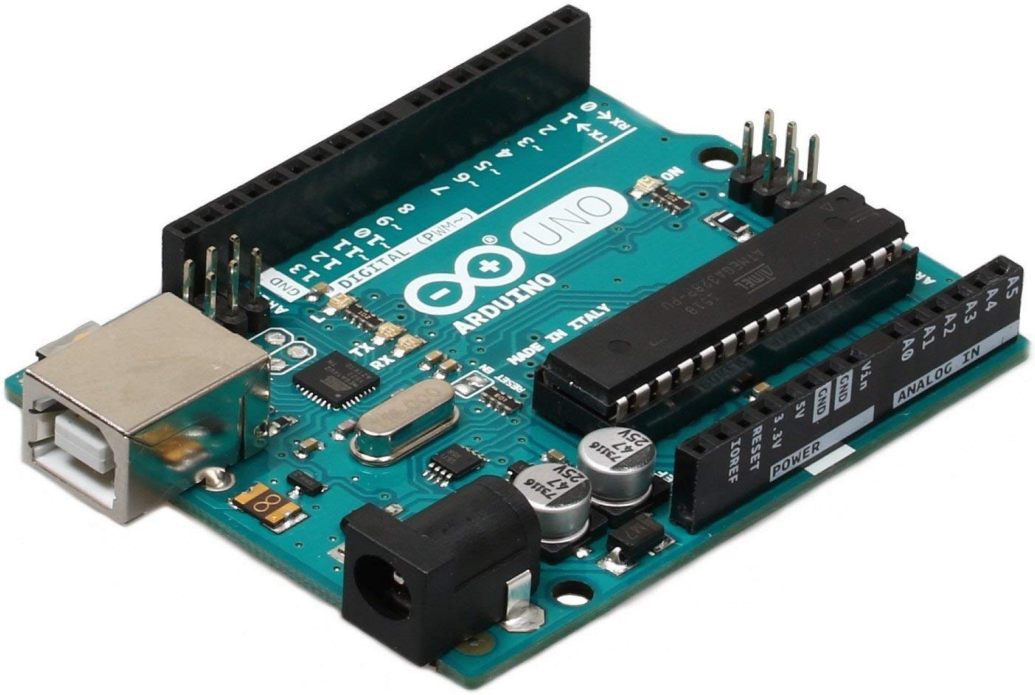
Arduino Project

Υλοποίηση έξυπνου θερμοστάτη με αισθητήρα κίνησης



(στο πλαίσιο του μαθήματος Μικροεπεξεργαστές και Περιφερειακά)

Ιάσων-Δημήτρης Καλαμποκίδης 8953

Δημήτρης Μουτζόγλου 8319

Αντικείμενο Εργασίας:

Ζητούμενο της εργασίας ήταν να υλοποιήσουμε έναν έξυπνο θερμοστάτη με αισθητήρα κίνησης ο οποίος θα συλλέγει μετρήσεις για την θερμοκρασία του χώρου και θα τις εμφανίζει μέσω μίας οθόνης LCD. Πιο συγκεκριμένα , ο αισθητήρας θερμοκρασίας καταγράφει μία θερμοκρασία για διάστημα 5 δευτερολέπτων ενώ παράλληλα υπολογίζει και μία μέση θερμοκρασία για διάστημα 2 λεπτών (120 δευτερολέπτων , 24 μετρήσεις). Σε περίπτωση που ο αισθητήρας καταγράψει ακραία τιμή θερμοκρασίας, ακραίες τιμές που ορίσαμε στον κώδικα τους 0°C και 40°C, αυτόματα εμφανίζει προειδοποιητικό μήνυμα στην οθόνη LCD ενώ παράλληλα ανάβουν τα κατάλληλα LEDs (κόκκινο - πράσινο για υψηλή θερμοκρασία, κυανό για χαμηλή). Αν οι μετρήσεις κυμαίνονται σε κανονικά πλαίσια το σύστημα υπολογίζει ανά 2 λεπτά την μέση θερμοκρασία και την παρουσιάζει. Εξαίρεση στη ροή δημιουργείται όταν ο αισθητήρας κίνησης αντιληφθεί δραστηριότητα γύρω του και αυτόματα εμφανίζεται στην οθόνη μήνυμα με την τελευταία μέτρηση της θερμοκρασίας και την πιο πρόσφατη μέση τιμή. Η ιεραρχία εμφάνισης μηνυμάτων ορίζεται ως εξής : 1) προειδοποιητικό μήνυμα για ακραία τιμή θερμοκρασίας , 2) τελευταία μέτρηση και μέση τιμή σε περίπτωση κίνησης γύρω από τον αισθητήρα, 3) μέση τιμή ανά χρονικό διάστημα 2 λεπτών. Το σύστημα αποτελείται από τον μικροεπεξεργαστή Wemos D1R2 , τον αισθητήρα εγγύτητας HC-SR04 Ultrasonic, τον αισθητήρα θερμοκρασίας DHT11, μία LCD 16X02 με ενσωματωμένο το module I2C, 3 LEDS ενώ η διασύνδεση έγινε σε breadboard.

Δυσκολίες που παρουσιάστηκαν:

1) Εξαιτίας της χρήσης του μικροεπεξεργαστή Wemos αντί του Arduino , απαιτούνταν "μετάφραση" των χρησιμοποιούμενων pins κατά την διαδικασία συνδεσμολογίας καθώς η βιβλιογραφία στο internet αφορά κατά βάση υλοποίηση σε arduino.

2) Λόγω της αρχικής έλλειψης του module I2C η σύνδεση της οθόνης LCD κατέστη αδύνατη εξαιτίας αστάθειας των pins ενώ αδυνατούσαμε να εμφανίσουμε μήνυμα σε αυτήν. Το πρόβλημα λύθηκε με την αγορά οθόνης LCD με ενσωματωμένο το απαραίτητο module που εξοικονομεί pins στον μικροεπεξεργαστή (υπήρχε σχετικό conflict με τον αισθητήρα εγγύτητας).

3) Η βιβλιογραφία όσον αφορά τον κώδικα στο internet αφορά υλοποίηση σε arduino οπότε απαιτούνταν μία πιο εξειδικευμένη αναζήτηση για την εύρεση των κατάλληλων βιβλιοθηκών.

4) Καταφύγαμε στην χρήση δεύτερου LED για σήμανση ακραίας θερμοκρασίας λόγω έλλειψης ρελέ.

5) γράψε τπτ άμα θες για την blinker ή κάποιο κομμάτι που κολλήσατε στον κώδικα

Ροή κώδικα:

```
#include <Ticker.h>  //Ticker Library
#include "DHTesp.h"
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C  lcd(0x27, 2, 1, 0, 4, 5, 6, 7);
DHTesp dht;

int echoPin = 13;
int trigPin = 12;
int t1 = 0, t2 = 0;
Ticker blinker;
```

Αρχικά, δηλώνουμε τις απαραίτητες βιβλιοθήκες για να μπορούμε να χρησιμοποιήσουμε τις συναρτήσεις που απαιτούνται για την διαχείριση των δεδομένων των αισθητήρων. Η **ticker.h** επιτρέπει την χρήση του timer μας , η **DHTesp.h** αφορά τον αισθητήρας θερμοκρασίας , η **LiquidCrystal_I2C.h** και η **Wire.h** απαιτούνται για την χρήση της LCD οθόνης. Έπειτα, δημιουργούμε το αντικείμενο dht για τον αισθητήρα και το lcd για την οθόνη, με το πρώτο όρισμα του constructor να αναφέρεται στην διεύθυνση της οθόνης . Στη συνέχεια ορίζουμε τις μεταβλητές echoPin, trigPin, t1,t2 για την χρήση του αισθητήρα εγγύτητας και την μέτρηση των θερμοκρασιών ενώ ***δηλώνουμε και ένα αντικείμενο blinker για τον timer.***

```
int changeState()
{
  t1++;
}

void setup()
{ pinMode(D6, OUTPUT);
  pinMode(D3, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  lcd.begin (16, 2);
  lcd.setBacklightPin(3, POSITIVE);
  lcd.setBacklight(HIGH);
  Serial.begin(115200);
  blinker.attach(5, changeState);
}
```

Ακολούθως , ορίζουμε την συνάρτηση changeState() που χρησιμοποιείται σε συνδυσμό με τον timer για την μέτρηση της θερμοκρασίας κάθε 5 δευτερόλεπτα και σειρά έχει το setup με την δήλωση των pins3,4,6 ως έξοδοι για τα LEDs, του trigPin ως εξόδου και του echoPin ως εισόδου για την καταγραφή κίνησης γύρω από τον αισθητήρα μέσω της αποστολής και λήψης σήματος, την παραμετροποίηση των ρυθμίσεων της LCD, τον καθορισμό του BAUD στα 115200, ***και την χρήση του timer blinker ανά 5 δεύτερα.***

```
void loop()
{
  int counter = 0;
  int distance;
  float avg = 0;
  float t;
  float sum = 0;
  int flag1 = 0,flag2=0;
  double start_time = 0;
  dht.setup(D5, DHTesp::DHT11);
```

Στη συνέχεια έχουμε τον κύριο βρόγχο του προγράμματος με την αρχικοποίηση του counter με τιμή μηδέν για την εμφάνιση της μέσης τιμής της θερμοκρασίας αργότερα στον κώδικα, της avg και του sum για τον υπολογισμό της μέσης τιμής, της t για την θερμοκρασία , των flag1, flag2 για την λάμψη των LEDs στην περίπτωση παρατήρησης ακραίας θερμοκρασίας, της distance για την μέτρηση

ΑΥΤΟ ΕΙΝΑΙ ΤΗΣ ΥΠΑΤΙΑΣ , ΚΑΝΕ ΜΕΤΑΤΡΟΠΗ/ΜΕΤΑΦΡΑΣΗ

Στη συνέχεια ξεκινάει ο κύριος βρόχος του προγράμματος μας loop().Στην μεταβλητή *time* αποθηκεύεται η μέτρηση του αισθητήρα κίνησης. Ο αισθητήρας ουσιαστικά μετράει την ώρα που έκανε το σήμα που έστειλε για να ταξιδέψει μέχρι να επιστρέψει στον δέκτη του αισθητήρα , και έτσι υπολογίζεται η απόσταση του από το εμπόδιο, αν λάβουμε υπόψιν ότι το σήμα ταξιδεύει με την ταχύτητα του φωτός(γνωστός τύπος x=C*t).Έχοντας υπόψην ότι θέλουμε να ενημερωθούμε σε περίπτωση που κάτι πλησιάσει το σύστημα πάνω από ένα μέτρο, υπολογίζουμε από τον τύπο x=U*t αυτό συμβαίνει όταν η μεταβλητή time παίρνει τιμή κάτω από 3000.

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
distance = pulseIn(echoPin, HIGH);
```

Το παραπάνω κομμάτι κώδικα αφορά τον χειρισμό του αισθητήρα εγγύτητας για τον έλεγχο κίνησης γύρω του αισθητήρα. Θέτοντας εναλλάξ από LOW σε HIGH την τιμή του trigPin(pin εξόδου) για διάστημα 10 μsec διερευνούμε για κίνηση και αποθηκεύουμε την τιμή του παλμού echoPin, μέσω της pulseIn() στη μεταβλητή distance.

```
if (distance < 3000) {
  if (flag1 == 0) {
    flag1 = 1;
    start_time = millis();
    Serial.println("open screen");
    lcd.home();
    lcd.print("Temperature=");
    lcd.print(t);
    lcd.setCursor(0, 1);
    lcd.print("Average=");
    lcd.print(avg);
  }
}


   if (flag1 == 1 && millis() - start_time > 10000) {
     Serial.println("close screen");
     flag1 = 0;
     lcd.clear();
     //turn off screen
   }

   if (counter == 24) {
     counter = 0;
     avg = sum / 24;
     sum = 0;
     if(flag1==0){
     flag1 = 1;
       start_time = millis();
       Serial.println("open screen");
       lcd.home();
       lcd.print("Average=");
       lcd.print(avg);
     }
   }
 }
}
```

Στους παραπάνω if ελέγχους, αρχικά εξετάζουμε αν η τιμή της distance είναι μικρότερη των 3000 (ουσιαστικά αν κάποιος πλησίασε τον αισθητήρα σε απόσταση μικρότερη του ενός μέτρου). Σε περίπτωση αλήθειας εμφανίζουμε στην οθόνη LCD την τελευταία μέτρηση και την τελευταία καταγεγραμμένη μέση τιμή.

Με την χρήση της συνάρτησης millis() και αλλεπάλληλων ελέγχων εξασφαλίζουμε πως το μήνυμα θα εμφανιστεί στην οθόνη για διάστημα 10 δευτερολέπτων. Στον επόμενο if έλεγχο , εξετάζουμε αν η τιμή του counter είναι ίση με 24 δηλαδή αν προέκυψε μέση τιμή για το παρελθόν δίλεπτο. Σε περίπτωση αλήθειας υπολογίζουμε την μέση τιμή , μηδενίζουμε το sum και τον counter και εμφανίζουμε την μέση τιμή για διάστημα 10 δευτερολέπτων στην οθόνη LCD.

```
16:52:07.803 -> 31.00
16:52:12.811 -> 31.00
16:52:17.821 -> 31.00
16:52:22.859 -> 31.00
16:52:27.821 -> 31.00
16:52:32.818 -> 31.00
16:52:37.781 -> 31.00
16:52:42.791 -> 31.00
16:52:45.978 -> open screen
16:52:47.792 -> 31.00
16:52:52.802 -> 31.00
16:52:55.982 -> close screen
16:52:57.810 -> 31.00
16:53:02.788 -> 31.00
16:53:07.796 -> 31.00
16:53:12.805 -> 31.00
16:53:17.815 -> 31.00
16:53:22.932 -> 31.00
16:53:27.820 -> 31.00
16:53:32.815 -> 31.00
16:53:37.806 -> 31.00
16:53:42.839 -> 31.00
16:53:47.819 -> 31.00
16:53:52.817 -> 31.00
16:53:52.817 -> open screen
16:53:57.781 -> 31.00
16:54:02.818 -> 31.00
16:54:02.818 -> close screen
16:54:07.812 -> 31.00
16:54:12.782 -> 31.00
16:54:17.810 -> 31.00
16:54:22.821 -> 31.00
16:54:27.791 -> 31.00
```

Στο παραπάνω στιγμιότυπο της κονσόλα του arduino IDE, βλέπουμε την τακτική μέτρηση των θερμοκρασιών σε διάστημα 5

δευτερολέπτων, στο διάστημα 16:52:45-16:52:55 έχουμε την εμφάνιση της μέσης τιμής της οθόνης και της τελευταίας μέτρησης λόγω κίνησης ενώ με την συμπλήρωση δύο λεπτών έχουμε την εμφάνιση της μέσης για χρονικό διάστημα 10 δευτερολέπτων.