```
;Spannungsmessung mit Dot matrix SPI LCD Anzeige
;Maximale Eingangsspannung Uemax=20V
;Maximale ADC Eingangsspaqnnung Uemax/4
;RA0 Analog Eingang
;Fosc=4MHz /Instruction cyrce=1µS (Osc <500kHz d.h. XT_OSC)
;Lasaros Goumas: April 2020
;*****************************************************************

;Assembler directives
                list,P16F627
                #include <p16F876A.inc>         ;Controller header file
                __CONFIG _CP_OFF & _PWRTE_OFF & _WDT_OFF & _XT_OSC  & _LVP_OFF
                ERRORLEVEL      -302            ;Supress BanK SELECTION MESSAGES
;*****************************************************************

;Variablen

w_copy          equ 0x20                ;saved w register
s_copy          equ 0x21        ;saved STATUS register
highbyte        equ 0x22        ;Obere 2 Bits von ADC
lowbyte         equ 0X23        ;Untere 8 Bits von ADC
high_copy       equ 0x24
low_copy        equ 0x25        ;High and low copys´from ADC
highbytex25     equ 0x26
lowbytex25      equ 0x27        ;ADC Wert mal 25
highx25_64      equ 0x28
lowx25_64       equ 0x29        ;DC Wert mal 25/64
highx25_256     equ 0x2A
lowx25_256      equ 0x2B        ;DC Wert mal 25/256
highv_hund      equ 0x2C
lowv_hund       equ 0x2D        ;Gemessener Spannungswert mal 100
highv           equ 0x2E
lowv            equ 0x2F        ;Anzuzeigende Spannung (Gemessen Spannung*4)

highsub1        equ 0x30
lowsub1         equ 0x31
highsub2        equ 0x32
lowsub2         equ 0x33        ;Subtrahend
disp_taus       equ 0x34        ;Wertigkeit 4
disp_hund       equ 0x35        ;Wertigkeit 3
disp_zehn       equ 0x36        ;Wertigkeit 2
disp_einer      equ 0x37        ;Wertigkeit 1
disp_store      equ 0x38        ;Display Daten Zwischenspeicher
counter1        equ 0x39        ;Allgemeines zählregister
counter2        equ 0x3A        ;Allgemeines zählregister
;*****************************************************************

;definitionen

#define DISP_CSB        PORTA, 4        ;Display Aktivierung
#define DISP_CLK        PORTB, 6        ;Display clock
#define DISP_SI         PORTB, 7        ;Display Datenleitung
```

```
#define DISP_RS         PORTA, 3        ;Command/Data selector
#define DISP_LED        PORTA, 2        ;Display activ
;*****************************************************************

;Ziele der Register Operationen

w               equ 0                   ;w Zielregister
f               equ 1                   ;f Zielregister
;*****************************************************************************

;Macro Definitionen

bank0           macro
                bcf     STATUS, 5
                bcf     STATUS, 6
                endm

bank1           macro
                bsf     STATUS, 5
                endm

display_init    macro
                movlw   0x38            ;"Function set"
                call    write_command
                movlw   0x39            ;"Function set"
                call    write_command
                movlw   0x14            ;"Bias set"
                call    write_command
                movlw   0x78            ;"Contrast set"
                call    write_command
                movlw   0x52            ;"Power/ICON/Contrast control"
                call    write_command
                movlw   0x69            ;"Follower control"
                call    write_command
                call    delay           ;200msec warten
                movlw   0x0C            ;"Display /ON/OFF control"
                call    write_command
                movlw   0x01            ;"Clear display"
                call    write_command
                movlw   0x02
                call    delay           ;2msec warten
                movlw   0x06            ;Entry mode set"
                call    write_command
                endm
;*****************************************************************************

                org        0x00
;Programmbeginn bei Adresse 00h
                goto   start
;*****************************************************************************

;Subroutines
```

```
;Subroutine Initialisierung

init            bank1
                movlw   B'10001110'
                movwf   ADCON1          ;Right justified/Fosc/8/AN0 analog
                movlw   B'00000001'
                movwf   TRISA           ;AN0 configured as input
                movlw   .7
                movwf   CMCON           ;AN<2:0> comparators "Off"
                clrf    TRISB           ;RB<7:0> outputs
                bank0
                bcf     ADCON0, 7
                bsf     ADCON0, 6       ;Fosc 8
                bcf     ADCON0, 0       ;ADC depowered
                clrf    PORTA
                clrf    PORTB
                bcf     INTCON, 7       ;Global interrupt disable
                return


;Subroutine für 30µsec delay

delay03         movlw   0x0F
                movwf   counter2
                decfsz  counter2, 1     ;30µsec vorbei?
                goto    $-01            ;Nein.
                return


;Subroutine für eine verzögerung zwischen 1msec und 200msec

delay           movwf   counter1        ;Übergabeparameter für Delay sichern
                movlw   .250            ;Bei Verwendung eines 4-MHz-Taktes
                movwf   counter2
                nop
                decfsz  counter2, 1     ;1msec vorbei?
                goto    $-02            ;Nein
                decfsz  counter1, 1     ;Ja. Delay abgelaufen?
                goto    $-06            ;Nein
                return


;Subroutine ADC

ADC             bcf     ADCON0, 5
                bcf     ADCON0, 4
                bcf     ADCON0, 3       ;Eingangsspannung in AN0
                bsf     ADCON0, 0       ;ADC powered up
                NOP
                NOP                     ;2 µsec aquisition time
                bsf     ADCON0, 2       ;Start ADC
```

```
                                Vmessung
                btfsc   ADCON0, 2       ;Wandlung beendet?
                goto    $-01            ;Nein. Weiter warten
                movf    ADRESH, 0
                movwf   highbyte        ;Obere 2 Bits auslesen
                bank1
                movf    ADRESL, 0
                bank0
                movwf   lowbyte         ;Untere 8 bits auslesen
                bcf     ADCON0, 0       ;ADC depowered
                return


;Suproutine ADC Wert mit 25 multiplizieren [5 = (2+1)*2*2*2+1]

multiplyx25     bcf     STATUS, 0       ;Carrry bit löschen
                rlf     lowbytex25, 1
                rlf     highbytex25, 1  ;ADC Wert mit 2 multipliziert
                bcf     STATUS, 0       ;Carry Bit löschen
                movf    low_copy, 0
                addwf   lowbytex25, 1
                btfss   STATUS, 0       ;Carry Bit?
                goto    $+02            ;Nein
                incf    highbytex25, 1  ;Ja.
                movf    high_copy, 0
                addwf   highbytex25, 1  ;ADC Wert mit 3 multipliziert
                bcf     STATUS, 0       ;Carry bit löschen
                rlf     lowbytex25, 1
                rlf     highbytex25, 1  ;ADC Wert mit 6 multipliziert
                bcf     STATUS, 0       ;Carry Bit löschen
                rlf     lowbytex25, 1
                rlf     highbytex25, 1  ;ADC Wert mit 12 multipliziert
                bcf     STATUS, 0       ;Carry Bit löschen
                rlf     lowbytex25, 1
                rlf     highbytex25, 1  ;ADC Wert mit 24 multipliziert
                bcf     STATUS, 0       ;Carry Bit löschen
                movf    low_copy, 0
                addwf   lowbytex25, 1   ;Low ADC Wert mit 25 multiplitiert
                btfss   STATUS, 0       ;Carry Bit?
                goto    $+02            ;Nein
                incf    highbytex25, 1  ;Ja.
                movf    high_copy, 0
                addwf   highbytex25, 1  ;High ADC Wert mit 25 multipliziert
                return


;Suproutine ADC Wert mit 25 multiplizieren und durch 64 dividieren

divide_64       bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 2 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
```

```
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 4 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 8 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 16 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 32 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_64, 1
                rrf     lowx25_64, 1    ;ADC Wertx25 durch 64 dividiert
                return


;Subroutine ADC Wert mit 25 multiplizieren und durch 256 dividieren

divide_256      bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_256, 1
                rrf     lowx25_256, 1   ;ADC Wertx25 durch 128 dividiert
                bcf     STATUS, 0       ;Carry Bit löschen
                rrf     highx25_256, 1
                rrf     lowx25_256, 1   ;ADC Wertx25 durch 256 dividiert
                return



;Subroutine für tausener Wertigkeit

sub_taus        movlw   0x03
                movwf   highsub2
                movlw   0xE8
                movwf   lowsub2         ;1.000 in highsub(3) und lowsub2(232)setzen
                clrf    counter1        ;Wertigkeitscounter löschen
taus1           incf    counter1
                bsf     STATUS, 0       ;Carry bit setzen
                movf    lowsub2, 0
                subwf   lowsub1, 1      ;0xE8 (232) von lowsub1 abziehen
                btfsc   STATUS, 0       ;Borrow bit?
                goto    taus2           ;Nein
                decf    highsub1        ;Ja.
                bcf     STATUS, 2       ;Flag Z löschen
                movlw   0xFF
                subwf   highsub1, 1
                btfss   STATUS, 2       ;Highsub1 "0"?
                goto    $+04            ;Nein.
                movf    lowsub2, 0      ;Ja.
                addwf   lowsub1, 1
                goto    taus3
                decf    highsub1, 1
                goto    taus4
```

```
taus2           decf    highsub1
                bcf     STATUS, 2       ;Flag Z löschen
                movlw   0xFF
                subwf   highsub1, 1
                btfss   STATUS, 2       ;Highsub1 "0"?
                goto    $+03            ;Nein.
                goto    taus1           ;Ja.
taus4           bsf     STATUS, 0       ;Carry bit setzen
                movf    highsub2, 0
                subwf   highsub1, 1
                btfsc   STATUS, 0       ;Borrow bit?
                goto    taus1           ;Nein
                bcf     STATUS, 0       ;Ja. Carry bit löschen
                movf    lowsub2, 0
                addwf   lowsub1, 1
                btfss   STATUS, 0       ;Carry bit?
                goto    $+02            ;Nein
                incf    highsub1        ;Ja.
                movf    highsub2, 0
                addwf   highsub1, 1
taus3           decf    counter1
                return


;Subroutine für hunderter Wertigkeit

sub_hund        clrf    highsub2        ;Wird für die Rechnung nicht benötigt
                movlw   0x64
                movwf   lowsub2         ;100 in lowsub2
                clrf    counter1        ;Wertigkeitscounter löschen
hund1           incf    counter1
                bsf     STATUS, 0       ;Carry bit setzen
                movf    lowsub2, 0
                subwf   lowsub1, 1      ;0x64 (100) von lowsub1 abziehen
                btfsc   STATUS, 0       ;Borrow bit?
                goto    hund1           ;Nein
                decf    highsub1        ;Ja.
                bcf     STATUS, 2       ;Flag Z löschen
                movlw   0xFF
                subwf   highsub1, 1
                btfss   STATUS, 2       ;Highsub1 "0"?
                goto    $+04            ;Nein.
                movf    lowsub2, 0      ;Ja.
                addwf   lowsub1, 1
                goto    $+03
                decf    highsub1, 1
                goto    hund1
                decf    counter1
                return


;Subroutine für Zehner Wertigkeit
```

```
sub_zehn        movlw   0x0A
                movwf   lowsub2         ;10 in lowsub2 setzen
                clrf    counter1        ;Wertigkeitscounter löschen
                incf    counter1, 1
                bsf     STATUS, 0       ;Carrry Bit setzen.
                movf    owsub2, 0
                subwf   lowsub1, 1      ;10 von lowsub1 abziehen
                btfsc   STATUS, 0       ;Borrow bit?
                goto    $-05            ;Nein
                movf    lowsub2, 0      ;Ja
                addwf   lowsub1, 1      ;10 zu lowsub1 addieren
                decf    counter1
                return


;Subroutine für Einer Wertigkeit

sub_einer       movlw   .1
                movwf   lowsub2
                clrf    counter1
                incf    counter1, 1
                bsf     STATUS, 0       ;Curry Bit setzen
                movf    lowsub2, 0
                subwf   lowsub1, 1
                btfsc   STATUS, 0       ;Borrow Bit?
                goto    $-05            ;Nein
                decf    counter1        ;Ja.
                return


;Subroutine für das schreiben eines Kommandos

write_command   movwf   disp_store      ;Daten im Speicherregister
                bcf     DISP_CSB        ;Display aktivieren
                bcf     DISP_RS         ;Komando empfangen
                bcf     DISP_CLK
                movlw   0x08
                movwf   counter1        ;Es werden 8 Bits übertragen
                rlf     disp_store, 1   ;Display Zeichenspeicher nach links.
                btfsc   STATUS, 0       ;Carry Bit?
                goto    $+03            ;Nein
                bcf     DISP_SI         ;Ja.
                goto    $+02
                bsf     DISP_SI
                NOP
                bsf     DISP_CLK
                NOP
                bcf     DISP_CLK
                decfsz  counter1, 1     ;Display Ausgabezähler "NULL" ?
                goto    $-0B            ;Nein
```

```
                bcf     DISP_SI
                bcf     DISP_RS
                call    delay03
                return


;Subroutine für das schreiben eines Zeichens

write_data      movwf   disp_store      ;Daten im Speicherregister
                bcf     DISP_CSB        ;Display aktivieren
                bsf     DISP_RS         ;Data empfangen
                bcf     DISP_CLK
                movlw   0x08
                movwf   counter1        ;Es werden 8 Bits übertragen
                rlf     disp_store, 1   ;Display Zeichenspeicher nach links.
                btfsc   STATUS, 0       ;Carry Bit?
                goto    $+03            ;Nein
                bcf     DISP_SI         ;Ja.
                goto    $+02
                bsf     DISP_SI
                NOP
                bsf     DISP_CLK
                NOP
                bcf     DISP_CLK
                decfsz  counter1, 1     ;Display Ausgabezähler "NULL" ?
                goto    $-0B            ;Nein
                bcf     DISP_SI
                bcf     DISP_RS
                call    delay03
                return
;*****************************************************************

;Hauptprogramm

start           call            init

                movlw           0x28
                call            delay   ;40msec warten
                display_init

mainloop        call    ADC             ;AN0test=15,625/4=3,906V=800Bits
                movf    lowbyte, 0
                movwf   low_copy
                movwf   lowbytex25
                movf    highbyte, 0
                movwf   high_copy
                movwf   highbytex25

                call    multiplyx25
                movf    highbytex25, 0
                movwf   highx25_64
                movf    lowbytex25, 0
```

```
movwf   lowx25_64

call    divide_64
movf    highx25_64, 0
movwf   highx25_256
movf    lowx25_64, 0
movwf   lowx25_256

call    divide_256
bcf     STATUS, 0       ;Carry Bit löschen
movf    lowx25_64, 0
addwf   lowx25_256, 0
movwf   lowv_hund       ;Lowbyte des gemessenen Spannungx100
btfss   STATUS, 0       ;Carry bit?
goto    $+02            ;Nein
incf    highx25_256, 1  ;Ja.
movf    highx25_64, 0
addwf   highx25_256, 0
movwf   highv_hund      ;Highbyte des gemessenen Spannungx100
bcf     STATUS, 0       ;Carry Bit löschen
rlf     highv_hund, 1
rlf     lowv_hund, 1
btfss   STATUS, 0       ;Carry bit?
goto    $+02            ;Nein
incf    highv_hund, 1   ;Ja.
bcf     STATUS, 0       ;Carry Bit löschen
rlf     highv_hund, 1
rlf     lowv_hund, 1
btfss   STATUS, 0       ;Carry bit?
goto    $+02            ;Nein
incf    highv_hund, 1   ;Ja.
movf    highv_hund, 0
movwf   highv           ;Highbyte der Anzuzeigenden Spannung
movwf   highsub1
movf    lowv_hund, 0
movwf   lowv            ;Lowbyte der anzuzeigenden Spannung
movwf   lowsub1
call    sub_taus
movf    counter1, 0
movwf   disp_taus

call    sub_hund
movf    counter1, 0
movwf   disp_hund

call    sub_zehn
movf    counter1, 0
movwf   disp_zehn

call    sub_einer
movf    counter1, 0
movwf   disp_einer
```

```
bsf     PORTA, 2        ;Übernahme von Daten beginnt
movlw   0x80
call    write_command   ;Position 1 der Zeile 1 (=0x80+0x00)
movlw   0x20
call    write_data      ;Leerzeichen 1 in Zeile 1
movlw   0x20
call    write_data
movlw   A'U'
call    write_data
movlw   A'o'
call    write_data
movlw   A'='
call    write_data
movlw   0x30
addwf   disp_taus, 0
call    write_data      ;Wertigkeit "tausend" Position 6 Zeile 1
movlw   0x30
addwf   disp_hund, 0
call    write_data      ;Wertigkeit "hundert" Position 7 Zeile 1
movlw   A','
call    write_data
movlw   0x30
addwf   disp_zehn, 0
call    write_data      ;Wertigkeit "zehner" Position 9 Zeile 1
movlw   0x30
addwf   disp_einer, 0
call    write_data      ;Wertigkeit "einer" Position 10;Zeile 1
movlw   A'['
call    write_data
movlw   A'V'
call    write_data
movlw   A']'
call    write_data

movlw   0xC0
call    write_command   ;Position 1 der Zeile 2 (=0x80+0x40)
movlw   A'.'
call    write_data      ;"." Zeichen 1 in Zeile 2
movlw   A'.'
call    write_data
movlw   A'U'
call    write_data
movlw   A'o'
call    write_data
movlw   A'm'
call    write_data
movlw   A'a'
call    write_data
movlw   A'x'
call    write_data
movlw   A'='
```

```
                call    write_data
                movlw   A'2'
                call    write_data
                movlw   A'0'
                call    write_data
                movlw   A'['
                call    write_data
                movlw   A'V'
                call    write_data
                movlw   A']'
                call    write_data
                movlw   A'.'
                call    write_data
                movlw   A'.'
                call    write_data
                movlw   A'.'
                movlw   .250
                call    delay
                movlw   .250
                call    delay
                movlw   .250
                call    delay           ;Warte 750msec
                bcf     PORTA, 2        ;Übernahme von Daten beendet
                movlw   .250
                call    delay
                movlw   .250
                call    delay
                movlw   .250
                call    delay           ;Warte 750msec
                goto    mainloop
                end
;*****************************************************************
```