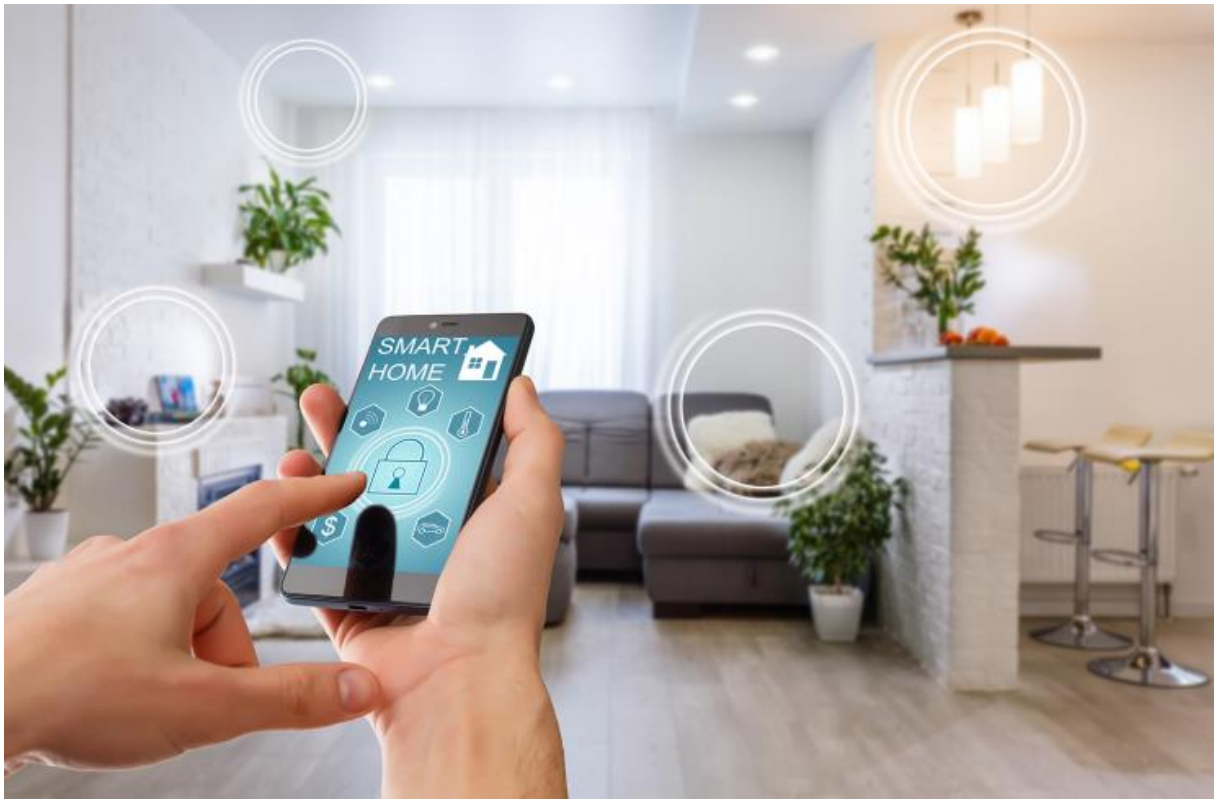
ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ Η/Υ

ΠΜΣ ΜΗΧΑΤΡΟΝΙΚΗΣ

Δ Ι Π Λ Ω Μ Α Τ Ι Κ Η Ε Ρ Γ Α Σ Ι Α



Απομακρυσμένη διαχείριση σπιτιού με τον μικροεπεξεργαστή Arduino

(κωδικός διπλωματικης:)

Γιαννούλης Ιωάννης (Α.Μ. 152)

Επιβλέπων καθηγητής: κ.ΑΣΗΜΟΠΟΥΛΟΣ ΝΙΚΟΛΑΟΣ

Θεσσαλονίκη 2022

Τίτλος πτυχιακής εργασίας στα ελληνικά:

Απομακρυσμένη διαχείριση σπιτιού με τον μικροεπεξεργαστή Arduino

Τίτλος πτυχιακής εργασίας στα αγγλικά:

Remote home management with microcontroller arduino

Ημερομηνία ανάληψης πτυχιακής εργασίας: 22/10/2021

Ημερομηνία περάτωσης πτυχιακής εργασίας: 28/06/2022

ΠΡΟΛΟΓΟΣ

Στη σημερινή εποχή έχουμε όλοι εξοικειωθεί να χρησιμοποιούμε την τεχνολογία του διαδικτύου για να εξυπηρετήσουμε διάφορες ανάγκες της καθημερινότητάς μας όπως οι διαδικτυακές τραπεζικές συναλλαγές, ψώνια από το διαδίκτυο, διαδικτυακά μαθήματα επιμόρφωσης και άλλες ανάγκες που ικανοποιούν πτυχές της ζωής μας. Μια παρόμοια τεχνολογική επανάσταση συμβαίνει τώρα και στον χώρο που ζούμε, δηλαδή στο σπίτι.

Αντικείμενο της εργασίας είναι η σχεδίαση, ανάπτυξη και κατασκευή συστήματος απομακρυσμένης διαχείρισης σπιτιού, βασισμένο στην αρχιτεκτονική Arduino με πρόσβαση στο παγκόσμιο διαδίκτυο μέσω ethernet. Δηλαδή σκοπός της είναι η περιγραφή του συστήματος ως προς το κατασκευαστικό μέρος και το λογισμικό μέρος για την εξ αποστάσεως διαχείριση των ηλεκτρικών συσκευών και εγκαταστάσεων ενός σπιτιού. Με τον όρο διαχείριση εννοούμε τον εύκολο και άμεσο χειρισμό των συσκευών και εγκαταστάσεων ενός σπιτιού. .

Βασική επιδίωξη του συστήματος είναι με το ‘’πάντρεμα’’ της αρχιτεκτονικής Arduino και του παγκόσμιου διαδικτύου internet να μπορούμε να ελέγχουμε το σπίτι μας και με τον όρο έλεγχο, εννοούμε να παρατηρούμε τα φυσικά φαινόμενα που συμβάινουν σπίτι μας, όπως το φως, η θερμοκρασία και η βροχή και να επεμβαίνουμε στις εγκαταστάσεις του σπιτού μας για να ικανοποιήσουμε τις ανάγκες μας.
Υπεύθυνος κατά την εκπόνηση της διπλωματικής εργασίας ήταν ο Καθηγητής κ. Ασημόπουλος Νικόλαος.

ΠΕΡΙΛΗΨΗ

Σκοπός της εργασίας είναι η κατασκευή ενός ηλεκτρονικού συστήματος απομακρυσμένης διαχείρισης σπιτιού, ώστε να επιτευχθεί το λεγόμενο έξυπνο σπίτι.

Οι ανάγκες που οδήγησαν τον άνθρωπο στην επινόηση του έξυπνου σπιτιου είναι οι εξής:

- **εξοικονόμηση χρημάτων και ενέργειας:** στο έξυπνο σπίτι μπορούμε να ρυθμίσουμε τις συσκευές σε συγκεκριμένες ώρες λειτουργίας και να κλείνουν αυτόματα. Ακόμα μπορούμε να παρακολουθούμε την κατανάλωση ενέργειας σε κάθε μία χωριστά, μειώνοντας τους λογαριασμούς ρεύματος.
- **Απόλυτος έλεγχος από απόσταση και άνεση:** να μπορούμε να ελέγχουμε τις συσκευές μας από απόσταση γρήγορα και εύκολα με το πάτημα ενός πλήκτρου, όπως το άνοιγμα των φώτων, κλείσιμο ρολών, το κατέβασμα τέντας, ενεργοποίηση/απενεργοποίηση θέρμανσης και κλιματισμού.
- **Μεγαλύτερη ασφάλεια:** με την εγκατάσταση συστήματος ασφαλείας που περιλαμβάνει ανιχνευτές κίνησης, κάμερες παρακολούθησης και έξυπνες κλειδαριές όπου το σύστημα αυτό συνδέεται στο διαδίκτυο και ο ιδιοκτήτης λαμβάνει ειδοποιήσεις σε περίπτωση παραβίασης

Όλα τα παραπάνω προσφέρουν στον άνθρωπο ηρεμία και ζωή χωρίς άγχος, διότι όλα μπορούν να ρυθμιστούν γρήγορα και απλά από μακριά προσφέροντας ξεγνοιασιά και ανεμελιά. Επίσης η προστασία του περιβάλλοντος είναι δεδομένη γιατί το έξυπνο σπίτι βελτιώνει την ενεργειακή απόδοσή του.

Σε αυτήν την διπλωματική εργασία το έξυπνο σπίτι βασίζεται στην αρχιτεκτονική του μικροελεγκτή Arduino που συνδέεται μέσω ethernet στον παγκόσμιο ιστό για να μπορέσουμε να διαχειριστούμε από μακριά διάφορες συσκευές του σπιτιού μας, όπως πρίζες, τέντες, καλοριφέρ, κλιματιστικό, ρολά παραθύρων. Φυσικά αυτό θα γίνεται με την βοήθεια των αισθητήρων που θα ενημερώνουν το λογισμικό του Arduino για τις στιγμιαίες μεταβολές των φυσικών μεγεθών όπως θερμοκρασία , υγρασία, φως.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Σύστημα έξυπνου σπιτιού, Arduino, Arduino ide, ethernet, shield for Arduino, προγραμματισμός μικροελενγκή, microcontroller, γλώσσα C

ABSTRACT

The purpose of the work is to build an electronic remote home management system to achieve the so-called smart home.

The needs that led man to the invention of the smart home are the following:

• save money and energy: in the smart home we can set the devices to specific operating hours and turn them off automatically. We can also monitor the energy consumption of each one separately, reducing the electricity bills.

• Absolute remote control and comfort: to be able to control our devices remotely quickly and easily at the touch of a button, such as opening the lights, closing the shutters, lowering the awning, turning on / off heating and air conditioning.

• Greater security: with the installation of a security system that includes motion detectors, surveillance cameras and smart locks where this system is connected to the internet and the owner receives notifications in case of violation

All of the above offer the person peace and life without stress, because everything can be adjusted quickly and simply from afar, offering carefree and carefree. Also the protection of the environment is a given because the smart home improves its energy efficiency.

In this dissertation the smart home is based on the architecture of the Arduino microcontroller that connects via ethernet to the World Wide Web so that we can remotely manage various devices in our home, such as sockets, awnings, radiators, air conditioners, window shutters. Of course this will be done with the help of sensors that will inform the Arduino software about the instantaneous changes of physical quantities such as temperature, humidity, light.


My Home
SECURITY SYSTEM
System
DISARMED
LOCKS
Front Door
LOCKED
THERMOSTATS
Living Room Thermostat
72°
LIGHTS
Kitchen
SECURITY
GARAGE
THERMOSTAT
LIGHTS
AUDIO
IMAGES
LOCKS
ENERGY
VIDEO
IRRIGATION
WATER
WELLNESS

ΕΥΧΑΡΙΣΤΙΕΣ

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω όλους αυτούς που συνέβαλλαν να ολοκληρωθεί η παρούσα διπλωματική εργασία. Αρχικά, θα ήθελα να ευχαριστήσω τον Κ. Ασημόπουλο Νικόλαο, για την υποστήριξη στο πρόσωπό μου, με την ανάθεση της παρούσας διπλωματικής, καθώς και για τη συμβολή του στην εργασία. Επίσης, θα ήθελα να ευχαριστήσω την οικογένειά μου, για τις θυσίες τους και την υποστήριξή τους αυτά τα χρόνια που ήμουν φοιτητής ώστε να ολοκληρώσω τις σπουδές μου.

ΔΙΑΡΘΡΩΣΗ ΚΕΙΜΕΝΟΥ ΕΡΓΑΣΙΑΣ

Στο πρώτο κεφάλαιο γίνεται εισαγωγή στην διπλωματική εργασία.

Στο δεύτερο κεφάλαιο, αναλύεται το θεωρητικό υπόβαθρο από το οποίο αντλήθηκαν οι γνώσεις για την επιλογή των εργαλείων που χρησιμοποιήθηκαν για την υλοποίηση του έργου. Δηλαδή αναλύονται ο μικροελεγκτής Arduino που ήταν η βάση της υλοποίησης, το περιβάλλον προγραμματισμού και η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για τον προγραμματισμό του Arduino, καθώς και η τεχνολογία δικτύου web στην οποία βασιστήκαμε.

Στο τρίτο κεφάλαιο, παρουσιάζεται η υλοποίηση του συστήματος έξυπνου σπιτιού δηλαδή τα μέρη από τα οποία αποτελείται. Δίνονται η περιγραφή και τα τεχνικά χαρακτηριστικά του κάθε τμήματος (πλακέτα) που αποτελεί αυτό το σύστημα.

Στο τέταρτο κεφάλαιο, παρουσιάζεται το λογισμικό μέρος του συστήματος. Πιο συγκεκριμένα, δίνεται το σχηματικό διάγραμμα του προγραμματισμού του Arduino και ο κώδικας προγραμματισμού σε γλώσσα C που έχει εγκατασταθεί στον Arduino.

Στο πέμπτο κεφάλαιο, συνοψίζεται το έργο της διπλωματικής εργασίας, και παρουσιάζονται προεκτάσεις που μπορούν να προστεθούν στην παρούσα εργασία.

Κ Ε Φ Α Λ Α Ι Ο 1ο

ΕΙΣΑΓΩΓΗ

1. Εισαγωγή

1.1. Πως προέκυψε η ανάγκη του έξυπνου σπιτιού.

Ο άνθρωπος έχει εντάξει στην καθημερινότητά του τον παγκόσμιο ιστό για να εξυπηρετήσει τις καθημερινές του ανάγκες. Αυτό που έκανε είναι να συνδέσει το παγκόσμιο διαδίκτυο με τις συσκευές του, ώστε να τον εξυπηρετούν καθημερινά γρήγορα και εύκολα. Το πάντρεμα του παγκόσμιου ιστού με τις συσκευές, δίνει το αποτέλεσμα των έξυπνων συσκευών. Δηλαδή συσκευές που θα συλλέγουν δεδομένα και θα αποφασίζουν για τον άνθρωπο χωρίς τον δικό του κόπο, κάνοντας την ζωή του ευκολότερη, π.χ smart phone, smart television, smart gps, smart aircodition κ.α.

Έτσι λοιπόν ο άνθρωπος κατακλύζεται από έξυπνες συσκευές. Η λέξη ‘έξυπνο’ δίνει στην συσκευή την έννοια ότι κατέχει ‘μυαλό’ για να παρατηρεί τα ερεθίσματα που δέχεται και να αποφασίζει για το συμφέρον του ανθρώπου.

Έτσι λοιπόν, ο άνθρωπος θέλησε να στεγάσει όλες τις έξυπνες συσκευές που τον εξυπηρετούν κάτω από μία στέγη, δηλαδή το ‘έξυπνο σπίτι’.

1.2. Τι είναι το έξυπνο σπίτι

Ως έξυπνο σπίτι, ορίζεται ένα σύστημα που διασυνδέονται σε αυτό διάφορα υποσυστήματα (έξυπνες συσκευές), που επικοινωνούν μεταξύ τους μέσω ενός κοινού δικτύου και επιτρέπουν τον απομακρυσμένο έλεγχο τους.

Το έξυπνο σπίτι, κοινώς smart home, είναι λοιπόν ένα σύστημα το οποίο διαθέτει τεχνητή νοημοσύνη. Έχει την δυνατότητα να δέχεται δεδομένα, να τα επεξεργάζεται, να παίρνει αποφάσεις και να τις εκτελεί. Παράλληλα ενημερώνει τον άνθρωπο και πολλές φορές πριν εκτελέσει αποφάσεις, ρωτάει τον ιδιοκτήτη εάν είναι αποδεκτές.

Με αυτό το σύστημα, ο ιδιοκτήτης κατέχει ένα σπίτι σύγχρονο, άνετο, με σημαντική εξοικονόμηση χρόνου και χρήματος.

Η ομαλή λειτουργία του καθίσταται από ένα σύνολο προτύπων τεχνολογίας, που εξελίσσονται διαρκώς, αποκτούν μεγαλύτερες δυνατότητες και εξασφαλίζουν μεγαλύτερες ανέσεις.

Το βασικό πλάνο του έξυπνου σπιτιού είναι μία κεντρική υπολογιστική μονάδα, η οποία παίρνει δεδομένα από διάφορα αισθητήρια, τα οποία με την σειρά τους διαβάζουν διάφορα φυσικά μεγέθη, πχ. Θερμοκρασία, υγρασία κ.α.

Στη συνέχεια η υπολογιστική μονάδα παίρνει τις αποφάσεις εκδηλώνοντας τες με ηλεκτρικά σήματα που δέχονται διάφορες συσκευές του σπιτιού για να ενεργοποιηθούν η να απενεργοποιηθούν.

Οι αποφάσεις που λαμβάνει η κεντρική μονάδα αποτελούν τα σενάρια και παρακάτω αναφέρονται μερικά παραδείγματα σεναρίων.

1.3. Παραδείγματα σεναρίων του έξυπνου σπιτιού.

1. Φωτισμός: Αυτοματοποιημένος φωτισμός που αυξομειώνεται κατά τη διάρκεια της μέρας, προκατασκευασμένα σενάρια φωτισμού, αυτόματη ενεργοποίηση και απενεργοποίηση των φώτων κτλ.

2. Ασφάλεια: Προστασία από βραχυκυκλώματα, πλημμύρες, πυρκαγιά ή οποιαδήποτε βλάβη. Σε αυτήν την περίπτωση, το «Έξυπνο σπίτι» λειτουργεί σαν σύνολο συναγερμών.

3. Έλεγχος θέρμανσης, κλιματισμού, αερισμού: Δυνατότητα να ρυθμιστεί εκ των προτέρων ή εξ αποστάσεως η επιθυμητή θερμοκρασία για την κατοικία. Αντίστοιχες ρυθμίσεις μπορούν να πραγματοποιηθούν και για τον κλιματισμό – αερισμό, ενώ υπάρχει επίσης δυνατότητα αυτόματης ενεργοποίησης του συστήματος εξαερισμού σε περίπτωση υψηλής συγκέντρωσης αερίων ή καπνού στο χώρο. Επιπλέον, η θέρμανση μπορεί να κλείνει αν υπάρξει ανοιχτό παράθυρο ή όποτε θεωρείται περιττή.

4. Έλεγχος ηλεκτρικών περσίδων και τεντών: Θα μπορούσε να εντάσσεται στην προηγούμενη ενότητα, ωστόσο αποτελεί αυτοτελές τμήμα των συστημάτων ελέγχου ενός σπιτιού. Οι τέντες, οι περσίδες και τα παράθυρα μπορούν να ανοιγοκλείνουν ανάλογα με τη θερμοκρασία, το φως, ακόμα και τον αέρα, ρυθμίζοντας απόλυτα τις συνθήκες διαβίωσης.

5. Πολυμέσα: Δυνατότητα διασύνδεσης τηλεοπτικών συσκευών, ηχοσυστημάτων, τηλεφωνικών συσκευών είτε μεταξύ τους, είτε με άλλες συσκευές σε όλο το σπίτι.

Πολύ σημαντικό, το έξυπνο σπίτι αφήνει και περιβαλλοντικό αποτύπωμα με θετικό πρόσημο, γιατί μπορούμε να ελέγξουμε την κατανάλωση ενέργειας από συσκευές

θέρμανσης, κλιματισμού και φωτισμού, καθορίζοντας τον χρόνο λειτουργίας που μας είναι αναγκαίος.

1.4. Το αντικείμενο της διπλωματικής εργασίας

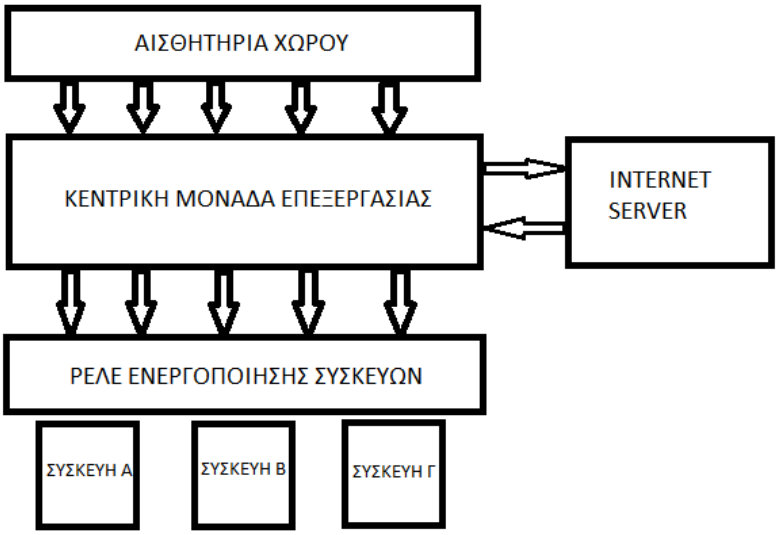
Το αντικείμενο της εργασίας είναι η δημιουργία συστήματος απομακρυσμένης διαχείρισης σπιτιού με τον μικροεπεξεργαστή Arduino, συνδεδεμένο με τον παγκόσμιο ιστό μέσω πρωτοκόλλου ethernet.

Το σύστημα αποτελείτε από τα εξής τμήματα:

- τα αισθητήρια που είναι , αισθητήρια θερμοκρασίας (ενός αισθητηρίου για χαμηλή θερμοκρασία και ενός υψηλής), 2 φωτοαντιστάσεις για μέτρηση φωτός εξωτερικά και εσωτερικά του σπιτιού και ένα αισθητήριο υγρασίας για την ανίχνευση βροχερού καιρού
- η αναπτυξιακή πλακέτα μαζί με τον Arduino, όπου εφαρμόζεται η εξωτερική τροφοδοσία και στον Arduino είναι εγκατεστημένος ο κώδικας σε γλώσσα C, ώστε να εκτελείτε η διαδικασία του συναγερμού με βάση τον κώδικα
- η πλακέτα shield που τοποθετείται στην αναπτυξιακή πλακέτα και αναλαμβάνει την σύνδεση του Arduino με το παγκόσμιο δίκτυο μέσω ethernet, για να επικοινωνεί ο Arduino με την τερματική συσκευή με την οποία ελέγχουμε το σπίτι, πχ smart phone, tablet, pc
- την πλακέτα των ρελέ όπου τα ρελέ θα ενεργοποιούνται ή όχι ανάλογα με τα ηλεκτρικά σήματα στις εξόδους τις κεντρική πλακέτας
- Led’s, η αλλιώς λεντάκια που θα συνδέονται σε σειρά με μία αντίσταση σε ράστερ και το ένα άκρο τους θα είναι συνδεδεμένα σε κάθε ένα από τα ρελέ. Τα λεντάκια θα παρομοιάζουν κάθε μία συσκευή που θέλουμε να ελέγξουμε σε αυτήν την εργασία, δλδ καλοριφέρ, κλιματιστικό, φώτα εσωτερικά, παντζούρια και τέντα μπαλκονιού.

Τα τμήματα του συστήματος έξυπνου σπιτιού παρουσιάζονται αναλυτικά σε επόμενο κεφάλαιο.

1.5. Μπλοκ-διάγραμμα συστήματος έξυπνου σπιτιού



Από το μπλοκ διάγραμμα ξεκινάμε από τα αισθητήρια χώρου που αντιλαμβάνονται διάφορα φυσικά μεγέθη και στέλνουν τα δεδομένα στην κεντρική μονάδα επεξεργασίας. Αυτή με την σειρά της επεξεργάζεται τα δεδομένα και επικοινωνεί μέσω διαδικτύου με κάποια τερματική συσκευή όπως smart phone, tablet, υπολογιστής για να διαχειριστούμε από μακριά τις συσκευές Α,Β και Γ.

Αυτό επιτυγχάνεται πάλι μέσω server, δίνοντας τις εντολές από την τερματική συσκευή στην κεντρική μονάδα που με την σειρά της ανοιγοκλείνει τις συσκευές μέσω των ρελε ενεργοποίησης-απενεργοποίησης.

Κ Ε Φ Α Λ Α Ι Ο 2ο

ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

2. Θεωρητικό υπόβαθρο

2.1. <u>Σύστημα απομακρυσμένης διαχείρισης σπιτιού</u>

Για να υλοποιηθεί ένα έξυπνο σπίτι απαιτούνται τεχνολογικές πλατφόρμες και ψηφιακά εργαλεία που να επικοινωνούν μεταξύ τους με πρωτόκολλα.

Σε αυτήν την εργασία χρησιμοποιείται η πλατφόρμα του Arduino και για να πραγματοποιήσει το σύστημά μας πρέπει να προγραμματιστεί με το εργαλείο της γλώσσας C.

Στην συνέχεια είναι απαραίτητο να δημιουργηθεί μία ιστοσελίδα η οποία θα ανεβαίνει στο server και θα είναι διαθέσιμη από κάποια τερματική συσκευή όπως το smart phone. Αυτή η ιστοσελίδα θα προγραμματιστεί με το εργαλείο της γλώσσας html.

Από το smart phone μέσω αυτής της ιστοσελίδας θα διαχειριζόμαστε τις συσκευές του σπιτιού, άρα θα υπάρχει μία αμφίδρομη επικοινωνία της τερματικής συσκευής (smart phone) και της κεντρικής μονάδας Arduino, που θα βρίσκεται στο χώρο του σπιτιού, και αυτή η επικοινωνία θα υποστηρίζεται με το πρωτόκολλο http. Επίσης ο Arduino συνδέεται με τον server internet μέσω πρωτοκόλλου ethernet.

Στο κεφάλαιο αυτό, σαν θεωρητικό υπόβαθρο, θα αναφερθούν και θα αναλυθούν τα πρωτόκολλα , πλατφόρμες και εργαλεία που απαιτούνται για την υλοποίηση του συστήματος έξυπνου σπιτιού.

2.2 Η υπολογιστική πλατφόρμα Arduino

Στην ενότητα αυτή θα γίνει αναλυτική αναφορά στην υπολογιστική πλατφόρμα Arduino, τα χαρακτηριστικά , τις δυνατότητες και τις λειτουργίες της. Μέσω τις βραχείας τεχνολογικής εξέλιξης οι πλατφόρμες Arduino αποτελούν ένα πανίσχυρο εργαλείο για απλές και πολύπλοκες εφαρμογές σε διαφορετικά πεδία της ανθρώπινης εξέλιξη

2.2.1 *Ιστορικά στοιχεία*

Το 2005, στην Ivrea της Ιταλίας (στην ίδια περιοχή με την εταιρεία υπολογιστών Olivetti), ένα έργο άρχισε να δημιουργείτε, μια συσκευή για τον έλεγχο σχεδίων, χτισμένο από μαθητές με λιγότερα έξοδα από ό, τι με άλλα πρωτότυπα συστήματα που ήταν διαθέσιμα εκείνην τη περίοδο. Ο Massimo Banzi και ο David Cuartielles ονόμασαν το έργο τους Arduin of Ivrea, «Arduino» που είναι ιταλικό όνομα, μεταφράζεται ελεύθερα ως «γενναίος φίλος» και έτσι σιγά-σιγά ξεκίνησαν την παραγωγή του σ ένα μικρό εργοστάσιο της περιοχής . Το Arduino είναι μια παραγόμενη έκδοση της πλατφόρμας ανοικτού κώδικα «Wiring Platform». Ο Κολομβιανός καλλιτέχνης και προγραμματιστής Hernando Barragán δημιούργησε τη συρμάτωση «Wiring» ως μια Μεταπτυχιακή διπλωματική εργασία στο Interaction Design Institute Ivrea υπό την εποπτεία του Massimo Banzi και του Casey Reas. Η Καλωδίωση «Wiring» βασίστηκε στην επεξεργασία και το ολοκληρωμένο περιβάλλον ανάπτυξης που είχε δημιουργηθεί από τον Casey Reas και τον Ben Fry.

Εικόνα 2.2.1 το «logo» της εταιρείας

2.2.2 *Αναφορά στην υπολογιστική πλατφόρμα Arduino*

Γενικότερα, το Arduino θα λέγαμε ότι είναι ένα εργαλείο που μπορούμε να κατασκευάσουμε ένα υπολογιστικό σύστημα με την έννοια ότι αυτό θα ελέγχει συσκευές του φυσικού κόσμου, σε αντίθεση με τον κοινό Ηλεκτρονικό Υπολογιστή. Βασίζεται σε ευέλικτο, εύκολο στη χρήση υλικό και λογισμικό, σε μια αναπτυξιακή πλακέτα που ενσωματώνει επάνω έναν μικροελεγκτή και συνδέεται με τον Η/Υ για να προγραμματιστεί μέσα από ένα απλό περιβάλλον ανάπτυξης. Με το Arduino δημιουργούνται συσκευές οι οποίες εξυπηρετούν διάφορους σκοπούς έχοντας την δυνατότητα να δέχονται ερεθίσματα από το

περιβάλλον τους (μέσω των αισθητήρων) και να αντιδρούν ανάλογα με το πως έχουν προγραμματιστεί.

Τα παραπάνω δεν ακούγονται πρωτότυπα. Υπάρχουν και άλλες πλατφόρμες και υλοποιήσεις που μπορούν να κάνουν τα ίδια πράγματα. Ποια είναι η ειδοποιός διαφορά; Το Arduino βασίζεται σε τεχνολογίες ανοιχτού κώδικα. Μπορεί να κατασκευαστεί από τον καθένα, μπορεί να ενσωματωθεί σε συσκευές ακόμα και για εμπορικούς σκοπούς και το σημαντικότερο είναι ότι υπάρχει μια ολόκληρη κοινότητα που χρησιμοποιεί το Arduino σε κατασκευές άρα υπάρχει μεγάλος όγκος ελεύθερης πληροφορίας. Γενικά, τα Projects στον εν λόγω Μικροελεγκτή μπορούν να είναι αυτόνομα (σε επίπεδο hardware) ή να επικοινωνούν με κάποιο software στον Η/Υ του προγραμματιστή (προγράμματα όπως τα Flash, Processing, MaxMSP).

Στην αρχητεκτονική της υπολογιστικής πλατφόρμας του Arduino υπάρχει ένας μικροελεγκτής atmel AVR(μοντέλο atmega 328), αντί του chip FTDI ώστε αυτό να επιτρέπει τόσο την πιο γρήγορη ταχύτητα μεταφοράς όσο και τη γρήγορη σειριακή επικοινωνία και συμπληρωματικά εξαρτήματα για την διευκόλυνση του χρήστη στον προγραμματισμό και την ενσωμάτωση του σε άλλα κυκλώματα. Όλες οι πλατφόρμες περιλαμβάνουν ένα γραμμικό ρυθμιστή τάσης 5V και έναν κρυσταλλικό ταλαντωτή 16MHz (ή κεραμικό αντηχητή σε κάποιες παραλλαγές).

Ο Atmega328 περιέχει 14 ψηφιακούς ακροδέκτες εισόδου-εξόδου(«I/O»), 6 αναλογικές εισόδους και 6 ψηφιακές εξόδους και έναν ταλαντωτή των 16 MHz. Η σύνδεση για προγραμματισμό του μικροελεγκτή γίνεται με «usb» καλώδιο, ενώ υπάρχει και υποδοχή σύνδεσης με ρεύμα, καθώς και δυνατότητα εντός του κυκλώματος, σειριακού προγραμματισμού-ICSP («In-Circuit Serial Programming») και ένα κουμπί επανεκκίνησης (reset) σε περίπτωση που βραχυκυκλώσει η πλατφόρμα χωρίς την θέλησή μας .

Ο μικροελεγκτής (atmega 328) ενός Arduino συνήθως προγραμματίζεται εκ των προτέρων (εσωτερικός κώδικας) ώστε να παρέχει κάποιο φορτωτή εκκίνησης (BootLoader). Ο φορτωτής εκκίνησης υπάρχει ώστε να απλοποιεί την διαδικασία της αποθήκευσης των προγραμμάτων στην Flash Memory του Arduino μέσω σειριακής USB θύρας.

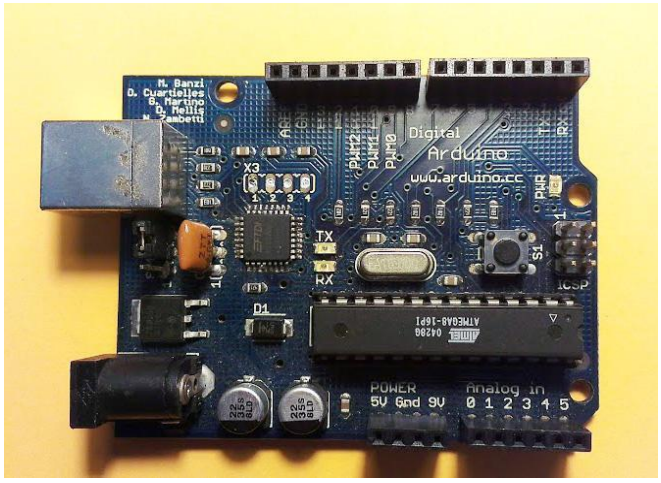
Επιπλέον, η γλώσσα προγραμματισμού, οι διάφορες βιβλιοθήκες και το ολοκληρωμένο περιβάλλον ανάπτυξης που υπάρχουν για τον προγραμματισμό της πλατφόρμας Arduino αποτελούν ανοιχτό λογισμικό προσφέροντας έτσι ανεκτίμητη γνώση σε όλους.

2.2.3 *Βασικά πλεονεκτήματα πλατφόρμας Arduino*

- Οικονομική: Η πλατφόρμα Arduino αποτελεί οικονομική λύση διότι είναι φθηνότερη. Επιπλέον, είναι αρχιτεκτονικά ανοιχτή και μπορεί ο οποιοσδήποτενα την αναπτύξει από μόνος του.

- Μεταφέρσιμη: Σε σχέση με τις υπάρχουσες πλατφόρμας στο εμπόριο η πλατφόρμα Arduino παρέχει πλήρη μεταφερσιμότητα με αποτέλεσμα να μπορείνα προγραμματιστεί στα περισσότερα λειτουργικά συστήματα.

- Επεκτάσιμη: Το υλικό και το λογισμικό της πλατφόρμας Arduino είναι ανοιχτάκαι ελεύθερα για όλους. Καθημερινά, χιλιάδες υποστηρικτές του ελεύθερουλογισμικού αναπτύσσουν διάφορες βιβλιοθήκες για την υποστήριξη τηςπλατφόρμας. Παράλληλα, τόσο η αρχιτεκτονική όσο και το υλικό της πλατφόρμας εξελίσσονται συνεχώς.



Εικόνα 2.2.3 Η πρώτη υπολογιστική πλατφόρμα Arduino τέλη του 2005

Το Arduino αποτελείται από δύο κύρια μέρη ,την υπολογιστική πλατφόρμα Arduino, η οποία είναι το κομμάτι του hardware πάνω στο οποίο εργάζεται ο κατασκευαστής όταν πραγματοποιεί μία κατασκευή, ενώ το δεύτερο τμήμα είναι το Arduino IDE, το κομμάτι του λογισμικού που τρέχει στον υπολογιστή. Το IDE χρησιμοποιείται για να δημιουργηθεί ένα «sketch» (ένα μικρό πρόγραμμα στον υπολογιστή) που φορτώνεται στον μικροελεγκτή της υπολογιστικής πλατφόρμας Arduino.

2.2.4 *Μοντέλα Arduino*

Υπάρχουν πολλές πλατφόρμες Arduino που έχουν αναπτυχθεί και όπου η κάθε μία είτε αποτελεί εξέλιξη κάποιας άλλης, είτε έχει αναπτυχθεί για κάποιο συγκεκριμένο σκοπό. Τρεις από τις βασικές πλατφόρμες, που είναι οικονομικές και καλύπτουν πολλές ανάγκες και απαιτήσεις πάνω στην ανάπτυξη συστημάτων παρουσιάζονται περιληπτικά παρακάτω:

2.2.4.1 *Υπολογιστική πλατφόρμα Arduino Leonardo*

Το Arduino Leonardo χρησιμοποιεί τον ATmega32u4 έχει 20 ψηφιακές θύρες εισόδου και εξόδου (I/O) εκ των οποίων οι 7 μπορούν να παράγουν P.W.M, 12 αναλογικές θύρες, 16MHz ταλαντωτή κρυστάλλου και κουμπί «reset». Επίσης έχει ενσωματωμένο («built-in») usb με το οποίο μπορεί να χρησιμοποιηθεί ως πληκτρολόγιο και ποντίκι

2.2.4.1 Arduino Leonardo

2.2.4.2 *Υπολογιστική πλατφόρμα Arduino Uno*

Το Arduino uno βασίζεται στον μικροελεγκτή της Atmel Atmega 328. Αποτελείται συνολικά από 14 ψηφιακές εισόδους/εξόδους (εκ των οποίων οι 6 έχουν την δυνατότητα να λειτουργήσουν ως έξοδοι παλμών PWM), 6 αναλογικές εισόδους, έναν κρύσταλλο 16MHz, μια θύρα USB, βύσμα τροφοδοσίας (power jack), ICSP header και ένα κουμπί reset.

2.2.4.2 Arduino Uno

2.2.4.3 *Υπολογιστική πλατφόρμα Arduino ATmega 2560*

Το Arduino Mega 2506 είναι η πιο ισχυρή και πιο πρόσφατη υπολογιστική πλατφόρμα Arduino με πάρα πολλές δυνατότητες και είναι η αναβάθμιση του Arduino Mega. Χρησιμοποιεί τον Atmega2560 μικροελεγκτή της ATMEL. Έχει 54 ψηφιακές θύρες εισόδου και εξόδου (I/O) εκ των οποίων οι 15 μπορούν να παράγουν «8-bit» P.W.M και άλλες 2 θύρες μπορούν να χρησιμοποιηθούν για «I2C» επικοινωνία καθώς και 6 εξωτερικά interrupt, 16 αναλογικές θύρες, 16MHz ταλαντωτή κρυστάλλου, κουμπί «reset», 4 σειριακές θύρες για «hardware» κεφαλή, και 4 θύρες «SPI» για την σύνδεση με άλλα περιφερειακά ή πλακέτες.

2.2.4.3 Arduino ATmega 2560

2.2.4.4 *Άλλες υπολογιστικές πλατφόρμες Arduino*

Άλλες υπολογιστικές πλατφόρμες Arduino που κυκλοφορούν στην αγορά και καλύπτουν και αυτές ανάγκες συστημάτων είναι:

2.2.4.4 arduino nano

2.2.4.4 arduino frio

2.2.4.4 arduino diecimila

Υ.Γ Υπάρχουν και άλλες εκδόσεις πλατφόρμων.

2.2.5 *Arduino shields*

Η υπολογιστική πλατφόρμα του Arduino από μόνη της δεν εκτελεί κάποια συγκεκριμένη λειτουργία και δεν αποτελεί ολοκληρωμένο σύστημα. Όμως μετά τον προγραμματισμό της αποτελεί την βάση ενός συστήματος που για να ολοκληρωθεί χρειάζεται να δημιουργήσουμε περαιτέρω hardware.

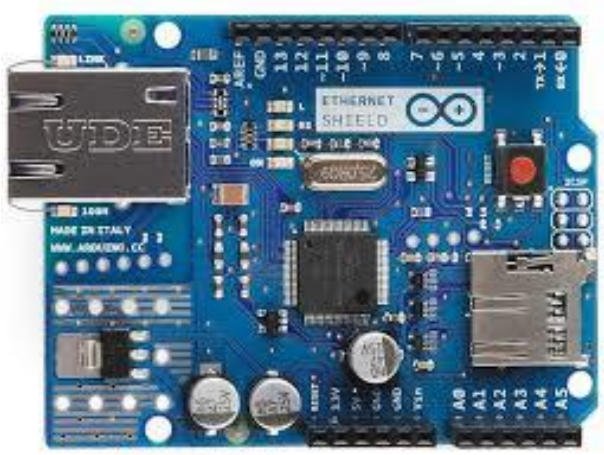
Όμως και σε αυτήν την περίπτωση ο Arduino έδωσε την λύση ώστε εύκολα να ολοκληρωθεί οποιοδήποτε σύστημα θέλουμε με τα επονομαζόμενα shields.

Τα Shield λοιπόν είναι ολοκληρωμένες πλακέτες που είναι σχεδιασμένες ώστε να κουμπώνουν πάνω στο Arduino προεκτείνοντας τη λειτουργικότητά του και τις δυνατότητές του. Είναι η hardware αντίστοιχη έννοια των plugin, addon και extension που υπάρχουν στο software. Το Arduino έχει πάρα πολλά Shields και ανάλογα με τη δουλειά που θέλουμε να κάνουμε μπορούμε να φτιάξουμε και το δικό μας. Αλλά ήδη αυτά που υπάρχουν στο εμπόριο έχουν το πλεονέκτημα ότι συνδέονται με ευκολία πάνω στον Arduino και έχουν χαμηλό κόστος παραγωγής.

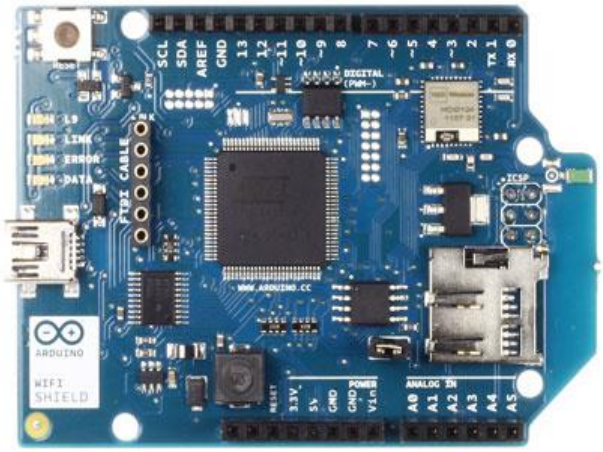
Τα Shield είναι σχεδιασμένα ώστε αφού κουμπωθούν πάνω στο Arduino να προωθούν τις υποδοχές του, ώστε να μπορείτε να συνδέσετε επιπλέον τα δικά σας εξαρτήματα ή να κουμπώσετε και επόμενο Shield. Φυσικά, το κάθε Shield χρησιμοποιεί 35 ορισμένους από τους πόρους συνδεσιμότητας του Arduino και έτσι δεν μπορείτε να συνδέσετε απεριόριστα Shield. Μάλιστα κάποια Shield μπορεί να μην είναι συμβατά μεταξύ τους γιατί χρησιμοποιούν τα ίδια pin του Arduino για επικοινωνία με αυτό. Επίσης, επειδή κάποια Shield δεν προωθούν τις συνδέσεις του Arduino (όπως π.χ. οι οθόνες οι οποίες δεν έχουν νόημα αν τις καλύψετε από πάνω με ένα επόμενο Shield), υπάρχουν ειδικά extender Shield που κουμπώνουν στο Arduino και δίνουν τηδυνατότητα σε δύο άλλα Shield να κουμπώσουν πάνω τους, λειτουργώντας σαν πολύπριζα. Όπως και για το ίδιο το Arduino, το βασικό

πλεονέκτημα των Shield δεν είναι τόσο το προφανές πλεονέκτημα του έτοιμου hardware όσο ότι συνοδεύονται συνήθως από έτοιμες βιβλιοθήκες που σας επιτρέπουν να προγραμματίζετε τα sketch σας σε high level.

Μερικά από τα πιο δημοφιλή Shield που κυκλοφορούν στο εμπόριο είναι:



2.2.5 Ethernet Shield: Δίνει στο Arduino τη δυνατότητα να δικτυωθεί σε ένα LAN ή στο internet μέσω ενός τυπικού καλωδίου Ethernet.



2.2.5 WiFi Shield: δίνει την δυνατότητα να συνδεθεί ο Arduino σ'ένα WiFi δίκτυο

2.2.5 GPS Shield: Προσθέτει GPS δυνατότητες στο Arduino (εντοπισμό στίγματος)

2.2.5 gsm shield: δίνει την δυνατότητα στον Arduino να συνδεθεί στο δίκτυο της κινητής τηλεφωνίας καθιστώντας τον στην ουσία ένα κινητό τηλέφωνο.

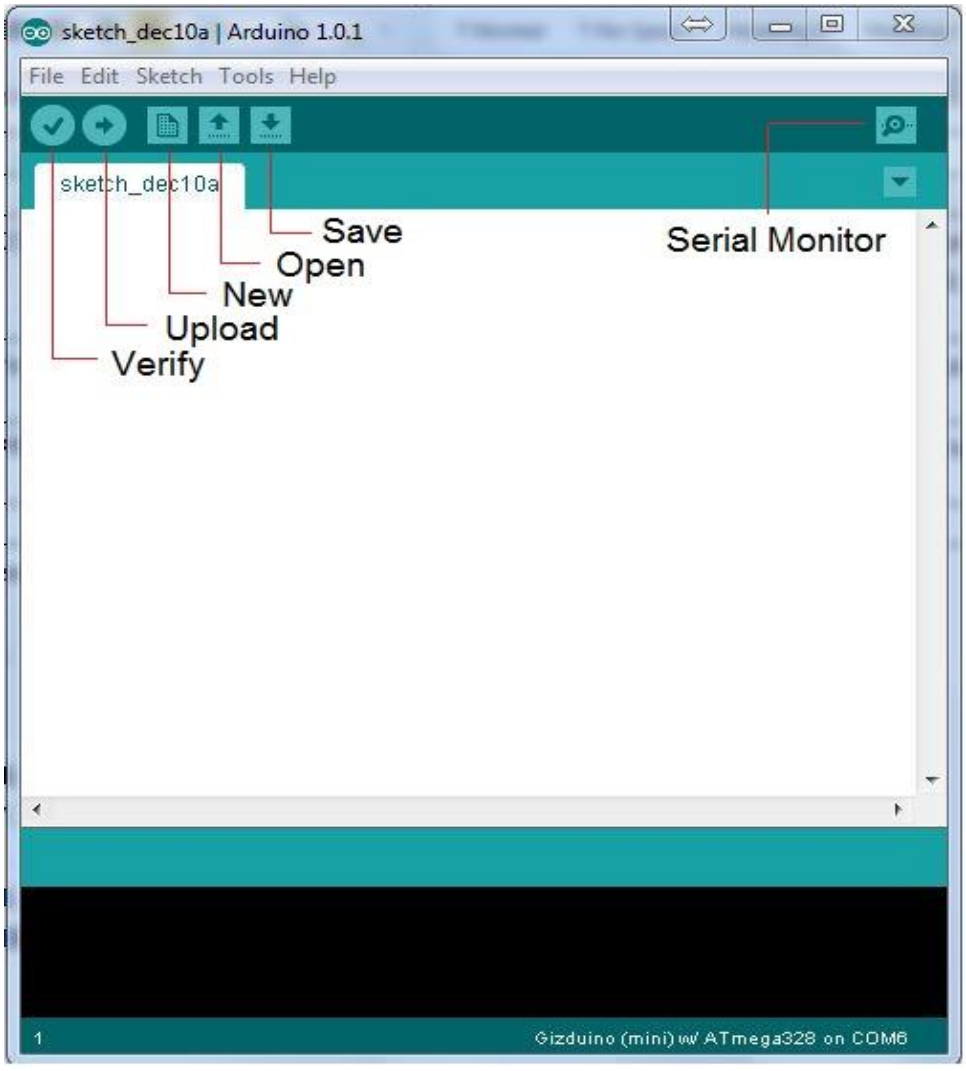
Επίσης υπάρχουν shield οθόνης, προσθέτοντας οθόνη στον Arduino. Motor shields, προσφέροντας την κατάλληλη οδήγηση για διάφορους τύπους moter. Και πάρα πολλά άλλα shields υποστηρίζοντας οποιαδήποτε επέκταση στο σύστημα.

2.3 <u>Περιβάλλον προγραμματισμού Arduino</u>

Το Arduino Ide αποτελεί το περιβάλλον ανάπτυξης του κώδικα για τον μικροελεγκτή της κάθε της κάθε τύπου πλατφόρμας (πλακέτα) του Arduino.

2.3.1 *Arduino Ide*

Το πλεονέκτημα του Arduino Ide είναι η απλότητά του στο να γράψει κανείς κώδικα, να τον ελέγξει και να τον εγκαταστήσει στον μικροελεγκτή. Το Arduino IDE περιέχει ένα πρόγραμμα επεξεργασίας κειμένου, για την σύνταξη του κώδικα, μια περιοχή στην οποία εμφανίζονται μηνύματα, μία κονσόλα κειμένου και κουμπιά στην γραμμή εργαλιών. Σε αυτή την γραμμή εργαλιών υπάρχει το compile δηλαδή της μεταγλώττισης του κώδικα, του «upload» δηλαδή της φόρτωσης του κώδικα και οι επιλογές «new», «open», «save» με τις οποίες ο χρήστης μπορεί να δημιουργήσει ανοίξει ή να σώσει ένα «σχέδιο». Μέσα στο μενού υπάρχουν οι επιλογές για την σειριακή θύρα («serial port») με την οποία επικοινωνεί το Arduino με τον Η/Υ και η καρτέλα επιλογής της υπολογιστικής πλατφόρμας Arduino που χρησιμοποιείται κάθε φορά («board»).



2.3.1 απεικόνιση του Arduino Ide

Το Arduino Ide συνδέεται με το hardware μέρος του Arduino για να φορτώσει προγράμματα και να επικοινωνεί μαζί τους.

Το Arduino Ide περιέχει:

- Περιβάλλον για τη συγγραφή των προγραμμάτων, με συντακτική χρωματική σήμανση
- Βιβλιοθήκες για προέκτασή της
- Compiler για την μεταγλώττιση του κώδικα
- Serial monitor που παρακολουθεί τις επικοινωνίες της σειριακής USB
- Το φόρτωμα του κώδικα στον Arduino

Επίσης είναι «freeware» λογισμικό πράγμα που βοηθά τόσο στη χρήση του αφού υποστηρίζεται σε όλα τα διαθέσιμα λειτουργικά Η/Υ Windows, MAC OS, Linux, όσο και στην ανάπτυξη του μέσω του διαδικτύου με σκοπό την απλούστευση του περιβάλλοντος και την φιλικότητα προς το χρήστη.

2.3.2 *Η γλώσσα της υπολογιστικής πλατφόρμας* Arduino

Η γλώσσα συγγραφής κώδικα Arduino βασίζεται στη γλώσσα Wiring μια παραλλαγή της C και της C++ και υποστηρίζει όλες τις βασικές δομές και χαρακτηριστικά αυτών των δύο γλωσσών. Άρα μπορούν να χρησιμοποιηθούν ίδιοι τύποι δεδομένων, ίδιοι τελεστές, βασικές εντολές και συναρτήσεις όπως στη γλώσσα C.

Τα προγράμματα του Arduino διαιρούνται σε τρία μέρη:

1. Δομή (structure)

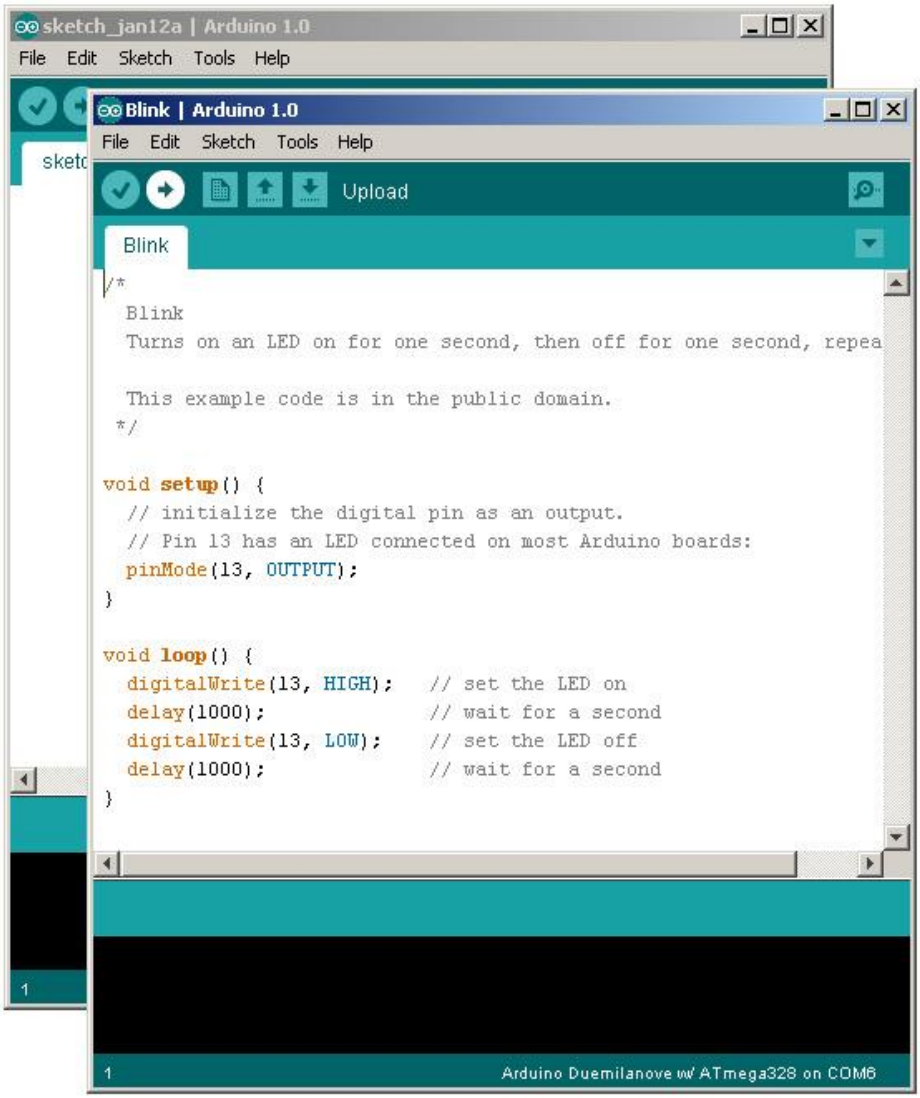
   Η δομή των προγραμμάτων στηρίζεται στις δύο βασικές συναρτήσεις setup() και loop().

   setup() : Μέσα στην συνάρτηση αυτή καθορίζουμε τις μεταβλητές που χρησιμοποιεί το πρόγραμμα, τις θύρες εισόδου εξόδου και τις βιβλιοθήκες που πρόκειται να χρησιμοποιήσουμε

   loop(): Η συνάρτηση αυτή αποτελεί τον κύριο ατέρμονα βρόγχο του κώδικα μας στον οποίο γράφουμε το κυρίως πρόγραμμα
2. Τιμές (values)
3. Συναρτήσεις (functions)

Ο τελικός κώδικας που έχει γραφτεί για τον Arduino ονομάζεται sketch.

Στην παρακάτω εικόνα εμφανίζεται η βασική δομή του κώδικα



2.3.3 Οι Βιβλιοθήκες της υπολογιστικής πλατφόρμας Arduino

Όλες οι βιβλιοθήκες της υπολογιστικής πλατφόρμας Arduino βρίσκονται στο διαδίκτυο και μπορεί να τις κατεβάσει ο καθένας ελεύθερα και σκοπός τους είναι η ευκολία στη χρήση διαφόρων παρελκόμενων μαζί τους. Το περιβάλλον Arduino (IDE) μπορεί να επεκταθεί με τη χρήση βιβλιοθηκών, ακριβώς όπως οι περισσότερες πλατφόρμες προγραμματισμού, αφού οι βιβλιοθήκες παρέχουν επιπλέον λειτουργίες για τη χρήση σε σχέδια όπου χωρίς αυτές η δημιουργία τους θα ήταν περίπλοκη. Μια σειρά από βιβλιοθήκες υπάρχουν προεγκατεστημένες, και είναι πολύ εύκολο να τις κατεβάσετε και να τις επεξεργαστείτε ή ακόμα και να δημιουργήσετε τη δική σας.

2.4 Πρωτόκολλο ethernet

Επειδή η πλατφόρμα Arduino πρέπει να επικοινωνήσει με το router και να στείλει τις πληροφορίες των αισθητηρίων στην ιστοσελίδα, πρέπει να συνδεθεί με το router μέσω ethernet.

2.4.1 *Περιγραφή πρωτοκόλλου ethernet*

Το βασικότερο πρωτόκολλο που χρησιμοποιείται ευρέως για μικρά δίκτυα είναι το Ethernet και αποτελεί την πλέον διαδεδομένη μέθοδο υλοποίησης τοπικών δικτύων (LAN) με τοπολογία αστέρα (Star) ή αρτηρίας (BUS), ενώ με βάση την αρχιτεκτονική που ακολουθούν τα δίκτυα χωρίζονται σε Ομότιμα (Peer-to-Peer) και στα δίκτυα πελάτη-διακομιστή (Server-based). Το πρωτόκολλο Ethernet περιλαμβάνει δύο βασικές υποκατηγορίες, οι οποίες ξεχωρίζουν κυρίως για το ρυθμό μεταφοράς δεδομένων. Η μία είναι η απλή Ethernet και χαρακτηρίζεται από την ταχύτητα των 10Mbps και η άλλη είναι η Fast Ethernet που έχει αντίστοιχη ταχύτητα τα 100Mbps. Υπάρχει και μία ακόμα υποκατηγορία η οποία υποστηρίζει ταχύτητες 1000Mbps(1Gbps) και ονομάζεται Gigabit Ethernet, αλλά δεν είναι τόσο διαδεδομένη ακόμα λόγο του υψηλού κόστους. Το Ethernet επιτρέπει τη μετάδοση πακέτων δεδομένων (Frames ή Packets) μεταβλητού μεγέθους από 72 έως και 1518Byte με την χρήση της τεχνολογίας CSMA/CD1. Κάθε πακέτο περιέχει μία κεφαλίδα (Header) στην οποία περιλαμβάνονται πληροφορίες όπως η διεύθυνση του μηχανήματος-αποστολέα, καθώς και αυτή του παραλήπτη.

1. **Απλή Ethernet:** Χαρακτηρίζεται από την ταχύτητα των 10Mbps και αποτελείται από τρεις υποκατηγορίες. Τις 10Base5, 10Base2 και 10BaseT. Αυτές έχουν κοινό χαρακτηριστικό το ρυθμό μεταφοράς δεδομένων.
2. **Fast Ethernet:** Χαρακτηρίζεται από την ταχύτητα των 100Mbps και είναι η κατηγορία 100BaseT, χρησιμοποιείται στην τοπολογία τύπου αστέρα (Star).

2.5 Το παγκόσμιο δίκτυο Web

Το router (δρομολογητής) επικοινωνεί με το παγκόσμιο δίκτυο για να μεταβιβάσει την ιστοσελίδα από τον Arduino στην τερματική συσκευή (smart phone).

Το παγκόσμιο δίκτυο **World Wide Web** (ή σκέτο **Web**) είναι ένα μεγάλο δίκτυο από υπολογιστές σε όλο τον κόσμο, οι οποίοι μπορούν να επικοινωνούν μεταξύ τους για να μοιραστούν ηλεκτρονικές πληροφορίες.

Οι πληροφορίες αυτές βρίσκονται σε έγγραφα που ονομάζονται **ιστοσελίδες**. Οι **ιστοσελίδες** είναι αρχεία που βρίσκονται σε υπολογιστές που ονομάζονται Web Servers (οι οποίοι είναι 24 ώρες το 24ωρό συνδεδεμένοι στο Internet ώστε να μπορούμε οποιαδήποτε στιγμή να συνδεθούμε και να ανακτήσουμε μια ιστοσελίδα που βρίσκεται σε αυτούς), ενώ οι υπολογιστές που συνδέονται στους **Web Servers** για να ανακτήσουν τις ιστοσελίδες λέγονται **Web Clients**.

Οι Web Clients χρησιμοποιούν ένα πρόγραμμα για να ανακτήσουν τα περιεχόμενα των ιστοσελίδων που βρίσκονται στους Web Servers. Αυτό το πρόγραμμα ονομάζεται Web browser (πρόγραμμα παρουσίασης ιστοσελίδων ή πρόγραμμα πλοήγησης). Δύο είναι οι κύριες δουλειές του Web browser:

1. να μπορεί να προσπελάσει τις ιστοσελίδες μετά από μια αίτηση του χρήστη και
2. να εμφανίζει τα περιεχόμενα των ιστοσελίδων.

Οι ιστοσελίδες περιέχονται από οδηγίες για τον τρόπο που θα εμφανίσουν τα περιεχόμενα τους στον Web browser. Οι Web browsers διαβάζουν τις οδηγίες αυτές και εμφανίζουν τις σελίδες στην οθόνη μας. Οι οδηγίες αυτές είναι γραμμένες στην γλώσσα HTML.

Οι υπολογιστές για να ανταλλάσσουν πληροφορίες χρησιμοποιούν το πρωτόκολλο επικοινωνίας HTTP.

2.6 Πρωτόκολλο http

Η ιστοσελίδα για να ανοιχθεί από κάποια τερματική συσκευή, αναρτάται από τον server στο web με το πρωτόκολο http

Το πρωτόκολλο HTTP (Hypertext Transfer Protocol) παρέχει ένα πρότυπο πρωτόκολλο δικτύου το οποίο χρησιμοποιούν τα προγράμματα περιήγησης και οι διακομιστές ιστού για την επικοινωνία. Είναι εύκολο να το αναγνωρίσετε όταν επισκέπτεστε έναν ιστότοπο επειδή είναι γραμμένο σωστά στη διεύθυνση URL (π.χ. *http: // www.*).

Αυτό το πρωτόκολλο είναι παρόμοιο με άλλα όπως το FTP , επειδή χρησιμοποιείται από ένα πρόγραμμα-πελάτη για να ζητήσει αρχεία από έναν απομακρυσμένο διακομιστή. Στην περίπτωση του HTTP, είναι συνήθως ένα πρόγραμμα περιήγησης ιστού που ζητά αρχεία HTML από έναν διακομιστή ιστού, τα οποία στη συνέχεια εμφανίζονται στο πρόγραμμα περιήγησης με κείμενο, εικόνες, υπερσυνδέσεις κ.λπ.

Το HTTP είναι αυτό που ονομάζεται "σύστημα χωρίς καθεστώς". Αυτό σημαίνει ότι σε αντίθεση με άλλα πρωτόκολλα μεταφοράς αρχείων, όπως το FTP , η σύνδεση HTTP

πέφτει μόλις γίνει το αίτημα. Έτσι, μόλις το πρόγραμμα περιήγησης ιστού σας στείλει το αίτημα και ο διακομιστής ανταποκριθεί με τη σελίδα, η σύνδεση είναι κλειστή.

Δεδομένου ότι η πλειοψηφία των προεπιλεγμένων προγραμμάτων περιήγησης ιστού σε HTTP, μπορείτε να πληκτρολογήσετε μόνο το όνομα τομέα και το πρόγραμμα περιήγησης να συμπληρώνει αυτόματα το τμήμα "http: //".

Πώς λειτουργεί το HTTP

HTTP είναι ένα πρωτόκολλο εφαρμογής στρώματος που είναι χτισμένο πάνω από το TCP που χρησιμοποιεί ένα μοντέλο επικοινωνίας πελάτη-διακομιστή . Οι υπολογιστές-πελάτες και οι διακομιστές HTTP επικοινωνούν μέσω μηνυμάτων αίτησης και απόκρισης HTTP. Οι τρεις κύριοι τύποι μηνυμάτων HTTP είναι GET, POST και HEAD.

- **Τα** μηνύματα **HTTP GET που** αποστέλλονται σε ένα διακομιστή περιέχουν μόνο μια διεύθυνση URL . Στο τέλος της διεύθυνσης URL μπορούν να επισυναφθούν μηδενικές ή περισσότερες προαιρετικές παράμετροι δεδομένων. Ο διακομιστής επεξεργάζεται το προαιρετικό τμήμα δεδομένων της διεύθυνσης URL, εάν υπάρχει, και επιστρέφει το αποτέλεσμα (μια ιστοσελίδα ή ένα στοιχείο μιας ιστοσελίδας) στο πρόγραμμα περιήγησης.
- **Τα** μηνύματα **HTTP POST** τοποθετούν τις προαιρετικές παραμέτρους δεδομένων στο σώμα του μηνύματος αίτησης αντί να τις προσθέτουν στο τέλος της διεύθυνσης URL.
- **Το** αίτημα **HTTP HEAD** λειτουργεί το ίδιο με τα αιτήματα GET. Αντί να απαντά με το πλήρες περιεχόμενο της διεύθυνσης URL, ο διακομιστής στέλνει πίσω μόνο τις πληροφορίες κεφαλίδας (που περιέχονται στο τμήμα HTML).

Το πρόγραμμα περιήγησης εκκινεί την επικοινωνία με ένα διακομιστή HTTP εκκινώντας μια σύνδεση TCP στο διακομιστή. Οι περιόδους περιήγησης στο Web χρησιμοποιούν την θύρα διακομιστή 80 από προεπιλογή, αν και χρησιμοποιούνται μερικές φορές άλλες θύρες όπως το 8080.

Μόλις δημιουργηθεί μια περίοδος σύνδεσης, ο χρήστης ενεργοποιεί την αποστολή και λήψη μηνυμάτων HTTP με την επίσκεψη στην ιστοσελίδα.

2.7 Γλώσσα προγραμματισμού html

Για να δημιουργήσει ο Arduino μία ιστοσελίδα και να την ανεβάσει στον server, αυτή πρέπει να γραφτεί σε γλώσσα html μέσα στον κώδικα του Arduino. Όπως αναφέρθηκε παραπάνω στην ουσία ο Arduino θα δώσει τις οδηγίες για το πως θα εμφανιστεί η σελίδα στον web και θα τις δώσει με την γλώσσα προγραμματισμού html.

HTML είναι το ακρώνυμο από το Hyper Text Markup Language που σημαίνει γλώσσα χαρακτηρισμού υπερκειμένου. Η χρήση μιας γλώσσας χαρακτηρισμού σημαίνει ότι γράφεται πρώτα το κείμενο και έπειτα προσθέτονται ειδικά σύμβολα γύρω από τις λέξεις ή από ολόκληρες προτάσεις ώστε να καθοριστεί η εμφάνιση τους στην οθόνη. Τα ειδικά σύμβολα στην HTML λέγονται ετικέτες (tags). Η HTML διαθέτει ένα πεπερασμένο αριθμό ετικετών που μπορούμε να χρησιμοποιήσουμε. Ωστόσο ο αριθμός αυτός δεν παραμένει σταθερός. Κατά διαστήματα το W3 Consortium, το οποίο ανάπτυξε και διαχειρίζεται τα πρότυπα της HTML, δημοσιεύει νέα πρότυπα στα οποία προσθέτει καινούργιες ετικέτες που καλύπτουν ή διορθώνουν μια λειτουργία στο προηγούμενο πρότυπο. Η τελευταία αναθεώρηση του HTML προτύπου είναι η HTML5

Οι ετικέτες ελέγχουν την δομή και την μορφή του κειμένου της ιστοσελίδας. Επίσης παρέχουν πληροφορίες προς τον web browser για την σελίδα που πρόκειται να εμφανίσουν, όπως ο τίτλος της σελίδας ή ο συγγραφέας της.

Το αρχείο που περιέχει HTML ετικέτες λέγεται HTML αρχείο και έχει επέκταση .html ή .htm (εκτός βέβαια κι' αν η ιστοσελίδα είναι δυναμική οπότε έχει επεκτάσεις όπως .php. asp, jsp κτλ.). Τα αρχεία αυτά είναι απλά αρχεία κειμένου σε μορφή ASCII και δεν περιέχουν πληροφορίες για το περιβάλλον ή τα προγράμματα με τα οποία θα λειτουργήσουν. Μπορείτε να ανοίξετε και να δείτε τα αρχεία htm ή html με οποιονδήποτε επεξεργαστή κειμένου, π.χ. Σημειωματάριο (Notepad) των Windows

παράδειγμα γραφής της γλώσσας html:

```
<html>
<head>
<title>My first web site</title>
</head>

<body>
This is <b>Great</b>!!! <b>YEAH!!!</b><br />
I can build my own <i>web site</i>. <b>YEAH!!!</b><br />
<i>Hey Ma look!!!</i> I can do it by <b>myself</b>
</body>
</html>
```

Κ Ε Φ Α Λ Α Ι Ο 3ο

ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ ΑΠΟΜΑΚΡΥΣΜΕΝΗΣ ΔΙΑΧΕΙΡΙΣΗΣ

3.1 Μέρη που αποτελείτε το σύστημα απομακρυσμένης διαχείρισης σπιτιού

Είσοδοι: το σύστημα αποτελείται από πέντε αισθητήρες συνολικά που θα διαβάζουν τα φυσικά μεγέθη που μας ενδιαφέρουν δηλαδή θερμοκρασία, ένταση φωτός και υγρασία. Οι τιμές αυτών των μεγεθών θα εμφανίζονται κάθε στιγμή στην ιστοσελίδα που θα ανοίγεται από το κινητό τηλέφωνο. Οι αισθητήρες είναι:

- δύο αισθητήρες θερμοκρασίας-υγρασίας (DHT11) που θα διαβάζουν τις θερμοκρασίες και υγρασία εντός σπιτιού ο πρώτος και εκτός σπιτιού ο δεύτερος, για να εποπτεύουμε τις θερμοκρασίες και να αποφασίζουμε αν θα έχουμε θέρμανση ή ψύξη.
- Δύο φωτοαντιστάσεις που θα τοποθετηθούν εντός και εκτός σπιτιού για να βλέπουμε την ένταση του φωτός σε κάθε μία από αυτές.
- Έναν αισθητήρα βροχής που θα είναι τοποθετημένος εξωτερικά και θα μας δίνει πληροφορία για την υγρασία, δηλαδή αν είναι βροχερός ο καιρός ή όχι.

Κεντρική μονάδα: αποτελείται από τον κεντρικό μικροεπεξεργαστή Arduino που βρίσκεται τοποθετημένος στην αναπτυξιακή πλατφόρμα του, ο οποίος θα δέχεται τις τιμές των αισθητηρίων, θα τις ανεβάζει στο web (δλδ στην ιστοσελίδα) και θα δέχεται τις εντολές από την ιστοσελίδα για να ενεργοποιήσει ή όχι τα ρελέ των συσκευών.

Έξοδοι: αποτελείτε από πέντε ρελέ και το κάθε ένα συνδέεται σε κάθε συσκευή που θέλουμε να ελέγξουμε, δηλαδή ένα ρελέ στο καλοριφέρ, ένα στο κλιματιστικό, ένα στα εσωτερικά φώτα, ένα στα εξωτερικά φώτα και το πέμπτο ρελέ στην τέντα του μπαλκονιού.

Η κεντρική μονάδα θα οπλίζει ή όχι τα ρελέ σύμφωνα με τις εντολές που θα δίνουμε από την ιστοσελίδα. Δηλαδή στην ιστοσελίδα θα βλέπουμε τις τιμές των αισθητηρίων και θα αποφασίζουμε ποια συσκευή θέλουμε να θέσουμε σε λειτουργία και ποια συσκευή να απενεργοποιήσουμε, από τα εικονικά κουμπιά on/off που θα εμφανίζονται στην ιστοσελίδα.

3.2 Συνολική περιγραφή του συστήματος απομακρυσμένης διαχείρισης

Η εργασία έχει σκοπό να επιτύχει απομακρυσμένο έλεγχο συσκευών ενός σπιτιού με την χρήση μικροελεγκτή Arduino.

Μέσω ιστοσελίδας που αναπτύσσεται από τον μικροελεγκτή θα ελεγχουμε συσκευές σπιτιού που είναι το καλοριφερ, το κλιματιστικό , η τεντα μπαλκονιού, τα εσωτερικά φώτα και τα εξωτερικά φώτα του σπιτιού.

Η ιστοσελίδα είναι γραμμένη μέσα στον κώδικα του Arduino και κάθε φορά που τίθεται σε λειτουργία ο Arduino θα διαβάζει την ιστοσελίδα και θα την ανεβάζει προσωρινά στον server. Δηλαδή είναι δυναμική ιστοσελίδα και όχι στατική. Η ιστοσελίδα θα εμφανίζει τρεις στήλες. Η πρώτη θα δείχνει τα φυσικά μεγέθη από τους αισθητήρες σε real time, δηλαδή στιγμιαίες τιμές. Άρα θα έχουμε πέντε τιμές, θερμοκρασία εντός σπιτιού, θερμοκρασία εκτός, ένταση φωτός εντός σπιτιού, ένταση φωτός εκτός και παρουσία υγρασίας εκτός σπιτιού. Στην δεύτερη στήλη δίπλα από τις τιμές των αισθητηρίων αντιστοιχίζονται τα ονόματα των συσκευών που θα ελέγχουμε, δηλαδή καλοριφέρ, κλιματιστικό, φώτα, παντζούρια και τέντα μπαλκονιού. Και στην τρίτη στήλη δίπλα από κάθε όνομα συσκευής θα βρίσκεται διακόπτης on/off για να ανοιγοκλείνουμε την συσκευή.

Στην σελίδα λοιπόν, θα εμφανίζονται σε real time οι τιμές πέντε αισθητηρίων που συνδέονται στον μικροελεγκτή. Τα αισθητήρια είναι δύο φωτοαντιστάσεις εντός και εκτός σπιτιού έκαστως, δύο ολοκληρωμένα μέτρησης θερμοκρασίας (DHT11) και αισθητήρα βροχής.

Ανάλογα με τις τιμές αυτών των αισθητηρίων, θα αποφασίζουμε αν θα ανεργοποιήσουμε τις συσκευές που συνδέονται μέσω ρελε με τον μικροελεγκτή. Ενεργοποίηση ή απενεργοποίηση θα γίνεται από την ίδια ιστοσελίδα που ανεβάζει στο server ο ελεγκτης όπου φαίνονται και οι τιμές των παραπάνω αισθητηρίων.

Συμπέρασμα, ανάλογα με τις τιμές των φυσικών μεγεθών (θερμοκρασία, ένταση φωτός, ανίχνευση βροχής) θα αποφασίζουμε αν θα ενεργοποιούμε κάποια συσκευή του σπιτιού και οι οποίες συνδέονται με τον Arduino.

Συνολικά, χρησιμοποιώ στον Arduino πέντε εισόδους για τα αισθητήρια και πέντε εξόδους που συνδέονται τα ρελε για τον έλεγχο των συσκευών (ενεργοποίηση/απενεργοποίηση).

Μέσω της ιστοσελίδας έχω αμφίδρομη επικοινωνία με τον μικροελεγκτή, δλδ στέλνει τιμές των αισθητηρίων και του δίνω εντολές ενεργοποίησης ή μη κάθε συσκευής.

Παρακάτω γίνεται αναφορά σε κάθε ένα κομμάτι/μέρος του συστήματος.

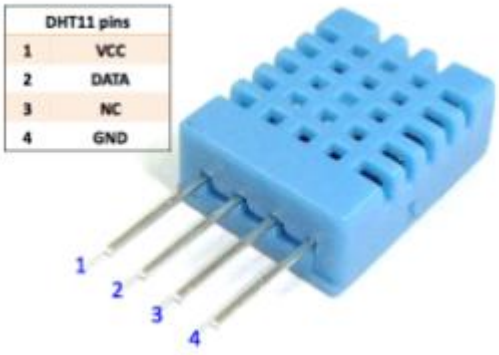
3.3 Αισθητήριο θερμοκρασίας

Τα αισθητήρια θερμοκρασίας που χρησιμοποιούμε είναι DHT11. Το DHT-11 είναι ένας βασικός, χαμηλού κόστους, αισθητήρας για την εύρεση υγρασίας και θερμοκρασίας στον χώρο. Στο εσωτερικό του υπάρχει ένας αισθητήρας υγρασίας και ένα θερμίστορ (μεταβλητή αντίσταση που η τιμή της αλλάζει σε σχέση με την θερμοκρασία) 'διαβάζοντας' έτσι τον αέρα που το περιβάλει.

Όπως φαίνεται στην εικόνα παρακάτω Στη σύνδεσή του από αριστερά προς τα δεξιά, το πρώτο pin είναι η τροφοδοσία, το δεύτερο είναι τα δεδομένα που θα στείλει στην αναλογική είσοδο του Arduino και το τέταρτο είναι η κοινή γείωση.

Τεχνικά χαρακτηριστικά:

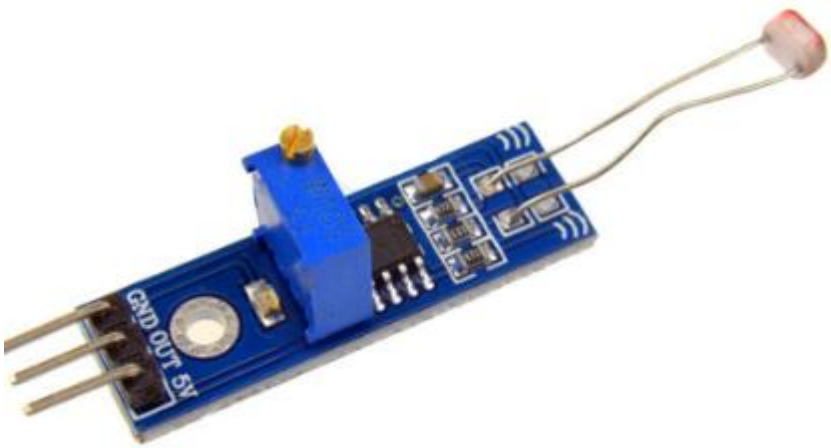
- *Πηγή : 3-5V*
- *Μέγιστο ρεύμα: 2.5mA*
- *Υγρασία: 20-80%, ακρίβεια 2-5%*
- *Θερμοκρασία: 0 to 50°C, ακρίβεια ±0.5°C*



3.3 αισθητήρας DHT11

3.4 Αισθητήριο φωτός

Για να διακρίνουμε πόσο φωτεινό είναι το σπίτι, μετράμε την ένταση του φωτός σε μονάδα μέτρησης lumen. Το αισθητήριο που χρησιμοποιούμε είναι η φωτοαντίσταση που είναι τοποθετημένη σε μία αναπτυξιακή πλακέτα που ονομάζεται ldr-module-sensor.



3.4 ldr-module-sensor

Σε αυτήν την πλακέτα διακρίνουμε την φωτοαντίσταση ldr, το ολοκληρωμένο lm-393, το μπλε ποτενσιόμετρο, το power led και το output led.

Η αντίσταση LDR ή Light Dependent Resistor είναι ένας τύπος μεταβλητής αντίστασης. Είναι επίσης γνωστό ως φωτοαντίσταση. Η εξαρτώμενη από το φως αντίσταση (LDR) λειτουργεί με βάση την αρχή της «Φωτοαγωγιμότητας» . Η αντίσταση LDR αλλάζει ανάλογα με την ένταση φωτός που πέφτει στο LDR. Όταν η ένταση του φωτός αυξάνεται στην επιφάνεια LDR, τότε η αντίσταση LDR θα μειωθεί και η αγωγιμότητα του στοιχείου θα αυξηθεί. Όταν η ένταση του φωτός μειώνεται στην επιφάνεια LDR, τότε η αντίσταση LDR θα αυξηθεί και η αγωγιμότητα του στοιχείου θα μειωθεί.

Η μονάδα αισθητήρα LDR διαθέτει μια ενσωματωμένη μεταβλητή αντίσταση ή ποτενσιόμετρο, αυτή η μεταβλητή αντίσταση είναι μια προεπιλογή 10k . Χρησιμοποιείται για τη ρύθμιση της ευαισθησίας αυτού του αισθητήρα LDR. Περιστρέφουμε το προρυθμισμένο κουμπί για να ρυθμίσουμε την ευαισθησία της ανίχνευσης έντασης φωτός. Εάν περιστρέψουμε το προκαθορισμένο κουμπί προς τη φορά των δεικτών του ρολογιού, η ευαισθησία της ανίχνευσης της έντασης του φωτός θα αυξηθεί . Εάν το περιστρέψουμε αριστερόστροφα, η ευαισθησία της ανίχνευσης της έντασης του φωτός θα μειωθεί .

Η ενσωματωμένη λυχνία power LED υποδεικνύει ότι η τροφοδοσία της μονάδας αισθητήρα LDR είναι ενεργοποιημένη ή απενεργοποιημένη.

Για την λυχνία output LED, όταν ο αισθητήρας LDR ανιχνεύσει το φως, το πράσινο LED ανάβει. Όταν ο αισθητήρας LDR ανιχνεύσει το σκοτάδι, το πράσινο LED σβήνει.

Όπως φαίνεται από την παραπάνω εικόνα συνδέεται με την τροφοδοσία στο pin 5volt, το pin out συνδέεται στην αναλογική είσοδο του Arduino και το τρίτο pin είναι η κοινή γείωση.

Τεχνικά χαρακτηριστικά:

- *Πηγή : 3-5V*
- *Μέγιστο ρεύμα: 2.5mA*
- *ακρίβεια 0.5%*
- *Digital output '0' and '1'*

3.5 Αισθητήριο βροχής

Για να ανιχνεύσουμε την παρουσία βροχής, χρησιμοποιούμε την πλακέτα με τον αισθητήρα βροχής (Water Level Sensor Module)

3.5water-sensor-module

Ο αισθητήρας στάθμης νερού έχει τρεις ακίδες:

- S (signal) pin: θα συνδεθεί στην ψηφιακή είσοδο του arduinio
- Vcc pin(+): τροφοδοτεί τον αισθητήρα με τάση 3.3 - 5volt
- GND pin(-): σύνδεση γείωσης με την γείωση του Arduino

Όσο περισσότερο βρέχεται ο αισθητήρας τόσο μεγαλύτερη θα είναι η τάση στο pin Signal.

Πιο συγκεκριμένα ο αισθητήρας έχει μία σειρά από δέκα εκτεθειμένες γραμμές χαλκού. Αυτές οι γραμμές δεν συνδέονται εκτός εάν γεφυρωθούν όταν βραχούν με νερό. Οι γραμμές λειτουργούν σαν μεταβλητή αντίσταση που η τιμή της αλλάζει ανάλογα με το ποσό του νερού που θα βρίσκεται σε αυτές.

Εάν υπάρχει πολύ νερό, περισσότερες γραμμές βραχυκυκλώνονται και αυξάνει η αγωγιμότητα άρα μειώνεται η αντίσταση του αισθητηρίου με αποτέλεσμα να αυξάνεται και η τάση εξόδου, που ισοδυναμεί με την παρουσία βροχής.

Specifications:

1. Product Name: water level sensor

2. Operating voltage: DC3-5V

3. Operating current: less than 20mA

4. Sensor Type: Analog

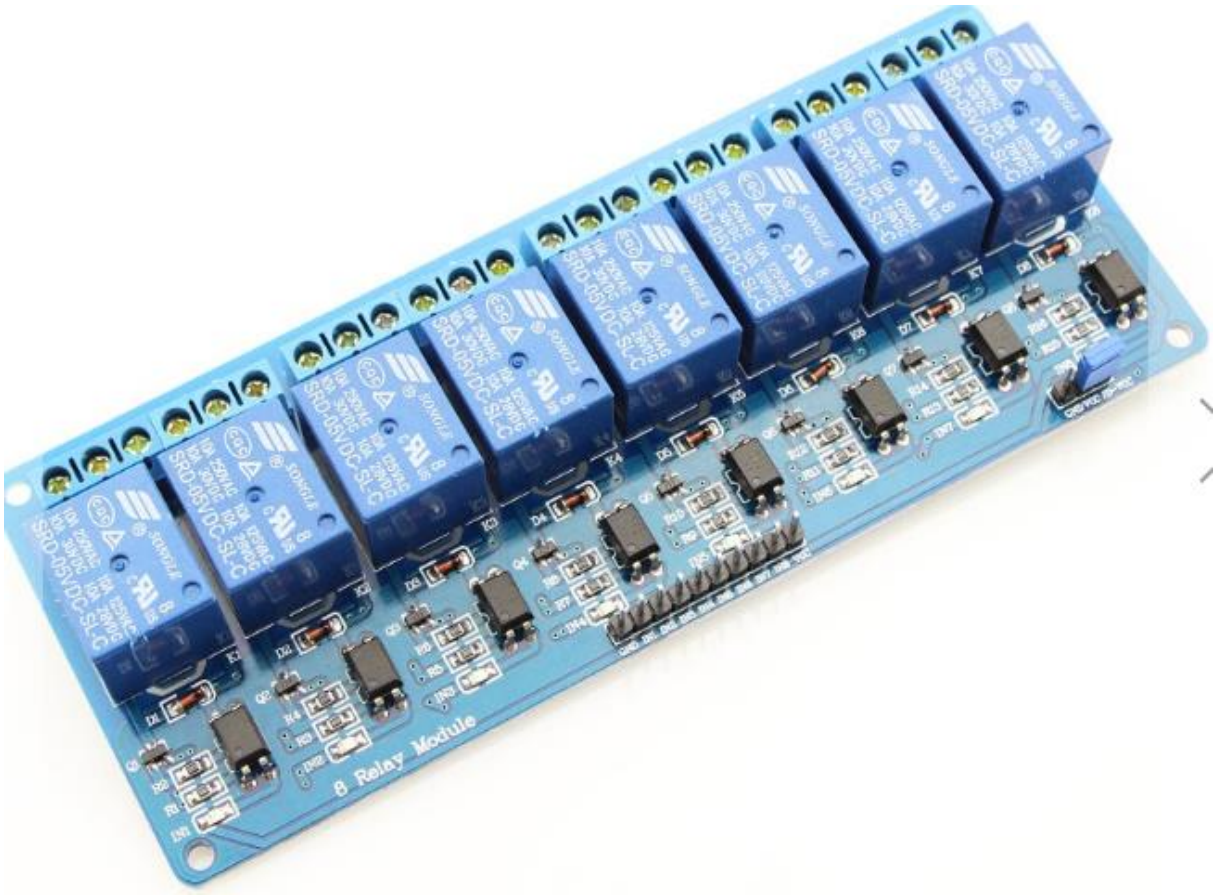
5. Detection Area: 40mmx16mm

6. Production process: FR4 double-sided HASL

3.6 Ρελέ στις εξόδους της κεντρικής μονάδας

Τα ρελέ θα ενεργοποιούν ή θα απενεργοποιούν τις συσκευές που θέλουμε. Το ρελέ είναι ένας ψηφιακός διακόπτης για τον έλεγχο πολύ υψηλότερων τάσεων και ρευμάτων από τις κανονικές πλακέτες σας Arduino. Όταν δέχεται μια λογική τάση, το ρελέ θα αλλάξει για να επιτρέψει τη ροή ρεύματος ή τη διακοπή, ανάλογα με την καλωδίωση. Ένα ρελέ αποτελείται από ένα πηνίο, έναν κοινό ακροδέκτη, έναν κανονικά κλειστό ακροδέκτη και έναν κανονικά ανοικτό ακροδέκτη. Όταν το πηνίο ενεργοποιείται, ο κοινός ακροδέκτης και ο κανονικά ανοιχτός ακροδέκτης θα έχουν συνέχεια, δηλαδή θα βραχυκυκλωθούν.

Στην εργασία χρησιμοποιούμε μία πλακέτα που περιέχει οκτώ ρελέ από τα οποία χρειαζόμαστε πέντε. Από την παρακάτω εικόνα φαίνονται δέκα ακίδες που συνδέονται στον Arduino, από τις οποίες οι οκτώ αντιστοιχούν στα ρελέ και οι υπόλοιπες δύο στην τροφοδοσία της πλακέτας. Κάθε ρελέ έχει από κάτω και μια λυχνία LED που ανάβει όταν είναι οπλισμένο, δηλαδή θέτει την αντίστοιχη συσκευή του σπιτιού σε λειτουργία.



3.6 8-Channel Relay Module

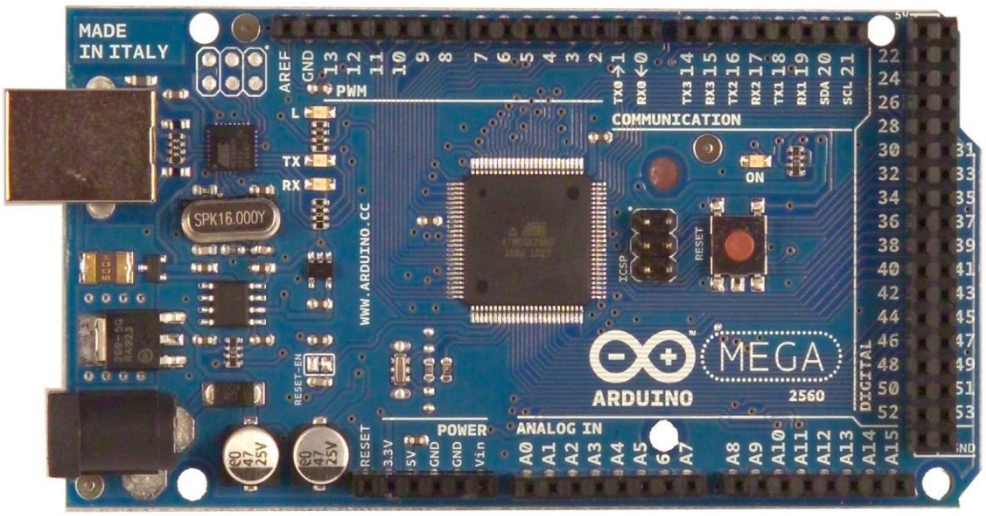
Τεχνικά χαρακτηριστικά:

- Control Voltage: 5V DC
- Max Control Capacity:10A@250VAC or 10A@30VDC

3.7 Κεντρική μονάδα επεξεργασίας

Την κεντρική μονάδα επεξεργασίας του συστήματος αποτελεί η υπολογιστική πλατφόρμα του Arduino με τον μικροελεγκτή ATmega 2560, επονομαζόμενη ως Arduino mega2560.

Ο Arduino mega 2560 είναι ένας μικροελεγκτής της οικογένειας Arduino που βασίζεται στον μικροελεγκτή ATmega2560 της Atmel, διαθέτει 54 ψηφιακές εισόδους/εξόδους (εκ των οποίων 14 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM), 16 αναλογικές εισόδους, 4 UARTs (σειριακές θύρες hardware), ένα κρύσταλλο ταλάντωσης στα 16 MHz, μια σύνδεση USB, μία είσοδο ρεύματος, μία ICSP, και ένα reset button.



3.3 arduino mega2560

Χαρακτηριστικά:

Μικροελεγκτής	ATmega2560
Τάση λειτουργίας	5V
Τάση εισόδου (συνίσταται)	7-12V
Τάση εισόδου (όρια)	6-20V
Ψηφιακές I/O καρφίτσες	54(εκ των οποίων οι 14 προβλέπουν PWM έξοδοι)
Αναλογικές Pins εισόδους	16
DC ρεύμα ανά I/O pin	40mA
DC ρεύμα για 3.3V pin	50mA
Flash memory	256KB εκ των οποίων 8KB που χρησιμοποιούνται από τον bootloader
SRAM	8KB
EEPROM	4KB
Ταχύτητα ρολογιού	16MHz

Τροφοδοσία

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

Εξωτερική (όχι USB) ισχύς μπορεί να προέρχεται από έναν AC-to-DC μετασχηματιστή ή μπαταρία. Ο μετασχηματιστής μπορεί να συνδεθεί με τη σύνδεση ενός βύσματος 2,1 χιλιοστά με κέντρο-θετικό στην υποδοχή ρεύματος.

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

Η Mega2560 διαφέρει από όλους τους προηγούμενους του, γιατί δεν χρησιμοποιεί FTDI USB to serial ως chip οδηγού, αλλά χρησιμοποιεί τον Atmega8U2 της Atmel που έχει προγραμματιστεί ως USB to serial μετατροπέας, με αποτέλεσμα πολύ μεγαλύτερες ταχύτητες στην σειριακή σύνδεση.

Οι ακροδέκτες σύνδεσης έχουν ως εξής:

VIN Η τάση εισόδου που τροφοδοτείτε ο Arduino, όταν χρησιμοποιούμε μια εξωτερική πηγή ισχύος

5V η ρυθμιζόμενη παροχή ηλεκτρικού ρεύματος που χρησιμοποιείται για την τροφοδοσία του μικροελεγκτή και άλλα στοιχεία του Arduino. Αυτ'η μπορεί να προέλθει είτε από Vin μέσω του ρυθμιστή τάσης πάνω στον Arduino, είτε μέσω USB

3V3 η 3.3V προμήθεια που παράγεται από το ενσωματωμένο ρυθμιστή. Μέγιστο ρεύμα είναι 50mA.

GND ακίδες γείωσης

Μνήμη

Έχει 256KB μνήμη Flash για την αποθήκευση κώδικα (εκ των οποίων 8KB από τον φορτωτή εκκίνησης), 8KB SRAM και 4KB EEPROM (που μπορεί να διαβάσει και να γράψει με την βιβλιοθήκη EEPROM).

Ακροδέκτες του μικροελεγκτή Arduino

Ο Arduino έχει 54 ψηφιακούς ακροδέκτες Εισόδου/Εξόδου. Αυτοί μπορούν να τεθούν ως είσοδοι ή ως έξοδοι με τις εντολές-συναρτήσεις pinMode(), digitalWrite(), and digitalRead(). Λειτουργούν στα 5 Volts και έχουν την δυνατότητα να παρέχουν ή να καταβυθίζουν ένταση της τάξεως των 40mA. Σε κάθε Pin υπάρχει εσωτερικά ένας Pull-up αντιστάτης στα 20-50KΩ. Επιπλέον έχει 16 Αναλογικούς ακροδέκτες Εισόδου. Αυτοί

μπορούν να διαβάσουν αναλογικές τιμές όπως η τάση μιας μπαταρίας κτλ και να τις μετατρέψουν σε έναν αριθμό από 0-1023. Η μέτρηση της τάσης γίνεται από προκαθορισμένα από 0 έως 5 volts.

Εκτός αυτού 15 εκ των 54 ψηφιακών ακροδεκτών έχουν την δυνατότητα να προγραμματιστούν ώστε να λειτουργούν ως Αναλογικές Έξοδοι. Κάποιοι ακροδέκτες έχουν συγκεκριμένες λειτουργίες όπως:

Σειριακή Λειτουργία: 0 (RX) and 1 (TX). Χρησιμοποιούνται για λήψη (RX) και εκπομπή (TX) TTL σειριακών δεδομένων. Αυτοί οι ακροδέκτες είναι συνδεδεμένοι με τους αντίστοιχους του ολοκληρωμένου FTDI USB-to-TTL Serial.

Εξωτερικές Διακοπές: 2 και 3. Αυτοί οι ακροδέκτες μπορούν να ενεργοποιούν διακοπές αν ανιχνευθεί παλμός χαμηλής τάσης. Με την συνάρτηση attachInterrupt().

PWM: 3, 5, 6, 9, 10, and 11. Παρέχουν Έξοδο 8-bit PWM με την συνάρτηση analogWrite().

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Αυτοί οι ακροδέκτες επιτρέπουν επικοινωνία SPI, η οποία αν και παρέχεται από το hardware δεν είναι ακόμα διαθέσιμη στην γλώσσα προγραμματισμού του Arduino.

LED: 13. Στον ακροδέκτη 13 υπάρχει ένα ενσωματωμένο LED. Όταν ο ακροδέκτης έχει τιμή HIGH, το LED φωτοβολεί Επιπλέον υπάρχουν κάποιοι ακροδέκτες.

Επιπλέον υπάρχουν κάποιοι ακροδέκτες για ειδικές λειτουργίες όπως:

I2C: 4 (SDA) and 5 (SCL). Υποστηρίζει το πρωτόκολλο I2C (TWI) χρησιμοποιώντας βιβλιοθήκες τις Γλώσσας προγραμματισμού Wiring

AREF. Reference voltage for the analog inputs. Χρησιμοποιείται με την συνάρτηση analogReference()

Reset. Αν τεθεί σε κατάσταση LOW τότε επανεκκινεί τον Μικροελεγκτή. Σε αυτή τη γραμμή τοποθετείται ένας διακόπτης.

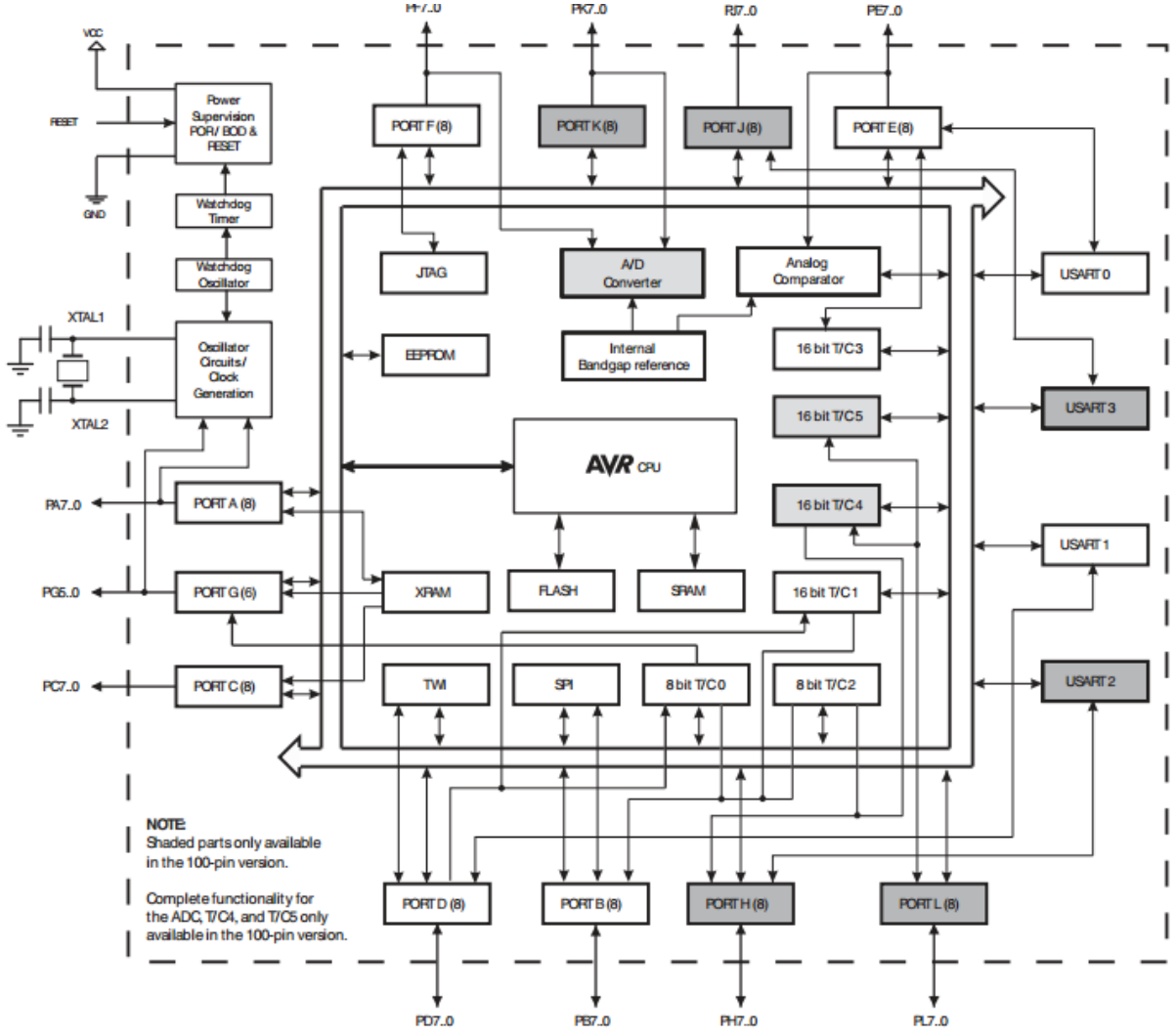
I2C: 4 (SDA) and 5 (SCL). Υποστηρίζει το πρωτόκολλο I2C (TWI) χρησιμοποιώντας βιβλιοθήκες τις Γλώσσας προγραμματισμού Wiring.

AREF. Reference voltage for the analog inputs. Χρησιμοποιείται με την συνάρτηση analogReference().

Reset. Αν τεθεί σε κατάσταση LOW τότε επανεκκινεί τον Μικροελεγκτή. Σε αυτή τη γραμμή τοποθετείται ένας διακόπτης.

Επικοινωνία

Ο Arduino mega2560 έχει την δυνατότητα να επικοινωνεί με τον Ηλεκτρονικό Υπολογιστή, έναν άλλον Arduino ή άλλους μικροελεγκτές. Το ολοκληρωμένο ATMega 2560 παρέχει σειριακή επικοινωνία TTL 5Volt UARTs, η οποία είναι διαθέσιμη από τους ακροδέκτες (λήψη RX) 0 και (εκπομπή TX) 1 του ολοκληρωμένου. Επιπλέον στην αναπτυξιακή πλακέτα του Arduino είναι ενσωματωμένο ένα ολοκληρωμένο το FTDI FT232RL το οποίο παρέχει σειριακή επικοινωνία με τον Ηλεκτρονικό Υπολογιστή για προγραμματισμό, πάνω από την θύρα USB με την βοήθεια των ανάλογων FTDI drivers. Οι drivers αυτοί περιλαμβάνονται στο software για τον Arduino και παρέχουν μια ιδεατή θύρα επικοινωνίας στον Ηλεκτρονικό Υπολογιστή για τους σκοπούς της επικοινωνίας



3.3 block-διάγραμμα arduinomega 2560

3.8 Arduino Ethernet Shield

Το Arduino Ethernet Shield V1 επιτρέπει σε μια πλακέτα Arduino να συνδεθεί στο διαδίκτυο. Βασίζεται στο τσιπ Wiznet W5100 ethernet. Το Wiznet W5100 παρέχει μια στοίβα δικτύου (IP) ικανή τόσο για TCP όσο και για UDP. Υποστηρίζει έως και τέσσερις ταυτόχρονες συνδέσεις πρίζας. Η πλακέτα (shield) ethernet συνδέεται με μια πλακέτα Arduino χρησιμοποιώντας τις ακίδες καλωδίων που υπάρχουν πάνω σε αυτήν. Το Ethernet Shield V1 έχει μια τυπική σύνδεση RJ-45, με ενσωματωμένο μετασχηματιστή γραμμής και ενεργοποιημένο Power over Ethernet.

Με αυτήν την πλακέτα που συνδέεται στην πλατφόρμα του Arduino μπορούμε να έχουμε μία αμφίδρομη επικοινωνία μεταξύ του smart phone και του Arduino για να διαχειριζόμαστε το σπίτι μας μέσω διαδικτύου.

Χαρακτηριστικά Ethernet W5100 R3 Shield Module

-W5100 TCP/IP by wire-wrap header with for arduino products

-4 socket

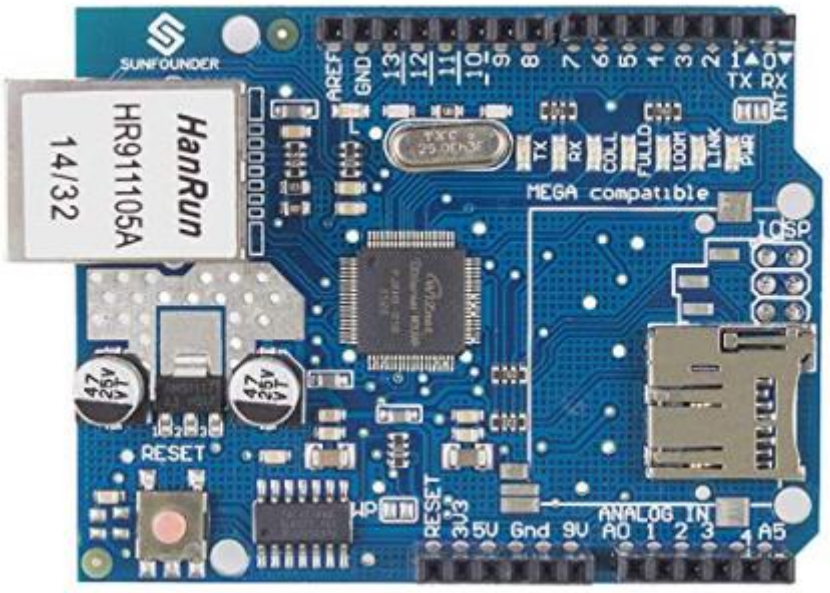
-4 1.0 interface beside ARFF and RESET.

-IOREF

-micro-SD shield

-PoE power-over-Ethernet, standard IEEE802.3af

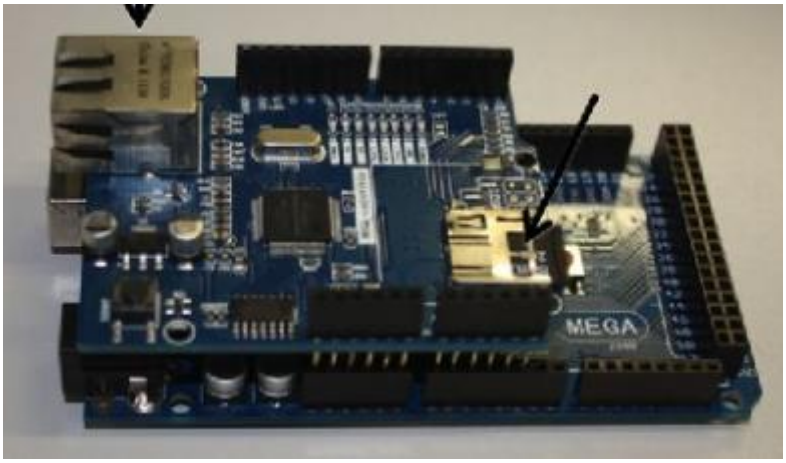
-1 x Ethernet Shield W5100 R3 Network Lan Board UNO Mega 2560 Duem for Arduino



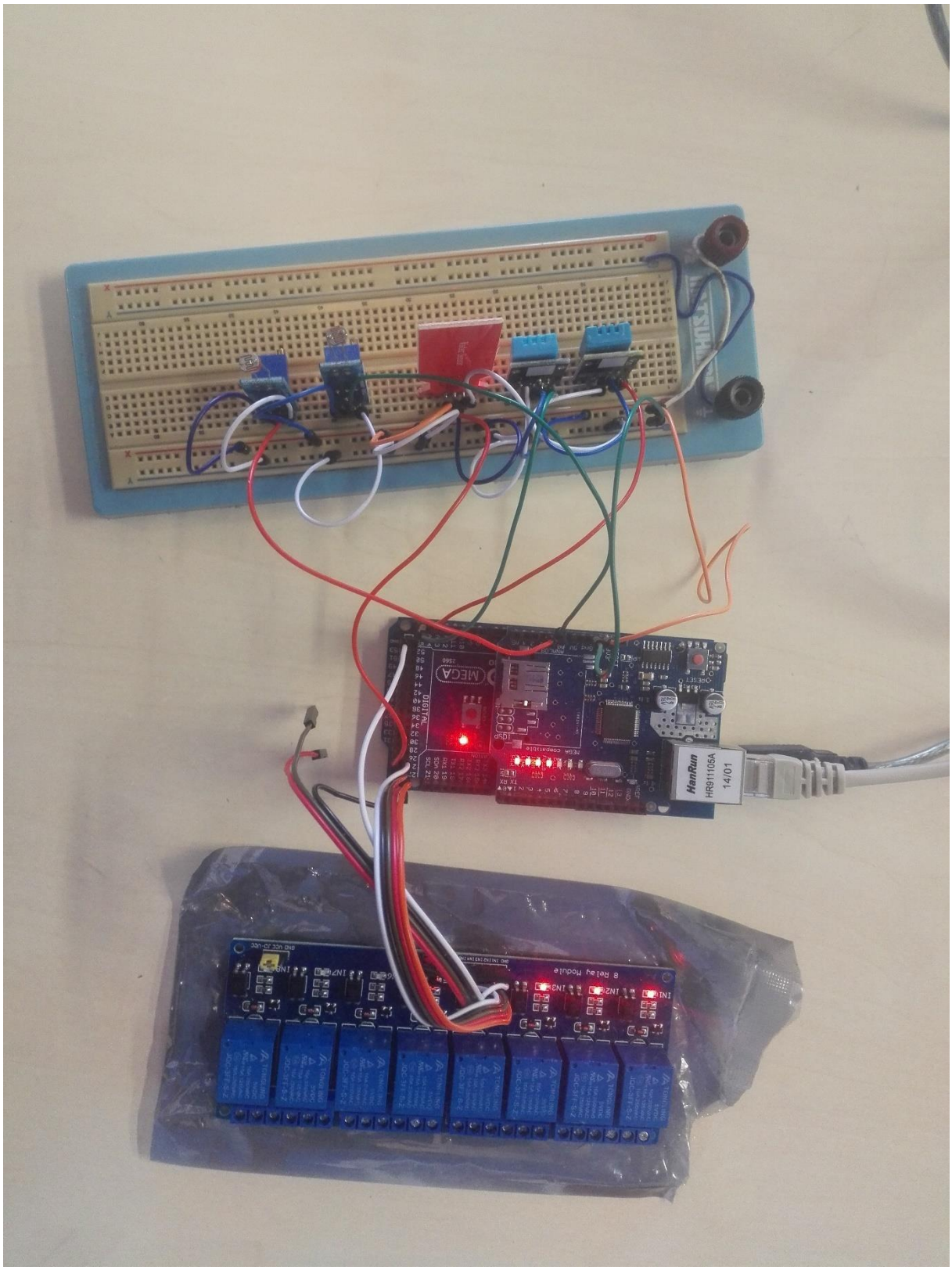
3.8 Arduino Ethernet Shield V1

Το Arduino επικοινωνεί τόσο με την κάρτα W5100 όσο και με την κάρτα SD χρησιμοποιώντας το δίαυλο SPI (μέσω της κεφαλίδας ICSP). Αυτό είναι στα ψηφιακά pin 50, 51 και 52 στο Mega. Ο ακροδέκτης 10 χρησιμοποιείται για την επιλογή του W5100 και ο ακροδέκτης 4 για την κάρτα SD. Αυτές οι ακίδες δεν μπορούν να χρησιμοποιηθούν για γενικές εισόδους/εξόδους. Στο Mega Arduino, η ακίδα SS (53), δεν χρησιμοποιείται για την επιλογή της κάρτας W5100 ή της κάρτας SD, αλλά πρέπει να διατηρηθεί ως έξοδος διαφορετικά η διεπαφή SPI δεν θα λειτουργήσει.
Επειδή το W5100 και η κάρτα SD μοιράζονται το ίδιο δίαυλο SPI, μόνο ένα μπορεί να είναι ενεργός κάθε φορά. Γι'αυτό τον λόγο απενεργοποιούμε την κάρτα SD, ορίζοντας τον ακροδέκτη 4 ως έξοδο και γράφουμε στον κώδικα 'high' σε αυτό το σημείο. Επίσης ορίζουμε την ψηφιακή ακίδα 10 ως υψηλή έξοδο.
Η πλακέτα περιέχει μια σειρά από ενημερωτικά LED:

- PWR: υποδεικνύει ότι η πλακέτα και η ασπίδα είναι τροφοδοτημένες
- LINK: υποδηλώνει την παρουσία μιας σύνδεσης δικτύου και αναβοσβήνει όταν η ασπίδα μεταδίδει ή λαμβάνει δεδομένα
- FULLD: υποδεικνύει ότι η σύνδεση δικτύου είναι πλήρως αμφίδρομη
- 100M: υποδηλώνει την παρουσία σύνδεσης δικτύου 100 Mb/s (σε αντίθεση με 10 Mb/s)
- RX: αναβοσβήνει όταν η ασπίδα λαμβάνει δεδομένα
- TX: αναβοσβήνει όταν η ασπίδα στέλνει δεδομένα
- COLL: αναβοσβήνει όταν εντοπίζονται συγκρούσεις δικτύου

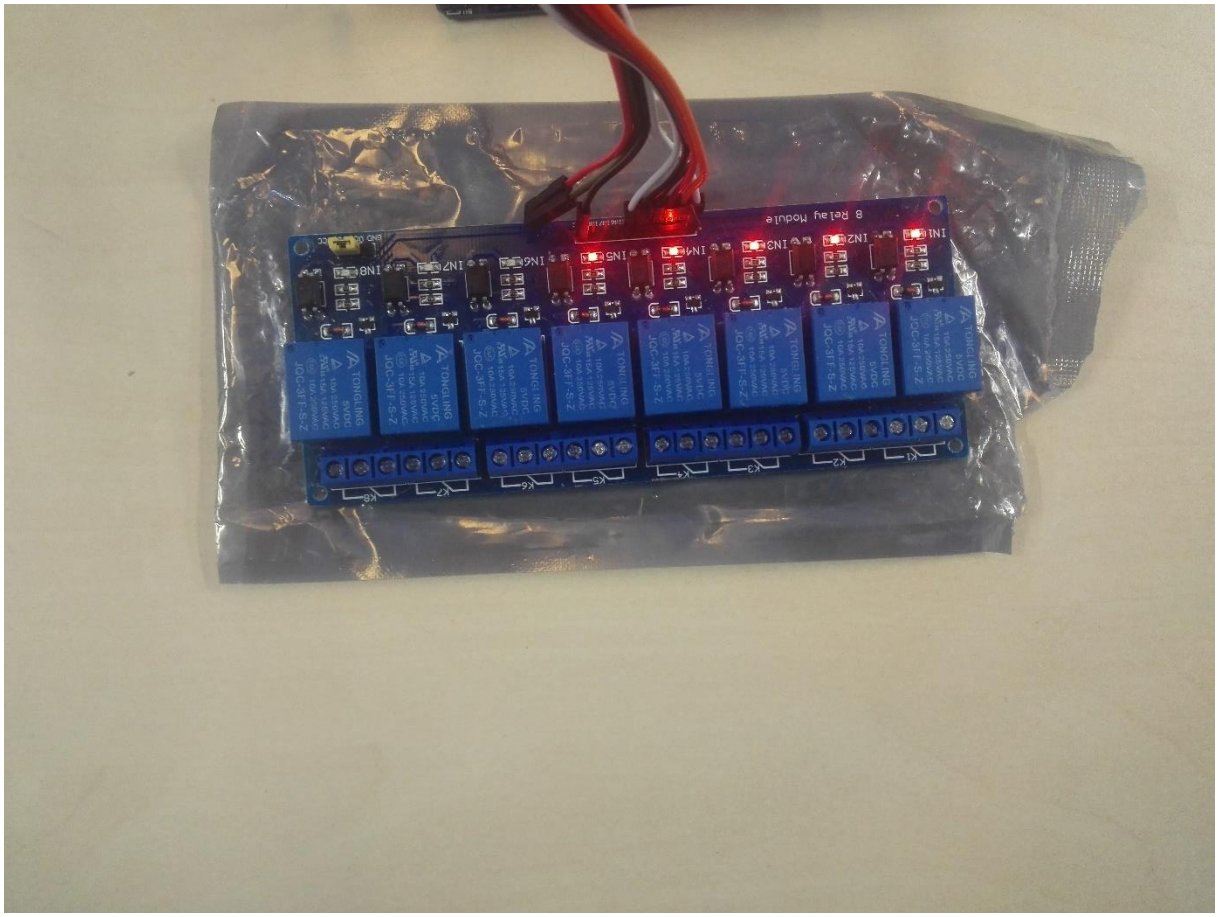


3.9 Τοποθέτηση πλακέτας ethernet στον Arduino atmega2560



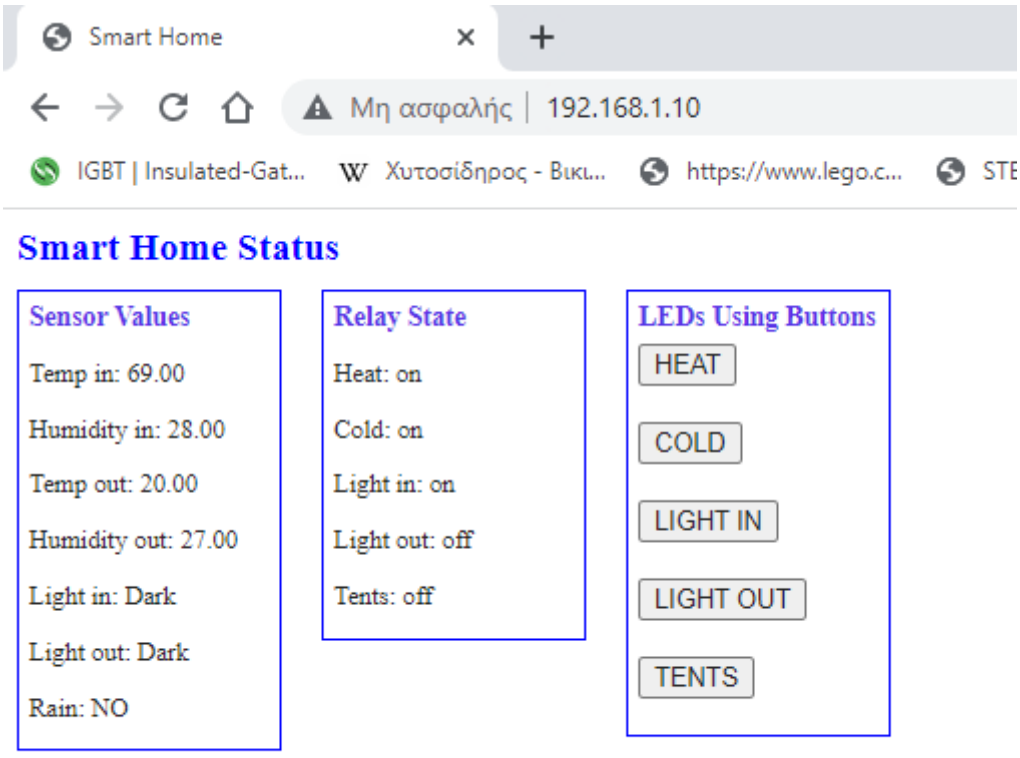
3.10 φωτογραφία της εργασίας που υλοποιεί το σύστημα απομακρυσμένης διαχείρισης σπιτιού

Στην φωτογραφία 3.10 φαίνονται τα αισθητήρια στο ράστερ που συνδέονται στην κεντρική μονάδα του Arduino καθώς και τα ρελέ που είναι οπλισμένα από τις φωτεινές ενδείξης των λυχνιών LEDs.



3.11 φωτογραφία από τα ρελέ

Από την φωτογραφία 3.11 φαίνονται τα ρελέ, όπου στο πάνω μέρος φαίνονται τα leds που είναι αναμμένα και αυτό σημαίνει ότι τα αντίστοιχα ρελέ είναι οπλισμένα και έχουν θέσει τις συσκευές που υποτίθεται θα ήταν συνδεδεμένες σε αυτά , σε κατάσταση λειτουργίας.
Κάτω από τα ρελέ φαίνονται οι κλέμες που μπορούμε να συνδέσουμε τις συσκευές του σπιτιού. Οι κλέμες αντέχουν τάση λειτουργίας 220volt



3.12 φωτογραφία από την ιστοσελίδα smart home

Από την φωτογραφία 3.12 βλέπουμε τα περιεχόμενα της ιστοσελίδας από την οποία διαχειριζόμαστε το σπίτι.

Στην πρώτη στήλη έχουμε τις τιμές των αισθητηρίων σε πραγματικό χρόνο που ανεβάζει στον server ο Arduino. Δηλαδή έχουμε θερμοκρασία και υγρασία επί % εντός σπιτιού, θερμοκρασία και υγρασία εκτός σπιτιού, εσωτερικό φως σπιτιού και εξωτερικό φως και ύπαρξη βροχής.

Στην δεύτερη στήλη έχουμε τις συσκευές που θα ελέγχουμε: heat= καλοριφέρ, cold= κλιματιστικό, lihgtin= εσωτερικά φώτα, lightout= εξωτερικά φώτα, tents= τέντα μπαλκονιού. Δίπλα από τις συσκευές εμφανίζονται οι καταστάσεις που βρίσκονται αυτή την στιγμή, δλδ αν είναι ενεργοποιημένες εμφανίζεται ‘on’, και το αντίθετο σε κατάσταση ‘off’.

Στην τρίτη στήλη είναι τα εικονικά button on/off που αν τα αγγίξουμε θέτουμε ή όχι την αντίστοιχη συσκευή σε λειτουργία.

3.13 Κόστος υλοποίησης κατασκευής

Υλικό	**Κόστος (euro)**
Arduino ATmega2560	13
W5100 ethernet shield	15
dht11 temperature and humidity sensor	14 (2 τεμάχια)
ldr light sensor module	14 (2 τεμάχια)
water level sensor	8
arduino relay module 8 channel	13
σύνολο	77

Κ Ε Φ Α Λ Α Ι Ο 4ο

ΛΟΓΙΣΜΙΚΟ ΜΕΡΟΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΔΙΑΧΕΙΡΙΣΗΣ ΣΠΙΤΙΟΥ

4.1 Λογισμικό μέρος

Το λογισμικό της διπλωματικής αποτελείτε από το κώδικα εντολών που είναι εγκατεστημένος στο hardware του μικροελεγκτή atmega2560 και η συγγραφή του έγινε με την γλώσσα wiring στο προγραμματιστικό περιβάλλον Arduino IDE.

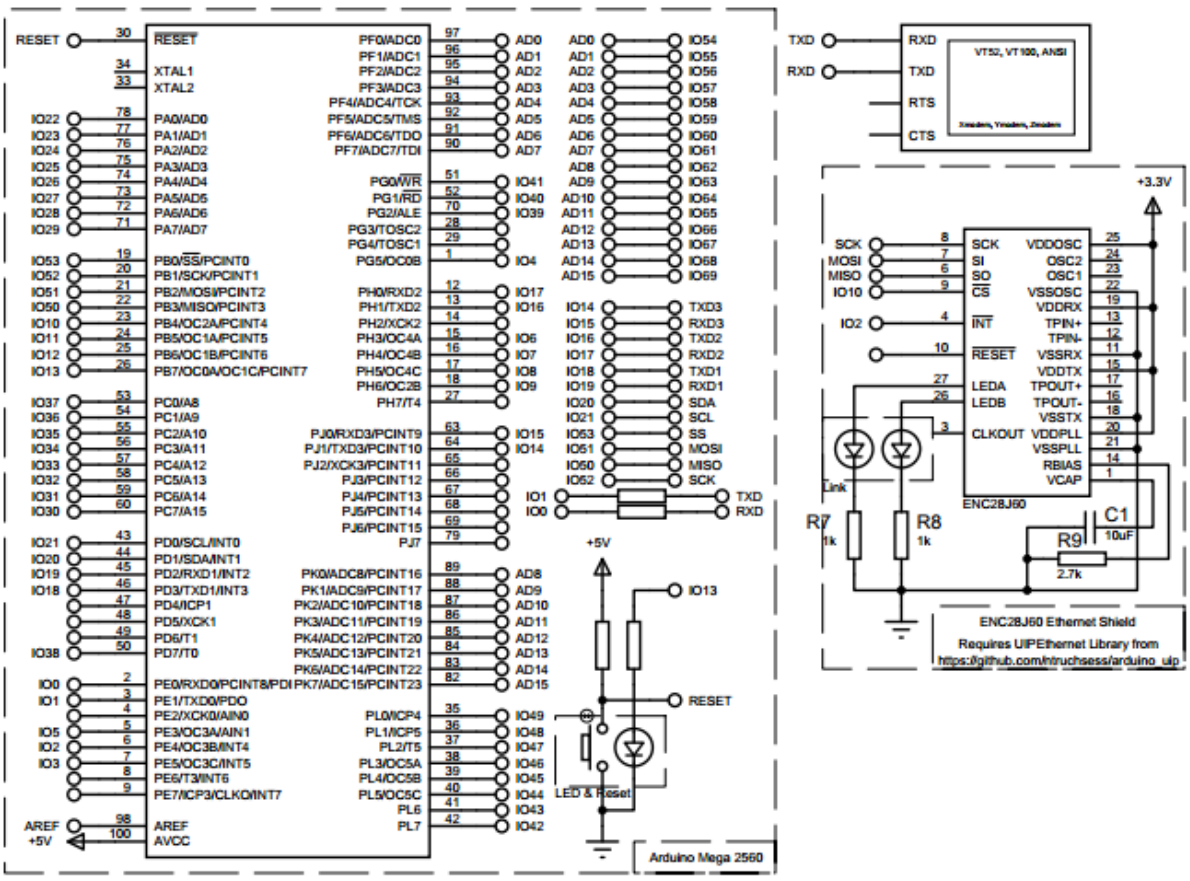
Με βάση αυτόν τον κώδικα που περιέχεται στο λογισμικό του Arduino υλοποιήθηκε το σύστημα για να διαχειριστούμε από μακριά ένα σπίτι. Συνοδεύεται από το διάγραμμα προσομοίωσης φτιαγμένο στο πρόγραμμα proteus, όπου προσομοιώνει το σύστημα στο 100% πριν προχωρήσουμε στην υλοποίησή του. Το πρόγραμμα proteus αναφέρεται συνοπτικά παρακάτω από το οποίο δημιουργήθηκε το σχηματικό διάγραμμα της εργασίας.

4.2 Πρόγραμμα προσομοίωσης Proteus

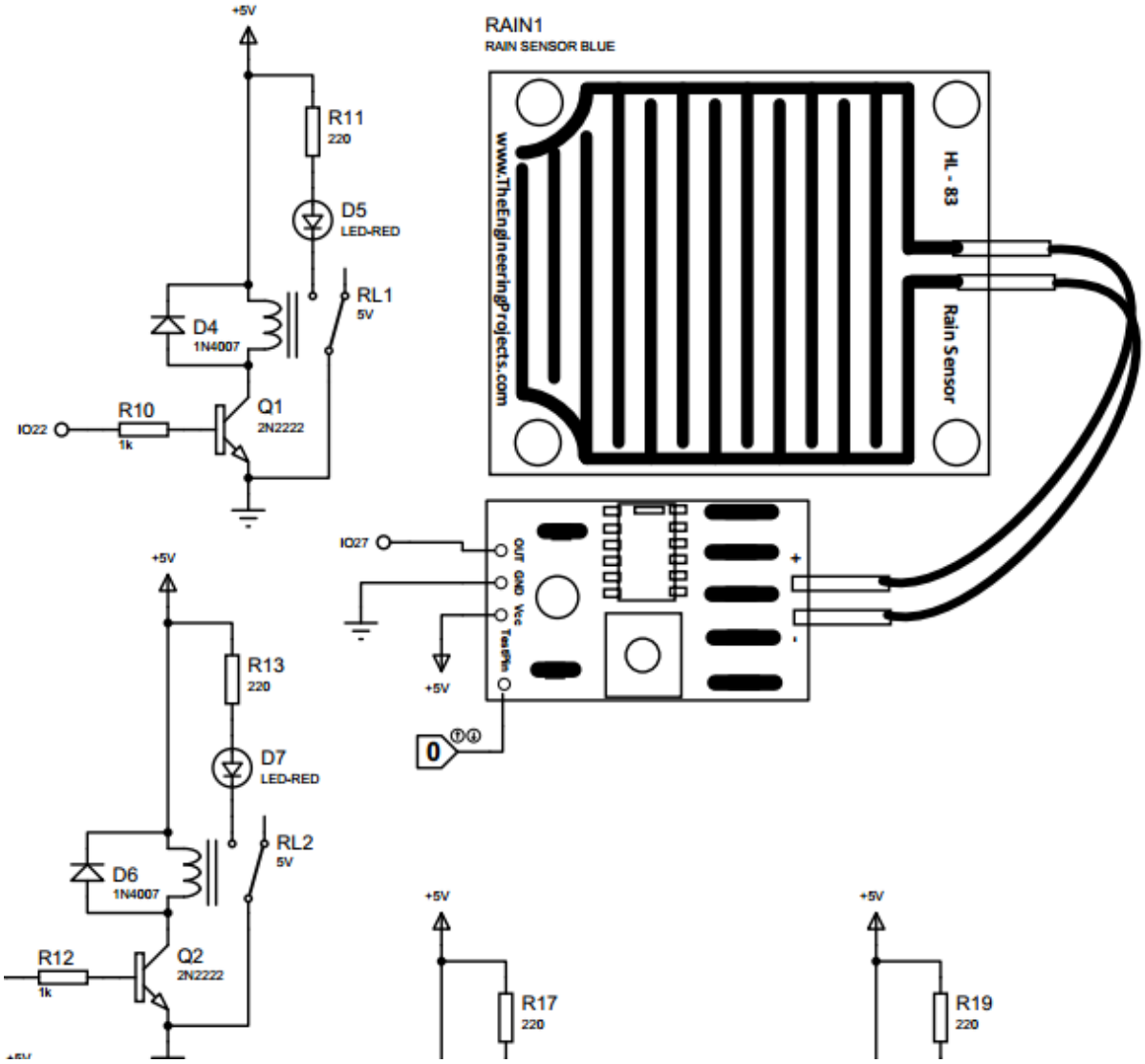
Το Proteus VSM (Virtual System Modeling) είναι η ικανότητά του να προσομοιώνει την αλληλεπίδραση μεταξύ του λογισμικού που εκτελείται σε έναν μικροελεγκτή και οποιωνδήποτε αναλογικών ή ψηφιακών ηλεκτρονικών που συνδέονται με αυτόν. Το μοντέλο μικροελεγκτή βρίσκεται στο σχηματικό μαζί με τα άλλα στοιχεία του σχεδιασμού του προϊόντος σας. Προσομοιώνει την εκτέλεση του αντικειμενικού κώδικα (κωδικός μηχανής), ακριβώς όπως ένα πραγματικό τσιπ. Εάν ο κωδικός προγράμματος γράψει σε μια θύρα, τα λογικά επίπεδα στο κύκλωμα αλλάζουν ανάλογα και αν το κύκλωμα αλλάξει την κατάσταση των ακίδων του επεξεργαστή, αυτό θα φανεί από τον κωδικό του προγράμματος, όπως και στο πραγματικό κύκλωμα.

Τα μοντέλα CPU VSM προσομοιώνουν πλήρως θύρες I/O, διακοπές, χρονόμετρα, USART και όλα τα άλλα περιφερειακά που υπάρχουν σε κάθε υποστηριζόμενο επεξεργαστή. Δεν είναι ένας απλός προσομοιωτής λογισμικού αφού η αλληλεπίδραση όλων αυτών των περιφερειακών με το εξωτερικό κύκλωμα είναι πλήρως μοντελοποιημένη σε επίπεδο κυματομορφής και επομένως ολόκληρο το σύστημα προσομοιώνεται.

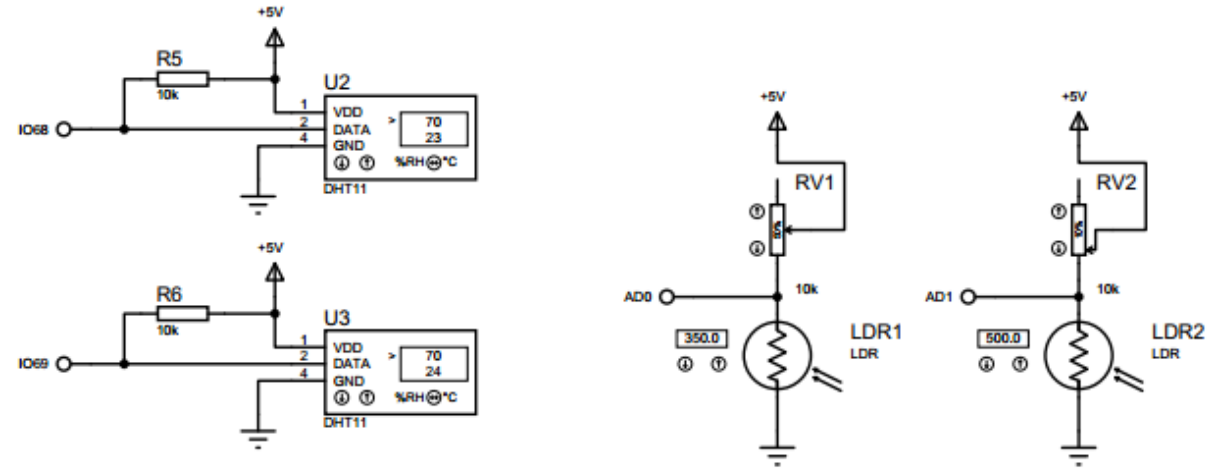
4.3 Σχηματικό διάγραμμα της εργασίας



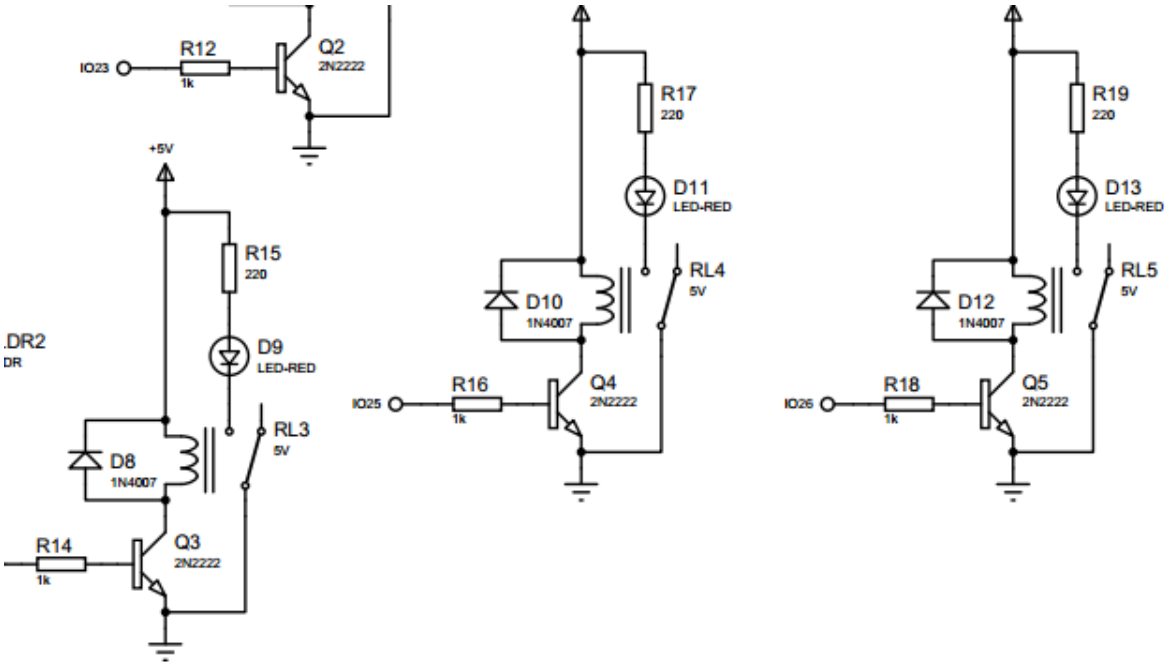
Αριστερά βρίσκεται ο Arduino 2560 και δεξιά είναι η πλακέτα ethernet με την οποία ανεβάζει την ιστοσελίδα στο web



Στο κέντρο φαίνεται ο αισθητήρας βροχής και αριστερά είναι δυο από τα πέντε ρελέ εξόδου που ενεργοποιούν τις συσκευές.



Φαίνονται αριστερά τα δύο αισθητήρια θερμοκρασίας και δεξιά οι δύο φωτοαντιστάσεις



Τα υπόλοιπα ρελέ που ενεργοποιούν τις συσκευές, όταν τα οπλίζει ο Arduino

4.4 Κώδικας για τον atmega2560

Ο κώδικας είναι γραμμένος σε γλώσσα προγραμματισμού C, όπου περιλαμβάνει και την δημιουργία ιστοσελίδας που αναρτάται στο web.

```
#include "dht.h"
#include <SPI.h>
#include <UIPEthernet.h>

dht DHT;

#define DHT11_PIN_in 68
#define DHT11_PIN_out 69

#define  REQ_BUF_SZ  60

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,1,100);

EthernetServer server(80);
EthernetClient client;

 char  HTTP_req[REQ_BUF_SZ]  = {0};
char req_index = 0;

boolean HEAT_state = false;
boolean COLD_state = false;
boolean LIGHTIN_state = false;
boolean LIGHTOUT_state  = false; boolean  TENTS_state  = false;

boolean  rain = false;

int Relay_1 = 22; //Temperature in
int Relay_2 = 23; //Temperature out
int Relay_3 = 24; //Rain
int Relay_4 = 25; //Light in
int Relay_5 = 26; //Light  out int  Rain = 27;

int heat_p = 53;
int cold_p = 54;
int tent_p = 55;
int light_in_p = 56;
int light_out_p = 57;

```

```
int sensorPin_in = A0;
int sensorValue_in  = 0; float Vout_in;
int sensorPin_out = A1;
int sensorValue_out  = 0; float Vout_out;

int chk_in, chk_out;
float h_in, t_in, h_out, t_out;

float heat_alert = 20;
float cold_alert = 30;
int light_in_alert = 147;
int light_out_alert = 390;

    void setup()
    {
    Serial.begin(115200);

    delay(200);

    pinMode(Relay_1, OUTPUT);
    pinMode(Relay_2, OUTPUT);
    pinMode(Relay_3, OUTPUT);
    pinMode(Relay_4, OUTPUT);
    pinMode(Relay_5, OUTPUT);    pinMode(Rain, INPUT);

    pinMode(heat_p, INPUT);
    pinMode(cold_p, INPUT);
    pinMode(tent_p, INPUT);
    pinMode(light_in_p, INPUT);
    pinMode(light_out_p, INPUT);

    Ethernet.begin(mac, ip);
    server.begin();
    Serial.print("server is at ");
    Serial.println(Ethernet.localIP());
    }

                void loop()
                {
                // listen for incoming clients
```

```
82                  client = server.available();
83                  if (client) {
84                  Serial.println("new client");
85                  // an http request ends with a blank line
86                  boolean currentLineIsBlank = true;
87                  while (client.connected()) {
88                  if (client.available()) {
89                  char c = client.read();
93
94                  Serial.write(c);
95
96  if     (req_index     <     (REQ_BUF_SZ     -     1))     {     97
HTTP_req[req_index] = c;          // save HTTP request character
98                      req_index++;
99                      }
100                     // if you've gotten to the end of the line
                        (received a newline
101                     // character) and the line is blank, the http
                        request has ended,
102                     // so you can send a reply
103                     if (c == '\n' && currentLineIsBlank) { 104     //
                        send a standard http response header
105                     client.println("HTTP/1.1 200 OK");
106
107                         //  remainder  of  header  follows  below,
                            depending on if
108                         // web page or XML page is requested
109                         // Ajax request - send XML file
110                         if (StrContains(HTTP_req, "ajax_inputs")) {
111                         // send rest of HTTP header
112                         client.println("Content-Type: text/xml");
113                         client.println("Connection: keep-alive");
114                         client.println();
115                         // send XML file containing input states
116                         XML_response(client);
117                         }
118                         else {  // web page request
119                         // send rest of HTTP header
120                         client.println("Content-Type: text/html");
121                         client.println("Connection: keep-alive");
```

```
107                         client.println();
123
124                         // HTTP request for web page
125                         // send web page - contains JavaScript with
    AJAX calls
126                         client.println("<!DOCTYPE html>");
127                         client.println("<html>");
128                         client.println("<head>");
129                         client.println("<title>Smart
                            Home</title>");
130                         client.println("<script>");
131
132                                     client.println("strHEAT =
                                        \"\";");
133                                     client.println("strCOLD =
                                        \"\";");
134                                     client.println("strLIGHTIN =
                                        \"\";");
135                                     client.println("strLIGHTOUT =
                                        \"\";");
136                                     client.println("strTENTS =
                                        \"\";");
137                                     client.println("var
                                        HEAT_state = 0;");
138                                     client.println("var
                                        COLD_state = 0;");
139                                     client.println("var
                                        LIGHTIN_state = 0;");
```

```
132                                         "var LIGHTOUT_state = 0;");
133                                         client.println("var
                                            TENTS_state = 0;");
142
143                         client.println("function GetArduinoIO()");
144                         client.println("{");
145                         client.println("nocache = \"&nocache=\" +
    Math.random() * 1000000;");
146                         client.println("var request = new
    XMLHttpRequest();");
147                         client.println("request.onreadystatechange
                            = function()");
148                         client.println("{");
149                         client.println("if (this.readyState == 4)
                            {");
150                         client.println("if (this.status == 200)
                            {");
151                         client.println("if (this.responseXML !=
                            null) {");
152                         client.println("// XML file received -
                            contains analog values, relay states");
153                         client.println("var count;");
154                         client.println("// get analog inputs");
155                         client.println("var num_an =
    this.responseXML.getElementsByTagName('analog').length;");
156                         client.println("for (count = 0; count <
                            num_an; count++) {");
157                         client.println(
    "document.getElementsByClassName(\"analog\")[count].innerHTML =");
158                         client.println(
    "this.responseXML.getElementsByTagName('analog')[count].childNodes
    [0].
    nodeValue;");
159                         client.println("}");
160                         client.println("// get relay state");
161                         client.println("var num_an =
    this.responseXML.getElementsByTagName('relay').length;");
162                         client.println("for (count = 0; count <
                            num_an; count++) {");
163                         client.println(
    "document.getElementsByClassName(\"relays\")[count].innerHTML =");
164                         client.println(
    "this.responseXML.getElementsByTagName('relay')[count].childNodes[
    0].n odeValue;");
```

```
143                         client.println("}");
166
167                                         client.println("// HEAT");
168                                         client.println("if
(this.responseXML.getElementsByTagName('relay')[0].childNodes[0].n
odeV alue === \"on\") {");
169                                         client.println(
"document.getElementById(\"HEAT\").innerHTML = \"HEAT is ON\";");
170                                         "HEAT_state = 1;");
171                                         client.println("}");
172                                         client.println("else {");
173                                         client.println(
"document.getElementById(\"HEAT\").innerHTML = \"HEAT is OFF\";");
174                                         client.println("HEAT_state   =
                                            0;");
175                                         client.println("}");
176                                         client.println("// COLD");
177                                         client.println("if
(this.responseXML.getElementsByTagName('relay')[1].childNodes[0].n
odeV alue === \"on\") {");
178                                         client.println(
"document.getElementById(\"COLD\").innerHTML = \"COLD is ON\";");
179                                         client.println("COLD_state   =
                                            1;");
180                                         client.println("}");
181                                         client.println("else {");
182                                         client.println(
"document.getElementById(\"COLD\").innerHTML = \"COLD is OFF\";");
183                                         client.println("COLD_state   =
                                            0;");
184                                         client.println("}");
185                                         client.println("// LIGHTIN");
186                                         client.println("if
(this.responseXML.getElementsByTagName('relay')[2].childNodes[0].n
odeV alue === \"on\") {");
187                                         client.println(
"document.getElementById(\"LIGHTIN\").innerHTML = \"LIGHTIN is
ON\";" );
188                                         client.println("LIGHT_state  =
                                            1;");
189                                         client.println("}");
190                                         client.println("else {");
191                                         client.println(
```

```
"document.getElementById(\"LIGHTIN\").innerHTML = \"LIGHTIN is
OFF\";");
167                                     client.println("LIGHTIN_state
                                        = 0;");
168                                     client.println("}");
169                                     client.println("//
                                        LIGHTOUT");
170                                     client.println("if
(this.responseXML.getElementsByTagName('relay')[3].childNodes[0].n
odeV alue === \"on\") {");
171                                     client.println(
"document.getElementById(\"LIGHTOUT\").innerHTML = \"LIGHTOUT is
ON\";");
172                                     client.println("LIGHTOUT_stat
                                        e = 1;");
173                                     client.println("}");
174                                     client.println("else {");
175                                     client.println(
"document.getElementById(\"LIGHTOUT\").innerHTML = \"LIGHTOUT is
OFF\";");
176                                     "LIGHTOUT_state = 0;");
177                                     client.println("}");
178                                     client.println("// TENTS");
179                                     client.println("if
(this.responseXML.getElementsByTagName('relay')[4].childNodes[0].n
odeV alue === \"on\") {");
180                                     client.println(
"document.getElementById(\"TENTS\").innerHTML = \"TENTS is
ON\";");
181                                     client.println("TENTS_state  =
                                        1;");
182                                     client.println("}");
183                                     client.println("else {");
184                                     client.println(
"document.getElementById(\"TENTS\").innerHTML = \"TENTS is
OFF\";");
185                                     client.println("TENTS_state  =
                                        0;");                       211
                                          client.println("}");
212
213                         client.println("}");
214                         client.println("}");
215                         client.println("}");
216                         client.println("}");
```

```
213                         client.println("// send HTTP GET request
                            with LEDs to switch on/off if any");
214                         client.println("request.open(\"GET\",
   \"ajax_inputs\" + strHEAT + strCOLD + strLIGHTIN + strLIGHTOUT +
   strTENTS + nocache, true);");
215                         client.println("request.send(null);");
216                         client.println("setTimeout('GetArduinoIO()'
                            ,
   1000);");
221
222                         client.println("strHEAT = \"\";");
223                         client.println("strCOLD = \"\";");
224                         client.println("strLIGHTIN = \"\";");
225                         client.println("strLIGHTOUT  = \"\";"); 226
                                client.println("strTENTS = \"\";");
227
228                        client.println("}");
229
230                         client.println("function GetButton1()");
231                         client.println("{");
232                         client.println("if (HEAT_state === 1) {");
233                         client.println("HEAT_state = 0;");
234                         client.println("strHEAT = \"&HEAT=0\";");
235                         client.println("}");
236                         client.println("else {");
237                         client.println("HEAT_state = 1;");
238                         client.println("strHEAT = \"&HEAT=1\";");
239                         client.println("}");
240                         client.println("}");
```

```
241
242                         client.println("function GetButton2()");
243                         client.println("{");
244                         client.println("if (COLD_state === 1) {");
245                         client.println("COLD_state = 0;");
246                         client.println("strCOLD = \"&COLD=0\";");
247                         client.println("}");
248                         client.println("else {");
249                         client.println("COLD_state = 1;");
250                         client.println("strCOLD = \"&COLD=1\";");
251                         client.println("}");
252                         client.println("}");
253
254                         client.println("function GetButton3()");
255                         client.println("{");
256                         client.println("if (LIGHTIN_state === 1)
                            {"); 257 client.println("LIGHTIN_state =
                            0;");
258                         client.println("strLIGHTIN                  =
                            \"&LIGHTIN=0\";"
    );
259                         client.println("}");
260                         client.println("else {");
261                         client.println("LIGHTIN_state = 1;");
262                         client.println("strLIGHTIN                  =
                            \"&LIGHTIN=1\";"
    );
263                         client.println("}");
264                         client.println("}");
265
266                         client.println("function GetButton4()");
267                         client.println("{");
268                         client.println("if (LIGHTOUT_state === 1)
                            {");
269                         client.println("LIGHTOUT_state = 0;");
270                         client.println("strLIGHTOUT =
    \"&LIGHTOUT=0\";");
```

```
266                         client.println("}");
267                         client.println("else {");
268                         client.println("LIGHTOUT_state = 1;");
269                         client.println("strLIGHTOUT =
    \"&LIGHTOUT=1\";");
270                         client.println("}");
271                         client.println("}");
277
278                         client.println("function GetButton5()");
279                         client.println("{");
280                         client.println("if (TENTS_state === 1) {");
281                         client.println("TENTS_state = 0;");
282                         client.println("strTENTS = \"&TENTS=0\";");
283                         client.println("}");
284                         client.println("else {");
278                         client.println("TENTS_state = 1;");
279                         client.println("strTENTS = \"&TENTS=1\";");
280                         client.println("}");
281                         client.println("}");
289
290                          client.println("</script>");
291
292                          client.println("<style>");
                        client.println(".IO_box {");
                        client.println("float: left;");
                        client.println("margin: 0 20px 20px
                        0;"); client.println("border: 1px solid
                        blue;"); client.println("padding: 0 5px
                        0 5px;"); client.println("width:
                        120px;"); client.println("}");
                        client.println("h1 {");
                        client.println("font-size: 120%;");
                        client.println("color: blue;");
                        client.println("margin: 0 0 10px 0;");
                        client.println("}");
                        client.println("h2 {");
                        client.println("font-size: 85%;");
                        client.println("color: #5734E6;");
                        client.println("margin: 5px 0 5px 0
                        ;"); client.println("}");
                        client.println("p, form, button {");
                        client.println("font-size: 80%;");
                        client.println("color: #252525;");
                        client.println("}");
                        client.println(".small_text {");
                        client.println("font-size: 70%;");
                        client.println("color: #737373;");
                        client.println("}");
                        client.println("</style>");
320                          client.println("</head>");
```

```
322                     client.println("<body
                        onload=\"GetArduinoIO()\">");
323                     client.println("<h1>Smart                Home
                        Status</h1>");
324
325                     client.println("<div class=\"IO_box\">");
326                     client.println("<h2>Sensor Values</h2>");
327                     client.println("<p>Temp in: <span
                        class=\"analog\">...</span></p>");
328                     client.println("<p>Humidity in: <span
                        class=\"analog\">...</span></p>");
329                     client.println("<p>Temp out: <span
```

```
class=\"analog\">...</span></p>");
325                         client.println("<p>Humidity out: <span
                            class=\"analog\">...</span></p>");
326                         client.println("<p>Light in: <span
                            class=\"analog\">...</span></p>");
327                         client.println("<p>Light out: <span
                            class=\"analog\">...</span></p>");
328                         client.println("<p>Rain: <span
                            class=\"analog\">...</span></p>");
329                         client.println("</div>");
335
336                         client.println("<div class=\"IO_box\">");
337                         client.println("<h2>Relay State</h2>");
338                         client.println("<p>Heat: <span
                            class=\"relays\">...</span></p>");
339                         client.println("<p>Cold: <span
                            class=\"relays\">...</span></p>");
340                         client.println("<p>Light in: <span
                            class=\"relays\">...</span></p>");
341                         client.println("<p>Light out: <span
                            class=\"relays\">...</span></p>");
342                         client.println("<p>Tents: <span
                            class=\"relays\">...</span></p>");
343                         client.println("</div>");
344
345                         client.println("<div class=\"IO_box\">");
346                         client.println("<h2>LEDs Using
                            Buttons</h2>");
347                         client.println("<button type=\"button\"
id=\"HEAT\" onclick=\"GetButton1()\">HEAT is OFF</button><br /><br
/>");
348                         client.println("<button type=\"button\"
id=\"COLD\" onclick=\"GetButton2()\">COLD is OFF</button><br /><br
/>");
349                         client.println("<button type=\"button\"
id=\"LIGHTIN\" onclick=\"GetButton3()\">LIGHTIN is OFF</button><br
/><br />");
350                         client.println("<button type=\"button\"
id=\"LIGHTOUT\" onclick=\"GetButton4()\">LIGHTOUT is
OFF</button><br /><br />");
351                         client.println("<button type=\"button\"
```

```
id=\"TENTS\" onclick=\"GetButton5()\">TENTS is OFF</button><br
/><br />");
345                         client.println("<p class=\"small_text\">D10
                            to D13 used by Ethernet shield</p>");
346                         client.println("</div>");
354
355                        client.print("</body>");
356
357                         client.println("</html>");
358                         }
359
360                       // display received HTTP request on serial port
361                       Serial.print(HTTP_req);
362                       // reset buffer index and all buffer elements to
                          0
363                       req_index = 0;
364                       StrClear(HTTP_req, REQ_BUF_SZ);
365
366                       break;
367                       }
368                       if (c == '\n') {
369                       // you're starting a new line
370                       currentLineIsBlank = true;
371                       }
372                       else if (c != '\r') {
373                       // you've gotten a character on the current
                          line
374                       currentLineIsBlank = false;
375                       }
376                       }
377                       }
378
379
380         // give the web browser time to receive the data
381         delay(1);
382         // close the connection:
383         client.stop();
384         Serial.println("client disconnected");
385         }
386         }
```

```
387
388         // checks if received HTTP request is switching Relays on/off
389         // also saves the state of the Relays
390         void SetRelays(void)
391         {
392         // HEAT relay
393         if (StrContains(HTTP_req, "HEAT=1")) {
394         HEAT_state = 1;  // save relay state
395         digitalWrite(Relay_1, HIGH);
396         }
397         else if (StrContains(HTTP_req, "HEAT=0")) {
398         HEAT_state = 0;  // save relay state
399         digitalWrite(Relay_1, LOW);
400         }
401
402         // COLD relay
403         if (StrContains(HTTP_req, "COLD=1")) {
404         COLD_state = 1;  // save relay state
405         digitalWrite(Relay_2, HIGH);
406         }
407         else if (StrContains(HTTP_req, "COLD=0")) {
408         COLD_state = 0;  // save relay state
409         digitalWrite(Relay_2, LOW);
410         }
411
412         // LIGHTIN relay
413         if (StrContains(HTTP_req, "LIGHTIN=1")) {
414         LIGHTIN_state = 1;  // save relay state
415         digitalWrite(Relay_3, HIGH);
416         }
417         else if (StrContains(HTTP_req, "LIGHTIN=0")) {
418         LIGHTIN_state = 0;  // save relay state
419         digitalWrite(Relay_3, LOW);
420         }
421
422         // LIGHTOUT relay
423         if (StrContains(HTTP_req, "LIGHTOUT=1")) {
424         LIGHTOUT_state = 1;  // save relay state
425         digitalWrite(Relay_4, HIGH);
426         }
```

```
422         else if (StrContains(HTTP_req, "LIGHTOUT=0")) {
423         LIGHTOUT_state = 0;  // save relay state
424         digitalWrite(Relay_4, LOW);
425         }
431
432         // TENTS relay
433         if (StrContains(HTTP_req, "TENTS=1")) {
434         TENTS_state = 1;  // save relay state
435         digitalWrite(Relay_5, HIGH);
436         }
437         else if (StrContains(HTTP_req, "TENTS=0")) {
438         TENTS_state = 0;  // save relay state
439         digitalWrite(Relay_5, LOW);
440         }
441         }
442
443     // send the XML file with analog values, switch status
444     //  and LED status
445     void XML_response(EthernetClient cl)
446     {
447     int analog_val;            // stores value read from analog
        inputs
448     int count;                 // used by 'for' loops
449     int sw_arr[] = {2, 3, 5};  // pins interfaced to switches
450
451     cl.print("<?xml version = \"1.0\" ?>");
452     cl.print("<inputs>");
453     // read analog inputs
454
455      chk_in = DHT.read11(DHT11_PIN_in);
456
457     h_in = DHT.humidity;
458     t_in = DHT.temperature;
459
460     cl.print("<analog>");
461     cl.print(h_in);
462     cl.println("</analog>");
463
464     cl.print("<analog>");
465     cl.print(t_in);
466     cl.println("</analog>");
```

```
467
468     delay(50);
469
470    chk_out = DHT.read11(DHT11_PIN_out);
471
472     h_out = DHT.humidity;
473     t_out = DHT.temperature;
474
475     cl.print("<analog>");
476     cl.print(h_out);
477     cl.println("</analog>");
478
479     cl.print("<analog>");
480     cl.print(t_out);
481     cl.println("</analog>");
482
483     delay(50);
484
485   sensorValue_in = analogRead(sensorPin_in); // read the value
    from the sensor
486
487    //Vout_in = sensorValue_in*5.0/1023.0;
488
489     cl.print("<analog>");
490     cl.print(sensorValue_in);  491
        cl.println("</analog>");
492
493     delay(50);
494
495   sensorValue_out  =  analogRead(sensorPin_out);  //  read  the  value
    from the sensor
496
497    //Vout_out = sensorValue_out*5.0/1023.0;
498
499     cl.print("<analog>");
500     cl.print(sensorValue_out);  501
        cl.println("</analog>");
502
503     rain = digitalRead(Rain);
504
505         cl.print("<analog>");
506         if (rain) {
```

```
505         cl.print("YES");
506         }
507         else {
508         cl.print("NO");
509         }
510         cl.println("</analog>");
513
514         // relay states
515         // HEAT
516         cl.print("<HEAT>");
517         if (HEAT_state) {
518         cl.print("on");
519         }
520         else {
521         cl.print("off");
522         }
523         cl.println("</HEAT>");
524
525         // COLD
526         cl.print("<COLD>");
527         if (COLD_state) {
528         cl.print("on");
529         }
530         else {
531         cl.print("off");
532         }
533         cl.println("</COLD>");
534
535         // LIGHTIN
536         cl.print("<LIGHTIN>");
537         if (LIGHTIN_state) {
538         cl.print("on");
539         }
540         else {
541         cl.print("off");
542         }
543         cl.println("</LIGHTIN>");
544
545         // LIGHTOUT
546         cl.print("<LIGHTOUT>");
547         if (LIGHTOUT_state) {
```

```
545         cl.print("on");
546         }
547         else {
548         cl.print("off");
549         }
550         cl.println("</LIGHTOUT>");
554
555         // TENTS
556         cl.print("<TENTS>");
557         if (TENTS_state) {
558         cl.print("on");
559         }
560         else {
561         cl.print("off");
562         }
563         cl.println("</TENTS>");
564
565     cl.print("</inputs>");
566     }
567
568         // sets every element of str to 0 (clears array)
569         void StrClear(char *str, char length)
570         {
571         for (int i = 0; i < length; i++) {
572         str[i] = 0;
573         }
574         }
575
576     // searches for the string sfind in the string str
577     // returns 1 if string found
578     // returns 0 if string not found
579     char StrContains(char *str, char *sfind)
580     {
581     char found = 0;
582     char index = 0; 583 char len;
584
585     len = strlen(str);
586
587                 if (strlen(sfind) > len) {
588                 return 0;
```

```
587                     }
588                     while (index < len) {
589                     if (str[index] == sfind[found]) {
590                     found++;
591                     if (strlen(sfind) == found) {
592                     return 1;
593                     }
594                     }
595                     else {
596                     found = 0;
597                     }600           index++;
601       }
602
603       return 0;
604       }
```

4.5 Επεξήγηση του κώδικα

ΠΑΡΑΚΑΤΩ ΓΙΝΕΤΑΙ ΣΥΝΟΠΤΙΚΗ ΕΠΕΞΗΓΗΣΗ ΤΟΥ ΚΩΔΙΚΑ.

Από την γραμμή 126 έως 359:

Ο Arduino φτιάχνει την ιστοσελίδα σε γραμμές και στήλες με την συνάρτηση println.

Από την γραμμή 76 έως 122:

Η σελίδα ανεβαίνει στον server και παραμένει εκεί για όσο χρόνο τρέχει ο κώδικας, είναι δυναμική σελίδα και όχι στατική δηλαδή να παραμένει μόνιμα στον server.

Από την γραμμή 360 έως 386:

Ταυτόχρονα γίνεται διασύνδεση του Arduino με τον server για να στέλνει σε πραγματικό χρόνο (κάθε στιγμή) τις τιμές από τα αισθητήρια στην αντίστοιχη στήλη της σελίδας.

Από την γραμμή 445 έως 512:

Το Arduino διαβάζει τις τιμές από τα αισθητήρια και τις στέλνει στην σελίδα σε πραγματικό χρόνο.

Από την γραμμή 514 έως 604:

Υπάρχει αμφίδρομη επικοινωνία, δηλαδή όταν θέτουμε από την σελίδα τους διακόπτες σε κατάσταση on ή off, τότε ο Arduino διαβάζει αυτές τις καταστάσεις από την σελίδα, ώστε να ανοιγοκλείσει ανάλογα τα ρελέ που αντιστοιχούν στις συσκευές που ελέγχουμε.

Από την γραμμή 390 έως 441:

Ο Arduino ανοιγοκλείνει τα ρελέ ανάλογα με τις καταστάσεις (on/off) που διαβάζει από την σελίδα.

Γενικά η ιστοσελίδα βρίσκεται μέσα στον κώδικα που τρέχει ο Arduino και ανεβαίνει προσωρινά στον σερβερ για όσο χρόνο έχουμε ανοιχτό τον Arduino.
Ο οποίος στέλνει στην σελίδα κάθε στιγμή τις τιμές που διαβάζει από τα αισθητήρια.
Η επικοινωνία της σελίδας με τον Arduino είναι αμφίδρομη γιατί μπορούμε να του στείλουμε από την σελίδα τις καταστάσεις on/off των ρελέ.

Κ Ε Φ Α Λ Α Ι Ο 5ο

ΣΥΝΟΨΗ- ΠΡΟΕΚΤΑΣΕΙΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

5.1 Σύνοψη-επίλογος

Είδαμε κάποιες από τις τεράστιες δυνατότητες που έχει η τεχνολογία για να κάνει την ζωή του ανθρώπου πιο εύκολη, πιο ασφαλή και πιο ενδιαφέρουσα. Επίσης στην περίληψη αναφέρθηκαν τα πλεονεκτήματα που θα έχει κάποιος αν προχωρήσει στην υλοποίηση του έξυπνου σπιτιού.

Παρόλα αυτά οι καταναλωτές ακόμα διστάζουν να κάνουν μία τέτοια μετατροπή κυρίως λόγω κόστους. Πάντοτε θα υπάρχουν δύσκολες οικονομικά εποχές και πολλοί δεν πιστεύουν θα αποσβέσει το κόστος μία τέτοια μετατροπή. Αλλά από την υπόθεση στην εφαρμογή θα φανεί τελικά αν θα αποσβέσει το κόστος, όπου τα μαθηματικά δείχνουν απόσβεση.

Άρα το έξυπνο σπίτι βρίσκεται σε πρώιμο στάδιο και νέες οικοδομές που χτίζονται δεν έχουν προεγκαταστάσεις για εφαρμογή έξυπνου σπιτιού και έτσι λοιπόν μεγάλο μέρος της αγοράς δεν κατακτήθηκε ακόμα.

Όμως θα έπρεπε γιατί ζούμε σε έναν κόσμο με κλιματική αλλαγή και απειλείται από φυσικές καταστροφές που προκαλούνται από την μόλυνση της γης. Είναι πολύ σημαντικό να επεκταθεί αυτή η εφαρμογή σε όλο και περισσότερους ανθρώπους και η ανθρωπότητα να αναλογιστεί ότι διαθέτοντας χρηματικούς πόρους για αυτή την εφαρμογή , στην ουσία διαμορφώνει έναν υγιή πλανήτη με πράσινη ανάπτυξη.

Αν αξιοποιήσουμε σωστά την τεχνολογία, όπως με το έξυπνο σπίτι τότε θα σώσουμε τον πλανήτη από την κλιματική αλλαγή γιατί θα εξοικονομήσουμε τεράστια ποσά ενέργειας.

5.2 Προεκτάσεις (Internet of Things)

Ένα έξυπνο σπίτι θα μπορούσε να γίνει ακόμα πιο έξυπνο με το internet of things. Ο όρος “Internet of Things” επινοήθηκε από τον Κέβιν Άστον της Procter & Gamble, και αργότερα από το MIT's Auto-ID Center, το 1999. (Ashton, 2009) Χρησιμοποιείται για να περιγράψει τη σύνδεση συσκευών και φυσικών αντικειμένων (Things) με τους υπολογιστές και το διαδίκτυο. Περιλαμβάνει ένα σύνολο αντικειμένων που χρησιμοποιούν οι άνθρωποι στην καθημερινότητα και η σύνδεσή τους στο διαδίκτυο έχει ως στόχο τη δυνατότητα

ελέγχου από τον άνθρωπο καθώς και τη μεταξύ τους αλληλεπίδραση. (Holler J., 2014) Το "Διαδίκτυο των Πραγμάτων" είναι ένα σύνολο 'πραγμάτων' ενσωματωμένα με ηλεκτρονικά, λογισμικά, αισθητήρες κλπ, τα οποία είναι συνδεδεμένα με τη χρήση του διαδικτύου για τη συλλογή και ανταλλαγή δεδομένων μεταξύ τους. (Yang, et al., 2017)

Στο έξυπνο αυτό σπίτι, η πόρτα ανοίγει αυτόματα μόλις πλησιάσει ο ιδιοκτήτης του σπιτιού με αναγνώριση προσώπου. Όταν αναχωρεί, κλείνει με ταυτόχρονη εντολή να κατέβουν τα παντζούρια, να ενεργοποιηθούν τα συστήματα ασφαλείας, κάμερες, αισθητήρες κίνησης, επιτρέποντας τον πλήρη έλεγχο και χειρισμό των συστημάτων εξ' αποστάσεως.

Η λειτουργία του φωτισμού διαχειρίζεται με τρόπο που να προσφέρει την άνεση με ταυτόχρονη εξοικονόμηση ενέργειας. Προσαρμόζεται αυτομάτως, αναλόγως την παρουσία ή όχι του ανθρώπου στο σπίτι ή σε κάποιο συγκεκριμένο χώρο, από τις συνθήκες φυσικού φωτισμού, την ώρα της ημέρας, ή ακόμη και τις συνήθειες μας όπως για παράδειγμα να χαμηλώνει αυτόματα όταν γίνεται παρακολούθηση κάποιας ταινίας.

Η θερμοκρασία του χώρου, όπως και η ανανέωση του εσωτερικού αέρα καθώς και η συνεχής παρακολούθηση της ποιότητας του αυτορυθμίζεται πλήρως χωρίς καμία παρέμβαση του ανθρώπου λαμβάνοντας δεδομένα από διάφορους εσωτερικούς και εξωτερικούς αισθητήρες και έξυπνους θερμοστάτες. Η βελτιστοποίηση της λειτουργίας αυτών των συστημάτων, θα συμβάλει σε μεγάλο βαθμό στην εξοικονόμηση ενέργειας στον οικιακό τομέα.

Οι οικιακές συσκευές θα αναβαθμιστούν επίσης. Διάφορες εταιρίες σήμερα έχουν κατασκευάσει πλυντήρια, στεγνωτήρια, ακόμη και ψυγεία που μπορούν να ελεγχθούν από ψηφιακούς βοηθούς.

Φυσικά οι έξυπνες εφαρμογές δεν θα μπορούσαν να λείπουν από το ψυχαγωγικό κομμάτι. Έξυπνες τηλεοράσεις, συστήματα ήχου και φωτισμός συνυπάρχουν και πλέον με τη δυνατότητα φωνητικού ελέγχου συνδυάζονται σε μια ενιαία ψυχαγωγική εμπειρία με πολύ απλό χειρισμό, αρκεί μια φωνητική εντολή.

Πηγή https://www.pachtas.gr/0DDB64B3.el.aspx

ΠΕΡΙΕΧΟΜΕΝΑ

Τίτλος πτυχιακής εργασίας……1

Πρόλογος……...2

Περίληψη……3

ABSTRACT……..4

Ευχαριστίες……...6

Διάρθρωση κειμένου εργασίας……7

ΚΕΦΑΛΑΙΟ 1Ο ……8

1. Εισαγωγή……...8

1.1. Πως προέκυψε η ανάγκη του έξυπνου σπιτιού……8

1.2. Τι είναι το έξυπνο σπίτι……8

1.3. Παραδείγματα σεναρίων του έξυπνου σπιτιού……9

1.4. Το αντικείμενο της διπλωματικής εργασίας……10

1.5. Μπλοκ-διάγραμμα συστήματος έξυπνου σπιτιού……..11

ΚΕΦΑΛΑΙΟ 2ο ……...12

2. Θεωρητικό υπόβαθρο……12

2.1. Σύστημα απομακρυσμένης διαχείρισης σπιτιού……..12

2.2 Η υπολογιστική πλατφόρμα Arduino……..13

2.3 Περιβάλλον προγραμματισμού Arduino……20

2.4 Πρωτόκολλο ethernet……...24

2.5 Το παγκόσμιο δίκτυο Web……24

2.6 Πρωτόκολλο http……..27

2.7 Γλώσσα προγραμματισμού html……27

ΚΕΦΑΛΑΙΟ 3ο ……28

3.1 Μέρη που αποτελείται το σύστημα απομακρυσμένης διαχείρισης σπιτιού……28

3.2 Συνολική περιγραφή του συστήματος……28

3.3 Αισθητήριο θερμοκρασίας……30

3.4 Αισθητήριο φωτός……30

3.5 Αισθητήριο βροχής……32

3.6 Ρελέ στις εξόδους της κεντρικής μονάδας……33

3.7 Κεντρική μονάδα επεξεργασίας……34

3.8 Arduino ethernet shield……39

3.9 έως 3.12 φωτογραφίες εργασίας και επεξήγηση……..40

3.13 Κόστος εργασίας……44

ΚΕΦΑΛΑΙΟ 4ο ……45

4.1 Λογισμικό μέρος Proteus ……45

4.2 Πρόγραμμα προσομοίωσης……45

4.3 Σχηματικό διάγραμμα……46

4.4 κώδικας για τον atmega2560……48

4.5 Επεξήγηση του κώδικα……67

ΚΕΦΑΛΑΙΟ 5ο ……69

5.1 Σύνοψη-επίλογος……69

5.2 Προεκτάσεις (Internet of Things)……69

βιβλιογραφία……73

ΒΙΒΛΙΟΓΡΑΦΙΑ INTERNET

1) https://dione.lib.unipi.gr/xmlui/bitstream/handle/unipi/12200/Karali_mtd1706.pdf?sequence=3&isAllowed=y
2) http://repository.library.teimes.gr/xmlui/bitstream/handle
3) http://okeanis.lib.puas.gr/xmlui/bitstream/handle/123456789/2619/Smart%20Home.pdf?sequence=1&isAllowed=y
4) https://el.wikipedia.org/wiki/HTML
5) https://maredu.hcg.gr/modules/document/file.php/MAK265/
6) https://www.iqpowersystem.gr/services/ejipno-spiti
7) https://smarterhome.gr/smart-home/smart-home-ofeli/
8) https://www.grohe.gr/el_gr/smarthome/why-the-smart-home-is-the-future/
9) https://dione.lib.unipi.gr/xmlui/bitstream/handle/unipi/10090/Mariolas_Panagiotis.pdf?sequence=1&isAllowed=y
10) https://www.spitogatos.gr/blog/smart-homes-to-mellon-ton-spition
11) https://simerini.sigmalive.com/article/2015/6/29/to-exupno-spiti-tou-mellontos/
12) https://arduinogetstarted.com/tutorials/arduino-water-sensor
13) https://create.arduino.cc/projecthub/electropeak/make-a-liquid-level-indicator-with-arduino-596bd3
14) https://www.electroduino.com/ldr-sensor-module-how-ldr-sensor-works/
15) https://en.wikipedia.org/wiki/World_Wide_Web
16) https://en.wikipedia.org/wiki/Proteus_Design_Suite
17) https://el.wikipedia.org/wiki/%CE%A0%CF%81%CF%89%CF%84%CF%8C%CE%BA%CE%BF%CE%BB%CE%BB%CE%BF_%CE%9C%CE%B5%CF%84%CE%B1%CF%86%CE%BF%CF%81%CE%AC%CF%82_%CE%A5%CF%80%CE%B5%CF%81%CE%BA%CE%B5%CE%B9%CE%BC%CE%AD%CE%BD%CE%BF%CF%85
18) https://en.wikipedia.org/wiki/Home_automation