



Tech Info Library

Apple IIGS: 6502 communications applications (1 of 2)

Revised: 6/10/87
Security: Everyone

Apple IIGS: 6502 communications applications (1 of 2)

=====

Some assembly language programmers may want to convert 6502 communications software to use the IIGS logic boards to the full. To insure future compatibility when using the Apple IIGS serial ports through assembly language, you should use the built-in firmware calls. The firmware works very well, is very fast, and also provides you with a built-in interrupt handler and input/output buffers. All of these features can be managed through ROM calls.

More advanced use (bit and register handling) would require familiarity with the "Z8030/Z8530 Serial Communication Chip Technical Manual" from Zilog. Information there reveals that communications on the 8530 is much more complicated than that on the 6551; a straight conversion may not be that simple.

Zilog Inc.
210 Hacienda ave.
Campbell CA. 95008

Here are some examples of how you might use the serial ports with the built-in firmware. NOTE: all these examples use the Pascal interface in the ROM.

To initialize your programs and the port you might use something like this:

```
InitVector    equ    $C20D        ; pointer to the init routine in ROM
ReadVector    equ    $C20E        ; Pointer to the read char routine
WriteVector    equ    $C20F        ; pointer to the Write routine
StatVector    equ    $C210
ExtendVect    equ    $C212        ; Pointer to the extended interface routine

InitPort      equ    $F8          ; set up some area's for indirect jumps
ReadChar      equ    $FA          ; to be used in the program to make the
WriteChar     equ    $FC          ; calls to ROM
StatusCall    equ    $FE          ;
ExtendCall    equ    $F6          ; New vector for extended interface
```

```

InitPort    lda    InitVector    ; First set up your indirect pointers
            sta    InitPort

            ldy    #$C2          ; make sure to set the high byte
            sty    InitPort+1
            lda    ReadVector
            sta    ReadChar
            sty    ReadChar+1
            lda    WriteVector
            sta    WriteChar
            sty    WriteChar+1
            lda    StatVector
            sta    StatCall
            sty    StatCall+1
            lda    ExtendVect
            sta    ExtendCall
            sty    ExtendCall+1

            ldx    #$C2          ; Now make the init call to the ROM
            ldy    #$20          ; Always set up the X and Y Regs first
            jsr    (InitPort)    ; and indirect jump to the init routine
            cpx    #0            ; test for an error
            beq    *+5           ; if its zero skip next jump
            jmp    Error         ; if non-zero an error occured call error rtn
            RTS

```

Tech Info Library Article Number:2226