Super Serial Card: Accessing it through Machine Language

Although Apple's Super Serial Card can be used from Applesoft Basic, it is often desirable to use machine language to increase the speed with which characters are sent and received. The assembler program below illustrates a method of communicating with another Apple computer through the Super Serial Card. You may use this routine as a starting point for your own program.

On page 291 of the Apple IIe Reference manual and on pages 261 to 265 of the Apple IIc Reference Manual there are listsof the registers and entry points used by routines resident in the Super Serial Card. The equates in the program below use these locations, as well as input/output hooks found in the Apple II family of computers.

The initialization routine (INIT) stores the address of the Super Serial Card's initialization routine in CSW (the Apple II monitor character output hook). This activates the card for output by jumping to COUT. Following this, DOS or Prodos hooks are reinstalled.

The OUTput routine checks the 6551 status port bit 4. If this is equal to zero, the previous character has not yet been sent, so we must check the status byte again until that register is clear. When the value in bit 4 becomes one, the 6551 is ready to send another character. To accomplish this, simply store the data in the transmit register (TDREG) of the chip.

Bit 3 of the status port is checked by the INput routine. If this bit is zero, the program either loops continuously or returns to the calling program, depending on the state of the return flag found in location $FF. If bit 3 is one, a character is waiting at the input port, and the character is then read from the read register (RDREG) of the 6551.

The DEMO portion of this program calls the INIT routine, and sends each letter of the alphabet to the connected device. After each character is sent, the program waits to see if a response has been received from the external device. If a character is waiting, the program ends.

```
1     * Super Serial Card - Demo of accessing it through machine language
10
*************************************************************************
11              ORG     $2000
12    COUT      EQU     $FDED     ; CHARACTER OUT IN MONITOR
13    CSWL      EQU     $36       ; OUTPUT HOOK
14    CSWH      EQU     $37
15    WAIT      EQU     $FCA8     ; MONITOR ROUTINE TO WAIT
```

```
16    ;
17    ;  SSC EQUATES
18    ;
19    DIPSW1     EQU    $C081    ; +N0  DIPSWITCH BLOCK 1
20    DIPSW2     EQU    $C082    ; +N0  DIPSWITCH BLOCK 2
21    TDREG      EQU    $C088    ; +N0  6551 DATA REGISTER
22    RDREG      EQU    $C088    ; +N0  6551 DATA REGISTER
23    STATUS     EQU    $C089    ; +N0  6551 STATUS REGISTER
24    RESET      EQU    $C089    ; +N0  6551 SOFTWARE RESET
25    COMMAND    EQU    $C08A    ; +N0  6551 COMMAND REG
26    CONTROL    EQU    $C08B    ; +N0  6551 CONTROL REG
27    ;
28    START      JMP    DEMO     ; SKIP AROUND ALL THE SUBROUTINES
29    ;
30    ; USE THE SSC FIRMWARE TO INITIALIZE THE 6551.
31    ;
32    INIT     LDA    CSWL       ; STORE THE CURRENT CSW
33             PHA               ; SO THAT WE DO NOT DISCONNECT
34             LDA    CSWH       ; DOS OR PRODOS
35             PHA
36             LDA    #$00       ; STORE $Cs00 IN CSW
37             STA    CSWL
38             STX    CSWH       ; THIS ALREADY CONTAINS $Cs
39             LDA    #$00
40             JSR    COUT       ; JUMP TO COUT TO INIT THE CARD
41             PLA
42             STA    CSWH       ; RESTORE THE DOS OR PRODOS
43             PLA               ; HOOKS AND THEN RETURN
44             STA    CSWL
45             RTS
46    ;
47    ; OUTPUT A CHARACTER TO 6551
48    ;
49    OUT    PHA                ; STORE DATA ON STACK
50    OLP    LDA    STATUS,Y    ; CHECK BIT 4 OF STATUS BYTE
51           AND    #$10        ; TO SEE IF IT'S OK TO SEND
52           BEQ    OLP         ; CHARACTER WAITING TO GO OUT
53           PLA                ; GET DATA BACK FROM STACK
54           STA    TDREG,Y     ; AND OUTPUT THE CHARACTER
55           RTS
56    ;
57    ; INPUT A CHARACTER FROM 6551
58    ;
59    IN       LDA    STATUS,Y
60             AND    #$08       ; BIT 3 OF STATUS
61             BEQ    INTST      ; NO CHAR WAITING TO BE RECEIVED
62             LDA    RDREG,Y    ; GET THE INPUT FROM 6551
63             RTS
64    INTST    LDA    $FF        ; CHECK RETURN FLAG
65             BNE    IN         ; IF NOT 0 THEN WAIT FOR INPUT
66             RTS               ; IF ZERO, DON'T WAIT
67    ;
68    ;    BEGIN THE DEMO PROGRAM
```

```
69    ;
70    DEMO       LDY    #$10     ; Y CONTAINS $s0 - DEMO USES SLOT 1
71               LDX    #$C1     ; LOAD X WITH $Cs
72               JSR    INIT     ; INIT THE CARD
73               LDA    #$FF     ; SET RETURN FLAG FOR INPUT
74               STA    $FF      ; FF MEANS WAIT FOR CHAR
75               JSR    IN       ; INPUT A CHARACTER - SEE ABOVE
76    OLOOP      LDX    #$41     ; OUTPUT THE ASCII CODES
77    OLP1       TXA             ; FROM A-Z TO THE SSC. IT WILL STOP
78               JSR    OUT      ; WHEN THE SSC RECEIVES A CHAR.
79               LDA    #$80     ; DELAY BETWEEN CHARACTERS
80               JSR    WAIT     ; TO ALLOW TIME FOR INPUT.
81               LDA    #$00
82               STA    $FF      ; RETURN IF NO CHARS WAITING
83               JSR    IN       ; CHECK FOR A CHARACTER
84               BNE    ALLDONE  ; THEY SENT SOMETHING - WE END
85               INX
86               CPX    #$5B     ; THE LETTER 'Z'
87               BNE    OLP1
88               LDA    #$0D
89               JSR    OUT      ; SEND A CARRIAGE RETURN
90               JMP    OLOOP    ; BEGIN THE ALPHABET AGAIN
91    ALLDONE    RTS             ; END ROUTINE
Apple Technical Communications


Keywords:  <None>

=====================================================================

This information is from the Apple Technical Information Library.

19960215 11:05:19.00
```

Tech Info Library Article Number: 1918