```
/* File:   newmain.c Author: khk Created on 29. März 2017, 16:58 */

#include <xc.h>
#define _XTAL_FREQ 500000     // interne Frequenz 500 kHz
#pragma config FOSC = INTOSCIO  // Oscillator Selection bits (INTRC oscillator; port I/O function
                                // on both RA6/OSC2/CLKO pin and RA7/OSC1/CLKI pin)
#pragma config WDTE = ON        // Watchdog Timer Enable bit (WDT enabled)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT disabled)
#pragma config MCLRE = OFF      // RA5/MCLR/VPP Pin Function Select bit (RA5/MCLR/VPP pin function
                                // is digital I/O, MCLR internally tied to VDD)
#pragma config BOREN = OFF      // Brown-out Reset Enable bit (BOR disabled)
#pragma config LVP = OFF        // Low-Voltage Programming Enable bit (RB3 is digital I/O, HV on
                                // MCLR must be used for programming)
#pragma config CPD = OFF        // Data EE Memory Code Protection bit (Code protection off)
#pragma config WRT = OFF        // Flash Program Memory Write Enable bits (Write protection off)
#pragma config CCPMX = RB0      // CCP1 Pin Selection bit (CCP1 function on RB0)
#pragma config CP = OFF         // Flash Program Memory Code Protection bit (Code protection off)
#pragma config FCMEN = OFF      // Fail-Safe Clock Monitor Enable bit (disabled)
#pragma config IESO = OFF       // Internal External Switchover bit (disabled)

int timeoutcounter = 0;
char PORTBTemp;
char PORTBvorher;
char PORTBvorvorher;
char ersteMessung = 1;
void teste_auf_stehendes_Auto(void);
void DummiMessung(void);
void main(void) {
    ANSEL = 0x00;
    TRISA = 0b10000000; // Ra6=trigg=Output; RA7=Echo=input; Rest=Output
    TRISB = 0; // alle B sind Output
//    OSCCON = 0b01100000; // 4 MHz
    OSCCON = 0b00110000; // 500 kHz
    PORTA = 0;
    PORTB = 0b00000001; // Einschaltmeldung durch LED1
    __delay_ms(50);     // 1. kurzer Blitz
    PORTB = 0b00000000;
     __delay_ms(200);
    PORTB = 0b00000001; // Einschaltmeldung durch LED1
    __delay_ms(50);     // 2. kurzer Blitz
    PORTB = 0b00000000;
    while (1) {
        CLRWDT();
        if(ersteMessung){
            DummiMessung();
        }
        PORTBTemp = 0; // erstmal alle LEDs aus
        PORTAbits.RA6 = 1; // Triggerpuls 20µs
 //       __delay_ms(1000);     // zum testen ob Stopwatch stimmt
        __delay_us(30);
        PORTAbits.RA6 = 0;
        NOP();
        while (!RA7)  { // warten das RA7 high wird
                        // wenn nicht, kommt reset durch watchdog
        }
        if (RA7 == 1) {
            __delay_us(150); // 2cm=120µs warten
            if (RA7 == 1) {
                PORTBTemp = 0b00000001;
            }
            __delay_us(40); // 1cm=58µs warten
            if (RA7 == 1) {
                PORTBTemp = 0b00000011;
            }
            __delay_us(40);
            if (RA7 == 1) {
                PORTBTemp = 0b00000111;
            }
            __delay_us(40);
            if (RA7 == 1) {
                PORTBTemp = 0b00001111;
            }
            __delay_us(60);
            if (RA7 == 1) {
```

```
                PORTBTemp = 0b00011111;
            }
            __delay_us(90);
            if (RA7 == 1) {
                PORTBTemp = 0b00111111;
            }
            __delay_us(120);
            if (RA7 == 1) {
                PORTBTemp = 0b01111111;
            }
            __delay_us(150);
            if (RA7 == 1) {
                PORTBTemp = 0b11111111;
            }
            CLRWDT();
            __delay_us(500);
            if (RA7 == 1) {
                PORTAbits.RA0 = 1;      // US-Modul aus
                PORTB = 0b10000000;     // 1. Schlafgrund: kein Echo
                __delay_ms(10);         // nur kurzer
                PORTB = 0b00000000;     // Blitz
                SLEEP();
                PORTAbits.RA0 = 0;      // US-Modul wieder ein
                PORTB = 0b10000000;     // ganz kurzer
                __delay_ms(2);          // Blitz
                PORTB = 0b00000000;
                __delay_ms(80);
            }
            else{
                PORTB = PORTBTemp;
                teste_auf_stehendes_Auto();
            }
            __delay_ms(50);
        }
    }
    return;
}
    void DummiMessung(void){
         PORTAbits.RA6 = 1; // Triggerpuls 20µs
        __delay_us(30);
        PORTAbits.RA6 = 0;
         __delay_ms(80);
        ersteMessung = 0;
        CLRWDT();
    }
    void teste_auf_stehendes_Auto(void) {
        if (timeoutcounter == 0){
            PORTBvorher = PORTBTemp;        // Erstinitialisierung
        }
        if(PORTBvorher == PORTBTemp){
            timeoutcounter++;               // wenn stabiler Wert -> hochzählen
        }
        else{
            timeoutcounter = 0;             // bei Änderung
            PORTBvorvorher = 0;             // -> wieder auf Anfang
        }
        if(timeoutcounter>=100 || PORTBvorvorher == PORTBTemp){
            // wenn lange stabil -> schlafen
            PORTBvorvorher = PORTBvorher;
            timeoutcounter = 0;
            PORTAbits.RA0 = 1;      // US-Modul aus
            PORTB = 0b01000000;     // 2.Schlafgrund lange gleicher Wert
            __delay_ms(10);         // nur kurzer
            PORTB = 0b00000000;     // Blitz
            SLEEP();
            PORTAbits.RA0 = 0;      // US-Modul wieder ein

            PORTB = 0b11000000;
            __delay_ms(2);
            PORTB = 0b00000000;
            ersteMessung = 1;
            __delay_ms(100);
        }
}
```