



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

**ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ
ΣΠΟΥΔΩΝ «ΠΛΗΡΟΦΟΡΙΚΗ ΚΑΙ ΥΠΟΛΟΓΙΣΤΙΚΗ
ΒΙΟΙΑΤΡΙΚΗ»**

**Εισαγωγή στην Open Source Πλατφόρμα Μικροελεγκτών
Arduino**

Καραγεώργος Παναγιώτης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων

Σταμούλης Γεώργιος

Λαμία 2017



UNIVERSITY OF THESSALY

SCHOOL OF SCIENCE

**INFORMATICS AND COMPUTATIONAL
OF BIOMEDICINE**

**Introduction to the Open Source Arduino Microcontroller
Platform**

Karagergos Panagiotis

MASTER THESIS

Supervisor

George Stamoulis

Lamia 2017

«Υπεύθυνη Δήλωση μη λογοκλοπής και ανάληψης προσωπικής ευθύνης»

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, και γνωρίζοντας τις συνέπειες της λογοκλοπής, δηλώνω υπεύθυνα και ενυπογράφως ότι η παρούσα εργασία με τίτλο [«Εισαγωγή στην Open Source Πλατφόρμα Μικροελεγκτών Arduino»] αποτελεί προϊόν αυστηρά προσωπικής εργασίας και όλες οι πηγές από τις οποίες χρησιμοποίησα δεδομένα, ιδέες, φράσεις, προτάσεις ή λέξεις, είτε επακριβώς (όπως υπάρχουν στο πρωτότυπο ή μεταφρασμένες) είτε με παράφραση, έχουν δηλωθεί κατάλληλα και ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύνανται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο ΔΗΛΩΝ

Ημερομηνία

Υπογραφή

Περιεχόμενα

ΚΕΦΑΛΑΙΟ 1	Μικροελεγκτές & Arduino	5
Εισαγωγή		5
Μικροελεγκτές		6
Προγραμματισμός Μικροελεγκτών		8
Η πλατφόρμα Arduino		9
Οι πλακέτες –Boards		10
Αρχιτεκτονική της πλακέτας Arduino		13
ΚΕΦΑΛΑΙΟ 2	Το Περιβάλλον Ανάπτυξης	16
Εγκατάσταση-Ρύθμιση του IDE		16
Γράφοντας κώδικα Arduino		18
Ο κώδικας (sketch) Arduino		19
Διασύνδεση (Interfacing) με το Serial Monitor		20
Έλεγχος θυρών I/O		22
Δοκιμάζοντας τον κώδικά σε ένα προσομοιωτή Arduino		23
ΚΕΦΑΛΑΙΟ 3	Επέκταση του Arduino με τη χρήση Shields	26
Χρήση Βιβλιοθηκών		27
Τρέχοντας τον κώδικα απευθείας από τον μικροεπεξεργαστή		28
ΚΕΦΑΛΑΙΟ 4	Βασικές Αρχές Αισθητήρων και Ενεργοποιητών	31
Ψηφιακοί Αισθητήρες		32
Αντιστάσεις Pull up-Down και Αισθητήρες		33
Εισαγωγή στη θεωρία δειγματοληψίας		35
Μετατροπή A/D		35
Παραδείγματα και αρχές λειτουργίας αισθητήρων		37
Ενεργοποιητές – Διακόπτες		44
Αναλογικοί Αισθητήρες		48
Ενσωμάτωση των αισθητήρων σε κύκλωμα		51
ΚΕΦΑΛΑΙΟ 5	Διαχείριση δεδομένων από αισθητήρες περιβάλλοντος	55
Διάταξη κυκλώματος		56

Σύνδεση με το Web Service και αποστολή δεδομένων.....	60
Ανάλυση κώδικα.....	64
ΚΕΦΑΛΑΙΟ 6 Arduino & ασφάλεια	70
Βασική Κωδικοποίηση με τη χρήση της συνάρτησης SHA1 hash	72
Κρυπτογράφηση των δεδομένων	74
Κώδικας Arduino με κρυπτογράφηση AES.....	74
ΚΕΦΑΛΑΙΟ 7 Εφαρμογές με Arduino	78
Παράδειγμα Ethernet Client	78
Ελέγχοντας το Arduino από περιβάλλον Android.....	84
Επίλογος	91

ΚΕΦΑΛΑΙΟ 1 Μικροελεγκτές & Arduino

Εισαγωγή

Η παρούσα διπλωματική εργασία αναπτύχθηκε ως μέρος μιας προσπάθειας για εισαγωγή της πλατφόρμας μικροεπεξεργαστών Arduino στη διδασκαλία μαθημάτων σε ακαδημαϊκό επίπεδο, που αφορούν τα ενσωματωμένα συστήματα υπολογιστών. Για τον λόγο αυτό το κείμενο είναι γραμμένο εξ αρχής σαν ένας πλήρης οδηγός για τον χρήστη που επιθυμεί να εντρυφήσει στον κόσμο της υλοποίησης ηλεκτρονικών συστημάτων έχοντας ως βάση τις πλακέτες και το software Arduino.

Στο πλαίσιο αυτό έχει γίνει μια προσπάθεια να αναλυθεί ένα μεγάλο τμήμα της αρχιτεκτονικής, των διατάξεων, της γραφής κώδικα και των εφαρμογών που υπάρχουν για το Arduino. Επιπρόσθετα, γίνεται εκτενής αναφορά στους διαφορους τύπους αισθητήρων που είναι απαραίτητοι ώστε να λειτουργήσει μια πλήρης διάταξη, καθώς και των βασικών γνώσεων για μια μετατροπή A/D. Παράλληλα, μια σημαντική πτυχή των εφαρμογών που αφορούν το Arduino είναι η χρήση του για την διαχείριση δεδομένων που προέρχονται από αισθητήρες περιβάλλοντος.

Τέλος, η ανάλυση της πλατφόρμας δεν θα ήταν πλήρης εάν δεν κάλυπτε ζητήματα που αφορούν την ασφάλεια / κρυπτογράφηση των δεδομένων που χρησιμοποιούνται, μαζί με παραδείγματα από τη χρήση του Arduino ως Ethernet client και επικοινωνίας με συσκευές Android.

Μικροελεγκτές

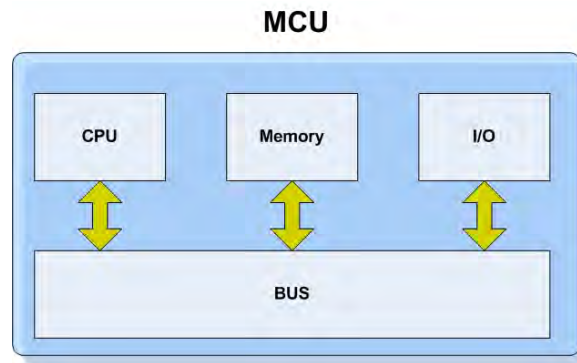
Ο μικροελεγκτής (ή MCU) μπορεί να θεωρηθεί ως ένας μικρός υπολογιστής σε ένα ενιαίο ολοκληρωμένο κύκλωμα (μπορείτε να το κατανοήσετε καλύτερα ως ένα «μικροτσίπ») που περιέχει έναν πυρήνα επεξεργαστή, τη μνήμη, και την προγραμματιζόμενη είσοδο και έξοδο για τα περιφερειακά. Όπως συζητήσαμε στο προηγούμενο κεφάλαιο, ως τυπικές συσκευές εισόδου – εξόδου νοούνται οι διακόπτες, τα ρελέ, τα πηνία, τα LED, οι μικρές ή προσαρμοσμένες οθόνες LCD, οι συσκευές ραδιοσυχνοτήτων, και οι αισθητήρες για δεδομένα όπως θερμοκρασία, υγρασία, φως, κλπ. Η μνήμη του προγράμματος (σε μορφή flash) συχνά περιλαμβάνεται στο τσιπ, καθώς και ένα, μικρό συνήθως, ποσό μνήμης RAM.

Η πλειοψηφία των μικροελεγκτών που χρησιμοποιούνται σήμερα είναι ενσωματωμένοι σε διάφορα είδη μηχανημάτων και συσκευών, όπως τα αυτοκίνητα, τα τηλέφωνα, οι συσκευές και τα περιφερειακά για τα συστήματα ηλεκτρονικών υπολογιστών. Αυτά ονομάζονται τα ενσωματωμένα συστήματα. Η Εικόνα 4.2 απεικονίζει ένα μικροελεγκτή, ο οποίος είναι επίσης ο ηλεκτρονικός εγκέφαλος του θέματος αυτού του κεφαλαίου, ο μικροελεγκτής Arduino. Οι μικροελεγκτές ως επί το πλείστον έχουν σχεδιαστεί έτσι ώστε να χρησιμοποιούνται σε ενσωματωμένες εφαρμογές, σε αντίθεση με τους μικροεπεξεργαστές που χρησιμοποιούνται σε προσωπικούς υπολογιστές.

Ο μικροελεγκτής είναι ένα πολύ κοινό συστατικό στα σύγχρονα ηλεκτρονικά συστήματα. Η χρήση του είναι τόσο διαδεδομένη που είναι σχεδόν αδύνατο να εργαστούμε πάνω σε ηλεκτρονικά χωρίς να τον συναντήσουμε. Τα βασικά στοιχεία μιας μονάδας μικροελεγκτών απεικονίζονται στην Εικόνα 4.1:

- I/O (Input/Output) θύρες. Είναι συνήθως τα pins, που συλλέγουν και παραδίδουν τα ψηφιακά σήματα σε άλλα κυκλώματα. Δηλαδή είναι οι διασυνδέσεις του μικροελεγκτή με τον εξωτερικό κόσμο. Οι αισθητήρες και οι ενεργοποιητές (actuators), καθώς και άλλες συσκευές που επικοινωνούν με τον μικροελεγκτή είναι συνδεδεμένες με αυτές τις θύρες. Δεδομένου ότι μιλάμε για ψηφιακές θύρες (αυτός είναι και ο λόγος που μόνο ψηφιακά σήματα μπορούν να μεταδοθούν) ο μικροελεγκτής λαμβάνει τις πληροφορίες σε μια σειρά από bit ή bytes.
- Η CPU: Η κεντρική μονάδα επεξεργασίας στην οποία γίνονται όλοι οι υπολογισμοί βασίζεται στον υπολογισμό και την αλληλεπίδραση με τα εξωτερικά κυκλώματα.
- Η μνήμη: Περιλαμβάνει το πρόγραμμα το οποίο εκτελείται και είναι επίσης διαθέσιμη για αποθήκευση των πληροφοριών που παράγονται από το πρόγραμμα.

- Μια σειριακή γραμμή από τον μικροεπεξεργαστή (Transmit ή TX) και μία σειριακή γραμμή στο μικροεπεξεργαστή (Receive ή RX) επιτρέποντας στα σειριακά δεδομένα που είναι σε μορφή Stream να μεταδοθούν ή να ληφθούν μέσω ενός interface δυο καλωδίων.



Εικόνα 4.1: Μια τυπική μονάδα μικροελεγκτή (MCU). Τα βασικά συστατικά είναι η CPU, η μνήμη και οι θύρες I / O.



Εικόνα 4.2: Ένα chip μικροελεγκτή. Η επικοινωνία με την πλακέτα και τα εξαρτήματα του κυκλώματος καθώς και η τροφοδοσία γίνεται μέσω των pins. Εδώ βλέπετε ένα AT Mega328-40- pin chip.

Οι περισσότεροι μικροελεγκτές ενσωματώνουν συσκευές όπως:

- Ένα χρονοδιακόπτη (Timer module) για να επιτρέπει στο μικροελεγκτή την εκτέλεση εργασιών για συγκεκριμένες χρονικές περιόδους.
- Έναν ADC (Analog to Digital Converter) ή μετατροπέα που επιτρέπει στον μικροελεγκτή να δέχεται αναλογική είσοδο δεδομένων για επεξεργασία.

Τα πιο δημοφιλή είδη μικροελεγκτών είναι οι Parallax Propeller, Basic Stamp, PIC, οι επεξεργαστές ARM και οι Atmel AVR. Το Arduino χρησιμοποιεί τη σειρά Atmel ATmega και το πιο διαδεδομένο είναι το ATmega328 chip (βλέπε Εικόνα 4.2).

Προγραμματισμός Μικροελεγκτών

Η κύρια λειτουργία ενός μικροελεγκτή είναι στην πραγματικότητα αυτό που δηλώνει το όνομα του: να ελέγχει τις συσκευές που συνδέονται, μέσω των διεπαφών (interfaces) I/O. Για να αναθέσετε στον μικροελεγκτή τι θα ελέγχει και πως θα το ελέγχει θα πρέπει να τον προγραμματίσετε. Αρχικά οι μικροελεγκτές προγραμματιζόντουσαν μόνο με τη χρήση της γλώσσας assembly, αλλά πλέον μπορείτε να χρησιμοποιήσετε αρκετές γλώσσες υψηλού επιπέδου για τον προγραμματισμό τους. Οι γλώσσες που χρησιμοποιούνται σήμερα είναι είτε ειδικά σχεδιασμένες για μικροελεγκτές ή γίνεται χρήση εκδόσεων κάποιας γλώσσας όπως η C με κάποιες παραλλαγές. Οι κατασκευαστές των μικροελεγκτών διαθέτουν συχνά κάποια ελεύθερα εργαλεία έτσι ώστε να είναι ευκολότερη η χρήση του υλικού τους.

Στις περισσότερες περιπτώσεις θα πρέπει να έχετε συγκεκριμένο hardware, προκειμένου να προγραμματίσετε το μικροελεγκτή. Αυτό το hardware είναι γνωστό ως προγραμματιστής μικροελεγκτών (Microcontroller programmer). Τα κύρια θέματα που θα μπορούσε κανείς να αντιμετωπίσει με τον προγραμματισμό μικροελεγκτών είναι τα εξής: α) η ανάγκη για χρήση ειδικού hardware το οποίο συχνά δεν είναι σε οικονομικές τιμές, β) η δυσκολία στον προγραμματισμό λόγω των αρκετών ωρών που μπορεί να χρειαστούν ή λόγω των πάρα πολλών διασυνδέσεων μεταξύ του ελεγκτή και του προγραμματιστή.

Πολλά από αυτά τα προβλήματα αντιμετωπίζονται με τη χρήση Boot loader. Ο Boot loader είναι ένα μικρό πρόγραμμα που έχει φορτωθεί στην πλακέτα του μικροελεγκτή. Αυτό το κομμάτι του λογισμικού έχει προγραμματιστεί άπαξ στη μνήμη του προγράμματος χρησιμοποιώντας ένα απλό προγραμματιστή. Μετά από αυτό, ο μικροελεγκτής μπορεί να προγραμματιστεί χωρίς τη χρήση εξωτερικού προγραμματιστή. Όταν ο boot loader τοποθετηθεί στον μικροεπεξεργαστή, είναι έτσι προγραμματισμένος ώστε κάθε φορά που γίνεται reset να λειτουργεί σαν οποιοδήποτε συνηθισμένο πρόγραμμα. Αυτό που κάνει όμως είναι διαφορετικό από ένα συνηθισμένο πρόγραμμα. Πρώτα απ' όλα ανάλογα με τον τύπο του Boot loader, ξεκινάει περιμένοντας για εισερχόμενα bytes μέσω ενός ειδικού interface. Για παράδειγμα, ένας UART boot loader περιμένει τον UART buffer του μικροελεγκτή, ελέγχοντας για εισερχόμενα bytes. Όταν τα bytes αρχίζουν να φτάνουν ο boot loader τα αποθηκεύει στην μνήμη προγράμματος με τη σειρά που τα παρέλαβε και σε προκαθορισμένες θέσεις. Όταν όλα τα bytes έχουν ληφθεί, ο Boot loader εκτελεί ένα jump στην αρχή της μνήμης και στη συνέχεια το "κανονικό" πρόγραμμα αρχίζει να λειτουργεί.

Η πλατφόρμα Arduino προσφέρει κυρίως τρία πράγματα τα οποία έχουν απομακρύνει τελείως κάθε πολυπλοκότητα και έχουν κάνει τον προγραμματισμό

των μικροελεγκτών εύκολο και προσιτό ακόμη και σε άπειρους χρήστες: α) έναν open-source Bootloader, β) open-schematic boards που μπορούν να χρησιμοποιηθούν εκτός του πλαισίου για τον προγραμματισμό και τη δημιουργία συνηθισμένων προγραμμάτων, γ) την ανάπτυξη ενός περιβάλλοντος για τη συγγραφή προγραμμάτων και αποστολή τους στις πλακέτες. Επιπλέον, υπάρχει μια μεγάλη κοινότητα η οποία επεκτείνεται πίσω από το Arduino, που τον υποστηρίζει και συμβάλει στην εξέλιξη του. Στη συνέχεια θα δείτε περισσότερες λεπτομέρειες σχετικά με την πλατφόρμα Arduino.

Η πλατφόρμα Arduino

Σύμφωνα με τους δημιουργούς της, το Arduino είναι μία ανοιχτού λογισμικού πρωτότυπη ηλεκτρονική πλατφόρμα που βασίζεται στην ευελιξία και την ευκολία του να χρησιμοποιεί το software και το hardware. Μπορεί να χρησιμοποιηθεί για την ανάπτυξη συνηθισμένων ηλεκτρονικών project λαμβάνοντας δεδομένα από μια ποικιλία από διακόπτες και αισθητήρες και ελέγχει μία σειρά από outputs. (φώτα, σερβομηχανισμούς, κλπ.)

Με λίγα λόγια η πλακέτα Arduino μπορεί να είναι 'το μυαλό' σε κάθε σας εφαρμογή. Τα προγράμματα Arduino μπορούν να είναι αυτόνομα (μπορούν να εκτελεστούν μόνο στην πλακέτα), ή μπορούν να χρησιμοποιηθούν για την αποστολή ή λήψη πληροφοριών από τις εφαρμογές που εκτελούνται στον υπολογιστή σας (χρησιμοποιώντας για παράδειγμα Flash ή Java / Processing). Τις πλακέτες μπορείτε να τις συναρμολογήσετε μόνοι σας ή να τις αγοράσετε έτοιμες (προσυναρμολογημένες). Το software χρησιμοποιείτε για να προγραμματίσετε την πλακέτα είναι ανοιχτού λογισμικού (open-source) και μπορείτε να το βρείτε - κατεβάσετε δωρεάν.

Τα κύρια χαρακτηριστικά, που είναι και τα ισχυρά πλεονεκτήματα του Arduino είναι οι τα ακόλουθα:

- Είναι χαμηλού κόστους: οι πλακέτες Arduino είναι σχετικά φθηνές σε σύγκριση με άλλες πλατφόρμες μικροελεγκτών. Σχεδόν όλοι οι τύποι Arduino μπορούν να συναρμολογηθούν με το χέρι, ακόμα και τα προσυναρμολογημένα κοστίζουν λιγότερο από \$ 50
- Είναι Cross-πλατφόρμα: Το λογισμικό Arduino τρέχει σε λειτουργικά συστήματα Windows, Macintosh OSX και Linux.
- Απλό, σαφές προγραμματιστικό περιβάλλον - Το περιβάλλον προγραμματισμού του Arduino είναι εύκολο στη χρήση για αρχάριους, αλλά είναι και αρκετά ευέλικτο έτσι ώστε να μπορούν να επωφεληθούν και οι έμπειροι χρήστες .
- Open source και επεκτάσιμο software – Το software του Arduino διαθέτει επεκτάσεις για έμπειρους χρήστες. Η γλώσσα προγραμματισμού μπορεί να

επεκταθεί μέσω των βιβλιοθηκών της C + +, αυτοί που θέλουν να κατανοήσουν τις τεχνικές λεπτομέρειες μπορούν να ασχοληθούν με τον προγραμματισμό του AVR με τη χρήση της γλώσσας C . Επίσης μπορείτε εάν θέλετε να προσθέσετε κατευθείαν τον κώδικα AVR-C στα προγράμματα Arduino σας.

- Open source και επεκτάσιμο hardware – Το Arduino είναι βασισμένο στους μικροελεγκτές ATMEGA8 και ATMEGA168 της Atmel. Τα σχέδια για τα modules δημοσιεύθηκαν σύμφωνα με την άδεια Creative Commons, έτσι έμπειροι σχεδιαστές κυκλωμάτων μπορούν να κάνουν τη δική τους module εκδοχή, με δυνατότητα επέκτασης και βελτίωσης. Ακόμη και σχετικά άπειροι χρήστες μπορούν να δημιουργήσουν την έκδοση breadboard της πλακέτας προκειμένου να κατανοήσουν πώς λειτουργεί, ενώ είναι και ένας τρόπος για να εξοικονομήσουν χρήματα.

Το Arduino δεν είναι από μόνο του ένας μικροελεγκτής. Είναι μία πλατφόρμα open source για προτυποποίηση που αποτελείται από hardware πλακέτες που περιλαμβάνουν Μικροελεγκτές) και ένα προγραμματιστικό περιβάλλον.

Ένα άλλο μεγάλο πλεονέκτημα του Arduino είναι ότι μπορείτε να δοκιμάσετε, να χτίσετε και να αναπτύξετε ενσωματωμένα προγράμματα πολύ γρήγορα. Μπορείτε να χρησιμοποιήσετε μια πλακέτα Arduino για να φτάξετε ένα πρωτότυπο, να το δοκιμάσετε και στη συνέχεια απλά να αφαιρέσετε το chip από το κύκλωμα και να το χρησιμοποιήσετε στην δική σας πλακέτα έτσι ώστε να δημιουργήσετε τη δική σας ενσωματωμένη συσκευή. Μπορείτε επίσης να πάρετε ένα ATmega chip και να το προγραμματίσετε χρησιμοποιώντας μια πλακέτα Arduino. Σημειώστε ότι τα chip θα πρέπει να προγραμματιστεί με το Arduino Bootloader (ώστε να μπορέσει να χρησιμοποιηθεί με το Arduino IDE). Είναι επίσης δυνατό να προγραμματίσετε ένα chip χρησιμοποιώντας ένα δεύτερο Arduino

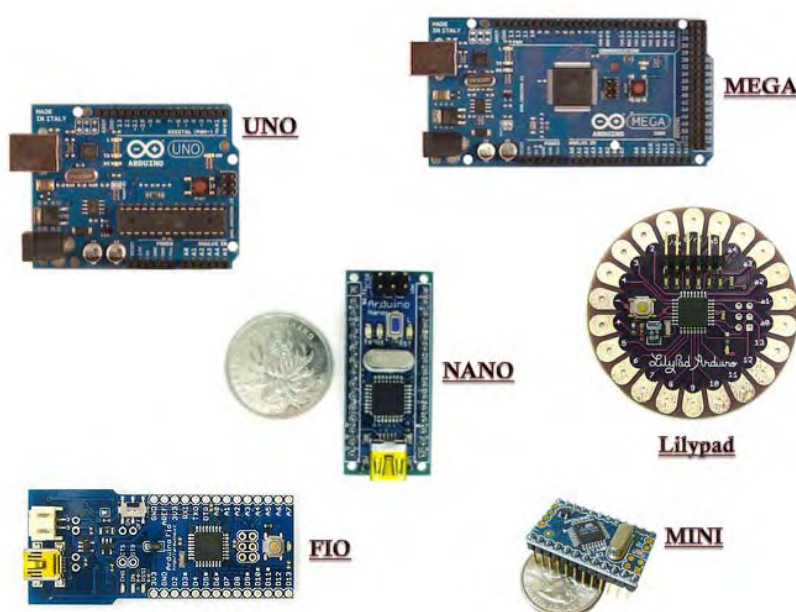
Οι πλακέτες –Boards

Το Arduino υπάρχει σε διάφορες παραλλαγές πλακέτας. Η επίσημη ομάδα του Arduino σχεδιάζει τα περισσότερα από αυτά. Μπορείτε επίσης να βρείτε διάφορους κλώνους από τις πιο κοινές πλακέτες (όπως η Uno ή η Mega) από άλλους προμηθευτές.

Μια πλακέτα Arduino αποτελείται από έναν 8-bit μικροελεγκτή AVR Atmel με όλα τα απαραίτητα συστατικά για να μπορείτε να προγραμματίσετε και να υπάρξει αλληλεπίδραση με τους αισθητήρες και άλλα κυκλώματα. Μια σημαντική πλευρά του Arduino είναι ο τρόπος που οι connectors είναι τοποθετημένοι, επιτρέποντας στη πλακέτα να συνδεθεί με μια ποικιλία των εναλλασσόμενων add-on modules (γνωστά ως shields). Τα Arduino έχουν χρησιμοποιήσει τα chips της

σειράς megaAVR, ειδικά τα ATmega8, ATmega168, ATmega328, ATmega1280, και ATmega2560. Οι περισσότερες πλακέτες-boards έχουν 5 Volt linear regulator και έναν κρυσταλλικό ταλαντωτή 16 MHz. Ο μικροελεγκτής ενός Arduino είναι επίσης προ-προγραμματισμένος με έναν φορτωτή εκκίνησης (boot loader) που απλοποιεί την μεταφορά των προγραμμάτων στην flash μνήμη του chip (on-chip flash memory), ενώ άλλες συσκευές συνήθως χρειάζονται έναν εξωτερικό προγραμματιστή chip.

Οι πλακέτες υπάρχουν σε διάφορες εκδόσεις οι οποίες διαφέρουν στο μέγεθος και στις δυνατότητες από την άποψη του επεξεργαστή και τον αριθμό των I/O των pins που μπορούν να χρησιμοποιηθούν.



Εικόνα 4.3: Μια απεικόνιση της πιο δημοφιλούς πλακέτας Arduino. Οι πλακέτες διαφέρουν στο μέγεθος και στις δυνατότητες.

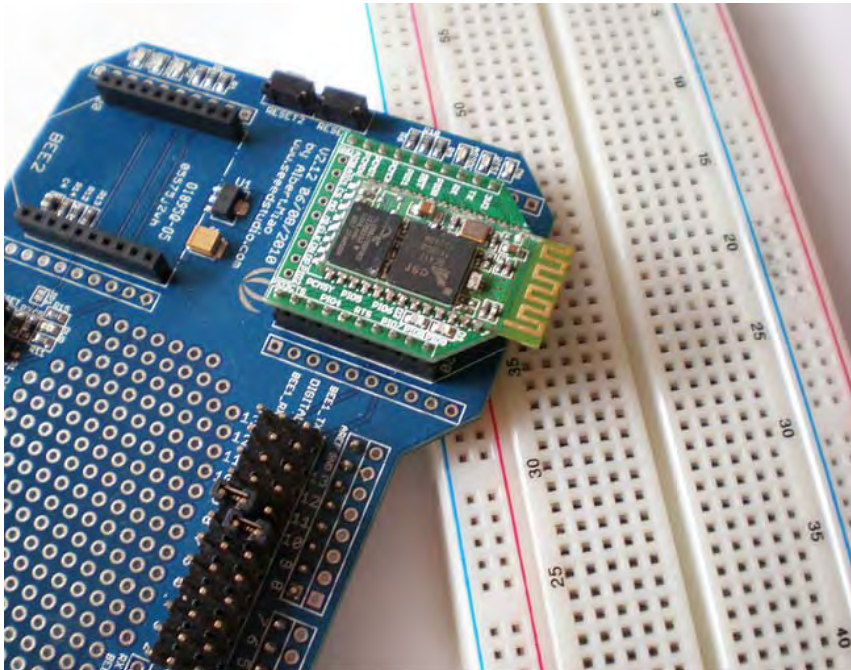
Μια σύντομη περιγραφή της κάθε πλακέτας:

- **Uno:** Βασίζεται στο chip ATmega328, έχει 14 ψηφιακά pin εισόδου-εξόδου, 6 αναλογικές εισόδους, έναν κρυσταλλικό ταλαντωτή 16 MHz, μία σύνδεση USB, μία υποδοχή ρεύματος, μία κεφαλίδα ICSP και ένα κουμπί επαναφοράς-επανεκίνησης (reset). Η πλακέτα έρχεται έτοιμη για χρήση, πράγμα που σημαίνει ότι περιέχει όλα όσα χρειάζονται για να την προγραμματίσετε και να τη χρησιμοποιήσετε. Το Arduino Uno μπορεί να τροφοδοτείται μέσω της σύνδεσης USB ή με εξωτερικό τροφοδοτικό (Η επιλογή της πηγής ενέργειας-ισχύος γίνεται αυτόματα).
- **Mega:** Βασίζεται στον ATmega2560 και διαθέτει 54 ψηφιακά pin εισόδου/εξόδου, 16 αναλογικές εισόδους, 4 UARTs (σειριακές θύρες), ένα

κρυσταλλικό ταλαντωτή 16 MHz, μια σύνδεση USB, υποδοχή ρεύματος, μία κεφαλίδα ICSP και ένα κουμπί επαναφοράς-επανεκίνησης (reset). Θεωρείται ως μια πιο προηγμένη έκδοση του Uno, χρησιμοποιείται κυρίως για μεγαλύτερο αριθμό I / O. Είναι παρόμοια με την Uno και μπορεί να χρησιμοποιηθεί κατευθείαν... (out-of-the box).

- **Nano:** Είναι μία μικρή αλλά ολοκληρωμένη πλακέτα, βασίζεται στον ATmega328 (Arduino Nano 3, 0) ή στον ATmega168 (Arduino Nano 2.x). Έχει λίγο πολύ την ίδια λειτουργικότητα με το Arduino Uno, αλλά σε διαφορετικό πακέτο. Στερείται μόνο μια υποδοχή ρεύματος DC και συνδέεται με ένα καλώδιο USB αντί για το συνηθισμένο. Θεωρείται επίσης εύκολα επεκτάσιμη λόγω του μεγέθους της και της διάταξης των pin που επιτρέπουν να μπουν σε μια πλακέτα breadboard.
- **Lilypad:** Αυτή η πλακέτα έχει σχεδιαστεί για εφαρμογές που φοριούνται επειδή μπορεί να ραφτεί στο ύφασμα και μπορεί να συνδεθεί με τις υπόλοιπες μονάδες με αγωγική ραφή. Βασίζεται στον ATmega168V (η έκδοση χαμηλής ισχύος του ATMEGA168) ή στον ATmega328V. Παρέχει τα ίδια pin I/O με την UNO αλλά χρειάζεται επιπλέον hardware για τον προγραμματισμό (FTDI USB adapter).
- **Mini:** Είναι η μικρότερη έκδοση της πλακέτα Arduino που μπορεί κανείς να αγοράσει. Όμως έχει όλες τις λειτουργίες της Uno (όσον αφορά τη μνήμη και τα pin I/O) και μπορεί να έρθει με το ίδιο chip μικροελεγκτή. Όπως και η Lilypad έτσι και η Mini χρειάζεται ένα εξωτερικό προσαρμογέα για να την προγραμματίσετε και μπορεί να τοποθετηθεί σε ένα breadboard, όπως η Nano.
- **Fio:** Αυτή είναι μία από τις καλύτερες επιλογές για εφαρμογές που απαιτούν σύνδεση στο internet. Βασίζεται στο μικροελεγκτή ATmega328P, λειτουργεί σε 3.3V και 8 MHz. Έχει 14 ψηφιακά pin εισόδου-εξόδου, 8 αναλογικές εισόδους, ένα on-board ταλαντωτή, ένα κουμπί επαναφοράς - επανεκίνησης (reset) και τρύπες για την τοποθέτηση των pin headers. Διαθέτει συνδέσεις για μπαταρία Lithium Polymer (LiPo) και περιλαμβάνει το δικό της κύκλωμα τροφοδοσίας μέσω USB. Είναι κατάλληλο για ασύρματες εφαρμογές, δεδομένου ότι περιέχει υποδοχή για Xbee.
- Η Xbee είναι μια ειδικά σχεδιασμένη υποδοχή που δέχεται 'Bee connection modules' (Εικόνα 4.4). Αυτά είναι μονάδες επικοινωνίας που έχουν σχεδιαστεί ως επεκτάσεις στις πλακέτες Arduino. Έχουν ένα συγκεκριμένο μέγεθος και τη διάταξη και τροφοδοτούνται απευθείας από την πλακέτα Arduino μέσω της υποδοχής. Η υποδοχή παρέχει επίσης άμεση επικοινωνία με τους ακροδέκτες RX/TX του Arduino. Η Xbee αρχικά παρουσιάστηκε από την Digi ως μονάδα ασύρματης επικοινωνίας για την υποστήριξη ZigBee. Εάν έχετε μία Bee Shield (ή Fio) μπορείτε να κάνετε ενσωμάτωση με μια μονάδα Bee και να αρχίσετε να εργάζεστε χωρίς να ανησυχείτε για τη σύνδεση και την τροφοδοσία. Υπάρχουν διαθέσιμες μονάδες-Modules Bee που παρέχουν

ZigBee, Bluetooth and WiFi συνδεσιμότητα. Μια σημαντική πτυχή του Arduino και των FiO Xbee modules είναι ότι χρησιμοποιώντας έναν τροποποιημένο για USB-to-Xbee προσαρμογέα (όπως είναι ο Xbee Explorer USB), μπορείτε να ανεβάσετε τον κώδικα ασύρματα και να προγραμματίσετε το Arduino σας από απόσταση!



Εικόνα 4.4: Ένα Bluetooth Bee module σε ένα XBee shield με δύο Bee sockets

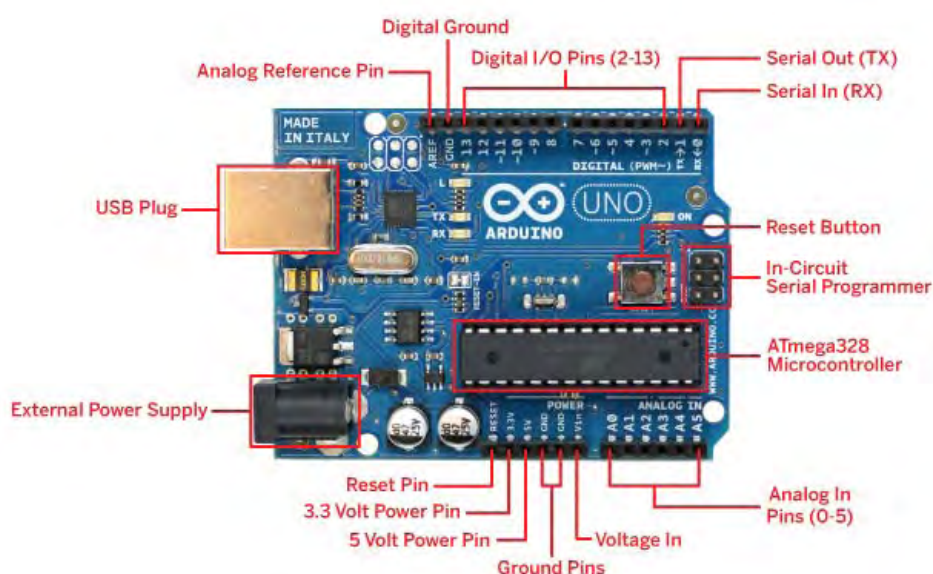
Άλλες πλακέτες Arduino είναι: το Arduino Mega ADK (είναι παρόμοια με τη Mega, αλλά με μια πρόσθετη υποδοχή USB για να συνδέετε με Android τηλέφωνα), το Arduino BT (που είναι παρόμοια με το Uno, αλλά με μια ενσωματωμένη μονάδα Bluetooth) και το Arduino Ethernet (είναι παρόμοια με το Uno, αλλά με μια ενσωματωμένη θύρα Ethernet. Επίσης είναι απαραίτητη μια εξωτερική μονάδα USB για τον προγραμματισμό της, όπως η Nano). Είναι σαφές ότι η ομάδα του Arduino και οι υπόλοιποι προμηθευτές-κατασκευαστές Arduino κινούνται έτσι ώστε να παρέχουν δυνατότητες δικτύου στις πλακέτες για να διευκολυνθεί η σύνδεση με το Διαδίκτυο και την επικοινωνία με τα συστήματα Cloud!

Αρχιτεκτονική της πλακέτας Arduino.

Όπως αναφέρθηκε προηγουμένως η πλακέτα Arduino αποτελείται από έναν μικροελεγκτή AVR (το κύριο τμήμα της πλακέτας) συνήθως ένα ATmega328, κατάλληλα ηλεκτρονικά για την λειτουργία και πρόσθετα στοιχεία που σας

επιτρέπουν να προγραμματίσετε και να υπάρχει επικοινωνία με τους αισθητήρες και άλλα κυκλώματα, όπως ενότητες-module επικοινωνίας.

Η Εικόνα 4.5 παρουσιάζει μία πλακέτα Arduino Uno (όπως φαίνεται από πάνω) με σχολιασμό των βασικών συστατικών της πλακέτας. Μπορείτε να δείτε μια εκδοχή από 40-pin τοποθετημένα σε ένα τσιπ υποδοχής κοντά στο κέντρο της πλακέτας, την υποδοχή ρεύματος, τη θύρα USB, το κουμπί επαναφοράς, το σειριακό προγραμματιστή και δύο σειρές pin I / O στο επάνω και το κάτω μέρος της πλακέτας. Το κουμπί επαναφοράς (Reset Button) επαναφέρει το κύκλωμα και ξεκινάει τη λειτουργία από την αρχή. Ο σειριακός προγραμματιστής μπορεί να χρησιμοποιηθεί σε περίπτωση που επιθυμείτε να προγραμματίσετε το κύκλωμα έναν εξωτερικό προγραμματιστή (ο οποίος θα σας επιτρέψει να τροποποιήσετε / ενημερώσετε τον Bootloader).



Εικόνα 4.5: Η ανατομία μιας πλακέτας Arduino. Μπορείτε να δείτε μεταξύ των άλλων τα κύρια pins για τη στερέωση των αναλογικών και ψηφιακών αισθητήρων, την τροφοδοσία και τα pin γείωσης, το τροφοδοτικό και το μικροελεγκτή ATmega328.

Μπορείτε να τροφοδοτήσετε το Arduino μέσω της σύνδεσης USB ή χρησιμοποιώντας ένα εξωτερικό τροφοδοτικό. Η πηγή τροφοδοσίας επιλέγεται αυτόματα. Στην περίπτωση της εξωτερικής πηγής τροφοδοσίας θα πρέπει να συνδέσετε έναν προσαρμογέα, βάζοντας ένα βύσμα 2.1 mm στην υποδοχή της πλακέτας. Εναλλακτικά οι πόλοι της μπαταρίας μπορούν να συνδεθούν στα Gnd και Vin pin header της πλακέτας. Η πλακέτα μπορεί να λειτουργήσει με εξωτερική παροχή από 6 έως 20 Volt. Εάν παρέχεται λιγότερο από 7V, το 5 Volt pin θα δώσει λιγότερο από 5 Volt και το κύκλωμα που θα συνδεθεί εκεί μπορεί να είναι ασταθές, οπότε το εύρος της συνιστώμενης τάσης είναι μεταξύ 7 και 12 βολτ. Στο κάτω δεξιά

μέρος της πλακέτας θα παρατηρήσετε τα αναλογικά pin εισόδου (0- 5). Αυτά είναι τα pins που μπορείτε να χρησιμοποιήσετε για να συνδέσετε αναλογικούς αισθητήρες και να διαβάσετε τις εισόδους τους απευθείας από το σχεδιάγραμμα του Arduino σας. Στην αριστερή πλευρά υπάρχει άλλη μία σειρά από 6 pin. Το pin επανεκκίνησης (reset pin) είναι μια ψηφιακή θύρα εισόδου. Μπορείτε για παράδειγμα να ελέγξετε ένα ενεργοποιητή και όταν θέσετε υψηλή τάση (π.χ. να περνάνε 5V) το pin θα ενεργοποιήσει την επαναφορά της πλακέτας. Τα δύο επόμενα pin παράγουν 3.3V και 5V αντίστοιχα και μπορούν να χρησιμοποιηθούν έτσι ώστε να μπορούν να τροφοδοτήσουν τους αισθητήρες και / ή άλλα εξαρτήματα της πλακέτας. Δίπλα τους υπάρχουν δύο GND pins (pin γείωσης) και μία τάση στο 5V. Όπως αναφέρθηκε προηγουμένως, μπορείτε να χρησιμοποιήσετε ένα ζευγάρι από το Ground pin (pin γείωσης) και το Voltage in pin για να τροφοδοτήσετε την πλακέτα σας μέσω μιας μπαταρίας.

Το πάνω μέρος της πλακέτας περιέχει τα ψηφιακά pin I / O και τα σειριακά pin επικοινωνίας. Το τελευταίο είναι το ζευγάρι που σημειώνεται ως Rx και Tx στην πλακέτα. Μπορείτε να χρησιμοποιήσετε αυτά τα pin για την επικοινωνία με τα modules που χρησιμοποιούν σειριακό πρωτόκολλο όπως Bluetooth modems, LCDs, κ.α. Τα περισσότερα shields που παρέχουν δυνατότητες επικοινωνίας στο Arduino σας χρησιμοποιούν τα τελευταία pin. Προχωρώντας προς τα αριστερά, υπάρχουν 12 ψηφιακά pin I / O. Αυτά μπορούν να χρησιμοποιηθούν για να διαβαστούν ψηφιακά (ακολουθίες από HIGH/LOW καταστάσεις) δεδομένα από εξαρτήματα όπως switches ή ψηφιακούς αισθητήρες ή για τον έλεγχο συσκευών όπως ένα διακοπτικό ρελέ. Η κατάσταση λειτουργίας τους (είσοδος / έξοδος) έχει οριστεί στον κώδικα. Επίσης υπάρχει ένα άλλο pin γείωσης (πολύ χρήσιμο εάν θέλετε να συνδέσετε κάτι απευθείας, όπως ένα led σε ένα ψηφιακό pin) και τέλος το αναλογικό pin αναφοράς. Το τελευταίο χρησιμοποιείται σε περιπτώσεις που πρέπει να έχετε μια τάση αναφοράς στο κύκλωμα σας, δηλαδή μία τάση που είστε βέβαιοι για το ποία είναι η τιμή της και δεν είναι από προεπιλογή 3.3V ή 5V. Η τάση του αναλογικού pin αναφοράς καθορίζεται από τον κώδικα σας. Οι διαθέσιμες επιλογές είναι οι εξής:

- DEFAULT: Η Default αναφορική τιμή 5V ή 3.3V ανάλογα με τον τύπο της πλακέτας (πχ για Uno είναι 5V)
- INTERNAL: ενσωματωμένη τιμή αναφοράς, ίση με 1.1V στον ATMEGA168 ή ATmega328 και 2,56 Volt στον ATMEGA8 (δεν διατίθεται για το Arduino Mega)
- INTERNAL1V1: μια ενσωματωμένη τιμή αναφοράς, ίση με 1.1V (διατίθεται μόνο με το Arduino Aa)
- INTERNAL2V56: μία ενσωματωμένη τιμή αναφοράς, ίση με 2.56V 1V (διατίθεται μόνο με το Arduino Aa)
- EXTERNAL: μία συνηθισμένη τάση που εφαρμόζεται στο AREF pin (μεταξύ 0 και 5V μόνο)

Το IDE του Arduino είναι γραμμένο σε Java και μπορεί να τρέξει σε πολλαπλές πλατφόρμες. Το περιβάλλον ανάπτυξης είναι βασισμένο στην Processing, ένα περιβάλλον ανάπτυξης σχεδιασμένο να εισαγάγει στον προγραμματισμό άτομα μη εξοικειωμένα με την ανάπτυξη λογισμικού. Η συγκεκριμένη γλώσσα προγραμματισμού προέρχεται από την Wiring. Περιλαμβάνει επεξεργαστή κώδικα με χαρακτηριστικά όπως είναι η επισήμανση σύνταξης και έχει επίσης τη δυνατότητα συλλογής και φόρτωσης προγραμμάτων στην πλακέτα με ένα μόνο κλικ (βλέπε. Σχήμα 4.6). Συνήθως δεν χρειάζεται να επεξεργαστείτε makefiles ή να τρέξετε προγράμματα από την γραμμή εντολών. Δεδομένου ότι είναι βασισμένο σε Java μπορεί να εγκατασταθεί και να εκτελεστεί σε κάθε υπολογιστικό περιβάλλον Windows, MacOSX και Linux.

Το Arduino IDE προσφέρει κυρίως τα ακόλουθα χαρακτηριστικά για τους χρήστες:

- Διαχείριση των προγραμμάτων (που ονομάζονται sketches): Επεξεργασία νέων προγραμμάτων στον επεξεργαστή κώδικα, φόρτωση και αποθήκευση αυτών.
- Επεξεργασία των sketches και εμφάνιση μηνύματος για οποιαδήποτε επιτυχία ή μηνύματα λάθους.
- Φόρτωση των sketches του χρήστη στην πλακέτα Arduino.
- Το Serial Monitor: Ένα γραφικό Interface που εμφανίζει όλες τις πληροφορίες που προέρχονται από τη σειριακή επικοινωνία με την πλακέτα Arduino και που επίσης επιτρέπει στους χρήστες να στέλνουν πληροφορίες πίσω στο Arduino.

Το IDE έρχεται επίσης με προεγκατεστημένο παραδείγματα, έτσι ώστε ο χρήστης να τα φορτώσει και να τα στείλει κατευθείαν σε μια πλακέτα Arduino. Αυτά τα παραδείγματα περιλαμβάνουν τις βασικές λειτουργίες του Arduino.

Εγκατάσταση-Ρύθμιση του IDE

Για την πιο πρόσφατη IDE μεταβείτε στη σελίδα λήψης του Arduino <http://arduino.cc/en/Main/Software> και κατεβάστε την κατάλληλη έκδοση ανάλογα με το λειτουργικό σας σύστημα.

Εγκατάσταση σε Windows XP

Αφού έχετε κατεβάσει την πιο πρόσφατη IDE, αποσυμπίεστε το αρχείο και ανοίξτε το. Θα δείτε τα αρχεία Arduino και κάποιους υπο-φακέλους. Στη συνέχεια, συνδέστε το Arduino σας χρησιμοποιώντας το καλώδιο USB και εξασφαλίστε ότι η πράσινη λυχνία (ένδειξη PWR) ανάβει. Σύμφωνα με την πλακέτα που χρησιμοποιείτε θα εμφανιστεί ένα παράθυρο που θα αναφέρει ότι : “Εντοπίστηκε

νέο υλικό: Arduino Uno - Found new hardware: Arduino Uno” και θα εμφανιστεί ο οδηγός “found new hardware”. Πατήστε next και τα windows θα προσπαθήσουν να φορτώσουν τους drivers. Αυτή η διαδικασία θα αποτύχει καθότι δεν υπάρχουν usb drivers της πλακέτας διαθέσιμοι στα Windows. Έπειτα κάντε δεξί κλικ στο εικονίδιο “My Computer” της επιφάνειας εργασίας και επιλέγεται “Manage”, το παράθυρο του “Computer Management” θα ανοίξει, στη συνέχεια κάνετε κλικ στο “Event Manager” από τη λίστα “System Tools” στην αριστερή πλευρά του παραθύρου θα δείτε μία λίστα από τις συσκευές σας. Το Arduino Uno θα εμφανιστεί στη λίστα, όπως μια συσκευή με ένα κίτρινο θαυμαστικό για να δείξει ότι η συσκευή δεν έχει εγκατασταθεί σωστά. Πατήστε δεξί κλικ και επιλέξτε “Update Driver”. Επιλέξτε “ No not this Time” από την πρώτη σελίδα και πατήστε “next”.. Στη συνέχεια επιλέξτε “ Install from a list or specific location (Advanced) “ και πατήστε ξανά “next”. Τώρα κάντε κλικ στο “ Include this location in the search” και πατήστε “Browse”. Μεταβείτε στο φάκελο Drivers από το αποσυμπίεσμένο Arduino IDE και πατήστε “next”. Τα windows θα εγκαταστήσουν τους drivers, οπότε μπορείτε να πατήσετε το κουμπί “Finish”. Το Arduino Uno θα εμφανιστεί στην “Device List” δείχνοντας το port το οποίο χρησιμοποιεί (πχ COM6). Για να ανοίξετε την IDE απλά πατήστε διπλό κλικ στο εικονίδιο “Arduino”

Εγκατάσταση σε Windows 7

Αφού έχετε κατεβάσει την πιο πρόσφατη IDE, αποσυμπίεστε το αρχείο και ανοίξετε το. Θα δείτε τα αρχεία Arduino και κάποιους υπο-φακέλους. Στη συνέχεια, συνδέστε το Arduino σας χρησιμοποιώντας το καλώδιο USB και εξασφαλίστε ότι η πράσινη λυχνία (ένδειξη PWR) ανάβει. Τα Windows θα προσπαθήσουν να εγκαταστήσουν αυτόματα τους drivers για το Arduino Uno και θα αποτύχουν. Κάντε κλικ στο κουμπί Έναρξη των Windows και στη συνέχεια κάντε κλικ στον Πίνακα Ελέγχου(Control Panel). Τώρα κάντε κλικ στο Σύστημα και Ασφάλεια (System and Security), τότε κάντε κλικ στο Σύστημα (System) και στη συνέχεια κάντε κλικ στη Διαχείριση Συσκευών (Device Manager) από τη λίστα στην αριστερή πλευρά. Το Arduino θα εμφανιστεί στη λίστα, όπως μια συσκευή με ένα κίτρινο θαυμαστικό για να δείξει ότι η συσκευή δεν έχει εγκατασταθεί σωστά. Κάντε δεξί κλικ στο Arduino Uno και επιλέξτε : “Ενημέρωση προγράμματος οδήγησης (Update Driver Software)”.

Στη συνέχεια, επιλέξτε “Αναζήτηση για λογισμικό προγράμματος οδήγησης (Browse my computer for driver software)” και στο επόμενο παράθυρο κάντε κλικ στο κουμπί Αναζήτηση (Browse). Κάντε κλικ στο κουμπί “ Install this driver software anyway”. Το παράθυρο “Installing Driver Software” θα κάνει τώρα τη δουλειά του. Αν όλα πάνε καλά, θα εμφανιστεί ένα άλλο παράθυρο που θα λέει: “Windows has successfully updated your driver software”. Τελειώνοντας πατήστε το “Close”. Για να ανοίξετε την IDE απλά πατήστε διπλό κλικ στο εικονίδιο “Arduino”.

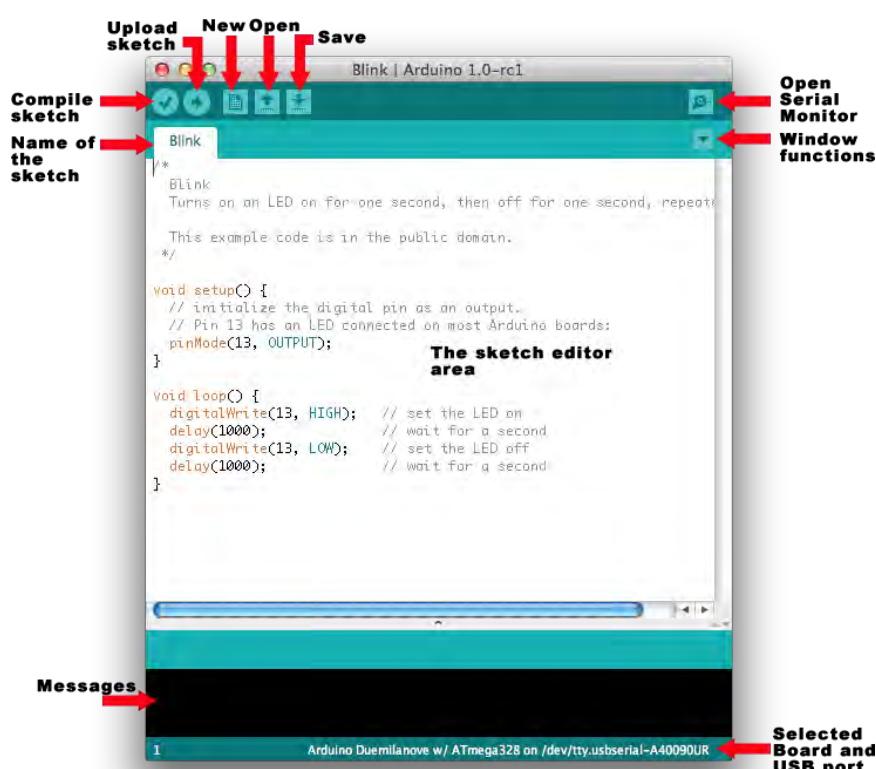
Εγκατάσταση σε MacOSX

Κατεβάστε το τελευταίο disk image αρχείο (.dmg) για το IDE. Ανοίξτε το αρχείο αυτό και κάντε drag and drop Το εικονίδιο Arduino στον φάκελο

“Applications”. Εάν χρησιμοποιείτε παλιότερο Arduino όπως ο Duemilanove, θα χρειαστεί να εγκαταστήσετε το FTDI USB Serial Driver. Κάντε διπλό κλικ στο εικονίδιο του πακέτου και ακολουθήστε τις οδηγίες για να το κάνετε αυτό. Για τον Uno και τον Mega2560 δεν απαιτείται εγκατάσταση κανενός driver.

Γράφοντας κώδικα Arduino

Πλέον έχετε δημιουργήσει το περιβάλλον ανάπτυξης Arduino (IDE) και είστε έτοιμοι να γράψετε το πρώτο σας πρόγραμμα. Το Arduino IDE έρχεται μία βιβλιοθήκη C / C ++ που ονομάζεται “Wiring”, η οποία εκτελεί πολλές κοινές λειτουργίες input/output αρκετά πιο εύκολα. Τα προγράμματα Arduino είναι γραμμένα σε C / C ++, αν και πραγματικά δεν χρειάζεται να είστε πολύ εξοικειωμένοι με το τελευταίο. Θα πρέπει όμως να γνωρίζετε κάποιες γενικές αρχές προγραμματισμού (όπως μεταβλητές, καταστάσεις ελέγχου, βρόχους, κ.λπ.) και πως λειτουργούν οι συναρτήσεις (functions) και οι μέθοδοι (methods). Επίσης παρέχονται πολλά από τα προεγκατεστημένα παράδειγμα από κώδικες (sketches) Arduino. Να σας παρουσιάσουμε επίσης την έννοια του προγραμματισμού Arduino. Ας δούμε πρώτα τα βασικά ενός κώδικα (Sketch) Arduino και στη συνέχεια να προχωρήσουμε σε μερικά παραδείγματα προγραμματισμού.



Σχήμα 4.6: Το Arduino περιβάλλον IDE, όπου μπορείτε να επεξεργαστείτε τον κώδικα (sketch) σας και το φορτώσετε στην πλακέτα Arduino σας. Μπορείτε επίσης να δείτε τα κουμπιά για τις βασικές λειτουργίες.

Ο κώδικας (sketch) Arduino

Όταν ανοίγετε το Arduino IDE, θα παρατηρήσετε την κύρια λευκή περιοχή που περιμένει από εσάς κάποια κώδικα. Αυτός είναι ο κύριος χώρος που το κείμενο του Arduino θα βρίσκεται όταν το επεξεργάζεστε, όταν το ανοίγεται από ένα source file και όταν το ανεβάζετε στην πλακέτα σας.

Ξεκινώντας με τη δομή Arduino sketch, θα πρέπει να έχετε κατά νου ότι ένα sketch έχει δύο υποχρεωτικά στοιχεία:

- Η συνάρτηση ρύθμισης (**setup**): Αυτή προσδιορίζεται στο sketch ως void setup(). Όπως υποδηλώνει το όνομα κατά πάσα πιθανότητα, θα είναι η κύρια συνάρτηση αρχικοποίησης. Ό, τι χρειάζεται να οριστεί ή αρχικοποιηθεί με μια τιμή όταν το sketch σας θα αρχίσει να λειτουργεί με το Arduino σας θα πρέπει να δηλωθεί εκεί. Η συνάρτηση ρύθμισης εκτελείται μία φορά και έτσι ο κώδικας εκτελείται μόνο μία φορά (εκτός και αν έχετε ορίσει κάποια βρόχους επανάληψης) και με την ίδια σειρά που το έχετε γράψει. Συνήθως θα βρείτε εδώ την αρχικοποίηση των μεταβλητών σας και/ή τις προσαρμοσμένες ρυθμίσεις του συστήματος (όπως την αρχικοποίηση και τη ρύθμιση του baud rate για το Serial Monitor), η αρχικοποίηση των pins της πλακέτας, ως pin εισόδου ή εξόδου, κλπ.
- Μπορείτε να καλέσετε εξωτερικές συναρτήσεις από τη συνάρτηση setup(). Μπορείτε επίσης να αναφερθείτε σε εξωτερικές μεταβλητές (όταν αυτές δηλώνονται ως γενικές, στην αρχή του κώδικα σας).
- Η συνάρτηση **loop (βρόχος)**: Αυτή προσδιορίζεται στο sketch ως void loop(). Αυτή η συνάρτηση εκτελείται αυτόματα (δεν χρειάζεται να κάνετε εσείς κάτι) μετά την εκτέλεση της συνάρτησης setup(). Οτιδήποτε περιέχετε σε αυτή τη συνάρτηση, θα πρέπει να εκτελείτε κατ'επανάληψη (σαν να ήταν σε κάποιο while(true) {...} loop).
- Το γεγονός αυτό είναι πολύ λογικό από τη στιγμή που πιθανότατα θα χρησιμοποιήσετε το Arduino σας για να παρακολουθεί κάποιες εισόδους ή εξόδους και για συνεχή αποστολή δεδομένων μέσω του διαδικτύου. Σε περίπτωση που χρειάζεστε να σταματήσει η εκτέλεση ολόκληρου του loop (βρόχου) ή ένα μέρος από αυτό θα πρέπει να προστεθεί το εν λόγω τμήμα του κώδικα μέσα σε μια δήλωση ελέγχου (if...then).
- Επιπλέον, θα διαπιστώσετε ότι στις περισσότερες περιπτώσεις υπάρχει μια καθυστέρηση στην εν λόγω συνάρτηση. Μπορεί να είναι οπουδήποτε μέσα στον κώδικα loop (ανάλογα με τις ανάγκες του project σας) και συνήθως οι περισσότεροι θα το βρείτε στο τέλος της συνάρτησης loop. Αυτή η καθυστέρηση (συνήθως ορίζεται σε msec) δίνει τη δυνατότητα στο board να κρατήσει την εκτέλεση του προγράμματος, πριν αυτό μεταβεί και πάλι στην αρχή του κώδικα loop ή στο επόμενο τμήμα κώδικα.

Εκτός από το τελευταίο μπορείτε να βρείτε στο sketch σας πρόσθετα στοιχεία κώδικα, όπως μεταβλητές, αναφορά σε εξωτερικές βιβλιοθήκες, προσαρμοσμένες συναρτήσεις, κλπ.

Ας δούμε μερικά βασικά παραδείγματα που θα σας βοηθήσουν να κατανοήσετε περισσότερο την κωδικοποίηση των Arduino sketches.

Μερικά Βασικά Παραδείγματα

Ας συζητήσουμε μερικά παραδείγματα κώδικα που θα σας βοηθήσουν να καταλάβετε τα βασικά χαρακτηριστικά του προγραμματισμού του Arduino.

Διασύνδεση (Interfacing) με το Serial Monitor

Πρώτα ξεκινάμε με το Serial Monitor. Μπορεί να θεωρηθεί ως το πιο σημαντικό χαρακτηριστικό του IDE, δεδομένου ότι επιτρέπει σε σας για να δείτε κάποια έξοδο κειμένου που δημιουργεί το Arduino σας με τη βοήθεια του κώδικά σας, το οποίο μπορεί να είναι πολύ χρήσιμο ειδικά όταν προσπαθείτε να διορθώσετε τα προγράμματά σας.

Το Serial Monitor (ως μέρος του IDE) είναι στην πραγματικότητα μια περιοχή όπου κάθε μήνυμα που μεταδίδεται από τον μικροελεγκτή σας εμφανίζεται στην οθόνη. Ομοίως, μπορείτε να στέλνετε μηνύματα κειμένου στην πλακέτα Arduino σας χρησιμοποιώντας το Serial Monitor. Και στις δύο περιπτώσεις, πρέπει να ορίσετε την επικοινωνία στο sketch σας.

Για να είστε σε θέση να χρησιμοποιήσετε το Serial Monitor πρέπει να έχετε δύο πράγματα στο sketch σας: α) Την εκκίνηση του Serial object με ένα συγκεκριμένο baud rate επικοινωνίας (που ονομάζεται baud rate), β) τη χρήση των βασικών συναρτήσεων Serial object που θα σας δώσει τη δυνατότητα είτε να εκτυπώσετε τα δεδομένα ή να τα διαβάσετε. Ας δούμε τις λεπτομέρειες στο παρακάτω κώδικα:

Υπόδειγμα 4.1: Απλό παράδειγμα χρήσης του Serial Monitor

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Hello Arduino, ");
  Serial.print("I am a great fan!");
  Serial.println("I like Arduino, Sensors and the Cloud");
}

void loop()
{
}
```

Στη συνάρτηση `setup()` αρχικοποιείται η σειριακή επικοινωνία σε 9600 baud rate. Αυτό γίνεται με τη μέθοδο `begin()`. Έπειτα ξεκινά η εκτύπωση των μηνυμάτων στο Serial Monitor. Παρατηρήστε τη διαφορά μεταξύ των συναρτήσεων `println()` και `print()`.

Προσθέστε αυτόν τον κώδικα στον editor σας και κάντε κλικ στο κουμπί upload (βεβαιωθείτε ότι η πλακέτα Arduino σας είναι συνδεδεμένη και έχετε επιλέξει την κατάλληλη πλακέτα και θύρα USB από Tools menu). Όταν γίνει upload κάντε κλικ στο εικονίδιο Serial Monitor (ή πηγαίνετε από το μενού Tools-> Serial Monitor). Το παράθυρο Serial Monitor θα εμφανιστεί, το sketch θα ξεκινήσει και πάλι στη πλακέτα (ο κώδικας εκτελείται στην πλακέτα αμέσως, αφού τελειώσει η διαδικασία upload του sketch). Για να είστε σε θέση να παρακολουθήσετε τις εξόδους στο Serial Monitor από την αρχή του sketch σας, κατά την εκτέλεση γίνεται επαναφορά κάθε φορά που ανοίγετε το Serial Monitor. Στην κάτω δεξιά γωνία του παραθύρου βεβαιωθείτε ότι το επιλεγμένο baud rate είναι η ίδια που έχετε χρησιμοποιήσει στην συνάρτηση `setup()` (9600). Θα δείτε το ακόλουθο κείμενο εξόδου:

```
Hello Arduino,  
I am a great fan! I like Arduino, Sensors and the Cloud
```

Ας δούμε τώρα πώς μπορείτε να στείλετε πληροφορίες στο Arduino σας μέσω του Serial Monitor. Δημιουργήστε ένα νέο sketch από την αρχή και προσθέστε τον ακόλουθο κώδικα:

Υπόδειγμα 4-2. Απλό Παράδειγμα Αποστολής δεδομένων στο Serial Monitor

```
void setup()  
{  
  Serial.begin(9600);  
}  
char in;  
void loop() {  
  while(Serial.available()>0)  
  {  
    in = Serial.read();  
    Serial.print(in);  
  }  
}
```

Κάντε upload το sketch σας στην πλακέτα σας και ανοίξτε το Serial Monitor όταν ολοκληρωθεί το upload. Στο επάνω μέρος του παραθύρου Serial Monitor είναι ένα επεξεργάσιμο πεδίο κειμένου. Πληκτρολογήστε ένα μήνυμα, οτιδήποτε, και πατήστε enter. Θα δείτε το περιεχόμενο του μηνυματός σας να εμφανίζεται στην οθόνη.

Ο κώδικας αρχικοποιεί το Serial Monitor σε baud rate 9600 όπως στο προηγούμενο παράδειγμα. Μπορείτε επίσης να ορίσετε μια μεταβλητή `char (in)` για

την αποθήκευση των εισερχόμενων δεδομένων. Στη συνέχεια στη συνάρτηση loop() χρησιμοποιείτε τη συνάρτηση available() της κλάσης Serial. Αυτή η μέθοδος επιστρέφει true σε περίπτωση που τα δεδομένα που λαμβάνονται από το Serial (δηλαδή όταν στέλνεται το μήνυμά σας). Σε αυτή την περίπτωση ο βρόχος (loop) εκτελείται μέχρι να μην λαμβάνονται άλλα δεδομένα (όταν ειδοποιείται ότι έχει τελειώσει). Μέσα στο βρόχο αποθηκεύονται τα εισερχόμενα δεδομένα τύπου char μέσα στη μεταβλητή In με τη χρήση της μεθόδου Serial.read(). Μπορείτε επίσης να εκτυπώσετε κάθε χαρακτήρα που έρχεται, έτσι ώστε να μπορείτε να δείτε την έξοδο στο Serial Monitor.

Έλεγχος θυρών I/O

Σε εφαρμογές Cloud είναι πολύ κοινό να ελέγχονται συσκευές ανάγνωσης ή καταστάσεις εισόδου χρησιμοποιώντας τις θύρες I/O. Οι ακόλουθες καταχωρήσεις κώδικα δείχνουν πώς μπορείτε να ελέγξετε την κατάσταση ενός LED με εναλλαγή μεταξύ υψηλής και χαμηλής κατάστασης μιας ψηφιακής θύρας και πώς να διαβάσετε την είσοδο ενός ψηφιακής θύρας για να πάρετε εξωτερικά γεγονότα, όπως το πάτημα ενός κουμπιού.

Υπόδειγμα 4.3: Ο έλεγχος της κατάστασης ενός LED μέσα από μια ψηφιακή θύρα

```
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
void loop() {  
  digitalWrite(13, HIGH); // set the LED on  
  delay(1000); // wait for a second  
  digitalWrite(13, LOW); // set the LED off  
  delay(1000); // wait for a second  
}
```

Υπόδειγμα 4.4: Αλλαγή μιας κατάστασης LED που στηρίζεται σε γεγονότα Button

```
const int buttonPin = 2; // the number of the pushbutton  
pin  
const int ledPin = 13; // the number of the LED pin  
// variables will change:  
int buttonState = 0; // variable for reading the  
pushbutton status  
void setup() {  
  // initialize the LED pin as an output:  
  pinMode(ledPin, OUTPUT);
```

```

// initialize the pushbutton pin as an input:
pinMode(buttonPin, INPUT);
}
void loop(){
// read the state of the pushbutton value:
buttonState = digitalRead(buttonPin);
// check if the pushbutton is pressed.
// if it is, the buttonState is HIGH:
if (buttonState == HIGH) {
// turn LED on:
digitalWrite(ledPin, HIGH);
}
else {
// turn LED off:
digitalWrite(ledPin, LOW);
}
}
}

```

Σε περίπτωση που χρειάζεστε περισσότερες πληροφορίες και παραδείγματα σχετικά με τους κώδικες Arduino και τα projects, μπορείτε να ελέγξετε την ιστοσελίδα της Amazon για τη σειρά βιβλίων που είναι κατάλληλα τόσο για αρχάριους όσο και για προχωρημένους χρήστες Arduino!

Δοκιμάζοντας τον κώδικά σε ένα προσομοιωτή Arduino

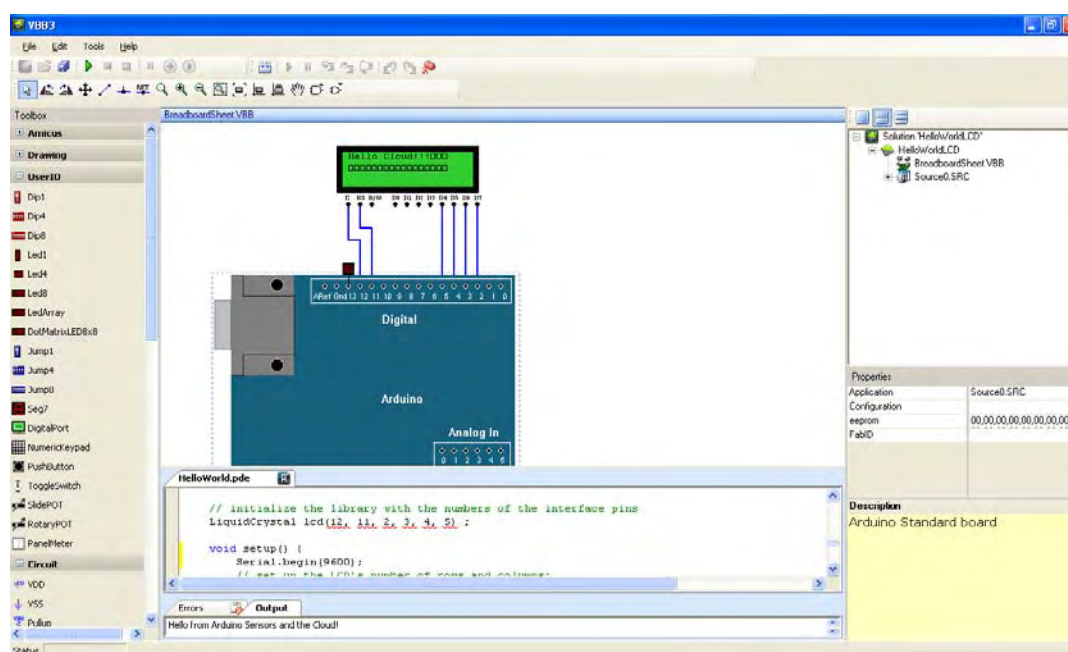
Όταν παίζετε με το Arduino σας και πειραματίζεστε με τις δυνατότητες και τα χαρακτηριστικά μπορεί να βρεθείτε στις ακόλουθες καταστάσεις: είτε έχετε μια καλή ιδέα για ένα νέο project, ή θέλετε να δοκιμάσετε ένα υπάρχον, αλλά δεν έχετε τα απαραίτητα εργαλεία. Ίσως να λείπουν τα κουμπιά LED, ένα ποτενσιόμετρο ακόμη και μία μονάδα Ethernet (δείτε Εικόνα 4.9) αλλά εξακολουθείτε να έχετε το sketch σας έτοιμο. Βλέποντας το sketch σας να μεταγλωττίζεται (compile) στο Arduino IDE δεν είναι τόσο ικανοποιητικό όσο βλέποντας το να λειτουργεί στο board σας και θα πρέπει επίσης να βεβαιωθείτε ότι η λογική του προγράμματος σας λειτουργεί. Τι μπορείτε να κάνετε χωρίς το hardware; Μπορείτε να χρησιμοποιήσετε ένα Arduino Simulator όπως το Arduino VirtualBreadboard.

Το VirtualBreadboard (<http://www.virtualbreadboard.net/>) είναι (δυστυχώς μόνο για Windows) περιβάλλον προσομοίωσης και ανάπτυξης για Μικροελεγκτές. Υποστηρίζει πολλές από τις PIC16 και PIC18 συσκευές μικροελεγκτή και την πλατφόρμα Arduino. Επιπλέον, παρέχει μία μεγάλη ποικιλία από προσομοιωμένων συστατικά όπως LCD, μοτέρ, logic και άλλες συσκευές I / O οι οποίες μπορούν να χρησιμοποιηθούν για την απεικόνιση και προσομοίωση κυκλωμάτων υψηλού επιπέδου. Όταν μιλάμε για περιβάλλον προσομοίωσης, σημαίνει ότι έχετε πρόσβαση σε εικονικούς πίνακες και στοιχεία που μπορούν να συνδεθούν μεταξύ τους εικονικά με ένα κατάλληλο γραφικό περιβάλλον (βλ. Εικόνα 4.7). Μπορείτε να προσθέσετε γραμμές κώδικα (με τον ίδιο τρόπο που θα προσθέτατε ένα sketch),

μπορείτε να μεταγλωττίσετε τον κώδικα και να τον βλέπετε να εκτελείται στην πλατφόρμα του μικροελεγκτή και να αλληλεπιδρά με τα συνδεδεμένα συστατικά (LEDs, sensors, actuators) με τον ίδιο τρόπο όπως αν το κύκλωμα σας ήταν πραγματικό!

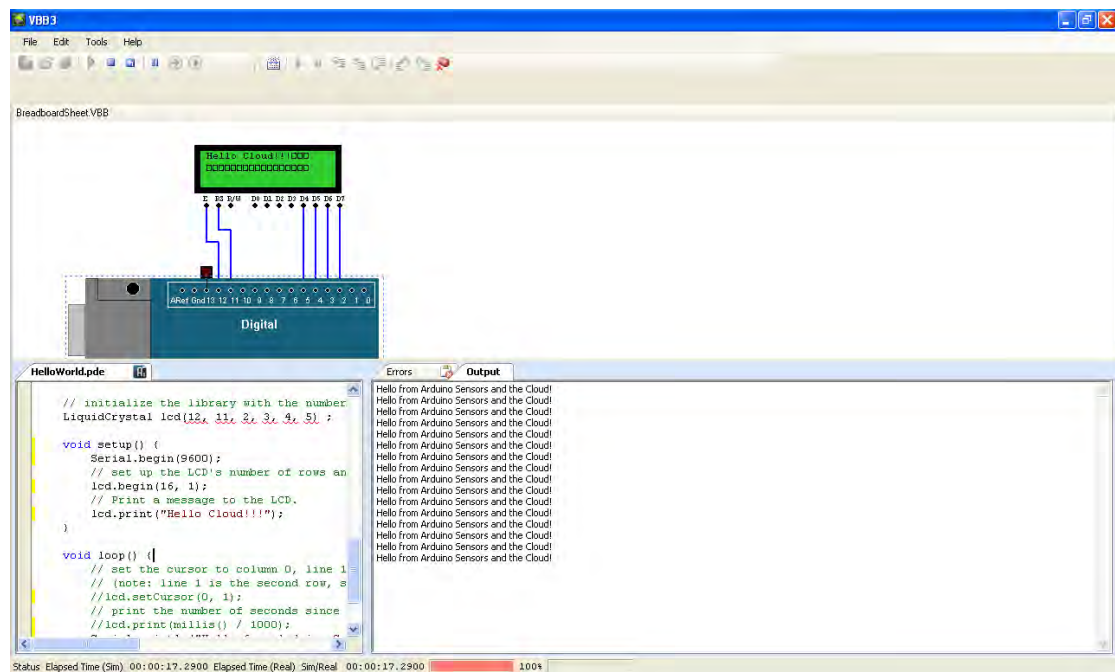
Το VirtualBreadboard υποστηρίζει όλα τα βασικά παραδείγματα sketch που περιλαμβάνονται στο Arduino IDE. Ο κώδικας sketch και τα βασικά στοιχεία του κυκλώματος (είναι ήδη συνδεδεμένα με μια εικονική πλακέτα Arduino), όπως LED, κουμπιά, οθόνες LCD, Servo motors κλπ. για την υποστήριξη της εικονικής εκτέλεσης των sketches.

Όπως φαίνεται στην Εικόνα 4.7 το κύριο γραφικό περιβάλλον του VirtualBreadboard σας προσφέρει μια εικονική αναπαράσταση μιας πλακέτα Arduino. Στα αριστερά μπορείτε να βρείτε μια ποικιλία από εξαρτήματα και στοιχεία κυκλώματος που μπορείτε να τα προσθέσετε στον εικονικό χώρο εργασίας σας μέσω drag-and-drop και να τα συνδέσετε στα pin του board σας.



Εικόνα 4.7: Το κύριο περιβάλλον VirtualBreadboard. Μπορείτε να συνδέσετε μια ποικιλία του hardware σε μια εικονική πλακέτα Arduino και γράψετε τον ίδιο κώδικα sketch που θα κάνατε με το πραγματικό Arduino σας.

Η κύρια οθόνη του γραφικού περιβάλλοντος χωρίζεται σε δύο μέρη. Το πάνω μέρος περιέχει την εικονική αναπαράσταση της εγκατάστασης του κυκλώματος και σας επιτρέπει να συνδέετε τις συσκευές μεταξύ τους, επίσης καθορίζει πιο pin της συσκευής πηγαίνει σε ποιο pin του Arduino pin, κλπ. Το κάτω μέρος περιέχει τον editor του sketch. Ο editor έχει παρόμοια χαρακτηριστικά με τον Arduino IDE editor (σύνταξη, κλπ). Βασικά μπορείτε να προσθέσετε ακριβώς τον ίδιο κώδικα (με κάποιες πρόσθετες πληροφορίες κεφαλίδας) που χρησιμοποιείτε για το προεπιλεγμένο σας IDE. Το VirtualBreadboard θα μεταγλωττίσει τον κώδικα και θα ξεκινήσει η εικονική εκτέλεση (δεδομένου ότι δεν βρίσκει λάθος στη σύνταξη) όπως φαίνεται στην Εικόνα 4.8.



Εικόνα 4.8: Το VirtualBreadboard εκτελεί το sketch σας και μπορείτε να δείτε τα αποτελέσματα του κώδικα σας, στην προσομοίωση της πλακέτα Arduino και τα συνδεδεμένα περιφερειακά και την προσομοίωση του Serial Monitor.

Κατά τη διάρκεια της εικονικής εκτέλεσης, μπορείτε να δείτε και τα αποτελέσματα του κώδικα σας για το κύκλωμα που έχετε δημιουργήσει (π.χ., την ενεργοποίηση των LED, την ενεργοποίηση ενός μοτέρ, χαρακτήρες που εκτυπώνονται στις οθόνες LCD κλπ), αλληλεπιδρούν με αυτό (πχ να γυρίσετε το κουμπί του ποτενσιόμετρου, πιεστικά κουμπιά κλπ), μπορείτε να δείτε επίσης την έξοδο από την εικονική σειριακή επικοινωνία με την πλακέτα Arduino σαν να παρακολουθείτε το Serial Monitor.

Εκτός από την εκτέλεση κώδικα σας εικονικά, το VirtualBreadboard μπορεί να φορτώσει κατευθείαν το sketch στην πραγματική πλακέτα όταν το συνδέεται με τον υπολογιστή σας. Έτσι, μπορείτε αρχικά να βεβαιωθείτε ότι η λογική του προγράμματός σας είναι σωστή και στη συνέχεια να το φορτώσετε για την εκτέλεση σε μία πραγματική πλακέτα.

Από το 2005 που το Arduino παρουσιάστηκε, συνεχώς ενισχύεται και συνεχώς εξελίσσεται. Η επίσημη ομάδα Arduino όχι μόνο φροντίζει για την ανάπτυξη νέων πλακετών, τη βελτίωση της IDE προσθέτοντας περισσότερες λειτουργίες, αλλά και οι εξωτερικοί προμηθευτές (όπως Sparkfun ή Seeedstudio) έχουν δημιουργήσει επεκτάσεις για το ίδιο hardware. Αυτές οι επεκτάσεις είναι γνωστές ως Arduino Shields.



Εικόνα 4.9: Το Arduino Ethernet Shield. Είναι σχεδιασμένο για να συνδεθεί απευθείας στην κορυφή της πλακέτας Arduino επεκτείνει τη λειτουργικότητα με την παροχή των κατάλληλων κυκλωμάτων και υποδοχή ethernet, και ταυτόχρονα επιτρέπει στο χρήστη να έχει πρόσβαση στα pin I/O του κυκλώματος.

Τα shields είναι κυκλώματα που μπορούν να συνδεθούν πάνω σε μία πλακέτα Arduino επεκτείνοντας έτσι τις δυνατότητές της. Τέτοιες δυνατότητες μπορούν να θεωρούν δικτύωσης, όπως το Ethernet, το WiFi ή τα Xbee Shields. Η Εικόνα 4.9 απεικονίζει το Ethernet shield που παρέχει μία θύρα Ethernet. Μπορεί να συνδεθεί στην κορυφή από ένα Uno ή Mega και ταυτόχρονα εκθέτουν τα pin I/O του κυκλώματος προς χρήση. Τα shields μπορούν επίσης να χρησιμοποιηθούν για να επεκτείνουν τις θύρες του κυκλώματος ή να επιτρέψουν στους χρήστες να συνδέσουν αισθητήρες πιο εύκολα όπως το ηλεκτρονικό Shield Brick της Seeedstudio, μπορούν να προσθέσουν εύκολη πρόσβαση σε SD κάρτα μνήμης, με Bluetooth και GPRS/3G μόντεμ και πολλά άλλα. Η λίστα των χαρακτηριστικών που

μπορούν να προστεθούν στο Arduino μέσω shields είναι πραγματικά ατελείωτες. Όποια και αν είναι η νέα τεχνολογία στο πλαίσιο της επικοινωνίας και της αλληλεπίδρασης με τα κυκλώματα και τα ηλεκτρονικά το πιθανότερο είναι ότι μπορεί να εφαρμοστεί ως shield για το Arduino. Η ιστοσελίδα του Arduino shield (<http://shieldlist.org/>) απαριθμεί σήμερα πάνω από 250 Shields Arduino με διαφορετικές λειτουργίες!

Τα διαφορετικά shields ακολουθούν πάντα την ίδια φιλοσοφία με το αρχικό Arduino: θα πρέπει να είναι εύκολο να τοποθετηθεί, και πάντα παρέχει στους χρήστες τις κατάλληλες βιβλιοθήκες για χρήση.

Χρηση Βιβλιοθηκών

Εκτός από την επέκταση του hardware του Arduino υπάρχουν πολλοί τρόποι να επεκτείνεται το Software επίσης που δεν παρέχονται στο αρχικό προγραμματιστικό περιβάλλον. Όπως σχεδόν σε κάθε γλώσσα προγραμματισμού, μπορείτε να επεκτείνετε τις δυνατότητες του προγράμματός σας με τη χρήση βιβλιοθηκών, είτε με ήδη υπάρχουσες (internal libraries) ή με εξωτερικές (που χρειάζεται να τις εισάγεται στο προγραμματιστικό σας περιβάλλον).

Όσον αφορά τις βιβλιοθήκες Arduino είναι αρχεία που είναι γραμμένα σε C ή C++ και παρέχουν στα sketches σας επιπλέον λειτουργίες (π.χ. την ικανότητα για έλεγχο μιας οθόνης LCD ή την επικοινωνία με το Internet, κ.λπ.)

Για να χρησιμοποιήσετε μια υπάρχουσα βιβλιοθήκη σε ένα κώδικα απλά πηγαίνετε στο sketch menu, επιλέξτε “Import Library” και επιλέξτε από τις διαθέσιμες βιβλιοθήκες. Αυτό θα εισάγει μια δήλωση # include στην κορυφή του κώδικα για κάθε αρχείο κεφαλίδας (.h) στο φάκελο της βιβλιοθήκης. Αυτές οι δηλώσεις κάνουν τις public συναρτήσεις και τις σταθερές που ορίζονται από τη βιβλιοθήκη διαθέσιμες στο πρόγραμμά σας. Επίσης προτρέπουν το περιβάλλον Arduino να συνδέσει αυτή τη βιβλιοθήκη με τον κώδικα σας όταν αυτό μεταγλωττιστεί και γίνει upload.

Σε περίπτωση που επιθυμείτε να χρησιμοποιήσετε μια εξωτερική βιβλιοθήκη θα πρέπει να την εισάγεται στο Arduino IDE. Οι βιβλιοθήκες που είναι δημιουργημένες από ένα χρήστη όπως η version 0017 πάνε σε έναν υποκατάλογο του βασικού καταλόγου του κώδικα. Για παράδειγμα στο OSX ο νέος κατάλογος θα είναι ~/Documents/Arduino/libraries/. Στα Windows, θα είναι My Documents\Arduino\libraries.

Τρέχοντας τον κώδικα απευθείας από τον μικροεπεξεργαστή

Εξ ορισμού, η πλακέτα Arduino είναι μία πρωτότυπη πλακέτα. Έχετε μια πλατφόρμα μικροελεγκτή και το περιβάλλον σε μεγάλο χαμηλό κόστος και μπορείτε να δοκιμάσετε εύκολα και γρήγορα προτυποποίηση των δικών σας σχεδίων. Αλλά τι γίνεται εάν θα πρέπει να προχωρήσετε από το πρωτότυπο και να αναπτύξετε ένα πιο μόνιμο σχέδιο ή σε μεγαλύτερη κλίμακα;

Η απάντηση είναι πως μπορείτε να προγραμματίσετε εύκολα το τσιπ των μικροελεγκτών, ενώ είναι επί της πλακέτας και στη συνέχεια να το αφαιρέσετε και να το χρησιμοποιήσετε ως αυτόνομη συσκευή στο κύκλωμα σας. Ας δούμε ένα μικρό παράδειγμα που θα σας δώσει τη βασική ιδέα και να σας παρουσιάσουμε τα υλικά που χρειάζεστε για να τρέξετε ένα Arduino sketch από την πλακέτα. Το παράδειγμα βασίζεται στο sketch Blink. Θα χρησιμοποιήσετε μια πλακέτα Arduino όπως η Uno για να προγραμματίσετε το μικροελεγκτή με το sketch και στη συνέχεια θα την αφαιρέσετε και θα το κάνετε να τρέχει από μόνο του.

Τι θα χρειαστείτε

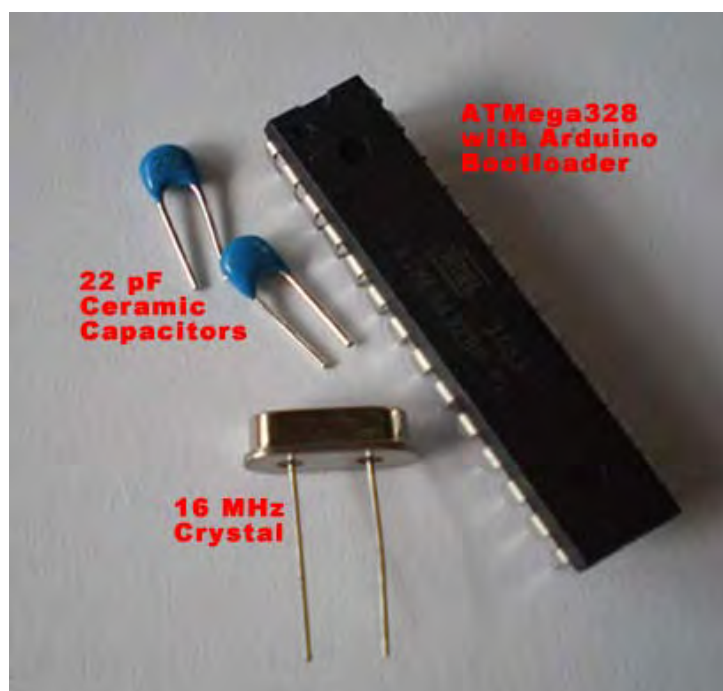
Θα χρειαστείτε έναν μικροελεγκτή ATMEGA (είτε ATmega128 ή ATmega328) με εγκατεστημένο Arduino Bootloader. Μπορείτε να χρησιμοποιήσετε είτε το τσιπ που θα βρείτε στην πλακέτα Uno ή να αγοράσετε ένα. Θα χρειαστείτε επίσης ένα ζευγάρι από 22 pF κεραμικών πυκνωτών και ένα 16 MHz ταλαντωτή κρυστάλλου. Η Εικόνα 4.10 απεικονίζει τους πυκνωτές και τον κρύσταλλο δίπλα στον ATmega328 μικροελεγκτή. Οι λειτουργίες ταλαντωτή ως εξωτερικό ρολόι - χρονόμετρο που δίνει κύκλους ρολογιού στον επεξεργαστή. Ο ATmega328 έρχεται με έναν εσωτερικό ταλαντωτή (σε 8MHz), αλλά είναι καλύτερα να γνωρίζεται πώς να χρησιμοποιήσετε έναν εξωτερικό, απαραίτητο, ειδικά σε πιο σύνθετα έργα.

Εκτός από τα βασικά συστατικά, έτσι ώστε το τσιπ μπορεί να τρέξει μόνο του, θα χρειαστείτε και τροφοδοσία (μπορείτε να τροφοδοτήσετε το chip εύκολα από την πλακέτα Arduino), ένα LED (για αυτό το παράδειγμα), ένα Breadboard να δημιουργήσετε το κύκλωμα σας και κάποια καλώδια βραχυκύκλωσης για να συνδεθούν όλα μαζί. Ως εναλλακτική λύση προς τον ταλαντωτή και τους δύο πυκνωτές μπορείτε να πάρετε ένα κεραμικό ταλαντωτή της ίδιας συχνότητας (16MHz).

Μερικά συστήματα έρχονται με την SMD έκδοση του μικροελεγκτή. Προφανώς δεν μπορείτε να καταργήσετε αυτό το τσιπ και να το χρησιμοποιήσετε ως αυτόνομο ή είναι αρκετά δύσκολο να χρησιμοποιήσετε ένα σε breadboard. Βεβαιωθείτε ότι έχετε την 40-pin έκδοση του τσιπ όπως φαίνεται στην Εικόνα 4.10.

Στη πράξη

Ας τα βάλουμε όλα μαζί για να τρέξει το sketch Arduino Blink για το μικροελεγκτή από μόνο του. Πρώτα θα πρέπει να προγραμματίσετε το μικροελεγκτή σας. Αν υποθέσουμε ότι είναι ήδη στον πλακέτα, απλά ανεβάστε το sketch από το Arduino IDE. Το sketch βρίσκεται στο: Examples -> Basics -> Blink. Δεν χρειάζεται να κάνετε οποιεσδήποτε τροποποιήσεις στο sketch. Βεβαιωθείτε ότι έχετε επιλέξει την κατάλληλη πλακέτα από το hardware menu που αντιστοιχεί στον τύπο του μικροελεγκτή σας. Στη συνέχεια απλά κάντε upload το sketch. Όταν τελειώσετε, θα παρατηρήσετε την εσωτερική πλακέτα LED (που ορίζεται ως ψηφιακή θύρα 13) να αναβοσβήνει.



Εικόνα 4.10: Τα βασικά υλικά για τη λειτουργία ενός sketch Arduino στο μικροελεγκτή (έξω από το κουτί)

Ας πάμε στο ενδιαφέρον κομμάτι τώρα. Αφαιρέστε πρώτα το καλώδιο USB από την πλακέτα (βεβαιωθείτε ότι δεν τροφοδοτείται με οποιοδήποτε άλλο μέσο). Στη συνέχεια, αφαιρέστε το τσιπ μικροελεγκτή από την υποδοχή της πλακέτας. Μπορείτε να χρησιμοποιήσετε ένα μικρό κατσαβίδι για να σας βοηθήσει στην ανύψωση προς τα επάνω. Να είστε προσεκτικοί, ενώ απελευθερώνετε τα pin του από την υποδοχή, ώστε να μην λυγίσει κάποια από αυτά.

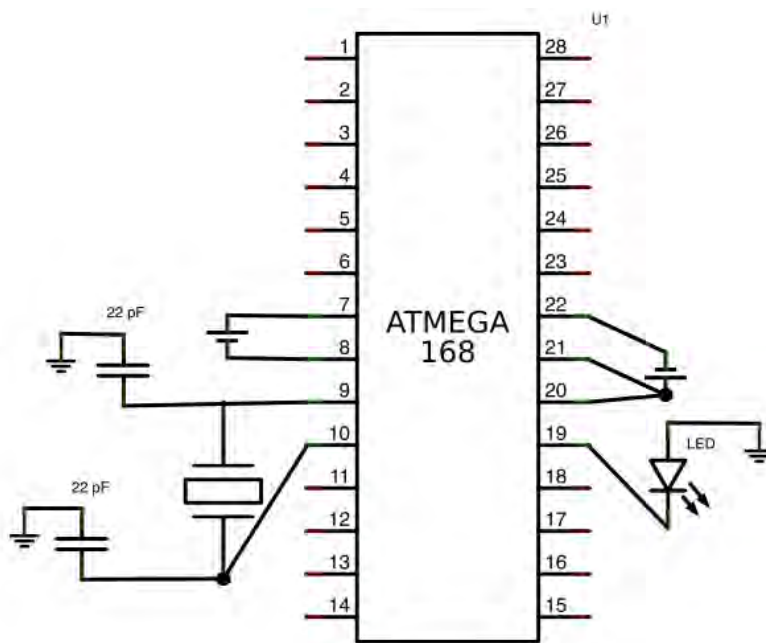
Τώρα που έχετε απελευθερώσει το τσιπ θα είστε έτοιμοι να ασχοληθείτε με την εγκατάσταση breadboard. Με βάση τους ATMEGA 168/328 Μικροελεγκτές σχηματικά (<http://arduino.cc/en/Hacking/PinMapping168>), θα χρειαστεί να συνδέσετε κάποιο από τα pin του στην πλακέτα και στη γείωση και να συνδέσετε τον ταλαντωτή με το τσιπ. Πιο συγκεκριμένα:

- Τα Pins 7, 21 και 20 πρέπει να συνδεθούν με μία πηγή τροφοδοσίας (Vcc)

- Τα Pins 8 και 22 πρέπει να συνδεθούν με τη γείωση
- Τα Pins 9 και 10 πρέπει να συνδεθούν με τον ταλαντωτή κρυστάλλου

Η πλειοψηφία των υπολοίπων pins λειτουργούν ως I / O pins. Με βάση το sketch σας και τον μικροελεγκτή σχηματικά, θα πρέπει να συνδέσετε τη θετικό pin του LED (το μεγαλύτερο pin) στο pin 19, το οποίο αντιστοιχεί στο ψηφιακό pin 13 για το Arduino.

Για την καλύτερη εξυπηρέτησή σας οι κατάλληλες συνδέσεις, όπως με το τσιπ και τα modules παρουσιάζονται στην Εικόνα 4.11.



Εικόνα 4.11: Ο σχηματισμός για τη σύνδεση των πυκνωτών, ο ταλαντωτής κρυστάλλου, το LED, Ισχύς (V_{CC}) και η γείωση (GND) με το τσιπ μικροελεγκτών.

ΚΕΦΑΛΑΙΟ 4 Βασικές Αρχές Αισθητήρων και Ενεργοποιητών

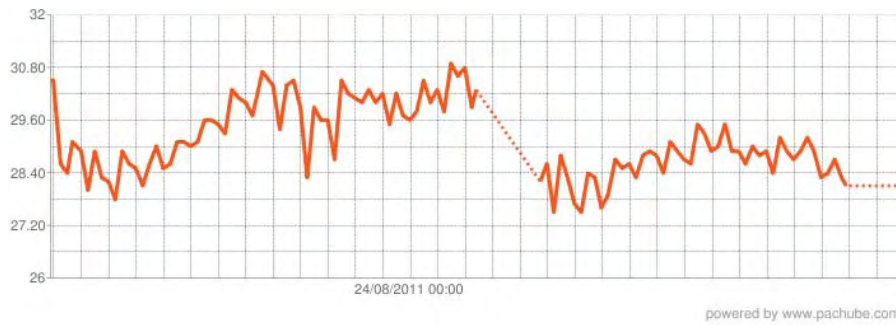
Για να κατασκευάσετε τη δική σας συσκευή, απαιτεί να καθορίσετε πρώτα τι θα συνδέσετε σε αυτή και τι είδους λειτουργίες θέλετε να εκτελεί. Αυτό μας οδηγεί στο δεύτερο τμήμα του κειμένου μας, που παρουσιάζει τις βασικές αρχές των αισθητήρων και των ενεργοποιητών. Θα σας βοηθήσει να καταλάβετε τι μπορείτε να συνδέσετε σε συσκευές που αλληλεπιδρούν με το internet, τι πληροφορίες μπορείτε να αξιοποιήσετε από το περιβάλλον και πως μπορείτε να ελέγξετε δεδομένα από αυτό.

Θα ξεκινήσουμε περιγράφοντας τι είναι και τι κάνουν οι αισθητήρες. Μέρος αυτής της διαδικασίας είναι και η επεξήγηση της δειγματοληψίας και της αναλογικό-ψηφιακής μετατροπής. Επιπρόσθετα παραθέτουμε μία λίστα με τους πιο συνηθισμένους αισθητήρες που μπορείτε να χρησιμοποιήσετε για να φτιάξετε ένα project.

Εισαγωγή

Για να συζητήσουμε τι είναι ένας αισθητήρας θα πρέπει πρώτα να κάνουμε μία εισαγωγή για τις αντιστάσεις. Ο λόγος που το κάνουμε αυτό είναι γιατί οι περισσότεροι αναλογικοί αισθητήρες λειτουργούν με παρόμοιο τρόπο με την αντίσταση. Ο αντιστάτης είναι το κομμάτι εκείνο ενός κυκλώματος που ρίχνει την τάση μεταξύ δύο ακροδεκτών. Η πτώση τάσης που προκαλείται είναι αντιστρόφως ανάλογη της πτώσης του ρεύματος που προκαλείται. Οι αντιστάσεις αποτελούνται από διάφορα υλικά και βρίσκονται σε διάφορες μορφές, σχεδόν σε κάθε κύκλωμα.

Ένας αναλογικός αισθητήρας όπως το θερμίστορ (αντιστάτης που μεταβάλλει την τιμή του ανάλογα με τη θερμοκρασία) ή ο αισθητήρας υγρασίας, είναι συνδεδεμένος στο κύκλωμα με τρόπο τέτοιο που θα δίνει έξοδο εντός ενός συγκεκριμένου ορίου, συνήθως μεταξύ 0 και 5V. Αυτό καλείται αναλογικό σήμα και είναι συνεχές και στο χρόνο και στην ενίσχυση (Εικόνα 2.1). Η έξοδος πηγαίνει σε μία αναλογική θύρα του μΥπολογιστή, ο οποίος τη μετατρέπει σε αντίστοιχη ψηφιακή τιμή, ώστε να μπορεί να τη διαβάσει, επεξεργαστεί και να τη στείλει στο internet. Ένα παράδειγμα του πως ένας αισθητήρας συνδέεται στον μΥπολογιστή αναλύεται στο αντίστοιχο κεφάλαιο (Εικόνα 1.4). Όλοι οι αναλογικοί αισθητήρες έρχονται συνήθως με 3 ακροδέκτες, έναν για τη συνεχή τροφοδοσία (Vcc), έναν για τη γείωση (GND) και έναν για την έξοδο (OUT)



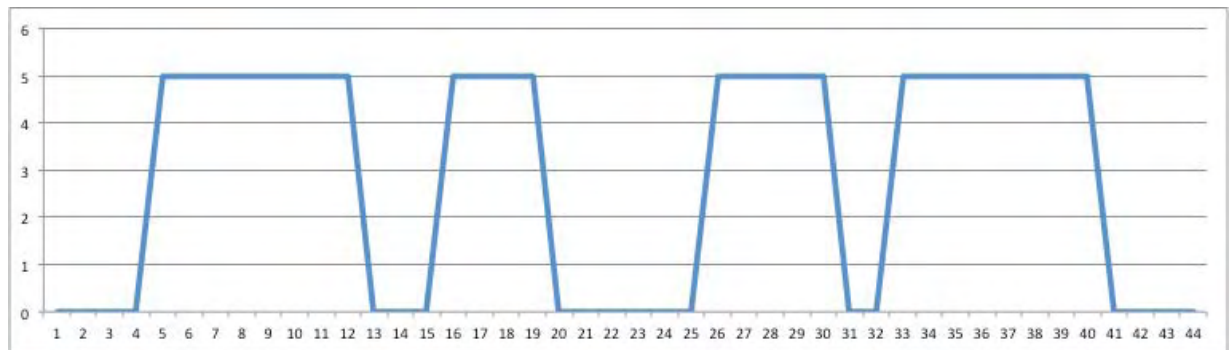
Εικόνα 2.1: Ένα αναλογικό σήμα όπως παράγεται από έναν αναλογικό αισθητήρα που μετράει θερμοκρασία σε ένα σπίτι. Το σήμα είναι συνεχές, τόσο στο χρόνο, όσο και στην τάση εξόδου (ενίσχυση).

Ψηφιακοί Αισθητήρες

Όπως υποδηλώνει και το ονομά τους, οι ψηφιακοί αισθητήρες παράγουν διακριτές τιμές σήματος ή τάσης στην έξοδό τους (ψηφία) που αντιπροσωπεύουν την τιμή που μετρούν. Οι ψηφιακοί αισθητήρες παράγουν μία ψηφιακή μορφή εξόδου, με μορφή λογικού '1' ή λογικού '0' (ή ON και OFF). Σε μορφή τάσης, το 1 αντιπροσωπεύεται συνήθως από τα 5V, ενώ το 0 από τα 0V. Αυτό σημαίνει ότι ένα ψηφιακό σήμα παράγει μόνο μη συνεχείς τιμές (που καλούμε ανεξάρτητες τιμές), οι οποίες μπορεί να είναι είτε ανεξάρτητα 'bit' (σειριακή μετάδοση), είτε συνδυάζοντας τα bit ώστε να παράγουν ένα 'byte' εξόδου (παράλληλη μετάδοση).

Συγκρινόμενα με τα αναλογικά σήματα τα ψηφιακά είναι πολύ περισσότερο ακριβή (δεν είναι επιρρεπή στο θόρυβο συγκρινόμενα με τα αναλογικά) και μπορούν να μετρηθούν και να δειγματοληφθούν σε πολύ υψηλές ταχύτητες ρολογιού. Η ακρίβεια των ψηφιακών σημάτων εξαρτάται από τον αριθμό των bits που χρησιμοποιούνται για να αντιπροσωπεύσουν τη μετρούμενη ποσότητα. Η Εικόνα 2.2 δείχνει ένα παράδειγμα ψηφιακού σήματος.

Ο απλούστερος ψηφιακός αισθητήρας είναι ένα κουμπί ON-OFF (δύο σταδίων) καθοριζόμενο από το χρήστη. Δίνει στην είσοδο του μικροελεγκτή είτε 5V είτε 0V. Για να διαβάσετε την είσοδο του αισθητήρα, απλώς συνδέεται στην είσοδο της πλακέτας σας το κύκλωμα του αισθητήρα (μπορεί να χρησιμοποιηθεί και μία αντίσταση pull up όπως θα εξηγηθεί παρακάτω) και ελέγχετε τότε υπάρχει χαμηλή ή υψηλή είσοδος όταν ανοίγει το κύκλωμα. Πιο αναβαθμισμένοι αισθητήρες είναι αυτοί που εμπεριέχουν κύκλωμα ή επεξεργαστή, ο οποίος μετατρέπει την αναλογική μέτρηση κάποιας τιμής περιβάλλοντος σε έξοδο bit ή bytes. Αυτά μεταφέρονται μέσω της εξόδου του αισθητήρα στον μικροεπεξεργαστή. Για να αναγνωριστούν τα εισερχόμενα bits απαιτείται κάποια επιπλέον κωδικοποίηση, καθώς και η γνώση του πρωτοκόλλου επικοινωνίας (πχ πως θα οριστούν τα bytes) που παρέχεται από τον προμηθευτή του αισθητήρα. Παρά του ότι οι αισθητήρες αυτοί είναι πιο δύσκολοι στη χρήση τους, μπορούν αν δώσουν πιο εξειδικευμένες μετρήσεις μεγεθών και να ενσωματώσουν περισσότερες από μία μετρήσεις σε έναν αισθητήρα (πχ θερμοκρασία και υγρασία).



Εικόνα 2.2: Ένα παράδειγμα ψηφιακού σήματος

Αντιστάσεις Pull up-Down και Αισθητήρες

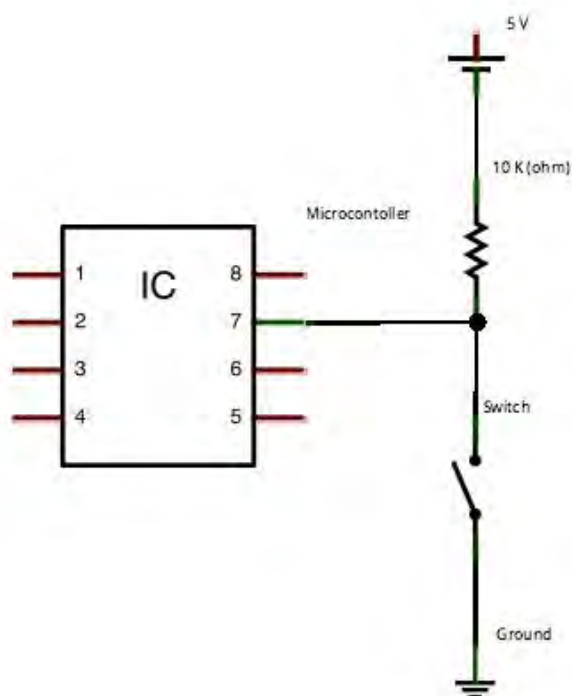
Μπορείτε να συνδέσετε στον μικροεπεξεργαστή σας απευθείας έναν αισθητήρα; Θεωρητικά ναι, αλλά στην πράξη παρεμβάλλουμε συνήθως μία αντίσταση η οποία καλείται pull up ή pull down.

Οι αντιστάσεις pull up χρησιμοποιούνται σε λογικές εισόδους (πχ στις εισόδους του μΥπολογιστή), ώστε να διατηρούν το σωστό επίπεδο τάσης, ακόμα και όταν αφαιρεθεί από το κύκλωμα ο αισθητήρας. Ονομάζονται έτσι γιατί “τραβάνε” (pull) την τάση σε ένα υψηλό (up) προκαθορισμένο επίπεδο. Εάν κάποιο άλλο στοιχείο του κυκλώματος θέσει την τάση σε διαφορετικό επίπεδο, η pull up αντίσταση δεν θα επηρεάσει το κύκλωμα. Η βασική λειτουργία της pull up αντίστασης είναι να περιορίζει το κύκλωμα από το να περάσει ένα υψηλό ρεύμα που μπορεί να προκαλέσει βλάβη και γι’ αυτό χρησιμοποιείται ευρύτατα.

Το περιβάλλον του μικροελεγκτή Arduino έχει ενσωματωμένες αντιστάσεις pull up. Αυτό σημαίνει πως μπορείτε να συνδέσετε απευθείας τους αισθητήρες στις εισόδους του, αλλά θα πρέπει παρόλα αυτά να έχετε υπόψη τη χρησιμότητα των αντιστάσεων αυτών.

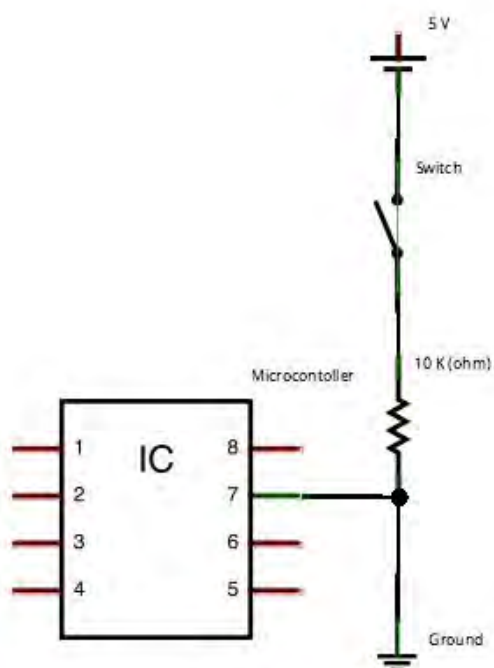
Σαν εναλλακτική λύση στη χρήση αντιστάσεων pull up, είναι η χρήση αντιστάσεων pull down. Όπως και οι pull up, χρησιμοποιούνται για να περιορίσουν το ρεύμα του κυκλώματος, ανάμεσα στο Vcc και το GND. Γι’ αυτό το λόγο μπορεί αν ενσωματωθεί στο κύκλωμα μία αντίσταση 10K ή 472KΩ. Η ακριβής τιμή δεν έχει ιδιαίτερη σημασία από τη στιγμή που είναι αρκετά μεγάλη ώστε να περιορίζει το διαρρέον ρεύμα στις εισόδους του μικροελεγκτή.

Ας δούμε ένα παράδειγμα χρήσης pull up αντίστασης. Σκεφθείτε ένα κύκλωμα διακόπτη και μικροελεγκτή, όπως στη Εικόνα 2.3. Ο στόχος είναι να εντοπίσετε μέσω του μικροελεγκτή πότε ο διακόπτης είναι κλειστός και πότε ανοιχτός. Ας συνδέσουμε τον έναν ακροδέκτη του διακόπτη στο GND και τον άλλον στην τάση 5V μέσω μιας αντίστασης 10KΩ (η αντίσταση pull up). Όταν ο διακόπτης είναι ανοιχτός, το ρεύμα ρέει από τα 5V στη είσοδο του μικροελεγκτή, ορίζοντας κατάσταση υψηλής τάσης (high). Όταν ο διακόπτης κλείσει, το ρεύμα θα περάσει από τα 5V μέσω της αντίστασης στη γείωση, ορίζοντας στον μικροελεγκτή κατάσταση χαμηλής τάσης (low). Εάν η αντίσταση pull up δεν υπήρχε, τότε στο κλείσιμο του διακόπτη θα είχαμε βραχυκύκλωμα μεταξύ των 5V και της γείωσης.



Εικόνα 2.3: Συνδέοντας μία αντίσταση pull up σε ένα κύκλωμα ψηφιακού αισθητήρα (διακόπτη)

Η εναλλακτική περίπτωση σύνδεσης με pull down αντίσταση απεικονίζεται στην Εικόνα 2.4



Εικόνα 2.4: Συνδέοντας μία αντίσταση pull down σε ένα κύκλωμα ψηφιακού αισθητήρα (διακόπτη)

Εισαγωγή στη θεωρία δειγματοληψίας

Τι είναι αναλογικό σήμα; Βασικά ότι αισθανόμαστε στον φυσικό κόσμο μέσω των αισθήσεων μας: ήχος, φώς, χρώματα, μυρωδιές, βασικά τα πάντα! Είμαστε σχεδιασμένοι να δεχόμαστε τα σήματα αυτά και να τα μεταφράζουμε σε κάτι που καταλαβαίνουμε, αλλά τι συμβαίνει με τους υπολογιστές; Για τέτοιες ψηφιακές συσκευές θα πρέπει αν μετατρέψουμε τα αναλογικά σήματα σε ψηφιακά. Ένα ψηφιακό σήμα είναι διακριτό και στο χρόνο και στην ενίσχυση (σε αντίθεση με τα αναλογικά που είναι συνεχή). Για να μετατραπεί ένα αναλογικό σήμα σε ψηφιακή μορφή, ώστε να είναι καταληπτό από έναν μικροεπεξεργαστή, χρησιμοποιούμε μία τεχνική που ονομάζεται “δειγματοληψία”. Δειγματοληψία ονομάζεται η διαδικασία διαίρεσης του σήματος σε πολλά μικρά κομμάτια και η μέτρηση της τιμής κάθε δείγματος. Τότε η κάθε μία ανεξάρτητη τιμή μετατρέπεται σε δυαδική τιμή (ψηφιοποιείται). Αυτή η διαδικασία συνήθως πραγματοποιείται από έναν μικροεπεξεργαστή, χρησιμοποιώντας την μονάδα μετατροπής αναλογικού σε ψηφιακό σήμα (Analog to Digital Converter-ADC) – βλέπε Εικόνα 2.5.

Ο ρυθμός δειγματοληψίας (δηλαδή το πόσο γρήγορα διαιρούμε το συνεχές σήμα και παίρνουμε δείγμα) το ονομάζουμε συχνότητα δειγματοληψίας. Πόσα όμως δείγματα είναι απαραίτητο να πάρουμε ώστε να εξασφαλίσουμε ότι πήραμε όλη την πληροφορία που περιέχεται στο σήμα; Εάν το σήμα περιέχει υψηλές συχνότητες (όπως η ανθρώπινη φωνή) τότε χρειαζόμαστε μεγαλύτερη συχνότητα δειγματοληψίας ώστε να εξασφαλίσουμε ότι δεν θα χαθεί πληροφορία. Σε γενικές γραμμές για να έχουμε την πλήρη πληροφορία ενός σήματος θα πρέπει η συχνότητα δειγματοληψίας να είναι διπλάσια της μέγιστης συχνότητας του σήματος (γνωστό και ως ρυθμός Nyquist).

Μετατροπή A/D

Η μετατροπή A/D αναφέρεται στην μετατροπή ενός σήματος από αναλογικό σε ψηφιακό (μία σειρά από 1 και 0). Αυτό γίνεται από το τμήμα του μΥπολογιστή που ονομάζεται A/D converter και λαμβάνει σήμα από αναλογική πηγή (πχ αισθητήρα). Η ανάλυση του μετατροπέα ορίζεται από τον αριθμό των διακριτών τιμών που μπορεί να παράγει, σε αντιστοιχία με τις αναλογικές τιμές του σήματος. Για παράδειγμα οι αναλογικοί αισθητήρες που χρησιμοποιούμε λειτουργούν στην κλίμακα 0-5V. Η ανάλυση σε αυτή την περίπτωση είναι οι διαφορετικοί ακέραιοι (που συνήθως ξεκινούν από το 0) και αντιπροσωπεύουν αυτή την κλίμακα. Οι τιμές αποθηκεύονται συνήθως ηλεκτρονικά σε δυαδική μορφή, γι’αυτό και η ανάλυσή τους εκφράζεται σε bit. Για παράδειγμα ένας μετατροπέας ADC με ανάλυση 8-bit μπορεί να κωδικοποιήσει ένα σήμα σε ένα σύνολο 256 ξεχωριστών τιμών (2^8). Αυτό σημαίνει ότι το αναλογικό σήμα θα παρουσιαστεί σε μία κλίμακα αριθμών από το 0 έως το 255.

Στην περίπτωση του Arduino Ο εσωτερικός ADC έχει ανάλυση 10-bit, που σημαίνει ότι το αναλογικό σήμα μετατρέπεται σε ακέραιο μεταξύ των τιμών 0 και 255.

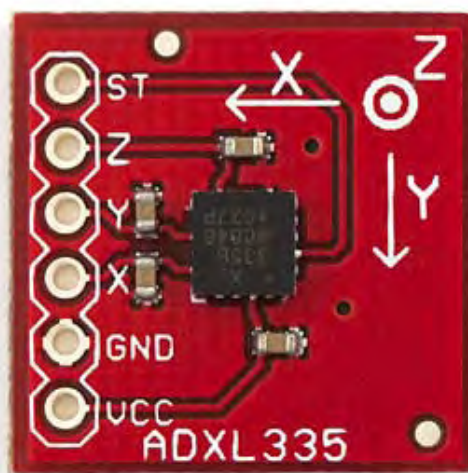
Παραδείγματα και αρχές λειτουργίας αισθητήρων

Αυτό το τμήμα παρουσιάζει τους περισσότερο συνηθισμένους αισθητήρες που χρησιμοποιούνται σε ιδιοκατασκευές και που θα χρησιμοποιήσουμε στις ασκήσεις.

Επιτάχυνση

Ένας από τους πιο συνηθισμένους τρόπους να αντιληφθούμε και να μετρήσουμε την ταχύτητα, είναι χρησιμοποιώντας αισθητήρες επιτάχυνσης. Το επιταχυνσιόμετρο είναι ένας αισθητήρας που χρησιμοποιείται για να μετράει ταχύτητα, επιτάχυνση, κλίση ή κραδασμούς και χρησιμοποιείται στη ρομποτική, την αυτοκινητοβιομηχανία και σε συσκευές όπως κινητά τηλέφωνα και κονσόλες παιχνιδιών.

Τα περισσότερα επιταχυνσιόμετρα είναι Μικρό-Ηλεκτρο-Μηχανικοί Αισθητήρες (Micro-Electro-Mechanical Sensors – MEMS). Καλούνται έτσι επειδή συνδυάζουν ηλεκτρονικά και μηχανικά τμήματα. Εμπεριέχουν μια μικρή μάζα ενσωματωμένη στο ολοκληρωμένο κύκλωμα, στηριγμένη σε μικρές δέσμες. Όταν ο αισθητήρας κινηθεί, η μάζα αυτή κινείται και προκαλεί αλλαγή στη χωρητικότητα του κυκλώματος, προκαλώντας πιεζοηλεκτρικό φαινόμενο που μεταφράζεται σε επιτάχυνση. Η εικόνα 2-7 παρουσιάζει ένα αναλογικό επιταχυνσιόμετρο που μετράει επιτάχυνση στους άξονες X, Y, Z.



Εικόνα 2-7 : Ένας αναλογικός αισθητήρας επιτάχυνσης. Μπορείτε να δείτε τους ακροδέκτες τάσης (Vcc), GND, και τις εξόδους για κάθε άξονα μέτρησης X,Y,Z.

Χωρητικότητα

Επιταχυνσιόμετρα που ενσωματώνουν μέτρηση χωρητικότητας, δίνουν μια έξοδο που εξαρτάται από την απόσταση που έχουν δυο επίπεδες επιφάνειες. Η μία από αυτές τις 'πλάκες' είναι φορτισμένη με ηλεκτρικό φορτίο. Αλλάζοντας το κενό μεταξύ των πλακών μεταβάλλουμε το φορτίο της επιφάνειας, που έχει ως αποτέλεσμα την μεταβολή της τάσης εξόδου. Αυτή η μέθοδος είναι γνωστή για την

ακρίβεια και τη σταθερότητά της. Τα χωρητικά επιταχυνσιόμετρα είναι λιγότερο επιρρεπή στον θόρυβο και στη διαφοροποίηση βάσης της θερμοκρασίας. Επίσης καταναλώνουν μικρότερη ισχύ και έχουν μεγαλύτερο εύρος ζώνης μετρήσεων εξαιτίας του εσωτερικού κυκλώματος ανατροφοδότησης.

Πιεζοηλεκτρισμός

Ο πιεζοηλεκτρικός τρόπος για την αίσθηση της επιτάχυνσης είναι φυσική μέθοδος, καθώς η επιτάχυνση προκαλεί ώθηση (πίεση). Όταν κάποιοι συγκεκριμένοι κρύσταλλοι πιέζονται, προκαλούν φορτίσεις αντίθετης πολικότητας στις αντίθετες πλευρές του κρυστάλλου. Αυτό είναι γνωστό σαν πιεζοηλεκτρικό φαινόμενο. Σε έναν πιεζοηλεκτρικό αισθητήρα, η φόρτιση που προκαλείται ενισχύεται και προωθείται στην έξοδο είτε ως διαφοροποίηση της τάσης, είτε του ρεύματος.

Οι πιεζοηλεκτρικοί επιταχυντές ανταποκρίνονται μόνο σε φαινόμενα AC όπως είναι η δόνηση ή το χτύπημα. Έχουν μεγάλο εύρος ζώνης, αλλά μπορεί να είναι ακριβοί στη περίπτωση που επιλέξουμε ποιοτικούς αισθητήρες.

Οι αισθητήρες επιτάχυνσης τύπου Piezo-film ανταποκρίνονται καλύτερα σε AC φαινόμενα και όχι τόσο σε DC όπως είναι η επιτάχυνση ή η βαρύτητα. Είναι οικονομικοί ως επιλογή και ανταποκρίνονται και σε μια σειρά από άλλα ερεθίσματα, όπως είναι η θερμοκρασία, ο ήχος και η πίεση.

Θερμοκρασία

Όταν αναζητούμε εφαρμογές οικιακού αυτοματισμού ή αισθητήρες Cloud εφαρμογών, διαπιστώνουμε πως ο πιο συνήθης αισθητήρας είναι αυτός της θερμοκρασίας. Ο λόγος γι' αυτό είναι πως είναι πολύ εύκολος να βρεθεί, πολύ οικονομικός στην αγορά, ενσωματώνεται εύκολα σε συστήματα μΥπολογιστών και τέλος, μετρά ένα βασικό μέγεθος : τη θερμοκρασία περιβάλλοντος. Είναι χωρισμένοι σε τρεις βασικές κατηγορίες : θερμοζεύγη, αντίστασης - θερμίστορ και θερμοευαίσθητα ολοκληρωμένα κυκλώματα.

Τα θερμίστορ είναι αυτά που χρησιμοποιούνται πιο συχνά στις ασκήσεις. Διαφοροποιεί την αντίστασή του ανάλογα με τη διαφοροποίηση της θερμοκρασίας. Η βασική κλίμακα λειτουργίας είναι μεταξύ -90°C και 130°C . Υπάρχουν δυο τύποι θερμίστορ : με θετικό θερμοκρασιακό συντελεστή (Positive Temperature Coefficient – PTC) και αρνητικού θερμοκρασιακού συντελεστή (NTC). Όταν η θερμοκρασία αυξάνει, στην πρώτη εκδοχή ο αισθητήρας αυξάνει αντίστοιχα την τιμή της αντίστασής του, ενώ στη δεύτερη εκδοχή τη μειώνει.

Η εικόνα 2-8 δείχνει ένα θερμίστορ. Συνήθως έρχονται με τρεις ακροδέκτες, ο ένας για την τροφοδοσία (Vcc), ο άλλος για τη γείωση και ο τρίτος για την έξοδο, που είναι ανάλογη με τη θερμοκρασία.



Εικόνα 2-8 : Ένας αναλογικός αισθητήρας θερμοκρασίας.

Υγρασία

Με τον όρο υγρασία εννοούμε την ποσότητα νερού που βρίσκεται εξαερωμένο στον αέρα του περιβάλλοντος και μετράται συνήθως με το ποσοστό της σχετικής υγρασίας (relative humidity – RH) που είναι ο λόγος της υγρασίας που υπάρχει στην ατμόσφαιρα, με το συμπιεσμένο επίπεδο υγρασίας στις ίδιες συνθήκες θερμοκρασίας και πίεσης. Σε μια βροχερή μέρα το ποσοστό της υγρασίας θα είναι πάνω από 90%. Το ποσοστό της υγρασίας είναι ένα από τα βασικότερα μεγέθη που εξετάζεται σε οικιακά συστήματα αυτοματισμού.

Η υγρασία μετράται με ειδικά όργανα που καλούνται υγρασιόμετρα και εμπεριέχουν τη χρήση θερμομέτρων 'υγρής σφαίρας' και 'ξηρής σφαίρας'. Στα ηλεκτρονικά κυκλώματα παρόλα αυτά κάνουμε χρήση υγρασιομέτρων που χρησιμοποιούν χωρητικούς αισθητήρες. Αυτοί αλλάζουν χωρητικότητα ανάλογα με τη σχετική υγρασία, σε δυο μη χωρητικές πλάκες. Η αλλαγή της χωρητικότητας είναι συνήθως μεταξύ των τιμών 0,2-0,5pF για μια αλλαγή στο ποσοστό της υγρασίας κατά 1%.

Η εικόνα 2-9 δείχνει έναν αναλογικό αισθητήρα της Sparkfun. Σε αυτή τη περίπτωση ο αισθητήρας παρέχει μια έξοδο στην οποία η τάση διαφοροποιείται ανάλογα με την υγρασία. Υπάρχουν άλλοι αισθητήρες που παρέχουν απευθείας στην έξοδό τους ποσοστό επί τοις εκατό, σε ψηφιακή μορφή.



Εικόνα 2-9 : Ένας αναλογικός αισθητήρας υγρασίας

Απόσταση

Η μέτρηση της απόστασης είναι σημαντική σε συστήματα που περιέχουν παρουσία – κίνηση (συστήματα συναγερμού) και σε συστήματα πλοήγησης (ρομπότ). Υπάρχουν δυο διαφορετικοί αισθητήρες σε αυτή τη περίπτωση : οι υπεριώδεις και των υπερήχων.

Αισθητήρες υπεριωδών ακτινών

Ένας αισθητήρας υπεριωδών (όπως της εικόνας 2-10) είναι ένας αισθητήρας που εκπέμπει μια στενή δέσμη φωτός μέσω ενός LED, και τη λαμβάνει πίσω. Όταν η ακτίνα ληφθεί, ο αισθητήρας μετρά τη γωνία ανάκλασης και υπολογίζει έτσι πόσο κοντά η μακριά είναι το κάθε αντικείμενο. Οι αισθητήρες αυτοί είναι πολύ χαμηλού κόστους, έχουν όμως πολύ σημαντικά μειονεκτήματα : θα πρέπει να υπάρχει μια ελάχιστη απόσταση από το προς μέτρηση αντικείμενο και ταυτόχρονα η ακρίβεια επηρεάζεται πολύ σημαντικά από την απόσταση στην οποία βρίσκεται το αντικείμενο. Όσο πιο μεγάλη είναι η απόσταση, τόσο λιγότερο ακριβής θα είναι η μέτρηση που θα έχετε. Γι'αυτό το λόγο αυτοί οι αισθητήρες χρησιμοποιούνται κυρίως για να μετρηθούν αντικείμενα που βρίσκονται σε απόσταση 10 – 80 cm. Επίσης, επηρεάζονται αρκετά από πηγές θερμοκρασίας και πηγές θορύβου, εκτός από το φως φθορισμού.



Εικόνα 2-10 : Ένας υπέρυθρος αισθητήρας φωτός. Και οι δύο δίοδοι (εκπομπής και λήψης φωτός) διακρίνονται. Το κόκκινο και το μαύρο καλώδιο χρησιμοποιούνται για τη τάση και τη γείωση αντίστοιχα. Το λευκό καλώδιο είναι της τάσης εξόδου, η οποία καθορίζεται από την απόσταση των αντικειμένων.

Αισθητήρες Υπερήχων

Οι αισθητήρες υπερήχων βασίζονται στην καθυστέρηση που υπάρχει όταν ένα ηχητικό σήμα ταξιδεύει : ηχητικά κύματα υψηλών συχνοτήτων (πάνω από τις συχνότητες που μπορεί να ακούσει ο άνθρωπος, γι'αυτό και το 'υπέρ' του ονόματος) εκπέμπονται από μια πηγή και λαμβάνονται πίσω από έναν δέκτη (και τα δυο αποτελούν τμήματα του αισθητήρα, όπως στους αισθητήρες IR). Ο ολικός χρόνος ταξιδιού χρησιμοποιείται για να καθορίσει την απόσταση ανάμεσα στον

αισθητήρα και στο αντικείμενο που στοχεύει. Ένα μεγάλο πλεονέκτημα των αισθητήρων αυτών είναι ότι δεν χρειάζονται άμεση επαφή με το αντικείμενο, παρόλα αυτά είναι ευαίσθητοι στη θερμοκρασία, τις αντανάκλασεις και τη γωνία με την οποία βλέπουν το αντικείμενο.

Για να παραχθούν υπέρηχοι, εφαρμόζεται ένα ταχύτατα ταλαντευόμενο σήμα σε έναν πιεζοηλεκτρικό κρύσταλλο που περιέχουν. Το σήμα αυτό αναγκάζει τον κρύσταλλο να διαστέλλεται και να συστέλλεται, παράγοντας τους επιθυμητούς ήχους. Ο δέκτης μετατρέπει τον ήχο που λαμβάνει πάλι σε τάση και από τον χρόνο που χρειάζεται για να ολοκληρωθεί η διαδικασία αυτή (μεταφορά και αντανάκλαση του ήχου), μπορεί να υπολογίσει την απόσταση (χρειάζεται περίπου 6msec για να ταξιδέψει ο ήχος 1 μέτρο).

Οι αισθητήρες υπερήχων απαντώνται συνήθως σε συστήματα αυτοματισμού και αυτοκινήτου, όπως ρομπότ, μετρητές απόστασης, κ.ά. Η εικόνα 2-11 δείχνει έναν τέτοιο αισθητήρα που μπορεί να συνδεθεί απευθείας με την πλακέτα Arduino. Οι μονάδες εκπομπής και λήψης φαίνονται καθαρά.



Εικόνα 2-11 : Ένας αισθητήρας υπερήχων από την Parallax. Είναι πολύ συνηθισμένη μονάδα που χρησιμοποιείται στη ρομποτική και μπορεί εύκολα να τοποθετηθεί σε μια πλακέτα.

Φως

Ένας αισθητήρας LDR (light depended resistor - αντιστάτης εξαρτώμενος από το φως), γνωστός επίσης και ως φωτοαντίσταση ή photocell, είναι ουσιαστικά μια μεταβλητή αντίσταση που έχει την ιδιότητα να μεταβάλλει την τιμή της ανάλογα με την πυκνότητα του περιβάλλοντος φωτισμού. Αυτό σημαίνει ότι τοποθετώντας την σε ένα απλό κύκλωμα που θα έχει την δυνατότητα αναγνώρισης των μεταβολών τάσης μπορείτε να ανιχνεύσετε και να εκτιμήσετε τις μεταβολές που προκαλούνται στον φωτισμό.

Οι αισθητήρες LDR κατασκευάζονται από υψηλής αντίστασης ημιαγώγιμα υλικά. Η αρχή λειτουργίας τους έγκειται στο ότι όταν απορροφώνται τα φωτόνια από το υλικό επηρεάζουν τα ηλεκτρόνια του υλικού, μεταβάλλοντας έτσι την αντίστασή του.

Οι φωτοαντιστάσεις χρησιμοποιούνται ευρύτατα για την ανίχνευση καπνού και φωτιάς (μιας και ο καπνός και η υψηλή θερμοκρασία επηρεάζουν την αντίσταση της επαφής). Είναι απλοί στην κατασκευή και χαμηλού κόστους αισθητήρες, γι' αυτό και συναντώνται σε μια σειρά από εφαρμογές αυτοματισμού, όπως συναγερμοί, συστήματα φωτισμού, κ.ά. Στην εικόνα 2-12 μπορείτε να δείτε με τι μοιάζει μια φωτοαντίσταση.

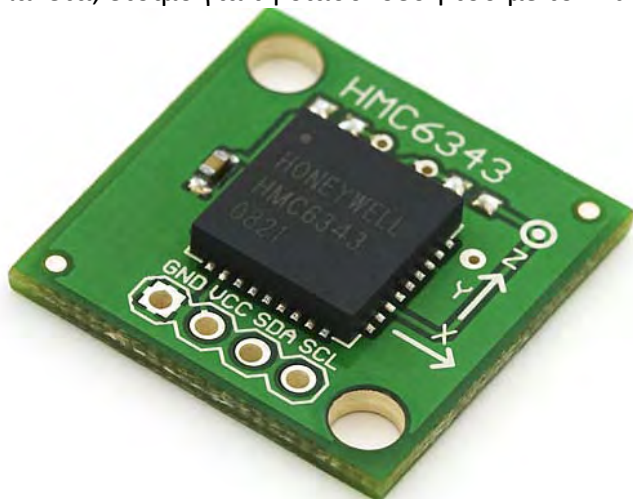


Εικόνα 2-12 : Ένας αισθητήρας LDR. Μπορεί να συνδεθεί με τον ίδιο τρόπο που συνδέεται μια αντίσταση. Η τιμή της αντίστασης αλλάζει εξαρτώμενη από τις συνθήκες φωτισμού που επηρεάζουν το φορτίο που ρέει διαμέσου του κυκλώματος.

Προσανατολισμός

Όπως γνωρίζετε, η πυξίδα είναι ένα όργανο πλοήγησης ευαίσθητο στο μαγνητικό πεδίο της γης. Η τυπική πυξίδα ενσωματώνει μια μαγνητική βελόνα που ευθυγραμμίζεται κάθε φορά με τον μαγνητικό βορά της γης, προσδιορίζοντας με αυτόν τον τρόπο τον προσανατολισμό.

Μια ηλεκτρονική ή μια ψηφιακή πυξίδα είναι είτε ένα μαγνητόμετρο είτε ένας γυροσκοπικός μηχανισμός οπτικής ίνας που ανιχνεύει μαγνητικά πεδία χωρίς κινούμενα μέρη. Οι πιο συνηθισμένοι αισθητήρες ενσωματώνουν στοιχεία Hall. Αυτά παράγουν μια τάση ανάλογη του μαγνητικού πεδίου που τα περιβάλλει και ανιχνεύουν και πολικότητα. Μια ψηφιακή πυξίδα δεν είναι ελεύθερη για περιστροφή όπως οι αναλογικές, αλλά είναι σταθερά τοποθετημένη σε μια πλακέτα. Η εικόνα 2-13 δείχνει έναν τυπικό αισθητήρα προσανατολισμού πάνω σε πλακέτα, έτοιμο για τη διασύνδεσή του με το Arduino.



Εικόνα 2-13 : Μια ηλεκτρονική ψηφιακή πυξίδα. Η πληροφορία σχετικά με τον προσανατολισμό δίνεται σε μορφή bit στην έξοδο του που συνδέεται με τον μΥπολογιστή. Ο κατάλληλος κώδικας είναι απαραίτητος ώστε η ακολουθία δεδομένων να μετατραπεί σε χρήσιμη πληροφορία.

Ήχος

Ο αισθητήρας που χρησιμοποιούμε για την καταγραφή του ήχου, καλείται μικρόφωνο. Μπορεί να ταξινομηθεί σε διάφορους τύπους, όπως ηλεκτροστατικό, δυναμικό ή πιεζοηλεκτρικό, εξαρτώμενο από τον τύπο και τον τρόπο της μετατροπής από ήχο σε ρεύμα.

Στις ασκήσεις μας συνήθως χρησιμοποιούμε μικρόφωνα ώστε να εντοπίσουμε την ύπαρξη ήχων στο περιβάλλον. Μπορούμε να χρησιμοποιήσουμε ήχο ώστε να ενεργοποιήσουμε συσκευές (π.χ. παλαμάκια που ανοίγουν έναν διακόπτη), υπερήχους που θα ελέγχουν σερβομηχανισμούς (ώστε να μην υπάρχει παρεμβολή από τους ήχους του περιβάλλοντος) ή ακόμα και να χρησιμοποιούμε αναγνώριση φωνής ώστε να ελέγχουμε τις εξόδους του μΥπολογιστή μας. Στην εικόνα 2-14 μπορείτε να δείτε ένα τυπικό παράδειγμα ενός τέτοιου αισθητήρα με μικρόφωνο που μπορεί να συνδεθεί απευθείας στην πλακέτα σας.



Εικόνα 2-14 : Ένας αισθητήρας μικροφώνου ενσωματωμένος σε πλακέτα (ιδιοκτησία εικόνας Seeedstudio)

Ηλεκτρικό ρεύμα

Ο αισθητήρας ρεύματος είναι ένας μετασχηματιστής ηλεκτρικού φορτίου (current transformer - CT) που ανιχνεύει την ύπαρξη ηλεκτρικού ρεύματος (AC ή DC) σε ένα καλώδιο και παράγει ένα ανάλογο σήμα. Το παραγόμενο σήμα μπορεί να είναι αναλογικό ρεύμα ή τάση, αλλά μπορεί επίσης να είναι και μια ψηφιακή έξοδος.

Το πιο κοινό σχέδιο ενός αισθητήρα ρεύματος αποτελείται από ένα μακρύ καλώδιο τυλιγμένο πολλές φορές γύρω από ένα δακτύλιο σιλικόνης, ενδιάμεσα του οποίου διέρχεται το κύκλωμα που θέλουμε να μετρήσουμε.



Εικόνα 2-15 : Ένας αναλογικός αισθητήρας τύπου clamp. Πιάνεται γύρω από ένα καλώδιο που διαρρέεται από ρεύμα και μπορεί να ενσωματωθεί στο κύκλωμα με

τον ίδιο τρόπο που ενσωματώνεται μια αντίσταση. Η αντίσταση του αισθητήρα διαφοροποιείται ανάλογα με το ρεύμα που διαρρέει το καλώδιο (ιδιοκτησία εικόνας Seeedstudio)

Η πιο χρήσιμη εφαρμογή τέτοιων αισθητήρων είναι η μέτρηση της ενέργειας που καταναλώνει μια ηλεκτρική συσκευή. Για να γίνει αυτό, χρειαζόμαστε μια διάταξη που να μετρά επίσης και την τάση, καθώς και τον απαραίτητο κώδικα που θα υπολογίζει τα watt. Οδηγίες γι' αυτό και τα απαραίτητα κυκλώματα μπορείτε να βρείτε στη διεύθυνση του OpenEnergyMonitorProject (<http://openenergymonitor.org>).

Ενεργοποιητές – Διακόπτες

Όπως υποδεικνύει το όνομα τους πρόκειται για συσκευές που βασίζουν τη λειτουργία τους σε κάποιες συνθήκες ή ένα συγκεκριμένο γεγονός που καλείται σκανδαλισμός (trigger). Υπάρχουν πολλά είδη ενεργοποιητών, αλλά εμείς θα αναφερθούμε σε αυτούς που χρησιμοποιούνται περισσότερο σε ιδιοκατασκευές και φυσικά, στις ασκήσεις του παρόντος βιβλίου. Θα αναφερθούμε δηλαδή στα ρελέ (relay switches) που μας επιτρέπουν τον χειρισμό κυκλωμάτων εξ' αποστάσεως (π.χ. το άναμμα των φώτων), και τα μοτέρ σέρβο (servo motors), που εκτελούν απλές κινήσεις (π.χ. να αλλάξουν την κατεύθυνση μιας webcam).

Διακόπτες ρελέ

Το ρελέ είναι κατά βάση ένας ηλεκτρονικά ενεργοποιούμενος διακόπτης. Το ρεύμα διαρρέει το πηνίο του ρελέ δημιουργώντας μαγνητικό πεδίο το οποίο ελκύει έναν ακροδέκτη, αλλάζοντας έτσι τις επαφές του διακόπτη. Το ρεύμα στο πηνίο μπορεί να έχει δυο καταστάσεις, ON ή OFF, οπότε αντίστοιχα και το ρελέ έχει δύο θέσεις στον διακόπτη του, ενώ πολλά ρελέ έχουν διπλούς εσωτερικούς διακόπτες. Τα ρελέ χρησιμοποιούνται συνήθως έτσι ώστε να μπορεί ένα κύκλωμα να διακόψει ή να ενεργοποιήσει ένα άλλο. Έτσι, ένα κύκλωμα μπαταρίας μπορεί να ελέγχει ένα άλλο κύκλωμα 220V. Δεν υπάρχει καμία άμεση επαφή μεταξύ των δύο κυκλωμάτων, η λειτουργία είναι μόνο μαγνητική και μηχανική.



Εικόνα 2-16 : Ένα τυπικό ρελέ. Συνήθως έχουν 4 ακροδέκτες

Το ρεύμα που διαρρέει το πηνίο του ρελέ είναι συνήθως υψηλό, τυπικά 30mA για ένα ρελέ 12V. Παρόλα αυτά το ρεύμα που θα χρειαστείτε για ένα μικρό ρελέ ιδιοκατασκευών είναι ακόμη μεγαλύτερο και μπορεί να φτάσει και τα 100mA. Τέτοιο ρεύμα δεν έχουν την ικανότητα να παρέχουν τα ολοκληρωμένα κυκλώματα. Γι'αυτόν το λόγο, χρησιμοποιούμε συνήθως ένα τρανζίστορ ανάμεσα στο ολοκληρωμένο κύκλωμα και το ρελέ για να ενισχύει το ρεύμα που φθάνει στο πηνίο του ρελέ.

Οι διακόπτες ρελέ μπορούν να έχουν πολλά σετ επαφών στο ίδιο κέλυφος, όπως π.χ. ρελέ με τέσσερα σετ επαφών. Τα περισσότερα ρελέ είναι σχεδιασμένα για τοποθέτηση σε πλακέτα, αλλά μπορείτε να κολλήσετε απευθείας τους ακροδέκτες τους με καλώδιο, προσέχοντας να μη λιώσει το κέλυφός τους.

Τα ρελέ μπορούν να παράγουν μια αιχμή ρεύματος τη στιγμή που ενεργοποιούνται, η οποία μπορεί να καταστρέψει τα τρανζίστορ και τους μικροελεγκτές ενός κυκλώματος. Για να αποφύγουμε τέτοιες βλάβες, τοποθετούμε μια δίοδο προστασίας στα άκρα του πηνίου του ρελέ.

Σερβομηχανισμοί (servo motors)

Ένας σερβομηχανισμός είναι παρόμοιος με ένα μοτέρ, με τη διαφορά ότι ο άξονάς του δεν περιστρέφεται συνεχώς, αλλά σε πολύ συγκεκριμένες γωνίες, εξαρτώμενες από το σήμα εισόδου. Συνήθως μπορεί να κινηθεί μεταξύ 0 και 180 μοιρών. Η τεχνική για να επιτευχθεί αυτό λέγεται διαμόρφωση με κωδικοποίηση παλμού (Pulse Coded Modulation) και η διάρκεια του παλμού καθορίζει το ποσό περιστροφής του άξονα. Ο σερβομηχανισμός λαμβάνει παλμό κάθε 20 milliseconds. Επομένως, ένας παλμός διάρκειας 1,5 milliseconds θα περιστρέψει τον άξονα του μηχανισμού κατά 90 μοίρες.

Ένας τυπικός σερβομηχανισμός απεικονίζεται στην εικόνα 2-17. Έρχεται με 3 ακροδέκτες, οι δύο για την παροχή της κατάλληλης τάσης και γείωσης και ο άλλος για το παλμικό σήμα που θα τον οδηγήσει. Το Arduino ενσωματώνει το κατάλληλο hardware και τις απαραίτητες βιβλιοθήκες ώστε να μπορεί να οδηγήσει τέτοιου είδους σερβομηχανισμούς.



Εικόνα 2-17 : Ένα τυπικό μοτέρ σέρβο για εφαρμογές οικιακού αυτοματισμού. Μπορεί να ελέγξει μικρούς ρομποτικούς βραχίονες ή να περιστρέψει οπτικά αντικείμενα (ιδιοκτησία εικόνας Seeedstudio).

Συμπεράσμα

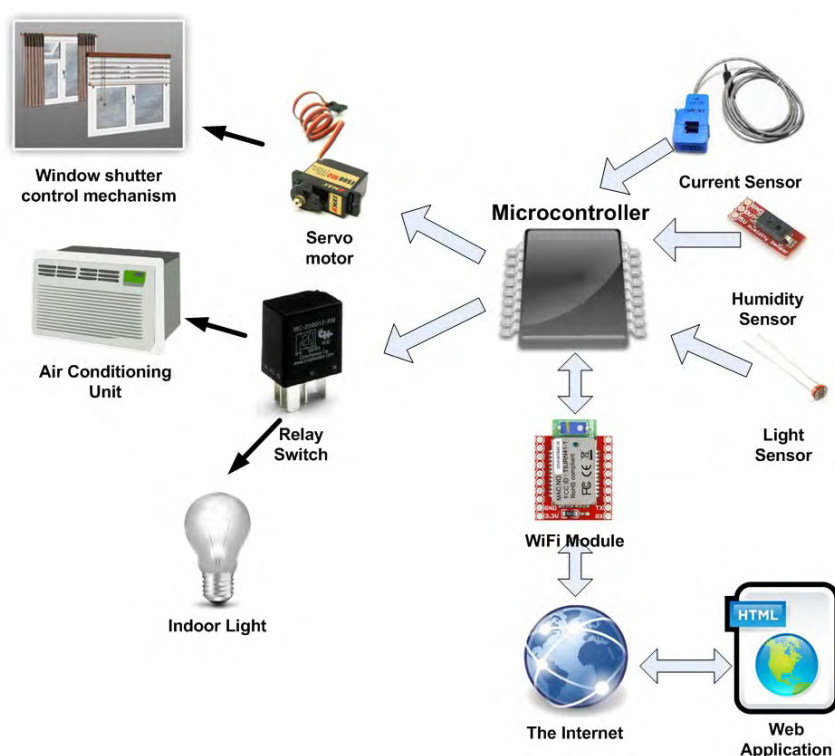
Σε αυτό το κεφάλαιο είδαμε τις συνηθέστερες μορφές αισθητήρων και ενεργοποιητών που υπάρχουν διαθέσιμοι και που μπορεί να χρησιμοποιήσετε σε εφαρμογές. Έχοντας συζητήσει και τις υπόλοιπες τεχνολογίες που επιτρέπουν να συνδέσετε τους διάφορους αυτοματισμούς στο internet, τι θα λέγατε να μπορούσαν να ενσωματωθούν όλα αυτά σε ένα ενιαίο σύστημα που αισθάνεται, αλληλεπιδρά με το περιβάλλον και έχει τη δυνατότητα να επικοινωνεί με άλλες συσκευές και με το internet;

Ας δούμε το παράδειγμα της εικόνας 2-18. Θα μπορούσατε να χρησιμοποιήσετε έναν αισθητήρα υγρασίας, έναν αισθητήρα φωτός και έναν ρεύματος ώστε να μετρήσετε τις συνθήκες του δωματίου που βρίσκεστε και την κατανάλωση ρεύματος των συσκευών που υπάρχουν. Αυτοί οι αισθητήρες μπορούν να συνδεθούν στις εισόδους του μικροελεγκτή σας. Επιπλέον θα μπορούσατε να συνδέσετε και μια σειρά από ρελέ που θα μπορούσαν να ελέγχουν μια λάμπα ή το air condition που υπάρχει στο δωμάτιο, ή ακόμα και έναν σερβομηχανισμό που ελέγχει τα σκίαστρα των παραθύρων. Μια μονάδα WiFi μπορεί να στέλνει τα δεδομένα αυτά απευθείας στο internet.

Θα μπορούσατε λοιπόν να προγραμματίσετε το σύνολο αυτών των συσκευών μέσω του Arduino σας, ώστε να ανοίγει π.χ. το air condition όταν η υγρασία του δωματίου ξεπεράσει ένα όριο, να ανοίγει και να κλείνει τα φώτα βασιζόμενο στις συνθήκες φωτισμού ή να ανοίγει ή αν κλείνει τα σκίαστρα των παραθύρων ανάλογα με το εξωτερικό φως. Ο αισθητήρας φορτίου θα μπορούσε να ελέγχει παράλληλα την ενέργεια που καταναλώνεται από όλες τις πιο πάνω συσκευές.

Το πιο ενδιαφέρον κομμάτι όμως είναι όταν το σύστημα αυτό θα συνδεθεί στο WiFi και όλο αυτό στο internet μέσω της μονάδας WiFi. Με αυτόν τον τρόπο ο μικροελεγκτής θα μπορεί να στείλει όλες τις μετρήσεις μέσω internet σε μια web εφαρμογή, ώστε να έχετε πάντα πρόσβαση από εκεί. Αυτό επιτρέπει τις ακόλουθες επιλογές :

- Μπορείτε να έχετε τις τιμές της κατανάλωσης καθώς και της κατάστασης που επικρατεί στο δωμάτιο χρησιμοποιώντας οποιαδήποτε συσκευή που έχει πρόσβαση στο internet.
- Να τροποποιήσετε τα όρια στα οποία ελέγχονται τα φώτα και ενεργοποιείται το air condition.
- Μπορείτε ανά πάσα στιγμή να στείλετε εντολή ώστε να διακοπεί η λειτουργία των συσκευών, όπως π.χ. των φώτων.



Εικόνα 2-18 : Συνδέοντας τα πάντα μαζί : μικροϋπολογιστής, αισθητήρες, ενεργοποιητές, ασύρματες μονάδες και internet

Αναλογικοί Αισθητήρες

Η τοποθέτηση των εισόδων των αναλογικών αισθητήρων εξαρτάται από το είδος της πλακέτας που χρησιμοποιείτε. Στο Arduino Uno βρίσκονται τοποθετημένοι στην κάτω δεξιά γωνία της πλακέτας (Εικόνα 5.1), και ονομάζονται A0-A5.



Εικόνα 5.1. Οι 6 αναλογικοί εισοδοί της πλακέτας Arduino Uno

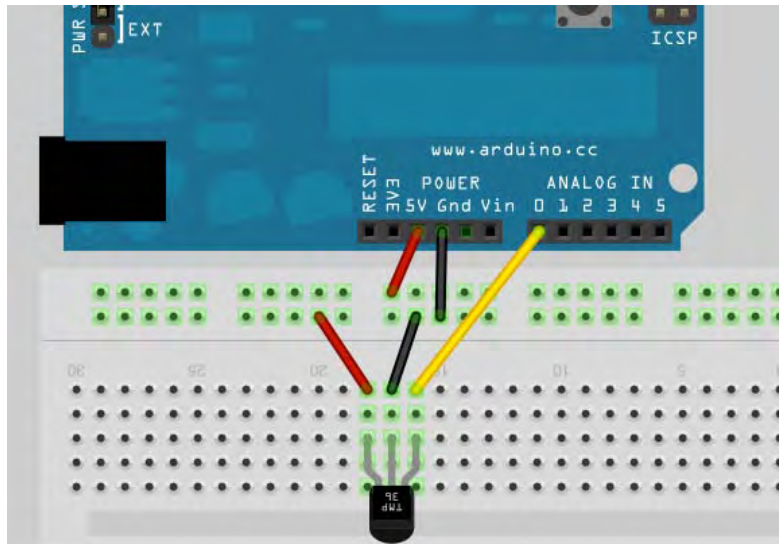
Ο καλύτερος τρόπος για να συνδέσετε έναν αισθητήρα είναι να χρησιμοποιήσετε ένα breadboard και ένα jumper. Τα jumper τροφοδοτούν αρκετά καλά την πλακέτα Arduino και μπορούν να σας εξοικονομήσουν πολύτιμο χρόνο και προσπάθεια από το να κόβετε μικρά καλώδια και να τα κολλάτε μεταξύ τους.

Συνήθως ένας αναλογικός αισθητήρας θα πρέπει να συνδεθεί με μία πηγή ενέργειας αλλά και να γειωθεί έτσι ώστε να μπορεί να σχηματίσει ένα κύκλωμα. Οι αναλογικοί αισθητήρες έρχονται συνήθως σε δύο παραλλαγές ανάλογα με τα pins εξόδου. Έχουν είτε μόνο δύο pins (όπως αντιστάσεις) ή τρία pins για την σύνδεση στην τάση, γειώνοντας και παρέχοντας την αναλογική έξοδο. (ανατρέξτε στο Κεφάλαιο 2 και δείτε μερικά παραδείγματα). Οι περισσότερες πλακέτες Arduino διαθέτουν δύο εξόδους τάσης, μία 5V και μία 3.3V και συνήθως τρεις συνδέσεις γείωσης (σημειωμένες ως GND).

Πρέπει να είστε προσεκτικοί με την τάση λειτουργίας των αισθητήρων. Παροχή 5V σε 3.3V αισθητήρα λειτουργίας μπορεί να τον καταστρέψει!

Όταν οι αναλογικοί αισθητήρες έρχονται με 3 συνδέσεις pin, είναι πιο βολικό, δεδομένου ότι μπορούν να συνδεθούν απευθείας με την πλακέτα Arduino. Ένα τέτοιο παράδειγμα με τον αισθητήρα θερμοκρασίας TMP36 φαίνεται στην Εικόνα 5.2

Θα παρατηρήσετε μια απόκλιση των τιμών ανάλογα με τη θερμοκρασία



Εικόνα 5.3. Σύνδεση του αισθητήρα θερμοκρασίας TMP36 στην πλακέτα Arduino με τη βοήθεια ενός breadboard.

Για να μετατρέψετε τις τιμές της θερμοκρασίας θα πρέπει πρώτα να μετατρέψετε το εύρος 0-1203 σε τάση. Για να γίνει αυτό πρέπει να πολλαπλασιαστεί η τιμή του αισθητήρα που διαβάζουμε με 5 ή με 3.3 (ανάλογα με την τάση που παρείχαμε στον αισθητήρα μέσω της πλακέτας, στην παραπάνω εικόνα συνδέεται με 5V) και να διαιρεθεί το αποτέλεσμα με το 1024:

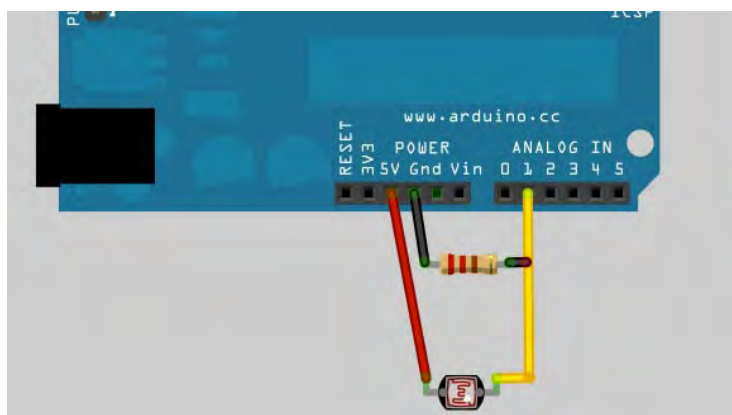
```
float voltage = reading * 5.0;  
voltage /= 1024.0;
```

Σύμφωνα με το datasheet του αισθητήρα θερμοκρασίας, κάθε μεταβολή του βαθμού της θερμοκρασίας αντιστοιχεί σε αλλαγή των 10mV και αντισταθμίζεται με 500mV. Έτσι, προκειμένου να πάρετε βαθμούς Κελσίου από την ανάγνωση της τάσης θα πρέπει να αφαιρέσετε το 0,5 offset και να πολλαπλασιάσετε το αποτέλεσμα επί 100.

```
float temperature = (voltage - 0.5) * 100 ;
```

Ας δούμε τώρα ένα άλλο παράδειγμα με έναν αναλογικό αισθητήρα, ο οποίος έχει δύο σύνδεσεις pin. Για αυτή την περίπτωση θα χρησιμοποιήσετε ένα light dependent resistor (LDR) για να καταγράφει τις συνθήκες φωτισμού. Όπως υποδηλώνει το όνομα, είναι μια αντίσταση και συνεπώς, συνδέεται σε ένα κύκλωμα όπως μια αντίσταση, έχει δύο pins και δεν έχει πολικότητα.

Συνδέστε το ένα pin του LDR στην έξοδο 5V της πλακέτας Arduino και το άλλο pin σε μια αναλογική θύρα εισόδου, π.χ την θύρα A1 όπως στην Εικόνα 5.4 Για να γίνει ένα σωστό κύκλωμα και να έχετε ροή ηλεκτρικού ρεύματος μέσω της LDR θα πρέπει να συνδέσετε επίσης τη γείωση. Για να γίνει αυτό μπορείτε να προσθέσετε μια pull-down αντίσταση των 10K Ohm μεταξύ του ακροδέκτη του LDR που πηγαίνει στην A1 είσοδο και στην θύρα GND της πλακέτας Arduino.



Εικόνα 5.4. Σύνδεση του αισθητήρα LDR απευθείας στην πλακέτα Arduino χρησιμοποιώντας μία pull-down αντίσταση.

Για τη μέτρηση των διακυμάνσεων της LDR, σύμφωνα με τις συνθήκες φωτισμού στο δωμάτιό σας, απλά χρησιμοποιήστε τον κώδικα της Εισαγωγής 5-1, αλλά θυμηθείτε να αλλάξετε την αναλογική θύρα εισόδου σε όποια έχετε συνδέσει την LDR σας (A1 στην περίπτωση του κυκλώματος του Σχήματος 5.4). Για να κάνετε την ανάγνωση αυτού του αποτελέσματος σε τιμές φωτισμού (ονομάζονται lux) θα πρέπει να συμβουλευτείτε το datasheet του αισθητήρα LDR σχετικά με τον τρόπο μετατροπής από τιμές αντίστασης σε τιμές lux.

Ενσωμάτωση των αισθητήρων σε κύκλωμα

Όταν θελήσετε να ξεκινήσετε το δικό σας project πιθανότατα θα αναζητήσετε τους διαθέσιμους αισθητήρες για το κύκλωμά σας. Θα παρατηρήσετε τότε πως η πλειονότητά τους είναι η ψηφιακοί, ενώ ακόμη και οι αναλογικοί έχουν εκδόσεις με ψηφιακή έξοδο. Πώς και υπάρχουν τόσοι πολλοί ψηφιακοί αισθητήρες, όταν τα φαινόμενα που μετρούν είναι φυσικά μεγέθη (όπως θερμοκρασία, υγρασία, απόσταση, πίεση κ.α) ;

Η απάντηση είναι πως οι ψηφιακοί αισθητήρες έχουν ενσωματωμένα κυκλώματα που κάνουν τη μαθηματική δουλειά για εσάς, δίνοντάς σας απευθείας τη μέτρηση της ποσότητας (π.χ βαθμούς Κελσίου αντί για τάση) και σε πολλές περιπτώσεις θα ανακαλύψετε ότι μπορούν να ενσωματώσουν πολλές τιμές σε ένα εξάρτημα (π.χ αισθητήρας που μετρά θερμοκρασία μαζί με υγρασία). Επιπρόσθετα είναι πιο ακριβείς και λιγότερο επιρρεπείς σε λαθος μετρήσεις. Ακόμα μπορούν να περιέχουν χρήσιμες

πληροφορίες στην εσωτερική τους μνήμη (όπως π.χ μετρήσεις καλιμπραρίσματος), ώστε να χρησιμοποιήσετε δικό σας κώδικα για να πάρετε τις μετρήσεις ενός φαινομένου.

Οι ψηφιακοί αισθητήρες εμπίπτουν σε διάφορες κατηγορίες, όχι βασιζόμενες στα pin εξόδου, αλλά στο πρωτόκολλο που χρησιμοποιούν για επικοινωνία.

Αρχικά ας δούμε πως μπορούν να συνδεθούν στην πλακέτα του Arduino. Η Εικόνα 5.5 απεικονίζει το πάνω τμήμα της πλακέτας Arduino Uno που περιέχει 14 διαθέσιμα digital ports. Όπως αναφέρθηκε πριν, ο αριθμός των διαθέσιμων ψηφιακών εισόδων και η τοποθέτηση τους εξαρτάται από την πλακέτα την οποία χρησιμοποιείται. Για παράδειγμα η πλακέτα Arduino mega 2650 έχει 54 ψηφιακά ports.



Εικόνα 5.5 : Τα 14 ψηφιακά ports στο Arduino Uno. Παρατηρήστε ότι τα δύο πρώτα αφορούν τις ψηφιακές επικοινωνίες.

Οι ψηφιακοί αισθητήρες μπορούν να κατηγοριοποιηθούν σε αυτούς που έχουν ON/OFF καταστάσεις όπως κουμπιά, αυτούς που έχουν σειριακό πρωτόκολλο επικοινωνίας, αυτούς που χρησιμοποιούν το πρωτόκολλο I2C, και τέλος αυτούς που χρησιμοποιούν το SPI πρωτόκολλο επικοινωνίας.

Ας δούμε στις επόμενες ενότητες μερικά παραδείγματα για την κάθε κατηγορία.

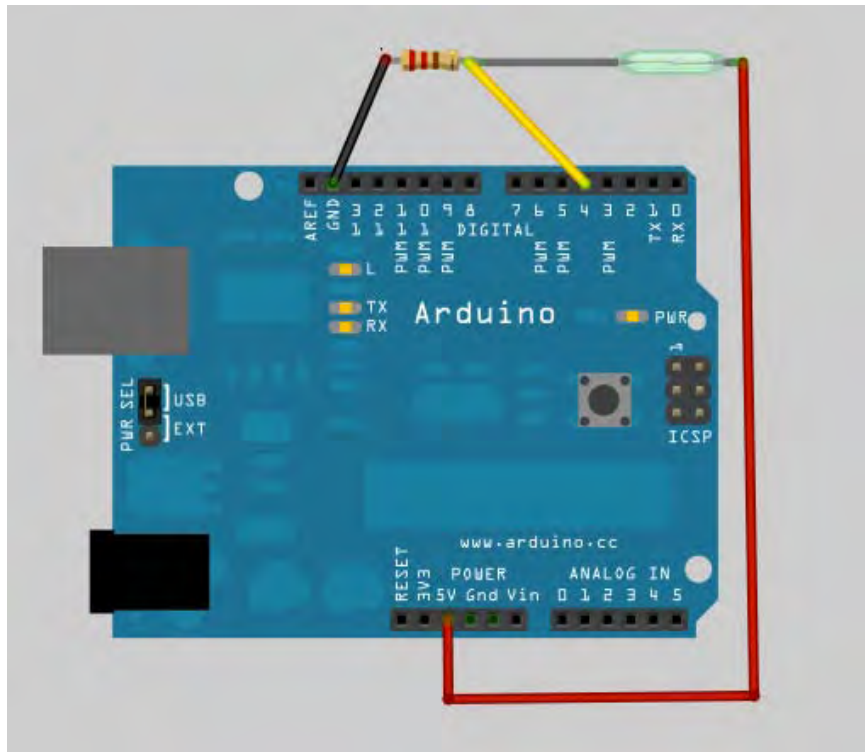
Αισθητήρες με καταστάσεις ON/OFF

Η πρώτη κατηγορία αφορά αισθητήρες που έχουν push buttons ή διακόπτες που δημιουργούν καταστάσεις ON/OFF. Μπορεί να φαίνονται αχρείαστοι σε πιο εξειδικευμένες καταστάσεις, αλλά μπορεί απλά κάποιος να θέλει να ελέγχει πότε η εξώθυρά του είναι ανοιχτή ή όχι και πόσο συχνά.

Για να καταλάβετε πώς θα δημιουργήστε έναν τέτοιο διακόπτη, θα πρέπει να δείτε τη λειτουργία ενός Reed διακόπτη και να κατανοήσετε πως παρουσιάζονται οι καταστάσεις ON/OFF. Ένας διακόπτης Reed είναι ένας διακόπτης που ενεργοποιείται όταν εκτεθεί σε μαγνητικό πεδίο. Σκεφτείτε τον αισθητήρα που υπάρχει στις πόρτες ενός σπιτιού που διαθέτει συναγερμό. Ο μαγνήτης είναι σταθερά κολλημένος στην πόρτα, ενώ το σταθερό τμήμα βρίσκεται πάνω στον σκελετό της πόρτας. Έτσι έχουμε έναν απλό διακόπτη που δείχνει πότε η πόρτα είναι ανοιχτή και πότε όχι.

Γι' αυτό το παράδειγμα μπορείτε να χρησιμοποιήσετε έναν απλό διακόπτη ή ένα ON/OFF κουμπί ως εναλλακτικό σε έναν Reed διακόπτη. Συνδέστε τον όπως στη Εικόνα 5.6

και βεβαιωθείτε ότι χρησιμοποιείτε και μία αντίσταση (περίπου 220 Ohm) ανάμεσα στο διακόπτη και στη σύνδεση GND (όπως αναφέρθηκε το Arduino έχει εσωτερικές pull up resistors που μπορούν επίσης να προστατεύσουν από το βραχυκύκλωμα μεταξύ 5V και GND όταν ο διακόπτης είναι στο On, αλλά παρόλα αυτά είναι καλή πρακτική να χρησιμοποιείτε αντίσταση ούτως ή άλλως).



Εικόνα 5.6: Το σχηματικό για την ανάγνωση ενός Reed Switch

Φτιάξτε το σχήμα στην πλακέτα και ανοίξτε το Serial Monitor.

Κείμενο 5.2: Διαβάστε τις καταστάσεις ενός Reed Switch και τυπώστε τις στο Serial Monitor.

```
// The pin the reed switch is attached to
int reedswitch = 4;
// A variable for reading the switch state
int switchState = 0;
void setup() {
  Serial.begin(9600);
  // Initialize the reed switch pin as an input:
  pinMode(reedswitch, INPUT);
}
void loop(){
  // Read the state of the switch
  switchState = digitalRead(reedswitch);
  // Check if the state is HIGH or LOW (switch activated/deactivated)
  if (switchState == HIGH) {
    Serial.println("Switch is activated!");
  }
}
```



```
}  
else {  
  Serial.println("Switch is deactivated!");  
}  
}
```

Πλησιάστε ένα μαγνήτη κοντά στον διακόπτη (ή ανοιγοκλείστε τον διακόπτη) και παρακολουθήστε την έξοδό του στο Serial Monitor.

Θα παρατηρήσετε πως οι διαφορές στο παράδειγμα με το κείμενο 5.1 είναι δύο:

1. Ορίζεται ότι το συγκεκριμένο digital port θα είναι είσοδος. Αυτό συμβαίνει γιατί τα digital ports είναι γενικά ports εισόδου-εξόδου, οπότε μπορούν να χρησιμοποιηθούν και για να παράγουν σήματα και για να διαβάζουν. Γι'αυτό το λόγο χρειάζεται να οριστεί η χρήση τους.
2. Η διαδικασία για την ανάγνωση από ένα digital port είναι:

digitalRead(reedswitch);

Το υπόλοιπο του κώδικα είναι ίδιο όπως όταν διαβάζουμε τιμές αναλογικών αισθητήρων. Οι αισθητήρες ON-OFF κατάστασης είναι γενικά απλοί, αφού ενεργοποιούνται μόνο ένα event στο digital port. Όταν φτάνει η ώρα για πιο σύνθετους αισθητήρες που παράγουν ακολουθία ON-OFF καταστάσεων, τότε απαιτείται επιπρόσθετος κώδικας, όπως θα αντιληφθείτε στη συνέχεια.

ΚΕΦΑΛΑΙΟ 5 Διαχείριση δεδομένων από αισθητήρες περιβάλλοντος

Το πρώτο σας project εξερεύνησης των data services πλατφόρμων που υπάρχουν θα αφορά στην αποθήκευση της εσωτερικής θερμοκρασίας, της υγρασίας και των συνθηκών φωτισμού που έχουν εισαχθεί στο Arduino που χρησιμοποιείτε. Τα δεδομένα που έχουν συλλεχθεί θα σταλούν στην εφαρμογή που διαχειρίζεται τα δεδομένα (IoT) και απο εκεί θα γίνει η οπτικοποίηση των δεδομένων και η αποθήκευσή τους σε ιστορικό. Θα διαχωρίσετε το project σε τρία μέρη : στο πρώτο βήμα απαιτείται η κατασκευή του κυκλώματος, στο δεύτερο αφορά στο στήσιμο του κατάλληλου περιβάλλοντος στο λογισμικό και τέλος, στο τρίτο βήμα θα ενσωματώσετε τον κώδικα του Arduino που απαιτείται ώστε να συλλέγουν και να μεταδοθούν τα δεδομένα του αισθητήρα.

Όπως περιγράφηκε στα προηγούμενα κεφάλαια, υπάρχουν πολλοί τρόποι σύνδεσης του Arduino στο internet, οπότε θα εστιάσετε κυρίως στη χρήση Ethernet shield (αντίστοιχο με ένα Arduino σε λειτουργία Ethernet), WiFi shield και ένα τηλέφωνο Android ως ενδιάμεσο κόμβο που συλλέγει δεδομένα από το Arduino και τα προωθεί στο internet.

Τι θα χρειαστείτε για το βασικό κύκλωμα:

1. Μια πλακέτα Arduino ή κάποιον κλώνο (η πλακέτα μπορεί να είναι η Uno ή Mega ή κάποια συμβατή).
2. Ένας αισθητήρας θερμοκρασίας και υγρασίας DHT22
3. Μια φωτοαντίσταση LDR
4. Μια αντίσταση 10K
5. Μια αντίσταση 47K
6. Μια πηγή τροφοδοσίας της πλακέτας Arduino (μπορεί να είναι μια θύρα USB ή μια μπαταρία 9V)

Για τη σύνδεση στο internet : μπορείτε να χρησιμοποιήσετε είτε μια μονάδα WiFi (ή ένα WiFi Arduino), μια μονάδα Ethernet, ή μια πλακέτα ADK και ένα τηλέφωνο Android.

Για τη εύκολη συναρμολόγηση και για να αποφύγετε τις συγκολλήσεις μπορείτε να χρησιμοποιήσετε :

Ένα breadboard

Ή jumper wires

Τα παραπάνω κομμάτια είναι αρκετά ώστε να κατασκευάσετε ένα κύκλωμα που θα μετρά θερμοκρασία, υγρασία και φως χρησιμοποιώντας τους αισθητήρες.

Διάταξη κυκλώματος

Ο αισθητήρας DHT22 είναι ένας ψηφιακός αισθητήρας, ενώ η φωτοαντίσταση LDR αναλογικός, γι' αυτό θα χρειαστείτε μία αναλογική και μία ψηφιακή θύρα στη πλακέτα του Arduino. Αντί να συνδέσετε τις εξόδους των αισθητήρων απευθείας στις θύρες θα πρέπει πρώτα να φτιάξετε ένα μικρό κύκλωμα.

Ο αισθητήρας LDR λειτουργεί όπως μια απλή αντίσταση μια δύο ακροδέκτες (χωρίς θέματα πολικότητας). Έτσι, συνδέετε τον έναν ακροδέκτη απευθείας στα 5V (ας τον ορίσουμε ως ακροδέκτη Α) της πλακέτας Arduino και τον άλλον ακροδέκτη (τον Β) μέσω της αντίστασης 10K στη γείωση της πλακέτας. Στη συνέχεια με ένα καλώδιο συνδέουμε το σημείο ανάμεσα στο Β και την αντίσταση με την αναλογική είσοδο) του Arduino. Με αυτόν τον τρόπο μπορούμε να διαβάσουμε από την αναλογική θύρα κάθε πτώση τάσης που προκαλείται από τη μεταβολή του φωτός.

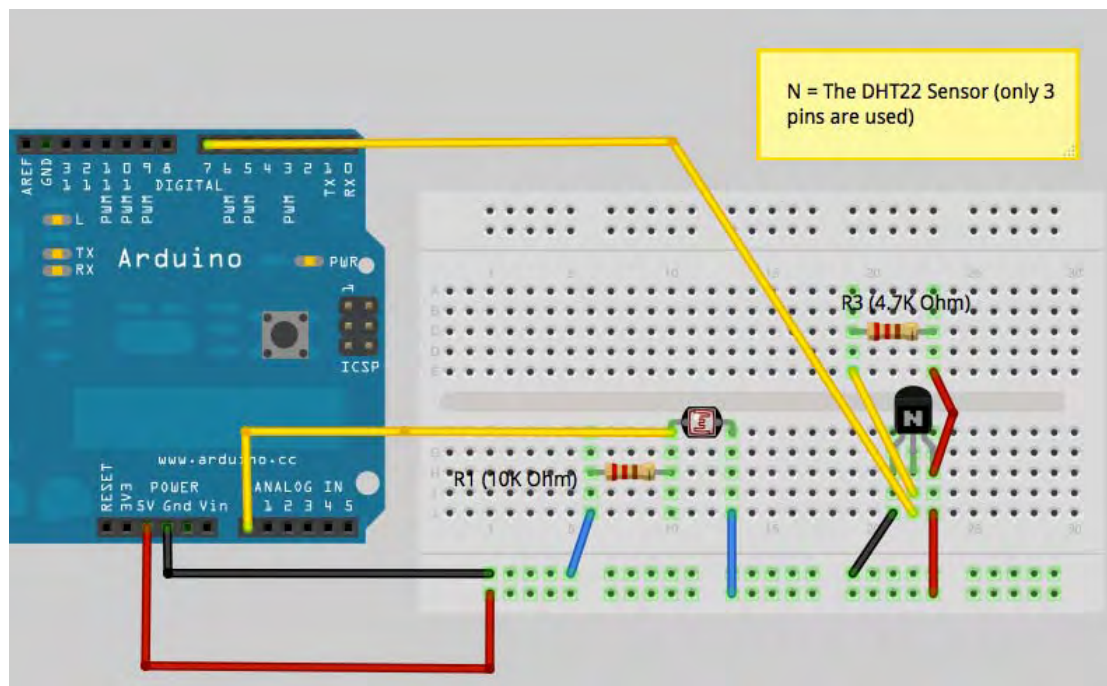
Ο λόγος που τοποθετείται η αντίσταση ανάμεσα είναι ώστε να είστε σίγουροι ότι παίρνετε έγκυρες μετρήσεις, ακόμα και όταν η LDR απομακρυνθεί ή δείχνει χαμηλή αντίσταση.

Ο αισθητήρας DHT22 έχει 4 ακροδέκτες, αλλά μόνον οι τρεις θα χρησιμοποιηθούν στη προκειμένη περίπτωση. Οι δύο από αυτούς είναι για την τροφοδοσία, οπότε και θα συνδεθούν στα 5V και στο GND του Arduino. Ο τρίτος ακροδέκτης είναι η έξοδος του αισθητήρα και εκεί θα πάρουμε τις τιμές της θερμοκρασίας και της υγρασίας. Το σχήμα 8-1 δίνει περισσότερες πληροφορίες σχετικά με τους αισθητήρες και τη χρήση τους. Τοποθετείτε μια αντίσταση 4.7K ανάμεσα στα 5V και την έξοδο για τον ίδιο λόγο που το κάνατε και για τον LDR. Στη συνέχεια συνδέετε τον ακροδέκτη εξόδου απευθείας στην ψηφιακή είσοδο του Arduino. Μπορείτε να ελέγξετε το σχήμα 8-2 για το πλήρες κύκλωμα και το σχήμα 8-3 για το αναλυτικό σχηματικό του κυκλώματος που δείχνει τις συνδέσεις πιο καθαρά.

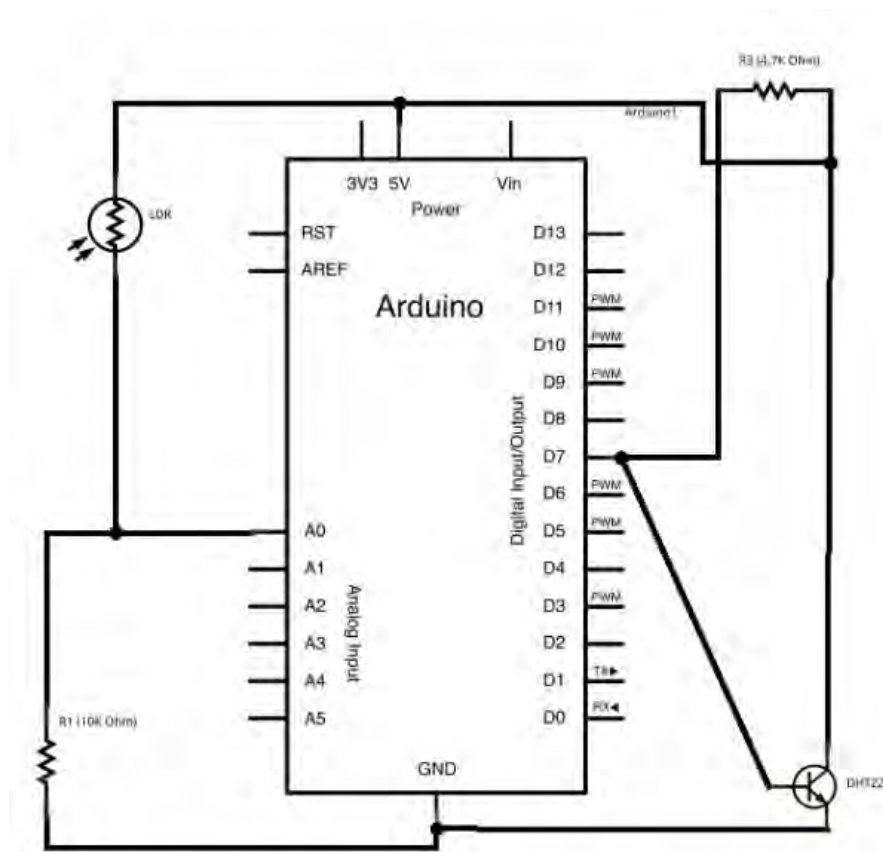


Σχήμα 8-1: Ο αισθητήρας DHT22 θερμοκρασίας και υγρασίας. Pin 1: Τροφοδοσία 5V. Pin2: Έξοδος δεδομένων. Pin3: Δεν χρησιμοποιείται. Pin 4: GND

Είστε έτοιμοι! Τροφοδοτώντας το Arduino σας θα τροφοδοτήσει ταυτόχρονα και το κύκλωμα των αισθητήρων, ξεκινώντας την αποστολή δεδομένων στην πλακέτα.



Σχήμα 8-2 : Κύκλωμα σε breadboard για το project 1.



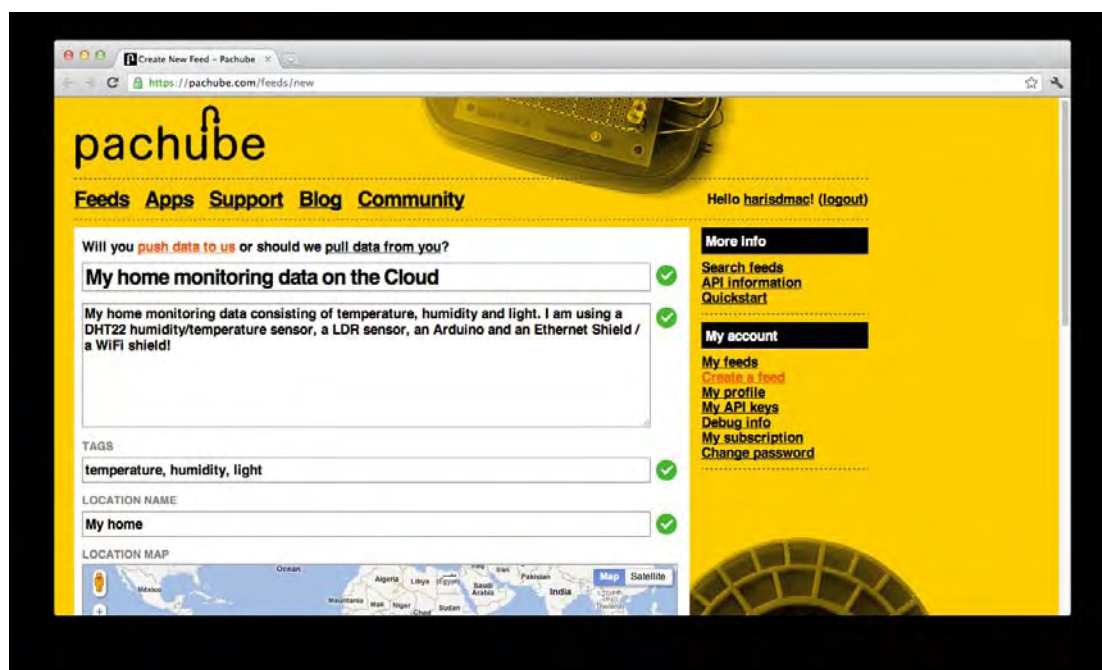
Σχήμα 8-3 : Το σχηματικό του κυκλώματος. Μπορείτε να δείτε τις συνδέσεις ανάμεσα στις αντιστάσεις και τους αισθητήρες LDR και DHT22, όπως επίσης και τις εισόδους του Arduino.

Ετοιμάζοντας το web service για να δεχτεί δεδομένα

Το επόμενο βήμα είναι να οριστεί η διαδικτυακή εφαρμογή κατάλληλα ώστε να μπορέσει να συλλέξει τα δεδομένα σας. Για το συγκεκριμένο πείραμα χρησιμοποιήθηκε η πλατφόρμα Pachube.

Σημείωση : Η πλατφόρμα Pachube εξαγοράστηκε από την εταιρία LogMeIn. Έτσι, το περιβάλλον της εφαρμογής έχει τροποποιηθεί και πλέον ονομάζεται Xively (xively.com). Επειδή όμως η χρήση οποιασδήποτε πλατφόρμας άπτεται στις προτιμήσεις του χρήστη, επιλέξαμε να διατηρήσουμε στην περιγραφή της παρουσίας την παλιά (πλέον) πλατφόρμα, ως έναν ενδεικτικό οδηγό υλοποίησης.

Η χρήση της πλατφόρμας προϋποθέτει τη δημιουργία λογαριασμού Pachube. Μετά την επιτυχή εγγραφή μπορείτε να επισκεφτείτε τη σελίδα 'My API Keys' ώστε να δείτε το Master API Key που θα χρησιμοποιήσετε ώστε να ανεβάσετε στην εφαρμογή τα δεδομένα σας. Στη συνέχεια θα πρέπει να ορίσετε ένα Pachube feed για του αισθητήρες περιβάλλοντος. Το feed αυτό θα περιέχει πληροφορίες για τις τρεις διαφορετικές ροές δεδομένων που παράγονται και κάποιες πρόσθετες πληροφορίες σχετικά με το project σας. Δημιουργώντας το feed, το Pachube σας παρέχει κάποιες πολύ σημαντικές πληροφορίες γι' αυτό, το Fee did. Χρησιμοποιώντας το Fee did μπορείτε ανά πάσα στιγμή να ανατρέχετε εκεί και να αναζητάτε πληροφορίες σχετικά με τους αισθητήρες σας, άλλα πιο σημαντικό, το χρειάζεστε ώστε να αποθηκεύετε τις πληροφορίες των αισθητήρων ώστε να είναι εύκολα προσβάσιμες. Μέσω του URL: <https://pachube.com/feeds/new>, μπορείτε να δημιουργήσετε ένα νέο feed.



Σχήμα 8-4: Ορίζοντας ένα Pachube feed. Εισάγουμε πληροφορίες όπως τίτλος, περιγραφή, τοποθεσία, κ.ά.

Θα ερωτηθείτε εάν θέλετε να στείλετε τα δεδομένα στην υπηρεσία ή εάν το Pachube θα τα αντλεί από το σύστημά σας κατά περιόδους. Κρατήστε την προεπιλογή (push data to us), καθώς γενικά είναι προτιμότερο να στέλνεται τα δεδομένα όποτε εσείς επιλέγετε.

Στη συνέχεια θα πρέπει να ορίσετε κάποιες γενικές πληροφορίες σχετικά με τα feed, όπως τίτλος, περιγραφή, την τοποθεσία του αισθητήρα κ.ά. Οι επιλογές αυτές είναι προαιρετικές.

Ακολουθώντας, φτάνετε στο πιο σημαντικό τμήμα, όπου θα ορίσετε τα datastreams. Απλά πατήστε **+ Add a datastream**

Όπως διευκρινίστηκε και προηγουμένως τα datastreams απεικονίζουν την ακολουθία δεδομένων από τους αισθητήρες. Επομένως κάθε datastream αντιστοιχεί σε έναν συγκεκριμένο αισθητήρα. Έτσι θα πρέπει να φτιάξετε τρία διαφορετικά datastreams, όπως απεικονίζεται στο σχήμα 8-5. Δεν είναι απαραίτητο να συμπληρώσετε κάθε πληροφορία που εμφανίζεται, απλά θα πρέπει να οριστεί τουλάχιστον ένα ID για κάθε datastream.

Όταν ολοκληρώσετε τη διαδικασία απλά πατήστε το κουμπί 'Save'. Αυτό ήταν! Μόλις δημιουργήσατε το πρώτο feed στο Pachube. Η επόμενη σελίδα περιέχει όλες τις σημαντικές πληροφορίες του FeedID, το οποίο μπορείτε να το βρείτε στην URL της σελίδας (θα είναι κάτι αντίστοιχο με το <https://pachube.com/feeds/28602>) κάτω από το tag 'WEBSITE'. Το feed είναι ενεργοποιημένο και μπορεί να λάβει και να αποθηκεύσει τα δεδομένα από τους αισθητήρες σας! Στη συνέχεια θα δούμε τα βασικά στοιχεία του κώδικα για να γίνει αυτό εφικτό.

The screenshot shows the 'Create New Feed' form on the Pachube website. The form is divided into several sections: Location (Latitude, Longitude, Elevation), Exposure (Indoor/Outdoor), Disposition (Fixed/Mobile), Domain (Physical/Virtual), Website (publicly visible), Contact Email (publicly visible), Feed Status (Public/Private), and Datastreams. The Datastreams section contains a table with three rows, each representing a datastream. Each row has columns for ID, Tags, Units, and Symbol, and a 'Remove' button. The first row is for temperature (temp, C, temp), the second for humidity (humidity, %, hum), and the third for light (light, L).

ID	TAGS	UNITS	SYMBOL	
1	temp	C	temp	Remove
2	humidity	%	hum	Remove
3	light	L	L	Remove

Σχήμα 8-5 : Καθορίζοντας τα δεδομένα θερμοκρασίας, υγρασίας και φωτός για το feed.

Σύνδεση με το Web Service και αποστολή δεδομένων

Έχετε έως τώρα φτιάξει το κύκλωμά σας, έχετε δημιουργήσει ένα feed και τα απαραίτητα datastreams για το Pachube. Το βήμα που απομένει είναι να προγραμματίσετε το Arduino ώστε να στέλνει δεδομένα στο Pachube ώστε να μπορέσετε να τα δείτε σε γραφικό περιβάλλον.

Όπως έχει ήδη αναλυθεί, θα καλύψουμε τρεις τρόπους επικοινωνίας: α) χρησιμοποιώντας Ethernet, β) με WiFi και γ) χρησιμοποιώντας τηλέφωνο Android. Για την τελευταία περίπτωση, θα απεικονίσете επίσης δύο διαφορετικούς τρόπους αποστολής δεδομένων στο κινητό: 1) με απευθείας USB σύνδεση χρησιμοποιώντας το Arduino Debug Bridge (ADB) σαν εναλλακτικό του Google ADK και μια ενεργοποιημένη ως ADK πλακέτα και 2) χρησιμοποιώντας σύνδεση Bluetooth.

Χρησιμοποιώντας το Arduino Ethernet ή το Ethernet Shield.

Για να χρησιμοποιήσετε την επικοινωνία Ethernet μέσω καλωδίου, μπορείτε να επιλέξετε μια πλακέτα Arduino Ethernet (<http://arduino.cc/en/Main/ArduinoBoardEthernet>) ή μια μονάδα όπως στην εικόνα 8-6. Το υπόλοιπο κύκλωμα παραμένει όπως περιγράφηκε προηγουμένως.

****Όταν χρησιμοποιείτε μια τρίτη μονάδα Ethernet, βεβαιωθείτε ότι κάνει χρήση του chip 'Wiznet Ethernet', που υποστηρίζεται από την επίσημη βιβλιοθήκη του Arduino.**



Εικόνα 8-6 : Μια πλακέτα Ethernet της SeedStudio βασισμένη στον επεξεργαστή Wiznet W5100 (Η εικόνα είναι ιδιοκτησία της SeedStudio).

Βεβαιωθείτε ότι έχετε συνδέσει την θύρα Ethernet με κατάλληλο καλώδιο με ένα LAN δίκτυο χρησιμοποιώντας ένα switch ή έναν router. Στη συνέχεια ανεβάστε τον

ακόλουθο κώδικα στην πλακέτα Arduino. Μπορείτε να βρείτε τον κώδικα γι' αυτό το project στη σελίδα: <http://www.buildinginternetofthings.com>.

Κείμενο : *Ο κώδικας του Project 1 χρησιμοποιώντας ένα Arduino Ethernet ή ένα ethernet module*

```
#include <DHT22.h>

#include <SPI.h>

#include <Ethernet.h>

// Data wire is plugged into port 7 on the Arduino

#define DHT22_PIN 7

// Setup a DHT22 instance

DHT22 myDHT22(DHT22_PIN);

DHT22_ERROR_t errorCode;

//Useful char buffers for creating the CSV datastream content

char buf1[16];

char buf2[16];

char DataBuff[16];

//Light sensor is connected to analog port 0

#define LIGHT_PIN 0

// Time variable for custom timer

long lastTime;

// sensor variables

float temp;

float humid;

int light;

//Set up ethernet and network related data

byte mac[] = { 0xCC, 0xAC, 0xBE, 0xEF, 0xFE, 0x91 };

byte ip[] = { 192, 168, 1, 124 }; // no DHCP so we set our own IP address

byte PachubeServer[] = { 173, 203, 98, 29 };

//create a Client object for handling connection
```

```

Client localClient(PachubeServer, 80);

//feed data function, provides data for the POST command in comma separated
values (CSV)
void feedData()
{
    ftoa(buf1, temp, 2); //we need to convert float temp to
    //char
    ftoa(buf2, humid, 2); //we need to convert float humid
    //to char
    sprintf(DataBuff,"%s,%s,%d", buf1, buf2,light); //we save
    //all variables including light into one char variable
    //DataBuff
}

//Define the sendData() function that handles the communication with Pachube
implementing the protocol for sending feeds based on Pachube API v2.
void sendData(){
    if (localClient.connect()) {
        feedData();
        int content_length = strlen(DataBuff);
        localClient.print("PUT /v2/feeds/");
        localClient.print("28602");
        localClient.print(".csv HTTP/1.1\nHost:
        api.pachube.com\nX-PachubeApiKey: ");
        localClient.print("NkyX---90s"); //Replace with your
        Pachube API Key
        localClient.print("\nUser-Agent: ");
        localClient.print("\nContent-Type: text/csv\nContent-Length: ");
        localClient.print(content_length);
        localClient.print("\nConnection: close\n\n");
    }
}

```

```

        localClient.print(DataBuff);

        localClient.print("\n");
    }
}

//Arduino setup function
void setup(void)
{
    //initialize sensor variables

    temp = 0.0f;

    humid = 0.0f;

    light = 0;

    pinMode(LIGHT_PIN, INPUT);

    //initiate Ethernet module
    Ethernet.begin(mac, ip);

    delay(500);

    //Note start time
    lastTime = millis();
}

//Arduino loop function
void loop(void)
{
    if ((millis() - lastTime) > 20000){

        //timer has expired

        lastTime = millis();

        //Read DHT22 Data here:

        errorCode = myDHT22.readData();

        switch(errorCode){

            case DHT_ERROR_NONE:

                temp = myDHT22.getTemperatureC();

```

```

        humid = myDHT22.getHumidity();

        break;
    }

    light = analogRead(LIGHT_PIN);

    //send the data over Pachube

    sendData();

}

//small delay for the main loop

delay(100);

}

//Convert double to char (due to currently sprintf in Arduino fails to do
so)

char *ftoa(char *a, double f, int precision)
{
    long p[] = {0,10,100,1000,10000,100000,1000000,10000000,100000000};

    char *ret = a;

    long heital = (long)f;

    itoa(heital, a, 10);

    while (*a != '\0') a++;

    *a++ = '.';

    long desimal = abs((long)((f - heital) * p[precision]));

    itoa(desimal, a, 10);

    return ret;
}

```

Ανάλυση κώδικα

#include <DHT22.h>

Ο αισθητήρας DHT22έχει ψηφιακή έξοδο. Αυτό σημαίνει ότι θα πρέπει να τροποποιήσετε τα εισερχόμενα bits σε bytes και στη συνέχεια σε πραγματικές τιμές θερμοκρασίας και υγρασίας. Για να ελαχιστοποιήσετε τον κόπο σας, μπορείτε να

χρησιμοποιήσετε την έτοιμη βιβλιοθήκη που είναι διαθέσιμη για Arduino από τον Ben Adams (<https://github.com/nethoncho/Arduino-DHT22>). Για να εγκαταστήσετε την βιβλιοθήκη πρώτα την κατεβάζετε και στη συνέχεια την τοποθετείτε στη θέση : ../Arduino installation folder/Resources/Java/Libraries in a folder named DHT22.

Υπάρχει επίσης και μια άλλη βιβλιοθήκη για τους αισθητήρες της σειράς DHTxx με παρόμοια εγκατάσταση και τρόπο χρήσης , στη διεύθυνση : <http://arduino.cc/playground/Main/DHTLib>

```
#include <SPI.h>
```

```
#include <Ethernet.h>
```

Η ομάδα του Arduino μας παρέχει έτοιμη τη βιβλιοθήκη Ethernet (<http://arduino.cc/en/Reference/Ethernet>), ώστε να απλοποιήσει την επικοινωνία με το internet (π.χ. σύνδεση με ports, πρωτόκολλα επικοινωνίας, κ.ά.) με απλές εντολές. Η χρήση της βιβλιοθήκης Ethernet απαιτεί και τη χρήση της βιβλιοθήκης SPI.

Στη συνέχεια καθορίζετε διάφορες μεταβλητές που σχετίζονται με τον DHT22, μια μεταβλητή χρόνου που χρησιμοποιείται για έναν ειδικό timer και μεταβλητές των αισθητήρων σχετικά με τη θερμοκρασία, την υγρασία και το φως.

```
// Data wire is plugged into port 7 on the Arduino
```

```
#define DHT22_PIN 7
```

```
// Setup a DHT22 instance
```

```
DHT22 myDHT22(DHT22_PIN);
```

```
DHT22_ERROR_t errorCode;
```

```
//Useful char buffers for creating the CSV datastream content
```

```
char buf1[16];
```

```
char buf2[16];
```

```
char DataBuff[16];
```

```
//Light sensor is connected to analog port 0
```

```
#define LIGHT_PIN 0
```

```
// Time variable for custom timer
```

```
long lastTime;
```

```
// sensor variables
```

```
float temp;
```

```
float humid;
```

```
int light;
```

Ακολουθώς καθορίζετε τις μεταβλητές που έχουν σχέση με την επικοινωνία με το Pachube:

```
//Set up ethernet and network related data
```

```
byte mac[] = { 0xCC, 0xAC, 0xBE, 0xEF, 0xFE, 0x91 };
```

```
byte ip[] = { 192, 168, 1, 124 }; // We need to set your own IP address,
```

```
make sure this address can be connected to your local gateway/router
```

```
byte PachubeServer[] = { 173, 203, 98, 29 }; //the IP address of Pachube
```

```
service
```

Θα παρατηρήσετε την τοπική IP την οποία χρησιμοποιεί το Ethernet module για να συνδεθεί με το internet μέσω του τοπικού gateway/router. Βεβαιωθείτε ότι την έχετε τροποποιήσει κατάλληλα. Παρατηρήσατε επίσης ότι δεν χρησιμοποιούνται URL όπως π.χ. www.pachube.com, αλλά IP διευθύνσεις αντί αυτού. Αυτό συμβαίνει επειδή το module και η βιβλιοθήκη δεν υποστηρίζει DNS ακόμη (μετατροπή του URL σε IP).

Ο ακόλουθος κώδικας δημιουργεί Client objects (όπως client socket) που θα συνδεθεί στο Pachube μέσω της παρεχόμενης IP και της πόρτας 80.

```
//create a Client object for handling connection
```

```
Client localClient(PachubeServer, 80);
```

Έχετε ολοκληρώσει την τοποθέτηση των μεταβλητών έως εδώ, οπότε τώρα είναι η σειρά να καθοριστούν οι απαιτούμενες λειτουργίες που θα χρησιμοποιηθούν. Θα πρέπει να ξεκινήσετε με τη εντολή `feedData()` που δημιουργεί ένα CSV μήνυμα που περιέχει τις τιμές των αισθητήρων και τις αποθηκεύει στη μεταβλητή `DataBuff`.

```
//feed data function, provides data for the POST command in comma separated
```

```
values (CSV)
```

```
void feedData()
```

```
{
```

```
ftoa(buf1, temp, 2);
```

```
ftoa(buf2, humid, 2);
```

```
sprintf(DataBuff,"%s,%s,%d", buf1, buf2,light);
```

```
}
```

Συνεχίζετε με την εντολή `SendData()` που περιέχει τις επικοινωνίες που θα πρέπει να γίνουν με το Pachube.

//Define the sendData() function that handles the communication with Pachube

implementing the protocol for sending feeds based on Pachube API v2.

```
void sendData(){  
    if (localClient.connect()) {  
        feedData();  
        int content_length = strlen(DataBuff);  
        localClient.print("PUT /v2/feeds/");  
        localClient.print("28602");  
        localClient.print(".csv HTTP/1.1\nHost:  
api.pachube.com\nX-PachubeApiKey: ");  
        localClient.print("NkyX---90s"); //Replace with your  
        Pachube API Key  
        localClient.print("\nUser-Agent: ");  
        localClient.print("\nContent-Type: text/csv\nContent-  
Length: ");  
        localClient.print(content_length);  
        localClient.print("\nConnection: close\n\n");  
        localClient.print(DataBuff);  
        localClient.print("\n");  
    }  
}
```

Εάν έχετε μια επιτυχή σύνδεση (localClient.connect()), συνεχίζετε με την εντολή feedData όπως αναφέρθηκε πιο πάνω. Στη συνέχεια λέτε στο Pachube server με τον οποίο μόλις έχετε συνδεθεί ότι θέλετε να στείλετε τα δεδομένα (POST)στη διεύθυνση /v2/feeds/28602.csv, η οποία είναι προφανώς το δικό σας FeedID. Με την τοποθέτηση του .csv, ορίζετε ότι θέλετε να χρησιμοποιήσετε το CSV format για την αποστολή δεδομένων. Ακολούθως προσθέτετε την επικεφαλίδα για τα δεδομένα και περιλαμβάνετε και το κλειδί Pachube API στον κώδικα.

Επίσης καθορίζετε τον τύπο των δεδομένων που θα έχουν τα προς αποστολή δεδομένα (text/csv), το μήκος των δεδομένων που στέλνετε (που υπάρχει στη μεταβλητή DataBuff) και τέλος κάποιες πρόσθετες πληροφορίες σχετικά με τη σύνδεση http.

Αυτά είναι όλα όσα χρειάζεστε για να επικοινωνήσετε με το Pachube!

Οι ακόλουθες γραμμές περιλαμβάνουν το βασικό setup του Arduino και τις εντολές loop.

//Arduino setup function

void setup(void)

{

//initialize sensor variables

temp = 0.0f;

humid = 0.0f;

light = 0;

pinMode(LIGHT_PIN, INPUT);

//initiate Ethernet module

Ethernet.begin(mac, ip);

delay(500);

//Note start time

lastTime = millis();

}

Στο εσωτερικό της εντολής setup που ορίζονται οι μεταβλητές του αισθητήρα με null τιμές, ορίζετε τη αναλογική θύρα 0 (LIGHT_PIN) στο INPUT, ώστε το Arduino να καταλάβει ότι περιμένετε αναλογική είσοδο από εκεί.

Επίσης, καθορίζετε το Ethernet module ορίζοντάς του τις διευθύνσεις mac και IP που έχετε επιλέξει, δίνοντας του λίγο χρόνο ώστε να βεβαιωθείτε ότι πρόλαβε να κάνει initialize προτού προχωρήσετε παρακάτω.

Τέλος, ελέγχετε τον χρόνο έναρξης του προγράμματός σας ώστε στη συνέχεια να ελέγχετε στο εσωτερικό της εντολής loop() εάν ο επιθυμητός χρόνος έχει φθάσει για να στείλετε τα δεδομένα στο Pachube (ένας απλός timer).

//Arduino loop function

void loop(void)

{

//checks if 20 seconds have elapsed

if ((millis() - lastTime) > 20000) {

// timer has expired

```

        lastTime = millis();

        //Read DHT22 Data here:

        errorCode = myDHT22.readData();

        switch(errorCode){

        case DHT_ERROR_NONE:

            temp = myDHT22.getTemperatureC();

            humid = myDHT22.getHumidity();

                break;

        }

        light = analogRead(LIGHT_PIN);

        //send the data over Pachube

        sendData();

    }

    //small delay for the main loop

    delay(100);

}

```

Η εντολή loop() ξεκινά ελέγχοντας εάν ο χρόνος έχει ολοκληρωθεί (καθορισμένος στα 20000 msec), ώστε να μπορέσετε να τραβήξετε και να ανεβάσετε τα δεδομένα του αισθητήρα. Εάν έχει ολοκληρωθεί, ορίζετε στον timer την παρούσα χρονική στιγμή για την επόμενη χρήση και προχωράτε στην ανάγνωση των δεδομένων του αισθητήρα DHT22 μέσω της βιβλιοθήκης που χρησιμοποιείτε. Επίσης διαβάσετε τα δεδομένα του αισθητήρα φωτός μέσω της αναλογικής εισόδου (LIGHT_PIN).

Μόλις ολοκληρωθούν, όλα τα δεδομένα σώζονται στις κατάλληλες μεταβλητές (temp, humid και light) και προχωράτε ενεργοποιώντας την εντολή sendData για να προωθηθούν στο Pachube. Έξω από το if statement του ελέγχου του timer, προσθέτετε μια μικρή καθυστέρηση 100 msec στο κύριο loop του Arduino.

Το πρώτο μέρος της ασφαλείας Arduino περιλαμβάνει την κατάλληλη πιστοποίηση του Arduino σας, ώστε να μπορείτε να είστε σίγουροι ότι η εφαρμογή σας λαμβάνει δεδομένα μόνο από την πλακέτα σας. Η ιδέα έχει εγκριθεί από τις κοινές web-based πιστοποίησης συστημάτων που απαιτούν από τον χρήστη όνομα χρήστη και κωδικό πρόσβασης. Για τέτοια συστήματα, τα τελευταία διαπιστευτήρια αποθηκεύονται συνήθως σε βάσεις δεδομένων εφαρμογής έτσι ώστε να μπορούν να συγκριθούν με το τι κάνουν οι χρήστες αφού γίνει πιστοποίηση. Ωστόσο, οι πληροφορίες των χρηστών δεν αποθηκεύεται ως απλό κείμενο, δεδομένου ότι μπορεί να προκαλέσει σοβαρούς κινδύνους ασφαλείας.

Οι Hash συναρτήσεις είναι μέθοδοι που δέχονται σαν είσοδο ένα string (ή ένα αντικείμενο) και επιστρέφουν ένα string σταθερού μεγέθους που είναι μοναδικό και συνδέεται στενά με την είσοδο. Αυτό σημαίνει δύο πράγματα:

α) δεν υπάρχει κανένας τρόπος (πιο σωστά, είναι εξαιρετικά απίθανο) να πάρει το ίδιο αποτέλεσμα για δύο διαφορετικές εισόδους,

β) ακόμη και η παραμικρή αλλαγή στα δεδομένα εισόδου παράγει μια εντελώς διαφορετική έξοδο. Επιπλέον, δεν υπάρχει τρόπος από το κωδικοποιημένο μήνυμα να γίνει ανάκτηση του αρχικού μηνύματος.

Αυτό καθιστά τις hash συναρτήσεις κατάλληλες για χρήση σε έλεγχο ταυτότητας, δεδομένου ότι μπορούν να εγγυηθούν ότι ακόμα και αν κάποιος πάρει τις κωδικοποιημένες πληροφορίες, δεν υπάρχει τρόπος για να αποκαλύψει τα πραγματικά δεδομένα. Οι Hash συναρτήσεις λειτουργούν παρόμοια με τις checksum που δημιουργούν ένα κώδικα σταθερού μεγέθους (checksum) για μια συγκεκριμένη είσοδο και χρησιμοποιούνται κυρίως για την επαλήθευση της ακεραιότητας του αρχείου δεδομένων κατά τη διάρκεια εργασιών, όπως μετάδοση του δικτύου, μέσα αποθήκευσης, κλπ.

Αρκετές Hash συναρτήσεις υπάρχουν και χρησιμοποιούνται για τέτοιους σκοπούς. Οι πιο δημοφιλείς είναι οι συναρτήσεις MD5 και SHA1/SHA256. Για τα project σας που βασίζονται σε εφαρμογές Cloud σας θα μπορούσε να είναι χρήσιμο να χρησιμοποιηθούν τέτοιες Hash συναρτήσεις για την πιστοποίηση της επικοινωνίας μεταξύ του Arduino σας και μιας web-based εφαρμογής. Σήμερα, οι

περισσότεροι αισθητήρες που σχετίζονται με εφαρμογές Cloud όπως θα δείτε στα Κεφάλαια 8 και 9) χρησιμοποιούνε μόνο κλειδιά πιστοποίησης μεταδίδοντας τα ως απλό κείμενο, αλλά μπορείτε να δημιουργήσετε τη δική σας εφαρμογή Cloud που θα υποστηρίζει hash πιστοποιήσεις με το Arduino σας.

Βασική Κωδικοποίηση / χρήστη με SHA1 hash συνάρτηση

Για το επόμενο Arduino Sketch χρησιμοποιείτε τη βιβλιοθήκη 'Cryptosuite' για το Arduino. Αυτή η βιβλιοθήκη παρέχει εφαρμογές συμβατές με Arduino και τις κατάλληλες συναρτήσεις για τις SHA1 και SHA256 hash συναρτήσεις. Θα χρειαστεί να κατεβάσετε τα αρχεία της βιβλιοθήκης από εδώ: <http://code.google.com/p/cryptosuite> και να εγκατασταθούν στο περιβάλλον Arduino σας όπως είπαμε προηγουμένως. Εκτελέστε το ακόλουθο sketch στο Arduino σας και ανοίξτε το Serial Monitor. Ελέγξτε την έξοδο και θα δείτε ότι η hash-κωδικοποιημένη έξοδος του αρχικού μηνύματος θα είναι "7c02a8797b4a101e242c095a32faad0d84d82ff4". Μπορείτε να δοκιμάσετε με διαφορετικά string ως είσοδο και στη συνέχεια, επιβεβαιώστε την έξοδο με μία web-based SHA1 γεννήτρια, όπως εδώ: <http://www.webtoolkit.info/demo/javascript-sha-1/>.

Υπόδειγμα 4-1. Δείγμα κώδικα Arduino χρησιμοποιώντας SHA1 hash συνάρτηση για να κωδικοποιήσετε ένα κλειδί ή ένα όνομα χρήστη για την πιστοποίηση του Arduino σας.

```
#include "sha1.h"
//The secret key (username) to store the encoded output by the hash function
uint8_t* secretKey;
//Help function for printing the output of the hash function
void printHash(uint8_t* hash) {
  int i;
  for (i=0; i<20; i++) {
    Serial.print("0123456789abcdef"[hash[i]>>4]);
    Serial.print("0123456789abcdef"[hash[i]&0xf]);
  }
  Serial.println();
}

void setup() {
  uint8_t* hash;
  Serial.begin(9600);
  //Initialize the SHA1 function
  Sha1.init();
  //Pass the username or sensor key to function
  Sha1.print("MyArduinoSecretName");
  //Get the hash of the username or sensor key
```

```

        sectretKey = Sha1.result();
        //print the key in the Serial monitor
        printHash(sectretKey);
    }

    void loop()
    {
        //here you can read some sensor data
        //...
        //before sending data to the Cloud
        //you can use the sectretKey to authenticate
    }
}

```

Βασική Κωδικοποίηση με τη χρήση της συνάρτησης SHA1 hash

Μπορείτε να ξεκινήσετε sketch σας, συμπεριλαμβάνοντας τα βασικά από την «Cryptosuite» βιβλιοθήκη. Σε αυτό το παράδειγμα που χρησιμοποιείτε η SHA1 Hash συνάρτηση, έτσι θα πρέπει να συμπεριλάβετε μόνο το αντίστοιχο αρχείο κεφαλίδας.

```
#include "sha1.h"
```

Εδώ θα χρησιμοποιήσετε μία συνάρτηση βοήθειας για την εκτύπωση της εξόδου της συνάρτησης hash. Ίδια με την είσοδο, η έξοδος είναι σε ανυπόγραφη (unsigned) ακέραια μορφή. Η συνάρτηση θα ελέγχει για ακέραιους αριθμούς στη μνήμη της μεταβλητής εισόδου και τη μετατροπή τους σε αριθμούς και χαρακτήρες και τα εκτυπώνει στο Serial Monitor.

```

void printHash(uint8_t* hash) {
    int i;
    for (i=0; i<20; i++) {
        Serial.print("0123456789abcdef"[hash[i]>>4]);
        Serial.print("0123456789abcdef"[hash[i]&0xf]);
    }
    Serial.println();
}

```

Μπορείτε να δηλώσετε την συνάρτηση setup () που αρχικοποιεί το sketch σας. Μπορείτε ρυθμίσετε το Serial στον κατάλληλο ρυθμό, ώστε να μπορείτε να παρακολουθήσετε την έξοδο στο Serial Monitor σας.

```

void setup() {
    Serial.begin(9600);
}

```

Μπορείτε να αρχικοποιήσετε την συνάρτηση SHA1

```
Sha1.init();
```

Στη συνέχεια δηλώνεται το όνομα χρήστη ή το κλειδί αισθητήρα για να κωδικοποιηθεί ως παράμετρος εισόδου για τη συνάρτηση:

```
Sha1.print("MyArduinoSectretName");
```

Αποθηκεύετε την κωδικοποιημένη έξοδο (γνωστή ως hash από το όνομα της μεθόδου) στη μεταβλητή uint8_t που έχετε ήδη ορίσει:

```
sectretKey = Sha1.result();
```

Τέλος, μπορείτε να εκτυπώσετε την έξοδο στο Serial Monitor και να δείτε πώς μοιάζει. Μπορείτε επίσης να προσπαθήσετε να τροποποιήσετε (ακόμη και να αλλάξετε ένα γράμμα από την κεφαλαίο σε μικρό) και να δείτε πόσο διαφορετική είναι η hash έξοδος.

```
printHash(sectretKey);
```

Δεν έχετε εφαρμόσει κάτι συγκεκριμένο στο εσωτερικό της συνάρτησης loop(). Συνήθως θα χρειάζεται να πιστοποιείτε τον Arduino σας μία φορά, είτε κατά την έναρξη της εκτέλεσης του sketch (δηλαδή στη συνάρτηση setup ()) ή μέσω χρησιμοποιώντας ένα flag στο εσωτερικό της συνάρτησης loop().

Για παράδειγμα, μπορείτε να ορίσετε μια μεταβλητή Boolean flag κάπου στις αρχές του sketch σας:

```
boolean once = true;
...
void loop() {
  if(once)
  {
    //authenticate
    once = false;
  }
}
```

Στη συνέχεια, μπορείτε να διαβάσετε κάποια δεδομένα από αισθητήρα και να τα προωθήσετε στο Internet. Περισσότερες πληροφορίες σχετικά με το πώς να το κάνετε αυτό σε επόμενα κεφάλαια!

Κρυπτογράφηση των δεδομένων

Για την κρυπτογράφηση των δεδομένων που θα χρησιμοποιήσετε τη βιβλιοθήκη `ArduinoAES256`. Αυτή η βιβλιοθήκη περιέχει μια υλοποίηση του συμμετρικού αλγορίθμου κρυπτογράφησης `AES`, ειδικά για την πλατφόρμα `Arduino`. Όπως υποδηλώνει το όνομα, θα χρειαστείτε ένα συμμετρικό κλειδί για την κρυπτογράφηση των δεδομένων σας. Το ίδιο κλειδί θα χρησιμοποιηθεί επίσης και στην εφαρμογή αποκρυπτογράφησης έτσι ώστε το αρχικό μήνυμα μπορεί να αποκαλυφθεί. Γι'αυτό το project θα χρησιμοποιήσετε την παρακάτω ακολουθία αλφαριθμητικών χαρακτήρων ως κλειδί:

000102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f

Θα δείτε επίσης την επίδραση της κρυπτογράφησης σε ένα μήνυμα και πώς μπορεί να αποκρυπτογραφηθεί πάλι χρησιμοποιώντας το ίδιο κλειδί. Μπορείτε να πάρετε τη βιβλιοθήκη κρυπτογράφησης από: <https://github.com/qistoph/ArduinoAES256> και να την εγκαταστήσετε στο περιβάλλον `Arduino` σας. Στη συνέχεια, πηγαίνετε και δοκιμάστε τον παρακάτω κώδικα!

Κώδικας Arduino με κρυπτογράφηση AES

Υπόδειγμα 4.2: Δείγμα κώδικα `Arduino` χρησιμοποιώντας `AES` για κρυπτογράφηση και αποκρυπτογράφηση ενός μηνύματος.

```
#include "aes256.h"
//Define the AES context
aes256_context ctxt;
//In the setup function you need to
//initialize the AES algorithm providing the
//encryption key
void setup()
{
  Serial.begin(9600);
  Serial.print("Initializing AES256... ");
  //Let's use the following alphanumeric sequence (as integers) as
  encryption key
  uint8_t key[] =
  {
    0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,
```



```

        0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
        0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17,
        0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f
    };
    //You initialize the AES algorithm with the key
    aes256_init(&ctxt, key);
    Serial.println("done");
    char *message = "Arduino, Sensors and the Cloud";
    aes256_encrypt_ecb(&ctxt, (uint8_t*)message);
    Serial.println("encoded message: ");
    Serial.println(message);
    aes256_decrypt_ecb(&ctxt, (uint8_t*)message);
    Serial.println("original message: ");
    Serial.println(message);
    aes256_done(&ctxt);
}
void loop() {
}

```

Επισκόπηση κώδικα

Μπορείτε να ξεκινήσετε το sketch συμπεριλαμβάνοντας την κατάλληλη βιβλιοθήκη, "aes256.h".

```
#include "aes256.h"
```

Στη συνέχεια, ορίσετε ένα αντικείμενο που απαιτείται από την βιβλιοθήκη, το aes256_context:

```
aes256_context ctxt;
```

Το context if χρησιμοποιείται για να κρατήσει τις πληροφορίες, όπως το κλειδί κρυπτογράφησης και προετοιμάζει το μηχανισμό κρυπτογράφησης. Η εκκίνηση θα πραγματοποιηθεί στο πλαίσιο της μεθόδου setup(). Επίσης παρέχετε το συμμετρικό κλειδί εκεί ως μια σειρά από ακέραιους αριθμούς. Το ίδιο κλειδί πρέπει να χρησιμοποιηθεί (και η ίδια μέθοδος φυσικά) στην αποκρυπτογράφηση.

```
void setup()
```

Θα χρησιμοποιήσετε το Serial για την παρακολούθηση των αποτελεσμάτων της μεθόδου κρυπτογράφησης:

```
Serial.begin(9600);
```

Η ακολουθία των ακεραίων ως κλειδί κρυπτογράφησης

```
uint8_t key[] =  
{  
    0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,  
    0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,  
    0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17,  
    0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f  
};
```

Έχουμε προετοιμάσει στη συνέχεια τον αλγόριθμο AES με το κλειδί:

```
aes256_init(&ctxt, key);
```

Γνωρίζετε ότι το Arduino έχει ολοκληρώσει τη διαδικασία προετοιμασίας

```
Serial.println("done");
```

Θα προχωρήσουμε με την κρυπτογράφηση του μηνύματος " Arduino, Sensors and the Cloud". Η βιβλιοθήκη AES256 παρέχει όλες τις απαραίτητες μεθόδους για την κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων. Η προεπιλεγμένη είσοδος κρυπτογράφησης (και η λειτουργία αποκρυπτογράφησης, όπως θα δείτε στη συνέχεια) είναι το πλαίσιο κρυπτογράφησης (που περιέχει το συμμετρικό κλειδί) και το μήνυμα για την κρυπτογράφηση ως ένας δείκτης σε έναν ακέραιο. Επειδή θέλετε να κρυπτογραφήσετε ένα μήνυμα string ως μια ακολουθία από char, θα πρέπει να εκτελέσετε την κατάλληλη σύνθεση uint8_t *.

```
char *message = "Arduino, Sensors and the Cloud";  
aes256_encrypt_ecb(&ctxt, (uint8_t*)message);
```

Η κρυπτογράφηση έχει ολοκληρωθεί και επίσης εκτυπώνετε το κρυπτογραφημένο μήνυμα στο Serial Monitor, έτσι ώστε να μπορείτε να δείτε τι μοιάζει.

```
Serial.println("encoded message: ");  
Serial.println(message);
```

Τώρα προχωρήστε για την αποκρυπτογράφηση του μηνύματος, κάνοντας την αντίθετη διαδικασία και να βεβαιωθείτε ότι μπορείτε να λάβετε το αρχικό μήνυμα.

```
aes256_decrypt_ecb(&ctxt, (uint8_t*)message);  
Serial.println("original message: ");  
Serial.println(message);
```

Τέλος, μπορείτε να χρησιμοποιήσετε τη μέθοδο _done () για να καθαρίσετε τη μνήμη από το πλαίσιο κρυπτογράφησης.

```
aes256_done(&ctxt);
```

Σε αυτό το project η συνάρτηση loop δεν εκτελεί τίποτα συγκεκριμένο. Μπορείτε να το χρησιμοποιήσετε, για την ανάγνωση αισθητήρων και την κρυπτογράφηση τους πριν να τους στείλουμε σε εφαρμογή Cloud. Περισσότερες λεπτομέρειες σχετικά με το πώς να διαβάσετε από αισθητήρες και την επικοινωνία του Arduino σας με το Internet θα βρείτε στα ακόλουθα κεφάλαια!

ΚΕΦΑΛΑΙΟ 7 Εφαρμογές με Arduino

Παράδειγμα Ethernet Client

Θα ξεκινήσετε δημιουργώντας ένα πρόγραμμα client που συνδέει ένα remote web server, που απαντά πίσω τις διευθύνσεις IP στο network router. Ο κώδικας διαβάσει τις διευθύνσεις και τις εμφανίζει στο Serial Monitor.

Ανεβάστε τον ακόλουθο κώδικα στο Ethernet-Arduino:

Κώδικας 7.1: Διάγραμμα απλού Ethernet Client.

```
#include <SPI.h>
#include <Ethernet.h>

// Enter a MAC address for your controller below.
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
//An IP Address and Router settings just in case there is no DHCP available
IPAddress ip(192,168,1, 200);
IPAddress gateway(192,168,1, 1);
IPAddress subnet(255, 255, 255, 0);

//The Web Server you are connecting to
char serverName[] = "whatismyip.org";

// Initialize the Ethernet client library
EthernetClient client;

void setup() {
  // start the serial library:
  Serial.begin(9600);
  // start the Ethernet connection:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP,
    trying to set IP manually");
    Ethernet.begin(mac, ip, gateway, subnet);
  }
  // give the Ethernet shield a second to initialize:
  delay(1000);
  Serial.println("connecting...");

  // if you get a connection, report back via serial:
  if (client.connect(serverName, 80)) {
    Serial.println("connected");
    // Make a HTTP request on the default page:
    client.println("GET / HTTP/1.0");
    client.println();
  }
  else {
    // If you didn't get a connection to the server:
    Serial.println("connection failed");
  }
}

void loop()
```

```

{
// if there are incoming bytes available
// from the server, read them and print them:
while (client.available()) {
char c = client.read();
Serial.print(c);
}
// if the server's disconnected, stop the client:
if (!client.connected()) {
Serial.println();
Serial.println("disconnecting.");
client.stop();

// do nothing forevermore:
for(;;)
;
}
}

```

Στη συνέχεια συνδέστε ένα Ethernet καλώδιο (βεβαιωθείτε ότι είναι συνδεδεμένο στον Router και ότι παρέχει πρόσβαση στο Internet) στη θύρα Ethernet του Arduino σας και ενεργοποιήστε το Serial Monitor στο Arduino IDE. Θα δείτε κάτι σαν την ακόλουθη έξοδο:

```

connecting...
connected
HTTP/1.0 200 OK
Content-Type: text/plain
79.130.123.43
disconnecting.

```

Ανάλυση Κώδικα

Ξεκινάτε το διάγραμμα προσθέτοντας τις Βασικές Ethernet επικεφαλίδες:

```

#include <SPI.h>
#include <Ethernet.h>

```

Εισάγεται χειροκίνητα μια Mac Address για το Ethernet Module (όπως αναφέρθηκε το Arduino δεν είναι ικανό να την καθορίσει από μόνο του). Επιπλέον στην περίπτωση όπου ο network router δεν υποστηρίζει αυτόματη αρίθμηση IP, θα πρέπει επίσης να θέσετε σωστές IP διευθύνσεις, την IP διεύθυνση του router (gateway) και μία subnet mask, με τις κατάλληλες μεταβλητές.

```

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,1, 200);
IPAddress gateway(192,168,1, 1);
IPAddress subnet(255, 255, 255, 0);

```

Στη συνέχεια καθορίζεται το URL του WebServer στον οποίο θα συνδεθείτε. Σε αυτή την περίπτωση είναι ο `whatismyip.org` που καθορίζει την IP που έχει ο network router.

```
char serverName[] = "whatismyip.org";
```

Ακολουθως καθορίζεται την κύρια μεταβλητή του Ethernet που θα χρησιμοποιηθεί για να κάνετε τις απαραίτητες ενέργειες δικτύου (να συνδεθείτε στο δίκτυο, στο server κ.τ.λ)

```
EthernetClient client;
```

Η λειτουργία setup() ως συνήθως παρέχει την απαραίτητη αρχικοποίηση. Ορίζετε την σειριακή θύρα και ξεκινάτε μία σύνδεση Ethernet στο δίκτυό σας με την καθορισμένη Mac address.

```
void setup() {  
// start the serial library:  
Serial.begin(9600);  
// start the Ethernet connection:  
if (Ethernet.begin(mac) == 0) {  
Serial.println("Failed to configure Ethernet using DHCP,  
trying to set IP manually");  
Ethernet.begin(mac, ip, gateway, subnet);  
}
```

Στην περίπτωση που η υπηρεσία DHCP δεν είναι διαθέσιμη στο τοπικό router, η σύνδεση ethernet θα οριστεί από την καθορισθείσα χειροκίνητα διεύθυνση IP. Όπως όταν ρυθμίζετε τον υπολογιστή σας, μπορείτε να καθορίσετε την διεύθυνση IP του router και τη Subnet mask.

Στη συνέχεια προσθέτετε ένα δευτερόλεπτο καθυστέρηση, ώστε να δώσετε χρόνο στη μονάδα Ethernet να αρχικοποιηθεί κατάλληλα. Ακολουθως προχωράτε με τη σύνδεση της URL διεύθυνσης του server στο port 80.

```
delay(1000);  
if (client.connect(serverName, 80)) {  
client.println("GET / HTTP/1.0");  
client.println();  
}
```

Ελέγχετε για μία επιτυχή σύνδεση (client.connect() θα επιστρέψει true ή false) και στη συνέχεια στέλνετε ένα request στο Web Server. Σύμφωνα με το πρωτόκολλο HTTP θα πρέπει να υπάρχει ένα ερώτημα GET ακολουθούμενο απο το path της πηγής (το ορισμένο directory page ορισμένο με '/' σε ευατή την περίπτωση) ακολουθούμενο απο την έκδοση του πρωτοκόλλου (HTTP/1.0). Θα χρειαστεί επίσης να στείλετε μία μαύρη γραμμή για να πείτε στον browser ότι έχετε τερματίσει το ερώτημα και περιμένετε απάντηση.

Την απάντηση του Server θα διαχειριστεί το loop() function. Το διάγραμμα ελέγχει για δεδομένα απο τον server με τη μέθοδο client.available() και στη συνέχεια διαβάζει κάθε χαρακτήρα που έρχεται απο τον server χρησιμοποιώντας την client.read().

```
void loop() {  
while (client.available()) {  
char c = client.read();  
Serial.print(c);  
}
```

Οι ακόλουθες γραμμές κώδικα απλώς ελέγχουν εάν ο server έχει αποσυνδεθεί, πράγμα το οποίο σημαίνει πως δεν υπάρχουν άλλα δεδομένα και οποιαδήποτε εναλλαγή δεδομένων που να έχει μείνει υπόλοιπο, οπότε και στέλνει το διάγραμμα σε ένα ατέρμον loop ώστε να μην υπάρξει οποιαδήποτε άλλη αλληλεπίδραση με το remote server.

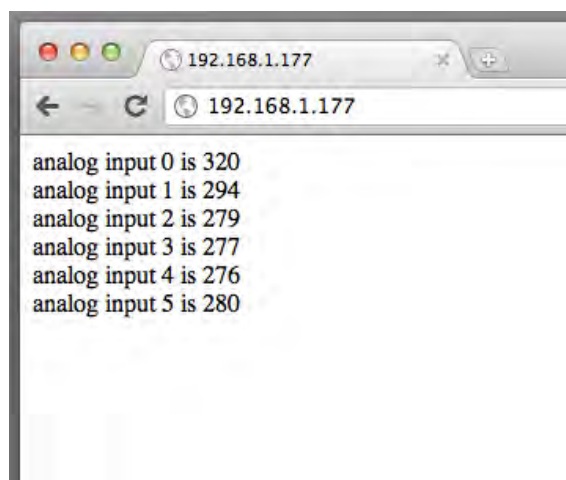
```
if (!client.connected()) {  
  Serial.println();  
  Serial.println("disconnecting.");  
  client.stop();  
  // do nothing forevermore:  
  for(;;)  
  ;  
}
```

Ένα απλό παράδειγμα Ethernet Server

Γιατί θα θέλατε να δημιουργήσετε έναν Server στο Arduino σας, αντί αν στείλετε απλά τα δεδομένα μέσω internet; Μια καλή απάντηση είναι ότι μπορεί να χρειαστείτε ένα εύκολο τρόπο ώστε να διαβάσετε δεδομένα από αισθητήρες συνδεδεμένους στον υπολογιστή σας (Pc, Laptop, Smartphone, ή οτιδήποτε άλλο), όταν είστε σπίτι χωρίς την ανάγκη εξωτερικής σύνδεσης στο internet.

Ο καλύτερος τρόπος για να απεικονίσετε αυτό το χαρακτηριστικό της Arduino Ethernet Library είναι αν χρησιμοποιήσετε τον προκαθορισμένο WebServer παράδειγμα που υπάρχει εκεί. Ανοίξετε το κείμενο επιλέγοντας File->Ethernet->WebServer. Αυτό το κείμενο απεικονίζει έναν WebServer στο Arduino σας που απεικονίζει ότι διαβάζει στις αναλογικές θύρες της πλακέτας σε html φόρμα, ώστε να μπορεί να απεικονιστεί από τον browser.

Σιγουρέψτε ότι η προεπιλεγθείσα διεύθυνση server (192.168.1.177) είναι διαθέσιμη στο τοπικό σας δίκτυο ή αλλάξτε την με κάποια άλλη και ανεβάστε τον κώδικα στην πλακέτα Ethernet. Μόλις ανέβει ανοίξετε τον browser και πληκτρολογήστε την Ip που επιλέχθηκε πιο πάνω. Θα δείτε κάτι όπως η ακόλουθη έξοδος:



Εικόνα 7.2: ο Browser συνδεδεμένος με το Arduino WebServer απεικονίζοντας τις αναλογικές τιμές των θυρών.

Ας δούμε τον κώδικα βήμα-βήμα. Αρχικά οι ουσιώδεις Ethernet και SPI κεφαλίδες (header) τοποθετούνται, ομοίως με το παράδειγμα του client.

```
#include <SPI.h>
#include <Ethernet.h>
```

Ομοίως καθορίζουμε την Mac address του Ethernet Module.

```
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
```

Ο Web Server θα πρέπει να έχει μία IP την οποία και θα γνωρίζετε ώστε να μπορέσετε να έχετε πρόσβαση απο τον browser σας. Γι'αυτό το λόγο δεν ζητάμε διεύθυνση IP απο το δίκτυο χρησιμοποιώντας την υπηρεσία DHCP, αλλά την ορίζετε χειροκίνητα.

```
IPAddress ip(192,168,1, 177);
```

Όταν αρχικοποιήστε τη βιβλιοθήκη του Ethernet Server καθορίζετε τη θύρα που θέλετε να χρησιμοποιήσετε (σε αυτή την περίπτωση τη θύρα 80 ως προεπιλογή για το πρωτόκολλο HTTP):

```
EthernetServer server(80);
```

Στη συνάρτηση setup() ορίζεται τη MAC μαζί με τη διεύθυνση IP. Επίσης εκκινείτε τον Server ώστε να περιμένει τις συνδέσεις χρησιμοποιώντας την εσωτερική συνάρτηση Ethernet begin():

```
{
// start the Ethernet connection and the server:
Ethernet.begin(mac, ip);
server.begin();
}
```

Η αποδοχή της σύνδεσης (π.χ με τον browser) απεικονίζεται στην συνάρτηση loop(). Για να γίνει η διαχείριση των συνδέσεων και των αιτημάτων, δημιουργείτε ένα αντικείμενο Ethernet Client. Το αντικείμενο αυτό συνεργάζεται με το server άμεσα (εάν έχει δημιουργηθεί επιτυχώς στα προηγούμενα βήματα).

```
void loop()
{
EthernetClient client = server.available();
if (client) {
boolean currentLineIsBlank = true;
```

Το ακόλουθο loop ενεργοποιείται όταν ο client συνδεθεί. Όταν υπάρχουν δεδομένα τα διβάζει και ελέγχει για κενή γραμμή (το οποίο σύμφωνα με το πρωτόκολλο HTTP σημαίνει ότι ο browser έχει ολοκληρώσει το αίτημα). Σε αυτή την περίπτωση ο Server απαντά με την προκαθορισμένη κεφαλίδα HTTP ("HTTP/1.1 200 OK") δείχνοντας ότι η αίτηση έχει λειφθεί επιτυχώς. Επίσης θα τυπώσει μία γραμμή ορίζοντας τον τύπο των περιεχομένων που θα σταλούν στον browser ("text/html").

```

while (client.connected()) {
  if (client.available()) {
    char c = client.read();
    if (c == '\n' && currentLineIsBlank) {
      // send a standard http response header
      client.println("HTTP/1.1 200 OK");
      client.println("Content-Type: text/html");
      client.println();

```

Στη συνέχεια ο server στέλνει στον browser την τιμή κάθε αναλογικής εισόδου, σε νέα γραμμή. Για να το κάνει αυτό, γράφει στο τέλος κάθε τιμής αισθητήρα τον html χαρακτήρα γραμμής ("
").

```

for (int analogChannel = 0; analogChannel < 6;
    analogChannel++) {
  client.print("analog input ");
  client.print(analogChannel);
  client.print(" is ");
  client.print(analogRead(analogChannel));
  client.println("<br />");
}

```

Τελικά δίνει στον browser χρόνο ώστε να λάβει τα δεδομένα και να ολοκληρώσει τη σύνδεση.

```

delay(1);
client.stop();

```

Ελέγχοντας το Arduino από περιβάλλον Android

Στη παρούσα εφαρμογή θα διερευνήσουμε το χειριστή καταστάσεων του ADB και θα το χρησιμοποιήσουμε ώστε να κάνουμε το Arduino μας να λαμβάνει εντολές από ένα τηλέφωνο Android. Η πλατφόρμα Android επιτρέπει στους δημιουργούς προγραμμάτων να επικοινωνούν μέσω SMS με τις εφαρμογές τους. Αυτό είναι μια σπουδαία ευκαιρία να ελέγξετε τις εξόδους του Arduino μέσω του τηλεφώνου σας, απλά στέλνοντας γραπτά μηνύματα.

Θέλοντας να δημιουργήσουμε κάτι πιο χρήσιμο από το να ανάβουμε απλώς ένα led, θα συνδέσουμε στο Arduino έναν διακόπτη ρελέ. Όπως εξηγήσαμε σε προηγούμενο κεφάλαιο, το ρελέ είναι ένας μηχανικός διακόπτης που ενεργοποιείται χρησιμοποιώντας ηλεκτρικό ρεύμα. Χρησιμοποιείται για να ελέγξει άλλα κυκλώματα (κυρίως επειδή υπάρχει αδυναμία στη διαχείριση της τάσης και του ρεύματός τους). Αποτελείται από έναν ηλεκτρικό μαγνήτη (πηνίο) και έναν μοχλοδιακόπτη. Όταν ηλεκτρικό ρεύμα διαρρέει το πηνίο, δημιουργείται μαγνητικό πεδίο που αλλάζει την κατάσταση του μοχλοδιακόπτη, ανοίγοντας ή κλείνοντας το άλλο κύκλωμα. Όταν απομακρύνεται η τάση από το κύκλωμα, το μαγνητικό πεδίο που έχει απομείνει δημιουργεί μια αιχμή τάσης που μπορεί να προκαλέσει βλάβη στο υπόλοιπο κύκλωμα. Γι'αυτό τον λόγο χρησιμοποιούμε πάντοτε μια δίοδο στα άκρα του πηνίου. Μπορείτε να ελέγξετε το κύκλωμα για περισσότερες λεπτομέρειες. Με τα ρελέ μπορείτε να ελέγξετε πολλά άλλα κυκλώματα και συσκευές, όπως φώτα, καλοριφέρ και βασικά οτιδήποτε ενεργοποιείται χρησιμοποιώντας διακόπτη.

Η εφαρμογή αποτελείται από δύο τμήματα κώδικα. Το ένα τμήμα είναι αυτό που θα λαμβάνει δεδομένα από τη συσκευή Android και θα ελέγχει ένα ρελέ, ενώ το άλλο θα δέχεται δεδομένα από τα εισερχόμενα SMS και θα στέλνει flags στον ADB server. Ας δούμε κατ'αρχήν το κύκλωμα της εφαρμογής.

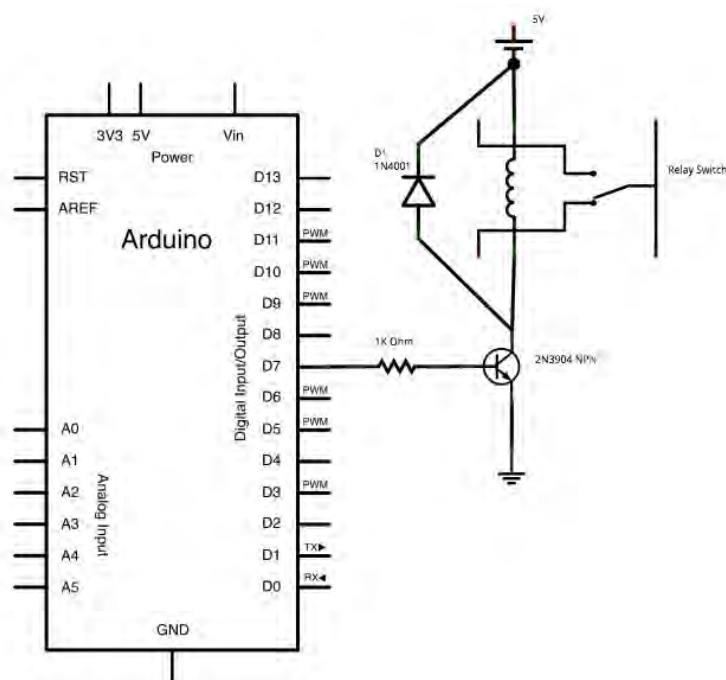
Το Κύκλωμα

Γι'αυτή την εφαρμογή, θα χρειαστείτε τα ακόλουθα:

- Ένα διακόπτη ρελέ
- Ένα NPN τρανζίστορ 2N3904
- Μια δίοδο 1N4001
- Μια αντίσταση 1KΩ
- Μια πλακέτα Arduino
- Ένα τηλέφωνο Android (έκδοσης 1.5 ή νεότερο) με μια ενεργή κάρτα SIM και γνωστό αριθμό
- Και (προαιρετικά) ένα breadboard για τη σύνδεση των εξαρτημάτων με το Arduino.

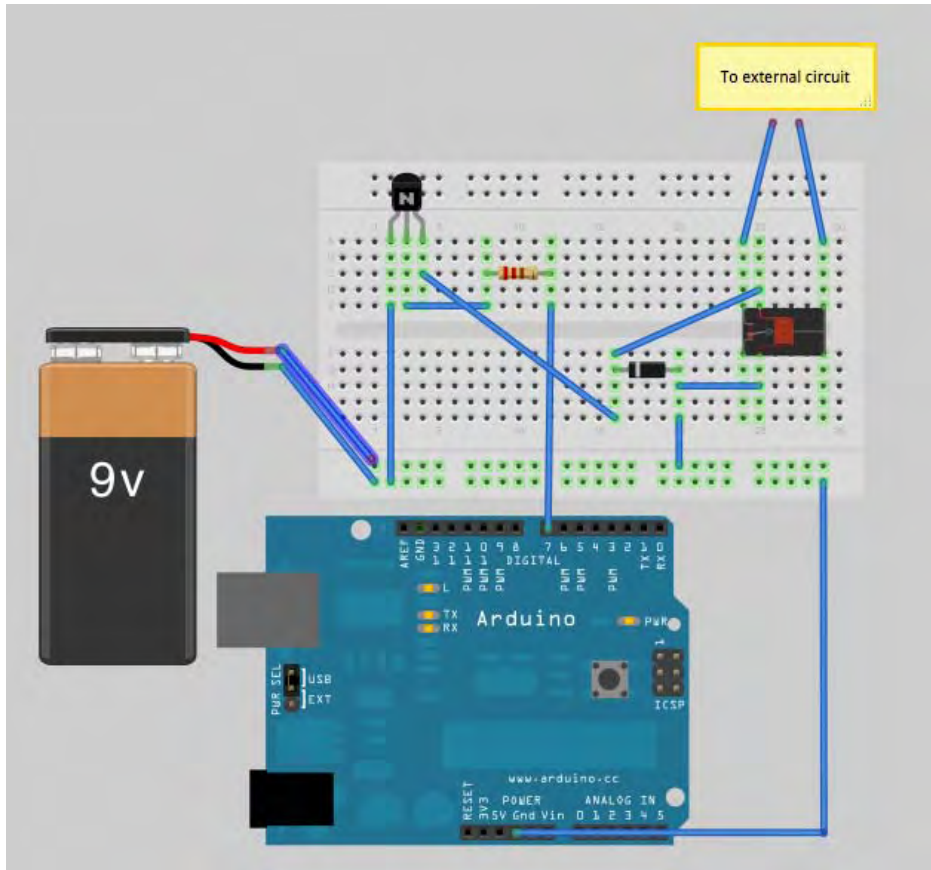
Θα χρειαστείτε επίσης και ένα άλλο κινητό τηλέφωνο για να στέλνετε SMS ή να κανετε χρήση on-line SMS πλατφόρμας.

Μελετήστε το σχηματικό της εικόνας 6-11. Το ρελέ που χρησιμοποιείται έχει τάση 5V, που σημαίνει ότι τροφοδοτώντας το πηνίο με 5V ενεργοποιεί το ρελέ. Ο ακροδέκτης εξόδου του Arduino μπορεί να δώσει αυτή τη τάση, το ρεύμα όμως δεν επαρκεί με αποτέλεσμα να πρέπει να χρησιμοποιηθεί το τρανζίστορ 2N3904 για να παρέχει περισσότερο ρεύμα. Η ψηφιακή έξοδος 7 του Arduino ελέγχει τη λειτουργία του τρανζίστορ. Όταν περάσει σε υψηλή κατάσταση (ψηφιακό «1»), ενεργοποιεί το τρανζίστορ και κλείνει κύκλωμα με τη γείωση, με αποτέλεσμα να διαρρέει ρεύμα το πηνίο του ρελέ. Αυτό το οδηγεί στο να αλλάξει κατάσταση, κλείνοντας (ενεργοποιώντας) το δευτερεύον κύκλωμα.



Εικόνα 6-11 : Το σχηματικό του κυκλώματος. Η ψηφιακή έξοδος του Arduino ελέγχει τη λειτουργία του τρανζίστορ. Όταν περάσει σε υψηλή κατάσταση, ενεργοποιεί το τρανζίστορ και κλείνει κύκλωμα με τη γείωση, με αποτέλεσμα να διαρρέει ρεύμα το πηνίο του ρελέ, ενεργοποιώντας το δευτερεύον κύκλωμα.

Στο παρακάτω σχήμα απεικονίζεται το πλήρες κύκλωμα με breadboard και το Arduino Uno. Θα πρέπει να αντικαταστήσετε την Uno με μια πλακέτα Seeeduino ADK Main Board για να κάνετε χρήση του USB Host Shield.



Εικόνα 6-12 : Η απεικόνιση του κυκλώματος. Όταν η έξοδος 7 τεθεί σε υψηλή κατάσταση, το κύκλωμα θα κλείσει με τη γείωση και η μπαταρία 9V θα ενεργοποιήσει το ρελέ. Έτσι το ρελέ θα αλλάξει κατάσταση, ενεργοποιώντας το εξωτερικό κύκλωμα.

Ο κώδικας Arduino

```
#include <SPI.h>
#include <Adb.h>
//relay switch pin
#define RELAY_PIN 7
// Adb connection.
Connection * connection;
// Event handler for the shell connection.
void adbEventHandler(Connection * connection, adb_eventType event, uint16_t
length, uint8_t * data)
{
    int i;
    // Data packet contains one byte, the flag for turning on
    or off the relay
    if (event == ADB_CONNECTION_RECEIVE)
    {
        if(data[0]==1) {
            digitalWrite(RELAY_PIN, HIGH);
        }
    }
}
```

```

        }
        if(data[0]==0) {
            digitalWrite(RELAY_PIN, LOW);
        }
    }
}
void setup()
{
    // Initialise serial port
    Serial.begin(57600);
    //Set pin to output for controlling the relay
    pinMode(RELAY_PIN, OUTPUT);
    // Initialise the ADB subsystem.
    ADB::init();
    // Open an ADB stream to the phone's shell. Auto-reconnect
    connection = ADB::addConnection("tcp:4568", true,
    adbEventHandler);
}
void loop()
{
    // Poll the ADB subsystem.
    ADB::poll();
    delay(100);
}

```

Ο κώδικας Arduino – Ανάλυση κώδικα.

Ξεκινάτε θέτοντας τις απαραίτητες βιβλιοθήκες για την επικοινωνία με τον ADB:

```

#include <SPI.h>
#include <Adb.h>

```

Επίσης καθορίζετε τον ακροδέκτη του Arduino στον οποίο είναι συνδεδεμένο το ρελέ:

```

//relay switch pin

```

Στη συνέχεια ορίζετε το αντικείμενο σύνδεσης με τον ADB:

```

Connection * connection;
// Event handler for the shell connection.
void adbEventHandler(Connection * connection, adb_eventType event, uint16_t
length, uint8_t * data)
{
    int i;
    // Data packet contains one byte, the flag for turning on
    or off the relay

```

```

        if (event == ADB_CONNECTION_RECEIVE)
        {
            if(data[0]==1) {
                digitalWrite(RELAY_PIN, HIGH);
            }
            if(data[0]==0) {
                digitalWrite(RELAY_PIN, LOW);
            }
        }
    }
}

void setup()
{
    // Initialise serial port
    Serial.begin(57600);
    //Set pin to output for controlling the relay
    pinMode(RELAY_PIN, OUTPUT);
    // Initialise the ADB subsystem.
    ADB::init();
    // Open an ADB stream to the phone's shell. Auto-reconnect
    connection = ADB::addConnection("tcp:4568", true,
    adbEventHandler);
}

void loop()
{
    // Poll the ADB subsystem.
    ADB::poll();
    delay(100);
}

```

Ο κώδικας Android

Λόγω περιορισμού στον χώρο, δεν συμπεριλήφθηκε ο κώδικας στο εγχειρίδιο. Μπορείτε να τον βρείτε στο <http://www.buildinginternetofthings.com>. Το κεφάλαιο που ακολουθεί περιλαμβάνει την ανάλυση του πλήρους κώδικα.

Ο κώδικας Android – Ανάλυση κώδικα

Όπως γίνεται συνήθως ξεκινάτε με τις κατάλληλες βιβλιοθήκες. Πρώτα είναι η βιβλιοθήκη MicroBridge, την οποία θα χρησιμοποιήσετε για το server object.

```
import org.microbridge.server.Server;
```

Στη συνέχεια θα πρέπει να ορίσετε κάποιες εξειδικευμένες βιβλιοθήκες Android, που είναι βασικές για την ενσωμάτωση των εισερχομένων μηνυμάτων SMS και κάποιες άλλες για την απεικόνιση μικρών μηνυμάτων στην οθόνη.


```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsMessage;
import android.util.Log;
import android.widget.Toast;
```

Γι' αυτή την εφαρμογή απαιτείται μόνο μια κλάση που δρά ως ακροατής SMS και που μπορεί να έχει πρόσβαση στα εισερχόμενα γεγονότα και τα περιεχόμενα του τηλεφώνου. Αυτό γίνεται από την κλάση Android BroadcastReceiver. Έχετε υπόψιν ότι δεν χρειάζεται να δημιουργήσετε κανένα περιβάλλον επικοινωνίας γι' αυτή την εφαρμογή, πράγμα το οποίο σημαίνει ότι όταν εγκαθιστάται αυτή η εφαρμογή, αυτόματα λαμβάνει τα εισερχόμενα γεγονότα SMS.

```
public class SMSReceiver extends BroadcastReceiver
```

Ξεκινάτε μέσα στην κυρίως κλάση ορίζοντας το κεντρικό αντικείμενο του Server.

```
Server server = null;
```

Συνεχίζετε ενσωματώνοντας τη μέθοδο Void onReceive(). Αυτή είναι απαραίτητη να υπάρχει στον κώδικα, μιας και αυτή η κλάση επεκτείνει την κλάση BroadcastReceiver.

```
@Override
public void onReceive(Context context, Intent intent)
```

Όπως δείχνει το όνομα, αυτή η μέθοδος ενεργοποιείται κάθε φορά που ένα SMS λαμβάνεται. Έτσι ο ακόλουθος κώδικας λαμβάνει το SMS ελέγχει τα περιεχόμενα. Επιπρόσθετα, ελέγχει για μια εντολή (μια σημαία 'true'/'false') και στέλνει τη δική του σημαία στο Arduino μέσω του ADB server.

Ξεκινάτε παίρνοντας ένα 'Bundle' αντικείμενο από τη κύρια δραστηριότητα (Android internal) που ελέγχει για γεγονότα SMS:

```
Bundle bundle = intent.getExtras();
SmsMessage[] msgs = null;
String flag = "0";
```

Στην περίπτωση που υπάρχουν περιεχόμενα, σημαίνει ότι υπάρχει νέο SMS:

```
if (bundle != null)
```

Λαμβάνετε το μήνυμα SMS:

```
Object[] pdus = (Object[]) bundle.get("pdus");
```

Στην περίπτωση που έχουν ληφθεί περισσότερα του ενός μηνύματα, γίνεται ένας κύκλος σε όλα τα μηνύματα, που καταλήγει στο κυρίως κείμενο και στα περιεχόμενά του:

```
msgs = new SmsMessage[pdus.length];  
//for each message get the main body content  
for (int i=0; i<msgs.length; i++){  
    msgs[i] =  
    SmsMessage.createFromPdu((byte[])pdus[i]);  
    flag = msgs[i].getMessageBody().toString();
```

Τα περιεχόμενα σώζονται στη μεταβλητή flag String. Ελέγχετε το flag για εντολές και επικοινωνείτε με το Arduino κατάλληλα:

```
if(flag.equals("true") || flag.equals("false")) {  
    try{
```

Καθορίζετε το ADB TCP server χρησιμοποιώντας το ίδιο port που χρησιμοποιήθηκε στο ADK Main Board (4568):

```
server = new Server(4568);  
server.start();
```

Ο server ξεκινά σε διαφορετικό tread και χρειάζεται να του δώσετε λίγο χρόνο να εκκινήσει προτού στείλετε δεδομένα στο Arduino:

```
Thread.sleep(3000);
```

Στη συνέχεια στέλνετε δεδομένα σε μορφή byte. Το μόνο που χρειάζεται είναι να στείλετε '1' ή '0', ανάλογα με το SMS flag των εισερχομένων μηνυμάτων. Το Arduino θα τα διαβάσει, βάσει του προηγούμενου κώδικα.

```
byte data;  
if(flag.equals("true")) data = 1;  
else data = 0;  
server.send(new byte[] {(byte) data});
```

Τελικά, θα πρέπει να σταματήσετε τον server. Αυτό το κάνετε ώστε στο επόμενο εισερχόμενο SMS να επανεκκινήσει η διαδικασία, ξεκινώντας από τη μέθοδο onReceive.

```
server.stop();
```

Επίλογος

Όπως το περιγράφει ο δημιουργός του, το Arduino είναι μια open source πλατφόρμα «προτυποποίησης» ηλεκτρονικών βασισμένη σε ευέλικτο και εύκολο στη χρήση hardware και software που προορίζεται για οποιονδήποτε έχει λίγη προγραμματιστική εμπειρία, στοιχειώδεις γνώσεις ηλεκτρονικών και ενδιαφέρεται να δημιουργήσει διαδραστικά αντικείμενα ή περιβάλλοντα.

Στην ουσία, πρόκειται για ένα ηλεκτρονικό κύκλωμα που βασίζεται σε μικροελεγκτή ATmega της Atmel και του οποίου όλα τα σχέδια, καθώς και το software που χρειάζεται για την λειτουργία του, διανέμονται ελεύθερα και δωρεάν ώστε να μπορεί να κατασκευαστεί από τον καθένα (απ' όπου και ο περιέργος -για hardware- χαρακτηρισμός «ανοικτού κώδικα»). Αφού κατασκευαστεί, μπορεί να συμπεριφερθεί σαν ένας μικροσκοπικός υπολογιστής, αφού ο χρήστης μπορεί να συνδέσει επάνω του πολλαπλές μονάδες εισόδου/εξόδου και να προγραμματίσει τον μικροελεγκτή να δέχεται δεδομένα από τις μονάδες εισόδου, να τα επεξεργάζεται και να στέλνει κατάλληλες εντολές στις μονάδες εξόδου.

Στην παρούσα διπλωματική διατριβή επιλέχθηκαν δειγματοληπτικά κάποια από τα πολλά στοιχεία που αφορούν στην παρουσίαση και την υλοποίηση εφαρμογών στην πλατφόρμα Arduino. Είναι σαφές πως δεν γίνεται στα πλαίσια μιας εργασίας να αναπτυχθούν όλες οι παράμετροι της χρήσης μιας open source πλατφόρμας, πολλώ δε μάλλον της δημοφιλέστερης και αυτής που περιλαμβάνει έναν τεράστιο αριθμό εφαρμογών τόσο στο κομμάτι του hardware (βασικές πλακέτες, shields και διάφοροι αισθητήρες & ενεργοποιητές), όσο και στο κομμάτι του software (με κυριολεκτικά άπειρες διαφοροποιήσεις κώδικα).

Το Arduino βέβαια, δεν είναι ούτε ο μοναδικός, ούτε και ο καλύτερος δυνατός τρόπος για την δημιουργία μιας οποιασδήποτε διαδραστικής ηλεκτρονικής συσκευής. Όμως το κύριο πλεονέκτημά του είναι η τεράστια κοινότητα που το υποστηρίζει και η οποία έχει δημιουργήσει, συντηρεί και επεκτείνει μια ανάλογου μεγέθους online γνωσιακή βάση. Έτσι, παρότι ένας έμπειρος ηλεκτρονικός μπορεί να προτιμήσει διαφορετική πλατφόρμα ή εξαρτήματα ανάλογα με την εφαρμογή που έχει στον νου του, το Arduino, με το εκτενές documentation, καταφέρνει να κερδίσει όλους αυτούς των οποίων οι γνώσεις στα ηλεκτρονικά είναι περιορισμένες.