Εργαστήριο Μικροϋπολογιστών: 7η Εργαστηριακή Άσκηση (2022)

Ομάδα 64

Ιωάννου Κωνσταντίνος AM:03119840

Μυτιληναίος Γιώργος AM:03119841

```
132:  int one_wire_reset()
133:  {
134:     DDRD = 0b0;
135:     PORTD= 0b0;
136:     uint8_t sav;
137:    DDRD|= 0b00010000;
138:   PORTD&=0b11101111;
139:     _delay_us(480);
140:     DDRD&= 0b11101111;
141:     PORTD&=0b11101111;
142:     _delay_us(100);
143:     sav = PIND;
144:    _delay_us(380);
145:    sav = sav&0b00010000;
146:    sav = sav>>4;
147:    if(sav==0) return 1;  //if a connected
devise is detected
148:    if(sav==1) return 0;  // if not
149: }
```

```
169: void one_wire_transmit_bit(uint8_t x)
170: {
171:    DDRD|= 0b00010000;
172:    PORTD&=0b11101111;
173:   _delay_us(2);
174:   x=(x&(0b001));
175:   if(x==0) PORTD &=0b11101111;
176:   else PORTD |=0b00010000;
177:     _delay_us(58);
178:     DDRD&= 0b11101111;
179:     PORTD&=0b11101111;
180:     _delay_us(1);
181: }


198: void one_wire_transmit_byte(uint8_t x) //egrafi enos
byte ston DS1820
199: {
200:    uint8_t y;
201:    for(int i =0 ;i<8;i++){
202:    if( (x&(0b01)) == 0) y=0b0;
203:    else y = 0b01;
204:    one_wire_trasnmit_bit(y);
205:    x= x>>1;
206:    }
207:    return 0;
208: }
```

Ζήτημα 7.1.c:

- Το πρώτο βήμα της άσκησης είναι η δημιουργία της one_wire_reset() όπου ανιχνεύει άμα υπάρχει συσκευή και την αρχικοποιεί. Μηδενίζουμε το PortD (γραμμές 134-135) για να ξέρουμε σε τι κατάσταση βρίσκεται. Έπειτα θέτουμε το PD4 ως έξοδο και pull down. Αυτό στέλνει σήμα στο DS1820, για να διαβαστεί το σήμα που θα στείλει το θερμόμετρο κάνουμε το PD4 είσοδο(γραμμές 140-141) κι αποθηκεύουμε την τιμή στην μεταβλητή sav (γραμμή 143) . Επιστρέφεται το αντίθετο της τιμή της sav ανάλογα αν έχει εντοπισθεί η συσκευή (γραμμές 147-148).
- Το επόμενο βήμα είναι η χρήση της εντολής 0xCC. Η συνάρτηση one_wire_transmit_bit στέλνει δεδομένα από την κάρτα NtuAboard_G1 στο θερμόμετρο. Ανάλογα του μονάμπιτου αριθμού που δοθεί στην συνάρτηση θέτεται το pull up ή pull down του PD4. Για να γίνει μεταφορά παραπάνω από ενός bit καλούμε την one_wire_transmit_byte όπου διαβάζει ένα 8bit αριθμό κι στέλνει ένα-ένα τα bit στην one_wire_reset().
- Παρόμοια διαδικασία κι για το 3^{ο} βήμα όπου στέλνουμε την εντολή 0x44 που λέει στο θερμόμετρο να αρχίσει να μετράει θερμοκρασία. Για να εντοπίσουμε αν τελείωσε, περιμένουμε
- να λάβουμε ένα παλμό από τον DS1820. Για να εντοπίσουμε τον παλμό καλούμε τη συνάρτηση one_wire_receive() όπου κοιτάει την γραμμή κι στέλνει την τιμή που διάβασε. Επειδή δεν ξέρουμε πότε θα τελειώσει να μετράει καλούμε την συνάρτηση μέσα σε ένα do-while loop μέχρι να λάβει τον παλμό.
- Στην συνέχεια καλούμε ξανά την one_wire_reset().
- Και στέλνουμε την εντολή με την συνάρτηση one_wire_transmit_byte()

```
185: uint8_t one_wire_receive_byte() //read a byte
from DS1820
186: {    uint8_t x;
187:    uint8_t y = 0;
188:
189:    for(int i =0 ;i<8;i++){
190:    x= one_wire_receive_bit();
191:    y = y>>1;
192:    if(x==0) y=x|y;
193:    else   y=0x080|y;
194:    }
195:    return y;
196: }
```

```
210: float find_temp()
211: {
212:    int number=0;
213:    int test = one_wire_reset();
214:    if (test == 0 ) return 0x800;
215:
216:    one_wire_trasnmit_byte(0xCC);
217:    one_wire_trasnmit_byte(0x44);
218:
219:   do{test = one_wire_receive_bit() ;}
220:    while(test==0) ;
221:
222:    one_wire_reset();
223:    one_wire_trasnmit_byte(0xCC);
224:    one_wire_trasnmit_byte(0xBE);
225:    uint8_t r25,r24; //r25::r24
226:
227:   r24 =one_wire_receive_byte() ; //read first LSB
and then MSB bits
228:   r25 =one_wire_receive_byte() ;
229:
230:    uint16_t temp;
231:
232:    temp = r25;
233:    temp = temp<<8;
234:    temp = temp|r24;
235:
236:    float x ;
237:    r25 =r25&0b10000000;
238:
239:    if(r25==0b0) {  x=(temp*(62.5));  return x; }
240:
241:    if(r25==0b10000000)
242:    {
243:       temp =~temp;
244:       temp = temp+ 0b01;
245:       x = - (temp*(62.5));
246:       return x; }
247:    }
```

- Για να πάρουμε την τιμή που μετρήθηκε στέλνουμε την τιμή 0xBE και διαβάζουμε την τιμή με την τιμή με την συνάρτηση one_wire_receive_byte(). Οι τιμές που διαβάζονται τις βάζουμε στις 8bit μεταβλητές r24/r25 (γραμμές 227-228). Κάνουμε shift την r25 ώστε να μπορούμε να βάλουμε και τις δύο στην temp όπου είναι μια 16bit μεταβλητή. Το r25 εκτός από δεδομένα για τον αριθμό μας δείχνει αν η θερμοκρασία είναι αρνητική ή θετική. Κοιτάμε το MSB της r25, αν είναι 1 τότε είναι αρνητικό (datasheet).

αν είναι αρνητικό τότε μετατρέπουμε το παίρνουμε το συμπλήρωμα της temp ώστε να μπορούμε να διαβάσουμε τον σωστό δυαδικό αριθμό. Το πολλαπλασιάζουμε 62,5 για να πάρουμε τον σωστό αριθμό σε C◦.

- Αν δεν υπάρχει συνδεδεμένη συσκευή (δηλαδή ανάλογα την τιμή που θα επιστρέψει η one_wire_reset()) τότε επιστρέφει τιμή 0x800 (γραμμή 213-214)

```
File: Untitled-1
01: void LCD_print( float   x ) {
02:
03:   if(x==0x800) {
04:       lcd_command(0x01);
05:     _delay_ms(3);
06:
07:       const char *txt= "NO Device";
08:       while (*txt)
09:          {
10:                       lcd_data(*txt++);
11:          }
12:      return 0;
13:   }
14:  int temp;
15:  temp = abs(x) ;
16: int  d1,d2,d3,d4,d5,d6;
17:
18: d1=temp/100000;
19: temp=temp-d1*100000;
20: d2=temp/10000;
21: temp=temp-d2*10000;
22: d3=temp/1000;
23: temp=temp-d3*1000;
24: d4=temp/100;
25: temp=temp-d4*100;
26: d5=temp/10;
27: temp=temp-d5*10;
28: d6=temp;
29:
30: unsigned  char ch1,ch2,ch3,ch4,ch5,ch6;
31: ch1 = d1 + '0';
32: ch2 = d2 + '0';
33: ch3 = d3 + '0';
34: ch4 = d4 + '0';
35: ch5 = d5 + '0';
36: ch6 = d6 + '0';
37:
38:     lcd_command(0x01);
39:     _delay_ms(3);
40:
41:     if (x<0) lcd_data('-');
42:     if (x>0) lcd_data('+');
43:       _delay_ms(10);
44:     lcd_data(ch1);
45:      _delay_ms(10);
46:     lcd_data(ch2);
47:      _delay_ms(10);
48:     lcd_data(ch3);
49:     _delay_ms(10);
50:     lcd_data('.');
51:       _delay_ms(10);
52:     lcd_data(ch4);
53:      _delay_ms(10);
54:     lcd_data(ch5);
55:      _delay_ms(10);
56:     lcd_data(ch6);
57:     _delay_ms(10);
58:
59: }
```

Ζήτημα 7.2.c:

Για να τυπώσουμε την τιμή που μας δίνει ο αισθητήρας χρησιμοποιούμε την συνάρτηση που φτιάξαμε (find_temp) κι να αποθηκεύσουμε την τιμή στην μεταβλητή x Έπειτα θα χρειαστεί να θέσουμε το PortD ως output και να ενεργοποιήσουμε τα pull down. Οι συναρτήσεις που χρησιμοποιούμε για την οθόνη είναι οι ίδιες με αυτές των προηγούμενων σειρών ασκήσεων εκτός της LCD_print. Η LCD_print διαβάζει την τιμή x και αν είναι ίση με 0x800 τότε τυπώνει το μήνυμα “NO Device”, αλλιώς παίρνει την τιμή και την χωρίζει σε δεκάδες και τις τυπώνει. Αν η τιμή είναι αρνητική τότε τυπώνει το “ - ” μπροστά από τον αριθμό.

```
249:   int main()
250:   {
251:       lcd_init();
252:
253:       while(1) {
254:          int x =  find_temp();
255:          DDRD |= 0b11111111;  // output for LCD
256:          PORTD=0;
257:
258:          lcd_command(0x01);
259:          _delay_ms(30);
260:          LCD_print(x) ;
261:          _delay_ms(1000);
262:          }
263:   }
264:
```