```
;-----------------------------------------------------------------------------------------------------------------------------------------
; Title         :   RS232_Senden_Empfangen_Interrupt
;-----------------------------------------------------------------------------------------------------------------------------------------
; Funktion      :   Sendet die ASCII-Zeichen von 000 bis 255 an COM1 (RS232) des Computers.
;               :   Eine Taste an D.2 löst den Vorgang aus. Sie muß gedrückt und wieder losgelassen werden.
;               :   1 s nach dem Loslassen der Taste beginnt der Sendevorgang.
;               :   Die Ausgabe von Zeichen zu Zeichen erfolgt mit dazwischenliegender Wartezeit.
;               :   Die Länge der Wartezeit ist einstellbar von 10 ms bis 630 ms mittels Brücken nach GND in D.3 bis D.7.
;               :   Der Zeichen-Code wird mit LEDs an B.0 (LSB), B.1, B.2, B.3, B.4, B.5, C.0, C.1 (MSB) angezeigt.
;               :   LED an C.2: Der Sendevorgang läuft.
;               :   LED an C.3: Der Sendevorgang ist beendet.
;               :   LED an C.4: Zeigt das Senden eines Bytes an.
;               :
;               :   Zeichen, welche von einem PC mittels RS232 gesendet werden, werden empfangen.
;               :   Nur ASCII - Zeichen von 0 bis 255 sind erlaubt.
;               :   Der Zeichen-Code wird mit LEDs an B.0 (LSB), B.1, B.2, B.3, B.4, B.5, C.0, C.1 (MSB) angezeigt.
;               :   LED an C.5: Zeigt den Empfang eines Bytes an.
;               :
;               :   Reset-Taste: Abbruch der Programmausführung.
;               :
;               :   Das Visual Basic Programm "RS232 Send Receive.exe" kann die mit diesem Programm kommunizieren.
;               :   Es wurde geschrieben, um die RS232-Funktionen auf unterster Ebene darzustellen.
;               :
; Schaltung     :   Portplatine
;               :   Brücken nach LEDs:  B.0 (LSB)
;               :                       B.1
;               :                       B.2
;               :                       B.3
;               :                       B.4
;               :                       B.5
;               :                       C.0
;               :                       C.1 (MSB)
;               :                       C.2 (Zeichenausgabe läuft)
;               :                       C.3 (Zeichenausgabe beendet)
;               :                       C.4 (signalisiert die Ausgabe eines Bytes)
;               :                       C.5 (signalisiert den Empfang eines Bytes)
;               :
;               :   Taste   nach GND:   D.2 (Startet den Ausgabezyklus)
;               :   Brücken nach LED:   D.3 (Wartezeit 0.02 s) Die gesetzten Wartezeiten werden addiert
;               :                       D.4 (Wartezeit 0.04 s) Beispiel: 0.20 s = D.6 + D.4
;               :                       D.5 (Wartezeit 0.08 s)
;               :                       D.6 (Wartezeit 0.16 s)
;               :                       D.7 (Wartezeit 0.32 s)
;               :
;               :   Taste nach GND:     IC-Beinchen #1: Reset
;               :
;               :   MAX232:             IC    Pin 03 TxD --> MAX232 Pin 11   TTL 1 In
;               :                       IC    Pin 02 RxD --> MAX232 Pin 12   TTL 3 Out
;               :                       RS232 Pin 02 TxD --> MAX232 Pin 14 RS232 1 Out (grün)
;               :                       RS232 Pin 03 RxD --> MAX232 Pin 13 RS232 3 In  (gelb)
;               :                       RS232 Pin 05 GND --> MAX232 Pin 15 GND
;-----------------------------------------------------------------------------------------------------------------------------------------
; Prozessor     :   ATmega8
; Takt          :   3,6864 MHz
; Sprache       :   Assembler
; Datum         :   24.11.2011
```

```
; Version       :   2.3
; Autor         :   Klaus38
;-----------------------------------------------------------------------------------------------------------------------------------------------
.include "AVR.H"
.equ    Temp    ,   16                      ; EQUAL
.equ    Wait11  ,   17
.equ    Wait12  ,   18
.equ    Wait13  ,   19
.equ    Charac  ,   20
.equ    PWert1  ,   21
.equ    PWert2  ,   22
.equ    Time    ,   23
;-----------------------------------------------------------------------------------------------------------------------------------------------
; Reset and Interrupt vector                Syntax                                      Beschreibung

Begin:  rjmp    Main                        ; RJMP  - Relative Jump                   :  1 POWER ON RESET
        rjmp    onInt0                      ; RJMP  - Relative Jump                   :  2 Int0-Interrupt
        reti                                ; RETI  - Return from Interrupt           :  3 Int1-Interrupt
        reti                                ; RETI  - Return from Interrupt           :  4 TC2 Compare Match
        reti                                ; RETI  - Return from Interrupt           :  5 TC2 Overflow
        reti                                ; RETI  - Return from Interrupt           :  6 TC1 Capture
        reti                                ; RETI  - Return from Interrupt           :  7 TC1 Compare Match A
        reti                                ; RETI  - Return from Interrupt           :  8 TC1 Compare Match B
        reti                                ; RETI  - Return from Interrupt           :  9 TC1 Overflow
        reti                                ; RETI  - Return from Interrupt           : 10 TC0 Overflow
        reti                                ; RETI  - Return from Interrupt           : 11 SPI, STC Serial Transfer Complete
        rjmp    onChar                      ; RJMP  - Relative Jump                   : 12 UART Rx complete
        reti                                ; RETI  - Return from Interrupt           : 13 UART Data Register Empty
        reti                                ; RETI  - Return from Interrupt           : 14 UART Tx complete
        reti                                ; RETI  - Return from Interrupt           : 15 ADC Conversion Complete
        reti                                ; RETI  - Return from Interrupt           : 16 EEPROM Ready
        reti                                ; RETI  - Return from Interrupt           : 17 Analog Comparator
        reti                                ; RETI  - Return from Interrupt           : 18 TWI (I²C) Serial Interface
        reti                                ; RETI  - Return from Interrupt           : 19 Store Program Memory Redy
;-----------------------------------------------------------------------------------------------------------------------------------------------
; Start, Power ON, Reset

Main:
        ldi     Temp    ,   lo8(RAMEND)     ; LDI   - Load Immediate                  : für  Stackpointer LOW
        out     SPL     ,   Temp            ; OUT   - Output register to I/O port     : INIT Stackpointer LOW
                                            ; SPL   - Stackregister Low
        ldi     Temp    ,   hi8(RAMEND)     ; LDI   - Load Immediate                  : für  Stackpointer HIGH
        out     SPH     ,   Temp            ; OUT   - Output register to I/O port     : INIT Stackpointer HIGH
                                            ; SPH   - Stackregister High
        ldi     Temp    ,   0b00111111      ;                                         : B.0 bis B.5 = Output
        out     DDRB    ,   Temp            ;                                         :
        ldi     Temp    ,   0b00111111      ;                                         : C.0 bis C.5 = Output
        out     DDRC    ,   Temp            ;                                         :

        ldi     Temp    ,   0b00000000      ;                                         : D.2 bis D.7 = Input
        out     DDRD    ,   Temp            ;                                         :
        ldi     Temp    ,   0b11111100      ;                                         : D.2 bis D.7 = Pullup
        out     PORTD   ,   Temp            ;                                         :

        ldi     Temp    ,   0b00000000      ;                                         :
        out     UBRRH   ,   Temp            ; UBRRH - USART Baud Rate Registers High  :
```

```
        ldi     Temp    ,   0b00000101      ;                                       : Dez. 5 = 38400 Baud
        out     UBRRL   ,   Temp            ; UBRRL - USART Baud Rate Registers Low     :

        ldi     Temp    ,   0b10000110      ; URSEL = 1 Register Select                : Auslesen
                                            ; UCSZ1 = 1 Character Size                 : | 8 Bit |
                                            ; UCSZ0 = 1 Character Size                 : | 8 Bit |
        out     UCSRC   ,   Temp            ; UCSRC -  USART Control and Status Register C

        ldi     Temp    ,   0b10011000      ; RXCIE = 1 RX Interrupt Enable            : Interrupt nach Empfange eines Bytes
                                            ; RXEN  = 1 RX aktivieren                  : Empfangen
                                            ; TXEN  = 1 TX aktivieren                  : Senden
        out     UCSRB   ,   Temp            ; UCSRB - USART Control and Status Register B

        ldi     Temp    ,   0b01000000      ;                                       ;
        out     GICR    ,   Temp            ; GICR - General Interrupt Mask Register    ; Bit 6 = INT0 aktiv

        ldi     Temp    ,   0b00000000      ;                                       : PORTB und PORTC: Alle Bits = 0
        out     PORTB   ,   Temp            ;                                       : LED   an Bit 0, 1, 2, 3, 4, 5
        out     PORTC   ,   Temp            ;                                       : LED   an Bit 0, 1, 2, 3, 4, 5

        ldi     Temp    ,   0               ; R16                                   :
        ldi     Wait11  ,   0               ; R17                                   :
        ldi     Wait12  ,   0               ; R18                                   :
        ldi     Wait13  ,   0               ; R19                                   :
        ldi     Charac  ,   0               ; R20                                   :
        ldi     PWert1  ,   0               ; R21                                   :
        ldi     PWert2  ,   0               ; R22                                   :
        ldi     Time    ,   0               ; R23                                   :

        sei
;-----------------------------------------------------------------------------------------------------------------------------------------------
; Hauptschleife

Mainloop:
        wdr                                 ; WATCHDOG RESET
        rjmp    Mainloop

;-----------------------------------------------------------------------------------------------------------------------------------------------
; Die Taste an D.2 wird geprüft, ob sie gedrückt und dann wieder losgelassen wurde.
; Nach 1 s wird die Subroutine "OutChar" aufgerufen zur Ausgabe von "000" bis "255" an TxD.

onInt0:
        push    Temp
        push    Wait11

        cli                                 ; CLI   - Clear Global Interrupt flag       : weiteren Interrupt verbieten

        ldi     Wait11  ,   0               ; LDI   - Load Immediate                    : Wait11 = 00000000
L1:
        ldi     Time    ,   0b00000010      ;                                       ; Dez.: 2 = 0.02 s
        rcall   Wait                        ; RCALL - Relative Call to Subroutine       : Wartezeit 0.02 s (entprellen)
        in      Wait11  ,   PIND            ; IN    - Load an I/O Port to Register      : D.2 Taste gedrückt ?
        sbr     Wait11  ,   0b11111011      ; SBR   - Set bits in Register              : Wait11 = 11111011
        com     Wait11                      ; COM   - One's Complement                  : Wait11 = 00000100
        ror     Wait11                      ; ROR   - Rotate Right through Carry        : Wait11 = X0000010
```

```
        ror     Wait11                  ; ROR   - Rotate Right through Carry       : Wait11 = XX000001
        cbr     Wait11  ,   0b11111110  ; CBR   - Clear Bits in Register          : Wait11 = 00000001
        cpi     Wait11  ,   0b00000001  ; CPI   - Compare with Immediate          : ist Wait11 = 00000001 = Taste an D.2 gedrückt ?
        breq    L2                      ; BREQ  - Branch if Equal                 : wenn ja,   springe nach L2:
        rjmp    L1                      ; RJMP  - Relative Jump                   : wenn nein, springe nach L1:
L2:
        ldi     Time    ,   0b00000010  ;                                         ; Dez.: 2 = 0.02 s
        rcall   Wait                    ; RCALL - Relative Call to Subroutine     : Wartezeit 0.02 s (entprellen)
        in      Wait11  ,   PIND        ; IN    - Load an I/O Port to Register    : D.2 Taste nicht mehr gedrückt ?
        sbr     Wait11  ,   0b11111011  ; SBR   - Set bits in Register            : Wait11 = 11111111
        com     Wait11                  ; COM   - One's Complement                : Wait11 = 00000000
        ror     Wait11                  ; ROR   - Rotate Right through Carry      : Wait11 = X0000000
        ror     Wait11                  ; ROR   - Rotate Right through Carry      : Wait11 = XX000000
        cbr     Wait11  ,   0b11111110  ; CBR   - Clear Bits in Register          : Wait11 = 00000000
        cpi     Wait11  ,   0b00000000  ; CPI   - Compare with Immediate          : ist Wait11 = 00000000 = Taste an D.2 nicht gedrückt ?
        breq    L3                      ; BREQ  - Branch if Equal                 : wenn ja,   springe nach L3:
        rjmp    L2                      ; RJMP  - Relative Jump                   : wenn nein, springe nach L2:
L3:
        ldi     Temp    ,   0           ; LDI   - Load Immediate                  : Temp   = 00000000
        out     PORTB   ,   Temp        ; OUT   - Output register to I/O port     : B.0 bis B.7 = 0
        out     PORTC   ,   Temp        ; OUT   - Output register to I/O port     : C.0 bis C.7 = 0
        ldi     Time    ,   0b01100100  ;                                         : Dez.: 100 = 1.00 s
        rcall   Wait                    ; RCALL - Relative Call to Subroutine     : springe zur Sub "Wait:"
        rcall   OutChar                 ; RCALL - Relative Call to Subroutine     : springe zur Sub "OutChar:"

        sei                             ; SEI   - Set Global Interrupt Flag       : Interrupts zulassen

        pop     Wait11
        pop     Temp

        reti                            ; RETI  - Return from Interrupt           :

;-----------------------------------------------------------------------------------------------------------------------------------------
; Zeichen-Generierung. Charac = 000 bis 255 zur Ausgabe an TxD und zur LED-Anzeige an B.0 (LSB) bis B.5, C.0 und C.1 (MSB)

OutChar:
        cli                             ; CLI   - Clear Global Interrupt flag     : weiteren Interrupt verbieten

        ldi     Charac  ,   0           ; LDI   - Load Immediate                  :
        rcall   putChar                 ; RCALL - Relative Call to Subroutine     : springe zur Sub putChar
Loop4:
                                        ;                                         ; Beispiel: Charac = 205
        inc     Charac                  ; INC   - Increment                       : Charac = Charac + 1 = 206
        mov     PWert1  ,   Charac      ; MOV   - Copy Register                   : PWert1 = 11001110
        cbr     PWert1  ,   0b11000000  ; CBR   - Clear Bits in Register          : PWert1 = 00001110
        mov     PWert2  ,   Charac      ; MOV   - Copy Register                   : PWert2 = 11001110
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = X1100111
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = XX110011
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = XXX11001
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = XXXX1100
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = XXXXX110
        ror     PWert2                  ; ROR   - Rotate Right through Carry      : PWert2 = XXXXXX11
        cbr     PWert2  ,   0b11111100  ; CBR   - Clear Bits in Register          : PWert2 = 00000011
        out     PORTB   ,   PWert1      ; OUT   - Output register to I/O port     : PORTB  = --001110
        out     PORTC   ,   PWert2      ; OUT   - Output register to I/O port     : PORTC  = --000011
        sbi     PORTC   ,   2           ; SBI   - Set bits in I/O Register        : C.2    = 1
```

```
        cbi     PORTC   ,   3               ; CBI   - Clear Bit in I/O Register        : C.3    = 0
        rcall   putChar                     ; RCALL - Relative Call to Subroutine      : springe zur Sub putChar: Ausgabe = Carac
        cpi     Charac  ,   255             ; CPI   - Compare with Immediate           : ist Charac = 255 (Maximalwert) ?
        breq    Loop5                       ; BREQ  - Branch if Equal                  : wenn ja  : springe nach Loop5:
        rjmp    Loop4                       ; RJMP  - Relative Jump                    : wenn nein: springe nach Loop4:
Loop5:
        rcall   putChar                     ; RCALL - Relative Call to Subroutine      : springe zur Sub putChar: Ausgabe = Carac = 255
        rcall   getTime                     ; RCALL - Relative Call to Subroutine      : springe zur Sub getTime: Ablesen von D.3 bis D.7
        rcall   Wait                        ; RCALL - Relative Call to Subroutine      : springe zur Sub Wait:
        ldi     Charac  ,   0               ; LDI   - Load Immediate                   : Charac = 0
        out     PORTB   ,   Charac          ; OUT   - Output register to I/O port      : PORTB  = --000000
        cbi     PORTC   ,   0               ; CBI   - Clear Bit in I/O Register        : C.0    = 0
        cbi     PORTC   ,   1               ; CBI   - Clear Bit in I/O Register        : C.1    = 0
        cbi     PORTC   ,   2               ; CBI   - Clear Bit in I/O Register        : C.2    = 0
        sbi     PORTC   ,   3               ; SBI   - Set Bit in I/O Register          : C.3    = 1

        ret                                 ; RET   - Return from Subroutine           :

;-------------------------------------------------------------------------------------------------------------------------------------------
; Senden der Zeichen

putChar:
        cli                                 ; CLI   - Clear Global Interrupt flag      : weiteren Interrupt verbieten
        ldi     Time    ,   0b00000000
        sbis    UCSRA   ,   5               ; SBIS  - Skip if bit in I/O register is Set: warten, bis UDR leer
                                            ; UCSRA - Control and Status Register A    :
        rjmp    putChar                     ; RJMP  - Relative Jump                    : springe nach putChar
        out     UDR     ,   Charac          ; OUT   - Output register to I/O port      : Charac = 000 bis 255
                                            ; UDR   - I/O Data Register               :
        sbi     PORTC   ,   4               ;                                          : C.4: Byte wird gesendet EIN
        rcall   getTime                     ; RCALL - Relative Call to Subroutine      : springe zur Sub getTime: Ablesen von D.3 bis D.7
        rcall   Wait                        ; RCALL - Relative Call to Subroutine      : springe zur Sub Wait
        cbi     PORTC   ,   4               ;                                          : C.4: Byte wird gesendet AUS
        rcall   getTime                     ; RCALL - Relative Call to Subroutine      : springe zur Sub getTime: Ablesen von D.3 bis D.7
        rcall   Wait                        ; RCALL - Relative Call to Subroutine      : springe zur Sub Wait

        ret                                 ; RET   - Return from Subroutine           :

;-------------------------------------------------------------------------------------------------------------------------------------------
; Empfangen der Zeichen

onChar:
        cli                                 ; CLI   - Clear Global Interrupt flag      : weiteren Interrupt verbieten

        sbi     PORTC   ,   5               ; SBI   - Set bits in I/O Register         : zeigt den Empfang eines Bytes an
        sbis    UCSRA   ,   0b00000111      ; SBIS  - Skip if bit in I/O register is Set
                                            ; Bit 7 – RXC: USART Receive Complete
        rjmp    onChar                      ; warten bis 8 Bit angekommen sind
        rcall   getTime                     ; RCALL - Relative Call to Subroutine      : springe zur Sub getTime: Ablesen von D.3 bis D.7
        rcall   Wait                        ; RCALL - Relative Call to Subroutine      : springe zur Sub Wait
        in      PWert1  ,   UDR             ; UDR   - UART I/O Data Register
        out     PORTB   ,   PWert1          ; alle Bits in PORTB  = 0
        mov     PWert2  ,   PWert1          ; MOV   - Copy Register
                                            ; PWert1 = NNNNNNNN (N = Platzhalter für 0 oder 1)
                                            ; PWert2 = NNNNNNNN (N = Platzhalter für 0 oder 1)
        cbr     PWert1  ,   0b11000000      ; PWert1 = 00NNNNNN (N = Platzhalter für 0 oder 1)
```

```
        out     PORTB   ,   PWert1
        ror     PWert2                  ; PWert2 = xNNxxxxx (N = Platzhalter für 0 oder 1)
        ror     PWert2                  ; PWert2 = xxNNxxxx (x = egal, ob da 0 oder 1 drin ist)
        ror     PWert2                  ; PWert2 = xxxNNxxx
        ror     PWert2                  ; PWert2 = xxxxNNxx
        ror     PWert2                  ; PWert2 = xxxxxNNx
        ror     PWert2                  ; PWert2 = xxxxxxNN
        cbr     PWert2  ,   0b11111100  ; PWert2 = 000000NN
        out     PORTC   ,   PWert2      ;                                       : hier wird auch C.5 gelöscht

        reti                            ; RETI - Return from Interrupt

;-----------------------------------------------------------------------------------------------------------------------------------
; Wartezeit: s. Programm "Sleep_10__2550_ms.sax"

Wait:
        cli                             ; CLI   - Clear Global Interrupt flag       : weiteren Interrupt verbieten

        push    Wait12
        push    Wait13

        cpi     Time    ,   0
        breq    Loop0
Loop1:
        ldi     Wait12  ,   110
Loop2:
        ldi     Wait13  ,   110
Loop3:
        dec     Wait13
        brne    Loop3
        nop
        nop
        dec     Wait12
        brne    Loop2
        dec     Time
        brne    Loop1
Loop0:
        pop     Wait13
        pop     Wait12

        ret                             ; RET   - Return from Subroutine

;-----------------------------------------------------------------------------------------------------------------------------------
; Wartezeit: Kodierung durch Brücken in D.3 bis D.7 abfragen

getTime:
        cli                             ; CLI   - Clear Global Interrupt flag       : weiteren Interrupt verbieten

        push    Temp

        ldi     Time    ,   1           ; LDI   - Load Immediate                    :  10 ms
        in      Temp    ,   PIND        ; IN    - Load an I/O Port to Register      :
        sbrs    Temp    ,   3           ; SBRS  - Skip if bit in Register is Set    :
        sbr     Time    ,   0b00000010  ; SBR   - Set bits in Register              :  20 ms
        sbrs    Temp    ,   4           ; SBRS  - Skip if bit in Register is Set    :
        sbr     Time    ,   0b00000100  ; SBR   - Set bits in Register              :  40 ms
```

```
        sbrs    Temp    ,   5               ; SBRS  - Skip if bit in Register is Set    :
        sbr     Time    ,   0b00000110      ; SBR   - Set bits in Register              :  80 ms
        sbrs    Temp    ,   6               ; SBRS  - Skip if bit in Register is Set    :
        sbr     Time    ,   0b00010000      ; SBR   - Set bits in Register              : 160 ms
        sbrs    Temp    ,   7               ; SBRS  - Skip if bit in Register is Set    :
        sbr     Time    ,   0b00100000      ; SBR   - Set bits in Register              : 320 ms

        pop     Temp

        ret

;-----------------------------------------------------------------------------------------------------------------------------------
```