```
/* Kettenzählwerk mit 3-Stellige 7-Segment Anzeige
 * Permanent Magnet N-35 M 50x22 (5x20mm)
 * Reed Sensor Schließer PIC MS-225-3
 * RB0 Digital input (Reed Sensor Eingang)
 * RA<2:0> Digital Outputs (LED Anoden).
 * RA<5&3> Digital inputs (RA3 UP:Down / RA5 LED Test)
 * RC<6:0> Digital outputs (LED Kathoden)
 * LED DOT: Kettenrichtungsanzeige ("ON going "Down"/"OFF" going "UP")
 * File:   chain_counter
 * Author: Lasaros Goumas
 * Created on 27. Juni 2022, 18:23
 */


/* Includes

*****************************************************************************
**/
#include <xc.h>
#include <p18cxxx.h>                //PIC 18F25K22 Controller


/*Configuration

*****************************************************************************
***/
#pragma config FOSC = INTIO67   //Internal oscillator block
#pragma config PWRTEN = ON     //Power up timer enabled
#pragma config WDTEN = OFF      //WDT disabled
#pragma config PBADEN = OFF     //PORTB<5:0> digital I/O on reset
#pragma config LVP = OFF
#pragma config CP0 = OFF
#pragma config CPD = OFF
#pragma config WRT0 = OFF
#pragma config WRTD = OFF
#pragma config EBTR0 = OFF
#pragma config EBTRB = OFF


/*Declarations

*****************************************************************************
**/
#define _XTAL_FREQ 8000000          // Fosc  frequency for _delay()
#define LED_DATA PORTC              //PORTC is dataport vor LED´s
#define LED_1 PORTAbits.RA0         //Anode LED1
#define LED_2 PORTAbits.RA1         //Anode LED2
#define LED_3 PORTAbits.RA2         //Anode LED3
#define status PORTAbits.RA3        //Windlass turning direction
(Up/Down)
#define test PORTAbits.RA5          //Press to test LED´s
#define direction PORTBbits.RB1     //Kettenrichtung (Down=0 / Up=1)
unsigned int drumup;                //Trommelimpulszähler "Up"
unsigned int drumdown;              //Trommelimpulszähler "Down"
unsigned int countup;               //Kettenlängezählregister "Up"
unsigned int countdown;             //Kettenlängezählregister "Down"
unsigned int hund;                  //Wertigkeit 100
```

```
unsigned int zehn;                  //Wertigkeit 10
unsigned int eins;                  //Wertigkeit eine
unsigned char chain;                //Deploid chain


/*Funktions

*************************************************************************
/

void init_PIC (void){
    ANSELA = 0x00;              //PORTA as digital
    TRISA = 0b00101000;         //RA3 & RA5 are inputs
    ANSELB = 0x00;              //PORTB as digital
    TRISB = 0b00000001;         //RB0 as input
    LED_1 = 1;
    LED_2 = 1;
    LED_3 = 1;                  //All LED´s disabled
    ANSELC = 0x00;              //PORTC as digital
    TRISC = 0x00;               //All PORTC pins are outputs
    chain = 0x00;               //Deploid chain cleared
    OSCCON = 0b01100111;        //8Mhz internal Oscillator stable
    T1CON = 0b00110110;         //16bit instr. source timer1;1/8
    PIE1bits.TMR1IE = 0x00;     //Timer1 overflow interrupt disabled
}

    void init_interrupts (void){
    RCONbits.IPEN = 1;          //Enable priority levels om imterrups
    INTCON = 0b11010000;        //Global; external & INTO enabled
    INTCON2bits.INTEDG0 = 1;    //External interrupt on rising egde
    ei();
}


    void __interrupt()_High_Prio (void){
    if (INT0IE && INT0IF)           //INTO interrupt?
        INTCONbits.INT0IF = 0;
    if (status == 0) drumdown++;
    else drumup++;}


    void init_wertigkeit (void){
        hund = 0;                   //Hunderter Wertigkeit
        LED_DATA = chain;
        while (1)
            if (LED_DATA>=100){
                ++hund;
                LED_DATA = LED_DATA-100;}
        else break;

        zehn = 0;                   //Zehner Wertigkeit
        while (1)
            if (LED_DATA>=10){
            ++zehn;
            LED_DATA = LED_DATA-10;}
        else break;
```

```
        eins = LED_DATA;                //Einer Wertigkeit
}


/*Main Routine

**************************************************************************
/

void main(void){
    init_PIC();
    init_interrupts();

    for(;;){
        while(1){
            if (test == 0x01){
                LED_DATA = 0x80;        //All 7 segments "ON"
                LED_1 = 0;
                LED_2 = 0;
                LED_3 = 0;}
            else {LED_1 = 1;
            LED_2 = 1;
            LED_3 = 1;
            break;}
        }


        if (drumdown>drumup)
            countdown = drumdown-drumup;
        else if (drumup>drumdown)
            countup = drumup-drumdown;
        else {drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}        //Clear all counters

        if (chain == 0x00){
            if (countdown >=0x07){
                chain++;
                direction = 0x00;       //Going Down
                drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}        //Clear all counters
        if (countup >= 0x07){
            chain--;
            chain=chain-254;
            direction = 0x01;           //Going Up
            drumdown = 0x00;
            drumup = 0x00;
            countdown = 0x00;
            countup = 0x00;}            //Clear all counters
        }

        else  if (chain >= 0x01){
            if (countdown >= 0x07){
                if (direction == 0x00){
```

```
                chain++;
                drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}          //Clear all counters
                if (direction == 0x01){
                chain--;
                drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}          //Clear all counters
                }
        if (countup >= 0x07){
            if (direction == 0x00){
                chain--;
                drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}          //Clear all counters
                if (direction == 0x01){
                chain++;
                drumdown = 0x00;
                drumup = 0x00;
                countdown = 0x00;
                countup = 0x00;}          //Clear all counters
                }
        }

        init_wertigkeit();
        unsigned char segment[10]={0xC0,0xF9,0xA4,0xB0,0x99,
        0x92,0x82,0xF8,0x80,0x90};

        LED_DATA = segment[eins];
        LED_1 = 0;
        T1CONbits.TMR1ON = 0x01;    //Start TMR1 timer
        TMR1 = 0x00;
        while (TMR1!=0x04E2);       //Warte 5msec (0x04E2 /1.250)
        LED_1 = 1;
        LED_DATA = segment[zehn];
        LED_2 = 0;
        TMR1 = 0x00;
        while (TMR1!=0x04E2);       //Warte 5msec
        LED_2 = 1;
        LED_DATA = segment[hund];
        LED_3 = 0;
        TMR1 = 0x00;
        while (TMR1!=0x04E2);       //Warte 5msec
        LED_3 = 1;
    }
}
```