ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΜΕΛΕΤΗ ΚΑΙ ΚΑΤΑΣΚΕΥΗ ΦΩΝΗΤΙΚΗΣ ΑΝΑΓΝΩΡΙΣΗΣ ΜΗΝΥΜΑΤΩΝ (ΟΡΟΦΟΣ) ΓΙΑ ΦΩΝΗΤΙΚΗ ΕΝΤΟΛΗ ΟΡΟΦΟΥ ΣΕ ΑΝΕΛΚΥΣΤΗΡΕΣ ΠΡΟΣΩΠΩΝ»

Των φοιτητών
Δεμερτζή Λουκά (512027)
Καβουσανάκη Βασίλη (512040)

Επιβλέπων
Ονοματεπώνυμο: Γιακουμής Άγγελος
Βαθμίδα: Λέκτορας

Ημερομηνία 18/09/2020

Τίτλος Δ.Ε.: Μελέτη και κατασκευή φωνητικής αναγνώρισης μηνυμάτων (όροφος) για φωνητική εντολή ορόφου σε ανελκυστήρες προσώπων
Κωδικός Δ.Ε.: 15132
Ονοματεπώνυμο φοιτητών: Δεμερτζής Λουκάς, Καβουσανάκης Βασίλης
Ονοματεπώνυμο εισηγητή: Γιακουμής Άγγελος
Ημερομηνία ανάληψης Δ.Ε.:08/03/2018
Ημερομηνία περάτωσης Δ.Ε.:18/09/2020

Βεβαιώνουμε ότι είμαστε οι συγγραφείς αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχαμε για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς, είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Στις οικογένειες μας

Πρόλογος

Η εν λόγω Διπλωματική Εργασία εκπονήθηκε στην δύσκολη και πρωτόγνωρη περίοδο της καραντίνας, λόγω της έξαρσης του κορωνοϊού κατά την θερινή περίοδο του ακαδημαϊκού έτους 2019 – 2020. Η παρούσα Διπλωματική Εργασία σηματοδοτεί την ολοκλήρωση των σπουδών μας στο τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων στο Διεθνές Πανεπιστήμιο της Ελλάδος. Αποτελεί το τέλος ενός μεγάλου κύκλου που κλείνει, στον οποίον αποκτήσαμε διδάγματα που θα μας συνοδεύουν στην μετέπειτα πορεία της ζωής μας.

Η εργασία πραγματοποιήθηκε υπό την επίβλεψη του κ. Γιακουμή Άγγελου, Λέκτορα του τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων.

Η έμπνευση της Διπλωματικής εργασία και η ανάληψη του συγκεκριμένου θέματος, έρχεται από την καθημερινότητα μας, από κουβέντες που έχουμε κάνει με άτομα με κινητικά προβλήματα και όχι μόνο και κατά πόσο ένα τέτοιο project θα τους διευκόλυνε στη ζωή τους.

Περίληψη

Στην εν λόγο Διπλωματική Εργασία θα πραγματευτούμε τη μελέτη και κατασκευή φωνητικής αναγνώρισης μηνυμάτων (όροφος) για φωνητική εντολή ορόφου σε ανελκυστήρες προσώπων. Ειδικότερα, θα ασχοληθούμε με τα προτερήματα ενός τέτοιου έργου, καθώς επίσης και τις εφαρμογές που μπορεί να έχει στην καθημερινότητα μας.

Ουσιαστικά πρόκειται για έναν ανελκυστήρα, ο οποίος πέρα από τον κλασσικό χειρισμό, μέσω πλήκτρων, θα είναι πλήρως ελεγχόμενος με φωνητικές εντολές. Η κατασκευή, την οποία φτιάξαμε για τις ανάγκες της Διπλωματικής Εργασίας, βασίζεται στον προγραμματισμό του μικροεπεξεργαστή της Atmel2560, μέσω της αναπτυξιακής πλακέτας Arduino MEGA και την χρήση ενός I2C module και μιας 20X4 LCD οθόνης. Ο προγραμματισμός γίνεται μέσω του προγράμματος Arduino IDE με τη χρήση ηλεκτρονικού υπολογιστή.

Στην παρακάτω Διπλωματική Εργασία θα αναφερθούμε στη διαδικασία προγραμματισμού του Arduino και την παραμετροποίηση του, ώστε να καλύπτει τις ανάγκες μας. Επιπλέον, θα αναφέρουμε τα βασικά χαρακτηριστικά ενός ανελκυστήρα.

Abstract

The main purpose of this final-year dissertation is to examine the study and the construction of voice recognition messages (floor) for voice commands in elevators. We will deal not only with the advantages of such a project, but also with how useful it will be in our daily lives.

This project mainly concerns an elevator that apart from the normal handling with its buttons, it will also be a voice-controlled elevator, allowing passengers to fully control the elevator without pressing the buttons. The construction of this project is based on the programming of the Atmel 2560 microprocessor, through the Arduino MEGA board, the use of an I2C module and a 20x4 LCD screen. Programming is done through the Arduino IDE program in a computer.

In the following dissertation we will refer to the Arduino programming process and its configuration in order to meet our needs. Furthermore, we will mention the basic features of an elevator.

Ευχαριστίες

Ολοκληρώνοντας αυτή την εργασία και μαζί με αυτήν και τις σπουδές μας στο προπτυχιακό επίπεδο, θα θέλαμε να ευχαριστήσουμε όλους αυτούς που στάθηκαν δίπλα μας αυτά τα χρόνια και όλους αυτούς που χωρίς την πολύτιμη βοήθεια τους δεν θα ήταν δυνατή η εκπόνηση αυτής της εργασίας.

Αρχικά, θα θέλαμε να εκφράσουμε τον σεβασμό μας προς το πρόσωπο του κ. Γιακουμή Άγγελου, ο οποίος είναι και ο επιβλέπων καθηγητής της Διπλωματικής Εργασίας μας, για τις πολύτιμες συμβουλές του και την αμέριστη βοήθεια του.

Στη συνέχεια, θα θέλαμε να ευχαριστήσουμε όλους τους καθηγητές που στελεχώνουν το τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων για τις γνώσεις που μας μεταδώσαν και τις συμβουλές που θα μας συνοδεύουν στην επαγγελματική μας πορεία και όχι μόνο.

Επίσης, οφείλουμε ένα τεράστιο ευχαριστώ στον Γιώργο Καβουσανάκη απόφοιτο του τμήματος ηλεκτρονικών μηχανικών, ο οποίος με τις συμβουλές και τις γνώσεις τους μας βοήθησε οποτεδήποτε προέκυπτε κάποιο τεχνικό πρόβλημα σε μια πρωτόγνωρη και περίεργη περίοδο, όπου η πίεση του χρόνου ήταν ασφυκτική, ώστε να ολοκληρώσουμε την Διπλωματική μας εργασία.

Τέλος, θέλουμε να ευχαριστήσουμε τις οικογένειας μας και τους φίλους μας, οι οποίοι μας στήριξαν σε κάθε εμπόδιο και δυσκολία που αντιμετωπίσαμε κατά την διάρκεια των σπουδών μας, ήταν διπλά μας σε κάθε στιγμή και γέμιζαν με θετική ενέργεια την καθημερινότητά μας.

Πίνακας Περιεχομένων

Λίστα Εικόνων

Λίστα Πινάκων

Λίστα Διαγραμμάτων

1. Γενικά για τον Ανελκυστήρα

1.1 Βασική έννοια ανελκυστήρα

Στη σημερινή πραγματικότητα, ο ανελκυστήρας ή ανυψωτήρας είναι αναπόσπαστο κομμάτι της καθημερινότητας μας. Ανελκυστήρας ή ανυψωτήρας ονομάζεται κάθε εγκατάσταση που χρησιμοποιείται για την κάθετη ανύψωση βαρών, προσώπων ή πραγμάτων. Σήμερα έχει επικρατήσει ο γαλλικός όρος ασανσέρ για τον ανελκυστήρα ή ανυψωτήρα που χρησιμοποιείται στα πολυώροφα κτίρια ή εμπορικά κέντρα.

Εικόνα 1. Ανελκυστήρας

Στη γεωργία, στις μεταφορές και στις κατασκευαστικές, ένας ανελκυστήρας είναι οποιοσδήποτε τύπος μεταφορικής συσκευής που χρησιμοποιείται για την ανύψωση υλικών σε συνεχή ροή σε κάδους

ή σιλό.

Εικόνα 2. Ανελκυστήρας Σιλό ή σιτηρών .

1.2 Ιστορία του ανελκυστήρα

Η ιστορική αναδρομή του ανελκυστήρα ξεκινάει από τα αρχαιά χρόνια. Κατά το 2700 π.Χ. η κατασκευή των πυραμίδων από τους Αιγύπτιους έγινε αφορμή χρησιμοποίησης κεκλιμένων επιπέδων, υπό μορφή επιχωματώσεων, προκειμένου να ανυψωθούν και να τοποθετηθούν στη θέση τους 2.300.000 ογκόλιθοι βάρους 2,5 τόνων ο καθένας που χρησιμοποιήθηκαν για την οικοδόμησή τους. Κατά τους χρόνους εκείνους, στην Παλαιά Βασιλεία, δεν είχε ανακαλυφθεί ο τροχός και δεν υπήρχαν τροχαλίες, που θα βοηθούσαν στην ανύψωση των βαρών. Οι 100.000 ανειδίκευτοι άνδρες του εργατικού δυναμικού της εποχής, αποτέλεσαν το "εμπόδιο" στη δημιουργία αποδοτικών ανυψωτικών τεχνικών μέσων. Αργότερα το 236 π.Χ. συντελέστηκε το πρώτο αξιόλογο βήμα για την εξέλιξη του ανελκυστήρα. Ο μεγάλος Έλληνας μαθηματικός και φυσικός Αρχιμήδης ανέπτυξε την αρχή του ατέρμονος κοχλία που ακόμα και σήμερα αποτελεί θεμελιώδη αρχή στη μηχανική ανυψώσεων.

Την εποχή του Μεσαίωνα οι ανελκυστήρες χρησιμοποιήθηκαν κατά κύριο λόγο στα μοναστήρια και ερημητήρια, σχεδόν σε όλη την Ευρώπη και τοποθετήθηκαν στις κορυφές δύσβατων και σχετικά απρόσιτων βουνών. Σημαντικό ρολό στην εξέλιξη του έπαιξε ο Leonardo Da Vinci που έθεσε συγκεκριμένες βασικές αρχές λειτουργίας για την κάθε ανυψωτική μηχανή. Βέβαια, ο ίδιος ήταν επηρεασμένος, όπως αναφέρουν σημειώσεις του, από τους αρχαίους Έλληνες Αρχιμήδη, Ήρωνα και Ευκλείδη.

Από τον 18^{ο}-19^{ο} αιώνα μ.Χ. άρχισε μια μεγάλη περίοδο ερευνών. Επιπρόσθετα, η τεχνική ανύψωσης ατόμων και φορτίων γνώρισε τότε άνθιση, αλλά ταυτόχρονα χρειάστηκε και το μάρκετινγκ της εποχής για να πειστούν οι άνθρωποι για την αξιοπιστία, αλλά και την ασφαλή λειτουργία των ανελκυστήρων της εποχής εκείνης. Την ίδια περίοδο, ο Αμερικάνος μηχανικός Elisha Otis, αφού στάθηκε στο πάνω μέρος του θαλάμου ανελκυστήρα, έκοψε το συρματόσχοινο και ο θάλαμος σταμάτησε στους οδηγούς, εξαιτίας του συστήματος ασφάλειας που δημιούργησε. Μπροστά στο άφωνο από την έκπληξη ακροατήριο, ο Otis με θρίαμβο φώναξε: "Όλα ασφαλή. Όλα ασφαλή και υπό έλεγχο".

Το 1892 μ.Χ. ανακαλύφθηκε η δυνατότητα μεταβολής στροφών σε κινητήρες ξένης διέγερσης συνεχούς ρεύματος με την άμεση αυξομείωση της τάσης που εφαρμόζεται στα άκρα τους. Η εφαρμογή αυτή χρησιμοποιήθηκε σε ανελκυστήρες υψηλών κτιρίων στην Αμερική. Οι ευρωπαίοι χρησιμοποίησαν βελτιωμένες τεχνικές, προκειμένου να πετύχουν υψηλές ταχύτητες κίνησης στην ανύψωση ατόμων και φορτίων.

Οι ανελκυστήρες που χρησιμοποιούνται στις μέρες μας έχουν αλλάξει αρκετά λόγω των αυξανομένων αναγκών του κοινού. Πλέον, δίνεται μεγάλη βαρύτητα στην εξοικονόμηση ηλεκτρικής ενέργειας, αλλά και στην οικολογική συμπεριφορά των ανελκυστήρων. Οι βασικές αρχές έχουν αλλάξει ελάχιστα ή και καθόλου σε σχέση με τους ανελκυστήρες των παλαιότερων εποχών. Αυτό που έχει διαφοροποιηθεί είναι η τεχνολογία των ημέρων μας και τα τεχνολογικά μέσα κατασκευής.

1.3 Κύρια χαρακτηριστικά του ανελκυστήρα

Τα κυριότερα μέρη, από τα οποία αποτελείται η εγκατάσταση ενός ανελκυστήρα είναι τα ακόλουθα:

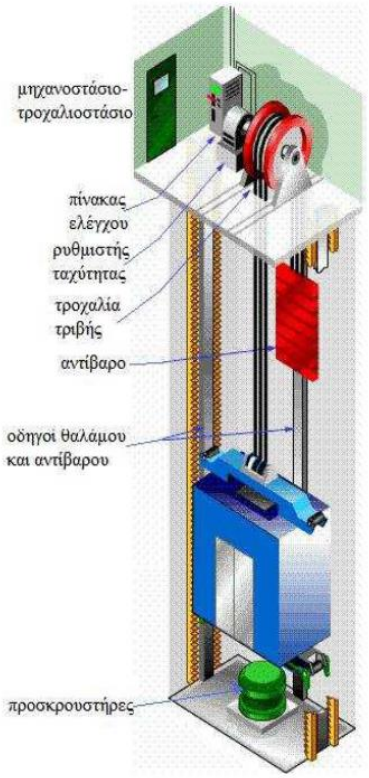
1. **Φρεάτιο**. Είναι ο χώρος που κινούνται τα μέρη του ανελκυστήρα που μετέχουν σε ευθύγραμμη κίνηση και ειδικότερα ο θάλαμος και το αντίβαρο, αν υπάρχει.
2. **Θάλαμος**. Ο θάλαμος αποτελείται από την καμπίνα και το πλαίσιο ανάρτησης και ολισθαίνει πάνω στις κατευθυντήριες ράβδους.
3. **Πόρτες Ανελκυστήρων.** Οι πόρτες των ανελκυστήρων διαιρούνται σε τρεις κατηγορίες.
    a. Χειροκίνητες: Οι πόρτες αυτές ανοίγουν και κλείνουν με ώθηση μόνο όταν ο θαλαμίσκος βρίσκεται πίσω από αυτές. Οι πόρτες του φρεατίου κλείνουν με ειδικό μάνδαλο/ασφάλεια, το οποίο δεν επιτρέπει τη λειτουργία του ανελκυστήρα αν ο πύρος μανδάλωσης δεν μπει ακριβώς μέσα στο φύλλο της πόρτας. Ταυτόχρονα το μάνδαλο αποτρέπει το άνοιγμα της πόρτας όταν ο θάλαμος βρίσκεται σε κίνηση.
    b. Ημιαυτόματες: Με τον όρο ''ημιαυτόματη'' πόρτα εννοούμε ότι η πόρτα κλείνει μόνη της και ανοίγει ύστερα από πίεση με το χέρι. Στην περίπτωση των ημιαυτόματων πορτών ο θάλαμος συνήθως δεν έχει δικές του πόρτες. Σε αυτή την κατηγορία υπάρχουν επίσης μανδαλώσεις όπως στην προηγούμενη.
    c. Αυτόματες: Μια πόρτα ανελκυστήρα ονομάζεται αυτόματη όταν ανοίγει και κλείνει μόνη της, χωρίς καμία ανθρώπινη επέμβαση από έξω ή από μέσα.

    Στις μέρες μας οι πιο συνηθισμένες πόρτες είναι οι ημιαυτόματες, κυρίως για πολυκατοικίες. Χειροκίνητες πόρτες συναντάμε σε πολύ παλιές πολυκατοικίες, ενώ αυτόματες πόρτες χρησιμοποιούνται κυρίως σε μεγάλα δημοσιά ή ιδιωτικά κτίρια, όπως για παράδειγμα νοσοκομεία, εμπορικά κέντρα κτλ.

4. **Αντίβαρο.** Ο ρόλος του αντίβαρου είναι να βοηθάει τον κινητήρα στην ανύψωση του θαλάμου, συνεπώς του ωφέλιμου φορτίου. Το βάρος του αντίβαρου πρέπει να είναι ίσο προς το βάρος του θαλαμίσκου συν το μισό του ωφέλιμου φορτίου.
5. **Οδηγοί-Ευθυντήριοι ράβδοι.** Οι οδηγοί χρησιμοποιούνται υποχρεωτικά σε κάθε ανελκυστήρα και χρησιμεύουν στην καθοδήγηση του πλαισίου του θαλάμου και του αντίβαρου
6. **Συρματόσχοινα ανάρτησης.** Χρησιμοποιούνται για το ανέβασμα και το κατέβασμα του θαλάμου και του αντίβαρο.
7. **Τροχαλία Τριβής.** Κατασκευάζεται από δύο επιμέρους τροχαλίες τοποθετημένες σε κοινό χαλύβδινο άξονα ισχυρής κατασκευής μέσω ενός ζεύγους ρουλεμάν η κάθε μία, που εδράζεται σε ανεξάρτητα αυτολιπαινόμενα έδρανα. Ο άξονας στηρίζεται στα δύο ακραία σημεία του πάνω σε μία σιδηροκατασκευή τοποθετημένη στην άνω απόληξη του εμβόλου. Οι τροχαλίες αυτές είναι κατασκευασμένες με μεγάλη ακρίβεια, με αυλάκια υποδοχής ημικυκλικού σχήματος για να αποφεύγεται η ανισοταχής κίνηση των συρματόσχοινων, η ολίσθησή και η γρήγορη φθορά τους. Σε παλαιότερους μηχανισμούς αντί για τροχαλία τριβής υπήρχε τύμπανο. Στο τύμπανο οι αυλακώσεις είχαν σχήμα έλικας και το συρματόσχοινο ή η αλυσίδα παρασυρόταν με οποιοδήποτε άλλο μέσο εκτός από την τριβή.
8. **Κινητήρας.** Για την κίνηση του θαλάμου χρησιμοποιείται συνήθως ένας ασύγχρονος ηλεκτροκινητήρας. Ο κινητήρας είναι συνήθως τριφασικός με τάση 380V/50 Hz. Συνήθως

είναι διπολικός ή τετραπολικός. Οι προδιαγραφές του πρέπει να είναι τέτοιες, ώστε η ροπή εκκινήσεως να είναι περίπου διπλάσια της ονομαστικής.

9. **Ηλεκτρομαγνητική Πέδη (Φρένο).** Ο ανελκυστήρας πρέπει να είναι εφοδιασμένος με σύστημα πέδησης που να ενεργοποιείται αυτόματα. Το φρένο χρησιμοποιείται για την ακινητοποίηση του ανελκυστήρα. Αποτελείται από ένα ηλεκτρομαγνήτη, δύο μπράτσα επενδυμένα εσωτερικά με φερμουΐτ(τακάκι) και ένα σύστημα μοχλών.
10. **Προσκρουστήρες.** Πρέπει να τοποθετούνται στο κατώτερο όριο της διαδρομής του θαλάμου και του αντίβαρου. Το σημείο λειτουργίας του προσκρουστήρα, κάτω από την προβολή του θαλάμου, πρέπει να χαρακτηρίζεται από ένα εμπόδιο με ύψος τέτοιο ώστε να ικανοποιείται ο σχετικός κανονισμός.



Εικόνα 3. Ολοκληρωμένη απεικόνιση εγκατάστασης ανελκυστήρα έλξεως

2. Εισαγωγή στο Arduino

2.1 Τι είναι ο Arduino

Ο Arduino είναι μια υπολογιστική πλατφόρμα βασισμένη σε μια απλή μητρική πλακέτα ανοικτού κώδικα, με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++) μέσω Η/Υ μέσα από ένα απλό περιβάλλον ανάπτυξης.

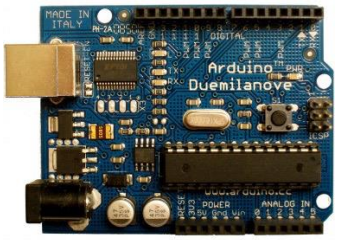
Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, Pure Data, SuperCollider. Οι περισσότερες εκδόσεις του Arduino μπορούν να αγοραστούν προ-συναρμολογημένες· το διάγραμμα και πληροφορίες για το υλικό είναι ελεύθερα διαθέσιμα για αυτούς που θέλουν να συναρμολογήσουν το Arduino μόνοι τους.

2.2 Ιστορία του Arduino

Το 2005, ένα σχέδιο ξεκίνησε προκειμένου να φτιαχτεί μία συσκευή για τον έλεγχο προγραμμάτων διαδραστικών σχεδίων από μαθητές, η οποία θα ήταν πιο φθηνή από άλλα πρωτότυπα συστήματα διαθέσιμα εκείνη την περίοδο. Οι ιδρυτές Massimo Banzi και David Cueartielles ονόμασαν το σχέδιο από τον Arduin της Ivrea, όπου με δυο ακόμη φοιτητές που επιλέχτηκαν να γράψουν το λογισμικό για τη συσκευή, ξεκίνησαν να παράγουν πλακέτες σε ένα μικρό εργοστάσιο στην Ιβρέα, κωμόπολη της επαρχίας Τορίνο στην περιοχή Πεδεμόντιο της βορειοδυτικής Ιταλίας, την ίδια περιοχή στην οποία στεγαζόταν η εταιρία υπολογιστών Olivetti. Ο Ηλεκτρολόγος Μηχανικός Gianluca Martino, κλήθηκε να κάνει μια αρχική παρτίδα των 200 μικροελεγκτών.

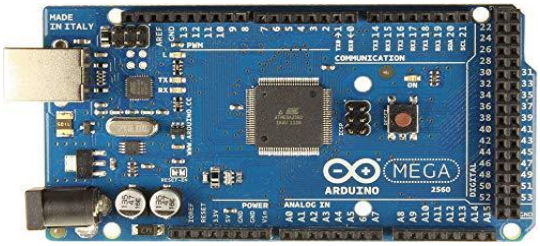
Το πρώτο Arduino που φτιάχτηκε ονομάστηκε “Serial Arduino” και περιλάμβανε ένα ATmega8 με άμεση σύνδεση RS-232 με τον μικροελεγκτή και όλα τα συστατικά του. Στη συνέχεια σχεδιάστηκε η έκδοση 2.0 και μια μονόπλευρη εκδοχή σαφέστερη για τους χομπίστες. Οι εκδόσεις που ακολούθησαν ήταν όλες FTDI USB μετατροπέα. Μετά το USB v1.0 και v2.0 κυκλοφόρησε το Arduino Extreme, το οποίο αύξησε την ποσότητα των επιφανειακών εξαρτημάτων. Το Arduino Nuova Generazione μεταβαίνει σε έναν απλούστερο μετατροπέα USB και μετατρέπεται από το ATmega8 σε ATmega168. Οι βελτιώσεις συνεχίστηκαν με το Diecimila και έτσι:

- Τον Οκτώβριο του 2008 ανακοινώθηκε το Duemilanove, όπου αρχικά βασίστηκε στο Atmel Atmega 168, αλλά μετά αντικαταστάθηκε με το ATmega 328



Εικόνα 4. Arduino Duemilavone

- Τον Μάρτιο του 2009 ανακοινώθηκε το Arduino Mega, το οποίο είναι βασισμένο στο Atmel ATmega 1280

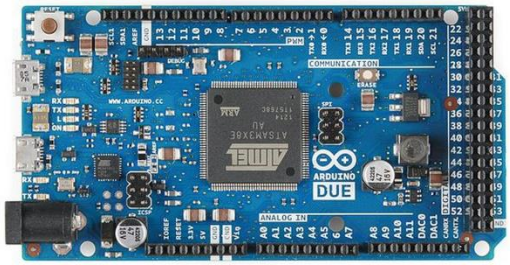


Εικόνα 5. Arduino Mega

- Από τον Μάιο του 2011 πάνω από 300,000 Arduino ήταν σε χρήση σε όλο τον κόσμο
- Τον Ιούλιο του 2012 ανακοινώθηκε το Arduino Leonardo, το οποίο είναι βασισμένο στο Atmel ATmega32u4.

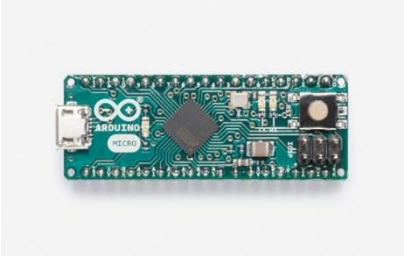
Εικόνα 6. Arduino Leonardo

- Τον Οκτώβριο του 2012 ανακοινώθηκε το Arduino Due, το οποίο είναι βασισμένο στο Atmel SAM3X8E, που είχε πυρήνα ARM Cortex-M3.



Εικόνα 7. Arduino Due

- Τον Νοέμβριο του 2012 ανακοινώθηκε το Arduino Micro, το οποίο είναι βασισμένο στο Atmel ATmega32u4.



Εικόνα 8. Arduino Micro

- Τον Μάιο του 2013 ανακοινώθηκε το Arduino Robot, το οποίο είναι βασισμένο στο Atmel ATmega32u4 και ήταν το πρώτο επίσημο Arduino με ρόδες.

Εικόνα 9. Arduino Robot

- Τον Μάιο του 2013 ανακοινώθηκε το Arduino Yun που είναι βασισμένο στο arATmega32u4 και στο Atheros AR9331 και ήταν το πρώτο προϊόν WiFi που συνδύαζε το Arduino με το Linux.

Εικόνα 10. Arduino Yun

2.3 Πλεονεκτήματα του Arduino

Στο εμπόριο έχουν χρησιμοποιηθεί μέχρι τώρα 20 εκδόσεις του Arduino. Βέβαια, υπάρχουν και άλλες λύσεις μικροεπεξεργαστών, όπως είναι ο Basic Stamp της Parallax, ο BX-24 της Netmedia, το Handyboard του MIT. Αυτό που κάνει το Arduino όμως να ξεχωρίζει είναι η απλότητα της χρήσης του και η ευκολία προγραμματισμού του. Εκτός από αυτά τα δύο πλεονεκτήματα υπάρχουν και άλλα που κάνουν το Arduino να ξεχωρίζει και να προτιμάται από πολλούς χρήστες, οι οποίοι είναι αρχάριοι με τα ηλεκτρονικά και τον προγραμματισμό.

Τα βασικά πλεονεκτήματα είναι:

- **Οικονομία**
  Το Arduino είναι εξαιρετικά φθηνό σε σχέση με τις άλλες πλακέτες μικροεπεξεργαστών. Επίσης, είναι εύκολο κάποιος πιο έμπειρος, σύμφωνα με τα σχηματικά που βρίσκονται στο ίντερνετ, να την κατασκευάσει μόνος του με ακόμα λιγότερα έξοδα. Στο ίντερνετ μπορεί κάποιος να βρει έτοιμη μια πλακέτα από 20€.

- **Συνδεσιμότητα**
  Η πλακέτα του Arduino μπορεί να συνδεθεί και να προγραμματιστεί στα περισσότερα λειτουργικά συστήματα. Μπορεί κάποιος να το προγραμματίσει σε περιβάλλον Windows, Macintosh OSX ή ακόμα και σε Linux. Αντιθέτως, οι αντίστοιχες λύσεις μικροεπεξεργαστών περιορίζονται συνήθως στα Windows**.**

- **Ευκολία προγραμματισμού**
  Το περιβάλλον προγραμματισμού ενός Arduino ενδείκνυται για αρχάριους, αλλά είναι ταυτόχρονα και ευέλικτο και για πιο προχωρημένους χρήστες.

- **Ανοιχτό λογισμικό και υλικό**
  Το υλικό και το λογισμικό του Arduino είναι ανοιχτό και ελεύθερο, με αποτέλεσμα καθημερινά χιλιάδες χρήστες να αναπτύσσουν βιβλιοθήκες για την υποστήριξη της πλατφόρμας, βοηθώντας στην εξέλιξή της. Ο Arduino βασίζεται στους μικροελεγκτές της Atmel ATmega8 και ATmega168. Τα σχηματικά για τα αναπτυξιακά είναι κάτω από την άδεια της Creative Commons, επιτρέποντας σε έμπειρους σχεδιαστές να κατασκευάσουν το δικό τους αναπτυξιακό, εξελίσσοντας το ήδη υπάρχον χωρίς να έχουν νομικά προβλήματα.

2.4 Arduino Mega

Για την υλοποίηση της πτυχιακής χρησιμοποιήθηκε το Arduino MEGA 2560 (Εικόνα 1). Το Arduino Mega 2560 είναι ένα ανάπτυγμα μικροελεγκτή βασισμένο στον επεξεργαστή ATMega2560.

Έχει 54 ψηφιακά pins εισόδου/εξόδου (σε 15 από τα οποία μπορούμε να έχουμε ψηφιακούς παλμούς μεταβλητού πλάτους), 16 αναλογικές εισόδους, έναν κρυσταλλικό ταλαντωτή στα 16MHz, μια σύνδεση USB, ένα βύσμα τροφοδοσίας, ένα βύσμα ICSP και ένα κουμπί επανεκκίνησης.

Είναι ένα ολοκληρωμένο σύστημα μικροεπεξεργαστή που λειτουργεί είτε αν το συνδέσουμε στην θύρα USB του υπολογιστή, είτε με τη χρήση μετασχηματιστή, είτε με τη χρήση κάποιας μπαταρίας. Το MEGA2560 είναι συμβατό με τα περισσότερα περιφερειακά που είναι σχεδιασμένα για το Arduino Duemilanove or Diecimila. Το MEGA2560 αποτελεί μια αναβαθμισμένη έκδοση του Arduino MEGA, το οποίο έχει αποσυρθεί. Η διαφορά του MEGA 2560 από τις υπόλοιπες πλακέτες είναι ότι δεν χρησιμοποιεί FTDI USBtoSerial τσιπ. Αντί αυτού έχει το ATmega16U2, το οποίο λειτουργεί ως μετατροπέας USBtoSerial.

Εικόνα 11. Arduino Mega 2560

2.4.1 Τεχνικά χαρακτηριστικά

Μικροεπεξεργαστής	Atmega2560
Τάση λειτουργίας	5V
Τάση εισόδου (προτείνεται)	7-12V
Τάση εισόδου (max)	6-20V
Ψηφιακά pins I/O	54
Pins αναλογικής εισόδου	16
Συνεχές ρεύμα ανά I/O pin	40mA
Συνεχές ρεύμα 3.3V pin	50mA
Μνήμη Flash	256 KB (8KB χρησιμοποιούνται από το πρόγραμμα εκκίνησης)
SRAM	8KB
EEPROM	4KB
Ταχύτητα ρολογιού	16MHz

Πίνακας 1. Τεχνικά χαρακτηριστικά Arduino Mega

Αναλυτικότερα:

- **Τροφοδοσία**

  Το Arduino MEGA2560 μπορεί να τροφοδοτηθεί μέσω σύνδεσης USB ή με μία εξωτερική παροχή ηλεκτρικού ρεύματος. Η πηγή ενέργειας επιλέγεται αυτόματα.

  Η εξωτερική παροχή (εκτός του USB) μπορεί να προέλθει είτε από έναν προσαρμογέα ρεύματος εναλλασσόμενου σε συνεχές, είτε από μία μπαταρία. Ο προσαρμογέας μπορεί να συνδεθεί με ένα βύσμα των 2.1mm στο βύσμα της πλακέτας. Τα καλώδια από μία μπαταρία μπορούν να συνδεθούν στις εισόδους Gnd και Vin των εισόδων τροφοδότησης. Η πλακέτα μπορεί να λειτουργήσει και με μία εξωτερική παροχή από 6 έως 20 volts. Αν παρέχεται τάση μικρότερη των 7V, ωστόσο, η είσοδος των 5V μπορεί να προμηθεύει μικρότερη τάση των 5V και η πλακέτα να είναι ασταθής. Αν παρέχεται τάση μεγαλύτερη των 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να βλάψει την πλακέτα. Το προτεινόμενο φάσμα τροφοδότησης είναι 7 με 12 volts.

  Οι ακροδέκτες τροφοδοσίας είναι οι ακόλουθοι:

  - **VIN**. Η τάση εισόδου στην πλακέτα Arduino όταν χρησιμοποιεί μια εξωτερική πηγή. Είναι δυνατό να παρέχεται τάση μέσω αυτής της εισόδου ή αν παρέχεται τάση μέσω του βύσματος, να έχουμε πρόσβαση σε αυτή μέσω αυτής της εισόδου.

  - **5V**. Η ρυθμισμένη παροχή ηλεκτρικού ρεύματος χρησιμοποιείται για να τροφοδοτήσει τον μικροεπεξεργαστή και άλλα εξαρτήματα στην πλακέτα. Αυτό μπορεί να προέλθει είτε από το VIN μέσω ενός ρυθμιστή επάνω στην πλακέτα, είτε να παρέχεται από USB ή έναν άλλο ρυθμιστή τάσης των 5V.

- **3V3**. Μια παροχή των 3.3 volt παράγεται από έναν ρυθμιστή, ο οποίος βρίσκεται επάνω στην πλακέτα. Το μέγιστο ρεύμα που μπορεί να μεταδοθεί είναι 50mA.

- **GND**. Είσοδοι γείωσης.

- **IOREF**. Αυτό το pin παρέχει το σημείο αναφοράς της τάσης λειτουργίας του μικροελεγκτή. Ένα σωστό ρυθμιζόμενο shield μπορεί να διαβάσει την τάση από το IOREF και να επιλέξει την τάση λειτουργίας στα 5V ή 3,3V.

- **Μνήμη**

Το ATmega2560 έχει 256 KB μνήμη flash για την αποθήκευση του κώδικα (εκ των οποίων 8KB χρησιμοποιούνται για το πρόγραμμα εκκίνησης), 8KB SRAM και 4KB EEPROM.

- **Είσοδοι και έξοδοι**

Κάθε ένα από τα 14 ψηφιακά pins στο Mega μπορούν να χρησιμοποιηθούν είτε ως είσοδοι, είτε ως έξοδοι, χρησιμοποιώντας τις συναρτήσεις pinMode(), digitalWrite() και digitalRead() στο sketch του προγράμματος. Λειτουργούν στα 5V. Κάθε pin μπορεί να παρέχει ή να λαμβάνει 20ma, ενώ το μέγιστο που μπορεί είναι 40mA. Έχει μία εσωτερική αντίσταση (προεπιλεγμένα αποσυνδεμένη) των 20-50kOhms.

Επιπροσθέτως, κάποια pin έχουν πιο ειδικές λειτουργίες:

- **Σειριακές:** 0 (RX) και 1 (TX) Σειριακή 1: 19(RX) και 18(TX), Σειριακή 2: 17(RX) και 16(TX), Σειριακή 3: 15(RX) και 14(TX). Χρησιμοποιούνται για την παραλαβή (RX) και τη διαβίβαση (TX) των σειριακών στοιχείων TTL. Τα pins 0 και 1 συνδέονται με τις αντίστοιχες εισόδους του τμηματικού τσιπ ATmega16U2 USB-to-TTL.

- **Εξωτερικοί διακόπτες**: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), και 21 (interrupt 2). Αυτά τα pins μπορούν να διαμορφωθούν για να προκαλέσουν διακοπή σε μια χαμηλή τιμή, μία αυξημένη ή μία πτωτική τιμή, ή μία αλλαγή στην τιμή.

- **PWM**: 2 μέχρι 13 και 44 μέχρι 46. Παρέχουν 8-bit PWM έξοδο μέσω της εντολής analogWrite().

- **SPI**: 53(SS), 51(MOSI), 50(MISO), 52(SCK). Αυτά τα pins υποστηρίζουν επικοινωνία SPI, κάνοντας χρήση της βιβλιοθήκης SPI.

- **LED 13**. Υπάρχει ένα ενσωματωμένο LED συνδεδεμένο στο ψηφιακό pin 13. Όταν το pin ορίζεται ως HIGH, το LED ανάβει, όταν το pin ορίζεται ως LOW το LED σβήνει.

- **TWI**. 20(SDA) και 21(SCL). Υποστηρίζουν την επικοινωνία TWI με χρήση της βιβλιοθήκης Wire.

Το Mega2560 έχει 16 αναλογικές εισόδους, κάθε μια από τις οποίες παρέχει 10 bits.

Υπάρχουν επίσης ακόμη 2 pins στην πλακέτα

- **AREF**: Αναφορά τάσης για τις αναλογικές εισόδους, χρησιμοποιώντας την εντολή analogReference().

- **Reset**: Η επαναφορά της γραμμής αυτής σε LOW τιμή, επαναρυθμίζει τον μικροεπεξεργαστή. Συνήθως χρησιμοποιείται για να προσθέσουμε ένα reset κουμπί στις ασπίδες που μπλοκάρουν αυτό της πλατφόρμας.

- **Προειδοποιήσεις**

Arduino Mega έχει πολλές επιλογές για την επικοινωνία με έναν υπολογιστή, έναν άλλο Arduino, ή άλλους μικροεπεξεργαστές. Ο ATmega2560 παρέχει UART TLL (5V) σειριακή επικοινωνία. Ένα ATmega16U2 μέσω usb παρέχει μια εικονική είσοδο COM του λογισμικού στον υπολογιστή.

Το λογισμικό του Arduino περιλαμβάνει μία σειριακή οθόνη, που επιτρέπει στα απλά δεδομένα κειμένου να αποσταλούν προς και από την πλατφόρμα Arduino. Το λογισμικό Arduino περιλαμβάνει μία σειριακή οθόνη, που επιτρέπει στα απλά δεδομένα κειμένου να αποσταλούν προς και από την πλατφόρμα Arduino.

Τα LEDs RX και TX της πλακέτας θα αναβοσβήσουν όταν μεταδίδονται δεδομένα μέσω του chip ATmega8U2/ATmega16U2 και της σύνδεσης USB του υπολογιστή (αλλά όχι για τη σειριακή επικοινωνία στους ακροδέκτες 0 και 1). Η βιβλιοθήκη Software Serial επιτρέπει τη σειριακή επικοινωνία με οποιαδήποτε ψηφιακή είσοδο του Duemilanove.

Επίσης, ο Mega 2560 υποστηρίζει τις επικοινωνίες TWI και SPI. Το λογισμικό Arduino περιλαμβάνει μια βιβλιοθήκη Wire για να απλοποιήσει τη χρήση της TWI επικοινωνίας.

Ο μικροεπεξεργαστής Mega 2560 μπορεί να προγραμματιστεί με το Arduino Software (IDE). Ο ATmega2560 είναι ήδη προγραμματισμένος με ένα πρόγραμμα εκκίνησης που επιτρέπει τη μεταφόρτωση νέου κώδικα σε αυτό χωρίς τη χρήση ενός εξωτερικού προγραμματιστή υλικού. Επικοινωνεί χρησιμοποιώντας το πρωτόκολλο STK500.

Επιπλέον, μπορεί να παρακαμφθεί το πρόγραμμα εκκίνησης και να προγραμματίσουμε τον μικροεπεξεργαστή μέσω του ICSP

2.5 Arduino Shields

Τα shields είναι ολοκληρωμένες πλακέτες που είναι σχεδιασμένες, ώστε να κουμπώνουν πάνω στο Arduino, προεκτείνοντας τη λειτουργικότητά του (Εικόνα 11). Είναι η hardware αντίστοιχη έννοια των plugin, addon και extension που υπάρχουν στο Software.

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

- **Ethernet shield**

  Δίνει στο Arduino τη δυνατότητα να δικτυωθεί σε ένα LAN ή στο Internet μέσω ενός τυπικού καλωδίου Ethernet.

- **WIFI shield**

  Όμοιο με το Ethernet shield, χωρίς φυσικά το καλώδιο.

- **Διάφορα shield 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).

- **Διάφορα Motor Shields**

  Σας επιτρέπουν να οδηγήσετε εύκολα μοτέρ διαφόρων τύπων (απλά DC, servo, stepper κλπ.) από το Arduino.

- **Porto shield**

  Μια προσχεδιασμένη πλακέτα προτυποποίησης, συμβατή στις διαστάσεις του Arduino και χωρίς εξαρτήματα για να φτιάξει ο χρήστης το δικό του shield**.**

- **Voice recognition shield**

Έχει τη δυνατότητα αναγνώρισης ομιλίας και φωνής σε σχεδόν οποιαδήποτε εφαρμογή.

Εικόνα 12. Arduino Shields

Τα shields είναι σχεδιασμένα, ώστε αφού κουμπωθούν πάνω στο Arduino, να προωθούν τις υποδοχές του, με σκοπό να μπορείτε να συνδέσετε επιπλέον τα δικά σας εξαρτήματα ή να κουμπώσετε και επόμενο shield. Φυσικά, το κάθε shield χρησιμοποιεί ορισμένους από τους πόρους συνδεσιμότητας του Arduino και έτσι δεν μπορείτε να συνδέσετε απεριόριστα shield.

Μάλιστα κάποια shield μπορεί να μην είναι συμβατά μεταξύ τους γιατί χρησιμοποιούν τα ίδια pin του Arduino για επικοινωνία με αυτό. Επίσης, επειδή κάποια shield δεν προωθούν τις συνδέσεις του Arduino (όπως π.χ. οι οθόνες οι οποίες δεν έχουν νόημα αν τις καλύψετε από πάνω με ένα επόμενο shield), υπάρχουν ειδικά extender shield που κουμπώνουν στο Arduino και δίνουν τη δυνατότητα σε δύο άλλα shield να κουμπώσουν πάνω τους, λειτουργώντας σαν πολύμπριζα.

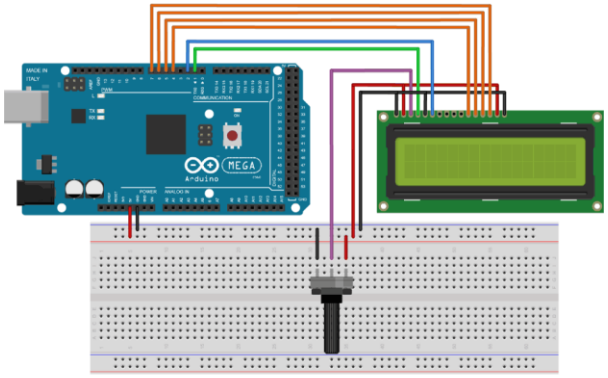
Όπως και για το ίδιο το Arduino, το βασικό πλεονέκτημα των shield δεν είναι τόσο το προφανές πλεονέκτημα του έτοιμου hardware, όσο ότι συνοδεύονται συνήθως από έτοιμες βιβλιοθήκες που σας επιτρέπουν να προγραμματίζετε τα sketch σας σε High level. Έτσι, λόγου χάρη, δεν χρειάζεται να διαβάζετε datasheet ή να γίνετε ηλεκτρονικός για να συνδέσετε και να λειτουργήσετε ένα GPS module πάνω στο Arduino. Απλά συνδέετε το shield, εγκαθιστάτε τη βιβλιοθήκη που το συνοδεύει και χρησιμοποιείτε μια έτοιμη συνάρτηση του στυλ getLocation- για να πάρετε το γεωγραφικό στίγμα και να το επεξεργαστείτε περαιτέρω στο sketch σας.

Τα shield σας λύνουν τα χέρια όταν θέλετε να δημιουργήσετε εύκολα ένα πραγματικά πρακτικό Project. Αυτός είναι και ο λόγος που δεν συνιστάται η αγορά κάποιας έκδοσης του Arduino που δεν είναι 100% συμβατή με τα shield.

2.6 Οθόνη LCD

Τα αναπτυξιακά κιτ έχουν σχεδιαστεί για να λειτουργούν αυτόνομα (χωρίς σύνδεση με τον υπολογιστή). Στις περισσότερες εφαρμογές υπάρχει η ανάγκη για απεικόνιση πληροφοριών που σχετίζονται με μετρήσεις, μηνύματα ή και την κατάσταση λειτουργίας της πλατφόρμας. Αυτό σημαίνει ότι θα πρέπει να υπάρχει η δυνατότητα προσάρτησης ειδικής οθόνης μικρού μεγέθους, ώστε να μπορεί να ενσωματωθεί στο τελικό σύστημα. Οι LCD οθόνες αποτελούν την πιο απλή και φτηνή λύση για την απεικόνιση πληροφοριών. Για το λόγο αυτό, θα χρησιμοποιήσουμε μια οθόνη LCD για την απεικόνιση κάποιων βασικών πληροφοριών.

Η απεικόνιση των πληροφοριών στην LCD οθόνη γίνεται με μερικές γραμμές κώδικα στην περίπτωση του Arduino.



Εικόνα 13. LCD συνδεδεμένη στο Arduino Mega

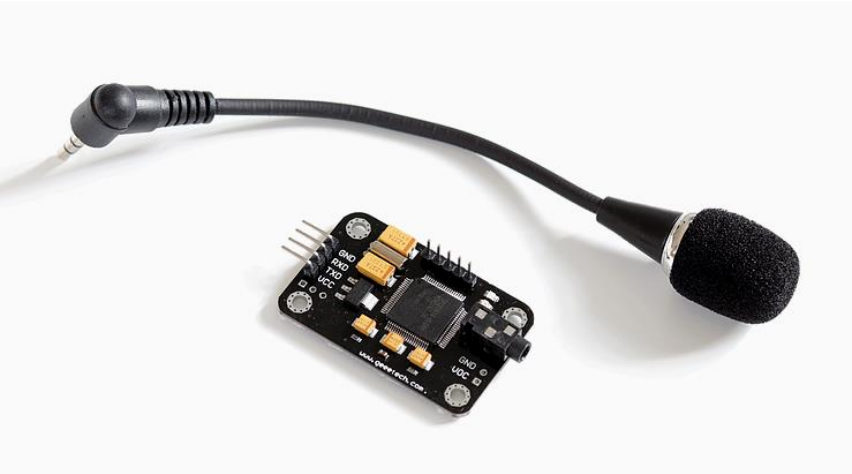
2.7 Module I2C

Το I2C module βασίζεται στο ολοκληρωμένο PCF8574. Η δουλειά του module I2C είναι να λειτουργεί σαν δίαυλος επικοινωνίας μεταξύ συσκευών που έχουν διαθέσιμο ένα ή και παραπάνω master και slave. Το I2C μπορεί να έχει μέχρι 128 συσκευές συνδεδεμένες ταυτόχρονα με 7-bit address ή 1024 συσκευές όταν χρησιμοποιείται 10-bit address. Έτσι, η κάθε συσκευή έχει την δικιά της address (διεύθυνση) και με αυτόν τον τρόπο η master (κύρια) συσκευή διαλέγει με ποια συσκευή θα επικοινωνήσει. Αυτό πραγματοποιείται συνδέοντας 2 PIN με τις άλλες συσκευές το SCL που συγχρονίζει τα δεδομένα μεταξύ των συσκευών και το SDA που μεταφέρει τα δεδομένα μεταξύ των συσκευών.

Εικόνα 14. I2C module

2.8 Voice recognition shield

Το συγκεκριμένο shield βοηθάει τον χρήστη να μπορεί να ελέγχει ηλεκτρονικές συσκευές μέσω φωνητικών εντολών. Για το λόγο αυτό, θα χρησιμοποιήσουμε ένα voice recognition shield μαζί με μικρόφωνο για τον φωνητικό έλεγχο του ανελκυστήρα. Το shield έχει τη δυνατότητα να μπορεί αποθηκεύσει έως 15 κομμάτια φωνητικής εντολής.



Εικόνα 15. Voice recognition shield

2.9 USB to TTL

Το USB to TTL module θα μας βοηθήσει στον προγραμματισμό του voice recognition shield. Το module συνδέει το voice recognition shield με τον ηλεκτρονικό υπολογιστή, ώστε να προγραμματίσει και να αποθηκεύσει τις εντολές στο voice recognition shield μέσω του προγράμματος accessport.

Εικόνα 16. USB to TTL module

3. Υλοποίηση κατασκευής και κώδικας

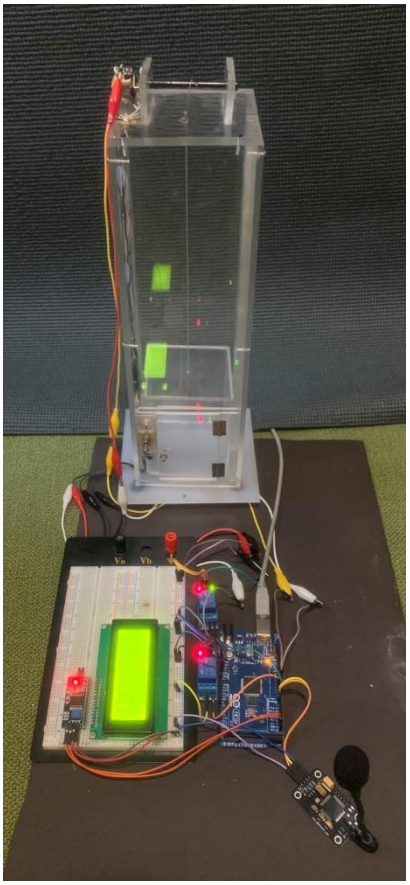
3.1 Υλικά και περιφερειακές συσκευές

Για τις ανάγκες τις Διπλωματικής εργασίας κατασκευάσαμε ανελκυστήρα ενός ορόφου ύψους 35εκ. εξ ολοκλήρου από πλεξιγκλάς. Η συγκεκριμένη κατασκευή εξομοιώνει σε μεγάλο βαθμό έναν πραγματικό ανελκυστήρα βασιζομένη στα κυριότερα χαρακτηριστικά του, φρεάτιο, θάλαμος, πόρτες, Οδηγοί-Ευθυντήριοι ράβδοι, κινητήρας κτλ.

Όπως αναφερθήκαμε και παραπάνω η εργασία στηρίζεται στον Arduino και πιο συγκεκριμένα στο Arduino Mega. Ο Arduino είναι αυτός που προγραμματίστηκε, έτσι ώστε να μπορεί ο χρήστης να επικοινωνεί μέσω φωνητικών εντολών με τον ανελκυστήρα και ο ανελκυστήρας να δουλεύει σύμφωνα με τις προδιαγραφές της.

Για την ολοκλήρωση της Διπλωματικής χρησιμοποιήθηκε μια σειρά από υλικά, τα οποία αναφέρονται παρακάτω:

- Ο ανελκυστήρας, ο οποίος κατασκευάστηκε από πλεξιγκλάς για να μπορούμε να εφαρμόσουμε πάνω του τις φωνητικές εντολές.
- Μια οθόνη LCD 20x4 συνδεδεμένο με το I2C module, όπου εμφανίζονται οι βασικές πληροφορίες για την κατάσταση, στην οποία βρίσκεται ο ανελκυστήρας.
- Voice recognition shield, όπου θα μας βοηθήσει ώστε ο χρήστης να ελέγχει με φωνητικές εντολές τον ανελκυστήρα.
- USB to TTL, με το οποίο μέσω υπολογιστή προγραμματίζουμε τις φωνητικές εντολές του voice recognition shield



Εικόνα 17. Κατασκευή

3.2 Εύρεση Address της LCD Οθόνης

Η σχεδίαση του κυκλώματος είναι απλή, καθώς χρειάζονται μόνο τα εξαρτήματα που αναφέρθηκαν στο κεφάλαιο 3.1. Για να λειτουργήσει η οθόνη και να μπορούμε να την λειτουργήσουμε θα χρειαστεί να βρούμε την Address της μέσα από τον παρακάτω κώδικα και με τη βοήθεια του I2C.

```
#include <Wire.h>

byte error, address;

int nDevices;

void setup()

{

 Wire.begin();

 Serial.begin(9600);

 Serial.println("\nI2C Scanner");

}

void loop()

{

 Serial.println("Scanning...");

 nDevices = 0;

 for(address = 1; address < 127; address++ )

 {

  Wire.beginTransmission(address);

  error = Wire.endTransmission();
```

```
  if (error == 0)
  {
    Serial.print("I2C device found at address 0x");
    if (address<16)
      Serial.print("0");
    Serial.print(address,HEX);
    Serial.println("  !");
    nDevices++;
  }
  else if (error==4)
  {
    Serial.print("Unknown error at address 0x");
    if (address<16)
      Serial.print("0");
    Serial.println(address,HEX);
  }
 }
 if (nDevices == 0)
  Serial.println("No I2C devices found\n");
 else
  Serial.println("done\n");
 delay(5000);
}
```

Παρακάτω ο κώδικας αναλυτικά:

```
#include <Wire.h>

byte error, address;

int nDevices;
```

Γίνεται η δήλωση της βιβλιοθήκης και των μεταβλητών που θα χρησιμοποιήσουμε στον κώδικα.

```
void setup()

{

 Wire.begin();

 Serial.begin(9600);

 Serial.println("\nI2C Scanner");

}
```

Με την εντολή wire.begin() δηλώνουμε ότι ξεκινάει η χρήση της wire βιβλιοθήκης και ενεργοποιούνται τα SDA και SCL της πλακέτας. Με την εντολή serial.begin() γίνεται χρήση της βοηθητικής οθόνης του Arduino IDE και με το serial.println(), εκτυπώνεται στη βοηθητική οθόνη το μήνυμα “I2C Scanner”.

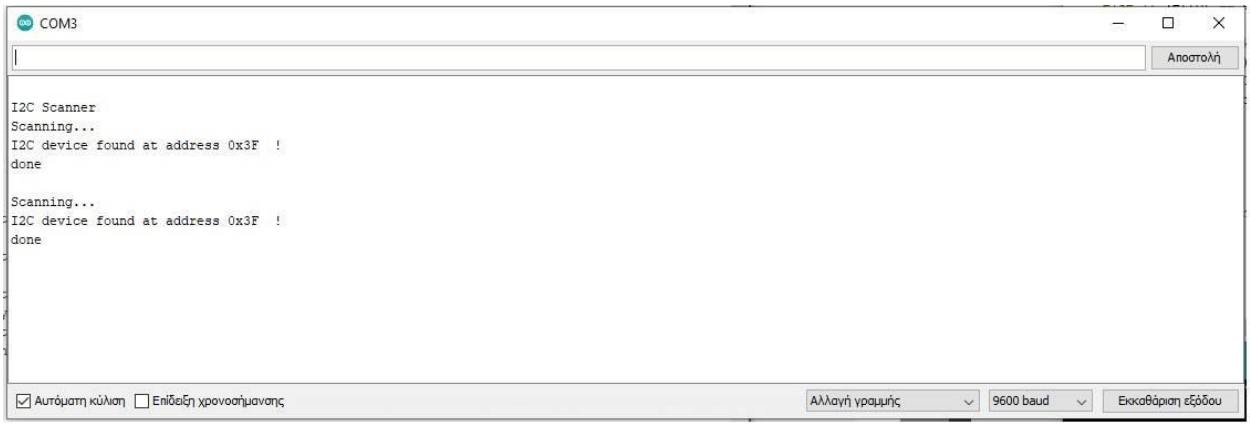
```
void loop()

{

 Serial.println("Scanning...");

 nDevices = 0;

 for(address = 1; address < 127; address++ )

 {

  Wire.beginTransmission(address);

  error = Wire.endTransmission();
```

```
  if (error == 0)
  {
    Serial.print("I2C device found at address 0x");
    if (address<16)
      Serial.print("0");
    Serial.print(address,HEX);
    Serial.println("  !");
    nDevices++;
  }
  else if (error==4)
  {
    Serial.print("Unknown error at address 0x");
    if (address<16)
      Serial.print("0");
    Serial.println(address,HEX);
  }
 }
 if (nDevices == 0)
  Serial.println("No I2C devices found\n");
 Else
  Serial.println("done\n");
 delay(5000);
}
```

Με την εντολή void loop() ξεκινάει να εκτελείται επαναλαμβανόμενα το πρόγραμμα που είναι μέσα στις αγκύλες. Στην αρχή εμφανίζεται το μήνυμα ότι ξεκινάει την ανίχνευση συσκευής i2c και γίνεται η αρχικοποίηση της μεταβλητής nDevices = 0. Στη συνέχεια γίνεται επανάληψη, ξεκινώντας με την τιμή της address που ισούται με 1, μέχρι η μεταβλητή address να γίνει 126 και κάθε φορά η address γίνεται address + 1. Κάθε επανάληψη ξεκινάει με την εντολή Wire.beginTransmission(address), η οποία χρησιμοποιείται για να αρχίσει τη μετάδοση σήματος σε I2C slaves συσκευές. Η Wire.beginTransmission μπορεί να μας δώσει 5 καταστάσεις:

1. 0: Επιτυχία
2. 1: Μεγάλη πληροφορία κατά τη μετάδοση
3. 2: Δεν επιστράφηκε κανένα σήμα
4. 3: Καμία πληροφορία από τη slave συσκευή
5. 4: Κάποιο άλλο πρόβλημα

Η κατάσταση αυτή αποθηκεύεται στη μεταβλητή error. Στη συνέχεια, γίνεται έλεγχος της μεταβλητής και αν είναι ίση με το 0, τότε είναι επιτυχής ο έλεγχος και εκτυπώνεται ο αριθμός 0. Ακολούθως, εκτυπώνεται η μεταβλητή address σε δεκαεξαδικό σύστημα. Αν στον έλεγχο η μεταβλητή είναι ίση με 4, τότε εμφανίζει μήνυμα ότι βρέθηκε η συσκευή, αλλά για κάποιο λόγο δεν γίνεται σωστά η επικοινωνία. Σε οποιαδήποτε άλλη κατάσταση της μεταβλητής, εμφανίζει μήνυμα ότι δεν βρέθηκε συσκευή. Στο τέλος, υπάρχει μια καθυστέρηση 5 δευτερολέπτων για να κάνει επανάληψη τις εντολές της void loop.

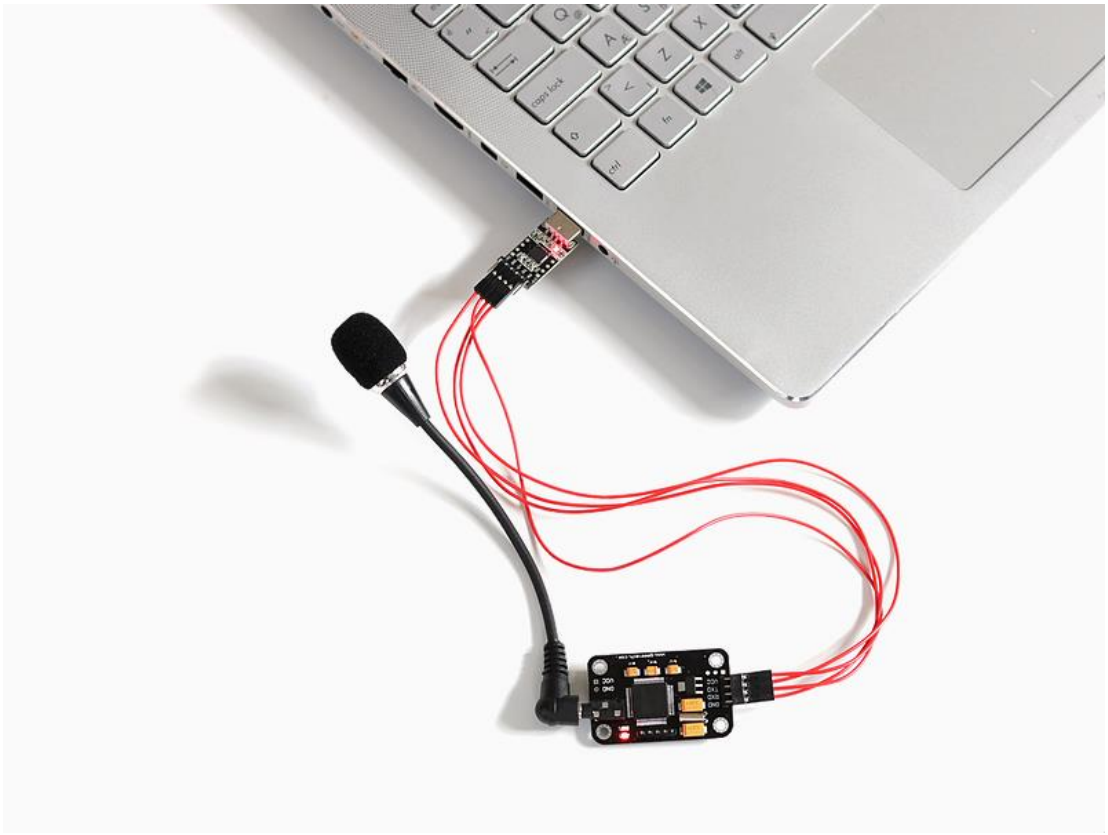


Εικόνα 18. Εύρεση της Address της LCD Οθόνης μέσω του Arduino IDE

3.3 Προγραμματισμός του Voice Recognition Shield

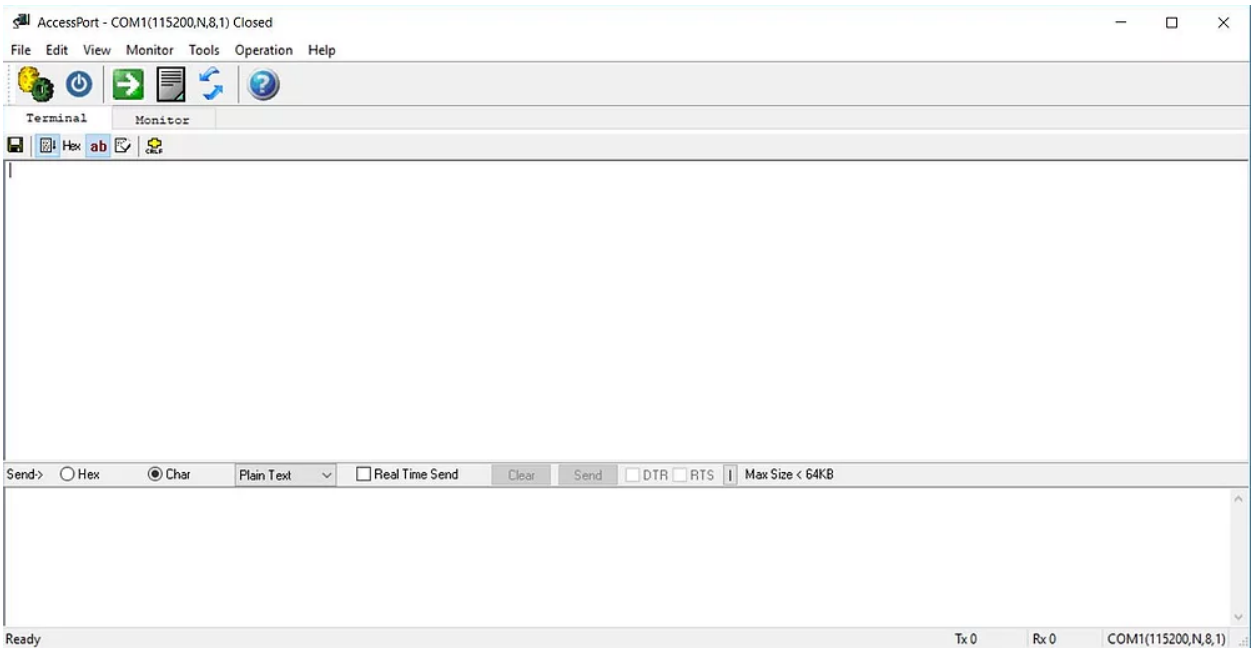
Για τον προγραμματισμό θα χρειαστεί να συνδέσουμε το USB to TTL σε έναν ηλεκτρονικό υπολογιστή και μέσω 4 καλωδίων θα συνδεθεί το USB to TTL με το Voice Recognition. Τα καλώδια συνδέονται με την παρακάτω σειρά:

Voice Recognition Module		USB to TTL Converter
VCC	———	5V
GND	———	GND
TXD	———	RXD
RXD	———	TXD



Εικόνα 19. Σύνδεση Voice Recognition Module με Η/Υ μέσω USB to TTL Module

Αφού το συνδέσουμε σε μια θύρα του υπολογιστή, θα χρησιμοποιήσουμε το AccessPort για να κατοχυρώσουμε τις φωνητικές εντολές.



Εικόνα 20. Περιβάλλον AccessPort

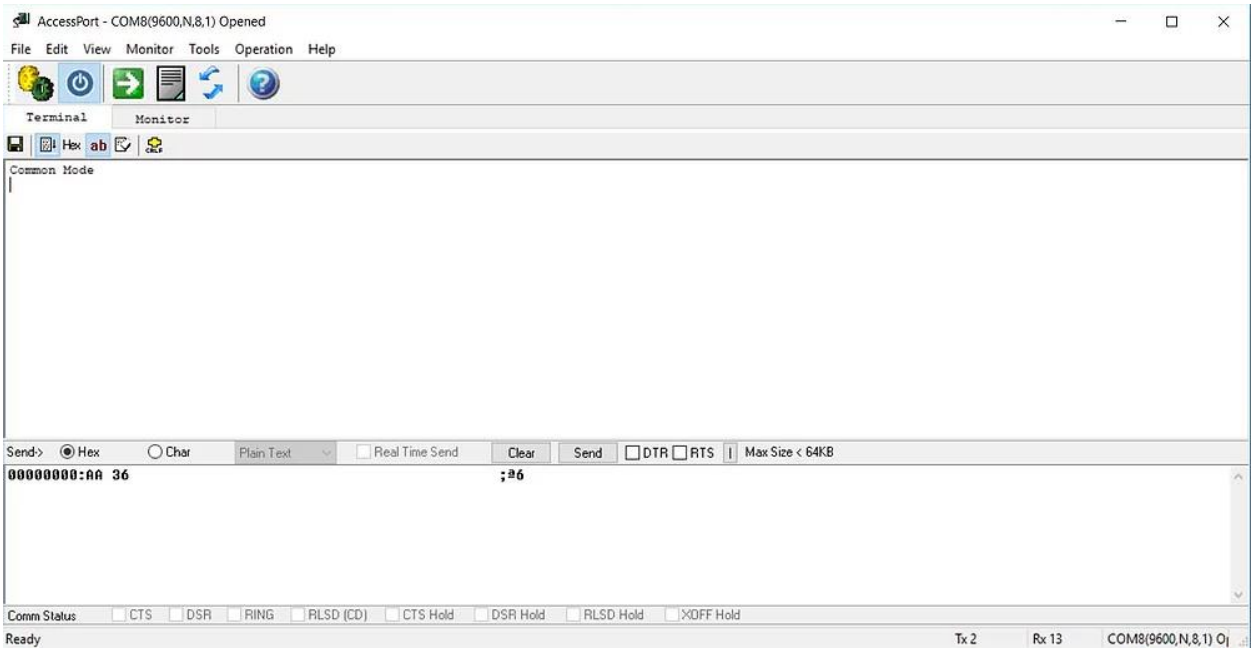
Κάνουμε τις απαιτούμενες ρυθμίσεις για να μπορεί να αναγνωρίσει το πρόγραμμα σε ποια θύρα είναι συνδεμένο το USB to TLL. Οι εντολές που θα δέχεται το πρόγραμμα είναι σε δεκαεξαδική μορφή και η μορφοποίηση των εντολών είναι AA XX, όπου XX είναι ο αριθμός της εντολής που θέλουμε να εκτελέσουμε από τον παρακάτω πίνακα:

Key (HEX format)	Description	Respond in Common Mode	Respond in Compact Mode
0x00	Enter into "Waiting" state	"Waiting! | n" : successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x01	Delete the instructions of group 1	"Group1 Deleted ! | n" : successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x02	Delete the instructions of group 2	"Group2 Deleted ! | n" : successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x03	Delete the instructions of group 3	"Group3 Deleted ! | n" : successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x04	Delete the instructions of all the 3 groups	" All Groups Deleted !|n " : successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x11	Begin to record instructions of group 1	"ERROR! | n" : Instruction error "START | n" : Ready for recording, you can speak now "No voice | n" : no voice detected "Again | n" : Speak the voice instruction again. Do not speak until getting the START message "Too loud | n" : Too loud to record "Different | n" : voice instruction confirming failed. Voice for the second chance is different with the first one. "Finish one | n" : recording one voice instruction successfully "Group1 finished! | n" : finish recording group 1	0xe0 : Instruction error 0x40 : Ready for recording, you can speak now 0x41 : no voice detected 0x42 : Speak the voice instruction again. Do not speak until getting the START message 0x43 : Too loud to record 0x44 : voice instruction confirming failed. Voice for the second chance is different with the first one. 0x45 : recording one voice instruction successfully 0x46 : finish recording group 1
0x12	Begin to record instructions of group 2	"ERROR! | n" : Instruction error "START | n" : Ready for recording, you can speak now "No voice | n" : no voice detected "Again | n" : Speak the voice instruction again. Do not speak until getting the START message "Too loud | n" : Too loud to record "Different | n" : voice instruction confirming failed. Voice for the second chance is different with the first one. "Finish one | n" : recording one voice instruction successfully "Group2 finished! | n" : finish recording group 2	0xe0 : Instruction error 0x40 : Ready for recording, you can speak now 0x41 : no voice detected 0x42 : Speak the voice instruction again. Do not speak until getting the START message 0x43 : Too loud to record 0x44 : voice instruction confirming failed. Voice for the second chance is different with the first one. 0x45 : recording one voice instruction successfully 0x47 : finish recording group 2
0x13	Begin to record instructions of group 3	"ERROR! | n" : Instruction error "START | n" : Ready for recording, you can speak now "No voice | n" : no voice detected "Again | n" : Speak the voice instruction again. Do not speak until getting the START message "Too loud | n" : Too loud to record	0xe0 : Instruction error 0x40 : Ready for recording, you can speak now 0x41 : no voice detected 0x42 : Speak the voice instruction again. Do not speak until getting the START message

		"Different | n" : voice instruction confirming failed. Voice for the second chance is different with the first one. "Finish one | n" : recording one voice instruction successfully "Group3 finished! | n" : finish recording group 3	0x43 : Too loud to record 0x44 : voice instruction confirming failed. Voice for the second chance is different with the first one. 0x45 : recording one voice instruction successfully 0x48 : finish recording group 3
0x21	Import group 1 and be ready for voice instruction	"Group1 Imported !|n" : Successful "ERROR! | n" : Instruction error "Import failed !|n" : Importing voice group failed	0xcc : Successful 0xe0 : Instruction error 0xe1 : Importing voice group failed
0x22	Import group 2 and be ready for voice instruction	"Group2 Imported !|n" : Successful "ERROR! | n" : Instruction error "Import failed !|n" : Importing voice group failed	0xcc : Successful 0xe0 : Instruction error 0xe1 : Importing voice group failed
0x23	Import group 3 and be ready for voice instruction	"Group3 Imported !|n" : Successful "ERROR! | n" : Instruction error "Import failed !|n" : Importing voice group failed	0xcc : Successful 0xe0 : Instruction error 0xe1 : Importing voice group failed
0x24	Query the recorded group	"Used group:0|n" : No group is recorded "Used group:1|n" : Group 1 is recorded "Used group:2|n" : Group 2 is recorded "Used group:3|n" : Group 3 is recorded "Used group:12|n" : Group 1 and Group 2 are recorded "Used group:13|n" : Group 1 and Group 3 are recorded "Used group:23|n" : Group 2 and Group 3 are recorded "Used group:123|n" : All the 3 groups are recorded "ERROR! | n" : Instruction error	0x00 : No group is recorded 0x01 : Group 1 is recorded 0x02 : Group 2 is recorded 0x04 : Group 3 is recorded 0x03 : Group 1 and Group 2 are recorded 0x05 : Group 1 and Group 3 are recorded 0x06 : Group 2 and Group 3 are recorded 0x07 : All the 3 groups are recorded 0xe0 : Instruction error
0x31	Change the baud rate to 2400bps	"Baud: 2400|n" : Successful "ERROR! | n" : Instruction error	0xcc : successful 0xe0 : Instruction error
0x32	Change the baud rate to 4800bps	"Baud: 4800|n" : Successful "ERROR! | n" : Instruction error	
0x33	Change the baud rate to 9600bps	"Baud: 9600|n" : Successful "ERROR! | n" : Instruction error	
0x34	Change the baud rate to 19200bps	"Baud: 19200|n" : Successful "ERROR! | n" : Instruction error	
0x35	Change the baud rate to 38400bps	"Baud: 38400|n" : Successful "ERROR! | n" : Instruction error	
0x36	Switch to Common Mode	"Common Mode|n" : Successful "ERROR! | n" : Instruction error	
0x37	Switch to Compact Mode	"Compact Mode|n" : Successful "ERROR! | n" : Instruction error	
0xbb	Query version information	Version information	No respond

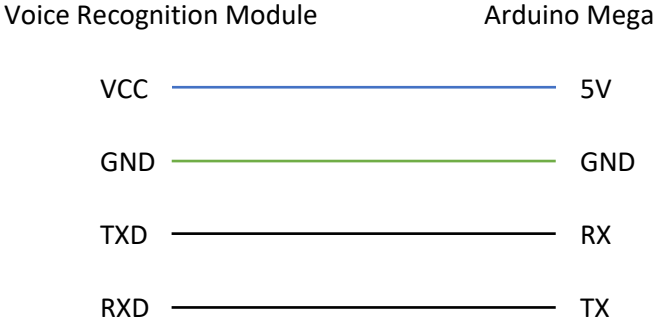
Πίνακας 2. Οι εντολές που δέχεται το AccessPort

Ξεκινάμε γράφοντας την εντολή AA 36 για να μπούμε σε Common Mode, ώστε να ξεκινήσει το USB to TTL να δέχεται τις εντολές.



Εικόνα 21. Common Mode στο AccessPort

Μόλις ο χρήστης στείλει την εντολή, τότε το πρόγραμμα θα εμφανίσει ότι πλέον είμαστε σε Common Mode. Στη συνέχεια, θα στείλουμε την εντολή AA 11 για να ξεκινήσει το πρόγραμμα να δέχεται τις φωνητικές εντολές του 1ου γκρουπ εντολών από τον χρήστη. Σε κάθε γκρουπ υπάρχει η δυνατότητα αποθήκευσης μέχρι και 5 φωνητικών εντολές. Κάθε φωνητική εντολή θα πρέπει να την επαναλάβουμε δύο φορές για να γίνει επιβεβαίωση ότι είναι σωστή. Μόλις στείλουμε την εντολή AA 11, εμφανίζει μήνυμα START που σημαίνει ότι θα πρέπει να δώσουμε την πρώτη εντολή και μετά ξανά START, ώστε να την επαναλάβουμε για να γίνει ο έλεγχος. Μόλις δώσουμε και τις 5 εντολές εμφανίζεται το μήνυμα ότι το GROUP 1 είναι έτοιμο. Για να αποθηκεύσουμε τις εντολές, στέλνουμε την εντολή AA 21. Για τα υπόλοιπα 2 γκρουπ εντολών ακολουθούμε τα ίδια βήματα με τις εντολές AA 12-AA 22 και AA 13-AA23 αντίστοιχα. Αφού τελειώσουμε με την καταχώρηση εντολών, αποσυνδέουμε από τον υπολογιστή το USB to TTL και συνδέουμε το Voice Recognition Module με την πλακέτα του Arduino Mega με την παρακάτω συνδεσμολογία:



3.4 Ο κώδικας του Ανελκυστήρα

3.4.1 Ο κώδικας

Σε αυτό το κεφάλαιο θα δούμε και θα αναλύσουμε τον κώδικα, ο οποίος γράφτηκε για τον σκοπό της παρούσας εργασίας.

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x3F, 20, 4);

byte com = 0;

void setup()
 {

        Serial.write(0xAA);

        Serial.write(0x37);

        delay(1000);

        Serial.write(0xAA);

        Serial.write(0x21);
```

```
        lcd.init();
        lcd.backlight();
        lcd.setCursor(4, 0);
        lcd.print("* ELEVATOR *");
        lcd.setCursor(6, 1);
        lcd.print("* WITH *");
        lcd.setCursor(1, 2);
        lcd.print("* VOICE COMMANDS *");
        lcd.setCursor(5, 3);
        lcd.print("LOADING...");

        pinMode(2, INPUT_PULLUP);
        pinMode(3, INPUT_PULLUP);

        pinMode(7, OUTPUT);
        pinMode(6, OUTPUT);

        lcd.clear();
        lcd.setCursor(1, 0);
        lcd.print("* PARAKALW PEITE *");
        lcd.setCursor(1,1);
        lcd.print("* TON OROFO TOU  *");
        lcd.setCursor(1,2);
        lcd.print("* PROORISMOU SAS *");
}

void loop()
{
```

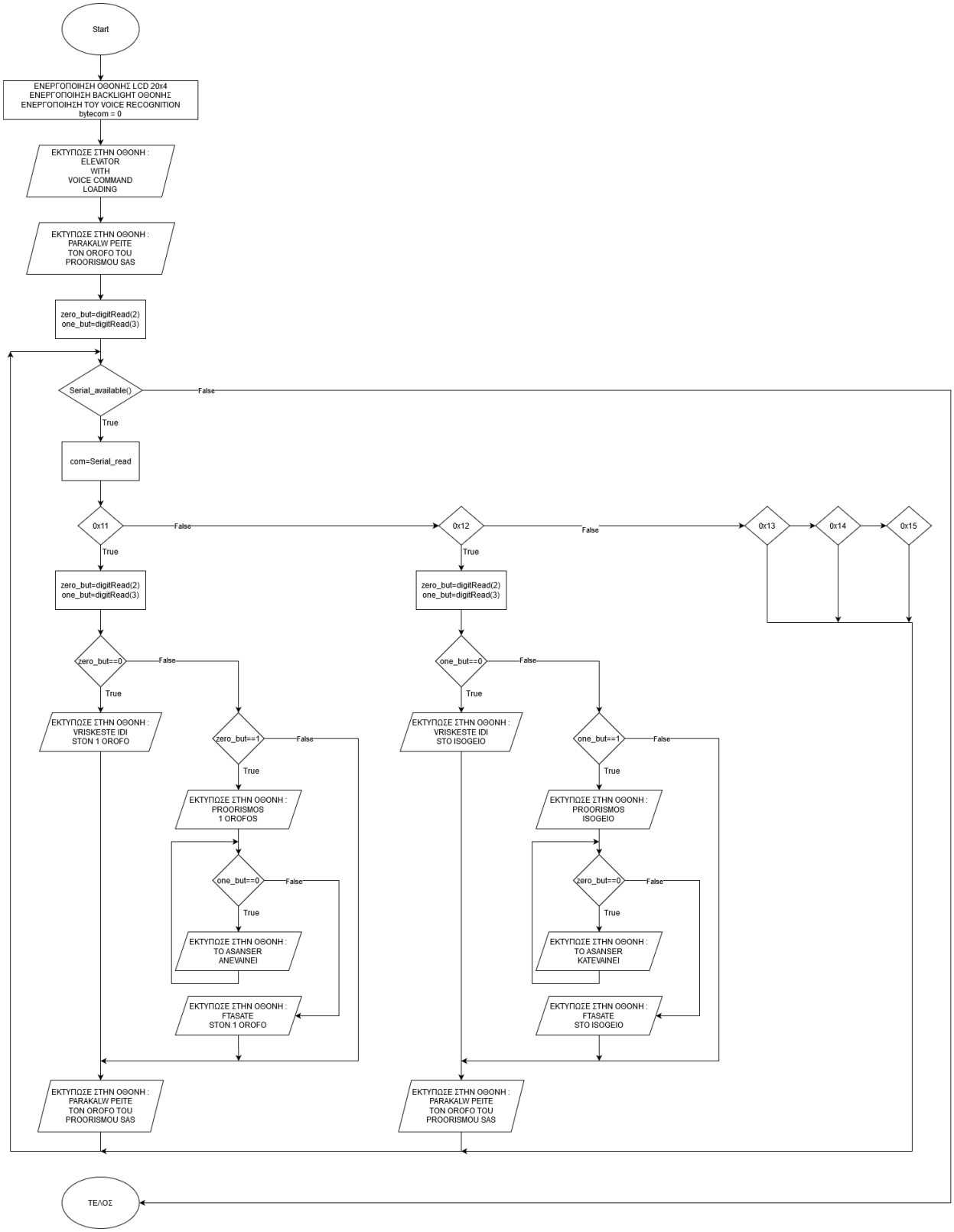
```
int zero_but = digitalRead(2);
int one_but = digitalRead(3);

digitalWrite(7, LOW);
digitalWrite(6, LOW);

delay(1000);

while(Serial.available()) {

com = Serial.read();

switch(com) {

case 0x11:

one_but = digitalRead(3);
zero_but = digitalRead(2);

if (zero_but == 0)
{
        lcd.clear();
        lcd.setCursor(1, 1);
        lcd.print("* VRISKESTE IDI *");
        lcd.setCursor(1, 2);
        lcd.print("* STON 1 OROFO *");
        delay(2000);
}
```

```
else if (zero_but == 1)
{
        lcd.clear();
        lcd.setCursor(2, 1);
        lcd.print("* PROORISMOS *");
        lcd.setCursor(2, 2);
        lcd.print("* 1 OROFOS *");
        digitalWrite(7, HIGH);
        digitalWrite(6, LOW);
        delay(2000);

while (one_but == 0)
{
        lcd.clear();
        lcd.setCursor(3, 1);
        lcd.print("* TO ASANSER *");
        lcd.setCursor(3, 2);
        lcd.print("* ANEVAINEI *");
        delay(200);
        one_but = digitalRead(3);
}

        digitalWrite(7, LOW);
        digitalWrite(6, LOW);
        delay(200);
        lcd.clear();
        lcd.setCursor(4, 1);
        lcd.print("* FTASATE *");
```

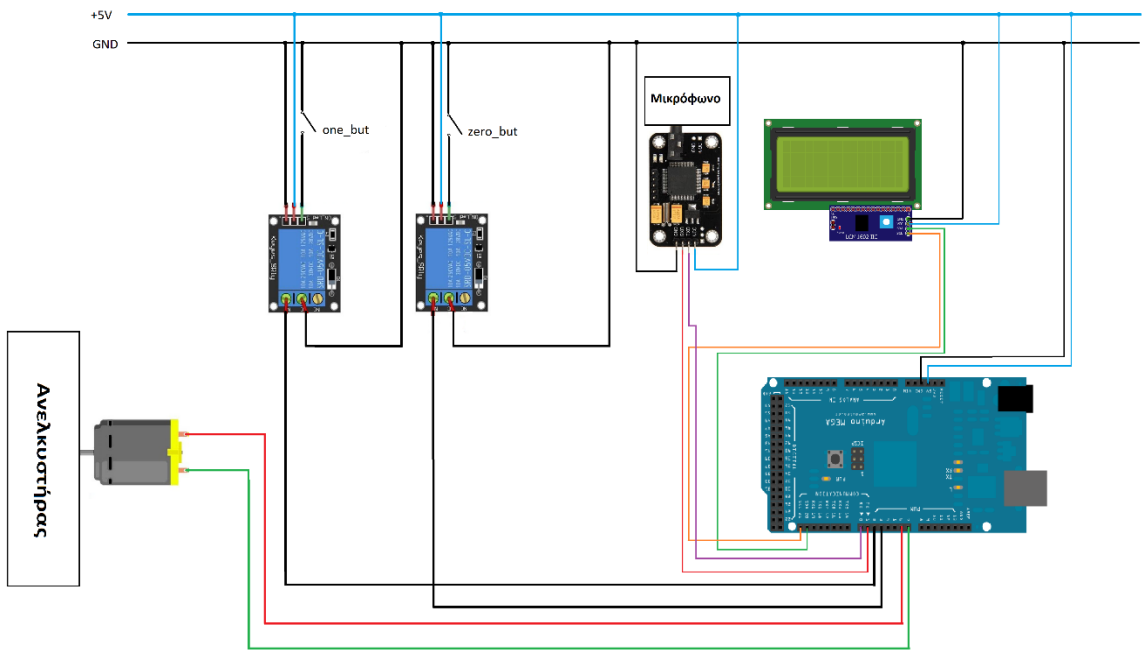
```
            lcd.setCursor(1, 2);
            lcd.print("* STO 1 OROFO *");
            delay(3000);
        }

        lcd.clear();
        lcd.setCursor(1, 0);
        lcd.print("* PARAKALW PEITE *");
        lcd.setCursor(1,1);
        lcd.print("* TON OROFO TOU  *");
        lcd.setCursor(1,2);
         lcd.print("* PROORISMOU SAS *");
        break;

        case 0x12:

        one_but = digitalRead(3);
        zero_but = digitalRead(2);

        if (one_but == 0)
        {
            lcd.clear();
            lcd.setCursor(1, 1);
            lcd.print("* VRISKESTE IDI *");
            lcd.setCursor(1, 2);
            lcd.print("* STO ISOGEIO *");
            delay(2000);
        }
```

```
else if (one_but == 1)
{
        lcd.clear();
        lcd.setCursor(2, 1);
        lcd.print("* PROORISMOS *");
        lcd.setCursor(2, 2);
        lcd.print("* ISOGEIO *");
        digitalWrite(7, LOW);
        digitalWrite(6, HIGH);
        delay(2000);

while (zero_but == 0)
{
        lcd.clear();
        lcd.setCursor(3, 1);
        lcd.print("* TO ASANSER *");
        lcd.setCursor(3, 2);
        lcd.print("* KATEVAINEI *");
        delay(200);
        one_but = digitalRead(2);
}

        digitalWrite(7, LOW);
        digitalWrite(6, LOW);
        delay(200);
        lcd.clear();
        lcd.setCursor(4, 1);
        lcd.print("* FTASATE *");
        lcd.setCursor(1, 2);
```

```
                lcd.print("* STO ISOGEIO *");
                delay(3000);
            }

            lcd.clear();
            lcd.setCursor(1, 0);
            lcd.print("* PARAKALW PEITE *");
            lcd.setCursor(1,1);
            lcd.print("* TON OROFO TOU  *");
            lcd.setCursor(1,2);
             lcd.print("* PROORISMOU SAS *");
            break;

            case 0x13:

            break;

            case 0x14:

            break;

            case 0x15:

            break;
          }
      }
}
```



Διάγραμμα 1. Διάγραμμα ροής του κώδικα



Σχέδιο 1. Συνδεσμολογία του κυκλώματος

3.4.2 Η ανάλυση του κώδικα

Κάθε κώδικας ξεκινάει με τη δήλωση των βιβλιοθηκών

```
#include <Wire.h>


#include <LiquidCrystal_I2C.h>
```

Θα χρησιμοποιήσουμε 2 βιβλιοθήκες:

Wire.h βιβλιοθήκη για το I2C

LiquidCrystal_I2C.h βιβλιοθήκη για την LCD που θα έχουμε συνδεμένη μέσω του I2C

```
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x3F, 20, 4);
```

Με την παραπάνω εντολή γίνεται αρχικοποίηση της οθόνης LCD 20x4 και χρήση της διεύθυνσης 0x3F του δίαυλου I2C, την οποία βρήκαμε, χρησιμοποιώντας τις εντολές στην παράγραφο 3.2.

```
byte com = 0;
```

Γίνεται δήλωση της μεταβλητής com.

```
void setup()
```

Οι παρακάτω εντολές θα εκτελεστούν μία φορά.

```
 {
        Serial.write(0xAA);

        Serial.write(0x37);

        delay(1000);

        Serial.write(0xAA);

        Serial.write(0x21);
```

Με τις εντολές Serial.write() ετοιμάζουμε το voice recognition, ώστε να είναι σε κατάσταση αναμονής των εντολών του χρήστη.

```
        lcd.init();
        lcd.backlight();
```

Με τις δύο παραπάνω εντολές ενεργοποιούμε την LCD.

```
lcd.setCursor(4, 0);
lcd.print("* ELEVATOR *");
lcd.setCursor(6, 1);
lcd.print("* WITH *");
lcd.setCursor(1, 2);
lcd.print("* VOICE COMMANDS *");
lcd.setCursor(5, 3);
lcd.print("LOADING...");
```

Με την εντολή lcd.setCursor() δηλώνουμε από ποιο σημείο επιθυμούμε να ξεκινήσουν να εμφανίζονται οι χαρακτήρες στην οθόνη LCD. Με την εντολή lcd.print() εμφανίζεται το μήνυμα εντός των αγκυλών.

Εικόνα 22. Η οθόνη ενεργοποίησης του συστήματος

```
pinMode(2, INPUT_PULLUP);
pinMode(3, INPUT_PULLUP);


pinMode(7, OUTPUT);
pinMode(6, OUTPUT);
```

Με την εντολή pinMode() δηλώνουμε ποια pins της πλακέτας θα χρησιμοποιήσουμε, ποια pins είναι έξοδοι και ποια pins είναι είσοδοι.

Pin 2 – είσοδος,

Pin 3 – είσοδος,

Pin 7 – έξοδος, μοτέρ

Pin 6 – έξοδος, μοτέρ

```
        delay(3000);
```

Το delay() χρησιμοποιείται, προκειμένου να έχουμε μια μικρή καθυστέρηση για τη σωστή λειτουργία του κυκλώματος

```
        lcd.clear();
        lcd.setCursor(1, 0);
        lcd.print("* PARAKALW PEITE *");
        lcd.setCursor(1,1);
        lcd.print("* TON OROFO TOU  *");
        lcd.setCursor(1,2);
        lcd.print("* PROORISMOU SAS *");

}
```

Εικόνα 23. Οθόνη αναμονής εντολών από τον χρήστη

Με την εντολή lcd.clear() διαγράφονται όλοι οι χαρακτήρες στην οθόνη lcd και εν συνεχεία εμφανίζεται το μήνυμα προς τον χρήστη για να δώσει την εντολή για τον όροφο της επιλογής του.

```
void loop() {
```

Οι παρακάτω εντολές θα εκτελούνται επαναλαμβανόμενα, όσο το κύκλωμα τροφοδοτείται με ρεύμα.

```
 int zero_but = digitalRead(2);
 int one_but = digitalRead(3);
```

Με τις παραπάνω εντολές δηλώνουμε τους δυο διακόπτες που θα χρησιμοποιήσουμε. Έναν στο ισόγειο και έναν στον 1^{ο} όροφο

Pin 2 – διακόπτης ισογείου, zero_but

Pin 3 – διακόπτης 1^{ου} ορόφου, one_but

```
 digitalWrite(7, LOW);
 digitalWrite(6, LOW);
```

Δηλώνουμε τις παραπάνω μεταβλητές σε λογικό 0, ώστε ο ανελκυστήρας μας να βρίσκεται σε κατάσταση αναμονής.

```
delay(1000);
```

```
while(Serial.available()) {

com = Serial.read();

switch(com) {
```

Με τις παραπάνω εντολές πραγματοποιείται ο έλεγχος για το αν είναι ενεργοποιημένο το voice recognition.

```
case 0x11:
```

Το case 0x11 είναι η πρώτη εντολή που έχει ηχογραφηθεί στο voice recognition. Μόλις ο χρήστης δώσει την εντολή, τότε εκτελούνται αυτόματα όλες οι εντολές της case 0x11. Στην περίπτωση μας το case0x11 είναι η εντολή ‘1ος όροφος΄ .

```
one_but = digitalRead(3);
zero_but = digitalRead(2);
```

Με την εντολή digitalRead() ενημερώνουμε τις μεταβλητές one_but και zero_but για την κατάσταση που έχουν το pin 3 και pin 2 αντίστοιχα.

```
if (zero_but == 0)
{
 lcd.clear();
 lcd.setCursor(1, 1);
 lcd.print("* VRISKESTE IDI *");
 lcd.setCursor(1, 2);
```

```
    lcd.print("* STON 1 OROFO *");
    delay(2000);
  }
```

Γίνεται έλεγχος της κατάστασης του zero_but, του διακόπτη, ο οποίος βρίσκεται στο ισόγειο. Αν η κατάσταση του είναι ίση με 0, αυτό σημαίνει ότι ο ανελκυστήρας βρίσκεται ήδη στη θέση που ζητάει ο χρήστης και εμφανίζει το αντίστοιχο μήνυμα 'VRISKESTE IDI STON 1 OROFO'.

Εικόνα 24. Οθόνη θέσης θαλάμου

```
else if (zero_but == 1)
```

Αν ο διακόπτης zero_but είναι ίσος με 1, θα εκτελεστούν οι παρακάτω εντολές:

```
{
    lcd.clear();
    lcd.setCursor(2, 1);
    lcd.print("* PROORISMOS *");
    lcd.setCursor(2, 2);
    lcd.print("* 1 OROFOS *");
    digitalWrite(7, HIGH);
    digitalWrite(6, LOW);
```

```
    delay(2000);
```

Εμφανίζεται το μήνυμα ‘PROORISMOS 1 OROFOS’, που υποδεικνύει ότι ο ανελκυστήρας έχει ξεκινήσει με προορισμό τον 1 όροφο, ενώ ταυτόχρονα ενεργοποιείται το μοτέρ μέσω του pin 7, ώστε να το τροφοδοτεί με +5V. Επιπλέον, υπάρχει μια εντολή καθυστέρησης delay(2000), με σκοπό να πραγματοποιηθεί ορθότερα ο παρακάτω έλεγχος.

Εικόνα 25. Οθόνη για τον προορισμό του Θαλάμου ‘1ος όροφος’

```
while (one_but == 0)
    {
     lcd.clear();
     lcd.setCursor(3, 1);
     lcd.print("* TO ASANSER *");
     lcd.setCursor(3, 2);
     lcd.print("* ANEVAINEI *");
     delay(200);
     one_but = digitalRead(3);
     }
```

Με την εντολή while() εκτελούνται οι εντολές της διαδρομής του θαλάμου, μέχρι να φτάσει στον 1^{ο} όροφο, για την κατάσταση του οποίου μας ενημερώνει συνεχώς η εντολή ελέγχου one_but = digitalRead(3);.

Εικόνα 26. Οθόνη για την κατάσταση του θαλάμου

```
            digitalWrite(7, LOW);
      digitalWrite(6, LOW);
      delay(200);
      lcd.clear();
      lcd.setCursor(4, 1);
      lcd.print("* FTASATE *");
      lcd.setCursor(1, 2);
      lcd.print("* STO 1 OROFO *");
      delay(3000);
    }
```

Οι παραπάνω εντολές θα εκτελεστούν μόλις ο διακόπτης του 1^{ου} ορόφου γίνει ίσος με 1, γεγονός που σημαίνει ότι απενεργοποιείται αμέσως το μοτέρ και εμφανίζεται αντίστοιχο μήνυμα ‘FTASATE STO 1 OROFO’.

Εικόνα 27. Οθόνη για την άφιξη του θαλάμου στον '1ο όροφο'

```
lcd.clear();
    lcd.setCursor(1, 0);
    lcd.print("* PARAKALW PEITE *");
    lcd.setCursor(1,1);
    lcd.print("* TON OROFO TOU  *");
    lcd.setCursor(1,2);
    lcd.print("* PROORISMOU SAS *");

    break;
```

Όταν ο χρήστης φτάσει στον επιθυμητό όροφο, τότε ο ανελκυστήρας εισέρχεται σε κατάσταση αναμονής, εμφανίζοντάς το αντίστοιχο μήνυμα ' PARAKALW PEITE TON OROFO TOU PROORISMOU SAS', αναμένοντας νέα εντολή από τον χρήστη για τον προορισμό του.

```
    case 0x12:  //command 2 is for Red LED
```

Το case 0x12 είναι η πρώτη εντολή που έχει ηχογραφηθεί στο voice recognition. Μόλις ο χρήστης δώσει την εντολή, εκτελούνται αυτόματα όλες οι εντολές της case 0x12. Στην περίπτωση μας, το case0x12 είναι η εντολή 'Ισόγειο'.

```
        one_but = digitalRead(3);
```

```
zero_but = digitalRead(2);
```

Με την εντολή digitalRead() ενημερώνουμε τις μεταβλητές one_but και zero_but για την κατάσταση που έχουν το pin 3 και pin 2 αντίστοιχα.

```
if (one_but == 0)
{
  lcd.clear();
  lcd.setCursor(1, 1); // paei sthn arxh ths LCD
  lcd.print("* VRISKESTE IDI *");
  lcd.setCursor(2, 2); // paei sthn arxh ths LCD
  lcd.print("* STO ISOGEIO *");
  delay(2000);
}
```

Γίνεται έλεγχος της κατάστασης του one_but, του διακόπτη, ο οποίος βρίσκεται στο ισόγειο. Σε περίπτωση που η κατάσταση του είναι ίση με 0, αυτό σημαίνει ότι ο ανελκυστήρας βρίσκεται ήδη στη θέση, την οποία ζητάει ο χρήστης και εμφανίζεται το αντίστοιχο μήνυμα 'VRISKESTE IDI STO ISOGEIO'.

Εικόνα 28. Οθόνη θέσης θαλάμου

```
else if (one_but == 1)
```

Αν ο διακόπτης one_but είναι ίσος με 1, τότε θα εκτελεστούν οι ακόλουθες εντολές:

```
{
lcd.clear();
lcd.setCursor(2, 1);
lcd.print("* PROORISMOS *");
lcd.setCursor(3, 2);
lcd.print("* ISOGEIO *");
digitalWrite(7, LOW); //
digitalWrite(6, HIGH); //
delay(2000);
```

Αρχικά, εμφανίζεται το μήνυμα 'PROORISMOS ISOGEIO', γεγονός που υποδηλώνει ότι ο ανελκυστήρας έχει ξεκινήσει με προορισμό το ισόγειο, ενώ ταυτόχρονα ενεργοποιείται το μοτέρ μέσω του pin 6, έτσι ώστε να το τροφοδοτεί με +5V. Επιπλέον, υπάρχει μια εντολή καθυστέρησης delay(2000), με στόχο να πραγματοποιηθεί ορθότερα ο παρακάτω έλεγχος.

Εικόνα 29. Οθόνη για τον προορισμό του Θαλάμου ' Ισόγειο '

```
while (zero_but == 0)
{
 lcd.clear();
 lcd.setCursor(3, 1);
 lcd.print("* TO ASANSER *");
 lcd.setCursor(3, 2);
 lcd.print("* KATEVAINEI *");
 delay(200);
 zero_but = digitalRead(2);
 }
```

Με την εντολή while() εκτελούνται οι εντολές της διαδρομής του θαλάμου, μέχρι αυτός να φτάσει στον 1ο όροφο, για την κατάσταση του οποίου μας ενημερώνει συνεχώς η εντολή ελέγχου zero_but = digitalRead(2);.

Εικόνα 30. Οθόνη για την κατάσταση του θαλάμου

```
      digitalWrite(7, LOW);
      digitalWrite(6, LOW);
      delay(200);
       lcd.clear();
       lcd.setCursor(4, 1);
       lcd.print("* FTASATE *");
       lcd.setCursor(2, 2);
       lcd.print("* STO ISOGEIO *");
       delay(3000);
    }
```

Οι παραπάνω εντολές θα εκτελεστούν μόλις ο διακόπτης του ισόγειου γίνει ίσος με 1, γεγονός που σημαίνει ότι απενεργοποιείται αμέσως το μοτέρ και εμφανίζεται αντίστοιχο μήνυμα 'FTASATE STO ISOGEIO'.

Εικόνα 31. Οθόνη για την άφιξη του θαλάμου στο 'Ισόγειο'.

```
    lcd.clear();
    lcd.setCursor(1, 0);
    lcd.print("* PARAKALW PEITE *");
    lcd.setCursor(1,1);
    lcd.print("* TON OROFO TOU  *");
    lcd.setCursor(1,2);
```

```
    lcd.print("* PROORISMOU SAS *");




    break;
```

Όταν ο χρήστης φτάσει στον επιθυμητό όροφο, τότε ο ανελκυστήρας εισέρχεται σε κατάσταση αναμονής, εμφανίζοντάς το αντίστοιχο μήνυμα ‘ PARAKALW PEITE TON OROFO TOU PROORISMOU SAS’, αναμένοντας νέα εντολή από τον χρήστη για τον προορισμό του.

```
    case 0x13:



    break;

    case 0x14:



    break;

    case 0x15:



    break;

        }
    }
}
```

Τα παραπάνω case μπορούμε να τα αξιοποιήσουμε σε περίπτωση που έχουμε περισσότερους ορόφους ή σε περίπτωση που δώσουμε κάποια άλλη πληροφορία για τον όροφο. Για παράδειγμα, σε ένα νοσοκομείο ο χρήστης θα δύναται μέσω φωνητικής εντολής να επικαλεστεί το ορθοπεδικό τμήμα και να μεταβεί έτσι αυτόματα σε αυτό.

4. Βιβλιογραφία - Αναφορές

4.1 Διαδικτυακές πήγες:

- Shenzhen Getech Technology Co.,Ltd(Geeetech) is an innovative, technology oriented enterprise that is specialized in the R&D
- Η εταιρεία ανελκυστήρων Vroutsislift με έδρα το Γαλάτσι Αττικής, δραστηριοποιείται στο πεδίο της μελέτης, εγκατάστασης και συντήρησης των ανελκυστήρων κάθε τύπου, εδώ και τουλάχιστον 35 χρόνια. http://www.vroutsislift.gr/elevator-history/
- Arduino is the world's leading open-source hardware and software ecosystem. The Company offers a range of software tools, hardware platforms and documentation enabling almost anybody to be creative with technology. https://www.arduino.cc/
- Creoqode is a London-based technology and design company, specialising in innovative products that enhance users' hardware and software skills in an enjoyable, creative and educational way. https://www.creoqode.com/nova-education-ch17?fbclid=IwAR3UH0n51JGp-BQbLtiuKhJ3Yy7ohxQm_H6HB2QjEiiRzIrqHJFsCx88RFs