



SSY
A4T

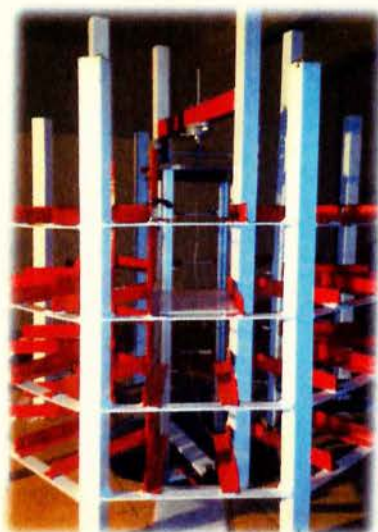
Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Αυτοματισμού

Πτυχιακή εργασία

ΘΕΜΑ:

«Αυτόματο Παρκινγκ»



Επιβλέπων καθηγητής: Μιχάλης Παπουτσιδάκης

Φοιτητές: Ανδρέας Τρίκος A.M: 37999

Σωτήρης Αχλάδης A.M: 37998

Ανδρέας Παραδεισιώτης A.M: 38213

ΒΙΒΛΙΟΘΗΚΗ
ΤΕΙ ΠΕΙΡΑΙΑ

Περιεχόμενα

Εισαγωγή	3
• Ιστορική ανάδρομη	5
• Ρομποτικά παρκινγκ στην Ελλάδα	8
Κεφάλαιο 1 – Το κτήριο	10
• Η κύρια δομή του κτηρίου	10
• Κατασκευή	11
• Η κίνηση του κτηρίου	12
Κεφάλαιο 2 – Μηχανισμός κίνησης (αναβατόριο)	13
• Η δομή του αναβατορίου	13
• Η μετάδοση κίνησης	15
Κυκλική κίνηση:	15
Κάθετη κίνηση:	16
Οριζόντια κίνηση:	17
Κεφάλαιο 3 – Συνδέσεις κυκλώματα	19
• Τροφοδοσία ηλεκτρικών – ηλεκτρονικών στοιχείων	19
• Οθόνη – Πληκτρολόγιο	20
• Οδηγοί βηματικών κινητήρων (stepper drivers)	21
• Οδηγός DC κινητήρα	22
• Αισθητήριο υπερύθρων	23
Κεφάλαιο 4 – Η «Νοημοσύνη» του κτηρίου (Arduino)	24
• Η σχετική νοημοσύνη	24
1) Παραλαβή-Τοποθέτηση από τη είσοδο:	24
2) Παραλαβή-Τοποθέτηση στην έξοδο:	24
• Προγραμματισμός-Επεξήγηση κώδικα (Arduino)	25
Κεφάλαιο 5 – Οδηγίες χρήσης	46
• Κεντρική περίπτωση	46
• Περιπτώσεις προς στάθμευση	46

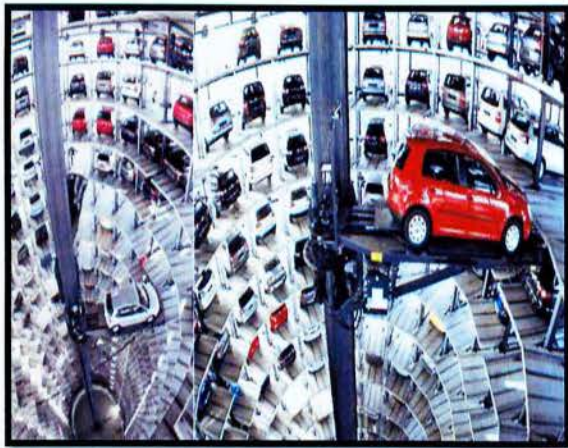
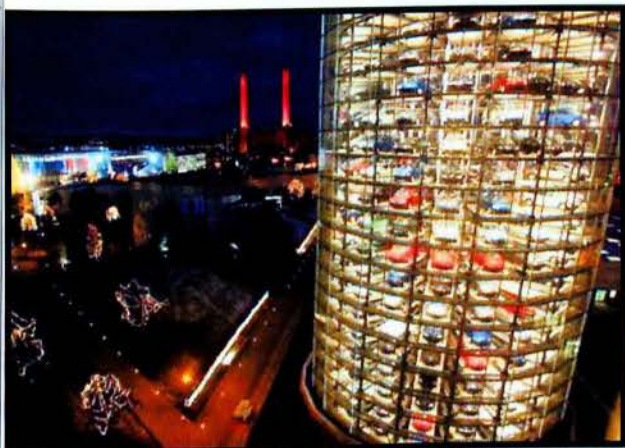
• Περίπτωση αποστάθμευσης	47
Κεφάλαιο 6 – Ολικό κύκλωμα και εξαρτήματα	48
• Ολικό κύκλωμα	48
• Ηλεκτρονικά - Ηλεκτρολογικά Εξαρτήματα	49
Κεφάλαιο 7 – Μηχανολογικά εξαρτήματα	56
Κεφάλαιο 8 – Μελλοντικές βελτιώσεις	63
• Πρόταση 1- Πηγές ανανεώσιμης ενέργειας	63
• Πρόταση 2- Online παρακολούθηση	64
• Πρόταση 3- Εντολές από Smartphone	64
Κεφάλαιο 9 - Μελλοντικές βελτιώσεις της κατασκευής μας	65
Κεφάλαιο 10 – Interfacing	67
• Περιγραφή	67
• Κώδικας	69
Κεφάλαιο 11 – Το Λογισμικό που χρησιμοποιήσαμε	77

Εισαγωγή

Η ραγδαία αύξηση της αυτοκινητοβιομηχανίας και η αυξημένη αγορά αυτοκινήτων από το καταναλωτικό κοινό δημιούργησε ένα νέο πρόβλημα και για τις δυο πλευρές .

Το πρόβλημα αυτό αφορά, την δυσκολία αποθήκευσης προϊόντων για τους μεν και την δυσκολία εύρεσης στάθμευσης και προφύλαξης του οχήματος για τους δε.

Η τεχνολογία είχε για ακόμα μια φορά την λύση. Φτιαχτήκαν τα πρώτα αυτόματα συστήματα αποθήκευσης –προφύλαξης οχημάτων . Όπως οι πύργοι φύλαξης της εταιρίας Volkswagen που βρίσκονται στο πάρκο Autostadt, στο Βόλφσμπουργκ της Γερμανίας και απεικονίζονται στις παρακάτω 3 εικόνες.



Το Autostadt στο Wolfsburg, είναι ένα εντυπωσιακό θεματικό πάρκο ιδιοκτησίας της Volkswagen Group που προσκαλεί τους επισκέπτες του να ταξιδέψουν στον κόσμο της αυτοκίνησης. Συγκλονιστική αρχιτεκτονική και σχεδίαση σε 400 στρέμματα ‘κυλιόμενων’ λόφων και λιμνών προσφέρουν έναν παράδεισο των αυτοκινήτων!!

- Εκεί βρίσκονται και οι 2 γιγάντιοι πύργοι με αυτοκίνητα , που έχουν μηχανήματα αυτόματης πώλησης. Κάθε νέο αυτοκίνητο που έρχεται από το εργοστάσιο, μεταφέρεται από ρομπότ σε μια κενή υποδοχή αποθήκευσης σε έναν από τους πύργους. Όταν ένας πελάτης διαλέξει το αυτοκίνητο που επιθυμεί για να αγοράσει , το ίδιο ρομποτικό σύστημα μεταφέρει το όχημα όπως στην δεύτερη εικόνα , και το φέρνει κάτω στο έδαφος , στην συνέχεια το μεταφέρει στο επόμενο κτίριο. Κάθε πύργος χωράει έως και 400 αυτοκίνητα

έχει 20 επίπεδα και μπορεί να μεταφέρει ένα αυτοκίνητο κάθε 45 δευτερόλεπτα κατά τις ώρες αιχμής.

- Το πρώτο εξολοκλήρου αυτοματοποιημένο παρκινγκ στις ΗΠΑ μη ιδιωτικής χρήσης, άνοιξε τις πόρτες του στο ενδιαφερόμενο καταναλωτικό κοινό το καλοκαίρι του 2002. Κατασκευάστηκε απτήν Robotic Parking Systems και βρίσκεται στο Hoboken, του New Jersey. Ο συγκεκριμένος χώρος στάθμευσης ύψους 17 μέτρων, απαρτίζεται από 7 ορόφους, έχει 4 εισόδους – εξόδους, λειτουργεί 24 ώρες το 24άωρο και διαθέτει 314 θέσεις που δίνονται κατά κύριο λόγο για μηνιαία ενοικίαση. Το μέσο κόστος ανά μηνά ανέρχεται στα 235 \$. Συμφώνα με κάποιες στατιστικές που έγιναν κατά τα πρώτα δυο χρόνια λειτουργιάς του, εξυπηρετεί 122 αυτοκίνητα την ώρα (ώρες αιχμής), έχει μέσο ορό εξόδου-οχηματος τα 145 δευτερόλεπτα και η πληρότητα του αγγίζει το 95,5% .

- Άλλο ένα επίτευγμα της εταιρίας Robotic Parking Systems που ξεκίνησε την λειτουργιά του στις 12 Αυγούστου του 2009 βρίσκεται στο Dubai . Είναι το πρώτο αυτοματοποιημένο γκαράζ στην περιοχή της Μέσης Ανατολής. Έχει 8 εισόδους- εξόδους, ύψος 17,5 μέτρα και απαρτίζεται από 765 θέσεις στάθμευσης. Αξιοσημείωτο είναι το γεγονός ότι το συγκεκριμένο αυτόματο παρκινγκ ίσως είναι το μόνο που μπορεί να εξυπηρετήσει περισσότερα από 250 οχήματα την ώρα, πράγμα που συμβαίνει κατά τις ώρες αιχμής.



Τα κτήρια αυτά όχι μονό δεν έχουν τίποτα να ζηλέψουν από άποψη design τους κλασικούς αποθηκευτικούς χώρους αλλά υπερτερούν και σε πολλά ακόμα θέματα όπως είναι:

1. Η μείωση του όγκου του κτηρίου . Λόγο του ότι η μηχανές που αποθηκεύουν αυτόματα τα αμαξώματα απαιτούν πολύ μικρότερο χώρο από τους διάδρομους που οδηγούν στις εκάστοτε αποθηκευτικές θέσεις .
2. Η αυξημένη ασφάλεια του οχήματος. Διότι μονό η μηχανές μπορούν να φτάσουν στις αποθηκευτικές θέσεις και να φέρουν το αμάξιωμα στην έξοδο.
3. Η φιλική συμπεριφορά στον χρήστη μιας και δεν είναι αναγκαίο ούτε να υπάρχει κάποιος να οδηγήσει το αμάξιωμα στην θέση αποθήκευσης αλλά ούτε να οδηγήσει ο ίδιος έως αυτήν. Όλα γίνονται με την εισαγωγή απλών εντολών σε μια οθόνη.
4. Η μείωση απαιτούμενου προσωπικού για φύλαξη και την διεκπεραίωση της διαδικασίας αποθήκευσης του οχήματος που επιφέρει την μείωση του αντίτιμου φύλαξης κτλ.

Έχοντας την έμπνευση απ' αυτά τα γεγονότα σας παρουσιάζουμε την δική μας πτυχιακή εργασία η οποία είναι σχετική με το παραπάνω θέμα με τίτλο «auto parking» κατασκευάσαμε ένα κτήριο αυτόματης αποθήκευσης-φύλαξης οχημάτων υπό κλίμακα. Η κατασκευή αναλύεται λεπτομερώς στις παρακάτω ενότητες.

Ιστορική ανάδρομη

Πρώτου όμως προβούμε στην ανάλυση του συστήματος μας ας κάνουμε πρώτα μια ιστορική ανάδρομη .

Πρώτα από όλα έγινε η ανακάλυψη του τροχού και έπειτα ακλούθησαν όλα τα αλλά! Η αρχαιότερη χρήση τροχών είναι πιθανά αυτή των ομώνυμων συσκευών της αγγειοπλαστικής την 5η χιλιετία π.χ. στη Μεσοποταμία. Κάρα με τροχούς έχουν βρεθεί σε προϊστορικούς τάφους στην περιοχή του Καυκάσου οι οποίοι χρονολογούνται γύρω στο 3700 π.Χ..



(Σουμεριακή απεικόνιση πολεμικών αρμάτων με τροχούς, γύρω στο 3200 π.Χ..)

Η εφεύρεση της άμαξας συμπίπτει με την επινόηση του τροχού. Φαίνεται ότι πρώτοι χρησιμοποίησαν τα αμάξια οι λαοί της Μεσοποταμίας, οι Σουμέριοι και οι Χετταίοι και όχι οι Ασύριοι όπως πίστευαν παλιότερα. Πρώτοι οι Αιγύπτιοι εφεύραν τον τροχό με

μεταλλική στεφάνη και τον ακτινωτό τροχό, που αργότερα τελειοποίησαν οι Ασσύριοι. Είναι χαρακτηριστικό ότι στις πρώτες άμαξες οι ρόδες ήταν καρφωμένες στον άξονα, ενώ αργότερα τελειοποιούνται και οι ρόδες γυρίζουν ελεύθερες, γύρω από του ακίνητο άξονα.

Για την κατασκευή του πρώτου χώρου στάθμευσης απαιτείται πρώτα η εφεύρεση του πρώτου αυτοκίνητου.

Το πρώτο ολοκληρωμένο βενζινοκίνητο αυτοκίνητο κατασκευάστηκε το 1908 από τον Χένρι Φόρντ. Την αρχή έκανε στην Γαλλία, το 1769, ο Νικολά Κουνιό (Nicolas Joseph Cugnot), δημιουργώντας το πρώτο αυτοκίνητο όχημα, ένα ατμοκινούμενο αμάξι, το fardier. Το ασταθές αυτό όχημα ανετράπη και χτύπησε σε ένα τοίχο, αποτελώντας έτσι και το πρώτο ατύχημα με αυτοκινούμενο όχημα στην ιστορία.



(Μοντέλο Karl Benz 1894)

Το όχημα του Marcus έχει ήδη ξεπεράσει το μηχανικό κινητήρα του Κουνιό σε μηχανική ενέργεια. 92 χρόνια αργότερα, ο Ετιέν Λενουάρ (Etienne Lenoir) έφτιαξε το πρώτο αυτοκίνητο με μηχανή εσωτερικής καύσης και ένα χρόνο αργότερα ο Λενουάρ πραγματοποίησε το 1ο ταξίδι με αυτοκίνητο στον κόσμο καλύπτοντας κυκλική διαδρομή 19,3 χλμ. με μέση ταχύτητα 6,4 χλμ /ώρα και ισχύ μόλις 0,5 ίππους.

Αργότερα, τα αυτοκίνητα εξελίχτηκαν και πλέον μπορούσαν να καλύπτουν μεγαλύτερες αποστάσεις σε λιγότερο χρόνο. Η πλειονότητα των αυτοκινήτων που κυκλοφορούσαν πριν το 1900 ήταν ηλεκτρικά.

Περισσότερα αυτοκίνητα σημαίνει περισσότερες θέσεις παρκινγκ.

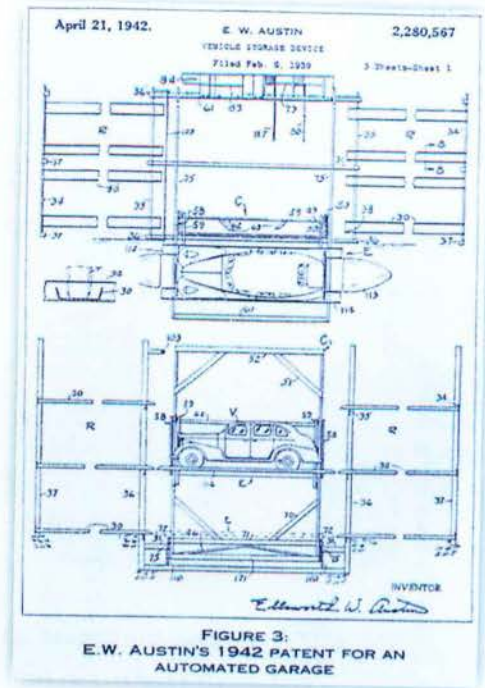
Παρκινγκ είναι η διαδικασία κατά την οποία κάποιος σταθμεύει το όχημα του σε μια από τις δύο πλευρές του δρόμου, χωρίς να εμποδίζει την διέλευση σε αυτές. Οι ευρεία χρήση του αυτοκίνητου όμως έφερε την ανάγκη να υπάρξουν ειδικοί χώροι στάθμευσης (parking lots). Οι ποιοι βρίσκονται εκτός δρόμου σε οικόπεδα οι ειδικά σχεδιασμένα κτήρια.

- Το 1897 πριν τον δεύτερο παγκόσμιο πόλεμο έγιναν τα πρώτα εμπορικά παρκινγκ στις ΗΠΑ τα οποία ήταν κυρίως για τη στάθμευση ηλεκτρικών ταξί
- Τα αυτοματοποιημένα παρκινγκ ξεκινούν την ιστορία τους το 1905 στην Γαλλία
- Το 1920 παρουσιάζονται τα πρώτα αυτόματα παρκινγκ στις Ηνωμένες Πολιτείες Αμερικής (Los Angeles, Chicago, New York). Λειτουργούσε σαν τον τροχό που συναντάμε στα λούνα Παρκ αλλά αντί για



βαγόνια περιστρέφει θέσεις με βάσεις για αυτοκίνητα όπως φαίνεται και στη φωτογραφία.

- Το 1932 κατασκευάζεται από την Nash Motor Company το πρώτο ομοίου τύπου παρκινγκ αλλά εγκλεισμένο σε γυαλί.
- Τον Απρίλιο του 1942, EW Austin χορηγήθηκε δίπλωμα ευρεσιτεχνίας για ένα σχέδιο αυτοματοποιημένου γκαράζ που έγινε ο πρόδρομος των αυτοματοποιημένων γκαράζ της δεκαετίας του '50 και του '60.
- Το Γκαράζ στο Hayward της Βοστώνης άνοιξε το Μάρτιο του 1958, το οποίο αποκάλεσαν ως "μεγαλύτερο μηχανικό χώρο στάθμευσης του κόσμου." Θα χρησιμοποιούσε "μπουτόν στάθμευσης» με τη βοήθεια των πέντε υψηλής ταχύτητας ανελκυστήρων. Χωρητικότητα: 707 θέσεις στάθμευσης.



Τη δεκαετία του '90 και έπειτα άρχισε η πλήρως αυτοματοποιημένη διαδικασία στάθμευσης οχημάτων.

Εταιρείες που πρωτοστατούν σε αυτόν τον τομέα είναι οι ακόλουθες :

1. Η Γερμανική WÖHR
2. Η Ισραηλινή UNITRONICS
3. Η Αμερικάνικη BOOMERANG
4. Η Αμερικάνικη Robotic Parking Systems
5. Η Αμερικάνικη Harding Steel

Εσωτερική και εξωτερική οψη του Boston Hayward Garage



Ρομποτικά παρκινγκ στην Ελλάδα.

Το πιο σύγχρονο παρκινγκ στην Ευρώπη εγκαινιάστηκε στον Άγιο Παντελεήμονα, στην πιο υποβαθμισμένη γειτονιά της Αθήνας. Στεγάζεται σε ένα απόλυτα εργονομικό κτίριο, σε 13 ορόφους (εννέα επάνω και τέσσερις υπόγειους) και έχει χωρητικότητα 180 αυτοκινήτων. Στο ισόγειο υπάρχει σύστημα αισθητήρων, το οποίο μετράει διαστάσεις αυτοκινήτου και αναλόγως η ρομποτική πλατφόρμα με τη βοήθεια των δύο ασανσέρ, που λειτουργούν με εξαιρετικά υψηλές ταχύτητες, επιλέγει τον κατάλληλο όροφο. Όλες οι πληροφορίες πηγαίνουν στον κεντρικό υπολογιστή, από όπου γίνεται αυτόματη διαχείριση του συστήματος. Τυχόν επιπλοκές του συστήματος είναι δυνατόν να διορθωθούν από απόσταση μέσω Διαδικτύου, καθώς το σύστημα είναι πλήρως αυτοματοποιημένο. Στον τελευταίο όροφο η ιδιοκτήτρια εταιρεία «Νέα Εποχή Α.Ε.» προγραμματίζει την τοποθέτηση φωτοβολταϊκών για εξοικονόμηση ενέργειας, ενώ το έργο επιδοτήθηκε κατά 40% από τον αναπτυξιακό νόμο 3299/04 για την πράσινη ανάπτυξη. Σήμερα ο πελάτης πληρώνει μηνιαίως 130 ευρώ.



Στόχος της επιχείρησης είναι να μειώσει το αντίτιμο στα 100 ευρώ το 2012. Η πρώτη ώρα στοιχίζει πέντε ευρώ και κάθε επόμενη ώρα 0,50 λεπτά. Είναι αρκετά φθηνότερο από τα συνηθισμένα παρκινγκ και λίγο ακριβότερο από την ελεγχόμενη υπαίθρια στάθμευση του Δήμου Αθηναίων.

Ρομποτικό παρκινγκ κατασκευάστηκε στην λεωφόρο Πλαστήρα, δίπλα από το υπό ανέγερση Πολιτιστικό Κέντρο Ηρακλείου, ενώ αναμένεται να δώσει λύσεις στο τεράστιο πρόβλημα με την στάθμευση που αντιμετωπίζουν οι κάτοικοι της πόλης αλλά και οι επισκέπτες. Με 700 θέσεις στάθμευσης, πρωτοποριακή και γρήγορη λειτουργία, χρήση υψηλών προδιαγραφών ρομποτικών μηχανημάτων και με γνώμονα την ασφάλεια των οχημάτων αποτελεί αναμφίβολα μια από τις σημαντικότερες επενδύσεις των τελευταίων χρόνων σε επίπεδο Κρήτης.

Ο οδηγός μόλις αφήσει το αυτοκίνητό του, το κλειδώνει και έχει πάντα αυτός το κλειδί, γεγονός που του επιτρέπει να αφήσει τα προσωπικά του αντικείμενα μέσα στο όχημα. Με μια ειδική πλατφόρμα το αυτοκίνητο θα μεταφερθεί στην ειδική θέση που υποδεικνύουν αυτόματα συστήματα και υπολογιστές. Το αυτοκίνητο μεταφέρεται στην θέση που θα παραμείνει χωρίς ο κινητήρας να βρίσκεται σε λειτουργία, γεγονός που συμβάλει στην μείωση των ρύπων και στην προστασία του περιβάλλοντος.



Κεφάλαιο 1 – Το κτήριο

Η κύριος δομή του κτηρίου.

Το κυρίως κτήριο είναι οκτάγωνο, αποτελείται από 4 ορόφους πλην του ισόγειου. Ο κάθε όροφος φέρει 8 αποθηκευτικές θέσεις οχημάτων, ήτοι ένα κτήριο «παρκινγκ» 32 θέσεων συνολικά. Ο πρώτος όροφος απέχει 10 cm από το ισόγειο του κτηρίου λόγω του ότι οι θέσεις στάθμευσης του δεν έχουν από κάτω αμαξώματα, ενώ οι επόμενοι όροφοι απέχουν μεταξύ τους 15cm για να είναι εφικτή η εξαγωγή του άξονα που φέρει την βάση με σκοπό να την τοποθετήσει σε κάποια θέση χωρίς να προσπίπτει στο αμάξωμα του κάτω ορόφου. Οι οδηγοί των θέσεων που θα στηρίζουν την βάση απέχουν μεταξύ τους κατά 15-16cm (λόγω σφαλμάτων που προέκυψαν κατά την κατασκευή) και έχουν μήκος 23cm για να μπορεί να εξαχεται και να τοποθετείται η βάση που έχει μήκος 20cm χωρίς να προσπίπτει ο άξονας εξαγωγής στο οριζόντιο μεταλλικό κομμάτι στήριξής τους. Οι εισοδοί των θέσεων βρίσκονται περιμετρικά ενός νοητού κύκλου στο κέντρο του κτηρίου όπου τοποθετείτε ο μηχανισμός κίνησης. Αυτό συμβαίνει για να μπορεί ο μηχανισμός του αναβατορίου να βρίσκει με την βοήθεια του κεντρικού άξονα του τις θέσεις στάθμευσης, όταν εμείς δίνουμε εντολή περιστροφής κάποιων αλλά και συγκεκριμένων μοιρών στο βηματικό μας κινητήρα . Η είσοδος των οχημάτων βρίσκεται αντικριστά από την έξοδο στο ισόγειο του κτηρίου.



Το κτήριο στηρίζεται σε μία ξύλινη βάση φτιαγμένη από ξύλο MDF. Έχει ύψος 25 cm μήκος 120 cm και πλάτος 120 cm. Στο εσωτερικό της είναι κενή διότι τοποθετήθηκαν τα ηλεκτρονικά και ηλεκτρολογικά στοιχεία. Επίσης μέσα και στο κάτω μέρος της βάσης στηρίχθηκε με την βοήθεια ενός άξονα και ενός ρουλεμάν, ώστε να είναι ελεύθερο κυκλικής κίνησης το αναβατόριο.

Κατασκευή

Η κατασκευή έλαβε μέρος σε ένα παλιό ξυλουργείο το οποίο μας παραχώρησε ο παππούς ενός εκ των τριών μας τον οποίον και ευχαριστούμε .Η κατασκευή έχει συνολικές διαστάσεις 1,20 X 1,20 με ύψος στο 1,10. Το κτήριο όπως προαναφέραμε κατασκευάστηκε από σίδηρο και στηρίχτηκε σε μια ξύλινη pdf βάση οπού μέσα θα κρυφτούν τα ηλεκτρονικά και ηλεκτρολογικά στοιχεία. Η κοπή των μεταλλικών στοιχείων έγινε με την βοήθεια τροχού ενώ η συγκόλληση τους επιτυγχάνετε με ηλεκτροκόλληση. Λόγω του ότι το κτήριο κατασκευάστηκε από σίδηρο έχουμε αυξημένο βάρος αλλά και περισσότερη αντοχή. Τα μέταλλα βιάστηκαν με αστάρι και περάστηκαν έπειτα με μπογιά ώστε να αντέχουν στην οξείδωση λόγω υγρασίας. Είναι μια κατασκευή η όποια έγινε από εμάς μέσα από συλλογική προσπάθεια. Χρειαστήκαμε δύο ολόκληρους μήνες για την κατασκευή του κτηρίου λόγω της μειωμένης εμπειρίας μας σε εργαλεία όπως ο τροχός και η ηλεκτροκόλληση που απαιτούν ασφάλεια και προσοχή.



Τα μέταλλα τα προμηθευτήκαμε από καταστήματα πώλησης σιδήρου στον Πειραιά ενώ τα χρώματα αλλά και διάφορα αλλά υλικά που χρησιμοποιήθηκαν για την κατασκευή και ολοκλήρωση του κτηρίου τα προμηθευτήκαμε από τοπικό κατάστημα πώλησης χρωμάτων και οικοδομικών υλικών.

Η κίνηση του κτηρίου

Η κίνηση του κτηρίου επιτυγχάνεται με την βοήθεια ενός κεντρικού μηχανισμού, τον οποίο εμείς καλούμε ως αναβατόριο.

Η κάθε θέση αποθήκευσης όπως φαίνεται στις παραπάνω εικόνες είναι κενή, γι' αυτό και συμπληρώνεται με μια βάση.

Η βάση είναι αυτή που παραλαμβάνεται από το αναβατόριο είτε με αμάξωμα πάνω της όταν οδεύουμε για να παρκάρουμε ή να το φέρουμε στην έξοδο. Είτε χωρίς αυτό όταν θέλουμε να φέρουμε την βάση στην είσοδο για παραλαβή αμαξώματος ή να την επανατοποθετήσουμε στην θέση της αφού το αμάξωμα φύγει από την έξοδο.

Το αναβατόριο για να είναι ικανό να αποθηκεύσει ένα αμάξωμα σε οποιαδήποτε θέση πρέπει να εκτελεί τρεις κινήσεις :

1. Κυκλική κίνηση, για την εύρεση της θέσης στο ανάλογο όροφο.
2. Κάθετη κίνηση (πάνω- κάτω), για την εύρεση του κατάλληλου ορόφου.
3. Οριζόντια κίνηση ενός άξονα (μέσα - έξω) για την παραλαβή ή την τοποθέτηση της βάσης που φέρει το αμάξωμα.

Στην περίπτωση που θέλουμε να τοποθετήσουμε ένα όχημα σε μια θέση στάθμευσης. Φέρουμε την βάση στην είσοδο, το αμάξωμα ανεβαίνει σε αυτήν και με την κατάλληλη εντολή από ένα πληκτρολόγιο ξεκινά η διαδικασία. Αφού γίνει η παραλαβή της βάσης και εκτελεστούν οι κινήσεις 1,2 (εύρεση θέσης), φροντίζουμε να είμαστε λίγο ψηλότερα από την θέση, εκτελούμε την κίνηση 1 δηλαδή φέρουμε πάνω από την θέση την βάση με το αμάξωμα και κατεβαίνουμε αργά επιτυγχάνοντας έτσι την τοποθέτηση της, τέλος μαζεύουμε το άξονα. Δια αυτό τον τρόπο γίνεται η τοποθέτηση μιας βάσης.

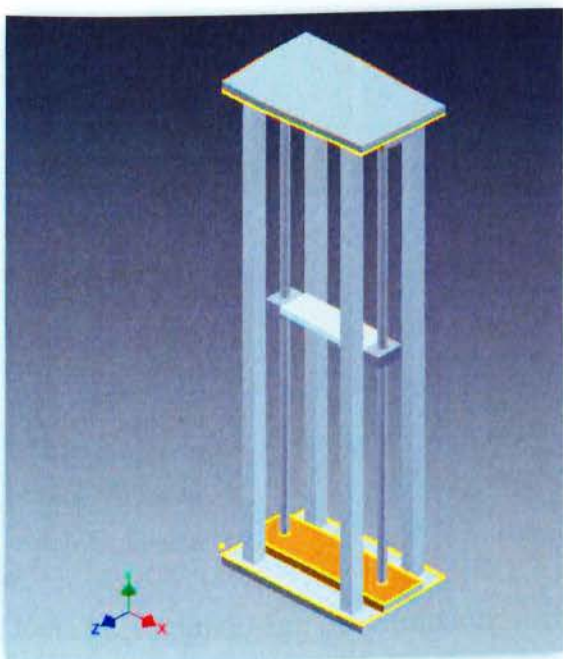
Στην περίπτωση της αποτοποθέτησης (παραλαβή) μιας βάσης εκτελούμε την ίδια διαδικασία μονό που αυτή την φορά, φροντίζουμε να είμαστε λίγο πιο χαμηλά από την θέση που επιθυμούμε. Αφού βγάλουμε τον άξονα, ανεβαίνουμε σιγά σιγά παραλαμβάνοντας την βάση και εκτελώντας την κίνηση 3 (μέσα) φέρουμε την βάση στο αναβατόριο.



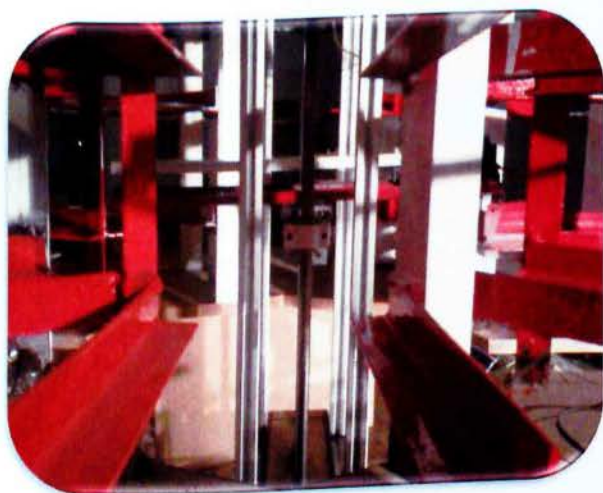
Στην εικόνα φαίνεται η εξαγωγή του άξονα σε μια θέση αλλά χωρίς να φέρει βάση .

Κεφάλαιο 2 – Μηχανισμός κίνησης (αναβατόριο)

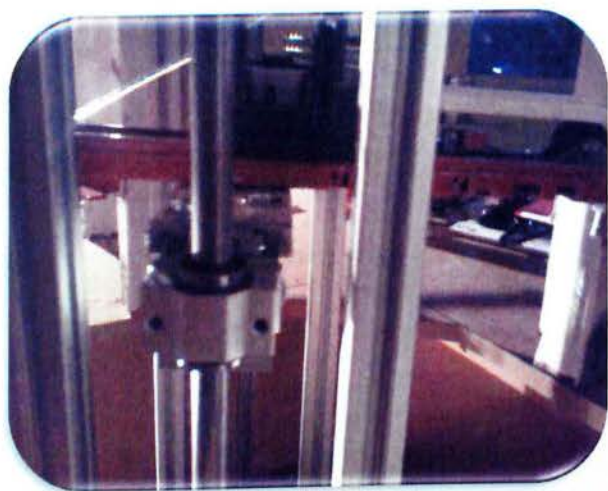
Η δομή του αναβατορίου.



Αριστερά μας φαίνεται το αναβατόριο. Αποτελείται από δυο φύλλα αλουμινίου (15X25 cm πάχους 15 mm), το ένα αποτελεί την βάση του μηχανισμού και το άλλο την κορυφή του. Οι 4 ορθογώνιοι πυλώνες (αλουμινίου) που παραβάλλονται ανάμεσα στα δυο φύλλα, στηρίζουν το μηχανισμό ενώνουν την βάση με την κορυφή και με την βοήθεια του ξύλινου κομματιού κρατούν σε κάθετη θέση, τις δυο μεταλλικές ράβδους. Οι δυο κάθετες μεταλλικές ράβδοι με επάλειψη γράσου και την βοήθεια δυο ρουλεμάν, που επάνω τους ενώνεται η κεντρική λάμα χρησιμοποιούνται ως οδηγοί κάθετης κίνησης της λάμας.



Πιο κάτω φαίνονται τα γραμμικά ρουλεμάν και οι κάθετες μεταλλικές ράβδοι που χρησιμοποιήθηκαν.



Πάνω στην άσπρη λάμα στηρίξαμε τον μηχανισμό που όπως φαίνεται στην φωτογραφία δεξιά έχει κόκκινο χρώμα και τον πήραμε από ένα παλιό εκτυπωτή (printer) που βρισκόταν σε αχρηστία. Την λειτουργία θα εξηγήσουμε περαιτέρω στην παρακάτω ενότητα. Οπού θα αναλύσουμε την μετάδοση των τριών απαιτούμενων κινήσεων τις οποίες προαναφέραμε ενδεικτικά στο προηγούμενο κεφάλαιο.

Η μετάδοση κίνησης.

Το κτήριο μας όπως είναι λογικό παραμένει σταθερό. Συνεπώς και οι τρεις προαναφερθείσες κινήσεις γίνονται από τον κεντρικό μηχανισμό (αναβατόριο). Ας δούμε πως:



Κυκλική κίνηση: Εξωτερικά και στο κέντρο, της βάσης και της κορυφής του αναβατορίου προσκολληθήκαν μεταλλικές ράβδοι. Αυτές καταλήγουν και ενώνονται σε δυο ρουλεμάν UCF στηριγμένα στο κέντρο της βάσης και της κορυφής του κτηρίου, με αποτέλεσμα το αναβατόριο να παραμένει στο κέντρο και να του επιτρέπεται η κυκλική κίνηση. Για την επίτευξη της, προσκολλήθηκε στην βάση αλουμινίου μια τροχαλία χρονισμού.

Στην μεγάλη τροχαλία τοποθετήθηκε ένας μάντας που το συνδέει με μια άλλη μικρότερη τροχαλία χρονισμού που είναι συνδεδεμένη στον άξονα ενός βηματικού κινητήρα που μεταδίδει την κίνηση.

Η μεγάλη τροχαλία έχει διάμετρο 6cm και το μικρό 4cm. Για να κάνει η μεγάλη τροχαλία ένα κύκλο 360 μοιρών πρέπει να γυρίσει ο βηματικός κινητήρας 540 μοίρες. Άρα ο συντελεστής είναι 1,5 αφού $1,5 * 360 = 540$. Έτσι υπολογίζουμε τις μοίρες που θα δίνουμε στο βηματικό κινητήρα .

Πιο κάτω φαίνεται α) το κάτω ρουλεμάν και β) το πάνω ρουλεμάν συνδεδεμένα στον πάνω και κάτω άξονα του αναβατορίου.

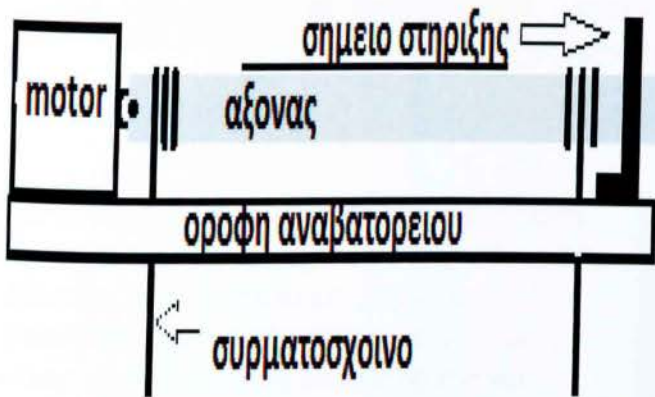




A)



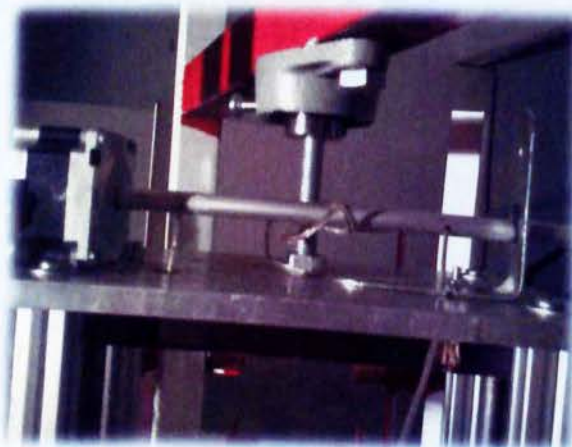
B)



Κάθετη κίνηση: Στην οροφή του αναβατορίου είναι τοποθετημένος ένας βηματικός κινητήρας. Στον κινητήρα είναι συνδεδεμένος ένας άξονας που στηρίζεται σε έναν οδηγό από την άλλη πλευρά.

Όπως φαίνεται και στο σχέδιο, πάνω στο άξονα έχουν προσκολληθεί συρματοσχοίνα τα όποια καταλήγουν στην λάμα. Συνεπώς όταν περιστρέφεται ο κινητήρας τα συρματοσχοίνα τυλίγονται γύρω του και ανεβάζουν την λάμα μαζί με τον μηχανισμό από τον printer. Περιστρέφοντας ανάποδα τον κινητήρα τα συρματοσχοίνα ξετυλίγονται και με την βοήθεια της βαρύτητας κατεβαίνει η λάμα που μεταφέρει τον μηχανισμό οριζόντιας κίνησης. Κατ' αυτόν τον τρόπο επιτυγχάνουμε την κάθετη κίνηση (πάνω- κάτω).

Στην εικόνα φαίνεται ο μηχανισμός ανύψωσης



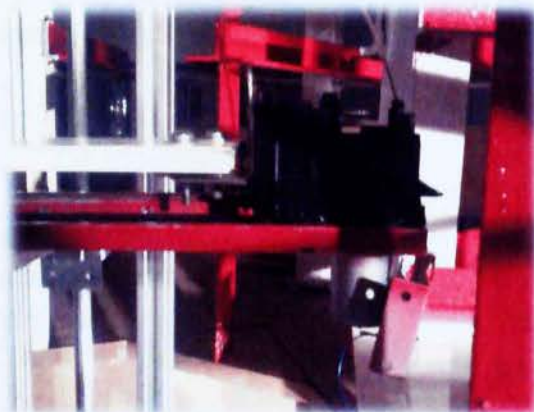
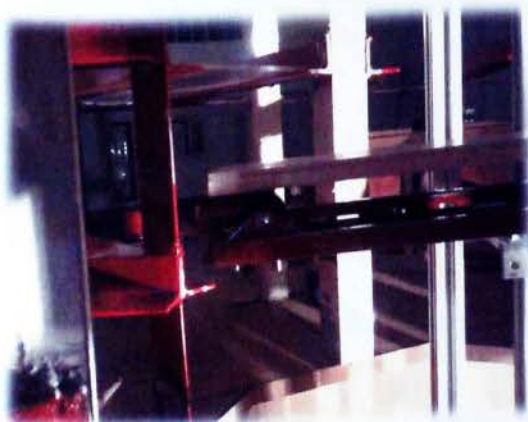


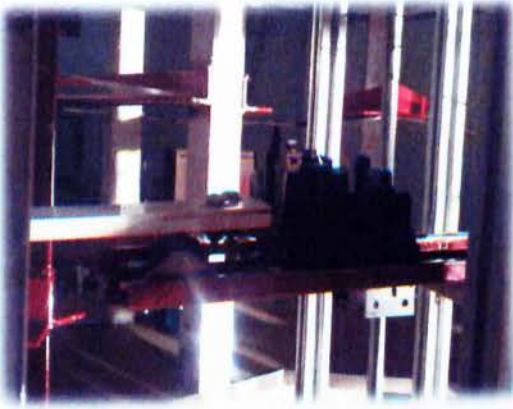
Οριζόντια κίνηση: Πάνω στην κόκκινη βάση στις δυο άκρες του εξαρτήματος του εκτυπωτή είναι τοποθετημένα δυο γρανάζια. Αυτά ενώνονται μεταξύ τους με έναν ιμάντα. Στον ιμάντα είναι κολλημένο το μαύρο εξάρτημα όπως φαίνεται και στην εικόνα. Το πίσω γρανάζι περιστρέφεται από έναν DC κινητήρα με αποτέλεσμα να περιστρέφει τον ιμάντα, αυτός με την σειρά του μεταφέρει το εξάρτημα πίσω όταν περιστρέφεται αριστερόστροφα και μπροστά όταν περιστρέφεται δεξιόστροφα.

Για να μπορούμε να παραλάβουμε μια βάση και να φτάσουμε σε μια θέση, τερματίσαμε εξάρτημα προς τα πίσω και πάνω του τοποθετήσαμε ένα άξονα που φθίνει έως μπροστά. Επομένως το αναβατόριο είναι ικανό να τοποθετήσει- παραλάβει μια βάση μόνο από την μπροστινή πλευρά.

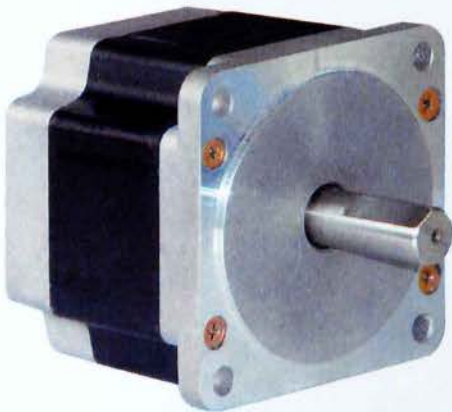


Στις εικόνες φαίνεται ο μηχανισμός όταν τερματίζει μέσα





Στην αριστερή εικόνα φαίνεται ο μηχανισμός όταν τερματίζει μέσα και στη δεξιά εικόνα φαίνετε ο DC κινητήρας που δημιουργεί τη ευθύγραμμη κίνηση.



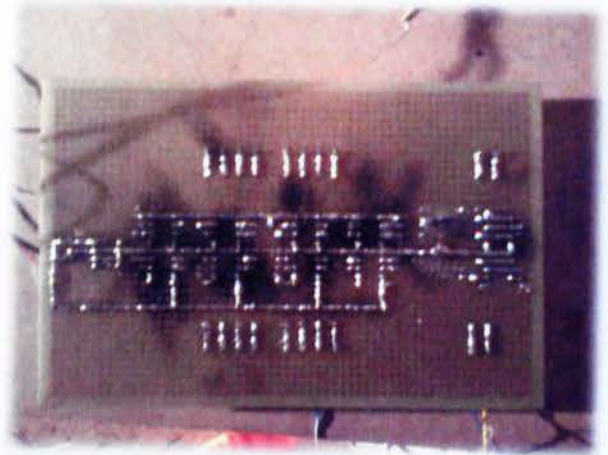
Για την ανύψωση και την κυκλική περιστροφή χρησιμοποιήσαμε βηματικούς κινητήρες, διότι μας δίνουν την δυνατότητα να περιστρέφουμε τον κινητήρα μας κατά τόσες μοίρες που εμείς επιθυμούμε. Κατά αυτόν τον τρόπο μπορούμε να βρούμε την θέση που θέλουμε να στείλουμε μια βάση χωρίς να χρειάζονται τερματικοί διακόπτες και επαφές έλεγχου. Επίσης περιορίζουμε τις εισόδους προς τον μικροεπεξεργαστή μας.

➤ Την λογική- νημοσύνη του συστήματος κίνησης την οποία θα αναλύσουμε σε παρακάτω κεφάλαιο, την παρέχουμε στο σύστημα μας με την βοήθεια ενός ARDUINO MEGA .

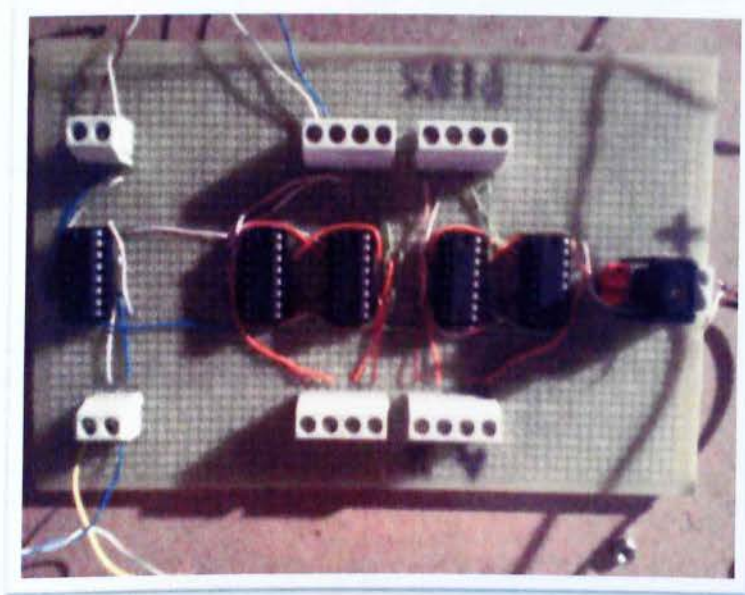
Κεφάλαιο 3 – Συνδέσεις κυκλώματα

Τροφοδοσία ηλεκτρικών – ηλεκτρονικών στοιχείων.

Η τροφοδοσία των ηλεκτρονικών – ηλεκτρικών στοιχείων γίνεται με την βοήθεια ενός τροφοδοτικού AC-DC των 24 volt και μιας αυτοσχέδιας πλακέτας.



Το τροφοδοτικό συνδέεται στα 220 volt AC και στην συνέχεια στην πλακέτα μας στην οποία παρέχει 24 volt DC και 4 Amper . Η πλακέτα κάνει τους ανάλογους μετασχηματισμούς και τροφοδοτεί τα στοιχεία μας με την χρήση voltage regulators. Όλες οι είσοδοι και έξοδοι από τα ολοκληρωμένα κυκλώματα καταφθάνουν σε κλέμες σύνδεσης όπου από αυτές με καλώδια συνδέονται στις κλεμοσειρές του μικροπεξεργαστή.



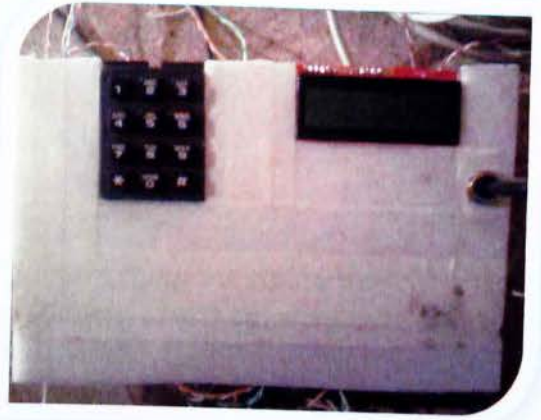
Στην εικόνα φαίνεται η αυτοσχέδια πλακέτα που φτιάξαμε

Οθόνη - Πληκτρολόγιο

Στην φωτογραφία αριστερά φαίνονται οι συνδέσεις της οθόνης LCD και του πληκτρολογίου μας από όπου θα δίνονται οι κατάλληλες εντολές και κωδικοί για από και προς στάθμευση.

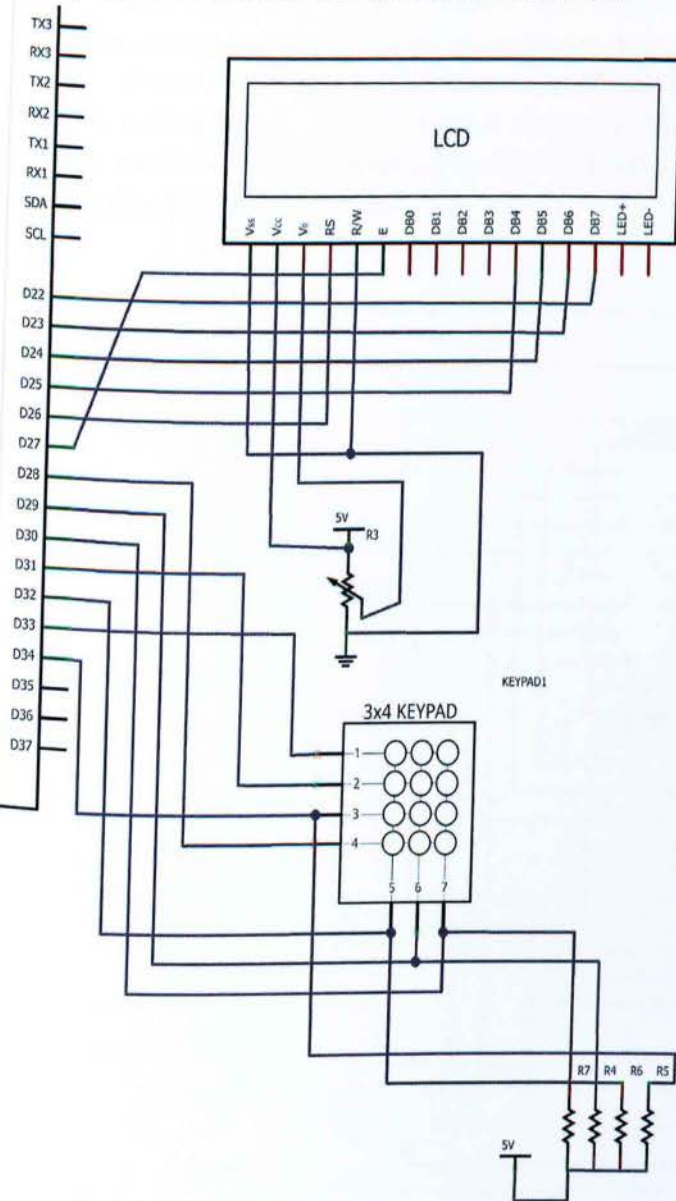
Οι τροφοδοσίες άλλα και οι γειώσεις συνδέονται στην πλακέτα μας.

Ο διαιρέτης τάσης (ποτενσιόμετρο) που παρεμβάλλεται στην Vcc χρησιμοποιείται για



την ρύθμιση φωτεινότητας της οθόνης.

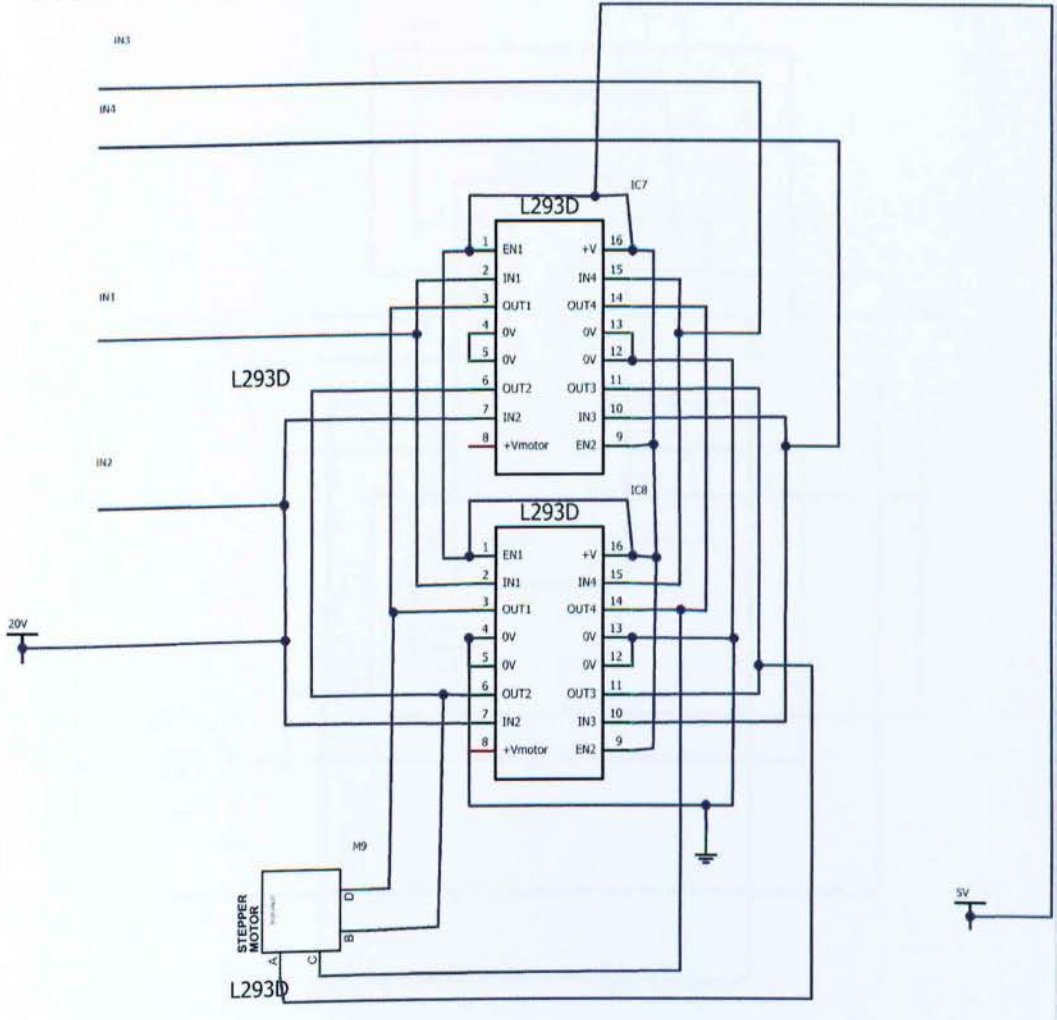
Οι αντιστάσεις που παρεμβάλλονται στην τροφοδοσία των μπουτόν έχουν ως σκοπό την αποφυγή «καψίματος» του πληκτρολογίου.



ΒΙΒΛΙΟΘΗΚΗ
ΤΕΙ ΠΕΙΡΑΙΑ

Οδηγοί βηματικών κινητήρων (stepper drivers)

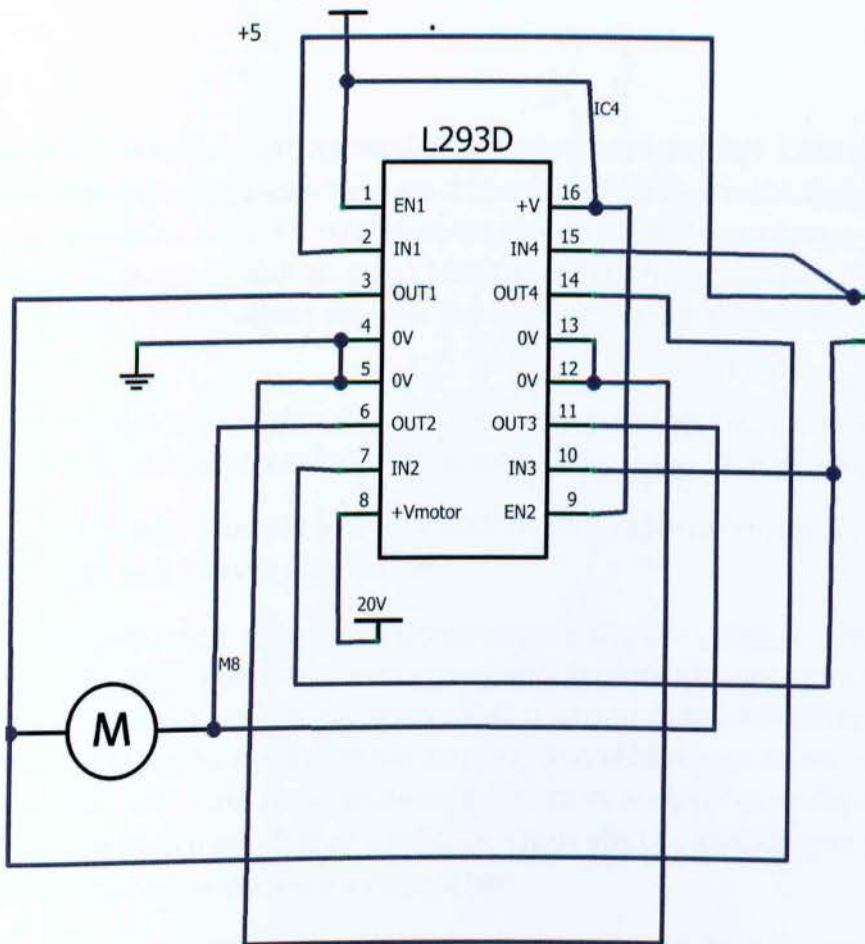
Για να φτιάξουμε τους οδηγούς των βηματικών κινητήρων χρησιμοποιήσαμε το ολοκληρωμένο L293D. Κάθε οδηγός περιέχει από ένα ζεύγος L293D και είναι συνδεδεμένα μεταξύ τους παράλληλα. Με αυτό τον οδηγό μπορούμε να ελέγξουμε την ταχύτητα του βηματικού κινητήρα, την κατεύθυνση του και φυσικά τα βήματα του. Στο πρόγραμμα γίνεται μετατροπή από βήματα σε μοίρες ανάλογα με το βηματικό κινητήρα που χρησιμοποιούμε. Τα IN 1,2,3,4 είναι συνδεδεμένα στον μικροεπεξεργαστή και παίρνουν εντολές μέσω αυτού. Τα OUT 1,2,3,4 είναι οι έξοδοι που τροφοδοτούν τα τυλίγματα του κινητήρα. Στο +Vmotor δίνουμε τροφοδοσία 20 Volt τα οποία είναι και η τάση τροφοδοσίας των κινητήρων.



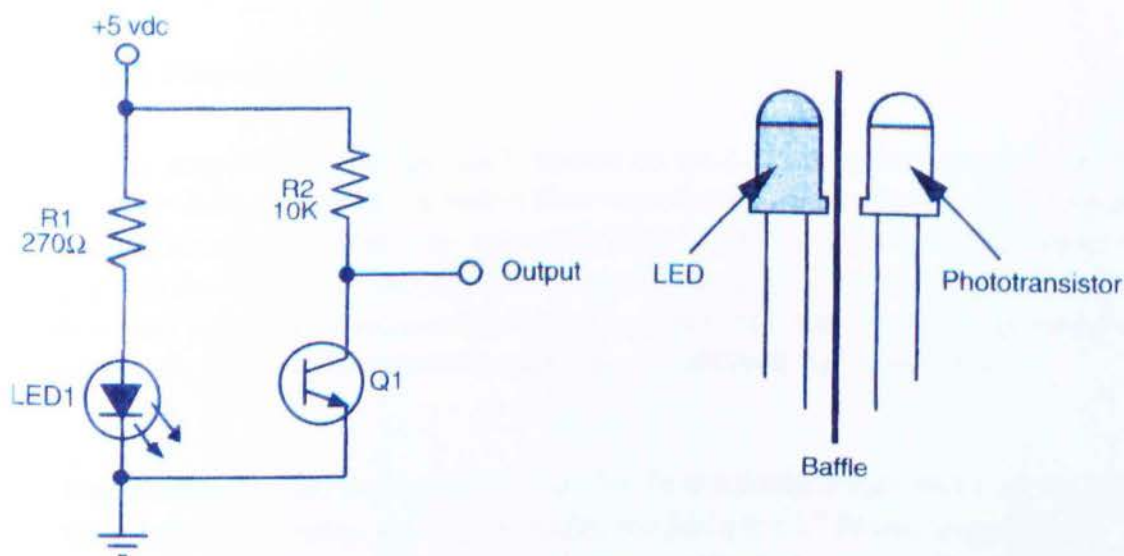
Οδηγός DC κινητήρα

Τον οδηγό για τον DC κινητήρα που χρησιμοποιούμε για την έξοδο – είσοδο του άξονα παλαβής και τοποθέτησης του αναβατορίου, τον κατασκευάσαμε με την βοήθεια του ολοκληρωμένου **L293d**.

Όπως βλέπουμε και στο σχήμα το OUT1 και OUT2 συνδέονται με τον DC. Ενώ οι εισόδοι IN1 και IN2 δέχονται εντολές από τον επεξεργαστή. Για IN1 = +5V ο κινητήρας περιστρέφεται δεξιόστροφα με αποτέλεσμα η βάση να βγαίνει προς τα έξω. Για IN2 = +5V έχουμε την αντίστροφη διαδικασία κίνησης. Στο pin8 έχουμε μόνιμα συνδεδεμένη την παροχή ρεύματος του κινητήρα (20 Volts). Τέλος το EN1,2 είναι συνδεδεμένο στα 5 Volt . Τα pin 4,5 γειώνονται. Επίσης για να έχουμε περισσότερη ισχύ εκμεταλλευτήκαμε το γεγονός ότι το ολοκληρωμένο μπορεί να ελέγξει και να τροφοδοτήσει δύο κινητήρες, έτσι συνδέσαμε παράλληλα τις εισόδους /εξόδους του κινητήρα A με τις εισόδους/εξόδους του κινητήρα B. Για να λειτουργήσει το ολοκληρωμένο χρειάζεται τάση 5 Volt στο pin 16 .



Αισθητήριο υπερύθρων



Όπως φαίνεται και στο σχήμα, χρησιμοποιήσαμε δύο αντιστάσεις των 270Ω και των 10KΩ. Συνδέσαμε ένα Led εκπομπής υπερύθρων και ένα φωτοτρανζίστορ παράλληλα. Τροφοδοτήσαμε το κύκλωμα με 5 V συνεχόμενου ρεύματος και διαχωρίσαμε με ένα πλαστικό κομμάτι το φωτοτρανζίστορ και το λαμπάκι εκπομπής υπερύθρων ώστε να μην επικοινωνούν. Στην έξοδο (Output) μετράμε την τάση που εφαρμόζεται στα άκρα του φωτοτρανζίστορ.

Το φωτοτρανζίστορ έχει την ιδιότητα να μεταβάλλει την συνολική του αντίσταση όταν ανιχνεύει φάσματα φωτός με αποτέλεσμα να μεταβάλλεται και η τάση στο σημείο Output.

Συνδέουμε το σημείο Output σε μία από τις εισόδους του μικροεπεξεργαστή μας έτσι ώστε να μετράμε την τάση που δίνει το αισθητήριο.

Το αισθητήριο εκπέμπει υπέρυθρες ακτίνες στον χώρο. Όταν δεν υπάρχει αντικείμενο αυτές δεν αντανακλούνται αλλά διαχέονται στον χώρο. Στην περίπτωση που υπάρχει αντικείμενο αυτές αντανακλούνται και ανιχνεύονται από το φωτοτρανζίστορ, αυτό με την σειρά του μεταβάλλει την αντίσταση του και την τάση εξόδου, την οποία εμείς μετράμε. Κατά αυτόν τον τρόπο καταλαβαίνουμε από το συγκεκριμένο αισθητήριο αν υπάρχει κάποιο αντικείμενο στα 7-10 cm. Βεβαίως έχουν γίνει οι απαραίτητες μετρήσεις τάσης εξόδου με αντικείμενο και άνευ αντικειμένου.

Το αισθητήριο αυτό χρησιμοποιείται για να διαπιστώσουμε αν υπάρχει ή μετακινήθηκε το αμάξιμα στην έξοδο.

Κεφάλαιο 4 – Η «Νοημοσύνη» του κτηρίου (Arduino).

Η σχετική νοημοσύνη.

Οι θέσεις του κτηρίου ξεκινούν απ' τον 1^ο όροφο και τους δίνεται μοναδικός αριθμός από το 1 έως το 32 ξεκινώντας από την πρώτη θέση αριστερά του 1^{ου} ορόφου (η οποία παίρνει τον αριθμό 1) και καταλήγοντας στην τελευταία θέση του 4^{ου} ορόφου (η οποία παίρνει τον αριθμό 32). Η λειτουργία του κτηρίου βασίζεται σε 2 κύριους τομείς, την παραλαβή από την είσοδο και τοποθέτηση του αμαξώματος σε μια θέση του παρκινγκ και την παραλαβή του αμαξώματος από μια θέση του παρκινγκ και τοποθέτηση του στην έξοδο.

- 1) Παραλαβή-Τοποθέτηση από τη είσοδο:** Το αναβατόριο έχει πάντα ως set point την είσοδο του κτηρίου και τοποθετημένη την βάση της 1^{ης} θέσης. Όταν δοθεί εντολή για το παρκάρισμα του 1^{ου} αμαξώματος εκτυπώνει μοναδικό κωδικό για την κάθε θέση ξεχωριστά, στην περίπτωση αυτή για την 1^η θέση. Μετά την τοποθέτηση του αμαξώματος το αναβατόριο παραλαμβάνει την επόμενη κενή βάση η οποία έχει τον κοντινότερο αριθμό στο 0 και την φέρνει στην είσοδο ώστε να είναι έτοιμη για την επόμενη τοποθέτηση. Επίσης αποθηκεύει σε μια θέση της ep-rom του arduino όπου έχουμε αντιστοιχήσει τις θέσεις, την τιμή 1 έτσι ξέρει ότι η εκάστοτε θέση είναι κατειλημμένη.
 - 2) Παραλαβή-Τοποθέτηση στην έξοδο:** Όταν δοθεί εντολή τοποθέτησης στην έξοδο και μετά την εισαγωγή του κωδικού που αντιστοιχεί στην επιθυμητή θέση, το αναβατόριο παραλαμβάνει το αμάξωμα και το φέρει στην έξοδο. Με την βοήθεια ενός αισθητηρίου υπερέθρων ελέγχουμε αν το αμάξωμα έχει αποσυρθεί, αφού γίνει αυτό το αναβατόριο επιστρέφει την βάση στην θέση της.
- Το αναβατόριο δεν εκτελεί πλήρη περιστροφή διότι υπάρχει περίπτωση να κοπούν τα καλώδια. Συνεπώς αφού τοποθετήσει κάποιο όχημα στις τελευταίες θέσεις ξανά γυρίζει προς τα πίσω

Προγραμματισμός-Επεξήγηση κώδικα (Arduino)

```
//=====LINBRARYS=====
```

```
#include <Keypad.h>
#include <EEPROM.h>
#include <LiquidCrystal.h>
#define DIR_PIN15 34
#define STEP_PIN15 36
#define DIR_PIN17 38
#define STEP_PIN17 40
#define STBY 9
#define PWMA 10
#define AIN1 11
#define AIN2 12
```

#include: δηλώνουμε τις βιβλιοθήκες που θα χρησιμοποιήσουμε

#define: ορίζουμε μεταβλητές στο αντίστοιχο PIN του Arduino

```
//=====KEYPAD_VARIABLES=====
```

```
const byte ROWS = 4; //four rows
const byte COLS = 3; //three columns
char keys[ROWS][COLS] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};
```

Ορίζουμε τις μεταβλητές που θα χρησιμοποιήσουμε το Keypad και τα PINS που θα χρησιμοποιήσουμε στον Arduino

```
byte rowPins[ROWS] = {5, 4, 3, 2}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {8, 7, 6}; //connect to the column pinouts of the keypad
```

```
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
```

```
//=====LCD=====
```

```
LiquidCrystal lcd(32, 30, 28, 26, 24, 22);
```

Ορίζουμε τα PINS που θα χρησιμοποιήσουμε η LCD στον Arduino

```
//=====PUBLIC_VARIABLES=====
```

```

int R;
int W;
int urn=0;
int upv=20;
int dov=-20;
int freee;
int cars;
char key;
String pass=" ";
int sens;//sensor
unsigned long StoredValue=0;//eeprom_couner
unsigned long w=0;//eeprom_counter
int deparkpos=0;
int a=0;
int b=0;//st15,st17;
int pos=0;
int St17[8] = {0,45,90,135,180,225,270,315};
int St15[4] = {0,90,180,270};
int ST15;
int ST17;
int ST17t;
int ST15t;
int dif=90;//diafora isogiou ipogiou se mires
String codes[32]=
{
  "0001","0002","0003","0004","0005","0006","0007","0008",
  "0009","0010","0011","0012","0013","0014","0015","0016",
  "0017","0018","0019","0020","0021","0022","0023","0024",
  "0025","0026","0027","0028","0029","0030","0031","0032"
};

```

Εδώ δηλώνουμε όλες τις δημόσιες μεταβλητές όπου τις «βλέπει» όλο το πρόγραμμα

//===VOID_SETUP=====

```
void setup() {
```

```
Serial.begin(9600);  
pinMode(DIR_PIN15, OUTPUT);  
pinMode(STEP_PIN15, OUTPUT);  
pinMode(DIR_PIN17, OUTPUT);  
pinMode(STEP_PIN17, OUTPUT);  
LiquidCrystal lcd(22, 27, 25, 24, 23, 26);  
lcd.begin(16, 2);  
pinMode(STBY, OUTPUT);  
pinMode(PWMA, OUTPUT);  
pinMode(AIN1, OUTPUT);  
pinMode(AIN2, OUTPUT);  
}
```

Στην συνάρτηση `void setup()`
δηλώνουμε την κατάσταση των PINS
του Arduino

```
//=====VOID_LOOP=====
```

```
void loop() {  
ReadFromEEPROM;  
Serial.print("EEPROM COUNTER:");Serial.println(w);  
Serial.println();  
epromread();  
countCars();  
mainMonitorLcd();  
mainMonitor();  
key=keypad.waitForKey();  
parkEvent(key);  
  
//delay(1000000);  
}
```

Η **loop** είναι η κύρια συνάρτηση μας
και περιμένει να δεχθεί κάποια
είσοδο από το πληκτρολόγιο για να
προβεί στις ανάλογες ενέργειες

```
//=====FUNCTIONS=====
```

Από εδώ και κάτω είναι η
συναρτήσεις που έχουμε
δημιουργήσει

//=====EEPROM_COUNTER=====

Για να έχουμε ένα αρχείο με τις γεμάτες και άδειες θέσεις στο **parking** χρησιμοποιούμε την μνήμη **EEPROM** του **Arduino** για να αποθηκεύουμε την συγκεκριμένη κατάσταση. Αυτό γίνεται χρησιμοποιώντας 32 θέσεις στην μνήμη **EEPROM (32 Bytes)** όπου το κάθε **byte** αντιστοιχεί σε μία θέση. Αν το **byte** αυτό είναι **0** τότε υπονοεί ότι η συγκεκριμένη θέση είναι άδεια. Αντιθέτως εάν είναι **1** η θέση είναι γεμάτη.

Σύμφωνα με τον κατασκευαστή η **EEPROM** έχει συγκεκριμένη διάρκεια ζωής όπου αυτή καθορίζεται ανάλογα με τις έγγραφες και τις διαγραφές που κάνουμε στην μνήμη **EEPROM**. Η μνήμη **EEPROM** έχει διάρκεια ζωής 100.000 έγγραφες/διαγραφές.

Για να ξέρουμε πόσες έγγραφες έχουμε κάνει φτιάξαμε ένα **counter** ο οποίος μας ενημερώνει για αυτό τον αριθμό. Στην περίπτωση που έχουμε ξεπεράσει το όριο αυτό οι θέσεις μνήμης που χρησιμοποιούμε δεν είναι αξιόπιστες και μπορούμε να ορίσουμε να γίνεται η αποθήκευση στα επόμενα 32 bytes τις μνήμης **EEPROM**. Η μνήμη **EEPROM** του **arduino mega** είναι 4kb. Εμείς χρησιμοποιούμε κάθε φορά 35 bytes. Τα 32 είναι για τις θέσεις και τα 3 για τον **counter**. Ο **counter** χρειάζεται 3bytes για να μπορεί να αναπαραστήσει τον αριθμό 100.000. Άρα με αυτά τα δεδομένα μπορούμε να κάνουμε περίπου 11.428.571 έγγραφές μέχρι ολόκληρη η μνήμη **EEPROM** να είναι αναξιόπιστη.

Η συνάρτηση **SaveToEEPROM** καλείται
όποτε γίνεται μια εγγραφή στην EEPROM
και δουλεία της είναι να αποθηκεύει τον
αριθμό του counter στα 3 bytes τις
μνήμης που κρατάμε αυτό τον αριθμό.

```
void SaveToEEPROM(unsigned long MyInteger){  
  
    //MyInteger is 4 byte (unsigned long int) so 3*8=24 bit long  
    //MyInteger LLLLLLLLXXXXXXXXXXXXXXXXXXXX  
  
    unsigned long wl;  
    wl=MyInteger<<8; // wl=XXXXXXXX0000000000000000  
    wl=wl>>16; // wl=0000000000000000XXXXXXXX  
  
    unsigned long wr;  
    wr=MyInteger<<16; // wr=YYYYYYYY0000000000000000  
    wr=wr>>16; // wr=0000000000000000YYYYYYYY  
  
    unsigned long w3;  
    w3=MyInteger>>16; // w3=0000000000000000LLLLLLLL  
  
    EEPROM.write(32,w3); //l write LLLLLLLL  
    EEPROM.write(33,wl); //l write XXXXXXXX  
    EEPROM.write(34,wr); //l write YYYYYYYY  
  
}
```

Η συνάρτηση **ReadFromEEPROM** καλείται
όποτε γίνεται μια εγγραφή στην **EEPROM**
και δουλειά της είναι να διαβάσει τον
αριθμό που είναι αποθηκευμένος στα
3bytes της μνήμης **EEPROM** που
αναπαριστάνουν τον **counter** ούτως ώστε
να αυξηθεί ο **counter** κατά ένα

```
void ReadFromEEPROM(){  
  
    unsigned long w3;  
    w3=EEPROM.read(32); // I read w3=0000000000000000LLLLLLLL  
    w3=w3<<16;    // w3=LLLLLLLL0000000000000000  
  
    unsigned long wl;  
    wl=EEPROM.read(33); // I read wl = 00000000XXXXXXXX  
    wl=wl<<8;    //wl=XXXXXXXX00000000  
  
    unsigned long wr;  
    wr=EEPROM.read(34); //I read wr=00000000YYYYYYYY  
  
    w=w3+wl+wr;  
    //w=LLLLLLLL0000000000000000+XXXXXXXXX00000000+00000000YYYYYYYY=LLLLLLLLXXXXXX  
    XYYYYYYYYY  
}
```

Η συνάρτηση **eepromread** καλείται για να διαβάσει την μνήμη **eeprom** και να αναπαραστήσει τις κενές και γεμάτες θέσεις

```
void eepromread(){  
for (int i=0;i<32;i++){  
    int r=EEPROM.read(i);  
    Serial.print(r);  
    int l = (i+1) % 8;  
    if (l==0)  
        Serial.println();  
}  
}
```

Η συνάρτηση **eepromRup** καλείται για να διαβάσει την μνήμη **eeprom** και να βρει την πρώτη άδεια θέση

```
void eepromRup(int i)  
{  
    for (i;i<32;i++){  
        R=EEPROM.read(i);  
        if (R==0)  
        {  
            pos = i;  
            i=32;  
        }  
    }  
}}
```


Η **parkEvent** είναι μια συνάρτηση η οποία βάση των εντολών που θα δεχθεί από το πληκτρολόγιο καλεί τις ανάλογες συναρτήσεις οι οποίες κάνουν τις προγραμματισμένες λειτουργίες

```
void parkEvent(KeypadEvent key){
  switch (keypad.getState()){
    case PRESSED:
      switch (key){
        case '#':
          if (free==0)
          {
            FullMonitorlcd();
            FullMonitor();
            delay(2000);
          }
        else
          {
            eppromRup(0);
            parkMonitorlcd();
            parkMonitor();
            park();
          }
          break;

        case '*':
          if (cars==0){
            emptyMonitorlcd();
            emptyMonitor();
            delay(2000);
          }
        else
          {
            pass=" ";
            codeMonitorlcd();
            codeMonitor();
          }
      }
    }
}
```

```

for (int i=0;i<4;i++){
    key=keypad.waitForKey();
    pass.setCharAt(i,key);
}
codeMonitorlcd();
codeMonitor();
delay(2000);
for (int i=0;i<32;i++)
{
    R=EEPROM.read(i);

    if ((R==1) && (codes[i]==pass))
    {
        deparkpos=i;
        deparkMonitorlcd();
        deparkMonitor();
        depark();
        i=32;
    }
    else
    {
        deparkpos=-1;
    }
}

if (deparkpos==-1)
{
    wrongMonitorlcd();
    wrongMonitor();
    delay(2000);
}
break;

}
break;
}}}

```

Η **countcars** περιέχει τους μετρητές cars και free οι οποίοι μετράνε τις γεμάτες και άδειες θέσεις αντίστοιχα

```
void countCars(){
  free=0;
  cars=0;
  for (int i=0;i<32;i++){
    R=EEPROM.read(i);
    if (R==0)
      freee++;
    else
      cars++;
  }
}
//=====SERIAL_MONITOR=====
```

Οι παρακάτω συναρτήσεις μας βοηθούν μέσω του Serial Monitor να βλέπουμε ότι παρουσιάζεται και στην LCD

```
void wrongMonitor(){
  Serial.println("=====");
  Serial.println("WRONG PASSWORD");
  Serial.println();
  Serial.println("=====");
}
```

Παρουσιάζεται όταν ο χρήστης πληκτρολογήσει λάθος κωδικό

```
void mainMonitor(){
  Serial.println("=====");
  Serial.print("#PARK|FREE:");Serial.println(freee);
  Serial.print("*GETCAR|CARS:");Serial.println(cars);
  Serial.println("=====");
}
```

Παρουσιάζεται σαν κεντρικό μενού

```
void parkMonitor(){
```



```
Serial.println("=====");
Serial.print("PARK ON SLOT:"); Serial.println(pos);
Serial.print("YOUR
CODE:");Serial.println(codes[pos]);
Serial.println("=====");
}
```

Παρουσιάζεται όταν ξεκινήσει η διαδικασία παρκαρίσματος και ενημερώνει τον χρήστη σε ποια θέση θα τοποθετηθεί το αμάξι και του δίνει και ένα μυστικό κωδικό για να τον χρησιμοποιήσει όταν έρθει να παραλάβει το αμάξι

```
void codeMonitor(){
Serial.println("=====");
Serial.println("ENTER CODE: ");
Serial.println(pass);
Serial.println("=====");
}
```

Παρουσιάζεται όταν ο χρήστης θέλει να παραλάβει το αμάξι και του ζητά να εισάγει το μυστικό κωδικό

```
void deparkMonitor(){
Serial.println("=====");
Serial.print("CAR ON SLOT ");Serial.println(deparkpos);
Serial.println("IS COMING.WAIT..");
Serial.println("=====");
}
```

Παρουσιάζεται όταν εκτελείται η διαδικασία τοποθέτησης του αυτοκινήτου στην έξοδο

```
void FullMonitor(){
Serial.println("=====");
Serial.println("PARKING FULL");
Serial.println("TRY LATER...");
Serial.println("=====");
}
```

Παρουσιάζεται όταν χρήστης δώσει εντολή να παρκάρει το αμάξι του αλλά το παρκινγκ είναι γεμάτο

```
void emptyMonitor(){
Serial.println("=====");
Serial.println("PARKING IS EMPTY");
Serial.println("");
Serial.println("=====");
}
```

Παρουσιάζεται όταν ο χρήστης δώσει εντολή να πάρει το αμάξι του αλλά το παρκινγκ είναι άδειο

Οι παρακάτω συναρτήσεις
τυπώνουν στην LCD τα κατάλληλα
μηνύματα

```
//=====LCD_MONITOR=====
```

```
void wrongMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("WRONG PASSWORD");  
  lcd.setCursor(0,1);  
  lcd.print("");  
}
```

Παρουσιάζεται όταν ο χρήστης
πληκτρολογήσει λάθος κωδικό

```
void mainMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0, 0);  
  lcd.print("#PARK|FREE:");lcd.print(free);  
  lcd.setCursor(0, 1);  
  lcd.print("*GETCAR|CARS:");lcd.print(cars);  
}
```

Παρουσιάζεται σαν κεντρικό μενού

```
void parkMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("PARK ON SLOT:"); lcd.print(pos);  
  lcd.setCursor(0,1);  
  lcd.print("YOUR CODE:");lcd.print(codes[pos]);  
}
```

Παρουσιάζεται όταν ξεκινήσει η
διαδικασία παρκαρίσματος και
ενημερώνει τον χρήστη σε ποια
θέση θα τοποθετηθεί το αμάξι και
του δίνει και ένα μυστικό κωδικό
για να τον χρησιμοποιήσει όταν
έρθει να παραλάβει το αμάξι

```
void codeMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("ENTER CODE: ");  
  lcd.setCursor(0,1);  
  lcd.print(pass);  
}
```

Παρουσιάζεται όταν ο χρήστης θέλει να παραλάβει το αμάξι και του ζητά να εισάγει το μουσικό κωδικό

```
void deparkMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("CAR ON SLOT ");lcd.print(deparkpos);  
  lcd.setCursor(0,1);  
  lcd.print("IS COMING.WAIT..");  
}
```

Παρουσιάζεται όταν εκτελείται η διαδικασία τοποθέτησης του αυτοκινήτου στην έξοδο

```
void FullMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("PARKING FULL");  
  lcd.setCursor(0,1);  
  lcd.print("TRY LATER...");  
}
```

Παρουσιάζεται όταν χρήστης δώσει εντολή να παρκάρει το αμάξι του αλλά το παρκινγκ είναι γεμάτο

```
void emptyMonitorlcd(){  
  lcd.begin(16,2);  
  lcd.setCursor(0,0);  
  lcd.print("PARKING IS EMPTY");  
  lcd.setCursor(0,1);  
  lcd.print("");  
}
```

Παρουσιάζεται όταν ο χρήστης δώσει εντολή να πάρει το αμάξι του αλλά το παρκινγκ είναι άδειο

```
//=====
```

Η συνάρτηση αυτή καλείται όταν γίνει το τοποθέτηση τις τελευταίας πλατφόρμας στο παρκινγκ, έτσι δεν υπάρχει κάποια άλλη πλατφόρμα για να πάει στην είσοδο, οπότε το αναβατήριο επιστρέφει χωρίς πλατφόρμα στη είσοδο.

```
void full(){  
  rotateDeg17(-St17[ST17t],1); //stepper17  
  delay(50);  
  rotateDeg15(-St15[ST15t]-dif,1); //stepper15  
  delay(50);  
}
```

```
//=====
```

Η συνάρτηση **rotateDeg15** παίρνει παραμέτρους γωνίας και ταχύτητας και όπου καλείται κινεί το βηματικό κινητήρα που ανυψώνει το αναβατήριο

```
void rotateDeg15(float deg, float speed) {
```

```
//rotate a specific number of degrees (negative for reverse movement)  
//speed is any number from .01 -> 1 with 1 being fastest - Slower is stronger
```

```
int dir = (deg > 0)? HIGH:LOW;  
digitalWrite(DIR_PIN15,dir);
```

```
int steps = abs(deg)*(1/0.1125);  
float usDelay = (1/speed) * 200;//70
```

```
for(int i=0; i < steps; i++){  
  digitalWrite(STEP_PIN15, HIGH);  
  delayMicroseconds(usDelay);
```

```
  digitalWrite(STEP_PIN15, LOW);  
  delayMicroseconds(usDelay);
```

```
}
```

```
}
```

Η συνάρτηση **rotateDeg17** παίρνει
 παραμέτρους γωνίας και ταχύτητας
 και όπου καλείται κινεί το βηματικό
 κινητήρα που γυρίζει το
 αναβατόριο

```
void rotateDeg17(float deg, float speed){
  //rotate a specific number of degrees (negative for reverse movement)
  //speed is any number from .01 -> 1 with 1 being fastest - Slower is stronger
  int dir = (deg > 0)? HIGH:LOW;
  digitalWrite(DIR_PIN17,dir);

  int steps = abs(deg)*(1/0.05625);
  float usDelay = (1/speed) * 200; //70

  for(int i=0; i < steps; i++){
    digitalWrite(STEP_PIN17, HIGH);
    delayMicroseconds(usDelay);

    digitalWrite(STEP_PIN17, LOW);
    delayMicroseconds(usDelay);
  }
}
//=====
```

Η **eepromW** αποθηκεύει στην
 EEPROM την κατάσταση των
 θέσεων στο παρκινγκ και αυξάνει
 τον κατά ένα τον counter της
 EEPROM

```
void eepromW(int a,int b){
  EEPROM.write(a,b);
  ReadFromEEPROM();
  StoredValue=w+1;
  SaveToEEPROM(StoredValue);
}
```

//=====

Η συνάρτηση **take_place_platform** χρησιμοποιείται στην περίπτωση που θέλουμε να πάρουμε μία πλατφόρμα και στην περίπτωση που θέλουμε να αφήσουμε μία πλατφόρμα.

```
void take_place_platform(int t_or_p){//kinisi tis platformas p=1 +10 out/in p=2 -10 out/in
Serial.print("KINISI ACTUATOR:");
if (t_or_p==1){
Serial.println("OUT/UP/IN");
//(1)dc eksw
move(1, 255, 1); //motor 1, full speed, left
delay(3000); //go for 1 second
stop(); //stop
//(2)paei panw 10cm h plaforma
rotateDeg15(upv,1);
delay(2000);
//(3)dc mesa
move(1, 255, 0); //motor 1, full speed, right
delay(3000);
stop();
}
else if (t_or_p==2){
Serial.println("OUT/DOWN/IN");
//(1)dc eksw
move(1, 255, 1); //motor 1, full speed, left
delay(3000); //go for 1 second
stop(); //stop
//(2)paei katw -10cm h plaforma
rotateDeg15(dov,1);
delay(2000);
//(3)dc mesa
move(1, 255, 0); //motor 1, full speed, right
delay(3000);
stop();
}
}
```

Ο σκοπός της **find_platform** είναι να βρίσκει την πιο κοντινή κενή θέση και να ενημερώνει τις μοίρες που πρέπει να μετακινηθούν οι βηματικοί για να πάνε σε αυτή τη θέση.

```
void find_plaform (){  
  
  eppromRup(0);  
  ST15 = (pos/ 8);  
  ST17 = (pos % 8);  
  Serial.print("(1) KENI THESI:OROFOS ");Serial.print(ST15);Serial.print(" DIAMERISMA  
");Serial.println(ST17);  
}
```

Σκοπός της **find_near_platform** είναι όταν το αναβατόριο αφήσει το αμάξι σε κάποια θέση να το στείλει να παραλάβει την πιο κοντινή άδεια πλατφόρμα από την αρχική θέση

```
void find_near_platform()  
{  
  a = (pos/ 8);  
  b = (pos % 8);  
  ST15=St15[a]-St15[ST15t];//degrees  
  ST17=St17[b]-St17[ST17t];//++  
  Serial.print("OROFOS ");Serial.print(a);Serial.print(" DIAMERISMA ");Serial.println(b);  
  Serial.println("(8) KINISI PROS TIN KENI PLATFORMA");Serial.print(" KINISI STEPPER17  
:");Serial.print(ST17);Serial.println(" DEGREES"); //stepper17  
  Serial.print(" KINISI STEPPER15 :");Serial.print(ST15);Serial.println(" DEGREES");  
  //stepper17  
}
```

Η συνάρτηση **park** αναλαμβάνει την διαδικασία παρκαρίσματος του οχήματος και παραλαβής μιας άδειας πλατφόρμας όπου θα την επιστρέψει στην αρχική θέση

```

void park(){
Serial.println("=====PARKING=====");
find_plaform ();
Serial.print("(2) ");
take_place_platform(1);
Serial.print("(3) KINISI STEPPER17 :");Serial.print(St17[ST17]);Serial.println(" DEGREES");
//stepper17
rotateDeg17(St17[ST17],1);
delay(2000);
Serial.print("(4) KINISI STEPPER15 :");Serial.print(St15[ST15]+dif);Serial.println("
DEGREES"); //stepper15
rotateDeg15(St15[ST15]+dif,1);
delay(2000);
Serial.print("(5) ");
take_place_platform(2);
eppromW(pos,1);
Serial.println("(6) EPPROMWRITE PARK POSITION");//write 1
ST15t = (pos/ 8);
ST17t = (pos % 8);
int posn=pos;
eppromRup(0);
if (posn==pos){
Serial.println("(7) PARKING FULL NO PLATFORMS");
full();
}
else
{
find_near_platform();Serial.print("(7) EVRESI KENIS PLATFORMAS GIA NA PAI STIN ARXIKI
THESI: ");
rotateDeg17(ST17,1); //stepper17
delay(2000);
rotateDeg15(ST15,1); //stepper15
delay(2000);
Serial.print("(9) ");
take_place_platform(1);

```

```

delay(2000);
Serial.println("(10) EPISTROFI STIN ARXIKI THESI:");
rotateDeg17(-St17[b],1);Serial.print(" KINISI STEPPER17 :");Serial.print(-
St17[b]);Serial.println(" DEGREES"); //stepper17
delay(2000);
rotateDeg15(-St15[a]-dif,1);Serial.print(" KINISI STEPPER15 :");Serial.print(-St15[a]-
dif);Serial.println(" DEGREES"); //stepper15
delay(2000);
Serial.print("(11) ");
take_place_platform(2);
}
Serial.println("=====");
}

```

Η συνάρτηση **depark** αναλαμβάνει την διαδικασία αποστάθμευσης του οχήματος και την επιστροφή της άδειας πλατφόρμας στην αρχική θέση. Η επιστροφή της πλατφόρμας γίνεται αφού το αμάξι φύγει από την πλατφόρμα μέσω ενός αισθητηρίου υπερύθρων.

```

void depark(){
Serial.println("=====DEPARK=====");
ST15= (deparkpos / 8);
ST17= (deparkpos % 8);
Serial.print("(1) FERNI TO AMAKSI APO TON OROFO ");Serial.print(ST15);Serial.print("
DIAMERISMA ");Serial.println(ST17);
Serial.print("(2) KINISI STEPPER17 :");Serial.print(St17[ST17]);Serial.println(" DEGREES");
//stepper17
Serial.print("(3) KINISI STEPPER15 :");Serial.print(St15[ST15]+dif);Serial.println("
DEGREES"); //stepper15
rotateDeg17(St17[ST17],1); //stepper17
delay(2000);
rotateDeg15(St15[ST15]+dif,1); //stepper15
delay(2000);
Serial.print("(4) ");
take_place_platform(1);
ST15=St15[0]-St15[ST15];
ST17=St17[4]-St17[ST17];
Serial.println("(5) PAI STIN ARXIKI THESI 4 GIA NA AFISI TO AMAKSI:");
}

```

```

Serial.print("(6) KINISI STEPPER17 :");Serial.print(ST17);Serial.println(" DEGREES");
//stepper17
Serial.print("(7) KINISI STEPPER15 :");Serial.print(ST15-dif);Serial.println(" DEGREES");
//stepper15
rotateDeg17(ST17,1); //stepper17
delay(2000);
rotateDeg15(ST15-dif,1); //stepper15
delay(2000);
Serial.print("(8) ");
take_place_platform(2);//aisthitirio
Serial.println("(9) PERIMENI TO AMAKSI NA FIGI APO TIN PLATFORMA");
while (sens>sensval)
{delay(1000);
}
delay(1000);
Serial.println("(11) KSANAPERNI TIN PLATFORMA GIA NA TIN PAREI STIN THESI THS");
Serial.print("(12) ");
take_place_platform(1);
Serial.print("(13) KINISI STEPPER17 :");Serial.print(-ST17);Serial.println(" DEGREES");
//stepper17
Serial.print("(14) KINISI STEPPER15 :");Serial.print(-ST15+dif);Serial.println(" DEGREES");
//stepper15
rotateDeg17(-ST17,1); //stepper17
delay(2000);
rotateDeg15(-ST15 + dif,1); //stepper15
Serial.print("(15) ");
take_place_platform(2);//-10 thesi pou afisame tin platforma
Serial.println("(16) EPPROMWRITE DEPARTK POSITION");
eppromW(departkpos,0);//write 0
Serial.println("(17) KINISI PISW STIN ARXIKI THESI 0:");
ST15= (departkpos / 8);
ST17= (departkpos % 8);
Serial.print("(18) KINISI STEPPER17 :");Serial.print(-St17[ST17]);Serial.println(" DEGREES");
//stepper17
Serial.print("(19) KINISI STEPPER15 :");Serial.print(-St15[ST15]-dif);Serial.println("
DEGREES"); //stepper15
rotateDeg17(-St17[ST17],1); //stepper17
delay(2000);
rotateDeg15(-St15[ST15]-dif,1); //stepper15
delay(2000);
Serial.println("=====");
}

```

Σκοπός της συνάρτησης **move** είναι να ελέγχει την κίνηση του **actuator** μέσα η έξω ανάλογα με την λειτουργία που θέλουμε να κάνουμε

```
void move(int motor, int speed, int direction){
//Move specific motor at speed and direction
//motor: 0 for B 1 for A
//speed: 0 is off, and 255 is full speed
//direction: 0 clockwise, 1 counter-clockwise

digitalWrite(STBY, HIGH); //disable standby

boolean inPin1 = LOW;
boolean inPin2 = HIGH;

if(direction == 1){
  inPin1 = HIGH;
  inPin2 = LOW;
}

if(motor == 1){
  digitalWrite(AIN1, inPin1);
  digitalWrite(AIN2, inPin2);
  analogWrite(PWMA, speed);
}
}

void stop(){
//enable standby
digitalWrite(STBY, LOW);
}
```

Κεφάλαιο 5 – Οδηγίες χρήσης

Όπως προαναφέραμε η φωτεινότητα της οθόνης ρυθμίζεται από ένα ποτενσιόμετρο στο πλαίσιο (οθόνη – πληκτρολόγιο – ποτενσιόμετρο) που έχουμε φτιάξει. Παρακάτω θα αναλύσουμε τις οδηγίες χρήσης του συστήματος και θα δείξουμε ποιες είναι οι σωστές ενδείξεις της οθόνης στις κατάλληλες εντολές υπό ορθή λειτουργία.

Κεντρική περίπτωση



Τροφοδοτώντας με ρεύμα το σύστημά μας στην οθόνη εμφανίζονται τα μηνύματα που φαίνονται αριστερά. Πατώντας '#' του δίνουμε εντολή προς στάθμευση. Δίπλα από το park ο αριθμός στο free δηλώνει τις κενές θέσεις.

Περιπτώσεις προς στάθμευση



Το επόμενο μήνυμα που εμφανίζεται είναι το 'park on slot' που υποδηλώνει ότι αμάξι θα πάει στην θέση 22. Και το 'your code' δίνει στον χρήστη των κωδικό τον οποίο θα χρησιμοποιήσει για την παραλαβή του σχήματος του.



Στην εξαιρετική περίπτωση που το κτήριο μας είναι γεμάτο τότε εμφανίζεται το μήνυμα που βρίσκεται αριστερά.

Περίπτωση αποστάθμευσης



Δίνοντας την εντολή "*" από το πληκτρολόγιο, ζητά τον κωδικό που έχει πάρει ο χρήστης και αν είναι σωστός ξεκινά η διαδικασία αποστάθμευσης.



Στην περίπτωση που ο χρήστης εισάγει λάθος κωδικό παρουσιάζεται το μήνυμα στα αριστερά. Και επιστρέφει στην κεντρική οθόνη μετά από 2sec.



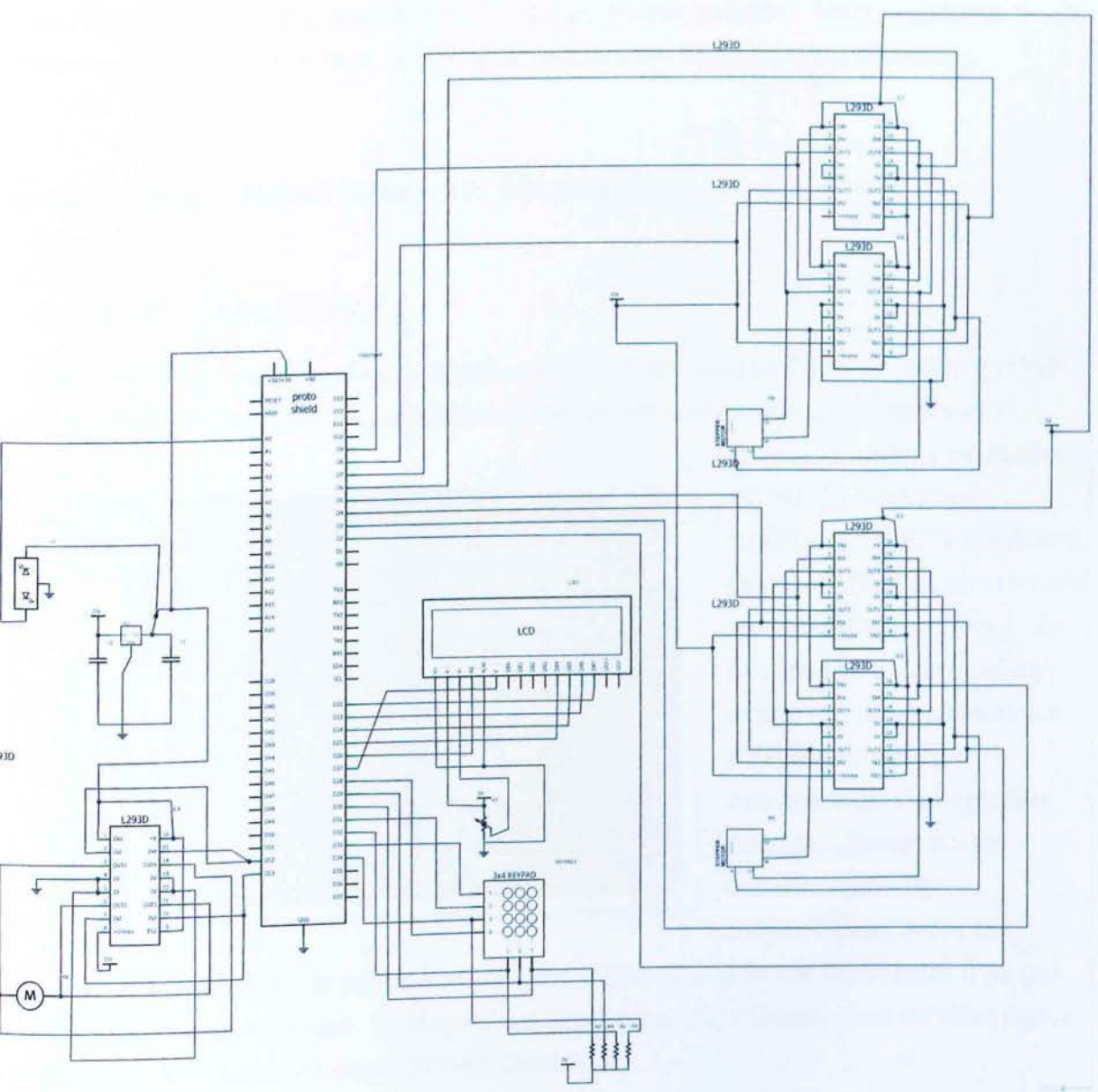
Στην περίπτωση που ο κωδικός είναι σωστός ξεκινάει η διαδικασία αποστάθμευσης. Η οθόνη παρουσιάζει το μήνυμα αριστερά και ο αριθμός 22 υποδηλώνει την θέση από την οποία έρχεται το όχημα.



Στην εξαιρετική περίπτωση που ο χρήστης δώσει εντολή αποστάθμευσης και το παρκινγκ είναι άδειο τότε εμφανίζεται το εξής μήνυμα.

Κεφάλαιο 6 - Ολικό κύκλωμα και εξαρτήματα

Ολικό κύκλωμα



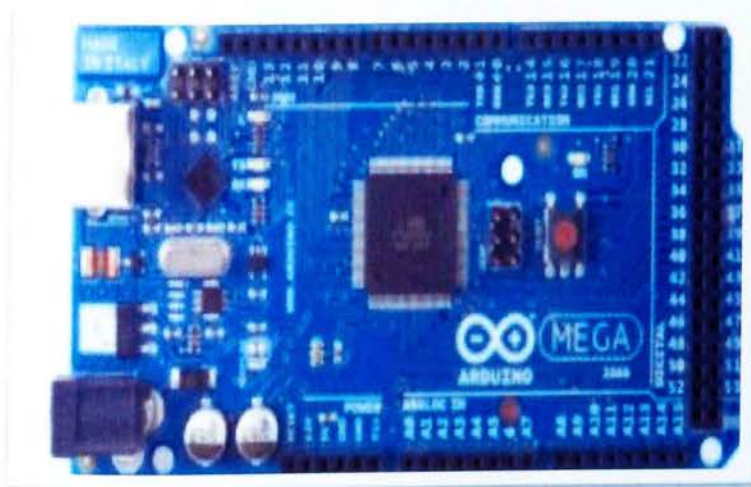
Τα κυκλώματα σχεδιάστηκαν με το πρόγραμμα Fritzing. Το πρόγραμμα είναι ανοικτού κώδικα και δίνει την δυνατότητα στους σχεδιαστές κυκλωμάτων την αναπαράσταση και τυποποίηση κυκλωμάτων σε breadboard, πλακέτα αλλά και σε σχέδιο. Ο σχεδιαστής μέσω των βιβλιοθηκών του προγράμματος μπορεί να αναζητήσει τυποποιημένα εξαρτήματα και ολοκληρωμένα στοιχεία που υπάρχουν στον εμπόριο και να τα χρησιμοποιήσει για την σχεδίαση του δικού του κυκλώματος. Επίσης ο χρήστης μπορεί να φτιάξει δικά του ηλεκτρονικά στοιχεία και να τα ανεβάσει στην ανοικτή κοινότητα του προγράμματος για διαμοιρασμό του στοιχείου με άλλους χρήστες. Τέλος ο χρήστης μπορεί να τυπώσει το κύκλωμα και να προχωρήσει στην παραγωγή της πλακέτας.

Ηλεκτρονικά - Ηλεκτρολογικά Εξαρτήματα

1. Arduino Mega 2560 R3

ο Arduino Mega 2560 έχει ένα ενσωματωμένο μικροελεγκτή με βάση την ATmega2560 Διαθέτει 54 ψηφιακά ακροδέκτες εισόδου / εξόδου (εκ των οποίων 14 μπορούν να

χρησιμοποιηθούν ως έξοδοι PWM), 16 αναλογικές εισόδους, 4 UARTs (hardware σειριακές θύρες), ένα 16 MHz ταλαντωτή κρυστάλλου, μια σύνδεση USB, μια υποδοχή ρεύματος, μια επικεφαλίδα ICSP, και ένα κουμπί επαναφοράς. Περιέχει όλα όσα χρειάζονται για να υποστηρίξουν την μικροελεγκτή. Απλά το



συνδέουμε σε έναν υπολογιστή με ένα καλώδιο USB ή με ένα AC-σε-DC adapter ή με μια μπαταρία για να ξεκινήσουμε. Το Mega είναι συμβατό με τις περισσότερες ασπίδες έχουν σχεδιαστεί για το Arduino Duemilanove ή Diecimila.

Τεχνικά χαρακτηριστικά

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz



2. NEMA-23 Bipolar 50mm Stepper

Ο NEMA-23 διπολικός βηματικός κινητήρας που διαθέτει υψηλή ανάλυση και υψηλή ροπή. Τον οποίο εμείς μαζί με μια μηχανική διάταξη χρησιμοποιήσαμε για την περιστροφή του αναβατορίου

Τεχνικά χαρακτηριστικά

- *Step Angle* 0.9°
- *Step Accuracy* 5%
- *Holding Torque* 9 kg-cm / 7.81 lb-in
- *Coil Resistance* 4 Ohms
- *Rated Current* 1.7A
- *Motor Length* 50mm
- *Mounting Plate Size* NEMA-23
- *Shaft Diameter* 6.35mm
- *Weight* 580g
- *Number of Leads* 4



3. NEMA-23 Bipolar 41mm Stepper

Ο NEMA-23 διπολικός βηματικός κινητήρας που διαθέτει υψηλή ανάλυση και υψηλή ροπή. Τον οποίο εμείς μαζί με μια μηχανική διάταξη χρησιμοποιήσαμε για την ανύψωση του αναβατορίου

Τεχνικά χαρακτηριστικά

- *Step Angle* 1.8°
- *Step Accuracy* 5%
- *Holding Torque* 4000 g-cm / 55.55 oz-in
- *Coil Resistance* 1.8 Ohms
- *Rated Current* 1.5A
- *Motor Length* 41mm
- *Mounting Plate Size* NEMA-23
- *Shaft Diameter* 6.35mm
- *Weight* 424g
- *Number of Leads* 4

4. Τροφοδοτικό

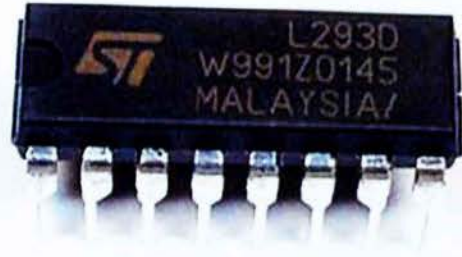
Χρησιμοποιήσαμε τροφοδοτικό 24 Volt DC 4 Ampere, για να τροφοδοτήσουμε όλα τα ηλεκτρονικά και ηλεκτρικά στοιχεία τους συστήματος.



5. Γέφυρες L293DNE

Για τον έλεγχο και την τροφοδοσία των βηματικών κινητήρων και του DC κινητήρα δημιουργήσαμε τρεις γέφυρες με 5 ολοκληρωμένα L293DNE.

Για του δύο βηματικούς κινητήρες χρησιμοποιήσαμε από ένα ζεύγος L293DNE για περισσότερη ισχύ. Για τον DC κινητήρα χρησιμοποιήσαμε ένα L293DNE.



Τεχνικά χαρακτηριστικά

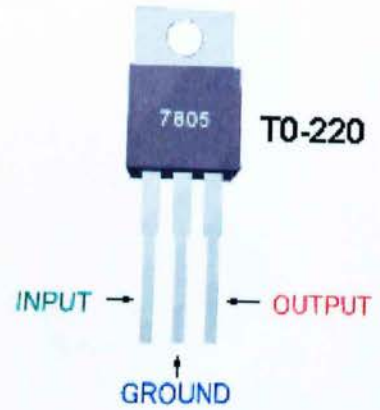
- *Featuring Unitrode L293 and L293D*
- *Products Now From Texas Instruments*
- *Wide Supply-Voltage Range: 4.5 V to 36 V*
- *Separate Input-Logic Supply*
- *Internal ESD Protection*
- *Thermal Shutdown*
- *High-Noise-Immunity Inputs*
- *Functionally Similar to SGS L293 and SGS L293D*
- *Output Current 1 A Per Channel (600 mA for L293D)*
- *Peak Output Current 2 A Per Channel (1.2 A for L293D)*
- *Output Clamp Diodes for Inductive*
- *Transient Suppression (L293D)*

6. 7805 Voltage Regulator

Με ένα 7805 δίνουμε ελεγχόμενη τροφοδοσία 5Volt στα L293D.

Τεχνικά χαρακτηριστικά

- *Output Current up to 1A*
- *Output Voltages of 5*
- *Thermal Overload Protection*
- *Short Circuit Protection*
- *Output Transistor Safe Operating Area Protection*



7. Οθόνη LCD

Τεχνικά χαρακτηριστικά

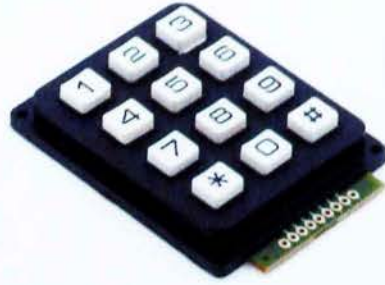
- *High quality STN 16x2 character LCD*
- *3.3V power supply*
- *Yellow LED Backlight*
- *5x8 dot characters*
- *ST7066 controller*
- *1/16 duty cycle*
- *Διαστάσεις: 0.34 x 1.4 x 3.2" (8.6 x 36 x 80mm)*



8. Πληκτρολόγιο

Τεχνικά χαρακτηριστικά

- Πληκτρολόγιο με 12 κουμπιά



9. DC Motor



10. Πλακέτα διάτρητη



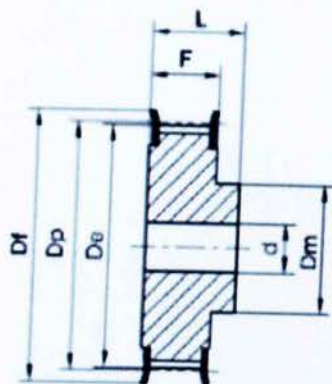
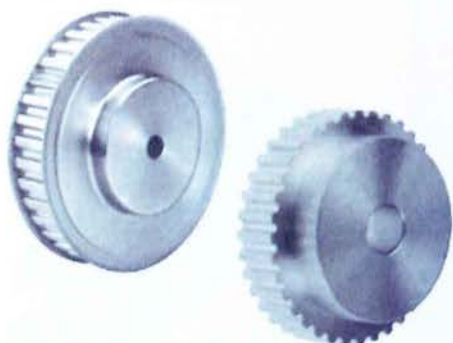
11. Αισθητήριο υπέρυθρων

- LED υπέρυθρης ακτινοβολίας (Πομπός)
- Φωτοτρανζίστορ (Δέκτης)

Κεφάλαιο 7 – Μηχανολογικά εξαρτήματα

1. Τροχαλία 6mm(μεγάλη)

Pulley XL-60

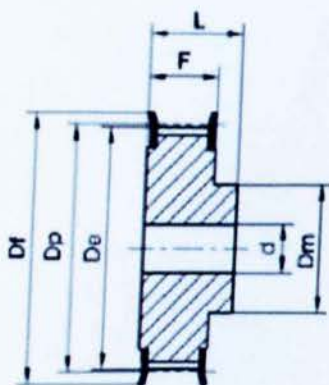
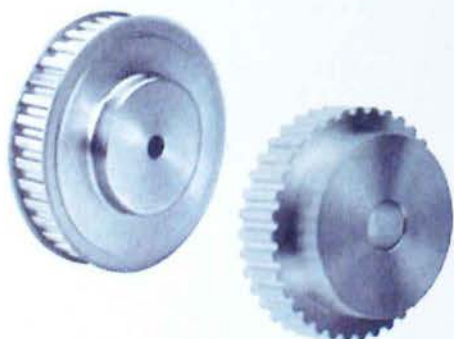


Type 1F

INFO	SIZE	TYPE	Z	Dp	De	Df	Dm	F	L	d	Flage No
ALUMINIUM											
WITHOUT FLANGF	60 XLO37	3	60	97.02	96.51	-----	45.0	14.3	25	10	-----

2. Τροχαλία 4mm(μικρή)

Pulley XL-40



Type 1F

INFO	SIZE	TYPE	Z	Dp	De	Df	Dm	F	L	d	Flage No
ALUMINIUM											
WITHOUT FLANGF	40 XLO37	2	40	64.68	64.17	-----	45.0	14.3	25	8	-----

3. 4 X Προφίλ αλουμινίου

Aluminium Industrial Profile 20x20



20X20 Profile

It is mostly used in the production of cabs of machine and its devices, displays, stands and billboard types.

Protective Cab for Plastic head mounting machine made of 20x20 profile

4. 2 Ρουλεμάν UCF 8mm



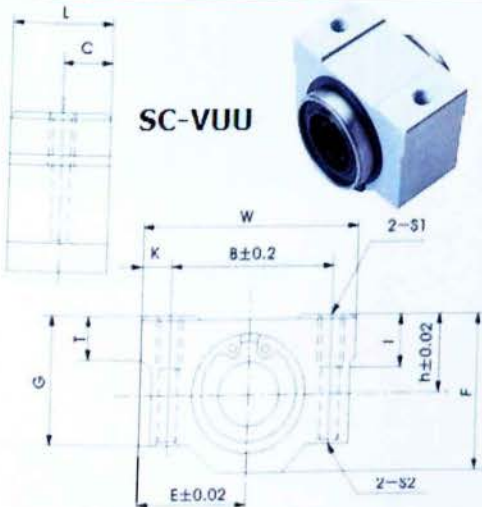
5. 2 Χ Φύλλα αλουμινίου

Solid Aluminum flat bar



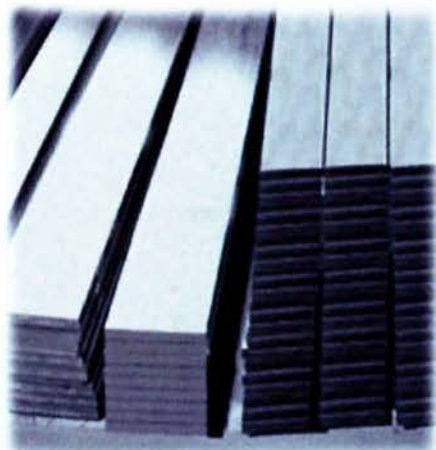
6. 2 Χ γραμμικά ρουλεμάν

SC12VUU - Linear Motion Slide Shaft Bearing D12mm Close



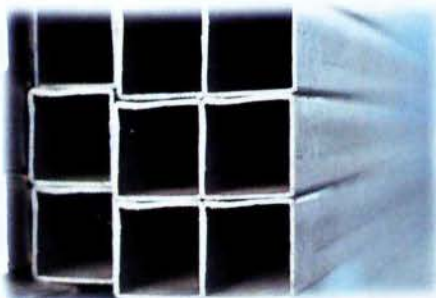
MODEL NO.	SHAFT DIAMETER	MAIN DIMENSIONS							MOUNTING DIMENSIONS					WEIGHT(kg)	
		h	E	W	L	F	G	T	B	C	K	S1	S2		I
SC12VUU	12	15	21	42	21	28	24	8	30.5	10.5	5.75	M5	4.3	12	0.06

7. Λαμες 3cm x 2mm



8. Σίδερα τετραγωνα 4cm x 4cm

Κοιλοδοκοί τετράγωνοι



9. 2 x Στρογγυλες ράβδοι

Steel round shaft C 45 DM 12mm



10. Ξύλο MDF

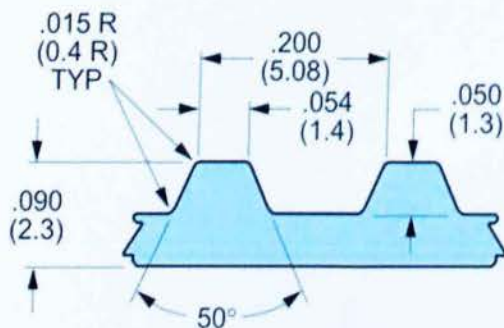


11. Ιμάντας χρονισμού

100XL Timing Belt 50 teeth 254mm



1/5 Pitch XL
(5.08 Pitch)



Passo - Pitch 1/5" = 5.08 mm

TIPO TYPE	N° Denti TEETH	SVILUPPO Length mm
60 XL	30	152.40
70 XL	35	177.80
80 XL	40	203.20
90 XL	45	228.60
100 XL	50	<u>254.00</u>

12. Πύρος



13. Συρματόσκοινο 3mm



14. Γωνίες στήριξης



15. Πλαστικό PVC(άσπρο)



16. Couplings



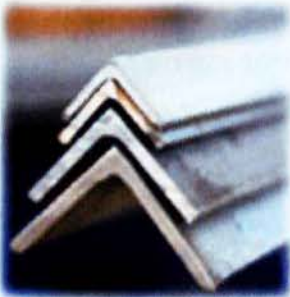
Shaft : 6.35mm X 5mm

Length: 25.5mm

Outside diameter: 20mm

Material: Aluminium

17. Γωνίες 3cm (λάμες)



Πρόταση 1- Πηγές ανανεώσιμης ενέργειας

Τα αυτόματα παρκινγκ εκτός από την πρακτικότητα και την ασφάλεια που παρέχουν στους καταναλωτές, καλό θα ήταν να γίνουν και πιο φιλικά προς το περιβάλλον. Για το λόγο αυτό και εμείς προτείνουμε την πλήρη τροφοδότηση των εγκαταστάσεων αυτών αποκλειστικά με ρεύμα που παράγεται από πηγές ανανεώσιμης ενέργειας.

Για την επίτευξη αυτού του στόχου θα μπορούσαν να χρησιμοποιηθούν φωτοβολταϊκά πάνελ ή και ακόμα ανεμογεννήτριες στα μέρη που έχουν υψηλούς ανέμους .



Οι πηγές ανανεώσιμης ενέργειας εκτός από φιλικές προς το περιβάλλον θα μειώσουν τα έξοδα του χώρου στάθμευσης διότι όχι μόνο θα τροφοδοτούν τις εγκαταστάσεις με ρεύμα αλλά δεν θα χρειάζονται εφεδρικές γεννήτριες για την περίπτωση πτώσης του δικτύου, με αποτέλεσμα να μειωθεί το αντίστοιχο αντίτιμο για την φύλαξη του εκάστοτε οχήματος . Έτσι θα γίνουν πιο προσιτά σε μεγαλύτερο αγοραστικό κοινό, θα αυξήσουν τα έσοδα τους και επίσης θα ακολουθήσει η ανέγερση και άλλων τέτοιων κτηρίων μιας και η ζήτηση πλέον θα είναι μεγαλύτερη της προσφοράς.

Τέλος ένας τρόπος οι χώροι αυτοί να γίνουν περισσότερο φιλικό προς το περιβάλλον θα ήταν να χρησιμοποιούν ανακυκλώσιμα υλικά για την κατασκευή τους. Καθώς και ανακυκλωμένα αναλώσιμα είδη όπως το χαρτί στις περιπτώσεις που χρειάζονται να εκτυπωθούν διάφορες ενδείξεις προς τους χρήστες.

Πρόταση 2- Online παρακολούθηση

Η ασφάλεια είναι ένας από τους κύριους λόγους που χρησιμοποιούμε αυτόματους χώρους στάθμευσης. Πολλοί από τους καταναλωτές ανησυχούν για την κατάσταση του οχήματος τους όσο εκείνοι δεν βρίσκονται εκεί.

Για τον λόγο αυτόν θα μπορούσαν να διαμορφωθούν ειδικοί ιστότοποι στους οποίους θα συνδέεται ο χρήστης και έπειτα από την εισαγωγή ενός κωδικού ο οποίος θα έχει δοθεί από το παρκινγκ και θα αντιστοιχεί στο όχημά του να παρακολουθεί online οπουδήποτε αν βρίσκεται την κατάσταση του οχήματος του. Βεβαίως αυτό απαιτεί και την εγκατάσταση ενός κλειστού κυκλώματος από κάμερες το οποίο θα αυξήσει το κόστος κατασκευής. Παρόλα αυτά θα καλύπτει την ανάγκη αυτή του καταναλωτή .

Πρόταση 3- Εντολές από Smartphone

Η τεχνολογία έχει προχωρήσει με ραγδαία βήματα στον τομέα της τηλεπικοινωνίας. Πλέον τα περισσότερα κινητά τηλέφωνα δεν θυμίζουν σε τίποτα τα παλαιά αντίθετα θα μπορούσαμε να πούμε ότι έχουν γίνει μικροί υπολογιστές.

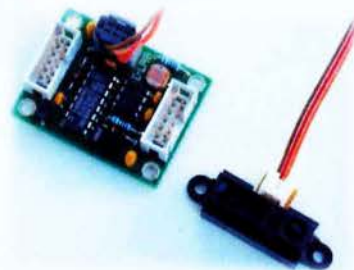
Η χρήση των smartphones είναι διαδεδομένη για τον λόγο αυτό θα μπορούσε να αναπτυχτεί μια εφαρμογή στην οποία θα συνδέετε ο χρήστης και αφού πρώτα έχει καταχωρηθεί ως πελάτης και χρήστης του χώρου αυτού στις βάσεις δεδομένων του κεντρικού υπολογιστή, να μπορεί μέσα από την εφαρμογή δηλαδή από το κινητό του τηλέφωνο να δίνει εντολές στο σύστημα όσον αφορά την διαδικασία προς στάθμευση αλλά και από στάθμευση. Έτσι θα γλυτώνουμε λειτουργικό χρόνο αλλά και θα προκαλούμε τις εντυπώσεις του καταναλωτή.

Εφικτή είναι και η δημιουργία μιας εφαρμογής online κρατήσεων αλλά και επιλογής συγκεκριμένης θέσης στο χώρο στάθμευσης .



- **Τοποθέτηση αισθητήρων**

Για την καλύτερη ασφάλεια και έλεγχο αν βρίσκονται αμάξι στην κάθε θέση του κτηρίου μας θα μπορούσαμε να τοποθετήσουμε σε κάθε θέση ένα αισθητήριο το οποίο θα μας ενημερώνει κατάλληλα αν στην συγκεκριμένη θέση βρίσκεται κάποιο αμάξι.



- **Τοποθέτηση καμερών**

Επίσης σε κάθε θέση θα μπορούσαμε να βάλουμε και κάμερα για καλύτερη παρακολούθηση του αμαξιού.



- **Λαμπάκια led**

Για περισσότερη καλαισθησία του κτηρίου μια λύση θα ήταν να βάλουμε κάποια led σε διάφορα σημεία που θα φωτίζουν το κτήριο μας αναλόγως την ενέργεια που εκτελείται (παρκάρισμα – ξεμπαρκάρισμα) .



- **Barcode**

Μια ακόμα βελτίωση που θα μπορούσε να είχε η κατασκευή μας θα ήταν να δίνουμε barcode σε ειδική κάρτα σε κάθε όχημα έτσι ώστε στην αποτοποθέτηση ο κάτοχος του αμαξιματος θα έδειχνε το barcode στο μηχάνημα και αυτό θα ξεκινούσε την διαδικασία αποτοποθέτησης.



- **Μηχανισμός**

Ακόμα θα μπορούσαμε να βελτιώσουμε τον μηχανισμό που κάνει την τοποθέτηση και την αποτοποθέτηση του αμαξώματος. Με ποιο σωστά υλικά και με έναν πιο δυνατό dc κινητήρα θα μπορούσε να γίνει η συγκεκριμένη κίνηση με πιο μεγάλη ασφάλεια και με πιο μεγάλη ακρίβεια.



- **Android πρόγραμμα**

Εξαιτίας της ραγδαίας ανάπτυξης στα κινητά τηλέφωνα θα ήταν χρήσιμη η δημιουργία ενός προγράμματος το οποίο θα μπορεί να ελέγχει και να παρακολουθεί την τοποθέτηση και αποτοποθέτηση των αμαξιών. Επίσης θα μας πληροφορεί σε ποιες θέσεις υπάρχουν οχήματα και ποιες είναι άδειες



Κεφάλαιο 10 - Interfacing

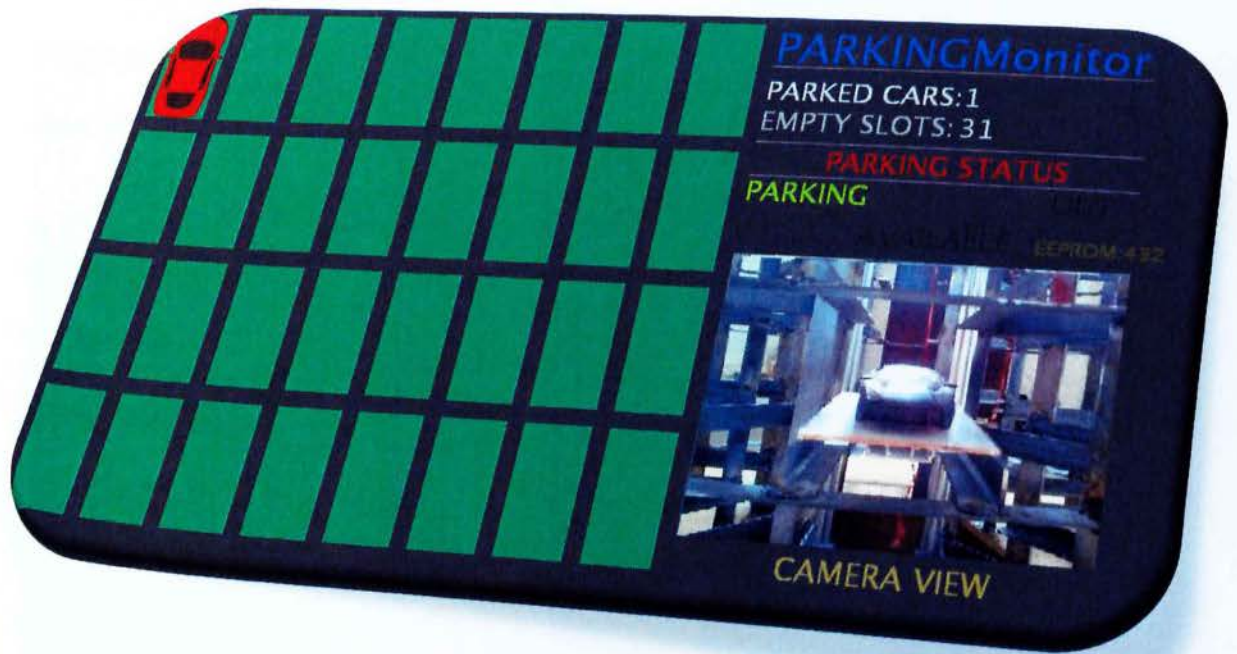
Περιγραφή

Για σκοπούς εποπτείας του συστήματος δημιουργήσαμε μια εφαρμογή όπου μας αναπαριστά χρήσιμες πληροφορίες για το σύστημα μας. Το πρόγραμμα αυτό δημιουργήθηκε με την γλώσσα Processing και έχει τις παρακάτω δυνατότητες:

1. Μετρητή κενών και γεμάτων θέσεων (Empty Slots & Parked Cars)
2. Γραφική αναπαράσταση κενών και γεμάτων θέσεων
3. Κάμερα η οποία μας φέρνει σε οπτική επαφή με το σύστημα
4. Κατάσταση του συστήματος
 - Parking
 - Out
 - Available
5. 5.Μετρητής EEPROM

Το Processing είναι μια γλώσσα προγραμματισμού ανοικτού κώδικα και παράλληλα ένα προγραμματιστικό περιβάλλον για ανθρώπους που θέλουν να προγραμματίσουν εικόνες, animation και ήχο. Όλα ξεκίνησαν το 2001 όταν δύο απόφοιτοι του πανεπιστημίου MIT, Benjamin Fry και Casey Reas ξεκίνησαν την ανάπτυξη της γλώσσας Processing πάνω σε Java. Παρόλο που η γλώσσα αναπτύχθηκε στη Java, το συντακτικό της είναι απλουστευμένο και το προγραμματιστικό της μοντέλο βασίζεται στα γραφικά. Απώτερος σκοπός των δύο δημιουργών είναι η εκμάθηση προγραμματισμού από αρχάριους χρήστες μέσω ενός οπτικού πλαισίου καθώς και η παροχή ενός επαγγελματικού εργαλείου παραγωγής πολυμεσικών εφαρμογών. Το περιβάλλον του Processing είναι γραμμένο σε Java. Προγράμματα που έχουν γραφτεί σε Processing είναι επίσης μεταφρασμένα σε Java και έπειτα τρέχουν σαν Java προγράμματα. Προγράμματα που είναι γραμμένα σε Java και Processing, συνήθως τρέχουν γρηγορότερα από προγράμματα που βασίζονται σε scripting γλώσσες όπως ActionScript και Lingo, κάτι το οποίο είναι πολύ σημαντικό για εφαρμογές με γραφικά. Μεγάλες διακρίσεις μεταξύ του Processing και της Java είναι οι βιβλιοθήκες γραφικών του Processing και το απλοποιημένο στυλ προγραμματισμού που δεν προϋποθέτει οι χρήστες να καταλαβαίνουν πιο προηγμένες έννοιες όπως κλάσεις, αντικείμενα, ή animation

(ενώ εξακολουθούν να είναι διαθέσιμες για προχωρημένους χρήστες). Τέτοιες τεχνικές λεπτομέρειες πρέπει να είναι ειδικά προγραμματισμένες σε Java, αλλά ολοκληρωμένες σε Processing, κάνοντας τα προγράμματα μικρότερα και ευκολότερα να διαβαστούν.



Στη φωτογραφία φαίνεται ότι στο σύστημα υπάρχουν 31 κενές θέσεις, 1 αμάξι, το σύστημα βρίσκεται σε κατάσταση παρκαρίσματος και ο EEPROM counter είναι 432. Το πρόγραμμα επικοινωνεί με το Arduino μέσω της σειριακής θύρας. Για να αναπαραστήσει γραφικά το πρόγραμμα τις κενές και γεμάτες θέσεις, το arduino στέλνει μέσω της σειριακής τα δεδομένα που βρίσκονται αποθηκευμένα στην μνήμη EEPROM. Το πρόγραμμα επεξεργάζεται αυτή την πληροφορία για να ενημερώσει τις μεταβλητές free, cars και EEPROM αλλά και για να αναπαραστήσει γραφικά τις κενές και γεμάτες θέσεις.

Κώδικας

```
import processing.video.*;
```

```
import processing.serial.*;
```

Βιβλιοθήκες κάμερας και σειριακής
επικοινωνίας

```
String val="";
```

```
Serial myPort;
```

```
Capture cam;
```

```
PImage img;
```

```
PImage img2;
```

```
int width1=320;
```

```
int height1=240;
```

```
void setup() {
```

Στην συνάρτηση *setup* ρυθμίζουμε
την λειτουργία της σειριακής θύρας
και κάμερας

```
background(51);
```

```
size(1280, 720);
```

```
String portName = Serial.list()[0];
```

```
myPort = new Serial(this, "COM13", 9600);
```

```
myPort.bufferUntil('\n');
```

```
String[] cameras = Capture.list();
```

```
if (cameras.length == 0) {
    println("There are no cameras available for capture.");
    exit();
}
else {
    println("Available cameras:");
    for (int i = 0; i < cameras.length; i++) {
        println(cameras[i]);
        println(i);
    }
    cam = new Capture(this, cameras[0]);
    cam.start();
}
if (frame != null) {
    frame.setResizable(true);
}

img = loadImage("green.png");
img2 = loadImage("greencar.png");

}
```

Η συνάρτηση `draw` είναι η κύρια συνάρτηση μας όπου γίνεται και ενημέρωση των ελεγχόμενων μεταβλητών και σχεδίαση των σχηματικών διατάξεων

```
void draw() {  
  
background(51);  
  
String a;  
String b;  
  
String list[] = split(val, ',');  
a =list[0];  
b =list[1];  
  
for (int j=0; j<4; j++)  
for (int i=0; i<8; i++)  
image(img,20+95*i, 20+170*j);  
  
if (cam.available() == true) {  
    cam.read();  
    image(cam, 780, 320,420,320);  
  
    textSize(52);  
    fill(0, 102, 153);  
    text("PARKINGMonitor", 777, 70);  
}
```



```
textSize(32);  
fill(193);  
text("PARKED CARS:", 777, 120);
```

```
textSize(32);  
fill(142,35,35);  
text("PARKING STATUS", 860, 210);
```

```
textSize(32);  
fill(153);  
text("EMPTY SLOTS:", 777, 160);
```

```
stroke(153);  
line(777, 175, 1200, 175);
```

```
stroke(153);  
line(777, 218, 1200, 218);
```

```
stroke(153);  
line(777, 75, 1200, 75);
```

```
if (int(list[35])==2)  
{  
textSize(32);  
fill(0,255,0);
```

```
text("PARKING", 777, 250);  
}  
else  
{  
textSize(32);  
fill(41,41,41);  
text("PARKING", 777, 250);  
}  
  
if (int(list[35])==3)  
{  
textSize(32);  
fill(0,255,0);  
text("OUT", 1110, 250);  
}  
else  
{  
textSize(32);  
fill(41,41,41);  
text("OUT", 1110, 250);  
}
```

```
if (int(list[35])==1)
{
  textSize(32);
  fill(0,255,0);
  text("AVAILABLE", 910, 300);
}
else
{
  textSize(32);
  fill(41,41,41);
  text("AVAILABLE", 910, 300);
}

textSize(22);
fill(79,79,47);
text("EEPROM:", 1100, 300);

textSize(22);
fill(79,79,47);
text(list[34], 1200, 300);

textSize(32);
fill(139,134,78);
text("CAMERA VIEW", 886, 680);
```

```
textSize(32);
```

```
fill(193);
```

```
text(a, 1000, 120);
```

```
textSize(32);
```

```
fill(193);
```

```
text(b, 1000, 160);
```

```
int c=0;
```

```
for (int i=0; i<=31; i++){
```

```
    c=i+2;
```

```
    if (int(list[c])==1)
```

```
    {
```

```
        int r;
```

```
        int m;
```

```
        r=i/8;
```

```
        m=i%8;
```

```
        image(img2,20+95*m, 20+170*r);
```

```
    }
```

```
}
```

```
}
```

```
void serialEvent(Serial p) {  
    val = (myPort.readString());  
}
```

```
boolean sketchFullScreen()  
{  
    return true;  
}
```

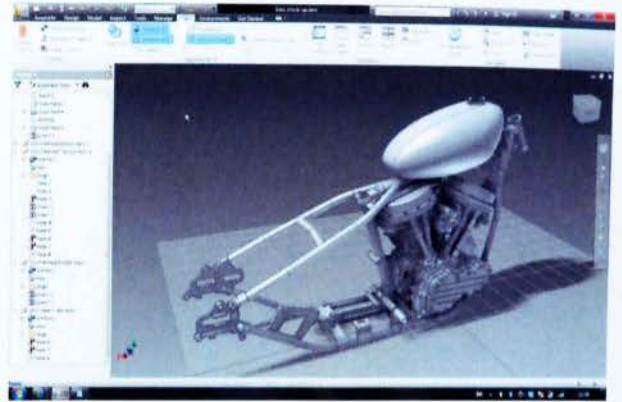
*Η συνάρτηση SerialEvent ρυθμίζει
την σειριακή επικοινωνία*

Κεφάλαιο 11 – Το Λογισμικό που χρησιμοποιήσαμε

- Για το σχεδιασμό του αναβατορίου χρησιμοποιήσαμε το Autodesk Inventor



Λίγα λόγια : Autodesk Inventor είναι ένα 3D λογισμικό σχεδίασης μηχανικών στερεών μοντέλων που αναπτύχθηκε από την Autodesk για τη δημιουργία 3D ψηφιακών πρωτοτύπων. Επιτρέπει στους χρήστες να παράγουν ακριβή 3D μοντέλα που βοηθούν στο σχεδιασμό, και την προσομοίωση τους πριν κατασκευαστούν. Περιλαμβάνει ενσωματωμένη προσομοίωση κίνησης και συναρμολόγησης, ανάλυση τάσεων,



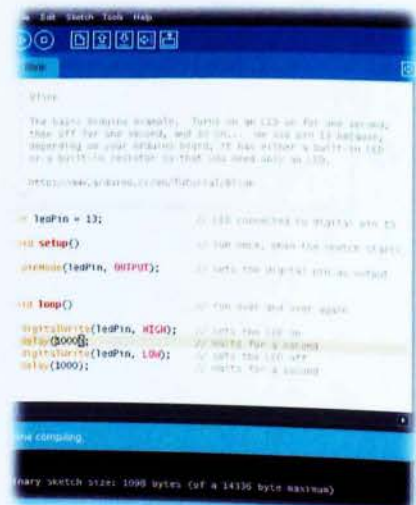
σύμφωνα με την οποία οι χρήστες έχουν επιλογές να εισάγουν φορτία οδήγησης εισόδου, δυναμικά στοιχεία, φορτία τριβής για να ελέγξετε πώς το προϊόν θα λειτουργήσει σε ένα πραγματικό σενάριο. Αυτά τα εργαλεία επιτρέπουν στους χρήστες προσομοίωση στο σχεδιασμό αυτοκινήτων ή εξαρτημάτων αυτοκινήτων,

για παράδειγμα, μπορούν βελτιστοποιήσουν τη δύναμη, το βάρος του προϊόντος, και να εντοπίσουν τις περιοχές υψηλής πίεσης. Βοηθά μέχρι και στον τον εντοπισμό-μείωση των ανεπιθύμητων δονήσεων.

- Το πρόγραμμα γράφτηκε στον Arduino IDE με Wiring Language

Λίγα λόγια : Το IDE του Arduino είναι γραμμένο σε Java και μπορεί να τρέξει σε πολλαπλές πλατφόρμες.

Περιλαμβάνει επεξεργαστή κώδικα (επεξεργαστή κειμένου με διάφορα εύχρηστα εργαλεία) και μεταγλωττιστής και έχει την ικανότητα να φορτώνει εύκολα το πρόγραμμα μέσω σειριακής θύρας από τον υπολογιστή στην πλακέτα.



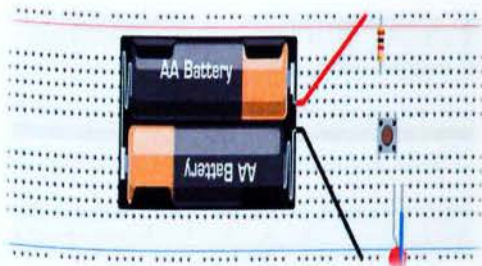
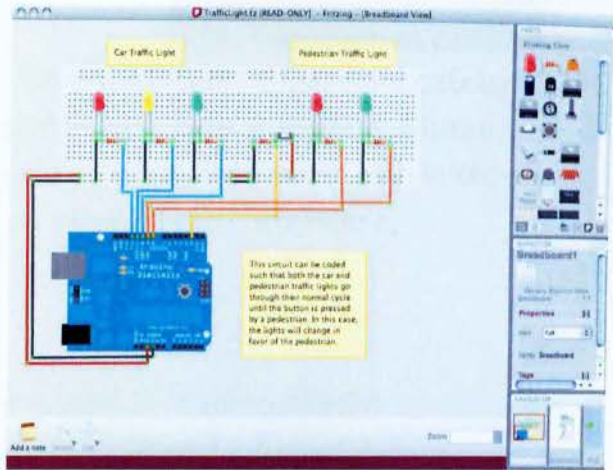
Το περιβάλλον ανάπτυξης είναι βασισμένο στην Processing, ένα περιβάλλον ανάπτυξης σχεδιασμένο να εισαγάγει στον προγραμματισμό μη εξοικειωμένους χρήστες με την ανάπτυξη λογισμικού. Η συγκεκριμένη γλώσσα προγραμματισμού προέρχεται από την Wiring, μια γλώσσα που μοιάζει με την C η οποία παρέχει παρόμοια λειτουργικότητα για μια πιο περιορισμένης σχεδίασης πλακέτα, της οποίας το περιβάλλον ανάπτυξης βασίζεται επίσης στην Processing.

- Τα κυκλώματα σχεδιάστηκαν με το πρόγραμμα Fritzing

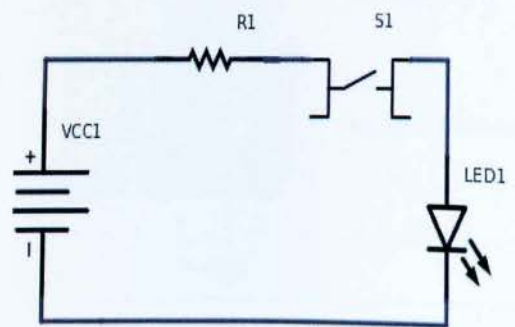


Λίγα λόγια : Fritzing είναι μια πρωτοβουλία ανοικτού λογισμικού για την υποστήριξη σχεδιαστών, ώστε να είναι έτοιμοι να προχωρήσουν από τη φυσική

προτυποποίηση ενός κυκλώματος σε πραγματικό προϊόν. Αναπτύχθηκε στο Πανεπιστήμιο Εφαρμοσμένων Επιστημών του Πότσταμ. Το Fritzing μπορεί να θεωρηθεί ως ένας ηλεκτρονικός αυτοματισμός σχεδιασμού (EDA) εργαλείο για μη-μηχανικούς: Ουσιαστικά είναι σαν να σχεδιάζουμε ένα κύκλωμα στο breadboard και το πρόγραμμα να μας τυπώνει το σχέδιο του



κυκλώματος.

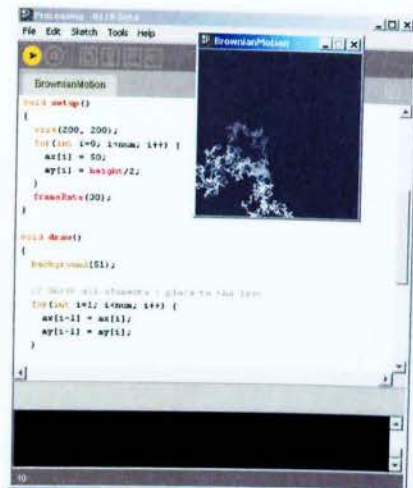


- Το interface φτιάχτηκε με το Processing

Λίγα λόγια : Το Processing είναι μια ανοικτή γλώσσα προγραμματισμού πηγή και με περιβάλλον για τους ανθρώπους που θέλουν να δημιουργήσουν εικόνες, κινούμενα σχέδια,



και τις αλληλεπιδράσεις τους. Αρχικά αναπτύχθηκε για να χρησιμεύσει ως ένα sketchbook λογισμικού και να διδάξει βασικές αρχές του προγραμματισμού

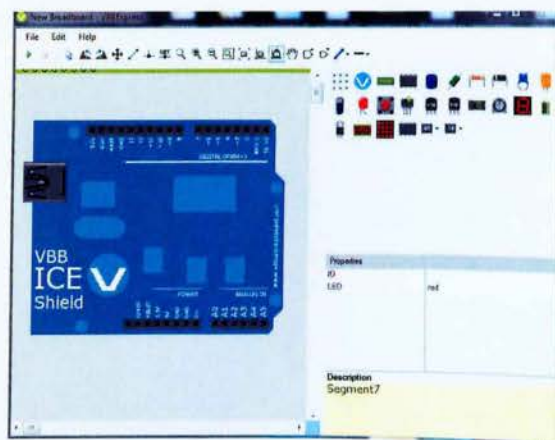


ηλεκτρονικών υπολογιστών μέσα σε ένα οπτικό πλαίσιο, έχει επίσης εξελιχθεί σε ένα εργαλείο για τη δημιουργία μίας επαγγελματικής δουλειά. Σήμερα, υπάρχουν δεκάδες χιλιάδες φοιτητές, καλλιτέχνες, σχεδιαστές, ερευνητές και χομπίστες που το χρησιμοποιούν για την εκμάθηση επεξεργασία, προτυποποίηση και την παραγωγή.

- Τα κυκλώματα και ο κώδικας εξομοιώθηκαν με το Virtual Breadboard



Λίγα λόγια : Το Virtual Breadboard είναι ένα λογισμικό που δίνει την δυνατότητα στο χρήστη να σχεδιάσει το κύκλωμα



του σε περιβάλλον bread board και να εξομοιώσει την λειτουργία του βάση του κώδικα που έχει γράψει στους μικροελεγκτές του κυκλώματος. Με την εξομοίωση δίνει την πραγματική λειτουργία του κυκλώματος σε οπτική απεικόνιση και λειτουργία. Αυτό δίνει την δυνατότητα στο χρήστη να κάνει βελτιώσεις στο κύκλωμα και στον κώδικα χωρίς να χρειάζεται να έχει πραγματικά εξαρτήματα για να το κάνει. Αυτό δίνει την ευχέρεια στον χρήστη να αναπροσαρμόζει το κύκλωμα πιο εύκολα και πιο γρήγορα με το εύκολο περιβάλλον σχεδίασης και εξομοίωσης. Το πρόγραμμα παρέχει μεγάλη γκάμα εξαρτημάτων αλλά και βιβλιοθηκών προγραμματισμού.

Βιβλιογραφία

Εισαγωγή	3
<ul style="list-style-type: none">• http://www.roboticparking.com/robotic_parking_projects_hoboken.htm• http://en.wikipedia.org/wiki/Autostadt• http://www.roboticparking.com/robotic_parking_projects_ibn_battuta.htm	
<ul style="list-style-type: none">• Ιστορική ανάδρομη	5
<ul style="list-style-type: none">• http://www.autoblog.gr/2010/12/24/the-secret-of-the-classiest-parking-lot-of-all-time/	
<ul style="list-style-type: none">• Ρομποτικά παρκινγκ στην Ελλάδα	8
<ul style="list-style-type: none">• http://el.wikipedia.org/wiki/%CE%91%CF%85%CF%84%CE%BF%CE%BA%CE%AF%CE%BD%CE%B7%CF%84%CE%BF• http://www.newsbeast.gr/car/arthro/184785/sughrono-robotiko-parking-ston-agio-padeleimona/	
Κεφάλαιο 1 – Το κτήριο	10
<ul style="list-style-type: none">• Η κύριος δομή του κτηρίου	10
<ul style="list-style-type: none">• Κατασκευή	11

- Η κίνηση του κτηρίου 12

Κεφάλαιο 2 – Μηχανισμός κίνησης (αναβατόριο) 13

- <http://usa.autodesk.com/autodesk-inventor/>

- Η δομή του αναβατορίου 13

- <http://www.cnccat.com/>

- Η μετάδοση κίνησης 15

- http://www.youtube.com/watch?v=8_iJ-XGpFvs

Κυκλική κίνηση: 15

- <http://www.youtube.com/watch?v=sRIBZF1xGIw>

Κάθετη κίνηση: 16

- <http://www.youtube.com/watch?v=zzBjbDathWQ>

Οριζόντια κίνηση: 17

- <http://www.youtube.com/watch?v=Yr8d61jDLJs>

Κεφάλαιο 3 – Συνδέσεις κυκλώματα 19

- <http://www.instructables.com/id/How-to-solder/step4/Soldering-components-onto-a-circuit-board/>

- Τροφοδοσία ηλεκτρικών – ηλεκτρονικών στοιχείων..... 19

- <http://jumptuck.com/2011/11/15/voltage-regulation-friend-7805/>

- Οθόνη – Πληκτρολόγιο 20

- <http://playground.arduino.cc/Main/KeypadTutorial>

- <http://arduino.cc/en/Tutorial/LiquidCrystal>

- Οδηγοί βηματικών κινητήρων (stepper drivers)..... 21

- <http://www.seattlerobotics.org/encoder/may98/steppers.html>

- Οδηγός DC κινητήρα 22

- www.cambierobotics.com/HTML/Robotics/L293%20Motor%20Driver%201.pdf

Αισθητήριο υπερύθρων..... 23

- http://www.pages.drexel.edu/~pyc23/ir_sensor.html

Κεφάλαιο 4 – Η «Νοημοσύνη» του κτηρίου (Arduino). 24

- <http://arduino.cc/en/Reference/EEPROM>
- <http://arduino.cc/en/Tutorial/EEPROMWrite>
 - Η σχετική νοημοσύνη 24
- <http://arduino.cc/en/Reference/Stepper?from=Tutorial.Stepper>
- <http://arduino.cc/en/Reference/serial>

Παραλαβή-Τοποθέτηση από τη είσοδο: 24

- <http://arduino.cc/en/Tutorial/MotorKnob>

Παραλαβή-Τοποθέτηση στην έξοδο: 24

- <http://arduino.cc/blog/category/actuators/motors/>
 - Προγραμματισμός-Επεξήγηση κώδικα (Arduino) 25
- <http://arduino.cc/en/Reference/HomePage>

Κεφάλαιο 5 – Οδηγίες χρήσης 46

- Κεντρική περίπτωση 46

- Περιπτώσεις προς στάθμευση46
- Περίπτωση αποστάθμευσης47

Κεφάλαιο 6 – Ολικό κύκλωμα και εξαρτήματα48

- Ολικό κύκλωμα48
- Ηλεκτρονικά - Ηλεκτρολογικά Εξαρτήματα49

1. Arduino Mega 2560 R3

- <http://arduino.cc/en/Main/arduinoBoardMega2560>

2. NEMA-23 Bipolar 50mm Stepper

- http://www.phidgets.com/products.php?product_id=3308

3. NEMA-23 Bipolar 41mm Stepper

- http://www.phidgets.com/products.php?product_id=3306_0

4. Τροφοδοτικό

- <http://www.ebw.gr/el/product/50561/%CE%A4%CE%A1%CE%9F%CE%A6%CE%9F%CE%94%CE%9F%CE%A4%CE%99%CE%9A%CE%9F-%CE%A0%CE%91%CE%9A-%CE%A0%CE%9F%CE%9B-%CE%95%CE%9E%CE%9F%CE%94%CE%A9%CE%9D-3500mA-MW7H350-MWA>

5. Γέφυρες L293DNE

- <http://www.datasheetcatalog.org/datasheet/texasinstruments/l293d.pdf>

6. 7805 Voltage Regulator

- <http://www.fairchildsemi.com/ds/LM/LM7805.pdf>

7. Οθόνη LCD

- <http://www.sparkfun.com/datasheets/LCD/ADM1602K-FSY-FBS-3.3V.pdf>

8. Πληκτρολόγιο

- http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Components/General/SparkfunCOM-08653_Datasheet.pdf

9. DC Motor

- <http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Robotics/KM-12FN20-100-06120.pdf>

10. Πλακέτα διάτρητη

- <http://www.acdcshop.gr/content/pc-1.pdf>

11. Αισθητήριο υπερύθρων

- http://www.everlight.com/datasheets/IR333_H0_L10_datasheet.pdf

Κεφάλαιο 7 – Μηχανολογικά εξαρτήματα.....56

1. Τροχαλία 6mm(μεγάλη)

- <http://www.cnccat.com/index.php?id=1&catid=34&lang=gr>

2. Τροχαλία 4mm(μικρή)

- <http://www.cnccat.com/index.php?id=1&catid=34&lang=gr>

3. 4 X Προφίλ αλουμινίου

- <http://www.cnccat.com/index.php?id=1&catid=57&lang=gr>

4. 2 Ρουλεμάν UCF 8mm

- http://www.alexandris.com/gr/product_pr.php?Id=143

5. 2 X Φύλλα αλουμινίου

- <http://www.cnccat.com/index.php?id=1&catid=88&lang=gr>

6. 2 X γραμμικά ρουλεμάν

- <http://www.cnccat.com/index.php?id=1&catid=105&lang=gr>

7. Λαμες 3cm x 2mm

8. Σίδερα τετραγωνα 4cm x 4cm

9. 2 x Στρογγυλες ράβδοι

- <http://www.cncat.com/index.php?id=1&catid=102&lang=gr>

10. Ξύλο MDF

11. Ιμάντας χρονισμού

- <http://www.cncat.com/index.php?id=1&catid=33&lang=gr>

12. Πύρος

13. Συρματόσκοινο 3mm

14. Γωνίες στήριξης

15. Πλαστικό PVC(άσπρο)

16. Couplings

- <http://www.cncat.com/index.php?id=1&lang=gr&catid=28>

17. Γωνίες 3cm (λάμες)

Κεφάλαιο 8 – Μελλοντικές βελτιώσεις..... 63

- Πρόταση 1- Πηγές ανανεώσιμης ενέργειας 63

- http://el.wikipedia.org/wiki/%CE%91%CE%BD%CE%B1%CE%BD%CE%B5%CF%8E%CF%83%CE%B9%CE%BC%CE%B5%CF%82_%CF%80%CE%B7%CE%B3%CE%AD%CF%82_%CE%B5%CE%BD%CE%AD%CF%81%CE%B3%CE%B5%CE%B9%CE%B1%CF%82

- Πρόταση 2- Online παρακολούθηση 64

- http://en.wikipedia.org/wiki/Remote_monitoring_and_control

- Πρόταση 3- Εντολές από Smartphone 64

- http://en.wikipedia.org/wiki/Mobile_application_development

Κεφάλαιο 9 - Μελλοντικές βελτιώσεις της κατασκευής μας 65

Κεφάλαιο 10 – Interfacing 67

- Περιγραφή 67

- <http://processing.org/>

- Κώδικας..... 69

- <http://processing.org/reference/>

Κεφάλαιο 11 – Το λογισμικό που χρησιμοποιήσαμε 77

1. Autodesk Inventor

- http://en.wikipedia.org/wiki/Autodesk_Inventor

2. Arduino IDE

- <http://el.wikipedia.org/wiki/Arduino>

3. Fritzing

- <http://en.wikipedia.org/wiki/Fritzing>

4. Processing

- [http://en.wikipedia.org/wiki/Processing_\(programming_language\)](http://en.wikipedia.org/wiki/Processing_(programming_language))

5. Virtual Breadboard

- <http://www.virtualbreadboard.com/>

