
# ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

## ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

### ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

**Σχεδίαση και κατασκευή ανελκυστήρα με αντίβαρο με Arduino**

**Γεωργαντάς Φίλιππος**
**Περλιάνης Άγγελος**

**Εισηγητής: Βελώνη Αναστασία, Λέκτορας Εφαρμογών**

**ΑΘΗΝΑ**
**ΜΑΡΤΙΟΣ 2017**

Περλιάνης Άγγελος
Γεωργαντάς Φίλιππος

# ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

**Σχεδίαση και κατασκευή ανελκυστήρα με αντίβαρο με Arduino**

**Γεωργαντάς Φίλιππος**
**Α.Μ. 40266**

**Περλιάνης Άγγελος**
**Α.Μ. 40747**

**Εισηγητής:**
**Βελώνη Αναστασία, Λέκτορας Εφαρμογών**

**Εξεταστική Επιτροπή:**

**Βελώνη Αναστασία**
**Φατούρος Σταύρος**
**Έλληνας Ιωάννης**

**Ημερομηνία εξέτασης :**

**19 Νοεμβρίου 2018**

Περλιάνης Άγγελος
Γεωργαντάς Φίλιππος

# ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΩΝ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Οι κάτωθι υπογεγραμμένοι Περλιάνης Άγγελος Γαβριήλ, του Κωνσταντίνου, με αριθμό μητρώου 40747 καθώς και ο Γεωργαντάς Φίλιππος, του Κωνσταντίνου, με αριθμό μητρώου 40266, φοιτητές του Τμήματος Μηχανικών Η/Υ Συστημάτων Τ.Ε. του Α.Ε.Ι. Πειραιά Τ.Τ. πριν αναλάβουμε την εκπόνηση της Πτυχιακής Εργασίας μας, δηλώνουμε ότι ενημερωθήκαμε για τα κάτωθι:

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

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

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

# ΕΥΧΑΡΙΣΤΙΕΣ

Θα θέλαμε να ευχαριστήσουμε ιδιαιτέρως την Δρ. Βελώνη, καθώς χάρη σε εκείνην, συλλάβαμε την ιδέα για την παρούσα πτυχιακή η οποία ξεκίνησε σαν ιδέα μεταξύ δυο σπουδαστών, ύστερα από συνεργασία τους στον εργαστήριο ψηφιακών συστημάτων αυτομάτου ελέγχου της Δρ. Βελώνη. Ύστερα από την σύλληψη της ιδέας, έγινε πρόταση στην υπεύθυνη καθηγήτρια του συγκεκριμένου μαθήματος, η οποία και δέχθηκε μετά χαράς, καθώς και στήριξε με κάθε τρόπο και συνεργασία την υλοποίηση της εργασίας αυτής.

Επιπλέον, θα θέλαμε να ευχαριστήσουμε όλους τους φίλους και συνεργάτες οι οποίοι μας στήριξαν και μας συμβούλεψαν σε όλη αυτή την προσπάθεια, καθώς και τα άτομα που ήταν δίπλα μας όλα αυτά τα χρόνια των σπουδών μας.

# ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία πραγματεύεται την σχεδίαση και κατασκευή μοντέλου ανελκυστήρα με αντίβαρο και αυτοματισμό σε Arduino. Οι ανελκυστήρες και τα ανυψωτικά μηχανήματα υπάρχουν από τα αρχαία χρόνια σε πολλούς πολιτισμούς στον πλανήτη. Η ανάπτυξή τους υπήρξε λόγω της ανάγκης μετακίνησης ανθρώπων και φορτίων σε διάφορα επίπεδα μέσω της κάθετης κίνησης. Με την έλευση της βιομηχανικής επανάστασης και την ραγδαία επέκταση των αστικών κέντρων, η ανάγκη ύπαρξης επιπλέον κάθετου χώρου υπήρξε άρρηκτα συνδεδεμένη με την ανάγκη ύπαρξης ανελκυστήρων. Συνεπώς, δραματική αλλαγή στην τεχνολογία των ανελκυστήρων ήρθε ουσιαστικά τον 19ο και 20ο αιώνα μΧ. Προς τα τέλη του 20ου αιώνα, η χρήση των ανελκυστήρων ήταν πλέον διαδεδομένη σε όλον τον κόσμο. Τα περισσότερα ανυψωτικά συστήματα από τις αρχές του 21ου αιώνα ήδη, αποτελούνταν πέραν των βασικών μηχανικών και ηλεκτρολογικών διεργασιών τους, και από τουλάχιστον ένα σύστημα αυτοματισμού. Όσο προχωρούσε η τεχνολογία, τόσο μεγαλύτερη γινόταν και η ανάγκη να εφαρμοσθούν λειτουργίες αυτοματισμού σε πολλούς κλάδους της βιομηχανίας και της πολιτικής ζωής. Φυσικά, οι ανελκυστήρες ήταν από τα πρώτα τεχνολογικά επιτεύγματα στα οποία εφαρμόστηκαν ΣΑΕ. Αυτά συνήθως ήταν κάποιο PLC, στην ύστερη μορφή του ή κάποιο κύκλωμα που υλοποιούσε μια λογική ΨΣΑΕ μέσω ολοκληρωμένων κυκλωμάτων. Τη σημερινή εποχή, οι ανελκυστήρες κατασκευάζονται χρησιμοποιώντας μεθόδους, τεχνογνωσίες και εξαρτήματα τελευταίας τεχνολογίας. Η παρούσα κατασκευή έχει υλοποιηθεί τόσο σε μηχανολογικό επίπεδο, όσο και σε ηλεκτρολογικό αλλά κυρίως και ηλεκτρονικό, πράγμα και το οποίο πραγματεύεται το εργαστήριο ΨΣΑΕ. Ο αυτοματισμός της συγκεκριμένης κατασκευής, υλοποιείται χάρη σε ένα κύκλωμα με Arduino. Έτσι, συνδυάζοντας το μηχανολογικό, το ηλεκτρολογικό και το ηλεκτρονικό μέρος, θα έχουμε ένα λειτουργικό μοντέλο ανελκυστήρα.

# ABSTRACT

The present thesis concerns the design and assembly of an elevator model with counterweight and automation with Arduino. Elevators and elevating machines existed since the ancient times in many civilizations on the planet. Their development came due to the need of transportation of humans and cargo in various levels through vertical move. With the coming of the industrial revolution and the rapid expansion of urban centers, the need for further vertical space was inexplicable bound with the need of existence for elevators. Therefore, dramatic advancement in elevators technology came during the 19$^{th}$ and 20$^{th}$ century AD. By the end of the 20$^{th}$ century, the use of elevators was widely spread throughout the globe. In the beginning of the 21$^{st}$ century, most elevation systems, apart from their basic mechanical and electrical processes, included at least one automation system. The more technology advanced, the more the need for implementation of automation systems became greater, in the industry but in civil life as well. Naturally, elevators were one of the first technological achievements that implemented automation systems. Usually, such automation systems were some form of PLC in its latter form, or an integrated circuit design that utilized a digital automation system logic. Nowadays, elevators are constructed using latest methods, expertise and state of the art components. The present structure was implemented in mechanical level and electrical, but mostly in electronic level, which is also the main subject of the digital automation systems laboratory. The automation of our design is implemented thanks to an Arduino. Thus, combining mechanical, electrical and electronic parts, we will have a functional elevator model.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Ψηφιακά Συστήματα Αυτομάτου Ελέγχου
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: ΣΑΕ, ΨΣΑΕ, Arduino, Rotors, Elevators

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

# ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

# ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

**ΣΑΕ** Συστήματα Αυτομάτου Ελέγχου

**ΨΣΑΕ** Ψηφιακά Συστήματα Αυτομάτου Ελέγχου

**PLC** Programmable Logic Controller

**DC** Direct Current

**AC** Alternating Current

**ΔΕΗ** Δημόσια Εταιρία Ηλεκτρισμού

**MOSFET** Metal Oxide Semi-conductor Field Effect Transistor

**KG** Kilogram

**PWM** Pulse Width Modulation

**FTDI** Future Technology Devices International

**USB** Universal Serial Bus

**ADC** Analog to Digital Converter

**DAC** Digital to Analog Converter

# ΕΙΣΑΓΩΓΗ

## ΚΕΦΑΛΑΙΟ 1

Στο παρόν κεφάλαιο γίνεται μια μικρή εισαγωγή η οποία περιλαμβάνει την ιστορική αναδρομή σχετικά με τους ανελκυστήρες και τα ανυψωτικά μέσα, ορισμοί για τα μέρη ενός ανελκυστήρα καθώς και μια μικρή επεξήγηση για τους διάφορους τύπους ανελκυστήρων και την χρήση και χρησιμότητά τους.

### 1.1 Περιγραφή του αντικειμένου της πτυχιακής εργασίας

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η ανάπτυξη μοντέλου ανελκυστήρα με αντίβαρο και αυτοματισμό υλοποιημένο σε Arduino. Η υλοποίηση του παρόντος μοντέλου έχει γίνει τόσο σε μηχανικό και μηχανολογικό επίπεδο όσο και σε ηλεκτρικό και ηλεκτρονικό. Όλα τα στοιχεία, τα εξαρτήματα, καθώς και τα σύνθετα μηχανικά μέρη της κατασκευής, υλοποιήθηκαν εκ του μηδενός. Κατά συνέπεια, η παρούσα κατασκευή σαν μοντέλο, είναι όσο το δυνατόν πιο ρεαλιστική προσομοίωση σε έναν κανονικό ανελκυστήρα τύπου L, από την εγκατάστασή του μέχρι την λειτουργία του.

### 1.2 Ορισμοί

Ο ανελκυστήρας είναι ένας τύπος κάθετης μετακίνησης όπου μεταφέρει ανθρώπους ή αγαθά μεταξύ ορόφων (επιπέδων, καταστρωμάτων) ενός κτηρίου, πλοίου ή άλλης κατασκευής. Οι ανελκυστήρες τυπικά τροφοδοτούνται από ηλεκτρικά μοτέρ τα οποία είτε κινούν συρματόσχοινα τριβής και συστήματα αντίβαρου, ή αντλούν υδραυλικά υγρά με σκοπό την ανύψωση κυλινδρικού πιστονιού.[1]

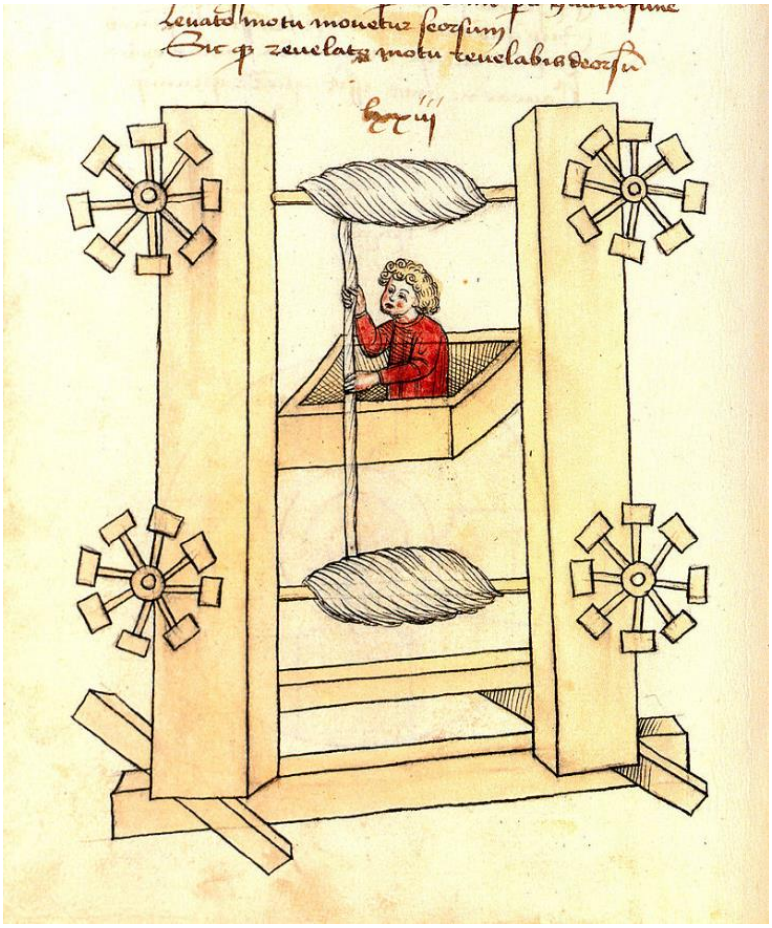
Με τον όρο φρεάτιο ανελκυστήρα, εννοούμε έναν κάθετο χώρο σε ένα κτήριο το οποίο επιτρέπει το πέρασμα του ανελκυστήρα από όροφο σε όροφο.[1]
Με τον όρο θάλαμο ανελκυστήρα, εννοούμε την κατασκευή εκείνη η οποία επιτρέπει την μεταφορά ατόμων ή αντικειμένων από όροφο σε όροφο σε ένα κτήριο μέσω ενός κάθετου φρεατίου.[2]

Με τον όρο αντίβαρο, εννοούμε την κατασκευή εκείνη η οποία ασκεί αντίθετη δύναμη, παρέχοντας ισορροπία και σταθερότητα σε ένα μηχανικό σύστημα.
Ο σκοπός ενός αντίβαρου, είναι να κάνει την ανύψωση ενός φορτίου πιο εργονομική με αποτέλεσμα την εξοικονόμηση ενέργειας, καθώς και την αύξηση ορίου ζωής μιας ανυψωτικής μηχανής. Οι μηχανισμοί αντιβάρων χρησιμοποιούνται συνήθως σε ανελκυστήρες έλξης, σε γερανούς αλλά και σε μηχανήματα αναψυχής σε λούνα πάρκ. [3]

## 1.3 Ιστορική αναδρομή

### 1.3.1 Προ-βιομηχανική εποχή

Η παλαιότερη γνωστή αναφορά σε ανελκυστήρα βρίσκεται στις εργασίες του Ρωμαίου αρχιτέκτονα Βιτρουβίου, ο οποίος ανέφερε ότι ο Αρχιμήδης (το 287 π.Χ. - περίπου 212 π.Χ.) έκτισε τον πρώτο του ανελκυστήρα πιθανότατα το 236 π.Χ. Ορισμένες πηγές από μεταγενέστερες ιστορικές περιόδους αναφέρουν ανελκυστήρες ως καμπίνες σε σχοινί κάνναβης που κινούνται με το χέρι ή με ζώα. [4]. Το 1000, το βιβλίο των μυστικών από τον al-Muradi στην Ισλαμική Ισπανία περιγράφει τη χρήση ανυψωτικού μηχανισμού, σαν ανελκυστήρα, προκειμένου να ανασηκωθεί ένας μεγάλος κριός για να καταστρέψει ένα φρούριο. [5] Τον 17ο αιώνα τα πρωτότυπα των ανελκυστήρων βρίσκονταν στα κτίρια του παλατιού της Αγγλίας και της Γαλλίας. Ο Λουδοβίκος XV της Γαλλίας είχε μια λεγόμενη «ιπτάμενη καρέκλα» που χτίστηκε για μια από τις ερωμένες του στο Chateau de Versailles το 1743. [6]
Οι αρχαίοι και μεσαιωνικοί ανελκυστήρες χρησιμοποιούσαν συστήματα κίνησης που βασίζονταν σε ανυψωτήρες ή αμάξωμα. Η εφεύρεση ενός συστήματος βασισμένου στην κίνηση της βίδας ήταν ίσως το πιο σημαντικό βήμα στην τεχνολογία του ανελκυστήρα από τους αρχαίους χρόνους, οδηγώντας στη δημιουργία σύγχρονων ανελκυστήρων για επιβάτες. Ο πρώτος ανελκυστήρας με βίδα που κατασκευάστηκε από τον Ivan Kulibin και εγκαταστάθηκε στο χειμερινό παλάτι το 1793. Αρκετά χρόνια αργότερα ένας άλλος ανελκυστήρας του Kulibin εγκαταστάθηκε στο Arkhangelskoye κοντά στη Μόσχα.



**Εικόνα 1 Σχέδιο ανελκυστήρα του Konrad Kyeser (1406)**

### 1.3.2 Βιομηχανική εποχή

Η ανάπτυξη ανελκυστήρων οδήγησε στην ανάγκη μετακίνησης πρώτων υλών, συμπεριλαμβανομένου του άνθρακα και ξυλείας από τις πλαγιές. Η τεχνολογία που αναπτύχθηκε από αυτές τις βιομηχανίες και η εισαγωγή κατασκευής δοκών από χάλυβα υπήρξαν οι συνδετικοί κρίκοι για την πρώτη εμφάνιση των ανελκυστήρων επιβατών και εμπορευμάτων που χρησιμοποιούνται σήμερα.

Ξεκινώντας από τα ανθρακωρυχεία, από τα μέσα του 19ου αιώνα οι ανελκυστήρες λειτουργούσαν με ατμό και χρησιμοποιούνταν για τη μεταφορά εμπορευμάτων χύμα σε ορυχεία και εργοστάσια. Αυτές οι ατμοκίνητες συσκευές εφαρμοζόταν σύντομα σε ποικίλους σκοπούς - το 1823, δύο αρχιτέκτονες που εργάζονταν στο Λονδίνο, Burton και Hormer, έχτισαν και λειτουργούσαν ένα νέο τουριστικό αξιοθέατο, το οποίο αποκαλούσαν «αύξουσα αίθουσα». Πελάτες οι οποίοι πλήρωναν το εισιτήριο, μπορούσαν να βρεθούν σε σημαντικό ύψος στο κέντρο του Λονδίνου, δίνοντάς τους μια υπέροχη πανοραμική θέα στο κέντρο της πόλης. [7]

Αρχικά, οι άκομψες κατασκευές ανελκυστήρων με ατμό εξευγενίστηκαν την επόμενη δεκαετία. το 1835, αναπτύχθηκε ένας πρωτοποριακός ανελκυστήρας που ονομάζεται "Teagle" από την εταιρεία Frost και Stutt στην Αγγλία. Ο ανελκυστήρας ήταν κινούμενος με ιμάντα και χρησιμοποιούσε ένα αντίβαρο για επιπλέον ισχύ. [8]
Ο υδραυλικός γερανός επινοήθηκε από τον Sir William Armstrong το 1846, κυρίως για χρήση στις αποβάθρες του Tyneside για φόρτωση φορτίων. Αυτά αντικατέστησαν γρήγορα τους προηγούμενους ανελκυστήρες με ατμό: εκμεταλλευόμενοι τον νόμο του Pascal, προσέφεραν πολύ μεγαλύτερη δύναμη. Μια αντλία νερού παρέσχε μεταβλητή στάθμη πίεσης νερού σε ένα έμβολο εγκλωβισμένο μέσα σε έναν κατακόρυφο κύλινδρο, επιτρέποντας την ανύψωση και την κάθοδο της στάθμης της πλατφόρμας (που φέρει ένα βαρύ φορτίο). Τα αντίβαρα και οι ισορροπίες χρησιμοποιήθηκαν επίσης για την αύξηση της ανυψωτικής ικανότητας της συσκευής.
Ο Henry Waterman της Νέας Υόρκης πιστώνεται με την επινόηση του "σχοινιού ορθού ελέγχου" για έναν ανελκυστήρα το 1850. [9]
Το 1845, ο ναπολιτάνιος αρχιτέκτονας Gaetano Genovese εγκατέστησε στο Βασιλικό Παλάτι της Caserta την «ιπτάμενη καρέκλα», έναν ανελκυστήρα μπροστά από το χρόνο του, καλυμμένο με ξύλο καστανιάς έξω και με ξύλο σφενδάμου μέσα. Περιλάμβανε ένα φως, δύο πάγκους και ένα χειροκίνητο σήμα και θα μπορούσε να ενεργοποιηθεί από έξω, χωρίς καμία προσπάθεια εκ μέρους των επιβατών. Η πρόσφυση ελέγχθηκε από έναν μηχανικό κινητήρα χρησιμοποιώντας ένα σύστημα οδοντωτών τροχών. Ένα σύστημα ασφαλείας σχεδιάστηκε για να δράσει εάν τα καλώδια έσπαγαν. Αποτελούσε μια ακτίνα που σπρωχνόταν προς τα έξω από χαλύβδινο ελατήριο.
Το 1852, ο Ελισσαιος Οτις εισήγαγε τον ανελκυστήρα ασφαλείας, ο οποίος εμπόδισε την πτώση της καμπίνας, εάν το καλώδιο έσπαγε. Επέδειξε την κατασκευή στην έκθεση της Νέας Υόρκης στο Crystal Palace το 1854 και ο πρώτος ανελκυστήρας επιβατών εγκαταστάθηκε στο 488 Broadway στη Νέα Υόρκη στις 23 Μαρτίου 1857.[10][11]

**Εικόνα 2 Ο Ελισσαίος Ότις επιδεικνύει το σύστημα ασφαλείας του**

Το πρώτο φρεάτιο ανελκυστήρα προηγήθηκε του πρώτου ανελκυστήρα κατά τέσσερα χρόνια. Η κατασκευή του κτιρίου Cooper Union Foundation του Peter Cooper στη Νέα Υόρκη άρχισε το 1853. Ένα φρεάτιο ανελκυστήρα συμπεριελήφθη στο σχέδιο, επειδή ο Cooper είχε την πεποίθηση ότι ένας ασφαλής ανελκυστήρας επιβατών θα εφευρεθεί σύντομα. Το φρεάτιο ήταν κυλινδρικό επειδή ο Cooper σκέφτηκε ότι ήταν ο πιο αποδοτικός σχεδιασμός. Αργότερα, ο Otis σχεδίασε έναν ειδικό ανελκυστήρα για το κτίριο.[12][13]

Ο πρώτος ηλεκτρικός ανελκυστήρας κατασκευάστηκε από τον Werner von Siemens το 1880 στη Γερμανία [20]. Ο εφευρέτης Anton Freissler ανέπτυξε τις ιδέες του Siemens και δημιούργησε μια επιτυχημένη επιχείρηση στην Αυστρο-ουγγαρία. Η ασφάλεια και η ταχύτητα των ηλεκτρικών ανελκυστήρων ενισχύθηκαν σημαντικά από τον Frank Sprague, ο οποίος πρόσθεσε τον έλεγχο δαπέδου, την αυτόματη ανέλκυση, τον έλεγχο επιτάχυνσης των θαλάμων και τις ασφάλειες. Ο ανελκυστήρας του κινούνταν ταχύτερα και με μεγαλύτερα φορτία από υδραυλικούς ή ανελκυστήρες ατμού και εγκαταστάθηκαν 584 ηλεκτρικοί ανελκυστήρες πριν ο Sprague πουλήσει την εταιρεία του στην εταιρεία Otis Elevator το 1895. Ο Sprague ανέπτυξε επίσης την ιδέα και την τεχνολογία για πολλαπλούς ανελκυστήρες σε ένα μόνο φρεάτιο.[14]

Το 1882, όταν η υδραυλική ισχύς ήταν μια καλά εδραιωμένη τεχνολογία, μια εταιρεία που αργότερα ονομάστηκε London Hydraulic Power σχηματίστηκε από τον Edward B. Ellington. Κατασκεύασε ένα σύστημα από δίκτυα υψηλής πίεσης και στις δύο πλευρές του Τάμεση, το οποίο τελικά επεκτάθηκε στα 184 μίλια και τροφοδοτούσε περίπου 8.000 μηχανές, κυρίως ανελκυστήρες και γερανούς.[15]
Μέχρι το 1900, υπήρξαν πλήρως αυτοματοποιημένοι ανελκυστήρες, όμως οι επιβάτες ήταν απρόθυμοι να τους χρησιμοποιούν. Η απεργία ενός χειριστή ανελκυστήρα το 1945 στην Νέα Υόρκη, η εφαρμογή κομβίου έκτακτης ανάγκης, η τοποθέτηση τηλεφώνου έκτακτης ανάγκης, καθώς και η χρήση αυτοματοποιημένης επεξηγηματικής φωνής σε χαλαρωτικό ύφος, βοήθησαν στο να γενικευθεί η χρήση ανελκυστήρων από το κοινό.[16]

### 1.3.3 Σύγχρονη εποχή

Οι σύγχρονοι ανελκυστήρες κατασκευάζονται σε ποικίλους τύπους για πολλούς σκοπούς. Εκτός από τις συνήθεις μεταφορές εμπορευμάτων και επιβατών, χρησιμοποιούνται σε πλοία, φράγματα και εξειδικευμένες δομές όπως εκτοξευτήρες ρουκετών. Τα ανυψωτικά μηχανήματα βαρέων φορτίων, ταχείας καθόδου χρησιμοποιούνται σε εργασίες κατασκευής ύψους. Πρακτικά όλοι κινούνται ηλεκτρικά, είτε με συρματόσχοινα, τροχαλία και αντίβαρο, είτε με μηχανισμό τυμπάνου τυλίγματος (που χρησιμοποιείται ακόμα σε πολλούς ανυψωτές φορτίων μικρού ύψους) είτε με ηλεκτρο-υδραυλικό συνδυασμό. Πολλαπλά σύρματα (τρία ή περισσότερα) αυξάνουν τόσο την επιφάνεια πρόσφυσης με την τροχαλία όσο και τον παράγοντα ασφαλείας. Η αποτυχία συρματόσχοινου είναι σπάνια.[17]
Ένας ανελκυστήρας είναι ουσιαστικά μια πλατφόρμα που είτε τραβιέται είτε ωθείται με μηχανικά μέσα. Ένας σύγχρονος ανελκυστήρας αποτελείται από μια καμπίνα (που ονομάζεται επίσης "κλωβός", "καροτσάκι", "αυτοκίνητο" ή ‘’θάλαμος’’) τοποθετημένη σε πλατφόρμα εντός κλειστού χώρου που ονομάζεται φρεάτιο. Στο παρελθόν, οι μηχανισμοί κίνησης του ανελκυστήρα τροφοδοτούνται με υδραυλικά έμβολα ατμού και νερού ή με το χέρι. Σε έναν ανελκυστήρα "έλξης", τα αυτοκίνητα τραβιούνται με τη βοήθεια κυλιόμενων σχοινιών χάλυβα πάνω σε μια βαθιά αυλακωτή τροχαλία, κοινώς ονομαζόμενη τροχαλία στον κλάδο. Το βάρος του αυτοκινήτου αντισταθμίζεται από ένα αντίβαρο. Μερικές φορές κατασκευάζονται

δύο ανελκυστήρες έτσι ώστε τα αυτοκίνητά τους να κινούνται πάντα συγχρονισμένα σε αντίθετες κατευθύνσεις και είναι αντισταθμισμένα μεταξύ τους.

**Εικόνα 3 Σύγχρονη κατασκευή ανελκυστήρων εμπορικού επιπέδου**

## 1.4 Τύποι ανελκυστήρων

### 1.4.1 Ανελκυστήρες ανά χρήση

Οι ανελκυστήρες γενικά μπορούν να κατηγοριοποιηθούν σε δυο μέρη, το πρώτο μέρος είναι οι ανελκυστήρες επιβατών και το δεύτερο μέρος οι ανελκυστήρες φορτίου. Στην πρώτη περίπτωση, οι ανελκυστήρες χρησιμοποιούνται για την μεταφορά ανθρώπων και ελαφριών φορτίων στα διάφορα επίπεδα. Γενικά, οι επιβατικοί ανελκυστήρες περιορίζουν την χωρητικότητά τους ανάλογα με τον διαθέσιμο επίπεδο χώρο. Συνήθως οι επιβατικοί ανελκυστήρες κυμαίνονται σε χωρητικότητες της τάξεως των 500 έως 2.700 kg. Οι περισσότεροι ανελκυστήρες σε κτήρια μέχρι οκτώ ορόφους είναι  είτε ηλεκτρικοί είτε υδραυλικοί. Ανάλογα με το ύψος του κτηρίου και τον αριθμό των ορόφων, διαφοροποιείται αντίστοιχα και η μέγιστη ταχύτητα την οποία μπορεί να φτάσει ο ανελκυστήρας κατά την άνοδό ή την κάθοδό του.

Ένας ανελκυστήρας εμπορευμάτων ή ανελκυστήρας φορτίου είναι ένας ανελκυστήρας σχεδιασμένος να μεταφέρει αγαθά και όχι επιβάτες. Οι ανελκυστήρες φορτίου πρέπει γενικά να εμφανίζουν γραπτή ειδοποίηση στο θάλαμο ότι απαγορεύεται η χρήση τους από τους επιβάτες (αν και όχι απαραιτήτως παράνομη), αν και ορισμένοι ανελκυστήρες φορτίου επιτρέπουν τη διπλή χρήση με τη χρήση ενός διακριτικού ανυψωτήρα. Προκειμένου ένας ανελκυστήρας να είναι νόμιμος για τη μεταφορά επιβατών σε ορισμένες περιπτώσεις, πρέπει να έχει μια σταθερή εσωτερική πόρτα. Οι ανελκυστήρες φορτίου είναι συνήθως μεγαλύτεροι και μπορούν να μεταφέρουν βαρύτερα φορτία από έναν ανελκυστήρα επιβατών, γενικά από 2.300 έως 4.500 kg. Οι ανελκυστήρες φορτίων μπορούν να έχουν χειροκίνητες πόρτες και συχνά έχουν τραχιά εσωτερικά τελειώματα για την πρόληψη ζημιών κατά τη φόρτωση και εκφόρτωση. Αν και υπάρχουν υδραυλικοί ανελκυστήρες φορτίου, οι ηλεκτρικοί ανελκυστήρες είναι ενεργειακά αποδοτικότεροι για την εργασία της ανύψωσης εμπορευμάτων.[1]

### 1.4.2 Ανελκυστήρες ανά τύπο

Η κυριότερη μορφή και πιο διαδεδομένη είναι αυτή του ηλεκτρικού ανελκυστήρα έλξης. Ο ηλεκτρικός ανελκυστήρας έλξης, χρησιμοποιεί έναν ηλεκτρικό κινητήρα για να οδηγήσει ένα συρματόσχοινο στου οποίου τις άκρες υπάρχει στην μία μεριά ο θάλαμος και στην άλλη το αντίβαρο.

Άλλη μορφή τεχνολογίας ανελκυστήρα είναι η υδραυλική. Οι υδραυλικοί ανελκυστήρες χρησιμοποιούν κυλίνδρους οι οποίοι είτε ξεκινάνε από το επίπεδο του εδάφους είτε είναι υπό του εδάφους. Η χαμηλή μηχανική πολυπλοκότητα των υδραυλικών ανελκυστήρων σε σύγκριση με τους ανελκυστήρες έλξης τους καθιστά ιδανικούς για εγκαταστάσεις χαμηλής ανόδου και ελαφριάς κυκλοφορίας. Είναι λιγότερο ενεργειακά αποδοτικοί, καθώς η αντλία λειτουργεί ενάντια στη βαρύτητα για να ωθήσει το θάλαμο και τους επιβάτες του προς τα πάνω. Αυτή η ενέργεια χάνεται όταν το αυτοκίνητο κατεβαίνει με το βάρος του. Η μεγάλη ανάγκη ρεύματος της αντλίας κατά την εκκίνηση θέτει επίσης υψηλότερες απαιτήσεις στο ηλεκτρικό σύστημα του κτιρίου. Υπάρχουν επίσης περιβαλλοντικές ανησυχίες εάν ο κύλινδρος ανύψωσης διαρρεύσει υδραυλικό υγρό στο υπέδαφος.

Επιπλέον τύπος ανελκυστήρα αποτελεί ο πνευματικός ανελκυστήρας. Ένας τέτοιος ανελκυστήρας χρησιμοποιεί ένα κενό αέρος στην κορυφή του θαλάμου και μια

βαλβίδα στην κορυφή του φρεατίου για να μετακινήσει την καμπίνα προς τα επάνω και κλείνει τη βαλβίδα προκειμένου να διατηρηθεί ο θάλαμος στο ίδιο επίπεδο. Ένα διάφραγμα ή ένα έμβολο χρησιμοποιείται ως "φρένο", εάν υπάρξει ξαφνική αύξηση της πίεσης πάνω από τον θάλαμο. Για να πέσει κάτω, ανοίγει τη βαλβίδα έτσι ώστε ο αέρας να μπορεί να πιέσει το πάνω μέρος του φρεατίου, επιτρέποντας στον θάλαμο να πέσει κάτω από το δικό του βάρος. Αυτό σημαίνει επίσης ότι σε περίπτωση διακοπής ρεύματος, ο θάλαμος θα πέσει αυτόματα. Το φρεάτιο είναι κατασκευασμένο από ακρυλικό και είναι πάντα στρογγυλεμένο λόγω του σχήματος της τουρμπίνας που δημιουργεί το κενό αέρος. Για να διατηρηθεί ο αέρας στο εσωτερικό του θαλάμου, χρησιμοποιούνται ελαστικές τσιμούχες. Λόγω τεχνικών περιορισμών, αυτοί οι ανελκυστήρες έχουν χαμηλή χωρητικότητα, επιτρέπουν συνήθως 1-3 επιβάτες.[1]

### 1.5 Διαχωρισμός έργου

Ο διαχωρισμός του έργου μας έχει γίνει σε δύο βασικά επίπεδα, το μηχανολογικό και το ηλεκτρονικό/προγραμματιστικό. Αρχικά υλοποιήθηκε το μηχανολογικό μέρος το οποίο περιλαμβάνει την κατασκευή του φρεατίου, την κατασκευή του θαλάμου, την κατασκευή του αντίβαρου και την κατασκευή του μηχανοστασίου. Στην συνέχεια υλοποιήθηκε το ηλεκτρολογικό μέρος το οποίο περιλαμβάνει απαραίτητες καλωδιώσεις, τροφοδοσία, διακόπτες και συνδεσμολογία κυκλώματος καθώς και ενοποίηση του ηλεκτρολογικού κομματιού με το κομμάτι του αυτοματισμού που αποτελείται από το Arduino.

#### 1.5.1 Μηχανολογικό μέρος

Το μηχανολογικό μέρος της κατασκευής αποτελείται από το φρεάτιο του ανελκυστήρα, τον θάλαμο, το αντίβαρο, το μηχανοστάσιο και τον ηλεκτροκινητήρα.

#### 1.5.2 Ηλεκτρολογικό μέρος

Το ηλεκτρολογικό μέρος αποτελείται από την ηλεκτρολογική ασφάλεια, ενδεικτική λυχνία λειτουργίας της κατασκευής και κλέμες διαχείρισης καλωδίωσης.

Η ηλεκτρολογική ασφάλεια είναι μια διπλή ηλεκτρολογική ασφάλεια αποκοπής της φάσης και του ουδετέρου προς την είσοδο της κατασκευής.

### 1.5.3 Ηλεκτρονικό μέρος

Το ηλεκτρονικό μέρος αποτελείται από τρείς βαθμίδες υλοποίησης, την διαχείριση τροφοδοσίας κινητήρα, την βαθμίδα κεντρικού ελέγχου και την βαθμίδα εντολοδότησης.

### 1.5.4 Προγραμματιστικό μέρος

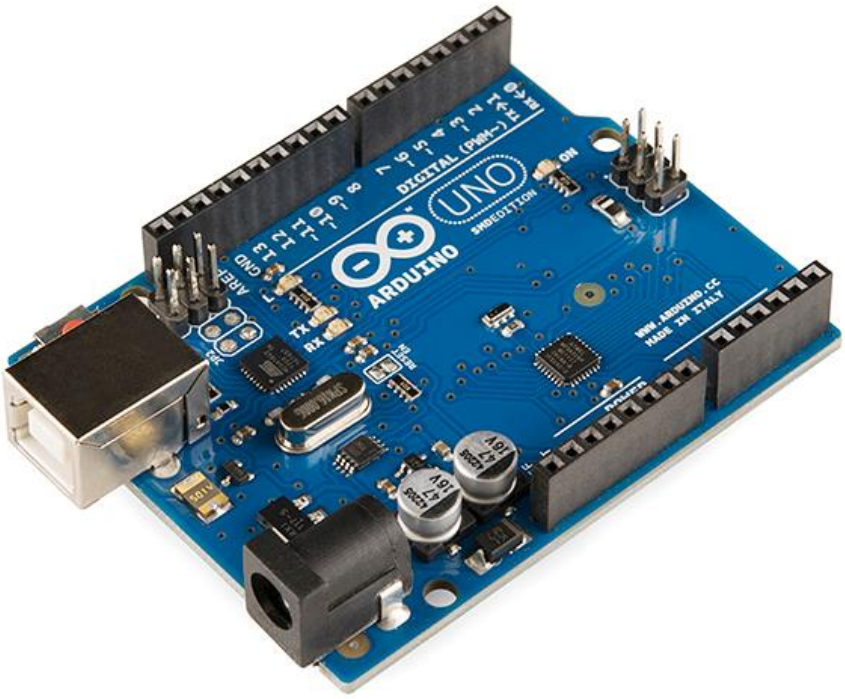
Στο προγραμματιστικό μέρος, έχει συμπεριληφθεί ο προγραμματισμός του Arduino έτσι ώστε να αποτελέσει το ΨΣΑΕ της κατασκευής μας.

# ΚΕΦΑΛΑΙΟ 2

Σε αυτό το κεφάλαιο θα γίνει μια εισαγωγή και επεξήγηση του τί εστί Arduino, ποιες οι πιθανές χρήσεις του και εφαρμογές στην βιομηχανία, καθώς και πως μας έχει εξυπηρετήσει εμάς στην κατασκευή μας.

## 2.1 Το Arduino

Το Arduino είναι μια ηλεκτρονική πλατφόρμα ανοικτού κώδικα βασισμένη σε υλικό και λογισμικό τα οποία είναι σχετικά εύκολο στο να χρησιμοποιηθούν. Οι πλακέτες Arduino είναι σε θέση να διαβάζουν εισόδους - φως σε έναν αισθητήρα, ένα δάκτυλο σε ένα κουμπί ή ένα μήνυμα Twitter - και να το μετατρέπουν σε έξοδο - ενεργοποιώντας έναν κινητήρα, ενεργοποιώντας ένα LED, δημοσιεύοντας κάτι online. Η πλακέτα μπορεί να προγραμματισθεί έτσι ώστε να εκτελεί μια λειτουργεία, διαβάζοντας μια σειρά εντολών που έχουν δοθεί στον μικροελεγκτή. Για να επιτευχθεί αυτό, χρησιμοποιείται η γλώσσα προγραμματισμού του Arduino (βασισμένη στην Wiring), και το λογισμικό περιβάλλον του Arduino (IDE) βασισμένο στο Processing.




**Εικόνα 4 Arduino Uno**

Με τα χρόνια, το Arduino έχει υπάρξει ο εγκέφαλος χιλιάδων έργων, από καθημερινές μικροεφαρμογές και αντικείμενα μέχρι επιστημονικά έργα και σύνθετες κατασκευές. Αποτελείται από μια παγκόσμια κοινότητα δημιουργών, μαθητών, χομπίστων, καλλιτεχνών, προγραμματιστών και επαγγελματιών, οι οποίοι έχουν μαζευτεί γύρω από αυτήν την πλατφόρμα ανοιχτού κώδικα, με αποτέλεσμα η συνεισφορά τους να έχει προστεθεί σε μια τεράστια ποσότητα προσβάσιμης γνώσης που μπορεί να βοηθήσει είτε αρχάριους του κλάδου είτε ειδικούς.

Το Arduino γεννήθηκε στο ινστιτούτο Σχεδιασμού Αλληλεπίδρασης της Ιρβέα, Ιταλία, ως ένα εύκολο εργαλείο για την γρήγορη σχεδίαση πρωτότυπων, έτσι ώστε να μπορούσε να χρησιμοποιηθεί κυρίως από μαθητές χωρίς ιδιαίτερες γνώσεις στα ηλεκτρονικά ή στον προγραμματισμό. Από την στιγμή που έγινε γνωστό στην ευρύτερη κοινότητα του κλάδου, η πλακέτα του Arduino άρχισε να προσαρμόζεται σε καινούριες ανάγκες και προκλήσεις, διαφοροποιώντας τον εαυτό του από απλές οκτάμπιτες πλακέτες, σε προϊόντα του IoT, 3D printing και φορετές συσκευές. Όλες οι πλακέτες Arduino είναι μέρος της κοινότητας ανοιχτού κώδικα, εμψυχώνοντας τους χρήστες τους να τις κατασκευάζουν ανεξάρτητα και να τις προσαρμόζουν στις ανάγκες τους. Το λογισμικό του Arduino είναι επίσης ανοιχτού κώδικα και μεγαλώνει διαρκώς μέσω των συνεισφορών των χρηστών Arduino παγκοσμίως.

Χάρη στην άμεσα προσβάσιμη εμπειρία των χρηστών του Arduino, έχει υπάρξει δυνατή η χρήση του σε χιλιάδες διαφορετικά έργα και εφαρμογές. Το λογισμικό του Arduino είναι σχετικά εύκολο να χρησιμοποιηθεί από αρχάριους αλλά συνάμα και αρκετά ευέλικτο για προχωρημένους χρήστες. Το λογισμικό τρέχει σε Windows, Linux και MacOS. Καθηγητές και μαθητές το χρησιμοποιούν για να κατασκευάσουν επιστημονικά όργανα χαμηλού κόστους, για να επαληθεύσουν θεωρίες της φυσικής και της επιστήμης της χημείας ή για εισαγωγή στον προγραμματισμό και στην ρομποτική. Σχεδιαστές και αρχιτέκτονες δημιουργούν διαδραστικά πρωτότυπα, μουσικοί και καλλιτέχνες το χρησιμοποιούν για τον εξοπλισμό τους ή για να πειραματιστούν με καινούρια μουσικά όργανα.

Υπάρχουν πολλές πλατφόρμες μικροελεγκτών οπού συγκεντρώνουν την φιλοσοφία ενός προγραμματιζόμενου μικροελεγκτή σε ένα περιβάλλον σχετικά εύκολο προς την χρήση του. Το Arduino είναι μια από αυτές τις πλακέτες, με την διαφορά ότι παρέχει κάποια πλεονεκτήματα σε ακαδημαϊκές εφαρμογές, καθηγητές και μαθητές.

Τα κύρια προτερήματα του Arduino είναι:

- Κόστος. Το κόστος μιας πλακέτας Arduino είναι σχετικά χαμηλό σε σχέση με κάποια άλλη παρόμοια πλακέτα. Ακόμα και η πιο φθηνή πλακέτα του Arduino μπορεί ουσιαστικά να συναρμολογηθεί με το χέρι και το κόστος είναι χαμηλότερο των 50$.
- Το λογισμικό του Arduino μπορεί να τρέξει σε διάφορα λειτουργικά συστήματα όπως Windows, Linux και MacOS.
- Το περιβάλλον προγραμματισμού του Arduino είναι απλό και εύκολο στην χρήση, πράγμα το οποίο το κάνει σχετικά φιλικό προς αρχάριους χρήστες.
- Το λογισμικό του Arduino είναι μέρος της κοινότητας ανοιχτού κώδικα, το οποίο σημαίνει οτι έμπειροι προγραμματιστές και χρήστες μπορούν να παρέχουν βελτιωμένες εκδόσεις του κώδικα.
- Όμως και το υλισμικό του Arduino είναι διαθέσιμα στο πλαίσιο της κοινότητας ανοιχτού κώδικα, συνεπώς έμπειροι τεχνικοί ηλεκτρονικών μπορούν να δημιουργήσουν δικές τους εκδόσεις του κυκλώματος ή να βελτιώσουν τις ήδη υπάρχουσες.[18]

### 2.1.1 Arduino pins(ακροδέκτες)

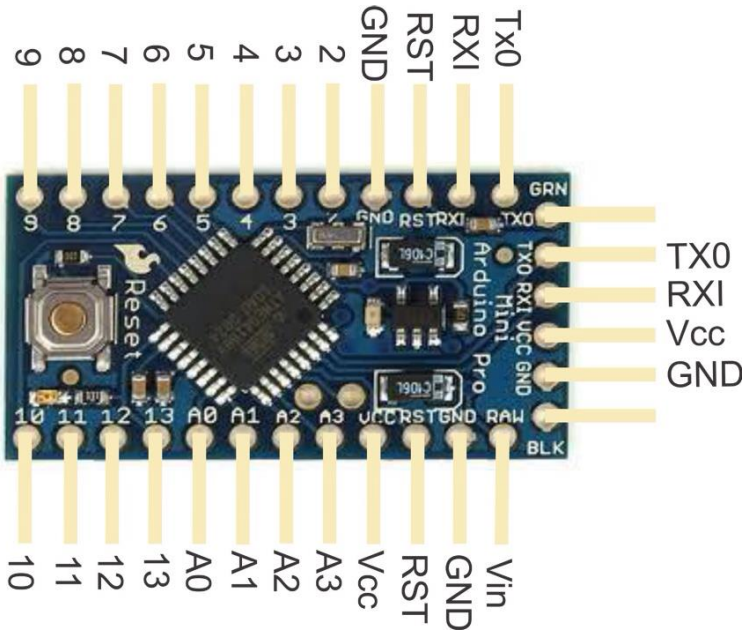
Για την κατασκευή μας, χρησιμοποιήθηκε το μοντέλο Arduino Pro Mini, του οποίου η εικόνα ακροδεκτών φαίνεται παρακάτω.

Το Arduino Pro Mini είναι ένας μικροελεγκτής βασισμένος στην τεχνολογία ATmega328P. Έχει δεκατέσσερεις ακροδέκτες ψηφιακής εισόδου/εξόδου, εκ των οποίων οι έξι μπορούν να χρησιμοποιηθούν ως έξοδοι PWM, έξι αναλογικές εισόδους, ενσωματωμένο ηχειάκι, κουμπί επαναφοράς και τρύπες διαθέσιμες για τοποθέτηση μεταλλικών ακροδεκτών. Ένας εξαπλός ακροδέκτης μπορεί να συνδεθεί σε καλώδιο πλακέτας τύπου FTDI με σκοπό την παροχή ενέργειας μέσω USB άλλα και να παρέχει κανάλι επικοινωνίας. Το Arduino Pro Mini προορίζεται για ημι-μόνιμη εγκατάσταση σε έργα ή εκθέματα. Η πλακέτα παρέχεται χωρίς τοποθετημένους μεταλλικούς ακροδέκτες, επιτρέποντας έτσι την χρήση διαφόρων τύπων ακροδεκτών, ανάλογα με τις προτιμήσεις του χρήστη. Το διάγραμμα ακροδεκτών είναι συμβατό με αυτό του Arduino Mini. Υπάρχουν δυο εκδόσεις του Arduino Pro Mini, μια έκδοση η οποία λειτουργεί στα 3.3V και 8MHz ρολόι

επεξεργαστή και μια η οποία λειτουργεί στα 5V, 16MHz αντίστοιχα. Το Arduino Pro Mini σχεδιάστηκε και κατασκευάστηκε από την SparkFun Electronics.[20]

| | |
|---|---|
| Tx0 (UART) | Θέσεις για σύνδεση με προγραμματιστή (FTDI) |
| RXI (UART) | |
| RST | Θέση για υποδοχή σήματος επανεκκίνησης |
| GND | Γείωση |
| 2-13 | Θέσεις ψηφιακής εισόδου/εξόδου. Τα 3, 5, 6, 9, 10 και 11 υποστηρίζουν PWM λόγω ενσωματομένου DAC. Το pin 13 συνοδευεται και από ένα LED επί της πλακέτας |
| A0 – A5 | Θέσεις αναλογικής εισόδου εύρους 1024 τιμών (έξοδος ενσωματομένου ADC) |
| Vcc (UART) | Θέση εισόδου κανονικοποιημένης τάσης 5V (ή 3.3V στην ανάλογη έκδοση) |
| Vcc | Θέση εισόδου κανονικοποιημένης τάσης (έως 12V) |
| RXI (SPI) | Θέσεις για σειριακή επικοινωνία με άλλες συσκευές |
| TX0 (SPI) | |
| RAW | Θέση εισόδου μη κανονικοποιημένης τάσης εισόδου. |

**Πίνακας 1 Επεξήγηση ακροδεκτών Arduino Pro Mini**




**Εικόνα 5 Ακροδέκτες Arduino Pro Mini**

# ΚΕΦΑΛΑΙΟ 3

Σε αυτό το κεφάλαιο θα επεκταθούμε στο Arduino ως σύστημα αυτοματισμού για την κατασκευή μας.

## 3.1 Εφαρμογή Arduino σε σύστημα ανελκυστήρα

### 3.1.1 Εισαγωγή

Όπως αναφέρθηκε στο εισαγωγικό κεφάλαιο, όλες οι σύγχρονες εφαρμογές ανελκυστήρων διαθέτουν κάποιο σύστημα αυτοματισμού. Κατά συνέπεια, το μοντέλο το οποίο κατασκευάζουμε, πρέπει να περιλαμβάνει ένα ΣΑΕ. Για τον λόγο αυτόν, επιλέξαμε να χρησιμοποιήσουμε Arduino και συγκεκριμένα το Arduino Pro Mini.

### 3.1.2 Το Arduino ως ΨΣΑΕ

Έχοντας ως βάση ανάπτυξης την πλακέτα Arduino, συνδυαστικά με διάφορα ηλεκτρικά και ηλεκτρονικά εξαρτήματα όπως ρελέ, MOSFET, διόδους και λοιπά στοιχεία, δημιουργούμε ένα ηλεκτρονικό κύκλωμα το οποίο θα έχει ως ρόλο τον «εγκέφαλο» της κατασκευής μας. Το κύκλωμα το οποίο δημιουργούμε, αποσκοπεί στο να οδηγεί τον ηλεκτροκινητήρα μας ανάλογα, αξιοποιώντας την λειτουργία αισθητήρων θέσης, έτσι ώστε το ΨΣΑΕ μας να καταλαβαίνει ουσιαστικά που βρίσκεται ο θάλαμος, που θέλουμε να πάει και πως θα φτάσει εκεί, καθώς η ταχύτητα κίνησης του θαλάμου του ανελκυστήρα διαφέρει αναλόγως με την διαφορά απόστασης που έχει από κάποιο από τα επίπεδα.

Για να επιτευχθεί αυτό, το Arduino μας έχει προγραμματιστεί κατάλληλα, έτσι ώστε στις εισόδους του να δέχεται σήμα θέσης κάθε φορά που ο θάλαμος ενεργοποιεί κάποιον από τους μαγνητικούς μας αισθητήρες. Εφόσον υπάρξει μαγνητική ενεργοποίηση κάποιου από τους αισθητήρες, το Arduino το αντιλαμβάνεται αυτό μέσα από τις εισόδους του, με αποτέλεσμα να πράττει ανάλογα όσον αφορά την λειτουργεία του κινητήρα.

Συνεπώς, το Arduino δύναται να διαχειριστεί τον ηλεκτρικό κινητήρα, αυξάνοντας ή μειώνοντας την ταχύτητά του ανάλογα, σύμφωνα με τα σήματα που δέχεται ενδεχομένως στις εισόδους του, έτσι ώστε η μετακίνηση του θαλάμου μεταξύ των επιπέδων να είναι όσο πιο ομαλή γίνεται.

# ΚΑΤΑΣΚΕΥΗ

## ΚΕΦΑΛΑΙΟ 4

Σε αυτό το κεφάλαιο γίνεται ανάλυση του μηχανολογικού μέρους της κατασκευής μας στα επί μέρους κομμάτια του, καθώς και η προσέγγιση υλοποίησης και η μέθοδος συναρμολόγησης του υλικού κομματιού.

### 4.1 Μηχανολογικό μέρος

Κάθε μέρος το οποίο αποτελεί τη μηχανολογική κατασκευή αναλύεται περαιτέρω έτσι ώστε να γίνεται πλήρως κατανοητή η σημασία ύπαρξής του καθώς και ο ρόλος που επιτελεί στην ορθή λειτουργία του ανελκυστήρα.

#### 4.1.1 Φρεάτιο

Η συγκεκριμένη κατασκευή φρεατίου, αποτελείται από έναν απλό σχετικά μεταλλικό σκελετό. Αυτή η μέθοδος επιλέχθηκε κυρίως για πρακτικούς λόγους που θα αναλυθούν παρακάτω, αλλά και για να διατηρηθεί η μινιμαλιστική φύση της κατασκευής.

**Εικόνα 6 Μεταλλικός σκελετός φρεατίου**

Το φρεάτιο της κατασκευής μας, είναι ουσιαστικά τέσσερις μεταλλικές κολώνες οι οποίες ενώνονται μεταξύ τους με μεταλλικές νευρώσεις σε διαγώνιο σχήμα.

**Εικόνα 7 Μεταλλικές νευρώσεις φρεατίου**

Οι μεταλλικές νευρώσεις διαγώνιου κατευθύνσεως από κολώνα σε κολώνα, εξυπηρετούν και για την τοποθέτηση μεταλλικών παράλληλων στο έδαφος βοηθητικών υποστηρικτικών σίδερων, τα οποία με την σειρά τους αποτελούν στηρίγματα εξαρτημάτων, απαραιτήτων για την ορθή λειτουργία του ανελκυστήρα.

**Εικόνα 8 Μεταλλική λάμα στήριξης παράλληλη στο έδαφος**

**Εικόνα 9 Μακρινή λεπτομέρεια βοηθητικών μετάλλων**

Επάνω στα μεταλλικά σίδερα τα οποία είναι τοποθετημένα στον σκελετό, έχουν τοποθετηθεί κάθετα με την κατεύθυνση του εδάφους, μεταλλικά σίδερα τα οποία αποτελούν δρόμους και για τον θάλαμο καθώς και για το αντίβαρο. Επιπλέον, έχει τοποθετηθεί ένα σίδερο επίσης κάθετης κατεύθυνσης, το οποίο χρησιμεύει σαν βάση για τους αισθητήρες ορόφου της κατασκευής. Περαιτέρω επεξήγηση στους αισθητήρες αυτούς, θα γίνει στο κεφάλαιο ηλεκτρονικής ανάλυσης.

**Εικόνα 10 Κάθετοι μεταλλικοί οδηγοί στήριξης θαλάμου – αντίβαρου**

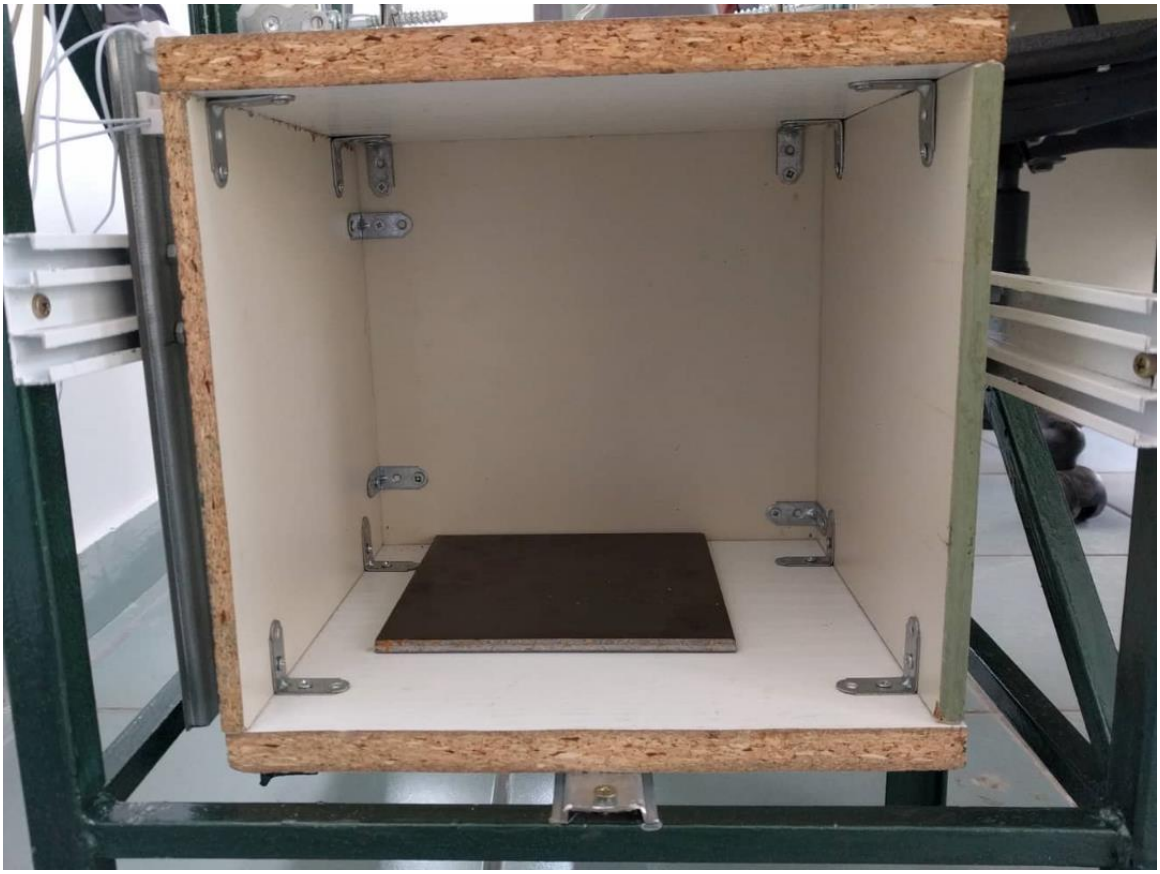
**Εικόνα 11 Βάση τοποθέτησης αισθητήρων θέσης θαλάμου**

Στο οριακό χαμηλό επίπεδο του μεταλλικού σκελετού, έχει τοποθετηθεί μια μεταλλική λάμα οριζόντιας κατεύθυνσης σε σχέση με το έδαφος, η οποία χρησιμεύει σαν βάση για τον θάλαμο, έτσι ώστε ο θάλαμος να μην έχει το περιθώριο να βγει εκτός ορίου της κατασκευής.

**Εικόνα 12 Βάση ανάπαυσης θαλάμου στο χαμηλό όριο**

### 4.1.2 Θάλαμος

Η παρούσα κατασκευή θαλάμου, αποτελείται από μια ξύλινη επί το πλείστον ορθογώνια κατασκευή.



**Εικόνα 13 Λεπτομέρεια θαλάμου**

Τα ξύλα κρατούνται ενωμένα μεταξύ τους με την χρήση μεταλλικών γωνιών και βιδών.

**Εικόνα 14 Λεπτομέρεια μεταλλικών γωνιών**

Στις παράλληλες πλευρές του θαλάμου, έχουν τοποθετηθεί ράουλα τα οποία έχουν ευθυγραμμισθεί έτσι ώστε να εφάπτονται στους κάθετους μεταλλικούς οδηγούς που είναι τοποθετημένοι στον μεταλλικό σκελετό.

**Εικόνα 15 Λεπτομέρεια ράουλου επαφής θαλάμου με βοηθητικό δρόμο φρεατίου**

Για τον λόγο αυτό, έχουν τοποθετηθεί μεταλλικές γωνίες στα ράουλα, έτσι ώστε η ρύθμιση για την απόστασή τους από τους μεταλλικούς οδηγούς, να είναι μεταβλητή, ανάλογα με τις ανάγκες της κατασκευής.

**Εικόνα 16 Λεπτομέρεια μεταλλικής γωνίας ράουλων θαλάμου**

Η συγκεκριμένη τεχνοτροπία εξυπηρετεί στο να μην αποκλίνει ο θάλαμος από τον παράλληλο προς το έδαφος άξονά του, καθώς εκτελεί την κάθετη κίνησή του.
Η κάθετη κίνηση του θαλάμου ικανοποιείται με την χρήση ενός συρματόσχοινου επικαλυμμένου με συνθετικό υλικό PVC με σκοπό την όσο το δυνατόν μεγαλύτερη ύπαρξη δύναμης τριβών στα μέρη κίνησής του.
Το συρματόσχοινο έρχεται σε επαφή με τον θάλαμο μέσω ενός γάντζου, ο οποίος είναι βιδωμένος στην εξωτερική άνω παράλληλη επιφάνεια του θαλάμου.
Στο άκρο του συρματόσχοινου, στο σημείο που έρχεται σε επαφή με τον γάντζο στην κορυφή του θαλάμου, έχει τοποθετηθεί σφιγκτήρας , έτσι ώστε το συρματόσχοινο να μην απομακρύνεται από τη θέση του.

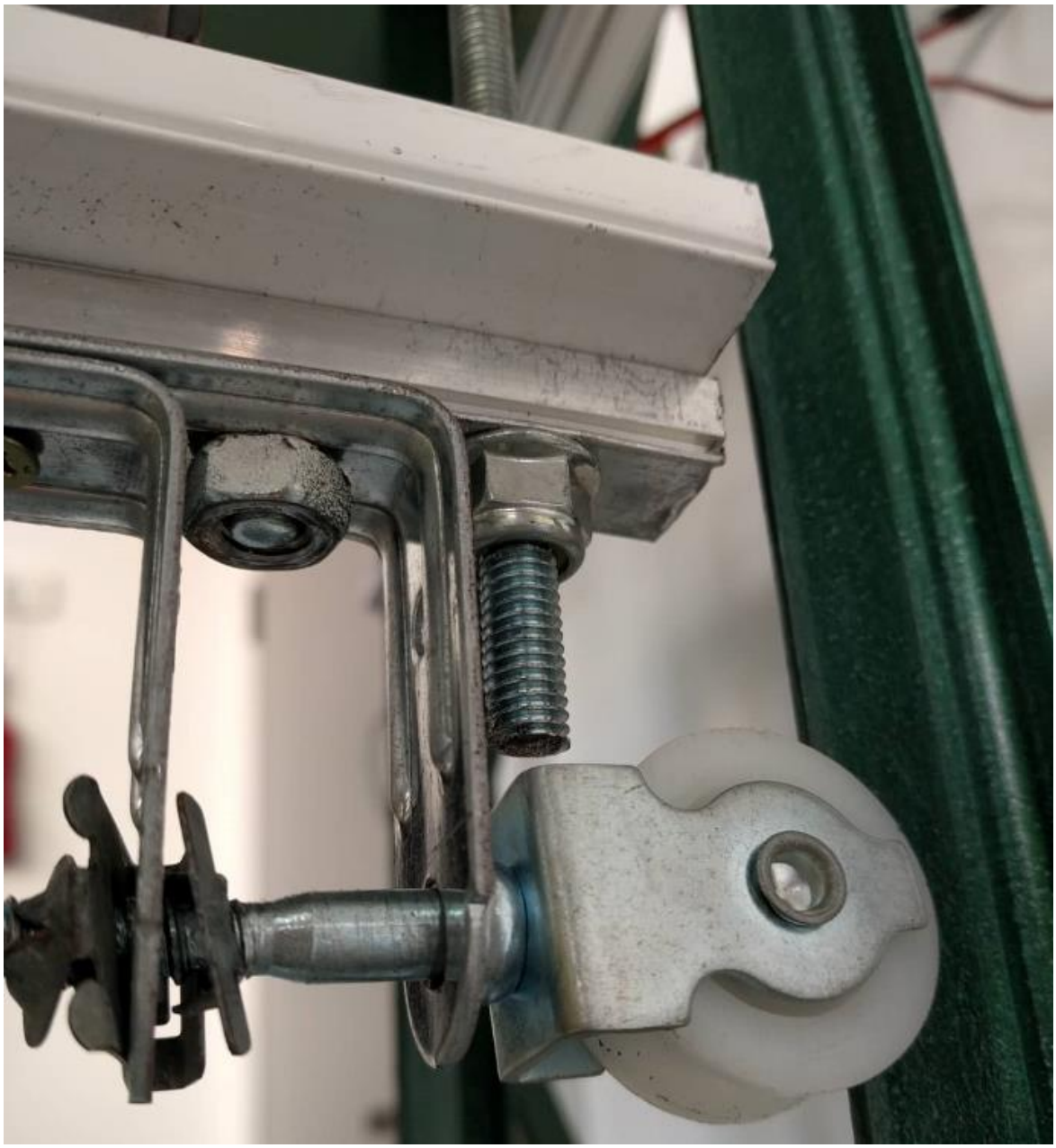
**Εικόνα 17 Λεπτομέρεια συρματόσχοινου - σφιγκτήρα - γάντζου**

### 4.1.3 Αντίβαρο

Ο μηχανισμός αντίβαρου της κατασκευής μας, αποτελείται από δυο αλουμινένιες βάσεις, παράλληλες στο έδαφος, μεταξύ των οποίων έχουν τοποθετηθεί τέσσερα βάρη της τάξεως του 1.5 κιλού έκαστος.

**Εικόνα 18 Μηχανισμός αντίβαρου**

Στις προαναφερθείσες μεταλλικές βάσεις, έχουν τοποθετηθεί επίσης ράουλα, με τις αντίστοιχες γωνίες ρύθμισής τους, όπως έχει γίνει και στην κατασκευή του θαλάμου, με ακριβώς ίδιο σκοπό ύπαρξης, τη διατήρηση ισορροπίας δηλαδή του αντιβάρου στον παράλληλο στο έδαφος άξονα κίνησης, όσο το αντίβαρο εκτελεί την κάθετη κίνησή του.



**Εικόνα 19 Λεπτομέρεια ράουλου στήριξης αντίβαρου**

Η κατασκευή του αντιβάρου περιλαμβάνει παξιμάδια ασφαλείας, με σκοπό την αποτροπή τυχαίων συμβάντων κατά τα οποία κάποιο μέρος του αντιβάρου λόγω της διαρκούς του κίνησης, χαλαρώσει και βγει από τη θέση του.

**Εικόνα 20 Λεπτομέρεια παξιμαδιού ασφαλείας αντίβαρου**

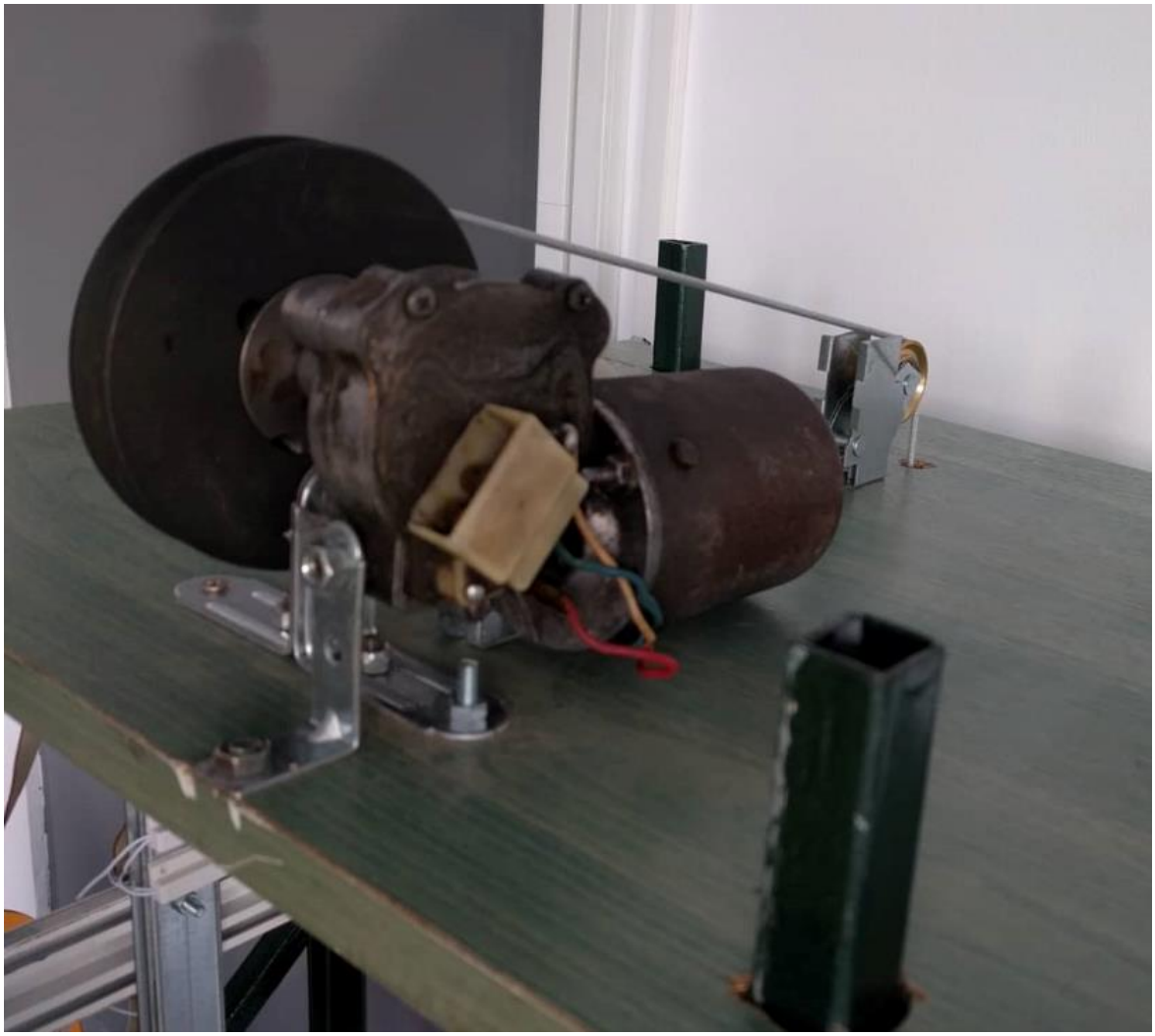
Όπως και στην κατασκευή του θαλάμου, το συρματόσχοινο είναι τοποθετημένο στην κορυφή του αντίβαρου, στο οποίο έχει τοποθετηθεί επίσης σφιγκτήρας, με σκοπό τη σίγουρη παραμονή του συρματόσχοινου στη θέση του.

### 4.1.4 Ηλεκτροκινητήρας

Με τον όρο ηλεκτροκινητήρας, εννοούμε την ηλεκτρική μηχανή η οποία έχει την ιδιότητα να μετατρέπει την ηλεκτρική ενέργεια σε μηχανική. Οι περισσότεροι ηλεκτροκινητήρες οφείλουν τη λειτουργία τους στην αλληλεπίδραση του μαγνητικού πεδίου του κινητήρα και του ηλεκτρικού ρεύματος. Ο συνδυασμός των δυο έχει ως αποτέλεσμα τη δημιουργία δύναμης σε μορφή περιστροφής. Οι ηλεκτροκινητήρες τροφοδοτούνται είτε από πηγή συνεχούς ρεύματος (DC) όπως μπαταρίες,

κινητήρες οχημάτων ή από  πηγή εναλλασσόμενου ρεύματος όπως ένα ηλεκτρικό δίκτυο (ΔΕΗ).

Για την κατασκευή μας, έχουμε επιλέξει έναν ηλεκτροκινητήρα ονομαστικής ισχύος 12V, οποίος προέρχεται από μηχανισμό υαλοκαθαριστήρων αυτοκινήτου.

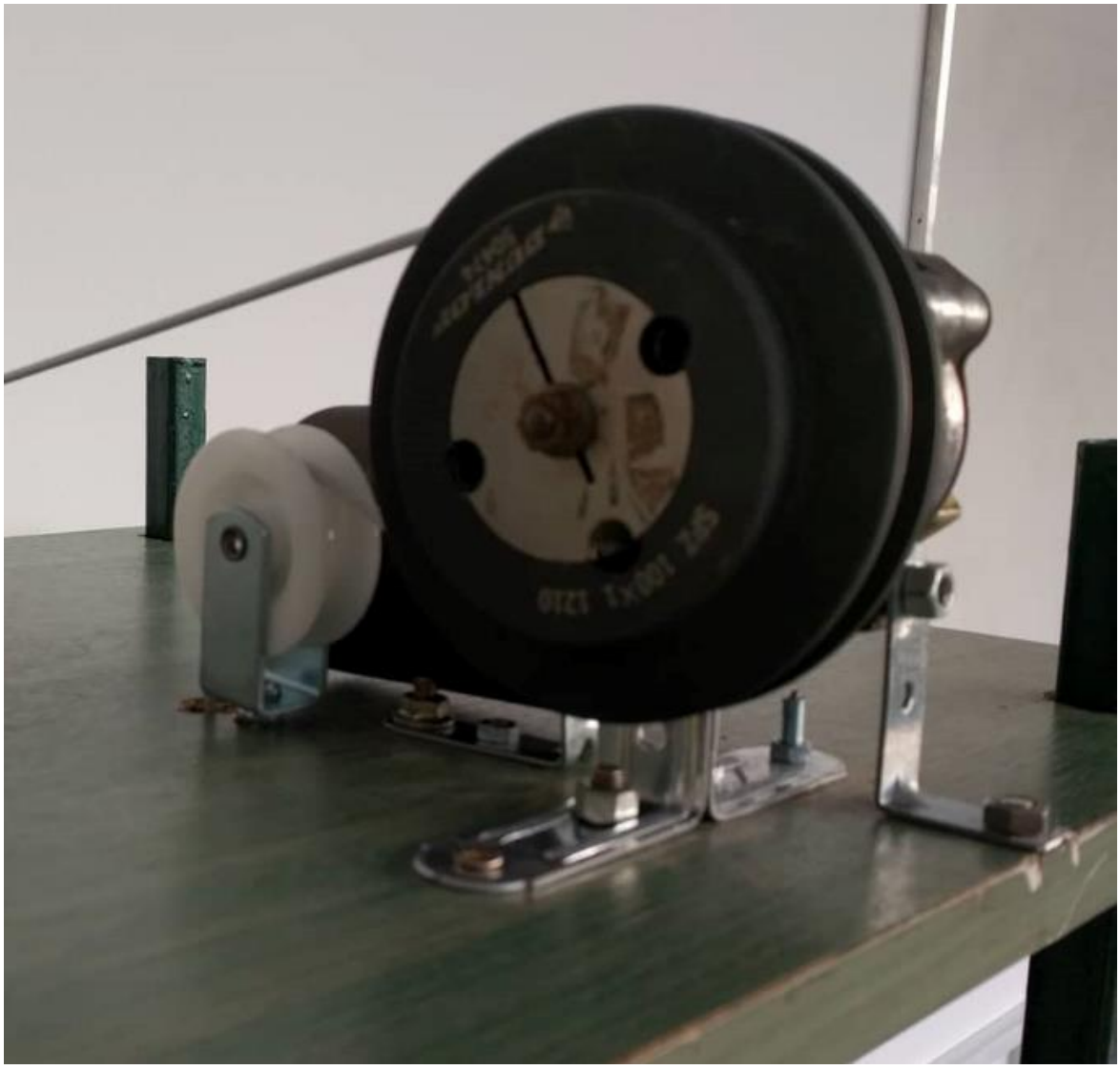


**Εικόνα 21 Ηλεκτροκινητήρας**

Ο λόγος που επιλέχθηκε ο συγκεκριμένος κινητήρας, είναι διότι περιλαμβάνει ατέρμονα άξονα.

Ο ατέρμων άξονας αποσκοπεί στην σταθεροποίηση του θαλάμου μέσω της προστασίας του από αντίρροπες προς την κίνησή του δυνάμεις, χάρη στην τάση που ασκείται στο συρματόσχοινο.

Στον άξονα κίνησης του ηλεκτροκινητήρα, έχει τοποθετηθεί τροχαλία, στης οποίας η επιφάνεια κίνησης έρχεται σε επαφή το επικαλυμμένο με PVC συρματόσχοινο, δίνοντας έτσι την απαραίτητη κίνηση στον συνδυασμό θαλάμου-αντίβαρου, με σκοπό την ορθή λειτουργία του ανυψωτικού συστήματος.



**Εικόνα 22 Λεπτομέρεια τροχαλίας κινητήρα**

Ο ηλεκτροκινητήρας είναι τοποθετημένος σε μια παράλληλη στο έδαφος ξύλινη επιφάνεια, η οποία βρίσκεται στην κορυφή του μεταλλικού σκελετού.

### 4.1.5 Εργαλεία

Τα εργαλεία που χρησιμοποιήθηκαν για τη συναρμολόγηση ποικίλουν από εργαλεία χειρός μέχρι ηλεκτρικά εργαλεία ισχύος.

Αρχικά για την κοπή των ξύλων για τον θάλαμο, χρησιμοποιήθηκε ηλεκτρικός τροχός ισχύος 1500watt καθώς και ηλεκτρική σέγα 700watt.

Για τις λοιπές τρύπες στον θάλαμο καθώς και στον μεταλλικό σκελετό, χρησιμοποιήθηκε ηλεκτρικό dremel 150watt.

Για την τοποθέτηση και σύσφιξη βιδών χρησιμοποιήθηκαν κατσαβίδια χειρός καθώς και ηλεκτρικό δραπανοκατσάβιδο ισχύος 20watt.

### 4.1.6 Συναρμολόγηση

Η συναρμολόγηση της κατασκευής ήταν μια σχετικά απλή διαδικασία. Ο μεταλλικός σκελετός ο οποίος αποτελείται από τις τέσσερεις μεταλλικές κολώνες και τις νευρώσεις, συγκολλήθηκε σε ειδικό εργαστήριο μεταλλουργικής.

Με το πέρας της κόλλησης των μεταλλικών στοιχείων μεταξύ τους, ξεκίνησε η κοπή των ξύλων έτσι ώστε να συναρμολογηθεί ο θάλαμος. Όλες οι πλευρές του θαλάμου είναι ξύλο μελαμίνης. Οι πλευρές αφού κολλήθηκαν μεταξύ τους με ειδική κόλλα ξύλου, στην συνέχεια τοποθετήθηκαν μεταλλικές γωνίες με την χρήση βιδών, έτσι ώστε η ξύλινη κατασκευή που συντελεί τον θάλαμο να παραμένει άρρηκτα ενιαία.

Ο θάλαμος είναι διαστάσεων 26cm * 25cm * 23cm (Πλάτος * Ύψος * Βάθος).

Στην κορυφή του θαλάμου έχει ανοιχθεί τρύπα στο κέντρο και έχει τοποθετηθεί μεταλλικός γάντζος ο οποίος συγκρατεί την άκρη του συρματόσχοινου.

Στην άνω παράλληλη πλευρά του θαλάμου και στο κέντρο του, έχουν τοποθετηθεί ρυθμιστικές μεταλλικές γωνίες, οι οποίες αποτελούν βάσεις για τα ράουλα ισορρόπησης του θαλάμου.

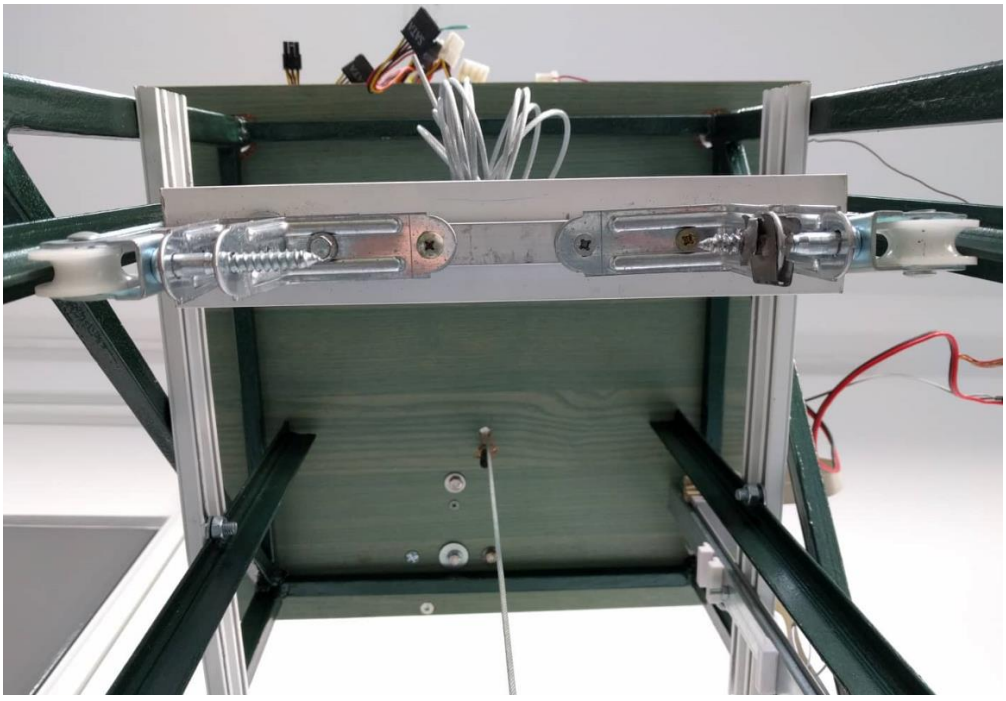
Η συναρμολόγηση του αντίβαρου ήταν επίσης μια σχετικά απλή διαδικασία. Ανάμεσα σε δυο μεταλλικές πλάκες, έχουν τοποθετηθεί τέσσερα βάρη της τάξεως του 1,5 κιλού έκαστος. Οι τέσσερις πλάκες στην συνέχεια είναι ενωμένες μεταξύ τους μεταλλικές ντίζες, οι οποίες με την σειρά τους είναι συσφιγμένες με τις μεταλλικές πλάκες με παξιμάδια ασφαλείας. Στις τέσσερις γωνίες του αντιβάρου και με κατεύθυνση προς τις δυο εσωτερικές πλευρές του μεταλλικού σκελετού, έχουν τοποθετηθεί με μεταλλικές ρυθμιστικές γωνίες ράουλα τα οποία αποσκοπούν στην

ισορροπία του αντιβάρου έτσι ώστε να μην αποκλίνει από τον παράλληλο προς το έδαφος άξονα κίνησης.

Στην συνέχεια, ανοίχθηκαν τρύπες στον μεταλλικό σκελετό, με την χρήση του dremel, οριζοντίως 3 σε κάθε πλευρά δεξιά και αριστερά του μεταλλικού σκελετού στην εσωτερική του μεριά, σε ύψος 26cm, 59cm, 92cm και 123cm από το έδαφος αντιστοίχως. Στις συγκεκριμένες τρύπες στην συνέχεια, βιδώθηκαν μεταλλικές πλάκες, οι οποίες αποτελούν βάσεις για την τοποθέτηση των μεταλλικών οδηγών και της βάσης των αισθητήρων θέσης.

Στις μεταλλικές βάσεις που έχουν τοποθετηθεί στις 2 εσωτερικές πλευρές του μεταλλικού σκελετού, συγκεκριμένα στην πρώτη και στην τελευταία, έχουν τοποθετηθεί βίδες με παξιμάδια, στις οποίες στην συνέχεια βιδώθηκαν οι μεταλλικοί δρόμοι, κάθετοι στο έδαφος, στους οποίους κινούνται τα ράουλα του θαλάμου και του αντιβάρου αντιστοίχως.

Στην αρχή των μεταλλικών λαμών, έχουν ανοιχθεί τρύπες με σκοπό την τοποθέτηση μεταλλικής βέργας, η οποία αποτελεί βάση για την τοποθέτηση μαγνητικών αισθητήρων θέσης θαλάμου.



**Εικόνα 23 Κάτοψη αντίβαρου**

**Εικόνα 24 Ολοκληρωμένη κατασκευή ανελκυστήρα**

# ΚΕΦΑΛΑΙΟ 5

Στο παρόν κεφάλαιο γίνεται μια μικρή επεξήγηση του ηλεκτρολογικού μέρους της κατασκευής, το οποίο είναι έναν απλό ηλεκτρολογικό κομμάτι από μια ασφάλεια, κλέμες και καλώδια. Επίσης αναλύεται το ηλεκτρονικό μέρος την κατασκευής, το οποίο αποτελείται από τα βασικά ηλεκτρονικά κυκλώματα που οδηγούν την κατασκευή στην ορθή λειτουργία της.

## 5.1 Ηλεκτρολογικό μέρος

### 5.1.1 Ασφάλεια

Η ασφάλεια της κατασκευής μας είναι μια διπλή ασφάλεια ράγας αποκοπής φάσης και ουδετέρου.


**Εικόνα 25 Ασφάλεια Siemens 40Amps**

### 5.1.2 Κλέμες

Οι κλέμες της κατασκευής μας χρησιμοποιούνται για την τακτοποίηση των καλωδίων.

### 5.1.3 Τροφοδοτικό

Το τροφοδοτικό της κατασκευής είναι ένα τροφοδοτικό φορητού υπολογιστή 15V μεταβλητής τάσης εξόδου.

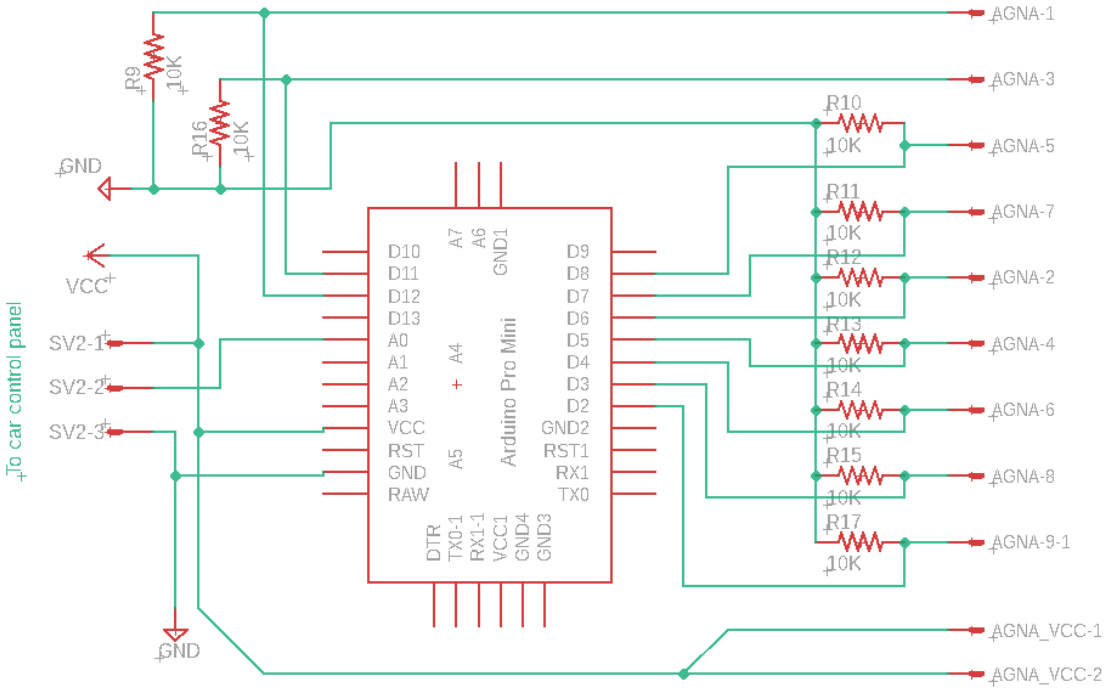

**Εικόνα 26 Τροφοδοτικό μεταβλητής τάσης Hama**

## 5.2 Ηλεκτρονικό μέρος

Το ηλεκτρονικό μέρος της κατασκευής μας αποτελείται από την βαθμίδα διαχείρισης (ΨΣΑΕ), την βαθμίδα εντολοδότησης (κουμπιά κλήσης ορόφων) και την βαθμίδα ρευμάτων.

### 5.2.1 Βαθμίδα διαχείρισης

Η βαθμίδα διαχείρισης ουσιαστικά αποτελεί το ΨΣΑΕ μας. Το βασικό στοιχείο του ΨΣΑΕ μας είναι το Arduino. Επιπλέον, η βαθμίδα διαχείρισης αποτελείται από αντιστάσεις και μαγνητικούς διακόπτες/επαφές. Οι αντιστάσεις υπάρχουν με σκοπό την ρύθμιση της τάσης και της ροής του ρεύματος έτσι ώστε να επιτυγχάνεται αποκοπή θορύβου στα σήματα που κινούνται προς τις εισόδους του Arduino.

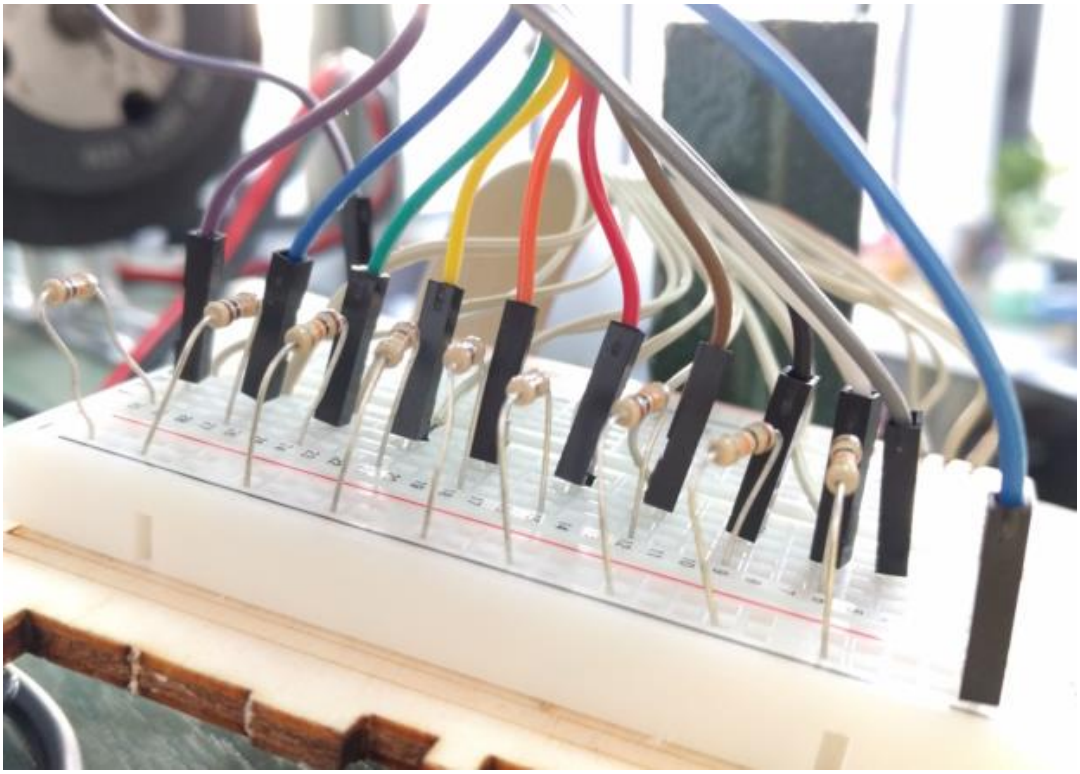



**Εικόνα 27Κύκλωμα Arduino**

Το κύκλωμα διαχείρισης ουσιαστικά λαμβάνει σήματα από δυο πηγές. Η πρώτη πηγή είναι το κύκλωμα (ή βαθμίδα) εντολοδότησης (κουμπιά κλήσης ανελκυστήρα) και η δεύτερη πηγή είναι ειδικά τοποθετημένοι σε μεταλλική ράβδο κάθετη στο έδαφος μαγνητικοί διακόπτες που λειτουργούν σαν αισθητήρες θέσης θαλάμου.

**Εικόνα 28 Μαγνητικοί διακόπτες**

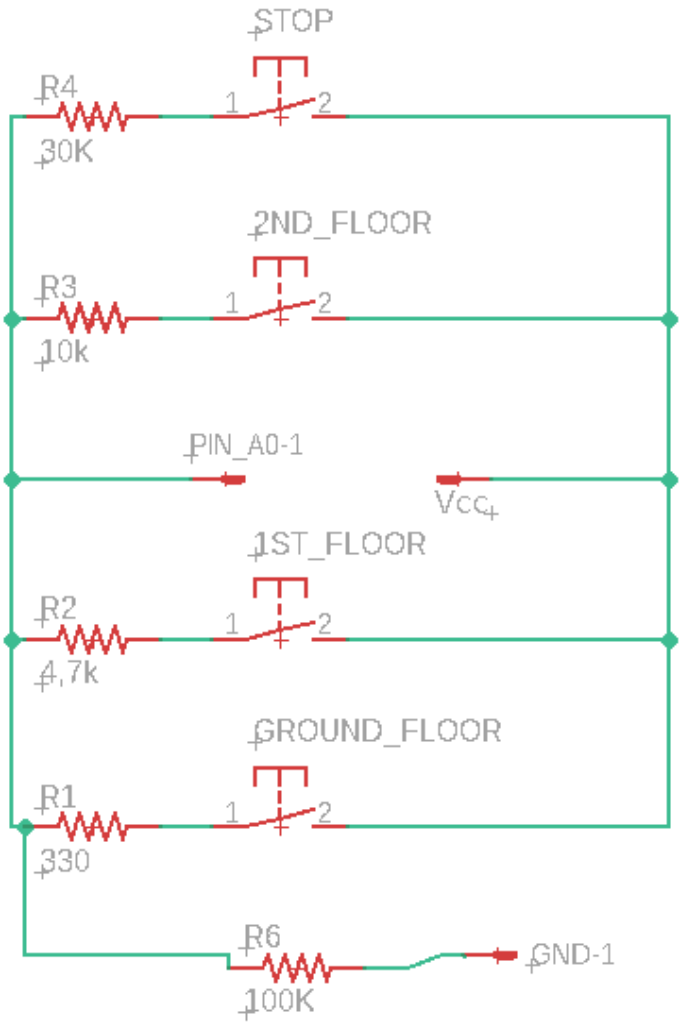
Οι μαγνητικοί διακόπτες δίνουν σήμα στους ακροδέκτες AGNA του κυκλώματος, και ενεργοποιούνται με σκοπό την σηματοδότηση του κυκλώματος έτσι ώστε το ΨΣΑΕ μας να «αναγνωρίζει» πότε ο θάλαμος βρίσκεται σε κάποιο επίπεδο. Η δεύτερη πηγή σήματος του ΨΣΑΕ, είναι η βαθμίδα εντολοδότησης (κουμπιά κλήσης) τα οποία ουσιαστικά με την ενεργοποίησή τους, οδηγούν το ΨΣΑΕ το οποίο με την σειρά του οδηγεί την βαθμίδα διαχείρισης κινητήρα προς ρευματοδότηση τέτοια ώστε να κινήσει ο κινητήρας την τροχαλία συρματόσχοινου προς την κατάλληλη κατεύθυνση από την οποία εκλήθη ο θάλαμος.



**Εικόνα 29 Λεπτομέρεια κυκλώματος διαχείρησης μαγνητικών διακοπτών**

### 5.2.2 Βαθμίδα εντολοδότησης

Η βαθμίδα εντολοδότησης αποτελείται από κουμπιά (buttons) κλήσης του θαλάμου στα διάφορα επίπεδα της κατασκευής και από αντιστάσεις.

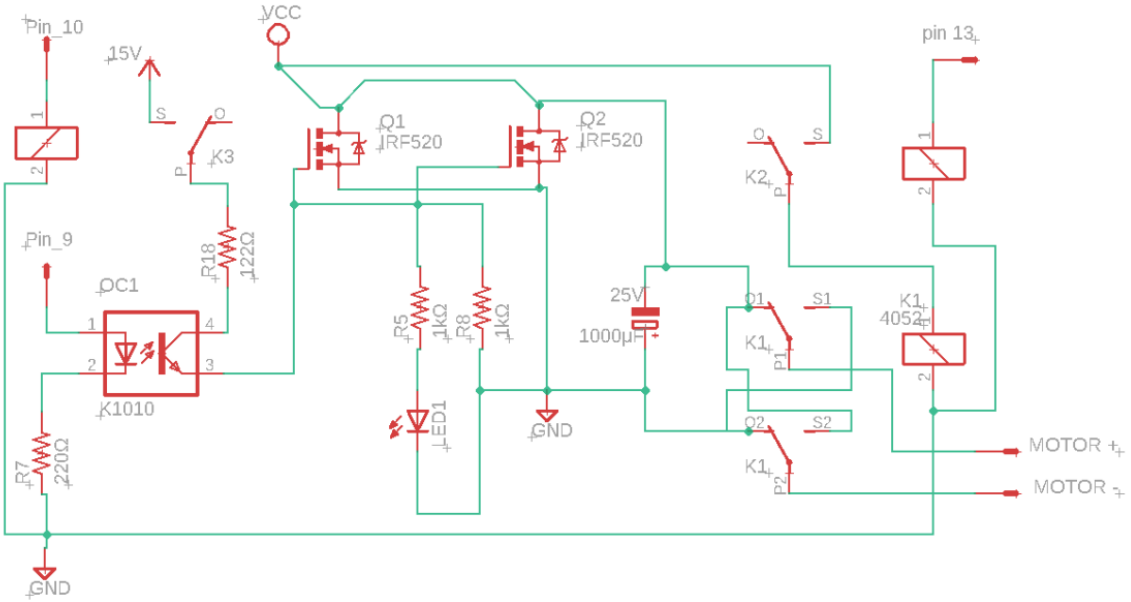



**Εικόνα 30 Κύκλωμα κουμπιών κλήσης θαλάμου**

Η βαθμίδα διαχείρισης κλήσεως θαλάμου, αποτελείται από αντιστάσεις και ηλεκτρονικά στοιχεία τύπου “button” (κουμπιά). Με την ενεργοποίησή τους (πάτημα), στέλνεται σήμα της τάξεως των 0-5V στο Arduino με σκοπό την επιλογή ορόφου. Ουσιαστικά η είσοδος του Arduino λαμβάνει το σήμα, το οποίο μετέπειτα μετατρέπεται σε ψηφιακή πληροφορία όπου στην συνέχεια επεξεργάζεται το Arduino με βάση τον κώδικά του, έτσι ώστε να οδηγήσει πάλι την βαθμίδα διαχείρισης κινητήρα στην αντίστοιχη απαραίτητη ηλεκτροδιοδώτηση για την ανάλογη κίνηση της τροχαλίας.

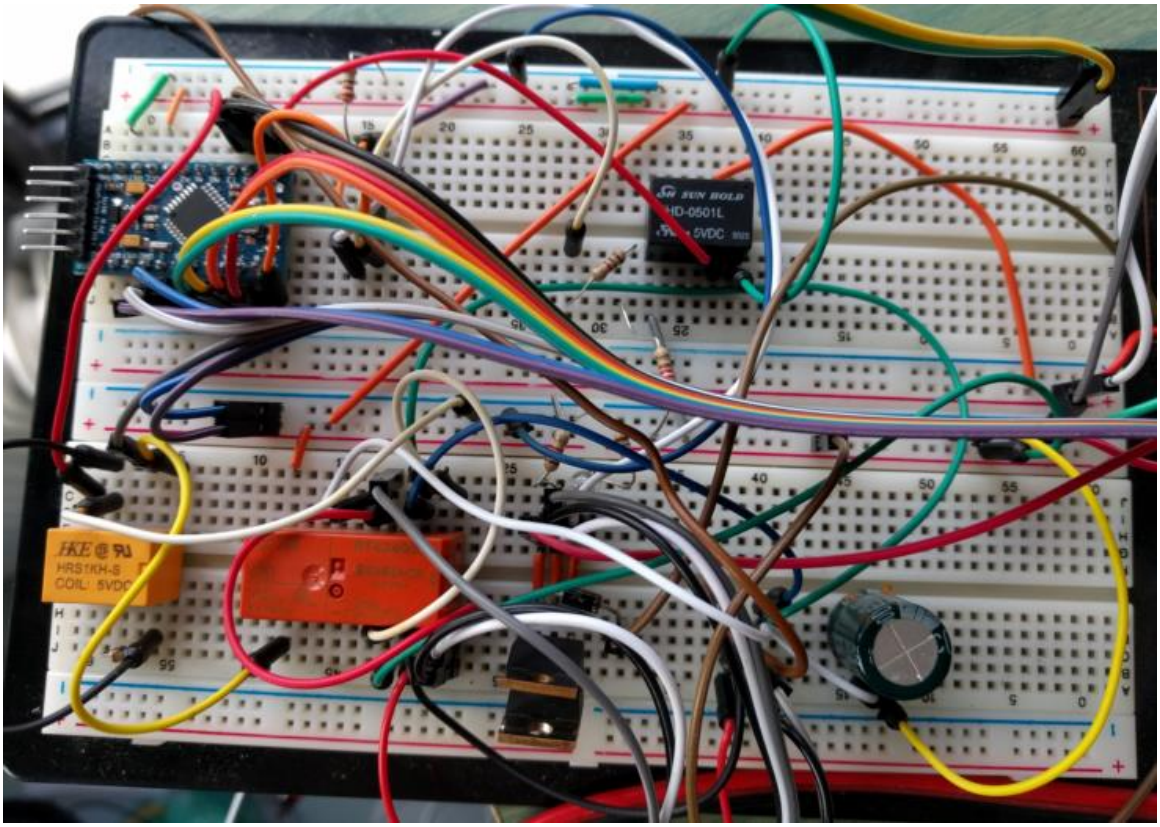
### 5.2.3 Βαθμίδα οδήγησης κινητήρα

Η βαθμίδα οδήγησης κινητήρα αποτελείται από ρελέ, τρανζίστορ και φωτοτρανζίστορ.




**Εικόνα 31 Κύκλωμα διαχείρισης κινητήρα**

Το κύκλωμα διαχείρισης του κινητήρα λαμβάνει ως είσοδο 0-5V στον optocoupler, ο οποίος στην συνέχεια στέλνει το σήμα στο τρανζίστορ και το τρανζίστορ στο αντίστοιχο ρελέ το οποίο θα τροφοδοτήσει τον κινητήρα με την απαραίτητη τάση για να κινηθεί με φορά τέτοια έτσι ώστε να κινήσει τον θάλαμο προς την αντίστοιχη θέση κλήσης του.




**Εικόνα 32 Κυκλώματα διαχείρισης κινητήρα - Arduino σε breadboard**

# ΚΕΦΑΛΑΙΟ 6

Σε αυτό το κεφάλαιο θα γίνει μια μικρή εισαγωγή σχετικά με την γλώσσα Wiring, επεξήγηση του τρόπου προγραμματισμού του Arduino μας, έτσι ώστε να αποτελέσει το ΨΣΑΕ της κατασκευής μας, οι βιβλιοθήκες που χρησιμοποιήθηκαν στον κώδικά μας, καθώς και επεξήγηση του ίδιου του κώδικα.

## 6.1 Προγραμματιστικό μέρος

### 6.1.1 Η γλώσσα Wiring

Η Wiring είναι μια γλώσσα προγραμματισμού ανοιχτού κώδικα η οποία χρησιμοποιείται για την δημιουργία framework(πλαίσιο) για μικροελεγκτές.
Η Wiring δημιουργήθηκε από τον Hernando Barragan στην Ιβρέα της Ιταλίας.
Η Wiring επιτρέπει την δημιουργία λογισμικού το οποίο μπορεί να λειτουργεί σε διαφορετικές πλατφόρμες, με σκοπό τον έλεγχο συσκευών οι οποίες είναι συνδεδεμένες σε ένα ευρύ φάσμα από διάφορες πλακέτες μικροελεγκτών, με σκοπό την δημιουργία όλων των ειδών δημιουργικού κώδικα, διαδραστικών αντικειμένων ή επιστημονικών εμπειριών. Οι δομές είναι σοφιστικά δημιουργημένες συμπεριλαμβάνοντας σχεδιαστές και καλλιτέχνες καθώς αποσκοπεί στο να ενθαρρύνει μια κοινότητα όπου ειδικοί του κλάδου μπορούν να μοιραστούν ιδέες με αρχάριους, γνώση καθώς και την συνολική τους εμπειρία. Υπάρχουν χιλιάδες μαθητές, καλλιτέχνες, σχεδιαστές, ερευνητές και χομπίστες οι οποίοι χρησιμοποιούν την Wiring για εκπαιδευτικούς σκοπούς, πρωτότυπες κατασκευές άλλα και ολοκληρωμένα επαγγελματικά έργα.[19]

Τα βασικά προτερήματα της Wiring είναι

- Ευρύς χάρτης οδηγιών για πολλαπλό υλισμικό πυρήνων διαφορετικής αρχιτεκτονικής
- Η τωρινή έκδοση AVR8 υποστηρίζει τον προγραμματισμό οποιουδήποτε υλικού που στηρίζεται στους επεξεργαστές AVR atmega
- Η απλή ενσωμάτωση υλισμικού κατηγορίας atmel τρίτων
- Δωρεάν παροχή μέσω του διαδικτύου, ανοιχτού κώδικα και ανοιχτού υλισμικού
- Για λειτουργικά προγράμματα όπως Windows, Linux, MacOS

- Πάνω από 100 διαφορετικές βιβλιοθήκες επέκτασης λογισμικού
- Ευρέως τεκμηριωμένο με πάρα πολλά βιβλία και οδηγούς διαθέσιμα

### 6.1.3 Επεξήγηση κώδικα

Όπως προαναφέρθηκε λοιπόν, η γλώσσα Wiring είναι μια γλώσσα βασισμένη στην C. Οι κυριότερες διαφορές που έχει από την C είναι η ύπαρξη ενός συνόλου εντολών που αφορούν συγκεκριμένα λειτουργίες του Arduino ως πλατφόρμα. Ξεκινώντας αφαιρετικά η δομή περιλαμβάνει δυο βασικές μεθόδους που είναι ικανή και αναγκαία συνθήκη για να εκτελεστεί ο κώδικας σωστά κατά την λήξη του προγραμματισμού και της επανεκκίνησης του Arduino.

sketch_nov12a

```
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

**Εικόνα 33 Οι δύο βασικές μέθοδοι, απαραίτητες για την ομαλή λειτουργεία του Arduino**

Αξίζει να αναφερθεί ότι ο κώδικας της συγκεκριμένης εργασίας περιέχει σχολιασμούς που αφορούν τους ρόλους σταθερών, μεταβλητών, περιπτώσεων, μεθόδων καθώς και εντολών.

```
/*
    Electronic Lift's master firmware version 1.1107a
    Georgantas Philip, 40266
    Perlianis Angelos, 40747
*/
//Strings
const String en_initializing = "Initializing...";
const String en_ready = "Ready";
const String en_stop_pressed = "Stop button pressed!";

static const uint8_t pin_enable_motor_power = 10; //Arduino sign
static const uint8_t pin_floor_select = A0;       //Arduino sign
static const uint8_t pin_emergency_top = 2;       //Arduino sign
static const uint8_t pin_emergency_bottom = 3;    //Arduino sign
static const uint8_t pin_2floor_stop = 4;         //Arduino sign
static const uint8_t pin_2floor_reach_under = 5;  //Arduino sign
static const uint8_t pin_1floor_reach_over = 6;   //Arduino sign
static const uint8_t pin_1floor_stop = 7;         //Arduino sign
static const uint8_t pin_1floor_reach_under = 8;  //Arduino sign
static const uint8_t pin_0floor_reach_over = 11;  //Arduino sign
static const uint8_t pin_0floor_stop = 12;        //Arduino sign
static const uint8_t pin_motor_speed = 9;         //Arduino sign
static const uint8_t pin_motor_direction = 13;    //Arduino sign

//Floor selection to motor direction calculation
uint8_t floor_select[5];
uint8_t floor_index = 0; //0 is the reset value to bring the car

uint8_t current_floor_selected = 0;
boolean car_stop_pressed = 0; //0 = Normal 1 = Pressed

//Automated error control variables
boolean car_error_state = 0; // 0 = Normal 1 = Error state

//Speed variable
uint16_t _speed = 0;
```

**Εικόνα 34 Δηλώσεις σταθερών και μεταβλητών εμβέλειας όλου του κώδικα**

Οι μεταβλητές τύπου String χρησιμεύουν στην συγκέντρωση των μηνυμάτων που αφορούν την ετοιμότητα του Arduino κατά την εκκίνηση και ετοιμότητα του Arduino-ανελκυστήρα για εκτέλεση εντολής.

Οι τιμές των σταθερών που ξεκινάνε με το πρόθεμα “*pin_*” αφορούν την εκμετάλλευση του υλισμικού του Arduino και του προσαρμοσμένου σε αυτό υλικού. Η πρώτη σταθερά αφορά την ενεργοποίηση ή μη των ρευμάτων (τάξεως 15V) προς τη βαθμίδα οδήγησης του κινητήρα. Η δεύτερη αφορά την είσοδο σημάτων που ο ενσωματωμένος ADC μετατρέπει το εύρος τάσης 0 έως 5V σε ακέραιες τιμές εύρους από 0 έως 1023 (σύνολο 1024 τιμές). Οι επόμενες εννέα (9) δηλώσεις αφορούν την εκμετάλλευση των μαγνητικών επαφών που είναι προσαρμοσμένες στο Arduino. Η προτελευταία δήλωση αφορά την εκμετάλλευση του optocoupler που οδηγεί την ταχύτητα του κινητήρα. Η τελευταία αφορά την εκμετάλλευση του ηλεκτρομαγνητικού διακόπτη (ρελέ) που ρυθμίζει την διεύθυνση περιστροφής του άξονα του κινητήρα. Οι μεταβλητές που περιλαμβάνουν το πρόθεμα “*floor_*” αφορούν τον υπολογισμό προς τα που θα κινηθεί ο θάλαμος σε περίπτωση κλήσης του σε κάποιον όροφο. Οι μεταβλητές που ξεκινάνε με το πρόθεμα “*car_*” αφορούν την κατάσταση του θαλάμου – αν βρίσκεται σε σφαλματική θέση – ή αν ο επιβάτης πάτησε ενώ ήταν εν κινήσει το κουμπί επείγουσας ακινητοποίησης του θαλάμου. Σημειώνουμε ότι οι σφαλματικές θέσεις του θαλάμου είναι αυτές που βρίσκονται πάνω από τον δεύτερο όροφο και κάτω

από το ισόγειο. Η μεταβλητή _speed είναι αυτή που χρησιμοποιείται στις εντολές ρύθμισης ταχύτητας.

```
void setup() {
  //Serial port initialization
  Serial.begin(9600);
  Serial.println(en_initializing);

  //Set of the pins appropriate role (I/O) in the project.
  pinMode(pin_enable_motor_power, OUTPUT);
  pinMode(pin_motor_speed, OUTPUT);
  pinMode(pin_motor_direction, OUTPUT);
  pinMode(pin_emergency_top, INPUT);
  pinMode(pin_emergency_bottom, INPUT);
  pinMode(pin_2floor_stop, INPUT);
  pinMode(pin_2floor_reach_under, INPUT);
  pinMode(pin_1floor_reach_over, INPUT);
  pinMode(pin_1floor_stop, INPUT);
  pinMode(pin_1floor_reach_under, INPUT);
  pinMode(pin_0floor_reach_over, INPUT);
  pinMode(pin_0floor_stop, INPUT);

  //Maintainance check for any wrong signals from the pins. Value depends on the position of the car.
  checkpins();
  digitalWrite(pin_enable_motor_power, HIGH);

  if (!digitalRead(pin_0floor_stop))
    Serial.println("Reseting car's position to ground floor");
  //    Serial.println("Car not in ground floor... Trying to detect if it is at the bottom of the shaft");
  if (!digitalRead(pin_emergency_bottom) and !digitalRead(pin_0floor_stop)) {
    if (!digitalRead(pin_emergency_top))
    {
      digitalWrite(pin_motor_direction, HIGH);
      analogWrite(pin_motor_speed, 180);
      delay(500);
      digitalWrite(pin_motor_speed, LOW);
      //    Serial.println("Car was at the bottom of the shaft.");
    }
    digitalWrite(pin_motor_direction, LOW);
  }
  else
    digitalWrite(pin_motor_direction, HIGH);

  while (!digitalRead(pin_0floor_stop))
    analogWrite(pin_motor_speed, 180);
  digitalWrite(pin_motor_speed, LOW);

  digitalWrite(pin_motor_direction, HIGH);
  digitalWrite(pin_enable_motor_power, LOW);
  Serial.println(en_ready);
}
```

**Εικόνα 35 Μέθοδος setup()**

Εδώ έχουμε την μέθοδο “*void setup()*”. Την μέθοδο που εκτελείται μία και μοναδική φορά στο κύκλο λειτουργίας του Arduino μαζί με τις δηλώσεις που προαναφέρθηκαν.

Εδώ ορίζουμε την έναρξη της εκμετάλλευσης της σειριακής εισόδου για επικοινωνία με τον υπολογιστή εν προκειμένω που θα παρακολουθεί και να εντολοδοτεί τον ανελκυστήρα.

Παρακάτω με την κλήση της εσωτερικής συνάρτησης “*pinMode*” ορίζουμε ρόλους στα pins που αναφέρονται οι τιμές των σταθερών με πρόθεμα “*pin_*”. Οι ρόλοι αυτοί είναι εισόδου/εξόδου.
Η κλήση της συνάρτησης “*checkpins()*”, εδώ, γίνεται για να δούμε κατά την εκκίνηση του Arduino που βρίσκεται ο θάλαμος και αν επαφίεται με κάποιον όροφο.
Οι υπόλοιπες εντολές που ακολουθούν αφορούν την μεταφορά του θαλάμου στο ισόγειο. Ουσιαστικά εδώ εκτελείται αρχικοποίηση της θέσης του ανελκυστήρα.
Τελειώνοντας, λοιπόν την αρχικοποίηση στέλνεται στην έξοδο της σειριακής θύρας το σήμα “Ready”.
Από αυτό το σημείο και μετά ο ανελκυστήρας είναι έτοιμος για λειτουργία.

```
87  void loop() {
88    int current_floor_selected = check_buttons();
89    switch (current_floor_selected) {
90      case 1: { //Call to ground floor
91          Serial.print("Floor call: ");
92          Serial.println(current_floor_selected);
93          Serial.println(car_error_state);
94          Serial.println(car_stop_pressed);
95          digitalWrite(pin_motor_speed, LOW);
96          digitalWrite(pin_enable_motor_power, HIGH);
97          digitalWrite(pin_motor_direction, LOW); //Turn direction to down
98          if (car_error_state or car_stop_pressed or digitalRead(pin_0floor_stop))
99            break;
100         else
101           accel(pin_0floor_stop);
102         Serial.println(digitalRead(pin_0floor_reach_over));
103         while (!digitalRead(pin_0floor_reach_over) and (!car_error_state)) {
104           if ((emergency_check() == -1) or (check_buttons() == 4))
105             break;
106         }
107         if (car_error_state or car_stop_pressed)
108           break;
109         else {
110           deccel(pin_0floor_stop);
111           while (!digitalRead(pin_0floor_stop) and (!car_error_state)) {
112             if ((emergency_check() == -1) or (check_buttons() == 4))
113               break;
114           }
115           //        if (car_error_state or car_stop_pressed)
116           //          break;
117           digitalWrite(pin_motor_direction, HIGH);
118           digitalWrite(pin_motor_speed, LOW);
119           digitalWrite(pin_enable_motor_power, LOW);
120           Serial.println("Stopped!");
121           break;
122         }
```

**Εικόνα 36 Μέθοδος loop()**

Η “*void loop()*” είναι η δεύτερη βασική μέθοδος του Arduino. Η συγκεκριμένη εκτελείται σε μια ατέρμονη επανάληψη σαν να περικλείεται μια while(1).
Εδώ σε κάθε επανάληψη εκτελούνται δυο βασικές εργασίες. Η πρώτη είναι να κληθεί η συνάρτηση check_buttons που ελέγχει το αν υπάρχει κάποιο αίτημα προς υλοποίηση. Πρόκειται όμως μόνο για κλήσεις του θαλάμου σε κάποιον όροφο ή την απόκριση στο κουμπί έκτακτης ακινητοποίησης του θαλάμου. Συνεπώς στην συνθήκη που ακολουθεί τύπου switch αφορά αυτά τα αιτήματα.
Σε κάθε περίπτωση (case) της switch, πλην της τελευταίας, οι εντολές ακολουθούν ένα μοτίβο.

1. Ενημέρωση μέσω σειριακής.
2. Ενεργοποίηση ρευμάτων προς κινητήρα, εναλλαγή μεταξύ ανόδου και καθόδου και χειροκίνητη ανάθεση LOW στο pin ταχύτητας και έλεγχοι θέσεως και κατάστασης θαλάμου.
3. Επιτάχυνση με κλήση της “*accel()*”.
4. Συνεχής έλεγχος αν πλησιάζει ο θάλαμος στον όροφο-προορισμό καθώς και έλεγχος αν υπάρχει κάποια εντολή από τη σειριακή είσοδο ή αν έχει φτάσει ο θάλαμος στα όρια του φρεατίου. Ο τελευταίος έλεγχος γίνεται στα πλαίσια πρόληψης σφάλματος ενδιάμεσων αισθητήρων.
5. Επιβράδυνση με κλήσης της “*deccel()*”.
6. Σταμάτημα.
7. Απενεργοποίηση ρευμάτων προς κινητήρα.
8. Ενημέρωση μέσω σειριακής.

Στη τελευταία περίπτωση της case ουσιαστικά απλά ενημερώνουμε τη σειριακή ότι ο θάλαμος είναι ήδη σταματημένος εφόσον ο έλεγχος για την έκτακτη ακινητοποίηση του θαλάμου γίνεται ενδιαμέσως της κίνησης. Στη τελευταία περίπτωση της case ουσιαστικά απλά ενημερώνουμε τη σειριακή ότι ο θάλαμος είναι ήδη σταματημένος εφόσον ο έλεγχος για την έκτακτη ακινητοποίηση του θαλάμου γίνεται ενδιαμέσως της κίνησης.

```
123     case 2: { //Call to 1st floor
124         Serial.print("Floor call: ");
125         Serial.println(current_floor_selected);
126         digitalWrite(pin_enable_motor_power, HIGH);
127         if (car_error_state or car_stop_pressed)
128           break;
129         else
130           accel(pin_1floor_stop);
131         while (!digitalRead(pin_1floor_reach_over) and !digitalRead(pin_1floor_reach_under) and (!car_error_state)) {
132           if ((emergency_check() == -1) or (check_buttons() == 4))
133             break;
134         }
135         if (car_error_state or car_stop_pressed)
136           break;
137         else
138           deccel(pin_1floor_stop);
139         while (!digitalRead(pin_1floor_stop) and (!car_error_state)) {
140           if ((emergency_check() == -1) or (check_buttons() == 4))
141             break;
142         }
143         if (car_error_state or car_stop_pressed)
144           break;
145         else
146           digitalWrite(pin_motor_speed, LOW);
147         digitalWrite(pin_enable_motor_power, LOW);
148         Serial.println("Stopped!");
149         break;
150       }
```

**Εικόνα 37 Μέθοδος loop() - switch_case_2**

```
151       case 3: { //Call to 2nd floor
152           Serial.print("Floor call: ");
153           Serial.println(current_floor_selected);
154           //Turn direction to up
155           digitalWrite(pin_motor_speed, LOW);
156           digitalWrite(pin_enable_motor_power, HIGH);
157           digitalWrite(pin_motor_direction, HIGH);
158           if (car_error_state or car_stop_pressed)
159             break;
160           else
161             accel(pin_2floor_stop);
162           while (!digitalRead(pin_2floor_reach_under) and (!car_error_state)) {
163             if ((emergency_check() == -1) or (check_buttons() == 4))
164               break;
165           }
166           if (car_error_state or car_stop_pressed)
167             break;
168           else
169             deccel(pin_2floor_stop);
170           while ((!digitalRead(pin_2floor_stop)) and (!car_error_state)) {
171             if ((emergency_check() == -1) or (check_buttons() == 4))
172               break;
173           }
174           digitalWrite(pin_motor_speed, LOW);
175           digitalWrite(pin_motor_direction, LOW);
176           digitalWrite(pin_enable_motor_power, LOW);
177           Serial.println("Stopped!");
178           break;
179         }
180       case 4: { //Stop button
181           //Turn direction to down
182           Serial.println("Allready stopped!");
183           break;
184         }
185       }
```

**Εικόνα 38 Μέθοδος loop() - switch_case_3_4**

Σε αυτό το σημείο τελειώνει η "*void loop()*" με την εντολή απενεργοποίησης ταχύτητας του κινητήρα καθώς και αρχικοποίηση των μεταβλητών κατάστασης του θαλάμου καθώς ενημερώνονται διαρκώς από τους συνεχείς ελέγχους.

```
187   digitalWrite(pin_motor_speed, LOW);
188   car_stop_pressed = false;
189   car_error_state = false;
190 }
```

**Εικόνα 39 Μέθοδος loop() - υπόλοιπο εντολών**

```
192 void accel(int floor_selected_stop) {
193   Serial.println("Starting acceleration");
194   for (_speed = 0; _speed <= 255; _speed++) {
195     analogWrite(pin_motor_speed, _speed);
196     if ((emergency_check() == -1) or (check_buttons() == 4))
197       break;
198     delay(5);
199   }
200   Serial.println("Finished acceleration...");
201 }
```

Η συνάρτηση επιτάχυνσης χρησιμοποιεί μια επανάληψη ορισμένων βημάτων τύπου for για να ρυθμίσει την ταχύτητα καθώς παράλληλα εκτελεί ελέγχους. Στην αρχή και στο τέλος της ενημερώνει τη σειριακή θύρα.

Η συνάρτηση επιβράδυνσης λειτουργεί ανάλογα με την συνάρτηση επιτάχυνσης με τη διαφορά ότι έχει μεταβαλλόμενη ορισμένη ταχύτητα ανάλογα με την πορεία του θαλάμου.

```
204 void deccel(int floor_selected) {
205   Serial.println("Starting decceleration...");
206   int y;
207   if (digitalRead(pin_motor_direction) == LOW)
208     y = 85;
209   else //To be changed with weight detection...?how?
210     y = 85;
211   for (_speed--; _speed >= y; _speed--) {
212     analogWrite(pin_motor_speed, _speed);
213     if ((emergency_check() == -1) or (check_buttons() == 4))
214       break;
215     delay(3);
216   }
217   Serial.println("Finished decceleration");
218 }
```

**Εικόνα 40 Μέθοδος deccel()**

```
220 void stop_pressed() {
221   digitalWrite(pin_motor_speed, LOW);
222   Serial.println(en_stop_pressed);
223   car_stop_pressed = true;
224 }
```

**Εικόνα 41 Μέθοδος stop_pressed()**

Η συνάρτηση “*stop_pressed()*” στέλνει άμεσα σήμα απενεργοποίησης ταχύτητας και ενημερώνει την μεταβλητή σήμανσης έκτακτης ακινητοποίησης.

```
226 int check_buttons() {
227   int t = analogRead(pin_floor_select);
228   if (Serial.available()) {
229     Serial.print("Serial available: ");
230     int x = Serial.parseInt(); //Check Serial for any remote command
231     Serial.println(x);
232     if ((x >= 1) && (x <= 4))
233     {
234       if (x == 4)
235         car_stop_pressed = true;
236       digitalWrite(pin_motor_speed, LOW);
237       return x;
238     }
239     else if ( x == 41)
240     {
241       Serial.println("Maintainance: Recovering car position to ground floor");
242       digitalWrite(pin_motor_direction, HIGH);
243       while (digitalRead(pin_emergency_bottom))
244         digitalWrite(pin_motor_speed, HIGH);
245       digitalWrite(pin_motor_speed, LOW);
246     } else if ( x == 43)
247     {
248       Serial.println("Maintainance: Recovering car position to 2nd floor");
249       digitalWrite(pin_motor_direction, LOW);
250       while (digitalRead(pin_emergency_top))
251         digitalWrite(pin_motor_speed, HIGH);
252       digitalWrite(pin_motor_speed, LOW);
253     }
254     else if (x < 40)
255     {
256       Serial.println("Maintainance: Checking sensors...");
257       checkpins();
258     }
259   }
260   if (t > 1005) {
261     return 1;
262   } else if (t > 960) {
263     return 2;
264   } else if (t > 900) {
265     return 3;
266   } else if (t > 740) {
267     Serial.println("Emergency stop.");
268     return 4; //Stop button
269   } else {
270     return -1;
271   }
272 }
273
```

**Εικόνα 42 Μέθοδος check_buttons()**

Η συνάρτηση check_buttons() είναι τύπου int και επιστρέφει τιμή ορόφου ή έκτακτης ακινητοποίησης. Σε κάθε κλήση της διαβάζει για κάποιο αίτημα από την είσοδο του πληκτρολογίου ελέγχου του θαλάμου. Έπειτα διαβάζει από την σειριακή θύρα, της οποίας δίνει προτεραιότητα καθώς περιλαμβάνει και εντολές συντήρησης (*maintainance*) για την επαναφορά του θαλάμου στα όρια των ορόφων αν έχει βγει εκτός καθώς και αιτήματος ενημέρωσης για τους αισθητήρες. Αν δεν υπάρχει

λοιπόν κάποιο αίτημα από την σειριακή θύρα ελέγχει το σήμα εισόδου του πληκτρολογίου και επιστρέφει το ανάλογο αίτημα.

```
274 int emergency_check() {
275   if ((digitalRead(pin_emergency_bottom)) or (digitalRead(pin_emergency_top))) {
276     digitalWrite(pin_motor_speed, 0);
277     car_error_state = true;
278     Serial.print("Error ");
279     Serial.println(analogRead(pin_emergency_top));
280     Serial.println(analogRead(pin_emergency_bottom));
281     Serial.println(car_error_state);
282     checkpins(); //debug
283     delay(500);
284     return -1;
285   }
286   else
287     car_error_state = false;
288   return 0;
289 }
```

**Εικόνα 43 Μέθοδος emergency_check()**

Η συνάρτηση “*emergency_check()*” κάνει αυστηρά έναν έλεγχο τον οποίο πρέπει να κάνει και εξαιρετικά γρήγορα. Αυτός είναι αν ο θάλαμος έχει περάσει τα όρια κίνησης του στο φρεάτιο.  Σε ανάλογη περίπτωση ενημερώνει τη σειριακή έξοδο, καθώς και την μεταβλητή σήμανσης σφάλματος.

```
291 void checkpins() {
292   Serial.println("Service check:");
293   Serial.print("pin_enable_motor_power: ");
294   Serial.println(digitalRead(pin_enable_motor_power));
295   Serial.print("pin_motor_speed: ");
296   Serial.println(digitalRead(pin_motor_speed));
297   Serial.print("pin_motor_direction: ");
298   Serial.println(digitalRead(pin_motor_direction));
299   Serial.print("pin_emergency_top: ");
300   Serial.println(digitalRead(pin_emergency_top));
301   Serial.print("pin_emergency_bottom: ");
302   Serial.println(digitalRead(pin_emergency_bottom));
303   Serial.print("pin_2floor_stop: ");
304   Serial.println(digitalRead(pin_2floor_stop));
305   Serial.print("pin_2floor_reach_under: ");
306   Serial.println(digitalRead(pin_2floor_reach_under));
307   Serial.print("pin_1floor_reach_over: ");
308   Serial.println(digitalRead(pin_1floor_reach_over));
309   Serial.print("pin_1floor_stop: ");
310   Serial.println(digitalRead(pin_1floor_stop));
311   Serial.print("pin_1floor_reach_under: ");
312   Serial.println(digitalRead(pin_1floor_reach_under));
313   Serial.print("pin_0floor_reach_over: ");
314   Serial.println(digitalRead(pin_0floor_reach_over));
315   Serial.print("pin_0floor_stop: ");
316   Serial.println(digitalRead(pin_0floor_stop));
317 }
```

**Εικόνα 44 Μέθοδος checkpins()**

Η συνάρτηση "*void check_pins()*" στέλνει στη σειριακή θύρα αναφορά της κατάστασης των άμεσων περιφερειακών (εισόδων και εξόδων) του Arduino. Χρησιμεύει σαν πρώτο στάδιο διαπίστωσης βλάβης προ φυσικής παρουσίας στο χώρο.

# ΚΕΦΑΛΑΙΟ 7

Στο συγκεκριμένο κεφάλαιο συμπεριλαμβάνονται κάποιες γενικές ιδέες, συμπεράσματα από την κατασκευή μας, προτάσεις και προοπτικές, δυνατότητες εξέλιξης της κατασκευής καθώς και πρόταση για δημιουργία ενότητας μαθήματος εργαστηρίου πάνω στην συγκεκριμένη κατασκευή.

## 7.1 Συμπεράσματα και προτάσεις

### 7.1.1 Η κατασκευή σαν αποτέλεσμα

Η συγκεκριμένη κατασκευή, δεν αποτελεί ένα απλό μοντέλο αναβατορίου, άλλα αντιθέτως, αποτελεί ένα ολοκληρωμένο μοντέλο ανελκυστήρα με αντίβαρο, καθώς οι περισσότεροι σύγχρονοι λειτουργικοί ανελκυστήρες έλξης, εφαρμόζουν την χρήση αντίβαρου για βέλτιστα αποτελέσματα με όσο το δυνατόν λιγότερες σπατάλες, τόσο σε ενέργεια, όσο και στην φθορά των εξαρτημάτων.
Συνεπώς, μπορούμε να πούμε ότι σε ένα πρώτο στάδιο τουλάχιστον, η κατασκευή μας, προσομοιώνει αρκετά έναν απλό ανελκυστήρα, ενδεχομένως οικιακής χρήσης και όχι εμπορικής ή και εταιρικής, σε ένα φάσμα τριών επιπέδων.

## 7.2 Δυνατότητες εξέλιξης

Οι δυνατότητες εξέλιξης της εργασίας είναι πάρα πολλές. Από ηλεκτρολογικής και ηλεκτρονικής φύσεως, μέχρι προγραμματιστικής και τηλεδιαχείρισης.
Από ηλεκτρολογικής σκοπιάς, πιθανές προσθήκες θα μπορούσαν να είναι πιο παχιές καλωδιώσεις για διαχείριση μεγαλύτερης έντασης ρεύματος, περισσότερες ασφάλειες για πιθανές διαρροές και φυσικά τοποθέτηση μεγαλύτερου τροφοδοτικού έτσι ώστε να παρέχεται η απαραίτητη ενέργεια σε ένα επίσης μεγαλύτερο κινητήρα ο οποίος θα μπορούσε να τοποθετηθεί, αναλόγως φυσικά και με τις απαιτήσεις της κατασκευής και το που θα μπορούσε να εφαρμοσθεί.
Ένας μεγαλύτερος κινητήρας ενδεχομένως, θα εξυπηρετούσε συνδυαστικά με την βελτίωση του ηλεκτρονικού κομματιού της κατασκευής, καθώς δύναται η πιθανότητα τοποθέτησης κατάλληλων αισθητήρων βάρους θαλάμου.
Μια εφαρμογή αισθητήρων βάρους, σε συνδυασμό με το ΨΣΑΕ της κατασκευής, ενδεχομένως θα έδινε διαφορετική εντολή ισχύος στον κινητήρα, έτσι ώστε ο

κινητήρας να εφάρμοζε αντίστοιχη ισχύ προκειμένου να δημιουργηθεί η απαραίτητη ροπή για την μεταφορά μεταβλητού βάρους.
Άλλες πιθανές εξελίξεις στο ηλεκτρονικό κομμάτι, θα μπορούσαν σίγουρα να είναι περαιτέρω μέτρα ασφάλειας, όπως τοποθέτηση κουμπιού STOP, τοποθέτηση κουμπιού συναγερμού, τοποθέτηση κουμπιού κλήσης βοήθειας καθώς και τοποθέτηση τηλεφώνου συνδεδεμένου σε δίκτυο άμεσης ανάγκης.
Επιπλέον, η εφαρμογή μιας μεθόδου τηλεδιαχείρισης του ανελκυστήρα είναι πολύ πιθανή, καθώς το ΨΣΑΕ μας χάρη στο Arduino, παρέχει ποικίλες μεθόδους επικοινωνίας από απόσταση, είτε μέσω της τεχνολογίας Bluetooth, ή ακόμα και μέσω του διαδικτύου. Συνεπώς, υλοποιώντας μια αντίστοιχη web εφαρμογή για την διαχείριση του ανελκυστήρα, κάλλιστα εντάσσεται η κατασκευή στην φιλοσοφία ενός smart home.

### 7.2.1 Ολοκλήρωση PLC

Στην παρούσα κατασκευή, η υλοποίηση του ΨΣΑΕ διαφέρει με αυτήν που εφαρμόζει ένας ελεγκτής τύπου PLC, καθώς το Arduino μπορεί να προγραμματιστεί σαν ΨΣΑΕ, πολύ πιο εύκολα από ένα PLC, εφαρμόζοντας κυρίως την φιλοσοφία του προγραμματισμού μιας γλώσσας υψηλού επιπέδου (C, C++) όπως η Wiring.
Παρόλα αυτά, ακόμα και μέσα στο ίδιο το Arduino θα μπορούσε κάλλιστα να εφαρμοσθεί η φιλοσοφία ενός PLC, χρησιμοποιώντας ακόμα και την γλώσσα Ladder.

### 7.2.2 Εισαγωγή Ladder framework

Η γλώσσα Ladder είναι μια πολυχρησιμοποιημένη γλώσσα όσον αφορά τον προγραμματισμό συστημάτων αυτοματισμού. Υπάρχουν πάμπολλα προγράμματα και πλαίσια (frameworks) υλοποίησης σε γλώσσες προγραμματισμού που ενσωματώνουν την Ladder στον κώδικά τους.
Ένα απο αυτά τα πλαίσια είναι το SoapBox Snap. Το SoapBox Snap είναι μια πλατφόρμα αυτοματοποίησης για υλοποίηση σε ηλεκτρονικούς υπολογιστές και είναι ανοιχτού κώδικα. Περιλαμβάνει έναν επεξεργαστή λογικής κλίμακας και ένα εύχρηστο περιβάλλον εκτέλεσης κατευθείαν από την εγκατάστασή του. Ο επεξεργαστής κλίμακας περιλαμβάνει συνήθεις στοιχεία που χρησιμοποιούνται στα ηλεκτρονικά κυκλώματα όπως επαφές, πηνία, χρονοδιακόπτες, μετρητές, και

οδηγίες και ρυθμίσεις επαναφοράς. Το εύχρηστο περιβάλλον συνοδεύεται από προγράμματα οδήγησης για συσκευές εισόδου / εξόδου Phidgets που συνδέονται απευθείας στη θύρα USB. Το SoapBox Snap έρχεται επίσης με ένα περιβάλλον εκτέλεσης Arduino, το οποίο σημαίνει ότι μπορούμε να φορτώσουμε προγράμματα Ladder σε ένα Arduino (UNO, Nano ή Mega board) και ακόμη και να κάνουμε απασφαλμάτωση κατά την διάρκεια ενεργής σύνδεσης του Arduino.
Το Snap ουσιαστικά αποδίδεται ως "Snap is Not a PLC", το οποίο πρακτικά σημαίνει ότι δεν ενδείκνυται σε καμία περίπτωση για βιομηχανικές χρήσεις. Αντ' αυτού, ενδείκνυται για ακαδημαϊκές χρήσεις, ή ερασιτεχνικές λύσεις σε μικρά προσωπικά προγράμματα και εφαρμογές.[21]

### 7.3 Πρόταση για δημιουργία μαθήματος

Η παρούσα κατασκευή θα μπορούσε να συμπεριληφθεί σαν ενότητα μαθήματος στο πλαίσιο των ΨΣΑΕ, δίνοντας έτσι στους σπουδαστές το ερέθισμα του να αναζητήσουν πληροφορίες για νέες τεχνολογίες όπως το Arduino,κάτι το οποίο μπορεί να τους δώσει πάρα πολλές ευκαιρίες και δυνατότητες υλοποίησης ποικίλων ιδεών, από απλές ατομικές εργασίες μέχρι και εργασίες διπλωματικού και μεταπτυχιακού επιπέδου. Επίσης, θα ενίσχυε το ενδιαφέρον στους σπουδαστές και το μηχανολογικό μέρος της κατασκευής, έτσι ώστε να κατανοήσουν καλύτερα ότι ένα ολοκληρωμένο σύστημα με αυτοματισμό, δεν αποτελείται αποκλειστικά από το ψηφιακό του μέρος ή ακόμα πιο μεμονωμένα από το προγραμματιστικό του μέρος, αλλά σίγουρα υπάρχει και το μηχανολογικό μέρος, όπου ουσιαστικά ο αυτοματισμός απλά καθοδηγεί. Γενικά, η παρούσα κατασκευή, μπορεί κάλλιστα να αποτελέσει αντικείμενο μελέτης και εργαστηριακής ενασχόλησης, πιάνοντας ενα πλήρες φάσμα του μηχανολογικού, ηλεκτρολογικού, ηλεκτρονικού και προγραμματιστικού περιβάλλοντος.

#### 7.3.1 Δημιουργία PLC με Arduino

Πιο συγκεκριμένα, θα μπορούσε να δημιουργηθεί ενότητα μαθήματος με σκοπό την διδασκαλία και ανάθεση δημιουργίας PLC σε μια πλακέτα Arduino, χρησιμοποιώντας κάποιο από τα υπάρχοντα προγραμματιστικά πλαίσια, χωρίς αυτό σαφώς να σημαίνει ότι δεν δίνεται η ευκαιρία στους σπουδαστές και συμμετέχοντες να αυτοσχεδιάσουν, εφόσον φυσικά παραμένουν στο πλαίσιο και

στην φιλοσοφία του PLC. Έτσι με αυτόν τον τρόπο, οι σπουδαστές θα διδαχθούν και το Arduino, το οποίο αποτελεί νέα τεχνολογία με πάρα πολλές εφαρμογές, και το PLC, το οποίο είναι μια πανίσχυρη τεχνολογία αυτοματισμού με βιομηχανικές προοπτικές εφαρμογής.

# ΠΑΡΑΡΤΗΜΑ

Στην παρούσα ενότητα παρατίθεται ο κώδικας του Arduino.

```
/*
   Electronic Lift's master firmware version 1.1107a
   Georgantas Philip, 40266
   Pearlianis Angelos, 40747
*/
//Strings
const String en_initializing = "Initializing...";
const String en_ready = "Ready";
const String en_stop_pressed = "Stop button pressed!";

static const uint8_t pin_enable_motor_power = 10; //Arduino signal OUTPUT pin: Enable power supply for motor driver
static const uint8_t pin_floor_select = A0;        //Arduino signal INPUT pin: Floor selector panel. A kind of keyboard.
static const uint8_t pin_emergency_top = 2;        //Arduino signal INPUT pin: Sets elevator in error state as reached top of the shaft.
static const uint8_t pin_emergency_bottom = 3;     //Arduino signal INPUT pin: Sets elevator in error state as reached bottom of the shaft.
static const uint8_t pin_2floor_stop = 4;        //Arduino signal INPUT pin: Car reached the 2nd floor level
static const uint8_t pin_2floor_reach_under = 5;  //Arduino signal INPUT pin: Car is bellow the 2nd floor level and reaching it
static const uint8_t pin_1floor_reach_over = 6;   //Arduino signal INPUT pin: Car is above the 1st floor level and reaching it
static const uint8_t pin_1floor_stop = 7;        //Arduino signal INPUT pin: Car reached the 1st floor level
static const uint8_t pin_1floor_reach_under = 8;  //Arduino signal INPUT pin: Car is bellow the 1st floor level and reaching it
static const uint8_t pin_0floor_reach_over = 11;  //Arduino signal INPUT pin: Car is above the gr0und floor level and reaching it
```

```
static const uint8_t pin_0floor_stop = 12;          //Arduino signal INPUT pin: Car reached the gr0und floor level
static const uint8_t pin_motor_speed = 9;         //Arduino signal OUTPUT pin: Sets the speed of the motor. (0-255 -> 0-12V)
static const uint8_t pin_motor_direction = 13;    //Arduino signal OUTPUT pin: Sets the motor turn direction. HIGH => Up, LOW => DOWN

//Floor selection to motor direction calculation
uint8_t floor_select[5];
uint8_t floor_index = 0; //0 is the reset value to bring the car to the ground floor. this is a, kind of, reset of the elevator's car position.

uint8_t current_floor_selected = 0;
boolean car_stop_pressed = 0; //0 = Normal 1 = Pressed

//Automated error control variables
boolean car_error_state = 0; // 0 = Normal 1 = Error state

//Speed variable
uint16_t _speed = 0;

void setup() {
  //Serial port initialization
  Serial.begin(9600);
  Serial.println(en_initializing);

  //Set of the pins appropriate role (I/O) in the project.
  pinMode(pin_enable_motor_power, OUTPUT);
  pinMode(pin_motor_speed, OUTPUT);
  pinMode(pin_motor_direction, OUTPUT);
  pinMode(pin_emergency_top, INPUT);
  pinMode(pin_emergency_bottom, INPUT);
  pinMode(pin_2floor_stop, INPUT);
```

```
  pinMode(pin_2floor_reach_under, INPUT);
  pinMode(pin_1floor_reach_over, INPUT);
  pinMode(pin_1floor_stop, INPUT);
  pinMode(pin_1floor_reach_under, INPUT);
  pinMode(pin_0floor_reach_over, INPUT);
  pinMode(pin_0floor_stop, INPUT);

  //Maintainance check for any wrong signals from the pins. Value depends on the position of the car.
  checkpins();
  digitalWrite(pin_enable_motor_power, HIGH);

  if (!digitalRead(pin_0floor_stop))
    Serial.println("Reseting car's position to ground floor");
  //    Serial.println("Car not in ground floor... Trying to detect if it is at the bottom of the shaft");
  if (!digitalRead(pin_emergency_bottom) and !digitalRead(pin_0floor_stop)) {
    if (!digitalRead(pin_emergency_top))
    {
      digitalWrite(pin_motor_direction, HIGH);
      analogWrite(pin_motor_speed, 180);
      delay(500);
      digitalWrite(pin_motor_speed, LOW);
      //    Serial.println("Car was at the bottom of the shaft.");
    }
    digitalWrite(pin_motor_direction, LOW);
  }
  else
    digitalWrite(pin_motor_direction, HIGH);

  while (!digitalRead(pin_0floor_stop))
    analogWrite(pin_motor_speed, 180);
  digitalWrite(pin_motor_speed, LOW);
```

```
  digitalWrite(pin_motor_direction, HIGH);
  digitalWrite(pin_enable_motor_power, LOW);
  Serial.println(en_ready);
}

void loop() {
  int current_floor_selected = check_buttons();
  switch (current_floor_selected) {
    case 1: { //Call to ground floor
        Serial.print("Floor call: ");
        Serial.println(current_floor_selected);
        Serial.println(car_error_state);
        Serial.println(car_stop_pressed);
        digitalWrite(pin_motor_speed, LOW);
        digitalWrite(pin_enable_motor_power, HIGH);
        digitalWrite(pin_motor_direction, LOW); //Turn direction to down
        if (car_error_state or car_stop_pressed or digitalRead(pin_0floor_stop))
          break;
        else
          accel(pin_0floor_stop);
        Serial.println(digitalRead(pin_0floor_reach_over));
        while (!digitalRead(pin_0floor_reach_over) and (!car_error_state)) {
          if ((emergency_check() == -1) or (check_buttons() == 4))
            break;
        }
        if (car_error_state or car_stop_pressed)
          break;
        else {
          deccel(pin_0floor_stop);
          while (!digitalRead(pin_0floor_stop) and (!car_error_state)) {
            if ((emergency_check() == -1) or (check_buttons() == 4))
              break;
```

```
        }
        //        if (car_error_state or car_stop_pressed)
        //          break;
        digitalWrite(pin_motor_direction, HIGH);
        digitalWrite(pin_motor_speed, LOW);
        digitalWrite(pin_enable_motor_power, LOW);
        Serial.println("Stopped!");
        break;
       }
      case 2: { //Call to 1st floor
        Serial.print("Floor call: ");
        Serial.println(current_floor_selected);
        digitalWrite(pin_enable_motor_power, HIGH); //Turn direction to up or down
depending on current floor (stop or previews select)
        if (car_error_state or car_stop_pressed)
          break;
        else
          accel(pin_1floor_stop);
        while                    (!digitalRead(pin_1floor_reach_over)                    and
!digitalRead(pin_1floor_reach_under) and (!car_error_state)) {
          if ((emergency_check() == -1) or (check_buttons() == 4))
            break;
        }
        if (car_error_state or car_stop_pressed)
          break;
        else
          deccel(pin_1floor_stop);
        while (!digitalRead(pin_1floor_stop) and (!car_error_state)) {
          if ((emergency_check() == -1) or (check_buttons() == 4))
            break;
        }
        if (car_error_state or car_stop_pressed)
          break;
```

```
      else
        digitalWrite(pin_motor_speed, LOW);
      digitalWrite(pin_enable_motor_power, LOW);
      Serial.println("Stopped!");
      break;
    }
  case 3: { //Call to 2nd floor
      Serial.print("Floor call: ");
      Serial.println(current_floor_selected);
      //Turn direction to up
      digitalWrite(pin_motor_speed, LOW);
      digitalWrite(pin_enable_motor_power, HIGH);
      digitalWrite(pin_motor_direction, HIGH);
      if (car_error_state or car_stop_pressed)
        break;
      else
        accel(pin_2floor_stop);
      while (!digitalRead(pin_2floor_reach_under) and (!car_error_state)) {
        if ((emergency_check() == -1) or (check_buttons() == 4))
          break;
      }
      if (car_error_state or car_stop_pressed)
        break;
      else
        deccel(pin_2floor_stop);
      while ((!digitalRead(pin_2floor_stop)) and (!car_error_state)) {
        if ((emergency_check() == -1) or (check_buttons() == 4))
          break;
      }
      digitalWrite(pin_motor_speed, LOW);
      digitalWrite(pin_motor_direction, LOW);
      digitalWrite(pin_enable_motor_power, LOW);
      Serial.println("Stopped!");
```

```
        break;
      }
    case 4: { //Stop button
        //Turn direction to down
        Serial.println("Allready stopped!");
        break;
      }
    }
    digitalWrite(pin_motor_speed, LOW);
    car_stop_pressed = false;
    car_error_state = false;

}

void accel(int floor_selected_stop) {
  Serial.println("Starting acceleration");
  for (_speed = 0; _speed <= 255; _speed++) {
    analogWrite(pin_motor_speed, _speed);
    if ((emergency_check() == -1) or (check_buttons() == 4))
      break;
    delay(5);
  }
  Serial.println("Finished acceleration...");
}


void deccel(int floor_selected) {
  Serial.println("Starting decceleration...");
  int y;
  if (digitalRead(pin_motor_direction) == LOW)
    y = 85;
  else //To be changed with weight detection...?how?
    y = 85;
```

```
  for (_speed--; _speed >= y; _speed--) {
    analogWrite(pin_motor_speed, _speed);
    if ((emergency_check() == -1) or (check_buttons() == 4))
      break;
    delay(3);
  }
  Serial.println("Finished decceleration");
}

void stop_pressed() {
  digitalWrite(pin_motor_speed, LOW);
  Serial.println(en_stop_pressed);
  car_stop_pressed = true;
}

int check_buttons() {
  int t = analogRead(pin_floor_select);
  if (Serial.available()) {
    Serial.print("Serial available: ");
    int x = Serial.parseInt(); //Check Serial for any remote command
    Serial.println(x);
    if ((x >= 1) && (x <= 4))
    {
      if (x == 4)
        car_stop_pressed = true;
      digitalWrite(pin_motor_speed, LOW);
      return x;
    }
    else if ( x == 41)
    {
      Serial.println("Maintainance: Recovering car position to ground floor");
      digitalWrite(pin_motor_direction, HIGH);
      while (digitalRead(pin_emergency_bottom))
```

```
        digitalWrite(pin_motor_speed, HIGH);
      digitalWrite(pin_motor_speed, LOW);
    } else if ( x == 43)
    {
      Serial.println("Maintainance: Recovering car position to 2nd floor");
      digitalWrite(pin_motor_direction, LOW);
      while (digitalRead(pin_emergency_top))
        digitalWrite(pin_motor_speed, HIGH);
      digitalWrite(pin_motor_speed, LOW);
    }
    else if (x < 40)
    {
      Serial.println("Maintainance: Checking sensors...");
      checkpins();
    }
  }
  if (t > 1005) {
    return 1;
  } else if (t > 960) {
    return 2;
  } else if (t > 900) {
    return 3;
  } else if (t > 740) {
    Serial.println("Emergency stop.");
    return 4; //Stop button
  } else {
    return -1;
  }
}

 int emergency_check() {

   if ((digitalRead(pin_emergency_bottom)) or (digitalRead(pin_emergency_top))) {
```

```
    digitalWrite(pin_motor_speed, 0);
    car_error_state = true;
    Serial.print("Error ");
    Serial.println(analogRead(pin_emergency_top));
    Serial.println(analogRead(pin_emergency_bottom));
    Serial.println(car_error_state);
    checkpins(); //debug
    delay(500);
    return -1;
  }
  else
    car_error_state = false;
  return 0;
}

 void checkpins() {
  Serial.println("Service check:");
  Serial.print("pin_enable_motor_power: ");
  Serial.println(digitalRead(pin_enable_motor_power));
  Serial.print("pin_motor_speed: ");
  Serial.println(digitalRead(pin_motor_speed));
  Serial.print("pin_motor_direction: ");
  Serial.println(digitalRead(pin_motor_direction));
  Serial.print("pin_emergency_top: ");
  Serial.println(digitalRead(pin_emergency_top));
  Serial.print("pin_emergency_bottom: ");
  Serial.println(digitalRead(pin_emergency_bottom));
  Serial.print("pin_2floor_stop: ");
  Serial.println(digitalRead(pin_2floor_stop));
  Serial.print("pin_2floor_reach_under: ");
  Serial.println(digitalRead(pin_2floor_reach_under));
  Serial.print("pin_1floor_reach_over: ");
  Serial.println(digitalRead(pin_1floor_reach_over));
```

```
    Serial.print("pin_1floor_stop: ");
    Serial.println(digitalRead(pin_1floor_stop));
    Serial.print("pin_1floor_reach_under: ");
    Serial.println(digitalRead(pin_1floor_reach_under));
    Serial.print("pin_0floor_reach_over: ");
    Serial.println(digitalRead(pin_0floor_reach_over));
    Serial.print("pin_0floor_stop: ");
    Serial.println(digitalRead(pin_0floor_stop));
  }
```

# ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] https://en.wikipedia.org/wiki/Elevator

[2] https://www.collinsdictionary.com/dictionary/english/elevator-car

[3] https://en.wikipedia.org/wiki/Counterweight

[4] "Laying the foundation for today's skyscrapers". San Francisco Chronicle. August 23, 2008.

[5] The Book of Secrets — Kitab al Asrar of al-Muradi

[6] "Louis XV's flying chair – Exposition Sciences et Curiosités à la Cour de Versailles – 26 octobre 2010 au 3 avril 2011"

[7] "Conveyor technology: Elevator". conveyor-tech.com

[8] Bellis, Mary. "Learn Who Invented the Elevator and More". Inventors.about.com

[9] "EW Museum". Theelevatormuseum.org

[10] "Skyscrapers," Magical Hystory Tour: The Origins of the Commonplace & Curious in America (September 1, 2010).

[11] "The Cooper Union Library: Foundation Building". cooper.edu.

[12] "Peter Cooper, a Brief Biography". ringwoodmanor.com.

[13] Mary Bellis. "History of the Elevator". About.com Money.

[14] Ralph Turvey, London Lifts and Hydraulic Power, Transactions of the Newcomen Society, Vol. 65, 1993–94, pp. 147–164

[15] "Remembering When Driverless Elevators Drew Skepticism". NPR. July 31, 2015. Retrieved April 26, 2017.

[16] https://www.britannica.com/technology/elevator-vertical-transport

[17] ACE Lifts. "Traction Lifts: an infographic on how they work". ACE Lifts.

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

[19] https://wiring.org.co/

[20] https://store.arduino.cc/arduino-pro-mini

[21] http://soapboxautomation.com/products/soapbox-snap/

[22] http://plcladdersimulator.weebly.com/pro-edition.html