Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ελεγκτής Δωματίου Υπολογιστών στοχευμένος στη Λήψη μετρήσεων και παροχής ασφάλειας, με Χρήση της Πλατφόρμας Arduino.** **Computer Room Controller aimed at taking messurements and security provision, based on Arduino Platform.**
Ονοματεπώνυμο Φοιτητή	**Χρήστος Δημόπουλος**
Πατρώνυμο	**Ιωάννης**
Αριθμός Μητρώου	**ΜΠΣΠ/ 13028**
Επιβλέπων	**Ευθύμιος Αλέπης, Επίκουρος**

Ημερομηνία Παράδοσης **Ιούλιος 2016**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Μαρία Βίρβου Καθηγήτρια	Ευθύμιος Αλέπης Επίκουρος Καθηγητής	Κωνσταντίνος Πατσάκης Επίκουρος Καθηγητής

Περίληψη

Η Διαθεσιμότητα, η Ακεραιότητα και η Εμπιστευτικότητα είναι οι τρεις βασικοί πυλώνες στους οποίους στηρίζεται κάθε πληροφοριακό σύστημα. Η σημαντικότητα αυτών γίνεται ακόμα πιο επιτακτική όταν η υποδομή απευθύνεται σε πληροφοριακά συστήματα και εγκαταστάσεις ειδικού ενδιαφέροντος όπως το Server Room.

Στόχος της παρούσας εργασίας είναι η δημιουργία ενός ολοκληρωμένου συστήματος ελέγχου του κεντρικού δωματίου όπου φιλοξενούνται οι υπολογιστές ενός οργανισμού – μιας εταιρείας, είτε ενός ιδρύματος (Server Room) και ο απομακρυσμένος έλεγχος τόσο των περιβαλλοντολογικών συνθηκών που επικρατούν σε αυτόν όσο και του ελέγχου πρόσβασης εξουσιοδοτημένου και μη προσωπικού στον χώρο αυτό.

Abstract

Availability, Integrity and Confidentiality are the three main pillars on which any information system depends. The significance of these further increases when the infrastructure is addressed to information systems and facilities of particular interest, such as the Server Room.

The objective of this work is the creation of an integrated control system of the central room which houses the computers of an organization - a company or an institution (Server Room) and the remote control of both the environmental conditions in this and the access control of authorized and unauthorized personnel in this area.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω το σύνολο των καθηγητών στο σύνολο της φοιτητικής μου πορείας για τις γνώσεις που μου παρήχαν, τον τρόπο με τον οποίο διαμόρφωσαν τη σκέψη και τον συλλογισμό μου, στοιχεία τα οποία αποτελούν σημαντικά εφόδια τόσο για την καριέρα όσο και για τη ζωή σε όλες τις εκφάνσεις της.. Επίσης θέλω να ευχαριστήσω την οικογένειά μου και την κοπέλα μου για την υπομονή που έδειξαν όλα αυτά τα χρόνια και τις προσπάθειες που έκαναν ώστε να μείνω αφοσιωμένος στους στόχους μου.

Ιούλιος 2016

Χρήστος Δημόπουλος

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

ΠΙΝΑΚΑΣ ΠΙΝΑΚΩΝ

Κεφάλαιο 1ο

1. Εισαγωγή Computer Room Guard

Η σύλληψη της ιδέας για τη δημιουργία του Computer Room Controller προήλθε από την σημαντικότητα της ασφάλειας που πρέπει να υπάρχει σε κάθε πληροφοριακή υποδομή. Ακόμα δε περισσότερο όταν αυτή εντάσσεται στο πλαίσιο ενός μεγάλου οργανισμού, ενός εκπαιδευτικού ιδρύματος είτε μίας μονάδας . Η Διαθεσιμότητα, η Ακεραιότητα και η Εμπιστευτικότητα είναι οι τρεις βασικοί πυλώνες στους οποίους στηρίζεται κάθε πληροφοριακό σύστημα.

Στόχος της παρούσας υλοποίησης είναι η ανάπτυξη ενός συστήματος ελέγχου της πληροφοριακής υποδομής, πυρήνας της οποίας είναι το κεντρικό δωμάτιο υπολογιστών (Server Room). Η εν λόγω ανάπτυξη θα επικεντρωθεί στην παρακολούθηση των περιβαλλοντικών συνθηκών που επικρατούν και επηρεάζουν άμεσα την εύρυθμη λειτουργία του συνολικού συστήματος και κατ' επέκταση τη διαθεσιμότητα της όλης υποδομής. Θα λαμβάνονται μετρήσεις θερμοκρασίας, υγρασίας, φωτεινότητας και περιεκτικότητας υδρογόνου του χώρου.

Επίσης σημαντικός παράγοντας είναι η ασφάλεια του χώρου και η αποτροπή εισόδου σε αυτόν, μη εξουσιοδοτημένου προσωπικού, είτε εκούσια είτε ακούσια, το οποίο αποτελεί παραβίαση της φυσικής ασφάλειας.

Μέσω της εν λόγω υλοποίησης, πέραν των μετρήσεων των περιβαλλοντικών συνθηκών και την ειδοποίηση όταν οποιαδήποτε τιμή υπερβεί τα επιτρεπτά όρια που έχουμε ορίσει, θα υλοποιηθεί σύστημα ελεγχόμενης εισόδου στον χώρο με καταγραφή ώρας και ατόμου που μπαίνει. Θα αποτρέπει τη μη εξουσιοδοτημένη είσοδο και σε περίπτωση παραβίασης θα παρέχεται ανάλογο σήμα συναγερμού.

Τέλος κάθε πληροφοριακό σύστημα δημιουργείται, συντηρείται και συνεχίζει να υπάρχει προκειμένου να εξυπηρετήσει τον άνθρωπο μέσω συγκεκριμένων λειτουργιών για τις οποίες σχεδιάστηκε. Συνεπώς, επίκεντρο όλων είναι ο άνθρωπος και γι' αυτό στην υλοποίηση συμπεριλήφθηκε η λήψη μετρήσεων για αέρια τα οποία μπορεί να τον βλάψουν. Τέτοια αέρια στον χώρο του computer room υπάρχει πιθανότητα να παραχθούν είτε από τις μπαταρίες των UPS είτε και από τα ίδια τα μηχανήματα.

1.1. Περιγραφή του υπό Μελέτη Προβλήματος

Ο χώρος του computer Room αποτελεί το κέντρο του πληροφοριακού συστήματος στο οποίο λειτουργεί το σύνολο των κεντρικών υπολογιστών που είναι υπεύθυνοι για την ομαλή λειτουργία του δικτύου στο σύνολό του. Συνεπώς οι συνθήκες, τόσο οι περιβαλλοντολογικές όσο και η ασφάλεια που πρέπει να παρέχεται στην εν λόγω υποδομή, είναι πρωταρχικής σημασίας. Με την εν λόγω υλοποίηση στόχο έχουμε τον έλεγχο της θερμοκρασίας του δωματίου, η οποία πρέπει να είναι σταθερή και να μην περνάει κάποια όρια τα οποία ενδεχομένως να επηρεάζουν την εύρυθμη λειτουργία των υπολογιστών. Σημαντικό είναι να αναφέρουμε ότι λόγω του ότι τα Server Rooms ακολουθούμενα τα πρότυπα ασφαλείας δεν πρέπει να διαθέτουν παράθυρο αλλά μόνο μία κεντρική είσοδο η οποία να παρέχει ασφάλεια στον χώρο, η ψύξη του χώρου γίνεται από ανεξάρτητα συστήματα τα οποία σε περίπτωση βλάβης πρέπει να υπάρχει άμεση ειδοποίηση του διαχειριστή και αντίστοιχη λήψη μέτρων. Μέσω του συστήματος ελέγχου του Computer Room θα ελέγχεται τόσο η θερμοκρασία του χώρου όσο και η εύρυθμη λειτουργία των μέσων ψύξης του χώρου.

Άλλος σημαντικός παράγοντας είναι η υγρασία που υπάρχει μέσα στον χώρο λόγω των ηλεκτρονικών που υπάρχουν και η οποία πρέπει να παραμένει σε χαμηλά επίπεδα. Μέσω ειδικού αισθητήρα υγρασίας θα λαμβάνονται μετρήσεις και θα αποστέλλονται για περαιτέρω επεξεργασία.

Επίσης στον χώρο του Computer Room υπάρχουν UPS προκειμένου να έχουμε τη συνεχή λειτουργία των υπολογιστών ακόμα και σε περίπτωση διακοπής ρεύματος. Η ύπαρξη όμως μπαταριών σε κλειστό χώρο μπορεί να προκαλέσει την έκλυση H2 και άλλων αερίων επιβλαβών για τον άνθρωπο. Σε ενδεχόμενη είσοδο προσωπικού για συντήρηση χωρίς να έχει ληφθεί ανάλογη προειδοποίηση και τα αντίστοιχα μέτρα ασφαλείας μπορεί να προκληθεί σωματική βλάβη σ'αυτό. Έτσι μέσω της κατασκευής θα λαμβάνονται μετρήσεις και θα αποστέλλονται κι αυτές στη διαδικτυακή εφαρμογή.

Συνδυάζοντας τα δεδομένα μετρήσεων αερίων, φωτεινότητας και θερμοκρασίας μπορούμε να υλοποιήσουμε σύστημα συναγερμού σε ενδεχόμενο φωτιάς στον χώρο.

Τέλος θα υπάρχει έλεγχος των ατόμων που εισέρχονται στον χώρο επιτρέποντας και καταγράφοντας -την είσοδο εξουσιοδοτημένου προσωπικού και αντίστοιχα αποτρέποντας και ειδοποιώντας για παράνομη είσοδο στον χώρο. Το μέσο πιστοποίησης του ατόμου που εισέρχεται στον χώρο είναι η προσωπική κάρτα Rfid.

1.2. Παραδοτέα Εργασίας

Η παρούσα εργασία λόγω της διττής φύσης της, δηλαδή το ότι αποτελείται τόσο από προγραμματιστικό όσο και από κατασκευαστικό κομμάτι, θα παραδοθεί στα εξής επιμέρους τμήματα:

1. Το παρόν έντυπο, κείμενο της πτυχιακής εργασίας, στο οποίο υπάρχει αναλυτική περιγραφή των στόχων που επιτυγχάνονται μέσω της υλοποίησης του Computer Room Controller. Η ανάλυση του συνόλου των αισθητήρων και των ηλεκτρονικών διατάξεων που χρησιμοποιήθηκαν για την υλοποίηση της συσκευής, το κομμάτι του κώδικα που αναπτύχθηκε για το καθένα από αυτά και τέλος το πρόγραμμα το οποίο κατασκευάστηκε από την πλευρά του Server.
2. Το δεύτερο παραδοτέο είναι η ίδια η συσκευή του Computer Room Controller.
3. Ο κώδικας που αναπτύχθηκε για την υποστήριξη του Server Room Controller από την πλευρά του Server και η διαδικτυακή εφαρμογή.

1.3. Δομή της Εργασίας

Η παρούσα εργασία λόγω της διττής φύσης της θα αναλυθεί σε επιμέρους τμήματα. Στο πρώτο θα γίνει περιγραφή του υλικού που θα χρησιμοποιηθεί για να επιτευχθεί η κάθε λειτουργία ξεχωριστά καθώς και ο επιμέρους κώδικας που έχει αναπτυχθεί για κάθε εξάρτημα με την αντίστοιχη συνδεσμολογία. Η επιμέρους ανάλυση του κάθε στοιχείου της κατασκευής γίνεται προκειμένου να εκτεθεί και να αναλυθεί πλήρως η κατασκευή και να γίνει πλήρως κατανοητός ο τρόπος λειτουργίας και η συνδεσμολογία της διάταξης. Στο δεύτερο τμήμα τα επιμέρους υποσυστήματα θα αναλυθούν επί του συνόλου με τα ανάλογα σχέδια και τον συνολικό κώδικα ο οποίος δεν βασίζεται πλέον αποκλειστικά στη λειτουργία του κάθε εξαρτήματος αλλά στον συνολικό στόχο της κατασκευής λαμβάνοντας υπόψη τους περιορισμούς του κάθε εξαρτήματος. Στο τρίτο μέρος θα γίνει περιγραφή του προγράμματος από την πλευρά της εφαρμογής. Τέλος θα περιγραφούν οι προβληματισμοί και οι στόχοι που επιτεύχθηκαν καθώς και οι μελλοντικοί στόχοι της συσκευής όσο και του προγράμματος, καθώς και η ανάλυση του κόστους υλοποίησης της κατασκευής.

Κεφάλαιο 2ο

2. Επισκόπηση

Καρδιά του Controler αποτελούν το σύνολο των αισθητήρων και των πλακετών που το αποτελούν και η αλληλεπίδραση αυτών με την εφαρμογή. Ο συνδυασμός των δύο θα προσδώσει στην κατασκευή το σύνολο των «αισθήσεων» που απαιτούνται για την επίτευξη του επιθυμητού αποτελέσματος. Το ανάπτυγμα αυτό αποτελείται από την κεντρική μονάδα, όπου στην περίπτωσή μας είναι η πλατφόρμα Arduino Mega, μαζί με το σύνολο των αισθητήρων και των επεκτάσεων shields που απαιτούνται.

Συγκεκριμένα οι αισθητήρες και τα shields που θα χρησιμοποιηθούν είναι οι εξής:

1. Αισθητήρας Θερμοκρασίας
2. Αισθητήρας Υγρασίας
3. Αισθητήρας Φωτεινότητας
4. Αισθητήρας Κίνησης
5. Δέκτης GPS
6. Πομποδέκτης RF
7. GSM-GPRS Module
8. Πιεζοδιακόπτες θυρών
9. Αισθητήρας αερίων MQ-5
10. Ethernet Shield
11. Αισθητήρας RFID

Για την κατανόηση του συνόλου της κατασκευής η περιγραφή θα γίνει σε δύο μέρη. Στο πρώτο θα αναλυθούν όλοι οι αισθητήρες και η κεντρική πλατφόρμα μόνες τους. Δηλαδή θα γίνεται περιγραφή του εξαρτήματος, αιτιολογία επιλογής του καθενός όπου αυτό απαιτείται, παρουσίαση της συνδεσμολογίας του και του κώδικα για το καθένα ξεχωριστά. Στο δεύτερο θα αναλυθεί το συνολικό διάγραμμα της κατασκευής και η αλληλεπίδραση μεταξύ τους όπως και το συνολικό διάγραμμα της κατασκευής. Με αυτό τον τρόπο θα αποσαφηνιστεί πλήρως ο τρόπος ανάπτυξης και υλοποίησης της κατασκευής.

2.1. Αισθητήρας Θερμοκρασίας

Μία από τις «αισθήσεις» που θέλουμε να προσδώσουμε στην κατασκευή μας είναι αυτή της μέτρησης θερμοκρασίας. Η θερμοκρασία είναι πολύ σημαντική για την κατασκευή μας αφού αποτελεί βασικό στοιχείο πολλών αποτελεσμάτων των οποίων θέλουμε να εξάγουμε.

Πρώτη χρήση των δεδομένων θερμοκρασίας είναι η έγκαιρη ειδοποίηση του εκάστοτε Administrator για τα εκτός ορίων επίπεδα θερμοκρασίας τα οποία επικρατούν στον χώρο και η άμεση λήψη μέτρων από την πλευρά του.

Προκειμένου να υπάρχει αποτελεσματικότερος έλεγχος και πρόβλεψη αιτιών ανόδου θερμοκρασίας στην όλη κατασκευή θα τοποθετηθούν δύο αισθητήρες. Ο λόγος για τη χρήση περισσότερων του ενός είναι τόσο για αναπληρωματικότητα όσο και σε περίπτωση αστοχίας του

ενός να υπάρχει η δυνατότητα συνέχισης λήψης μετρήσεων μέχρι την επιδιόρθωση του άλλου. Δεύτερος λόγος είναι ότι οι δύο αισθητήρες θα τοποθετηθούν σε διαφορετικά σημεία. Ο πρώτος θα τοποθετηθεί σε μικρή απόσταση από τους υπολογιστές, ενώ ο δεύτερος κοντά στο σύστημα ψύξης. Με αυτό τον τρόπο θα μπορούμε να ελέγχουμε μέσω της εφαρμογής πιθανή αστοχία του συστήματος ψύξης μέσω της εξίσωσης των τιμών των δύο θερμοκρασιών.

Εικόνα 1: Αισθητήρας Θερμοκρασίας DS18B20

2.1.1. Τεχνικά Χαρακτηριστικά του Αισθητήρα Θερμοκρασίας

Τα τεχνικά χαρακτηριστικά του DS18B20 είναι τα εξής:

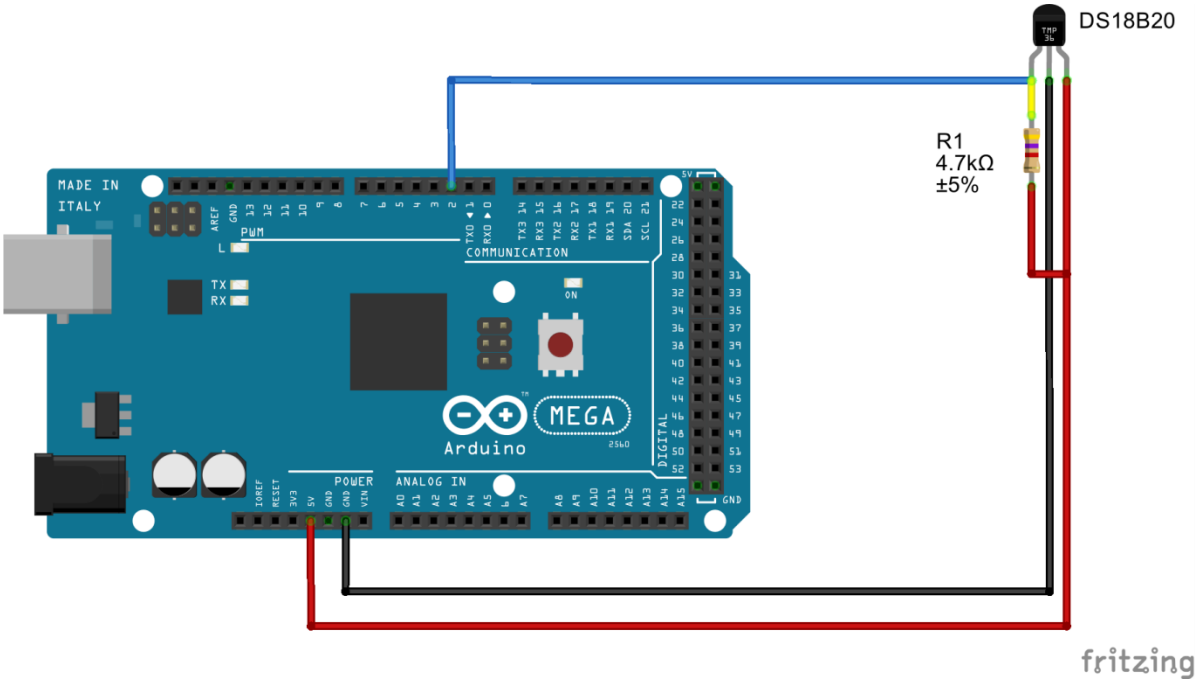
1. Εύρος θερμοκρασίας -55 °C έως και +125 °C
2. Ακρίβεια μέτρησης ±0.5°C για τιμές θερμοκρασιών ανάμεσα στους -10°C και +80 °C
3. Τα δεδομένα λαμβάνονται από ένα pin
4. Τάση τροφοδοσίας που κυμαίνεται από τα 3-5,5 Volt
5. Αδιάβροχο περίβλημα
6. Μέτρηση θερμοκρασίας υγρών

Όπως διαπιστώνουμε και από τα τεχνικά χαρακτηριστικά του DS18B20 ο συγκεκριμένος αισθητήρας καλύπτει πλήρως τις ανάγκες της εφαρμογής μας. Το εύρος μέτρησης θερμοκρασιών υπερκαλύπτει τις πιθανές τιμές μέτρησης στον χώρο εγκατάστασης, η ακρίβεια είναι πολύ μεγάλη για το εύρος τιμών αφού μέγιστη ακρίβεια απαιτείται για θερμοκρασίες από 5 έως και 45 °C. Η συνδεσμολογία του απαιτεί μόνο μία είσοδο η οποία επιτρέπει τη μη χρησιμοποίηση άλλων υπαρχόντων εισόδων οι οποίες θα χρησιμοποιηθούν σε άλλους αισθητήρες και shields. Σημαντικό στοιχείο είναι οι χαμηλές ενεργειακές απαιτήσεις και το γεγονός ότι δεν απαιτεί χρόνο μέχρι να ληφθεί σωστή μέτρηση. Τέλος η κατασκευή του και το μεταλλικό περίβλημα τον κάνει αδιάβροχο και του δίνει τη δυνατότητα να μετρά τόσο σε ξηρό όσο και σε υγρό περιβάλλον.

2.1.2. Διάγραμμα Συνδεσμολογίας Αισθητήρα Θερμοκρασίας

Όπως αναφέρθηκε και στα τεχνικά χαρακτηριστικά ο αισθητήρας θερμοκρασίας αποτελείται από 3 Pin- ακροδέκτες. Ο πρώτος, μαύρου χρώματος, είναι η γείωση, ο δεύτερος, κόκκινου χρώματος, είναι η τάση και ο τρίτος είναι ο ακροδέκτης δεδομένων ο οποίος θα συνδεθεί με την πλακέτα του Arduino Mega.

Για την συνδεσμολογία θα χρησιμοποιηθεί και μία αντίσταση R=4,7 ΚΩ όπως φαίνεται στο διάγραμμα που ακολουθεί:



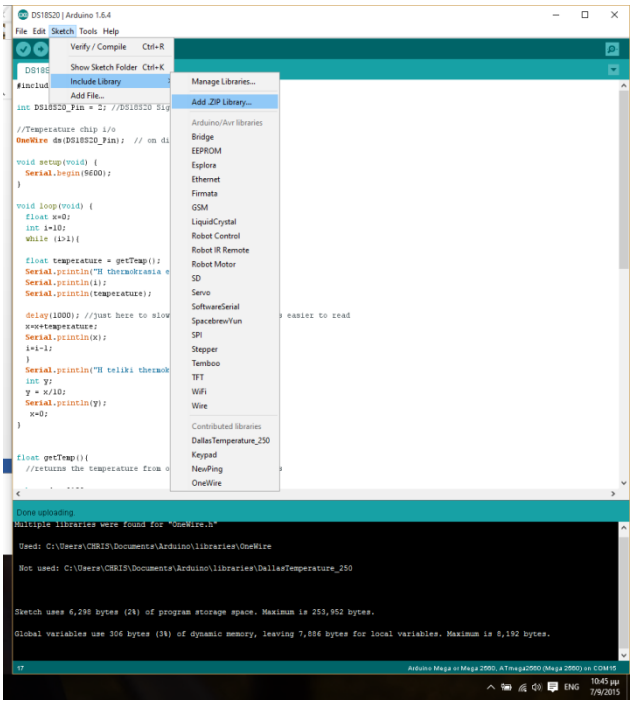
Εικόνα 2: Διάγραμμα Συνδεσμολογίας DS18B20

Το σύνολο των ηλεκτρονικών διαγραμμάτων έχει αναπτυχθεί με τη βοήθεια του προγράμματος fritzing.

2.1.3. Ανάλυση Προγράμματος για Χρήση Αισθητήρα Θερμοκρασίας

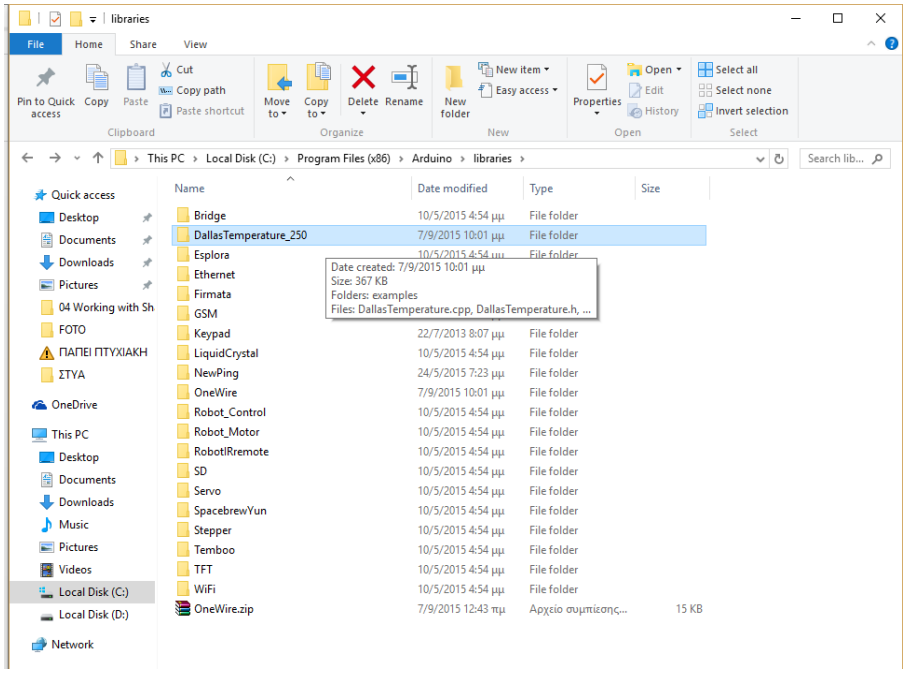
Για τη χρήση του Αισθητήρα θερμοκρασίας DS18B20 πρέπει να χρησιμοποιηθούν οι βιβλιοθήκες DallasTemperature και OneWireLibrary οι οποίες πρέπει να κατεβούν και να περαστούν στις βιβλιοθήκες του Arduino IDE.

Η διαδικασία εγκατάστασης μιας βιβλιοθήκης, όταν αυτή είναι σε μορφή συμπιεσμένου αρχείου, γίνεται μέσω του προγράμματος Arduino IDE στο menu “Sketch” → “Include Library” → “Add ZIP Library…” και έπειτα επιλέγοντας το αρχείο της βιβλιοθήκης στον χώρο που επιλέξαμε να το αποθηκεύσουμε, το εισάγουμε στις βιβλιοθήκες.



Εικόνα 3: Εγκατάσταση Βιβλιοθήκης σε μορφή ZIP

Άλλος τρόπος εισαγωγής μιας βιβλιοθήκης είναι αφού έχουμε κατεβάσει το/τα αρχεία των βιβλιοθηκών μεταβαίνουμε στον «My PC» → επιλέγουμε τον σκληρό δίσκο που έχουμε περάσει το Arduino IDE και ανάλογα με την έκδοση της πλατφόρμας που έχουμε x32 είτε x64 επιλέγουμε είτε το "Program File" είτε το "Program Files(x86)" → επιλέγουμε το φάκελο "Arduino" → τέλος μέσα στο φάκελο "libraries" αποσυμπιέζουμε το αρχείο της βιβλιοθήκης μας.



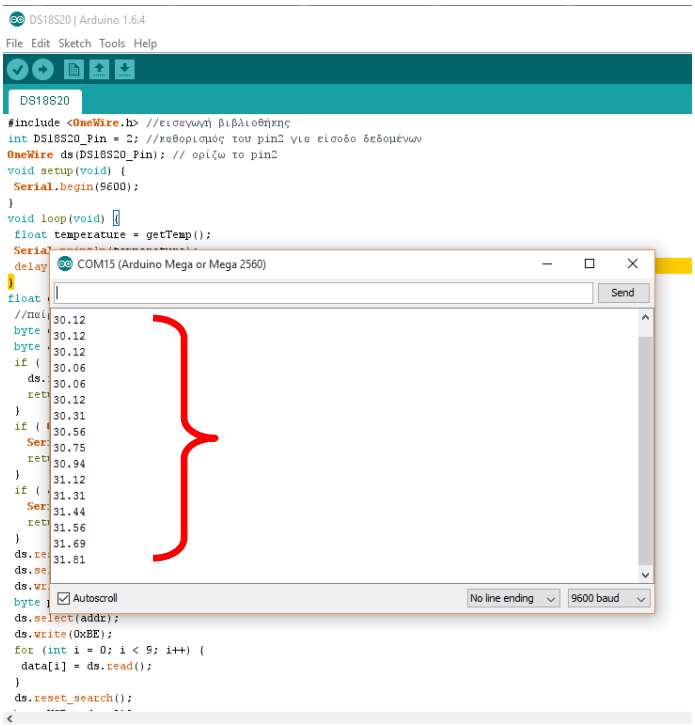
Εικόνα 4: Εγκατάσταση Βιβλιοθήκης

Το πρόγραμμα για τον αισθητήρα θερμοκρασίας προκειμένου να λαμβάνονται συνεχώς μετρήσεις είναι το εξής:

```
#include <OneWire.h>
//εισαγωγή βιβλιοθήκης
int DS18S20_Pin = 2;
//καθορισμός του pin2 για είσοδο δεδομένων
OneWire ds(DS18S20_Pin); // ορίζω το pin2
void setup(void) {
 Serial.begin(9600);
}
void loop(void) {
 float temperature = getTemp();
 Serial.println(temperature);
 delay(1000); //καθυστέρηση 1 δευτερόλεπτο ή 1000 msec
}
float getTemp(){
 //παίρνουμε την τιμή από τον αισθητήρα σε βαθμούς Κελσίου
 byte data[12];
 byte addr[8];
 if ( !ds.search(addr)) {
   ds.reset_search();
   return -1000;
 }
 if ( OneWire::crc8( addr, 7) != addr[7]) {
   Serial.println("CRC is not valid!");
   return -1000;
 }
 if ( addr[0] != 0x10 && addr[0] != 0x28) {
   Serial.print("Device is not recognized");
   return -1000;
 }
 ds.reset();
 ds.select(addr);
 ds.write(0x44,1);
 byte present = ds.reset();
 ds.select(addr);
 ds.write(0xBE);
 for (int i = 0; i < 9; i++) {
  data[i] = ds.read();
 }
 ds.reset_search();
 byte MSB = data[1];
 byte LSB = data[0];
 float tempRead = ((MSB << 8) | LSB);
 float TemperatureSum = tempRead / 16;
 return TemperatureSum;
}
```

Το ανωτέρω πρόγραμμα αρχικά εισάγει όλες τις απαραίτητες βιβλιοθήκες, στη συνέχεια καθορίζει τον ακροδέκτη στον οποίο έχουμε συνδέσει τον αισθητήρα μας και εκτελεί τις εντολές που βρίσκονται εντός του void loop(void). Εντός του βρόχου οι εντολές αφορούν τη λήψη της μέτρησης από τον αισθητήρα, την εμφάνιση της θερμοκρασίας στη σειριακή θύρα και τέλος αναμονή 1sec για τη λήψη της επόμενης μέτρησης.

Με τη φόρτωση του παραπάνω κώδικα στην πλατφόρμα του Arduino Mega και ανοίγοντας το παράθυρο επικοινωνίας της σειριακής θύρας λαμβάνουμε τη θερμοκρασία του χώρου όπως φαίνεται και στην εικόνα που ακολουθεί:



Εικόνα 5: Λήψη Θερμοκρασίας

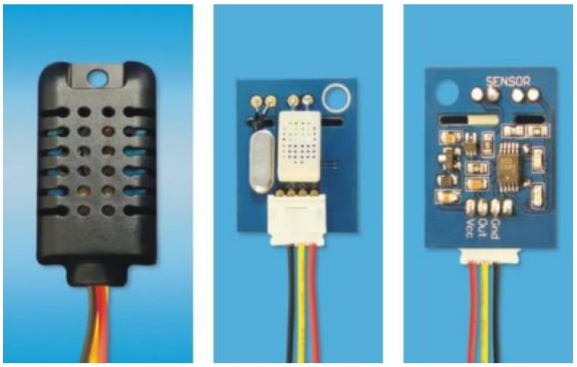
2.2. Αισθητήρας Υγρασίας

Δεύτερη «αίσθηση» που θέλουμε να προσδώσουμε στην κατασκευή μας είναι αυτή της μέτρησης υγρασίας. Ο αισθητήρας ο οποίος επιλέχθηκε για τη χρήση αυτή είναι ο AM2301 της εταιρείας AOSONG. Η επιλογή του συγκεκριμένου αισθητήρα έγινε λόγω της υψηλής ποιότητας κατασκευής, της γρήγορης απόκρισης (που συνεπάγεται ελαχιστοποίηση του χρόνου ενεργοποίησης), της μικρής πιθανότητας λάθους, του μικρού μεγέθους, της δυνατότητας μέτρησης με ακρίβεια ακόμα και με σύνδεση με καλώδιο έως και 20 μέτρα, του περιβλήματος προστασίας και της χαμηλής ενεργειακής κατανάλωσης. Τέλος ένα σημαντικό χαρακτηριστικό είναι η δυνατότητα λήψης πέραν της υγρασίας και της θερμοκρασίας. Με αυτό τον τρόπο μπορούμε να έχουμε μία δεύτερη αναφορά στη μέτρηση σε συνδυασμό με τη μέτρηση από τον αισθητήρα DS18B20 έτσι ώστε να έχουμε ειδοποίηση σε περίπτωση αστοχίας κάποιου εκ των δύο και ειδοποίηση του Administrator.

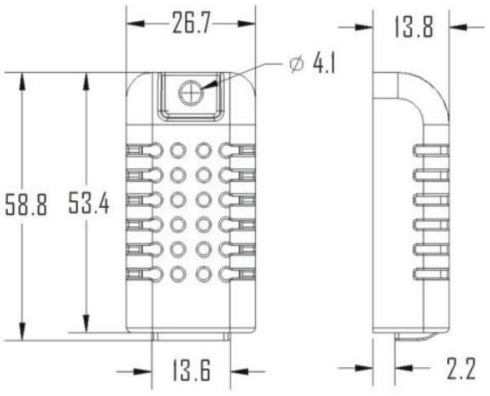
Εικόνα 6: Αισθητήρας AM2301 της εταιρίας AOSONG

2.2.1. Τεχνικά Χαρακτηριστικά του AM2301

Γενικά τεχνικά χαρακτηριστικά του AM2301:



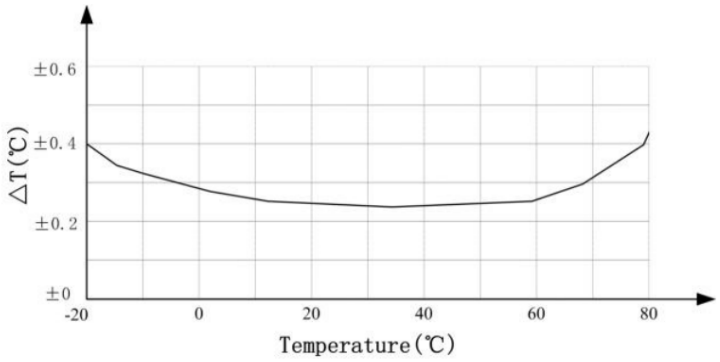
Εικόνα 7: Απεικόνιση AM2301 εμπρός(αριστερά), εσωτερικού εμπρός (κέντρο) και εσωτερικό πίσω όψη (δεξιά)



Εικόνα 8: Διαστάσεις του AM2301 σε mm

2.2.1.1. Τεχνικά Χαρακτηριστικά του AM2301για Θερμοκρασία

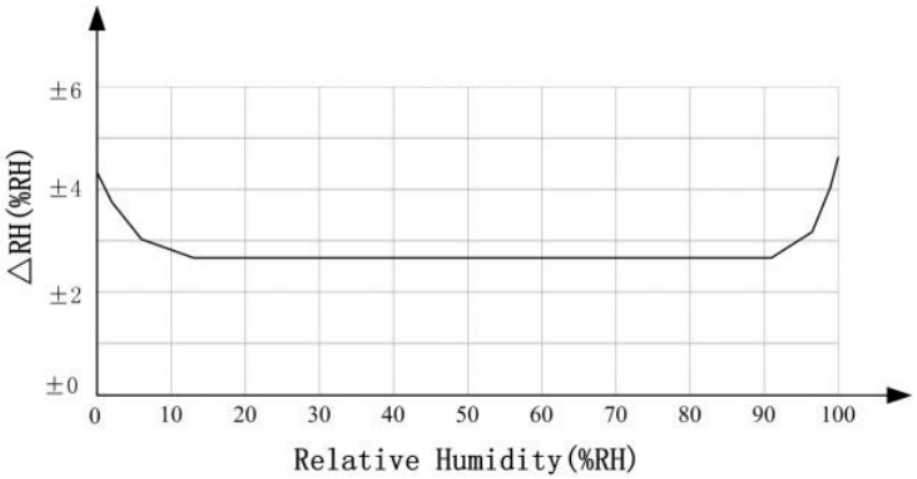
1. Ανάλυση- διακριτότητα μέτρησης θερμοκρασίας 0,1 ℃ - 16bit
2. Ακρίβεια μέτρησης θερμοκρασίας ελάχιστη : ±0,3 μέγιστη: ±1 ℃
3. Εύρος μέτρησης θερμοκρασίας -40 έως 80 ℃
4. Διαφοροποίηση τιμής ανά έτος χρήσης ±0.3 ℃/χρόνο



Εικόνα 9: Διάγραμμα μέγιστου λάθους θερμοκρασίας

2.2.1.2. Τεχνικά Χαρακτηριστικά του AM2301 για Υγρασία

1. Εύρος τιμών υγρασίας 0 έως 99,9 %RH
2. Ακρίβεια μέτρησης υγρασίας ±3 %RH στους 25℃
3. Διαφοροποίηση ανά έτος χρήσης <0,5 %RH/χρόνο



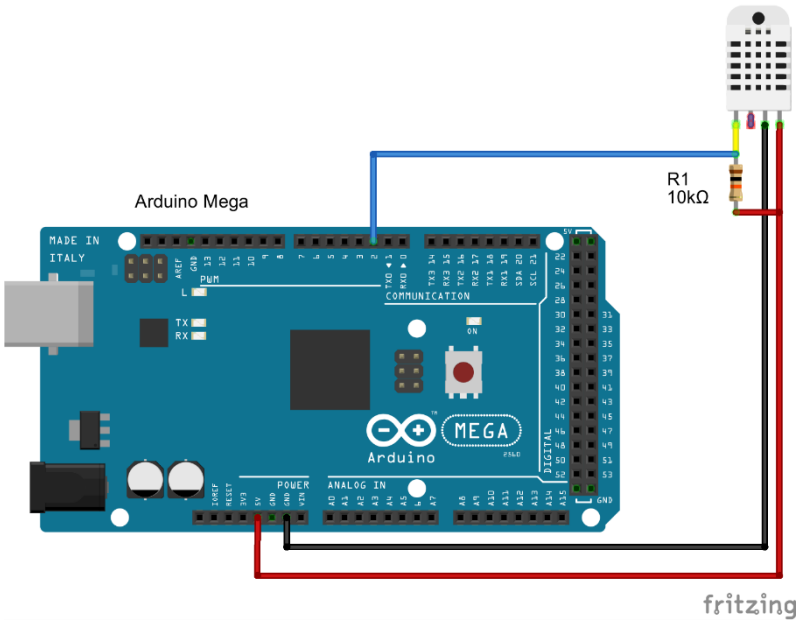
Εικόνα 10: Διάγραμμα σχετικού λάθους στη μέτρηση υγρασίας

2.2.2. Διάγραμμα Συνδεσμολογίας Αισθητήρα AM2301

Ο αισθητήρας AM2301 αποτελείται από τρία pin – ακροδέκτες. Ο πρώτος, μαύρου χρώματος, αποτελεί τη γείωση, ο δεύτερος, κόκκινου χρώματος, την τάση και ο τρίτος, κίτρινου/μπλε χρώματος, τον ακροδέκτη δεδομένων, ο οποίος θα στέλνει τα δεδομένα υγρασίας και θερμοκρασίας στην πλακέτα Arduino Mega όπως και ο προηγούμενος αισθητήρας.

Η συνδεσμολογία του AM2301 φαίνεται στο παρακάτω διάγραμμα:

Για τη συνδεσμολογία θα χρησιμοποιηθεί και μία αντίσταση R=10 kΩ όπως φαίνεται στο διάγραμμα που ακολουθεί:



Εικόνα 11: Διάγραμμα Συνδεσμολογίας AM2301

2.2.3. Ανάλυση Προγράμματος για τον Αισθητήρα AM2301

Για τη χρήση του Αισθητήρα AM2301 πρέπει να χρησιμοποιηθούν η βιβλιοθήκη DTH.h η οποία πρέπει να κατεβεί και να εισαχθεί στις βιβλιοθήκες του Arduino IDE όπως περιγράφηκε και με τον αισθητήρα θερμοκρασίας στην προηγούμενη ενότητα.

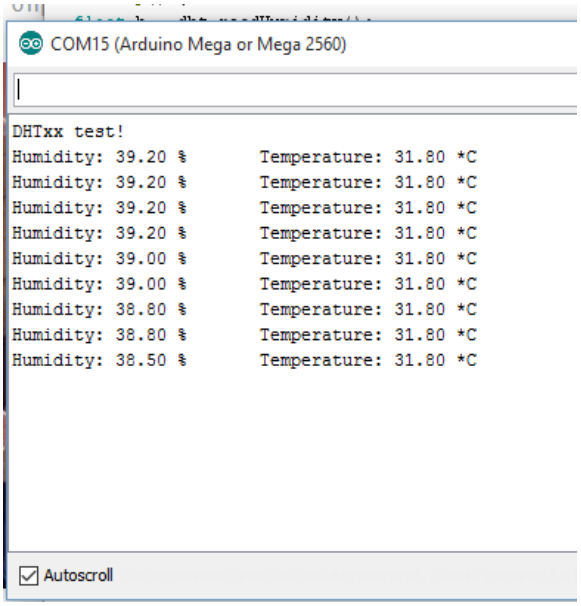
Το πρόγραμμα για τον αισθητήρα AM2301 προκειμένου να λαμβάνουμε συνεχώς μετρήσεις είναι το εξής:

```
// Το πρόγραμμα για τον αισθητήρα AM2301 προκειμένου να λαμβάνω συνεχόμενες μετρήσεις
// Γραμμένο από: ladyada, public domain

#include "DHT.h"
#define DHTPIN 2     // Καθορισμός του Pin2 σας είσοδο
#define DHTTYPE DHT21   // DHT 21 (AM2301)
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(9600);
  Serial.println("DHTxx test!");

  dht.begin();
}
void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(t) || isnan(h)) {
    Serial.println("Failed to read from DHT");
  } else {
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
    Serial.print(t);
    Serial.println(" *C");
  }
  Delay(1000);
}
```

Το παραπάνω πρόγραμμα ξεκινά κάνοντας include τη βιβλιοθήκη DTH . Στη συνέχεια καθορίζεται ο τύπος του αισθητήρα που χρησιμοποιείται, επιλέγεται η είσοδος του Arduino Mega που θα συνδεθεί με το Pin δεδομένων του αισθητήρα (κίτρινο/μπλε) καλώδιο, και στη συνέχεια εγκαθίσταται η επικοινωνία υπολογιστή και Arduino με ταχύτητα 9600 bit/sec. Εμφανίζεται στη σειριακή θύρα το μήνυμα «DHTxx test!» και ξεκινάει η χρήση του αισθητήρα όπου λαμβάνουμε μετρήσεις και τις αποθηκεύουμε σε μεταβλητές κινητής υποδιαστολής προκειμένου να τις απεικονίσουμε στη σειριακή θύρα. Στο τέλος των μετρήσεων προσθέτουμε καθυστέρηση της τάξης του ενός δευτερολέπτου αν και ο αισθητήρας για να μας δώσει νέα τιμή πρέπει να μεσολαβήσει διάστημα μεγαλύτερο των δύο δευτερολέπτων. Το αποτέλεσμα των μετρήσεων φαίνεται στην εικόνα που ακολουθεί:



Εικόνα 12: Λήψη Υγρασίας και θερμοκρασίας

2.3. Αισθητήρας Φωτός

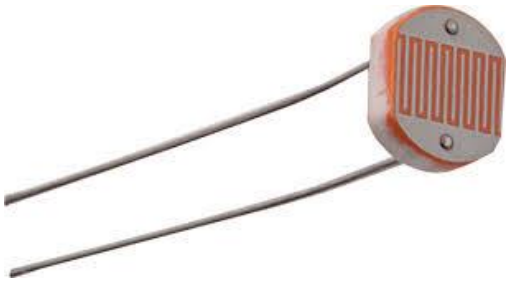
Μια ακόμα αίσθηση την οποία θα προσδώσουμε στην κατασκευή μας είναι αυτή της λήψης τιμής επιπέδου φωτεινότητας στον χώρο εγκατάστασης, αν και σε πρώτη φάση ο συγκεκριμένος αισθητήρας θα συλλέγει πληροφορίες χωρίς αυτές να αξιοποιούνται άμεσα ούτε στην κατασκευή αλλά ούτε και στην εφαρμογή μας.

Ο αισθητήρας που θα χρησιμοποιηθεί για τη λήψη αυτών των μετρήσεων είναι μία φωτοαντίσταση ή αλλιώς photo resistor αντίστασης 10 έως 20 ΚΩhm.

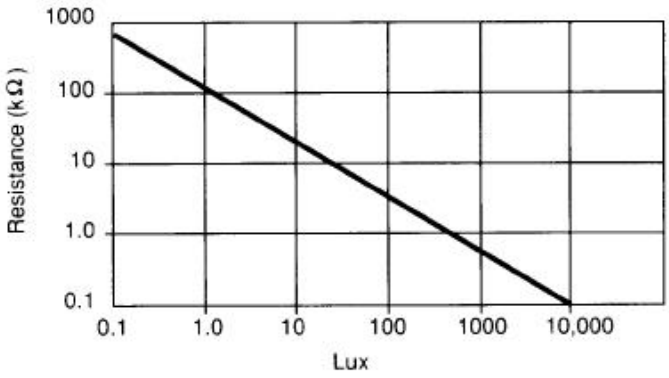
2.3.1. Τεχνικά Χαρακτηριστικά Αισθητήρα Φωτός

Ουσιαστικά πρόκειται για μία απλή αντίσταση μεταβλητής τιμής, όπου η τιμή της επηρεάζεται από τη φωτεινότητα του χώρου. Η συγκεκριμένη φωτοαντίσταση που θα χρησιμοποιηθεί έχει εύρος τιμών από 10 έως 20 ΚΩhm. Όταν σε αυτήν προσπίπτουν ακτίνες φωτός η τιμή της αντίστασης κυμαίνεται σε τιμές της τάξεως 10 ΚΩhm ενώ όταν ο χώρος στον οποίο βρίσκεται είναι σκοτεινός τότε το εύρος αντίστασης κυμαίνεται στα 20 ΚΩhm. Συνεπώς μπορούμε να καταλάβουμε μακροσκοπικά τα επίπεδα φωτεινότητας του χώρου εγκατάστασης. Η μέγιστη τάση λειτουργίας είναι τα 105 Volt και η μέγιστη ενεργειακή κατανάλωση είναι 100 mW. Έχει διαστάσεις 2x4x5 mm και τέλος εύρος θερμοκρασίας λειτουργίας από -30 oC έως 70 oC.

Πώς όμως μπορεί να μεταβάλλεται η τιμή της αντίστασης; Η φωτοαντίσταση είναι ένας ημιαγωγός κατασκευασμένος συνήθως από υλικά όπως σελήνιο (Se), θειούχο κάδμιο (CdS), ενώσεις του μολύβδου και του θείου, στον οποίο όταν προσπίπτει φωτεινή ακτινοβολία, ένα ποσοστό του φωτός ανακλάται στην επιφάνειά του, ένα άλλο ποσοστό εξέρχεται από τον ημιαγωγό και τέλος αυτό που μας ενδιαφέρει και εξετάζουμε είναι το ποσοστό το οποίο απορροφάται. Μεγαλύτερο ποσοστό απορροφημένου φωτός (συνεπώς μεγαλύτερη ένταση φωτός περιβάλλοντος) συνεπάγεται γραμμική μείωση της τιμής της αντίστασης όπως φαίνεται για παράδειγμα στο διάγραμμα που ακολουθεί.



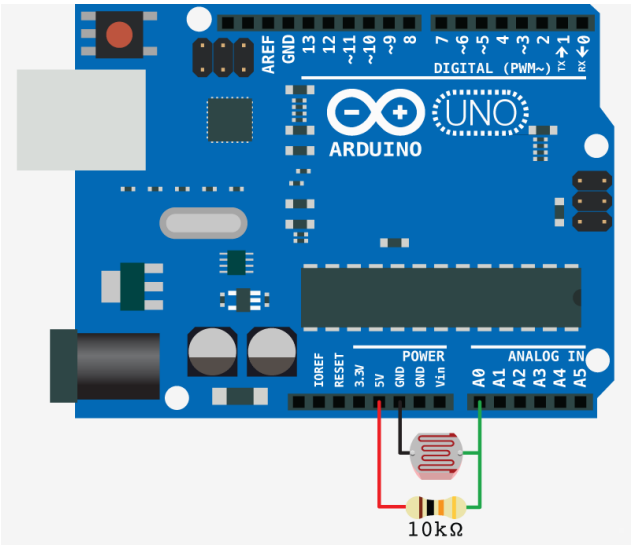
Εικόνα 13: Αναπαράσταση Φωτοαντίστασης με το φωτοαγώγιμο υλικό



Εικόνα 14: Μεταβολή της αντίστασης με την ένταση φωτεινής ακτινοβολίας

2.3.2. Διάγραμμα Συνδεσμολογίας Αισθητήρα Φωτός

Η συνδεσμολογία του Arduino με τον αισθητήρα φωτεινότητας ουσιαστικά υλοποιεί έναν διαιρέτη τάσης . Τροφοδοτούμε την αντίσταση από την έξοδο των 5 Volt του Arduino, διαβάζουμε μέσω μιας αναλογικής εισόδου (στην περίπτωσή μας από την είσοδο A0) και γειώνουμε το άλλο άκρο.



Εικόνα 15:Συνδεσμολογία Αισθητήρα Φωτός

2.3.3. Ανάλυση προγράμματος Αισθητήρα Φωτός

Το πρόγραμμα και σ' αυτή την περίπτωση είναι άρρηκτα συνδεδεμένο με τον τρόπο που έχει πραγματοποιηθεί η συνδεσμολογία. Στο παρακάτω πρόγραμμα φαίνεται η αρχικοποίηση της επικοινωνίας με ρυθμό 9600bps και έπειτα μέσα στην Main loop η ανάγνωση της αναλογικής θύρας A0 που έχουμε τον αισθητήρα. Έπειτα τυπώνουμε την τιμή αυτή στην οθόνη και για να προλαβαίνουμε να βλέπουμε τις τιμές βάζουμε καθυστέρηση της τάξης των 250 ms.

```
int LDR_Pin = A0;
void setup(){
  Serial.begin(9600);
}
void loop(){
  int LDRReading = analogRead(LDR_Pin);
  Serial.println(LDRReading);
  delay(250); //just here to slow down the output for easier reading
}
```

2.4. Πομποδέκτης RF

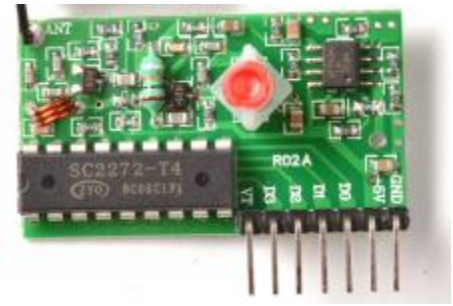
Η χρήση της κατασκευής μας απαιτεί κάποια μορφή τηλεχειρισμού μέσω του οποίου ο υπεύθυνος θα ενεργοποιεί ή θα απενεργοποιεί τους αισθητήρες όπως επίσης και το σύστημα του συναγερμού για να εισέλθει στο Server Room. Μέσω του τηλεχειριστηρίου θα μπορούσε να επιτρέπεται η πρόσβαση στον χώρο χωρίς τη χρήση κάποιας μορφής κωδικού εισόδου.

Εικόνα 16: Πομπός και Δέκτης RF

2.4.1. Τεχνικά Χαρακτηριστικά RF

Ο πομπός RF που στην περίπτωσή μας είναι το τηλεχειριστήριο στέλνει τέσσερις (4) διαφορετικούς παλμούς και αναλόγως ενεργοποιείται διαφορετική έξοδος στον δέκτη. Ο δέκτης μας είναι τύπου "RF M4 Receiver - 315MHz Momentary Type". Με το πάτημα του κουμπιού του πομπού ενεργοποιείται συγκεκριμένη έξοδος και παραμένει αναμμένη μέχρι το κουμπί του πομπού να αφεθεί.

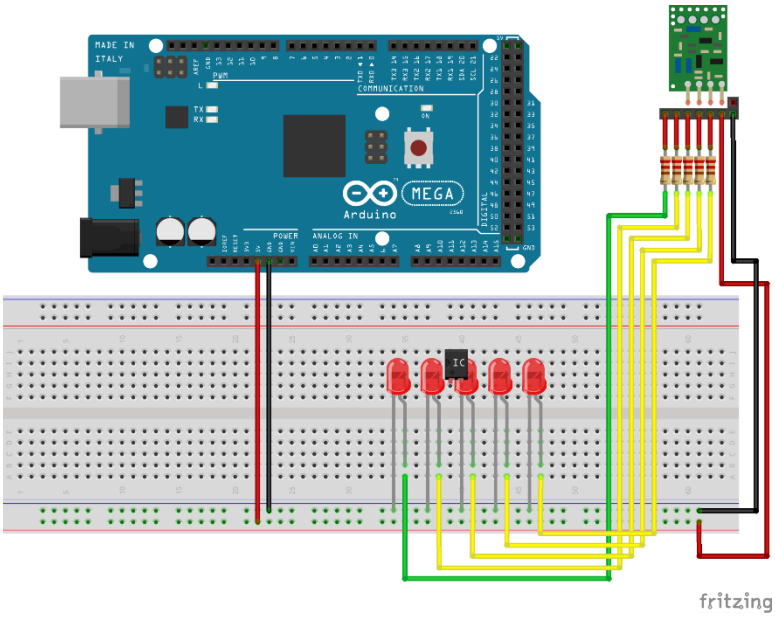
Ο δέκτης διαθέτει επτά (7) pin όπως φαίνεται και στην εικόνα 14 που ακολουθεί. Τα pin αντιστοιχούν από δεξιά προς αριστερά στη γείωση, την τροφοδοσία, την έξοδο A, την έξοδο B, την έξοδο C, την έξοδο D του τηλεχειριστηρίου και τέλος το pin ενεργοποίησης όταν δέχεται παλμό.



Εικόνα 17: RF M4 Receiver - 315MHz Momentary Type

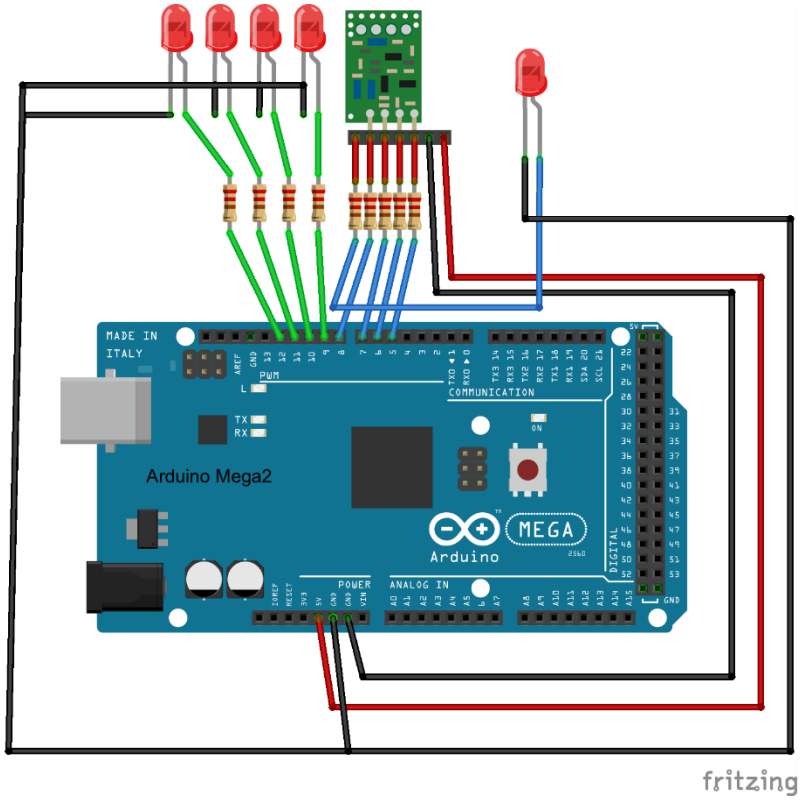
2.4.2. Διάγραμμα Συνδεσμολογίας RF Receiver

Η συνδεσμολογία που παρουσιάζεται στην εικόνα δεν αποτελεί την τελική συνδεσμολογία του δέκτη RF. Συγκεκριμένα οι έξοδοι συνδέονται με Led προκειμένου να γίνει κατανοητός ο τρόπος λειτουργίας της συγκεκριμένης πλακέτας. Οι έξοδοι του Receiver θα ενωθούν με εισόδους του Arduino Mega προκειμένου να ενεργοποιεί είτε να απενεργοποιεί μέσω του προγράμματος τις εκάστοτε λειτουργίες.



Εικόνα 18: Συνδεσμολογία RF Receiver

Το παρακάτω διάγραμμα αντιστοιχεί στη λειτουργία του Receiver οδηγώντας τις εξόδους του στις εισόδους του Arduino και εμφανίζοντας σχετικό μήνυμα κάθε φορά που πιέζουμε διαφορετικό πλήκτρο από το τηλεχειριστήριο, όπως επίσης και την αφή διαφορετικού χρώματος Led το οποίο έχουμε συνδέσει στο Arduino και όχι απευθείας στον αισθητήρα μας.



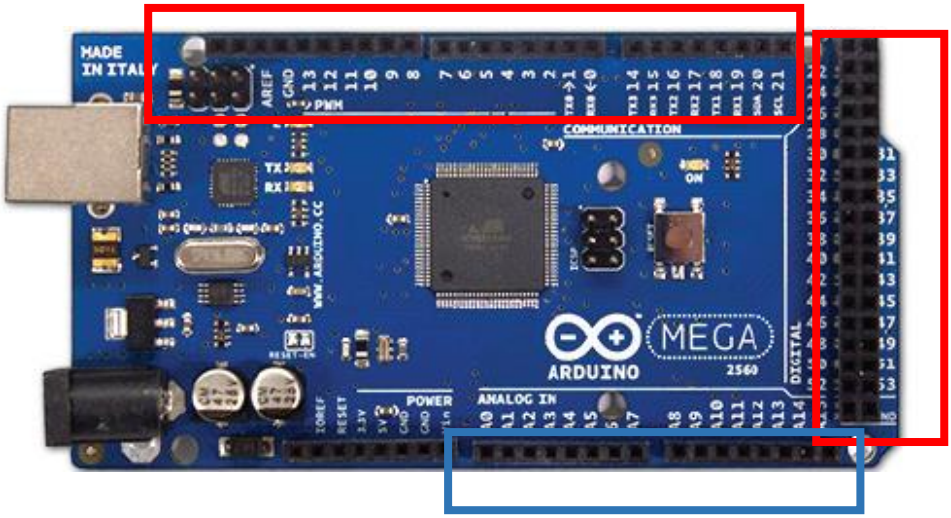
Εικόνα 19: Συνδεσμολογία RF Receiver με Mega και RGB Led.

Η διαφορά με τη συνδεσμολογία που παρουσιάστηκε στην εικόνα 18 είναι ότι σε αυτήν την περίπτωση τις εξόδους τις οδηγούμε σε pin του Mega τα οποία θα ορίσουμε σαν εισόδους. Με αυτό τον τρόπο μπορούμε να καταγράφουμε τις εντολές που παίρνουμε από το σύστημα τηλεχειρισμού και να «οδηγούμε» το πρόγραμμά μας σε εκτέλεση διαφορετικών ενεργειών. Στο παράδειγμα κώδικα που ακολουθεί θα αναπτυχθεί πρόγραμμα το οποίο θα εμφανίζει στην οθόνη του υπολογιστή διαφορετικό μήνυμα για κάθε εντολή που λαμβάνει και στη συνέχεια μέσω του Led θα ανάβει διαφορετικό χρωματισμό.

2.4.3. Ανάλυση Προγράμματος για τη Χρήση του RF Receiver

Το πρόγραμμα κάθε φορά που αναπτύσσεται είναι άρρηκτα συνδεδεμένο με τη συνδεσμολογία την οποία έχουμε υλοποιήσει. Στην εικόνα 19 βλέπουμε ότι τα Pin 5,6,7 και 8 του Arduino Mega είναι είσοδοι των εντολών που θα λαμβάνει από το RF Receiver ενώ το Pin 9, 10, 11 και 12 είναι έξοδοι προκειμένου να αναπαριστούμε χρωματικά τις εντολές που λαμβάνουμε.

Τα Pin 1 έως 53 όπως φαίνονται και στην εικόνα που ακολουθεί με κόκκινο πλαίσιο είναι ψηφιακά σε αντίθεση με τα Pin A0 έως A14 τα οποία είναι αναλογικά και επισημαίνονται με μπλε πλαίσιο.



Εικόνα 20: Arduino Mega 2560 PinOut

Επίσης, μπορούμε να ορίσουμε τη συμπεριφορά των digital pins ,δηλαδή αν θα δεχόμαστε είσοδο από το συγκεκριμένο Pin είτε θα το χρησιμοποιούμε σαν έξοδο. Οι καταστάσεις οι οποίες μπορεί να πάρει το Pin είτε σαν είσοδος είτε σαν έξοδος είναι δύο. Η κατάσταση High και η κατάσταση Low. Η πρώτη αντιστοιχεί στη λογική κατάσταση 1 (Boolean Logic) ενώ η δεύτερη κατάσταση αντιστοιχεί σε λογικό 0. Η τάση που θεωρείται λογικό 1 είναι από 3,7 ως 5 volt ενώ το λογικό 0 είναι οι τιμές τάσης κοντά στην περιοχή του μηδενός.

Για να ορίσουμε ένα Digital pin πρέπει εντός της συνάρτησης setup() να εκτελέσουμε την εντολή pinMode (Χ,Ψ), όπου το πρώτο όρισμα θα δηλώνει τον αριθμό του Pin και το δεύτερο αν θέλουμε να το θέσουμε είτε ως είσοδο είτε ως έξοδο. Συνεπώς για να κάνουμε τα Pins 5 ,6,7 και 8 εισόδους πρέπει να εκτελέσουμε τις εξής εντολές:

```
Void setup()
{
      pinMode(5, INPUT);
      pinMode(6, INPUT);
      pinMode(7, INPUT);
      pinMode(8, INPUT);
}
```

Και αντίστοιχα τα pins 9, 10, 11 και 12 ως έξοδοι :

```
void setup()
{
      pinMode(9, OUTPUT);
      pinMode(10, OUTPUT);
      pinMode(11, OUTPUT);
      pinMode(12, OUTPUT);
}
```

Προκειμένου να διαβάσουμε τις τιμές από τα Pins που έχουμε ορίσει ως εισόδους θα χρησιμοποιήσουμε την εντολή:

```
val = digitalRead(5);
```

Με την παραπάνω εντολή καταχωρούμε την τιμή που λαμβάνουμε από το Pin 5 στη μεταβλητή val.

Αντίστοιχα για να περάσουμε την τιμή που θέλουμε σε ένα digital pin που έχουμε ορίσει ως έξοδο εκτελούμε την εντολή:

```
digitalWrite(ledPin, val);
```

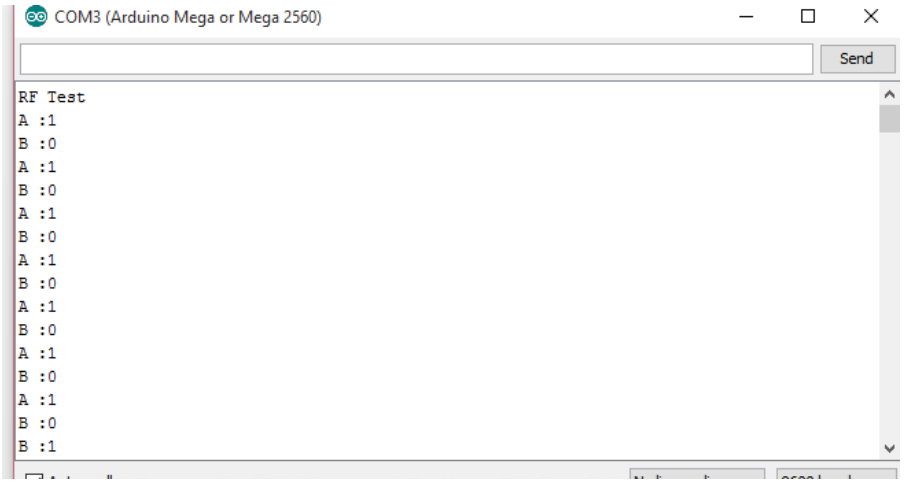
Όπου η πρώτη μεταβλητή στην περίπτωσή μας είναι τα Pins 9,10, 11 και 12 στα οποία περνάμε την τιμή που έχουμε στη μεταβλητή val (Δεύτερο όρισμα της συνάρτησης).

Το πρόγραμμα το οποίο παρατίθεται, σκοπό έχει τη λήψη του σήματος από το τηλεχειριστήριο, την αποθήκευση της επιλογής αυτής και, στην περίπτωση που πατήσουμε το κουμπί «A», θα ενεργοποιείται το Led A, θα απενεργοποιείται το Led B και θα εμφανίζεται ανάλογο μήνυμα. Αντίστοιχη θα είναι και η συμπεριφορά με τα κουμπιά του τηλεχειριστηρίου «C» και «D».

```
      #define A 5
      #define B 6
      #define C 7
      #define D 8
      // καθορισμός ονόματος των 5 ,6 ,7 και 8 σε A,B,C και D
      #define exodosA 9
      #define exodosB 10
      #define exodosC 11
      #define exodosD 12
      // καθορισμός ονόματος των 9, 10, 11 και 12 σε exodosA,
//exodosB, exodosC και exodosD
      //Αρχή της συνάρτησης setup
      void setup() {
        Serial.begin(9600);
      //καθορισμός ταχύτητας επικοινωνίας με
      //τον υπολογιστή για την εμφάνιση των μηνυμάτων
        Serial.println("RF Test"); //εμφάνιση μηνύματος
      // ορισμός των pins εισόδου
        pinMode(A, INPUT);
        pinMode(B, INPUT);
        pinMode(C, INPUT);
        pinMode(D, INPUT);
      // ορισμός των pins εξόδου
        pinMode(exodosA, OUTPUT);
        pinMode(exodosB, OUTPUT);
        pinMode(exodosC, OUTPUT);
        pinMode(exodosD, OUTPUT);
      }
      //αρχή της συνάρτησης loop
      void loop() {
      //διάβασμα των τιμών των Pins εισόδου και αποθήκευση αυτών σε
//μεταβλητές τύπου Boolean
      bool  valA = digitalRead(A);
      bool  valB = digitalRead(B);
      bool  valC = digitalRead(C);
      bool  valD = digitalRead(D);
      //Συνθήκες με τις ανάλογες εντολές η κάθε μία
      //Αν έχει πατηθεί το κουμπί A ενεργοποίησε το A και ταυτόχρονα
      //απενεργοποίησε το B
      //εμφάνισε στην οθόνη το A και την τιμή του που είναι 1 όλες
//τις φορές σε αυτή τη συνθήκη
        if (digitalRead(A)){
          digitalWrite(exodosA, HIGH);
          digitalWrite(exodosB, LOW);
          Serial.print("A ");
          Serial.print(valA);
```

```
    }
  //αντίστοιχο σετ εντολών με το προηγούμενο
 else if(digitalRead(B)){
      digitalWrite(exodosA, LOW);
      digitalWrite(exodosB, HIGH);
      Serial.print("B ");
      Serial.print(valB);
      }
    if (valC){
        digitalWrite(exodosC, HIGH);
        digitalWrite(exodosD, LOW);
        Serial.print("C ");
        Serial.print(valC);
    }
    else if(valD){
      digitalWrite(exodosC, LOW);
      digitalWrite(exodosD, HIGH);
      Serial.print("D ");
      Serial.print(valD);
      }
}
//τέλος προγράμματος
```

Στην παρακάτω εικόνα βλέπουμε τα αποτελέσματα από την εκτέλεση του παραπάνω προγράμματος πατώντας το πλήκτρο «A»:



Εικόνα 21: Αποτελέσματα που λαμβάνουμε από τη σειριακή Θύρα

2.5. Πιεζοδιακόπτες θυρών

Για τον έλεγχο και την ασφάλεια του χώρου απαιτείται ένα σύστημα ανάδρασης προκειμένου να μπορούμε να αντιληφθούμε την είσοδο ενός εισβολέα ή ενός μη εξουσιοδοτημένου ατόμου στο Server Room. Για τον σκοπό αυτό υπάρχουν πολλοί μέθοδοι που θα μπορούσαμε να ακολουθήσουμε. Θα μπορούσαμε να τοποθετήσουμε Radar κίνησης, αισθητήρα βάρους στην είσοδο, δέσμη lazer, μαγνητική παγίδα κλπ. Ένας από τους ευκολότερους και ταυτόχρονα

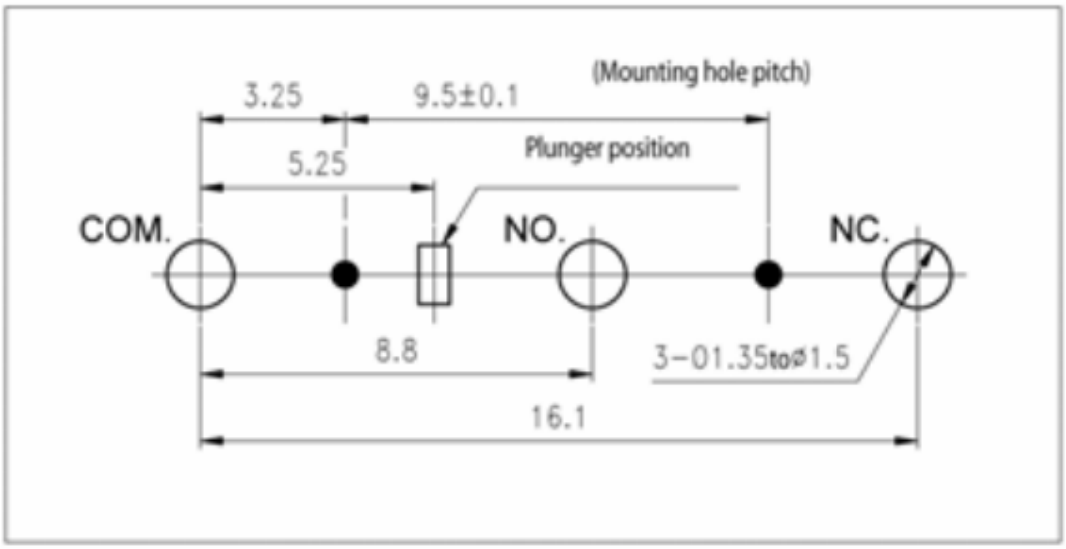
αξιόπιστους τρόπους είναι με τη χρήση πιεζοδιακόπτη τον οποίο θα τοποθετήσουμε στην είσοδο ή στις εισόδους του χώρου. Ουσιαστικά πρόκειται για έναν διακόπτη on-off. Μέσω του διακόπτη αυτού, θα παίρνουμε έναν ηλεκτρικό παλμό όταν η είσοδος ανοίξει και εκ των υστέρων, αν δεν έχει παρακαμφθεί νόμιμα, το σύστημα συναγερμού θα ενεργοποιείται και θα αποστέλλεται μήνυμα τόσο στο διαχειριστή όσο και στην εφαρμογή. Αν και δεν θα υλοποιηθεί στην εν λόγω εργασία , μπορούμε να κατασκευάσουμε ένα σύστημα συναγερμού το οποίο θα ενεργοποιείται από το Arduino.

2.5.1. Τεχνικά Χαρακτηριστικά Πιεζοδιακόπτη

Όπως αναφέρθηκε και στην εισαγωγή, για να αντιληφθούμε την είσοδο κάποιου στον χώρο θα χρησιμοποιήσουμε έναν πιεζοδιακόπτη. Η συμπεριφορά του και τα χαρακτηριστικά του δε διαφέρουν σε μεγάλο βαθμό από εκείνη ενός κουμπιού με τη διαφορά ότι έχει 3 ακροδέκτες. Ο πρώτος είναι για την τροφοδοσία, ο δεύτερος είναι ενεργοποιημένος – βραχυκυκλωμένος όταν ο διακόπτης είναι πιεσμένος και ο τρίτος ενεργοποιείται όταν ο διακόπτης αφεθεί. Πρακτικά θα χρησιμοποιήσουμε τον πρώτο και τον τρίτο ακροδέκτη. Στον πρώτο θα παρέχουμε μέσω της πλακέτας μας τροφοδοσία και από τον τρίτο θα λαμβάνουμε το σήμα όταν η πόρτα του Server Room ανοίξει.

Εικόνα 22: Πιεζοδιακόπτης

Βλέποντας επιγραμματικά τα χαρακτηριστικά του, αναφέρουμε ότι δέχεται μέγιστη τάση 125Volt και ένταση 5 A, έχει κύκλο ζωής 100000 φορές χρήσης το ηλεκτρικό του μέρος και 1000000 το μηχανικό του. Οι τιμές αυτές υπερκαλύπτουν τους στόχους μας. Το ηλεκτρικό του σχέδιο και οι διαστάσεις φαίνονται στην εικόνα που ακολουθεί.

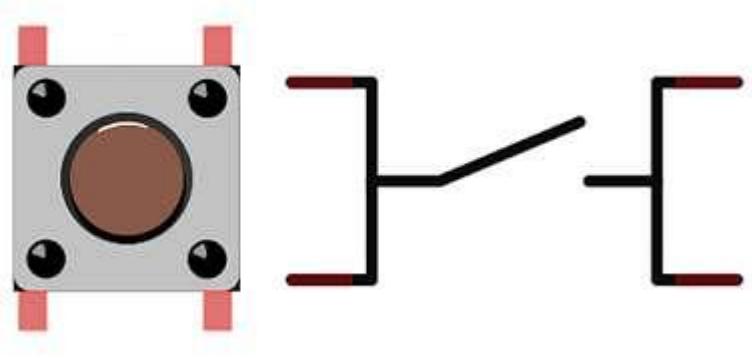


Εικόνα 23: Ηλεκτρικό Σχέδιο Πιεζοδιακόπτη

2.5.2. Διάγραμμα Συνδεσμολογίας Πιεζοδιακόπτη

Η συνδεσμολογία του διακόπτη που θα χρησιμοποιήσουμε δε διαφέρει στην αρχή λειτουργίας του με έναν απλό διακόπτη, εκτός της δυνατότητας που μας δίνει να μπορούμε να πάρουμε το σήμα, είτε όταν είναι πατημένος είτε ελεύθερος (on-off). Στην περίπτωσή μας θα επιλέξουμε να παίρνουμε σήμα όταν αυτός είναι σε θέση off, δηλαδή όταν είναι ελεύθερος. Αυτό σημαίνει πρακτικά ότι η πόρτα είναι ανοιχτή. Επειδή το πρόγραμμα Fritzing που χρησιμοποιούμε για τη σχεδίαση των sketch μας δεν έχει διακόπτη που να αντιστοιχεί πλήρως στο ηλεκτρικό σχέδιο της παραπάνω εικόνας, θα χρησιμοποιηθεί στο πρώτο παράδειγμα απλός διακόπτης και στο δεύτερο παράδειγμα διπλός διακόπτης, εξομοιώνοντας λειτουργικά το διακόπτη που έχουμε) .

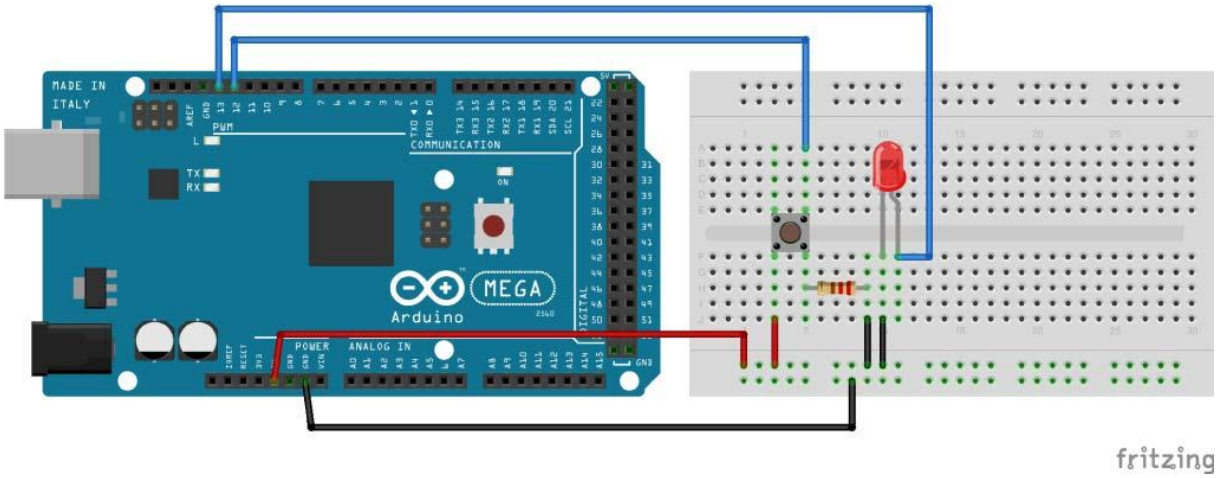
Το Ηλεκτρολογικό σχέδιο ενός απλού διακόπτη όπως αυτός που χρησιμοποιεί το πρόγραμμα έχει την εξής μορφή:



Εικόνα 24: Απλό Push Button

Αν χρησιμοποιήσουμε ένα τέτοιο Push Button προκειμένου να ανάβουμε και να σβήνουμε ένα led μέσω του Arduino, θα φτιάξουμε το κύκλωμα που παρουσιάζεται στην εικόνα που ακολουθεί. Να σημειωθεί ότι η λειτουργία του συγκεκριμένου διακόπτη δεν έχει σχέση με τον ηλεκτρολογικό διακόπτη αφού εδώ διαβάζουμε την κατάσταση του διακόπτη από το Pin 12 του

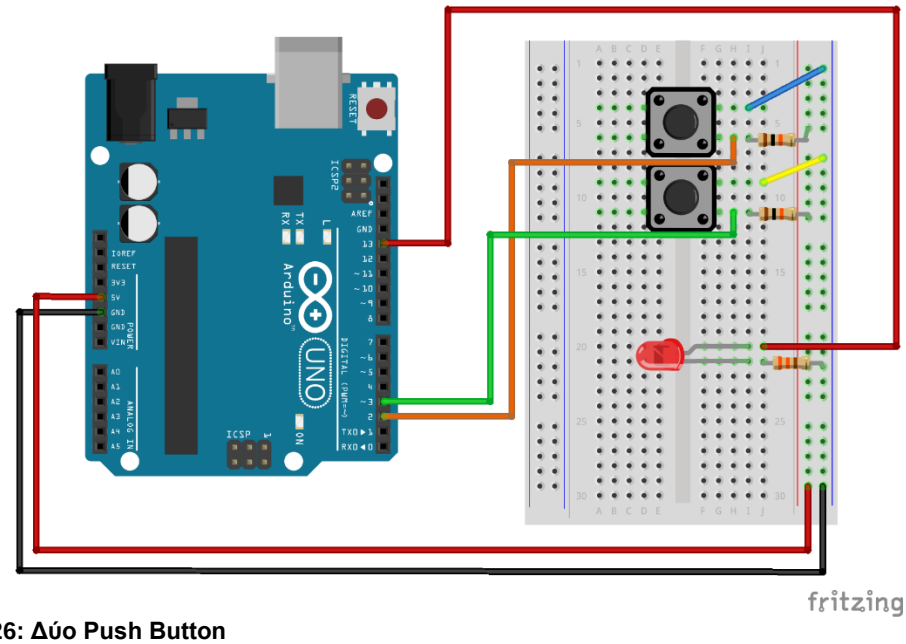
Arduino Mega και έπειτα μέσω του Pin 8 ενεργοποιούμε το Led. Συνεπώς δεν τροφοδοτεί άμεσα ο διακόπτης το Led και το τι θα κάνουμε όταν πατηθεί ο διακόπτης εξαρτάται από την υλοποίηση του κώδικά μας.



Εικόνα 25: Σύνδεση Arduino - Push Button & Led

Επίσης στο παραπάνω διάγραμμα χρησιμοποιήθηκε και μία αντίσταση τιμής 10ΚΩ.

Για να εξομοιώσουμε ακριβώς τη λειτουργία που θα έχει ο διακόπτης μας, όπως αναφέρθηκε και παραπάνω, υλοποιούμε το κύκλωμα που ακολουθεί:



Εικόνα 26: Δύο Push Button

2.5.3. Ανάλυση Προγράμματος για τη Χρήση του πιεζοδιακόπτη

Στο πρόγραμμα που ακολουθεί ορίζουμε το Pin που χρησιμοποιείται από το Arduino ως είσοδος που είναι το 12, το Pin 13 που χρησιμοποιείται σαν έξοδος στο Led και μία μεταβλητή buttonState για να

κρατάμε την κατάσταση του Push Button. Έπειτα διαβάζουμε την κατάσταση του Button και αναλόγως ανάβουμε είτε όχι το Led.

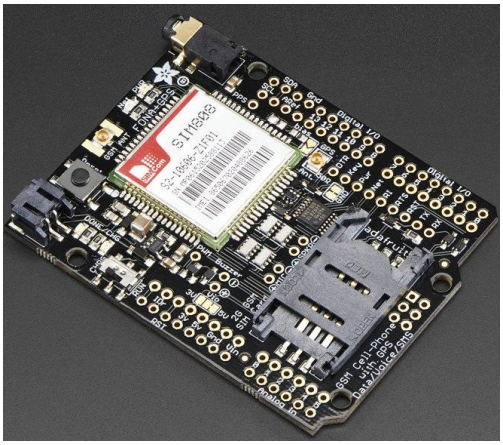
```
const int buttonPin = 12;
const int ledPin =  13;
int buttonState = 0;
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}
void loop() {
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

2.6. Δέκτης GPS - GSM

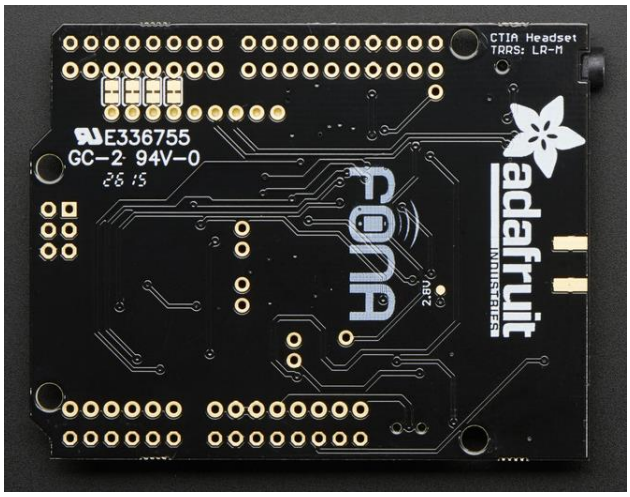
Βασικό shield του Computer Room Guard είναι το Fona 808. Μέσω αυτού, με τη χρήση μίας κάρτας κινητής τηλεφωνίας έχουμε τη δυνατότητα να λαμβάνουμε και να στέλνουμε μηνύματα, κλήσεις, να εντοπίζουμε τη θέση της κατασκευής μέσω των κυψελών κινητής τηλεφωνίας και τέλος να συνδέσουμε το εν λόγω Shield με δέκτη Gps ώστε να έχουμε μία δεύτερη πηγή γεωγραφικής θέσης. Αυτή τη γεωγραφική θέση μπορούμε να την αντιδιαστείλουμε με την πρώτη για ακριβέστερα αποτελέσματα και αναπληρωματικότητα.

Η δυνατότητα εντοπισμού θέσης δεν θα υλοποιηθεί στην εν λόγω κατασκευή αλλά αποτελεί ένα μελλοντικό πιθανό βήμα το οποίο θα μπορούσε να γίνει και θα πρόσθετε επιπλέον ασφάλεια και δυνατότητα εντοπισμού υλικών και συσκευών που έχουν κλαπεί.

Υπάρχουν πολλά Shield τα οποία θα μπορούσαμε να χρησιμοποιήσουμε για τη λήψη και αποστολή μηνυμάτων, κλήσεων, δεδομένων από και προς τη βάση και τέλος της γεωγραφικής θέσης αλλά αυτό που επιλέχθηκε ήταν της εταιρίας Adafruit και συγκεκριμένα το μοντέλο «Fona 808». Ο λόγος επιλογής της συγκεκριμένης πλατφόρμας είναι η δυνατότητα που προσδίδει με την προσθήκη δέκτη – κεραίας GPS να χρησιμοποιείται για τη λήψη της θέσης. Ταυτόχρονα με τη χρήση κεραίας GSM να μπορούμε να το χρησιμοποιήσουμε για αποστολή και λήψη δεδομένων μέσω του δικτύου κινητής τηλεφωνίας. Συνεπώς, με τη χρήση μίας πλατφόρμας επιτυγχάνεται ο διπλός στόχος της κατασκευής. Το κόστος της πλατφόρμας παρόλα αυτά είναι αυξημένο και κυμαίνεται στα 70 €, μη συμπεριλαμβανομένου του επιπλέον εξοπλισμού που απαιτείται, ο οποίος είναι η κεραία λήψης GPS, η κεραία GSM και εξωτερική πηγή τροφοδοσίας.



Εικόνα 27 : FONA 808 Εμπρός Όψη



Εικόνα 28: FONA 808 Πίσω Όψη

2.6.1. Τεχνικά Χαρακτηριστικά FONA 808

Όπως αναφέρθηκε, το FONA 808 είναι ένα Module το οποίο μπορεί να χρησιμοποιηθεί τόσο σαν GPS όσο και για σύνδεση με δίκτυα GSM με κάρτα SIM 2G. Το Fona 808 στηρίζεται στο ολοκληρωμένο SIM808 με την προσθήκη κυκλώματος για το GPS. Λεπτομερέστερα τα τεχνικά χαρακτηριστικά του είναι τα εξής:

1. Σύνδεση σε δίκτυα Quad-band 850/900/1800/1900MHz
2. Λειτουργία GPS με ευαισθησία -165 dBm βασιζόμενο στο MT3337 chipset με προσεγγιστική ακρίβεια 2,5 μέτρων.
3. Δυνατότητα αποστολής και λήψης ήχου με χρήση ηλεκτρολητικού μικροφώνου και ηχείου αντίστασης 32Ω.
4. Αποστολή λήψη SMS, GPRS Data
5. Δυνατότητα σύνδεσης buzzer (vibration motor)
6. Σύνδεση uFL με τις κεραίες GPS και GSM
7. Ενδεικτικό Led για λειτουργία και σύνδεση στο δίκτυο
8. Ενσωματωμένο κύκλωμα φόρτισης μπαταρίας τύπου Li-Po 500mAh και άνω (Στο κύκλωμα της εφαρμογής μας χρησιμοποιούμε 4000mAh)
9. Διαστάσεις 69.0mm x 54.0mm x 4.0mm
10. Βάρος 19.0gr

2.6.2. Ανάλυση πλακέτας Fona 808

Το Fona 808 είναι μία πλακέτα που ενσωματώνει σχεδόν το σύνολο των δυνατοτήτων ενός κινητού τηλεφώνου συμπεριλαμβάνοντας το ολοκληρωμένο κύκλωμα του GPS (διαφοροποίηση από το Fona 800 που δε διαθέτει GPS). Πρόκειται για μία μικρών διαστάσεων πλακέτα μόλις 24*24*2,6 mm και βάρους 3,5gr. Τροφοδοτείται με τάση 3,4 έως 4,4 Volt από μπαταρία τύπου Li-Ion είτε Lipo. Έχει κατανάλωση 1 mA σε κατάσταση αδράνειας (sleep mode), ενώ σε κατάσταση λειτουργίας διαφοροποιείται βάση των λειτουργιών που χρησιμοποιούμε. Για τον έλεγχο της κατανάλωσης

υπάρχουν τρία επίπεδα λειτουργίας. Το πρώτο επίπεδο “Normal Operation”, το δεύτερο “Power Down” και τέλος το “Minimum functionality mode”.

Normal Operation:

Χωρίζεται σε πέντε επιμέρους καταστάσεις οι οποίες είναι οι εξής:

1. GSM/GPRS SLEEP: Χαμηλή ενεργειακή κατανάλωση, απενεργοποιημένες οι περισσότερες συμπεριλαμβανομένων και των interrupts και των σειριακών θυρών αλλά παραμένει η δυνατότητα λήψης SMS.
2. GSM IDLE: Ενεργοποιημένο το σύστημα, συνδεδεμένο με το δίκτυο και σε κατάσταση αναμονής έτοιμο για επικοινωνία.
3. GSM TALK: Σε αυτή την περίπτωση το σύστημα είναι ενεργοποιημένο και βρίσκεται σε κατάσταση σύνδεσης και επικοινωνίας.
4. GPRS STANDBY: Το σύστημα είναι έτοιμο για μεταφορά δεδομένων GPRS αλλά δεν αποστέλλονται είτε λαμβάνονται ακόμα δεδομένα. Η κατανάλωση σ’ αυτή την περίπτωση εξαρτάται από τις ρυθμίσεις του δικτύου που είμαστε όπως περιγράφονται παρακάτω.
5. GPRS DATA: Υπάρχει εξέλιξη μεταφοράς δεδομένων GPRS.
6. Charge: Το κύκλωμα έχει δυνατότητα υποστήριξης διαδικασίας φόρτισης.

Power Down: Είναι η κατάσταση φυσιολογικής απενεργοποίησης του συστήματος με AT command. Σ’ αυτή την κατάσταση το λογισμικό είναι απενεργοποιημένο και η πλειοψηφία του hardware είναι απενεργοποιημένη με εξαίρεση του RTC (Real Time Clock) το οποίο παραμένει. Οι σειριακές πόρτες είναι απενεργοποιημένες.

Minimum Function Mode: Σ’ αυτή την κατάσταση η κάρτα SIM όπως και η κεραία είναι απενεργοποιημένες και οι σειριακές θύρες είναι κλειστές. Η ενεργειακή κατανάλωση σ’ αυτή τη λειτουργία είναι χαμηλότερη σε σχέση με την κατάσταση normal operation.

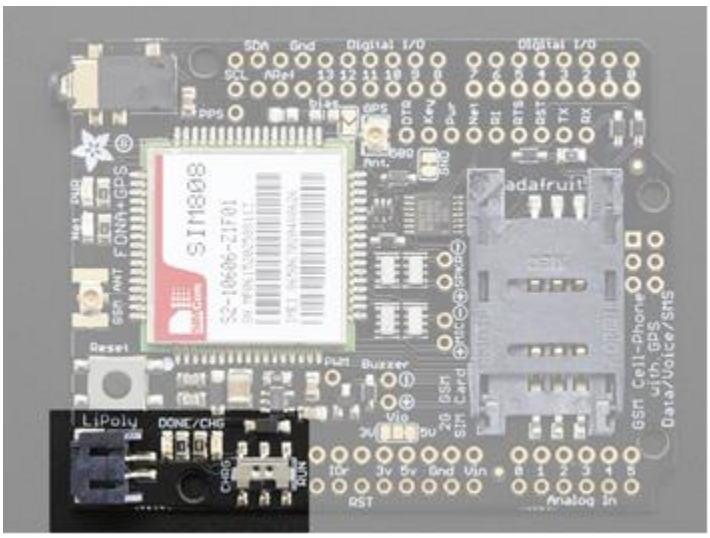
Το Fona 808 εκπέμπει στις συχνότητες GSM 850MHz, EGSM 900 MHz, DCS 1800 MHz και EGSM 900 MHz. Η εκπεμπόμενη ισχύς σε συχνότητες GSM850 MHz και EGSM 900 MHz κυμαίνεται στα 2W, ενώ σε συχνότητες DCS 1800 MHz και EGSM 900 MHz στο 1W.

Η πλακέτα λειτουργεί σε εύρος θερμοκρασιών από τους -40 oC έως 85 ^{o}C με ιδανικό εύρος τιμών από -30 oC έως και 80 oC καθιστώντας την ιδανική για την κατασκευή μας. Σε περίπτωση υπέρβασης των ορίων -30 oC έως και 80 ^{o}C λαμβάνεται σχετικό μήνυμα, ενώ σε περίπτωση υπέρβασης των ορίων -40 oC έως 85 ^{o}C απενεργοποιείται η πλακέτα για λόγους προστασίας.

Άλλη δυνατότητα που διαθέτει είναι η σειριακή επικοινωνία με ρυθμούς μετάδοσης από 1200bps έως 115200bps (για την εφαρμογή μας χρησιμοποιούμε ρυθμό 9600bps).

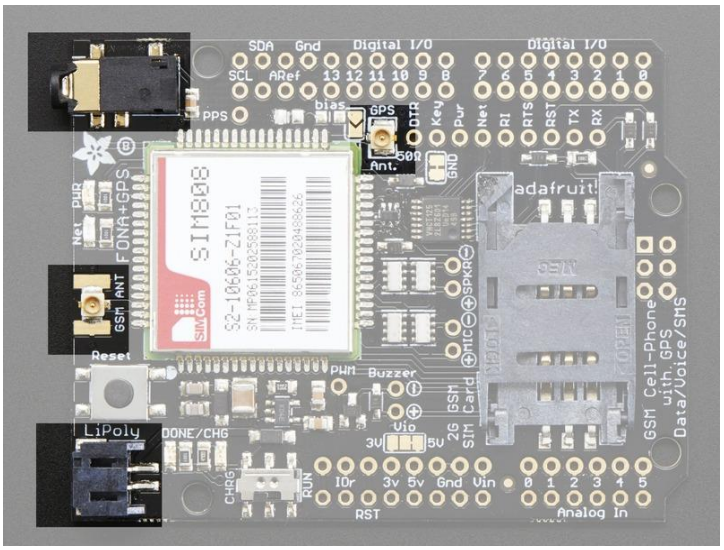
Ο ρυθμός μετάδοσης δεδομένων μπορεί να φτάσει το μέγιστο με GPRS τα 85,6 kbps (up/down).

Στη συνέχεια θα αναλύσουμε τμηματικά την πλακέτα προκειμένου να δούμε χωριστά τα κυκλώματα και κατ' επέκταση τη συνδεσμολογία που θα πραγματοποιηθεί με την εν λόγω πλακέτα.



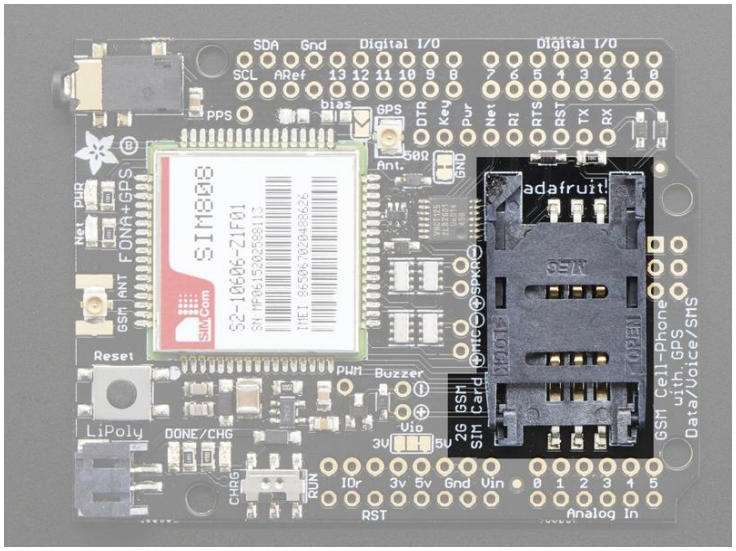
Εικόνα 29: Κύκλωμα Τροφοδοσίας

Στην αριστερή πλευρά της πλακέτας υπάρχουν τρεις εξωτερικοί connectors. Πάνω αριστερά είναι η υποδοχή για τα ακουστικά. Είναι διαστάσεων 3,3mm με δυνατότητα σύνδεσης τόσο με stereo όσο και με mono ακουστικά. Στη μέση αριστερά είναι τοποθετημένος ο connector της κεραίας κινητής τηλεφωνίας (GSM Ant) τύπου uFL. Κάτω αριστερά βλέπουμε την υποδοχή τροφοδοσίας της μπαταρίας Lipo είτε Li-Ion που θα τροφοδοτεί επικουρικά την πλακέτα μας. Τέλος στη μέση βλέπουμε άλλο ένα connector τύπου uFl μέσω του οποίου συνδέουμε την κεραία του GPS.



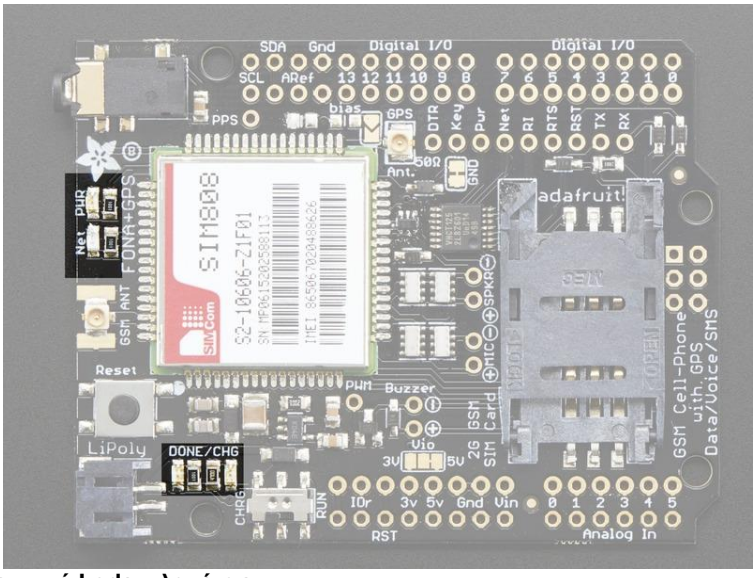
Εικόνα 30: Headset jack, uFL RF connector, JST 2-pin, uFL GPS connector

Στο κάτω αριστερά μέρος βλέπουμε τον 2 pin connector της μπαταρίας με δύο ενδεικτικά led τα οποία υποδηλώνουν την κατάσταση στην οποία βρίσκεται το κύκλωμα, δηλαδή αν είμαστε σε κατάσταση φόρτισης το led CGH θα είναι αναμμένο ενώ αν η φόρτιση έχει τελειώσει το led Done θα ανάψει και το άλλο θα σβήσει. Για την επιλογή της φόρτισης είτε της λειτουργίας του κυκλώματος η επιλογή γίνεται από τον διακόπτη CHRG/RUN. Πέρα από αυτό τον τρόπο φόρτισης μπορούμε να φορτίσουμε την μπαταρία μέσω του SIM808. Η φόρτιση γίνεται μέσω του Pin 5Volt του Arduino στο οποίο έχουμε συνδέσει την πλακέτα μας με ρυθμό φόρτισης 200mA.



Εικόνα 31: Sim Card Slot

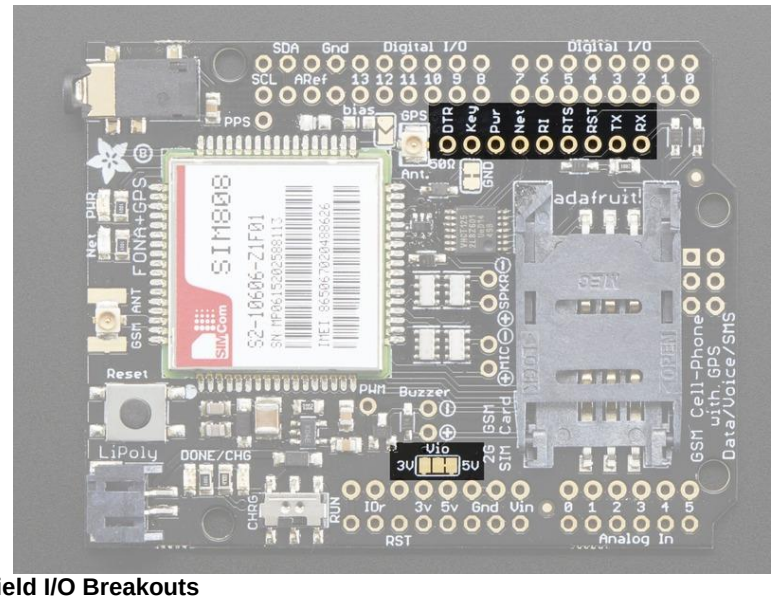
Για να χρησιμοποιήσουμε το GSM δίκτυο το μόνο που χρειαζόμαστε πέρα από την κεραία είναι μία κάρτα 2G την οποία θα τοποθετήσουμε στην είσοδο όμοια με αυτή του τηλεφώνου μας που φαίνεται στην εικόνα 23.



Εικόνα 32: Ενδεικτικά Leds πλακέτας

Στην αριστερή πλευρά της πλακέτας υπάρχουν άλλα δύο led πέρα αυτών που περιγράφηκαν στο κύκλωμα φόρτισης. Το πρώτο led PWR ανάβει πράσινο όταν έχουμε τοποθετήσει την κάρτα Sim και το module λειτουργεί κανονικά. Μέσω δεύτερου led χρώματος μπλε μπορούμε

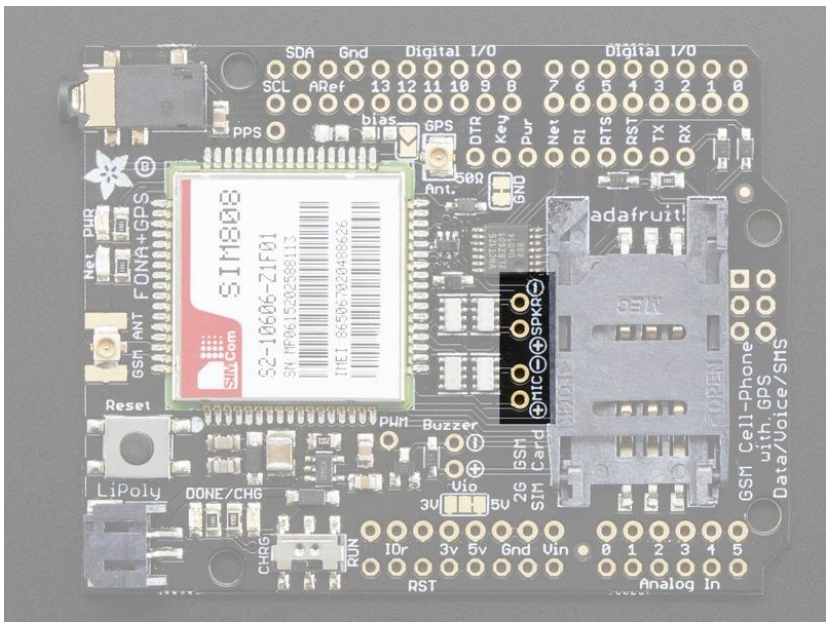
να καταλάβουμε την κατάσταση στην οποία βρίσκεται η σύνδεσή μας. Συγκεκριμένα όταν είναι αναμμένο για 64ms και σβηστό για 800 ms το Module είναι αναμμένο αλλά δεν έχει γίνει σύνδεση με το δίκτυο κινητής τηλεφωνίας, όταν το led είναι αναμμένο για 64 ms και σβηστό για 3 sec τότε υπάρχει σύνδεση με το δίκτυο και υπάρχει δυνατότητα αποστολής και λήψης φωνής και SMS. Τέλος όταν είναι αναμμένο για 64 ms και σβηστό για 300 ms είναι ενεργοποιημένο και συνδεδεμένο το GPRS.



Εικόνα 33: Shield I/O Breakouts

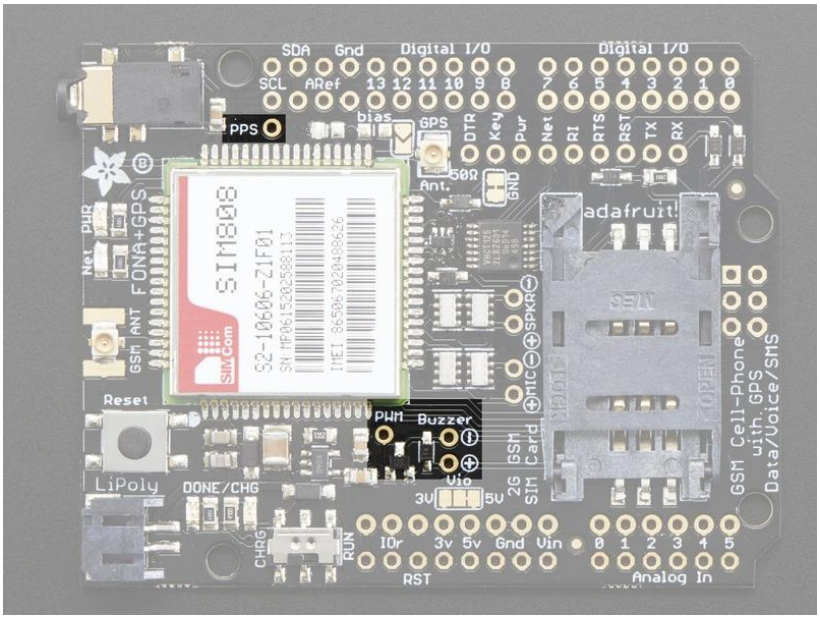
Στην εικόνα 25 βλέπουμε βλέπουμε τα pins που χρησιμοποιούνται για την επικοινωνία της πλακέτας του Fona με αυτή του Arduino. Αυτά τα pins χρησιμοποιούν τάσεις 3 έως 5 volt. Προεπιλεγμένη τιμή είναι τα 5 Volt τα οποία χρησιμοποιεί και η πλακέτα του Arduino. Για να αλλάξουμε αυτή την επιλογή θα πρέπει να χαράξουμε – κόψουμε το βραχυκύκλωμα που φαίνεται στο κάτω αριστερά μέρος της κάρτας Sim της εικόνας 25 και να το βραχυκυκλώσουμε με την επιλογή των 3 Volt. Επεξηγώντας τα pins από δεξιά προς αριστερά έχουμε, τα RX και TX τα οποία χρησιμοποιούνται για την αποστολή και λήψη δεδομένων. Ο ρυθμός μετάδοσης ρυθμίζεται αυτόματα. Τα συγκεκριμένα Pins είναι συνδεδεμένα με τα 2 και 3 του Arduino μέσω των digital pins που βρίσκονται ακριβώς στο πάνω μέρος τους. Τρίτο κατά σειρά είναι το RST pin δηλαδή το RESET pin το οποίο προεπιλεγμένη τιμή του είναι σε λογικό 1 High. Αν εφαρμόσουμε παλμό λογικού μηδέν – γειώσουμε το Pin για χρόνο μεγαλύτερο των 100ms τότε κάνουμε hard reset στην πλακέτα. Το Pin RST είναι συνδεδεμένο με το Digital pin 4 συνεπώς μέσω του Arduino μπορούμε να κάνουμε reset αν υπάρξει κάποιο πρόβλημα με την πλακέτα μας. Το τέταρτο κατά σειρά Pin είναι το RTS (Ready To Send) μέσω του οποίου μπορούμε να ελέγχουμε πόσο γρήγορα είτε αργά μεταφέρονται data από το Arduino. Το πέμπτο pin είναι το RI (Ring Indicator). Το συγκεκριμένο pin δεν είναι συνδεδεμένο εξ αρχής με το Arduino. Μέσω αυτού μπορούμε να καταλαβαίνουμε εισερχόμενες κλήσεις είτε ακόμα εισερχόμενο SMS. Η κατάστασή του είναι μόνιμα σε λογικό ένα High και όταν λαμβάνεται κλήση μεταβαίνει σε κατάσταση λογικού μηδέν – Low για χρονικό διάστημα 120ms. Έκτο Pin είναι το Net μέσω του οποίου μπορούμε να καταλάβουμε την κατάσταση στην οποία βρίσκεται η σύνδεση του δικτύου. Η λειτουργία του είναι ίδια με αυτή του led Net με τη διαφορά ότι το ένα μας δίνει οπτική απεικόνιση της κατάστασης ενώ το άλλο μπορούμε να το χρησιμοποιήσουμε σαν είσοδο στην πλακέτα του Arduino και να προβαίνουμε σε διαφορετικές ενέργειες αναλόγως της κατάστασης στην οποία είναι το δίκτυο. Έβδομο κατά σειρά pin είναι το Pwr το οποίο είναι το Power Status Pin. Είναι σε κατάσταση μηδέν Low όταν η πλακέτα είναι απενεργοποιημένη και σε High όταν τροφοδοτείται – λειτουργεί. Την κατάσταση του PWR pin μπορούμε να τη δούμε και στο αντίστοιχο Led. Σε συνδυασμό με το επόμενο Pin το Key μπορούμε να κάνουμε monitor τη λειτουργία της πλακέτας. Τέλος το Pin Key χρησιμοποιείται για να ενεργοποιήσουμε είτε να απενεργοποιήσουμε την πλακέτα. Δεν ακολουθεί τη λογική των προηγούμενων pin. Δηλαδή για να ενεργοποιήσουμε την πλακέτα από

λογικό High εφαρμόζουμε παλμό Low για 2 sec και η πλακέτα αλλάζει κατάσταση. Αν είναι ενεργοποιημένη απενεργοποιείται και το αντίθετο.



Εικόνα 34: Audio Pins

Όπως αναφέρθηκε και στην αρχή η πλακέτα του Fona 808 μπορεί να χρησιμοποιηθεί και να ενσωματώσει σχεδόν το σύνολο των λειτουργιών ενός κινητού τηλεφώνου. Πέραν των ακουστικών και μικροφώνου που μπορούν να συνδεθούν στο πάνω αριστερά μέρος της πλακέτας όπως είδαμε και παραπάνω υπάρχει η δυνατότητα να συνδέσουμε στα SPKR Speaker pins εξωτερικό μεγάφωνο αντίστασης 32 Ωhm και 0,2W. Η διαφορά με τα ακουστικά που συνδέουμε στην άλλη θύρα είναι ότι σ' αυτή δεν μπορούμε να συνδέσουμε stereo. Δεύτερο Pin στη εικόνα 26 είναι αυτά του μικροφώνου.



Εικόνα 35: Buzzer PWM και PPS Pins

Άλλα Pins και επιπλέον δυνατότητες τις οποίες μας δίνουν είναι το Buzzer pin που φαίνεται στο κέντρο της πλακέτας στο οποίο μπορούμε να συνδέσουμε ένα μικρό vibration motor (μικρού τύπου κινητήρας που προκαλεί δόνηση όμοια με αυτή του κινητού). Από Pin PWM Pulse Width Modulation

μπορούμε να πάρουμε παλμό συγκεκριμένης περιόδου το οποίο θα μπορούσαμε να τον χρησιμοποιήσουμε σαν παλμό συγχρονισμού μεταξύ δύο συσκευών και τέλος το PPS pin το οποίο μας δίνει ένα παλμό ανά δευτερόλεπτο από το κύκλωμα του GPS. Από αυτό το Pin το λογικό ένα δεν έχει την τιμή 5Volt αλλά την τιμή 2,5Volt με αποτέλεσμα να μην μπορεί να χρησιμοποιηθεί απευθείας στην πλακέτα του Arduino αλλά να μετατρέψουμε τον παλμό μέσω ενός λογικού Swifter.

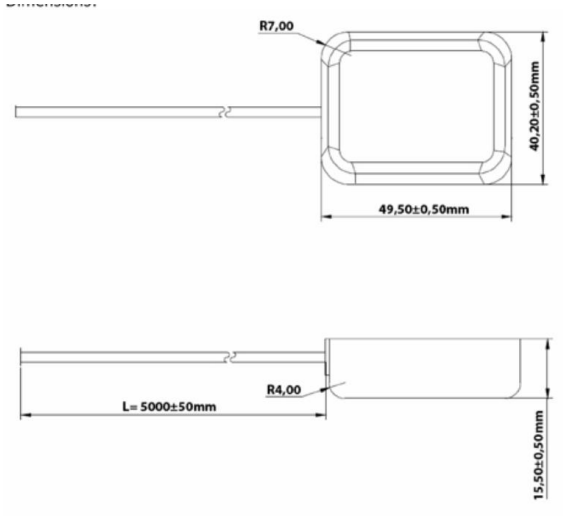
2.6.3. Τεχνικά Χαρακτηριστικά Λοιπού Εξοπλισμού

Όπως αναφέρθηκε η πλακέτα του Fona 808 απαιτεί τη χρήση επιπλέον εξαρτημάτων αναλόγως τις λειτουργίες τις οποίες θέλουμε να προσδώσουμε στο κύκλωμα. Συνεπώς για την εφαρμογή μας χρησιμοποιούμε επίσης κεραία GPS, κεραία GSM, connector Ufl, κάρτα SIM 2G και τέλος μπαταρία Li-Po 4000mAh.

Η κεραία GPS που επιλέχθηκε είναι η «Bluebird 3-5 Volt GPS Antenna» συχνότητας 1575,42MHz, με conector SMA-A καλώδιο σύνδεσης 5 μέτρων που προσδίδει ευκολία και άνεση στην τοποθέτηση στην εφαρμογή μας, δυνατότητα τοποθέτησης και σε εξωτερικό χώρο. Το κόστος της κυμαίνεται στα 15 €. Έχει δυνατότητα σύνδεσης με τους κάτωθι δορυφόρους: Alpine Blackbird PMD-B100, GlobalSat AT-65-SMA, GlobalSat G5010U, GlobalSat G5040, GlobalSat TT-300, Holux GM 82 Engine Board, Motorola 361, Motorola digital radio-telephone DM341, Navman Tracker 5100, Navman Tracker 5100i, Navman Tracker 5110, Navman Tracker 5110i, Navman Tracker 5380, Navman Tracker 5380i, Navman Tracker 5430, Navman Tracker 5430i, Navman Tracker 5500, Navman Tracker 5500i, Navman Tracker 5600, Navman Tracker 5605, Rikaline AR-10S GPS SMA Antenna 1001414, Rikaline Active Antenna SMA A-10302-SA, Rikaline CP 160, Rikaline CP 170, Rikaline GPS-300 Black Box, Rikaline SMA A-10305-SA, Rikaline Standard Horizon CP 150, Rikaline Synergy Systems AR-10K GPS Antenna SMA, Trimble TNL1000 Aviation GPS. Η επιλογή της εν λόγω κεραίας έγινε τόσο για την αντοχή της, τη δυνατότητα έκθεσης σε εξωτερικό χώρο με δυνατότητα μετρήσεων σε εύρος θερμοκρασιών από -20 oC έως 89 oC, την καταλληλόλητα τροφοδοσίας 3- 5 Volt τα οποία λαμβάνει από την πλακέτα του Fona, την ευελιξία στην τοποθέτηση που μας παρέχει λόγω του μεγάλου μήκους καλωδίου της, τη μεγάλη ακρίβεια και τέλος όλα αυτά τα χαρακτηριστικά σε λογικό κόστος.

Εικόνα 36: Arduino Mega Εμπρός Όψη – Σειριακές Θύρες

Οι διαστάσεις της κεραίας καλύπτουν τις ανάγκες της εφαρμογής μας αν και δεν είναι περιοριστικές λόγω της τοποθέτησης αυτής σε εξωτερικό χώρο.

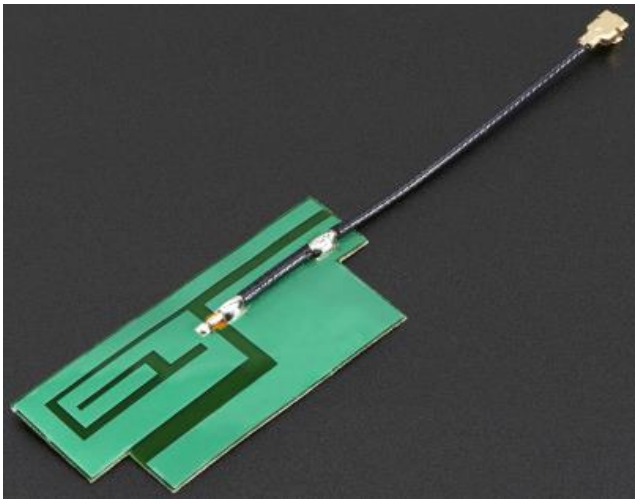


Εικόνα 37 : Διαστάσεις κεραίας GPS

Για τη σύνδεση του GPS με την πλακέτα του Fona 808 απαιτείται η χρήση προσαρμογέα – αντάπτορα Ufl σε SMA όπως φαίνεται στην εικόνα που ακολουθεί.

Εικόνα 38: Adaptor Ufl to SMA

Για την επίτευξη της επικοινωνίας με το δίκτυο κινητής τηλεφωνίας απαιτείται η χρήση κεραίας GSM και μίας κάρτας SIM 2G. Η κεραία η οποία χρησιμοποιήθηκε είναι η «Slim Sticker-type GSM/Cellular Quad-Band Antenna - 3dBi uFL». Το κόστος της κυμαίνεται στα 5 €.



Εικόνα 39 : Κεραία GSM

Εναλλακτικά αυτής της κεραίας θα μπορούσε να χρησιμοποιηθεί κεραία όπως της εικόνας που ακολουθεί επιτυγχάνοντας ελαφρώς καλύτερα αποτελέσματα.

Εικόνα 40: Right-angle Mini GSM/Cellular Quad-Band Antenna - 2dBi SMA Plug

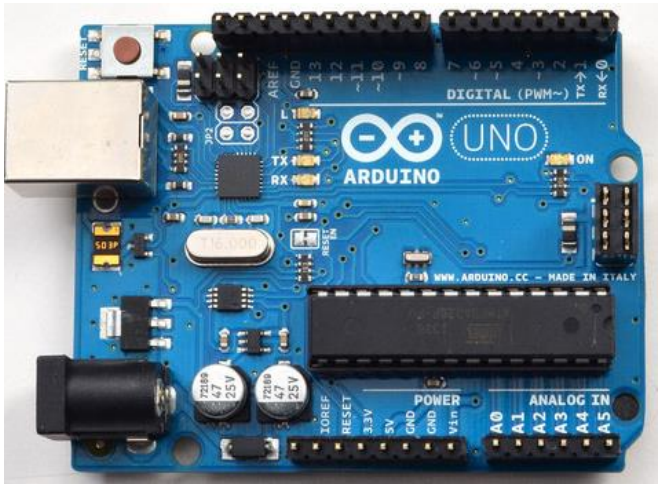
Τέλος τελευταίο εξάρτημα είναι η μπαταρία η οποία απαιτείται για τη χρήση του Fona 808. Για να πετύχουμε μεγάλη διάρκεια ζωής της μπαταρίας χρησιμοποιήθηκε μπαταρία Li-Po 4000 mAh η οποία σε συνδυασμό με την αδράνεια στην οποία θα θέτουμε τη συσκευή για τις περιόδους μη χρήσης επιτυγχάνουμε μεγάλη διάρκεια ζωής. Επίσης μέσω της πλακέτας του Fona υπάρχει η δυνατότητα επαναφόρτισης της μπαταρίας και επαναχρησιμοποίησή της. Η τάση λειτουργίας της είναι 3,7 Volt, έχει ενσωματωμένο κύκλωμα προστασίας. Οι διαστάσεις της είναι 7 x 35 x 138mm. Το κόστος της μπαταρίας λόγω του τύπου της και της μεγάλης χωρητικότητας κυμαίνεται στα 20 €.

Εικόνα 41: Polymer Lithium Ion Battery - 3.7v 4000mAh

2.6.4. Διάγραμμα Συνδεσμολογίας

Το Fona 808 λόγω του ότι είναι πλακέτα σχεδιασμένη με τρόπο ώστε να προσαρμόζεται σε αυτή του Arduino απαιτεί μόνο την τοποθέτησή της στα ανάλογα Pin του Arduino. Συγκεκριμένα η πλακέτα θα καταλάβει το σύνολο των pins που είναι κοινά στο Arduino Mega και Uno δηλαδή τα Digital pins 0 έως 13, τα αναλογικά Pins 0 έως 5 και τα pins, τροφοδοσίας τάσης, γείωσης και reset. Στην παρακάτω φωτογραφία φαίνεται η πλακέτα του Arduino Uno και έπειτα η πλακέτα του Arduino Mega με επισήμανση των Pin των οποίων θα δεσμευτούν για τη χρήση του Fona808. Στη συνέχεια κατά την περιγραφή της ανάπτυξης της κατασκευής θα χρησιμοποιηθούν και άλλα Pin προκειμένου να τσεκάρουμε μέσω του προγράμματος αν η πλακέτα λειτουργεί κανονικά, αν έχει συνδεθεί στο

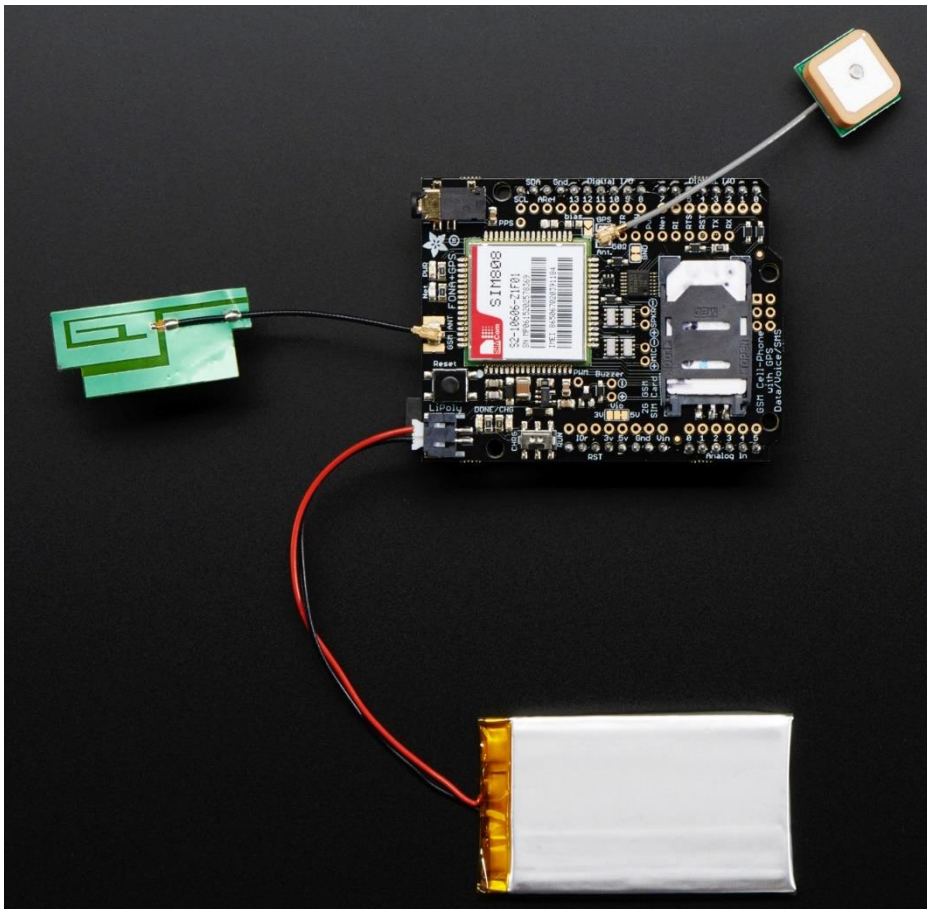
δίκτυο και αν πραγματοποιούνται οι ανάλογες εντολές. Γι' αυτές τις λειτουργίες θα χρησιμοποιήσουμε τα pins RI,RST,RTS,NET,PWR και Key όπως αυτά περιγράφηκαν παραπάνω.



Εικόνα 42: Arduino Uno

Εικόνα 43: Arduino Mega

Για να λειτουργήσει η πλακέτα με όλες τις λειτουργίες που περιγράψαμε θα πρέπει να συνδέσουμε στην πλακέτα του Fona την κεραία του GPS με τον αντάπτορά της, την κεραία GSM, την μπαταρία και τέλος την 2G Sim κάρτα.

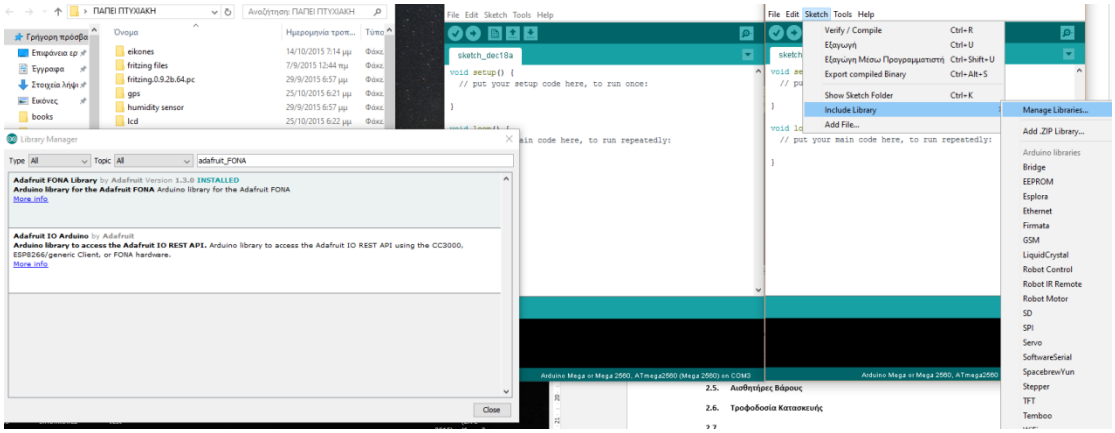


Εικόνα 44: Σύνδεση Fona με Arduino και επιπλέον εξαρτήματα

2.6.5. Ανάλυση Προγράμματος

Για να χρησιμοποιήσουμε το Fona 808 θα πρέπει αρχικά να κατεβάσουμε και να εγκαταστήσουμε τη βιβλιοθήκη Adafruit_FONA, στη συνέχεια να την εγκαταστήσουμε και να επανακινήσουμε το περιβάλλον ανάπτυξης IDE με όμοια διαδικασία όπως έχει περιγραφεί και στις προηγούμενες βιβλιοθήκες.

Άλλος τρόπος εισαγωγής βιβλιοθήκης πέρα από αυτό που περιγράφηκε είναι από το Menu εργαλεία Sketch, επιλέγουμε Include Library ,έπειτα Manage Libraries… και στο πλαίσιο αναζήτησης εισάγουμε το όνομα της βιβλιοθήκης που επιθυμούμε να βρούμε και κάνουμε αναζήτηση. Μ’ αυτό τον τρόπο μπορούμε να δούμε και τις εκδόσεις της βιβλιοθήκης την οποία εισάγουμε και την περιγραφή του περιεχομένου και της προέλευσής της. Τέλος επιλέγουμε το install, κλείνουμε το παράθυρο διαλόγου και ανακινούμε το IDE για να φορτώσει τη νέα βιβλιοθήκη. Η διαδικασία φαίνεται στην εικόνα που ακολουθεί.



Εικόνα 45: Εισαγωγή Βιβλιοθήκης για το Fona808

Στο παρακάτω πρόγραμμα “FONATest” της Adafruit παρατίθενται οι βασικές εντολές μέσω των οποίων μπορούμε να ελέγξουμε την πλακέτα του Fona808. Συγκεκριμένα θα δούμε πώς θα δίνουμε εντολές για να ελέγξουμε τα εξής:

1. Πώς να διαβάσουμε την ADC τάση
2. Πώς να δούμε τη στάθμη της μπαταρίας που έχουμε προσαρτήσει στη πλακέτα του Fona 808 και σε mAh και επί τοις εκατό
3. Να διαβάσουμε τον αριθμό CCID της κάρτας SIM που έχουμε εισάγει
4. Να ξεκλειδώνουμε την κάρτα SIM
5. Να δούμε σε τι κατάσταση βρίσκεται το δίκτυο. (εκτός σύνδεσης, έτοιμο για αποστολή είτε σε διαδικασία αποστολής)
6. Μπορούμε να πραγματοποιήσουμε κλήση από την πλακέτα προς κάποιον αριθμό που θα του εισάγουμε
7. Να κάνουμε απόρριψη εισερχόμενης κλήσης
8. Να απαντήσουμε σε εισερχόμενη κλήση
9. Να δούμε το πλήθος των SMS που έχουμε λάβει
10. Να διαβάσουμε SMS που έχουμε λάβει
11. Να διαβάσουμε όλα τα SMS είτε να διαγράψουμε κάποιο από αυτά
12. Να ενεργοποιήσουμε είτε να απενεργοποιήσουμε το GPRS
13. Να ενεργοποιήσουμε είτε να απενεργοποιήσουμε το GPS
14. Να ενεργοποιήσουμε τον συγχρονισμό της ώρας με αυτό του δικτύου στο οποίο είμαστε, είτε να πάρουμε την ώρα από το GPS
15. Να διαβάσουμε δεδομένα
16. Και τέλος να αποστείλουμε δεδομένα

Φυσικά οι δυνατότητες της πλακέτας μας δεν περιορίζονται εδώ.

Πρόγραμμα :

Το πρόγραμμα δίδεται ολόκληρο στα παραρτήματα της πτυχιακής. Σ' αυτή την ενότητα το πρόγραμμα θα παρουσιαστεί περιοδικά με ενσωματωμένα σχόλια και ταυτόχρονα εικόνες από την εκτέλεσή του και τα αποτελέσματα τα οποία λαμβάνουμε σε κάθε περίπτωση.

// εισάγουμε την βιβλιοθήκη του

```
#include "Adafruit_FONA.h"
```

//Καθορίζουμε τα pins για τη σειριακή εκπομπή - λήψη και reset

```
#define FONA_RX 2
#define FONA_TX 3
#define FONA_RST 4

char replybuffer[255];
```

/*Εισάγουμε τη βιβλιοθήκη softserial γιατί δε θα χρησιμοποιήσουμε τις hardware σειριακές θύρες του Arduino που είναι οι 0 και 1 αλλά τις 2 και 2 */

```
#include <SoftwareSerial.h>
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);
SoftwareSerial *fonaSerial = &fonaSS;
Adafruit_FONA fona = Adafruit_FONA(FONA_RST);
uint8_t readline(char *buff, uint8_t maxbuff, uint16_t timeout = 0);
uint8_t type;
```

//Αρχή της Void setup

```
void setup() {
  while (!Serial);
```

/* Ο ρυθμός που ορίζουμε είναι 9600bps για να είναι συμβατός και το ίδιο εύρος με τους άλλους αισθητήρες που έχουμε χρησιμοποιήσει. Εξ ορισμού η ταχύτητα ήταν ορισμένη στα 115200bpd*/

```
Serial.begin(9600);
  Serial.println(F("FONA basic test"));
  Serial.println(F("Initializing....(May take 3 seconds)"));
```

/*το Arduino επικοινωνεί σειριακά με το Fona με ρυθμό 4800bps*/

```
fonaSerial->begin(4800);
```

//Αν δεν έχουμε επιτυχή επικοινωνία τότε λαμβάνουμε σχετικό μήνυμα.

```
if (! fona.begin(*fonaSerial)) {
    Serial.println(F("Couldn't find FONA"));
    while (1);
  }
```

//Αλλιώς παίρνουμε μήνυμα σύνδεσης

```
type = fona.type();
  Serial.println(F("FONA is OK"));
  Serial.print(F("Found "));
```

Από την εκτέλεση του παραπάνω κώδικα εφόσον έχουμε συνδέσει την πλακέτα μας, έχουμε τροφοδοτήσει με εξωτερική πηγή την πλακέτα του Fona και έχουμε ανοίξει τη σειριακή οθόνη παίρνουμε το εξής μήνυμα:

```
FONA basic test
Initializing....(May take 3 seconds)
      ---> AT
      <--- AT
      ---> AT
      <--- AT
      ---> AT
      <--- AT
      ---> ATE0
      <--- ATE0
      ---> ATE0
      <--- OK
      ---> AT+CVHU=0
      <--- OK
      ---> ATI
      <--- SIM808 R14.18

OK
FONA is OK
```

/*Διαβάζουμε και εμφανίζουμε τον τύπο της πλακέτας που έχουμε συνδέσει.Στην περίπτωσή μας Fona 808(v2)*/

```
switch (type) {
    case FONA800L:
      Serial.println(F("FONA 800L")); break;
    case FONA800H:
      Serial.println(F("FONA 800H")); break;
    case FONA808_V1:
      Serial.println(F("FONA 808 (v1)")); break;
    case FONA808_V2:
      Serial.println(F("FONA 808 (v2)")); break;
    case FONA3G_A:
      Serial.println(F("FONA 3G (American)")); break;
    case FONA3G_E:
      Serial.println(F("FONA 3G (European)")); break;
    default:
      Serial.println(F("???")); break;
  }
```

//Τυπώνει το IMEI της συσκευής

```
char imei[15] = {0}; // MUST use a 16 character buffer for IMEI!
  uint8_t imeiLen = fona.getIMEI(imei);
  if (imeiLen > 0) {
    Serial.print("Module IMEI: "); Serial.println(imei);
  }
  printMenu();
}
```

//Τέλος της void setup()

Από τον παραπάνω κώδικα παίρνουμε την έκδοση της πλακέτας που διαθέτουμε και τον αριθμό IMEI της συσκευής μας.

```
Found FONA 808 (v2)
      ---> AT+GSN
      <--- 865067020747054
Module IMEI: 865067020747054
```

/*Τυπώνουμε τα γράμματα τα οποία πρέπει να εισάγει ο χρήστης στο παράθυρο διαλόγου της σειριακής οθόνης και αναλόγως της επιλογής εκτελούμε διαφορετική εντολή όπως αυτές περιγράφονται.*/

```
void printMenu(void) {
```

/*Τις εντολές μπορούμε να τις ομαδοποιήσουμε σε κατηγορίες βάση της λειτουργίας τους: Phone, SMS, Time, GPRS, GPS και γενικές. */

```
Serial.println(F("-------------------------------------"));
  Serial.println(F("[?] Print this menu"));
  Serial.println(F("[a] read the ADC 2.8V max (FONA800 & 808)"));
  Serial.println(F("[b] read the Battery V and % charged"));
  Serial.println(F("[C] read the SIM CCID"));
  Serial.println(F("[U] Unlock SIM with PIN code"));
  Serial.println(F("[i] read RSSI"));
  Serial.println(F("[n] get Network status"));
  Serial.println(F("[v] set audio Volume"));
  Serial.println(F("[V] get Volume"));
  Serial.println(F("[H] set Headphone audio (FONA800 & 808)"));
  Serial.println(F("[e] set External audio (FONA800 & 808)"));
  Serial.println(F("[T] play audio Tone"));
  Serial.println(F("[P] PWM/Buzzer out (FONA800 & 808)"));
  Serial.println(F("[f] tune FM radio (FONA800)"));
  Serial.println(F("[F] turn off FM (FONA800)"));
  Serial.println(F("[m] set FM volume (FONA800)"));
  Serial.println(F("[M] get FM volume (FONA800)"));
  Serial.println(F("[q] get FM station signal level (FONA800)"));
  // Phone
  Serial.println(F("[c] make phone Call"));
  Serial.println(F("[A] get call status"));
  Serial.println(F("[h] Hang up phone"));
  Serial.println(F("[p] Pick up phone"));
  // SMS
  Serial.println(F("[N] Number of SMSs"));
  Serial.println(F("[r] Read SMS #"));
  Serial.println(F("[R] Read All SMS"));
  Serial.println(F("[d] Delete SMS #"));
  Serial.println(F("[s] Send SMS"));
  Serial.println(F("[u] Send USSD"));
  // Time
  Serial.println(F("[y] Enable network time sync (FONA 800 & 808)"));
  Serial.println(F("[Y] Enable NTP time sync (GPRS FONA 800 &
808)"));
  Serial.println(F("[t] Get network time"));
  // GPRS
  Serial.println(F("[G] Enable GPRS"));
  Serial.println(F("[g] Disable GPRS"));
  Serial.println(F("[l] Query GSMLOC (GPRS)"));
  Serial.println(F("[w] Read webpage (GPRS)"));
  Serial.println(F("[W] Post to website (GPRS)"));
```

//Αρχή του κυρίως προγράμματος void loop()

```
void loop() {
```

//Το πρόγραμμα περιμένει να εισαχθεί μια επιλογή από τον χρήστη

```
Serial.print(F("FONA> "));
  while (! Serial.available() ) {
    if (fona.available()) {
      Serial.write(fona.read());
    }
  }

  char command = Serial.read();
  Serial.println(command);
```

//Αναλόγως της επιλογής εκτελείται διαφορετικό case

```
switch (command) {
    case '?': {
        printMenu();
        break;
      }
//Διαβάζουμε την τάση του ADC

case 'a': {
        uint16_t adc;
        if (! fona.getADCVoltage(&adc)) {
          Serial.println(F("Failed to read ADC"));
        } else {
          Serial.print(F("ADC = ")); Serial.print(adc);
Serial.println(F(" mV"));
        }
        break;
      }
```

//Διαβάζουμε την τάση της μπαταρίας και επίσης παίρνουμε ποσοστό της φόρτισής της

```
case 'b': {
        uint16_t vbat;
        if (! fona.getBattVoltage(&vbat)) {
          Serial.println(F("Failed to read Batt"));
        } else {
          Serial.print(F("VBat = ")); Serial.print(vbat);
Serial.println(F(" mV"));
        }
```

Πληκτρολογώντας το «b» στη γραμμή εντολών της σειριακής οθόνης παίρνουμε το εξής μήνυμα:

```
b
      ---> AT+CBC
      <--- +CBC: 0,69,3957
VBat = 3957 mV
      ---> AT+CBC
      <--- +CBC: 0,69,3957
```

```
VPct = 69%
```

Ταυτόχρονα με το σύμβολο το οποίο εισάγουμε στην οθόνη μας βλέπουμε και την εκτύπωση των AT commands, την απάντηση που παίρνουμε και μετά την εκτύπωση του αποτελέσματος. Στη συγκεκριμένη περίπτωση η τάση της μπαταρίας μας είναι VBat = 3957 mV και ποσοστό VPct = 69%.

```
        if (! fona.getBattPercent(&vbat)) {
          Serial.println(F("Failed to read Batt"));
        } else {
          Serial.print(F("VPct = ")); Serial.print(vbat);
Serial.println(F("%"));
        }

        break;
      }
```

//Ξεκλειδώνουμε την κάρτα με κωδικό PIN

Απαραίτητο για να συνδεθούμε στο δίκτυο είναι να έχουμε ξεκλειδώσει την κάρτα SIM. Αυτό μπορούμε να το κάνουμε είτε από το κινητό μας είτε πατώντας αρχικά το «U» και έπειτα εισάγοντας τον τετραψήφιο της κάρτας μας. Πριν εισάγουμε όμως τον κωδικό ας δούμε την κατάσταση στην οποία βρίσκεται αυτή τη στιγμή η σύνδεσή μας ώστε στη συνέχεια να διαπιστώσουμε τη διαφορά αφού ξεκλειδώσουμε την SIM.

Εισάγουμε το «n» και παίρνουμε τα εξής:

```
FONA> n
      ---> AT+CREG?
      <--- +CREG: 0,0
Network status 0: Not registered
```

Έπειτα πληκτρολογούμε το «U» και τον κωδικό και ενεργοποιούμε τη σύνδεση.

```
FONA> U
Enter 4-digit PIN
XXXX
Unlocking SIM card:     ---> AT+CPIN=XXXX
      <--- OK
OK!
FONA>
+CPIN: READY
Call Ready
SMS Ready
```

Αν εκτελέσουμε πάλι τον έλεγχο της κατάστασης του δικτύου παίρνουμε τα εξής:

```
FONA> n
      ---> AT+CREG?
      <--- +CREG: 0,1
```

Network status 1: Registered (home)

```
case 'U': {
        char PIN[5];
        flushSerial();
        Serial.println(F("Enter 4-digit PIN"));
        readline(PIN, 3);
        Serial.println(PIN);
        Serial.print(F("Unlocking SIM card: "));
        if (! fona.unlockSIM(PIN)) {
```

```
          Serial.println(F("Failed"));
        } else {
          Serial.println(F("OK!"));
        }
        break;
      }
```

//Διαβάζει το RSSI (=Received signal strength indication)

```
case 'i': {
        uint8_t n = fona.getRSSI();
        int8_t r;

        Serial.print(F("RSSI = ")); Serial.print(n); Serial.print(":
");
        if (n == 0) r = -115;
        if (n == 1) r = -111;
        if (n == 31) r = -52;
        if ((n >= 2) && (n <= 30)) {
          r = map(n, 2, 30, -110, -54);
        }
        Serial.print(r); Serial.println(F(" dBm"));

        break;
      }
```

Από την εκτέλεση του παραπάνω κώδικα πληκτρολογώντας το «i» παίρνουμε τα εξής:

```
i
      ---> AT+CSQ
      <--- +CSQ: 9,0
RSSI = 9: -96 dBm
```

//Διαβάζει την κατάσταση που βρίσκεται το δίκτυο

```
   case 'n': {
        uint8_t n = fona.getNetworkStatus();
        Serial.print(F("Network status "));
        Serial.print(n);
        Serial.print(F(": "));
        if (n == 0) Serial.println(F("Not registered"));
        if (n == 1) Serial.println(F("Registered (home)"));
        if (n == 2) Serial.println(F("Not registered (searching)"));
        if (n == 3) Serial.println(F("Denied"));
        if (n == 4) Serial.println(F("Unknown"));
        if (n == 5) Serial.println(F("Registered roaming"));
        break;
      }
```

//Για τον ήχο

```
    case 'v': {
        // set volume
        flushSerial();
        if ( (type == FONA3G_A) || (type == FONA3G_E) ) {
          Serial.print(F("Set Vol [0-8] "));
        } else {
          Serial.print(F("Set Vol % [0-100] "));
        }
        uint8_t vol = readnumber();
```

```
          Serial.println();
          if (! fona.setVolume(vol)) {
            Serial.println(F("Failed"));
          } else {
            Serial.println(F("OK!"));
          }
          break;
        }
case 'V': {
          uint8_t v = fona.getVolume();
          Serial.print(v);
          if ( (type == FONA3G_A) || (type == FONA3G_E) ) {
            Serial.println(" / 8");
          } else {
            Serial.println("%");
          }
          break;
        }
case 'H': {
         if (! fona.setAudio(FONA_HEADSETAUDIO)) {
            Serial.println(F("Failed"));
          } else {
            Serial.println(F("OK!"));
          }
          fona.setMicVolume(FONA_HEADSETAUDIO, 15);
          break;
        }
case 'e': {
          if (! fona.setAudio(FONA_EXTAUDIO)) {
            Serial.println(F("Failed"));
          } else {
            Serial.println(F("OK!"));
          }

          fona.setMicVolume(FONA_EXTAUDIO, 10);
          break;
        }
case 'T': {
          flushSerial();
          Serial.print(F("Play tone #"));
          uint8_t kittone = readnumber();
          Serial.println();
          // play for 1 second (1000 ms)
          if (! fona.playToolkitTone(kittone, 1000)) {
            Serial.println(F("Failed"));
          } else {
            Serial.println(F("OK!"));
          }
          break;
        }
//Για το PWM Pulse Width Modulator
case 'P': {
          // PWM Buzzer output @ 2KHz max
          flushSerial();
          Serial.print(F("PWM Freq, 0 = Off, (1-2000): "));
          uint16_t freq = readnumber();
          Serial.println();
          if (! fona.setPWM(freq)) {
            Serial.println(F("Failed"));
```

```
        } else {
          Serial.println(F("OK!"));
        }
        break;
      }
```

//Για το τηλέφωνο

// κλήση ενός τηλεφώνου

```
case 'c': {
        // κλήση ενός τηλεφώνου
        char number[30];
        flushSerial();
        Serial.print(F("Call #"));
        readline(number, 30);
        Serial.println();
        Serial.print(F("Calling ")); Serial.println(number);
        if (!fona.callPhone(number)) {
          Serial.println(F("Failed"));
        } else {
          Serial.println(F("Sent!"));
        }

        break;
      }
```

//Παίρνει την κατάσταση στην οποία βρίσκεται η κλήση

```
case 'A': {
        int8_t callstat = fona.getCallStatus();
        switch (callstat) {
          case 0: Serial.println(F("Ready")); break;
          case 1: Serial.println(F("Could not get status")); break;
          case 3: Serial.println(F("Ringing (incoming)")); break;
          case 4: Serial.println(F("Ringing/in progress
(outgoing)")); break;
          default: Serial.println(F("Unknown")); break;
        }
        break;
      }
```

//κλείσιμο του τηλεφώνου

```
case 'h': {
        if (! fona.hangUp()) {
          Serial.println(F("Failed"));
        } else {
          Serial.println(F("OK!"));
        }
        break;
       }
```

//Απάντηση της κλήσης

```
case 'p': {

        if (! fona.pickUp()) {
          Serial.println(F("Failed"));
        } else {
          Serial.println(F("OK!"));
        }
        break;
      }
```

/*** SMS ***/

//Διαβάζει τον αριθμό των εισερχόμενων που υπάρχουν στην κάρτα

```
    case 'N': {

        int8_t smsnum = fona.getNumSMS();
        if (smsnum < 0) {
          Serial.println(F("Could not read # SMS"));
        } else {
          Serial.print(smsnum);
          Serial.println(F(" SMS's on SIM card!"));
        }
        break;
      }
```

Από την εκτέλεση του παραπάνω κώδικα παίρνουμε τα εξής πληκτρολογώντας το «N»:

```
FONA> N
      ---> AT+CMGF=1
      <--- OK
      ---> AT+CPMS?
      <--- +CPMS: "SM_P",34,40,"SM_P",34,40,"SM_P",34,40
34 SMS's on SIM card!
```

//Διαβάζει ένα SMS

```
case 'r': {
        flushSerial();
        Serial.print(F("Read #"));
        uint8_t smsn = readnumber();
        Serial.print(F("\n\rReading SMS #")); Serial.println(smsn);
```

//Εμφανίζει τον αποστολέα του SMS – τον αριθμό του τηλεφώνου

```
        if (! fona.getSMSSender(smsn, replybuffer, 250)) {
          Serial.println("Failed!");
          break;
        }
Serial.print(F("FROM: ")); Serial.println(replybuffer);


        uint16_t smslen;
        if (! fona.readSMS(smsn, replybuffer, 250, &smslen)) {
          Serial.println("Failed!");
          break;
        }
        Serial.print(F("***** SMS #")); Serial.print(smsn);
   Serial.print(" ("); Serial.print(smslen);     Serial.println(F(")
bytes *****"));
        Serial.println(replybuffer);
        Serial.println(F("*****"));

        break;
      }
```

//Διαβάζει όλα τα SMS

```
case 'R': {
```

```
        int8_t smsnum = fona.getNumSMS();
        uint16_t smslen;
        int8_t smsn;

        if ( (type == FONA3G_A) || (type == FONA3G_E) ) {
          smsn = 0;
          smsnum--;
        } else {
          smsn = 1;
        }

        for ( ; smsn <= smsnum; smsn++) {
          Serial.print(F("\n\rReading SMS #")); Serial.println(smsn);
          if (!fona.readSMS(smsn, replybuffer, 250, &smslen)) {  //
pass in buffer and max len!
            Serial.println(F("Failed!"));
            break;
          }
          if (smslen == 0) {
            Serial.println(F("[empty slot]"));
            smsnum++;
            continue;
          }

          Serial.print(F("***** SMS #")); Serial.print(smsn);
          Serial.print(" ("); Serial.print(smslen);
Serial.println(F(") bytes *****"));
          Serial.println(replybuffer);
          Serial.println(F("*****"));
        }
        break;
      }
```

//Διαγράφει ένα SMS

```
case 'd': {

        flushSerial();
        Serial.print(F("Delete #"));
        uint8_t smsn = readnumber();

Serial.print(F("\n\rDeleting SMS #")); Serial.println(smsn);
        if (fona.deleteSMS(smsn)) {
          Serial.println(F("OK!"));
        } else {
          Serial.println(F("Couldn't delete"));
        }
        break;
      }
```

//Στέλνει SMS

```
case 's': {

        char sendto[21], message[141];
        flushSerial();
        Serial.print(F("Send to #"));
        readline(sendto, 20);
        Serial.println(sendto);
```

```
        Serial.print(F("Type out one-line message (140 char): "));
        readline(message, 140);
        Serial.println(message);
        if (!fona.sendSMS(sendto, message)) {
          Serial.println(F("Failed"));
        } else {
          Serial.println(F("Sent!"));
        }

        break;
      }
```

Για παράδειγμα εκτελώντας τον παραπάνω κώδικα στέλνουμε ένα μήνυμα όπως φαίνεται παρακάτω:

```
FONA> s
Send to #698350XXXX
Type out one-line message (140 char): eimai to arduino mazi me to
fona kai sou stelnw ena minima
      ---> AT+CMGF=1
      <--- OK
      ---> AT+CMGS="698XXXXXXX"
      <--- >
> eimai to arduino mazi me to fona kai sou stelnw ena minima
^Z
Sent!
```

//Στέλνει USSD (=Unstructured Supplementary Service Data)

```
case 'u': {
      char message[141];
      flushSerial();
      Serial.print(F("Type out one-line message (140 char): "));
      readline(message, 140);
      Serial.println(message);

      uint16_t ussdlen;
      if (!fona.sendUSSD(message, replybuffer, 250, &ussdlen)) {
        Serial.println(F("Failed"));
      } else {
        Serial.println(F("Sent!"));
        Serial.print(F("***** USSD Reply"));
Serial.print(" ("); Serial.print(ussdlen); Serial.println(F(") bytes
*****"));
        Serial.println(replybuffer);
        Serial.println(F("*****"));
      }
    }
```

//Για το χρόνο

//Ενεργοποιεί το συγχρονισμό του Fona με τον χρόνο του Δικτύου

```
case 'y': {
        if (!fona.enableNetworkTimeSync(true))
          Serial.println(F("Failed to enable"));
        break;
      }
```

//Ενεργοποιεί τον συχρονισμό NTP (=Network Time Protocol)

```
    case 'Y': {
        if (!fona.enableNTPTimeSync(true, F("pool.ntp.org")))
          Serial.println(F("Failed to enable"));
        break;
      }
//Διαβάζει τον χρόνο
    case 't': {
        char buffer[23];
        fona.getTime(buffer, 23);
        Serial.print(F("Time = ")); Serial.println(buffer);
        break;
      }
```

//Για το GPS

```
    case 'o': {
//Απενεργοποιεί το GPS
        if (!fona.enableGPS(false))
          Serial.println(F("Failed to turn off"));
        break;
      }
    case 'O': {
//Ενεργοποιεί το GPS
        if (!fona.enableGPS(true))
          Serial.println(F("Failed to turn on"));
        break;
      }
    case 'x': {
        int8_t stat;
```

//Ελέγχει τη σύνδεση του GPS

```
        stat = fona.GPSstatus();
        if (stat < 0)
          Serial.println(F("Failed to query"));
        if (stat == 0) Serial.println(F("GPS off"));
        if (stat == 1) Serial.println(F("No fix"));
        if (stat == 2) Serial.println(F("2D fix"));
        if (stat == 3) Serial.println(F("3D fix"));
        break;
      }
    case 'L': {
```

//Λαμβάνει πληροφορίες του GPS longitude, latitude, altitude κλπ

```
        char gpsdata[120];
        fona.getGPS(0, gpsdata, 120);
        if (type == FONA808_V1)
          Serial.println(F("Reply in format:
mode,longitude,latitude,altitude,utctime(yyyymmddHHMMSS),ttff,satelli
tes,speed,course"));
        else
          Serial.println(F("Reply in format:
mode,fixstatus,utctime(yyyymmddHHMMSS),latitude,longitude,altitude,sp
eed,course,fixmode,reserved1,HDOP,PDOP,VDOP,reserved2,view_satellites
,used_satellites,reserved3,C/N0max,HPA,VPA"));
        Serial.println(gpsdata);

        break;
      }
case 'E': {
        flushSerial();
        if (type == FONA808_V1) {
```

```
          Serial.print(F("GPS NMEA output sentences (0 = off, 34 =
RMC+GGA, 255 = all)"));
        } else {
          Serial.print(F("On (1) or Off (0)? "));
        }
        uint8_t nmeaout = readnumber();
        fona.enableGPSNMEA(nmeaout);
        break;
      }
//Για το GPRS
```

Πίνακας 1: Στοιχεία Εισαγωγής Παρόχων

	COSMOTE	VODAFONE	WIND	Q
GPRS APN	GPRS: mms internet	GPRS σύνδεση: internet Σύνδεση: internet.vodafone.gr Καρτοκινητό: webkarta.vodafone.gr	gnet.b-online.gr ή gint.b-online.gr	Q Internet
Username	user	Null	Null	myq
Password	pass	Null	Null	Null
MCC	202	202	202	Null
MNC	01	05	10	Null
APN	Default	Default	Default	Null
DNS	195.167.065.194	213.249.17.10, 213.249.17.11	212.152.79.19, 212.152.79.20	Null

```
    case 'g': {
// Κλείνει το GPRS
        if (!fona.enableGPRS(false))
          Serial.println(F("Failed to turn off"));
        break;
      }
    case 'G': {
// Ανοίγει το GPRS
        if (!fona.enableGPRS(true))
          Serial.println(F("Failed to turn on"));
        break;
      }
```

Για την εκτέλεση του συγκεκριμένου τμήματος κώδικα είναι απαραίτητη η εισαγωγή της συνάρτησης με τα στοιχεία του παρόχου όπως αυτά φαίνονται στον παραπάνω πίνακα:

```
fona.setGPRSNetworkSettings(F("internet"), F(""), F(""));
```

Από την εκτέλεση του παραπάνω κώδικα παίρνουμε τα εξής:

```
FONA> G
      ---> AT+CIPSHUT
```

```
<--- SHUT OK
---> AT+CGATT=1
<--- OK
---> AT+SAPBR=3,1,"CONTYPE","GPRS"
<--- OK
---> AT+SAPBR=3,1,"APN","internet"
<--- OK
---> AT+SAPBR=3,1,"USER",""
<--- OK
---> AT+SAPBR=3,1,"PWD",""
<--- OK
---> AT+SAPBR=1,1
<--- OK
```

Και αντίστοιχα για να απενεργοποιήσουμε το GPRS πληκτρολογούμε το “g” και εκτελούνται οι ακόλουθες εντολές:

```
FONA> g
      ---> AT+CIPSHUT
      <--- SHUT OK
      ---> AT+SAPBR=0,1
      <--- OK
      ---> AT+CGATT=0
      <--- OK
```

```
case 'l': {
   // ελέγχει το GSMLOC (ενεργοποιημένο το GPRS)
        uint16_t returncode;

        if (!fona.getGSMLoc(&returncode, replybuffer, 250))
          Serial.println(F("Failed!"));
        if (returncode == 0) {
          Serial.println(replybuffer);
        } else {
            Serial.print(F("Fail code #"));
            Serial.println(returncode);
        }

        break;
      }
```

Εφόσον έχουμε ενεργοποιήσει το GPRS (“G”) εκτελώντας την παραπάνω εντολή “l” παίρνουμε την τοποθεσία, τον χρόνο και την ώρα. Συγκεκριμένα παίρνουμε τα εξής δεδομένα με σειρά εμφάνισης: <locationcode>,<longitude>,<latitude>,<date>,<time>.

```
FONA> l
      ---> AT+CIPGSMLOC=1,1
      <--- +CIPGSMLOC: 0,23.668779,37.998703,2016/01/05,20:22:42
23.668779,37.998703,2016/01/05,20:22:42
```

```
case 'w': {
        // Γράφει σε ένα URL
```

```
        uint16_t statuscode;
        int16_t length;
        char url[80];

        flushSerial();
        Serial.println(F("NOTE: in beta! Use small webpages to
read!"));
        Serial.println(F("URL to read (e.g.
www.adafruit.com/testwifi/index.html):"));
        Serial.print(F("http://")); readline(url, 79);
        Serial.println(url);

        Serial.println(F("****"));
        if (!fona.HTTP_GET_start(url, &statuscode, (uint16_t
*)&length)) {
          Serial.println("Failed!");
          break;
        }
        while (length > 0) {
          while (fona.available()) {
            char c = fona.read();


#if defined(__AVR_ATmega328P__) || defined(__AVR_ATmega168__)
              loop_until_bit_is_set(UCSR0A, UDRE0);           UDR0
              = c;
#else
            Serial.write(c);
#endif
            length--;
            if (! length) break;
          }
        }
        Serial.println(F("\n****"));
        fona.HTTP_GET_end();
        break;
      }
case 'W': {
   //Ποστάρει δεδομένα σε ένα Site
        uint16_t statuscode;
        int16_t length;
        char url[80];
        char data[80];

        flushSerial();
        Serial.println(F("NOTE: in beta! Use simple websites to
post!"));
        Serial.println(F("URL to post (e.g. httpbin.org/post):"));
        Serial.print(F("http://")); readline(url, 79);
        Serial.println(url);
        Serial.println(F("Data to post (e.g. \"foo\" or
\"{\"simple\":\"json\"}\"):"));
        readline(data, 79);
        Serial.println(data);

        Serial.println(F("****"));
        if (!fona.HTTP_POST_start(url, F("text/plain"), (uint8_t *)
data, strlen(data), &statuscode, (uint16_t *)&length)) {
          Serial.println("Failed!");
          break;
```

```
        }
        while (length > 0) {
          while (fona.available()) {
            char c = fona.read();

#if defined(__AVR_ATmega328P__) || defined(__AVR_ATmega168__)
            loop_until_bit_is_set(UCSR0A, UDRE0);            UDR0 =
c;
#else
            Serial.write(c);
#endif

            length--;
            if (! length) break;
          }
        }
        Serial.println(F("\n****"));
        fona.HTTP_POST_end();
        break;
      }
```

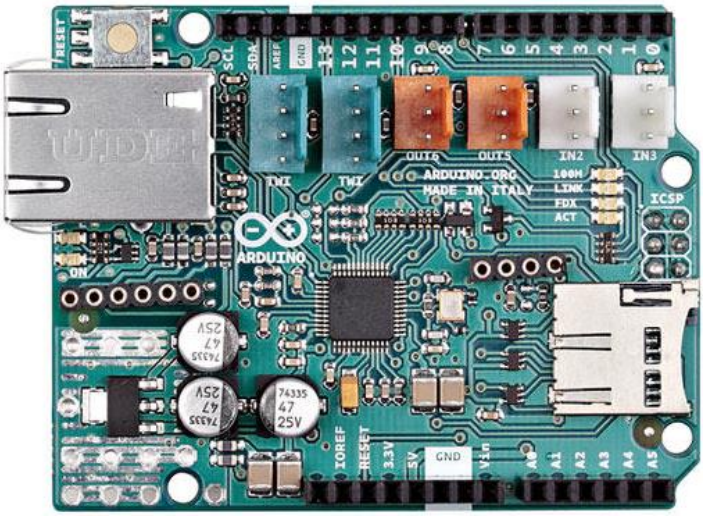
2.7. Ethernet Shield

Από τα πιο σημαντικά Shield της εφαρμογής μας είναι το Ethernet, μέσω του οποίου η κατασκευή μας επικοινωνεί με τον έξω κόσμο. Έχοντας σύνδεση στο internet μπορούμε να στέλνουμε και να λαμβάνουμε δεδομένα από οπουδήποτε στον κόσμο. Τα δεδομένα τα οποία συλλέγονται από τους αισθητήρες αποθηκεύονται σε μεταβλητές και στη συνέχεια αποστέλλονται στον Server μας για να χρησιμοποιηθούν από την εφαρμογή μας.

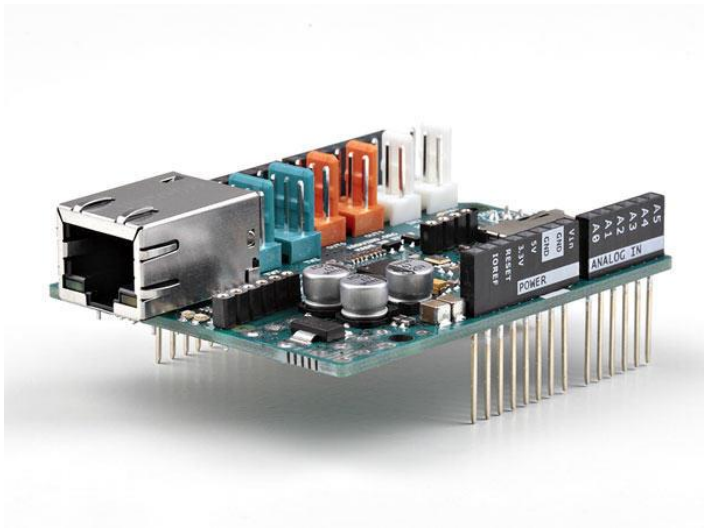
2.7.1. Τεχνικά Χαρακτηριστικά

Η έκδοση Ethernet Shield που χρησιμοποιήθηκε είναι η Rev 2 όπως φαίνεται και στις εικόνες που ακολουθούν. Το μόνο που απαιτείται επιπλέον είναι είτε ένα καλώδιο τύπου RJ-45 κατηγορίας CAT5 είτε CAT6 για σύνδεση με το router, ενώ για σύνδεση με υπολογιστή χρησιμοποιούμε καλώδιο τύπου cross-over .Η τάση λειτουργίας της πλακέτας είναι 5 Volt και τροφοδοτείται άμεσα από την πλακέτα μας. Το shield είναι βασισμένο στον Ethernet Controller W5500, ο οποίος διαθέτει εσωτερικό buffer 32K. Η ταχύτητα σύνδεσης είναι 10/100 Mbs και για τη σύνδεση με το Arduino χρησιμοποιείται η πόρτα SPI. Στην πλακέτα στο κάτω αριστερά τμήμα υπάρχει υποδοχή για micro SD Card, στην οποία έχουμε τη δυνατότητα να αποθηκεύουμε δεδομένα.

Στο συγκεκριμένο Shield δεν έχει τοποθετηθεί PoE (=Power Over Ethernet) Module. Φυσικά, αν η εφαρμογή μας το απαιτούσε θα μπορούσε να τοποθετηθεί στα τέσσερα κενά Pins που υπάρχουν ακριβώς πάνω από την SD Card. Το pin out της πλακέτας είναι όμοιο με αυτό του Arduino.



Εικόνα 46: Ethernet Shield Εμπρός Όψη



Εικόνα 47: Ethernet Shield

Πάνω στην πλακέτα υπάρχουν ενδεικτικά led κατάστασης μέσω των οποίων μπορούμε να δούμε αν το shield λειτουργεί, αν έχει συνδεθεί κλπ.

Συγκεκριμένα το ON led ανάβει όταν το Shield λειτουργεί. Το Link led δείχνει την κατάσταση της σύνδεσης. Όταν το module εκπέμπει είτε λαμβάνει δεδομένα τότε αυτό αναβοσβήνει. Το «FDX» είναι αναμμένο όταν η σύνδεση είναι Full Duplex. Το «100M» είναι αναμμένο όταν η μετάδοση είναι 100Mbps ενώ όταν αυτό είναι σβηστό σημαίνει ότι η ταχύτητα είναι 10 Mbps. Τέλος το ACT Led ανάβει όταν υπάρχει εκπομπή είτε λήψη.

Οι διαστάσεις του PCB είναι 2.7 και 2.1 inches με το Ethernet connector να προεξέχει. Τέλος το Shield έρχεται με τοποθετημένο αυτοκόλλητο με την mac address του, η οποία φυσικά είναι μοναδική. Φυσικά, επόμενες ενέργειες που πρέπει να γίνουν είναι ο καθορισμός της IP που εξαρτάται από το εκάστοτε δίκτυο. Αν στο δίκτυο χρησιμοποιείται DHCP τότε η IP θα ληφθεί δυναμικά. Προαιρετικά μπορούμε να καθορίσουμε και τη subnet mask και το gateway.

2.7.2. Διάγραμμα Συνδεσμολογίας

Κάθε shield που έχει σχεδιαστεί για το Arduino, έχει δημιουργηθεί κατά τέτοιο τρόπο ώστε να τοποθετείται με ευκολία σ' αυτό χωρίς να απαιτούνται επιπλέον καλωδιώσεις. Οι θύρες στις οποίες προσαρμόζονται είναι όμοιες με αυτές του FONA 808.

Εικόνα 48: Pins που καταλαμβάνει το Ethernet Shield

Όπως είδαμε και στο Ethernet Shield αλλά και σ' αυτό του Fona 808 μπορεί να τοποθετηθεί στα Pin του Arduino Mega που είναι όμοια με αυτά του Uno. Η δυνατότητα όμως τοποθέτησης σε αυτά τα Pin μας περιορίζει στο να χρησιμοποιούμε ένα μόνο Shield τη φορά. Ο λόγος της χρήσης ενός καθορίζεται από το ότι και τα δύο χρησιμοποιούν τις παράλληλες θύρες επικοινωνίας για τη μεταξύ Arduino και Shield επικοινωνία. Για το λόγο αυτό θα δούμε παρακάτω στο συνολικό διάγραμμα πώς επιτυγχάνεται η σύνδεση όλων αξιοποιώντας και άλλες διαθέσιμες θύρες σειριακής επικοινωνίας που διαθέτει το Mega.

2.7.3. Ανάλυση Προγράμματος

Το shuield που θα χρησιμοποιήσουμε στην εφαρμογή είναι το Rev2 της Sparrkfun, συνεπώς η προγενέστερη βιβλιοθήκη Ethernet1.h δεν είναι συμβατή. Συνεπώς θα χρησιμοποιήσουμε την Ethernet2.h. Επίσης το Ethernet shield με τον αντίστοιχο κώδικα μπορεί να μετατρέψει την κατασκευή μας τόσο σε client όσο και σε Server. Για τους σκοπούς της κατασκευής μας η δημιουργία ενός Server δεν έχει κάποιο σκοπό αφού δε θέλουμε στο ίδιο το Arduino να φιλοξενούμε την ιστοσελίδα μας (και αν είναι στατική). Συνεπώς ο κώδικας ο οποίος θα αναπτυχθεί θα είναι για τη χρήση του Arduino Mega ως client ο οποίος «σερβίρει» - παρέχει – αποστέλλει δεδομένα στον Server μας.

Για τη χρήση του Ethernet Shield πρώτη κίνηση είναι ο ορισμός της mac ο οποίος αναγράφεται στο πίσω μέρος του. Έπειτα πρέπει να ορίσουμε την IP του client η οποία πρέπει να μην επαναλαμβάνεται φυσικά με άλλη IP στο δίκτυο που ανήκουμε. Αν κάτι τέτοιο συμβεί τόσο εμείς όσο ο άλλος υπολογιστής θα έχουμε πρόβλημα και θα παίρνουμε το μήνυμα duplicate IP. Έπειτα όπως είπαμε αφού η κατασκευή μας θα λειτουργεί ως client θα πρέπει να αποστέλλουμε τα δεδομένα σε κάποιον Server του οποίου θα πρέπει να ορίσουμε τη διεύθυνση. Τόσο για την IP του client όσο και του Server θα δημιουργήσουμε σύμφωνα με τον ορισμό της βιβλιοθήκης αντικείμενα τύπου IP.

```
#include <Ethernet2.h>
```

```
byte mac[] = { 0x90, 0xA2, 0xDA, 0x10, 0x8C, 0x94 };
IPAddress ip(192, 168, 1, 68);//για την ip του Arduino
IPAddress server(192, 168, 1, 2);
EthernetClient client;
int server_id = 1;
void setup() {
  Serial.begin(9600);
  Ethernet.begin(mac, ip);
}
Void loop(){
}
void serverinfookeanos() {
  if (client.connect(server, 80)) {
    Serial.println("-> Connected serverinfookeanos");
    //client.print(
"83.212.116.254/webserver/measurements/serverinfo.php?");
    client.print( "GET /webserver/measurements/serverinfo.php?");
    client.print("server_id=");
    client.print( server_id );
    client.print("&");
    client.print("arduino_sr=");
    client.print( arduino_sr );
    client.print("&");
    client.print("longtitude=");
    client.print( longitude , 8 );
    client.print("&");
    client.print("latitude=");
    client.print( latitude , 8);
    Serial.println("-> Connected");
    client.println( " HTTP/1.1");
    client.print( "Host: " );
    client.println(server);
    client.println( "Connection: close" );
    client.println();
    client.println();
    client.stop();
    delay(5000);
  }
  else {
    // you didn't get a connection to the server:
    Serial.println("--> connection failed/n");
  }
}
```

Στο πρόγραμμα που παρατέθηκε φαίνεται το Include την βιβλιοθήκης, ο καθορισμός της mac, των ip του Server και του client. Στη συνέχεια ο ορισμός του τρόπου λειτουργίας της κατασκευής, θέτουμε το id του server το οποίο θα το χρησιμοποιήσουμε στην εφαρμογή μας όπως θα δούμε παρακάτω. Στο void setup() ξεκινάμε τη σειριακή επικοινωνία και εκκινούμε τις λειτουργίες του shield. Σ' αυτό το σημείο του κώδικα δεν κρίνεται σκόπιμο να αναπτυχθεί όλο το πρόγραμμα αλλά ενδεικτικά φαίνεται μία συνάρτηση η serverinfookeanos() μέσω της οποίας αποστέλλουμε με τη μέθοδο Get στον Server μας τον κωδικό του Arduino, το γεωγραφικό μήκος και πλάτος όπως αυτό έχει υπολογιστεί από αντίστοιχη συνάρτηση που αναπτύξαμε αξιοποιώντας το Fona 808 shield και αυτά τα αποστέλλουμε. Στον κώδικα που φαίνεται τα δεδομένα αποστέλλονται σε τοπικό Server στην port 80.

Για την αποστολή δεδομένων στον Server συνολικά υλοποήθηκαν 5 συναρτήσεις οι οποίες είναι οι void entersuccessokeanos(String rfid), void serverinfookeanos(), void templogsokeanos() , void enterenterattemptsokeanos(String rfid), void serverstateokeanos(boolean alarm, boolean sensors) και φαίνονται στον κώδικα που περιλαμβάνεται στο αρχείο all.ino που συνοδεύει την πτυχιακή.

2.8. Αισθητήρας Αερίων

Άλλη αίσθηση που θέλουμε να προσδώσουμε στην κατασκευή μας, είναι η μέτρηση επικίνδυνων αερίων στον χώρο των Server. Αυτά μπορούν να παραχθούν από τις μπαταρίες, οι οποίες στην πλειοψηφία των περιπτώσεων είναι τοποθετημένες μέσα στον ίδιο χώρο. Η τοποθέτηση ενός αισθητήρα αερίων θα διασφαλίσει αφενός τη μη ύπαρξη επικίνδυνων αερίων στον χώρο που θα εισέλθει το προσωπικό και αφετέρου με τη χρήση του μπορούμε να αντιληφθούμε την ύπαρξη καπνού και κατ' επέκταση φωτιάς.

Στο εμπόριο συναντά κανείς μεγάλο αριθμό αισθητήρων αερίων οι οποίοι διαφέρουν στο είδος των αερίων που αυτοί ανιχνεύουν, στην ακρίβεια των μετρήσεων, στην ταχύτητα λήψης των μετρήσεων και λογικό επακόλουθο όλων αυτών είναι και η διαφοροποίηση της τιμής τους. Ο αισθητήρας που επιλέχθηκε στην κατασκευή μας είναι ο MQ-8 της Hanwei. Μέσω αυτού μπορούμε να λάβουμε μετρήσεις που καλύπτουν τη ζητούμενη ακρίβεια, σε σχετικά μικρό χρονικό διάστημα και με κόστος αγοράς 5 €.

Εικόνα 49: Αισθητήρας MQ-5

2.8.1. Τεχνικά Χαρακτηριστικά του Αισθητήρα Αερίων

Στην εικόνα που ακολουθεί βλέπουμε τα τεχνικά χαρακτηριστικά του αισθητήρα MQ-8, τις διαστάσεις και το διάγραμμα απόκρισης, όπως αυτά δίνονται από την κατασκευάστρια εταιρία.

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
P_L	Load resistance	10K Ω	
R_H	Heater resistance	31±5%	Room Tem
P_H	Heating consumption	less than800mW	

B. Environment condition

Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10℃-50℃	
Tas	Storage Tem	-20℃-70℃	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

C. Sensitivity characteristic

Symbol	Parameter name	Technical parameter	Ramark 2
Rs	Sensing Resistance	10K Ω - 60K Ω (1000ppm H_2)	Detecting concentration scope: 100-10000ppm Hydrogen (H_2)
α (1000ppm/ 500ppmH_2)	Concentration slope rate	≤0.6	
Standard detecting condition	Temp: 20℃±2℃ Humidity: 65%±5%	Vc:5V±0.1 Vh: 5V±0.1	
Preheat time	Over 24 hour		

D. Structure and configuration, basic measuring circuit

	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al_2O_3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

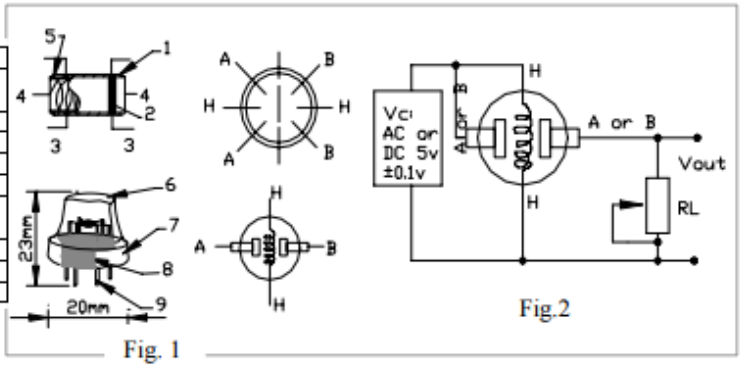
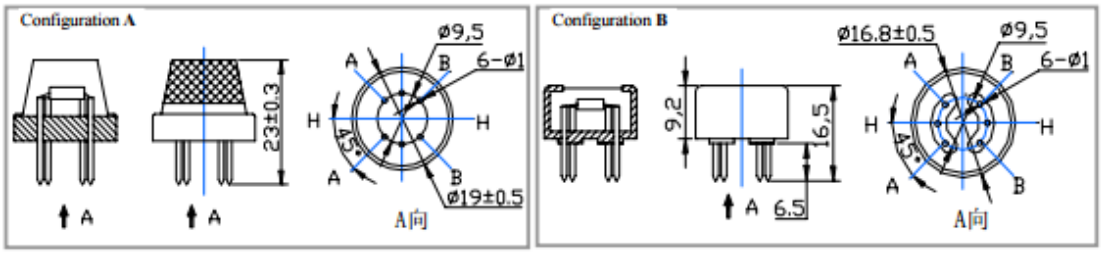
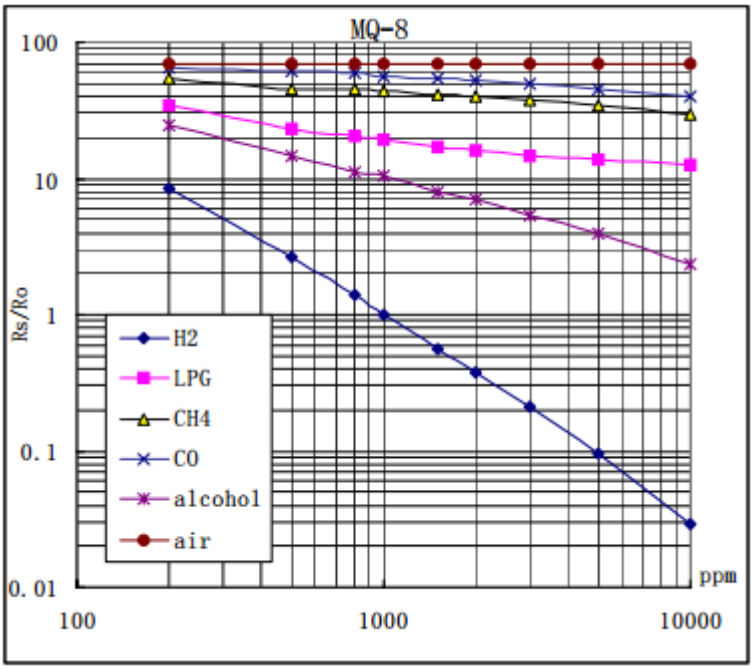


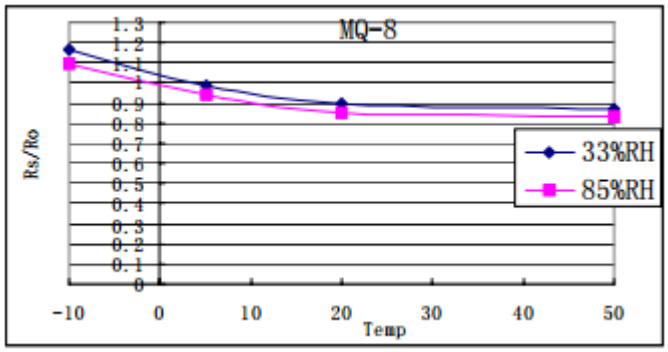
Fig. 1 Fig.2



Εικόνα 50: Τεχνικά Χαρακτηριστικά και διαστάσεις MQ-5



Εικόνα 51: Διάγραμμα MQ-5

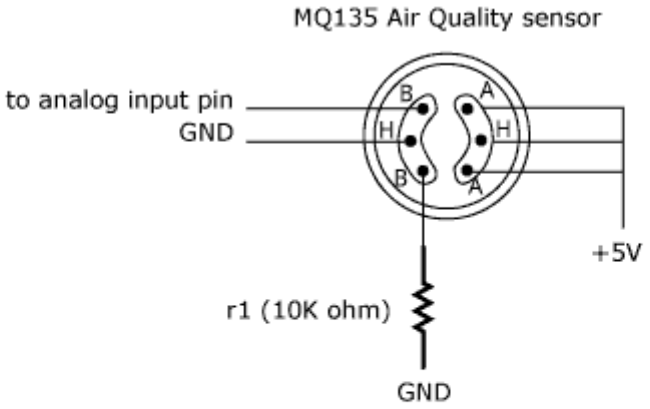


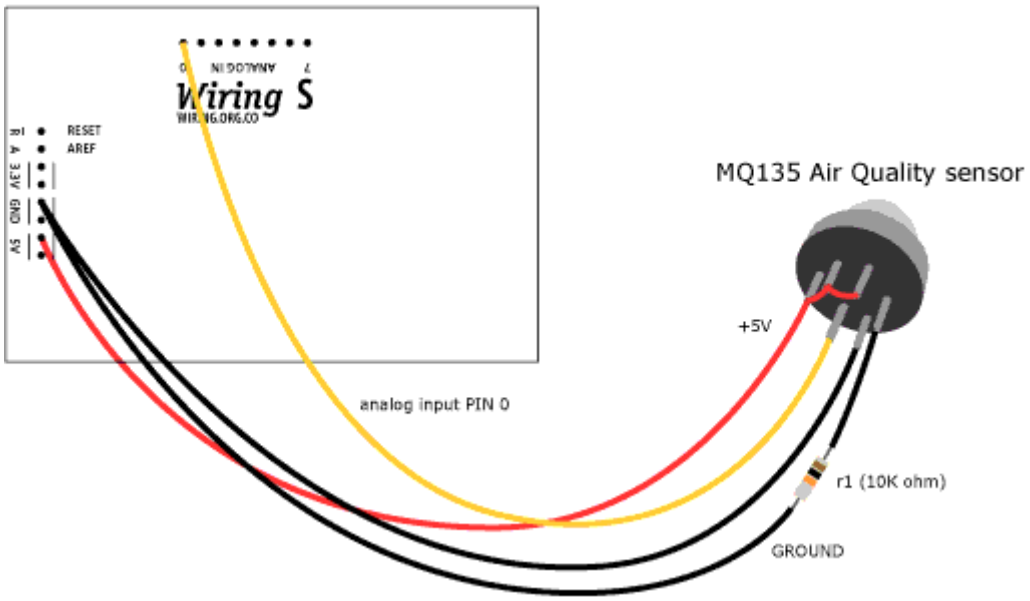
Εικόνα 52: Διάγραμμα MQ-5

2.8.2. Διάγραμμα Συνδεσμολογίας Αισθητήρα Αερίων

Ο αισθητήρας απαιτεί τροφοδοσία 5 Volt DC και απαιτεί χρόνο προθέρμανσης, το οποίο συνεπάγεται ότι πριν τη λήψη τιμών πρέπει να έχει ενεργοποιηθεί το κύκλωμα τροφοδοσίας του τουλάχιστον 40'' πριν. Αυτό γίνεται γιατί στο εσωτερικό του έχει μία αντίσταση η οποία πριν λάβει τις μετρήσεις πρέπει να έχει φτάσει σε συγκεκριμένη θερμοκρασία.

Το ηλεκτρικό διάγραμμα συνδεσμολογίας του αισθητήρα φαίνεται στην εικόνα που ακολουθεί.





Εικόνα 53: Συνδεσμολογία Αισθητήρα MQ-5

2.8.3. Ανάλυση Προγράμματος για Χρήση του Αισθητήρα Αερίων

Το πρόγραμμα και σ' αυτή την περίπτωση είναι απλό αφού δε διαφέρει απ' αυτό του αισθητήρα θερμοκρασίας και υγρασίας. Συνεπώς συνδέουμε τον αισθητήρα σε μία αναλογική θύρα και διαβάζουμε τις τιμές του.

```
int sensorValue;
void setup()
{
  Serial.begin(9600);      // sets the serial port to 9600
}
void loop()
{
  sensorValue = analogRead(0);       // read analog input pin 0
```

```
  Serial.println(sensorValue, DEC);  // prints the value read
  delay(100);                        // wait 100ms for next reading
}
```

Όπως και στους άλλους αισθητήρες, έτσι και σ' αυτή την περίπτωση η επικοινωνία γίνεται με 9600bps. Η αναλογική θύρα η οποία χρησιμοποιείται είναι η μηδέν. Θα μπορούσαμε να βάλουμε άλλη αναλογική θύρα όπως και θα κάνουμε όταν συνδέσουμε όλους τους αισθητήρες μαζί. Η τιμή που λαμβάνεται αποθηκεύεται στη μεταβλητή sensorValue και η διαδικασία επαναλαμβάνεται μετά από 100ms.

2.9. Αισθητήρας Κίνησης

Όπως περιγράψαμε και σε προηγούμενη παράγραφο, στόχος της κατασκευής μας είναι όχι μόνο η λήψη μετρήσεων του χώρου αλλά και η διασφάλιση της μη πρόσβασης σ' αυτόν μη εξουσιοδοτημένου προσωπικού. Για το σκοπό αυτό, πρώτο μέτρο το οποίο λάβαμε, είναι ο αισθητήρας που τοποθετήθηκε στην πόρτα μέσω του οποίου καταλαβαίνουμε αν κάποιος ανοίξει χωρίς να απενεργοποιήσει το σύστημα του συναγερμού. Επιπρόσθετη ασφάλεια επιτυγχάνουμε με έναν αισθητήρα εντοπισμού κίνησης. Ο αισθητήρας που χρησιμοποιήθηκε είναι όμοιος με αυτούς που χρησιμοποιούνται σε συστήματα συναγερμών, δηλαδή ένας Pir Sensor με δυνατότητα ανίχνευσης εντοπισμού κίνησης σε αποστάσεις μέχρι 7 μέτρα.

2.9.1. Τεχνικά Χαρακτηριστικά Αισθητήρα Κίνησης

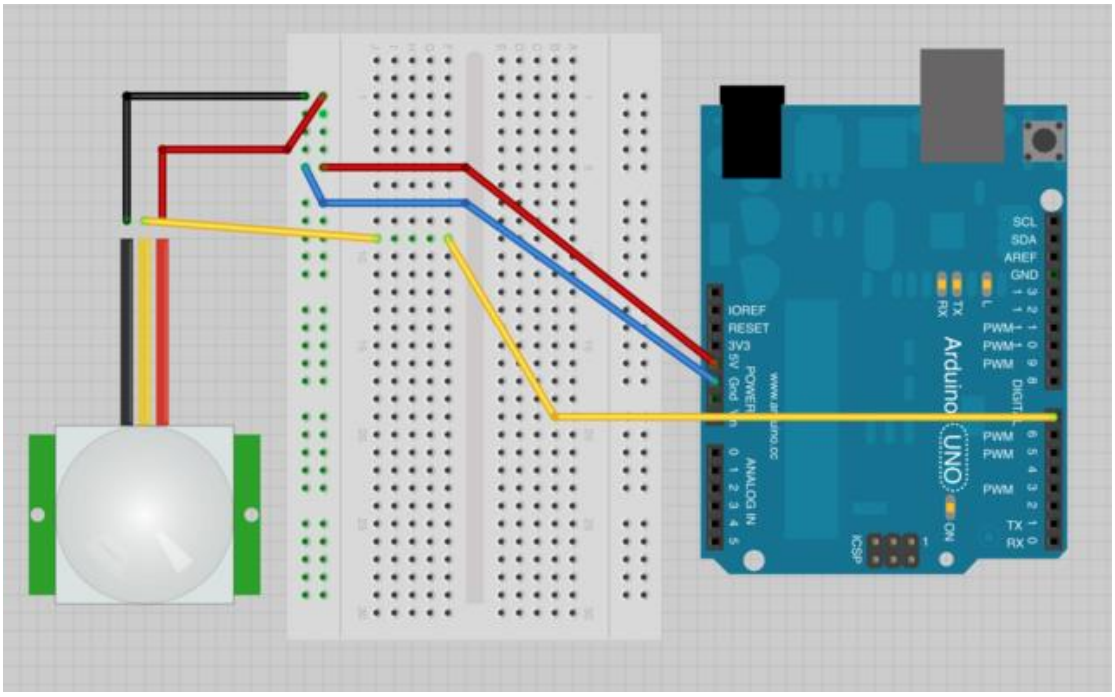
Ο αισθητήρας κίνησης όπως αναφέρθηκε και παραπάνω είναι όμοιος με αυτούς των συστημάτων συναγερμών. Είναι τύπου Pir με δυνατότητα ανίχνευσης κίνησης σε απόσταση 7 μέτρων, τάση τροφοδοσίας DC 4.5- 20 Volt, τάση εξόδου (λογικό 1-0) 3,3 Volt, ρεύμα που καταναλώνει 60μA, χρόνος καθυστέρησης ενεργοποίησης – πυροδότησης 5-200 Sec, γωνία κάλυψης 140ο. Οι διαστάσεις της πλακέτας είναι 3,2cm x 2,4cm x 1,8cm.

Εικόνα 54: Pir Sensor

2.9.2. Διάγραμμα Συνδεσμολογίας Αισθητήρα Κίνησης

Η συνδεσμολογία του αισθητήρα κίνησης είναι πολύ απλή και δε διαφοροποιείται της λογικής των προηγούμενων. Συνολικά απαιτούνται τρεις συνδέσεις, η δεξιά είναι της τάσης τροφοδοσίας όπου απεικονίζεται με κόκκινο χρώμα και συνδέεται με το Pin 5 Volt του Arduino. Η αριστερά είναι της

γείωσης (Gnd) και η μεσαία είναι το ψηφιακό σήμα εξόδου το οποίο το οδηγούμε σε μία ψηφιακή είσοδο του Arduino. Στην εικόνα που ακολουθεί φαίνεται η εν λόγω συνδεσμολογία.



Εικόνα 55: Διάγραμμα Συνδεσμολογίας Pir Sensor

2.9.3. Ανάλυση Προγράμματος για τη χρήση του Αισθητήρα Κίνησης

Το πρόγραμμα για τη χρήση του εν λόγω αισθητήρα είναι πολύ απλό, αφού οι ουσιαστικές ενέργειες οι οποίες πρέπει να κάνουμε είναι να ορίσουμε ένα Pin σαν είσοδο στην οποία θα συνδέσουμε την ψηφιακή είσοδο από τον αισθητήρα. Απ' αυτή την είσοδο, αφού τη διαβάζουμε, θα παίρνουμε την τιμή της η οποία θα είναι λογικό High είτε Low. Το λογικό High σημαίνει ότι ο αισθητήρας έχει ανιχνεύσει κίνηση ενώ αντίστοιχα το Low ότι δεν έχει ανιχνεύσει. Στο πρόγραμμα που ακολουθεί βλέπουμε αυτή τη διαδικασία μέσω μιας συνάρτησης που δημιουργήσαμε γι' αυτό το σκοπό (move()).

```
String movement;
int PirState;
void setup() {
  Serial.begin(9600);
  pinMode(PirSensor, INPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  movement = move();
  Serial.println(movement);
}
//Function move για τον pir sensor
```

```
String move() {
  PirState = digitalRead (PirSensor); // read the PinSensor state

  if (PirState == HIGH) // check if the PirState is HIGH
  {
    return ("Motion Detected!");
    delay(100);
  }

  else
  {
    return ("No motion");
    delay(100);
  }
}
```

2.10. Αισθητήρας RFID

Επεκτείνοντας τη λειτουργικότητα της κατασκευής πέρα από τις μετρήσεις στον χώρο του computer room και την προβολή τους στην εφαρμογή, εισήχθει και μέθοδος ελέγχου πρόσβασης στον χώρο των Server. Για το σκοπό αυτό χρησιμοποιήθηκε ένας αισθητήρας – controller NFC/RFID ο οποίος θα χρησιμοποιείται έξω από την είσοδο του δωματίου και θα επιτρέπει είτε όχι την είσοδο του ατόμου στον χώρο. Για τον έλεγχο πρόσβασης χρησιμοποιούνται κάρτες RFID 13.56 MHz. Η κάθε κάρτα διαθέτει δικό της κωδικό μέσω του οποίου μπορούμε να επιτρέπουμε την είσοδο σε άτομα με συγκεκριμένους αριθμούς κάρτας.

Εικόνα 57: RFID Shield

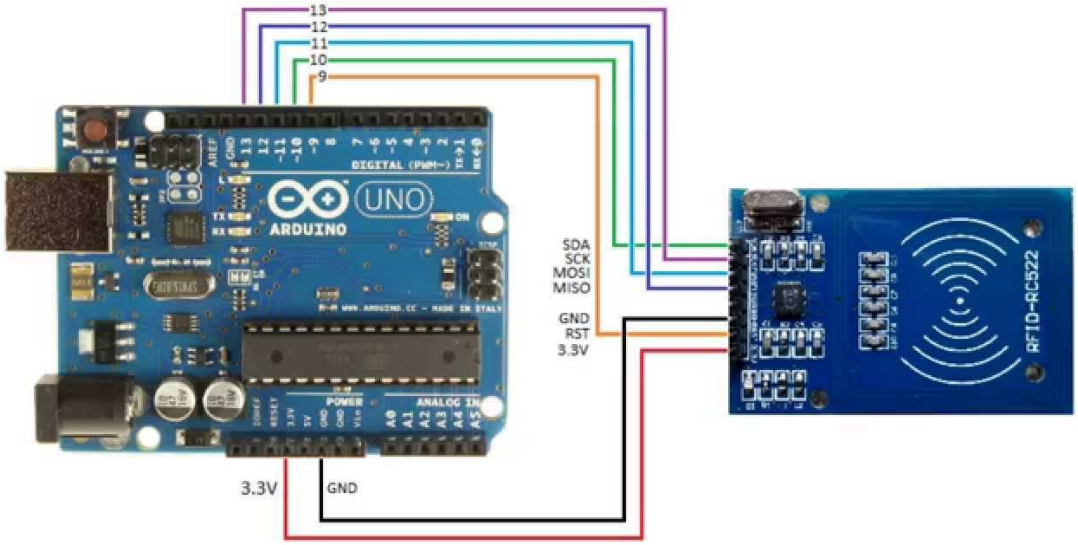
Εικόνα 56: RFID Cards

2.10.1. Τεχνικά Χαρακτηριστικά

Για την υλοποίηση χρησιμοποιήθηκε η πλακέτα MFRC-522 NFC/RFID Controller. Με τη συγκεκριμένη πλακέτα έχουμε τη δυνατότητα τόσο ανάγνωσης όσο και εγγραφής της πλακέτας. Τάση τροφοδοσίας 3,3 Volt DC, ρεύμα 13-26mA , συχνότητα λειτουργίας 13,56MHz. Μέγιστη απόσταση ανάγνωσης κάρτας είναι τα 60mm. Η πλακέτα επικοινωνεί με το Arduino με SPI και μέγιστη ταχύτητα μεταφοράς δεδομένων είναι 10Mbit/sec. Διαστάσεις πλακέτας 40mm × 60mm.
Οι κάρτες οι οποίες χρησιμοποιούνται είναι συχνότητας 13,56MHz κατασκευασμένες από PVC, έχουν μνήμη 8KBit EEPROM.

2.10.2. Διάγραμμα Συνδεσμολογίας RFID

Απαραίτητο για τη χρήση της συγκεκριμένης επέκτασης είναι να συνδεθεί σε SPI θύρες του Arduino. Αυτές διαφέρουν από μοντέλο σε μοντέλο. Η συνδεσμολογία με το Arduino Uno φαίνεται στην εικόνα που ακολουθεί:



Εικόνα 58: RFID και UNO συνδεσμολογία

2.10.3. Ανάλυση προγράμματος για τη χρήση του RFID

Για τον σκοπό του ελέγχου της εισόδου των ατόμων που έχουν συγκεκριμένο αριθμό κάρτας, δημιουργήθηκε η συνάρτηση rfid όπως φαίνεται παρακάτω:

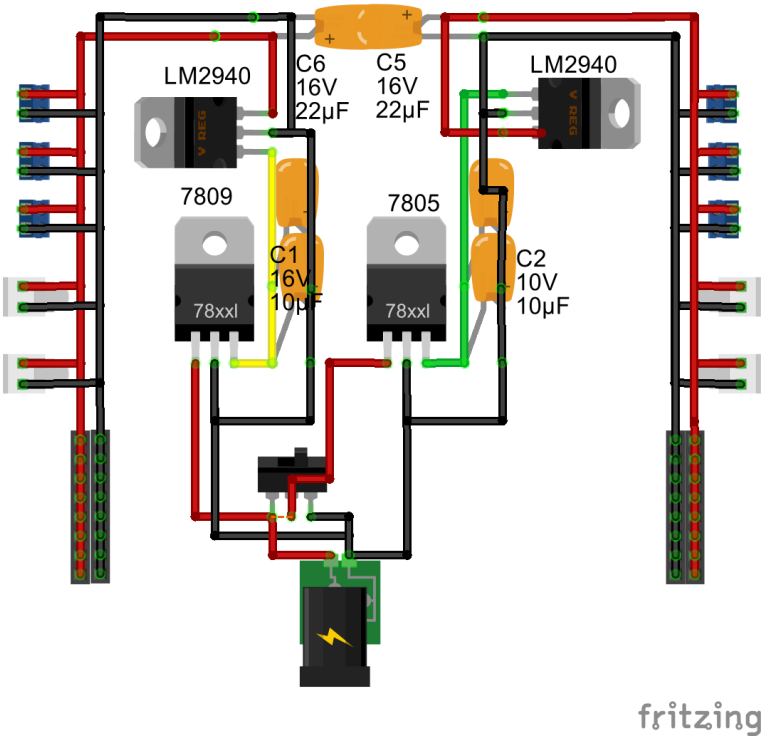
```
boolean rfid() {
  byte status;
  byte data[MAX_LEN];
  byte serial[5];
  int i, j, pos;
```

```
  String name = "";
  boolean check = false;
  // Send a general request out into the aether. If there is a tag in
  // the area it will respond and the status will be MI_OK.
  status = nfc.requestTag(MF1_REQIDL, data);
  if (status == MI_OK) {
    //Serial.println("Tag detected.");
    //Serial.print("Type: ");
    //Serial.print(data[0], HEX);
    //Serial.print(", ");
    //Serial.println(data[1], HEX);
    status = nfc.antiCollision(data);
    memcpy(serial, data, 5);
    //Serial.println("The serial nb of the tag is:");
    for (i = 0; i < 3; i++) {
      name += serial[i];
    }
    name += serial[3];
    //Serial.println(name);
    //Serial.println("rfidcheck function");
    for (int i = 0; i <= 4; i++) {
      //Serial.println(acceptrfid[i][0]);
      if (acceptrfid[i][0] == name) {
        check = true;

        Serial.print("Welcome ");
        Serial.println(acceptrfid[i][1]);
      }
    }
    nfc.selectTag(serial);
    // Stop the tag and get ready for reading a new tag.
    nfc.haltTag();
    delay(100);
  }
  Serial.print("Check=");
  Serial.println(check);
  if (check == true) {
    //??????????????????????edw tha mpei apostoli sti basi an einai
epitixis h sundesi stin basi enter_events
    return true;
  } else {
    //??????????????????????edw tha mpei apostoli sti basi an einai
apotiximeni h sundesi stin basi enter_attempts
    return false;
  }
}
```

2.11. Τροφοδοσία Κατασκευής

Όπως αναλύθηκε στην περιγραφή του κάθε αισθητήρα, απαιτείται για τον καθένα τροφοδοσία 5 Volt η οποία κατά τη φάση της περιγραφής τους παρέχονταν από την ίδια την πλακέτα του Arduino. Κάτι τέτοιο είναι εφικτό στην περίπτωση που χρησιμοποιούμε το πολύ δύο αισθητήρες, λόγω της αδυναμίας της πλακέτας να παρέχει όσο ρεύμα απαιτείται για την κάλυψη όλων των αισθητήρων. Για τον λόγο αυτό κατασκευάστηκε βοηθητική πλακέτα η οποία τροφοδοτείται απευθείας από την πηγή και μέσω αυτής τροφοδοτούνται το σύνολο των αισθητήρων και η πλακέτα του Arduino. Συγκεκριμένα, τα 12 Volt που παρέχονται από την πλακέτα θα μετασχηματίζονται σε 9 Volt για την παροχή ρεύματος στην πλακέτα του Arduino και σε 5 Volt για την παροχή ρεύματος στους αισθητήρες. Σημαντικό για τη λήψη σωστών μετρήσεων είναι ότι τόσο η πλακέτα του Arduino όσο και οι αισθητήρες θα πρέπει να χρησιμοποιούν κοινή γείωση που είναι αυτή της πλακέτας που κατασκευάστηκε.

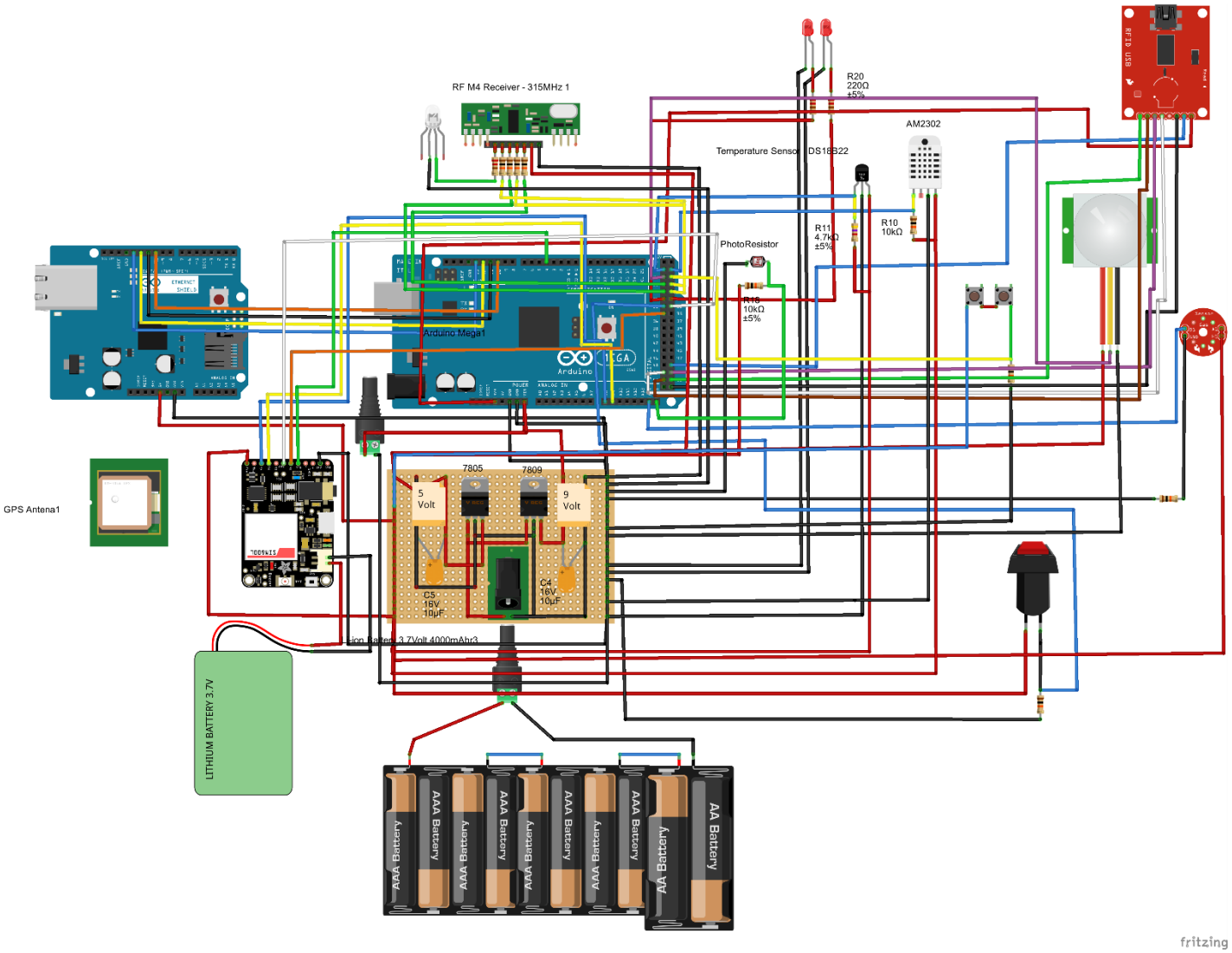


Εικόνα 59: Τροφοδοσία Κατασκευής

Κεφάλαιο 3ο

3. Ανάλυση της κατασκευής στο σύνολό της

Μέχρι τώρα παρουσιάστηκαν όλα τα κομμάτια της κατασκευής μας μεμονωμένα. Η σύνδεση και η λειτουργία του κάθε μέρους της κατασκευής ξεχωριστά είναι το πρώτο βήμα της υλοποίησης αλλά για την υλοποίηση του συνόλου της κατασκευής προκειμένου να παίζουν όλα ταυτόχρονα όπως είδαμε και στο προηγούμενο κεφάλαιο υπάρχουν πολλές τεχνικές δυσκολίες. Η τροφοδοσία του Arduino δεν επαρκεί και έτσι απαιτείται η δημιουργία επιπλέον τροφοδοτικού. Επίσης η πολυπλοκότητα της συνολικής κατασκευής αυξάνεται κατακόρυφα. Στο σχήμα που ακολουθεί φαίνονται οι συνδεσμολογίες όλων των επιμέρους εξαρτημάτων και της τροφοδοσίας μαζί.



Εικόνα 60: Συνολική Συνδεσμολογία Κατασκευής

Στο διάγραμμα που φαίνεται στην παραπάνω εικόνα παρουσιάζεται η νέα τοποθέτηση που πραγματοποιήθηκε για τα shield Ethenret και Fona 808. Όπως είπαμε και στα σχετικά κεφάλαια τα δύο αυτά Shields λειτουργούν σε σειριακές θύρες επικοινωνίας. Το ίδιο ισχύει και με το Shield του Rfid. Συνεπώς το Ethernet συνδέθηκε στις 10, 11, 12 και 13, το Fona 808 A0 και A1 που έχουν τη

δυνατότητα πέρα από αναλογικές να λειτουργήσουν και ως ψηφιακές σειριακές και τέλος το Rfid στις 50, 51, 52 και 53.

Στο διάγραμμα η τροφοδοσία φαίνεται να παρέχεται από δύο διαφορετικές πηγές. Η πρώτη είναι από την μπαταρία Lipo 4000mAh που τροφοδοτεί το Fona 808 και η δεύτερη είναι από εξωτερική μπαταρία αν και σε πραγματική εγκατάσταση η μπαταρία αντικαθίσταται από μετασχηματιστή 220VAc σε 12VDc

Κεφάλαιο 4ο

4. Επισκόπηση των Τεχνολογιών που Χρησιμοποιήθηκαν

Σ' αυτό το κεφάλαιο θα αναφερθούμε στο σύνολο των τεχνολογιών που χρησιμοποιήθηκαν για την εκπόνηση της συγκεκριμένης πτυχιακής εργασίας. Τα αντικείμενα τα οποία θα περιγραφούν είναι οι γλώσσες HTML, CSS, PHP, JavaScript, Bootstrap, Ajax, Apache, MySQL. Ο λόγος για τον οποίο η εφαρμογή μας αποτελείται από το πρόγραμμα το οποίο ο χρήστης θα πρέπει να έχει πρόσβαση από τον υπολογιστή του, δημιουργήθηκε σε μηχάνημα – Server που στήθηκε στον okeano, μία cloud υπηρεσία που παρέχεται από το δημόσιο ακαδημαϊκό δίκτυο Grnet.

4.1. HTML

Τα αρχικά HTML προέρχονται από τις λέξεις HyperText Markup Language, δηλαδή στα ελληνικά Γλώσσα Σήμανσης Υπερκειμένου. Προκύπτει, λοιπόν, ότι η HTML δεν είναι γλώσσα προγραμματισμού. Είναι μια περιγραφική γλώσσα (markup language),δηλαδή ένας ειδικός τρόπος γραφής κειμένου, η κύρια γλώσσα σήμανσης για τις ιστοσελίδες και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων.

Εικόνα 61: HTML Logo

Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από ετικέτες (tags), οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» και «μικρότερο από» (για παράδειγμα <html>, μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη (για παράδειγμα <h1> και </h1>, με την πρώτη να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης (ή σε άλλες περιπτώσεις ετικέτα ανοίγματος και ετικέτα κλεισίματος αντίστοιχα). Ανάμεσα στις ετικέτες, οι σχεδιαστές ιστοσελίδων μπορούν να τοποθετήσουν κείμενο, πίνακες, εικόνες κλπ.

Web browser είναι το πρόγραμμα που διαβάζει τα έγγραφα HTML και τα συνθέτει σε σελίδες που να μπορεί ο άνθρωπος να διαβάσει ή να ακούσει. Ο browser δεν εμφανίζει τις ετικέτες HTML, αλλά τις χρησιμοποιεί για να ερμηνεύσει το περιεχόμενο της σελίδας.

Τα στοιχεία της HTML χρησιμοποιούνται για να συνθέσουν όλους του ιστότοπους. Η HTML επιτρέπει την ενσωμάτωση εικόνων και άλλων αντικειμένων μέσα στη σελίδα, και μπορεί να χρησιμοποιηθεί για να εμφανίσει διαδραστικές φόρμες. Παρέχει τις μεθόδους δημιουργίας δομημένων εγγράφων (δηλαδή εγγράφων που αποτελούνται από το περιεχόμενο που μεταφέρουν και από τον κώδικα μορφοποίησης του περιεχομένου) καθορίζοντας δομικά σημαντικά στοιχεία για το κείμενο, όπως κεφαλίδες, παραγράφους, λίστες, συνδέσμους, παραθέσεις και άλλα. Μπορούν επίσης να ενσωματώνονται εντολές σε γλώσσες όπως η JavaScript, στην οποία θα αναφερθούμε σε επόμενη ενότητα, τα οποία επηρεάζουν τη συμπεριφορά των ιστοσελίδων HTML.

Οι Web browsers μπορούν επίσης να αναφέρονται σε στυλ μορφοποίησης CSS, στο οποίο επίσης θα αναφερθούμε σε άλλη ενότητα, για να ορίζουν την εμφάνιση και τη διάταξη του κειμένου και του υπόλοιπου υλικού. Ο οργανισμός W3C, ο οποίος δημιουργεί και συντηρεί τα πρότυπα για την HTML και τα CSS, ενθαρρύνει τη χρήση των CSS αντί διαφόρων στοιχείων της HTML για σκοπούς παρουσίασης του περιεχομένου.

4.2. CSS

Η CSS (Cascading Style Sheets-Διαδοχικά Φύλλα Στυλ) ή (αλληλουχία φύλλων στυλ) είναι μια γλώσσα υπολογιστή που ανήκει στην κατηγορία των γλωσσών φύλλων στυλ που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης. Χρησιμοποιείται δηλαδή για τον έλεγχο της εμφάνισης ενός εγγράφου που γράφτηκε στις γλώσσες HTML και XHTML, δηλαδή για τον έλεγχο της εμφάνισης μιας ιστοσελίδας και γενικότερα ενός ιστοτόπου. Η CSS είναι μια γλώσσα υπολογιστή προορισμένη να αναπτύσσει στιλιστικά μια ιστοσελίδα, δηλαδή να διαμορφώνει περισσότερα χαρακτηριστικά, χρώματα, στοίχιση και να δίνει περισσότερες δυνατότητες σε σχέση με την html. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται ως απαραίτητη.

Εικόνα 62: CSS Logo

4.3. Bootstrap

Το Bootstrap είναι μια συλλογή εργαλείων ανοιχτού κώδικα για τη δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Περιέχει HTML και CSS, κουμπιά πλοήγησης και άλλα στοιχεία περιβάλλοντος, καθώς και προαιρετικές επεκτάσεις JavaScript. Αναπτύχθηκε από τον Mark Otto και τον Jacob Thornton ως ένα εσωτερικό εργαλείο το οποίο θα χρησιμοποιούνταν για το Twitter. Τον Αύγουστο του 2011 κυκλοφόρησε Twitter Bootstrap ως λογισμικό ανοιχτού κώδικα. Τον Φεβρουάριο του 2012, ήταν το πιο δημοφιλές έργο ανάπτυξης στο GitHub.

Εικόνα 63: Bootstrap Logo

Το Bootstrap έχει σχετικά ελλιπή υποστήριξη για HTML5 και CSS, αλλά είναι συμβατό με όλους τους φυλλομετρητές (browsers). Από την έκδοση 2.0 υποστηρίζει επίσης responsive design.

4.4. JavaScript (JS)

Η JavaScript (JS) είναι διερμηνευμένη γλώσσα προγραμματισμού για ηλεκτρονικούς υπολογιστές. Αρχικά αποτέλεσε μέρος της υλοποίησης των φυλλομετρητών Ιστού, ώστε τα σενάρια από την πλευρά του πελάτη (client-side scripts) να μπορούν να επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται.

Εικόνα 64: JavaScript Logo

Η JavaScript είναι μια γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική. Η σύνταξή της είναι επηρεασμένη από τη C. Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοδοσίας από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία. Οι βασικές αρχές σχεδιασμού της JavaScript προέρχονται από τις γλώσσες προγραμματισμού Self και

Scheme. Είναι γλώσσα βασισμένη σε διαφορετικά προγραμματιστικά παραδείγματα (multi-paradigm), υποστηρίζοντας αντικειμενοστραφές, προστακτικό και συναρτησιακό στυλ προγραμματισμού.

4.5. Ajax

Το Ajax (Asynchronous JavaScript And XML) , δεν αποτελεί κάποια νέα τεχνολογία, αλλά ουσιαστικά είναι συνδυασμός ήδη υπαρχόντων τεχνικών ανάπτυξης ιστοσελίδων, που χρησιμοποιεί πολλές διαδικτυακές τεχνολογίες στην πλευρά client (πελάτη), ώστε να δημιουργήσει ασύγχρονες διαδικτυακές εφαρμογές. Οι τεχνολογίες αυτές είναι:

Εικόνα 65: Ajax Logo

1. Html/XHtml Css για την παρουσίαση της πληροφορίας
2. Δυναμική διαχείριση του περιεχομένου μέσω DOM.
3. Ανταλλαγή πληροφορίας και δεδομένων μέσω XML.
4. Ασύγχρονη συλλογή δεδομένων μέσω του XMLHttpRequest αντικειμένου
5. •avascript για τη σύνθεση όλων των ανωτέρω

Ένα από τα μεγαλύτερα προβλήματα των διαδικτυακών εφαρμογών, είναι το γεγονός ότι προκειμένου ο χρήστης να δει οποιαδήποτε σελίδα ήταν αναγκαία η επανεμφάνιση (reload) ολόκληρης της σελίδας. Στην πλειονότητα των περιπτώσεων, το μεγαλύτερο τμήμα της σελίδας που εμφανιζόταν περιείχε το ίδιο περιεχόμενο με την προηγούμενη, εκτός από ένα συγκεκριμένο τμήμα το οποίο μεταβαλλόταν, προσθέτοντας μία σημαντική καθυστέρηση στην εμφάνιση της ζητούμενης πληροφορίας. Η εμφάνιση του Ajax,έλυσε το ανωτέρω πρόβλημα προσδίδοντας στις διαδικτυακές εφαρμογές την αίσθηση εφαρμογών desktop.Το Ajax, ενεργεί ως διαμεσολαβητής μεταξύ του χρήστη (client) και του εξυπηρετητή (server) ανταλλάσσοντας ασύγχρονα πληροφορίες μεταξύ των δύο, εξαλείφοντας παράλληλα την ανάγκη επαναφόρτωσης της σελίδας, κάθε φορά που ο χρήστης ζητάει κάποια νέα πληροφορία. Με τον τρόπο αυτό, κάθε νέα αίτηση πληροφορίας από τον server καλείται ασύγχρονα από τον χρήστη, ενώ παράλληλα αυτός έχει στη διάθεσή του τη διεπαφή μέχρι την εμφάνιση της πληροφορίας. Επιπλέον, μειώνεται σημαντικά και ο όγκος της πληροφορίας που ανταλλάσσεται μεταξύ χρήστη και εξυπηρετητή.

4.6. PHP

Η PHP είναι μια γλώσσα προγραμματισμού για τη δημιουργία σελίδων web με δυναμικό περιεχόμενο. Αρχικά σήμαινε Personal Home Page(Προσωπική Αρχική Σελίδα), αλλά άλλαξε σύμφωνα με τη σύμβαση GNU σε PHP Hypertext Preprocessor (Προεπεξεργαστής Κειμένου PHP). Μια σελίδα PHP περνά από επεξεργασία από ένα συμβατό διακομιστή του Παγκόσμιου Ιστού (π.χ.Apache), ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο, που θα σταλεί στο πρόγραμμα περιήγησης των επισκεπτών σε μορφή κώδικα HTML. Με λίγα λόγια, παρόλο που ένα αρχείο .php περιέχει κώδικα php, ο browser ποτέ δεν τον βλέπει παρά μόνο τον κώδικα html που προκύπτει όταν τρέχει ο κώδικας php στον server.

Εικόνα 66: PHP Logo

Όλα τα αρχεία ενός website αποθηκεύονται στον server(.php,.html,.css), αλλά δεν εκτελούνται όλα από αυτόν. HTML, CSS και αρχεία εικόνων, στέλνονται απευθείας στον client

browser χωρίς να έχει σημασία το τι περιέχουν. Τα αρχεία PHP διαφέρουν γιατί περιέχουν κώδικα που εκτελείται και τρέχει μόνο στον web server. Στον browser δε στέλνεται ο κώδικας PHP αλλά τα αποτελέσματα του PHP κώδικα.

4.7. MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων που μετρά περισσότερες από 11 εκατομμύρια εγκαταστάσεις. Έλαβε το όνομά της από την κόρη του Μόντυ Βιντένιους, τη Μάιο (αγγλ.My). Το πρόγραμμα τρέχει έναν εξυπηρετητή (server) παρέχοντας πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων. Παρόμοια, φθηνή και ελαφριά βάση δεδομένων είναι και η mSQL που δημιουργήθηκε πριν την MySQL.

Εικόνα 67: MySql Logo

Ο κωδικός του εγχειρήματος είναι διαθέσιμος μέσω της GNU General Public License, καθώς και μέσω ορισμένων ιδιόκτητων συμφωνιών. Ανήκει και χρηματοδοτείται από μία και μοναδική κερδοσκοπική εταιρία, τη σουηδική MySQL AB, η οποία σήμερα ανήκει στην Oracle.

Η MySQL υποστηρίζει προγραμματισμό σε C, Perl, Java (μέσω της συνδεσιμότητας API της βάσης δεδομένων Java [JDBC]), και Python. Με τα εργαλεία MySQL να παρέχουν αυτές τις γλώσσες, είναι δυνατό να δημιουργηθούν πραγματικές εφαρμογές πελατών / κεντρικών υπολογιστών και ιστοσελίδες με ολοκληρωμένες βάσεις δεδομένων χωρίς να ξοδεύονται μεγάλα ποσά.

Η ανέξοδη, σε μερικές περιπτώσεις, ελεύθερη φύση της MySQL δεν έρχεται με ελεύθερο κόστος. Κανένα DBMS (πρόγραμμα διαχείρισης βάσεων δεδομένων) δεν υποστηρίζει το πλήρες εύρος της SQL. Αυτές οι μηχανές στερούνται μερικά χαρακτηριστικά γνωρίσματα που μπορεί να απαιτούνται από τις πιο σύνθετες εφαρμογές. Για μερικές εφαρμογές πρέπει επίσης να εργαστεί κανείς λίγο σκληρότερα από την πλευρά του πελάτη (client) για να ικανοποιήσει τις ανάγκες τις οποίες παίρνει δωρεάν μαζί με τις ακριβές μηχανές βάσεων δεδομένων.

4.8. APACHE

Όταν ένας χρήστης επισκέπτεται ένα ιστότοπο το πρόγραμμα πλοήγησης (browser) επικοινωνεί με έναν διακομιστή (server) μέσω του πρωτοκόλλου HTTP, ο οποίος παράγει τις ιστοσελίδες και τις αποστέλλει στο πρόγραμμα πλοήγησης. Ο Apache είναι ένας από τους δημοφιλέστερους εξυπηρετητές ιστού, εν μέρει γιατί λειτουργεί σε διάφορες πλατφόρμες όπως τα Windows, το Linux, το Unix και το Mac OS X. Το project ρυθμίζεται από κοινού από μια ομάδα εθελοντών που βρίσκονται σε όλο τον κόσμο, χρησιμοποιώντας το Διαδίκτυο και τον Ιστό ως μέσο επικοινωνίας, για να προγραμματίζουν και να αναπτύσσουν τον server και τη σχετική τεκμηρίωσή του. Κυκλοφόρησε υπό την άδεια λογισμικού Apache και είναι λογισμικό ανοιχτού κώδικα. Επιπλέον, οι εκατοντάδες των χρηστών έχουν συμβάλλει στις ιδέες, στον κώδικα και στην τεκμηρίωση του project.

Εικόνα 68: Apache Logo

Ο Apache χρησιμοποιείται και σε τοπικά δίκτυα σαν διακομιστής συνεργαζόμενος με συστήματα διαχείρισης Βάσης Δεδομένων π.χ. Oracle, MySQL. Το XAMPP είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, λογισμικού ανοικτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει τον εξυπηρετητή ιστοσελίδων http Apache, τη βάση δεδομένων MySQL και έναν διερμηνέα για

κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl. Το XAMPP είναι ακρωνύμιο και αναφέρεται στα παρακάτω αρχικά:

1. X(αναφέρεται στο «cross-platform» που σημαίνει λογισμικό ανεξάρτητο πλατφόρμας) (Lamp για Linux, Wamp για Windows)
2. Apache HTTP εξυπηρετητής
3. MySQL
4. PHP
5. Perl

Κεφάλαιο 5ο

5. Επισκόπηση των Εργαλείων που χρησιμοποιήθηκαν

Τα εργαλεία και τα αναπτυξιακά που χρησιμοποιήθηκαν για την εκπόνηση της παρούσας διπλωματικής είναι το WorkBench για τη σχεδίαση και την υλοποίηση της βάσης δεδομένων, το NetBeans για την ανάπτυξη του κώδικά της εφαρμογής, το Arduino IDE για την ανάπτυξη του προγράμματος της πλατφόρμας του Arduino. Τέλος υπηρεσίες οι οποίες χρησιμοποιήθηκαν είναι ο okeanos για τη δημιουργία ενός VM (=Virtual Machine) το οποίο θα παίξει το ρόλο του Server της εφαρμογής μας.

5.1. Fritzig

Για την κατασκευή του ηλεκτρολογικού σχεδίου θα χρησιμοποιηθεί το πρόγραμμα Fritzing. Το Fritzing είναι ένα project που ξεκίνησε σε πανεπιστημιακό επίπεδο και έπειτα λόγω του ανοιχτού λογισμικού γνώρισε μεγάλη απήχηση. Αυτή τη στιγμή είναι το κορυφαίο σε σχεδίαση και απεικόνιση της συνδεσμολογίας πλακετών Arduino με τις βιβλιοθήκες του να εμπλουτίζονται καθημερινά.

Εικόνα 69: Fritzing Logo

Φυσικά οι δυνατότητές του δεν περιορίζονται μόνο στη σχεδίαση των κυκλωμάτων Arduino αλλά μέσω αυτού μπορούμε να δημιουργήσουμε PCB κυκλώματα, να δούμε γραφικά τις συνδεσμολογίες (όπως αυτές παρατίθενται στην παρούσα πτυχιακή), να δημιουργήσουμε το ηλεκτρολογικό σχέδιο και τέλος να τυπώσουμε το αρνητικό του σχεδίου ώστε να το χρησιμοποιήσουμε στη δημιουργία φωτοευαίσθητων πλακετών. Στα πλαίσια της πτυχιακής θα χρησιμοποιηθεί το γραφικό μέρος της σχεδίασης του προγράμματος καθώς και σε λίγα τμήματα το ηλεκτρολογικό.

5.2. Okeanos

Ο Okeanos είναι η πλατφόρμα – υπηρεσία που παρέχεται δωρεάν στους φοιτητές και τους δίνει τη δυνατότητα δημιουργίας δικών τους εικονικών μηχανημάτων τα οποία λειτουργούν στο cloud. Για να μπορούμε να έχουμε πρόσβαση από οπουδήποτε τόσο στην εφαρμογή όσο και στη βάση δημιουργούμε ένα μηχάνημα με Windows Server 2012.

Εικόνα 70: Okeanos

5.3. MySQL Workbench

Το MySQL Workbench είναι ένα εργαλείο μέσω του οποίου μπορεί ο χρήστης οπτικά να σχεδιάσει και να αναπτύξει μία βάση δεδομένων καθώς και να συντηρήσει μία υπάρχουσα. Το Workbench δίνει στον χρήστη πληθώρα δυνατοτήτων και ενεργειών με απλουστευμένο γραφικό τρόπο όπως αυτές θα εκτεθούν παρακάτω. Πρόγονος του Workbench ήταν το MySQL GUI Tools Bundle το οποίο συνέχισε να υποστηρίζεται μέχρι και το 2010. Η πρώτη έκδοση του Workbench ήταν το 2005 με τις εκδόσεις 5.0 και 5.1 . Έπειτα ακολούθησε η 5.2, η 6.0 το 2013, η 6.1 και 6.2 το 2014 και η τελευταία έκδοση η 6.3 στις 5 Μαρτίου του 2015. s

Εικόνα 71: Workbench Logo

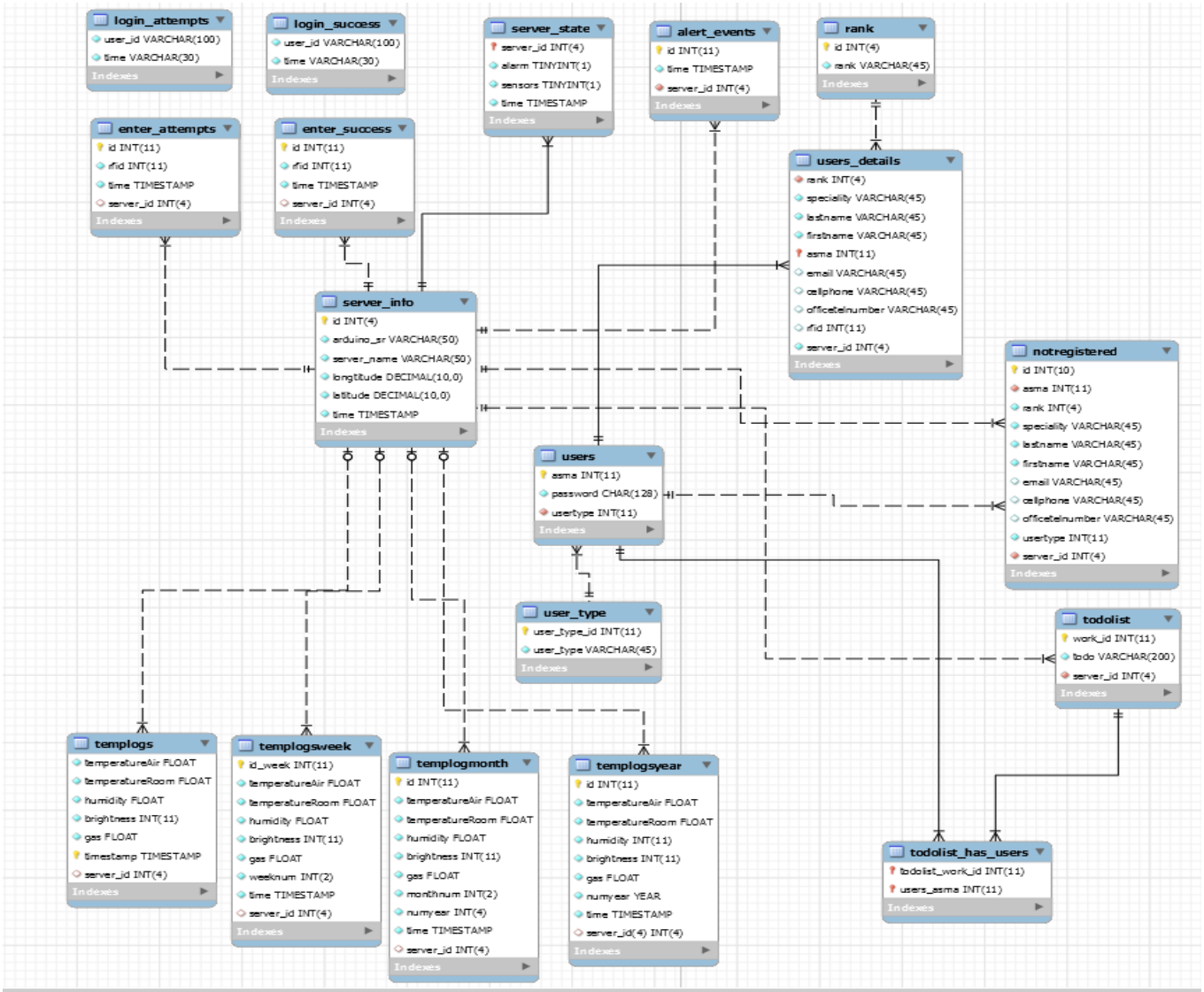
Χαρακτηριστικά του Workbench είναι : SQL Editor, Data modeling, Database administration , Performance monitoring, Database migration.

Το Workbench έχει δύο εκδόσεις. Την "Community Edition" η οποία διανέμεται δωρεάν και την business.

Κεφάλαιο 6ο

6. Ανάλυση του Προγράμματος

Η ανάλυση του προγράμματος θα γίνει σε τρία μέρη. Το πρώτο περιλαμβάνει την ανάλυση της βάσης δεδομένων, το δεύτερο την ανάλυση της εφαρμογής μας στην πλευρά του Server και το τρίτο την ανάλυση του προγράμματος που έχει φορτωθεί στην κατασκευή.



Εικόνα 72: Shema Data Base

6.1. Ανάλυση Βάσης Δεδομένων

Η βάση αποτελείται συνολικά από 15 πίνακες, οι οποίοι είναι οι alert_events, enter_attempts, enter_success, login_attempts, login_success, notregistered, server_info, server_state, templogs, users, users_details, user_type.

Οι πίνακες που χρησιμοποιούνται από την κατασκευή είναι:

Enter_attempts : καταγράφονται τα Rfid των χρηστών που προσπάθησαν να μπουν στον χώρο χωρίς να έχουν κάρτα με έγκυρο αριθμό.

Enter_success: καταγράφονται όλες οι επιτυχημένες προσπάθειες χρηστών για να εισέλθουν στο χώρο.

Server_info: περιέχει τις πληροφορίες για τον κάθε server. Τον κωδικό του, τον κωδικό Serial Number που αντιστοιχεί στην κατασκευή- Arduino, ένα όνομα του Server που είναι μια περιγραφή που την έχουμε εισάγει εμείς. Ακόμα, Logtitude και Latitude που έχει τις συντεταγμένες του server, οι οποίες στέλνονται από την κατασκευή και τέλος τον χρόνο που γίνεται κάθε φορά η ενημέρωση του πίνακα.

Templogs: καταγράφει όλες τις μετρήσεις των αισθητήρων.Εδώ καταγράφονται οι θερμοκρασίες του συστήματος ψύξης και του δωματίου, οι μετρήσεις υγρασίας, τα επίπεδα φωτεινότητας, οι μετρήσεις αερίων και ο χρόνος που γίνεται η καταχώρηση. Ανά μία εβδομάδα θα υπολογίζεται ο μέσος όρος των μετρήσεων που καταγράφονται στον πίνακα αυτόν, θα εκχωρούνται στον πίνακα Templogsweek και στη συνέχεια θα διαγράφονται.

Τα ίδια στοιχεία με διαφορετικό χρονικό προσδιορισμό καταγράφονται και στους πίνακες Templogsweek, Templogsmonth και Templogsyear.

Server_state: καταγράφεται η κατάσταση στην οποία βρίσκεται ο server,δηλαδή αν ο συναγερμός είναι ενεργοποιημένος είτε απενεργοποιημένος και η ώρα που έγινε η τελευταία μεταβολή.

Alert_events: καταγράφει οποιαδήποτε μεταβολή έχουμε σε κάποια παράμετρο που μας ενδιαφέρει να πάρουμε ειδοποίηση. Τέτοιες παράμετροι είναι το αν η υγρασία πέρασε τα όρια που έχουμε θέσει, αν η θερμοκρασία του δωματίου είναι αυξημένη, αν το air-condition δε λειτουργεί, αν ο αισθητήρας αερίων έχει δώσει υψηλές τιμές, αν πήγε να εισέλθει κάποιος στον χώρο χωρίς εγκεκριμένη κάρτα και τέλος αν χτύπησε ο συναγερμός. Όλα αυτά τα στοιχεία συσχετίζονται με το χρόνο που συνέβησαν και τον αριθμό του Server στον οποίο ανήκει η ειδοποίηση.

Οι πίνακες που χρησιμοποιούνται μόνο από την εφαρμογή είναι:

Υπάρχουν πίνακες που χρησιμοποιούνται μόνο από την εφαρμογή. Αυτοί είναι οι login_attempts, login_success, notregistered, user_type, rank, user_details και users.

login_attempts: καταγράφει τις αποτυχημένες προσπάθειες εισαγωγής ενός χρήστη στην εφαρμογή. Σε αυτόν τον πίνακα καταγράφονται το όνομα και ο χρόνος που προσπάθησε να μπει στην εφαρμογή. Αν ο χρήστης προσπαθήσει να μπει πάνω από τρεις φορές τότε η εφαρμογή τον κόβει για τις επόμενες δύο ώρες.

Login_success: καταγράφει επιτυχημένες προσπάθειες εισαγωγής στην εφαρμογή.

Notregistered: είναι ο πίνακας που καταγράφει όλα τα στοιχεία ενός ατόμου που θέλει να δημιουργήσει λογαριασμό. Τα στοιχεία αυτά καταγράφονται στον πίνακα μόνο αν εγκριθούν από το χρήστη, τοποθετούνται στους υπόλοιπους πίνακες χρηστών και αντίστοιχα διαγράφονται από αυτόν.

User_type: Είναι ο πίνακας που περιέχει όλους τους δυνατούς τύπους χρηστών.

Όλοι οι πίνακες φαίνονται στις εικόνες που ακολουθούν.

1 alert_events
2 enter_attempts
3 enter_success
4 login_attempts
5 login_success
6 notregistered
7 rank
8 server_info
9 server_state
10 templogmonth
11 templogs
12 templogsweek
13 templogsyear
14 todolist
15 todolist_has_users
16 users
17 users_details
18 user_type

Εικόνα 73: Όλοι οι πίνακες

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(11)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Όχι			-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 74: Table Alert_event

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(11)		Όχι		auto_increment	
rfid	int(11)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 76:enter_attempts

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(11)		Όχι		auto_increment	
rfid	int(11)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 75: enter_success

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
user_id	varchar(100)		Όχι			
time	varchar(30)		Όχι			

Εικόνα 77: login_attempts

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
user_id	varchar(100)		Όχι			
time	varchar(30)		Όχι			

Εικόνα 78: login_success

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(10)		Όχι		auto_increment	
asma	int(11)		Όχι			-> users.asma ON UPDATE RESTRICT ON DELETE RESTRICT
rank	int(4)		Όχι			
speciality	varchar(45)		Όχι			
lastname	varchar(45)		Όχι			
firstname	varchar(45)		Όχι			
email	varchar(45)		Ναι	NULL		
cellphone	varchar(45)		Ναι	NULL		
officetelnumber	varchar(45)		Ναι	NULL		
usertype	int(11)		Όχι			
server_id	int(4)		Όχι			-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 79: notregistered

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(4)		Όχι			
rank	varchar(45)		Όχι			

Εικόνα 80: rank

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(4)		Όχι			
arduino_sr	varchar(50)		Όχι			
server_name	varchar(50)		Όχι			
longtitude	decimal(10, 0)		Όχι			
latitude	decimal(10, 0)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	

Εικόνα 81: Server_info

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(11)		Όχι		auto_increment	
temperatureAir	float		Όχι			
temperatureRoom	float		Όχι			
humidity	float		Όχι			
brightness	int(11)		Όχι			
gas	float		Όχι			
monthnum	int(2)		Όχι			
numyear	int(4)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 84: templogmonth

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id	int(11)		Όχι			
temperatureAir	float		Όχι			
temperatureRoom	float		Όχι			
humidity	int(11)		Όχι			
brightness	int(11)		Όχι			
gas	float		Όχι			
numyear	year(4)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id(4)	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 83: server_state

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
server_id	int(4)		Όχι			-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION
alarm	tinyint(1)		Όχι	0		
sensors	tinyint(1)		Όχι	0		
time	timestamp		Όχι	CURRENT_TIMESTAMP		

Εικόνα 82: server_state

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
id_week	int(11)		Όχι			
temperatureAir	float		Όχι			
temperatureRoom	float		Όχι			
humidity	float		Όχι			
brightness	int(11)		Όχι			
gas	float		Όχι			
weeknum	int(2)		Όχι			
time	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 87: templogsweek

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
temperatureAir	float		Όχι	0		
temperatureRoom	float		Όχι	0		
humidity	float		Όχι	0		
brightness	int(11)		Όχι	0		
gas	float		Όχι	0		
timestamp	timestamp		Όχι	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP	
server_id	int(4)		Ναι	NULL		-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 86: templogsyear

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
work_id	int(11)		Όχι			
todo	varchar(200)		Όχι			
server_id	int(4)		Όχι			-> server_info.id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 85: todolist

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
todolist_work_id	int(11)		Όχι			-> todolist.work_id ON UPDATE NO_ACTION ON DELETE NO_ACTION
users_asma	int(11)		Όχι			-> users.asma ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 88: todolist_has_users

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
asma	int(11)		Όχι			
password	char(128)		Όχι			
usertype	int(11)		Όχι			-> user_type.user_type_id ON UPDATE NO_ACTION ON DELETE NO_ACTION

Εικόνα 89: users

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
user_type_id	int(11)		Όχι			
user_type	varchar(45)		Όχι			

Εικόνα 90: user_type

Στήλη	Τύπος	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα	Σύνδεση με
rank	int(4)		Οχι			-> rank.id ON UPDATE NO_ACTION ON DELETE NO_ACTION
speciality	varchar(45)		Οχι			
lastname	varchar(45)		Οχι			
firstname	varchar(45)		Οχι			
asma	int(11)		Οχι			-> users.asma ON UPDATE RESTRICT ON DELETE RESTRICT
email	varchar(45)		Ναι	NULL		
cellphone	varchar(45)		Ναι	NULL		
officetelnumber	varchar(45)		Ναι	NULL		
rfid	int(11)		Ναι	NULL		
server_id	int(4)		Οχι			

Εικόνα 91:user_details

6.2. Ανάλυση Προγράμματος Κατασκευής

Το πρόγραμμα ουσιαστικά αποτελεί τη συγχώνευση όλων των επιμέρους προγραμμάτων όπως αναλύθηκαν στο αντίστοιχο κεφάλαιο. Η κατασκευαστική υλοποίηση διαφοροποιεί τις δηλώσεις των Pin αναλόγως το που έχουμε συνδέσει τον κάθε αισθητήρα.

Στο πάνω μέρος του προγράμματος γίνονται οι αρχικοποιήσεις, ακολουθεί το setup και η εκκίνηση της σειριακής επικοινωνίας. Στο κυρίως μέρος υπάρχουν οι κλήσεις των συναρτήσεων, οι οποίες έπονται του κυρίως loop.
Οι συναρτήσεις που υλοποιήθηκαν και οι λειτουργίες τους είναι οι εξής:

int light() : Διαβάζει τον αισθητήρα φωτός και δίνει τιμές στο εύρος 1-1023. Οι τιμές πάνω από 995 σημαίνει ότι ο λαμπτήρας στο δωμάτιο είναι αναμμένος.

boolean doorAlarm() : Ανιχνεύει το αν ο διακόπτης της πόρτας είναι πιεσμένος είτε όχι. Συνεπώς ανάλογα με το αν ο συναγερμός είναι ενεργοποιημένος είτε όχι, ενεργοποιείται το σύστημα.

int getTemp() : Λαμβάνει τιμές από τον αισθητήρα θερμοκρασίας ο οποίος μετράει τη θερμοκρασία του air-condition. Για την πιο αξιόπιστη λήψη μετρήσεων δε λαμβάνεται μόνο μία τιμή αλλά 10 και εξάγεται ο μέσος όρος.

int dhthum() : Λαμβάνει τιμές θερμοκρασίας από τον αισθητήρα υγρασίας- θερμοκρασίας.

int dhttemp(): Λαμβάνει τιμές υγρασίας από τον αισθητήρα.

boolean move() : Διαβάζει τον αισθητήρα κίνησης.

void state() : Ανιχνεύει το πάτημα κάποιου κουμπιού από το τηλεχειριστήριο. Μέσω αυτού μπορούμε να ενεργοποιήσουμε ή να απενεργοποιήσουμε τους αισθητήρες και το συναγερμό.

boolean rfid() : Ανιχνεύει την ύπαρξη rfid κάρτας στον αισθητήρα και σε περίπτωση ανίχνευσης ελέγχει αν αυτή είναι έγκυρη είτε όχι.

void alertalarm() : Διαβάζει τους αισθητήρες κίνησης και τον διακόπτη της πόρτας και αν ο συναγερμός είναι ενεργοποιημένος και κάποιος από τους άλλους δύο τεθεί σε κατάσταση High τότε ενεργοποιείται ο συναγερμός.

boolean alertsensor() : Ελέγχει τις τιμές των αισθητήρων αερίου, τους δύο αισθητήρες θερμοκρασίας, υγρασίας και σε περίπτωση που κάποιος μας δώσει τιμές εκτός ορίων ενεργοποιείται ο συναγερμός.

void led() : Ενεργοποιεί είτε απενεργοποιεί τα led που αντιστοιχούν στις καταστάσεις του συναγερμού και των αισθητήρων.

int gas() : Διαβάζει τις τιμές του αισθητήρα αερίων.

boolean exitEnableAlarm() : Ενεργοποιεί τον συναγερμό μετά το πάτημα κουμπιού από τον χρήστη για έξοδο. Αυτή η μέθοδος μπορεί να χρησιμοποιηθεί εναλλακτικά με το πέρασμα του rfid από τον δέκτη αφού αυτό θα συνεπάγεται και την έξοδο του χρήστη από τον χώρο. Και στις δύο περιπτώσεις ενεργοποιείται ο συναγερμός.

void templogsokeanos() : Στέλνει στη βάση μας τα δεδομένα τα οποία έχουν συλλεχθεί από τους αισθητήρες. Αυτή η ενέργεια εκτελείται κάθε 5 λεπτά και αποστέλλεται ο μέσος όρος των τιμών που λαμβάνονται ανά 1 λεπτό από την εφαρμογή. Ο αριθμός των μετρήσεων που λαμβάνονται για την εξαγωγή της κάθε μέτρησης αποτελεί μέσο όρο 10 μετρήσεων. Με αυτό τον τρόπο μειώνουμε το σφάλμα λανθασμένης μέτρησης. Η μέθοδος αποστολής δεδομένων είναι η GET αφού η εφαρμογή μας τρέχει σε intranet και δεν είναι διαβαθμισμένα. Φυσικά δεν υπάρχει αποστολή κανενός κωδικού παρά μετρήσεων και ονομάτων.

void entersuccessokeanos(String rfid) : Στέλνει στη βάση τις πετυχημένες εισόδους ενός χρήστη στον χώρο. Ο χρόνος εισόδου δε δίνεται από την κατασκευή αλλά δίνεται on update από τη βάση ώστε η κέντρική αναφορά σε ώρα να είναι αυτή του server.

void enterenterattemptsokeanos(String rfid) : Στέλνει στη βάση τις αποτυχημένες εισόδους ενός χρήστη στον χώρο με αριθμό κάρτας που δεν είναι έγκυρος.

void serverinfookeanos(): Εκτελείται κατά την εκκίνηση του μηχανήματος και αποστέλει στη βάση πληροφορίες όπως ο κωδικός του Arduino που είναι τοποθετημένο και συγκεκριμένα ο κωδικός της πλακέτας του Fona 808, το γεωγραφικό μήκος και πλάτος και ο κωδικός του server που αντιστοιχεί.

void serverstateokeanos(boolean alarm, boolean sensors) : Στέλνει στη βάση την κατάσταση ενεργοποίησης είτε απενεργοποίησης του συναγερμού και του συστήματος καταγραφής των μετρήσεων.

void alerteventsokeanos(String description) : Στέλνει στη βάση οτιδήποτε θεωρείται ως alert για την εφαρμογή.

void initialise() : Εκτελεί όλες τις βασικές παραμετροποιήσεις για τη λειτουργία του Fona 808. Αρχικά ενεργοποιεί το module, ξεκλειδώνει την κάρτα, συνδέεται με το δίκτυο κινητής τηλεφωνίας, έπειτα θέτει τους κωδικούς για ενεργοποίηση του GPRS, συνδέεται με το δίκτυο. Σε επόμενη φάση μέσα στο initialize γίνεται ο πρώτος εντοπισμός θέσης μέσω του δικτύου κινητής τηλεφωνίας. Οι συντεταγμένες αυτές παράγονται από τριχοτόμηση των κεραιών και όχι από το GPS που διαθέτει η πλακέτα. Συνεχίζει λαμβάνοντας τιμές για τα επίπεδα της μπαταρίας που είναι συνδεδεμένη σε αυτό, το επίπεδο σήματος σύνδεσης και η σωστή επικοινωνία με το Arduino.

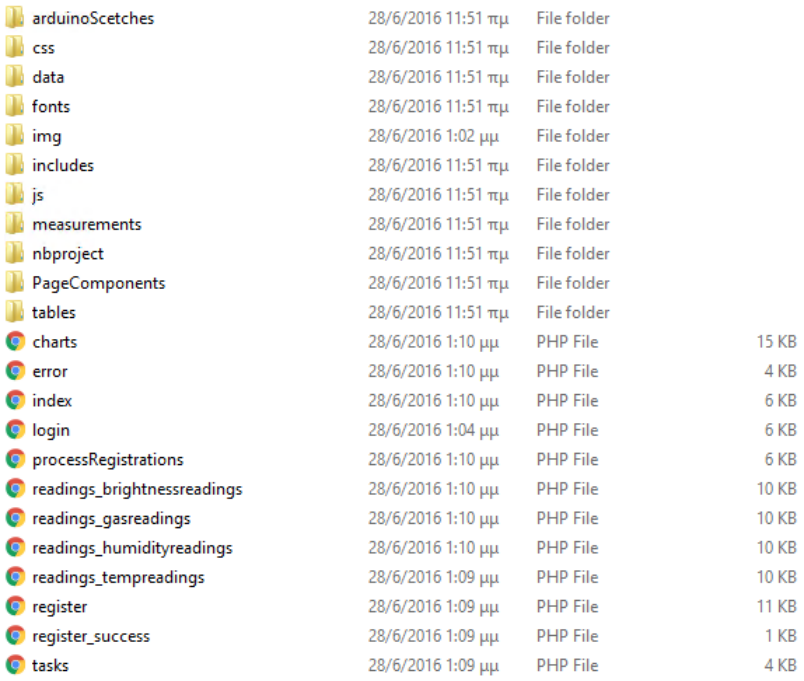
void location() : Η συγκεκριμένη συνάρτηση επιστρέφει το γεωγραφικό μήκος και πλάτος του χώρου εγκατάστασης. Για να λάβουμε σωστή μέτρηση και να μειώσουμε το πιθανό σφάλμα είτε τυχαία αστοχία του υλικού, όπως και μη εξεύρεση δορυφόρων η μέτρηση γίνεται πολλές φορές αξιοποιώντας τόσο το GPS όσο και το GPRS. Αρχικά γίνεται μία μέτρηση με τον αισθητήρα του GPS χωρίς να λαμβάνεται υπόψιν. Αυτή η μέτρηση γίνεται γιατί κατά την πρώτη σύνδεση ο χρόνος που απαιτείται είναι πολλαπλάσιος των επόμενων. Επίσης το ποσοστό αποτυχίας λήψης μέτρησης είτε

λανθασμένης μέτρησης είναι πολύ μεγάλο από κάθε επόμενη φορά. Η προσπάθεια λήψης τοποθεσίας επαναλαμβάνεται μέχρι 20 φορές. Η διαδικασία σταματάει αν έχουμε επιτυχή λήψη μέτρησης για 10 φορές (τα αποτελέσματα ελέγχονται να είναι μέσα στο προκαθορισμένο εύρος τιμών) είτε αν υπερβούμε τις 20 προσπάθειες. Στη συνέχεια από τα 10 αποτελέσματα που πήραμε εξάγουμε τον μέσο όρο. Δεύτερος τρόπος λήψης μετρήσεων είναι μέσω των κεραιών κινητής τηλεφωνίας. Η πιθανότητα σφάλματος σ' αυτή την περίπτωση είναι πάρα πολύ μικρή αλλά στον αντίποδα η ακρίβεια είναι πολύ μικρότερη αφού μπορούμε να έχουμε απόκλιση έως και 150 μέτρα. Αφού λάβουμε 10 μετρήσεις και με το δεύτερο τρόπο εξάγουμε τον μέσο όρο. Στη συνέχεια αν έχουμε λάβει και από τα δύο αποδεκτές μετρήσεις συγκρίνονται μεταξύ τους. Αν οι αποστάσεις μεταξύ τους είναι σχετικά μικρή τότε βγαίνει ο μέσος όρος αυτών ενώ αν η απόσταση είναι μεγαλύτερη και κάτω ενός κατωφλίου λαμβάνεται υπόψιν η μέτρηση του GPS. Σε κάθε άλλη περίπτωση όπου έχουμε λάβει μέτρηση μόνο μία από τις δύο μεθόδους τότε λαμβάνεται υπόψιν μόνο αυτή.

6.3. Ανάλυση Προγράμματος από την πλευρά του Sever

Για την καλύτερη επεξήγηση του προγράμματος, στο CD που συνοδεύει την εργασία περιέχεται ο κώδικας της εφαρμογής με σχόλια σε κάθε σημείο. Επίσης εντός της εφαρμογής υπάρχει menu το οποίο δείχνει με point που δημιουργεί πάνω στο site τη λειτουργία και τη χρήση κάθε menu.

Συνοπτικά στην εικόνα που ακολουθεί φαίνεται η δομή των αρχείων.



Εικόνα 92: Δομή αρχείων

Οι σελίδες είναι οι εξής:

Login: Είναι η σελίδα που κάνει ο χρήστης login στην εφαρμογή χρησιμοποιώντας το ΑΣΜΑ και τον κωδικό του. Πριν την υποβολή της φόρμας, το περιεχόμενο του πεδίου του κωδικού κρυπτογραφείται με τον αλγόριθμο SHA512 με Javascript και το αρχικό περιεχόμενο του πεδίου διαγράφεται.

Έτσι πετυχαίνουμε να μην «ταξιδεύει» ποτέ ο κωδικός του χρήστη στο δίκτυο προτού υποστεί κρυπτογράφηση.

Εν συντομία η διαδικασία η οποία έχει ακολουθηθεί προκειμένου να εξασφαλίσουμε τόσο ένα ασφαλές περιβάλλον εισόδου στην εφαρμογή μας, όσο και την προστασία της ίδιας της εφαρμογής από γνωστές τεχνικές επιθέσεων αναφέρουμε ενδεικτικά :

1. Ο κωδικός δεν ταξιδεύει ποτέ μη κρυπτογραφημένος.

2. Ο ίδιος ο κωδικός έχει υποστεί κατά τη διαδικασία της αίτησης εγγραφής του χρήστη τόσο κρυπτογράφηση από τον αλγόριθμο SHA512 όσο και προσθήκη ενός επιπλέον στόματος ασφάλειας μέσω εφαρμογής salt.

Register: Κατά τη διαδικασία της αποθήκευσης των στοιχείων που υποβάλλουν οι χρήστες μέσω της σελίδας αυτής στην εφαρμογή μας, διενεργούνται έλεγχοι τόσο σε επίπεδο client (αδυναμία υποβολής της φόρμας και ολοκλήρωσης της εγγραφής εάν τα στοιχεία που υποβάλλονται δεν υπακούουν στο προκαθορισμένο format) όσο και σε επίπεδο server.

Register success : Είναι η σελίδα που οδηγούμαστε όταν επιτευχθεί επιτυχή καταχώρηση νέου χρήστη στη βάση μας προκειμένου να εγκριθεί από τον διαχειριστή.

Task : Είναι η σελίδα στην οποία μπορούνε όλοι οι διαπιστευμένοι χρήστες που παρακολουθούν την εφαρμογή από το ίδιο πόστο (ίδιο σταθμό εργασίας) να προσθέτουν και να αφαιρούν εργασίες που πρέπει να γίνουν και αφορούν τα καθήκοντά τους. Οι παραπάνω εγγραφές που προσθαφαιρούν στη σελίδα είναι ορατές προς το παρόν μόνο σε τοπικό επίπεδο αφού αποθηκεύονται στο local storage του Browser (συγκεκριμένο pc- εφαρμογή)

readings tempreadings: Είναι η σελίδα που χρησιμοποιείται για να παρουσιάσει τις εγγραφές που σχετίζονται με τη θερμοκρασία του χώρου. Δίνεται η δυνατότητα στον χρήστη να ελέγχει τι συμβαίνει στο computer room σε πραγματικό χρόνο, ενεργοποιώντας τους διαθέσιμους διακόπτες που φέρουν την ένδειξη «Ανανέωση»

Αντίστοιχοι πίνακες με τον readings_tempreadings είναι οι readings_humidityreadings, readings_gasreadings, readings_brightnessreadings.

processRegistrations: Είναι η σελίδα μέσω της οποία ο admin εγκρίνει την εγραφή ενός user στην εφαρμογή παρέχοντάς του τα ανάλογα δικαιώματα.

Error: Σελίδα που εμφανίζεται σε περίπτωση λάθους κατά τη φάση του Login.

Charts: Είναι η σελίδα που προσφέρει συνολική εικόνα της κατάστασης του server και στατιστικά στοιχεία για παρελθούσες περιόδους.

Φάκελος PageComponents:

1. Αρχείο androidLogin.php δημιουργήθηκε για την επεξεργασία των πιστοποιητικών που αποστέλλονται από την android συσκευή- εφαρμογή.
2. Τα αρχεία arduinoDescriptionPanels1 και arduinoDescriptionPanels2 περιέχουν ένα panel έκαστο με πληροφορίες σχετικά με τους αισθητήρες και τα shields που χρησιμοποιήθηκαν.

Ο λόγος που δεν μπήκαν απευθείας στην index σελίδα αλλά περιλαμβάνονται με include είναι η δυνατότητα εύκολης ανανέωσης και εμπλουτισμού τους.

3. Αρχείο navbar.php είναι το navbar της εφαρμογής μας.
4. Αρχείο navbarmenu.php είναι το πλαϊνό menu
5. Αρχείο productTour.php ανάλογα με το ενεργό url συμπεριλαμβάνει στη σελίδα μία unoriented list <ul> της οποίας το κάθε στοιχείο li περιέχει πληροφορίες οι οποίες απεικονίζονται σε «bubble» δίπλα από το στοιχείο που έχουν οριστεί να περιγράφουν. Σε κάθε σελίδα ανάλογα με το όνομά της –url εμφανίζονται διαφορετικές πληροφορίες.
6. Αρχείο todolist.php περιέχει τον πίνακα ανακοινώσεων καθώς και τον απαραίτητο κώδικα javascript που απαιτείται για να λειτουργήσει. Αυτό έχει ως αποτέλεσμα να μπορεί να συμπεριληφθεί πολύ εύκολα σε όποια σελίδα μας εξυπηρετεί.

Φάκελος CSS: Είναι ο φάκελος που περιέχει όλο το CSS της εφαρμογής μαζί με τα αρχεία του Bootstrap.

Φάκελος measurements: Είναι ο φάκελος που περιέχει όλα τα αρχεία στα οποία «χτυπάει» το Arduino προκειμένου να εισάγει τις τιμές στη βάση. Το Arduino δεν έχει δικαίωμα να εισάγει απευθείας τιμή αλλά μόνο μέσω συναρτήσεων μπορεί αυτό να επιτευχθεί. Ο φάκελος αυτός περιλαμβάνει συνολικά 7 αρχεία.

1. Το add.php καταχωρεί στη βάση και συγκεκριμένα στον πίνακα templogs τις μετρήσεις της θερμοκρασίας δωματίου, υγρασίας, θερμοκρασίας air-condition, το επίπεδο φωτεινότητας, τα επίπεδα αερίου και τον κωδικό του server που λαμβάνονται οι μετρήσεις. Πριν καταχωρηθούν οι μετρήσεις φιλτράρεται και στρογγυλοποιείται η τιμή τους.

2. Το αρχείο alarm.php εισάγει στον πίνακα alert_events όλες τις ειδοποιήσεις που αφορούν παραβίαση της θύρας είτε ανίχνευση κίνησης στον χώρο και το server_id.

3. Το αρχείο alertevents.php καταγράφει τον κωδικό του server και την περιγραφή του alert που έλαβε χώρα.

4. Το αρχείο rfidenterenterattempts.php δέχεται όλες τις τιμές μη εγκεκριμένων καρτών rfid που προσπαθούν να εισέλθουν σε έναν server. Πέρα από τον αριθμό του Rfid αποστέλλεται και ο κωδικός του server όπως είναι λογικό .

5. Το αρχείο Rfidenteraccess.php εισάγει στον πίνακα enter_success τις τιμές των καρτών που εισέλχονται με επιτυχία στον server. Όπως σε κάθε περίπτωση έτσι και εδώ ταυτόχρονα στέλνεται και ο κωδικός του server.

6. Το αρχείο server_info.php κάνει update στις τιμές του γεωγραφικού μήκους και πλάτους και του κωδικού της πλακέτας arduino_sr η οποία αντιστοιχεί στο IMEI της πλακέτας του Fona 808. Φυσικά όλα αυτά γίνονται για τον κωδικό του server που έχει τεθεί αρχικά.

7. Τέλος το αρχείο serverstate.php κάνει update την κατάσταση στην οποία βρίσκεται ο κάθε server. Δηλαδή ανανεώνει την κατάσταση ενεργοποίησης ή μη του συναγερμού και των αισθητήρων.

Φάκελος js:

1. Αρχείο adjustProductBubblePosition.js βρίσκει την πάνω αριστερά γωνία του block που περικλείει το επιλεγμένο element σε απόλυτα Pixels όπως αυτά αντιστοιχούν στην οθόνη που προβάλλεται κάθε φορά η σελίδα.

2. Αρχείο ajax.js περιέχει τις συναρτήσεις function updateLineChartData και updateBarChartData οι οποίες εκτελούν ένα ajax αίτημα στο δοθέν url περνώντας του μέσω της μεθόδου GET τις απαραίτητες παραμέτρους τις οποίες χρειάζεται η συνάρτηση php που τρέχει στις αντίστοιχες σελίδες. Ρόλος των συναρτήσεων αυτών είναι η παραγωγή ενός

response text σε μορφή json το οποίο εκμεταλευόμαστε με την javascript μέθοδο JSON.parse(xhttp.responseText) για να παράγουμε τα αντικείμενα που χρειάζεται η βιβλιοθήκη chart.js για να απεικονίσει γραφικά τα δεδομένα.

3. Το αρχείο bootstraptableparams.js περιέχει τη συνάρτηση chooseTableData η οποία ανάλογα με την παράμετρο που θα περαστεί (χρονικό διάστημα) αλλάζει το url από το οποίο ο πίνακας αντλεί τα δεδομένα του και ταυτόχρονα τα ανανεώνει.

4. Το αρχείο displayTableColumms.js περιέχει την ομώνυμη συνάρτηση η οποία ανάλογα με την παράμετρο που θα της περαστεί (είδος γραφήματος προς απεικόνιση) αποκρύπτει / εμφανίζει τις ανάλογες στήλες στο bootstraptable.

5. Το αρχείο main.js είναι η απαραίτητη javascript για την απεικόνιση της φυσαλίδας περιγραφής στα επιλεγμένα elements κάθε σελίδας.

6. Το αρχείο sha512.js περιλαμβάνει τη συνάρτηση – αλγόριθμο μέσω της οποίας κρυπτογραφούμε τους κωδικούς μας.

7. Το αρχείο forms.js περιλαμβάνει συναρτήσεις. Η συνάρτηση formhash χρησιμοποιείται στην Login page προκειμένου να κρυπτογραφηθεί ο κωδικός του χρήστη πριν την υποβολή της φόρμας. Η συνάρτηση regformhash χρησιμοποιείται στην register page για να διενεργήσει τους απαραίτητους client side ελέγχους (αντιπαραβολή με regular expressions, κενά υποχρεωτικά πεδία κλπ πριν την υποβολή της φόρμας).

8. Το αρχείο alertify.js περιέχει τη βιβλιοθήκη που χρησιμοποιούει για τα custom alerts.

Φάκελος includes:

1. Τα αρχεία db_connect.php και psl-config.php υπάρχουν για τη σύνδεση στη βάση. Η χρήση ενός configuration file είναι χρήσιμη γιατί καθιστά εύκολη τη μετάβαση – σύνδεση σε διαφορετική βάση τροποποιώντας μόνο τα περιεχόμενα αυτού του αρχείου.

2. Το αρχείο registerCheck.php περιλαμβάνει τις συναρτήσεις για την υλοποίηση του register. Ελέγχει server side τα στοιχεία της φόρμας (ύπαρξη χρήστη με τον αριθμό μητρώου που υποβλήθηκε, format δεδομένων που υποβλήθηκαν μέσω της φόρμας και τελική εισαγωγή εγγραφής στον ενδιάμεσο πίνακα notregistered.

3. Το αρχείο process_login.php δημιουργήθηκε για την παρουσίαση πιθανού σφάλματος κατά την υποβολή της login form στον χρήστη.

4. Το αρχείο logout.ph κάνει έξοδο από την εφαρμογή, διαγράφει το cookie και καταστρέφει το session.

5. Το αρχείο functions.php περιέχει την πλειοψηφία των συναρτήσεων php που χρησιμοποιήθηκαν στο project. Ενδεικτικά αναφέρουμε custom session start function, έλεγχος επιθέσεων brute force, ανανέωση δεδομένων μετρήσεων πινάκων/ γραφημάτων.

Φάκελος img: Περιέχει τις εικόνες που χρησιμοποιήθηκαν.

Φάκελος fonts: Περιέχει glyphicons.

Φάκελος data: Περιέχει τα url που επισκέπτονται όλες οι ajax συναρτήσεις που έχουμε δημιουργήσει για τις ανάγκες του project. Οι σελίδες αυτές δεν προστατεύονται από το ασφαλές session διότι δημιουργούσαν προβλήματα σε συνδυασμό με την τεχνολογία ajax γι’ αυτό έχουμε περιορίσει τις διευθύνσεις ip που μπορούν να τις επισκεφτούν.

Κεφάλαιο 7ο

7. Ανάλυση Κόστους Κατασκευής

Το κόστος της συνολικής κατασκευής καθορίζεται από το κόστος των επιμέρους υλικών που χρησιμοποιήθηκαν. Στο κόστος δεν υπολογίζονται τα αναλώσιμα καθώς και επιμέρους μικρά εξαρτήματα.

Πίνακας 2: Μερικά Κόστη Κατασκευής

Όνομα Εξαρτήματος	Τιμή	Ποσότητα
Arduino Mega 2560 ADK Rev3	€54.99	1
Adafruit FONA 808 Shield - Mini Cellular GSM + GPS for Arduino	€61.40	1
Slim Sticker-type GSM/Cellular Quad-Band Antenna - 3dBi uFL	€3.60	1
Antenna GPS 3V Magnetic Mount SMA	€15.20	1
Temperature Sensor - Waterproof (DS18B20)	€5.00	1
Humidity Sensor	€6.50	1
Photo Resistor LDR 5mm	€0.4	1
Load sensor	€ 2	4
Mini Load Sensor	€ 4	1
Κόστος κατασκευής κουτιού	€ 35	1
Polymer Lithium Ion Battery - 3.7v	€17.99	1
Μπαταρία Μολύβδου 12V	€ 20	1
4-Button RF Remote Control	€8.50	1
RF L4 Receiver - 315MHz	€6.10	1

MFRC-522 NFC/RFID Controller Breakout Board	€8.00	1
RFID Card 13.56Mhz	€0.8	4

Το συνολικό κόστος κατασκευής των κύριων μερών ανέρχεται περίπου στα 385€.

Το κόστος της εν λόγω κατασκευής είναι ακόμα αρκετά υψηλό αλλά σε σχέση με τα προϊόντα που αυτή τη στιγμή κυκλοφορούν στην αγορά το κόστος του είναι περίπου 50% χαμηλότερο και ταυτόχρονα τα δεδομένα τα οποία λαμβάνονται αναλογικά με τα υπάρχοντα είναι πολύ περισσότερα.

Κεφάλαιο 8ο

8. Συμπεράσματα – Μελλοντικές Προτάσεις

Φυσικά το project θα μπορούσε να βελτιωθεί προσθέτοντας και άλλες λειτουργίες όπως για παράδειγμα ο έλεγχος με σχετικό αισθητήρα της παροχής ρεύματος στο χώρο, είτε και σε κάθε rack ξεχωριστά. Επίσης μελλοντικά θα μπορούσε οι αριθμοί των καρτών για την είσοδο να δίνονται δυναμικά από την εφαρμογή και όχι να είναι στατικά περασμένα μέσα στον κώδικα της κατασκευής του Arduino. Επίσης θα μπορούσαμε να υλοποιήσουμε και συνάρτηση χρησιμοποιώντας το Fona 808 μέσω της οποίας θα στέλνουμε μηνύματα σε προκαθορισμένους χρήστες όταν συμβαίνει κάποια από τις προϋποθέσεις κινδύνου.

Κεφάλαιο 9ο

9. Βιβλιογραφικές Πηγές

1. **Sheng Wu.Sunm, Xeugang Wang**. *SIM808_Hardware Design*, Shanghai SIMCom Wireless Solutions, 27-03-2014
2. Dallas Semiconductor. *DS18B20 Programmable Resolution Digital Thermometer*
3. Aosong. *Temperature and humidity module AM2301 Product Manual*, Aosong(Guangzhou) Electronics Co.,Ltd, 29-9-2013
4. Susannah **Davidson Pfalzer**, *Arduino A Quick-Start Guide*, Pragmatic Programmers, LLC, 2011
5. Harold Timmis, *Practical Arduino Engineering*, Technology in Action, Friends of Apress, 2011
6. *SIM800 Series_AT Command Manual*, Shagnhai SIMCom Wireless Solutions Ltd. 32-7-2013
7. *SIM808_GPS_Application Note_V1.00* , Shagnhai SIMCom Wireless Solutions Ltd. 26-01-2014
8. *SIM800 Series_IP_Aplication Note_V1.00* Shagnhai SIMCom Wireless Solutions Ltd. 08-01-2013
9. *Adafruit FONA 808 Cellular + GPS Shield for Arduino*, Adafruit learning system, 03-12-2015
10. *Wikipedia*
11. W3C
12.Lynn Beighley & Michael Morrison, *Head First PHP & MySQL*,O'REILLY 2nd Edition