```
.include "8515def.inc"

; Warning: set up FUSE-Bits ! See Datasheet!

.def temp       = r16
.def address    = r21
.def value      = r22
.def mask       = r23           ; for BitModify
.def stack      = r24
.def stack2     = r25
; mcp2515 Instructions:
.equ WRITE      = 0b00000010
.equ READ       = 0b00000011
.equ RESET      = 0b11000000
.equ BITMODIFY  = 0b00000101

; mcp2515 addresses:
.equ RXB0D0     = 0b01100110
.equ RXB0D1     = 0b01100111
.equ CANINTF    = 0b00101100
.equ CANINTE    = 0b00101011
.equ CNF1       = 0b00101010
.equ CNF2       = 0b00101001
.equ CNF3       = 0b00101000
.equ BFPCTRL    = 0b00001100
.equ CANCTRL    = 0b00001111
.equ TEC        = 0b00011100
.equ REC        = 0b00011101
.equ EFLG       = 0b00101101

;_____________________________________________________________________
.org 0x0000
        rjmp    main

.org 0x0001
        rjmp    interrupt0                      ; Low on INT0 means "message received o
n RX0"
;_____________________________________________________________________



interrupt0:                                     ; Message on RX0B
        in      stack2,     SREG                ; save Status Register

        sbi     PortA,      0                   ; Receive-LED On

        ; Reading RXB0D0
        ldi     address,    RXB0D0
        rcall   getbyte
        rcall   serout                          ; Send received Byte to a PC

        ; clear RX0BF Flag:
        ldi     address,    CANINTF
        ldi     mask,       0b00000001
        ldi     value,      0b00000000
        rcall   modifybyte

        cbi     PortA,      0                   ; Receive-LED Off

        out     SREG,       stack2              ; restore Status Register

        reti

;_____________________________________________________________________
main:
        ; Stackpointer init
        ldi     temp,       LOW(RAMEND)
        out     SPL,        temp
        ldi     temp,       HIGH(RAMEND)
        out     SPH,        temp

        ; Port A output
```

```
        ldi     temp,       0xFF
        out     DDRA,       temp

        ; Port C output
        ldi     temp,       0xFF
        out     DDRC,       temp


        ; enable Interrups on INT0-Pin (PD2)
        ldi     temp,       (1<<ISC01)          ; INT0 = falling Edge
        out     MCUCR,      temp
        ldi     temp,       (1<<INT0)           ; INT0 INT enable
        out     GIMSK,      temp                ; General INT Control Register

        ; mcp Startup Delay, now SPI can be used with mcp2515
        rcall   wait_500ms

        ; now SPI can be used in mcp2515

        ; UART init, 9600 baud
        ldi     temp,       25
        out     UBRR,       temp
        sbi     UCR,        TXEN                ; TX aktivieren

        ; SPI Master Init
        ldi     temp,       0b10110000          ; Output = SCK & MOSI & /SS
        out     DDRB,       temp
        sbi     PortB,      4                   ; /CS  High

        ; SPIEnabled, MasterMode, SPI Clock Rate = OSC/128
        ;(0<<SPIE)|(1<<SPE)|(0<<DORD)|(1<<MSTR)|(0<<CPOL)|(0<<CPHA)|(1<<SPR1)|(1<<SPR0)
        ldi     temp,       0b01010011
        out     SPCR,       temp

        ; MCP2515 init
        rcall   mcp_reset
        rcall   wait_100ms                      ; a must be

; ==================== MCP2515 CONFIGURATION MODE ==========================

    ; MCP on Quarz, 4MHz, Can Speed 100kbps, 1 Bittime = 10 TQ

    ; CNF1:
        ldi     address,    CNF1
        ldi     value,      0x01
        rcall   sendbyte

    ; CNF2:
        ldi     address,    CNF2
        ldi     value,      0xA0
        rcall   sendbyte

    ; CNF3:
        ldi     address,    CNF3
        ldi     value,      0x02
        rcall   sendbyte

    ; INTERRUPTS
        ldi     address,    CANINTE
        ldi     value,      0b00000001          ; Receive Buffer INT Enable
        rcall   sendbyte

    ; PIN FUNCTIONS:
        ldi     address,    BFPCTRL
        ldi     value,      0b00000101          ; Use Pin B0BFM as Interupt, when messa
ge arrived
        rcall   sendbyte


; ==================== MCP2515 NORMAL MODE ==========================

    ; Normal Mode:
        ldi     address,    CANCTRL
```

```
        ldi     value,      0b00000000
        rcall   sendbyte

        sei                                 ; Global Interupt Enable


loop:   rjmp    loop                        ; Wait for Interupt









;..... Sub-Procedures:
;___________________________________________________________________
getbyte:
        in      stack,      SPCR
        push    stack
        nop
        nop
        cbi     PortB,      4               ; /SS low

        ; READ COMMAND
        ldi     temp,       READ
        out     SPDR,       temp
wait_spi_g1:
        sbis    SPSR,       SPIF            ; Transmission complete?
        rjmp    wait_spi_g1
        nop
        nop
        in      temp,       SPDR            ; release SPIF here

        ; SET ADDRESS
        out     SPDR,       address
wait_spi_g2:
        sbis    SPSR,       SPIF            ; Transmission complete?
        rjmp    wait_spi_g2
        nop
        nop
        in      temp,       SPDR            ; release SPIF here

        ; DUMMY BYTE
        ldi     temp,       0b10101010
        out     SPDR,       temp
wait_spi_g3:
        sbis    SPSR,       SPIF            ; Transmission complete?
        rjmp wait_spi_g3

        ; RESULT in temp:
        nop
        nop
        in      temp,       SPDR            ; release SPIF here

        sbi     PortB,      4               ; /SS high
        nop
        nop
        pop     stack                       ; restore SPCR
        out     SPCR,       stack
        ret
;___________________________________________________________________
modifybyte:
        in      stack,      SPCR
        push    stack
        nop
        nop
```

```
        cbi     PortB,      4                   ; /SS low

        ; BITMODIFY COMMAND
        ldi     temp,       BITMODIFY
        out     SPDR,       temp
wait_spi_b1:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_b1
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        ; SET ADDRESS
        out     SPDR,       address
wait_spi_b2:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_b2
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        ; MASK BYTE
        out     SPDR,       mask
wait_spi_b3:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_b3
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        ; BITS TO BE CHANGED
        out     SPDR,       value
wait_spi_b4:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_b4
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        sbi     PortB, 4                        ; /SS high
        nop
        nop
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret
;_____________________________________________________________________
mcp_reset:
        in      stack,      SPCR
        push    stack
        nop
        nop
        cbi     PortB,      4                   ; /SS low

        ; RESET-Instruction
        ldi     temp,       RESET
        out     SPDR,       temp
wait_spi_r:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_r
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        sbi     PortB,      4                   ; /SS high
        nop
        nop
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret
;_____________________________________________________________________
sendbyte:
        in      stack,      SPCR
        push    stack
```

```
        nop
        nop
        cbi     PortB,      4                   ; /SS low

        ; WRITE COMMAND
        ldi     temp,       WRITE
        out     SPDR,       temp
wait_spi_w1:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_w1
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        ; SET ADDRESS
        out     SPDR,       address
wait_spi_w2:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_w2
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        ; DATA BYTE
        out     SPDR,       value
wait_spi_w3:
        sbis    SPSR,       SPIF                ; Transmission complete?
        rjmp    wait_spi_w3
        nop
        nop
        in      temp,       SPDR                ; release SPIF here

        sbi     PortB,      4                   ; /SS high
        nop
        nop
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret

;_____________________________________________________________________
erroroutput:
        in      stack,      SPCR
        push    stack

        ldi     address,    TEC
        rcall   getbyte
        rcall   serout

        ldi     address,    REC
        rcall   getbyte
        rcall   serout

        ldi     address,    EFLG
        rcall   getbyte
        rcall   serout

        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret

;_____________________________________________________________________
wait_500ms:
        in      stack,      SPCR
        push    stack
;     2000000 Zyklen:
; ----------------------------
; warte 1999998 Zyklen:
          ldi   R17, $12
WGLOOP0v:  ldi      R18, $BC
WGLOOP1v:  ldi      R19, $C4
WGLOOP2v:  dec      R19
          brne  WGLOOP2v
          dec   R18
```

```
          brne  WGLOOP1v
          dec   R17
          brne  WGLOOP0v
; ----------------------------
; warte 2 Zyklen:
          nop
          nop
; ============================
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret


;_____________________________________________________________________________
wait_100ms:
        in      stack,      SPCR
        push    stack
;     400000 Zyklen:
; ----------------------------
; warte 399999 Zyklen:
          ldi   R17, $97
WGLOOP0s:  ldi  R18, $06
WGLOOP1s:  ldi  R19, $92
WGLOOP2s:  dec  R19
          brne  WGLOOP2s
          dec   R18
          brne  WGLOOP1s
          dec   R17
          brne  WGLOOP0s
; ----------------------------
; warte 1 Zyklus:
          nop
; ============================
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret

;_____________________________________________________________________________
wait_10ms:
        in      stack,      SPCR
        push    stack

;     40000 Zyklen:
; ----------------------------
; warte 39999 Zyklen:
          ldi  R17, $43
WGLOOP0:  ldi  R18, $C6
WGLOOP1:  dec  R18
          brne WGLOOP1
          dec  R17
          brne WGLOOP0
; ----------------------------
; warte 1 Zyklus:
          nop
; ============================
        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret

;_____________________________________________________________________________
serout:
        in      stack,      SPCR
        push    stack
wait_ser:
        sbis    USR,        UDRE                ; wait UDR
        rjmp    wait_ser
        out     UDR,        temp                ; SPI-Data Register to UDR (sending to
PC)

        pop     stack                           ; restore SPCR
        out     SPCR,       stack
        ret
```