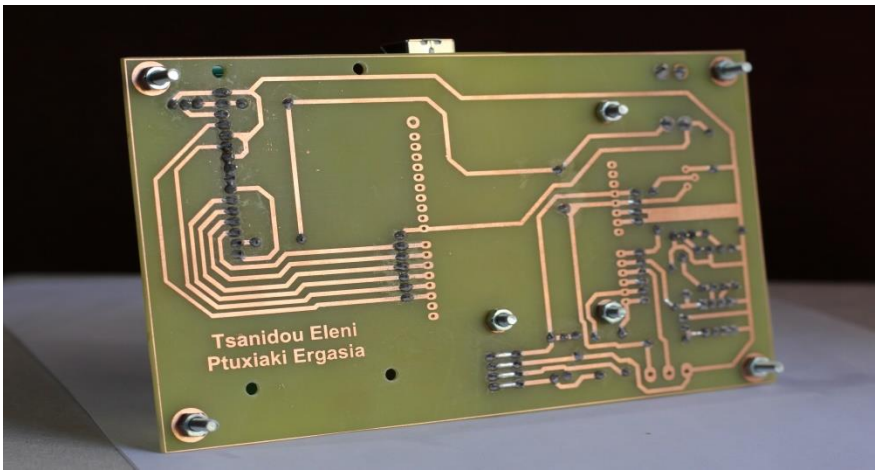
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

# ΘΕΜΑ : ΥΛΟΠΟΙΗΣΗ ΜΕΤΕΩΡΟΛΟΓΙΚΟΥ ΣΤΑΘΜΟΥ ΜΕ ARDUINO ΚΑΙ RASPBERRY PI




ΣΠΟΥΔΑΣΤΡΙΑ : ΤΣΑΝΙΔΟΥ ΕΛΕΝΗ, ΑΕΜ : 3542

ΥΠΕΥΘΥΝΟΣ ΚΑΘΗΓΗΤΗΣ : ΚΑΛΟΜΟΙΡΟΣ ΙΩΑΝΝΗΣ

ΣΕΡΡΕΣ, ΜΑΪΟΣ 2019

# ΠΕΡΙΕΧΟΜΕΝΑ

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

## ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ

Βεβαιώνω ότι εγώ η Τσανίδου Ελένη είμαι η συγγραφέας αυτής της πτυχιακής εργασίας και κάθε υποστήριξη ή βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται μέσα σε αυτήν. Επίσης, σε αυτή την πτυχιακή εργασία αναφέρονται όλες οι πηγές από τις οποίες έκανα χρήση δεδομένων και πληροφοριών. Τέλος, βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά και ειδικά για τις αναγκαίες απαιτήσεις του προγράμματος σπουδών του Τμήματος Μηχανικών Πληροφορικής του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Σερρών.

# Περίληψη

Σκοπός της εργασίας αυτής είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος μετεωρολογικού σταθμού με τον μικροελεγκτή Arduino και το Raspberry Pi. Στην εργασία αυτή υπάρχει λεπτομερής περιγραφή για τον τρόπο και την λειτουργία του μετεωρολογικού σταθμού. Απώτερος στόχος είναι να μπορεί να εγκατασταθεί σε οποιοδήποτε απομακρυσμένο χώρο και τα αποτελέσματά του να είναι ορατά είτε στην οθόνη του συστήματος είτε ακόμα και μέσω ενός υπολογιστή.

Συνδέοντας το σύστημα σε τροφοδοσία μπορεί να εμφανίζει τις μετρήσεις που παίρνει για τα καιρικά φαινόμενα στην οθόνη της κατασκευής ή και στην οθόνη ενός υπολογιστή. Τέλος, όλα τα δεδομένα που λαμβάνονται ανεβαίνουν σε μια διαδικτυακή βάση στην οποία έχει πρόσβαση ο καθένας, έφοσον γνωρίζει τα στοιχεία που απαιτούνται για την σύνδεση στον λογαριασμό. Οι μετρήσεις που ανεβαίνουν στο διαδίκτυο, παρέχονται στους χρήστες σε μορφή διαγράμματος.

Το σύστημα υλοποιήθηκε σε τυπωμένη πλακέτα δικής μας κατασκευής, όπου ενσωματώθηκαν τα ηλεκτρονικά εξαρτήματα και μια ιδιοκατασκευή ενός αισθητήρα για την μέτρηση της ταχύτητας και του προσανατολισμού του ανέμου.

Η εργασία αυτή απέδωσε σημαντικά εκπαιδευτικά αποτελέσματα, καθώς βοήθησε στην εξοικείωση με τον μικροελεγκτή Arduino, τον υπολογιστή μονής κάρτας Raspberry Pi, με αρκετούς αισθητήρες, πρωτόκολλα επικοινωνίας και λογισμικό.

# 1. Εισαγωγή

Η Μετεωρολογία είναι η επιστήμη η οποία μελετά την ατμόσφαιρα και τα φαινόμενα τα οποία συμβαίνουν μέσα σ' αυτήν. Παράλληλα, μέσα στο χώρο των εφαρμοσμένων επιστημών, η Μετεωρολογία, επιδιώκει την απόλυτη κατανόηση και την ακριβή πρόβλεψη των ατμοσφαιρικών φαινομένων , τα οποία τελικά είναι υπεύθυνα για τις καιρικές συνθήκες που επικρατούν σε κάθε σημείο του πλανήτη, οποιαδήποτε στιγμή της ημέρας (auth).

Ο μετεωρολογικός σταθμός είναι ένα σύστημα το οποίο μέσω κατάλληλων αισθητηρίων μπορεί να καταγράψει χρήσιμα δεδομένα όπως η θερμοκρασία, η υγρασία, η βαρομετρική πίεση, η ταχύτητα και η κατεύθυνση του ανέμου καθώς και η ατμοσφαιρική κατακρήμνιση (Λάτμος, 2017).

Στη παρούσα εργασία αναλύθηκε ένας τρόπος δημιουργίας μετεωρολογικού σταθμού με έναν μικροελεγκτή μονής πλακέτας, το Arduino, και έναν υπολογιστή μικρού μεγέθους, το Raspberry PI. Συνδέθηκαν κατάλληλα οι απαραίτητοι αισθητήρες, έτσι ώστε να ληφθούν αναλογικά και ψηφιακά τα σωστά καιρικά στοιχεία, όπως θερμοκρασία, υγρασία, ταχύτητα ανέμου και προσανατολισμός του, υψόμετρο και ατμοσφαιρική πίεση.

Για την ολοκλήρωση του εν λόγω σταθμού χρησιμοποιήθηκαν το Arduino/Genuino, μια οθόνη LCD για την εμφανίση των ενδείξεων, ένα κουμπί για εναλλαγή των ενδείξεων της οθόνης, 2 ποντενσιόμετρα των 10K, αντιστάσεις των 10K, 150K, 2K7, 100K και 1M, αντιστάσεις R0 για το πέρασμα του χαλκού, ένας πυκνωτής 100nF, η δίοδος 1N4145, μια δίοδος Zenner 5V1, το ολοκληρωμένο LM358, οι αισθητήρες BMP180, HIH6120, TMP36 και το Raspberry Pi.

Τα δεδομένα που συλλέχθηκαν, εμφανίζονται σειριακά στο περιβάλλον ανάπτυξης του Arduino με δύο τρόπους: ή πατώντας το γράμμα «A» στο σειριακό παράθυρο του προγράμματος, ή πατώντας το κουμπί της πλακέτας που επιτρέπει στα στοιχεία να περνούν στην οθόνη. Όλα αυτά τα στοιχεία μέσω του raspberry pi ταξινομήθηκαν σε ημερήσιους φακέλους, δημιουργώντας δηλαδή, φακέλους την μέρα που πραγματοποιήθηκαν μετρήσεις. Σε περίπτωση πολλαπλών μετρήσεων σε μια μέρα, τα στοιχεία τοποθετούνται στον ίδιο φάκελο. Επίσης, το raspberry pi προγραμματίστηκε μέσω της Python με τέτοιο τρόπο ώστε το πάτημα του πλήκτρου «a» να επιτρέπει το διάβασμα των ονομάτων των φακέλων. Αυτοί εμφανίζονται με τυποποιημένη ονομασία ημερομηνίας, τύπου Meteo_ημερομηνία στην ολοκληρωμένη μορφή "yyyymmdd"(4 ψηφία για την χρονολογία, 2 ψηφία για την ένδειξη του μήνα και 2 για την μέρα). Εν συνεχεία, εμφανίζεται το μήνυμα που επιτρέπει την επιλογή του επιθυμητού φακέλου βάσει της ονομασίας που έχει δωθεί. Σε περίπτωση λάθους εμφανίζεται το αντίστοιχο μήνυμα.

Ταυτόχρονα, αυτά τα δεδομένα ανεβαίνουν σε μια ελεύθερη βάση την thingspeak, όπου γίνονται και τα αντιστοίχα διαγράμματα. Αυτό εξυπηρετεί στη σύγκριση των δεδομένων αλλά και στην συγκέντρωση όλων των στοιχείων με μορφή διαγράμματος. Επιλέχθηκε η συγκεκριμένη βάση γιατί είναι ελευθέρη στο διαδίκτυο, δεν απαιτεί ιδιαίτερο προγραμματισμό, ενώ παράλληλα η κατασκευή των διαγραμμάτων γίνεται πιο αυτόματα. Υπάρχουν και άλλες ελεύθερες βάσεις, όπως phpMyAdmin, MySQL, PostgreSQL, Microsoft SQL Server, Firebird, MongoDB, MariaDB, SQLAlchemy.

Μια τέτοια κατασκευή μπορεί να αποδειχθεί εξαιρετικά χρήσιμη σε ανθρώπους των οποίων το επάγγελμα σχετίζεται με τις εκάστοτε καιρικές συνθήκες, π.χ. αγρότες, αλλά και σε όλες τις περιπτώσεις που θέλει κανείς να γνωρίζει τα συγκεκριμένα καιρικά δεδομένα.

# 2. Περιγραφή Στοιχείων Διάταξης

Σε αυτό το κεφάλαιο παρουσιάζονται όλες οι πληροφορίες που θα πρέπει κανείς να γνωρίζει για τα υλικά που χρησιμοποιήθηκαν, για τον τρόπο που συνδέθηκαν μεταξύ τους, για τον εκάστοτε προγραμματισμό που συγγράφηκε και για όσα μέρη κατασκευάστηκαν.

## 2.1. ARDUINO UNO

### 2.1.1 Περιγραφή Μικροελεγκτή

Ο μικροελεγκτής Arduino αποτελείται από έναν μικροεπεξεργαστή, τον ATmega328P της Atmel. Μπορεί να τροφοδοτηθεί μέσω:

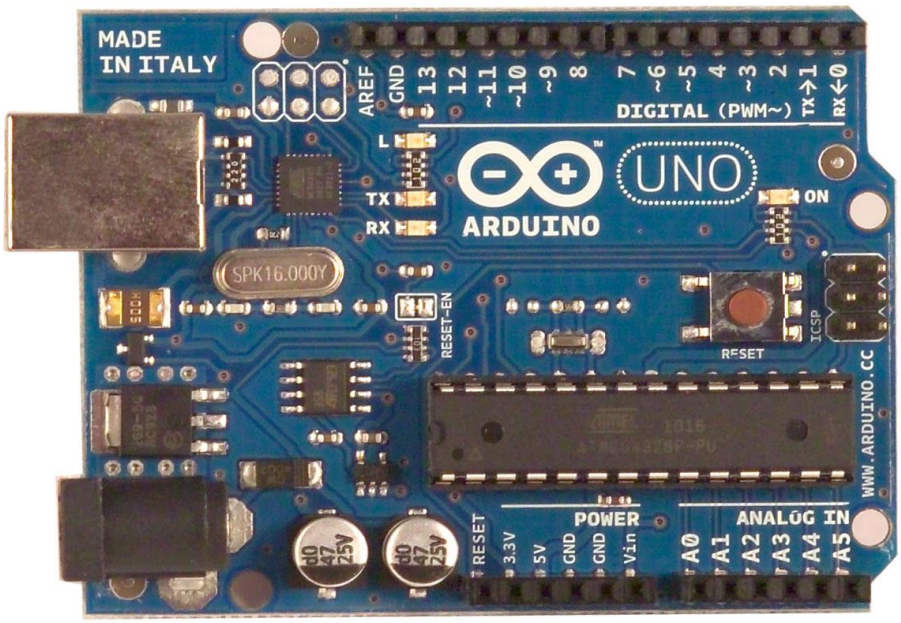
- εξωτερικής πηγής ρεύματος με λειτουργία τάσης από 7 V εώς 12V, ώστε να μπορεί τελικά να δώσει σταθεροποιημένα 5V για την λειτουργία του μικροεπεξεργαστή και των λοιπών κυκλωμάτων,
- μπαταρίας,
- ή ακόμα και μέσω της σειριακής σύνδεσης του USB με τον υπολογιστή.

Έχει 14 ψηφιακές θύρες εισόδου ή εξόδου (digital input/outpit pins) και έξι αναλογικές εισόδους (analog input pins).

- Οι ψηφιακές θύρες απεικονίζονται με νούμερα από το 0 έως το 13,
- ενώ οι αναλογικές με το γράμμα Α το οποίο ακολουθείται από ένα νούμερο από 0 μέχρι το 5 (π.χ. A3).

Από τις 14 ψηφιακές θύρες ιδιαίτερη σημασία έχουν οι θύρες με τους αριθμούς 3, 5, 6, 9, 10, 11, διότι χαρακτηρίζονται ως θύρες PWM (Pulse Width Modulation), δηλαδή μπορούν να προσομοιώσουν αναλογικές εξόδους. Οι ψηφιακές θύρες είτε είναι είσοδοι είτε έξοδοι, μπαίνουν σε λειτουργία με την τάση να βρίσκεται στα 0V ή στα 5V με την λέξη-κλειδί LOW ή HIGH αντίστοιχα (Πουλάκης, 2015).

Ο μικροελεγκτής προγραμματίζεται από τον υπολογιστή μέσω της σειριακής θύρας του Arduino η οποία συνδέεται με τη θύρα USB του υπολογιστή. Συνδέοντας τον Arduino με τον υπολογιστή επιτυγχάνεται η αμφίδρομη μεταφορά δεδομένων: δηλαδή, το πρόγραμμα του υπολογιστή στέλνει σειριακά έναν κωδικό στην πλακέτα η οποία απαντά επιστρέφοντας ένα πακέτο δεδομένων. Η αρχιτεκτονική Arduino, αποτελείται από ένα σύνολο μικροελεγκτών υλικών συστημάτων που με την κατάλληλη παροχή λογισμικού, αυτοματοποιούν δραστηριότητες που επιθυμούν οι χρήστες (Γρηγορόπουλος, 2017).




**Εικόνα 1: Εμπρόσθια όψη του Arduino**

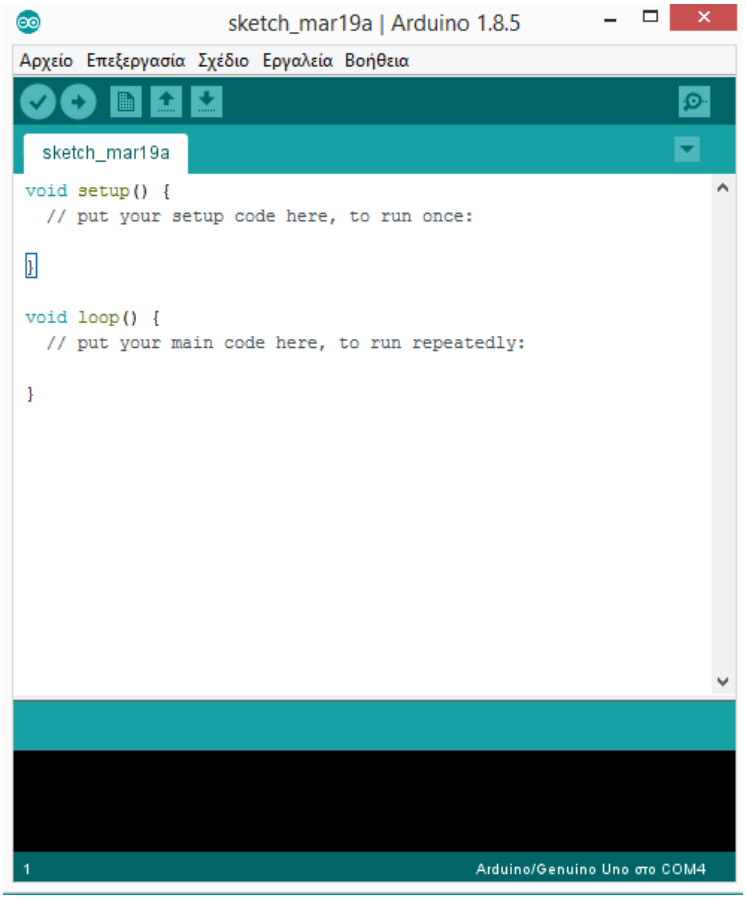
### 2.1.2 Περιβάλλον Ανάπτυξης

Το ολοκληρωμένο περιβάλλον ανάπτυξης του λογισμικού (Integrated Development Environment - IDE) έχει τη βάση του στις γλώσσες προγραμματισμού Processing και Wiring, είναι ανοιχτού κώδικα (open source) και μπορεί κάποιος να το κατεβάσει χωρίς κόστος. Πρόκειται για μια τροποποίηση της γλώσσας προγραμματισμού C/C++. Για τον προγραμματισμό του Arduino χρησιμοποιείται το Arduino IDE που περιέχει το περιβάλλον σύνταξης των εντολών, όπου κάθε νέο αρχείο ονομάζεται αυτόματα “sketch”. Επίσης, περιλαμβάνονται κάποιες έτοιμες βιβλιοθήκες, εκ των οποίων μερικές μπορεί να χρειαστούν ενημέρωση(update), ή αλλιώς μπορεί κάποιος να κατεβάσει μόνο την κατάλληλη βιβλιοθήκη που χρειάζεται για το τρέχον πρόγραμμα, εφόσον βρεθεί στο διαδίκτυο. Σε αυτή την περίπτωση είναι σημαντικό να αποθηκευτεί στον ίδιο φάκελο του “sketch” για να μπορεί να την αναγνωρίσει το πρόγραμμα. Επίσης, έχει compiler (μεταγλωττιστή) για τη μεταγλώττιση του προγράμματος και ένα σειριακό παράθυρο, όπου μπορεί να εμφανίζει την σειριακή επικοινωνία, εφόσον αύτη είναι σε λειτουργία. Για τη σύνταξη του κώδικα υπάρχει μια πρότυπη μορφή :

- η συνάρτηση set up() και
- η συνάρτηση loop().

Πριν τη συνάρτηση set up() δηλώνονται οι μεταβλητές και εισάγονται οι απαραίτητες βιβλιοθήκες που θα χρησιμοποιηθούν μέσα στο πρόγραμμα. Μέσα στην συνάρτηση set up() αρχικοποιούνται οι μεταβλητές και ορίζονται οι ακροδέκτες . Είναι άξιο να σημειωθεί, ότι αυτή η συνάρτηση εκτελείται μόνο μία φορά, επομένως ότι δηλωθεί σε αυτή ισχύει για όλο το πρόγραμμα. Αμέσως μετά, ακολουθεί η συνάρτηση loop() η οποία εκτελεί

διαδοχικές λειτουργίες που επαναλαμβάνονται μέσα στο πρόγραμμα μέχρι να σταματήσει η τροφοδότηση του, ή να πατηθεί το κουμπί reset που υπάρχει πάνω στην πλακέτα.




Εικόνα 2: Περιβάλλον Ανάπτυξης Arduino

## 2.2. Raspberry Pi

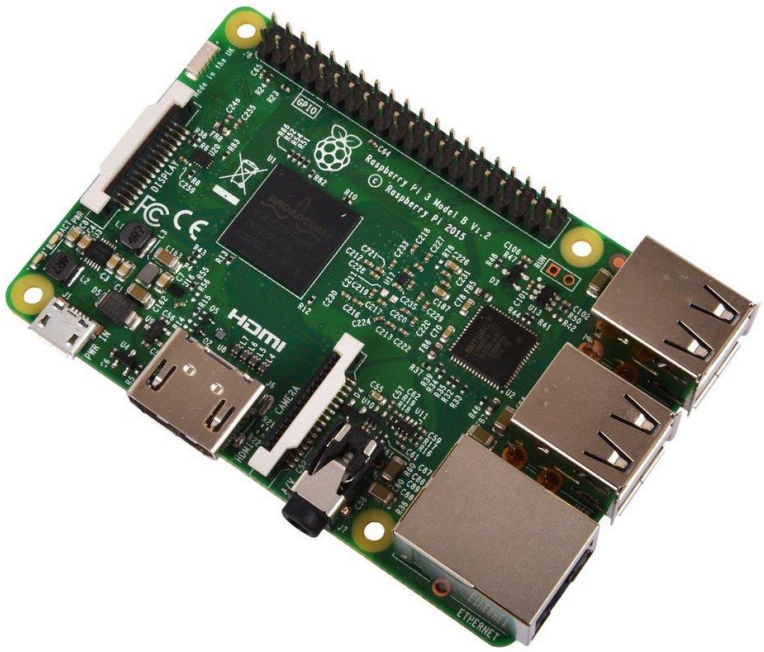
### 2.2.1 Περιγραφή

Το raspberry pi ανήκει στην κατηγορία των “single board computers (SBC)”, δηλαδή μικροί υπολογιστές μονής πλακέτας. Είναι ένας πλήρης υπολογιστής μικρού μεγέθους, ο οποίος μπορεί να χρησιμοποιηθεί σε ηλεκτρονικές εφαρμογές ή ακόμα και για προσωπική χρήση εκτελώντας λειτουργίες ενός κανονικού υπολογιστή. Τα κύρια χαρακτηριστικά του είναι:

- ο τετραπύρηνος επεξεργαστής Broadcom BCM2837 ARMv8 όπου τρέχει στα 1200MHz,
- 40 pins γενικής χρήσης (General Purpose Input/Output - GPIO pins),
- 1GB RAM,
- 4 θύρες USB,
- μία θύρα HDMI,

- μια θύρα Ethernet,
- επαφή για σύνδεση με οθόνη και κάμερα χωριστά,
- μια ενιαία θύρα για video και ήχο,
- υποδοχή για Micro SD card,
- τροφοδοσία μέσω Micro USB 5V.

Υπάρχουν διάφορα μοντέλα στο εμπόριο, αλλά σε αυτή την πτυχιακή χρησιμοποιήθηκε το Raspberry Pi 3 Model B. Κατά κύριο λόγο, το raspberry pi αναπτύχθηκε για εκπαιδευτικούς σκοπούς για να εισαχθεί στα σχολεία η βασική αρχή του προγραμματισμού.




**Εικόνα 3: Raspberry Pi 3 Model B**

### 2.2.2 Περιβάλλον Ανάπτυξης

Στο raspberry pi πρέπει να εγκατασταθεί το λειτουργικό σύστημα Raspbian, το οποίο βασίζεται σε Debian Linux. Αυτό γίνεται εύκολα με την εγκατάσταση των αρχείων του φακέλου NOOBS, το οποίο υπάρχει διαθέσιμο στο διαδίκτυο, στην κάρτα Micro SD. Το Raspbian διαθέτει εργαλεία για :

- προβολή εικόνων,
- διαχείρηση αρχείων και φακέλων,
- σημειωματάριο,
- την αντίστοιχη γραμμή εντολών (command prompt - cmd)
- και άλλες βασικές λειτουργίες ενός κανονικού υπολογιστή.

Μπορεί να γίνει περιήγηση στο Internet απο δύο διαφορετικά προγράμματα περιήγησης(browsers). Τέλος, μας δίνει την δυνατότητα προγραμματισμού με μια γλώσσα υψηλού επιπέδου, την Python, Integrated development and learning environmen - IDLE σε δυο εκδόσεις, αλλά και με τη Scratch , η οποία χρησιμοποιείται περισσότερο για

εκπαιδευτικούς σκοπούς, καθώς και τη Squeak, η οποία είναι η πλατφόρμα που τρέχει η Scratch.

## 2.3. Αισθητήρες

Αισθητήρας είναι μια συσκευή η οποία καταμετρά ένα φυσικό μέγεθος και το μετατρέπει σε μετρήσιμη έξοδο (Φραγκιαδάκης, 2006). Πιο συγκεκριμένα, οι αισθητήρες που χρησιμοποιήθηκαν στην παρούσα κατασκευή είναι ο TMP36 για την αναλογική μέτρηση της θερμοκρασίας, ο HIH6120 για την μέτρηση της υγρασίας και της θερμοκρασίας ψηφιακά και ο BMP180 για την μέτρηση της ατμοσφαιρικής πίεσης, του υψομέτρου και της ψηφιακής θερμοκρασίας. Όλοι οι αισθητήρες έχουν κάποια τεχνικά χαρακτηριστικά, γνωστά και ως datasheets, που δημοσιεύονται από την κατασκευαστική εταιρεία τους. Κάποια από αυτά θα αναλυθούν συνοπτικά παρακάτω για την κατανοήση του τρόπου λειτουργίας του κάθε αισθητήρα .

### 2.3.1 Τεχνικά Χαρακτηριστικά

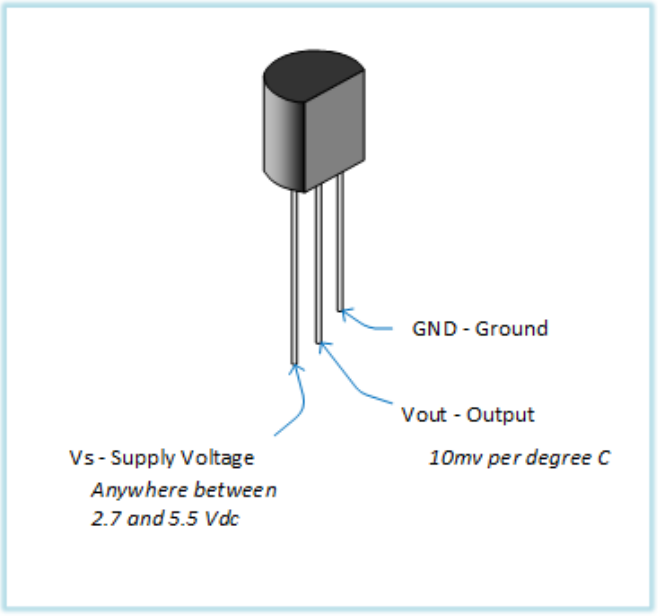
#### *2.3.1.1 Αισθητήρας TMP36*

Αρχικά, ο TMP36 χρησιμοποιείται κατά βάση για περιβαλλοντικά συστήματα ελέγχου και βαθμονομεί κατευθείαν σε βαθμούς Κελσίου, γεγονός το οποίο διευκολύνει πολύ τον προγραμματισμό του. Έχει τρία ποδαράκια (pins), όπου το πρώτο αναφέρεται στη γείωση. Το μεσαίο δίνει την μετρήσιμη τιμή, δηλαδή βγάζει σαν έξοδο την τιμή που μετράει σε 10mV/°C (millivolt ανά βαθμούς Κελσίου). Η έξοδος που βγάζει ο αισθητήρας από το μεσαίο pin ονομάζεται και συντελεστής κλίμακας εξόδου. Το τελευταίο ποδαράκι δίνει την τάση στον αισθητήρα, σε τιμές που κυμαίνονται από περίπου 2,7V εώς 5,5V.

- Προσδιορίζεται για θερμοκρασίες από -40°C εώς +125°C και έχει ακρίβεια θερμοκρασίας ±2°C.
- Παρέχει 750mV έξοδο(output) στους 25°C και λειτουργεί εώς τους +125°C με 2,7V τάση τροφοδοσίας.

(TMP36 datasheet)

Για τον συγκεκριμένο αισθητήρα είναι σημαντικό να αναφερθεί ο κώδικας που χρειάστηκε για να προγραμματιστεί κατάλληλα ώστε να εμφανίζονται οι σωστές θερμοκρασίες, το οποίο επιτυγχάνεται συντάσσοντας στον κώδικα μία συγκεκριμένη συνάρτηση. Αυτό θα αναλυθεί σε επόμενη ενότητα με την αντίστοιχη επεξήγηση των εντολών.

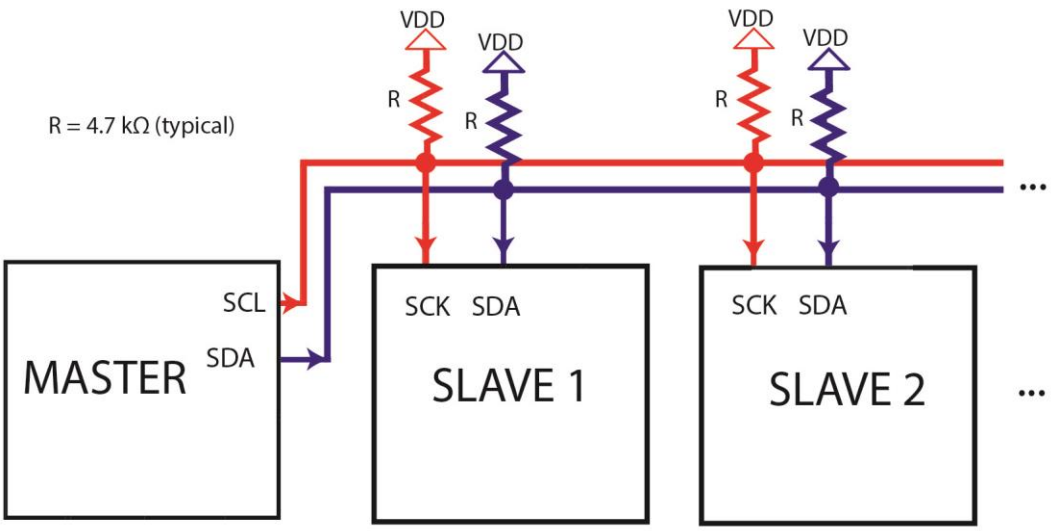



**Εικόνα 4: Επεξήγηση των pins του αισθητήρα TMP36**

#### *2.3.1.2 Αισθητήρας HIH6120*

Ο αισθητήρας HIH6120 προσδιορίζει την ψηφιακή θερμοκρασία και υγρασία μέσω του I2C πρωτόκολλου (Inter – integrated circuit protocol) . Είναι ,δηλαδή, ειδικά σχεδιασμένος ώστε να συνδέεται σε έναν μικροελεγκτή και να επικοινωνεί μέσω της διεπαφής I2C(Two Wire) που διαθέτει. Λίγο πιο συγκεκριμένα, η I2C διεπαφή επιτρέπει την επικοινωνία δύο ή περισσοτέρων συσκευών ή αισθητήρων με τη χρήση δύο μόνο καλωδίων. Το ένα καλώδιο αναφέρεται στον παλμό (clock - SCK) και το άλλο στην μεταφορά των δεδομένων(data - SDA). Οι αισθητήρες που λειτουργούν με βάση αυτή την διεπαφή έχουν σαν χαρακτηριστικό ότι στα ποδαράκια τους πέρα από τη γείωση και την τάση, έχουν και το clock - SCK ,το οποίο είναι η γραμμή που δίνει σταθερό παλμό στον αισθητήρα για το χρόνο στον οποίο πρόκειται να πάρει τις μετρήσεις. Επίσης, έχουν και το data – SDA το οποίο είναι αυτό που μεταφέρει τα δεδομένα από τον αισθητήρα στον μικροελεγκτή.

Στην Εικόνα 5, απεικονίζεται η βασική αρχή λειτουργίας της I2C διεπαφής, για να γίνει καλύτερα αντιληπτό πως ακριβώς αυτή λειτουργεί. Master και Slave είναι οι συσκευές που μπορούν να συνδεθούν. Η καθεμία από τις Slave συσκευές πρέπει να διαθέτει προκαθορισμένη και μοναδική διεύθυνση. Η διεύθυνση αυτή εμπεριέχεται από πρωτόκολλο επικοινωνίας, έτσι ώστε ο Master να επικοινωνεί με συγκεκριμένη Slave συσκευή χωρίς να εμπλέκονται στην επικοινωνία οι υπόλοιπες.

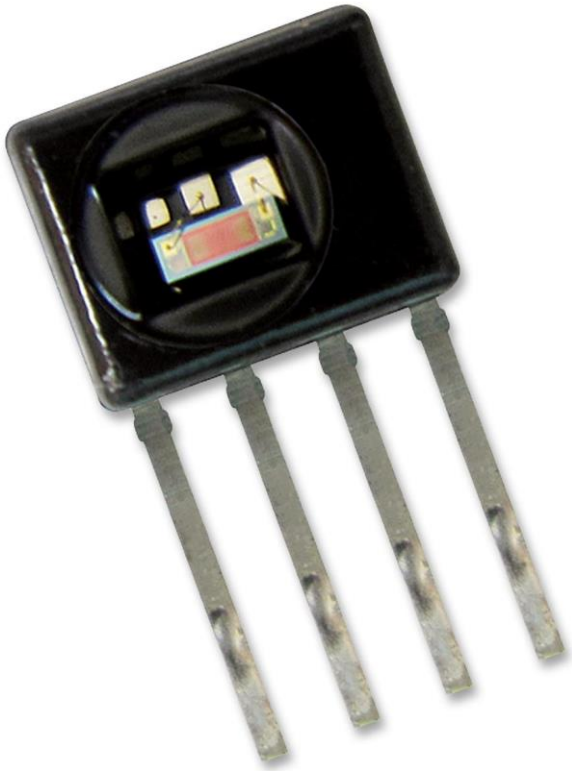



Εικόνα 5: I2C Interface

Ο αισθητήρας HIH6120 γενικά προτιμάται διότι:

- έχει υψηλή ακρίβεια στις μετρήσεις του σε πραγματικές καιρικές συνθήκες.
- το εύρος της θερμοκρασίας του είναι μεταξύ των -25°C και +85°C.
- το εύρος της υγρασίας που μπορεί να μετρήσει ξεκινά από 10% εώς 90%.
- η τάση που χρησιμοποιεί μπορεί να ξεκινήσει από τα 2,3 V και να φτάσει εώς τα 5,5V, με τυπική τροφοδοσία στα 3,3V.
- κάτι ιδιαίτερο για αυτόν τον αισθητήρα είναι το γεγονός ότι όταν δεν παίρνει μετρήσεις, μπαίνει σε sleep mode το οποίο μεγαλώνει τη διάρκεια ζωής του.
- έχει 4 ποδαράκια (Εικόνα 6), τάση, γείωση, SCK, SDA , με την επιφάνεια του γυαλιού να είναι η μπροστινή του όψη.

(Honeywell, 2012)



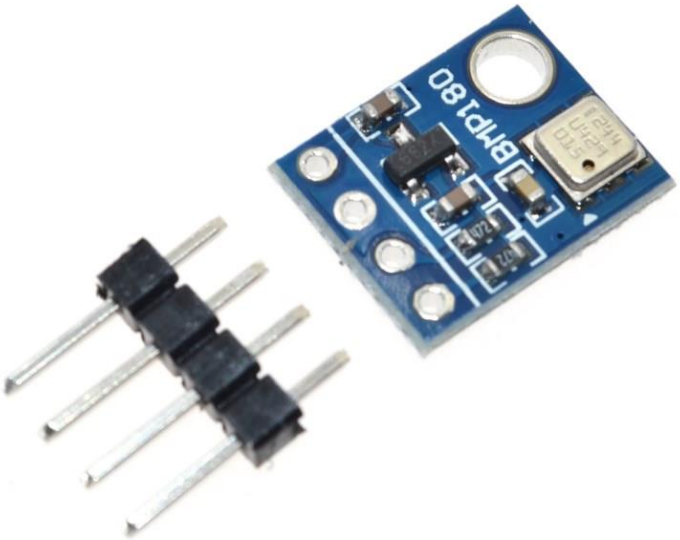
Εικόνα 6: Απεικόνιση αισθητήρα HIH6120

#### *2.3.1.3 Αισθητήρας BMP180*

Ο BMP180 αισθητήρας έχει υψηλή ακρίβεια και γραμμικότητα καθώς και σταθερότητα στα αποτελέσματά του. Συνίσταται για την χρήση του σε εφαρμογές όπως τα κινητά τηλέφωνα, PDAs – Personal digital assistant και συσκευές GPS – Global Positioning System . Λειτουργεί με βάση το I2C πρωτόκολλο που προαναφέρθηκε στην περιγραφή του HIH6120.

- Οι τιμές που δέχεται για τη θερμοκρασία ξεκινούν από -40°C εώς +85°C.
- Η τάση του από -0,3V εώς 4,25V και το εύρος πίεσης που δέχεται ξεκινά από +9000m εώς -500 m αναλογικά με το επίπεδο της θάλασσας, όπου τα μετατρέπει σχετικά σε αριθμούς με μονάδες μέτρησης το hPa(hectoPascal). Για να μετρηθεί όμως απευθείας °C και hPa πρέπει να γίνουν οι κατάλληλες μετατροπές στον κώδικα.
- Έχει 4 ποδαράκια τα οποία με τη σειρά, όπως φαίνονται στη φωτογραφία που παρατίθεται είναι η τάση, η γείωση, το SCL και το SDA.

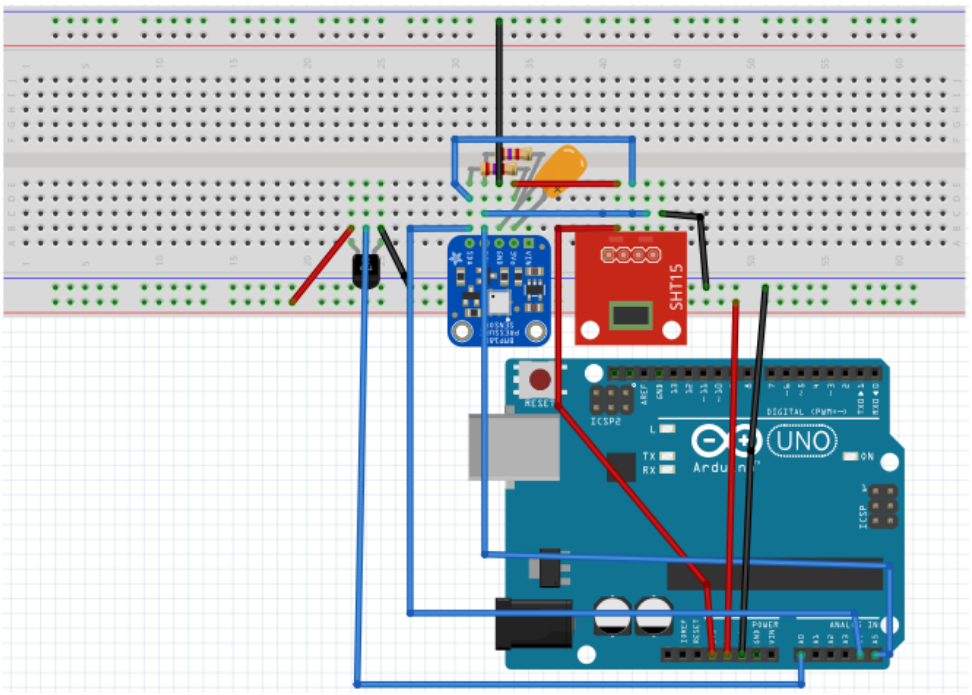
(Bosch, 2013)




Εικόνα 7: Απεικόνιση αισθητήρα BMP180

### 2.3.2 Συνδεσμολογία Αισθητήρων

Όπως αναφέρθηκε και στο προήγουμενο κεφάλαιο οι αισθητήρες συνδέονται με έναν συγκεκριμένο τρόπο, με τον μικροελεγκτή Arduino. Ο TMP36 που μετρά την αναλογική θερμοκρασία έχει τρία ποδαράκια, ένα για τη γείωση, το μεσαίο που βγάζει την μετρήσιμη έξοδο στο A0 του Arduino και το άλλο ποδαράκι για την τάση. Ο BMP180 που μετρά την ψηφιακή θερμοκρασία, ατμοσφαιρική πίεση και υψόμετρο έχει τέσσερα ποδαράκια τα οποία πάνε με τη σειρά, τάση, γείωση, SCL και τελευταίο το SDA. Την ίδια ακριβώς σειρά ακολουθούν και τα ποδαράκια του HIH6120. Τα SCL και SDA των δύο αισθητήρων συνδέονται και μεταξύ τους μαζί με αντιστάσεις των 2,7K και με έναν πυκνωτή των 100nF. Αυτό δημιουργήθηκε έτσι, για την καλύτερη δυνατή ασφάλεια των αισθητήρων. Η λογική του πυκνωτή είναι ότι σε οποιαδήποτε περίπτωση πτώσης της τάσης, να μην δημιουργηθεί πρόβλημα απότομης έλλειψης ρεύματος. Είναι προτιμότερο να εξομαλυνθεί, δηλαδή, η πτώση της καθώς αυτό που χαρακτηρίζει έναν πυκνωτή είναι ότι παρουσιάζεται ως μια αποθήκη ρεύματος. Για να γίνει πιο σαφές και εύκολο στην κατανόηση χρησιμοποίηθηκε ένα πρόγραμμα σχεδίασης, το Fritzing. Είναι ένα ελεύθερο πρόγραμμα σχεδίασης κυκλωμάτων, έχει μεγάλη ποικιλία εργαλείων και μπορεί κανείς να κατεβάσει από το διαδίκτυο τις κατάλληλες βιβλιοθήκες έτσι ώστε να εισάγει στο πρόγραμμα τα αντικείμενα που επιθυμεί να εντάξει στο κύκλωμά του. Στο διάγραμμα που θα εμφανιστεί φαίνεται καθαρά η συνδεσμολογία μόνο των αισθητήρων TMP36, BMP180 και HIH6120 με το Arduino. Με το μαύρο καλώδιο προβλέπεται η σύνδεση με τη γείωση και με το κόκκινο η τάση.




Εικόνα 8: Συνδεσμολογία Arduino και αισθητήρων με το πρόγραμμα Fritzing

### 2.3.3 Προγραμματισμός Αισθητήρων

Κάθε αισθητήριο χρειάζεται διαφορετικό τρόπο διαχείρησης στον προγραμματισμό του. Σε αυτή την ενότητα παρατίθεται ο κώδικας που χρησιμοποιήθηκε στην παρούσα εργασία για κάθε αισθητήριο χωριστά, και στη συνέχεια γίνεται μια μικρή επεξήγηση του τρόπου με τον οποίον αυτοί προγραμματίστηκαν.

- Για τον TMP36:

```
int temp36;


int AnalogTemp() {
    int sensorValue = analogRead(A0);
    double x = (double)sensorValue;
    x *= 5000;
    x /= 1023;
    x /= 30;
    return (int)x;
```

```
}

void Measure_All(){
temp36 = AnalogTemp();
}
```

Στην αρχή του κώδικα, πριν την συνάρτηση set up() δηλώνονται όλες οι μεταβλητές που χρησιμοποιήθηκαν στο πρόγραμμα με ένα σχετικό όνομα το οποίο θα τις χαρακτηρίζει. Δηλώνεται το όνομα της μεταβλητής, "temp36" για τον συγκεκριμένο αισθητήρα, και εν συνεχεία δηλώνεται ότι είναι της μορφής int ώστε να εμφανίζει σαν αποτέλεσμα ακέραια τιμή. Εφόσον δηλώθηκαν οι μεταβλητές, συντάσσεται μια συνάρτηση για κάθε μεταβλητή που λαμβάνει μετρήσιμες τιμές. Η ρουτίνα που δημιουργήθηκε λαμβάνει διαφορετικό όνομα από αυτό που δώθηκε στην μεταβλητή, για αποφυγή λαθών στο πρόγραμμα. Έπειτα, διαβάζεται η αναλογική έξοδος του αισθητήρα TMP36 που είναι συνδεδεμένη στον Arduino. Πιο συγκεκριμένα, δηλώνεται η μορφή της τιμής εξόδου του αισθητήρα και ταυτόχρονα εξισώνεται με την αναλογική έξοδο που έχει συνδεθεί στο αναλογικό pin του Arduino για να πραγματοποιείται η μέτρηση. Τίθεται ως όνομα της μεταβλητής το «x», που ισούται με την τιμή που λαμβάνεται απο την έξοδο του Arduino. Επειδή γίνεται μια μετατροπή χωρητικότητας της τιμής προσδιορίζεται ως double η μορφή της μεταβλητής. Αυτό συμβαίνει διότι η μορφή int καταλαμβάνει 2 bytes, δηλαδή 16 bits, ενώ η double 4 bytes, δηλαδή 32 bits.

Τα αναλογικά pin του Arduino δέχονται ως είσοδο αναλογικές τιμές. Παρόλα αυτά, στον μικροελεγκτή του Arduino επεξεργάζονται μόνο ψηφιακά δεδομένα καθώς είναι ψηφιακός. Επομένως, για να χρησιμοποιηθούν αναλογικά δεδομένα είναι αναγκαίο πρώτα να μετατραπούν σε ψηφιακά. Γιαυτό τον λόγο τα αναλογικά pin είναι συνδεδεμένα με έναν ενσωματωμένο μετατροπέα αναλογικού σε ψηφιακό (A/D converter) εύρους 10bit.

Ο μετατροπέας A/D ορίζει $2^{10}$ = 1024 διακριτές τιμές μέσα στο εύρος τιμών της τάσης εισόδου(0-5Volt) και αντιστοιχεί σε καθεμία από αυτές έναν αριθμό από το 0 μέχρι το 1023. Έτσι επιτυγχάνεται ο μετατροπέας να επιστρέφει 0 για λαμβανόμενη τάση 0Volt και 1023 για λαμβανόμενη τάση 5Volt. Αυτό σημαίνει ότι για το διάστημα 0-5Volt υπάρχει ανάλυση 0,00488V(διαιρώντας το 5 με το 1023). Για οποιαδήποτε ενδιάμεση τιμή τάσης επιστρέφεται ο αριθμός που αντιστοιχεί στην κοντινότερη σε αυτήν διακριτή τιμή (Παπάζογλου, 2014).

Ο αριθμός 1023 για το Arduino είναι τα 10bit που έχει διαθέσιμα η κεντρική μονάδα επεξεργασίας (Central Processing Unit - CPU) και είναι θέμα ακρίβειας. Σε άλλες πλακέτες μπορεί να έχει περισσότερα bits διαθέσιμα, άρα και άλλον αριθμό ακρίβειας.

Το αποτέλεσμα της πράξης που έχει γίνει διαιρείται με το 30 που ορίζεται από τον κατασκευαστή για την μετατροπή από mV σε βαθμούς Κελσίου.

Στο τέλος της ρουτίνας ζητείται να επιστραφεί η τιμή «x» ορισμένη ως int, διότι έχει γίνει η μετατροπή και μπορεί ο αριθμός που δίνεται να χωρέσει στα 2 bytes. Δημιουργείται μια άλλη συνάρτηση που ονομάζεται Measure_All() και εκεί δηλώνεται ότι η μεταβλητή που ονομάσθηκε στην αρχή temp36 κάθε φορά που διαβάζεται απο

τον κώδικα, τρέχει την ρουτίνα AnalogTemp(). Αναγνωρίζεται ότι είναι ρουτίνα λόγω των παρενθέσεων ().

- Για τον BMP180:

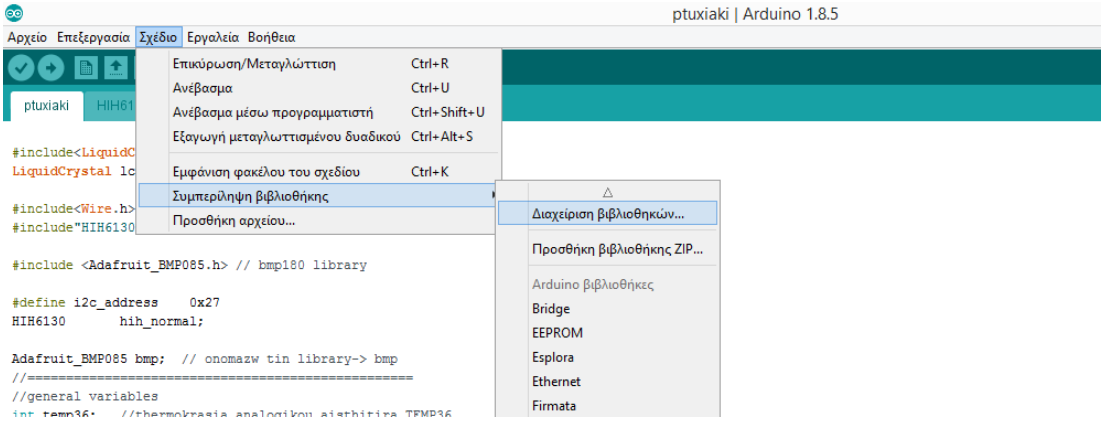
```
#include <Adafruit_BMP085.h>

Adafruit_BMP085 bmp;
float bmptemp;
float alti;
int presf;

float BMP180_Temp(){
        return bmp.readTemperature();
}
float BMP180_Altitude(){
        return bmp.readAltitude(101500);
}
int BMP180_Pressure(){
        long pres = bmp.readPressure();
        return (int)(pres/100);
}

void Measure_All()
bmptemp = BMP180_Temp();
        alti = BMP180_Altitude();
        presf = BMP180_Pressure();
```

Για αυτόν τον αισθητήρα θα πρέπει να ενεργοποιηθεί η κατάλληλη βιβλιοθήκη, όπου υπάρχουν κάποιες έτοιμες συναρτήσεις, έτσι ώστε κάθε φορά που παίρνει μετρήσεις να τρέχει ταυτόχρονα και στο περιεχόμενο της βιβλιοθήκης εάν αυτό χρειαστεί. Ακολουθούνται τα βήματα στο IDLE του Arduino Σχέδιο → Συμπερίληψη Βιβλιοθήκης → Διαχείρηση Βιβλιοθήκης όπως φαίνονται και στην Εικόνα 9 :




**Εικόνα 9: Βήματα για συμπερίληψη βιβλιοθήκης**

Ανοίγεται ένα παράθυρο οπού βρίσκονται όλες οι βιβλιοθήκες που υπάρχουν στο πρόγραμμα, κάποιες εκ των οποίων είναι ήδη ενεργοποιημένες και κάποιες δίνεται η δυνατότητα να ενεργοποιηθούν. Αφού βρεθεί η βιβλιοθήκη “Adafruit_BMP085.h” που απευθύνεται και στον αισθητήρα BMP180 πρέπει να ενεργοποιηθεί. Με τη λέξη #include δηλώνεται ότι πρόκειται για βιβλιοθήκη που εντάσσεται στο πρόγραμμα που ακολουθεί. Στη συνέχεια, προαιρετικά, επιλέχθηκε να ονομαστεί η βιβλιοθήκη για το πρόγραμμα που θα τρέχει “bmp”, έτσι ώστε κάθε φορά που τρέχει μια ρουτίνα που υπάρχει ήδη ενσωματωμένη σε αυτήν, να καλείται μέσα απο την καινουργιά ονομασία της. Σε ακολουθία αυτών, δηλώνονται οι μεταβλητές και η μορφή τους. Με int δηλώνονται οι ακέραιες και με float οι δεκαδικές. Έπειτα, δημιουργήθηκαν τρεις ρουτίνες μία για κάθε στοιχείο που μετρά ο συγκεκριμένος αισθητήρας. Η πρώτη αναφέρεται στην θερμοκρασία, η δεύτερη στο υψόμετρο και η τρίτη στην ατμοσφαιρική πίεση. Σε κάθε ρουτίνα δίνεται ένα σχετικό όνομα και ζητάται να επιστρέφει την τιμή με βάση την βιβλιοθήκη που ενεργοποιήθηκε για αυτόν τον αισθητήρα. Όσον αφορά την ατμοσφαιρική πίεση, η μέτρηση που παίρνει είναι σε Pascal ενώ είναι επιθυμητό να εμφανίζεται η τιμή σε millibar (hectopascal). Για το σκοπό αυτό, γίνεται “casting” στην μεταβλητή από float σε int , λόγω της διαίρεσης που ζητήθηκε να γίνει για την μετατροπή της μονάδας. Στο τέλος, ορίζονται οι ρουτίνες με άλλο όνομα μεταβλητής για λόγους ευκολίας.

- Για τον HIH6120:

```
#include<Wire.h>    //i2C επικοινωνια

#include"HIH6130.h"


#define i2c_address    0x27

HIH6130       hih_normal;


float humidity;  // ugrasia HIH6120

float temperature; //thermokrasia HIH6120
```

```
void HIH_Values(){
  hih_normal.begin(i2c_address);
  hih_normal.fetch_data(&humidity, &temperature);
}

void Measure_All()
HIH_Values();
```

Για αυτόν τον αισθητήρα χρειάστηκε η ενεργοποιήση της βιβλιοθήκη που υπάρχει για την I2C διεπαφή δηλώνοντάς τη με #include . Αμέσως μετά , πρέπει να ληφθεί από το διαδίκτυο η αντίστοιχη βιβλιοθήκη του HIH6130, όπου για τους αισθητήρες HIH6130/31 και HIH6120/21 υπάρχει η ίδια βιβλιοθήκη και εισάγεται στο πρόγραμμα με #include. Ιδιαίτερη προσοχή απαιτεί το γεγονός ότι για να μπορεί το πρόγραμμα να την διαβάζει, θα πρέπει να αποθηκευτεί το αρχείο τύπου C/C++ που κατεβαίνει για την βιβλιοθήκη στον ίδιο φάκελο που τοποθετείται το πρόγραμμα. Στη συνέχεια, με την εντολή #define ορίζεται η διεύθυνση που έχει ο σχετικός αισθητήρας. Αυτό συμβαίνει για να μην δημιουργηθεί πρόβλημα με τον BMP180 λόγω του ότι και εκεί χρησιμοποιείται το πρωτόκολλο της I2C επικοινωνίας. Οι διευθύνσεις των δύο αισθητηρίων είναι διαφορετικές, σε άλλη περίπτωση θα έπρεπε να αλλάχθεί η διεύθυνση του ενός. Προαιρετικά μετονομάζεται η βιβλιοθήκη σε hih_normal για την κλήση των συναρτήσεων που υπάρχουν μέσα και είναι απαραίτητη για να ληφθούν οι μετρήσεις. Δηλώνονται ως float τιμές η υγρασία και η θερμοκρασία που μετριούνται και δίνεται ένα αντίστοιχο σχετικό όνομα σε κάθε μεταβλητή. Δημιουργήθηκε μία συνάρτηση για αυτόν τον αισθητήρα που ονομάστηκε HIH_Values(). Μέσα στο σώμα της δόθηκε η εντολή να ξεκίνησει την I2C επικοινωνία μέσω της διεύθυνσης που έχει δωθεί, διότι όταν υπάρχουν περισσότερες από μία συσκευές που ακολουθούν αυτή την επικοινωνία κάπου συγχέεται η διαδρομή με αυτήν που πρέπει κανονικά να ακολουθήσει. Στην ίδια συνάρτηση δίνεται η εντολή να βγάλει τις τιμές των μεταβλητών που έχουν δηλωθεί με το πρόθεμα &, δηλαδή δηλώνονται τα στοιχεία ως δείκτες (pointers). Pointers είναι οι μεταβλητές που περιέχουν την διεύθυνση των αντίστοιχων μεταβλητών. Μπορούν να χρησιμοποιηθούν σαν παράμετροι στις συναρτήσεις, έτσι ώστε κάθε αλλαγή που γίνεται να εκχωρείται κατευθείαν στην διεύθυνση που είναι αποθηκευμένη η μεταβλητή. Τέλος, στη ρουτίνα που έχει δημιουργηθεί με όλες τις αντιστοιχίες των μεταβλητών με τις δικιές τους συναρτήσεις, δηλώνεται μόνο η συνάρτηση HIH_Value() διότι κάθε φορά μπορεί να την τρέξει για να πάρει τις μετρήσεις που ζητήθηκαν.

## 2.4. Οθόνη LCD

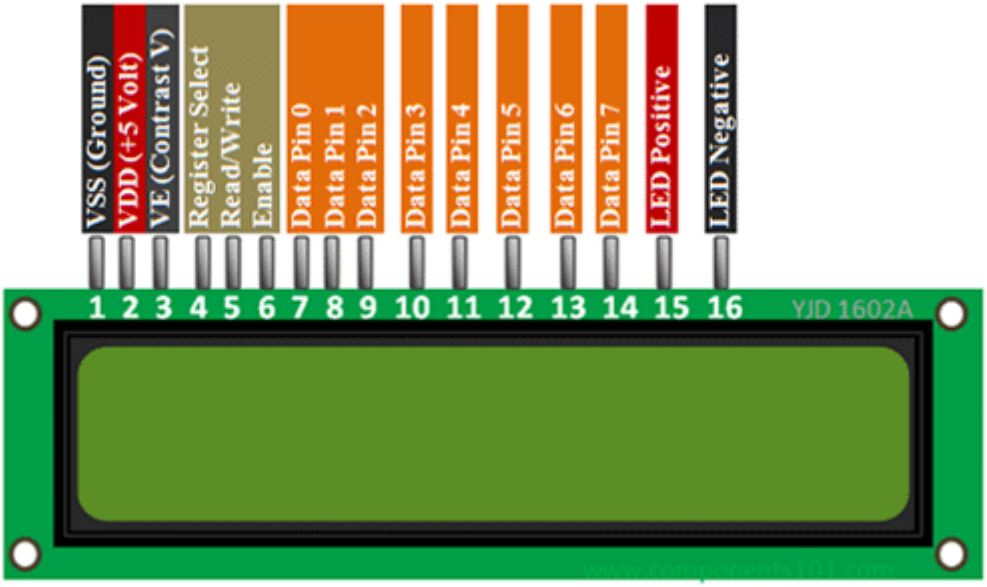
Στην παρούσα εργασία επιλέχθηκε να χρησιμοποιηθεί μια οθόνη υγρών κρυστάλλων (Liduid crystal display - LCD). Η οθόνη αυτή διαθέτει 16 pins(ακροδέκτες) τα οποία πρέπει

να συνδεθούν με έναν συγκεκριμένο τρόπο. Επίσης, στην οθόνη εμφανίζονται 2 σειρές, όπου κάθε σειρά μπορεί να δεχτεί 16 χαρακτήρες. Γι'αυτό άλλωστε ονομάζεται LCD 16x2. Είναι σημαντικό να αναφερθεί ότι υπάρχει συγκεκριμένος κώδικας για τις ενδείξεις που εμφανίζονται στην οθόνη και συγκεκριμένη βιβλιοθήκη που εισάγεται. Επιπροσθέτως, να αναφερθεί ότι με την κατάλληλη συνδεσμολογία της οθόνης, υπάρχει η δυνατότητα για ρύθμιση της φωτεινότητας.

Εικόνα 10: Liquid crystal display - LCD 16x2

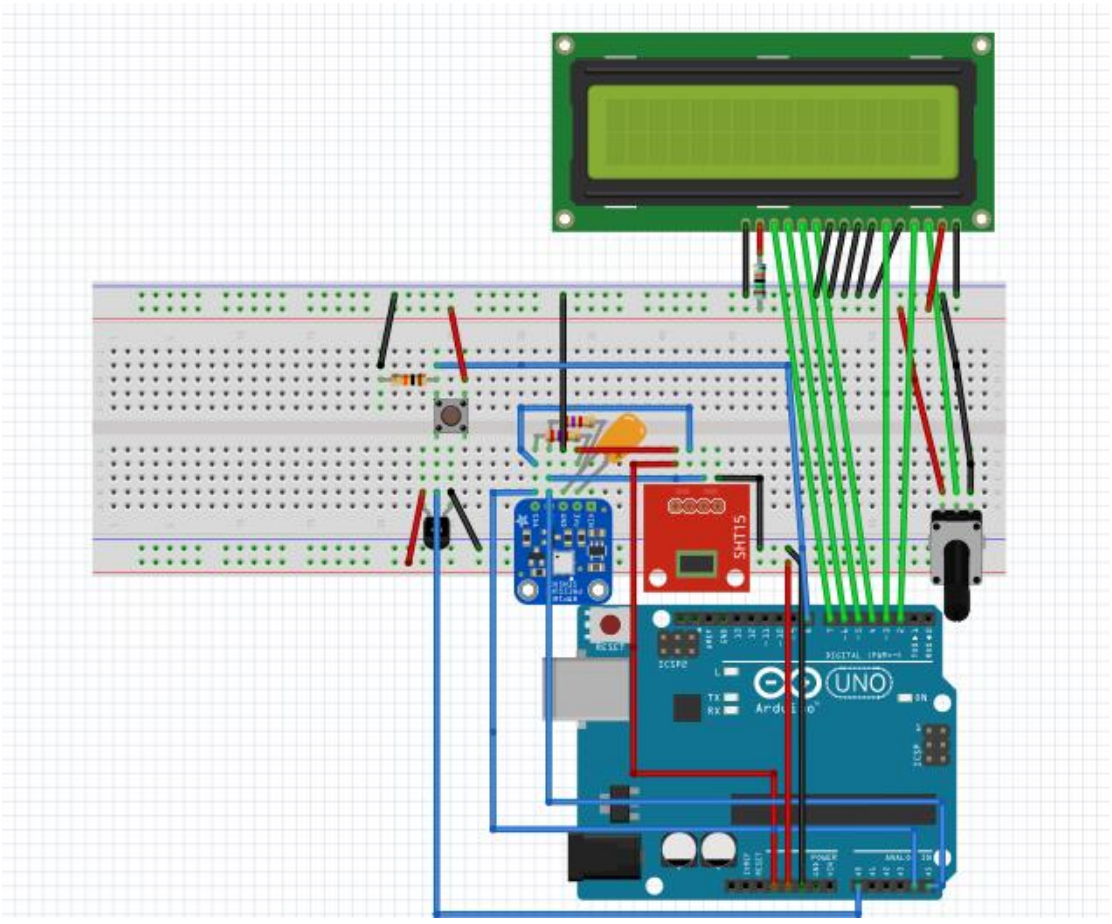
### 2.4.1 Συνδεσιμότητα

Η οθόνη LCD έχει 16 ποδαράκια, όπου το καθένα προορίζεται για συγκεκριμενή διεργασία, όπως φαίνεται στη Εικόνα 11.




Εικόνα 11: Οι έξοδοι της LCD

Η συνδεσμολογία που έχει επιτευχθεί για την συγκεκριμένη κατασκευή φαίνεται στην Εικόνα 12. Για να ρυθμιστεί η ένταση της φωτεινότητας της οθόνης προσθέτεται ένα ποτενσιόμετρο των 10K, έτσι ώστε περιστρέφοντάς το να φτάσει στην επιθυμητή φωτεινότητα. Προστίθεται μία αντίσταση των 150K για την τροφοδοσία του οπίσθιου φωτισμού της οθόνης. Επίσης, προστέθηκε ένα κουμπί (button) με μία αντίσταση των 10K. Κάθε φορά που πιέζεται το κουμπί η οθόνη αλλάζει ενδείξεις και εμφανίζονται με κάποια σειρά οι προκαθορισμένες ενδείξεις οθόνης που έχουν προγραμματιστεί μέσω του Arduino IDLE. Ο κώδικας που συντάχθηκε για την οθόνη παρατίθεται και αναλύεται σε επόμενη ενότητα.




**Εικόνα 12: Συνδεσμολογία κατασκευής με LCD μέσω του προγράμματος Fritzing**

### 2.4.2 Λειτουργία και Ενδείξεις

Για να λειτουργήσει η οθόνη που έχει συνδεθεί, χρειάζεται να προγραμματιστεί κατάλληλα μέσω του IDLE Arduino, όπως αναφέρθηκε και παραπάνω. Ο προγραμματισμός της είναι απαραίτητος, διότι η οθόνη έχει συνδεθεί ώστε να στέλνονται τα δεδομένα που λαμβάνονται από τους αισθητήρες πίσω σε αυτήν με την σωστή μορφή. Πιο συγκεκριμένα, η λειτουργία που έχει καθοριστεί για την οθόνη είναι να εμφανίζει με κάθε πάτημα κουμπιού που έχει συνδεθεί μαζί της, την εναλλαγή των ενδείξεων. Με την λέξη εναλλαγή

εννοείται ότι λόγω των 16 χαρακτήρων που χωράει μόνο η κάθε σειρά, ανανεώνεται η οθόνη, σβήνοντας δηλαδή την προηγούμενη ένδειξη και εμφανίζοντας την καινούργια. Για να ληφθούν όλα τα δεδομένα που κρίθηκαν απαραίτητα, χρειάστηκαν τέσσερις εναλλαγές οθόνης. Αυτό βέβαια μπορεί να τροποποιηθεί εφόσον είναι επιθυμητό, όπως επίσης και το περιεχόμενο του μηνύματος, η σειρά ή ακόμη και ο τρόπος γραφής του. Η διασύνδεση της οθόνης στην κατασκευή εξυπηρετεί στην άμεση οπτική επαφή των αποτελεσμάτων αλλά και στην ολοκλήρωση της λογικής του μετεωρολογικού σταθμού. Πλεόν με την προσθήκη αυτής της οθόνης, οπουδήποτε και να πραγματοποιηθεί η σύνδεση της κατασκευής αυτής, οι μετρήσεις που θα ληφθούν μπορούν να εμφανιστούν στην οθόνη LCD χωρίς να χρειάζεται κάποια άλλη σύνδεση, π.χ. με υπολογιστη.

Η πρώτη ένδειξη εμφανίζεται με την σύνδεση σε τάση του Arduino. Λόγω έναρξης της τροφοδοσίας επιλέχθηκαν οι πρώτες πληροφορίες να είναι σχετικές με το θέμα. Αναγράφεται, δηλαδή, το ονοματεπώνυμο του συντάκτη της εργασίας και ο προσδιορισμός της εργασίας.


Εικόνα 13: Πρώτη απεικόνιση οθόνης LCD

Με το επόμενο πάτημα του κουμπίου γίνεται καθαρισμός της οθόνης των προηγούμενων στοιχείων, αλλάζει η ένδειξη της οθόνης και καταγράφονται τα πρώτα στοιχεία σχετικά με τις μετρήσεις. Πιο συγκεκριμένα, αναγράφεται η θερμοκρασία του αισθητήρα TMP36, η ταχύτητα του ανέμου, η θερμοκρασία και η υγρασία του αισθητήρα HIH6120.


Εικόνα 14: Δεύτερη απεικόνιση οθόνης LCD

Στο επόμενο πάτημα σβήνονται από την οθόνη  τα προηγούμενα στοιχεία, αλλάζει η ένδειξη και εμφανίζονται τα στοιχεία του αισθητήρα BMP180, δηλαδή η θερμοκρασία, το υψόμετρο και η ατμοσφαιρική πίεση.


Εικόνα 15: Τρίτη απεικόνιση οθόνης LCD

Στην τελευταία ένδειξη που έχει προκαθοριστεί σαν τέταρτη εναλλαγή της οθόνης, αναγράφεται στο πάνω μέρος της με ένα σχετικό μήνυμα ότι τα στοιχεία που μετρήθηκαν στέλνονται στην Python. Έτσι, δείχνεται ότι λήφθηκαν όλες οι δυνατές μετρήσεις, τα στοιχεία αυτά καταγράφτηκαν και αποστέλονται στο Raspberry Pi. Εάν μετά από αυτή την κίνηση πατηθεί ξανά το κουμπί, ο Arduino έχει προγραμματιστεί κατάλληλα για να ξεκινήσει την εμφάνιση των μηνυμάτων από την αρχή. Επιτυγχάνεται, δηλαδή, μια συνεχής επανάληψη βρόγχου, διότι δεν έχει δωθεί κάποια εντολή τερματισμού.


Εικόνα 16: Τέταρτη απεικόνιση οθόνης LCD

#### *2.4.2.1 Προγραμματισμός οθόνης*

```
#include<LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 4, 5, 6, 7);


int buttonState;

int othoni;
```

```
void Show_Results_LCD(unsigned char a){
    if(a==0){
            lcd.clear();
            lcd.print("Eleni Tsanidou");
            lcd.setCursor(0, 1);
          lcd.print("PTUXIAKI ERGASIA");

    }else if(a==1){
    lcd.clear();
   lcd.print("T36:");
   lcd.setCursor(4, 0);
   lcd.print(temp36,1);
   lcd.print("oC W:");
   lcd.print(fanValue, 1);
   lcd.setCursor(0,1);
   lcd.print("H:");
   lcd.print(String(humidity));
   lcd.print("%,T:");
  lcd.print(String(temperature));
  lcd.print("oC");

  }else if(a==2){
  lcd.clear();
  lcd.print("BMP TEMP: ");
  lcd.print(bmptemp, 1);
  lcd.setCursor(0, 1);
  lcd.print("Alt:");
  lcd.print(alti, 1);
  lcd.print(",P:");
  lcd.print(presf, 1);

 }else if(a==3){
  lcd.clear();
```

```
    lcd.print("Sending results");
    lcd.setCursor(0, 1);
    lcd.print("to Python");

  }else{
    }
}

void setup() {
  pinMode(8, INPUT);
  lcd.begin(16, 2);
  Show_Results_LCD(0);
  delay(3000);

if (!bmp.begin())
  {
  lcd.clear();
  lcd.print("BMP180 sensor not found");
  while (1) {}
  }
 othoni = 0;
}

if (Serial.available() > 0) {
  unsigned char rec_byte=0;
  rec_byte = Serial.read();
  if(rec_byte=='A'){
    Measure_All();
    Serial.println(Create_Pack());
    Show_Results_LCD(3);

  }else if(rec_byte=='B'){
    Show_Results_LCD(0);
```

```
 }else if(rec_byte=='C'){
   Show_Results_LCD(1);

 }else if(rec_byte=='D'){
   Show_Results_LCD(2);

 }else{
   int dummy=0;
 }
}
```

```
buttonState = digitalRead(8);
 if(buttonState==HIGH){
    Measure_All();
   Show_Results_LCD(othoni);
   if(othoni<4){
     othoni +=1;
   }else{
     othoni = 0;
   }
```

Για τον ορθό προγραμματισμό της οθόνης LCD πρέπει να εισαχθεί η κατάλληλη βιβλιοθήκη στο πρόγραμμα. Στην αρχή του κώδικα δηλώνεται η βιβλιοθήκη #include<LiquidCrystal.h> και έτσι μπορούν να κληθούν οι ρουτίνες που υπάρχουν μέσα στο σώμα της. Η εντολή «LiquidCrystal lcd(2,3,4,5,6,7);» είναι της πρότυπης μορφής «LiquidCrystal lcd(rs,enable,d4,d5,d6,d7);». Το rs(register select) είναι το pin του Arduino στο οποίο είναι συνδεδεμένο το pin RS της οθόνης. Το enable είναι το pin του Arduino που συνδέεται το enable της οθόνης. Τα d4- d7 είναι τα data pins της οθόνης που συνδέονται στις θύρες του Arduino. Σε αυτά τα στοιχεία αντικαθίστανται οι αριθμοί που αντιστοιχούν στην σύνδεση με το Arduino.

Στο σημείο που έχει επιλεχθεί για την δήλωση όλων των μεταβλητών, δηλώνονται δύο ακόμη μεταβλητές. Μία για την κατάσταση του κουμπιού που έχει προστεθεί για την εναλλαγή των ενδείξεων της οθόνης ως “buttonState”, και μία μεταβλητή η οποία αντιπροσωπεύει τον αριθμό της οθόνης που είναι ως “othoni”. Δηλαδή, αν είναι η πρώτη ένδειξη που έχει προκαθοριστεί να δείχνει, η μεταβλητή “othoni” ισούται με 0. Όταν πατηθεί το κουμπί για πρώτη φορά, αλλάζει η κατάσταση του κουμπιού, άρα αλλάζει και η

ένδειξη της οθόνης , επομένως η οθόνη περνάει στην επόμενη μετρήσιμη κατάσταση που ισούται με 1.

Πριν την ρουτίνα setup() δηλώνεται το περιεχόμενο των ενδείξεων της οθόνης. Κάποιες βασικές εντολές για την οθόνη είναι :

- lcd.begin(cols,rows); : Αρχικοποιείται η διεπαφή για την οθόνη LCD και καθορίζονται οι διαστάσεις της οθόνης (στήλες, γραμμές). Στη ρουτίνα setup() πρέπει αυτή να καλείται πριν από οποιαδήποτε άλλη εντολή βιβλιοθήκης LCD.
- lcd.clear(); : Καθαρίζει την οθόνη και τοποθετεί τον κέρσορα στην επάνω αριστερή γωνία, δηλαδή στην πρώτη γραμμή και πρώτη θέση χαρακτήρα.
- lcd.print(); : Εκτυπώνει το περιεχόμενο που υπάρχει μέσα στις παρενθέσεις. Εάν αυτό βρίσκεται μέσα σε εισαγωγικά " ", εκτυπώνει ακριβώς ότι αναγράφεται μέσα. Μπορεί όμως, να εκτυπώσει και το περιεχόμενο μιας μεταβλητής. Αυτό γίνεται γράφοντας μόνο το όνομα της μεταβλητής.
- lcd.setCursor(col, row); : Τοποθετεί τον κέρσορα της οθόνης LCD στη θέση που δηλώνεται μέσα στις παρενθέσεις. Δηλαδή, ορίζει τη θέση στην οποία θα εμφανιστεί το επόμενο κείμενο που έχει γραφτεί στην οθόνη LCD.

Δημιουργήθηκε στην αρχή του κώδικα πριν την setup() μια ρουτίνα στην οποία εντάσσονται όλες οι πιθανές ενδείξεις που έχει καθοριστεί να βγάζει η οθόνη. Η λέξη "void" στην αρχή της δήλωσης της συνάρτησης υποδεικνύει ότι η συνάρτηση αυτή δεν θα επιστρέφει κάποια πληροφορία. Το "unsigned char a" που έχει δηλωθεί στην ρουτίνα υποδηλώνει το χαρακτήρα που θα μετράει τα πατήματα του κουμπιού. Ξεκινάει με μια συνθήκη επανάληψης "if" , η οποία ουσιαστικά λαμβάνει υπόψιν τα πατήματα. Δηλαδή, εάν δεν έχει πατηθεί το κουμπί (a==0), τότε αρχικά καθαρίζει την οθόνη από τυχόν υπολείμματα προηγούμενης σύνδεσης. Αμέσως μετά δίνεται η εντολή να γράψει το ονοματεπώνυμο του συντάκτη της πτυχιακής με την εντολή "lcd.print" και θέτει σε συγκεκριμένη θέση τον κέρσορα από όπου επιθυμεί να ξεκινήσει την εκτύπωση που περιέχεται στην εντολή. Ο κέρσορας τοποθετείται με την κατάλληλη εντολή στην θέση της πρώτης γραμμής και πρώτου χαρακτήρα της σειράς. Αυτό επιτυγχάνεται με την μέτρηση των θέσεων. Για τις γραμμές η σωστή μέτρηση είναι ότι η πρώτη γραμμή λαμβάνεται υπόψιν με τον αριθμό 0 και η δεύτερη με τον αριθμό 1. Αντίστοιχα και για τους χαρακτήρες των γραμμών ισχύει η μέτρηση από 0 εώς 15. Στην συγκεκριμένη εκτύπωση έχει επιλεχθεί να βγαίνει στη θέση 1 παίρνοντας όμως υπόψιν ότι η μέτρηση ξεκινάει από το 0. Αυτόματα ,για την επόμενη εκτύπωση χωρίς να χρειάζεται να γίνει ξανά η τοποθέτηση του κέρσορα,εκτυπώνει την φράση που ζητείται με την εντολή στην επόμενη σειρά, από την θέση 0 αυτή τη φορά.

Εφόσον πατηθεί μια φορά το κουμπί και το "a" γίνει ίσο με 1 αλλάζει η ένδειξη της οθόνης. Αυτό επιτυγχάνεται με τα παρακάτω:

- πρώτα πρέπει να καθαριστεί η οθόνη, διότι αλλιώς δεν θα εμφανίσει κανένα μήνυμα.
- Στην συνέχεια τυπώνει τα στοιχεία που δηλώνονται με τις σωστές εντολές και όπου χρειάζεται τοποθετείται ο κέρσορας στην σωστή θέση, ώστε τα αποτελέσματα να βγαίνουν σε μια θέση που μπορούν να είναι ευανάγνωστα.
- Στην εντολή που υπάρχει πρώτα μεταβλητή κ μετά ένας αριθμός υποδηλώνεται ότι θα εκτυπωθεί το περιεχόμενο τιμής αυτής της μεταβλητής με ακρίβεια ψηφίων τόσα όσα υποδεικνύει ο αριθμός.

- Υπάρχει και η περίπτωση εντολής που δηλώνεται το “print” με την μορφή String(μεταβλητή). Σε αυτή την εκδοχή δηλώνεται ότι το περιεχόμενο είναι String, δηλαδή μια σειρά από αλφαριθμητικά στοιχεία.

Όταν πατηθεί το κουμπί για δεύτερη φορά, έχει συνταχθεί η αντίστοιχη εντολή επανάληψης (a==2). Ζητούνται να εκτυπωθούν κάποια στοιχεία, ο κέρσορας να ξεκινήσει την εκτύπωση από κάποια συγκεκριμένη θέση και όπου χρειάστηκε να δηλωθεί και η ακρίβεια των ψηφίων.

Στο επόμενο πάτημα του κουμπιού το “a” γίνεται ίσο με 4 και εκτυπώνεται στην οθόνη LCD το αντίστοιχο μήνυμα, με τον κέρσορα στην θέση που προκαθορίζεται από την αντίστοιχη εντολή.

Τέλος, κλείνει η επανάληψη αυτή με την εντολή “else” της οποίας το περιεχόμενο είναι κενό. Αυτό συμβαίνει διότι έχει επιλεχθεί από τον τρόπο σύνταξης όλων αυτών των εντολών ότι όσα πατήματα και αν έχει το κουμπί οι ενδείξεις της οθόνης θα επαναλαμβάνονται. Δηλαδή, η εντολή αυτή δεν κλείνει κάπου.

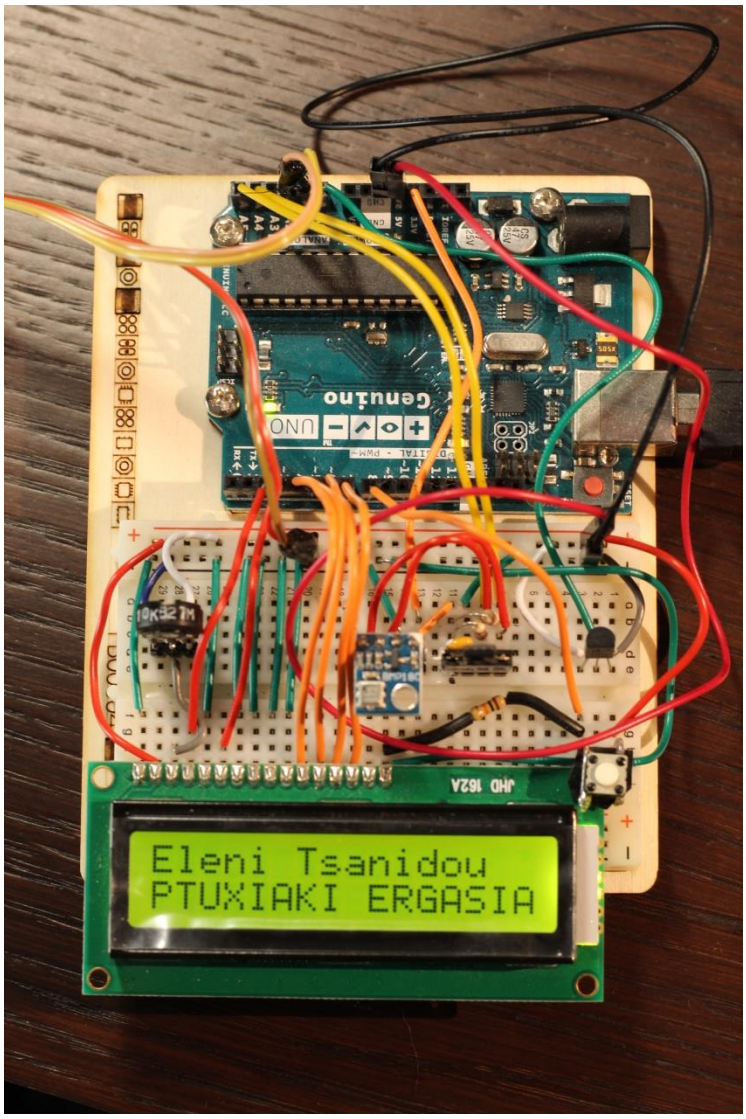
Στην επόμενη φάση πρέπει να δηλωθεί η έναρξη της ρουτίνας μέσα στην setup(). Έτσι, δηλώνεται με την εντολή “lcd.begin(16,2);” όπου σημαίνει ότι ενεργοποιείται η ρουτίνα της οθόνης. Ο αριθμός 16 που βρίσκεται μέσα στην παρένθεση υποδηλώνει ότι χωράνε 16 χαρακτήρες ανά σειρά και ο αριθμός 2 ότι υπάρχουν 2 σειρές. Αμέσως μετά, τρέχει η ρουτίνα χωρίς το πάτημα κουμπιού, δηλαδή όταν το “a” είναι ίσο με μηδέν. Η καθυστέρηση που έχει συνταχθεί σαν εντολή ακριβώς μετά, υπάρχει για να γίνεται εμφανής η ένδειξη που αναγράφεται στην οθόνη.

Έχει συνταχθει μια ρουτίνα για τον αισθητήρα BMP180, η οποία σχετίζεται με την οθόνη.

Είναι μια επανάληψη βρόγχου που επιστρέφει “True” εάν βρεθεί σε σύνδεση ο αισθητήρας, ή αλλιώς “False” , όπου πρέπει να γίνει έλεγχος μετά για την σύνδεση του. Αυτό υποδηλώνεται με την εντολή “if(!bmp.begin())”. Μέσα στο σώμα της συνάρτησης καθαρίζεται η οθόνη από οποιαδήποτε πληροφορία μπορεί να υπάρχει από πριν και τυπώνεται ένα αντίστοιχο μήνυμα στην οθόνη. Η εντολή “while(1)” δηλώνει ότι ισχύει πάντα το “while(1>0)” οπότε η ένδειξη της οθόνης θα ισούται με μηδέν, σαν να αρχικοποιείται πάντα η τιμή “othoni”.

Μέσα στο σώμα της ρουτίνας loop() καλείται η συνάρτηση “Show_Results_LCD();”. Σε αυτήν περιέχονται όλα όσα αναφέρθηκαν παραπάνω. Έχει συνταχθεί για τέσσερα διαφορετικά γράμματα, που όταν πατηθούν, να εμφανίζεται στο σειρακό παράθυρο η αντίστοιχη ένδειξη στην οθόνη LCD. Δηλαδή, διασυνδέθηκε η σειριακή σύνδεση με αυτή της οθόνης. Η εντολή που κάνει ίση με μηδέν την μεταβλητή “dummy” υποδεικνύει ότι είναι άχρηστη εντολή απλά για να μην εμφανίζει τίποτα άλλο στην οθόνη σε σχέση με αυτά τα τέσσερα γράμματα που έχουν δηλωθεί.

Το σώμα της loop() κάνει ασταμάτητες επαναλήψεις. Γι΄αυτό έχει επιλεχθεί να γίνεται εκεί ο έλεγχος για την καταστασή του κουμπιού, αν είναι πατημένο ή όχι, για να μας δίνει τα αποτελέσματα που χρειάζονται. Δηλώνεται η διασύνδεση του κουμπιού με το pin του Aruino που είναι συνδεδεμένο. Γίνεται ένας διαρκής έλεγχος για το αν η κατάσταση του κουμπιού είναι HIGH ή LOW, πατημένο ή όχι. Μέσα σε αυτό τον έλεγχο καταμετριούνται τα πατήματα της οθόνης και προστίθενται. Αν ξεπεράσουν τα τέσσερα πατήματα που έχουμε δηλώσει, τότε μηδενίζει την κατάσταση της οθονης, δηλαδή βγάζει το μήνυμα που αντιστοιχεί στα μηδέν πατήματα.




Εικόνα 17: Κατασκευή με την αρχική ένδειξη της οθόνης LCD

## 2.5. Κατασκευή Ανεμόμετρου

Το ανεμόμετρο, γενικά, μετρά την ταχύτητα του ανέμου και προσδιορίζει τον προσανατολισμό του. Η βασική ιδέα για την κατασκευή του ανεμόμετρου ήταν η αποφυγή αγοράς ενός έτοιμου αισθητηρίου. Ταυτόχρονα, λειτούργησε θετικά στην εξοικίωση σχεδίασης κυκλώματος, πλακετών και στην υλοποίηση τους. Γι΄αυτό το λόγο έγινε η προσπάθεια ενός εικονικού ανεμόμετρου.

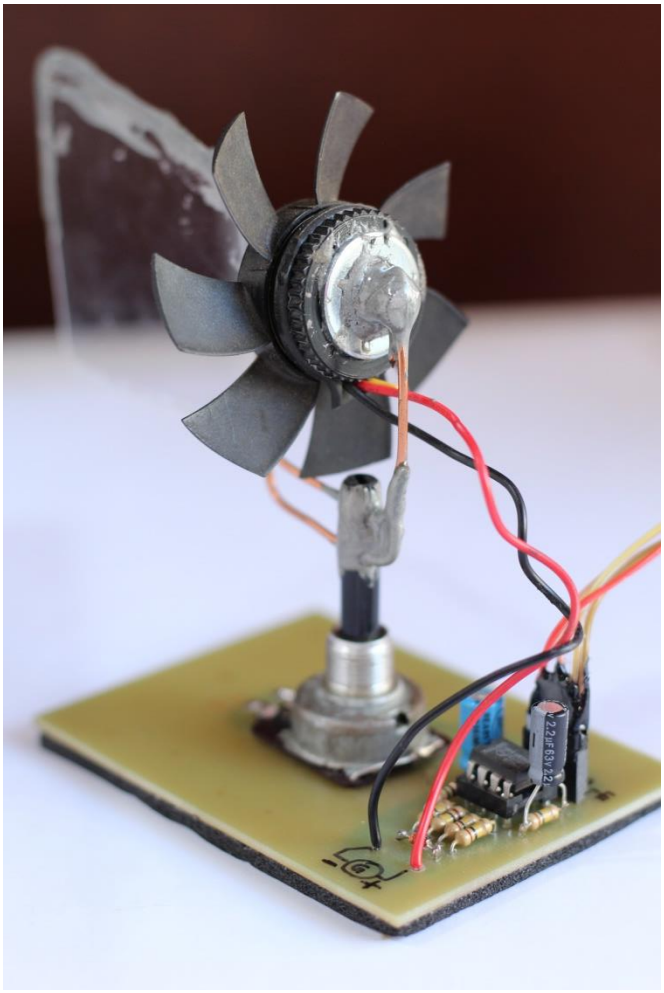
### 2.5.1 Θεωρητικό Υπόβαθρο

Για την δημιουργία ενός ανεμόμετρου χρειάζονται:

- Ένα κοινό ανεμιστηράκι,
- Ένα πτερύγιο για τον προσανατολισμό του ανέμου,
- Μια βάση για να στέκεται η κατασκευή, και όπου είναι εφικτό να σχεδιαστεί και να αντικατασταθεί από την κατάλληλη πλακέτα.

Με βάση τα παραπάνω, είναι χρήσιμο ένα ποτενσιόμετρο το οποίο συνδέει την περιστροφή του πτερυγίου με την βάση του κυκλώματος. Το ανεμιστηράκι συνδέεται με το πτερύγιο έτσι ώστε η μέτρηση που θα πάρει τη δεδομένη χρονική στιγμή να ισχύει για την ταχύτητα και τον προσανατολισμό του ανέμου. Απαιτείται ιδιαίτερη προσοχή στην σύνδεση αυτών των δύο. Πρέπει να υπολογιστεί ο σωστός χώρος ώστε το ανεμιστηράκι να έχει περιθώριο να γυρνάει χωρίς να το εμποδίζει τίποτα. Από το ανεμιστηράκι βγαίνουν δύο καλωδιάκια, το ένα για την γείωσή του και το άλλο για την τάση του. Στην συγκεκριμένη εργασία, επιλέχθηκε ένας ανεμιστήρας από παλαιότερη χρήση όπου προυπήρχαν αυτά τα καλωδιάκια και απλά συνδέθηκαν κατάλληλα στην πλακέτα που δημιουργήθηκε για αυτό τον σκοπό.

Η τάση που δίνει το ανεμιστηράκι αυτό είναι πολύ μικρή με αποτέλεσμα να χρειάζεται πιο δυνατός αέρας για να περιστραφεί και να πάρει μετρήσεις. Επειδή όμως, χρειάζεται να υπάρχουν μετρήσεις και για πιο αδύναμο αέρα επιλέχθηκε η τοποθέτηση ενός ολοκληρωμένου διπλού τελεστικού ενισχυτή, του LM358. Πρόκειται για ενισχυτή τάσης άμεσης σύζευξης με πολύ μεγάλη τιμή ενίσχυσης τάσης, πολύ υψηλή αντίσταση εισόδου και πολύ χαμηλή αντίσταση εξόδου (ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ). Ο ενισχυτής αυτός μεγενθύνει την τάση που λαμβάνει. Αυτό έχει ως αποτέλεσμα την εμφάνιση μετρήσεων ακόμα και μικρών τάσεων, γεγονός που εάν δεν είχε προστεθεί ο ενισχυτής αυτός δεν θα γινόταν αντιληπτό, καθώς για τις μικρές τιμές δεν εμφανίζεται τάση. Η εμφάνιση του ανεμόμετρου φαίνεται στην Εικόνα 18.



Εικόνα 18: Κατασκευή ανεμόμετρου

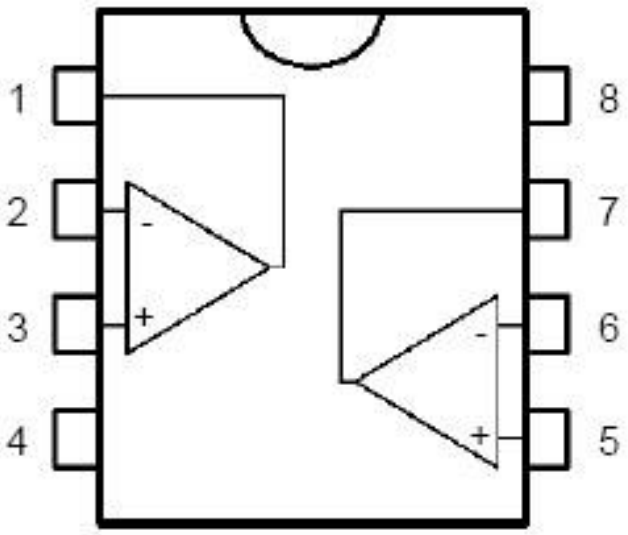
### 2.5.2 Συνδεσμολογία

Για την κατασκευή αυτή έχει συνδεθεί ένα ποτενσιομέτρο με το πτερύγιο του ανεμομέτρου. Σε αυτό έχει στερεωθεί το ανεμιστηράκι, όπως προαναφέρθηκε. Αυτά καταλήγουν σε μια πλακέτα η οποία περιέχει πανω της:

- τέσσερεις αντιστάσεις των 100K,
- μία αντίσταση των 1M,
- μία δίοδο Zenner των 5,1V,
- μία κοινή δίοδο 1N4145,
- ένα ολοκληρωμένο, τελεστικό ενισχυτή, τον LM358,
- έναν ηλεκτρολυτικό πυκνωτή των 2,2μF
- και έναν ακόμα πυκνωτή των 100μF.

Αυτά σχεδιάστηκαν και κολλήθηκαν πάνω στην πλακέτα κατάλληλα.

Το ολοκληρωμένο περιέχει δύο τελεστικούς ενισχυτές, εκ των οποίων χρησιμοποιήθηκε μόνο ο ένας για την ενίσχυση του σήματος. Το περιεχόμενο του φαίνεται στην Εικόνα 19.

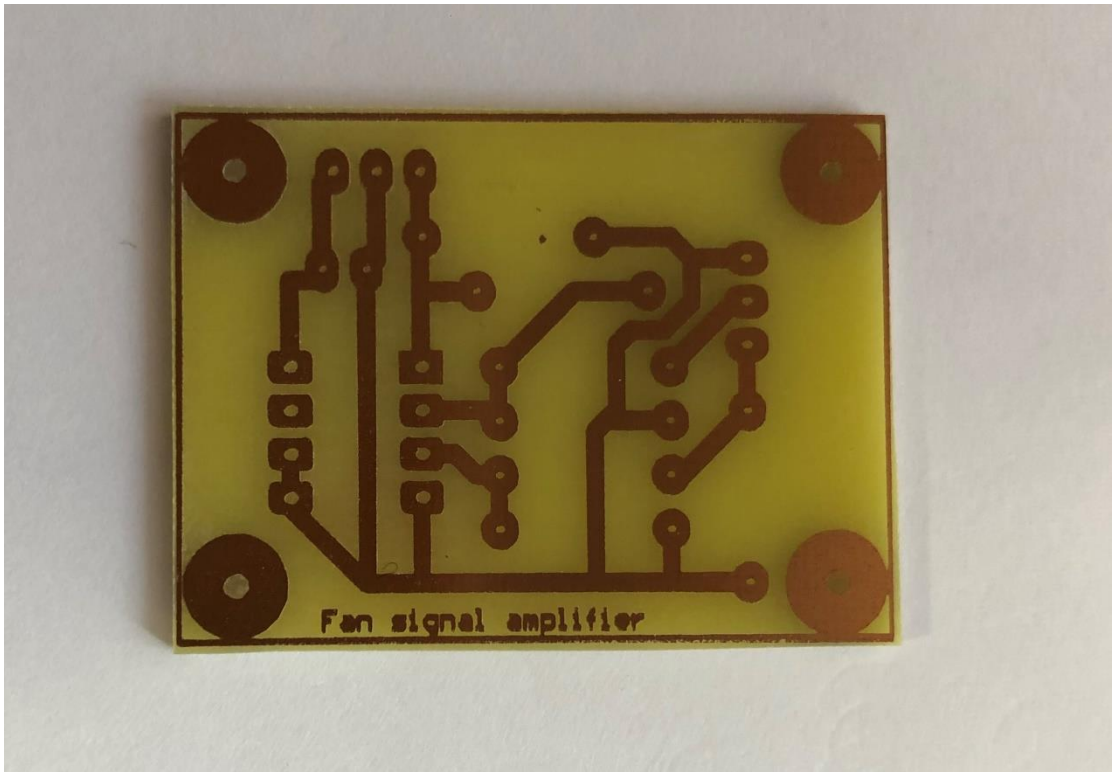



1 - Output 1
2 - Inverting input 1
3 - Non-inverting input 1
4 - $V_{CC}^-$
5 - Non-inverting input 2
6 - Inverting input 2
7 - Ouput 2
8 - $V_{CC}^+$

Εικόνα 19: Ολοκληρωμένο LM358

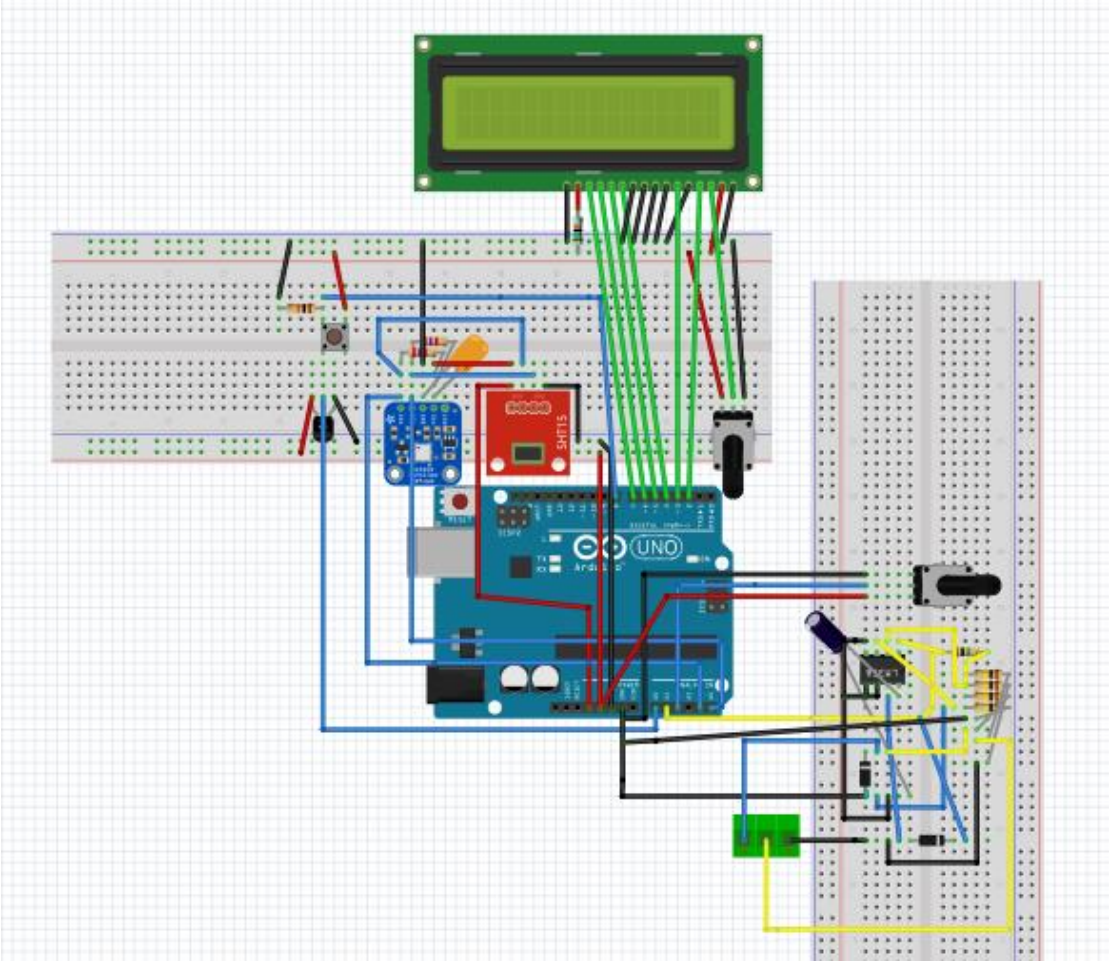
Στα ποδαράκια 1 εώς 4, όπως αριθμούνται και στην Εικόνα, συνδέονται κατάλληλα οι αντιστάσεις. Τα ποδαράκια 6,7,8 δεν συνδέονται κάπου και είναι αυτά του δεύτερου τελεστικού ενισχυτή. Αναλυτικότερα, το πρώτο ποδαράκι συνδέεται με την αντίσταση 1M, το οποίο είναι η έξοδος του τελεστικού ενισχυτή. Ταυτόχρονα, συνδέεται το δεύτερο ποδαράκι στα οποία είναι συνδεδεμένες δύο αντιστάσεις των 100K εκ των οποίων η μία γειώνεται. Το τρίτο ποδαράκι συνδέεται και αυτό με δύο αντιστάσεις των 100K με τη μία να είναι γειωμένη. Η άλλη αντίσταση συνδέεται με την δίοδο Zenner όπου προστατεύει την φορά που λαμβάνεται η τάση. Όλο αυτό συνδέεται με έναν ηλεκτρολυτικό πυκνωτή στο ποδαράκι της τάσης του, ενώ στο άλλο απλά γειώνεται. Από το ποδαράκι ένα, βγαίνει η μετρήσιμη τιμή του αέρα. Στο ποδαράκι με αριθμό 8 συνδέεται ο άλλος ηλεκτρολυτικός πυκνωτής.

Για όλα τα παραπάνω έγινε σχεδίαση πλακέτας όπου φαίνεται στην Εικόνα 20.




**Εικόνα 20: Απεικόνιση κυκλώματος ανεμόμετρου σε πλακέτα**

Στην συνέχεια, αυτή η μικρή κατασκευή ανεμόμετρου συνδέεται με το υπόλοιπο κύκλωμα που έχει κατασκευαστεί. Έχει τοποθετηθεί πάνω στην πλακέτα μια τετράδα με θηλυκά pins, όπου κάθε ποδαράκι βγάζει διαφορετική έξοδο του κυκλώματος. Από το πρώτο βγαίνει η τάση στα 5V, από το δεύτερο βγαίνει ο προσανατολισμός του ανεμού, δηλαδή προς ποιά κατεύθυνση φυσάει. Το τρίτο ποδαράκι βγάζει την μετρήσιμη έξοδο του ανέμου. Τέλος, το τέταρτο ποδαράκι βγάζει την γείωση. Αυτά τα τέσσερα pins συνδέεονται στην κατασκευή του μετεωρολογικού σταθμού. Η τάση και η γείωση πάνε αντίστοιχα στην τάση και γείωση της κατασκευής, και ο προσανατολισμος και η ταχύτητα του ανέμου συνδέεονται αντίστοιχα στα αναλογικά ποδαράκια του Arduino A2 και A1. Θα γίνει πιο κατανοητό με την συνδεσμολογία της κατασκευής από το προγράμμα Fritzing στην επόμενη εικόνα.




Εικόνα 21: Απεικόνιση κυκλώματος με Fritzing

### 2.5.3 Προγραμματισμός ανεμόμετρου

Για την κατασκευή του ανεμόμετρου είναι απαραίτητος και ο κατάλληλος προγραμματισμός του. Η ταχύτητα του ανέμου και ο προσανατολισμός του μετριούνται από αυτή την κατασκευή που δημιουργήθηκε για αυτό το σκοπό. Παρακάτω θα αναλυθεί η λογική του προγραμματισμού του.

```
int fanValue;
unsigned int orient;

int WindSpeed(){
 return analogRead(A1);
 }

int Orientation(){
```

```
  double ox = (double)analogRead(A2);
  ox *= 359;
  ox /= 1023;
  return (int)ox;
}

void Measure_All(){
   fanValue = WindSpeed();
   orient = Orientation();
}
```

Αρχικά, πριν από την συνάρτηση setup() δηλώθηκαν οι μεταβλητές οι οποίες εκπροσωπούν αντίστοιχα την ταχύτητα του ανέμου και τον προσανατολισμό του. Με το “unsigned” δηλώνεται ότι λόγω του ποντενσιόμετρου και της περιστροφής του είναι επιθυμητό οι τιμές που θα μετράει να βγαίνουν χωρίς πρόσημο ώστε να θεωρούνται μόνο θετικές. Στην συνέχεια δημιουργήθηκαν οι κατάλληλες ρουτίνες για τα δύο αυτά στοιχεία χωριστά. Για την ταχύτητα του ανεμόμετρου δημιουργήθηκε η ρουτίνα “WindSpeed” και η μόνη εντολή που διαβάζει η ρουτίνα είναι αυτή που υποδηλώνει με ποιο αναλογικό pin του Arduino συνδέεται. Για τον προσανατολισμό του το περιεχόμενο είναι λίγο διαφορετικό. Ονομάστηκε “Orientation” και στο σώμα αυτής της ρουτίνας πρέπει να διαβαστεί η τάση του ποντενσιόμετρου έτσι ώστε να μετατραπεί ως προσανατολισμός του ανεμόμετρου. Γίνεται μια μετατροπή χωρητικότητας της τιμής που δέχεται για τον προσανατολισμό σε double και ταυτόχρονα δηλώνεται το αναλογικό pin που συνδέεται. Η μεταβλητή αυτή πολλαπλασιάζεται με τον αριθμό 359 και το αποτέλεσμα διαιρείται με το 1023 όπου αναφέρθηκε στο κεφάλαιο 2.3.3 τι υποδηλώνει αυτός ο αριθμός.

Ο συντελεστής 359 αντιστοιχεί στις 360 μοίρες του ορίζοντα (0 - 359). Η άμεση ψηφιοποιημένη ανάγνωση της αναλογικής τάσης που δίνει το ποτενσιόμετρο προσανατολισμού (x = 0 έως 1023) μετατρέπεται σε κλασμα της μέγιστης τιμής του (1023), οπότε πολλαπλασιαζόμενο με τις 360 μοίρες του ορίζοντα μας δίνει τον τρέχοντα προσανατολισμό (ox = 0 – 359°).

Προσανατολισμός = ( μετρούμενη τιμή X 359 μοίρες ) / 1023

Προγραμματιστικά, η παραπάνω διαδικασία υλοποιήθηκε ώς εξής:

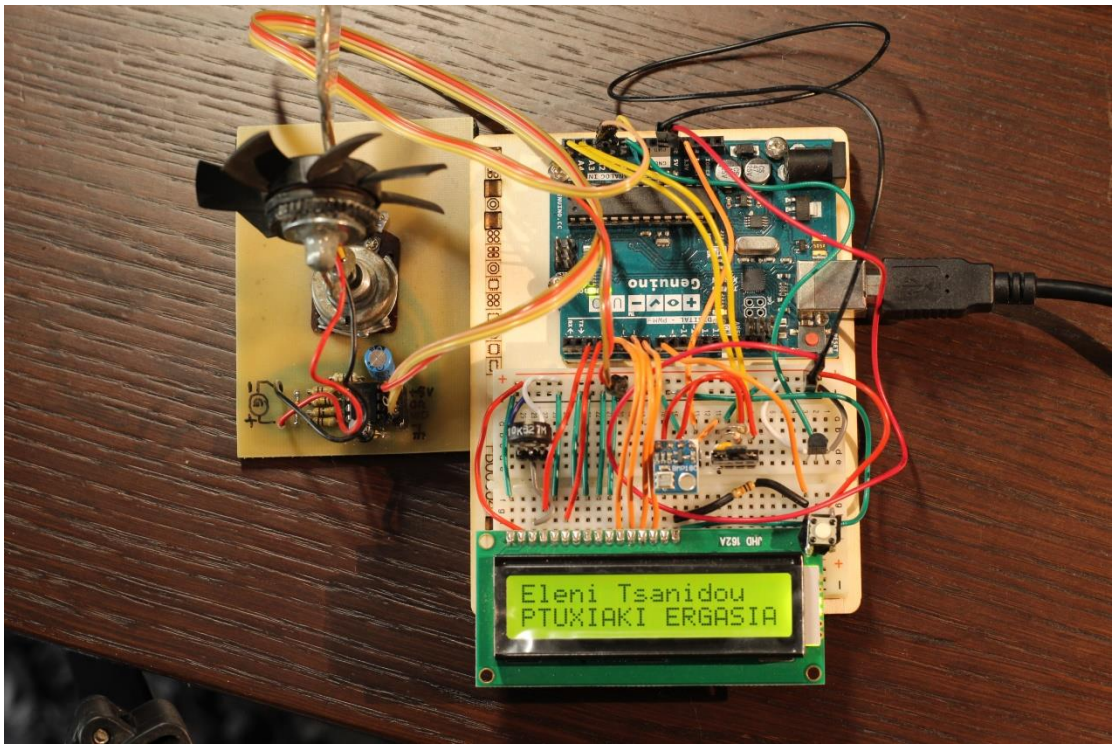
```
double ox =0;  //Εισάγεται μία μεταβλητή μεγάλου εύρους (double)
ox = (double) analogRead(A2);  //Διαβάζεται η αναλογική είσοδος A2 (ποτενσιόμετρο)
ox *= 359; //Πρώτα πολλαπλασιάζεται με τις μέγιστες μοίρες(359, αφού 360° = 0°)
ox /= 1023; //Τέλος, διαιρείται με την μέγιστη δυνατή της αναλογικής εισόδου A2( =  1023)
return (int)ox;  //Επιστρέφεται ο προσανατολισμός σε μοίρες.
```

Ετσι, στο τέλος της ρουτίνας επιστρέφεται η τιμή της μεταβλητής “ox” που έχει οριστεί για τον προσανατολισμό του ανέμου.

Στην συνάρτηση “Measure_All()” ορίζονται οι ρουτίνες που δηλώθηκαν παραπάνω με άλλο όνομα για λόγους ευκολίας.

### 2.5.4 Περίγραφή Κυκλώματος

Στην παρακάτω Εικόνα φαίνεται η τελική μορφή του μετεωρολογικού σταθμού που κατασκευάστηκε.




**Εικόνα 22: Τελική απεικόνιση μετεωρολογικού σταθμού**

# 3. Κατασκευή πλακέτας

Το τελικό στάδιο της κατασκευαστικής εφαρμογής είναι να μπεί σε ένα αεροστεγές κουτί όπου θα προστατεύεται καλύτερα. Λόγω όμως της περίπτωσης οξείδωσης του κυκλώματος, έγινε σχεδίαση όλης της κατασκευής αυτής σε πλακέτα. Έτσι εξοικονομείται χώρος, αποφεύγεται η περίπτωση οξείδωσης του κυκλώματος και καθιστά πιο εύκολη την τοποθέτηση του σε ένα οποιοδήποτε κουτί.

## 3.1. Θεωρητικό υπόβαθρο

Αυτή η ιδέα προήλθε ως επέκταση του ανεμόμετρου. Καθώς η τεχνολογία προχωρά με ραγδαίες ταχύτητες, ο καθένας μπορεί να κατασκευάσει ότι χωράει στη φαντασία του. Έτσι επιλέχθηκε να κατασκευαστεί όλος ο μετεωρολογικός σταθμός σε μία μόνο πλακέτα. Αυτή η κατασκευή προυποθέτει γνώσεις σχεδίασης και υλοποίησης της πλακέτας, καθώς και κάποια συγκεκριμένα εξαρτήματα.

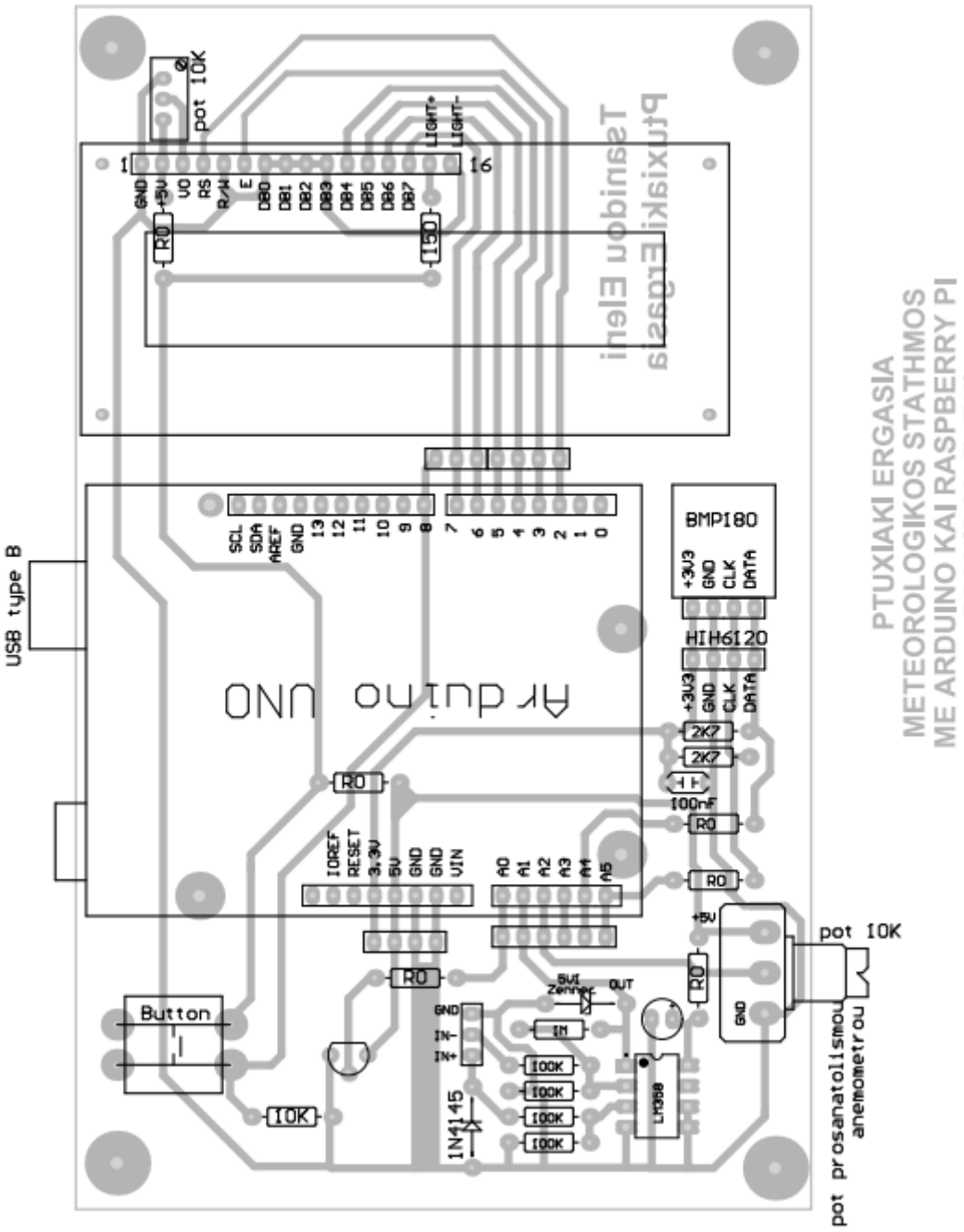
Με αυτόν τον τρόπο κατασκευής πλακέτας δεν υπάρχει πλέον φόβος μήπως βγεί από τη θέση του κάποιο καλωδιάκι, μήπως κάποιο δεν είναι σωστά τοποθετημένο στο pin του, μήπως κάποιο έχει κουνηθεί και δεν εφαρμόζεται πλήρως. Αποφεύγονται δηλαδή κάποιες πιθανότητες κάτι να πάει στραβά. Σαφώς, η κατασκευή της πλακέτας δεν είναι μονόδρομος. Δηλαδή αν η κατασκευή σταματούσε εδώ, χωρίς την υλοποίηση της πλακέτας, το σύστημα λαμβάνει μετρήσεις και εμφανίζει τα επιθυμητά αποτελέσματα.

## 3.2. Τρόπος κατασκευής

### 3.2.1 Σχεδίαση πλακέτας

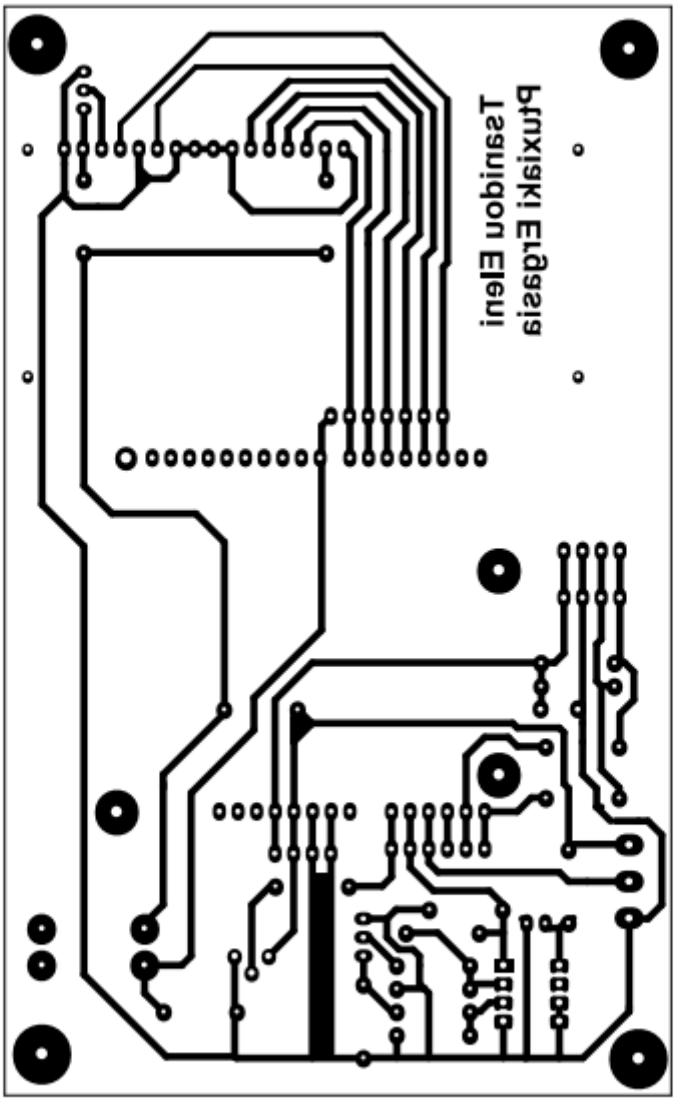
Για την κατασκευή της πλακέτας χρειάστηκε πρώτα να γίνει μια προσομοίωση σε ένα πρόγραμμα, το Altium. Φυσικά υπάρχουν διάφορα προγράμματα σχεδίασης, επιλέχθηκε το συγκεκριμένο για λόγους ακρίβειας. Έχει απεριόριστες δυνατότητες, αλλά για την κατασκευή αυτή έγινε χρήση κάποιων βασικών λειτουργιών του. Στο συγκεκριμένο πρόγραμμα υπάρχουν κάποιες έτοιμες βιβλιοθήκες, αλλά υπάρχει και η δυνατότητα σχεδίασης εξαρτημάτων με μεγάλη ακρίβεια. Μετρήθηκαν οι διαστάσεις των εξαρτημάτων που δεν υπήρχαν στο πρόγραμμα, και σχεδιάστηκαν με ακρίβεια. Τοποθετήθηκαν όλα σε συγκεκριμένες θέσεις ώστε να γίνεται η πιο μικρή διαδρομή χαλκού. Βέβαια είναι σημαντικό να τονιστεί ότι απαραίτητη προυπόθεση είναι να μην πλησιάζουν αρκετά κοντά οι διαδρομές και τα εξαρτήματα. Αυτό συμβαίνει διότι μπορεί στην αποχάλκωση να έρθουν πολύ κοντά δύο διαδρομές με αποτέλεσμα να ενωθούν. Επίσης, είναι καλό να αναφερθεί ότι για την σχεδίαση οποιουδήποτε εξαρτήματος πρέπει να ακολουθείται από μεγάλη ακρίβεια. Έτσι μπορεί να υπολογιστεί ακριβώς ο χώρος που θα απαιτηθεί για την τοποθέτηση του. Αυτή η ακρίβεια επιτυγχάνει την αποφυγή λάθους στο χώρο που απομένει αλλά ταυτόχρονα και την αποφυγή για σπατάλη χώρου. Το σχέδιο που δημιουργήθηκε για τον μετεωρολογικό σταθμό απεικονίζεται στην Εικόνα 23, όπου φαίνονται όλα τα εξαρτήματα που χρησιμοποιήθηκαν. Αυτό εξυπηρετεί στη διευκόλυνση αυτού που θα

κατασκευάσει την πλακέτα και θα πρέπει να κολλήσει τα εξαρτήματα ώστε να γνωρίζει με σιγουριά που μπαίνει τι. Είναι δηλαδή ένα σχέδιο – οδηγός για την κατασκευή.




Εικόνα 23: Απεικόνιση πλακέτας μέσω Altium

Για την εμφάνιση μόνο των αγωγών (χαλκόδρομων) θα πρέπει να χρησιμοποιηθεί άλλη επιλογή εκτύπωσης. Με το ίδιο ακριβώς σχέδιο που έχει δημιουργηθεί στην Εικόνα 23, αλλάζω την επιλογή των στρώσεων (layers), επιτρέποντας την εκτύπωση του κάτω στρώματος (Bottom Layer) και αποκλείοντας την εμφάνιση του περιγράμματος των εξαρτημάτων (Top Overlay). Το σχέδιο που θα εμφανιστεί θα είναι αυτό που θα αποτυπωθεί στην πλακέτα και απεικονίζεται στην Εικόνα 24. Εφόσον όλα έχουν σχεδιαστεί με μεγάλη ακρίβεια, δεν θα υπάρξει πρόβλημα στην αποτύπωση του. Όλα έχουν τυπωθεί ανάποδα, με σκοπό στην αποτύπωση του σχεδίου πάνω στην πλακέτα, να είναι ευανάγνωστα τα στοιχεία που αναγράφονται. Αυτό το σχέδιο εκτυπώνεται σε μια διαφάνεια.



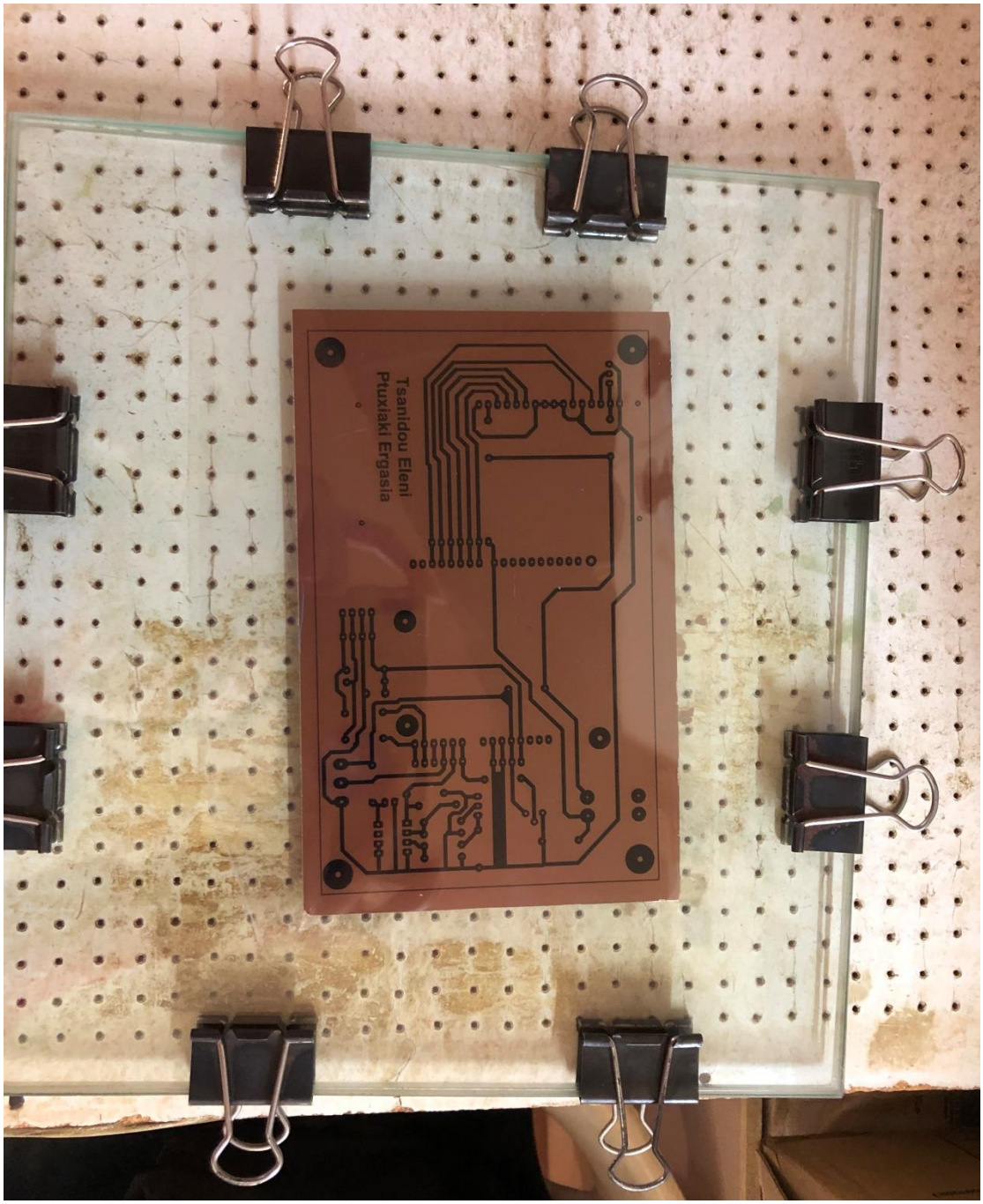

Εικόνα 24: Απεικόνιση σχεδίου για την εκτύπωση της πλακέτας

### 3.2.2 Υλοποίηση κατασκευής

#### *3.2.2.1 Εμφάνιση πλακέτας*

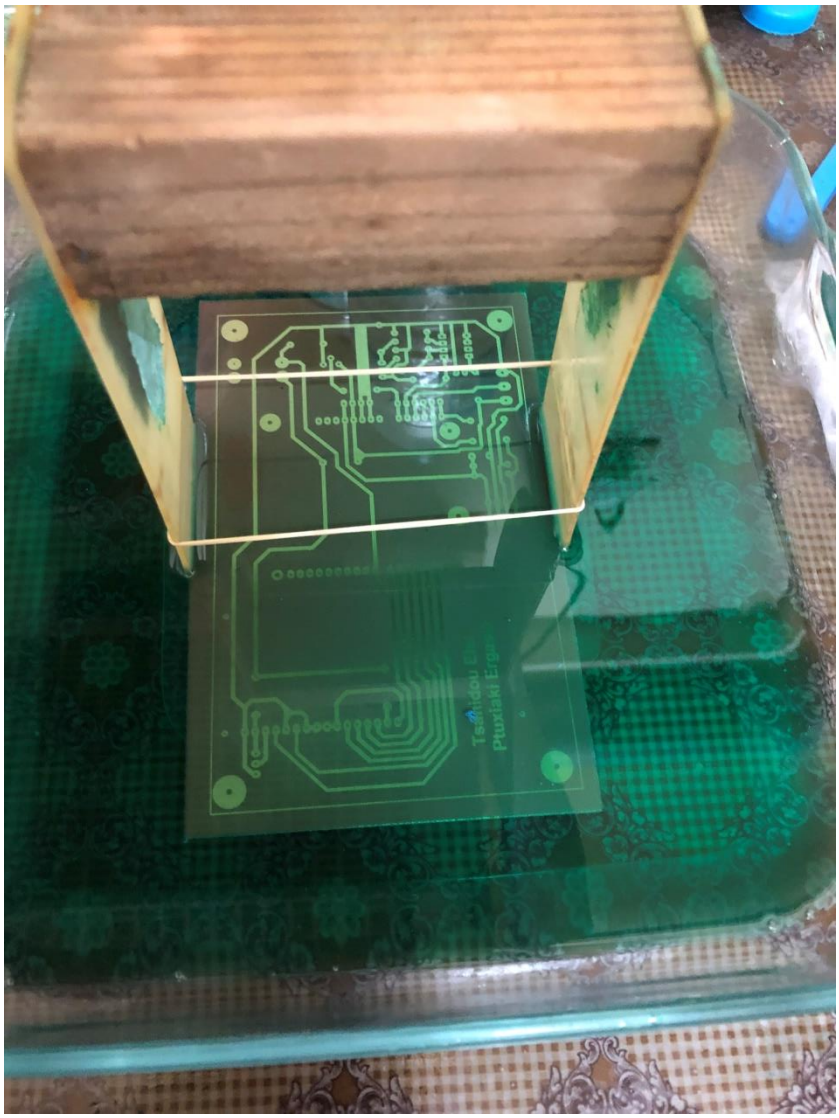
Το πρώτο βασικό που χρειάζεται να γίνει είναι να μετρηθούν σωστά οι διαστάσεις της πλακέτας. Αυτό μπορεί να γίνει από το πρόγραμμα που σχεδιάστηκε η πλακέτα. Εφόσον οι

διαστάσεις είναι γνωστές, επιλέγεται η κατάλληλη πλακέτα που πληρεί τις σωστές αναλογίες για την εκτύπωση της. Η πλακέτα που χρησιμοποιήθηκε αποτελείται από ένα λεπτό στρώμα χακλού πάχους 35μm, κολλημένο πάνω σε εποξικό φύλλο πάχους 1,5mm. Η επιφάνεια του χαλκού είναι καλυμμένη με «θετική» φωτοπαθής ουσία (Positive). Το προστατευτικό αυτό φιλμ αποσυντίθεται αν εκτεθεί σε υπεριώδη ακτινοβολία, ενώ όπου σκιάζεται παραμένει κολλημένο στην ελεύθερη επιφάνεια του χαλκού προστατεύοντάς τον. Τοποθετείται λοιπόν η κενή πλακέτα με το εκτυπωμένο σχέδιο ανάμεσα σε δύο πλάκες γυαλιού. Για την αποφυγή λάθους στην εκτύπωση είναι καλό οι πλάκες αυτές να είναι πιασμένες γερά με την πλακέτα, όπως φαίνεται στην Εικόνα 25.



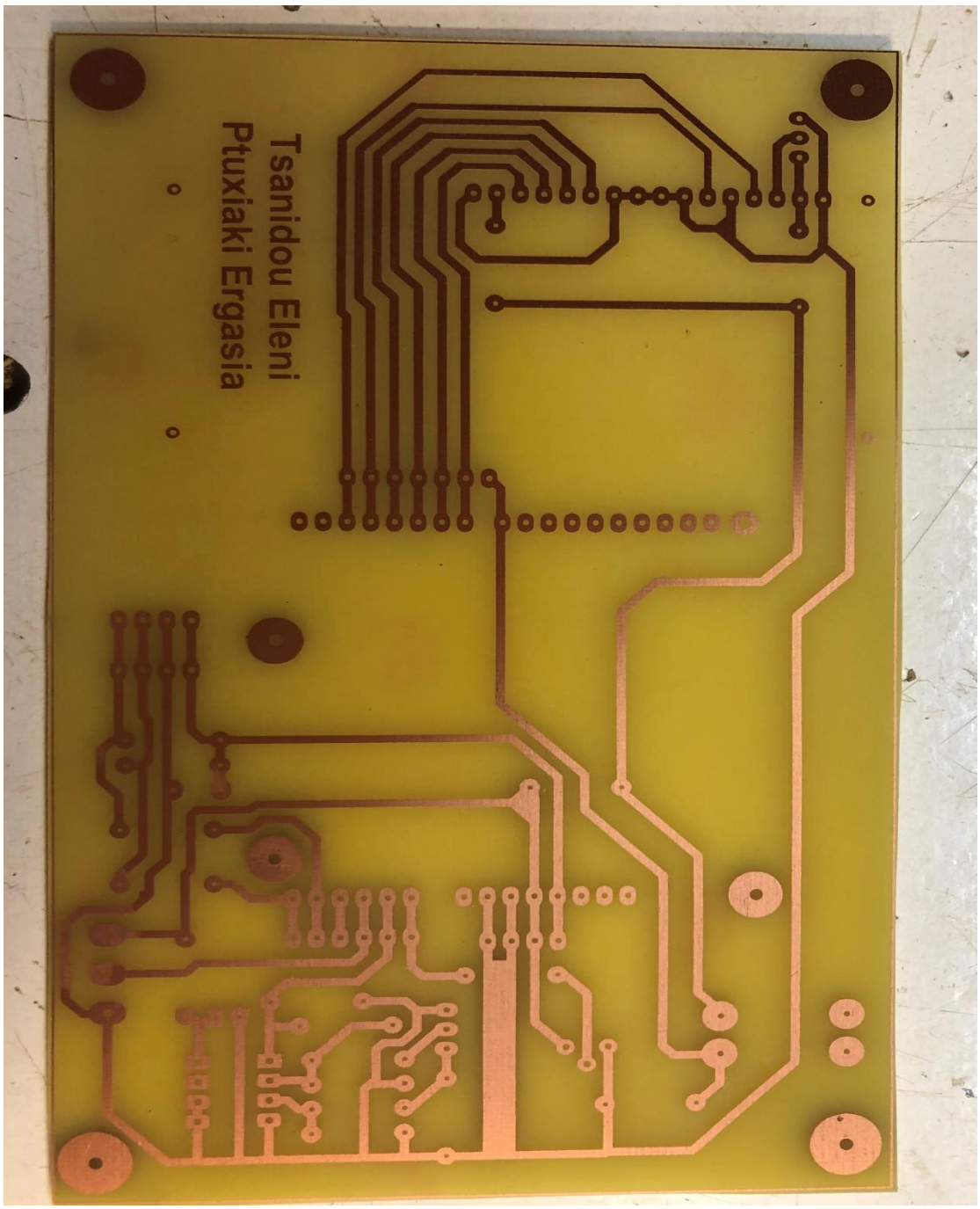

**Εικόνα 25: Απεικόνιση πλακέτας**

Όλο αυτό μπαίνει σε έναν κλειστό χώρο με υπεριώδη ακτινοβολία για δύομιση λεπτά. Όταν βγεί η πλακέτα από εκεί, δεν φαίνεται να έχει απεικονιστεί κάτι. Γιαυτό στην συνέχεια, δημιουργείται ένα μείγμα με νερό και καυστική σόδα(αποφρακτικό), όπου τοποθετείται μέσα η πλακέτα. Παρατηρείται με την πάροδο των πρώτων δευτερολέπτων σιγά σιγά η εμφάνιση της πλακέτας. Αφού εμφανιστεί πλήρως το σχέδιο, η πλακέτα τοποθετείται μέσα σε ένα μείγμα υδροχλωρικού οξέως(κεζαπ) με περιντρόλ(οξυζενέ), όπως φαίνεται στην Εικόνα 26. Με αυτόν τον τρόπο αποχαλκώνεται η πλακέτα, δηλαδή μένει χαλκός μόνο όπου είναι απαραίτητος, προστατευμένος από το φωτοπαθές φιλμ. Στα υπόλοιπα σημεία που η καυστική σόδα απομακρύνει την φωτισμένη φωτοπαθή, ο χαλκός διαλύεται από τα χημικά διαλύματα που προαναφέρθηκαν.



Εικόνα 26: Αποχάλκωση της πλακέτας

Αυτή η διαδικασία που προηγήθηκε παραπάνω είναι γνωστή ως «Μέθοδος εμφάνισης και αποχάλκωσης της πλακέτας». Αφού βγει από αυτά τα μείγματα, η πλακέτα παρουσιάζει την μορφή που φαίνεται στην Εικόνα 27.

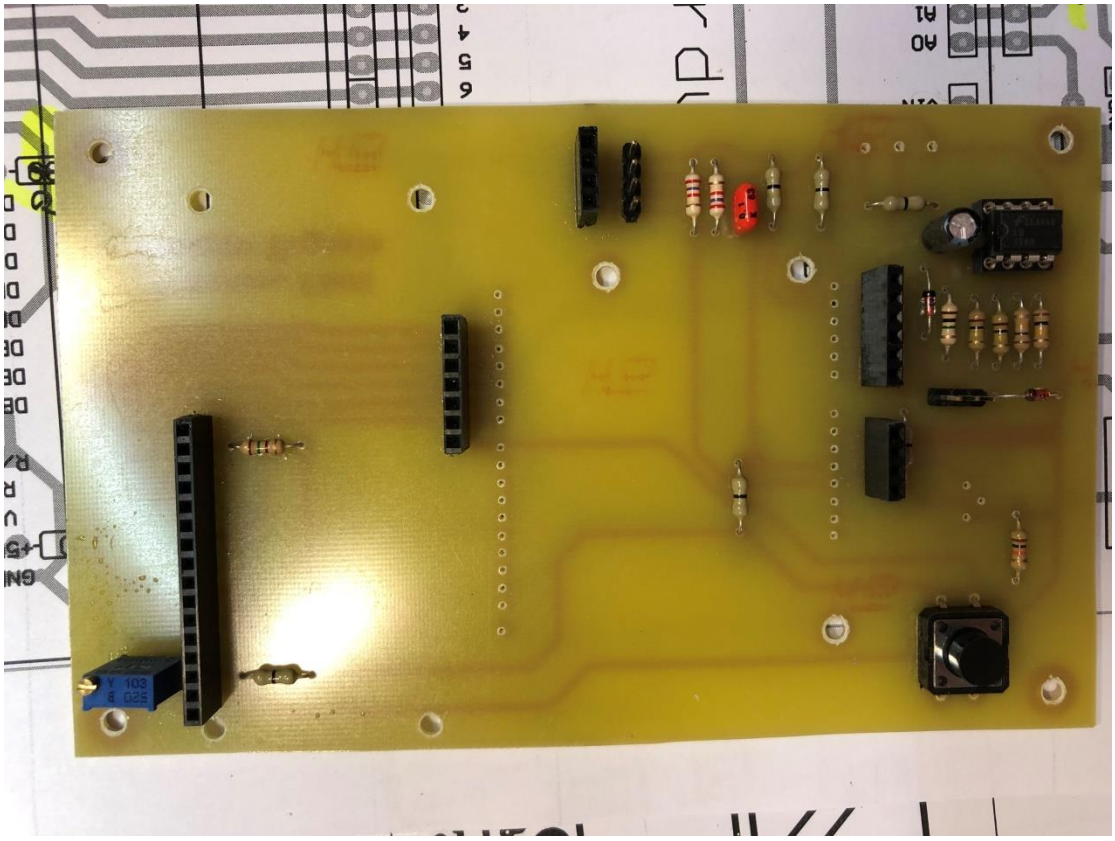



**Εικόνα 27: Τελική εμφάνιση πλακέτας**

### *3.2.2.2 Ολοκλήρωση πλακέτας*

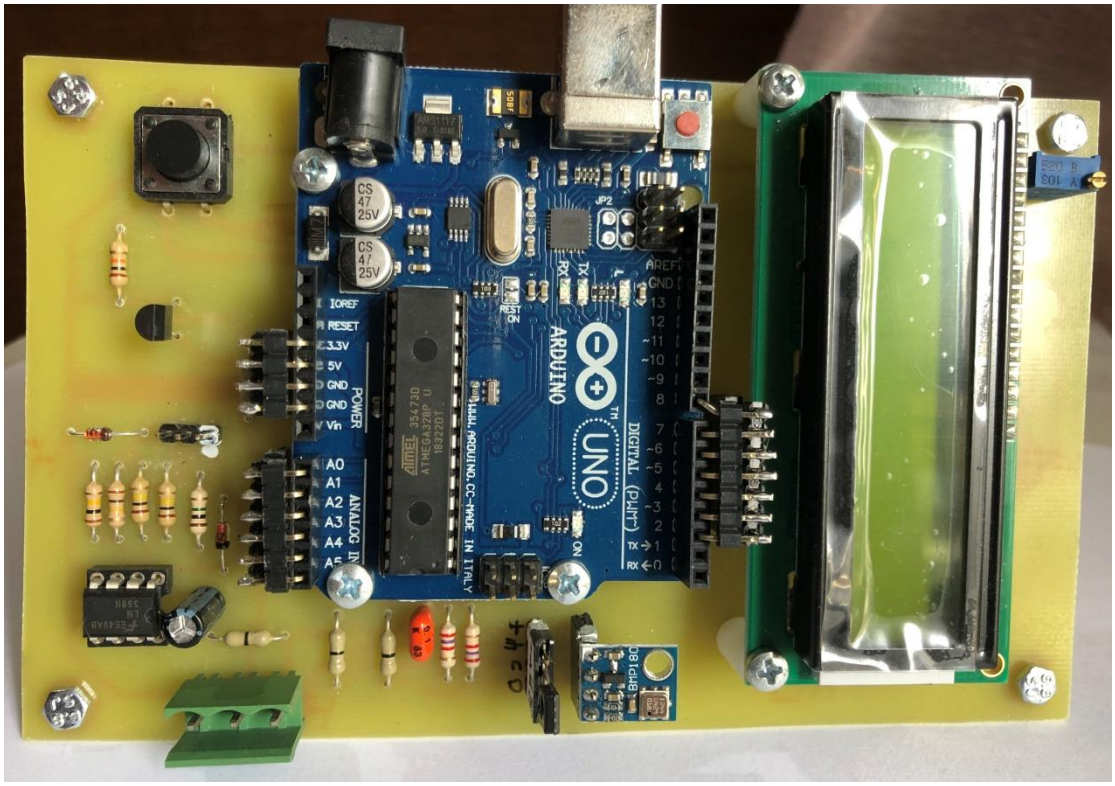
Για την ολοκλήρωση της πλακέτας, όπου χρειάστηκε ανοίχτηκαν οι κατάλληλες τρύπες με διάμετρο τα σωστά χιλιοστά που είχαν καθοριστεί από το πρόγραμμα. Χρειάστηκαν επίσης

να συγκολληθούν κάποια εξαρτήματα, όπως αντιστάσεις, ποτενσιόμετρα κλπ. Για την συγκόλληση των αντικειμένων χρειάστηκε ένα κολλητήρι και καλάι. Περάστηκαν τα εξαρτήματα από τις τρύπες που ανοίχτηκαν, κόπηκαν εκεί που χρειάστηκε ώστε να μην προεξέχουν τα ποδαράκια, και τέλος συγκολλήθηκαν με προσοχή. Όπου θεωρήθηκε χρήσιμο συγκολλήθηκαν θηλυκά pins για την ανεξάρτητη τοποθέτηση και αφαίρεση των εξαρτημάτων.



Εικόνα 28: Στάδιο συγκόλλησης εξαρτημάτων

Σαν τελικό στάδιο τοποθετούνται η οθόνη LCD και ο Arduino. Οπότε η πλακέτα παίρνει την μορφή της Εικόνας 29.




**Εικόνα 29: Τελική μορφή πλακέτας**

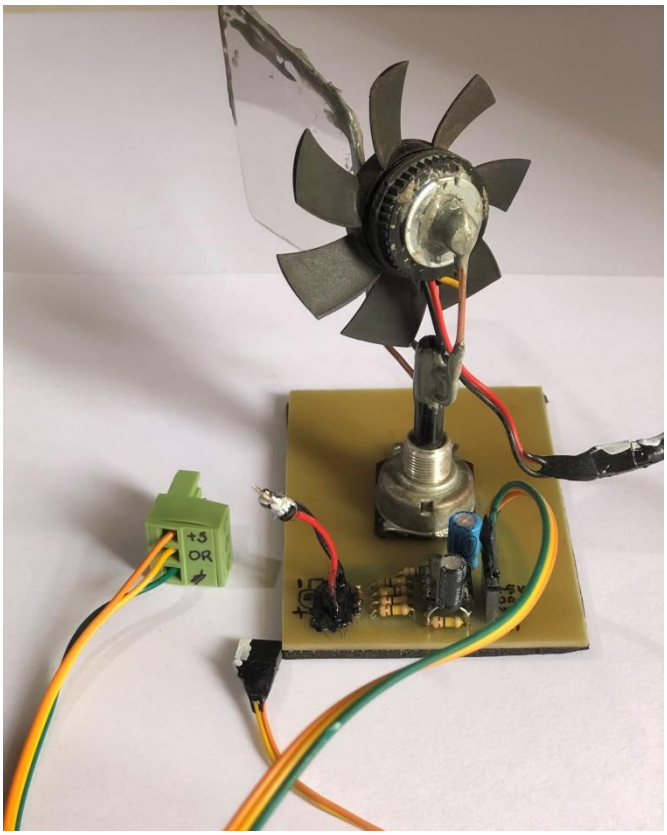
### 3.2.3 Προσαρμογή του ανεμόμετρου

Μετά την κατασκευή της πλακέτας αποφασίστηκε να αξιοποιηθεί το ίδιο ανεμόμετρο με μια μικρή τροποποίηση. Δηλαδή να προσαρμοστεί κατάλληλα με κάποιο τρόπο και στις δύο κατασκευές που έγιναν.

Για αρχή κόπηκαν στη μέση τα καλώδια τάσης και γείωσης του ανεμόμετρου. Συγκολλήθηκαν ανάλογα με τις ανάγκες της κατασκευής θηλυκά και αρσενικά pins όπου ήταν απαραίτητο, ώστε όποτε είναι επιθυμητό να γίνεται η συνδεσμολογία για την πρώτη κατασκευή (raster) ή για την δεύτερη(πλακέτα). Στην περίπτωση που το ανεμόμετρο συνδέεται με την πλακέτα, δημιουργήθηκε μια προέκταση καλωδίων τάσης και γείωσης που θα συνδέονται με αυτά της πλακέτας. Δηλαδή το καλώδιο της τάσης συνδέεται με την τάση που δίνουμε στην πλακέτα για το ανεμόμετρο και το καλώδιο της γείωσης με το αντίστοιχο που βρίσκεται πάνω στην κατασκευή. Από την τάση αυτή μετριέται η ταχύτητα

του ανέμου. Πιο αναλυτικά, η πρώτη υποδοχή είναι για την παροχή τάσης 5V όπου συνδέεται στο ένα άκρο του ποντεσιομέτρου του προσανατολισμού. Στην περίπτωση του raster παρέχει τάση λειτουργίας στον εξωτερικό τελεστικό ενισχυτή. Η δεύτερη υποδοχή δίνει την τάση προσανατολισμού του ποντεσιομέτρου. Η τρίτη είναι η έξοδος του εξωτερικού τελεστικού ενισχυτή, δηλαδή της έντασης του ανέμου, που χρησιμοποιείται μόνο στην σύνδεση της κατασκευής του raster. Στην ολοκληρωμένη πλακέτα ενσωματώθηκε άλλος τελεστικός ενισχυτής στον οποίο συνδέονται απευθείας οι ακροδέκτες του ανεμόμετρου.Η τέταρτη υποδοχή είναι η γείωση του κυκλώματος του εξωτερικού τελεστικού ενισχυτή και συνδέεται στο άλλο άκρο του ποτενσιόμετρου προσανατολισμού.

Για τις ανάγκες του ανεμόμετρου η υποδοχή των τεσσάρων pins που είχε τοποθετηθεί για την ταχύτητα και τον προσανατολισμό πάνω στην κατασκευή του ανεμομέτρου τροποποιήθηκε αχρηστεύοντας από εκεί την υποδοχή της ταχύτητας καθώς μετριέται πλέον από αλλού. Πάνω στην πλακέτα τοποθετήθηκε μια κλέμα, δηλαδή ένας μονωμένος σύνδεσμος των ηλεκτρικών καλωδίων, για την εναλλαγή των τεσσάρων υποδοχών σε τρεις. Αντίστοιχα συνδέθηκαν αυτά τα τρία καλώδια του ανεμόμετρου, τάση, προσανατολισμός και γείωση στην πλακέτα.



**Εικόνα 30: Τελική μορφή του ανεμόμετρου**

# 4. Περιγραφή Λογισμικού

Σε αυτό το κεφάλαιο περιγράφονται οι κώδικες που συντάχθηκαν για την υλοποίηση των προγραμμάτων ώστε να εμφανίζονται τα δεδομένα που ζητήθηκαν.

## 4.1. Για το Arduino Uno

Για τον κώδικα του Arduino Uno αναφέρθηκαν και σε προηγούμενες ενότητες κάποιες σημαντικές εντολές που καθορίζουν κάποιες ενέργειες του. Ωστόσο, παρακάτω επιχειρείται μια ολοκληρωμένη περιγραφή της λογικής ροής του.

Καταρχήν, ενεργοποιήθηκε η βιβλιοθήκη που αφορά την οθόνη LCD και δηλώθηκαν οι θύρες που συνδέουν την οθόνη με το Arduino. Κατόπιν, ενεργοποιήθηκε η επικοινωνία I2C, καθώς και οι απαραίτητες βιβλιοθήκες για τους αισθητήρες και δηλώθηκαν οι μορφές των μεταβλητών που χρησιμοποιούνται στο πρόγραμμα.

Ακολουθούν οι διάφορες υπορουτίνες, με την βοήθεια των οποίων εκτελούνται ορισμένες ειδικές εργασίες, οπότε αυτές καλούνται μέσα από το κυρίως πρόγραμμα :

- AnalogTemp() : Διάβασμα της αναλογικής εξόδου του TMP36 και μετατροπή σε βαθμούς Κελσίου
- WindSpeed() : Διάβασμα της τάσης που παράγει το ανεμόμετρο
- Orientation() :Διάβασμα του ποτενσιομέτρου προσανατολισμού
- HIH_Values() : Διάβασμα υγρασίας & θερμοκρασίας από το HIH6120
- BMP_180_Temp() : Διάβασμα θερμοκρασίας από το BMP_180
- BMP180_Altitude() : Διάβασμα υψομέτρου από το BMP_180
- BMP180_Pressure() : Διάβασμα ατμοσφοσφαιρικής πίεσης από το BMP_180
- Create_Pack() : Δημιουργία πακέτου δεδομένων για σειριακή αποστολή.

Προκειμένου το αποστελλόμενο πακέτο να έχει σταθερά αναγνωρίσιμη δομή :

- προτάσσεται ένας χαρακτήρας εκκίνησης (#)
- στη συνέχεια παρατίθονται οι μετρηθείσες τιμές σε προκαθορισμένη σειρά, διαχωρισμένες με έναν κενό χαρακτήρα (space)
- και καταλήγεται με έναν χαρακτήρα τερματισμού ($).

- Show_Results_LCD(a) : Εμφάνιση δεδομένων στην LCD οθόνη, σε διαφορετικές μορφές (παράθυρα), οι οποίες επιλέγονται βάσει της τιμής της μεταβλητής a.
- Measure_All() : Γενική υπορουτίνα, η οποία καλεί εν σειρά τις υπορουτίνες που διαβάζουν όλες τις μετεωρολογικές παραμέτρους.

Η λειτουργία του Arduino ξεκινά πάντοτε με την setup() συνάρτηση, η οποία περιέχει την αρχικοποίηση των μεταβλητών και εκτελεί τις απαραίτητες λειτουργίες εκκίνησης συσκευών. Σε αυτή δηλώνεται η έναρξη εμφάνισης δεδομένων της οθόνης με μια καθυστέρηση ώστε να είναι εμφανή τα αποτελέσματα που θα δείχνει. Επίσης, δηλώνεται η

έναρξη της σειριακής επικοινωνίας και της I2C. Επιπρόσθετα, γίνεται ένας μικρός έλεγχος για τον αισθητήρα BMP180 για το αν βρέθηκε συνδεδεμένος ή όχι. Αν δεν είναι συνδεδεμένος βγαίνει αντίστοιχο μήνυμα. Ακολούθως, εκκινεί η συνάρτηση loop(). Πρόκειται για την βασική, διαρκώς ανακυκλούμενη αλληλουχία ενεργειών, ώς εξής:

- Ελέγχεται η σειριακή σύνδεση για τυχόν παραληφθέν μήνυμα. Αν διαπιστωθεί ότι έλαβε τον χαρακτήρα 'A', τότε εκτελεί πλήρη σειρά μετρήσεων(Measure_All), δημιουργεί πακέτο δεδομένων(Create_Pack), το στέλνει σειριακά (Serial.println) καί δηλώνει στη LCD οθόνη την ενέργεια. Εάν λάβει τούς χαρακτήρες 'B' , 'C' , ή 'D', εμφανίζει στην LCD οθόνη διάφορα παράθυρα δεδομένων.
- Ελέγχεται η κατάσταση ενος κουμπιού. Εάν διαπιστωθεί ότι είναι πατημένο, εκτελείται πλήρης σειρά μετρήσεων(Measure_All) καί εμφανίζεται στην LCD οθόνη το επόμενο παράθυρο δεδομένων, καλώντας την Show_Results_LCD(a). Με κάθε πάτημα, η μεταβλητή a παίρνει την αμέσως μεγαλύτερη τιμή, ανακυκλούμενη (a=0 >> a=1 >> a=2 >> a=0 >>...). Ετσι, καλείται διαφορετική οθόνη με κάθε πάτημα.

Μετά την εκτέλεση των παραπάνω, το πρόγραμμα περιμένει να ελευθερωθεί το κουμπί και στη συνέχεια περιμένει για επιπλέον 250 msec, προκειμένου να αποφευχθεί ανεπιθύμητη επανάληψη της παραπάνω διαδικασίας, αλλά και να αποσβεστούν τυχόν μηχανικές ταλαντώσεις του κουμπιού. Η προφύλαξη αυτή συνήθως καλείται “debounce”.

## 4.2. Για το Raspberry Pi

Σε αυτή την ενότητα αναλύονται οι λειτουργίες του προγράμματος Python για το Raspberry Pi. Αρχικά, αξίζει να αναφερθεί ότι πριν την σύνταξη του προγράμματος έτρεξαν κάποιες εντολές στο παράθυρο command του Raspberry. Συγκεκριμένα:

- pip install pynput, για το πληκτρολόγιο που συνδέεται στο Raspberry,
- pip install pyserial, για την σειριακή επικοινωνία,
- sudo apt-get install python-serial, για την σειριακή επικοινωνία,
- sudo pip install pyserial, για την σειριακή επικοινωνία,
- sudo apt-get install Arduino, για την ενημέρωση του Raspberry Pi για το Arduino,
- git clone https://github.com/adafruit/Adafruit_Python_DHT.git, για την διαδικτυακή βάση δεδομένων,
- sudo apt-get install build-essential python-dev, για την διαδικτυακή βάση δεδομένων,
- sudo python setup.py install, για την διαδικτυακή βάση δεδομένων.

Αφού έτρεξαν με επιτυχία, είναι πλέον ενεργοποιημένες οι βιβλιοθήκες που προστέθηκαν στο πρόγραμμα:

- import serial, βιβλιοθήκη για την σειριακή επικοινωνία,
- import time, βιβλιοθήκη ενημέρωσης χρόνου,
- import datetime, ενημέρωση ημερομηνίας,
- import os, αυτή και η επόμενη βιβλιοθήκη αφορούν τον τρόπο χρήσης του λειτουργικού συστήματος,
- import sys

- import urllib, οι υπόλοιπες βιβλιοθήκες είναι για την διαδικτυακή βάση,
- import urllib2,
- import urllib.parse,
- import urllib.request,
- import urllib.error.

Το πρόγραμμα που αναφέρεται σε αυτή την ενότητα αφορά την επικοινωνία του Raspberry Pi με το Arduino και τις ενέργειες που γίνονται όταν αυτό τρέχει. Μόλις πατηθεί το κατάλληλο κουμπί για να ξεκινήσει το πρόγραμμα γίνεται ένας έλεγχος όπου βλέπει αν είναι έγκυρη η σειριακή επικοινωνία μεταξύ Raspberry Pi και Arduino. Στον κώδικα έχει δηλωθεί ποια θύρα του Arduino είναι συνδεδεμένη με την εντολή στο παράθυρο cmd "ls /dev/tty". Αμέσως μετά εμφανίζεται με αντίστοιχο μήνυμα ποια θύρα είναι ενεργή για οπτικούς λόγους. Εάν όμως, δεν υπάρχει σωστή σύνδεση εκτυπώνεται στην οθόνη ένα μήμυμα όπου δηλώνει το πρόβλημα στην σύνδεση και τερματίζει το πρόγραμμα. Στην περίπτωση της σωστής σύνδεσης, με την εντολή "ser.write(b'A')" γράφεται το περιεχόμενο που υπάρχει μέσα στα μονά εισαγωγικά ως ASII κώδικας. Αυτό συμβαίνει διότι τα μονά εισαγωγικά υποδηλώνουν την μορφή byte και όχι string. Δηλαδή, καταγράφεται το περιεχόμενο του αρχείου που στέλνει σειριακά από το Arduino ανά σειρά  τα δεδομένα, καθώς είναι δηλωμένο με σειριακή επικοινωνία.

Στην συνέχεια εμφανίζεται ένα μήνυμα στην οθόνη όπου δίνεται η επιλογή να επιλεχθεί η εμφάνιση όλων των αρχείων που υπάρχουν πατώντας το γράμμα "a". Εφόσον πατηθεί το "a", το πρόγραμμα μπαίνει σε έναν βρόχο επανάληψης, όπου εμφανίζονται όλοι οι φάκελοι που είναι αποθηκευμένοι με την κατάληξη ".txt" είτε με κεφαλαία είτε με μικρά.

Αμέσως μετά από αυτό, βγαίνει ένα μήνυμα όπου δίνει την δυνατότητα να επιλέξεις έναν φάκελο γράφοντας την ημερομηνία. Γίνεται έλεγχος εγκυρότητας της ημερομηνίας και αν υπάρχει τέτοιος φάκελος γίνεται έλεγχος στην κατάληξη που είναι αποθηκευμένο το αρχείο. Αν δεν είναι της μορφής ".txt", τότε το αποθηκεύει εκείνη την στιγμή με αυτή. Εφόσον γίνουν αυτοί οι έλεγχοι, ανοίγεται ο φάκελος που ζητήθηκε και εμφανίζεται το περιεχόμενο του. Εάν όμως, δεν είναι έγκυρη η ημερομηνία τότε τυπώνεται το αντίστοιχο μήνυμα όπου ενημερώνει ότι δεν υπάρχει αρχείο με την ημερομηνία που καταγράφτηκε. Σε όλη αυτή τη διαδικασία δίνεται η επιλογή πατώντας από το πληκτρολόγιο ctrl + c να τερματίσει το πρόγραμμα.

Εάν δεν πατηθεί το "a", ανοίγεται ένα αρχείο που έχει δημιουργηθεί από πριν, το "RepeatTime.txt", στο οποίο καταγράφεται ένας αριθμός. Αυτός δηλώνει τον ρυθμό επανάληψης για τον οποίο λαμβάνονται οι μετρήσεις. Είναι ένας εύκολος τρόπος ώστε ανά πάσα στιγμή να μπορεί να αλλαχθεί χωρίς να χρειαστεί να γίνουν τροποποιήσεις στο πρόγραμμα. Ο αριθμός που υπάρχει μετριέται σε δευτερόλεπτα(secs). Τότε, τυπώνεται στην οθόνη το αντίστοιχο μήνυμα για τον ρυθμό επανάληψης μέσω της εντολής " print(" ")". Μετά από κάθε κλήση του αρχείου αυτού πρέπει να υπάρχει η αντίστοιχη εντολή για το κλείσιμο του, ώστε να αποφευχθεί κάποιο πρόβλημα. Αυτό γίνεται με την εντολή "file.close()".

Έχει δημιουργηθεί μια ρουτίνα για την μετατροπή της ημερομηνίας σε ακέραιο αριθμό προκειμένου να χρησιμοποιηθεί για σύγκριση ημερομηνιών. Αυτή η ρουτίνα ονομάστηκε "date_2_int(xdate)".

Δημιουργήθηκε ένα αρχείο που αποθηκεύει τις μετρήσεις που λαμβάνονται από το Arduino και ονομάστηκε "data_file_name". Για αυτό το αρχείο υπάρχει η αντίστοιχη βιβλιοθήκη που

υπάρχουν κάποιες ρουτίνες για τις ημερομηνίες. Επιλέχθηκε να δίνεται σε κάθε αρχείο ένας συγκεκριμένος τρόπος ονομασίας ώστε να είναι πιο εύκολη η αναζήτηση αργότερα. Δηλώνεται η ακριβής διαδρομή που ακολουθεί το αρχείο για την αποθήκευση του, η οποία είναι : "C:/pythonworks/Meteo_Data/Meteo_". Στο τέλος προσθέτει για κάθε αρχείο την ημερομηνία που λήφθηκαν οι μετρήσεις με την διαδεδομένη μορφή yyyymmdd και το αποθηκεύει σε μορφή κειμένου ".txt". Αντίστοιχα εμφανίζεται το κατάλληλο μήνυμα για αυτή την λειτουργία.

Δημιουργήθηκε έλεγχος για το αν είναι το αρχείο που ζητήθηκε με τις μετρήσεις. Αυτό ελέγχεται ως σωστό αρχείο εάν ξεκινάει με το σύμβολο "#" και τελειώνει με το σύμβολο "$". Δημιουργείται ένας πίνακας στοιχείων όπου περιέχει αυτά τα στοιχεία που ονομάστηκε "packlist[ ]". Το χαρακτηριστικό κάθε πίνακα άλλωστε είναι στο τελείωμα της ονομασίας οι παρενθέσεις [ ]. Κάθε μέτρηση που στέλνεται απο το Arduino ονομάζεται με ένα αναγνωριστικό όνομα και τοποθετείται για ένα συγκεκριμένο στοιχείο μέσα στον πίνακα που δημιουργήθηκε. Αυτό γίνεται για όλα τα στοιχεία.

Επίσης, δημιουργήθηκε ένας πίνακας όπου παίρνει τις τωρινές ενημερώσεις όσον αφορά την ημερομηνία και αποθηκεύει αυτά τα δεδομένα σε μια μεταβλητή. Κάνει ταυτόχρονα μια εκκαθάριση του περιεχομένου που αποθηκεύεται έτσι ώστε να μείνουν μόνο οι μετρήσεις και όχι άχρηστα σύμβολα. Αυτός ο πίνακας ονομάστηκε "pack_to_save". Μόλις εμφανίσει τις μετρήσεις που έχουν ληφθεί εκτυπώνει στην οθόνη την ακριβή ώρα λήψης.

Δημιουργείται ένας πίνακας "perigrafi" που περιέχει τις ονομασίες των μετρήσεων και ένας ακόμα "monades" όπου περιέχει τις μονάδες στις οποίες μετριούνται τα δεδομένα. Γίνεται ένας βρόχος επανάληψης όπου εμφανίζει τα στοιχεία των πινάκων που αναφέρθηκαν χωρίς τα σύμβολα που μπορεί να έχουν μπει μπροστά και πίσω των μετρήσεων καθώς τα λαμβάνει ως string όπως αναφέρθηκε και παραπάνω. Αυτό γίνεται με την εντολή ".decode('utf-8')".

Επιπροσθέτως, έχει δημιουργηθεί ένας βρόχος επανάληψης που ελέγχει την ημέρα. Δηλαδή, αν οι μετρήσεις που λαμβάνονται είναι της ίδιας ημέρας με αυτής που λήφθηκαν οι τελευταίες που υπάρχει αντίστοιχος φάκελος. Εάν είναι διαφορετικής ημερομηνίας, τότε αλλάζει φάκελο αποθήκευσης και τον ονομάζει με την δεδομένη ημερομηνία. Αποθηκεύεται με την μορφή ".txt". Το τελικό αρχείο που αποθηκεύει τα δεδομένα που λαμβάνονται ονομάζεται "data_file_name" και αφού το ανοίξει για να εμφανίσει τα στοιχεία το κλείνει.

Τέλος, εφόσον έχουν γίνει οι παραπάνω λειτουργίες έχει συνταχθεί στο πρόγραμμα τα δεδομένα να ανεβαίνουν σε μια διαδικτυακή βάση. Για να επιτευχθεί αυτό σωστά, στην αρχή του κώδικα δηλώνονται το "API key" της βάσης δεδομένων που είναι επιθυμητό να ανέβουν και το URL, ή αλλιώς το link της σελίδας στην οποία είναι η βάση. Στην συνέχεια δηλώνω τις μεταβλητές με την σειρά που ανεβαίνουν κάνοντας μια ταυτόχρονη εκκαθάριση των στοιχείων που υπάρχουν στα δεδομένα και δεν είναι αριθμοί. Πιο συγκεκριμένα, υπάρχει ένα «b' '» που περιέχεται μέσα ο αριθμός της μέτρησης που λαμβάνεται σε μια σειρά χωρισμένος με κενά. Με την εντολή «.decode('utf-8')» διώχνεται αυτό το «b' '» που υποδηλώνει τον αριθμό. Όλα τα δεδομένα που έχουν οριστεί ως μεταβλητές στέλνονται στους κατάλληλους χώρους για να διαβαστούν από την βάση. Τέλος, απαιτείται ιδιαίτερη προσοχή στο τέλος του κώδικα να υπάρχει η αντίστοιχη εντολή που κλείνει την αποστολή των δεδομένων και την σειριακή επικοινωνία με μια μικρή καθυστέρηση χρόνου ώστε να γίνονται αντιληπτές οι λειτουργίες στο πρόγραμμα.

Γενικά, η κενή εντολή “print()” υπάρχει για λόγους ομοιομορφίας του προγράμματος, τυπώνει δηλαδή μια σειρά κενή.

Ο τελικός κώδικας αναγράφεται στο παράρτημα ΙΙ ολοκληρωμένο.

# 5. Βάσεις Δεδομένων

Γενικά, μια βάση δεδομένων (database) αποτελεί τη βάση για την συλλογή και την οργάνωση πληροφοριών (Microsoft). Αποτελεί, δηλαδή, μία οργανωμένη συλλογή ειδικά ταξινομημένων δεδομένων, σχεδιασμένη με τέτοιο τρόπο, ώστε να μπορεί να εξυπηρετήσει αποτελεσματικά πολλές εφαρμογές, μειώνοντας τις άσκοπες επαναλήψεις των δεδομένων.

Αντί να δημιουργούνται ξεχωριστά αρχεία για κάθε εφαρμογή, όλα τα δεδομένα καταχωρούνται με τέτοια μέθοδο, ώστε να παρουσιάζονται στο χρήστη με έναν ενιαίο τρόπο. Κάθε εφαρμογή μπορεί να κάνει χρήση εκείνων των δεδομένων που είναι απαραίτητα για την επεξεργασία και την παραγωγή των αντίστοιχων πληροφοριών. Με την απλούστερη έννοια, βάση δεδομένων είναι μια συλλογή από εγγραφές και αρχεία, τα οποία είναι οργανωμένα έτσι ώστε να εξυπηρετούν ένα συγκεκριμένο σκοπό (Μαστρογάλια, 2004).

Πιο συγκεκριμένα, κάθε βάση δεδομένων που σχετίζεται με μετεωρολογικούς σταθμούς βρίσκεται στον διακομιστή ή εξυπηρετητή (server) και αποθηκεύει τα στοιχεία ενός σταθμού, τις μετρήσεις που καταγράφει κατά τη διάρκεια του έτους, καθώς και την κατάσταση της εξόδου απομακρυσμένου ελέγχου.

## 5.1. Thingspeak

Γενικά υπάρχουν πολλές διαθέσιμες διαδικτυακές βάσεις δεδομένων. Κάποιες από αυτές είναι phpMyAdmin, MySQL, PostgreSQL, Microsoft SQL Server, Firebird, MongoDB, MariaDB, SQLAlchemy όπως έχει αναφερθεί και σε προηγούμενο κεφάλαιο. Η καθεμία εκ των οποίων έχει συγκεκριμένα χαρακτηριστικά.

Η βάση που επιλέχθηκε για την συγκεκριμένη εργασία είναι η Thingspeak. Είναι μια διαδικτυακή ελεύθερη βάση δεδομένων η οποία είναι συμβατή με το Arduino και το Raspberry Pi. Βασικό χαρακτηριστικό της είναι ότι συλλέγει τα δεδομένα της είτε σε δημόσιο είτε σε ιδιωτικό κανάλι, αναλόγως τη χρήση που ζητείται κάθε φορά. Είναι παγκοσμίως διαδεδομένη βάση και τα διαγράμματα που δημιουργούνται δεν χρειάζονται ιδιαίτερη εργασία από τον χρήστη. Η βάση δεδομένων αυτή λειτουργεί με βάση την διεπαφή IoT(Internet of Things).

Η βάση ThingSpeak επιτρέπει στους αισθητήρες, τα όργανα και τους ιστότοπους να στέλνουν δεδομένα και να αποθηκεύονται είτε σε ιδιωτικό είτε σε δημόσιο κανάλι. Αποθηκεύει τα δεδομένα σε ιδιωτικά κανάλια από προεπιλογή. Υπάρχει και η δυνατότητα αποθήκευσης σε δημόσια κανάλια όπου μπορούν να χρησιμοποιηθούν για την κοινή χρήση δεδομένων με άλλους. Μόλις τα δεδομένα βρίσκονται σε ένα κανάλι ThingSpeak, είναι δυνατό να αναλυθούν και να απεικονισθούν, να υπολογιστούν νέα δεδομένα ή να αλληλεπιδράσουν με τα κοινωνικά μέσα, τις υπηρεσίες ιστού και άλλες συσκευές.

Η διαδικασία εγγραφής είναι πολύ απλή. Δημιουργείται ένας λογαριασμός, και ανοίγει την βάση δεδομένων. Είναι δυνατή η δημιουργία καναλιών, η επεξεργασία των δεδομένων, η εμφάνιση των διαγραμμάτων, ακόμη και η ρύθμιση των διαστημάτων που λαμβάνονται οι μετρήσεις για την εμφάνιση τους στα διαγράμματα. Αυτές είναι κάποιες από τις λειτουργίες της συγκεκριμένης βάσης.

## 5.2. Περιγραφή Διεπαφής

Η διεπαφή για την οποία θα γίνει αναφορά ειναι η IoT(Internet of Things). Το Internet of Things είναι μία έννοια που αφορά τα αντικείμενα και συσκευές που χρησιμοποιούν ενσωματωμένους αισθητήρες για τη συλλογή δεδομένων και την ανάληψη κάποιας δράσης σε αυτά μέσα σε ένα δίκτυο (SAS). Δηλαδή, περιλαμβάνει συσκευές με διαφορετικές δυνατότητες, όπως κάμερες, θερμοστάτες ή ακόμα και ψυγεία. Είναι ένας όρος που περιλαμβάνει οποιαδήποτε ηλεκτρική ή ηλεκτρονική συσκευή συνδέεται στο ίντερνετ χωρίς απαραίτητα να είναι υπολογιστής, κινητό, ή tablet. Πιο απλά, αφορά την σύνδεση ηλεκτρονικών συσκευών μεταξύ τους ή και με το διαδίτκυο.

Η διαδρομή που ακολουθεί η διεπαφή αυτή δεν είναι συγκεκριμένη. Στόχος του όρου IoT είναι να δώσει στους χρήστες όσο τον δυνατό περισσότερο έλεγχο διαφορετικών συσκευών μέσω ίντερνετ και από απόσταση. Οι επιλογές που θα κάνει ο χρήστης μεταφέρονται στο Cloud Server της IoT συσκευής ή της εφαρμογής και επεξεργάζονται από τους κατάλληλους αλγόριθμους που υπάρχουν. Τέλος, τα δεδομένα αποθηκεύονται στα λεγόμενα IoT Gateways, χωρίς να γίνεται κάποια επεξεργασία. Αναλαμβάνουν, δηλαδή, να συνδέσουν την κατάλληλη συσκευή με το σωστό κινητό ή υπολογιστή. Οι πληροφορίες αυτές περνάνε στους ενσωματωμένους αισθητήρες των IoT συσκευών, οι οποίοι και αποκρυπτογραφούν το μήνυμα, έτσι ώστε να εκτελέσουν την εντολή.

## 5.3. Ανάρτηση Δεδομένων

Εφόσον έχουν ληφθεί τα δεδομένα από την κατασκευή και έχουν εμφανιστεί στην οθόνη που είναι συνδεδεμένη, είναι επιθυμητό να ανεβαίνουν σε μια διαδικτυακή βάση δεδομένων. Αυτό συμβαίνει διότι μπορεί η κατασκευή να τοποθετηθεί σε έναν απομακρυσμένο χώρο και να είναι ορατές οι μετρήσεις σε οποιοδήποτε υπολογιστή, εφόσον το κανάλι είναι δημόσιο. Ειδάλλως, θα πρέπει ο χρήστης να έχει κωδικούς πρόσβασης για το ιδιωτικό κανάλι. Οι μετρήσεις που λήφθηκαν βγαίνουν με την μορφή διαγράμματος που έχει επιλεγεί. Μπορεί να ρυθμιστεί ο ρυθμός που δείχνει το διάγραμμα τα δεδομένα. Επιπλέον, μπορεί ο διαχειριστής της βάσης να επεξεργαστεί ή ακόμα και να καθαρίσει τα δεδομένα που ήδη υπάρχουν καταχωρημένα.

Αφού έχει δημιουργηθεί ο λογαριασμός για την βάση, δημιουργείται ένα κανάλι που ονομάστηκε “show meteo data” όπου αναρτούνται οι μετρήσεις. Σε αυτό το κανάλι δημιουργήθηκαν οχτώ διαγράμματα, ένα για κάθε στοιχείο που λαμβάνει μετρήσεις. Κάθε διάγραμμα έχει ονομαστεί κατάλληλα ώστε να γίνεται εύκολα αντιληπτή η μέτρηση την οποία αφορά. Όλες οι μετρήσεις που βγάζει απεικονίζονται σε σχέση με τον χρόνο.

Κάθε κανάλι έχει κάποια συγκεκριμένα χαρακτηριστικά. Πολύ σημαντικό είναι να εντοπίσει κανείς τους κωδικούς για τα κλειδιά που δίνονται για κάθε κανάλι. Εφόσον αυτό είναι γνωστό, μπορεί να συνταχθεί ο κατάλληλος κώδικας για την εμφάνιση των δεδομένων στα διαγράμματα. Πρέπει στην αρχή του κώδικα να δηλωθεί το “API key” το οποίο είναι μονδαικό για κάθε κανάλι. Ο σκοπός αυτού είναι για να σταλθούν τα δεδομένα στο συγκεκριμένο κανάλι που είναι επιθυμητό. Δηλώνεται με την εντολή “ myAPI = 'FADFVCH1VSILNF44' ” και αφορά το κανάλι στο οποίο θα εμφανιστούν τα δεδομένα.

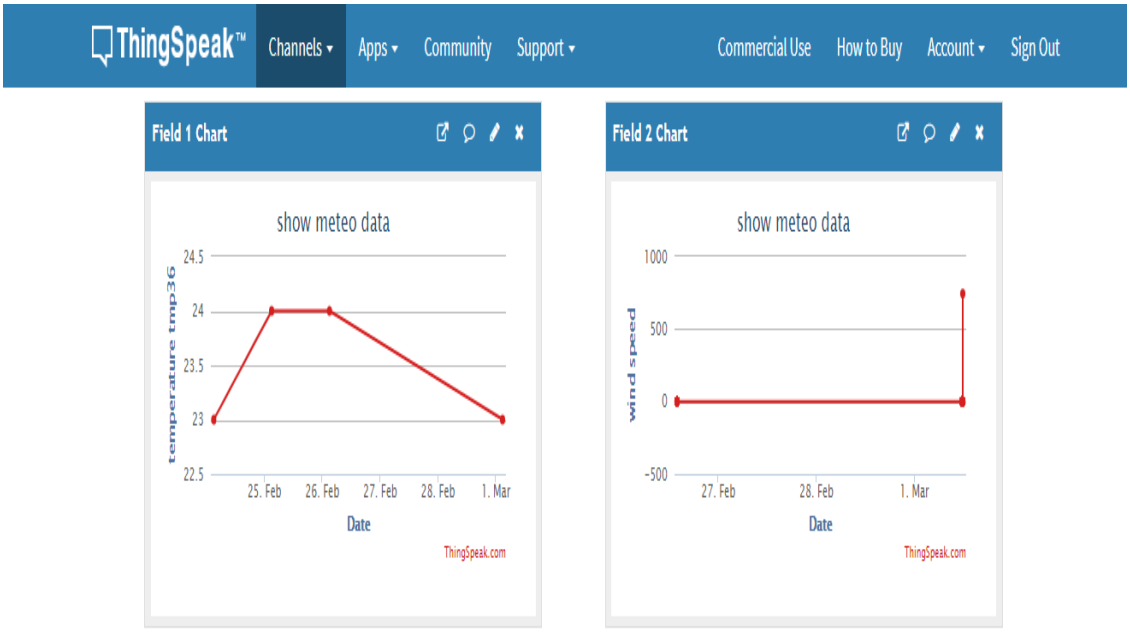
Αμέσως μετά, πρέπει να δηλωθεί το URL που αφορά την βάση. Αυτό γίνεται με την εντολή “baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI ”. Αυτή η διαδρομή είναι που ακολουθούν τα δεδομένα για να σταλθούν και δεν αλλάζει. Στην συνέχεια του κώδικα δηλώνονται οι μεταβλητές με “.decode(‘utf-8’)” για να μην υπάρχουν σύμβολα που συγχέουν τις μετρήσεις. Έπειτα, στέλνονται οι μετρήσεις στα σωστά πεδία που είναι επιθυμητό να εμφανιστούν. Αφού διαβαστούν αυτά από τη βάση είναι σημαντικό να κλείσει η επικοινωνία. Για να υλοποιηθούν τα παραπάνω όσον αφορά την συγκεκριμένη κατασκευή και βάση πρέπει να προστεθούν στον κώδικα οι παρακάτω εντολές.

conn=urllib.request.urlopen(baseURL+'&field1=%s&field2=%s&field3=%s&field4=%s&field5=%s&field6=%s&field7=%s&field8=%s'%(xtemp36,xfanValue,xorient,xhumidity,xtemperature,xbmptemp,xalti,xpresf))

conn.read()

conn.close()

Τα διαγράμματα που εμφανίζονται τελικά, φαίνονται στις παρακάτω Εικόνες.




**Εικόνα 31: Διαγράμματα θερμοκρασίας TMP36 και ταχύτητας ανέμου**


**Εικόνα 32: Διαγράμματα προσανατολισμού ανέμου και υγρασίας HIH6120**


**Εικόνα 33: Διαγράμματα θερμοκρασίας HIH6120 και BMP180**


**Εικόνα 34: Διαγράμματα υψόμετρου και ατμοσφαιρικής πίεσης BMP180**

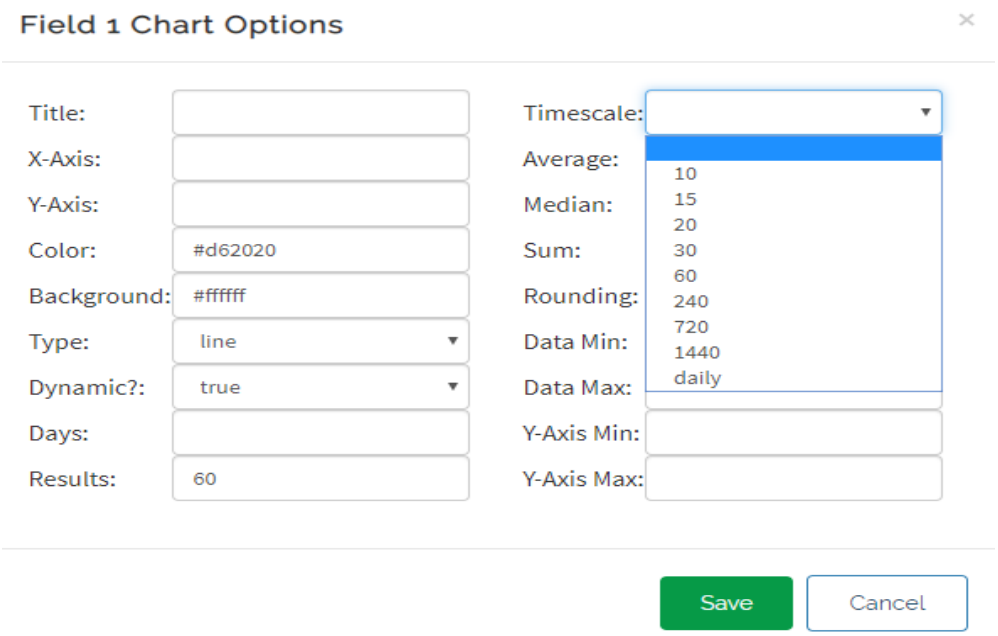
# 6. Λειτουργία Διάταξης

Το παρόν κεφάλαιο σχετίζεται με τη λειτουργία διάταξης, δηλαδή την αναλυτική περιγραφή της εργασίας. Αναφέρεται τι ακριβώς έχει υλοποιηθεί και πως αυτό λειτουργεί.

Αφού υλοποιηθεί η κατασκευή άρτια, δηλαδή σχεδιαστεί η πλακέτα κατάλληλα μέσω του προγράμματος Altium, τυπωθεί, υλοποιηθεί και συγκοληθεί με τα κατάλληλα εξαρτήματα, πρέπει να προγραμματιστεί. Γράφεται ο αντίστοιχος κώδικας στο Arduino IDE ώστε να προγραμματιστεί σωστά η κατασκευή. Ο κώδικας του Arduino παραθέτεται ολόκληρος στο παράρτημα 1. Άπαξ και αποθηκευτεί μία φορά στην μνήμη του Arduino, κάθε φορά που θα συνδέεται σε ρεύμα τρέχει τον κώδικα που έτρεξε και αποθήκευσε τελευταία φορά. Αυτό το γεγονός διευκολύνει την μετέπειτα ανεξάρτητη τοποθέτησή του. Με τον κατάλληλο προγραμματισμό του εμφανίζονται στην σειριακή οθόνη του Arduino IDE τα αποτελέσματα των μετρήσεων και μπορούν με τα πατήματα των κουμπιών να εμφανιστούν οι εναλλαγές της οθόνης. Αφού συνταχθεί αυτός ο κώδικας, τα αποτελέσματα είναι φανερά στις οθόνες και ταυτοποιούνται ότι είναι αληθής η εργασία προχωράει στο επόμενο βήμα. Γίνεται η αντίστοιχη σύνδεση της ήδη υπάρχουσας κατασκευής με το Raspberry Pi το οποίο φυσικά πρέπει να προγραμματιστεί αντίστοιχα με την Python. Αυτός ο κώδικας παρατίθεται στο παράρτημα 2. Εκεί δημιουργήθηκαν οι κατάλληλοι πίνακες ώστε να καταλαμβάνουν θέσεις για τα στοιχεία των μετρήσεων. Δηλώθηκε η σειριακή επικοινωνία του Arduino με το Raspberry Pi. Ταυτόχρονα, δηλώθηκε και η σύνδεση με την βάση για το ανέβασμα των δεδομένων στα διαγράμματα. Με τις κατάλληλες εντολές δηλώθηκε το URL της βάσης που ανεβαίνουν τα αποτελέσματα. Κάπως έτσι τελειώνει η εργασία αυτή αφού έχουν αναρτηθεί και τα αποτελέσματα. Είναι δηλαδή, ένας πλήρης μετεωρολογικός σταθμός όπου μπορεί να τοποθετηθεί σε όποιο σημείο ζητηθεί αρκεί να συνδεθεί με ρεύμα. Κατ’ επέκταση τα αποτελέσματα απλά θα εμφανίζονται είτε στην οθόνη, είτε στην διαδικτυακή βάση.

# 7. Αποτελέσματα

Ως αποτέλεσμα αυτής της διεργασίας δημιουργήθηκε ένα ολοκληρωμένο μετεωρολογικό σύστημα. Το σύστημα αυτό καταγράφει, εμφανίζει και ανεβάζει σε διαγράμματα στο ίντερνετ όλα τα δεδομένα που λαμβάνει. Ο ρυθμός λήψης των μετρήσεων μπορεί να αλλάξει εφόσον αυτό είναι επιθυμητό. Ανεξάρτητα από αυτό μπορεί να αλλάξει η εμφάνιση των μετρήσεων που απεικονίζονται στα διαγράμματα. Αυτό μπορεί να γίνει από την γραμμή εργαλείων κάθε διαγράμματος που υπάρχει το σύμβολο του μολυβιού και ονομάζεται “edit”. Αν πατηθεί αυτό, εμφανίζεται στην οθόνη ένα παράθυρο από το οποίο μπορούν να ρυθμιστούν κάποια χαρακτηριστικά του εκάστοτε διαγράμματος. Εφόσον είναι επιθυμητό να αλλαχθεί ο ρυθμός που εμφανίζονται τα δεδομένα επιλέγεται το πλαίσιο “Timescale” που δίνονται κάποιες προεπιλογές. Παρόλα αυτά μπορεί να καταγραφεί ο ρυθμός που επιθυμείται. Πιο αναλυτικά φαίνεται και στην Εικόνα 35.

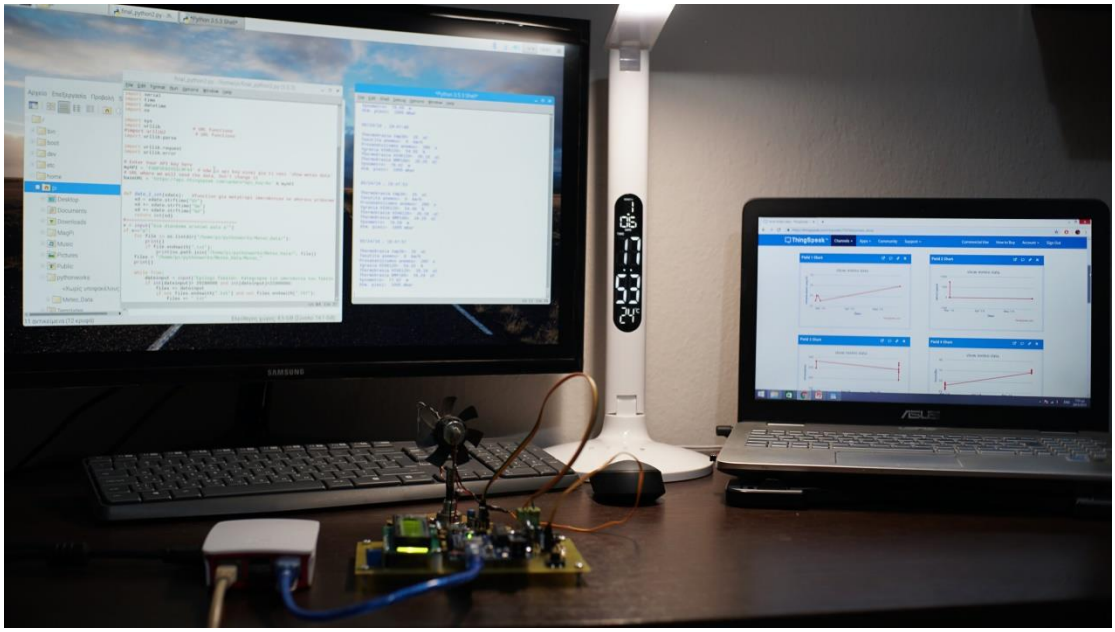



Εικόνα 35: Ρυθμός εμφάνισης των δεδομένων

# 8. Συμπεράσματα

Για την διεκπαιρέωση της εργασίας αυτής αντιμετωπίστηκαν κάποια προβλήματα. Για αρχή έπρεπε να επιλεχθούν μέσα από μια μεγάλη ποικίλια αισθητήρων οι συγκεκριμένοι. Μέσα από αντίστοιχη έρευνα αγοράς μέσω του διαδικτύου κατέληξε να γίνει επιλογή των συγκεκριμένων. Παράλληλα έπρεπε να ελεγθεί ο σωστός προγραμματισμός τους μέσω αντίστοιχων πήγων. Σημαντικό ρόλο έπαιξε η σωστή διαχείριση των πληροφοριών του διαδικτύου. Πιο συγκεκριμένα, τα προβλήματα που αφορούν τον προγραμματισμό των αισθητήρων επιλύθηκαν κυρίως με την σχετική αναζήτηση στον ιστότοπο "https://stackoverflow.com ". Κατά την εξέλιξη της πτυχιακής αντιμετωπίστηκε επίσης πρόβλημα με το Raspberry Pi. Με την κατάλληλη ενημέρωση που έγινε από το κατάστημα αγοράς έγινε αντιληπτό ότι λόγω του τροφοδοτικού μπορούν να απορυθμιστούν τα εγκατεστημένα αρχεία στην μνήμη. Το Raspberry Pi χρειάζεται μεγαλύτερη ένταση από αυτή που δίνει ένα κοινό τροφοδοτικό μικρότερο των 2A. Αυτό είχε ως αποτέλεσμα η οθόνη να παραμένει μαύρη κατά την διάρκεια της τροφοδοσίας της πλακέτας. Η ενέργεια που επίλθε για την επίλυση του προβλήματος αυτού ήταν η διαγραφή του περιεχομένου μνήμης και η εγκατάσταση του από την αρχή. Έτσι λειτούργησε ξανά.

Το συμπέρασμα που μπορεί να βγεί από όλη αυτή την εργασία είναι ότι στο τέλος της κατασκευής δημιουργήθηκε ένας ολοκληρωμένος μετεωρολογικός σταθμός που μπορεί να φανεί χρήσιμος σε διάφορες ειδικότητες. Φυσικά η κατασκευή αυτή προαπαιτεί τις αντίστοιχες γνώσεις για την σωστή υλοποίηση. Καθώς όμως, η κοινότητα του διαδικτύου είναι τεράστια, ο κάθε ενδιαφερόμενος μπορεί να ενημερωθεί από πηγές που υπάρχουν και να προβεί σε διαδικασίες κατασκευής και δημιουργίας κάποιου αντίστοιχου πρότζεκτ. Αξιοσημείωτο είναι επίσης ότι οι πλακέτες που χρησιμοποιήθηκαν έχουν αντίστοιχα φόρουμ – πλατφόρμες όπου χρήστες των αντικειμένων καταθέτουν τις εμπειρίες τους και τα προβλήματα που αντιμετώπισαν με τις λύσεις τους.



Εικόνα 36: Ολοκληρωμένη σύνδεση του μετεωρολογικού σταθμού

# 9. Βιβλιογραφία


auth. (n.d.). *"Climatology"*. Ανάκτηση από www.geo.auth.gr/431/th/Climatology.pdf

Bosch. (2013). *BMP180 datasheet.* Ανάκτηση από https://cdn-shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf

Honeywell. (2012, July ). *HIH6120 datasheet.* Ανάκτηση από https://www.mouser.com/catalog/specsheets/HumidIconHIH6130.pdf

Microsoft. (n.d.). *Microsoft support*. Ανάκτηση από https://support.office.com/el-gr/article/%CE%92%CE%B1%CF%83%CE%B9%CE%BA%CE%AD%CF%82-%CF%80%CE%BB%CE%B7%CF%81%CE%BF%CF%86%CE%BF%CF%81%CE%AF%CE%B5%CF%82-%CE%B3%CE%B9%CE%B1-%CF%84%CE%B9%CF%82-%CE%B2%CE%AC%CF%83%CE%B5%CE%B9%CF%82-%CE%B4%CE%B5%CE%B4%CE%BF%CE%B

SAS. (n.d.). Ανάκτηση από https://www.sas.com/el_gr/insights/big-data/internet-of-things.html

*TMP36 datasheet.* (n.d.). Ανάκτηση από https://www.analog.com/media/en/technical-documentation/data-sheets/tmp35_36_37.pdf

Γρηγορόπουλος, Ν. (2017). *Ανάπτυξη φορητού μετεωρολογικού σταθμού με Arduino και αποστολή των μετρήσεων σε συσκευή Android.* Ανάκτηση από http://okeanis.lib.puas.gr/xmlui/handle/123456789/3947?show=full

Λάτμος, Ν. (2017). *Σχεδιασμός και ανάπτυξη μετεωρολογικού σταθμού με ενσωματωμένα αισθητήρια και επικοινωνίες WiFi και GPRS για τη μέτρηση παραμέτρων μικρο-κλίματος σε πραγματικό χρόνο.* Ανάκτηση από https://ikee.lib.auth.gr/record/294384/files/Nikolaos%20Latmos.pdf

Μαστρογάλια, Π. (2004). *Διαχείρηση και διασύνδεση σχεσιακών βάσεων δεδομένων.* Ανάκτηση από http://apothetirio.teiep.gr/xmlui/bitstream/handle/123456789/998/fin_20040053.pdf?sequence=1

Παπάζογλου, Π. Μ. ( 2014). *Ανάπτυξη εφαρμογών με το Arduino.* Τζιόλας.

Πουλάκης, Ε. (2015). *Προγραμματίζοντας με τον μικροελεγκτή Arduino.* Ανάκτηση από http://users.sch.gr/manpoul/docs/arduino/ProgrammingArduino.pdf

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ, Τ. Δ. (n.d.). *Τελεστικοί Ενισχυτές.* Ανάκτηση από https://eclass.pat.teiwest.gr/eclass/modules/document/file.php/487188/%CE%A3%CE%B7%CE%BC%CE%B5%CE%B9%CF%8E%CF%83%CE%B5%CE%B9%CF%82%20%CE%BC%CE%B1%CE%B8%CE%AE%CE%BC%CE%B1%CF%84%CE%BF%CF%82/%CE%9A%CE%B5%CF%86%CE%B1%CE%BB%CE%B1%CE%B9%CE%BF%207%20-%20%CE%A4%C

Φραγκιαδάκης, Ν. (2006). *Αισθητήρες - Μετατροπείς.* Ανάκτηση από http://nefeli.lib.teicrete.gr/browse/sefe/hlk/2006/KritsotakisIoannis/attached-document/2006Kritsotakis.pdf

# Παράρτημα Ι

- Κώδικας για το Arduino:

Λειτουργία του προγράμματος:

- Δηλώνονται οι βιβλιοθήκες,
- Δηλώνονται οι μεταβλητές που θα χρησιμοποιηθούν,
- Δημιουργούνται συναρτήσεις για κάθε μεταβλητή, όπου περιλαμβάνει αντίστοιχες πράξεις για την εκάστοτε μέτρηση,
- Δημιουργείται ένα πακέτο που περιέχει όλες τις τιμές των μετρήσεων,
- Δημιουργείται μια ρουτίνα επανάληψης για την εμφάνιση των δεδομένων της οθόνης LCD με το πάτημα του κουμπιού,
- Ενεργοποιείται η σειριακή επικοινωνία,
- Δημιουργείται μια συνάρτηση για την επιθυμητή εμφάνιση στην σειριακή οθόνη του υπολογιστή.

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

#include<Wire.h>    //i2C επικοινωνια
#include"HIH6130.h"

#include <Adafruit_BMP085.h> // bmp180 library

#define i2c_address    0x27
HIH6130       hih_normal;

Adafruit_BMP085 bmp;  // onomazw tin library-> bmp
//================================================
//general variables
int temp36;   //thermokrasia analogikou aisthitira TEMP36
int fanValue;   //taxituta anemou
unsigned int orient;   //prosanatolismos-dieuthunsi anemou(pontesiometro)
float humidity;  // ugrasia HIH6120
float temperature; //thermokrasia HIH6120
float bmptemp; // thermokrasia BMP180
```

```
float alti; // upsometro
int presf; //atmosfairiki piesi BMP180
int buttonState; //dilwnw to koumpi
int othoni; // dilwnw poia leitourgia tou koumpiou
//================================================
//functions(routines)
//.....

int AnalogTemp() {
  //διαβαζω τον sensor TMP36
 int sensorValue = analogRead(A0); // διαβαζω την αναλογικη εξοδου του TMP36
 double x = (double)sensorValue;
 //μετατροπη σε mVolt
 x *= 5000;
 x /= 1023;
 //.......
 //μετατροπη απο mVolt σε βαθμους Κελσιου (10mV/βαθμο Κελσιου)
 x /= 30;
 //......
  return (int)x;
}
//.....................................................
int WindSpeed(){
//διαβαζω την ταχυτητα του (fan)-ανεμιστηρας
 return analogRead(A1); // διαβαζω την ταση του fan οταν περιστρεφεται απο τον ανεμο(γινεται γεννητρια)

}//.....................................
int Orientation(){
 double ox = (double)analogRead(A2);
 ox = analogRead(A2);
 ox *= 359;
 ox /= 1023;
```

```
 return (int)ox;   // diavazw thn tasi tou pontesiometrou apo ton fan pou dinei ton prosanatolismo tou anemou
}
//.....
void HIH_Values(){
  hih_normal.begin(i2c_address);
  hih_normal.fetch_data(&humidity, &temperature);
}
//.....

float BMP180_Temp(){
   return bmp.readTemperature();
}
//.....
float BMP180_Altitude(){
 return bmp.readAltitude(101500);
}
//.....
int BMP180_Pressure(){
  long pres = bmp.readPressure(); //i metrisi pou pairnei einai se pascal
  // pres /=100; //metatropi se millibar( i hectopascal)
  return (int)(pres/100);   //casting se int apo float (logw diairesis)
}
//.....
String Create_Pack(){
  //dimiourgw to diko mou protokollo epikoinwnias my packet, gia na steilw to paketo twn stoixeiwn sti Python, xrisimopoiw initiator # kai terminator $ gia na katalavei
  //pou ksekinaei kai pou teleiwnei to paketo, tin (space) ti vazw gia seperator gia ti split sti python
 String mypack = "# ";
 mypack += String(temp36);
 mypack += " ";
 mypack += String(fanValue);
 mypack += " ";
 mypack += String(orient);
```

```
  mypack += " ";
  mypack += String(humidity);
  mypack += " ";
  mypack += String(temperature);
  mypack += " ";
  mypack += String(bmptemp);
  mypack += " ";
  mypack += String(alti);
  mypack += " ";
  mypack += String(presf);
  mypack += " $";
  return mypack;
}
//.....
void Show_Results_LCD(unsigned char a){
  if(a==0){
   lcd.clear();
   lcd.print("Eleni Tsanidou");
   lcd.setCursor(0, 1);
   lcd.print("PTUXIAKI ERGASIA");

  }else if(a==1){
   //αποτυπωση αποτελεσματων στο lcd
  lcd.clear();
  lcd.print("T36:");
  lcd.setCursor(4, 0);
  lcd.print(temp36,1); // ( ,1) simainei akriveia arithmou me ena dekadiko
  lcd.print("oC W:");
  lcd.print(fanValue, 1);
  lcd.setCursor(0,1);
  lcd.print("H:");
  lcd.print(String(humidity));
  lcd.print("%,T:");
  lcd.print(String(temperature));
```

```
  lcd.print("oC");

 }else if(a==2){
  lcd.clear();
  lcd.print("BMP TEMP: ");
  lcd.print(bmptemp, 1);
  lcd.setCursor(0, 1);
  lcd.print("Alt:");
  lcd.print(alti, 1);
  lcd.print(",P:");
  lcd.print(presf, 1);

}else if(a==3){
  lcd.clear();
  lcd.print("Sending results");
  lcd.setCursor(0, 1);
  lcd.print("to Python");

}else{
  }

}
//.....................................
void Measure_All(){
  //read sensors
    temp36 = AnalogTemp();
    fanValue = WindSpeed();
    orient = Orientation();
    HIH_Values();
    bmptemp = BMP180_Temp();
    alti = BMP180_Altitude();
    presf = BMP180_Pressure();
}
```

```
//================================================

void setup() {
  // put your setup code here, to run once:
  pinMode(8, INPUT);
  lcd.begin(16, 2);
  Show_Results_LCD(0);
  delay(3000);

  Serial.begin(9600);
  digitalWrite(13, LOW);  //san alarm
  Wire.begin();  //εντολη μια φορα, εισαγει την i2C επικοινωνια

   // gia to bmp180

  if (!bmp.begin())
  {
  lcd.clear();
  lcd.print("BMP180 sensor not found");
  while (1) {}   // while(1>0) panta isxuei
  }
 othoni = 0;
}

void loop() {
  // put your main code here, to run repeatedly:

if (Serial.available() > 0) {
  unsigned char rec_byte=0;
  rec_byte = Serial.read();
  if(rec_byte=='A'){  //i entoli A zitaei na stalei to paketo olwn twn metrisewn
    Measure_All();
    Serial.println(Create_Pack());
    Show_Results_LCD(3);
```

```
 }else if(rec_byte=='B'){
   Show_Results_LCD(0);

 }else if(rec_byte=='C'){
   Show_Results_LCD(1);

 }else if(rec_byte=='D'){
   Show_Results_LCD(2);

 }else{
   int dummy=0; //axristi entoli gia na min emfanizei stin othoni
 }
}

//diavasma button gia allagi othonis
buttonState = digitalRead(8);
 if(buttonState==HIGH){     // deixnei tis teleutaies metriseis pou exw parei
    Measure_All();
   Show_Results_LCD(othoni);
   if(othoni<4){
     othoni +=1;
   }else{
     othoni = 0;
   }

  while(digitalRead(8)==HIGH){}   //perimenei na eleutherwsw to koumpi
   delay(250);                // debounce diadikasia
 }



}
```

## Παράρτημα ΙΙ

- Κώδικας για Raspberry Pi:

Λειτουργία του προγράμματος:

- Εισαγωγή των κατάλληλων βιβλιοθηκών,
- Δήλωση κάποιων στοιχείων για την διαδικτυακή βάση που ανεβαίνουν τα δεδομένα των μετρήσεων,
- Δημιουργία συνάρτησης για την μετατρο΄πή της ημερομηνίας σε ακέραιο,
- Δημιουργία βρόγχου επανάληψης για να ελέγξει ποιο πλήκτρο πατήθηκε και αντίστοιχα να γίνουν οι κατάλληλες ενέργειες που προβλέπονται,
- Δήλωση όλων των δεδομένων σε μορφή πίνακα,
- Δημιουργία αρχείου με περιεχόμενο τις μετρήσεις και συγκεκριμένο όνομα αρχείου που ορίζεται μέσα στον κώδικα,
- Έλεγχος για την ορθή σύνδεση του μετεωρολογικού συστήματος,
- Έλεγχος για την αλλαγή ημέρας, άρα και φακέλου, καθώς και για την μορφή που αποθηκέυεται το αρχείο,
- Αποστολή των δεδομένων που λήφθηκαν από τις μετρήσεις στην διαδικτυακή βάση.

```
import serial
import time
import datetime
import os

import sys
import urllib          # URL functions
#import urllib2        # URL functions
import urllib.parse

import urllib.request
import urllib.error

# Enter Your API key here
myAPI = 'FADFVCH1VSILNF44' # edw to api key einai gia ti vasi 'show meteo data'
# URL where we will send the data, Don't change it
```

```
baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI



def date_2_int(xdate):    #function gia metatropi imerominias se akeraio prokeimenou na xrisimopoihthei se sugkrisi imerominiwn
   xd = xdate.strftime("%Y")
   xd += xdate.strftime("%m")
   xd += xdate.strftime("%d")
   return int(xd)
#=======================================
x = input("Gia diavasma arxeiwn pata a!")
if x=="a":
   for file in os.listdir("/home/pi/pythonworks/Meteo_Data/"):
      print()
      if file.endswith(".txt"):
         print(os.path.join("/home/pi /pythonworks/Meteo_Data/", file))
   filex = "/home/pi/pythonworks/Meteo_Data/Meteo_"
   print()


   while True:
      dateinput = input("Epilogi fakelou: Kategrapse tin imerominia tou fakelou pou thes se morfi yyyymmdd  ")
      if int(dateinput)> 20180000 and int(dateinput)<21000000:
         filex += dateinput
         if not filex.endswith(".txt") and not filex.endswith(".TXT"):
            filex += ".txt"
         try:
            file = open(filex, "r")
            print()
            for line in file:
               print(line)
            break
         except :
            print("To arxeio pou epelekses den uparxei!")
```

```
            filex = "/home/pi/pythonworks/Meteo_Data/Meteo_"

  #teliki pagida gia na dwseis ti dunatotita na deis ta stoixeia tou arxeiou
      try:
         while True:
            dummy = 0
      except KeyboardInterrupt:   #dunatotita termatismou programmatos me ctrl +C
          exit()




print()
print()
#leitourgia meteorologikou stathmou
perigrafi = ["Thermokrasia tmp36= ", "Taxutita anemou= ", "Prosanatolismos anemou= ", "Ygrasia HIH6120= ", "Thermokrasia HIH6120= ", "Thermokrasia BMP180= ", "Ypsometro= ", "Atm. piesi= "]
monades =[" oC", " km/h"," o" , " %", " oC", " oC", " m", "mbar"]

file = open("/home/pi/pythonworks/RepeatTime.txt", "r")
freq = int(file.read())   #diavazw tin epithumiti suxnotita epanalipsis metrisewn
print ("Ruthmos epanalipsis: ", str(freq)," secs")
file.close()

atim = datetime.datetime.now()      #to arxeio pou dimiourgw tha exei kai tin mera(oti tha dilwthei ws append gia na prosthetei ola ta stoixeia tin idia mera)
dtim = "/home/pi/pythonworks/Meteo_Data/Meteo_"
dtim += atim.strftime("%Y")
dtim += atim.strftime("%m")
dtim += atim.strftime("%d")
dtim += ".txt"
data_file_name = dtim
print("Arxeio apothikeusis dedomenwn: ",data_file_name)
```

```
try:

   ser = serial.Serial('/dev/ttyACM0', 9600, timeout=5)
except :
   print("Provlima sti sundesi me to Arduino")
   time.sleep(3)
   exit()

print("Active serial port: ",ser.name)       # check which port was really used

while True:
   ser.write(b'A')   #b-dilwnei oti tha einai byte, kai na to diaxeiristei san ASCII kwdika,oxi " " eisagwgika giati to pairnei san string
   s = ser.readline()
   #if diapistwnei oti einai to mypack, oti ksekinaei me # kai teleiwnei me $, kai katharizei mpros kai pisw ta stoixeia pou de thelw na fainontai
   if s.startswith(b'#'):
      f=s.find(b'$')
      s=s[2:f-1]
      packlist = s.split()

      temp36=packlist[0]


      fanValue=packlist[1]


      orient=packlist[2]


      humidity=packlist[3]
```

```
        temperature=packlist[4]


        bmptemp=packlist[5]


        alti=packlist[6]


        presf=packlist[7]









        tim=datetime.datetime.now()
        stim=tim.strftime("%Y")
        stim += tim.strftime("%m")
        stim += tim.strftime("%d")
        stim += tim.strftime("%H")
        stim += tim.strftime("%M")
        stim += tim.strftime("%S")
        stim +=" "
        pack_to_save = stim + str(s.decode('utf-8'))   # to paketo pou thelw na swsw se arxeio
        pack_to_save += "\n"
        # to minima stin othoni pou vgazei
        #os.system('cls')        #   os.system('clear')   on linux / os x  ,!!! isxuei mono gia ti mauri
othoni tis python
        print()
        print(" ", tim.strftime("%x ,"),tim.strftime("%X"))
        print()
        for i in range(0,8):
            print(" ", perigrafi[i],packlist[i].decode('utf-8'),monades[i])
```

```
print()
#elegxos allagis imeras + fakelou
if date_2_int(tim)>date_2_int(atim):
   newtim = "/home/pi/pythonworks/Meteo_Data/Meteo_"
   newtim += tim.strftime("%Y")
   newtim += tim.strftime("%m")
   newtim += tim.strftime("%d")
   newtim += ".txt"
   data_file_name = newtim

#swsimo paketou se fakelo
file= open(data_file_name, "a")

file.write(pack_to_save)

file.close()

# Sending the data to thingspeak
#dilwnw tis metavlites me .decode gia na fugei to b' '
xtemp36 = temp36.decode('utf-8')
xfanValue = fanValue.decode('utf-8')
xorient = orient.decode('utf-8')
xhumidity = humidity.decode('utf-8')
xtemperature = temperature.decode('utf-8')
xbmptemp = bmptemp.decode('utf-8')
xalti = alti.decode('utf-8')
xpresf = presf.decode('utf-8')
conn = urllib.request.urlopen(baseURL + '&field1=%s&field2=%s&field3=%s&field4=%s&field5=%s&field6=%s&field7=%s&field8=%s' % (xtemp36,xfanValue,xorient,xhumidity,xtemperature,xbmptemp,xalti,xpresf))
#prit
conn.read()
# Closing the connection
conn.close()
```

```
    time.sleep(freq)


ser.close()          # close port
```

## Παράρτημα ΙΙΙ

- Συντομογραφίες

PWD – Pulse Width Modulation : Διαμόρφωση εύρους παλμού

IDE – Intergrated Development Enviroment : Ολοκληρωμένο περιβάλλον ανάπτυξης

GPIO – General Purpose Input/Output : Είσοδος/έξοδος γενικού σκοπού

IDLE – Intergrated Development and Learning Enviroment : Ολοκληρωμένο περιβάλλον ανάπτυξης και μάθησης

Cmd – Command promt : Γραμμή εντολών

Browser : πρόγραμμα περιήγησης

PDAs – Personal Digital Assistant : Προσωπικός ψηφιακός βοηθός

GPS – Global Positioning System : Παγκόσμιο σύστημα εντοπισμού οχημάτων

I2C – Inter-Intergrated Circuit (protocol – Two Wire) : Πρωτόκολλο διασύνδεσης προηγμένων βαθμίδων

hPa – hectoPascal

SCL – serial clock : παλμός στην I2C επικοινωνία

SDA – serial data : δεδομένα στην I2C επικοινωνία

Pointers : δείκτες

LCD – Liquid Crystal Display : οθόνη υγρών κρυστάλλων

Button : Κουμπί

Cols – columns : στήλες

Rows : γραμμές

Ctrl – control : πλήκτρο στο πληκτρολόγιο του υπολογιστή

Secs : δευτερόλεπτα

Database : βάση δεδομένων

Server : εξυπηρετητής, διακομιστής

IOT – Internet Of Things : Διαδίκτυο των πραγμάτων

CPU – Central Prosseccing Unit : Κεντρική Μονάδα Επεξεργασίας

URL – Uniform Resource Locator : Ενιαίος Εντοπιστής Πόρων