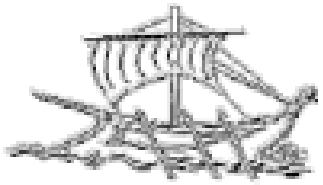


ΑΕΙ ΠΕΙΡΑΙΑ Τ.Τ.
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Έλεγχος ρομποτικού οχήματος με Arduino Uno

Ιωάννης Μπέλεσης

Εισηγήτρια: Δρ Αναστασία Βελώνη, Καθηγήτρια

ΑΘΗΝΑ

ΙΟΥΝΙΟΣ 2017

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Έλεγχος ρομποτικού οχήματος με Arduino Uno

Ιωάννης Μπέλεσης
Α.Μ:35570

Εισηγήτρια:

Δρ Αναστασία Βελώνη, Καθηγήτρια

Εξεταστική Επιτροπή:

Έλληνας Ιωάννης, Καθηγητής
Μυριδάκης Νικόλαος, Καθηγητής

Ημερομηνία εξέτασης 16/06/2017

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Μπέλεσης Ιωάννης, του Απόστολου, με αριθμό μητρώου 35570 φοιτητής του Τμήματος Μηχανικών Η/Υ Συστημάτων Τ.Ε. του Α.Ε.Ι. Πειραιά Τ.Τ. 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 ενδιαφερομένου, του αναθέτει εκ νέου την εκπόνηση 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 άρθρου 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία ολοκληρώθηκε μετά από επίπονες προσπάθειες, σε ένα ενδιαφέρον γνωστικό αντικείμενο, όπως αυτό της ελεγχόμενης κίνησης ρομποτικού οχήματος με Arduino Uno. Την προσπάθεια μου αυτή υποστήριξε η επιβλέπουσα καθηγήτριά μου, την οποία θα ήθελα να ευχαριστήσω.

Ακόμα θα ήθελα να ευχαριστήσω και την οικογένειά μου για τη στήριξή της όλα αυτά τα χρόνια.

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία βασίζεται στο Arduino Uno κι ασχολείται με την κατασκευή ρομποτικού οχήματος που να κινείται μόνο του στο χώρο, να ελέγχεται η κίνησή του με την τεχνολογία Bluetooth μέσω android εφαρμογής, να εκτελεί παράλληλα και τη χειροκίνητη και την αυτόματη λειτουργία και να κινείται προς όλες τις κατευθύνσεις με φωνητικές εντολές μέσω Bluetooth με τη χρήση android εφαρμογής.

Στην αρχή γίνεται μια εισαγωγή στο Arduino για τον τρόπο λειτουργίας του και τις δυνατότητες και τα πλεονεκτήματα που έχει, ακολουθεί μια αναφορά στο υλικό (hardware) και λογισμικό (software) του Arduino και στη συνέχεια γίνεται μια ανάλυση για τους ακροδέκτες (pins) του Arduino Uno για να κατανοηθεί η λειτουργία του.

Στη συνέχεια περιγράφονται τα βήματα υλοποίησης της κατασκευής του ρομποτικού οχήματος καθώς και τα υλικά και τα εργαλεία που θα χρειαστούν για την ολοκλήρωση της κατασκευής. Μετά αναλύεται με κάθε λεπτομέρεια η συναρμολόγηση των εξαρτημάτων του ρομποτικού οχήματος καθώς και η συνδεσμολογία τους. Επίσης γίνεται μια αναλυτική περιγραφή της λειτουργίας του κάθε εξαρτήματος και του ρόλου του στο ρομποτικό όχημα.

Στο τέλος αναλύονται τα βήματα προγραμματισμού του Arduino Uno για την ελεγχόμενη κίνηση του ρομποτικού μας οχήματος μέσω android συσκευών (smartphone ή tablet) και γράφονται οι κώδικες με τους οποίους θα προγραμματίσουμε το Arduino Uno σε σύνδεση με τον υπολογιστή μέσω του προγράμματος Arduino IDE.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Αρχιτεκτονική Μικροελεγκτή ATmega328 του Arduino

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Arduino, Bluetooth, Android,Ακροδέκτης,Ρομποτικό Όχημα

ABSTRACT

This thesis is based on the Arduino Uno and engaged in manufacturing robotic vehicle to move in space by itself, to control its movement with Bluetooth technology through android application, to perform parallel manual and automatic mode and finally moves with voice commands with Bluetooth through android application.

In the first place, an introduction to Arduino is made about how it works, its features and advantages, following by a reference to Arduino hardware and software. Then, an analysis for the Arduino Uno pins is made in order to understand its function.

Subsequently, we report the steps of implementing the construction of the robotic vehicle as well as the materials and tools that will be needed to complete the construction. The assembly of the robotic vehicle components as well as their connections are analyzed in detail. A detailed description on the function of each component and its role on the robotic vehicle is also made.

In the end, we analyze the Arduino Uno programming steps for the controlled movement of our robotic vehicle through android devices (smartphone or tablet) and write the codes that will program the Arduino Uno in connection with the computer through the Arduino IDE program.

SCIENTIFIC AREA: Arduino ATmega328 Microcontroller Architecture

KEYWORDS: Arduino, Bluetooth, Android,Pin,Robotic Vehicle

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

GND Ground

PWM Pulse Width Modulation

DC Direct Current

NC Normally Close

NO Normally Open

COM Common

V Voltage

BT Bluetooth

ADC Analog To Digital Converter

TRIG Trigger

USB Universal Serial Bus

CONST Constant

I/O Input/Output

IN Input

INT Integer

MM MilliMeter

CM CentiMeter

S Second

MS MilliSecond

HZ Hertz

KHZ KiloHertz

MHZ MegaHertz

KOHM KiloOhm

A Ampere

MA MilliAmpere

WPAN Wireless Personal Area Networks

PDA Personal Digital Assistant

MAC Media Access Control

LLC Logical Link Control

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο γίνεται μια εισαγωγή στο Arduino για τον τρόπο λειτουργίας του και τις δυνατότητες και τα πλεονεκτήματα που έχει, ακολουθεί μια αναφορά στο υλικό (hardware) και λογισμικό (software) του Arduino και στη συνέχεια γίνεται μια ανάλυση για τους ακροδέκτες (pins) του Arduino Uno.

1.1 Περιγραφή του αντικειμένου της πτυχιακής εργασίας

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η υλοποίηση της κατασκευής ενός ρομποτικού οχήματος το οποίο αναπτύχθηκε με το Arduino Uno και μπορεί να κινείται μόνο του στο χώρο, να ελέγχεται η κίνησή του από το χρήστη μέσω android εφαρμογής χρησιμοποιώντας την τεχνολογία bluetooth, να εκτελεί και τις 2 λειτουργίες παράλληλα, δηλαδή και να κινείται μόνο του ελεύθερα στο χώρο και να ελέγχεται από το χρήστη μέσω της android εφαρμογής κι ακόμα να κινείται μέσω φωνητικών εντολών με τη χρήση android εφαρμογής και την τεχνολογία bluetooth.

1.2 Εισαγωγή στο Arduino

Σε αυτή την κατηγορία θα ασχοληθούμε με το Arduino (Εικόνα 1.1). Το Arduino είναι ένα εργαλείο που μπορεί να αντιληφθεί περισσότερα από τον υλικό κόσμο απ' ότι μπορεί ο ηλεκτρονικός υπολογιστής.

Εικόνα 1.1: ARDUINO

Αυτή η διαδραστικότητα που έχει με το φυσικό περιβάλλον είναι αυτό που το ξεχωρίζει από τους απλούς υπολογιστές. Είναι μια ανοικτού κώδικα υπολογιστική πλατφόρμα βασισμένη σε μια απλή πλακέτα με εισόδους/εξόδους, έναν μικροελεγκτή και ένα προγραμματιστικό περιβάλλον για το γράψιμο του κώδικα.

Στην ουσία πρόκειται για ένα ηλεκτρονικό κύκλωμα που βασίζεται στον μικροελεγκτή ATmega της Atmel. Το Arduino διαθέτει σειριακό interface. Ο μικροελεγκτής ATmega υποστηρίζει σειριακή επικοινωνία, την οποία το Arduino προωθεί μέσα από έναν ελεγκτή Serial-over-USB ώστε να συνδέεται με τον υπολογιστή μέσω usb. Η σύνδεση αυτή χρησιμοποιείται για την μεταφορά προγραμμάτων που σχεδιάζονται από τον υπολογιστή στο Arduino αλλά και για αμφίδρομη επικοινωνία του Arduino με τον υπολογιστή μέσα από το πρόγραμμα την ώρα που εκτελείται.

Η γλώσσα προγραμματισμού του Arduino είναι η Wiring C, η οποία στην ουσία είναι η C++ τροποποιημένη και προσαρμοσμένη στις ανάγκες του, ενώ μέσα από τις βιβλιοθήκες του δημιουργούνται καθημερινά περισσότερες λειτουργίες και δυνατότητες.

Το Arduino μπορεί να χρησιμοποιηθεί για να αναπτύξει διαδραστικά αντικείμενα, εισάγοντας δεδομένα από ένα πλήθος διακοπτών ή αισθητήρων και να ελέγχει ένα πλήθος από λαμπάκια, ηλεκτροκινητήρες και άλλα φυσικά αντικείμενα. Τα projects (οι κατασκευές) του μπορούν να είναι αυτόνομα ή μπορούν να επικοινωνούν με κάποιο πρόγραμμα που τρέχει στον προσωπικό μας υπολογιστή (πχ flash,processing). Οι πλατφόρμες μπορούν να συναρμολογηθούν στο χέρι ή να αγοραστούν έτοιμες.

1.3 Τα πλεονεκτήματα του Arduino

- **Το Χαμηλό κόστος**

Η τιμή κόστους μιας πλακέτας Arduino είναι αρκετά προσιτή. Πιο συγκεκριμένα το Arduino Uno (Εικόνα 1.2) που είναι η ναυαρχίδα της Arduino μπορεί να αγοραστεί μέσω διαδικτύου από κάποιο ηλεκτρονικό κατάστημα σε χαμηλότερη τιμή από αυτή του φυσικού καταστήματος. Επίσης υπάρχουν Arduino Starter Kits τα οποία εκτός από την ίδια πλακέτα του Arduino Uno περιέχουν διάφορα άλλα εξαρτήματα και εργαλεία που μπορεί να χρειαστούν οι χρήστες για τις πρώτες τους εφαρμογές (όπως το απαραίτητο καλώδιο usb για την σύνδεση με τον

υπολογιστή, rasters, καλώδια, leds, διακόπτες, ποτενσιόμετρα, αντιστάσεις, διόδους, τρανζίστορ κλπ).

Εικόνα 1.2: Arduino Uno Breadboard

- **Είναι Ανεξαρτήτου πλατφόρμας (cross-platform)**

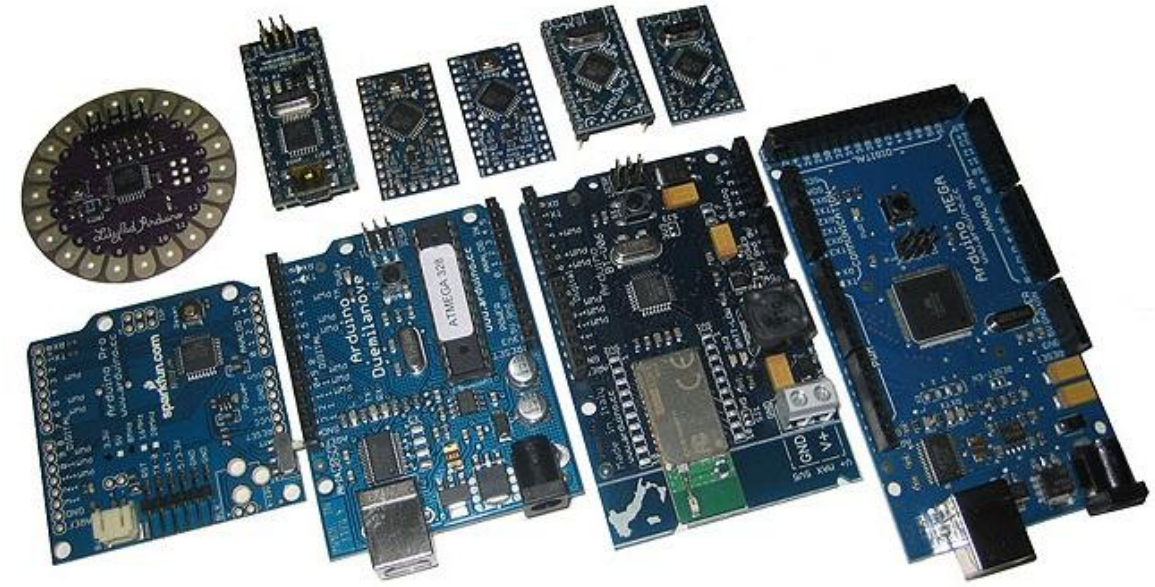
Το πρόγραμμα του Arduino εκτελείται και στα τρία λειτουργικά συστήματα (Windows, Mac OS, Linux) εξυπηρετώντας όλο το εύρος των χρηστών προσωπικών υπολογιστών.

- **Η Απλότητα του**

Ίσως το πιο σημαντικό πλεονέκτημα του Arduino είναι η απλότητά του. Μέσα σε σύντομο χρονικό διάστημα ο χρήστης μπορεί να δημιουργήσει την πρώτη του κατασκευή. Αποτελεί ιδανικό δημιουργικό εργαλείο για την απόκτηση ηλεκτρονικών και μηχανικών δεξιοτήτων καθώς επίσης και δημιουργική απασχόληση για τους χρήστες.

- **Οι εκδόσεις – πλακέτες (breadboards) του**

Η Arduino είναι μια οικογένεια από πλακέτες μικροελεγκτών (Εικόνα 1.3) που σκοπό έχουν να κάνουν ευκολότερη την κατασκευή διαδραστικών αντικειμένων. Κάθε έκδοση καλύπτει διαφορετικές ανάγκες και έχει διαφορετικές δυνατότητες (Duemilanove, Diecimila, Nano, Mega, Bluetooth, LilyPad, Mini, Mini USB, Pro, Pro Mini, Serial και Serial SS). Όπως γίνεται κατανοητό πάντα θα υπάρχουν διάφορες εκδόσεις του Arduino που θα καλύπτουν την τρέχουσα τεχνολογία.



Εικόνα 1.3: Arduino Breadboards

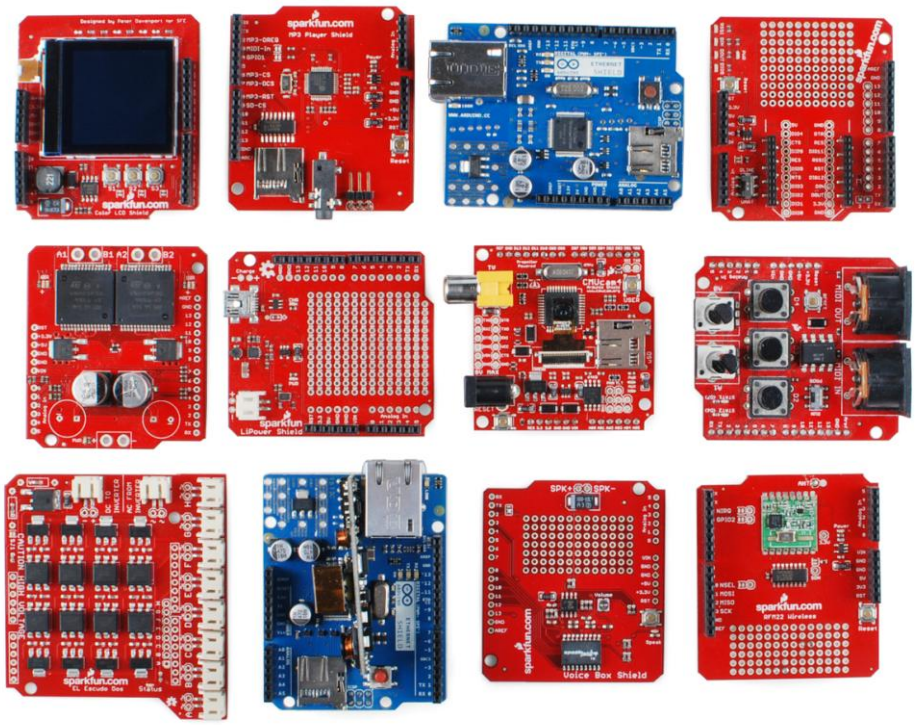
- **Οι πλακέτες επέκτασης (shields) του**

Υπάρχουν και οι πλακέτες επέκτασης (shields) (Εικόνα 1.4) που έρχονται να ενισχύσουν και να δώσουν νέες δυνατότητες στις πλατφόρμες του Arduino. Τα shields είναι ολοκληρωμένες πλακέτες που είναι σχεδιασμένες ώστε να κουμπώνουν πάνω στο Arduino προεκτείνοντας την λειτουργικότητά του. Είναι η hardware αντίστοιχη έννοια των plugin, addon και extension που υπάρχουν στο software.

Μερικά από τα πιο δημοφιλή shields που κυκλοφορούν στο εμπόριο για το Arduino είναι:

- **Ethernet shield:** Δίνει στο Arduino την δυνατότητα να δικτυωθεί σε ένα LAN ή στο Internet (Διαδίκτυο) μέσω ενός τυπικού καλωδίου Ethernet.
- **WiFi shield:** Όμοιο με το Ethernet shield, χωρίς φυσικά το καλώδιο.
- **Shields οθόνης**: Προσθέτουν οθόνη στο Arduino. Κυκλοφορούν από απλές οθόνες τύπου calculator μέχρι OLED touchscreen υψηλής ανάλυσης τύπου iPhone.
- **Wave shield:** Δίνει στο Arduino την δυνατότητα να παίζει ήχους/μουσική από κάρτες SD.
- **GPS shield:** Προσθέτει GPS δυνατότητες στο Arduino (εντοπισμό στίγματος).

- **Motor Shields:** Μας επιτρέπουν να οδηγήσουμε εύκολα μοτέρ διάφορων τύπων (απλά DC, servo, stepper κλπ) από το Arduino.
- **Proto Shield:** Μια προσχεδιασμένη πλακέτα πρωτοτυποποίησης, συμβατή στις διαστάσεις του Arduino και χωρίς εξαρτήματα για να φτιάξουμε το δικό μας shield.



Εικόνα 1.4: Arduino Shields

- **Η οικογένεια του Arduino**

Ένα ακόμα από τα σημαντικότερα του πλεονεκτήματα είναι το πλήθος των ανθρώπων που ασχολούνται με κάθε τομέα του Arduino (υλικό και λογισμικό). Έτσι υπάρχουν αμέτρητα forums και ιστοσελίδες που αναφέρονται στο Arduino και μπορούν να καθοδηγήσουν, βοηθήσουν, διδάξουν και εμπνεύσουν τον κάθε χρήστη.

- **Το Ανοικτού κώδικα και επέκτασης λογισμικό (software) του**

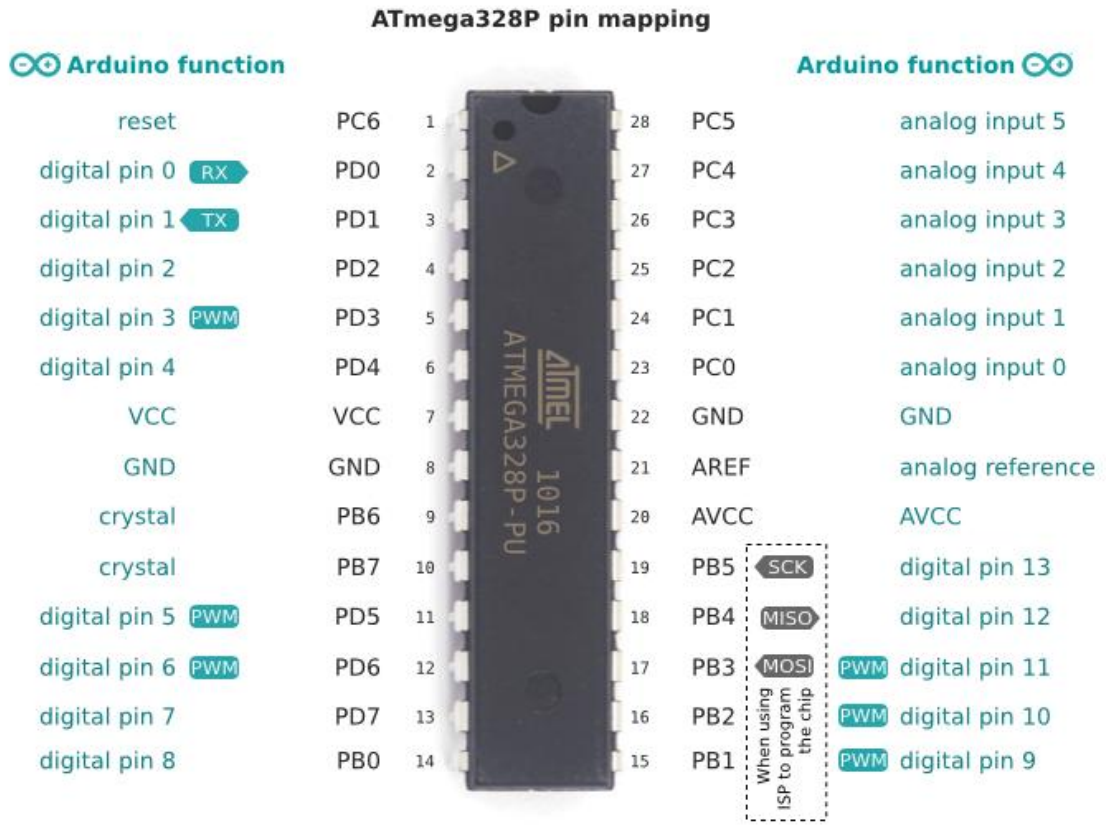
Ο καθένας μπορεί να βρει τον πηγαίο κωδικό, να τον μελετήσει και να τον τροποποιήσει σύμφωνα με τις ανάγκες του. Οι χρήστες μπορούν μέσα από τις βιβλιοθήκες τις C++ αλλά και μέσα από τις βιβλιοθήκες του προγράμματος του Arduino να γράψουν τον δικό τους κώδικα και να τον μοιραστούν.

- **Το Ανοικτού κώδικα και επέκτασης υλικό (hardware) του**

Τα σχέδια των πλατφορμών είναι ανοικτού κώδικα, πράγμα που σημαίνει ότι οι χρήστες μπορούν να επεκτείνουν και να αναβαθμίσουν τις πλατφόρμες.

1.4 Ανάλυση των ακροδεκτών (pins) του Arduino Uno

Υπάρχουν πολλές εκδόσεις και παραλλαγές του Arduino αλλά στην παρούσα πτυχιακή εργασία θα αναλύσουμε το πιο γνωστό μοντέλο που είναι το Arduino Uno και το οποίο θα χρησιμοποιήσουμε για την ανάπτυξη του αντικειμένου της πτυχιακής μας εργασίας που είναι ένα ρομποτικό όχημα. Στην Εικόνα 1.5 φαίνονται αναλυτικά όλοι οι ακροδέκτες του μικροελεγκτή ATmega328P του Arduino Uno.



Εικόνα 1.5: ATmega328P Pin Mapping

Συγκεκριμένα υπάρχουν αναλογικοί (analog) και ψηφιακοί (digital) ακροδέκτες (pins), ακροδέκτες επανεκκίνησης (reset), ακροδέκτες τροφοδοσίας (power) και γείωσης (GND).

Αναλυτικά:

- **Αναλογικοί ακροδέκτες (Analog pins)**

Στην κάτω δεξιά πλευρά της πλακέτας υπάρχουν 6 pins αριθμημένα από το 0 έως το 5 με την σήμανση ANALOG IN (Εικόνα 1.6) τα οποία λειτουργούν ως αναλογικές είσοδοι κάνοντας χρήση του ADC που είναι ενσωματωμένος στον μικροελεγκτή ATmega. Η μέτρηση της τάσης γίνεται στο εύρος από 0V έως 5V και διαβάζεται από το Arduino ως τιμή από το 0 (που ισούται με 0V) έως 1023 (που ισούται με 5V). Αυτά τα 6 pins μπορούν επίσης να μετατραπούν σε ψηφιακά pins εισόδου/εξόδου αλλά τότε μετονομάζονται σε pins 14-19.

- **Ακροδέκτες τροφοδοσίας, γείωσης και επανεκκίνησης (Power, Ground and Reset pins)**

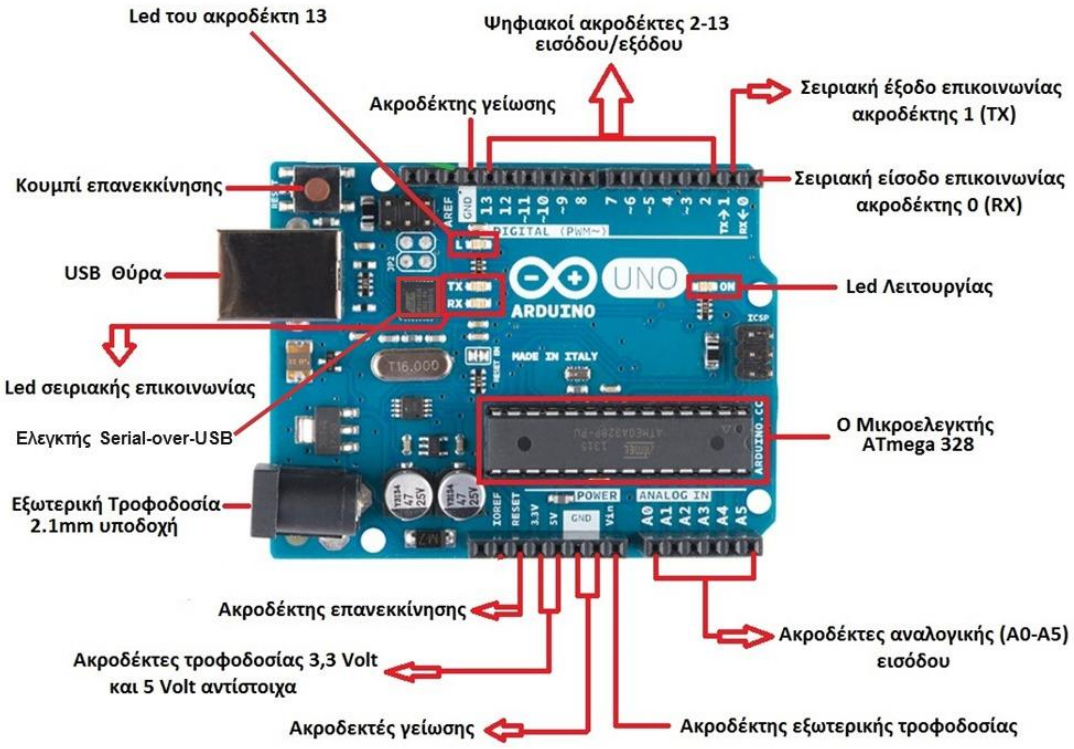
Πιο αριστερά έχουμε τους ακροδέκτες με την ένδειξη RESET, 3,3V, 5V, GND και Vin (Εικόνα 1.6). Συγκεκριμένα:

- **Ο πρώτος ακροδέκτης με την ένδειξη RESET**, όταν συνδεθεί (μέσω καλωδίου) με οποιονδήποτε από τους 3 ακροδέκτες με ένδειξη GND κάνει επανεκκίνηση το Arduino. Μπορούμε ακόμα να κάνουμε επανεκκίνηση το Arduino Uno πατώντας απλά το κουμπί επανεκκίνησης (εικόνα 1.6), που είναι ένας διακόπτης micro-switch με την ένδειξη RESET.
- **Ο δεύτερος ακροδέκτης με ένδειξη 3,3V** δίνει τάση της τάξεως των 3,3V με μέγιστη ένταση 50 mA.
- **Ο τρίτος ακροδέκτης με ένδειξη 5V** δίνει τάση της τάξεως των 5V με μέγιστη ένταση 40 mA.
- **Ο τέταρτος και πέμπτος ακροδέκτης με ένδειξη GND** είναι οι γειώσεις του Arduino.
- **Ο έκτος και τελευταίος ακροδέκτης με ένδειξη Vin** έχει δύο λειτουργίες **α)** μαζί με έναν από τους ακροδέκτες της γείωσης μπορεί να δουλέψει ως εξωτερική τροφοδοσία για την πλακέτα του Arduino ή **β)** αν το Arduino ήδη τροφοδοτείται με ρεύμα από την θύρα usb ή την 2,1mm υποδοχή, τότε μπορούμε να τροφοδοτήσουμε τα εξαρτήματα μας με την πλήρη τάση της εξωτερικής μας πηγής (7-12V).

- **Ψηφιακοί ακροδέκτες (Digital pins)**

Στην πάνω πλευρά του Arduino (Εικόνα 1.6) βρίσκονται 12 ψηφιακοί θηλυκοί ακροδέκτες (pins) εισόδου/εξόδου, αριθμημένοι από το 2 έως 13. Λειτουργούν στα 5V και έχουν τη δυνατότητα να δεχτούν ή να παρέχουν τάση μέχρι 40 mA. Οι

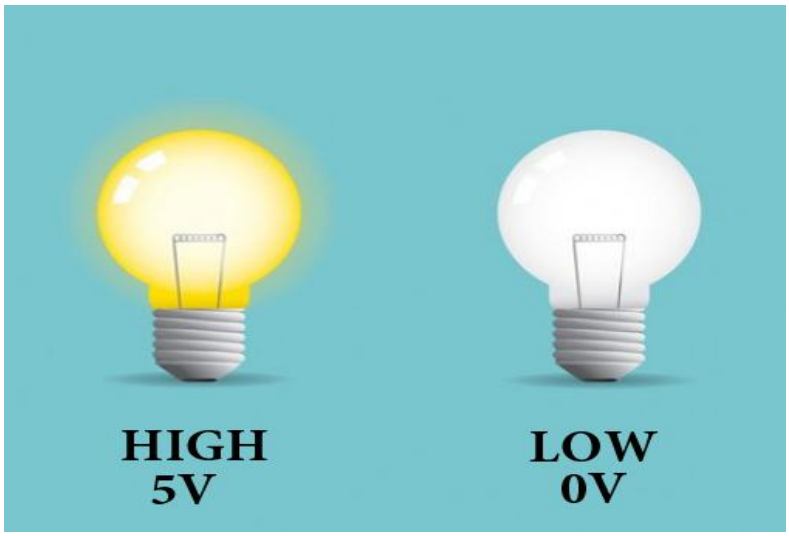
ακροδέκτες αυτοί μπορούν να τεθούν ως είσοδοι ή έξοδοι μέσα από το προγραμματιστικό περιβάλλον του Arduino Uno και να χρησιμοποιήσουμε τις συναρτήσεις digitalWrite() και digitalRead() στον κώδικά μας. Επίσης υπάρχουν και 2 ψηφιακοί θηλυκοί ακροδέκτες (pins) σειριακής επικοινωνίας αριθμημένοι από το 0 έως 1, αλλά δεν μπορούν να χρησιμοποιηθούν ως ψηφιακές είσοδοι/έξοδοι (digital i/o).



Εικόνα 1.6: Arduino Uno Pins

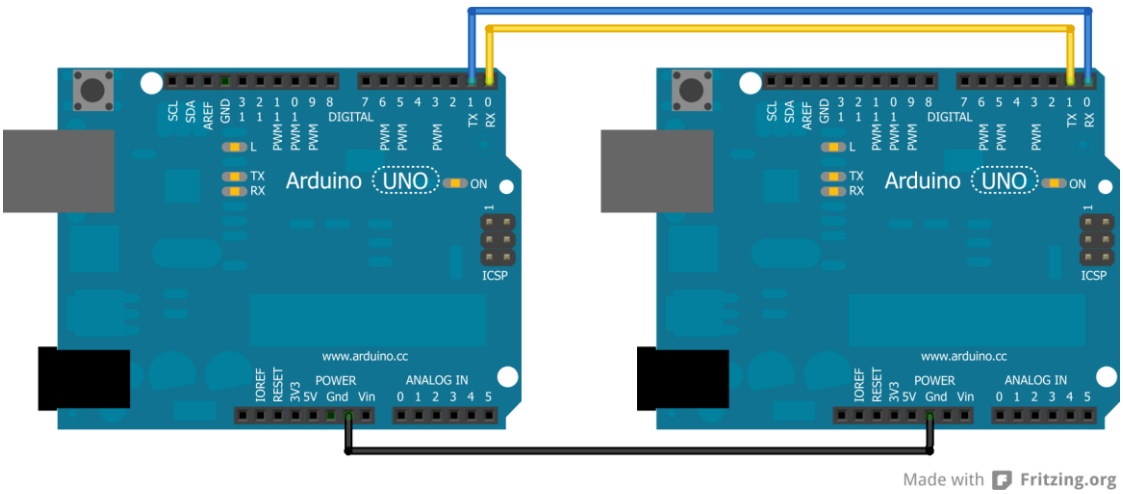
Ως έξοδος ένας ψηφιακός ακροδέκτης (pin) μπορεί να είναι σε μία από τις δύο καταστάσεις, ή σε κατάσταση 5V το οποίο ονομάζουμε HIGH, ή σε κατάσταση 0V το οποίο ονομάζουμε LOW. Φανταστείτε το άναμμα και το σβήσιμο μιας λάμπας, όταν η λάμπα φωτίζει είμαστε σε κατάσταση HIGH και όταν δεν φωτίζει είμαστε σε κατάσταση LOW. Με τον ίδιο τρόπο όταν το Arduino διοχετεύει ρεύμα στον ακροδέκτη είμαστε σε κατάσταση HIGH και όταν δεν διοχετεύει ρεύμα είμαστε σε κατάσταση LOW.

Ως είσοδος το Arduino δέχεται ή όχι Volts από το ψηφιακό ακροδέκτη. Πάλι και σε αυτή την περίπτωση έχουμε δύο καταστάσεις. Όταν το Arduino δέχεται 5V είναι σε κατάσταση HIGH και όταν δέχεται (ή δεν δέχεται) 0V είναι σε κατάσταση LOW (Εικόνα 1.7).



Εικόνα 1.7: HIGH(5V) - LOW(0V)

- **Οι ακροδέκτες 0 και 1** χρησιμοποιούνται για την σειριακή επικοινωνία (Εικόνα 1.8). Το Arduino μπορεί και στέλνει ή δέχεται δεδομένα μέσα από την θύρα του usb (ελεγκτής Serial-over-Usb) και μέσα από τους ακροδέκτες 0 (RX) και 1 (TX), όπως φαίνεται στην εικόνα 1.6. Ο ακροδέκτης 0 (RX) είναι για την λήψη δεδομένων προς το Arduino και ο ακροδέκτης 1 (TX) είναι για την αποστολή δεδομένων από το Arduino. Επίσης υπάρχουν δυο μικρά leds πάνω στην πλακέτα με την επωνυμία RX και TX (εικόνα 1.6) τα οποία αναβοσβήνουν όταν το Arduino δέχεται ή στέλνει δεδομένα αντίστοιχα δίνοντας μας οπτική επιβεβαίωση της σειριακής επικοινωνίας.

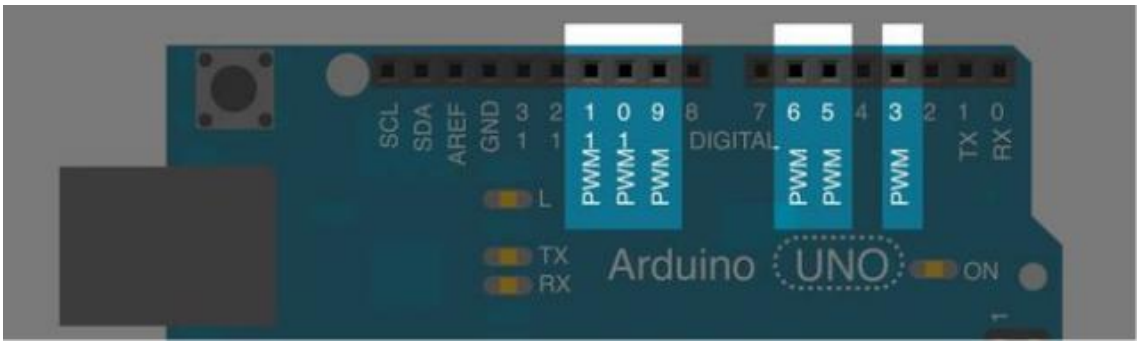


Εικόνα 1.8: Arduino Uno Pins – RX (0) & TX(1)

- **Οι ακροδέκτες 2 και 3** μπορούν να λειτουργήσουν και ως εξωτερικοί διακόπτες (interrupt). Σαν διακόπτες μπορούν να διακόψουν την κανονική ροή του προγράμματος, όταν ανιχνευτεί αλλαγή στην κατάσταση τους και να εκτελεστεί μια σειρά από εντολές. Η διαδικασία αυτή είναι ιδιαίτερα χρήσιμη όταν απαιτείται συγχρονισμός ακριβείας.
- **Οι ακροδέκτες 3, 5, 6, 9, 10, 11** μπορούν να λειτουργήσουν και ως αναλογικοί έξοδοι με το σύστημα PWM ή αναλυτικότερα pulse width modulation (διαμόρφωση διάρκειας παλμών).
- **Ο ακροδέκτης 13** είναι συνδεδεμένος με ένα led πάνω στην πλακέτα με την σήμανση L (Εικόνα 1.6) και αποτελεί στην ουσία κομμάτι του πρώτου πειραματισμού με το Arduino, αφού ακόμα κι αν δεν έχουμε συνδέσει τίποτα στον συγκεκριμένο ακροδέκτη, αναθέτοντας του την τιμή HIGH μέσα από το πρόγραμμά μας χρησιμοποιώντας τη συνάρτηση digitalWrite(pin,HIGH) θα ανάψει αυτό το ενσωματωμένο led.

Οι ακροδέκτες PWM του Arduino Uno

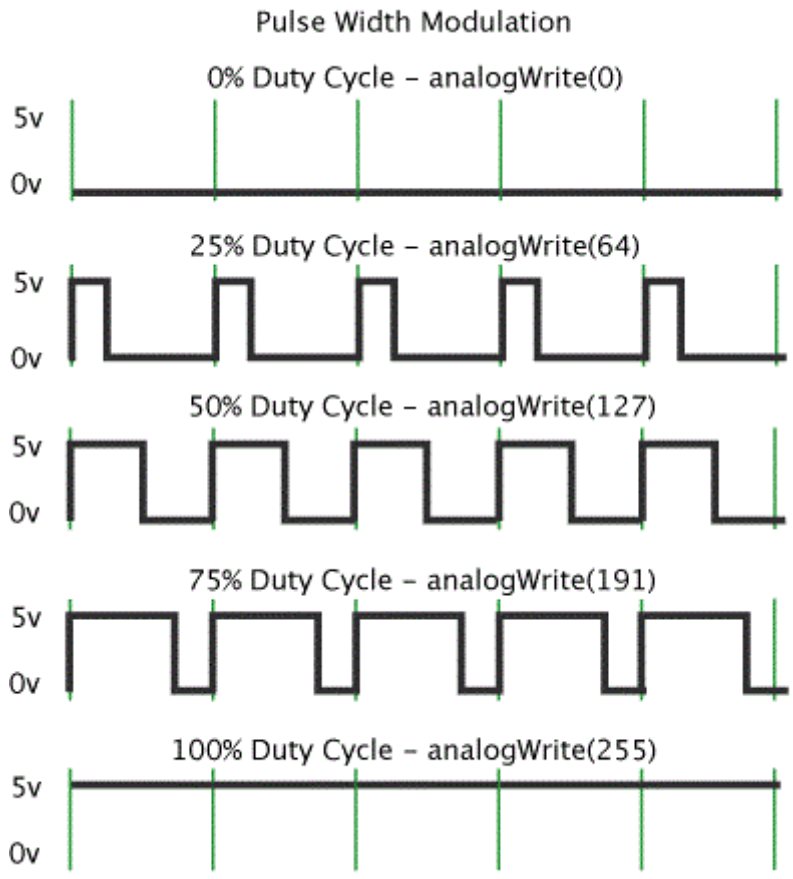
Με τους παλμούς PWM (Pulse Width Modulation) (Εικόνα 1.9) μπορούμε να στείλουμε ένα αναλογικό σήμα με την ψηφιακή του έννοια όμως. Υπό τη γενική έννοια, η μέση τιμή της τάσης (και του ρεύματος) που τροφοδοτείται στο φορτίο ελέγχεται με το γύρισμα του διακόπτη on/off πολύ γρήγορα. Όσο πιο πολύ μένει ο διακόπτης στην λειτουργία on, τόσο μεγαλύτερη είναι και η τροφοδοσία στο φορτίο. Αντίθετα, όσο πιο πολύ μένει ο διακόπτης στην λειτουργία off, τόσο λιγότερη είναι η τροφοδοσία στο φορτίο.



Εικόνα 1.9: Arduino Uno - PWM Pins

Pulse Width Modulation

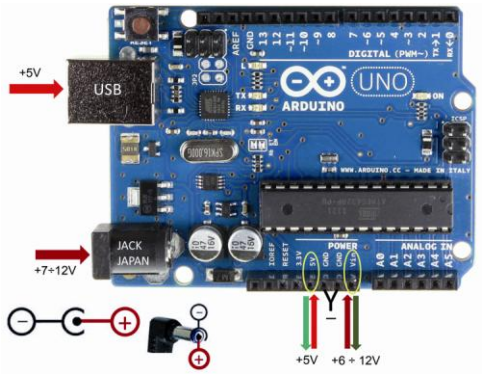
Ο ψηφιακός έλεγχος χρησιμοποιείτε για να δημιουργηθεί ένας τετραγωνικός παλμός, ένας διακόπτης δηλαδή μεταξύ on και off (HIGH/LOW). Στην “on” λειτουργία του παλμού έχουμε σήμα 5V (δηλαδή HIGH) ενώ στην “off” λειτουργία έχουμε σήμα 0V (δηλαδή LOW). Το χρονικό διάστημα που απαιτείται για να γίνει αυτή η εναλλαγή ονομάζεται πλάτος παλμού (pulse width). Οι τιμές που μπορούν να δοθούν είναι από 0 έως 255 με τη χρήση της συνάρτησης analogWrite() στο προγραμματιστικό περιβάλλον του Arduino Uno (Εικόνα 1.10).



Εικόνα 1.10: PWM Pins- Duty Cycles

Τροφοδοσία

Τέλος το Arduino τροφοδοτείται με ρεύμα μέσω α) της υποδοχής 2.1mm, β) του usb βύσματος και γ) τους ακροδέκτες εξωτερικής τροφοδοσίας (Εικόνα 1.11). Αυτή είναι και η σειρά προτεραιότητας του Arduino αν και η επιλογή της τροφοδοσίας γίνεται αυτόματα. Δηλαδή αν το Arduino συνδεθεί με ρεύμα μέσα από την υποδοχή 2.1mm και μέσα από το usb βύσμα ταυτόχρονα, θα χρησιμοποιήσει την υποδοχή 2.1mm για τροφοδοσία και θα θεωρήσει το usb βύσμα ότι είναι για την επικοινωνία με τον υπολογιστή. Επίσης το Arduino μπορεί να λειτουργήσει με εξωτερική τροφοδοσία από 6-20V. Ιδανικά Volts για την λειτουργία του είναι τα 9V. Όταν βρίσκεται σε λειτουργία το Arduino ανάβει το led λειτουργίας (εικόνα 1.6).



Εικόνα 1.11: Arduino Uno Power Supplies

1.5 Ανασκόπηση της πτυχιακής εργασίας

Η παρούσα πτυχιακή εργασία βασίζεται στο Arduino Uno κι ασχολείται με την κατασκευή ρομποτικού οχήματος που να κινείται μόνο του στο χώρο, να ελέγχεται η κίνησή του με την τεχνολογία Bluetooth, να εκτελεί παράλληλα και τη χειροκίνητη και την αυτόματη λειτουργία και να κινείται με φωνητικές εντολές. Στη συνέχεια περιγράφονται τα βήματα κατασκευής του ρομποτικού οχήματος καθώς και η συνδεσμολογία των εξαρτημάτων του και τέλος αναλύονται τα βήματα προγραμματισμού του Arduino Uno σε σύνδεση με τον υπολογιστή μέσα από το προγραμματιστικό του περιβάλλον το Arduino IDE για την ελεγχόμενη κίνηση του ρομποτικού οχήματος.

ΚΕΦΑΛΑΙΟ 2

ΚΑΤΑΣΚΕΥΗ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΟΧΗΜΑΤΟΣ

Σε αυτό το κεφάλαιο αναλύονται τα βήματα υλοποίησης της κατασκευής του ρομποτικού οχήματος καθώς και τα υλικά και τα εργαλεία που θα χρειαστούν για την ολοκλήρωση της κατασκευής. Ακόμα περιγράφεται αναλυτικά η συναρμολόγηση των εξαρτημάτων του ρομποτικού οχήματος.

2.1 Τα υλικά που θα χρειαστούμε

- **Genuino Uno Rev3 και το καλώδιο usb του**

Εικόνα 2.1: Genuino Uno Rev3 & Usb Cable

- **Μικρή πλακέτα (Male to Male Breadboard)**

Εικόνα 2.2: Male to Male Breadboard

- **Bluetooth module HC-06**

Εικόνα 2.3: Bluetooth Module HC-06

- **Robot Kit**

Περιλαμβάνει ρομποτικό σασί με 2 μεγάλες ρόδες και μία πίσω μικρή ρόδα, 2 μοτέρ (DC motors), θήκη-βάση για 4 μπαταρίες (AA), αποστάτες, βίδες και παξιμάδια.

Εικόνα 2.4: Robot Kit

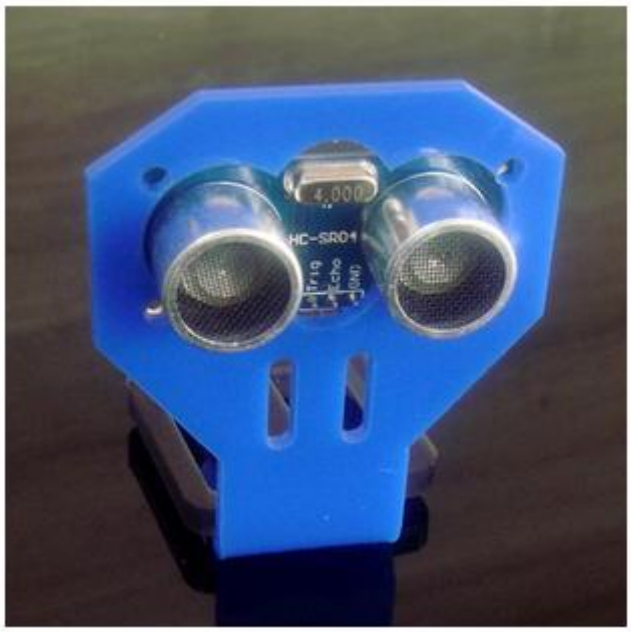
- **Μπαταριοθήκη για μπαταρία 9V**

Εικόνα 2.5: Battery Holder 9V

- **2 Διακόπτες ON/OFF**

Εικόνα 2.6: Rocker Switch ON-OFF

- **Αισθητήρας υπερήχων HC-SR04 μαζί με την βάση του (mounting bracket for ultrasonic sensor)**

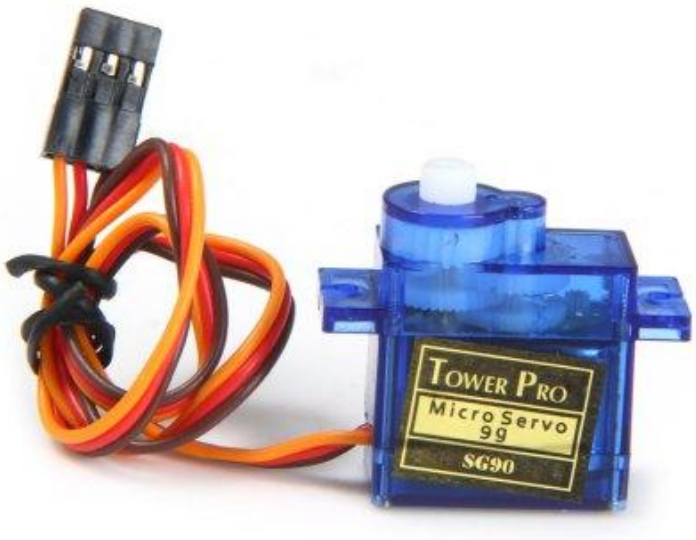


Εικόνα 2.7: Ranging Detector HC-SR04 & mounting bracket for ultrasonic sensor

- **L298N Motor Driver Module**

Εικόνα 2.8: L298N Motor Driver Module

- **Micro Servo-Motor**



Εικόνα 2.9: Micro Servo-Motor

- **Servo Mounting Bracket μαζί με βίδες και παξιμάδια**

Εικόνα 2.10: Servo Mounting Bracket

- **Δύο διακόπτες "επαφής" (Micro Switches)**

Εικόνα 2.11: Micro Switch

- **Buzzer**

Εικόνα 2.12: Buzzer

- **Το DC (Direct Current) Power Plug του Arduino Uno**

Εικόνα 2.13: ARDUINO DC power plug

- **Καλώδια (wires)**
  - **Male to Male (αρσενικό σε αρσενικό) wires**

Εικόνα 2.14: Male to Male wires

- **Female to Male (θηλυκό σε αρσενικό) wires**

Εικόνα 2.15: Female to Male wires

- **Θερμοσυστελλόμενα (Heat-Shrinkable)**

Εικόνα 2.16: Heat-Shrinkable

- **3 Αντιστάσεις (Resistors)**

Αντιστάσεις 100ohm, 220ohm και 10Kohm όπως φαίνονται στις εικόνες 2.17, 2.18 και 2.19 αντίστοιχα.

Εικόνα 2.17: Resistor 100ohm

Εικόνα 2.18: Resistor 220ohm

Εικόνα 2.19: Resistor 10Kohm

- **2 leds (10mm-χιλιοστά διάμετρο)**
  - **Πράσινο (green) led**

Εικόνα 2.20: Green led

  - **Κόκκινο (red) led**

Εικόνα 2.21: Red led

2.2 Τα εργαλεία που θα χρειαστούμε

- Δραπανοκατσάβιδο για να ανοίξουμε επιπλέον τρύπες που θα χρειαστούμε στο ρομποτικό σασί.
- Κατσαβίδι για να βιδώσουμε τις βίδες.
- Κολλητήρι και καλάι.
- Πιστόλι θερμοκόλλας και ράβδους θερμοκόλλας.
- Πολύμετρο
- Κόφτης-Απογυμνωτής καλωδίων
- Μυτοτσίμπιδο

2.3 Συναρμολόγηση του ρομποτικού οχήματος

Στην πάνω πλευρά του ρομποτικού σασί έχουμε τοποθετήσει στο πίσω μέρος το driver module L298N, δίπλα το Arduino Uno και πιο δίπλα το μικρό breadboard.

Το driver module L298N και το Arduino Uno τα έχουμε βιδώσει χρησιμοποιώντας αποστάτες και παξιμάδια πάνω στο ρομποτικό σασί. Τη μικρή πλακέτα (breadboard) την έχουμε κολλήσει πάνω στο ρομποτικό σασί με ταινία που διαθέτει στο κάτω μέρος της.

Πάνω στην πλακέτα έχουμε τοποθετήσει το Bluetooth module HC-06, το buzzer καθώς και τα 2 leds το πράσινο και το κόκκινο.

Στο μπροστινό μέρος του ρομποτικού σασί έχουμε τοποθετήσει τον αισθητήρα υπερήχων HC-SR04 μαζί με τη βάση του (mounting bracket for ultrasonic) κι από κάτω το micro servo-motor μαζί με τη βάση του (servo mounting bracket).

Στην κάτω πλευρά του ρομποτικού σασί έχουμε τοποθετήσει στο πίσω μέρος τη μικρή πίσω ρόδα, δίπλα τη θήκη για τη μπαταρία των 9V με την οποία θα λειτουργεί το Arduino Uno και η οποία θήκη συνδέεται με το Arduino με το DC power plug στο οποίο έχουμε χρησιμοποιήσει θερμοσυστελλόμενο για να είναι ομοιόμορφο το καλώδιο και πιο δίπλα τη θήκη για τις 4 μπαταρίες AA με τις οποίες θα λειτουργεί το driver module που θα ενεργοποιεί τα 2 μοτέρ.

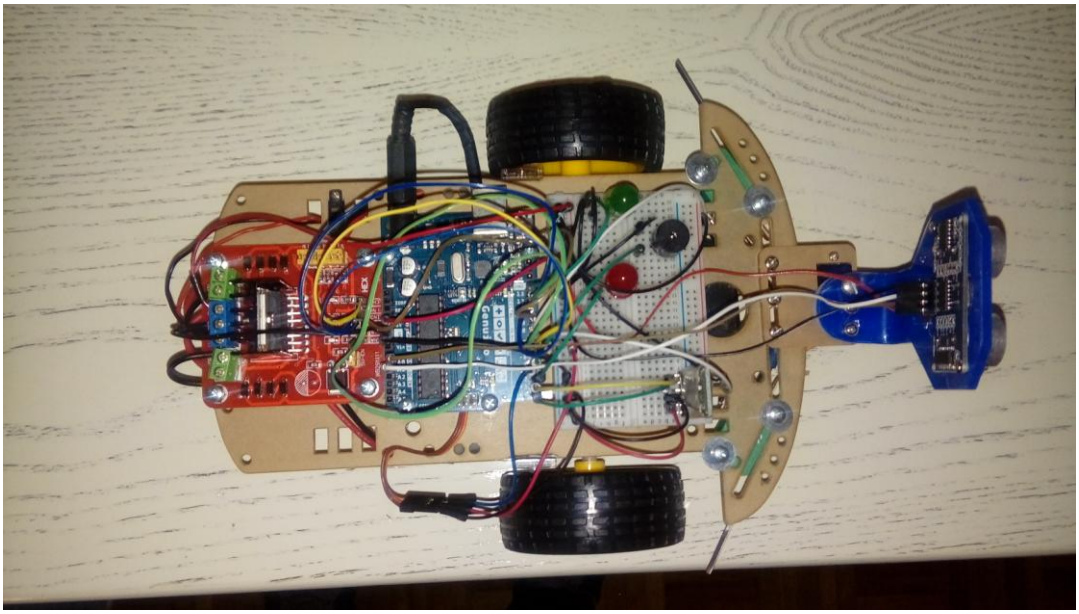
Τις μπαταριοθήκες τις έχουμε κολλήσει με θερμοκόλληση και τις έχουμε συνδέσει με διακοπτάκια ON/OFF κάνοντας τομή στα κόκκινα καλώδια.

Δεξιά κι αριστερά του ρομποτικού σασί έχουμε τοποθετήσει τις μεγάλες ρόδες και δίπλα σε κάθε ρόδα το μοτέρ χρησιμοποιώντας τις ανάλογες βίδες κι

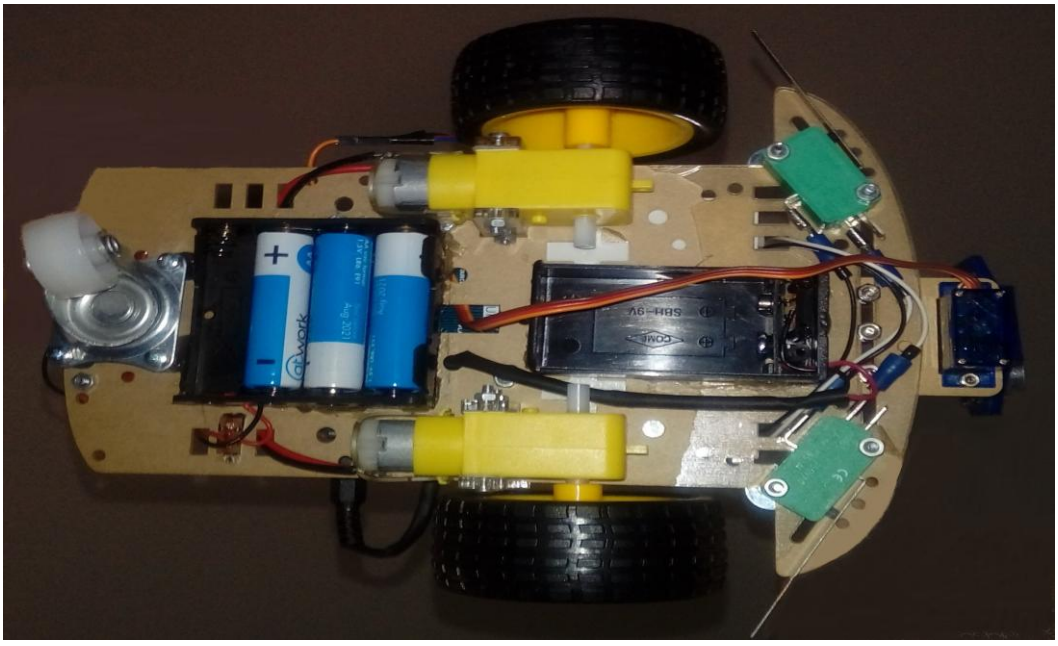
αποστάτες για να φέρουμε τις ρόδες στο σωστό ύψος ώστε να ισορροπεί το ρομποτικό όχημα. Έχουμε κολλήσει τα κόκκινα καλώδια στους πάνω ακροδέκτες των μοτέρ και τα μαύρα καλώδια στους κάτω ακροδέκτες των μοτέρ.

Δεξιά κι αριστερά στο μπροστινό μέρος του ρομποτικού σασί έχουμε τοποθετήσει τους διακόπτες “επαφής”.

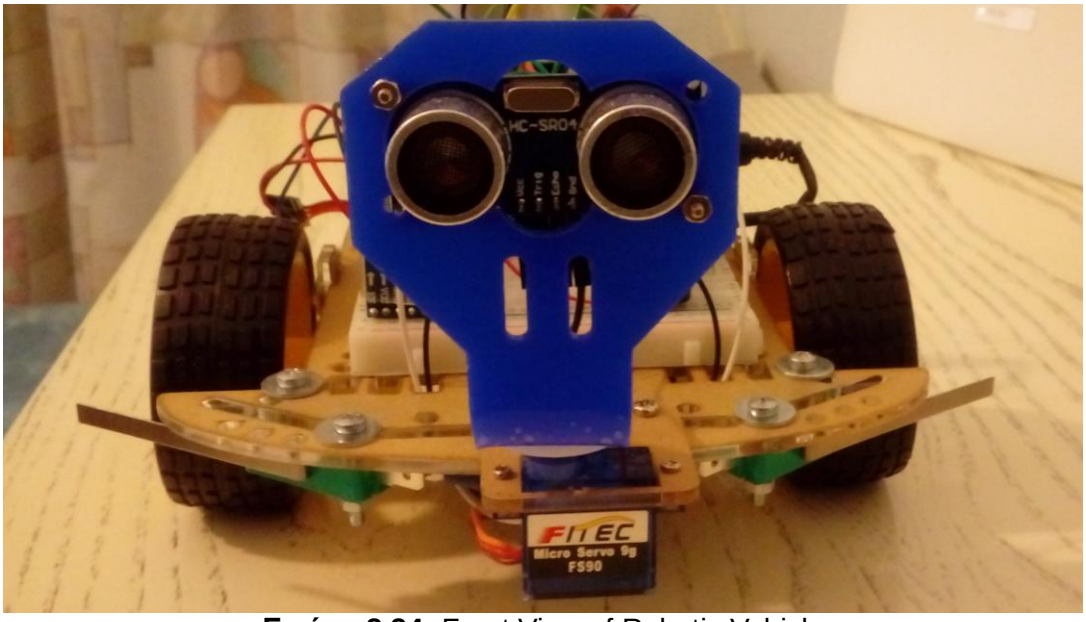
Η πάνω πλευρά, η κάτω πλευρά και το μπροστινό μέρος του ρομποτικού μας οχήματος φαίνονται στις εικόνες 2.22, 2.23 και 2.24 αντίστοιχα.



Εικόνα 2.22: Top View of Robotic Vehicle



Εικόνα 2.23: Bottom View of Robotic Vehicle



Εικόνα 2.24: Front View of Robotic Vehicle

Το DC Power Plug του Arduino Uno

Στο **tip** κολλήσαμε το **κόκκινο καλώδιο (VCC)** και στο **sleeve** κολλήσαμε το **μαύρο καλώδιο (GND)** το οποίο είναι τοποθετημένο μέσα στο **strain relief**,όπως φαίνεται στην εικόνα 2.25, για να φτιάξουμε το DC power plug με το οποίο θα συνδέουμε το Arduino Uno με τη τροφοδοσία του (βάση μπαταρίας των 9V).

Εικόνα 2.25: Tip & Sleeve

ΚΕΦΑΛΑΙΟ 3

ΣΥΝΔΕΣΜΟΛΟΓΙΑ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΟΧΗΜΑΤΟΣ

Σε αυτό το κεφάλαιο αναλύεται η συνδεσμολογία των εξαρτημάτων του ρομποτικού οχήματος και περιγράφεται αναλυτικά η λειτουργία του κάθε εξαρτήματος και ο ρόλος του στη λειτουργία του ρομποτικού οχήματος.

3.1 L298N Motor Driver Module

- **L298N driver module (Power, Ground & Motor Pins)**

Εικόνα 3.1: L298N Power, Ground & Motor pins

- **Motor A**: Vcc +GND
- **Power/Ground:** VMS (+5V έως +35V) + GND + 5V (external power)
- **Motor B**: Vcc + GND

Σύνδεση των Motor pins (Εικόνα 3.1) του driver module L298N με τα μοτέρ

- **MOTORA1:** Μαύρο καλώδιο - Δεξί μοτέρ
- **MOTORA2:** Κόκκινο καλώδιο - Δεξί μοτέρ
- **MOTORB1:** Κόκκινο καλώδιο - Αριστερό μοτέρ
- **MOTORB2:** Μαύρο καλώδιο - Αριστερό μοτέρ

Σύνδεση των Power και Ground pins (Εικόνα 3.1) του driver module L298N με την τροφοδοσία του (βάση για 4 μπαταρίες AA) και το Arduino Uno

- **Power VMS** - Κόκκινο καλώδιο απ' την τροφοδοσία
- **GND** - Μαύρο καλώδιο απ' την τροφοδοσία κι ένα επιπλέον προς το GND pin του Arduino Uno
- **Power 5V** – Κόκκινο καλώδιο προς το 5V pin του Arduino Uno

- **L298N driver module (motor signals)**

Εικόνα 3.2: L298N Signal pins

Σύνδεση των Signal pins (Εικόνα 3.2) του driver module L298N με το Arduino Uno

- **ENB** (Motor B enable PIN) – Arduino PWM pin 6
- **IN4** (Motor B direction 2 PIN) - Arduino pin 12
- **IN3** (Motor B direction 1 PIN) - Arduino pin 9
- **IN2** (Motor A direction 2 PIN) - Arduino pin 10
- **IN1** (Motor A direction 1 PIN) - Arduino pin 11
- **ENA** (Motor A enable PIN) – Arduino PWM pin 5

Χρησιμοποιούμε τα **PWM (Διαμόρφωση Πλάτους Σήματος-Παλμού) pins** του Arduino Uno για να τα συνδέσουμε με τα pins ENA και ENB του driver module L298N ώστε να είμαστε σε θέση να μπορούμε να **ελέγξουμε την ταχύτητα** του ρομποτικού οχήματος. Συγκεκριμένα χρησιμοποιούμε τα pwm pins 5 και 6 γιατί έχουν μεγαλύτερη συχνότητα σήματος (980 Hz) σε σύγκριση με τα υπόλοιπα pwm pins (490 Hz) κι ανταποκρίνονται καλύτερα στις τιμές που τους αναθέτουμε μέσα στο πρόγραμμα για τις ανάλογες ταχύτητες. Στον κώδικα μας χρησιμοποιούμε τη συνάρτηση analogWrite (pin, value), όπου pin **το ENA ή ENB** που ενεργοποιεί (enable) το αντίστοιχο μοτέρ με συχνότητα στροφών ανάλογα με την τιμή value ορίζουμε. Η ελάχιστη τιμή value είναι 0 και η μέγιστη τιμή value είναι 255.

Η συνάρτηση analogWrite (pin,0) κρατάει το μοτέρ στο 0% της ταχύτητας των στροφών που παίρνει άρα είναι μόνιμα απενεργοποιημένο(always off),το ίδιο κάνει και η χρήση της συνάρτησης digitalWrite (pin, LOW).

Η συνάρτηση analogWrite (pin, 255) κρατάει το μοτέρ στο 100% της ταχύτητας των στροφών που παίρνει άρα είναι μόνιμα ενεργοποιημένο (always on), το ίδιο κάνει και η χρήση της συνάρτησης digitalWrite (pin, HIGH).

Τα **pins IN1, IN2, IN3, IN4** στέλνουν σήμα στα μοτέρ και καθορίζουν τη φορά των στροφών τους (προς τα μπροστά ή προς τα πίσω) και συνεπώς ορίζουν τη φορά που κινούνται οι ρόδες και κατ' επέκταση την **κατεύθυνση (direction)** κίνησης του ρομποτικού οχήματος, ανάλογα με την εντολή που δώσουμε για κάθε pin μέσα στον κώδικα. Χρησιμοποιούμε τη συνάρτηση digitalWrite (pin, value), όπου pin το IN1, IN2, IN3, IN4 και value η τιμή LOW ή HIGH.

Τα **pins IN1 και IN2** καθορίζουν τη λειτουργία του **MOTORA (δεξί μοτέρ)** και κατ' επέκταση την κίνηση της δεξιάς ρόδας.

Τα **pins IN3 και IN4** καθορίζουν τη λειτουργία του **MOTORB (αριστερό μοτέρ)** και κατ' επέκταση την κίνηση της αριστερής ρόδας.

- **Λειτουργίες Μοτέρ**

Στον πίνακα 3.1 βλέπουμε αναλυτικά τις λειτουργίες των μοτέρ ανάλογα με την τιμή value που αναθέτουμε μέσα στον κώδικα σε κάθε signal pin, χρησιμοποιώντας την εντολή digitalWrite (pin,LOW) ή digitalWrite (pin, HIGH).

Πίνακας 3.1: Motor Functions

Motor Function	**Pin - Value**	**Pin - Value**
Motor A,B ON	ENA - HIGH	ENB - HIGH
Motor A Forward	IN1 – HIGH	IN2 – LOW
Motor A Backward	IN1 – LOW	IN2 – HIGH
Motor B Forward	IN3 – LOW	IN4 – HIGH
Motor B Backward	IN3 – HIGH	IN4 – LOW
Motor A Stop	IN1 – HIGH	IN2 – HIGH
Motor B Stop	IN3 – HIGH	IN4 – HIGH
Motor A,B OFF	ENA - LOW	ENB - LOW

- **Λειτουργίες Κίνησης Ρομποτικού Οχήματος**

Στον πίνακα 3.2 βλέπουμε αναλυτικά τις κινήσεις του ρομποτικού οχήματος ανάλογα με τη λειτουργίες των μοτέρ. Οι κινήσεις του ρομποτικού οχήματος είναι μπροστά (forward), πίσω (backward), δεξιά (right), αριστερά (left) και σταμάτημα (stop).

Πίνακας 3.2: Movement Functions

Robot movement	**Motor A movement**	**Motor B movement**
Robot Forward	Motor A Forward	Motor B Forward
Robot Backward	Motor A Backward	Motor B Backward
Robot Left	Motor A Forward	Motor B Backward
Robot Right	Motor A Backward	Motor B Forward
Robot Stop	Motor A Stop	Motor B Stop

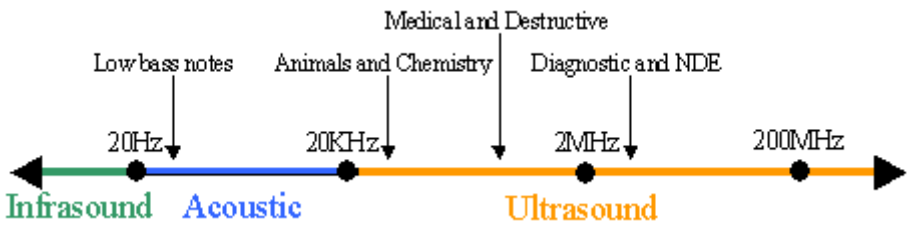
3.2 Αισθητήρας υπερήχων HC – SR04 (Ultrasonic sensor)

- **Σύνδεση του αισθητήρα υπερήχων HC-SR04 (Ultrasonic Sensor) με το Arduino Uno**
  - **VCC** - Arduino pin 5V (θέση breadboard)
  - **Trig** - Arduino pin A0 (analog pin 0)
  - **Echo** - Arduino pin A1 (analog pin 1)
  - **GND** - Arduino pin GND (θέση breadboard)

Χρησιμοποιούμε το breadboard για τα καλώδια τροφοδοσίας 5V και γείωσης GND αλλά και για μικρά καλώδια που δεν φτάνουν στο Arduino, ως προέκταση.

Οι υπέρηχοι και η λειτουργία του αισθητήρα υπερήχων

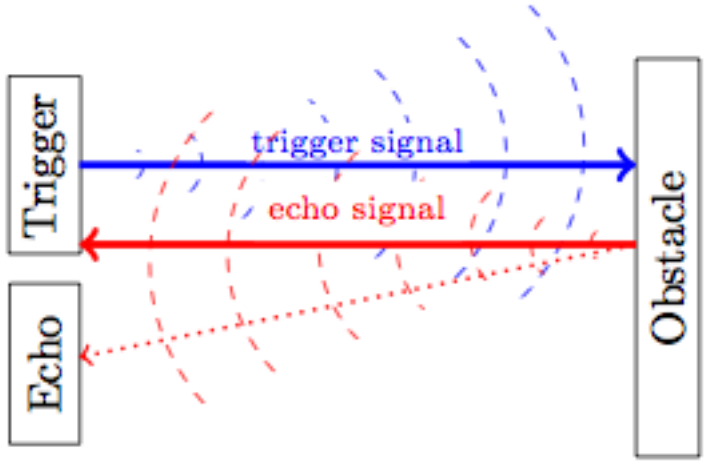
Όπως φαίνεται στην Εικόνα 3.3, οι υπέρηχοι βρίσκονται πάνω απ' τις ακουστικές συχνότητες, έτσι δε μπορεί να τους ακούσει το ανθρώπινο αυτί. Παρόλο πάντως που εμείς δεν τους ακούμε κάποια ζώα μπορούν και να τους ακούν αλλά και να τους χρησιμοποιούν. Χαρακτηριστικά παραδείγματα που μας το δείχνουν αυτό 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 αυτοκίνητα μας κατά τη διαδικασία της στάθμευσης (parking sensors).



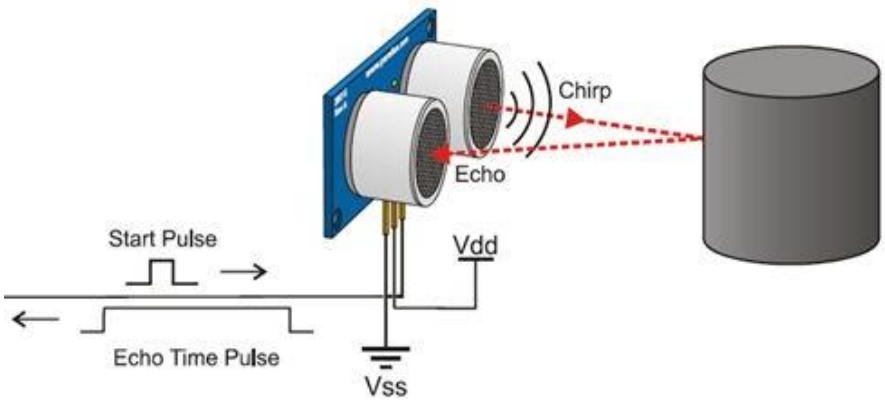
Εικόνα 3.3: Acoustic Signal Frequency Range

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

Όπως φαίνεται στις Εικόνες 3.4 και 3.5, δημιουργούν υψηλής συχνότητας κύματα (Trigger signal – Εικόνα 3.4) και χρησιμοποιώντας το επιστρεφόμενο σήμα (Echo signal – Εικόνα 3.4) καθορίζουν την απόσταση ή ακόμα και την ταχύτητα του στόχου. Για να το πετύχουν αυτό χρησιμοποιούν τον χρόνο (Echo Time Pulse – Εικόνα 3.5) που έκανε το σήμα για να καλύψει την απόσταση από τον αισθητήρα στο αντικείμενο και πίσω.



Εικόνα 3.4: Trigger, Echo signals



Εικόνα 3.5: Ultrasonic Sensor - Object Detection

Στο Arduino Uno η διαδικασία αντιστοίχισης χρόνου σε απόσταση γίνεται με τη χρήση συγκεκριμένης βιβλιοθήκης που υπάρχει για τον αισθητήρα HC-SR04. Εμείς αρκεί να κάνουμε χρήση μόνο μιας εντολής στον κώδικά μας. Η εντολή αυτή θα μας επιστρέψει την απόσταση σε εκατοστά (cm).

3.3 Διακόπτες “επαφής” (micro switches)

- **Ο κάθε διακόπτης έχει 3 επαφές (Εικόνα 3.6)**
  - **NC:** normaly close
  - **NO:** nornaly open
  - **COM:** common

- **Σύνδεση των διακοπτών “επαφής” με το Arduino Uno**
  - Διακόπτης αριστερά:**COM** - Arduino pin GND (θέση breadboard)
  - Διακόπτης αριστερά:**NO** - Arduino pin 7
  - Διακόπτης δεξιά :**COM** - Arduino pin GND (θέση breadboard)
  - Διακόπτης δεξιά:**NO** - Arduino pin 8

Στην περίπτωση μας δεν χρησιμοποιούμε την επαφή **NC**.

Εικόνα 3.6. Micro Switch pins

Ο αισθητήρας υπερήχων που χρησιμοποιούμε για την αποφυγή εμποδίων δεν έχει πολύ μεγάλη οπτική γωνία και μερικές φορές το όχημα μας μπορεί να χτυπήσει πάνω σε αντικείμενα που βρίσκονται κοντά στις ρόδες, δηλαδή στα άκρα του, γιατί δεν βρίσκονται στο οπτικό πεδίο του αισθητήρα. Αυτό αντιμετωπίζεται με τη χρήση των διακοπτών “επαφής”.

3.4 Micro Servo-Motor

Το servo motor έχει 3 καλώδια το **κόκκινο (VCC)**, το **καφέ (GND)** και το **πορτοκαλί** το οποίο το συνδέουμε σε κάποιο digital pin του Arduino Uno (Εικόνα 2.9).

- **Σύνδεση του micro servo-motor με το Arduino Uno**
  - **Κόκκινο καλώδιο VCC** - Arduino pin 5V (θέση breadboard)
  - **Καφέ καλώδιο GND** - Arduino pin GND (θέση breadboard)
  - **Πορτοκαλί καλώδιο** - Arduino pin 3

Ο servo κινητήρας γυρνάει τον αισθητήρα υπερήχων δεξιά και αριστερά.

3.5 Buzzer

Το buzzer έχει 2 ακροδέκτες (pins).Ο ένας ακροδέκτης συνδέεται με το GND του Arduino Uno κι ο άλλος ακροδέκτης με κάποιο digital pin του Arduino Uno.

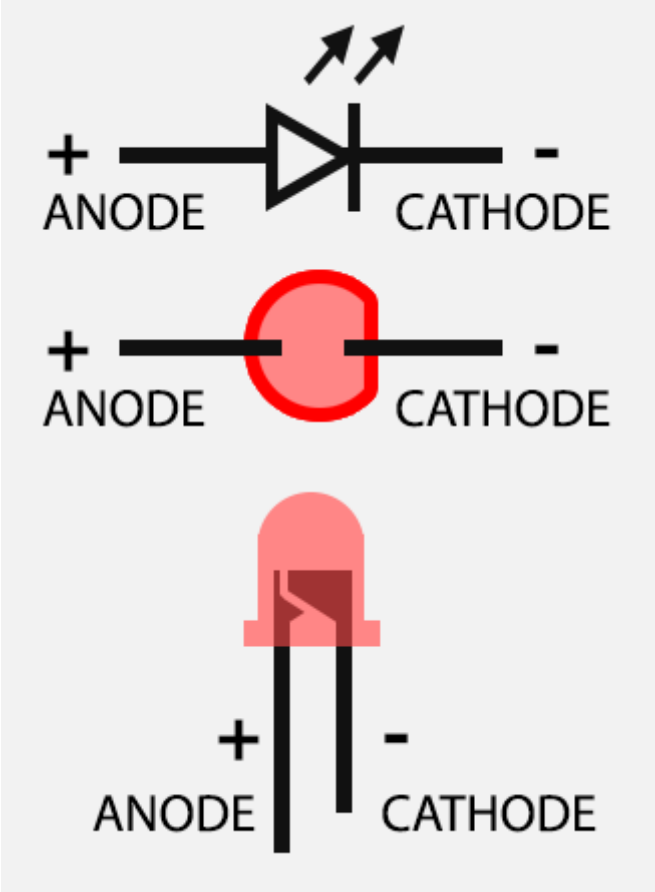
- **Σύνδεση του buzzer με το Arduino Uno**
  - **Buzzer Pin1** - Arduino pin GND (θέση breadboard)
  - **Buzzer Pin2** – Arduino pin 4

Χρησιμοποιούμε τη συνάρτηση tone (pin, frequency, duration) για να στείλουμε σήμα στο digital pin του Arduino, όπου είναι συνδεδεμένος ο buzzer για να βγάλει ήχο με συχνότητα frequency και διάρκεια duration και τη noTone (pin) για να σταματήσουμε αυτό το σήμα. Προαιρετικά μπορούμε να βάλουμε μια αντίσταση 100ohm στη σύνδεση του buzzer με το digital pin 4 του Arduino για να ενισχύσουμε τον ήχο.

3.6 Leds

Τα leds διαθέτουν δύο ακροδέκτες. Το θετικό άκρο, δηλαδή ο μεγαλύτερος ακροδέκτης ονομάζεται **"άνοδος" (Anode)** και συνδέεται σε κάποιο digital pin του Arduino ενώ το αρνητικό άκρο λέγεται **"κάθοδος" (Cathode)** και συνδέεται στη γείωση (GND) του Arduino (Εικόνα 3.7).

- **Σύνδεση των leds(10mm) με το Arduino Uno**
  - **Πράσινο (green) led**
    - **GreenLed Anode** - Arduino pin GND (θέση breadboard)
    - **GreenLed Cathode** – Arduino pin 13
  - **Κόκκινο(red) led**
    - **RedLed Anode** - Arduino pin GND (θέση breadboard)
    - **RedLed Cathode** – Arduino pin 2



Εικόνα 3.7 Led pins

Με τη χρήση της συνάρτησης analogWrite (pin, 255) το led ανάβει στο 100% της φωτεινότητας του, όπως θα έκανε και η digitalWrite (pin, HIGH), ενώ με τη χρήση της συνάρτησης analogWrite(0) το led παραμένει σβηστό, όπως θα έκανε και η digitalWrite (pin, LOW). Στην περίπτωσή μας μπορούμε να χρησιμοποιήσουμε μόνο τη συνάρτηση digitalWrite() κι όχι την analogWrite() γιατί δεν έχουμε συνδέσει τα leds σε κάποιο PWM pin του Arduino.

Προαιρετικά μπορούμε να βάλουμε μια αντίσταση 220ohm στη σύνδεση των leds πράσινου και κόκκινου με τα digital pins 13 και 2 αντίστοιχα του Arduino για να ενισχύσουμε τη φωτεινότητα.

3.7 Bluetooth Module HC-06

- **Σύνδεση του bluetooth HC-06 με το Arduino Uno:**
  - **Bluetooth VCC** - Arduino pin 5V (θέση breadboard)
  - **Bluetooth TXD** - Arduino pin RXD (Digital pin 0)
  - **Bluetooth RXD** - Arduino pin TXD (Digital pin 1)
  - **Bluetooth GND** - Arduino pin GND (θέση breadboard)

Το Bluetooth είναι ένα βιομηχανικό πρότυπο για ασύρματα προσωπικά δίκτυα υπολογιστών 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 PDA, κινητά τηλέφωνα, ηλεκτρονικούς υπολογιστές, εκτυπωτές, καθώς και ψηφιακές φωτογραφικές μηχανές ή ψηφιακές κάμερες, μέσω μιας ασφαλούς, φθηνής και παγκοσμίως διαθέσιμης χωρίς ειδική άδεια ραδιοσυχνότητας μικρής εμβέλειας. Από τεχνικής άποψης το Bluetooth είναι ένα πρωτόκολλο ασύρματης δικτύωσης σε φυσικό επίπεδο, υποεπίπεδο MAC και προαιρετικά, υποεπίπεδο LLC.

Παρακάτω αναφέρονται τα pins (ακροδέκτες) του BT όπως φαίνονται στην Εικόνα 3.8

- **STATE:** Κατάσταση σύνδεσης
- **RXD:** receive pin
- **TXD:** transmit pin
- **GND:** Ground
- **VCC:** 3,3V ή 5V
- **WAKEUP:** HIGH=προγραμματισμός, LOW=λειτουργία

Εικόνα 3.8: Bluetooth Module HC-06 – Pins

Κάθε Bluetooth Module έχει ένα όνομα, έναν κωδικό πρόσβασης και έναν **ρυθμό μετάδοσης συμβόλων** (baud rate), που είναι η μετάδοση αναλογικά διαμορφωμένων ψηφιακών σημάτων. Στην περίπτωσή μας ο **κωδικός πρόσβασης** του bluetooth module είναι **1234** κι ο ρυθμός μετάδοσης του είναι **9600 bauds per second**. Παρακάτω θα προγραμματίσουμε το bluetooth για να αλλάξουμε το **όνομα** του σε **YIANNIS.** Ο χρήστης θα ενεργοποιεί το bluetooth στον υπολογιστή του, στο smartphone ή στο tablet του και θα ανιχνεύει το bluetooth module με το **όνομα YIANNIS** και πληκτρολογώντας τον **κωδικό πρόσβασης** που είναι **1234** θα κάνει **σύζευξη** της συσκευής του με τη συσκευή bluetooth.

- **Βήματα για τον προγραμματισμό του bluetooth module HC-06**

Για να προγραμματίσουμε το Bluetooth module χρησιμοποιήσαμε το Arduino Uno board χωρίς τον **μικροελεγκτή ATMega328P**, οπότε με προσοχή τον αφαιρούμε πριν ξεκινήσουμε.

- Συνδέουμε το **RXD του Bluetooth** με το **RXD του Arduino** και το **TXD του Bluetooth** με το **TXD του Arduino.**
- Συνδέουμε το **VCC του Bluetooth** με το **5V του Arduino** και το **GND του Bluetooth** με το **GND του Arduino.**
- Συνδέουμε το **WAKEUP** με μια **αντίσταση 10 Kohm** στο **VCC του Bluetooth (3.3V)** - **Προγραμματισμός (WAKEUP:HIGH).**
- Συνδέουμε το Arduino Uno με τον υπολογιστή μέσω του USB καλωδίου.
- Ανοίγουμε την επικοινωνία στο **serial monitor** του Arduino IDE.
- Επιλέγουμε τον σωστό **ρυθμό μετάδοσης** (baud rate) που έχει ήδη το bluetooth module ο οποίος είναι **9600**.
- Μετά την σύνδεση επιλέγουμε **"No line ending".**
- Πληκτρολογούμε **AT**, εάν πάρουμε απάντηση **OK** τότε είμαστε σε θέση να προγραμματίσουμε το Bluetooth.
- Για αλλαγή του ονόματος πληκτρολογούμε **AT+NAMEYIANNIS**. Το Bluetooth απαντάει **Oksetname**. Αν θέλουμε να αλλάξουμε τον κωδικό πληκτρολογούμε **AT+PINνέος_κωδικός.** Το Bluetooth απαντάει **Οκsetpin**. Αν θέλουμε να αλλάξουμε το ρυθμό μετάδοσης **AT+BAUDn** (n=1, 2, 3, 4, 5, 6, 7, 8). Το Bluetooth απαντάει **OKbaudrate** (όπου n=1, 2, 3, 4, 5, 6, 7, 8 αντίστοιχα 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200).
- Αποσυνδέουμε το **WAKEUP** από την **αντίσταση** και από το **VCC του Bluetooth (WAKEUP:LOW)** και το Bluetooth είναι έτοιμο για **λειτουργία** με το **νέο όνομα (YIANNIS)**.

ΚΕΦΑΛΑΙΟ 4

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΤΟΥ ARDUINO UNO

Σε αυτό το κεφάλαιο αναλύονται τα βήματα προγραμματισμού του Arduino Uno για την ελεγχόμενη κίνηση του ρομποτικού μας οχήματος και γράφονται οι κώδικες με τους οποίους προγραμματίζουμε το Arduino Uno μέσα από το προγραμματιστικό περιβάλλον του, το Arduino IDE.

4.1 Το προγραμματιστικό περιβάλλον (software) του Arduino Uno

Το πρόγραμμα Arduino IDE και η σύνδεση του Arduino Uno με τον υπολογιστή

Την τελευταία έκδοση του προγράμματος Arduino IDE (Εικόνα 4.1) μπορούμε να την κατεβάσουμε από τον επίσημο ιστότοπο για καθένα από τα τρία δημοφιλέστερα λειτουργικά συστήματα (Windows, Mac OS και Linux). Το Arduino IDE είναι βασισμένο σε Java και συγκεκριμένα παρέχει:

1) Ένα πρακτικό περιβάλλον για την συγγραφή των προγραμμάτων-κωδικών μας (τα οποία ονομάζονται sketch στην ορολογία του Arduino) με συντακτική χρωματική σήμανση.

2) Αρκετά έτοιμα παραδείγματα (examples).

3) Μερικές έτοιμες βιβλιοθήκες (libraries) για προέκταση της γλώσσας και για να χειριζόμαστε εύκολα μέσα από τον κώδικα μας τα εξαρτήματα που συνδέουμε στο Arduino.

4) Τον compiler για τη μεταγλώττιση των κωδικών μας.

5) Ένα serial monitor που παρακολουθεί τις επικοινωνίες της σειριακής (usb), αναλαμβάνει να στείλει αλφαριθμητικά τις εντολές μας στο Arduino μέσω αυτής και είναι ιδιαίτερα χρήσιμο για το debugging των κωδικών μας.

6) Την επιλογή να ανεβάσουμε τον μεταγλωττισμένο κώδικα στο Arduino.

Για να λειτουργήσουν βέβαια οι επιλογές 5 και 6 στο πρόγραμμα Arduino IDE, το Arduino Uno πρέπει να έχει συνδεθεί σε μια από τις θύρες USB του

υπολογιστή μας και, λόγω του ελεγκτή Serial-over-USB (Εικόνα 1.6), θα πρέπει να αναγνωριστεί από το λειτουργικό μας σύστημα ως εικονική σειριακή θύρα.

Αν όλα έγιναν σωστά, το κεντρικό παράθυρο του Arduino IDE θα εμφανιστεί όταν το εκτελέσουμε και στο μενού Tools –> Port θα πρέπει να εμφανίζεται η εικονική σειριακή θύρα (συνήθως COM για τα Windows, /dev/ttyusbserial για το Mac OS και /dev/ttyusb για το Linux).

Επιλέγουμε την εικονική σειριακή θύρα COM για τα Windows και στη συνέχεια επιλέγουμε τον τύπο πλακέτας του Arduino μας (Arduino/Genuino Uno) από το μενού Tools –> Board.

Το Arduino Uno είναι πλέον έτοιμο να δεχτεί τους κώδικες μας.

Εικόνα 4.1: Arduino IDE

Γλώσσα προγραμματισμού

Η γλώσσα του Arduino βασίζεται στη γλώσσα Wiring, μια παραλλαγή C/C++ για μικροελεγκτές αρχιτεκτονικής AVR όπως ο ATmega και υποστηρίζει όλες τις βασικές δομές της C καθώς και μερικά χαρακτηριστικά της C++. Για compiler χρησιμοποιείται ο AVR gcc και ως βασική βιβλιοθήκη C χρησιμοποιείται η AVR libc.

Λόγω της καταγωγής της από την C, στην γλώσσα του Arduino μπορούμε να χρησιμοποιήσουμε ουσιαστικά τις ίδιες βασικές εντολές και συναρτήσεις, με την

ίδια σύνταξη, τους ίδιους τύπους δεδομένων και τους ίδιους τελεστές όπως και στην C.

Στη γλώσσα του Arduino κάθε πρόγραμμα αποτελείται από δύο βασικές ρουτίνες ώστε να έχει την γενική δομή:

```
//Ενσωματώσεις βιβλιοθηκών(#include), δηλώσεις μεταβλητών(int,const int)

void setup() { //βασική ρουτίνα(μια φορά)
// συναρτήσεις pinMode(pin,OUTPUT);
}

void loop() { //βασική ρουτίνα-βρόγχος(επανάληψη)
// συναρτήσεις, συνθήκες if(){}..else{}, βρόγχοι επανάληψης for(){}
}

//Υπόλοιπες συναρτήσεις, υπορουτίνες void (){}
```

Η βασική ρουτίνα setup() εκτελείται μια φορά μόνο κατά την εκκίνηση του προγράμματος ενώ η βασική ρουτίνα loop() περιέχει τον βασικό κορμό του προγράμματος και η εκτέλεσή της επαναλαμβάνεται συνέχεια σαν ένας βρόγχος while (true).

4.2 Έλεγχος λειτουργίας των μοτέρ (DC motors) του ρομποτικού οχήματος

Αρχικά τεστάρουμε τα μοτέρ για να διαπιστώσουμε αν όντως το ρομποτικό όχημα κινείται κανονικά σύμφωνα με τις εντολές που του δίνουμε μέσα στον κώδικα. Επομένως προγραμματίζουμε το Arduino Uno με τον συγκεκριμένο κώδικα ώστε το ρομποτικό όχημα να εκτελεί κινήσεις προς όλες τις κατευθύνσεις και με τη χρήση της συνάρτησης delay (duration) ορίζουμε τη διάρκεια (duration) με την οποία θα εκτελεί την κάθε κίνηση. Μετά τον προγραμματισμό του Arduino Uno τοποθετούμε τις 4 AA μπαταρίες και προς το παρών δεν βάζουμε τη μπαταρία των 9V. Έχουμε συνδεδεμένο το Arduino στον υπολογιστή κι ανοίγουμε την επικοινωνία (serial monitor) στο Arduino IDE για να εκτελεστεί ο κώδικας και

να κάνει το ρομποτικό όχημα τις κινήσεις με τη σειρά σύμφωνα με τις εντολές που του έχουμε δώσει μέσα στον κώδικα.

Ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο λειτουργίας των μοτέρ (DC motors) του ρομποτικού οχήματος φαίνεται στο Παράρτημα Α΄.

4.3 Κίνηση του ρομποτικού οχήματος στο χώρο

Στη συνέχεια προγραμματίζουμε το Arduino Uno έτσι ώστε το ρομποτικό όχημα να κινείται μόνο του στο χώρο και να αποφεύγει τα εμπόδια με τη χρήση του αισθητήρα υπερήχων (ultrasonic sensor HC – SR04) και των διακοπτών “επαφής” (micro switches).

Τοποθετούμε τώρα και τη μπαταρία των 9V αφού προγραμματίσουμε το Arduino Uno. Στον κώδικά μας χρησιμοποιούμε τις βιβλιοθήκες Ultrasonic.h και Servo.h.

Περιγραφή Λειτουργίας Ρομποτικού οχήματος

Όταν το ρομποτικό όχημα ανιχνεύσει αντικείμενο σε απόσταση μικρότερη ή ίση με 15cm τότε σβήνει το πράσινο led (που είναι αναμμένο όταν κινείται το ρομποτικό όχημα) κι ανάβει το κόκκινο led, ενεργοποιείται ο buzzer που βγάζει ήχο, σταματάει το ρομποτικό όχημα, κοιτάζει αριστερά και δεξιά, διαβάζει τις αποστάσεις και ψάχνει τον καλύτερο τρόπο διαφυγής. Στη συνέχεια στρίβει κατά 90 μοίρες δεξιά ή αριστερά και μετά κινείται μπροστά ενώ ο buzzer απενεργοποιείται. Κατά την πορεία του είναι πιθανόν να πέσει πάνω σε αντικείμενα ή εμπόδια που δεν μπορεί να ανιχνεύσει. Οι διακόπτες επαφής όταν ακουμπήσουν σε κάποιο αντικείμενο το ρομποτικό όχημα κάνει λίγο πίσω και μετά μια μικρή στροφή αριστερά ή δεξιά ανάλογα με το ποιος διακόπτης επαφής ακουμπήσει για να το αποφύγει. Και σ' αυτή την περίπτωση σβήνει το πράσινο led κι ανάβει το κόκκινο led.

Ο κώδικας προγραμματισμού του Arduino Uno για την κίνηση του ρομποτικού οχήματος στο χώρο αποφεύγοντας τα εμπόδια φαίνεται στο Παράρτημα Β΄.

4.4 Έλεγχος κίνησης του ρομποτικού οχήματος με την τεχνολογία Bluetooth

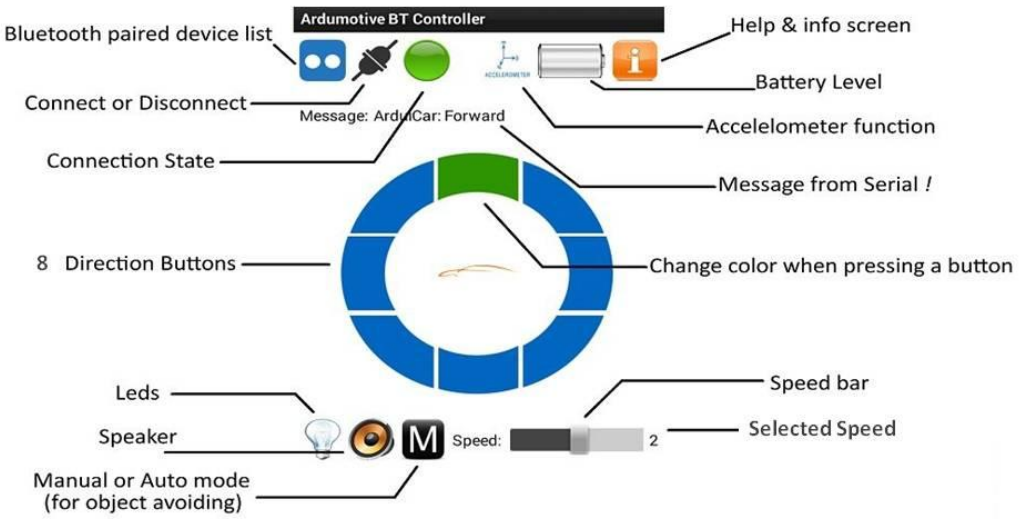
Μετά προγραμματίζουμε το Arduino Uno ώστε να ελέγχουμε την κίνηση του ρομποτικού οχήματος μέσα από μια Android συσκευή (smartphone ή tablet) χρησιμοποιώντας την τεχνολογία bluetooth.

Κάθε φορά που περνάμε πρόγραμμα στο Arduino Uno πρέπει να αποσυνδέουμε τα καλώδια απ' τα pins RXD και TXD.

Το τρέχων πρόγραμμα μας δίνει τη δυνατότητα να ελέγξουμε το ρομποτικό μας όχημα και να το κινήσουμε προς όλες τις κατευθύνσεις μέσω της android εφαρμογής **ardumotivebt_v2.1** (Εικόνα 4.2). Δεν κάνει χρήση του servo motor, των διακοπτών “επαφής”, του ultrasonic sensor, των leds και του buzzer.

Ανάλογα με το κουμπί που πατάει ο χρήστης μέσα στην εφαρμογή δίνεται και η αντίστοιχη εντολή στο Arduino Uno σύμφωνα με τη συνθήκη if που έχουμε γράψει στον κώδικά μας για κάθε κουμπί.

Ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο κίνησης του ρομποτικού οχήματος με την τεχνολογία Bluetooth φαίνεται στο Παράρτημα Γ΄.



Εικόνα 4.2: Android Application ardumotivebt_v2.1

Σημείωση: Στην παρούσα φάση σύμφωνα με τον κώδικα που έχουμε προγραμματίσει το Arduino Uno οι επιλογές που λειτουργούν μέσα στην εφαρμογή είναι τα κουμπιά αριστερά, δεξιά, πίσω και μπροστά (8 Direction Buttons – Εικόνα 4.2). Στην περίπτωσή μας χρησιμοποιούνται τα 4 Direction Buttons (left, right, backward, forward). Όταν ο χρήστης πατάει ένα direction

button αυτό αλλάζει το μπλε χρώμα του σε πράσινο (Change color when pressing a button – Εικόνα 4.2).

4.5 Έλεγχος χειροκίνητης κι αυτόματης λειτουργίας του ρομποτικού οχήματος

Τώρα προγραμματίζουμε το Arduino Uno ώστε το ρομποτικό όχημα να εκτελεί και τις 2 λειτουργίες, δηλαδή και να κινείται μόνο του ελεύθερα στο χώρο αλλά και να το κινεί ο χρήστης μέσω της android εφαρμογής ardumotivebt_v2.1 (Εικόνα 4.2).

Η Android εφαρμογή ardumotivebt_v2.1 και οι λειτουργίες της

Αρχικά ο χρήστης επιλέγει το ρομποτικό όχημα που θέλει να συνδεθεί μέσω bluetooth μέσα από την εφαρμογή της android συσκευής του στη λίστα με τις συζευγμένες bluetooth συσκευές (Bluetooth paired device list - Εικόνα 4.2) κι αφού επιλέξει τη συσκευή που επιθυμεί πατώντας πάνω σε αυτή (στην περίπτωσή μας είναι η συσκευή YIANNIS) στη συνέχεια πατάει το εικονίδιο της πρίζας (Connect or Disconnect – Εικόνα 4.2) για να συνδεθεί (connect). Όταν γίνει πράσινος ο κύκλος δίπλα στην πρίζα (Connection State - Εικόνα 4.2) σημαίνει ότι το smartphone ή tablet έχει συνδεθεί επιτυχώς μέσω bluetooth με το ρομποτικό όχημα μέσα από την εφαρμογή και μπορεί πια ο χρήστης να το ελέγξει.

Το ρομποτικό όχημα ξεκινάει στη χειροκίνητη λειτουργία κι ο χρήστης μπορεί να το κινεί προς όλες τις κατευθύνσεις πατώντας τα αντίστοιχα κουμπιά μέσα στην εφαρμογή (Direction Buttons – Εικόνα 4.2) και με το πάτημα ενός κουμπιού (Manual or Auto mode – Εικόνα 4.2) μπορεί να κάνει εναλλαγή χειροκίνητης κι αυτόματης λειτουργίας. Το ρομποτικό όχημα όταν μπαίνει στην αυτόματη λειτουργία κινείται μόνο του στο χώρο αποφεύγοντας τα εμπόδια.

Ο χρήστης μπορεί να ρυθμίσει και την ταχύτητα μετακινώντας τον δείκτη μπάρας (Speed bar – Εικόνα 4.2) αριστερά για να χαμηλώσει την ταχύτητα και δεξιά για να ανεβάσει την ταχύτητα (Speed – Εικόνα 4.2) του ρομποτικού οχήματος και στη χειροκίνητη και στην αυτόματη λειτουργία. Οι ταχύτητες είναι από 0 μέχρι 4 (Selected speed – Εικόνα 4.2).

Αν θέλει ο χρήστης να αποσυνδέσει (disconnect) την android συσκευή του από το ρομποτικό όχημα θα ξαναπατάει το εικονίδιο της πρίζας (Connect or

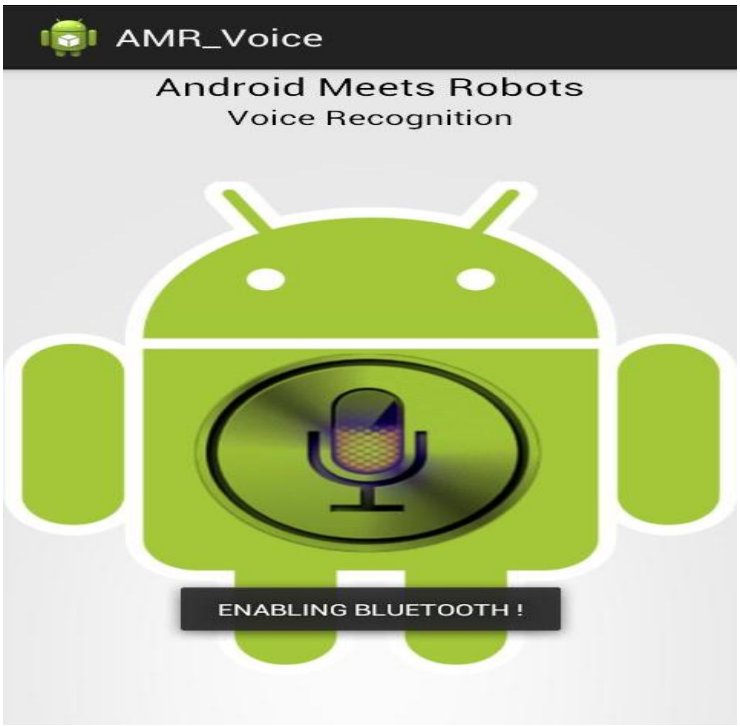
Disconnect – Εικόνα 4.2) κι όταν ο κύκλος δίπλα στην πρίζα γίνει κόκκινος (Connection State - Εικόνα 4.2) σημαίνει ότι έχει αποσυνδεθεί επιτυχώς από το ρομποτικό όχημα.

Ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο της χειροκίνητης κι αυτόματης λειτουργίας του ρομποτικού οχήματος φαίνεται στο Παράρτημα Δ΄.

4.6 Έλεγχος κίνησης του ρομποτικού οχήματος με φωνητικές εντολές

Επίσης μπορούμε να προγραμματίσουμε το Arduino Uno για να κινήσουμε το ρομποτικό μας όχημα προς όλες τις κατευθύνσεις με φωνητικές εντολές.

Η Android εφαρμογή που θα χρησιμοποιήσουμε λέγετε **"BT Voice Control for Arduino"** και μπορούμε να την κατεβάσουμε δωρεάν από το play store(Εικόνα 4.3).



Εικόνα 4.3: AMR_Voice Voice Recognition

Αυτό που κάνει η εφαρμογή (Εικόνα 4.3) είναι να “αναγνωρίζει” τις φωνητικές εντολές και να στέλνει τις λέξεις σειριακά στο Arduino (μέσω Bluetooth) οι οποίες ξεκινάνε με '*' και τελειώνουν με '#'. Το Arduino "καταλαβαίνει" μόνο λατινικούς χαρακτήρες οπότε οι λέξεις με τις οποίες θα δίνουμε την φωνητική εντολή πρέπει

να είναι στην Αγγλική γλώσσα. Κάνουμε click στο κουμπί (button) με το μικρόφωνο για να πούμε τη φωνητική εντολή, όπως φαίνεται στην εικόνα 4.4.

Εικόνα 4.4: AMR_Voice Talk to Your Robot

- **Οι φωνητικές εντολές**
  - **On - *On#** (για να ενεργοποιηθούν τα μοτέρ)
  - **Forward - *Forward#** (για να κινηθεί μπροστά)
  - **Backward - *Backward#** (για να κινηθεί πίσω)
  - **Left - *Left#** (για να κάνει στροφή αριστερά)
  - **Right - *Right#** (για να κάνει στροφή δεξιά)
  - **Stop - *Stop#** (για να σταματήσει)
  - **Off - *Off#** (για να απενεργοποιηθούν τα μοτέρ)

Καμιά φορά η εφαρμογή στέλνει τις λέξεις με το πρώτο γράμμα μικρό. Αυτό μπορούμε να το ελέγξουμε μέσα στη συνθήκη if του παρακάτω κώδικα. Αν μια εντολή δεν στέλνεται σωστά επειδή δεν μπορούμε να την προφέρουμε σωστά μπορούμε να δοκιμάσουμε κάποια άλλη λέξη συνώνυμη και να την προσθέσουμε στον κώδικα. Για παράδειγμα για να κινηθεί προς τα πίσω το ρομποτικό όχημα μπορούμε να του πούμε Backward ή Reverse, οπότε η εντολή που περιμένει το Arduino μπορεί να είναι *Backward# ή *backward# ή *Reverse# ή *reverse#.

Ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο κίνησης του ρομποτικού οχήματος με φωνητικές εντολές φαίνεται στο Παράρτημα Ε΄.

ΚΕΦΑΛΑΙΟ 5

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΟΠΤΙΚΕΣ

5.1 Σύνοψη της πτυχιακής εργασίας

Στην παρούσα πτυχιακή εργασία αφού κάναμε μια εισαγωγή στο Arduino Uno κι αναλύσαμε τους ακροδέκτες του (pins) για να κατανοηθεί η λειτουργία του, στη συνέχεια ασχοληθήκαμε με την κατασκευή ρομποτικού οχήματος αναλύοντας με κάθε λεπτομέρεια όλα τα βήματα υλοποίησης της κατασκευής του αλλά και του προγραμματισμού του Arduino Uno ώστε να μπορούμε να ελέγχουμε την κίνηση του ρομποτικού οχήματος μέσω android εφαρμογών με την τεχνολογία Bluetooth χρησιμοποιώντας smartphone ή tablet.

5.2 Συμπεράσματα

Η κατασκευή του ρομποτικού οχήματος καθώς και τα βήματα προγραμματισμού του Arduino Uno ολοκληρώθηκαν με επιτυχία και μπορούμε να ελέγχουμε την κίνηση του ρομποτικού οχήματος δίνοντας του εντολές μέσα από android εφαρμογές ανάλογα με τον κώδικα με τον οποίο έχουμε προγραμματίσει το Arduino Uno και μπορούμε να τροποποιήσουμε τους κώδικες για να αλλάξουμε την πορεία κίνησης του ρομποτικού οχήματος ή τον τρόπο με τον οποίο ελέγχουμε την κίνηση του.

5.3 Προοπτικές

Μέσα από την παρούσα πτυχιακή εργασία διαπιστώσαμε ότι το Arduino Uno είναι πολύ οικονομικό σε σχέση με τη χρησιμότητά του και τις δυνατότητες που έχει. Η συγκεκριμένη κατασκευή που ολοκληρώσαμε εκτός από τις λειτουργίες που μπορεί να εκτελέσει έχει και προοπτικές. Μπορούμε για παράδειγμα να εγκαταστήσουμε κάμερα που να καταγράφει την πορεία του ρομποτικού οχήματος όταν κινείται.

ΚΕΦΑΛΑΙΟ 6

ΠΑΡΑΡΤΗΜΑΤΑ

6.1 Παράρτημα Α'

Στο παράρτημα αυτό παρατίθεται ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο λειτουργίας των μοτέρ (DC motors) του ρομποτικού οχήματος.

test_motors.ino

```
//Testing the DC Motors

//Define Pins

//Motor A
const int enableA = 5;
const int motorA1 = 11;
const int motorA2 = 10;

//Motor B
const int enableB = 6;
const int motorB1 = 9;
const int motorB2 = 12;

void setup() {

 //configure pin modes
 pinMode (enableA, OUTPUT);
 pinMode (motorA1, OUTPUT);
 pinMode (motorA2, OUTPUT);

 pinMode (enableB, OUTPUT);
 pinMode (motorB1, OUTPUT);
 pinMode (motorB2, OUTPUT);
```

```
  //Test movements
enableMotors();
delay(2000);
forward();
delay(4000);
backward();
delay(4000);
right();
delay(5000);
left();
delay(5000);
brakeRobot();
delay(2000);
disableMotors();
}

void loop()
{
  //empty for now....
}

//Defining functions

//motor functions
void motorAforward() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, LOW);
}
void motorBforward() {
 digitalWrite (motorB1, LOW);
 digitalWrite (motorB2, HIGH);
}
```

```
void motorAbackward() {
 digitalWrite (motorA1, LOW);
 digitalWrite (motorA2, HIGH);
}
void motorBbackward() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, LOW);
}
void motorAstop() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, HIGH);
}
void motorBstop() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, HIGH);
}
void motorAon() {
 digitalWrite (enableA, HIGH);
}
void motorBon() {
 digitalWrite (enableB, HIGH);
}
void motorAoff() {
 digitalWrite (enableA, LOW);
}
void motorBoff() {
 digitalWrite (enableB, LOW);
}
// Movement functions
void forward () {
 motorAforward();
 motorBforward();
}
```

```
void backward () {
 motorAbackward();
 motorBbackward();
}
void left () {
 motorAforward();
 motorBbackward();
}
void right () {
 motorAbackward();
 motorBforward();
}
void brakeRobot () {
 motorAstop();
 motorBstop();
}
void disableMotors() {
 motorAoff();
 motorBoff();
}
void enableMotors() {
 motorAon();
 motorBon();
}
```

6.2 Παράρτημα Β'

Στο παράρτημα αυτό παρατίθεται ο κώδικας προγραμματισμού του Arduino Uno για την κίνηση του ρομποτικού οχήματος στο χώρο αποφεύγοντας τα εμπόδια.

avoid_objects.ino

```
//Avoid Objects

//Libraries

#include "Ultrasonic.h"
#include <Servo.h>

//Define Pins

//Motor A
const int enableA = 5;
const int motorA1 = 11;
const int motorA2 = 10;

//Motor B
const int enableB = 6;
const int motorB1 = 9;
const int motorB2 = 12;

const int servo = 3;

const int buzzer = 4;

const int redled = 2;
const int greenled = 13;

const int rightSW = 8;
const int leftSW  = 7;
```

```
Ultrasonic ultrasonic(A0,A1); // Trig, Echo
Servo myservo;

//Variables
int distance=100;
int rightDistance;
int leftDistance;
int rightSWState;
int leftSWState;
int pos = 90; // for servo, 90 degrees looking forward

void setup()
{
 //configure pin modes
 pinMode (enableA, OUTPUT);
 pinMode (motorA1, OUTPUT);
 pinMode (motorA2, OUTPUT);

 pinMode (enableB, OUTPUT);
 pinMode (motorB1, OUTPUT);
 pinMode (motorB2, OUTPUT);

 pinMode(leftSW,  INPUT_PULLUP);
 pinMode(rightSW, INPUT_PULLUP);

 pinMode (buzzer, OUTPUT);

 pinMode (redled, OUTPUT);
 pinMode (greenled, OUTPUT);

 //Initialize servo and starting position
 myservo.attach(servo);
```

```
  myservo.write(pos);

  //Wait 5 seconds before start
  delay(5000);
}
void loop()
{
  enableMotors();
  noTone(buzzer);
  greenledON();
  redledOFF();

  distance = ultrasonic.Ranging(CM);

  //If an object detected at 15cm, stop the robot and find a way out
  if (distance <= 15){
    brakeRobot();
    greenledOFF();
    redledON();
    tone(buzzer,500);
    // look left and read left distance
    for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
       myservo.write(pos);              // tell servo to go to position in variable position
       delay(5);                        // waits 5ms for the servo to reach the position
     }

     leftDistance = ultrasonic.Ranging(CM);
     delay(100);

     // look right and read right distance
     for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
       myservo.write(pos);              // tell servo to go to position in variable position
       delay(5);                        // waits 5ms for the servo to reach the position
```

```
    }

    rightDistance = ultrasonic.Ranging(CM);
    delay(100);

    pos = 90; // look forward
    myservo.write(pos);

    //Finally compare left and right distances and make the best turn decision
    if (leftDistance > rightDistance){
      left();
      delay(500);
    }
    else if (leftDistance < rightDistance){
      right();
      delay(500);
    }
    else{ //if the two distances are equal
      backward();
      delay(1000);// Go back for 1 s
      left();
      delay(500);
    }

}
//All clear, move forward! And check left and right "contact" switches.
else{
  leftSWState = digitalRead(leftSW);
  rightSWState = digitalRead(rightSW);

  //Check for "contact"
  if (leftSWState == LOW){
    greenledOFF();
```

```
    redledON();
    backward();
    delay(500);
    right();
    delay(400);
   }
   if (rightSWState == LOW){
    greenledOFF();
    redledON();
    backward();
    delay(500);
    left();
    delay(400);
   }

   forward();
 }
}

//motor functions
void motorAforward() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, LOW);
}
void motorBforward() {
 digitalWrite (motorB1, LOW);
 digitalWrite (motorB2, HIGH);
}
void motorAbackward() {
 digitalWrite (motorA1, LOW);
 digitalWrite (motorA2, HIGH);
}
void motorBbackward() {
```

```
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, LOW);
}
void motorAstop() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, HIGH);
}
void motorBstop() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, HIGH);
}
void motorAon() {
 digitalWrite (enableA, HIGH);
}
void motorBon() {
 digitalWrite (enableB, HIGH);
}
void motorAoff() {
 digitalWrite (enableA, LOW);
}
void motorBoff() {
 digitalWrite (enableB, LOW);
}
//led functions
void greenledON () {
  digitalWrite(greenled,HIGH);
}
void greenledOFF () {
  digitalWrite(greenled,LOW);
}
void redledON () {
  digitalWrite(redled,HIGH);
}
```

```
void redledOFF () {
  digitalWrite(redled,LOW);
}
// Movement functions
void forward () {
 motorAforward();
 motorBforward();
}
void backward () {
 motorAbackward();
 motorBbackward();
}
void left () {
 motorAforward();
 motorBbackward();
}
void right () {
 motorAbackward();
 motorBforward();
}
void brakeRobot () {
 motorAstop();
 motorBstop();
}
void disableMotors() {
 motorAoff();
 motorBoff();
}
void enableMotors() {
 motorAon();
 motorBon();
}
```

6.3 Παράρτημα Γ'

Στο παράρτημα αυτό παρατίθεται ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο κίνησης του ρομποτικού οχήματος με την τεχνολογία Bluetooth.

test_bluetooth.ino

```
//Test Bluetooth

//Define Pins

//Motor A
const int enableA = 5;
const int motorA1 = 11;
const int motorA2 = 10;

//Motor B
const int enableB = 6;
const int motorB1 = 9;
const int motorB2 = 12;

//Variables
int state;

void setup()
{
  //configure pin modes
 pinMode (enableA, OUTPUT);
 pinMode (motorA1, OUTPUT);
 pinMode (motorA2, OUTPUT);

 pinMode (enableB, OUTPUT);
 pinMode (motorB1, OUTPUT);
 pinMode (motorB2, OUTPUT);
```

```
  //Start serial communication
  Serial.begin(9600);
}

void loop()
{
  if (Serial.available() > 0) {
      // read the state:
        state = Serial.read();
    }
    if (state=='E'){
      enableMotors();
    }
    else if (state=='F'){
      forward();
    }
    else if (state=='B'){
      backward();
    }
    else if (state=='R'){
      right();
    }
    else if (state=='L'){
      left();
    }
    else if (state=='S'){
      brakeRobot ();
    }
    else if (state=='D'){
      disableMotors();
    }
}
```

```
//motor functions
void motorAforward() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, LOW);
}
void motorBforward() {
 digitalWrite (motorB1, LOW);
 digitalWrite (motorB2, HIGH);
}
void motorAbackward() {
 digitalWrite (motorA1, LOW);
 digitalWrite (motorA2, HIGH);
}
void motorBbackward() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, LOW);
}
void motorAstop() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, HIGH);
}
void motorBstop() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, HIGH);
}
void motorAon() {
 digitalWrite (enableA, HIGH);
}
void motorBon() {
 digitalWrite (enableB, HIGH);
}
void motorAoff() {
 digitalWrite (enableA, LOW);
```

```
}
void motorBoff() {
 digitalWrite (enableB, LOW);
}
// Movement functions
void forward () {
 motorAforward();
 motorBforward();
}
void backward () {
 motorAbackward();
 motorBbackward();
}
void left () {
 motorAforward();
 motorBbackward();
}
void right () {
 motorAbackward();
 motorBforward();
}
void brakeRobot () {
 motorAstop();
 motorBstop();
}
void disableMotors() {
 motorAoff();
 motorBoff();
}
void enableMotors() {
 motorAon();
 motorBon();
}
```

6.4 Παράρτημα Δ'

Στο παράρτημα αυτό παρατίθεται ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο της χειροκίνητης κι αυτόματης λειτουργίας του ρομποτικού οχήματος.

dual_function.ino

```
//Arduino Robot Vehicle Created By Ioannis Belesis
//Bluetooth Controlled and Object Avoiding Function

//Libraries

#include "Ultrasonic.h"
#include <Servo.h>

//Define Pins

//Motor A
const int enableA = 5;
const int motorA1 = 11;
const int motorA2 = 10;

//Motor B
const int enableB = 6;
const int motorB1 = 9;
const int motorB2 = 12;

//servo
const int servo = 3;

//buzzer
const int buzzer = 4;

//leds
const int redled = 2;
```

```
const int greenled = 13;

//contacts
const int rightSW = 8;
const int leftSW  = 7;

Ultrasonic ultrasonic(A0,A1); // Trig, Echo
Servo myservo;

//Variables
int distance=100;
int rightDistance;
int leftDistance;
int rightSWstate;
int leftSWstate;
int pos = 90; // for servo, 90 degrees looking forward
char mode = 'm';
int state;

void setup(){
  //configure pin modes
 pinMode (enableA, OUTPUT);
 pinMode (motorA1, OUTPUT);
 pinMode (motorA2, OUTPUT);

 pinMode (enableB, OUTPUT);
 pinMode (motorB1, OUTPUT);
 pinMode (motorB2, OUTPUT);

 pinMode(leftSW,  INPUT_PULLUP);
 pinMode(rightSW, INPUT_PULLUP);

 pinMode (buzzer, OUTPUT);
```

```
 pinMode (redled, OUTPUT);
 pinMode (greenled, OUTPUT);

  myservo.attach(servo);
  myservo.write(pos);

  // Initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

void loop(){

  noTone(buzzer); //buzzerOFF
  greenledOFF();
  redledOFF();
  //Save income data to variable 'state'
  if(Serial.available() > 0){
       state = Serial.read();
       Serial.println(state);

       if (state == 'M'){mode = 'a'; }
       if (state == 'A'){mode = 'm';
          brakeRobot ();
          pos=90;
          myservo.write(pos);
       }
    }
    //speeds
    if (state == '0'){
      speed0();}
    else if (state == '1'){
      speed1();}
```

```
else if (state == '2'){
  speed2();}
else if (state == '3'){
  speed3();}
else if (state == '4'){
  speed4();}

//Manual mode
if (mode == 'm'){
//movements
  if (state=='F'){
    forward();
}
  else if (state=='B'){
    backward();
  }
  else if (state=='R'){
    right();
  }
  else if (state=='L'){
    left();
  }
  else if (state=='S'){
    brakeRobot ();
  }
}
//Automatic mode
else if (mode == 'a'){
  greenledON();
  distance = ultrasonic.Ranging(CM);
//Object detected(ultrasonic sensor HC-SR04)
if (distance <= 15){
  brakeRobot ();
```

```
  greenledOFF();
  redledON();
  tone(buzzer,500); //buzzerON,500Hz
  //look left and read left distance
  for (pos = 0; pos <= 180; pos += 1) {
    myservo.write(pos);
     delay(5);
    }
   leftDistance = ultrasonic.Ranging(CM);

    delay(100);
    //look right and read right distance
    for (pos = 180; pos >= 0; pos -= 1) {
      myservo.write(pos);
      delay(5);
    }
    rightDistance = ultrasonic.Ranging(CM);

    delay(100);
    //look forward
    pos = 90;
    myservo.write(pos);
  //compare distances and move to avoid the object
    if (leftDistance > rightDistance){
      left();
      delay(500);
    }
    else if (leftDistance < rightDistance){
      right();
      delay(500);
    }
    else{
      backward();
```

```
        delay(1000);
        left();
        delay(500);
      }
  }
  else{
  //check contacts
    leftSWstate = digitalRead(leftSW);
    rightSWstate = digitalRead(rightSW);

    if (leftSWstate == LOW){
      greenledOFF();
      redledON();
      backward();
      delay(500);
      right();
      delay(400);
    }
    if (rightSWstate == LOW){
      greenledOFF();
      redledON();
      backward();
      delay(500);
      left();
      delay(400);
    }
  //All clear,move forward
    forward();
  }
  }
}

//motor functions
```

```
void motorAforward() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, LOW);
}
void motorBforward() {
 digitalWrite (motorB1, LOW);
 digitalWrite (motorB2, HIGH);
}
void motorAbackward() {
 digitalWrite (motorA1, LOW);
 digitalWrite (motorA2, HIGH);
}
void motorBbackward() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, LOW);
}
void motorAstop() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, HIGH);
}
void motorBstop() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, HIGH);
}
void motorAon() {
 digitalWrite (enableA, HIGH);
}
void motorBon() {
 digitalWrite (enableB, HIGH);
}
void motorAoff() {
 digitalWrite (enableA, LOW);
}
```

```
void motorBoff() {
 digitalWrite (enableB, LOW);
}
void disableMotors() {
 motorAoff();
 motorBoff();
}
void enableMotors() {
 motorAon();
 motorBon();
}
//speed functions
void speed0() {
  analogWrite(enableA,0);
  analogWrite(enableB,0);
}
void speed1() {
  analogWrite(enableA,150);
  analogWrite(enableB,150);
}
void speed2() {
  analogWrite(enableA,180);
  analogWrite(enableB,180);
}
void speed3() {
  analogWrite(enableA,200);
  analogWrite(enableB,200);
}
void speed4() {
  analogWrite(enableA,255);
  analogWrite(enableB,255);
}
//led functions
```

```
void greenledON () {
  digitalWrite(greenled,HIGH);
}
void greenledOFF () {
  digitalWrite(greenled,LOW);
}
void redledON () {
  digitalWrite(redled,HIGH);
}
void redledOFF () {
  digitalWrite(redled,LOW);
}
// Movement functions
void forward () {
 motorAforward();
 motorBforward();
}
void backward () {
 motorAbackward();
 motorBbackward();
}
void left () {
 motorAforward();
 motorBbackward();
}
void right () {
 motorAbackward();
 motorBforward();
}
void brakeRobot () {
 motorAstop();
 motorBstop();
}
```

6.5 Παράρτημα Ε'

Στο παράρτημα αυτό παρατίθεται ο κώδικας προγραμματισμού του Arduino Uno για τον έλεγχο κίνησης του ρομποτικού οχήματος με φωνητικές εντολές.

voice.ino

```
//Voice Control For Arduino Robot Vehicle

//Define Pins

//Motor A
const int enableA = 5;
const int motorA1 = 11;
const int motorA2 = 10;

//Motor B
const int enableB = 6;
const int motorB1 = 9;
const int motorB2 = 12;

//Variables
char state;
String command;

void setup()
{
 //configure pin modes
 pinMode (enableA, OUTPUT);
 pinMode (motorA1, OUTPUT);
 pinMode (motorA2, OUTPUT);

 pinMode (enableB, OUTPUT);
 pinMode (motorB1, OUTPUT);
```

```
 pinMode (motorB2, OUTPUT);



 // Initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

void loop()
{
  while(Serial.available()) {
      state = Serial.read();
      command.concat(state);
    }
    //voice commands
    if (command == "*on#" || command == "*On#") {
      enableMotors();
    }
    else if (command == "*forward#" || command == "*Forward#") {
      forward();
    }
    else if (command == "*backward#" || command == "*Backward#" || command==
"*reverse#" || command == "*Reverse#"){
      backward();
    }
    else if (command == "*left#" || command == "*Left#") {
      left();
      delay(500);
      brakeRobot();
    }
    else if (command == "*right#" || command == "*Right#" || command == "*write#") {
      right();
      delay(500);
      brakeRobot();
```

```
  }
  else if (command == "*stop#" || command == "*Stop#") {
    brakeRobot();
  }
  else if (command == "*off#" || command == "*Off#") {
    disableMotors();
  }
  //Wait for new command
  if (state == '#'){
      command="";
  }
  delay(100); //wait 100 ms
}

//motor functions
void motorAforward() {
 digitalWrite (motorA1, HIGH);
 digitalWrite (motorA2, LOW);
}
void motorBforward() {
 digitalWrite (motorB1, LOW);
 digitalWrite (motorB2, HIGH);
}
void motorAbackward() {
 digitalWrite (motorA1, LOW);
 digitalWrite (motorA2, HIGH);
}
void motorBbackward() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, LOW);
}
void motorAstop() {
 digitalWrite (motorA1, HIGH);
```

```
 digitalWrite (motorA2, HIGH);
}
void motorBstop() {
 digitalWrite (motorB1, HIGH);
 digitalWrite (motorB2, HIGH);
}
void motorAon() {
 digitalWrite (enableA, HIGH);
}
void motorBon() {
 digitalWrite (enableB, HIGH);
}
void motorAoff() {
 digitalWrite (enableA, LOW);
}
void motorBoff() {
 digitalWrite (enableB, LOW);
}
void disableMotors() {
 motorAoff();
 motorBoff();
}
void enableMotors() {
 motorAon();
 motorBon();
}
// Movement functions
void forward () {
 motorAforward();
 motorBforward();
}
void backward () {
 motorAbackward();
```

```
 motorBbackward();
}
void left () {
 motorAforward();
 motorBbackward();
}
void right () {
 motorAbackward();
 motorBforward();
}
void brakeRobot () {
 motorAstop();
 motorBstop();
}
```

ΔΙΑΔΙΚΤΥΑΚΕΣ ΠΗΓΕΣ

[1] http://www.projectmaniacs.gr/

[2] http://blog.whatgeek.com.pt/Arduino/l298-dual-h-bridge-motor-driver/

[3] http://www.ardumotive.com/

[4] https://grobotronics.com/

[5] https://www.arduino.cc/

[6] https://el.wikipedia.org/wiki/Bluetooth

[7] https://play.google.com/store/apps/details?id=robotspace.simplelabs.amr_voice&hl=el

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