ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ασύρματος έλεγχος του οχήματος 3-in-1 all terrain robot OWI 536 μέσω Arduino και εφαρμογής android – υλοποίηση εκπαιδευτικών ασκήσεων.

Γραμματικός Παναγιώτης

Εισηγητής: Μαστοροκώστας Πάρις

ΑΘΗΝΑ, ΟΚΤΩΒΡΙΟΣ 2018

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ασύρματος έλεγχος του οχήματος 3-in-1 all terrain robot OWI 536 μέσω Arduino και εφαρμογής android - υλοποίηση εκπαιδευτικών ασκήσεων

Παναγιώτης Β. Γραμματικός
Α.Μ. 42931

Εισηγητής: Δρ Πάρις Μαστοροκώστας, Καθηγητής

Εξεταστική Επιτροπή: Μαστοροκώστας Πάρις, Έλληνας Ιωάννης, Αμοργίνος Ιωάννης

Ημερομηνία εξέτασης : 10/10/2018

ΑΘΗΝΑ, ΟΚΤΩΒΡΙΟΣ 2018

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος **Παναγιώτης Γραμματικός** του **Βασιλείου**, με αριθμό μητρώου: **42931** φοιτητής του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών του Πανεπιστημίου Δυτικής Αττικής, πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασή της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσής της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, κύριο Πάρι Μαστοροκώστα, για την πολύτιμη βοήθειά του καθ΄όλη τη διάρκεια της εκπόνησης της παρούσας πτυχιακής εργασίας.

Ευχαριστώ, επίσης, όλους τους καθηγητές μου για τις γνώσεις και δεξιότητες με τις οποίες με εφοδίασαν τα χρόνια των σπουδών μου.

Ευχαριστώ, τέλος, την οικογένειά μου, που με στήριξε και κατά τη διάρκεια της φοίτησής μου και κατά τη συγγραφή της εργασίας μου.

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία ασχολείται με τον έλεγχο ενός ρομποτικού οχήματος μέσω Arduino και εφαρμογής Android, ενώ συμπληρώνεται με την υλοποίηση εκπαιδευτικών ασκήσεων για το εργαστήριο του μαθήματος της Ρομποτικής. Για να γίνει αυτό εφικτό, έχουν αφαιρεθεί ο επεξεργαστής και η πλακέτα ελέγχου που συνοδεύουν το όχημα OWI 536 και έχουν αντικατασταθεί από Arduino. Έχει, επίσης, συγγραφεί ο κώδικας Android για τον εξ αποστάσεως έλεγχο του οχήματος από κινητή συσκευή μέσω Bluetooth. Το εκπαιδευτικό όφελος της εργασίας είναι η εμβάθυνση στα αντικείμενα της Ρομποτικής, των μικροεπεξεργαστών, καθώς και η ανάπτυξη της εφαρμογής Android. Έτσι, των εφαρμογών έχει προηγηθεί η απαιτούμενη, κατά τη γνώμη μας, θεωρητική ανάλυση των στοιχείων που χρησιμοποιήθηκαν κατά την υλοποίησή τους (κινητήρες, αισθητήρια, υπολογιστική πλατφόρμα Arduino, Android, Bluetooth κλπ).

ABSTRACT

The present thesis deals with the control of a robotic vehicle through Arduino and an Android application, as a tool for educational purposes regarding the robotics laboratory. To do so, the processor and control board that accompanied the OWI 536 vehicle, have been removed and replaced by an Arduino board. Android code has also been written, to remotely control the vehicle from a mobile device via Bluetooth. The educational benefit of this work is to gain insight into the fields of robotics, microprocessors, and Android application development. Thus, the applications have been preceded by the theoretical analysis of the elements used in their implementation (mobile, sensors, Arduino, Android, Bluetooth, etc.).

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Ρομποτική
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Arduino, Android, Bluetooth, ρομπότ, αισθητήρια

SCIENTIFIC AREA: Robotics
KEY WORDS: Arduino, Android, Bluetooth, robot, sensors

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

ΚΑΤΑΛΟΓΟΣ ΣΥΝΤΜΗΣΕΩΝ

AC	Alternating Current
ACL	Asynchronous Connection-Less
ADC	Analog to Digital Converter
ADT	Android Development Tools
API	Application Programming Interface
APK	Android PacKage
AREF	Analog Reference
ASPI	Advanced SCSI Programming Interface
BLE	Bluetooth Low Energy
DC	Direct Current
DMP	Digital Motion Processor
EEPROM	Electrically Erasable Programmable Read-Only Memory
HAL	Harware Abstraction Layer
FEC	Forward Error Connection
FHSS	FrequencyHopping Spread Spectrum
GPS	Global Positioning System
GUI	Graphical User Interface
IC	Integrated Circuit
ICSP	In-Circuit Serial Programming
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IOREF	Input/output Reference
IoT	Internet of Things
IR	Infrared
ISM	Industrial,Scientific and Medical
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LSI/VLSI	Large/ Very Large Scale Integrated
M/E	Μικροεπεξεργαστής
MISO	Master In Slave Out

MOSI	Master Out Slave In
MP	Megapixel
NDK	Native Development Kit
OpenGLES	Open Graphics Library for Embedded Systems
OSX	Operating System 10
PDA	Personal Digital Assistant
PIN	Personal Indentification Number
POSIX	Portable Operating System Interface for Unix
PWM	Pulse Width Modulation
RF	Radio Frequency
SCK	Serial Clock
SCL	Serial Clock
SCO	Synhronous Connection Oriented
SDA	Serial Data Access
SIG	Special Interest Group
SPI	Serial Peripheral Interface
SPP	Serial Port Profile
SRAM	Static Random Access Memory
TDMA	Time Division Multiple Access
TTL	Transistor–transistor logic
TWI	Two Wire Interface
WPAN	Wireless Personal Area Network
XML	Extensible Markup Language

ΚΕΦΑΛΑΙΟ 1ο: ΕΙΣΑΓΩΓΗ

1.1 Γενικά

Το αποτέλεσμα της παρούσας πτυχιακής εργασίας προήλθε από τις γνώσεις που αποκτήθηκαν παρακολουθώντας τα μαθήματα ΡΟΜΠΟΤΙΚΗ και ΜΗΧΑΤΡΟΝΙΚΗ και είναι η διαμόρφωση και ο ασύρματος έλεγχος ενός εκπαιδευτικού ρομποτικού οχήματος OWI-536. Ο έλεγχος του οχήματος στη εργασία αυτή,θα γίνει με την υπολογιστική πλατφόρμα Arduino Uno R3,έτσι ώστε να τελεί τις παρακάτω συγκεκριμένες κινήσεις:

1.Κίνηση εμπρός –πίσω,δεξιά –αριστερά,πάνω –κάτω,ανυψωτική περόνη.

2.Κίνηση και παρακολούθηση των χαρακτηριστικών αυτής με τοποθέτηση και συλλογή μετρήσεων από αισθητήρα επιταχυνσιομέτρου.

3.Κίνηση σε προκαθορισμένη μαύρη γραμμή με χρήση υπερύθρου (infrared) αισθητήρα KY033.

4.Αποφυγή εμποδίων με τη χρήση αισθητήρα απόστασης HC-SR04.

Οι κινήσεις του οχήματος αυτού θα ελέγχονται από το χρήστη μέσω εφαρμογής Android εγκατεστημένης σε κινητή συσκευή (Smartphone, tablet), ενώ η επικοινωνία θα γίνεται μέσω της τεχνολογίας Bluetooth. Έτσι, στην εργασία θα παρουσιαστούν βασικές προαπαιτούμενες πληροφορίες που είναι απαραίτητες για τη υλοποίηση εφαρμογών,που θα οδηγήσουν στην εκτέλεση των προαναφερθέντων κινήσεων.

1.2. Σκοπός και στόχοι της εργασίας

Το κίνητρο για την εκπόνηση της παρούσας εργασίας δόθηκε από τον καθηγητή μου, κύριο Μαστοροκώστα Πάρι, ο οποίος δέχτηκε να με κατευθύνει σε αυτήν και μου πρότεινε να ασχοληθώ με τη δημιουργία ασκήσεων-εφαρμογών ρομποτικής, που θα αξιοποιηθούν για τη διδασκαλία του αντίστοιχου εργαστηρίου.

Αφού καθορίστηκε ο βασικός σκοπός της εργασίας, όπως αναφέρθηκε παραπάνω, τέθηκαν και οι επιμέρους στόχοι αυτής, οι οποίοι σχετίζονται με την εμβάθυνση στο αντικείμενο της μηχατρονικής και ρομποτικής και ιδιαίτερα στην εφαρμογή του Arduino και του συστήματος Android, τα οποία αποτελούν μερικά από τα πιο υψηλά πεδία της επιστήμης των μηχανικών πληροφορικής.

1.3. Δομή της εργασίας

Στο δεύτερο κεφάλαιο, παρουσιάζονται οι κινητήρες DC, τα βασικά μέρη τους και η λειτουργία τους, καθώς το όχημα φέρει μικρής ισχύος τέτοιου είδους κινητήρες.

Στο τρίτο κεφάλαιο, γίνεται αναφορά στους μικροεπεξεργαστές, στους μικροελεγκτές με έμφαση στην υπολογιστική πλατφόρμα Arduino και ιδιαίτερα στο Arduino Uno3, το οποίο και θα χρησιμοποιήσουμε.

Το τέταρτο κεφάλαιο είναι αφιερωμένο στο λειτουργικό σύστημα Android,τις εκδόσεις και τα βασικά χαρακτηριστικά του,έτσι ώστε να γίνει κατανοητός και ο ρόλος του στις εφαρμογές μας.

Το πέμπτο κεφάλαιο αναφέρεται στην τεχνολογία Bluetooth και τις εφαρμογές της, μία από τις οποίες υλοποιούμε στην παρούσα εργασία.

Φυσικά, εκτός αυτών, παρουσιάζεται στο έκτο κεφάλαιο και ο υπόλοιπος εξοπλισμός που χρησιμοποιήθηκε στο τηλεχειριζόμενο όχημά μας. Πρόκειται τόσο για τα αισθητήρια, όσο και για τα υπόλοιπα περιφερειακά στοιχεία (διακόπτες, μπαταρίες κλπ.) που συνέβαλαν στην ανάδειξη των δυνατοτήτων της πλατφόρμας Arduino.

Το έβδομο κεφάλαιο περιλαμβάνει την πορεία που ακολουθήθηκε για την κατασκευή του οχήματος με τρόπο ώστε να τελεί τις επιθυμητές κινήσεις:

Το πόνημα αυτό συμπληρώνεται από ένα παράρτημα, το οποίο περιλαμβάνει τον κώδικα Android που χρησιμοποιήθηκε για την εφαρμογή που θα επιτρέπει τον ασύρματο έλεγχο από την κινητή συσκευή.

Στο τέλος, υπάρχει η απαιτούμενη αναφορά των πηγών, ηλεκτρονικών και εντύπων, στις οποίες βασίστηκε το θεωρητικό μέρος της εργασίας.

ΚΕΦΑΛΑΙΟ 2^{ο}: ΚΙΝΗΤΗΡΕΣ

Στις εφαρμογές μας, θα χρησιμοποιήσουμε το ρομποτικό όχημα OWI-536, οι κινήσεις του οποίου γίνονται με τη χρήση κινητήρων DC μικρής ισχύος. Κρίθηκε, λοι-πόν, σκόπιμη η αναφορά στις ηλεκτρικές μηχανές και ειδικότερα στις αρχές λει-τουργίας των κινητήρων DC, δηλαδή των κινητήρων συνεχούς ρεύματος, όπως αυτοί της συγκεκριμένης εργασίας.

2.1 Τι είναι ο κινητήρας

Είναι ηλεκτρική μηχανή, η οποία μετατρέπει την ηλεκτρική ενέργεια σε μηχανική. Θα μπορούσε, επίσης, να λειτουργήσει και ως ηλεκτρική γεννήτρια, μετατρέπο-ντας τη μηχανική-κινητική ενέργεια σε ηλεκτρική.

Η αρχή λειτουργίας του ηλεκτρικού κινητήρα είναι η δύναμη Laplace. Όταν ένας αγωγός, από τον οποίο διαρρέει ηλεκτρικό ρεύμα, βρεθεί μέσα σε ένα μαγνητικό πεδίο, ασκείται πάνω του δύναμη ίση με: $\mathbf{F = I \cdot L \cdot B \cdot \sin\varphi}$. ,όπου: **I** = Ένταση Ρεύματος - **L** = Μήκος Αγωγού - **B** = Ένταση Μαγνητικού πεδίου και **φ** = η γωνία που σχηματίζει ο αγωγός με τη διεύθυνση των δυναμικών γραμμών.

Για να επιτευχθεί αυτό, θα πρέπει πρώτα να δημιουργηθεί μαγνητικό πεδίο στο εσωτερικό της μηχανής μέσω της τροφοδοσίας του τυλίγματος διέγερσης και στη συνέχεια να τροφοδοτηθεί με ρεύμα το τύλιγμα του δρομέα μέσω των ψηκτρών και του συλλέκτη. Αν υποθέσουμε ότι οι αγωγοί του τυλίγματος είναι κάθετοι στο μαγνητικό πεδίο, τότε το μέτρο της δύναμης που θα ασκείται σε αυτό θα είναι ίσο με: $\mathbf{F = 2 \cdot I \cdot L \cdot B \cdot N}$,όπου **N** ο αριθμός σπειρών του τυλίγματος του δρομέα.

Η ροπή στρέψης που θα αναπτυχθεί στον δρομέα θα είναι ίση με:

$\boldsymbol{\tau = 2 \cdot N \cdot I \cdot B \cdot L \cdot r}$, όπου **r** η ακτίνα του δρομέα.

Από τη στιγμή που ο δρομέας του κινητήρα αρχίσει να περιστρέφεται, αναπτύσ-σεται στα τυλίγματά του μια αντιηλεκτρεγερτική δύναμη, η οποία αντιτίθεται στην τάση της πηγής τροφοδοσίας του δρομέα. Συνεπώς, και στην περίπτωση του κινητήρα, συνυπάρχουν τα φαινόμενα ανάπτυξης δυνάμεων Laplace (εμφανίζο-νται πρώτα) και επαγωγής τάσης στους αγωγούς (η τάση αναπτύσσεται αφού ο δρομέας αρχίσει να περιστρέφεται).

2.2 Είδη ηλεκτρικών κινητήρων

Οι ηλεκτροκινητήρες μπορούν να τροφοδοτούνται από πηγές συνεχούς ρεύμα-

τος (DC), όπως μπαταρίες (π.χ. στα αυτοκίνητα) ή ανορθωτές, ή από πηγές εναλλασσόμενου ρεύματος (AC), όπως ηλεκτρικό δίκτυο, μετατροπείς ή γεννήτριες.

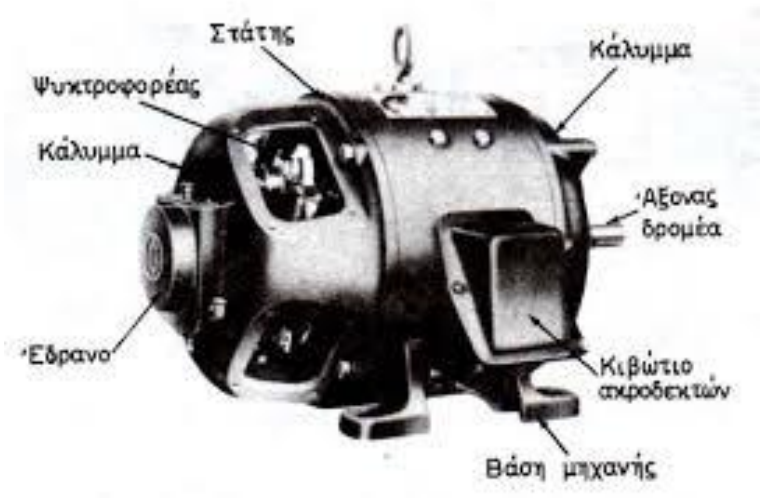
2.2.1 Κινητήρες συνεχούς ρεύματος (DC)

Ένας κινητήρας συνεχούς ρεύματος είναι ένας από τους τύπους περιστροφικών ηλεκτρικών μηχανών που μετατρέπει την ηλεκτρική ενέργεια συνεχούς ρεύματος σε μηχανική ενέργεια. Οι συνηθέστεροι τύποι βασίζονται στις δυνάμεις που παράγονται από τα μαγνητικά πεδία. Σχεδόν όλοι οι τύποι κινητήρων συνεχούς ρεύματος έχουν κάποιο εσωτερικό μηχανισμό, είτε ηλεκτρομηχανικό είτε ηλεκτρονικό, για να αλλάζουν περιοδικά την κατεύθυνση της ροής ρεύματος σε μέρος του κινητήρα.

Οι κινητήρες συνεχούς ρεύματος ήταν οι πρώτοι τύποι που χρησιμοποιούνταν ευρέως, δεδομένου ότι μπορούσαν να τροφοδοτηθούν από τα υπάρχοντα συστήματα διανομής ηλεκτρικού ρεύματος. Οι κινητήρες αυτής της κατηγορίας τροφοδοτούνται από κάποια πηγή συνεχούς τάσης, ενώ βασικό πλεονέκτημά τους αποτελεί η ευκολία ελέγχου της ροπής και της ταχύτητάς τους σε ένα μεγάλο εύρος τιμών.

Ο κινητήρας συνεχούς ρεύματος (DC κινητήρας) περιστρέφεται όταν μία συνεχής τάση εφαρμόζεται στα άκρα του. Ρυθμίζουμε την ταχύτητα περιστροφής του, μεταβάλλοντας την τάση στον κινητήρα. Αλλάζουμε τη φορά περιστροφής του κινητήρα, αλλάζοντας την πολικότητα της τάσης στα άκρα του.

Υπάρχουν DC κινητήρες με ψήκτρες (Brushed DC motors) και χωρίς ψήκτρες (brushless DC motors). Σε αντίθεση με τους κινητήρες χωρίς ψήκτρες, όπως είναι οι βηματικοί κινητήρες, οι DC κινητήρες με ψήκτρες είναι φθηνότεροι και η ταχύτητα περιστροφής τους ρυθμίζεται πιο εύκολα.Όμως, φθείρονται και πιο γρήγορα, γιατί φθείρονται οι ψήκτρες. Οι DC κινητήρες με ψήκτρες λειτουργούν στη βάση της αρχής της επαγωγής. Όταν μέσα από το πηνίο περνάει ρεύμα, τότε δημιουργείται ένα μαγνητικό πεδίο. Το μαγνητικό πεδίο έλκεται ή απωθείται από τους πόλους του ηλεκτρομαγνήτη, ανάλογα με την πολικότητά του. Χρησιμοποιώντας τις ψήκτρες, μπορούμε να αλλάζουμε τη πολικότητα του μαγνητικού πεδίου κάθε μισή στροφή, δημιουργώντας έτσι μια στροφορμή που περιστρέφει τον δρομέα .



Εικόνα 2.1 Μηχανή DC

2.2.2 Κινητήρες εναλλασσόμενου ρεύματος (AC)

Ο κινητήρας εναλλασσόμενου ρεύματος αποτελείται συνήθως από δύο βασικά μέρη, έναν εξωτερικό στάτορα, ο οποίος διαθέτει πηνία που τροφοδοτούνται με εναλλασσόμενο ρεύμα για την παραγωγή ενός περιστρεφόμενου μαγνητικού πεδίου, και έναν εσωτερικό ρότορα προσαρτημένο στον άξονα εξόδου δημιουργώντας ένα δεύτερο περιστρεφόμενο μαγνητικό πεδίο.

2.3 Ιστορική αναδρομή

Οι πρώτοι ηλεκτροκινητήρες ήταν απλοί ηλεκτροστατικοί μηχανισμοί, που περιγράφηκαν σε πειράματα από τον σκωτσέζο μοναχό Andrew Gordon και τον αμερικανό πειραματιστή Benjamin Franklin στη δεκαετία του 1740. Στη συνέχεια, το 1821, ο άγγλος επιστήμονας Michael Faraday απέδειξε τη μετατροπή της ηλεκτρικής ενέργειας σε μηχανική με ηλεκτρομαγνητικά μέσα.

Το 1827, ο ούγγρος φυσικός Ányos Jedlik άρχισε να πειραματίζεται με ηλεκτρομαγνητικά πηνία. Αφού ο Jedlik έλυσε τα τεχνικά προβλήματα συνεχούς περιστροφής με την εφεύρεση του μεταγωγέα, κάλεσε τις πρώτες του συσκευές "ηλεκτρομαγνητικούς αυτορρυθμιστές". Παρόλο που χρησιμοποιήθηκαν μόνο για διδασκαλία, ο Jedlik απέδειξε το 1828 ότι η πρώτη συσκευή περιείχε τα τρία βασικά συστατικά των πρακτικών κινητήρων DC: τον στάτορα (stator), τον ρότορα (rotor) και τον μεταλλάκτη (commutator). Η συσκευή δεν χρησιμοποιούσε μόνιμους μαγνήτες, καθώς τα μαγνητικά πεδία τόσο των σταθερών όσο και των περιστρεφό-

μενων εξαρτημάτων παράγονται αποκλειστικά από τα ρεύματα που ρέουν μέσω των περιελίξεών τους.

2.4 Κατασκευαστικά στοιχεία μηχανών DC

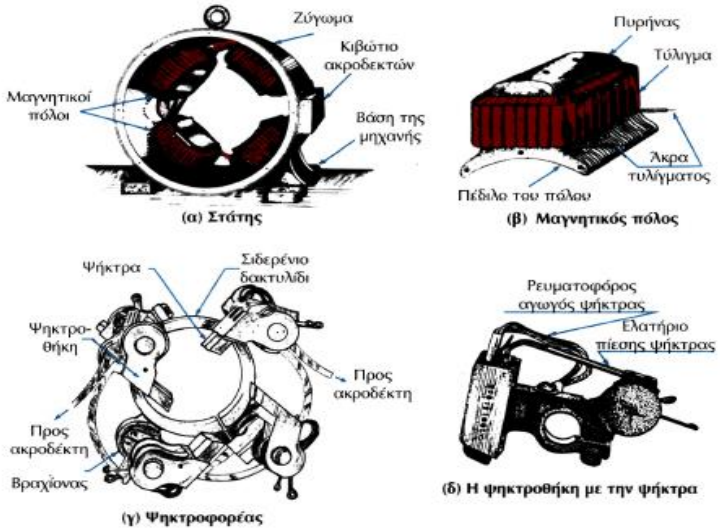
Κάθε μηχανή συνεχούς ρεύματος αποτελείται από το ακίνητο μέρος, το οποίο ονομάζεται στάτης, και από το κινητό μέρος,το οποίο ονομάζεται δρομέας.

Εικόνα 2.2 Ρότορας (αριστερά)και στάτορας (δεξιά)

2.4.1 Στάτης (Stator)

Είναι το σύνολο των ακίνητων τμημάτων της μηχανής, το οποίο έχει ως κύριο προορισμό του τη δημιουργία της προκαθορισμένης μαγνητικής ροής. Αποτελείται από τα πιο κάτω επιμέρους μέρη:

1.Το ζύγωμα
2.Μαγνητικούς πόλους
3.Τα πέδιλα των πόλων
4. Το τύλιγμα του πόλου (πηνίο διέγερσης)
5 Τα καλύμματα (καπάκια)
6.Τον ψηκτροφορέα
7.Τις ψήκτρες
8.Το σιδερένιο δακτυλίδι
9.Τους βραχίονες
10.Τις ψηκτροθήκες
11.Τα ελατήρια πίεσης των ψηκτρών



Εικόνα 2.3 Ακίνητα τμήματα μηχανής DC

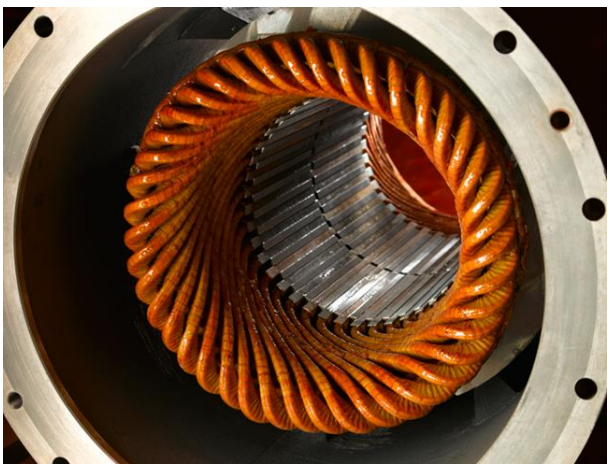
2.4.2 Δρομέας - Ρότορας (Rotor)

Είναι το συγκρότημα των κινητών τμημάτων της μηχανής και αποτελείται από:

1.Τον άξονα

2.Τον πυρήνα του επαγωγικού τυμπάνου

3.Τα τυλίγματα του επαγωγικού τυμπάνου ή περιελίξεις (Windings): Είναι ειδικά σύρματα, που χρησιμοποιούνται για την κατασκευή πηνίων, τα οποία είτε θα δημιουργούν το κατάλληλο μαγνητικό πεδίο, είτε θα περιστρέφονται μέσα σε αυτό.



Εικόνα 2.4. Περιελίξεις ηλεκτρικού κινητήρα

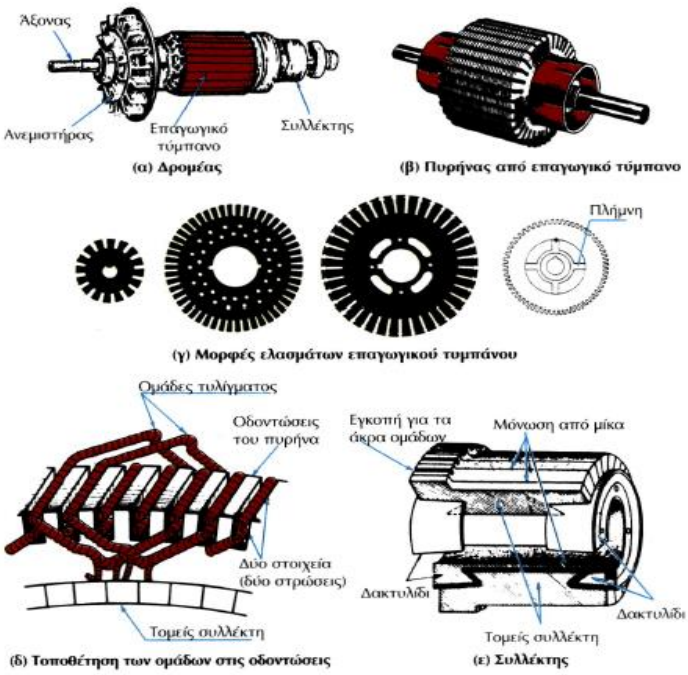
4.Τον συλλέκτη ή μεταλλάκτη (commutator):είναι ένας μηχανισμός που χρησιμοποιείται για την αλλαγή της εισόδου των περισσότερων DC μηχανών και ορισμένων μηχανών AC. Αποτελείται από τμήματα ολίσθησης που είναι μονωμένα με-

ταξύ τους και από τον άξονα. Το ρεύμα του οπλισμού του κινητήρα τροφοδοτείται από σταθερές ψήκτρες, που έρχονται σε επαφή με τον περιστρεφόμενο κινητήριο διακόπτη,πράγμα που προκαλεί την απαιτούμενη αναστροφή ρεύματος και εφαρμόζει τη βέλτιστη ενέργεια στη μηχανή καθώς ο ρότορας περιστρέφεται .

5.Τον ανεμιστήρα,ο οποίος χρησμοποιείται, για να διώχνει τη θερμότητα που παράγεται κατά τη λειτουργία.

6.Τα ρουλεμάν (Bearings): Ο ρότορας υποστηρίζεται από ρουλεμάν, τα οποία του επιτρέπουν να γυρίζει στον άξονά του. Αυτά υποστηρίζονται με τη σειρά τους από το περίβλημα του κινητήρα. Ο άξονας του κινητήρα εκτείνεται μέσω των ρουλεμάν στο εξωτερικό μέρος του κινητήρα, όπου εφαρμόζεται το φορτίο.

Εικόνα 2.5 διάφοροι τύποι ρουλεμάν



Εικόνα 2.6 Κινητά τμήματα της μηχανής DC

2.5 Είδη κινητήρων DC ως προς τον τρόπο διέγερσης

Ανάλογα με τον τρόπο με τον οποίο είναι συνδεδεμένο το τύλιγμα διέγερσης των κινητήρων DC, αυτοί διακρίνονται σε : α) Κινητήρες με ξένη διέγερση β) Κινητήρες με παράλληλη διέγερση γ) Κινητήρες με διέγερση σειράς δ) Κινητήρες με σύνθετη διέγερση, η οποία μπορεί να είναι είτε αθροιστική είτε διαφορική.

2.6 Χαρακτηριστικά κινητήρων

2.6.1 Ροπή των κινητήρων DC

Η ροπή Τ, η οποία ασκείται σε ένα πραγματικό κινητήρα DC δίνεται από τη σχέση $\mathbf{T = \frac{P \cdot S \cdot W}{2\pi \cdot a} \cdot \Phi \cdot I_T}$ **(σε N.m)**,όπου: **P** ο αριθμός των ζευγών των μαγνητικών πόλων της μηχανής – **S** ο αριθμός των στοιχείων του τυλίγματος – **W** ο αριθμός των αγωγών κάθε στοιχείου - **a** ο αριθμός των ζευγών των παράλληλων κλάδων- **Φ** η μαγνητική ροή κάθε μαγνητικού πόλου (V.s) και **Iτ** η ένταση του ρεύματος του τυμπάνου (σε Α).

2.6.2 Ταχύτητα περιστροφής των κινητήρων DC

Η ταχύτητα περιστροφής των κινητήρων DC μπορεί να ρυθμιστεί με μεγάλη ακρίβεια με απλό και εύκολο τρόπο.Η σχέση που δίνει τη ταχύτητα της περιστροφής του κινητήρα είναι η εξής**:** $\mathbf{n = \frac{V - I_T \cdot R_T}{\kappa \cdot \Phi}}$ (σε στρ/min), όπου **V** η τάση της πηγής που τροφοδοτεί το κινητήρα σε (V) - **Iτ** η ένταση του ρεύματος μέσα από το τύλιγμα του επαγωγικού τυμπάνου (σε Α) - **Rτ** η αντίσταση του τυλίγματος του επαγωγικού τυμπάνου και των ψηκτρών(σε Ω) – **Φ** η μαγνητική ροή κάθε μαγνητικού πόλου (σε V.s) και **κ** η P.S.W/2.π.α : σταθερή ποσότητα.

Από τη σχέση που δίνει τη ταχύτητα περιστροφής, παρατηρούμε ότι οι δύο παράγοντες που προσδιορίζουν την ταχύτητα είναι η τάση V και η μαγνητική ροή Φ των πόλων, δηλαδή η ένταση διέγερσης. Έτσι, ανάλογα με αυτούς τους παράγοντες, οι μέθοδοι ρύθμισης της ταχύτητας περιστροφής είναι:

α) Με ρυθμιστική αντίσταση στο παράλληλο τύλιγμα διέγερσης.

β) Με ρυθμιστική αντίσταση στο επαγωγικό τύμπανο.

γ) Με μεταβολή της τάσης τροφοδοσίας του επαγωγικού τυμπάνου.

2.6.3 Ισχύς

Ο κινητήρας απορροφά ισχύ από την πηγή τροφοδοσίας του, η οποία δίνεται από τη σχέση : $\mathbf{P_{εισ} = \frac{V \cdot I}{1000}}$ (σε KW)

Η ισχύς που αποδίδει ο κινητήρας στον άξονά του δίνεται από τη σχέση:

$$\mathbf{P = \frac{T_α \cdot n}{9554}} \text{ (σε KW) ή } \mathbf{P = \frac{T_α \cdot n}{7025}} \text{ (σε HP)}$$

όπου $\mathbf{T_α}$ η ροπή που αναπτύσσει ο κινητήρας στον άξονά του και **n** η ταχύτητα περιστροφής του κινητήρα σε στρ/min.

2.6.4 Απώλειες

Η ισχύς P που αποδίδει ο κινητήρας στον άξονά του υπό μορφή μηχανικής ενέργειας είναι πάντοτε μικρότερη από την ισχύ P εισαγ. που απορροφά από το δίκτυο.

Η ισχύς P που παίρνουμε από τον κινητήρα είναι πάντοτε μικρότερη της ισχύος τροφοδοσίας P εισαγ. Αν P απ. παριστάνει την ισχύ των απωλειών,ισχύει: $\mathbf{P = P_{εισ} - P_{απ}}$

2.6.5 Βαθμός απόδοσης n

Ο λόγος της ισχύος P,την οποία μας δίνει ο κινητήρας στον άξονά του ,προς την ισχύ Pεισ., την οποία παίρνει από το δίκτυο,ονομάζεται βαθμός απόδοσης και είναι πάντοτε μικρότερος της μονάδας.

$$\mathbf{n = \frac{P}{P_{εισ}} = \frac{P}{P + P_{απ}} < 1}$$

Ο βαθμός απόδοσης n στις ηλεκτρικές μηχανές δεν είναι σταθερός, αλλά εξαρτάται από το φορτίο και μειώνεται απότομα,όταν το φορτίο του κινητήρα μειωθεί σημαντικά. Ο βαθμός απόδοσης στους κινητήρες γίνεται μέγιστος σε εκείνη την τιμή φορτίου κατά την οποία οι μεταβλητές απώλειες γίνονται ίσες με τις σταθερές. Αυτό συμβαίνει όταν ο κινητήρας αποδίδει ισχύ ίση με την ονομαστική τους ή και ελαφρώς μικρότερη αυτής.

2.7 Ειδικοί τύποι κινητήρων

2.7.1.Βηματικοί κινητήρες (stepper motors)

Είναι κινητήρες DC που κινούνται σε διακριτά βήματα. Έχουν πολλά πηνία που

οργανώνονται σε ομάδες και ονομάζονται "φάσεις". Με την ενεργοποίηση κάθε φάσης σε σειρά, ο κινητήρας θα περιστραφεί ένα βήμα κάθε φορά. Με τη βοήθεια ενός υπολογιστή, μπορούμε να επιτύχουμε ακριβή ρύθμιση θέσης και ταχύτητας. Για τον λόγο αυτό, οι βηματικοί κινητήρες είναι οι κινητήρες που επιλέγονται για πολλές εφαρμογές ελέγχου ακριβείας.

Εικόνα 2.7 Βηματικός κινητήρας

2.7.2 Σερβοκινητήρες (servomotors)

Είναι μικρές συσκευές που έχουν έναν άξονα, ο οποίος μπορεί να περιστρέφεται σε συγκεκριμένη γωνία (0 έως 180 μοίρες) , ελεγχόμενη από τον χρήστη. Ο έλεγχος ενός σερβοκινητήρα γίνεται με τη βοήθεια παλμών που επαναλαμβάνονται κάθε 20 msec (ή 1/0,02= 50 Hz). Το μήκος του παλμού ρυθμίζει και τη γωνία του άξονα. Τέτοιες συσκευές υπάρχουν σε τηλεκατευθυνόμενα αεροπλανάκια, αυτοκίνητα, σε robots και σε πάρα πολλές άλλες εφαρμογές.

2.8 Εφαρμογές

Οι κινητήρες γενικής χρήσης με τυπικές διαστάσεις και χαρακτηριστικά παρέχουν βολική μηχανική ισχύ για βιομηχανική χρήση. Οι μεγαλύτεροι ηλεκτροκινητήρες χρησιμοποιούνται στην κίνηση οχημάτων, όπως ηλεκτρικών σιδηροδρόμων, τρόλεϊ, τραμ, στην εκκίνηση κινητήρων αυτοκινήτων (μίζα) κλπ.

Οι ηλεκτροκινητήρες βρίσκονται, επίσης, σε βιομηχανικούς ανεμιστήρες, φυσητήρες και αντλίες, εργαλειομηχανές, διάφορες ηλεκτρικές συσκευές οικιακής ή επαγγελματικής χρήσης, όπως ψυγεία, πλυντήρια, σκούπες, μίξερ, ανεμιστήρες, ραδιομαγνητόφωνα, δράπανα κλπ, ως εφεδρικές πηγές ηλεκτρικής ενέργειας σε

νοσοκομεία, εργοστάσια, αεροδρόμια κλπ. Μικροί κινητήρες μπορεί να βρεθούν και σε ηλεκτρικά ρολόγια.

Εικόνα 2.8 Σερβοκινητήρας

ΚΕΦΑΛΑΙΟ 3^{ο} : ARDUINO

3.1. Μικροεπεξεργαστές - Μικροελεγκτές

3.1.1 Μικροεπεξεργαστές (Microprocessors)

Είναι ολοκληρωμένα κυκλώματα (IC) κατηγορίας LSI/VLSI (Large/ Very Large Scale Integrated), τα οποία περιέχουν μεγάλο αριθμό ψηφιακών κυκλωμάτων ομαδοποιημένα σε υπομονάδες. Εκτελούν βασικές αριθμητικές και λογικές λειτουργίες καθώς και λειτουργίες ελέγχου και μεταφορά δεδομένων από και προς τη μνήμη και τις περιφερειακές συσκευές. Καθοδηγούνται από σειρές εντολών γλώσσας μηχανής (πρόγραμμα), οι οποίες συντάσσονται με βάση ένα πεπερασμένο σετ εντολών, ειδικό για κάθε μικροεπεξεργαστή.

Οι μικροεπεξεργαστές χαρακτηρίζονται από το εύρος του διαύλου δεδομένων (μήκος λέξης), που ταυτίζεται με το μέγεθος των βασικών καταχωρητών και είναι 4, 8, 16, 32, 64, 128 bits κτλ,ενώ η μέγιστη συχνότητα λειτουργίας τους (100KHz-10GHz) καθορίζεται από ένα κύκλωμα χρονισμού (clock). Στη συσκευασία τους, έχουν μεγάλο αριθμό pins (16...478) για σύνδεση με την τροφοδοσία, τους διαύλους, τις γραμμές ελέγχου και διακοπών και τις υπόλοιπες υπομονάδες ενός υπολογιστικού συστήματος.

Οι μικροεπεξεργαστές (Μ/Ε) εκτοπίζουν τα παραδοσιακά ηλεκτρονικά stoixe;ia από σχεδόν κάθε πεδίο που περιλαμβάνει προγραμματισμό ή αυτόματο έλεγχο. Έτσι, εκτός από την κατασκευή Η/Υ, χρησιμοποιούνται σε οικιακές συσκευές, συσκευές γραφείου, ηλεκτρονικά παιχνίδια, στην αυτοκινητοβιομηχανία και αλλού.

Η ιστορία των Μ/Ε ξεκινά με τον 4-bit Μ/Ε 4004 της Intel το 1971. Το chip αποτελείτο από 2.300 τρανζίστορ και εκτελούσε περίπου 60.000 πράξεις/sec. Προορίζονταν για την υλοποίηση υπολογιστών τσέπης (calculator), αλλά αποτέλεσε μεγάλη εμπορική επιτυχία και άνοιξε τον δρόμο για τη δημιουργία των μικροϋπολογιστών. Η Intel εξέλιξε τη σειρά των Μ/Ε της και κυριαρχεί στον χώρο αυτόν μέχρι σήμερα με τη σειρά 80Χ86 και αργότερα τη σειρά των Pentium. Άλλη μια σειρά που γνώρισε εμπορική επιτυχία είναι η σειρά 68Χ00 της Motorola. Διάφορες άλλες σχεδιάσεις εμφανίστηκαν με επιτυχία για κάποιες περιόδους στην αγορά, αλλά δεν συνέχισαν να εξελίσσονται, όπως ο Z-80 της Zilog, ο 6502 της Rockwell κ.α. Για να έχει ο αναγνώστης ένα μέτρο σύγκρισης, θα πρέπει να αναφέρουμε ότι

οι σύγχρονοι Μ/Ε έχουν περισσότερα από 500 εκατομμύρια τρανζίστορ και εκτελούν μερικές δεκάδες εκατομμύρια πράξεις/sec.

3.1.2 Μικροελεγκτές(Microcontrollers)

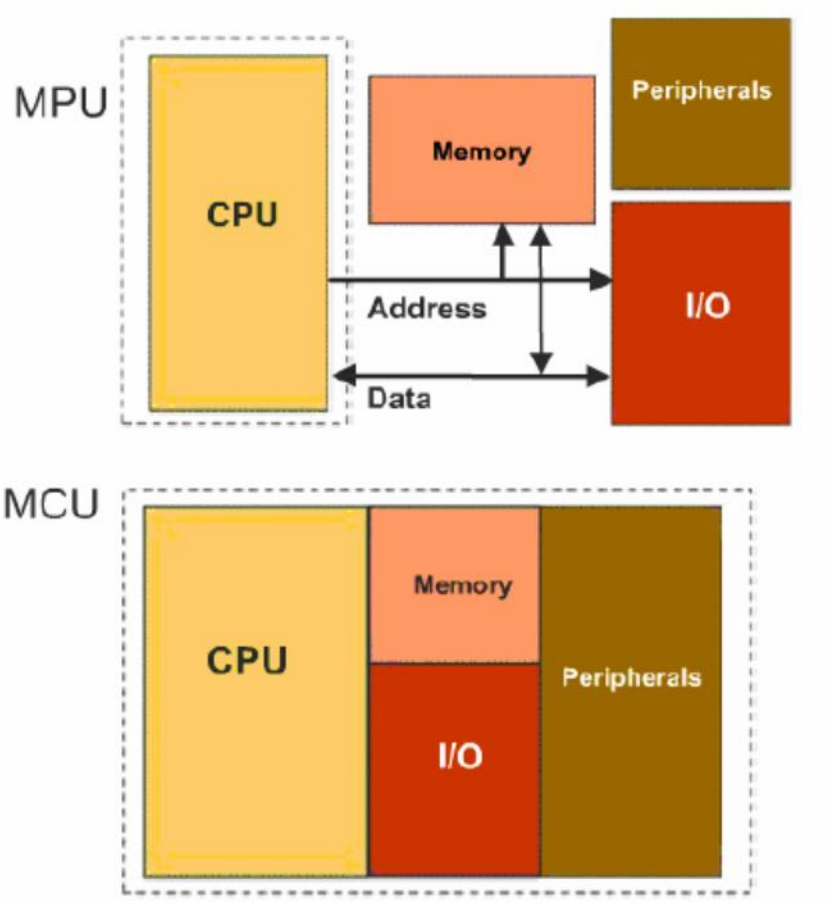
Είναι ένα προγραμματιζόμενο ολοκληρωμένο κύκλωμα (IC), το οποίο διαθέτει επεξεργαστή, μνήμη, διάφορα περιφερειακά κυκλώματα, καθώς επίσης και θύρες εισόδου/εξόδου για επικοινωνία με εξωτερικές συσκευές. Θα μπορούσε να παρομοιαστεί με έναν μικροϋπολογιστή. Όπως ακριβώς ένας μικροϋπολογιστής έχει επεξεργαστή, μνήμη, περιφερειακές συσκευές και εκτελεί προγράμματα, έτσι κι ένας μικροελεγκτής διαθέτει τα παραπάνω χαρακτηριστικά και μάλιστα ολοκληρωμένα σε ένα μόνο chip. Το πρόγραμμα που εκτελεί ο μικροελεγκτής αποθηκεύεται μόνιμα στη μνήμη προγράμματος.

Οι μικροελεγκτές βρίσκουν εφαρμογή σε συστήματα αυτοματισμών, κυκλώματα τηλεπικοινωνιών, στις ηλεκτρονικές και ηλεκτρικές συσκευές, σε συστήματα τηλεματικής και συλλογής δεδομένων (Data Acquisition), σε εφαρμογές ηλεκτρονικών ισχύος, σε συστήματα διασύνδεσης, σε εφαρμογές δικτύων κλπ. Γενικότερα, οι μικροελεγκτές χρησιμοποιούνται οπουδήποτε απαιτείται έλεγχος συστημάτων.

3.1.3 Διαφορές του Μικροελεγκτή από τον Μικροεπεξεργαστή

Ο μικροελεγκτής είναι ένα μικρό αυτόνομο υπολογιστικό σύστημα, προγραμματισμένο να εκτελεί μία συγκεκριμένη λογική ακολουθία εντολών, οι οποίες έχουν καταχωρηθεί στην προγραμματιζόμενη μόνιμη μνήμη του. Κάθε φορά που επανεκκινείται, ο μικροελεγκτής εκτελεί την ίδια λογική. Ανακαλεί τα δεδομένα, τα επεξεργάζεται και με βάση τα αποτελέσματα της επεξεργασίας ελέγχει το περιβάλλον του. Πρόκειται, δηλαδή, για σύστημα ειδικού σκοπού, αφιερωμένο (dedicated) στον έλεγχο και την εξυπηρέτηση ενός συγκεκριμένου αυτοματισμού.

Ο μικροεπεξεργαστής, μετά την εκκίνησή του, δεν είναι από μόνος του σε θέση να εκτελέσει κάποια λογική ακολουθία. Αν και μπορεί να συνδεθεί με μνήμες RAM και ROM, αυτές αποτελούν ξεχωριστές μονάδες, που συνήθως δεν ολοκληρώνονται μέσα στον ίδιο τον μικροεπεξεργαστή.



Εικόνα 3.1 Διαφορές Μικροεπεξεργαστή- Μικροελεγκτή

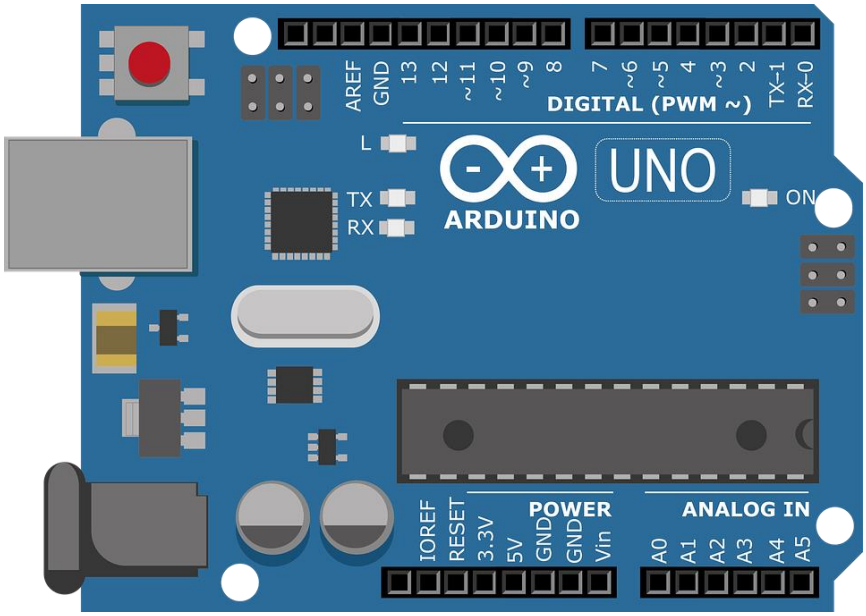
3.2 Η υπολογιστική πλατφόρμα Arduino

Το Arduino είναι μια υπολογιστική πλατφόρμα βασισμένη σε μια απλή μητρική πλακέτα ανοικτού κώδικα, με ενσωματωμένο ή αποσπώμενο μικροελεγκτή και φυσικά διαθέσιμες εισόδους/εξόδους, και η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++).

Το Arduino έχει χρησιμοποιηθεί σε χιλιάδες έργα και εφαρμογές, λόγω της απλότητας στη χρήση του και της χαμηλής τιμής του. Οι κάρτες Arduino είναι σχετικά φθηνές σε σύγκριση με άλλες πλατφόρμες μικροελεγκτών.Ένα άλλο του πλεονέκτημα είναι ότι το λογισμικό Arduino (IDE) λειτουργεί σε λειτουργικά συστήματα Windows, Macintosh OSX και Linux,τη στιγμή που τα περισσότερα συστήματα μικροελεγκτών περιορίζονται στα Windows.

Το παραπάνω λογισμικό είναι εύκολο στη χρήση για αρχάριους, αλλά αρκετά ευέλικτο, για να επωφεληθούν και οι προηγμένοι χρήστες, ιδανικό για εκπαιδευτικές εφαρμογές. Λογισμικό ανοιχτού κώδικα και επεκτάσιμο λογισμικό , το λογισμι-

κό Arduino δημοσιεύεται ως εργαλείο ανοικτού κώδικα, διαθέσιμο για επέκταση από έμπειρους προγραμματιστές.



Εικόνα 3.2 Η υπολογιστική πλατφόρμα Arduino

3.2.1 Ιστορική αναδρομή

Το 2005, δημιουργήθηκε ένα σχέδιο προκειμένου να φτιαχτεί μια συσκευή για τον έλεγχο προγραμμάτων διαδραστικών σχεδίων από μαθητές, η οποία θα ήταν πιο φθηνή από τα άλλα πρωτότυπα συστήματα που ήταν διαθέσιμα εκείνη την περίοδο. Οι ιδρυτές Massimo Banzi και David Cueartielles ξεκίνησαν να παράγουν πλακέτες σε ένα μικρό εργοστάσιο στην Ιβρέα, η οποία είναι κωμόπολη της επαρχίας Τορίνο, στην περιοχή Πεδεμόντιο της βορειοδυτικής Ιταλίας, στην ίδια περιοχή στην οποία στεγαζόταν η εταιρία υπολογιστών Olivetti. Το σχέδιο ονομάστηκε από τους ιδρυτές του Arduino προς τιμή του βασιλιά της Ιταλίας Arduin.

Από τότε μέχρι σήμερα,έχουν ανακοινωθεί οι εξής εκδόσεις:

- Τον Σεπτέμβριο του 2006, ανακοινώθηκε το Arduino Mini .
- Τον Οκτώβριο του 2008, ανακοινώθηκε το Arduino Duemilanove. Αρχικά , βασίστηκε στο Atmel Atmega168, αλλά μετά βασίστηκε στο ATmega328.
- Τον Μάρτιο του 2009, ανακοινώθηκε το Arduino Mega, βασισμένο στο Atmel ATmega1280.
- Στις 24 Σεπτεμβρίου 2010, ανακοινώθηκε το Arduino Uno, βασισμένο στο Atmega328P.

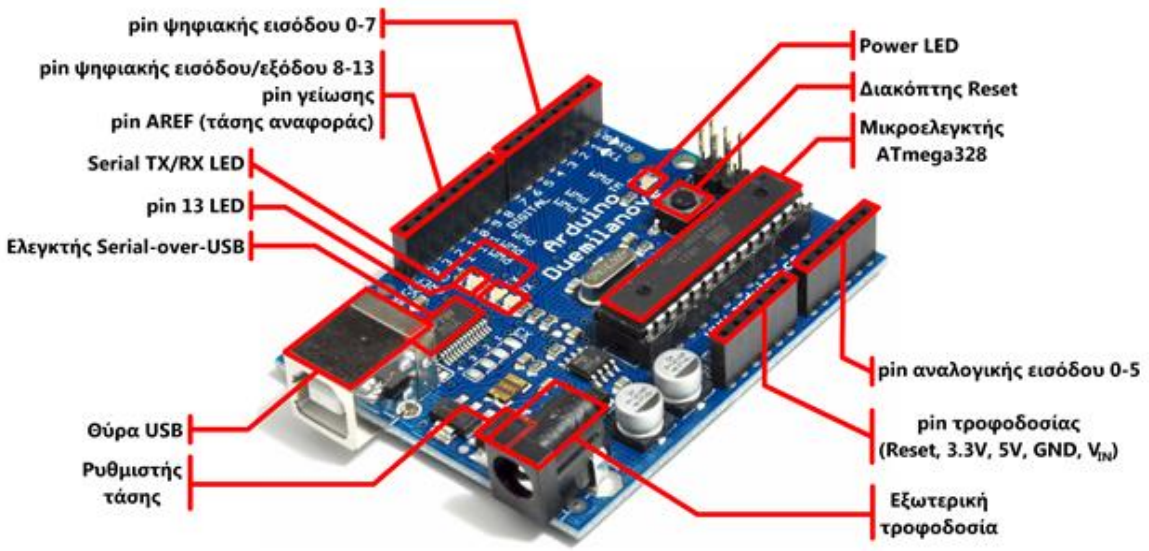
- Από τον Μάιο του 2011, πάνω από 300.000 Arduino ήταν σε χρήση σε όλο τον κόσμο.
- Τον Ιούλιο του 2012, ανακοινώθηκε το Arduino Leonardo, βασισμένο στο Atmel ATmega32u4.
- Τον Οκτώβριο του 2012, ανακοινώθηκε το Arduino Due, βασισμένο στο Atmel SAM3X8E, που είχε πυρήνα ARM Cortex-M3.
- Τον Νοέμβριο του 2012, ανακοινώθηκε το Arduino Micro, βασισμένο στο Atmel ATmega32u4.
- Τον Μάιο του 2013, ανακοινώθηκε το Arduino Robot, βασισμένο στο Atmel ATmega32u4 και ήταν το πρώτο επίσημο Arduino με ρόδες.
- Τον Μάιο του 2013, ανακοινώθηκε το Arduino Yun, βασισμένο στο Atmega 32u4 και στο Atheros AR9331 και ήταν το πρώτο προϊόν wifi που συνδύαζε το Arduino με το Linux.
- Στις 16 Οκτωβρίου 2015, ανακοινώθηκε το Arduino 101, βασισμένο στο Intel® Curie.

3.2.2 Arduino Uno

Το Arduino Uno είναι μία πλακέτα μικροελεγκτών που βασίζεται στο ATmega 328P. Διαθέτει 14 ψηφιακούς ακροδέκτες εισόδου / εξόδου (από τους οποίους 6 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM), 6 αναλογικές εισόδους κρυστάλλων quartz 16 MHz, σύνδεση USB, υποδοχή τροφοδοσίας, κεφαλίδα ICSP και κουμπί επαναφοράς.

Περιέχει,δηλαδή, όλα τα απαραίτητα για την υποστήριξη του μικροελεγκτή,αρκεί να συνδεθεί με έναν υπολογιστή με καλώδιο USB ή να τροφοδοτηθεί με έναν προ-σαρμογέα AC ή DC ή μπαταρία για να ξεκινήσει.

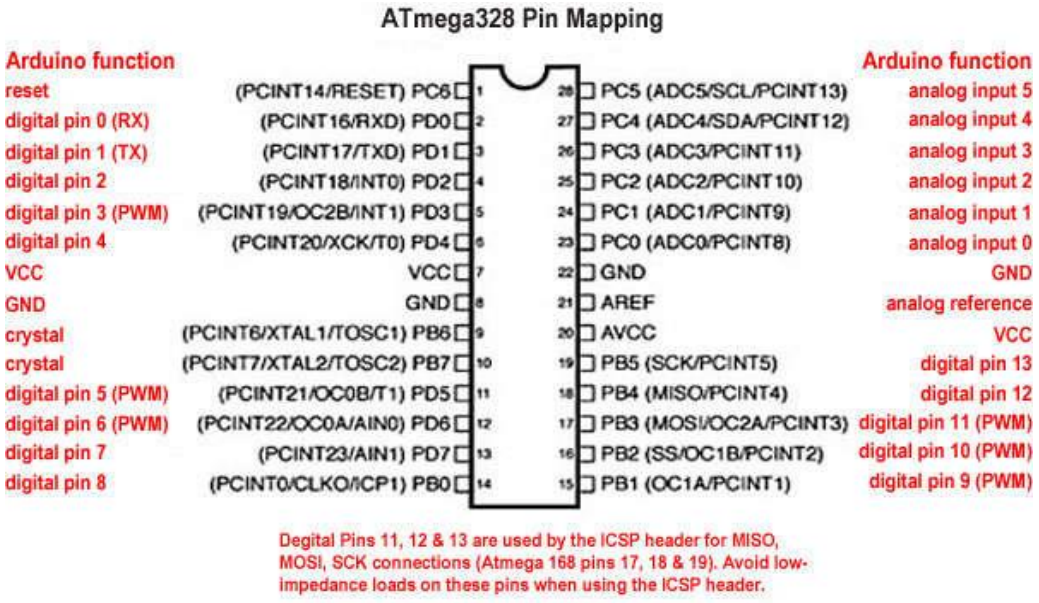
Το "Uno" σημαίνει ένα στην ιταλική γλώσσα και επιλέχθηκε για να σηματοδο-τήσει την κυκλοφορία του Arduino Software (IDE) 1.0. Η πλατφόρμα Uno και η έκδοση 1.0 του λογισμικού Arduino (IDE) ήταν οι εκδόσεις αναφοράς του Arduino, που τώρα εξελίχθηκαν σε νεότερες εκδόσεις. Η πλακέτα Uno είναι η πρώτη σε μια σειρά από κάρτες USB Arduino και το μοντέλο αναφοράς για την πλατφόρμα Arduino.



Εικόνα 3.3 Arduino Uno

3.2.2.1 Χαρακτηριστικά

- Μικροελεγκτής: ATmega328
- Τάση λειτουργίας : 5V
- Τάση εισόδου (συνιστάται): 7-9V
- Τάση εισόδου (όρια): 6-20V
- Ψηφιακοί ακροδέκτες: I / O 14 (από τους οποίους 6 παρέχουν έξοδο PWM)
- Αναλογικοί ακροδέκτες εισόδου: 6
- Τάση DC ανά είσοδο εισόδου / εξόδου: 40 mA
- Ρεύμα DC για 3,3V ακροδέκτη: 50 mA
- Μνήμη Flash: 32 KB (ATmega328) (0,5 KB που χρησιμοποιείται από το bootloader)
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Ταχύτητα ρολογιού: 16 MHz



Εικόνα 3.4 Χάρτης pin του μικροελεγκτή ATmega328

3.2.2.2 Τροφοδοσία

Η πλακέτα Arduino Uno μπορεί να τροφοδοτηθεί μέσω της σύνδεσης USB ή με εξωτερικό τροφοδοτικό. Η πηγή ενέργειας επιλέγεται αυτόματα. Η εξωτερική (χωρίς USB) ισχύς μπορεί να προέρχεται είτε από προσαρμογέα AC-to-DC (wall-wart) είτε από μπαταρία. Ο προσαρμογέας μπορεί να συνδεθεί με ένα κεντρικό βύσμα 2,1 mm στην υποδοχή τροφοδοσίας του πίνακα.

Οι ακροδέκτες της μπαταρίας μπορούν να εισαχθούν στις κεφαλές των pins GND και Vin της υποδοχής POWER. Η πλακέτα μπορεί να λειτουργεί σε εξωτερική τροφοδοσία από 6 έως 20V. Εάν τροφοδοτείται με λιγότερα από 7V, ωστόσο, ο ακροδέκτης 5V μπορεί να παρέχει λιγότερα από 5V και η πλακέτα μπορεί να γίνει ασταθής. Εάν χρησιμοποιηθούν περισσότερα από 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να βλάψει την πλακέτα. Το συνιστώμενο εύρος τιμών είναι 7 έως 12V.

ΣΗΜ.Οι ακροδέκτες τροφοδοσίας είναι οι εξής:

- **Vin**: Η τάση εισόδου στην πλακέτα Arduino όταν χρησιμοποιεί εξωτερική πηγή τροφοδοσίας (σε αντίθεση με τα 5V από τη σύνδεση USB ή άλλη ρυθμισμένη πηγή τροφοδοσίας). Μπορούμε να τροφοδοτήσουμε τάση μέσω αυτού του pin ή, εάν τροφοδοτήσουμε τάση μέσω της υποδοχής τροφοδοσίας, να έχουμε πρόσβαση σε αυτήν μέσω αυτού του pin.

- **5V:** Αυτός ο ακροδέκτης εξάγει ρυθμισμένο 5V από τον ρυθμιστή στην πλακέτα, η οποία μπορεί να τροφοδοτηθεί είτε από την υποδοχή συνεχούς ρεύματος (7 - 12V) από τον συνδετήρα USB (5V) είτε από τον ακροδέκτη VIN (7-12V). Η παροχή τάσης μέσω των ακροδεκτών 5V ή 3,3V παρακάμπτει τον ρυθμιστή και μπορεί να βλάψει την πλακέτα.
- **3V3:** Μια τροφοδοσία 3,3V, που παράγεται από τον ρυθμιστή. Η μέγιστη έλξη ρεύματος είναι 50 mA
- **GND**: Ακροδέκτες γείωσης
- **IOREF**: Αυτός ο ακροδέκτης στην πλακέτα Arduino παρέχει την τάση αναφοράς, με την οποία λειτουργεί ο μικροελεγκτής. Μια κατάλληλα ρυθμισμένη θωράκιση μπορεί να διαβάσει την τάση IOREF pin και να επιλέξει την κατάλληλη πηγή ισχύος ή να ενεργοποιήσει τους μεταφραστές τάσης στις εξόδους, για να λειτουργήσει με τα 5V ή τα 3.3V.

3.2.2.3 Μνήμη

Όπως προαναφέρθηκε, το Arduino βασίζεται στον ATmega328, έναν 8-bit RISC μικροελεγκτή,τον οποίο χρονίζει στα 16MHz. Ο ATmega328 διαθέτει ενσωματωμένη μνήμη τριών τύπων:

- 2Kb μνήμης SRAM, που είναι η ωφέλιμη μνήμη που μπορούν να χρησιμοποιήσουν τα προγράμματα, για να αποθηκεύουν μεταβλητές, πίνακες κ.λπ. κατά το runtime. Όπως και σε έναν υπολογιστή, αυτή η μνήμη χάνει τα δεδομένα της, όταν η παροχή ρεύματος στο Arduino σταματήσει ή αν γίνει reset.
- 1Kb μνήμης EEPROM η οποία μπορεί να χρησιμοποιηθεί για «ωμή» εγγραφή/ανάγνωση δεδομένων (χωρίς datatype) ανά byte από τα προγράμματα κατά το runtime. Σε αντίθεση με την SRAM, η EEPROM δεν χάνει τα περιεχόμενά της με απώλεια τροφοδοσίας ή reset, οπότε είναι το ανάλογο του σκληρού δίσκου.
- 32Kb μνήμης Flash, από τα οποία τα 2Kb χρησιμοποιούνται από το firmware του Arduino που έχει εγκαταστήσει ήδη ο κατασκευαστής του. Το firmware αυτό, που στην ορολογία του Arduino ονομάζεται bootloader, είναι αναγκαίο για την εγκατάσταση προγραμμάτων στον μικροελεγκτή μέσω της θύρας USB, χωρίς δηλαδή να χρειάζεται εξωτερικός hardware

programmer. Τα υπόλοιπα 30Kb της μνήμης Flash χρησιμοποιούνται για την αποθήκευση αυτών ακριβώς των προγραμμάτων, αφού πρώτα μεταγλωττιστούν στον υπολογιστή.

Η μνήμη Flash, όπως και η EEPROM, δεν χάνει τα περιεχόμενά της με απώλεια τροφοδοσίας ή reset. Επίσης, ενώ υπό κανονικές συνθήκες δεν προορίζεται για χρήση runtime μέσα από τα προγράμματα, λόγω της μικρής συνολικής μνήμης που είναι διαθέσιμη σε αυτά (2Kb SRAM + 1Kb EEPROM), έχει σχεδιαστεί μια βιβλιοθήκη που επιτρέπει τη χρήση όσου χώρου περισσεύει (30Kb μείον το μέγεθος του προγράμματος σε μεταγλωττισμένη μορφή).

3.2.2.4 Arduino pins

Κάθε ένας από τους 14 ψηφιακούς ακροδέκτες στο Uno μπορεί να χρησιμοποιηθεί ως είσοδος ή έξοδος με λειτουργίες pinMode (), digitalWrite () και digitalRead () στα 5V. Κάθε ακροδέκτης μπορεί να παρέχει ή να λαμβάνει 20 mA ως συνιστώμενη κατάσταση λειτουργίας και έχει εσωτερική αντίσταση έλξης (αποσυνδεδεμένη από προεπιλογή) 20-50k Ohm.Η τιμή που δεν πρέπει να ξεπεραστεί σε οποιονδήποτε ακροδέκτη εισόδου / εξόδου, για να αποφευχθεί η μόνιμη βλάβη του μικροελεγκτή, είναι το μέγιστο 40mA.

Το Uno διαθέτει, επίσης, 6 αναλογικές εισόδους (A0 έως A5), κάθε μία από τις οποίες παρέχει 10 ψηφία ανάλυσης (δηλαδή 1024 διαφορετικές τιμές). Από προεπιλογή, μετρούνται από το έδαφος σε 5V, αν και είναι δυνατόν να αλλαχθεί το ανώτερο άκρο της εμβέλειάς τους, χρησιμοποιώντας τον AREF ακροδέκτη και τη λειτουργία analogReference ().

Επιπλέον, μερικά pins έχουν εξειδικευμένες λειτουργίες, όπως:

- **Serial**: 0 (RX) και 1 (TX). Χρησιμοποιείται για τη λήψη σειριακών δεδομένων TTL (RX) και εκπομπής (TX). Αυτά συνδέονται με τους αντίστοιχους ακροδέκτες του σειριακού τσιπ ATmega8U2 USB-to-TTL.
- **Εξωτερικές Διακοπές**: 2 και 3. Αυτοί οι ακροδέκτες μπορούν να διαμορφωθούν για να προκαλέσουν διακοπή είτε σε χαμηλή τιμή είτε σε αλλαγή τιμής(θετική ή αρνητική ακμή).
- **PWM**: 3, 5, 6, 9, 10 και 11. Παρέχεται έξοδος PWM 8-bit με τη λειτουργία analogWrite ().

- **SPI**: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Αυτοί οι ακροδέκτες υποστηρίζουν την επικοινωνία SPI, χρησιμοποιώντας τη βιβλιοθήκη SPI.
- **LED:** 13. Υπάρχει μια ενσωματωμένη λυχνία LED που ενεργοποιείται από τον ψηφιακό ακροδέκτη 13. Όταν ο ακροδέκτης είναι HIGH, η λυχνία LED είναι αναμμένη, όταν ο ακροδέκτης είναι LOW, είναι σβηστή.
- **TWI**: Ακροδέκτης A4 ή SDA και ακροδέκτης A5 ή SCL. Υποστηρίζουν την επικοινωνία TWI, χρησιμοποιώντας τη βιβλιοθήκη Wire.
- **AREF**: Τάση αναφοράς για τις αναλογικές εισόδους. Χρησιμοποιείται με analogReference ().
- **RESET**: Όταν πάρει την τιμή LOW, επαναφέρει τον μικροελεγκτή.

3.3 Arduino IDE

Το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino είναι μια εφαρμογή (για Windows, MacOS, Linux), που είναι γραμμένη στη γλώσσα προγραμματισμού Java.Χρησιμοποιείται για τη συγγραφή προγραμμάτων που ονομάζονται σκίτσα (sketches). Τα σκίτσα αποθηκεύονται στον υπολογιστή ανάπτυξης ως αρχεία κειμένου με την επέκταση αρχείου .ino.

Περιλαμβάνει έναν επεξεργαστή κώδικα με δυνατότητες όπως: κοπή και επικόλληση κειμένου, αναζήτηση και αντικατάσταση κειμένου, αυτόματη εσοχή, αντιστοίχιση ισχίων και επισήμανση σύνταξης. Επίσης, περιλαμβάνει τη δυνατότητα εισαγωγής βιβλιοθηκών για την επέκταση του κώδικα και την εύκολη διαχείριση των εξωτερικών εξαρτημάτων που είναι συνδεδεμένα στο Arduino, δυνατότητα μεταγλώττισης (compile) και ανεβάσματος (upload) ενός προγράμματος στο Arduino με μόνο ένα κλικ, serial monitor για την απεικόνιση της σύνδεσης του υπολογιστή με το Arduino μέσω USB καθώς και πολλά παραδείγματα.

Στην περιοχή μηνύματος, εμφανίζονται μηνύματα που έχουν σχέση με την διαχείριση του Arduino, όπως η επιβεβαίωση μεταγλώττισης και φόρτωσης του σκίτσου καθώς και μηνύματα λαθών.

3.3.1 Εγκατάσταση του προγράμματος

1. Επισκεπτόμαστε το http://www.arduino.cc/en/main/software, για να κατεβάσουμε την τελευταία έκδοση του Arduino IDE για το λειτουργικό σύστημα του υπολογιστή μας. Υπάρχουν εκδόσεις για συστήματα Windows, Mac και Linux.Στη

σελίδα λήψης, κάνουμε κλικ στην επιλογή "Windows Installer" για την ευκολότερη εγκατάσταση.

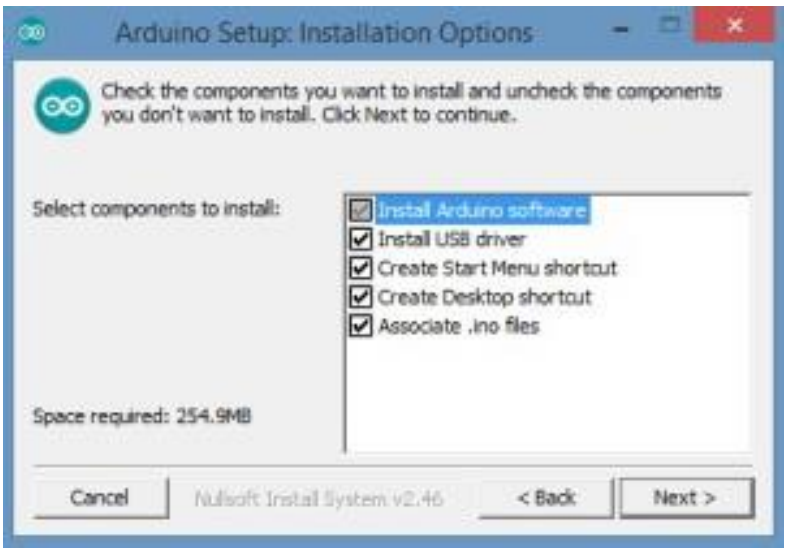
2. Αποθηκεύουμε το αρχείο .exe στον σκληρό δίσκο.

3. Ανοίγουμε το αρχείο .exe.

4. Κάνουμε κλικ στο “I Agree” για να συμφωνήσουμε με την αδειοδότηση.

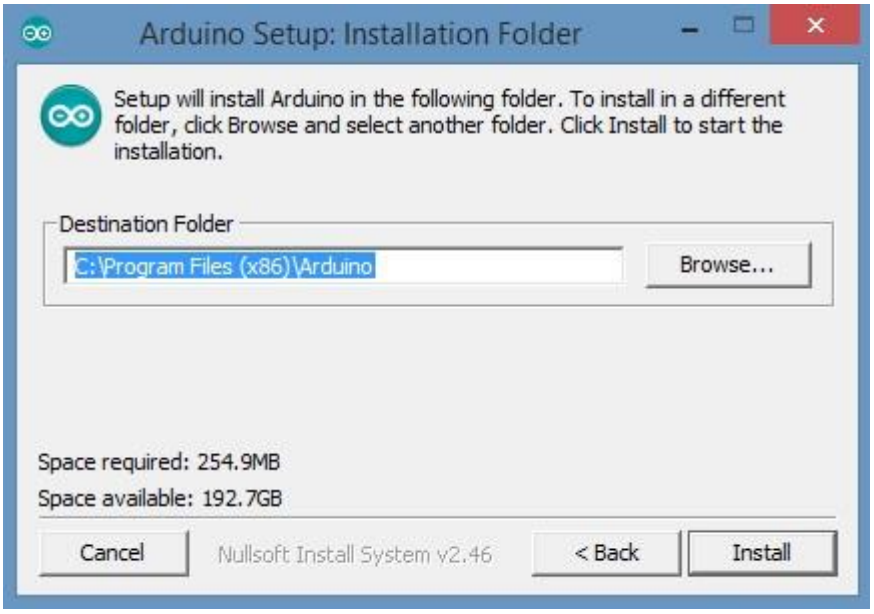
Εικόνα 3.5 Εγκατάσταση Arduino-Βήμα 4

5. Επιλέγουμε ποια στοιχεία θα εγκατασταθούν και, στη συνέχεια, κάνουμε κλικ στο "Next".



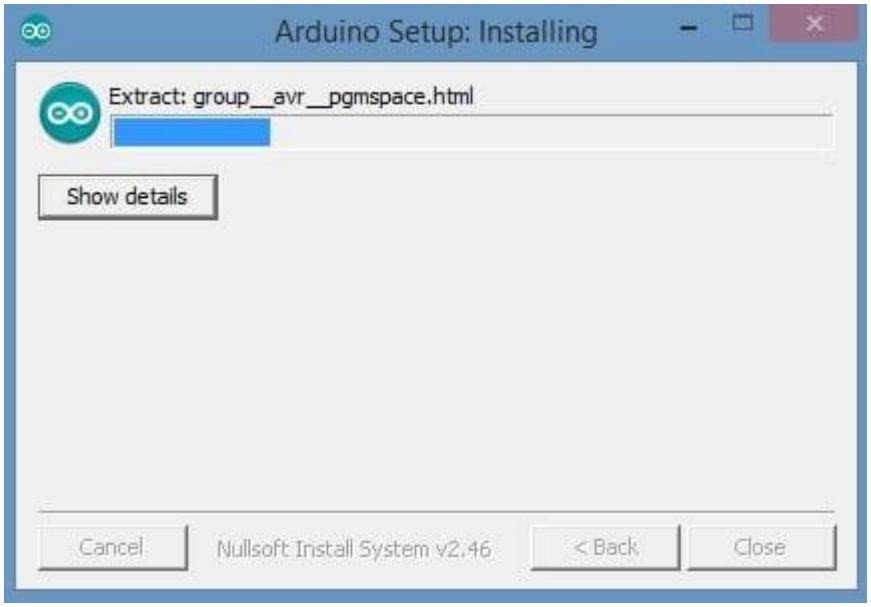
Εικόνα 3.6 Εγκατάσταση Arduino-Βήμα 5

6. Επιλέγουμε τον φάκελο στον οποίο θα εγκατασταθεί το πρόγραμμα και, στη συνέχεια, κάνουμε κλικ στο "Install".



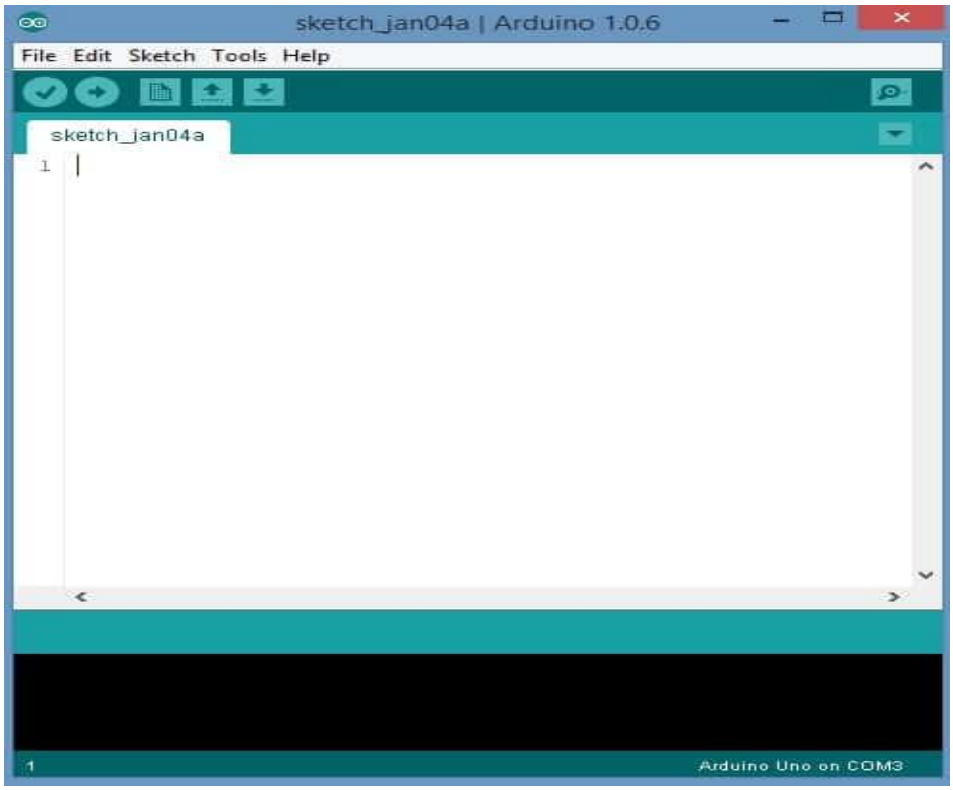
Εικόνα 3.7 Εγκατάσταση Arduino- Βήμα 6

7. Περιμένουμε να ολοκληρωθεί η εγκατάσταση του προγράμματος και κάνουμε κλικ στο κουμπί Close".



Εικόνα 3.8 Εγκατάσταση Arduino- Βήμα 7

8. Βρίσκουμε τη συντόμευση Arduino στην επιφάνεια εργασίας και κάνουμε κλικ σε αυτήν. Το IDE θα ανοίξει και θα εμφανιστεί ο επεξεργαστής κώδικα.



Εικόνα 3.9 Διεπαφή Arduino IDE

Στη συνέχεια, πρέπει να βεβαιωθούμε ότι το λογισμικό έχει ρυθμιστεί για τη συγκεκριμένη πλακέτα Arduino. Πηγαίνουμε στο μενού "Tools" και επιλέγουμε "Board". Θα εμφανιστεί ένα άλλο μενού, από το οποίο μπορούμε να επιλέξουμε από μία λίστα μοντέλων Arduino. Εδώ, επειδή χρησιμοποιούμε το Arduino Uno R3, επιλέγουμε "Arduino Uno".Τέλος, ελέγχουμε ότι η σωστή σειριακή θύρα είναι επιλεγμένη πηγαίνοντας στο “Tools” και μετά στο “Serial Port”.Το Arduino είναι έτοιμο για λειτουργία.

3.3.2 Γραμμή εργαλείων

Τα κουμπιά που υπάρχουν στη γραμμή εργαλειών του Arduino IDE κάνουν τις παρακάτω λειτουργίες:

- **Επικύρωση(Verify)**: Μεταγλωττίζει(Compile) το πρόγραμμα και ελέγχει για λάθη.
- **Ανέβασμα(Upload)**:Φορτώνει το πρόγραμμα στην πλακέτα.

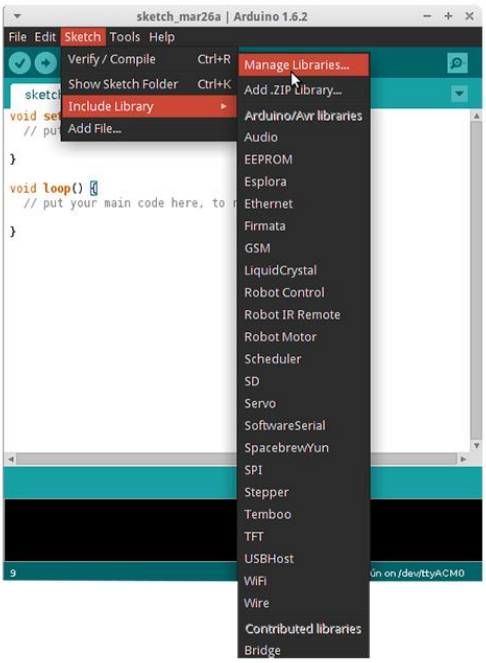
- **Δημιουργία(New)**:Δημιουργεί ένα καινούργιο σκίτσο.
- **Άνοιγμα(Open)**:Επιλέγουμε ποιο σκίτσο θέλουμε να ανοίξουμε.
- **Αποθήκευση(Save)**:Αποθηκεύει το σκίτσο μας.
- **Παρακολούθηση σειριακής(Serial Monitor)**:Ανοίγει τη σειριακή οθόνη.

Πρόσθετες λειτουργίες θα βρούμε στα μενού:Αρχείο(File), Επεξεργασία(Edit), Σχέδιο(Sketch), Εργαλία(Tools), Βοήθεια(Help).

3.3.3 Βιβλιοθήκες

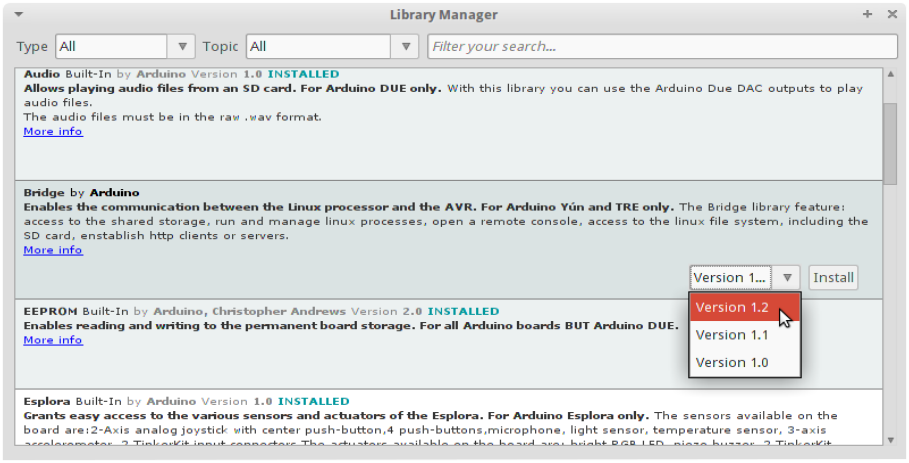
Το περιβάλλον Arduino μπορεί να επεκταθεί μέσω της χρήσης βιβλιοθηκών, όπως και οι περισσότερες πλατφόρμες προγραμματισμού. Οι βιβλιοθήκες παρέχουν επιπλέον λειτουργικότητα για χρήση σε σκίτσα, π.χ. εργασία με υλικό ή χειρισμό δεδομένων. Για να χρησιμοποιήσουμε μια βιβλιοθήκη σε ένα σκίτσο, την επιλέγουμε από το Sketch> Import Library.Ένας αριθμός βιβλιοθηκών είναι εγκατεστημένος με το IDE, αλλά μπορούμε επίσης να κάνουμε λήψη ή να δημιουργήσουμε τη δική μας.

Για να εγκαταστήσουμε μια νέα βιβλιοθήκη στο IDE του Arduino, μπορούμε να χρησιμοποιήσουμε τη Διαχείριση Βιβλιοθήκης (διαθέσιμη από την έκδοση 1.6.2 του IDE). Ανοίγουμε το IDE και κάνουμε κλικ στο μενού "Sketch" και, στη συνέχεια, επιλέγουμε «Include Library > Manage Libraries».



Εικόνα 3.10 Εμφάνιση λίστας βιβλιοθηκών

Στη συνέχεια, θα ανοίξει ο Διαχειριστής Βιβλιοθήκης(Library Manager) και θα εμφανιστεί μια λίστα βιβλιοθηκών που είναι ήδη εγκατεστημένες ή έτοιμες για εγκατάσταση. Μετακινούμαστε με κύλιση στη λίστα για να βρούμε τη βιβλιοθήκη που θέλουμε, κάνουμε κλικ σε αυτήν και, στη συνέχεια, επιλέγουμε την επιθυμητή έκδοση της βιβλιοθήκης.

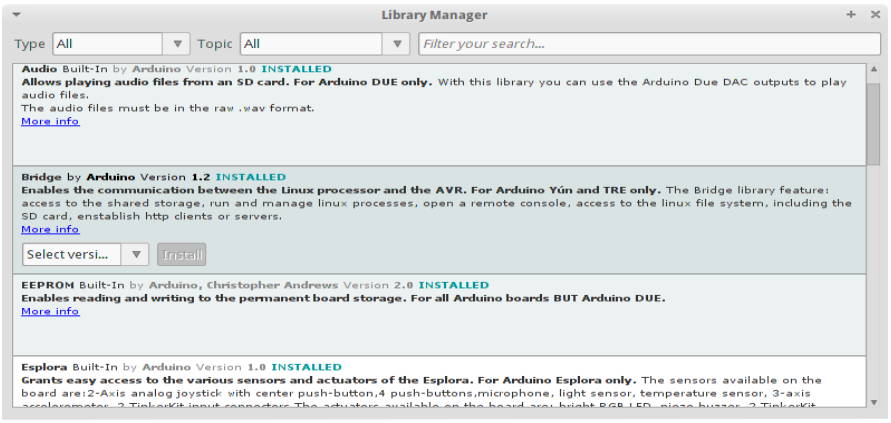


Εικόνα 3.11 Διαχειριστής βιβλιοθήκης

Τέλος, κάνουμε κλικ στην εγκατάσταση και περιμένουμε να εγκατασταθεί η IDE για τη νέα βιβλιοθήκη. Η λήψη ενδέχεται να χρειαστεί χρόνος ανάλογα με την ταχύτητα σύνδεσης. Αφού τελειώσει, πρέπει να εμφανιστεί μια ετικέτα εγκατεστημένη δίπλα στη βιβλιοθήκη μας.

3.3.4 Προγραμματισμός

Το IDE του Arduino υποστηρίζει τις γλώσσες C και C ++ χρησιμοποιώντας ειδικούς κανόνες για τη δομή του κώδικα. Ένα πρόγραμμα γραμμένο με το IDE του Arduino ονομάζεται sketch. Τα sketches αποθηκεύονται στον υπολογιστή ως αρχεία κειμένου με την επέκταση αρχείου .ino.



Εικόνα 3.12 Η βιβλιοθήκη εγκαταστάθηκε

3.3.4.1 Οι συναρτήσεις setup και loop

Ένα ελάχιστο πρόγραμμα Arduino αποτελείται απο δύο βασικές συναρτήσεις:

- ***setup():***Αυτή η συνάρτηση καλείται κατά την έναρξη του προγράμματος ή μετά από Reset και εκτελείται μόνο μία φορά. Χρησιμοποιείται για την αρχικοποιήση των μεταβλητών, της λειτουργικότητας των χρησιμοποιούμενων pins, την εισαγωγή κατάλληλων βιβλιοθηκών κλπ.
- ***loop():*** Αυτή η συνάρτηση καλείται μετά την setup( ) και είναι το κύριο πρόγραμμα που εκτελείται συνεχώς μέχρι να σβήσει η πλακέτα ή να γίνει Reset.

3.3.4.2 Βασικές εντολές

- **pinMode (pin, INPUT/OUTPUT)**: Ο ψηφιακός ακροδέκτης pin γίνεται είσοδος/έξοδος.
- **pinMode (pin, INPUT_PULLUP)**: Ο ψηφιακός ακροδέκτης pin γίνεται είσοδος, ενεργοποιώντας την εσωτερική αντίσταση pull-up.
- **int x= digitalRead(pin)**: Αναγνώριση της λογικής κατάστασης του ψηφιακού ακροδέκτη pin.
- **digitalWrite(pin,LOW/HIGH)**:Εγγραφή 0/1(0 V/+5V) στον ψηφιακό ακροδέκτη pin.

- **analogWrite(pin,value)**:Εγγραφή της τιμής value (0 εώς 255) στον ψηφιακό ακροδέκτη pin.
- **int x = analogRead(pin)** :Ανάγνωση της αναλογικής εισόδου pin (0 εώς).
- **analogReference("DEFAULT/INTERNAL/EXTERNAL")**:Τοποθέτηση της τάσης αναφοράς του A/D στα +5V/+1.1 V/AREF.
- **Serial.begin(9600)**: Ενεργοποίηση της σειριακής επικοινωνίας με baud rate=9600 bps.
- **Serial.println(x)**: Εκτύπωση στη σειρακή οθόνη της μεταβλητής x.
- **delay(ms)**: Καθυστέρηση ms. Ο επεξεργαστής δεν μπορεί να κάνει κάτι άλλο για ms milliseconds. Δεν επηρεάζονται η σειριακή επικοινωνία, οι έξο-δοι PWM και οι διακοπές.
- **millis()**: Επιστρέφει τον αριθμό από την έναρξη του προγράμματος (σαν unsigned long). Μηδενίζεται μετά από 50 ημέρες.

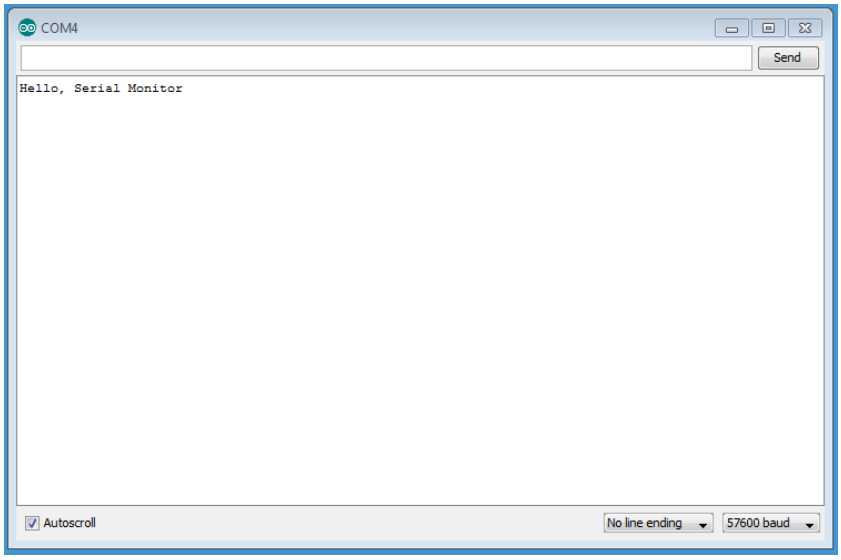
3.3.5 Σειριακή επικοινωνία (Serial)

Ένα από τα πλεονεκτήματα της πλατφόρμας είναι ότι το Arduino χρησιμοποιεί σειριακή επικοινωνία RS232, για να αλληλεπιδράσει με τους υπολογιστές. Αυτό σημαίνει ότι μπορούμε να στέλνουμε εντολές στην πλακέτα Arduino από έναν υπολογιστή που χρησιμοποιούμε και να λαμβάνουμε μηνύματα που στέλνονται από το Arduino μέσω USB.

Για να επιτευχθεί αυτό, είναι απαραίτητο να ανοίξουμε το παράθυρο του Arduino Serial Monitor, το οποίο είναι στην πραγματικότητα μέρος του λογισμικού της πλατφόρμας και μπορεί να βρεθεί στη γραμμή εργαλείων του IDE του Arduino. Το βοηθητικό πρόγραμμα επιτρέπει να διαβάσουμε εύκολα το σειριακό Arduino και να ελέγξουμε τη συμπεριφορά των συσκευών που είναι διασυνδεδεμένες με την πλακέτα. Επιπλέον, για την προβολή δεδομένων που δημιουργούνται με το board, το Serial Monitor μπορεί επίσης να βοηθήσει πραγματικά κατά την ανάπτυξη και την αποσφαλμάτωση ενός σκίτσου Arduino. Με τη λειτουργία Serial.print () που παρέχει, μπορούμε να έχουμε σειριακή έξοδο Arduino που αποστέλλεται στον υπολογιστή μας και εμφανίζεται στην οθόνη του.

Η σειριακή οθόνη είναι βολική για την ανάπτυξη σκίτσων Arduino,επειδή, μόλις μεταφορτωθεί ο κώδικας στην πλακέτα Arduino, το αποτέλεσμα μπορεί να είναι δια-φορετικό από αυτό που περιμέναμε.

Για παράδειγμα, μια λυχνία LED μπορεί να αναβοσβήνει πιο συχνά από ό, τι θα έπρεπε ή να μην κάνει τίποτα. Ο λόγος μπορεί να είναι ότι μια μεταβλητή μεταβάλλεται συχνά. Έτσι, το Arduino Serial Monitor φαίνεται να είναι ο ευκολότερος τρόπος για να δούμε τις μεταβαλλόμενες τιμές της.



Εικόνα 3.13 Σειριακή οθόνη

3.4 Arduino Shields

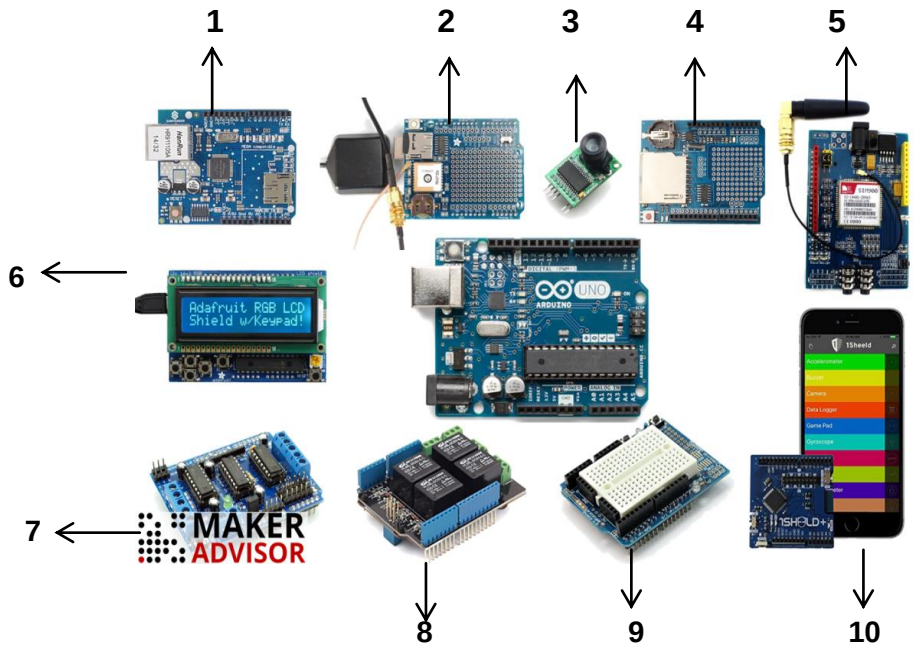
Οι ασπίδες (shields) Arduino είναι πλακέτες κυκλωμάτων που τοποθετούνται πάνω στο Arduino, για να του δώσουν επιπλέον λειτουργικότητα. Οι ασπίδες Arduino συχνά είναι στοιβαζόμενες, έτσι μπορούμε να συνδέσουμε πολλές μαζί και το Arduino να επιτελεί πολλές λειτουργίες ταυτόχρονα.

Οι ασπίδες συχνά παρέχονται είτε με ένα παράδειγμα (που υπάρχει μέσα στο Arduino) είτε με μια βιβλιοθήκη. Έτσι, όχι μόνο απλά συνδέονται με το Arduino, αλλά το μόνο που χρειάζεται να κάνουμε για να δουλεύουν είναι να ανεβάσουμε κάποιο παράδειγμα κώδικα στο Arduino.

Κάποιες από τις πιο δημοφιλείς ασπίδες Arduino παρουσιάζονται παρακάτω:

1. **Ethernet Shield**:Συνδέει το Arduino στο internet με καλώδίο Ethernet
2. **Adafruit GPS Logger Shield**: βοηθάει στη κατασκεύη ενός GPS tracker
3. **Arducam Mini Module Camera Shield**: περιέχει μια κάμερα με 5MP που διασυνδέεται με το Arduino μέσω SPI επικοινωνία.
4. **Data Logging Shield**: είναι χρήσιμη για την καταγραφή δεδομένων με κάρτα SD.

5. **SIM900 GSM GPRS Shield**: μας επιτρέπει να στέλνουμε μηνύματα SMS, MMS, GPRS και ήχο μέσω UART χρησιμοποιώντας εντολές AT
6. **Adafruit RGB LCD Shield w/ I2C Interface**: περιέχει μία οθόνη LCD RGB και ένα πληκτρολόγιο πέντε πλήκτρων
7. **L293D Motor Drive Shield**:Ελέγχει μέχρι 4 DC κινητήρες ή 2 βηματικούς ή 2 σερβοκηνητήρες
8. **Relay Shield:** Προσφέρει έναν εύκολο τρόπο για τον έλεγχο υψηλών τάσεων με τα τέσσερα ρελέ που περιέχει
9. **Proto Shield**:Περιέχει ένα μικρό breadboard.
10. **1Sheeld+**: η ασπίδα για iOS και Android. Μπορεί να μετατρέπει το smartphone σε πολλές διαφορετικές ασπίδες.



Εικόνα 3.14 Ασπίδες Arduino

ΚΕΦΑΛΑΙΟ 4ο : ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

Το Android είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας, το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά, αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με τη χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας τη συσκευή μέσω βιβλιοθηκών λογισμικού αναπτυγμένων από την Google.

Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

Εικόνα 4.1 Το λογότυπο του Android

4.1 Ιστορική αναδρομή

Το λειτουργικό σύστημα Android αναπτύχθηκε αρχικά από την ομώνυμη εταιρία Android Inc., η οποία εξαγοράστηκε από την Google το 2005. Η πρώτη επίσημη παρουσίασή του, ωστόσο, πραγματοποιήθηκε δύο χρόνια αργότερα, τον Νοέμβριο του 2007 με την ταυτόχρονη ίδρυση της κοινοπραξίας Open Handset Alliance, η οποία έχει ως στόχο την ανάπτυξη και εξέλιξη προτύπων για κινητές συσκευές.

Το λειτουργικό διατίθεται ελεύθερα, σύμφωνα με την άδεια λογισμικού Apache License. Στο πλαίσιο αυτό, επιτρέπεται σε τρίτους κατασκευαστές να ενσωματώσουν δικό τους κώδικα γραμμένο σε Java, χρησιμοποιώντας εργαλειοθήκες της εταιρίας.Οι λειτουργίες του Android είναι πάρα πολλές, οι οποίες, όπως είναι λογικό, αλλάζουν ανάλογα με την έκδοση.Το λογότυπο για το λειτουργικό σύστημα

Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Irina Blok.

4.2 Εκδόσεις

Android 1.0 : 23 Σεπτεμβρίου 2008

Android 1.1: 9 Φεβρουάριου 2009

Android 1.5: Cupcake, 27 Απριλίου 2009

Android 1.6: Donut, 15 Σεπτεμβρίου 2009

Android 2.0-2.1: Éclair, 26 Οκτωβρίου 2009

Android 2.2-2.2.3: Froyo, 20 Μαΐου 2010

Android 2.3-2.3.7: Gingerbread, 6 Δεκεμβρίου 2010

Android 3.0-3.2.6: Honeycomb, 22 Φεβρουάριου 2011

Android 4.0-4.04: Ice Cream Sandwich, 18 Οκτωβρίου 2011

Android 4.1-4.3.1: Jelly Bean, 9 Ιουλίου 2012

Android 4.4-4.4.4: KitKat, 31 Οκτωβρίου 2013

Android 5.0.5.1.1: Lollipop, 12 Νοεμβρίου 2014

Android 6.0-6.01: Marshamallow, 5 Οκτωβρίου 2015

Andorid 7.0-7.1.2: Nougat, 22 Αυγούστου 2016

Android 8.0-8.1: Oreo, 21 Αυγούστου 2017

Android 9.0: Pie, 6 Αυγούστου 2018

Εικόνα 4.2 Οι εκδόσεις Android και τα λογότυπά τους

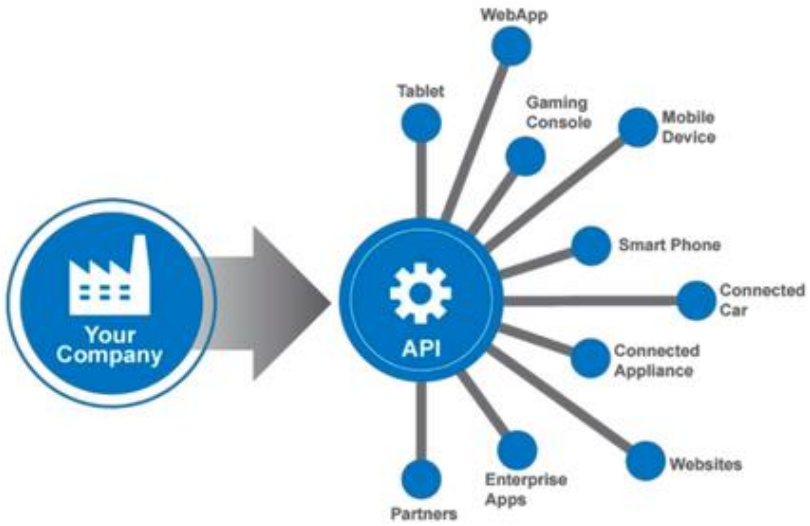
4.3 Application programming interface (API)

Μια διεπαφή προγραμματισμού εφαρμογών (API) είναι ένα σύνολο πρωτοκόλλων, ρουτινών, λειτουργιών ή / και εντολών που χρησιμοποιούν οι προγραμματιστές για την ανάπτυξη λογισμικού ή για τη διευκόλυνση της αλληλεπίδρασης μεταξύ διαφορετικών συστημάτων.

Τα API είναι διαθέσιμα τόσο για χρήση σε επιτραπέζιους υπολογιστές όσο και για κινητά και είναι συνήθως χρήσιμα για τον προγραμματισμό στοιχείων GUI (γραφική διεπαφή χρήστη), καθώς και για να επιτρέπεται σε ένα πρόγραμμα λογισμικού να ζητά και να δέχεται υπηρεσίες από άλλο πρόγραμμα.

Ένα καλό API, δηλαδή, διευκολύνει την ανάπτυξη ενός προγράμματος υπολογιστή, παρέχοντας όλα τα δομικά στοιχεία, τα οποία στη συνέχεια συνθέτονται από τον προγραμματιστή. Ένα API μπορεί να αφορά ένα web-based σύστημα, λειτουργικό σύστημα, σύστημα βάσης δεδομένων, υλικό υπολογιστή ή βιβλιοθήκη λογισμικού.

Μια προδιαγραφή API μπορεί να λάβει πολλές μορφές, αλλά συχνά περιλαμβάνει προδιαγραφές για ρουτίνες, δομές δεδομένων, τάξεις αντικειμένων, μεταβλητές ή απομακρυσμένες κλήσεις. Τα POSIX, το API των Windows και το ASPI είναι παραδείγματα διαφορετικών μορφών API.



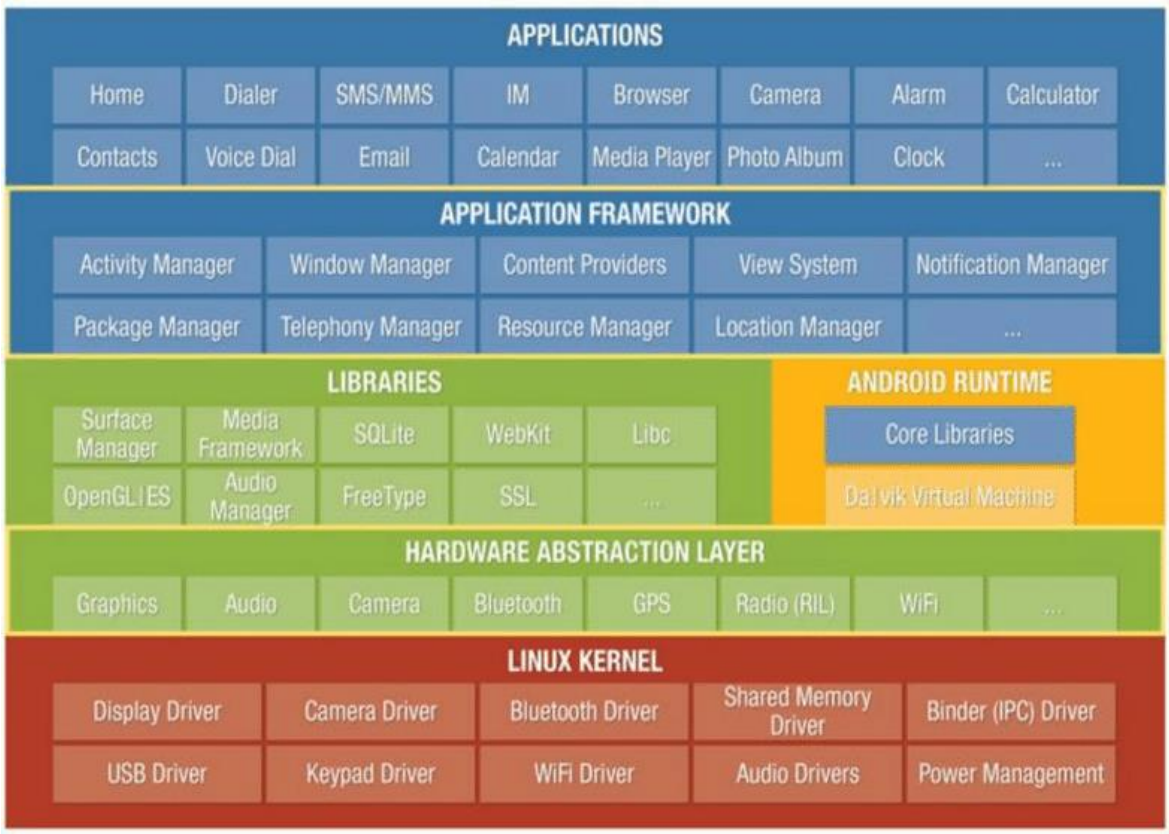
Εικόνα 4.3 Application programming interface

4.4 Η Αρχιτεκτονική του Android

Το Android έχει σχεδιαστεί με τη μορφή μιας στοίβας λογισμικού που περιλαμβάνει εφαρμογές, λειτουργικό σύστημα, περιβάλλον χρόνου εκτέλεσης, μεσαίο

λογισμικό, υπηρεσίες και βιβλιοθήκες. Αυτή η αρχιτεκτονική μπορεί, ίσως, να αναπαρασταθεί καλύτερα οπτικά, όπως περιγράφεται στην εικόνα 4.4.

Κάθε στρώμα της στοίβας και τα αντίστοιχα στοιχεία σε κάθε στρώμα είναι στενά ενσωματωμένα και προσεκτικά ρυθμισμένα, για να παρέχουν το βέλτιστο περιβάλλον ανάπτυξης και εκτέλεσης εφαρμογών για κινητές συσκευές.



Εικόνα 4.4 Η στοίβα λογισμικού του Android

4.4.1 Linux Kernel

Τοποθετημένο στο κάτω μέρος της στοίβας λογισμικού Android, ο πυρήνας Linux (Linux Kernel) παρέχει ένα επίπεδο επικοινωνίας μεταξύ του υλικού της συσκευής και των ανώτερων επιπέδων της στοίβας λογισμικού Android. Βασισμένο στην έκδοση 2.6 του Linux, ο πυρήνας παρέχει προληπτικές υπηρεσίες multitasking, χαμηλού επιπέδου υπηρεσίες πυρήνα, όπως η μνήμη, η διαδικασία και η διαχείριση ενέργειας, εκτός από την παροχή μιας στοίβας δικτύου και προγραμμάτων οδήγησης συσκευών, όπως η οθόνη συσκευής, Wi-Fi και ήχος.

Ο αρχικός πυρήνας του Linux αναπτύχθηκε το 1991 από τον Linus Torvalds και συνδυάστηκε με ένα σύνολο εργαλείων, βοηθητικών προγραμμάτων και μετα-

γλωττιστών που ανέπτυξε ο Richard Stallman στο Ίδρυμα Ελεύθερου Λογισμικού, για να δημιουργήσει ένα πλήρες λειτουργικό σύστημα που ονομάστηκε GNU / Linux. Διάφορες διανομές Linux προέρχονται από αυτές τις βασικές βάσεις, όπως το Ubuntu και το Red Hat Enterprise Linux.

Είναι σημαντικό να σημειωθεί, ωστόσο, ότι το Android χρησιμοποιεί μόνο τον πυρήνα του Linux. Ωστόσο, ο πυρήνας του Linux σχεδιάστηκε αρχικά για χρήση σε παραδοσιακούς υπολογιστές με τη μορφή επιτραπέζιων υπολογιστών και διακομιστών (servers). Στην πραγματικότητα, το Linux είναι το πλέον ευρύτερα αναπτυγμένο σε περιβάλλοντα κρίσιμων επιχειρηματικών διακομιστών. Είναι μια απόδειξη τόσο για τη δύναμη των σημερινών κινητών συσκευών όσο και για την αποδοτικότητα του πυρήνα του Linux.

4.4.2 Hardware Abstraction Layer (HAL)

Το Hardware Abstraction Layer (HAL) παρέχει τυποποιημένες διεπαφές που εκθέτουν τις δυνατότητες υλικού συσκευών στο πλαίσιο API ανώτερου επιπέδου Java. Το HAL αποτελείται από πολλαπλές ενότητες βιβλιοθήκης, κάθε μία από τις οποίες υλοποιεί μια διεπαφή για έναν συγκεκριμένο τύπο υλικού, όπως η κάμερα ή η μονάδα bluetooth. Όταν ένα API πλαισίου κάνει μια κλήση για πρόσβαση στο υλικό της συσκευής, το σύστημα Android φορτώνει τη λειτουργική μονάδα της βιβλιοθήκης για το εν λόγω στοιχείο υλικού.

4.4.3 Android Runtime

Αυτή η ενότητα παρέχει ένα βασικό στοιχείο, που ονομάζεται Dalvik Virtual Machine, ένα είδος Java Virtual Machine ειδικά σχεδιασμένο και βελτιστοποιημένο για το Android.Το Dalvik VM κάνει χρήση βασικών λειτουργιών του Linux, όπως διαχείριση μνήμης και πολλαπλών σπειρωμάτων.Το Dalvik VM επιτρέπει σε κάθε εφαρμογή Android να τρέχει στη δική της διαδικασία, με την παρουσία της virtual machine Dalvik.

Οι εφαρμογές που εκτελούνται σε εικονικές μηχανές παρέχουν ορισμένα πλεονεκτήματα. Πρώτα απ' όλα, δεν μπορούν να επηρεάσουν αρνητικά (σκόπιμα ή όχι) το λειτουργικό σύστημα ή άλλες εφαρμογές, ούτε μπορούν να έχουν άμεση πρόσβαση στο υλικό της συσκευής. Δεύτερον, καθίστανται ουδέτερες, επειδή δεν συνδέονται ποτέ με κάποιο συγκεκριμένο υλικό.

Η εικονική μηχανή Dalvik αναπτύχθηκε από την Google και βασίζεται στον υποκείμενο πυρήνα Linux για λειτουργικότητα χαμηλού επιπέδου. Είναι πιο αποδοτική από την τυπική Java VM, όσον αφορά τη χρήση της μνήμης, και έχει σχεδιαστεί ειδικά για να επιτρέπει σε πολλαπλές περιπτώσεις να εκτελούνται αποτελεσματικά εντός των περιορισμένων πόρων μιας κινητής συσκευής.

Το Runtime Android παρέχει, επίσης, ένα σύνολο βασικών βιβλιοθηκών (core libraries) που επιτρέπουν στους προγραμματιστές να γράφουν Android εφαρμογές, χρησιμοποιώντας την τυπική γλώσσα προγραμματισμού Java.

4.4.4 Οι Βιβλιοθήκες του Android

Αυτή η κατηγορία περιλαμβάνει τις βιβλιοθήκες που βασίζονται στην Java και οι οποίες είναι συγκεκριμένες για την ανάπτυξη του Android. Μεταξύ άλλων, περιλαμβάνονται οι βιβλιοθήκες πλαισίου εφαρμογών εκτός από εκείνες που διευκολύνουν τη δημιουργία περιβάλλοντος εργασίας χρήστη, την κατάρτιση γραφικών και την πρόσβαση στη βάση δεδομένων. Μια σύνοψη ορισμένων βασικών βιβλιοθηκών Android είναι η εξής:

- **android.app** : Παρέχει πρόσβαση στο μοντέλο εφαρμογής και είναι η βάση όλων των εφαρμογών Android.
- **android.content** : Διευκολύνει την πρόσβαση σε περιεχόμενο,τη δημοσίευση και την ανταλλαγή μηνυμάτων μεταξύ εφαρμογών και εξαρτημάτων εφαρμογών.
- **android.database**: Χρησιμοποιείται για την πρόσβαση σε δεδομένα που έχουν δημοσιευτεί από παρόχους περιεχομένου και περιλαμβάνει μαθήματα διαχείρισης βάσεων δεδομένων SQLite.
- **android.opengl** : Μια διεπαφή Java με το Api απεικόνισης 3D OpenGL ES.
- **android.os** : Παρέχει στις εφαρμογές πρόσβαση σε τυπικές υπηρεσίες λειτουργικού συστήματος, συμπεριλαμβανομένων μηνυμάτων, υπηρεσιών συστήματος και επικοινωνίας μεταξύ διαδικασιών.
- **android.text** : Χρησιμοποιείται για την απόδοση και τον χειρισμό κειμένου σε μια οθόνη συσκευής.
- **android.view** : Οι θεμελιώδεις δομικές μονάδες των διεπαφών χρήστη εφαρμογής.

- **android.widget** : Μια πλούσια συλλογή από προκατασκευασμένα στοιχεία διεπαφής χρήστη, όπως κουμπιά, ετικέτες, προβολές λίστας, διαχειριστές διάταξης, ραδιοφωνικά κουμπιά κλπ.
- **android.webkit** : Ένα σύνολο κλάσεων που επιτρέπουν στις εφαρμογές να έχουν δυνατότητες περιήγησης στο web.

4.4.5 Application Framework

Η Δομή Εφαρμογών (Application Framework) είναι ένα σύνολο υπηρεσιών, οι οποίες σχηματίζουν συλλογικά το περιβάλλον στο οποίο εκτελούνται και διαχειρίζονται εφαρμογές Android. Αυτό το πλαίσιο υλοποιεί την ιδέα ότι οι εφαρμογές Android κατασκευάζονται από επαναχρησιμοποιήσιμα, ανταλλάξιμα και αντικαταστάσιμα στοιχεία. Αυτή η ιδέα είναι ένα βήμα πιο πέρα από το ότι μια εφαρμογή είναι σε θέση να δημοσιεύσει τις δυνατότητές της, μαζί με οποιαδήποτε αντίστοιχα δεδομένα, έτσι ώστε να μπορούν να βρεθούν και να επαναχρησιμοποιηθούν από άλλες εφαρμογές.

Μερικές από τις κυριότερες υπηρεσίες του Android Framework είναι οι εξής:

- **Activity Manager** : Ελέγχει όλες τις πτυχές του κύκλου ζωής και της στοίβας δραστηριοτήτων.
- **Content Providers:** Επιτρέπει στις εφαρμογές να δημοσιεύουν και να μοιράζονται δεδομένα με άλλες εφαρμογές.
- **Resource Manager**: Παρέχει πρόσβαση σε ενσωματωμένους πόρους χωρίς κώδικα, όπως χορδές, ρυθμίσεις χρωμάτων και διατάξεις διασύνδεσης χρήστη.
- **Notifications Manager**: Επιτρέπει στις εφαρμογές την εμφάνιση ειδοποιήσεων στο χρήστη.
- **View System**: Ένα εκτεταμένο σύνολο προβολών, που χρησιμοποιούνται για τη δημιουργία διεπαφών χρήστη και εφαρμογών.
- **Package Manager** : Το σύστημα με το οποίο οι εφαρμογές είναι σε θέση να βρουν πληροφορίες σχετικά με άλλες εφαρμογές που είναι εγκατεστημένες στη συσκευή.
- **Telephony Manager**: Παρέχει πληροφορίες στην εφαρμογή σχετικά με τις υπηρεσίες τηλεφωνίας που είναι διαθέσιμες στη συσκευή, όπως η κατάσταση και οι πληροφορίες συνδρομητών.

- **Location Manager** : Παρέχει πρόσβαση στις υπηρεσίες τοποθεσίας, επιτρέποντας σε μια εφαρμογή να λαμβάνει ενημερώσεις σχετικά με αλλαγές τοποθεσίας.

4.4.6 Applications

Το Android διαθέτει ένα σύνολο βασικών εφαρμογών για μηνύματα ηλεκτρονικού ταχυδρομείου, μηνύματα SMS, ημερολόγια, περιήγηση στο Internet, επαφές και πολλά άλλα. Οι εφαρμογές που περιλαμβάνονται στην πλατφόρμα δεν έχουν ιδιαίτερη θέση μεταξύ των εφαρμογών που επιλέγει ο χρήστης για εγκατάσταση. Έτσι, μια εφαρμογή τρίτου μέρους μπορεί να γίνει το προεπιλεγμένο πρόγραμμα περιήγησης ιστού, ο αποστολέας SMS ή ακόμα και το προεπιλεγμένο πληκτρολόγιο (ισχύουν ορισμένες εξαιρέσεις, όπως η εφαρμογή ρυθμίσεων του συστήματος).

Οι εφαρμογές του συστήματος λειτουργούν ως εφαρμογές για χρήστες και παρέχουν βασικές δυνατότητες, στις οποίες μπορούν να έχουν πρόσβαση οι προγραμματιστές από τη δική τους εφαρμογή. Για παράδειγμα, εάν η εφαρμογή μας θα ήθελε να παραδώσει ένα μήνυμα SMS, δεν χρειάζεται να το κατασκευάσουμε μόνοι μας, μπορούμε να επικαλεστούμε όποια εφαρμογή SMS έχει ήδη εγκατασταθεί, για να αποστείλουμε ένα μήνυμα στον παραλήπτη που καθορίσαμε.

4.5 Android Studio

Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για ανάπτυξη εφαρμογών Android και είναι βασισμένο στο λογισμικό IntelliJ IDEA της JetBrains. Είναι διαθέσιμο για λήψη σε λειτουργικά συστήματα που βασίζονται σε Windows, MacOS και Linux. Πρόκειται για αντικατάσταση του Eclipse Android Development Tools (ADT) ως πρωτεύον IDE για την ανάπτυξη εφαρμογών Android.

Το Android Studio ανακοινώθηκε στις 16 Μαΐου 2013 στο συνέδριο I / O Google. Ήταν σε στάδιο πρώιμης προεπισκόπησης πρόσβασης, ξεκινώντας από την έκδοση 0.1 τον Μάιο του 2013, στη συνέχεια μπήκε σε beta στάδιο, ξεκινώντας από την έκδοση 0.8, που κυκλοφόρησε τον Ιούνιο του 2014. Το πρώτο σταθερό build κυκλοφόρησε τον Δεκέμβριο του 2014, ξεκινώντας από την έκδοση 1.0. Η τρέχουσα έκδοση είναι 3.1.3 και κυκλοφόρησε τον Ιούνιο του 2018.

Εικόνα 4.5 Το λογότυπο του Android Studio

Το Android Studio προσφέρει ακόμα περισσότερες λειτουργίες που βελτιώνουν την παραγωγικότητα κατά την ανάπτυξη εφαρμογών Android, όπως:

1.Ένα ευέλικτο σύστημα κατασκευής με βάση το Gradle

2.Έναν γρήγορο και πλούσιο σε χαρακτηριστικά εξομοιωτή

3.Ένα ενοποιημένο περιβάλλον, όπου μπορούμε να αναπτύξουμε εφαρμογές για όλες τις συσκευές Android

4. Άμεση Εκτέλεση (Instant Run), για να προωθηθούν αλλαγές στην τρέχουσα εφαρμογή χωρίς να δημιουργηθεί ένα νέο APK

5.Πρότυπα κώδικα και ενσωμάτωση GitHub για τη δημιουργία κοινών λειτουργιών εφαρμογής και την εισαγωγή δείγματος κώδικα

6.Εκτεταμένα εργαλεία και πλαίσια

7.Εργαλεία Lint για απόδοση, χρηστικότητα,συμβατότητα έκδοσης

8.Υποστήριξη C ++ και NDK

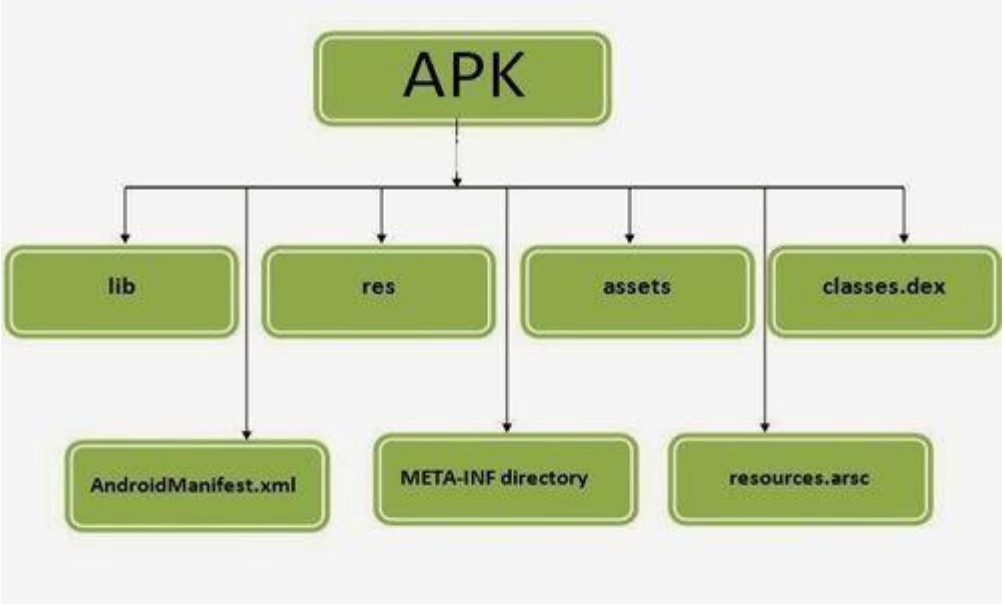
9.Ενσωματωμένη υποστήριξη για την πλατφόρμα Google Cloud, καθιστώντας εύκολη την ενσωμάτωση του Google Cloud Messaging και του App Engine

4.5.1 Android PacKage (APK)

Είναι αρχείο πακέτων που χρησιμοποιείται από το λειτουργικό σύστημα Android για τη διανομή και εγκατάσταση εφαρμογών για κινητές συσκευές. Για να δημιουργηθεί ένα αρχείο APK, πρώτα μεταγλωττίζεται ένα πρόγραμμα για Android και στη συνέχεια όλα τα μέρη του συσκευάζονται σε ένα αρχείο.

Ένα αρχείο APK περιέχει όλους τους κωδικούς ενός προγράμματος (όπως αρχεία .dex), τους πόρους (resources), τα στοιχεία ενεργητικού (assets), τα πιστοποιητικά (certificates) και το αρχείο δήλωσης (manifest file). Όπως συμβαίνει με πολλές μορφές αρχείων, τα αρχεία APK μπορούν να έχουν οποιοδήποτε όνομα που απαιτείται, υπό την προϋπόθεση ότι το όνομα του αρχείου τελειώνει στην επέκταση αρχείου ".apk".

Τα αρχεία APK μπορούν να εγκατασταθούν σε συσκευές που λειτουργούν με Android, όπως ακριβώς και η εγκατάσταση λογισμικού σε έναν υπολογιστή. Όταν ένας χρήστης κάνει λήψη και εγκαθιστά μια εφαρμογή Android, είτε από μια επίσημη πηγή (όπως το Google Play Store) είτε από έναν ανεπίσημο ιστότοπο, εγκαθιστά ένα αρχείο APK στη συσκευή του. Ένας χρήστης ή προγραμματιστής μπορεί, επίσης, να εγκαταστήσει ένα αρχείο APK απευθείας σε μια συσκευή (δηλαδή όχι μέσω λήψης από το δίκτυο) από έναν επιτραπέζιο υπολογιστή, χρησιμοποιώντας ένα πρόγραμμα επικοινωνίας όπως adb ή από μια εφαρμογή διαχείρισης αρχείων σε μια διαδικασία γνωστή ως περιστροφή (sideloading).



Εικόνα 4.6 Η δομή ενός αρχείου APK

4.5.2 Δομή του Android Project

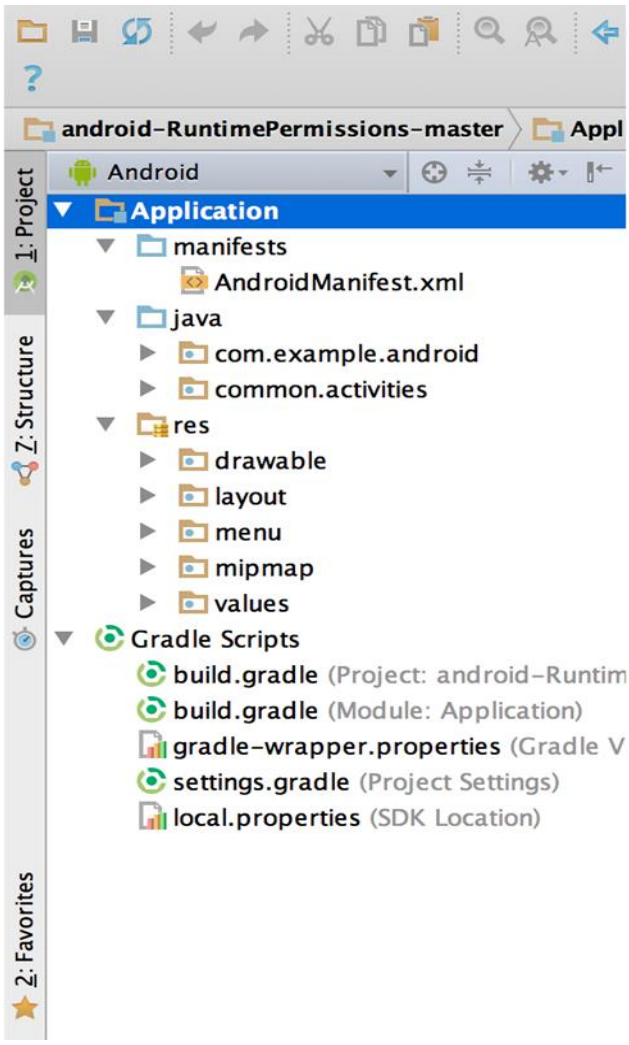
Κάθε έργο στο Android Studio περιέχει μία ή περισσότερες ενότητες με αρχεία πηγαίου κώδικα (source code files) και αρχεία πόρων (resource files). Οι τύποι ενοτήτων περιλαμβάνουν:

1.Μονάδες εφαρμογών Android (Android app modules)

2.Μονάδες βιβλιοθήκης(Library modules)

3.Υπομονάδες Google App Engine

Από προεπιλογή, το Android Studio εμφανίζει τα αρχεία του έργου στην προβολή έργου Android (Android project view), όπως φαίνεται στο σχήμα 4.7. Αυτή η προβολή οργανώνεται από ενότητες για γρήγορη πρόσβαση στα βασικά πηγαία αρχεία του έργου.



Εικόνα 4.7 Τα Project files

Όλα τα αρχεία δημιουργίας (build files) είναι ορατά στο επάνω επίπεδο κάτω από τα Gradle Scripts και κάθε ενότητα εφαρμογών περιέχει τους ακόλουθους φακέλους/καταλόγους:

manifests: Περιέχει το αρχείο AndroidManifest.xml.

java: Αυτό περιέχει τα αρχεία προέλευσης .java για το έργο μας. Από προεπιλογή, περιλαμβάνει το αρχείο προέλευσης MainActivity.java με μία κλάση δραστηριότητας, η οποία εκτελείται κατά την εκκίνηση της εφαρμογής μας.
res/drawable: Περιέχει συρόμενα αντικείμενα που έχουν σχεδιαστεί για οθόνες υψηλής ευκρίνειας.
res/layout:Περιέχει αρχεία που ορίζουν τη διεπαφή χρήστη της εφαρμογής μας.
res/values:Περιέχει διάφορα αρχεία XML που περιλαμβάνουν μία συλλογή πόρων, όπως ορισμοί συμβολοσειρών και χρωμάτων.
Gradle Scripts: Περιέχει όλα τα αρχεία σχετικά με το εργαλείο δημιουργίας Gradle.

4.5.2.1 Το αρχείο Main Activity

```
package com.example.paradeigma;


import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;


public class MainActivity extends AppCompatActivity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
   }
}
```

Εικόνα 4.8 Ο προεπιλεγμένος κώδικας του MainActivity.java

Ο κύριος κώδικας δραστηριότητας είναι ένα αρχείο Java, το MainActivity.java. Αυτό είναι το πραγματικό αρχείο εφαρμογής, που τελικά μετατρέπεται σε εκτελέσιμο Dalvik και τρέχει την εφαρμογή μας. Ο προεπιλεγμένος κώδικας που δημιουργήθηκε από τον οδηγό εφαρμογής παρουσιάζεται στην εικόνα 4.8. Εδώ, το R.layout.activity_main αναφέρεται στο αρχείο activity_main.xml, που βρίσκεται

στο φάκελο res / layout. Η μέθοδος onCreate () είναι μια από τις πολλές μεθόδους που υπολογίζονται όταν φορτώνεται μια δραστηριότητα.

4.5.2.2 Το αρχείο Manifest

Όποιο μέρος της εφαρμογής κι αν αναπτύξουμε, πρέπει να δηλώσουμε όλα τα συστατικά της σε ένα manifest.xml, που βρίσκεται στη ρίζα του καταλόγου έργου εφαρμογής. Αυτό το αρχείο λειτουργεί ως διεπαφή μεταξύ του Android OS και της εφαρμογής, οπότε αν δεν δηλώσουμε το στοιχείο μας σε αυτό το αρχείο, τότε δεν θα ληφθεί υπόψη από το λειτουργικό σύστημα. Για παράδειγμα, ένα προεπιλεγμένο αρχείο δήλωσης θα μοιάζει με το ακόλουθο:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.paradeigma">

   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:supportsRtl="true"
      android:theme="@style/AppTheme">

      <activity android:name=".MainActivity">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

Εικόνα 4.9 Το αρχείο manifest.xml

Εδώ, οι ετικέτες *<application> ... </ application>* περικλείουν τα στοιχεία που σχετίζονται με την εφαρμογή. Το *android:icon* θα παραπέμπει στο εικονίδιο της εφαρμογής που είναι διαθέσιμο στο αρχείο *res / drawable-hdpi.*

Η ετικέτα *<activity> ... </activity>* χρησιμοποιείται για τον προσδιορισμό μιας δραστηριότητας και το *android: name* προσδιορίζει το πλήρες όνομα κλάσης της υποκλάσεως *Activity* και το *android: label* καθορίζει μια συμβολοσειρά, που θα χρησιμοποιηθεί ως ετικέτα για τη δραστηριότητα. Μπορούμε, βεβαίως, να ορίσουμε πολλαπλές δραστηριότητες, χρησιμοποιώντας ετικέτες <activity>.

Το *action* του *<intent-filter>* ονομάζεται *android.intent.action.MAIN*, για να υποδείξει ότι η δραστηριότητα αυτή χρησιμεύει ως σημείο εισόδου για την εφαρμογή.

Επίσης, το *category* του *<intent-filter>* ονομάζεται *android.intent.category. LAUNCHER*, για να υποδείξει ότι η εφαρμογή μπορεί να εκκινηθεί από το εικονίδιο εκκίνησης της συσκευής.

Το *@string* αναφέρεται στο αρχείο *strings.xml* που εξηγείται παρακάτω. Επομένως, το *@ string / app_name* αναφέρεται στη συμβολοσειρά *app_name* που ορίζεται στο αρχείο *strings.xm*l. Με παρόμοιο τρόπο, γεμίζουν στην εφαρμογή και άλλες συμβολοσειρές.

4.5.2.3 Το αρχείο String

Το αρχείο *strings.xml* βρίσκεται στον φάκελο *res / values* και περιέχει όλο το κείμενο που χρησιμοποιεί η εφαρμογή μας. Για παράδειγμα, τα ονόματα κουμπιών, ετικετών, προεπιλεγμένου κειμένου και παρόμοιων τύπων συμβολοσειρών πηγαίνουν σε αυτό το αρχείο,το οποίο είναι υπεύθυνο για το περιεχόμενο κειμένου. Έτσι, ένα αρχείο προεπιλεγμένων συμβολοσειρών θα μοιάζει με το ακόλουθο:

```
<resources>

   <string name="app_name">HelloWorld</string>

   <string name="hello_world">Hello world!</string>

   <string name="menu_settings">Settings</string>

   <string name="title_activity_main">MainActivity</string>

</resources>
```

Εικόνα 4.10 Ένα τυπικό αρχείο strings.xml

4.5.2.4 Το αρχείο Layout

Το *activity_main.xml* είναι ένα αρχείο διαμόρφωσης διαθέσιμο στον κατάλογο *res / layout* και αναφέρεται από την εφαρμογή μας, όταν δημιουργεί τη διεπαφή

της. Θα χρειαστεί να τροποποιήσουμε αυτό το αρχείο πολύ συχνά για να αλλάξουμε το γραφικό περιβάλλον της εφαρμογής μας.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:padding="@dimen/padding_medium"
      android:text="@string/hello_world"
      tools:context=".MainActivity" />

</RelativeLayout>
```

Εικόνα 4.11 Ένα τυπικό αρχείο activity_main.xml

Το *TextView* είναι ένα στοιχείο ελέγχου Android που χρησιμοποιείται για τη δημιουργία του GUI και έχει διάφορα χαρακτηριστικά, όπως το *android: layout_width*,το *android:layout_height* κλπ, που χρησιμοποιούνται για τον καθορισμό του πλάτους, του ύψους και άλλων παραγόντων. Το *@string* αναφέρεται στο αρχείο *strings.xml*, που βρίσκεται στον φάκελο *res / values*. Για παράδειγμα, το *@string / hello_world* αναφέρεται στη συμβολοσειρά hello που ορίζεται στο αρχείο *strings.xml*, της οποίας το περιεχόμενο είναι "Hello World!".

4.5.3 Σύστημα κατασκευής Gradle

Το Android Studio χρησιμοποιεί το Gradle ως το θεμέλιο του συστήματος δημιουργίας με περισσότερες δυνατότητες που σχετίζονται με το Android και παρέχονται από το Android plugin για το Gradle. Αυτό το σύστημα δημιουργίας λει-

τουργεί ως ένα ολοκληρωμένο εργαλείο από το μενού Android Studio και ανεξάρτητα από τη γραμμή εντολών. Μπορούμε να χρησιμοποιήσουμε τις δυνατότητες του συστήματος δημιουργίας, για να κάνουμε τα εξής:

1.Να προσαρμόσουμε, να διαμορφώσουμε και να επεκτείνουμε τη διαδικασία δημιουργίας.

2.Να δημιουργήσουμε πολλά APK για την εφαρμογή μας, με διαφορετικές λειτουργίες που χρησιμοποιούν το ίδιο έργο και ενότητες.

3.Να επαναχρησιμοποιήσουμε τον κώδικα και τους πόρους σε όλες τις πηγές.

Χρησιμοποιώντας την ευελιξία του Gradle, μπορούμε να επιτύχουμε όλα αυτά χωρίς να τροποποιήσουμε τα βασικά πηγαία αρχεία της εφαρμογής μας. Τα αρχεία δημιουργίας του Android Studio ονομάζονται build.gradle. Πρόκειται για αρχεία απλού κειμένου που χρησιμοποιούν τη σύνταξη Groovy, για να διαμορφώσουν την κατασκευή με στοιχεία που παρέχονται από το Android plugin για Gradle. Κάθε έργο έχει ένα αρχείο δημιουργίας κορυφαίου επιπέδου για ολόκληρο το έργο και ξεχωριστά αρχεία κατασκευής σε επίπεδο ενότητας για κάθε ενότητα. Όταν εισάγουμε ένα υπάρχον έργο, το Android Studio δημιουργεί αυτόματα τα απαραίτητα αρχεία δημιουργίας.

4.5.4 Τρέξιμο μίας εφαρμογής

Όταν δημιουργούμε μια εφαρμογή Android, είναι σημαντικό να δοκιμάζουμε πάντα την εφαρμογή μας σε μια πραγματική συσκευή πριν την διαθέσουμε στους χρήστες.Προτού ξεκινήσουμε να εντοπίζουμε σφάλματα στη συσκευή μας, πρέπει να κάνουμε τα εξής:

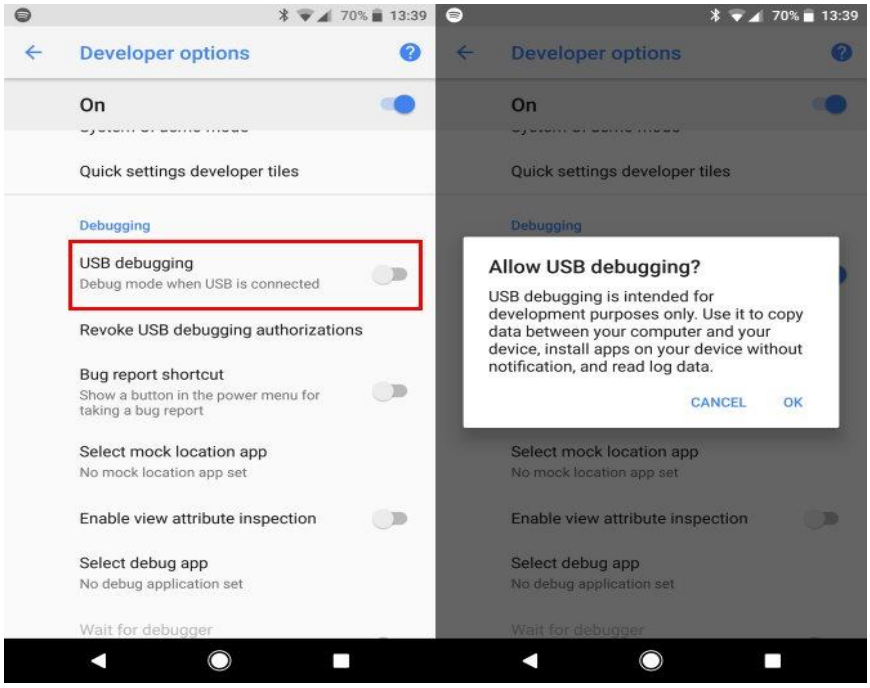
1.Στη συσκευή, ανοίγουμε την εφαρμογή *Ρυθμίσεις* (Settings), επιλέγουμε *Επιλογές προγραμματιστή* (Developers options) και στη συνέχεια, ενεργοποιούμε την *εντολή σφαλμάτων USB* (USB debugging).

2. Ρυθμίζουμε το σύστημά μας για να εντοπίσουμε τη συσκευή μας.

3. Windows: Εγκαθιστούμε ένα πρόγραμμα οδήγησης USB για το Bridge Debug Android (adb).

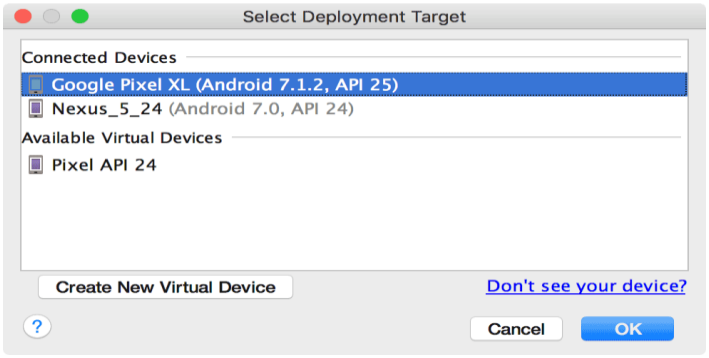
4. Mac OS X: Απλά λειτουργεί. Παραλείπουμε αυτό το βήμα.

5. Ubuntu Linux: Χρησιμοποιούμε το *apt-get install,* για να εγκαταστήσουμε το πακέτο *android-tools-adb*. Αυτό μάς δίνει ένα προκαθορισμένο σύνολο κανόνων udev που διατηρεί η κοινότητα για όλες τις συσκευές Android.



Εικόνα 4.12 Ενεργοποίηση του USB debugging από το Developer options

Όταν ολοκληρώσουμε τις ρυθμίσεις και συνδέσουμε τη συσκευή μέσω USB, μπορούμε να κάνουμε κλικ στην επιλογή *Εκτέλεση* (Run) στο Android Studio, για να δημιουργήσουμε και να εκτελέσουμε την εφαρμογή μας. Εάν πρόκειται για την πρώτη φορά που εκτελούμε την εφαρμογή, το Android Studio μάς ζητά να επιλέξουμε έναν στόχο ανάπτυξης (Deployment target), όπως φαίνεται στην εικόνα 4.13. Επιλέγουμε μια συσκευή ή μια εικονική μηχανή, για να εγκαταστήσουμε και να εκτελέσουμε την εφαρμογή μας.



Εικόνα 4.13 Επιλογή συσκευής, στην οποία θα δημιουργηθεί η εφαρμογή

ΚΕΦΑΛΑΙΟ 5^{ο} Η ΤΕΧΝΟΛΟΓΙΑ BLUETOOTH

Ένας από τους βασικούς στόχους της παρούσας εργασίας είναι να επιτευχθεί ο ασύρματος έλεγχος του ρομποτικού οχήματος από την κινητή συσκευή μέσω της εφαρμογής Android. Ως προσφορότερος τρόπος για την ενέργεια αυτή, επιλέχθηκε η τεχνολογία Bluetooth, διότι είναι πολύ διαδεδομένη και ιδιαίτερα εύχρηστη και οικονομική.

Στο παρόν κεφάλαιο, λοιπόν, θα παρουσιαστεί συνοπτικά το πρωτόκολλο Bluetooth.

Εικόνα 5.1 Το λογότυπο του Bluetooth

5.1 Τι είναι το Bluetooth

Είναι ένα βιομηχανικό πρότυπο για ασύρματα προσωπικά δίκτυα υπολογιστών (Wireless Personal Area Networks, WPAN). Πρόκειται για μια ασύρματη τηλεπικοινωνιακή τεχνολογία μικρών αποστάσεων, η οποία μπορεί να μεταδώσει σήματα μέσω μικροκυμάτων σε ψηφιακές συσκευές.Επομένως, το Bluetooth είναι ένα πρωτόκολλο, το οποίο παρέχει προτυποποιημένη, ασύρματη επικοινωνία ανάμεσα σε προσωπικούς ψηφιακούς οδηγούς ή υπολογιστές παλάμης (personal digital assistant, PDA), κινητά τηλέφωνα, φορητούς υπολογιστές, προσωπικούς υπολογιστές, εκτυπωτές, καθώς και ψηφιακές φωτογραφικές μηχανές ή ψηφιακές κάμερες, μέσω μιας ασφαλούς, χαμηλού κόστους και παγκοσμίως διαθέσιμης (χωρίς ειδική άδεια) ραδιοσυχνότητας μικρής εμβέλειας.

5.2 Ιστορική αναδρομή.

Ως τα τέλη της δεκαετίας του 1990, δεν υπήρχε κάποιο ευρέως αποδεκτό πρότυπο WPAN, ούτε φυσικά ανάλογες εμπορικές εφαρμογές / πομποδέκτες. Όμως, τότε, η Ericsson έθεσε τις βάσεις για την ανάπτυξη μιας τεχνολογίας, η οποία θα επέτρεπε τον σχηματισμό τοπικών δικτύων πολύ μικρής εμβέλειας, με σκοπό την ασύρματη επικοινωνία ετερογενών φορητών συσκευών.

Το πρότυπο που προέκυψε υιοθετήθηκε στη συνέχεια από την IEEE ως το πρότυπο 802.15 για WPAN. Οι σχεδιαστές του κλήθηκαν να επιλέξουν το όνομα με το οποίο αυτή η τεχνολογία θα γινόταν αργότερα γνωστή σε όλο τον κόσμο. Οι Σουηδοί εμπνευστές του 802.15 ήταν βέβαιοι ότι το νέο πρότυπο θα επικρατούσε και θα έφερνε ακόμη πιο κοντά τους ανθρώπους και τις συσκευές τους. Έλεγαν, μάλιστα, ότι κάτι ανάλογο έκανε και ο Δανός Βασιλιάς Χάραλντ ο Κυανόδους, ο οποίος έζησε στα τέλη του 10ου αιώνα μ.Χ.: κατέλαβε με τα στρατεύματά του πολλές χώρες, ενώ λέγεται ότι κατάφερε να ενώσει τη Δανία με τη Νορβηγία. Από το όνομά του, που έχει τις ρίζες του σε δύο αρχαίες δανέζικες λέξεις: bla (σκουρόδερμος) και tan (γενναίος άνδρας),γεννήθηκε το “Bluetooth”. Αυτά τουλάχιστον αναφέρονται σε ένα δελτίο τύπου της Ericsson το οποίο δημοσιεύθηκε το 1999.

Τις προδιαγραφές της συγκεκριμένης τεχνολογίας ανέπτυξε και υποστηρίζει το Bluetooth Special Interest Group (SIG), ενώ η τελευταία «δημόσια» έκδοσή του είναι η 1.1, η οποία ενσωματώνεται πλέον στις περισσότερες συμβατές συσκευές μέσω κατάλληλων πομποδεκτών και καρτών δικτύου.

5.3 Εφαρμογές

Το Bluetooth επιτρέπει την κατάργηση όλων των καλωδίων, τα οποία παλαιότερα ήταν απαραίτητα για τη «διασύνδεση» μεταξύ υπολογιστών, φορητών υπολογιστών χειρός, κινητών τηλεφώνων και άλλων ψηφιακών συσκευών, όπως ψηφιακές κάμερες, σαρωτές, εκτυπωτές, μικρόφωνα, ακουστικά, ραδιόφωνα κ.α. Το Bluetooth επιτρέπει τη σύνδεση του κινητού με τον υπολογιστή, τη μεταφορά δεδομένων,όπως εικόνες, επαφές και σημειώσεις από κινητό προς κινητό, τη σύνδεση στο Internet κ.α. Όλα αυτά χωρίς καλώδια και πολύπλοκες ρυθμίσεις.

Οι εφαρμογές του λοιπόν είναι πολλαπλές:

- Ασύρματη δικτύωση μεταξύ επιτραπέζιου και φορητού υπολογιστή σε έναν περιορισμένο χώρο με ελάχιστο διαθέσιμο εύρος ζώνης

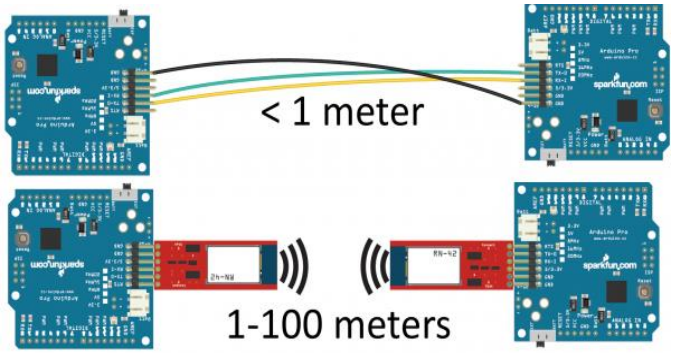
- Ασύρματα περιφερειακά, όπως εκτυπωτές, ποντίκια και πληκτρολόγια, τα οποία επικοινωνούν με κάποιον επιτραπέζιο ή φορητό υπολογιστή
- Ασύρματη μεταφορά ψηφιακών αρχείων (εικόνες, mp3 κλπ) ανάμεσα σε κινητά τηλέφωνα και PDA
- Ασύρματα ακουστικά για κινητά τηλέφωνα και Smartphone
- Ιατρικές εφαρμογές : δοκιμάζονται συσκευές από εταιρίες που παρέχουν ηλεκτρονικές συσκευές προχωρημένης ιατρικής.
- Ορισμένοι δέκτες GPS μεταφέρουν πληροφορίες NMEA μέσω Bluetooth.
- Ασύρματη τηλεφωνία στο αυτοκίνητο: Το Bluetooth δίνει τη δυνατότητα σε χρήστες καταλλήλως εξοπλισμένων κινητών τηλεφώνων να χρησιμοποιούν κάποιες βασικές λειτουργίες τους με ασύρματα ακουστικά. Ανάλογο σύστημα υπάρχει ενσωματωμένο και σε κράνη οδηγών μοτοσικλέτας, επιτρέποντας τη συνομιλία κατά την οδήγηση.
- Απομακρυσμένος έλεγχος συσκευών (έως την εμφάνιση του Bluetooth, χρησιμοποιούσαν τεχνολογία υπέρυθρων ακτίνων)

Οι άλλες τεχνολογίες ασύρματης πρόσβασης, όπως το 802.11 (WiFi), προσδιορίζουν τα ίδια χαρακτηριστικά, ανεξάρτητα από τις εφαρμογές για τις οποίες χρησιμοποιούνται. Αντίθετα, το Bluetooth SIG ορίζει παραμετροποιημένες εκδοχές του προτύπου για την ικανοποίηση της εκάστοτε εφαρμογής. Οι εκδοχές αυτές, που αποκαλούνται προφίλ, περιλαμβάνουν πρότυπα για όλα τα επίπεδα της στοίβας πρωτοκόλλων του Bluetooth. Ενώ οι προδιαγραφές Bluetooth ορίζουν πώς λειτουργεί η τεχνολογία, τα προφίλ ορίζουν πώς χρησιμοποιείται.

Ένα παράδειγμα προφίλ, το οποίο χρησιμοποιείται και στο Arduino, είναι το προφίλ σειριακής θύρας (Serial Port Profile, SPP). Ο σκοπός του είναι η αντικατάσταση των καλωδίων RS-232, ώστε να αυξηθεί η εμβέλεια της επικοινωνίας μεταξύ δύο συσκευών. Για παράδειγμα, τα δύο Arduino της εικόνας 5.2 στέλνουν και λαμβάνουν δεδομένα από μεγάλη απόσταση, μέσω των θυρών Tx και Rx αντίστοιχα.

5.4 Λειτουργία

Το Bluetooth λειτουργεί στην ελεύθερη φασματική ζώνη συχνοτήτων ISM (Industrial, Scientific and Medical), η οποία εκτείνεται από τα 2400 έως τα 2483.5MHz. Στην ίδια ζώνη, συνυπάρχουν και άλλα πρωτόκολλα RF (Radio Frequency),όπως το ZigBee και το WiFi.



Εικόνα 5.2 Επικοινωνία δύο Arduino ασύρματα μέσω Bluetooth και ενσύρματα

Το Bluetooth προδιαγράφει τρία επίπεδα (τάξεις) ισχύος εκπομπής, από τα οποία εξαρτάται και η εμβέλεια επικοινωνίας. Ορισμένες συσκευές είναι σε θέση να λειτουργούν μόνο σε μία τάξη ισχύος, ενώ άλλες μπορούν να μεταβάλλουν την ισχύ εκπομπής τους.

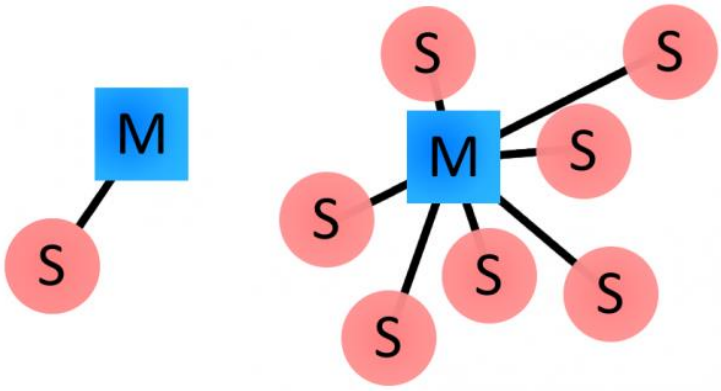
Πίνακας 5.1 Χαρακτηριστικά εκπομπής του Bluetooth

Αριθμός κλάσης	Μέγιστη ισχύς εξόδου (dBm)	Μέγιστη ισχύς εξόδου (mW)	Εμβέλεια επικοινωνίας
1	20	100	100 m
2	4	2.5	10 m
3	0	1	10 cm

Τα δίκτυα Bluetooth, γνωστά και ως μικροσκοπικά δίκτυα (piconets), χρησιμοποιούν ένα μοντέλο master/slave, για να ελέγχουν πότε οι συσκευές μπορούν να στείλουν δεδομένα. Ένα piconet διαθέτει έναν κόμβο master, ενώ οι υπόλοιποι (έως και 7) συμπεριφέρονται ως slaves, καθένας από τους οποίους επικοινωνεί αποκλειστικά με τον master. Ένας κόμβος slave μπορεί να ανήκει ταυτόχρονα σε δύο ή περισσότερα piconets στον ίδιο χώρο. Δύο ή περισσότερα piconets που επικοινωνούν μεταξύ τους σχηματίζουν ένα διάσπαρτο δίκτυο (scatternet).

Για να συντονιστούν οι μεταδόσεις των κόμβων ενός piconet, το Bluetooth εκμεταλλεύεται την τεχνική πολλαπλής πρόσβασης με διαίρεση χρόνου (Time Division Multiple Access, TDMA). Κατά την TDMA, κάθε συσκευή στέλνει

δεδομένα σε αυστηρά καθορισμένες χρονοθυρίδες (time slots) διάρκειας 625μs. Ο master εκπέμπει στις περιττές χρονοθυρίδες, ενώ οι slaves στις άρτιες.



Εικόνα 5.3 Piconet, Master και Slaves

Προκειμένου να μεταδώσει δεδομένα, ένας slave πρέπει να λάβει ένα μήνυμα παραχώρησης 1, 3 ή 5 χρονοθυρίδων, τα οποία εκπέμπονται από τον master με περιοδικό τρόπο. Στο χρονικό διάστημα που θα του αποδοθεί, ο slave μπορεί να εκπέμψει ένα πλαίσιο που περιέχει έως και 2745 bits δεδομένων.

Ένα πρόβλημα που προκύπτει από τη λειτουργία του Bluetooth σε ελεύθερη φασματική ζώνη είναι οι παρεμβολές από γειτονικές συσκευές στην ίδια ζώνη (πχ. WiFi). Προκειμένου να μειωθεί η επίδραση των παρεμβολών, το Bluetooth κάνει χρήση της μεθόδου διασποράς φάσματος με εναλλαγή συχνότητας (Frequency Hopping Spread Spectrum, FHSS). Σύμφωνα με την FHSS, η μετάδοση μιας συσκευής μπορεί να πραγματοποιείται σε διαφορετική συχνότητα σε κάθε χρονοσχισμή (έως 1600 εναλλαγές ανά δευτερόλεπτο), όμως η συχνότητα δεν μπορεί να αλλάξει κατά τη διάρκεια εκπομπής ενός πλαισίου. Η τακτική αλλαγή της συχνότητας εκπομπής καθορίζεται με ψευδοτυχαίο τρόπο από τον master.

5.5 Διευθύνσεις Bluetooth και ονόματα

Κάθε συσκευή Bluetooth χαρακτηρίζεται από μια παγκοσμίως μοναδική διεύθυνση μήκους 48 bits, η οποία παρουσιάζεται συνήθως σε δεκαεξαδική μορφή. Τα 24 πιο σημαντικά bits της διεύθυνσης αντιστοιχούν στον κατασκευαστή της συσκευής, ενώ τα υπόλοιπα είναι μοναδικά για την εκάστοτε συσκευή. Σε κάποιες περιπτώσεις (πχ. πομποδέκτες Bluetooth) η διεύθυνση αυτή αναγράφεται πάνω

στη συσκευή, ενώ σε άλλες (υπολογιστές, smartphones) μπορεί να βρεθεί μέσω του λειτουργικού συστήματος που τρέχει σε αυτήν. Προκειμένου να διευκολυνθεί η αναζήτηση συσκευών Bluetooth, εκτός από τη διεύθυνση, κάθε συσκευή διαθέτει ένα φιλικό προς τους χρήστες όνομα που επιλέγεται από αυτούς. Τα ονόματα των συσκευών μπορούν να έχουν μήκος έως 248 bytes, ενώ δύο συσκευές μπορούν να χαρακτηρίζονται από το ίδιο όνομα, σε αντίθεση με τη μοναδικότητα των διευθύνσεων.

5.6 Διαδικασία σύνδεσης

Ο σχηματισμός ενός piconet εκκινεί από έναν κόμβο, ο οποίος επιθυμεί να αποκτήσει το ρόλο του master. Αρχικά, ο εν δυνάμει master κόμβος αναζητά την ύπαρξη πιθανών κόμβων slaves με την εκπομπή ενός μηνύματος-ερώτησης (inquiry). Όσοι κόμβοι λάβουν αυτό το μήνυμα, απαντούν με ένα πλαίσιο που περιλαμβάνει πληροφορίες αναγνώρισης αυτών, όπως το όνομα και τη διεύθυνσή τους. Αφού ο master λάβει τις απαντήσεις των slaves, αποστέλλει στον καθένα ξεχωριστά ένα μήνυμα καλέσματος (paging) αυτού. Όταν ολοκληρωθεί η διαδικασία paging, μετά από πρόσθετες ανταλλαγές μηνυμάτων του master με τους slaves, οι συσκευές είναι συνδεδεμένες. Όσο μία συσκευή παραμένει συνδεδεμένη στο piconet, μπορεί να βρίσκεται σε μία από τις ακόλουθες καταστάσεις:

- **Ενεργή λειτουργία (active mode)**: Οι σταθμοί είναι συνδεδεμένοι στο δίκτυο και μπορούν να ανταλλάσσουν δεδομένα.
- **Λειτουργία Ακρόασης (sniff mode)**: Πρόκειται για λειτουργία εξοικονόμησης ενέργειας, στην οποία μία συσκευή ακούει για μεταδώσει μόνο σε συγκεκριμένες χρονοσχισμές (πχ. κάθε 100ms).
- **Λειτουργία κράτησης (hold mode)**: Πρόκειται, επίσης, για λειτουργία εξοικονόμησης ενέργειας, κατά την οποία ένας slave είναι ανενεργός, ενώ επανέρχεται στην κατάσταση active μετά το πέρας ενός δεδομένου χρονικού διαστήματος. Η μετάβαση μιας συσκευής στην κατάσταση hold μπορεί να εκκινήσει από τον master.
- **Λειτουργία στάθμευσης (parking mode)** : Κατά τη λειτουργία αυτή, ένας slave είναι ανενεργός, όμως, σε αντίθεση με την προηγούμενη περίπτωση, η επαναφορά στην ενεργή κατάσταση μπορεί να πραγματοποιηθεί αποκλειστικά από τον master.

Οι κόμβοι που βρίσκονται σε καταστάσεις Active, Sniff και Hold αναγνωρίζονται από διευθύνσεις μεγέθους 3 bits, εξ ου και ο μέγιστος αριθμός των 7 slaves. Αντίθετα, οι κόμβοι στην κατάσταση parked, που δεν θεωρούνται συνδεδεμένοι στο δίκτυο, χαρακτηρίζονται από διευθύνσεις μεγέθους 8 bits, οπότε ο μέγιστος αριθμός των ανενεργών slaves ανέρχεται στους 256.

Αφού δύο συσκευές Bluetooth συνδεθούν μεταξύ τους, υπάρχει η δυνατότητα εγκαθίδρυσης νέας σύνδεσης χωρίς να προηγηθεί η ανωτέρω διαδικασία. Ο «δεσμός» (bonding) δύο συσκευών επιτυγχάνεται με την ολοκλήρωση μιας διαδικασίας που ονομάζεται «ζευγάρωμα» (pairing). Πριν αρχίσει την μετάδοση δεδομένων, κάθε συσκευή πρέπει να αναγνωρίσει την άλλη μέσω μίας διαδικασίας επιβεβαίωσης. Παλαιότερα, οι χρήστες επέλεγαν έναν τετραψήφιο αριθμό προσωπικής αναγνώρισης (Personal Identification Number, PIN), ο οποίος θα ήταν γνωστός και στις δύο συσκευές. Η μέθοδος αυτή, όμως, υστερούσε σε θέματα ασφάλειας, καθώς ένας τρίτος χρήστης μπορούσε να μαντέψει εύκολα το PIN. Με τη νέα μέθοδο, κάθε χρήστης πρέπει να εισάγει έναν μεγαλύτερο κωδικό που είναι ήδη γνωστός στην άλλη συσκευή, δηλαδή δεν επιλέγεται από τους χρήστες.

Αφού ολοκληρωθεί το ζευγάρωμα, πρέπει να σχηματιστεί μία σύνδεση μεταξύ των συσκευών, ώστε αυτές να μπορούν να ανταλλάξουν δεδομένα. Οι συνδέσεις εγκαθιδρύονται από ένα πρωτόκολλο που λειτουργεί στο στρώμα συνδέσμου μετάδοσης δεδομένων (το αντίστοιχο MAC για το Bluetooth). Υπάρχουν δύο τύποι συνδέσεων για τη μεταφορά δεδομένων:

1) **Σύγχρονες ή SCO** (Synchronous Connection Oriented): Χρησιμοποιούνται για τη μεταφορά χρονικά κρίσιμων δεδομένων, όπως η φωνή. Κάθε κόμβος μπορεί να δεσμεύει μέχρι μία χρονοθυρίδα, στην οποία μεταδίδονται δεδομένα με μέγιστο ρυθμό 64 kbps. Επιπλέον, ένας slave μπορεί να διαθέτει έως και τρεις συνδέσεις SCO με τον master, οι οποίες είναι αυστηρά σημείο προς σημείο. Λόγω της μεταφοράς κρίσιμων δεδομένων, οι SCO δεν κάνουν χρήση επαναμεταδόσεων, όμως μπορούν να χρησιμοποιηθούν κώδικες ανίχνευσης και διόρθωσης σφαλμάτων FEC (Forward Error Correction) για την προστασία των δεδομένων.

2) **Ασύγχρονες ή ACL** (Asynchronous Connection-Less): Τυπικά χρησιμοποιούνται για τη μετάδοση δεδομένων μεταγωγής πακέτου. Σε αυτήν την περίπτωση, ένας κόμβος μπορεί να δεσμεύσει 1, 3 ή 5 χρονοθυρίδες για την εκπομπή ενός πλαισίου με μέγιστο ρυθμό μετάδοσης 724 kbps. Κάθε slave μπορεί να διαθέτει μοναδική σύνδεση ACL με τον Master, ενώ παρέχεται και η δυνατότητα πολυ-

διανομής (multicasting). Επειδή τα δεδομένα είναι ανεκτικά σε καθυστέρηση, κάποια πλαίσια μπορεί να χαθούν, οπότε αυτά αναμεταδίδονται.

5.7 Εκδόσεις Bluetooth

Το Bluetooth εξελίσσεται συνεχώς μετά την πρώτη έκδοσή του, το 1994. Οι επόμενες εκδόσεις σχεδιάσθηκαν ώστε να προσφέρουν υψηλότερους ρυθμούς μετάδοσης, χαμηλότερη κατανάλωση ισχύος και μεγαλύτερη εμβέλεια, ενώ οι νεότερες υποστηρίζουν και την επικοινωνία μεταξύ συσκευών Internet of Things (IoT). Στον Πινακα 5.2 συγκρίνονται τα κυριότερα χαρακτηριστικά των εκδόσεων του Bluetooth.

Πίνακας 5.2 Σύγκριση εκδόσεων Bluetooth

Έκδοση	Έτος έκδοσης	Ρυθμός μετάδοσης	Εμβέλεια επικοινωνίας (m)	Πρόσθετα χαρακτηριστικά
1.x	1994	723 kbps	10	-
2	2004	2.1 Mbps	10	Βελτιωμένος ρυθμός μετάδοσης
2.1	2007	3 Mbps	10	Ασφαλές ζευγάρωμα
3	2009	24 Mbps	10	Υψηλή ταχύτητα με τη βοήθεια του Wi-Fi
4.0	2010	25 Mbps	50	Χαμηλή κατανάλωση ισχύος
4.1	2013	25 Mbps	50	Υποστήριξη συσκευών IoT
4.2	2014	25 Mbps	50	Πρόσβαση στο Internet με χρήση του IPv6
5	2016	50 Mbps	200	Τετραπλάσια εμβέλεια και διπλάσια ταχύτητα

5.8 Σύγκριση με άλλες τεχνολογίες

Το Bluetooth δεν είναι, προφανώς, η μοναδική διαθέσιμη τεχνολογία για ασύρματες επικοινωνίες. Δύο ακόμη σημαντικά πρότυπα, που ανήκουν επίσης στη σειρά 802.x του IEEE, είναι το ZigBee και το WiFi. Το ZigBee χρησιμοποιείται για την επικοινωνία συσκευών που χαρακτηρίζονται από χαμηλή κατανάλωση ισχύος, μεγάλη διάρκεια ζωής μπαταρίας και χαμηλούς ρυθμούς μετάδοσης.

Παραδείγματα εφαρμογών του ZigBee αποτελούν τα οικιακά και βιομηχανικά συστήματα ελέγχου, τα οποία παρακολουθούν την κατάσταση οικιακών συσκευών ή βιομηχανικού εξοπλισμού και τα χειρίζονται εξ αποστάσεως. Το Bluetooth

χαμηλής ενέργειας της έκδοσης 4.0 (Bluetooth Low Energy, BLE) έχει παρόμοια χαρακτηριστικά με το ZigBee ως προς τη χαμηλή κατανάλωση ισχύος, όμως δεν μπορεί να το συναγωνιστεί ως προς το μέγιστο εφικτό μέγεθος του δικτύου. Το γνωστό σε όλους WiFi αποτελεί κατεξοχήν πρότυπο για την πρόσβαση των συσκευών στο Internet. Σε σύγκριση με το Bluetooth και το ZigBee, το WiFi προσφέρει υψηλότερους ρυθμούς μετάδοσης και μεγαλύτερη εμβέλεια, όμως χαρακτηρίζεται από υψηλότερη κατανάλωση ισχύος. Στον Πινακα 5.3, συγκρίνονται τα βασικά χαρακτηριστικά των τριών προαναφερθέντων προτύπων.

Πίνακας 5.3 Σύγκριση Bluetooth με το ZigBee και το WiFi

	Bluetooth	**ZigBee**	**WiFi**
Προδιαγραφή IEEE	802.15.1	802.15.4	802.11 (a, b, g, n)
Συχνότητα (GHz)	2.4	0.868 (Ευρώπη) 0.915 (Β. Αμερική) 2.4 (Παγκοσμίως)	2.4 και 5
Μέγιστος ρυθμός μετάδοσης (Mbps)	1-50 ανάλογα με την έκδοση	0.25	11 (b) 54 (g) 600 (n)
Μέγιστη εμβέλεια επικοινωνίας (μέτρα)	10-200	10-100	100-250
Τυπική διάρκεια μπαταρίας	Ημέρες για το κλασσικό Bluetooth Μήνες έως χρόνια για την έκδοση 4.0	Μήνες έως χρόνια	Ώρες
Μέγεθος δικτύου	7 (απροσδιόριστο για νεότερες εκδόσεις)	>64000	255

ΚΕΦΑΛΑΙΟ 6^{ο} :ΣΥΜΠΛΗΡΩΜΑΤΙΚΑ ΣΤΟΙΧΕΙΑ

Στο κεφάλαιο αυτό, θα παρουσιαστούν τα συμπληρωματικά, αλλά όχι δευτερεύουσας σημασίας στοιχεία της κατασκευής μας, τα οποία ρύθμισαν τη λειτουργία ή συνέδεσαν τα στοιχεία που αναφέρθηκαν στα προηγούμενα κεφάλαια, ώστε το όχημά μας να εκτελέσει τις επιθυμητές κινήσεις.Πρόκειται, βέβαια, για τους αισθητήρες που χρησιμοποιήθηκαν στις ασκήσεις, τις μπαταρίες, τους διακόπτες,τα καλώδια,το breadboard,τις screw shields, καθώς και τη μονάδα Bluetooth HC-05 και την L293D H-bridge Motor Driver Shield για την πλακέτα Arduino.

6.1 Αισθητήρια

6.1.1 KY-033 Tracking sensor module

6.1.1.1 Γενικά στοιχεία

Το συγκεκριμένο αισθητήριο είναι ιδανικό για τον σχεδιασμό εφαρμογών σε ρομπότ με τροχούς και έχει δυνατότητα ρύθμισης της απόστασης αποφυγής εμποδίων. Είναι ένας αισθητήρας που παρακολουθεί μία γραμμή για να καθοδηγήσει το ρομπότ με ακρίβεια. Είναι προσαρμόσιμος στο φως, υψηλής ακρίβειας, και αποτελείται από ένα ζευγάρι υπέρυθρων πομπού και δέκτη.Ο αισθητήρας ανιχνεύει εάν μια περιοχή ανάκλασης ή απορρόφησης φωτός βρίσκεται μπροστά του και δείχνει ποια από τις δύο περιοχές θα μας δώσει ψηφιακή έξοδο. Η ευαισθησία (ελάχιστη εμβέλειά) του μπορεί να ρυθμιστεί από τον ελεγκτή.

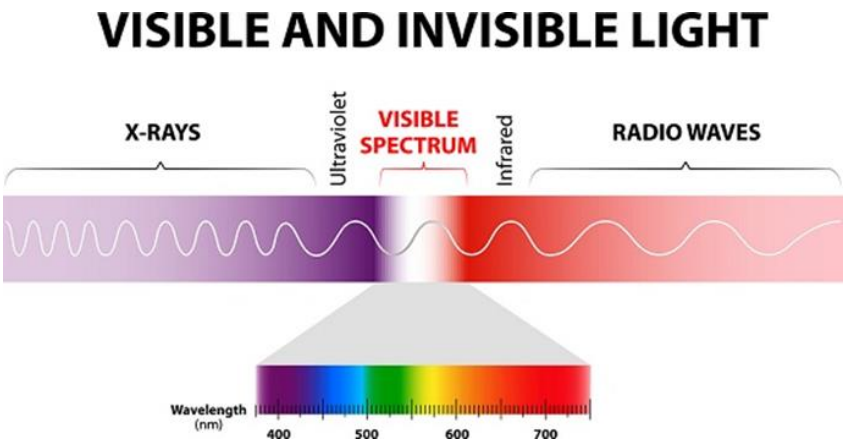
Εικόνα 6.1 Αισθητήριο KY-033 Tracking sensor module

Η διάταξη του πομπού εκπέμπει μια ορισμένη συχνότητα υπέρυθρης ακτινοβολίας και, όταν ανιχνεύει την κατεύθυνση ενός εμποδίου, η διάταξη του υπέρυθρου του δέκτη την ανακλά πίσω. Όταν η ενδεικτική λυχνία ανάβει μέσω του κυ-

κλώματος, η διεπαφή εξόδου σήματος βγάζει το ψηφιακό σήμα, το οποίο μπορεί να ανιχνευθεί από το ποτενσιόμετρο και να ρυθμιστεί η απόσταση (από 2 ~ 40cm). Αυτή η συμπεριφορά μπορεί να χρησιμοποιηθεί, ώστε το ρομπότ στο οποίο έχει τοποθετηθεί το συγκεκριμένο αισθητήριο,να μπορεί να ακολουθήσει για αυτόματα μία γραμμή.

6.1.1.2 Υπέρυθρη ακτινοβολία

Η υπέρυθρη ακτινοβολία είναι ένας τύπος ηλεκτρομαγνητικής ακτινοβολίας. Το υπέρυθρο (IR) φως είναι το μέρος του ηλεκτρομαγνητικού φάσματος που οι άνθρωποι συναντούν περισσότερο στην καθημερινή ζωή, αν και μεγάλο μέρος του περνά απαρατήρητο. Είναι αόρατο στα ανθρώπινα μάτια, αλλά οι άνθρωποι μπορούν να το αισθανθούν σαν θερμότητα.

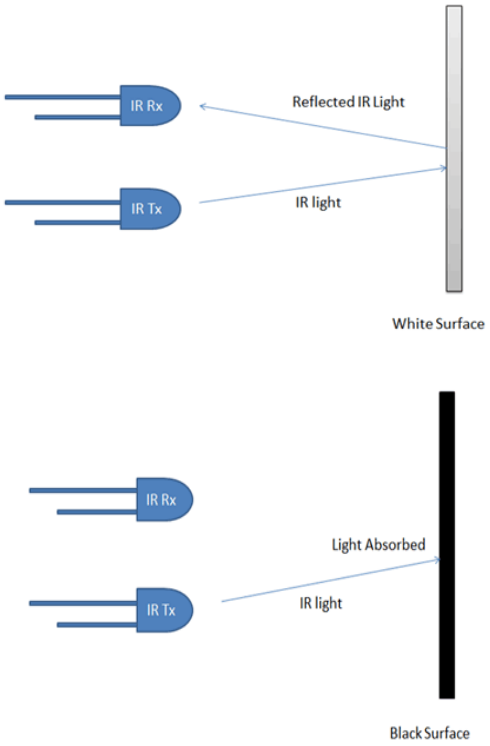


Εικόνα 6.2 Ηλεκτρομαγνητικό φάσμα

6.1.1.3 Τρόπος λειτουργίας

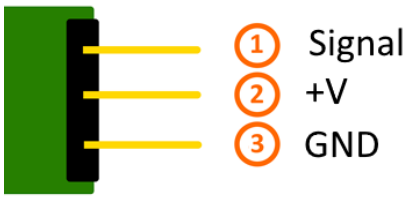
Η έννοια της εργασίας του ακόλουθου γραμμής σχετίζεται με το φως. Χρησιμοποιούμε εδώ τη συμπεριφορά του φωτός σε ασπρόμαυρη επιφάνεια.

Το αισθητήριο αυτό χρησιμοποιεί πομπούς και δέκτες IR(infrared), που ονομάζονται επίσης φωτοδίοδοι και χρησιμοποιούνται για την αποστολή και λήψη φωτός. Όταν οι ακτίνες υπέρυθρης ακτινοβολίας πέφτουν σε λευκή επιφάνεια, ανακλώνται πίσω και συλλαμβάνονται από φωτοδιόδους που δημιουργούν κάποιες αλλαγές τάσης. Όταν το φως IR πέφτει σε μια μαύρη επιφάνεια, το φως απορροφάται από αυτήν και οι ακτίνες δεν ανακλώνται πίσω, οπότε η φωτοδίοδος δεν δέχεται φως ή ακτίνες.



Εικόνα 6.3 Λειτουργία του φωτός σε άσπρη και μαύρη επιφάνεια

Όταν ο υπέρυθρος πομπός εκπέμπει ακτίνες σε ένα κομμάτι χαρτί, αν οι ακτίνες πέσουν σε μια λευκή επιφάνεια, θα ανακλαστούν και θα ληφθούν από τον δέκτη και ο ακροδέκτης S θα έλθει σε τιμή HIGH. Αν οι ακτίνες συναντήσουν μαύρες γραμμές, θα απορροφηθούν κι έτσι ο δέκτης δεν παίρνει τίποτα και ο ακροδέκτης S θα έλθει σε τιμή LOW.



Εικόνα 6.4 Pinout αισθητηρίου

6.1.1.4 Χαρακτηριστικά

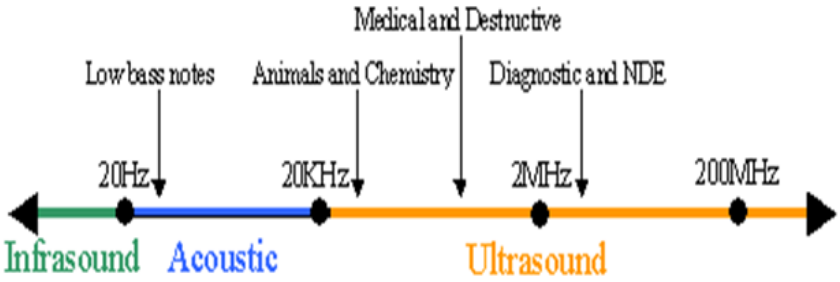
1. Τάση λειτουργίας: DC 3.3V-5V
2. Ρεύμα λειτουργίας: ≥ 20mA

3. Θερμοκρασία λειτουργίας: -10 ℃ ~ + 50 ℃
4. Απόσταση ανίχνευσης: 2-40cm
5. Διεπαφή IO: Διεπαφές 4 καλωδίων (- / + / S / EN)
6. Σήμα εξόδου: Επίπεδο TTL (χαμηλό επίπεδο, υπάρχει εμπόδιο, υψηλό επίπεδο, δεν υπάρχει εμπόδιο)
7. Ρύθμιση: προσαρμόζουμε την αντίσταση πολλαπλών στροφών
8. Αποτελεσματική γωνία: 35 °
9. Μέγεθος: 28mm × 23mm

6.1.2 Αισθητήρας υπερήχων HC-SR 04

6.1.2.1 Γενικά στοιχεία

Όπως φαίνεται και στην παρακάτω εικόνα (6.5), οι υπέρηχοι βρίσκονται πάνω απ' τις ακουστικές συχνότητες, έτσι δεν μπορεί να τους ακούσει το ανθρώπινο αυτί. Παρόλο, πάντως, που εμείς δεν τους ακούμε, κάποια ζώα μπορούν και να τους ακούν, αλλά και να τους χρησιμοποιούν. Χαρακτηριστικά παραδείγματα που μας το δείχνουν αυτό είναι η κίνηση των νυχτερίδων και η σφυρίχτρα που χρησιμοποιείται για τους σκύλους. Αισθητήρες υπερήχων συναντάμε σε πολλές εφαρμογές στην ιατρική, στην πλοήγηση σκαφών/πλοίων ακόμα και στα αυτοκίνητά μας, στα γνωστά park sensors.



Εικόνα 6.5 Ακουστικό φάσμα

Οι αισθητήρες υπερήχων λειτουργούν με την ίδια αρχή που λειτουργούν τα ραντάρ και τα σόναρ. Εκτιμούν την απόσταση ενός στόχου, λαμβάνοντας υπόψη τους την αντανάκλαση ενός ραδιοκύματος ή ενός ηχητικού σήματος πάνω στον στόχο .

Έτσι, δημιουργούν υψηλής συχνότητας κύματα και, χρησιμοποιώντας το επιστρεφόμενο σήμα, καθορίζουν την απόσταση ή ακόμα και την ταχύτητα του στό-

χου. Για να το επιτύχουν αυτό, χρησιμοποιούν τον χρόνο που έκανε το σήμα για να καλύψει την απόσταση από τον αισθητήρα στο αντικείμενο και πίσω.

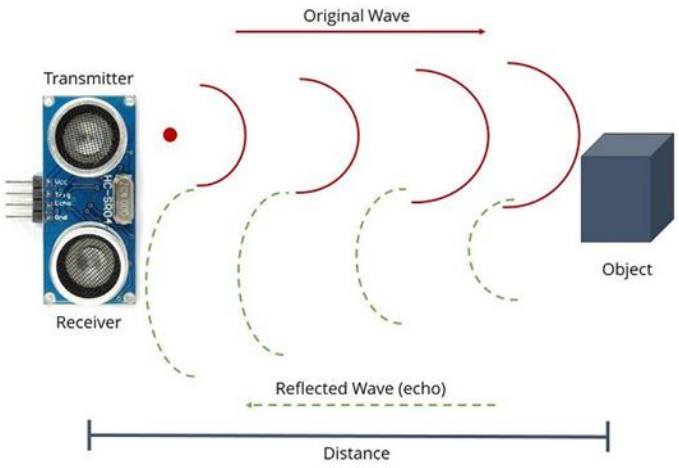
6.1.2.2 Τρόπος λειτουργίας

Στο Arduino, η διαδικασία αυτή (αντιστοίχιση χρόνου σε απόσταση) γίνεται με το αισθητήριο υπερήχων HC-SR04. Προσφέρει εξαιρετική ανίχνευση εύρους (από 2cm έως 400 cm) χωρίς επαφή με υψηλή ακρίβεια και σταθερές ενδείξεις σε μια εύκολη στη χρήση συσκευασία.Η λειτουργία του δεν επηρεάζεται από την ηλιακή ακτινοβολία ή το μαύρο υλικό,όπως τα αιχμηρά φωτοκύτταρα (αν και ακουστικά μαλακά υλικά, όπως το ύφασμα, μπορεί να είναι δύσκολο να ανιχνευθούν**).**

Το HC-SR04 έχει 4 ακροδέκτες, Ground, VCC, Trig και Echo**.** Οι ακροδέκτες Ground και VCC της μονάδας πρέπει να συνδέονται με τη γείωση και τους ακροδέκτες των 5V στον πίνακα Arduino αντιστοίχως και οι ακροδέκτες Trig και Echo με οποιοδήποτε ακροδέκτη ψηφιακής εισόδου / εξόδου στην πλακέτα Arduino.

Εικόνα 6.6 Αισθητήριο υπερήχων HC-SR04 (διακρίνονται οι 4 ακροδέκτες)

Για να ξεκινήσει να μετρά το αισθητήριο, το pin εισόδου Trig του SR04 πρέπει να λάβει έναν θετικό παλμό (5V) για τουλάχιστον 10μs. Τότε, αυτό θα εκκινήσει τον αισθητήρα.Στη συνέχεια, ο πομπός (trig pin) στέλνει ένα σήμα (ήχο) υψηλής συχνότητας μέχρι να βρει ένα αντικείμενο.Όταν συμβεί αυτό, το σήμα ανακλάται και ο δέκτης (echo pin) το λαμβάνει.Συγκεκριμένα ο δέκτης λαμβάνει τον χρόνο που διάνυσε το κύμα ήχου σε μικροδευτερόλεπτα.



Εικόνα 6.7 Σχηματική παρουσίαση της λειτουργίας του HC-SR04

Επειδή η ταχύτητα **v** του ήχου στον αέρα είναι γνωστή, μπορούμε να υπολογίσουμε την απόσταση **d** ενός αντικειμένου από τον αισθητήρα, από τον χρόνο **t** που ένα ηχητικό σήμα, το οποίο εκπέμπει ο αισθητήρας, κάνει να επιστρέψει στον αισθητήρα: **d = v · t / 2.**

Για παράδειγμα, εάν το αντικείμενο απέχει 10 cm από τον αισθητήρα και η ταχύτητα του ήχου είναι 340 m / s ή 0,034 cm / μs, το ηχητικό κύμα πρέπει να ταξιδέψει περίπου 294 μs. Αλλά η τιμή που θα πάρουμε από τον ακροδέκτη Echo θα είναι διπλάσια αυτού του αριθμού, επειδή το ηχητικό κύμα πρέπει να ταξιδέψει προς τα εμπρός και να επιστρέψει προς τα πίσω. Επομένως, για να έχουμε την απόσταση σε cm, πρέπει να πολλαπλασιάσουμε την τιμή του λαμβανόμενου χρόνου ταξιδιού από τον ακροδέκτη Echo κατά 0.034 και να διαιρέσουμε το αποτέλεσμα με το 2.

6.1.2.3 Χαρακτηριστικά

Τροφοδοσία: + 5V DC

Πραγματικό ρεύμα: <2mA

Ρεύμα λειτουργίας: 15mA

Εφαρμοσμένη γωνία: <15 °

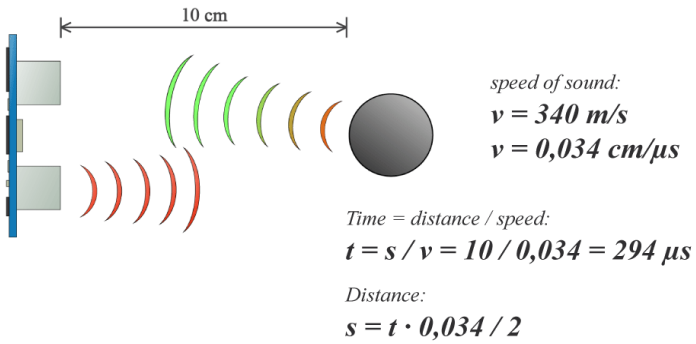
Διαστάσεις: 2cm - 400cm / 1 "- 13ft

Ανάλυση: 0,3 cm

Γωνία μέτρησης: 30 μοίρες

Είσοδος ενεργοποίησης Πλάτος παλμού: 10uS

Διάσταση: 45mm x 20mm x 15mm



Εικόνα 6.8 Υπολογισμός καθυστέρησης και απόστασης

6.1.3 Αισθητήρας MPU-9250/6500

Είναι μια συσκευή MotionTracking 9 αξόνων, που συνδυάζει: Ένα γυροσκόπιο 3 αξόνων, ένα Accelerometer (επιταχυνσιόμετρο) 3 αξόνων, ένα μαγνητόμετρο 3 αξόνων και ένα ψηφιακό Motion Processor ™ (DMP), όλα σε ένα μικρό τσίπ πακέτο 3x3x1mm που διατίθεται ως αναβάθμιση συμβατή με pin από το MPU-6515.

Με τον αποκλειστικό δίαυλο αισθητήρα I^2C, ο MPU-9250/6500 παρέχει απευθείας έξοδο MotionFusion ™ με 9 άξονες. Η συσκευή MPU-9250 MotionTracking, με την ενσωμάτωσή της σε 9 άξονες, το MotionFusion ™ on-chip και το υλικολογισμικό (firmware) βαθμονόμησης χρόνου εκτέλεσης σχεδιάστηκε επίσης για να διασυνδέεται με πολλαπλούς μη αδρανειακούς ψηφιακούς αισθητήρες, όπως αισθητήρες πίεσης, στη βοηθητική θύρα I^2C.

6.1.3.1 Γυροσκόπιο 3 αξόνων

Είναι μια συσκευή, η οποία μπορεί να διατηρεί σταθερό τον προσανατολισμό της μέσω της περιστροφής των μερών της και της αρχής της διατήρησης της στροφορμής.

Το κυριότερο εξάρτημα του γυροσκοπίου αποτελεί μια περιστρεφόμενη,επίπεδη μάζα, που περιστρέφεται γρήγορα γύρω από έναν κατακόρυφο ως προς αυτήν άξονα.

Τα γυροσκόπια λειτουργούν με βάση δύο αρχές:α) την αρχή της διατήρησης της στροφορμής, σύμφωνα με την οποία ένα σώμα που περιστρέφεται αντιστέκεται στις δυνάμεις που προσπαθούν να αλλάξουν την κατεύθυνση του άξονα περιστροφής του και β) την αρχή της μετάπτωσης, σύμφωνα με την οποία ένα πε-

ριστρεφόμενο γυροσκόπιο με ελευθερία μετακίνησης κινείται κάθετα ως προς τις εξωτερικές δυνάμεις που του ασκούνται.

Εικόνα 6.9 Αισθητήριο MPU-9250/6500

6.1.3.2 Επιταχυνσιόμετρο (Accelerometer) τριών αξόνων

Είναι μια ηλεκτρονική συσκευή που έχει την ικανότητα να μετράει δυνάμεις επιτάχυνσης. Τέτοιες δυνάμεις μπορεί να είναι στατικές, όπως είναι η δύναμη της βαρύτητας ή δυναμικές, που είναι ευρέως γνωστές από τις κινητές συσκευές για τη μέτρηση δονήσεων και κινήσεων. Με άλλα λόγια, το επιταχυνσιόμετρο είναι μία συσκευή που μετράει επιταχύνσεις.

Η επιτάχυνση ορίζεται ως η μεταβολή της ταχύτητας ως προς τον χρόνο Επίσης, το επιταχυνσιόμετρο μετράει και την επιτάχυνση που συνδέεται με τις δυνάμεις βάρους. Για παράδειγμα, ένα επιταχυνσιόμετρο σε ηρεμία στην επιφάνεια της γης θα μετρήσει επιτάχυνση g = 9.81m/s λόγω του βάρους του. Αντιθέτως, ένα επιταχυνσιόμετρο, το οποίο βρίσκεται είτε σε ελεύθερη πτώση είτε σε ηρεμία στο διάστημα, θα μετρήσει τιμή ίση με το μηδέν.

6.1.3.3 Μαγνητόμετρο τριών αξόνων

Γενικά, με τον όρο «μαγνητόμετρο», ονομάζεται κάθε κατάλληλο όργανο για τη μέτρηση στοιχείων του γήινου μαγνητισμού.

Τα μαγνητόμετρα, κατάλληλα τροποποιούμενα, μπορούν να χρησιμοποιηθούν και ως βαριόμετρα, δηλαδή όχι μόνο για τις στιγμιαίες τιμές των τριών μεγεθών του γήινου μαγνητισμού, αλλά και τις μεταβολές τους.

PINOUT του Αισθητήρα MPU-9250/6500

- **VCC** : 3.6 έως 6V τροφοδοσία (ενσωματωμένος ρυθμιστής 3.3V)
- **GND** :0V
- **SCL** : σειριακό ρολόι I^2C
- **SDA** : σειριακά δεδομένα I^2C
- **EDA**: Βοηθητικά σειριακά δεδομένα κύριας μονάδας I^2C
- **ECL**: Βοηθητικό σειριακό ρολόι I^2C Master
- **AD0** : Διεύθυνση Slave I^2C LSB (AD0)
- **INT** : Διακοπή ψηφιακής εξόδου
- **NCS** : Επιλογή τσιπ (μόνο λειτουργία SPI)
- **FSYNC**:Ψηφιακή είσοδος συγχρονισμού πλαισίου. Μπορούμε να συνδεθούμε στο GND αν δεν χρησιμοποιηθεί.

6.1.3.4 Γενικά χαρακτηριστικά

- Τρεις μετατροπείς αναλογικού σε ψηφιακό σήμα (ADC) 16 ψηφίων για την ψηφιοποίηση των εξόδων γυροσκοπίου
- Τρεις ADC 16 bits για ψηφιοποίηση των εξόδων επιταχυνσιόμετρου
- Τρεις ADC 16 bits για ψηφιοποίηση των εξόδων μαγνητόμετρου
- Εύρος πλήρους κλίμακας για κλίμακα Gyroscope ± 250, ± 500, ± 1000 και ± 2000 ° / sec (dps)
- Επιτάχυνση πλήρους κλίμακας ± 2g, ± 4g, ± 8g και ± 16g.
- Μαγνητόμετρο πλήρους κλίμακας ± 4800μT
- I^2C και SPI σειριακές διασυνδέσεις
- Τροφοδοσία: 3-5v

6.1.3.5 Χαρακτηριστικά γυροσκοπίου

- Ψηφιακοί αισθητήρες γωνιακής ταχύτητας X-, Y- και Z-Axis (γυροσκόπια) με εύρος πλήρους κλίμακας προγραμματιζόμενου από τον χρήστη ± 250, ± 500, ± 1000 και ± 2000 ° / sec και ενσωματωμένους ADC 16 bits
- Ψηφιακό προγραμματιζόμενο φίλτρο χαμηλής διέλευσης
- Ρεύμα λειτουργίας με γυροσκόπιο: 3,2 mA
- Ρεύμα κατάστασης αναστολής λειτουργίας: 8μA
- Εργοστασιακός βαθμονομημένος συντελεστής κλίμακας ευαισθησίας

- Τεστ αυτοαξιολόγησης

6.1.3.6 Χαρακτηριστικά επιταχυνσιόμετρου

- Επιταχυνσιόμετρο τριών αξόνων ψηφιακής εξόδου με προγραμματιζόμενη κλίμακα πλήρους κλίμακας ± 2g, ± 4g, ± 8g και ± 16g και ολοκληρωμένα ADC 16 bits
- Κανονικό ρεύμα λειτουργίας επιταχυνσιόμετρο: 450 mA
- Ρεύμα λειτουργίας επιταχυνσιόμετρου χαμηλής ισχύος: 8.4 μA στα 0.98Hz, 19.8 μA στα 31.25Hz
- Ρεύμα κατάστασης αναστολής λειτουργίας: 8 μA
- Προγραμματισμένες από τον χρήστη διακοπές
- Διακοπή αφύπνισης κατά την κίνηση για λειτουργία χαμηλής ισχύος του επεξεργαστή εφαρμογών
- Αυτόματη δοκιμή

6.1.3.7 Χαρακτηριστικά μαγνητόμετρου

- 3-αξονικός μονολιθικός μαγνητικός αισθητήρας Hall πυριτίου με μαγνητικό συγκεντρωτή
- Ευρεία περιοχή δυναμικών μετρήσεων και υψηλή ανάλυση με χαμηλότερη κατανάλωση ρεύματος
- Ανάλυση δεδομένων εξόδου 14 bits (0.6 μT / LSB) ή 16 bits (15 μT / LSB).
- Η κλίμακα μέτρησης πλήρους κλίμακας είναι ± 4800 μT
- Κανονικό ρεύμα λειτουργίας μαγνητόμετρου: 280 μA σε ρυθμό επανάληψης 8Hz
- Λειτουργία αυτοελέγχου με εσωτερική μαγνητική πηγή για επιβεβαίωση της λειτουργίας μαγνητικού αισθητήρα στα τελικά προϊόντα

6.1.3.8 Επιπρόσθετα χαρακτηριστικά

- Βοηθητικός δίαυλος I^2C master για την ανάγνωση δεδομένων από εξωτερικούς αισθητήρες (π.χ. αισθητήριο πίεσης)
- Ρεύμα λειτουργίας 3,5 m A όταν είναι ενεργοποιημένοι και οι 9 άξονες ανίχνευσης κίνησης και το DMP
- Εύρος τάσης τροφοδοσίας V_{DD} (2,4 - 3,6V)

- Τάση αναφοράς V_{DD} I/O για βοηθητικές συσκευές I^2C
- Μικρότερη και λεπτότερη συσκευασία QFN για φορητές συσκευές: 3x3x1mm
- Ελάχιστη ευαισθησία σε εγκάρσιο άξονα μεταξύ των αξόνων επιταχυνσιομέτρου, γυροσκοπίου και μαγνητομέτρου
- 512 bytes buffer FIFO επιτρέπει στον επεξεργαστή εφαρμογών να διαβάζει τα δεδομένα σε εκρήξεις.
- Αισθητήρας θερμοκρασίας ψηφιακής εξόδου
- Ψηφιακά φίλτρα προγραμματιζόμενα από τον χρήστη για γυροσκόπιο, επιταχυνσιόμετρο και αισθητήρα θερμοκρασίας
- 10.000 g αντοχή στον κλονισμό
- 400 kHz Fast Mode I^2C για επικοινωνία με όλους τους καταχωρητές
- 1MHz σειριακή διεπαφή SPI για επικοινωνία με όλους τους καταχωρητές

6.2 Πηγή τροφοδοσίας

Για τις εφαρμογές μας, θα χρειαστούμε ισχύ μεγαλύτερη από αυτή που μπορεί να μας δώσουν οι 4ΧΑΑ μπαταρίες που έχουν προβλεφτεί από τον κατασκευαστή του οχήματος. Γι'αυτό, θα χρησιμοποιηθεί μια μπαταρία επαναφορτιζόμενη μολύβδου, που μας παρέχει την απαιτούμενη ισχύ με χαμηλό κόστος.

6.2.1 Γενικά στοιχεία για τις μπαταρίες

Ηλεκτρικό στοιχείο ή μπαταρία ή ηλεκτρικός συσσωρευτής ή απλά συσσωρευτής, είναι η συσκευή, η οποία αποθηκεύει χημική ενέργεια και την αποδίδει σε ηλεκτρική ενέργεια. Η βασική μονάδα αποθήκευσης της ηλεκτρικής ενέργειας είναι το στοιχείο. Τα στοιχεία τα συναντάμε ως πρωτογενή (μιας χρήσης) ή ως δευτερογενή (επαναφορτιζόμενα). Η ανάπτυξη των στοιχείων-μπαταριών άρχισε με την κατασκευή της Βολταϊκής στήλης από τον Αλεσάντρο Βόλτα (Alesandro Volta).

6.2.2 Είδη μπαταριών

6.2.2.1 Πρωτογενείς μπαταρίες

Οι πρωτογενείς μπαταρίες μπορούν να χρησιμοποιηθούν στις ηλεκτρικές συσκευές, σε φωτογραφικό εξοπλισμό, στα ρολόγια, στους υπολογιστές και σε πολ-

λές άλλες χρήσεις της καθημερινής μας ζωής.Οι περισσότερες πρωτογενείς μπαταρίες είναι κυλινδρικές, επίπεδες ή κομβιόσχημες (κουμπιά) με χωρητικότητα κάτω από 20m A. Συνήθως είναι οικιακής χρήσης, σε αντίθεση με τις δευτερογενείς, που είναι συνήθως βιομηχανικής χρήσης. Οι κυριότεροι τύποι πρωτογενών μπαταριών είναι οι:

- **Ψευδαργύρου / 'Ανθρακα (Zn/C)**: οι γνωστές σε όλους απλές μπαταρίες για τις απλούστερες χρήσεις και με τη μικρότερη διάρκεια ζωής.
- **Ψευδαργύρου / Χλωριδίου (Zn/Cl):** με λίγο μεγαλύτερη διάρκεια ζωής. Χρησιμοποιούνται εκεί όπου υπάρχουν μεγαλύτερες απαιτήσεις σε ενέργεια.
- **Αλκαλικές Μαγγανίου**: με μεγαλύτερη διάρκεια ζωής από τα δύο προηγούμενα είδη. Είναι μάλιστα και φιλικότερες προς το περιβάλλον.
- **Αργύρου***:* συνήθως κομβιόσχημες, περιέχουν οξείδιο του αργύρου, και χρησιμοποιούνται κυρίως σε ρολόγια.
- **Λιθίου***:* μεγάλης διάρκειας ζωής, περιέχουν μεταλλικό λίθιο και χρησιμοποιούνται ευρέως στον φωτογραφικό εξοπλισμό και στα κινητά τηλέφωνα.
- **Ψευδαργύρου - αέρα***:* επίσης κομβιόσχημες, έχουν την καινοτομία ότι αντί θετικού πόλου, χρησιμοποιείται το ατμοσφαιρικό οξυγόνο.
- **Υδραργύρου***:* με οξείδιο του υδραργύρου, χρησιμοποιούνται κυρίως σε ιατρικές συσκευές, όπως ακουστικά βαρηκοΐας. Δυστυχώς, ο υδράργυρος που περιέχουν είναι επικίνδυνος για το περιβάλλον.

6.2.2.2 Δευτερογενείς μπαταρίες

Οι δευτερογενείς μπαταρίες επαναφορτίζονται ηλεκτρικά και μπορούν να χρησιμοποιηθούν σχεδόν παντού. Χωρίζονται σε τρία βασικά συστήματα:

- **Επαναφορτιζόμενο σύστημα νικελίου - καδμίου (Ni-Cd)***:* Οι πρώτες επαναφορτιζόμενες μπαταρίες που φτιάχτηκαν ποτέ. Χρησιμοποιούνται σε ηλεκτρικά εργαλεία, φορητά τηλέφωνα, φορητούς υπολογιστές, παιχνίδια, κ.λ.π., με διάρκεια ζωής 4-5 χρόνια. Δυστυχώς, το κάδμιο είναι βλαβερό. Έτσι, γίνονται προσπάθειες να απομακρυνθεί αυτό το είδος μπαταρίας από την αγορά, και, όπου είναι δυνατόν, να αντικατασταθεί.
- **Επαναφορτιζόμενο σύστημα μολύβδου (Pb):** Η ανακάλυψή τους έφερε την επανάσταση στην αυτοκινητοβιομηχανία, αφού οι περισσότερες μπαταρίες αυτοκινήτων ανήκουν σε αυτήν την κατηγορία. Δυστυχώς, ο μόλυβ-

δος είναι και αυτός επικίνδυνος για το περιβάλλον, γι αυτό γίνεται ήδη προσπάθεια να συλλέγονται οι άδειες μπαταρίες από τα συνεργεία αυτοκινήτων και να στέλνονται για ανακύκλωση.

- **Σύστημα νικελίου - μετάλλου υδριδίου (NiMH)***:* Φιλικότερες προς το περιβάλλον από τις Ni-Cd, τις οποίες τείνουν να αντικαταστήσουν και με μεγαλύτερη διάρκεια ζωής.

6.2.3 Μπαταρία μολύβδου

Η μπαταρία μας αποτελείται από τρία (3) στοιχεία των 2,1V, συνδεδεμένα σε σειρά, έτσι ώστε στους ακροδέκτες της να έχει διαφορά δυναμικού 6,3V. Η πραγματική τάση της μπαταρίας δεν είναι πάντα η ονομαστική των 6V. Κυμαίνεται από 6,75V αμέσως μετά από μία πλήρη φόρτιση μέχρι τα 4,8V αν είναι τελείως αφόρτιστη.

Στις μπαταρίες μολύβδου, χρησιμοποιούνται πλάκες κράματος μολύβδου – αντιμονίου καλυμμένες με πάστα οξειδίου του μολύβδου.Μετά από τη φόρτιση, έχουμε μεταβολή του οξειδίου του μολύβδου σε διοξείδιο του μολύβδου στις θετικές πλάκες και σε πορώδη μόλυβδο στις αρνητικές. Έτσι, όταν η μπαταρία είναι σε πλήρη φόρτιση, έχουμε δύο διαφορετικά υλικά (διοξείδιο του μολύβδου και πορώδη μόλυβδο) βυθισμένα σε αραιό θειικό οξύ. Όσο η μπαταρία απεκφορτίζεται, τότε οι επιφάνειες των πλακών μετατρέπονται σε θειικό μόλυβδο.

Εικόνα 6.10 Μπαταρία μολύβδου

Η μπαταρία έχει σταθερή πολικότητα και, για να την φορτίσουμε, χρειάζεται ρεύμα σταθερής πολικότητας. Μπορεί να μην είναι συνεχές αλλά διακοπτόμενο μέσης τιμής I_{dc}, ποτέ όμως εναλλασσόμενο. Αν η ηλεκτρεγερτική δύναμη (δηλαδή η τάση χωρίς φορτίο) της μπαταρίας είναι 12V, τότε, για να την φορτίσουμε, χρειαζόμαστε τάση της τάξεως των 14 – 16V ανάλογα με τον ρυθμό φορτίσεως που θέλουμε, αλλά και με την εσωτερική αντίσταση της μπαταρίας (δηλαδή η τάση στα άκρα του φορτιστή πρέπει να ξεπερνά την ηλεκτρεγερτική δύναμη της μπαταρίας, αλλά και

την πτώση τάσεως στο εσωτερικό της μπαταρίας λόγω της εσωτερικής της αντιστάσεως).

ΣΗΜ.Η μπαταρία χρειάζεται επαναφόρτιση ανά ατακτά χρονικά διαστήματα με κατάλληλο φορτιστή.

6.3 Διακόπτες

Οι διακόπτες είναι εξαρτήματα που χρησιμοποιούνται σε ηλεκτρικά και ηλεκτρονικά κυκλώματα και μας επιτρέπουν να ελέγχουμε τον ηλεκτρισμό. Αν και τους θεωρούμε σαν απλές διατάξεις για να θέτουν συσκευές ON-OFF,στην πράξη είναι πολύ περισσότερο από αυτό. Οι διακόπτες είναι διατάξεις για το κλείσιμο, το άνοιγμα και την αλλαγή συνδέσεων σε ένα ηλεκτρικό κύκλωμα.

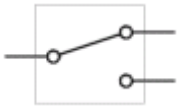
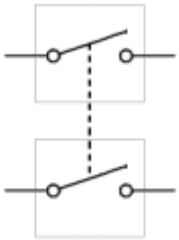
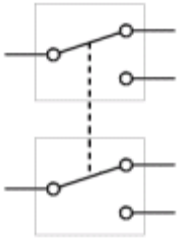
Οι διακόπτες κατηγοριοποιούνται με τον αριθμό των πόλων και των διαδρομών τους ως εξής:

- **Πόλοι** (**Poles**). Ένας πόλος είναι ένα σετ επαφών που ανήκουν σε ένα κύκλωμα.
- **Θέσεις** (**Throws**). Αντιστοιχούν σε αυτό που καταλαβαίνουμε σαν θέση ενός μηχανικού διακόπτη (πχ πατημένος και ελεύθερος).

Στις εφαρμογές μας θα χρησιμοποιήσουμε έναν διακόπτη SPDT (Single-Pole, Double-Throw) :

Εικόνα 6.11. Διακόπτης SPDT

Πίνακας 6.1 Κατηγορίες διακοπτών με βάση τους πόλους και τις θέσεις

SPST (Single-Pole, Single-Throw)	
	1 πόλος και 1 θέση. Πρόκειται για τον κλασικό διακόπτη on-off.
SPDT (Single-Pole, Double-Throw)	
	1 πόλος και 2 θέσεις. Είναι η απλούστερη μορφή μεταγωγικού διακόπτη. Με μία είσοδο μπορούμε να επιλέξουμε ανάμεσα σε δύο διαφορετικές εξόδους (ή το ανάποδο). Εάν ο διακόπτης διαθέτει και τρίτη ενδιάμεση θέση off (δεν κλείνει κύκλωμα μεταξύ επαφών) περιγράφεται σαν SPCO (Single-Pole, Changeover ή Single-Pole, Centre-Off) και SPTT (Single-Pole, Triple-Throw).
DPST (Double-Pole, Single-Throw)	
	2 πόλοι και 1 θέση. Πρόκειται για 2 διακόπτες τύπου SPST που είναι συζευγμένοι (ενεργοποιούνται ταυτόχρονα).
DPDT (Double-Pole, Double-Throw)	
	2 πόλοι και 2 θέσεις. Πρόκειται για 2 διακόπτες τύπου SPDT που είναι συζευγμένοι (ενεργοποιούνται ταυτόχρονα). Εάν ο διακόπτης διαθέτει και τρίτη, ενδιάμεση, θέση off (δεν κλείνει κύκλωμα μεταξύ επαφών) περιγράφεται σαν DPCO (Double-Pole, Changeover ή Double-Pole, Centre-Off).

6.4 Μονάδα Bluetooth HC-05

Η μονάδα HC-05 είναι μια εύκολη στη χρήση μονάδα Bluetooth SPP (Serial Port Protocol), σχεδιασμένη για διαφανή ρύθμιση ασύρματης σειριακής σύνδεσης. Η

μονάδα Bluetooth HC-05 μπορεί να χρησιμοποιηθεί σε διαμόρφωση Master ή Slave, καθιστώντας την εξαιρετική λύση για ασύρματη επικοινωνία**.**

Η εργοστασιακή ρύθμιση είναι SLAVE. Ο ρόλος της μονάδας (Master ή Slave) μπορεί να ρυθμιστεί μόνο από τις εντολές AT COMMANDS. Οι Slave μονάδες δεν μπορούν να πραγματοποιήσουν σύνδεση με άλλη συσκευή Bluetooth, αλλά μπορούν να δεχθούν συνδέσεις. Η μονάδα Master μπορεί να εκκινήσει μια σύνδεση σε άλλες συσκευές.

PINOUT

KEY/ENABLE: Αν η τιμή είναι HIGH πριν από την εφαρμογή της τροφοδοσίας,τότε επιβάλλεται η λειτουργία ρύθμισης της εντολής AT (AT Command Setup Mode).

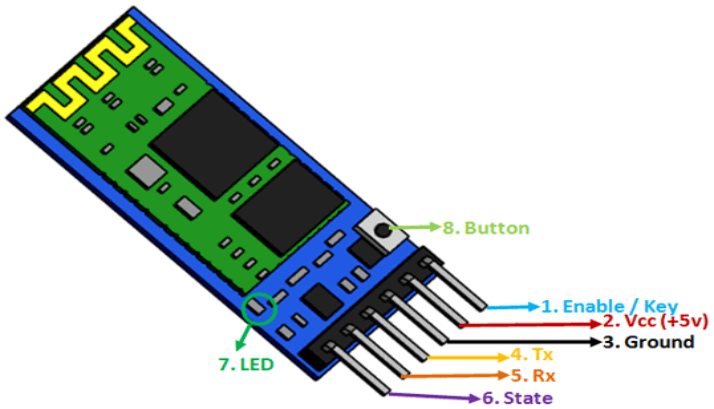
VCC: Ενδείκνυται στην περιοχή των 3.6V-6V.

GND: Γείωση

TXD: Σειριακή έξοδος της μονάδας, για σύνδεση με το RX του μικροελεγκτή. Αυτό το σήμα χρησιμοποιεί επίπεδο λογικής 3.3V.

RXD: Σειριακή είσοδος της μονάδας, που θα συνδεθεί στο TX του μικροελεγκτή. Αυτό το σήμα χρησιμοποιεί λογικά επίπεδα 3.3V.

STATE: Συνδεμένο με το LED2 (Pin32) της μονάδας.



Εικόνα 6.12 HC-05 - Bluetooth Module

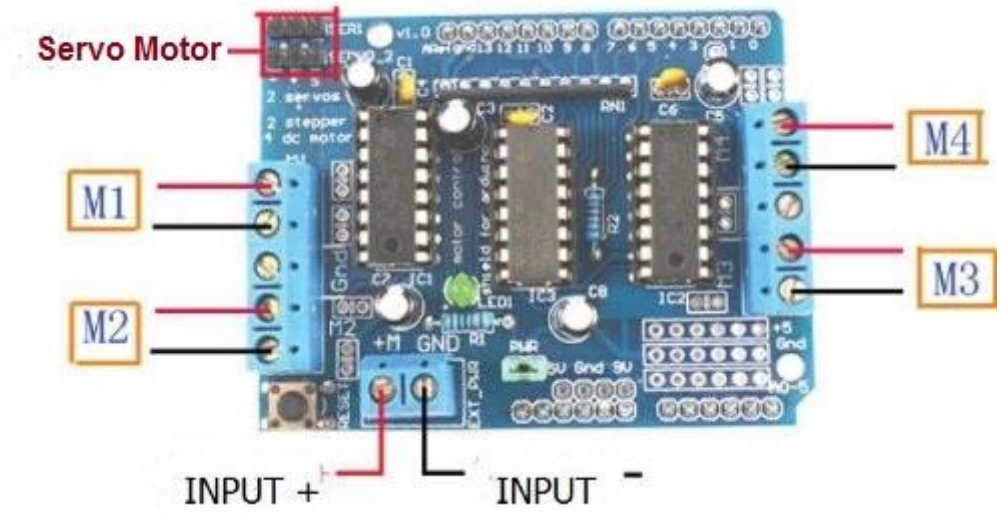
6.5 L293D H-bridge Motor Driver Shield

Μια Motor Driver Shield για πλακέτες Arduino μπορεί να ελέγξει έως και τέσσερις διπλής κατεύθυνσης κινητήρες συνεχούς ρεύματος με μεμονωμένη επιλογή ταχύτητας οκτώ δυαδικών ψηφίων ή δύο βηματικούς κινητήρες (μονοπολικούς ή

διπολικούς) με μονό πηνίο, διπλό πηνίο, παρεμβαλλόμενο ή μικροδιακόπτη. Επίσης, μπορεί να συνδέσει δύο σερβοκινητήρες 5V συνδεδεμένους με τον ειδικό χρονιστή υψηλής ανάλυσης του Arduino .

Η ασπίδα (shield) περιέχει δύο οδηγούς κινητήρα L293D και έναν καταχωρητή μετατόπισης 74HC595. Ο καταχωρητής μετατόπισης διευρύνει τους τρεις ακροδέκτες του Arduino σε οκτώ pins, για να ελέγξει την κατεύθυνση των οδηγών κινητήρα. Η δυνατότητα εξόδου του L293D συνδέεται άμεσα με τις εξόδους PWM του Arduino.

Για να αυξήσει το μέγιστο ρεύμα, το L293D επιτρέπει επιπλέον οδηγούς με "piggyback". Το piggyback συγκολλάει έναν ή δύο ή τρεις επιπλέον οδηγούς L293D πάνω από τους οδηγούς L293D του πίνακα. Το L293D επιτρέπει παράλληλη λειτουργία.



Εικόνα 6.13 L293D H-bridge Motor Driver Shield

6.5.1 Χαρακτηριστικά

- Δύο συνδέσεις για σερβοκινητήρες 5V που συνδέονται με τον χρονοδιακόπτη υψηλής ανάλυσης του Arduino.
- Έως τέσσερις κινητήρες συνεχούς ρεύματος διπλής κατεύθυνσης με επιμέρους επιλογή ταχύτητας 8 bits .
- Μέχρι δύο βηματικούς κινητήρες (μονοπολικούς ή διπολικούς) με μονή σπείρα, διπλό πηνίο, παρεμβαλλόμενο ή μικροκλιμάκωση.
- Τέσσερις H-Γέφυρες: Το L293D παρέχει 0.6 A ανά γέφυρα (κορυφή 1.2A) με θερμική απενεργοποίηση προστασίας 4.5V έως 25V.

- Pull down αντιστάσεις, που κρατούν τους κινητήρες απενεργοποιημένους κατά την ενεργοποίηση.
- Κουμπί επαναφοράς Arduino .
- Κλειδαριά ακροδεκτών δύο pins για τη σύνδεση εξωτερικής τροφοδοσίας.

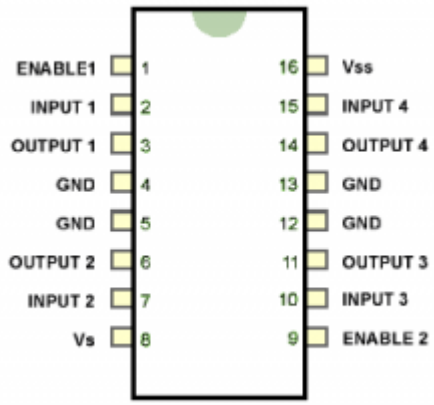
6.5.2 Οδηγός κινητήρα L293D

Το L293D είναι ένας τυπικός οδηγός κινητήρα ή IC Driver, ο οποίος επιτρέπει στον κινητήρα DC να κινείται προς οποιαδήποτε κατεύθυνση. Είναι ένα ψηφιακό κύκλωμα 16 pins, το οποίο μπορεί ταυτόχρονα να ελέγχει ένα σύνολο δύο κινητήρων συνεχούς ρεύματος. Σε ένα ενιαίο L293D τσιπ, υπάρχουν δύο H-Bridge κυκλώματα μέσα στο IC που μπορούν να περιστρέψουν δύο ανεξάρτητα μοτέρ DC.

Η γέφυρα (bridge) είναι ένα κύκλωμα που επιτρέπει την τάση να κινείται προς οποιαδήποτε κατεύθυνση.Η γέφυρα Ic είναι ιδανική για την οδήγηση ενός DC κινητήρα. Λόγω του μεγέθους του, το τσιπ χρησιμοποιείται σε ρομποτικές εφαρμογές για τον έλεγχο DC κινητήρων.

6.5.3 Λειτουργία L293D

Υπάρχουν τέσσερα pins εισόδου για το L293D, τα pins 2 και 7 στα αριστερά και τα pins 15 και 10 δεξιά, όπως φαίνεται στο διάγραμμα pin (εικόνα 6.14). Τα pins αριστεράς εισόδου ρυθμίζουν την περιστροφή του μοτέρ που συνδέεται στην αριστερή πλευρά και τα pins δεξιάς εισόδου για τον κινητήρα στη δεξιά πλευρά. Οι κινητήρες περιστρέφονται με βάση τις εισόδους που παρέχονται στα pins εισόδου ως Λογικό 0 ή Λογικό 1.

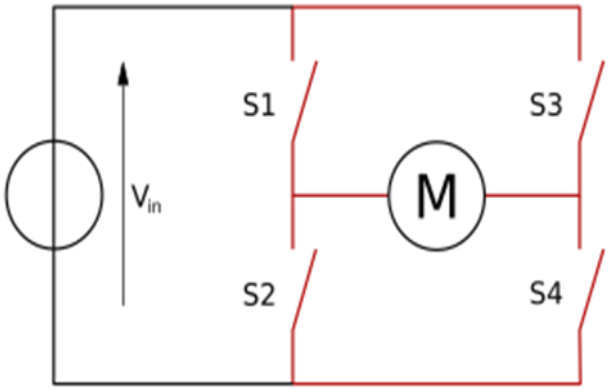


Εικόνα 6.14 Διάγραμμα pin L293D

Για την περιστροφή του κινητήρα κατά τη φορά των δεικτών του ρολογιού, τα pins εισόδου πρέπει να είναι εφοδιασμένα με Λογικό 1 και Λογικό 0 αντιστοίχως.Τα pins Enable 1 και 9 (αντίστοιχα στους δύο κινητήρες) πρέπει να είναι HIGH για να αρχίσουν να λειτουργούν οι κινητήρες. Όταν η είσοδος ενεργοποίησης είναι HIGH, ενεργοποιείται το σχετικό πρόγραμμα οδήγησης. Ως αποτέλεσμα, οι έξοδοι ενεργοποιούνται και λειτουργούν σε φάση με τις εισόδους τους. Ομοίως, όταν η είσοδος ενεργοποίησης είναι LOW, ο driver είναι απενεργοποιημένος και οι έξοδοί του είναι απενεργοποιημένες και σε κατάσταση υψηλής σύνθετης αντίστασης.

6.5.4 Λειτουργία H-Bridge

Έχει αυτό το όνομα, επειδή μπορεί να διαμορφωθεί ως τέσσερις διακόπτες στις γωνίες του 'H'.



Εικόνα 6.15 Βασικό διάγραμμα γέφυρας
https://en.wikipedia.org/wiki/H_bridge

Με αυτό το διάγραμμα, μπορούμε να εξηγήσουμε τη λειτουργία του κινητήρα ως μέρος της γέφυρας. Το βέλος στα αριστερά δείχνει την υψηλότερη πιθανή πλευρά της τάσης εισόδου του κυκλώματος.

Αν οι διακόπτες S1 & S4 διατηρούνται σε κλειστή θέση (ON) ενώ οι διακόπτες S2 & S3 διατηρούνται σε ανοικτή θέση (OFF), το κύκλωμα βραχυκυκλώνεται στους διακόπτες S1 & S4. Αυτό δημιουργεί μια διαδρομή για να ρέει το ρεύμα, ξεκινώντας από την είσοδο V, για να μεταβεί το S1 στον κινητήρα, στη συνέχεια για να μεταβεί το S4 και στη συνέχεια το εξερχόμενο από το κύκλωμα. Αυτή η ροή του ρεύματος θα κάνει τον κινητήρα να γυρίσει προς μία κατεύθυνση. Η κατεύθυνση κίνησης του κινητήρα μπορεί να είναι δεξιόστροφη ή αριστερόστροφη, επειδή η

περιστροφή του κινητήρα εξαρτάται από τη σύνδεση των ακροδεκτών του κινητήρα με τους διακόπτες.Ας υποθέσουμε ότι σε αυτή την κατάσταση ο κινητήρας περιστρέφεται κατά τη φορά των δεικτών του ρολογιού.

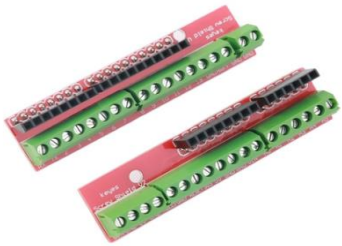
Τώρα, όταν τα S3 και S2 είναι κλειστά και τα S1 και S4 παραμένουν ανοιχτά, τότε το ρεύμα ρέει από την άλλη κατεύθυνση και ο κινητήρας σίγουρα περιστρέφεται αντίθετα προς τη φορά των δεικτών του ρολογιού.

Όταν τα S1 και S3 είναι κλειστά και τα S2 και S4 είναι ανοικτά, τότε θα εμφανιστεί η κατάσταση 'STALL' (ο κινητήρας θα σπάσει). Δηλαδή, όταν εφαρμοστεί θετική τάση και στις δύο πλευρές, τότε η τάση και από τις δύο πλευρές φέρνει τον άξονα του κινητήρα σε αναστολή.

Τέλος, αν κλείσουμε τους διακόπτες S1 και S2 ή τους S3 και S4, προκαλούμε βραχυκύκλωμα της πηγής τάσεως, κατάσταση που είναι ανεπιθύμητη.

6.6 Screw shield

Η Screw shield είναι μια ασπίδα που μοιάζει με πτερύγιο και εκτείνεται και στις δύο πλευρές του Arduino σε ανθεκτικά και αξιόπιστα μπλοκ ακροδεκτών.Στην πλευρά των ψηφιακών ακίδων, περιέχει 18 ακροδέκτες: 16 κανονικά τερματικά και 2 πρόσθετα GND. Όσον αφορά την πλευρά αναλογικών ακροδεκτών, υπάρχουν 16 ακροδέκτες (12 κανονικοί ακροδέκτες, 4 επιπλέον ακροδέκτες για το Vin, 2* GND και 5V) και 2 για πρόσθετες ακίδες ADC. Ενα ειδικό τεμάχιο επέκτασης για τη διασύνδεση SPI περιλαμβάνεται σε αυτήν, για να καταστεί μια πραγματικά ισχυρή ασπίδα για άλλες ασπίδες όπως η οθόνη αφής TFT.

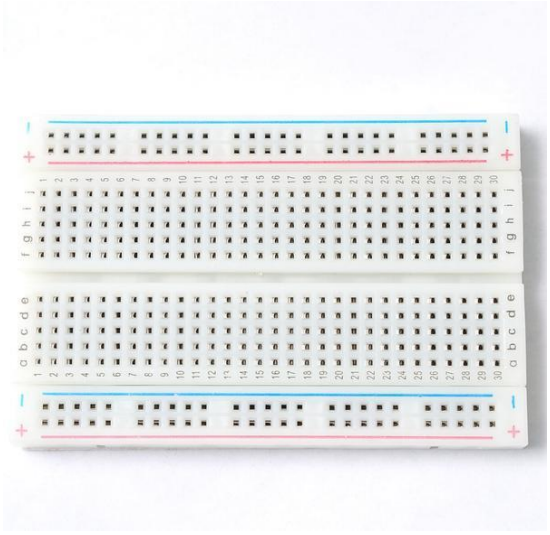


Εικόνα 6.16 Screw shields

6.7 Breadboard

Ένα breadboard είναι ένας ορθογώνιος πλαστικός πίνακας με μια δέσμη μικροσκοπικών τρυπών στην επιφάνειά του. Αυτές οι τρύπες συνδέονται κάτω

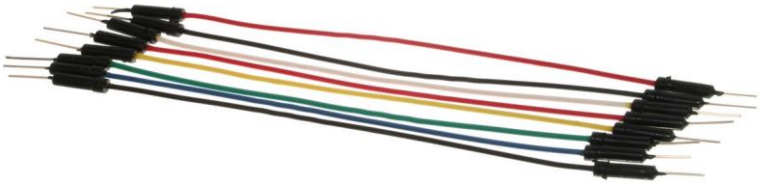
από την επιφάνεια γαλβανικά είτε σε οριζόντια διάταξη και αυτό σημειώνεται με συνεχείς γραμμές (κόκκινες,μπλε) είτε κατακόρυφα ανά πεντάδες που αριθμούνται με συνδυασμό γραμμάτων και αιθμών. Αυτή η διάταξη μάς επιτρέπει την εύκολη εισαγωγή ηλεκτρονικών εξαρτημάτων, για να χτίσουμε και να δοκιμάσουμε μια πρώιμη έκδοση ενός ηλεκτρονικού κυκλώματος.Οι συνδέσεις δεν είναι μόνιμες, επομένως είναι εύκολο να αφαιρέσουμε ένα στοιχείο εάν γίνει κάποιο λάθος ή απλά να ξεκινήσουμε ένα νέο έργο.



Εικόνα 6.17 Breadboard

6.8 Jump wire

Ένα jump wire (γνωστό και ως jumper, jumper wire, jumper cable, καλώδιο DuPont ή DuPont wire/cable - ονομασία λόγω του κατασκευαστή του) είναι ένα ηλεκτρικό καλώδιο ή μια ομάδα καλωδίων, που έχουν ακροδέκτες θηλυκούς ή αρσενικούς σε κάθε άκρο τους (μπορεί, όμως, και να μην έχουν ακροδέκτες) και τα χρησιμοποιούμε, συνήθως, για τη σύνδεση των εξαρτημάτων σε έ-να breadboard ή σε άλλα κυκλώματα δοκιμής ή με άλλο εξοπλισμό ή εξαρτήματα χωρίς να απαιτείται συγκόλληση.



Εικόνα 6.18 Jump wires

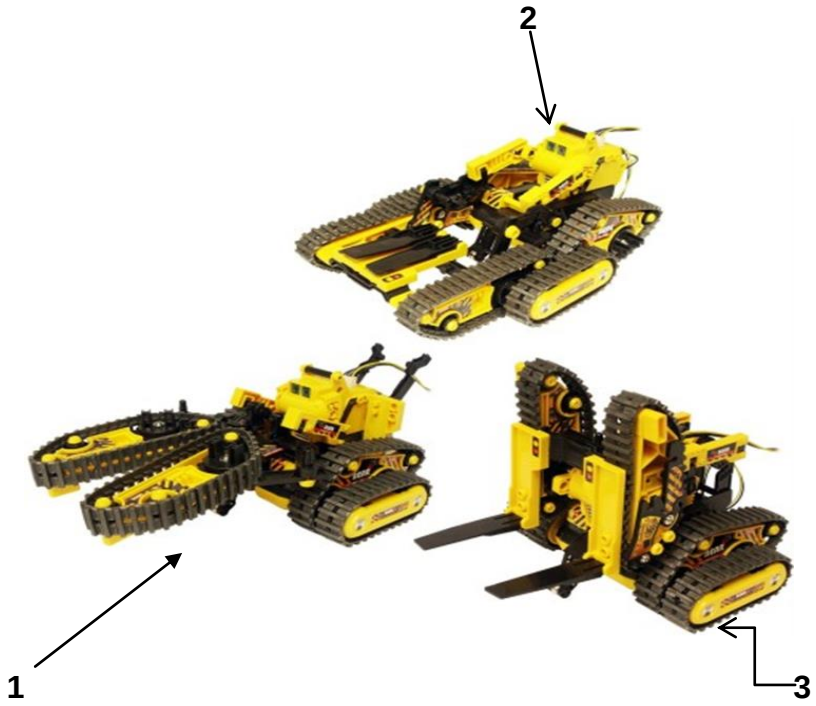
ΚΕΦΑΛΑΙΟ 7^{ο}: ΥΛΟΠΟΙΗΣΗ ΤΩΝ ΑΣΚΗΣΕΩΝ

Στο παρόν κεφάλαιο, παρουσιάζεται η πορεία υλοποίησης των τεσσάρων κατασκευών, οι οποίες αποτέλεσαν την αφόρμηση της εργασίας μας, παρουσίαση που περιλαμβάνει τόσο τα υλικά που χρησιμοποιήθηκαν, όσο και το τελικό αποτέλεσμα.

Έτσι, το κεφάλαιο αυτό αποτελεί έμπρακτη εφαρμογή του θεωρητικού μέρους της πτυχιακής μας εργασίας.

Το OWI 536 3in1 ATR (ALL TERRAIN ROBOT), που χρησιμοποιούμε στις εφαρμογές μας, είναι ένα κινητό ρομποτικό κιτ πολλαπλών λειτουργιών με χαμηλό κόστος απόκτησης και πολλές δυνατότητες για εφαρμογές εκπαιδευτικής ρομποτικής. Το συγκεκριμένο όχημα μπορεί να διαμορφωθεί σε μία από τις παρακάτω λειτουργίες:

1. Gripper (Τσιμπίδα)
2. Rover (Ρόβερ)
3. Forklift (Ανυψωτικό): μορφή που χρησιμοποιείται στις εφαρμογές μας



Εικόνα 7.1 Λειτουργίες ρομποτικού οχήματος

Αρχικά, αποσυνδέουμε από το όχημα το ενσύρματο χειριστήριο και τα καλώδια τροφοδοσίας των τριών DC κινητήρων, τα οποία τα σημειώνουμε σε ετικέτες πά-

νω στο όχημα ως M1, M2, M3 και τα επεκτείνουμε με καλώδια κίτρινου χρώματος για τροφοδοσία (+) και μαύρου για τροφοδοσία (-).

Στη συνέχεια, τοποθετούμε στο διαμορφωμένο όχημα μία βάση από πλέξ γκλάς, στο σημείο που δηλώνεται στην εικόνα 7.3 (σημείο Α).Πάνω σε αυτήν,θα τοποθετηθούν τα υπόλοιπα υλικά και συσκευές που θα χρησιμοποιηθούν για την υλοποίηση τω εφαρμογών που ακολουθούν.

Εικόνα 7.2 Η βάση στήριξης

Εικόνα 7.3 Το ανυψωτικό όχημα με το σημείο τοποθέτησης της βάσης

7.1 ΕΦΑΡΜΟΓΗ 1: Κίνηση ρομποτικού οχήματος μέσω Arduino και εφαρμογής Android

Σκοπός της εφαρμογής είναι να μπορούν οι σπουδαστές να προγραμματίζουν μία πλατφόρμα arduino, που θα κινεί ρομποτικό όχημα και στη συνέχεια θα συνδεθεί μέσω Bluetooth με εφαρμογή android.

7.1.1 Στόχοι

- Να συνδέουν τα διάφορα απαραίτητα υλικά και συσκευές για τη δημιουργία του ρομποτικού οχήματος και της κίνησής του.
- Να προγραμματίζουν την πλατφόρμα arduino για την κίνηση του ρομποτικού οχήματος.
- Να συνδέουν μέσω Bluetooth το arduino με την αντίστοιχη εφαρμογή android.

7.1.2 AFMotorLibrary

Η βιβλιοθήκη Fmotor της Adafruit είναι μια χρήσιμη βιβλιοθήκη για τον έλεγχο DC και βηματικών κινητήρων με την Adafruit Motor Shield.

- **AF_DCMotor Class:**Η κλάση AF_DCMotor παρέχει έλεγχο ταχύτητας και κατεύθυνσης μέχρι και για τέσσερις ηλεκτροκινητήρες DC, όταν χρησιμοποιείται με την Adafruit Motor Shield. Για να χρησιμοποιηθεί σε ένα σκίτσο (sketch), πρέπει πρώτα να προστεθεί η ακόλουθη γραμμή στην αρχή :
  #include

- **AF_DC Motormotorname (portnum,freq):** Αυτός είναι ο κατασκευαστής για κινητήρα DC. Καλούμε τον κατασκευαστή μία φορά για κάθε κινητήρα στο δικό μας σκίτσο. Κάθε παράσταση κινητήρα πρέπει να έχει διαφορετικό όνομα.

Παράμετροι:

- **port num**: Επιλέγει σε ποιο κανάλι (1-4) του ελεγκτή κινητήρα θα είναι συνδε-δεμένος ο κινητήρας
- **freq**: Επιλέγει τη συχνότητα PWM. Εάν δεν έχει καθοριστεί συχνότητα, χρησιμοποιείται 1KHz από προεπιλογή

Οι συχνότητες για το κανάλι 1 & 2 είναι:

- MOTOR12_64KHZ
- MOTOR12_8KHZ
- MOTOR12_2KHZ
- MOTOR12_1KHZ

Οι συχνότητες για το κανάλι 3 & 4 είναι:

- MOTOR34_64KHZ

- MOTOR34_8KHZ
- MOTOR34_1KHZ

- **setSpeed(speed):**Ορίζει την ταχύτητα του κινητήρα.

Παράμετροι:

- **speed**: Οι έγκυρες τιμές για την ταχύτητα είναι μεταξύ 0 και 255 με το 0 να είναι απενεργοποιημένο και το 255 για πλήρη ταχύτητα.

Σημείωση: Η απόκριση μοτέρ DC δεν είναι τυπικά γραμμική και συνεπώς οι πραγματικές RPM δεν είναι απαραιτήτως ανάλογες με την προγραμματισμένη ταχύτητα.

- **run(cmd)**

Παράμετροι:

- **cmd**: η επιθυμητή λειτουργία λειτουργίας για τον κινητήρα.

Οι έγκυρες τιμές για cmd είναι:

- **FORWARD**: κίνηση προς τα εμπρός (η πραγματική κατεύθυνση περιστροφής εξαρτάται από την καλωδίωση του κινητήρα)
- **BACKWARD**: κίνηση προς τα πίσω (η περιστροφή θα είναι προς την αντίθετη κατεύθυνση από την FORWARD)
- **RELEASE**: Σταματάμε τον κινητήρα. Αυτό αφαιρεί την ισχύ από τον κινητήρα και ισοδυναμεί με setSpeed(0).

7.1.3 Συσκευές και υλικά

1. Όχημα OWI-536 .
2. Βάση στήριξης
3. Screw shields
4. HC-05 - Bluetooth Module
5. Διακόπτης
6. Πινακίδες πειραμάτων (breadboard)
7. Πλατφόρμα ArduinoUNORev3
8. Μπαταρία Μολύβδου MHB MS1.3-6 (Πηγή τροφοδοσίας)
9. L293D H-bridge Motor Driver Shield
10. Κινητή συσκευή με λειτουργικό android

11. Καλώδια σύνδεσης

Εικόνα 7.4 Υλικά εφαρμογής 1

7.1.4 Βήματα εκτέλεσης εργασίας

1. Πάνω στη βάση τοποθετούμε την μπαταρία 6V/1.3mA.

2.Συνδέουμε τα βοηθητικά Screw shields πάνω στο Arduino.

3.Μέσω των Screw shields, συνδέουμε την motor driver shield με το Arduino και την τοποθετούμε πάνω στη βάση στήριξης.

4.Τοποθετούμε ένα breadboard στο σημείο που δηλώνεται στην εικόνα 7.6 (σημείο Β), τοποθετούμε πάνω τον διακόπτη τροφοδοσίας και συνδέουμε το κόκκινο καλώδιο τροφοδοσίας της μπαταρίας στο M+ της εξωτερικής τροφοδοσίας (EXT_PWR) μέσω του διακόπτη και το μαύρο καλώδιο γείωσης στο GND.

5.Συνδέουμε το κίτρινο καλώδιο του δεξιού κινητήρα στο πάνω μέρος του M1(+) και το μαύρο καλώδιο στο κάτω μέρος του M1(-). Αντίστοιχα και για τον αριστερό κινητήρα.

6.Συνδέουμε το κόκκινο καλώδιο του μεσαίου κινητήρα με το κάτω μέρος του M3(+) και το μαύρο καλώδιο με το πάνω μέρος του M3(-).

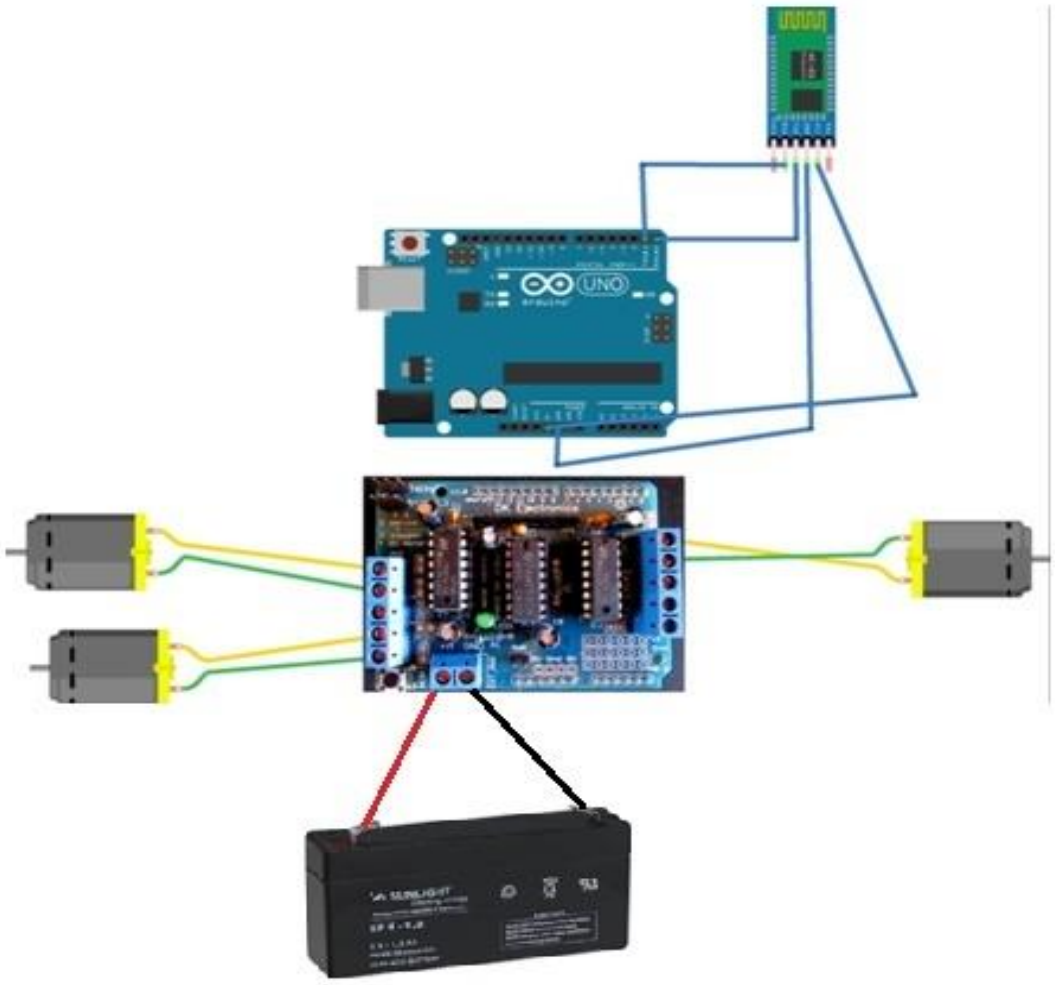
7.Τοποθετούμε το Bluetooth στο breadboard και το συνδέουμε ως εξής:

RXD → TX (digital pin 1)

TXD → RX (digital pin 0)

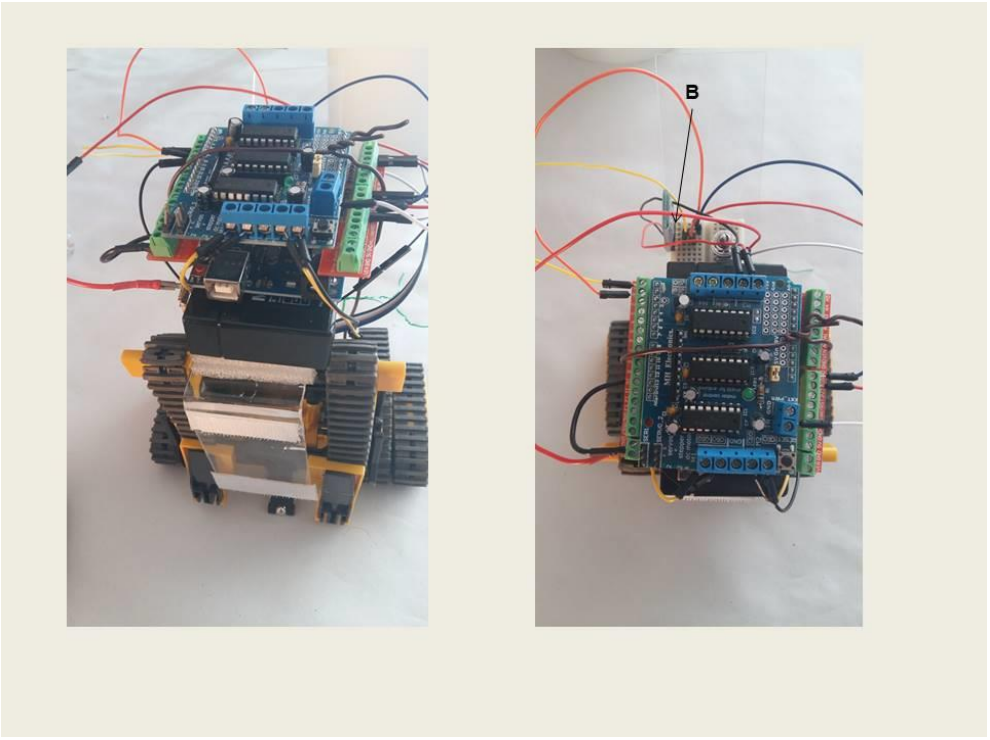
GND → GND

VCC → 5V



Εικόνα 7.5 Κύκλωμα κίνησης ρομποτικού οχήματος

΄Υστερα από τα παραπάνω,το όχημά μας παίρνει τη μορφή της εικόνας 7.6.



Εικόνα 7.6 Εφαρμογή 1

7.1.5 Κώδικας Arduino

Ξεκινάμε με την εντολή #include, που μας δίνει την δυνατότητα να εισαγάγουμε βιβλιοθήκες στο σκίτσο μας και συγκεκριμένα την AFMotor.

#include <AFMotor.h>

Ο κατασκευαστής AF_DC Motor δημιουργεί 3 κλάσεις DC κινητήρων motor, motor2, motor3 στις πόρτες 1,2,3 της shield με PWM συχνότητα 64 KHZ.

AF_DCMotormotor(1, MOTOR12_64KHZ);

AF_DCMotormotor2(2, MOTOR12_64KHZ);

AF_DCMotormotor3(3, MOTOR12_64KHZ);

Η μεταβλητή incomingByte είναι μέγεθος ενός χαρακτήρα και αποθηκεύει τον χαρακτήρα που βλέπει η σειριακή από το android

charincomingByte;

Στη συνάρτηση setup() γίνεται έναρξη της σειριακής οθόνης με bit ανα δευτερόλεπτο(baud rate) 9600.

Serial.begin(9600);

Η ταχύτητα των κινητήρων ορίζεται με την εντολή setSpeed με τιμες 0-255.Λόγω του ότι το όχημα κινείται σε 8 κατευθύνσεις, δε χρειάζεται να πηγαίνει τέλεια ευθεία, οπότε δε χρειάζεται να ανησυχούμε για τη μη γραμμικότητα των μοτέρ των κινητήρων, οπότε μπορούμε να βάλουμε μέγιστη τιμή 255. Διαφορετικά, πρέπει να αλλάζουμε την τιμή μέχρι να βρούμε μία τιμή που να κινεί το όχημα τελέιως ευθεία.

motor.setSpeed(255);
motor2.setSpeed(255);
motor3.setSpeed(255);

Στη συνάρτηση loop() ξεκινάμε ελέγχοντας αν υπάρχουν bytesγια διάβασμα.

if (Serial.available() > 0)

Όπως είπαμε παραπάνω, η τιμή της incomingByte είναι ό,τι δείχνει η σειριακή, η οποία παίρνει τις τιμές της από την outputstream του android.

incomingByte = Serial.read();

Ανάλογα με το τι χαρακτήρα διαβάζει, κινεί τους κινητήρες προς την κατάλληλη κατεύθυνση , για να πετύχουμε την επιθυμητή κίνηση του οχήματος.

Για κίνηση εμπρός:

if(incomingByte=='f'){
motor.run(BACKWARD); /
motor2.run(FORWARD);
}

Για κίνηση προς τα πίσω :

if(incomingByte=='b'){
motor.run(FORWARD);
motor2.run(BACKWARD);
}

Για κίνηση δεξιά:

if(incomingByte=='q'){
motor.run(FORWARD);

```
motor2.run(FORWARD);
}
```

Για κίνηση αριστερά:

```
 if(incomingByte=='w'){
 motor.run(BACKWARD);
 motor2.run(BACKWARD);
 }
```

Για κίνηση μπροστά-αριστερά:

```
if(incomingByte=='l'){
motor.run(BACKWARD);
motor2.run(RELEASE);
}
```

Για κίνηση μπροστά-δεξιά:

```
if(incomingByte=='r'){
motor2.run(FORWARD);
motor.run(RELEASE);
}
```

Για κίνηση πίσω-αριστερά:

```
if(incomingByte=='k'){
motor.run(FORWARD);
motor2.run(RELEASE);
   }
```

Για κίνηση πίσω-δεξιά:

```
if(incomingByte=='j'){
motor2.run(BACKWARD);
motor.run(RELEASE);
   }
```

Για την κίνηση του ανυψωτικού, κινούμε τον τρίτο κινητήρα προς την μία κατεύθυνση για κίνηση προς τα πάνω και προς την άλλη για κίνηση προς τα κάτω.

```
if(incomingByte=='d'){
motor3.run(FORWARD);
}
if(incomingByte=='u'){
motor3.run(BACKWARD);
}
```

Για να σταματήσουμε το όχημα, δίνουμε την παράμετρο RELEASE σε όλους τους κινητήρες.

```
if(incomingByte=='s'){
motor.run(RELEASE);  //stop
motor2.run(RELEASE);
motor3.run(RELEASE);
}
```

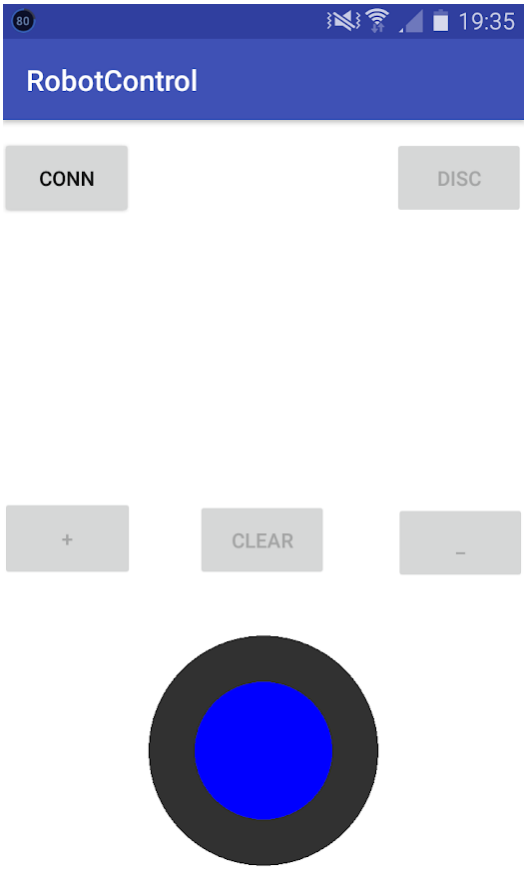
7.1.6 Λειτουργία εφαρμογής ANDROID

Η εφαρμογή θα ελέγχει το όχημα ασύρματα μέσω Bluetooth module, το οποίο είναι συνδεδεμένο με το Arduino.

- Το κουμπί CONN ελέγχει για το συγκεκριμένο Bluetooth module και, αν το βρεί, συνδέει τη συσκευή με αυτό, οπότε και με το Arduino. Επίσης, ενεργοποιεί τις μεθόδους outputstream και inputsream, που επιτρέπουν την αμφίδρομη επικοινωνία της συσκευής android με τη σειριακή του Arduino.
- Το κουμπί DISC διακόπτει όποια επικοινωνία του android με το Arduino.
- Έχουμε ένα joystick, το οποίο, ανάλογα με τη θέση του, θα στέλνει χαρακτήρες στη σειριακή του Arduino και αυτό θα κινεί τους κινητήρες του οχήματος ανάλογα.
- Τα 2 πλήκτρα + και – έλεγχουν την έξτρα λειτουργία του οχήματος με τον τρίτο κινητήρα.
- Το Textview στη μέση της οθόνης χρησιμεύει για την εμφάνιση των αποτελεσμάτων του επιταχυνσιόμετρου της επόμενης άσκησης.

- Το πλήκτρο CLEAR σβήνει τα υπάρχοντα εμφανιζόμενα δεδομένα για να βλέπουμε τα πιο πρόσφατα καλύτερα.

Για αναλυτική περιγραφή του κώδικα,βλ. Παράρτημα.



Εικόνα 7.7 Η διεπαφή της εφαρμογής

7.2 ΕΦΑΡΜΟΓΗ 2: Παρακολούθηση της επιτάχυνσης του ρομποτικού οχήματος

Σκοπός της εφαρμογής είναι να μπορούν οι σπουδαστές να εμφανίζουν τα αποτελέσματα του επιταχυνσιόμετρου σε οθόνη κινητής συσκευής μέσω της εφαρμογής android.

7.2.1 Στόχοι

- Να συνδέουν το επιταχυνσιόμετρο και τις διάφορες συσκευές πάνω στο ρομποτικό όχημα.

- Να παρακολουθούν τα αποτελέσματα του επιταχυνσιόμετρου στην οθόνη της κινητής συσκευής μέσω της αντίστοιχης εφαρμογής android.

7.2.2 Wire library

Αυτή η βιβλιοθήκη επιτρέπει να επικοινωνούμε με συσκευές I2C / TWI. Στις πλακέτες Arduino με διάταξη R3 (1.0 pinout), οι γραμμές SDA (γραμμή δεδομένων) και SCL (γραμμή ρολογιού) βρίσκονται στις κεφαλίδες των ακίδων κοντά στην ακίδα AREF. Το Arduino Due έχει δύο διασυνδέσεις I2C / TWI Οι SDA1 και SCL1 είναι κοντά στην AREF pin και η πρόσθετη είναι στις ακίδες 20 και 21.

Τα παρακάτω δεδομένα δείχνουν πού βρίσκονται οι ακροδέκτες TWI σε διάφορες πλακέτες Arduino (Πίνακας 7.1).

Πίνακας 7.1 Θέση ακροδεκτών TWI στις πλακέτες Arduino

Πλακέτες	**I2C / TWI**
Uno, Ethernet	A4 (SDA), A5 (SCL)
Mega2560	20 (SDA), 21 (SCL)
Leonardo	2 (SDA), 3 (SCL)
Due	20 (SDA), 21 (SCL), SDA1, SCL1

Από το Arduino 1.0, η βιβλιοθήκη κληρονομεί τις λειτουργίες Stream, καθιστώντας το συνεπές με άλλες βιβλιοθήκες ανάγνωσης / εγγραφής. Εξαιτίας αυτού, οι send () και receive () έχουν αντικατασταθεί με read () και write ().

Σημείωση: Υπάρχουν εκδόσεις των διευθύνσεων I2C σε 7bits και 8bits. Τα 7 bits εντοπίζουν τη συσκευή και το όγδοο κομμάτι καθορίζει αν πραγματοποιείται εγγραφή η ανάγνωση.

Η βιβλιοθήκη Wire χρησιμοποιεί διευθύνσεις 7 bits καθ' όλη τη διάρκειά της. Αν έχουμε ένα φύλλο δεδομένων (datasheet) ή δείγμα κώδικα που χρησιμοποιεί διεύθυνση 8 bits, θα χρειαστεί να μετακινήσουμε την τιμή ένα bit προς τα δεξιά, αποδίδοντας μια διεύθυνση μεταξύ 0 και 127. Ωστόσο, οι διευθύνσεις από 0 έως 7 δεν χρησιμοποιούνται, επειδή είναι δεσμευμένες, έτσι η πρώτη διεύθυνση που μπορεί να χρησιμοποιηθεί είναι η 8. Σημειώνουμε ότι απαιτείται αντίσταση pull-up όταν συνδέουμε τις ακίδες SDA / SCL.

Βασικές εντολές

- **Wire.begin():**Ξεκινά τη βιβλιοθήκη Wire και συνδέει το δίαυλο I2C ως master ή slave. Αυτό πρέπει κανονικά να καλείται μόνο μία φορά.

Σύνταξη

- Wire.begin(address)

Παράμετροι

- address: η διεύθυνση slave 7-bit (προαιρετική). εάν δεν έχει οριστεί, συνδέουμε τον δίαυλο ως master.

- **Wire.requestFrom():**Χρησιμοποιείται από τον master για να ζητήσει bytes από μια slave συσκευή. Τα bytes μπορούν στη συνέχεια να ανακτηθούν με τις διαθέσιμες λειτουργίες () και read (). Από το Arduino 1.0.1, το requestFrom () δέχεται ένα boolean όρισμα που αλλάζει τη συμπεριφορά του για συμβατότητα με ορισμένες συσκευές I2C.

Αν είναι αληθές, το requestFrom () αποστέλλει ένα μήνυμα διακοπής μετά το αίτημα, απελευθερώνοντας τον δίαυλο I2C.

Αν είναι ψευδές, το requestFrom () στέλνει ένα μήνυμα επανεκκίνησης μετά το αίτημα. Ο δίαυλος δεν θα απελευθερωθεί, πράγμα που εμποδίζει μια άλλη κύρια συσκευή να ζητάει μηνύματα. Αυτό επιτρέπει σε μία κύρια συσκευή να στέλνει πολλαπλά αιτήματα ενώ βρίσκεται υπό έλεγχο. Η προεπιλεγμένη τιμή είναι αληθής.

Σύνταξη

- Wire.requestFrom (address, quantity)
- Wire.requestFrom(address, quantity, stop)

Παράμετροι

- **address**: η διεύθυνση 7-bit της συσκευής από τη οποία ζητά bytes
- **quantity**: ο αριθμός των bytes που απαιτούνται.
- **stop**: boolean.

Αν είναι αληθές, θα στείλει ένα μήνυμα διακοπής μετά το αίτημα, απελευθερώνοντας το δίαυλο.Αν είναι ψευδές, θα στείλει ένα μήνυμα επανεκκίνησης μετά από το αίτημα, κρατώντας τη σύνδεση ενεργή.

- **Wire.beginTransmission()**:Ξεκινά μια μετάδοση στη συσκευή slave του I2C με τη δεδομένη διεύθυνση. Στη συνέχεια, βάζει τα bytes σε μια σειρά για μετάδοση με τη συνάρτηση write () και τα μεταδίδει καλώντας endTransmission ().

Σύνταξη

- Wire.beginTransmission(address)

Παράμετροι

- **address:** η διεύθυνση 7-bits της συσκευής για τη μετάδοση (slave). Εάν δίνονται 8 bits, το κορυφαίο κομμάτι απλά περικόπτεται.

- **Wire.endTransmission():**Τερματίζει μια μετάδοση σε μια slave συσκευή που ξεκίνησε από την beginTransmission() και μεταδίδει τα bytes που ήταν σε σειρά με write(). Από το Arduino 1.0.1, το endTransmission() δέχεται ένα boolean όρισμα, που αλλάζει τη συμπεριφορά του για συμβατότητα με ορισμένες συσκευές I2C.Αν είναι αληθές, η endTransmission () αποστέλλει ένα μήνυμα διακοπής μετά τη μετάδοση, απελευθερώνοντας το δίαυλο I2C. Εάν είναι ψευδές, το endTransmission () στέλνει ένα μήνυμα επανεκκίνησης μετά τη μετάδοση. Ο δίαυλος δεν θα απελευθερωθεί, πράγμα που εμποδίζει τη μετάδοση άλλης κύριας συσκευής μεταξύ μηνυμάτων. Αυτό επιτρέπει σε μία κύρια συσκευή να στείλει πολλές μεταδόσεις ενώ είναι υπό έλεγχο.Η προεπιλεγμένη τιμή είναι αληθής.

Σύνταξη

- Wire.endTransmission()
- Wire.endTransmission(stop)

Παράμετροι

- **stop:** boolean.

Αν είναι αληθές, θα στείλει ένα μήνυμα διακοπής μετά το αίτημα, απελευθερώνοντας το δίαυλο. Αν είναι ψευδές,θα στείλει ένα μήνυμα επανεκκίνησης μετά από το αίτημα, κρατώντας τη σύνδεση ενεργή.

Επιστρέφει: byte, το οποίο υποδεικνύει την κατάσταση της μετάδοσης:

- 0: επιτυχία

- 1: Δεδομένα πολύ μεγάλα για να χωρέσουν στην προσωρινή μνήμη μετάδοσης
- 2: έλαβε NACK για τη μετάδοση της διεύθυνσης
- 3: έλαβε NACK για τη μετάδοση δεδομένων
- 4: άλλο σφάλμα

- **read():**Διαβάζει ένα byte που μεταδόθηκε από μια slave συσκευή σε έναν master μετά από μια κλήση προς requestFrom () ή μεταδόθηκε από έναν master σε έναν slave.

Σύνταξη

- Wire.read()

Παράμετροι:καμία

- **write():**Καταγράφει δεδομένα από μια slave συσκευή ως απόκριση σε ένα αίτημα από master ή δρομολογεί bytes για μετάδοση από master σε slave (μεταξύ κλήσεων για startTransmission () και endTransmission ()).

Σύνταξη

- Wire.write(value)
- Wire.write(string)
- Wire.write(data, length)

Παράμετροι

- ✓ **value**: τιμή για αποστολή ως μοναδικό byte
- ✓ **string**: μια συμβολοσειρά για αποστολή ως μια σειρά από bytes
- ✓ **data:** μια σειρά δεδομένων για αποστολή ως bytes
- ✓ **length:** ο αριθμός των bytes που μεταδίδουν

7.2.3 Συσκευές και υλικά

1. Όχημα OWI-536
2. Βάση στήριξης
3. Κινητή συσκευή με λειτουργικό android
4. Πλατφόρμα Arduino UNO Rev3
5. L293D H-bridge Motor Driver Shield
6. Καλώδια σύνδεσης

7. Μπαταρία Μολύβδου MHB MS1.3-6 (Πηγή τροφοδοσίας)
8. HC-05 - Bluetooth Module)
9. Αισθητήρας MPU 92500/6500
10. Πινακίδες πειραμάτων (breadboard
11. Διακόπτης
12. Screw shields

Εικόνα 7.8 Υλικά εφαρμογής 2

7.2.4 Βήματα εκτέλεσης της εργασίας

Για την εφαρμογή 2, ακολουθούμε τα βήματα 1-7 από την προηγούμενη άσκηση και στη συνέχεια τοποθετούμε ένα breadboard στο σημείο που δηλώνεται στην εικόνα 7.10 (σημείο Γ).Πάνω σε αυτό, τοποθετούμε το MPU 9250/6500 και το συνδέουμε ως εξής:

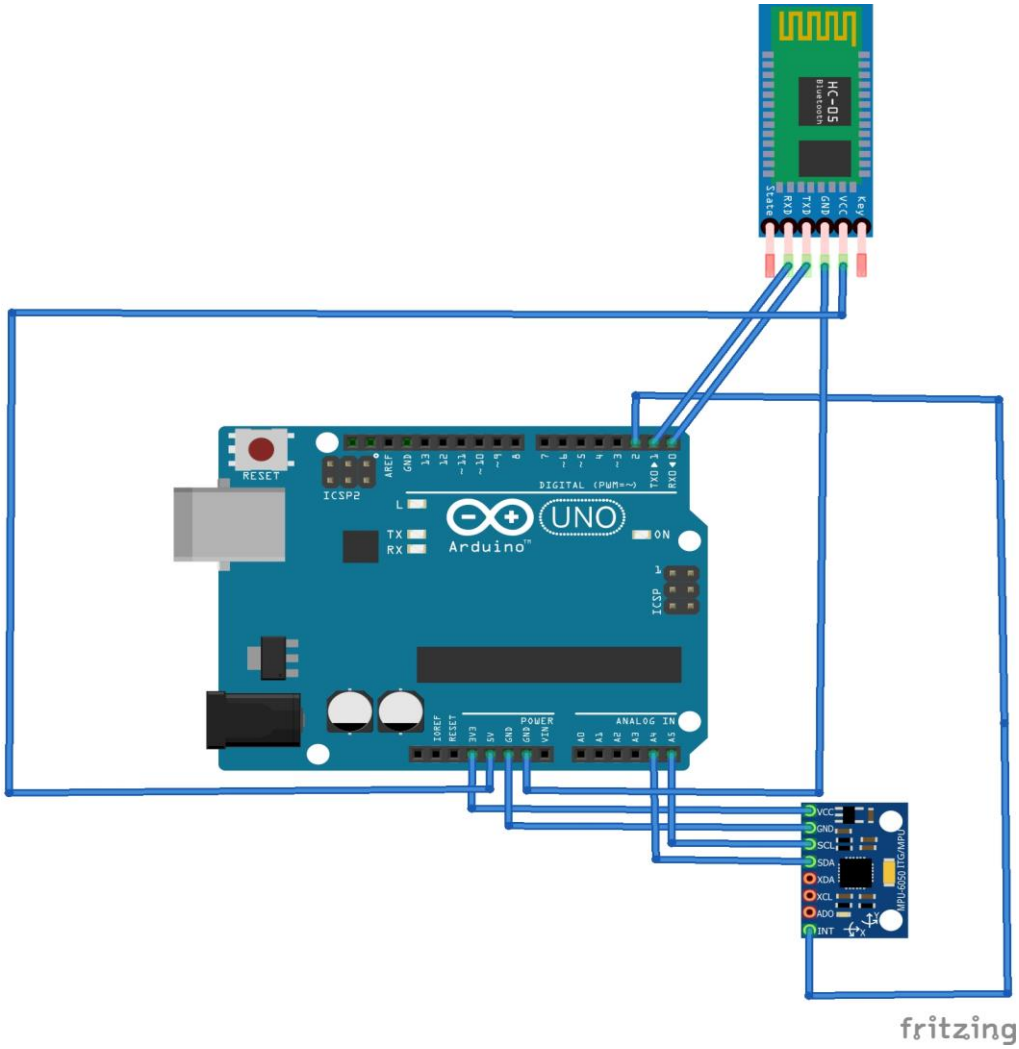
VCC → 3.3V

GND → GND

SCL → A5

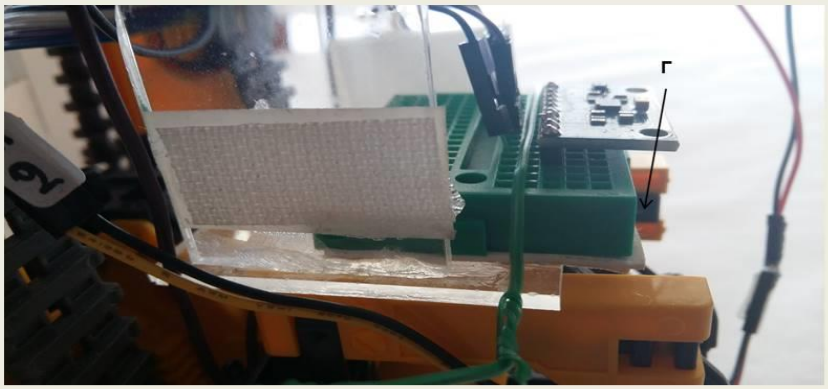
SDA → A4

INT → Digital pin 2



Εικόνα 7.9 Κύκλωμα σύνδεσης επιταχυνσιόμετρου MPU 9250/650

΄Υστερα από τα παραπάνω,το όχημά μας παίρνει τη μορφή της εικόνας 7.10.



Εικόνα 7.10 Εφαρμογή 2

7.2.5 Κώδικας ARDUINO

Ξεκινάμε με την συμπερίληψη των βιβλιοθηκών Wire και AFMotor με την εντολή include.

#include<Wire.h>

#include <AFMotor.h>

Ο κατασκευαστής AF_DCMotor δημιουργεί 3 κλάσεις DC κινητήρων motor, motor2, motor3 στις πόρτες 1,2,3 της shield με PWM συχνότητα 64 KHZ.

AF_DCMotor motor(1, MOTOR12_64KHZ);

AF_DCMotor motor2(2, MOTOR12_64KHZ);

AF_DCMotor motor3(3, MOTOR12_64KHZ);

Η μεταβλητή MPU_addr παίρνει την τιμή 0x68h που είναι η I2C διεύθυνση του MPU-9250 , ενώ οι μεταβλητές AcX,AcY,AcZ, θα αποθηκεύσουν τις τιμές του επι-ταχυνσιόμετρου στους 3 άξονες. Η μεταβλητή incomingByte είναι μέγεθος ενός χαρακτήρα και αποθηκεύει τον χαρακτήρα που βλέπει η σειριακή από το android. Η previousMillis δείχνει την τελευταία φορά που εμφανίστηκαν τα δεδομένα στην σειριακή και η interval κάθε πότε να εμφανίζονται.

const int MPU_addr=0x68;

double AcX,AcY,AcZ,;

char incomingByte;
unsigned long previousMillis=0;
long interval=500;

Στη συνάρτηση setup() , ξεκινάμε μία I2C σύνδεση με την Wire.begin() και επιλέγουμε τη σύνδεση με την I2C διεύθυνση του MPU-9250 με την Wire.-beginTransmission().

Στη συνέχεια, με την Wire.write πηγαίνουμε στον καταχωρητή PWR_MGMT_1 στη θέση 0x6B του register map του MPU-9250 και του δίνουμε την τιμή 0 για να ξυπνήσει το αισθητήριο και κλείνουμε τη σύνδεση με την Wire.endTransmission().

Τέλος,ενεργοποιούμε τη σειριακή με baud rate=9600 με την Serial.begin(),

Wire.begin();
Wire.beginTransmission(MPU_addr);
Wire.write(0x6B);
Wire.write(0);
Wire.endTransmission(true);
Serial.begin(9600);

Η ταχύτητα των κινητήρων ορίζεται με την εντολή setSpeed με τιμες 0-255.Λόγω του ότι το όχημα κινείται σε 8 κατευθύνσεις, δεν χρειάζεται να πηγαίνει τέλεια ευθεία, οπότε δεν χρειάζεται να ανησυχούμε για τη μη γραμμικότητα των μοτέρ των κινητήρων.Έτσι, μπορούμε να βάλουμε μέγιστη τιμή 255. Διαφορετικά, πρέπει να αλλάζουμε την τιμή μέχρι να βρούμε μία τιμή που να κινεί το όχημα τελέιως ευθεία.

motor.setSpeed(255);
motor2.setSpeed(255);
motor3.setSpeed(255);

Στη συνάρτηση loop(), ξεκινάμε ελέγχοντας αν υπάρχουν bytes για διάβασμα.

if (Serial.available() > 0)

Όπως είπαμε παραπάνω, η τιμή της incomingByte είναι ό,τι δείχνει η σειριακή, η οποία παίρνει τις τιμές της από την outputstream του android.

incomingByte = Serial.read();

Ανάλογα με το τι χαρακτήρα διαβάζει, κινεί τους κινητήρες προς την κατάλληλη κατεύθυνση για να πετύχουμε την επιθυμητή κίνηση του οχήματος.

```
if(incomingByte=='r'){
 motor2.run(FORWARD); //frontright
 motor.run(RELEASE);
}
if(incomingByte=='l'){
 motor.run(BACKWARD); //frontleft
 motor2.run(RELEASE);
}
if(incomingByte=='k'){
    motor.run(FORWARD); //backleft
    motor2.run(RELEASE);
   }
 if(incomingByte=='j'){
 motor2.run(BACKWARD); //backright
 motor.run(RELEASE);
}
  if(incomingByte=='f'){
  motor.run(BACKWARD); //forw
  motor2.run(FORWARD);
}
if(incomingByte=='b'){
  motor.run(FORWARD);  //back
  motor2.run(BACKWARD);
}
if(incomingByte=='s'){
   motor.run(RELEASE);  //stop
   motor2.run(RELEASE);
   motor3.run(RELEASE);
}
if(incomingByte=='d'){
  motor3.run(FORWARD);
```

```
  }
  if(incomingByte=='u'){
    motor3.run(BACKWARD);
  }
if(incomingByte=='q'){
     motor.run(FORWARD); //right
     motor2.run(FORWARD);
  }
if(incomingByte=='w'){
    motor.run(BACKWARD); //left
     motor2.run(BACKWARD);            }
```

Στη συνέχεια, ξεκινάμε τη σύνδεση με την I2C διεύθυνση του MPU-9250 με την Wire.beginTransmission().Οι μετρήσεις των τριών αξόνων του επιταχυνσιόμετρου αποθηκεύονται σε 2 καταχωρητές ο καθένας (HIGH και LOW byte). Με την Wire.write(0x3B), ξεκινάμε στον καταχωρητή 3B που βρίσκεται το HIGH byte του ACCEL_X.

Η Wire.endTransmission(false) θα στείλει μια επανεκκίνηση, κρατώντας τη σύνδεση ενεργή.

```
Wire.beginTransmission(MPU_addr
Wire.write(0x3B);
Wire.endTransmission(false);
```

Η Wire.requestFrom θα ζητήσει 6 bytes (2 για κάθε μέτρηση) από την slave συσκευή MPU-9250.

```
Wire.requestFrom(MPU_addr,6,true)
```

Με την Wire.read() διαβάζουμε το περιεχόμενο των καταχωρητών και το αποθηκεύουμε στην αντίστοιχη μεταβλητή.Θα χρειαστεί να κάνουμε bitshiftleft για 8 bits και να ξανακάνουμε Wire.read() στην ίδια μεταβλητή, για να πάρουμε και τα 2 τμήματα (αυτό το πετυχαίνουμε κάνοντας τη λογική πράξη OR).

Το πρόβλημα εδώ είναι ότι οι τιμές που παίρνουμε δεν είναι λογικές στην πραγματικότητα. Αυτό συμβαίνει επειδή το επιταχυνσιόμετρο χρησιμοποιεί διαφορετικούς συντελεστές κλιμάκωσης για διαφορετικά όρια επιτάχυνσης ή αλλιώς

ρυθμίσεις ευαισθησίας, οι οποίες μπορεί να είναι μία από τις +/- 2g,4g,8g ή 16g και κάθε μία έχει διαφορετικούς συντελεστές κλιμάκωσης ή ευαισθησία ως εξής:

Όριο επιτάχυνσης	**Ευαισθησία**
2g	**16,384**
4g	**8,192**
8g	**4.096**
16g	**2,048**

Ο τύπος για μετατροπή των τιμών του επιταχυνσιόμετρου είναι: **ΠΡΑΓΜΑΤΙΚΗ ΤΙΜΗ=(ΜΕΤΡΟΥΜΕΝΗ ΤΙΜΗ/ΕΥΑΙΣΘΗΣΙΑ)*G,** όπου G=9.81 m/s^2.

Στο συγκεκριμένο αισθητήριο, το όριο επιτάχυνσης είναι ρυθμιζόμενο στα 2g(που είναι και η προκαθορισμένη τιμή), οπότε για να βρούμε την πραγματική τιμή πολλαπλασιάζουμε την τιμή για κάθε διάσταση του επιταχυνσιόμετρου με 9.81 και την διαιρούμε με 16.384.

```
AcX=(Wire.read()<<8|Wire.read())*9.81/16384;
AcY=(Wire.read()<<8|Wire.read())*9.81/16384;
AcZ=(Wire.read()<<8|Wire.read())*9.81/16384;
```

Τέλος,πρέπει να εμφανίζουμε τις τιμές των τριών διαστάσεων στη σειριακή. Χρειαζόμαστε μια καθυστέρηση για να προλάβουμε να δούμε την κάθε μέτρηση πριν έρθει η επόμενη.Για να μην επέμβουμε στην ροή του υπόλοιπου προγράμματος, δεν μπορούμε να χρησιμοποιήσουμε την delay.Θα μετράμε πόσα μς έχουν περάσει από το ξεκίνημα του προγράμματος και κάθε 500 θα εμφανίζουμε τα δεδομένα και θα ανανεώνουμε την previousMillis.

```
unsigned long currentMillis= millis();
if (currentMillis - previousMillis >= interval) {
   previousMillis = currentMillis;
   printAccel();
 }
```

Η printAccel εμφανίζει τα δεδομένα στη σειριακή

```
void printAccel(){
Serial.print("AcX = "); Serial.print(AcX);
Serial.print(" | AcY = "); Serial.print(AcY);
```

```
Serial.print(" | AcZ = "); Serial.println(AcZ);
}
```

7.3 ΕΦΑΡΜΟΓΗ 3: Κίνηση ρομποτικού οχήματος σε προκαθορισμένη μαύρη γραμμή με χρήση υπερύθρου αισθητήρα (infrared) KY033

Σκοπός της εφαρμογής είναι να μπορούν οι σπουδαστές να προγραμματίζουν μία πλατφόρμα Arduino, η οποία θα κινεί ρομποτικό όχημα με δυνατότητες κίνησης σε προκαθορισμένη μαύρη γραμμή που θα ανιχνεύεται από Infrared αισθητήρα.

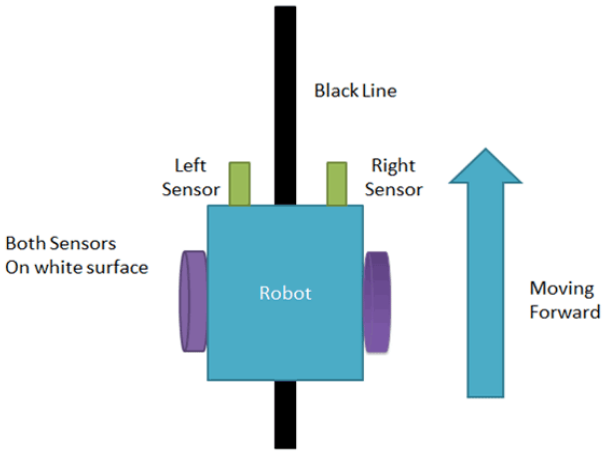
7.3.1 Στόχοι

- Να συνδέουν τα διάφορα απαραίτητα υλικά και συσκευές για τη δημιουργία του ρομποτικού οχήματος και της τροχιάς κίνησής του.
- Να προγραμματίζουν την πλατφόρμα Arduino για την κίνηση του ρομποτικού οχήματος σε συγκεκριμένη πορεία, με τη βοήθεια του αισθητήρα KY033.

7.3.2 Λειτουργία ρομπότ ακολουθίας γραμμής

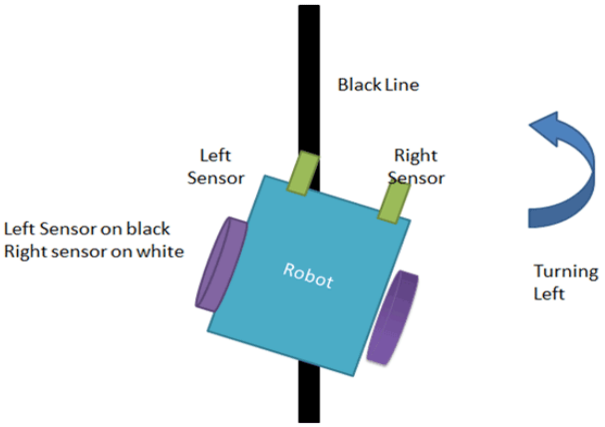
Το ρομπότ ακολουθίας γραμμής αντιλαμβάνεται τη μαύρη γραμμή χρησιμοποιώντας αισθητήρα και έπειτα στέλνει το σήμα στο Arduino. Στη συνέχεια, το Arduino οδηγεί τον κινητήρα σύμφωνα με την έξοδο των αισθητήρων. Εδώ, χρησιμοποιούμε δύο αισθητήρες υπερύθρων, αριστερό και δεξιό.

Όταν και οι δύο αισθάνονται λευκό, τότε το ρομπότ προχωράει προς τα εμπρός.



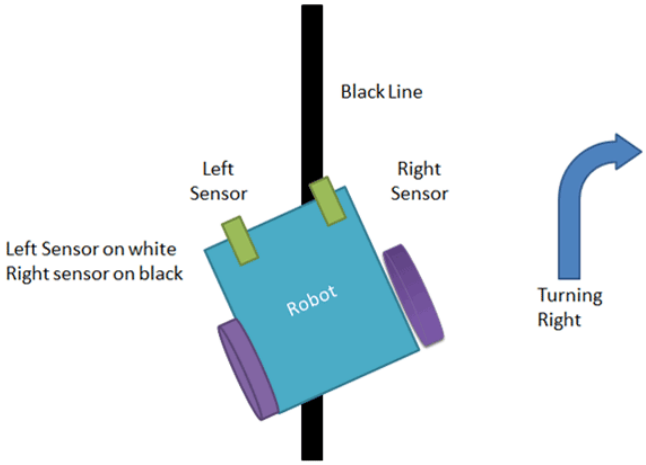
7.11 Κίνηση μπρος τα εμπρός

Εάν ο αριστερός αισθητήρας έρχεται σε μαύρη γραμμή, τότε το ρομπότ στρίβει αριστερά.



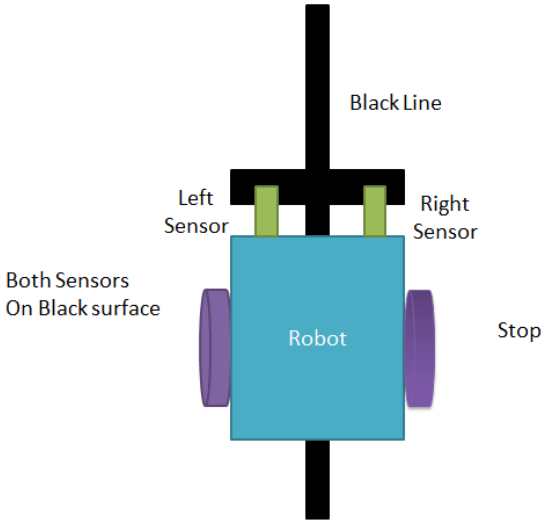
Εικόνα 7.12 Κίνηση αριστερά

Εάν ο δεξιός αισθητήρας αισθάνεται μαύρη γραμμή, τότε το ρομπότ στρίβει δεξιά, μέχρι ο αισθητήρας να έρθει σε λευκή επιφάνεια. Όταν έρχεται η λευκή επιφάνεια, το ρομπότ αρχίζει να κινείται πάλι προς τα εμπρός.



Εικόνα 7.13 Κίνηση δεξιά

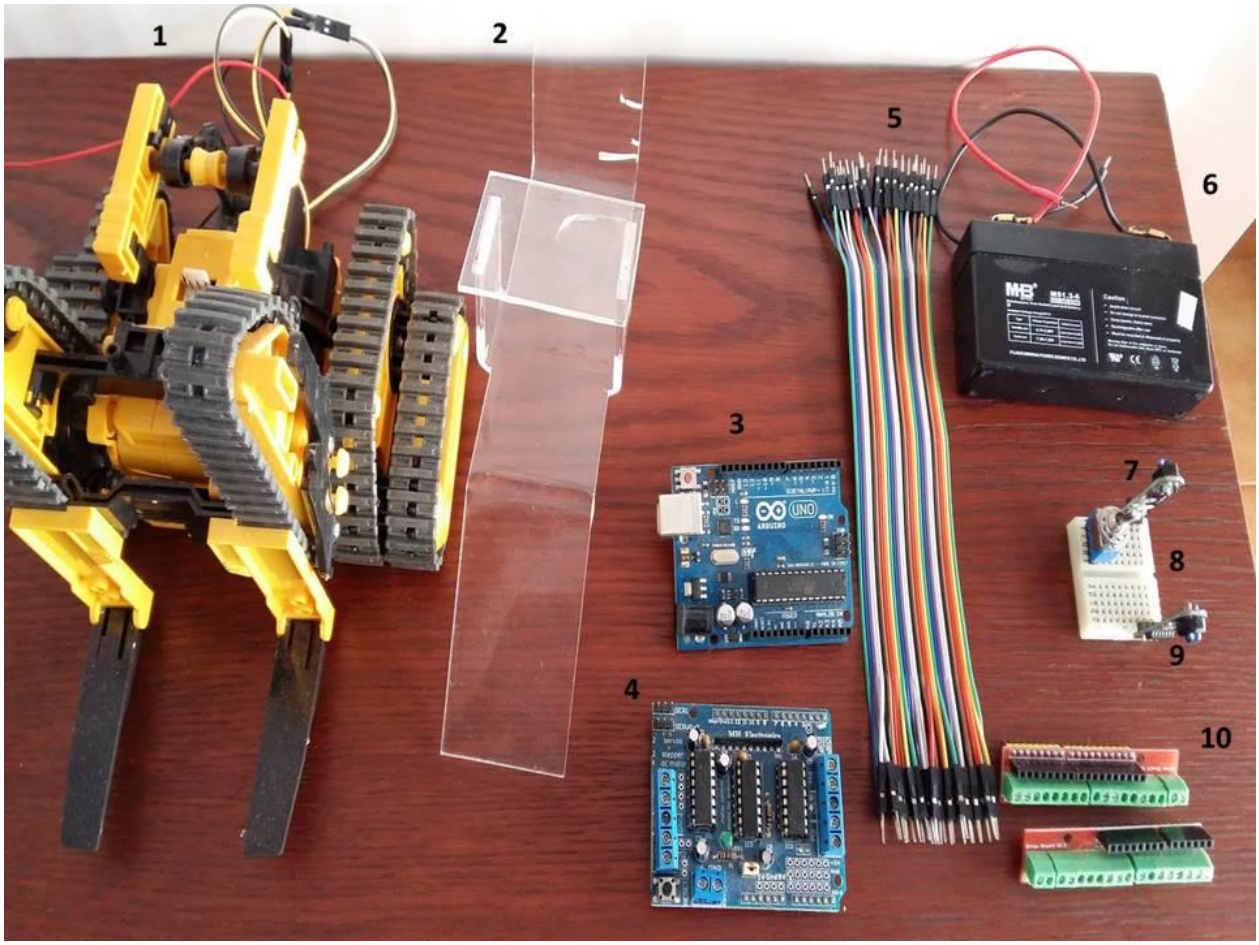
Εάν και οι δύο αισθητήρες έρχονται σε μαύρη γραμμή, το ρομπότ σταματά.



Εικόνα 7.14 Το όχημα σταματάει

7.3.3 Συσκευές και υλικά

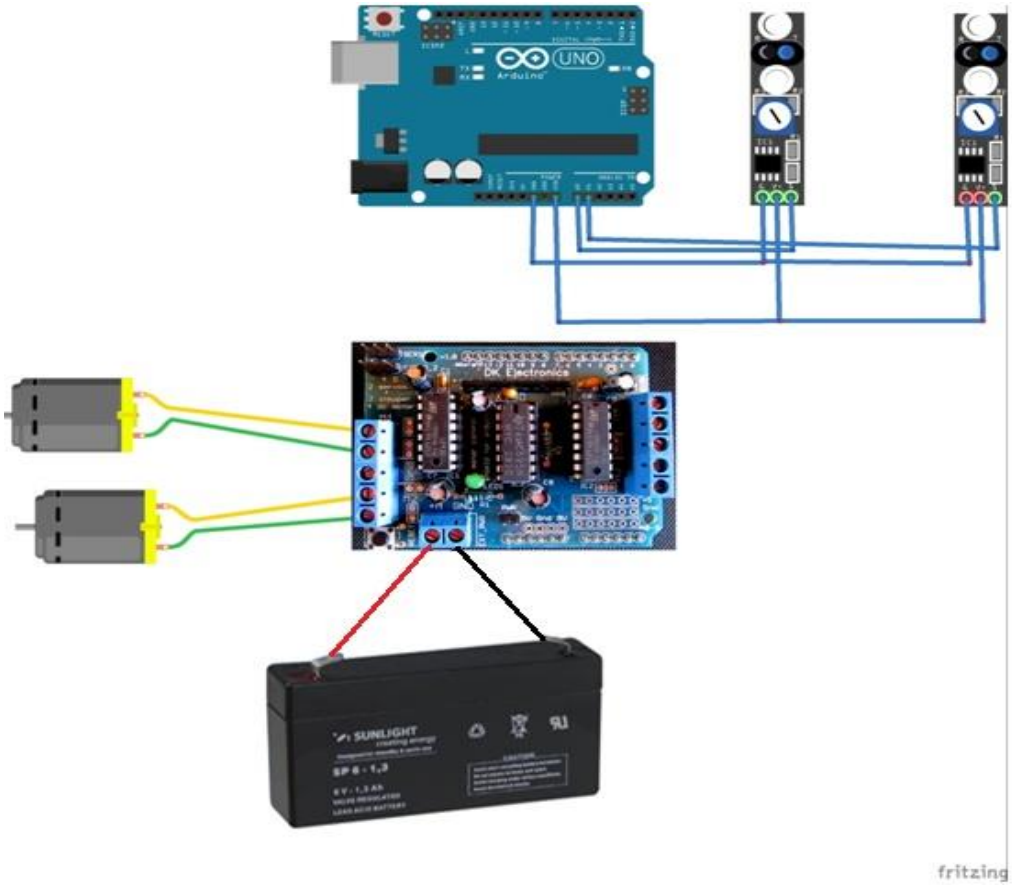
1. Όχημα OWI-536
2. Βάση στήριξης
3. Πλατφόρμα Arduino UNO Rev3
4. Μονάδα τροφοδοσίας κινητήρων (H-Bridge motor driver shield).
5. Καλώδια σύνδεσης
6. Μπαταρία Μολύβδου MHB MS1.3-6 (Πηγή τροφοδοσίας)
7. Διακόπτης
8. Πινακίδες πειραμάτων (breadboard).
9. Αισθητήρας infrared KY033
10. Screw shields



Εικόνα 7.15 Υλικά εφαρμογής 3

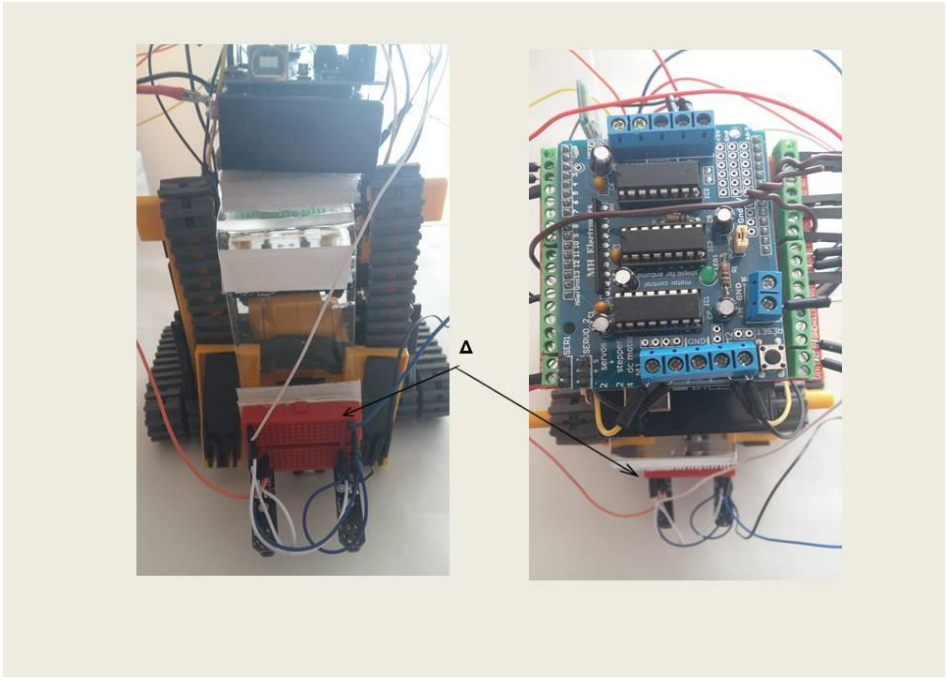
7.3.4 Βήματα εκτέλεσης εργασίας

Για την εφαρμογή 3, ακολουθούμε τα βήματα 1-6 από την πρώτη άσκηση και επιπλέον συνδέουμε το pin signal(S) του αριστερού αισθητηρίου στο digital pin 15(A1) του Arduino και το αντίστοιχο pin του δεξιού αισθητηρίου στο digital pin 14(A0). Συνδέουμε τα pin V+ και G κάθε αισθητηρίου σε οπή 5V(ή VIN) και GND αντίστοιχα.Τα δύο αισθητήρια συνδέονται μέσω ενός breadboard που στο σημείο δηλώνεται στην εικόνα 7.17 (σημείο Δ).



Εικόνα 7.16 Κύκλωμα ρομπότ ακολουθίας γραμμής

Ύστερα από τα παραπάνω,το όχημά μας παίρνει τη μορφή της εικόνας 7.17



Εικόνα 7.17 Εφαρμογή 3

7.3.5 Κώδικας ARDUINO

Ξεκινάμε με την #include που εισάγει την βιβλιοθήκη AFMotor στο σκίτσο μας.

#include <AFMotor.h>

Ο κατασκευαστής AF_DCMotor δημιουργεί 2 κλάσεις DC κινητήρων motor, motor2 στις πόρτες 1,2 της shield με PWM συχνότητα 64 KHZ.Ο τρίτος κινητήρας για το ανυψωτικό δεν χρειάζεται.

AF_DCMotor motor(1, MOTOR12_64KHZ);

AF_DCMotor motor2(2, MOTOR12_64KHZ);

Η εντολή #define μάς επιτρέπει να δώσουμε ένα όνομα σε μία σταθερή τιμή. Στον κώδικά μας, όπου βλέπει LS(Left Sensor), θα το αντικαταστήσει με 15 και όπου βλέπει RS(Right Sensor), με 14.

#define LS 15

#define RS 14

Στη συνάρτηση setup(), γίνεται έναρξη της σειριακής οθόνης με bit ανα δευτερόλεπτο(baud rate) 9600.

Serial.begin(9600);

Η ταχύτητα των κινητήρων ορίζεται με την εντολή setSpeed με τιμές 0-255. Επειδή η απόκριση μοτέρ DC δεν είναι τυπικά γραμμική, και συνεπώς οι πραγματικές RPM δεν είναι απαραιτήτως ανάλογες με την προγραμματισμένη ταχύτητα, ο δεύτερος κινητήρας κινείται με ταχύτητα 177.

motor.setSpeed(255);

motor2.setSpeed(177);

Με την εντολή pinMode, ορίζουμε τα pins του arduino που είναι σε σύνδεση με τα OUT pins των infrared ως είσοδοι εδώ τα pins 15 και 14.

pinMode(LS,INPUT);

pinMode(RS,INPUT);

Στη συνάρτηση loop(), διαβάζουμε το αποτέλεσμα των 2 αισθητηρίων με την digitalRead(). Αν οι αισθητήρες βλέπουν μάυρο, επιστρέφουν λογικό '0' και αν βλέπουν λευκό, λογικό '1'.

Οπότε έχουμε: Όσο και τα δύο αισθητήρια βλέπουν λευκό, το όχημα προχωράει εμπρός.

```
if(digitalRead(LS) && digitalRead(RS))
 {
  //forw
  motor.run(BACKWARD);
  motor2.run(FORWARD);
 }
```

Αν το αριστερό αισθητήριο δει μαύρο, τότε το όχημα στρίβει αριστερά.

```
if(!(digitalRead(LS)) && digitalRead(RS))
 {
   //left
  motor.run(BACKWARD);
  motor2.run(BACKWARD);
 }
```

Αν το δεξιό αισθητήριο δει μαύρο, τότε το όχημα στρίβει δεξιά.

```
if(digitalRead(LS) && !(digitalRead(RS)))
 {
  //right
  motor.run(FORWARD);
  motor2.run(FORWARD);
    }
```

Αν και τα δύο αισθητήρια δουν μαύρο, τότε το όχημα σταματάει.

```
if(!(digitalRead(LS)) && !(digitalRead(RS)))
 {
  //stop
   motor.run(RELEASE);
   motor2.run(RELEASE);      }
```

7.4 ΕΦΑΡΜΟΓΗ 4: Κίνηση ρομποτικού οχήματος (αποφυγή εμποδίων) με χρήση αισθητήρα απόστασης HC-SR04

Σκοπός της εφαρμογής είναι να μπορούν οι σπουδαστές να μετατρέψουν ένα όχημα 3 σε 1 παντός εδάφους (OWI 536) ενσύρματα τηλεκατευθυνόμενο σε πλήρες ρομποτικό όχημα με δυνατότητες έλεγχου μέσω Arduino και αισθητήρων υπερήχων για αποφυγή εμποδίων που εμφανίζονται στην πορεία κίνησής του.

7.4.1 Στόχοι

- Να συνδέουν τα διάφορα απαραίτητα υλικά και συσκευές για τη δημιουργία του ρομποτικού οχήματος.
- Να προγραμματίζουν την πλατφόρμα Arduino για την κίνηση του εκπαιδευτικού οχήματος με τη βοήθεια του αισθητήρα υπερήχων HC-SR04 , ώστε να αποφεύγει τα παρουσιαζόμενα εμπόδια κατά την κίνηση του.

7.4.2 Συσκευές και υλικά

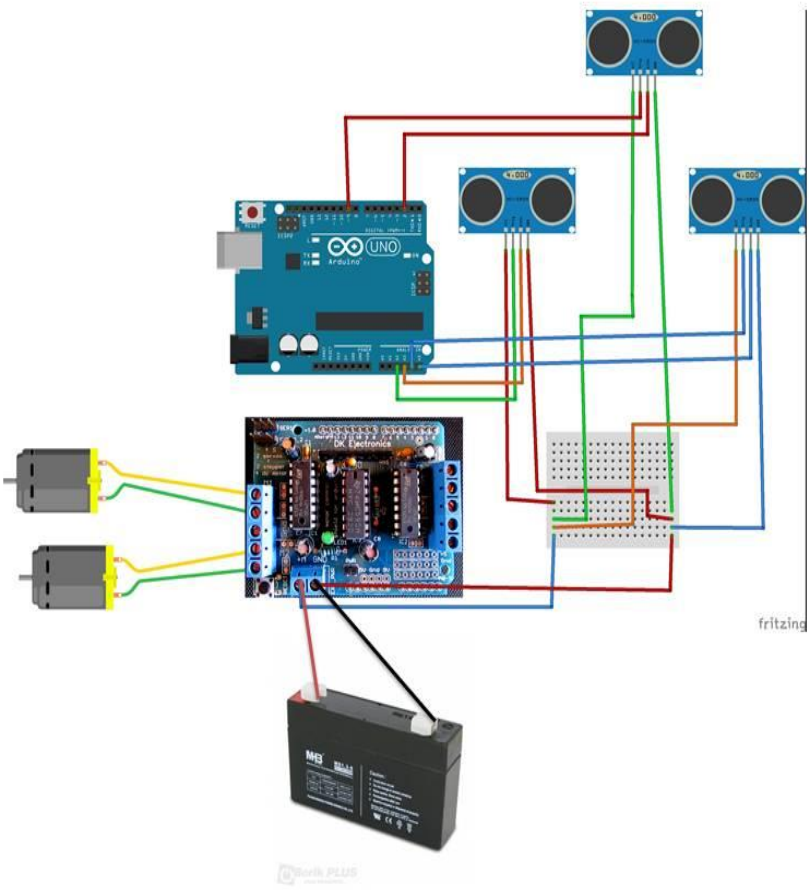
1. Όχημα OWI-536
2. Βάση στήριξης
3. Αισθητήρας HC-SR04
4. Πλατφόρμα Arduino UNO Rev3
5. L293D H-bridge Motor Driver Shield
6. Καλώδια σύνδεσης
7. Μπαταρία Μολύβδου MHB MS1.3-6 (Πηγή τροφοδοσίας)
8. Διακόπτης
9. Πινακίδες πειραμάτων (breadboard)
10. Screw shields

Εικόνα 7.18 Υλικά εφαρμογής 4

7.4.3 Βήματα εκτέλεσης της εργασίας

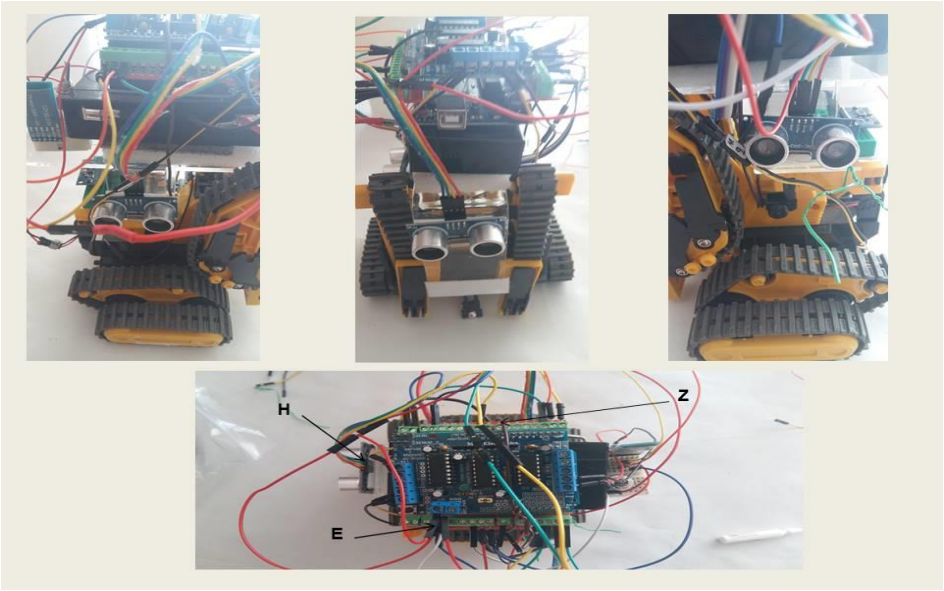
Για την εφαρμογή 4, ακολουθούμε τα βήματα 1-6 από την πρώτη άσκηση και και στη συνέχεια τοποθετούμε 3 breadboards στα σημεία που δηλώνονται στην εικόνα 7.20 (σημεία E,Z,H). Τέλος, συνδέουμε ένα αισθητήριο σε κάθε breadboard ως εξής:

1. Συνδέουμε το trig του αριστερού αισθητηρίου (σημείο E) στο digital pin 16(A2) και το echo στο digital pin 17(A3).
2. Συνδέουμε το trig του δεξιού αισθητηρίου (σημείο Z) στο digital pin 18(A4) και το echo στο digital pin 19(A5).
3. Συνδέουμε το trig του μεσαίου αισθητηρίου (σημείο H) στο digital pin 9 και το echo στο digital pin 2.



Εικόνα 7.19 Κύκλωμα ρομπότ αποφυγής εμποδίων

Ύστερα από τα παραπάνω,το όχημά μας παίρνει τη μορφή της εικόνας 7.20.



Εικόνα 7.20 Εφαρμογή 4

7.4.4 Κώδικας ARDUINO

Ξεκινάμε με την #include που εισάγει την βιβλιοθήκη AFMotor στο σκίτσο μας.

#include <AFMotor.h>

Ο κατασκευαστής AF_DCMotor δημιουργεί 2 κλάσεις DC κινητήρων motor, motor2 στις πόρτες 1,2 της shield με PWM συχνότητα 64 KHZ.Ο τρίτος κινητήρας για το ανυψωτικό δε χρειάζεται.

AF_DCMotor motor(1, MOTOR12_64KHZ);

AF_DCMotor motor2(2, MOTOR12_64KHZ);

Στη συνέχεια, ακολουθεί η δήλωση των μεταβλητών που είναι η διάρκεια που κάνει το σήμα να βρεί ένα σήμα και να επιστρέψει και η απόσταση του σώματος με τα 3 αισθητήρια.

Long frontDuration, frontDistance, leftDuration, leftDistance, rightDuration, rightDistance ;

Η εντολή #define μάς επιτρέπει να δώσουμε ένα όνομα σε μία σταθερή τιμή. Συγκεκριμένα, θα ορίσουμε τα pins του Arduino, με τα οποία θα συνδέσουμε τα trigpin και echopin των τριών αισθητηρίων.

#define trigPinFront 9

#define echoPinFront 2

#define trigPinLeft 16

#define echoPinLeft 17

#define trigPinRight 18

#define echoPinRight 19

Τέλος, λέμε στο όχημα να πάει μπροστά για 500 μς για να βεβαιωθούμε ότι θα ξεκινήσει να πηγαίνει προς τα εμπρός.

motor.run(BACKWARD); //forw

motor2.run(FORWARD);

delay(500);

Στη συνάρτηση setup() γίνεται έναρξη της σειριακής οθόνης με bit ανα δευτερόλεπτο(baud rate) 9600.

Serial.begin(9600);

Η ταχύτητα των κινητήρων ορίζεται με την εντολή setSpeed με τιμες 0-255.Επειδή η απόκριση μοτέρ DC δεν είναι τυπικά γραμμική και συνεπώς οι πραγματικές RPM δεν είναι απαραιτήτως ανάλογες με την προγραμματισμένη ταχύτητα, ο δεύτερος κινητήρας κινείται με τιμή ταχύτητας 177.

motor.setSpeed(255);
motor2.setSpeed(177);

Με την εντολή pinMode ορίζουμε ποια pin θα είναι είσοδοι και πια έξοδοι.Τα trig pin είναι έξοδοι και τα echo pin είσοδοι.

```
pinMode(trigPinFront, OUTPUT);
pinMode(echoPinFront, INPUT);
pinMode(trigPinLeft, OUTPUT);
pinMode(echoPinLeft, INPUT);
pinMode(trigPinRight, OUTPUT);
pinMode(echoPinRight, INPUT);
```

Στη συνάρτηση loop() πρώτα ελέγχουμε την απόσταση από εμπόδιο που δείχνει το μπροστά αισθητήριο με την checkFrontDistance(). Αν είναι μικρότερο των 20 cm, ελέγχει πόση απόσταση από εμπόδιο βρισκονται τα πλάγια αισθητήρια με τις checkLeftDistance() και checkRightDistance().Το όχημα θα στρίψει στην πλευρά που έχει πιο μεγάλη απόσταση από εμπόδιο.Αν το μπροστά αισθητήριο δεν βρίσκει εμπόδιο σε 30 cm, συνεχίζει ευθεία.

```
checkFrontDistance();
  if (frontDistance < 20) {
    Serial.println("Too close");
    checkLeftDistance();
    delay(20);
    checkRightDistance();
    delay(20);
    if (leftDistance < rightDistance){
```

```
motor.run(FORWARD); //right
motor2.run(FORWARD);
delay(750);
 motor.run(BACKWARD); //forw
motor2.run(FORWARD)
} else if (leftDistance >= rightDistance) {
  motor.run(BACKWARD); //left
  motor2.run(BACKWARD);
  delay(750);
   motor.run(BACKWARD); //forw
   motor2.run(FORWARD);
   }
 } else {
Serial.println("OK");
motor.run(BACKWARD); //forw
motor2.run(FORWARD);
  }
```

Οι συναρτήσεις checkFrontDistance(), checkLeftDistance(), checkRight Distance() κάνουν την ίδια δουλειά , βρίσκουν πόση είναι η απόσταση του κάθε αισθητηρίου από το κοντινότερο εμπόδιο. Πρώτα πρέπει να βεβαιωθούμε ότι το trigPin είναι καθαρό, έτσι πρέπει να το ρυθμίσουμε σε κατάσταση LOW για μόλις 2 μs. Για τη δημιουργία του ηχητικού κύματος Ultra, θα πρέπει να ρυθμίσουμε το trigPin στο HIGH State για 10 μs. Αυτό θα στείλει μια ηχητική έκρηξη 8 κύκλων, η οποία θα ταξιδέψει στον ήχο της ταχύτητας και θα ληφθεί στο Echopin .

```
digitalWrite(trigPint, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPinL, LOW);
```

Χρησιμοποιώντας τη λειτουργία pulseIn (), πρέπει να διαβάσουμε τον χρόνο ταξιδιού και να βάλουμε αυτήν την τιμή στη μεταβλητή "duration". Αυτή η

λειτουργία έχει δύο παραμέτρους, η πρώτη είναι το όνομα του echo pin και για τη δεύτερη μπορούμε να γράψουμε είτε HIGH είτε LOW.

Σε αυτή την περίπτωση, το HIGH σημαίνει ότι η συνάρτηση pulseIn () θα περιμένει το pin να γίνει HIGH λόγω του ανακλώμενου ηχητικού κύματος και θα ξεκινήσει τη χρονομέτρηση, μέχρι ο ακροδέκτης γίνει LOW (όταν το ηχητικό κύμα τελειώσει).Αυτή τη στιγμή, θα σταματήσει τη χρονομέτρηση. Στο τέλος, η λειτουργία θα επαναφέρει το μήκος του παλμού σε ms.

Duration = pulseIn(echoPin, HIGH);

Τέλος, πρέπει να μετατρέψουμε τη διάρκεια του παλμού σε απόσταση από το αισθητήριο μέχρι το εμπόδιο.

Ξέρουμε, βέβαια,ότι τα ηχητικά κύματα διαδίδονται στον αέρα με ταχύτητα v=340 m/s (το οποίο θα μετατρέψουμε σε v=0.034 cm/μs). Οπότε σε χρονικό διάστημα t , θα έχει διανύσει απόσταση **s=t*v**.

Επομένως,έχουμε: Distance(s) = Duration*0.034/2 (διαιρούμε με 2 επειδή η διάρκεια αυτή είναι για να πάει και να επιστρέψει ο παλμός, κάνει,δηλαδή την απόσταση δύο φορές σε αυτό το χρόνο). Διαφορετικά, επειδή 1/0.034=29.4, το γράφουμε Distance(s)=(Duration/2)/29.4 .

Distance = (Duration/2) / 29.4;

7.5.Τελικές ρυθμίσεις

Αφού έχουμε ολοκληρώσει τα βήματα που απαιτούνται σε κάθε εφαρμογή, ακολουθούμε την παρακάτω κοινή πορεία εργασίας:συνδέουμε με καλώδιο USB 2.0 Type A σε Type B M/M (1.8m) το Arduino με το PC και μεταφέρουμε τον κώδικα που έχουμε γράψει, πατώντας «Ανέβασμα» (Upload). Αυτό θα μεταγλωττίσει τον κώδικα και μετά θα τον περάσει στο Arduino. Διαφορετικά, μπορούμε πρώτα να πατήσουμε «Επικύρωση» (Compile) για να ελέγχουμε για λάθη. Μετά την επιτυχή μεταφορά του κώδικα, κάνουμε τον έλεγχο λειτουργίας του οχήματος. Το ρομποτικό μας όχημα είναι έτοιμο να εκτελέσει την επιθυμητή ενέργεια.

ΠΑΡΑΡΤΗΜΑ

ΚΩΔΙΚΑΣ ANDROID

Π.1 Το αρχείο activity_main.xml

Εδώ θα γίνει η σχεδίαση της διεπαφής χρήστη της εφαρμογής μας, η οποία χτίζεται με χρήση μιας ιεραρχίας διατάξεων (layout) και widgets. Ένα layout ορίζει τη δομή της διεπαφής χρήστη και συνήθως περιέχει widgets, διάφορες υποκλάσεις που σχεδιάζουν κάτι στην διεπαφή όπως button και textview.

Στην εφαρμογή μας, χρησιμοποιούμε ένα *ConstraintLayout* το οποίο μας επιτρέπει να τοποθετούμε τα widgets στη διεπαφή σε σχέση μεταξύ τους ή με το μητρικό layout. Αυτό επιτυγχάνεται είτε με το να ορίσουμε τα όρια κάθε widget χειροκίνητα πατώντας στην επιλογή design στο κάτω μέρος της οθόνης μας, είτε πηγαίνοντας στον κώδικα κάθε υποκλάσης, για να ορίσουμε τη θέση του αντίστοιχου widget.Η διεπαφή αποτελείται από πέντε buttons (CONN, DISC, +, -, CLEAR), ένα textview και ένα joystick , στο οποίο θα δώσουμε δυνατότητες στην κύρια εφαρμογή.

```
<Button
    android:id="@+id/disc"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/down"
    android:layout_marginEnd="1dp"
    android:layout_marginRight="1dp"
    android:layout_marginTop="16dp"
    android:layout_toEndOf="@+id/textView"
    android:layout_toRightOf="@+id/textView"
    android:onClick="onClickDisc"
    android:text="disc"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<TextView
    android:id="@+id/textView"
    android:layout_width="362dp"
    android:layout_height="199dp"
    android:layout_alignParentTop="true"
    android:layout_marginBottom="1dp"
    android:layout_marginEnd="3dp"
    android:layout_marginLeft="4dp"
    android:layout_marginRight="3dp"
    android:layout_marginStart="4dp"
    android:gravity="bottom"
    android:scrollbars="vertical"
    android:text=""
    app:layout_constraintBottom_toTopOf="@+id/up"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/conn" />
```

```
<Button
    android:id="@+id/conn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_alignTop="@+id/textView"
    android:layout_marginTop="16dp"
    android:onClick="onClickConn"
    android:text="conn"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<Button
    android:id="@+id/down"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="204dp"
    android:text="_"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/disc" />
```

```
<Button
    android:id="@+id/disc"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/down"
    android:layout_marginEnd="1dp"
    android:layout_marginRight="1dp"
    android:layout_marginTop="16dp"
    android:layout_toEndOf="@+id/textView"
    android:layout_toRightOf="@+id/textView"
    android:onClick="onClickDisc"
    android:text="disc"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<TextView
    android:id="@+id/textView"
    android:layout_width="362dp"
    android:layout_height="199dp"
    android:layout_alignParentTop="true"
    android:layout_marginBottom="1dp"
    android:layout_marginEnd="3dp"
    android:layout_marginLeft="4dp"
    android:layout_marginRight="3dp"
    android:layout_marginStart="4dp"
    android:gravity="bottom"
    android:scrollbars="vertical"
    android:text=""
    app:layout_constraintBottom_toTopOf="@+id/up"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/conn" />
```

```
    <Button
        android:id="@+id/clear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="2dp"
        android:layout_marginEnd="61dp"
        android:layout_marginLeft="58dp"
        android:layout_marginRight="61dp"
        android:layout_marginStart="58dp"
        android:layout_marginTop="3dp"
        android:onClick="onClickClear"
        android:text="clear"
        app:layout_constraintBottom_toTopOf="@+id/JOystick"
        app:layout_constraintEnd_toStartOf="@+id/down"
        app:layout_constraintStart_toEndOf="@+id/up"
        app:layout_constraintTop_toBottomOf="@+id/textView" />
</android.support.constraint.ConstraintLayout>
```

Π.2 Το αρχείο AndoidManifest.xml

Ο κώδικας αυτός καθορίζει πότε πρέπει να ξεκινήσει η εφαρμογή, ποια δικαιώματα χρειάζεται και σε ποιο υλικό χρειάζεται να έχει πρόσβαση. Το πρόθετο αρχείο περιγράφει βασικές πληροφορίες σχετικά με την εφαρμογή στα εργαλεία δημιουργίας Android, στο λειτουργικό σύστημα Android και στο Google Play. Η μόνη αλλαγή που κάνουμε από το προκαθοριζόμενο περιεχόμενο του αρχείου είναι η προσθήκη της γραμμής: *<uses-permission android:name="android.permission.BLUETOOTH" />*, η οποία επιτρέπει στην εφαρμογή να χρησιμοποιεί τις δυνατότητες επικοινωνίας Bluetooth της συσκευής.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.robotcontrol">
<uses-permission android:name="android.permission.BLUETOOTH" />
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="RobotControl"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>

</manifest>
```

Π.3 Η κλάση MainActivity.java

Εδώ πηγαίνει ο βασικός κώδικας Java για τη δημιουργία της εφαρμογής, ο κώδικας για τη σύνδεση και αποσύνδεση της κινητής συσκευής με το Bluetooth module του arduino και ο κώδικας για τη λειτουργία των δύο κουμπιών που αφορούν τον έλεγχο κίνησης του τρίτου κινητήρα.

Ξεκινάμε με τον ορισμό του πακέτου που βρίσκεται η κλάση και με την εισαγωγή των απαραίτητων κλάσεων και διασυνδέσεων (interfaces) με την εντολη import.

Στη συνέχεια, ξεκινάει η κλάση μας, η οποία είναι public, δηλαδή είναι ορατή από όλα τα πακέτα. Πρώτα ξεκινάμε με τη δήλωδη των μεταβλητών.Η μέθοδος onCreate καλείται κατά την εκτέλεση της εφαρμογής και εκεί γίνονται όλες οι στατικές αρχικοποιήσεις.Η μέθοδος setUiEnabled μάς δίνει την επιλογή για το πότε και αν τα buttons και το textview θα είναι ενεργά ή όχι.Η μέθοδος BTinit() καλείται όταν πατηθεί το κουμπί CONN. Ελέγχει αν η συσκεύη υποστηρίζει δυνατότητες επικοινωνίας Bluetooth και αν ναι, ενεργοποιεί την επικοινωνία Bluetooth, αν δεν είναι ήδη, και τέλος ψάχνει συσκευή που είναι έτοιμη για σύνδεση με την συσκευή μας.Αυτή η μέθοδος είναι τύπου Boolean, οπότε επιστρέφει true ή false. Η τιμή αποθηκεύεται στη μεταβλητή found, η οποία ξεκινάει ως false και αλλάζει σε true αν βρούμε συσκευή για σύνδεση.

Η μέθοδος BTconnect() καλείται όταν η BTinit() επιστρέψει true. Η *createRfcommSocketToServiceRecord* δημιουργεί ένα Bluetooth socket τύπου RFCOMM για σύνδεση με μία συσκευή με γνωστό UUID(υπηρεσία εγγραφής). Η connect() θα επιχειρήσει σύνδεση με αυτήν τη συσκευή. Αν η σύνδεση ήταν επιτυχής, τότε θα ενεργοποιήσει τις κλάσεις outputStream και inputStream για τη μεταφορά ροής byte προς και από το arduino (μέσω της σειριακής οθόνης).

Το κουμπί CONN, όταν πατηθεί, θα καλέσει τη μέθοδο onClickConn, η οποία, όπως είπαμε παραπάνω, θα καλέσει την BTinit(). Αυτή, αν επιστρέψει true, θα καλέσει την BTconnect() και η δεύτερη με τη σειρά της, αν επιστρέψει true, θα ενεργοποιήσει όλα τα άλλα widgets της διεπαφής (ενώ ταυτόχρονα θα απενεργοποιήσει το κουμπί CONN). Τέλος, θα καλέσει τη μέθοδο *beginListenForData();* για την αποστολή δεδομένων προς το arduino.

Η μέθοδος onClickDisc καλείται όταν πατηθεί το κουμπι DISC και διακόπτει όποια σύνδεση είχε δημιουργηθεί νωρίτερα, ενώ επιτρέπει το πάτημα μόνο του κουμπιού CONN. Η μέθοδος onClickClear καλείται όταν πατηθεί το κουμπί CLEAR και σβήνει το περιεχόμενο του textvie.

Η μέθοδος *touchListener* χρησιμοποιεί ένα γεγονός κίνησης(motion event) που σχετίζεται με τα κουμπιά + και -. Όταν κρατάμε πατημένο ένα από τα δύο πλήκτρα (ACTION_DOWN), στέλνουμε τον αντίστοιχο χαρακτήρα για κίνηση πάνω ή κάτω του κινητήρα του ανυψωτικού στη σειριακή του arduino μέσω της *outputStream.* Όταν αφήσουμε το κουμπί που πατάγαμε(ACTION_UP), στέλνουμε τον χαρακτήρα για σταμάτημα των κινητήρων.

Η μέθοδος beginListenForData(), τέλος, παίρνει τα δεδομένα της σειριακής του arduino και με τη βοήθεια της inputStream τα αποθηκεύει σε ένα buffer και από εκεί τα εμφανίζει στο textview της εφαρμογής.

```
package com.example.robotcontrol;

import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.widget.TextView;
import java.io.InputStream;
import java.util.Set;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;
import android.widget.Button;
import android.widget.Toast;
import java.io.IOException;
import java.io.OutputStream;
import java.util.UUID;

public class MainActivity extends AppCompatActivity
{
    private final String DEVICE_ADDRESS="98:D3:31:FB:71:EF";
    private final UUID PORT_UUID = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb");
    private BluetoothDevice device;
    private BluetoothSocket socket;
    private OutputStream outputStream;
    private static TextView textView;
    private Joystick joystick;
    private OnTouchListener touchListener = (view, motionEvent) → {
            int action = motionEvent.getAction();
            char current = ' ';
            if (action == MotionEvent.ACTION_DOWN) {
                if (view == up) {
                    current = 'u';
                } else if (view == down) {
                    current = 'd';
                }
            }else if (action == MotionEvent.ACTION_UP){
                current = 's';
            }
            try {
                outputStream.write(current);
            } catch (IOException e) {
                e.printStackTrace();
            }
            return false;
    };
```

```
    Button conn,disc,up,down,clear;
    boolean deviceConnected=false;
    boolean stopThread;
    byte buffer[];
    private InputStream inputStream;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        textView = (TextView) findViewById(R.id.textView);
        textView.setMovementMethod(new ScrollingMovementMethod());
        conn = (Button) findViewById(R.id.conn);
        disc = (Button) findViewById(R.id.disc);
        up = (Button) findViewById(R.id.up);
        down = (Button) findViewById(R.id.down);
        clear = (Button) findViewById(R.id.clear);
        joystick = (Joystick) findViewById(R.id.JOystick);
        setUiEnabled(false);
        up.setOnTouchListener(touchListener);
        down.setOnTouchListener(touchListener);
    }
    public void setUiEnabled(boolean bool)
    {
        conn.setEnabled(!bool);
        disc.setEnabled(bool);
        up.setEnabled(bool);
        down.setEnabled(bool);
        clear.setEnabled(bool);
        textView.setEnabled(bool);
    }
```

```
public boolean BTinit()
{
    boolean found=false;
    BluetoothAdapter bluetoothAdapter=BluetoothAdapter.getDefaultAdapter();
    if (bluetoothAdapter == null) {
        Toast.makeText(getApplicationContext(), text: "Device doesnt Support Bluetooth",Toast.LENGTH_SHORT).show();
    }
    if(!bluetoothAdapter.isEnabled())
    {
        Intent enableAdapter = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableAdapter, requestCode: 0);
        try {
            Thread.sleep( millis: 1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
```

```
    Set<BluetoothDevice> bondedDevices = bluetoothAdapter.getBondedDevices();
    if(bondedDevices.isEmpty())
    {
        Toast.makeText(getApplicationContext(), text: "Please Pair the Device first",Toast.LENGTH_SHORT).show();
    }
    else
    {
        for (BluetoothDevice iterator : bondedDevices)
        {
            if(iterator.getAddress().equals(DEVICE_ADDRESS))
            {
                device=iterator;
                found=true;
                break;
            }
        }
    }
    return found;
}
```

```
public boolean BTconnect()
{
    boolean connected=true;
    try {
        socket = device.createRfcommSocketToServiceRecord(PORT_UUID);
        socket.connect();
    } catch (IOException e) {
        e.printStackTrace();
        connected=false;
    }
    if(connected)
    {
        try {
            outputStream=socket.getOutputStream();
            joystick.setOutputStream(outputStream);
        } catch (IOException e) {
            e.printStackTrace();
        }
        try {
            inputStream=socket.getInputStream();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    return connected;
}
```

```
public void onClickConn(View view) {
    if(BTinit())
    {
        if(BTconnect())
        {
            setUiEnabled(true);
            conn.setEnabled(false);
            beginListenForData();
            deviceConnected=true;
            Toast.makeText(getApplicationContext(), text: "Connection Opened!",Toast.LENGTH_SHORT).show();
        }
    }
}
public void onClickDisc(View view) throws IOException {
    stopThread = true;
    outputStream.close();
    socket.close();
    setUiEnabled(false);
    conn.setEnabled(true);
    inputStream.close();
    deviceConnected=false;
    Toast.makeText(getApplicationContext(), text: "Connection Closed!",Toast.LENGTH_SHORT).show();
    joystick.setOutputStream(null);
    textView.setText("");
}
public void onClickClear(View view){
    textView.setText("");
}
```

```
 void beginListenForData()
 {
     final Handler handler = new Handler();
     stopThread = false;
     buffer = new byte[1024];
     Thread thread  = new Thread((Runnable) () → {
             while(!Thread.currentThread().isInterrupted() && !stopThread)
             {
                 try
                 {
                     int byteCount = inputStream.available();
                     if(byteCount > 0)
                     {
                         byte[] rawBytes = new byte[byteCount];
                         inputStream.read(rawBytes);
                         final String string=new String(rawBytes, charsetName: "UTF-8");
                         handler.post(() → {
                                 textView.append(string);
                         });
                     }
                 }
                 catch (IOException ex)
                 {
                     stopThread = true;
                 }
             }
     });
     thread.start();
 }
}
```

Π.4 Η κλάση Joystick.java

Εδώ σχεδιάζουμε το joystick και του δίνουμε δυνατότητες έλεγχου κίνησης του οχήματος.

Ξεκινάμε με τον ορισμό του πακέτου που βρίσκεται η κλάση και με την εισαγωγή των απαραίτητων κλάσεων και διασυνδέσεων (interfaces) με την εντολη import.

Στη συνέχεια, ξεκινάει η κλάση Joystick.Ξεκινάμε με τη δήλωση των μεταβλητών. Η surfaceCreated είναι μία μέθδος που καλείται όταν δημιουργείται το joystick. Καλεί τη setupDimensions() και την drawJoystick(). Η πρώτη ρυθμίζει τις διαστάσεις των 2 επιπέδων του joystick και τη θέση του κέντρου τους.Η δεύτερη σχεδιάζει το joystick με τη χρήση ενός αντικειμένου canvas.Ορίζουμε και ένα αντικείμενο Paint για τον ορισμό των χρωμάτων.Με την myCanvas.drawColor, ετοιμαζόμαστε να ορίσουμε ένα χρώμα για σχεδίαση και με την colors.setARGB, ορίζουμε ποιο χρώμα θέλουμε. Με την myCanvas.drawCircle, σχεδιάζουμε τον κύκλο με παραμέτρους τις συντεταγμένες του κέντρου, την ακτίνα του κύκλου και το χρώμα του. Αυτήν την διαδικασία την κάνουμε άλλη μία φορά και για τον εξωτερικό κύκλο.

Η μέθοδος *onTouch* δίνει δυνατότητες ελέγχου του οχήματος στο joystick. Πρώτα πρέπει να βεβαιωθούμε ότι το joystick δεν ξεπερνάει τα όρια του εξωτερικού κύκλου. Για να το πετύχουμε αυτό, υπολογίζουμε τη θέση του joystick στη μεταβλητή displacement και την συγκρίνουμε με την ακτίνα του κύκλου. Αν ξεπερνάει το όριο, δεν αφήνουμε το joystick να σχεδιαστεί πέρα από αυτό.

Στη συνέχεια, υπολογίζουμε τις συντεταγμένες του joystick στις xCoord και yCoord και βρίσκουμε τη γωνία στην οποία βρίσκεται το joystick από τον τύπο atan2(x,y)*180/π και το αποθηκεύουμε στην μεταβλητή angle. Ανάλογα με την γωνία, στέλνουμε μέσω της outputstream στη σειριακή του arduino το ψηφίο για την κατάλληλη κίνηση του οχήματος. Αυτό συμβαίνει όσο κρατάμε πατημένο το joystick. Όταν αφήσουμε το joystick, αυτό επιτρέφει στην αρχική του θέση στο κέντρο του κύκλου και στέλνουμε στο arduino τον χαρακτήρα παύσης όλων των κινητήρων του οχήματος.

```
package com.example.robotcontrol;

        import android.content.Context;
        import android.graphics.Canvas;
        import android.graphics.Color;
        import android.graphics.Paint;
        import android.util.AttributeSet;
        import android.view.MotionEvent;
        import android.view.SurfaceHolder;
        import android.view.SurfaceView;
        import android.view.View;
        import java.io.IOException;
        import java.io.OutputStream;

public class Joystick extends SurfaceView implements SurfaceHolder.Callback, View.OnTouchListener
{
    private float centerX, centerY, baseRadius, hatRadius, xCoord, yCoord;
    private double angle;
    private OutputStream outputStream;
    private char lastsent=' ';

    public void setOutputStream(OutputStream outputStream) {
        this.outputStream = outputStream;
    }
    private void setupDimensions()
    {
        centerX=getWidth()/2;
        centerY=getHeight()/2;
        baseRadius=Math.min(getWidth(), getHeight() / 3);
        hatRadius=Math.min(getWidth(), getHeight() / 5);
    }
    public Joystick(Context context)
    {
        super(context);
        getHolder().addCallback(this);
        setOnTouchListener(this);
    }
    public Joystick(Context context, AttributeSet attributes, int style)
    {
        super(context, attributes, style);
        getHolder().addCallback(this);
        setOnTouchListener(this);
    }
    public Joystick(Context context, AttributeSet attributes)
    {
        super(context, attributes);
        getHolder().addCallback(this);
        setOnTouchListener(this);
    }
```

```
private void drawJoystick(float newX, float newY)
{
    if(getHolder().getSurface().isValid())
    {
        Canvas myCanvas = this.getHolder().lockCanvas();
        Paint colors= new Paint();
        myCanvas.drawColor(Color.WHITE);
        colors.setARGB( a: 255, r: 50, g: 50, b: 50);
        myCanvas.drawCircle(centerX, centerY, baseRadius, colors);
        colors.setARGB( a: 255, r: 0, g: 0, b: 255);
        myCanvas.drawCircle(newX, newY, hatRadius, colors);
        getHolder().unlockCanvasAndPost(myCanvas);
    }
}

@Override
public void surfaceCreated(SurfaceHolder holder)
{
    setupDimensions();
    drawJoystick(centerX, centerY);
}
@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width, int height)
{

}
@Override
public void surfaceDestroyed(SurfaceHolder holder)
{

}
```

```
public boolean onTouch(View v, MotionEvent e) {
    if (outputStream != null) {
        if (v.equals(this)) {
            if (e.getAction() != e.ACTION_UP) {
                float displacement = (float) Math.sqrt((Math.pow(e.getX() - centerX, 2)) + Math.pow(e.getY() - cente
                if (displacement < baseRadius) {
                    drawJoystick(e.getX(), e.getY());
                } else {
                    float ratio = baseRadius / displacement;
                    float constrainedX = centerX + (e.getX() - centerX) * ratio;
                    float constrainedY = centerY + (e.getY() - centerY) * ratio;
                    drawJoystick(constrainedX, constrainedY);
                }
                xCoord = e.getX() - centerX;
                yCoord = centerY - e.getY();
                angle = Math.atan2(yCoord, xCoord) * 180 / Math.PI;
                char current = 's';
                if ((angle <= 22.5 && angle > 0) || (angle >= -22.5 && angle < 0)) {
                    current = 'q';
                } else if (angle > 22.5 && angle <= 67.5) {
                    current = 'r';
                } else if (angle > 67.5 && angle <= 112.5) {
                    current = 'f';
                } else if (angle > 112.5 && angle <= 157.5) {
                    current = 'l';
                } else if ((angle > 157.5 && angle <= 180) || (angle <= -157.5)) {
                    current = 'w';
                } else if (angle > -157.5 && angle <= -112.5) {
                    current = 'j';
```

```
            } else if (angle > -112.5 && angle <= -67.5) {
                current = 'b';
            } else if (angle > -67.5 && angle < -22.5) {
                current = 'k';
            }
            try {
                if(current != lastsent) {
                    outputStream.write(current);
                    lastsent = current;
                }
            } catch (IOException e1) {
                e1.printStackTrace();
            }
        } else {
            drawJoystick(centerX, centerY);
            xCoord = 0;
            yCoord = 0;
            try {
                outputStream.write( b: 's');
                lastsent = 's';
            } catch (IOException ex) {
                ex.printStackTrace();
                    }
                }
            }
        }
        return true;
    }
}
```

ΒΙΒΛΙΟΓΡΑΦΙΑ ΚΑΙ ΑΝΑΦΟΡΕΣ

[1] Ηλεκτρολογία, Βιδιαδάκης Α κτλ, Γ' ΓΕ.Λ. ΟΕΔΒ, 1999

[2] Ανάπτυξη εφαρμογών με το Arduino Παπάζογλου Π., Λιώνης Σ.-Π., Εκδ.Τζιόλα, 2015

[3] Ηλεκτρικές Μηχανές, Champan Stephen J., Εκδόσεις Τζιόλα, 2013

[4] https://www.indiamart.com/proddetail/ac-rotor-coil-15378681491.html

[5] https://www.indiamart.com/ts-electricalworks/motor-rewinding-service.html

[6] https://www.skf.com/binary/124-295519/INSOCOAT_SKF.jpg

[7]https://www.emartee.com/product/41424/24BYJ%2048%20High%20Quality%20 Stepper %20Motor%2012V

[8] https://www.instructables.com/id/Servo-Motor-Control-by-using-Microcontroller-PIC16/

[9] https://store.arduino.cc/arduino-uno-rev3

[10] https://www.rapidonline.com/pdf/73-4443.pdf

[11] https://www.instructables.com/id/Hardware-Structure-of-ARDUINO-UNO/

[12] https://www.allaboutcircuits.com/technical-articles/understanding-arduino-uno-hardware-design/

[13] https://www.arduino.cc/en/Guide/Introduction

[14] http://www.circuitbasics.com/arduino-basics-installing-software/

[15] https://www.arduino.cc/en/Reference/Libraries

[16] https://www.arduino.cc/en/Guide/Libraries

[17] Σημειώσεις Σ.Α.Μ. ΤΕΙ Πειραιά, Ιωάννη Έλληνα. Σεπτέμβριος 2016

[18] https://www.eltima.com/arduino-serial-monitor-alternative/

[19] https://makeradvisor.com/top-10-most-useful-arduino-shields/

[20] https://github.com/RIOT-OS/RIOT/wiki/Board:-Arduino-Uno

[21] https://deltahacker.gr/arduino-intro/

[22] https://www.symbols.com/symbol/android-logo

[23] http://www.codekul.com/blog/know-history-android-versions/

[24] https://www.techopedia.com/definition/24407/application-programming-interface-api

[25] https://www.javatpoint.com/api-full-form

[26] https://www.quora.com/What-is-the-architecture-of-an-android-app

[27] https://developer.android.com/guide/platform/
[28] https://www.tutorialspoint.com/android/android_architecture.htm
[29] https://www.techotopia.com/index.php/An_Overview_of_the_Android _ Architecture
[30] https://www.androidauthority.com/android-studio-tutorial-beginners-637572/
[31] https://www.quora.com/What-is-an-APK-file
[32] https://developer.android.com/studio/intro/
[33] https://developer.android.com/studio/run/device
[34] https://developer.android.com/studio/run/
[35] https://www.tenorshare.com/android-data/enable-usb-debugging-android-7-nougat.html
[36] https://www.tutorialspoint.com/android/android_hello_world_example.htm
[37] https://learn.sparkfun.com/tutorials/bluetooth-basics
[38] https://medium.com/jaycon-systems/bluetooth-technology-what-has-changed-over-the-years-385da7ec7154
[39] Δίκτυα υπολογιστών, Tanenbaum Andrew S., Wetherall David J. Εκδ. Κλειδάριθμος, 2011
[40] https://circuitdigest.com/microcontroller-projects/line-follower-robot-using -arduino
[41] http://sensorkit.en.joy-it.net/index.php?title=KY-033_Tracking_sensor_module
[42] https://www.tindie.com/products/ICStation/ky-033-infrared-tracking-sensor -for-arduino2787/
[43] https://pixers.us/posters/visible-and-invisible-light-59783212
[44] https://www.tindie.com/products/andygrove73/hc-sr04-ultrasonic-sensor -pack-of-8/
[45] https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc -sr04/
[46] https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/
[47] https://hobbycomponents.com/sensors/720-mpu9250-9-axis-9dof-acc-gyro -compass-module
[48] https://www.hotmcu.com/9dof-imu-module-with-mpu9250-p-172.html
[49] https://www.batteries.gr/gr/mhb-ms1-3-6-genikis-xrisis.html
[50] https://www.amazon.com/Other-Tools-Position-Off-Waterproof/dp/B0762FP5 M4

[51] https://components101.com/wireless/hc-05-bluetooth-module
[52] https://wiki.eprolabs.com/index.php?title=L293D_H-bridge_Motor_Driver _ Shield
[53] https://www.cableworks.gr/ilektronika/arduino-and-microcontrollers/prototy ping/proto - screw-shield-expansion-board- module-for-arduino/
[54] https://www.seeedstudio.com/Screw-Shield-p-1238.html
[55] https://www.sciencebuddies.org/science-fair-projects/references/how-to-use- a-breadboard
[56] https://shop.rasp.io/products/half-size-breadboard-for-electronics-and-proto typing- raspberry-pi
[57] https://grobotronics.com/prototyping/jumper-wires/
[58] https://www.amazon.com/OWI-536-All-Terrain-Robot-Kit/dp/B004P4WTB0
[59] https://cdn-learn.adafruit.com/downloads/pdf/adafruit-motor-shield.pdf
[60] https://www.instructables.com/id/Arduino-bot-Android-remote-control/
[61] https://www.arduino.cc/en/Reference/Wire
[62] https://playground.arduino.cc/Main/MPU-6050
[63] https://electronics.stackexchange.com/questions/39714/how-to-read-a-gyro-accelerometer
[64] https://www.invensense.com/wp-content/uploads/2015/02/MPU-9250-Re gister-Map.pdf
[65] https://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01 -v 1.1.pdf
[66] https://www.arduino.cc/en/Tutorial/BlinkWithoutDelay
[67] https://create.arduino.cc/projecthub/jrance/arduino-obstacle-avoidance-robot-with-ultrasonic-hc-sr04-23035d
[68] https://www.allaboutcircuits.com/projects/communicate-with-your-arduino -through-android/
[69] https://www.allaboutcircuits.com/projects/control-an-arduino-using-your -phone/
[70] https://www.instructables.com/id/A-Simple-Android-UI-Joystick/