Apple IIGS: 6502 communications applications (2 of 2)

Revised: 6/10/87
Security: Everyone

Apple IIGS: 6502 communications applications (2 of 2)

==========

...

```
CharWait    NOP              ; This rtn checks the input buffer for chars
            ldx   #$C2       ; and returns the result in the carry flag
            ldy   #$20
            lda   #$01       ; status call 1 is input status
            jsr   (StatusCall)
            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

OutEmpty    NOP              ;This rtn checks the output buffer for chars
            ldx   #$C2       ; and returns the result in the carry flag
            ldy   #$20
            lda   #$01       ; status call 1 is input status
            jsr   (StatusCall)
            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

GetChar     NOP              ; This routine gets an input char from the
                             ; serial port and places it into <A>
            ldx   #$C2
            ldy   #$20
            jsr   (ReadChar)
            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

PutChar     NOP              ; This routine writes a char in <A> to the
                             ; serial port
            ldx   #$C2
            ldy   #$20
```

```
              jsr   (WriteChar)
              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

DTROn         NOP                ; This routine turns the DTR line on
              lda   #$00         ; this rtn uses the extended interface calls
              sta   DTRData+4    ; set up the data block
              sta   DTRData+5
              ldy   #$00         ; on entry <x> <y> and <A> contain address of
              ldx   DTRDPtr+1    ; the call parm block
              lda   DTRDPtr
              jsr   (ExtendCall)
              rts

DTROn         NOP                ; This routine turns the DTR line off
              lda   #$80         ; this rtn uses the extended interface calls
              sta   DTRData+4    ; set up the data block
              sta   DTRData+5
              ldy   #$00         ; on entry <x> <y> and <A> contain address of
              ldx   DTRDPtr+1    ; the call parm block
              lda   DTRDPtr
              jsr   (ExtendCall)
              rts

DTRDPtr       dw    DTRData      ; pointer to our data structure
DTRData       DFB   03           ; # of params in call
              DFB   $0B          ; call # (0B means set DTR line)
              DW    0000         ; word for result code
              DW    0000         ; call data
                                 ;(0000 means clear DTR, 8000 means set it)

SwitchBaud    NOP                ; This rtn shows how to change the Baud Rate
                                 ; it turns on 1200 Baud and buffering. It
could
                                 ; be expanded to handle all port commands
              ldx   #00          ; Init the string index
SB0010        lda   theString,x  ; get the next byte to send
              phx                ; save index
              jsr   PutChar      ; Write out the char to the port
              plx                ; get the index back
              cpx   StrLength    ; are we done?
              beq   SB0020       ; if so then end
              inx                ; if not get the next char and continue
              jmp   SB0010
SB0020        rts

theString     dfb   01           ; command char (Control-A)
              asc   '8B'         ; 1200 baud command
StrLength     dfb   02
```

Tech Info Library Article Number:2227