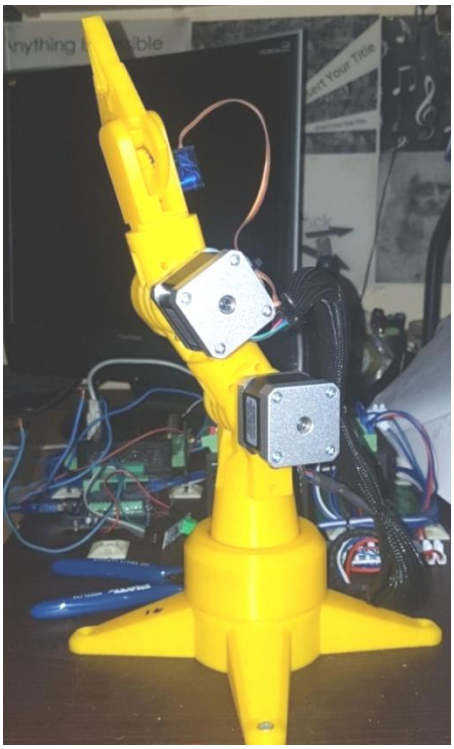
Διεθνές Πανεπιστήμιο της Ελλάδος
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Διπλωματική Εργασία Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική

Μελέτη και κατασκευή ρομποτικού βραχίονα 3 βαθμών ελευθερίας και ανάπτυξη συστήματος ελέγχου με χρήση PLC και GUI



Εργασία που υποβλήθηκε στο
Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική,
του Διεθνούς Πανεπιστημίου της Ελλάδος,
για τη μερική εκπλήρωση υποχρεώσεων
για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Εκπονητής: Ηλιάδης Γεώργιος

Επιβλέπων Καθηγητής: Δημήτριος Σαγρής

Σέρρες, 16-6-2020

Διεθνές Πανεπιστήμιο της Ελλάδος
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Διπλωματική Εργασία Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική

Μελέτη και κατασκευή ρομποτικού βραχίονα 3 βαθμών ελευθερίας και ανάπτυξη συστήματος ελέγχου με χρήση PLC και GUI

Εργασία που υποβλήθηκε στο Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική, του Διεθνούς Πανεπιστημίου της Ελλάδος, για τη μερική εκπλήρωση υποχρεώσεων για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Εκπονητής: Ηλιάδης Γεώργιος

Επιβλέπων Καθηγητής: Δημήτριος Σαγρής

Σέρρες, 16-6-2020

Υπεύθυνη Δήλωση Φοιτητών:

Ο κάτωθι υπογεγραμμένος φοιτητής , έχοντας επίγνωση των συνεπειών του Νόμου περί λογοκλοπής, δηλώνει υπεύθυνα ότι είναι συγγραφέας αυτής της Μεταπτυχιακής Εργασίας, αναλαμβάνοντας την ευθύνη επί ολοκλήρου του κειμένου εξ ίσου, έχοντας δε αναφέρει στην Βιβλιογραφία όλες τις πηγές τις οποίες χρησιμοποίησε. Δηλώνει επίσης ότι, οποιοδήποτε στοιχείο ή κείμενο το οποίο έχει ενσωματώσει στην εργασία του προερχόμενο από βιβλία, άλλες εργασίες ή το διαδίκτυο, γραμμένο επακριβώς ή παραφρασμένο, το έχει πλήρως αναγνωρίσει ως πνευματικό έργο άλλου συγγραφέα και έχει αναφέρει ανελλιπώς το όνομά του και την πηγή προέλευσης.

Ο Φοιτητής:

Ηλιάδης Γεώργιος

Περίληψη

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η κατασκευή ρομποτικού βραχίονα 3 βαθμών ελευθερίας. Ο έλεγχος του ρομποτικού βραχίονα επιτυγχάνεται μέσω του συστήματος ελέγχου Programmable Logical Controller (PLC) και Graphical User Interface (GUI).

Παρουσιάζεται βήμα-βήμα ένα οικείο αυτοματιστικό εργαλείο που έχει υιοθετηθεί από για τάχιστη και ακριβή κίνηση με μικρότερο κόστος σε πολλούς τομείς εργασίας.

Αναλύεται η θεωρία των αρθρώσεων, των βαθμών ελευθερίας του βραχίονα, οι εξισώσεις που επιλύουν το ευθύ και αντίστροφο κινηματικό πρόβλημα, το πρωτόκολλο Modbus, τα servomotor,οι βηματικοί κινητήρες και όλα τα μέσα που χρησιμοποιήθηκαν. Αναπτύσσεται η δομή των PLCσυστημάτων γενικότερα και ειδικότερα στην εργασία μας.

Τέλος παρουσιάζεται ο ρομποτικός μας βραχίονας, ξεκινώντας από την σχεδίαση του, την τοποθέτηση του μειωτήρα, την υλοποίηση των ηλεκτρικών κυκλωμάτων, την εκτέλεση του κώδικά για το PLCκαι τέλος την επεξήγηση μας για το GUIκαι τα συμπεράσματα μας

Στο κεφάλαιο 1 αναλύεται η επεξήγηση του θέματος της εργασίας και οι κυριότεροι λόγοι χρήσης της τεχνικής που υλοποιήθηκε

Στο κεφάλαιο 2 παρατίθεται η ιστορική αναδρομή για τον ρομποτικό βραχίονα, την πλακέτα Arduino, τις εφαρμογές PLC και τους πλανητικούς μειωτήρες

Στο κεφάλαιο 3 παρουσιάζεται η μεθοδολογία της εργασίας και τα μέσα που χρησιμοποιήθηκαν

Στο κεφάλαιο 4 αναπτύσσεται βήμα-βήμα η υλοποίηση της εργασίας

Τέλος στο κεφάλαιο 5 κάνουμε έναν απολογισμό της εργασίας και παρουσιάζουμε την μελλοντική εξέλιξη της.

Abstract

The subject of this dissertation as developed below is the construction of a 3 degree freedom robotic arm and development of a control system using Programmable Logical Controller (PLC) and Graphical User Interface (GUI).

A familiar automation tool has been introduced step by step that has been adopted for fast and accurate movement at a lower cost in many areas of work.

The theory of the joints, the degrees of freedom of the arm, the equations that solve the straight and reverse kinematic problem, the Modbus protocol, the servomotor, the stepper motors and all the means used are analyzed. The structure of PLC systems in general and in our work in particular is being developed.

Finally, our robotic arm is presented, starting with its design, the placement of the reducer, the implementation of the electrical circuits, the execution of the code for the PLC and finally our explanation for the GUI and our conclusions.

Chapter 1 analyzes the explanation of the subject of the work and the main reasons for using the technique that was implemented

Chapter 2 lists the historical flashback to the robotic arm, the Arduino board, the PLC applications and the planetary reducers.

Chapter 3 presents the methodology of the work and the means used

In Chapter 4, the implementation of the work is developed step by step

Finally, in Chapter 5 we make a report of the work and present its future development

Περιεχόμενα

Κατάλογος εικόνων

Κατάλογος Πινάκων

1. Εισαγωγή

1.1 Επεξήγηση αντικείμενου εργασίας

Η παρούσα εργασία έχει σκοπό την υλοποίηση ενός ρομποτικού βραχίονα σε βαθμό σχεδίασης, κατασκευής όπου θα εκτελεστεί με 3D εκτυπωτή ενώ παράλληλα εκτελείταιο προγραμματισμός ελεγκτών και η δημιουργία συστήματος GUI. Ο βραχίονας μας διαθέτει έχει 3 βαθμούς ελευθερίας, κάτι που αναλύει το ευθύ του κινητικού προβλήματος αλλά και του αντίστροφου κινητικού προβλήματος ώστε να εφαρμοστεί από τον μικρό-ελεγκτή. Ο έλεγχος του βραχίονα πραγματοποιείται από το PLC ενώ η ανάλυση και η υλοποίηση του GUIπραγματοποιείται από το Arduino. Ο χειρισμός του βραχίονα πραγματοποιείται από μια οθόνη TFT και ένα πληκτρολόγιο.

1.2. Ανάλυση των κυριότερων λόγων χρήσης της τεχνικής που υλοποιήθηκε

Οι εφαρμογές οπού χρειάζεται το PLC είναι πολλαπλές από μια βιομηχανική εγκατάσταση μέχρι στο έλεγχο σιδηροδρομικού σταθμού. Ο λόγος που γίνεται απαραίτητη η χρήση του είναι ότι μπορεί να παρέχει για 365 μέρες το χρόνο και 24 ώρες την ημέρα, αδιάκοπη λειτουργιά που σε κάποιες εφαρμογές είναι θανάσιμο π.χ. πυρηνικό εργοστάσιο. Παρακάτω παρουσιάζονται κάποια από τα πιο σημαντικά πλεονεκτήματα και μειονεκτήματα.

Τα πλεονεκτήματα είναι τα εξής:

1) Μείωση του λειτουργικού κόστους και αύξηση της αξιοπιστίας του συστήματος, τα οποία οφείλονται στους εξής παράγοντες:

- Απαιτείται ελάχιστο ανθρώπινο δυναμικό, καθώς οι αυτοματισμοί αναλαμβάνουν σημαντικό τμήμα της εργασίας το οποίο φέρουν εις πέρας με μεγαλύτερη αποτελεσματικότητα. Παράλληλα, με αυτό τον τρόπο, μειώνονται τα ανθρώπινα σφάλματα σε λειτουργίες ρουτίνας τα οποία είναι και τα πλέον συνήθη.
- Εντοπίζονται και αντιμετωπίζονται ταχύτερα και αποτελεσματικότερα τα διάφορα σφάλματα αλλά και οι καταστάσεις συναγερμού (alarms). Προλαμβάνονται έτσι καταστάσεις, που θα μπορούσαν να οδηγήσουν σε μη αναστρέψιμες ή εκτεταμένες βλάβες και επιπλέον οι όποιες αστοχίες αντιμετωπίζονται στον ελάχιστο δυνατό χρόνο.

2) Μείωση του κόστους συντήρησης, το οποίο αποτελεί έναν πάγιο στόχο εκείνων που σχεδιάζουν και υλοποιούν τέτοια συστήματα.

- Μείωση των αστοχιών και του χρόνου αποκατάστασής τους. Παλαιοτέρα, και το πλέον απλό σύστημα αυτοματισμού και διαχείρισης μιας εγκατάστασης, απαιτούσε αρκετά χιλιόμετρα καλωδίωσης αλλά και πολύπλοκο εξοπλισμό, γεγονός που δημιουργούσε πολλές εστίες πιθανών σφαλμάτων και επιπλέον, όταν αυτά συνέβησαν ο εντοπισμός τους αλλά και η διόρθωση τους αποτελούσαν σύνθετη και επίπονη διαδικασία. Η εισαγωγή ταχύτατων δικτύων υπολογιστών έλυσε σε σημαντικό βαθμό όλα αυτά τα προβλήματα.
- Μείωση του κόστους τακτικής συντήρησης του κύριου εξοπλισμού μίας εγκατάστασης, αφού μπορούμε να έχουμε μια πραγματική εικόνα του τρόπου λειτουργίας αλλά και των καταπονήσεών του.
- Μείωση του κόστους συντήρησης και λειτουργίας των συστημάτων ελέγχου και προστασίας. Η εισαγωγή των PLC (προγραμματιζόμενων λογικών ελεγκτών) αλλά και των μικροεπεξεργαστών ,έδωσε δυνατότητα πραγματοποίησης εξαιρετικά σύνθετων λειτουργιών ελέγχου και προστασίας, οι οποίες με χρήση απλών αναλογικών ηλεκτρονόμων και διακοπτών θα ήταν όχι απλώς δύσκολες αλλά και σε πολλές περιπτώσεις αδύνατες. Οι σύγχρονες εγκαταστάσεις δεν είναι μόνο πιο πολύπλοκες, αλλά έχουν και μεγαλύτερες διαστάσεις, οπότε η χρήση σύγχρονων πληροφοριακών συστημάτων είναι μονόδρομος.

3) Μείωση του κόστους του εγκατεστημένου υλικού, η οποία μπορεί να προέλθει από τα εξής σημεία:

- Μείωση του κόστους καλωδίωσης για τα συστήματα ελέγχου και προστασίας. Κάθε μεγάλη εγκατάσταση ελέγχου και προστασίας, απαιτεί αρκετά χιλιόμετρα καλωδίων που πέρα από το υψηλό κόστος προκαλεί και άλλα προβλήματα και περιορισμούς, όπως αυτεπαγωγές, απώλειες σήματος και ηλεκτρομαγνητικές παρεμβολές.
- Μείωση του κόστους εγκατάστασης εξοπλισμού σχετιζόμενου με συγκεκριμένες λειτουργίες. Με τα συμβατικά μέσα, κάθε νέα λειτουργία

απαιτούσε την αγορά και εγκατάσταση νέου εξοπλισμού καθώς και καλωδίωσης. Κάτι τέτοιο δεν είναι απαραίτητο με τα νέα συστήματα SCADA, αφού το μόνο που πιθανόν να απαιτείται επιπλέον, είναι καινούργιοι αισθητήρες, οι οποίοι προσαρμόζονται στον υπάρχοντα εξοπλισμό με κατάλληλη επέμβαση στο λογισμικό. Επιπλέον, η εγκατάσταση του νέου εξοπλισμού, στις περισσότερες περιπτώσεις, δεν έχει σημαντικές απαιτήσεις ως προς την κτιριακή υποδομή, αφού τέτοια συστήματα καταλαμβάνουν πολύ μικρό χώρο.

4) Καλύτερη και αποτελεσματικότερη διαχείριση της παραγωγής και κατανάλωσης ενέργειας.

Το πρόβλημα της βέλτιστης χρησιμοποίησης της ηλεκτρικής αλλά και των λοιπών μορφών ενέργειας, είναι ένα θέμα που απασχολεί σήμερα όλα τα αναπτυγμένα κράτη, όχι μόνο επειδή αυξάνεται η τιμή της kWh, αλλά και επειδή απαιτούνται όλο και μεγαλύτερα ποσά ενέργειας, ενώ παράλληλα, τα ενεργειακά αποθέματα εξαντλούνται. Προκειμένου οι βιομηχανίες να αντιμετωπίσουν το φλέγον αυτό ζήτημα κινούνται σε δύο βασικούς άξονες, ένα μακροπρόθεσμο και ένα βραχυπρόθεσμο. Μακροπρόθεσμα, οι βιομηχανίες θέλουν να γνωρίζουν με τη μέγιστη δυνατή ακρίβεια, τη ζήτηση φορτίου ανά ώρα, ημέρα, μήνα και έτος, ώστε να ελαχιστοποιούν την κατανάλωσή τους, με κριτήριο την εξοικονόμηση ενέργειας και επιπλέον να σχεδιάζουν τη στρατηγική τους για τις μελλοντικές χρονικές περιόδους. Βραχυπρόθεσμα, απαιτείται η -κατά το δυνατόν- ομαλοποίηση της ροής της ενέργειας, προκειμένου να αποφεύγουμε εξαιρετικά οδυνηρές καταστάσεις, όπως οι υπερβάσεις του φορτίου αλλά και του συντελεστή ισχύος.

5) Η λειτουργιά του αυτοματισμού μπορεί να αλλάξει σε οποιοδήποτε στάδιο.

- Όταν καταστεί αναγκαίο στη βιομηχανία η επέκταση μία νέας μονάδας, είναι εφικτό να προστεθούν τα νέα μηχανήματα απλά προσθέτοντας νέες κάρτες μνήμης στα PLC μας.
- Οποιαδήποτε αλλαγή σε κάποια υπάρχουσα γραμμή παράγωγης μπορεί να πραγματοποιηθεί δημιουργώντας καινούρια μπλοκ στο PLC.
- Πρόσθεση κάποιου καινούριου αισθητήριου οργάνου ή κινητήρα.

6) Η τοποθέτηση μπορεί να γίνει ακίνδυνα ανάμεσα σε πεδία ισχύος.

7) Η γλώσσα προγραμματισμού είναι προσαρμοσμένη στο βιομηχανικό αυτοματισμού με συνέπεια να είναι προσιτή στο προσωπικό που μέχρι σήμερα συντηρούσε τους κλασσικούς πίνακες αυτοματισμού.

8) Τα PLC έχουν μηδενική συντήρηση ,σχεδόν απεριόριστη διάρκεια ζωής λόγω των ασθενών ρευμάτων που καταναλώνουν οπού και αυτό έχει οικονομικότερη από πλευράς κατανάλωσης ηλεκτρική ενέργεια .

Βέβαια υπάρχουν και τα παρακάτω μειονεκτήματα:

1) Θα μπορούσε να θεωρηθεί πως για να πραγματοποιηθεί μια σημαντική αλλαγή στο πρόγραμμα του PLC σε επίπεδο εντολών πρέπει να υπαρχή ειδικευμένο άτομο ή προσωπικό .Για οποιοδήποτε άλλη αλλαγή του τύπου χρονικών , μετρητών και καταχωρητών μπορεί να γίνει από ένα panel που το χρησιμοποιεί ένα άτομο απλώς να μελετώντας το εγχειρίδιο.

2) Το μικρό μέγεθος μνήμης, η αδυναμία αποθήκευσης μεγάλων ποσοτήτων δεδομένων βελτιώνεται με τις καινούριες σειρές.

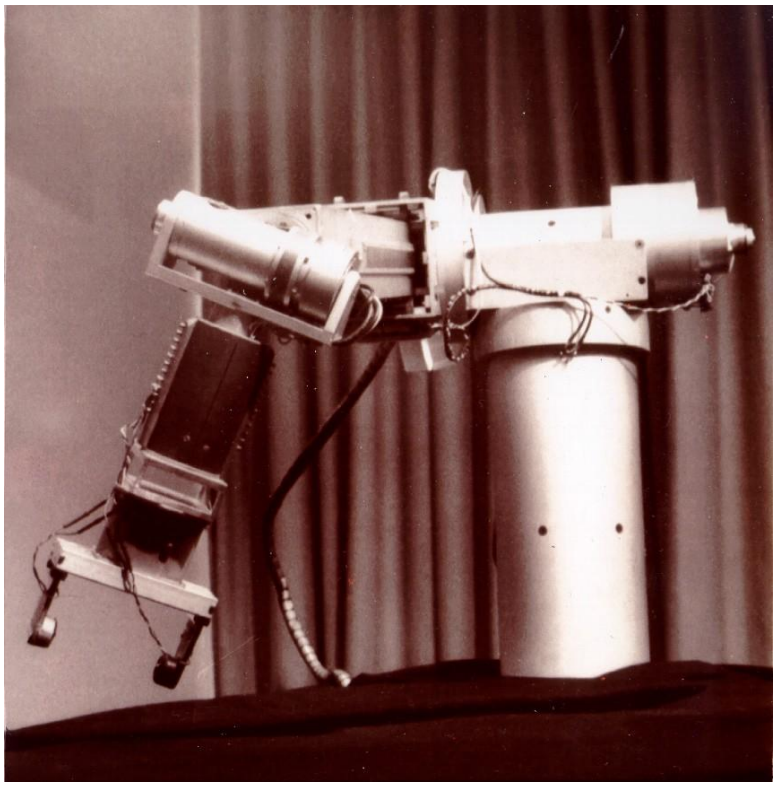
3) Το πρωτόκολλο επικοινωνίας του PLC όπως το Ethernet πρέπει να λειτουργεί ξεχωριστά από το αντίστοιχο πρωτόκολλο του εταιρικού δικτύου γιατί σε περιπτώσεις συνεχούς επικοινωνίας , οποιαδήποτε υπερφόρτωση του δικτύου προκαλεί βλάβες στο σύστημα του αυτοματισμού.

2. Ιστορική αναδρομή

2.1 Ο ρομποτικός βραχίονας στο πέρασμα των χρόνων

Η πρώτη εταιρεία που παρήγαγε ρομπότ ήταν η Unimation, που ιδρύθηκε από τον Devol και τον Joseph F. Engelberger το 1956 και αρχικά βασίστηκε στο δίπλωμα ευρεσιτεχνίας του Devol. Τα ρομπότ της Unimation που ονομαζόταν επίσης και μηχανές προγραμματισμένων μεταφορών, λόγω της κύριας λειτουργίας τους που ήταν η μεταφορά αντικειμένων από ένα σημείο σε κάποιο άλλο, για αποστάσεις 4 μέτρων το πολύ. Χρησιμοποιούσαν υδραυλικούς ενεργοποιητές και είχαν προγραμματιστεί σε κοινές συντεταγμένες, δηλαδή οι γωνίες των διαφόρων αρθρώσεων αποθηκεύονταν κατά τη διάρκεια μιας φάσης διδασκαλίας και να αναπαράγονταν κατά τη λειτουργία. Ήταν ακριβή κατά 1/10,000 της ίντσας. (σημ.: αν και η ακρίβεια δεν είναι το κατάλληλο μέτρο για τα ρομπότ, που συνήθως αξιολογούνται από τον ορισμό της επανάληψης). Η Unimation αργότερα αδειοδότησε την Kawasaki Heavy Industries και την Guest-Nettlefolds κατασκευάζοντας τα Unimates στην Ιαπωνία και την Αγγλία αντίστοιχα. Για αρκετό καιρό ο μοναδικός ανταγωνιστής της Unimation ήταν η Cincinnati Milacron Inc. του Οχάιο. Αυτό άλλαξε ριζικά στα τέλη της δεκαετίας του 1970, όταν πολλοί μεγάλοι ιαπωνικοί όμιλοι άρχισαν να παράγουν παρόμοια βιομηχανικά ρομπότ.

Το 1969 ο Victor Scheinman στο Πανεπιστήμιο του Στάνφορντ ανακάλυψε το "βραχίονα του Στάνφορντ", έναν πλήρως ηλεκτρικό, 6 - αρθρωτό ρομποτικό άξονα σχεδιασμένο για να καταστεί δυνατή η λύση του βραχίονα. Αυτό επέτρεψε να ακολουθεί με ακρίβεια αυθαίρετες διαδρομές στο χώρο και διεύρυνε τις δυνατότητες χρήσης του ρομπότ σε πιο εξελιγμένες εφαρμογές, όπως η συναρμολόγηση και συγκόλληση. Ο Scheinman σχεδίασε κι ένα δεύτερο βραχίονα για το εργαστήριο Τεχνητής Νοημοσύνης του MIT. Αφού έλαβε μια υποτροφία από την Unimation για να εξελίξει τα σχέδια του, στη συνέχεια τα πούλησε στην ίδια εταιρία, όπου συνέχισε να τα εξελίσσει με την υποστήριξη της General Motors και έπειτα το έβγαλε στην αγορά ως την καθολικά προγραμματιζόμενη μηχανή για συναρμολόγηση (PUMA).[13]



Εικόνα 2.1: Ο βραχίονας του Stanford[2]

Η ρομποτική βιομηχανία απογειώθηκε πολύ γρήγορα στην Ευρώπη, τόσο από την ABB Robotics όσο και από την KUKA Robotics όπου έφεραν ρομπότ στην αγορά το 1973. Η ABB Robotics (πρώην ASEA) Εισήγαγε την IRB 6, μεταξύ των πρώτων στον κόσμο που διατίθεντο στο εμπόριο, εξολοκλήρου ηλεκτρικά ρομπότ που ελέγχονταν από μικροεπεξεργαστή. Τα δύο πρώτα ρομπότ IRB 6 πωλήθηκαν στην Magnusson στη Σουηδία για λείανση και στίλβωση των γωνιών σε σωλήνες και εγκαταστάθηκαν στην παραγωγή τον Ιανουάριο του 1974. Επίσης, το 1973 η KUKA Robotics δημιούργησε το πρώτο ρομπότ, γνωστό ως FAMULUS,[2] επίσης, ένα από τα πρώτα αρθρωτά ρομπότ που δούλευαν με έξι ηλεκτρομηχανικούς άξονες.

Το ενδιαφέρον στη ρομποτική αυξήθηκε στα τέλη του 1970 και πολλές εταιρείες των ΗΠΑ εισήλθαν στον τομέα, συμπεριλαμβανομένων των μεγάλων εταιρειών όπως η General Electric, και η General Motors (η οποία σχημάτισε με κοινοπραξία την FANUC Robotics με την FANUC LTD της Ιαπωνίας). Στις πρωτοπόρες εταιρίες περιλαμβάνονται η Automatrix και η Adept Technology Inc. Στην κορύφωση της έκρηξης της ρομποτικής το 1984 η

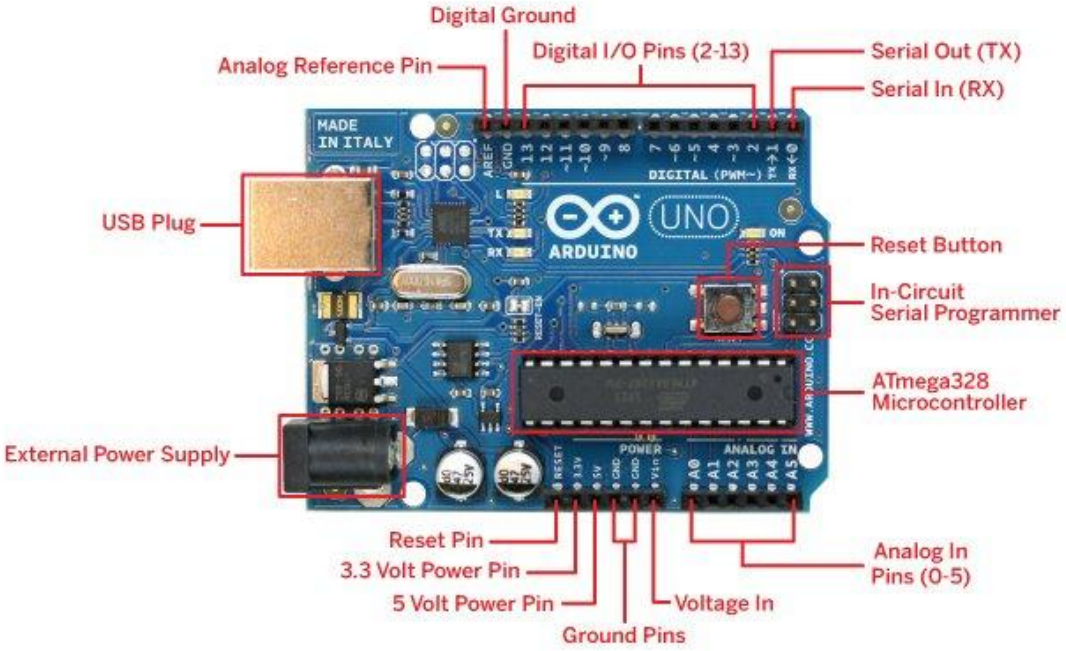
Unimation εξαγοράστηκε από την Westinghouse Electric Corporation έναντι 107 εκατομμυρίων δολαρίων. Η Westinghouse πούλησε την Unimation στην Γαλλική Stäubli Faverges SCA το 1988, η οποία ακόμα παράγει αρθρωτά ρομπότ για γενικές βιομηχανικές εφαρμογές, η οποία αγόρασε ακόμη και το ρομποτικό τμήμα της Bosch στα τέλη του 2004.

Μόνο λίγες μη Ιαπωνικές εταιρίες κατάφεραν να επιβιώσουν σε αυτή την αγορά, οι κυριότερες είναι η Adept Technology, η Stäubli-Unimation, η Swedish-Swiss η ABB Asea Brown Boveri και η Γερμανική KUKA Robotics.[13]

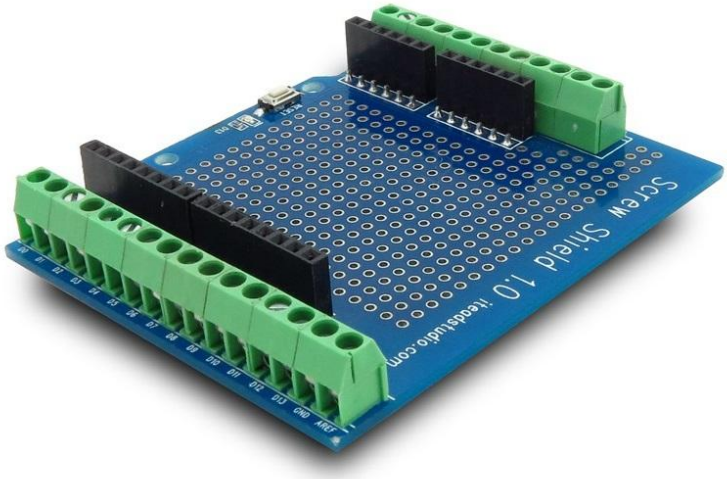
2.2 Η πλακέτα Arduino

Το Arduino είναι μια ανοιχτού κώδικα πλακέτα - μητρική. Αναπτύχτηκε το 2005 στην Ιταλία με ιδρυτές Massimo Banzi και David Cueartielles. Σκοπός ήταν να δημιουργηθεί ένα οικονομικό αντικείμενο ώστε να προσφέρει στους μαθητές διακρατικό έλεγχο προγραμμάτων. Το Arduino προγραμματίζεται στο περιβάλλον Arduino IDE,παρέχει ψηφιακές εισόδους , ψηφιακές εξόδους, αναλογικές εισόδους - εξόδους, μπορεί να επικοινωνήσει σειριακά αλλά και με διάφορα Shields μπορεί με Ethernet η Wi-Fi. Η τροφοδοσία του μπορεί να είναι είτε με usb 5V είτε με μπαταρία 9-12V.

Επίσης χρησιμοποιεί τον μικροελεγκτή Atmega 328 αλλά αυτό αλλάζει ανάλογα το μοντέλο. Υπάρχουν περισσότερα από δέκα διαφορετικά Arduino,στην παρούσα εφαρμογή θα χρησιμοποιηθεί το ArduinoUNO.Ο Arduino μπορεί να δεχτεί διάφορα Shields για την αλλαγή εφαρμογή στην παρούσα χρησιμοποιεί ScrewShield που φαίνεται στην εικόνα 2.2. και 2.3 [1]



Εικόνα 2.2 Επεξήγηση πλακέτας[6]



Εικόνα 2.3:ScrewShield

2.3 Εφαρμογές των PLC στην ιστορία

Στις αρχές της δεκαετίας οπού είχε ήδη πραγματοποιηθεί η πρώτη κατασκευή μικροελεγκτή, έγινε η προσπάθεια εισαγωγής του μικροελεγκτή για τον βιομηχανικό έλεγχο και επιτήρηση. Σε αυτό το σημείο εμφανίζεται το PLC με πλήρης ονομασία programmable logic controller. Η μετάβαση ομως στην βιομηχανία απαιτούσε μεγάλο κόστος και δυσκολίες ώστε να γίνει αντικατάσταση των ηλεκτρολογικών πινάκων, έτσι η μετάβαση έγινε αντιγράφοντας τα ηλεκτρικά κυκλώματα στο πρόγραμμα του PLC όπως γίνεται μέχρι σήμερα. Πλέον από τους κύριους κατασκευαστές είναι Siemens,Omron,Allen-Bradley,Mitsubishi,ABB.

Εικόνα 2.4PLC της Mitsubishifx3u

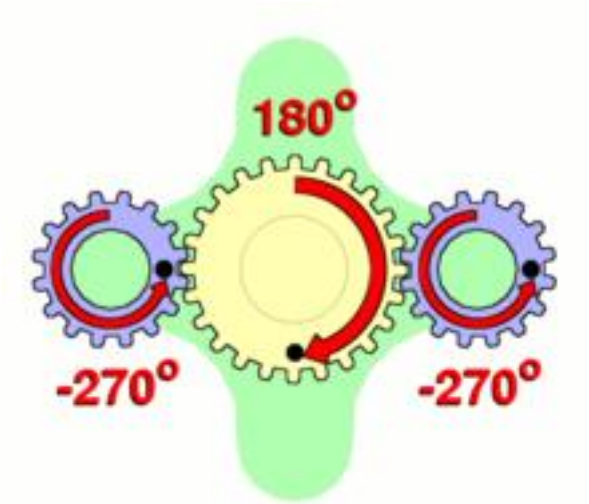
2.4 Πλανητικοί μειωτήρες

Περίπου το 500 π.Χ., οι Έλληνες ανακάλυψαν την ιδέα των επίκυκλων, κύκλων που ταξιδεύουν στις κυκλικές τροχιές. Με αυτή τη θεωρία ο Πτολεμαίος στο Almagest το 148 μ.Χ. μπόρεσε να προβλέψει πλανητικές τροχιές. Ο Μηχανισμός των Αντικυθήρων, γύρω στο 80 π.Χ., είχε έναν εξοπλισμό που μπόρεσε να προσεγγίσει το ελλειπτικό μονοπάτι του φεγγαριού μέσα από τους ουρανούς, και ακόμη και να διορθώσει την εννέα χρόνων ύφεση αυτού του μονοπατιού. Ο Πτολεμαίος χρησιμοποίησε περιστρεφόμενες αναβολές και επίκυκλους που σχηματίζουν επικυκλικά τρένα για να προβλέψουν τις κινήσεις των πλανητών. Ακριβείς προβλέψεις για την κίνηση του Ήλιου, της Σελήνης και των πέντε πλανητών, του Ερμή, της Αφροδίτης, του Άρη, του Δία και του Κρόνου, πέρα από τον

ουρανό, υποθέτουν ότι ο καθένας ακολούθησε μια τροχιά που εντοπίστηκε από ένα σημείο στον πλανητικό γρανάζι ενός επικυκλικού γραναζιού. Αυτή η καμπύλη ονομάζεται επιτονοειδές.[14]

Το Epicyclic gearing χρησιμοποιήθηκε στον Μηχανισμό των Αντικυθήρων, περίπου το 80 π.Χ., για να προσαρμόσει την εμφανιζόμενη θέση του φεγγαριού για την ελλειπτικότητα της τροχιάς του, ακόμη και για την απροσδόκητη πορεία της τροχιάς του. Δύο γρανάζια που στραφούν περιστράφηκαν γύρω από ελαφρώς διαφορετικά κέντρα, και το ένα οδήγησε το άλλο όχι με πλεγμένα δόντια αλλά με έναν πείρο τοποθετημένο σε μια σχισμή στο δεύτερο. Καθώς η υποδοχή οδήγησε τη δεύτερη ταχύτητα, η ακτίνα οδήγησης θα άλλαζε, προκαλώντας έτσι μια επιτάχυνση και επιβράδυνση της κίνησης σε κάθε επανάσταση.

Τον 11ο αιώνα μ.Χ., η επικυκλική μετάδοση επινοήθηκε εκ νέου από τον Ibn Khalaf al-Muradi στο Al-Andalus. Το ρολόι με γρανάζια νερού χρησιμοποίησε έναν πολύπλοκο μηχανισμό μετάδοσης κίνησης που περιλάμβανε τόσο τμηματική όσο και επικυκλική ταχύτητα.



Εικόνα 2.5: Απεικόνιση πλανητικού γραναζιού[1]

Ένα επικόκυκλο γρανάζι (επίσης γνωστό ως πλανητικό γρανάζι) αποτελείται από δύο γρανάζια τοποθετημένα έτσι ώστε το κέντρο του ενός γραναζιού να περιστρέφεται γύρω από το κέντρο του άλλου. Ένας μεταφορέας συνδέει τα κέντρα των δύο γραναζιών και περιστρέφεται για να φέρει ένα γρανάζι, που ονομάζεται γρανάζι πλανήτη, γύρω από το άλλο,

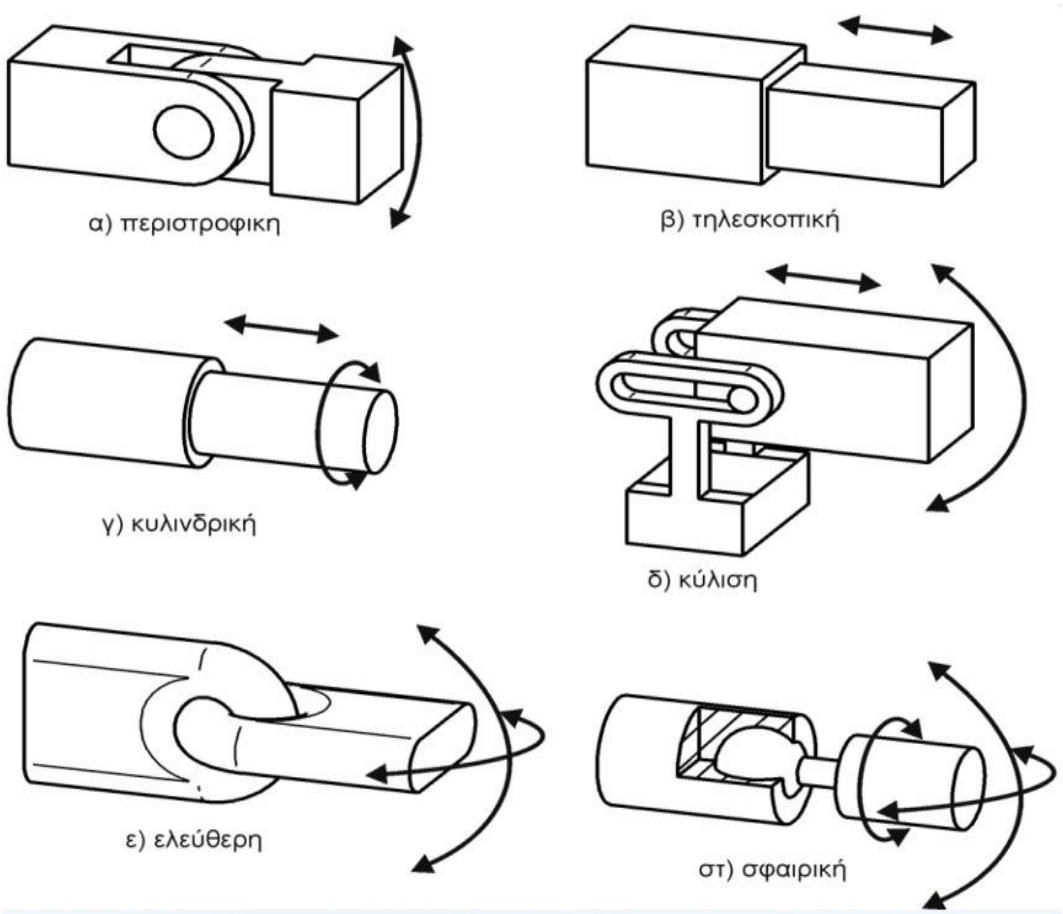
που ονομάζεται γρανάζι ή ηλιακό τροχό. Τα γρανάζια του πλανήτη και του ήλιου πλένονται έτσι ώστε οι κύκλοι του βήματος να κυλούν χωρίς ολίσθηση. Ένα σημείο στον κύκλο βήματος του πλανητικού γραναζιού εντοπίζει μια επικυκλοειδή καμπύλη. Σε αυτήν την απλοποιημένη περίπτωση, το γρανάζι είναι σταθερό και τα πλανητικά γρανάζια περιστρέφονται γύρω από το γρανάζι του ήλιου.

Μπορεί να συναρμολογηθεί ένα επικόκυκλο γρανάζι, έτσι ώστε το γρανάζι του πλανήτη να κυλά στο εσωτερικό του κύκλου βήματος ενός σταθερού, εξωτερικού δακτυλίου γραναζιού ή δακτυλίου, που μερικές φορές ονομάζεται δακτυλιοειδές γρανάζι. Σε αυτήν την περίπτωση, η καμπύλη που εντοπίζεται από ένα σημείο στον κύκλο βήματος του πλανήτη είναι υποκυκλοειδές.[14]

3. Μεθοδολογία εργασίας και ανάπτυξη επίλυσης

3.1.Αρθρώσεις

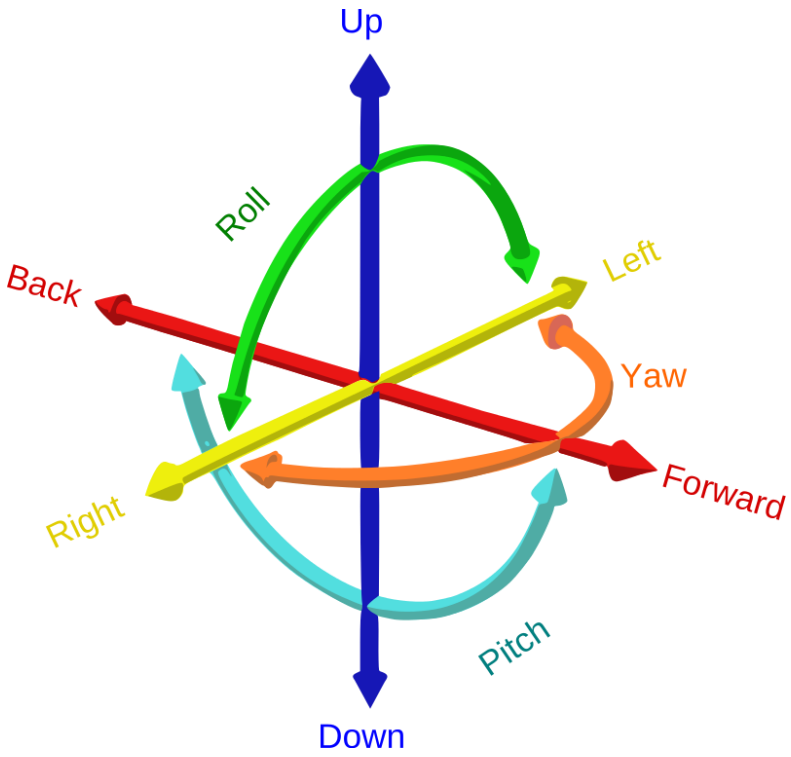
Οι αρθρώσεις είναι μηχανισμοί που επιτρέπουν τη κίνηση μεταξύ των συνδέσμων και με την βοήθεια των ενεργοποιητών κινούν τους συνδέσμους. Κυρίως χρησιμοποιούνται περιστροφικές αρθρώσεις. Στην παρακάτω εικόνα 3.1 αναπαρίστανται τα ειδή των αρθρώσεων.



Εικόνα 3.1: Ειδή αρθρώσεων[3]

3.2 Βαθμοί ελευθερίας (Dof) και βαθμοί κινητικότητας (Dom)

Οι βαθμοί ελευθερίας(Dof) στα ρομποτικά συστήματα σχετίζονται με την δυνατότητα του συστήματος καισε πόσες διαστάσεις στο χώρο μπορεί να κινηθεί(x,y,z) και σε ποιον προσανατολισμό έχει(roll,pitch,yaw).

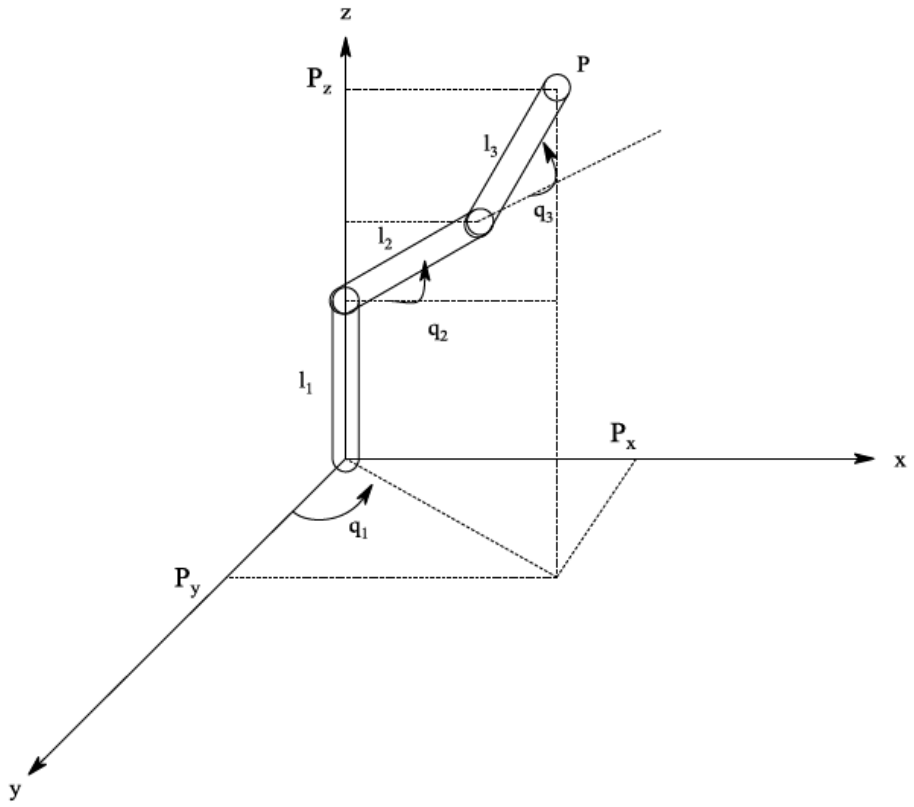


Εικόνα 3.2: Έξι βαθμοί ελευθερίας[1]

Οι βαθμοί κινητικότητας είναι άμεσα συνδεδεμένοι με τις αρθρώσεις δηλαδή όσες αρθρώσεις έχει το σύστημα τόσους βαθμούς κινητικότητας αποκτά, εκτός αν η άρθρωση δεν προσφέρει βαθμό κινητικότητας και το σύστημα πλεονεκτεί.

3.3 Ευθύ κινηματικό πρόβλημα

Το Ευθύ κινηματικό πρόβλημα έχει στόχο την εφεύρεση της θέσης και τον προσανατολισμό του τελικού σημείου δράσης με δεδομένα τις μεταβλητές των αρθρώσεων. Στην εικόνα 3.3 φαίνεται ο βραχίονας που έχει αντικείμενο η διπλωματική εργασία όπου θα παρουσιαστεί η ανάλυση του βραχίονα[5].



Εικόνα 3.3 Βραχίονας 3Dof[4]

Από την τριγωνομετρία η εφαπτόμενη έχει περίοδο π ενώ το συνημίτονο και το ημίτονο έχει 2π. Είναι γνωστό ότι:

$$x = r\cos\theta, y = r\sin\theta, r > 0$$

όπου συνεπάγετε

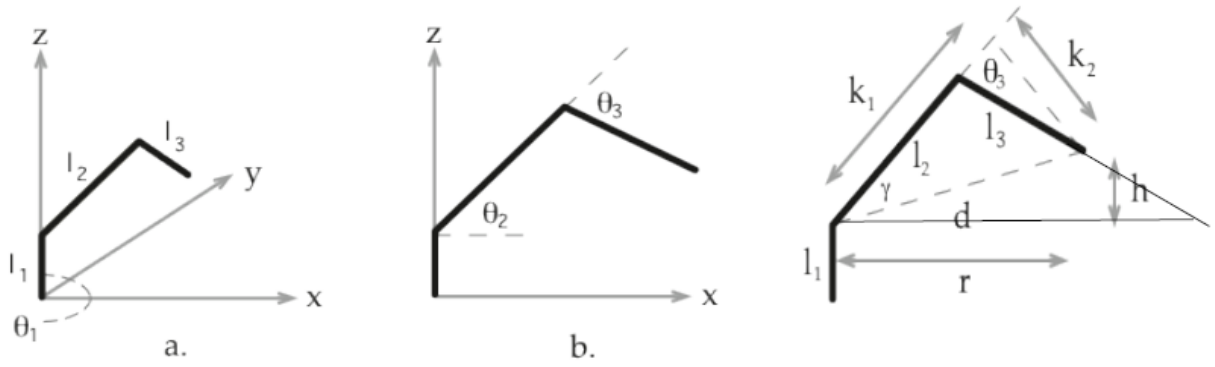
$$\theta = arc\tan\left(\frac{y}{x}\right) = arc\tan\left(\frac{r\sin\theta}{r\cos\theta}\right)$$

όμως έχουν λύσεις μόνο στο πρώτο και τέταρτο τεταρτημόριο. Οι τιμές του x και του y καθορίζουν σε πιο τεταρτημόριο είναι, έτσι προκύπτει:

$$\theta \leftarrow \text{atan2}(y, x)$$

και αυτό συνεπάγετε:

$$\tan\theta = \frac{y}{x}, sin\theta = \frac{y}{\sqrt{x^2+y^2}}, \cos\theta = \frac{x}{\sqrt{x^2+y^2}}$$



Εικόνα 3.4Απεικόνιση βραχίονα σε δυο διαστάσεις[5]

Για συντομία θα γραφτούν τα εξής:

$$S1 = \sin\theta 1$$

$$C1 = \cos\theta 1$$

$$\theta_{3-2} = \theta_3 - \theta_2$$

$$s_{3-2} = \sin\theta_{3-2} = \sin(\theta_3 - \theta_2)$$
$$c_{3-2} = \cos\theta_{3-2} = \cos(\theta_3 - \theta_2)$$

Έτσι προκύπτουν οι εξισώσεις:

$$r = \sqrt{x^2 + y^2} = l_2 c_2 + l_3 c_{3-2}$$

$$h = z - l_1 = l_2 s_2 - l_3 s_{3-2}$$

$$x = [l_2 c_2 + l_3 c_{3-2}] c_1$$
$$y = [l_2 c_2 + l_3 c_{3-2}] s_1$$
$$z = l_1 + l_2 s_2 - l_3 s_{3-2}$$

3.4 Αντίστροφο κινηματικό πρόβλημα

Προηγουμένως στο ευθύ κινηματικό πρόβλημα τα δεδομένα ήταν οι μεταβλητές των αρθρώσεων όπου το ζητούμενο ήταν η θέση και ο προσανατολισμός του τελικού σημείου δράσης. Σε αυτό το πρόβλημα θα εκτελεστεί η αντίστροφη ανάλυση με δεδομένα την θέση και των προσανατολισμό του τελικού σημείου δράσης ώστε να προσδιοριστούν οι μεταβλητές των αρθρώσεων.[5]

Το αντίστροφο κινηματικό πρόβλημα είναι πιο περίπλοκο σε σχέση με το ευθύ κινηματικό πρόβλημα για τους εξής λόγους:

- Μη γραμμικότητα
- Αντίστροφη τριγωνομετρία
- Περισσότερες μεταβλητές
- Πολλαπλές λύσεις για μία καρτεσιανή θέση ασυνέχειες και μοναδικότητες
- Μπορεί να χαθεί ένας ή περισσότεροι βαθμοί ελευθερίας σε ορισμένες διαμορφώσεις
- Πολλοί περιορισμένοι ώστε να μην υπάρχει ακριβής λύση
- Χρήση προσεγγίσεων και επαναληπτικών αλγορίθμων
- Η εφαρμογή στην πραγματικότητα απαιτεί υπολογισμό δυνάμεων, ροπών, φυσικούς περιορισμός

Από την παρακάτω εξίσωση προκύπτει ότι η θ1 ισούται με την αντίστροφη εφαπτόμενη που σχηματίζουν τα σημεία y και x:

$$\theta_1 \leftarrow \text{atan2}(y, x)$$

$$r = \sqrt{x^2 + y^2} = l_2 c_2 + l_3 c_{3-2}$$
$$h = z - l_1 = l_2 s_2 - l_3 s_{3-2}$$

όπου εφαρμόζοντας το Πυθαγόρειο θεώρημα προκύπτει:

$$x^2 + y^2 + (z - l_1)^2 = r^2 + h^2$$
$$= l_2^2 c_2^2 + l_3^2 c_{3-2}^2 + 2 l_2 l_3 c_2 [c_3 c_2 + s_3 s_2] + l_2^2 s_2^2 + l_3^2 s_{3-2}^2 - 2 l_2 l_3 s_2 [s_3 c_2 - c_3 s_2]$$
$$= l_2^2 + l_3^2 + 2 l_2 l_3 [c_2^2 c_3 + s_2^2 c_3]$$
$$= l_2^2 + l_3^2 + 2 l_2 l_3 c_3$$

και λύνοντας ως προς cosθ3:

$$c_3 = \frac{x^2 + y^2 + (z - l_1)^2 - l_2^2 - l_3^2}{2 l_2 l_3}$$

Από την εξίσωση:

$$sin^2\theta + cos^2\theta = 1$$

προκύπτει :

$$s_3 = +\sqrt{1 + c_3^2}$$

όπου μπορεί να προσδιοριστεί ο πάνω αγκώνας ενώ από την παρακάτω εξίσωση ο κάτω αγκώνας:

$$s_3 = -\sqrt{1 - c_3^2}$$

Έτσι ώστε από τα παραπάνω προκύπτει:

$$\theta_3 = \text{atan2}(s_3, c_3)$$

Λύνοντας τις παρακάτω εξισώσεις:

$$r = l_2 c_2 + l_3 c_{3-2} = l_2 c_2 + l_3 c_3 c_2 + l_3 s_3 s_2 = k_1 c_2 + k_2 s_2$$
$$r = l_2 c_2 - l_3 s_{3-2} = l_2 s_2 - l_3 s_3 c_2 + l_3 c_3 s_2 = k_1 s_2 - k_2 c_2$$

όπου

$$k_1 = l_2 + l_3 c_3$$
$$k_2 = l_3 s_3$$

Και

$$d = +\sqrt{k_1^2 + k_2^2} \quad k_1 = d \cos\gamma$$
$$\gamma = \text{atan2}(k_2, k_1) \qquad k_2 = d \sin\gamma$$

με αντικατάσταση προκύπτει:

$$r = d \cos\gamma \cos\theta_2 + d \sin\gamma \sin\theta_2 = d \cos(\theta_2 - \gamma)$$
$$h = d \cos\gamma \sin\theta_2 - d \sin\gamma \cos\theta_2 = d \sin(\theta_2 - \gamma)$$

λύνοντας ως προς θ2:

$$\theta_2 - \gamma = \text{atan2}\left(\frac{h}{d}, \frac{r}{d}\right) = \text{atan2(h, r)}$$

$$d > 0$$

$$\theta_2 \leftarrow \text{atan2}\left(z - l_1, \quad \sqrt{x^2 + y^2}\right) - \text{atan2}(l_3 s_3, l_2 + l_3 c_3)$$

Άρα για τον βραχίονα ισχύουν τα παρακάτω δεδομένα:

$$\theta_1 \leftarrow \text{atan2}(y, x)$$

$$c_3 \leftarrow \frac{x^2 + y^2 + (z - l_1)^2 - l_2^2 - l_3^2}{2 l_2 l_3}$$

$$s_3 \leftarrow +\sqrt{1 - c_3^2}$$

$$\theta_3 \leftarrow \text{atan2}(s_3 c_3)$$

$$\theta_2 \leftarrow \text{atan2}\left(z - l_1, \sqrt{x^2 + y^2}\right) - \text{atan2}(l_3 s_3, l_2 + l_3 c_3)$$

3.5 Modbus

Το Modbus είναι ένα πρωτόκολλο επικοινωνίας που βασίζεται στην αρχιτεκτονική master-slave. Χρησιμοποιεί τις διασυνδέσεις RS-485, RS-422, RS-232, καθώς και τα δίκτυα TCP / IP Ethernet (πρωτόκολλο TCP Modbus) για τη μεταφορά δεδομένων.[7]

Στην συγκεκριμένη εφαρμογή θα χρησιμοποιηθεί το Modbus RTU που αποτελείται από τη διεύθυνση της συσκευής SlaveID, τον κωδικό λειτουργίας και τα ειδικά δεδομένα, ανάλογα με τον κωδικό λειτουργίας και το CRC άθροισμα ελέγχου.

Το SlaveID είναι η διεύθυνση της συσκευής και μπορεί να πάρει τιμές από το 0 έως το 247, διατηρούνται οι διευθύνσεις από 248 έως 255.Τα δεδομένα στην μονάδα αποθηκεύονται σε 4 πίνακες. Δύο πίνακες είναι μόνο για ανάγνωση και δύο είναι ανάγνωσης-εγγραφής. Οι 9999 τιμές χωράνε σε κάθε πίνακα

Υλικό	**Διεύθυνσης Μητρώου HEX**	**Τύπος**	**Όνομα**	**Τύπος**
1-9999	0000 με 270^{E}	Ανάγνωση-Εγγραφή	Discrete Output Coils	DO
1001-19999	0000 με 270^{E}	Ανάγνωση	Discrete Input Contacts	DI
30001-39999	0000 με 270^{E}	Ανάγνωση	Analog input registers	AI
40001-49999	0000 με 270^{E}	Εγγραφή	Analog output holding Registers	AO

Πίνακας 1: Πίνακας Modbus[7]

Το μήνυμα Modbus χρησιμοποιεί τη διεύθυνση μητρώου. Για παράδειγμα, το πρώτο μητρώο του AO Holding Register έχει τον αριθμό 40001, αλλά η διεύθυνσή του είναι 0000.Η διαφορά μεταξύ αυτών των δύο ποσοτήτων είναι "αντισταθμισμένη». Ο κάθε πίνακας έχει τη δική του αντιστάθμιση, αντίστοιχα: 1, 10001, 30001 και 40001.Ακολουθεί ένα παράδειγμα αιτήματος

Modbus RTU για την απόκτηση της τιμής AI αναλογικής εξόδου (holding registers) από τα μητρώα # 40108 έως 40110 με τη διεύθυνση της συσκευής 17.

11 03 006B 0003 7687

11	Η διεύθυνση της συσκευής SlaveID (17=11 HEX)
03	Κωδικός λειτουργίας Function Code
006B	Η διεύθυνση του πρώτου μητρώου (40108-40001=107=6Bhex)
0003	Ο αριθμός των απαιτούμενων μητρώων (η ανάγνωση των 3 μητρώων από 40108 έως 40110
7687	CRC άθροισμα ελέγχου

Πίνακας 2: Ανάλυση διεύθυνσης[7]

Λαμβάνουμε ως απάντηση της συσκευής Modbus RTU Slave:

11 03 06 AE41 5652 4340 49AD

11	διεύθυνση της συσκευής (17=11 HEX)	SlaveID
03	Κωδικός λειτουργίας	Function Code
06	Ο αριθμός των bitπαρακάτω(6 bytesfollow)	Byte Count
AE	Η τιμή bittου υψηλού μητρώου (AEhex)	Register Value Hi (AO0)
41	Η τιμή bit του χαμηλού μητρώου (41 hex)	Register Value Lo(AO0)
56	Η τιμή bit του υψηλού μητρώου (56 hex)	Register Value Hi (AO1)
52	Η τιμή bit του χαμηλού μητρώου (52 hex)	Register Value Lo(AO1)
43	Η τιμή bit του υψηλού μητρώου(43 hex)	Register Value Hi (AO2)
40	Η τιμή bit του χαμηλού μητρώου (40 hex)	Register Value Lo(AO2)
49	Άθροισμα ελέγχου	CRC value Hi
AD	Άθροισμα ελέγχου	CRC value Lo

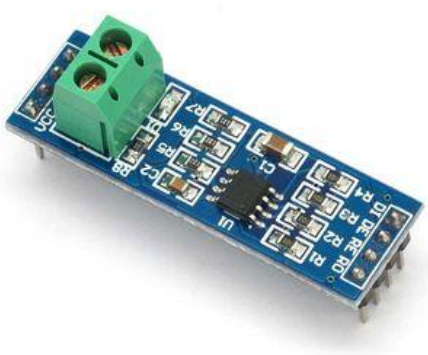
Πίνακας 3 Απόκριση Modbus[7]

Οι εντολές Modbus RTU:

Κώδικας Λειτουργίας	Τι κάνει ο Κωδικός λειτουργίας		Τύπος τιμών	Τύπος πρόσβασης
01	Ανάγνωση DO	Read Coil Status	Ψηφιακός	Ανάγνωση
02	Ανάγνωση DI	Read input status	Ψηφιακός	Ανάγνωση
03	Ανάγνωση AO	Read Holding Registers	16bit	Ανάγνωση
04	Ανάγνωση AI	Read input Registers	16bit	Ανάγνωση
05	Εγγραφή ενόςDO	Force Single Coil	Ψηφιακός	Write
06	Εγγραφή ενόςAO	Preset Single Register	16bit	Write
15	ΠολλαπλήDO Εγγραφή	Force Multiple Coils	Ψηφιακός	Write
16	ΠολλαπλήAO Εγγραφή	Preset Multiple Registers	16bit	Write

Πίνακας 4:Κώδικας Λειτουργίας [7]

Tomodule που θα χρησιμοποιηθεί είναι ένας μετατροπέας οπού του P.L.C. το RS485 έχει διάφορα δυναμικού 15V και αυτό θα το μετατρέψει σε 5V.Χρησιμοποιεί ενσωματωμένο τσιπ MAX485 που είναι ένας πομποδέκτης χαμηλής ισχύος και περιορισμένης ταχύτητας που χρησιμοποιείται για την επικοινωνία RS-485. Λειτουργεί σε ένα μόνο τροφοδοτικό + 5V και το ονομαστικό ρεύμα είναι 300 mA.

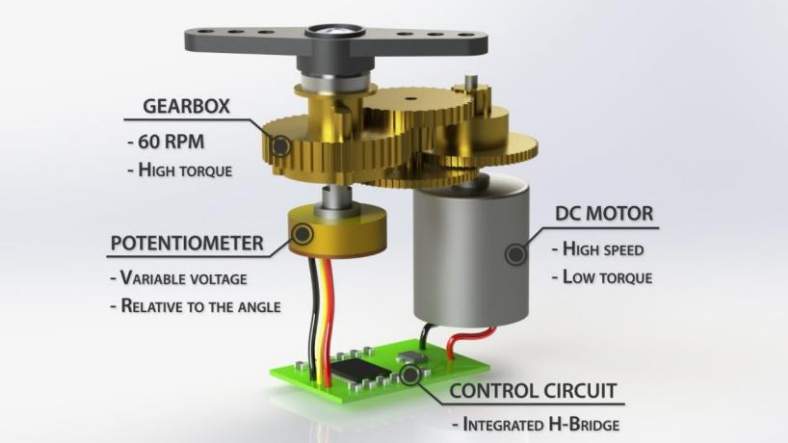


Εικόνα 3.5 MAX485 RS-485 TTL

3.6 Servomotor για το Arduino UNO

Ο συγκεκριμένος σέρβο-κινητήρας δεν είναι ακριβώς σέρβο-κινητήρας παρόλο που έχει αυτή την ονομασία. Είναι κινητήρας συνεχούς ρεύματος με μείωση(εικόνα 3.6 και εικόνα 3.7) οπού του επιτρέπει να κινείται από -180 μοίρες έως 180 μοίρες και τα χαρακτηριστικά του είναι τα εξής:

- Ρεύμα χωρίς φορτίο = 6V: 180 mA
- Ρεύμα με φορτίο = 6V: 800 mA
- Ταχύτητα = 6V: 0.11 sec/60°
- Ροπή =6V: 2.7 kg·cm

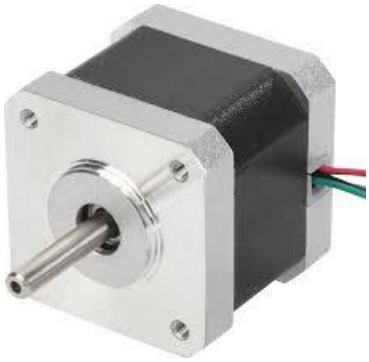


Εικόνα 3.6 Servomotor[8]

Εικόνα 3.7 Grippeservomotor

3.7 Βηματικοί κινητήρες

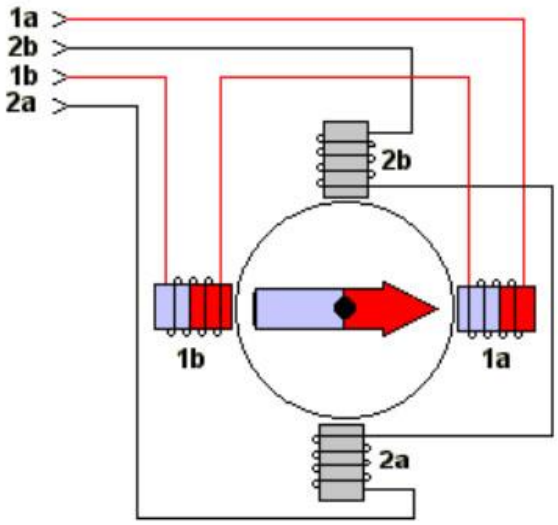
Οι βηματικοί κινητήρες είναι επαγωγικοί κινητήρες οπού έχουν διακριτές θέσεις. Για να πετύχει μια κίνηση χρειάζεται ένα ηλεκτρονικό κύκλωμα οπού θα τον τροφοδοτεί με παλμούς-βήματα. Ο κάθε παλμός είναι ένα βήμα οπού αυτό μπορεί να υποδιαιρεθεί ανάλογα με την εφαρμογή σε σχέση με το πραγματικό βήμα π.χ. στους κινητήρες που χρησιμοποιούνται στην εφαρμογή το πραγματικό βήμα είναι 1.8 μοίρες και χρησιμοποιείται 0.45 μοίρες.[9]



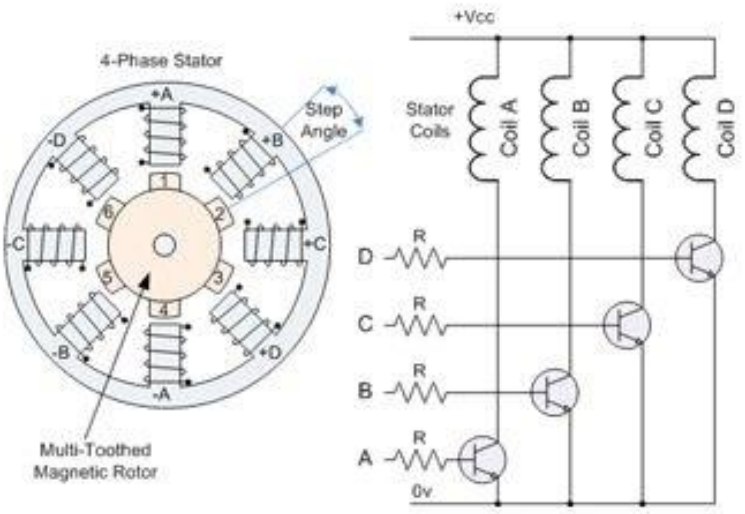
Εικόνα 3.8 Βηματικός κινητήρας Nema 17

Ανάλογα με την κατασκευή του δρομέα οι βηματικοί κινητήρες διακρίνονται στους εξής:

- Βηματικός κινητήρας μόνιμου μαγνήτη
- Βηματικός κινητήρας μεταβλητής μαγνητικής αντίδρασης
- Υβριδικός βηματικός κινητήρας



Εικόνα 3.9 Απεικόνιση ρότορα σε σχέση με τα μαγνητικά πεδία[9]



Εικόνα 3.10 Απεικόνιση ρότορα σε σχέση με τα πηνία[9]

3.8 Χαρακτηρίστηκα PLC

- **Δομή ενός Προγραμματιζόμενου Λογικού Ελεγκτή**

Στην αγορά υπάρχουν σήμερα πάρα πολλά μοντέλα PLC κατασκευασμένα από πολλές εταιρίες. Η επιλογή ενός προγραμματιζόμενου ελεγκτή (τύπος, μέγεθος, κόστος) εξαρτάται από το πλήθος των στοιχείων που δίνουν εντολή σ' αυτόν (είσοδοι) και το πλήθος των στοιχείων που δέχονται εντολή απ' αυτόν (έξοδοι), καθώς και από το πλήθος των λειτουργιών που απαιτείται να κάνει ο αυτοματισμός (μέγεθος προγράμματος, δηλαδή απαιτούμενη μνήμη και δυνατότητες της κεντρικής μονάδας)

Ανεξάρτητα όμως από τύπο και μέγεθος, ένας προγραμματιζόμενος ελεγκτής, συνιστάται από τα εξής απαραίτητα στοιχεία:

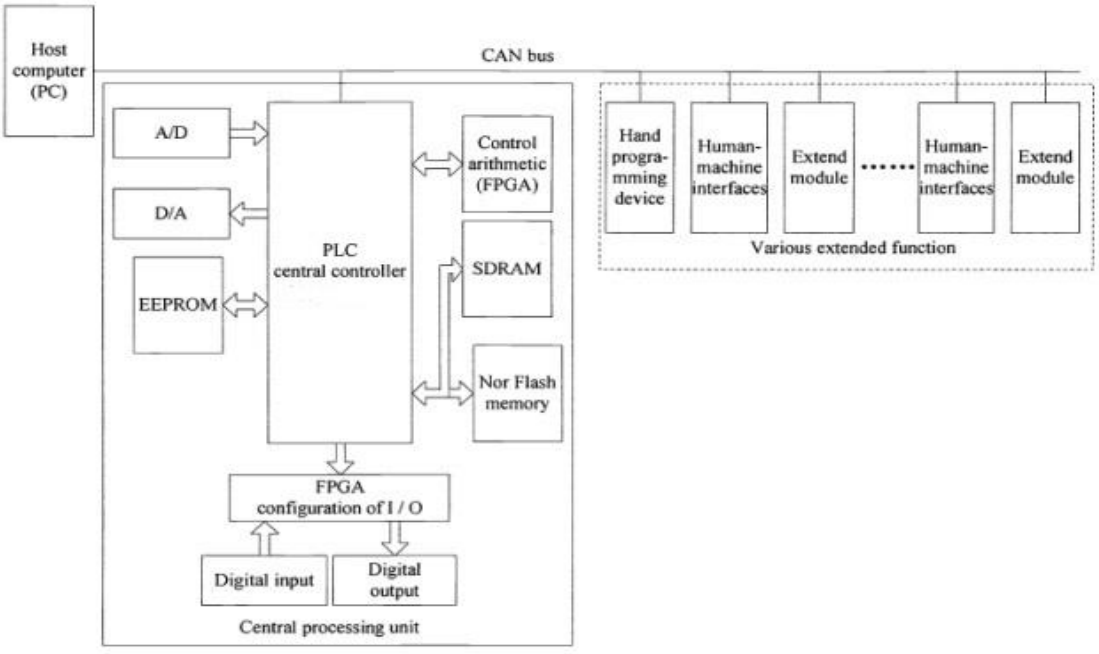
Α. Πλαίσιο για τοποθέτηση των μονάδων

Β. Μονάδα τροφοδοσίας.

Γ. Κεντρική μονάδα επεξεργασίας (CPU) που αποτελεί τον εγκέφαλο του PLC

Δ. Μονάδες εισόδων / εξόδων.

Ε. Συσκευή προγραμματισμού



Εικόνα 3.11Δομή PLC (Προγραμματιζόμενου Λογικού Ελεγκτή)[7]

Στην συνέχεια αναπτύσσονται τα απαραίτητα στοιχεία ενός προγραμματιζόμενου ελεγκτή με λεπτομέρειες

Α. Πλαίσιο τοποθέτησης μονάδων

Όλες οι μονάδες, από τις οποίες αποτελείται ένας προγραμματιζόμενος ελεγκτής, πρέπει να τοποθετηθούν σε κάποιο πλαίσιο. Σ' αυτό είναι ενσωματωμένο το σύστημα αγωγών (BUS), μέσω των οποίων επικοινωνούν οι διάφορες μονάδες μεταξύ τους για την ανταλλαγή πληροφοριών και για την τροφοδοσία τους.

Αν οι θέσεις του κεντρικού πλαισίου . που διατίθεται, δεν επαρκούν για να τοποθετηθούν οι μονάδες εισόδων και εξόδων που απαιτούνται σε μια συγκεκριμένη εφαρμογή, τότε χρησιμοποιούνται περισσότερα πλαίσια επέκτασης για την τοποθέτηση των επιπλέον μονάδων. Κάθε πλαίσιο επέκτασης συνδέεται με το κεντρικό πλαίσιο ή με τα άλλα πλαίσια μέσω ειδικής μονάδας διασύνδεσης και καλωδίου.

Β. Μονάδα τροφοδοσίας

Η μονάδα τροφοδοσίας χρησιμεύει για να δημιουργήσει από την τάση του δικτύου τις απαραίτητες εσωτερικές τάσεις για την τροφοδοσία αποκλειστικά των ηλεκτρονικών εξαρτημάτων, που υπάρχουν μέσα στον προγραμματιζόμενο ελεγκτή (τρανζίστορ, ολοκληρωμένα κ.λπ.). Οι τυπικές εσωτερικές τάσεις των ελεγκτών είναι συνήθως: DC 5V, DC 9V, DC 24V.

Γ. Κεντρική μονάδα επεξεργασίας (CPU)

Είναι η βασική μονάδα του ελεγκτή, η οποία είναι υπεύθυνη για τη λειτουργία του αυτοματισμού. Η κεντρική μονάδα επεξεργασίας είναι στην ουσία ένας μικροϋπολογιστής και διακρίνουμε σ' αυτήν όλα τα κύρια μέρη ενός μικροϋπολογιστή, δηλαδή τον

μικροεπεξεργαστή και τη μνήμη. Ο μικροεπεξεργαστής είναι ο αυτός που εκτελεί όλες τις λειτουργίες του προγραμματιζόμενου ελεγκτή.

Δ. Μονάδες εισόδων / εξόδων

Οι μονάδες των εισόδων και των εξόδων αποτελούν τις μονάδες επικοινωνίας της κεντρικής μονάδας με τον έξω κόσμο, δηλ. με τους αισθητήρες, τους διακόπτες, τα μπουτόν κ.α., που δίνουν τις πληροφορίες (εντολές)στη κεντρική μονάδα, καθώς και με τα ρελέ ισχύος των κινητήρων, ηλεκτρομαγνητικές βαλβίδες, ενδεικτικές λυχνίες και γενικά τους αποδέκτες που εκτελούν τις εντολές της κεντρικής μονάδας. Η κεντρική μονάδα μπορεί να δεχτεί ψηφιακά σήματα εισόδου και εξόδου χαμηλής τάσης και πολύ μικρού ρεύματος. Η τάση που δέχεται είναι συνήθως 0 V για το λογικό “0” και 5 V για το λογικό “1”. Το ρεύμα εισόδου καθώς και το ρεύμα εξόδου δεν μπορεί να ξεπεράσει τα λίγα mA. Οι μονάδες εισόδων και εξόδων αναλαμβάνουν να προσαρμόσουν τα σήματα εισόδου και εξόδου, που έχουμε στον αυτοματισμό, σε σήματα που μπορεί να δεχτεί η κεντρική μονάδα, τόσο από άποψη τάσεων όσο και από άποψη ρευμάτων. Η προσαρμογή αυτή γίνεται με χρήση ηλεκτρονικών στοιχείων ισχύος, είτε με τη χρήση των κατάλληλων μικρό-ρελέ.
Κάθε σύστημα PLC καταλήγει πάντα σε ακροδέκτες (κλέμες). Οι ακροδέκτες αυτοί ανήκουν στις μονάδες εισόδων και εξόδων του. Στους ακροδέκτες εισόδων καταλήγουν οι αγωγοί που έρχονται από αισθητήρες η τερματικούς διακόπτες, πιεσοστάτες, διακόπτες μπουτόνς, κτλ. Στους ακροδέκτες εξόδων καταλήγουν οι αγωγοί που τροφοδοτούν πηνία ρελέ ισχύος, ηλεκτρομαγνητικές βαλβίδες, λυχνίες ένδειξης και λοιπούς αποδέκτες. Στους διάφορους τύπους των PLC που υπάρχουν, οι μονάδες εισόδων και εξόδων αντιμετωπίζονται με διαφορετικό τρόπο.

Καθολικά ισχύουν τα παρακάτω:

1) Μια μονάδα εισόδων ή εξόδων μπορεί να λειτουργεί με συνεχή τάση ή με εναλλασσόμενη τάση. Τυπικές τάσεις λειτουργίας είναι: DC 24V, 48V, 60V & AC 24V, 48V, 115V, 230V, με συνηθέστερες τις DC 24V, AC 115V & AC 230V.

2)Η τάση αυτή συνήθως δεν παρέχεται από τη μονάδα τροφοδοσίας του PLC .Πρέπει να τη δημιουργήσουμε εμείς με άλλη τροφοδοτική μονάδα.

3) Τα κυκλώματα και οι τάσεις των εισόδων είναι τελείως ανεξάρτητα από τα αντίστοιχα κυκλώματα των εξόδων. Επομένως η τάση για τις εισόδους μπορεί να είναι διαφορετική από την τάση για τις εξόδους. Αν τώρα αυτές οι τάσεις είναι ίδιες μπορεί να χρησιμοποιηθεί το ίδιο τροφοδοτικό (για συνεχείς τάσεις), ή μετασχηματιστής χειρισμού (για AC τάσεις) για τις εισόδους και για τις εξόδους.

4) Η τάση εισόδων (δηλ. η τάση που φτάνει σε μια είσοδο, όταν ενεργοποιηθεί ο αντίστοιχος αισθητήρας) συνήθως διαχωρίζεται γαλβανικά από το υπόλοιπο εσωτερικό κύκλωμα του PLC. Τα ίδια ισχύουν και για τις εξόδους. Αν σε κάποιες μονάδες εισόδων ή εξόδων δεν έχουμε γαλβανική απομόνωση πρέπει να προσέξουμε ιδιαίτερα το θέμα των γειώσεων.

5) Οι μονάδες εισόδων μεταφέρουν τις τιμές των εισόδων και οι μονάδες εξόδων μεταφέρουν την κατάσταση των εξόδων. Οι είσοδοι και οι έξοδοι αποτελούν βασικό στοιχείο επιλογής στο PLC.

	Τάσεις - Ρεύματα			
	Είσοδος		**Έξοδος**	
	Τάση	**Ρεύμα**	**Τάση**	**Ρεύμα**
Αναλογική	0-10V	4-20mA	0-10V	4-20mV
Ψηφιακή	24V	220V	Ρελέ	5 mA-0.5A

Πίνακας 5: Πίνακας Τιμών τάσεων και ρευμάτων, αναλογικών και ψηφιακών εισόδων/εξόδων

4. Υλοποίηση ρομποτικού βραχίονα

4.1 Σχεδίαση βραχίονα

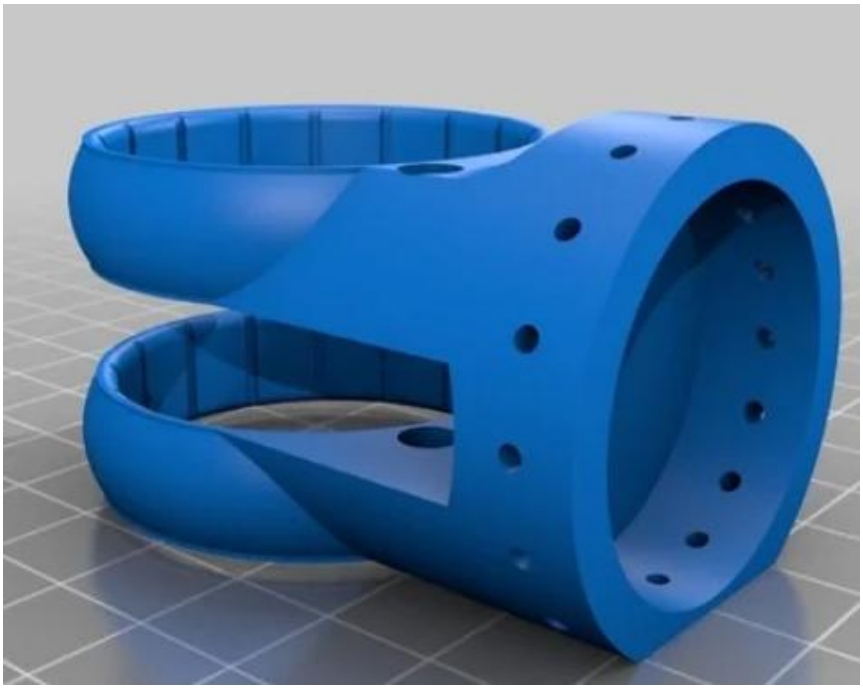
Το σχέδιο του ρομποτικού βραχίονα είναι σχέδια ανοιχτού κώδικα όπου υπάρχει στην ιστοσελίδα του thingverse. Τα σχέδια αφορούν 3DPrinter ώστε κάθε κομμάτι του βραχίονα ακόμη και τα γρανάζια από τους πλανητικούς μειωτήρες είναι εκτυπωμένα.

Το περιβάλλον που χρησιμοποιήθηκε για την εκτύπωση είναι IdeaMaker 3.5.2, το μοντέλο του εκτυπωτή είναι Printer Raise3D N2 και το υλικό της εκτύπωσης είναι PLA 1.75mm.

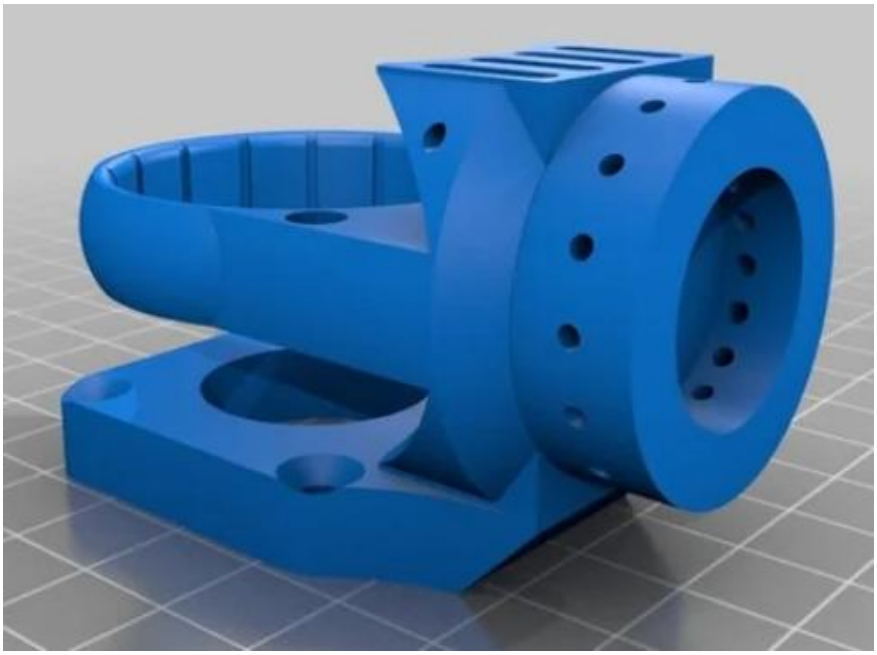
Οι παράμετροι που χρειάστηκε να τροποποιηθούν είναι:

- LayerHeight=0.1500mm το πάχος της κάθεστρώσης.
- Shell=2 να δημιουργεί δυο στρώσεις παράλληλα σε κάθε κατακόρυφη στρώση.
- Infill Density‘πυκνότητα πλήρωσης =10%
- InfillSpeed=90% η ταχύτητα της κεφαλής
- PrimaryExtruder=215C θερμοκρασία εξόδου κεφαλής
- HeatedBedTemperature =60ºCθερμοκρασίαβάσης

Παρακάτω παρουσιάζονται τα αρχεία πριν την εκτύπωση:

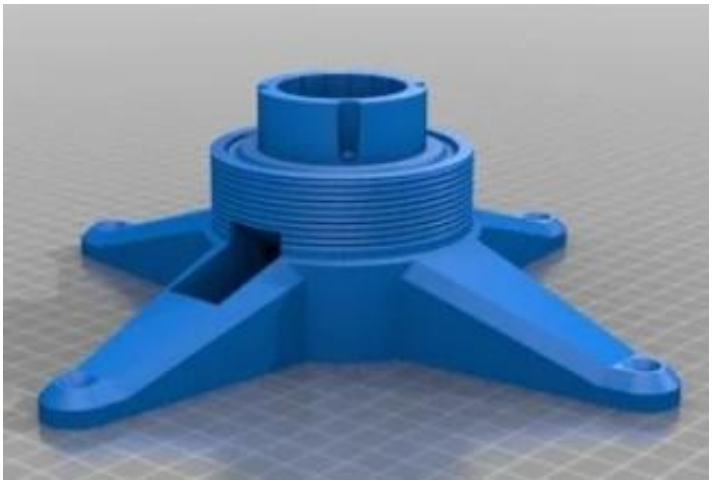


Εικόνα 4.1 Βάση συνδέσμου μειωτήρα [11]

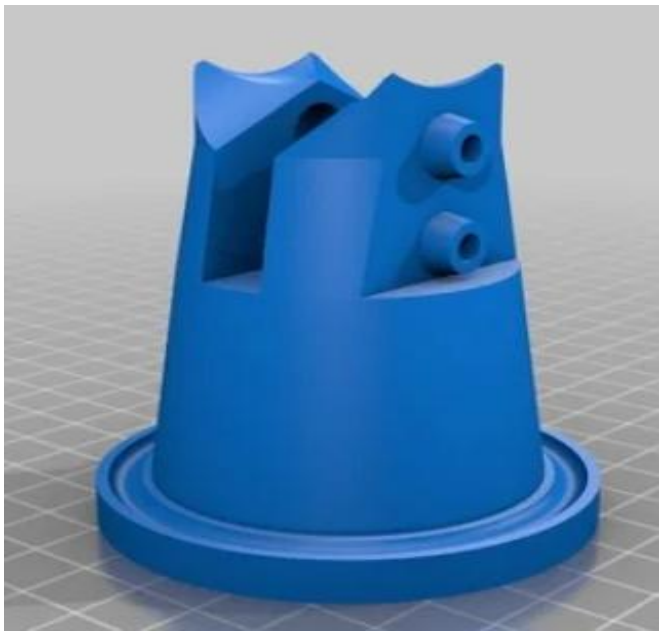


Εικόνα 4.2 Βάση τρίτου κινητήρα- σύνδεσμος μειωτήρα [11]

Εικόνα 4.3 Βάση τρίτου μειωτήρα τέταρτου δαχτυλιδιού [11]

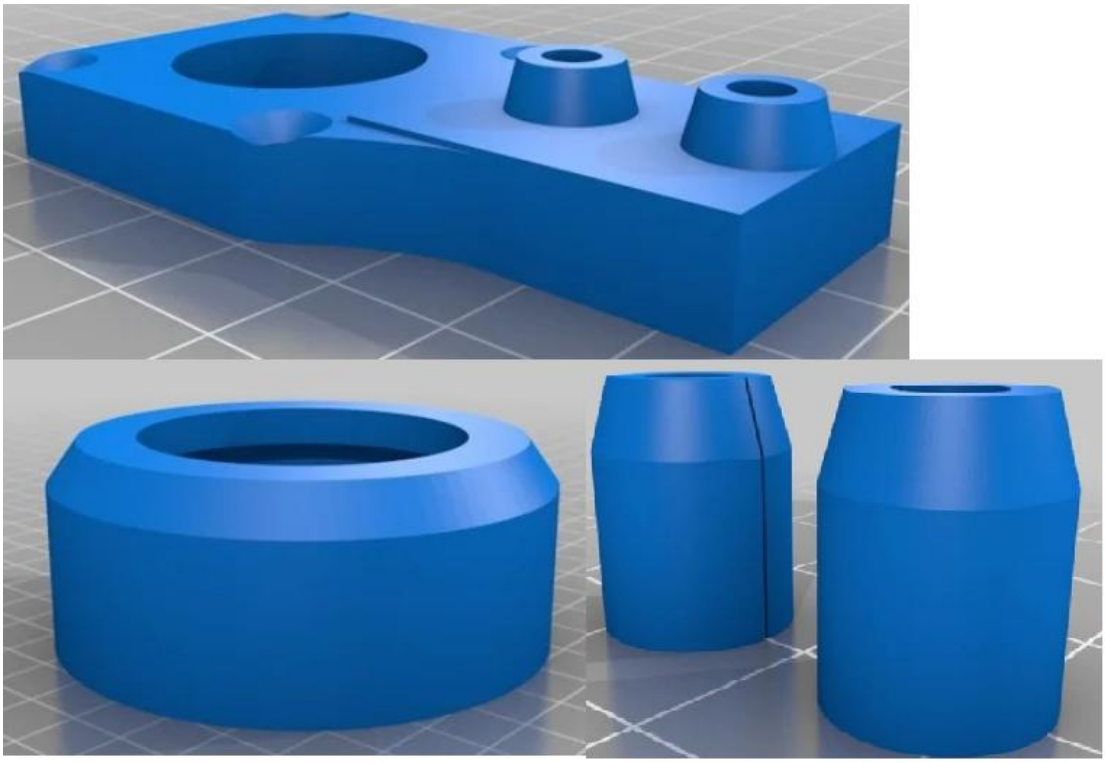


Εικόνα 4.4 Βάση βραχίονα και βάση πρώτου μειωτήρα [11]

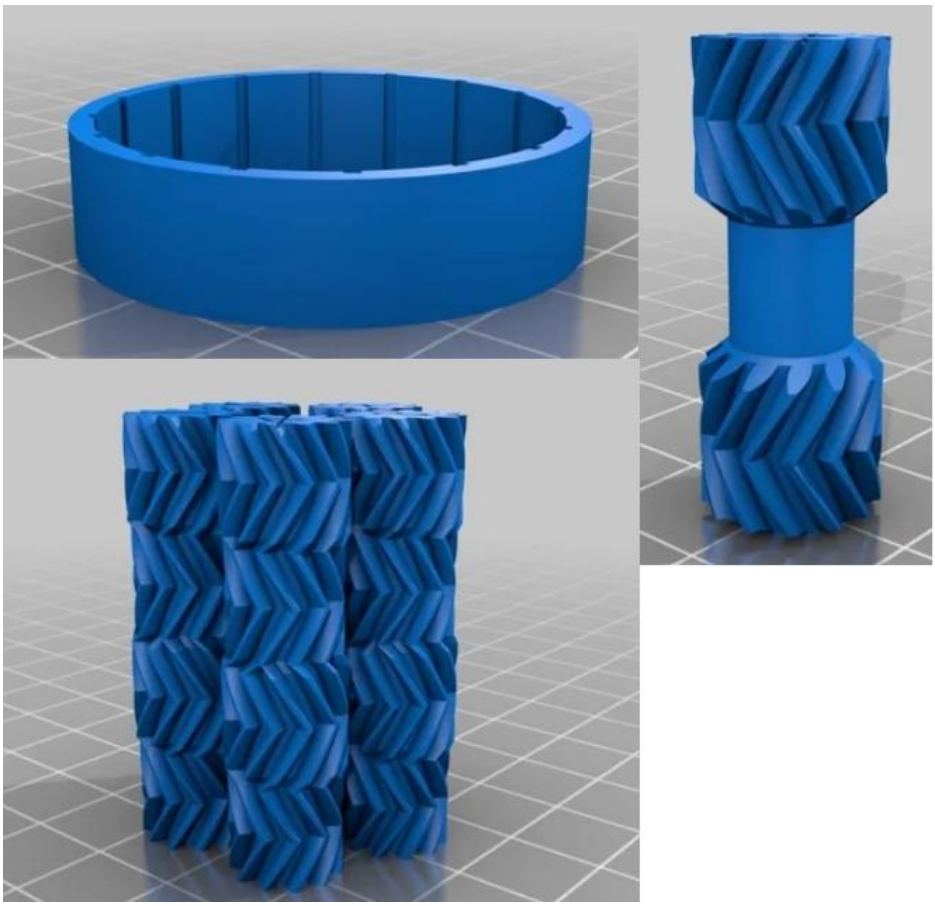


Εικόνα 4.5 Βάση δεύτερης άρθρωσης και βάση πρώτου μειωτήρα[11]

Στην βάση της εικόνας 4.5 τοποθετούνται και σφαίρες ρουλεμάν για την μείωση τριβών.

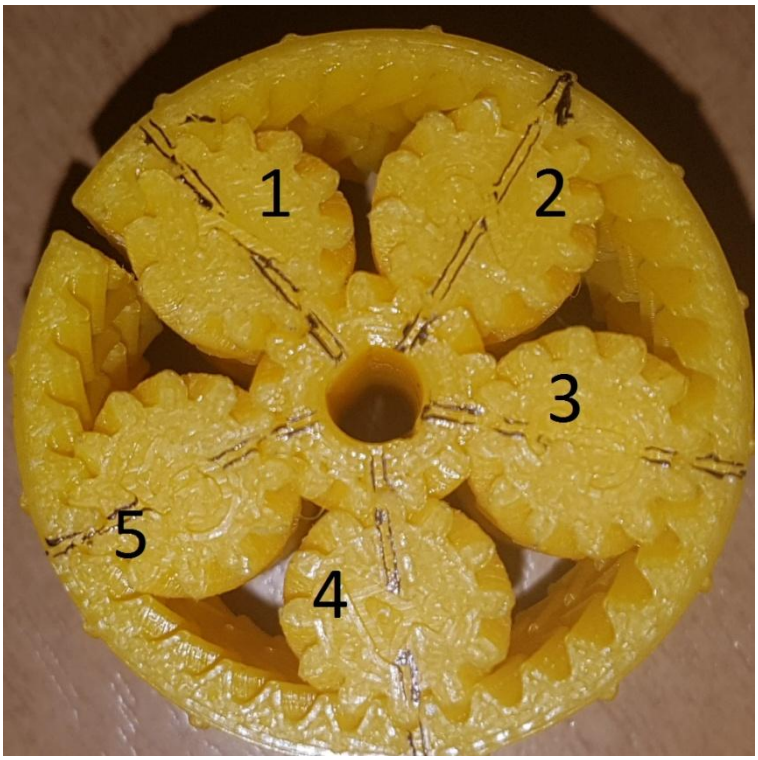


Εικόνα 4.6 Βάση δευτέρου κινητήρα, παξιμάδι δεύτερης άρθρωσης και αποστάτες[11]

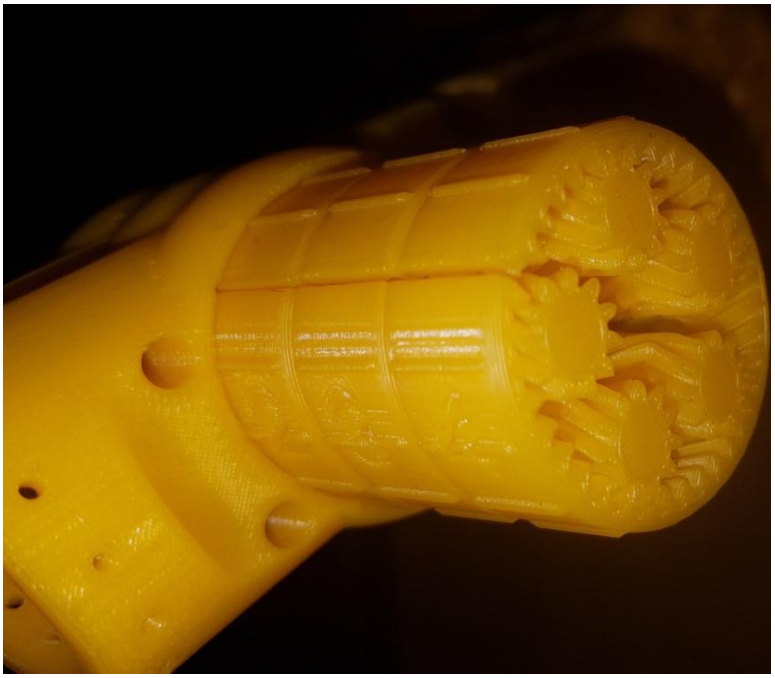


Εικόνα 4.7 Εξαρτήματα πλανητικού μειωτήρα[11]

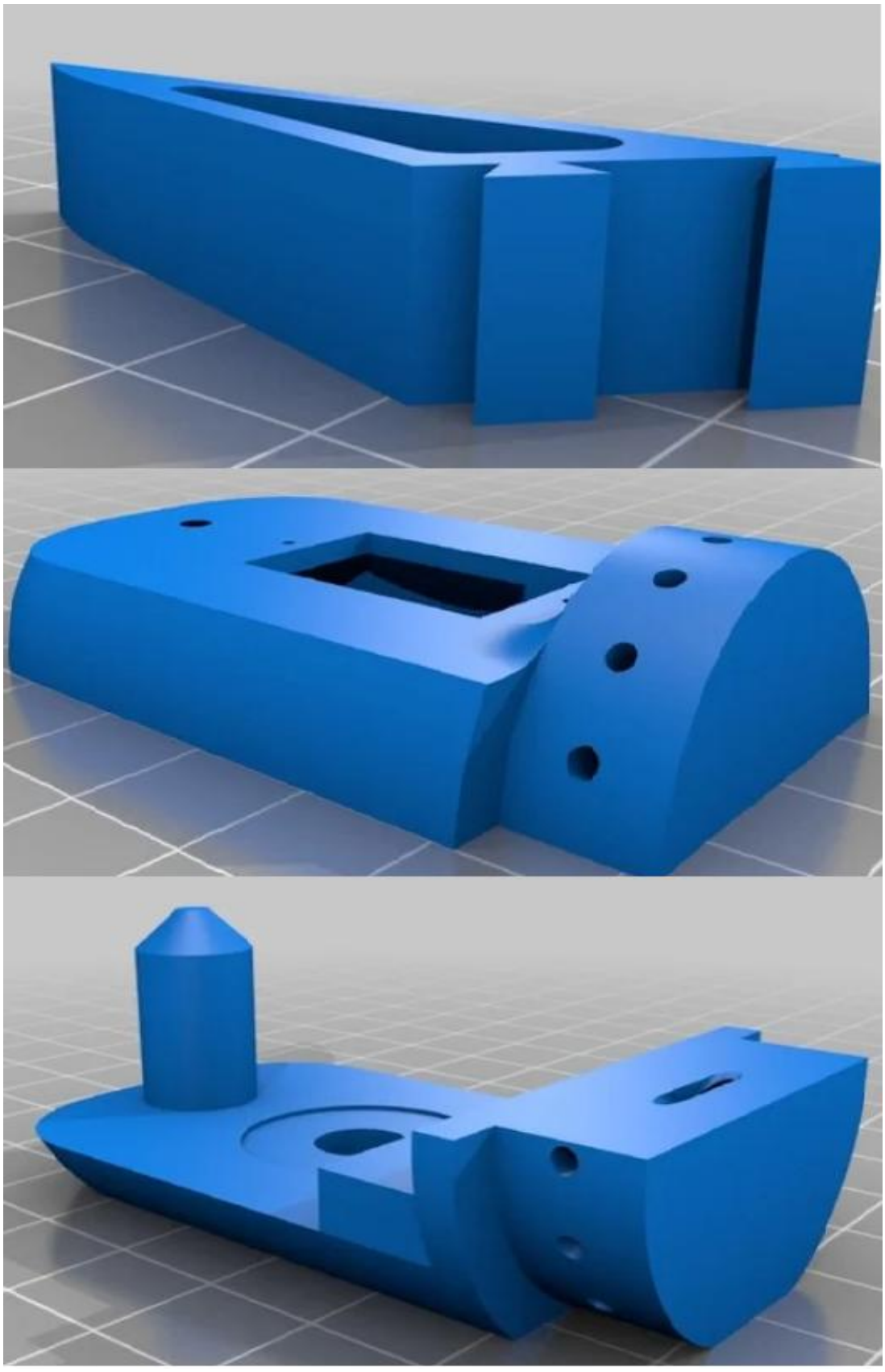
Ο πλανητικός μειωτήρας έχει τέσσερα δαχτυλίδια με διαφορετικό αριθμό αναγραφόμενο επάνω(εικόνα 4.9), το κάθε δαχτυλίδι έχει διαφορετικό αριθμό γραναζιών έτσι ώστε να επιτυγχάνεται η μείωση 66:1 και εγκοπές έτσι ώστε στην συναρμολόγηση να είναι όλα τα γρανάζια στην ιδία απόσταση μεταξύ τους. Το ίδιο ισχύει για τα πέντε γρανάζια που εκτυπώνονται μαζί έχουν συγκεκριμένο αριθμό που τοποθετούνται με την συγκεκριμένη σειρά(εικόνα 4.8). Ο άξονας που έχει επάνω του δυο γρανάζια είναι εκεί που τοποθετείται ο άξονας του κινητήρα.



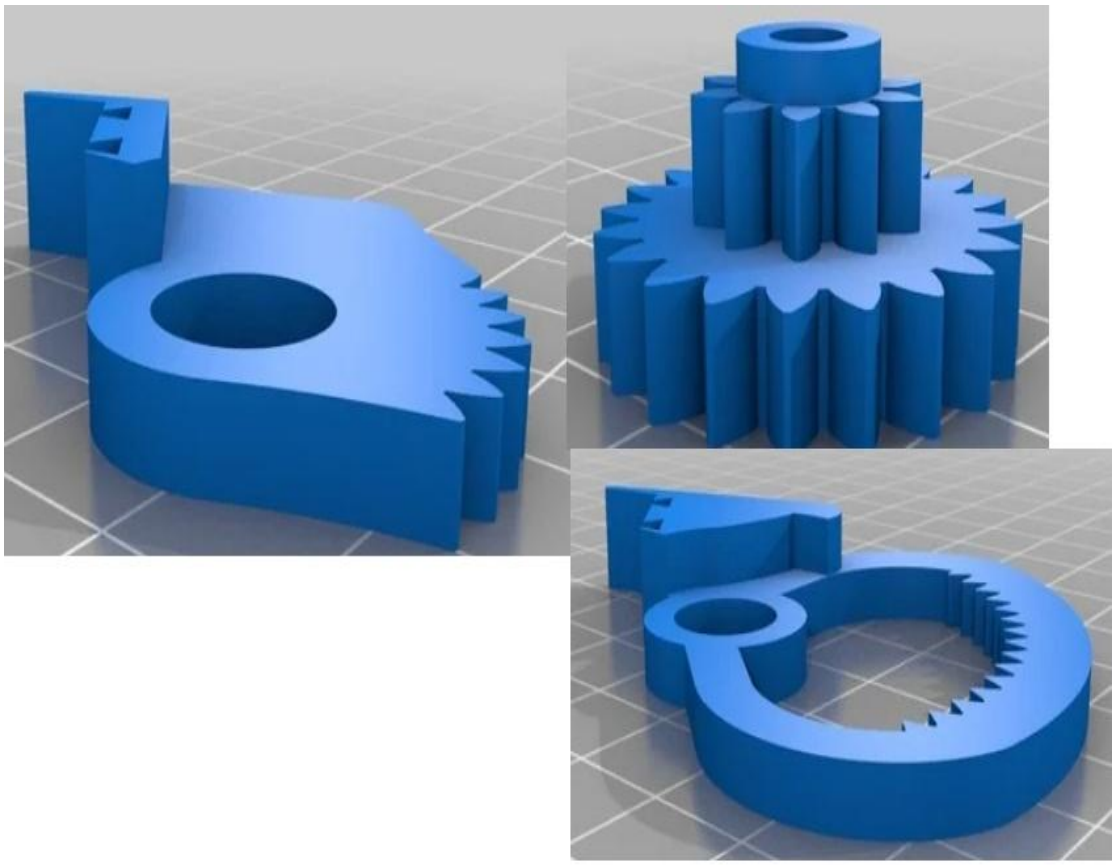
Εικόνα 4.8 Ευθυγράμμιση γραναζιών



Εικόνα 4.9 Τοποθέτηση μειωτήρα 66:1 στον σύνδεσμο

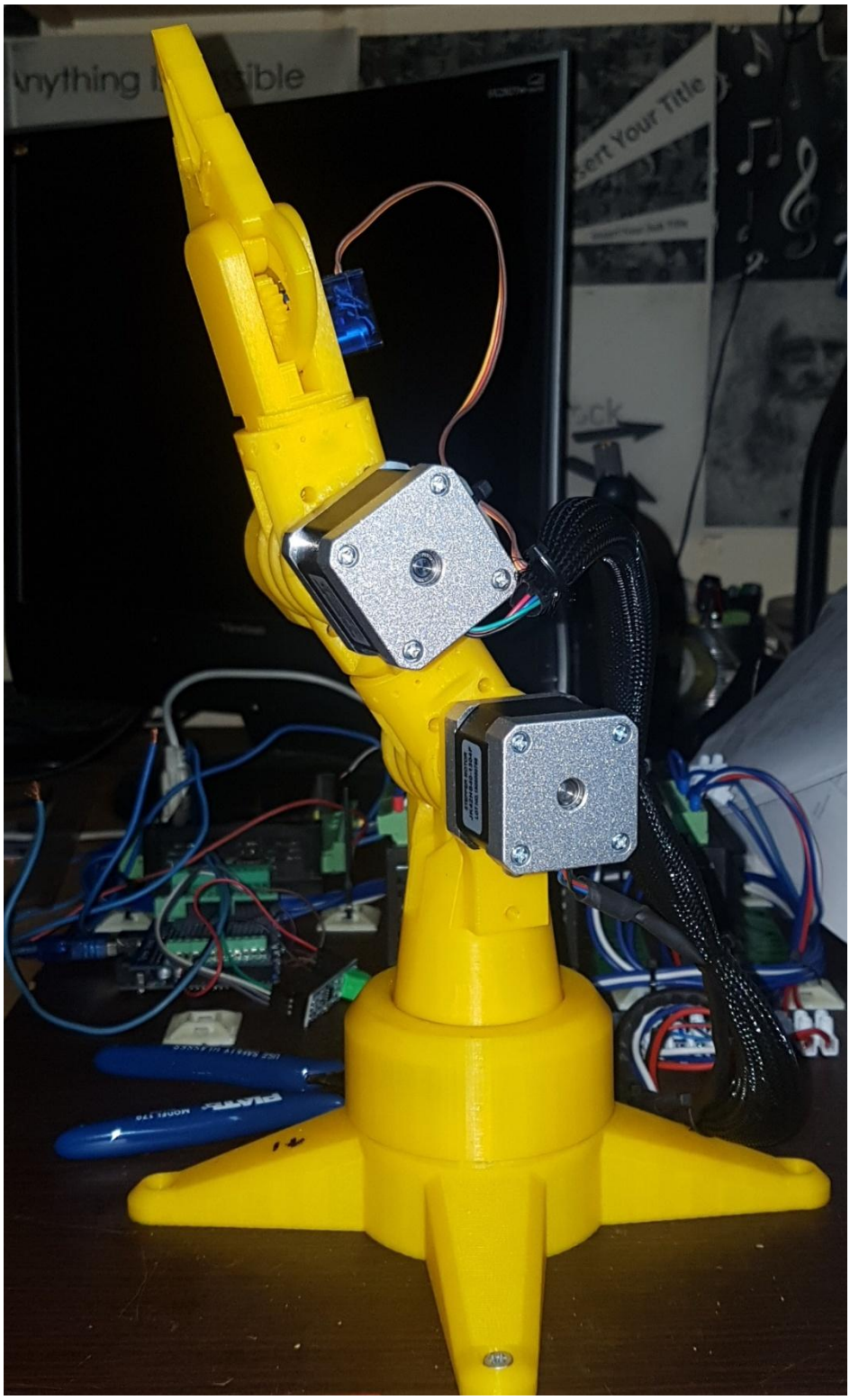


Εικόνα 4.10 Βάση και σύνδεσμος gripper[11]



Εικόνα 4.11 Γρανάζια gripper[11]

Εικόνα 4.12 Εκτυπωμένα εξαρτήματα gripper

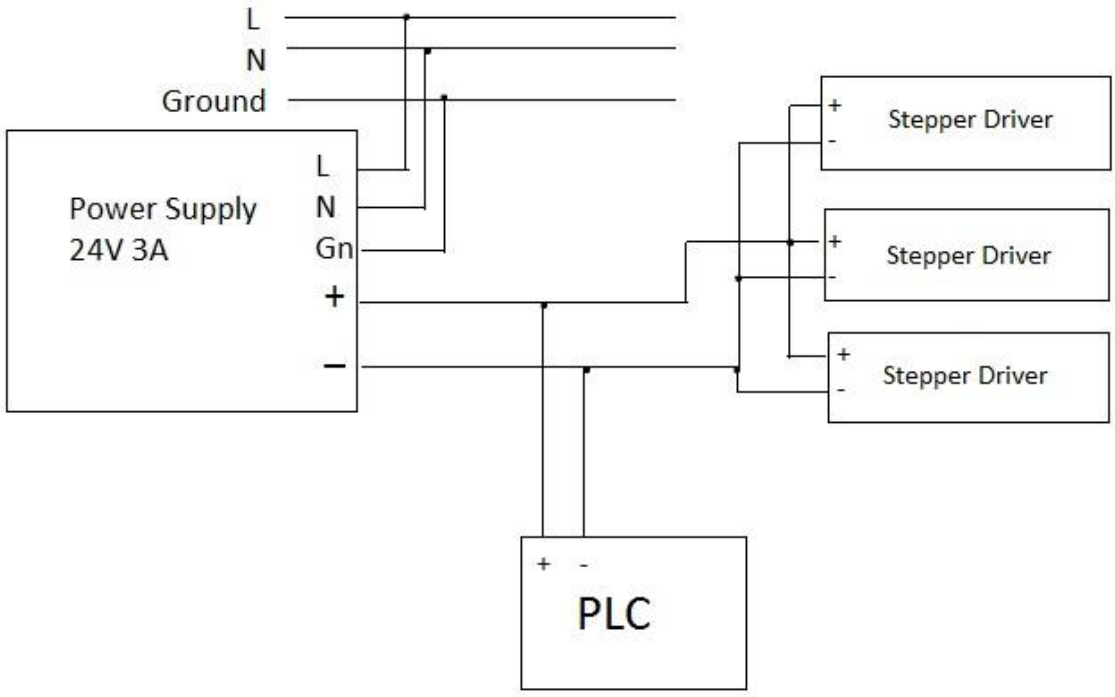


Εικόνα 4.13 Ο συναρμολογημένος βραχίονας 3DOF

4.2 Υλοποίηση ηλεκτρικών κυκλωμάτων

Τροφοδοσία

Το τροφοδοτικό έχει είσοδο 220VAC 3A όπου μετά την ανόρθωση δίνει 24VDC. Τροφοδοτεί το PLC και τους οδηγούς από τους βηματικούς κινητήρες έκτος το Arduino. Το Arduino έχει ξεχωριστή τροφοδοσία μέσω USB δηλαδή 5V 200mA.



Εικόνα 4.14 Ηλεκτρικό σχέδιο τροφοδοσίας 24V

Ηλεκτρολογική σύνδεση PLC

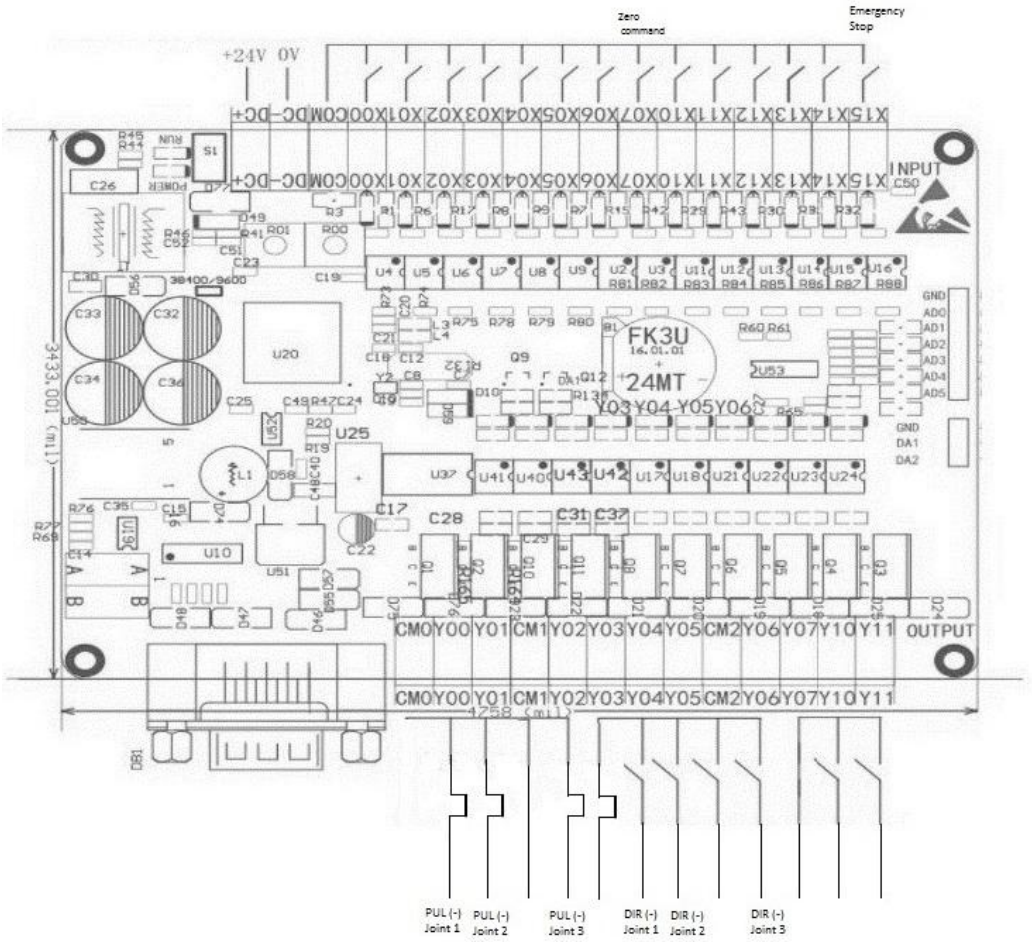
Το PLCπου χρησιμοποιήθηκε είναι Κινέζικης κατασκευής με πρωτόκολλο ίδιο με την σειρά της MitsubishiFX3U-MR. Στα pinDC+ και DC- στο PLC θα συνδεθεί η τροφοδοσία του. Τα X### συμβολίζουν τις εισόδους του PLC,σε αυτές δεν θα συνδεθεί τίποτα διότι όποια ενεργεία θα προέρχεται από την οθόνη του Arduino. Τα Y### συμβολίζουν τις εξόδους του PLC,το συγκεκριμένο PLC έχει ενσωματωμένα ρελέ. Επειδή θα χρειαστούν τρεις HighSpeedPulse έξοδοι για τους βηματικούς οδηγούς θα αφαιρεθούν τα ρελέ από το

Y000,Y001,Y002(στο σχέδιο συμβολίζονται με παλμό). Οι συγκεκριμένες έξοδοι επιλεχτήκαν διότι μόνο αυτές βάση hardwareυποστηρίζουν HighSpeedPulseOutputs.

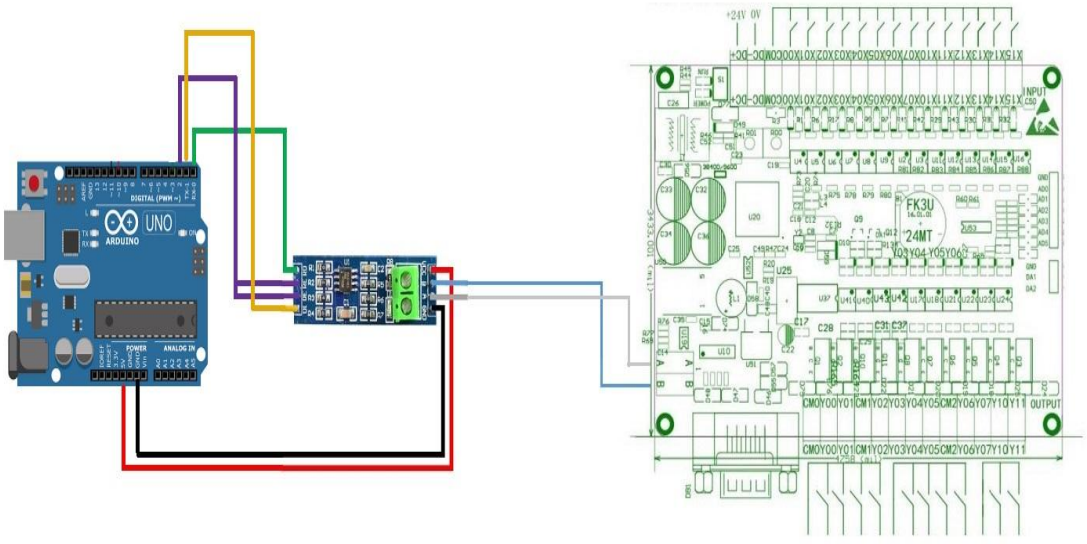
Στο Com από την μεριά των εξόδων είναι ο κοινός των υπολοίπων ρελέ που θα συνδεθεί το πλην. Αυτό γίνεται γιατί τις εξόδους Y004,Y005,Y006(στο σχέδιο συμβολίζονται με επαφή) θα χρησιμοποιηθούν για την κατεύθυνση των κινητήρων.

Στα A και B θα συνδεθούν τα A και B τουMAX485 RS485 TTL shield. Έπειτα από τοMAX485 RS485 TTL shield το RO θα συνδεθεί στο pin 0 του Arduino,το DE και RE θα συνδεθούν μαζί στο pin 2 του Arduino ώστε να ενεργοποιήσουν την επικοινωνία και το DI στο pin1 του Arduino όπου έτσι θα γίνεται η επικοινωνία με τον Arduino μέσω Modbus (Εικόνα 4.16).Τα pins από Ad0-Ad6 είναι αναλογικές είσοδοι και έξοδοι άλλα δεν θα χρησιμοποιηθούν στην εφαρμογή.

Η επικοινωνία του PLC με τον υπολογιστή γίνεται μέσω της θύρας DB9(RS232) 38400baurate 7biteven 1bit.Στο PLC υπάρχει μια μπαταριά για να διατηρεί τα δεδομένα από ορισμένους καταχωρητές από το D200-D7999 όπου και θα είναι χρήση αυτών ώστε να διατηρεί σταθερή τη θέση του βραχίονα σε περίπτωση διακοπής του ρεύματος. Επίσης, είναι χρήσιμη στην λειτουργιά ενός RealTimeClock και ο διακόπτης Run-Stop βάζει στην ανάλογη λειτουργιά το PLC.



Εικόνα 4.15 Ηλεκτρικό σχέδιο PLC



Εικόνα 4.16 Ηλεκτρικό σχέδιο σύνδεσης Arduino-MAX485 RS485 TTLshield-PLC

Ηλεκτρολογική σύνδεση PLC με StepperDrivers

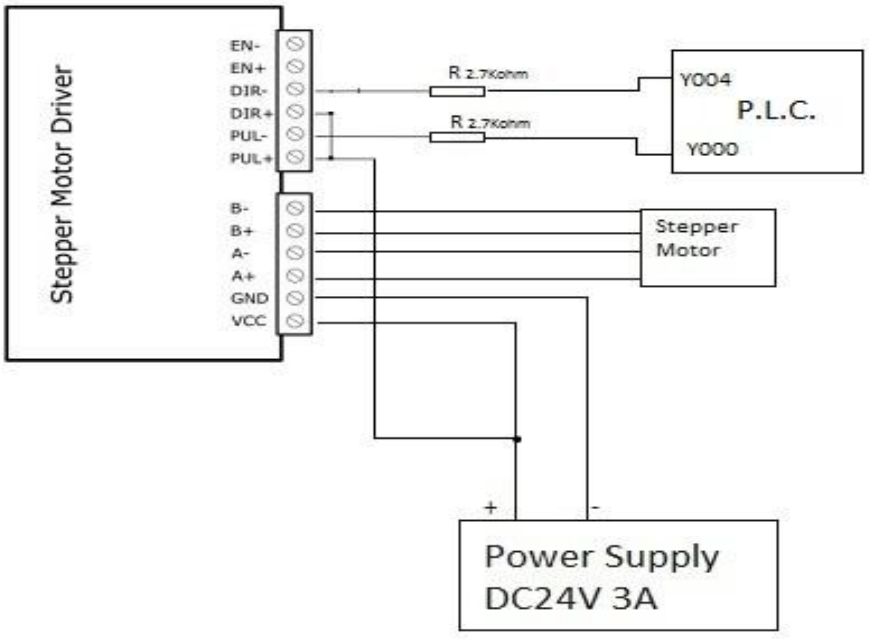
Το κύκλωμα ισχύος είναι η τροφοδοσία 24V και τα pins A+ με A- και B+ με B-.Στο κύκλωμα ελέγχου θα συνδεθεί με common-anodeconnectionκαι όλα τα συν είναι συνδεμένα μαζί με το συν της τροφοδοσίας ώστε ο έλεγχος θα γίνεται από τα αρνητικά pins. Αυτό συμβαίνει διότι το συγκεκριμένο PLC παράγει παλμούς με αρνητικό πρόσημο.

Σε κάθε σήμα που προέρχεται από το PLC προς τους οδηγούς θα συνδέεται στην σειρά μια αντίσταση 2.7Kohm επειδή η τάση από τις εξόδους είναι 24V και δεν πρέπει να διαρρέει ρεύμα μεγαλύτερο των 16mA αλλιώς θα χαλάσει το octo-coupler του οδηγού.

Στο PUL- θα συνδεθούν οι HighSpeedPulse έξοδοι Y000,Y001,Y002 αντίστοιχα σε κάθε οδηγό.

Στο DIR- θα συνδεθούν οι έξοδοι Y004,Y005,Y006 για την κατεύθυνση τον κινητήρων. Οι έξοδοι Y000-Y004 αφορούν την πρώτη άρθρωση, οι Y001-Y005 την δεύτερη και οι Y002-Y006 για την τρίτη.

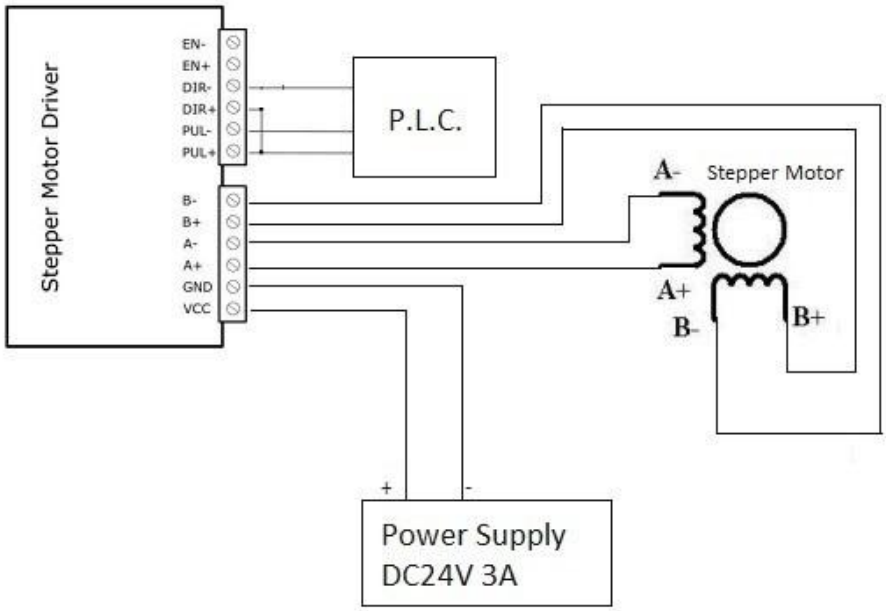
Το EN δεν θα χρησιμοποιηθεί στην εφαρμογή διότι όταν ενεργοποιείται απελευθερώνει το ηλεκτρικό φρένο των κινητήρων.



Εικόνα 4.17 Ηλεκτρικό σχέδιο σύνδεσης πρώτου κινητήρα PLC μεStepperDriver

Ηλεκτρολογική σύνδεση Stepper DriversμεStepper motors και ρύθμιση StepperDrivers

Τα χρώματα από τα πηνία των κινητήρων είναι κόκκινο-μπλε-μαύρο-πράσινο. Το κόκκινο και το μπλε είναι οι δυο άκρες από το πρώτο πηνίο όπου θα συνδεθούν αυθαίρετα στο A+ και A-(δεν έχει σημασία η πολικότητα). Ενώ το μαύρο και το πράσινο είναι το δεύτερο πηνίο που θα συνδεθούν αυθαίρετα στο B+ και B-(οποιοδήποτε πηνίο μπορεί να πάει σε οποιαδήποτε είσοδο του οδηγού).



Εικόνα 4.18 Ηλεκτρικό σχέδιο σύνδεσης κινητήρα με οδηγό

Στον κάθε Stepperdriver υπάρχουν έξι switch όπου τα τρία από αυτά ρυθμίζει την αναλογία βήματος / μοίρας και τα άλλα τρία το μέγιστο ονομαστικό ρεύμα που μπορεί να περάσει από τον κινητήρα.

Εικόνα 4.19 Διακοπές οδηγού

Οι παρακάτω πίνακες δείχνουν τις λειτουργιές:

Micro Step	Pulse/Rev	S1	S2	S3
NC	NC	ON	ON	ON
1	200	ON	ON	OFF
2/A	400	ON	OFF	ON
2/B	400	OFF	ON	ON
4	800	ON	OFF	OFF
8	1600	OFF	ON	OFF
16	3200	OFF	OFF	ON
32	6400	OFF	OFF	OFF

Πίνακας αναλογίας βήματος / μοίρας

Στο παραπάνω πίνακα φαίνονται οι διάφορες επιλογές. Στην εφαρμογή θα επιλεχτεί το microstep 4 για όλους τους οδηγούς. Δηλαδή οι κινητήρες με επιλογή το 1 microstep βάση hardware αντίστοιχα 1 βήμα για 1.8 μοίρες, στην επιλογή microstep 4 ο υπολογισμός γίνεται 1.8 / 4=0.45 μοίρες. Οπότε αυτό σημαίνει για κάθε παλμό ο άξονας του κινητήρα θα περιστρέφει 0.45 μοίρες.

Current(A)	S4	S5	S6
0.5	ON	ON	ON

1.0	ON	OFF	ON
1.5	ON	ON	OFF
2.0	ON	OFF	OFF
2.5	OFF	ON	ON
2.8	OFF	OFF	ON
3.0	OFF	ON	OFF
3.5	OFF	OFF	OFF

Πίνακας 6: Πίνακας επιλογής ονομαστικού ρεύματος

Οι κινητήρες της εφαρμογής έχουν ονομαστικό ρεύμα 1.7A για αυτό θα επιλεγεί η επιλογή 1.5A για την ασφάλεια των κινητήρων.

4.3 Πρόγραμμα P.L.C.

Το προγραμματιστικό περιβάλλον που θα αναπτυχτεί το πρόγραμμα είναι το GxWorks 2 της Mitsubishi. Ο κώδικας είναι σε γλωσσά ladder η όποια είναι η πιο διαδεδομένη γλωσσά για PLC. Η ladder κυρίως έχει ως εντολές ηλεκτρολογικές επαφές ώστε να απεικονιστεί ένας ηλεκτρολογικός πίνακας.

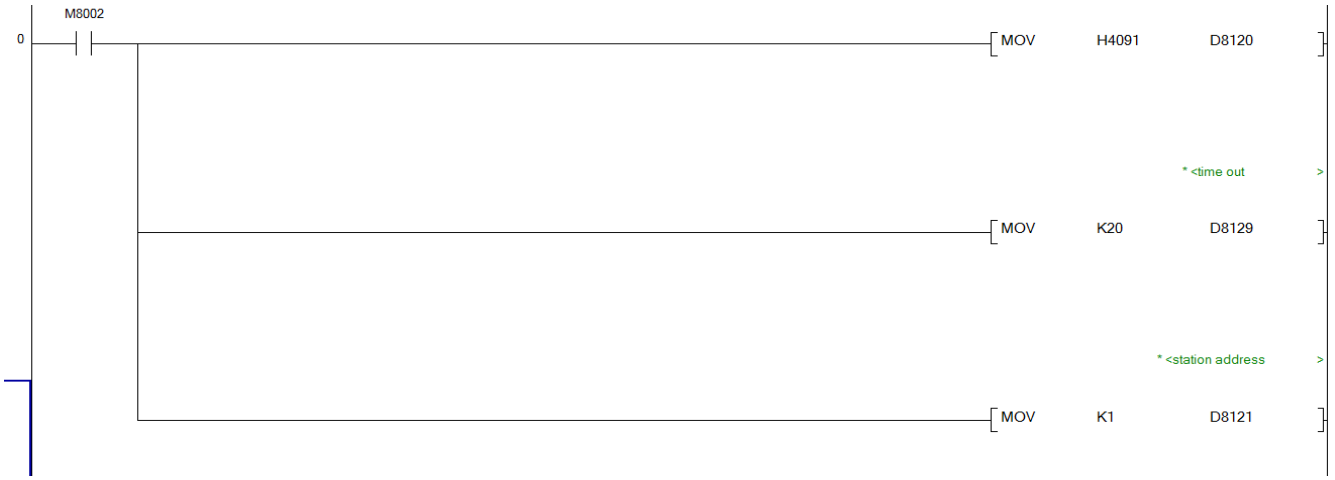
Παρακάτω παρατίθεται η επεξήγηση του κώδικα:

Modbussetup

Η ειδική μνήμη M8002 θα δίνει ένα παλμό κάθε φορά που η CPU είναι σε κατάσταση Stop και μεταβαίνει σε Run ώστε να στήνει τις επιλογές για το Modbus. Στην πρώτη γραμμή με την εντολή Dmov θα μεταφερθεί ο δεκαεξαδικός αριθμός H4091 στον ειδικό καταχωρητής D8120 ώστε να ορίσει τις επιθυμητές επιλογές. Ο δεκαεξαδικός αριθμός H4091 προκύπτει από τον πίνακα στην εικόνα 4.2.1 όπου η ρυθμίσεις είναι 1 biton,8bitlength, noneparity, baudrate 19200bps.

Στην δεύτερη γραμμή ορίζεται ο χρόνοςtimeout 20ms στον ειδικό καταχωρητή D8129.

Στην τρίτη γραμμή ορίζεται ο αριθμός 1 του σταθμού που θα είναι το PLC.



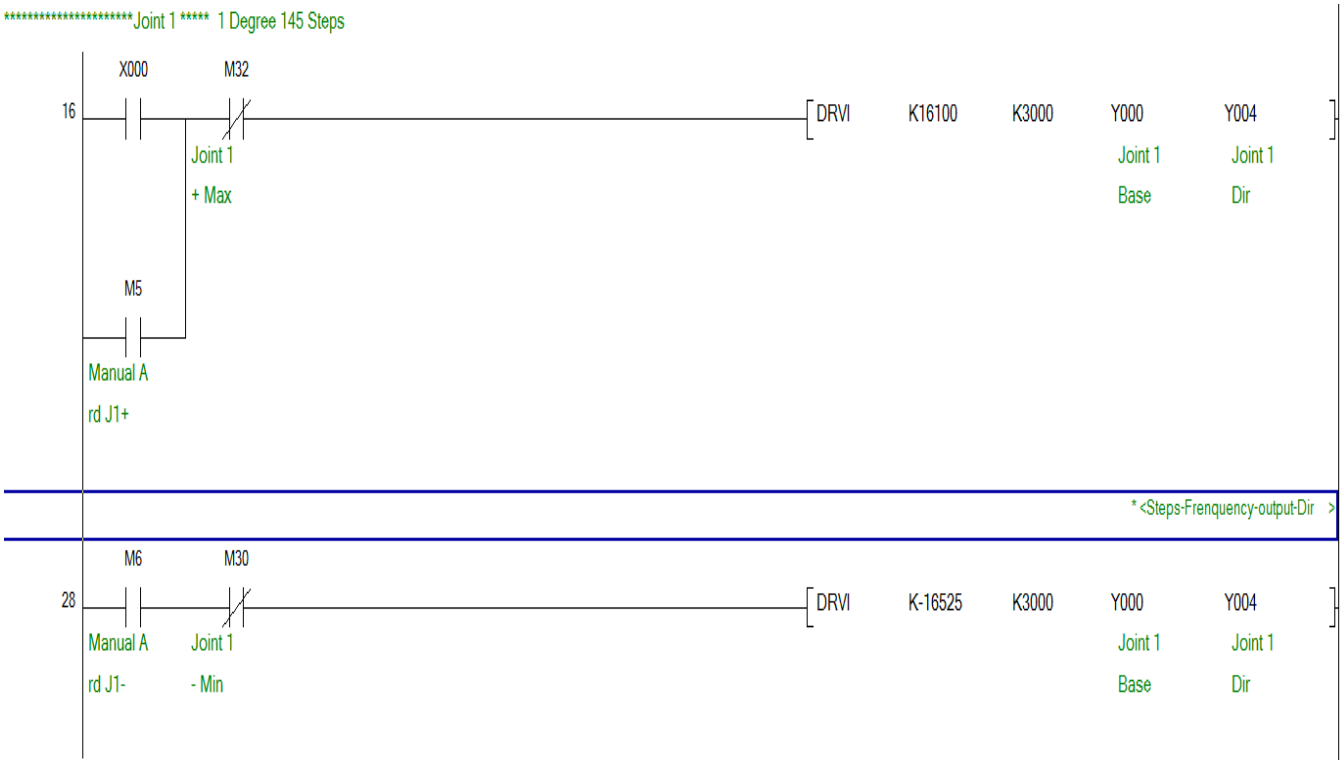
Εικόνα 4.20Modbussetup

Bit No.	Name	Description	
		0 (bit = OFF)	1 (bit = ON)
b0	Data length	7 bit	8 bit
b1 b2	Parity	(b2, b1) (0, 0) : None (0, 1) : Odd (1, 1) : Even	
b3	Stop bit	1 bit	2 bit
b4 b5 b6 b7	Baud rate(bps)	(b7, b6, b5, b4) (0, 0, 1, 1) : 300 (0, 1, 0, 0) : 600 (0, 1, 0, 1) : 1,200 (0, 1, 1, 0) : 2,400	(b7, b6, b5, b4) (0, 1, 1, 1) : 4,800 (1, 0, 0, 0) : 9,600 (1, 0, 0, 1) : 19,200
b8*1	Header	None	Effective (D8124) Default : STX (02H)
b9*1	Terminator	None	Effective (D8125) Default : ETX (03H)
b10 b11 b12	Control line	No protocol: (b12, b11, b10) (0, 0, 0) : No use <RS-232C interface> (0, 0, 1) : Terminal mode <RS-232C interface> (0, 1, 0) : Interlink mode <RS-232C interface > (FX2N V2.00 or more) (0, 1, 1) : Normal mode 1 <RS-232C interface>, <RS-485 (RS-422) interface>*3 (1, 0, 1) : Normal mode 2 <RS-232C interface> (FX, FX2C only)	
		Computer link: (b12, b11, b10) (0, 0, 0) : RS-485 (RS-422) interface (0, 1, 0) : RS-232C interface	
b13*2	Sum check	Sum check code is not added	Sum check code is added automatically
b14*2	Protocol	No protocol	Dedicated protocol
b15*2	Transmission control protocol	Protocol format 1	Protocol format 4

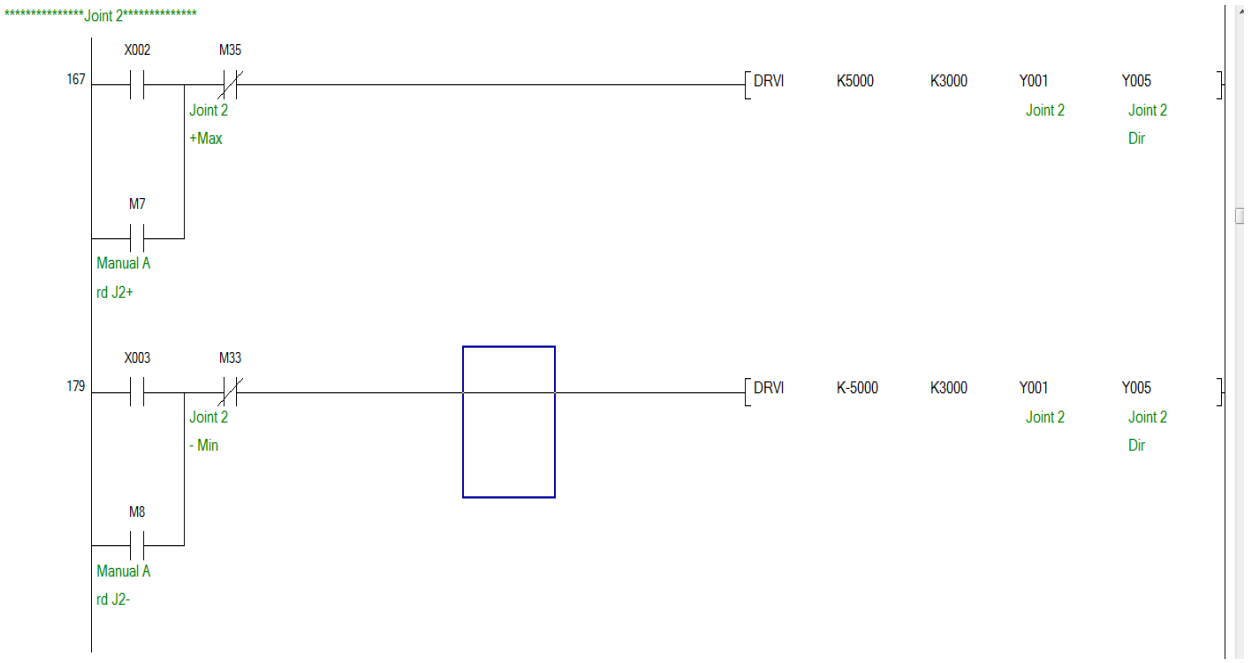
Εικόνα 4.21ModbussetupD8120

Χειροκίνητες εντολές

Η μνήμη M5(εικόνα 4.22) ενεργοποιείται από την οθόνη ώστε να κινηθεί ο κινητήρας προς την θετική πλευρά που έχει οριστεί, όσο είναι ενεργοποιημένη η M5 τόσο εκτελεί την κίνηση. Η M32 είναι το μέγιστο όριο που μπορεί να κινηθεί ο πρώτος κινητήρας για αυτό είναι κλειστή επαφή ώστε όταν ενεργοποιηθεί να κόψει το κύκλωμα. Η εντολή DRVI(Drive to Increment) είναι εντολή για positioning όπου το 16100 είναι τα βήματα που θα εκτελέσει, το 3000 είναι η συχνότητα που θα αναπαράγει τους παλμούς, το Y000 είναι η highspeedpulseoutput και η Y004 είναι η ψηφιακή έξοδος που καθορίζει την κατεύθυνση της κίνησης. Για να εκτελεστεί αντίθετη κίνηση του κινητήρα αρκεί να τοποθετηθεί ένα μείον μπροστά από τον αριθμό ων βημάτων. Το X000 είναι ψηφιακή είσοδος που χρειάστηκε στην υλοποίηση της εφαρμογής. Το ίδιο ισχύει για τους άλλους δυο κινητήρες με τις αντίστοιχες μνήμες και εξόδους(εικόνες 4.23 & 4.24).



Εικόνα 4.22 Εντολές για το χειροκίνητο για τον πρώτο σύνδεσμο

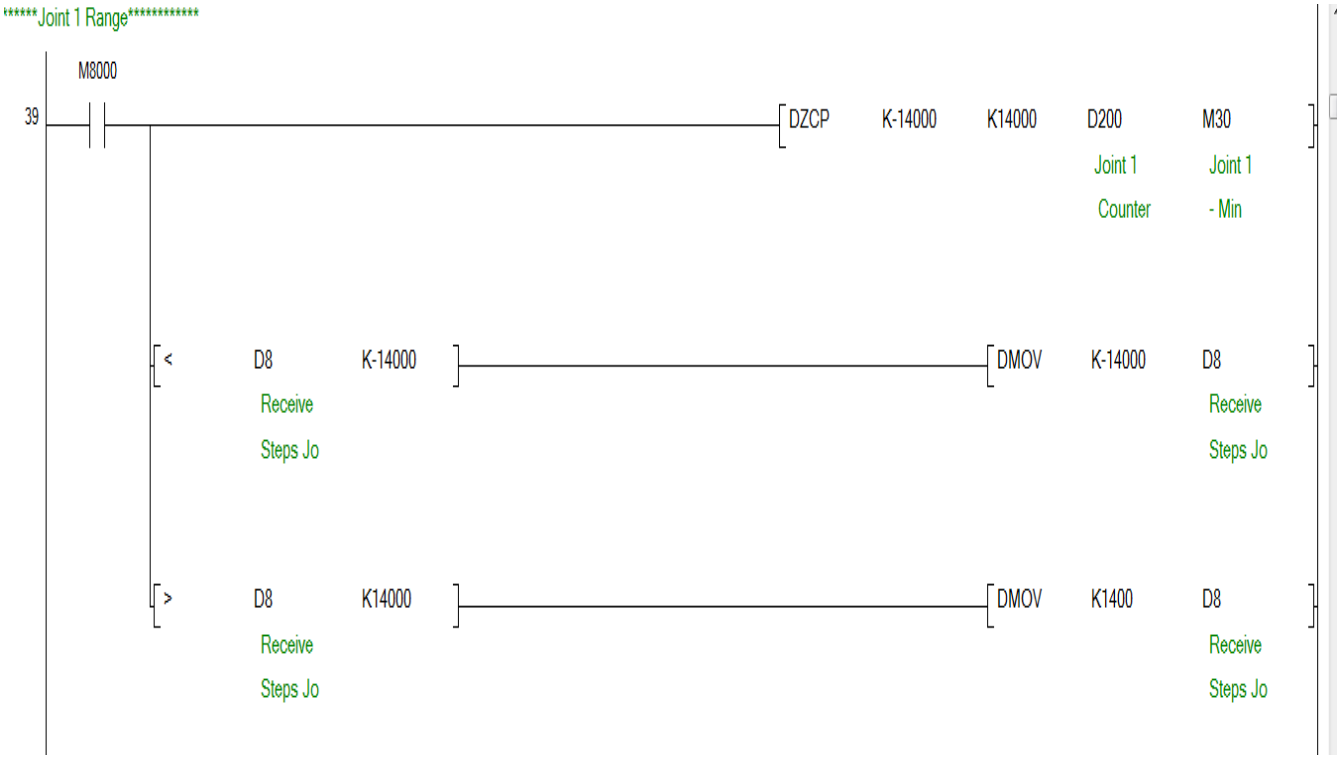


Εικόνα 4.23 Εντολές για το χειροκίνητο για τον δεύτερο σύνδεσμο

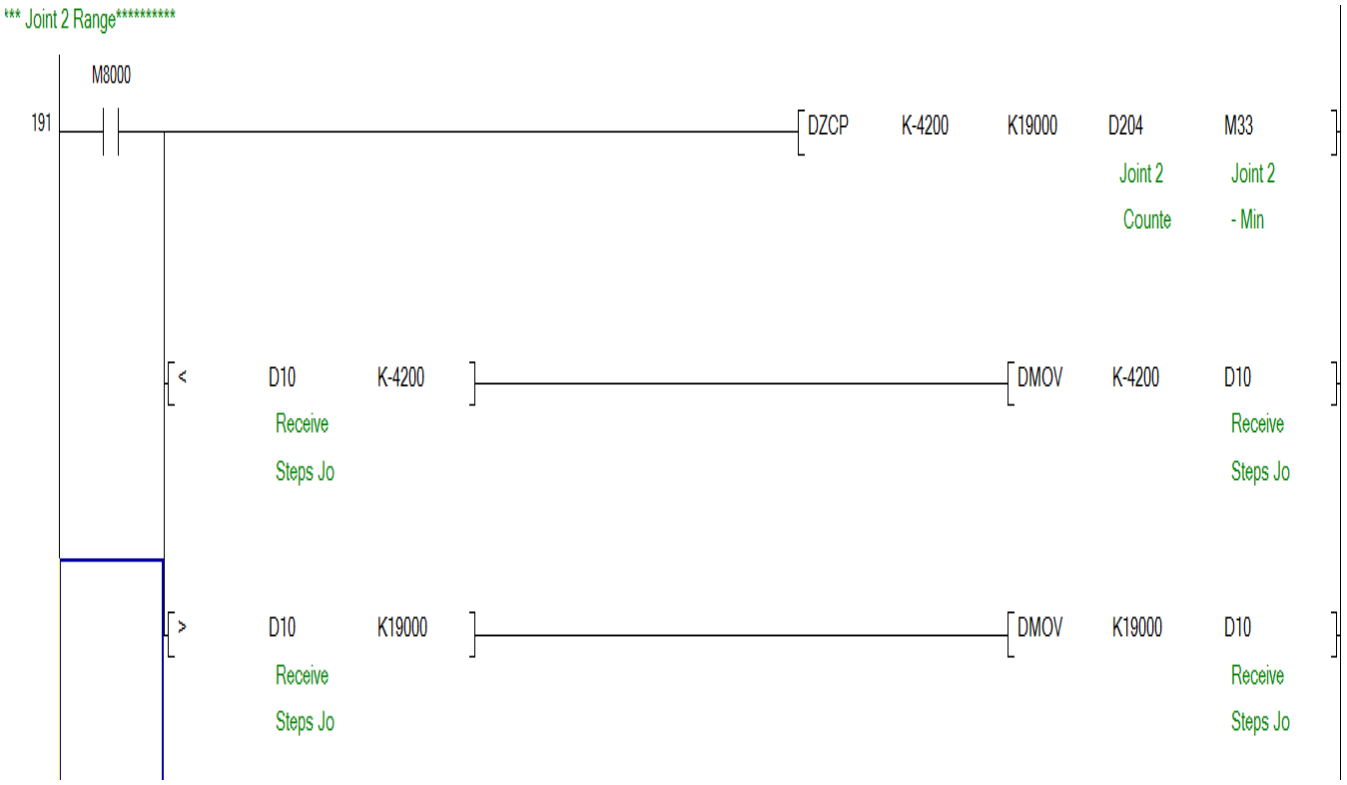
Εικόνα 4.24 Εντολές για το χειροκίνητο για τον τρίτο σύνδεσμο

Όρια κινητήρων

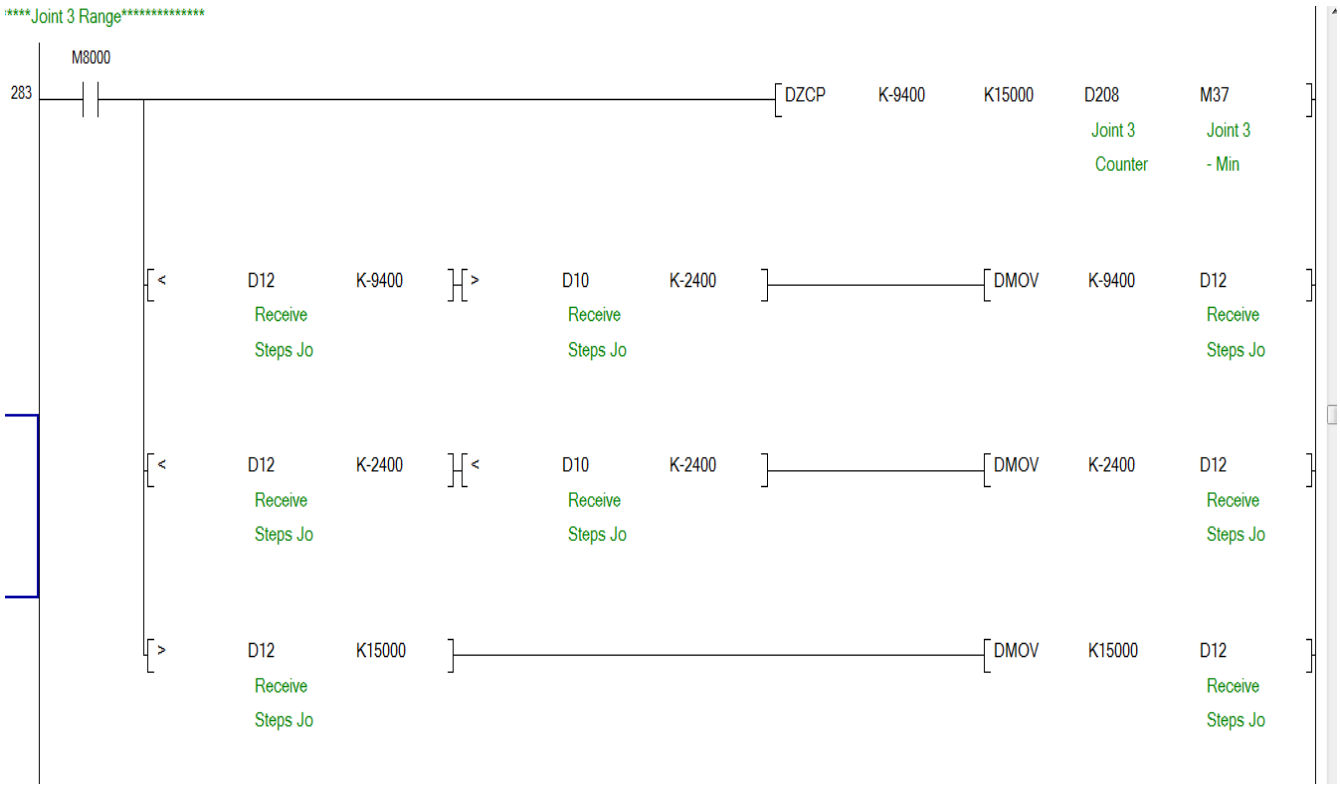
Η ειδική μνήμη M8000 είναι ενεργοποιημένη όσο η CPU είναι σε Run (εικόνα 4.25). Η εντολή DZCP ενεργοποιεί τρεις μνήμες και ανάλογα την τιμή του καταχωρείτε D200. Δηλαδή όταν ισχύει -14000 >D200 ενεργοποιείται η M30 μνήμη, όταν ισχύει -14000 <D200 < 14000 είναι ενεργοποιημένη η M31 και όταν ισχύει D200>14000 ενεργοποιείται η M32. Παρακάτω γίνονται δυο συγκρίσεις οι όποιες αφορούν τον καταχωρητή D8 που αποθηκεύεται η τιμή που στέλνει ο Arduino. Οι συγκρίσεις αυτές είναι offset ώστε αν η τιμή που θα λάβει το PLC ξεπερνούν τα όρια να μεταφέρει την αντίστοιχη τιμή είτε αρνητική είτε θετική στον D8. Το ίδιο ισχύει για τον δεύτερο κινητήρα με τις αντίστοιχες μνήμες και αντίστοιχους καταχωρητές. Στον τρίτο κινητήρα υπάρχει και μια τρίτη σύγκριση- offset ώστε αν ο δεύτερος κινητήρας έχει λάβει τιμή μικρότερη από το -1 τότε το αρνητικό όριο του τρίτου κινητήρα είναι -2400, αυτό έχει σκοπό ώστε να μην χτυπήσει η αρπαγή στο έδαφος(εικόνα 4.27).



Εικόνα 4.25 Εντολές για όρια πρώτου συνδέσμου



Εικόνα 4.26 Εντολές για όρια δευτέρου συνδέσμου

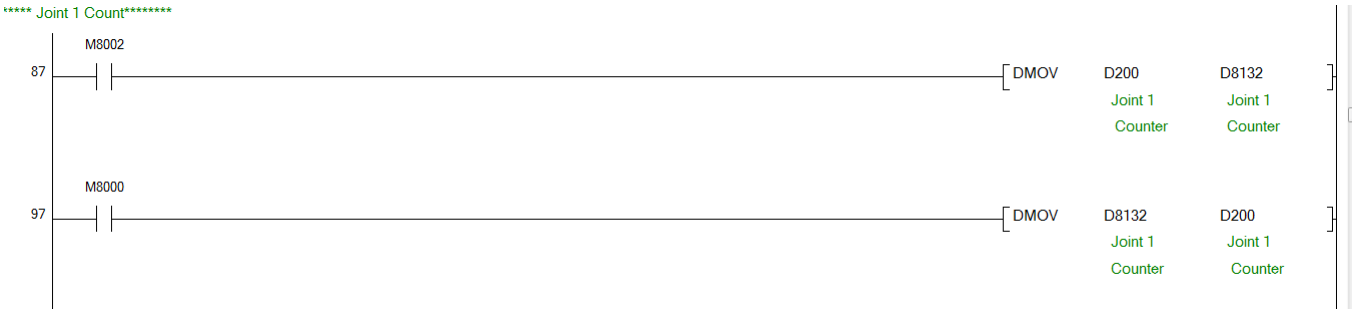


Εικόνα 4.27 Εντολές για όρια τρίτου συνδέσμου

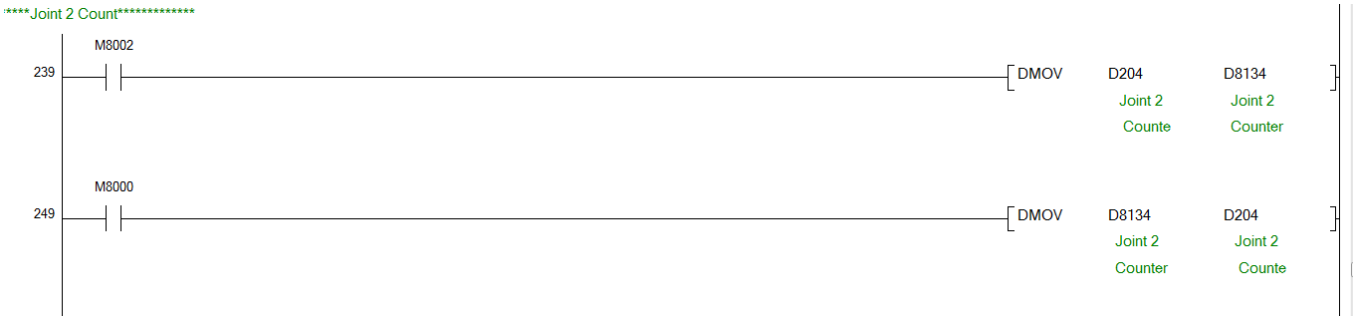
Μετρητές

Οι μετρητές στην εφαρμογή αντλούν τις τιμές τους από τους παλμούς που στέλνονται στις εξόδους είτε αφαιρούν είτε προσθέτουν. Για τον πρώτο κινητήρα(εικόνα 4.28) όταν η CPU μεταβεί σε κατάσταση Run θα μεταφέρει την τιμή που έχει ο D200 στον D8132. Όσο η CPU είναι σε κατάσταση Run, η τιμή του D8132 θα μεταφερθεί στον D200.

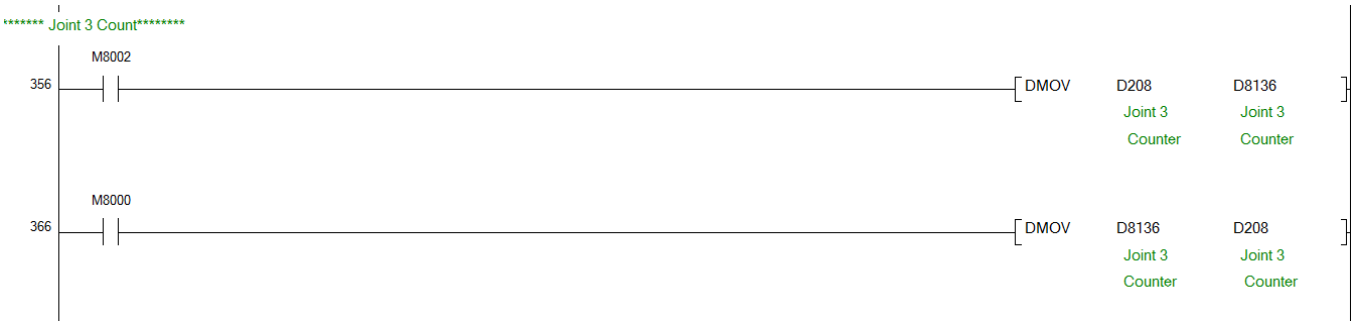
Αυτό γίνεται γιατί ο D8132 είναι ειδικός καταχωρητής που μετράει του παλμούς για την Y000 και αντίστοιχα ο D8134 για την Y001 και ο D8136 για την Y002. Όμως αν γίνει διακοπή ρεύματος αυτοί οι ειδικοί καταχώρησες χάνουν τις τιμές τους, και για να μην γίνεται κάθε φορά μηδενισμός του βραχίονα μεταφέρονται αυτές οι τιμές στους καταχώρησες D200, D204 και D208 ώστε αυτοί οι καταχωρητές δεν χάνουν ποτέ τις τιμές τους. Αυτό γίνεται βάση τουhardware του PLC και με την μπαταριά που διαθέτει.



Εικόνα 4.28 Εντολές μετρητών πρώτου συνδέσμου



Εικόνα 4.29 Εντολές μετρητών δευτέρου συνδέσμου

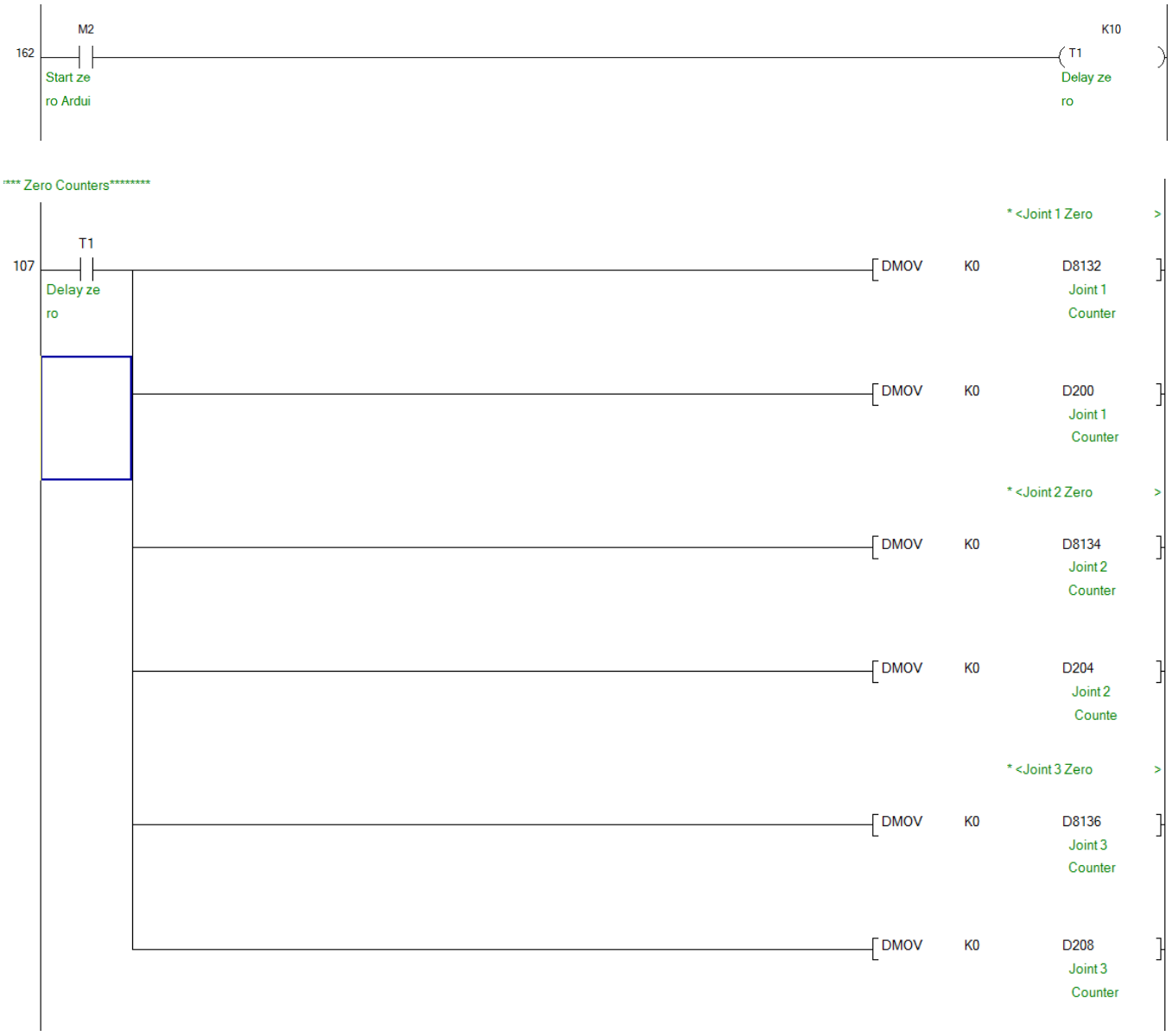


Εικόνα 4.30 Εντολές μετρητών τρίτου συνδέσμου

Μηδενισμός μετρητών και εντολή GoHome

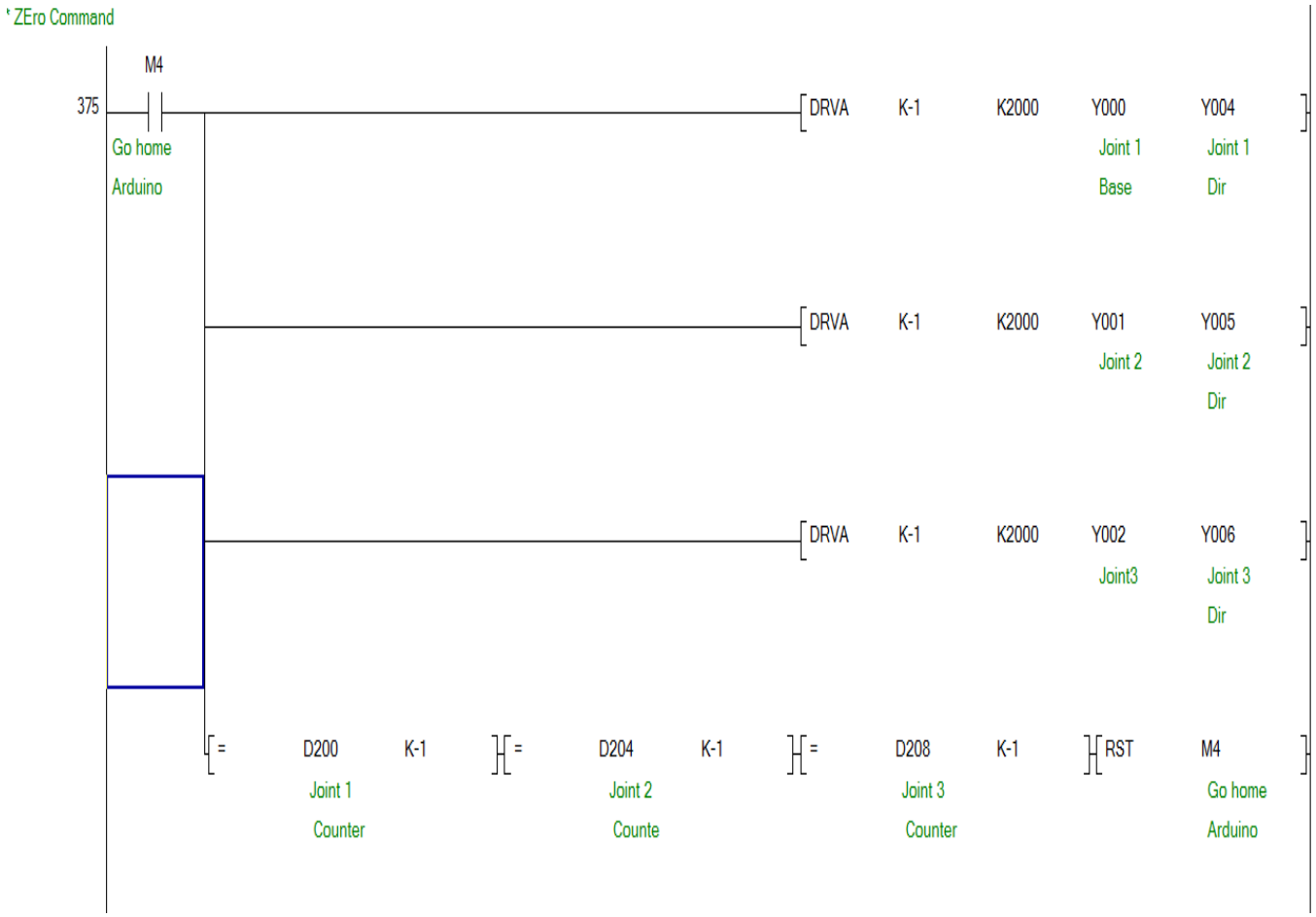
Όταν ενεργοποιηθεί η μνήμη M2 από τον Arduino μετά από 1 δευτερόλεπτο θα ενεργοποιηθεί το χρονικό T1 όπου αυτό με την σειρά θα μεταφέρει μέσο της εντολής Dmov την τιμή μηδέν σε όλους τους μετρητές.

Ο σκοπός του χρονικού είναι καθαρά για λογούς ασφάλειας.



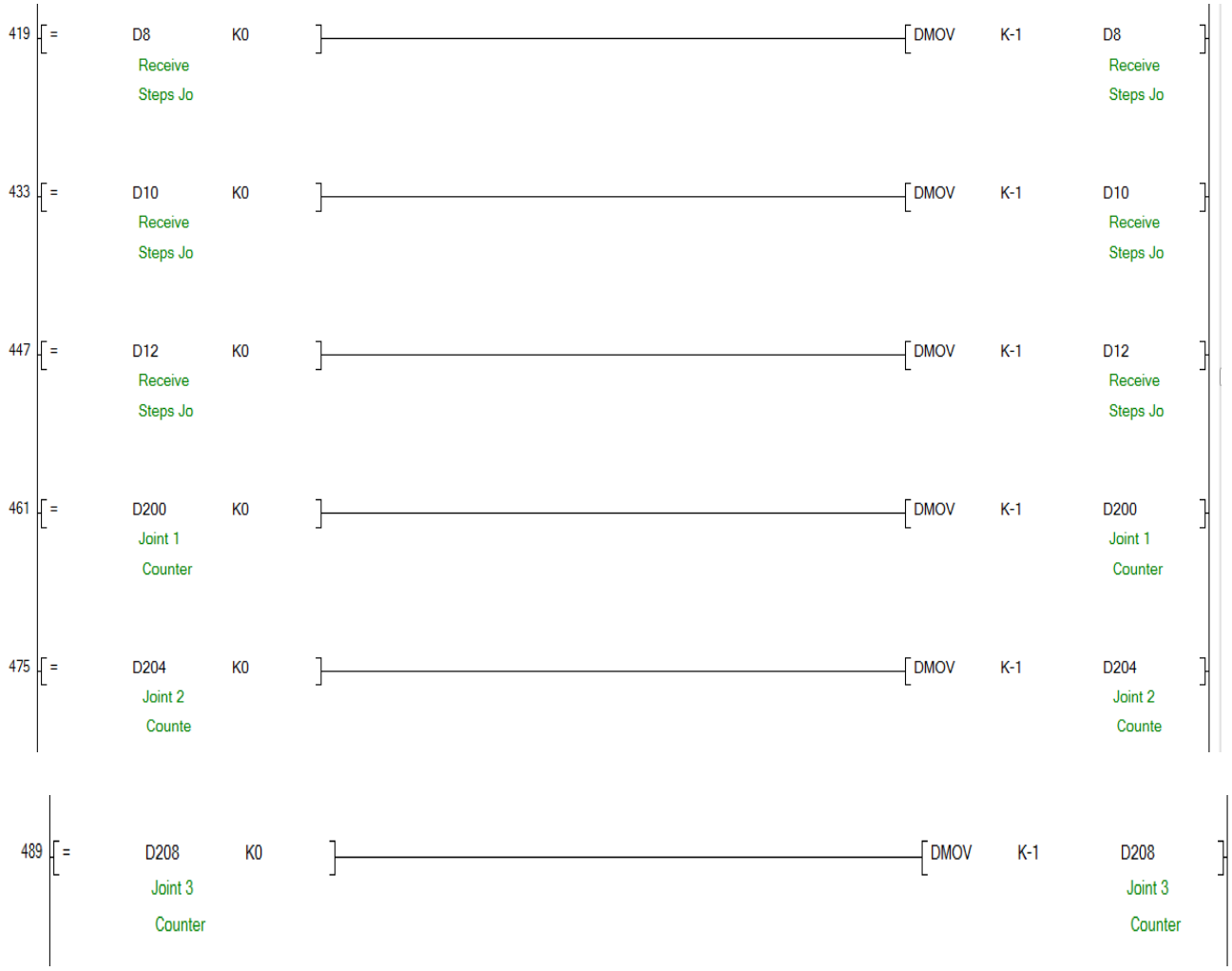
Εικόνα 4.31 Εντολές μηδενισμός μετρητών

Όταν ενεργοποιηθεί η επιλογή από την οθόνη GoHome ο Arduino θα ενεργοποιήσει την μνήμη M4. Όπου αυτή θα ενεργοποιήσει τρεις εντολές DRVA που σημαίνει Drive to Absolute στο -1 για κάθε κινητήρα. Το -1 επιλέχτηκε αντί του μηδενός διότι στην πράξη άμα ήταν μηδέν θα εκτελούταν κίνηση μόνο από τα θετικά προς το μηδέν. Οι παράμετροι της εντολής DRVA είναι παρόμοιοι με της εντολής DRVI. Τέλος όταν οι τρεις συγκρίσεις είναι αληθείς τότε θα γίνει απενεργοποίηση της M4.



Εικόνα 4.32 Εντολές GoHome

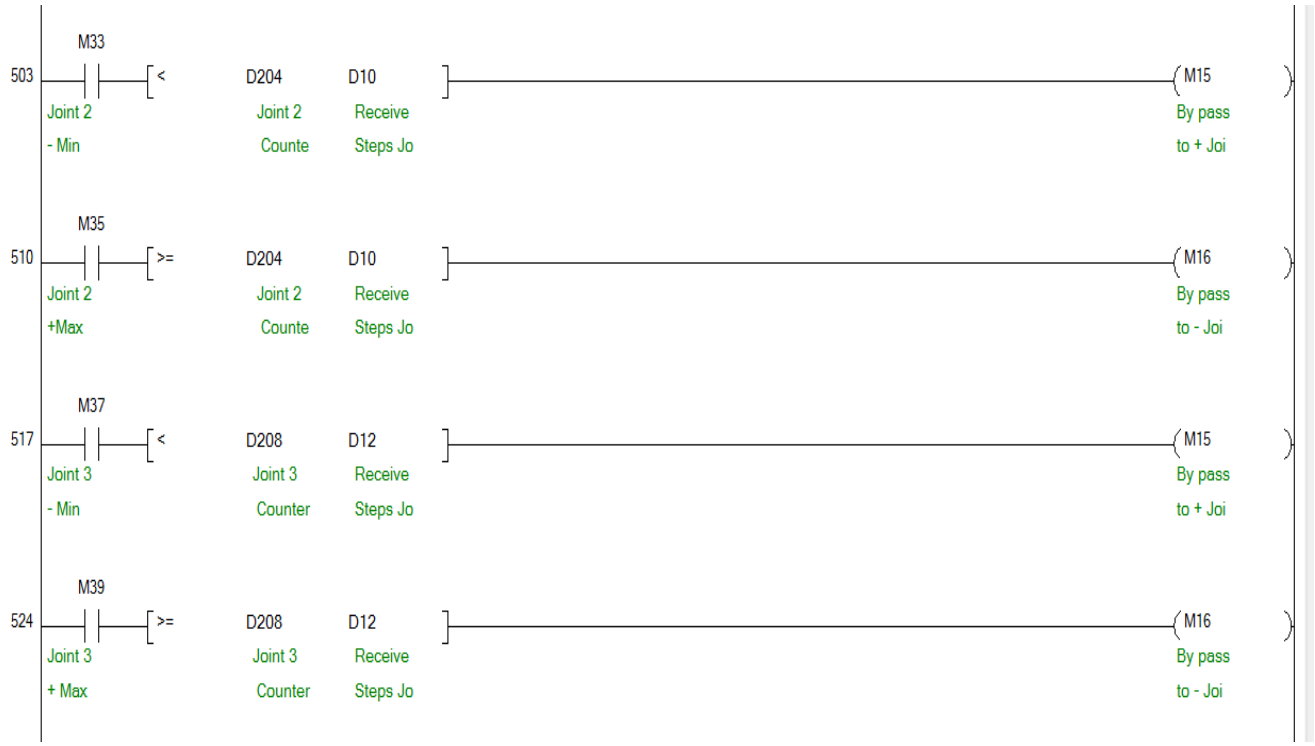
Στην εικόνα 4.33 φαίνονται έξι συγκρίσεις όλων των καταχωρητών ακόμα και αυτών που αποθηκεύουν τις τιμές που στέλνει ο Arduinoκαι η μεταφορά της τιμής -1 στον καθένα αν είναι ίσος με το μηδέν. Αυτό γίνεται για την ιδιότητα της εντολής DRVA που αναφέρθηκε παραπάνω.



Εικόνα 4.33 Εντολές μεταφορά -1 στους μετρητές

Εκτέλεση απεσταλμένων τιμών

Στην εικόνα 4.34 έχουν δημιουργηθεί συνθήκες σε συνδυασμό με τις μνήμες των ορίων των κινητήρων ώστε να γίνεται παράκαμψη των ορίων για τον αντίστοιχο όριο του κινητήρα άμα είναι να κινηθεί σε αντίθετη κατεύθυνση από το όριο που ενεργοποιήθηκε. Οι παρακάτω συνθήκες ισχύουν μόνο για τον δεύτερο και τρίτο κινητήρα.



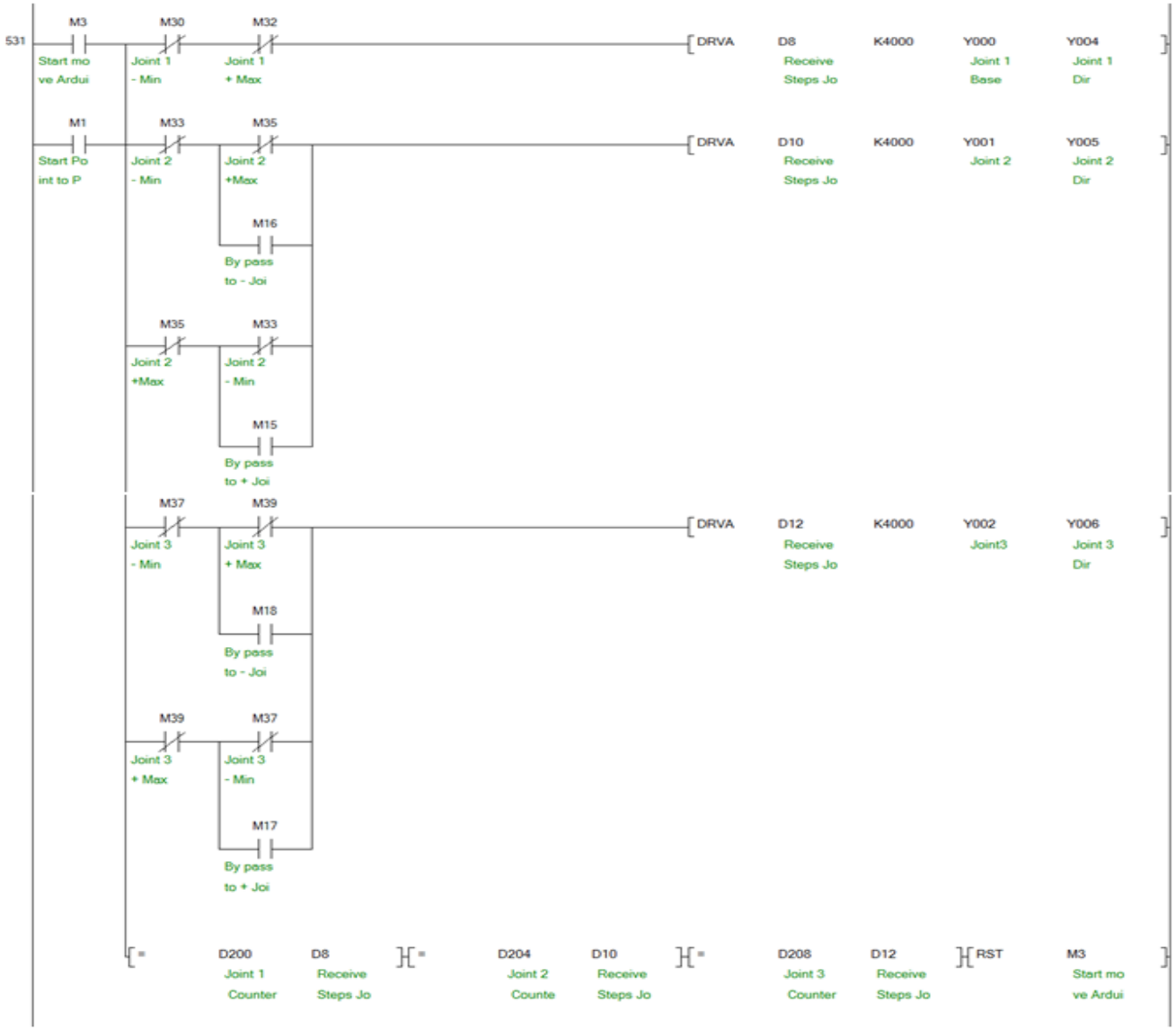
Εικόνα 4.34 Εντολές bypass ορίων

Παρακάτω όταν ενεργοποιηθεί η μνήμη M3 θα ενεργοποιήσουν τρεις εντολές DRVA για κάθε κινητήρα ανάλογα με την τιμή που έστειλε οArduino όταν λύσει το αντίστροφο κινηματικό πρόβλημα. Οι τρεις συγκρίσεις στο τέλος του network χρησιμοποιούνται ώστε όταν γίνουν αληθής σημαίνει ότι είναι στο επιθυμητό σημείο ο βραχίονας και θα γίνει απενεργοποίηση της M3.

Η μνήμη M1 ενεργοποιείται όταν έχει επιλεχτεί η λειτουργιά PointtoPoint. Η απενεργοποίηση της M1 γίνεται μετά από ορισμένο χρόνο.

Επίσης στην σειρά με τις μνήμες ενεργοποίησης έχουν τοποθετηθεί η μνήμες ορίων και bypass ώστε αν σε κάποια περίπτωση είναι να κινηθεί ο κινητήρας πέρα από το όριο να

σταματήσει ο αντίστοιχος κινητήρας ή άμα έχει ενεργοποιηθεί κάποια μνήμη ορίου και η καινούρια θέση είναι προς την αντίθετη κατεύθυνση του ορίου να μπορεί να κινηθεί.



Εικόνα 4.35 Εντολές εκτελείς των απεσταλμένων τιμών από τον Arduino

4.4 Πρόγραμμα Arduino

Η πλακέτα του Arduino προγραμματίζεται στο περιβάλλον Arduino IDE. Η γλωσσά προγραμματισμού που χρησιμοποιεί ονομάζεται Wiring η οποία είναι βασισμένη στην C και στην C++. Μερικά στοιχειά του κώδικα χρησιμοποιήθηκαν από το χρηστή στον σελιδοδείκτη[12].

Παρακάτω γίνεται η επεξήγηση του κώδικα:

Δήλωση βιβλιοθηκών

Στην αρχή δηλώνονται οι βιβλιοθήκες που θα χρησιμοποιηθούν. Η math για να εκτελεί μαθηματικές πράξεις όπως όταν τετραγωνίζεται μια μεταβλητή ή υπολογίζεται μια μαθηματική ριζά, η Modbusmaster είναι για να δημιουργεί η επικοινωνία με πρωτόκολλο ModbusRS485 και οι υπόλοιπες τέσσερεις είναι για την δημιουργία γραφικών και διαχείριση της οθόνης TFT.

```
#include <math.h>
#include <ModbusMaster.h>
#include <SPI.h>
#include <MCUFRIEND_kbv.h>
#include <Adafruit_GFX.h>
#include <Adafruit_TFTLCD.h>
```

Δήλωση Pins και Αντικειμένων
Παρακάτω δηλώνονται τα pins που θα χρησιμοποιηθούν για την οθόνη.

```
//TFT pins
#define LCD_CS A3     // Chip Select goes to Analog 3
#define LCD_CD A2     // Command/Data goes to Analog 2
#define LCD_WR A1     // LCD Write goes to Analog 1
#define LCD_RD A0     // LCD Read goes to Analog 0
#define LCD_RESET A4  // Can alternately just connect to
Arduino's reset pin
#define YP A2  // must be an analog pin, use "An" notation!
```

```
#define XM A3  // must be an analog pin, use "An" notation!
#define YM 8   // can be a digital pin
#define XP 9   // can be a digital pin
```

Ενεργοποίηση την επικοινωνίας Modbus.

```
//modbuspin
#defineMAX485_RE_NEG  2
```

Δήλωση λέξεων με 16δικο αριθμό που αντιστοιχεί στο κάθε χρώμα.

```
// Define some TFT readable colour codes to human readable
names
#define BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF
```

Ορισμός pins πληκτρολογίου

```
//keyboard pins
#define enter 9
#define plusup 10
#define minusdown 11
#define right 12
```

Δήλωση ονόματος επικοινωνίας.

```
ModbusMasternode;
```

Δήλωση ονόματος οθόνης.

```
MCUFRIEND_kbvtft;
```

Δήλωση Μεταβλητών

Οι μεταβλητές δημιουργήθηκαν για τον σκοπό του menu της οθόνης.

```
//tft vars
int choose=0;
int choose1=0;
int choose2=0;
int chose3=0;
int currentpage = 0;
double u1=0,u2=0,u3=0; //tft inverse vars
int tp11, tp12, tp13, tp21, tp22, tp23,tp31, tp32, tp33, tp41,
tp42, tp43; //point vars
```

Οι μεταβλητές χρειάστηκε να είναι global για τον υπολογισμό του αντίστροφου κινηματικού προβλήματος. Οι μεταβλητές l1,l2,l3 είναι τα μήκη κάθε συνδέσμου αντίστοιχα με αρχή την βάση. Οιτιμέςτους είναι σε εκατοστά και η μέτρηση τους έγινε από το κέντρο του κάθε μειωτή. Τα όρια των γωνιών των αρθρώσεων Th2_max_negative, Th2_max_positive, Th3_max_negative, Th3_max_positive είναι σε μοίρες και έγινε η μέτρηση με μοιρογνωμόνιο συμφώνα με τα φυσικά όρια των μειωτήρων.

```
doublel1=14, l2=7.5, l3=6; //thelengthsofthearms
int x, y, z; //for the position in space
double Th1=0, Th2=0, Th3=0; //initial angles (when arduino
starts)
double  C3_Result;
bool Th3_Ph_Range_Error=false, Th2_Ph_Range_Error=false,
C3_Error=false;
double Th1_Calc_1, Th2_Calc_1, Th3_Calc_1; //used for first
solution
double Th1_Calc_2, Th2_Calc_2, Th3_Calc_2; //used for second
solution
/Angle limits
double Th2_max_negative= -140;
```

```
double Th2_max_positive= 150;
double Th3_max_negative= -80;
doubleTh3_max_positive= 150;
```

Δημιουργία συναρτήσεων

Οι συναρτήσεις preTransmission και postTransmission χρησιμοποιήθηκαν για να ενεργοποιούν ή να απενεργοποιούν την επικοινωνία.

```
void preTransmission(){
digitalWrite(MAX485_RE_NEG, 1);
}
void postTransmission(){
digitalWrite(MAX485_RE_NEG, 0);
}
```

Οι συναρτήσεις μετατρέπουν τις γωνιές που υπολογιστήκαν σε ακέραιους.

```
double TH1(int Deg){
  return Deg;
}
double TH2(int Deg){
return Deg ;
}
double TH3(int Deg){
return Deg ;
}
```

Η συνάρτηση Rad_To_Deg μετατρέπει έναν αριθμό από Rad σε μοίρες.

```
double Rad_To_Deg(double Conv){
  return (Conv* 4068) / 71;
}
```

Η παρακάτω συνάρτηση C3 υπολογίζει το C3 που έχει αναφερθεί στην ενότητα 3.4 και έλεγχοι αν είναι στα όρια 1 με -1 ώστε να επιστρέφει την ανάλογη ψηφιακή μεταβλητή.

```
double C3(int x,int y,int z)
{
  C3_Result = (sq(x)+sq(y)+sq(z-l1)-sq(l2)-sq(l3))/(2*l2*l3);
```

```
  //C3 has limitation due to square root. these limitations
should not be exceeded (dont enter big x,y,z)
if(C3_Result<(-1) || C3_Result>1)
  {
    C3_Error = true;
node.writeSingleCoil(0x00, 1);
  }
else{
    C3_Error=false;
    return C3_Result;
}
}
```

Η συνάρτηση S3 υπολογίζει το αριθμό S3 που έχει αναφερθεί στην ενότητα 3.4 και ανάλογα ποια λύση ικανοποιεί τις παραμέτρους, επιστρέφει την ανάλογη λύση.

```
double S3(int x,int y,int z, bool Second_Solution=false)
{
if (Second_Solution==false){//up elbow slution
return sqrt(1-sq(C3(x,y,z)));
  }
else if(Second_Solution==true){//down elbow solution
return -sqrt(1-sq(C3(x,y,z)));
}
}
```

Οι παρακάτω συναρτήσεις υπολογίζουν τις γωνιές Θ1,Θ2,Θ3 συμφώνα με τα x-y-z που δηλώθηκαν. Οι μαθηματικές εξισώσεις αναφέρονται στην ενότητα 3.4

```
//AngleTh1(base)
doubleTh_1(intx,inty) {
if (C3_Error==false){
double result1=Rad_To_Deg(atan2 (y,x));
```

```
     return result1;
    }
}

//Angle Th2
double Th_2(int x, int y, int z, bool Second_Solution=false)
{
if (C3_Error==false){
   double result2=Rad_To_Deg( atan2(z-l1, sqrt(sq(x)+ sq(y)))
- atan2(l3*S3(x,y,z,Second_Solution), l2+(l3*C3(x,y,z))));
     return result2;
  }
}

//Angle Th3
double Th_3(int x,int y,int z,bool Second_Solution=false)
{
if (C3_Error == false){
return Rad_To_Deg(atan2(S3(x,y,z,
Second_Solution),C3(x,y,z)));
}
}
```

Η συνάρτηση Physical_Limitations_Check εκτελεί έλεγχο αν οι υπολογισμένες γωνιές είναι μέσα στα φυσικά όρια του βραχίονα. Η παρούσα συνάρτηση θα καλεστεί δυο φορές μέσα στον υπολογισμό του αντίστροφου κινηματικού προβλήματος διότι υπάρχουν δυο λύσεις.

```
void Physical_Limitations_Check(double Th2_Calc, double
Th3_Calc)
{
        if(Th2_Calc<(Th2_max_negative) ||
Th2_Calc>(Th3_max_positive))
      {
```

```
        Th2_Ph_Range_Error=true;
      }
else{
        Th2_Ph_Range_Error=false;
      }
if(Th3_Calc< (Th3_max_negative) ||
Th3_Calc>(Th2_max_positive))
      {
        Th3_Ph_Range_Error=true;
      }
      else
      {
        Th3_Ph_Range_Error=false;
}
}
```

Η συνάρτηση Inverse_Calc στην αρχή ελέγχει αν ο C3 είναι έγκυρος και εάν είναι τότε υπολογίζει την πρώτη λύση για τις γωνιές Θ1,Θ2,Θ3 με την πρώτη λύση του S3 , αν όχι τότε η συνάρτηση τερματίζει. Αφού υπολογίσει τις γωνιές από την πρώτη λύση εκτελεί έλεγχο ότι οι γωνιές βρίσκονται μέσα στα φυσικά όρια, αν είναι μέσα στα όρια τότε στέλνει την Θ1 στο plc. Η Θ2 έχει offset πριν σταλθεί οπότε ανάλογα με την γωνία θα σταλθεί και η ανάλογη τροποποιημένη τιμή. Η Θ3 στέλνεται χωρίς offset και τερματίζει η συνάρτηση. Οι γωνιές πριν σταλθούν πολλαπλασιάζονται με τον αριθμό 145 διότι 145 βήματα είναι μια μοίρα.

```
void Inverse_Calc(int x,int y,int z)  // Inverse Kinetics
Function{
   C3(x,y,z);
if(C3_Error == false){
      Th1_Calc_1=Th_1(x,y);
      Th2_Calc_1=Th_2(x,y,z);
      Th3_Calc_1=Th_3(x,y,z);
node.writeSingleCoil(0x00, 0);
   }
```

```
else{
return(1); //if there is an C3_Error, this function will exit
here 1=true
   }
  Physical_Limitations_Check(Th2_Calc_1, Th3_Calc_1);
if ((Th2_Ph_Range_Error==false) && (Th3_Ph_Range_Error==false)
&& y>=0){
    //Offset Theta 1
if(Th1_Calc_1>=90){
node.writeSingleRegister(0x08,90*145);
          }
else if(Th1_Calc_1<=-90){
node.writeSingleRegister(0x08,(-90)*145);
          }
else  {
node.writeSingleRegister(0x08,Th1_Calc_1*145);
          }

  //Offset Theta 2
if(Th2_Calc_1>131){
node.writeSingleRegister(0x00a,131*145);
           }
else  if(Th2_Calc_1<-180){
node.writeSingleRegister(0x00a,(180+Th2_Calc_1)*145);
           }
else if (Th2_Calc_1<-29){
node.writeSingleRegister(0x00a,-29*145);
              }
           else {
node.writeSingleRegister(0x00a,Th2_Calc_1*145);
              }
    //Offset Theta 3
node.writeSingleRegister(0x00c,Th3_Calc_1*145);
  }
```

```
else{
    Th2_Ph_Range_Error=false;
    Th3_Ph_Range_Error=false;
```

Αν όταν γίνεται ο πρώτος έλεγχος των φυσικών ορίων και έστω μια γωνία ξεπερνά τα φυσικά όρια τότε θα ξανά υπολογιστούν οι γωνιές με την δεύτερη τιμή του S3. Αφού υπολογιστούν δεύτερη φορά οι γωνιές τότε θα γίνουν πάλι οι ίδιες ενέργειες. Αν ούτε η δεύτερη λύση δεν ικανοποιεί τις παραμέτρους τότε θα τερματιστεί η συνάρτηση και θα ζητήσει να ενεργοποιηθεί η Y007 του PLC ως ένδειξη σφάλματος

```
//*****************Second Solution*********************

      Th1_Calc_2=Th_1(x,y);
      Th2_Calc_2=Th_2(x,y,z,true);
      Th3_Calc_2=Th_3(x,y,z,true);
 Physical_Limitations_Check(Th2_Calc_2, Th3_Calc_2);
if ((Th2_Ph_Range_Error==false) && (Th3_Ph_Range_Error==false)
&& y>=0){
if(Th1_Calc_2>=90){
node.writeSingleRegister(0x08,90*145);
          }
else if(Th1_Calc_2<=-90){
node.writeSingleRegister(0x08,(-90)*145);
          }
else  {
node.writeSingleRegister(0x08,Th1_Calc_2*145);
           }
     //Offset Theta 2
if(Th2_Calc_2>131){
node.writeSingleRegister(0x00a,131*145);
           }
else  if(Th2_Calc_2<-180){
node.writeSingleRegister(0x00a,(180+Th2_Calc_2)*145);
           }
```

```
else if (Th2_Calc_2<-29){
node.writeSingleRegister(0x00a,-29*145);
                }
           else {
node.writeSingleRegister(0x00a,Th2_Calc_2*145);
                 }
    //Offset Theta 3
node.writeSingleRegister(0x00c,Th3_Calc_2*145);
 }
}
      C3_Error= false;
      Th2_Ph_Range_Error=false;
      Th3_Ph_Range_Error=false;
}
```

Η συνάρτηση point_point όταν κληθεί υπολογίζει τις γωνιές για τις πρώτες συντεταγμένες , στέλνει τις τιμές στο PLC και εκτελεί την κίνηση. Όσο εκτελείται η πρώτη κίνηση υπολογίζονται και οι γωνιές για το δεύτερο σημείο. Μόλις εκτελεστεί η πρώτη κίνηση θα ξεκινήσει η δεύτερη και ούτω καθεξής μέχρι να ολοκληρωθεί και η κίνηση του τέταρτου σημείου.

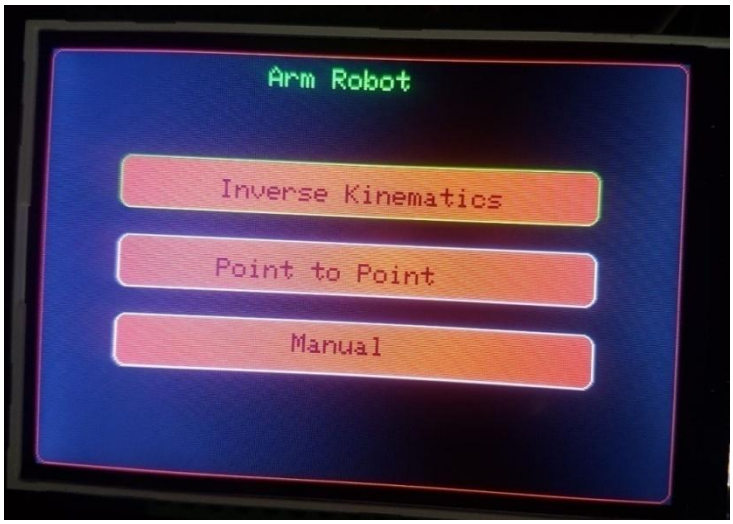
```
double point_point(int p11, int p12, int p13, int p21, int
p22, int p23,int p31, int p32, int p33, int p41, int p42, int
p43 ) {
 //Point 1
   Inverse_Calc(p11,p12,p13);
delay(200);
node.writeSingleCoil(0x0001, 1);
delay(2000);
node.writeSingleCoil(0x0001, 0);
delay(200);

     //Point 2
     Inverse_Calc(p21,p22,p23);
```

```
delay(200);
node.writeSingleCoil(0x0001, 1);
delay(2000);
node.writeSingleCoil(0x0001, 0);
delay(200);

    //Point 3
     Inverse_Calc(p31,p32,p33);
delay(200);
node.writeSingleCoil(0x0001, 1);
delay(2000);
node.writeSingleCoil(0x0001, 0);
delay(200);

     //Point 4
      Inverse_Calc(p41,p42,p43);
delay(200);
node.writeSingleCoil(0x0001, 1);
delay(2000);
node.writeSingleCoil(0x0001, 0);
     //delay(3000);
}
```

Δημιουργία σελίδων οθόνης

Στην συνάρτηση drawHomeScreenπαρουσιάζεται η αρχική οθόνη(εικόνα 4.36)

```
void drawHomeScreen(){
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, RED);
tft.setTextSize(2);
tft.setCursor(170, 10);
tft.setTextColor(GREEN);
tft.print("Arm Robot");
tft.fillRoundRect(60, 80, 360, 40, 8, RED);
// red box
tft.drawRoundRect(60, 80, 360, 40, 8, WHITE);
tft.fillRoundRect(60, 140, 360, 40, 8, RED);
tft.drawRoundRect(60, 140, 360, 40, 8, WHITE);
tft.fillRoundRect(60, 200, 360, 40, 8, RED);
tft.drawRoundRect(60, 200, 360, 40, 8, WHITE);
tft.setTextColor(BLACK);
 //box text
tft.setCursor(135, 95);
tft.print("Inverse Kinematics");
tft.setCursor(135, 155);
tft.print("Point to Point");
tft.setCursor(195, 210);
  tft.print("Manual");
}
```



Εικόνα 4.36 Αρχική σελίδα menu

Η συνάρτηση drawBackButtonσχεδιάζει το πλήκτρο επιστροφής(εικόνα 4.37).

```
void drawBackButton(){
 //back box
tft.setTextSize(2);
tft.fillRoundRect(370, 275, 70, 40, 8, BLUE);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);
tft.setCursor(380, 285);
tft.setTextColor(
}
```

Εικόνα 4.37 Πλήκτρο επιστροφής οθόνης

Η συνάρτηση drawinverse σχεδιάζει το επόμενο του αντίστροφου κινηματικού προβλήματος(εικόνα 4.38). Εμφανίζει τις τρέχον τιμές του x-y-z, των γωνιών από την πρώτη λύση και την δεύτερη λύση. Περιέχει τα κουμπιά έναρξη υπολογισμός γωνιών, έναρξη κίνησης και έναρξη επιστροφής στην αρχική θέση μηδέν. Επίσης περιέχει και τα γραφήματα , πια είναι η τρέχον επιλογή.

```
void drawinverse(int u11,int u22,int u33){
//x box
tft.setCursor(40, 30);
tft.setTextColor(RED);
  tft.print("X");
tft.fillRoundRect(20, 50, 50, 40, 8, RED);
  //y box
tft.setCursor(100, 30);
tft.setTextColor(RED);
  tft.print("Y");
tft.fillRoundRect(80, 50, 50, 40, 8, RED);
  //z box
tft.setCursor(160, 30);
tft.setTextColor(RED);
  tft.print("Z");
tft.fillRoundRect(140, 50, 50, 40, 8, RED);
  //display vals xyz
tft.setTextSize(1);
tft.setTextColor(BLACK);
tft.setCursor(30,65);
              tft.print(u11);
tft.setCursor(100,65);
              tft.print(u22);
tft.setCursor(160,65);
              tft.print(u33);
//calc theta box
tft.setCursor(205, 30);
tft.setTextColor(RED);
tft.print("Calc Thetas");
tft.fillRoundRect(240, 50, 70, 40, 8, BLUE);
tft.drawRoundRect(240, 50, 70, 40, 8, WHITE);

tft.setCursor(245, 62);
tft.setTextColor(BLACK);
```

```
  tft.print("Start");
  //start move box
tft.setCursor(370, 30);
tft.setTextColor(RED);
tft.print("Go X-Y-Z");
tft.fillRoundRect(380, 50, 70, 40, 8, BLUE);
tft.drawRoundRect(380, 50, 70, 40, 8, WHITE);

tft.setCursor(385, 62);
tft.setTextColor(BLACK);
  tft.print("Start");
  //home box
tft.setCursor(240, 100);
tft.setTextColor(RED);
  tft.print("GoHome");
tft.fillRoundRect(240, 120, 70, 40, 8, BLUE);
tft.drawRoundRect(240, 120, 70, 40, 8, WHITE);

tft.setCursor(245,132);
tft.setTextColor(BLACK);
  tft.print("Start");
  //Solution 1
tft.setCursor(40, 120);
tft.setTextColor(GREEN);
tft.print("Solution 1");
//th1calc1 box
tft.setCursor(40, 140);
tft.setTextColor(RED);
  tft.print("Th1");
tft.fillRoundRect(30, 160, 55, 40, 8, CYAN);
  //th2calc1 box
tft.setCursor(100, 140);
tft.setTextColor(RED);
  tft.print("Th2");
```

```
tft.fillRoundRect(90, 160, 55, 40, 8, CYAN);
  //th3calc1 box
tft.setCursor(160, 140);
tft.setTextColor(RED);
  tft.print("Th3");
tft.fillRoundRect(150, 160, 55, 40, 8, CYAN);
  //Solution 2
tft.setCursor(40, 220);
tft.setTextColor(GREEN);
tft.print("Solution 2");
//th1calc2 box
tft.setCursor(40, 240);
tft.setTextColor(RED);
  tft.print("Th1");
tft.fillRoundRect(30, 260, 55, 40, 8, CYAN);
  //th2calc2 box
tft.setCursor(100, 240);
tft.setTextColor(RED);
  tft.print("Th2");
tft.fillRoundRect(90, 260, 55, 40, 8, CYAN);
  //th3calc2 box
tft.setCursor(160, 240);
tft.setTextColor(RED);
  tft.print("Th3");
tft.fillRoundRect(150, 260, 55, 40, 8, CYAN);
   //C3
tft.setCursor(240, 285);
tft.setTextColor(RED);
  tft.print("C3");
tft.fillRoundRect(270, 275, 50, 35, 8, YELLOW);
          //display vals
tft.setTextSize(1);
tft.setTextColor(BLACK);
tft.setCursor(35,175);
```

```
              tft.print(Th1_Calc_1);

tft.setCursor(95,175);
              tft.print(Th2_Calc_1);

tft.setCursor(155,175);
              tft.print(Th3_Calc_1);

tft.setCursor(35,275);
              tft.print(Th1_Calc_2);

tft.setCursor(95,275);
              tft.print(Th2_Calc_2);

tft.setCursor(155,275);
              tft.print(Th3_Calc_2);

tft.setTextSize(1);
tft.setCursor(275,290);
              tft.print(C3_Result);

//choose1 if
if(digitalRead(right)==LOW && choose1==6){
          choose1=0;
    }
else if (digitalRead(right)==LOW ){
         choose1=choose1+1;
 }
              if (choose1==0) {
tft.drawRoundRect(20, 50, 50, 40, 8, GREEN);

               }
else if(choose1==1 ){
```

```
tft.drawRoundRect(80, 50, 50, 40, 8, GREEN);

                  }
else if(choose1==2 ){

tft.drawRoundRect(140, 50, 50, 40, 8, GREEN);

                    }
else if(choose1==3 ){

tft.drawRoundRect(240, 50, 70, 40, 8, RED);
tft.drawRoundRect(380, 50, 70, 40, 8, WHITE);
tft.drawRoundRect(240, 120, 70, 40, 8, WHITE);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);
                  }
else if(choose1==4 ){
tft.drawRoundRect(240, 50, 70, 40, 8, WHITE);
tft.drawRoundRect(380, 50, 70, 40, 8, RED);
tft.drawRoundRect(240, 120, 70, 40, 8, WHITE);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);

                    }

else if(choose1==5 ){
tft.drawRoundRect(240, 50, 70, 40, 8, WHITE);
tft.drawRoundRect(380, 50, 70, 40, 8, WHITE);
tft.drawRoundRect(240, 120, 70, 40, 8, RED);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);


                  }
else if(choose1==6 ){

tft.drawRoundRect(240, 50, 70, 40, 8, WHITE);
```

```
tft.drawRoundRect(380, 50, 70, 40, 8, WHITE);
tft.drawRoundRect(240, 120, 70, 40, 8, WHITE);
tft.drawRoundRect(370, 275, 70, 40, 8, RED);

}

}
```

Εικόνα 4.38 Σελίδα αντίστροφου κινηματικού προβλήματος

Η συνάρτηση drawpoint περιέχει έξι μεταβλητές οι όποιες είναι οι γωνιές των δυο θέσεων που θα εκτελέσει ο βραχίονας. Η κύριες λειτουργιές είναι η παράγωγη της σελίδας pointtopoint (εικόνα 4.39) ,η εμφάνιση των τρεχόντων τιμών των γωνιών που θα επιλεχτούν και το εφέ της τρέχοντος επιλογής.

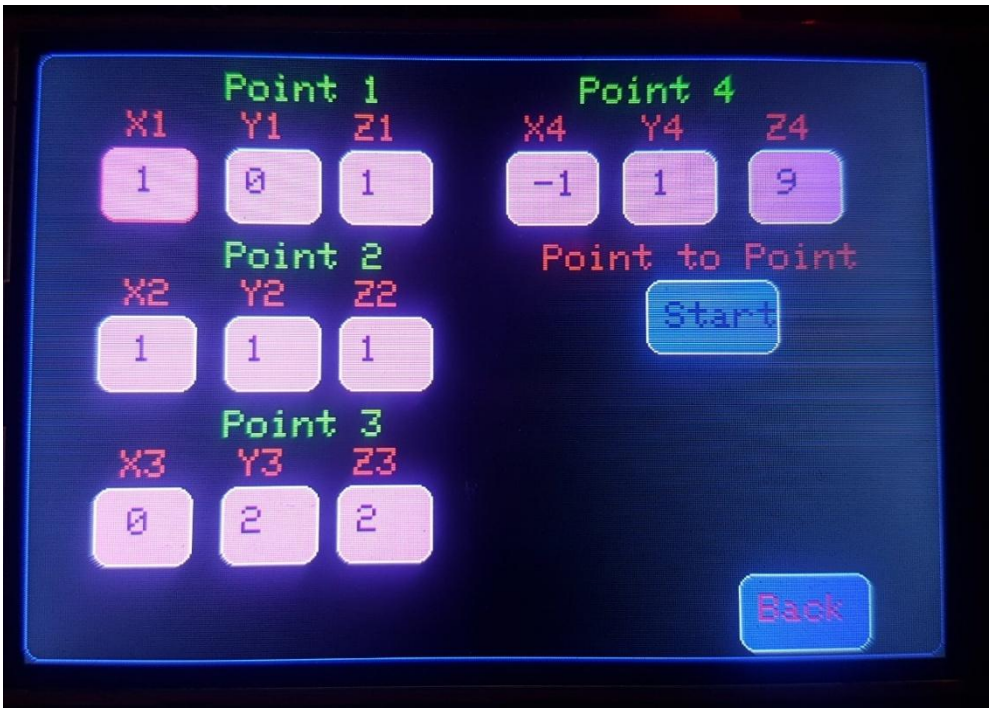
```
void drawpoint(int pp11,int pp12,int pp13,int pp21,int
pp22,int pp23){
  //Point box
tft.setTextSize(2);
tft.setCursor(160, 30);
tft.setTextColor(RED);
tft.print("Point to Point");
tft.fillRoundRect(205, 50, 70, 40, 8, BLUE);
tft.drawRoundRect(205, 50, 70, 40, 8, WHITE);
```

```
tft.setCursor(210, 62);
tft.setTextColor(BLACK);
  tft.print("Start");
  //Point
tft.setCursor(120, 120);
tft.setTextColor(GREEN);
tft.print("Point 1 Angles");
//th1-1 box
tft.setCursor(80, 140);
tft.setTextColor(RED);
  tft.print("Th1-1");
tft.fillRoundRect(75, 160, 70, 40, 8, MAGENTA);
tft.drawRoundRect(75, 160, 70, 40, 8, WHITE);
  //th1-2 box
tft.setCursor(160, 140);
tft.setTextColor(RED);
  tft.print("Th1-2");
tft.fillRoundRect(155, 160, 70, 40, 8, MAGENTA);
tft.drawRoundRect(155, 160, 70, 40, 8, WHITE);
  //th1-3 box
tft.setCursor(240, 140);
tft.setTextColor(RED);
  tft.print("Th1-3");
tft.fillRoundRect(235, 160, 70, 40, 8, MAGENTA);
tft.drawRoundRect(235, 160, 70, 40, 8, WHITE);
//Point
tft.setCursor(120, 210);
tft.setTextColor(GREEN);
tft.print("Point 2 Angles");
//th2-1 box
tft.setCursor(80, 240);
tft.setTextColor(RED);
  tft.print("Th2-1");
tft.fillRoundRect(75, 260, 70, 40, 8, MAGENTA);
```

```
tft.drawRoundRect(75, 260, 70, 40, 8, WHITE);
  //th2-2 box
tft.setCursor(160, 240);
tft.setTextColor(RED);
  tft.print("Th2-2");
tft.fillRoundRect(155, 260, 70, 40, 8, MAGENTA);
tft.drawRoundRect(155, 260, 70, 40, 8, WHITE);
  //th2-3 box
tft.setCursor(240, 240);
tft.setTextColor(RED);
  tft.print("Th2-3");
tft.fillRoundRect(235, 260, 70, 40, 8, MAGENTA);
tft.drawRoundRect(235, 260, 70, 40, 8, WHITE);
          //display vals
tft.setTextSize(2);
tft.setTextColor(BLACK);
              //th11
tft.setCursor(85,172);
              tft.print(pp11);
              //th12
tft.setCursor(170,172);
              tft.print(pp12);
             //th13
tft.setCursor(240,172);
              tft.print(pp13);
              //th21
tft.setCursor(85,272);
              tft.print(pp21);
             //th22
tft.setCursor(170,272);
              tft.print(pp22);
tft.setCursor(240,272);
              tft.print(pp23);
```

```
//if choose2
           if(digitalRead(right)==LOW && choose2==13)
          choose2=0;

else if(digitalRead(right)==LOW )
         choose2=choose2+1;

              if (choose2==0) {
tft.drawRoundRect(35, 50, 50, 40, 8, RED);

               }
else if(choose2==1 ){

tft.drawRoundRect(100, 50, 50, 40, 8, RED);
                }
else if(choose2==2 ){

tft.drawRoundRect(160, 50, 50, 40, 8, RED);

                    }
else if(choose2==3 ){

tft.drawRoundRect(35, 140, 50, 40, 8, RED);

                  }
else if(choose2==4 ){

tft.drawRoundRect(100, 140, 50, 40, 8, RED);

                    }

else if(choose2==5 ){
tft.drawRoundRect(160, 140, 50, 40, 8, RED);
                  }
```

```
else if(choose2==6 ){

tft.drawRoundRect(35, 230, 50, 40, 8, RED);
                }
else if(choose2==7 ){
tft.drawRoundRect(100, 230, 50, 40, 8, RED);

                    }
else if(choose2==8 ){

tft.drawRoundRect(160, 230, 50, 40, 8, RED);

                  }
else if(choose2==9 ){

tft.drawRoundRect(250, 50, 50, 40, 8, RED);
                    }
else if(choose2==10 ){
tft.drawRoundRect(313, 50, 50, 40, 8, RED);

                  }
else if(choose2==11 ){
tft.drawRoundRect(380, 50, 50, 40, 8, RED);
                                  }
else if(choose2==12 ){
tft.drawRoundRect(325, 120, 70, 40, 8, RED);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);
                    }
else if(choose2==13 ){
tft.drawRoundRect(325, 120, 70, 40, 8, WHITE);
tft.drawRoundRect(370, 275, 70, 40, 8, RED);
}
}
```



Εικόνα 4.39ΣελίδαPointtoPoint

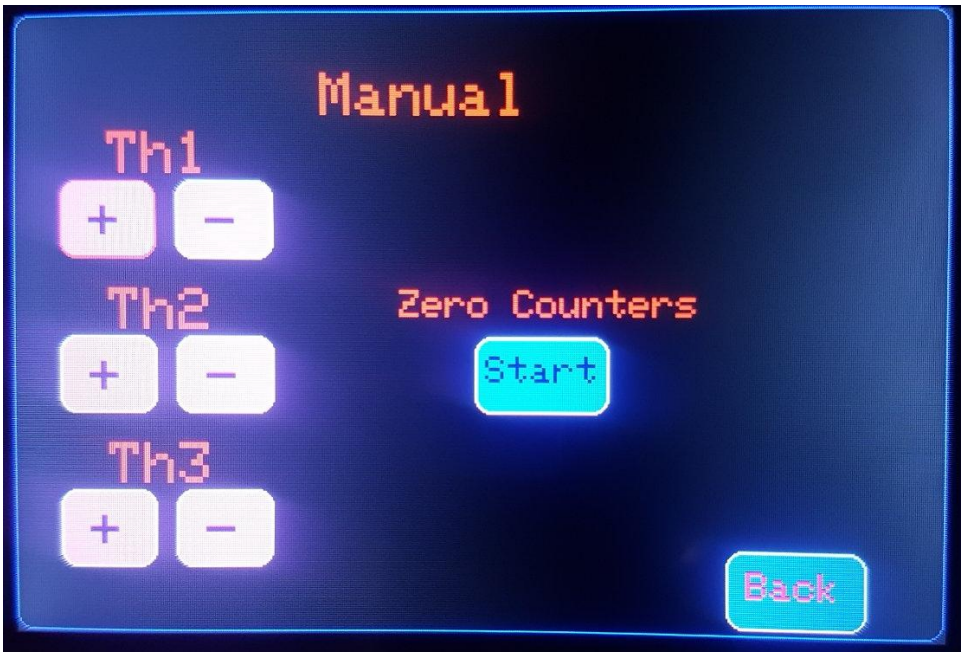
Η συνάρτηση drawmanual έχει τον ίδιο σκοπό όπως η προηγούμενες δυο. Να σχεδιάσει την σελίδα του χειροκίνητου(εικόνα 4.40) και να δημιουργήσει το εφέ της τρέχοντος επιλογής.

```
void drawmanual(){
tft.setTextSize(3);
tft.setCursor(160, 30);
tft.setTextColor(RED);
  tft.print("Manual");
  //th1+ box
tft.setCursor(50, 60);
tft.setTextColor(RED);
  tft.print("Th1");
tft.fillRoundRect(25, 85, 50, 40, 8, MAGENTA);
tft.drawRoundRect(25, 85, 50, 40, 8, WHITE);

tft.setCursor(40, 95);
tft.setTextColor(BLACK);
tft.print("+");
  //th1- box
tft.fillRoundRect(85, 85, 50, 40, 8, MAGENTA);
tft.drawRoundRect(85, 85, 50, 40, 8, WHITE);
tft.setCursor(100, 95);
tft.setTextColor(BLACK);
tft.print ("-");

  //th2+ box
tft.setCursor(50, 140);
tft.setTextColor(RED);
  tft.print("Th2");
tft.fillRoundRect(25, 165, 50, 40, 8, MAGENTA);
tft.drawRoundRect(25, 165, 50, 40, 8, WHITE);

tft.setCursor(40, 175);
```

```
tft.setTextColor(BLACK);
tft.print("+");
  //th2- box
tft.fillRoundRect(85, 165, 50, 40, 8, MAGENTA);
tft.drawRoundRect(85, 165, 50, 40, 8, WHITE);
tft.setCursor(100, 175);
tft.setTextColor(BLACK);
tft.print("-");

  //th3+ box
tft.setCursor(50, 220);
tft.setTextColor(RED);
  tft.print("Th3");
tft.fillRoundRect(25, 245, 50, 40, 8, MAGENTA);
tft.drawRoundRect(25, 245, 50, 40, 8, WHITE);

tft.setCursor(40, 255);
tft.setTextColor(BLACK);
tft.print("+");
  //th2- box
tft.fillRoundRect(85, 245, 50, 40, 8, MAGENTA);
tft.drawRoundRect(85, 245, 50, 40, 8, WHITE);
tft.setCursor(100, 255);
tft.setTextColor(BLACK);
tft.print("-");
  //zero box
tft.setTextSize(2);
tft.setCursor(200, 140);
tft.setTextColor(RED);
tft.print("Zero Counters");
tft.fillRoundRect(240, 165, 70, 40, 8, BLUE);
tft.drawRoundRect(240, 165, 70, 40, 8, WHITE);
tft.setCursor(245, 175);
tft.setTextColor(BLACK);
```

```
  tft.print("Start");

//choose3 if
           if(digitalRead(right)==LOW && chose3==7)
          chose3=0;

else if(digitalRead(right)==LOW )
         chose3=chose3+1;

              if (chose3==0) {
tft.drawRoundRect(25, 85, 50, 40, 8, RED);

               }
else if(chose3==1 ){
tft.drawRoundRect(85, 85, 50, 40, 8, RED);
                  }
else if(chose3==2 ){

tft.drawRoundRect(25, 165, 50, 40, 8, RED);

                    }
else if(chose3==3 ){

tft.drawRoundRect(85, 165, 50, 40, 8, RED);

                  }
else if(chose3==4 ){

tft.drawRoundRect(25, 245, 50, 40, 8, RED);

                    }

else if(chose3==5 ){
```

```
tft.drawRoundRect(85, 245, 50, 40, 8, RED);

                  }
else if(chose3==6 ){
tft.drawRoundRect(25, 85, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 85, 50, 40, 8, WHITE);
tft.drawRoundRect(25, 165, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 165, 50, 40, 8, WHITE);
tft.drawRoundRect(25, 245, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 245, 50, 40, 8, WHITE);
tft.drawRoundRect(240, 165, 70, 40, 8, RED);
tft.drawRoundRect(370, 275, 70, 40, 8, WHITE);
                    }
else if(chose3==7 ){
tft.drawRoundRect(25, 85, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 85, 50, 40, 8, WHITE);
tft.drawRoundRect(25, 165, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 165, 50, 40, 8, WHITE);
tft.drawRoundRect(25, 245, 50, 40, 8, WHITE);
tft.drawRoundRect(85, 245, 50, 40, 8, WHITE);
tft.drawRoundRect(240, 165, 70, 40, 8, WHITE);
tft.drawRoundRect(370, 275, 70, 40, 8, RED);
}

  }
```



Εικόνα 4.40 Σελίδα Manual

Στην συνάρτηση Setup γίνεται η εγκατάσταση των παραπάνω pins ,μεταβλητών και στοιχείων βιβλιοθηκών που θα χρησιμοποιηθούν όπως η επικοινωνία και η οθόνη. Αυτή η συνάρτηση τρέχει μια φορά στην αρχή.

```
voidsetup() {
```

Δήλωση ψηφιακών pins. Τα pins του πληκτρολογίου δηλώνονται ως PULLUP δηλαδή θα έχουν σε κατάσταση ηρεμίας 5V και θα ενεργοποιούνται όταν συνδεθούν στο μηδέν

```
pinMode(MAX485_RE_NEG, OUTPUT);
  // Init in receive mode
digitalWrite(MAX485_RE_NEG, 0);
pinMode(enter, INPUT_PULLUP);
pinMode(plusup, INPUT_PULLUP);
pinMode(minusdown, INPUT_PULLUP);
pinMode(right, INPUT_PULLUP);
```

Σε αυτό το σημείο γίνεται η δήλωση της επικοινωνίας Modbus . Δηλαδή το baudrate είναι 19200bps, η διεύθυνση του plc ο σταθμός 1 και γίνεται ο ορισμός πως θα ενεργοποιούνται η συνάρτηση για την ενεργοποίηση και απενεργοποίηση της επικοινωνίας.

```
Serial.begin(19200);
  // Modbus slave ID 1
node.begin(1, Serial);
  // Callbacks allow us to configure the RS485 transceiver
correctly
node.preTransmission(preTransmission);
node.postTransmission(postTransmission);
```

Τέλος γίνεται μηδενισμός της οθόνης, μετά γίνεται αναγνώριση των διαστάσεων της π.χ. 420x380 και μετά εμφανίζεται η αρχική σελίδα της οθόνης.

```
//    TFT
Serial.print(F("Finding  TFT  : "));
tft.reset();
uint16_t identifier = tft.readID();
  Serial.print(identifier, HEX);
tft.begin(identifier);
tft.setRotation(1);
tft.fillScreen(BLACK);
```

```
delay(100);
drawHomeScreen();
delay(100);
}
```

Η συνάρτηση loop τρέχει σε κάθε κύκλο που εκτελεί ο Arduino. Η loop περιέχει τις λειτουργιές τις κάθε οθόνης και κουμπιού που υπάρχει. Μόνο για την αρχική σελίδα περιέχει και το εφέ την επιλογής. Η λογική είναι ότι κάθε σελίδα και επιλογή έχει αρίθμηση. Με αποτέλεσμα να γίνονται οι κατάλληλες συγκρίσεις ώστε να εκτελείται η ανάλογη λειτουργία

```
void loop() {
   //main page
if(currentpage==0){
    if(digitalRead(plusup)==LOW && choose==0)
      choose=2;
else if(digitalRead(plusup)==LOW )
     choose=choose-1;
    if(digitalRead(minusdown)==LOW && choose==2)
      choose=0;
else if(digitalRead(minusdown)==LOW )
     choose=choose+1;
           if (choose==0) {
tft.drawRoundRect(60, 80, 360, 40, 8, GREEN);
tft.drawRoundRect(60, 140, 360, 40, 8, WHITE);
tft.drawRoundRect(60, 200, 360, 40, 8, WHITE);
         }
else if(choose==1 ){
tft.drawRoundRect(60, 140, 360, 40, 8, GREEN);
tft.drawRoundRect(60, 80, 360, 40, 8, WHITE);
tft.drawRoundRect(60, 200, 360, 40, 8, WHITE);
          }
```

```
else if(choose==2 ){
tft.drawRoundRect(60, 200, 360, 40, 8, GREEN);
tft.drawRoundRect(60, 140, 360, 40, 8, WHITE);
tft.drawRoundRect(60, 80, 360, 40, 8, WHITE);
            }
if (choose==0 && digitalRead(enter)==LOW){
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
              choose1=0;
            currentpage=currentpage+1;
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
drawinverse(u1,u2,u3);
delay(100);
           }
else if (choose==1 && digitalRead(enter)==LOW){
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
              choose2=0;
            currentpage=currentpage+2;
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
drawpoint(tp11,tp12,tp13,tp21,tp22,tp23,tp31, tp32, tp33,
tp41, tp42, tp43);              }
else if (choose==2 && digitalRead(enter)==LOW){
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
              chose3=0;
            currentpage=currentpage+3;
```

```
tft.fillScreen(BLACK);
tft.drawRoundRect(0, 0, 479, 319, 8, BLUE);
drawBackButton();
drawmanual();
          }
delay(500);
 }
    //inverse page
if(currentpage==1){
if (choose1==0 && digitalRead(plusup)==LOW){
                  u1=u1+1;
                  }
else if (choose1==0 && digitalRead(minusdown)==LOW){
                   u1=u1-1;
                   }
if (choose1==1 && digitalRead(plusup)==LOW){
                  u2=u2+1;
                  }
else if (choose1==1 && digitalRead(minusdown)==LOW){
                   u2=u2-1;
                   }
if (choose1==2 && digitalRead(plusup)==LOW){
                  u3=u3+1;
                  }
else if (choose1==2 && digitalRead(minusdown)==LOW){
                   u3=u3-1;
                   }
if (choose1==3 && digitalRead(enter)==LOW){
                   Inverse_Calc(u1,u2,u3);
delay(500);
                   }
if (choose1==4 && digitalRead(enter)==LOW){
node.writeSingleCoil(0x003, 1);
                   }
```

```
if (choose1==5 && digitalRead(enter)==LOW){
node.writeSingleCoil(0x004, 1);
                    }
drawinverse(u1,u2,u3);
if (choose1==6 && digitalRead(minusdown)==LOW  ){
                       currentpage=0;
drawHomeScreen();
                    }
delay(500);
     }

  //ptp screen
if(currentpage==2){


if (choose2==0 && digitalRead(plusup)==LOW){
                  tp11=tp11+1;
                   }
else if (choose2==0 && digitalRead(minusdown)==LOW){
                    tp11=tp11-1;
                   }

if (choose2==1 && digitalRead(plusup)==LOW){
                   tp12=tp12+1;
                   }
else if (choose2==1 && digitalRead(minusdown)==LOW){
                   tp12=tp12-1;
                    }
if (choose2==2 && digitalRead(plusup)==LOW){
                   tp13=tp13+1;
                   }
else if (choose2==2 && digitalRead(minusdown)==LOW){
                   tp13=tp13-1;
                    }
```

```
if (choose2==3 && digitalRead(plusup)==LOW){
                tp21=tp21+1;
                 }
else if (choose2==3 && digitalRead(minusdown)==LOW){
                  tp21=tp21-1;
                  }

if (choose2==4 && digitalRead(plusup)==LOW){
                 tp22=tp22+1;
                 }
else if (choose2==4 && digitalRead(minusdown)==LOW){
                 tp22=tp22-1;
                  }
if (choose2==5 && digitalRead(plusup)==LOW){
                 tp23=tp23+1;
                 }
else if (choose2==5 && digitalRead(minusdown)==LOW){
                 tp23=tp23-1;
                  }

if (choose2==6 && digitalRead(plusup)==LOW){
                tp31=tp31+1;
                 }
else if (choose2==6 && digitalRead(minusdown)==LOW){
                  tp31=tp31-1;
                  }

if (choose2==7 && digitalRead(plusup)==LOW){
                 tp32=tp32+1;
                 }
else if (choose2==7 && digitalRead(minusdown)==LOW){
                 tp32=tp32-1;
                  }
if (choose2==8 && digitalRead(plusup)==LOW){
```

```
                    tp33=tp33+1;
                    }
else if (choose2==8 && digitalRead(minusdown)==LOW){
                    tp33=tp33-1;
                     }
if (choose2==9 && digitalRead(plusup)==LOW){
                   tp41=tp41+1;
                    }
else if (choose2==9 && digitalRead(minusdown)==LOW){
                     tp41=tp41-1;
                     }

if (choose2==10 && digitalRead(plusup)==LOW){
                    tp42=tp42+1;
                    }
else if (choose2==10 && digitalRead(minusdown)==LOW){
                    tp42=tp42-1;
                     }
if (choose2==11 && digitalRead(plusup)==LOW){
                    tp43=tp43+1;
                    }
else if (choose2==11 && digitalRead(minusdown)==LOW){
                    tp43=tp43-1;
                     }
if (choose2==12 && digitalRead(enter)==LOW){

point_point(tp11,tp12,tp13,tp21,tp22,tp23,tp31, tp32, tp33,
tp41, tp42, tp43);
                     }

drawpoint(tp11,tp12,tp13,tp21,tp22,tp23,tp31, tp32, tp33,
tp41, tp42, tp43);

if (choose2==13 && digitalRead(minusdown)==LOW ){
```

```
                    currentpage = 0;
drawHomeScreen();
                    }


 }

//manual screen
if(currentpage==3){

if (chose3==0 && digitalRead(plusup)==LOW){
node.writeSingleCoil(0x005, 1);
          }
else if (chose3==0 && digitalRead(plusup)==HIGH){
node.writeSingleCoil(0x005, 0);
             }
if (chose3==1 && digitalRead(minusdown)==LOW){
node.writeSingleCoil(0x006, 1);
                  }
else if (chose3==1  && digitalRead(minusdown)==HIGH){
node.writeSingleCoil(0x006, 0);
          }
if (chose3==2 &&  digitalRead(plusup)==LOW){
node.writeSingleCoil(0x007, 1);
                  }
else if (chose3==2 && digitalRead(plusup)==HIGH){
node.writeSingleCoil(0x007, 0);
               }
if (chose3==3 &&  digitalRead(minusdown)==LOW){
node.writeSingleCoil(0x008, 1);
                  }
else if (chose3==3 && digitalRead(minusdown)==HIGH){
node.writeSingleCoil(0x008, 0);
                }
```

```
if (chose3==4 &&  digitalRead(plusup)==LOW){
node.writeSingleCoil(0x009, 1);
                  }
else if (chose3==4 && digitalRead(plusup)==HIGH){
node.writeSingleCoil(0x009, 0);
                 }
if (chose3==5 &&  digitalRead(minusdown)==LOW){
node.writeSingleCoil(0x00a, 1);
                  }
else if (chose3==5 &&  digitalRead(minusdown)==HIGH){
node.writeSingleCoil(0x00a, 0);
                 }
if (chose3==6 &&  digitalRead(plusup)==LOW){
node.writeSingleCoil(0x002, 1);
                  }
else if (chose3==6 &&  digitalRead(plusup)==HIGH) {
node.writeSingleCoil(0x002, 0);
                 }
drawmanual();
if (chose3==7 && digitalRead(minusdown)==LOW ){
currentpage = 0;
                   drawHomeScreen();
                   }
   }

 delay(200);
}
```

4.5 Επεξήγηση GUI

Σε αυτή την ενότητα θα γίνει επεξήγηση της λειτουργίας του μενού της οθόνης. Στην αρχική σελίδα(εικόνα 4.36) υπάρχουν τρεις επιλογές όπου η τρέχων επιλογή έχει πράσινο περίγραμμα και αυτό ισχύει εξίσου για τα επόμενα.

Inverse kinematics

Αν επιλεγεί η Inverse kinematics(εικόνα 4.36) δίνεται η δυνατότητα να δηλωθούν οι συντεταγμένες x-y-z,όταν βρίσκεται η επιλογή στην αντίστοιχη συντεταγμένη μπορεί να αυξηθεί η επιλογή κατά 1 με το κουμπί Up και αντίστοιχα να μειωθεί κατά 1 με το κουμπί Down. Η επιλογή CalcThetas υπολογίζει τις γωνιές και στέλνει τα αντίστοιχα βήματα στο plc. Η επιλογή Gox-y-z εκτελεί την κίνηση που έχει υπολογιστεί προηγουμένως. Η επιλογή GoHome πηγαίνει τον βραχίονα στην θέση μηδέν που έχει οριστεί. Τέλος η επιλογή Back επιστρέφει στην αρχική οθόνη πατώντας το Down.

Point to Point

Στην επιλογή Point-to-point(εικόνα 4.37) δίνεται η δυνατότητα να δηλωθούντέσσερασημείαόπου το καθένα έχει τρεις συντεταγμένες (X-Y-Z)έπειτα πατώντας enter στην επιλογή startεκτελείτε διαδοχική κίνηση από τον βραχίονα ώστε να βρει το κάθε σημείο. Οι τιμές των συντεταγμένων μπορούν να αυξηθούν και να μειωθούν κατά 1 με τα αντίστοιχα κουμπιά Up και Down.

Manual

Η επιλογή Manual(εικόνα 4.38) έχει την δυνατότητα να κινηθεί κάθε σύνδεσμος ξεχωριστά και να γίνει αρχικοποίηση της καινούριας θέσης άμα χρειαστεί. Η κίνηση πετυχαίνετε αν η τρέχων επιλογή βρίσκεται στον αντίστοιχο σύνδεσμο και προς ποια φορά είναι κίνηση (συν-πλην). Τότε με το κουμπί up ενεργοποιείται η κίνηση για θετική φορά και με το down γίνεται η κίνηση στην αρνητική φορά όταν η επιλογή βρίσκεται σε κάποια θέση πλην. Εκτελείται η κίνηση όσο είναι πατημένο το αντίστοιχο κουμπί.

5. Συμπεράσματα, μελλοντικές βελτιώσεις , κόστος κατασκευής

Υστέρα από αρκετές δόκιμες ο στόχος της εφαρμογής επιτεύχθηκε. Η συγκεκριμένη εφαρμογή είναι για εκπαιδευτικούς σκοπούς αλλά στα θεμέλια της έχει βασικές αρχές και λογικές που χρησιμοποιούνται στην βιομηχανία.

Βασικότερες δυσκολίες υπήρξαν στην επικοινωνία Arduino-PLC διότι δεν υπήρχε αντίστοιχη υλοποίηση στο διαδίκτυο για το συγκεκριμένο τύπο PLC. Επίσης πολλές βιβλιοθήκες Modbus για το Arduino δεν εκτελούνταν σωστά. Μια δυσκολία ήταν το αποτέλεσμα από το αντίστροφο κινηματικό πρόβλημα που δεν συμβάδιζαν με την πραγματικότητα και χρειαζόταν τροποποίηση αυτών. Τέλος αρκετά μεγάλη δυσκολία ήταν ο σχεδιασμός του μενού αλλά και ο περιορισμός της μνήμης.

Μελλοντικές βελτιώσεις μπορούν να γίνουν στην αλλαγή του Arduino διότι έχει περιορισμένες δυνατότητες, αλλά και η αλλαγή της οθόνης σε touchscreen και προσθήκη ενός συνδέσμου ακόμη ώστε να αποκτήσει ο βραχίονας ένα ακόμα βαθμό ελευθερίας.

Κόστος Υλικών

Υλικό	**Τιμή(ευρώ)**
Arduino UNO	4.5
Mitsubishi FX3U 24MR	42
3 x Stepper motor Nema 17	52.11
MaX485 RS485 TTL shield RS 485	2.8
Screw Shield	2.65
TFT 3.5 Inch 380x420	16.32
Καλώδια-κλέμες	5
3D Εκτύπωση βραχίονα	80
Servo motor	3.30
3x TB6600 Stepper Drivers	20.10
Τροφοδοτικό 24Volt 3A	4.74
Σύνολο	**233.52**

Βιβλιογραφία

[1] Lahart, Justin (2009-11-27). «Taking an Open-Source Approach to Hardware». The Wall Street Journal.

[2] http://infolab.stanford.edu/pub/voy/museum/pictures/display/Calculators.htm

[3] https://ilektroaytomatismoi.blogspot.com/2018/07/blog-post_4.html

[4] Li Z, Ge SS, Wang Z. Robust adaptive control of coordinated multiple mobile manipulators. Mechatronics. 2008

[5] http://www.academia.edu/9165706/Forward_and_inverse_Kinematics_complete_solutions_3DOF_good_reference_for_CrustCrawler_Smart_Arm_Users

[6] http://centerforneurotech.org/sites/default/files/Lesson%208.pdf

[7] https://cy.ipc2u.com/articles/articles-and-reviews/to-modbus-rtu-me-apla-logia-me-leptomereis-perigrafes-kai-paradeigmata/

[8] https://howtomechatronics.com/how-it-works/how-servo-motors-work-how-to-control-servos-using-arduino/

[9] https://www.getcert.gr/pos-leitourgei-to-stepper-motor/

[10] https://books.google.gr/books?id=5PDx2Q9Ea_YC&pg=PA248&lpg=PA248&dq=ratio+gearbox+degrees+arm+robot+calculator&source=bl&ots=uDuwK9Tpiz&sig=ACfU3U0nl1lHjzBF_Tq74kZFGLDoZmkCzw&hl=el&sa=X&ved=2ahUKEwidxvG9x4npAhXGSEEAHTM3CVQQ6AEwCnoECAoQAQ#v=onepage&q=ratio%20gearbox%20degrees%20arm%20robot%20calculator&f=false

[11] https://www.thingiverse.com/Gear_Down_For_What/about

[12] https://github.com/Basilisvirus/Inverse-Kinematics-3-DOF-Robotic-arm

[13] ISO Standard 8373:1994, Manipulating Industrial Robots – Vocabulary

[14] J. J. Uicker, G. R. Pennock and J. E. Shigley, 2003, Theory of Machines and Mechanisms, Oxford University Press, New York.

[15] Φ.Ν. Κουμπούλης, Β.Γ. Μέρτζιος Εισαγωγή στη Ρομποτική Εκδόσεις Παπασωτηρίου Αθήνα 2002

[16] Π. Παπάζογλου, Σ.Π. Λιώνης Ανάπτυξη Εφαρμογών με το Arduino Εκδότης Τζιόλας 2017

[17] Ζ. Δουλγέρη Ρομποτική Κινηματική, Δυναμική και Έλεγχος Αρθρωτών Βραχιόνων Εκδότης Κριτική 2017

[18] Denis Collins-Eamonn Lane Προγραμματιζόμενοι Ελεγκτές Εκδόσεις Α. Τζιόλα Α.Ε. Θεσσαλονίκη 2000

[19] Ε. Δασκαλόπουλος, Γ.Κρανάς Βιομηχανικοί Αυτοματισμοί Και Προγραμματιζόμενοι Λογικοί Ελεγκτές PLC Εκδόσεις Ίων 2001

[20] John Ridley Mitsubishi FX Programmable Logic Controllers: Applications and Programming Newnes 2 edition 2004

[21] Β. Δ. Μπιτζιώνης Βιομηχανικές ηλεκτρικές εγκαταστάσεις Εκδόσεις Τζιόλα 2011