ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη συστήματος συγχρονισμού σερβοκινητήρα με έλεγχο θέσης και υλοποίηση στον αυτοματισμό μηχανουργικού τόρνου.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΑΡΟΥΖΟΣ Α. ΚΩΝΣΤΑΝΤΙΝΟΣ

Επιβλέπων : Γεώργιος Καμπουράκης
Επίκουρος καθηγητής Ε.Μ.Π.

Αθήνα, 21 Οκτωβρίου 2011

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη συστήματος συγχρονισμού σερβοκινητήρα με έλεγχο θέσης και υλοποίηση στον αυτοματισμό μηχανουργικού τόρνου.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΑΡΟΥΖΟΣ Α. ΚΩΝΣΤΑΝΤΙΝΟΣ

Επιβλέπων : Γεώργιος Καμπουράκης
Επίκουρος καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την .....

.....	.....	.....
Καμπουράκης Γ.	Καγιάφας Ε.	Κουκούτσης Η.
Επίκουρος Καθηγητής	Καθηγητής	Επίκουρος Καθηγητής

Αθήνα, 21 Οκτωβρίου 2011

.....

Καρούζος Κωνσταντίνος του Ανδρέα

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Πίνακας περιεχομένων

Πρόλογος

Η παρούσα διπλωματική εργασία αποτελεί το επιστέγασμα της ολοκλήρωσης των σπουδών μου στην σχολή των Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσοβίου Πολυτεχνείου. Το θεματικό περιεχόμενο της εργασίας και η εκπόνησή της εμπλέκονται με μια ευρύτητα αγαπημένων κλάδων της επιστήμης που σπουδάζω, όπως ο αυτόματος έλεγχος, τα μικροϋπολογιστικά συστήματα, η ηλεκτρονική, η επεξεργασία σήματος και η ανάπτυξη εφαρμογών. Επι πλέον, το κατασκευαστικό μέρος της εργασίας μου έδωσε την πολύτιμη εμπειρία της ανάπτυξης ενός φυσικού ηλεκτρομηχανικού συστήματος. Η διαδικασία αυτή με έφερε αντιμέτωπο με πληθώρα προκλήσεων δυσδιάκριτων κατα την θεωρητική μελέτη των πραγμάτων ενώ, όπως είναι φυσικό, οι λεπτομέρειες και τα μικροπροβλήματα είχαν την τιμητική τους. Ωστόσο, η αντιμέτώπιση τέτοιων πραγματικών προβλημάτων και η υλοποίηση μιας λειτουργικής εφαρμογής δίνει ιδιαίτερη ικανοποίηση. Γι'αυτούς τους λόγους η εκπόνηση της παρούσας εργασίας έγινε με μεγάλο ενδιαφέρον και όρεξη.

Σε αυτό το σημείο θα ήθελα να αποδώσω τα οφειλόμενα εύσημα σε όλους όσους συνετέλεσαν στην εκπόνηση της παρούσας εργασίας.

Κατ'αρχάς θα ήθελα να ευχαριστήσω θερμά τον κ. Καμπουράκη Γεώργιο, επίκουρο καθηγητή του Εθνικού Μετσοβίου Πολυτεχνείου που με την εμπειρία και την γνώση του με καθοδήγησε σωστά κατα την διεκπεραίωση της εργασίας αυτής, αλλα και για για την φιλοξενεία της διόλου μικρής και αθόρυβης κατασκευής στον χώρο του εργαστηρίου.

Δεν θα μπορούσα να παραλείψω τον τεχνικό του εργαστηρίου αισθητήρων κ. Ψαρρό Απόστολο, ο οποίος παραχώρησε τον εξοπλισμό του εργαστηρίου και προσέφερε τις γνώσεις και την βοήθειά του για το τύπωμα των πλακετών που φέρουν τα κυκλώματα του συστήματος.

Ιδιαίτερες ευχαριστίες οφείλω και στον υποψήφιο διδάκτωρα κ. Πιπερίδη Δημήτριο. Η μακρά εμπειρία και η εξειδικευμένες γνώσεις του επάνω στην ανάπτυξη εφαρμογών με μικροεπεξεργαστές βοήθησαν καταλυτικά στην αντιμετώπιση προβλημάτων που προέκυπταν κατα την διεξαγωγή της εργασίας.

Θα ήθελα επίσης να ευχαριστήσω θερμά τον φίλο Διονυσόπουλο Ιωάννη, σπουδαστή στην σχολή Μηχανολόγων μηχανικών για την πολύτιμη βοήθεια και υποστήριξη. Οι μηχανολογικές γνώσεις του, αλλα και η συμμετοχή του στην κατασκευή του συστήματος διευκόλυναν σημαντικά την εκπόνηση της εργασίας.

Τέλος, θα ήθελα να ευχαριστήσω απο καρδιάς την οικογένειά μου, την Αλωνιστιώτη Δέσποινα για τους φίλους Αντωνάκο Παναγιώτη και Ρόκκα Ηλία για την υποστηριξή τους όλο αυτό το διάστημα.

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται την μελέτη, σχεδιασμό και υλοποίηση ενός συστήματος ελέγχου θέσης για σερβοκινητήρα, ωστε η κίνησή του να εμφανίζει συγκεκριμένη σχέση με την κίνηση κάποιου άλλου άξονα αναφοράς. Η διαδικασία αυτή ονομάζεται «ηλεκτρονικό γρανάζωμα», καθώς προσομοιώνει την μηχανική σύνδεση δύο αξόνων με κάποια σχέση μετάδοσης. Το σύστημα δέχεται σαν είσοδο τα σήματα απο αισθητήρες κίνηση στους δύο άξονες, καθώς και την επιθυμητή σχέση μεταξύ τους. Επειτα απο υπολογισμούς οδηγεί τον κινητήρα με την κατάλληλη ισχύ ωστε να επιτευχθεί το ζητούμενο αποτέλεσμα. Ο έλεγχος εκτελείται σε πραγματικό χρόνο απο ελεγκτή τριών όρων (PID), υλοποιημένο σε λογισμικό. Το σύστημα υλοποιήθηκε και εφαρμόστηκε στον αυτοματισμό μηχανουργικού τόρνου, όπου η χρήση του επιτρέπει την κοπή σπειρωμάτων.

Ιδιαίτερη προσοχή έχει δοθεί κατα τον σχεδιασμό του συστήματος ωστε να είναι δυνατή η επιτευξη σχέσεων μετάδοσης με αρκετά δεκαδικά ψηφία αποφεύγοντας σφάλματα στρογγυλοποίησης και υπολογισμού. Η ζητούμενη σχέση μετάδοσης στην περίπτωση της εφαρμογής εισάγεται με την μορφή επιθυμητού βήματος σπειρώματος σε mm.

Το κείμενο της εργασίας έχει χωριστεί σε κεφάλαια, αντίστοιχα με την συλλογιστική και διαδικαστική πορεία που ακολουθήθηκε κατα την εκπόνησή της. Τα κεφάλαια αυτά είναι: Το Θεωρητικό υπόβαθρο των στοιχείων και εννοιών που χρησιμοποιήθηκαν, ο σχεδιασμός του συστήματος, η κατασκευή του και εφαρμογή στον τόρνο και, τέλος, η ανάλυση της λειτουργίας του. Πιο συγκεκριμένα:

Στο πρώτο κεφάλαιο αναλύονται θεωρητικά οι παρακάτω έννοιες: Οι ηλεκτρικοί κινητήρες με έμφαση στους κινητήρες ΣΡ, ο αυτόματος έλεγχος και οι σερβομηχανισμοί με έμφαση στον σερβοκινητήρα ΣΡ, ο ελεγκτής τριών όρων (PID), οι αισθητήρες περιστροφικής κίνησης, οι μετατροπείς συνεχούς τάσης, τα κυκλώματα οδήγησης κινητήρων ΣΡ, οι μικροελεγκτές με έμφασης στον μΕ AVR ATmega16, η δομή και η λειτουργία του (μηχανουργικού) τόρνου, καθώς και η δομή και τα χαρακτηριστικά των σπειρωμάτων.

Στο δεύτερο κεφάλαιο αναπτύσσεται η συλλογιστική πορεία υλοποίησης του εν λόγω συστήματος. Αρχικά μελετάται ο τρόπος με τον οποίο το σύστημά μας θα επεξεργάζεται αποδοτικά τα σήματα των αισθητήρων, ωστε να ελαχιστοποιήσουμε τα σφάλματα στους υπολογισμούς. Στην συνέχεια επιλέγονται και παρουσιάζονται τα στοιχεία τα οποία θα απαρτίζουν το σύστημα και η διασύνδεση και επικοινωνία μεταξύ τους. Σε αυτό το σημείο η μελέτη επικεντρώνεται στην εφαρμογή του συστήματος στον τόρνο, με την ανάλυση του αλγορίθμου λειτουργίας του συστήματος. Φσυικά, η συγκεκριμένη εφαρμογή εισάγει επι πλέον απαιτήσεις που χρειάζεται να ικανοποιηθούν και οι οποίες αναλύονται. Επειτα περιγράφεται η διεπαφή του χρήστη με το σύστημα. Τέλος παρουσιάζεται ο κώδικας που εκτελείται απο τον μικροελεγκτή αναλυτικά για κάθε διαδικασία του συστήματος.

Στο τρίτο κεφάλαιο παρουσιάζεται η κατασκευή του συστήματος. Αρχικά περιγράφεται η διαδικασία σχεδιασμού και τυπώματος της πλακέτας με το αντίστοιχο φωτογραφικό υλικό.Στην συνέχεια μελετώνται κάποιες παράμετροι λειτουργίας του κυκλώματος, όπως ο θόρυβος, οι παρεμβολές, η ισχύς που καταναλώνεται και οι απαιτήσεις ψύξης των στοιχείων ισχύος. Στην συνέχεια περιγράφεται η διαδικασία προγραμματισμού του μΕ με τον αναπτυχθέντα κώδικα. Τέλος, παρουσιάζεται η μηχανική κατασκευή και σύνθεση του συστήματος με το ανάλογο φωτογραφικό υλικό.

Στο τέταρτο και τελευταίο κεφάλαιο γίνεται η ανάλυση της λειτουργίας του συστήματος. Μελετώνται οι δυνάμεις που ασκούνται στο μηχανικό σύστημα και εξετάζονται ενδεχόμενες μή ελέγξιμες παράμετροι λειτουργίας. Στην συνέχεια γίνεται εκτενής ανάλυση της ρύθμισης του ελεγκτή ωστε να ταιριάξει στο συγκεκριμένο ηλεκτρομηχανικό σύστημα και να επιφέρει τα ζητούμενα αποτελέσματα. Η διαδικασία που

ακολουθείται είναι προσεγγιστική αναγνώριση του συστήματος μέσω δεδομένων που συλλέγονται απο τον μΕ και αναλυτική επεξεργασία τους. Έπειτα χρησιμοποιούνται σχεδιαστικές μέθοδοι για την ρύθμιση του ελεγκτή. Τέλος, γίνονται μετρήσεις και έλεγχος της λειτουργίας του συστήματος, καθώς και των σφαλμάτων.

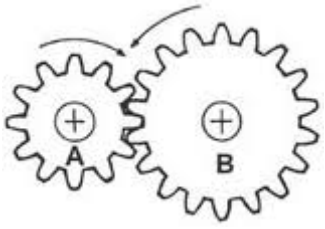
Κεφάλαιο 1 : Θεωρητικό υπόβαθρο

1.1 Εισαγωγή – Μηχανικό ισοδύναμο

Ας θεωρήσουμε ένα μηχανικό σύστημα με στοιχεία που εκτελούν περιστροφική κίνηση. Το σύστημα αυτό αποτελείται τυπικά απο έναν η περισσότερους κινητήρες, οι οποίοι παρέχουν την κινητική ενέργεια στο σύστημα. Στην συνέχεια, η παρεχόμενη ισχύς μεταφέρεται μέσω διαφόρων μηχανισμών και δομών στα επι μέρους κινούμενα μέρη. Οι μεταφορά της ισχύος πραγματοποιείται συνήθως μέσω γραναζιών, ιμάντων και τροχαλιών, αλυσίδων, δίσκων τριβής κ.α. Κατα την μεταφορά της ισχύος μπορούμε να κάνουμε διάφορες μετατροπές στον τρόπο με τον οποίο αυτή παρέχεται. Προφανώς η ποσότητα της ισχύος είναι συγκεκριμένη και καθορίζεται απο την παρεχόμενη ισχύ του κινητήρα. Η ισχύς αυτή τροφοδοτεί τα δίαφορα σημεία κίνησης και τις απώλειες στις επιφάνειες τριβής. Σε έναν περιστρεφόμενο άξονα, η ισχύς ισούται με το γινόμενο της ταχύτητας περιστροφής με την ροπή στρέψης που ασκείται στον άξονα αυτόν: $P = \omega * T$ (1). Επομένως, για δεδομένη ισχύ τα μεγέθη γωνιακής ταχύτητας και ροπής είναι αντιστρόφως ανάλογα. Ρυθμίζοντας τα χαρακτηριστικά των διάφορων στοιχείων μετάδοσης κίνησης, μπορούμε να μεταβάλλουμε κάποιο απο αυτά τα μεγέθη, εις βάρος φυσικά του άλλου.

Ας μελετήσουμε πιο συγκεκριμένα την περίπτωση των γραναζιών. Τα γρανάζια δεν είναι παρα μεταλλικοί δίσκοι με οδοντωτή περιφέρεια. Το είδος και ο τύπος των οδόντων της περφέρειάς τους διαφέρουν ανάλογα με την χρήση και το υλικό κατασκευής. Μπορούμε να βρούμε πάρα πολλές παραλλαγές στον σχεδιασμό των οδόντων, άλλες με σκοπό την ενίσχυση της αντοχής, αλλες την μείωση του θορύβου κ.α... Η χρήση των οδόντων αυτών δεν είναι άλλη παρα η αύξηση της τριβής μεταξύ δύο γραναζιών που εφάπτονται και η αποτροπή της μεταξύ τους ολίσθησης.

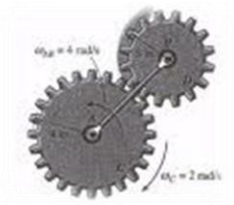
Το αποτέλεσμα της σύμπλεξης αυτής είναι πως οι δύο δίσκοι στο σημείο επαφής έχουν την ίδια εφαπτομενική ταχύτητα.



Σχήμα 1.1

Η εφαπτομενική ταχύτητα του κάθε δίσκου συνδέεται με την γωνιακή ταχύτητα του αντίστοιχου άξονα σύμφωνα με τον τύπο $\upsilon_{\varepsilon\phi}=\omega * R$ (2) . Επομένως, ρυθμίζοντας την

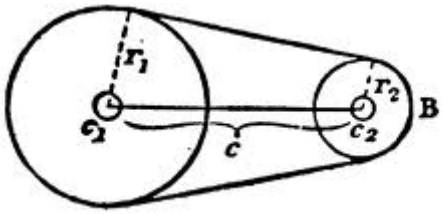
ακτίνα του κάθε γραναζιού μπορούμε να ρυθμίσουμε τον λόγο των δύο γωνιακών ταχυτήτων.



Σχήμα 1.2

Παραδείγματος χάρη, έστω οτι έχουμε δύο συμπλεκόμενα γρανάζια με ακτίνες r_1 και r_2. Η εφαπτομενική ταχύτητα στο σημείο επαφής είναι κοινή: $υ_{εφ1} = υ_{εφ2}$. Επομένως, απο τον τύπο (2) προκύπτει πως $\frac{ω1}{ω2} = \frac{r2}{r1} = n$. Αυτός ο λόγος είναι γνωστός ώς λόγος ή σχέση μετάδοσης. Ο αντίστροφος του λόγου αυτού είναι ίσος με τον λόγω των ροπών στους αντίστοιχους άξονες: $\frac{T1}{T2} = \frac{r1}{r2}$, ωστε με εφαρμογή του τύπου (1) να ισχύει οτι $\frac{P1}{P2} = 1$.

Ομοίως, με την χρήση ιμάντων και τροχαλίων, η περιφερειακή ταχύτητα μεταδίδεται μέσω του ιμάντα ο οποίως λόγω τριβής με τις τροχαλίες μεταφέρει την ισχύ. Και πάλι, ο λόγος ακτίνων των τροχαλιών καθορίζει την σχέση μετάδοσης.



Σχήμα 1.3

Ας επιστρέψουμε στα γρανάζια. Συγκεκριμένα στην περίπτωση τους, οι διαστάσεις των οδόντων σε όλα τα εφαπτόμενα στοιχεία πρέπει να είναι οι ίδιες, ωστε να επιτύχει η σωστή μεταξύ τους σύμπλεξη.Επομένως, οι ακτίνες των δύο γραναζιώς δεν μπορούν να πάρουν οποιεσπήποτε τιμές, παρά μόνον αυτές που επιτρέπουν ακέραιο αριθμό οδόντων. Ετσι, η σχέση μετάδοσης στην περίπτωση των γραναζιών ανάγεται στον λόγο του πλήθους οδόντων των δύο γραναζιών. Παραδειγματος χάριν, η σύμπλεξη δύο γραναζιών με 31 και 57 οδόντες αντίστοιχα, εισάγει στο σύστημα έναν λόγο μετάδοσης $\frac{57}{31}$.

Συνεπώς, προκειμένου να ρυθμίσουμε την σχέση μετάδοσης μεταξύ δύο αξόνων κίνησης, αρκεί να χρησιμοποιήσουμε γρανάζια με τους κατάλληλους αριθμούς οδόντων. Η σχέση μετάδοσης που επιτυγχάνουμε, μας εξασφαλίζει σταθερή σχέση μεταξύ των δύο γωνιακών ταχυτήτων και απουσία ολίσθησης μεταξύ των γραναζιών. Επομένως , η σχεση αυτή περιγράφει και των λόγο των γωνιακών μετατοπίσεων των δύο αξόνων ανα πάσα χρονική στιγμή. Πρόκειται, δηλαδή, για μηχανικό έλεγχο θέσης και όχι μονο ταχύτητας.

Το ζητούμενο της εργασίας αυτής είναι η εφαρμογή μίας σχέσης μετάδοσης μεταξύ δύο αξόνων χωρίς την μηχανική σύνδεσή τους.

Στην περίπτωση των γραναζιών σε κάθε σύνδεση μεταξύ δύο στοιχείων, έχουμε ροή ισχύος απο ένα γρανάζι , έστω το Γ1, προς αυτό με το οποίο έρχεται σε επαφή, έστω

Γ2.Προφανώς στην υλοποίηση μας δεν μπορούμε να μιλάμε για ουσιαστική ροή ισχύος, αφού δεν υπάρχει μηχανική σύνδεση, και γενικά σύνδεση ισχύος. Μπορούμε ωστόσο να ονομάζουμε τον άξονα του Γ1 ως “master” άξονα, και τον άξονα του Γ2 ώς “slave”.

Απο τον master άξονα χρειαζόμαστε μόνο την πληροφορία για την γωνιακή θέση και ταχύτητα ανα πάσα στιγμή.

Στον slave άξονα χρειάζεται να παράσχουμε ισχύ, επομένως να χρησιμοποιήσουμε έναν κινητήρα. Ο κινητήρας αυτός πρέπει να μπορεί να ελεγχθεί ως προς την θέση και την ταχύτητα. Επομένως, μιλάμε για έναν «σερβοκινητήρα», ο οποίος, όπως θα αναπτυχθεί αργότερα, πρόκειται για έναν κινητήρα με κάποιο στοιχείο ανάδρασης στον άξονα κίνησης. Ο κινητήρας αυτός πρέπει να οδηγηθεί απο κατάλληλο κύκλωμα. Επίσης, είναι απαραίτητο να γίνει κάποια επεξεργασία των πληροφοριών απο τους αισθητήρες των δύο αξόνων, ωστε η οδήγηση του μοτέρ να έχει ως αποτέλεσμα την ζητούμενη σχέση μετάδοσης.

1.2 Ηλεκτρικοί κινητήρες

1.2.1 Χαρακτηριστικά - κατηγορίες

Με τον όρο «κινητήρας» αναφερόμαστε σε οποιαδήποτε διάταξη η οποία δέχεται ενέργεια σε κάποια μορφή και την μετατρέπει σε κινητική ενέργεια.Συνήθως η κινητική ενέργεια εμφανίζεται ως στροφική κίνηση κάποιου άξονα.

Στην περίπτωση των ηλεκτρικών κινητήρων, η ενέργεια εισάγεται σε ηλεκτρική μορφή. Τα είδη και οι τεχνολογίες των ηλεκτρικών μηχανών τους χωρίζουν σε αρκετές κατηγορίες, όπως θα δούμε παρακάτω. Ωστόσο, κάποια κοινά χαρακτηριστικά τα συναντούμε στην πλειοψηφία των μηχανών. Κοινός παρονομαστής όλων των ηλεκτρικών κινητήρων είναι οτι στην ηλεκτρομηχανική μετατροπή της ενέργειας παρεμβάλλεται κάποιο μαγνητικό κύκλωμα. Η μετατροπή,δηλαδη, περνάει και απο το στάδιο της μαγνητικής ενέργειας. Οι ηλεκτρικοι κινητήρες αποτελούνται, ως επι το πλείστον, απο δύο μέρη. Τον στάτη και τον δρομέα.

Στατης ονομάζεται το τμήμα της μηχανής το οποίο δεν κινείται ως προς την επιφάνεια πάκτωσης του κινητήρα.Περιλαμβάνει μηχανικά τμήματα και κάποια μαγνητικά στοιχεία. Αυτά μπορεί να είναι είτε μόνιμοι μαγνήτες , είτε αγώγιμα τυλίγματα.

Ο δρομέας είναι το περιστρεφόμενο τμήμα της μηχανής και φέρει τον κεντρικό άξονα κίνησης. Περιλαμβάνει και αυτός κάποια μαγνητικά στοιχεία, όπως στην περίπτωση του στάτη.

Ανάλογα με τον τρόπο με τον οποίο απορροφούν ηλεκτρική ενέργεια οι κινητήρες, χωρίζονται σε δύο βασικές κατηγορίες: τις μηχανές εναλλασσόμενου ρεύματος και τις μηχανές συνεχούς ρεύματος:

Οι ΜΗΧΑΝΕΣ ΕΡ τροφοδοτούνται απο πηγή ισχύος υπο εναλασσόμενη τάση και διαρρέονται κυρίως απο εναλασσόμενο ρεύμα. Χρησιμοποιούνται ευρέως σε βιομηχανικές αλλα και οικιακές εφαμρογές. Στις μηχανές ΕΡ υπάρχουν δύο βασικά τυλιγματα: το τύλιγμα τυμπάνου και το τύλιγμα πεδίου.Το ένα απο αυτά, ανάλογα την κατασκευή, είναι στον στάτη και το άλλο στον δρομέα.Οι μηχανές ΕΡ χωρίζονται με την σειρά τους σε δύο βασικές κατηγορίες, τις σύγχρονες και τις ασύγχρονες.

Σύγχρονες είναι οι μηχανες στις οποίες το τυλιγμα πεδίου διαρρέεται απο συνεχές ρεύμα, επομένως έχει σταθερούς μαγνητικούς πόλους.Σε κάποιες περιπτώσεις μάλιστα

χρησιμοποιείται μόνιμος μαγνήτης.Το τύλιγμα τυμπάνου διεγείρεται απο εναλασσόμενο ρεύμα και ο δρομέας κινείται με την σύγχρονη ταχύτητα, που εξαρτάται απο την συχνότητα του ρεύματος και κάποιες κατασκευαστικές σταθερές.

Στις ασύγχρονες μηχανές τροφοδοτείται μόνο το τύλιγμα τυνμπάνου, ενώ το τύλιγμα πεδίου είναι βραχυκυκλωμένο, ή συνδεδεμένο σεσταθερή αντίσταση.Εφ'όσον δεν τροφοδοτείται απ'ευθείας, τα ρεύματα που το διαρρέουν επάγονται απο τον στάτη, εξ'ού και το όνομα. Ο δρομέας κινείται με ταχύτητα κατάτι μικρότερη της σύγχρονης, εμφανίζοντας την λεγόμενη ολίσθηση.

Οι μηχανές ΕΡ επίσης κατηγοριοποιούνται και ανάλογα με τον αριθμό και την συνδεσμολογία των τυλιγμάτων τους.Ετσι βλέπουμε τριφασικές, διφασικές και μονοφασικές μηχανές, σε σύνδεση αστέρα-τριγώνου, αστέρα-αστέρα κ.λπ.

Οι κινητήρες ΕΡ είναι αποδοτικοί και ευρέως χρησιμοποιούμενοι. Ωστόσο, ο έλεγχος της κίνησής τους απαιτεί έλεγχο της συχνότητας της τάσης τροφοδοσίας τους. Αυτή η διαδικασία χρειάζεται ειδικούς ηλεκτρονικούς ελεγκτές ισχύος, τους αντιστροφείς.

Οι μηχανές Συνεχούς Ρεύματος (ΣΡ) τροφοδοτούνται απο σταθερή τάση και απορροφούν συνεχές ρεύμα.Αποτελούνται απο δύο τυλίγματα, ή ένα τύλιγμα που αλληλεπιδρά με κάποιον μόνιμο μαγνήτη. Ανάλογα με την τεχνολογία λειτουργίας, οι κινητήρες ΣΡ χωρίζονται στις εξής κατηγορίες: κινητηρες με ψυκτρες και κινητήρες χωρίς ψήκτρες.

Οι κινητήρες χωρίς ψήκτρες χρησιμοποιούν μόνιμους μαγνήτες στον δρομέα, ενώ τα τυλίγματα διέγερσης βρίσκονται στον στάτη. Ετσι, δεν χρειάζεται αγώγιμη σύνδεση με τον δρομέα και οι κινητήρες αυτοί παρουσιάζουν μικρο κόστος συντήρησης και μεγάλο χρόνο ζωής. Ωστόσο, οι κινητήρες αυτοί δεν μπορούν να οδηγηθούν απ'ευθείας απο μία σταθερή τάση, παρά χρειάζεται κύκλωμα οδήγησης που παράγει συγκεκριμένα σήματα στα διάφορα τυλίγματα του κινητήρα. Ετσι, παρουσιάζουν αυξημένο κόστος κατασκευής.Διάφορα ειδη κινητήρων χωρίς ψύκτρες είναι ο σύγχρονος κινητήρας συνεχούς ρεύματος και ο βηματικός κινητήρας.

1.2.2 Κινητήρας ΣΡ με ψύκτρες

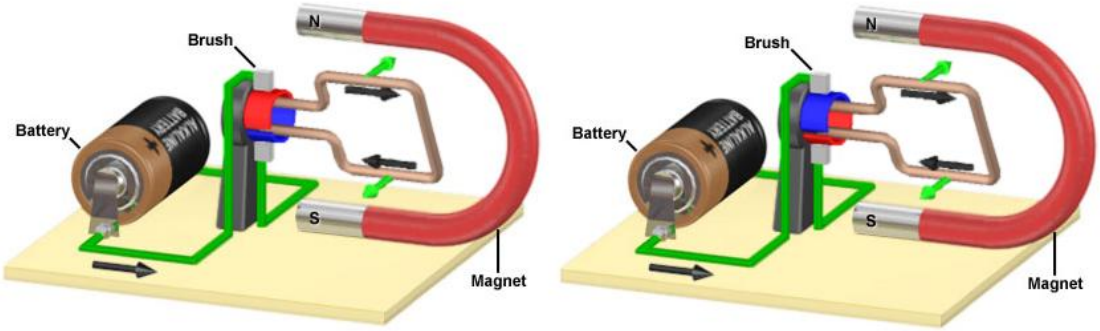
Οι κινητήρες με ψήκτρες είναι η πιο απλή μορφή κινητήρα συνεχούς ρεύματος. Λόγω της ύπαρξης των ψυκτρών, που είναι στοιχεία τριβής, οι κινητήρες αυτοί παρουσιάζουν περιορισμένο χρόνο ζωής και έχουν αυξημένες απαιτήσεις συντήρησης.Ωστόσο, ο έλεγχος αυτών τους γίνεται αρκετά εύκολα και η σχέση ταχύτητας/ροπής που παρουσιάζουν είναι ικανοποιητική.Γι'αυτόν τον λόγο χρησιμοποιούνται κατα κόρον σε εφαρμογές ελέγχου.

Ο κινητήρας συνεχούς ρεύματος, όπως όλοι οι κινητήρες αποτελείται απο τον στάτη και τον δρομέα. Ο στάτης παρέχει ένα σταθερό μαγνητικό πεδίο. Αυτό επιτυγχάνεται είτε με μόνιμους μαγνήτες, είτε με αγώγιμο τύλιγμα που ονομάζεται τύλιγμα πεδίου(ή διέγερσης).

Ο δρομέας περιλαμβάνει ένα τύλιγμα που λέγεται τύλιγμα οπλισμού.Το τύλιγμα αυτό πρέπει να τροφοδοτείται κατα την κίνηση του δρομέα, πράγμα που επιτυγχάνεται με την χρήση ψυκτρών.

Οι ψύκτρες είναι αγώγιμες επιφάνειες οι οποίες είναι ακίνητες ως προς τον στάτη και ωθούνται προς τον δρομέα απο ελατήρια, ωστε να εξασφαλίζεται η διαρκής επαφή τους επαφή. Στο σημείο επαφής των ψυκτρών στον δρομέα υπάρχουν αγώγιμες επιφάνειες ηλεκτρικά συνδεδεμένες με τους πόλους του τυλίγματος οπλισμού. Έτσι, κλείνει το κύκλωμα του τυλίγματος οπλισμού κατα την περιστροφή του δρομέα.

Το ρεύμα που διαρρέει τον δρομέα, λόγω του μαγνητικού πεδίου του στάτη προκαλεί μία κάθετη δύναμη στον δρομέα σύμφωνα με τον νόμο του Lorentz. το φαινόμενο αυτό είναι η αρχή λειτουργίας του κινητήρα ΣΡ. Η δύναμη αυτή έχει τέτοια φορά ωστε να τείνει να ευθυγραμμήσει το πεδίο του δρομέα με αυτό του στάτη. Οι αγώγιμες επιφάνειες κάτω απο τις ψύκτρες, όμως, είναι έτσι τοποθετημένες ωστε μόλις τα δύο μαγνητικά πεδία ευθυγραμιστούν, το ρεύμα να αλλάζει φορά διέλευσης στο τύλιγμα του δρομέα.Ετσι, η δύναμη συνεχίζει να έχει την ίδια φορά και ο κινητήρας περιστρέφεται συνεχώς.



Σχήμα 1.4

Οι κινητήρες συνεχούς ρεύματος με τύλιγμα πεδίου κατατάσσονται στις εξής κατηγορίες, ανάλογα με τον τρόπο τροφοδοσίας των δύο τυλιγμάτων:

- κινητήρας ανεξάρτητης διέγερσης. Στην περίπτωση αυτή το τύλιγμα διέγερσης τροφοδοτείται απο ανεξάρτητη πηγή τάσης απο αυτήν που τροφοδοτεί το τύλιγμα του δρομέα.Η συνδεσμολογία αυτή δίνει τις περισσότερες δυνατότητες για την οδήγηση και τον έλεγχο του κινητήρα.
- κινητήρας παράλληλης διέγερσης.Στην περίπτωση αυτή, τα τυλίγματα στάτη και δρομέα είναι συνδεδεμένα παράλληλα και, επομένως , μπορούν να τροφοδοτηθούν απο την ίδια τάση.Η συνδεσμολογία αυτή έχει το όφελος οτι παρουσιάζει αρκετά με γάλη σταθερότητα της ταχύτητας περιστροφής για μεγλάλο έυρος φορτίων. Ωστόσο δεν ενδείκνυται για φορτία που απαιτούν μεγάλη ροπή.
- κινητήρας διέγερσης σειράς.Στην περίπτωση αυτή, το τύλιγμα του πεδίου είναι συνδεδεμένο σε σειρά με το τύλγμα του στάτη, μοιραζόμενο το ίδιο ρεύμα.Η συνδεσμολογία αυτή παρουσιάζει μεγάλες διακυμάνσεις της ταχύτητας και ενδείκνυται μόνο για μεγάλα φορτία. Ωστόσο, δεν θα πρέπει να χρησιμοποιείται καθόλου σε λειτουργία εν κενω (δηλ. χωρις φορτίου) γιατι η ταχύτητα θα αυξάνεται μέχρι μη επιτρεπτά επίπεδα.θεωρητικά,χωρίς φορτίο η ταχύτητα απειρίζεται.
- Τέλος, ο κινητήρας σύνθετης διέγερσης χρησιμοποιεί δύο τυλίγματα πεδίου.Το ένα είναι παράλληλο με το τύλιγμα του δρομέα και το άλλο σε σειρά.η συνδεσμολογία αυτή ισορροπεί ανάμεσα στα χαρακτηριστικά των δύο προηγούμενων

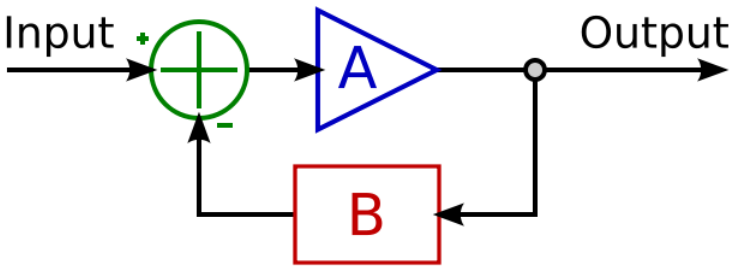
περιπτώσεων, συγκεντρώνοντας αρκετά οφέλη τόσο στην σταθερότητα της ταχύτητας, όσο και στην δυνατότητα άσκησης ροπής.

Στην εργασία αυτή έχουμε επικεντρωθεί στον κινητήρα συνεχούς ρεύματος με ψύκτρες και πεδίο απο μόνιμο μαγνητη.

1.3 Σερβομηχανισμός - Αυτόματος έλεγχος – Ελεγκτές

1.3.1 Γενικές έννοιες

Γενικά με τον όρο Σερβομηχανισμός αναφερόμαστε σε οποιοδήποτε σύστημα ελέγχεται με την βοήθεια ανάδρασης. Ανάδραση είναι η πληροφορία για κάποιο φυσικό μέγεθος του συστήματος το οποίο μας ενδιαφέρει, και μετράται με την βοήθεια κάποιου αισθητήρα. Η μέτρηση αυτή συνήθως αφαιρείται απο την επιθυμητή τιμή για αυτό το μέγεθος και η διαφορά τους αποτελεί το σφάλμα του συστήματος. Ο έλεγχος έχει σαν σκοπό τον μηδενισμό του σφάλματος. Η ιδέα του σερβομηχανμισμού και της ανάδρασης γενικότερα αποτελούν τα θεμελιώδη στοιχεία στην θεωρία του αυτόματου ελέγχου. Ενα σύστημα ελεγχου στο οποίο υπάρχει ανάδραση ονομάζεται κλειστού βρόχου, λόγω του βρόχου που εμφανίζεται στο αντίστοιχο διάγραμμα βαθμίδων του συστήματος. Ενα ενδεικτικό σχήμα εμφανίζεται παρακάτω, όπου ο κλάδος της ανάδρασης περιέχει το σύστημα B:



Σχήμα 1.5

Σε έναν σερβομηχανισμό, επομένως γνωρίζουμε ανα πάσα στιγμή την πραγματική κατάσταση του συστήματος, ως προς το μέγεθος που θέλουμε να ελέγξουμε, και άρα το σφάλμα του συστήματος που είναι,όπως είπαμε, η διαφορά της επιθυμητής τιμής απο την πραγματική. Με βάση την τιμή του σφάλματος μπορούμε να επιδράσουμε στο σύστημα με διάφορους τρόπους. Κατασκευάζουμε ειδικά συστήματα, τους ελεγκτές, οι οποίοι παρεμβάλλονται συνήθως ανάμεσα στο σφάλμα και το φυσικό σύστημα, ή οπουδήποτε αλλού μέσα στον βρόχο. Οι ελεγκτες αυτοί πραγματοποιούν μαθηματικές πράξεις στην τιμή του σφάλματος και το αποτέλεσμά τους αποτελεί το σήμα ελέγχου που δέχεται το φυσικό σύστημα.

Τα ζητούμενα απο την προσθήκη των ελεγκτών είναι τα εξής:

- Το συνολικό σύστημα που προκύπτει απο το αρχικό μετα την προσθήκη του ελεγκτή οφείλει κατ’αρχήν να είναι ευσταθές. Αυτό σημαίνει οτι η έξοδος του

συστήματος δεν θα πρέπει να απειρίζεται για κάποια πεπερασμένη είσοδο, σύμφωνα με τον ορισμό της BIBO ευστάθειας.

- Το σύστημα πρέπει επίσης να εμφανίζει κάποια σθεναρότητα. Αυτο σημαινει οτι θα παραμείνει ευσταθές και μετα την επίδραση απο ενδεχόμενες μή προβλέψιμες πηγές παρεμβολών, μέσα σε κάποια όρια.
- Το συνολικό προκύπτον σύστημα θα πρέπει να μπορεί να ακολουθήσει την συνάρτηση εισόδου, δηλαδή η έξοδος να μπορεί να πάρει τις ζητούμενες τιμές.
- Το προκύπτον σύστημα θα πρέπει να έχει καλή χρονική απόκριση. Δηλαδή, η έξοδός του θα πρέπει να φθάνει στις επιθυμητές τιμές σχετικά σύντομα απο όταν αυτό ζητηθεί, ανάλογα βέβαια και με τις απαιτήσεις της κάθε εφαρμογής.
- Ανάλογα με τις απαιτήσεις της κάθε εφαρμογής, το σύστημα ενδεχομένως να καλείται να πληροι και κάποια επι πλέον χαρακτηριστικά, όπως παραδείγματος χάριν να μην παρουσιάζει ταλαντώσεις ή να μήν υπερβεί την επιθυμητή τιμή κατα περισσότερο απο ένα όριο

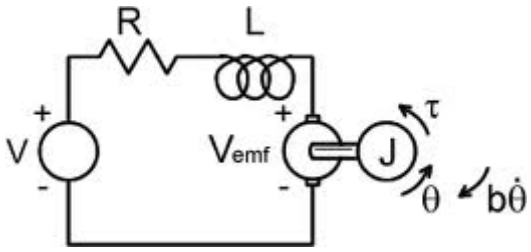
Με βάση τα ζητούμενα που θέλουμε για κάποιο σύστημα και κάποια εφαρμογή και ανάλογα με το αρχικό μας σύστημα, σχεδιάζουμε τον κατάλληλο ελεγκτή. Αυτη η διαδικασία είναι σε κάποιες περιπτώσεις αρκετά πολύπλοκη. Ωστόσο, υπάρχει πλήθος σχεδιαστικών εργαλείων που μας βοηθούν σε αυτήν την δουλειά.

1.3.2 Σερβοκινητήρας ΣΡ

Σύμφωνα με τα παραπάνω, για να μετατρέψουμε έναν κινητήρα σε σερβομηχανισμό, δηλαδή σερβοκινητήρα, αρκεί να προσθέσουμε ένα ή περισσότερα στοιχεία ανάδρασης (αισθητήρες). Στους κινητήρες τα μεγέθη που μας ενδιαφέρουν είναι η γωνιακή θέση ή/και η ταχύτητα, και ίσως η επιτάχυνση. Σε επόμενη ενότητα αναλύονται οι αιθητήρες που μετρούν τα μεγέθη αυτά.

Επίσης απαραίτητος είναι ένας ελεγκτής και κάποιο κύκλωμα οδήγησης, ωστε το σήμα ελέγχου του ελεγκτή να μετατρέπεται σε σήμα ισχύος και να οδηγείται απο αυτό ο κινητήρας. Για την αναζήτηση ενός κατάλληλου ελεγκτή πρέπει να έχουμε πρώτα μοντελοποιήσει σωστά το σύστημα του κινητήρα. Θα ασχοληθούμε με την περίπτωση του κινητήρα ΣΡ με ψύκτρες και στάτη με μόνιμους μαγνήτες, καθώς αυτό είναι το είδος που έχει επιλεχθεί για την υλοποίηση της εργασίας.

Στο παρακάτω σχήμα φαίνεται το κυκλωματικό ισοδύναμο του δρομέα ενός κινητήρα ΣΡ με ψύκτρες :



Σχήμα 1.6

Το τύλιγμα οπλισμού ισοδυναμεί με μία αντίσταση σε σειρά με ένα πηνίο. Η κίνηση του δρομέα μέσα στο μαγνητικό πεδίο που δημιουργείται απο τον στάτη προκαλεί την

εμφάνιση μίας αντι-ηλεκτρεγερτικής δύναμης (αντι-ΗΕΔ). Αυτή εμφανίζεται ώς μία πηγή τάσης με αντίθετη πολικότητα απο ότι η τροφοδοσία. Το ρεύμα που διαρρέει τον δρομέα έχει σαν αποτέλεσμα να ασκείται ροπή στον άξονά του.

$$T_m = K_t * i \qquad (3)$$

, όπου T_m η μηχανική ροπή , K_t η ηλεκτρομηχανική σταθερά οπλισμού και i το ρεύμα που διαρρέει το κύκλωμα οπλισμού

$$V_{emf} = K_b * \omega \qquad (4)$$

, όπου V_{emf} η αντιηλεκτρεγερτική δύναμη, K_b η ηλεκτρομηχανική σταθερά του κινητήρα και ω η γανιακή ταχύτητα.

Να σημειώσω οτι οι ηλεκτρομηχανικές σταθερές εξαρτώνται και απο την ένταση του μαγνητικού πεδίου που επιβάλλει ο στάτης. Ωστόσο στην εργασία αυτήν θα χρησιμοποιήσουμε κινητήρα με μόνιμους μαγνήτες στον στάτη. Γι'αυτόν τον λόγο τα μεγέθη K_t. K_b έχουν ληφθεί ως σταθερές.

Ο δρομέας, εφόσον έχει κάποια μάζα και κάποια σημεια τριβής με τον στάτη, παρουσιάζει μία συγκεκριμένη ροπή αδρανείας **J** και απόσβεση **b**.Εφαρμόζοντας τον νόμο του Νεύτωνα για την στροφική κίνηση σε συνδυασμό με την (3), προκύπτει:

$$J\frac{d\omega}{dt} + b * \omega = K_t * i \qquad (5)$$

και εφαρμόζοντας τον νόμο τάσεων του kirchoff στο κύκλωμα του δρομέα σε συνδυασμό με την (4) έχουμε:

$$L * \frac{di}{dt} + R*i = V - K_b * \omega \quad (6)$$

Απο τις εξισώσεις αυτές μπορεί να προκύψει η περιγραφή του συστήματος στον χώρο καταστασης

$$\dot{X} = AX + B * u$$

$$Y = CX + Du$$

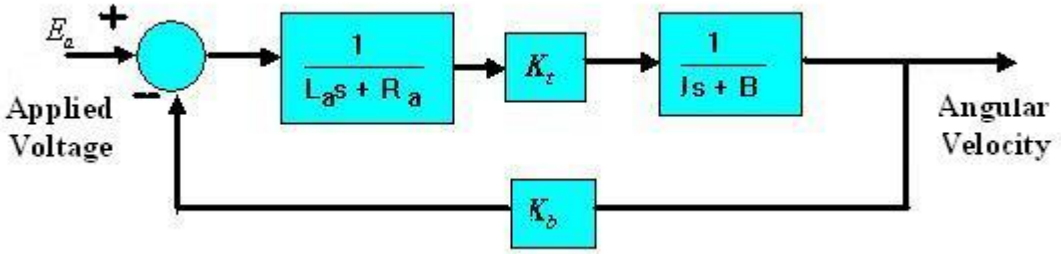
,όπου X είναι το διάνυσμα κατάστασης και $\dot{X}$ η παράγωγός του, ενώ A,B,C,D πίνακες. Y είναι η έξοδος του συστήματος

Στην περίπτωση του κινητήρα μας, ας θεωρήσουμε ως στοιχεία του διανύσματος κατάστασης την γωνιακή ταχύτητα και το ρέυμα του οπλισμού, και σαν έξοδο του συστήματός της την γωνιακή ταχύτητα. Είσοδος είναι η τάση V. Τότε η περιγραφή που προκύπτει είναι η εξής:

$$\frac{d}{dt}\begin{bmatrix}\omega \\ i\end{bmatrix} = \begin{bmatrix}-\frac{b}{J} & \frac{K_t}{J} \\ -\frac{K_b}{L} & -\frac{R}{L}\end{bmatrix} * \begin{bmatrix}\omega \\ i\end{bmatrix} + \begin{bmatrix}0 \\ \frac{1}{L}\end{bmatrix} * V$$

$$\omega = \begin{bmatrix}1 & 0\end{bmatrix} * \begin{bmatrix}\omega \\ i\end{bmatrix}$$

Μετασχηματίζοντας κατα Laplace τις διαφορικές εξισώσεις (5) και (6) μπορούμε να σχηματίσουμε και το διάγραμμα βαθμίδων του κινητήρα ΣΡ:



Σχήμα 1.7

και να γράψουμε την συνάρτηση μεταφοράς:

$$\frac{\omega}{V} = \frac{Kt}{(L * s + R) * (J * s + b) + Kt * Kb}$$

Αν θέλουμε να θεωρήσουμε την γωνιακή θέση (θ) του κινητήρα ως έξοδο και όχι την γωνιακή ταχύτητα, αρκεί να προσθέσουμε εναν ολοκληρωτή ($\frac{1}{s}$) στο τέλος του διαγράμματος. Ετσι, η συνάρτηση μεταφοράς γίνεται:

$$\frac{\theta}{V} = \frac{Kt}{\big((L * s + R) * (J * s + b) + Kt * Kb\big) * s}$$

Ας μελετήσουμε αυτό το σύστημα $\frac{\theta}{V}$:

Βλέπουμε οτι έχει 3 πόλους, απο τους οποίους ο ένας είναι επάνω στην αρχή των αξόνων, λόγω του ολοκληρωτή.Οι άλλοι δύο πόλοι προκύπτουν ώς ρίζες του χαρακτηριστικού πολυωνύμου:

$$(L*J)\, s^2 + (L*b + R*J)\, s + (R*b + K_t*K_b) = 0$$

Οι πόλοι αυτοί είναι και οι μοναδικοί πόλοι του συστήματος αν θεωρήσουμε ως έξοδο την γωνιακή ταχύτητα. Τα χαρακτηριστικά μεγέθη του κινητήρα (R,L,J,b,K_b,K_t) λαμβάνουν όλα μη μηδενικές θετικές τιμές. Με θεώρηση του τριωνύμου βλέπουμε οτι όλοι οι όροι είναι άθροισματα θετικών τιμών, άρα είναι όλοι θετικοί και μή μηδενικοί. Επομένως, απο το κριτήριο ευστάθειας των Routh-Hurwitz προκύπτει οτι το σύστημα είναι ευσταθές.

Συνεπώς, οι δύο πόλοι βρίσκονται στο αριστερό μιγαδικό ημιεπίπεδο, δηλαδή έχουν αρνητικό πραγματικό μέρος και μπορούμε να χρησιμοποιήσουμε το εξής μοντέλο για το χαρακτηριστικό πολυώνυμο της συνάρτησης μεταφοράς:

$$s^2 + 2\zeta\omega_n s + \omega_n^2$$

,όπου ζ η απόσβεση και ω_n η φυσική ιδιοσυχνότητα του συστήματος

Τα στοιχεία ζ και ω_n μας δίνουν χρήσιμες πληροφορίες για την χρονική απόκρση του συστήματος. Ακόμα και σε ένα σύστημα με περισσότερους απο δύο πόλους, μπορούμε να χρησιμοποιήσουμε αυτό το μοντέλο για τους δύο επικραστέστερους (δηλαδή πιο κοντά στον άξονα των φανταστικών) πόλους και να λάβουμε μία καλή προσέγγιση της συμπεριφοράς του συστήματος. Ενας απο τους πιο βασικούς στόχους της χρήσης των ελεγκτών είναι η δημιουργία ενός νέου συστήματος απο τον συνδυασμό του αρχικού με τον ελεγκτή, το οποίο θα έχει τους πόλους του σε επιθυμητά σημεία. Δηλαδή, η απόκρισή του θα εμφανίζει τα επιθυμητά χαρακτηριστικά.

Οι ελεγκτές που μπορούμε να χρησιμοποιήσουμε είναι πολλοί, και οι σχεδιαστικές μέθοδοι επίσης. Σε αυτήν την εργασία αποφασίσαμε να χρησιμοποιήσουμε έναν ιδιαίτερα δημοφιλή τύπο ελεγκτή, τον ελεγκτή τριών όρων (αναλογικού-παραγώγου-ολοκληρώματος, PID). Ο ελεγκτής αυτός παρουσιάζεται αναλυτικα στην επόμενη ενότητα.

1.4 Ελεγκτής τριών όρων – PID

1.4.1 Χαρακτηριστικά - συμπεριφορά

Ο ελεγκτής PID είναι ίσως ο πιο ευρέως χρησιμοποιούμενος ελεγκτής συστημάτων κλειστού βρόχου στην βιομηχανία. Η χρήση του είχε ξεκινήσει ήδη το 1890 περίπου, ενώ η πρώτη θεωρητική ανάλυση πραγματοποιήθηκε απο τον Ρωσσο-αμερικανό Nicolas Minorsky. Ο μηχανικός αυτός σχεδίαζε συστήματα αυτόματης πλοήγησης πλοίων, και η ιδέα του στηρίχθηκε στην παρακολούθηση των πηδαλιούχων στα πλοία κατα τον χειρισμό του τιμονιού. Ο Minorsky παρατήρησε οτι οι χειριστές του πηδαλίου ήλεγχαν το πλοίο βασιζόμενοι όχι μόνο στο παρόν σφάλμα της τροχιάς, αλλα και σε προηγούμενα σφάλματα, καθώς και στον ρυθμό μεταβολής τους. Αυτή η διαδικασία μοντελοποιήθηκε μαθηματικά και έδωσε τον ελεγκτή τριών όρων.

Όπως οι περισσότεροι ελεγκτές, ο ελεγκτής τριών όρων πραγματοποιεί μαθηματικές πράξεις στο σήμα του σφάλματος (e(t)) και τροφοδοτεί το αποτέλεσμα (u(t)) στο σύστημα με σκοπό το σφάλμα να μηδενιστεί. Δηλαδή, η τιμή της υπο έλεγχο παραμέτρου να συμπέσει με την επιθυμητή.

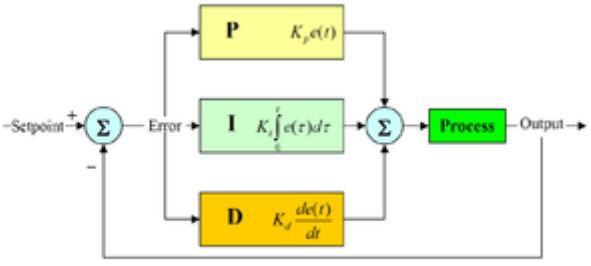
Ο ελεγκτής PID αποτελείται απο τρείς παράλληλους κλάδους, το αποτέλεσμα των οποίων προστίθεται. Η συνεισφορά του κάθε κλάδου ονομάζεται P , I και D όρος. Ας μελετήσουμε κάθε κλάδο ξεχωριστά:

- Ο P κλάδος πολλαπλασιάζει την τρέχουσα τιμή του του σφάλματος με κάποιο κέρδος, το K_p.
- Ο I κλάδος ολοκληρώνει το σφάλμα στον χρόνο, και πολλαπλασιάζει το αποτέλεσμα της ολοκλήρωσης με τον κέρδος K_i.
- Ο D κλάδος παραγωγίζει το σφάλμα ως προς τον χρόνο και πολλαπλασιάζει την παράγωγό του με την σταθερά K_d.

Οι τρείς αυτοί όροι προστιθέμενοι αποτελούν την έξοδο του ελεγκτή:

$$u(t) = K_p * e(t) + K_i \int e(t)dt + K_d * \frac{de(t)}{dt}$$

Το σήμα u(t) αποτελεί το σήμα εισόδου στο υπο έλεγχο σύστημα. Ας δούμε και ένα σχετικό διάγραμμα βαθμίδων:



Σχήμα 1.8

Εχοντας δεί την εξίσωση του ελεγκτή στο πεδίο του χρόνου, ας δούμε και στο πεδίο συχνοτητας, που προκύπτει μετά απο μετασχηματισμό κατα Laplace:

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d * s$$

,όπου U(s) και E(s) οι μετασχηματισμοί κατα Laplace των σημάτων u(t) και ε(t) αντίστοιχα.

Ο ελεγκτής συμπεριφέρεται ως ελεγκτής καθυστέρησης-προήγησης φάσης και βελτιώνει τόσο το μόνομο σφάλμα όσο και την χρονική απόκριση του συστήματος. Συγκεκριμένα, το κέρδος του κάθε όρου έχει την εξής επίδραση:

- Το αναλογικό κέρδος K_p βελτιώνει τον χρόνο ανόδου της βηματικής απόκρισης του συστήματος και μειώνει το μόνιμο σφάλμα,όταν υπάρχει, χωρίς όμων να το εξαλείφει ποτέ τελείως.
- Το ολοκληρωτικό κέρδος K_i χρησιμοποιείται μόνο όταν υπάρχει μόνιμο σφάλμα για την εξάλειψή του, ενω παράλληλα χειροτερεύει την μεταβατική απόκριση του συστήματος και εισάγει ταλαντώσεις αν αυξηθεί περισσότερο απο όσο χρειάζεται.
- Το διαφορικό κέρδος ρυθμίζει μόνο την μεταβατική αποκριση του συστήματος. μειώνει τον χρόνο ανόδου, τον χρόνο αποκατάστασης και την υπαρύψωση. Χρησιμοποιείται γενικά για να κάνει το σύστημα πιο γρήγορο και πιο σταθερό.

Διακριτός χρόνος:

Αφού η πλειοψηφία των συστημάτων ελέγχου στις μέρες μας αποτελούνται απο ψηφιακά συστήματα, είναι χρήσιμο να περιγράψουμε την μορφή ενός ελεγκτή PID διακριτού χρόνου:

Αρχικά, διακριτός χρόνος σημαίνει οτι η τιμή ενός σήματος που μας ενδιαφέρει δεν μας είναι διαθέσιμη ανα πάσα στιγμή, αλλα σε καθορισμένα χρονικά διαστήματα. Εχουμε, δηλαδή, μία δειγματοληψία του σήματος με περίοδο δειγματοληψίας T_s. Ετσι, ενα σήμα e(t) λαμβάνεται ως μία ακολουθία τιμών e(n). Χρειάζεται, επομένως να κάνουμε τις ακόλουθες προσεγγίσεις για τις πράξεις της ολοκλήρωσης και διαφόρισης:

διακριτή ολοκλήρωση: $\int_0^t e(t)dt \approx T_s * \sum_0^n e(n)$

διακριτή διαφόριση: $\frac{de(t)}{dt} \approx \frac{e(n) - e(n-1)}{T_s}$

Τώρα, αν γράψουμε την συνάρτηση του ελεγκτή PID συνεχούς χρόνου στο πεδίο του χρόνου:

$$u(t) = K_p * e(t) + K_i \int e(t)dt + K_d * \frac{de(t)}{dt}$$

και κάνουμε χρήση των προσεγγίσεων που αναφέραμε, ετασχηματίζεται στο εξής σύστημα διακριτού χρόνου:

$$u(n) = K_p * e(n) + K_i * T_s * \sum e(n) + \frac{K_d}{T_s} * (e(n) - e(n-1)) \ . \quad (1)$$

Αυτή η μετατροπή μας δείχνει οτι μπορούμε να σχεδιάσουμε τον ελεγκτή μας στο σύστημα συνεχούς χρόνου και στην συνέχεια να χρησιμοποιήσουμε τις αλλαγές:
K_i <-> $K_i * T_s$ και K_d <-> K_d / T_s , για υλοποίηση σε σύστημα διακριτού.
Η παραπάνω μορφή του ελεγκτή ονομάζεται ελεγκτής *θέσης* επειδή υπολογίζει την κάθε τιμή εξόδου ξεχωριστά.

Η μετατροπή του συνεχούς ελεγκτή στον αντίστοιχο διακριτό μπορεί να γίνει και στο πεδίο της συχνότητας, όπου απο το πεδίο της μεταβλητής s του μετασχηματισμού του Laplace θα μεταβούμε στην αντίστοιχη σχέση του μετασχηματισμού-z. Η μετατροπή γίνεται με τους εξής τρόπους:

- Αντικατάσταση $s \leftarrow \frac{(1 - z^{-1})}{T_s}$

  Η αντικατάσταση αυτή βασίζεται στην προσέγγιση $\Delta x = x(t) - x(t - T_s)$.Ο όρος z^i δηλώνει μετατόπιση της συνάρτησης κατα i δείγματα και το πρόσημο του i δηλώνει την κατεύθυνση της μετατόπισης (‘+’ : μπροστά στον χρόνο,’-‘: πίσω), δηλαδή $z^{-n} x(t) = x(t - nT_s)$. Η μέθοδος αυτή προκύπτει απο την μέθοδο ολοκλήρωσης του Euler.

- Αντικατάσταση $s \leftarrow \frac{2(1 - z^{-1})}{T_s(1 + z^{-1})}$

  Η μέθοδος αυτή είναι γνωστή ώς *Διγραμμικός μετασχηματισμός* και προκύπτει απο τον κανόνα τραπεζίου για την ολοκλήρωση

Κάνοντας χρήση της πρώτης μεθόδου στην εξίσωση του ελεγκτή έχουμε:

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d * s \xrightarrow{s \leftarrow \frac{(1 - z^{-1})}{T_s}} u(t) = e(t)[K_p + \frac{K_i T_s}{(1 - z^{-1})} + K_d \frac{(1 - z^{-1})}{T_s}]$$

και μετά απο απλοποιήσεις:

$$u(t) = u(t-1) + K_p[e(t) - e(t-1)] + K_i T_s e(t) + \frac{K_d}{T_s}[e(t) - 2e(t-1) + e(t-2)] \quad (2)$$

Η μορφή αυτή του ελεγκτή ονομάζεται ελεγκτής *ταχύτητας* επειδή υπολογίζει την κάθε τιμή της εξόδου βασισμένος στην προηγούμενη τιμή της. Δηλαδή, υπολογίζεται η *μεταβολή* στην τιμή εξόδου, και όχι αυτούσια η τιμή.

Οι δύο αυτές μορφές του ελεγκτή είναι ισοδύναμες. Αυτό μπορεί να αποδειχτεί αν μεταθέσουμε την εξίσωση (1) κατα ένα δείγμα πίσω στον χρόνο:

$$u(t-1) = K_p * e(t-1) + K_i * T_s * \sum e(t-1) + \frac{K_d}{T_s} * (e(t-1) - e(t-2))$$

και την αφαιρέσουμε απο την αρχική εξίσωση (1), όπου προκύπτει η εξίσωση (2):

$$u(t) = u(t-1) + K_p[e(t) - e(t-1)] + K_i T_s e(t) + \frac{K_d}{T_s}[e(t) - 2e(t-1) + e(t-2)]$$

1.4.2 Ρύθμιση PID

Η συνάρτηση μεταφοράς του ελεγκτή όπως είπαμε είναι η ακόλουθη:

$$K_p + \frac{K_i}{s} + K_d * s = \frac{K_d * s^2 + K_p * s + K_i}{s}$$

Παρατηρούμε οτι ο ελεγκτής έχει έναν πόλο στην αρχή των αξόνων και δύο μηδενικά στις θέσεις:

$$s_1 = \frac{-K_p + \sqrt{K_p{}^2 - 4K_d K_i}}{2K_d} \text{ και } s_2 = \frac{-K_p - \sqrt{K_p{}^2 - 4K_d K_i}}{2K_d}.$$

Ρυθμίζοντας τις τιμές των κερδών K_p,K_i,K_d καθορίζουμε τις θέσεις των μηδενικών του ελεγκτή ωστε να τοποθετήσουμε τους πόλους του συνολικού κλειστού συστήματος στα επιθυμητά σημεία. Η αναλυτική διαδικασία αυτή απαιτεί πλήρη γνώση του υπο ελέγχου συστήματος και μπορεί να γίνει με την βοήθεια διάφορων εργαλείων, όπως ο Γεωμετρικός Τόπος Ριζών, τα διαγράμματα Bode και τα διαγράμματα Niquist.

Προσεγγιστικές μέθοδοι:

Σε πολλές περιπτώσεις, ίσως στην πλειοψηφία των εφαρμογών, η συνάρτηση μεταφοράς του υπο ελέγχου συστήματος δεν μας είναι γνωστή. Συνεπώς, η ρύθμιση του ελεγκτή δεν μπορεί παρα να γίνει με προσεγγιστικές μεθόδους. Η πιο διαδεδομένη μέθοδος ρύθμισης των ελεγκτών PID είναι η μέθοδος Ziegler-Nichols.Η μέθοδος αυτή στηρίζεται στην μελέτη της απόκρισης του συστήματος. Η διαδικασία ρύθμισης μπορεί να γίνει με δύο τρόπους:

- Πείραμα ανοιχτού βρόχου

  Η διαδικασία αυτή περιλαμβάνει την μέτρηση της βηματικής απόκρισης του συστήματος ανοιχτού βρόχου. Χρησιμεύει για συστήματα με ευσταθείς και μονότονες αποκρίσεις που ενδεχομένως παρουσιάζουν κάποια χρονική καθυστέρηση. Τα συστήματα αυτά μπορούν να προσεγγιστθούν μέσω της συνάρτησης μεταφοράς:

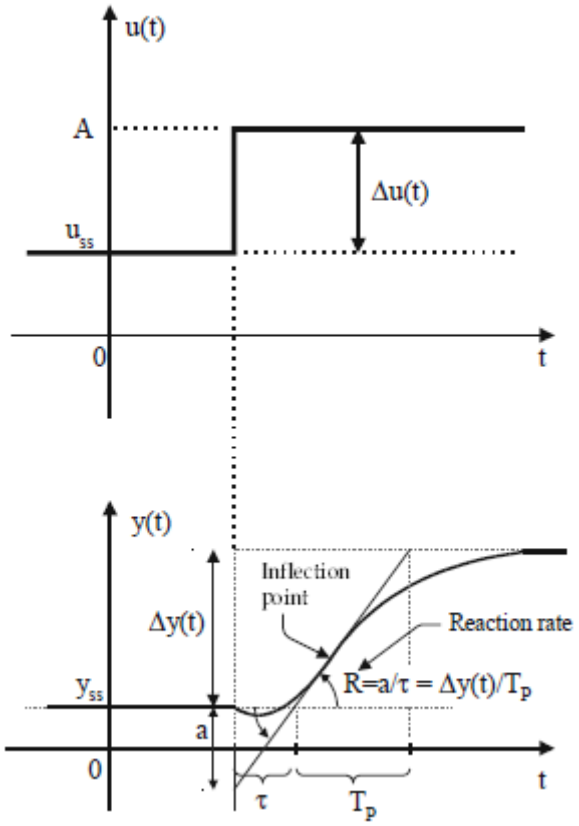
$$G_p(s) = \frac{K * e^{-\tau s}}{1 + sT_p}$$

  Απο την θεώρηση της βηματική ςαπόκρισης προκύπτουν οι τιμές των :

  K: DC κέρδος , K = Δy(t)/Δu(t)

  τ: χρονική καθυστέρηση

  T_ρ: χρονική σταθερά



Σχήμα 1.9

Απο τις τιμές αυτές προκύπτει η σταθερά $a = \frac{\tau}{T_p} * K$

Στην συνέχεια τα κέρδη του ελεγκτή ρυθμίζονται σύμφωνα με τον πίνακα:

K_p	K_i	K_d
1.2/α	0,6/(α*τ)	(0,6*τ)/α

• Πείραμα κλειστού βρόχου

Η διαδικασία αυτή δεν απαιτεί την υπόθεση κάποιου μοντέλου για το σύστημα, ούτε την ευστάθειά του. Αποτελεί μέθοδο δοκιμής-λάθους. Στην αρχή το σύστημα δοκιμάζεται στην διάταξη κλειστού βρόχου με επενέργηση μλόνο του ελεγκτή P. Στην συνέχεια το αναλογικό κέρδος αυξάνεται μέχρις ότου να φτάσει στην τιμή **υπέρτατου κέρδους K_u**. Στην τιμή αυτή η έξοδος του συστήματος παρουσιάζει αμείωτες ταλαντώσεις με περίοδο T_u. Η περίοδος αυτή χρειάζεται να μετρηθεί. επίσης, μπορέι να μετρηθεί και το DC κέρδος K=Δy(t)/Δu(t). Στην συνέχεια, οι τιμές των τριών κερδών ρυθμίζονται σύμφωνα με τον παρακάτω πίνακα:

K_p	K_i	K_d

$0.6 * K_u$	$2*K_p/T_u$	$K_p*T_u/8$

Συνίσταται ο πίνακας αυτός να χρησιμοποιείται μονο εφ'όσον ισχύει: $2<K*K_u<20$.

Τέλος, να σημειωθεί οτι η προσεγγιστικές μέθοδοι Ziegler-Nichols δεν βρίσκουν την βέλτιστη ρύθμιση για τον ελεγκτή. Οι πίνακες έχουν προκύψει απο πειράματα σε διάφορα συστήματα και εχουν ως σκοπό κυρίως την σθεναρή λειτουργία με μικρή ευαισθησία σε αστάθμητες παραμέτρους που επηρεάζουν την διαδικασία, και όχι την ταχύτητα απόκρισης. Ωστόσο, είναι ιδιαίτερα χρήσιμοι για την αρχική ρύθμιση του ελεγκτή και την σωστή λειτουργία. Στην συνέχεια μπορεί να αναζητηθεί η βελτίωση της απόκρισης του συστήματος μέσω δοκιμών με μικροαλλαγές στα κέρδη.

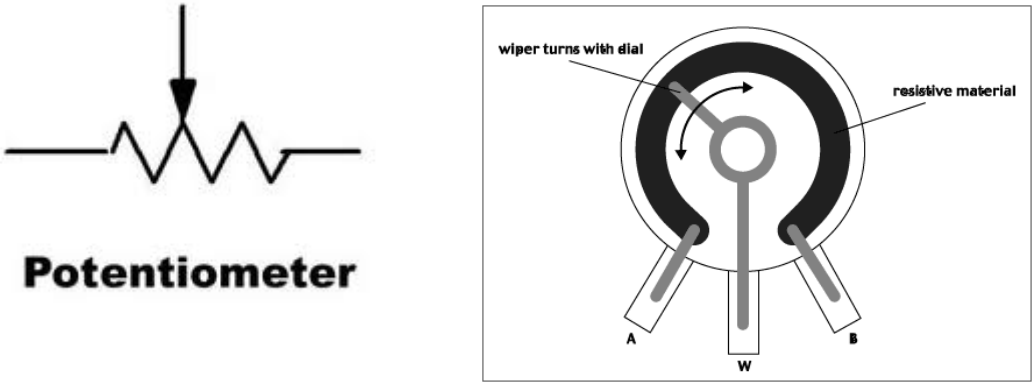
1.5 Αισθητήρες περιστροφικής κίνησης

1.5.1 Είδη αισθητήρων

Για την μέτρηση των φυσικών μεγεθών της κίνησης ενός άξονα, χρησιμοποιούνται οι παρακάτω αισθητήρες:

Αισθητήρες θέσης:

- Ποτεσιόμετρο. Πρόκειται για μία μεταβλητή αντίσταση.Η αντίσταση αυτή έχει δύο σταθερούς ακροδέκτες και έναν κινούμενο, ο οποίος ολισθαίνει κατα μήκος της αντίστασης.Η κίνηση συνήθως δίνεται απο έναν άξονα. Εαν ο άξονας αυτός προσαρμοστεί στον άξονα του κινητήρα και στους σταθερούς ακροδέκτες του εφαρμοστεί μια τάση, τοτε το ποτενσιόμετρο λειτουργεί ως διαιρέτης τάσης και η τάση του τρίτου ακροδέκτη είναι ανάλογη με την γωνιακή μετατόπιση του άξονα.Ετσι, έχουμε την μετατροπή της γωνίας του άξονα σε DC τάση, ένα πολύ εύχρηστο σήμα τόσο για αναλογικά, όσο και για ψηφιακά συστήματα.



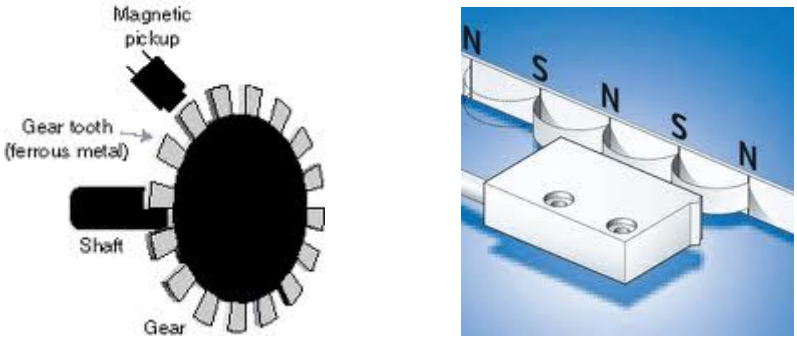
Σχήμα 1.10

- Οπτικός κωδικοποιητής (optical encoder). Πρόκειται για ένα η δύο ζεύγη διόδου LED-φωτοτρανζίστορ, σε δύο είδη διάταξης: την φωτο-διακοπτική και την φωτοανακλαστική. Ο αισθητήρας αυτός λειτουργεί σε συνδυασμό με έναν δίσκο με

εγκοπές ή με χρωματισμένες ρίγες, ο οποίος είναι προσαρμοσμένος στον άξονα που επιθυμούμε να παρακολουθούμε. Η έξοδός του παράγει παλμούς ο κάθε ένας απο τους οποίους ατιστοιχεί σε κάποια γωνιακή μετατόπιση του άξονα. Αυτός είναι ο τύπος αισθητήρα που έχει επιλεγεί για αυτήν την εργασία, και η λειτουργία του θα αναλυθεί στην συνέχεια.

Σχήμα 1.11

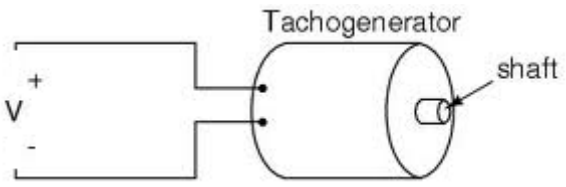
- Μαγνητικός κωδικοποιητής (magnetic encoder). Στηρίζεται στο ίδιο σκεπτικό με τον προηγούμενο, με την διαφορα οτι αντί για την χρήση φωτός, χρησιμοποιεί τα μαγνητικά χαρακτηριστικά ενός δίσκου. Αυτά είναι είτε η μαγνητική διαπερατότητα, οπότε έχουμε έναν αισθητήρα που προσαρμόζεται εύκολα σε μεταλλικά γρανάζια, ή η μεταβολή του μαγνητικού πεδίου. Σε αυτήν την περίπτωση, χρησιμοποιούμεέναν δίσκο με διαδοχικούς μόνιμους μαγνήτες και έναν αισθητήρα που εκμεταλλεύεται το φαινόμενο hall.



Σχήμα 1.12

Αισθητήρες ταχύτητας:

- Ταχογεννήτρια. Δεν είναι τίποτα άλλο παρα ένας κινητήρας ΣΡ ο οποίος τοποθετείται ομοαξονικά με τον άξονα που μας ενδιαφέρει και παίρνει κίνηση απο αυτόν. η γωνιακή ταχύτητα προκαλεί μία αντι-ηλεκτρεγερτική δύναμη στους ακροδέκτες της, ανάλογης τιμής. Πρόκειται για ένα αξιόπιστο αναλογικό σήμα τάσης.



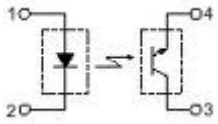
Σχήμα 1.13

Εδώ να σημειώσω οτι αφού η ταχύτητα είναι η παράγωγος της θέσης, καθένας απο τους παραπάνω αισθητήρες μπορεί να μας δώσει και τις δύο πληροφορίες, αν χρησιμοποιήσουμε παραγώγιση ή ολοκλήρωση. Ωστόσο, ειδικά η παραγώγιση αναλογικών σημάτων αποφεύγεται στην πράξη για τον λόγο οτι ενισχύει τυχον θόρυβο, ειδικά σε μεγάλες συχνότητες, και έτσι το αποτέλεσμα καθίσταται μή αξιόπιστο. Συνήθως επιλέγεται κάποιος συνδυασμός σισθητήρων.

Ας μιλήσουμε τώρα πιο αναλυτικά για τους οπτικούς κωδικοποιητές:

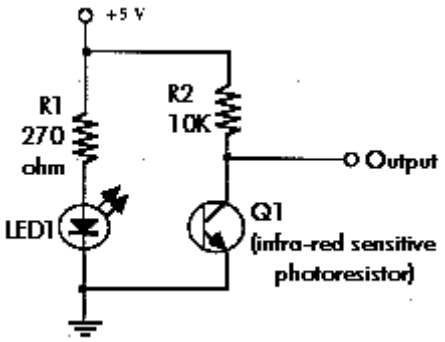
1.5.2 Οπτικοί κωδικοποιητές

Οι οπτικοί κωδικοποιητές ανήκουν στην κατηγορία των ψηφιακών αισθητήρων, αν και υπάρχουν και κάποιοι τύποι με αναλογική έξοδο. Χωρίζονται όπως είδαμε σε δύο κατηγορίες. Τους φωτο-διακοπτικούς και τους φωτο-ανακλαστικούς. Η αρχή λειτουργίας είναι κοινή. Ο εκπομπός είναι ένα LED το οποίο λειτουργεί συνήθως στο υπέρυθρο φάσμα, και τροφοδοτείται συνεχώς. Ο δέκτης είναι ένα φωτοευαίσθητο στοιχείο, συνηθως φωτοτρανζίστορ. Βέβαια, υπάρχουν και αισθητήρες που έχουν για δέκτη φωτοδίοδο ή και φωτοαντίσταση.Το φωτοτρανζίστορ είναι ένα BJT, συνήθως NPN τρανζίστορ που αντί για ακροδέκτη βάσης έχει μία φωτοευαίσθητη επιφάνεια. Σε κάποιες περιπτώσεις υπάρχει και ακροδέκτης βάσης για να ρυθμίζεται με ακρίβεια η πόλωση. Εν τέλει, ανάλογα με την ένταση του φωτός που δέχεται η βάση διαμορφώνεται το ρεύμα συλλέκτη-εκπομπού.



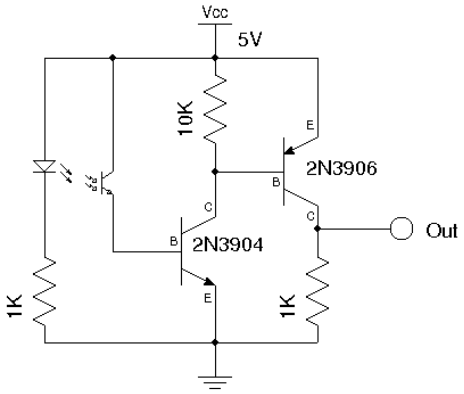
Σχήμα 1.14

Ο συλλέκτης είναι συνδεδεμένος μέσω μίας pull-up αντίστασης στην θετική τροφοδοσία και η έξοδος του κυκλώματος είναι ο ακροδέκτης του συλλέκτη. Ο εκπομπός είναι συνδεδεμένος απ'ευθείας στην γη. Επιθυμητό είναι να λειτουργούμε το τρανζίστορ ανάμεσα στις περιοχές αποκοπής και κόρου, δηλαδή ως διακόπτη. Οταν η φωτεινότητα της βάσης είναι μικρή, η αντισταση κρατάει τον ακροδέκτη του συλλέκτη σε υψηλό δυναμικό. Οταν η βάση δέχεται έντονο φώς τότε το τρανζίστορ άγει και η έξοδος έχει το δυναμικό της γής.



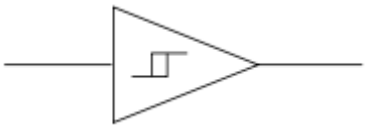
Σχήμα 1.15

Αυτή είναι η βασική αρχή λειτουργίας. Κανείς μπορεί να προσθέσει στοιχεία ωστε να βελτιστοποιήσει τα χαρακτηριστικά εξόδου με σκοπό να κάνει τον αισθητήρα λιγότερο ευαίσθητο στο φως του περιβάλλοντος, στον θόρυβο, να βελτιώσει τον χρόνο ανόδου των παλμών κ.α.

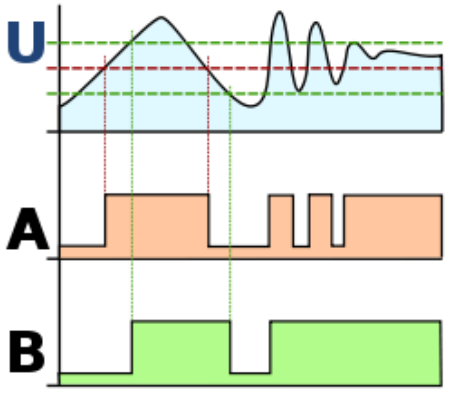


Σχήμα 1.16

Μία συνήθης τοπολογία σύνδεσης οπτικών αισθητήρων και γενικά αισθητήρων που παράγουν παλμοσειρές είναι η σύνδεση της εξόδου τους σε ένα σύστημα scmitt trigger. Το σύστημα αυτό πρόκειται ουσιαστικά για έναν συγκριτή, ο οποίος έχει θετική ανάδραση. Συγκεκριμένα, το σύστημα δουλεύει ώς εξής: Όταν η τάση του σήματος εισόδου είναι υψηλότερη απο μία τάση κατωφλίου, η έξοδος είναι σε υψηλό δυναμικό. οταν η τάση εισόδου είναι χαμηλότερη απο μία άλλη τάση κατωφλίου, η έξοδος είναι χαμηλά. Αυτή η διαφορά μεταξύ των δύο τιμών κατωφλίου ονομάζεται «υστέρηση». Η ύπαρξη της υστέρησης δίνει στο σύστημα αυτό χαρακτήρα μανδαλωτή, αφού όταν η τάση εισόδου βρίσκεται ανάμεσα στις δύο τιμές κατωφλίου, η έξοδος του συστήματος διατηρεί την προηγούμενη τιμή της. Τέτοια συστήματα χρησιμοποιούνται ευρέως για τον «τετραγωνισμό» παλμοσειρών και την απόρριψη του θορύβου και της κυμάτωσης σε τετραγωνικές κυματομορφές.



Σχήμα 1.15: Σύμβολο διάταξης schmitt trigger

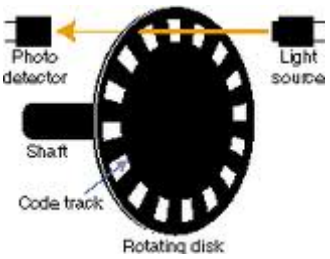


Σχήμα 1.17

Σχήμα 1.16: Εξοδος απλού συγκριτή (A) και schmitt trigger (B) για είσοδο U

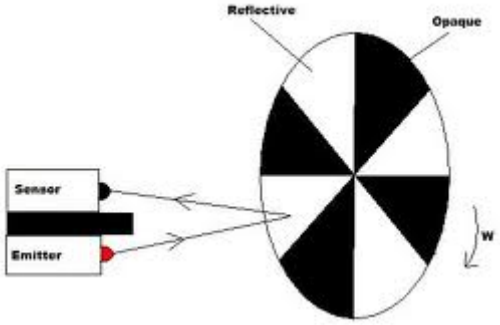
Αυτό που διαφέρει ανάμεσα στα δύο είδη αισθητήρων είναι ο τρόπος που γίνεται η οπτική σύζευξη πομπού-δέκτη.

Στην περίπτωση των φωτο-διακοπτικών αισθητήρων, ο πομπός είναι απέναντι απο τον δέκτη και παρεμβάλλεται μεταξύ τους ένας δίσκος με εγκοπές ή οπές. Το υλικό του δίσκου πρέπει να είναι αδιαφανες στο υπέρυθρο φάσμα. Καθώς ο δίσκος περιστρέφεται, η βάση δέχεται διαδοχικά ριπές φωτός και αυτές μεταφράζονται σε παλμούς στην έξοδο.Το ίδιο αποτέλεσμα πετυχαίνουμε αν τυπώσουμε ρίγες με αδιαφανές μελάνι σε δίσκο απο διαφανές υλικό.



Σχήμα 1.18 Φωτο-διακοπτική διάταξη

Στην περίπτωση των φωτο-ανακλαστικών αισθητήρων, ο πομπός και ο δέκτης είναι στραμμένοι προς την ίδια κατεύθυνση, ο ένας πάνω απο τον άλλον. Εδώ αντί για δίσκο με εγκοπές έχουμε έναν δισκο με ζωγραφισμένες σκούρες και λευκές ρίγες (ή κυκλικούς τομείς). Οι λευκές επιφάνειεςανακλούν το φώς προς τον δέκτη, ενώ οι σκούρες όχι. Ετσι η οπτική σύζευξη επιτυγχάνεται όχι απ’ευθείας, αλλα με ανάκλαση (ή διάχυση).Απαραίτητο είναι δε να εμποδίζεται η απ’ευθείας οπτική ζεύξη πομπού-δέκτη.



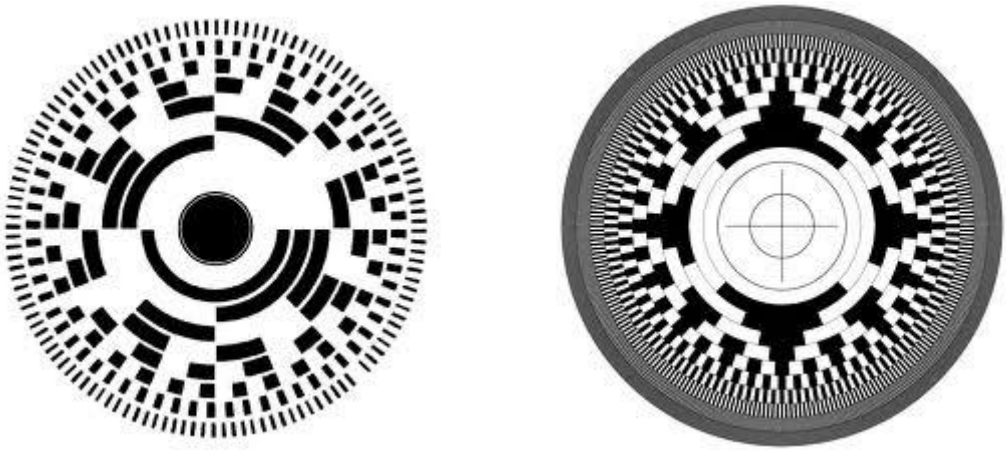
Σχήμα 1.19 Φωτο-ανακλαστική διάταξη

Και στις δύο περιπτώσεις το αποτέλεσμα είναι το ίδιο. Εχουμε μια παλμοσειρά στην έξοδο, ο κάθε παλμός της οποίας αντιστοιχεί σε συγκεκριμένη γωνιακή μετατόπιση του άξονα. Επειδή ο αισθητήρας είναι ψηφιακός, δεν είναι τόσο ευάλωτος στον θόρυβο και όπου υπάρχει είναι πιο εύκολο να αντιμετωπιστεί.Ετσι, μπορούμε να εξάγουμε πληροφορίες τόσο για την θέση, όσο και για ητν ταχύτητα του άξονα.

Μέτρηση θέσης: Το πλήθως παλμών που λαμβάνουμε απο έναν τέτοιο αισθητήρα μας δείχνει την συνολική γωνιακή μετατόπιση απο τότε που αρχίσαμε την μέτρηση. Δίνει, δηλαδή, μία μέτρηση μετατόπισης.

Απόλυτοι κωδικοποιητές: Υπάρχουν κάποια είδη αισθητήρα που μας δίνουν την δυνατότητα να παρακολουθήσουμε την απόλυτη θέση του άξονα. Αυτό γίνεται με την

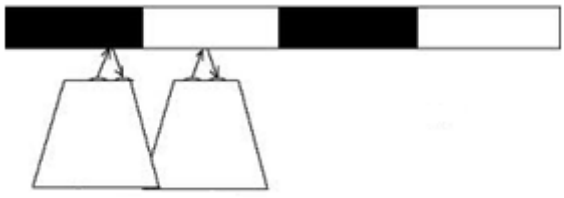
χρήση ειδικών δίσκων και πολλαπλών δεκτών, ο καθένας απο τους οποίους μπορεί να χρησιμοποιηθεί σαν ένα ψηφίο ενός δυαδικού αριθμού. Ετσι, μια πλήρης περιστροφή εξάγει μία ακολουθία 2^N αριθμών, όπου N ο αριθμός των ανεξάρτητων δεκτών(σχήμα). Αυτή η μέθοδος έχει και το επιπλέον πλεονέκτημα οτι παρέχει και την πληροφορία της φοράς περιστροφής, κάτι που δεν προκύπτει άμεσα απο την απλή παλμοσειρά.



Σχήμα 1.20: Απολυτος κωδικοποιητης δυαδικων αριθμων(αριστερά)και με κώδικα gray(δεξιά)

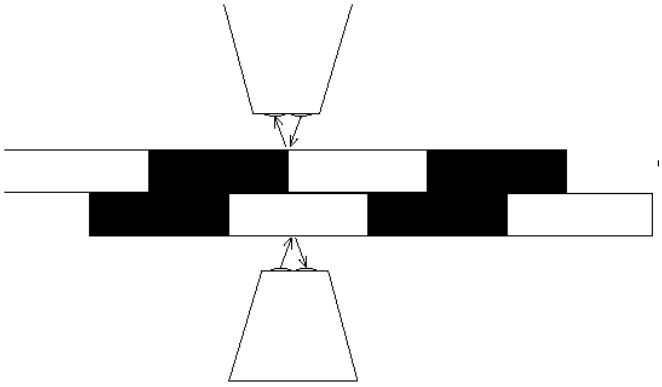
Υπολογισμός ταχύτητας: Η ταχύτητα μπορεί να υπολογιστεί με δύο τρόπους: είτε μετρώντας τον χρόνο ανάμεσα σε δύο παλμούς, οπότε έχουμε μία μέτρηση ανάλογη της περιόδου, είτε μετρώντας το πλήθος των παλμών που λαμβάνουμε μέσα σε ένα συγκεκριμένο χρονικό διάστημα.Σε αυτήν την περίπτωση έχουμε μία μέτρηση ανάλογη της συχνότητας.

Υπολογισμός φοράς: (ορθογωνικός κωδικοποιητής – quadrature encoder). Υπάρχει μία μέθοδος να εξάγουμε την πληροφορία για την φορά περιστροφής απο έναν απλό δίσκο με εγκοπές/ρίγες και δύο ξεχωριστούς αισθητήρες. Εαν τοποθετήσουμε τους αισθητήρες έτσι ώστε οι δύο παλμοσειρές εξόδου να έχουν διαφορά φάσης 90 μοίρες μπορούμε να αποφανθούμε για την φορά περιστροφής ελέγχοντας την κατάσταση της μίας παλμοσειρας σε κάθε ακμή της άλλης. Με άλλα λόγια, όταν η παλμοσειρά Α «προπορεύεται» απο την Β, η φορά περιστροφής είναι αντίθετη απο τη περίπτωση που «ακολουθεί» την Β.

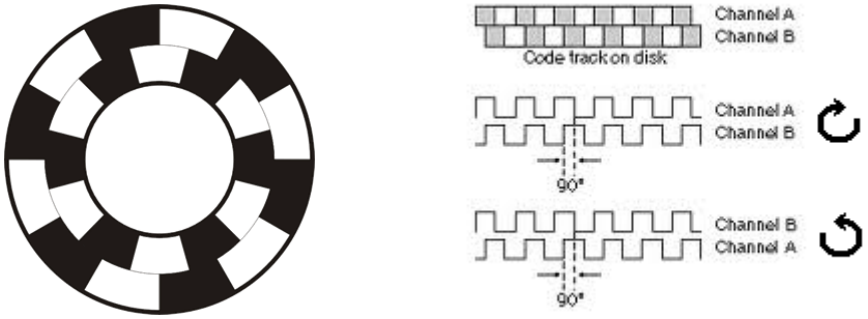


Σχήμα 1.21 Ορθογωνική διάταξη αισθητήρων (α)

Το ίδιο αποτέλεσμα προκύπτει και με κατάλληλη διαμόρφωση του δίσκου και δύο αισθητήρες στο ίδιο σημείο:

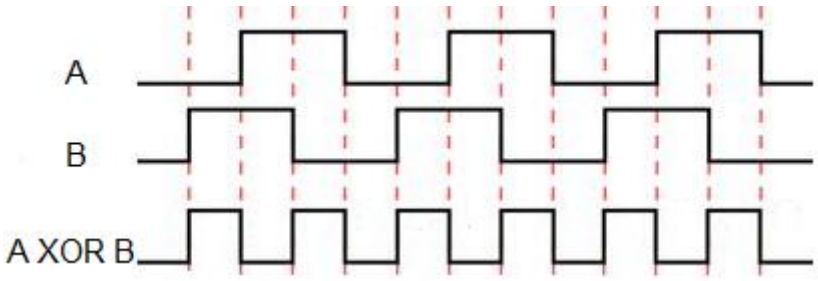


Ορθογωνική διάταξη αισθητήρων (β)



Σχήμα 1.22

Επι πλέον, αν οι παλμοσειρές έχουν ακριβώς 50% duty cycle (δηλαδή ο χρόνος παραμονής σε υψηλή τάση είναι ίδιος με τον χρόνο παραμονής σε χαμηλή τάση σε κάθε περίοδο) και η διαφορά φάσης είναι ακριβώς π/2, μπορούμε να διπλασιάσουμε την ανάλυση των αισθητήρων. Συγκεκριμένα, αν εκτελέσουμε την λογική πράξη ΑΠΟΚΛΕΙΣΤΙΚΟ-Ή στις δύο παλμοσειρές προκύπτει μία νέα παλμοσειρά με διπλάσια συχνότητα. (σχήμα)



Σχήμα 1.23

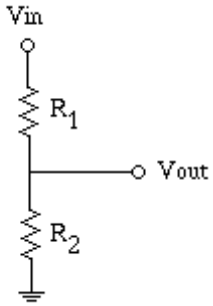
Εαν οδηγούμε με την παλμοσειρά που προέκυψε κάποια συσκευή με δυνατότητα ανίχνευσης ακμών μπορούμε να μετούμε τις δύο ακμές κάθε περιόδου(ανοδική - καθοδική), έχοντας τελικά τετραπλασιάσει την ανάλυση που μας παρέχει ο αισθητήρας.

1.6 Μετατροπείς τάσης απο DC σε DC

Η οδήγηση του κινητήρα ΣΡ απαιτεί μία ελεγχόμενη πηγή συνεχούς τάσης. Η διαδικασία αυτή πραγματοποιείται απο έναν ηλεκτρονικά ελεγχόμενο μετατροπέα απο συνεχή σε συνεχή τάση. Ας μελετήσουμε τρόπους με τους οποίους μπορούμε να αλλάξουμε την τιμή μίας συνεχούς τάσης.

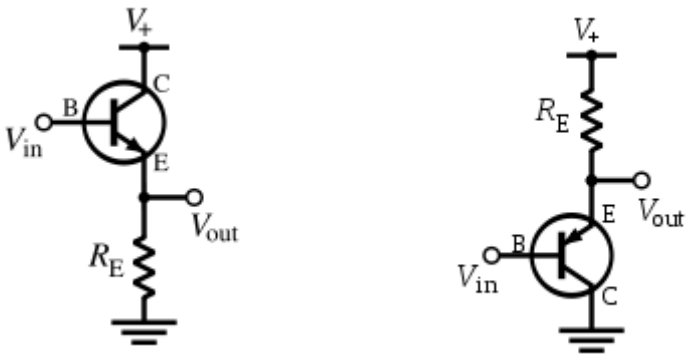
1.6.1 Γραμμικοί μετατροπείς

-Διαιρέτης τάσης. (σχήμα) Ο νόμος του Ohm μας παρέχει έναν απλούστατο τρόπο να μειώσουμε την τιμή της τάσης μεταξύ δύο ακροδεκτών, με το να τοποθετήσουμε δύο εν σειρά αντιστάσεις. Η πτώση τάσης στα άκρα της κάθε αντίστασης είναι ανλάλογη της τιμής της αντίστασής της, αφού ο κλάδος διαρέεται απο κοινό ρεύμα, σύμφωνα με τον νόμο: V = I*R. Φυσικά, η τάση εξόδου μπορέι να είναι μόνο μικρότερη απο την τάση τροφοδοσίας του κυκλώματος. Ωστόσο, η τάση εξόδου εξαρτάται απο την αντίσταση του φορτίου που θα συνδεθεί στους ακροδέκτες εξόδου. Η τιμή, δηλάδή της τάσης επηρρεάζεται έντονα απο το φορτίο. Αυτό συμβαίνει επειδή ο διαιρέτης τάσης ως διθυρο δίκτυο παρουσιάζει μεγάλη αντίσταση εξόδου.

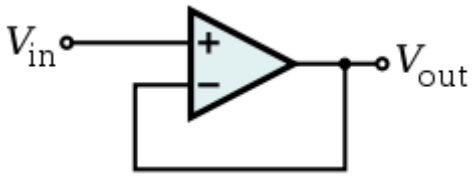


Σχήμα 1.24 Διαιρέτης τάσης

Η τάση εξόδου του διαιρέτη πρέπει να απομονωθεί απο καποια διάταξη η οποία να παρουσιάζει μικρή αντίσταση εξόδου και, επομένως, μικρή ευαισθησία της τάσης με τις διακυμάνσεις του φορτίου. Τέτοιες διατάξεις απομονωτών (buffers) μπορούν να βρεθούν αρκετές, ανάλογα με τις απαιτήσεις ισχύος και ακρίβειας την εφαρμογής. Η απλούστερη υλοποίηση είνα η χρήση ενός διπολικού τρανζίστορ σε συνδεσμολόγία ακόλουθου εκπομπού στην έξοδο του διαιρέτη, ή χρήση τελεστικού ενισχυτή σε συνδεσμολογία απομονωτή.



Σχήμα 1.25 Συνδεσμολογία κοινού συλλέκτη (ακόλουθου εκπομπού)



Σχήμα 1.26 Τελεστικός ενισχυτής σε διαταξη απομονωτή (buffer)

Ενας διαιρέτης τάσης συνδεδεμένος σε ένα κύκλωμα απομονωτή ισχύος αποτελεί το πιο σύνηθες σχεδιο γραμμικού μετατροπέα απο συνεχή σε συνεχή τάση. Ο διαιρέτης μπορεί να αντικατασταθεί απο ένα οποιοδήποτε αναλογικό σήμα τάσης.

Το βασικό μειονέκτημα που παρουσιάζουν οι γραμμικοί μετατροπείς συνεχούς τάσης είναι η απώλεια ισχύος. Οι ακροδέκτες εξόδου του απομονωτή τάσης παρέχουν στο συνδεδεμένο φορτίο ρευμα I_o υπο την τάση εξόδου V_o. Ωστόσο, το ίδιο ρεύμα αντλείται απο την πηγή τροφοδοσίας υπο τάση V_{in} . Αυτό συνεπάγεται πως η καταναλισκόμενη ισχύς είναι πάντοτε I_o*V_{in} , ανεξάρτητα απο την τάση εξόδου. Η ισχύς που δεν παρέχεται στο φορτίο, και είναι ίση με $(V_{in} - V_o)*I_o$ χάνεται ώς θερμότητα στο εσωτερικό του απομονωτή.

Η μεγάλες απώλειες ισχύος προκαλούν σημαντική παραγωγή θερμότητας στο εσωτερικό των γραμμικών μετατροπέων, αυξάνοντας έτσι τις απαιτήσεις τους σε ψύξη. Όπως είδαμε προηγουμένως, το πρόβλημα αυτό μεγιστοποιείται στις περιπτώσεις όπου η τάση εξόδου είναι αρκετά μικρότερη απο την τάση εισόδου.

Να σημειώσω οτι το μειονέκτημα αυτό αποτέλεσε καταλυτικό παράγοντα στην υιοθέτηση της εναλλασσόμενης τάσης στα δίκτυα μεταφοράς ισχύος ανα την ιστορία. Στην περίπτωση της εναλλασσόμενης τάσης η μετατροπή γίνεται ιδιαίτερα αποδοτικά απο τους μετασχηματιστές, μέσω μαγνητικών κυκλωμάτων.

Τα τελευταία χρόνια, ωστόσο, έχει γίνει δυνατή η αποδοτική μετατροπή απο συνεχή σε συνεχή τάση με την χρήση των διακοπτικών μετατροπέων.

1.6.2 Διακοπτικοί μετατροπείς

1.6.2.1 Αρχές λειτουργίας

Πριν αναφέρουμε την λέιτουργία των διακοπτικών μετατροπέων συνεχούς τάσης, ας εξετάσουμε τα χαρακτηριστικά ενός περιοδικού σήματος:

Ενα οποιοδήποτε περιοδικό σήμα μπορεί να αναπτυχθεί σε μία άπειρη ή πεπερασμένη σειρά Fourier. Δηλαδή, μπορεί να αναλυθει ως το άθροισμα άπειρων η πεπερασμένων ημιτόνων και συνημιτόνων συγκεκριμένων συχνοτήτων, πλατών και φάσης. Οι όροι πέρα του σταθερού όρου και της θεμελιώδης συχνότητας ονομάζονται αρμονικές του σήματος. Ας δούμε το ανάπτυγμα ενός περιοδικού σήματος f(x) με περίοδο L σε σειρές fourier για το διάστημα [-L, L]:

$$f(x) \approx \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(\frac{n\pi x}{L}) + \sum_{n=1}^{\infty} b_n \sin(\frac{n\pi x}{L}) \quad \text{, όπου}$$

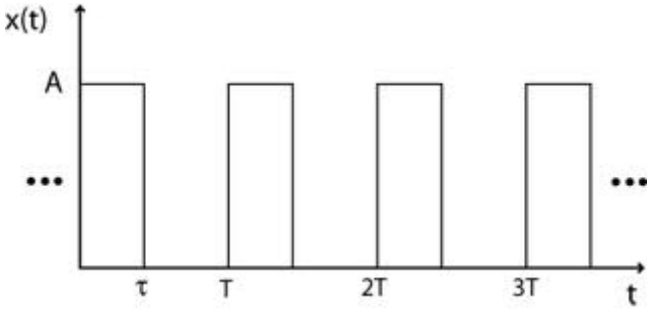
$$a_n = \frac{1}{L}\int_{-L}^{L} f(x)\cos(\frac{n\pi x}{L})dx$$

$$b_n = \frac{1}{L}\int_{-L}^{L} f(x)\sin(\frac{n\pi x}{L})dx$$

Ο πρώτος όρος της σειράς fourier περιγράφει ένα συνημίτονο μηδενικής συχνότητας, επομένως μία σταθερή συνάρτηση. Επειδή κάθε συνημιτονική ή ημιτονική συνάρτηση έχει μέση τιμή μηδέν, ο πρώτος όρος περιγράφει την μέση τιμή της αρχικής συνάρτησης, γι'αυτο και ονομάζεται DC συνιστώσα. Άλλωστε, φαίνεται και απο τον τύπο οτι ο συντελεστής του πρώτου όρου συμπίπτει με τον τύπο εύρεσης της μέσης τιμής μιάς περιοδικής συνάρτησης, αφού για n=0, ο τύπος α_n δίνει:

$$a_0 = \frac{1}{L}\int_{-L}^{L} f(x)dx \text{ , αφού } \cos(0) = 1 \text{ και η περίοδος είναι L}$$

Ας δούμε τώρα ένα ειδικό περιοδικό σήμα, την ακολουθία τετραγωνικών παλμών.

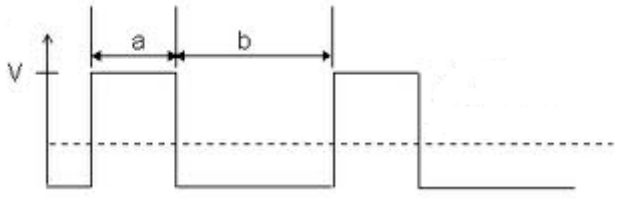


Σχήμα 1.27 Τετραγωνικοί παλμοί πλάτους Α

Η κυματομορφή αυτή αποτελείται απο μια διαδοχή παλμών πλάτους Α και μηδενική τάση ενδιάμεσά τους. η παλμοσειρά αυτή χαρακτηρίζεται απο τρεία μεγέθη: Το πλάτος (Α) , την περίοδο (Τ) και το duty cycle.

Περίοδος ονομάζεται ο χρόνος μέσα στον οποίο η κυματομορφή επαναλαμβάνεται. Την ελάχιστη περίοδο ενός σήματος την ονομάζουμε θεμελιώδη περίοδο. Απο εδώ και στο εξής με τον όρο «περίοδος» θα αναφερομαι στην θεμελιώδη.

Με τον όρο duty cycle αναφερόμαστε στο ποσοστό του χρόνου της περιόδου κατα το οποίο η παλμοσειρά έχει τιμή Α. Παραδείγματος χάριν η παλμοσειρά του επάνω σχηματος έχει duty cycle 50%, ενώ η παλμοσειρά με duty cycle 100% είναι η σταθερή συνάρτηση x(t) = A.



Σχήμα 1.28

Στο σχήμα, duty cycle = $\frac{a}{a+b}$ * 100%

Αν εφαρμόσουμε τον τύπο της μέσης τιμής σε κάποια παλμοσειρά βλέπουμε οτι η μέση τιμή εξαρτάται απο το duty cycle, για σταθερό πλάτος, και μπορεί να πάρει τιμές απο 0 έως Α. Στο σχήμα η μέση τιμή φαίνεται με την διακεκομμένη γραμμή.

Για να κρατήσουμε την μέση τιμή ενός πραγματικού περιοδικού σήματος χρειάζεται με κάποια διάταξη να απομακρύνουμε τις υπόλοιπες συχνότητες. Γενικά η μορφοποίηση ενός σήματος με βάση το συχνοτικό του περιεχόμενο γίνεται με την χρήση των ηλεκτρονικών φίλτρων:

1.6.2.2 Ηλεκτρονικά φίλτρα

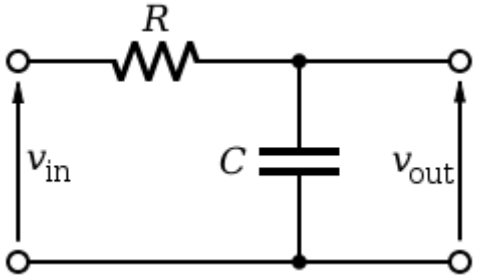
Τα ηλεκτρονικά φίλτρα είναι κυκλώματα τα οποία επεξεργάζονται το σήμα που δέχονται ως είσοδο με αποτέλεσμα να απορροφούν ή να ενισχύουν συγκεκριμένες συχνότητες που περιέχονται σε αυτό. Τα φίλτρα χωρίζονται σε διάφορες κατηγορίες, ανάλογα με την κατασκευή τους. Ετσι έχουμε τα εξής είδη φιλτρων:

- Παθητικά /Ενεργητικα,
- Γραμμικά/μη γραμμικά
- Αναλογικά /Ψηφιακά
- Συνεχούς χρόνου/Διακριτού χρόνου
- Πεπερασμένης κρουστικής απόκρισης/ Μη πεπερασμένης κρουστικής απόκρισης

Στην εργασία αυτή θα μας απασχολήσουν μόνο ιδιαίτερα απλά αναλογικά γραμμικά φίλτρα με παθητικά στοιχεία.Τα φίλτρα αυτά μπορούμε να τα θεωρήσουμε ως δίθυρα δίκτυα τα οποία εμφανίζουν διαφορετική συμπεριφορά ανάλογα με την συχνότητα την οποία δέχονται. Σαν αποτέλεσμα, κάποιες συχνοτικές συνιστώσες του σήματος εισόδου τα διαρρέουν αυτούσια, ενώ κάποιες άλλες απορροφούνται και καταναλώνονται στο εσωτερικό των φίλτρων.

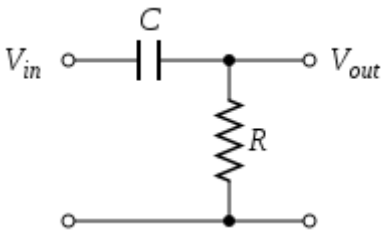
Μια άλλη πολύ βασική κατηγοριοποίηση των φίλτρων αφορά στην συνμπεριφορά τους, και τα διακρίνει στις εξής κατηγορίες:

- Βαθυπερατά φίλτα είναι τα δίθυρα τα οποία εμφανίζουν μεγαλύτερη αντίσταση στις χαμηλές συχνότητες απο ότι στις υψηλές. κατ'άυτον τον τρόπο στην έξοδο τους οι χαμηλές συχνότητες εμφανίζονται σχεδόν αυτούσιες ενώ οι υψηλές εξασθενημένες.Το απλούστερο βαθυπερατό φίλτρο είναι το δίκτυο R-C.



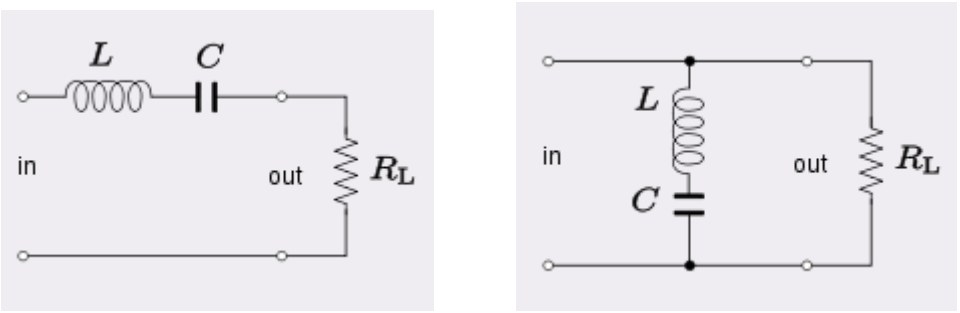
Σχήμα 1.29 Βαθυπερατο φίλτρο

- Υψιπερατά είναι τα φίλτρα που επιτρεπουν την διέλευση στις υψηλές συχνότητες, ενώ καταναλώνουν τις χαμηλές.Το απλούστερο υψιπερατό φίλτρο αποτελείται επίσης απο ένα R-C δίκτυο.



Σχήμα 1.30 Υψιπερατό φίλτρο

- Τέλος, ζωνοπερατά είναι τα φίλτρα που επιτρέπουν την διέλευση σε μία ζώνη συχνοτήτων ενώ απορροφούν όλες τις άλλες, και αντίστοιχα τα ζωνοκοπτικά απορρόφούν μόνο τα σήματα σε μία ζώνη συχνοτήτων.



Σχήμα 1.31 Ζωνοπερατό και Ζωνοκοπτικό φίλτρο

Τα γραμμικά φίλτρα χαρακτηρίζονται και απο την τάξη τους, που δεν είναι παρα το πλήθος των πόλων που έχει η συνάρτηση μεταφοράς τους. Οσο πιο μεγάλη είναι η τάξη τους, τόσο πιο πολλά αντιδραστικά στοιχεία περιλαμβάνει το κύκλωμα και τόσο πιο ακριβής είναι ο διαχωρισμός των συχνοτήτων στις οποίες αλλάζει η συμπεριφορά του φίλτρου.Για παράδειγμα, τα δίκτυα R-C που αναφέρθηκαν προηγουμένως είναι πρώτης τάξης.

Ας μιλήσουμε τώρα για διακοπτικά στιχεία. Οπως θα δούμε στην συνέχεια, αποτελούν το βασικό στοιχείο των διακοπτικών τροφοδοτικών.

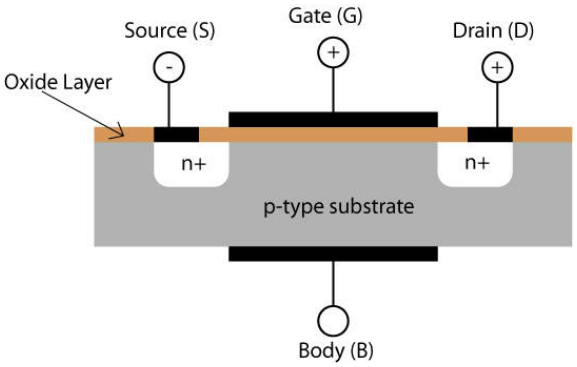
1.6.2.3 Διακοπτικα στοιχεία

Αρχικά, θα χρειαστούμε κάποια ηλεκτρονικά διακοπτικά στοιχεία, δηλαδή στοιχεία που μπορούν να βραχυκυκλώσουν ή να ανοιχτοκυκλώσουν δύο ακροδέκτες και ελέγχονται με ηλεκτρονικό τρόπο.

Τα κατάλληλα στοιχεία για αυτόν τον σκοπό είναι τα τρανζίστορ. Τόσο τα διπολικά όσο και τα τρανζίστορ επίδρασης πεδίου όταν λειτουργούν μεταξύ των περιοχών αποκοπής και κόρου μπορούν να θεωρηθούν ως διακόπτες, αφού εμφανίζουν άλλοτε πολύ μεγάλη και άλλοτε ελάχιστη αντίσταση μεταξύ δύο ακροδεκτών τους.

Το είδος τρανζίστορ που χρησιμοποιείται κατα κόρον για διακοπτικές εφαρμογές είναι το τρανζίστορ επίδρασης πεδίου μετάλλου-οξειδίου-ημιαγωγού (MOSFET). Ο λόγος είναι οι μεγάλες διακοπτικές συχνότητες στις οποίες μπορούν να λειτουργήσουν και η μικρές απώλειες ισχύος που εμφανίζουν.

Ας επικεντρωθούμε στα MOSFET n καναλιού, το οποίο έχει την εξής δομή:



Σχήμα 1.32 Τομή Τρανζίστορ MOSFET

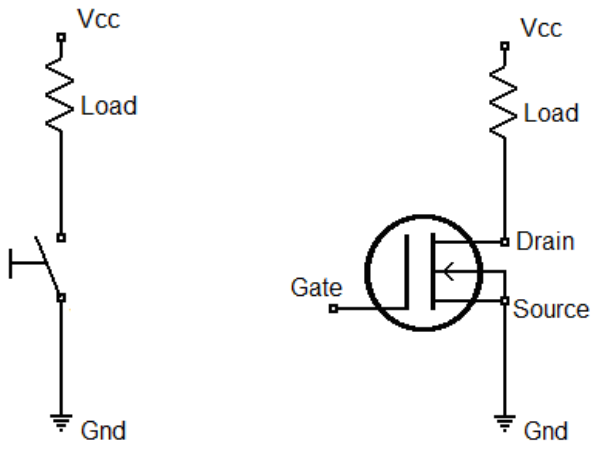
Οι μεταλλικοί ακροδέκτες εμφανίζονται με μάυρο χρώμα, το μονωτικό στρώμα οξειδίου με καφέ, η περιοχή νόθευσης τύπου-p με γκρί , ενώ τύπου-n με άσπρο.Ας δούμε τις καταστάσεις λειτουργίας του τρανζίστορ:

- Εαν η τάση μεταξύ πύλης και πηγής V_{GS} είναι μικρότερη απο την τάση κατωφλίου V_{th}, που αποτελεί κατασκευαστική σταθερά του τρανζίστορ, το τρανζίστορ λέμε οτι βρίσκεται στην αποκοπή, καθώς δεν παρουσιάζει αγωγή μεταξύ πηγής και υποδοχής.
- Εαν $V_{GS} > V_{th}$, και $V_{DS} < (V_{GS} - V_{th})$, δηλαδή η τάση πύλης-πηγής ξεπερνάει την τάση κατωφλίου αλλα η τάση υποδοχής-πηγής είναι μικρότερη απο την διαφορά τους , τότε το τρανζίστορ λέμε οτι λειτουργεί στην γραμμική περιοχή, ή περιοχή τριόδου. Αυτό συμβαίνει επειδή το μέταλλό της πύλης έλκει ηλεκτρόνια προς το μέρος του δημιουργώντας έναν ασθενή αγώγιμο δίαυλό στο υπόστρωμα, μεταξύ πηγής και υποδοχής. η αντίσταση που εμφανίζεται μεταξύ πηγής και υποδοχής εξαρτάται σε μεγάλο βαθμό απο την τάση πύλης και το τρανζίστορ και το τρανζίστορ λειτουργεί ως αντίσταση ελεγχόμενη απο τάση.
- Εαν $V_{GS} > V_{th}$, και $V_{DS} > (V_{GS} - V_{th})$, ο αγώγιμος δίαυλος είναι πλέον αρκετά φαρδύς και η αντίσταση μεταξύ πηγής-υποδοχής εξαρτάται ελάχιστα απο την τάση πύλης. Το τρανζίστορ τότε λέμε οτι βρίσκεται στην περιοχή του κόρου.

Όπως είπαμε και προηγουμένως, εναλλάσοντας τις καταστάσεις λειτουργίας του τρανζίστορ μεταξύ περιοχής αποκοπής και κόρου έχουμε μία λειτουργία όμοια με εκείνη του διακόπτη μεταξύ των ακροδεκτών πηγής και υποδηχής, ελεγχόμενη απο την τάση της πύλης.

1.6.2.4 Διαμόρφωση PWM

Αν συνδέσουμε ένα τρανζίστορ ώς διακοπτικό στοιχείο μεταξύ μίας πηγής και ενός φορτίου, όπως στο σχήμα



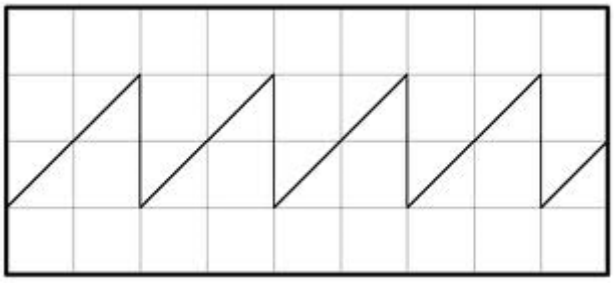
Σχήμα 1.33

ελέγχοντας την τάση της πύλης μπορούμε να εφαρμόσουμε στους ακροδέκτες του φορτίου διαφορά δυναμικού σε μορφή παλμοσειράς, με το να «ανοιγοκλείνουμε» τον διακόπτη όπως περιγράψαμε προηγουμένως. Η παλμοσειρά αυτή θα έχει πλάτος την τιμή της τάσης τροφοδοσίας Vcc, αλλα περίοδο και duty cycle που εξαρτώνται απο την διαχείρηση του διακόπτη . Η μέση τιμή της παλμοσειράς εξαρτάται όπως έχουμε πεί αποκλειστικά απο το duty cycle.

Επειδη η παλμοσειρά είναι τετραγωνική, η ανάλυση σε σειρά fourier θα μας δώσει μία DC συνιστώσα ίση με την μέση τιμή της παλμοσειράς, πρώτη αρμονική με περίοδο την θεμελιώδη περίοδο της παλμοσειράς και επόμενες αρμονικές στα περιττά πολλαπλάσια της θεμελιώδους συχνότητας. Συνδέοντας ένα βαθυπερατό φίλτρο μεταξύ της εξόδου του διακοπτικού στοιχείου και του φορτίου, μπορούμε να εξαλείψουμε τις υψηλές συχνότητες απο την κυματομορφή και να κρατήσουμε μόνο την DC συνιστώσα. Επομένως έχουμε μία σταθερή τάση με τιμή που ελέγχεται απο διαχείρηση της πύλης του τρανζίστορ.

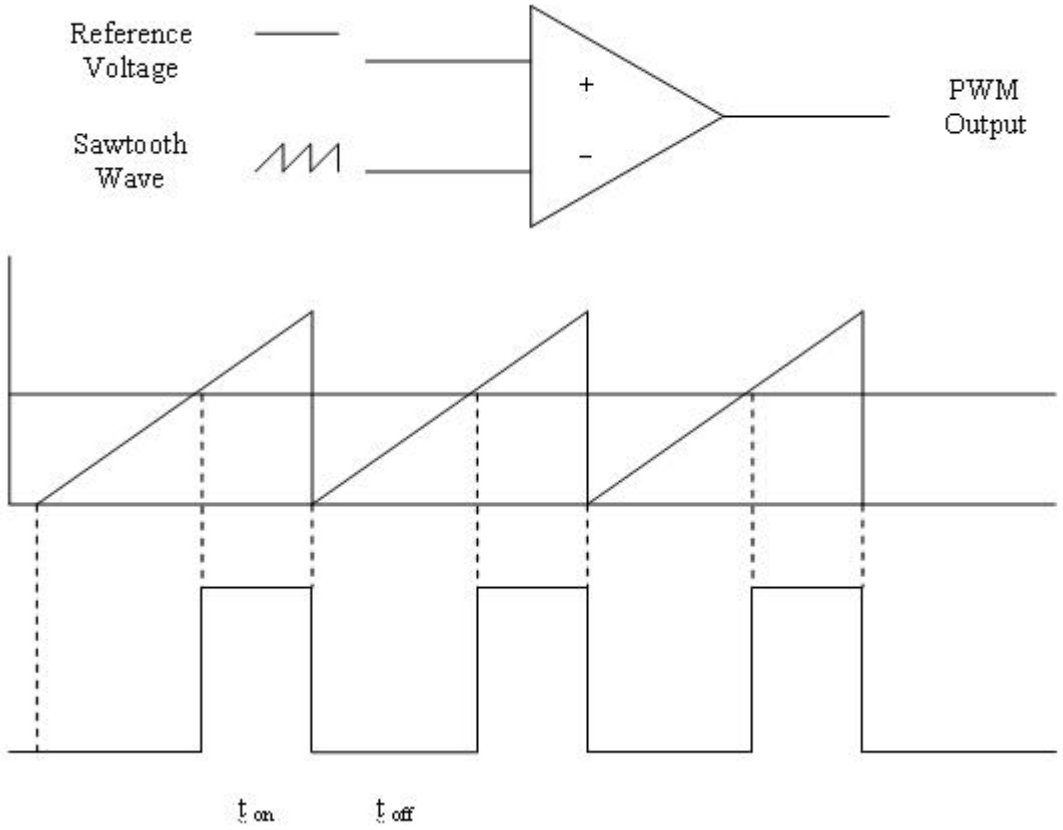
Η τεχνική που αναφέρθηκε, δηλαδή η δημιουργία μίας τιμής τάσης μεσω του ελέγχου του εύρους παλμών μιάς παλμοσειράς, ονομάζεται διαμόρφωση εύρους παλμών (PWM). Συνήθως χρησιμοποιείται σε συνδυασμό με κάποιο βαθυπερατό φίλτρο ωστε να απορρόφούνται οι αρμονικές κοντά στην διακοπτική συχνότητα.

Ας δούμε με ποιόν τρόπο μπορούμε να παράξουμε μία παλμοσειρά με ελεγχόμενο duty cycle. Εστω οτι έχουμε μία πριονωτή περιοδική συνάρτηση με θεμελιώση περίοδο T και πλάτος A:



Σχήμα 1.34

Εφαρμόζουμε την συνάρτηση αυτής στην μία είσοδο ενός συγκριτή, και την άλλη είσοδο την συνδέουμε σε έναν διαιρέτη τάσης με τα άκρα του σε δυναμικά Α και 0, ενώ ο συγκριτής τροφοδοτείται απο τάση V. Η έξοδος του συγκριτή είναι ίση με V,όταν η μη αναστρέφουσα είσοδός του εχει τιμή μεγαλύτερη απο τιμή της αναστρέφουσας εισόδου, και –V όταν συμβαίνει το αντίθετο. Αν πολώσουμε την έξοδο του συγκριτή με δυναμικό V τότε έχουμε μία παλμοσειρά όπως την περιγράψαμε παραπάνω, με πλάτος 2V, συχνότητα ίδια με την συχνότητα της πριονωτής συνάρτησης, και duty cucle που ρυθμίζεται απο την τάση εξόδου του διαιρέτη τάσης στην είσοδο του συγκριτή.



Σχήμα 1.35

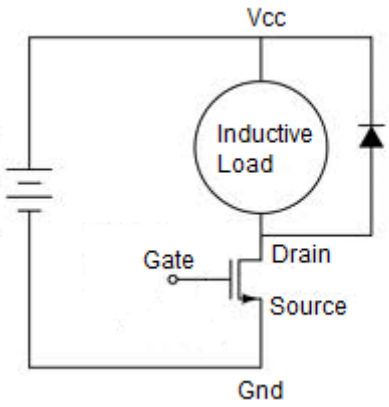
1.7 Κυκλώματα οδήγησης κινητήρων ΣΡ

1.7.1 Οδήγηση μονής κατεύθυνσης

Οι κινητήρες ΣΡ όπως έχουμε αναφέρει είναι ιδιαίτερα απλοί στον έλεγχό τους. Η ταχύτητα περιστροφής τους εξαρτάται απο την επιβαλλόμενη τάση και η διεύθυνση περιστροφής τους απο την πολικότητα της τάσης τροφοδοσίας τους. Στα προηγούμενα αναπτύξαμε τρόπους με τους οποίους μπορούμε να παράξουμε μία συνεχή τάση

οποιασδήποτε τιμής απο ένα τροφοδοτικό συνεχούς τάσης μεγαλύτερης τιμής στα άκρα ενός φορτίου. Στην θέση του φορτίου αυτού μπορεί να τοποθετηθεί το κύκλωμα οπλισμού ενός κινητήρα ΣΡ. Ετσι, έχουμε φτιάξει έναν σύστημα οδήγησης κινητήρα με ελεγχόμενη ταχύτητα.

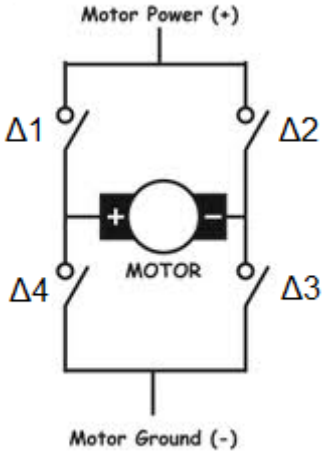
Ωστόσο, επειδή το κύκλωμα οπλισμού του κινητήρα εμφανίζει έντονα επαγωγικό χαρακτήρα, κατα τις απότομες μεταβολές της τάσης στα άκρα του, εμφανίζονται στιγμιαια πολύ υψηλές τάσεις, με αντίθετη πολικότητα απο ότι προηγουμένως. Αυτό συμβαίνει επειδή ο επαγωγέας αντιδράει στις μεταβολές του ρεύματος που τον διαρρέει συμφωνα με την σχέση: $u(t) = L\frac{di(t)}{dt}$, όπου u(t) η τάση στα άκρα του, i(t) το ρεύμα που τον διαπερνάει και L η αυτεπαγωγή του. Επομένως, όσο πιο απότομες είναι οι μεταβολές της τάσης στα άκρα του, τόσο υψηλότερες ριπές τάσης εμφανίζονται, πολλές φορές με τιμές αρκετών χιλιάδων volts. Οι υψηλές αυτές τάσεις παράγουν τόξα που είναι ικανά να βλάψουν τα υπόλοιπα κυκλωματικά στοιχεία. Για αυτόν τον λόγο, όταν συνδέουμε επαγωγικά φορτία σε διακοπτικές διατάξεις , φροντίζουμε να τοποθετήσουμε διόδους προστασίας στα άκρα του φορτίου με πόλωση αντίθετη απο την πόλωσης λειτουργίας του φορτίου. Αυτές οι δίοδοι έχουν σαν σκοπό να απορροφήσουν τις αντιδράσεις του επαγωγέα και ονομάζονται flyback diodes.



Σχήμα 1.36 Απλή οδήγηση κινητήρα ΣΡ

1.7.2 Οδήγηση διπλής κατεύθυνσης – H-bridge

Μέχρι στιγμής έχουμε βρεί ένα ασφαλές κύκλωμα οδήγησης κινητήρα ΣΡ. Αν όμως θέλουμε το κύκλωμά μας να έχει την δυνατότητα να οδηγεί το μοτέρ και με τις δύο φορές περιστροφής, τότε αυτό το σχέδιο δεν είναι επαρκές. Την λύση δίνει η χρήση τριών ακόμα διακοπτικών στοιχείων σε συνδεσμολογία γέφυρας, γνωστή ώς H-bridge.



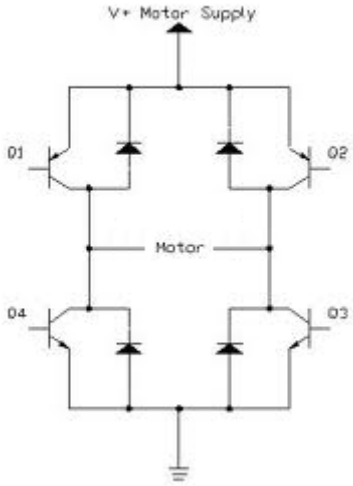
Σχήμα 1.37 Διάταξη H-bridge

Στην διάταξη αυτή ενεργοποιώντας μόνο δύο απο τους τέσσερις διακόπτες κάθε φορά και μάλιστα σε διαγώνιες θέσεις, τροφοδοτούμε τον κινητήρα με τις δύο διαφορετικές πολικότητες (διακόπτες Δ1 και Δ3 ή Δ2 και Δ4) . Φυσικά, κατα την αλλαγή πολικότητας πρέπει να εξασφαλίζουμε οτι δεν θα βρεθούν ποτέ δύο διακόπτες στην ίδια πλευρά σε αγωγή, γιατι έτσι θα βραχυκυκλώναμε την τροφοδοσία με την γή. Η παραπάνω συνδεσμολογία μας δίνει δύο επιπλέον δυνατότητες. Οταν όλοι οι διακόπτες βρίσκονται στην αποκοπή, οι ακροδέκτες του κινητήρα αιωρούνται και αυτός λειτουργεί ως γεννήτρια εν κενώ. Όταν οι δύο επάνω ή οι δύο κάτω διακόπτες και μόνο άγουν, οι ακροδέτες του κινητήρα είναι βραχυκυκλωμένοι και τότε λέμε οτι ο κινητήρας βρίσκεται σε κατάσταση πέδησης, όπου λειτουργεί ως γεννήτρια με μέγιστο φορτίο(διακόπτες Δ1 και Δ2 ή Δ3 και Δ4).

Η οδήγηση της γέφυρας μπορεί να γίνει με διάφορους τρόπους:

- Η παλμοσειρά PWM μπορέι να εφαρμόζεται ταυτόχρονα στα δύο τρανζίστορ που θέλουμε να λειτουργούμε
- Η παλμοσειρά PWM μπορεί να εφαρμόζεται σε ένα πέμπτο τρανζίστορ που συνδέει ολόκληρη την γέφυρα με την γή (ή την τάση τροφοδοσίας), και το ζευγάρι τρανζίστορ που θέλουμε να λειτουργεί, να άγει χωρίς διακοπές.
- Ενας ιδιαίτερος τρόπος οδήγησης που χρησιμοποιείται σε εφαρμογές που απαιτούν μεγάλη ακρίβεια θέσης αφορά την τροφοδότηση του σήματος PWM και στα τέσσερα τρανζίστορ της γέφυρας, με τρόπο ώστε όταν η παλμοσειρά τουPWM βρίσκεται σε υψηλο δυναμικό να άγει το ένα ζευγάρι και όταν βρίσκεται σε χαμηλό δυναμικό το άλλο.Ετσι με το duty cycle του παλμού ελέγχουμε τόσο την ταχύτητα, όσο και την φορά περιστροφής του κινητήρα. Βέβαια, χρειάζονται επιπλέον στοιχεία αντιστροφής του σήματος για το ένα ζευγάρι τρανζίστορ.

Να σημειώσουμε οτι στην συνδεσμολογία της γέφυρας ο κιντήρας δεν τροφοδοτείται με συγκεκριμένη πολικότητα, οπότε δεν είναι δυνατή η τοποθέτηση διόδου προστασίας στα άκρα του. Αντίθετα, 4 δίοδοι προστασίας συνδέονται στα άκρα των διακοπτικών στοιχείων, όπου η φορά διέλευσης του ρεύματος είναι πάντα συγκεκριμένη:



Σχήμα 1.38 Οδήγηση δύο κατευθύνσεων κινητήρα ΣΡ

Πηγή: wikipedia

1.8 Μικροελεγκτές

1.8.1 Εισαγωγή

Με τον όρο Μικροελεγκτής αναφερόμαστε σε ειδικά ολοκληρωμένα συστήματα μικροεπεξεργαστών μαζί με διάφορα περιφερειακά υποσυστήματα. Η αυτόνομία που τους παρέχει η ενσωμάτωση διάφορων μονάδων στο ίδιο πλακίδιο τους καθιστά ιδιαίτερα δημοφιλείς και εύχρηστους για εφαρμογές χαμηλού και μεσαίου κόστους. Χρησιμοποιούνται κατα κόρον σε συστηματα αυτοματισμού, ηλεκτρονικές και ηλεκτρικές συσκευές και κάθε είδους οχήματα. Ενας συνήθης μικροεπεξεργαστής περιλαμβάνει μία αριθμητική και λογική μονάδα, τους καταχωρητές, μνήμη RAM και κάποιες φορές ελεγκτή μνήμης. Αντίθετα,οι μικροελεγκτές διαθέτουν ένα ισχυρό οπλοστάσιο περιφερειακών μονάδων που συνήθως περιλαμβάνει:

- Κύκλωμα συνδετικής λογικής (glue logic) για τη σύνδεση των εξωτερικών μνημών και άλλων περιφερειακών παράλληλης σύνδεσης στην αρτηρία δεδομένων (bus) του επεξεργαστή.
- Μνήμη προγράμματος (τύπου ROM, FLASH, EPROM κλπ) η οποία περιέχει το λογισμικό του συστήματος. Σε κάποια μοντέλα, είναι δυνατό το κλείδωμα αυτής της μνήμης, μετά την εγγραφή της, ώστε να προστατευτεί το περιεχόμενό της από αντιγραφή.
- Μεγάλη ποσότητα μνήμης RAM.
- Μόνιμη μνήμη αποθήκευσης παραμέτρων λειτουργίας (τύπου EEPROM ή NVRAM) η οποία να μπορεί να γράφεται απο τον πυρήνα του μικροελεγκτή. Αυτή η μνήμη έχει, έναντι της FLASH, το πλεονέκτημα της δυνατότητας διαγραφής και εγγραφής οποιουδήποτε μεμονωμένου byte.
- Κύκλωμα αρχικοποίησης (reset).
- Διαχειριστή αιτήσεων διακοπής (interrupt request controller) από τα περιφερειακά.

- Κύκλωμα επιτήρησης τροφοδοσίας (brown-out detection) το οποία παρακολουθεί την τροφοδοσία και αρχικοποιεί ολόκληρο το σύστημα όταν αυτή πέσει κάτω από τα ανεκτα όρια, προλαμβάνοντας έτσι την αλλοίωση των δεδομένων.
- Κύκλωμα επιτήρησης λειτουργίας (watchdog timer) το οποίο αρχικοποιεί το σύστημα, αν αυτό εμφανίσει σημάδια δυσλειτουργίας λόγω κολλήματος (hang).
- Τοπικό ταλαντωτή για την παροχή παλμών χρονισμού (internal clock).
- Έναν ή περισσότερους χρονιστές-μετρητές υψηλής ταχύτητας (hardware timer-counter) για τη δημιουργία καθυστερήσεων, μέτρηση διάρκειας γεγονότων, απαρίθμηση γεγονότων και άλλων λειτουργιών ακριβούς χρονισμού.
- Ρολόι πραγματικού χρόνου (Real Time Clock, RTC) το οποίο τροφοδοτείται από ανεξάρτητη μπαταρία και γι αυτό πρέπει να έχει πολύ χαμηλή κατανάλωση ρεύματος.
- Σειρά ανεξάρτητων ψηφιακών εισόδων και εξόδων (Parallel Input-Output, PIO).

,ενώ συχνά ενσωματώνονται και πιο εξειδικευμένα υποσυστήματα, όπως:

- Μία ή περισσότερες ασύγχρονες σειριακές θύρες επικοινωνίας (Universal Asynchronous Receiver Transmitter, UART).
- Σύγχρονες σειριακές θύρες επικοινωνίας (πχ I^2C, SPI, Ethernet).
- Ολόκληρα υποσυστήματα για την άμεση υποστήριξη από υλικολογισμικό (hardware) των πιο σύνθετων πρωτοκόλλων επικοινωνίας όπως CAN, HDLC, ISDN, ADSL.
- Μονάδα άμεσης εκτέλεσης πράξεων κινητής υποδιαστολής (Floating Point Processing Unit, FPU), η οποία είναι πάντοτε πιο γρήγορη από την ALU του επεξεργαστή. Τέτοιες μονάδες χαρακτηρίζουν τους μικροελεγκτές με δυνατότητες ψηφιακής επεξεργασίας σήματος (Digital Signal Processing, DSP). Τα τελευταία χρόνια, με την ευρύτατη διάδοση των φορητών συσκευών ήχου και εικόνας, παρατηρείται μια τάση σύγκλισης τωνν μικροελεγκτών με τους DSP.
- Περισσότερες από μία εισόδους για μετατροπή αναλογικού σήματος σε ψηφιακό (Analog to Digital converter, ADC).
- Μετατροπέα ψηφιακού σε αναλογικό σήμα (Digital to Analog converter, DAC).
- Ελεγκτή οθόνης υγρών κρυστάλλων (Liquid Crystal Display, LCD).
- Υποσύστημα προγραμματισμού πάνω στο κύκλωμα (τύπου ISP, βλ. παραπάνω). Χάρη σε αυτό το κύκλωμα, είναι δυνατός ο επαναπρογραμματισμός (αναβάθμιση λογισμικού) της εφαρμογής, συνδέοντας στη συσκευή μια εξωτερική συσκευή προγραμματισμού (συνήθως σε θύρα UART RS232) ή ακόμη και από το διαδίκτυο, . Αυτή η δυνατότητα απαιτεί την προϋπαρξη λογισμικού υποδοχής (bootstrap) μέσα στη μνήμη προγράμματος και επομένως δεν μπορεί να γίνει σε τελείως άδεια μνήμη προγράμματος.
- Υποσύστημα προγραμματισμού (τύπου ISP) και διάγνωσης (συνήθως είναι το καθιερωμένο πρότυπο JTAG). Χάρη σε αυτό, είναι δυνατός ο προγραμματισμός της μνήμης προγράμματος χωρίς να προαπαιτείται κάποιο πρόγραμμα υποδοχής. Γι αυτό το λόγο, είναι ιδιαίτερα χρήσιμο στον αρχικό προγραμματισμό, πχ κατά τη συναρμολόγηση, ή σε περίπτωση σφάλματος (bug) στο λογισμικό υποδοχής το οποίο να καθιστά αδύνατη την κανονική αναβάθμιση.

1.8.2 Μικροελεγκτής AVR Atmega16 της εταιρείας ATMEL

Στην εργασία αυτή έχει επιλεχθεί ο μικροελεγκτής AVR ATmega16 της εταιρείας ATMEL.Ο μικροελεγκτής αυτός περιλαμβάνει επιγραμματικά τα εξής:
επεξεργαστή αρχιτεκτονικής RISC που δέχεται εντολές μήκους 8 bits

- Μνήμη Flash 16 Kbyte, SRAM 1Kbyte και EEPROM 512 byte
- δυνατότητα προγραμματισμού ISP
- 2 χρονιστές-μετρητές των 8 bits και έναν των 16 bits
- 32 προγραμματιζόμενες θύρες εισόδου-εξόδου
- 8 κανάλια μετατροπέων αναλογικού σε ψηφιακό σήμα με ανάλυση 10 bits
- αναλογικό συγκριτή
- κ.α.

Παρατίθεται ο κατάλογος χαρακτηριστικών του συγκεκριμένου μικροελεγκτή, όπως κυκλοφορεί απο την κατασκευάστρια εταιρεία με τις λεπτομερείς τεχνικές πληροφορίες. Ωστόσο, θα αναλύσουμε κάποια περιφερειακά που είναι ιδιαίτερα σημαντικά για την υλοποίηση που συζητάμε.

Θύρες εισόδου-εξόδου

Όλες οι θύρες εισόδου-εξόδου του μΕ είναι τρισταθείς και ανεξάρτητα προγραμματιζόμενες. Αυτό σημαίνει οτι ο κάθε ακροδέκτης μπορεί να λειτουργήσει ως είσοδος ή ως έξοδος, χωρίς να επηρεάζει την λειτουργία των υπόλοιπων. Οι καταστάσεις στις οποίες μπρούν να βρεθούν είναι λογικό 0, λογικό 1 και κατάσταση υψηλής αντίστασης. Όταν ένας ακροδέκτης έχει ρυθμίστεί ώς έξοδος, μπορεί να τροφοδοτήσει ή να γειώσει μέχρι και 20 mA ρεύμα, οδηγώντας απ'ευθείας λαμπτήρες LED. Όταν ένας ακροδέκτης ρυθμιστεί ως είσοδος, υπάρχει η δυνατότητα να συνδεθεί με μία εσωτερική pull up αντίσταση η οποία ωθεί το δυναμικό στο λογικό 1 όταν δεν συνδέεται εξωτερικά με την γή. Ο έλεγχος της συμπεριφοράς της κάθε θύρας γίνεται μέσω τριών 8-bit καταχωρητών:

- Ο καταχωρητής DDRx, δέχεται έναν 8-bit αριθμό. Το κάθε bit αυτού του αριθμού αντιστοιχεί σε έναν ακροδέκτη της θύρας x και η τιμή του ρυθμίζει την φορά δεδομένων (1: output, 0: input)
- Ο καταχωρητής PORTx έχει διπλή λειτουργία. Εαν η θύρα x έχει ρυθμιστεί ώς έξοδος, θέτουμε το αντίσοιχο bit στην τιμή που θέλουμε να εμφανίσει κάποιος ακροδέκτης. Εαν είναι είσοδος, η τιμή του κάθε bit αφορά την ενεργοποίηση ή όχι της εσωτερικής pull up αντίστασης για τον αντίστοιχο ακροδέκτη Οταν δεν έχει ενεργοποιηθεί η εχωτερική αντίσταση ο ακροδέκτης λειτουργεί σε κατάσταση υψηλής αντίστασης.
- Ο καταχωρητής PINx έχει χρήσιμότητα μόνο στην περίπτωση που κάποιος ακροδέκτης έχει οριστεί ώς είσοδος. Τότε, η λογική τιμή του κάθε ακροδέκτη βρίσκεται στο αντίστοιχο bit του καταχωρητή. Αναλυτική περιγραφή της λειτουργλιας των θυρών I/O βρίσκεται στα τεχνικά χαρακτηριστικά.

Χρονιστές-μετρητές – fast PWM.

Οι χρονιστές του μΕ είναι δομές υλικού οι οποίες χρονίζονται απο το ρολόι του μΕ. Η συχνότητα λειτουργίας τους μπορεί να είναι η ίδια ή κάποιο κλάσμα της συχνότητας ρολογιού με την χρήση των prescalers. Ουσιαστικά η λειτουργία τους περιλαμβάνει την αύξηση της τιμής ενός καταχωρητή κατα μία μονάδα σε κάθε πυροδότηση απο το ρολόι. Η συσσώρευση σταματάει όταν ο καταχωρητής υπερχειλίσει ή όταν η τιμή του φτάσει μία καθορισμένη τιμή.Σ την συνέχεια μηδενίζεται ή μετράει αντίστροφα. Οι λειτουργίες αυτές σε συνδυασμό με διάφορα χρήσιμα εργαλεία του μΕ δίνουν πολλές δυνατότητες. Για παράδειγμα δίνεται η δυνατότητα κάθε φορά που η τιμή του χρονιστεί φθάνει μια συγκεκριμένη τιμή, να πυροδοτείται μία διακοπή.Μία άλλη διευκόλυνση είναι η παραγωγή σήματος PWM αποκλειστικά σε επίπεδο υλικού.

Με την ρύθμιση ενός καταχωρητή, ο χρονιστής ρυθμίζεται να παράγει σήμα PWM. Σε αυτήν την ρύθμιση, ο χρονιστής καταμετρά μέχρι κάποια συγκεκριμένη τιμή, μετά μηδενίζει και συνεχίζει ξανά. Ενας άλλος καταχωρητής προγραμματίζεται με μία τιμή κατωφλίου. ενας ακροδέκτης του μΕ μπορεί συνεχώς να συγκρίνει την τιμή του χρονιστή με το κατώφλι

αυτό και να εμφανίζει διαφετική τιμή αν η διαφορά τους είναι θετική ή αρνητική. Κατ'αυτόν τον τρόπο, στον ακροδέκτη αυτόν παράγουμε ένα σήμα PWM με duty cycle που ρυθμίζεταιαι απο την τιμή κατωφλίου και συχνότητα που ρυθμίζεται απο την τιμή μηδενισμού του χρονιστή.

Αναλυτικά η λειτουργία των χρονιστών υπάρχει στις τεχνικές πληροφορίες του μικροελεγκτή.

Διακοπές.

Οι διακοπές είναι ένα πολύ ισχυρό εργαλείο των μΕ. Πρόκειται για τμήματα κώδικα τα οποία είναι ανεξάρτητα απο το υπόλοιπο σώμα του αλγορίθμου. Οταν η εκτέλεση τους ενεργοποιηθεί η κανική εκτέλεση του προγράμματος σταματάει και παρεμβάλλεται η εκτέλεση της ρουτίνας εξυπηρέτησης της διακοπής που προέκυψε. Οταν τερματίσει, η εκτέλεση συνεχίζιε απο εκεί που σταμάτησε.

Η εκτέλεση μιας διακοπής μπορεί να πυροδοτηθεί απο διάφορες συνθήκες, ανάλογα με τον ορισμό της. Συνηθης χρήση είναι η πυροδότηση σε κάθε υπερχείληση κάποιου χρονιστή. Αυτό προκαλεί εκτέλεση της διακοπής ανα σταθερά χρονικά διαστήματα.

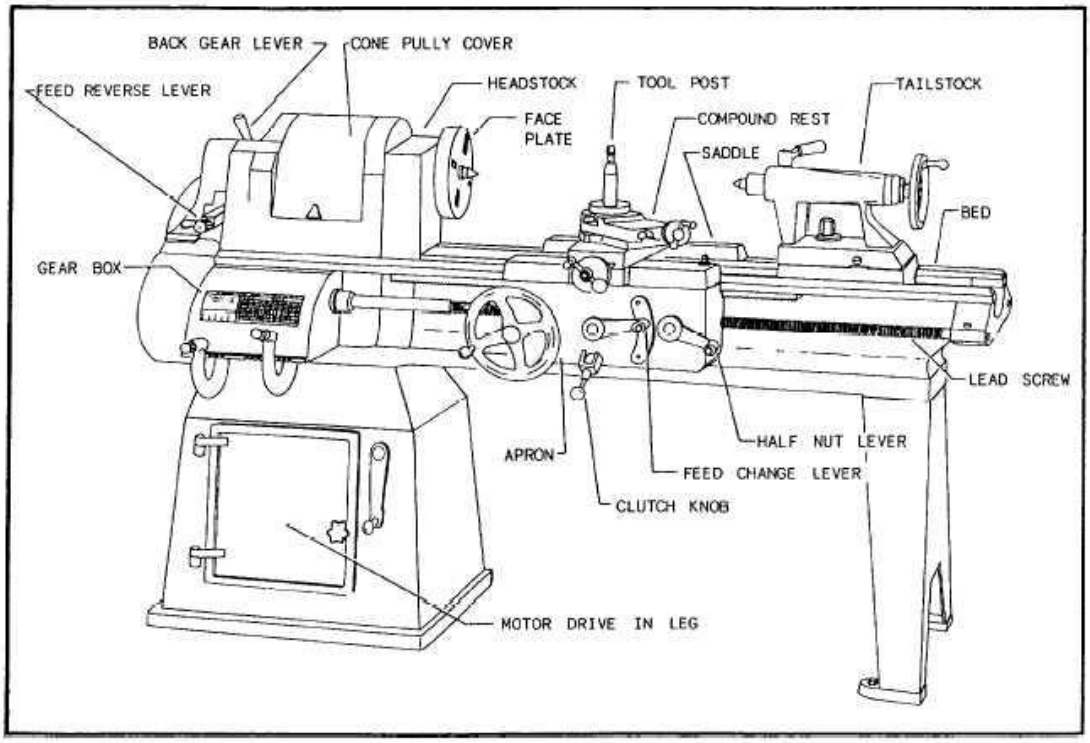
Υπάρχουν επίσης και οι εξωτερικές διακοπές. Αυτές ενεργοποιούνται απο την έλευση κάποιου παλμού στον αντίστοιχο ακροδέκτη του μΕ. Στον Atmega16 υπαρχουν τρείς τέτοιοι ακροδέκτες. η πυροδότηση της διακοπής μπορεί να γίνεται στην θετική ακμή του παλμού, την αρνητική ή στην κατάσταση λογικού «0».

Περισσότερα για την λειτουργλια των διακοπών υπάρχουν στα τεχνικά χαρακτηριστικά του μΕ.

1.9 Τορνος. Λειτουργία-εφαρμογές

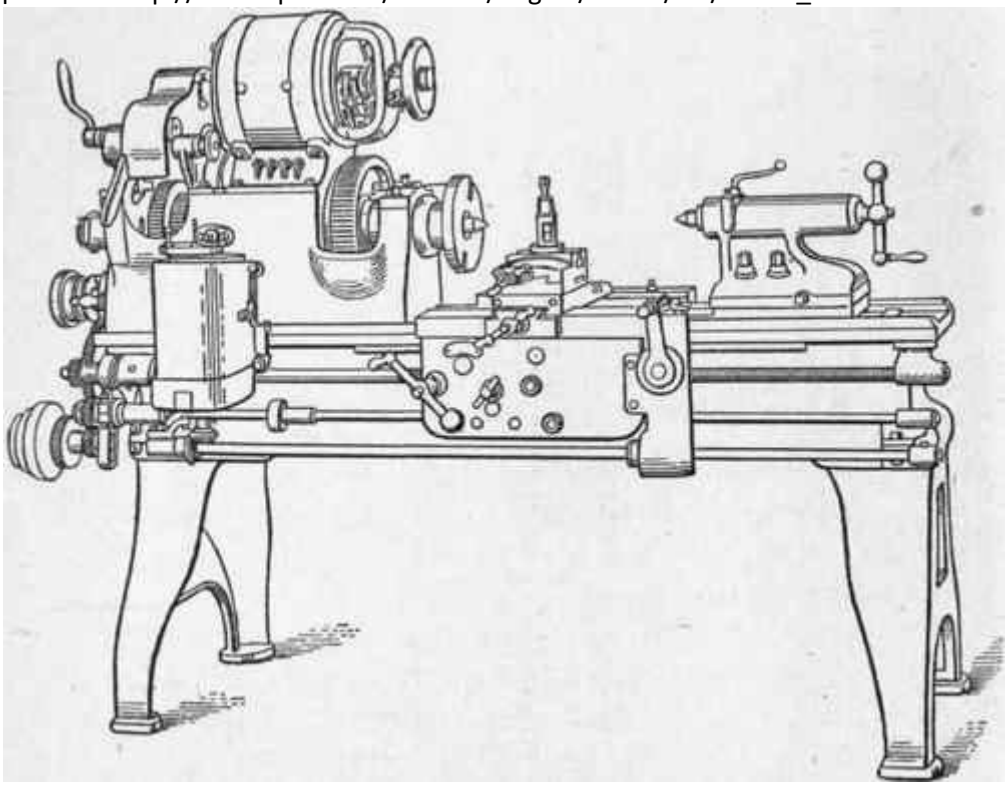
Η πρακτική εφαρμογή του συστήματος της εργασίας αυτής αφορά στον αυτοματισμό του τόρνου. Γι'αυτόν τον λόγο, αναφέρω κάποια στοιχεία για την δομή και την λειτουργία αυτού του μηχανήματος.

Ο τόρνος είναι μία εργαλειομηχανή που χρησιμοποιείται για την επεξεργασία και κατασκευή εξαρτημάτων κυλινδρικής συμμετρίας. Το υπο επεξεργασία τεμάχιο συγκρατείται απο έναν σφικτήρα και περιστέφεται γύρω απο κάποιον άξονα. Διάφορα κοπτικά εργαλεία αλληλεπιδρούν με το τεμάχιο ,κινούμενα είτε αξονικά είτε ακτινικά ως προς την φορά περοστροφής του τεμαχίου. Η συνήθεις χρήσεις περιλαμβάνουν κόψιμο, διαμόρφωση, διάνοιξη αξονικών τρυπών, λείανση, εντύπωση αναγλύφου, κοπή σπειρώματος.Η δομή ενός τυπικού τόρνου φαινεται στην φωτογραφία:



Σχήμα 1.39

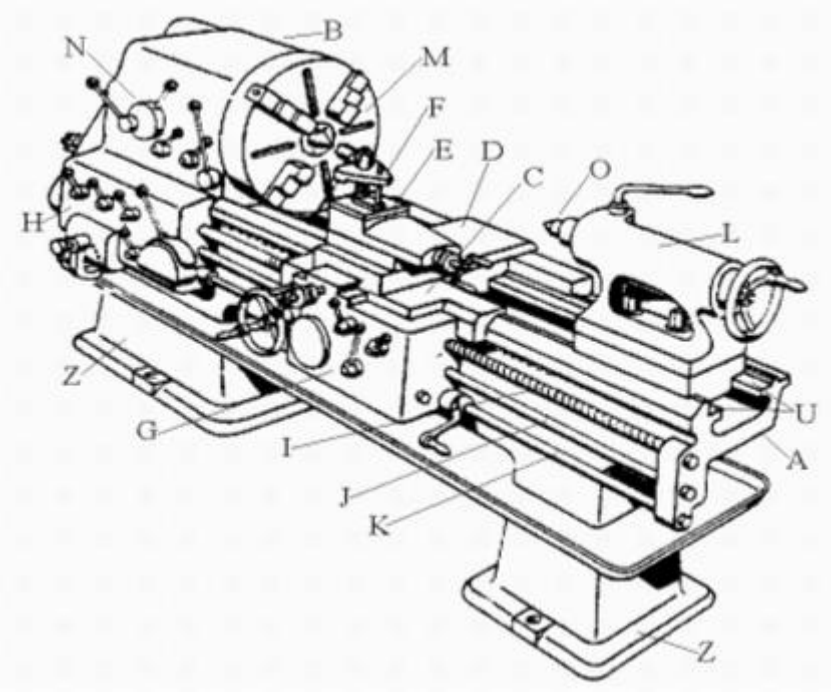
πηγή: http://www.tpub.com/content/engine/14076/css/14076_161.htm



Σχήμα 1.40

πηγη: http://chestofbooks.com/home-improvement/woodworking/Lathe-Operation/Chapter-XXII-Electrically-Driven-Lathes.html

Σχήμα 1.41



Σχήμα 1.42
πηγή: http://www.datuopinion.com/torno

Τα στοιχεία ενός τυπικού τόρνου είναι τα εξής, όπως φαίνονται και στην φωτογραφία:

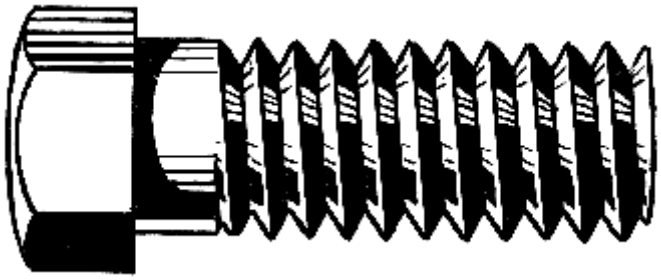
Α – Το «κρεββάτι» του τόρνου. Πρόκειται για το σώμα του μηχανήματος, πάνω στο οποίο βρίσκονται οι αγωγοι ολίσθησης όλων των κινούμενων στοιχείων.
Β – η κεφαλή του τορνου. Πρόκειται για τον χώρο που φιλοξενεί τα στοιχεία μετάδοσης ισχύος απο το μοτέρ, κάποιους συμπλέκτες και τα έδρανα κύλισης του άξονα του τσωκ.
C – Το σεπόρτι. Πρόκειται για ένα σύστημα το οποίο ολισθαίνει κατα μήκος του κρεββατιού παράλληλα με τον άξονα περιστροφής οτυ τσώκ. Φέρει το σύστημα πάκτωσης του εργαλείου κοπής και ειδικός συμπλέκτη που επιτρέπει την οδήγησή του απο έναν κοχλία
D – Σύστημα κίνησης με φορά ακτινική ως προς την κίνηση του τσώκ, το οποίο είναι πακτωμένο επάνω στο σεπόρτι και φέρει τον εργαλειοδέτη.
E - Δευτερεύον σύστημα κίνησης παράλληλα με την κίνηση του σεπόρτι. Είναι πακτωμένο επάνω στο σύστημα D και φέρει τον εργαλειοδέτη
F – Εργαλειοδέτης. Δομή πάκτωσης του εργαλείου κοπής
G – Συμπλέτης σεπόρτι. Πρόκειται για μηχανισμό δέσμευσης του σεπόρτι επάνω στον κοχλιωτό άξονα ώστε να οδηγείται απο αυτόν. απεμπλοκή του συμπλέκτη σποδεσμεύει το σεπόρτι
Η – Κιβώτιο ταχυτήτων. Επιλογέας της σχέσης μετάδοσης μεταξύ του άξονα του τσώκ και του κοχλιωτού άξονα που δύναται να μεταφέρει το σεπόρτι. Χρησιμευει για εργασίες λείανσης και κοπής σπειρωμάτων
I – κοχλίας οδήγησης του σεπόρτι. οπως είπαμε και πρίν, λαμβάνει ισχύ απο τον άξονα του τσώκ και μετακινεί το σεπόρτι, εφ’όσον ο αντίστοιχος συμπλέκτης βρίσκεται σε σύμπλεξη.
J – K – Πρόκεινται για οδηγούς παράλληλους στον κοχλία που υποβοηθούν την κύλιση του σεπόρτι. Ενίοτε χρησιμεύουν και για τοποθέτηση μηχανισμών ασφαλείας.
L – Κεντροφορέας ή «κουκουβάγια» του τόρνου. Πρόκειται για έναν μηχανισμό στήριξης με οπή ακριβώς παράλληλη και ομοκεντρική με τον άξονα του τσώκ.

Χρησιμέυει για την αντιστήριξη του τεμαχίου υπο επεξεργασία. επίσης, δέχεται δευτερεύον τσωκ για την διάνοιξη αξονικών οπών στο τεμάχιο.
Μ – Τσώκ. Ο μηχανισμός συγκράτησης του τεμαχίου υπο επεξεργασία. Περιστρέφεται δεχόμενο ισχύ απο το μοτέρ του τόρνου
Ν – Επιλογέας τοτης ταχύτητας περιστροφής του τσώκ. Ρυθμίζει τον τρόπο μετάδοσης της ισχύος απο το μοτέρ στον άξονα του τσώκ
Ο – Κέντρο του τόρνου. Πρόκειται για κωνικό εξάρτημα που τοποθετείται στην οπή του κεντροφόρου και στηρίζει το τεμάχιο που βρίσκεται υπο επεξεργασία.
U – Οδηγοί ολίσθησης. βρίσκονται επάνω στο «κρεββάτι» του τόρνου και επιτρέπουν την ολίσθηση των κινούμενων μερών μόνο προν της αξονική διεύθυνση.
Z – Βάση στήριξης του μηχανήματος. Πίσω απο αυτήν, απο την μεριά της κεφαλής του τόρνου βρίσκεται πακτωμένος ο κινητήρας του μηχανήματος, ο οποίος μεταφέρει ισχύ στον άξονα του τσώκ μέσω ιμάντων και τροχαλιών.

1.10 Σπείρωμα - κοχλίες

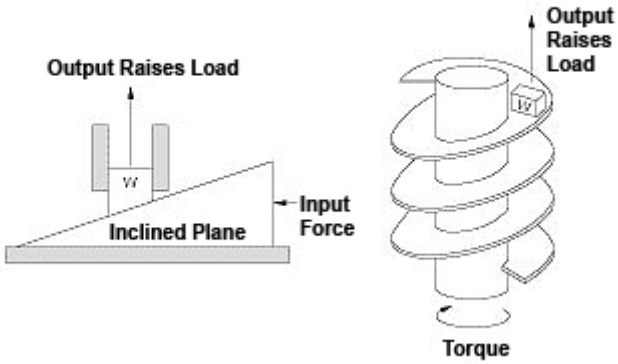
Όπως αναφέραμε, η εφαρμογή του συστήματος στον αυτοματισμό τόρνου δίνει την δυνατότητα στον χρηστη του μηχανήματος να τον χρησιμοποιήσει ωστε να δημιουργήσει σπειρώματα σε διάφορα τεμάχια υπο επεξεργασία. Ας δούμε τί ακριβώς είναι το σπείρωμα:

Σπείρωμα ονομάζεται η απλούστερη και πιο διαδεδομένη δομή για την μετατροπή της περιστροφικής κίνησης σε γραμμική. Πρόκειται για μία έλικα (σπείρα) τυλιγμένη γύρω απο έναν κύλινδρο.



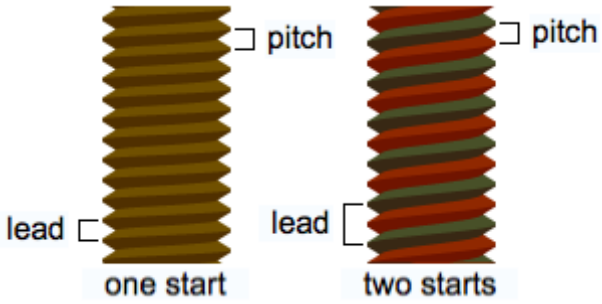
Σχήμα 1.43

Ο κοχλίας λειτουργεί ακριβώς όπως το κεκλιμένο επίπεδο. Το βάρος ενός σώματος επάνω σε ένα κεκλιμένο επίπεδο αναλύεται σε δύο συνιστώσες, μία κάθετη και μία παράλληλη στο επίπεδο. Η παράλληλη συνιστώσα ωθεί το σώμα σε ολίσθηση επάνω το επίπεδο. Η κλίση του επιπέδου καθορίζει την τιμή της συνιστώσας αυτής. Κατ' αντιστοιχία, η περιστροφική κίνηση του κοχλία έχει σαν αποτέλεσμα η έλικα του να ασκεί και μία αξονική δύναμη σε ένα σώμα που αλληλεπιδρά με αυτήν.



Σχήμα 1.44
πηγή: http://www.roton.com/application_engineering.aspx

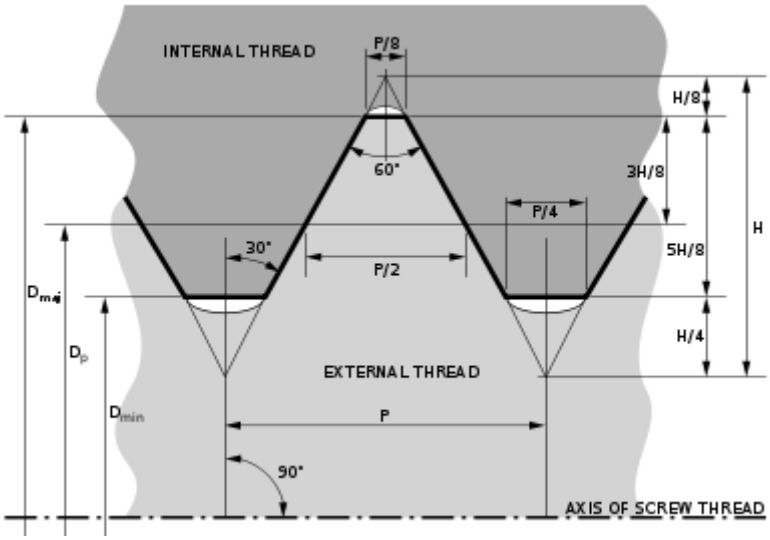
Και σε αυτή την περίπτωση, η κλίση της έλικας καθορίζει το ποσοστό απο την ροπή που ασκείται στον κοχλία, το οποίο θα ωθήσει αξονικά το φορτίο. Η κλίση αυτή λειτουργεί ώς μείωση, αφού μεγαλύτερο τόξο περιστροφής προκαλεί μικρή αξονική μετατόπιση. Ετσι, δίνεται ατην χρήστη του κοχλία μηχανικό πλεονέκτημα δύναμης. Χαρακτηριστικό μέγεθος ενός κοχλία είναι το «βήμα» του (lead). Δηλαδή, το μήκος κατα το οποίο ο κοχλίας ή το φορτίο του μετακινείται αξονικά σε μία περιστροφή του κοχλία. Αυτό συνήθως ταυτίζεται με την απόσταση δύο διαδοχικών τμημάτων του σπείρας πάνω σε μία αξονική γραμμή ως προς τον κύλινδρο .Το μέγεθος αυτό ονομάζεται pitch. Ωστόσο, υπάρχουν κοχλίες πολλαπλών διαδρομών όπου τα μεγέθη lead και pitch δεν ταυτίζονται.



Σχήμα 1.45
πηγή: wikipedia

Στην εργασία αυτή θα ασχοληθούμε κυρίως με σπειρώματα μονής διαδρομής.

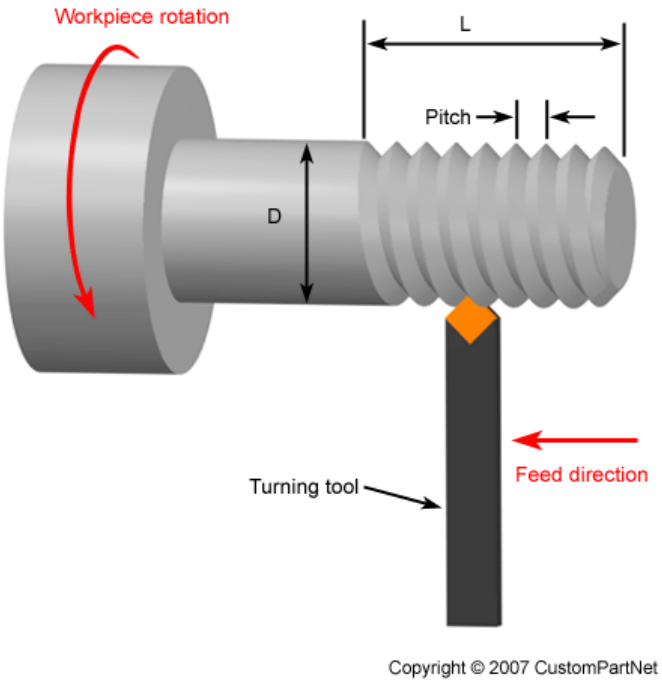
Άλλα χαρακτηριστικά μεγέθη ενός σπειρώματος είναι η γεωμετρία του, δηλαδή το σχήμα της κάθε σπείρας, η εξωτερική και η εσωτερική διάμετρός του. Τα διάφορα είδη σπειρωμάτων έχουν τυποποιηθεί έντονα λόγω της ευρείας χρήσης τους. Ενδεικτικά παρουσιάζω την μορφολογία των μετρικών σπειρωμάτων, που είναι και η πιο δημοφιλής τυποποίηση:



Σχήμα 1.46 Τομή βασικού κατα ISO μετρικού σπειρώματος
πηγή: wikipedia

Ας δούμε την διαδικασία κατασευής ενός σπειρώματος με την χρήση τόρνου:

Ο κύλινδρος στον οποίο θέλουμε να δημιουργήσουμε το σπείρωμα στερεώνεται στο τσώκ του τόρνου, και περιστρέφεται γύρω απο τον άξονά του. η αρχική διάμετρος του κυλίνδρου πρέπει να είναι μεγαλύτερη ή ίση με την εξωτερική διάμετρο του επιθυμητού σπειρώματος, αφού θα χρησιμοποιηθεί τεχνική αφαίρεσης υλικού και όχι προσθήκης. Ενα κοπτικό εργαλείο με επιφάνεια κοπής στο επιθυμητό σχήμα τοποθετείται στον εργαλειοδέτη με διεύθυνση κάθετη ως προς την επιφάνεια του κυλίνδρου. Οταν το εργαλείο κοπής αλληλεπιδρά με τον κύλινδρο, το σεπόρτι (που φέρει τον εργαλειοδέτη) πρέπει να εκτελεί τέτοια ευθύγραμμη κίνηση, ωστε να εξασφαλίζεται οτι διανύει συγκεκριμένη απόσταση ανα περιστροφή του κυλίνδρου. Η απόσταση αυτή είναι, προφανώς, ίση με το βήμα του σπειρώματος που επιθυμούμε να κατασκευάσουμε. Η διαδικασία αυτή επιτυγχάνεται με την σύνδεση του κοχλία του σεπόρτι με τον άξονα του τσώκ μέσω συγκεκριμένης σχέσης μετάδοσης.



Σχήμα 1.47
πηγή: http://www.custompartnet.com/wu/machining

Συνήθως τα κοπτικά εργαλεία κόβουν προς μία μόνο κατεύθυνση. Επομένως, η φορά περιστροφής του κυλίνδρου είναι αυτή της φωτογραφίες όσο το τεμάχιο επεξεργάζεται. Η διεύθυνση κίνησης του σεπόρτι καθορίζει την φορά του σπειρώματος. Στην αγορά υπάρχουν τόσο δεξιόστροφα όσο και αριστερόστροφα σπειρώματα, αν και τα πρώτα χρησιμοποιούνται στην συντριπτική πλειοψηφία των εφαρμογών.

Τελος, η κοπή του σπειρώματος είναι πολύ δύσκολο να ολοκληρωθεί σε ένα στάδιο. Συνήθως το βάθος κοπής που απαιτείται είναι αρκετά μεγαλύτερο απο την ποσότητα υλικού που μπορούμε να αφαιρούμε με το κοπτικό εργαλείο. Αυτό συμβαίνει επειδή η διαδικασία κοπής πρέπει να βρίσκεται μέσα σε κάποια πλάισια ασφαλούς λειτουργίας και η ποιότητα της επιφάνειας να πληροί κάποιες προδιαγραφές. Ετσι, απαιτούνται επαναλαμβανόμενα περάσματα του κοπτικού εργαλείου απο την ίδια διαδρομή με ολοένα αυξανόμενο βάθος διείσδησης , ώστε να προκύψει το τελικό αποτέλεσμα.Το γεγονός αυτό εισάγει αρκετές δυσκολίες στον αυτοματισμό της διαδικασίας, τις οποίες θα αναλύσουμε σε επόμενη ενότητα.

Κεφάλαιο 2: Σχεδιασμός συστήματος

2.1 Μεθοδος μέτρησης σφάλματος

Εχοντας παρουσιάσει αναλυτικά τον σκοπό της εργασίας και τα στοιχεία τα οποία θα χρησιμοποιήσουμε, ας προχωρήσουμε στον σχεδιασμό. Αρχίζουμε με την ανάλυση του σκεπτικού στο οποίο θα στηριχθούμε.

Ας συνοψίσουμε το πρόβλημα: Εχουμε έναν άξονα αναφοράς με προσαρμοσμένο έναν οπτικό κωδικοποιητή ως αισθητήρα και έναν σερβοκινητήρα, στον άξονα του οποίου έχουμε τοποθετήσει άλλον έναν οπτικό κωδικοποιητή. Ζητούμενο είναι να ελέγξουμε κατάλληλα την τροφοδοσία του σερβοκινητήρα ωστε ο άξονάς του να περιστρέφεται με συγκεκριμένη σχέση μετάδοσης ως προς τον άξονα αναφοράς. Ας θεωρήσουμε τις παλμοσειρές εξόδου των δύο αισθητήρων και ας ονομάσουμε την παλμοσειρά εξόδου του αισθητήρα στον άξονα αναφοράς m(t) και του σερβοκινητήρα s(t). Με τα γράμματα m και s θα αναφερόμαστε και στους αντίστοιχους άξονες. (master- slave axis).Το ζητούμενο ανάγεται στην απαίτηση η συχνότητα της παλμοσειράς m να είναι ανα πάσα στιγμή το γινόμενο της συχνότητας της παλμοσειράς s πολλαπλασιασμένης επι έναν σταθερό παράγοντα λ. Φυσικά δεν μπορούμε να υποθέσουμε οτι η παλμοσειρά m έχει συνεχώς σταθερή συχνότητα.

Ο παράγοντας λ εξαρτάται απο την ζητούμενη σχέση μετάδοσης n και την ακρίβεια του κάθε αισθητήρα, η οποία μετράται σε π.α.π. (παλμούς ανα περιστροφή – ppr). Επίσης, δεν μπορούμε να υποθέσουμε οτι οι δύο αισθητήρες έχουν την ίδια ακρίβεια.Παράλληλα, θέλουμε ο έλεγχος που θα εφαρμόσουμε να συντονίζει την σχετική *θέση* των δύο αξόνων και όχι μόνο την *ταχύτητα*. Αυτό σημαίνει οτι αν για κάποιον λόγο ο αξονας s εμφανίσει κάποια καθυστέρηση στον συγχρονισμό του με τον άξονα m, η γωνιακή μετατόπιση την οποία πλέον υστερείται θα πρέπει να ληφθεί υπ'όψιν κατα τον έλεγχο. Ετσι τελικά να συγχρονιστεί ως προς την γωνιακή μετατόπιση και την ταχύτητα με τον άξονα m.

Ο ελεγκτής PID χρειάζεται μία μεταβλητή σφάλματος την οποία θα επεξεργάζεται και ο μηδενισμός της θα συνεπάγεται την σωστή σχετική θέση των δύο αξόνων. Ας μελετήσουμε τρόπους κατασκευής αυτής της μεταβλητής.

Μέτρηση περιόδου:

Μπορούμε να χρησιμοποιήσουμε χρονιστές οι οποίοι θα χρονομετρούν μονίμως την διάρκεια ανάμεσα σε δύο παμούς του κάθε αισθητήρα.Ετσι κατασκευάζουμε δύο σήματα που περιέχουν την περίοδο της κάθε παλμοσειράς, έστω τα mPeriod(n) και sPeriod(n). Φυσικά τα σήματα αυτά είναι διακριτά αφου διαμορφώνονται σε κάποια ακμή των παλμών. Μπορούμε να τα μετατρέψουμε σε συνεχή, κρατώντας στσθερή την τιμή τους ανάμεσα στα δείγματα (συγκράτηση μηδενικής τάξης). Ο μικροελεγκτής μπορεί να πολλαπλασιάζει την τρέχουσα τιμή της mPeriod (t) επι τον δοσμένο λόγο λ, ωστε να προκύψει η επιθυμητή τιμή της sPeriod για την εκάστοτε χρονική στιγμή. Η τιμή αυτή μπορεί να συγκρίνεται με την πραγματική τιμή της sPeriod και έτσι να πρκύπτει το σφάλμα περιόδου.

Μέτρηση συχνότητας:

Ενας χρονιστής-μετρητής μπορεί να ρυθμιστεί ώστε να μετράει μέχρι κάποια συγκεκριμένη τιμή, να μηδενίζεται και να επαναλαμβάνει συνεχώς αυτόν τον κύκλο. Φυσικά, επειδή ο χρονιστής πυροδοτείται απο το ρολόι του συστήματος, ο κάθε κύκλος διαρκέι συγκεκριμένο χρόνο. Όσο ο μετρητής αυξάνει, ο αριθμός των παλμών των παλμοσειρών m και s αθροίζονται σε αντίστοιχες μεταβλητές. Καθε φορά που ο μετρητής μηδενίζεται, τα αθροίσματα που έχουν συσσωρευτεί αποθηκεύονται στις μεταβλητές mFreq και sFreq αντίστοιχα, ενώ οι θέσεις μνήμης στις οποίες συσσωρεύονταν μηδενίζονται. Αυτη η διαδικασία επαναλαμβάνεται παράλληλα με την λειτουργία του μετρητή.Ετσι οι μεταβλητές mFreq και sFreq περιέχουν μία τιμή ανάλογη της συχνότητας του κάθε σήματος ανα πάσα στιγμή. Αντίστοιχα με πρίν , αν ο μικροελεγκτής διαιρεί την εκάστοτε τιμή της mFreq με τον λόγο λ, προκύπτει η επιθυμητή τιμή για την sFreq. Η διαφορά της επιθυμητής απο την πραγματική τιμή μας δίνουν το σφάλμα συχνότητας.

Οι δύο προηγούμενες μεθοδοι κάνουν χρήση κάποιου χρονιστή-μετρητή. Στον μικροελεγκτή υπάρχουν τρείς χρονιστές υλοποιημένοι με υλικό (hardware), η λειτουργία των οποίων δεν επηρεάζει καθόλου την εκτέλεση του κώδικα. Ωστόσο, η ακρίβεια των μετρήσεων επηρεάζεται έντονα απο την ακρίβεια των χρονιστών, την συχνότητά τους, την ακρίβεια των αισθητήρων και την ταχύτητα περιστροφής των αξόνων. Επίσης, μπορεί να προκύψουν φαινόμενα υπερχείλισης στις περιπτώσεις που η περίοδος είναι πολύ μεγάλη για να μετρηθεί και η συχνότητα πολύ μικρή. Παράλληλα, ο μικροελεγκτής πθανότατα καλείται να κάνει διαιρέσεις σε πραγματικό χρόνο (αν λ δεκαδικός), μία πράξη ιδιαιτερα χρονοβόρα για τον μικροελεγκτή. Σημειώνω οτι ο μικροελεγκτής δεν διαθέτει μονάδα εκτέλεσης πράξεων σε αριθμούς κινητής υποδιαστολής. Επίσης, ενώ η μέτρηση συχνότητας είναι ανάλογη της γωνιακής ταχύτητας του κινητήρα, το σφάλμα περιόδου είναι σχετικά δύσχρηστο για το σύστημα ελέγχου, αφού δεν σχετίζεται αυτούσιο με κάποιο φυσικό μέγεθος της κίνησης του κινητήρα.

Μέτρηση θέσης:

Αν σκεφτούμε την δομή των οπτικών αισθητήρων, βλέπουμε οτι ο κάθε παλμός στην παλμοσειρά εξόδου τους αντιστοιχεί σε κάποια συγκεκριμένη γωνιακή μετατόπιση του άξονα.Ετσι, μπορούμε να αθροίζουμε το πλήθος των παλμών της κάθε παλμοσειράς σε αντίστοιχες μεταβλητές, έστω mPos και sPos. Οι τιμές των μεταβλητών αυτών δείχνουν την τρέχουσα γωνιακή θέση του κάθε άξονα. Πολλαπλασιάζοντας την εκάστοτε τιμή της mPos επι λ και αφαιρώντας την αντίστοιχη τιμή της sPos μας δίνει το σφάλμα θέσης.

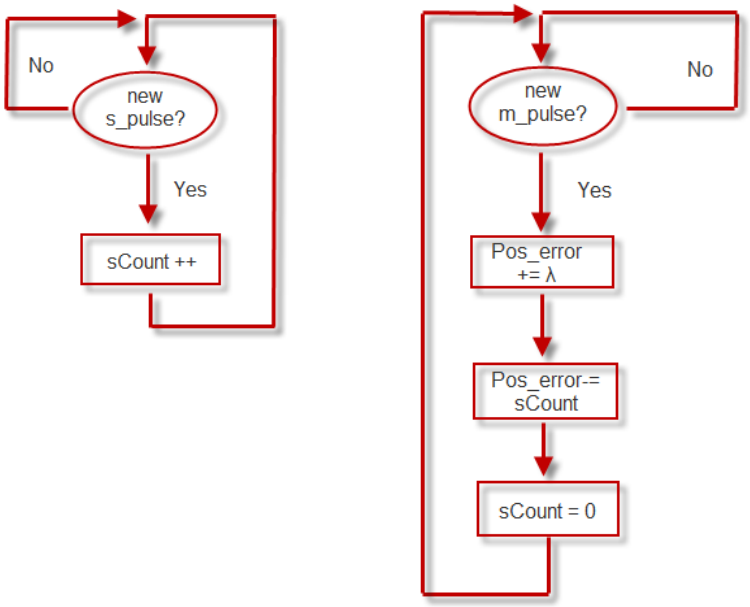
Αυτή η μέθοδος έχει το προφανές πρόβλημα οτι οι μεταβλητές mPos, sPos θα αυξάνουν συνεχώς, μάλλον μέχρι απαγορευτικές για τον μικροελεγκτή τιμές. Ωστόσο, αποφεύγεται η χρήση των χρονιστών. Παράλληλα, η εύρεση του σφάλματος θέσης είναι ιδιαίτερα χρήσιμη για το σύστημα ελέγχου. Το ενδεχόμενο πρόβλημα των χρονοβόρων πράξεων σε πραγματικό χρόνο παραμένει και σε αυτήν την μέθοδο.

Σε αναζήτηση μιας καλύτερης μεθόδου ας σκεφτούμε το εξής: Εφόσον το ζητούμενο είναι να συγχρονίσουμε τους δύο άξονες μεταξύ τους, ίσως δεν χρειάζεται ο υπολογισμός των απόλυτων μεγεθών της κίνησής τους, παρά μόνο των σχετικών. Εφ'όσον το σφάλμα θέσης είναι ιδιαίτερα εύχρηστο για το σύστημα ελέγχου, ας ασχοληθούμε με την εύρεση της σχετικής θέσης, ή σχετικής μετατόπισης.

Αφού η σχέση μετάδοσης μεταξύ των αξόνων είναι n, αυτό σημαίνει οτι κάθε φορά που ο άξονας master μετατοπίζεται κατα 1 μονάδα, ο άξονας slave θα έπρεπε να μετατοπίζεται κατα n μονάδες. Όσον αφορά στις παλμοσειρές των αισθητήρων, μέσα στην διάρκεια ενός παλμού της m(t), θα πρέπει να αντιστοιχούν λ παλμοί της s(t). Θυμίζω οτι ο

λόγος λ περιλαμβάνει την επιθυμητή σχέση μετάδοσης n και την διόρθωση λόγω ενδεχόμενης διαφοράς στην ανάλυση των αισθητήρων.

Ετσι, αν σε κάθε παλμό της m(t) αυξάνουμε μία μεταβλητή κατα λ, ενώ αφαιρούμε το πλήθος των παλμών της s(t) που κατέφθασαν στην διάρκεια αυτού του παλμού της m(t), η μεταβλητή αυτή περιέχει ανα πάσα στιγμή το σφάλμα θέσης εκφρασμένο σε παλμούς.Ας δούμε την διαδικασία αυτή σε μορφή διαγραμμάτων ροής. Τα δύο διαγράμματα ροής μπορεί να θεωρηθούν οτι εκτελούνται παράλληλα απο τον μικροελεγκτή. Αυτό επιτυγχάνεται κατα κάποιον τρόπο με την υλοποίηση μέσω ρουτίνων διακοπών. Κάθε νέος παλμός που παράγεται απο τους αισθητήρες αναγνωρίζεται στην θετική ή αρνητική ακμή του και πυροδοτεί την εκτέλεση κάποιου τμήματος κώδικα, ανεξάρτητα απο το σημείο που βρισκόταν η εκτέλεση προηγουμένως. Επίσης, η μεταβλητή sCount είναι η ίδια και στα δύο διαγράμματα:

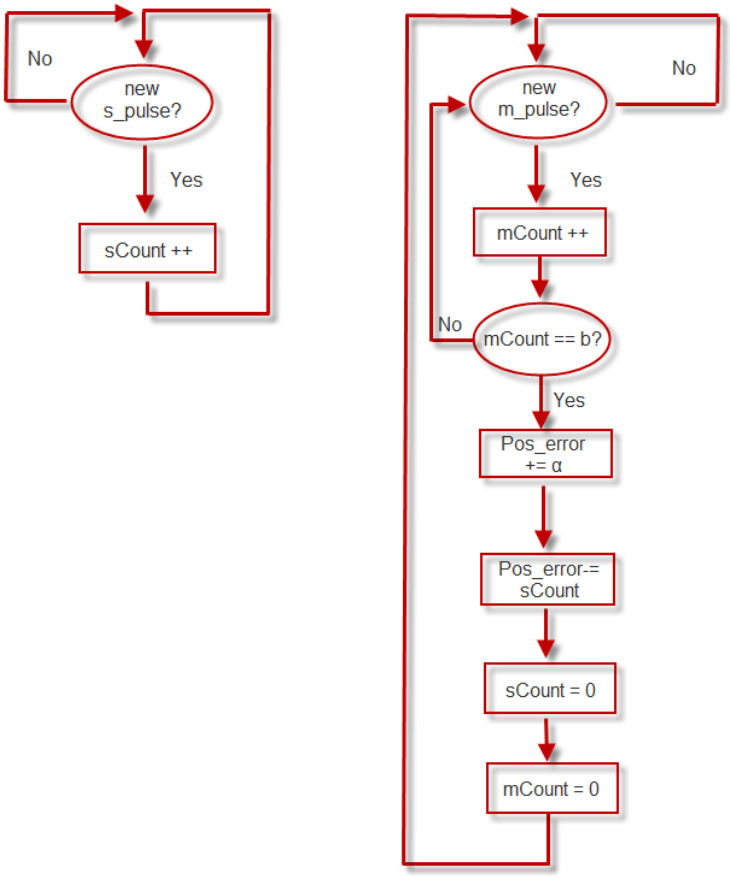


Σχήμα 2.1

Η μέθοδος αυτή υπολογίζει το σφάλμα θέσης του άξονα s κάνοντας μόνο προσθέσεις και αφαιρέσεις σε πραγματικό χρόνο. Το γεγονός αυτό την καθιστά ιδιαίτερα αποδοτική. Ωστόσο, υπάρχει ένα σοβαρό πρόβλημα το οποίο έχουμε παραβλέψει. Ο παράγοντας λ δεν μπορεί να θεωρηθεί ακέραιος στην γενική περίπτωση. Εκτός απο το γεγονός οτι ο χειρισμός αριθμών κινητής υποδιαστολής είναι ιδιαίτερα δυσχερής για τον μικροελεγκτή, η χρήση δεκαδικού αριθμου για το λ θα εμφάνιζε προβλήματα, καθώς η μέτρηση παλμών μπορεί να έχει μόνο ακέραιο αποτέλεσμα. Στην συνέχεια θα γίνει μελέτη για την αντιμετώπιση αυτού του προβλήματος.

Κατ'αρχήν μπορούμε να εκφράσουμε τον λόγο λ ως κλάσμα με ακέραιους όρους: $\frac{a}{b}$. Η υπόθεση οτι ο λ είναι ρητός αριθμός είναι εύλογη.

Ο παραπάνω αλγόριθμος μπορεί να προσαρμοστεί στα νέα δεδομένα με την σύμβαση η μεταβλητή σφάλματος θέσης να ανανεώνεται όχι σε κάθε παλμό της m(t) , αλλα σε κάθε b παλμούς. Η επιθυμητή μετατόπιση του s άξονα για κάθε b παλμούς του m άξονα είναι τώρα α. Πλέον ολοι οι αριθμοί είναι ακέραιοι, και τα διαγράμματα ροής έχουν διαμορφωθεί ώς εξής:



Σχήμα 2.2

Ο αλγόριθμος αυτός είναι ιδιαίτερα αποδοτικός και εύχρηστος. Ωστόσο, έχει ένα μειονέκτημα: Η μεταβλητή σφάλματος θέσης πλέον δεν ανανεώνεται σε κάθε παλμό της m(t), αλλα σε κάθε b παλμούς. Προφανώς η ταχύτητα του ελέγχου μειώνεται b φορές, αφού το b επηρεάζει τον ρυθμό δειγματοληψίας του σφάλματος θέσης. Για τιμές του λ με κάποια δεκαδικά ψηφία ο παρονομαστής αυτός ενδεχομένως να είναι αρκετά μεγάλος. Ετσι, θα πρέπει να έχουμε πολύ μεγάλη ανάλυση στον αισθητήρα του άξονα m ωστε να επιτύχουμε σχετικά γρήγορο έλεγχο. Επίσης, ο αριθμός b αλλάζει ανάλογα με τον επιθυμητό λόγο λ που θέλουμε να επιβάλλουμε στους άξονες. Το γεγονός αυτό καθιστά την συμπεριφορά του συστήματος αρκετά απρόβλεπτη για διάφορες επιλογές της τιμής του λ απο τον χρήστη.

Ας επιστρέψουμε στην θεώρηση του αριθμού λ ως δεκαδικού. Ας τον εκφράζουμε σε μορφή ακεραίου και δεκαδικού μέρους:

$$\lambda = i+d \text{ , με } i \in \mathbb{Z} \text{ και } d \in \mathbb{R} : 0 \le d < 1.$$

Ας θεωρήσουμε οτι αθροίζουμε n διαδοχικές φορές τον αριθμό λ και το άθροισμα το ονομάζουμε L, με αντίστοιχο ακέραιο και δεκαδικό μέρος:

$$L = I + D,\ I \in \mathbb{Z} \text{ και } D \in \mathbb{R} : 0 \le D < 1.$$

Εχουμε, επομένως τις ακολουθίες L(n) = I(n)+D(n), όπου οι ακολουθίες I(n) και D(n) εκφράζουν το ακέραιο και δεκαδικό μέρος αντίστοιχα του αριθμού L(n) για n αθροίσεις του αριθμού λ. Προφανώς ισχύει:

$$L(0)=I(0) = D(0) = 0.$$

Ας εξετάσουμε αυτές τις ακολουθίες.
Εχουμε τους εξής αναδρομικούς τύπους:

$$L(n) = L(n-1)+\lambda$$

$$I(n)= \begin{cases} I(n-1)+ i, & \text{αν } D(n-1)+d <1 \text{ ή} \\ I(n-1)+i+1, & \text{αν } D(n-1)+d \geq 1 \end{cases}$$

$$D(n)= \begin{cases} D(n-1)+d, & \text{αν } D(n-1)+d<1 \text{ ή} \\ D(n-1)+d-1, & \text{αν } D(n-1)+d\geq 1 \end{cases}$$

Δηλαδή, το ακέραιο μέρος του αθροίσματος αυξάνεται κατα i για κάθε άθροιση, εκτός απο τις περιπτώσεις οπου το δεκαδικό μέρος υπερβαίνει την τιμή 1. Τότε γίνεται διόρθωση μεταφέροντας μία μονάδα απο το δεκαδικό στο ακέραιο μέρος, ωστε το δεκαδικό μέρος να παραμένει μικρότερο απο 1.

Ας ορίσουμε την ακολουθία B(n) = I(n) – I(n-1) με B(0)=0, ώς το *διακριτό διαφορικό* της I(n). Εχουμε:

$$B(n) = \begin{cases} i, & \text{αν } D(n-1)+d <1 \text{ ή} \\ i+1, & \text{αν } D(n-1)+d \geq 1 \end{cases}$$

Η ακολουθία B(n) αποτελείται απο τις τιμές i και i+1 σε κάποια αλληλουχία, η οποία παρουσιάζει κάποια περιοδικότητα.

*Απόδειξη περιοδικότητας:

Για να αποδείξουμε την περιοδικότητα της ακολουθίας B(n) επιστρέφουμε για λίγο στην θεώρηση του αριθμού λ ώς κλάσμα ακεραίων: λ=$\frac{\alpha}{b}$. Εστω οτι αθροίζουμε διαδοχικά τον αριθμό λ και διαμορφώνονται αντίστοιχα οι τιμές των ακολουθιών I(n) , D(n).
Εστω, επίσης οτι για κάποιο $k \in \mathbb{Z}$ ισχύει D(k)=ξ.

Αυτό σημαίνει οτι ο αριθμός $L(k) = k * \frac{a}{b}$ έχει δεκαδικό μέρος ίσο με ξ.

Ας θεωρήσουμε τώρα οτι αθροίζουμε τον αριθμό λ (k+b) φορές. Το άθροισμα είναι:

$$L(k+b) = (k+b) * \frac{a}{b} = k * \frac{a}{b} + b * \frac{a}{b} = a + k * \frac{a}{b}$$

Ο αριθμός α όμως είναι ακέραιος εξ'ορισμού. Αρα, το αθροισμα αυτό έχει επίσης δεκαδικό μέρος ίσο με ξ. Επομένως αποδείξαμε οτι

$$\forall k, D(k) = D(k+b).$$

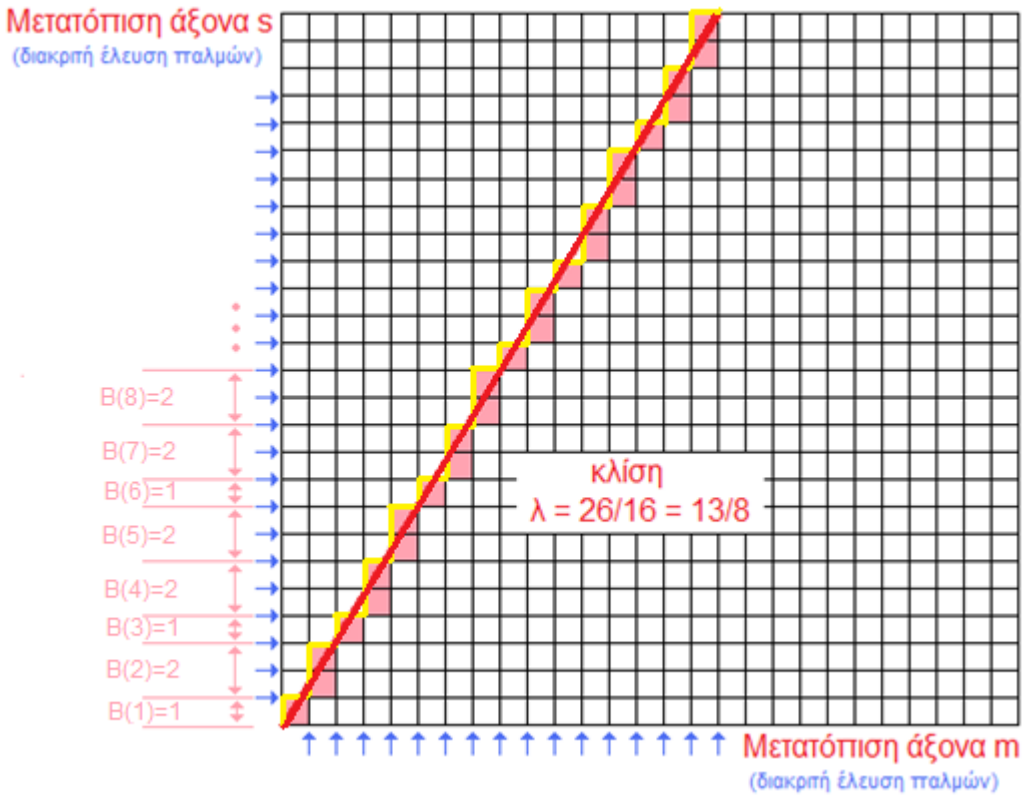
Αρα η ακολουθία D(n) είναι περιοδική με περίοδο b. Αφού η ακολουθία B(n) εξαρτάται μόνο απο τις τιμές της ακολουθίας D(n) (το i και το d είναι σταθερά), τότε και η B(n) είναι περιοδική με την ίδια περίοδο.
*τέλος απόδειξης.

Η περίοδος b (ο παρονομαστής του λόγου λ) είναι και η θεμελιώδης περίοδος της ακολουθίας στην περίπτωση που το κλάσμα $\frac{a}{b}$ δεν απλοποιείται. Δηλαδή, όταν οι αριθμοί α και b είναι πρώτοι μεταξύ τους. Γενικά η θεμελιώδης περίοδος είναι b/ΜΚΔ(α,b), όπου ΜΔΚ() ο Μέγιστος Κοινός Διαιρέτης

Ας μελετήσουμε τώρα την φυσική σημασία της ακολουθίας B(n).Η ακολουθία αυτή μας δείχνει την ακέραια μεταβολή του αριθμού L(n) για κάθε διαδοχική άθροιση του λόγου λ. Αν επιστρέψουμε τον συλλογισμό μας στην θεώρηση των παλμοσειρών, θα δούμε οτι αν δύο παλμοσειρές (m(t), s(t)) έχουν λόγο συχνοτήτων λ, αυτό σημαίνει οτι στην διάρκεια ενός παλμού της m(t) το ακέραιο πλήθος των παλμών που λαμβάνουμε απο την s(t) είναι οι

όροι της ακολουθίας B(n). Φυσικά, αν ο λόγος λ είναι ακέραιος, η περιοδικότητα της ακολουθίας είναι 1, (παρονομαστής b=1) και η ακολουθία B(n) αποτελείται απο επανάληψη της τιμής λ.

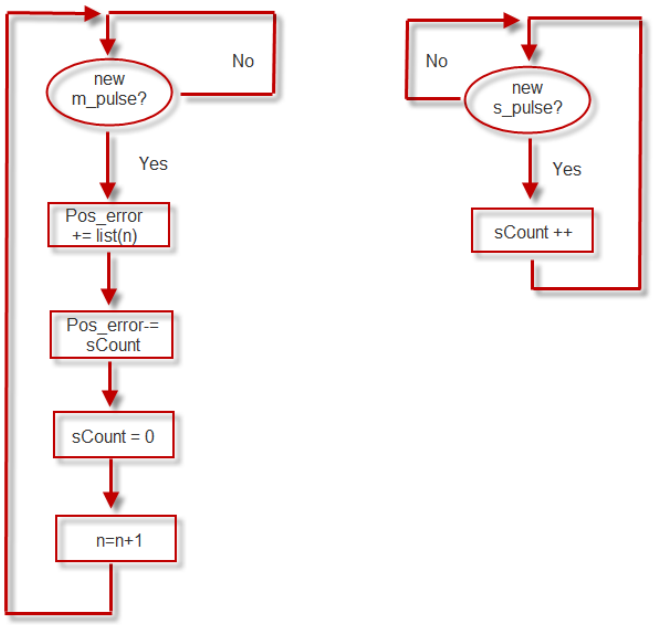
Θυμίζω οτι ο κάθε παλμός που παράγεται απο τους αισθητήρες αντιστοιχεί σε συγκεκριμένη γωνιακή μετατόπιση του αντίστοιχου άξονα. Ομοίως, ο λόγος λ αναλογεί στην σχέση μεταξύ των μετατοπίσεων και ταχυτήτων των δύο αξόνων (είναι ανάλογος του λόγου μετάδοσης n).Ας δούμε ένα σχετικό σχήμα:



Σχήμα 2.3

Στο σχήμα φαίνεται η ευθεία με κλίση λ (κοκκινο) που αναφέρεται στην σχετική θέση των δύο αξόνων. Το τετραγωνικό πλέγμα αντιπροσωπεύει τους μετρήσιμους παλμούς, αφού είναι η ελάχιστη μετατόπιση που μπορούμε να μετρήσουμε.Με κίτρινο χρώμα εμφανίζεται η ακολουθία I(n), όπου n οι διακριτοί παλμοί του άξονα m. Τέλος, με ροζ χρώμα έχει γραφεί η ακολουθία B(n) που αντιστοιχεί στην διαφορά των διαδοχικών τιμών της I(n). Η ακολουθία έχει περιοδο $n_0 = 8$, αφού ο παρονομαστής του λ είναι 8 και στο παράδειγμά μας έχει τιμές: 1,2,1,2,2,1,2,2,...

Πλέον με την χρήση της ακολουθίας B(n) μπορούμε να χρησιμοποιήσουμε την απλούστερη μέθοδο μέτρησης του σφάλματος θέσης, που παρουσιάσαμε προηγουμένως, όταν υποθέσαμε οτι λ ακέραιος. Αρκεί απλώς να κατασκευάσουμε μία λίστα αριθμών που να περιέχει τις τιμές της ακολουθίας B(n). Η κάθε τιμή της ακολουθίας χρησιμοποιείται στην θέση του λόγου λ διαδοχικά.Ας δούμε το τροποποιημένο διάγραμμα ροής:



Σχήμα 2.4

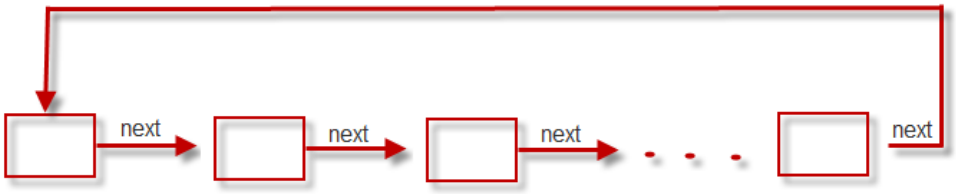
Η λίστα αυτή εξαρτάται μόνο απο την τιμή του λόγου λ, αρα μπορεί να κατασκευαστεί πριν την εκτέλεση του προγράμματος χωρίς να επιβαρρύνει τον μικροελεγκτή σε πραγματικό χρόνο. Ας δούμε ένα παράδειγμα ψευδοκώδικα για την κατασκευή των τιμών της:

Εστω δοσμένοι οι όροι του κλάσματος λ = $\frac{a}{b}$

```
n =1;
I(0)=0;
do {
I(n) = (n*a) div b;     //  = trunc((float)((n*a)/b))
B(n) = I(n)-I(n-1);
n++;
}
while( n*a  mod b != 0)
```

Η εκτέλεση θα σταματήσει στην τιμή n=b,αν (α,b) πρώτοι μεταξύ τους. Αν και η εκτέλεση του αλγορίθμοι γίνεται ανεξάρτητα απο το πρόγραμμα ελέγχου,ωστόσο υπάρχει πιο αποδοτικός αλγόριθμος ο οποίος περιλαμβάνει μόνο προσθέσεις και αφαιρέσεις, χωρίς διαίρεση. Περισσότερα στο τμήμα όπου αναλύεται ο κώδικας.

Επίσης, έχουμε αποδείξει οτι η λίστα είναι περιοδική. Επομένως, αρκεί να κατασκευάσουμε μόνο τις τιμές μίας περιόδου. Η λίστα που θα δημιουργήσουμε μπορεί να είναι κυκλικά συνδεδεμένη ωστε να εξασφαλίσουμε την ανακύκλωση των τιμών χωρίς επι πλέον υπολογισμούς.



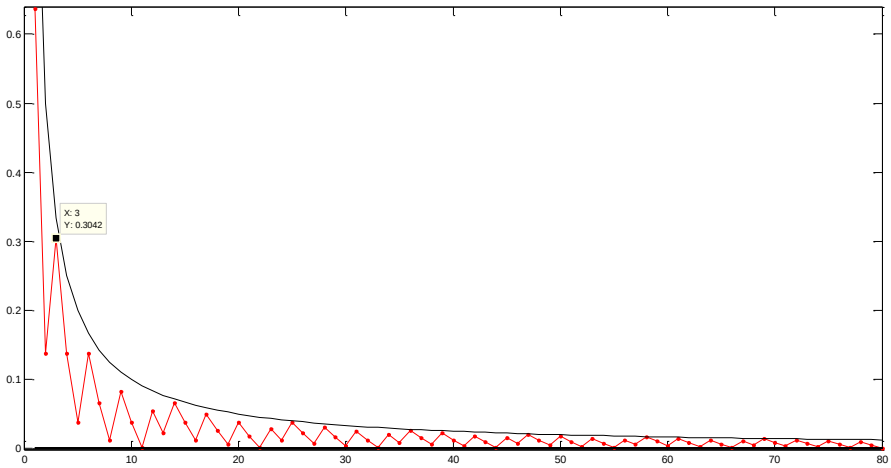
Σχήμα 2.5 Κυκλικά συνδεδεμένη λίστα μεταβλητών

Συνοψίζοντας, έχουμε βρεί μία μέθοδο υπολογισμού του σφάλματος θέσης ιδιαίτερα αποδοτική και γρήγορη, ενώ με την χρήση της ακολουθίας B(n) αποφεύγονται σφάλματα στρογγυλοποίησης.

2.2 Κλασματικές προσεγγίσεις

Το μειονέκτημα αυτής της μεθόδου είναι οτι η λίστα πρέπει να καταλαμβάνει θέση στην μνήμη και το μήκος της στην χειρότερη περίπτωση είναι b. Επομένως για τιμές του λόγου λ με πολλά δεκαδικά ψηφία ενδεχομένως το μήκος να είναι απαγορευτικό για την περιορισμένη μνήμη του μΕ. Πρέπει, συνεπώς, σε αυτές τις περιπτώσεις να γίνεται μία προσέγγιση της λίστας απο μία άλλη με λιγότερα στοιχεία.Εφ'όσον το μήκος της λίστας ισούται με τον παρονομαστή του λόγου λ, το πρόβλημα ανάγεται στην προσέγγιση του κλάσματος $\frac{\alpha}{b}$ απο ένα άλλο κλάσμα $\frac{\gamma}{\delta}$, με δ < d_{max}, όπου d_{max} ένα ανώτατο όριο που έχουμε θέσει για τον παρονομαστή.

Αν θυμιθούμε το Σχήμα 2.3 θα δούμε οτι η ακολουθία I(n) αποτελεί μία ακέραιη προσέγγιση της ευθείας λ*n . Δηλαδή ο αριθμός $\frac{\mathrm{I}(n)}{n}$ αποτελεί μία προσέγγιση του $\frac{a}{b}$. Μαλιστα, αφού το δεκαδικό μέρος D(n) είναι πάντα μικρότερο της μονάδας, προκύπτει οτι η προσέγγιση θα έχει πάντα σφάλμα μικρότερο απο $\frac{1}{n}$. Μία πρώτη σκέψη που κάνουμε είναι να κρατάμε πάντα τον μέγιστο επιτρεπτό πλήθος στοιχείων της λίστας (δηλαδή παρονομαστή), πετυχαίνοντας σφάλμα μικρότερο απο $\frac{1}{d_{\max}}$. Ωστόσο, στις περισσότερες περιπτώσεις αυτό δεν είναι η καλύτερη προσέγγιση. Το γεγονός αυτό μπορούμε να το δούμε αν απεικονίσουμε το σφάλμα των διαδοχικών προσεγγίσεων για κάποια παραδείγματα:



Σχήμα 2.6

Στο παράδειγμα εμφανίζονται τα σφάλματα των διαδοχικών προσεγγίσεων του κλάσματος 131/80. Στον άξονα χ βρίσκονται οι παρονομαστές της κάθε προσέγγισης και στον y οι τιμές σφάλματος. Με μαύρο χρώμα εμφανίζεται το άνω φράγμα των σφαλμάτων,

η καμπύλη $\frac{1}{n}$. Παρατηρώντας τις διακυμάνσεις του σφάλματος βλέπουμε οτι για παράδειγμα η προσέγγιση με παρονομαστή 11 έχει σφάλμα 0,001136 , ενώ με παρονομαστή 50 το σφάλμα είναι 0,0175 (πάντα μικρότερο απο 1/50=0,02), το οποίο είναι πάνω απο 15 φορές μεγαλύτερο! Επομένως δεν αρκεί ο μεγάλος παρονομαστής και απαραίτητη είναι η διερεύνηση των προσεγγίσεων ωστε να χρησιμοποιήσουμε την καλύτερη δυνατή μέσα στα επιτρεπτά όρια.

Ωστόσο πρέπει να παρατηρήσουμε οτι οι προσεγγίσεις $\frac{I(n)}{n}$ δεν είναι οι βέλτιστες για τον κάθε παρονομαστή. Αυτό συμβαίνει επειδή ο αλγόριθμος εύρεσης της ακολουθίας I(n) βρίσκει μόνο ακεραίους μικρότερους απο την πραγματική τιμή. Θυμίζω την γραμμή του ψευδοκώδικα:

```
I(n) = (n*a) div b; // = trunc((float)((n*a)/b))
```

Καλύτερη προσέγγιση θα ήταν να ψάχνουμε τον κοντινότερο ακέραιο στην πραγματική τιμή, μεγαλύτερο ή μικρότερο. Η αντίστοιχη γραμμή κώδικα θα ήταν:

```
I2(n) = [(n*a) + b/2] div b; //= round((float)((n*a)/b))
```

Με αυτήν την μέθοδο το σφάλμα των προσεγγίσεων είναι κατα απόλυτη τιμή μικρότερο ή ίσο του $\frac{1}{2*n}$, ενώ παίρνει τόσο θετικές όσο και αρνητικές τιμές.

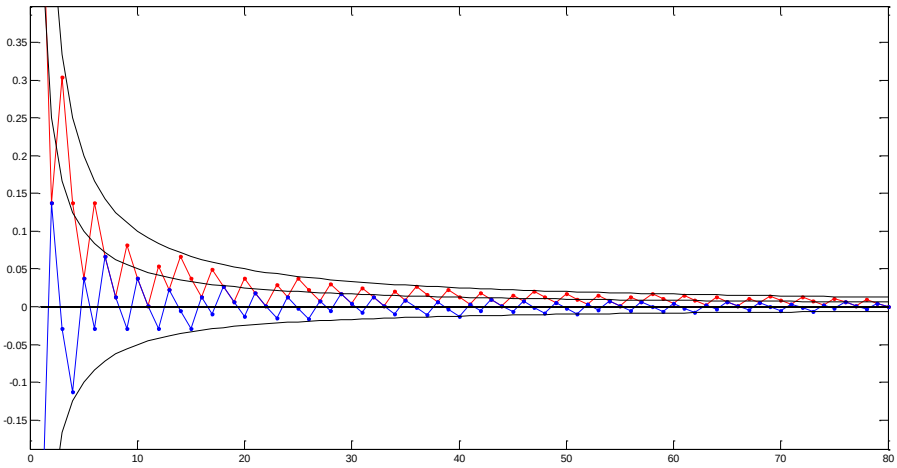
*Απόδειξη:

Η στρογγυλοποίηση ενός αριθμού round(x) = trunc(x+0.5) επιστρέφει τον ακέραιο ο οποίος απέχει την μικρότερη απόσταση απο τον x επάνω στον άξονα των πραγματικών αριθμών.Το μέσον της απόστασης δύο ακεραίων είναι 0,5. Επομένως, ο ακέραιος που επιστρέφεται απο την round(x) δεν μπορεί να διαφέρει απο τον x κατα περισσότερο απο 0,5. Ετσι, η προσέγγιση: $\frac{I2(n)}{n} = \frac{round(\frac{n*a}{b})}{n}$ απέχει απο την τιμή $\frac{n*\frac{a}{b}}{n} = \frac{a}{b}$ κατα λιγότερο απο $\frac{0.5}{n} = \frac{1}{2*n}$ ή ακριβώς τόσο.

Επομένως, αυτό είναι το μέγιστο κατ'απόλυτη τιμή σφάλμα της εν λόγω προσέγγισης.

*τέλος απόδειξης

Στο διάγραμμα που ακολουθεί εμφανίζονται τα σφάλματα και των δύο μεθόδων για το προηγούμενο παράδειγμα:



Σχήμα 2.7

Με μπλέ χρώμα εμφανίζεται η μέθοδος στρογγυλοποίησης, η οποία φράσσεται απο τις καμπύλες $\frac{1}{2*n}$ και $\frac{-1}{2*n}$ και με κόκκινο η μέθοδος αποκοπής . Βλέπουμε οτι οι προσεγγίσεις με *θετικό* σφάλμα μικρότερο του $\frac{1}{2*n}$ είναι κοινές, ενώ στις υπόλοιπες η μέθοδος στρογγυλοποίησης είναι σαφώς καλύτερη. Μάλιστα, είναι η *βέλτιστη* μέθοδος για κλασματική προσέγγιση με συγκεκριμένο παρονομαστή.

*Απόδειξη:

Εστω η προσέγγιση $\frac{\gamma}{n}$ και $e=\frac{a}{b}-\frac{\gamma}{n}$ το σφάλμα της προσέγγισης. Ισχύει οτι

$$|e|\le\frac{1}{2*n}\Leftrightarrow\frac{-1}{2*n}\le e\le\frac{1}{2*n} \quad (1)$$

Επίσης, ισχύει οτι οποιαδήποτε προσέγγιση πρέπει να έχει ακέραιο αριθμητή. Εστω μία άλλη προσέγγιση με αριθμητή $k\neq\gamma$ και τον ίδιο παρονομαστή. Αφού $k\neq\gamma$ τότε η προσέγγιση αυτή θα διαφέρει απο την προηγούμενη κατα τουλάχιστον $\frac{1}{n}$. Δηλαδή:

$$\left|\frac{k}{n}-\frac{\gamma}{n}\right|\ge\frac{1}{n}$$

Αρα για το καινούριο σφάλμα e1 θα ισχύει:

$$e1=e+\frac{1}{n} \text{ ή } e1=e-\frac{1}{n}.$$

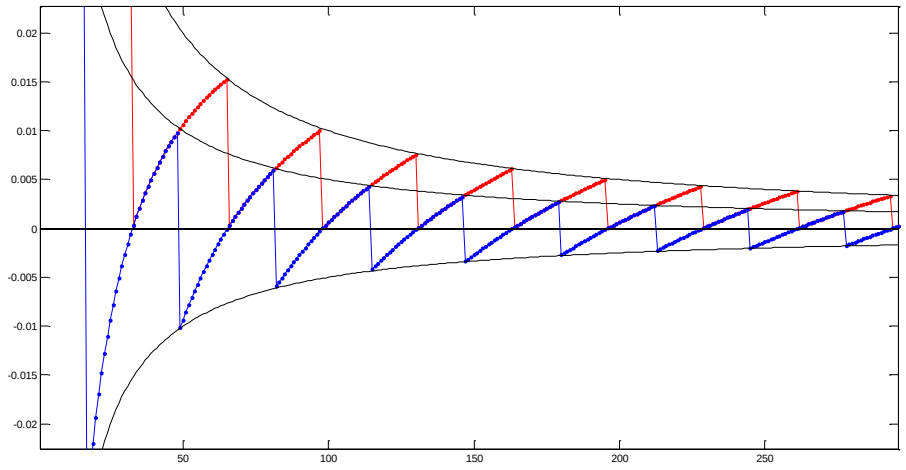
Και οι δύο περιπτώσεις με την χρήση της ανισότητας (1) οδηγούν στις ανισότητες :

$$e\ge\frac{1}{2*n} \text{ και } e\le-\frac{1}{2*n} \text{ αντίστοιχα.}$$

Επομένως, η προσέγγιση $\frac{k}{n}$ έχει μεγαλύτερο σφάλμα απο την αρχική $\frac{\gamma}{n}$ ή ίσο, και άρα η αρχική προσέγγιση είναι βέλτιστη. Η περίπτωση που η καινούρια προσέγγιση παρουσιάζει ακριβώς το ίδιο σφάλμα με την αρχική, είναι όταν ισχύει η ισότητα: $e=\frac{1}{2*n}$ ή $e=-\frac{1}{2*n}$

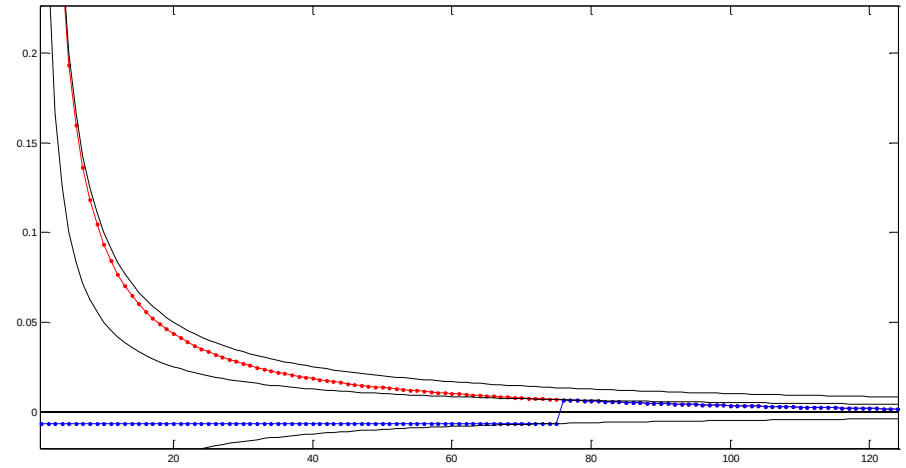
*τέλος απόδειξης

Ας δούμε και μερικά άλλα παραδείγματα:



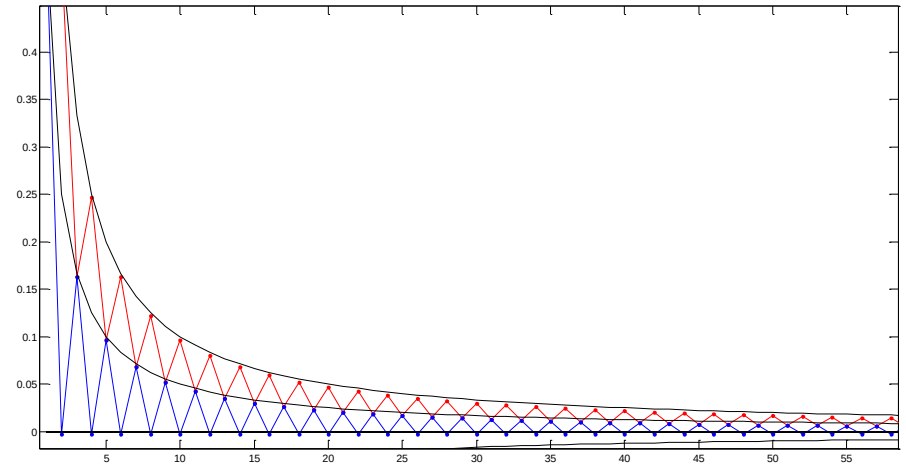
Σχήμα 2.8

Επάνω:σφάλματα προσεγγίσεων του λόγου 17/555



Σχήμα 2.9

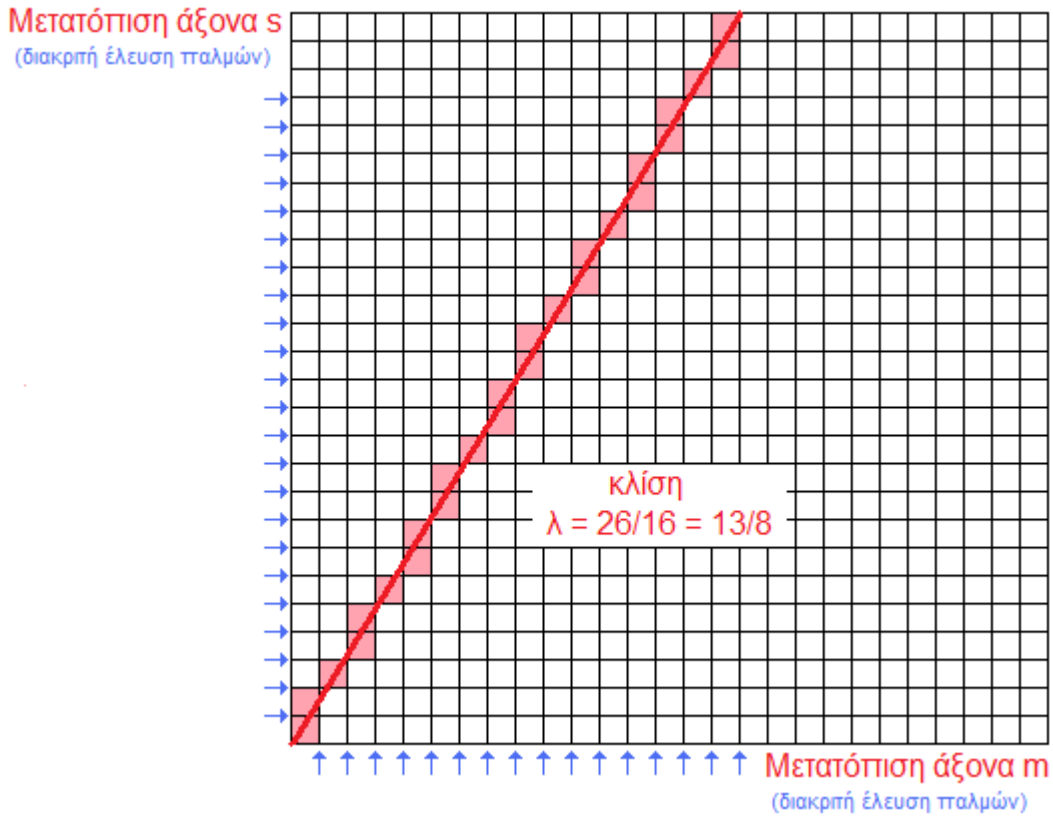
Επάνω: σφάλματα προσεγγίσεων του λόγου 150/151 (εμφανής αδυναμία μεθόδου αποκοπής)



Σχήμα 2.10

επάνω: σφάλματα προσεγγίσεων του λόγου 75/151 (επίσης φαίνεται η υπεροχή της μεθόδου στρογγυλοποίησης)

Ενδεικτικά ας δούμε και το σχήμα της προσέγγισης με την μέθοδο στρογγυλοποίησης στο προηγούμενο παράδειγμα:



Σχήμα 2.11

Το σφάλμα της κάθε προσέγγισης μπορεί να υπολογίζεται απο τον αλγόριθμο των δύο μεθόδων με την προσθήκη της ακόλουθης γραμμής ψευδοκώδικα:

```
error(n)= (n*a)mod b  ή

error(n)=[(n*a)+b/2] mod b
```

Ετσι κραταμε μία μεταβλητή ανάλογη του σφάλματος για κάθε προσέγγιση και στην συνέχεια μπορούμε να επιλέξουμε την χαμηλότερη. Η πραγματική τιμή σφάλματος της κάθε προσέγγισης είναι $\frac{error(n)}{n*b}$. Σε επόμενη ενότητα αναφέρεται ακριβώς ο αλγόριθμος που χρησιμοποιήθηκε.

Σε αυτό το σημείο αναφέρω οτι παρόμοια λειτουργία επιτελεί ο σχεδιαστικός αλγόριθμος του Bresenham, που χρησιμοποιείται για την σχεδίαση γραμμών με την χρήση ψηφίδων σε πίνακες. Σημαντικό παράδειγμα χρήσης είναι η σχεδίαση ευθύγραμμων τμημάτων σε οθόνες με ψηφίδες ή σε χαρτί απο καρτεσιανούς εκτυπωτές. Περισσότερα βρίσκονται σε ανάλογο παράρτημα

Τελος , συνοψίζοντας, ας δούμε επιγραμματικά την διαδικασία που θα ακολουθήσουμε για τον υπολογισμό του σφάλματος θέσης:

Για δοθέντα λόγο λ = $\frac{a}{b}$ αρχικά γίνεται μία προσέγγιση, αν χρειάζεται, με την χρήση ενός κλάσματος με μικρότερο παρονομαστή που προκύπτει απο την μέθοδο στρογγυλοποίησης. Το όριο d_{max} που θέτουμε για τον παρονομαστή εξαρτάται απο την ελεύθερη μνήμη που έχουμε στην διάθεσή μας. Το σφάλμα είναι πάντα μικρότερο απο $\frac{1}{2*d_{\max}}$, ενώ συνήθως παίρνει πολύ μικρότερες τιμές. Στην συνέχεια εκτελείται ο αλγόριθμος κατασκευής των τιμών μιάς περιόδου της ακολουθίας B(n) με την μέθοδο αποκοπής. Οι τιμές αυτές αποθηκεύονται σε μία απλά συνδεδεμένη κυκλική λίστα. Ο αλγόριθμος κατασκευής του σφάλματος θέσης ανανεώνει την μέτρηση σε κάθε παλμό του σήματος m(t), σύμφωνα με την διαδικασία που έχουμε περιγράψει.

2.3 Στοιχεία συστήματος

Εχουμε περιγράψει θεωρητικά τα στοιχεία που είναι απαραίτητα για την υλοποίηση του συστήματός μας. Ας δούμε τώρα ποια φυσικά στοιχεία χρησιμοποιήθηκαν:

Κινητήρας:
Κινητήρας ΣΡ με προσαρμοσμένο μειωτήρα μείωσης 1:26. Η ονομαστική τάση λειτουργίας του κινητήρα είναι τα 22 Volts και η ταχύτητα περιστροφής υπο ονομαστική τάση και χωρίς φορτίο μετά την μείωση είναι 110 Σ.Α.Λ.

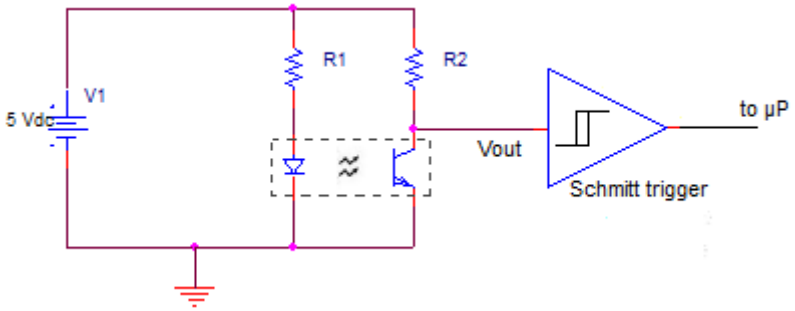
Σχήμα 2.12

Αισθητήρες :
Ως αισθητήρας του κινητήρα χρησιμοποιήθηκε ένας πλαστικός δίσκος με 30 εγκοπές ο οποίος τοποθετήθηκε στον άξονα του κινητήρα πριν την μείωση. Η έξοδος του κινητήρα παράγεται απο ένα ζεύγος λαμπτήρα LED και φωτοτρανζίστορ, με κωδικό H21A1 (παρατίθεται τεχνικά χαρακτηριστικά). Πρόκειται για έναν απλό οπτικό κωδικοποιητή μονού καναλιού.

Σχήμα 2.13

Στον άξονα αναφοράς (του τσώκ, δηλαδή) τοποθετήθηκαν δύο οπτικοί κωδικοποιητές. Ο Ενας αποτελείται απο έναν μεταλλικό δίσκο με 40 εγκοπές, ο οποίος παρεμβάλλεται στην οπτική επαφή του οπτικού ζεύγους LED-φωτοτρανζίστορ SHARP GP1S73P (παρατίθενται τεχνικές πληροφορίες). Και σε αυτήν την περίπτωση έχουμε απλό οπτικό κωδικοποιητή μονού καναλιού. Επιπλέον, χρησιμοποιήθηκε και ένας ανακλαστικός οπτικός αισθητήρας ο οποίος αλληλεπιδρά με την επιφάνεια του μεταλλικού δίσκου.Η μισή κυκλική επιφάνεια έχει καλυφθεί με ανακλαστικό στρώμα ενώ η άλλη μισή είναι μαύρη. Σαν αποτέλεσμα, ο αιθητήρας αυτός παράγει έναν παλμό ανα περιστροφή του άξονα. Η ύπαρξη αυτού του βοηθητικού αισθητήρα είναι ιδιαίτερα χρήσιμη αφού η πληροφορία που μας παρέχει είναι απόλυτη. Δηλαδή, η έναυση του παλμού συμβαίνει πάντα σε συγκεκριμένη γωνιακή θέση του άξονα.

Τα κυκλώματα οδήγησης των αισθητήρων είναι πανομοιότυπα, της εξής μορφής:



Σχήμα 2.14

Η αντίσταση R1 περιορίζει το ρεύμα που διαρρέει τον λαμπτήρα LED και η αντίσταση R2 ωθεί το δυναμικό εξόδου στην τιμή Vdc όταν το φωτοτρανζίστορ βρίσκεται στην αποκοπή. Γι'αυτόν τον λόγο, η αντίσταση R2 ονομάζεται "pull-up". Επειδή η κυματομορφή εξόδου που λαμβάνεται απο τον συλλέκτη του τρανζίστορ δεν είναι εντελώς τετραγωνική, προστίθεται ένα στοιχείο schmitt trigger. Η λειτουργία του κυκλώματος έχει περιγραφεί αναλυτικά στην θεωρητική μελέτη των αισθητήρων.

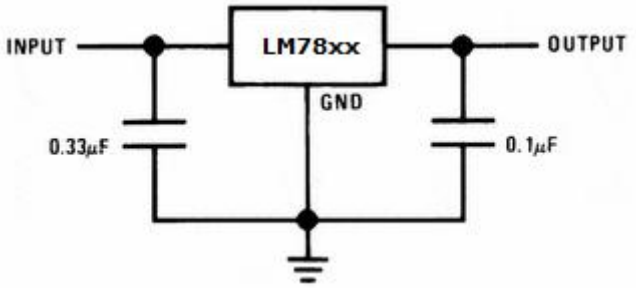
Τροφοδοτικά

Για την τροφοδοσία όλων των κυκλωμάτων χρησιμοποιήθηκε ένα τροφοδοτικό συνεχούς τάσης 24Volts. Τα λογικά κυκλώματα που χρησιμοποιήθηκαν για στην κατασκευή ακολουθούν τις προδιαγραφές της TTL λογικής. Αυτό προυποθέτει τροφοδοσία συνεχούς τάσης 5 Volt. Για την τάση αυτή δεν χρησιμοποιήθηκε ανεξάρτητο τροφοδοτικό, αλλα σταθεροποιητής τάσης που μετατρέπει την dc τάση εισόδου σε σταθερά 5volt ανεξάρτητα με το φορτίο. Ο σταθεροποιητής που χρησιμοποιήθηκε είναι της σειράς LM78xx, όπου xx η τάση εξόδου.

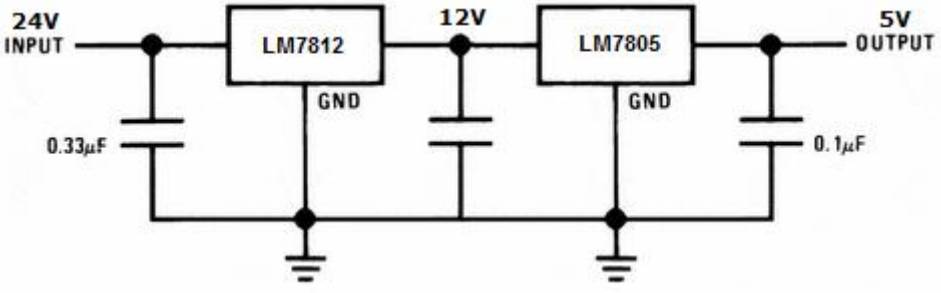
Σχήμα 2.15

Να σημειώσουμε οτι οι σταθεροποιητές αυτοί είναι γραμμικοί και όχι διακοπτικοί. Συνεπώς, όπως έχει αναφερθεί στην ενότητα θεωρητικής περιγραφής των μετατροπέων συνεχούς σε συνεχή τάση, καταναλώνουν σημαντική ισχύ σε μορφή θερμότητας. Μάλιστα, όσο μεγάλύτερη είναι η διαφορα μεταξύ τάσης εισοδου-εξόδου και το ρεύμα που ζητείται, τόσο περισσότερη θερμότητα παράγεται. Γι'αυτόν τον λόγο χρησιμοποιήθηκαν δύο ρυθμιστές σε σειρά, παράγοντας διαδοχικά 12 και 5 volt.

Ετσι η παραγωγή θερμότητας μοιράστηκε στην επιφάνεια δύο ολοκληρωμένων αντί για ένα. Φυσικά, η ανάγκη ψύξης παραμένει, και γι'αυτό χρησιμοποιήθηκε ψύκτρα και ανεμιστήρας. Αναλυτικα θα μελετήσουμε την ψύξη σε επόμενη ενότητα. Να σημειώσω οτι δεν χρησιμοποιήσαμε διακοπτικό σταθεροποιητή τάσης για να αποφύγουμε την είσοδο υψίσυχνων αρμονικών στις γραμμές τροφοδοσίας των λογικών κυκλωμάτων. Οι σταθεροποιητές συνδέθηκαν όπως στο σχήμα: (παρατίθενται τα τεχνικά χαρακτηριστικά)

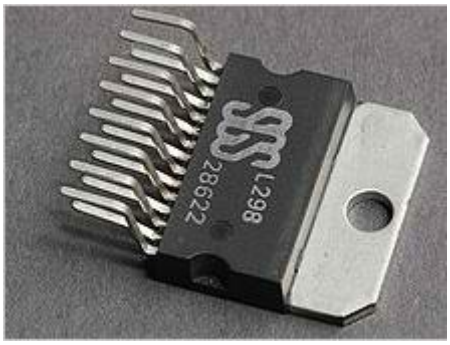


Σχήμα 2.16



Σχήμα 2.17

Κύκλωμα οδήγησης κινητήρα:
Για την οδήγηση του κινητήρα έχει χρησιμοποιηθεί η διάταξη γέφυρας-Η. Χρησιμοποιήθηκε το έτοιμο ολοκληρωμένο L298 σε συσκευασία Multiwatt 15.



Σχήμα 2.18

Το ολοκληρωμένο αυτό περιλαμβάνει δύο γέφυρες-Η, κάθε μία απο τις οποίες μπορεί να οδηγήσει μέχρι 2 Amper ρεύμα. Για λόγους αύξησης ισχύος, οι δύο αυτές γέφυρες συνδέθηκαν παράλληλα ωστε να λειτουργούν ώς μία. Το ολοκληρωμένο αυτό κάνει χρήση διπολικών τρανζίστορ ως διακοπτικά στοιχεία, εισάγοντας έτσι μία πτώση τάσης περιπου 2 volt. Ετσι, στον κινητήρα εφαρμόζεται η ονομαστική τάση των 22 volts στην πλήρη λειτουργία, αφού η γέφυρα τροφοδοτείται απο τα 24 Volt. Η γέφυρα ελέγχεται απο τρία σήματα. Ενα σήμα για την ενεργοποίηση των τρανζίστορ (ENABLE) και δύο για την επιλογή της φοράς ροής του ρεύματος (DIR1,DIR2). Οι καταστάσεις λειτουργίας της γέφυρας φαίνονται στο ακόλουθο πινακάκι. (ENABLE: Ven , DIR1:C, DIR2=D)

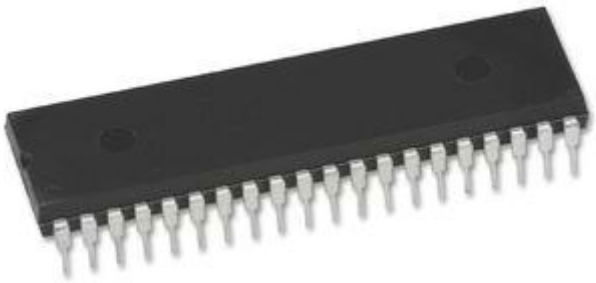
Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Πίνακας 2.1

Το συγκεκριμένο ολοκληρωμένο παρέχει και ένα σήμα ανάδρασης για τον έλεγχο του ρεύματος που διαρρέει το κύκλωμα. Παρατίθενται και τα τεχνικά χαρακτηριστικά.
Παράλληλα με τα διακοπτικά στοιχεία της γέφυρας τοποθετήθηκαν και δίοδοι προστασίας IN4007 για την απορρόφηση των υψηλών τάσεων αντίδρασης του επαγωγικού φορτίου. Η λειτουργία τους έχει αναφερθεί σε προηγούμενη ενότητα.

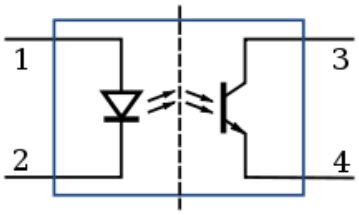
Μικροελεγκτής:
Όπως έχω αναφέρει, ο μικροελεγκτής που επιλέχθηκε είναι ο AVR ATmega16. Το ολοκληρωμένο που χρησιμοποιήθηκε έιναι της μορφής DIP-40 και τροφοδοτείται απο τον σταθεροποιητή 5V. Για τον προγραμματισμό του χρησιμοποιήθηκε η τεχνική ISP, σύμφωνα με την οποία ο μΕ δεν χρειάζεται να αφαιρεθεί απο την πλακέτα λειτουργίας του. Πιο πολλά για την διαδικασία προγραμματισμού θα αναπτυχθούν στην συνέχεια.



Σχήμα 2.19

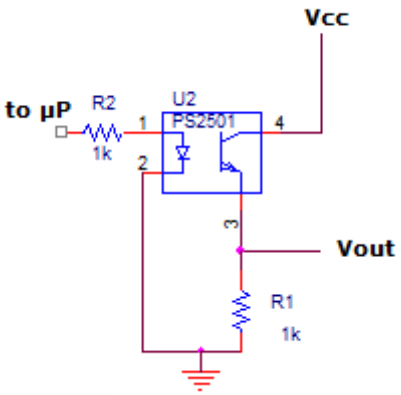
Οπτικοί απομονωτές:

Για επιπλέον ασφάλεια, τα σήματα που εξέρχονται απο τον μΕ και συνδέονται προς την γέφυρα οδήγησης του μοτέρ έχουν απομονωθεί με την χρήση οπτικών απομονωτών (opto-isolators) K814P και PC817. Οι οπτικοί απομονωτές είναι παρόμοιοι σε δομή με τους αισθητήρες των οπτικών κωδικοποιητών. Αποτελούνται απο ένα LED και ένα φωτοτρανζίστορ, τα οποία όμως έχουν αδιατάραχτη οπτική επαφή στο εσωτερικό του ολοκληρωμένου.



Σχήμα 2.20

Ετσι, τα σήματα που τα τροφοδοτούν εμφανίζονται στην έξοδό τους, χωρίς όμως να υπάρχει ηλεκτρική επικοινωνία εισόδου-εξόδου. Το γεγονός αυτό ασφαλίζει το κύκλωμα εισόδου απο τυχόν επιβλαβή παρασιτικά ρεύματα. Το κύκλωμα σύνδεσης φαίνεται στην εικόνα:

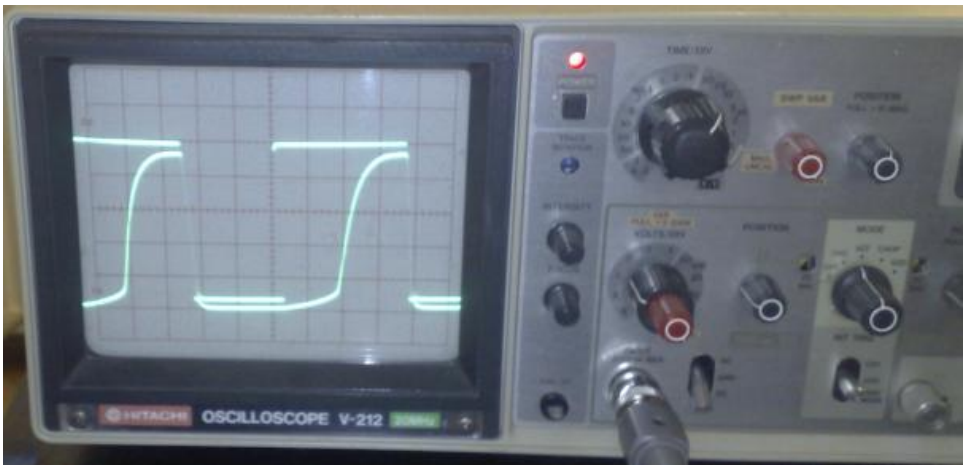


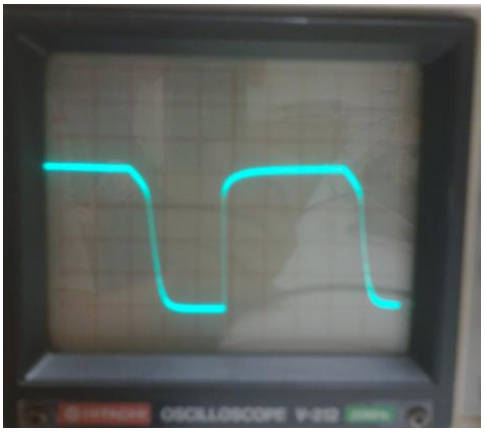
Σχήμα 2.21

Εχουμε συνδέσει τον εκπομπό του φωτοτρανζίστορ με μία pull down αντίσταση και έχουμε φροντίσει με την ρύθμιση της R2 ωστε το τρανζίστορ να μεταβαίνει απο την αποκοπή στην γραμμική περιοχή, και όχι στον κόρο. Ετσι πετυχαίνουμε καλύτερη ταχύτητα ανόδου των παλμών εξόδου.

Σημείωση: Αρχικά δοκιμάστηκε συνδεσμολογία παρόμοι με αυτής των αισθητήρων, όπου ο συλλέκτης του φωτοτρανζίστορ συνδέεται μέσω μιας pull up αντίστασης στο υψηλό δυναμικό και το τρανζίστορ δουλεύει μεταξύ των περιοχών αποκοπής και κόρου. Ωστόσο, αυτή η συνδεσμολογία παρουσιάζει μεγάλο χρόνο ανόδου στην βηματική απόκρισή της με αποτέλεσμα να αδυνατεί να παρακολουθήσει υψίσυχνες παλμοσειρές όπως το σήμα PWM.

Στις φωτογραφίες εμφανίζεται η μορφή των παλμών εξόδου με την καθυστέρηση και μετά την αλλαγή του κυκλώματος. Είναι εμφανής η βελτίωση της απόκρισης.





Σχήμα 2.22

Scmitt trigger

Οπως έχουμε αναφέρει νωρίτερα το παλμικό σήματων οπτικών κωδικοποιητών είναι σύνηθες να φιλτράρεται απο την διάταξη schmitt trigger. Το ολοκληρωμένο που χρησμοποιήθηκε για αυτήν την λειτουργία είναι το 74hc14. Περιέχει έξι ανεξάρτητες διατάξεις scmitt trigger, οι οποίες επι πλέον αντιστρέφουν το σήμα εισόδου.

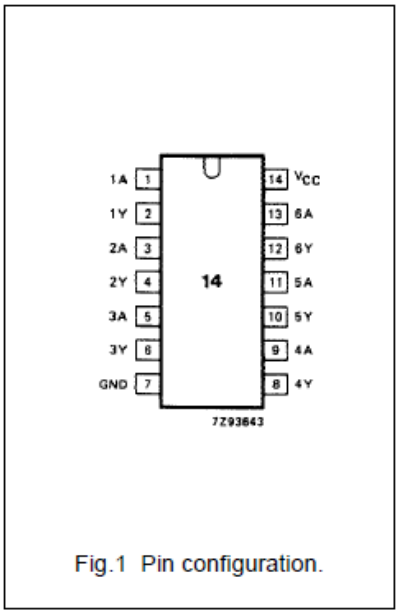


Fig.1 Pin configuration.

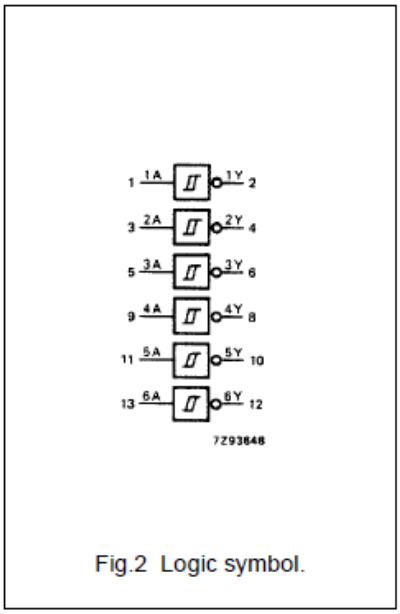


Fig.2 Logic symbol.

Σχήμα 2.23

Το ολοκληρωμένο αυτό τοποθετήθηκε επανω στην πλακέτα και δέχεται σαν είσοδο τα σήματα των τριών οπτικών αισθητήρων του συστήματος. Οι έξοδου του ολοκληρωμένου οδηγούν τις θύρες εισόδου του μΕ.

Τέλος, το σύστημα πρέπει να έχει την δυνατότητα αλληλεπίδρασης με τον χρήστη. Αυτό συνεπάγεται συστήματα εισόδου και εξόδου:

Οθόνη.

Η οθόνη που επιλέχθηκε είναι μία LCD οθόνη χαρακτήρων που διαθέτει 2 γραμμές των 16 χαρακτήρων. Η οθόνη αυτή έχει ενσωματωμένο ελεγκτή για αλληλεπίδραση σύμφωνα με το πρωτόκολλο HD44780.

Σχήμα 2.24

Οι οθόνες αυτές διαθέτουν τυπικά 16 ακροδέκτες, στην εξής διαρρύθμιση:

1. Γείωση
2. τάση τροφοδοσίας (+3.3V ως +5V)
3. ρύθμιση αντίθεσης
4. επιλογέας καταχωρητή (0: εντολών , 1: δεδομένων)
5. επιλογέας Εγγραφής/Ανάγνωσης (0:εγγραφή , 1: ανάγνωση)
6. Ρολόι ενεργοποίησης (ακμοπυροδότητο)

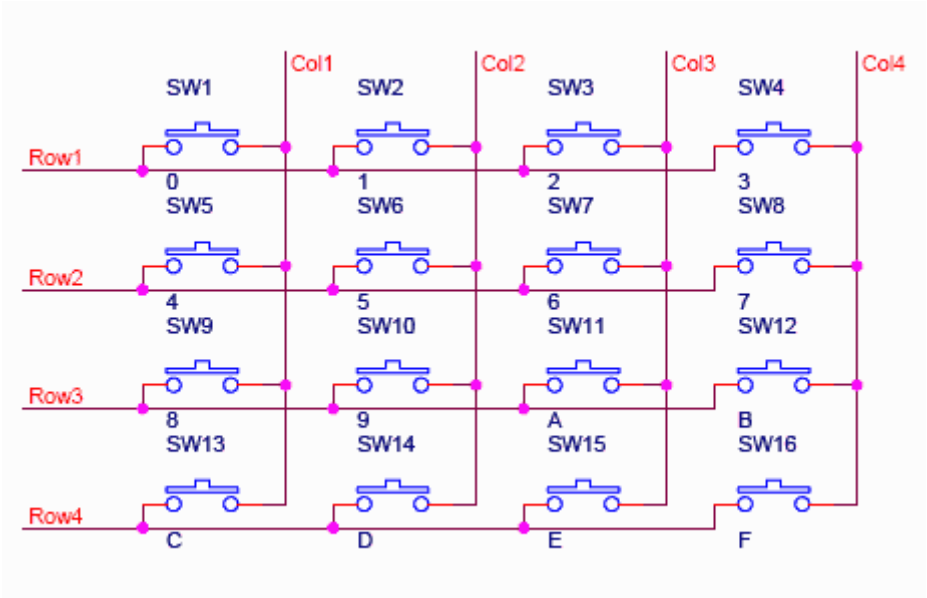
7 – 10 Bit 0-3. (Δεν χρησιμοποιούνται στην λειτουργία 4-bit σύνδεσης)

11-14 Bit 4-7.
15. ανοδος φωτισμού οθόνης
16. κάθοδος φωτισμού οθόνης.

Η οθόνες αυτές μπορούν να συνδεθούν με τον μΕ με δύο τρόπους: Είτε με σύνδεση των ανκροδεκτών 4 – 14 , είτε με σύνδεση μόνο των ακροδεκτών 4,5,6,11-14 στην λειτουργία 4-bit σύνδεσης. Στην εργασία επιλέχθηκε ο δεύτερος τρόπος σύνδεσης. Οι ρουτίνες επικοινωνίας του μΕ με την οθόνη βρέθηκαν αυτούσιες για την γλώσσα προγραμματισμού C σε ανάλογη βιβλιογραφία. Παρατίθεται η αντίστοιχη βιβλιοθήκη καθως και οι προδιαγραφές του πρωτοκόλλου HD44780.

Πληκτρολόγιο:
Για την είσοδο δεδομένων απο τον χρήστη χρησιμοποιήθηκε ένα πληκτρολόγιο 16 πλήκτρων σε συνδεσμολογία πίνακα. Ο πίνακας αυτός περιλαμβάνει 4 στήλες και 4 γραμμές. Τα πλήκτρα λειτουργούν ώς μηχανικοί διακόπτες , που παραμένουν κλειστοί όσο το κάθε πλήκτρο πατιέται. Οταν πατιέται ένα πλήκτρο, ο αγωγός της αντίστοιχης στήλης βραχυκυκλώνεται με τον αγωγό της αντίστοιχης γραμμής.

Σχήμα 2.25

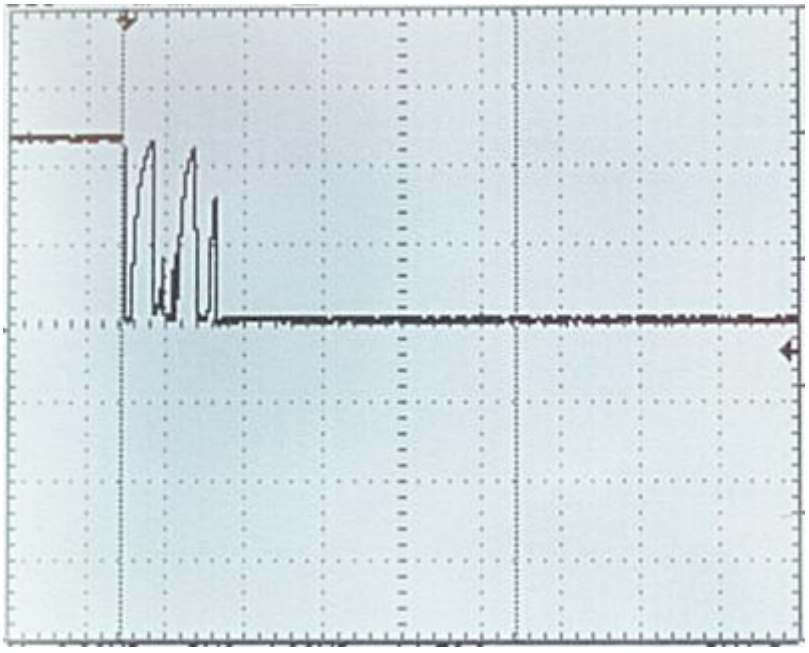


Σχήμα 2.26 Σχηματικό διάγραμμα του πληκτρολογίου

Η σύνδεση με τον μΕ έγινε μέσω 8 bit και η αναγνώριση των πλήκτρων που πατιούνται γίνεται ώς εξής:

Απο τα 8 bit του μΕ τα 4 συνδέονται με τις στήλες και τα άλλα 4 με τις γραμμές του πληκτρολογίου. Οι στήλες αντιμετωπίζονται απο τον μΕ ώς έξοδοι και οι γραμμές ώς είσοδοι,αφού συνδεθούν με pull-up αντιστάσεις προς το λογικό 1.

Για την αναγνώριση ενός πλήκτρου που πατήθηκε, ο μΕ θέτει σε λογικό 0 μία απο τις στήλες, ενώ οι άλλες βρίσκονται σε λογικό 1. Στην συνέχεια, εξετάζει τα bit των γραμμών και αν κάποιο έχει την τιμή 0, αυτό σημαίνει οτι πατιέται το πλήκτρο που βραχυκυκλώνει αυτήν την γραμμή με την στήλη που γειώθηκε. Διαφορετικά όλα τα bits των γραμμών είναι στο λογικό 1, λόγω των pull up αντιστάσεων. Αν δεν βρεθεί κάποιο αποτέλεσμα, η διαδικασία επαναλαμβάνεται και για τις άλλες στήλες. Δηλαδή η ανάγνωση απο το πληκτρολόγιο επιτυχγάνεται με μιά διαδικασία σάρωσης των στηλών μία-μία και ελέγχου όλων των γραμμών κάθε φορά.
Όπως σε κάθε μηχανικό διακόπτη, τα πληκτρολόγια αυτά αντιμετωπίζουν το πρόβλημα της αναπήδησης (contact bounce – chatter) . Το πρόβλημα αυτό έγκειται στο γεγονός οτι ένας μηχανικός διακόπτης όταν πατιέται δεν κλείνει σταθερά το κύκλωμά του, αλλα αναπηδά αρκετές φορές μέχρι να σταθεροποιηθεί. Σε κάθε πάτημα, δηλαδή, ο διακόπτης αρχικά ανοιγοκλείνει αρκετές φορές σε μια περίοδο σταθεροποίησης που μπορέι να κρατήσει έως και 200msec.



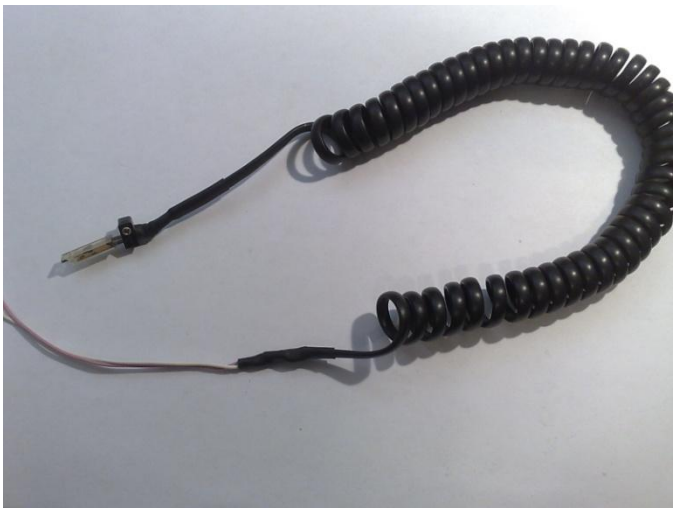
Σχήμα 2.27 Φαινόμενο αναπήδησης μηχανικών επαφών

Εικόνα απο παλμογράφο. Τάση στα άκρα μηχανικού διακόπτη. Το φαινόμενο της αναπήδησης μηχανικής επαφής.
Πηγη: http://www.electroons.com/8051/electroons/images/debounce.png

Στον χρόνο αυτό ο μΕ μπορεί να λάβει απρόβλεπτες πληροφορίες και γι'αυτό καλό είναι να τοποθετούμε μία ρουτίνα καθυστέρησης μετα την πρώτη αναγνώριση κάποιου πλήκτρου. Ετσι ο μΕ προστατεύεται για ένα ασφαλές διάστημα και ο έλεγχος επανέρχεται όταν το πλήκτρο έχει σταθεροποιηθεί. Η τεχνική αυτή ονομάζεται debouncing.

Διακόπτης περιορισμού (limit switch)
Χρησιμοποιήθηκε ένας απλός μηχανικός διακόπτης τοποθετημένος επάνω στο σώμα του τόρνου. Ο διακόπτης αυτός είναι κανονικά ανοικτός και κλείνει όταν έρθει σε επαφή με τον εργαλειοφόρο του τόρνου. Το σήμα του διακόπτη αυτού διαβάζεται απο τον μΕ και

σταματάει την κίνηση του κινητήρα. Σκοπός είναι η αποφυγή μηχανικής βλάβης σε περίπτωση που ο κινητήρας συνεχίζει να κινείται ενώ το σεπόρτι έχει συναντήσει κάποιο εμπόδιο.



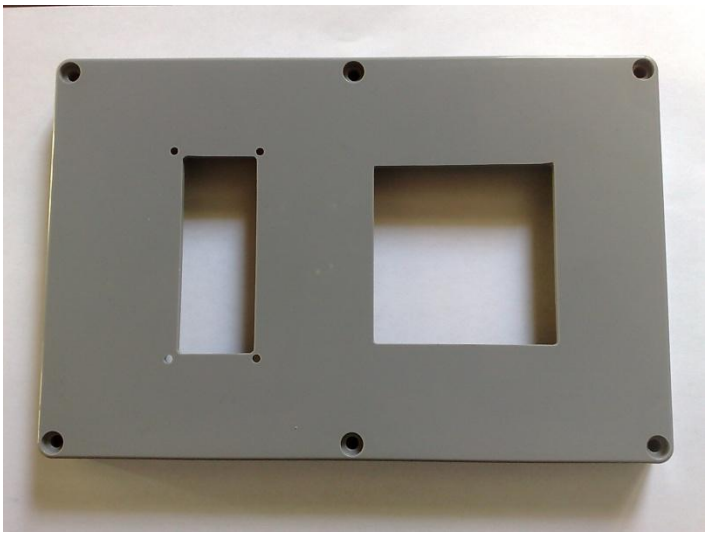
Σχήμα 2.28

Πλακέτα:

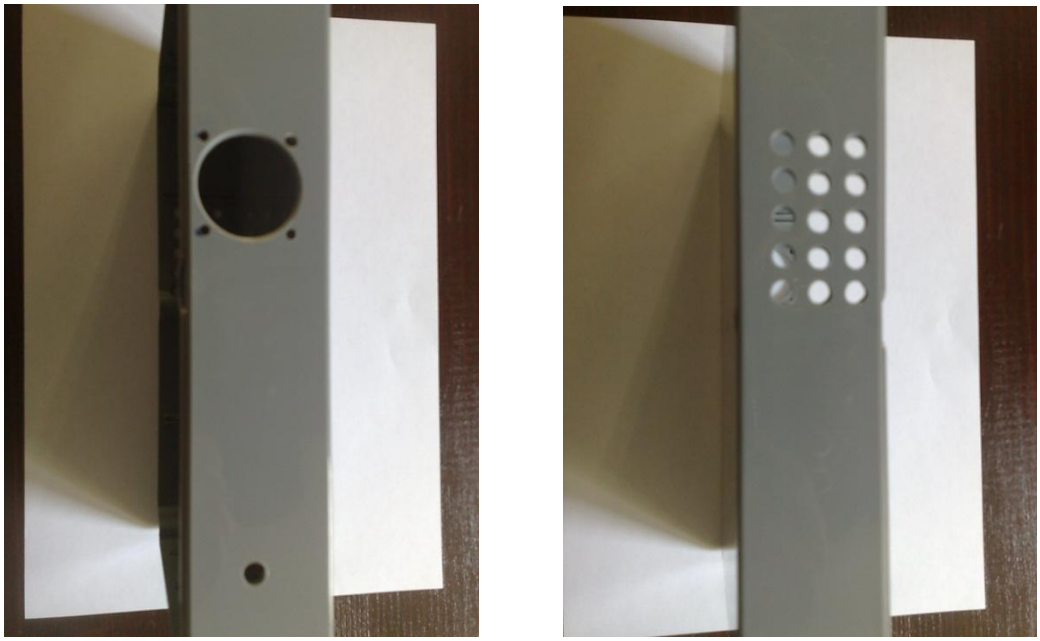
Το κύκλωμα του συστήματος σχηματίστηκε σε μία πλακέτα απο πλεξι-γκλάς με επικάλυψη φύλλου χαλκού και στις δύο όψεις. Τα φύλλα χαλκού προστατεύονται απο επίστρωση φωτοευαίσθητου υλικού. Το στρώμα αυτό είναι ανθεκτικό στα χημικά που διαβρώνουν τον χαλκό αλλα διαλύεται με την έκθεση του σε υπεριώδες φώς. Αυτή η ιδιότητά του χρησιμοποιείται στην τύπωση του κυκλώματος, όπως θα δούμε παρακάτω.

Συσκευασία:

Η πλακέτα με τα στοιχεία του κυκλώματος τοποθετήθηκε σε μία πλαστική συσκευασία. Σε αυτήν την συσκευασία τοποθετήθηκε και ο ανεμιστήρας ψύξης του κυκλώματος και σχηματίστηκαν ειδικές υποδοχές για την οθόνη, το πληκτρολόγιο και την κυκλοφορία του αέρα.



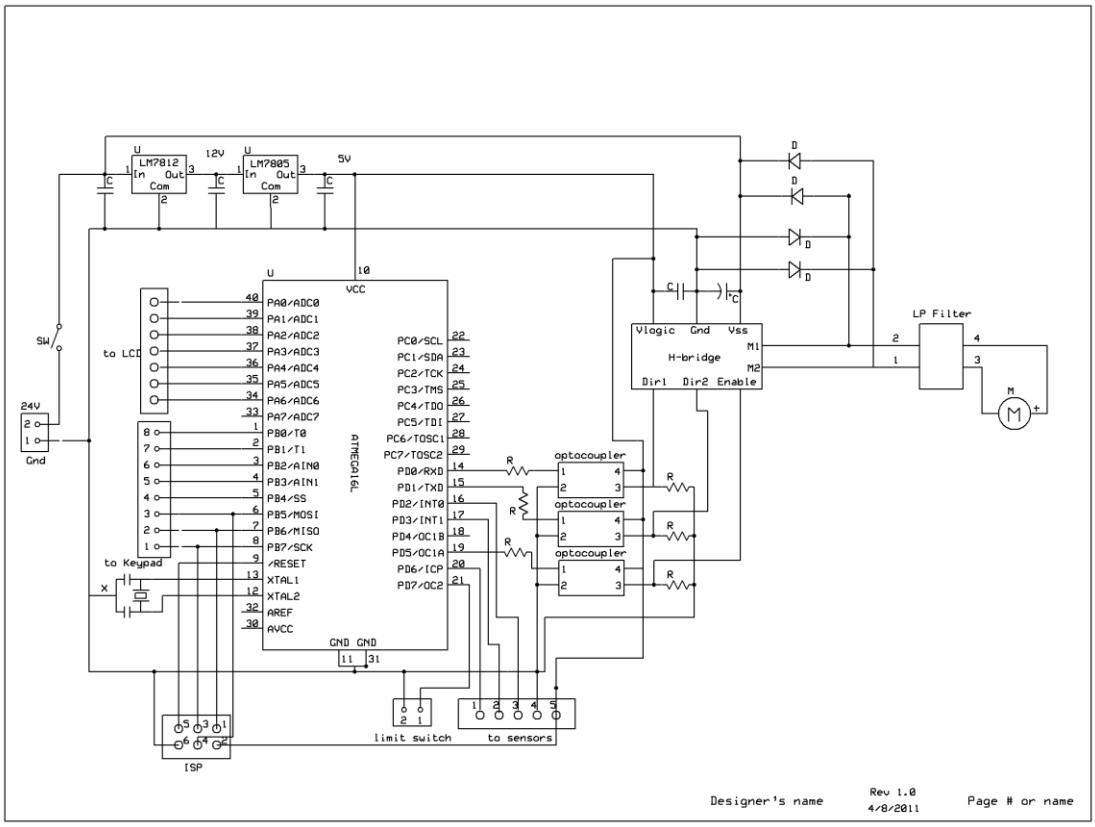
Σχήμα 2.29



Σχήμα 2.30

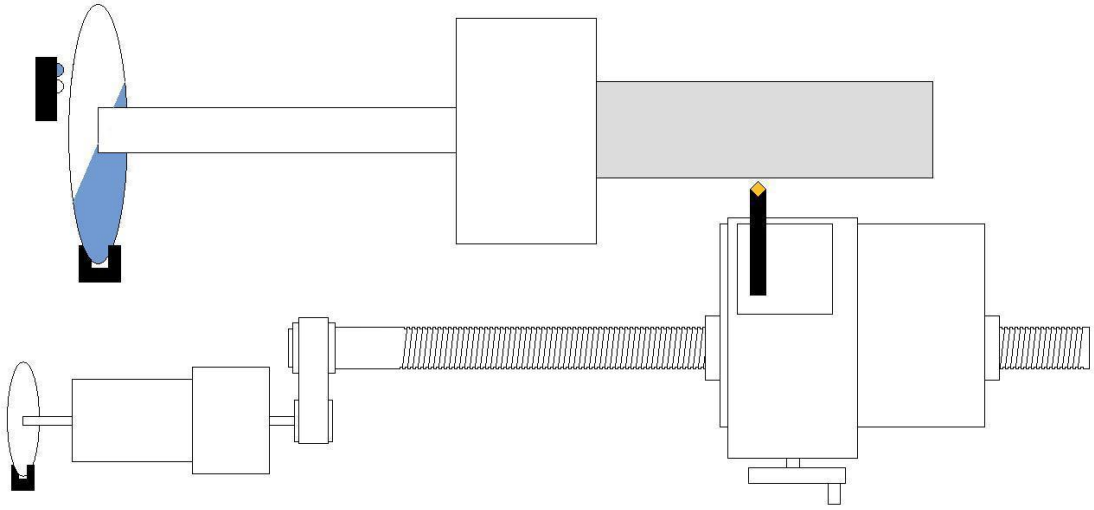
2.4 Διασύνδεση ηλεκτρονικών και μηχανικών στοιχείων

Σε αυτό το στάδιο έχουμε αναφέρει όλα τα στοιχεία που συνθέτουν το σύστημά μας. Ας δούμε το κύκλωμα διασύνδεσής τους:



Σχήμα 2.31

*το κύκλωμα σχεδιάστηκε με το πρόγραμμα expressPCB και expressSCH
Και η σύνδεση των μηχανικών μερών:



Σχήμα 2.32 Κατοψη μηχανικής διάταξης

Σε αυτό το σημείο έχουμε περιγράψει όλα τα στοιχεία που συνθέτουν το σύστημα μας και τον τρόπο διασύνδεσής τους. Στην ενότητα που ακολουθεί, θα αναλύσουμε τον αλγόριθμο του συστήματος.

2.5 Αλγόριθμος συστήματος

2.5.1 Διαδικασία κοπής σπειρώματος

Η εφαρμογή του συστήματος στον αυτοματισμό του τόρνου εισάγει κάποιες ειδικές προκλήσεις και δυσκολίες. Ας δούμε πιο αναλυτικά:

Για διευκόλυνση ας ορίσουμε τις εξής συντεταγμένες στο σύστημα του τόρνου: Ο **άξονας x** είναι παράλληλος στον άξονα του τσώκ. Ο **άξονας y** είναι κάθετος στον x και παράλληλος με το εργαλείο κοπής, και η **γωνία φ** δείχνει την γωνιακή θέση του τσώκ, και άρα του τεμαχίου υπο επεξεργασία.

Όπως είπαμε νωρίτερα, η κοπή του σπειρώματος απαιτεί διαδοχικά περάσματα του κοπτικού εργαλείου απο την διαδρομή της σπείρας. Για να επιτευχθεί αυτό, πρέπει κάθε φορά που επαναλαμβάνουμε ένα πέρασμα να εξασφαλίζουμε οτι η εκκίνηση γίνεται ακριβώς στο ίδιο σημείο του άξονα x και πως το τεμάχιο βρίσκεται ακριβώς στην ίδια γωνιακή θέση $\phi_{εκκ}$ της γωνίας φ. Αν κάποια απο αυτές τις συνθήκες δεν ικανοποιείται, τότε το καινούριο πέρασμα θα γίνει σε μία διαδρομή παράλληλη στην αρχική, και το σπείρωμα δεν θα χαραχθεί ορθά.

Η απαίτηση το τεμάχιο να βρίσκεται στην ίδια γωνιακή θέση $\phi_{εκκ}$ σε κάθε εκκίνηση κοπής αντιμετωπίζεται με την χρήση του δευτερεύοντος, ανακλαστικού αισθητήρα στον άξονα του τσώκ. Οπως έχω αναφέρει, η πληροφορία αυτού του αισθητήρα είναι απόλυτη και αντιστοιχεί στην φυσική θέση του άξονα. Επομένως, αν πυροδοτούμε την εκκίνηση της κοπής σε κάποια ακμή της παλμοσειράς απο αυτόν τον αισθητήρα, έχουμε εξασφαλίσει οτι βρίσκομαστε στην ίδια ακριβώς γωνιακή θέση του τσώκ κάθε φορά.

Εναλλακτικά θα μπορούσαμε να χρησιμοποιήσουμε τον κύριο αισθητήρα του άξονα αναφοράς. Δηλαδή, να αθροίζουμε σε μια μεταβλητή τους παλμούς αυτής της παλμοσειράς, και κάθε φορά που αυτός ο αριθμός φτάνει την τιμή της ανάλυσης του αισθητήρα, να τον μηδενίζουμε. Τότε, συγκεκριμένος αριθμός στην μεταβλητή αυτή θα

σήμαινε συγκεκριμένη σχετική θέση του άξονα απο την εκκίνηση του συστήματος. Ωστόσο, τυχόν σφάλματα στην άθροιση ή θόρυβος στην παλμοσειρά θα συσσωρεύονταν και θα εμπόδιζαν τον σωστό συγχρονισμό. Γι'αυτον τον λόγο επιλέχθηκε η χρήση απόλυτου αισθητήρα για αυτήν την λειτουργία.

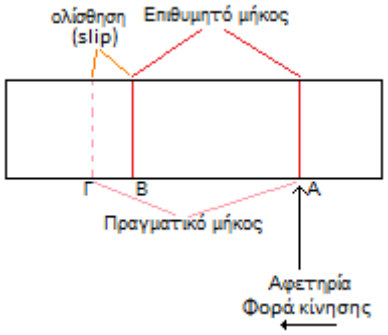
Η απαίτηση για συγκεκριμένη θέση εκκίνησης στον έξονα x έχει να κάνει με την οδήγηση του σερβοκινητήρα.

Θυμίζω οτι η μονάδα που χρησιμοποιούμε για την μέτρηση της γωνιακής μετατόπισης του σερβοκινητήρα είναι το πλήθος παλμών απο τον αισθητήρα του. Σαν επέκταση, η ίδια μονάδα μετράει γραμμική μετατόπιση του εργαλειου κοπής στον άξονα x. Η μετατροπή χρειάζεται γνώση μόνο της **ανάλυσης του αισθητήρα**, της **μείωσης** κατα την μετάδοση της ισχύος στον κοχλία του σεπόρτι και **του βήματος του κοχλία** αυτού. Ολα αυτά είναι σταθερές απαραίτητες για τους υπολογισμούς στον μΕ γενικότερα.

Επομένως όταν ο χρήστης δηλώνει επιθυμητό μήκος σπειρώματος, αυτό μετατρέπεται σε πλήθος παλμών της παλμοσειράς s(t) (θυμίζω, του αισθητήρα του σερβοκινητήρα). Μια μεταβλητή καταμετρά τους παλμούς αυτούς και έτσι ελέγχουμε πότε ο κινητήρας έχει καλύψει την απαιτούμενη απόσταση.

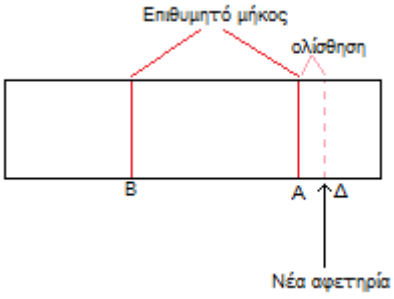
Ωστόσο, αν και η γέφυρα οδήγησης παρέχει την δυνατότητα πέδησης του κινητήρα, η ακινητοποίηση δεν μπορεί να υποτεθεί ακαριαία. Εμφανίζεται σαφώς κάποια ολίσθηση που πρέπει να ληφθεί υπ'όψιν.Η ολίσθηση αυτή μπορεί να μετρηθεί, με το να εισάγουμε μια καθυστέρηση μετά την επιβολή της πέδης και να μετρούμε μετά απο αυτήν (ωστε πλέον ο κινητήρας να είναι ακίνητος) τους επι πλέον καταμετρημένους παλμούς. Αυτην την μετατόπιση την ονομάζουμε slip. Να σημειώσουμε οτι δεν μπορέι να θεωρηθεί σταθερή, αφού εξαρτάται απο ένα πλήθος παραγόντων, όπως η ταχύτητα πριν την πέδηση, η αντισταση , η τριβή ολίσθησης στο συγκεκριμένο σημείο κ.α.

Ας δούμε το φαινόμενο σχηματικά:



Σχήμα 2.33

Εχοντας μετρήσει το μήκος του slip, θέτουμε ως μήκος επιστροφής την απόσταση επιθυμητό μήκος + slip. Στην επιστροφή ωστόσο, πιθανότατα θα έχουμε εκ νέου ολίσθηση:



Σχήμα 2.34

Και πάλι χρειάζεται να ρυθμίσουμε το μήκος κοπής ως επιθυμητό μήκος + slip. Ωστόσο το πρόβλημα δεν έχει διορθωθεί εντελώς.Θυμίζω οτι η σωστή ελικοειδής διαδρομή που πρέπει να ακολουθήσουμε περνάει απο την γωνιακή θέση $\phi_{εκκ}$ του τεμαχίου όταν βρίσκεται στην θέση Α του άξονα x και όχι Δ. Αν ξεκινήσει η κοπή απο την θέση Δ θα διανύσουε μία σπείρα παράλληλη στην επιθυμητή.

Η λύση προκύπτει αν αρχικόποιήσουμε την μεταβλητή σφάλματος θέσης με την τιμή slip. Θυμίζω οτι η μεταβλητή σφάλματος θέσης είναι η είσοδος του ελεγκτή PID και περιέχει την διαφορά της επιθυμητής θέσης του εργαλείου απο την πραγματική. Αρχικοποιώντας την μεταβλητή σφάλματος με την τιμή slip, ο ελεγκτής θα επενεργήσει και στο αρχικό σφάλμα. Φυσικά κατα την πρώτη εκκίνηση του κύκλου κοπής η μεταβλητή σφάλματος αρχικοποιείται με μηδενική τιμή.

2.5.2 Στάδια διεπαφής χρήστη – συστήματος

Ας δούμε τώρα τις καταστάσεις του συστήματος σε υψηλό επίπεδο:

Το σύστημα παρέχει στον χρήστη τις εξής λειτουργίες: Την κοπή σπειρώματος,την αφαίρεση υλικού, την λείανση και την δευτερεύουσα λειτουργία της απλής μετακίνησης.
Να σημειώσω οτι η αφαίρεση υλικού και η λείανση είναι ουσιαστικά διαδικασίες κοπής σπειρωματος και αυτές. Η διαφορά τους είναι οτι το βήμα του σπειρώματος στην περίπτωσή τους είναι αρκετά μικρό ωστε τελικά να αφαιρείται όλη η ποσότητα υλικού και όχι μόνο μία σπείρα. Η λείανση έχει ακόμα μικρότερο βήμα και χρησιμοποιείται για να δημιουργήσει μια ιδιαίτερα λεία επιφάνεια.

Ξεκινάμε με την περίπτωση που ο χρήστης επιλέξει να κόψει κάποιο σπείρωμα:

Βημα 0: Ο χρήστης επιλέγει την λειτουργία κοπής.

Βήμα 1: Ο χρήστης εισάγει το επιθυμητό βήμα σε mm. Εχει την δυνατότητα να εισάγει δεκαδικό αριθμό με έως δύο δεκαδικά ψηφία. Το στάδιο αυτό γίνεται αυτόματα με έτοιμα αποθηκευμένα βήματα για τις διαδικασίες αφαίρεσης υλικού και λείανσης. Η υπόλοιπη διαδικασία είναι κοινή.

Βήμα 2: Στην συνέχεια ο χρήστης δηλώνει αν επιθυμεί δεξιόστροφο ή αριστερόστροφο σπείρωμα.

Βήμα 3: Επόμενο βήμα είναι η εισαγωγή του μήκους του σπειρώματος. Το νούμερο δίνεται σε mm με έως δύο δεκαδικά ψηφία.Πρόκειται για το μήκος επι του άξονα x της επιφάνειας του κυλίνδρου στην οποία θέλουμε να δημιουργήσουμε το σπείρωμα.Ο χρήστης έχει την δυνατότητα να μήν εισάγει κάποιο μήκος, οπότε η διαδικασία θα συνεχίζει μέχρι να την σταματήσει χειροκίνητα.

Βήμα 4: Εν συνεχεία το σύστημα περιμένει απο τον χρήστη μία επιβεβαιωση ωστε να αρχίσει να τροφοδοτεί τον κινητήρα. Σε αυτό το στάδιο ο χρήστης κάνει τις απαιτούμενες χειροκίνητες ρυθμίσεις στο εργαλείο κοπής,όπως το να αυξήσει το βάθος

διείσδησης. Ο χρήστης μπορεί επίσης να ακυρώσει την όλη διαδικασία επιστρέφοντας στο βήμα 0.

Βήμα 5: Οταν δοθεί έγκριση απο τον χρήστη, ενεργοποιείται η οδήγηση του μοτέρ και ο ελεγκτής PID.Η τροφοδοσία του κινητήρα τερματίζεται όταν το σεπόρτι έχει μετακινηθεί κατα το επιθυμητό μήκος ή όταν πατηθεί το πλήκτρο τερματισμού ή κάποιος διακόπτης ασφαλείας.

Βήμα 6: Στην συνέχεια, το σύστημα περιμένει την έγκριση του χρήστη ωστε να επιστρέψει το σεπόρτι στην θέση εκκίνησης του σπειρώματος.Στο σημείο αυτό ο χρήστης κάνει όποιες απαραίτητες ρυθμίσεις, όπως το να επιστρέψει το εργαλείο σε μηδενικό βάθος διείσδησης. Επίσης, μπορέι να ακυρώσει την όλη διαδικασία επιστρέφοντας στο βήμα 0.

βήμα 7:Μόλις δοθεί η έγκριση ο κινητήρας οδηγείται με πλήρη ταχύτητα προς την αντίθετη φορά. Δεν λειτουργεί ο ελεγκτής, και ζητούμενο είναι μόνο να καλυφθεί η σωστή απόσταση.Η διαδικασία τερματίζεται όταν το σεπόρτι βρεθεί στην θέση αφετηρίας ή πατηθεί κάποιο πλήκτρο διακοπής.

Βήμα 8: Το σύστημα επανέρχεται στην κατάσταση του βήματος 4.

Στην περίπτωση της απλής μετακίνησης ,ως επιλογή στο βήμα 0:

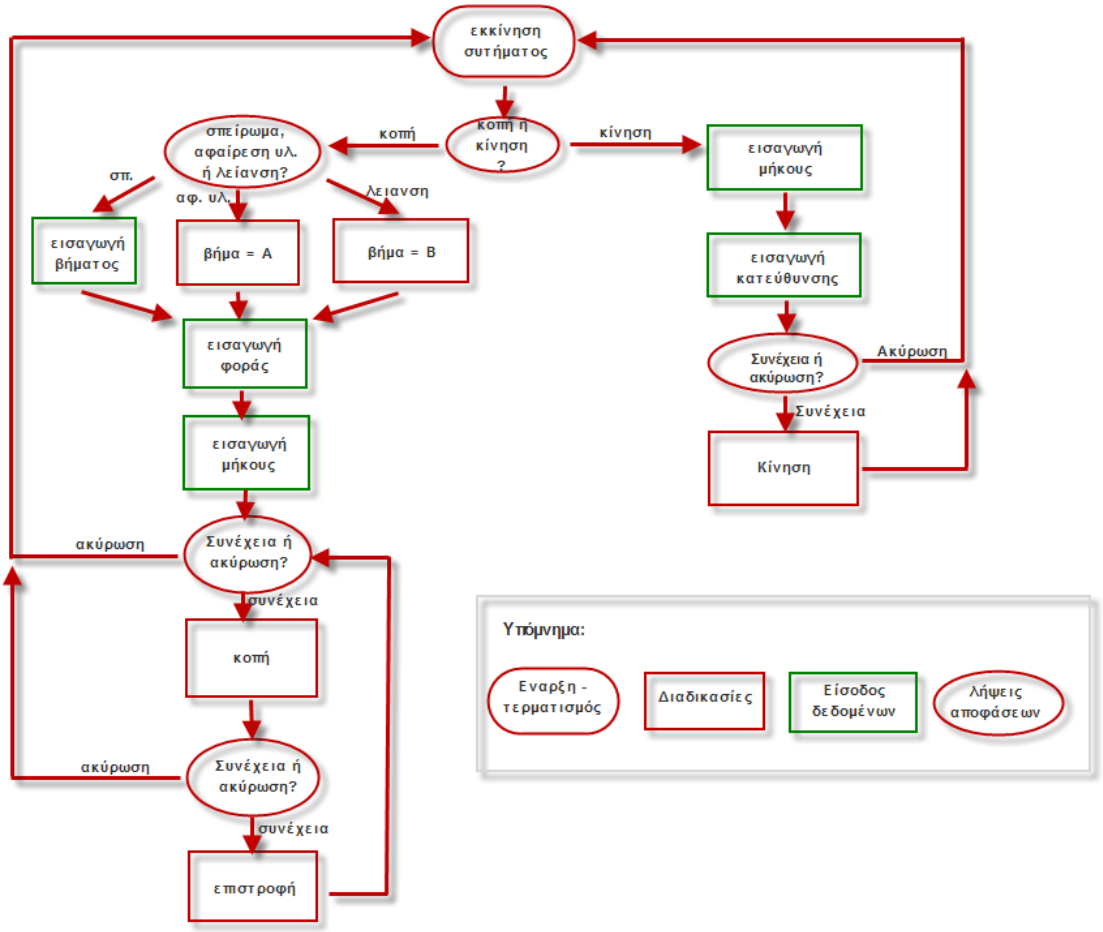
Βήμα 1: Ο χρήστης εισάγει το μήκος της μετακίνησης που επιθυμεί σε mm όπως και στην προηγούμενη περίπτωση

Βήμα 2: Ο χρήστης επιλέγει την φορά της κίνησης.

Βήμα3: Το συστημα περιμένει απο τον χρήστη την έγκριση για την έναυση της κίνησης. Ο χρήστης μπορεί επίσης να ακυρώσει την όλη διαδικασία επιστρέφοντας στο βήμα 0.
Βήμα 4: Το μοτέρ οδηγείται με πλήρη ταχύτητα. Η διαδικασία τερματίζει όταν καλυφθεί η απαιτούμενη απόσταση ή όταν πατηθεί κάποιο πλήκτρο διακοπής

Βήμα 5: Το σύστημα επανέρχεται στην κατάσταση του βήματος 0.

Ας δούμε και το αντίστοιχο διάγραμμα ροής:



Σχήμα 2.35

2.5.3 Κώδικας αναλυτικά

Ας δούμε τώρα τον κώδικα του συστήματος αναλυτικά:

Ο μικροελεγκτής που χρησιμοποιήσαμε διαθέτει σύνολο 131 εντολών. Ωστόσο η πολυπλοκότητα του προγράμματος καθιστά ιδιαίτερα δυσχερή τον προγραμματισμό σε επίπεδο assembly. Επομένως, επέλεξα την χρήση της γλώσσας υψηλού επιπέδου C. Λόγος για την επιλογή αυτή είναι κυρίως η ευρέως διαδεδομένη χρήση της και η ύπαρξη έτοιμων προγραμμάτων για την μεταγλώττισή της στη γλώσσα assembly των μικροελεγκτών AVR.

Έγινε προσπάθεια ο κώδικας να είναι συνοπτικός και περιεκτικός και ακολουθήθηκαν οι αρχές του δομημένου προγραμματισμού. Ο κώδικας αποτελείται στο μεγαλύτερό του μέρος απο υπορουτίνες και μία συνάρτηση main() , η οποία αναλαμβάνει κυρίως την αρχικοποίηση του συστήματος, τον συντονισμό των υπορουτίνων και την διεπαφή με τον χρήστη.

Ο κώδικας θα παρουσιαστεί ανα υπορουτίνα. Σε κάθε τμήμα κώδικα θα αναφέρονται οι βιβλιοθήκες , οι δομές και οι καθολικές μεταβλητές που χρησιμοποιούνται, καθώς και επεξηγηματικά σχόλια της λειτουργίας τους. Το πλήρες κείμενο του αλγορίθμου παρατίθεται επίσης αυτούσιο σε παράρτημα.

Να σημειώσω οτι οι καθολικές μεταβλητές που χρειάζεται να είναι διαθέσιμες τόσο στις διάφορες υπορουτίνες, όσο και στις ρουτίνες εξυπηρέτησης των διακοπών, δεν αρκεί να δηλωθούν ως καθολικές. Χρειάζεται να χαρακτηριστούν volatile. Η ρύθμιση αυτή δίνει

την εντολή στον compiler να αναθέσει μία θέση μνήμης σε αυτές τις μεταβλητές ακόμα και αν φαίνεται απο τον κώδικα πως δεν θα μεταβληθεί η τιμή τους.

Ας δούμε τώρα τις επιμέρους ρουτίνες. Θα ξεκινήσουμε με τις ρουτίνες διεπαφής χρήστη-συστήματος:

Ρουτίνες εμφάνισης στην οθόνη

Η οθόνη έχει συνδεθεί στην θύρα A του μΕ. Οι ρουτίνες επικοινωνίας διαχειρίζονται το πρωτόκολλο HD44780 και έχουν βρεθεί έτοιμες σε αντίστοιχη βιβλιογραφία. Προστέθηκαν στον κώδικα μέσω της βιβλιοθήκης lcd.h

Συγκεκριμένα, χρησιμοποιήθηκαν οι εξής ρουτίνες:

- lcd_init(attitude) για αρχικοποίηση της οθόνης. Το όρισμα attitude ρυθμίζει τον τρόπο εμφάνισης του κέρσορα στην οθόνη.
- lcd_putc() για εμφάνιση χαρακτήρων
- lcd_puts() για εμφάνιση συμβολοσειρών.

Για την εμφάνιση αριθμών χρειάστηκε η ρουτίνα itoa() απο την βιβλιοθήκη stdlib.h, η οποία μετατρέπει αριθμούς σε αριθμητικές συμβολοσειρές. Η χρήση της απαιτεί έναν πίνακα χαρακτήρων , στην θέση του οποίου χρησιμοποιήθηκε η καθολική μεταβλητή uint8_t str[10]

Η συνάρτηση lcd_puts() δέχεται ορίσματα απο την μνήμα RAM του μΕ. Λόγω περιορισμένου όγκου της μνήμης αυτής, θεωρήθηκε σκόπιμο κάποια μηνύματα προς εμφάνιση που δεν αλλάζουν κατα την εκτέλεση του προγραμματος, να βρίσκονται στην μνήμα προγράμματος Flash. Για να γίνει αυτό χρησιμοποιήθηκε η επίσης διαθέσιμη παραλλαγή της συνάρτησης , η lcd_puts_p(). Η συνάρτηση αυτή δέχεται ορίσματα απο την μνήμη προγράμματος. Επίσης, χρειάστηκε η μακροεντολή PSTR() της βιβλιοθήκης pgmspace.h. Η μακροεντολή αυτή δηλώνει στον compiler σε ποιά μνήμη να αποθηκεύσει το όρισμά της κατα την μεταγλώτιση του κώδικα και την φόρτωσή του στον μΕ.

Οι αναλυτικοί κώδικες των συναρτήσεων που αναφέρθηκαν παρατίθενται σε παράρτημα.

Ρουτίνα ανάγνωσης απο το πληκτρολόγιο.

Το πληκτρολόγιο έχει συνδεθεί στην θύρα PORTB, με τον εξής τρόπο:
pin7 : στήλη 1 (output)
pin6 : στήλη 2 >>
pin5 : στήλη 3 >>
pin4 : στήλη 4 >> >>
pin3 : γραμμή 1 (input), χρήση της εσωτερικής pull up αντίστασης
pin2 : γραμμή 2 >> >>
pin1 : γραμμή 3 >> >>
pin0 : γραμμή 4 >> >>

Η ρύθμιση της θύρας PORTB γίνεται στο παρακάτω τμήμα κώδικα στην main(), όπου ρυθμίζονται οι είσοδοι-έξοδοι, ενεργοποιούνται οι pull-up αντιστάσεις των εισόδων και οι έξοδοι αρχικοποιούνται με λογικό 0 στη $4^{η}$ στήλη. Αυτή η κατάσταση επιτρέπει τον έλεγχο των πλήκτρων της $4^{ης}$ στήλης χωρίς την εκτέλεση της ρουτίνας ανάγνωσης απο το πληκτρολόγιο. Ετσι, χρησιμοποιούμε τα πλήκτρα αυτά ως πλήκτρα γενικού σκοπού για την

επιβεβαίωση ή ακύρωση δεδομένων.Η καθυστέρηση εισάγεται ωστε να σταθεροποιηθούν οι έξοδοι πριν χρειαστεί να ελεγχθούν.

```
KEYPAD_DDR =0xF0;
KEYPAD_PORT =0b11101111;
_delay_ms (1);
```

Η συναρτήσεις _delay_ms() και _delay_us () ανήκουν στην βιβλιοθήκη util/delay.h και για την ορθή λειτουργία τους χρειάζεται ο ορισμός μίας σταθεράς για την συχνότητα του ρολογιού του μΕ, μέσω της γραμμής:

```
#define F_CPU 8000000UL
```

Εχει οριστεί μία μακροεντολή για την καθυστέρηση για την απαλοιφή της αναπήδησης των μηχανικών επαφών που κάνει χρήση των ρουτίνων καθυστέρησης:

```
#define DEBOUNCE() (_delay_ms(200))
```

Η βιβλιοθήκη inttypes.h μας δίνει την δυνατότητα να δηλώνουμε μεταβλητές με τύπο uintx_t για μή προσεσημασμένο αριθμό με x bits ή intx_t αντίστοιχα για προσεσημασμένο.

Η συνάρτηση read_key() χρησιμοποιεί την μέθοδο που έχουμε περιγράψει για να σαρώσει το πληλτρολόγιο και να αναζητήσει πατημένο πλήκτρο. Οταν βρεθεί, επιστρέφεται η μεταβλητή key που περιέχει τον αριθμό του πλήκτρου που πατήθηκε. Οταν η μεταβλητή αυτή έχει τιμή 16, σημαίνει οτι δεν έχει πατηθεί καένα πλήκτρο και η αναζήτηση συνεχίζεται.

```
uint8_t read_key(){


   uint8_t key = 16;  //key=16 indicates that no key is pressed

   KEYPAD_DDR =0xF0; // columns : outputs, lines :  inputs
   KEYPAD_PORT=0x0F; // inputs with internal pull up resistors

   while (key == 16) //loop key check until a key  is pressed
    {
        //first column
        KEYPAD_PORT =0b01111111;
        //check for rows and send key number
        _delay_ms(1);      //for the port bits to settle

        if ((KEYPAD_PIN & 0b00001000) == 0) {DEBOUNCE(); key = 1;}
        if ((KEYPAD_PIN & 0b00000100) == 0) {DEBOUNCE();  key = 4;}
        if ((KEYPAD_PIN & 0b00000010) == 0) {DEBOUNCE();  key = 7;}
        if ((KEYPAD_PIN & 0b00000001) == 0) {DEBOUNCE(); key = 14;}

        //second column
        KEYPAD_PORT =0b10111111;
        _delay_ms(1);
        if ((KEYPAD_PIN & 0b00001000) == 0) {DEBOUNCE();  key = 2;}
        if ((KEYPAD_PIN & 0b00000100) == 0) {DEBOUNCE();  key = 5;}
        if ((KEYPAD_PIN & 0b00000010) == 0) {DEBOUNCE();  key = 8;}
        if ((KEYPAD_PIN & 0b00000001) == 0) {DEBOUNCE();  key = 0;}

        //third column
        KEYPAD_PORT =0b11011111;
```

```
        _delay_ms(1);
        if ((KEYPAD_PIN & 0b00001000) == 0) {DEBOUNCE();  key = 3;}
        if ((KEYPAD_PIN & 0b00000100) == 0) {DEBOUNCE();  key = 6;}
        if ((KEYPAD_PIN & 0b00000010) == 0) {DEBOUNCE();  key = 9;}
        if ((KEYPAD_PIN & 0b00000001) == 0) {DEBOUNCE();  key = 15;}

        //fourth column
        KEYPAD_PORT =0b11101111;
        _delay_ms(1);
        if ((KEYPAD_PIN & 0b00001000) == 0) {DEBOUNCE();  key = 10;}
        if ((KEYPAD_PIN & 0b00000100) == 0) {DEBOUNCE(); key = 11;}
        if ((KEYPAD_PIN & 0b00000010) == 0) {DEBOUNCE(); key = 12;}
        if ((KEYPAD_PIN & 0b00000001) == 0) {DEBOUNCE();  key = 13;}
    }
    return(key);
}
```

Οπως βλέπουμε, η επιστροφή της συνάρτησης θα γίνεται πάντα μετά τον έλεγχο της 4^{ης} στήλης. Ετσι, παραμένει γειωμένη η έξοδός της και μπορούμε να την ελέγχουμε και χωρίς την κλήση της συνάρτησης read_key(), απλώς με έλεγχο της τιμής των bits των γραμμών του πληκτρολογίου. Τα πλήκτρα της 4^{ης} στήλης είναι τα «A , B , C , D» και ο έλεγχός τους γίνεται απο τις εξής μακροεντολές:

```
#define PRESSED_A (!(KEYPAD_PIN & 0b00001000))
#define PRESSED_B (!(KEYPAD_PIN & 0b00000100))
#define PRESSED_C (!(KEYPAD_PIN & 0b00000010))
#define PRESSED_D (!(KEYPAD_PIN & 0b00000001))
```

Οι μακροεντολές αυτές χρησιμοποιούνται ευρέως μεσα στον κώδικα.Συγκεκριμένα:

- Το πλήκτρο D χρησιμοποιείται ώς «Enter» για επιβεβαίωση δεδομένων απο τονχρήστη.Επίσης, τερματίζει όποια διαδικασία κίνησης του κινητήρα, ώς πλήκτρο ασφαλείας.Εαν πατηθεί κατα την ενεργοποίηση του συστήματος, το σύστημα εισέρχεται σε κατάσταση αλλαγής των ρυθμίσεων του.
- Το πλήκτρο C χρησιμοποιείται για ακύρωση δεδομένων και επιστροφή στην αρχική κατάσταση του συστήματος.Αν πατηθεί κατα την ενεργοποίηση το σύστημα εισέρχεται σε κατάσταση ελέχγου της λειτουργίας των αισθητήρων.
- Τα πλήκτρα B και A έχουν βοηθητικες λειτουργίες.

Ρουτίνα ανάγνωσης δεκαδικού αριθμού απο το πληκτρολόγιο.

Στο πληκτρολόγιο το πλήκτο της δίεσης, με κωδικό 15 χρησιμοποιείται ώς υποδιαστολή. Τα υπόλοιπα μή αριθμητικά πλήκτρα του πληκτρολογίου αγνοούνται. Η ρουτίνα read_numb() επιτρέπει την είσοδο ενός αριθμού με ακέραιο μέρος απεριόριστο (θεωρητικά) και μέχρι δύο δεκαδικά ψηφία. Οτιδήποτε περισσότερο αγνοείται. Η εισαγωγή αριθμού τερματίζεται με το πλήκτρο D, που αντιμετωπίζεται ώς «enter». Στην συνέχεια, αυτός ο αριθμός επιστρέφεται απο την συνάρτηση στο καλόν πρόγραμμα αφού πολλαπλασιαστεί επι 100, ωστε να είναι πάντα ακέραιος.

```
uint16_t read_numb(){
    uint8_t y=0;           //every new digit
    uint16_t x=0;          // the whole number

    y = read_key();
    while ( y != 13 && y != 15){  //13 = 'OK',  15= comma','
       if (y<=9){
                              //ignore digits >9
          lcd_putc (y + '0');  //show digit in char form
```

```
            x = x*10;
            x=x+y;
        }
        y = read_key();
    }

    if(y == 15){lcd_putc('.');}

    x=x*100;                        //no more integer part
    while((y>9 && y<13)||y>13 ){y = read_key();}//ignore any y>9 except 13("OK")
    if (y == 13){return(x);}        //in case there is no decimal part

    lcd_putc (y + '0');

    y = y*10;                       //first decimal digit
    x = x+y;
    y = read_key();
    while((y>9 && y<13)|| y>13 ){y = read_key();}
    if (y == 13){return(x);}        //in case there is only one decimal digit

    lcd_putc('0'+y);
    x = x+y;                        //second decimal digit
    while (y != 13){y = read_key();}//waiting for "OK". Anything else is ignored
    return(x);
}
```

Ρουτίνα ελεγκτή PID

Η ρουτίνα αυτή εκτελείται εφόσον είναι ενεργοποιημένος ο έλεγχος απο το πρόγραμμα, ανα σταθερά χρονικά διαστήματα. Η περιοδική εκτέλεσή της επιτυγχάνεται με την δήλωση της ώς ρουτίνα διακοπής, η οποία πυροδοτείται απο τον χρονιστή timer0. Ο χρονιστής αυτός ρυθίζεται ωστε να μετράει μέχρι έναν συγκεκριμένο αριθμό, να προκαλεί την διακοπή οταν τον φθάσει, στην συνέχεια να μηδενίζει και να επαναλαμβάνει τον κύκλο. Η περίοδος του κύκλου αυτού είναι η περίοδος εκτέλεσης του ελεγκτή.Η τιμή της, μαζί με τις τιμές των κερδών Kp, Ki, Kd του ελεγκτή αποθηκεύονται απο τον χρήστη στην μόνιμη μνήμη EEPROM μέσω της ρουτίνας:

```
void configuration(){
    lcd_clrscr();
    lcd_puts_p(PSTR("Configuration\n PID data"));
    _delay_ms(1000);

    lcd_clrscr();
    lcd_puts_p(PSTR("time intervals\n (ms/8):"));
    eeprom_write_byte ((uint8_t*)3, (uint8_t)(read_numb()/100));

    lcd_clrscr();
    lcd_puts_p(PSTR("K_p * 32: "));
    eeprom_write_word ((uint16_t*)4,(read_numb()/100));

    lcd_clrscr();
    lcd_puts_p(PSTR("K_i * 32: "));
    eeprom_write_word ((uint16_t*)6,(read_numb()/100));

    lcd_clrscr();
    lcd_puts_p(PSTR("K_d * 32: "));
    eeprom_write_word ((uint16_t*)8,(read_numb()/100));
return;
}
```

Να σημειώσω οτι για λόγους ακρίβειας τα κέρδη του ελεγκτή δίνονται πολλαπλασιασμένα επι τον αριθμό `2^SCALING_FACTOR_BITS` που είναι ορισμένος ως

```
#define SCALING_FACTOR_BITS  5
```

Επομένως, τελικά οι τιμές που εισάγονται είναι πολλαπλασιασμένες επι 32. Επίσης, το κέρδος Kd είναι ουσιαστικά το κέρδος του ελεγκτή πολλαπλασιασμένο επι την περίοδο εκτέλεσής του και το κέρδος Ki διηρημένο με τον ίδιο αριθμό. Ο λόγος έχει αναφερθεί στην θεωρητική ανάλυση του ελεγκτή, όπου παρουσιάστηκε η μετατροπή του ελεγκτή συνεχούς χρόνου σε διακριτού. Ωστόσο, καταχρηστικά τις τιμές αυτές θα τις αναφέρουμε ώς K_p,K_i,K_d.

Η αρχικοποίηση του χρονιστή γίνεται ώς εξής:

```
TCCR0 = 0b00001000;    //ctc mode
OCR0 = eeprom_read_byte((uint8_t*)3); //pid execution time intervals
TIMSK |= (1 << 1);     //enable compare match interrupt
```

Τα 3 lsb bits του καταχωρητή δηλώνουν τον διαιρέτη συχνότητας (prescaler) της συχνότητας ρολογιού που θα χρονίζει τον μετρητή. Η τιμή 000 ουσιαστικά σημαίνει απενεργοποίηση του μετρητή. Η ενεργοποίηση και απενεργοποίηση του χρονιστή, και άρα και της εκτέλεσης του ελεγκτή ρυθμίζεται απο αυτά τα bits μέσω των μακροεντολών:

```
#define PID_START() (TCCR0 |= 0b101)        //run timer0 with 1/1024 prescaler
#define PID_STOP()  (TCCR0 &= 0b11111010)
```

Ο ελεγκτής εκτός απο την μεταβλητή σφάλματος και τις τιμές των κερδών Kp, Ki, Kd, που αποθηκεύονται απο τον χρήστη στην μνήμη EEPROM χρειάζεται και κάποιες άλλες μεταβλητές. Αυτές είναι μία μεταβλητή όπου να αποθηκεύει την συσσώρευση του σφάλματος και μία άλλη μεταβλητή όπου θα βρίσκεται συνεχώς η προηγούμενη τιμή του σφάλματος. Επίσης, χρειάζονται κάποιες τιμές-όρια για να αποφευχθεί η υπερχείλιση των μεταβλητών. Ολα αυτά τα στοιχεία αποθηκεύονται στην εξής ειδική δομή:

```
typedef struct PID_DATA{
  //! Last process value, used to find derivative of process value.
  int16_t lastError;
  //! Summation of errors, used for integrate calculations
  int32_t sumError;
  //! The Proportional tuning constant,
  uint16_t P_Factor;
  //! The Integral tuning constant,
  uint16_t I_Factor;
  //! The Derivative tuning constant,
  uint16_t D_Factor;
  //! Maximum allowed error, avoid overflow
  int16_t maxError;
  //! Maximum allowed sumerror, avoid overflow
  int32_t maxSumError;
} pidData_t;
```

Η καθολική μεταβλητή-δείκτης pid_st χρησιμοποιείται για την πρόσβαση σε αυτήν την δομή, μεσω των εντολών:

```
pidData_t   *pid_st;
pid_st = (pidData_t*)malloc(sizeof (pidData_t));
```

Στην εκκίνηση του προγράμματος, οι τιμές των πεδίων της δομής αρχικοποιούνται μέσω της ρουτίνας:

```
void pid_init ( struct PID_DATA *pid){
```

```
        pid->sumError = 0;
        pid->lastError = 0;

        pid->P_Factor = eeprom_read_word((uint16_t*)4);    //= Kp
*2^SCALING_FACTOR_BITS
        pid->I_Factor = eeprom_read_word((uint16_t*)6);    //= Ki
*2^SCALING_FACTOR_BITS
        pid->D_Factor = eeprom_read_word((uint16_t*)8);    //= Kd
*2^SCALING_FACTOR_BITS

        pid->maxError = MAX_INT / (pid->P_Factor + 1);
        pid->maxSumError = MAX_I_TERM / (pid->I_Factor + 1);
    }
```

Σε αυτήν την υπορουτίνα αντιγράφονται οι τιμές των κερδών απο την μνήμη EEPROM στην RAM. Επίσης, αποθηκεύονται τα όρια για τις διάφορες μεταβλητές που έχουν οριστεί στις γραμμές:

```
#define MAX_INT         INT16_MAX
#define MAX_LONG        INT32_MAX
#define MAX_I_TERM      (MAX_LONG / 2)
```

Οι σταθερές που χρησιμοποιήθηκαν δίνονται στην βιβλιοθήκη stdint.h Ας δούμε τώρα το σώμα της ρουτίνας του ελεγκτή:

```
ISR (TIMER0_COMP_vect, ISR_NOBLOCK){
//PID execution code here
   int16_t pid_error, p_term, d_term;
   int32_t i_term, ret, temp;
   pid_error = Error;
   //Error =0;
   // Calculate Pterm and limit error overflow

   if (pid_error > pid_st->maxError){
      p_term = MAX_INT;
   }
   else if (pid_error < -pid_st->maxError){
      p_term = -MAX_INT;
   }
   else{
      p_term = pid_st->P_Factor * pid_error;
   }
   // Calculate Iterm and limit integral runaway
   temp = pid_st->sumError + pid_error;

   if(temp > pid_st->maxSumError){
      i_term = MAX_I_TERM;
      pid_st->sumError = pid_st->maxSumError;
   }
   else if(temp < -pid_st->maxSumError){
      i_term = -MAX_I_TERM;
      pid_st->sumError = -pid_st->maxSumError;
   }
   else{
      pid_st->sumError = temp;
      i_term = pid_st->I_Factor * pid_st->sumError;
   }
   // Calculate Dterm
   d_term = pid_st->D_Factor * (pid_error  - pid_st->lastError );
   pid_st->lastError = pid_error;

   // Output:
   ret = (p_term + i_term + d_term)>> SCALING_FACTOR_BITS;
```

```
    if(ret > PWM_TOP){
      ret = PWM_TOP;
    }
    else if(ret < 0){
      ret =0;
    }
    OCR1A = ret;
  }
```

Η έξοδος του ελεγκτή, όπως ξέρουμε, είναι η τάση οδήγησης του κινητήρα. Η τάση αυτή εκφράζεται σε duty cycle του PWM σήματος. Η τιμή του duty cycle ρυθμίζεται απο τον καταχωρητή OCR1A, όπως θα δούμε στην συνέχεια.

Παραγωγή σήματος PWM.

Η δημιουργία της κυματομορφής PWM γίνεται σε επίπεδο υλικού απο τον χρονιστή timer1 του μΕ. Ετσι, δεν χρειάζεται η υλοποίηση με κώδικα, εκτός απο την ρύθμιση των παραμέτρων του χρονιστή:

```
ICR1= PWM_TOP;
TCCR1A = 0b00000010; // fast PWM mode,non inverting,
TCCR1B = 0b00011001; //not working, until TCCR1A becomes 0b10000010
OCR1A = 0;
```

Ο χρονιστής ρυθμίζεται στην λειτουργία fastPWM. Το άνω όριο της μέτρησης αποθηκεύεται στον καταχωρητή ICR1 και, μαζί με τον prescaler και το ρολόι του μΕ καθορίζει την συχνότητα του σήματος. Η τιμή που έχουμε ορίσει είναι:

```
#define PWM_TOP 0x03FF
```

Στην λειτουργία fastPWM ο ακροδέκτης PORTD5 διαμορφώνει την τιμή του ανάλογα με την σύγκριση των τιμών του μετρητή με τον καταχωρητή OCR1A. Ετσι , ο καταχωρητής αυτός ρυθμίζει το duty cycle της παλμοσειράς εξόδου.Στην αρχικοποίηση τίθεται ίσος με 0. Η ενεργοποίηση και απενεργοποίηση της παραγωγής του σήματος PWM ελέγχεται μέσω δύο μακροεντολών που αλλάζουν την ρύθμιση του prescaler του χρονιστή:

```
#define PWM_START()(TCCR1A |= 0b10000000) // set MSB == start PWM with no prescaler
#define PWM_STOP() (TCCR1A &= 0b01111111)  // reset MSB == stop PWM
```

Το σήμα PWM τροφοδοτείται στην είσοδο Enable της γέφυρας οδήγησης του κινητήρα.

Ρουτίνα δημιουργίας λίστας αριθμών.

Αφού ο χρήστης εισάγει το επιθυμητό βήμα του σπειρώματος σε mm, εκτελείται η ρουτίνα δημουργίας της λιστας αριθμών.Η ρουτίνα αυτή χρειάζεται τις εξής σταθερές: Την ανάλυση των αισθητήρων σε παλμούς ανα περιστροφή, τον λόγο μείωσης στον μειωτήρα του κινητήρα και τη βήμα του κοχλία τυ σεπόρτι. Η μείωση του κινητήρα μπορεί να συμπεριληφθεί στην ανάλυση του αισθητήρα του κινητήρα. Αυτό ισοδυναμεί με την θεώρηση πως ο αισθητήρας βρίσκεται απ'ευθείας στον άξονα του κοχλία του σεπόρτι.Οι πληροφορίες αυτές ορίζονται ως εξής:

```
//system characteristics:
#define SP_PPR 40     //Spindle encoder pulses per rotation
#define LS_PPR 780    //Leadscrew encoder pulses per rotation (30 ppr * 26 times
reduction)
```

```
#define LS_PITCH 3175
```

Το βήμα του κοχλία έχει μετρηθεί ίσο με 3.175 mm. Λόγω της δυσκολίας του μΕ να χειριστεί δεκαδικούς αριθμούς, χρησιμοποιούμε το γινόμενό του επι 1000.

Το κλάσμα που θα επεξεργαστεί ο αλγόριθμος, που είναι και η σχέση που επιθυμούμε να επιτύχουμε μεταξύ των παλμοσειρών των αισθητήρων, είναι:

$$\frac{x}{y} = \frac{pitch}{LS\_PITCH} * \frac{LS\_PPR}{SP\_PPR}$$

Για την αποθήκευση της ακολουθίας των αριθμών έχει δημιουργηθεί μία δομή απλά συνδεδεμένης κυκλικής λίστας. Ο κάθε κόμβος αποτελείται απο μία αριθμητική μεταβλητή 8-bit και έναν δείκτη προς τον επόμενο κόμβο:

```
struct listNode {
uint8_t info;
struct listNode *next;
};
typedef struct listNode list;
```

Στην δομή αυτή αρχικοποιούνται οι μεταβλητές-δείκτες *Ptr και *dummyPtr. Στην θέση που ορίζει η μεταβλητή *dummyPtr δημιουργείται μία υποτυπώδης λίστα που περιλαμβάνει μόνο έναν κόμβο με την τιμή 0, ο οποίος δείχνει στον εαυτό του:

```
dummyPtr = (list*) malloc(sizeof(list));
dummyPtr->info=0;
dummyPtr->next = dummyPtr;
Ptr = dummyPtr;
```

Η λίστα αυτή χρησιμοποιείται στην θέση της κύριας λίστας για να αποφευχθεί τυχόν πρόβλημα κατα τις διαδικασίες κατασκευής και διαγραφής της κύριας λίστας. Ο ενδεχόμενος κίνδυνος έγκειται στο γεγονός οτι η κύρια λίστα διαβάζεται απο ρουτίνα εξυπηρέτησης εξωτερικής διακπής, όπως θα δούμε στην συνέχεια. Οι ρουτίνες αυτές παρεμβάλλονται στην ομαλή εκτέλεση του κώδικα όταν πυροδοτηθούν. Η ρουτίνα κατασκευής της λίστας δέχεται σαν όρισμα το επιθυμητό βήμα του σπειρώματος σε mm*100 και επιστρέφει έναν δείκτη στην λίστα που δημιουργεί.

```
list * makelist( uint16_t pitch){
// pitch = real pitch(mm)*100 (integer)
//makes a list with the series of the desired ls_pulses for each sp_pulse
//the list is circular
   lcd_clrscr();
   lcd_puts_p(PSTR("making list "));
//our initial fraction is x/y:
   uint32_t x = (uint32_t)LS_PPR*pitch*10;  //*10 because pitch is in mm*100 while
LS_PITCH in mm*1000
   uint32_t y = (uint32_t)SP_PPR*LS_PITCH;
//the initial error is entire x
   int32_t bError = x;

   uint8_t count=0;
   uint16_t countx=0;
   uint16_t iterations =0;

   int32_t bestError = bError;  //bestError will hold the least error value so far
   uint16_t bestx;    // bestx and besty are the candidates for the terms of the new
factors
```

```
   uint16_t besty;

   while((bError !=0)&&(iterations < 150)){ //a general memory limit is used

        iterations++;

        while(bError > y/2){          //starting the round() approximation
           bError = bError - y;
           countx++;
        }

        if(bError < bestError){   // we have found a new least error, so we save the
terms of current approximation
            bestError = bError;
            bestx = countx;
            besty = iterations
        }

        bError = bError +x;
   }

// approximation has finished. Either we have a new fraction, or the old one.
// The list is now guaranteed not to exceed the length limit

 bError =bestx;

//we start making the list structure:
   list * listPtr;
   listPtr = (list*) malloc(sizeof(list));

   if (!listPtr ){  //listPtr == NULL >>>> system out of memory. This is a safety
check
            lcd_clrscr();
            lcd_puts_p(PSTR("out of memory"));
            _delay_ms(1000);
            return(dummyPtr);
        }

   list * start = listPtr;

//we are starting to compute the list values, accorfing to the truncate method
//first node is made separately
   while(bError >= besty){
            bError = bError - besty;
            count++;
        }
   listPtr -> info = count;
// now we are sure that the error will become 0.
//remember, we are using the approximation fraction
   while(bError !=0){
        count =0;
//constructing every new node of the list
        listPtr -> next = (list*) malloc (sizeof(list));
        if (!(listPtr -> next)){  //NULL means end of memory again
            lcd_clrscr();
            lcd_puts_p(PSTR("out of memory\nstopped at: "));
            lcd_puts(itoa(iterations,str,10));
            _delay_ms(1000);
            break;
        }


        listPtr = listPtr -> next;
        bError = bError +bestx;

        while(bError >= besty){
            bError = bError - besty;
```

```
            count++;
        }
        listPtr -> info = count;
    }
    // closing the list by connecting head with tail. Now it is circular.
    listPtr->next = start;
    //displaying the list length (debugging value only)
    lcd_clrscr();
    lcd_puts_p(PSTR("List length: "));
    lcd_puts(itoa(besty,str,10));
    //displaying error of approximation in percentage out of 10000.
    lcd_puts_p(PSTR("\nError:"));
    lcd_puts(ltoa((bestError*10000)/(y*besty),str,10));
    lcd_puts_p(PSTR("/10000"));
    _delay_ms(1000);

    return (start);
}
```

Ρουτίνα απελευθέρωσης λίστας.

Οταν ο χρήστης ολοκληρώσει ή τερματίσει την διαδικασία κοπής σπειρώματος, η λίστα δεν είναι πλέον χρήσιμη. Επομένως, ο δεσμευμένος χώρος στην μνήμη πρέπει να απελευθερωθεί. Ετσι, το σύστημα μπορεί να κατασκευάσει εκ νέου μία καινούρια λίστα αν ζητηθεί χωρίς να προκύψει πρόβλημα πόρων.

```
void freelist(){
//set the memory of the list free and poits Ptr to the dummy one-block list

   list * start = Ptr;
   Ptr = dummyPtr;

   list * tempptr;
   uint8_t iter=1;

   while(start->next != start){   //until there is only one node left

       tempptr = start->next->next;
       free(start->next);
       start->next = tempptr;
       iter++;
   }
   free(start);

   lcd_clrscr();
   lcd_puts_p(PSTR("deleted nodes:\n"));
   lcd_puts(utoa(iter,str,10));
   _delay_ms(1000);
   return;
}
```

Ρουτίνες παρακολούθησης αισθητήρων.

Οι παλμοσειρές απο τους δύο βασικούς αισθητήρες του συστήματος αντιμετωπίζονται ως εξωτερικές διακοπές. Ο αισθητήρας του άξονα του τσώκ συνδέεται στον ακροδέκτη int0 και του κινητήρα στον ακροδέκτη int1. Οι ρουτίνες εξυπηρέτησης των δύο διακοπών ενεργοποιούνται στην αρνητική ακμή των παλμών.

```
MCUCR = 0b00001010; // external interrupts triggered at falling edge
GICR = 0b11000000;  // both interupt pins active
```

Η ενεργοποίηση/απενεργοποίηση των διακοπών γενικά πραγματοποιείται με τις μακροεντολές :

```
sei()     //  enable interrupts
cli()      //disable interrupts
```

- *Ρουτίνα εξυπηρέτησης διακοπων για τον αισθητήρα του κινητήρα:*

```
ISR (INT0_vect,ISR_NOBLOCK){
//ls_pulse_event interrupt service routine
        ls_pulses++;       //counting pulses per master axis pulse
        total_ls_pulses++; // for total distance measuring
}
```

Η λειτουργία της έχει περιγραφεί σε προηγούμενη ενότητα. Στην μεταβλητή ls_pulses συσσωρεύεται προσωρινά το πλήθος παλμών που έχουν καταφθάσει ανάμεσα σε δύο παλμούς του άλλου αισθητήρα, ενώ στην μεταβλητή total_ls_pulses συσσωρεύεται το συνολικό πλήθος παλμών. Χρησιμοποιείται για την μέτρηση της μετατόπισης του σεπόρτι.Η μετβλητή sp_index είναι βοηθητική. μπορέι να χρησιμοποιηθεί αντί για τον επιπρόσθετο αισθητήρα για ένδειξη της γωνιακής θέσης του άξονα αναφοράς. επίσης χρησιμοποιείτα για έλεγχο του αισθητήρα μέσω της συνάρτησης test_mode() που θα δούμε στην συνέχεια.

- *Ρουτίνα εξυπηρέτησης διακοπων για τον αισθητήρα του άξονα του τσώκ:*

```
ISR (INT1_vect,ISR_NOBLOCK){
    //sp_pulse_event interrupt service routine

    sp_index++;
    if (sp_index == SP_PPR){sp_index = 0;}  //angle indexer of spindle

    Error += Ptr->info - ls_pulses;         //refreshing the angle error
variable with current angle error
    previous_ls_pulses = ls_pulses;

    ls_pulses = 0;      // initialising the counter to count the next
(ls_pulses per sp_pulse)
    Ptr = Ptr->next;    // poiter to next bresenham calculated number at
the circular list
}
```

Η εκτέλεση των δύο αυτών ρυτίνων γίνεται ανεξάρτητα απο όλο το υπόλοιπο πρόγραμμα σε χρόνους που καθορίζονται απο τους αισθητήρες. Ετσι, η μεταβλητή Error είναι διαρκώς ενημερωμένη με το τρέχον σφάλμα θέσης του συστήματος.

Να σημειώσω οτι στην δήλωση όλων των ρουτίνων των διακοπών έχει προστεθεί το όρισμα ISR_NOBLOCK. Αυτή η παράμετρος επιτρέπει στην ρουτίνα μίας διακοπής να εκτελεστεί ενόσω εκτελείται μία διαφορετική διακοπή. Αυτή η ρύθμιση είναι επικίνδυνη καθώς επιτρέπει το συνεχές φώλιασμα συναρτήσεων (nesting) το οποίο μπορέι να οδηγήσει σε εξάντληση της μνήμης του συστήματος και επομένως επανεκκινηση του μΕ. Ωστόσο, σε αυτήν την περίπτωση έχει μελετηθεί ο χρόνος εκτέλεσης της κάθε διακοπής και η μέγιστη συχνότητα παλμών απο τους αισθητήρες και έχει βρεθέι πως είναι ασφαλής ρύθμιση για το συγκεκριμένο σύστημα.

Ρουτίνα πέδησης του κινητήρα:

Οπως έχουμε αναφέρει, η γέφυρα οδήγησης του κινητήρα παρέχει την δυνατότητα πέδησης του μοτέρ με το να βαρχυκυκλώνονται οι δύο ακροδέκτες του. Η ολοκληρωμένη γέφυρα του συστήματος δέχεται εντολές μέσω τριών ακροδεκτών : του ακροδέκτη Enable και δύο ακροδεκτών για την διεύθυνη, dir1 και dir2. Οι καταστάσεις της γέφυρας είναι οι εξής:

Enable	Dir1	Dir2	κατασταση
1	0	1	Δεξιόστροφη περ.
1	1	0	Αριστερόστροφη
1	0	0	Πέδηση
1	1	1	Πέδηση
0	Αδιάφορο	Αδιάφορο	free running

Οπως βλέπουμε η πέδηση της γέφυρας επιτυγχάνεται με ανάθεση της ίδιας τιμής στους ακροδέκτες dir1 και dir2 της γέφυρας. Η τιμή αυτή καθορίζει ποιό ζεύγος τρανζίστορ θα βραχυκυκλώσουν τους ακροδέκτες του κινητήρα.

Επειδή η ρύθμιση της φοράς περιστροφής, όπως και πολλών άλλων λειτουργιών απαιτεί την ρύθμιση μεμονομένων bit απο κάποιες θύρες, έχει προστεθεί η παρακάτω δομή ωστε να απομονώνονται κάποι bit κάποιων καταχωρητών:

```
typedef struct          {
  unsigned int bit01:2;// this one contains 2 bits (used for direction pins)
  unsigned int bit2:1; // all the rest contain 1 bit per variable ...
  unsigned int bit3:1;
  unsigned int bit4:1;
  unsigned int bit5:1;
  unsigned int bit6:1;
  unsigned int bit7:1;
} bit_reg;
```

Και αυτή η μακροεντολή επιτρέπει την ανάθεση μεταβλητής σε μεμονωμένα bits κάποιου καταχωρητή/θύρας:

```
#define REGISTER_BIT(rg,bt) ((volatile bit_reg*)&rg)->bit##bt
```

Με την χρήση της αποθηκεύουμε την μεταβλητή που περιλαμβάνει τα δύο bits διεύθυνσης:

```
#define DIRECTION REGISTER_BIT(PORTD,01)
```

Ορίζουμε επίσης την τιμή για κάθε κατεύθυνση:

```
#define RIGHT 0b01;
#define LEFT 0b10;
```

Η συνάρτηση πέδησης του κινητήρα φροντίζει ωστε να μην χρησιμοποιείται πάντα το ίδιο ζεύγος τρανζίστορ για να βραχυκυκλώσουν οι ακροδέκτες του κινητήρα, και εισάγει μια καθυστέρηση έως ότου ο κινητήρας σταματήσει εντελώς. Κατα την διάρκεια της καθυστέρησης αυτής το σήμα Enable σταθεροποιείται στο λογικό 1 ωστε να έχουμε μέγιστο

αποτέλεσμα απο την πέδη. Στην συνέχεια επαναφέρουμε την αρχική τιμή των bit διεύθυνσης.

```
void doBrake(){
   if(DIRECTION == LEFT){DIRECTION = 0b00;}else{DIRECTION = 0b11;}
   PORTD |= (1<<5);              //for effective braking
   _delay_ms(500);               //until it is likely to have stopped
   PORTD &= 0b11011111;         //leave the motor switched off
   if(DIRECTION == 0b00){DIRECTION = LEFT;}else{DIRECTION = RIGHT;}
}
```

Ρουτίνα κοπής:

Η ρουτίνα αυτή εκτελείται όταν ο χρήστης δώσει έγκριση για έναρξη ενός κύκλου κοπής. Αρχικά αρχικοποιούνται όλες οι μεταβλητές. Στην συνέχεια το σύστημα χρησιμοποιεί τον απολυτο αισθητήρα αναφοράς, ώστε να εξασφαλιστεί οτι η κοπή θα ξεκινήσει όταν το τεμάχιο υπο επεξεργασία βρίκσεται στην γωνιακή θέση $\phi_{\epsilon\kappa\kappa}$. Στην συνέχεια, η μεταβλητή σφάλματος αρχικοποιείται στην τιμή της προηγούμενης ολίσθησης, αν υπάρχει, ενεργοποιείται ο κινητήρας και ο ελεγκτής. Ο συγχρονισμός του κινητήρα γίνεται απο τις ρουτίνες διακοπών. Η μόνη εργασία της υπορουτίνας πλέον είναι να ελέγχει πότε θα ολοκληρωθεί το συνολικό μήκος κοπής ή άν θα πατηθεί κάποιο πλήκτρο ή διακόπτης διακοπής ωστε να τερματίσει την διαδικασία. Οταν κάτι απο αυτά συμβαίνει φρενάρει ο κινητήρας, απενεργοποιείται ο ελεγκτής και υπολογίζεται η νέα τιμή ολίσθησης. Σε περίπτωση που ο χρήστης δεν έχει δηλώσει μήκος κοπής τότε το επιθυμητό μήκος τίθεται στην τιμή -1, που είναι αδύνατο να επιτευχθεί. Ετσι, το σύστημα σταματάει μόνο με πάτημα του πλήκτρου «οκ».

Ο απόλυτος αισθητήρας του άξονα αναφοράς έχει συνδεθεί στον ακροδέκτη 3 της θύρας D και έχει οριστεί ώς:

```
#define INDEX REGISTER_BIT(PIND,6)    // Spindle index sensor
```

Το σύστημα επίσης περιλαμβάνει έναν διακότη ο οποίος ενεργοποιείται αν το σεπόρτι βρεθεί σε σημείο που ενδεχομένως να χτυπήσει σε κάποιο άλλο στοιχείο του μηχανήματος. Ο διακόπτης αυτός τερματίζει την όποια διαδικασία κίνησης του κινητήρα. Εχει συνδεθεί στον ακροδέκτη 7 της θύρας D και οριστεί ώς:

```
#define LIMIT REGISTER_BIT(PIND,7)    // limit switch Normally 1
```

Ας δούμε τον κώδικα της ρουτίνας:

```
void doCut(){
   total_ls_pulses = 0;
   ls_pulses =0;
   OCR1A = 0;
   pid_st->sumError =0;
   pid_st->lastError =0;
   lcd_clrscr();
   lcd_puts_p(PSTR("waiting for sp"));
   while(!INDEX){}
   while(INDEX){}      // always leaves this point at INDEX falling edge
   Error = slip;
   PID_START();
   PWM_START();
   while((!PRESSED_D)&&(LIMIT)&&((total_ls_pulses < length_pulses + slip
)||(length_pulses == -1))){}
   PID_STOP();
   PWM_STOP();
```

```
        doBrake();
        if ((length_pulses == -1)||(total_ls_pulses < length_pulses + slip)){   //if
length not specified
            length_pulses = total_ls_pulses - slip;                             //or
cut stopped abruptly
            slip = 0;
        }
        else {
            slip = total_ls_pulses - (length_pulses + slip);
        }

        lcd_clrscr();
        lcd_puts_p(PSTR("slip: "));
        lcd_puts(itoa(slip,str,10));
        _delay_ms(1000);
        return;
    }
```

Ρούτινα κίνησης:

Η ρουτίνα αυτή χρησιμοποιείται όταν επιστρέφουμε στος σημείο εκκίνησης μετά απο μία διαδικασία κοπής, ή όταν ζητηθεί εξ'αρχής λειτουργία απλής κίνησης απο τον χρήστη.Η ρουτίνα δέχεται σαν όρισμα το μήκος της επιθυμητής κίνησης. Επίσης μή ορισμός του μήκους σημαίνει χειροκίνητος τερματισμός ή τερματισμός ασφαλείας.η κίνηση γίνεται σε πλήρη ταχύτητα και χωρίς χρήση του ελεγκτή. Λίγο πριν την ολοκλήρωση της κίνησης η ταχύτητα πέφτει σε μία καθορισμένη χαμηλή τιμή, ωστε να περιοριστεί το φαινόμενο της ολίσθησης. μετά την ολοκήρωση της ζητούμενης μετατόπισης ο κινητήρας ακινητοποιείται. Οι δύο τιμές που παίρνει η ταχύτητα έχουν ορισθεί ώς:

```
#define PWM_TOP 0x03FF
#define PWM_SLOW 600
```

Κατα την μετακίνηση, στην οθόνη εμφανίζεται μία μπάρα προόδου. Το ποσοστό της μπάρας που είναι γεμάτο αντιστοιχεί στο ποσοστό της κίνησης που έχει ολοκληρωθεί ανα πάσα στιγμή. Ας δούμε τον κώδικα αναλυτικά:

```
void doMove(int32_t length_puls){

    uint16_t progress_step;             //progress bar division
    uint8_t i=1;                        //progress bar steps
    progress_step = length_puls >> 4;   //length_puls/16,LCD has 16 columns
    lcd_init(LCD_DISP_ON);
    lcd_clrscr();
    lcd_puts_p(PSTR("moving \n----------------\n\n"));

    total_ls_pulses = 0;
    OCR1A = PWM_TOP;            //full speed
    PWM_START();

    while((!PRESSED_D)&&(LIMIT)&&((total_ls_pulses  <  length_puls)||(length_puls
== -1))){
        if ((length_puls > 0)&&((total_ls_pulses - length_puls + 2*LS_PPR == 0
)||(length_puls < 2*LS_PPR))){OCR1A = PWM_SLOW;}
        if      ((length_puls       !=      -1)&&(total_ls_pulses       >
(progress_step*i))){lcd_putc(0xFF);i++;} //progress bar update
    }

    PWM_STOP();
    doBrake();

    lcd_init(LCD_DISP_ON_CURSOR_BLINK);
```

```
    slip = total_ls_pulses - length_puls;
    if (length_puls == -1){slip =0;}
    return;
}
```

Ρουτίνα καταγραφής βηματικής απόκρισης:

Η ρουτίνα αυτή δεν έχει λειτουργική σημασία για το σύστημα. Χρησιμοποιείται για την καταγραφή της βηματική απόκρισης του συστήματος κίνησης, όσον αφορά την ταχύτητα και την αποθήκευσή της στην μνήμη EEPROM. Τα δεδομένα αυτά μπορούν να χρησιμοποιηθούν για αναγνώριση της συνάρτησης μεταφοράς του συστήματος με σκοπό την ακριβή ρύθμιση του ελεγκτή PID.

Η ρουτίνα χρησιμοποιεί τον χρονιστή timer2 και μία ρουτίνα διακοπών για μέτρηση χρόνου. Τροφοδοτεί τον κινητήρα με πλήρη ισχύ και καταγράφει τον χρόνο ανάμεσα στους παλμούς απο τον αισθητήρα του.Γίνεται προσαρμογή του χρονιστή ωστε η καταγραφή να έχει όσο το δυνατόν καλύτερη ακρίβεια. Συνολικά καταγράφονται 400 τιμές σε έναν πίνακα. Στην συνέχεια αυτός ο πίνακας αντιγράφεται στην μνήμη EEPROM.

Η ρύθμιση του χρονιστή γίνεται ώς εξής:

```
    TCCR2 = 0b00001011;  //prescaler: clk/32 ,CTC mode
    TIMSK |= (1 << 7);   //compare match interrupt enable
```

Ο χρονιστής πλέον μετράει μέχρι τον αριθμό που βρίσκεται στον καταχωρητή OCR2. Οταν φτάσει αυτή η τιμή εγείρεται μία διακοπή, ο χρονιστής μηδενίζει και η διαδικασία συνεχίζεται. Η ρουτίνα της διακοπής είναι η εξής:

```
ISR (TIMER2_COMP_vect, ISR_NOBLOCK){
    if (periodcount == 0xFF){overflow =1;}
    periodcount++;
}
```

Η καθολική μεταβλητή χρονομετράει την περίοδο σε μονάδες OCR2*prescaler/ClockFrequency. Η τιμή του OCR2 ρυθμίζεται ώς την ελάχιστη τιμή που η μεταβλητή periodcount δεν παρουσιάζει υπερχείλιση. Ετσι επιτυγχάνουμε την μέγιστη ακρίβεια στις μετρήσεις.Η τιμή του OCR2 που χρησιμοποιήθηκε καταγράφεται επίσης στην EEPROM.Ας δούμε το σώμα της ρουτίνας:

```
void tuning_mode(){

    uint8_t data[400];
    uint16_t address =0;
    sei();

    lcd_clrscr();
    lcd_puts_p(PSTR("tuning"));
    OCR2 = 1;      //compare number

  while (address <400){

    address =0;
    periodcount=0;
    total_ls_pulses=0;

    DIRECTION = LEFT;
    OCR1A = PWM_TOP;     //top speed
    PWM_START();

    while(address < 400){
       if(total_ls_pulses != address){
```

```
            data[address] = periodcount;
            periodcount = 0;
            address = total_ls_pulses;
         }
         if(overflow){break;}
      }
      PWM_STOP(); //stop motor
      TCCR2 =0;   //stop timer2
      if(overflow){
         lcd_clrscr();
         lcd_puts_p(PSTR("overflow\n press OK"));
         overflow = 0;
         while(!PRESSED_D){}
         DEBOUNCE();
         OCR2++;
      }
    }
      lcd_clrscr();
      lcd_puts_p(PSTR("copying \nto EEPROM..."));
      eeprom_write_byte ((uint8_t*)18, (uint8_t)(OCR2)); //the timer compare value
      for (int i = 0;i<400;i++){
         eeprom_write_byte ((uint8_t*)i+20, (uint8_t)(data[i]));//from RAM->EEPROM
      }
      lcd_clrscr();
      lcd_puts_p(PSTR("tuning complete"));
      while(KEYPAD_PIN & 0b00000001){}
      DEBOUNCE();
return;
}
```

Ρουτίνα εμφάνισης ρυθμίσεων:

Η ρουτίνα αυτή εμφανίζει τις ρυθμίσεις του ελεγκτή PID που βρίσκονται στην μνήμη EEPROM κατα τηνεκκίνηση του συστήματος. Ετσι, ο χρήστης μπορέι να επιβεβαιώσει οτι δεν έχει προκύψει κάποια αλλαγή ή βλάβη στα δεδομένα.

```
void show_settings(){
   lcd_clrscr();
   lcd_puts_p(PSTR("p:"));
   lcd_puts(utoa(pid_st->P_Factor,str,10));
   lcd_puts_p(PSTR("i:"));
   lcd_puts(utoa(pid_st->I_Factor,str,10));
   lcd_puts_p(PSTR("d:"));
   lcd_puts(utoa(pid_st->D_Factor,str,10));
   lcd_puts_p(PSTR("\ntime intrvl:"));
   lcd_puts(itoa(OCR0,str,10));

   _delay_ms(2000);
return;
}
```

Ρουτίνα ελέγχου αισθητήρων:

Η ρουτίνα αυτή εμφανίζει στην οθόνη τις μεταβλητές των αισθητήρων οι οποίες διαμορφώνονται σε πραγματικό χρόνο.Ετσι, ο χρήστης μπορέι να επιβεβαιώσει αν λειτουργουν σωστά και είναι σωστα συνδεδεμένοι.

```
void test_mode(){
sei();
while(!PRESSED_D){   //while button is not pressed
lcd_clrscr();
lcd_puts_p(PSTR("ls:"));
```

```
lcd_puts(itoa(total_ls_pulses,str,10));
lcd_puts_p(PSTR(" ls/sp:"));
lcd_puts(itoa(previous_ls_pulses,str,10));
lcd_puts_p(PSTR("\nsp index:"));
lcd_puts(itoa(sp_index,str,10));
if(sp_index >9){lcd_puts_p(PSTR(" "));}
else{lcd_puts_p(PSTR("  "));}
lcd_puts(utoa(INDEX,str,10));
_delay_ms(50);   //display refresh
}
DEBOUNCE();
return;
}
```

Σε αυτό το σημείο έχουμε αναφέρει όλες τις υπορουτίνες. Ας δούμε τώρα την κεντρική ρουτίνα του συστήματος:

Η κεντρική ρουτίνα main()

Η ρουτίνα main() είναι η καρδιά του συστήματος. Οπως είπαμε, αναλαμβάνει την διεπαφή με τον χρήστη και καλεί την κατάλληλη υπορουτίνα για την κάθε εργασία. Κατα την ανάλυση της δομής της θα δούμε τις καταστάσεις του συστήματος (που αναλύθηκαν προηγουμένως) πιο αναλυτικά.

Εκκίνηση του συστήματος: Αν κατα τη εκκίνηση κάποιο πλήκτρο είναι πατημένο, η main() καλεί κάποια απο τις συναρτήσεις ελέγχου ή ρύθμισης παραμέτρων.

Στην συνέχεια εμφανίζεται στην οθόνη ένα μήνυμα καλωσορίσματος και ένα άλλο μήνυμα που αναφέρει τις ρυθμίσεις του συστήματος.

Βήμα 0: Ο χρήστης επιλέγει την εργασία που επιθυμεί

Ας ξεκινήσουμε με την περίπτωση της επιλογής κοπής.

Βήμα 1: Ο χρήστης εισάγει το επιθυμητό βήμα, ή κάποια απο τις δύο έτοιμες επιλογές. Η main() καλεί την ρουτίνα makebresenham() η οποία δέχεται σαν όρισμα το επιθυμητό βήμα και κατασκευάζει την κυκλική λίστα αριθμών. Η λίστα επιστρέφεται σε μία καθολική μεταβλητή-δείκτη

Βήμα 2: Ο χρήστης επιλέγει την φορά της κίνησης. Η επιλογή αυτή καθορίζει την φορά περιστροφής του κινητήρα, μέσω των εισόδων dir1 και dir2 της γέφυρας οδήγησης.

Βήμα 3:Ο χρήστης δηλώνει το επιθυμητό μήκος κοπής σε mm. Καθώς επιτρέπονται μέχρι 2 δεκαδικά ψηφία,ο αριθμός πολλαπλασιάζεται επι 100 ωστε η τιμή να είναι ακέραια. Για να μετατραπεί η τιμή αυτή σε πλήθος παλμών, γίνεται η ακόλουθη πράξη:

$$\text{παλμοί} = \frac{\left(\frac{\mu\acute{\eta}\kappa o\varsigma}{100}\right) * \alpha\nu\acute{\alpha}\lambda\upsilon\sigma\eta\ \alpha\iota\sigma\theta\eta\tau\acute{\eta}\rho\alpha\ s}{\beta\eta\mu\alpha\ \kappa o\chi\lambda\acute{\iota}\alpha\ \sigma\epsilon\pi\acute{o}\rho\tau\iota}$$

, όπου στην ανάλυση του αισθητήρα έχει συμπεριληφθεί και η μείωση του κινητήρα.Αν δεν δηλωθεί επιθυμητό μήκος, το πλήθος παλμών τίθεται στην τιμή -1.

Βήμα 4: το σύστημα αναμένει το πάτημα του πλήκτρου «D» ωστε να συνεχίσει ή του «C» ωστε να επιστρέψει στην αρχική κατάσταση

Βήμα 5:Η main() καλεί την συνάρτηση *doCut* η οποία εκτελείται όσο διαρκεί η κοπή του σπειρώματος

Βήμα 6: Και πάλι το πλήκτρο «D» σημαίνει τηνσυνέχεια ενώ το πλήκτρο «C» την επιστροφή στην αρχική κατάσταση.

Βήμα 7:η main() αντιστρέφει την φορά περιστροφής του κινητήρα και καλεί την συνάρτηση *doMove* η οποία εκτελείται όσο ο κινητήρας επιστρέφει προς την αρχική του θέση.

Βήμα 8: το πρόγραμμα επιστρέφει στο βήμα 4

Ας δούμε την περίπτωση της επιλογής απλής κίνησης:

Βήμα 1: ομοίως με το βήμα 3 ης προηγούμενης περίπτωσης

Βήμα 2: ομοίως με το βήμα 2 της προηγούμενης περίπτωσης

Βήμα 3: ομοίως με το βήμα 4 της προηγούμενης περίπτωσης

Βήμα 4: καλείται η συνάρτηση doMove με όρισμα το επιθυμητό μήκος σε παλμούς.

Βήμα 5: Το πρόγραμμα επιστρέφει στο βήμα 0.

Σε αυτό το σημείο θα δούμε τον κώδικα της κεντρικής συνάρτησης. Κάποιες αρχικοποιήσεις καταχωρητών που αφορούν συγκεκριμένες λειτουργίες θα παραλείφθούν καθώς έχουν αναφερθει στις σχετικές υπορουτίνες.

```
int main() {
   if(PRESSED_D){  // if turned on with button OK pressed
      configuration();               // go to configuration mode
   }
   if(PRESSED_B){  //if turned on with button B pressed
      tuning_mode();                 //go to tuning mode
   }
   if(PRESSED_C){  // if turned on with button C presses
      test_mode();                // go to test mode
   }

   //variables:
   uint16_t pitch;    //desired thread pitch
   uint16_t length;   //desired length of cut in mm * 100
   uint32_t temp;     //just to manage with big numbers calculations
   uint8_t selection;      //type of cut selection

   show_settings();

   lcd_clrscr();
   lcd_puts_p(PSTR("  E-LEADSCREW   \nkostas Karouzos"));
   _delay_ms(1000);

   // main menu loop:
   while(1){
          DEBOUNCE();         //debouncing, if MAIN MENU pressed
          selection = 0;

          lcd_clrscr();
          lcd_puts_p(PSTR("  Select task   \n1: cut  2: move"));
          if (read_key() == 2){              //move selected

                lcd_clrscr();
                lcd_puts_p(PSTR("give length(mm):\n"));
                length = read_numb();    //returns length in mm*100
```

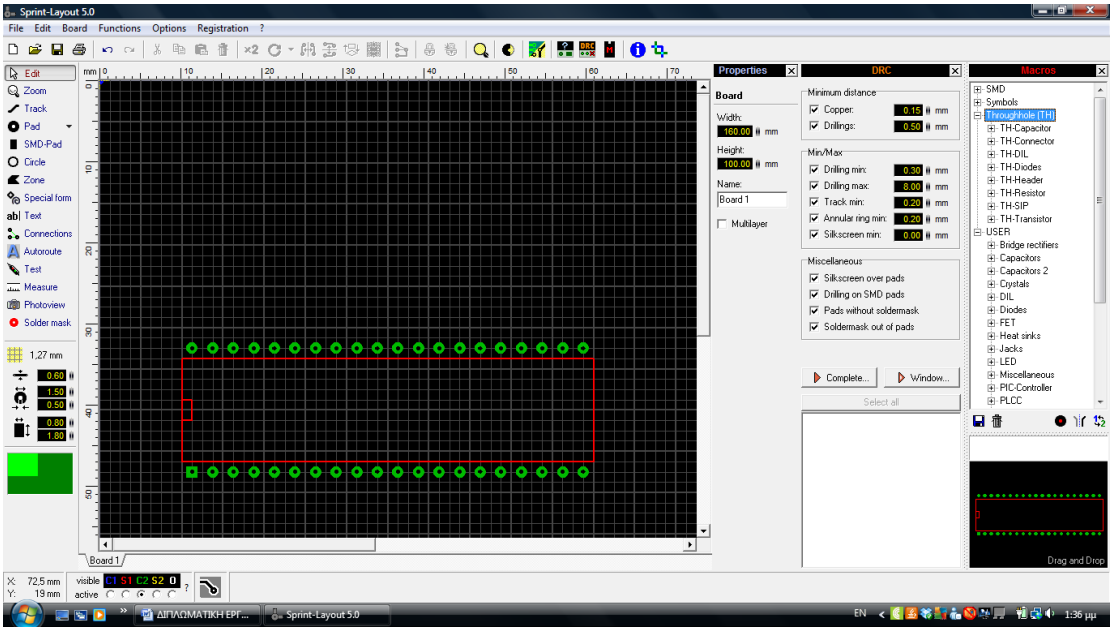
```
                    temp = (uint32_t)LS_PPR*length*10;
                    temp = temp/LS_PITCH;
                    length_pulses = temp;
                    if (length_pulses == 0){length_pulses = -1;} //if length not
specified, dummy value -1

                    lcd_clrscr();
                    lcd_puts_p(PSTR("1:left 2:right\n"));
                    if (read_key() == 2){DIRECTION = RIGHT;}
                    else{DIRECTION = LEFT;}
                    sei();        //enable interrupts
                    lcd_clrscr();
                    lcd_puts_p(PSTR("press OK to move"));
                    while(!PRESSED_D && !PRESSED_C){}  //wait for OK button or MAIN MENU
                    if (!PRESSED_C){                          // if not MAIN MENU,move
                         DEBOUNCE();
                         doMove(length_pulses);
                    }
               }
               else {      // cut selected
                    lcd_clrscr();
                    lcd_puts_p(PSTR("1:coarse 2:fine\n3:thread"));
                    selection = read_key();
                    if (selection == 3){
                        lcd_clrscr();
                        lcd_puts_p(PSTR("give pitch(mm):\n"));
                        pitch = read_numb();
                    }
                    else if (selection == 2){               // polishing, pitch = 0.04 mm
                        pitch = 4;
                    }
                    else{
                        pitch = 8;                     // facing, pitch = 0.08 mm
                    }
                    Ptr = makeBresenhamModel(pitch);

                    lcd_clrscr();
                    lcd_puts_p(PSTR("1:Right handed \n2:Left handed"));
                    if (read_key() == 2){DIRECTION = RIGHT;}
                    else{DIRECTION = LEFT;}

                    lcd_clrscr();
                    lcd_puts_p(PSTR("give length(mm):\n"));
                    length = read_numb();

                    temp = (uint32_t)LS_PPR*length*10;
                    temp = temp/LS_PITCH;
                    length_pulses = temp;
                    if (length_pulses == 0){length_pulses = -1;}

                    sei();
                    slip =0;

                    while(1){        // cutting loop. back and forth
                        lcd_clrscr();
                        lcd_puts_p(PSTR("press OK to cut"));
                        while(!PRESSED_D && !PRESSED_C){}
                        if (PRESSED_C){break;}   //main menu pressed
                        DEBOUNCE();
                        doCut();
                        lcd_clrscr();
                        lcd_puts_p(PSTR("OK to retract"));
                        while(!PRESSED_D && !PRESSED_C){}
                        if (PRESSED_C){break;}
                        DEBOUNCE();
                        DIRECTION = ~DIRECTION;
                        doMove(length_pulses + slip);
                        DIRECTION = ~DIRECTION;
                    }// cutting-retracting loop recycle point
                    freelist();  //finished a cut, freeing memory space
               }//task selection procedure
          } // main menu loop
          return 0;
}
```

Κεφάλαιο 3: Κατασκευή

3.1 Κατασκευή πλακέτας

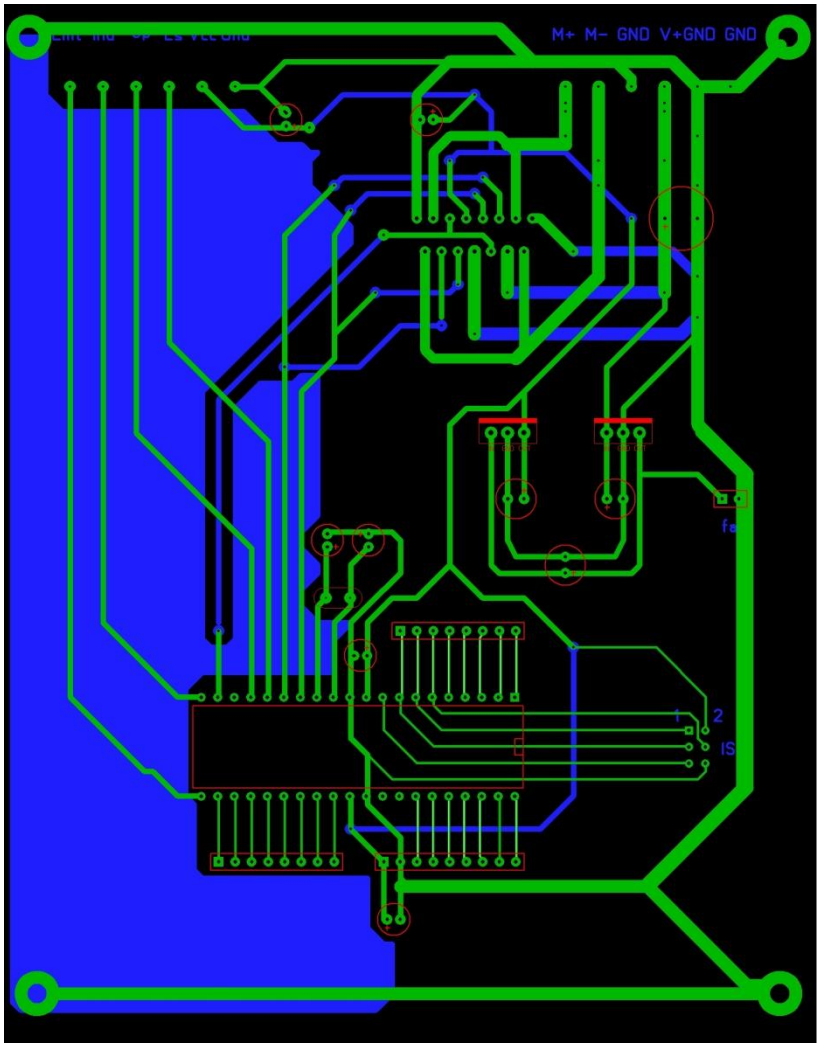
Το κύκλωμα του συστήματυος σχεδιάστηκε με την βοήθεια του προγράμματος Sprint-Layout και τυπώθηκε σε πλακέτα διπλής όψεως . Τα στοιχεία που χρησιμοποιήθηκαν είναι στοιχεία με μακρείς ακροδέκτες που διαπερνούν την πλακέτα μέσω οπών.

Το περιβάλλον του προγράμματος διαθέτει εργαλεία για τον σχεδιασμό των διάφορων μερών ενός κυκλώματος και βιβλίοθήκες με τα αποτυπώματα (footprints) των περισσότερων στοιχείων και συσκευασιών. Μπορούν να σχεδιαστούν μέχρι και 4 στρώματα σε κάθε πλακέτα, καθώς και ένα στρώμα εκτύπωσης. Διαθέτει λειτουργία αυτόματης διασύνδεσης ακροδεκτών και ρύθμισης παραμέτρων για διάνοιξη τρυπών απο αυτόματο μηχάνημα.



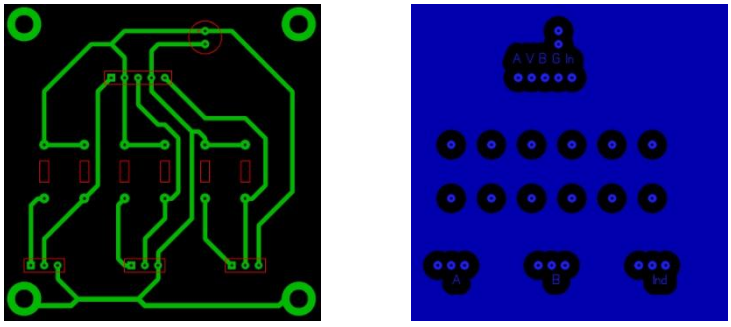
Σχήμα 3.1 Screenshot προγράμματος Sprint-Layout

Παρακάτω φαίνονται τα κυκλώματα της πλακέτα του συστήματος καθώς και της πλακέτας οδήγησης των αισθητήρων. Με πράσινο χρώμα εμφανίζεται το κύκλωμα της πλευράς κυκλώματος της πλακέτας και με μπλέ της πλευράς στοιχείων:



Σχήμα 3.2

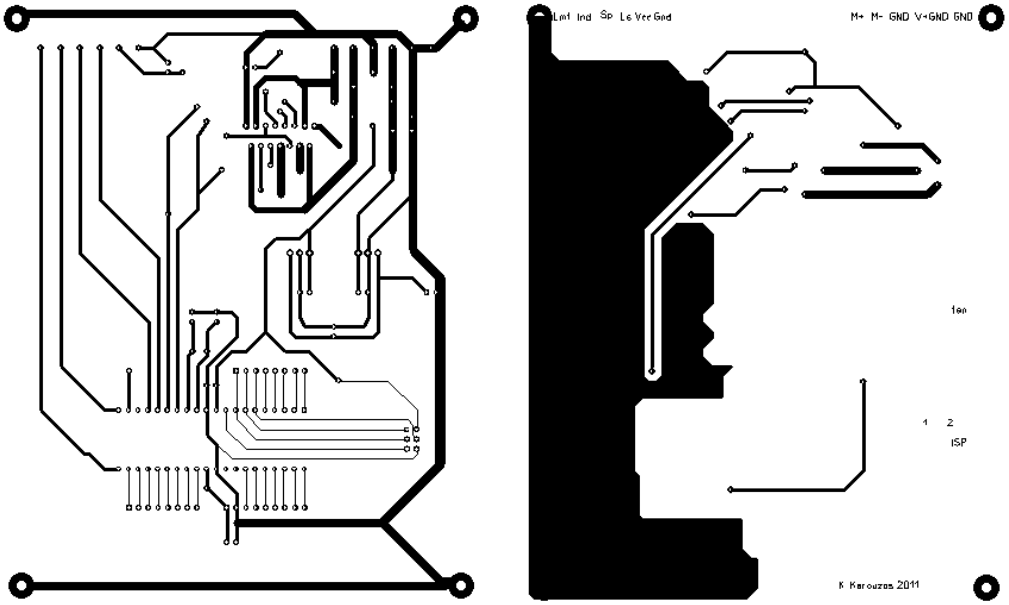
Και η πλακέτα των σισθητήρων:



Σχήμα 3.3

Η πλακέτα που χρησιμοποιήθηκε για την τύπωση του κυκλώματος είναι απο πλεξι-γκλάς, διπλής όψεως και με επικάλυψη φωτοευαίσθητης επίστρωσης στις επιφάνειες χαλκού.
Η διαδικασία που ακολουθήθηκε είναι η εξής:

Αρχικά τυπώθηκε η κάθε πλευρά της πλακέτας ξεχωριστά με μελάνι σε διαφάνειες:



Σχήμα 3.4

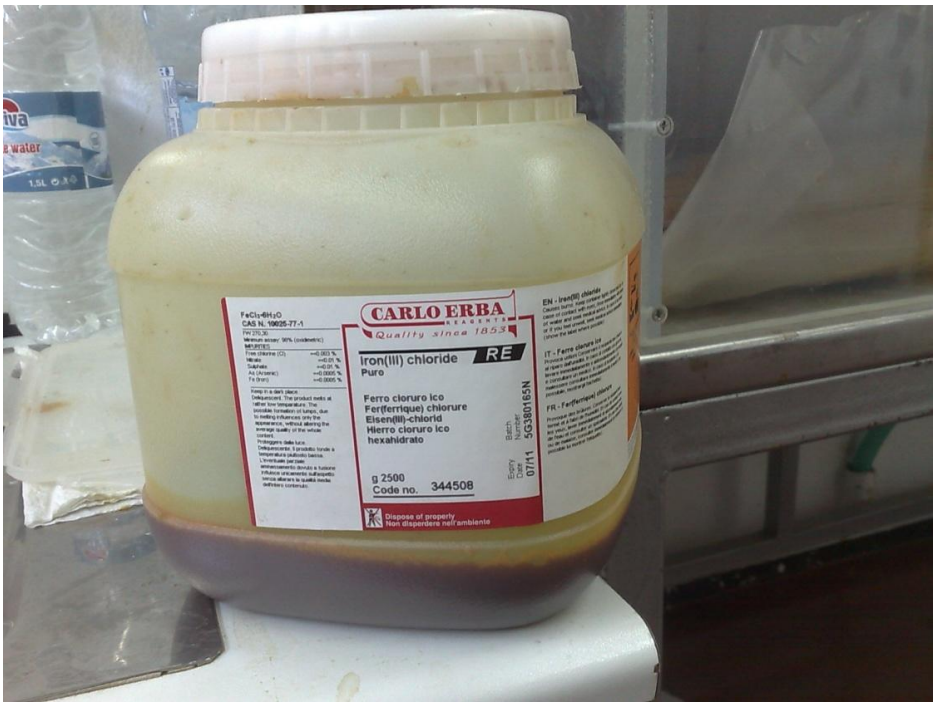
Στην συνέχεια οι διαφάνειες στερεώθηκαν στις αντίστοιχες πλευρές μίας κενής πλακέτας και στην συνέχεια μπήκαν σε μία συσκευή όπου εκτέθηκαν σε υπεριώδη ακτινοβολία και απο τις δύο πλευρές.

Σχήμα 3.5

Η φωτοευαίσθητη επιφάνεια στις πλευρές της πλακέτας αποικοδομείται κατα την έκθεσή της σε υπεριώδη ακτινοβολία. Η ακτινοβολία προσβάλει τα σημεία εκείνα που δεν

προστατεύονται απο το μελάνι. Για το σωστό αποτέλεσμα η πλακέτα χρειάστηκε να παραμείνει στην συσκευή για περίπου 5-6 λεπτά απο κάθε πλευρά.

Στην συνέχεια οι διαφάνειες απορρίπτονται και η εκτεθειμένη πλακέτα τοποθετείται σε ένα δοχείο με διάλυμμα $FeCl_3$. Το διάλυμμα αυτό διαλύει την επίστρωση χαλκού της πλακέτας στα σημεία τα οποία δεν προστατεύονται απο φωτοευαίσθητο υλικό. Η διαδικασία αυτή επιταχύνεται απο μηχανισμούς θέρμανσης και κυκλοφορίας του διαλύμματος μέσα στο δοχείο.



Σχήμα 3.6

Σχήμα 3.7

Λόγω των βλαβερών ανανθυμιάσεων που εκλύονται απο αυτήν την διαδικασία, η συσκευή διαθέτει σύστημα εξαερισμού.

Σχήμα 3.8

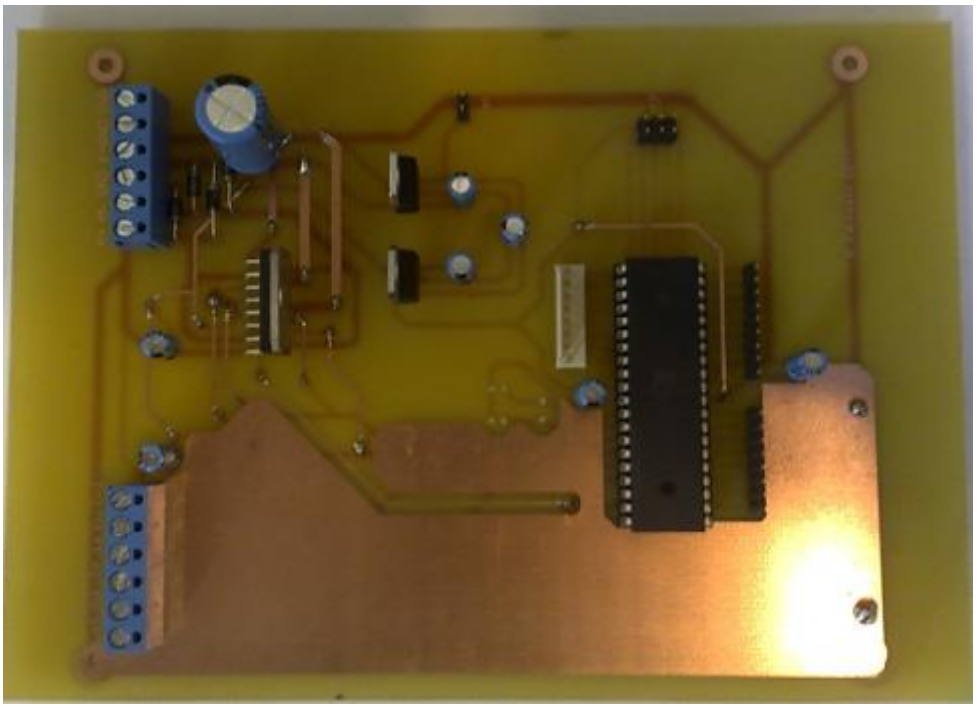
Υστερα απο αυτήν την διαδικασία η πλακέτα διατηρεί την χάλκινη επίστρωσή της μόνο στα σημεία που αντιστοιχούν στις τυπωμένες περιοχές του αρχικής διαφάνειας. Επομένως, έχουμε πλέον σχεδιάσει το αρχικό κύκλωμα με τον χαλκό στις πλευρές της πλακέτας. Το φωτοευαίσθητο υλικό που έχει παραμείνει επάνω στις επιφάνειες του χαλκού αφαιρείται με την χρήση διαλύμματος ακετόνης.

Η πλακέτα πλέον είναι έτοιμη για την διάνοιξη των οπών για την κόλληση των στοιχείων και την επικοινωνία των δύο πλευρών. Οι οπές διανοίχτηκαν με την χρήση τρυπανιού διαμέτρου 0,8 και 1 mm, ανάλογα με την περίπτωση.

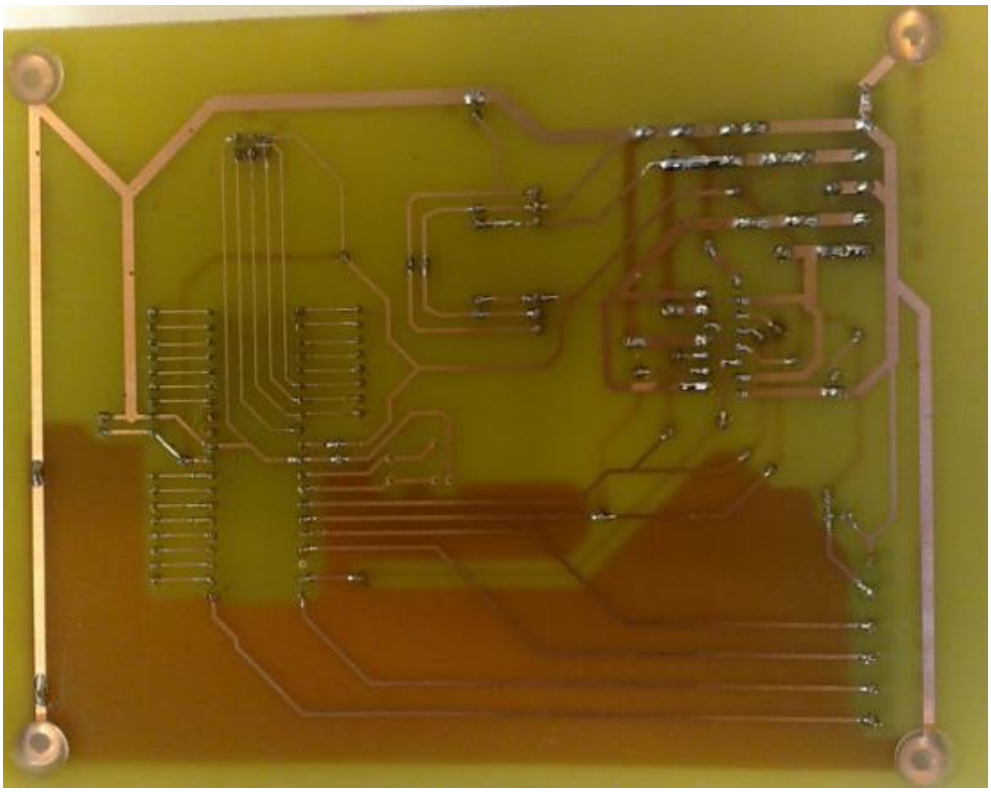
Εν συνεχεία, τοποθετήθηκαν τα στοιχεία και έγιναν οι κολλήσεις με την χρήση κράμματος κόλλησης που περιέχει 60%κασσίτερο και 40%μόλυβδο. Οι κολλήσεις έγιναν με την βοήθεια θερμικού εργαλείου.

Σχήμα 3.9

Η τελική μορφή των πλακετών είναι η εξής:

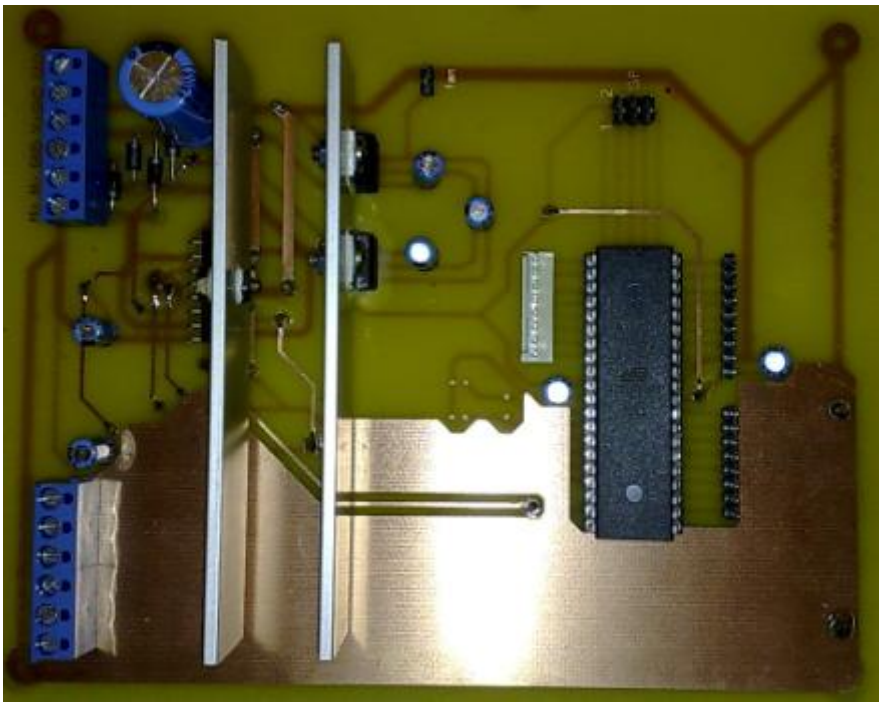


Σχήμα 3.10



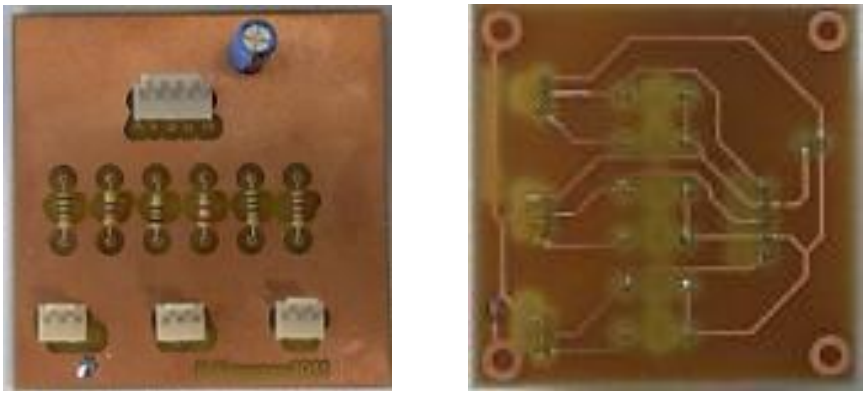
Σχήμα 3.11

Στην συνέχεια, προστέθηκαν ψύκτρες στα στοιχεία ισχύος.Τα στοιχεία αυτά είναι οι σταθεροποιητές τάσης και η γέφυρα οδήγησης του κινητήρα. Οι σταθεροποιητές έχουν τοποθετηθεί σε κοινή ψύκτρα.



Σχήμα 3.12

Οι πλακέτα των αισθητήρων διαμορφώθηκε ώς εξής:



Σχήμα 3.13

Η έξοδος ισχύος της πλακέτας για την τροφοδότηση του κινητήρα περιέχει υψίσυχνες αρμονικές λόγω του PWM σήματος. για τον λόγο αυτό, έχει χρησιμοποιηθεί ένα βαθυπερατό φίλτρο της εταιρείας TOKIN:

Σχήμα 3.14

3.2 Ανάλυση λειτουργίας πλακέτας

3.2.1 Ανάλυση θορυβου - παρεμβολών.

Οι παρεμβολές που ενδεχομένως προκύπτουν στο κύκλωμά μας είναι δύο ειδών: ηλεκτρομανητικές παρεμβολές (EMI) και cross-talk. Ο θόρυβος είναι κυρίως θερμικός.

Ηλεκτρομαγνητικές παρεμβολές:

Κάθε αγωγός που διαρέεται απο μή σταθερό ρεύμα εκπέμπει ένα ηλεκτρομαγνητικό κύμα. Η ενέργεια του κύματος αυτού είναι αναλογη της έντασης του ρεύματος του αγωγού. Το ηλεκτρομαγνητικό κύμα αλληλεπιδρά με γειτονικούς αγωγούς επάγοντας ρεύματα σε αυτούς,τα οποία εμφανίζονται ως παρεμολές.

Στο σύστημα μας ο κινητήρας διαρρέεται απο ρεύμα έντασης μέχρι 4 Amper και η τάση τροφοδοσίας του χαρακτηρίζεται απο ιδιαίτερα υψίσυχνες αρμονικές λόγω της οδήγησης με PWM. Επομένως, αποτελεί μία σημαντική πηγή ηλεκτρομαγνητικών παρεμβολών. Το κεντρικό τροφοδοτικό του συστήματος περιέχει μετασχηματιστές και διακοπτικά στοιχεία που επίσης εκπέμπουν ακτινοβολία. Το κεντρικό μοτέρ του μηχανήματος αποτελεί άλλη μία πηγή παρμεβολών, κυρίως λόγω της μεγάλης ισχύος του.

Τα καλώδια του συστήματος λειτουργούν ώς κεραίες που συλλέγουν την ηλεκτρομαγνητική ακτινοβολία και έτσι υπάρχει κίνδυνος να δημιουργηθούν παρεμβολές στα σήματα που μεταφέρουν. Ιδιαίτερα ευαίσθητα είναι τα καλώδια των αισθητήρων, καθώς διανύουν μεγάλη απόσταση κοντά σε πηγές παρεμβολών και μεταφέρουν σήματα ιδιαίτερης σημασίας.

Για μείωση του φαινομένου παρεμβολών τα καλώδια αυτά έχουν προστατευτεί απο αγώγιμο περίβλημα το οποίο είναι συνδεδεμένο στην γείωση. Ετσι, φορτία που επάγονται σε αυτό απο την ηλεκτρομαγνητική ακτινοβολία οδηγούνται στην γή απ’ευθείας. Για τον ίδιο λόγο, στην απέναντι πλευρά απο τα σημεία με δρόμους που μεταφέρουν σήματα στην πλακέτα έχει τοποθετηθεί γειωμένη επιφάνεια χαλκού.

Cross-talk

Δυο αγωγοί που εκτείνονται παράλληλοι ο ένας στον άλλον λειτουργούν ώς αγώγιμες επιφάνειες πυκνωτή. Ετσι, το φορτίο του ενός αγωγού μπορεί να προκαλέσει αντίθετη φόρτηση στον άλλον, η οποία εκδηλώνεται ώς παρεμβολή. Το φαινόμενο αυτό εντείνεται με την μείωση της απόστασής τους, την αύξηση του μήκους τους και την μείωση του πλάτους του κάθε δρόμου. Η ένταση του φαινομένου επίσης μειώνεται αν υπάρχει αγώγιμο επίπεδο στην απέναντι πλευρά της πλακέτας.

Στη υλοποίηση της πλακέτας του συστήματος έχει καταβληθεί προσπάθεια ωστε οι αγωγοί που μεταφέρουν ευαίσθητα σήματα να είναι όσο το δυνατόν απομακρυσμένοι μεταξύ τους. Στις περιπτώσεις που κάποιοι αγωγοί τέμνονται (οπου φυσικά έχουν χρησιμοποιηθεί και οι δύο πλευρές της πλακέτας ωστε να μην υπαρχει ηλεκτρική επαφή), έχει φροντιστεί ωστε οι διευθύνσεις τους να είναι όσο το δυνατόν κάθετες.

Θερμικός θόρυβος.

Θερμικό θόρυβο ονομάζουμε τον θόρυβο που εισάγεται στα σήματα λόγω των μεταβολών στην λειτουργία των ημιαγωγών καθώς μεταβάλλεται η θερμοκρασία τους. Το φαινόμενο αυτό έχει αντιμετωπιστεί στην συσκευασία της πλακέτας με σωστή ψύξη και κυκλοφορία του αέρα.Ανεμιστήρας έχει τοποθετηθεί ωστε να εισάγεται αέρας απο το εξωτερικό περιβάλλον στην συσκευασία και , αφού αλληλεπιδράσει με τις ψύκτρες να εξέρχεται απο αυτήν.

Φιλτράρισμα γραμμών τροφοδοσίας.

Τα λογικά κυκλώματα και ιδιαίτερα ο μΕ απορροφούν ισχύ απο την τροφοδοσία με μή συνεχή τρόπο. Συγκεκριμένα, αντλούν ρεύμα σε συχνότητες κοντά στην συχνότητα του ρολογιού και σε διακριτές στιγμές. Το γεγονός αυτό εισάγει υψίσυχνες συνιστώσες στις γραμμές τροφοδοσίας τους. Οι συνιστώσες αυτές είναι επικίνδυνες και για τα άλλα λογικά κυκλώματα με την ίδια τροφοδοσία. Γι'αυτόν τον λόγο έχουν τοποθετηθεί κεραμικοί by-pass πυκνωτές κοντά σε όλα τα λογικά κυκλώματα της πλακέτας με σκοπό να βραχυκυκλώνουν και να απορροφούν όποια υψίσυχνη συνιστώσα βρίσκεται στην γραμμή τροφοδοσίας τους.Η χωρητικότητα αυτών των πυκνωτών είναι 0.1 και 1 μF.

Οι γραμμές τροφοδοσίας έχουν φιλτραριστεί τόσο απο κεραμικούς by-pass πυκνωτές, όσο και απο μεγαλύτερους ηλεκτρολυτικούς πυκνωτές οι οποίοι δρούν ώς προσωρινές αποθήκες ισχύος για την ικανοποίηση ενδεχόμενης στιγμιαία υψηλής ζήτησης. Ειδικά η τροφοδοσία των 24 volt, επειδή τροφοδοτεί τόσο την οδήγηση του κινητήρα, όσο και τους σταθεροποιητές των χαμηλότερων τάσεων, υποβοηθάται απο έναν μεγάλο ηλεκτρολυτικό πυκνωτή 1000μF.

3.2.2 Ανάλυση ισχύος- ψύξης.

Τα στοιχεία ισχύος στο σύστημά μας είναι τα εξής: Οι σταθεροποιητές τάσης και η γέφυρα οδήγησης του κινητήρα. Θα μελετήσουμε την θερμική συμπεριφορά τους και θα εξετάσουμε τις ανάγκες ψύξης.

Σταθεροποιητές τάσης:

Οι γραμμικοί σταθεροποιητές τάσης όπως έχουμε αναλύσει καταναλώνουν σημαντικά ποσά ενέργειας μετατρέποντάς την σε θερμότητα. Η ισχύς των απωλειών αυτών είναι το γινόμενο της πτώσης τάσης μέσα στον σταθεροποιητή επι το ρεύμα που τον διαρρέει. Ας μελετήσουμε τα μεγέθη αυτά:

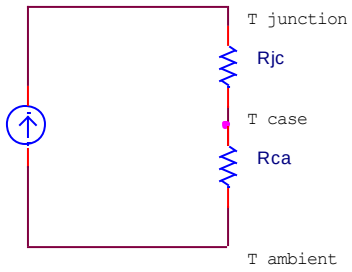
Σταθεροποιητής 5V:
Ο σταθεροποιητής αυτός τροφοδοτεί τα εξής στοιχεία:
μΕ, οθόνη, αισθητήρες και κύκλωμα οδήγησής τους, λογικό κύκλωμα γέφυρας κινητήρα
Πτώση τάσης : 12(τάση εισόδου) – 5(τάση εξόδου) =7V.
Υπολογισμένο ρεύμα περίπου 160mA
Καταναλισκόμενη ισχύς: 7*0,16 = 1,12 Watt
Σταθεροποιητής 12V:
Πτώση τάσης: 24(είσοδος) – 12(τάση εξόδου) = 12V
Ρεύμα: το ρεύμα που διαρρέει τον σταθεροποιητή 5V (αφού είναι σε σειρά συνδεδεμένος) + το ρεύμα του ανεμιστήρα που τροφοδοτείται απο τα 12V (120 mA) = 280mA
Καταναλισκόμενη ισχύς: 12*0,28 = 3,36 Watt

Συνολικά οι σταθεροποιητές τάσης παράγουν 4,48 Watt θερμότητος. Επομένως η χρήση ψύκτρας είναι απαραίτητη, αφού η επιφάνεια των ολοκληρωμένων δεν είναι αρκετή για την απαγωγή τόσο μεγάλων ποσών θερμότητας. Συγκεκριμένα, η συσκευασία των σταθεροποιητών μπορεί να απάγει περίπου 1 watt θερμότητας χωρίς τεχνητή ψύξη, σύμφωνα με τις προδιαφραφές τους.

Ας μελετήσουμε το σύστημα ψύξης που χρειάζεται ένα ολοκληρωμένο. Η ανάλυση του θερμικού συστήματος θα γίνει στο ανάλογο ηλεκτρικό σύστημα, με τις αναλογίες:

θερμότητα ~ ρεύμα
Θερμοκρασία ~ τάση

Η πηγή θερμότητας αναπαρίσταται απο μία πηγή ρεύματος και τα διάφορα στοιχεία μετάδοσης θερμότητας απο (θερμικές) αντιστάσεις. Η θερμική αντίσταση μετράται σε μονάδες °C/W . Η τιμή της τάσης σε κάθε σημείο του κυκλώματος αναλογεί στην θερμοκρασία του αντίστοιχου σημείου. Σε περίπτωση που δεν χρησιμοποιήσουμε ψύκτρα, οι αντιστάσεις του κυκλώματος είναι 2:

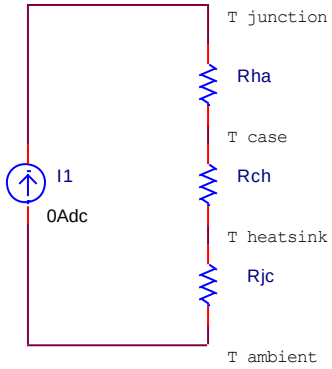


Σχήμα 3.15

- R_{jc}, η θερμική αντίσταση μεταξύ του ημιαγωγού και της συσκευασίας του. Αυτή η αντίσταση είναι κατασκευστική σταθερά και δεν μπορεί να μεταβληθεί. Για τους στραθεροποιητές η αντίσταση αυτή αναφέρεται στα τεχνικά χαρακτηριστικά 5 °C/W (στην συσκευασία TO-220). Η τιμή αυτή δηλώνει οτι για κατανάλωση 4,5Watt θερμικής ισχύος η θερμοκρασία του ημιαγωγού θα είναι 4,5*5=22,5 °C υψηλότερα απο την θερμοκκρασία της συσκευασίας του.
- R_{ca}, η θερμική αντίσταση μεταξύ της συσκευασίας του ημιαγωγού και του περιβάλλοντος. Η αντίσταση αυτή δίνεται στα τεχνικά χαρακτηριστικά των σταθεροποιητών ώς 65°C/W. Η τιμή αυτή είναι ιδιαίτερα μεγάλη. Ενδεικτικά βλέπουμε οτι για 4,5 Watt ισχύος η θερμοκρασία συσκευασίας είναι κατα 65*4,5 =292.5°C υψηλότερα απο την θερμοκρασία περιβάλλοντος! Δεδομένου οτι η μέγιστη επιτρεπόμενη θερμοκρασία για το εσωτερικό του ημιαγωγού δηλώνεται ώς 125°C στα τεχνικά χαρακτηριστικά, βλέπουμε οτι η ανάγκη χρήσης ψύκτρας είναι επιτακτική. Να σημειωθεί οτι η μεταφορά της ενέργειας απο την συσκευασία

κατ'ευθείαν στο περιβάλλον γίνεται με τους μηχανισμούς ακτινοβολίας και μεταγωγής, οι οποίοι ειναι ιδιαίτερα αναποτελεσματικοί για την μικρή επιφάνεια του ολοκληρωμένου.

Με την χρήση της ψύκτρας ουσιαστικά αντικαθιστούμε την αντίσταση R_{ca} με δύο εν σειρά αντιστάσεις: τις R_{ch} και R_{ha} :



Σχήμα 3.16

- R_{ch} είναι η αντίσταση μεταξύ της συσκευασίας και της ψύκτρας. Η αντίσταση αυτή εχει να κάνει με τον τρόπο που γίνεται η επαφή των δύο σωμάτων. Η τιμή της αν χρησιμοποιηθεί θερμοαγώγιμη πάστα για την ένωση είναι περίπου 0,5-0,8 °C/W. Δηλαδή η θερμοκρασία της συσκευασίας είναι περίπου 3°C υψηλότερη απο αυτήν της ψύκτρας για θερμική ισχύ 4.5 Watt
- R_{ha}, η αντίσταση μεταξύ ψύκτρας και περιβάλλοντος. Η αντίσταση αυτή είναι η βασική παράμετρος στην οποία επεμβαίνουμε. Η τιμή της εξαρτάται απο το υλικό κατασκευής της ψύκτρας και τα γεωμετρικά χαρακτηριστικά της. Ζητούμενο είναι να κρατήσουμε την τιμή της αρκετά χαμηλή ώστε η θερμοκρασία του ημιαγωγού (~τάση) να μήν υπερβεί την μέγιστη επιτρεπόμενη.

Αν και με την χρήση ψύκτρας προσθέτουμε μία επι πλέον αντίσταση, η ολική αντίσταση του συστήματος είναι αρκετά μικρότερη. Αυτό γίνεται επειδή η μεταφορά της θερμότητας απο την ψύκτρα στο περιβάλλον γίνεται με πολύ πιο αποδοτικό τρόπο απο ότι κατ'ευθείαν απο την επιφάνεια του ολοκληρωμένου. Δεδομένης της θερμικής ισχύος των σταθεροποιητών και της μέγιστης επιτρεπτής θερμοκρασίας του ημιαγωγού μπορούμε να βρούμε την μεγιστη επιτρεπτή τιμή για την αντίσταση R_{ha}, σύμφωνα με τον τύπο :

$$R_{ha} = \frac{\Delta T}{Pth} - (R_{jc} - R_{ch}),$$

όπου Δτ η συνολική διαφορά θερμοκρασίας μεταξύ περιβάλλοντος και μέγιστης θερμοκρασίας ημιαγωγού.

Η χρήση ανεμιστήρα βελτιώνει την διαδικασία ψύξης με δύο τρόπους: Μειώνει την θερμοκρασία του αέρα στο περιβάλλον της ψύκτρας, αφού τον ανανεώνει συνεχώς με αέρα απο το εξωτερικό του κουτιού συσκευασίας, και επίσης μειώνει την θερμική αντίσταση της ψύκτρας, αφού επιταχύνει την θερμική αλληλεπίδραση της με τον αέρα. Αν θεωρήσουμε επομένως της θερμοκρασία περιβάλλοντος ίση με 25 °C και την θερμοκρασία στο εσωτερικό του συστήματος κατα προσέγγιση ίση με αυτήν του περιβάλλοντος προκύπτει η τιμή ΔT = 125-25 = 100 °C. Αντικαθιστώντας τώρα τις τιμές στον παραπάνω τύπο έχουμε την μέγιστη επιτρεπτή αντίσταση R_{ha} ίση με 18 °C/W.

Οι ψύκτρες του εμπορίου συνήθως συνοδεύονται απο την τιμή της θερμικής τους αντίστασης στα τεχνικά χαρακτηριστικά τους. Ζητούμενο είναι να γίνει χρήση μίας ψύκτρας με αντίσταση μικρότερη απο την μέγιστη επιτρεπτή που μόλις υπολογίσαμε. Στην υλοποίηση της πλακέτας μας δεν επιλέξαμε κάποια ψύκτρα του εμπορίου, αλλα χρησιμοποιήσαμε μία απλή ορθογώνια πλάκα αλουμινίου, διαστάσεων 2,5x25x100 mm, στην οποία προσκολλήσαμε τους δύο σταθεροποιητές τάσης. Για να εξασφαλίσουμε οτι το σύστημα ψύξης που έχουμε χρησιμοποιήσει καλύπτει τις ανάγκες μας, χρειάζεται να υπολογίσουμε την θερμική αντίσταση της ψύκτρας, ή να γίνει δοκιμή λειτουργίας και με μέτρηση της θερμοκρασίας της ψύκτρας να αποφανθούμε αν η απαγωγή θερμότητας απο τα ολοκληρωμένα είναι επαρκής. Ας δούμε την διαδικασία υπολογισμού της θερμικής αντίστασης της ψύκτρας:

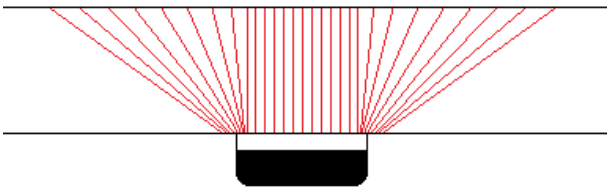
Η θερμική αντίσταση της ψύκτρας απο τελείται απο την εν σειρά σύνδεση δύο αντιστάσεων: Της εσωτερικής και εξωτερικής αντίστασης, R_{in} και R_{ex} αντίστοιχα. Η R_{in} είναι η αντίσταση που συναντάει η θερμότητα κατα την διέλευση της απο το εσωτερικό του αλουμινίου μέχρι την ελεύθερη επιφάνειά του, και η R_{ex} είναι η αντίσταση μετάδοσης της θερμότητας απο την εξωτερική επιφάνεια της ψύκτρας στον αέρα.

Η αντίσταση R_{in} προκύπτει απο τον νόμο του Fourier για την διάδοση της θερμότητας

$$R_\theta = \frac{x}{k*A},$$

όπου x το πάχος του υλικού σε m, k ειδική θερμική αγωγιμότητα υλικού σε W/(K*m), A επιφάνεια του υλικού κάθετα στην μεταφορά της θερμότητας σε m^2.

Στην περίπτωση του αλουμινίου έχουμε k=237 (καθαρή μορφή) ή k=120-180 (διάφορα κράμματα του εμπορίου). Ωστόσο, ο τύπος αυτός είναι έγκυρος μόνο στην περίπτωση που οι παράμετροι x,k,A είναι σταθερές. Στην περίπτωση μας όμως η πηγή της θερμότητας (τα ολοκληρωμένα) καταλαμβάνει μία περιορισμένη επιφάνεια της ψύκτρας. Συγκεκριμένα, το κάθε τρανζίστορ καταλαμβανει 1,2 cm^2 σύμφωνα με τα τεχνικά χαρακτηριστικά. Επομένως, η θερμική αντίσταση αποτελείται απο πολλαπλές αντιστάσεις συνδεδεμένες παράλληλα. Η κάθε μία αντίσταση καθορίζεται απο το διαφορετικό μήκος που πρέπει να διανύσει η θερμότητα μέχρι την εξωτερική επιφάνεια της ψύκτρας. Το φαινόμενο αυτό ονομάζεται spreading resistance:



Σχήμα 3.17

Κάτοψή ψύκτρας και ολοκληρωμένου. Διάδοση της θερμότητας

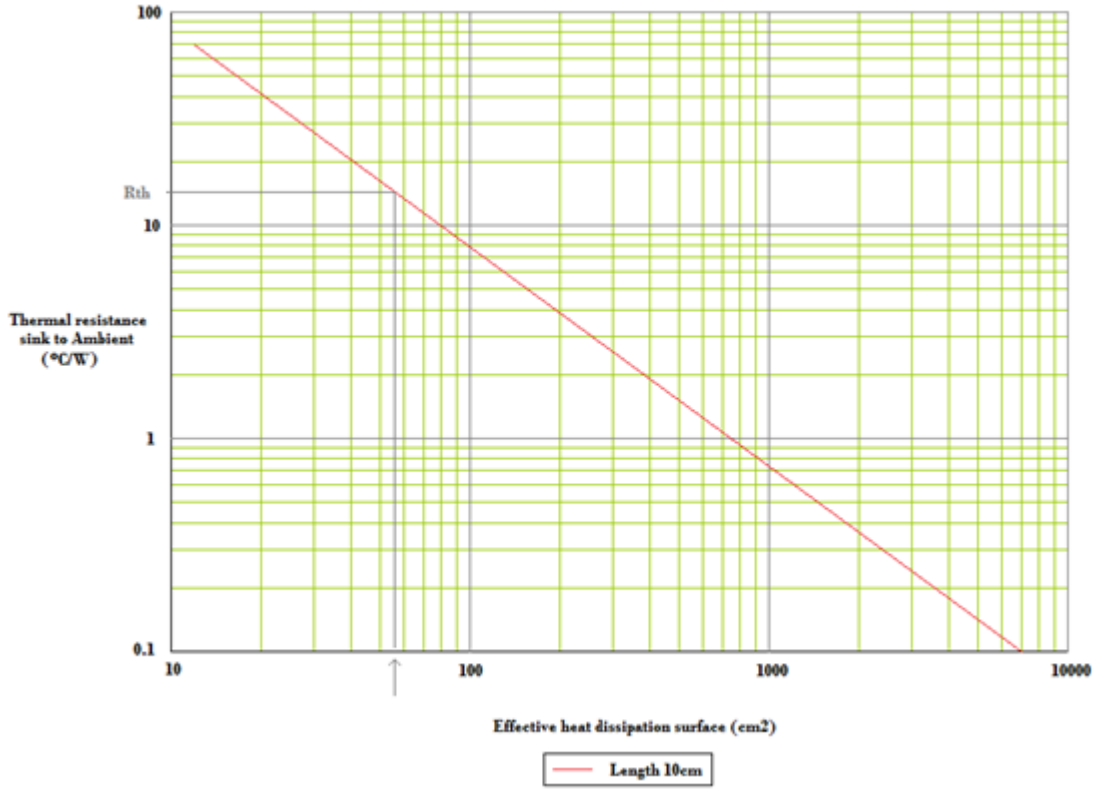
Ας θεωρήσουμε προς στιγμήν οτι η ψύκτρα περιλαμβάνει μόνο ένα τμήμα με επιφάνεια ίση με την επιφάνεια του ολοκληρωμένου. Στην περίπτωση αυτή μπορεί να χρησιμοποιηθεί ο νόμος του Fourier. Αντικαθιστώντας τις τιμές των παραμέτρων στον τύπο, βρισκουμε θερμική αντίσταση ίση με 0,14 °C/W. Η τιμή αυτή είναι ιδιαίτερα μικρή και ο

συνυπολογισμός των παράπλευρων παράλληλων αντιστάσεων θα οδηγήσει στην περαιτέρω μείωσή της. Επομένως, μπορούμε να θεωρήσουμε την αντίσταση R_{in} αμελητέα.

Ο υπολογισμός της αντίστασης R_{ex} είναι ένα εγχείρημα αρκετά πιο περίπλοκο, ειδικά στην περίπτωση που γίνεται χρήση ανεμιστήρα. Ο μηχανισμός μεταφοράς της θερμότητας γίνεται κυρίως μέσω συναγωγής, ενώ ένα μέρος της μεταφέρεται και μέσω ακτινβολίας. Η αναλυτική επίλυση ενός τέτοιου προβλήματος ξεφεύγει απο το σκοπό αυτής της εργασίας, ωστόσο μπορούμε να χρησιμοποιήσουμε κάποιες προσεγγιστικές τεχνικές. Μία απο αυτές είναι η χρήση του παρακάτω διαγράμματος. Το διάγραμμα αυτό συνδέει την θερμική αντίσταση μίας ψύκτρας με την ενεργό επιφάνειά της (την επιφάνεια αλληλεπίδρασης με τον ωθούμενο αέρα) και το μήκος της (διάσταση παράλληλη στην ροή του αέρα). Το διάγραμμα αυτό λαμβάνει ως προυπόθεση της ύπαρξη στρωτής και όχι τυρβώδους ροής του αέρα.

Στον άξονα x έχουμετην ενεργό επιφάνεια της ψύκτρας, σε cm^2 σε λογαριθμική κλίμακα.Στον άξονα y έχουμε τις τιμές θερμικής αντίστασης σε °C/W σε λογαριθμική κλίμακα.Η ευθεία που έχει χαραχθεί αφορά μήκος ψύκτρας 10 cm.

η ενεργός επιφάνεια της ψύκτρας μας σε cm^2 είναι: 2*10*2,5+2*10*0,25+2*2,5*0,25 = 56,25 cm^2. Το σημείο της ευθειας με τετμημένη x=56,25 έχει τεταγμένη με τιμή μικρότερη απο την μέγιστη τιμή θερμικής αντίστασης που υπολογίισαμε προηγουμένως.



Σχήμα 3.18

πηγή: http://64.19.142.12/homepages.which.net/~paul.hills/Heatsinks/Graph1ex.gif
πηγη: http://robots.freehostia.com/Heatsinks/HeatsinksBody.html

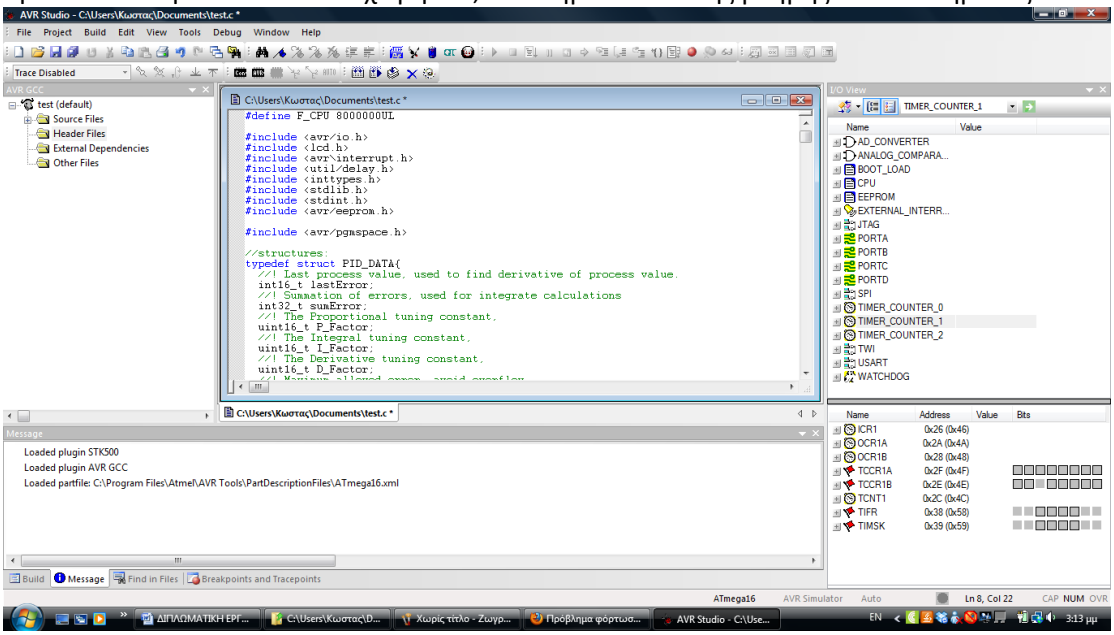
Γέφυρα οδήγησης κινητήρα

Η οδήγηση του κινητήρα γίνεται απ'ευθείας απο το τροφοδοτικό των 24 Volts μέσω της γέφυρας – Η. Το ρεύμα που αντλεί ο κινητήρας σε φυσιολογική λειτουργία μετρήθηκε περίπου στα 0,8 Α. Η διακοπτική συμπεριφορά της γέφυρας περιορίζει ιδιαίτερα το ποσό της ισχύος που καταναλώνεται στο εσωτερικό της (και εμφανίζεται ώς θερμότητα). Συγκεκριμένα, η ισχύς αυτή είναι περίπου ίση με την πτώση τάσης που εμφανίζεται στα άκρα της γέφυρας επι το ρεύμα που την διαρρέει , πολλαπλασιασμένα επι το ποσοστό duty cycle του PWM σήματος, αφού η γέφυρα άγει κατα το ποσοστό αυτό επι του χρόνου λειτουργίας της. Εφ'όσον η πτώση τάσης της γέφυρας είναι περίπου 2Volt προκύπτει οτι η θερμότητα που παράγεται απο το ολοκληρωμένο αυτό είναι αρκετά μικρότερη απο την περίπτωση των σταθεροποιητών τάσης. Ωστόσο, ο κινητήρας ενδέχεται να αντιμετωπίσει καταστάσεις έντονου μηχανικού φορτίου κατα τις οποίες οι απαιτήσεις σε ρεύμα αυξάνονται. Παρόμοια συμπεριφορά συναντάται κατα την εκκίνηση του κινητήρα απο στάση. Γι'αυτόν τον λόγο χρησιμοποιήθηκε μία ψύκτρα όμοια με αυτήν των σταθεροποιητών, στο ρεύμα αέρα που οδηγεί ο ίδιος ανεμιστήρας.

Τέλος, η συσκευή δοκιμάστηκε σε συνεχή λειτουργία με κλειστη την συσκευασία για να επιβεβαιωθεί οτι η θερμοκρασία των ψυκτρών σταθεροποιείται σε τιμή επιτρεπτή, σύμφωνα με την παραπάνω ανάλυση.

3.3 Προγραμματισμός μικροελεγκτή

Ο προγραμματισμός του μΕ έγινε με την χρήση του προγράμματος AVR Studio 4. Το πρόγραμμα αυτό παρέχει εργαλεία συγγραφής κώδικα και διασύνδεσης του υπολογιστή με κάποιο σύστημα προγραμματισμού. Επίσης, διαθέτει προσομοιωτή των περισσότερων μΕ της οικογένειας avr για έλεγχο και debugging του κώδικα.Κατα την προσομοίωση της εκτέλεσης του κώδικα, ο χρήστης έχει την δυνατότητα να παρακολουθήσει και να ελέγξει την κατάσταση όλων των καταχωρητών, των σημαιών και της μνήμης του συστήματος.



Σχήμα 3.19

Επειδή ο προγραμματισμός έγινε σε γλώσσα C , στο πρόγραμμα προστέθηκε ο μεταγλωτιστής gcc μέσω του πακέτου winavr, ο οποίος αναλαμβάνει την μετατροπή του κώδικα σε γλώσσα assembly των μικροελεγκτών avr.

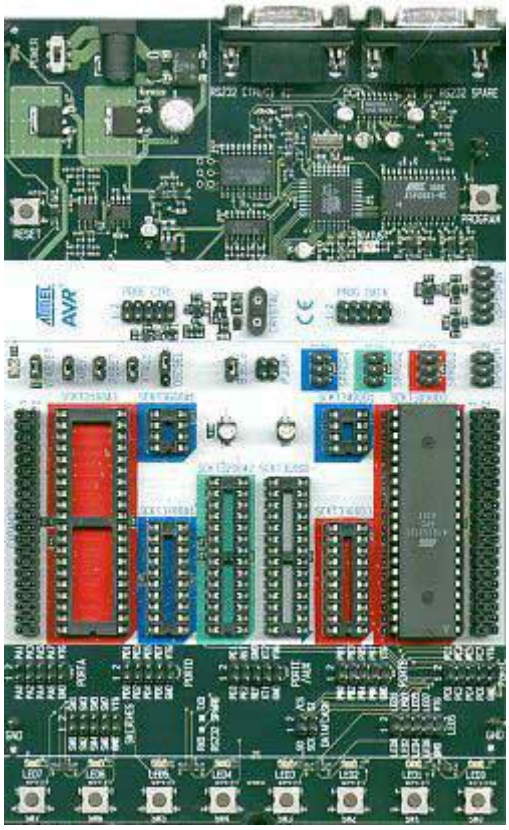
Το πρόγραμμα AVR studio επίσης παρέχει την δυνατότητα εγγραφής και ανάγνωσης της μνήμης EEPROM του μικροελεγκτή. Αυτή η λειτουργία είναι ιδιαίτερα χρήσιμη όταν ο μΕ χρησιμοποιείται για την καταγραφή δεδομένων. Στην περίπτωση του συστήματός μας έγινε χρήση της για την ανάκτηση των μετρήσεων της βηματικής απόκρισης του κινητήρα.

Για την φόρτωση του προγράμματος στον μΕ χρησιμποιήθηκε η πλακέτα προγραμματισμού Atmel STK500 και η μέθοδος ISP (In-System-programming). Η μέθοδος αυτή παρέχει την δυνατότητα προγραμματισμού του μΕ ενώ αυτός βρίσκεται στην πλακέτα λειτουργίας του, μέσω μία σύνδεσης 6 ή 10 ακροδεκτών, οι οποίοι συνδέονται στους κατάλληλους ακροδέκτες του μΕ όπως στην εικόνα:

ISP Connectors: 6-pin & 10-pin

Σχήμα 3.20

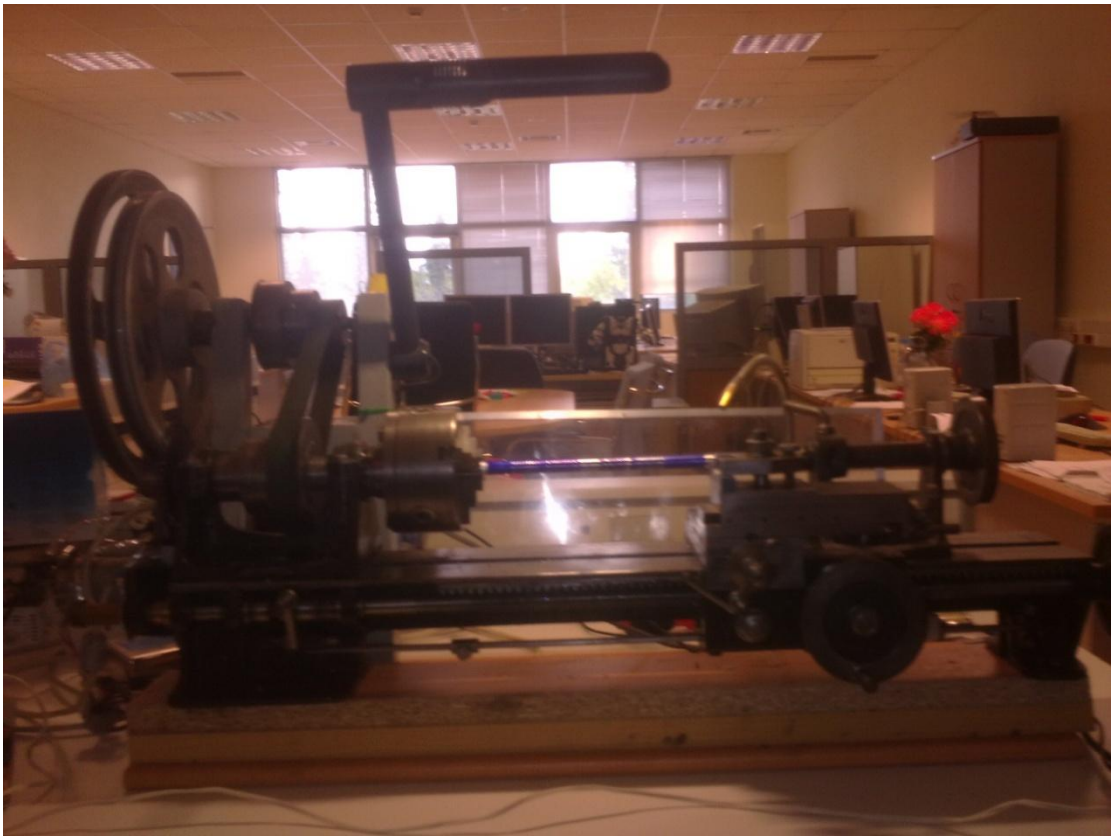
Το πρόγραμμα AVR studio συνδέεται μέσω της σειριακής θύρας του υπολογιστή με την πλακέτα προγραμματισμού STK500. Ηπλακέτα αυτή αναλαμβάνει την ISP σύνδεση με τον μικροελεγκτή ο οποίος μπορεί να βρίσκεται στην ίδια την πλακέτα, ή σε κάποια ανεξάρτητη.



Σχήμα 3.21

Η πλακέτα STK500 παρέχει και συνδέσεις για τις διάφορες θύρες του μΕ, καθώς και 8 πλήκτρα και 8 LED για την δοκιμαστική εκτέλεση προγραμμάτων. Παρατίθεται το εγχειρίδιο χρήσης όπως κυκλοφορεί απο την κατασκευάστρια εταιρεία.

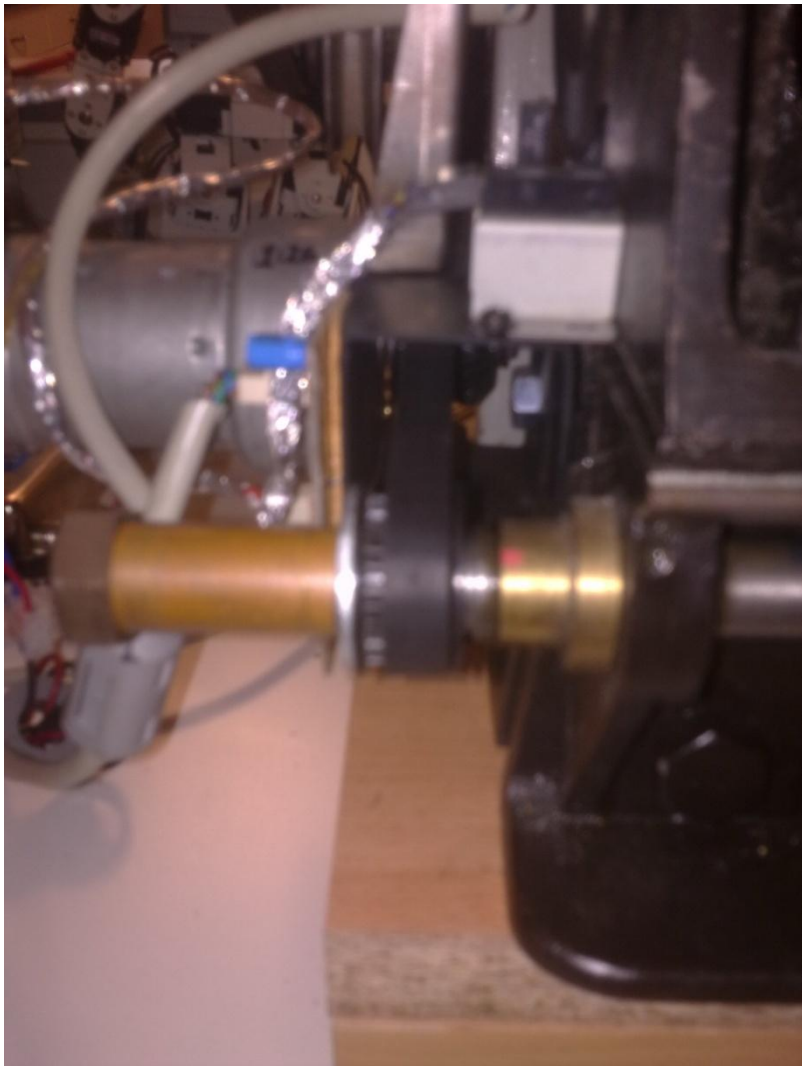
3.4 Μηχανική κατασκευή



Σχήμα 3.22

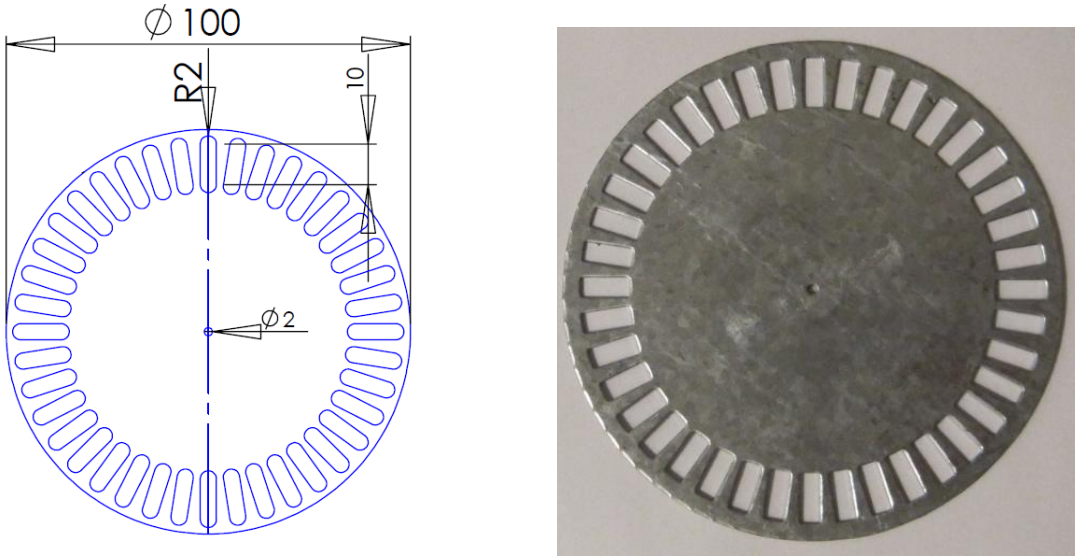
Η μηχανική κατασκευή περιλαμβάνει τις τροποποιήσεις πού έγιναν ωστε να προσαρμοστεί ο κινητήρας στον μέχρι πρότινος χειροκίνητο τόρνο. Το γραναζωτό σύστημα μετάδοσης που μετέφερε την ισχύ απο τον άξονα του τσώκ στον κοχλία του σεπόρτι αφαιρέθηκε ώστε να αντικατασταθεί απο το ηλεκτρονικό σύστημα.

Για την τοποθέτηση του κινητήρα κατασκευάστηκε αποστάτης απο ορθογώνιο σιδερένιο σωλήνα. Το κομμάτι που χρησιμοποιήθηκε κόπηκε στο επιθυμητό σχήμα και έγιναν οι κατλάλληλες οπές για την στερέωση του κινητήρα. Στην συνέχεια στερεώθηκε επάνω στο σώμα του τόρνου. Η μετάδοση της ισχύος απο τον σερβοκινητήρα στον κοχλία του σεπόρτι έγινε με ιμάντα χρονισμού και κατάλληλες τροχαλίες.



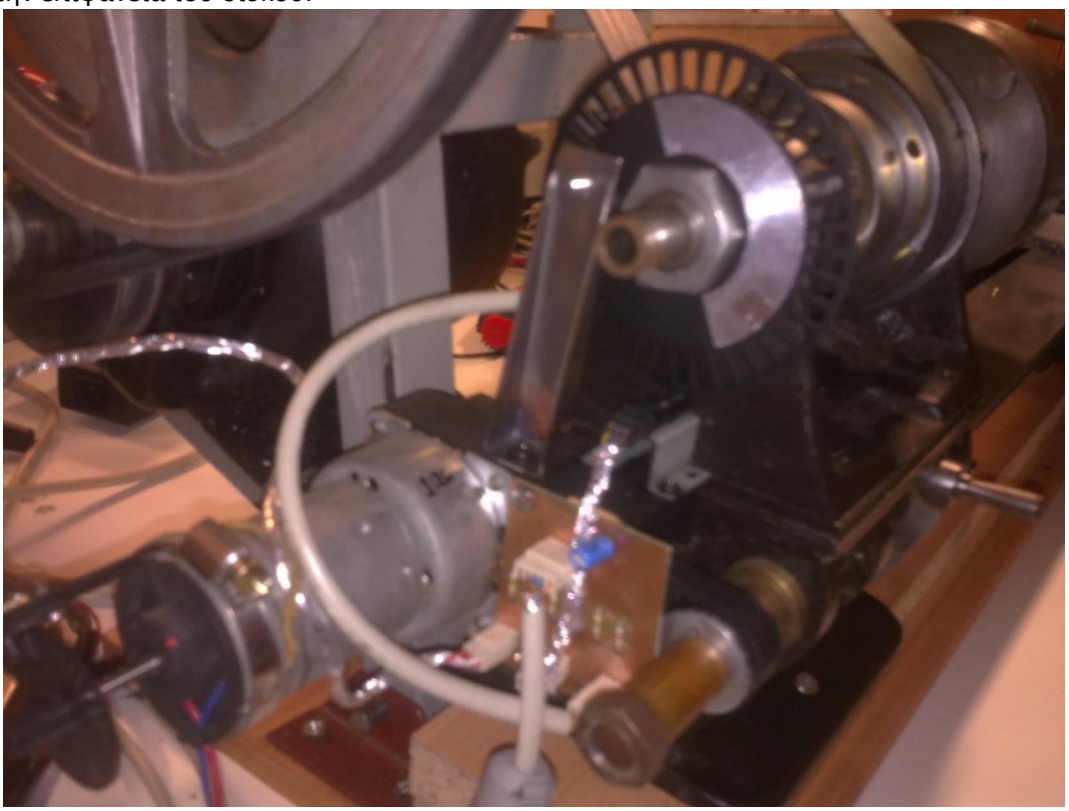
Σχήμα 3.23

Ο αισθητήρας του άξονα αναφοράς δημιουργήθηκε απο έναν μεταλλικό δίσκο στον οποίο διανοίχτηκαν οπές σε ειδικό πνευματικό μηχάνημα (punching machine). Το σχέδιο της κατασκευής έγινε στο πρόγραμμα σχεδίασης solidWorks.

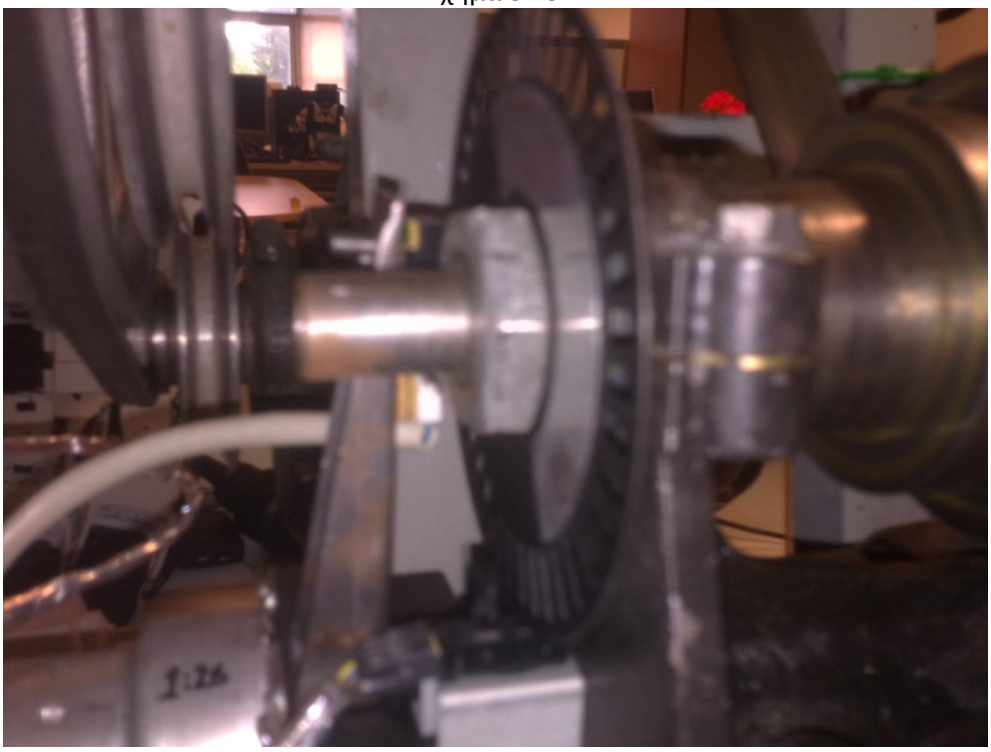


Σχήμα 3.24

Ο δίσκος τοποθετήθηκε απ’ευθείας στον άξονα του τσώκ. Στην επιφάνεια του δίσκου δημιουργήθηκε ένα ημικύκλιο απο ανακλαστικό υλικό, ενώ ο υπόλοιπος δίσκος είναι χρωματισμένος με μαύρο χρώμα.Τα οπτικά στοιχεία των δύο αισθητήρων τοποθετήθηκαν επάνω στο σωμα του τόρνου ωστε να βρίσκονται στην σωστή θέση σε σχέση με τον δίσκο.Το ένα στοιχείο τοποθετήθηκε έτσι ώστε ο δίσκος να παρμβάλλεται στην οπτική επαφή πομπού-δέκτη και το άλλο άτσι ώστε η οπτική επαφή να γίνεται μέσω ανάκλασης στην επιφάνεια του δίσκου.

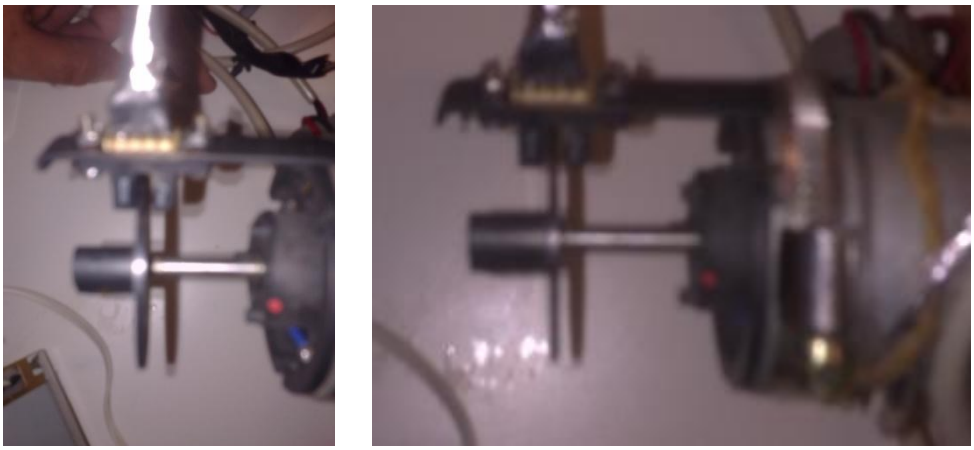


Σχήμα 3.25



Σχήμα 3.26

Ο αισθητήρας του σερβοκινητήρα αποτελείται απο έναν πλαστικό δίσκο με οπές, ο οποίος προσαρμόστηκε στον άξονα του κινητήρα πριν απο τον μειωτήρα. Τα οπτικά στοιχεία στερεώθηκαν επάνω στον κινητήρα.



Σχήμα 3.27

Η πλακέτα που φέρει το σύστημα οδήγησης των αισθητήρων τοποθετήθηκε στο σώμα του τόρνου, σε μικρή απόσταση απο αυτούς. Τα καλώδια απο τα οπτικά στοιχεία στην πλακέτα αυτήν και απο την πλακέτα στην κεντρική πλακέτα του συστήματος έχουν προστατευτεί με γειωμένο φύλλο αλουμινίου για τις παρεμβολές και πλαστικό περίβλημα για μηχανική προστασία.

Το σώμα του τόρνου, του σερβοκινητήρα και το τροφοδοτικό του όλου συστήματος είναι συνδεδεμένα στην γείωση. Εκεί επίσης είναι συνδεδεμένη η χαμηλή τάση του συστήματος, τα περιβλήματα των καλωδίων και η γείωση της πλακέτας.

Κεφάλαιο 4: Ανάλυση λειτουργίας συστήματος - ρυθμίσεις

4.1 Μελέτη ροπής

Η ισχύς του κινητήρα προκύπτει απο το γινόμενο της γωνιακής ταχύτητάς του επι την ροπή που ασκεί στονν άξονά του. Η τελική δύναμη που ασκεί το κοπτικό εργαλείο στο υπο επεξεργασία τεμάχιο προκύπτει απο την ισχύ του κινητήρα και τον τρόπο μετάδοσής της. Συγκεκριμένα, η κίνηση του κινητήρα περνάει απο τα εξής στάδια:

- Μειωτήρας κινητήρα: Η ροπή του κινητήρα αυξάνεται κατα 26 φορές σε αυτό το στάδιο, ενώ η ταχύτητα μειώνεται κατα τον ίδιο λόγο.
- Σχέση μετάδοσης τροχαλιών χρονισμού: Σε αυτό το στάδιο μπορούμε να μεταβάλλουμε κι άλλο την ροπή.
- Μετάδοση του κοχλία:

Το σπείρωμα του κοχλία ασκεί μία δύναμη στο σεπόρτι. Η ροπή που ασκείται στον κοχλία ισοδυναμεί με μία εφαπτομενική γραμμική δύναμη στην περιφέρεια του σπειρώματος. Αν θεώρήσουμε το σπείρωμα ώς ένα κεκλιμένο επίπεδο, η κλίση του σε συνδυασμό με την περίμετρο του σπειρώματος καθορίζουν την σχέση μετάδοσης του

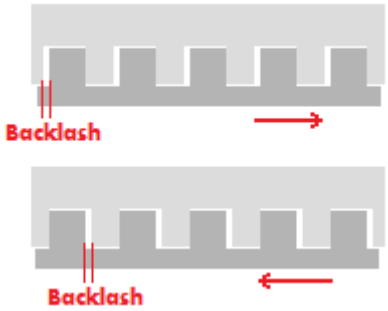
κοχλία και το μηχανικό πλεονέκτημα που αυτός εισάγει. Μπορούμε να απλοποιήσουμε το πρόβλημα ως εξής: Η ροπή που ασκείται στον κοχλία προέρχεται απο την εφαπτομενική (γραμμική) δύναμη που ασκεί ο ιμάντας μετάδοσης στην τροχαλία. Η τροχαλία έχει συγκεκριμένη ακτίνα. Η ισχύς αυτής της δύναμης είναι το γινόμενο του μέτρου της επι την ταχύτητα του σημείου εφαρμογής της. Σε μία περιστροφή του κοχλία το σημείο εφαρμογής της δύναμης αυτής έχει μετατοπιστεί κατα την περιμετρο της τροχαλίας. Παράλληλα, το σεπόρτι έχει μετατοπιστεί κατα ένα βήμα του σπειρώματος του κοχλία. Θυμίζω οτι βήμα ενός κοχλία είναι η αξονική μετατόπισή του σε μία περιστροφή. Επομένως, οι ταχύτητες του σεπόρτι και του ιμάντα χρονισμού σχετίζονται με τον λόγο $\frac{\beta\acute{\eta}\mu\alpha\ \kappa o\chi\lambda\acute{\iota}\alpha}{\pi\varepsilon\rho\acute{\iota}\mu\varepsilon\tau\rho o\varsigma\ \tau\rho o\chi\alpha\lambda\acute{\iota}\alpha\varsigma}$. Η απαίτηση για σταθερή ισχύ μας δίνει οτι η δύναμη έχει μεταβληθεί κατα τον αντίστροφο λόγο. Ετσι προκύπτει η μείωση που εισάγει ο κοχλίας του σεπόρτι.

Η πλευρική δύναμη που ασκείται απο το εργαλείο κοπής στο τεμάχιο υπο επεξεργασία δοκιμάστηκε τοποθετώντας τεμάχιο απο χάλυβα, που είναι ιδιαίτερα σκληρό υλικό, και δοκιμάζοντας λειτουργία βαθιάς κοπής. Παρατηρήθηκε οτι η δύναμη είναι ικανοποιητική και γι'αυτό δεν χρησιμοποιήθηκε τροχαλία μείωσης στην σύνδεση του κινητήρα με τον κοχλία.

4.2 Μη ελέγξιμες παράμετροι

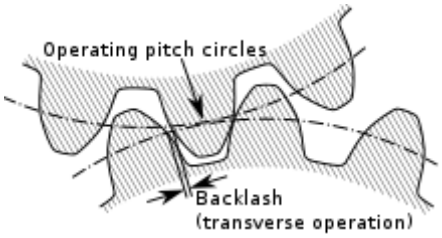
4.2.1 Πρόβλημα backlash:

Για να κατανοήσουμε το φαινόμενο backlash, ας θεωρήσουμε την αλληλεπίδραση ενός κοχλία με το αντίτοιχο περικόχλιο. Ο τρόπος μετάδοσης της κίνησης, είναι η μηχανική συμπλοκή των δύο σπειρωμάτων. Η συμπλοκή επιτυγχάνεται μεταξύ της προεξέχουσας σπείρας στον κοχλία και της αντίστοιχης σπειροειδούς αύλακας στο περικόχλιο. Οι διαστάσεις των δύο αυτών όγκων δεν μπορούν να είναι ακριβώς ίσες. Σε τέτοια περίπτωση η τριβή μεταξύ τους θα εμπόδιζε την ολίσθηση. Αντίθετα, η σπείρα είναι ελαφρώς μικρότερη απο την αντίστοιχη αύλακα. Σαν αποτέλεσμα, κάθε φορά που η φορά κίνησης αντιστρέφεται, η σπείρα διανύει μία απόσταση στο εσωτερικό της αύλακας, κατα την οποία δεν υπάρχει μηχανική συμπλοκή των δύο σωμάτων. Το φαινόμενο αυτό ονομάζεται backlash.



Σχήμα 4.1

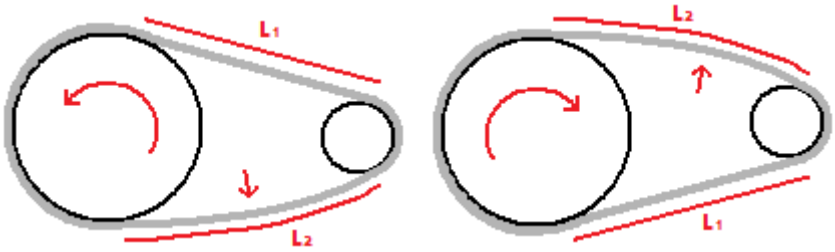
Το φαινόμενο αυτό παρουσιάζεται σε κάθε είδος μηχανικής μετάδοσης κίνησης. Τα γρανάζια είναι επίσης μία τετριμμένη περίπτωση:



Σχήμα 4.2

πηγή: wikipedia/backlash

Στην περίπτωση της μετάδοσης μέσω ιμάντων, ένα παρόμοιο φαινόμενο προκαλείται απο την διαφορά της τάσης του ιμάντα ανάλογα με την φορά περιστροφής. Το μήκος του ιμάντα ανάμεσα στις τροχαλίες δεν είναι ίδιο για κάθε πλευρά (L2 > L1). Η διαφορά στο μήκος αυτό αποτελεί ισοδύναμο backlash (L2 – L1)



Σχήμα 4.3

Στο σύστημά μας έχουμε μετάδοση ισχύος και με τα τρία μέσα. Γρανάζια υπάρχουν στον μειωτήρα του κινητήρα, κινούμε μέχω ιμάντα τον κοχλία, και αυτός κινεί το σεπόρτι μέσω του σπειρώματός του. Οπως είπαμε, το φαινόμενο αυτό εμφανίζεται μόνο κατα την αλλαγή της φοράς κίνησης. Κατα την κοπή σπειρώματος η φορά κίνησης είναι σταθερή. Σε κάθε κύκλο κοπής-επιστροφής το φαινόμενο εκδηλώνεται δύο φορές με αντίθετη φορά.

Επίσης, το μήκος του backlash είναι πάντα σταθερό και ανεξάρτητο απο την φορά της κίνησης. Επομένως, στην περίπτωσή μας ο εκδηλώσεις του φαινομένου αλληλοαναιρούνται, και καμία δεν συμβαίνει κατα την διάρκεια της κοπής του σπειρώματος. Επομένως, είναι ένα φαινόμενο που δεν χρήζει αντιμετώπισης για την συγκεκριμένη εφαρμογή. Αξίζει να σημειωθεί οτι το φαινόμενο αυτό επικρατεί και στα μηχανικά συστήματα κοπής σπειρωμάτων χωρίς αρνητικές επιδράσεις.

Ωστόσο, το σύστημα της μας μπορεί να μετρήσει το backlash του μηχανήματος, κάνοντας χρήση ενός διακόπτη ασφαλείας και κάποιας ειδικής ρουτίνας. Θυμίζω οτι ο διακόπτης ασφαλείας βρίσκεται επάνω στον τόρνο και ενεργοποιείται ώστε να σταματήσει την κίνηση του σεπόρτι σε περίπτωση που αυτό βρίσκεται κοντά σε κάποιο εμπόδιο. Μία μέθοδος θα ήταν η εξής:

1. Κίνηση μέχρι την ενεργοποίηση του διακόπτη
2. Όταν ο διακόπτης ενεργοποιηθεί, πέδηση του κινητήρα
3. Μόλις ο κινητήρας ακινητοποιηθεί, μέτρηση ολίσθησης (slip)
4. Κίνηση προς την αντίθετη φορα. Μέτρηση μήκους κίνησης μέχρι την απελευθέρωση του διακόπτη (back_length)
5. backlash = back_length – slip

4.2.2 Ελαστικότητα ιμάντα.

Οπως έχουμε αναλύσει, ο αισθητήρας του κινητήρα μας παρέχει μία μέτρηση σχετικής θέσης και όχι απόλυτης. Εκτός απο αυτό, η τοποθέτηση του στον άξονα του κινητήρα, έχει σαν αποτέλεσμα κάποια στοιχεία κίνησης να βρίσκονται μετά τον αισθητήρα και να μήν μπορούμε να μετρήσουμε την επίδρασή τους. Αυτά τα στοιχεία είναι ο μειωτήρας, ο ιμάντας και ο κοχλίας. Η απόφαση αυτή δεν προκαλεί κάποιο πρόβλημα μόνο στην περίπτωση που τα στοιχεία αυτά παρουσιάζουν σταθερή γεωμετρία και χαρακτηριστικά. Ενα τέτοιο χαρακτηριστικό είναι, όπως είπαμε το backlash. Ο μειωτήρας και ο κοχλίας δεν εκδηλώνουν κάποια άλλη δυναμική συμπεριφορά. Ο ιμάντας ωστόσο, έχει μία ελαστικότητα. Αυτο σημαίνει οτι το μήκος του εξαρτάται απο την δύναμη τάσης του. Εφ'όσον η τάση του εξαρτάται απο την αντίσταση που δέχεται το σεπόρτι κατα την κίνησή του, δεν μπορεί να υποτεθεί σταθερή. Αυτό το φαινόμενο εισάγει ένα σφάλμα στην μέτρηση θέσης.

Ενας τρόπος αντιμετώπισης του φαινομένου αυτού θα ήταν ο αισθητήρας του κινητήρα να τοποθετηθεί στον άξονα του κοχλία και όχι του κινητήρα.Ετσι η όποια επίδραση της ελαστικότητας του ιμάντα θα μετρούταν απο τον αισθητήρα.

Ωστόσο, ο ιμάντας που έχει χρησιμοποιηθεί είναι ιδιαίτερης σκληρύτητας και οι προδιαγραφές του είναι για εφαρμογές αρκετά πιο ισχυρές απο την συγκεκριμένη. Ετσι, η επίδραση του φαινομένου αυτού έχει κρατηθεί σε επίπεδο μηδαμινό.

4.3 Ρυθμιση PID. Μελετη βηματικής απόκρισης

Η ρύθμιση των κερδών του ελεγκτή PID,όπως έχουμε αναλύσει, είναι καθοριστικής σημασίας για την συμπεριφορά του συστήματος. Η μεθοδολογία ρύθμισης μπορεί να είναι έιτε αναλυτική, είτε προσεγγιστική.

Η συνάρτηση μεταφοράς του κινητήρα που χρησιμοποιήθηκε δεν είναι γνωστή. Ωστόσο, ακόμα και στην περίπτωση που γνωρίζαμε πλήρως τα χαρακτηριστικά του κινητήρα, αυτός είναι συνδεδεμένος με άλλα μηχανικά μέρη που συνεισφέρουν στο σύστημα τουλάχιστον ως επι πλέον ροπές αδράνειας και αποσβέσεις. Επομένως είναι χρήσιμη η προσπάθεια αναγνώρισης του συνολικού υπο ελέγχου συστήματος.Η προσπάθεια αυτή θα στηριχθεί στην καταγραφή και μελέτη της βηματικής του απόκρισης.

Για αυτόν τον σκοπό θεωρούμε το μηχανικό σύστημα με είσοδο την τάση τροφοδοσίας του κινητήρα και έξοδο την γωνιακή ταχύτητά του.Για την καταγραφή της βηματικής απόκρισης του συστήματος έχει χρησιμοποιηθεί ειδικό τμήμα του κώδικα, που περιλαμβάνει την χρήση του χρονιστή timer2, μία ρουτίνα διακοπών και την ρουτίνα tuning_mode(). Η λειτουργία της ρουτίνας αυτής έχει αναλυθεί σε προηγούμενη ενότητα. Το σκεπτικό λειτουργίας είναι το εξής:

Τροφοδοτούμε τον κινητήρα με πλήρη ισχύ και καταγράφουμε τις χρονικές περιόδους μεταξύ της έλευσης των παλμών απο τον αισθητήρα του. Η χρονομέτρηση γίνεται με την βοήθεια του χρονιστή ο οποίος εκτελείται σε CTC λειτουργία, και μία ρουτίνα διακοπών αυξάνει μία μεταβλητή κάθε φορά που ο χρονιστής φτάνει στην τιμή του καταχωρητή σύγκρισης. Η μονάδα του χρόνου με την οποία μετρούμε εξαρτάται απο την συχνότητα ρολογιού, τον prescaler του χρονιστή και τον καταχωρητή σύγκρισης. Ο τελευταίος ρυθμίζεται στην χαμηλότερη δυνατή τιμή, ώστε να μετρούμε με την καλύτερη ακρίβεια χωρίς υπερχείλιση. Οι χρονικές περίοδοι αποθηκεύονται σε έναν πίνακα κάθε

φορά που έχουμε έλευση ενός νέου παλμού και η μεταβλητή μέτρησης μηδενίζεται ξανά. Συνολικά μετρούνται 400 τιμές.

Οι τιμές αυτές είναι αντιστρόφως ανάλογες της μέσης ταχύτητας του κινητήρα στην διάρκεια στην οποία μετρήθηκαν. Επίσης, όταν αθροιστούν μας δίνουν τις χρονικές στιγμές έλευσης των παλμών απο τον αισθητήρα του κινητήρα.

Για να έχουμε πρόσβαση στα δεδομένα που καταγράψαμε η ρουτίνα τα αντιγράφει στην μνήμη EEPROM μετα την απόκτησή τους. Αυτή η διαδικασία δεν θα μπορούσε να γίνει απ'ευθείας καθώς είναι ιδιαίτερα χρονοβόρα. Η μνήμη EEPROM έχει την ιδιότητα να συγκρατεί τα δεδομένα ακόμα και όταν βρίσκεται εκτός τροφοδοσίας. Επίσης, το λογισμικό AVR Studio 4 μας δίνει την δυνατότητα ναμεταφέρουμε τα δεδομένα αυτά στον υπολογιστή μέσω της πλακέτας STK500 και να τα εμφανίσουμε σε μορφή κειμένου. Για την καταγραφή στην EEPROM χρησιμοποιήθηκε η βιβλιοθήκη avr/eeprom.h και οι ρουτίνες:

```
eeprom_write_byte()
eeprom_write_word()
```

Η μνήμη EEPROM του μΕ ATmega16 εμφανίζεται ώς ένας πίνακας 512 θέσεων απο διαδοχικές τιμές μήκους 8-bit στο δεκαεξαδικό σύστημα. Ο πίνακας αυτός έχει 32 γραμμές και 16 στήλες . Στην αρχή κάθε γραμμής έχει επι πλέον κάποια στοιχεία διεύθυνσης και στο τέλος έναν αριθμό για τον έλεγχο της εγκυρότητας των δεδομένων της γραμμής. Ολες οι θέσεις της μνήμης αρχικοποιούνται με την τιμή FF_{16} (255_{10}).Ας δούμε τα αποτελέσματα των μετρήσεων:

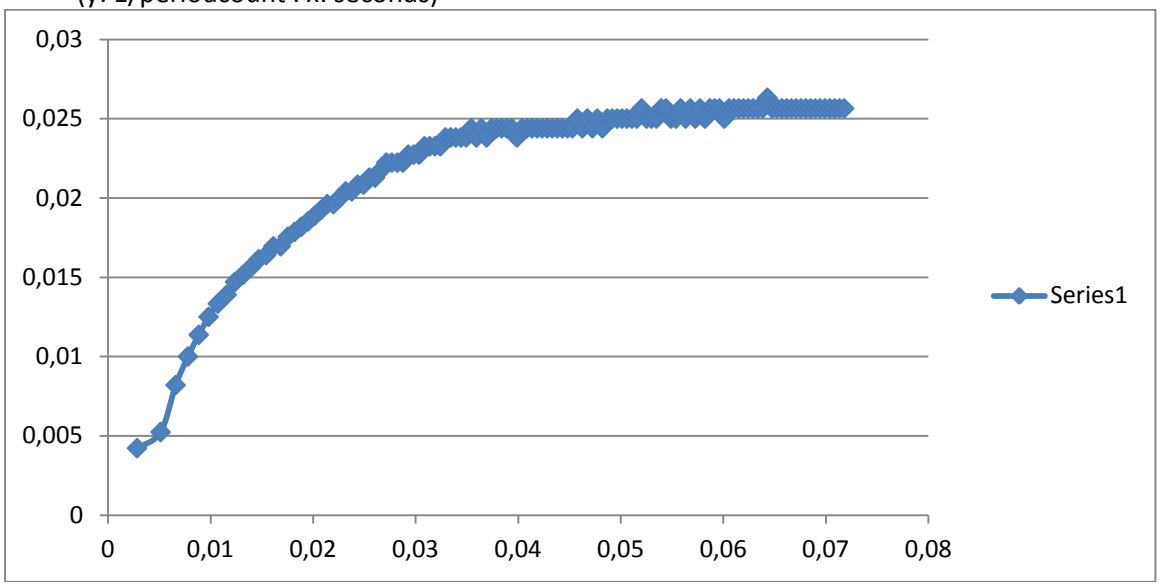
Μέτρηση χρόνου παλμών με μονάδα χρόνου $\frac{3*32}{8000000}$ sec. (F_cpu: 8MHz, prescaler: 32, compare register value: 3) οι μετρηθείσες τιμές ξεκινούν απο την 20η θέση του πίνακα. (στην 18η θέση είναι η τιμή του compare register).

```
:10000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF00
:10001000FFFF03FFEDBF7A6458504B484442403E17
:100020003D3B3B393837363534333332313130307C
:100030002F2F2E2D2D2D2D2C2C2C2B2B2B2B2A2AFC
:100040002A2A2A292A292A29292929292A2929291A
:1000500029292929292929292829282928292828 15
:1000600028282828282728282827272828272827 15
:1000700028272827272728272727272727272726 0E
:1000800027272727272727272727272728272727FF
:1000900027272727272727272726262727262 7F3
:1000A00027262726272727272727272727272726E3
:1000B00026272727262727272627262727262726D6
:1000C00027272627272627272727262727272727C3
:1000D00026262727262727262627262727262727B7
:1000E00027262727272727272727272627272726A3
:1000F00027272627262726272627262727262726 97
:10010000272727272627272727262627272672783
:10011000262726272627262727262726272672776
:10012000272627272727272726272727262727266 3
:10013000272726262726272627272726272727254
:10014000272726272727272627262727262726245
:10015000262626272726262726272726272727 37
:10016000262727272726272726272726272672 25
:10017000272627262726272627272726272726715
:10018000272627262626272627262627262626 0A
:10019000262627262627262726272627262726F8
:1001A000272627 26FFFFFFFFFFFFFFFFFFFFFFC1
:1001B000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF4F
:1001C000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF3F
:1001D000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF2F
```

```
:1001E000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF1F
:1001F000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0F
:00000001FF
```

Για να εμφανίζουμε το διάγραμμα της βηματικής απόκρισης μεταφέρουμε τις τιμές στο δεκαδικό σύστημα και τις αντιστρέφουμε. Πλέον είναι ανάλογες της ταχύτητας του κινητήρα.Στον άξονα x έχουμε το άθροισμα των μετρηθεισών τιμών. Ετσι, η κάθε τιμή απεικονίζεται στην χρονική στιγμή όπου μετρήθηκε.

(y: 1/periodcount . x: seconds)



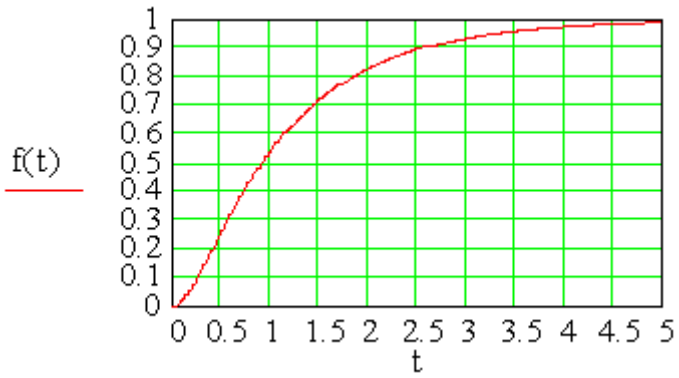
Σχήμα 4.4

Η δημιουργία του διαγράμματος έγινε με την βοήθεια του πργράμματος Microsoft Excel. Τα δεδομένα εισήχθησαν σε αυτό απο το κείμενο που δημιουργήσαμε με το AVR studio, και απεικονίστηκαν σε διάγραμμα ως προς τον χρόνο μετά απο κατάλληλες μετατροπές.

Να σημειώσουμε οτι η πρώτη τιμή των μετρήσεων δεν είναι αξιόπιστη, επειδή δεν έχουμε εξασφαλίσει οτι αφορά τον χρόνο μεταξύ δύο παλμών. Δηλαδή, το οπτικό στοιχείο του αισθητήρα μπορεί πριν την εκκίνησης του πειράματος να βρισκόταν σε σημείο ανάμεσα σε δύο οπές του δίσκου, και γι'αυτόν τον λόγο η μέτρηση να είναι μικρότερη απο ότι θα έπρεπε. Επομένως δεν πρέπει να την λάβουμε υπ'όψιν. Το γεγονός αυτό αναφέρεται επειδή η τιμή αυτή δίνει την αίσθηση της ύπαρξης μίας εκθετικής συνιστώσας με μεγάλο συντελεστή η οποία μηδενίζεται γρήγορα στην βηματική απόκριση.

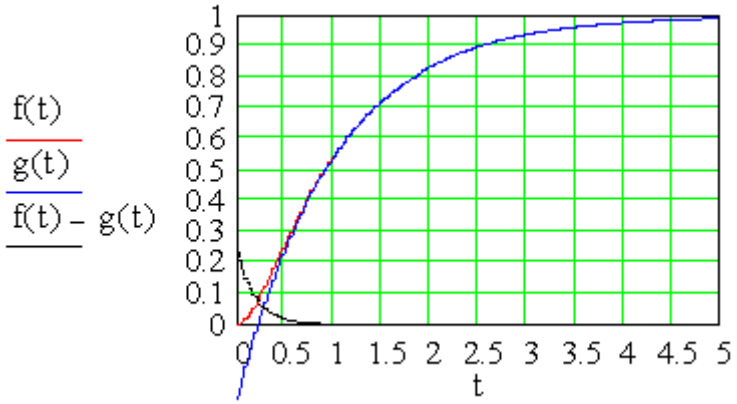
Ας κάνουμε κάποιες επιφανειακές παρατηρήσεις επάνω στην βηματική απόκριση. Οπως έχουμε αναναλύσει σε προηγούμενο κεφάλαιο, ο κινητήρας έχει μοντελοποιηθεί ώς ένα σύστημα δεύτερου βαθμού.

Απο την βηματική απόκριση προκύπτει οτι δεν παρουσιάζονται ταλαντώσεις ή υπερπήδηση. Το γεγονός αυτό οδηγεί στο συμπέρασμα οτι οι πόλοι του συστήματος είναι πραγματικοί. Η βηματική απόκριση σε αυτην την περίπτωση είναι της μορφής:



Σχήμα 4.5

Η οποία περιέχει την σύνθεση δύο φθίνουσων εκθετικών συναρτήσεων που αντιστοιχούν στους δύο πόλους:



Σχήμα 4.6

πηγή: http://www.facstaff.bucknell.edu/mastascu/econtrolhtml/Ident/Ident1.html

εκ των οποίων ο επικρατών πόλος χαρακτηρίζει το μεγαλύτερο μέρος της καμπύλης. Για να βρούμε την μόνιμη κατάσταση κάνουμε το εξής:

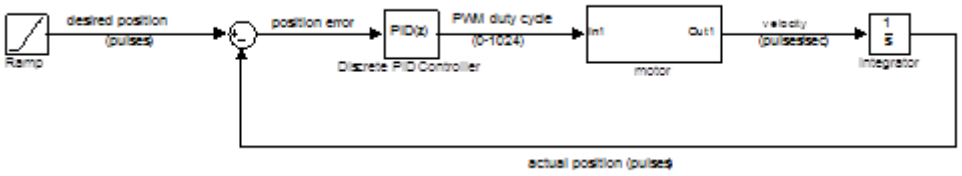
Βλέπουμε οτι αντιστοιχεί σε περίοδο μεταξύ των παλμών διάρκειας

$$27_{16} * \frac{3*32}{8000000} = 39 * \frac{3*32}{8000000} = 468 * 10^{-6} \text{ sec}$$

Επομένως η συχνότητα των παλμών είναι $(468*10^{-6})^{-1}$ =2136,75 Hz στην μόνιμη κατάσταση. Η συχνότητα του κοχλία είναι 2136,75/780 = 2,739 Hz, αφού 780 παλμοί του αισθητήρα αντιστοιχούν σε μία πλήρη περιστροφή του κοχλία.
Η γωνιακή ταχύτητα του κοχλία είναι 2,739*2*π = 17,21 rad/sec

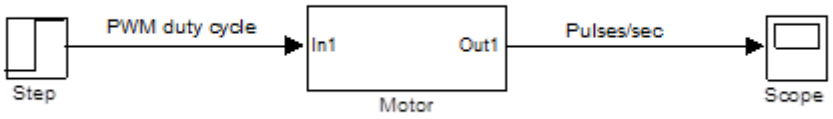
Σε αυτό το σημείο να θυμίσω οτι η πλήρης ισχύς στον κινητήρα δίνεται στα 22Volt, αφού η τάση τροφοδοσία είναι 24 και η γέφυρα παρουσιάζει πτώση τάσης 2 volt. Επομένως, το DC κέρδος του συτήματος είναι 17,21/22 = 0,782

Σε αυτό το σημείο για την απλοποίηση των πράξεων είναι βολικό να θεωρήσουμε ώς μονάδα μέτρησης μετατόπισης το πλήθος παλμών αντι για ακτίνια, ταχύτητας την συχνότητα παλμών αντί για ακτίνια/δευτερόλεπτο και είσοδο στον κινητήρα την τιμή του Duty Cycle (0 έως 1024) αντί για τιμή τάσης. Ετσι, δεν χρειάζεται η προσθήκη κανενός επι πλέον κέρδους ούτε της μείωσης του κινητήρα. Το κλειστό σύστημα πλέον είναι το εξής:



Σχήμα 4.7

,ενώ το ανοιχό σύστημα όπου γίνονται οι μετρήσεις είναι το:



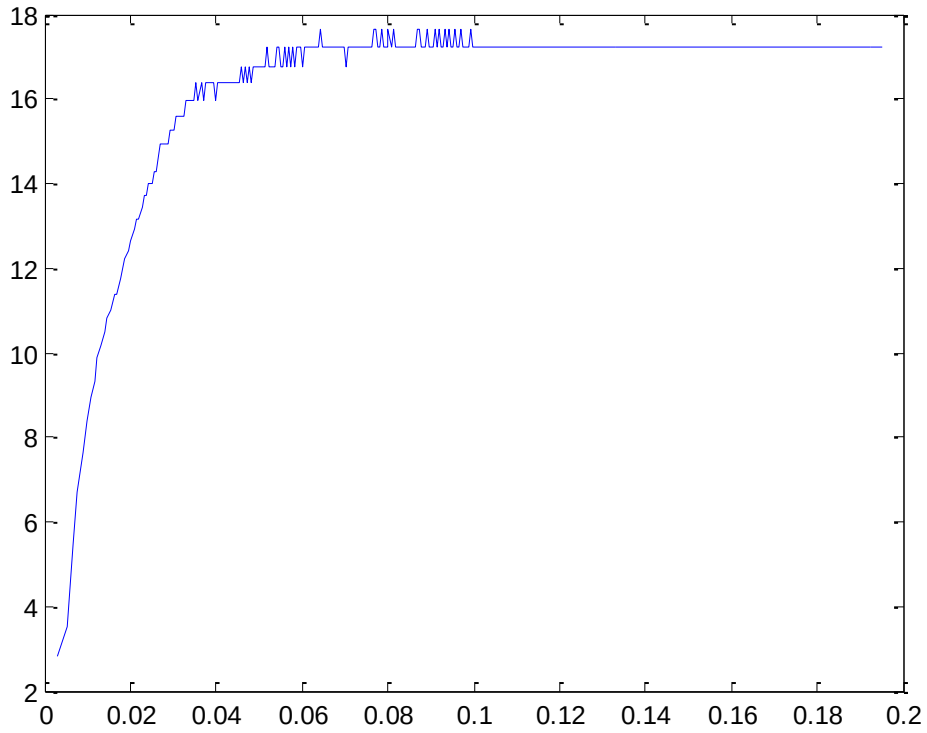
Σχήμα 4.8

Με την σύμβαση αυτή το νέο DC κέρδος του συστήματος υπολογίζεται ώς K=2136,75/1024=2,0866

Οσον αφορά την μεταβατική κατάσταση του κινητήρα, χρειάζεται να υπολογίσουμε τις χρονικές σταθερές.Αυτό μπορεί να γίνει με τον εξής τρόπο:

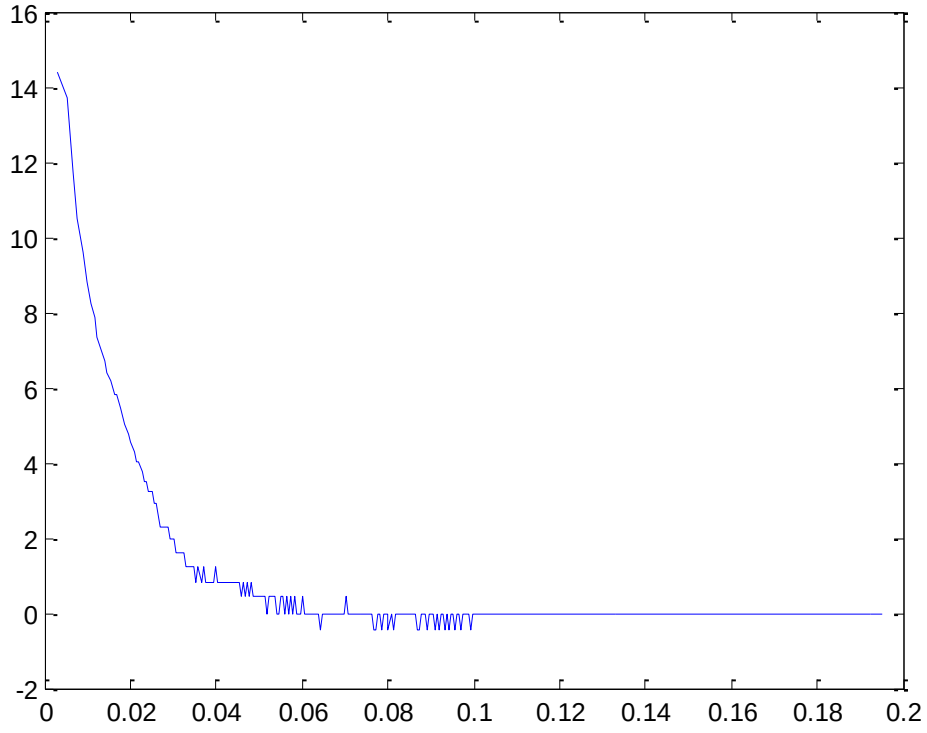
Εαν θέλαμε να προσεγγίσουμε το σύστημα με ένα πρωτοβάθμιο σύστημα θα μπορούσαμε νε μετρήσουμε την χρονική σταθερά μετρώντας τον χρόνο στον οποίο η απόκριση φθάνει στο 63% της μόνιμης κατάστασης. Στην θεώρηση δευτεροβάθμιου συστήματος η διαδικασία είναι πιο πολύπλοκή, καθώς το σύστημα έχει δύο χρονικές σταθερές. Η μέθοδος που ακολουθήσαμε χρησιμοποιεί το λογισμικό MATLAB για ανάλυση των δεδομένων.

Συγκεκριμένα, οι τιμές που είχαν μετατραπεί στο δεκαδικό σύστημα απο το περιβάλλον του Excel μεταφέρθηκαν με την μορφή πίνακα-διανύσματος στο περιβάλλον MATLAB. Οι τιμές αυτές θυμίζω οτι αντιστοιχούν στην διάρκεια μεταξύ των διαδοχικών παλμών. Με άθροισή τους δημιουργήθηκε το διάνυσμα χρόνου, που περιλαμβάνει τις χρονικές στιγμές που έγινε η κάθε μέτρηση. Στην συνέχεια το αρχικό διάνυσμα αντιστράφηκε και πολλαπλασιάστηκε με κατάλληλες σταθερές ωστε να περιέχει πλέον τιμές σε μονάδες rad/sec (γωνιακής ταχύτητας) του κοχλία:



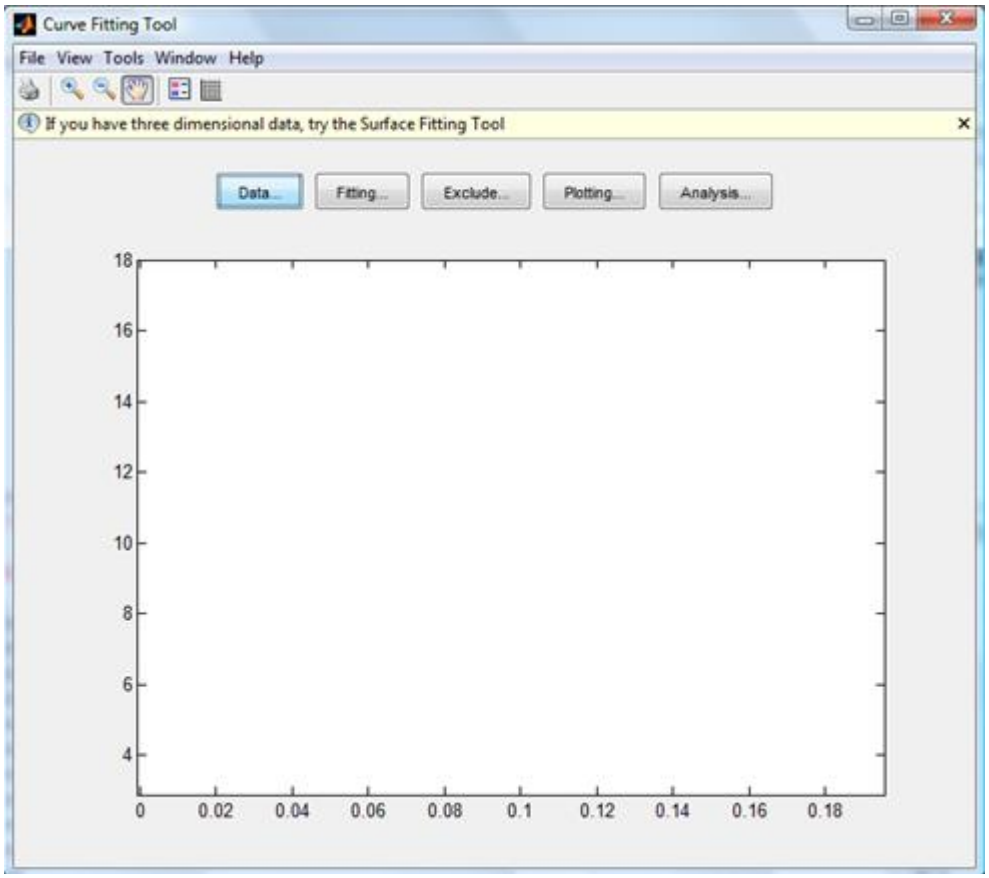
Σχήμα 4.9

Στην συνέχεια, τα δεδομένα αφαιρέθηκαν απο την μόνιμη τιμή της απόκρισης,ώστε να παραμείνουν μόνο οι εκθετικές συναρτήσεις:



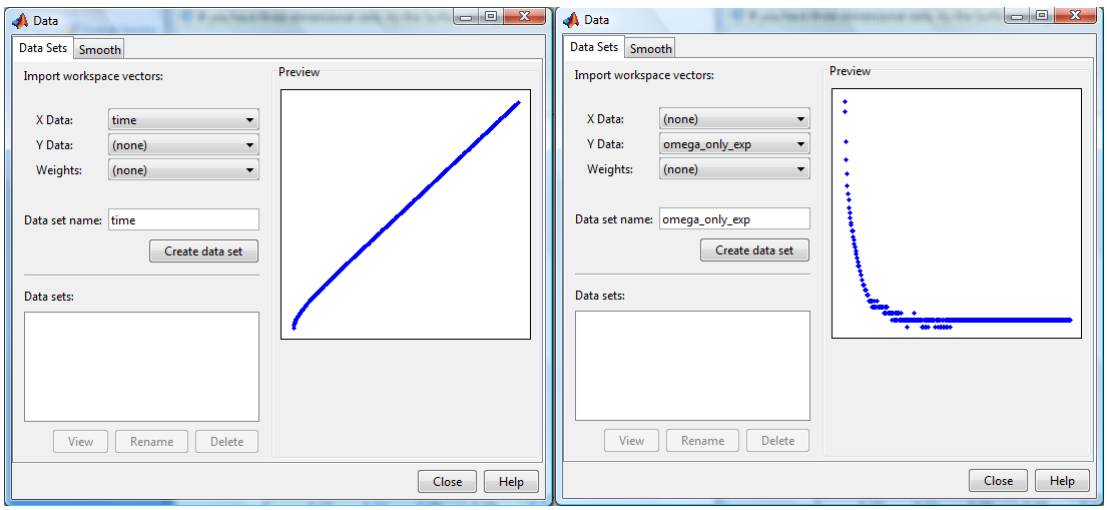
Σχήμα 4.10

Στην συνέχεια χρησιμοποιήθηκε το εργαλείο Curve fitting tool της MATLAB ,με την εντολή cftool;.



Σχήμα 4.11 Screenshot απο Curve fitting tool της MATLAB

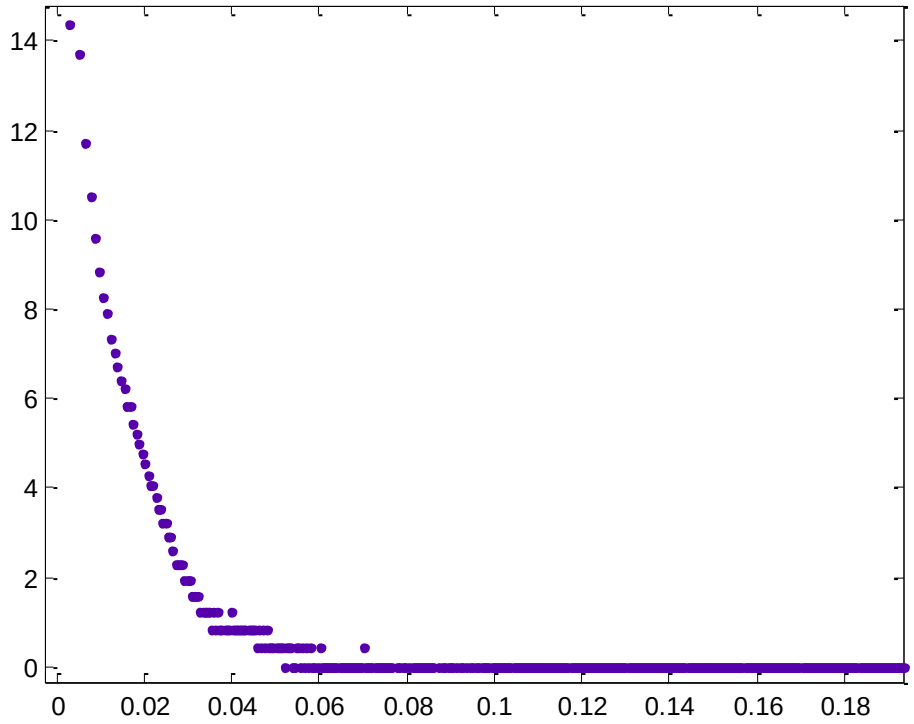
Στο περιβάλλον του εργαλείου εισήχθησαν τα δεδομένα:



Σχήμα 4.12 Εισαγωγή των διανυσμάτων του X και Y αξονα στο curve fitting tool

φωτο: η εισαγωγή των διανυσμάτων του άξονα χ και y αντίστοιχα

Και σχηματίστηκε το Data set:



Σχήμα 4.13

Στην συνέχεια χρησιμοποιηθηκε η μέθοδος προσαρμογής συνάρτησης με την επιλογή εκθετικού τύπου. Εγινα δύο δοκιμές. Η μία αφορά πρωτοβάθμιο σύστημα (ένας πόλος- μία εκθετική συνάρτηση) και η άλλη δευτεροβάθμιο:

Αποτελέσματα για πρωτοβάθμια προσέγγιση:

General model Exp1:

f(x) = a*exp(b*x)

Coefficients (with 95% confidence bounds):

a = 18.88 (18.61, 19.15)

b = -74.29 (-75.31, -73.26)

Goodness of fit:

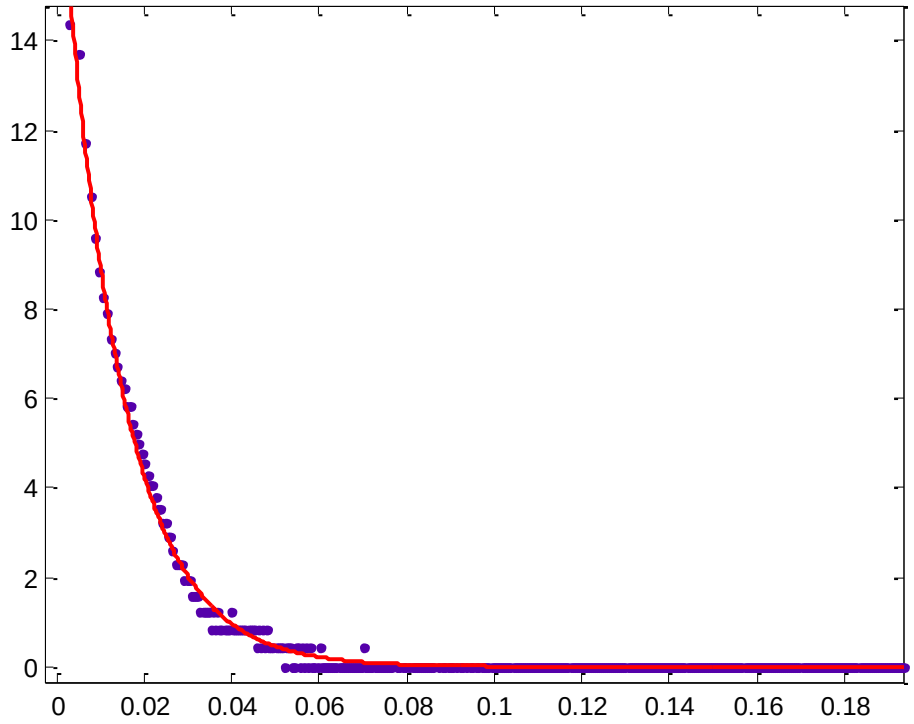
SSE: 10.69

R-square: 0.9925

Adjusted R-square: 0.9925

RMSE: 0.1664

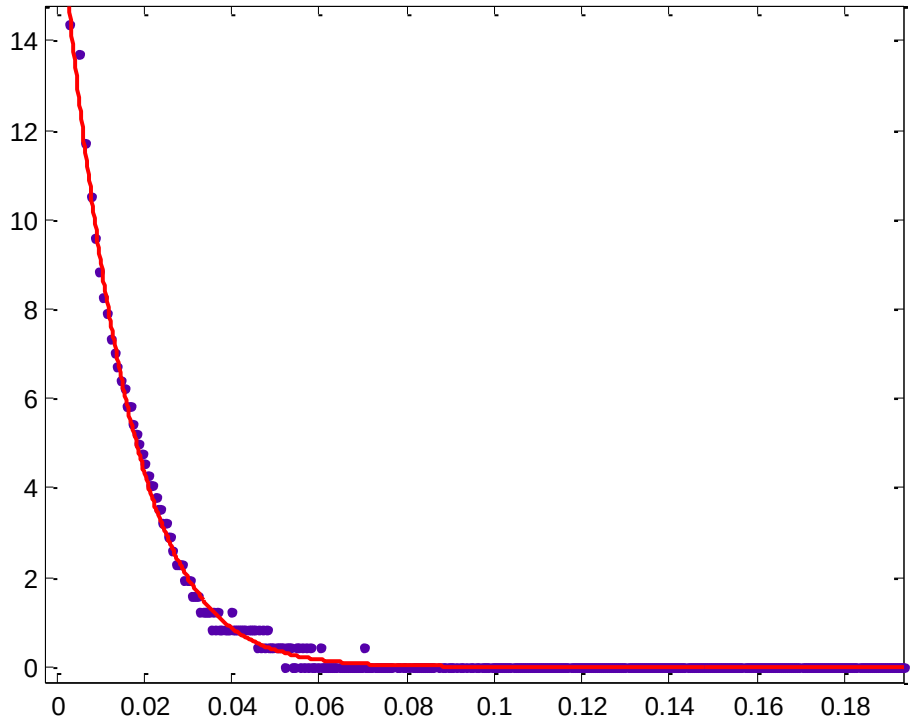
και σχηματικά:



Σχήμα 4.14

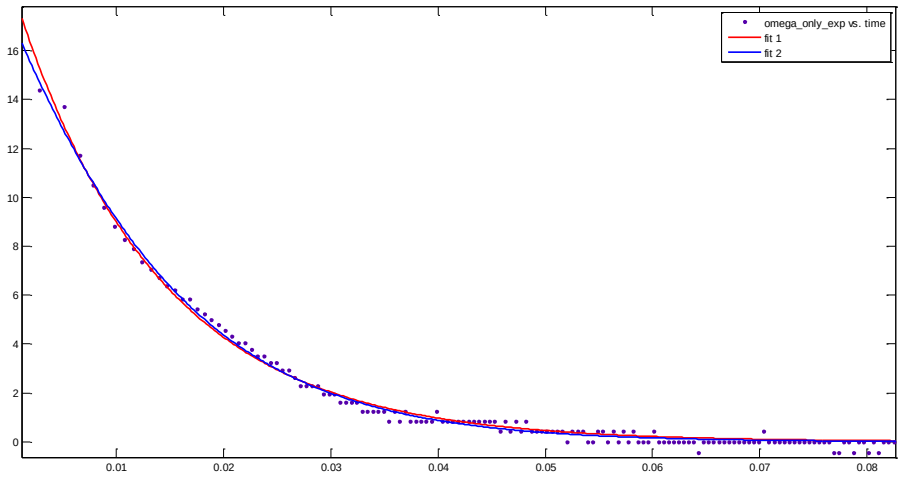
Ενώ για δευτεροβάθμιο σύστημα έχουμε:
General model Exp2:
f(x) = a*exp(b*x) + c*exp(d*x)
Coefficients (with 95% confidence bounds):
a = -3.944e+004 (-3.658e+011, 3.658e+011)
b = -98.9 (-8.305e+004, 8.285e+004)
c = 3.946e+004 (-3.658e+011, 3.658e+011)
d = -98.88 (-8.299e+004, 8.279e+004)

Goodness of fit:
SSE: 9.249
R-square: 0.9935
Adjusted R-square: 0.9934
RMSE: 0.1552
και σχηματικά:



Σχήμα 4.15

Ας δούμε και τις δύο προσεγγίσεις σε ένα γράφημα:



Σχήμα 4.16

Βλέπουμε οτι οι δύο προσεγγίσεις είναι σχεδόν όμοιες, ιδιαίτερα όσο αυξάνεται η τιμή του χρόνου. Η διαφορές τους εντοπίζονται για πολύ χαμηλές τιμές του χρόνου, περιοχή για την οποία δεν μπορούμε να έχουμε μετρήσεις απο τον μΕ. ενδεικτικά ας παρατηρήσουμε οτι οι πρώτη έγκυρη μέτρηση ($2^{η}$) εμφανίζεται σε σημείο που πλέον οι προσεγγίσεις είναι σχεδόν πανομοιότυπες.

Επίσης, αν παρατηρήσουμε τις τιμές των εκθετών της δευτεροβάθμιας προσέγγισης θα δούμε οτι είναι σχεδόν όμοιες και οι συντελεστές τους σχεδόν αντίθετοι. Αν παρατηρήσουμε, δε, τα όρια σφάλματος 5% θα δούμε οτι αυτά είναι πάρα πολύ μεγάλα.

Απο αυτά συμπεραίνουμε οτι η πρωτοβάθμια προσέγγιση είναι ιδιαίτερα ικανοποιητική για το σύστημά μας και πως ο δεύτερος πόλος του συτήματος μας είναι αρκετά ευσταθής αφού η επίδρασή του παύει να είναι εμφανής ήδη απο τις πρώτες τιμές της απόκρισης. Επομένως μπορούμε να υποθέσουμε οτι το σύστημά μας έχει έναν πόλο κοντά στην θέση -74,29 και άλλον έναν πόλο αρκετά πιο αριστερά. Αυτό επιβεβαιώνεται αν απαιτήσουμε απο το πρόγραμμα προσεγγίσεων να τοποθετήσει τον έναν πόλο κοντά στην τιμή -74,29 (απο -80 έως -70):

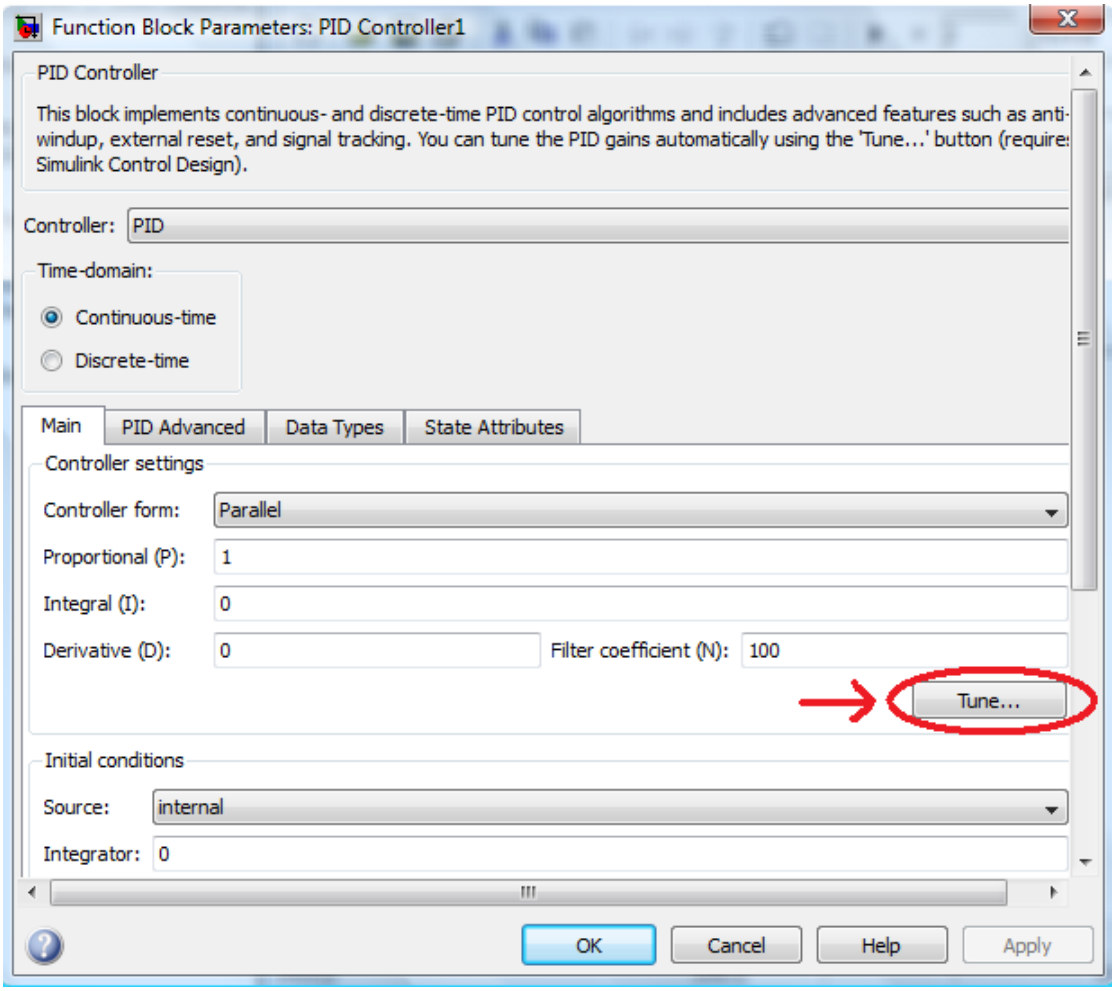
General model Exp2:
f(x) = a*exp(b*x) + c*exp(d*x)
Coefficients (with 95% confidence bounds):
a = 22.46 (21.45, 23.47)
b = -80 (fixed at bound)
c = -5.045 (-5.857, -4.232)
d = -163.6 (-206.3, -120.9)
Goodness of fit:
SSE: 9.524
R-square: 0.9933
Adjusted R-square: 0.9933
RMSE: 0.1573
, όπου βλέπουμε οτι ο δεύτερος πόλος τοποθετήθηκε στην τιμή -163.6.

Χρησιμοποιώντας την πρωτοβάθμια προσέγγιση , με έναν πόλο στην θέση -74,29 και κέρδος DC ίσο με 2,0866 έχουμε την συνάρτηση μεταφοράς: $G(s) = \dfrac{2.0866}{\dfrac{s}{74.29}+1}$

Με την χρήση της συνάρτησης αυτής μπορούμε να προσομοιώσουμε το πλήρες σύστημά μας στο περιβάλλον του προγράμματος Simulink και να ρυθμίσουμε κατάλληλα τον ελεγκτή PID.

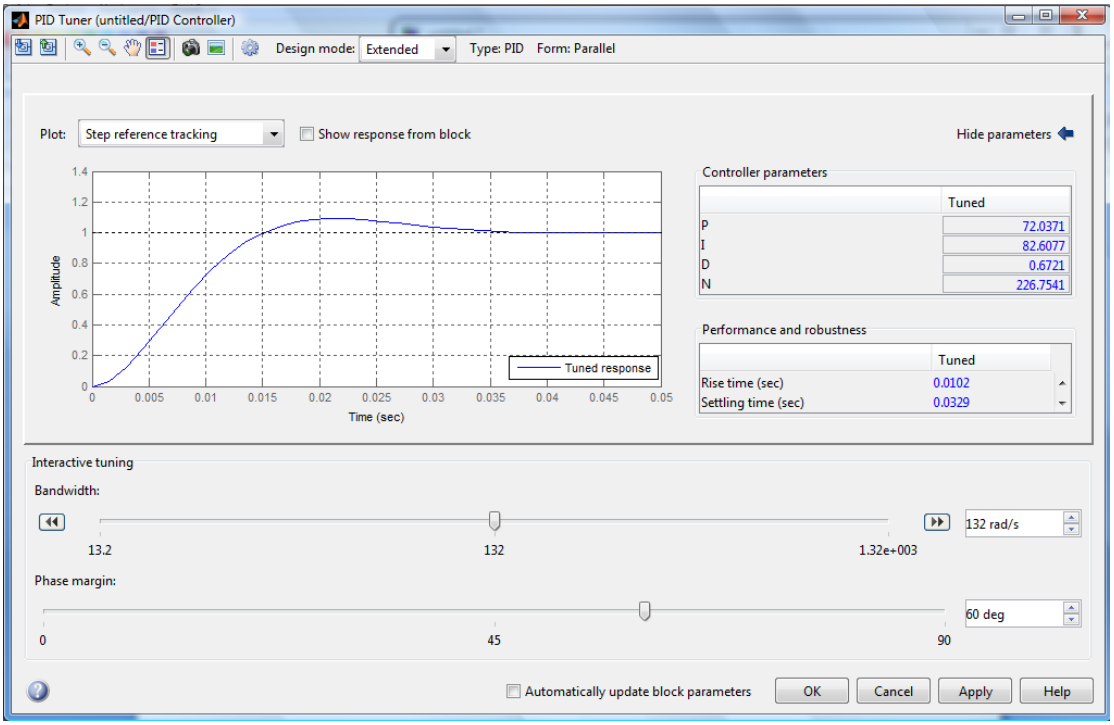
Η προσομοίωση γίνεται χρησιμοποιώντας την διάταξη του σχήματος 4.7 χρησιμοποιώντας στοιχεία συνεχούς χρόνου.

Ο ελεγκτής PID είναι μοντελοποιημένος και παρέχεται σαν βαθμίδα απο το πρόγραμμα. Η ρύθμιση των κερδών του συμφωνα με την εκάστοτε διάταξη και τα επιθυμητά χαρακτηριστικά μπορεί να γίνει μέσω της λειτουργίας *tune* που παρέχεται απο το πρόγραμμα:



Σχήμα 4.17

Με την επιλογή της αυτόματης ρύθμισης, ανοίγεται ένα νέο παράθυρο διαλόγου, όπου βλέπουμε τα χαρακτηριστικά του ρυθμισμένου ελεγκτή και κάνουμε τις επιθυμητές αλλαγές.



Σχήμα 4.18

Ο αλγόριθμος παράγει τα ιδανικά κέρδη K_p, K_i, K_d και τον συντελεστή φίλτρου N.

Ο συντελεστής αυτός αφορά την δράση της παραγώγισης. Η παραγώγιση σε πραγματικά συστήματα δεν γίνεται ποτέ ιδανικά, δηλαδή αφιλτράριστα. Ο λόγος έχει αναφερθεί στην θεωρητική ανάλυση των ελεγκτών PID, όπου είπαμε οτι η διαφόριση επηρεάζεται έντονα απο υψίσυχνο θόρυβο και παρεμβολές. Η αντιμετώπιση του προβλήματος αυτού γίνεται με την εισαγωγή ενός πόλου στο σύστημα παραγώγισης, εκτός απο το μηδενικό. Δηλαδή η συνάρτηση μεταφοράς του κλάδου διαφόρισης του ελεγκτή είναι $\frac{K_d * s}{\frac{s}{N}+1}$ αντί για $K_d * s$ που θα ήταν ιδανικά.

Τα αποτελέσματα, όπως βλέπουμε είναι οι ακόλουθες τιμές:

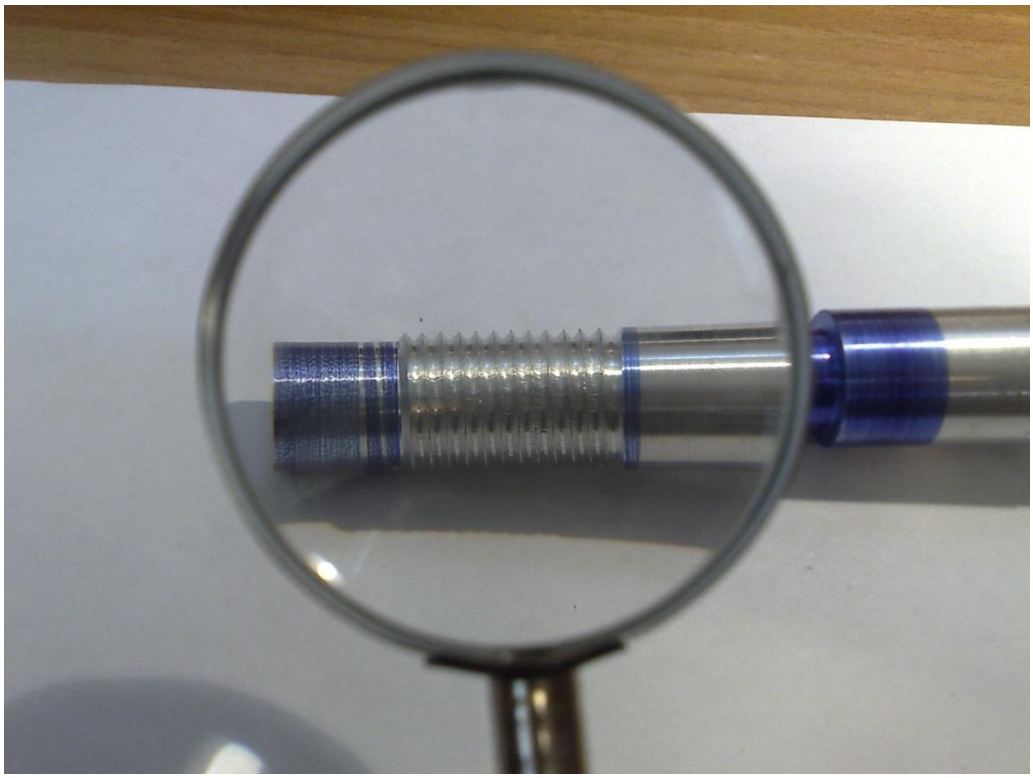
K_p=72.0371
K_i=82.6077
K_d=0.6721
N=226.7541

Τις τιμές αυτές θα τις εισάγουμε στον μικροελεγκτή αφού πρώτα κάνουμε τις κατάλληλες μετατροπές για υλοποίηση σε διακριτό χρόνο. Ετσι, όπως έχουμε ανανύσει σε προηγούμενο κεφάλαιο, στην ρουτίνα pid_init() θα εισάγουμε τις τιμές K_p, K_i*T_s , K_d/T_s στις μεταβλητές P, I, D αντίστοιχα. Θυμίζω οτι T_s είναι η περίοδος δειγματοληψίας, η οποία δηλώνεται και αυτή στην ρουτίνα pid_init(). Το φίλτρο για την διαφόριση στην περίπτωσή μας δεν χρειάζεται να χρησιμοποιηθεί γιατι ο έλεγχος δεν εφαρμόζεται κατ'ευθείαν σε κάποιο αναλογικό σήμα, αλλα σε μία μεταβλητή ακεραίων που διαμορφώνεται απο τον

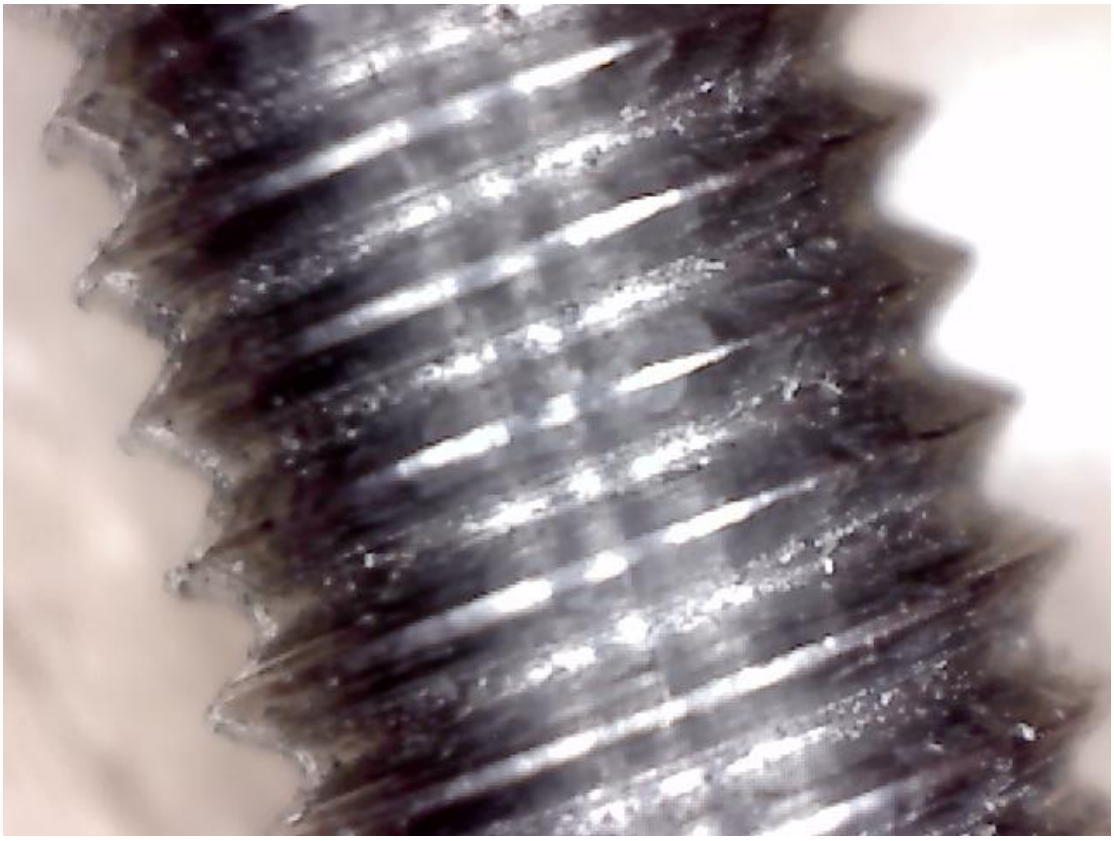
μικροελεγκτή. Τα σήματα των αισθητήρων που παρακολουθούνται για την κατασκευή αυτής της μεταβλητής σφάλματος είναι ουσιαστικά ήδη φιλτραρισμένα απο την διάταξη shmitt trigger. Επειδή το DC κέρδος του φίλτρου $\frac{1}{\frac{s}{N}+1}$ είναι ίσο με 1, μπορούμε απλώς να το αγνοήσουμε χωρίς να χρειάζεται κάποια αλλαγή στο κέρδος K_d.

4.4 Δοκιμές – Ποιότητα σπειρώματος

Το σύστημα δοκιμάστηκε στην κοπή διαφόρων σπειρωμάτων. Χρησιμοποιήθηκαν κυρίως σπειρώματα με τυποποιημένα βήματα σύμφωνα με το πρότυπο ISO Metric threads. σε διάφορα υλικά. Στην συνέχεια τα σπειρώματα συγκρίθηκαν με έτοιμα σπειρώματα του εμπορίου. Στην συνέχεια παρουσιάζονται κάποιες φωτογραφίες σε μεγέθυνση .



Σχήμα 4.19 Σπείρωμα βήματος 2.5mm σε αλουμίνιο



Σχήμα 4.20 Σπείρωμα βήματος 1mm (M6)

Σχήμα 4.21 Συγκριση σπειρώματος (αριστερά) με βίδα M6 εμπορίου (δεξιά)

Σχήμα 4.22 Εποπτεία σε μεγαλύτερη μεγέθυνση

Προτάσεις για περαιτέρω ανάπτυξη

Τελειώνοντας το κύριο σώμα της παρούσας εργασίας θα αναφερθούν ιδέες και προτάσεις για την περαιτέρω ανάπτυξη του συστήματος της εργασίας.

Η ιδέα της εργασίας αφορά ένα σύστημα αυτοματισμού ενός κινητήρα ο οποίος οδηγείται σε κίνηση σύμφωνα με κάποιον αισθητήρα αναφοράς και κάποια επιθυμητή σχέση μετάδοσης. Το σύστημα παρέχει την δυνατότητα επιλογής οποιασδήποτε ρητής σχέσης μετάδοσης απο τον χρήστη.

Μία ενδιαφέρουσα και χρήσιμη προοπτική ανάπτυξης του συστήματος θα ήταν η δοθείσα σχέση μετάδοσης να αλλάζει δυναμικά εν ώρα λειτουργίας. Κατι τέτοιο θα προυπέθετε την επικοινωνία του συστήματος με κάποιον υπολογιστή, ο οποίος θα παρέχει τις ανάλογες τιμές για τις μεταβλητές ελέγχου. Κατ' αυτόν τον τρόπο το σύστημα θα μπορούσε να ακολουθεί πιο πολύπλοκα προφίλ κίνησης απο τα απλά γραμμικά προφίλ που ακολουθεί η παρούσα υλοποίηση. Επι πλέον, ο ελέγχων υπολογιστής θα μπορούσε να συντονίζει μεταξύ τους περισσότερα απο ένα παρόμοια συστήματα, ωστε πλέον να οδηγείται ένα σύστημα αυτοματισμού πολλών βαθμών ελευθερίας.

Η εφαρμογή μίας τέτοιας διαφοροποίησης δεν απέχει ιδιαίτερα απο την τρέχουσα δομή του αλγορίθμου του συστήματος. Θυμίζω οτι η μεταβλητή σφάλματος θέσης για τον έλεγκτή του κινητήρα διαμορφώνεται σε κάθε παλμό του αισθητήρα αναφοράς με την πρόσθεση του επόμενου όρου της ακολουθίας B(n). Η ακολουθία αυτή βρίσκεται στην μνήμη του μΕ. Στο νέο υπο συζήτηση σύστημα αρκεί οι τιμές αυτές να μην αναζητώνται στην εσωτερική μνήμη, αλλα σε κάποια θύρα εισόδου του μΕ. Στην θύρα αυτή στέλνει τα δεδομένα ο ελέγχων υπολογιστής. Ο υπολογιστής αυτός πρέπει επίσης να λαμβάνει την παλμοσειρά του αισθητήρα αναφοράς, ωστε να χρονίζεται σωστά η αποστολή των δεδομένων. Απο αυτό το σημείο ο έλεγχος επαφύεται αποκλειστικά στο λογισμικό που εκτελείται στον ελέγχοντα υπολογιστή. Απαραίτητη προυπόθεση είναι επίσης οι αισθητήρες να παρέχουν και την πληροφορία της φοράς περιστροφής των αξόνων στο σύστημα, κάτι που δεν κρίθηκε αναγκαίο στην παρούσα υλοποίηση. Η δομή και η αποκωδικοποίηση τέτοιων αισθητήρων έχουν αναλυθεί σε προηγούμενο κεφάλαιο.

Μια άλλη πρόταση είναι η χρήση αλγορίθμου αυτόματης ρύθμισης του ελεγκτή PID. Η διαδικασία αυτή μπορεί να εκμεταλλεύεται την αναγνώριση συτήματος που πραγματοποιείται ήδη απο ειδική ρουτίνα και να συνεχίζει με αναλυτικές μεθόδους, ή να πραγματοποιούνται δοκιμές με διάφορες τιμές των κερδών του ελεγκτή μέχρις ότου να προκύψει ικανοποιητική λύση. Στην υλοποίηση αυτή θα μπορούσε να χρησιμοποιείται γενετικός αλγόριθμος.

Τέλος, πέρα απο την εφαρμογή στον τόρνο που επιλέχθηκε σε αυτήν την εργασία, υπάρχει πληθώρα άλλων εφαρμογών όπου θα ήταν χρήσιμο το σύστημα ηλεκτρονικού γραναζώματος. Τέτοιες εφαρμογές είναι η διαγραφή τοξοειδών τροχιών απο αυτοκινούμενες μηχανές με τροχούς, όπου οι εξωτερικοί τροχοί χρειάζεται να διατηρούν συγκεκριμένη σχέση ταχύτητας με τους εσωτερικούς, η οδήγηση ιμάντων μεταφοράς απο ράουλα διαφορετικής διαμέτρου, όπου χρειάζεται να ρυθμίζεται η τάση των ιμάντων,κ.α.

Βιβλιογραφία

Ακαδημαϊκά συγγράμματα:

-Richard C. Dorf, Robert H. Bishop *Συγχρονα Συστήματα αυτόματου ελέγχου*, Εκδόσεις Τζιόλα
-Σ. Τζαφέστας *Αυτόματος έλεγχος γραμμικών και μη γραμμικών συστημάτων συνεχούς και διακριτού χρόνου* Τόμος 1
-Ι.Ν. Αβαριτσιώτης *Τεχνολογία πολυψηφιδικών πακέτων* Αθήνα 1997 Εσωτερικές εκδόσεις Ε.Μ.Π.
-M. Morris Mano, Michael D. Ciletti *Ψηφιακή σχεδίαση* Εκδόσεις Παπασωτηρίου
-Jacob Millman, Χρήστος Χαλκιάς *Ωλοκληρωμένη Ηλεκτρονική,* εκδόσεις Συμμετρία
-Κ.Ζ. Πεκμεστζή *Συστήματα μικροϋπολογιστών* Εκδόσεις Συμετρία
-Κ.Ζ. Πεκμεστζή *Μικροελεγκτές AVR και PIC*, Εσωτερικές εκδόσεις Ε.Μ.Π.
-Robert Sedgewick *Αλγόριθμοι σε C* Εκδόσεις Κλειδάριθμος
-Ν. Μισυρλής *Δομές δεδομένων με C*
-Στέφανος Ν. Μανιας, Αθανάσιος Καλετσάνος *Βιομηχανικά ηλεκτρονικά*, εκδόσεις Συμεών

Online εγκυκλοπαίδεια Wikipedia – Αναφορά με τίτλο άρθρου:

DC motor
Rotary encoder
Gray code
Linear regulator
Switched mode power supply
electronic filter
DC to DC converter
Pulse width modulation
Voltage regulator
PID controller
Ziegler Nichols method
Flyback diode
lathe(metal)
Schmitt trigger
opto-isolator
Transistor-Transistor logic
HD44780 character LCD
In System Programming
Noise (electronics)
Junction temperature
Backlash (engineering)
motion control
Continued fraction
Motor controller

H-bridge
Fourier series
ISO metric screw thread
Forced convection
Heat sink
Thermal resistance
Bresenham's line algorithm
malloc

Online αρθρα και εργασίες

http://machinedesign.com/article/motor-follower-electronic-gearing-1115
http://lorien.ncl.ac.uk/ming/digicont/digimath/dpid1.htm
http://www.ecircuitcenter.com/circuits/pid1/pid1.htm
http://arri.uta.edu/acs/jyotirmay/EE4343/Labs_Projects/pidcontrollers.pdf
http://www.hitex.com/fileadmin/img/download/Basic_DC_Motor_Speed_PID_Control_With_The_Infineon_C167_Family.pdf
http://robots.freehostia.com/Heatsinks/HeatsinksBody.html
http://www.compliance-club.com/archive/old_archive/021132.htm

Παράρτημα Α': Ο κώδικας του συστήματος

Α.1 Ρουτίνες χειρισμού οθόνης

Η οθόνη που χρησιμοποιήθηκε στην υλοποίηση του συστήματος περιλαμβάνει μνήμη και μικροΕπεξεργαστή προγραμματισμένα ωστε να επικοινωνεί με άλλα συστήματα σύμφωνα με το πρωτόκολλο HD44780U. Οι ρουτίνες σε γλώσσα C που χρησιμοποιήθηκαν βρέθηκαν αυτούσιες στον ιστότοπο http://jump.to/fleury. Η εισαγωγή τους στον κώδικα του συστήματος έγινε με την χρήση ενός αρχείου header και ενός αρχείου κώδικα, τα οποία παρατίθενται παρακάτω:

Αρχείο Header lcd.h:

```
#ifndef LCD_H
#define LCD_H
/************************************************************************
 Title  :   C include file for the HD44780U LCD library (lcd.c)
 Author:    Peter Fleury <pfleury@gmx.ch>  http://jump.to/fleury
 File:      $Id: lcd.h,v 1.13.2.2 2006/01/30 19:51:33 peter Exp $
 Software:  AVR-GCC 3.3
 Hardware:  any AVR device, memory mapped mode only for AT90S4414/8515/Mega
************************************************************************/

/**
 @defgroup pfleury_lcd LCD library
 @code #include <lcd.h> @endcode
```

```
 @brief Basic routines for interfacing a HD44780U-based text LCD display

 Originally based on Volker Oth's LCD library,
 changed lcd_init(), added additional constants for lcd_command(),
 added 4-bit I/O mode, improved and optimized code.

 Library can be operated in memory mapped mode (LCD_IO_MODE=0) or in
 4-bit IO port mode (LCD_IO_MODE=1). 8-bit IO port mode not supported.

 Memory mapped mode compatible with Kanda STK200, but supports also
 generation of R/W signal through A8 address line.

 @author Peter Fleury pfleury@gmx.ch http://jump.to/fleury

 @see The chapter <a href="http://homepage.sunrise.ch/mysunrise/peterfleury/avr-lcd44780.html"
target="_blank">Interfacing a HD44780 Based LCD to an AVR</a>
      on my home page.

*/

/*@{*/

#if (__GNUC__ * 100 + __GNUC_MINOR__) < 303
#error "This library requires AVR-GCC 3.3 or later, update to newer AVR-GCC compiler !"
#endif

#include <inttypes.h>
#include <avr/pgmspace.h>

/**
 *  @name  Definitions for MCU Clock Frequency
 *  Adapt the MCU clock frequency in Hz to your target.
 */
#define XTAL 4000000              /**< clock frequency in Hz, used to calculate delay timer */


/**
 * @name  Definition for LCD controller type
 * Use 0 for HD44780 controller, change to 1 for displays with KS0073 controller.
 */
#define LCD_CONTROLLER_KS0073 0  /**< Use 0 for HD44780 controller, 1 for KS0073 controller */

/**
 *  @name  Definitions for Display Size
 *  Change these definitions to adapt setting to your display
 */
#define LCD_LINES           2     /**< number of visible lines of the display */
#define LCD_DISP_LENGTH    16     /**< visibles characters per line of the display */
#define LCD_LINE_LENGTH  0x40     /**< internal line length of the display    */
#define LCD_START_LINE1  0x00     /**< DDRAM address of first char of line 1 */
#define LCD_START_LINE2  0x40     /**< DDRAM address of first char of line 2 */
#define LCD_START_LINE3  0x14     /**< DDRAM address of first char of line 3 */
#define LCD_START_LINE4  0x54     /**< DDRAM address of first char of line 4 */
#define LCD_WRAP_LINES      0     /**< 0: no wrap, 1: wrap at end of visibile line */


#define LCD_IO_MODE      1         /**< 0: memory mapped mode, 1: IO port mode */
#if LCD_IO_MODE
/**
 *  @name Definitions for 4-bit IO mode
 *  Change LCD_PORT if you want to use a different port for the LCD pins.
 *
 *  The four LCD data lines and the three control lines RS, RW, E can be on the
 *  same port or on different ports.
 *  Change LCD_RS_PORT, LCD_RW_PORT, LCD_E_PORT if you want the control lines on
 *  different ports.
 *
 *  Normally the four data lines should be mapped to bit 0..3 on one port, but it
 *  is possible to connect these data lines in different order or even on different
 *  ports by adapting the LCD_DATAx_PORT and LCD_DATAx_PIN definitions.
 *
 */
#define LCD_PORT         PORTA        /**< port for the LCD lines   */
#define LCD_DATA0_PORT   LCD_PORT     /**< port for 4bit data bit 0 */
#define LCD_DATA1_PORT   LCD_PORT     /**< port for 4bit data bit 1 */
```

```
#define LCD_DATA2_PORT   LCD_PORT     /**< port for 4bit data bit 2 */
#define LCD_DATA3_PORT   LCD_PORT     /**< port for 4bit data bit 3 */
#define LCD_DATA0_PIN    0            /**< pin for 4bit data bit 0  */
#define LCD_DATA1_PIN    1            /**< pin for 4bit data bit 1  */
#define LCD_DATA2_PIN    2            /**< pin for 4bit data bit 2  */
#define LCD_DATA3_PIN    3            /**< pin for 4bit data bit 3  */
#define LCD_RS_PORT      LCD_PORT     /**< port for RS line         */
#define LCD_RS_PIN       4            /**< pin  for RS line         */
#define LCD_RW_PORT      LCD_PORT     /**< port for RW line         */
#define LCD_RW_PIN       5            /**< pin  for RW line         */
#define LCD_E_PORT       LCD_PORT     /**< port for Enable line     */
#define LCD_E_PIN        6            /**< pin  for Enable line     */

#elif defined(__AVR_AT90S4414__) || defined(__AVR_AT90S8515__) || defined(__AVR_ATmega64__) || \
      defined(__AVR_ATmega8515__)|| defined(__AVR_ATmega103__) || defined(__AVR_ATmega128__) || \
      defined(__AVR_ATmega161__) || defined(__AVR_ATmega162__)
/*
 *  memory mapped mode is only supported when the device has an external data memory interface
 */
#define LCD_IO_DATA      0xC000    /* A15=E=1, A14=RS=1                */
#define LCD_IO_FUNCTION  0x8000    /* A15=E=1, A14=RS=0                */
#define LCD_IO_READ      0x0100    /* A8 =R/W=1 (R/W: 1=Read, 0=Write   */
#else
#error "external data memory interface not available for this device, use 4-bit IO port mode"

#endif


/**
 *  @name Definitions for LCD command instructions
 *  The constants define the various LCD controller instructions which can be passed to the
 *  function lcd_command(), see HD44780 data sheet for a complete description.
 */

/* instruction register bit positions, see HD44780U data sheet */
#define LCD_CLR               0      /* DB0: clear display                */
#define LCD_HOME              1      /* DB1: return to home position      */
#define LCD_ENTRY_MODE        2      /* DB2: set entry mode               */
#define LCD_ENTRY_INC         1      /*   DB1: 1=increment, 0=decrement   */
#define LCD_ENTRY_SHIFT       0      /*   DB2: 1=display shift on         */
#define LCD_ON                3      /* DB3: turn lcd/cursor on           */
#define LCD_ON_DISPLAY        2      /*   DB2: turn display on            */
#define LCD_ON_CURSOR         1      /*   DB1: turn cursor on             */
#define LCD_ON_BLINK          0      /*     DB0: blinking cursor ?        */
#define LCD_MOVE              4      /* DB4: move cursor/display          */
#define LCD_MOVE_DISP         3      /*   DB3: move display (0-> cursor) ?  */
#define LCD_MOVE_RIGHT        2      /*   DB2: move right (0-> left) ?    */
#define LCD_FUNCTION          5      /* DB5: function set                 */
#define LCD_FUNCTION_8BIT     4      /*   DB4: set 8BIT mode (0->4BIT mode) */
#define LCD_FUNCTION_2LINES   3      /*   DB3: two lines (0->one line)    */
#define LCD_FUNCTION_10DOTS   2      /*   DB2: 5x10 font (0->5x7 font)    */
#define LCD_CGRAM             6      /* DB6: set CG RAM address           */
#define LCD_DDRAM             7      /* DB7: set DD RAM address           */
#define LCD_BUSY              7      /* DB7: LCD is busy                  */

/* set entry mode: display shift on/off, dec/inc cursor move direction */
#define LCD_ENTRY_DEC            0x04   /* display shift off, dec cursor move dir */
#define LCD_ENTRY_DEC_SHIFT      0x05   /* display shift on,  dec cursor move dir */
#define LCD_ENTRY_INC_           0x06   /* display shift off, inc cursor move dir */
#define LCD_ENTRY_INC_SHIFT      0x07   /* display shift on,  inc cursor move dir */

/* display on/off, cursor on/off, blinking char at cursor position */
#define LCD_DISP_OFF             0x08   /* display off                            */
#define LCD_DISP_ON              0x0C   /* display on, cursor off                 */
#define LCD_DISP_ON_BLINK        0x0D   /* display on, cursor off, blink char     */
#define LCD_DISP_ON_CURSOR       0x0E   /* display on, cursor on                  */
#define LCD_DISP_ON_CURSOR_BLINK 0x0F   /* display on, cursor on, blink char      */

/* move cursor/shift display */
#define LCD_MOVE_CURSOR_LEFT     0x10   /* move cursor left  (decrement)          */
#define LCD_MOVE_CURSOR_RIGHT    0x14   /* move cursor right (increment)          */
#define LCD_MOVE_DISP_LEFT       0x18   /* shift display left                     */
#define LCD_MOVE_DISP_RIGHT      0x1C   /* shift display right                    */

/* function set: set interface data length and number of display lines */
#define LCD_FUNCTION_4BIT_1LINE  0x20   /* 4-bit interface, single line, 5x7 dots */
```

```
#define LCD_FUNCTION_4BIT_2LINES 0x28   /* 4-bit interface, dual line,   5x7 dots */
#define LCD_FUNCTION_8BIT_1LINE  0x30   /* 8-bit interface, single line, 5x7 dots */
#define LCD_FUNCTION_8BIT_2LINES 0x38   /* 8-bit interface, dual line,   5x7 dots */


#define LCD_MODE_DEFAULT     ((1<<LCD_ENTRY_MODE) | (1<<LCD_ENTRY_INC) )



/**
 *  @name Functions
 */


/**
 @brief    Initialize display and select type of cursor
 @param    dispAttr \b LCD_DISP_OFF display off\n
                    \b LCD_DISP_ON display on, cursor off\n
                    \b LCD_DISP_ON_CURSOR display on, cursor on\n
                    \b LCD_DISP_ON_CURSOR_BLINK display on, cursor on flashing
 @return  none
*/
extern void lcd_init(uint8_t dispAttr);


/**
 @brief    Clear display and set cursor to home position
 @param    void
 @return   none
*/
extern void lcd_clrscr(void);


/**
 @brief    Set cursor to home position
 @param    void
 @return   none
*/
extern void lcd_home(void);


/**
 @brief    Set cursor to specified position

 @param    x horizontal position\n (0: left most position)
 @param    y vertical position\n   (0: first line)
 @return   none
*/
extern void lcd_gotoxy(uint8_t x, uint8_t y);


/**
 @brief    Display character at current cursor position
 @param    c character to be displayed
 @return   none
*/
extern void lcd_putc(char c);


/**
 @brief    Display string without auto linefeed
 @param    s string to be displayed
 @return   none
*/
extern void lcd_puts(const char *s);


/**
 @brief    Display string from program memory without auto linefeed
 @param    s string from program memory be be displayed
 @return   none
 @see      lcd_puts_P
*/
extern void lcd_puts_p(const char *progmem_s);
```

```
/**
 @brief    Send LCD controller instruction command
 @param    cmd instruction to send to LCD controller, see HD44780 data sheet
 @return   none
*/
extern void lcd_command(uint8_t cmd);


/**
 @brief    Send data byte to LCD controller

 Similar to lcd_putc(), but without interpreting LF
 @param    data byte to send to LCD controller, see HD44780 data sheet
 @return   none
*/
extern void lcd_data(uint8_t data);


/**
 @brief macros for automatically storing string constant in program memory
*/
#define lcd_puts_P(__s)         lcd_puts_p(PSTR(__s))

/*@}*/
#endif //LCD_H
```

Αρχείο κώδικα lcd.c:

```
/*************************************************************************
 Title  :   HD44780U LCD library
 Author:    Peter Fleury <pfleury@gmx.ch>  http://jump.to/fleury
 File:      $Id: lcd.c,v 1.14.2.1 2006/01/29 12:16:41 peter Exp $
 Software:  AVR-GCC 3.3
 Target:    any AVR device, memory mapped mode only for AT90S4414/8515/Mega

 DESCRIPTION
       Basic routines for interfacing a HD44780U-based text lcd display

       Originally based on Volker Oth's lcd library,
       changed lcd_init(), added additional constants for lcd_command(),
       added 4-bit I/O mode, improved and optimized code.

       Library can be operated in memory mapped mode (LCD_IO_MODE=0) or in
       4-bit IO port mode (LCD_IO_MODE=1). 8-bit IO port mode not supported.

       Memory mapped mode compatible with Kanda STK200, but supports also
       generation of R/W signal through A8 address line.

 USAGE
       See the C include lcd.h file for a description of each function

*************************************************************************/
#include <inttypes.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include "lcd.h"



/*
** constants/macros
*/
#define DDR(x) (*(&x - 1))      /* address of data direction register of port x */
#if defined(__AVR_ATmega64__) || defined(__AVR_ATmega128__)
    /* on ATmega64/128 PINF is on port 0x00 and not 0x60 */
    #define PIN(x) ( &PORTF==&(x) ? _SFR_IO8(0x00) : (*(&x - 2)) )
#else
    #define PIN(x) (*(&x - 2))    /* address of input register of port x          */
#endif
```

```
#if LCD_IO_MODE
#define lcd_e_delay()   __asm__ __volatile__( "rjmp 1f\n 1:" );
#define lcd_e_high()    LCD_E_PORT  |=  _BV(LCD_E_PIN);
#define lcd_e_low()     LCD_E_PORT  &= ~_BV(LCD_E_PIN);
#define lcd_e_toggle()  toggle_e()
#define lcd_rw_high()   LCD_RW_PORT |=  _BV(LCD_RW_PIN)
#define lcd_rw_low()    LCD_RW_PORT &= ~_BV(LCD_RW_PIN)
#define lcd_rs_high()   LCD_RS_PORT |=  _BV(LCD_RS_PIN)
#define lcd_rs_low()    LCD_RS_PORT &= ~_BV(LCD_RS_PIN)
#endif

#if LCD_IO_MODE
#if LCD_LINES==1
#define LCD_FUNCTION_DEFAULT    LCD_FUNCTION_4BIT_1LINE
#else
#define LCD_FUNCTION_DEFAULT    LCD_FUNCTION_4BIT_2LINES
#endif
#else
#if LCD_LINES==1
#define LCD_FUNCTION_DEFAULT    LCD_FUNCTION_8BIT_1LINE
#else
#define LCD_FUNCTION_DEFAULT    LCD_FUNCTION_8BIT_2LINES
#endif
#endif

#if LCD_CONTROLLER_KS0073
#if LCD_LINES==4

#define KS0073_EXTENDED_FUNCTION_REGISTER_ON  0x24   /* |0|010|0100 4-bit mode extension-bit RE =
1 */
#define KS0073_EXTENDED_FUNCTION_REGISTER_OFF 0x20   /* |0|000|1001 4 lines mode */
#define KS0073_4LINES_MODE                    0x09   /* |0|001|0000 4-bit mode, extension-bit RE =
0 */

#endif
#endif

/*
** function prototypes
*/
#if LCD_IO_MODE
static void toggle_e(void);
#endif

/*
** local functions
*/



/*************************************************************************
 delay loop for small accurate delays: 16-bit counter, 4 cycles/loop
*************************************************************************/
static inline void _delayFourCycles(unsigned int __count)
{
    if ( __count == 0 )
        __asm__ __volatile__( "rjmp 1f\n 1:" );    // 2 cycles
    else
        __asm__ __volatile__ (
            "1: sbiw %0,1" "\n\t"
            "brne 1b"                              // 4 cycles/loop
            : "=w" (__count)
            : "0" (__count)
           );
}


/*************************************************************************
delay for a minimum of <us> microseconds
the number of loops is calculated at compile-time from MCU clock frequency
*************************************************************************/
#define delay(us)  _delayFourCycles( ( ( 1*(XTAL/4000) )*us)/1000 )


#if LCD_IO_MODE
/* toggle Enable Pin to initiate write */
```

```
static void toggle_e(void)
{
    lcd_e_high();
    lcd_e_delay();
    lcd_e_low();
}
#endif


/************************************************************************
Low-level function to write byte to LCD controller
Input:    data   byte to write to LCD
          rs     1: write data
                 0: write instruction
Returns:  none
************************************************************************/
#if LCD_IO_MODE
static void lcd_write(uint8_t data,uint8_t rs)
{
    unsigned char dataBits ;


    if (rs) {   /* write data        (RS=1, RW=0) */
       lcd_rs_high();
    } else {    /* write instruction (RS=0, RW=0) */
       lcd_rs_low();
    }
    lcd_rw_low();

    if ( ( &LCD_DATA0_PORT == &LCD_DATA1_PORT) && ( &LCD_DATA1_PORT == &LCD_DATA2_PORT ) && (
&LCD_DATA2_PORT == &LCD_DATA3_PORT )
      && (LCD_DATA0_PIN == 0) && (LCD_DATA1_PIN == 1) && (LCD_DATA2_PIN == 2) && (LCD_DATA3_PIN ==
3) )
    {
        /* configure data pins as output */
        DDR(LCD_DATA0_PORT) |= 0x0F;

        /* output high nibble first */
        dataBits = LCD_DATA0_PORT & 0xF0;
        LCD_DATA0_PORT = dataBits |((data>>4)&0x0F);
        lcd_e_toggle();

        /* output low nibble */
        LCD_DATA0_PORT = dataBits | (data&0x0F);
        lcd_e_toggle();

        /* all data pins high (inactive) */
        LCD_DATA0_PORT = dataBits | 0x0F;
    }
    else
    {
        /* configure data pins as output */
        DDR(LCD_DATA0_PORT) |= _BV(LCD_DATA0_PIN);
        DDR(LCD_DATA1_PORT) |= _BV(LCD_DATA1_PIN);
        DDR(LCD_DATA2_PORT) |= _BV(LCD_DATA2_PIN);
        DDR(LCD_DATA3_PORT) |= _BV(LCD_DATA3_PIN);

        /* output high nibble first */
        LCD_DATA3_PORT &= ~_BV(LCD_DATA3_PIN);
        LCD_DATA2_PORT &= ~_BV(LCD_DATA2_PIN);
        LCD_DATA1_PORT &= ~_BV(LCD_DATA1_PIN);
        LCD_DATA0_PORT &= ~_BV(LCD_DATA0_PIN);
        if(data & 0x80) LCD_DATA3_PORT |= _BV(LCD_DATA3_PIN);
        if(data & 0x40) LCD_DATA2_PORT |= _BV(LCD_DATA2_PIN);
        if(data & 0x20) LCD_DATA1_PORT |= _BV(LCD_DATA1_PIN);
        if(data & 0x10) LCD_DATA0_PORT |= _BV(LCD_DATA0_PIN);
        lcd_e_toggle();

        /* output low nibble */
        LCD_DATA3_PORT &= ~_BV(LCD_DATA3_PIN);
        LCD_DATA2_PORT &= ~_BV(LCD_DATA2_PIN);
        LCD_DATA1_PORT &= ~_BV(LCD_DATA1_PIN);
        LCD_DATA0_PORT &= ~_BV(LCD_DATA0_PIN);
        if(data & 0x08) LCD_DATA3_PORT |= _BV(LCD_DATA3_PIN);
        if(data & 0x04) LCD_DATA2_PORT |= _BV(LCD_DATA2_PIN);
        if(data & 0x02) LCD_DATA1_PORT |= _BV(LCD_DATA1_PIN);
```

```
        if(data & 0x01) LCD_DATA0_PORT |= _BV(LCD_DATA0_PIN);
        lcd_e_toggle();

        /* all data pins high (inactive) */
        LCD_DATA0_PORT |= _BV(LCD_DATA0_PIN);
        LCD_DATA1_PORT |= _BV(LCD_DATA1_PIN);
        LCD_DATA2_PORT |= _BV(LCD_DATA2_PIN);
        LCD_DATA3_PORT |= _BV(LCD_DATA3_PIN);
    }
}
#else
#define lcd_write(d,rs) if (rs) *(volatile uint8_t*)(LCD_IO_DATA) = d; else *(volatile
uint8_t*)(LCD_IO_FUNCTION) = d;
/* rs==0 -> write instruction to LCD_IO_FUNCTION */
/* rs==1 -> write data to LCD_IO_DATA */
#endif


/*************************************************************************
Low-level function to read byte from LCD controller
Input:    rs     1: read data
                 0: read busy flag / address counter
Returns:  byte read from LCD controller
*************************************************************************/
#if LCD_IO_MODE
static uint8_t lcd_read(uint8_t rs)
{
    uint8_t data;


    if (rs)
        lcd_rs_high();                      /* RS=1: read data      */
    else
        lcd_rs_low();                       /* RS=0: read busy flag */
    lcd_rw_high();                          /* RW=1  read mode      */

    if ( ( &LCD_DATA0_PORT == &LCD_DATA1_PORT) && ( &LCD_DATA1_PORT == &LCD_DATA2_PORT ) && (
&LCD_DATA2_PORT == &LCD_DATA3_PORT )
      && ( LCD_DATA0_PIN == 0 )&& (LCD_DATA1_PIN == 1) && (LCD_DATA2_PIN == 2) && (LCD_DATA3_PIN
== 3) )
    {
        DDR(LCD_DATA0_PORT) &= 0xF0;        /* configure data pins as input */

        lcd_e_high();
        lcd_e_delay();
        data = PIN(LCD_DATA0_PORT) << 4;    /* read high nibble first */
        lcd_e_low();

        lcd_e_delay();                      /* Enable 500ns low       */

        lcd_e_high();
        lcd_e_delay();
        data |= PIN(LCD_DATA0_PORT)&0x0F;   /* read low nibble        */
        lcd_e_low();
    }
    else
    {
        /* configure data pins as input */
        DDR(LCD_DATA0_PORT) &= ~_BV(LCD_DATA0_PIN);
        DDR(LCD_DATA1_PORT) &= ~_BV(LCD_DATA1_PIN);
        DDR(LCD_DATA2_PORT) &= ~_BV(LCD_DATA2_PIN);
        DDR(LCD_DATA3_PORT) &= ~_BV(LCD_DATA3_PIN);

        /* read high nibble first */
        lcd_e_high();
        lcd_e_delay();
        data = 0;
        if ( PIN(LCD_DATA0_PORT) & _BV(LCD_DATA0_PIN) ) data |= 0x10;
        if ( PIN(LCD_DATA1_PORT) & _BV(LCD_DATA1_PIN) ) data |= 0x20;
        if ( PIN(LCD_DATA2_PORT) & _BV(LCD_DATA2_PIN) ) data |= 0x40;
        if ( PIN(LCD_DATA3_PORT) & _BV(LCD_DATA3_PIN) ) data |= 0x80;
        lcd_e_low();

        lcd_e_delay();                      /* Enable 500ns low       */

        /* read low nibble */
```

```
        lcd_e_high();
        lcd_e_delay();
        if ( PIN(LCD_DATA0_PORT) & _BV(LCD_DATA0_PIN) ) data |= 0x01;
        if ( PIN(LCD_DATA1_PORT) & _BV(LCD_DATA1_PIN) ) data |= 0x02;
        if ( PIN(LCD_DATA2_PORT) & _BV(LCD_DATA2_PIN) ) data |= 0x04;
        if ( PIN(LCD_DATA3_PORT) & _BV(LCD_DATA3_PIN) ) data |= 0x08;
        lcd_e_low();
    }
    return data;
}
#else
#define lcd_read(rs) (rs) ? *(volatile uint8_t*)(LCD_IO_DATA+LCD_IO_READ) : *(volatile
uint8_t*)(LCD_IO_FUNCTION+LCD_IO_READ)
/* rs==0 -> read instruction from LCD_IO_FUNCTION */
/* rs==1 -> read data from LCD_IO_DATA */
#endif


/*************************************************************************
loops while lcd is busy, returns address counter
*************************************************************************/
static uint8_t lcd_waitbusy(void)

{
    register uint8_t c;

    /* wait until busy flag is cleared */
    while ( (c=lcd_read(0)) & (1<<LCD_BUSY)) {}

    /* the address counter is updated 4us after the busy flag is cleared */
    delay(2);

    /* now read the address counter */
    return (lcd_read(0));  // return address counter

}/* lcd_waitbusy */


/*************************************************************************
Move cursor to the start of next line or to the first line if the cursor
is already on the last line.
*************************************************************************/
static inline void lcd_newline(uint8_t pos)
{
    register uint8_t addressCounter;


#if LCD_LINES==1
    addressCounter = 0;
#endif
#if LCD_LINES==2
    if ( pos < (LCD_START_LINE2) )
        addressCounter = LCD_START_LINE2;
    else
        addressCounter = LCD_START_LINE1;
#endif
#if LCD_LINES==4
#if KS0073_4LINES_MODE
    if ( pos < LCD_START_LINE2 )
        addressCounter = LCD_START_LINE2;
    else if ( (pos >= LCD_START_LINE2) && (pos < LCD_START_LINE3) )
        addressCounter = LCD_START_LINE3;
    else if ( (pos >= LCD_START_LINE3) && (pos < LCD_START_LINE4) )
        addressCounter = LCD_START_LINE4;
    else
        addressCounter = LCD_START_LINE1;
#else
    if ( pos < LCD_START_LINE3 )
        addressCounter = LCD_START_LINE2;
    else if ( (pos >= LCD_START_LINE2) && (pos < LCD_START_LINE4) )
        addressCounter = LCD_START_LINE3;
    else if ( (pos >= LCD_START_LINE3) && (pos < LCD_START_LINE2) )
        addressCounter = LCD_START_LINE4;
    else
        addressCounter = LCD_START_LINE1;
#endif
```

```
#endif
    lcd_command((1<<LCD_DDRAM)+addressCounter);

}/* lcd_newline */


/*
** PUBLIC FUNCTIONS
*/

/*************************************************************************
Send LCD controller instruction command
Input:   instruction to send to LCD controller, see HD44780 data sheet
Returns: none
*************************************************************************/
void lcd_command(uint8_t cmd)
{
    lcd_waitbusy();
    lcd_write(cmd,0);
}


/*************************************************************************
Send data byte to LCD controller
Input:   data to send to LCD controller, see HD44780 data sheet
Returns: none
*************************************************************************/
void lcd_data(uint8_t data)
{
    lcd_waitbusy();
    lcd_write(data,1);
}



/*************************************************************************
Set cursor to specified position
Input:    x  horizontal position  (0: left most position)
          y  vertical position    (0: first line)
Returns:  none
*************************************************************************/
void lcd_gotoxy(uint8_t x, uint8_t y)
{
#if LCD_LINES==1
    lcd_command((1<<LCD_DDRAM)+LCD_START_LINE1+x);
#endif
#if LCD_LINES==2
    if ( y==0 )
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE1+x);
    else
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE2+x);
#endif
#if LCD_LINES==4
    if ( y==0 )
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE1+x);
    else if ( y==1)
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE2+x);
    else if ( y==2)
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE3+x);
    else /* y==3 */
        lcd_command((1<<LCD_DDRAM)+LCD_START_LINE4+x);
#endif

}/* lcd_gotoxy */


/*************************************************************************
*************************************************************************/
int lcd_getxy(void)
{
    return lcd_waitbusy();
}


/*************************************************************************
Clear display and set cursor to home position
```

```
*************************************************************************/
void lcd_clrscr(void)
{
    lcd_command(1<<LCD_CLR);
}


/*************************************************************************
Set cursor to home position
*************************************************************************/
void lcd_home(void)
{
    lcd_command(1<<LCD_HOME);
}


/*************************************************************************
Display character at current cursor position
Input:    character to be displayed
Returns:  none
*************************************************************************/
void lcd_putc(char c)
{
    uint8_t pos;


    pos = lcd_waitbusy();   // read busy-flag and address counter
    if (c=='\n')
    {
        lcd_newline(pos);
    }
    else
    {
#if LCD_WRAP_LINES==1
#if LCD_LINES==1
        if ( pos == LCD_START_LINE1+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE1,0);
        }
#elif LCD_LINES==2
        if ( pos == LCD_START_LINE1+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE2,0);
        }else if ( pos == LCD_START_LINE2+LCD_DISP_LENGTH ){
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE1,0);
        }
#elif LCD_LINES==4
        if ( pos == LCD_START_LINE1+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE2,0);
        }else if ( pos == LCD_START_LINE2+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE3,0);
        }else if ( pos == LCD_START_LINE3+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE4,0);
        }else if ( pos == LCD_START_LINE4+LCD_DISP_LENGTH ) {
            lcd_write((1<<LCD_DDRAM)+LCD_START_LINE1,0);
        }
#endif
        lcd_waitbusy();
#endif
        lcd_write(c, 1);
    }

}/* lcd_putc */


/*************************************************************************
Display string without auto linefeed
Input:    string to be displayed
Returns:  none
*************************************************************************/
void lcd_puts(const char *s)
/* print string on lcd (no auto linefeed) */
{
    register char c;

    while ( (c = *s++) ) {
        lcd_putc(c);
    }
```

```
}/* lcd_puts */


/*************************************************************************
Display string from program memory without auto linefeed
Input:     string from program memory be be displayed
Returns:   none
*************************************************************************/
void lcd_puts_p(const char *progmem_s)
/* print string from program memory on lcd (no auto linefeed) */
{
    register char c;

    while ( (c = pgm_read_byte(progmem_s++)) ) {
        lcd_putc(c);
    }

}/* lcd_puts_p */


/*************************************************************************
Initialize display and select type of cursor
Input:    dispAttr LCD_DISP_OFF            display off
                   LCD_DISP_ON             display on, cursor off
                   LCD_DISP_ON_CURSOR      display on, cursor on
                   LCD_DISP_CURSOR_BLINK   display on, cursor on flashing
Returns:  none
*************************************************************************/
void lcd_init(uint8_t dispAttr)
{
#if LCD_IO_MODE
    /*
     *  Initialize LCD to 4 bit I/O mode
     */

    if ( ( &LCD_DATA0_PORT == &LCD_DATA1_PORT) && ( &LCD_DATA1_PORT == &LCD_DATA2_PORT ) && (
&LCD_DATA2_PORT == &LCD_DATA3_PORT )
      && ( &LCD_RS_PORT == &LCD_DATA0_PORT) && ( &LCD_RW_PORT == &LCD_DATA0_PORT) && (&LCD_E_PORT
== &LCD_DATA0_PORT)
      && (LCD_DATA0_PIN == 0 ) && (LCD_DATA1_PIN == 1) && (LCD_DATA2_PIN == 2) && (LCD_DATA3_PIN
== 3)
      && (LCD_RS_PIN == 4 ) && (LCD_RW_PIN == 5) && (LCD_E_PIN == 6 ) )
    {
        /* configure all port bits as output (all LCD lines on same port) */
        DDR(LCD_DATA0_PORT) |= 0x7F;
    }
    else if ( ( &LCD_DATA0_PORT == &LCD_DATA1_PORT) && ( &LCD_DATA1_PORT == &LCD_DATA2_PORT ) && (
&LCD_DATA2_PORT == &LCD_DATA3_PORT )
           && (LCD_DATA0_PIN == 0 ) && (LCD_DATA1_PIN == 1) && (LCD_DATA2_PIN == 2) &&
(LCD_DATA3_PIN == 3) )
    {
        /* configure all port bits as output (all LCD data lines on same port, but control lines
on different ports) */
        DDR(LCD_DATA0_PORT) |= 0x0F;
        DDR(LCD_RS_PORT)    |= _BV(LCD_RS_PIN);
        DDR(LCD_RW_PORT)    |= _BV(LCD_RW_PIN);
        DDR(LCD_E_PORT)     |= _BV(LCD_E_PIN);
    }
    else
    {
        /* configure all port bits as output (LCD data and control lines on different ports */
        DDR(LCD_RS_PORT)    |= _BV(LCD_RS_PIN);
        DDR(LCD_RW_PORT)    |= _BV(LCD_RW_PIN);
        DDR(LCD_E_PORT)     |= _BV(LCD_E_PIN);
        DDR(LCD_DATA0_PORT) |= _BV(LCD_DATA0_PIN);
        DDR(LCD_DATA1_PORT) |= _BV(LCD_DATA1_PIN);
        DDR(LCD_DATA2_PORT) |= _BV(LCD_DATA2_PIN);
        DDR(LCD_DATA3_PORT) |= _BV(LCD_DATA3_PIN);
    }
    delay(16000);        /* wait 16ms or more after power-on       */

    /* initial write to lcd is 8bit */
    LCD_DATA1_PORT |= _BV(LCD_DATA1_PIN);  // _BV(LCD_FUNCTION)>>4;
    LCD_DATA0_PORT |= _BV(LCD_DATA0_PIN);  // _BV(LCD_FUNCTION_8BIT)>>4;
    lcd_e_toggle();
```

```
    delay(4992);         /* delay, busy flag can't be checked here */

    /* repeat last command */
    lcd_e_toggle();
    delay(64);           /* delay, busy flag can't be checked here */

    /* repeat last command a third time */
    lcd_e_toggle();
    delay(64);           /* delay, busy flag can't be checked here */

    /* now configure for 4bit mode */
    LCD_DATA0_PORT &= ~_BV(LCD_DATA0_PIN);   // LCD_FUNCTION_4BIT_1LINE>>4
    lcd_e_toggle();
    delay(64);           /* some displays need this additional delay */

    /* from now the LCD only accepts 4 bit I/O, we can use lcd_command() */
#else
    /*
     * Initialize LCD to 8 bit memory mapped mode
     */

    /* enable external SRAM (memory mapped lcd) and one wait state */
    MCUCR = _BV(SRE) | _BV(SRW);

    /* reset LCD */
    delay(16000);                           /* wait 16ms after power-on     */
    lcd_write(LCD_FUNCTION_8BIT_1LINE,0);   /* function set: 8bit interface */
    delay(4992);                            /* wait 5ms                     */
    lcd_write(LCD_FUNCTION_8BIT_1LINE,0);   /* function set: 8bit interface */
    delay(64);                              /* wait 64us                    */
    lcd_write(LCD_FUNCTION_8BIT_1LINE,0);   /* function set: 8bit interface */
    delay(64);                              /* wait 64us                    */
#endif

#if KS0073_4LINES_MODE
    /* Display with KS0073 controller requires special commands for enabling 4 line mode */
    lcd_command(KS0073_EXTENDED_FUNCTION_REGISTER_ON);
    lcd_command(KS0073_4LINES_MODE);
    lcd_command(KS0073_EXTENDED_FUNCTION_REGISTER_OFF);
#else
    lcd_command(LCD_FUNCTION_DEFAULT);      /* function set: display lines  */
#endif
    lcd_command(LCD_DISP_OFF);              /* display off                  */
    lcd_clrscr();                           /* display clear                */
    lcd_command(LCD_MODE_DEFAULT);          /* set entry mode               */
    lcd_command(dispAttr);                  /* display/cursor control       */

}/* lcd_init */
```

A.2. Αλγόριθμος Bresenham

Ο Jack E. Bresenham το 1962 στα εργαστήρια της εταιρείας IBM επινόησε εναν αλγόριθμο για την σχεδίαση γραμμών σε τετραγωνικό καμβά. Ο αλγόριθμος αυτός επιλέγει ποιές ψηφίδες του καμβά πρέπει να σχεδιαστούν , ώστε να προσεγγίζεται η δεδομένη ευθεία. Η διαδικασία αυτή έχει κάποιες ομοιότητες με τον αλγόριθμο που χρησιμοποιήθηκε στο σύστημά μας ωστε να βρούμε διαδοχικές κλασματικές προσεγγίσεις ενός δοθέντος κλάσματος. Παραθέτω την απλή μορφή του αλγόριθμου του Bresenham σε ψευδοκώδικα:

Η συνάρτηση line δέχεται σαν ορίσματα τις συντεταγμένες αρχής και τέλους ενός ευθύγραμμου τμήματος που εκτείνεται απο το σημείο A (x0,x1) στο B(y0,y1). Προυπόθεση σε αυτήν την έκδοση του κώδικα είναι να ισχύει x0<x1 και y0<y1:

```
function line(x0, x1, y0, y1)
```

```
     int deltax := x1 - x0
     int deltay := y1 - y0
     real error := 0
     real deltaerr := abs (deltay / deltax)     // Assume deltax != 0 (line is not
vertical),
           // note that this division needs to be done in a way that preserves the
fractional part
     int y := y0
     for x from x0 to x1
         plot(x,y)
         error := error + deltaerr
         if error >= 0.5 then
             y := y + 1
             error := error - 1.0
```

Ο αλγόριθμος μπορεί να γραφεί σε μορφή που να χειρίζεται οποιαδήποτε ευθεία, αλλα επιλέχθηκε να παρουσιαστεί η πιο απλή μορφή του. Επίσης, θα παρουσιαστεί μια παραλλαγή του κώδικα η οποία δεν κάνει χρήση δεκαδικών αριθμών, αλλα χρησιμοποιεί μόνο απλές πράξεις ακεραίων. Η υλοποίηση αυτή χρησιμοποιήθηκε και στο σύστημα της εργασίας για τους ανάλογους υπολογισμούς. Παρατίθεται ο αλγόριθμος σε ψευδοκώδικα:

```
function line(x0, x1, y0, y1)
     int deltax := x1 - x0
     int deltay :=y1 - y0
     int error := deltax / 2
     int ystep := 1
     int y := y0
     for x from x0 to x1
         plot(x,y)
         error := error - deltay
         if error < 0 then
             y := y + ystep
             error := error + deltax
```

Παράρτημα Β': Τα τεχνικά χαρακτηριστικά των στοιχείων

Στο παράρτημα αυτό θα παρατεθούν links για τα τεχνικά εγχειρίδια όλων των ηλεκτρονικών στοιχείων που χρησιμοποιήθηκαν στην κατασκευή:

Μικροελεγκτής ATMEL ATmega16:
http://pdf1.alldatasheet.com/datasheet-pdf/view/78532/ATMEL/ATMEGA16.html

Γέφυρα-Η οδήγησης κινητήρα L298:
http://pdf1.alldatasheet.com/datasheet-pdf/view/22437/STMICROELECTRONICS/L298.html

Σταθεροποιητές τάσης LM78XX:
http://pdf1.alldatasheet.com/datasheet-pdf/view/82833/FAIRCHILD/LM7805.html

Αισθητήρες-οπτικοί κωδικοποιητές H21A1 και GP1S73P:

http://pdf1.alldatasheet.com/datasheet-pdf/view/52733/FAIRCHILD/H21A1.html
http://pdf1.alldatasheet.com/datasheet-pdf/view/42819/SHARP/GP1S73P.html

Οπτικοι απομονωτές K814P:
http://pdf1.alldatasheet.com/datasheet-pdf/view/26342/VISHAY/K814P.html

Hex inverting Schmitt trigger 74hc14:
http://pdf1.alldatasheet.com/datasheet-pdf/view/15537/PHILIPS/74HC14.html