2023

Αναφορά 7ης Εργαστηριακής Άσκησης

Εργαστήριο Μικροϋπολογιστών

ΠΑΠΑΔΟΠΟΥΛΟΣ ΣΠΥΡΙΔΩΝ (ΑΜ): 03120033
ΕΜΜΑΝΟΥΗΛ ΞΕΝΟΣ (ΑΜ):03120850

Ζήτημα πρώτο

Στο ζήτημα αυτό υλοποιήθηκε συνάρτηση σε γλώσσα C για την επικοινωνία με τον αισθητήρα θερμοκρασίας DS18B20 για την λήψη της 16bit τιμής που μέτρησε. Η συνάρτηση που υλοποιείται σε αυτό το ζήτημα θα αποθηκεύει τα 2 bytes που προέρχονται από την μέτρηση της θερμοκρασίας του DS18B20 σε δύο global 8bit μεταβλητές high και low καθώς και στους καταχωρητές r24 και r25.

Για την επικοινωνία με τον αισθητήρα DS18B20, μέσω του one wire protocol θα χρησιμοποιηθούν οι συναρτήσεις που μας δίνονται στην θεωρία του φυλλαδίου. Αρχικά θα «μεταφράσουμε» αυτές τις συναρτήσεις από AVR Assembly σε C. Οι συναρτήσεις σε C είναι οι εξής:

```c
uint8_t one_wire_reset() {
  DDRD |= (1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(480);

  DDRD &= ~(1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(100);

  uint8_t input = PIND;
  _delay_us(380);

  return input & (1 << PD4) ? 0 : 1;
}

uint8_t one_wire_receive_bit() {
  DDRD |= (1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(2);

  DDRD &= ~(1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(10);

  uint8_t input = PIND;
  _delay_us(49);
  return input & (1 << PD4) ? 1 : 0;
}

void one_wire_transmit_bit(uint8_t data){
  DDRD |= (1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(2);
```

```
  PORTD|=(data<<PD4);
  _delay_us(58);

  DDRD &= ~(1 << PD4);
  PORTD &= ~(1 << PD4);
  _delay_us(1);
}

uint8_t one_wire_receive_byte(){
  uint8_t i=0;
  uint8_t receive_result=0;
  while(i<8){
    receive_result+= (one_wire_receive_bit()<<i);
    i++;
  }
  return receive_result;
}

void one_wire_transmit_byte(uint8_t data){
  uint8_t i=0;
  while(i<8){
    one_wire_transmit_bit(data & 0x01);
    data=data>>1;
    i++;
  }
}
```

Με βάση λοιπόν τις παραπάνω συναρτήσεις θα γραφεί η συνάρτηση measureTemp() που θα παίρνει την τιμή που μετρήθηκε από τον αισθητήρα DS18B20 και θα την αποθηκεύει στις 8bit global μεταβλητές high και low καθώς και στους καταχωρητές r24 και r25. Όπως προαναφέρθηκε η συνάρτηση αυτή απλά παίρνει την μέτρηση του αισθητήρα και δεν υπολογίζει την θερμοκρασία με βάση την μέτρησή του. Ο υπολογισμός της θερμοκρασία θα γίνει μέσω συνάρτησεως που θα υλοποιηθεί στο επόμενο ζήτημα.

Ακολουθεί η συνάρτηση measureTemp() καθώς και η εξήγησή της.

```
uint8_t low, high;

void measureTemp(){
  if(!one_wire_reset()){
    high=0x080;
    low=0x00;
    __asm__ __volatile__(
    "mov r24, %1" "\n\t"
    "mov r25, %0" "\n\t"
    :
    : "r" (high), "r" (low)
    );
    return;
```

```
  }

  one_wire_transmit_byte(0xCC);

  one_wire_transmit_byte(0x44);

  while(one_wire_receive_bit()!=1);

  if(!one_wire_reset()){
    high=0x80;
    low=0x00;
    __asm__ __volatile__(
    "mov r24, %1" "\n\t"
    "mov r25, %0" "\n\t"
    :
    : "r" (high), "r" (low)
    );
    return;
  }



  one_wire_transmit_byte(0xCC);

  one_wire_transmit_byte(0xBE);

  low=one_wire_receive_byte();
  high=one_wire_receive_byte();
  __asm__ __volatile__(
    "mov r24, %1" "\n\t"
    "mov r25, %0" "\n\t"
    :
    : "r" (high), "r" (low)
    );

}
```

Στην αρχή της συνάρτησης γίνεται η αρχικοποιήση του προτοκόλου one wire μέσω της συνάρτησης one_wire_reset και γίνεται έλεγχος αν υπάρχει συνδεδεμένη συσκευή. Σε περίπτωση που δεν υπάρχει συνδεδεμένη συσκευή τότε γράφονται στις global μεταβλητές high και low οι τιμές 0x80 και 0x00 αντίστοιχα (καθώς και στους καταχωρητές r24, r25). Αυτή η τιμή συμβολίζει την περίπτωση που δεν υπάρχει συσκευή συνδεδεμένη στον δίαυλο. Αφού ελέγξουμε ότι υπάρχει συσκευή συνδεδεμένη στέλνεται η εντολή 0xCC, μέσω της συναρτήσεως one_wire_transmit_byte(), που παρακάμπτει την επιλογή συσκευής στον διάδρομο καθώς στην πλακέτα NtuAboard_G1,καθώς υπάρχει μόνο μία συνδεδεμένη συσκευή στο δίαυλο επικοινωνίας (το θερμόμετρο). Στην συνέχεια στέλντεται, με την χρήση της one_wire_transmit_byte() η εντολή 0x44 η οποία ξεκινάει την μέτρηση της θερμοκρασίας στον αισθητήρα DS18B20. Έπειτα περιμένουμε μέχρι να λάβουμε bit με τιμή 1, η λήψη bit

γίνεται μέσω της εντολής one_wire_receive_bit(), από τον διάυλο one wire που σηματοδοτεί ότι ολοκληρώθηκε η μέτρηση της θερμοκρασίας από τον αισθητήρα DS18B20. Αφού ολοκληρώθηκε η μέτρηση του προαναφερθέντος αισθητήρα αρχικοποιούμε και πάλι το προτόκολο one wire (ελέγχοντας ξανά αν υπάρχει συσκευή σε περίπτωση που αποσυνδέθηκε), στέλνουμε και πάλι την εντολή 0xCC για την παράκαμψη επιλογής συσκευής και στην συνέχεια στέλνεται η εντολή 0xBE που μας επιτρέπει να διαβάσουμε την 16bit μέτρηση του αισθητήρα. Η λήψη της 16bit μέτρησης γίνεται μέσω της εντολής one_wire_receive_byte() και διαβάζεται πρώτα το χαμηλό (low) byte και έπειτα το high byte. Τέλος χρησιμοποιώντας την εντολή του compiler __asm__ __volatile__() για να γράψουμε την τιμή των μεταβλητών high και low στους καταχωρητές r25 και r24 αντίστοιχα.

Ζήτημα δεύτερο

Στο ζήτημα αυτό μας ζητήθηκε να λάβουμε την έξοδο που μας δίνει η συνάρτηση measureTemp() που υλοποιήσαμε στο πρώτο ζήτημα και να προβάλουμε την θερμοκρασία στο lcd display της πλακέτας. Επιπλέον όταν δεν συνδέεται καμία συσκευή στον δίαυλο σειριακής επικοινωνίας μας ζητείται στο display να γράφει “No Device”. Ακολούθως παραθέτουμε τον κώδικα μας για το πρόβλημα αυτό:

```
void lcd_temp(){
    int temperature;
    lcd_clear_display();
    measureTemp();
    temperature= high; // assign high to temp to left shift it 8 times
left
    temperature= (low | temperature<<8); // the temperature is high,low
    if(temperature==0x8000){            // if temperature=0x8000 no
device
                                        // is connected to the bus
        lcd_string("No Device");
        _delay_ms(1000);
        return ;
    }
    if((temperature & 0xF800) == 0xF800 ){  // check if temperature is
negative
        temperature=~temperature+1; // if temperature is negative find
the compliment of 2
        lcd_data('-');              // and output a - to indicate the
negative value
    }
    double temp;
    temperature=temperature & 0x07FF; // keep only the bits needed
    temp=temperature;
    temp*=0.0625;   // temperature is calculated with a step of 0.0625
```

```
                    // that means that the basic step of a binary
number instead of 1
                    // becomes 0.0625
    char temperature_str[10];
    dtostrf(temp, 2, 3, temperature_str); // Convert temperature to
string with 2 decimal places
    lcd_string(temperature_str); // output the temperature
    lcd_data(0xDF); // output the degree symbol
    lcd_data('C');  // symbol for Celsius
    _delay_ms(1000); // delay in order for the output to be visible
}

int main(){
    twi_init();
    PCA9555_0_write(REG_CONFIGURATION_0, 0x00); //Set EXT_PORT0 as
output
    lcd_init();
    while(1){
        lcd_temp();
    }
}
```

Αρχικά στο πρόγραμμα μας αρχικοποιούμε την επικοινωνία μέσω του πρωτοκόλλου twi με το ολοκληρωμένο PCA9555 προκειμένου να χρησιμοποιήσουμε το IO0 σαν είσοδο του lcd display μέσω του EXT_PORT0. Στην συνέχεια θέτουμε το IO0 σαν έξοδο ώστε να γράφουμε σε αυτό την είσοδο του display και αρχικοποιούμε και το ίδιο το display. Εφόσον, μας ζητείται πρόγραμμα συνεχής λειτουργίας εκτελούμε ένα infinite loop που σε κάθε επανάληψη καλεί την lcd_temp() , η οποία υλοποιεί την ζητούμενη λειτουργία. Η lcd_temp αρχικά καθαρίζει το display, εφόσον η ίδια θα γράψει σε αυτό, και καλεί την measureTemp () για να λάβει την μέτρηση της θερμοκρασίας. Η μέτρηση της θερμοκρασίας αποθηκεύεται σε δύο θέσεις μνήμης από την measureTemp () (δηλαδή δύο global μεταβλητές) που λέγονται high και low και υποδηλώνουν το high byte και το low byte της θερμοκρασίας αντίστοιχα. Στην συνέχεια αποθηκεύουμε σε μία 16-bit μεταβλητή το high byte το κάνουμε 8 shift αριστερά (ώστε να πάρει την σωστή θέση στην 16-bit μεταβλητή) και μετά πραγματοποιούμε ένα λογικό or με το low ώστε να πάρουμε στην μεταβλητή temperature την έξοδο που δίνει ο μετρητής της θερμοκρασίας. Στην συνέχεια ελέγχουμε εάν το temperature είναι ίσο με 0x8000. Στην περίπτωση που είναι ίσα τότε γράφουμε στο display “No Device” και εκτελούμε ένα delay 1sec και η συνάρτηση επιστρέφει. Εάν δεν ισχύει η ισότητα τότε το temperature έχει την μέτρηση της θερμοκρασίας. Πρώτα ελέγχουμε το πρόσημο της θερμοκρασίας μέσω των πρώτων 5 bit του temperature. Αν η θερμοκρασία είναι αρνητική τα πρώτα 5 bit θα είναι όλα 1. Σε αυτή την περίπτωση βρίσκουμε το συμπλήρωμα ως προς 2 του temperature (αυτό θα είναι η απόλυτη τιμή της θερμοκρασίας μας) και τυπώνουμε ένα – για να δηλώσουμε την αρνητική θερμοκρασία. Μετά τον έλεγχο αυτό ξέρουμε ότι το temperature έχει θετική τιμή. Αποθηκεύουμε την θερμοκρασία σε ένα float και αλλάζουμε ουσιαστικά την βάση του αριθμού από 1 σε 0.0625 πολλαπλασιάζοντας το temp με 0.0625. Πλέον το temp έχει την θερμοκρασία που θέλουμε να δείξουμε. Άρα μετατρέπουμε το temp σε string μέσω της dtostrf και γράφουμε το string αυτό στο display μέσω της lcd_string() . Επίσης γράφουμε στο

display το σύμβολο˚ και το C (που υποδηλώνει πως ο αριθμός είναι βαθμοί κελσίου). Αυτή είναι η υλοποίηση μας για το ζήτημα 2.

Σε αυτό το πλαίσιο αξίζει να σημειώσουμε ότι το πρωτόκολλο επικοινωνίας one-wire που χρησιμοποιήσαμε σε αυτή την άσκηση είναι σημαντικά πιο αργό από το twi που χρησιμοποιήσαμε σε προηγούμενη άσκηση. Αυτό οφείλεται στο γεγονός ότι το 1-wire έχει μόνο ένα δίαυλο για επικοινωνία και συνεπώς όλος ο συγχρονισμός πρέπει να γίνει χρησιμοποιώντας αυτό τον δίαυλο και καθυστερήσεις, σε αντίθεση με το twi όπου ο συγχρονισμός πραγματοποιούταν από τον SCL δίαυλο.