Τεχνολογικό Εκπαιδευτικό Ίδρυμα Πελοποννήσου
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής Τ.Ε.

Κατασκευή μικρού κινητού οχήματος με την χρήση Arduino

Πτυχιακή εργασία

του Γιάνναρου Αναστάσιου

Α.Μ. 2011086

Επιβλέπων καθηγητής

Μπάρδης Γεώργιος

Σπάρτη

Δεκέμβριος 2017

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάση επιστημονικής παράφρασης.

Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων.

Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δε μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

Όνομα και Επώνυμο Συγγραφέα (Με Κεφαλαία): .....

Υπογραφή (Ολογράφως, χωρίς μονογραφή): .....

Ημερομηνία (Ημέρα – Μήνας – Έτος): .....

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της παρούσας πτυχιακής κ. Μπάρδη Γιώργο για τα ερεθίσματα που μου παρείχε κατά την διάρκεια της φοίτησης μου στην σχολή και για την άψογη συνεργασία που είχαμε τα χρόνια αυτά.

Θα ήθελα επίσης να ευχαριστήσω τους γονείς μου οι οποίοι πιστεύουν σε εμένα και με στηρίζουν καθημερινά.

ΠΕΡΙΛΗΨΗ

Το τελευταίο διάστημα η ρομποτική και οι αυτοματισμοί έχουν εισέλθει στην ζωή του μέσου ανθρώπου περισσότερο από ποτέ άλλοτε. Αυτό οφείλετε στις αυξανόμενες ανάγκες του ανθρώπου για διευκόλυνση στην καθημερινότητα του, στην ανάπτυξη της τεχνολογίας αλλά και στην μείωση του κόστους αυτής. Ταυτόχρονα συνεχίζει να αυξάνεται η χρήση κινητών συσκευών όπως και οι υπηρεσίες που προσφέρονται μέσα από αυτές. Πλέον ο μέσος άνθρωπος περνάει αρκετές ώρες της ημέρας του χρησιμοποιώντας κινητή συσκευή.

Η παρούσα πτυχιακή εργασία παρουσιάζει ένα αυτοματοποιημένο όχημα μικρού μεγέθους το οποίο μέσω αυτοματισμού αποφεύγει τα εμπόδια που συναντάει στον δρόμο του ενώ ταυτόχρονα επικοινωνεί ασύρματα με μία κινητή συσκευή και μέσω της χρήσης εφαρμογής σε αυτή δέχεται εντολές κίνησης.

Σκοπός της εργασίας αυτής είναι ο πειραματισμός με τις εμπλεκόμενες τεχνολογίες τόσο από την μεριά του Arduino και του αυτοματισμού όσο και από την μεριά της ανάπτυξης εφαρμογής για κινητές συσκευές.

Η πτυχιακή χωρίζεται στα εξής τρία κομμάτια:

1. Πληροφορίες για το Arduino και τα περιφερειακά του
2. Συναρμολόγηση και προγραμματισμός του οχήματος
3. Προγραμματισμός εφαρμογής για έλεγχο αυτού

Το πρώτο μέρος εμπεριέχει την εισαγωγή στην οποία αναλύεται ο μικροελεγκτής και δίνονται κάποια σημαντικά στοιχεία για αυτόν και την λειτουργία του. Στην συνέχεια υπάρχουν πληροφορίες για το Arduino ως συσκευή και το περιβάλλον του ενώ παρουσιάζονται και τα υποσυστήματα που απαρτίζουν το όχημα της παρούσας πτυχιακής εργασίας.

Στο δεύτερο μέρος παρουσιάζεται η διαδικασία συναρμολόγησης του οχήματος και ο τρόπος λειτουργίας του. Επίσης, εμπεριέχεται ο κώδικας ο οποίος αναλαμβάνει τον έλεγχο καλής λειτουργίας των υποσυστημάτων του. Τέλος, περιλαμβάνεται ο τελικός κώδικας προγραμματισμού του οχήματος που καθιστά δυνατή την λειτουργία του υπό την επιθυμητή μορφή.

Το τρίτο και τελευταίο μέρος, μελετάει την υλοποίηση της εφαρμογής απομακρυσμένου ελέγχου του οχήματος. Πιο συγκεκριμένα, παρουσιάζεται η έρευνα που έγινε για την επιλογή της φύσης της εφαρμογής όπως και για τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της. Τέλος, παρουσιάζεται ο κώδικας της και η λειτουργία της .

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΧΑΡΙΣΤΙΕΣ 5

ΠΕΡΙΛΗΨΗ 7

ΠΕΡΙΕΧΟΜΕΝΑ 9

1 ΕΙΣΑΓΩΓΗ 11

1.1 Η έννοια του μικροελεγκτή 11

1.2 Συμπεράσματα 14

2. ARDUINO 17

2.1 Γενικά για το Arduino 17

2.2 Βασικές Έννοιες 22

2.2.1 Arduino IDE 22

2.2.2 Βιβλιοθήκες 23

2.2.3 Πλακέτες Επέκτασης (Shields) 24

2.2.4 Αισθητήρες 25

2.2.5 Πρωτοτυποποίηση 26

2.3 Λίστα Εξαρτημάτων 28

2.3.1 Infiduino Uno R3 29

2.3.2 Breadboard 31

2.3.3 Βάση οχήματος 32

2.3.4 Αισθητήρας Απόστασης 33

2.3.5 Προσαρμογές Bluetooth 35

2.3.6 Κινητήρες 36

2.3.7 Συσκευή Ελέγχου Κινητήρα με το υποσύστημα L298 37

2.3.8 Οθόνη LCD Nokia 5510 39

2.3.9 Θήκη Μπαταριών 41

3. ΚΑΤΑΣΚΕΥΗ ΟΧΗΜΑΤΟΣ 44

3.1 Ανάλυση Βημάτων Συναρμολόγησης 44

3.2 Λειτουργία 49

3.3 Έλεγχος Σωστής Λειτουργίας 50

3.4 Προγραμματισμός Οχήματος 54

4. ΕΦΑΡΜΟΓΗ ΧΕΙΡΙΣΜΟΥ ΟΧΗΜΑΤΟΣ 59

4.1 Γλώσσα και Περιβάλλον Προγραμματισμού 59

4.1.1 Επιλογή Φύσης Εφαρμογής 59

4.1.2 Ορισμός Framework 61
4.1.3 Επιλογή Framework 62
4.1.4 React Native 63
4.1.5 Λειτουργία Εφαρμογής 64
4.1.6 Υλοποίηση Εφαρμογής 65
ΒΙΒΛΙΟΓΡΑΦΙΑ – ΑΝΑΦΟΡΕΣ 75

1 ΕΙΣΑΓΩΓΗ

1.1 Η έννοια του μικροελεγκτή

Ως μικροελεγκτή *(εικόνα 1.1.1)* χαρακτηρίζουμε κάθε ενιαίο ολοκληρωμένο κύκλωμα (Integrated Circuit) το οποίο αποτελείται από έναν η περισσότερους επεξεργαστές σε συνδυασμό με μνήμη και άλλα ενσωματωμένα υποσυστήματα όπως θύρες επικοινωνίας. Σε αντίθεση με τους γενικής χρήσης επεξεργαστές που βρίσκουμε στους ηλεκτρονικούς υπολογιστές, οι μικροελεγκτές χρησιμοποιούνται για την δημιουργία ενσωματωμένων συστημάτων τα οποία εξυπηρετούν κάποιον συγκεκριμένο σκοπό. Χαρακτηριστικό τους είναι το χαμηλό τους κόστος, ο εξειδικευμένος σκοπός που καλούνται να ικανοποιήσουν και η δυνατότητα τους να χρησιμοποιήσουν μία πληθώρα από εξωτερικά περιφερειακά με σκοπό την επίτευξη του στόχου αυτού.

Αναλυτικότερα, τα σημαντικότερα πλεονεκτήματα τους απέναντι στους επεξεργαστές γενικής χρήσης είναι τα εξής:

- θύρες και κύκλωμα αρχικοποίησης λειτουργίας. Ως αποτέλεσμα, είναι σε θέση να πραγματοποιήσουν μία ευρεία γκάμα εργασιών χωρίς να εξαρτώνται από άλλα εξωτερικά υποσυστήματα. Επίσης, δεν παρατηρούνται προβλήματα συνδεσιμότητας και συνεργασίας των ενσωματωμένων υποσυστημάτων καθώς λειτουργούν όλα ως μία κεντρική συσκευή. Έτσι, προσφέρεται απλουστευμένη διαδικασία

προγραμματισμού και λειτουργία τους αλλά και μεγαλύτερη αξιοπιστία.

- Μικρό μέγεθος κυκλώματος και χαμηλό κόστος απόκτησης και λειτουργίας.
- Μειωμένες εκπομπές ηλεκτρομαγνητικών παρεμβολών αλλά ταυτόχρονα και μειωμένη ευαισθησία σε αυτές.

Οι μικροελεγκτές χωρίζονται σε κατηγορίες ανάλογα με την χρήση για την οποία προορίζονται και τον αριθμό των ακροδεκτών τους ο οποίος είναι επίσης ανάλογος των bit που μπορούν να επεξεργαστούν.

Πιο συγκεκριμένα, οι μικροελεγκτές γενικής χρήσης διακρίνονται στις εξής κατηγορίες:

- Μικροελεγκτές 4 (τεσσάρων) ή 8 (οχτώ) bit:

Χαρακτηριστικό τους είναι το πολύ μικρό κόστος, η χαμηλή κατανάλωση ενώ ταυτόχρονα μπορούν να εξυπηρετήσουν μέχρι και 8 ακροδέκτες. Παραδείγματα τέτοιων μικροελεγκτών ανήκουν οι περισσότεροι των σειρών PIC, AVR και 8051.

- Μικροελεγκτές 8 (οχτώ) έως 32 (τριάντα δύο) bit

Στην κατηγορία αυτή ανήκουν μικροελεγκτές χαμηλού κόστους με μέτριο έως σχετικά μεγάλο αριθμό ακροδεκτών. Λόγω της μεγαλύτερης επεξεργαστικής ικανότητας αλλά και της ύπαρξης περισσότερων ακροδεκτών, είναι σε θέση να προσφέρουν μία ευρεία γκάμα περιφερειακών τα οποία επεκτείνουν τις δυνατότητες του μικροελεγκτή προσθέτοντας σε αυτόν πρωτόκολλα σύνδεσης και μετατροπείς αναλογικού προς ψηφιακό και αντιστρόφως σήμα.

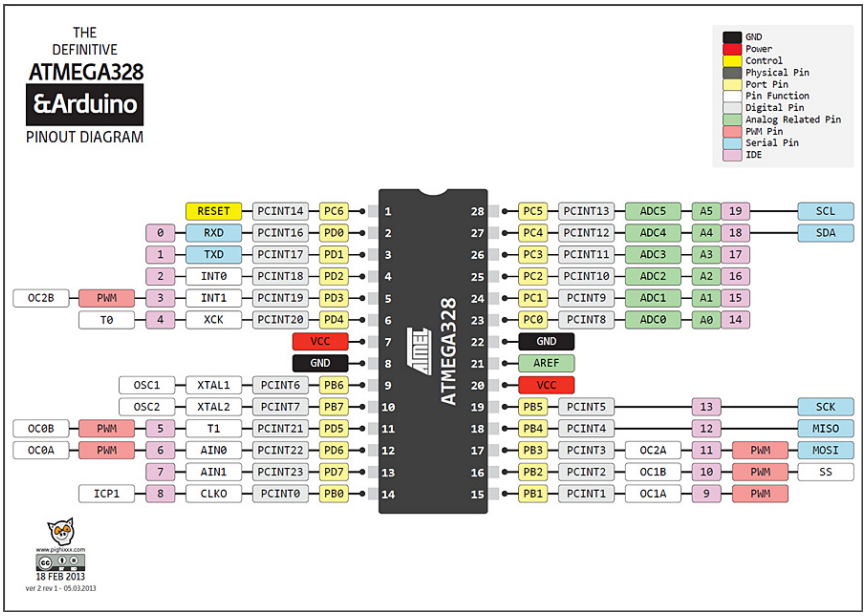
- Μικροελεγκτές 32 (τριάντα δύο) bit:

Στην τρίτη κατά σειρά κατηγορία, συναντάμε μικροελεγκτές μεσαίου κόστους με πολλούς ακροδέκτες. Εξαιτίας του κατά πολύ αυξημένου αριθμού των bit επεξεργασίας αλλά και τον ακροδεκτών, προσφέρουν πολύ μεγαλύτερη ταχύτητα αλλά και την δυνατότητα σύνδεσης πολλών περισσοτέρων περιφερειακών σε σχέση με τις προηγούμενες δύο κατηγορίες. Ιδιαίτερη αξία στους μικροελεγκτές αυτούς δίνει η δυνατότητα αναβάθμισης της μνήμης *Flash* και *RAM* μέσω εξωτερικού περιφερειακού. Επίσης, το λογισμικού τους χαρακτηρίζεται από υψηλή μεταφερσιμότητα το οποίο έχει ως αποτέλεσμα μεγάλο μέρος των εντολών που δέχονται να είναι κοινό ανάμεσα στις διαφορετικές υλοποιήσεις μικροελεγκτών.

Τέλος, υπάρχει και μία κατηγορία μικροελεγκτών στους οποίους ανήκουν αυτοί που προορίζονται για εξειδικευμένη χρήση και προσφέρουν ειδικά υλοποιημένο υλικό το οποίο έχει κατασκευαστεί για να εξυπηρετεί μία πολύ συγκεκριμένη εργασία.

Ένα από τα σημαντικότερα σημεία που χαρακτηρίζουν τον μικροελεγκτή είναι τα εργαλεία ανάπτυξης κώδικα στα οποία περιλαμβάνονται το περιβάλλον προγραμματισμού (IDE) και τα εργαλεία αποσφαλμάτωσης (debuggers). Η γλώσσα προγραμματισμού παίζει επίσης σημαντικό ρόλος στην επιλογή μικροελεγκτή. Έτσι, οι περισσότεροι μικροελεγκτές χρησιμοποιούν πλέον την C ή κάποιες παραλλαγές της καθώς έχουν ως στόχο την ευκολότερη εμπειρία χρήσης.

Οι μικροελεγκτές συναντιούνται κυρίως σε συστήματα αυτόματου ελέγχου όπως ο έλεγχος θερμοκρασίας του ψυγείου, το υποσύστημα ρύθμισης της πρόσφυσης ενός οχήματος ή η θέρμανση με πετρέλαιο ενός σπιτιού. Εξυπηρετούν επίσης και εφαρμογές με λιγότερο ενεργοβόρες και απαιτητικές εργασίες όπως το σύστημα ενός τηλεχειριστηρίου, ενός τρυπανιού ή ακόμα και ενός παιδικού παιχνιδιού όπως ένα τηλεχειριζόμενο όχημα. Τέλος, μπορούν να βρεθούν σε συσκευές καθημερινής χρήσης όπως ένας φούρνος μικροκυμάτων, τηλεόραση, ραδιοφωνική συσκευή, εκτυπωτές ή ακόμα και σε λάμπες τεχνολογίας LED.



Εικόνα 1.1.1: Μικροελεγκτής ATMEGA328

1.2 Συμπεράσματα

Οι μικροελεγκτές βρίσκουν πολλές εφαρμογές στην καθημερινή μας ζωή κάνοντας την πιο εύκολη. Ξεκινώντας από τους πρώτους μικροελεγκτές με πολύ περιορισμένη μνήμη και επεξεργαστική ισχύ οι οποίοι

προγραμματίζονταν μόνο με την χρήση γλώσσας μηχανής (Assembly), οι τεχνολογικές εξελίξεις σε συνδυασμό με τις ολοένα αυξανόμενες ανάγκες του ανθρώπου για εύκολα, απλά και απροβλημάτιστα συστήματα χαμηλού κόστους οδήγησε στους μικροελεγκτές που χρησιμοποιούμε σήμερα. Αυτοί καλούνται να καλύψουν πολλές από τις καθημερινές μας απλές τεχνολογικές ανάγκες όπως τηλεχειριστήρια και συστήματα αυτόματου ελέγχου προσφέροντας ταυτόχρονα αξιοπιστία και χαμηλό κόστος.

Οι δυνατότητες των σύγχρονων μικροελεγκτών σε συνδυασμό με το χαμηλό τους κόστος και την ευκολία προγραμματισμού τους προσφέρουν την δυνατότητα σε ανθρώπους με τις σχετικές γνώσεις να τους εκμεταλλευτούν για να δημιουργήσουν τα δικά τους έργα. Ένα τέτοιο έργο παρουσιάζεται στα επόμενα κεφάλαια αυτής της πτυχιακής εργασίας.

2. ARDUINO

2.1 Γενικά για το Arduino

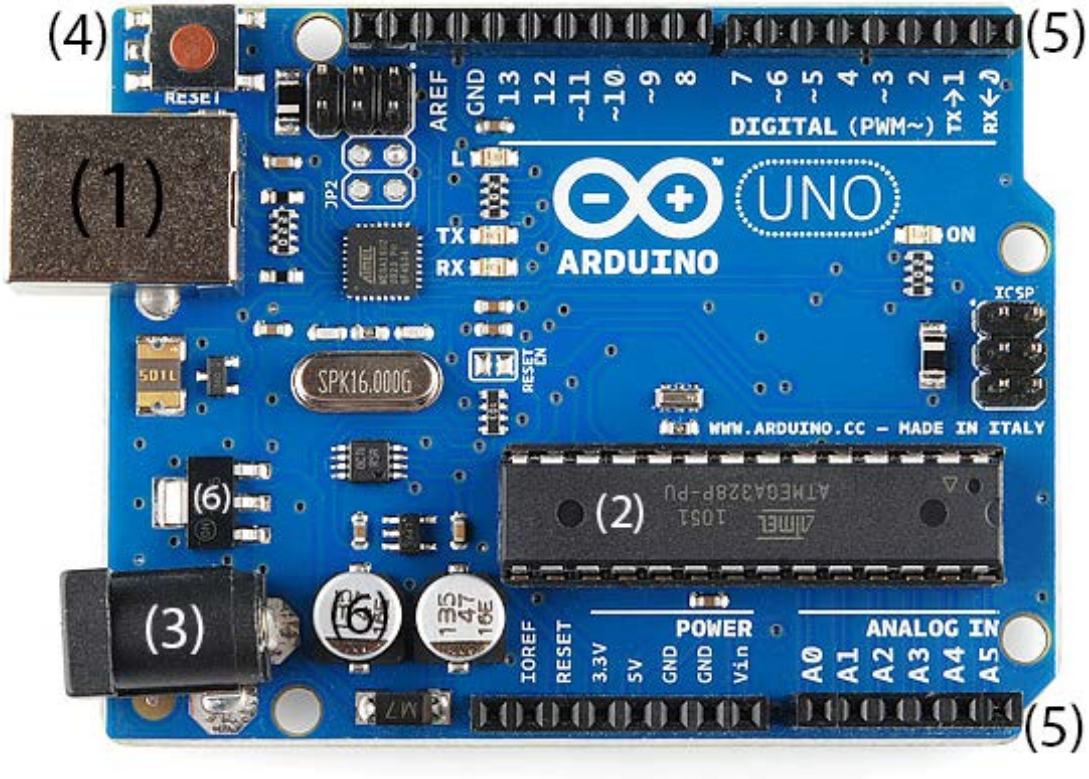
Το Arduino είναι μία πλατφόρμα ανοικτού κώδικα το οποίο βοηθάει στην κατασκευή ηλεκτρονικών έργων όπως αυτό που θα παρουσιάσουμε στα επόμενα κεφάλαια. Η πλατφόρμα αυτή αποτελείται από μία προγραμματιζόμενη ηλεκτρονική πλακέτα και από το λογισμικό το οποίο καθιστά δυνατό τον προγραμματισμό της. Πιο αναλυτικά, η προγραμματιζόμενη ηλεκτρονική πλακέτα έχει ως κύριο συστατικό της έναν μικροελεγκτή *(στοιχείο 2 εικόνας 2.1.1)* ενώ ταυτόχρονα στεγάζει και άλλα υποσυστήματα όπως η θύρα USB *(στοιχείο 1 εικόνας 2.1.1)*, τις εσοχές εισόδου – εξόδου και παροχής ρεύματος *(στοιχείο 5 εικόνας 2.1.1)*, την είσοδο ρεύματος *(στοιχείο 3 εικόνας 2.1.1)*, τον διακόπτη επαναφοράς λειτουργίας *(στοιχείο 4 εικόνας 2.1.1)* αλλά και διάφορα ηλεκτρονικά υποσυστήματα *(στοιχείο 6 εικόνας 2.1.1)* όπως πυκνωτές και τρανζίστορ ενώ το λογισμικό που την συνοδεύει είναι στην ουσία ένα περιβάλλον προγραμματισμού (IDE).

Η συγκεκριμένη πλατφόρμα είναι μία εκ των πολλών διαθέσιμων επιλογών η οποία φαίνεται ιδανική περίπτωση για κάποιον που κάνει τα πρώτα του βήματα στον χώρο των ηλεκτρονικών και των μικροελεγκτών. Ανάμεσα στα πλεονεκτήματα του Arduino απέναντι στις υπόλοιπες υλοποιήσεις είναι:

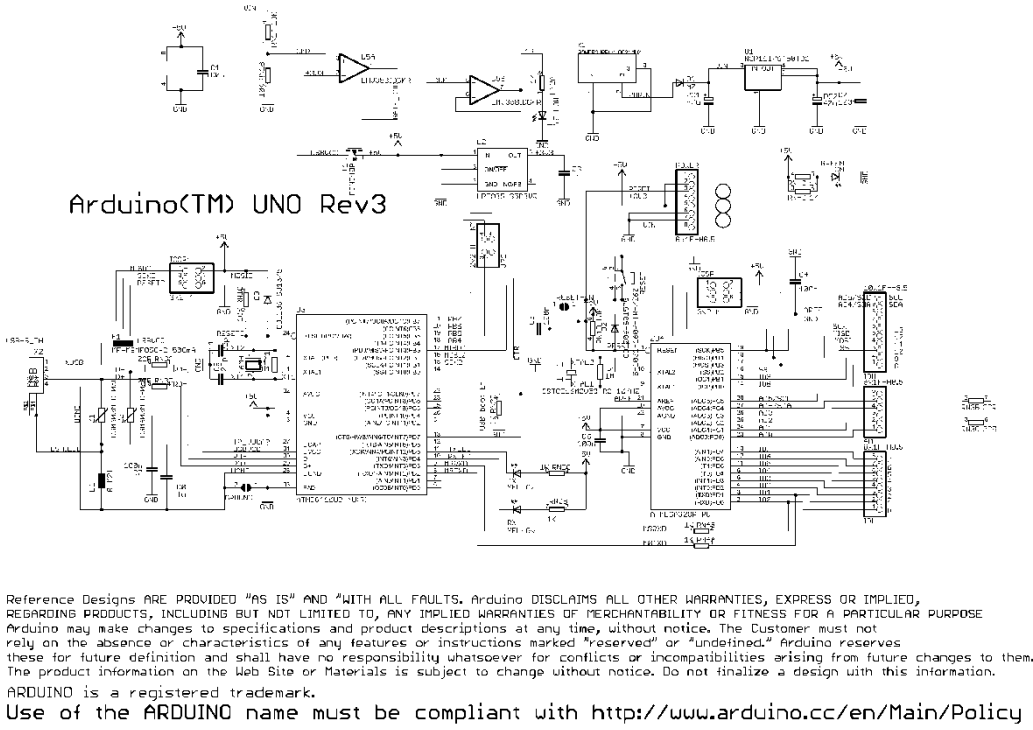
- το αρκετά χαμηλότερο κόστος. Όντας μία πλατφόρμα ανοικτού κώδικα, δίνεται η δυνατότητα στον καθένα να χρησιμοποιήσει το σχεδιάγραμμα του ηλεκτρικού κυκλώματος του Arduino (εικόνα 2.1.2) για να το κατασκευάσει ο ίδιος αφού προμηθευτεί τα απαραίτητα υλικά όπως ο ίδιος ο μικροελεγκτής, μία θύρα USB για τον προγραμματισμό του συστήματος αλλά και την πλακέτα η οποία θα στεγάζει όλα τα απαραίτητα υποσυστήματα. Μία τέτοια προσέγγιση θα κρατήσει το κόστος σε μονοψήφιο νούμερο δίνοντας ταυτόχρονα στον δημιουργό του την δυνατότητα να αντιληφθεί καλύτερα την λειτουργία του. Φυσικά, υπάρχει και η δυνατότητα αγοράς έτοιμου συστήματος Arduino το οποίο προσφέρεται σε μεγάλο εύρος τιμών ανάλογα με τις ανάγκες του χρήστη του.
- η δυνατότητα χρήσης του προγραμματιστικού περιβάλλον σε πολλαπλά λειτουργικά συστήματα. Ενώ οι περισσότερες πλατφόρμες απαιτούν Windows για να ικανοποιήσουν τις προγραμματιστικές τους ανάγκες, το προγραμματιστικό περιβάλλον του Windows είναι συμβατό και με Linux αλλά και με MacOSX ικανοποιώντας με αυτόν τον τρόπο τον κάθε πιθανό χρήστη του.
- το εύκολο στον χειρισμό προγραμματιστικού του περιβάλλον (IDE) το οποίο δεν απαιτεί ιδιαίτερη εμπειρία από τον χρήστη του. Ταυτόχρονα δίνει την δυνατότητα στους προχωρημένους χρήστες να εκμεταλλευτούν την κάθε του δυνατότητα.
- η χρήση θύρας USB αντί για σειριακής για τον προγραμματισμό του. Η ευκολία που προσφέρει η ύπαρξη της θύρας USB είναι

μεγάλη καθώς είναι ένα πρότυπο επικοινωνίας που υπάρχει σε κάθε υπολογιστή της εποχής μας σε αντίθεση με την σειριακή θύρα που τείνει προς εξαφάνιση.

- η στήριξη του από μία πολύ μεγάλη κοινότητα της οποίας τα μέλη προσφέρουν σε αυτή μέσω παροχής έτοιμου κώδικα (βιβλιοθήκη) ή βοήθειας σε άλλους χρήστες.
- το λογισμικό ανοικτού κώδικα το οποίο προσφέρεται για επέκταση από οποιονδήποτε έχεις τις γνώσεις και θέλει αν συνεισφέρει στην κοινότητα.
- το ανοικτού κώδικα υλικό του για το οποίο όπως προαναφέρθηκε είναι διαθέσιμο σχεδιάγραμμα *(εικόνα 2.1.2)* για να το φτιάξει μόνος του και για να το επεκτείνει αν αυτό κριθεί απαραίτητο.



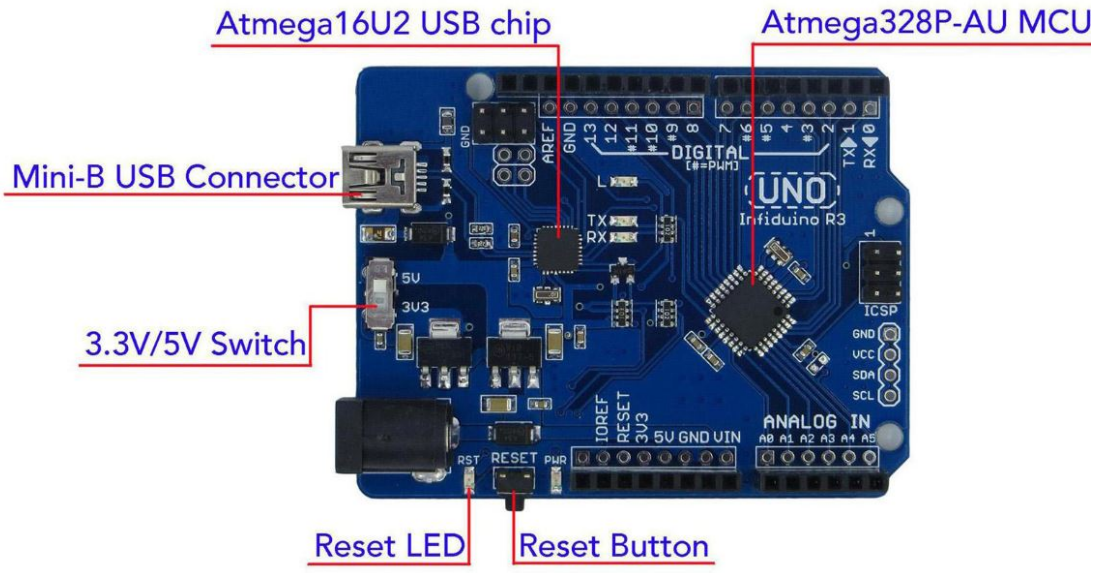
Εικόνα 2.1.1: Ηλεκτρονική πλακέτα Arduino



Εικόνα 2.1.2: Σχεδιάγραμμα κατασκευής του υλικού της πλατφόρμας

Στην αγορά υπάρχουν και μερικές παραλλαγές του Arduino τις οποίες προσφέρουν εταιρίες εκμεταλλευόμενες την ανοικτού κώδικα φύση της πλατφόρμας. Έτσι, προσφέρουν στην αγορά δικές τους υλοποιήσεις του Arduino οι οποίες προσπαθούν να διαφοροποιηθούν σε μερικούς τομείς. Οι παραλλαγές αυτές του υλικού χωρίζονται σε δύο κατηγορίες, τις επίσημες και τις ανεπίσημες. Στην πρώτη κατηγορία ανήκουν οι αυτές για τις οποίες ο κατασκευαστής πληρώνει ένα μικρό ποσό ανά τακτά χρονικά διαστήματα στην εταιρία που κατασκευάζει το αυθεντικό Arduino προσφέροντας έτσι στο έργο της ενώ ταυτόχρονα ο χρήστης του διασφαλίζεται ως προς την ποιότητα του προϊόντος. Στις ανεπίσημες υλοποιήσεις, ανήκουν αυτές για τις οποίες ο κατασκευαστής του αρχικού Arduino δεν έχει καμία συμμετοχή αλλά ούτε και έσοδα από αυτό. Για το παρόν έργο, θα χρησιμοποιήσουμε το **Infiduino Uno R3** *(εικόνα 2.1.3)* το οποίο είναι μία παραλλαγή του Arduino Uno με

κύριο χαρακτηριστικό η υποστήριξη τάσεων 3.3 και 5 Volt με την χρήση του αντίστοιχου διακόπτη κάτι το οποίο αποδεικνύεται ιδιαίτερα χρήσιμο στην πλειοψηφία των έργων που μπορεί να πραγματοποιήσει κάποιος με την βοήθεια ενός Arduino.



Εικόνα 2.1.3: Ηλεκτρονική πλακέτα Infiduino

Το Arduino, όπως και τα περισσότερα τεχνολογικά επιτεύγματα της εποχής μας, δε μπορούν να λειτουργήσουν αυτόνομα καθώς χρειάζονται τον ανθρώπινο παράγοντα για να τα προγραμματίσει και να τους πει τι να κάνουν. Ο τρόπος κατά τον οποίο προγραμματίζεται να λειτουργεί το Arduino είναι ιδιαίτερα απλός: λαμβάνει ως είσοδο κάποια δεδομένα τα οποία πυροδοτούν μία αντίδραση του συστήματος η οποία αντικατοπτρίζεται στην έξοδο αυτού. Για παράδειγμα, μελετώντας το σύστημα κεντρικής θέρμανσης πετρελαίου ενός διαμερίσματος, παρατηρούμε ότι η λειτουργία του είναι εξαιρετικά απλή. Πιο συγκεκριμένα, αφού ρυθμίσει ο άνθρωπος την επιθυμητή θερμοκρασία (είσοδος συστήματος), το σύστημα ελέγχει την θερμοκρασία του περιβάλλοντος (ανατροφοδότηση) και αν αυτή είναι χαμηλότερη από την επιθυμητή αναλαμβάνει να την ανεβάσει (έξοδος

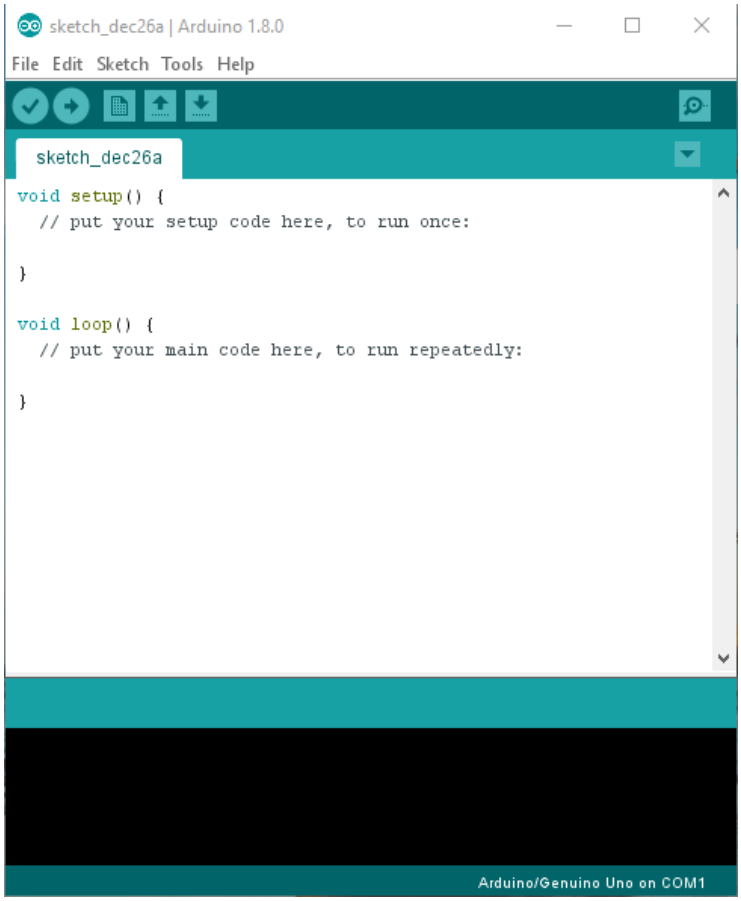
συστήματος) έως ότου φτάσει την επιθυμητή. Σε διαφορετική περίπτωση βρίσκεται σε αδράνεια μέχρι να η θερμοκρασία του περιβάλλοντος να πέσει κάτω από την επιθυμητή.

2.2 Βασικές Έννοιες

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, το Arduino χαρακτηρίζεται ως πλατφόρμα λόγω των πολλών συνιστωσών που το αποτελούν. Στο παρόν κεφάλαιο θα αναφέρουμε και θα αναλύσουμε μερικά από τα σημαντικότερα κομμάτια που απαρτίζουν την πλατφόρμα του Arduino και κατά επέκταση όλων των εναλλακτικών λύσεων που βασίζονται σε αυτό.

2.2.1 Arduino IDE

Ως Arduino Integrated Development Environment *(εικόνα 2.2.1.1)* χαρακτηρίζουμε το περιβάλλον ανάπτυξης κώδικα για το Arduino. Το περιβάλλον μας προσφέρει εργαλεία και δυνατότητες που βελτιστοποιούν την δουλειά του προγραμματιστή. Μερικές από αυτές τις δυνατότητες είναι η δυνατότητα αποστολής του προγράμματος (γνωστό και ως sketch) στο Arduino μέσω μίας απλής σύνδεσης του Arduino με καλώδιο USB και ο ενσωματωμένος debugger ο οποίος καλείται να εντοπίσει πιθανά λάθη στον κώδικα και πολλές φορές να προτείνει τρόπους αντιμετώπισης τους. Επίσης, αναλαμβάνει την εγκατάσταση όλων των απαραίτητων συνοδευτικών προγραμμάτων οδήγησης (drivers) αφήνοντας στον προγραμματιστή την ουσιαστική δουλειά σύνταξης του κώδικα.



Εικόνα 2.2.1.1: Προγραμματιστικό Περιβάλλον Arduino IDE

2.2.2 Βιβλιοθήκες

Οι βιβλιοθήκες ή αλλιώς libraries, όπως σε κάθε γλώσσα προγραμματισμού, δίνουν την δυνατότητα επέκτασης του προγράμματος αφού προσθέτουν σε αυτό κώδικα τον οποίο έχει ήδη γράψει κάποιος άλλος προγραμματιστής και εκτελεί κάποια πολύ συγκεκριμένη εργασία. Συνήθως, ο κώδικας αυτός εκτελεί βασικές εργασίες οι οποίες συναντιόνται σε μεγάλο αριθμό έργων.

Οι βιβλιοθήκες ικανοποιούν την αρχή του αντικειμενοστραφή προγραμματισμού κατά την οποία υπάρχει επαναχρησιμοποίηση κώδικα αφού μπορούν να χρησιμοποιηθούν από οποιοδήποτε προγραμματιστή

για να εκτελέσουν ενέργειες οι οποίες επαναλαμβάνονται σε σεβαστό αριθμό έργων. Μερικές από αυτές καλούνται να ικανοποιήσουν ανάγκες μαθηματικών υπολογισμών, να επεξεργαστούν ένα αρχείο ή να αναλάβουν την επικοινωνία με ένα άλλο μηχάνημα. Στην περίπτωση του Arduino, μία βιβλιοθήκη μπορεί να κληθεί για να εξασφαλίσει πρόσβαση του μηχανήματος σε δίκτυο μέσω θύρας Ethernet, για να συνδέσει το Arduino σε δίκτυο GSM, για να απεικονίσει στοιχεία σε μία οθόνη LCD, για να επικοινωνήσει με περιφερειακά, για να δώσει εντολές σε κινητήρες κίνησης και πολλά ακόμα.

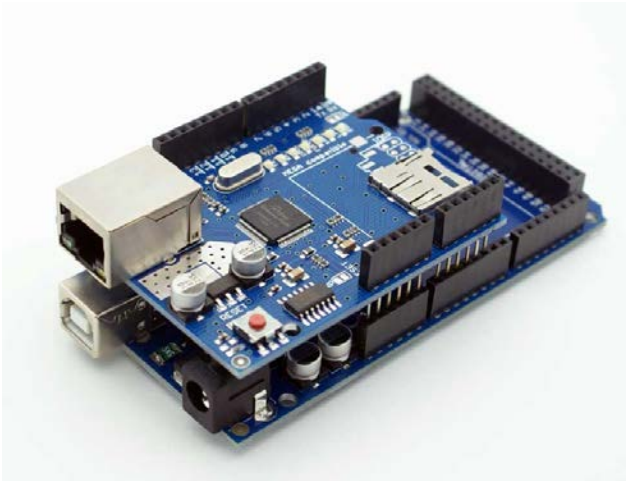
Η ύπαρξη πληθώρας βιβλιοθηκών και η ελεύθερη χρήση τους από τους προγραμματιστές είναι ένας από τους λόγους που το Arduino χρησιμοποιείται από όλο και περισσότερους ανθρώπους. Αυτό έχει ως αποτέλεσμα η κοινότητα πίσω από αυτό ολοένα να μεγαλώνει με πολλά από τα μέλη της να προσφέρουν ενεργά σε αυτή βιβλιοθήκες.

2.2.3 Πλακέτες Επέκτασης (Shields)

Στοχεύοντας στον χρήστη με έλλειψη εμπειρίας σε κατασκευή ηλεκτρονικών κυκλωμάτων, προσφέρονται για το Arduino ολοκληρωμένες πλακέτες επέκτασης οι οποίες επεκτείνουν την λειτουργία του προσθέτοντας του νέες δυνατότητες. Αυτές καλούνται να προσθέσουν δυνατότητες που ζητούνται κατά κόρων από το αγοραστικό κοινό όπως κάρτα ασύρματης (Wifi) ή ενσύρματης (Ethernet) δικτύωσης *(εικόνα 2.2.3.1)*, οθόνη και GPS. Οι πλακέτες αυτές συνοδεύονται από τις δικές τους βιβλιοθήκες οι οποίες όπως αναφέρθηκε προηγουμένως

καλούνται να προσθέσουν στον κώδικα τον δικό τους απαραίτητο κώδικα για ομαλή λειτουργία.

Οι περισσότερες πλακέτες επέκτασης είναι φτιαγμένες με τέτοιον τρόπο έτσι ώστε να επεκτείνουν τις υποδοχές του Arduino δίνοντας έτσι την δυνατότητα στον χρήστη να στοιβάξει πάνω σε ένα Arduino όσες πλακέτες χρειαστεί με μόνη προϋπόθεση αυτές να μην χρησιμοποιούν κοινούς ακροδέκτες.



Εικόνα 2.2.3.1: Πλακέτα επέκτασης ethernet

2.2.4 Αισθητήρες

Οι αισθητήρες είναι βασικά εξαρτήματα του Arduino τα οποία συνδέονται σε αυτό μέσω των ακροδεκτών σύνδεσης της πλακέτας. Ο ρόλος τους σε ένα έργο είναι πολύ κρίσιμος καθώς παίζουν τον ρόλο των ματιών και των αυτιών του μηχανήματος. Σκοπός τους είναι να παρέχουν στο μηχάνημα περιβαλλοντικά δεδομένα όπως η θερμοκρασία, η υγρασία και η απόσταση από κάποιο αντικείμενο.

Κατά την λειτουργία τους, οι αισθητήρες ειδοποιούν το Arduino για τις όποιες περιβαλλοντικές αλλαγές στέλνοντας τάση στους

ακροδέκτες αυτού. Ως αποτέλεσμα, οι πληροφορίες που λαμβάνει το Arduino είναι ανάλογες της τάσης που θα λάβει. Για παράδειγμα, ένας παθητικού τύπου αισθητήρας υπέρυθρων *(εικόνα 2.2.4.1)* εντοπίζει την διαφορά θερμοκρασίας ενός σώματος που θα εισέλθει στο οπτικό του πεδίο σε σχέση με το περιβάλλον που το περιβάλλει και στέλνει σήμα στο Arduino.

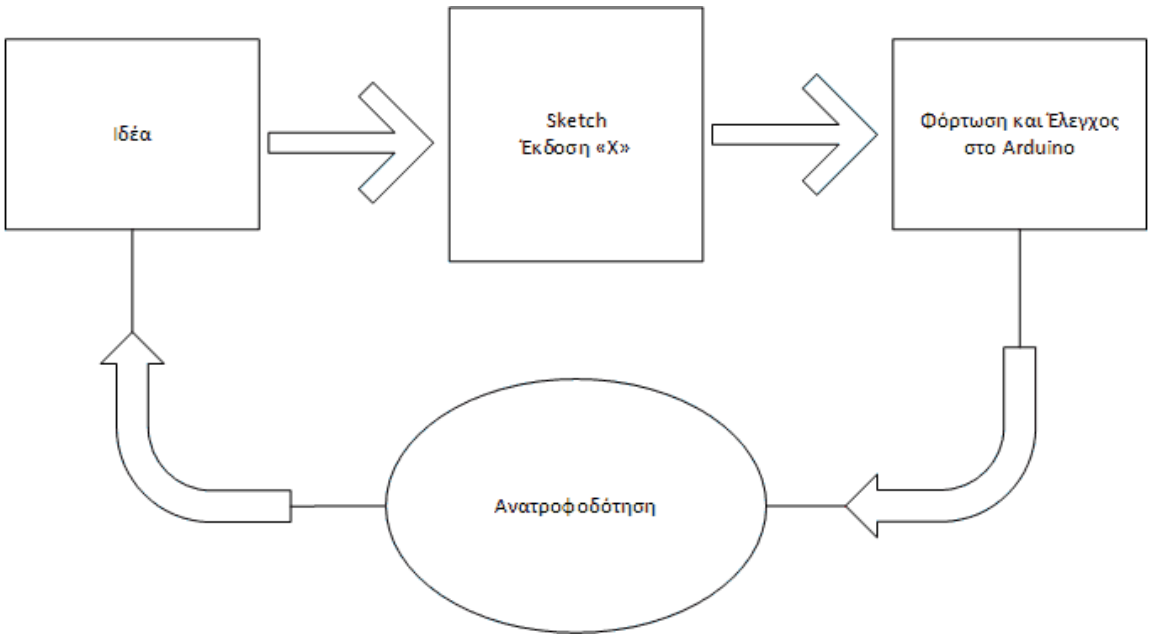
Για την σωστή λειτουργία τους, οι αισθητήρες απαιτούν χρήση των βιβλιοθηκών που τους συνοδεύει στο πρόγραμμα που έχει φορτωθεί στο Arduino. Μόνο με αυτές μπορεί το μηχάνημα να αντιληφθεί και να μεταφράσει τις πληροφορίες που λαμβάνει από τους αισθητήρες.

Εικόνα 2.2.4.1: Αισθητήρας παθητικού τύπου υπέρυθρων

2.2.5 Πρωτοτυποποίηση

Ένα κομμάτι της φιλοσοφίας του Arduino είναι η διαδικασία πρωτοτυποποίησης *(εικόνα 2.2.5.1)*. Ως Sketch χαρακτηρίζεται το σύνολο του κώδικα που απαρτίζει το πρόγραμμα το οποίο γράφουμε στο προγραμματιστικό περιβάλλον Arduino IDE με σκοπό να το «ανεβάσουμε» στην συνέχεια στο Arduino μας. Κατά την διαδικασία

πρωτοτυποποίησης, γράφουμε ένα κομμάτι κώδικα (Sketch) το οποίο στην συνέχεια θα δοκιμάσουμε στο Arduino. Αφού λάβουμε την κατάλληλη ανατροφοδότηση από το σύστημα μας θα κάνουμε τις απαραίτητες τροποποιήσεις σε αυτό και θα δοκιμάσουμε να τρέξουμε την νέα βελτιωμένη έκδοση του κώδικα μας. Αυτή η διαδικασία της προτυποποίησης επαναλαμβάνεται όσες φορές κρίνεται αυτό απαραίτητο μέχρι το έργο μας να φτάσει το επίπεδο τελειότητας που θέλουμε.



Εικόνα 2.2.5.1: Διαδικασία Προτυποποίησης

2.3 Λίστα Εξαρτημάτων

Στην παρακάτω λίστα *(πίνακας 2.3.1)*, έχουν καταγραφεί όλα τα προϊόντα που χρησιμοποιήθηκαν για το έργο που πραγματεύεται η παρούσα πτυχιακή εργασία. Μαζί με τα ονόματα των προϊόντων, γίνεται αναφορά και στις ποσότητες στις οποίες αγοράστηκε το κάθε προϊόν αλλά και στο κόστος που είχε την δεδομένη στιγμή της αγοράς.

Προϊόν	Ποσότητα	Κόστος / Μονάδα σε ευρώ
Πλακέτα Infiduino Uno R3	1	10,19
Breadboard	1	4,00
Βάση οχήματος	1	16,21
Αισθητήρας Υπερήχων	1	2,80
Κινητήρας συνεχούς ρεύματος	4	2,50
Κάρτα επέκτασης Bluetooth - HC06	1	8,90
Ρόδα	4	1,50
Συσκευή ελέγχου κινητήρα με το υποσύστημα L298	1	5,90
Καλώδια	20	0,09
Οθόνη LCD Nokia 5510	1	2,15
Θήκη μπαταριών	1	0,99

Πίνακας 2.3.1: Λίστα εξαρτημάτων που απαρτίζουν το έργο

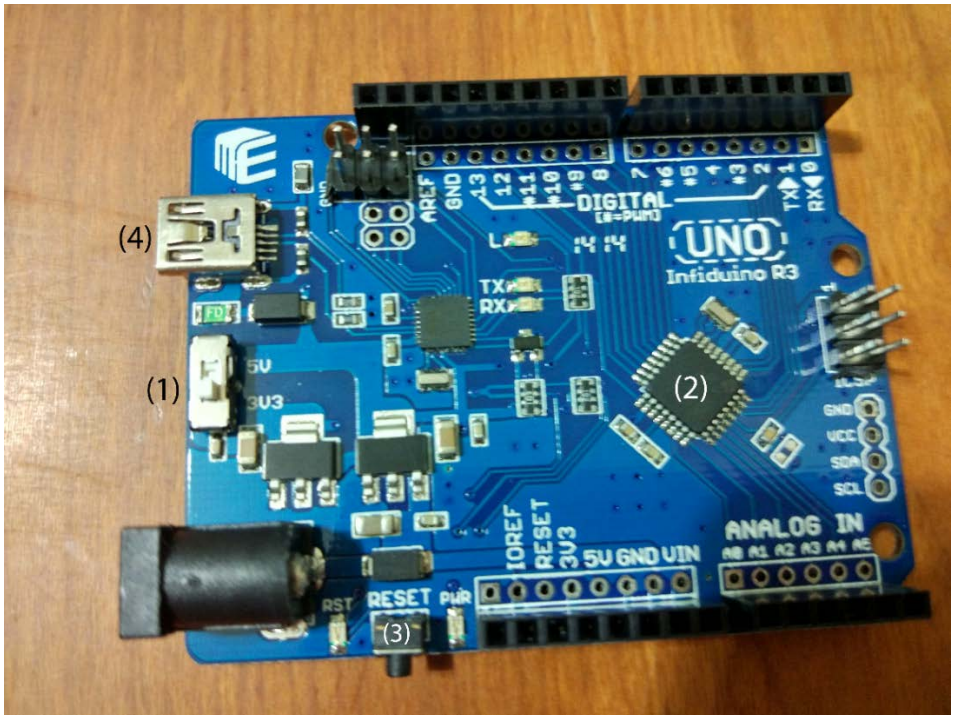
Έχοντας ως σημείο αναφορά τον πίνακα 2.3.1, ακολουθεί αναλυτική περιγραφή των βασικών εξαρτημάτων.

2.3.1 Infiduino Uno R3

Το Infiduino Uno R3 *(εικόνα 2.3.1.1)* αποτελεί μία ανεπίσημη παραλλαγή του Arduino Uno R3. Όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, όντας ανεπίσημη παραλλαγή του πρωτοτύπου, δεν συμβάλει με κανέναν οικονομικό ή άλλο τρόπο στην ανάπτυξη του οικοσυστήματος του Arduino παρά μόνο εκμεταλλεύεται την μέχρι τώρα ανάπτυξη του η οποία συνοδεύεται από την κοινότητα, την υποστήριξη και τα εξαρτήματα που είναι αναπόσπαστο κομμάτι του.

Σε σχέση με το πρωτότυπο Arduino Uno R3, προσφέρει κάποιες επιπλέον δυνατότητες όπως η δυνατότητα επιλογής τάσης λειτουργίας (3.3 ή 5 Volt) μέσω ενός απλού διακόπτη *(σημείο 1 εικόνας 2.3.1.1)*. Αυτό έχει ως αποτέλεσμα την απροβλημάτιστη λειτουργία μερικών περιφερειακών όπως η κάρτα ασύρματης σύνδεσης Bluetooth η οποία εμπεριέχεται στην λίστα αντικειμένων που θα χρησιμοποιηθεί για το παρόν έργο και δουλεύει μέχρι και τα 3.3 Volt σύμφωνα με τις πληροφορίες που διαθέτει ο κατασκευαστής. Στην περίπτωση που δοθεί τάση μεγαλύτερη από αυτή, η συσκευή υπολειτουργεί ενώ υπάρχει και η πιθανότητα να καεί. Για να λειτουργήσουν τέτοιου είδους εξαρτήματα με το πρωτότυπο Arduino, χρειάζονται κάποια επιπλέον ηλεκτρονικά εξαρτήματα και κατά επέκταση και κυκλώματα τα οποία αναλαμβάνουν να ελαττώσουν την τάση του ρεύματος στα επιθυμητά επίπεδα. Μεγάλης σημασίας διαφοροποίηση είναι και ο μικροελεγκτής που χρησιμοποιεί ο οποίος είναι ο Atmega328P-AU *(σημείο 2 εικόνας 2.3.1.1)*. Ο συγκεκριμένο μικροελεγκτής είναι πολύ μικρότερου μεγέθους από τον Atmega328P-PU που βρίσκεται στο Arduino Uno R3 αφήνοντας

περισσότερο χώρο στον χρήστη και στις πλακέτες επέκτασης που εκείνος χρησιμοποιεί. Επίσης, ο διακόπτη επαναφοράς *(σημείο 3 εικόνας 2.3.1.1)* έχει μετακινηθεί καθιστώντας τον πιο εύκολα προσβάσιμο στον τελικό χρήστη. Τέλος, η χρήση του προτύπου USB Mini-B προσφέρει μικρότερη θύρα USB *(σημείο 4 εικόνας 2.3.1.1)* στην πλακέτα ενώ ταυτόχρονα διευκολύνει τον χρήστη καθώς είναι πολύ πιο διαδεδομένο πρότυπο σε σχέση με το USB Type A που χρησιμοποιεί το αυθεντικό Arduino Uno R3. Όλα τα παραπάνω προσθέτουν αξία στην συσκευή με αποτέλεσμα το κόστος απόκτησης του να είναι διπλάσιο από αυτό του πιο απλού αυθεντικού Arduino Uno R3.



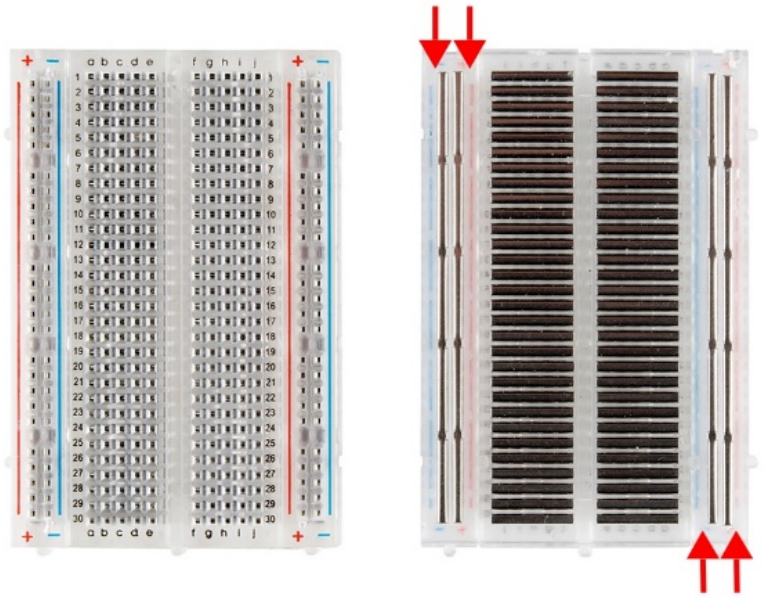
Εικόνα 2.3.1.1: Infiduino Uno R3

2.3.2 Breadboard

Το Breadboard αποτελεί βασικό βοήθημα κάθε πειραματικής κατασκευής καθώς δίνει την δυνατότητα στον χρήστη του να κάνει τις απαραίτητες φυσικές συνδέσεις ανάμεσα στα υποσυστήματα του έργου του χωρίς την ανάγκη κολλήσεων. Λόγω της ιδιότητας του αυτής, το Breadboard είναι κατάλληλο για την δημιουργία προσωρινών συνδέσεων κατά την διαδικασία πρωτοτυποποίησης ενός έργου.

Όπως φαίνεται και στην εικόνα 2.3.2.1, η σύνδεση των υποσυστημάτων στο Breadboard γίνεται στον οριζόντιο αλλά και στον κάθετο άξονα. Η πλακέτα χωρίζεται στην μέση ενώ στις δύο άκρες υπάρχουν απομονωμένα κανάλια που επεκτείνονται στον κάθετο άξονα. Καθ' όλη την έκταση της πλακέτας, υπάρχουν μεταλλικές αγώγιμες πλάκες οι οποίες επιτυγχάνουν την σύνδεση των υποσυστημάτων χωρίς την ανάγκη κολλήσεων. Ο κάθετος διαχωρισμός στην μέση της πλακέτας εξυπηρετεί στην τοποθέτηση μικροελεγκτών που ανήκουν στην οικογένεια Dual Inline Package (DIP). Τοποθετώντας τους μικροελεγκτές αυτούς κάθετα στην μέση της πλακέτας με τους μισούς ακροδέκτες να συνδέονται στην μία πλευρά ενώ τους άλλους μισούς στην άλλη,

επιτυγχάνουμε απομόνωση των ακροδεκτών κάτι το οποίο είναι απαραίτητο για την σωστή λειτουργία τους.



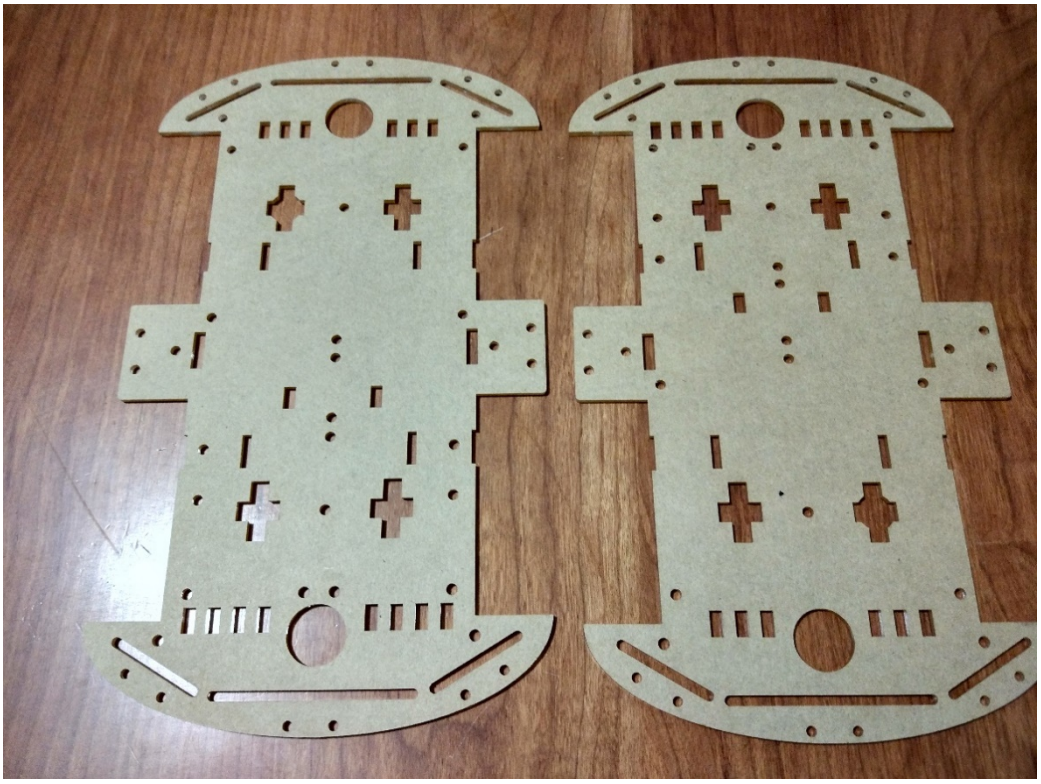
Εικόνα 2.3.2.1: Breadboard

2.3.3 Βάση οχήματος

Για τον «σκελετό» του συστήματος μας θα χρησιμοποιήσουμε τα κομμάτια που απεικονίζονται στην εικόνα 2.3.3.1. Τα κομμάτια αυτά είναι κατασκευασμένα από διάφανο πλαστικό ενώ αποτελούν την ιδανική λύση για το έργο στο οποίο αποσκοπεί η παρούσα πτυχιακή εργασία.

Η βάση αυτή αποτελείται από δύο πανομοιότυπα κομμάτια τα οποία αντιπροσωπεύουν τα επίπεδα του οχήματος. Στο κάτω επίπεδο θα ενσωματωθούν οι κινητήρες μαζί με τις ρόδες ενώ ενδιάμεσα τους θα

μπει η συσκευή ελέγχου των κινητήρων. Στο πάνω επίπεδο θα ενσωματωθεί το Infiduino Uno R3 και η θήκη των μπαταριών.



Εικόνα 2.3.3.1: Σκελετός Συστήματος

2.3.4 Αισθητήρας Απόστασης

Ο αισθητήρας απόστασης *(εικόνα 2.3.4.1)* καλείται να εντοπίσει τα εμπόδια που συναντάει στον δρόμο του το όχημα και να το ειδοποιεί για αυτά. Έτσι, ελέγχει συνεχώς τον χώρο μπροστά του και στην περίπτωση που εντοπίσει κάποιο αντικείμενο υπολογίζει την απόσταση του και ενημερώνει το σύστημα το οποίο στην συνέχεια καλείται να πάρει μία απόφαση για την συνέχεια της πορείας του.

Ο αισθητήρας απόστασης που επιλέχθηκε για το συγκεκριμένο έργο χρησιμοποιεί υπέρηχους για να λειτουργήσει. Ο λόγος που επιλέχθηκε αισθητήρας τύπου υπερήχων έχει να κάνει με τον τρόπο με τον οποίον

δουλεύουν αυτού του είδους οι αισθητήρες ο οποίος τους καθιστά ιδανική επιλογή. Πιο συγκεκριμένα, το σύστημα εκπέμπει ένα ήχο συχνότητας πολύ υψηλότερης από ότι μπορεί να ακούσει ο άνθρωπος και περιμένει για την ηχώ του. Την στιγμή που ο ήχος αυτός φτάσει ένα αντικείμενο, παράγεται η ηχώ του η οποία ταξιδεύει στην αντίθετη κατεύθυνση επιστρέφοντας στον αισθητήρα. Μόλις ο αισθητήρας λάβει την ηχώ, αναλαμβάνει να υπολογίσει την απόσταση του από το αντικείμενο έχοντας ως στοιχείο το χρόνο που πέρασε από την στιγμή που έγινε η εκπομπή του σήματος μέχρι την στιγμή που η ηχώ του έφτασε πίσω.

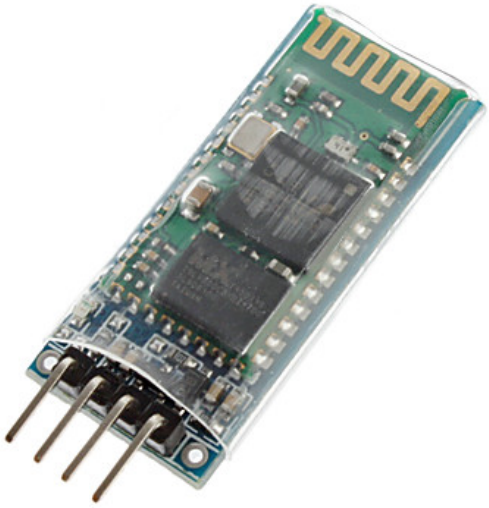
Ενδιαφέρον έχει ο τρόπος με τον οποίο υπολογίζει την απόσταση. Έστω ότι η συσκευή βρίσκεται σε ιδανικές συνθήκες όπου η θερμοκρασία του χώρου ανέρχεται στους 20 βαθμούς Κελσίου. Στην θερμοκρασία αυτή, η ταχύτητα του ήχου είναι 343,5 m/s σύμφωνα με τον τύπο $s = 331{,}5 + 0{,}6 * temp$ με μονάδα μέτρησης το μέτρα ανά δευτερόλεπτο. Μετατρέποντας τις μονάδες του παραπάνω νούμερου σε εκατοστά του μέτρου ανά χιλιοστό του δευτερολέπτου έχουμε 0,03435 cm/ms. Με άλλα λόγια, ο ήχος μπορεί να διανύσει 0,03435 εκατοστά του μέτρου κάθε χιλιοστό του δευτερολέπτου όταν η θερμοκρασία του περιβάλλοντος βρίσκεται στους 20 βαθμούς κελσίου. Σε κάθε περίπτωση, αν ο ήχος και η ηχώ χρειάζονται x ms για να κάνουν ολόκληρη την διαδρομή, η συνολική απόσταση που θα έχει διανύσει σε cm είναι $\mathrm{x} * 0{,}03435$. Επειδή στο παρόν παράδειγμα μελετάμε την απόσταση που θα διανύσει η ηχώ για να φτάσει από το αντικείμενο πίσω στον αισθητήρα χωρίς να συνυπολογίζουμε την απόσταση που διανύει ο ήχος για να φτάσει από τον αισθητήρα στο αντικείμενο, διαιρούμε το αποτέλεσμα με το δύο. Έτσι, ο τύπος διαμορφώνεται ως εξής: $\mathrm{d} = \mathrm{x}/2 * 0{,}03435 \frac{cm}{ms}$

Εικόνα 2.3.4.1: Αισθητήρας υπερήχων HC-SR04

2.3.5 Προσαρμογές Bluetooth

Ο προσαρμογές Bluetooth HC-06 *(εικόνα 2.3.5.1)* καλείται να καλύψει μία βασική ανάγκη του συστήματος μας η οποία είναι η σύνδεση του οχήματος με την συσκευή που είναι υπεύθυνη για να στέλνει εντολές στο όχημα μας. Ο προσαρμογέας αυτός προσομοιώνει τις λειτουργίες της σειριακής θύρας μέσω της οποίας αποστέλλονται και λαμβάνονται δεδομένα και μεταφέρει τα δεδομένα αυτά ασύρματα.

Ο προσαρμογέας αυτός χρησιμοποιεί την έκδοση 2.0 του Bluetooth σε συνδυασμό με την τεχνολογία Enhanced Data Rate η οποία προσφέρει ταχύτητες μεταφοράς δεδομένων μέχρι και 2.1 Mbps. Η συσκευή υποστηρίζει ταχύτητα μετάδοσης (baud rate) της τάξης των 9600 bps ενώ η εμβέλεια του σήματος του φτάνει μέχρι και τα 10 μέτρα.



Εικόνα 2.3.5.1: Bluetooth HC-06 module

2.3.6 Κινητήρες

Ως κινητήρα χαρακτηρίζεται μία ηλεκτρική μηχανή η οποία μετατρέπει την ηλεκτρική ενέργεια σε μηχανική. Οι κινητήρες διακρίνονται σε δύο κατηγορίες: i) κινητήρες συνεχούς ρεύματος και ii) κινητήρες εναλλασσόμενου ρεύματος. Οι κινητήρες συνεχούς ρεύματος *(εικόνα 2.3.6.1)* παρουσιάζονται ως η βέλτιστη επιλογή για το παρόν έργο καθώς είναι αρκετά φθηνότεροι και μπορούν να εφαρμοστούν σε πληθώρα έργων κλίμακας όπως ένα ελικόπτερο, μία βαρκούλα ή ένα ρομπότ.

Οι κινητήρες αυτού του είδους υπακούν στο φαινόμενο της ηλεκτρομαγνητικής επαγωγής το οποίο και αξιοποιούν για να λειτουργήσουν. Πιο συγκεκριμένα, έστω ένας αγωγός ο οποίος βρίσκεται μέσα σε ένα μαγνητικό πεδίο και βρίσκεται σε αδράνεια. Την στιγμή που το ηλεκτρικό ρεύμα θα περάσει μέσα από τον αγωγό, το μαγνητικό πεδίο που το περιβάλλει θα του ασκήσει δύναμη η οποία θα το ωθήσει ως προς

μία κατεύθυνση. Η κατεύθυνση αυτή εξαρτάται από την πολικότητα του ρεύματος που διαπερνάει τον αγωγό ενώ η ταχύτητα με την οποία θα κινηθεί ο κινητήρας είναι ανάλογη της δύναμης που ασκεί το μαγνητικό πεδίο στον αγωγό η οποία με την σειρά της εξαρτάται από την ένταση του ηλεκτρικού ρεύματος, την ένταση του μαγνητικού πεδίου και από το μήκος του αγωγού.

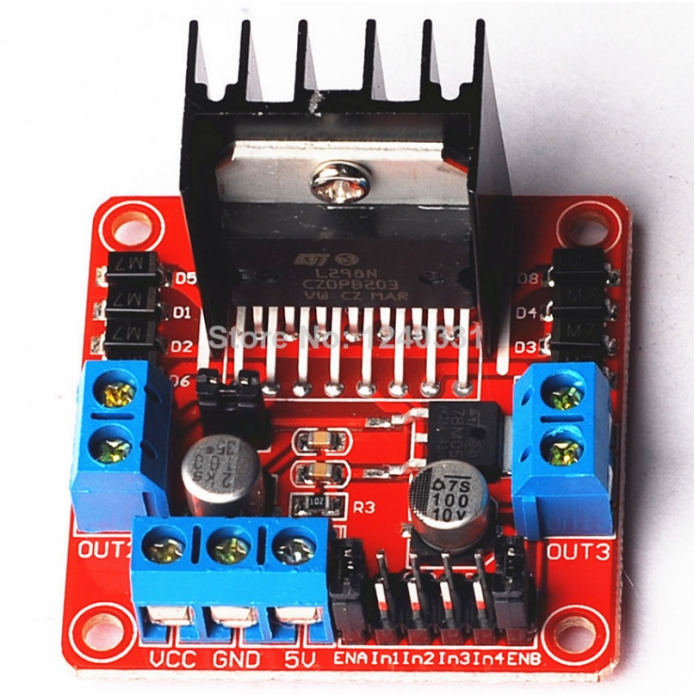
Εικόνα 2.3.6.1: Κινητήρες

2.3.7 Συσκευή Ελέγχου Κινητήρα με το υποσύστημα L298

Για την χρήση των κινητήρων απαραίτητο κρίνεται ένα κύκλωμα ελέγχου με βάση τον οδηγό L298N *(εικόνα 2.3.7.1)*. Το κύκλωμα αυτό αναλαμβάνει την σωστή τροφοδοσία του συστήματος παρέχοντας στους κινητήρες την απαραίτητη ενέργεια από εξωτερική πηγή για την ομαλή λειτουργία του κινητήρα. Στην περίπτωση που η τακτική αυτή δεν εφαρμοστεί στο έργο και αντίθετα την τροφοδοσία των κινητήρων την αναλάβει απευθείας το Infiduino, οι κινητήρες δεν θα μπορούν να

δουλέψουν καθώς το Infiduino δεν έχει την δυνατότητα να παράσχει το ρεύμα που χρειάζονται για να λειτουργήσουν ομαλά. Ως αποτέλεσμα, οι κινητήρες θα υπολειτουργούν ενώ υπάρχει περίπτωση οι κινητήρες να στείλουν ρεύμα πίσω στο υπόλοιπο σύστημα κάτι το οποίο θα του προκαλέσει πολύ σοβαρή ζημιά.

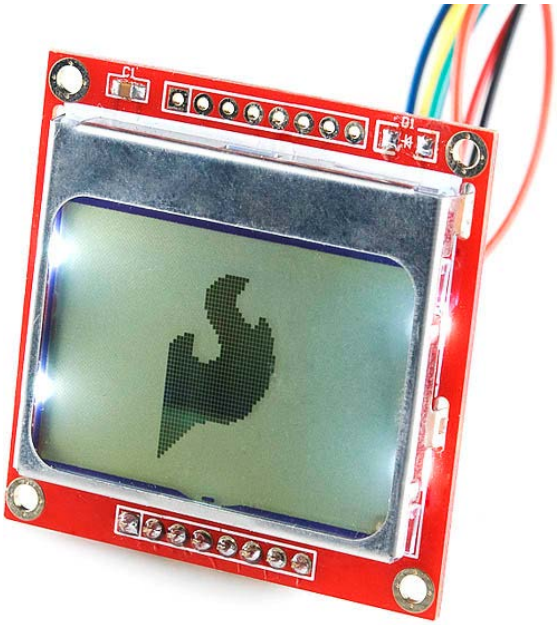
Για την σωστή εξυπηρέτηση των κινητήρων από τον οδηγό L298N, απαιτείται σύνδεση του οδηγού με το Infiduino μέσω της οποίας το τελευταίο θα στέλνει εντολές στο κύκλωμα ελέγχου. Πιο συγκεκριμένα, οι εντολές αυτές αναφέρονται στην κατεύθυνση προς την οποία γυρίζουν οι κινητήρες αλλά και στην ταχύτητα με την οποία λειτουργούν. Η διαδικασία σύνδεσης των κινητήρων με το κύκλωμα ελέγχου και το Infiduino παρουσιάζεται με λεπτομέρεια σε επόμενο κεφάλαιο. Η ένταση του ρεύματος που φτάνει στον κινητήρα και κατά επέκταση ταχύτητα του είναι ανάλογη με την διαφορά τάσης που συναντάται ανάμεσα στους δύο ακροδέκτες του οδηγού L298N στον οποίο έχει συνδεθεί ενώ αντιστρέφοντας την σύνδεση των καλωδίων στους ακροδέκτες αλλάζει η πολικότητα του κινητήρα με αποτέλεσμα την μεταβολή της κατεύθυνσης του.



Εικόνα 2.3.7.1: Κέντρο ελέγχου κινητήρων

2.3.8 Οθόνη LCD Nokia 5510

Στο έργο αυτό θα γίνει χρήση της οθόνης υγρών κρυστάλλων *(εικόνα 2.3.8.1)* που κατασκευάζεται για το κινητό τηλέφωνο Nokia 5510. Στην οθόνη θα εμφανίζονται χρήσιμες πληροφορίες όπως η απόσταση σε εκατοστά (cm) από το κοντινότερο εμπόδιο αλλά και προειδοποιητικό μήνυμα όταν το αυτοκινούμενο όχημα πλησιάζει σε αντικείμενο.



Εικόνα 2.3.8.1: Οθόνη υγρών κρυστάλλων τύπου Nokia 5510

Για την επίτευξη των λειτουργιών της συσκευής, γίνεται χρήση του ολοκληρωμένου κυκλώματος PCD8544 της Philips. Στο μπροστινό μέρος της συσκευής βρίσκεται η οθόνη ενώ και από τις δύο πλευρές θα έχουμε πρόσβαση στους ακροδέκτες της συσκευής μέσω των οποίων συνδέουμε την οθόνη μας με το Arduino. Στον πίνακα που ακολουθεί *(πίνακας 2.3.8.2)* αναφέρονται οι ακροδέκτες και η λειτουργία τους.

Αριθμός Ακροδέκτη (pin)	Ονομασία	Λειτουργία
1	VCC	Παροχή Ρεύματος
2	GND	Γείωση
3	SCE	Επιλογή μικροελεγκτή
4	RST	Επαναφορά
5	D/C	Επιλογή Λειτουργίας
6	DN(MOSI)	Είσοδος Σειριακής Θύρας
6	SCLK	Ρολόι Σειριακής Θύρας
7	LED	Παροχή Ρεύματος Οπίσθιου Φωτισμού

Πίνακας 2.3.8.2: Λίστα ακροδεκτών κυκλώματος οθόνης

2.3.9 Θήκη Μπαταριών

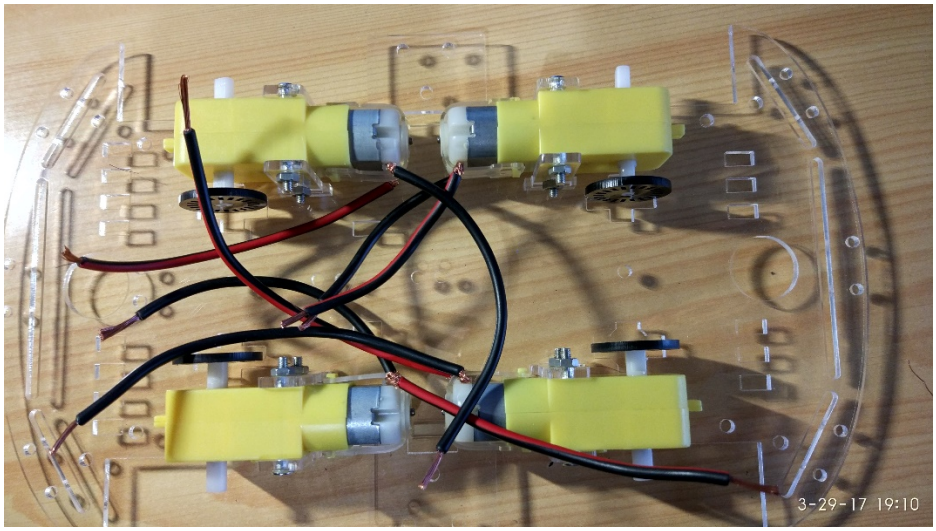
Η θήκη μπαταριών *(εικόνα 2.3.9.1)* είναι υπεύθυνη για να στεγάσει τις έξι μπαταρίες τύπου AA οι οποίες θα τροφοδοτούν επαρκώς τους κινητήρες και την συσκευή ελέγχου αυτών. Η σύνδεση των μπαταριών μεταξύ τους είναι σε σειρά με αποτέλεσμα η συνολική τάση παροχής ρεύματος στο σύστημα των κινητήρων να ανέρχεται στα 7,2 Volt. Η τιμή αυτή υπολογίζεται από τον τύπο: Σ=1,2 Volt*6

Εικόνα 2.3.9.1: Θήκη μπαταριών

3. ΚΑΤΑΣΚΕΥΗ ΟΧΗΜΑΤΟΣ

3.1 Ανάλυση Βημάτων Συναρμολόγησης

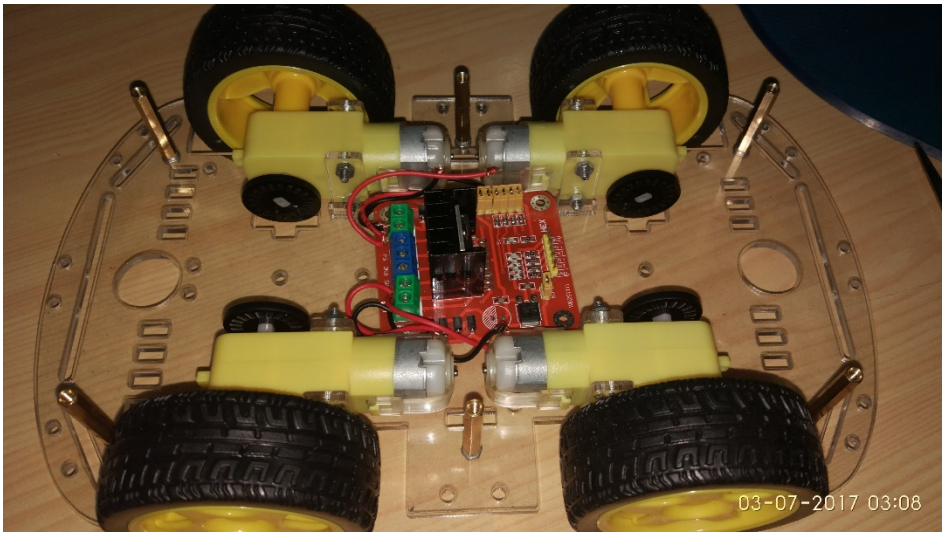
Έχοντας συγκεντρώσει τα υλικά που αναφέρθηκαν στα προηγούμενα κεφάλαια, είμαστε σε θέση να ξεκινήσουμε την συναρμολόγηση του οχήματος. Για αρχή, εγκαθιστούμε πάνω στην βάση του οχήματος *(εικόνα 2.3.3.1)*, τους τέσσερις κινητήρες *(εικόνα 2.3.6.1)* ενώ στην συνέχεια τοποθετούμε καλώδια σε αναμονή πάνω στους κινητήρες *(εικόνα 3.1.1)*.



Εικόνα 3.1.1: Εγκατάσταση κινητήρων στην βάση του οχήματος

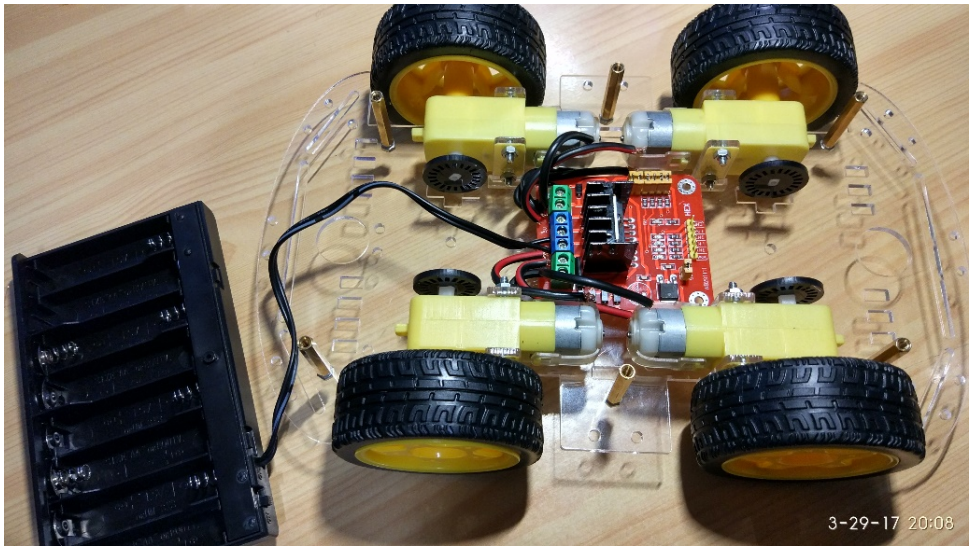
Στην συνέχεια εγκαθιστούμε τις ρόδες πάνω στους κινητήρες ενώ ταυτόχρονα τοποθετούμε στην βάση του οχήματος τη συσκευή ελέγχου των κινητήρων *(εικόνα 2.3.7.1)*. Για την σωστή σύνδεση των κινητήρων με την συσκευή ελέγχου, ενώνουμε τους κινητήρες σε ομάδες ανά δύο μεταξύ τους. Κάθε μία από τις δύο μεγαλύτερες πλευρές του οχήματος αντιπροσωπεύει και μία ομάδα κινητήρα. Έπειτα, συνδέουμε τα τέσσερα καλώδια (δύο κόκκινα και δύο μαύρα) στις εσοχές «Motor A» και «Motor B» στην συσκευή

ελέγχου όπως φαίνεται στην εικόνα 3.1.2. Η συνδεσμολογία αυτή βοηθάει στον έλεγχο της κατεύθυνσης του οχήματος και πιο συγκεκριμένα δίνει την δυνατότητα σε αυτό να αλλάζει κατεύθυνση. Αυτό επιτυγχάνεται απενεργοποιώντας το «Motor B» (δεξιός πλευρικός άξονας) στην περίπτωση που θέλουμε το όχημα να στρίψει δεξιά ενώ αντίστοιχα απενεργοποιώντας το «Motor A» (αριστερός πλευρικός άξονας) στην περίπτωση που θέλουμε το όχημα να στρίψει αριστερά.



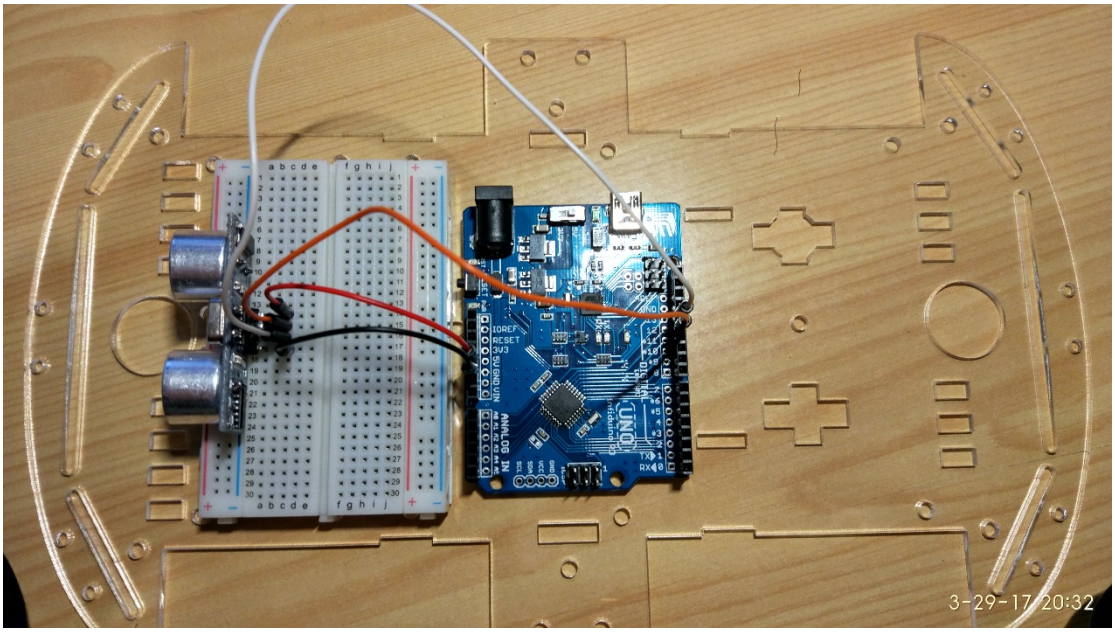
Εικόνα 3.1.2: Εγκατάσταση κινητήρων στην βάση του οχήματος

Έπειτα συνδέουμε στην συσκευή ελέγχου των κινητήρων την θήκη μπαταριών *(εικόνα 3.1.3)* συνδέοντας το μαύρο καλώδιο στην είσοδο "GND" και το κόκκινο στην είσοδο "VMS".



Εικόνα 3.1.3: Σύνδεση θήκης μπαταριών στον ελεγκτή των κινητήρων

Στο προτελευταίο βήμα της συναρμολόγησης, εγκαθιστούμε τον αισθητήρα υπερήχων στο breadboard και τοποθετούμε το Infiduino και το breadboard στο πάνω μέρος του οχήματος (εικόνα 3.1.4).



Εικόνα 3.1.4: Σύνδεση αισθητήρα υπερήχων, breadboard και Infiduino στο πάνω επίπεδο του οχήματος

Όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, το breadboard θα είναι η βάση πάνω στην οποία θα γίνουν όλες οι συνδέσεις των υποσυστημάτων. Στην συνέχεια, συνδέουμε το Infiduino με τον αισθητήρα

υπερήχων χρησιμοποιώντας το breadboard ακολουθώντας την συνδεσμολογία του πίνακα 3.1.6.

Ονομασία Ακροδέκτη Αισθητήρα	Ονομασία Ακροδέκτη Infiduino	Λειτουργία
VCC	5V	Παροχή Ρεύματος
Trig	12	Μεταφορά Δεδομένων
Echo	13	Μεταφορά Δεδομένων
Gnd	GND	Γείωση

Πίνακας 3.1.6: Σύνδεση αισθητήρα υπερήχων με Infiduino

Τέλος, ενώνουμε τα επίπεδα του οχήματος *(εικόνα 2.3.3.1)* μεταξύ τους, τοποθετούμε την θήκη των μπαταριών στο πίσω μέρος του πάνω επιπέδου και συνδέουμε τον ελεγκτή των κινητήρων με το Infiduino μέσω των ψηφιακών ακροδεκτών του ακολουθώντας την συνδεσμολογία του πίνακα 3.1.8.

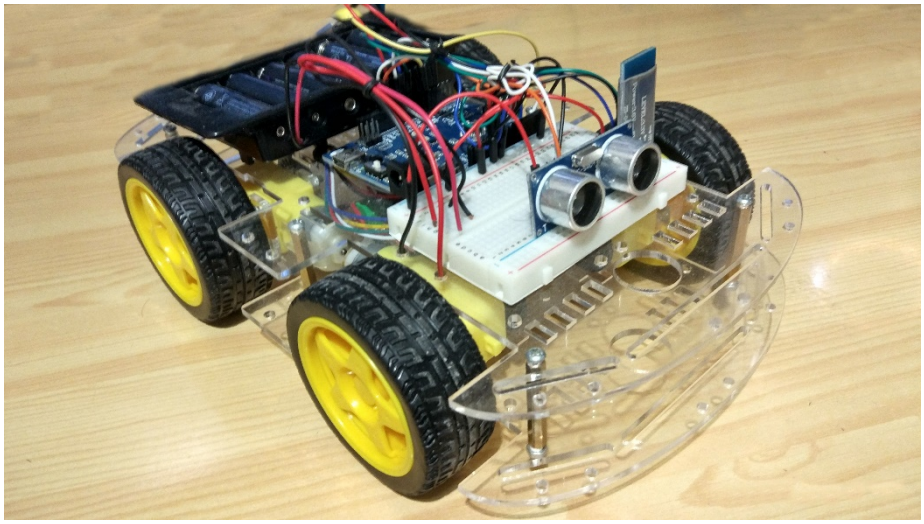
Ονομασία Ακροδέκτη Υποσυστήματος Κινητήρων	Ονομασία Ακροδέκτη Infiduino
ENA	11
IN1	6
IN2	5
IN3	4
IN4	3
ENB	10

Πίνακας 3.1.8: Σύνδεση Infiduino με συσκευή ελέγχου κινητήρων

Αφού συνδέσουμε και τον Bluetooth προσαρμογέα βάση του πίνακα 3.1.9, το όχημα είναι έτοιμο προς λειτουργία *(εικόνα 3.1.10)*.

Ονομασία Ακροδέκτη Προσαρμογέα Bluetooth	Ονομασία Ακροδέκτη Infiduino
VCC	5V
GND	GND
TXD	9
ENB	10

Πίνακας 3.1.9: Σύνδεση Infiduino με συσκευή ελέγχου κινητήρων

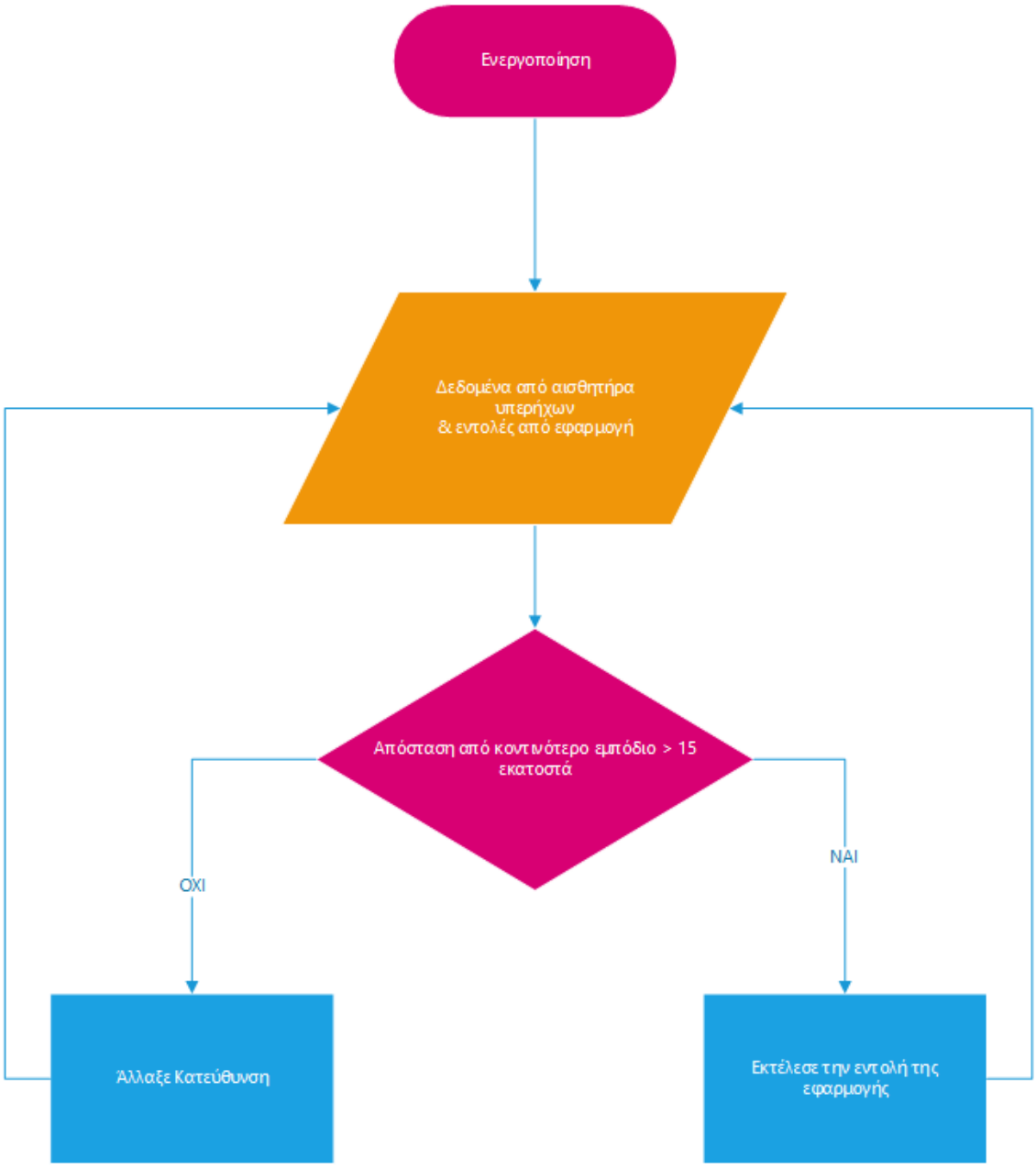


Εικόνα 3.1.10: Τελική μορφή οχήματος

3.2 Λειτουργία

Για την επιτυχημένη λειτουργία του οχήματος, το Infiduino προγραμματίζεται με τέτοιον τρόπο έτσι ώστε να δίνει εντολές στην συσκευή ελέγχου κινητήρων με βάση κάποια ερεθίσματα που δέχεται. Πιο συγκεκριμένα, το Infiduino δέχεται τις εντολές οι οποίες αποστέλλονται σε αυτό μέσω Bluetooth από κάποια άλλη συσκευή με χρήση της εφαρμογής η οποία παρουσιάζεται σε επόμενο κεφάλαιο καθώς και δεδομένα από τον αισθητήρα υπερήχων. Οι εντολές που δέχεται από την εφαρμογή μέσω Bluetooth αφορούν την πορεία κατεύθυνσης του οχήματος ενώ τα δεδομένα που δέχεται από τον αισθητήρα υπερήχων αφορούν την απόσταση που έχει το όχημα από το κοντινότερο εμπόδιο.

Αναλυτικότερα, η διαδικασία που ακολουθεί το όχημα κατά την λειτουργία του παρουσιάζεται στο διάγραμμα ροής 3.2.1



Διάγραμμα Ροής 3.2.1: Λειτουργία Οχήματος

3.3 Έλεγχος Σωστής Λειτουργίας

Για τον έλεγχο σωστής λειτουργίας του οχήματος, το Infiduino προγραμματίστηκε έτσι ώστε να κινείται στον χώρο μόνο του χωρίς να

λαμβάνει εντολές από την εφαρμογή ενώ ταυτόχρονα αποφεύγει όποια εμπόδια εντοπίσει μπροστά του.

Ακολουθεί ο κώδικας ο οποίος συντάχθηκε για τον σκοπό αυτό.

```
#include <NewPing.h>
#include <PCD8544.h>

/* Αρχή Αισθητήρα */
//Δηλώνω στο Infiduino σε ποιους ακροδέκτες είναι συνδεδεμένος ο αισθητήρας υπερήχων
NewPing sonar(12, 13);
/* Τέλος Αισθητήρα */
/* Αρχή Κινητήρων */
// Ακροδέκτες για τον κινητήρα A (enableA = ενεργοποίηση κινητήρα, pinA1 = κίνηση εμπρός, pinA2 = κίνηση πίσω)
int enableA = 11;
int pinA1 = 6; /* πράσινο καλώδιο */
int pinA2 = 5; /* μπλε καλώδιο */
// Ακροδέκτες για τον κινητήρα B (enabledB = ενεργοποίηση κινητήρα, pinB2 = κινηση εμπρός, pinB2 = κίνηση πίσω)
int enableB = 10;
int pinB1 = 4; /* μαύρο καλώδιο */
int pinB2 = 3; /* κίτρινο καλώδιο */
long cm;
/* Τέλος Κινητήρων */
/* Αρχή Οθόνης */
// Εμφάνιση ενός emoticon
static const byte glyph[] = { B00010000, B00110100, B00110000, B00110100, B00010000 };
static PCD8544 lcd;
/* Τέλος Οθόνης */

void setup() {
  /* Αρχή Αισθητήρα */
  // Ενεργοποίηση λειτουργίας Σειριακής Παρακολούθησης έτσι ώστε να βλέπουμε την έξοδο του αισθητήρα
  Serial.begin(9600);
  /* Τέλος Αισθητήρα */
  /* Αρχή Κινητήρων */
  pinMode(enableA, OUTPUT);
  pinMode(pinA1, OUTPUT);
  pinMode(pinA2, OUTPUT);
  pinMode(enableB, OUTPUT);
  pinMode(pinB1, OUTPUT);
  pinMode(pinB2, OUTPUT);
  /* Τέλος Κινητήρων */
  /* Αρχή Οθόνης */
  lcd.begin(84, 48);
  /*
  // Προσθήκη του emoticon που δημιουργήθηκε πριν στην θέση 0 του πίνακα ASCII
```

```
  lcd.createChar(0, glyph);
  */
  /* Τέλος Οθόνης */
}

void loop() {
  /* Αρχή Κινητήρων */
  // Λειτουργία κινητήρων σε χαμηλότερη ταχύτητα από την μέγιστη
  analogWrite(enableA, 170);
  analogWrite(enableB, 170);
  /* Τέλος Κινητήρων */
  /* Αρχή Οθόνης */
  // Χρήση του αισθητήρα υπερήχων για να δούμε την απόσταση σε εκατοστά
από το κοντινότερο εμπόδιο
  cm = sonar.ping_cm();
  // Εμφάνιση της παραπάνω πληροφορίας στην Σειριακή Παρακολούθηση
  Serial.print(cm);
  Serial.print(" cm.");
  Serial.print("\n");
  // Εμφάνιση της παραπάνω πληροφορίας στην οθόνη του οχήματος
  lcd.clear();   // Εκκαθάριση οθόνης και προσωρινής μνήμης buffer
  if (cm < 15) {
    lcd.setCursor(0, 0);
    lcd.print("cm:");
    lcd.setCursor(0, 1);
    lcd.print(cm, DEC);
    lcd.setCursor(0, 2);
    lcd.print("Danger!!!! Going back....");
  }
  else {
    lcd.setCursor(0, 0);
    lcd.print("cm:");
    lcd.setCursor(0, 1);
    lcd.print(cm, DEC);
  }
  delay(200);
  // Αν το όχημα εντοπίσει εμπόδιο σε απόσταση μικρότερη των δεκαπέντε
εκατοστών τότε θα κάνει πίσω και θα στρίψει αριστερά. Αν δεν εντοπίσει
κάποιο εμπόδιο, θα κινηθεί προς τα εμπρός
  if (cm < 15) {
    analogWrite(enableA, 255);
    analogWrite(enableB, 255);
    backward(700);
    turnLeft(700);
  }
  else {
    forward(1);
  }
}

// Δημιουργία εντολών προς το κύκλωμα ελέγχου των κινητήρων
void enableMotors() {
  motorAOn();
  motorBOn();
}
```

```
void disableMotors() {
  motorAOff();
  motorBOff();
}
void forward(int time) {
  motorAForward();
  motorBForward();
  delay(time);
}
void backward(int time) {
  motorABackward();
  motorBBackward();
  delay(time);
}
void turnLeft(int time) {
  motorABackward();
  motorBForward();
  delay(time);
}
void turnRight(int time) {
  motorAForward();
  motorBBackward();
  delay(time);
}
void brake(int time) {
  motorABrake();
  motorBBrake();
  delay(time);
}
void motorAOn() {
  digitalWrite(enableA, HIGH);
}
void motorBOn() {
  digitalWrite(enableB, HIGH);
}
void motorAOff() {
   digitalWrite(enableB, LOW);
}
void motorBOff() {
  digitalWrite(enableA, LOW);
}
void motorAForward() {
  digitalWrite(pinA1, HIGH);
  digitalWrite(pinA2, LOW);
}
void motorABackward() {
  digitalWrite(pinA1, LOW);
  digitalWrite(pinA2, HIGH);
}
void motorBForward() {
  digitalWrite(pinB1, HIGH);
  digitalWrite(pinB2, LOW);
}
void motorBBackward() {
  digitalWrite(pinB1, LOW);
  digitalWrite(pinB2, HIGH);
```

```
}
void motorABrake() {
  digitalWrite(pinA1, HIGH);
  digitalWrite(pinA2, HIGH);
}
void motorBBrake() {
  digitalWrite(pinB1, HIGH);
  digitalWrite(pinB2, HIGH);
}
```

3.4 Προγραμματισμός Οχήματος

Το επόμενο στάδιο μετά τον επιτυχή έλεγχο λειτουργίας του οχήματος είναι ο προγραμματισμός αυτού για να δέχεται εντολές μέσω Bluetooth από την εφαρμογή.

Κατά τις δοκιμές που έγιναν παρατηρήθηκε αργή απόκριση του οχήματος στα ερεθίσματα που παίρνει από το περιβάλλον και στις αντιδράσεις του. Για την αντιμετώπιση του προβλήματος αυτού, αφαιρέθηκε από το όχημα εντελώς η οθόνη στην οποία χρησιμοποιούταν για την εμφάνιση μηνυμάτων. Με τον τρόπο αυτό, το πρόβλημα διορθώθηκε σε κάποιο βαθμό. Το παραπάνω φαινόμενο πιθανό να οφείλεται στις περιορισμένες δυνατότητες του Infiduino.

Στην συνέχεια, ακολουθεί ο κώδικας που συντάχθηκε και χρησιμοποιήθηκε για το όχημα του παρόντος έργου.

```
#include <NewPing.h>
#include <SoftwareSerial.h>

// Δηλώνω στο Infiduino σε ποιους ακροδέκτες είναι συνδεδεμένος η
Bluetooth κεραία
SoftwareSerial BTserial(9, 8); // RX, TX
// Δηλώνω στο Infiduino σε ποιους ακροδέκτες είναι συνδεδεμένος ο
αισθητήρας υπερήχων
NewPing sonar(12, 13);
/* Αρχή Κινητήρων */
// Ακροδέκτες για τον κινητήρα Α (enableA = ενεργοποίηση κινητήρα, pinA1
= κίνηση εμπρός, pinA2 = κίνηση πίσω)
```

```
int enableA = 11;
int pinA1 = 6; /* πράσινο καλώδιο */
int pinA2 = 5; /* μπλε καλώδιο */
// Ακροδέκτες για τον κινητήρα B (enabledB = ενεργοποίηση κινητήρα,
pinB2 = κινηση εμπρός, pinB2 = κίνηση πίσω)
int enableB = 10;
int pinB1 = 4; /* μαύρο καλώδιο */
int pinB2 = 3; /* κίτρινο καλώδιο */
long cm;
/* Τέλος Κινητήρων */
char data = 0;  // Μεταβλητή για αποθήκευση εισερχόμενων δεδομένων

void setup() {
  /* Αρχή Κινητήρων */
  pinMode(enableA, OUTPUT);
  pinMode(pinA1, OUTPUT);
  pinMode(pinA2, OUTPUT);
  pinMode(enableB, OUTPUT);
  pinMode(pinB1, OUTPUT);
  pinMode(pinB2, OUTPUT);
  /* Τέλος Κινητήρων */
  BTserial.begin(9600);   // Θέτει τον ρυθμό μετάδοσης δεδομένων μέσω
του Bluetooth
  BTserial.println("Serial begun");
}

void loop() {
  // Χρήση του αισθητήρα υπερήχων για να δούμε την απόσταση σε εκατοστά
από το κοντινότερο εμπόδιο
  cm = sonar.ping_cm();
  // Εμφάνιση της παραπάνω πληροφορίας στην Σειριακή Παρακολούθηση
  BTserial.print("The distance to the nearest object is ");
  BTserial.print(cm);
  BTserial.print(" cm.");
  BTserial.print("\n");
  if (cm < 15) {  // Αν το όχημα εντοπίσει εμπόδιο σε απόσταση μικρότερη
των δεκαπέντε εκατοστών τότε θα κάνει πίσω και θα στρίψει αριστερά. Αν
δεν εντοπίσει κάποιο εμπόδιο, θα κινηθεί προς τα εμπρός
    analogWrite(enableA, 255);
    analogWrite(enableB, 255);
    backward(700);
    turnLeft(700);
  }
  if (BTserial.available() && cm > 15) {
    BTserial.println("Serial connection is up!!!");
    data = BTserial.read();     // Λήψη εισερχόμενων μέσω Bluetooth
δεδομένων και ανάθεση τους στην μεταβλητή data
    BTserial.println(data);    // Εκτύπωση του περιεχόμενου της
μεταβλητής data στην Σειριακή Παρακολούθηση
    enableMotors();
    if(data == 'f') {
      forward(100);
      BTserial.println("Going forwards");

    }
    else if(data == 'b') {
```

```
      backward(100);
      BTserial.println("Going backwards");

    }
    else if(data == 'r') {
      turnRight(200);
    }
    else if (data == 'l') {
      turnLeft(200);
    }
    else if(data == 's') {
      disableMotors();
      BTserial.println("Stopped");
    }
  }
}

// Δημιουργία εντολών προς το κύκλωμα ελέγχου των κινητήρων
void enableMotors() {
  motorAOn();
  motorBOn();
}
void disableMotors() {
  motorAOff();
  motorBOff();
}
void forward(int time) {
  motorAForward();
  motorBForward();
  delay(time);
}
void backward(int time) {
  motorABackward();
  motorBBackward();
  delay(time);
}
void turnLeft(int time) {
  motorABackward();
  motorBForward();
  delay(time);
}
void turnRight(int time) {
  motorAForward();
  motorBBackward();
  delay(time);
}
void brake(int time) {
  motorABrake();
  motorBBrake();
  delay(time);
}
void motorAOn() {
  digitalWrite(enableA, HIGH);
}
void motorBOn() {
  digitalWrite(enableB, HIGH);
}
void motorAOff()
   digitalWrite(enableB, LOW);
}
```

```
void motorBOff() {
  digitalWrite(enableA, LOW);
}
void motorAForward() {
  digitalWrite(pinA1, HIGH);
  digitalWrite(pinA2, LOW);
}
void motorABackward() {
  digitalWrite(pinA1, LOW);
  digitalWrite(pinA2, HIGH);
}
void motorBForward() {
  digitalWrite(pinB1, HIGH);
  digitalWrite(pinB2, LOW);
}
void motorBBackward() {
  digitalWrite(pinB1, LOW);
  digitalWrite(pinB2, HIGH);
}
void motorABrake() {
  digitalWrite(pinA1, HIGH);
  digitalWrite(pinA2, HIGH);
}
void motorBBrake() {
  digitalWrite(pinB1, HIGH);
  digitalWrite(pinB2, HIGH);
}
```

4. ΕΦΑΡΜΟΓΗ ΧΕΙΡΙΣΜΟΥ ΟΧΗΜΑΤΟΣ

4.1 Γλώσσα και Περιβάλλον Προγραμματισμού

4.1.1 Επιλογή Φύσης Εφαρμογής

Στον τομέα της ανάπτυξης εφαρμογής για φορητές συσκευές η πρώτη απόφαση που καλείται να πάρει ο προγραμματιστής είναι το αν η εφαρμογή που θα αναπτύξει θα είναι Διαδικτυακής (Web), Native ή Υβριδικής φύσης. Κάθε μία από τις επιλογές αυτές έχει τα πλεονεκτήματα και τα μειονεκτήματα της όπως αυτά αναφέρονται στον πίνακα 4.1.1.1.

Κριτήριο	Διαδικτυακή (Web)	Native	Υβριδική (Hybrid)
Λειτουργικότητα	Περιορισμένη	Μέγιστη	Μέτρια
Επιδόσεις	Περιορισμένες	Μέγιστες	Μέτριες
Χρόνος Ανάπτυξης	Μικρός	Μεγάλος	Μέτριος
Κόστος Ανάπτυξης	Μικρό	Μεγάλο	Μέτριο
Εμπειρία Χρήστη	Περιορισμένη	Μέγιστη	Μέτρια
Φορητότητα σε λειτουργικά συστήματα	Ναι	Όχι	Ναι
Συντήρηση Εφαρμογής	Εύκολη	Δύσκολη	Εύκολη
Διανομή μέσω ηλεκτρονικού καταστήματος πλατφόρμας	Όχι	Ναι	Ναι

Πίνακας 4.1.1.1: Σύγκριση μεταξύ Web, Native και Hybrid υλοποίησης εφαρμογής

Είναι ξεκάθαρο από τον παραπάνω πίνακα ότι οι Διαδικτυακές εφαρμογές έχουν πολύ περιορισμένες δυνατότητες και επιδόσεις, προσφέρουν περιορισμένη εμπειρία χρήστη (User Experience) ενώ όμως λειτουργούν σε οποιαδήποτε πλατφόρμα (Cross - Platform Compatibility) και απαιτούν πολύ λίγο χρόνο ανάπτυξης κάτι το οποίο έχει ως αποτέλεσμα την μείωση του κόστους ανάπτυξης.

Οι Native εφαρμογές προσφέρουν πλήρη λειτουργικότητα, την καλύτερη εμπειρία χρήσης (User Experience) και μέγιστη απόδοση λόγω δυνατότητας άμεσης συνεργασίας με το υλικό και το λογισμικό της συσκευής στην οποία εκτελούνται. Ταυτόχρονα όμως απαιτούν μεγάλο χρόνο ανάπτυξης κυρίως λόγω της αδυναμίας τους να εκτελεστούν σε πολλαπλές πλατφόρμες (Cross – Platform Compatibility) κάτι το οποίο επιδρά αρνητικά στην πολυπλοκότητα και στο κόστους ανάπτυξης της εφαρμογής.

Το κενό μεταξύ των δύο αυτών λύσεων καλύπτουν οι Υβριδικές εφαρμογές οι οποίες βασίζονται σε διαδικτυακές τεχνολογίες απεικόνισης και αλληλεπίδρασης με το περιβάλλον της εφαρμογής. Ταυτόχρονα όμως χρησιμοποιούν ένα native κέλυφος το οποίο υπερκαλύπτει την διαδικτυακή φύση της εφαρμογής προσφέροντας σε αυτή σε κάποιο βαθμό τα πλεονεκτήματα μιας Native εφαρμογής.

Για το παρόν έργο, η επιλογή ανάπτυξης Διαδικτυακής (Web) εφαρμογής δεν είναι δυνατή καθώς το να έχουμε άμεση επαφή με το υλικό της κινητής συσκευής ελέγχου του οχήματος και πιο συγκεκριμένα με το Bluetooth είναι απαραίτητο.

4.1.2 Ορισμός Framework

Ανέκαθεν, ένα από τα πιο βασικά στάδια σχεδίασης της υλοποίησης μίας εφαρμογής ήταν η επιλογή της γλώσσας προγραμματισμού. Τα τελευταία χρόνια, το στάδιο αυτό έχει εξελιχθεί καθώς μελετάει περισσότερο την επιλογή του κατάλληλου framework παρά της γλώσσας για την υλοποίηση της εφαρμογής.

Ως framework στον προγραμματισμό χαρακτηρίζουμε μία αφαιρετική δομή η οποία έχει δημιουργηθεί με σκοπό να προσφέρει συγκεντρωμένα βασικά εργαλεία, δομές και λειτουργίες οι οποίες διευκολύνουν σε πολύ μεγάλο βαθμό τον προγραμματιστή. Αυτό το επιτυγχάνει ικανοποιώντας μία από τις βασικότερες αρχές του αντικειμενοστραφούς προγραμματισμού την επαναχρησιμοποίηση κώδικα. Αναλυτικότερα, προσφέρει ένα σύνολο κώδικα και λειτουργιών το οποίο παρέχει λύση σε ανάγκες που υπάρχουν στις περισσότερες εφαρμογές όπως η σωστή εμφάνιση μίας διαδικτυακής εφαρμογής σε όλα τους τύπους των συσκευών και κατά επέκταση οθονών είτε αυτές είναι κινητά τηλέφωνα είτε σταθεροί υπολογιστές ή η διαχείριση χρηστών συστήματος. Έτσι, ο προγραμματιστής δεν αφιερώνει χρόνο στο να συντάξει κώδικα για κάποιες λειτουργίες (χαμηλού επιπέδου) του προγράμματος καθώς αυτές έχουν ήδη υλοποιηθεί στο παρελθόν από κάποιους άλλους προγραμματιστές δίνοντας του έτσι την δυνατότητα να εστιάσει περισσότερο στην ουσιαστική φύση της εφαρμογής που αναπτύσσει.

4.1.3 Επιλογή Framework

Κατά την διάρκεια μελέτης και υλοποίησης του συγκεκριμένου έργου, υπήρχαν δύο κύριες επιλογές frameworks σύγκριση των οποίων γίνεται στον πίνακα 4.1.2.1.

Κριτήριο	React Native	AngularJS
Χρόνος Εκμάθησης	Μέτριος	Μεγάλος
Πλήθος Δυνατοτήτων	Περιορισμένο	Μεγάλο
Ταχύτητα	Μεγάλη	Μέτρια
Δημιουργός	Facebook	Google
Έτος Κυκλοφορίας	2013	2010
Αρχιτεκτονική	View	Model View Controller
Υποστήριξη Πλατφόρμων	Android, iOS	Android, iOS, Windows Phone
Φύση Εφαρμογής	Native	Hybrid
Γλώσσα προγραμματισμού	Javascript / JSX	HTML, CSS, JS
Επίπεδο Αφαιρετικότητς	Μεγάλο	Μέτριο
Χρήση Bluetooth υλικού συσκευής	Υποστηρίζεται	Υποστηρίζεται

Πίνακας 4.1.2.1: Σύγκριση frameworks

Βασικό στοιχείο και των δύο επιλογών είναι η δυνατότητα χρήσης του υλικού Bluetooth της συσκευής ελέγχου όπως και η υποστήριξη πολλαπλών

λειτουργικών συστημάτων για κινητές συσκευές. Για το παρόν έργο θα γίνει χρήση του React Native framework καθώς προσφέρει το πλεονέκτημα δημιουργίας Native εφαρμογής.

4.1.4 React Native

Χαρακτηριστικό του React Native framework είναι η δυνατότητα χρήσης της γλώσσας προγραμματισμού Javascript. Ταυτόχρονα δίνεται η δυνατότητα στον προγραμματιστή να χρησιμοποιήσει την γλώσσα σήμανσης eXtensible Markup Language (XML) για το κομμάτι του Περιβάλλοντος Διεπαφής με τον χρήστη (User Interface). Ο συνδυασμός των δύο αυτών γλωσσών δημιούργησε την JSX χρήση της οποίας γίνεται στην παρούσα εφαρμογή.

Κατά την διαδικασία δημιουργίας εφαρμογής με το εν λόγω framework, δημιουργούνται αρχεία κάθε ένα από τα οποία εμπεριέχει ένα Συστατικό (Component) του προγράμματος. Το Συστατικό αυτό χρησιμοποιεί μία δομή Λειτουργίας (Function) ή Κλάσης (Class) ανάλογα με τις ανάγκες του. Πιο συγκεκριμένα, στην πρώτη περίπτωση η Λειτουργία δέχεται μία είσοδο η οποία ονομάζεται "props" και επιστρέφει ένα αντικείμενο. Αυτή είναι η πιο απλή δομή του framework και προτείνεται για μη δυναμικό περιεχόμενο. Η Κλάση σαν δομή υποστηρίζει πρόσθετες λειτουργίες όπως ο έλεγχος για συμβάντα και η αντίδραση σε αυτά ενώ ελέγχει ταυτόχρονα την κατάσταση του κώδικα που εκτελείται μέσω αυτής.

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, το React Native είναι ένα αρκετά περιορισμένο framework το οποίο χρησιμεύει στην δημιουργία Περιβάλλοντος Διεπαφής Χρήστη. Για την χρήση της λειτουργίας Bluetooth

της κινητής συσκευής, απαραίτητη κρίθηκε η χρήση ενός plugin και πιο συγκεκριμένα του React Native Bluetooth Serial το οποίο καλύπτει το χάσμα αυτό. Το εν λόγω plugin είναι στην ουσία ένα σύνολο κώδικα το οποίο έχει δημιουργηθεί ικανοποιώντας τις συνθήκες επαναχρησιμοποίησης κώδικα και αφαιρετικότητας με σκοπό την μεταφορά σειριακών δεδομένων μέσω του πρωτοκόλλου Bluetooth.

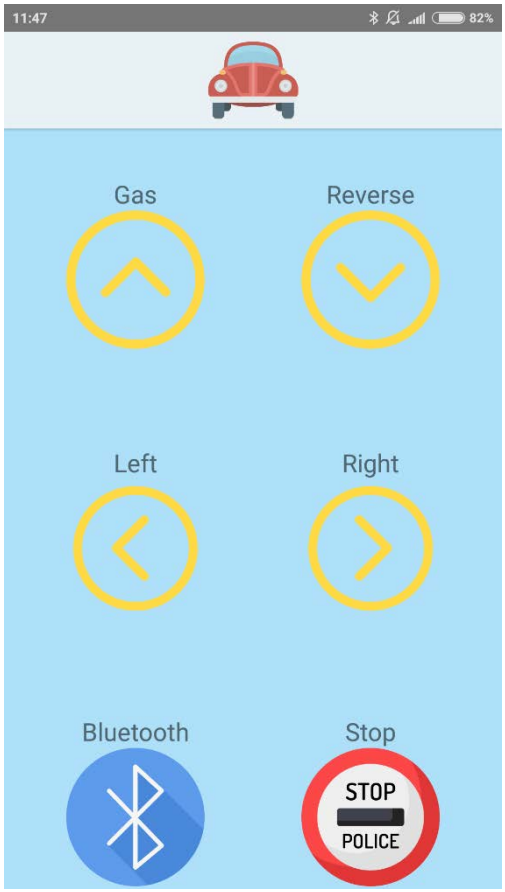
Για την επιτυχής σύνταξη του κώδικα για Android περιβάλλον, απαραίτητη είναι και η παρουσία της εφαρμογής Android Studio η οποία εκτός από εξομοιωτή (emulator) Android συσκευής παρέχει πρόσβαση στα απαραίτητα αρχεία συστήματος Android όπως το Software Development Kit (SDK) και το Android Native Development Kit. Για το περιβάλλον iOS απαιτούνται τα αντίστοιχα εργαλεία και πιο συγκεκριμένα το Xcode το οποίο όπως και τα SDK και NDK είναι ένα Ενσωματωμένο Περιβάλλον Προγραμματισμού.

Λόγω μη υποστήριξης του Xcode στο λειτουργικό σύστημα Windows και έλλειψης κινητής συσκευής που να χρησιμοποιεί το λειτουργικό σύστημα iOS, στην παρούσα πτυχιακή εργασία θα μελετήσουμε την δημιουργία της εφαρμογής μόνο για το περιβάλλον Android.

4.1.5 Λειτουργία Εφαρμογής

Στόχος της εφαρμογής που παρουσιάζεται στο παρόν κεφάλαιο είναι να στέλνει εντολές ελέγχου στο όχημα. Για την επίτευξη αυτού, η εφαρμογή παρέχει στον χρήστη μία οθόνη μέσω της οποίας εκείνος μπορεί να επιλέξει την εντολή που θα στείλει στο όχημα *(εικόνα 4.2.1)*.

Κατά την έναρξη της εφαρμογής και αφού το Bluetooth είναι ενεργοποιημένο στην κινητή συσκευή, η πρώτη εντολή που καλείται να επιλέξει ο χρήστης είναι αυτή του Bluetooth. Κατά την εκτέλεση αυτής της εντολής, η κινητή συσκευή αναλαμβάνει να εντοπίσει και να συνδεθεί με το όχημα, έχοντας ως αναγνωριστική την μοναδική διεύθυνση MAC της Bluetooth κεραίας του οχήματος.



Εικόνα 4.2.1.: Περιβάλλον Χρήσης της εφαρμογής

Αφού επιτευχθεί η σύνδεση, ο χρήστης έχει την δυνατότητα να χρησιμοποιήσει κάποια από τις υπόλοιπες διαθέσιμες εντολές έτσι ώστε να θέσει το όχημα σε κίνηση. Στην περίπτωση που χρησιμοποιήσει την εντολή "Stop", το όχημα ακινητοποιείται.

4.1.6 Υλοποίηση Εφαρμογής

Για την δημιουργία της εφαρμογής έγινε χρήση της έκδοσης 0.42.0 του React Native framework σε συνδυασμό με την εφαρμογή Atom ως περιβάλλον ανάπτυξης λογισμικού (Integrated Development Environment).

Ακολουθεί ο κώδικας της εφαρμογής χωρισμένος ανά αρχείο. Σε κάθε αρχείο υπάρχει από ένα Συστατικό (Component) της εφαρμογής).

4.1.6.1 Αρχείο index.android.js

Το αρχείο αυτό είναι το πρώτο το οποίο χρησιμοποιείται κατά την εκτέλεση της εφαρμογής σε περιβάλλον Android. Στόχος του αρχείου είναι να «δηλώσει» την τοποθεσία της εφαρμογής και να κατευθύνει την εκτέλεση της εφαρμογής στην τοποθεσία αυτή *(./src/App)* μέσω της εντολής *RegisterComponent* του πακέτου *AppRegistry* του framework.

```
import { AppRegistry } from 'react-native';
import App from './src/App';

AppRegistry.registerComponent('arduauto', () => App);
```

4.1.6.2 Αρχείο index.ios.js

Το αρχείο αυτό απευθύνεται σε συσκευές που χρησιμοποιούν το λειτουργικό σύστημα iOS, κάνει την ίδια ακριβώς δουλειά που κάνει το αρχείο index.android.js, και έχει ακριβώς το ίδιο περιεχόμενο.

4.1.6.3 Συστατικό App.js

Το συστατικό αυτό αποτελεί στην ουσία την αρχή της εφαρμογής. Σε αυτό, καλούνται τα απαραίτητα στοιχεία του React Native framework και ξεκινάει η διαδικασία σχεδιασμού (Rendering) του πλαισίου της εφαρμογής στην οθόνη της κινητής συσκευής. Κατά την διαδικασία αυτή, το τρέχων συστατικό καλεί με την σειρά του τα συστατικά *Header.js* και *CommandsFrame.js* που αναλύονται στην συνέχεια.

```
  // Εισαγωγή απαραίτητων λειτουργιών από το React και των  συστατικών που θα κληθούν από αυτό
  import React, { Component } from 'react';
  import { View } from 'react-native';
  import Header from './components/Header';
  import CommandsFrame from './components/CommandsFrame';

  // Δημιουργία συστατικού
  class App extends Component {
    render() {
      return (
        // Στο tag style του View παρατηρούμε την χρήση της XML που αναφέρθηκε σε προηγούμενο κεφάλαιο και χρησιμεύει στον σχεδιασμό της εμφάνισης της εφαρμογής
        <View style={{ flex: 1, backgroundColor: '#AEE0F9' }}>
        // Το string ArduAuto περνάει στο συστατικό Header ως “prop” μέσω του tag headerText
          <Header headerText={'ArduAuto'} />
          <CommandsFrame />
        </View>
      );
    }
  }
  // Εδώ δίνουμε εντολή να ξεκινήσει το rendering
  export default App;
```

4.1.6.4 Συστατικό Header.js

Σε αυτό το αρχείο σχεδιάζουμε την οριζόντια γραμμή η οποία βρίσκεται στην κορυφή της εφαρμογής και εμπεριέχει το λογότυπο της εφαρμογής.

```
import React from 'react';
import { View, Image } from 'react-native';

const Header = () => {
      // Ενημέρωση της Λειτουργίας ότι υπάρχει κομμάτι κώδικα σχετικό με τον τρόπο εμφάνισης της λειτουργίας
  const { viewStyle, imageStyle } = styles;

  return (
        // Κλήση του κώδικα που προαναφέρθηκε
    <View style={viewStyle}>
      <Image
        style={imageStyle}
        source={require('./../../images/car.png')}
      />
    </View>
```

```
  );
};
// Δήλωση κώδικα τύπου CSS για ιδιότητες εμφάνισης των στοιχείων της εφαρμογής
const styles = {
  viewStyle: {
    backgroundColor: '#FFF',
    opacity: 0.8,
    height: 70,
    justifyContent: 'center',
    alignItems: 'center',
    shadowColor: '#424242',
    shadowOffset: { width: 0, height: 70 },
    shadowOpacity: 0.1,
    elevation: 2,
    position: 'relative'
  },
  imageStyle: {
    width: 65,
    height: 65
  }
};

// Ενημερώνουμε τα υπόλοιπα συστατικά του προγράμματος για την ύπαρξη του Header συστατικού
export default Header;
```

4.1.6.5 Συστατικό CommandsFrame.js

Στο συστατικό αυτό δημιουργούμε το πλαίσιο στο οποίο εμφανίζονται οι εντολές που υποστηρίζει η εφαρμογή. Εισάγοντας και χρησιμοποιώντας το "ScrollView" από το React Native, δίνουμε την δυνατότητα στον χρήστη να κάνει πάνω και κάτω κύλιση της οθόνης στην περίπτωση που δεν εμφανίζονται όλες οι εντολές λόγω ελλιπούς χώρου. Το τρέχον συστατικό καλεί το επόμενο συστατικό το οποίο είναι το "CommandsList".

```
import React from 'react';
import { ScrollView } from 'react-native';
import CommandsList from './CommandsList';

const CommandsFrame = () => {
  return (
```

```
    <ScrollView style={{ flex: 1, backgroundColor: '#AEE0F9' }}>
      <CommandsList />
    </ScrollView>
  );
};

export default CommandsFrame;
```

4.1.6.6 Συστατικό CommandsList.js

Το παρόν συστατικό είναι ίσως το πιο σημαντικό καθώς σε αυτό γίνεται η δημιουργία των επιλογών της εφαρμογής και η ανάθεση εντολών σε αυτές. Η στοίχιση των αντικειμένων (εικονίδια εντολών) στην οθόνη γίνεται με την βοήθεια του flexbox το οποίο είναι ένα σύνολο εντολών που δημιουργήθηκε για αυτόν ακριβώς τον σκοπό.

```
import React from 'react';
import { View } from 'react-native';
// Εισαγωγή του plugin που είναι υπεύθυνο για την λειτουργία του
Bluetooth
import bluetoothSerial from 'react-native-bluetooth-serial';
// Εισαγωγή απαραίτητων συστατικών
import Command from './Command';
import CommandsListSection from './CommandsListSection';

const CommandsList = () => {
  return (
    <CommandsListSection>
          // Χρήση flexbox μέσω ορισμού του flexDirection
      <View style={{ flexDirection: 'row' }}>
            // Δήλωση στοιχείων (props) που περνάνε στο συστατικό
Command ο οποίο καλείται στο σημείο αυτό
        <Command
          itemText='Gas' [ /* ονομασία εντολής */ ]
          itemImage={require('./../../images/up.png')} [ /* μονοπάτι
εικόνας εντολής */ ]
          itemFunction={bluetoothSerial.write} [ /* εντολή που καλείται
από το Bluetooth plugin */ ]
          itemFunctionParameter={'f'} [ /* Παράμετρος που δίνεται στην
παραπάνω εντολή αφού κληθεί */ ]
        />
        <Command
          itemText='Reverse'
          itemImage={require('./../../images/down.png')}
          itemFunction={bluetoothSerial.write}
          itemFunctionParameter={'b'}
```

```
        />
      </View>
      <View style={{ flexDirection: 'row' }}>
        <Command
          itemText='Left'
          itemImage={require('./../../images/left.png')}
          itemFunction={bluetoothSerial.write}
          itemFunctionParameter={'l'}
        />
        <Command
          itemText='Right'
          itemImage={require('./../../images/right.png')}
          itemFunction={bluetoothSerial.write}
          itemFunctionParameter={'r'}
        />
      </View>
      <View style={{ flexDirection: 'row' }}>
        <Command
          itemText='Bluetooth'
          itemImage={require('./../../images/bluetooth.png')}
          itemFunction={bluetoothSerial.connect}
          itemFunctionParameter={'20:16:08:04:05:68'}
        />
        <Command
          itemText='Stop'
          itemImage={require('./../../images/stop.png')}
          itemFunction={bluetoothSerial.write}
          itemFunctionParameter={'s'}
        />
      </View>
    </CommandsListSection>
  );
};

export default CommandsList;
```

4.1.6.7 Συστατικό CommandsListSection.js

Το συγκεκριμένο συστατικό δεν έχει κάποια ιδιαίτερη λειτουργία. Η μοναδική χρησιμότητα είναι να διαμορφώνει την εμφάνιση του πλαισίου των εντολών που αναφέρθηκα προηγούμενως.

```
import React from 'react';
import { View } from 'react-native';

const CommandsListSection = (props) => {
  const { containerStyle } = styles;

  return (
      <View style={containerStyle}>
```

```
        {props.children}
      </View>
  );
};

const styles = {
  containerStyle: {
    flexDirection: 'column',
    justifyContent: 'space-between',
    alignItems: 'center',
    elevation: 1,
    marginLeft: 5,
    marginRight: 5
  }
};

export default CommandsListSection;
```

4.1.6.8 Συστατικό Command.js

Σε αυτό το συστατικό διαμορφώνουμε την εμφάνιση και την λειτουργία της κάθε επιλογής από τις εντολές που προσφέρει η εφαρμογή.

```
import React from 'react';
import { Text, Image, TouchableOpacity } from 'react-native';
import CommandContainer from './CommandContainer';

const Command = (props) => {

  const { imageTitleTextStyle, itemStyle, imageStyle } = styles;
// αποθήκευση των αντίστοιχων props στις αντίστοιχες σταθερές
const fn = props.itemFunction;
const p = props.itemFunctionParameter

  return (
      <CommandContainer>
            // το TouchableOpacity αντικείμενο προκαλεί ένα ιδιαίτερο εφέ κάθε φορά που το αντικείμενο αυτό επιλέγεται από τον χρήστη επιβεβαιώνοντας έτσι απέναντι σε αυτόν την επιλογή του
            // Κατά το πάτημα του καλείται η σταθερά fn η οποία εμπεριέχει την αντίστοιχη εντολή του Bluetooth plugin έχοντας ως όρισμα τα δεδομένα που αποστέλλονται στο όχημα μέσω του Bluetooth
        <TouchableOpacity style={itemStyle} onPress={() => fn(p)} >
          <Text style={imageTitleTextStyle}>{props.itemText}</Text>
          <Image style={imageStyle} source={props.itemImage} />
        </TouchableOpacity>
      </CommandContainer>
  );
};

const styles = {
```

```
  itemStyle: {
    justifyContent: 'space-around',
    alignItems: 'center'
  },
  imageTitleTextStyle: {
    fontSize: 18
  },
  imageStyle: {
    width: 100,
    height: 100,
    opacity: 1.0
  }
};

export default Command;
```

4.1.6.9 Συστατικό CommandContainer.js

Το τελευταίο συστατικό όπως και το "CommandListSection", δεν έχει κάποια λειτουργία εκτός από την διαμόρφωση της εμφάνισης του πλαισίου κάθε εντολής.

```
import React from 'react';
import { View } from 'react-native';

const CommandContainer = (props) => {
  const { containerStyle } = styles;

  return (
    <View style={containerStyle}>
      {props.children}
    </View>
  );
};

const styles = {
  containerStyle: {
    flexDirection: 'column',
    padding: 10,
    margin: 25,
    backgroundColor: '#AEE0F9', /* {90caf9} */
    position: 'relative'
  }
};

export default CommandContainer;
```

ΒΙΒΛΙΟΓΡΑΦΙΑ – ΑΝΑΦΟΡΕΣ

[1] Arnold, Ken. Embedded controller hardware design, Newnes, 2001. ISBN 1-878707-52-3.
[2] Microcontroller – Wikipedia: https://en.wikipedia.org/wiki/Microcontroller
[3] Μικροελεγκτής – Wikipedia: https://el.wikipedia.org/wiki/Μικροελεγκτής
[4] Arduino – Introduction: https://www.arduino.cc/en/Guide/Introduction
[5] What is an Arduino? – learn.sparkfan.com: https://learn.sparkfun.com/tutorials/what-is-an-arduino
[6] Arduino For Dummies by John Nussey ISBN-13: 978-1118446379
[7] Εισαγωγή στο Arduino – Το απόλυτο geek toy: https://deltahacker.gr/arduino-intro/
[8] Arduino Shields – learn.sparkfun.com: https://learn.sparkfun.com/tutorials/arduino-shields
[9] Infiduino Uno R3 Atmega 328P: http://www.epalsite.com/store/infiduino-uno-r3-atmega328p-atmega16u2-3-3v-5v-selectable-arduino-compatible.html
[10] Speed of Sound – Wikipedia: https://en.wikipedia.org/wiki/Speed_of_sound
[11] What is Bluetooth 2.0+EDR? – Definition: http://whatis.techtarget.com/definition/Bluetooth-20EDR
[12] DC Motor – Wikipedia: https://en.wikipedia.org/wiki/DC_motor
[13] How to use the L298N Dual H-Bridge Motor Driver: https://www.bananarobotics.com/shop/How-to-use-the-L298N-Dual-H-Bridge-Motor-Driver
[14] Graphic LCD Hookup Guide – learn.sparkfun.com: https://learn.sparkfun.com/tutorials/graphic-lcd-hookup-guide
[15] [Arduino] L298 Dual H-Bridge Motor Driver: https://billwaa.wordpress.com/2012/06/06/arduino-l298-dual-h-bridge-motor-driver/
[16] Mobile App Development: How to Decide on Hybrid vs. Native: https://www.grapecity.com/en/blogs/mobile-app-development-how-to-decide-on-hybrid-vs-native
[17] Angular vs. React - the tie breaker: https://www.airpair.com/angularjs/posts/angular-vs-react-the-tie-breaker
[18] React (JavaScript library) - Wikipedia: https://en.wikipedia.org/wiki/React_(JavaScript_library)
[19] Introducing JSX – React: https://reactjs.org/docs/introducing-jsx.html
[20] Car free icon - Flaticon http://www.flaticon.com/free-icon/car_214280

[21] Up arrow free icon – Flaticon
https://www.flaticon.com/free-icon/up-arrow_467158
[22] Down arrow free icon – Flaticon
https://www.flaticon.com/free-icon/down_467131
[23] Left arrow free icon – Flaticon
https://www.flaticon.com/free-icon/left-arrow_118744
[24] Right arrow free icon – Flaticon
https://www.flaticon.com/free-icon/right-arrow_118745
[25] Bluetooth free icon – Flaticon
https://www.flaticon.com/free-icon/bluetooth_341058
[26] Stop free icon – Flaticon
https://www.flaticon.com/free-icon/stop_567440