Ακαδημια Εμπορικου Ναυτικου Μακεδονιας

Σχολη Πλοιαρχων

Διαδικτυακός Μετεωρολογικός Σταθμός

Πτυχιακή Εργασία του:

Ιωάννη Κων. Χρήστου

Α.Γ.Μ.:3186

Επίβλεψη: Στεφανία Λάμπουρα

Νέα Μηχανιώνα, Ιούνιος 2015

ΑΚΑΔΗΜΙΑ ΕΜΠΟΡΙΚΟΥ ΝΑΥΤΙΚΟΥ

Α.Ε.Ν ΜΑΚΕΔΟΝΙΑΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: Στεφανία Λάμπουρα

***ΘΕΜΑ*: Διαδικτυακός Μετεωρολογικός Σταθμός**

ΤΟΥ ΣΠΟΥΔΑΣΤΗ: Ιωάννη Κων. Χρήστου

Α.Γ.Μ: 3186

Ημερομηνία ανάληψης της εργασίας: 20/05/2014

Ημερομηνία παράδοσης της εργασίας: 02/06/2015

Α/Α	*Ονοματεπώνυμο*	*Ειδικότης*	*Αξιολόγηση*	*Υπογραφή*
1	*ΤΣΟΥΛΗΣ ΝΙΚΟΛΑΟΣ*	*ΠΛΟΙΑΡΧΟΣ Α`*		
2				
3				
ΤΕΛΙΚΗ ΑΞΙΟΛΟΓΗΣΗ				

Ο ΔΙΕΥΘΥΝΤΗΣ ΣΧΟΛΗΣ : ΤΣΟΥΛΗΣ ΝΙΚΟΛΑΟΣ

Π ε ρ ι ε χ ό μ ε ν α

Περίληψη 1
1 Κεφάλαιο 1ο – Εισαγωγή 3
2 Κεφάλαιο 2ο – Αντικείμενο Εργασίας 6
2.1 Αυτόνομος Μετεωρολογικός Σταθμός 7
2.2 Μετεωρολογικά Φαινόμενα 7
2.2.1 Ατμοσφαιρική Πίεση 8
2.2.2 Υγρασία 8
2.2.3 Άνεμος 9
2.2.4 Θερμοκρασία 9
2.2.5 Σημείο Δρόσου 10
3. Κεφάλαιο 3ο – Πλατφόρμα Arduino 12
3.1 Γενικά Στοιχεία 12
3.2 Μοντέλα μικροελεγκτών Arduino 13
3.2.1 Το υλικό (Arduino Mega 2560) 14
3.2.2 Το λογισμικό λειτουργίας του Arduino 16
3.3 Αισθητήρες καταγραφής 17
3.3.1 Αισθητήρας θερμοκρασίας / υγρασίας DHT - 22/ AM - 2302 17
3.3.2 Αισθητήρας Ατμοσφαιρικής / Βαρομετρικής Πίεσης BMP - 180 19
3.3.3 Αισθητήρας Κατεύθυνσης Ανέμου 80422 21
3.3.4 Ψηφιακός Μετρητής Φωτός BH - 1750FVI 22
3.4 Περιφερειακές Συσκευές 22
3.4.1 Ρολόι Χρονισμού RTC DS - 3231 23
3.4.2 Κάρτα Επέκτασης Δικτύου (Arduino Ethernet Shield) W5100 24
4. Κεφάλαιο 4ο – Προγραμματισμός Arduino 26
4.1 Ολοκληρωμένο Περιβάλλον Ανάπτυξης του Arduino 26
4.2 Σειριακή Οθόνη (Serial Monitor) 27
4.3 Η Δομή του προγράμματος 28
4.4 Ψηφιακές Ακίδες (Digital Pins) 28
4.5 Αναλογικές Ακίδες Εισόδου (Analog Input Pins) 30
5. Κεφάλαιο 5ο – Ανάλυση και σχεδίαση του σταθμού 32
5.1 Σχεδιασμός Λογισμικού Σταθμού 32
5.2 Δομή Weather Data Σταθμού 34

5.3 Βιβλιοθήκη Weather Station Σταθμού 35
6. Κεφάλαιο 6ο – Σύνδεση Περιφερειακών – Αισθητήρων & Υλοποίηση 38
6.1 Κάρτα Επέκτασης Δικτύου (Ethernet Shield) 38
6.2 Αισθητήρας Καταγραφής Θερμοκρασίας / Υγρασίας DHT – 22/AM – 2302 40
6.3 Σύνδεση Αισθητήρα Ατμοσφαιρικής Πίεσης BMP – 180 43
6.4 Σύνδεση Αισθητήρα Διεύθυνσης Ανέμου 80422 44
6.5 Υλοποίηση 46
6.5.1 Πρόγραμμα και αξιοποιήσιμες βιβλιοθήκες 46
6.5.2 Ιστοσελίδες και Javascript 46
6.5.3 Επιπρόσθετες βιβλιοθήκες υποστήριξης έργου 47
6.5.4 Ανάλυση Αλγορίθμων μετεωρολογικών δεδομένων 48
7. Κεφάλαιο 7ο – Διεπαφή Χρήστη 50
7.1 Αρχική Σελίδα 51
7.2 Προβολή Στατιστικών 54
7.3 Έλεγχος Μετεωρολογικών Δεδομένων 55
8. Κεφάλαιο 8ο – Συμπεράσματα 57
Βιβλιογραφία 58
Ιστοσελίδες 58
ΠΑΡΑΡΤΗΜΑΤΑ 60
Παράρτημα Α – Συνάρτηση Διασύνδεσης Κάρτας Ethernet Shield 60
Παράρτημα Β – συνάρτηση διασύνδεσης αισθητήρα DHT-22 / AM - 2302 60
Παράρτημα Γ – συνάρτηση Διασύνδεσης αισθητήρα BMP-180 61
Παράρτημα Δ – συνάρτηση διασύνδεσης αισθητήρα διεύθυνσης ανέμου 80422 62
Παράρτημα Ε – Συνάρτηση καταγραφής δειγμάτων Ατμ. Πίεσης 62
Παράρτημα ΣΤ - Συνάρτηση Υπολογισμού Βαρομετρικής τάσης 64
Παράρτημα Ζ – Συνάρτηση αποθήκευσης βαρομετρικής τάσης 64
Παράρτημα Θ – Συνάρτηση λειτουργίας εξυπηρετητή διαδικτύου. 65

Π ί ν α κ ε ς

Πίνακας 1: Μοντέλα μικροελεγκτών Arduino 13
Πίνακας 2: Χαρακτηριστικά Arduino Mega 2560 15
Πίνακας 3: Τεχνικά Χαρακτηριστικά DHT -22 / AM – 2302 17
Πίνακας 4: Επεξήγηση κουμπιών της γραμμής εργαλείων 27
Πίνακας 5: Στοιχεία Δομής Weather Data 34
Πίνακας 6: Μέθοδοι / Ιδιότητες Κλάσης 36
Πίνακας 7: Αντιστοιχίες μεταξύ της διεύθυνσης ανέμου και αναλογικού σήματος. ..45

Εικόνες – Φωτογραφικό Υλικό

Εικόνα 1: Arduino UNO 13
Εικόνα 2: Arduino Leonardo 13
Εικόνα 3: Arduino Mega 2560 13
Εικόνα 4: Arduino LilyPad 13
Εικόνα 5: Arduino Mega ADK 13
Εικόνα 6: Arduino Fio 13
Εικόνα 7: Arduino Pro 13
Εικόνα 8: Arduino BT 13
Εικόνα 9: Arduino Nano 13
Εικόνα 10: Κάτω όψη Arduino Mega 2560 14
Εικόνα 11: Άνω όψη Arduino Mega 2560 14
Εικόνα 12: Το περιβάλλον ανάπτυξης του Arduino 16
Εικόνα 13: Διάγραμμα ροής λειτουργίας της γλώσσας Wiring 16
Εικόνα 14: Αισθητήρας θερμοκρασίας / υγρασίας DHT - 22/ AM - 2302 17
Εικόνα 15: Διάγραμμα μεταφοράς δεδομένων DHT - 22 / AM - 2302 18
Εικόνα 16: Υπόδειγμα συνδεσμολογίας του αισθητήρα με το Arduino 19
Εικόνα 17: Γραφική παράσταση σχέσης ύψους και ατμοσφαιρικής πίεσης 19
Εικόνα 18: Συνδεσμολογία του αισθητήρα BMP - 180 με την πλακέτα Arduino 20
Εικόνα 19: Αισθητήρας κατεύθυνσης ανέμου 80422 21
Εικόνα 20: Συνδεσμολογία αντιστάσεων - Εντοπισμός κατεύθυνσης του ανέμου 21
Εικόνα 21: Αντιστοιχία μεταξύ μοιρών και τάσης εξόδου του αισθητήρα 21
Εικόνα 22: Αισθητήρας μέτρησης φωτός BH - 1750FVI 22
Εικόνα 23: Συνδεσμολογία BH - 1750FVI με την πλακέτα Arduino 22
Εικόνα 24: Ρολόι Χρονισμού RTC DS - 3231 23
Εικόνα 25: Συνδεσμολογία Ρολογιού RTC DS - 3231 με την πλακέτα Arduino 23
Εικόνα 26: Άνω όψη του Arduino Ethernet Shield w5100 24
Εικόνα 27: Κάτω όψη του Arduino Ethernet Shield w5100 24
Εικόνα 28: Συνδεσμολογία Ethernet Shield με το Arduino Mega 2560 24
Εικόνα 29: Ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino 26
Εικόνα 30: Serial Monitor 27
Εικόνα 31: Διάγραμμα δραστηριότητας του Αυτόνομου Μετεωρολογικού Σταθμού 33
Εικόνα 32: Διάγραμμα δομής Weather Data 34
Εικόνα 33: Διάγραμμα κλάσης Weather Station 35
Εικόνα 34: Διάγραμμα ροής λειτουργίας Ethernet Shield 39
Εικόνα 35: Αποτελέσματα Δοκιμών διασύνδεσης κάρτας Ethernet Shield 40
Εικόνα 36: Διάγραμμα ροής διασύνδεσης με τον αισθητήρα DHT – 22/AM - 2302 . 42
Εικόνα 37: Αποτελέσματα ελέγχου αισθητήρα DHT – 22 / AM - 2302 42
Εικόνα 38: Διάγραμμα ροής αισθητήρα ατμοσφαιρικής πίεσης BMP - 180 43
Εικόνα 39: Έλεγχος λειτουργίας του αισθητήρα ατμοσφαιρικής πίεσης BMP - 180 . 44
Εικόνα 40: Έξοδος αισθητήρα κατεύθυνσης ανέμου 80422 46
Εικόνα 41: Έλεγχος διασύνδεσης με τον διακομιστή NTP 47
Εικόνα 42: Διάγραμμα ροής διαδικασίας εξυπηρετητή διαδικτύου 50

Εικόνα 43: Αρχική Σελίδα του Σταθμού 51
Εικόνα 44: Σελίδα προβολής Στατιστικών stats.html 54
Εικόνα 45: Πρόγνωση Καιρού για τη Θεσσαλονίκη στις 01-02/06/2015 55

Περίληψη

Στην παρούσα εργασία παρουσιάζεται η διαδικασία προγραμματισμού της υπολογιστικής πλατφόρμας Arduino με σκοπό την αξιοποίηση της ως αυτόνομο μετεωρολογικό σταθμό, με δυνατότητες καταγραφής, επεξεργασίας και αποθήκευσης των μετεωρολογικών δεδομένων της περιοχής, με σκοπό την βραχυπρόθεσμη πρόβλεψη του καιρού σύμφωνα με τα δεδομένα που έχουν συλλεχθεί, χρησιμοποιώντας ένα απλοποιημένο μετεωρολογικό μοντέλο πρόγνωσης καιρού.

Αρχικά γίνεται μια ανάλυση των μετεωρολογικών φαινομένων και παρουσιάζεται η σημασία τους για την πρόγνωση καιρού καθώς και η μεθοδολογία καταγραφής τους, σύμφωνα με τους αλγορίθμους που έχει θεσπίσει ο Παγκόσμιος Οργανισμός Μετεωρολογίας.

Παρουσιάζονται οι αισθητήρες που χρησιμοποιήθηκαν, τα κριτήρια επιλογής τους, ο τρόπος συνδεσμολογίας και οι βιβλιοθήκες που αξιοποιήθηκαν για την διεπαφή τους με την πλακέτα Arduino.

Αφού ολοκληρωθούν όλοι οι απαραίτητοι έλεγχοι για την διαπίστωση σφαλμάτων στον αλγόριθμο συλλογής και καταγραφής των μετεωρολογικών δεδομένων, σχεδιάζεται η κύρια οθόνη διεπαφής με την οποία αλληλεπιδρά ο χρήστης με τον σταθμό (Graphic User Interface).

Γίνεται επεξήγηση της διαδικασίας καταγραφής των μετεωρολογικών δεδομένων του σταθμού και παρουσιάζονται τα προβλήματα που εντοπίστηκαν κατά την διάρκεια των δοκιμών και αναλύονται οι τρόποι αντιμετώπισης τους.

Για να διαπιστωθεί η αξιοπιστία των αισθητήρων καθώς και των αλγορίθμων καταγραφής, γίνεται σύγκριση των δεδομένων που συλλέχθηκαν με τα αντίστοιχα από υπάρχον πιστοποιημένο μετεωρολογικό σταθμό της περιοχής, το αποτέλεσμα της οποίας είναι ικανοποιητικό.

Στην τελική φάση γίνεται έλεγχος της ποιότητας του μετεωρολογικού μοντέλου όπου καταγράφονταν η πρόγνωση που εμφάνιζε ο σταθμός με βάση τα δεδομένα που είχαν συλλεχθεί. Στην συνέχεια μετά από παρατήρηση των καιρικών συνθηκών της περιοχής γίνονταν σύγκριση αυτών με την πρόβλεψη του σταθμού.

Ο Αυτόνομος Μετεωρολογικός Σταθμός με χρήση της υπολογιστικής πλατφόρμας Arduino, αποτελεί ιδανική λύση σε όσους επιθυμούν να γνωρίζουν τα καιρικά φαινόμενα που θα επικρατήσουν τις επόμενες δώδεκα ώρες στην περιοχή εμβέλειάς του, χωρίς η δαπάνη κατασκευής του σταθμού να απαιτεί υψηλό κόστος.

1

Κεφάλαιο 1ο

Εισαγωγή

1 Κεφάλαιο 1^{ο} – Εισαγωγή

Για την παρατήρηση των καιρικών φαινομένων έχουν κατασκευαστεί ποικίλα μετεωρολογικά συστήματα τα οποία κυμαίνονται σε ένα ευρύ φάσμα τιμών και δυνατοτήτων, ώστε να απευθύνονται σε κοινό το οποίο αποτελείται από ερασιτέχνες καθώς και επαγγελματίες όπως αγρότες, μετεωρολόγους, αλιείς κτλ. Αυτές οι λύσεις μπορεί να είναι από μια απλή απεικόνιση των δεδομένων, έως πολύπλοκα συστήματα τα οποία περιλαμβάνουν την καταγραφή και τη βραχυπρόθεσμη πρόβλεψη του καιρού. Όμως σε τι θα χρησίμευε ένα σύστημα καταγραφής και πρόβλεψης μετεωρολογικών φαινομένων στον απλό χρήστη; Η αλήθεια είναι ότι στις καθημερινές του ασχολίες, ο σύγχρονος άνθρωπος της πόλης δεν επηρεάζεται ιδιαίτερα από τα φαινόμενα του καιρού. Όμως υπάρχουν άνθρωποι οι οποίοι επηρεάζονται άμεσα και σημαντικά από τις μεταβολές των καιρικών συνθηκών. Για παράδειγμα ο αγρότης, ο οποίος θέλει να γνωρίζει με ακρίβεια τις μεταβολές του καιρού ώστε να προγραμματίζει τις εργασίες του και να προετοιμάζεται κατάλληλα για κάθε αλλαγή των καιρικών φαινομένων. Τα περισσότερα όμως δελτία καιρού, τα οποία μεταδίδονται από τα τηλεοπτικά μέσα, αναφέρονται συνήθως στις αστικές περιοχές όπου και διαμένει το μεγαλύτερο ποσοστό του πληθυσμού της χώρας. Στις περιπτώσεις αυτές λοιπόν, όπου οι πληροφορίες οι οποίες δίδονται στους επαγγελματίες είναι λιγοστές και ελλιπείς, την λύση έρχονται να δώσουν οι φορητοί μετεωρολογικοί σταθμοί, οι οποίοι συλλέγουν δεδομένα για την περιοχή που είναι εγκατεστημένοι και εξάγουν μια σχετικά ασφαλή πρόγνωση. Συνήθως, τα μηχανήματα αυτά έχουν υψηλό κόστος, με αποτέλεσμα για ορισμένους χρήστες και σε συνάρτηση πάντα με τα οφέλη τα οποία προκύπτουν από την χρήση τέτοιον σταθμών, να είναι απαγορευτικά.

Στον αντίποδα αυτών των συστημάτων, το Arduino αποτελεί ένα χαμηλού κόστους υπολογιστικό σύστημα, με μεγάλο εύρος περιφερειακών συσκευών, εύκολα προγραμματίσιμο και με μεγάλη κοινότητα μελών, οι οποίοι είναι έτοιμοι να παράσχουν κάθε βοήθεια μεταξύ τους.

Στόχος της παρούσης εργασίας είναι η κατασκευή ενός μετεωρολογικού σταθμού ο οποίος θα συλλέγει τα απαραίτητα δεδομένα μέσων των αισθητήρων του και στην συνέχεια με βάση τα δεδομένα αυτά θα αποδίδει μια αξιόπιστη πρόβλεψη για τις

επόμενες 12 έως 24 ώρες.

Ένα βασικό μειονέκτημα των περισσοτέρων μετεωρολογικών σταθμών είναι ότι απαιτείται η φυσική παρουσία του χρήστη στην τοποθεσία που βρίσκεται εγκατεστημένος ο σταθμός. πρόβλημα αυτό αντιμετωπίζεται εν μέρη στα ποιο ακριβά μοντέλα τα οποία διαθέτουν ασύρματη μετάδοση των δεδομένων τους. Ο υπό κατασκευή σταθμός θα πρέπει να έχει όσο το δυνατόν μεγαλύτερο εύρος δράσης και η πρόσβαση σε αυτόν θα επιτρέπεται με μέσα που διαθέτει ήδη ο χρήστης ώστε να μην απαιτηθεί η κατασκευή ή αγορά του δέκτη, μειώνοντάς με αυτόν τον τρόπο το τελικό κόστος κατασκευής του.

Σε αυτήν την περίπτωση η ιδανική λύση που προτείνεται είναι η σύνδεση του σταθμού στο διαδίκτυο, καθώς οι τεχνολογίες του μας επιτρέπουν να εκτελούμε εργασίες οι οποίες μας διευκολύνουν είτε σε επαγγελματικό είτε σε προσωπικό επίπεδο. Με αυτόν τον τρόπο, η προβολή των δεδομένων θα γίνεται μέσω ιστοσελίδας έτσι ώστε με την χρήση ενός προσωπικού υπολογιστή ή ακόμα ενός «έξυπνου» κινητού τηλεφώνου θα επιτρέπεται η πρόσβαση στα δεδομένα του σταθμού από οποιοδήποτε σημείο και να βρίσκεται ο χρήστης, χωρίς επιπλέον κόστος.

2

Κεφάλαιο 2ο

Αντικείμενο Εργασίας

2 Κεφάλαιο 2^{ο} – Αντικείμενο Εργασίας

Αντικείμενο της παρούσης εργασίας είναι η κατασκευή ενός αυτόνομου μετεωρολογικού σταθμού (ΑΜΣ) με δυνατότητα πρόγνωσης καιρού, κάνοντας χρήση απλοποιημένων μοντέλων μετεωρολογίας. Ο ΑΜΣ θα βασίζει την λειτουργία του στην υπολογιστική πλατφόρμα Arduino, μία πρότυπη τεχνολογία η οποία διαθέτει διαδραστικότητα με το περιβάλλον μέσω των υφιστάμενων αισθητήρων και αποτελεί μια ιδανική λύση για εφαρμογές αυτοματισμού και απλοποιημένης ρομποτικής. Ο μικροελεγκτής θα δέχεται στις εισόδους του, αισθητήρες καταγραφής μετεωρολογικών φαινομένων οι οποίοι είναι:

- αισθητήρας θερμότητας
- αισθητήρας υγρασίας
- αισθητήρας ατμοσφαιρικής πίεσης
- αισθητήρας διεύθυνσης ανέμου

Στην συνέχεια οι τιμές των αισθητήρων θα καταγράφονται σύμφωνα με τα διεθνή πρότυπα που έχουν καθοριστεί από τον Παγκόσμιο Οργανισμό Μετεωρολογίας (World Meteorological Organization - WMO), και θα αποθηκεύονται στην κάρτα SD για ιστορικούς και στατιστικούς λόγους. Το Arduino θα προγραμματισθεί έτσι ώστε να λειτουργεί και ως εξυπηρετητής διαδικτύου (web server), με σκοπό την εμφάνιση των δεδομένων των αισθητήρων στο διαδικτυακό τόπο του ΑΜΣ με την χρήση οποιουδήποτε φυλλομετρητή. Τέλος, με χρήση απλοποιημένων μετεωρολογικών μοντέλων στην ιστοσελίδα του σταθμού, θα εμφανίζεται και η πρόγνωση του καιρού της περιοχής της οποίας είναι εγκατεστημένος ο σταθμός.

Αρχικά θα πρέπει να συνδεθούν οι αισθητήρες υγρασίας, ατμοσφαιρικής πίεσης και θερμοκρασίας. Στην συνέχεια, αφού γίνει ο έλεγχος της λειτουργίας τους θα υλοποιηθεί το λογισμικό διασύνδεσης τους. Απαιτείται η μελέτη και κατασκευή του γραφικού περιβάλλοντος διεπαφής (Graphic User Interface – GUI) υπό την μορφή ιστοσελίδας, στο οποίο θα εισέρχεται ο χρήστης από φυλλομετρητή της επιλογής του και θα προβάλλονται τα δεδομένα και η πρόγνωση του καιρού.

Τέλος κρίνεται σκόπιμο, χωρίς να είναι απαραίτητο, η τοποθέτηση μιας οθόνης

υγρών κρυστάλλων (LCD) η οποία θα έχει ως σκοπό την εμφάνιση των ενδείξεων του σταθμού (θερμοκρασία, υγρασία, ατμοσφαιρική πίεση, ταχύτητα ανέμου), και αν απαιτηθεί, πληροφορίες για την λειτουργία του (διεύθυνση ip, κατάσταση αισθητήρων κλπ).

2.1 Αυτόνομος Μετεωρολογικός Σταθμός

Αυτόνομος μετεωρολογικός σταθμός (ΑΜΣ) ορίζεται ένας μετεωρολογικός σταθμός στον οποίο γίνονται λήψεις μετεωρολογικών δεδομένων με αυτοματοποιημένη διαδικασία και στην συνέχεια τοπικά ή σε κεντρικές μονάδες όπου αποστέλλονται, γίνεται η επεξεργασία των δεδομένων τους. Στους ΑΜΣ, οι μετρήσεις των οργάνων λαμβάνονται από μια κεντρική μονάδα δεδομένων, στην περίπτωση μας από το Arduino. Ο αυτόνομος σταθμός καιρού μπορεί να αποτελούν μέρος ενός δικτύου συνολικής καταγραφής, ονομάζονται δε συνήθως «Αυτοματοποιημένο Σύστημα Παρατήρησης Καιρικών Συνθηκών» (Automated Weather Observing System - AWOS) ή «Αυτοματοποιημένο Σύστημα Παρατήρησης Επιφάνειας» (Automatic System Observation Surface - ASOS). Όμως η ευρεία χρήση αναφοράς σε ένα τέτοιο σύστημα είναι ΑΜΣ, αν και ο όρος "σταθμός" δεν ανταποκρίνεται πλήρως σε αυτόν τον ορισμό. Παρ 'όλα αυτά, στην παρούσα εργασία η αναφορά στον όρο ΑΜΣ θα εννοεί ένα τέτοιο σύστημα.

2.2 Μετεωρολογικά Φαινόμενα

Καιρός ονομάζεται το σύνολο των μετεωρολογικών φαινομένων που παρατηρούνται στην ατμόσφαιρα της Γης σε καθορισμένο τόπο και χρόνο. Τέτοια φαινόμενα είναι η θερμοκρασία, η ατμοσφαιρική πίεση, οι κινήσεις των ανέμων, η παρουσία νεφών κλπ. Ο καιρός προσδιορίζεται για μια συγκεκριμένη χρονική στιγμή και η πρόγνωση του βασίζεται στα αποτελέσματα των παρατηρήσεων του. Στην πραγματικότητα είναι μια συνάρτηση στην οποία εισάγονται μετεωρολογικά δεδομένα, ο τόπος, ο χρόνος κλπ. και με βάση τις καταγραφές και παρατηρήσεις που έχουν γίνει, εξάγεται ένα αποτέλεσμα. Το αποτέλεσμα αυτό αποτελεί την πρόγνωση του καιρού και η ακρίβεια του εξαρτάται από το μετεωρολογικό μοντέλο που θα χρησιμοποιηθεί. Με την εξέλιξη των υπολογιστικών συστημάτων τα οποία πλέον εκτελούν τρισεκατομμύρια

πράξεις ανά δευτερόλεπτο και τους δορυφόρους από τους οποίους μπορούμε να παρατηρούμε τα μετεωρολογικά φαινόμενα με μεγαλύτερη ακρίβεια και για μεγαλύτερο γεωγραφικό εύρος, τα μοντέλα αυτά έχουν εξελιχθεί και εξάγουν ασφαλέστερα και πιο μακροπρόθεσμα αποτελέσματα.

2.2.1 Ατμοσφαιρική Πίεση

Η ατμοσφαιρική πίεση είναι ένα από τα πιο σημαντικά μετεωρολογικά στοιχεία, γιατί οι καιρικές καταστάσεις και οι μεταβολές τους συνδέονται άμεσα μαζί της. Για τον λόγο αυτό, το μοντέλο που θα χρησιμοποιηθεί για την πρόγνωση του ΑΜΣ βασίζεται κυρίως στις μεταβολές της ατμοσφαιρικής πίεσης. Ατμοσφαιρική πίεση ή «Βαρομετρική πίεση» ονομάζεται η πίεση που ασκεί η ατμόσφαιρα, με το βάρος της, στην επιφάνεια της Γης.

Στην επιφάνεια της Γης η ατμοσφαιρική πίεση ισούται κατά μέσον όρο με το βάρος στήλης ύδατος ύψους 11 μ.(m) περίπου, ή 1 εκατομμύριο δύνες ανά cm2. Στην μετεωρολογία 1000 δύνες/cm2 αντιστοιχούν σε ένα χιλιοστόμετρο (milibar). Κατά μέσο όρο στην επιφάνεια της γης η ατμοσφαιρική πίεση είναι 1000 milibar. Με απόφαση του Διεθνούς Μετεωρολογικού Οργανισμού (WMO) καθιερώθηκε ως μονάδα μέτρησης της ατμοσφαιρικής πίεσης το Εκτοπασκάλ (1 hPa) το οποίο η αντιστοιχία είναι 1 hPa=1 milibar.

Η πίεση μειώνεται ανάλογα με υψόμετρο. Για τον λόγο αυτό οι μετρήσεις που γίνονται σε ύψος μεγαλύτερο από αυτό της επιφάνειας της θάλασσας, πρέπει να τροποποιηθούν ώστε να αντιστοιχούν σε μηδενικό ύψος. Η ενέργεια αυτή ονομάζεται ως αναγωγή στην επιφάνεια της θάλασσας.

2.2.2 Υγρασία

Ο ατμοσφαιρικός αέρας περιέχει υδρατμούς με διαφορετική ποσότητα από τόπο σε τόπο και από ώρα σε ώρα. Ο αέρας όμως δεν είναι δυνατόν να περιέχει απεριόριστη ποσότητα υδρατμών, αλλά για κάθε θερμοκρασία υπάρχει μια μέγιστη δυνατή περιεκτικότητα υδρατμών. Όταν ο αέρας περιέχει τη μέγιστη τέτοια ποσότητα ονομάζεται κορεσμένος.

Όσο ψυχρότερος είναι ο αέρας τόσο μικρότερη ποσότητα υδρατμών μπορεί να συγκρατήσει. Αν λοιπόν μια μάζα υγρού και θερμού αέρα ψυχθεί θα φτάσει σε μια θερμοκρασία όπου δεν είναι δυνατόν πλέον να συγκρατήσει άλλους τους υδρατμούς

από τους οποίους περιέχει. Οι υδρατμοί που περισσεύουν θα συμπυκνωθούν ως σταγονίδια πάνω στα αιωρούμενα μικροσωματίδια και θα δημιουργήσουν το νέφος. Αν δε, συμπυκνωθούν πάνω σε ψύχρα αντικείμενα θα δημιουργήσουν τη δρόσο. Η θερμοκρασία στην οποία ο ακόρεστος αέρας καθώς ψύχεται φτάνει στο κορεσμό, ονομάζεται σημείο δρόσου.

Η υγρασία παίζει σημαντικό ρόλο στην μορφή του κλίματος, τη βλάστηση και τη ζωή ενός τόπου. Μετράμε την υγρασία με τα υγρόμετρα, τα οποία δείχνουν πόσους υδρατμούς περιέχει η ατμόσφαιρα επί της εκατό (%) (όπου 100 θεωρούνται οι υδρατμοί οι οποίοι θα περιέχονταν για την ιδία θερμοκρασία αν είχαμε κορεσμό).

2.2.3 Άνεμος

Άνεμος ονομάζεται η ροή του αέρα πάνω από την επιφάνεια της Γης και οφείλεται στη μετακίνηση του, ανάμεσα σε δυο περιοχές διαφορετικής πίεσης. Η ροή του αέρα γίνεται από τις περιοχές με υψηλότερη πίεση σε αυτή με την χαμηλότερη. Ο όρος άνεμος αναφέρεται πάντοτε σε οριζόντιες μετακινήσεις του ατμοσφαιρικό αέρα.

Η διεύθυνση ανέμου είναι το σημείο του ορίζοντα από όπου πνέει ο άνεμος. Η ένταση του ανέμου εκφράζεται από την ταχύτητα του. Η διεύθυνση μετριέται με τον ανεμοδείκτη. Η διαίρεση του ορίζοντα σε 16 διευθύνσεις (ανά 22,5 μοίρες) λέγεται ανεμολόγιο. Οι κύριοι άνεμοι είναι οι Βορράς, Ανατολικός, Νότιος, Δυτικός στις 00, 90, 180, 270 μοίρες αντίστοιχα. Δευτερεύοντες άνεμοι είναι ο Βορειοανατολικός, Νοτιοανατολικός, Νοτιοδυτικός, Βορειοδυτικός με μοίρες 45, 135, 225, 315 αντίστοιχα.

2.2.4 Θερμοκρασία

Θερμοκρασία ατμόσφαιρας ονομάζεται η θερμοκρασία την οποία έχει ο ατμοσφαιρικός αέρας πάνω από μια περιοχή. Η πρόγνωση του καιρού σε μια περιοχή βασίζεται κυρίως στη γνώση της εκάστοτε ατμοσφαιρικής πίεσης και της θερμοκρασίας της ατμόσφαιρας της υπ΄ όψιν περιοχής και των γύρω αυτής εκτάσεων.

Συνεπώς η αναφορά της θερμοκρασίας σχετίζεται πάντα με κάποια περιοχή, είτε μικρή, είτε μεγάλη, στην περίπτωση μας δε στην περιοχή εμβέλειας του σταθμού. Η

αναφορά σε παγκόσμια κλίμακα, ανάγεται σε αντικείμενο άλλης επιστήμης.

Η θερμοκρασία της ατμόσφαιρας μετριέται με τα θερμόμετρα και υπάρχουν διάφορες κλίμακες μέτρησης, με συνηθισμένες κλίμακες τις Κελσίου (Celsius, σύμβολο Cº) , Κέλβιν (Kelvin, σύμβολο Kº) και Φαρενάιτ (Fahrenheit, σύμβολο Fº). Στην Ελλάδα χρησιμοποιείται η κλίμακα Κελσίου και ορίζεται ως « Το σημείο βρασμού του νερού είναι στους 100 Cº και το σημείο παγοποίησης του, στους 0 Cº».

2.2.5 Σημείο Δρόσου

Σημείο δρόσου χαρακτηρίζεται το σημείο εκείνο της θερμοκρασίας που όταν οι υδρατμοί ψυχθούν δημιουργούν το φαινόμενο της δρόσου, δηλαδή τις σταγόνες δρόσου. Στη θερμοκρασία αυτή εξυπακούεται πως όταν ο αέρας είναι κορεσμένος και δεν μπορεί να συγκρατήσει άλλους υδρατμούς η σχετική υγρασία να είναι 100%.

Σημειώνεται όμως ότι η θερμοκρασία κορεσμού της ατμόσφαιρας ή του "σημείου δρόσου" μπορεί να είναι οποιαδήποτε θερμοκρασία, πάνω από τους 0°C.

Η θερμοκρασία αυτή εξαρτάται μόνο από την ποσότητα των υδρατμών που περιέχει 1 κυβικό μέτρο αέρος, συνεπώς εξαρτάται από την απόλυτη υγρασία. Η θερμοκρασία του σημείου δρόσου αποτελεί σπουδαίο μετεωρολογικό στοιχείο για ένα τόπο και γι’ αυτό πάντοτε αναφέρεται στους μετεωρολογικούς χάρτες με τα σύμβολα D.P. από τα αρχικά του αγγλικού όρου Dew Point (Δρόσου Σημείο).

3

Κεφάλαιο 3ο

Πλατφόρμα Arduino

3. Κεφάλαιο 3^{ο} – Πλατφόρμα Arduino

3.1 Γενικά Στοιχεία

Το Arduino είναι μια πλακέτα που βασίζεται σε ανοιχτού υλικού μικροεπεξεργαστή, ο οποίος προγραμματίζεται σε γλώσσα Wiring, μια παραλλαγή της γνωστής γλώσσας C++. Το Arduino χαρακτηρίζεται για το ελάχιστο κόστος κατασκευής του, την χαμηλή κατανάλωση σε ισχύ, κάτι που αξιοποιείται κυρίως σε φορητές ή και αυτόνομες λύσεις, το μεγάλο πλήθος από περιφερειακές κάρτες για πιο πολύπλοκες και εξειδικευμένες λύσεις, όπως η κάρτα Ardupilot η οποία περιέχει συναρτήσεις και αισθητήρες για την δημιουργία συστήματος αυτόματης πλοήγησης και χρησιμοποιείται για την κατασκευή τηλεκατευθυνόμενων αεροσκαφών.

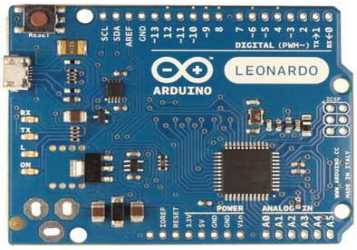
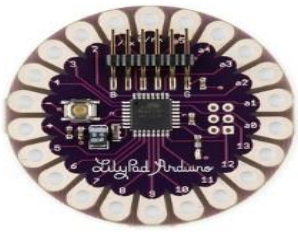
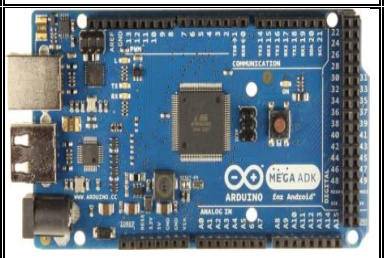
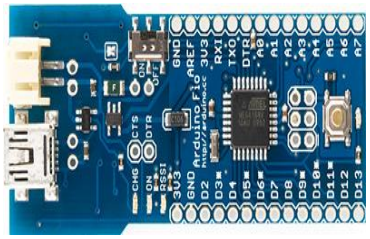
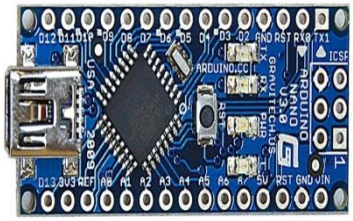
Η αρχιτεκτονική Arduino, αποτελείται από ένα σύνολο μικροελεγκτών υλικών συστημάτων που με την κατάλληλη παροχή λογισμικού, αυτοματοποιούν δραστηριότητες που επιθυμούν οι χρήστες. Στην ενότητα αυτή, παρουσιάζεται ο μικροελεγκτής Arduino που χρησιμοποιήθηκε, τα χαρακτηριστικά του, διαθέσιμα εξαρτήματα που μπορούν να συνδεθούν απ` ευθείας, αλλά και αυτά ανάλυση εκείνων που χρησιμοποιήθηκαν για την ανάπτυξη του συστήματος.

Το σύστημα στήθηκε πάνω σε μικροελεγκτή της αρχιτεκτονικής Arduino. Πρόκειται για ανοικτού λογισμικού πλατφόρμα πρωτοτύπων ηλεκτρονικών συσκευών που βασίζονται στην ευελιξία και στην ευκολία χρήσης υλικού και λογισμικού. Οι Arduino συσκευές μπορούν να αλληλεπιδρούν με το περιβάλλον κάνοντας λήψη σημάτων μέσα από μια ποικιλία αισθητήρων. Τα έργα που βασίζονται σε αυτούς τους μικροελεγκτές μπορούν να είναι αυτόνομα ή μπορούν να επικοινωνούν με το λογισμικό που τρέχει σε έναν υπολογιστή. Για την ανάπτυξη του Διαδικτυακού Μετεωρολογικού σταθμού χρησιμοποιήθηκε το Arduino Mega 2560.

3.2 Μοντέλα μικροελεγκτών Arduino

Στον παρακάτω πίνακα παρουσιάζονται τα διάφορα μοντέλα μικροελεγκτών Arduino. Οι διαφορές τους συνήθως είναι στους ακροδέκτες (pins) που έχουν, στην τάση εισόδου και εξόδου, καθώς και στα χαρακτηριστικά των συστημάτων που υλοποιούν:

- Arduino UNO (Συνήθως για συστήματα έρευνας ή ερασιτεχνικά)
- Arduino Leonardo
- Arduino Mega 2560 (Συνήθως για βιομηχανικά συστήματα)
- Arduino LilyPad
- Arduino Mega ADK
- Arduino Fio
- Arduino Pro
- Arduino BT
- Arduino Nano

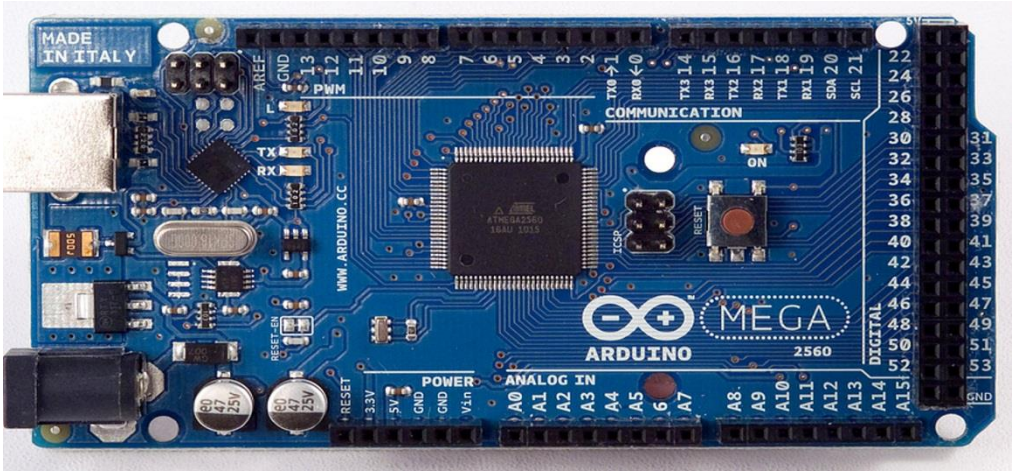
		
Εικόνα 1: Arduino UNO	**Εικόνα 2: Arduino Leonardo**	**Εικόνα 3: Arduino Mega 2560**
		
Εικόνα 4: Arduino LilyPad	**Εικόνα 5: Arduino Mega ADK**	**Εικόνα 6: Arduino Fio**
		
Εικόνα 7: Arduino Pro	**Εικόνα 8: Arduino BT**	**Εικόνα 9: Arduino Nano**

Πίνακας 1: Μοντέλα μικροελεγκτών Arduino

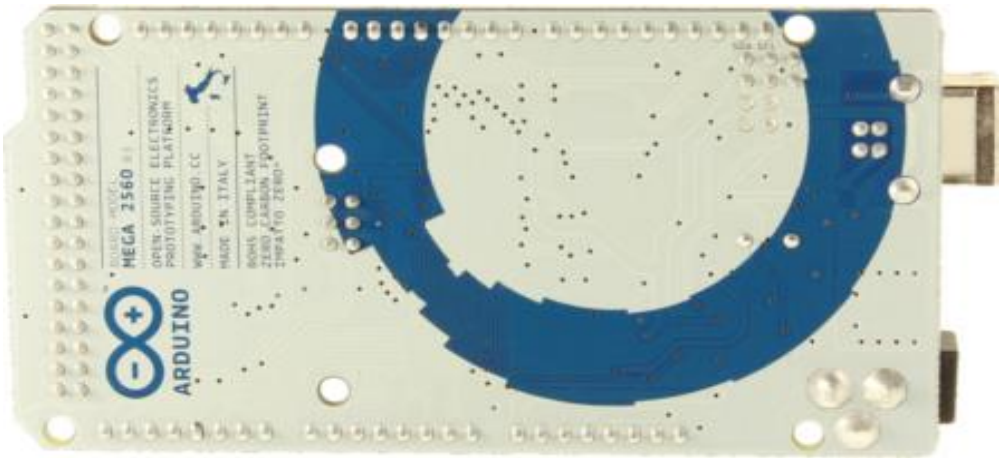
3.2.1 Το υλικό (Arduino Mega 2560)

Ο Arduino Mega 2560 είναι ένας μικροελεγκτής βασισμένος στο Chip ATmega2560 της Atmel. Διαθέτει 54 ψηφιακές εισόδους / εξόδους από τις οποίες οι 14 μπορούν να χρησιμοποιηθούν και σαν PWM έξοδοι. Επίσης διαθέτει 16 αναλογικές εισόδους, 4 σειριακές θύρες (UART), ένα κρυσταλλικό ταλαντωτή συχνότητας 16MHz, μια USB σύνδεση, είσοδο τροφοδοσίας, ένα ICSP Header και κουμπί για Reset. Η τάση λειτουργίας του είναι 5V. Η τάση τροφοδοσίας που συνίσταται είναι από 7V έως 12V και το ρεύμα λειτουργίας για κάθε είσοδο και έξοδο είναι 40mA.

Παράθεση των δύο όψεων του Arduino Mega 2560:



Εικόνα 11: Άνω όψη Arduino Mega 2560



Εικόνα 10: Κάτω όψη Arduino Mega 2560

Παράθεση πίνακα χαρακτηριστικών του Arduino Mega 2560:

Μικροελεγκτής	ATmega2560
Τάση λειτουργίας	5 V
Τάση εισόδου (συνιστάται)	7-12 V
Τάση εισόδου (όρια)	6-20 V
Ψηφιακές I / O καρφίτσες	54 (εκ των οποίων οι 14 προβλέπουν PWM εξόδου)
Αναλογικές Είσοδοι (Pins)	16
DC Ρεύμα ανά I / O Pin	40 mA
DC Ρεύμα για 3,3V Pin	50 mA
Flash Memory	256 KB εκ των οποίων 8 KB που χρησιμοποιούνται από τον Boot loader.
SRAM	8 KB
EEPROM	4 KB
Ταχύτητα Ρολογιού	16 MHz

Πίνακας 2: Χαρακτηριστικά Arduino Mega 2560

Τροφοδοσία

Ο Arduino Mega 2560 μπορεί να τροφοδοτείται μέσω της σύνδεσης USB ή με εξωτερικό τροφοδοτικό. Η πηγή ενέργειας επιλέγεται αυτόματα.

Εξωτερική (μη – USB) ισχύς μπορεί να προέρχεται είτε από έναν AC σε DC μετασχηματιστή ή μπαταρία. Ο μετασχηματιστής μπορεί να συνδεθεί με τη σύνδεση ενός βύσματος 2,1mm με κέντρο θετικό στην υποδοχή ρεύματος.

Ο Arduino Mega 2560 μπορεί να λειτουργήσει με εξωτερική παροχή των 6 έως 20 Volt, όμως σε περίπτωση που τροφοδοτηθεί με λιγότερο από 7 V μπορεί να είναι ασταθής. Εάν τροφοδοτηθεί με περισσότερο από 12 V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να καταστραφεί. Η συνιστώμενη τάση τροφοδοσίας είναι από 7 – 12 Volt.

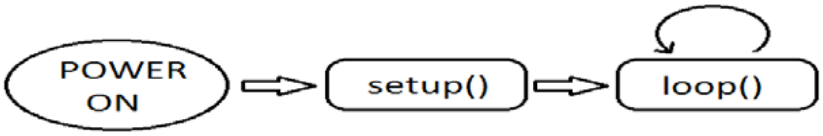
Ο Arduino Mega 2560 διαφέρει από όλους τους προηγούμενούς του, γιατί δεν χρησιμοποιεί FTDI USB – to – Serial ως Chip οδηγού, αλλά χρησιμοποιεί τον Atmega8U2 της Atmel που έχει προγραμματιστεί ως USB – to – Serial μετατροπέας, με αποτέλεσμα πολύ μεγαλύτερες ταχύτητες στην σειριακή σύνδεση.

3.2.2 Το λογισμικό λειτουργίας του Arduino

Το Ολοκληρωμένο Περιβάλλον Ανάπτυξης (Intergraded Development Environment – IDE) με το οποίο θα προγραμματίσουμε το λογισμικό του μετεωρολογικού σταθμού, έχει υλοποιηθεί με την γλώσσα Java η οποία εκτελείται ανεξάρτητα από το λειτουργικό ή υλικό του υπολογιστή, είναι ευρέως διαδεδομένη, και έχει πολλές ομοιότητες με την γλώσσα C. Κατά συνέπεια, το συγκεκριμένο IDE είναι πολλαπλού λειτουργικού (multi-platform) και μπορεί να δουλεύει άψογα σε κάθε λειτουργικό σύστημα (Windows, Linux, OSX). Η γλώσσα wiring είναι μια παραλλαγή της C++ και όπως και το IDE, είναι μια πλατφόρμα ανοιχτού λογισμικού που χρησιμοποιείται για προγραμματισμό μικροελεγκτών. Έχει επίσης σχεδιαστεί λαμβάνοντας υπόψη μια ευρεία γκάμα χρηστών, από αρχάριους μέχρι επαγγελματίες, με σκοπό την δημιουργία μιας κοινότητας η οποία θα ανταλλάσσει ιδέες, τεχνογνωσία και εμπειρία, με τελικό στόχο την κατασκευή "έξυπνων συσκευών" οι οποίες θα περιορίζονται μόνο από την φαντασία του εκάστοτε δημιουργού τους.

Εικόνα 12: Το περιβάλλον ανάπτυξης του Arduino

Κύριο χαρακτηριστικό της wiring είναι ότι απαιτεί την ύπαρξη δυο τουλάχιστον συναρτήσεων. Της setup() και της loop(). Η setup() περιέχει όλες εκείνες τις μεταβλητές οι οποίες απαιτούνται για την αρχικοποίηση του προγράμματος. Η loop(), αφού έχει εκτελεστεί η setup(), περιέχει την δομή του υπόλοιπου προγράμματος και εκτελείται συνεχόμενα πλην της περίπτωσης να έχει δοθεί εντολή τερματισμού μετά από συγκεκριμένες συνθήκες.



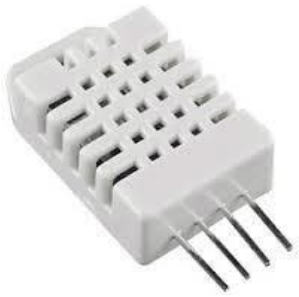
Εικόνα 13: Διάγραμμα ροής λειτουργίας της γλώσσας Wiring

3.3 Αισθητήρες καταγραφής

Οι αισθητήρες καταγραφής συνδέονται στις ψηφιακές και αναλογικές εισόδους του Arduino το οποίο κάνοντας χρήση των κατάλληλων βιβλιοθηκών, διαβάζει τις τιμές τάσης στην είσοδο του και στην συνέχεια μετατρέπει την εισερχόμενη τάση σε τιμές τις οποίες μπορούμε να αξιοποιήσουμε.

3.3.1 Αισθητήρας θερμοκρασίας / υγρασίας DHT - 22/ AM - 2302

Ο αισθητήρας DHT 22 είναι ένας φθηνός και αξιόπιστος αισθητήρας. Έχει μικρό μέγεθος, μικρή κατανάλωση ρεύματος και μπορεί να μεταδίδει δεδομένα στην πλακέτα κάθε 2 sec. Λόγω αυτών των ιδιαιτεροτήτων του έχει χρησιμοποιηθεί σε πολλές εφαρμογές και υπάρχει πληθώρα βιβλιοθηκών στο διαδίκτυο.



Εικόνα 14: Αισθητήρας θερμοκρασίας / υγρασίας DHT - 22/ AM - 2302

Τα τεχνικά χαρακτηριστικά του είναι:

	Υγρασία	Θερμοκρασία
Τάση	3,3 – 5 V	3,3 – 5 V
Έξοδος	Ψηφιακή	Ψηφιακή
Αισθητήρας	Πολυμερής Πυκνωτής	DS18B20
Εύρος	0 – 100% Rh	-40 – 125 *C*°
Ακρίβεια	2 – 5 %	± 0,5 *C*°
Δειγματοληψία	0,5 Hz	0,5 Hz

Πίνακας 3: Τεχνικά Χαρακτηριστικά DHT -22 / AM – 2302

Σύμφωνα με τον WMO η ευαισθησία του αισθητήρα θερμοκρασίας για καταγραφή περιβαλλοντολογικής θερμοκρασίας έχει οριστεί από -30 *C*° έως 50 *C*° και για την υγρασία από 5% έως 100%, άρα η χρήση του εν λόγω αισθητήρα είναι η ιδανική δεδομένου του κόστους του.

Όπως παρατηρούμε στο σχήμα το Arduino στέλνει ένα σήμα έναρξης της τάξης των 500us περίπου και αναμένει απάντηση από τον αισθητήρα. Ο αισθητήρας με την μειώνει το σήμα στα 80us στέλνει εκ νέου ένα όμοιο σήμα και τέλος στέλνει ένα ακόμα σήμα της τάξης των 50us (διαδικασία χειραψίας). Τα επόμενα σήματα που ακολουθούν είναι ψηφιακά με αντιστοιχία στα 28 us για το 0 και 70 για το λογικό 1. Κάθε δύο δευτερόλεπτα, ο αισθητήρας εκπέμπει 40 bit πληροφορίας τα οποία αντιστοιχούν σε:

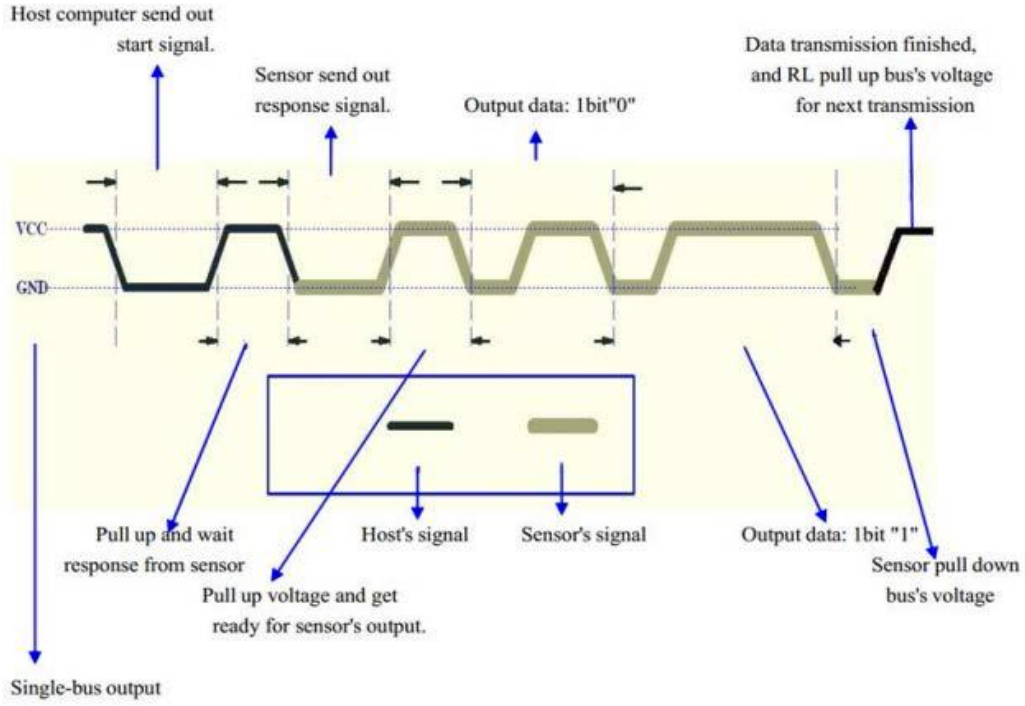
• 8 bit για την τιμή της υγρασίας

• 8 bit για την ακέραια τιμή της υγρασίας

• 8 bit για την τιμή της θερμοκρασίας

• 8 bit για την ακέραια τιμή της θερμοκρασίας

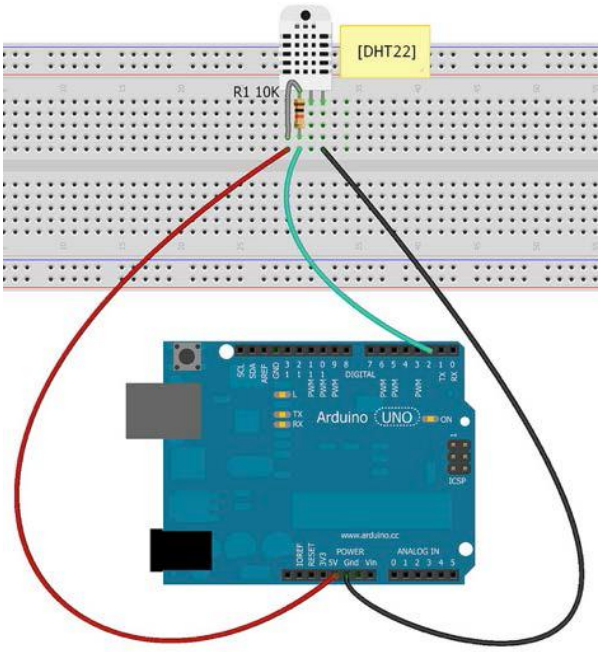
• 8 bit για το bit ελέγχου ισοτιμίας των δεδομένων.



Εικόνα 15: Διάγραμμα μεταφοράς δεδομένων DHT - 22 / AM - 2302

Συνδεσμολογία με την πλακέτα Arduino

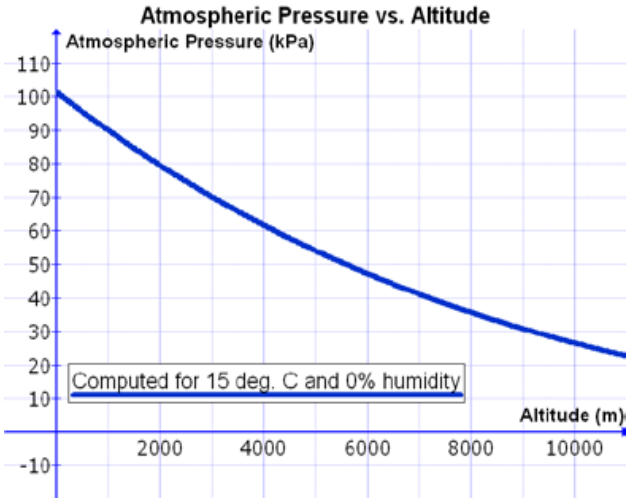
Η συνδεσμολογία μεταξύ του αισθητήρα και της πλακέτας Arduino είναι αρκετά απλή. Στον ακροδέκτη #1 του αισθητήρα συνδέεται η γείωση (ground), στον ακροδέκτη #4 η τάση (+5v ή 3.3v) και στον ακροδέκτη #2 μία από τις ψηφιακές εισόδους του Arduino, στην εργασία αυτή έχει συνδεθεί στην θύρα #5. Στο σχήμα βλέπουμε τον τρόπο συνδεσμολογίας του αισθητήρα με την πλακέτα.



Εικόνα 16: Υπόδειγμα συνδεσμολογίας του αισθητήρα με το Arduino

3.3.2 Αισθητήρας Ατμοσφαιρικής / Βαρομετρικής Πίεσης BMP - 180

Ο αισθητήρας BMP - 180 έχει σχεδιαστεί για την μέτρηση ειδικά της ατμοσφαιρικής πίεσης αλλά και της θερμοκρασίας της ατμόσφαιρας. Επίσης λόγω της συσχέτισης της ατμοσφαιρικής πίεσης και του ύψους στην οποία γίνεται η μέτρηση, η βιβλιοθήκη παρέχει και την επιπλέον λειτουργία, της ένδειξης του τρέχοντος υψομέτρου της περιοχής του αισθητήρα. Η γραφική παράσταση της συνάρτησης μεταξύ του ύψους και της ατμοσφαιρικής πίεσης εμφανίζεται στο σχήμα.



Εικόνα 17: Γραφική παράσταση σχέσης ύψους και ατμοσφαιρικής πίεσης

Οι τεχνικές προδιαγραφές του αισθητήρα είναι :

• Εύρος ευαισθησίας αισθητήρα: 300-1100 HPa (-500m έως 9000m υψόμετρο)

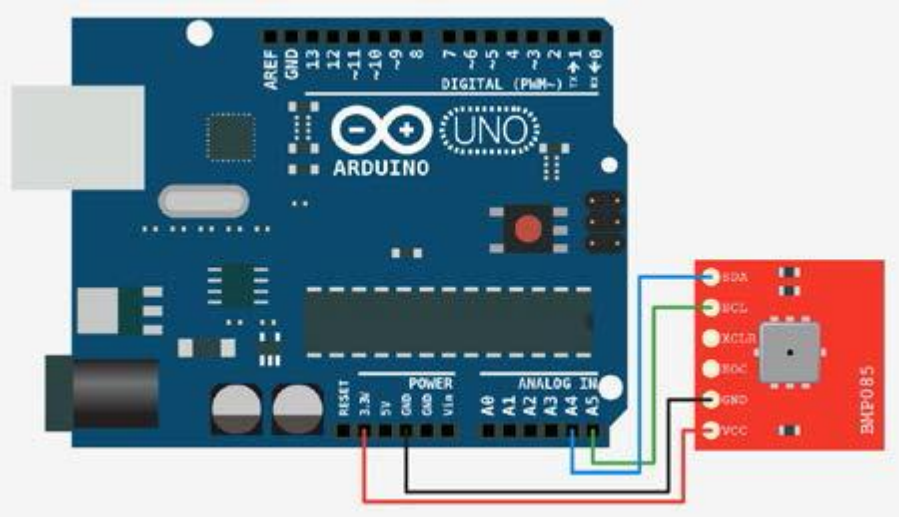
• Ευαισθησία: 0.03 HPa / 0.25m

• Θερμοκρασία περιβάλλοντος: -40 C έως +85 C

• Τροφοδοσία: 3,3 V – 5V

Σύμφωνα με τον WMO οι απαιτήσεις μέτρησης της ατμοσφαιρικής πίεσης είναι από 500 έως 1080 HPa και με ευαισθησία 0.1 HPa, το οποίο καθιστά τον αισθητήρα ιδανικό για ακριβής μετρήσεις, σε σχέση με το κόστος του.

Συνδεσμολογία με την πλακέτα Arduino



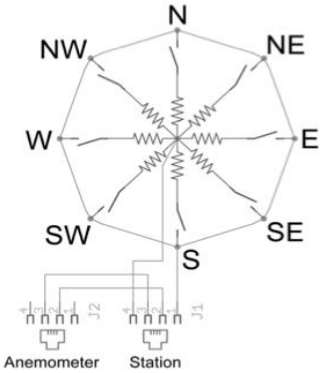
Εικόνα 18: Συνδεσμολογία του αισθητήρα BMP - 180 με την πλακέτα Arduino

Στο σχήμα βλέπουμε την συνδεσμολογία του αισθητήρα με την πλακέτα. στον ακροδέκτη VCC συνδέεται η τάση 3.3V, στο GRD η γείωση. Ο ακροδέκτης SCL συνδέεται με το ρολόι του Arduino για τον συγχρονισμό του αισθητήρα. Για το Arduino UNO αυτό βρίσκεται στην αναλογική είσοδο #5. Ο ακροδέκτης SDA συνδέεται με το πρωτόκολλο επικοινωνίας i2c και είναι υπεύθυνο για την ανάγνωση των δεδομένων του αισθητήρα.

3.3.3 Αισθητήρας Κατεύθυνσης Ανέμου 80422

Πρόκειται για έναν σχετικά οικονομικό αισθητήρα, ο οποίος βασίζεται στην εξής λειτουργία: υπάρχει ένα δίκτυο από 8 διαφορετικές αντιστάσεις, μια για κάθε κατεύθυνση του ανέμου. Όπως φαίνεται στο σχήμα όταν ο δείκτης στραφεί προς μια κατεύθυνση τότε γίνεται η σύνδεση μεταξύ των 2 αντιστάσεων. Στην έξοδο (ακροδέκτης 4) του αισθητήρα διέρχεται ένα μοναδικό αναλογικό σήμα, το οποίο ανταποκρίνεται σε μια κατεύθυνση.

Εικόνα 19: Αισθητήρας κατεύθυνσης ανέμου 80422



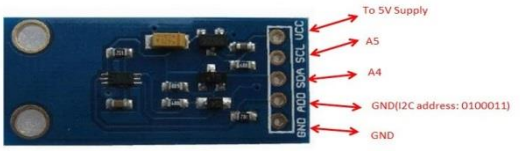
Εικόνα 20: Συνδεσμολογία αντιστάσεων - Εντοπισμός της κατεύθυνσης του ανέμου

Direction (Degrees)	Resistance (Ohms)	Voltage (V=5v, R=10k)
0	33k	3.84v
22.5	6.57k	1.98v
45	8.2k	2.25v
67.5	891	0.41v
90	1k	0.45v
112.5	688	0.32v
135	2.2k	0.90v
157.5	1.41k	0.62v
180	3.9k	1.40v
202.5	3.14k	1.19v
225	16k	3.08v
247.5	14.12k	2.93v
270	120k	4.62v
292.5	42.12k	4.04v
315	64.9k	4.78v
337.5	21.88k	3.43v

Εικόνα 21: Αντιστοιχία μεταξύ μοιρών και τάσης εξόδου του αισθητήρα

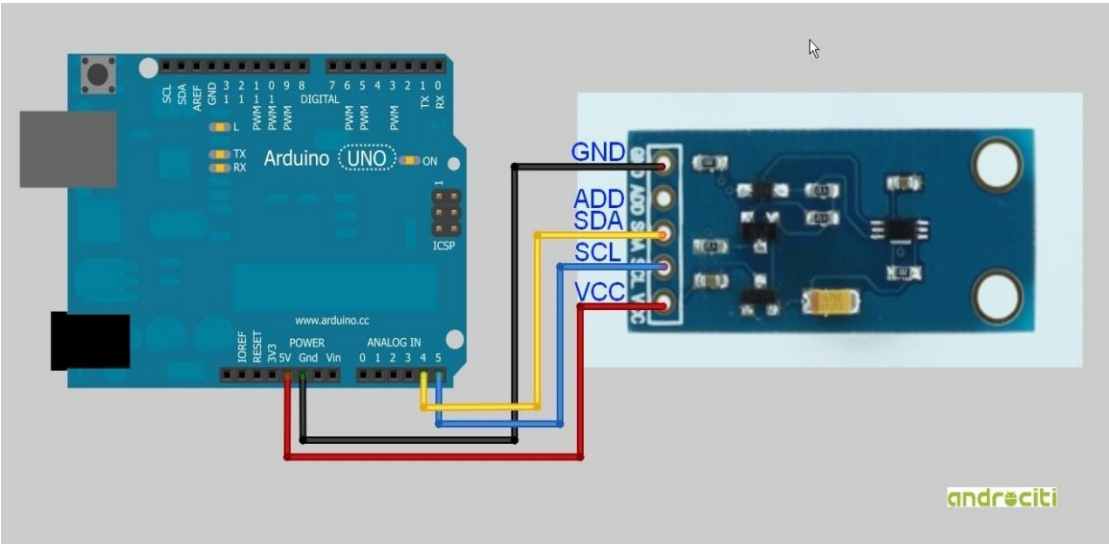
3.3.4 Ψηφιακός Μετρητής Φωτός BH - 1750FVI

Ο συγκεκριμένος αισθητήρας μετρά την ένταση φωτός του περιβάλλοντος με απλοποιημένη διαδικασία. Έχει μία αναλογική έξοδο δεδομένων και στέλνει ανά 2 sec τη μέτρηση που έχει ληφθεί. Ο τρόπος λειτουργίας του καθορίζεται από το κύκλωμα του φωτομετρητή την αντίσταση καθώς και τη δίοδο που χρησιμοποιείται. Γενικότερα όσο πιο έντονο είναι το φως της ημέρας όταν ο ήλιος είναι στο υψηλότερο σημείο τόσο μεγαλύτερη θα είναι και η τιμή που θα σταλεί. Άρα έχουμε αύξηση έντασης ρεύματος και συνεπώς μεγαλύτερη τάση στην έξοδο.



Εικόνα 22: Αισθητήρας μέτρησης φωτός BH - 1750FVI

Συνδεσμολογία με την πλακέτα Arduino



Εικόνα 23: Συνδεσμολογία BH - 1750FVI με την πλακέτα Arduino

3.4 Περιφερειακές Συσκευές

Οι πρόσθετες περιφερειακές συσκευές που είναι απαραίτητες για την επιτυχή λειτουργία και ολοκλήρωση του σταθμού μας είναι:

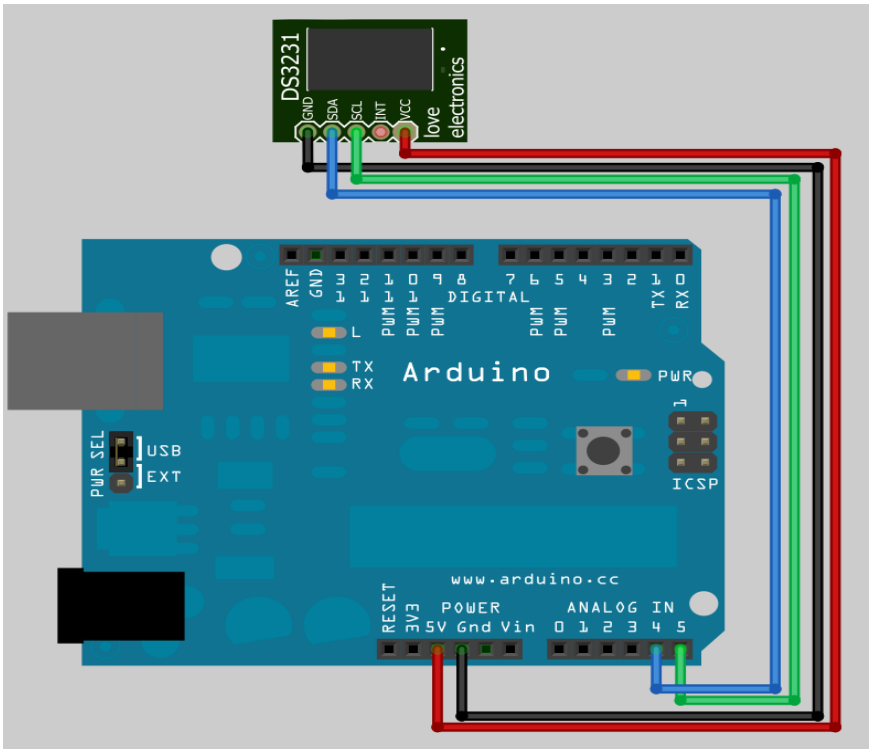
- Ρολόι Χρονισμού RTC (Real Time Clock) DS – 3231
- Κάρτα Επέκτασης Δικτύου (Arduino Ethernet Shield) W5100

3.4.1 Ρολόι Χρονισμού RTC DS - 3231

Το ρολόι χρονισμού RTC (Real Time Clock) DS – 3231 είναι μία συσκευή μέτρησης πραγματικού χρόνου με εξαιρετική ακρίβεια. Η συσκευή ενσωματώνει μία είσοδο μπαταρίας διατηρώντας την ακριβή χρονομέτρηση όταν η κύρια τροφοδοσία έχει διακοπεί. Ο τύπος της μπαταρίας που χρησιμοποιείται είναι CR – 2032 τε τάση λειτουργίας τα 3 V.

Εικόνα 24: Ρολόι Χρονισμού RTC DS - 3231

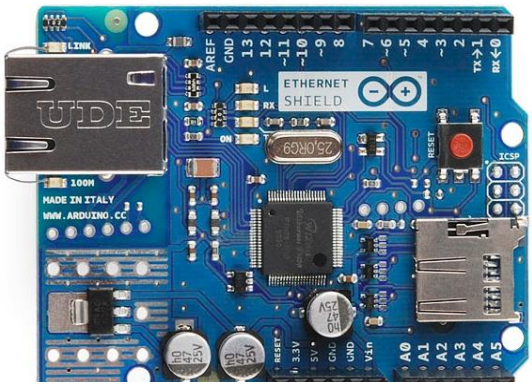
Συνδεσμολογία με την πλακέτα Arduino



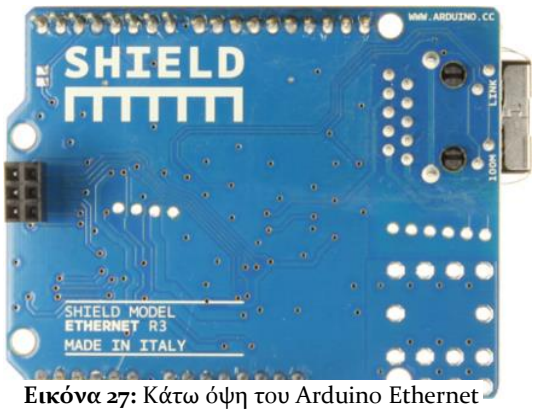
Εικόνα 25: Συνδεσμολογία Ρολογιού RTC DS - 3231 με την πλακέτα Arduino

3.4.2 Κάρτα Επέκτασης Δικτύου (Arduino Ethernet Shield) W5100

Το Arduino Ethernet Shield είναι μια πλακέτα που για να λειτουργήσει εφάπτεται πάνω στο μικροελεγκτή Arduino Mega 2560 καθώς και σε άλλες εκδόσεις του Arduino. Χρησιμοποιείται για συνδεσμολογία του Arduino με το διαδίκτυο, καθώς και για αποθήκευση δεδομένων στην SD κάρτα που διαθέτει. Διαθέτει θύρα LAN και κουμπί Reset το οποίο βοηθά στην επανεκκίνηση του Arduino και του όλου συστήματος αφού καθώς εφάπτεται με το Arduino Mega 2560 ο χρήστης είναι αδύνατο να επανεκκινήσει το σύστημα χειροκίνητα αφού το κουμπί του Arduino βρίσκεται μεταξύ των δύο πλακετών.

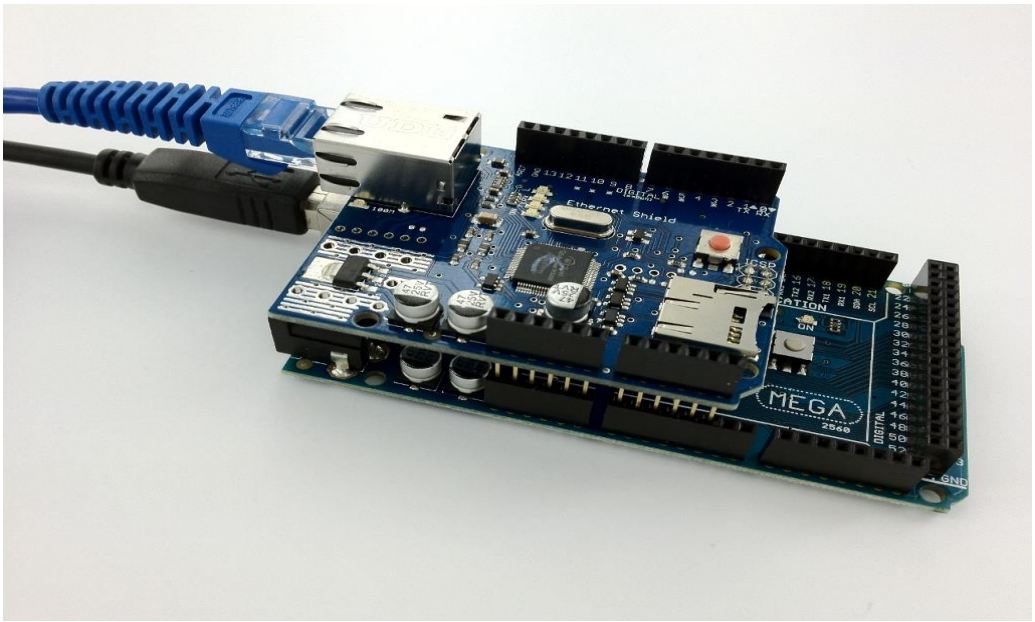


Εικόνα 26: Άνω όψη του Arduino Ethernet Shield w5100



Εικόνα 27: Κάτω όψη του Arduino Ethernet Shield w5100

Συνδεσμολογία με την πλακέτα Arduino



Εικόνα 28: Συνδεσμολογία Ethernet Shield με το Arduino Mega 2560

4

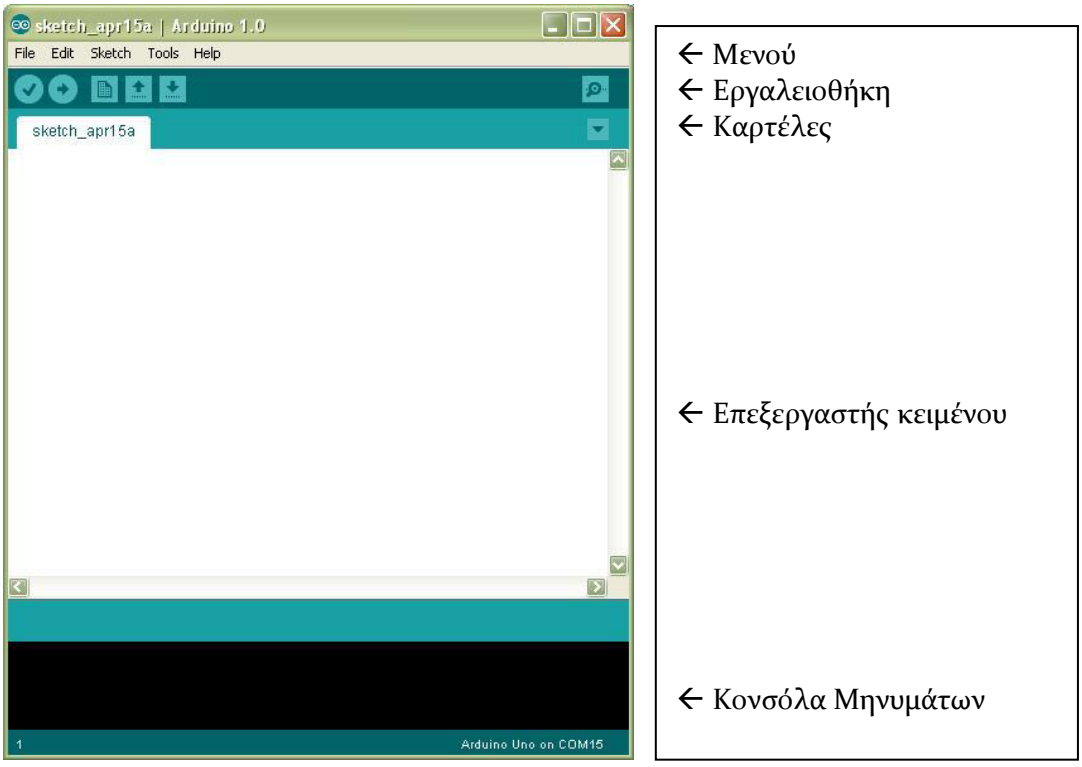
Κεφάλαιο 4ο

Προγραμματισμός Arduino

4. Κεφάλαιο 4^{ο} – Προγραμματισμός Arduino

4.1 Ολοκληρωμένο Περιβάλλον Ανάπτυξης του Arduino

Το περιβάλλον ανάπτυξης Arduino περιέχει μια περιοχή επεξεργασίας κειμένου για τη συγγραφή κώδικα, μια περιοχή μηνυμάτων, ένα μενού, μια γραμμή εργαλείων με κουμπιά για κοινές λειτουργίες, καθώς και μια σειρά από μενού. Συνδέεται με το υλικό Arduino για τη φόρτωση προγραμμάτων και για να επικοινωνούν μεταξύ τους.
Ένα ολοκληρωμένο πρόγραμμα συνήθως ονομάζεται sketch. Αυτό το sketch είναι γραμμένο με το πρόγραμμα επεξεργασίας κειμένου. Έχει δυνατότητες για την αντιγραφή / επικόλληση και για την αναζήτηση/αντικατάσταση κειμένου. Η κονσόλα απεικονίζει την έξοδο του κειμένου από το περιβάλλον Arduino συμπεριλαμβάνοντας πλήρη μηνύματα λάθους και άλλες πληροφορίες. Τα κουμπιά της γραμμής εργαλείων επιτρέπουν τον έλεγχο και το ανέβασμα των προγραμμάτων, τη δημιουργία νέου sketch, το άνοιγμα και την αποθήκευση των sketch και άνοιγμα της σειριακής οθόνης.



Εικόνα 29: Ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino

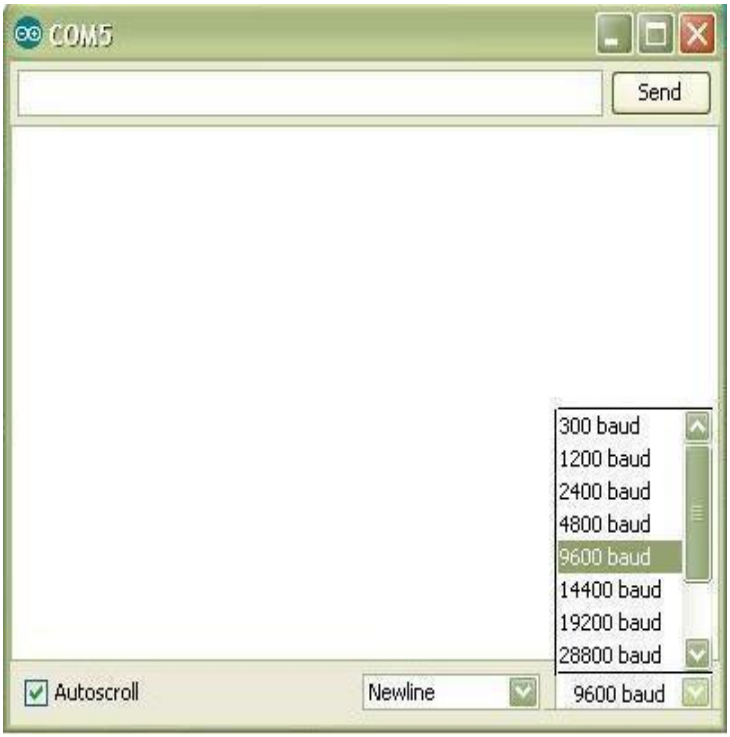
Τα κουμπιά της γραμμής εργαλείων:

	Verify/Compile (Έλεγχος / Μεταγλώττιση): Έλεγχος για λάθη στον κώδικα
	Upload: Ανέβασμα του κώδικα στον μικροελεγκτή
	New(Νέο): Δημιουργεί ένα νέο sketch
	Open(Άνοιγμα): Παρουσιάζει ένα μενού με όλα τα sketch, κάνοντας κλικ σε ένα από αυτά θα ανοίξει μέσα στο τρέχον παράθυρο
	Save(Αποθήκευση): Αποθηκεύει το sketch
	Serial Monitor(Σειριακή οθόνη): Ανοίγει την σειριακή οθόνη ώστε να μπορούμε να δώσουμε δεδομένα από το πληκτρολόγιο

Πίνακας 4: Επεξήγηση κουμπιών της γραμμής εργαλείων

4.2 Σειριακή Οθόνη (Serial Monitor)

Εμφανίζει τα σειριακά δεδομένα που αποστέλλονται από την πλακέτα Arduino. Πιο συγκεκριμένα, η αποστολή δεδομένων στην πλακέτα γίνεται, εισάγοντας κείμενο και πατώντας το κουμπί send ή πατώντας το Enter. Επίσης, στο κάτω μέρος της σειριακής οθόνης, μπορεί να γίνει η επιλογή της κατάλληλης ταχύτητας (baud) από την λίστα που εμφανίζεται ανάλογα με την τιμή που θα επιλεχθεί στο προγραμματισμό του Arduino με το Serial.begin().



Εικόνα 30: Serial Monitor

4.3 Η Δομή του προγράμματος

Ένα τυπικό πρόγραμμα Arduino έχει την παρακάτω δομή:

```
//δήλωση μεταβλητών
void setup ()
{
//αρχικοποιήσεις
}
void loop ()
{
// Κώδικας
}
```

Υπάρχουν δυο ειδικές συναρτήσεις που είναι μέρος του κάθε sketch του Arduino οι οποίες είναι η setup() και η loop(). Η setup() καλείται μια φορά, όταν το sketch ξεκινά ή όποτε κάνει επαναφορά (reset) η πλατφόρμα Arduino. Κυρίως, σε αυτήν γίνονται οι αρχικοποιήσεις των μεταβλητών, η ρύθμιση της κατάστασης των ακίδων (pins) και η προετοιμασία των βιβλιοθηκών. Αντιθέτως, η συνάρτηση loop() καλείται ξανά και ξανά επιτρέποντας έτσι στο πρόγραμμα να ανταποκριθεί σε εξωτερικά ερεθίσματα. Και οι δυο συναρτήσεις πρέπει να περιλαμβάνονται στο sketch, ακόμα και αν δεν περιέχουν κάτι και να είναι κενές.

4.4 Ψηφιακές Ακίδες (Digital Pins)

Οι ακίδες αυτές στο Arduino μπορούν να ρυθμιστούν είτε ως είσοδοι είτε ως έξοδοι, όμως από προεπιλογή είναι ρυθμισμένες ως είσοδοι. Επίσης αξίζει να σημειωθεί, ότι η πλειοψηφία των αναλογικών ακίδων του Arduino (Atmega), μπορεί να ρυθμιστεί και να χρησιμοποιηθεί, με τον ίδιο ακριβώς τρόπο όπως οι ψηφιακές ακίδες. Οι συναρτήσεις ψηφιακής εισόδου και εξόδου είναι οι παρακάτω:

- pinMode(): Ρυθμίζει τη συγκεκριμένη ακίδα να συμπεριφέρεται ως είσοδος/ έξοδος.

  Σύνταξη: pinMode(pin, mode)
  Παράμετροι:
  pin: Ο αριθμός της ακίδας της οποίας η λειτουργία είναι επιθυμητό να αλλάξει.
  mode: INPUT/OUTPUT

- digitalWrite(): Γράφει μια υψηλή (HIGH) ή μια χαμηλή (LOW) τιμή σε μια ψηφιακή ακίδα. Αν η ακίδα έχει ρυθμιστεί ως έξοδος με την συνάρτηση pinMode(), τότε η τάση της θα καθορίσει στην αντίστοιχη τιμή: 5V για HIGH και 0V για LOW. Αν η ακίδα έχει ρυθμιστεί ως είσοδος, γράφοντας HIGH στην συνάρτηση digitalWrite() θα ενεργοποιήσει μια εσωτερική pullup-αντίσταση των 20 K ενώ γράφοντας LOW θα την απενεργοποιήσει.

  Σύνταξη: digitalWrite(pin,value)
  Παράμετροι:
  pin: Ο αριθμός της ακίδας της οποίας η λειτουργία είναι επιθυμητό να αλλάξει.
  Value: INPUT/OUTPUT

- digitalWrite(): Γράφει μια υψηλή (HIGH) ή μια χαμηλή (LOW) τιμή σε μια ψηφιακή ακίδα. Αν η ακίδα έχει ρυθμιστεί ως έξοδος με την συνάρτηση pinMode(), τότε η τάση της θα καθορίσει στην αντίστοιχη τιμή: 5V για HIGH και 0V για LOW. Αν η ακίδα έχει ρυθμιστεί ως είσοδος, γράφοντας HIGH στην συνάρτηση digitalWrite() θα ενεργοποιήσει μια εσωτερική pullup-αντίσταση των 20 K ενώ γράφοντας LOW θα την απενεργοποιήσει.

  Σύνταξη: digitalWrite(pin,value)
  Παράμετροι:
  pin: Ο αριθμός της ακίδας της οποίας η λειτουργία είναι επιθυμητό να αλλάξει.
  Value: INPUT/OUTPUT

- digitalRead(): Διαβάζει την τιμή από μια συγκεκριμένη ψηφιακή ακίδα, που είναι είτε HIGH είτε LOW.

  Σύνταξη: digitalRead(pin)
  Παράμετροι:
  pin: Ο αριθμός της ακίδας της οποίας η λειτουργία είναι επιθυμητό να αλλάξει.
  Επιστρέφει: HIGH/LOW

4.5 Αναλογικές Ακίδες Εισόδου (Analog Input Pins)

Οι ελεγκτές Atmega που χρησιμοποιούνται για την πλατφόρμα Arduino περιέχουν έναν ενσωματωμένο αναλογικό-σε-ψηφιακό μετατροπέα 6 καναλιών. Ο μετατροπέας διαθέτει ανάλυση 10 bit, επιστρέφοντας ακέραιους από 0 έως 1023. Ενώ η κύρια λειτουργία της αναλογικής ακίδας για τους περισσότερους χρήστες Arduino είναι να διαβάζει αναλογικούς αισθητήρες, οι αναλογικές ακίδες έχουν επίσης όλες τις λειτουργίες των γενικών ακίδων εισόδου/εξόδου. Οι συναρτήσεις αναλογικής εισόδου και εξόδου είναι οι παρακάτω:

- analogWrite(): Γράφει μια αναλογική τιμή (PWM κύμα) σε μια ακίδα. Μπορεί να χρησιμοποιηθεί για παράδειγμα να ανάψει ένα LED σε διάφορες φωτεινότητες ή να οδηγήσει ένα κινητήρα σε διάφορες ταχύτητες. Μετά από μια κλήση της analogWrite(), η ακίδα θα δημιουργήσει ένα σταθερό τετραγωνικό κύμα του καθορισμένου κύκλου λειτουργίας μέχρι την επόμενη κλήση της analogWrite() (ή μια κλήση της digitalWrite() ή digitalRead() για την ίδια ακίδα). Η συχνότητα του σήματος PWM είναι περίπου 490 Hz. Στις περισσότερες πλατφόρμες Arduino η συνάρτηση αυτή λειτουργεί στις ακίδες 3, 5, 6, 9, 10, 11.

  Σύνταξη: analogWrite(pin, value)
  Παράμετροι:
  pin: Ο αριθμός της ακίδας της οποίας θα γράψει επάνω
  value: ο κύκλος λειτουργίας μεταξύ 0 και 255

- analogRead(): Διαβάζει την τιμή από την καθορισμένη αναλογική ακίδα.
  Σύνταξη: analogRead(pin)
  Παράμετροι:
  pin: Ο αριθμός της αναλογικής ακίδας εισόδου από όπου θα διαβάζει
  Επιστέφει: ακέραιο από 0 έως 1023

5

Κεφάλαιο 5ο

Ανάλυση και σχεδίαση του σταθμού

5. Κεφάλαιο 5ο – Ανάλυση και σχεδίαση του σταθμού

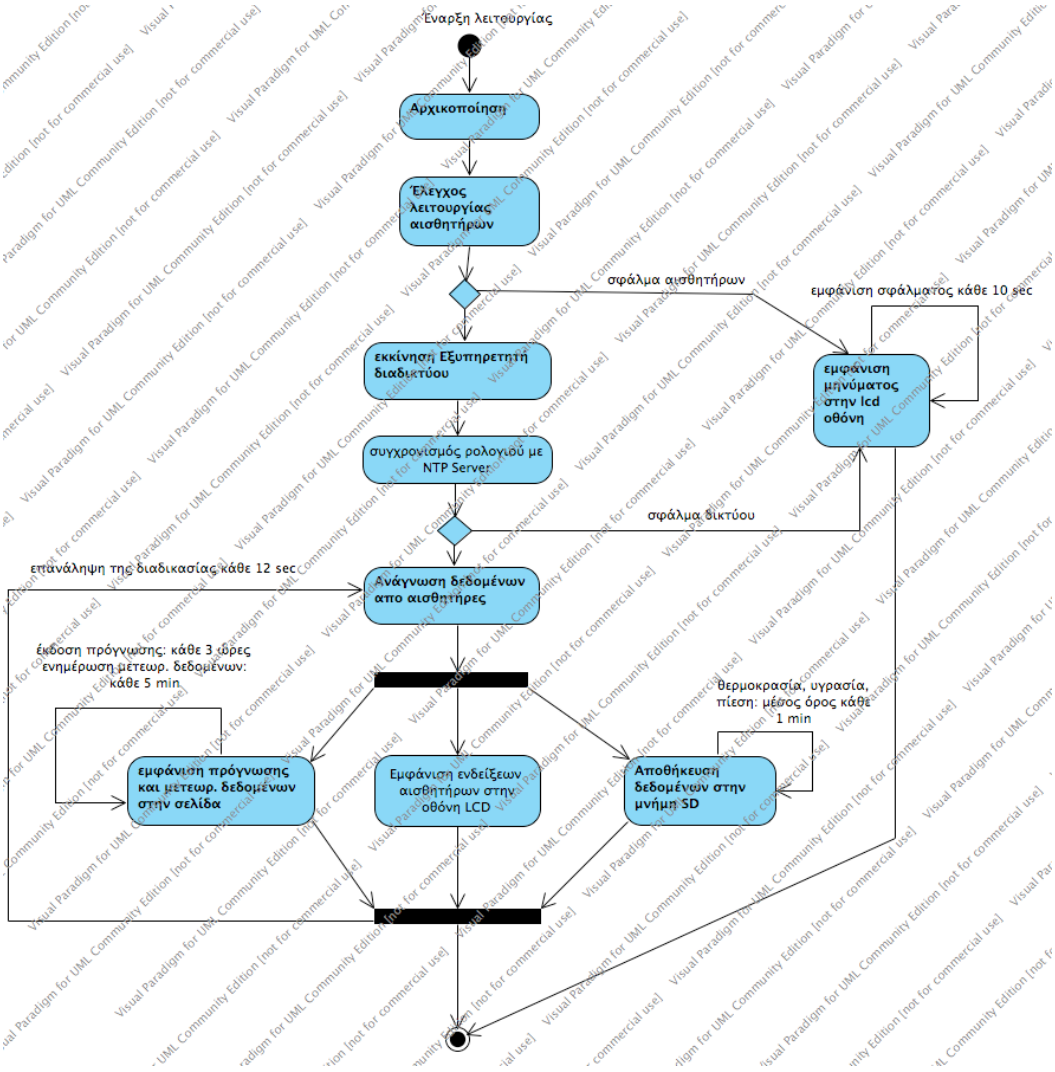
5.1 Σχεδιασμός Λογισμικού Σταθμού

Ο σταθμός κατά την εκκίνηση του θα ελέγχει ότι οι αισθητήρες είναι συνδεμένοι και ότι η επικοινωνία μεταξύ τους θα είναι ορθή. Στην συνέχεια θα γίνεται σύνδεση με το δίκτυο, φορτώνοντας τις αντίστοιχες βιβλιοθήκες .Ο σταθμός θα πρέπει να είναι όσο το δυνατόν ακριβής με τις μετρήσεις του, κατά συνέπεια κρίνεται σκόπιμο στην επόμενη φάση να γίνεται σύνδεση με εξυπηρετητή NTP από τον οποίο θα συγχρονίζεται και θα λαμβάνει την τρέχουσα ώρα. Σε κάθε περίπτωση σφάλματος θα πρέπει να ενημερώνεται ο χρήστης στην ιστοσελίδα του σταθμού καθώς και στην οθόνη LCD.

Ο σταθμός πλέον βρίσκεται σε πλήρη λειτουργία και εκτελεί τις ακόλουθες ενέργειες οι οποίες είναι κοινές για όλους τους αισθητήρες:

1. Κάθε 12 δευτερόλεπτα και συγκεκριμένα στα 12΄΄, 24΄΄, 36΄΄, 48΄΄ και 00΄΄ του επομένου λεπτού γίνεται σύνδεση με τους αισθητήρες από όπου αντλεί τα δεδομένα και τα αποθηκεύει στην προσωρινή μνήμη.
2. Μόλις φτάσει στο δευτερόλεπτο 00΄΄ (του επόμενου λεπτού) έχουν ήδη συλλεχθεί 5 δείγματα, οπότε υπολογίζουμε τον μέσο όρο αυτών, το αποτέλεσμα του οποίου αποτελεί την τιμή του μετεωρολογικού φαινομένου για το προηγούμενο λεπτό. Στην συνέχεια αποθηκεύεται η τιμή αυτή στην προσωρινή μνήμη του σταθμού καθώς και στην κάρτα SD.
3. Κάθε ώρα και συγκεκριμένα από το 50΄ έως 58΄ λεπτό της ώρας, καθορίζεται ο χρόνος στον οποίο θα γίνει ο υπολογισμός της μέσης ωριαίας τιμής με βάση τις τελευταίες 60 μετρήσεις (μία για κάθε λεπτό).
4. Με βάση το γ) γίνεται ο υπολογισμός της βαρομετρικής τάσης.
5. Κάθε πέντε δευτερόλεπτα η ιστοσελίδα του ΑΜΣ ανανεώνεται, ενημερώνοντας έτσι τον χρήστη για τις τιμές των μετεωρολογικών φαινομένων που έχουν καταγραφεί.

Από τα παραπάνω προκύπτει το διάγραμμα δραστηριότητας του σταθμού το οποίο θα είναι και ο οδηγός μας για την κατασκευή της συνάρτησης **loop()**.

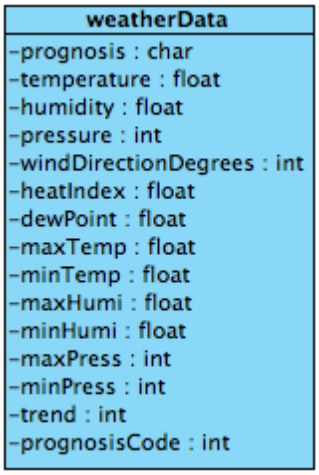


Εικόνα 31: Διάγραμμα δραστηριότητας του Αυτόνομου Μετεωρολογικού Σταθμού

5.2 Δομή Weather Data Σταθμού

Η δομή WeatherData αποτελεί τον ενδιάμεσο κρίκο μεταξύ της βιβλιοθήκης WeatherStation και της συνάρτησης **loop**(). Η δομή περιέχει συγκεντρωτικά όλες τις πληροφορίες που απαιτούνται για την διαχείριση των μετεωρολογικών δεδομένων από την εφαρμογή.

Το διάγραμμα της δομής παρουσιάζεται στο σχήμα:



Εικόνα 32: Διάγραμμα δομής Weather Data

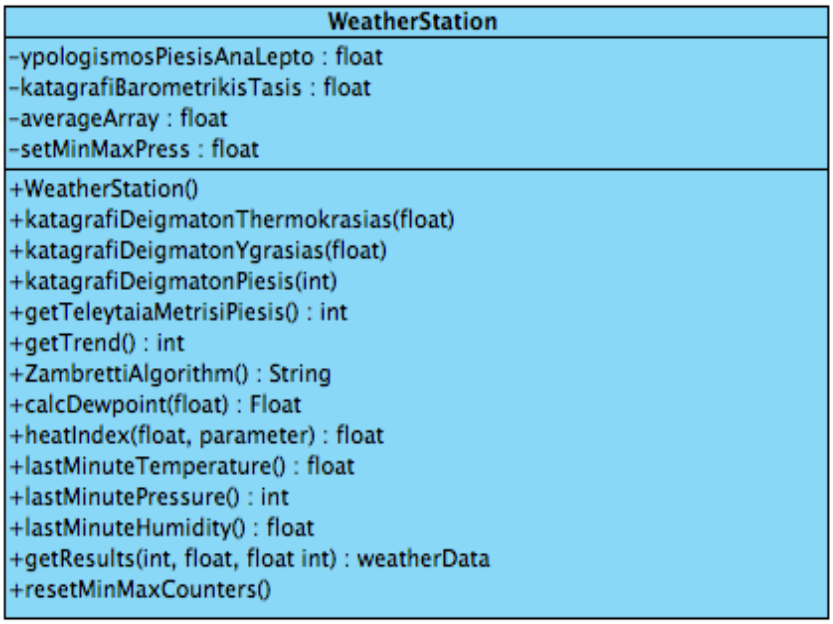
Αναλυτικά τα στοιχεία της δομής είναι:

prognosis	char	Αποθηκεύεται το κείμενο της πρόγνωσης καιρού
temperature	Float	Αποθηκεύεται η τιμή της θερμοκρασίας
himidity	Float	Αποθηκεύεται η τιμή της υγρασίας
pressure	Int	Αποθηκεύεται η τιμή της ατμοσφαιρικής πίεσης
windDirectionDegrees	Int	Αποθηκεύεται η τιμή της διεύθυνσης ανέμου σε μοίρες
heatIndex	Float	Αποθηκεύεται η τιμή του δείκτη δυσφορίας
dewPoint	Float	Αποθηκεύεται η τιμή του σημείου δρόσου
maxTemp	Float	Αποθηκεύεται η τιμή της μέγιστης θερμοκρασίας
minTemp	Float	Αποθηκεύεται η τιμή της ελάχιστης θερμοκρασίας
maxHumi	Float	Αποθηκεύεται η τιμή της μέγιστης υγρασίας
minHumi	Float	Αποθηκεύεται η τιμή της ελάχιστης υγρασίας
maxPres	Float	Αποθηκεύεται η ημερήσια μέγιστη τιμή της ατμ. πίεσης
minPres	Float	Αποθηκεύεται η ημερήσια ελάχιστη τιμή της ατμ. πίεσης
trend	int	Αποθηκεύεται η βαρομετρική τάση με την εξής κωδικοποίηση. ✓ Είναι σταθερή ✓ 1 είναι αυξητική ✓ 2 είναι μειούμενη ✓ -1 όταν δεν μπορεί να υπολογιστεί η βαρομετρική τάση
prognosisCode	int	Αποθηκεύεται ο κωδικός της πρόγνωσης καιρού.

Πίνακας 5: Στοιχεία Δομής Weather Data

5.3 Βιβλιοθήκη Weather Station Σταθμού

Η βιβλιοθήκη WeatherStation περιέχει όλες εκείνες τις συναρτήσεις, τις οποίες χρειάζεται ο σταθμός για τον υπολογισμό της μέσης τιμής υγρασίας, θερμοκρασίας, ατμοσφαιρικής πίεσης, κατεύθυνσης ανέμου (σε μοίρες). Επίσης περιέχει και την συνάρτηση υπολογισμού της βαρομετρικής τάσης. Το διάγραμμα κλάσεως της εν λόγω βιβλιοθήκης παρουσιάζεται στην εικόνα.



Εικόνα 33: Διάγραμμα κλάσης Weather Station

Η κλάση περιέχει τις ακόλουθες μεθόδους / ιδιότητες:

WeatherStation	Μέθοδος	Είναι ο κατασκευαστής της κλάσης. Δεν απαιτεί μεταβλητές.
katagrafiDeigmatonThermokrasias	Μέθοδος	Η μέθοδος αυτή δέχεται ως είσοδο την τιμή της τρέχουσας θερμοκρασίας και την τοποθετεί στην προσωρινή μνήμη για τον υπολογισμό της μέσης τιμής.
katagrafiDeigmatonYgrasias	Μέθοδος	Η μέθοδος αυτή δέχεται ως είσοδο την τιμή της τρέχουσας υγρασίας και την τοποθετεί στην προσωρινή μνήμη για τον υπολογισμό της μέσης τιμής.
katagrafiDeigmatonPiesis	Μέθοδος	Η μέθοδος αυτή δέχεται ως είσοδο την τιμή της τρέχουσας ατμοσφαιρικής πίεσης και την

		τοποθετεί στην προσωρινή μνήμη για τον υπολογισμό της μέσης τιμής.
getTrend	Μέθοδος	Όταν καλείται, επιστρέφει σε αριθμητική απεικόνιση την βαρομετρική τάση του τελευταίου τρίωρου. Όταν επιστρέφει 0 τότε δεν υπάρχει μεταβολή, όταν επιστρέφει 1 τότε υπάρχει αύξηση, ενώ όταν επιστρέφει 2 υπάρχει πτώση.
katagrafiBarometrikisTasis	Μέθοδος	Δέχεται ως είσοδο τον μέσο όρο των 60 τελευταίων μετρήσεων και τον αποθηκεύει για τον υπολογισμό της βαρομετρικής τάσης.
ZambrettiAlgorithm	Μέθοδος	Δέχεται ως ορίσματα την ελάχιστη και μέγιστη τιμή της ατμοσφαιρικής πίεσης για την περιοχή που θα εγκατασταθεί ο σταθμός, την βαρομετρική τάση (με την κωδικοποιημένη μορφή που επιστρέφει η συνάρτηση getTrend), η κατεύθυνση του ανέμου σε μοίρες, και η τρέχουσα ατμοσφαιρική πίεση.
calcDewpoint	Μέθοδος	Δέχεται ως όρισμα την τρέχουσα θερμοκρασία και υγρασία και επιστρέφει την θερμοκρασία του σημείου δρόσου.
heatIndex	Μέθοδος	Δέχεται ως όρισμα την τρέχουσα θερμοκρασία και υγρασία και επιστρέφει τον δείκτη δυσφορίας.
lastMiniuteTemperature	Μέθοδος	Επιστρέφει τον μέσο όρο των τελευταίων 5 μετρήσεων της θερμοκρασίας.
lastMiniutePressure	Μέθοδος	Επιστρέφει τον μέσο όρο των τελευταίων 5 μετρήσεων της ατμοσφαιρικής πίεσης.
lastMiniuteHumidity	Μέθοδος	Επιστρέφει τον μέσο όρο των τελευταίων 5 μετρήσεων της υγρασίας.
getResult	Μέθοδος	Δέχεται ως ορίσματα την τρέχουσα θερμοκρασία, υγρασία, ατμ. πίεση, διεύθυνση ανέμου και επιστρέφει τη δομή weatherData συμπληρωμένη.
resetMinMaxCounters	Μέθοδος	Καθημερινά στις 00:00 διαγράφει τα δεδομένα των προσωρινών μνημών, για τον υπολογισμό νέων.

Πίνακας 6: Μέθοδοι / Ιδιότητες Κλάσης

6

Κεφάλαιο 6ο

Σύνδεση Περιφερειακών – Αισθητήρων & Υλοποίηση

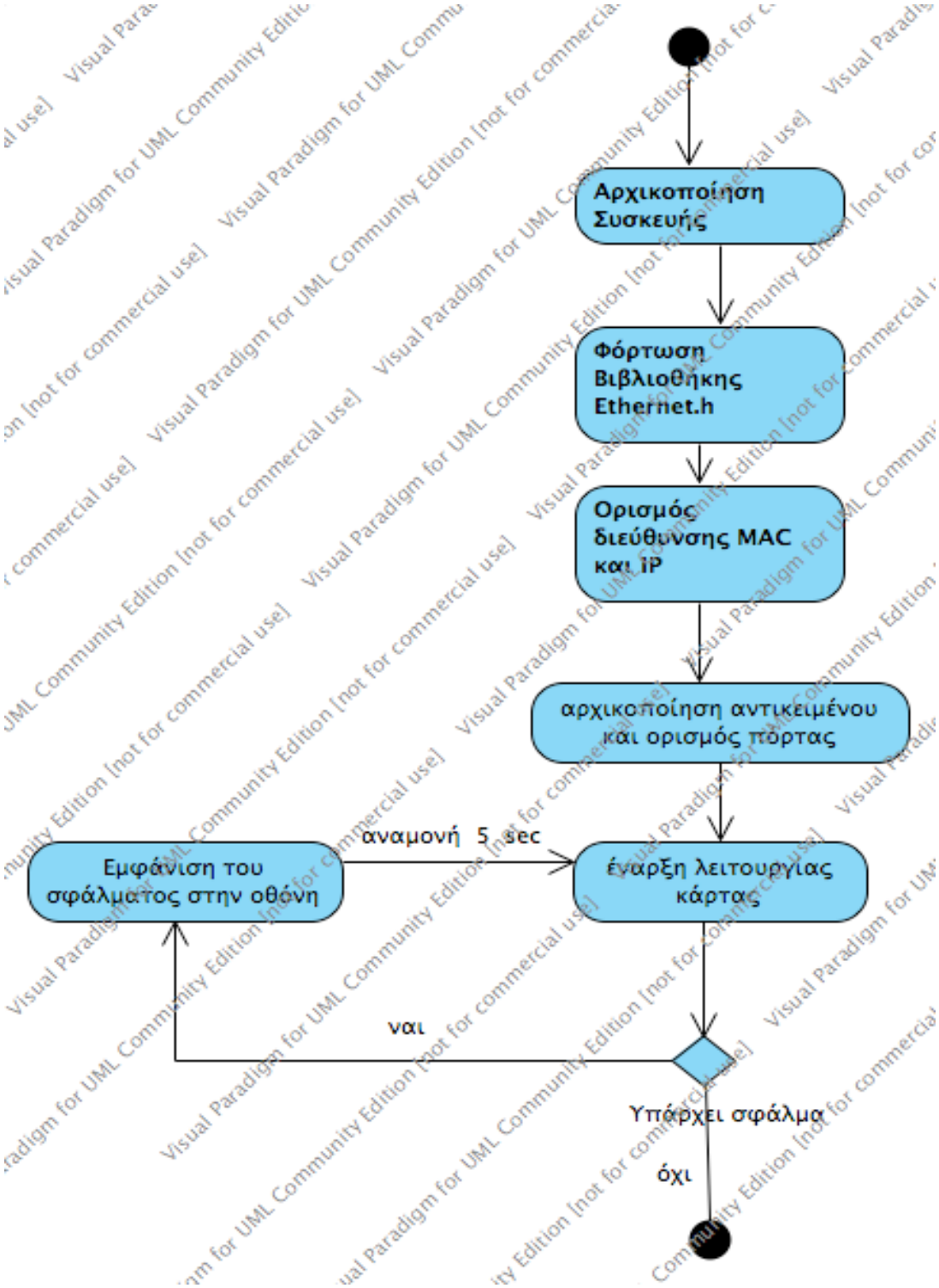
6. Κεφάλαιο 6ο – Σύνδεση Περιφερειακών – Αισθητήρων & Υλοποίηση

6.1 Κάρτα Επέκτασης Δικτύου (Ethernet Shield)

Κύρια λειτουργία της κάρτας Ethernet είναι η διασύνδεση της πλακέτας Arduino με δίκτυο με σκοπό την μετάδοση πληροφοριών. Στην παρούσα εργασία ο σταθμός έχει τοποθετηθεί σε ενσύρματο δίκτυο τύπου Ethernet (οικιακό δίκτυο), από όπου γίνεται η πρόσβαση από όλους τους σταθμούς εργασίας του τοπικού δικτύου. Χρησιμοποιείται το πρωτόκολλο TCP/IP καθώς στις απαιτήσεις της εργασίας προαπαιτούμενος τρόπος διασύνδεσης με τον σταθμό είναι μέσω φυλλομετρητή ιστοσελίδων.

Κατά την εκκίνηση του σταθμού γίνεται αρχικοποίηση των ρυθμίσεων της κάρτας δικτύου. Αρχικά εισάγουμε την βιβλιοθήκη **Ethernet.h [LIB]** η οποία βρίσκεται προεγκατεστημένη στο IDE του Arduino και περιέχει όλες τις συναρτήσεις για την διασύνδεση της κάρτας με το Arduino. Στην συνέχεια ορίζουμε την διεύθυνση IP του δικτύου, το υποδίκτυο στο οποίο ανήκει ο σταθμός, την διεύθυνση MAC της κάρτας, αρχικοποιούμε (instantiate) το αντικείμενο και ορίζουμε την πόρτα στην οποία θα ανταποκρίνεται ο εξυπηρετητής. Εξ' ορισμού η πόρτα στην οποία ανταποκρίνονται όλες οι αιτήσεις του HTTP πρωτοκόλλου είναι η 80. Στη συνάρτηση **setup()** γίνεται η αρχικοποίηση και έναρξη λειτουργίας της κάρτας. Σε περίπτωση που υπάρχει σφάλμα κατά την διαδικασία σύνδεσης της κάρτας, τότε ενημερώνεται ο χρήστης με σχετικό μήνυμα στην οθόνη LCD. Η ανωτέρω διαδικασία επαναλαμβάνεται έως ότου η επικοινωνία αποκατασταθεί.

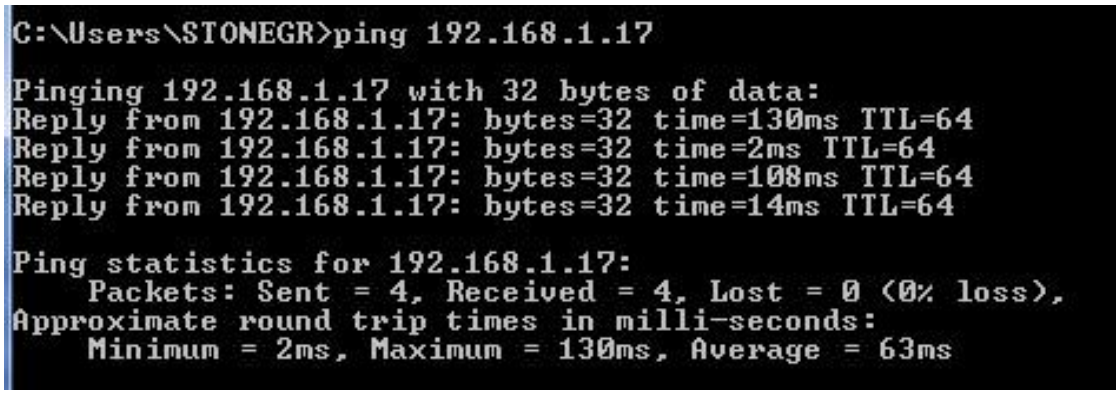
Στο σχήμα παρατίθεται το διάγραμμα ροής της λειτουργίας της κάρτας δικτύου Ethernet Shield:



Εικόνα 34: Διάγραμμα ροής λειτουργίας Ethernet Shield

Ο κώδικας με τον οποίο υλοποιείται η σύνδεση περιλαμβάνεται στο **παράρτημα Α**.

Για να ελέγξουμε την ορθή λειτουργία του σταθμού εκτελούμε την εντολή ping από την γραμμή εντολών (ισχύει σε όλα τα λειτουργικά συστήματα, η δοκιμή έγινε σε Windows 7) και παρατηρούμε ότι ο σταθμός απαντά στο ερώτημα μας με επιτυχία.



Εικόνα 35: Αποτελέσματα Δοκιμών διασύνδεσης κάρτας Ethernet Shield

6.2 Αισθητήρας Καταγραφής Θερμοκρασίας / Υγρασίας DHT – 22 / AM – 2302

Ο αισθητήρας DHT-22 είναι υπεύθυνος για την ανάγνωση της θερμοκρασίας και υγρασίας. Για την διασύνδεση του αισθητήρα με την πλακέτα Arduino γίνεται μέσω της βιβλιοθήκης DHTlib. Η σύνδεση με το Arduino γίνεται μέσω ψηφιακής εισόδου στην οποία εισέρχεται τάση η οποία έχει σταθμιστεί ανάλογα με την τιμή θερμοκρασίας / υγρασίας του περιβάλλοντος. Η τάση αυτή μετατρέπεται σε τιμή η οποία αντιστοιχεί σε αυτή της θερμοκρασίας και της υγρασίας. Η βιβλιοθήκη αναλαμβάνει να μετατρέψει την εισερχόμενη τάση στις τιμές που αντιστοιχούν. Η κλάση όταν αρχικοποιείται θα πρέπει να δηλωθεί η αναλογική είσοδος στην οποία είναι συνδεδεμένη. Η κλάση περιέχει τρεις μεθόδους:

readData(): επιστρέφει την κατάσταση του αισθητήρα. Επιστρέφονται οι τιμές:

• DHT_ERROR_NONE: δεν υπάρχουν σφάλματα

• DHT_ERROR_CHECKSUM: σφάλμα κατά τον έλεγχο με το bit ισοτιμίας

• DHT_BUS_HUNG: σφάλμα διαύλου

• DHT_ERROR_NOT_PRESENT: δεν εντοπίστηκε ο αισθητήρας

• DHT_ERROR_ACK_TOO_LONG: το σήμα ACK έληξε

• DHT_ERROR_SYNC_TIMEOUT: σφάλμα κατά τον συγχρονισμό

• DHT_ERROR_DATA_TIMEOUT: σφάλμα δεδομένων

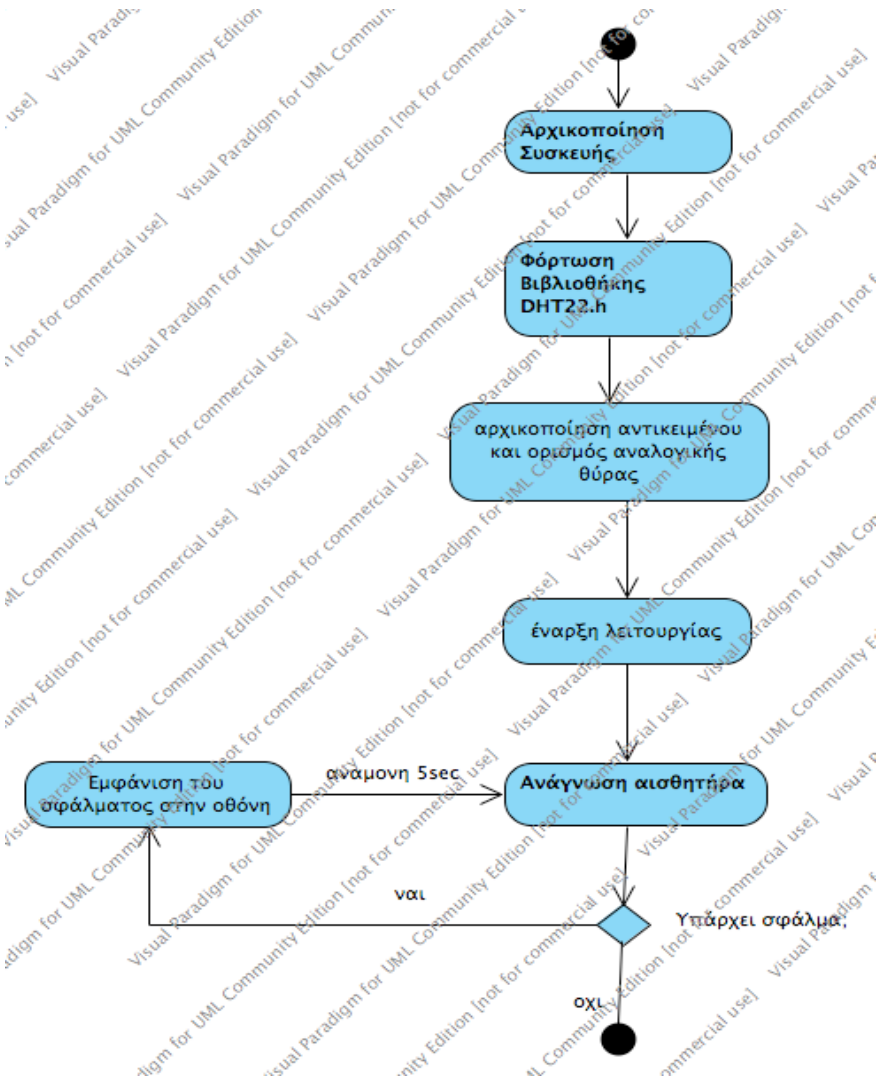
• DHT_ERROR_TOOQUICK: ζητήθηκε γρηγορότερα από τον αναμενόμενο χρόνο (<2sec) αίτημα για νέα δεδομένα

getHumidity(): επιστρέφει την τρέχουσα τιμή του αισθητήρα για την υγρασία σε

getTemperatureC() επιστρέφει την τρέχουσα τιμή της θερμοκρασίας σε βαθμούς κελσίου.

Αρχικά εισάγουμε την βιβλιοθήκη <**DHT22.h**> στο πρότυπο. Στην συνέχεια αρχικοποιούμε (instantiate) το αντικείμενο ορίζοντας την αναλογική είσοδο στην οποία θα συνδεθεί ο αισθητήρας. Τέλος, ελέγχουμε την κατάσταση του αισθητήρα έτσι ώστε σε περίπτωση λάθους να ενημερώνει τον χρήστη και γίνεται έλεγχος λειτουργίας του αισθητήρα κάθε 5 δευτερόλεπτα.

Στο ακόλουθο διάγραμμα ροής βλέπουμε την διαδικασία που θα ακολουθήσουμε για την σύνδεση και τον έλεγχο του αισθητήρα.



Εικόνα 36: Διάγραμμα ροής διασύνδεσης με τον αισθητήρα DHT – 22 / AM - 2302

Ο κώδικας με τον οποίο γίνεται η σύνδεση με την πλακέτα Arduino, παρουσιάζεται στο **παράρτημα Β**.

Για να ελέγξουμε την ορθή λειτουργία του αισθητήρα ανοίγουμε το παράθυρο σειριακής θύρας και παρατηρούμε ότι ο σταθμός αποστέλλει την τρέχουσα θερμοκρασία και υγρασία.

```
Integer-only reading: Temperature 23.7 C, Humidity 53.3 % RH
Requesting data...Got Data 23.70C 53.30%
Integer-only reading: Temperature 23.7 C, Humidity 53.3 % RH
Requesting data...Got Data 23.70C 53.40%
Integer-only reading: Temperature 23.7 C, Humidity 53.4 % RH
Requesting data...Got Data 23.70C 53.40%
Integer-only reading: Temperature 23.7 C, Humidity 53.4 % RH
Requesting data...Got Data 23.60C 53.30%
Integer-only reading: Temperature 23.6 C, Humidity 53.3 % RH
Requesting data...Got Data 23.60C 53.40%
Integer-only reading: Temperature 23.6 C, Humidity 53.4 % RH
Requesting data...Got Data 23.60C 53.40%
Integer-only reading: Temperature 23.6 C, Humidity 53.4 % RH
Requesting data...Got Data 23.60C 53.40%
Integer-only reading: Temperature 23.6 C, Humidity 53.4 % RH
```

Εικόνα 37: Αποτελέσματα ελέγχου αισθητήρα DHT – 22 / AM - 2302

6.3 Σύνδεση Αισθητήρα Ατμοσφαιρικής Πίεσης BMP – 180

Ο αισθητήρας BMP - 180 είναι υπεύθυνος για την λήψη της ατμοσφαιρικής πίεσης και της θερμοκρασίας. Για την διασύνδεση με το Arduino θα γίνει χρήση της βιβλιοθήκης Adafruit_BMP180.h την οποία έχει αναπτύξει η εταιρία Adafruit και είναι ανοιχτού κώδικα. Η κλάση περιέχει τις μεθόδους:

- **begin():** γίνεται η έναρξη της ανάγνωσης από τον αισθητήρα.

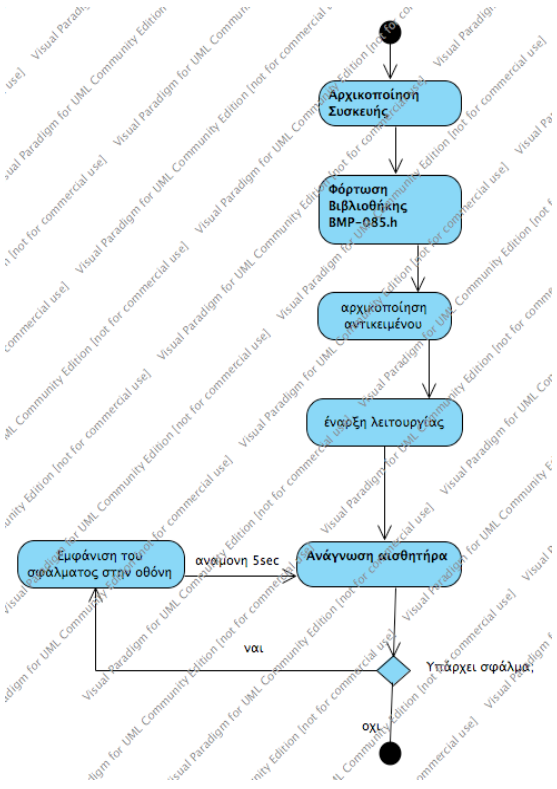
• **readTemperature**(): επιστρέφει σε float την τιμή της τρέχουσας θερμοκρασίας.

• **readPressure**(): επιστρέφει σε float την τιμή της τρέχουσας ατμοσφαιρικής πίεσης.

• **readAltitute(int)**: δέχεται ως όρισμα την τιμή της ατμοσφαιρικής πίεσης και επιστρέφει το υψόμετρο.

• **seaLevelPressure(υψος, πίεση, ατμόσφαιρα)**: δέχεται ως ορίσματα την τιμή της ατμοσφαιρικής πίεσης σε (hPa), του ύψους (σε μέτρα) και της θερμοκρασίας (σε βαθμούς Celsius) και επιστρέφει την τιμή της ατμοσφαιρικής πίεσης της επιφάνειας της θάλασσας.

Για να χρησιμοποιήσουμε την βιβλιοθήκη αρκεί η αναφορά της στην αρχικοποίηση του προγράμματος. Στην συνάρτηση **setup()** γίνεται η σύνδεση του αισθητήρα με το Arduino. Στην περίπτωση σφάλματος ενημερώνεται ο χρήστης με σχετικό μήνυμα και κάθε 5 δευτερόλεπτα γίνεται νέα προσπάθεια σύνδεσης με τον αισθητήρα έως ότου η σύνδεση ολοκληρωθεί. Όπως και στους προηγούμενους αισθητήρες έτσι και εδώ τονίζεται πως ο έλεγχος γίνεται κατά την εκκίνηση και όχι κατά την λειτουργία των αισθητήρων όπου εκεί υπάρχει άλλη διαδικασία.



Εικόνα 38: Διάγραμμα ροής αισθητήρα ατμοσφαιρικής πίεσης BMP - 180

Ο κώδικας με τον οποίο γίνεται η σύνδεση παρουσιάζεται στο **παράρτημα Γ**.

Στο σχήμα βλέπουμε την έξοδο του αισθητήρα κατά την διάρκεια των δοκιμών. Θα πρέπει να λάβουμε υπόψη ότι η έξοδος της βιβλιοθήκης επιστρέφει την τιμή της ατμοσφαιρικής πίεσης σε Pa οπότε θα πρέπει να γίνει μετατροπή της σε hPa. Η μετατροπή εκφράζεται ως:

$$hPa = Pa \times 0{,}01$$

```
Temperature = 22.70 *C
Pressure = 101365 Pa

Temperature = 22.71 *C
Pressure = 101368 Pa

Temperature = 22.73 *C
Pressure = 101369 Pa

Temperature = 22.71 *C
Pressure = 101364 Pa

Temperature = 22.69 *C
Pressure = 101371 Pa
```

Εικόνα 39: Έλεγχος λειτουργίας του αισθητήρα ατμοσφαιρικής πίεσης BMP - 180

6.4 Σύνδεση Αισθητήρα Διεύθυνσης Ανέμου 80422

Για την διασύνδεση του αισθητήρα διεύθυνσης ανέμου χρειάζεται να αναπτυχθεί η συνάρτηση ανάγνωσης των σημάτων του αισθητήρα καθώς δεν έχει υπάρχει σχετική βιβλιοθήκη. Σύμφωνα με το εγχειρίδιο του αισθητήρα στην έξοδο παρουσιάζεται μια μοναδική μεταβαλλόμενη τάση η τιμή της οποίας εξαρτάται από την κατεύθυνση του ανέμου. Έτσι για παράδειγμα όταν ο δείκτης είναι προσανατολισμένος στον Βορρά, η έξοδος είναι 3,84 V.

Για την διαχείριση των αναλογικών σημάτων, το Arduino διαθέτει ενσωματωμένο έναν μετατροπέα σήματος από αναλογικό σε ψηφιακό (Analog to Digital Converter) με ακρίβεια δέκα ψηφίων (10 bit) άρα το αναλογικό σήμα δειγματίζεται σε 210=1024 βαθμίδες.

Η συνάρτηση που θα κατασκευαστεί, θα διαβάζει το εισερχόμενο σήμα από την αναλογική είσοδο και στην συνέχεια ανάλογα με την τιμή που θα παρουσιάζει, θα επιστρέφει το ανάλογο μήνυμα. Κατά συνέπεια θα πρέπει να εντοπίσουμε τις διακριτές τιμές στις οποίες μετατρέπεται το εισερχόμενο αναλογικό σήμα.

Στον Πίνακα φαίνεται η αντιστοιχία του αναλογικού σήματος συναρτήσει της κατεύθυνσης του αισθητήρα.

Direction (Degrees)	Resistance (Ohms)	Voltage (V=5v, R=10k)
0	33k	3.84v
22.5	6.57k	1.98v
45	8.2k	2.25v
67.5	891	0.41v
90	1k	0.45v
112.5	688	0.32v
135	2.2k	0.90v
157.5	1.41k	0.62v
180	3.9k	1.40v
202.5	3.14k	1.19v
225	16k	3.08v
247.5	14.12k	2.93v
270	120k	4.62v
292.5	42.12k	4.04v
315	64.9k	4.78v
337.5	21.88k	3.43v

Πίνακας 7: Αντιστοιχίες μεταξύ της διεύθυνσης ανέμου και αναλογικού σήματος.

Στην συνέχεια από την σχέση υπολογίζουμε την τιμή την οποία επιστρέφει η συνάρτηση **analogRead()** αναλόγως του εισερχόμενου αναλογικού σήματος.

$$abc = [\left(\frac{VIN}{VREF}\right) * 1023]$$

όπου:

• adc η διακριτή τιμή του εισερχόμενου σήματος.

• VIN η τιμή του εισερχόμενου σήματος.

• VREF η μέγιστη (θεωρητική) τιμή του μετατροπέα. Στο Arduino είναι 5 (Volt).

Στο **παράρτημα Δ** παρουσιάζεται η διαδικασία διασύνδεσης του αισθητήρα με την πλακέτα Arduino. Στο Σχήμα βλέπουμε την έξοδο του Arduino κατά την διάρκεια των δοκιμών για τον έλεγχο ανάγνωσης του αισθητήρα. Επισημαίνεται ότι οι 2 τελευταίες ενδείξεις αναφέρουν σφάλμα ανάγνωσης σκοπίμως, καθώς αποσυνδέθηκε ο αισθητήρας για να ελεγχθεί την αντίδραση του συστήματος.

```
NW
W
S
SW
SE
E
NE
NE
N
NW
NW
NW
NW
sensor disconnected
sensor disconnected
```

Εικόνα 40: Έξοδος αισθητήρα κατεύθυνσης ανέμου 80422

6.5 Υλοποίηση

6.5.1 Πρόγραμμα και αξιοποιήσιμες βιβλιοθήκες

Ο συνολικός κώδικας, γράφτηκε στο IDE Arduino 1.6.4 για τον προγραμματισμό της πλακέτας Arduino. Αξιοποιήθηκαν βιβλιοθήκες που έχουν δημιουργηθεί από τους κατασκευαστές των αισθητήρων και υπάγονται στην κατηγορία του "ανοιχτού κώδικα". Για την αποθήκευση των καταγραφέντων δεδομένων χρειαζόμαστε μια κάρτα SD όπου θα αποθηκεύονται τα δεδομένα σε μορφή CSV.

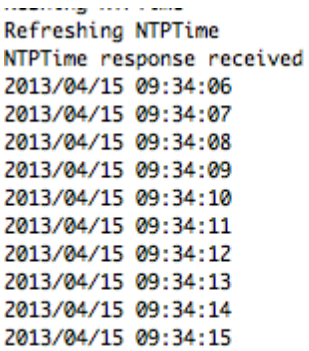
6.5.2 Ιστοσελίδες και Javascript

Οι ιστοσελίδες που δημιουργήθηκαν, υλοποιήθηκαν με το εργαλείο Netbeans 7.2. Η παρουσίαση των γραφημάτων γίνεται με την χρήση JavaScript μέσω της δωρεάν βιβλιοθήκης digraph.

6.5.3 Επιπρόσθετες βιβλιοθήκες υποστήριξης έργου

Εκτός των βιβλιοθηκών που χρειάστηκαν για την υποστήριξη των αισθητήρων αξιοποιήθηκαν και άλλες οι οποίες είχαν μεν υποστηρικτικό έργο αλλά όμως κρίθηκε απαραίτητη η εκμετάλλευση τους για την σωστή και εύρυθμη λειτουργία του σταθμού. Αυτές επιγραμματικά είναι:

ClockLibrary: Πρόκειται για μια βιβλιοθήκη η οποία είναι υπεύθυνη για τον συγχρονισμό του σταθμού με εξυπηρετητή συγχρονισμού ρολογιού (NTP Server). Η ενέργεια αυτή είναι απαραίτητη καθώς το Arduino δεν μπορεί να διατηρήσει την ημερομηνία / ώρα, με αποτέλεσμα σε κάθε επανεκκίνηση του, το ρολόι να χάνει τις ρυθμίσεις του γεγονός το οποίο επηρεάζει την ποιότητα των δεδομένων που συλλέγονται. Στο σχήμα 32 βλέπουμε την έξοδο κατά τον συγχρονισμό του Arduino με τον εξυπηρετητή πρωτοκόλλου NTP.



Εικόνα 41: Έλεγχος διασύνδεσης με τον διακομιστή NTP

LiquidDisplay: πρόκειται για την βιβλιοθήκη η οποία είναι υπεύθυνη για την προβολή δεδομένων στην οθόνη υγρών κρυστάλλων (Liquid Crystal Display – LCD) του σταθμού. Αν και τα δεδομένα παρουσιάζονται αναλυτικά στον ιστότοπο του σταθμού, κρίθηκε σκόπιμη η ενσωμάτωση της οθόνης για την άμεση προβολή πληροφοριών που αφορούν κυρίως την συντήρηση του. Τέτοιες πληροφορίες είναι η μέτρηση της τάσης του σταθμού, η IP διεύθυνση του, σε περιπτώσεις λάθους να εμφανίζει το σχετικό σφάλμα πχ: σε περίπτωση σφάλματος ενός αισθητήρα τότε εμφανίζεται, σχετικό μήνυμα στην οθόνη κλπ.

SD (Secure Disk): τα δεδομένα του σταθμού θα καταγράφονται σε αρχείο τύπου CSV έτσι ώστε να είναι αξιοποιήσιμα από εφαρμογές (όπως πχ το Microsoft Excel). Τα δεδομένα αυτά θα αποθηκεύονται σε κάρτα SD καθώς η κάρτα επέκτασης Ethernet Shield παρέχει αυτή την δυνατότητα. Αυτή η δυνατότητα κρίνεται σκόπιμο να εγκατασταθεί καθώς με αυτόν τον τρόπο αποφεύγεται η συχνή σύνδεση του

σταθμού με τον εξυπηρετητή, με αποτέλεσμα τον περιορισμό του όγκου δεδομένων που διακινούνται από και προς τον σταθμό.

6.5.4 Ανάλυση Αλγορίθμων μετεωρολογικών δεδομένων

Γνωρίζουμε ότι απαιτούνται 5 δείγματα ανά λεπτό για την συλλογή των δειγμάτων. Ο μέσος όρος των δειγμάτων αυτών αποτελεί την μέτρηση για το συγκεκριμένο λεπτό που γίνεται η καταγραφή. Το αποτέλεσμα αποθηκεύεται στην κάρτα SD καθώς και στον πίνακα για τον υπολογισμό της ωριαίας τιμής για περαιτέρω επεξεργασία. Ο κώδικας που εκτελεί την διαδικασία είναι ίδιος για όλα τα μετεωρολογικά δεδομένα (θερμοκρασία, υγρασία, ατμοσφαιρική πίεση). Στο **παράρτημα Ε** περιλαμβάνεται ο κώδικας καταγραφής της ατμοσφαιρικής πίεσης.

Για την εκτέλεση αυτής της συνάρτησης απαιτείται η πλήρη συμπλήρωση του πίνακα **mesiTimiPiesis** της προηγούμενης συνάρτησης. Για τον λόγο αυτό η συνάρτηση καλείται μόλις συμπληρωθεί και η πέμπτη θέση του πίνακα. Σύμφωνα με τον WMO ο υπολογισμός της ωριαίας μέσης τιμής θα πρέπει να γίνεται μεταξύ του 52ου έως 58ου λεπτού της ώρας. Κατά σύμβαση έχουμε ορίσει, όλες τις μετρήσεις θα γίνονται στο 55ο λεπτό της ώρας. Η μέση τιμή της θερμοκρασίας/υγρασίας και διεύθυνσης του ανέμου, δεν έχουν καμία αξία στην παρούσα εργασία, οπότε δεν κρίνεται σκόπιμη η υλοποίηση τους. Για τον λόγο αυτό αναπτύχθηκε μόνο η μέτρηση της ατμοσφαιρικής πίεσης καθώς είναι απαραίτητη για τον υπολογισμό της βαρομετρικής τάσης. Στο **παράρτημα ΣΤ** παρατίθεται ο κώδικας με τον οποίο υπολογίζεται η βαρομετρική τάση.

Οι μετρήσεις της ατμοσφαιρικής πίεσης πρέπει να είναι όσο το δυνατόν ακριβέστερες καθώς η πρόγνωση του καιρού είναι άρρηκτα συνδεδεμένη μαζί τους. Για τον λόγο αυτό θα πρέπει να γίνει η απαραίτητη αναγωγή στην επιφάνεια της θάλασσας. Ο συνάρτηση είναι απλή εφαρμογή της σχέσης και περιγράφεται στο **παράρτημα Ζ.**

7

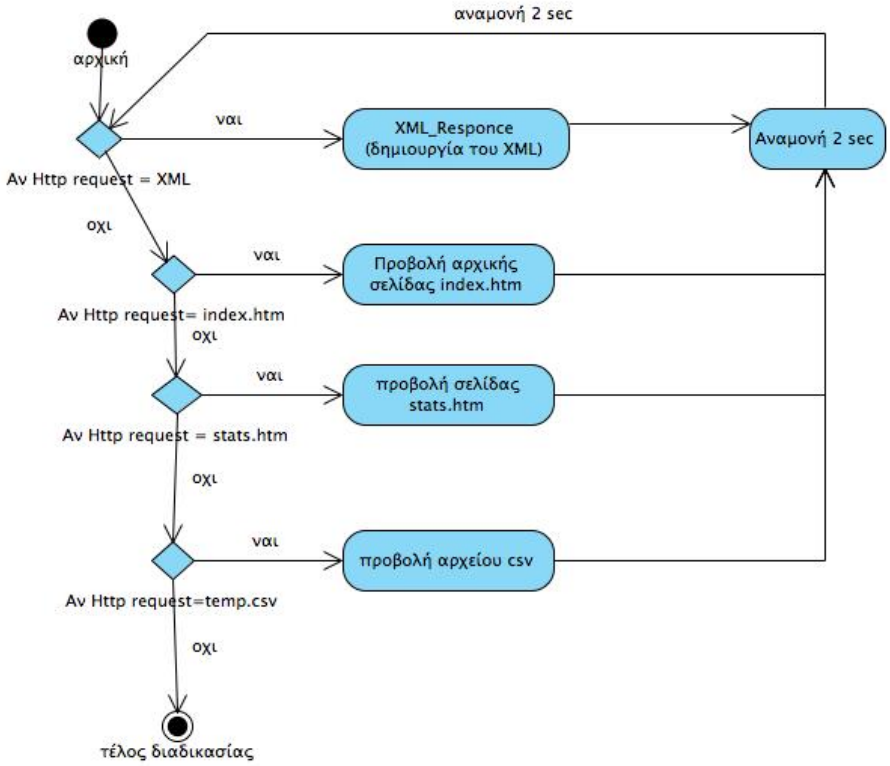
Κεφάλαιο 7ο

Διεπαφή Χρήστη

7. Κεφάλαιο 7ο – Διεπαφή Χρήστη

Το περιβάλλον χρήστη βασίζεται στην τεχνολογία HTML με χρήση JavaScript. Η μεταφορά δεδομένων από το Arduino στην ιστοσελίδα πραγματοποιείται μέσω της τεχνολογίας XML, ενώ η προβολή των γραφημάτων γίνεται με την χρήση έτοιμης βιβλιοθήκης JavaScript η οποία είναι ανοιχτού κώδικα και διατίθεται δωρεάν. Η διαδικασία λειτουργίας είναι η ακόλουθη:

Αρχικά αποστέλλεται απο τον φυλλομετρητή, http αίτημα στο Arduino. Στην συνέχεια το αίτημα αναλύεται και παρουσιάζεται στον φυλλομετρητή. Οι ιστοσελίδες βρίσκονται αποθηκευμένες στην κάρτα μνήμης SD υπό την μορφή αρχείων HTML. Για τον λόγο αυτό θα πρέπει να ενσωματωθεί η βιβλιοθήκη SD για την διαχείριση της κάρτας SD. Η διαδικασία λειτουργίας του Arduino ως εξυπηρετητή απεικονίζεται στο Σχήμα 35. Για οικονομία χώρου καθώς και της κίνησης μικρότερου όγκου δεδομένων, οι εικόνες και βιβλιοθήκες JavaScript που χρησιμοποιούνται δεν είναι αποθηκευμένες στον σταθμό, αλλά σε άλλους εξυπηρετητές.



Εικόνα 42: Διάγραμμα ροής διαδικασίας εξυπηρετητή διαδικτύου

Ο κώδικας της λειτουργίας ως εξυπηρετητή διαδικτύου παρατίθεται στο ***παράρτημα Θ***.

7.1 Αρχική Σελίδα

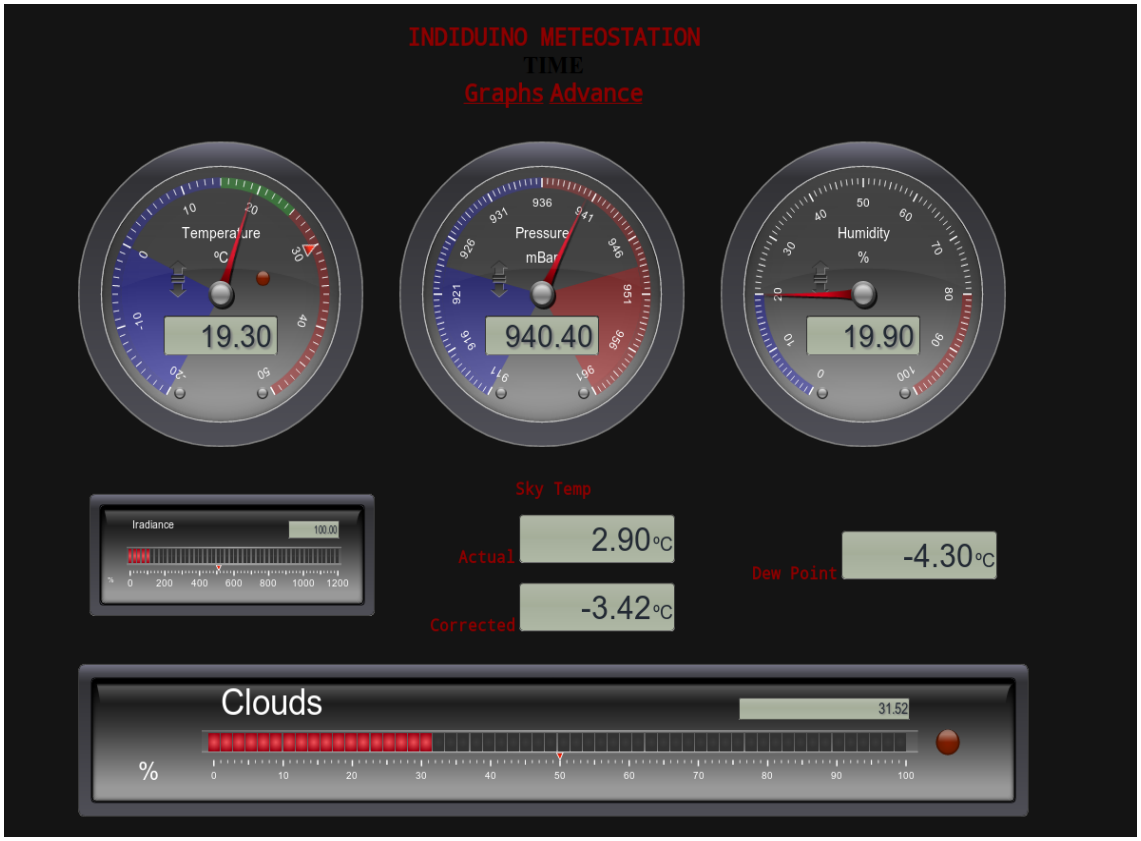
Είναι η σελίδα την οποία ο χρήστης παρακολουθεί όλες τις πληροφορίες του καιρού. Αρχικά ο χρήστης εισάγει την διεύθυνση IP ή το πλήρως αναγνωρισμένο όνομα τομέα (Fully Qualified Domain Name) στην οποία ανήκει ο σταθμός. Για τις ανάγκες της παρούσης εργασίας, η διεύθυνση IP δοκιμάστηκε σε τοπικό δίκτυο με διεύθυνση 192.168.2.20.

Μόλις ο χρήστης πληκτρολογήσει την διεύθυνση στην γραμμή εντολών του φυλλομετρητή, αποστέλλεται το αίτημα http

"HTTP/1.1 GET /" ή "HTTP/1.1 GET /index.htm"

το οποίο μεταφράζεται ως: «πρωτόκολλο HTTP έκδοση 1.1 ζητώ την σελίδα index.html" Το Arduino εφόσον ελέγξει την ύπαρξη του αρχείου στην κάρτα SD, το αποστέλλει στον χρήστη. Η σελίδα που προβάλλεται είναι η κεντρική σελίδα του σταθμού και περιέχει τα μετεωρολογικά και άλλα δεδομένα που περιέχουν πληροφορίες σχετικά με την κατάσταση του σταθμού.

Παρακάτω εμφανίζεται η κεντρική σελίδα του σταθμού.



Εικόνα 43: Αρχική Σελίδα του Σταθμού

Αναλυτικά τα δεδομένα που προβάλλονται είναι:

• Οπτικοποιημένη ένδειξη της πρόγνωσης του καιρού με χρήση εικόνων

• Την πρόγνωση του καιρού

• Τρέχουσα θερμοκρασία καθώς και η μέγιστη και ελάχιστη ημερήσια τιμή της

• Τρέχουσα Υγρασία καθώς και η μέγιστη και ελάχιστη ημερήσια τιμή της

• Τρέχουσα ατμ. πίεση καθώς και η μέγιστη και ελάχιστη ημερήσια τιμή της

• Το σημείο δρόσου

• Τον δείκτη δυσφορίας

• Την διεύθυνση ανέμου σε μοίρες

• Ημερομηνία ώρα σταθμού

• Κατάσταση αισθητήρων

• Τιμή της ηλεκτρικής τάσης σε βολτ

Όταν δεν αιτηθεί η προβολή κάποιου αρχείου, τότε το Arduino αποστέλλει ροή υπό την μορφή XML η οποία περιλαμβάνει όλες τις παραπάνω πληροφορίες. Στην συνέχεια ο φυλλομετρητής, κάνοντας χρήση της τεχνολογίας JavaScript και AJAX αναλύει το εισερχόμενο XML και δυναμικά ενημερώνει την ιστοσελίδα με τα τρέχοντα μετεωρολογικά δεδομένα. Με τον τρόπο αυτό αποφεύγεται η συνεχής ανανέωση της ιστοσελίδας του σταθμού, περιορίζοντας τον μεταφερόμενο όγκο δεδομένων, στον ελάχιστο δυνατό.

Το παραγόμενο XML έχει την παρακάτω δομή:

<?xmlversion = “1.0” ?>

<inputs>

<trend></trend>// βαρομετρική τάση

<prognosis></prognosis> //πρόγνωση καιρού

<temperature></temperature>//τρέχουσα θερμοκρασία

<humidity></humidity>//τρέχουσα υγρασία

<pressure></pressure>//τρέχουσα πίεση

<windDirection></windDirection>//κατεύθυνση ανέμου (λεκτικό)

<windDirecionDegrees></windDirectionDegrees>//κατεύθυνση ανέμου (μοίρες)

<maxTemp></maxTemp> //μέγιστη θερμοκρασία

<minTemp></minTemp> //ελάχιστη θερμοκρασία

<maxPress></maxPress> //μέγιστη πίεση

<minPress></minPress> //ελάχιστη πίεση

<maxHumi></maxHumi> //μέγιστη υγρασία

<minHumi></minHumi>// ελάχιστη υγρασία

<heatIndex></heatIndex>// δείκτης δυσφορίας

<dewPoint></dewPoint>// σημείο δρόσου

<prognosisCode></prognosisCode> //κωδικός πρόγνωσης

<stationDate></stationDate> // ημερομηνία σταθμού

<stationDate></stationDate> // ώρα σταθμού

<voltage></voltage> // ηλεκτρική τάση σταθμού

</inputs>

Η χρήση των ελληνικών χαρακτήρων στο Arduino δεν είναι εύκολη υπόθεση καθώς λόγω της wiring, υποστηρίζεται ο πίνακας χαρακτήρων ASCII μόνο των 127 πρώτων χαρακτήρων. Βέβαια είναι δυνατή η προβολή των ελληνικών, αλλά απαιτείται προγραμματισμός και φυσικά δέσμευση της πολύτιμης μνήμης SRAM. Για τον λόγο αυτό ο σταθμός αποστέλλει στην σελίδα έναν κωδικό. Στην σελίδα υπάρχει ένας πίνακας **forecast** σε γλώσσα *JavaScript* ο οποίος περιέχει την μετάφραση του αντίστοιχου λεκτικού στα ελληνικά. Για την άμεση κατανόηση της πρόγνωσης του καιρού εκτός του λεκτικού κειμένου, χρησιμοποιείται οπτικό βοήθημα που αναπαριστά την αντίστοιχη πρόγνωση με χρήση εικονιδίων. Εκτός του **forecast** υπάρχει και ο πίνακας **forecastIcons** ο οποίος περιέχει την πλήρη διαδρομή των εικόνων της κάθε κατάστασης του καιρού στα ελληνικά.

Τέλος υπάρχει δυνατότητα προβολής των καταγραμμένων μεγίστων και ελαχίστων τιμών της θερμοκρασίας υπό την μορφή γραφήματος και εικονικών οργάνων.

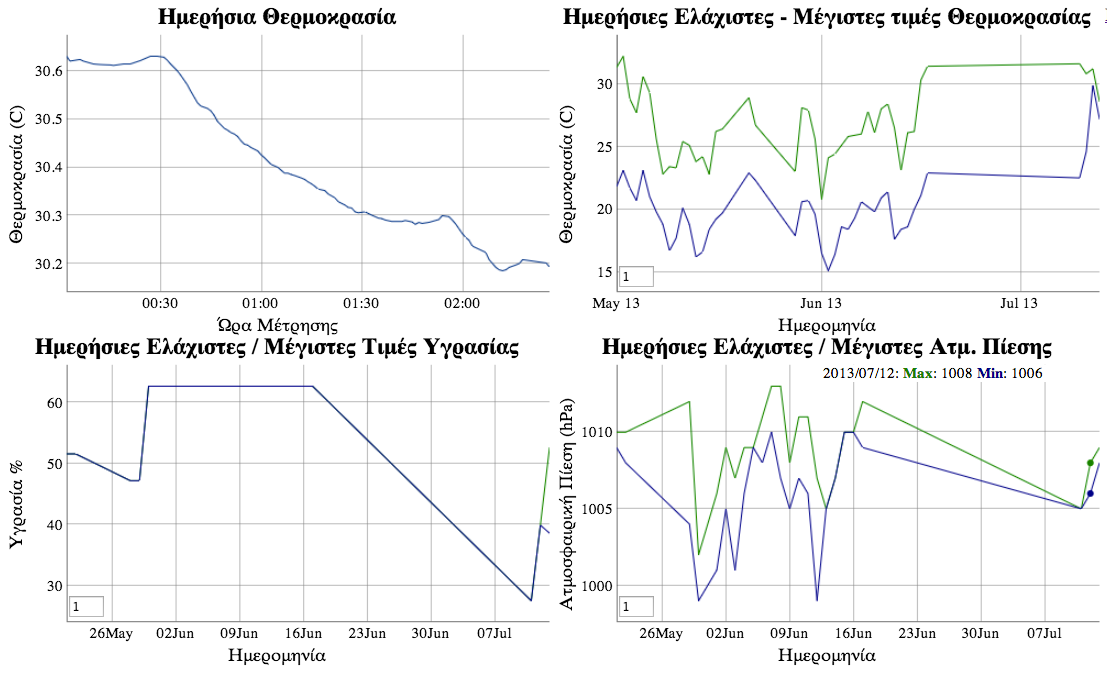
7.2 Προβολή Στατιστικών

Ο σταθμός κατά την ώρα 23:59 κάθε ημέρας αποθηκεύει την μέγιστη και ελάχιστη στις τιμές της θερμοκρασίας, υγρασίας και ατμοσφαιρικής πίεσης. Η αποθήκευση γίνεται σε μορφή csv στα αρχεία *temp.csv*, *humi.csv*, *pres.csv* αντίστοιχα . Το αρχείο αποθηκεύεται σύμφωνα με το πρότυπο μορφοποίησης ημερομηνίας και αριθμών ISO 8601. Τα περιεχόμενα του αρχείου είναι:

date, max, min

ΕΕΕΕΜΜΗΗ,0.00,0.00

Στην αρχική σελίδα ο χρήστης έχει την δυνατότητα να επιλέξει τον υπερσύνδεσμο *"για να δείτε τα στατιστικά πατήστε εδώ"* και να εμφανίσει την σελίδα *stats.html*. Σε αυτή την σελίδα ο χρήστης μπορεί να δει υπό την μορφή γραφημάτων, τις μέγιστες και ελάχιστες τιμές των μετεωρολογικών δεδομένων (ανά ημέρα) καθώς και τις μετρήσεις της τρέχουσας ημέρας.

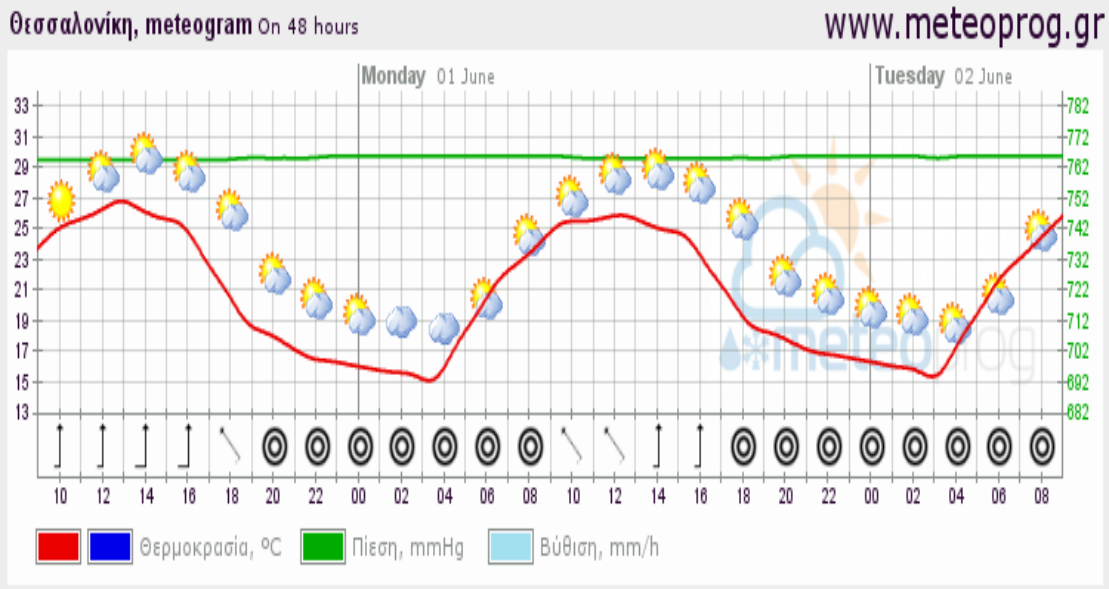


Εικόνα 44: Σελίδα προβολής Στατιστικών stats.html

Στην Εικόνα 44 εμφανίζεται η προβολή της σελίδας στατιστικών. Για την παραγωγή του γραφήματος χρησιμοποιήθηκε η βιβλιοθήκη *digraph* σε γλώσσα *JavaScript*. Η βιβλιοθήκη είναι ανοιχτού λογισμικού, και διατίθεται δωρεάν. Η βιβλιοθήκη δεν έχει αποθηκευτεί στο Arduino αλλά γίνεται η σύνδεση της απ' ευθείας από την σελίδα του δημιουργού της, για εξοικονόμηση χώρου και όγκου των δεδομένων από τον σταθμό. Ο σταθμός άλλωστε, έχει σχεδιαστεί για διασύνδεση στο διαδίκτυο, οπότε η αναφορά σε άλλους ιστότοπους θεωρείται δεδομένη.

7.3 Έλεγχος Μετεωρολογικών Δεδομένων

Σε αυτό το στάδιο συγκρίθηκαν τα δεδομένα που συνέλεξε ο σταθμός και βρίσκονται αποθηκευμένα στα αρχεία *temp.csv*, *humi.csv*, *pres.csv* με τα αντίστοιχα μετεωρολογικά δεδομένα που βρίσκονται στην διεύθυνση http://penteli.meteo.gr/stations/neamichaniona/.Με βάση αυτά τα δεδομένα δημιουργήθηκαν γραφικές παραστάσεις για να αξιολογηθεί η απόδοση των αισθητήρων και του αλγορίθμου καταγραφής.



Εικόνα 45: Πρόγνωση Καιρού για τη Θεσσαλονίκη στις 01-02/06/2015

8

Κεφάλαιο 8ο

Συμπεράσματα

8. Κεφάλαιο 8ο – Συμπεράσματα

Ο στόχος της παρούσης Πτυχιακής Εργασίας ήταν η υλοποίηση ενός αυτόνομου μετεωρολογικού σταθμού για την παρακολούθηση και καταγραφή μετεωρολογικών φαινομένων και στην συνέχεια με βάση αυτά τα στοιχεία να γίνεται πρόγνωση καιρού. Το επίπεδο ευαισθησίας των αισθητήρων είναι σε ικανοποιητικό βαθμό, όπως και οι αλγόριθμοι καταγραφής και αποθήκευσης των δεδομένων. Παρατηρούμε ελάχιστες αποκλίσεις σε σχέση με τους επαγγελματικού τύπου αισθητήρες.

Η τεχνολογία που χρησιμοποιήθηκε για την κατασκευή του εν λόγω σταθμού είναι:

α) Γλώσσα προγραμματισμού Wiring για την κατασκευή του κώδικα λειτουργίας της πλακέτας Arduino.

β) Γλώσσα HTML για την κατασκευή του γραφικού περιβάλλοντος του σταθμού μέσω διαδικτυακής ιστοσελίδας.

γ) Γλώσσα JavaScript για την παρουσίαση των γραφικών και στατιστικών στην ιστοσελίδα του σταθμού.

δ) Γλώσσα Ajax για την προβολή των απεσταλμένων από τον σταθμό Arduino στην ιστοσελίδα.

στ) Γλώσσα XML για την ομαδοποίηση και αποστολή των δεδομένων στην ιστοσελίδα του σταθμού.

Η διαχείριση μνήμης αποδείχτηκε σωτήρια καθώς ο μετεωρολογικός σταθμός, λόγω της περιορισμένης προσωρινής μνήμης SRAM, υπολειτουργούσε χωρίς να παρουσιάζεται κάποιο πρόβλημα κατά την μεταγλώττιση του κώδικα.

Το Arduino αποτελεί μια μικρή επανάσταση στον κόσμο της ρομποτικής. Η χρήση της Wiring ως γλώσσα προγραμματισμού, η εύκολη διασύνδεσή του με διάφορους αισθητήρες ή άλλες πλακέτες το καθιστούν ελκυστικό από άτομα που δεν είναι εξοικειωμένα με την ρομποτική. Η δημιουργία ενός αυτόνομου μετεωρολογικού σταθμού είναι μόνο η κορυφή του παγόβουνου καθώς οι δυνατότητες του εξελίσσονται με τέτοιο τρόπο, που το μόνο πλέον που μπορεί να το περιορίσει είναι η ανθρώπινη φαντασία.

Βιβλιογραφία

Ελληνόγλωσση

1. Καραπιπέρης Λ. 1997. *Ναυτική Μετεωρολογία*, Εκδόσεις Ίδρυμα Ευγενίδου, Αθήνα

Ξενόγλωσση

1. Margolis Michael, "Arduino Cookbook",(O'Reilly,Sebastopol,2011)
2. R.E. Munn, Compendium of Meteorology, (WMO, Volume II, 1979)
3. World Meteorological Organization, "Guide to Meteorological Instruments and Methods of Observation", (7th edition, 2008)
4. Warren John-David, "Arduino Robotics", (Apress, New York, 2012)
5. Vasquez Tim, "International Weather Watchers Observer Handbook", (1998)
6. Riley Mike, «Programming your home with Arduino, Android, and your Computer» (The Pragmatic Programmers ,2012)

Ιστοσελίδες

[1]. Wikipedia, "Weather Focasting", http://en.wikipedia.org/wiki/Weather_forecasting , (Ημ. Προσπέλασης 15/3/2015)

[2]. Wikipedia Θερμοκρασία Ατμόσφαιρας, http://el.wikipedia.org/wiki/%CE%98%CE%B5%CF%81%CE%BC%CE%BF%CE%BA%CF%81%CE%B1%CF%83%CE%AF%CE%B1_%CE%B1%CF%84%CE%BC%CF%8C%CF%83%CF%86%CE%B1%CE%B9%CF%81%CE%B1%CF%82, , (Ημ. Προσπέλασης 16/4/2015)

[3]. World Meteorological Organization, "Procedures And Formats For Exchange Of MONITORING RESULTS", https://www.wmo.int/pages/prog/www/DPS/Monitoring-home/mon-procedures.htm , (Ημ. προσπέλασης 18/4/2015)

[4]. Εθνικό Αστεροσκοπείο Αθηνών,
http://www.meteo.gr/pdf/deiktis_thermotitas.pdf , (Ημ Προσπέλασης 20/4/2015)

[5]. Wikipedia,
http://en.wikipedia.org/wiki/Heat_index , (Ημ Προσπέλασης 20/4/2015)

[6]. Arduino Επίσημη Ιστοσελίδα,
http:// www.arduino.cc (Ημ προσπέλασης 26/4/2015)

[7]. Ardupilot,
http://www.diydrones.com/notes/ArduPilot , (Ημ προσπέλασης 26/4/2015)

[8]. Datasheet BMP-180,
http://www.adafruit.com/datasheets/BMP180_DataSheet_Rev.1.0_01July2008.pdf ,
(Ημ προσπέλασης 26/4/2015)

[9]. "Atmospheric Pressure and Altimeters",
http://aviationweather.gov/general/pubs/front/docs/oct-02.pdf (Ημ .προσπέλασης
26/4/2015)

[10]. Wind Sensor Datasheet,
https://www.sparkfun.com/datasheets/Sensors/Weather/Weather%20Sensor%20Asse
mbly..pdf , (Ημ. προσπέλασης: 1/5/2015)

[11]. Arduino AnalogRead,
http://arduino.cc/en/Tutorial/AnalogInputPins , (Ημ προσπέλασης 1/5/2015)

[12]. Wikipedia,
http://en.wikipedia.org/wiki/Power_over_Ethernet , (Ημ Προσπέλασης 10/5/2015)

[13]. Wikipedia,
http://en.wikipedia.org/wiki/Network_Time_Protocol , (Ημ Προσπέλασης 10/5/2015)

[14]. Πληροφορίες για το EthernetShield,
http://arduino.cc/en/Main/ArduinoEthernetShield , (Ημ. προσπέλασης 10/5/2015)

[15]. Πληροφορίες για την βιβλιοθήκη DHTlib,
https://github.com/ringerc/Arduino-DHT22 , (Ημ. Προσπέλασης 11/5/2015)

[16]. Μετεωρολογικός Σταθμός NTUA,
http://openmeteo.org/db/timeseries/d/3813/, (Ημ προσπέλασης: 11/5/2015)

[17]. AJAX http://startingelectronics.com/tutorials/arduino/ethernet-shield-web-
server-tutorial/SD-card-AJAX-web-server/ , (Ημ. Προσπέλασης: 11/05/2015)

[18]. ISO 8601, https://en.wikipedia.org/wiki/ISO_8601 , (Ημ προσπ. 20/5/2015)

ΠΑΡΑΡΤΗΜΑΤΑ

Παράρτημα Α – Συνάρτηση Διασύνδεσης Κάρτας Ethernet Shield

```
#include <Ethernet.h> //δήλωση βιβλιοθήκης
//δήλωση υπολοίπων βιβλιοθηκών
...
//δήλωση διευθύνσεων MAC και IP
byte mac_addr[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED};
byteip_addr[]={192,168,2,100};
test_host[]={192,168,2,1};//
//υπόλοιπες δηλώσεις
...
EthernetClient client;
//συνάρτηση setup()
void setup() { Ethernet.begin(mac_addr, ip_addr);//έναρξη
λειτουργίας
client.connect(test_host, 80);//σύνδεση στο router για
έλεγχο λειτουργίας δικτύου
while (!client.connected()){ //όσο δεν έχει συνδεθεί στο
δίκτυο
Serial.println('error connect to client');
LCD.print('sfalmadiktyoy');//εκτύπωση στην οθόνη LCD
delay(5000); αναμονή 5 sec
}
… υπόλοιπος κώδικας.
}
```

Παράρτημα Β – συνάρτηση διασύνδεσης αισθητήρα DHT-22 / AM - 2302

```
#include<DHT22.h> //εισαγωγή βιβλιοθήκης
// υπόλοιπες βιβλιοθήκες
...
// αρχικοποίηση αισθητήρα
DHT22 senson(7)// ορίζουμε την αν. είσοδο 7.
//συνάρτηση ανάγνωσης αισθητήρα
voidreadHumAndTemp(void){
DHT22_ERROR_t errorCode; //ανάγνωση σφαλμάτων αισθητήρα
// ανάννωση κάθε 2 δευτερόλεπτα
delay(2000);
errorCode = myDHT22.readData(); //ανάγνωση σφαλμάτων
//για κάθε περίπτωση σφάλματος κάνε τα ακόλουθα
switch(errorCode)
```

```
{
//αν δεν εχει σφάλματα εκτύπωσε τα δεδομένα
case DHT_ERROR_NONE:
LCD.print("Got Data ");
LCD.print(myDHT22.getTemperatureC());
LCD.print("C ");
LCD.print(myDHT22.getHumidity());
LCD.println("%");
break;
//περίπτωσηλάθουςbit ισοτιμίας
case DHT_ERROR_CHECKSUM:
LCD.print("check sum error ");
break;
case DHT_BUS_HUNG:
Serial.println("BUS Hung ");
break;
case DHT_ERROR_NOT_PRESENT:
Serial.println("Not Present ");
break;
case DHT_ERROR_ACK_TOO_LONG:
Serial.println("ACK time out ");
break;
case DHT_ERROR_SYNC_TIMEOUT:
Serial.println("Sync Timeout ");
break;
case DHT_ERROR_DATA_TIMEOUT:
Serial.println("Data Timeout ");
break;
case DHT_ERROR_TOOQUICK:
Serial.println("Polled to quick ");
break;
}
```

Παράρτημα Γ – συνάρτηση Διασύνδεσης αισθητήρα BMP-180

```
#include "Adafruit_BMP180.h"//εισαγωγή βιβλιοθήκης
....
Adafruit_BMP180 pSensor; //αρχικοποίηση αισθητήρα
.....
void setup(){
....
while(!pSenson.begin()){;//αν δεν ξεκινήσει η ανάγνωση
LCD.print('sfalma aisthitira piesis);//τύπωσε σφάλμα
Delay(5000)//αναμονή 5 sec
}
pSensor.begin();//έναρξη ανάγνωσης αισθητήρα
....
```

```
}
//συνάρτηση ανάγνωσης και εκτύπωσης δεδομένων
readPSensor(){
LCD.print("Temperature = ");
LCD.print(bmp.readTemperature());
LCD.println(" *C");
LCD.print("Pressure = ");
LCD.print(bmp.readPressure());
LCD.println(" hPa");
Serial.println();
}
```

Παράρτημα Δ – συνάρτηση διασύνδεσης αισθητήρα διεύθυνσης ανέμου 80422

```
int getWindDirectionDegrees() {
unsigned int adc;
adc = analogRead(WIND_DIRECTION_PIN); // get the current
reading from the sensor
if (adc > 80 && adc < 104) return (90); //E
if (adc > 172 && adc < 196) return (135);//SE
if (adc > 274 && adc < 298) return (180);//S
if (adc > 488 && adc < 472) return (45);//NE
if (adc > 618 && adc < 642) return (225);//SW
if (adc > 774 && adc < 798) return (0);//N
if (adc > 966 && adc < 990) return (315);//NW
if (adc > 933 && adc < 957) return (270);//W
return (-1); // error, disconnected?
}
```

Παράρτημα Ε – Συνάρτηση καταγραφής δειγμάτων Ατμ. Πίεσης

```
int mesiTimiPiesis[5];//πίνακας στον οποίο αποθηκεύεται η
μέση τιμή. // Σε αυτόν τον πίνακα αποθηκεύεται η
ατμοσφαιρική πίεση.
loop(){
Int hPa=bmp.readPressure()/100;//λήψη τιμής ατμ. Πίεσης
και μετατροπής αποPa σε hPa
katagrafiDeigmatonPiesis(hPa); //κλήση συνάρτησης
καταγραφής
…υπόλοιπος κώδικας…
}
void katagrafiDeigmatonPiesis(intpressu){
int seconds = second();
```

```
//αν τα δευτερόλεπτα του ρολογιού είναι 12, 24, 36, 48 ,
00 τότε καταγράφεται η τρέχουσα τιμή στον πίνακα
//επειδή θέλουμε να υπολογίσει τα 00 του επόμενου λεπτού.
Για παράδειγμα η καταγραφή είναι μεταξύ των
//12:12, 12:24, 12:36, 12:48, 13:00. Άρα όταν το ρολόι
δείξει 0 θα πρέπει να το προσπεράσει ενώ όταν θα δείξει
59 θα περιμένει 1 δευτερόλεπτο ( θα φτάσει δηλαδή 0) και
θα γίνει η καταγραφή.
if (seconds==0) {
delay(1000);
seconds=1;
} else if (seconds==59) {
delay(1000);
seconds=0;
}
switch (seconds){
case 12:
mesiTimiPiesis[0] = pressu;
Serial.print("katagrafi 1ou Deigmatos:");
Serial.println(pressu);
break;
case 24:
mesiTimiPiesis[1] = pressu;
Serial.print("katagrafi 2ou Deigmatos:");
Serial.println(pressu);
break;
case 36:
mesiTimiPiesis[2] = pressu;
Serial.print("katagrafi 3ou Deigmatos:");
Serial.println(pressu);
break;
case 48:
mesiTimiPiesis[3] = pressu;
Serial.print("katagrafi 4ou Deigmatos:");
Serial.println(pressu);
break;
case 0:
mesiTimiPiesis[4] = pressu;
Serial.print("katagrafi 5ou Deigmatos:");
Serial.println(pressu);
kataxwrisiDeigmatosAnaLepto(mode(mesiTimiPiesis,5));//ότα
ν φτάσει στο τελευταίο δείγμα ξεκινάει η διαδικασία
καταχώρησης της μέτρησης του λεπτου.
break;
}
}
```

Παράρτημα ΣΤ - Συνάρτηση Υπολογισμού Βαρομετρικής τάσης

```
MesiTimiOrieasPiesis[60];
void kataxwrisiDeigmatosAnaLepto(inttimiPiesis){
int minutes = minute();
// symfwna me ton algoritmo katagrafis gia ton ypologismo
tou mesou orou tis oriaias barometrikis piesis
// h katagrafi ginete anamesa sta 50 kai oxi parapanw apo
to 58 lepto tis wras. edw orizw oti
// h katagrafi tha oloklironetai sto 55 lepto tis wras
// an den einaisto 55 lepto tote kategrapse tin mesitimi
ton proigoumenwn 5 metrisewn
// etsi apothikeuetai ston pinaka stin uesi px 8 i mesi
timi sta 8 lepta tis wras klp.
if (minutes !=55){
MesiTimiOrieasPiesis[minutes]=timiPiesis;
}
// an einai tote kategrapse tin timikai bale kataxorise
ton meso oro tis wras ston pinaka gia ton
// ypologismo tis barometrikis tasis
else if (minutes==55){
MesiTimiOrieasPiesis[55]=timiPiesis;
float pressure =mode(MesiTimiOrieasPiesis,60);
katagrafiBarometrikisTasis(pressure);//καταγραφή ωριαίας
μέσης τιμής ατμοσφαιρικής πίεσης για τον υπολογισμό της
βαρομετρικής τάσης.
}
Serial.print("katagrafiDeigmatos ");
printLCDStatus("katagrafiDeigmatos:");
Serial.print(minutes);
Serial.print(" leptou: ");
Serial.println(mode(mesiTimiPiesis,5));
printLCDStatus((char*)mode(mesiTimiPiesis,5));
```

Παράρτημα Ζ – Συνάρτηση αποθήκευσης βαρομετρικής τάσης

```
void katagrafiBarometrikisTasis(int pressure){
//Μετακίνηση των προηγούμενων τιμών στις n+1 θέσεις
for(int i=3;i=1;i++){
hourPressureTrend[i]=hourPressureTrend[i-1];
}
hourPressureTrend[0]=pressure;// καταχώρηση της τρέχουσας
τιμής t0
}
```

Παράρτημα Θ – Συνάρτηση λειτουργίας εξυπηρετητή διαδικτύου.

```
void createWebPage(){
EthernetClient client = server.available();
if (client) { // αν ο υπαρχει συνδεση
boolean currentLineIsBlank = true;
while (client.connected()) {
if (client.available()) { // αν τα δεδομενα του πελατη
διαβαζονται
charc = client.read(); // διαβασε το 1 byte (χαρακτηρα)
του πελατη
HTTP_req += c; // αποθηκευσε το HTTP request 1 char at a
time
// αν η τελευταια γραμμη του αιτηματος ειναι κενη και
τελειωνει με \n
// να απαντησεις εφοσον παρεις την κενη γραμμη
if (c == '\n' && currentLineIsBlank) {
// απαντησε εναstandard http response header
client.println("HTTP/1.1 200 OK");
// ελεγχοςαντοεισερχομενοαιτημαειναιxmlηαρχειοhtm
if (HTTP_req.indexOf("ajax_inputs") > -1) {
// αποστολη της υπολοιπης HTTP κεφαλιδας
client.println("Content-Type: text/xml");
client.println("Connection: keep-alive");
client.println();
// αποστολη του XML το οποιο περιεχει τα δεδομενα για
προβολη στην ιστοσελιδα
XML_response(client);
}
else { // αιτημα για προβολη ιστοσελιδας
// Αιτημα αρχικης σελιδα
if ((HTTP_req.indexOf("GET / ") > -1)
|| (HTTP_req.indexOf("GET /index.htm") > -1)) {
client.println("Content-Type: text/html");
client.println("Connection: keep-alive");
client.println();
webFile = SD.open("newide.htm"); // ανοιγμα αρχειου απο
καρτα sd
}
//αιτημα προβολης στατιστικων
else if (HTTP_req.indexOf("GET /stats.htm") > -1) {
client.println("Content-Type: text/html");
client.println("Connection: keep-alive");
client.println();
webFile = SD.open("stats.htm"); // ανοιγμα αρχειου απο
καρτα sd
}
//αιτημα προβολης αρχειου temp.csv
```

```
else if (HTTP_req.indexOf("GET /temp.csv") > -1) {
webFile = SD.open("temp.csv");
if (webFile) {
client.println("HTTP/1.1 200 OK");
client.println();}
}
else if (HTTP_req.indexOf("GET /pres.csv") > -1) {
webFile = SD.open("pres.csv");
if (webFile) {

client.println("HTTP/1.1 200 OK");
client.println();}
}
else if (HTTP_req.indexOf("GET /humi.csv") > -1) {
webFile = SD.open("humi.csv");
if (webFile) {
client.println("HTTP/1.1 200 OK");
client.println();}
}
//αιτημα προβολης αρχειου csv τρεχουσας ημερας
else if (HTTP_req.indexOf("GET /today.csv") > -1) {
//οι μετρησεις αποθηκευονται σε αρχειο csv με την μορφη
εεεεμμηη.csv για να μπορει καθε ημερα το αρχειο
//να διαβαζεται απο το javascript "εξαπατω" το αιτημα.
οταν μου ζηταει το today.csv εγω αποστελω το αρχειο με
την
//τρεχουσα ημερομηνια
char filenameZ[20];
sprintf(filenameZ, "%02d%02d%02d.csv",year(), month(),
day());
char* filename=filenameZ;
webFile = SD.open(filename);
if (webFile) {
client.println("HTTP/1.1 200 OK");
client.println();}
}
if (webFile) {
while(webFile.available()) {
client.write(webFile.read()); // ανοιγμα αρχειου απο
καρτα sd
}
webFile.close();
}
}
HTTP_req = ""; // finished with request, empty string
break;
}
// every line of text received from the client ends with
\r\n
if (c == '\n') {
// last character on line of received text
```

```
// starting new line with next character read
currentLineIsBlank = true;
}
else if (c != '\r') {
// a text character was received from client
currentLineIsBlank = false;
}
}
}
delay(1); // αναμονή 1 ms ώστε να δημιουργηθεί η σελίδα
client.stop(); // κλείσιμο σύνδεσης
}
}
```