ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ (MICROLAB)

7η Εργαστηριακή Αναφορά στο Μάθημα "**Εργαστήριο Μικροϋπολογιστών**" του 7ου Εξαμήνου

των φοιτητών της ομάδας 17,

Εμμανουήλ Αναστασίου Σερλή, ΑΜ 03118125
Ευάγγελου Περσόπουλος, ΑΜ 03121701

Ημερομηνία εργαστηριακής εξέτασης: Τρίτη 14 Δεκεμβρίου 2022
Ημερομηνία υποβολής: Κυριακή 18 Δεκεμβρίου 2022

1η Άσκηση -> micro_lab07_ex01

Για την υλοποίηση του ζητούμενου προγράμματος χρησιμοποιήσαμε τις δοθείσες, σε assembly, ρουτίνες αφού πρώτα τις μετατρέψαμε σε γλώσσα C.

Για τη μέτρηση της θερμοκρασίας αναπτύξαμε τη ρουτίνα _temperature_value η οποία διαβάζει τον αισθητήρα θερμοκρασίας DS18B20 και επιστρέφει τη τιμή της μέτρηση ή σε περίπτωση αδυναμίας εύρεσης της συσκευής επιστρέφει τη τιμή 0x8000

C Program:

```
#define F_CPU 16000000UL //running
#include<avr/io.h>
#include<avr/interrupt.h>
#include<util/delay.h>

int _sign;  //for sign of temperature

unsigned int one_wire_reset(void)
{
    unsigned int x=0;
    DDRD |= (1<<PD4); //set PD4 as output
    PORTD &= (0<<PD4);
    _delay_us(480);
    DDRD &= (0<<PD4); //set PD4 as input
    PORTD &= (0<<PD4); //disable pull-up
    _delay_us(100);
    x = PIND;
    x = x >>4;
    _delay_us(380);
    if ((x & 0x01) == 0x01) return 0;   //not connected device
    else return 1;
}

unsigned int one_wire_receive_bit (void)
{
    uint8_t x;
    DDRD |= (1<<PD4); //set PD4 as output
    PORTD &= (0<<PD4);
    _delay_us(2);
    DDRD &= (0<<PD4);
    PORTD &= (0<<PD4);
    _delay_us(10);
    x = PIND;
    x &= 0x10;
    x = x >> 4;
    _delay_us(49);
    return x;
}
```

```
void one_wire_transmit_bit (unsigned int y)
{
   DDRD |= (1<<PD4); //set PD4 as output
   PORTD &= (0<<PD4);
   _delay_us(2);
   if (y == 1) PORTD |= (1<<PD4);
   _delay_us(58);
   DDRD &= (0<<PD4);
   PORTD &= (0<<PD4);
   _delay_us(1);   //recovery time
}

unsigned int one_wire_receive_byte (void)
{
   unsigned int value=0, x = 0;
   for (int i=0; i<8; i++)
   {
      value = value >> 1;
      x = one_wire_receive_bit();
      if (x == 1) value |= 0x80;
   }
   return value;
}

void one_wire_transmit_byte (unsigned int z)
{
   for (int i=0; i<8; i++)
   {
      one_wire_transmit_bit (z & 0x01);
      z = z >> 1;
   }
}

int _temperature_value (void)
{
   int value_H=0, value_L=0, x;
   x = one_wire_reset();
   if (x==0) return 0x8000;          //check for connected device
   one_wire_transmit_byte (0xCC);
   one_wire_transmit_byte (0x44);
   x = 0;
   while (x == 0)
   {
      x = one_wire_receive_bit();  //wait here until the measurement is over
   }
   x = one_wire_reset();
   if (x==0) return 0x8000;
   one_wire_transmit_byte (0xCC);
   one_wire_transmit_byte (0xBE);
   value_L = one_wire_receive_byte();   //in value_L we store the 8 LSB of the temperature
                                        //value

   value_H = one_wire_receive_byte();  //in value_H we store the 8 MSB of the temperature
```

```
                                        //value

    int temp2 = value_H & 0xF8;    //we isolate the 5 MSB
    if (temp2 == 0xF8) _sign = 1;   //if the 5 MSB of value_H are set we store 1 in the global
                                    //variable _sign to represent negative temperature
    value_L &= 0xFF;        //isolate the 8 LSB
    value_H &= 0x07;        //discard sign bits
    value_H = value_H << 8; //shift bits 0-2 to position 8-10
    value_L |= value_H; //and combine the value in the 16bit variable value_L
    return value_L;    //now value_L has the temperature without the sign bits

}
```

2η Άσκηση -> micro_lab07_ex02

Για την υλοποίηση του ζητούμενου προγράμματος αναπτύξαμε τις εξής ρουτίνες:

fix_temp_for_lcd: παίρνει σαν ορίσματα τη τιμή της θερμοκρασίας όπως την επιστρέφει η _temperature_value όπως επίσης και το πρόσημο της (0 για θετικές τιμές και 1 για αρνητικές τιμές). Αν έχουμε αρνητική τιμή κάνει τη μετατροπή του συμπληρώματος ως προς 2 και στη συνέχεια τυπώνει στην LCD τη τιμή της θερμοκρασίας.

zero_temp: σε περίπτωση μηδενικής θερμοκρασίας καλείται η zero_temp να τυπώσει το 0 στην οθόνη και να αποφύγουμε περιττούς υπολογισμούς.

no_dev: αν η _temperature_value επιστρέψει 0x8000 καλείται η no_dev για να τυπώσει στην οθόνη NO Device.

```
#define F_CPU 16000000UL //running
#include<avr/io.h>
#include<avr/interrupt.h>
#include<util/delay.h>
#include<math.h>

int _sign;  //for sign of temperature

void write_2_nibbles(char x)
{
        char y=PIND & 0x0f;
        char x1=x & 0xf0;
        x1=x1+y;
        PORTD=x1;
        PORTD=PORTD | (1<<PD3);
        PORTD=PORTD & (0<<PD3);
        x=x<<4 | x>>4;
        x=x & 0xf0;
        PORTD=x+y;
        PORTD=PORTD | (1<<PD3);
        PORTD=PORTD & (0<<PD3);
}
```

```
void lcd_data(char x)
{
        PORTD=PORTD | (1<<PD2);
        write_2_nibbles(x);
        _delay_us(50);
}

void lcd_command(char x)
{
        PORTD=PORTD | (0<<PD2);
        write_2_nibbles(x);
        _delay_us(50);
}

void lcd_init (void)
{
        _delay_ms(40);
        PORTD=0x30;
        PORTD=PORTD | (1<<PD3);
        PORTD=PORTD & (0<<PD3);
        _delay_us(38);
        PORTD=0x30;
        PORTD=PORTD | (1<<PD3);
        PORTD=PORTD & (0<<PD3);
        _delay_us(38);
        PORTD=0x20;
        PORTD=PORTD | (1<<PD3);
        PORTD=PORTD & (0<<PD3);
        _delay_us(38);
        lcd_command(0x28);
        lcd_command(0x0c);
        lcd_command(0x01);
        _delay_ms(500);
        lcd_command(0x06);
}

unsigned int one_wire_reset(void)
{
   unsigned int x=0;
   DDRD |= (1<<PD4); //set PD4 as output
   PORTD &= (0<<PD4);
   _delay_us(480);
   DDRD &= (0<<PD4); //set PD4 as input
   PORTD &= (0<<PD4); //disable pull-up
   _delay_us(100);
   x = PIND;
   x = x >>4;
   _delay_us(380);
   if ((x & 0x01) == 0x01) return 0;   //not connected device
   else return 1;
}
```

```
unsigned int one_wire_receive_bit (void)
{
    uint8_t x;
    DDRD |= (1<<PD4); //set PD4 as output
    PORTD &= (0<<PD4);
    _delay_us(2);
    DDRD &= (0<<PD4);
    PORTD &= (0<<PD4);
    _delay_us(10);
    x = PIND;
    x &= 0x10;
    x = x >> 4;
    _delay_us(49);
    return x;
}

void one_wire_transmit_bit (unsigned int y)
{
    DDRD |= (1<<PD4); //set PD4 as output
    PORTD &= (0<<PD4);
    _delay_us(2);
    if (y == 1) PORTD |= (1<<PD4);
    _delay_us(58);
    DDRD &= (0<<PD4);
    PORTD &= (0<<PD4);
    _delay_us(1);   //recovery time
}

unsigned int one_wire_receive_byte (void)
{
    unsigned int value=0, x = 0;
    for (int i=0; i<8; i++)
    {
        value = value >> 1;
        x = one_wire_receive_bit();
        if (x == 1) value |= 0x80;
    }
    return value;
}

void one_wire_transmit_byte (unsigned int z)
{
    for (int i=0; i<8; i++)
    {
        one_wire_transmit_bit (z & 0x01);
        z = z >> 1;
    }
}

int _temperature_value (void)
{
    int value_H=0, value_L=0, x;
    x = one_wire_reset();
```

```
    if (x==0) return 0x8000;          //check for connected device
    one_wire_transmit_byte (0xCC);
    one_wire_transmit_byte (0x44);
    x = 0;
    while (x == 0)
    {
       x = one_wire_receive_bit();  //wait here until the measurement is over
    }
    x = one_wire_reset();
    if (x==0) return 0x8000;
    one_wire_transmit_byte (0xCC);
    one_wire_transmit_byte (0xBE);
    value_L = one_wire_receive_byte();  //in value_L we store the 8 LSB of the temperature
                                        //value

    value_H = one_wire_receive_byte();  //in value_H we store the 8 MSB of the temperature
                                        //value

    int temp2 = value_H & 0xF8;   //we isolate the 5 MSB
    if (temp2 == 0xF8) _sign = 1;   //if the 5 MSB of value_H are set we store 1 in the global
                                    //variable _sign to represent negative temperature
    value_L &= 0xFF;       //isolate the 8 LSB
    value_H &= 0x07;       //discard sign bits
    value_H = value_H << 8; //shift bits 0-2 to position 8-10
    value_L |= value_H; //and combine the value in the 16bit variable value_L
    return value_L;    //now value_L has the temperature without the sign bits

}


void fix_temp_for_lcd(int a, int sign)
{
    DDRD |= 0b11111111;    //PORTD as output
    lcd_init();  //initialize LCD
    int number1=0, number2=0, number3=0, check = 0;
    float dec=0;
    if (sign)     //for negative values
    {
       lcd_data('-');
       a = (~a) + 1;   //2 complement conversion
       a &= 0x7FF;
    }
    for (int i=4; i>0; i--)        //we isolate and store the decimal part to dec
    {
       check = a & 0x01;
       if (check == 1) dec += 1/(pow(2,i));
       a = a >> 1;
       check = 0;
    }                              //now a has only the integer value of the temperature
    number1 = a/100;
    if (number1 == 1) lcd_data(number1 + '0');
    number2 = (a-(number1*100))/10;
    _delay_ms(10);
```

```
    lcd_data(number2 + '0');
    number3 = a-(number1*100 + number2*10);
    _delay_ms(10);
    lcd_data(number3 + '0');
    if(dec!=0) {
        lcd_data('.');
        for (int i=0; i<4; i++)    //after '.' we print the decimal value to LCD
        {
            check = dec*10;
            _delay_ms(10);
            if (check != 0) lcd_data (check + '0');
            dec = dec*10 - (float)check;
            _delay_ms(10);
        }
    }
}

void zero_temp(void)
{
    DDRD |= 0b11111111;
    lcd_init();
    lcd_data ('0');
    lcd_data ('0');
    lcd_data ('.');
    lcd_data ('0');
    lcd_data(' ');
    lcd_data('°');
    lcd_data('C');
}

void no_dev(void)
{
    DDRD |= 0b11111111;
    lcd_init();
    lcd_data ('N');
    lcd_data ('O');
    lcd_data (' ');
    lcd_data ('D');
    lcd_data ('e');
    lcd_data ('v');
    lcd_data ('i');
    lcd_data ('c');
    lcd_data ('e');
}

int main(void)
{
    int temp = 0;

    while(1)
    {
        _delay_ms(1500);      //every 1.5 sec we take a measurement
        _sign = 0;
```

```
        temp = _temperature_value();
        if (temp == 0)
        {
            zero_temp();
            continue;
        }
        if (temp == 0x8000)
        {
            no_dev();
            continue;
        }
        fix_temp_for_lcd(temp, _sign);
    }
    return 0;
}
```