      RS232 Driver -- Status Request
000573  ;
000574  ;-----
000575
000576  RS_STAT        .EQU          *
000577          BIT          OPENFLG        ;Serial Port open?

```

```

000578          BMI          $05
000579          JMP          NOTOPEN
000580 $05      SWITCH     CTLSTAT,3,STATSW
000581
000582
000583 BADCTL    LDA          #XCTLCODE          ;Invalid control code
000584          JSR          SYSERR
000585
000586
000587 STATSW     .WORD       STAT00-1
000588          .WORD       STAT01-1
000589          .WORD       STAT02-1
000590          .WORD       STAT03-1
000591
000592 STAT00     RTS          ;0 -- NOP
000593
000594
000595 STAT01     LDY          #0          ;1 -- Retrieve device control
000596 ; parameters (including RD_IMMEDIATE
000597 ; STAT_REG and STAT_LATCH)
000598          LDA          (CSLIST),Y
000599          CMP          CNTL_PARAM        ;Check for room in status list
000600          BCS          $01          ; >= OK
000601
000602          LDA          #XCTLPARM        ; < NG
000603          JSR          SYSERR
000604
000605 $01      LDY          #CNTL_LN-1
000606          PHP          ;Save interrupt status
000607          SEI          ;Disable interrupt system
000608
000609 $05      LDA          CNTL_PARAM,Y
000610          STA          (CSLIST),Y
000611          DEY
000612          BPL          $05
000613
000614          INY
000615          STY          STAT_LATCH        ;Clear status latch bits
000616          PLP          ;Restore interrupt status
000617          RTS
000618
000619
000620 STAT02     LDY          #0          ;2 -- Get newline character
000621
000622          LDA          IS_NEWLINE

```

```

000623         STA         (CSLIST),Y
000624         INY
000625         LDA         NEWLINE
000626         STA         (CSLIST),Y
000627
000628         RTS
000629
000630 STAT03     LDY         #0                ;3 -- Retrieve driver buffer info
000631
000632         LDA         #0FF                ;Output buffer size
000633         JSR         CNTOUT
000634         LDA         OBUFCNT            ;Number of chars in output buffer
000635         JSR         CNTOUT
000636         LDA         #0FF                ;Input buffer size
000637         JSR         CNTOUT
000638         LDA         IBUFCNT            ;Number of chars in input buffer
000639         JSR         CNTOUT
000640
000641         RTS
000642
000643 CNTOUT      STA         (CSLIST),Y
000644         INY
000645         LDA         #0                ; high byte (0)
000646         STA         (CSLIST),Y
000647         INY
000648
000649         RTS
000650
000651         .PAGE
000652 ;-----
000653 ;
000654 ;       RS232 Driver -- Control Request
000655 ;
000656 ;-----
000657
000658 RS_CNTL     .EQU         *
000659         BIT         OPENFLG            ;Serial Port open?
000660         BMI         $05                ; Ok
000661         JMP         NOTOPEN
000662 $05        SWITCH     CTLSTAT,3,CNTLSW
000663         JMP         BADCTL
000664
000665
000666 CNTLSW     .WORD        CNTL00-1
000667         .WORD        CNTL01-1

```



```

000668      .WORD      CNTL02-1
000669      .WORD      CNTL03-1
000670
000671
000672 CNTL00      .EQU      *                ;0 -- Reset device
000673
000674      BIT          IN_PROG          ;XOFF in progress ?
000675      BPL          $020              ;No, continue
000676
000677      LDA          #80                ;Yes, send XON
000678      STA          SEND_XON          ;Set flag
000679      JSR          PRIME_OUT         ;Prime output routine
000680
000681 $015      BIT          SEND_XON          ;Wait until XON gets out
000682      BMI          $015
000683
000684 $020      PHP                ;Save interrupt status
000685      SEI                ;Disable interrupt system
000686      LDA          BAUD              ;Validate data rate
000687      AND          #00F
000688      STA          BAUD
000689
000690      SET_1MHZ
000691
000692      LDA          #0
000693      STA          IBUFCNT           ;Zero Input Buffer count
000694      STA          OBUFCNT           ;Zero Output Buffer count
000695      STA          DLYCNT            ;Zero delay count
000696      STA          BRK_CNT           ;Zero interval count
000697
000698      STA          OSPRODPTR         ;Zero pointers
000699      STA          OCSMRPTR
000700      STA          ISCSMRPTR
000701      STA          IPRODPTR
000702
000703      STA          RTS_FALSE         ;Clear RTS false flag
000704      STA          NO_OUTPUT        ;Clear suspend output flag
000705      STA          CHAR_OUT         ;Zero output character count
000706      STA          IN_PROG1        ;ENQ in progress flag off
000707      STA          STAT_LATCH       ;Clear status latch bits
000708      LDA          ACIASTAT
000709      STA          STAT_REG         ;Save status reg
000710
000711      LDA          DFORMAT          ;Validate data format
000712      AND          #0E0

```

```

000713          ORA          #BITON4          ;Set receiver clock source to internal
000714          ORA          BAUD
000715          LDX          #03
000716          CPX          BAUD          ;If data rate is 110 baud
000717          BNE          $025
000718
000719          ORA          #BITON7          ; force two stop bits
000720
000721  $025          STA          ACIACTL          ;Set up ACIA control register
000722          LDA          DFORMAT
000723          ASL          A
000724          ASL          A
000725          ASL          A
000726          ASL          A
000727          AND          #0E0
000728          ORA          #09          ;Xmit disabled, Rcv enabled
000729  ;DTR and RTS on
000730          STA          ACIACMD          ;Set up ACIA command register
000731
000732          PLP          ;Restore interrupt status
000733          RTS
000734
000735
000736  CNTL01          LDY          #0          ;1 -- Load device control parameters
000737  ; except STAT_REG and STAT_LATCH
000738          LDA          (CSLIST),Y
000739          CMP          CNTL_PARAM          ;Check length of control list
000740          BEQ          $01          ; = OK
000741
000742          LDA          #XCTLPARM          ; NG
000743          JSR          SYSERR
000744
000745  $01          LDY          #CNTL_LN-3
000746
000747  $05          LDA          (CSLIST),Y
000748          STA          CNTL_PARAM,Y
000749          DEY
000750          BPL          $05
000751
000752          JSR          CNTL00          ;Set up ACIA
000753
000754          RTS
000755
000756  CNTL02          .EQU          *          ;2 -- Set New Line Character
000757

```

```

000758         LDY         #0
000759         LDA         (CSLIST),Y
000760         STA         IS_NEWLINE
000761         INY
000762         LDA         (CSLIST),Y
000763         STA         NEWLINE
000764
000765         RTS
000766
000767 CNTL03      .EQU      *                ;3 -- Transmit Break
000768
000769 $05         LDA         OBUFCNT        ;Wait for write completion
000770         BNE         $05
000771
000772         TAY
000773         LDA         (CSLIST),Y        ;Get number of break intervals
000774         BMI         $050             ;Too large, return
000775         BEQ         $050             ;Zero, return
000776         CMP         #101.           ;Check if > 100 (23.3 sec)
000777         BCS         $050             ;Too large, return
000778
000779         STA         BRK_CNT          ;Save interval count
000780         PHP
000781         SEI
000782         SET_1MHZ          ;Set 1 MHz mode
000783         LDA         ACIACMD          ;Transmit Break
000784         ORA         #0C
000785         STA         ACIACMD          ;Set to [xxxx11xx]
000786         LDA         #0
000787         STA         RTS_FALSE        ;Clear RTS false
000788         PLP
000789         ;Restore interrupt status
000790 $010        LDY         #181.         ;This double loop takes 233 ms
000791         LDX         #0                ; in 1 MHz mode
000792 $015        DEX
000793         BNE         $015
000794         DEY
000795         BNE         $015
000796
000797         DEC         BRK_CNT          ;Loop for interval count
000798         BNE         $010
000799
000800         JSR         PRIME_OUT        ;Prime output routine
000801
000802 $050        RTS

```

```

000803
000804         .PAGE
000805 ;-----
000806 ;
000807 ;         ACIA MASTER INTERRUPT HANDLER
000808 ;
000809 ;-----
000810
000811 ACIAMIH         .EQU         *
000812
000813         STY         STAT_REG         ;Save current status reg
000814
000815         TYA
000816         AND         #BITON3         ;Check receiver data reg full
000817         BEQ         $010         ;No, continue
000818
000819         TYA         ;Input interrupt
000820         AND         #67
000821         ORA         STAT_LATCH
000822         STA         STAT_LATCH         ;Latch status bits
000823         JMP         RS_IN
000824
000825 $010         TYA         ;Treat as output interrupt
000826         AND         #60
000827         ORA         STAT_LATCH
000828         STA         STAT_LATCH         ;Latch status bits
000829         JMP         RS_OUT
000830
000831 RS_IN         .EQU         *         ;Receive next character
000832
000833         SET_1MHZ
000834         LDX         ACIADATA         ;Read character
000835         SET_2MHZ
000836         TXA
000837
000838         BIT         PROTOCOL         ;Is XON/XOFF protocol mode set?
000839         BPL         $016         ;No, continue
000840
000841         CMP         CTLCHR1         ;Yes, check for XOFF
000842         BNE         $010         ;No
000843
000844         LDA         #TRUE         ;Yes, suspend output
000845         STA         NO_OUTPUT
000846         JMP         RS_OUT
000847

```

```

000848 $010      CMP      CTLCHR2      ;Check for XON
000849          BNE      $015          ;No
000850
000851          LDA      #FALSE          ;Yes, resume output
000852          STA      NO_OUTPUT
000853          BEQ      RS_OUT          ;Always taken
000854
000855 $015      LDX      IBUFCNT      ;Check if max buffer level exceeded
000856          CPX      MAXBUF          ;      (IBUFCNT >= MAXBUF ?)
000857          BCC      $020          ;No, continue
000858
000859          BIT      IN_PROG          ;Yes, check if XOFF in progress
000860          BMI      $020          ;Yes, continue
000861
000862          LDX      #BITON6          ;No, set XOFF needs to be sent
000863          STX      IN_PROG
000864          BNE      $020          ;Branch always taken
000865
000866 $016      BVC      $017          ;Is ENQ/ACK protocol mode set?
000867
000868          CMP      CTLCHR2          ;Yes, check for ACK
000869          BNE      $020          ;No, continue
000870
000871          LDA      CHARCNT          ;Yes, reset output char count
000872          STA      CHAR_OUT
000873          LDA      #0
000874          STA      IN_PROG1          ;Clear ENQ in progress
000875          BEQ      RS_OUT          ;Always taken
000876
000877 $017      BIT      HDW_HSHAKE      ;Is Hardware handshake enabled?
000878          BPL      $020          ;No, continue
000879
000880          LDX      IBUFCNT          ;Check if max buffer level exceeded
000881          CPX      MAXBUF          ;      (IBUFCNT >= MAXBUF ?)
000882          BCC      $020          ;No, continue
000883
000884          LDX      BRK_CNT          ;Check for Break in progress
000885          BNE      $020          ;Yes, continue (can't change RTS)
000886
000887          PHA                      ;No, save character on stack
000888          LDA      #BITON7
000889          STA      RTS_FALSE
000890          SET_1MHZ
000891          LDA      ACIACMD          ;Set RTS to false
000892          AND      #0F3            ; Xmit interrupt will be disabled

```

```

000893          STA          ACIACMD          ;   ACIA set to [xxxx00xx]
000894          SET_2MHZ
000895          PLA                      ;Retrieve character from stack
000896
000897   $020      LDX          IBUFCNT        ;Is buffer full ?
000898          INX
000899          BNE          $025             ;No, continue
000900
000901          LDA          #BITON7          ;Yes, latch char lost bit
000902          ORA          STAT_LATCH
000903          STA          STAT_LATCH
000904          BMI          RS_OUT          ;Always taken
000905
000906   $025      LDX          IPRODPTR       ;Address in local buffer to store data
000907          STA          ILOCBUF,X       ;Store char in local input buffer
000908          INC          IBUFCNT
000909          INC          IPRODPTR
000910
000911   RS_OUT    .EQU          *           ;Output next character
000912
000913          LDA          BRK_CNT          ;Check for Break in progress
000914          BEQ          $001             ;No, continue
000915          JMP          RETURN          ;Yes, return
000916
000917   $001      BIT          HDW_HSHAKE     ;Hardware handshake mode enabled ?
000918          BPL          $003             ;No, continue
000919
000920          BIT          RTS_FALSE        ;Yes, check for RTS false
000921          BPL          $002             ;RTS true, continue
000922          JMP          RETURN          ;RTS false, return
000923
000924   $002      LDA          STAT_REG       ;Check DSR and DCD status
000925          AND          #60
000926          BEQ          $003             ;DSR and DCD true, continue
000927
000928          LDA          DLYCNT           ;DSR or DCD false, disable xmit int
000929          BNE          $011             ; unless delay in progress
000930          JMP          D_XMIT
000931
000932   $003      LDA          #BITON4        ;Check xmit data reg empty
000933          BIT          STAT_REG
000934          BNE          $004             ;Reg empty, continue
000935
000936          JMP          E_XMIT           ;Reg not empty, enable xmit interrupt
000937

```

```

000938 $004      BIT      IN_PROG      ;XOFF need to be sent ?
000939          BVC      $005          ;No, continue
000940
000941          LDA      #BITON7        ;Yes, set XOFF in progress
000942          STA      IN_PROG
000943          LDA      CTLCHR1        ;Send XOFF
000944          JMP      $020
000945
000946 $005      BIT      SEND_XON      ;XON need to be sent ?
000947          BPL      $010          ;No, continue
000948
000949          LDA      #0              ;Yes, clear flags
000950          STA      SEND_XON
000951          STA      IN_PROG
000952          LDA      CTLCHR2        ;Send XON
000953          JMP      $020
000954
000955 $010      LDA      DLYCNT        ;Any transmit delay in progress ?
000956          BEQ      $015          ;No
000957
000958 $011      DEC      DLYCNT        ;Yes, decrement count
000959          JMP      E_XMIT
000960
000961 $015      LDX      OBUFCNT        ;Is local output buffer count zero ?
000962          BEQ      D_XMIT        ;Yes, disable xmit interrupt and return
000963
000964          BIT      NO_OUTPUT        ;Is output suspended ?
000965          BMI      D_XMIT        ;Yes, disable xmit interrupt and return
000966
000967          BIT      PROTOCOL        ;Is, ENQ/ACK protocol mode set?
000968          BVC      $018          ;No, continue
000969
000970          LDA      CHAR_OUT        ;Yes, check output char count
000971          BNE      $016          ;Count not yet exhausted, send char
000972
000973          BIT      IN_PROG1        ;Check for ENQ in progress
000974          BMI      D_XMIT        ;Yes, disable xmit interrupt and return
000975
000976          LDA      #BITON7        ;No, set ENQ in progress
000977          STA      IN_PROG1
000978          LDA      CTLCHR1        ;Send ENQ
000979          JMP      $020
000980
000981 $016      DEC      CHAR_OUT        ;Decrement output char count
000982 $018      LDX      OCSMRPTR      ;No, get consumer pointer

```

```

000983          LDA          OLOCBUF,X          ;Get character from buffer
000984          DEC          OBUFCNT
000985          INC          OCSMRPTR
000986
000987 $020          TAX
000988          SET_1MHZ
000989          STX          ACIADATA          ;Send character
000990
000991          CPX          #ASC_CR          ;Check for any delay
000992          BNE          $022
000993          LDA          CRDELAY
000994          JMP          $024
000995
000996 $022          CPX          #ASC_LF
000997          BNE          $023
000998          LDA          LFDELAY
000999          JMP          $024
001000
001001 $023          CPX          #ASC_FF
001002          BNE          E_XMIT
001003          LDA          FFDELAY
001004
001005 $024          STA          DLYCNT
001006
001007 E_XMIT          SET_1MHZ
001008          LDA          ACIACMD          ;Enable transmit interrupt
001009          AND          #0F2
001010          ORA          #05          ;Set to [xxxx01x1]
001011          STA          ACIACMD
001012
001013          RTS          ;Return to user
001014
001015 D_XMIT          SET_1MHZ
001016          LDA          ACIACMD          ;Disable transmit interrupt
001017          AND          #0F2
001018          ORA          #09          ;Set to [xxxx10x1]
001019          STA          ACIACMD
001020 RETURN          RTS          ;Return to user
001021
001022 PRIME_OUT      .EQU          *          ;Called by Read, Write and Control
001023 ; request routines
001024
001025          PHP          ;Save interrupt status
001026          SEI          ;Disable interrupt system
001027          BIT          RTS_FALSE

```

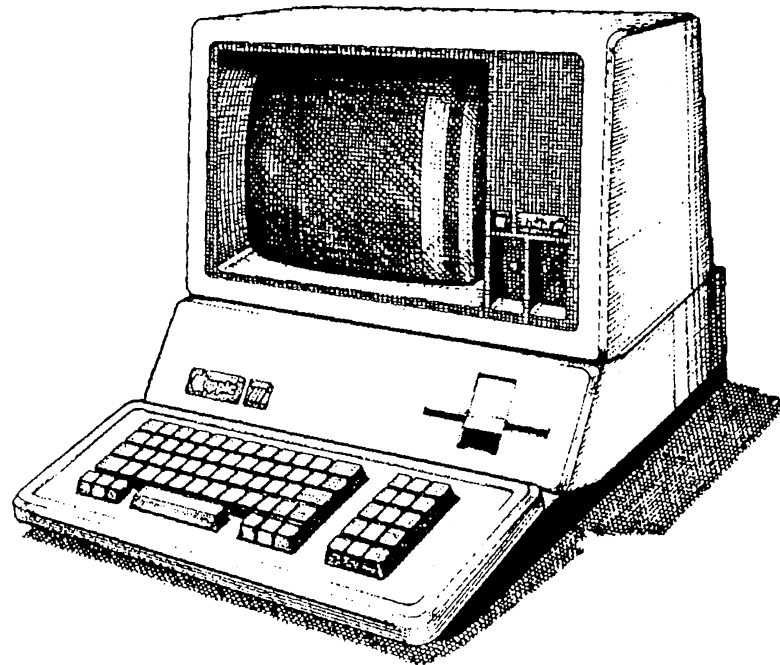


```

001028          BMI          $010          ;Return if RTS false
001029          JSR          E_XMIT        ;Enable transmit interrupt
001030          SET_2MHZ
001031  $010      PLP          ;Restore interrupt status
001032
001033          RTS          ;Return
001034
001035          .END
001036

; #####
; #   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