silver surfer polling mode driver for cc65

```
;------------------------------------------------------------------------
------------------
; silver surfer polling mode driver for cc65
; - work from here to create a full featured driver with interupts.
; gpz fixed 20020828: fatal bug fixed in _ss232_params
;------------------------------------------------------------------------
------------------

.include "ports/silversurfer.inc"

                        .export _ss232_init
                        .export _ss232_done
                        .export _ss232_params
                        .export _ss232_put
                        .export _ss232_get

                        .importzp ptr1, ptr2
                        .import   popa, popax

;------------------------------------------------------------------------
------------------
; Error codes. Beware: The codes must match the codes in the C header file

ErrNotInitialized       = $01
ErrBaudTooFast          = $02
ErrBaudNotAvail         = $03
ErrNoData               = $04
ErrOverflow             = $05

;------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_init (char hacked);
;/* Initialize the serial port, install the interrupt handler. The parameter
; * has no effect for now and should be set to 0.
; */
;------------------------------------------------------------------------
------------------

         .code

_ss232_init:
         ; enable ssurfer-port
         lda $de01
         ora #$01
         sta $de01

         ; disable nmi's from ssurfer
```

```
         lda #%00000000
         sta fifo_ier

         ; activate dtr
         lda #%00000001
         sta fifo_mcr

         lda #$00       ; ok
         tax
         rts

;------------------------------------------------------------------------
-------------------
;unsigned char __fastcall__ rs232_done (void);
;/* Close the port, deinstall the interrupt hander. You MUST call this
function
; * before terminating the program, otherwise the machine may crash later.
If
; * in doubt, install an exit handler using atexit(). The function will do
; * nothing, if it was already called.
; */
;------------------------------------------------------------------------
-------------------

_ss232_done:
         ; disable nmi's from ssurfer
         lda #%00000000
         sta fifo_ier

         ; deactivate dtr
         sta fifo_mcr

         ; disable ssurfer-port
         lda $de01
         and #$fe
         sta $de01

         lda #$00       ; ok
         tax
         rts

;------------------------------------------------------------------------
-------------------
;unsigned char __fastcall__ rs232_params (unsigned char params, unsigned
char parity);
;/* Set the port parameters. Use a combination of the #defined values above.
*/
;------------------------------------------------------------------------
-------------------

         .data
```

```
_ss232_baudrates:

         .word          (7372800 / (      50 * 16))
         .word          (7372800 / (     110 * 16))
         .word          (7372800 / (     269 *  8))
         .word          (7372800 / (     300 * 16))
         .word          (7372800 / (     600 * 16))
         .word          (7372800 / (    1200 * 16))
         .word          (7372800 / (    2400 * 16))
         .word          (7372800 / (    4800 * 16))
         .word          (7372800 / (    9600 * 16))
         .word          (7372800 / (   19200 * 16))
         .word          (7372800 / (   38400 * 16))
         .word          (7372800 / (   57600 * 16))
         .word          (7372800 / (  115200 * 16))
         .word          (7372800 / (  230400 * 16))

         .bss

_ss232_tmp1:
         .res 1

         .code

_ss232_params:

         sta _ss232_tmp1 ; save parity

         ; reset fifo
         lda #%10000111
         sta fifo_fcr

   ; that delay thing really needed ?!
   ; (original datasheet mentions a delay here)
   ;      ldy #$00
   ;      dey
   ;      bny *-1

         ; set dlab
         lda #%10000011 ; we assmume 8n1
         sta fifo_lcr

         jsr popa
         tay             ; save param

         ; set baudrate
         clc
         lsr a
         lsr a
         lsr a
         lsr a
```

```
         asl a
         tax
         lda _ss232_baudrates,x
         sta fifo_dll
         lda _ss232_baudrates+1,x
         sta fifo_dlm

         tya             ; param
         and #$0f
         ora _ss232_tmp1 ; parity

         ; reset dlab
         sta fifo_lcr

         lda #$00       ; ok
         tax
         rts

;-------------------------------------------------------------------------
------------------
; check if byte available, returns AKKU=0 if none

ss_getlsr:
         lda fifo_lsr
         and #$01
         rts

;-------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_get (char* b);
;/* Get a character from the serial port. If no characters are available,
the
; * function will return RS_ERR_NO_DATA, so this is not a fatal error.
; */
;-------------------------------------------------------------------------
------------------
; get byte (non blocking, returns byte in A or CARRY=1 - error)

_ss232_get:
         sta ptr1
         stx ptr1+1

         jsr ss_getlsr  ; check if byte available
;         bne sk32 ; yes
         bne sk33 ; yes

         ; activate rts
         lda #%00000011
         sta fifo_mcr
sk32:
```

```
         ; deactivate rts
;         lda #%00000001
;         sta fifo_mcr

         jsr ss_getlsr  ; check if byte available
         bne sk33 ; yes

         ; deactivate rts
         lda #%00000001
         sta fifo_mcr

         lda #ErrNoData      ; no data
         ldx #0
         rts
sk33:
         ; deactivate rts
         lda #%00000001
         sta fifo_mcr

         ; get byte
         ldy #$00
         lda fifo_rxd
         sta (ptr1),y

         lda #0      ; ok
         tax
         rts

;------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_put (char b);
;/* Send a character via the serial port. There is a transmit buffer, but
; * transmitting is not done via interrupt. The function returns
; * RS_ERR_OVERFLOW if there is no space left in the transmit buffer.
; */
;------------------------------------------------------------------------
------------------

_ss232_put:
         tax
         ; transmit buf ready?
         lda fifo_lsr
         and #%00100000
         bne @sk1
@sk2:
         lda #ErrOverflow       ; overflow
         ldx #$00
         rts
@sk1:
         ; reciever ready?
         lda fifo_msr
```

```
         and #%00010000
         beq @sk2

         stx fifo_txd

         lda #$00               ; ok
         tax
         rts

;------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_pause (void);
;/* Assert flow control and disable interrupts. */
;------------------------------------------------------------------------
------------------

_ss232_pause:
         ; activate rts
         lda #%00000011
         sta fifo_mcr

         lda #$00       ; ok
         tax
         rts

;------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_unpause (void);
;/* Re-enable interrupts and release flow control */
;------------------------------------------------------------------------
------------------

_ss232_unpause:
         ; deactivate rts
         lda #%00000001
         sta fifo_mcr

         lda #$00       ; ok
         tax
         rts

;------------------------------------------------------------------------
------------------
;unsigned char __fastcall__ rs232_status (unsigned char* status,
;                                         unsigned char* errors);
;/* Return the serial port status. */
;------------------------------------------------------------------------
------------------

_ss232_status:
        sta     ptr2
```

```
        stx     ptr2+1
        jsr     popax
        sta     ptr1
        stx     ptr1+1

        ldy     #$00

        ; Get status
        lda     fifo_iir
        and     #%00000001
        sta     _ss232_tmp1
        lda     fifo_msr
        lsr     a
        and     #%01010000
        ora     _ss232_tmp1
        sta     _ss232_tmp1
        lda     fifo_lsr
        and     #%00101110
        ora     _ss232_tmp1
        sta     (ptr1),y

        ; Get errors
        lda     #$00    ; ok
        sta     (ptr2),y

        lda     #$00    ; ok
        tax
        rts
```

From:
https://codebase64.org/ - **Codebase 64 wiki**

Permanent link:
https://codebase64.org/doku.php?id=base:rs232silversurfer.s

Last update: **2015-04-17 04:33**