```
//
//  Serialcode.s
//  256 Byte Prom P8 and 512 Byte PROM P9A (second version) for Apple II Serial Card
//  P9A differs from P9 by adding RTS/ACK software flow control to output and
//  by removing batch transfer support.  RTS/ACK cannot be disabled making this firmware useless
//  when interfacing to peripherials that don't support RTS/ACK protocol
//
//  Created by mwillega circa 5/29/13.
//
//
```

```
; Entry point for slot# control-P, slot# control-K, PR slot#, IN slot#.
; during initialization, separate entry points for read and write are
; set up
;
; Assembled for slot 1, thus the C1XX address
; Default entry point is CN00 where N is slot number
;
; The bit instruction is used to set V status bit
C100- 2C 58 FF  BIT $FF58     ;
C103- 70 04     BVS $C109     ;Always branch to CN09

; Once Initialized, monitor character in calls to CN05
; set carry for input, to be tested CN35
C105- 38        SEC

; CLEVER CODE ALERT
; due to the set carry, the following instruction never branches
; the second byte  of the conditional branch instruction
; is not used for branching, but to
; clear the carry (0x18) at monitor character output entry point
; this use of the conditional branch acts as a
; skip always instruction
C106- 90 18     BCC $C120

; CN07  second byte of BCC instrucion (duplicated in this listing)
; this is entry point for monitor character output calls
; and is skipped by
; monitor character input calls
C107- 18        CLC

; clear overflow
C108-           B8                    CLV

; init, input and output calls all merge back here
; init has V set,  output has carry clear, and input carry set
C109- 08        PHP
C10A- 78        SEI
C10B- 86 35     STX $35
C10D- 48        PHA
C10E- 8A        TXA
C10F- 48        PHA
C110- 98        TYA
C111- 48        PHA

; Cfff access clears all I/O shared memory space  (C8000 to Ceff) flip flip
; Since code is running out of this card, the flip flop on this card will
; imediately be set, again, leaving this card in possession of c8000-ceff
; address range
C112- AD FF CF LDA $CFFF
; JSR to RET instruction puts current address on stack
;(interrupts blocked)
C115- 20 58 FF JSR $FF58
C118- BA TSX                   ; move stack pointer to X
C119- BD 00 01 LDA $0100,X    ;retrieve MSB of address CN
; unused space in the graphics page are used to save slot and peripheral
;card info, see the reference manual
C11C- 8D F8 07 STA $07F8      ;save CN to 7F8 as active card
                              ;(apple II convention)
C11F- 0A        ASL
C120- 0A        ASL
C121- 0A        ASL
C122- 0A        ASL
; Convert to N0 and save in 6F8, used later for access to spaces in
; graphics area reserved for I/O cards
C123- 8D F8 06 STA $06F8

C126- A8        TAY
C127- 68        PLA
C128- 68        PLA
C129- 68        PLA
C12A- 28        PLP
C12B- 9A        TXS
C12C- AE F8 07  LDX $07F8
C12F- 48        PHA
; overflow was cleared during entry by the init call
; and set by read and write calls
; now we test it and branch to init handler at C800
; otherwise the card is already initialized and we skip init
C130- 50 03 BVC $C135
C132- 20 00 C8 JSR $C800      ; call init in prom P9/P9A
; if output (carry clear) goto CN7A
C135- 90 43     BCC $C17A
```

```
; input handling starts here
C137- AD 78 06  LDA $0678
C13A- 48        PHA
C13B- 08        PHP
C13C- 30 0F     BMI $C14D
C13E- A4 35     LDY $35
C140- F0 0B     BEQ $C14D
C142- 88        DEY
C143- D9 00 02 CMP $0200,Y
C146- D0 05     BNE $C14D
C148- 09 E0     ORA #$E0
C14A- 99 00 02  STA $0200,Y
;
; Call Shift in Handler in P9/P9A
;
C14D- 20 4D C8  JSR $C84D
C150- 28        PLP
C151- 68        PLA
C152- 49 9B     EOR #$9B
C154- D0 01     BNE $C157
C156- 18        CLC
C157- A9 DF     LDA #$DF
C159- 90 03     BCC $C15E
C15B- 1D 38 07  ORA $0738,X
C15E- 85 35     STA $35
C160- 68        PLA
C161- A4 24     LDY $24
C163- 08        PHP
C164- 4C DC C8  JMP $C8DC    ;call to PROM P9/P9A
C167- A0 00     LDY #$00
C169- BD 38 05  LDA $0538,X
C16C- FD 38 06  SBC $0638,X
C16F- C9 F8     CMP #$F8
C171- 90 03     BCC $C176
C173- 69 27     ADC #$27
C173- 69 27     ADC #$27
C175- A8        TAY
C176- 38        SEC
;
; exit input handling in P9/P9A prom
;
C177- 4C 23 C9  JMP $C923
;
; start output handling here
;
C17A- BD B8 04  LDA $04B8,X
C17D- 10 1F     BPL $C19E
C17F- 29 7F     AND #$7F
C181- 9D B8 04  STA $04B8,X
C184- BD 38 07  LDA $0738,X
C187- 4A        LSR
C188- 68        PLA
C189- 49 8D     EOR #$8D
C18B- D0 09     BNE $C196
C18D- 9D 38 05  STA $0538,X
C190- 90 04     BCC $C196
C192- A9 8A     LDA #$8A
C194- B0 34     BCS $C1CA
C196- BD 38 07  LDA $0738,X
C199- 30 CC     BMI $C167
C19B- 4C C9 C8  JMP $C8C9
C19E- BD 38 07  LDA $0738,X
C1A1- 10 14     BPL $C1B7
C1A3- A5 24     LDA $24
C1A5- DD 38 05  CMP $0538,X
C1A8- B0 0D     BCS $C1B7
C1AA- C9 11     CMP #$11
C1AC- B0 09     BCS $C1B7
C1AE- 09 F0     ORA #$F0
C1B0- 3D 38 05  AND $0538,X
C1B3- 65 24     ADC $24
C1B5- 85 24     STA $24
C1B7- BD 38 05  LDA $0538,X
C1BA- C5 24     CMP $24
C1BC- 68        PLA
C1BD- B0 03     BCS $C1C2
C1BF- 48        PHA
C1C0- A9 A0     LDA #$A0
C1C2- 2C 58 FF  BIT $FF58
C1C5- F0 03     BEQ $C1CA
C1C7- FE 38 05  INC $0538,X
C1CA- 08        PHP
C1CB- 48        PHA
C1CC- 9D B8 05  STA $05B8,X
; Call Shift out in P9 or P9a
C1CF- 20 AA C9  JSR $C9AA
C1D2- 68        PLA
C1D3- 49 8D     EOR #$8D
C1D5- D0 16     BNE $C1ED
C1D7- 9D 38 05  STA $0538,X
C1DA- 85 24     STA $24
C1DC- BD 38 07  LDA $0738,X
C1DF- 48        PHA
C1E0- 29 40     AND #$40
C1E2- D0 03     BNE $C1E7
C1E4- 20 A8 FC  JSR $FCA8
C1E7- 68        PLA
C1E8- 4A        LSR
C1E9- A9 8A     LDA #$8A
C1EB- B0 DE     BCS $C1CB
C1ED- BD 38 06  LDA $0638,X
C1F0- F0 07     BEQ $C1F9
C1F2- FD 38 05  SBC $0538,X
C1F5- A9 8D     LDA #$8D
C1F7- 90 D2     BCC $C1CB
C1F9- 28        PLP
C1FA- 90 BB     BCC $C1B7
C1FC- B0 98     BCS $C196
C1FE- 00        BRK
C1FE- 00        BRK
C1FF- 00        BRK
; end of PROM P8
```

```
;
; C800-CEFF area  (512 byte x 8 PROM 9A)
;
; Initialization code - same start address as PROM P9
;
; read status register at C080 + N0 where N = slot number
; entry into this routine has Y = N0
; X = CN where N = slot number (1-7)
;
; read switches
; 1-3(bits 0-2) = baud
; 4(bit 3) = CR delay
; 5-6(bits 4-5) line width
; 7(bit 6) = line feed
C800- B9 80 C0  LDA $C080,Y
C803- 48        PHA
; save baud rate and convert to index
C804- 29 07     AND #$07
C806- A8        TAY
; fetch counter for SW based baud rate generation
C807- B9 31 C9  LDA $C931,Y
; save in slot dependant memory
; CN+3B8 = 479 through 47F depending upon N
C80A- 9D B8 03  STA $03B8,X
C80D- 68        PLA
C80E- 4A        LSR
C80F- 09 03     ORA #$03
; save switches 4-7
C811- 9D B8 04  STA $04B8,X
C814- 4A        LSR
C815- 4A        LSR
C816- 4A        LSR
; now we have switches 5 - 7
C817- 48        PHA
; save line width
C818- 29 03     AND #$03
C81A- A8        TAY
; use as index to get number of columns
C81B- B9 39 C9  LDA $C939,Y
; save in CN+438 = 4F9 through 4FF depending upon slot
C81E- 9D 38 04  STA $0438,X
C821- 68        PLA
; shift off bit 5
C822- 4A        LSR
; switch 6 set and 7 not?
C823- C0 01     CPY #$01
; move switch 6 to to MSB
C825- 6A        ROR
; or switch 6 together with 0x60
C826- 09 60     ORA #$60
; save 60 or 70 to  CN+738 = 7F9 through 7FF
C828- 9D 38 07  STA $0738,X
; preinitialize RTS/ACK counter
C82B- A9 00     LDA #$00
C82D- 9D 38 06  STA $0638,X
C830- 9D B8 06  STA $06B8,X
;
; at this point we check to see if read (KSWL/KSWH) or
; write (CSWL/CSWH) output vectors have been set up already
; The SLOT control-P or PR#N  command should have
; set the CSWL/CSWH to CN00
; if SLOT control-K or IN#N was typed, the KSWL/KSWH vector
; is also set to CN00
;
;
; if CSWL/CSWH is already set up for this card, we know
; that the init call is for read since the
; write driver vector is already set up
;
; X=CN = CSWH & 0 != CSWL
; then set up output
C833- E4 37     CPX $37
C835- D0 0C     BNE $C843
C837- C5 36     CMP $36
C839- D0 08     BNE $C843
; CN00 is in CSWL/CSWH after SLOT control-P
;
; Change the write vector to CN07, so we call
; output driver next time a character needs to be output
; instead of initializing again
;
C83B- A9 07     LDA #$07
C83D- 85 36     STA $36
;
; done with initialization, clear carry to execute output
C83F- 18        CLC
C840- 60        RTS
C841- 18        CLC
C842- 60        RTS
;
; if not SLOT control-P, then check input vector
C843- E4 39     CPX $39
; something wacky if KSWH != CN, exit
C845- D0 FA     BNE $C841
;
; KSWH is set already, no need to check KSWL
; now set input vector KSWH/KSWL
; to call CN05 next time character needs to be input
; instead of calling initialization entry point
;
C847- A9 05     LDA #$05
C849- 85 38     STA $38
; set carry to execute input when we return to caller
C84B- 38        SEC
C84C- 60        RTS
```

```
;
; Shift IN
; input handler - same start address as PROM P9
;
C84D- 78        SEI
C84E- BD B8 04  LDA $04B8,X
C851- 29 03     AND #$03
C853- 4A        LSR
C854- 6E 78 07  ROR $0778
C857- 48        PHA
C858- A9 09     LDA #$09
C85A- EA        NOP
C85B- 85 35     STA $35
C85D- AC F8 06  LDY $06F8
C860- 38        SEC
C861- B0 08     BCS $C86B
C863- AC F8 06  LDY $06F8
C866- C6 35     DEC $35
C868- F0 40     BEQ $C8AA
C86A- 18        CLC
C86B- B9 80 C0  LDA $C080,Y
C86E- 90 1F     BCC $C88F
C870- 10 0D     BPL $C87F
C872- AD 00 C0  LDA $C000
C875- 10 F4     BPL $C86B
C877- 9D B8 05  STA $05B8,X
C877- 9D B8 05  STA $05B8,X
C87A- 2C 10 C0  BIT $C010
C87D- 68        PLA
C87E- 60        RTS
C87F- BD B8 03  LDA $03B8,X
C882- A0 09     LDY #$09
C884- 88        DEY
C885- D0 FD     BNE $C884
C887- E9 01     SBC #$01
C889- F0 D8     BEQ $C863
C88B- A0 0E     LDY #$0E
C88D- D0 F5     BNE $C884
C88F- A8        TAY
C890- 2A        ROL
C891- 7E B8 05  ROR $05B8,X
C894- 98        TYA
C895- 4D 78 07  EOR $0778
C898- 8D 78 07  STA $0778
C89B- BD B8 03  LDA $03B8,X
C89E- 38        SEC
C89F- E9 01     SBC #$01
C8A1- F0 C0     BEQ $C863
C8A3- A0 09     LDY #$09
C8A5- 88        DEY
C8A6- D0 FD     BNE $C8A5
C8A8- F0 F5     BEQ $C89F
C8AA- 68        PLA
C8AB- D0 08     BNE $C8B5
C8AD- B9 80 C0  LDA $C080,Y
C8B0- 4D 78 07  EOR $0778
C8B3- 30 96     BMI $C84B
C8B5- B9 80 C0  LDA $C080,Y
C8B8- 10 FB     BPL $C8B5
C8BA- A9 0A     LDA #$0A
C8BC- E9 09     SBC #$09
C8BE- EA        NOP
C8BF- A8        TAY
C8C0- 88        DEY
C8C1- F0 89     BEQ $C84C
C8C3- 7E B8 05  ROR $05B8,X
C8C6- 38        SEC
C8C7- B0 F7     BCS $C8C0
C8C9- 20 48 C9  JSR $C948
C8CC- 68        PLA
C8CD- A8        TAY
C8CE- 68        PLA
C8CF- AA        TAX
C8D0- 68        PLA
C8D1- 09 80     ORA #$80
C8D3- 49 E0     EOR #$E0
C8D5- C9 1F     CMP #$1F
C8D7- 90 6B     BCC $C944
C8D9- B0 67     BCS $C942
C8DB- 00        BRK
C8DC- 91 28     STA ($28),Y    ;entry point from prom P8
C8DE- BD B8 05  LDA $05B8,X
C8E1- 09 80     ORA #$80
C8E3- C9 95     CMP #$95
C8E5- D0 02     BNE $C8E9
C8E7- B1 28     LDA ($28),Y
C8E9- C9 E0     CMP #$E0
C8EB- 90 11     BCC $C8FE
C8ED- 25 35     AND $35
C8EF- C9 E0     CMP #$E0
C8F1- 90 0B     BCC $C8FE
C8F3- A4 37     LDY $37
C8F5- C0 FD     CPY #$FD
C8F5- C0 FD     CPY #$FD
C8F7- F0 03     BEQ $C8FC
C8F9- 29 7F     AND #$7F
C8FB- AC 29 1F  LDY $1F29
C8FE- 8D 78 06  STA $0678
C901- 28        PLP
C902- 1E B8 04  ASL $04B8,X
C905- 38        SEC
C906- 7E B8 04  ROR $04B8,X
C909- 90 1A     BCC $C925
C90B- 20 48 C9  JSR $C948
C90E- A0 00     LDY #$00
C910- BD 38 05  LDA $0538,X
C913- 38        SEC
C914- FD 38 04  SBC $0438,X
C917- C9 F8     CMP #$F8
C919- 90 03     BCC $C91E
C91B- 69 27     ADC #$27
C91D- A8        TAY
```

```
C91E- 84 24     STY $24
C920- 38        SEC
C921- B0 02     BCS $C925
C923- B0 E6     BCS $C90B     ; exit entry point from PROM P8
C925- 68        PLA
C926- A8        TAY
C927- 68        PLA
C928- AA        TAX
C929- 68        PLA
C92A- B0 03     BCS $C92F
C92C- AD 78 06  LDA $0678
C92F- 28        PLP
C930- 60        RTS
;
; table converts baud rate switches to counter
;
C931- B0                    ;110
C932- 90                    ;134.5
C933- 40                    ;300
C934- 10                    ;1200
C935- 08                    ;2400
C936- 04                    ;4800
C937- 02                    ;9600
C938- 01                    ;19200
;
; terminal width table
;
C939- 29                    ;40
C93A- 48                    ;72
C93B- 50                    ;80
C93E- 84                    ;132
;garbage here
C93D- 60        RTS
C93E- 00        BRK
C93F- 00        BRK
C940- 00        BRK
C941- 60        RTS
C942- 49        E0 EOR #$E0
C944- 28        PLP
C945- 4C        F0 FD JMP $FDF0
;
; backspace handling - used to be batch code here
;
C948- A5        3C LDA $3C
C94A- 49        08 EOR #$08
C94C- 0A        ASL
C94D- F0 04     BEQ $C953
C94F- 49 EE     EOR #$EE
C951- D0 08     BNE $C95B
C953- DE 38 05  DEC $0538,X
C956- 10 03     BPL $C95B
C958- 9D 38 05  STA $0538,X
C95B- 60        RTS
C95C- 00        BRK
C95D- 00        BRK
C95E- 00        BRK
;
; This new version PROM P9A
; does CTS/ASK handshake before actually sending
; replaces  batch transfer code which originally
; was here
;
C95F- BD B8 05  LDA $05B8,X
C962- 85 3C     STA $3C
C964- BD B8 06 LDA $06B8,X
C967- C9 67     CMP #$67
C967- C9 67     CMP #$67
C969- 90 10     BCC $C97B
C96B- C9 6C     CMP #$6C
C96D- B0 1F     BCS $C98E
C96F- C9 6B     CMP #$6B
C971- A5 3C     LDA $3C
C973- 49 9B     EOR #$9B
C975- 29 7F     AND #$7F
C977- D0 15     BNE $C98E
C979- B0 16     BCS $C991
C97B- A9 83     LDA #$83
C97D- 20 93 C9  JSR $C993
C980- 20 4D C8  JSR $C84D
C983- BD B8 05  LDA $05B8,X
C986- 49 86     EOR #$86
C988- 0A        ASL
C989- D0 F0     BNE $C97B
C98B- 9D B8 06  STA $06B8,X
C98E- DE B8 06  DEC $06B8,X
C991- A5 3C     LDA $3C
C993- 9D B8 05  STA $05B8,X
C996- 49 8D     EOR #$8D
C998- 0A        ASL
C999- D0 0A     BNE $C9A5
C99B- BD B8 04  LDA $04B8,X
C99E- 29 04     AND #$04
C9A0- D0 03     BNE $C9A5
C9A2- 9D B8 06  STA $06B8,X
C9A5- 78        SEI
C9A6- 18        CLC
C9A7- 08        PHP
C9A8- 90 03     BCC $C9AD
;
; Shift out routine starts here - - same start address as PROM P9
; this address used to maintain
; compatibility with PROM P8
;
C9AA- 4C 5F C9  JMP $C95F
```

```
;
; shift out here
;
C9AD- 38        SEC
C9AE- 08        PHP
C9AF- A0 09     LDY #$09
C9B1- EA        NOP
C9B2- 84 35     STY $35
C9B4- A9 01     LDA #$01
C9B6- 3D B8 04  AND $04B8,X
C9B9- 0D F8 06  ORA $06F8
C9BC- 8D 78 07  STA $0778
C9BF- A9 00     LDA #$00
C9C1- 1E B8 05  ASL $05B8,X
C9C4- 7E B8 05  ROR $05B8,X
C9C7- 2A        ROL
C9C8- 0D F8 06  ORA $06F8
C9CB- A8        TAY
C9CC- 29 01     AND #$01
C9CE- 4D 78 07  EOR $0778
C9D1- 8D 78 07  STA $0778
C9D4- 38        SEC
C9D5- B9 81 C0  LDA $C081,Y
C9D8- BD B8 03  LDA $03B8,X
C9DB- E9 01     SBC #$01
C9DD- F0 07     BEQ $C9E6
C9DF- A0 09     LDY #$09
C9E1- 88        DEY
C9E2- D0 FD     BNE $C9E1
C9E4- F0 F5     BEQ $C9DB
C9E6- A0 02     LDY #$02
C9E8- EA        NOP
C9E9- C6 35     DEC $35
C9EB- D0 D7     BNE $C9C4
C9ED- 28        PLP
C9ED- 28        PLP
C9EE- 90 0E     BCC $C9FE
C9F0- 84 35     STY $35
C9F2- BD B8 04  LDA $04B8,X
C9F5- 29 02     AND #$02
C9F7- 4A        LSR
C9F8- 0D 78 07  ORA $0778
C9FB- A8        TAY
C9FC- D0 D6     BNE $C9D4
C9FE- A8        TAY
C9FF- 60        RTS
```