```
;**************************************************************************

; LCD-Unterprogramme (erste Beispiele)

;**************************************************************************

.DEVICE                 AT90S4414

;***** I/O Register Definitions
.equ     SREG  =$3f
.equ     SPH     =$3e
.equ     SPL     =$3d
.equ     GIMSK          =$3b
.equ     TIMSK          =$39
.equ     TIFR   =$38
.equ     MCUCR          =$35
.equ     TCCR0          =$33
.equ     TCNT0          =$32
.equ     TCCR1A         =$2f
.equ     TCCR1B         =$2e
.equ     TCNT1H         =$2d
.equ     TCNT1L         =$2c
.equ     OCR1AH         =$2b
.equ     OCR1AL         =$2a
.equ     OCR1BH         =$29
.equ     OCR1BL         =$28
.equ     ICR1H          =$25
.equ     ICR1L          =$24
.equ     WDTCR          =$21
.equ     EEARL          =$1e
.equ     EEDR  =$1d
.equ     EECR  =$1c
.equ     PORTA          =$1b
.equ     DDRA           =$1a
.equ     PINA   =$19
.equ     PORTB          =$18
.equ     DDRB           =$17
.equ     PINB   =$16
.equ     PORTC          =$15
.equ     DDRC           =$14
.equ     PINC   =$13
.equ     PORTD          =$12
.equ     DDRD           =$11
.equ     PIND   =$10
.equ     SPDR  =$0f
.equ     SPSR  =$0e
.equ     SPCR  =$0d
.equ     UDR    =$0c
```

```
.equ	USR	=$0b
.equ	UCR	=$0a
.equ	UBRR	=$09
.equ	ACSR	=$08


;****************** Dedicated Ports *****************


.equ	CTLO	=PORTA
.equ	CTLIN		=PINA
.equ	CTLDR		=DDRA
.equ	DATO		=PORTC
.equ	DATIN		=PINC
.equ	DATDR		=DDRC


;***** Bit Definitions
.equ	INT1	=7
.equ	INT0	=6

.equ	TOIE1		=7
.equ	OCIE1A		=6
.equ	OCIE1B		=5
.equ	TICIE	=3
.equ	TOIE0		=1

.equ	TOV1	=7
.equ	OCF1A		=6
.equ	OCF1B		=5
.equ	ICF1	=3
.equ	TOV0	=1

.equ	SRE	=7
.equ	SRW	=6
.equ	SE	=5
.equ	SM	=4
.equ	ISC11	=3
.equ	ISC10	=2
.equ	ISC01	=1
.equ	ISC00	=0

.equ	CS02	=2
.equ	CS01	=1
.equ	CS00	=0

.equ	COM1A1		=7
.equ	COM1A0		=6
.equ	COM1B1		=5
```

```
.equ	COM1B0		=4
.equ	PWM11		=1
.equ	PWM10		=0

.equ	ICNC1		=7
.equ	ICES1	=6
.equ	CTC1	=3
.equ	CS12	=2
.equ	CS11	=1
.equ	CS10	=0

.equ	WDE	=3
.equ	WDP2	=2
.equ	WDP1	=1
.equ	WDP0	=0

.equ	EEWE		=1
.equ	EERE	=0

.equ	PA7	=7
.equ	PA6	=6
.equ	PA5	=5
.equ	PA4	=4
.equ	PA3	=3
.equ	PA2	=2
.equ	PA1	=1
.equ	PA0	=0

.equ	DDA7	=7
.equ	DDA6	=6
.equ	DDA5	=5
.equ	DDA4	=4
.equ	DDA3	=3
.equ	DDA2	=2
.equ	DDA1	=1
.equ	DDA0	=0

.equ	PINA7		=7
.equ	PINA6		=6
.equ	PINA5		=5
.equ	PINA4		=4
.equ	PINA3		=3
.equ	PINA2		=2
.equ	PINA1		=1
.equ	PINA0		=0

.equ	PB7	=7
.equ	PB6	=6
.equ	PB5	=5
```

```
.equ	PB4	=4
.equ	PB3	=3
.equ	PB2	=2
.equ	PB1	=1
.equ	PB0	=0

.equ	DDB7	=7
.equ	DDB6	=6
.equ	DDB5	=5
.equ	DDB4	=4
.equ	DDB3	=3
.equ	DDB2	=2
.equ	DDB1	=1
.equ	DDB0	=0

.equ	PINB7		=7
.equ	PINB6		=6
.equ	PINB5		=5
.equ	PINB4		=4
.equ	PINB3		=3
.equ	PINB2		=2
.equ	PINB1		=1
.equ	PINB0		=0

.equ	PC7	=7
.equ	PC6	=6
.equ	PC5	=5
.equ	PC4	=4
.equ	PC3	=3
.equ	PC2	=2
.equ	PC1	=1
.equ	PC0	=0

.equ	DDC7	=7
.equ	DDC6	=6
.equ	DDC5	=5
.equ	DDC4	=4
.equ	DDC3	=3
.equ	DDC2	=2
.equ	DDC1	=1
.equ	DDC0	=0

.equ	PINC7		=7
.equ	PINC6		=6
.equ	PINC5		=5
.equ	PINC4		=4
.equ	PINC3		=3
.equ	PINC2		=2
.equ	PINC1		=1
```

```
.equ	PINC0		=0

.equ	PD6	=6
.equ	PD5	=5
.equ	PD4	=4
.equ	PD3	=3
.equ	PD2	=2
.equ	PD1	=1
.equ	PD0	=0

.equ	DDD6	=6
.equ	DDD5	=5
.equ	DDD4	=4
.equ	DDD3	=3
.equ	DDD2	=2
.equ	DDD1	=1
.equ	DDD0	=0

.equ	PIND6		=6
.equ	PIND5		=5
.equ	PIND4		=4
.equ	PIND3		=3
.equ	PIND2		=2
.equ	PIND1		=1
.equ	PIND0		=0

.equ	SPIF	=7
.equ	WCOL		=6

.equ	SPIE	=7
.equ	SPE	=6
.equ	DORD		=5
.equ	MSTR	=4
.equ	CPOL	=3
.equ	CPHA	=2
.equ	SPR1	=1
.equ	SPR0	=0

.equ	RXC	=7
.equ	TXC	=6
.equ	UDRE	=5
.equ	FE	=4
.equ	OR	=3

.equ	RXCIE		=7
.equ	TXCIE		=6
.equ	UDRIE		=5
.equ	RXEN	=4
.equ	TXEN	=3
```

```
.equ    CHR9  =2
.equ    RXB8  =1
.equ    TXB8  =0

.equ    ACD   =7
.equ    ACO   =5
.equ    ACI   =4
.equ    ACIE  =3
.equ    ACIC  =2
.equ    ACIS1          =1
.equ    ACIS0          =0

.def    XL     =r26
.def    XH     =r27
.def    YL     =r28
.def    YH     =r29
.def    ZL     =r30
.def    ZH     =r31

.def    temp   =r16            ;temporary storage variable .def        lopco1 =r17
;loop counter 1
.def    ioreg  =r18            ;io buffer register
.def    char   =r19            ;current character
.def    ckc    =r20            ;checkpoint pulses
.def    timr1  =r21            ;timer compare value 1
.def    timr2  =r22

.def    lopco2 =r21

.equ    RAMBEG       =$60            ;1st SRAM Byte
.equ   RAMEND =$20+$40+$ff     ;Adjust for registers and I/O (Value = 15F)
.equ    INT0addr=$001          ;External Interrupt0 Vector Address
.equ    INT1addr=$002          ;External Interrupt1 Vector Address
.equ    ICP1addr=$003          ;Input Capture1 Interrupt Vector Address
.equ    OC1Aaddr=$004          ;Output Compare1A Interrupt Vector Address
.equ    OC1Baddr=$005          ;Output Compare1B Interrupt Vector Address
.equ    OVF1addr=$006          ;Overflow1 Interrupt Vector Address
.equ    OC0addr =$007          ;Output Compare0 Interrupt Vector Address
.equ    OVF0addr=$008          ;Overflow0 Interrupt Vector Address
.equ    SPIaddr =$009          ;SPI Interrupt Vector Address
.equ    URXCaddr=$00a          ;UART Receive Complete Interrupt Vector Address
.equ    UDREaddr=$00b          ;UART Data Register Empty Interrupt Vector Address
.equ    UTXCaddr=$00c          ;UART Transmit Complete Interrupt Vector Address
.equ    ACIaddr =$00d          ;Analog Comparator Interrupt Vector Address


; ****** LCD specific Equates *******
```

```
.equ    linlg   =$10    ; line Length of LCD (= 16)
.equ    line1   =$80    ; 1st char pos. in line 1
.equ    line2   =$c0    ; 1st char pos. in line 2


.equ    rs      =0
.equ    rw      =1
.equ    ena     =2
.equ    key     =3      ; Taste
.equ    busy    =7
.equ    ckp     =7      ; Checkpoint Pin


.equ    lcdl    =RAMBEG             ;low LCD digit - right position
.equ    lcdh    =RAMBEG + 1         ;high LCD digit - left position

.equ    rel1    =4
.equ    rel2    =5

.equ    switch  =0
.equ    key1    =1
.equ    key2    =2

.CSEG

.ORG 0x0000

        rjmp start

        rjmp intr       ; INT0
        rjmp intr       ; INT1
        rjmp intr       ; Timer 1 Capture
        rjmp intr       ; Timer 1 Compare Match A
        rjmp intr       ; Timer 1 Compare Match B
        rjmp intr       ; Timer 1 Overflow
        rjmp intr       ; Timer 0 Overflow
        rjmp intr       ; SPI Serial Transfer Complete
        rjmp intr       ; UART RX Complete
        rjmp intr       ; UART Data Register Empty
        rjmp intr       ; UART TX Complete
        rjmp intr       ; Analog Comparator



; ********************* Constants in Program Memory ********************

inemit:
        .db 0b00111000, 0b00001111, 0b00000001, 0b00000110, $80, $00
; LCD initialization data
```

```
hexemit:
        .db $30,$31,$32,$33,$34,$35,$36,$37,$38,$39,$41,$42,$43,$44,$45,$46 ;hex
characters

segemit:
        .db 0b00111111, 0b00000110          ;0, 1

        .db 0b01011011,       0b01001111    ;2, 3

        .db 0b01100110,       0b01101101    ;4, 5

        .db 0b01111101, 0b00000111          ;6, 7

        .db 0b01111111, 0b01101111          ;8, 9

        .db 0b01110111, 0b01111100          ;A, b

        .db 0b00111001,       0b01011000    ;C, c

        .db 0b01011110,       0b01111001    ;d, E

        .db 0b01110001, 0b01110110          ;F, H

        .db 0b01110100, 0b00111000          ;H, L

        .db 0b01010100, 0b01110011          ;n, P

        .db 0b01010000, 0b01000000          ;r, -



consta:         .db $48, $45, $4c, $4c, $4f, $00



;

intr:


;********************** Begin of Main Program *************************
start:
        ldi       temp,low(RAMEND)
        out     SPL,temp        ;init Stack Pointer Low
        ldi     temp,high(RAMEND)
      out     SPH, temp      ;init Stack Pointer High
```

```
;                       initialize IO


            ldi     temp,0                  ; initialize all ports
            out     CTLDR,temp              ; Ports to input
            out     DATDR,temp
            out     PORTA,temp
            out     PORTB,temp
            out     PORTC,temp
            out     PORTD,temp
            ldi     temp,$ff
            out     DDRD,temp
            ldi     temp,0b00110111
            out     CTLDR,temp              ; Pins 5, 4, 2, 1, 0 Port A => output
            ldi     temp,0b11001000                 ; activate Pullups at Port A input pins
            out     CTLO,temp
            ldi     temp,0b00000000                 ; Port B  -> input
            out     DDRB,temp
            ldi     temp,0b11111111                 ; Pullups on
            out     PORTB,temp



; initialize



            rcall   ready           ; LCD must be ready

            ldi     lopco1,5
            ldi     ZL,low(inemit)
            ldi     ZH,high(inemit)
            add     ZL,ZL
            adc     ZH,ZH

inill:      lpm                     : LCD initialization data (inemit) will be transferred
            mov     ioreg,r0
            rcall   wrictl
            adiw    ZL,1
            dec     lopco1
            brne    inill


            ldi     ioreg,line1     ; Example of an initial display
            rcall   wrictl

            ldi     ioreg,$b0       ;fill with -------
            rcall   fill
```

```
        ldi     ioreg,line2
        rcall   wrictl

        ldi     ioreg,$b0
        rcall   fill

;Begin of main program (useful work)


; ***************************** Subroutines ***************************


disro:  push    ioreg                   ; Displays a single hex character in line 1
        ldi     ioreg,line1
        rcall   wrictl
        rcall   clearl
        ldi     ioreg,line1
        rcall   wrictl
        pop     ioreg
        rcall   hexco
        ret


wridat:                                         ; write data byte
        sbi     CTLO,rs
        rjmp    wriby

wrictl:
        cbi     CTLO,rs

wriby:
        cbi     CTLO,rw                         ; write mode
        ldi     temp,0
        out     DATO,temp               ; pullups off
        ldi     temp,$ff
        out     DATDR,temp              ; data port to output
        out     DATO,ioreg              ; data output
        sbi     CTLO,ena                ; pulse enable line
        rcall   del
        cbi     CTLO,ena
        ldi     temp,$00
        out     DATDR,temp              ; data port to input
        ldi     temp,$ff
        out     DATO,temp               ; activate pullups
        sbi     CTLO,rw                         ; read mode

ready:                                  ; wait until BUSY = Low
        cbi     CTLO,rs
        sbi     CTLO,rw
```

```
waitl:  cbi     CTLO,ena
        sbi     CTLO,ena                ; enable
        rcall   del
        sbic    DATIN,busy
        rjmp    waitl
        cbi     CTLO,ena
        ret


clearl:                                 ; clear line
        ldi     ioreg,$20               ; blank char

fill:                           ; fill line. character in ioreg. line has to be selected in lcd
        ldi     lopco1,linlg            ; character count
fillo:  rcall   wridat                  ; write character       dec     lopco1
        brne    fillo
        ret


hexca:                          ; display Byte in hex. Byte in char
        push    ZH
        push    ZL
        mov     ioreg,char
        swap    ioreg
        rcall   hexco
        mov     ioreg,char
        rcall   hexco
        pop     ZL
        pop     ZH
        ret



hexco:                          ; convert a 4-bit nibble to hex and display. Nibble in ioreg

        andi    ioreg,$0f
        ldi     ZL,low(hexemit)
        ldi     ZH,high(hexemit)
        add     ZL,ZL
        adc     ZH,ZH
        add     ZL,ioreg
        ldi     ioreg,0
        adc     ZH,ioreg
        lpm
        mov     ioreg,r0
        rcall   wridat
        ret
```

```
delay:                  ;debouncing loop. uses ckc for counting
        ldi     ckc,0
dell:   dec     ckc
        brne    dell
dela:   dec     ckc
        brne    dela
delb:   dec     ckc
        brne    delb
        ret


stripg:                 ; display text string out of PGM memory. String Adrs in Z-register
                        ; string is zero-terminated (C-style)
        lpm
        mov     ioreg,r0
        cpi     ioreg,0             ; zero string termination
        breq    stre1
        rcall   wridat
        adiw    ZL,1
        rjmp    stripg
strend: ret

stre1:  rcall   wridat
        ret

hexpg:                  ; display hex string out of PGM memory. String Adrs in Z-register
                        ; String length in lopco1
        lpm
        mov     char,r0
        adiw    ZL,1
        rcall   hexca
        dec     lopco1
        brne    hexpg
        ret


del:                    ; short delay to ensure proper pulse width
        ldi     temp,5
del1:   dec     temp
        brne    del1
        ret


ad16:
        add     xl,temp
        ldi     temp,0
        adc     xh,temp
        ret
```