beneater / **wozmon.s**

Created last year • Report abuse

Code | Revisions 2 | Stars 29 | Forks 10 | Embed <script src="https://| | Download ZIP

Revisions

Split | Unified

beneater revised this gist on May 31, 2023.

1 changed file with **38 additions** and **35 deletions**.

73 wozmon.s

```diff
@@ -12,76 +12,76 @@ MODE  = $2B                          ; $00=XAM, $7F=STOR, $AE=BLOCK XAM

 IN    = $0200                        ; Input buffer

-KBD   = $D010                        ; PIA.A keyboard input
-KBDCR = $D011                        ; PIA.A keyboard control register
-DSP   = $D012                        ; PIA.B display output register
-DSPCR = $D013                        ; PIA.B display control register
+ACIA_DATA   = $5000
+ACIA_STATUS = $5001
+ACIA_CMD    = $5002
+ACIA_CTRL   = $5003

 RESET:
-                CLD                  ; Clear decimal arithmetic mode.
-                CLI
-                LDY     #$7F         ; Mask for DSP data direction reg.
-                STY     DSP          ; Set it up.
-                LDA     #$A7         ; KBD and DSP control register mask.
-                STA     KBDCR        ; Enable interrupts, set CA1, CB1 for
-                STA     DSPCR        ;  positive edge sense/output mode.
+                LDA     #$1F         ; 8-N-1, 19200 baud.
+                STA     ACIA_CTRL
+                LDA     #$0B         ; No parity, no echo, no interrupts.
+                STA     ACIA_CMD
+                LDA     #$1B         ; Begin with escape.

 NOTCR:
-                CMP     #$DF         ; Backspace key?
                 BEQ     BACKSPACE    ; Yes.
-                CMP     #$9B         ; ESC?
                 BEQ     ESCAPE       ; Yes.
                 INY                  ; Advance text index.
                 BPL     NEXTCHAR     ; Auto ESC if line longer than 127.

 ESCAPE:
-                LDA     #$DC         ; "\".
                 JSR     ECHO         ; Output it.

 GETLINE:
-                LDA     #$8D         ; Send CR
                 JSR     ECHO

                 LDY     #$01         ; Initialize text index.
 BACKSPACE:      DEY                  ; Back up text index.
                 BMI     GETLINE      ; Beyond start of line, reinitialize.

 NEXTCHAR:
-                LDA     KBDCR        ; Key ready?
-                BPL     NEXTCHAR     ; Loop until ready.
-                LDA     KBD          ; Load character. B7 should be '1'.
+                LDA     ACIA_STATUS  ; Check status.
+                AND     #$08         ; Key ready?
+                BEQ     NEXTCHAR     ; Loop until ready.
+                LDA     ACIA_DATA    ; Load character. B7 will be '0'.
                 STA     IN,Y         ; Add to text buffer.
                 JSR     ECHO         ; Display character.
-                CMP     #$8D         ; CR?
                 BNE     NOTCR        ; No.

                 LDY     #$FF         ; Reset text index.
                 LDA     #$00         ; For XAM mode.
                 TAX                  ; X=0.
+SETBLOCK:
+                ASL
 SETSTOR:
                 ASL                  ; Leaves $7B if setting STOR mode.
-SETMODE:
-                STA     MODE         ; $00 = XAM, $7B = STOR, $AE = BLOK XAM.
+                STA     MODE         ; $00 = XAM, $74 = STOR, $B8 = BLOK XAM.

 BLSKIP:
                 INY                  ; Advance text index.
 NEXTITEM:
                 LDA     IN,Y         ; Get character.
-                CMP     #$8D         ; CR?
+                CMP     #$0D         ; CR?
                 BEQ     GETLINE      ; Yes, done this line.
-                CMP     #$AE         ; "."?
                 BCC     BLSKIP       ; Skip delimiter.
-                BEQ     SETMODE      ; Set BLOCK XAM mode.
-                CMP     #$BA         ; ":"?
+                BEQ     SETBLOCK     ; Set BLOCK XAM mode.
+                CMP     #$3A         ; ":"?
```

```
75                  BEQ     SETSTOR         ; Yes, set STOR mode.
76  -               CMP     #$D2            ; "R"?
77                  BEQ     RUN             ; Yes, run user program.
78                  STX     L               ; $00 -> L.
79                  STX     H               ;    and H.
80                  STY     YSAV            ; Save Y for comparison
81
82  NEXTHEX:
83                  LDA     IN,Y            ; Get character for hex test.
84  -               EOR     #$B0            ; Map digits to $0-9.
85                  CMP     #$0A            ; Digit?
86                  BCC     DIG             ; Yes.
87                  ADC     #$88            ; Map letter "A"-"F" to $FA-FF.
```

```
75                  BEQ     SETSTOR         ; Yes, set STOR mode.
76  +               CMP     #$52            ; "R"?
77                  BEQ     RUN             ; Yes, run user program.
78                  STX     L               ; $00 -> L.
79                  STX     H               ;    and H.
80                  STY     YSAV            ; Save Y for comparison
81
82  NEXTHEX:
83                  LDA     IN,Y            ; Get character for hex test.
84  +               EOR     #$30            ; Map digits to $0-9.
85                  CMP     #$0A            ; Digit?
86                  BCC     DIG             ; Yes.
87                  ADC     #$88            ; Map letter "A"-"F" to $FA-FF.
```

```
...  @@ -132,17 +132,17 @@ SETADR:         LDA     L-1,X           ; Copy hex data to
132
133 NXTPRNT:
134                 BNE     PRDATA          ; NE means no address to print.
135 -               LDA     #$8D            ; CR.
136                 JSR     ECHO            ; Output it.
137                 LDA     XAMH            ; 'Examine index' high-order byte.
138                 JSR     PRBYTE          ; Output it in hex format.
139                 LDA     XAML            ; Low-order 'examine index' byte.
140                 JSR     PRBYTE          ; Output it in hex format.
141 -               LDA     #$BA            ; ":".
142                 JSR     ECHO            ; Output it.
143
144 PRDATA:
145 -               LDA     #$A0            ; Blank.
146                 JSR     ECHO            ; Output it.
147                 LDA     (XAML,X)        ; Get data byte at 'examine index'.
148                 JSR     PRBYTE          ; Output it in hex format.
```

```
132
133 NXTPRNT:
134                 BNE     PRDATA          ; NE means no address to print.
135 +               LDA     #$0D            ; CR.
136                 JSR     ECHO            ; Output it.
137                 LDA     XAMH            ; 'Examine index' high-order byte.
138                 JSR     PRBYTE          ; Output it in hex format.
139                 LDA     XAML            ; Low-order 'examine index' byte.
140                 JSR     PRBYTE          ; Output it in hex format.
141 +               LDA     #$3A            ; ":".
142                 JSR     ECHO            ; Output it.
143
144 PRDATA:
145 +               LDA     #$20            ; Blank.
146                 JSR     ECHO            ; Output it.
147                 LDA     (XAML,X)        ; Get data byte at 'examine index'.
148                 JSR     PRBYTE          ; Output it in hex format.
```

```
...  @@ -173,15 +173,18 @@ PRBYTE:
173
174 PRHEX:
175                 AND     #$0F            ; Mask LSD for hex print.
176 -               ORA     #$B0            ; Add "0".
177 -               CMP     #$BA            ; Digit?
178                 BCC     ECHO            ; Yes, output it.
179                 ADC     #$06            ; Add offset for letter.
180
181 ECHO:
182 -               BIT     DSP             ; DA bit (B7) cleared yet?
183 -               BMI     ECHO            ; No, Wait for display.
184 -               STA     DSP             ; Output character. Sets DA.
185                 RTS                     ; Return.
186
187   .org $FFFA
...
```

```
173
174 PRHEX:
175                 AND     #$0F            ; Mask LSD for hex print.
176 +               ORA     #$30            ; Add "0".
177 +               CMP     #$3A            ; Digit?
178                 BCC     ECHO            ; Yes, output it.
179                 ADC     #$06            ; Add offset for letter.
180
181 ECHO:
182 +               PHA                     ; Save A.
183 +               STA     ACIA_DATA       ; Output character.
184 +               LDA     #$FF            ; Initialize delay loop.
185 + TXDELAY:      DEC                     ; Decrement A.
186 +               BNE     TXDELAY         ; Until A gets to 0.
187 +               PLA                     ; Restore A.
188                 RTS                     ; Return.
189
190   .org $FFFA
```

beneater **created** this gist on May 31, 2023.

191 wozmon.s

```
...  @@ -0,0 +1,191 @@
1  +   .org $8000
2  +   .org $ff00
3  +
4  + XAML  = $24                           ; Last "opened" location Low
5  + XAMH  = $25                           ; Last "opened" location High
6  + STL   = $26                           ; Store address Low
7  + STH   = $27                           ; Store address High
8  + L     = $28                           ; Hex value parsing Low
9  + H     = $29                           ; Hex value parsing High
10 + YSAV  = $2A                           ; Used to see if hex value is given
11 + MODE  = $2B                           ; $00=XAM, $7F=STOR, $AE=BLOCK XAM
12 +
13 + IN    = $0200                         ; Input buffer
14 +
15 + KBD   = $D010                         ; PIA.A keyboard input
16 + KBDCR = $D011                         ; PIA.A keyboard control register
17 + DSP   = $D012                         ; PIA.B display output register
18 + DSPCR = $D013                         ; PIA.B display control register
```

```
+
+ RESET:
+                CLD                     ; Clear decimal arithmetic mode.
+                CLI
+                LDY     #$7F            ; Mask for DSP data direction reg.
+                STY     DSP             ; Set it up.
+                LDA     #$A7            ; KBD and DSP control register mask.
+                STA     KBDCR           ; Enable interrupts, set CA1, CB1 for
+                STA     DSPCR           ;  positive edge sense/output mode.
+
+ NOTCR:
+                CMP     #$DF            ; Backspace key?
+                BEQ     BACKSPACE       ; Yes.
+                CMP     #$9B            ; ESC?
+                BEQ     ESCAPE          ; Yes.
+                INY                     ; Advance text index.
+                BPL     NEXTCHAR        ; Auto ESC if line longer than 127.
+
+ ESCAPE:
+                LDA     #$DC            ; "\".
+                JSR     ECHO            ; Output it.
+
+ GETLINE:
+                LDA     #$8D            ; Send CR
+                JSR     ECHO
+
+                LDY     #$01            ; Initialize text index.
+ BACKSPACE:     DEY                     ; Back up text index.
+                BMI     GETLINE         ; Beyond start of line, reinitialize.
+
+ NEXTCHAR:
+                LDA     KBDCR           ; Key ready?
+                BPL     NEXTCHAR        ; Loop until ready.
+                LDA     KBD             ; Load character. B7 should be '1'.
+                STA     IN,Y            ; Add to text buffer.
+                JSR     ECHO            ; Display character.
+                CMP     #$8D            ; CR?
+                BNE     NOTCR           ; No.
+
+                LDY     #$FF            ; Reset text index.
+                LDA     #$00            ; For XAM mode.
+                TAX                     ; X=0.
+ SETSTOR:
+                ASL                     ; Leaves $7B if setting STOR mode.
+ SETMODE:
+                STA     MODE            ; $00 = XAM, $7B = STOR, $AE = BLOK XAM.
+ BLSKIP:
+                INY                     ; Advance text index.
+ NEXTITEM:
+                LDA     IN,Y            ; Get character.
+                CMP     #$8D            ; CR?
+                BEQ     GETLINE         ; Yes, done this line.
+                CMP     #$AE            ; "."?
+                BCC     BLSKIP          ; Skip delimiter.
+                BEQ     SETMODE         ; Set BLOCK XAM mode.
+                CMP     #$BA            ; ":"?
+                BEQ     SETSTOR         ; Yes, set STOR mode.
+                CMP     #$D2            ; "R"?
+                BEQ     RUN             ; Yes, run user program.
+                STX     L               ; $00 -> L.
+                STX     H               ;    and H.
+                STY     YSAV            ; Save Y for comparison
+
+ NEXTHEX:
+                LDA     IN,Y            ; Get character for hex test.
+                EOR     #$B0            ; Map digits to $0-9.
+                CMP     #$0A            ; Digit?
+                BCC     DIG             ; Yes.
+                ADC     #$88            ; Map letter "A"-"F" to $FA-FF.
+                CMP     #$FA            ; Hex letter?
+                BCC     NOTHEX          ; No, character not hex.
+ DIG:
+                ASL
+                ASL                     ; Hex digit to MSD of A.
+                ASL
+                ASL
+
+                LDX     #$04            ; Shift count.
+ HEXSHIFT:
```

```
+                ASL                     ; Hex digit left, MSB to carry.
+                ROL     L               ; Rotate into LSD.
+                ROL     H               ; Rotate into MSD's.
+                DEX                     ; Done 4 shifts?
+                BNE     HEXSHIFT        ; No, loop.
+                INY                     ; Advance text index.
+                BNE     NEXTHEX         ; Always taken. Check next character for hex.
+
+ NOTHEX:
+                CPY     YSAV            ; Check if L, H empty (no hex digits).
+                BEQ     ESCAPE          ; Yes, generate ESC sequence.
+
+                BIT     MODE            ; Test MODE byte.
+                BVC     NOTSTOR         ; B6=0 is STOR, 1 is XAM and BLOCK XAM.
+
+                LDA     L               ; LSD's of hex data.
+                STA     (STL,X)         ; Store current 'store index'.
+                INC     STL             ; Increment store index.
+                BNE     NEXTITEM        ; Get next item (no carry).
+                INC     STH             ; Add carry to 'store index' high order.
+ TONEXTITEM:    JMP     NEXTITEM        ; Get next command item.
+
+ RUN:
+                JMP     (XAML)          ; Run at current XAM index.
+
+ NOTSTOR:
+                BMI     XAMNEXT         ; B7 = 0 for XAM, 1 for BLOCK XAM.
+
+                LDX     #$02            ; Byte count.
+ SETADR:        LDA     L-1,X           ; Copy hex data to
+                STA     STL-1,X         ;  'store index'.
+                STA     XAML-1,X        ; And to 'XAM index'.
+                DEX                     ; Next of 2 bytes.
+                BNE     SETADR          ; Loop unless X = 0.
+
+ NXTPRNT:
+                BNE     PRDATA          ; NE means no address to print.
+                LDA     #$8D            ; CR.
+                JSR     ECHO            ; Output it.
+                LDA     XAMH            ; 'Examine index' high-order byte.
+                JSR     PRBYTE          ; Output it in hex format.
+                LDA     XAML            ; Low-order 'examine index' byte.
+                JSR     PRBYTE          ; Output it in hex format.
+                LDA     #$BA            ; ":".
+                JSR     ECHO            ; Output it.
+
+ PRDATA:
+                LDA     #$A0            ; Blank.
+                JSR     ECHO            ; Output it.
+                LDA     (XAML,X)        ; Get data byte at 'examine index'.
+                JSR     PRBYTE          ; Output it in hex format.
+ XAMNEXT:       STX     MODE            ; 0 -> MODE (XAM mode).
+                LDA     XAML
+                CMP     L               ; Compare 'examine index' to hex data.
+                LDA     XAMH
+                SBC     H
+                BCS     TONEXTITEM      ; Not less, so no more data to output.
+
+                INC     XAML
+                BNE     MOD8CHK         ; Increment 'examine index'.
+                INC     XAMH
+
+ MOD8CHK:
+                LDA     XAML            ; Check low-order 'examine index' byte
+                AND     #$07            ; For MOD 8 = 0
+                BPL     NXTPRNT         ; Always taken.
+
+ PRBYTE:
+                PHA                     ; Save A for LSD.
+                LSR
+                LSR
+                LSR                     ; MSD to LSD position.
+                LSR
+                JSR     PRHEX           ; Output hex digit.
+                PLA                     ; Restore A.
+
+ PRHEX:
```

```
+                AND     #$0F          ; Mask LSD for hex print.
+                ORA     #$B0          ; Add "0".
+                CMP     #$BA          ; Digit?
+                BCC     ECHO          ; Yes, output it.
+                ADC     #$06          ; Add offset for letter.
+
+ ECHO:
+                BIT     DSP           ; DA bit (B7) cleared yet?
+                BMI     ECHO          ; No, Wait for display.
+                STA     DSP           ; Output character. Sets DA.
+                RTS                   ; Return.
+
+   .org $FFFA
+
+                .word   $0F00         ; NMI vector
+                .word   RESET         ; RESET vector
+                .word   $0000         ; IRQ vector
```