ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ ΚΑΙ ΥΛΙΚΟΥ

ΣΥΣΤΗΜΑ ΣΥΛΛΟΓΗΣ ΔΕΔΟΜΕΝΩΝ ΓΙΑ ΙΔΕΑΤΟ ΜΟΥΣΕΙΟ ΒΑΣΙΣΜΕΝΟ ΣΕ ΜΙΚΡΟΕΛΕΓΚΤΗ

Διπλωματική εργασία

Βαρσαμίδης Χαρίσης

Επιβλέπων Καθηγητής: Καθηγητής Απόστολος Δόλλας

Εξεταστική επιτροπή: Καθηγητής Απόστολος Δόλλας
Καθηγητής Μιχάλης Ζερβάκης
Αν. Καθηγητής Ιωάννης Παπαευσταθίου

Χανιά 2015

Ευχαριστίες:

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή κύριο Απόστολο Δόλλα, επιβλέποντα της παρούσας διπλωματικής, για τις πολύτιμες συμβουλές του και την κατανόησή του σε όλη τη διάρκεια της υλοποίησης, καθώς και τον κύριο Μάρκο Κιμιωνή για την άμεση διάθεση των απαραίτητων εργαλείων. Θα ήθελα να ευχαριστήσω επίσης το φίλο και συνάδελφό μου Γιώργο Χαϊλαζόπουλο για τις πολύτιμες συμβουλές του ,τους συγκατοίκους μου στα Χανιά Σιδέρη Παναγιώτη και Δελή Γιώργο, το ζεύγος Λελεδάκη, τη Χρυσάνθη και τον Άρη για τη στήριξη τους σε όλους τους τομείς. Περισσότερο απ' όλους όμως, θα ήθελα να ευχαριστήσω τους γονείς μου, για την πολυετή υποστήριξη, ανοχή και αντοχή τους και τα αδέρφια μου Γιώργο και Λιάνα.

Περίληψη

Με τη γρήγορη ανάπτυξη του διαδικτύου τις τελευταίες δύο δεκαετίες, έχει αυξηθεί ο αριθμός των ιστοσελίδων που προσφέρουν εικονικές ξεναγήσεις σε μουσεία και εκθέσεις, στις οποίες ο επισκέπτης μπορεί να περιηγηθεί μέσω του διαδικτύου. Έτσι παρέχεται η δυνατότητα στον επισκέπτη να μελετήσει 3D μοντέλα των εκθεμάτων, όπως θα συνέβαινε με την παρουσία του σε ένα φυσικό μουσείο ή έκθεση.

Για το σχεδιασμό των 3D εκθεμάτων ενός Ιδεατού Μουσείου χρησιμοποιούνται διάφορες μέθοδοι λήψης δεδομένων και σχεδίασης. Η μέθοδος λήψης δεδομένων με την οποία ασχολείται η παρούσα διπλωματική εργασία, αφορά τη λήψη φωτογραφιών ενός εκθέματος από διάφορες γωνίες εντός 360 μοιρών, με στόχο την τελική χρήση των φωτογραφιών για μελλοντική δημιουργία του 3D μοντέλου του εκθέματος.

Για την υλοποίηση της παραπάνω εφαρμογής, κατασκευάστηκε ένα σύστημα στρεφόμενης πλατφόρμας για την υποστήριξη του εκθέματος και την επακόλουθη φωτογράφισή του. Η κίνηση της πλατφόρμας είναι βασισμένη σε βηματικό κινητήρα (stepper motor) ενώ ο έλεγχος του συστήματος βασίζεται σε μικροελεγκτή AVR, τον ATmega8515.

Για τη διεπαφή του συστήματος με το χρήστη, αναπτύχθηκε εφαρμογή Windows, από την οποία ο χρήστης μπορεί να χειριστεί το σύστημα και να επιλέξει τις επιθυμητές γωνίες λήψης. Η εφαρμογή είναι επίσης υπεύθυνη για τη φωτογράφιση του εκθέματος μέσω της ενεργοποίησης της web camera του υπολογιστή. Μετά τη φωτογράφιση του εκθέματος οι φωτογραφίες αποθηκεύονται σε ένα επιλεγμένο directory του υπολογιστή, από το οποίο ο χρήστης μπορεί να τις ανακτήσει και να τις χρησιμοποιήσει.

Μελλοντικά, οι φωτογραφίες μπορούν να χρησιμοποιηθούν για την κατασκευή του 3D μοντέλου του εκάστοτε εκθέματος, ή να αποθηκευθούν σε κάποια βάση δεδομένων παράλληλα με πληροφορίες για το κάθε έκθεμα, με τελικό στόχο τη δημιουργία ενός πλήρους συστήματος Ιδεατού Μουσείου.

Πίνακας περιεχομένων

1. ΕΙΣΑΓΩΓΗ....10
1.1 Γενικά το πρόβλημα της αυτοματοποίησης συλλογής δεδομένων....10
1.2 Αντικείμενο και Σκοπός της Διπλωματικής....11
1.3 Οργάνωση κεφαλαίων....12
2. ΣΧΕΤΙΚΗ ΕΡΕΥΝΑ....13
2.1 Ιδεατό μουσείο....13
2.1.1 Ιστορική αναδρομή....13
2.1.2 Τεχνολογίες απεικόνισης σε σύγχρονα Ιδεατά Μουσεία....14
2.1.3 Ιδεατά μουσεία μη βασισμένα σε φυσικά....15
2.2 Στρεφόμενες πλατφόρμες αυτόματης λήψης φωτογραφιών....16
3. ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ....18
3.1 Λειτουργικό διάγραμμα του συστήματος....18
3.2 Ο μικροελεγκτής ATMEGA8515....20
3.2.1 Η αρχιτεκτονική του ATMEGA8515....21
3.2.2 Το USART του ATmega8515....23
3.3 Το Stepper Motor....31
3.4 Ο Stepper Motor Driver ULN2003....33
3.5 Η γραφική εφαρμογή....33
4. ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ....36
4.1 Ο προγραμματισμός του ATmega8515....36
4.1.1 Συναρτήσεις κίνησης του Stepper Motor....39
4.1.2 Η συνάρτηση αρχικοποίησης του USART....41
4.1.3 Η συνάρτηση λήψης δεδομένων....43
4.1.4 Η συνάρτηση αποστολής δεδομένων....43
4.1.5 Μετατροπή από χαρακτήρα σε ακέραιο:....44
4.2 Η ανάπτυξη της εφαρμογής....45
4.2.1 Η γραφική σχεδίαση της εφαρμογής....45
4.2.2 Οι λειτουργίες της εφαρμογής....48
4.2.3 Επικοινωνία μεταξύ PC και μικροελεγκτή....50
4.2.4 Απόρριψη Προηγούμενων Υλοποιήσεων....52
5. ΕΠΙΒΕΒΑΙΩΣΗ ΛΕΙΤΟΥΡΓΙΑΣ....54

5.1 Επιβεβαίωση λειτουργίας του Stepper 54
5.2 Επιβεβαίωση σειριακής επικοινωνίας 55
5.3 Επιβεβαίωση λειτουργίας συνολικού Hardware 55
5.4 Επιβεβαίωση λειτουργίας της web camera 55
6. ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ 57
7. ΒΙΒΛΙΟΓΡΑΦΙΑ 59
ΠΑΡΑΡΤΗΜΑ Α 62
1 Stepper Motors 62
1.1 Γενικά για τα DC Motors 62
1.2 Stepper Motors 65
1.3 Ροπή ενός stepper motor 67
1.4 Εφαρμογές Stepper motors 70
2 Stepper motor drivers 71
2.1 Darlington Transistor Array 71
3 Μικροελεγκτές 73
3.1 Μικροελεγκτές AVR 73
4 Πίνακας ASCII 76

1. ΕΙΣΑΓΩΓΗ

1.1 Γενικά το πρόβλημα της αυτοματοποίησης συλλογής δεδομένων

Εδώ και αρκετές δεκαετίες και τα τελευταία χρόνια όλο και περισσότερο, η αυτοματοποίηση συλλογής δεδομένων βρίσκει εφαρμογή σε ολοένα και περισσότερες υλοποιήσεις, από εφαρμογές οικιακής χρήσης μέχρι βιομηχανικά συστήματα. Αυτοματισμοί οικίας, δίκτυα αισθητήρων, μετρητές κατανάλωσης ισχύος, όλα τα παραπάνω βασίζονται στη λήψη και τη διαχείριση δεδομένων από κυκλώματα αυτοματισμού. Η παρούσα διπλωματική εργασία ασχολείται με το θέμα της λήψης δεδομένων και πιο συγκεκριμένα φωτογραφιών, για τη σχεδίαση συστήματος Ιδεατού Μουσείου. Η χρήση του συστήματος που αναπτύχθηκε και υλοποίησή του θα αναλυθούν στα ακόλουθα κεφάλαια.

1.2 Αντικείμενο και Σκοπός της Διπλωματικής

Σκοπός της παρούσας εργασίας με τίτλο "Σύστημα Συλλογής Δεδομένων για Ιδεατό Μουσείο Βασισμένο σε Μικροελεγκτή", είναι η εισαγωγή στο θέμα του αυτοματισμού με χρήση μικροελεγκτών και η απόκτηση τεχνογνωσίας στη σχεδίαση και την αρχιτεκτονική συστημάτων που χρησιμοποιούν διεπαφές υλικού-λογισμικού, stepper motors και συλλογής δεδομένων. Στην παρούσα διπλωματική, σχεδιάστηκε και αναπτύχθηκε ένα σύστημα λήψης φωτογραφιών μουσειακού ή συλλεκτικού εκθέματος, τοποθετημένου σε στρεφόμενη πλατφόρμα με τελικό αποτέλεσμα την αυτόματη, κατόπιν απαίτησης του χρήστη, φωτογράφιση του εκθέματος υπό κατ' επιλογή γωνίες, εντός μίας ολόκληρης περιστροφής. Για την περάτωση της διπλωματικής αυτής χρησιμοποιήθηκαν επιγραμματικά τα παρακάτω εργαλεία , τα οποία θα αναλυθούν εκτενώς σε επόμενα κεφάλαια:

Ο πυρήνας λειτουργίας του συστήματος που αναπτύχθηκε είναι ο μικροελεγκτής της ATMEL Avr ATmega8515, ο οποίος είναι υπεύθυνος για την κίνηση του stepper motor, τη σειριακή επικοινωνία με τον υπολογιστή και την ενορχήστρωση της διαδικασίας λήψης φωτογραφιών (η εντολή λήψης φωτογραφίας δεν δίνεται από τον ίδιο το μικροελεγκτή καθότι ήταν ευκολότερο να γίνει από τον υπολογιστή, αλλά παρόλα αυτά, για τη λήψη μίας φωτογραφίας πρέπει να δοθεί πρώτα επιβεβαίωση από το μικροελεγκτή). Ο μικροελεγκτής προγραμματίστηκε με το εργαλείο προγραμματισμού μικροελεγκτών AVR της ATMEL, STK 500 , ενώ η πλατφόρμα προγραμματισμού που χρησιμοποιήθηκε για τη συγγραφή του κώδικα είναι το Atmel studio 6.2.

Για την κίνηση της πλατφόρμας, χρησιμοποιήθηκε ένα απλό stepper motor με 64 βήματα ανά περιστροφή, ενώ για την οδήγηση αυτού του stepper motor χρησιμοποιήθηκε ο stepper motor driver ULN2003, υπεύθυνος για τη μετάφραση των ψηφιακών σημάτων του μικροελεγκτή σε τάση και ρεύμα. Το σύστημα θα μπορούσε να είναι λειτουργικό και χωρίς την παρουσία του driver, καθώς το λογικό '1' του μικροελεγκτή αντιστοιχεί σε 5V τάση, όσο είναι και το ελάχιστο ποσό τάσης που απαιτεί το εν λόγω stepper για τη λειτουργία του. Με αυτήν την υλοποίηση όμως θα παρουσιαζόταν πρόβλημα στην ακρίβεια του stepper από άποψη θέσεων και βημάτων, καθώς επίσης θα εμφανιζόταν μείωση της προσφερόμενης ροπής του, καθώς ο μικροελεγκτής από μόνος του δεν είναι σε θέση να δώσει το απαραίτητο ρεύμα για την ορθή κίνηση του stepper motor , ή αν επιχειρήσουμε να γίνει αυτό υπάρχει κίνδυνος βλάβης στον ίδιο το μικροελεγκτή.

Για τη λήψη των φωτογραφιών χρησιμοποιήθηκε μια απλή webcam συνδεδεμένη με τον υπολογιστή για λόγους απλότητας και χώρου. Θεωρήθηκε πιο σοφό και αποτελεσματικό οι φωτογραφίες να μην αποθηκεύονται αρχικά στο μικροελεγκτή λόγω έλλειψης αποθηκευτικού χώρου αλλά να αποθηκεύονται απ' ευθείας σε ένα επιλεγμένο directory του υπολογιστή.

Για την επικοινωνία μεταξύ μικροελεγκτή και υπολογιστή χρησιμοποιήθηκε το USART(Universal Synchronous Asynchronous Receiver Transmitter) του μικροελεγκτή ATmega8515, με τις προκαθορισμένες συναρτήσεις transmit και receive, καθώς και τους default καταχωρητές USART. Λόγω της επικοινωνίας μέσω USART ήταν απαραίτητη η

ύπαρξη σειριακής θύρας RS232 στο κύκλωμα του μικροελεγκτή και για το λόγο αυτό δεν κατασκευάστηκε ειδικό κύκλωμα για το μικροελεγκτή, αλλά χρησιμοποιήθηκε η ήδη υπάρχουσα θύρα σειριακής επικοινωνίας του STK 500.

Ο χρήστης επιλέγει αρχικά από μία εφαρμογή Windows που αναπτύχθηκε και θα περιγραφεί παρακάτω, όσες θέλει από τις 64 πιθανές θέσεις λήψης φωτογραφίας. Στη συνέχεια η πληροφορία αυτή μεταφέρεται στο μικροελεγκτή μέσω του USART και εκείνος ξεκινάει τον αλγόριθμο κίνησης που θα περιγραφεί παρακάτω. Σε κάθε πιθανή θέση φωτογραφίας ο μικροελεγκτής διακόπτει την κίνηση και δίνει την επιβεβαίωση στον υπολογιστή μέσω του USART να ενεργοποιήσει την camera για τη λήψη φωτογραφίας. Μόλις η φωτογραφία ληφθεί και αποθηκευθεί σε ένα directory, τότε μπορεί να σταλεί πληροφορία για νέα θέση από τον υπολογιστή στο μικροελεγκτή.

1.3 Οργάνωση κεφαλαίων

Στα κεφάλαια που θα ακολουθήσουν, θα αναλυθούν τα τμήματα της παρούσας εργασίας σε θεωρητικό επίπεδο καθώς και σε επίπεδο σχεδίασης και υλοποίησης. Πιο συγκεκριμένα, το 2ο κεφάλαιο με τίτλο **Σχετική Έρευνα**, θα γίνει μία ιστορική αναδρομή στο θέμα των ιδεατών μουσείων και θα αναφερθούν ορισμένα συστήματα ιδεατού μουσείου και αυτόματης λήψης φωτογραφιών που υπάρχουν ήδη στην αγορά. Στο 3ο κεφάλαιο με τίτλο **Αρχιτεκτονική του Συστήματος** θα αναλυθούν θέματα σχεδίασης του hardware τμήματος του συστήματος, η δομή των επί μέρους κυκλωμάτων που χρησιμοποιήθηκαν, η δομή του ολοκληρωμένου συστήματος που αναπτύχθηκε καθώς και ο τρόπος λειτουργίας της σειριακής επικοινωνίας USART μεταξύ του μικροελεγκτή και του υπολογιστή-χειριστή της εφαρμογής. Στο 4ο κεφάλαιο με τίτλο **Σχεδίαση και Υλοποίηση**, θα περιγραφεί η σχεδιαστική μέθοδος που χρησιμοποιήθηκε, οι μέθοδοι προγραμματισμού του μικροελεγκτή, η σχεδίαση του γραφικού περιβάλλοντος της εφαρμογής και οι συναρτήσεις πάνω στις οποίες βασίζεται η λειτουργία του συστήματος (τόσο στο μικροελεγκτή όσο και στην εφαρμογή με την οποία αλληλεπιδρά ο χρήστης). Το 5ο κεφάλαιο με τίτλο **Επιβεβαίωση Λειτουργίας** περιλαμβάνει τις μεθόδους που ακολουθήθηκαν για την επιβεβαίωση της λειτουργίας των τμημάτων της εφαρμογής. Στο 6ο κεφάλαιο με τίτλο **Συμπεράσματα και Μελλοντική Εργασία** θα αναφερθούν πιθανές εφαρμογές του συστήματος καθώς και μελλοντικές προσθήκες και πιθανές επεκτάσεις. Στο **Παράρτημα** αναφέρονται γενικές πληροφορίες πάνω στα επιμέρους συστήματα που χρησιμοποιήθηκαν (αρχές λειτουργίας κλπ) ,ενώ τέλος αναφέρεται η **Βιβλιογραφία** η οποία μελετήθηκε για τη σχεδίαση του συστήματος.

2.
ΣΧΕΤΙΚΗ ΕΡΕΥΝΑ

2.1 Ιδεατό μουσείο

2.1.1 Ιστορική αναδρομή

Ένα **Ιδεατό Μουσείο**, (virtual museum) αποτελεί ουσιαστικά μία ψηφιακή οντότητα με τα χαρακτηριστικά ενός μουσείου, με στόχο να προσφέρει μία ψηφιακή εμπειρία ξενάγησης σε έναν φυσικό χώρο όπως μουσεία, συλλογές και εκθέσεις. Τα ιδεατά μουσεία μπορεί να αποτελούν είτε την ψηφιακή αναπαράσταση ενός υπαρκτού μουσείου, είτε ένα ξεχωριστό μουσείο, προσφέροντας και στις δύο περιπτώσεις τη δυνατότητα της εικονικής ξενάγησης. Τα εκθέματα που μπορεί να περιέχει ένα ιδεατό μουσείο μπορεί να είναι είτε ψηφιακές απεικονίσεις υλικών εκθεμάτων, είτε ψηφιακά εκθέματα εικονικής πραγματικότητας και ψηφιακής τέχνης. Αξίζει να αναφερθεί ότι η πρώτη ιδέα για την ανάπτυξη ενός είδους Ιδεατού Μουσείου έγινε στην αναγέννηση και είχε το όνομα Kunsthammer, (σε ελεύθερη μετάφραση σφυρί τέχνης) ή αλλιώς αποθήκη γνώσης. Ουσιαστικά ήταν ένας πίνακας στον οποίο ήταν ζωγραφισμένα διάφορα εκθεσιακά αντικείμενα σε πολύ μικρή κλίμακα. Ο επισκέπτης αποφάσιζε ποιο από όλα θέλει να δει, και το επιλεγμένο έκθεμα εμφανιζόταν με ένα σύστημα ανελκυστήρων πάνω σε μία πλατφόρμα, προερχόμενο από μία υπόγεια αποθήκη.[22],[23]

Πριν την ανάπτυξη του παγκόσμιου ιστού, μία από τις πρώτες προσπάθειες ανάπτυξης λογισμικού ιδεατού μουσείου έγινε από την Apple Computer.Το λογισμικό κυκλοφόρησε το 1992 με τον τίτλο “The Virtual Museum” σε μορφή CD-ROM. Η υλοποίηση του εν λόγω project ξεκίνησε ως διπλωματική εργασία η οποία στη συνέχεια χρηματοδοτήθηκε από την Apple Computer. Το CD-ROM διατέθηκε δωρεάν παγκοσμίως για εκπαιδευτικούς σκοπούς. Τα εκθέματα ήταν εκπαιδευτικού περιεχομένου και μοιρασμένα σε τομείς όπως ιατρική,

γεωπονία, περιβάλλον και διάστημα. Η πλοήγηση στον τρισδιάστατο χώρο του μουσείου επέτρεπε στο χρήστη να αλληλεπιδρά με τα 3D εκθέματα δίνοντας τη δυνατότητα αναπαραγωγής ήχου ή εικόνας με την επιλογή ενός αντικειμένου.[22]

Τα πρώτα συστήματα online ιδεατού μουσείου εμφανίστηκαν μαζί με την ανάπτυξη του παγκόσμιου ιστού στις αρχές της δεκαετίας του ’90. Λόγω της έλλειψης αξιόπιστων μηχανών αναζήτησης, μία online πλατφόρμα με τίτλο Virtual Library προσέφερε λύση στο πρόβλημα της αναζήτησης πληροφοριών στο διαδίκτυο χωρίς συγκεκριμένη γνώση της διεύθυνσης κάποιου ιστότοπου, καθώς περιείχε συγκεντρωμένες διευθύνσεις από ιστοσελίδες με κυρίως ακαδημαϊκό περιεχόμενο. Αργότερα στη Virtual Library προστέθηκαν διευθύνσεις ιστοσελίδων με πληροφορίες για διάφορα μουσεία και εκθέματα. Στη συνέχεια εμφανίστηκαν και άλλες ιστοσελίδες με περιεχόμενο Ιδεατού Μουσείου, όπως η ιστοσελίδα του παλαιοντολογικού μουσείου της California που προσέφερε τη δυνατότητα ξενάγησης σε εκθέματα online[24], ή το WebLouvre που αργότερα για νομικούς λόγους μετονομάστηκε σε WebMuseum, το οποίο παρείχε πληροφορίες για εκθέματα του Λούβρου, καθώς και τη γραφική απεικόνισή τους, σε δωρεάν διάθεση για τον online επισκέπτη.[21]

Από τα πρώτα μουσεία που άρχισαν να παρέχουν τη δυνατότητα της online ξενάγησης ήταν εκτός από το παλαιοντολογικό μουσείο της California, το μουσείο επιστημών του Λονδίνου και το μουσείο φυσικής ιστορίας του Λονδίνου τα οποία χρησιμοποίησαν και δικό τους server για τις ιστοσελίδες τους.[21]

2.1.2 Τεχνολογίες απεικόνισης σε σύγχρονα Ιδεατά Μουσεία

Τα περισσότερα φυσικά μουσεία σήμερα παρέχουν τη δυνατότητα online ξενάγησης μέσω της ιστοσελίδας τους. Οι τρόποι με τους οποίους αλληλεπιδρά ο χρήστης ενός ιδεατού μουσείου με το τρισδιάστατο περιβάλλον και τα ψηφιακά εκθέματα είναι διάφοροι. Συνήθως ο χρήστης αλληλεπιδρά με αντικείμενα τα οποία αποτελούν 3D αναπαραστάσεις των φυσικών εκθεμάτων των μουσείων. Για την τρισδιάστατη μοντελοποίηση των εκθεμάτων χρησιμοποιούνται οι γλώσσες VRML(Virtual Reality Modeling Language) και X3D(ο διάδοχος της VRML). Προκειμένου να “τοποθετηθούν” τα εκθέματα ενός φυσικού μουσείου ως εκθέματα στο αντίστοιχο ιδεατό μουσείο χρησιμοποιούνται διάφορες τεχνικές ψηφιακής απεικόνισης που εφαρμόζουν είτε υπέρυθρη ακτινοβολία, είτε σάρωση με laser είτε ακτίνες Χ. Η τεχνική απεικόνισης με την οποία ασχολείται η παρούσα διπλωματική (στο κομμάτι της λήψης δεδομένων) είναι βασισμένη στη λήψη φωτογραφιών των εκθεμάτων. Στη συνέχεια η 3D απεικόνιση γίνεται με τεχνικές IBMR (Image Based Rendering and Modeling).

Τα ιδεατά μουσεία απαιτούν υψηλής ανάλυσης φωτογραφίες, με στόχο να παρέχουν όσο περισσότερη πληροφορία γίνεται αναφορικά με τα ψηφιακά εκθέματα. Το επίπεδο των λεπτομερειών των εκθεμάτων ενός Ιδεατού Μουσείου εξαρτάται κατά πολύ από την ανάλυση των φωτογραφιών, όμως υψηλής ανάλυσης φωτογραφίες παράγουν πολύ μεγάλα και δύσκολα στη διαχείριση και μετάδοση αρχεία, λόγω των περιορισμών που προκύπτουν από αργές συνδέσεις Internet.[25] Το πρόβλημα αυτό αντιμετωπίζεται με τη χρήση της τεχνικής

"Russian doll", κατά την οποία πολλαπλές αναλύσεις της ίδιας εικόνας είναι αποθηκευμένες στο ίδιο αρχείο και έτσι είναι δυνατή η προοδευτική μετάδοση μίας εικόνας. Οι πιο δημοφιλείς τύποι αρχείων που χρησιμοποιούν αυτήν την τεχνική είναι οι JPEG2000 και FlashPix.[26]

Η πιο δημοφιλής τεχνολογία απεικόνισης online ονομάζεται Web3D και χρησιμοποιεί τα προαναφερθέντα εργαλεία VRML και X3D. Τεχνολογίες όπως μηχανές γραφικών παιχνιδιών χρησιμοποιούνται επίσης για τη σχεδίαση Ιδεατών Μουσείων. Άλλου είδους εκθέσεις αποτελούν οι εκθέσεις επαυξημένης πραγματικότητας, (Augmented Reality Exhibitions), στις οποίες η εικονική πληροφορία μεταδίδεται παράλληλα με video frames από κάμερα, δίνοντας την εντύπωση στο χρήστη ότι το αντικείμενο που βλέπει υπάρχει στο φυσικό χώρο. Επίσης με την τεχνολογία Haptics, δίνεται η δυνατότητα στο χρήστη να "αγγίξει" ένα αντικείμενο, προσομοιώνοντας την αίσθηση της αφής. [26]

2.1.3 Ιδεατά μουσεία μη βασισμένα σε φυσικά

Εκτός από τη χρήση των ιδεατών μουσείων για την περιήγηση σε μία ψηφιακή έκδοση ενός φυσικού μουσείου, υπάρχουν επίσης ιδεατά μουσεία εκπαιδευτικής χρήσης για εξειδικευμένα κοινά. Μία πρόταση που αφορά την κατασκευή ιδεατού μουσείου με θέμα την εκ γενετής καρδιοπάθεια, είναι το Global Virtual Museum of Congenital Cardiac Pathology. Η συγκεκριμένη πρόταση απευθύνεται σε ιατρική χρήση και αφορά τη συλλογή δεδομένων και φωτογραφιών ανατομίας για τη μελέτη της ανθρώπινης καρδιάς. Άλλο παράδειγμα αφορά το ιδεατό μουσείο Neothemi το οποίο περιλαμβάνει ψηφιακή ξενάγηση σε εκθέματα και μνημεία πολιτιστικής κληρονομιάς από χώρες της Ευρώπης. [27],[28].

Αξίζει να αναφερθεί και το project Scan the World, το οποίο έχει ως στόχο την αρχειοθέτηση ψηφιακών εικόνων από μνημεία και έργα τέχνης από όλο τον κόσμο και την αποθήκευσή τους σε 3D printable μορφή.[29] Επίσης στην Ελλάδα, ένα χαρακτηριστικό παράδειγμα είναι το Εικονικό Μουσείο Ελληνικών Παραδοσιακών Οργάνων, το οποίο παρέχει δυναμική πλοήγηση σε εικονικούς εκθεσιακούς χώρους όπου τα κύρια εκθέματα είναι 3D αναπαραστάσεις μουσικών οργάνων.[30]

2.2 Στρεφόμενες πλατφόρμες αυτόματης λήψης φωτογραφιών

Συστήματα παρόμοια με το σύστημα που αναπτύχθηκε στην παρούσα εργασία υπάρχουν ευρέως στο εμπόριο με το όνομα photography turntables και κυκλοφορούν είτε με ενσωματωμένη camera και δικό τους software ελέγχου, είτε όχι. Η κύρια χρήση τους είναι η λήψη φωτογραφιών από αντικείμενα όχι εκθεσιακά για τη χρήση σε κάποιο ιδεατό μουσείο, αλλά εμπορικά, με σκοπό την ανάρτηση του 3D μοντέλου των αντικειμένων αυτών στην αντίστοιχη ιστοσελίδα (product photography). Τέτοιες στρεφόμενες πλατφόρμες χρησιμοποιούνται επίσης για την προβολή αντικειμένων σε εκθέσεις, μουσεία ή δημοπρασίες. Ένα χαρακτηριστικό παράδειγμα είναι το λογισμικό και το υλικό της εταιρίας Iconasys, τα οποία παρέχουν στο χρήστη τη δυνατότητα επιλογής διαφορετικής πλατφόρμας ανάλογα με το είδος του αντικειμένου που θέλει να φωτογραφίσει και τη δυνατότητα επιλογής ως 72 πιθανών θέσεων λήψης σε έναν κύκλο. Το συγκεκριμένο λογισμικό επίσης προσφέρει τη δυνατότητα της κατασκευής ενός 3D μοντέλου βασισμένου στις φωτογραφίες. Οι συνηθισμένες θέσεις λήψης σε μία περιστροφή είναι 8, 12, 18, 24, 36 ή 72, και από τον αριθμό τους εξαρτάται και η ποιότητα του 3D μοντέλου που θα κατασκευαστεί εν τέλει.[31]

Πέραν των turntables του εμπορείου, έχουν κατασκευαστεί διάφορα άλλα αυτοσχέδια turntables για προσωπική χρήση όπως φαίνεται στις αναφορές [32],[33]. Μάλιστα, στην αναφορά [33] το έκθεμα είναι σταθερό και η camera περιστρέφεται γύρω από αυτό για τη λήψη των φωτογραφιών. Ο ευκολότερος τρόπος που χρησιμοποιείται σε αυτοσχέδια συστήματα για ένα είδος "3D" απεικόνισης, είναι η ταυτόχρονη προβολή των ληφθέντων φωτογραφιών σε ένα αρχείο τύπου .gif, η οποία μπορεί να γίνει είτε από εργαλεία όπως το Photoshop είτε και σε online σελίδες. Αξίζει να αναφερθεί επίσης το παράδειγμα [34] , όπου η το turntable ελέγχεται πλήρως από μία εφαρμογή smartphone, το οποίο είναι υπεύθυνο και για τη λήψη των φωτογραφιών. Πέραν των μηχανικών συστημάτων, υπάρχουν επίσης και χειροκίνητες πλατφόρμες, τις οποίες ο χρήστης ρυθμίζει σε μία θέση και έπειτα τραβάει τη φωτογραφία.

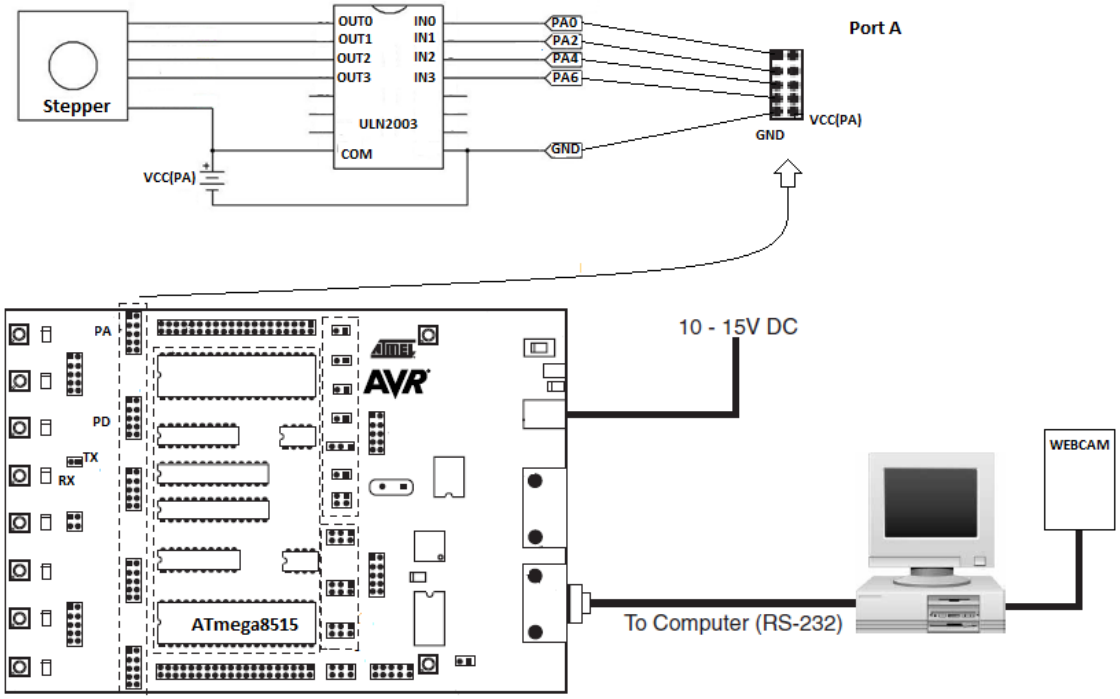
Η βασική διαφορά των συστημάτων που χρησιμοποιούνται για τη λήψη φωτογραφιών σε σύγκριση με το σύστημα που υλοποιήθηκε κατά την εκπόνηση της παρούσας διπλωματικής, είναι στον τρόπο κίνησης της πλατφόρμας. Η πλατφόρμα στα συνήθη συστήματα κινείται σε προκαθορισμένες θέσεις εντός μίας πλήρους περιστροφής προς την ίδια φορά και σε κάθε μία από αυτές τις θέσεις φωτογραφίζεται είτε χειροκίνητα είτε με τη βοήθεια κάποιου λογισμικού, με τελικό αποτέλεσμα τη συγκέντρωση των λήψεων που απαιτούνται για το τελικό 3D μοντέλο. Στο σύστημα που υλοποιήθηκε, ο χρήστης επιλέγει την ή τις συγκεκριμένες γωνίες στις οποίες θέλει να γίνει η λήψη, με επακόλουθη περιστροφή τις πλατφόρμας προς την ή τις επιλεγμένες θέσεις, με τη φορά της περιστροφής να εξαρτάται

από την απόσταση της τρέχουσας από την επόμενη θέση. Οι φωτογραφίες αποθηκεύονται σε directory του υπολογιστή, όμως δεν έχει υλοποιηθεί κάποια μέθοδος 3D απεικόνισης στην παρούσα εργασία.

3. ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

3.1 Λειτουργικό διάγραμμα του συστήματος

Τα τμήματα του συστήματος το οποίο σχεδιάστηκε και αναπτύχθηκε στην παρούσα διπλωματική θα αναλυθούν στο παρόν κεφάλαιο. Αυτά ονομαστικά είναι ο μικροελεγκτής AVR ATMEGA8515, το Stepper Motor, ο Stepper Motor Driver και η γραφική εφαρμογή που σχεδιάστηκε για τη διεπαφή του χρήστη με τον υπολογιστή και το χειρισμό της web cam. Για την εκπόνηση της παρούσας διπλωματικής δεν σχεδιάστηκε ξεχωριστό κύκλωμα. Χρησιμοποιήθηκε ως κύκλωμα η πλατφόρμα προγραμματισμού του μικροελεγκτή Atmega8515, STK 500, με τις εξόδους προς τα επιμέρους τμήματα της διπλωματικής να κατευθύνονται από εκεί. Παρακάτω παρουσιάζεται το σχεδιάγραμμα της συνδεσμολογίας του συστήματος, με τις λειτουργίες των αριθμημένων τμημάτων να περιγράφονται:



Εικόνα 1: Schematics συστήματος

1. Η πηγή τροφοδοσίας του κυκλώματος: Το STK 500 προσφέρει τη δυνατότητα χρήσης οποιουδήποτε μετασχηματιστή DC μεταξύ 10-15 Volt λόγω των ενσωματωμένων σταθεροποιητών τάσης που διαθέτει. Η ρύθμιση της τελικής τροφοδοσίας του μικροελεγκτή έγινε μέσω της πλατφόρμας προγραμματισμού Atmel Studio, και ορίστηκε στα 5 V. Το STK δίνει τη δυνατότητα ρύθμισης της τάσης σε ένα εύρος από 0-6V.[35]
2. Η σειριακή θύρα επικοινωνίας RS232: Μέσω της συγκεκριμένης σειριακής θύρας γίνεται η επικοινωνία του MCU με τον υπολογιστή χρησιμοποιώντας το USART του μικροελεγκτή Atmega8515. Η ελεύθερη (στο σχήμα) σειριακή θύρα του STK 500 χρησιμοποιήθηκε για τον προγραμματισμό του μικροελεγκτή.

3. Τα Pins SPROG,ISP6: Για τον προγραμματισμό του ATmega8515 συνδέθηκαν τα 6 pins ISP6 με τα pins SPROG3 που αντιστοιχούν στο Socket προγραμματισμού του Atmega8515 (40 pins, κόκκινο χρώμα).

4. Τα Pins RX και TX του STK 500: Για τη σειριακή επικοινωνία μεταξύ του μικροελεγκτή και του υπολογιστή και τη χρήση της σειριακής θύρας RS232 είναι

απαραίτητη η σύνδεση των Pins RX και TX του STK 500 με τα pins PD0 και PD1 του STK 500, που αντιστοιχούν στα pins 0 και 1 του PORTD του Atmega8515.

5. Ο stepper motor driver ULN2003: Οι είσοδοι του ULN2003 IN0 ως IN3, συνδέονται με τα pins PD0, PD2, PD4, PD6 κατ' αντιστοιχία, τα οποία αντιστοιχούν στα ισοδύναμα pins του ATmega8515 για την ενεργοποίηση και απενεργοποίηση των πόλων του stepper motor.

6. Το stepper motor: Οι πόλοι του stepper motor ενεργοποιούνται σειριακά από τις εξόδους του ULN2003 για την επίτευξη της κίνησης.

7. Τέλος ο υπολογιστής είναι συνδεδεμένος με τη webcam μέσω USB, για τον έλεγχο της λήψης φωτογραφιών από συγκεκριμένες θέσεις κίνησης.

3.2 Ο μικροελεγκτής ATMEGA8515

Με το development kit STK 500 το οποίο χρησιμοποιήθηκε στον προγραμματισμό του μικροελεγκτή της παρούσας εργασίας, ο συμπεριλαμβανόμενος μικροελεγκτής είναι ο AT90S8515. Ο συγκεκριμένος μικροελεγκτής AVR ήταν από τους πρώτους μικροελεγκτές της συγκεκριμένης αρχιτεκτονικής

Για τις ανάγκες της παρούσας εργασίας ο AT90S8515 αντικαταστάθηκε από την επόμενη έκδοσή του , τον ATMEGA8515, λόγω της μη συμβατότητας του προηγούμενου με τα τελευταία εργαλεία προγραμματισμού της ATMEL.

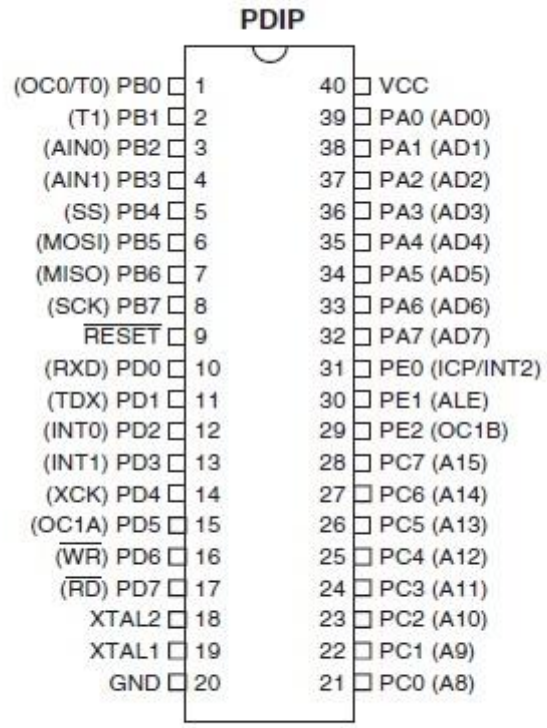
3.2.1 Η αρχιτεκτονική του ATMEGA8515

Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, ο ATmega8515 είναι ένας 8 bit AVR μικροελεγκτής, βασισμένος στην αρχιτεκτονική RISC. Εκτελώντας εντολές σε έναν κύκλο ρολογιού, ο ATmega8515 επιτυγχάνει ταχύτητες του 1 MIPS ανά MHz, επιτρέποντας τη βελτιστοποίηση της κατανάλωσης ισχύος έναντι της υπολογιστικής ταχύτητας.[3] Ο πυρήνας του AVR συνδυάζει ένα πλούσιο instruction set με 32 καταχωρητές γενικής χρήσης. Όλοι οι 32 καταχωρητές είναι απ' ευθείας συνδεδεμένοι με την ALU, επιτρέποντας έτσι σε δύο ανεξάρτητους καταχωρητές να μπορούν να προσπελασθούν από μία εντολή εκτελέσιμη σε έναν κύκλο ρολογιού. Ο ATmega8515 έχει τα παρακάτω χαρακτηριστικά : 8K bytes In-System Programmable Flash (ISP Flash), 512 bytes EEPROM, 512 bytes SRAM, ένα interface εξωτερικής μνήμης, 35 I/Os, 32 καταχωρητές γενικής χρήσης, , δύο Timer/Counters, εσωτερικά και εξωτερικά interrupts, μέσο επικοινωνίας ένα σειριακό προγραμματιζόμενο USART(Universal Synchronous Asynchronous Receiver Transmitter), έναν προγραμματιζόμενο Watchdog Timer με εσωτερικό κρύσταλλο (ρολόι), μία σειριακή θύρα SPI και τρεις επιλογές λειτουργίας για εξοικονόμησης ενέργειας. Η λειτουργία Idle σταματάει τον CPU επιτρέποντας όμως στην SRAM, τους Timer/Counters, το SPI port, και το σύστημα εξυπηρέτησης Interrupt να συνεχίζουν να λειτουργούν. Η λειτουργία Power-down αποθηκεύει τα περιεχόμενα των καταχωρητών αλλά παγώνει το ρολόι, απενεργοποιώντας έτσι όλες τις άλλες λειτουργίες μέχρι να κληθεί κάποιο Interrupt ή να γίνει Reset. Τέλος η λειτουργία Standby επιτρέπει στο ρολόι να λειτουργεί όσο όμως η υπόλοιπη συσκευή κοιμάται. Αυτό επιτρέπει πολύ γρήγορο startup σε συνδυασμό με χαμηλή κατανάλωση ισχύος.

Όσο αφορά τις μνήμες του μικροελεγκτή, η ISP Flash επιτρέπει στην προγραμματιζόμενη μνήμη του συστήματος να μπορεί να αναπρογραμματιστεί ,μέσω μίας SPI διεπαφής είτε από έναν συμβατικό προγραμματιστή μνήμης (STK 500 πχ) , ή από ένα πρόγραμμα boot το οποίο τρέχει στον ίδιο το μικροελεγκτή.

Περιγραφή των I/O pins

Εικόνα: Ο μικροελεγκτής ATMEGA8515 σε 40 pin PDIP μορφή με τις λειτουργίες του κάθε pin.



Εικόνα 2: Το pinout του ATmega8515

VCC

Παροχή τάσης VCC.

GND

Ground.

Ports A (PA7...PA0), B (PB7…PB0), C (PC7…PC0), D (PD7…PD0), E (PE2…PC0)

Τα Ports του μικροελεγκτή είναι 8-bit bi-directional I/O ports με εσωτερικούς αντιστάτες,(pull-up resistors) για κάθε bi(μόνο το Port E είναι 3 bit)t.Οι buffers εξόδου των Ports, έχουν την ικανότητα να μεταφέρουν ένα φορτίο εξίσου προς το VCC(up) και το Ground(low). Η οδήγηση ενός φορτίου προς το VCC ονομάζεται source, ενώ προς το Ground ονομάζεται sink. Η ικανότητα μεταφοράς φορτίου το ίδιο καλά και προς τις δύο διευθύνσεις ονομάζεται συμμετρική οδήγηση φορτίου και δίνει τη δυνατότητα στο μικροελεγκτή να οδηγεί ρεύμα μέχρι και 40mA ανά pin [4],[5]. Όταν τα pins Px ως Px7 χρησιμοποιηθούν σαν είσοδοι και είναι εξωτερικά low, θα πάρουν ρεύμα αν οι εσωτερικοί αντιστάτες είναι ενεργοί. Τα pins των ports είναι tri-stated σε κάθε reset, ακόμα και αν το ρολόι δεν λειτουργεί.

RESET

Η είσοδος Reset. Ένα σήμα low σ' αυτό το pin με διάρκεια για λίγο περισσότερο από έναν παλμό ρολογιού θα παράγει ένα Reset ακόμα και αν το ρολόι δεν λειτουργεί.

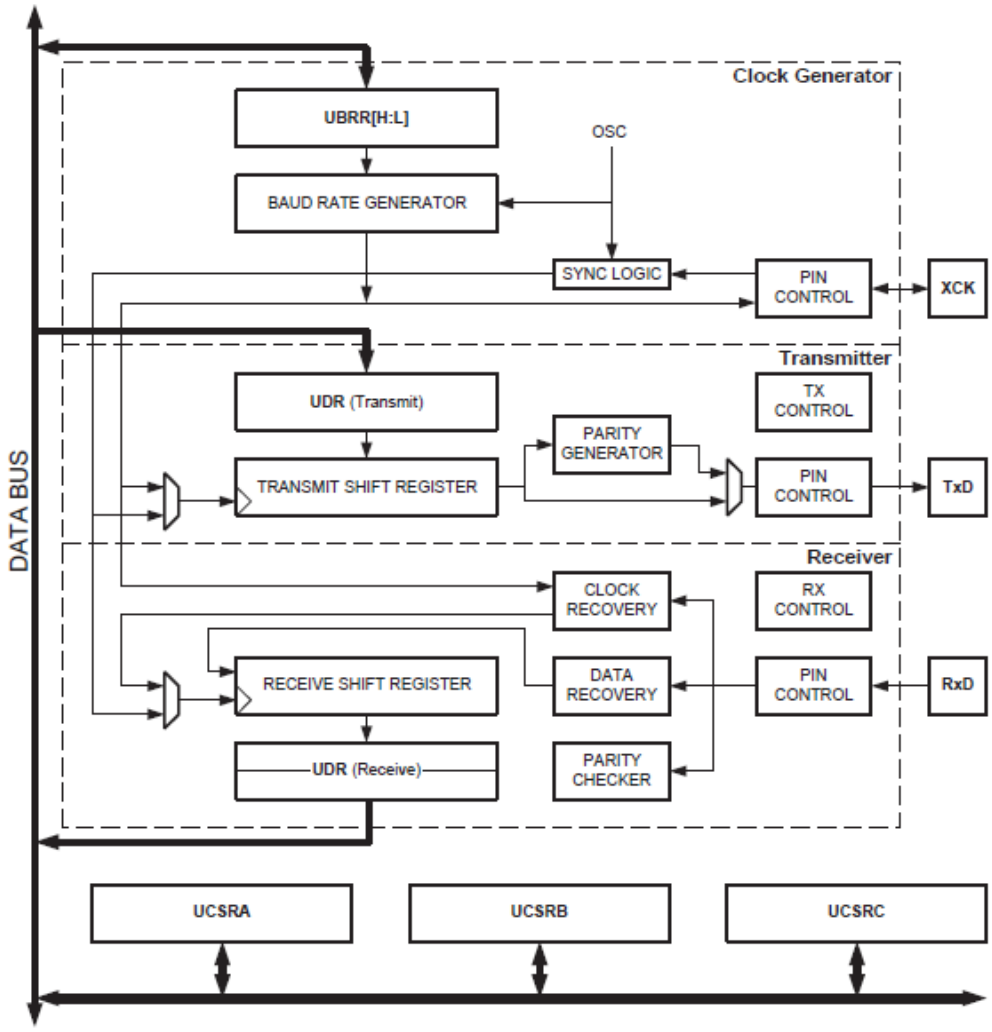
XTAL1

Είσοδος προς τα ρολόγια του μικροελεγκτή
XTAL2

Έξοδος από τα ρολόγια του μικροελεγκτή

3.2.2 Το USART του ATmega8515

Για τη σειριακή επικοινωνία μεταξύ του μικροελεγκτή και του υπολογιστή, χρησιμοποιήθηκε το USART του ATmega8515 μέσω της σειριακής θύρας RS232 του STK 500. Ο atmega8515 έχει ένα μόνο USART. Στο παρακάτω σχήμα απεικονίζεται το block diagram του .



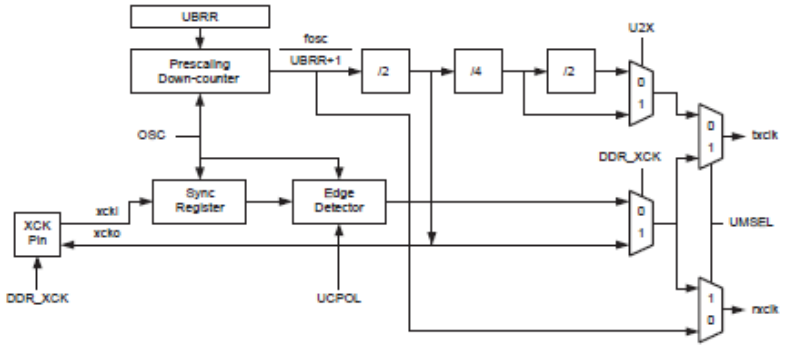
Εικόνα 3: Atmega8515 USART block diagram

Τα τρία τμήματα που διαχωρίζονται στο block diagram αντιστοιχούν στα τρία βασικά μέρη του USART. Από πάνω προς τα κάτω είναι το ρολόι, ο πομπός(transmitter) και ο δέκτης (receiver). Οι καταχωρητές ελέγχου χρησιμοποιούνται και από τα τρία μέρη. Η δημιουργία παλμών ρολογιού αποτελείται από μία λογική συγχρονισμού για εξωτερικό ρολόι που χρησιμοποιείται στη σύγχρονη λειτουργία του USART σαν slave καθώς και από τη γεννήτρια του Baud rate. Το pin XCK(Transfer Clock) χρησιμοποιείται μόνο στη λειτουργία σύγχρονης μετάδοσης. Ο Transmitter αποτελείται από ένα απλό write buffer, ένα σειριακό Shift Register, έναν parity generator και λογική ελέγχου για τη διαχείριση διαφορετικών σειριακών format.Ο write buffer επιτρέπει τη συνεχόμενη μεταφορά δεδομένων χωρίς καθυστέρηση ανάμεσα στα frames. Το πιο πολύπλοκο κομμάτι της αρχιτεκτονικής του USART είναι ο Receiver εξ αιτίας των μονάδων ρολογιού και ανάκτησης δεδομένων που διαθέτει. Η μονάδα

ανάκτησης δεδομένων χρησιμοποιείται για την ασύγχρονη λήψη δεδομένων. Εκτός από τη μονάδα ανάκτησης, ο Receiver περιλαμβάνει επίσης έναν Parity Checker, μία λογική ελέγχου, έναν Shift Register και ένα UDR.

Η μονάδα παραγωγής παλμών ρολογιού:

Η λογική παραγωγής παλμών παράγει το βασικό ρολόι για τον Transmitter και τον Receiver. Το USART υποστηρίζει τέσσερεις λειτουργίες ρολογιού: normal asynchronous, double speed asynchronous, master synchronous slave synchronous. Στον καταχωρητή UCSRC, το UMSEL bit επιλέγει μεταξύ σύγχρονης και ασύγχρονης λειτουργίας. Η λειτουργία double speed ελέγχεται από το bit U2X στον ίδιο καταχωρητή. Στη σύγχρονη λειτουργία ο Data Direction Register για το XCK pin (DDR_XCK) καθορίζει αν το σήμα ρολογιού είναι εσωτερικό ή εξωτερικό (master και slave mode αντίστοιχα). Το XCK pin είναι ενεργό μόνο σε σύγχρονη λειτουργία.



Εικόνα 4: Το block diagram του ρολογιού.

Η γεννήτρια του Baud Rate

Η γεννήτρια του εσωτερικού ρολογιού χρησιμοποιείται για τη σύγχρονη και την ασύγχρονη λειτουργία master. Ο καταχωρητής USART Baud Rate Register (UBRR) και ο συνδεδεμένος με αυτόν down-counter, λειτουργούν σαν έναν προγραμματιζόμενο prescaler ή αλλιώς baud rate generator. Ο down-counter, ο οποίος τρέχει με τη συχνότητα του ρολογιού του συστήματος, (fosc), φορτώνει την τιμή του UBRR κάθε φορά που φτάνει στο μηδέν, η κάθε φορά που γράφεται κάτι στον καταχωρητή UBRRL Register. Παλμός ρολογιού παράγεται κάθε φορά που ο down counter φτάνει στο μηδέν. Το ρολόι αυτό είναι η έξοδος του baud rate generator (= fosc/(UBRR+1)). Ο Transmitter διαιρεί την τιμή αυτή του ρολογιού με το 2, 8 ή 16 ανάλογα το mode. Η έξοδος του baud rate generator output χρησιμοποιείται απ' ευθείας από τις μονάδες ρολογιού και ανάκτησης δεδομένων του Receiver. Παρόλα αυτά , οι μονάδες ανάκτησης χρησιμοποιούν μία μηχανή καταστάσεων με 2, 8, ή 16 καταστάσεις ανάλογα με το mode που καθορίστηκε από τα UMSEL, U2X, and DDR_XCK bits.
Στον πίνακα φαίνεται ο υπολογισμός του baud rate ή του UBRR για τα πιθανά modes:

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal mode (U2X = 0)	$BAUD = \frac{f_{OSC}}{16(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed mode (U2X = 1)	$BAUD = \frac{f_{OSC}}{8(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master mode	$BAUD = \frac{f_{OSC}}{2(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{2BAUD} - 1$

Εικόνα 5: Πίνακας υπολογισμού Baud Rate

Για την υλοποίηση σειριακής επικοινωνίας στην παρούσα διπλωματική χρησιμοποιήθηκε το πρώτο mode λειτουργίας και στην αρχικοποίηση του USART χρησιμοποιήθηκε ο τύπος υπολογισμού της τιμής του καταχωρητή UBRR και στη συνέχεια η τιμή αυτή αποθηκεύτηκε στον UBRR με σκοπό τον ορισμό του επιθυμητού Baud Rate.

Αρχικοποίηση του USART

Πριν κάθε επικοινωνία, το USART πρέπει να αρχικοποιηθεί. Η αρχικοποίηση αφορά τον ορισμό του baud rate, του frame format και την ενεργοποίηση του Transmitter ή του Receiver ανάλογα τη χρήση. Για λειτουργίες του USART που χρησιμοποιούν interrupts, τα interrupts πρέπει να είναι ανενεργά κατά τη διάρκεια της αρχικοποίησης.

Σε περίπτωση επόμενης αρχικοποίησης με διαφορετικά τα baud rate ή frame format, δεν πρέπει να υπάρχουν ενεργές μεταδόσεις πληροφορίας κατά τη χρονική περίοδο που αλλάζουν τιμές οι Registers. Το TXC Flag μπορεί να χρησιμοποιηθεί για να ελέγξει ότι ο Transmitter έχει εκτελέσει όλες τις μεταφορές δεδομένων, ενώ το RXC Flag μπορεί να χρησιμοποιηθεί για να ελέγξει ότι δεν υπάρχουν μη αναγνωσμένα δεδομένα στο δίαυλο λήψης. Αξίζει να σημειωθεί ότι το TXC Flag πρέπει να καθαρίζει πριν κάθε μετάδοση και πριν γραφτεί το UDR, στην περίπτωση που χρησιμοποιηθεί με το σκοπό που προαναφέρθηκε.

Ακολουθεί ένα παράδειγμα αρχικοποίησης του USART σε γλώσσα C, όπως έγινε και στην υλοποίηση αυτής της διπλωματικής. Το παράδειγμα υποθέτει ασύγχρονη λειτουργία χωρίς ενεργά interrupts και ένα προκαθορισμένο frame format. Το baud rate δίνεται σαν παράμετρος συνάρτησης. Όταν η συνάρτηση γράφει στον καταχωρητή UCSRC, το URSEL bit (MSB) πρέπει να καθοριστεί λόγω της από κοινού χρήσης της θέσης I/O από τους UBRRH και UCSRC καταχωρητές:

```
void USART_Init( unsigned int baud )
{
/* Set baud rate */
        UBRRH = (unsigned char)(baud>>8);
        UBRRL = (unsigned char)baud;
/* Enable receiver and transmitter */
        UCSRB = (1<<RXEN)|(1<<TXEN);
/* Set frame format: 8data, 2stop bit */
        UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);
}
```

Πιο πολύπλοκες ρουτίνες αρχικοποίησης μπορεί να περιλαμβάνουν ως παραμέτρους το frame format ή να απενεργοποιούν τα interrupts κλπ. Παρόλα αυτά, σε πολλές εφαρμογές χρησιμοποιούνται συγκεκριμένες ρυθμίσεις στους καταχωρητές Baud και Control, και για αυτούς τους τύπους εφαρμογών ο κώδικας αρχικοποίησης μπορεί να γραφεί απ' ευθείας στη συνάρτηση main ή να συνδυαστεί με τον κώδικα αρχικοποίησης άλλων επί μέρους modules I/O.

Αποστολή δεδομένων – Ο USART Transmitter

Ο USART Transmitter ενεργοποιείται με τον ορισμό του *Transmit Enable* (TXEN) bit στον καταχωρητή UCSRB. Όταν ο Transmitter είναι ενεργός, η φυσική λειτουργία του TxD pin ακυρώνεται γιατί το εν λόγω pin χρησιμοποιείται απ' το USART για τη σειριακή έξοδο του Transmitter. Τα baud rate, mode λειτουργίας και frame format, πρέπει να ρυθμιστούν πριν από οποιαδήποτε αποστολή δεδομένων.

Αποστολή πακέτων με 5 ως 8 Data Bits

Μία μετάδοση δεδομένων αρχίζει φορτώνοντας στον transmit buffer τα δεδομένα προς αποστολή. Ο μικροελεγκτής φορτώνει τον transmit buffer γράφοντας στον UDR I/O . Τα δεδομένα στον transmit buffer θα μεταφερθούν στον shift register μόλις ο shift register είναι έτοιμος να στείλει ένα νέο πακέτο δεδομένων. Ο shift register φορτώνει νέα δεδομένα είτε στην κατάσταση Idle (δηλαδή όταν δεν υπάρχει ενεργή αποστολή δεδομένων) είτε αμέσως μετά την αποστολή του τελευταίου stop bit του προηγούμενου πακέτου. Μόλις ο shift register πάρει τα νέα δεδομένα , μεταφέρει ένα πλήρες πακέτο με ρυθμό που καθορίζεται από το U2X bit του Baud Register ή από το XCK, ανάλογα με το mode λειτουργίας.

Το παρακάτω παράδειγμα κώδικα παρουσιάζει μία απλή συνάρτηση transmit USART βασισμένη σε polling του (UDRE) Flag. Χρησιμοποιώντας μοντέλα πακέτων με λιγότερα από οκτώ bits, τα most significant bits του UDR αγνοούνται. Το USART πρέπει να αρχικοποιηθεί πριν τη χρήση αυτής της συνάρτησης. Η συνάρτηση περιμένει τον transmit

buffer να αδειάσει ελέγχοντας το UDRE Flag, πριν γεμίσει με νέα δεδομένα προς αποστολή. Αν χρησιμοποιηθεί το Data Register Empty Interrupt, η ρουτίνα του interrupt γράφει τα δεδομένα στον buffer.

C Code Example(1)

```
void USART_Transmit( unsigned char data )
{
/* Wait for empty transmit buffer */
while ( !( UCSRA & (1<<UDRE)) )
;
/* Put data into buffer, sends the data */
UDR = data;
}
```

Λήψη δεδομένων – Ο USART Receiver

Ο USART Receiver ενεργοποιείται θέτοντας το Receive Enable (RXEN) bit στον UCSRB Register σε '1'. Όταν ο Receiver είναι ενεργός, η οποιαδήποτε φυσική λειτουργία του RxD pin του μικροελεγκτή επικαλύπτεται από τη λειτουργία του USART και λαμβάνει τη λειτουργία της σειριακής εισόδου του Receiver. Όπως και στην περίπτωση του Transmitter, το baud rate, το mode of operation και το frame format(η διαμόρφωση των πακέτων δεδομένων) πρέπει να προκαθοριστούν πριν από οποιαδήποτε λήψη δεδομένων. Σε σύγχρονη λειτουργία, το ρολόι του XCK pin χρησιμοποιείται σαν ρολόι μεταφοράς δεδομένων.

Λήψη πακέτων με 5 ως 8 Data Bits

Ο Receiver ξεκινάει λήψη δεδομένων μόλις ανιχνεύσει ένα έγκυρο start bit. Το κάθε επόμενο bit δειγματοληπτείται είτε στο baud rate ή στο ρολόι XCK, και μεταφέρεται στον Shift Register μέχρι τη λήψη του stop bit του πρώτου πακέτου. Όταν το πρώτο stop bit φτάσει, δηλαδή όταν έχει φτάσει πλήρως ένα πακέτο, τα περιεχόμενα του Shift Register

μεταφέρονται στον receive buffer. Ο receive buffer τότε μπορεί να διαβαστεί διαβάζοντας το UDR I/O.

Ο παρακάτω κώδικας δείχνει ένα απλό παράδειγμα της συνάρτησης λήψης του USART, βασισμένος σε polling του Receive Complete (RXC) Flag. Για πακέτα με λιγότερα από 8 bits, τα most significant bits του των δεδομένων του UDR θεωρούνται 0. Το USART πρέπει να αρχικοποιηθεί πριν τη χρήση της συνάρτησης:

```
unsigned char USART_Receive( void )
{
/* Wait for data to be received */
while ( !(UCSRA & (1<<RXC)) )
;
/* Get and return received data from buffer */
return UDR;
}
```

Η συνάρτηση περιμένει να υπάρξουν δεδομένα στον receive buffer ελέγχοντας το RXC Flag, πριν διαβάσει τον buffer και επιστρέψει την τιμή.

.

3.3 Το Stepper Motor

Για την επιλογή του κατάλληλου stepper motor έπρεπε να ληφθούν υπ' όψιν παράμετροι όπως:

Παράμετροι φορτίου

α) οι διαστάσεις του φορτίου

β) η μάζα του φορτίου

Επίσης έπρεπε να ληφθούν υπ' όψιν και οι παρακάτω παράμετροι λειτουργίας :

α) η ταχύτητα και ο χρόνος λειτουργίας

β) η ακρίβεια στη στάση

γ) τα steps per revolution

δ) η τάση και η ισχύς τροφοδοσίας

ε) το περιβάλλον λειτουργίας

στ) ο stepper motor driver

Δεδομένου του ότι στα πλαίσια αυτής της διπλωματικής, η πλατφόρμα που κατασκευάστηκε και το φορτίο το οποίο τοποθετήθηκε επάνω είναι της τάξεως γραμμαρίων, από άποψη μάζας δεν τέθηκε λόγος επιλογής stepper motor με μεγάλη ροπή κίνησης. Παρόλα αυτά σε πιθανές εφαρμογές της διπλωματικής αυτής με μεγαλύτερη έκταση, θα πρέπει να ληφθούν υπ' όψιν τα παραπάνω δεδομένα αναφορικά με τη μάζα και τις διαστάσεις του φορτίου ώστε να είναι βέλτιστο το stepper motor από άποψη παραγόμενης ροπής.

Μία σκέψη για χρήση της ίδιας εφαρμογής σε μεγαλύτερα φορτία χωρίς αλλαγές στο stepper motor, θα ήταν το stepper αντί να περιστρέφει την πλατφόρμα να περιστρέφει μόνιμα την κάμερα σε μία τροχιά 360 μοιρών γύρω από το έκθεμα, ώστε να μην χρειάζεται συνεχής

υπολογισμός της απαραίτητης ροπής που θα πρέπει να έχει το εκάστοτε stepper motor, παρά μόνο μία φορά υπολογισμός, λαμβάνοντας υπ' όψιν μόνο τη μάζα της κάμερας. Έτσι η εφαρμογή σε μεγαλύτερη έκταση γίνεται λιγότερο ακριβή και πιο εύκολα υλοποιήσιμη.

Για τις ανάγκες της διπλωματικής αυτής επιλέχθηκε το 28BYJ-48, stepper motor 5 VDC τεσσάρων πόλων. Κατόπιν μελέτης της κίνησής του, παρατηρήθηκε ότι οι συνεχόμενες ενεργοποιήσεις και απενεργοποιήσεις μαγνητικών πόλων εντός του stepper, που αντιστοιχούν σε μία περιστροφή είναι 2048, παρά το γεγονός ότι ο κατασκευαστής αναφέρει 4096. Ο κατασκευαστής συγκεκριμένα, αναφέρει ότι στο συγκεκριμένο stepper motor τα steps per revolution είναι 64, όμως το εν λόγω stepper motor έχει ενσωματωμένο σύστημα γραναζιών στο εσωτερικό με λόγο πολλαπλασιασμού 1/64, πράγμα που σημαίνει ότι οι πιθανές θέσεις στο μέγιστο της ακρίβειας του stepper είναι 64*64=4096 θέσεις. Παρόλα αυτά, παρατηρήθηκε ότι για μία πλήρη περιστροφή χρειάζονται 512 συνεχείς ενεργοποιήσεις και απενεργοποιήσεις των $4^{ων}$ πόλων, πράγμα που σημαίνει ότι τα πιθανά steps στη μέγιστη ανάλυση που μπορεί να προσφέρει το stepper motor είναι 2048. Στην περίπτωση που είχε χρησιμοποιηθεί half stepping, τότε θα υπήρχαν 4096 half steps με τη διπλάσια ανάλυση Αυτό σημαίνει ότι θεωρητικά μπορούμε να έχουμε ακρίβεια μέχρι και 360/4096=0.088 μοίρες με half steps ενώ 360.2048=0.176 μοίρες με full steps. Δεν χρησιμοποιήθηκε τέτοιου μεγέθους ακρίβεια στην εν λόγω διπλωματική, θεωρήθηκε πως 64 steps per revolution είναι αρκετά, επομένως η ακρίβεια της λειτουργίας του stepper motor είναι 5.625 μοίρες. Η ροπή του stepper motor που επιλέχθηκε είναι 350g/cm, υπέρ αρκετή για τη μάζα του εκθέματος και της πλατφόρμας αθροιστικά.

3.4 Ο Stepper Motor Driver ULN2003

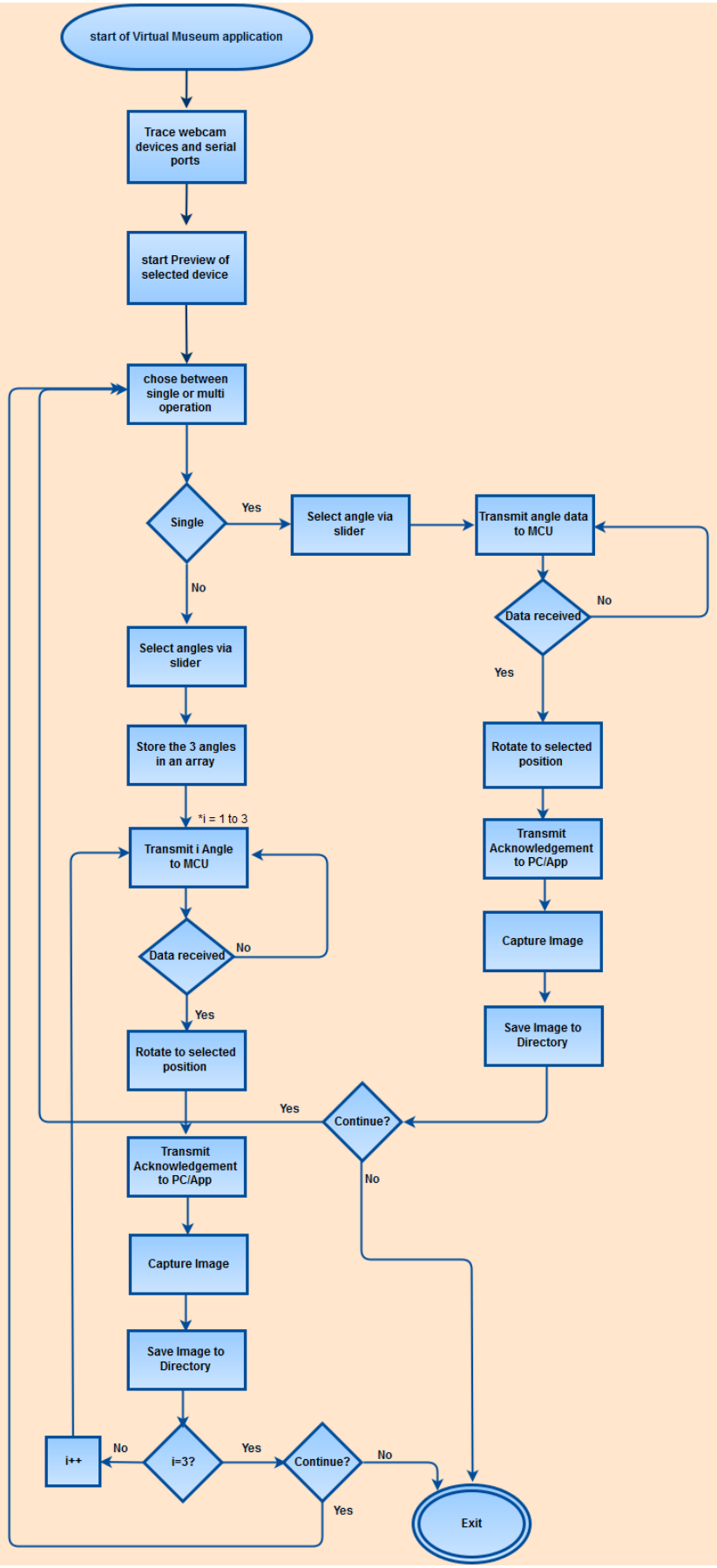
Για το driving του stepper motor χρησιμοποιήθηκε ο stepper motor driver της Adafruit που χρησιμοποιεί ένα chip ULN2003. Ο ULN2003 είναι ένας πίνακας από 7 NPN Darlington transistors ικανά να παρέχουν έξοδο έως και 500mA ρεύμα και , 50V τάση. Για καλύτερη απομόνωση και αποφυγή τόξων χρησιμοποιεί fly back διόδους, για τις εναλλαγές στα επαγωγικά φορτία. Στην ίδια οικογένεια ανήκουν οι drivers ULN2002A, ULN2004A, ULQ2003A και ULQ2004A. Περισσότερες πληροφορίες για τον ULN2003 και τα Darlington transistor arrays αναφέρονται στο παράρτημα.

3.5 Η γραφική εφαρμογή

Για την υλοποίηση της παρούσας εργασίας αναπτύχθηκε μία πλατφόρμα διαχείρισης της επικοινωνίας μεταξύ του χρήστη και του μικροελεγκτή. Η πλατφόρμα που αναπτύχθηκε είναι της μορφής Windows Presentation Foundation, και οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν για την υλοποίηση είναι η xaml για τη γραφική σχεδίαση και η C# για τις λειτουργίες. Η εφαρμογή αναπτύχθηκε στο περιβάλλον του Microsoft Visual Studio 2015. Η εφαρμογή διαχειρίζεται επίσης την επικοινωνία του υπολογιστή στον οποίο τρέχει με τη μικροελεγκτή και με τη web camera. Η εφαρμογή μπορεί να έχει πρόσβαση σε οποιαδήποτε web camera είναι συνδεδεμένη στον υπολογιστή με τη βοήθεια της βιβλιοθήκης ανοιχτού κώδικα WebcamControl.dll[6]. Ο χρήστης είναι σε θέση να επιλέξει τη σειριακή θύρα στην οποία είναι συνδεδεμένος ο μικροελεγκτής. Η επιλογή αυτή υπάρχει για την περίπτωση που μελλοντικά συνδεθούν περισσότερα από ένα συστήματα Ιδεατού Μουσείου στον ίδιο υπολογιστή.

Ο χρήστης καλείται να επιλέξει ανάμεσα σε απλή λειτουργία ή πολλαπλή λειτουργία. Στην απλή λειτουργία, επιλέγεται μία γωνία πολλαπλάσια του step (5.625 μοίρες) με τη βοήθεια

μίας κυλιόμενης μπάρας (slider). Με την ενεργοποίηση του κουμπιού λήψης, η γραφική εφαρμογή στέλνει δεδομένα που αφορούν τη θέση στην οποία πρέπει να βρεθεί το Stepper Motor για τη λήψη της φωτογραφίας στο μικροελεγκτή μέσω της σειριακής θύρας RS232. Ο μικροελεγκτής στη συνέχεια δίνει εντολή στο Stepper να αρχίσει να περιστρέφεται μέχρι να φτάσει στην επιλεγμένη θέση, και μόλις αυτό φτάσει, αποστέλλεται επιβεβαίωση μέσω της σειριακής θύρας πίσω στην εφαρμογή. Με τη λήψη της επιβεβαίωσης η εφαρμογή δίνει εντολή στην επιλεγμένη web camera να κάνει capture ένα snapshot, το οποίο αποθηκεύεται στο επιλεγμένο από την εφαρμογή directory. Έπειτα ο χρήστης μπορεί να επιλέξει επόμενη θέση λήψης. Στην επιλογή πολλαπλής λειτουργίας, ο χρήστης μπορεί να επιλέξει τρεις πιθανές γωνίες λήψης, και η παραπάνω διαδικασία επαναλαμβάνεται τρεις φορές. Και τα τρία νέα snapshots αποθηκεύονται στον επιλεγμένο φάκελο. Ακολουθεί flowchart των ενεργειών που εκτελούνται κατά τη διαχείριση της εφαρμογής.



Εικόνα 6: Flowchart ενεργειών

4.
ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ

4.1 Ο προγραμματισμός του ATmega8515

Ο προγραμματισμός του μικροελεγκτή έγινε με το εργαλείο της ATMEL, Atmel Studio 6.2 σε γλώσσα προγραμματισμού C. Το development kit STK 500 το οποίο χρησιμοποιήθηκε, προσφέρει δυνατότητες δύο διαφορετικών ειδών προγραμματισμού του μικροελεγκτή, την ISP(In System Programming) και HVP(High Voltage Programming). Στην υλοποίηση της παρούσας εφαρμογής χρησιμοποιήθηκε ISP προγραμματισμός, κάτι που δίνει τη δυνατότητα στο μικροελεγκτή να προγραμματίζεται ήδη εγκατεστημένος στο σύστημα το οποίο πρόκειται να ελέγξει, στη συγκεκριμένη περίπτωση συνδεδεμένος με το Stepper Motor μέσω του Stepper Motor Driver ULN2003. Παρακάτω θα αναλυθούν τα σχεδιαστικά βήματα που ακολουθήθηκαν για τον προγραμματισμό του ATmega8515 στην τελική μορφή του συστήματος.

Αξίζει σ' αυτό το σημείο να αναφερθούν κάποιες βασικές εντολές του προγραμματισμού ενός μικροελεγκτή AVR, οι οποίες ανήκουν στη βιβλιοθήκη της C, <avr/io.h>.

Η εντολή DDRx

Η εντολή DDRx , όπου x το όνομα ενός από τα 5 Ports του AVR όπως αναφέρθηκαν στο κεφάλαιο 3, καθορίζει ποια pins του επιλεγμένου Port θα θεωρηθούν είσοδοι και ποια έξοδοι. Αυτό μπορεί να γίνει με δύο τρόπους, είτε σε δυαδική κωδικοποίηση είτε σε δεκαεξαδική. Γενικά, η θέση ενός pin στην εντολή DDR ως '1' ορίζει το συγκεκριμένο pin ως έξοδο, ενώ το '0' ως είσοδο. Για να γίνει πιο αντιληπτό αυτό, αξίζει να αναφερθεί ως παράδειγμα η παρούσα υλοποίηση. Στην παρούσα υλοποίηση έχει χρησιμοποιηθεί το PortA για την κίνηση του Stepper Motor. Δηλαδή τα pins του PortA δίνουν τα σήματα κίνησης προς τα πηνία του Stepper. Έτσι έπρεπε να τεθούν ως έξοδοι. Γι' αυτό το λόγο, η εντολή DDRx κωδικοποιήθηκε ως εξής: DDRA=0xFF. Είναι εμφανές ότι η μορφή της εντολής είναι δεκαεξαδική. Το αντίστοιχο σε δυαδική μορφή θα ήταν DDRA=0b11111111. Στην περίπτωση που υπήρχε η ανάγκη να χρησιμοποιηθεί ένα Port ως είσοδος, η εντολή θα ήταν διαμορφωμένη ως εξής : DDRx= 0x00 ή DDRx=0b00000000. Υπάρχει επίσης η περίπτωση κάποια pins ενός Port να χρησιμοποιηθούν ως είσοδοι και κάποια ως έξοδοι : DDRx= 0x0F ή DDRx=0b00001111, κάτι που θα καθιστούσε τα πρώτα 4 pins(Px0…Px3) του Port x ως εξόδους και τα τελευταία 4 ως εισόδους (Px4…Px7).

Η εντολή PORTx

Η εντολή PORTx, όπου x το όνομα ενός από τα 5 Ports του AVR όπως αναφέρθηκαν στο κεφάλαιο 3, καθορίζει τις τιμές που θα πάρουν τα pins εξόδου ενός Port με ίδια κωδικοποίηση όπως προηγουμένως, είτε σε δυαδικό είτε σε δεκαεξαδικό σύστημα. Στην περίπτωση της παρούσας εργασίας για παράδειγμα, το PortA που έχει το ρόλο της μετάδοσης των σημάτων κίνησης του Stepper, παίρνει τη μορφή PORTA=0b00000001 όταν πρέπει να ενεργοποιηθεί το 1^{ο} πηνίο, PORTA=0b00000100 για το 2^{ο} και ου το καθεξής (ο λόγος για τον οποίο η δεύτερη ενεργοποίηση είναι 0b00000100 και όχι 0b00000010 είναι καθαρά θέμα τοποθέτησης των pins πάνω στο STK 500 και θα αναλυθεί περισσότερο παρακάτω).

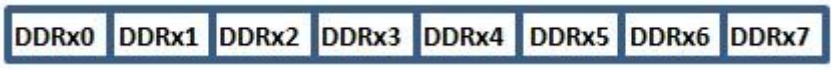
Η εντολή PINx

Επίσης υπάρχει και η εντολή PINx η οποία όμως δεν χρησιμοποιήθηκε στην υλοποίηση της παρούσας εφαρμογής. Η εντολή PINx χρησιμοποιείται για να διαβάσει από τα Pins των Ports

που έχουν καταχωρηθεί ως είσοδοι με την εντολή DDRx. Η ανάγνωση των Pins γίνεται με τον ίδιο τρόπο που περιγράφηκε στις άλλες δύο εντολές.[9],[10]

Οι εντολές DDR, PORT και PIN απευθύνονται σε τρεις καταχωρητές[9], που ονομάζονται I/O registers:

DDRx - Port X Data Direction Register

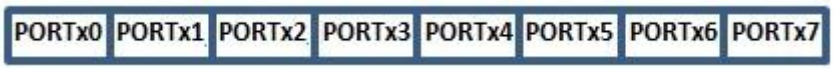


Ο DDRx είναι ένας 8-bit καταχωρητής, ο οποίος αποθηκεύει πληροφορίες ρυθμίσεων για τα pins του κάθε Port. Όπως αναφέρθηκε παραπάνω, γράφοντας ‘1’ στη θέση ενός pin στον καταχωρητή DDRx κάνει τη φυσική λειτουργία του συγκεκριμένου pin να είναι έξοδος, ενώ το ‘0’ την κάνει είσοδο.

PINx - Port X Input Pins Register

Ο καταχωρητής PINx είναι ένας καταχωρητής 8-bit ο οποίος αποθηκεύει τη λογική τιμή και την τρέχουσα κατάσταση των Pins του κάθε port. Οπότε για να διαβάσει ο επεξεργαστής τις τιμές στο Portx, πρέπει πρώτα να διαβάσει τις τιμές στον καταχωρητή PIN.

PORTx - Port X Data Register



Ο καταχωρητής PORTx είναι ένας 8-bit καταχωρητής ο οποίος αποθηκεύει τις λογικές τιμές εξόδου στα pins του καθορισμένου Port, αν τα pins αυτά έχουν ορισθεί ως έξοδοι. Οπότε, για να δοθούν τιμές στα pins ενός Port πρέπει να δοθούν τιμές στον καταχωρητή PORT αυτού του συγκεκριμένου Port.

4.1.1 Συναρτήσεις κίνησης του Stepper Motor

Όπως αναφέρθηκε στο κεφάλαιο 3, ένα stepper motor περιστρέφεται με συνεχόμενη ενεργοποίηση και απενεργοποίηση των πόλων του. Στο stepper motor που επιλέχθηκε και χρησιμοποιήθηκε για την υλοποίηση της παρούσας εφαρμογής, παρατηρήθηκε ότι 8 συνεχείς ενεργοποιήσεις και των 4^{ων} πόλων αντιστοιχούν στην επιθυμητή μετατόπιση των 5,625 μοιρών, και αυτό λόγω του λόγου πολλαπλασιασμού των γραναζιών που υπάρχουν στον άξονα του stepper, όπως περιγράφηκε και στο 3^{ο} κεφάλαιο αναλυτικά. Έτσι προέκυψε η συνάρτηση κίνησης του stepper motor. Για κίνηση αντίθετης φοράς, οι πόλοι του stepper motor απλώς ενεργοποιούνται και απενεργοποιούνται με αντίθετη σειρά. Η συνάρτηση κίνησης κατασκευάστηκε έτσι ώστε μετά από κάθε μετατόπιση σε οποιοδήποτε πολλαπλάσιο των 5,625 μοιρών το stepper motor θα βρίσκεται στη θέση αρχικοποίησης από άποψη κώδικα. Για να γίνει καλύτερα αντιληπτό αυτό, παρατίθεται το ακόλουθο παράδειγμα κώδικα:

(το PORTA αρχικοποιείται στην τιμή 00000001 έξω από τη συνάρτηση κίνησης και τους βρόχους επανάληψης)

```
PORTA=0b00000100;

_delay_ms(20);

PORTA=0b00010000;

_delay_ms(20);

PORTA=0b01000000;

_delay_ms(20);

PORTA=0b00000001;

_delay_ms(20);
```

Στο παραπάνω παράδειγμα, το οποίο αποτελεί τμήμα της συνάρτησης κίνησης του stepper motor, παρουσιάζεται η συνεχόμενη ενεργοποίηση 4^{ων} πόλων. Το κομμάτι αυτό κώδικα,

βρίσκεται σε έναν βρόχο επανάληψης for, και όσες φορές και να εκτελεστεί , η τελευταία κατάσταση του καταχωρητή PORTA θα είναι πάντα η "0b00000001". Έτσι αποφεύχθηκε το πρόβλημα του να σταματήσει η κίνηση σε κάποια ενδιάμεση κατάσταση. Για την αντίστροφη φορά, η ενεργοποίηση γίνεται απλώς με την αντίθετη σειρά. Δηλαδή, αντιστοιχίζοντας τους τέσσερεις πόλους σε αριθμούς από το 0 ως το 3, η ενεργοποίηση τους για αντίρροπη φορά θα είναι η 3-2-1-0-3-2-1-0… κ.ο.κ. Σε κώδικα αυτό διατυπώνεται ως εξής:

```
PORTA=0b01000000;

_delay_ms(20);

PORTA=0b00010000;

_delay_ms(20);

PORTA=0b00000100;

_delay_ms(20);

PORTA=0b00000001;

_delay_ms(20);
```

Κατά την υλοποίηση των συναρτήσεων κίνησης παρουσιάστηκε το εξής πρόβλημα : στην περίπτωση που το stepper βρίσκεται για παράδειγμα στη θέση 0 και καλείται να περιστραφεί στη θέση 4(22,5 μοίρες), εκτελεί κανονικά την κίνηση. Όμως αν η επόμενη θέση είναι πχ η θέση 1(5.625 μοίρες), τότε το stepper έπρεπε να εκτελέσει μία πλήρη περιστροφή, να ξαναφτάσει στο 0 και από εκεί στο 1. Το πρόβλημα αυτό αντιμετωπίσθηκε με την προσθήκη μίας συνάρτησης εντοπισμού της βέλτιστης τροχιάς. Η συνάρτηση ονομάσθηκε findBestRotation και ο ρόλος της είναι ο εξής:

Η εν λόγω συνάρτηση δέχεται ως όρισμα έναν ακέραιο αριθμό, ο οποίος αντιστοιχεί στη θέση στην οποία πρέπει να βρεθεί το stepper motor μετά την ολοκλήρωση της μετατόπισής του. Να σημειωθεί σ' αυτό το σημείο ότι η τρέχουσα θέση (θα ονομαστεί curSteps) του stepper motor είναι αποθηκευμένη σε μία μεταβλητή. Στην περίπτωση που η επόμενη θέση (θα ονομαστεί goal) είναι μεγαλύτερη από την τρέχουσα, τότε αφαιρούνται και αν το αποτέλεσμα είναι

μεγαλύτερο από 32(το μισό της μετατόπισης) τότε η μετατόπιση γίνεται αντίρροπα(counter clockwise) κατά x θέσεις, όπου x = 64-(goal-curSteps). Πχ, εάν το stepper βρίσκεται στη θέση 1(5.625 μοίρες) και κληθεί να μετατοπιστεί στη θέση 63(354.375 μοίρες), για εξοικονόμηση ενέργειας και χρόνου είναι προφανώς προτιμότερο να κινηθεί 2 θέσεις προς τα πίσω παρά να κινηθεί 62 προς τα εμπρός. Έτσι ο αριθμός x θα πάρει την τιμή x=64-(63-1) = 2 και μ αυτόν τον τρόπο το stepper θα κινηθεί προς τα πίσω κατά δύο θέσεις. Εάν το goal-curSteps είναι μικρότερο του 32, τότε η μετατόπιση γίνεται clockwise, κατά goal- curSteps. Για παράδειγμα αν το stepper βρίσκεται στη θέση 1(5.625 μοίρες) και κληθεί να πάει στη θέση 4(22.5 μοίρες), θα εκτελέσει clockwise μετατόπιση κατά 4-1=3 θέσεις. Στην περίπτωση τώρα που το goal είναι μικρότερο από την τρέχουσα θέση, με όμοιο τρόπο αποφασίζεται η κίνηση που θα εκτελέσει το stepper motor. Δηλαδή αν η διαφορά τους είναι μικρότερη του 32, τότε εκτελείται counter clockwise κίνηση κατά τόσες θέσεις όση η διαφορά τους, ενώ αν η διαφορά είναι μεγαλύτερη του 32, τότε εκτελείται clockwise κίνηση κατά 64-(curSteps-goal).

4.1.2 Η συνάρτηση αρχικοποίησης του USART

Στο κεφάλαιο 3 περιγράφηκε αναλυτικά η λειτουργία του USART. Το πρώτο πράγμα που έπρεπε να υλοποιηθεί για τη σωστή λειτουργία της σειριακής επικοινωνίας είναι η συνάρτηση αρχικοποίησης του USART. Στη συνάρτηση αρχικοποίησης του USART ορίσθηκαν οι βασικές πληροφορίες που είναι απαραίτητες για την επικοινωνία. Για τον ορισμό του baud rate χρησιμοποιήθηκε ο καταχωρητής **UBRR**.Ο καταχωρητής **UBRR** είναι καταχωρητής 16 bit, και το high Byte ορίζεται ως **UBRRH** ενώ το low Byte ορίζεται ως **UBRRL**. Πρέπει η τιμή του prescaler(ο ακέραιος αριθμός ο οποίος διαιρεί τη συχνότητα ρολογιού ώστε να φτάσει στο επιθυμητό baud rate) να αποθηκευθεί στον καταχωρητή **UBRR**. Το επιθυμητό baud rate ορίσθηκε 1200, και η πράξη η οποία δίνει την τιμή του **UBRR**, είναι η UBRR=((F_CPU / (BAUD * 16))) - 1. Η συχνότητα του επεξεργαστή ορίσθηκε by default 1MHz, επομένως η τιμή του UBRR που προκύπτει στρογγυλοποιημένη στον πλησιέστερο ακέραιο είναι 51. Η τιμή 51 αποθηκεύεται στον 16 bit καταχωρητή ως εξής: η τιμή στον **UBRRH** θα είναι 0x00, ενώ η τιμή στον **UBRRL** θα είναι 0x33.

Για την αρχικοποίηση του USART χρησιμοποιήθηκε ο καταχωρητής USART Control and Status Register B, **UCSRB**. Τα δύο MSB του συγκεκριμένου καταχωρητή αφορούν τη χρήση των Interrupts των Transmit και Receive. Δεδομένου ότι στην υλοποίηση της παρούσας εφαρμογής δεν χρησιμοποιήθηκαν Interrupts, η αρχικοποίηση του καταχωρητή **UCSRB** έγινε ως εξής: UCSRB=(1<<RXEN)|(1<<TXEN);

Η συγκεκριμένη γραμμή κώδικα ενεργοποιεί τα bits του καταχωρητή **UCSRB TXEN** και **RXEN**, για το transmit και το receive αντίστοιχα. Πέραν της ενεργοποίησης του transmit και του receive, κατά την αρχικοποίηση της σειριακής επικοινωνίας πρέπει να ορισθούν δεδομένα αναφορικά με τη μορφή των πακέτων που αποστέλλονται ή λαμβάνονται. Λόγω της ρύθμισης του baud rate στο 1200, θεωρήθηκε βέλτιστο για μείωση σφαλμάτων η χρήση 2 stop bits στα 8 bit των δεδομένων. Επομένως τα πακέτα(frames) που αποστέλλονται ή λαμβάνονται έχουν συνολικό μέγεθος 11 bit , τα οποία αναλύονται σε 1start bit, 8 data bits και 2 stop bits. Το παραπάνω frame format αποθηκεύεται στον καταχωρητή **UCSRC**. Στο σημείο αυτό είναι σημαντικό να αναφερθεί ότι οι καταχωρητές **UCSRC** και **UBRRH** έχουν την ίδια διεύθυνση. Επομένως η επιλογή του σε ποιον από τους δύο γράφονται δεδομένα καθορίζεται από το bit 7. Αν το bit 7 είναι ‘1’ τα δεδομένα αποθηκεύονται στον **UCSRC** ενώ αν είναι ‘0’ , στον **UBRRH**[11]. Το bit 7 του **UCSRC** ονομάζεται **URSEL** και στην παρούσα υλοποίηση τίθεται ‘1’. Όπως προαναφέρθηκε, χρησιμοποιήθηκαν 2 stop bits. Ο ορισμός της χρήσης 2 stop bits στο πακέτο των δεδομένων ορίζεται στο bit 3 του καταχωρητή **UCSRC**, με όνομα **USBS**. Το bit **USBS** τίθεται ‘0’ για χρήση ενός stop bit, ενώ τίθεται ‘1’ για χρήση 2. Στην παρούσα υλοποίηση ορίσθηκε ‘1’ για τη χρήση 2 stop bits όπως προαναφέρθηκε. Κάτι ακόμα που αξίζει να αναφερθεί , είναι ότι στη γλώσσα C για MCU, ένας χαρακτήρας (char) έχει μέγεθος 8 bit. Στην παρούσα υλοποίηση, τα δεδομένα που αποστέλλονται και λαμβάνονται μεταξύ PC και MCU έχουν μήκος ενός χαρακτήρα. Για τον ορισμό των data bits της μετάδοσης, πρέπει να ορισθούν πρώτα τα **UCSZ1, UCSZ0**(bits 2 και 1 του **UCSRC** αντίστοιχα)**.** Για χρήση 8 data bits τα παραπάνω bits πρέπει να ορισθούν και τα δύο ως ‘1’, επομένως η τελική μορφή του καταχωρητή **UCSRC** για τη μορφή του πακέτου δεδομένων, ορίζεται στον κώδικα ως: UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);

4.1.3 Η συνάρτηση λήψης δεδομένων

Για τη λήψη δεδομένων χρησιμοποιήθηκε η συνάρτηση λήψης χαρακτήρα usart_getchar. Πριν τη λήψη δεδομένων, πρέπει ο read buffer του μικροελεγκτή να είναι έτοιμος. Όταν είναι έτοιμος να ληφθεί δεδομένα, τότε το bit 7 του καταχωρητή **USCRA** με όνομα **RXC** γίνεται '1', και τα δεδομένα αποθηκεύονται στον καταχωρητή **UDR** προς ανάγνωση:

```
char usart_getchar(void) {

        // Wait for incoming data

        while (!(UCSRA & (1<<RXC))) ;



        // Return the data

        return UDR;

}
```

Τα δεδομένα που λαμβάνονται όμως είναι απλοί χαρακτήρες, οι οποίοι πρέπει να αντιστοιχίζονται στις θέσεις του stepper motor. Για να μπορέσει η συνάρτηση κίνησης να εκτελεστεί ανάλογα με τη θέση η οποία λαμβάνεται ως χαρακτήρας μέσω του USART, οι χαρακτήρες που λαμβάνονται τοποθετούνται σε μία ακέραια μεταβλητή η οποία στη συνέχεια χρησιμοποιείται ως όρισμα στη συνάρτηση κίνησης.

4.1.4 Η συνάρτηση αποστολής δεδομένων

Για την αποστολή δεδομένων χρησιμοποιήθηκε η συνάρτηση αποστολής χαρακτήρα USART_Transmit. Η συνάρτηση αποστολής χαρακτήρα λειτουργεί με παρόμοιο τρόπο με τη συνάρτηση λήψης χαρακτήρα. Πριν σταλεί ο χαρακτήρας, γράφεται στον write buffer του

USART. Για να ξεκινήσει αυτή η διαδικασία πρέπει ο buffer να είναι άδειος. Εάν είναι άδειος, το **UDRE** bit (**U**SART **D**ata **R**egister **E**mpty) του **UCSRA**(το bit 5) γίνεται '1':

```
void USART_Transmit(char data)

{

        // Wait for empty transmit buffer

        while (!( UCSRA & (1<<UDRE)));



        // Put data into buffer, sends the data

        UDR = data;

}
```

Τα δεδομένα που αποστέλλονται από το μικροελεγκτή στο PC μέσω της σειριακής θύρας αφορούν τις πιθανές καταστάσεις του stepper motor όπως θα περιγραφεί παρακάτω.

4.1.5 Μετατροπή από χαρακτήρα σε ακέραιο:

Όπως προαναφέρθηκε, τα δεδομένα που αποστέλλονται από την εφαρμογή στο μικροελεγκτή αφορούν τη θέση στην οποία πρέπει να βρεθεί το stepper motor για τη λήψη μίας φωτογραφίας. Το πρόβλημα που παρουσιάστηκε στην παρούσα υλοποίηση είναι ότι τα δεδομένα είναι της μορφής χαρακτήρα ενώ για την κίνηση του stepper motor τα ορίσματα των συναρτήσεων κίνησης είναι ακέραια. Έτσι παρουσιάστηκε η ανάγκη μετατροπής των χαρακτήρων (ή συμβολοσειρών) που δέχεται ο μικροελεγκτής σε ακέραιους αριθμούς. Η εφαρμογή αποστέλλει χαρακτήρες ASCII 8 bit μέσω της σειριακής θύρας. Η δυαδική τιμή των χαρακτήρων ASCII που λαμβάνει ο μικροελεγκτής αποθηκεύεται σε μία μεταβλητή integer και στη συνέχεια χρησιμοποιείται ως όρισμα στις συναρτήσεις κίνησης.

4.2 Η ανάπτυξη της εφαρμογής

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το εργαλείο Microsoft Visual Studio 2015, και επιλέχθηκε ο τύπος της εφαρμογής να είναι WPF γιατί με το συγκεκριμένο τύπο προσφέρονται περισσότερες δυνατότητες σχεδιασμού γραφικών χειροκίνητα (κώδικας xaml). Οι λειτουργίες των εργαλείων της εφαρμογής προγραμματίστηκαν σε C#. Αρχικά, για την ενεργοποίηση της web camera έπρεπε να ενσωματωθεί η βιβλιοθήκη WebcamControl.dll στον κώδικα του κυρίου παραθύρου της εφαρμογής, καθώς και η κλάση WebcamControl η οποία σχεδιάζει το χώρο για το camera preview εντός του κυρίου παραθύρου[6]. Στη βιβλιοθήκη WebcamControl.dll περιέχονται οι συναρτήσεις που εντοπίζουν τις συνδεδεμένες συσκευές camera στον υπολογιστή και οι συναρτήσεις που εκτελούν λήψη φωτογραφίας και αποθήκευσή της. Για την υλοποίηση της εφαρμογής ήταν επίσης αναγκαίο να ενσωματωθεί στον κώδικα η βιβλιοθήκη του Microsoft Expressions Encoder.

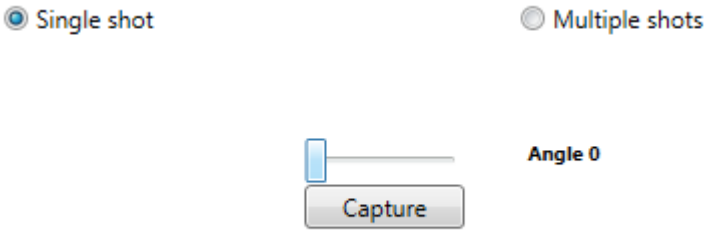
Επίσης για τη δυνατότητα σειριακής επικοινωνίας μεταξύ της εφαρμογής και της θύρας RS232 είναι απαραίτητη η προσθήκη της βιβλιοθήκης System.IO και System.IO.Ports.

4.2.1 Η γραφική σχεδίαση της εφαρμογής

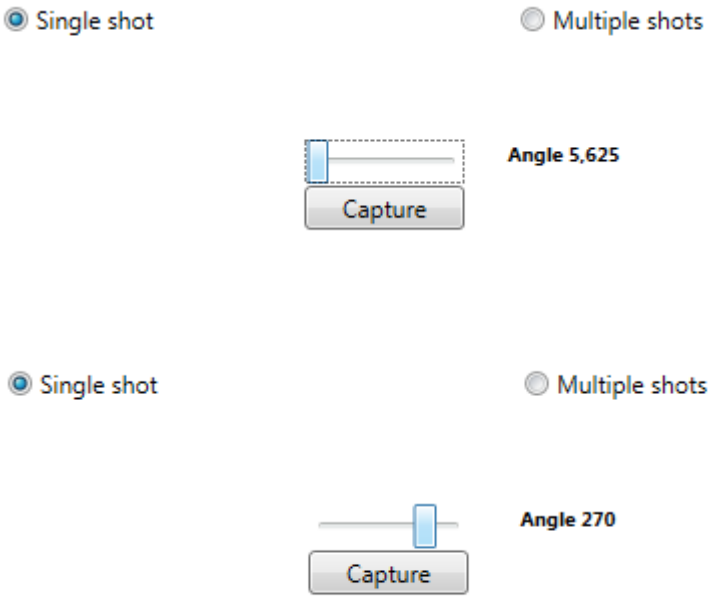
Η σχεδίαση του παραθύρου της εφαρμογής έγινε με τα εργαλεία σχεδίασης του Microsoft Visual Studio 2015 στη γλώσσα σχεδιασμού xaml. Το μέγεθος του παραθύρου ορίσθηκε σε 800x1000 pixels. Αναλυτικά, χρησιμοποιήθηκαν τα εξής components:

Δύο Radio Buttons καθορίζουν αν η λειτουργία είναι απλή ή πολλαπλή, με ονομασίες Single shot και Multi shot, το κάθε ένα ομαδοποιημένο με Sliders και ένα Capture button. Με την έναρξη της εφαρμογής, τα δύο Radio Buttons έχουν αρχικοποιηθεί σε μη επιλεγμένα, με αποτέλεσμα τα περιεχόμενά τους να παραμένουν κρυφά μέχρι ο χρήστης να επιλέξει ένα από τα δύο πιθανά mode λειτουργίας, όπως φαίνεται στο παρακάτω στιγμιότυπο της εφαρμογής:

Στην περίπτωση που ο χρήστης επιλέξει λειτουργία Single shot, χρησιμοποιείται το ένα Slider, με βήμα 5.625, για την επιλογή της επιθυμητής γωνίας, ενώ τα περιεχόμενα του Radio Button Multiple shots μένουν κρυφά, όπως φαίνεται παρακάτω:

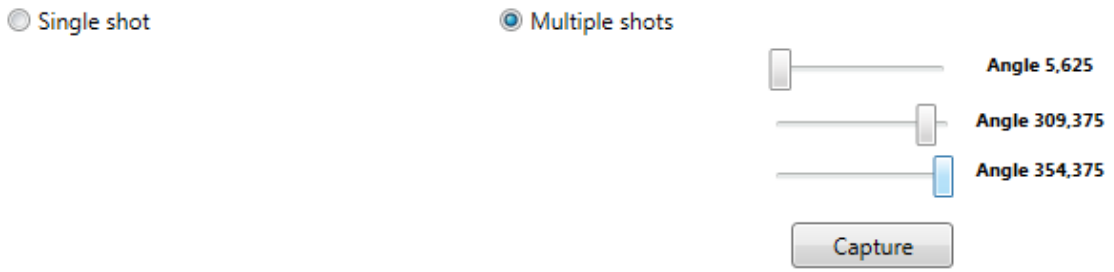


Ο χρήστης κυλώντας το slider επιλέγει μία θέση λήψης πολλαπλάσια των 5.625 μοιρών:



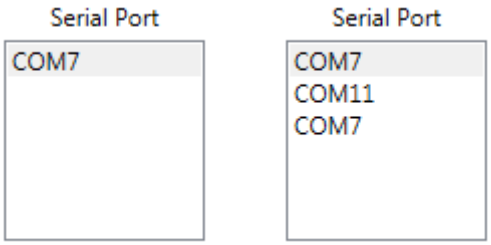
Η επιλογή που θα δώσει ο χρήστης καθορίζει την επόμενη θέση στην οποία θα βρεθεί το stepper motor για τη λήψη της επόμενης φωτογραφίας.

Στη λειτουργία Multiple shots, χρησιμοποιούνται ακόμη 3 sliders για την επιλογή τριών πιθανών γωνιών για περιστροφή. Εάν ο χρήστης επιλέξει τη λειτουργία Multiple shots τότε τα περιεχόμενα του Radio Button Single shot μένουν κρυφά, όπως φαίνεται στο ακόλουθο στιγμιότυπο:



Μετά την επιλογή της γωνίας ή των γωνιών, ο χρήστης πρέπει να πατήσει το κουμπί Capture και στους δύο τρόπους λειτουργίας, ώστε η εφαρμογή να στείλει στο μικροελεγκτή τη θέση ή τις θέσεις στις οποίες πρέπει να κινηθεί το stepper motor για τη λήψη μίας φωτογραφίας.

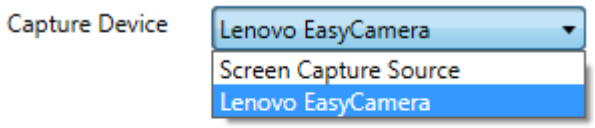
Επιπλέον, χρησιμοποιήθηκε ένα List Box το οποίο περιέχει όλες τις πιθανές σειριακές θύρες για επικοινωνία (για την περίπτωση που χρησιμοποιηθούν περισσότερες από μία σειριακές συσκευές στον ίδιο υπολογιστή, ή περισσότερα από ένα συστήματα Ιδεατού Μουσείου):



Στο σημείο αυτό, να σημειωθεί ότι πριν την επιλογή οποιασδήποτε θέσης για λήψη, ο χρήστης πρέπει πρώτα να επιλέξει τη σωστή σειριακή θύρα επικοινωνίας. Στην περίπτωση που η σωστή θύρα επικοινωνίας δεν έχει επιλεγεί, και ο χρήστης επιλέξει να στείλει δεδομένα (που δε θα φτάσουν ποτέ), η εφαρμογή θα περιμένει μάταια να πάρει απάντηση από το μικροελεγκτή για 60 δευτερόλεπτα και μετά θα κλείσει. Η σωστή σειριακή θύρα μπορεί να

εντοπιστεί από το χρήστη από το μενού του Πίνακα Ελέγχου του υπολογιστή, στη διαδρομή Πίνακας Ελέγχου-Διαχείριση Υπολογιστή-Διαχείριση Συσκευών-Θύρες(COM& LTP).

Επίσης έχει προστεθεί ένα Combo Box, στο οποίο εμφανίζονται σε drop down menu οι συνδεδεμένες στον υπολογιστή συσκευές video:



Ο χώρος στον οποίο εμφανίζεται το Preview της web camera ορίζεται επίσης στο κάτω μέρος του παραθύρου.

4.2.2 Οι λειτουργίες της εφαρμογής

Για να γίνει preview της web camera καλείται η μέθοδος startPreview της βιβλιοθήκης WebcamControl.

Για τη λήψη στιγμιότυπου σε επιλεγμένη γωνία, καλείται η μέθοδος TakeSnapshot της βιβλιοθήκης WebcamControl.

Για να γίνεται γνωστή σε κάθε δεδομένη στιγμή η κατάσταση κίνησης του stepper motor κατασκευάστηκε μία μηχανή καταστάσεων με τις ακόλουθες τρεις καταστάσεις : IDLE, ROTATING, READY_TO_CAPTURE. Οι καταστάσεις αυτές ενεργοποιούνται από τα δεδομένα κατάστασης του stepper που στέλνει κάθε φορά ο μικροελεγκτής μέσω της σειριακής θύρας. Στην περίπτωση που το stepper είναι ακίνητο και περιμένει νέα γωνία, τότε ο μικροελεγκτής στέλνει το χαρακτήρα ‘0’ ο οποίος μεταφέρει το σύστημα στην κατάσταση IDLE, ώστε να έχει γνώση η εφαρμογή ότι ο μικροελεγκτής είναι έτοιμος να δεχθεί νέα

δεδομένα κίνησης. Όσο το stepper περιστρέφεται για να φτάσει στην επιθυμητή γωνία, ο μικροελεγκτής στέλνει το χαρακτήρα ‘1’ ώστε η κατάσταση να γίνει ROTATING, στην οποία δεν επιτρέπεται να δοθούν επιπλέον εντολές. Μόλις το stepper φτάσει στην επιθυμητή θέση, ο μικροελεγκτής στέλνει το χαρακτήρα ‘2’ ο οποίος θα σηματοδοτήσει την εφαρμογή να μεταπηδήσει στην κατάσταση READY_TO_CAPTURE, όπου καλείται η μέθοδος TakeSnapshot για τη λήψη της φωτογραφίας.

Οι φωτογραφίες αποθηκεύονται στο φάκελο WebCamSnapShots ο οποίος αν δεν υπάρχει ήδη στο φάκελο της εφαρμογής δημιουργείται από τη συνάρτηση:
Directory.CreateDirectory(imagePath)

Η διαδικασία της αποθήκευσης των φωτογραφιών καθορίζεται επίσης από τη βιβλιοθήκη WebcamControl.

Για τη σωστή επικοινωνία μεταξύ του μικροελεγκτή και της εφαρμογής, η σειριακή επικοινωνία αρχικοποιείται στην εφαρμογή με τις ίδιες συνθήκες που αρχικοποιήθηκαν και στο μικροελεγκτή (data bits, stop bits). Επίσης η κωδικοποίηση της επικοινωνίας ορίζεται σε ASCII, για το λόγο που αναφέρθηκε στην παράγραφο **4.1.5.**

Οι γωνίες που επιλέγει ο χρήστης, είναι πολλαπλάσιες του 5,625 όπως προαναφέρθηκε (όσο και το βήμα του stepper). Για την υλοποίηση της επικοινωνίας μεταξύ PC και μικροελεγκτή, η κάθε επιλεγμένη γωνία διαιρείται με το 5,625 πριν την αποστολή των δεδομένων, επομένως ο αριθμός που θα αποσταλεί (σε πακέτο 8 bit) θα είναι από το 0 ως το 63 αφού τα steps είναι 64 και οι γωνίες είναι από 0 ως 354.375 μοίρες. Για την αποστολή των δεδομένων στο μικροελεγκτή χρησιμοποιείται η μέθοδος SerialPort.Write. Η εν λόγω μέθοδος παίρνει ως ορίσματα τον byte[] buffer, το offset και το count. Το byte είναι μεταβλητή της βιβλιοθήκης System και περιέχει τα δεδομένα προς εγγραφή στη σειριακή θύρα. Το offset δηλώνει το σημείο στο οποίο ξεκινάνε να αποθηκεύονται bytes στον buffer της σειριακή θύρας, ενώ το count είναι ο αριθμός των bytes που θα αποθηκευθούν. Τα δεδομένα προς αποστολή, αρχικά αποθηκεύονται σε μία μεταβλητή χαρακτήρα. Για να σταλεί το σωστό format δεδομένων, το count , το οποίο ορίζει το πόσα byte θα σταλούν, ορίζεται ‘1’.

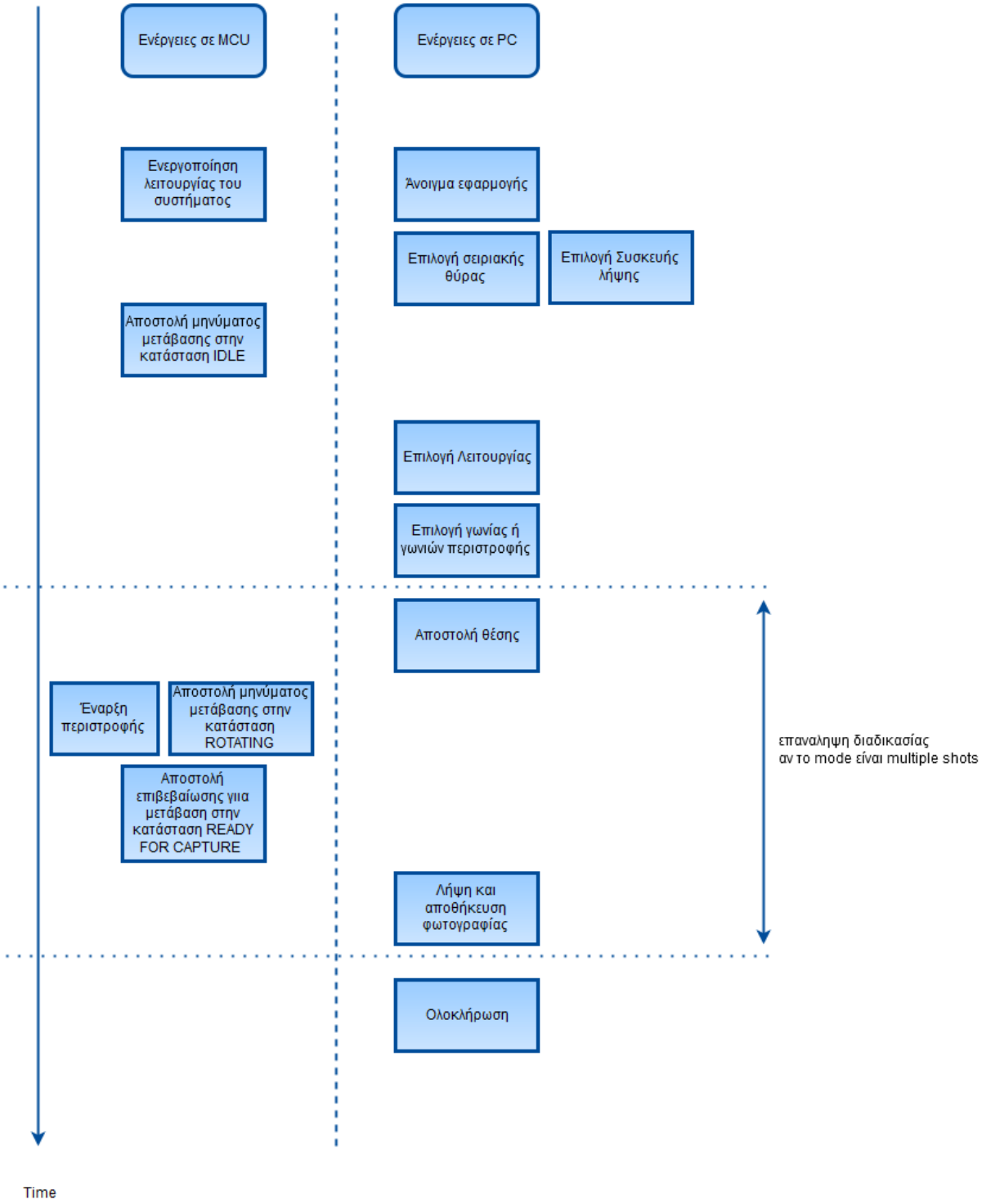
Η μέθοδος BitConverter.GetBytes παίρνει ως είσοδο τη μεταβλητή χαρακτήρα που περιέχει την τιμή της θέσης του stepper motor, και τη μετατρέπει σε Byte, το οποίο στη συνέχεια αποστέλλεται μέσω της σειριακής θύρας στο μικροελεγκτή.

Σε περίπτωση πολλαπλής λειτουργίας, (multi shot), οι τρεις επιλεγμένες γωνίες από τα τρία slider αφού διαιρεθούν με το 5.625, αποθηκεύονται σε έναν πίνακα και αποστέλλονται η μία μετά την άλλη στο μικροελεγκτή, με τη συνολική διαδικασία που έχει περιγραφεί, να εκτελείται τρεις φορές (όλες οι αλλαγές στα states και οι διαδικασίες συνομιλίας μεταξύ του μικροελεγκτή και του PC εκτελούνται τρεις φορές).

4.2.3 Επικοινωνία μεταξύ PC και μικροελεγκτή

Με την ενεργοποίηση της σειριακής θύρας, ο μικροελεγκτής στέλνει στον υπολογιστή το χαρακτήρα ‘0’, και μ αυτόν τον τρόπο το σύστημα μπαίνει στην κατάσταση IDLE, όπου ο χρήστης μπορεί να δώσει την επιλογή των γωνιών. Μόλις ο χρήστης επιλέξει γωνίες και ενεργοποιήσει το κουμπί Capture, τότε αποστέλλονται στο μικροελεγκτή τα δεδομένα που περιέχουν τη θέση στην οποία πρέπει να βρεθεί. Να σημειωθεί σ’ αυτό το σημείο, ότι αυτό που μεταφέρεται είναι η δυαδική τιμή των χαρακτήρων σε κωδικοποίηση ASCII. Δηλαδή, αν επιλεγεί για παράδειγμα η θέση 0, ο χαρακτήρας που θα ληφθεί από το μικροελεγκτή δεν θα είναι ο χαρακτήρας ‘0’, ο οποίος σε ASCII έχει δυαδική τιμή “00110000” ή 48 σε δεκαδικό, αλλά ο χαρακτήρας με δυαδική τιμή “00000000” ο οποίος αντιστοιχεί στο χαρακτήρα NULL. Αντίστοιχα η θέση 32 αντιστοιχεί στο χαρακτήρα ASCII ‘!’ κλπ. Ένας πίνακας του κώδικα ASCII υπάρχει στο παράρτημα. Μόλις ο μικροελεγκτής δεχτεί τη γωνία, δίνει εντολή στο stepper να ξεκινήσει να κινείται. Όταν το stepper ξεκινήσει να κινείται στέλνει στην εφαρμογή το χαρακτήρα ‘1’, ώστε να μπει στην κατάσταση ROTATING. Μόλις σταματήσει η κίνηση, ο μικροελεγκτής στέλνει το χαρακτήρα ‘2’, και τότε εμφανίζεται στην οθόνη το μήνυμα “capturing and saving”.Η εφαρμογή μπαίνει στην κατάσταση READY_TO_CAPTURE, όπου λαμβάνεται και αποθηκεύεται η φωτογραφία.

Παρακάτω παρουσιάζεται ένα διάγραμμα των ενεργειών που εκτελούνται από τις δύο πλευρές (μικροελεγκτής και εφαρμογή υπολογιστή) στην πορεία του χρόνου:



Εικόνα 7: Διάγραμμα ενεργειών-χρόνου

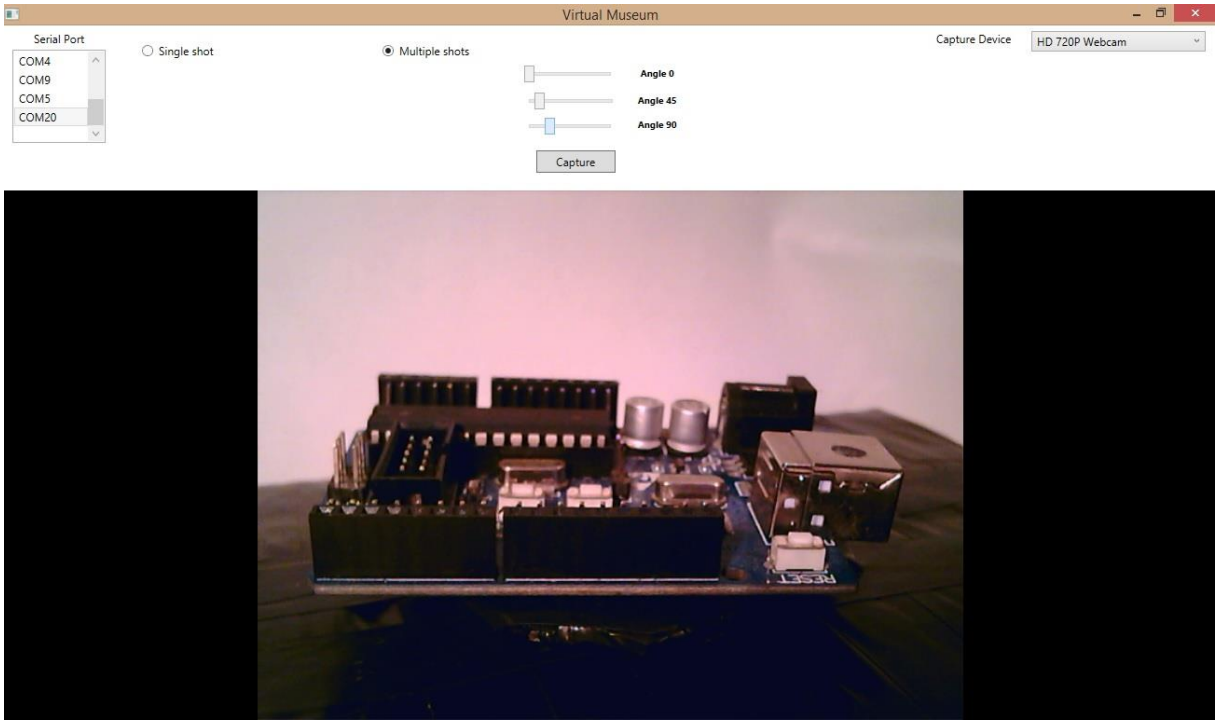
4.2.4 Απόρριψη Προηγούμενων Υλοποιήσεων

Πριν την παραπάνω υλοποίηση του συστήματος, με τα προαναφερθέντα components, έγινε απόπειρα υλοποίησής του με διαφορετικούς τρόπους, οι οποίοι και απορρίφθηκαν για διαφορετικούς λόγους ο καθένας.

Η πρώτη προσέγγιση που ακολουθήθηκε και μελετήθηκε ήταν η χρήση ενός stepper motor από σκληρό δίσκο υπολογιστή. Η προσέγγιση αυτή απορρίφθηκε γρήγορα καθότι τα stepper motors που χρησιμοποιούνται σε σκληρούς δίσκους ανήκουν στην κατηγορία Variable Reluctance Motors που περιγράφηκε στο κεφάλαιο 2 και δεν προσφέρονται για κίνηση σε χαμηλές ταχύτητες. Επίσης η ακρίβειά τους βασίζεται πάρα πολύ στον controller τους, ο οποίος ταυτόχρονα είναι και ο controller του σκληρού δίσκου. Έτσι καθίσταται πολύ δύσκολη η αναπαραγωγή ενός τέτοιου κυκλώματος. Έτσι προτιμήθηκε το stepper 28BYJ-48.

Εκτός από το stepper motor, και στη σχεδίαση της εφαρμογής ακολουθήθηκε διαφορετική μέθοδος αρχικά. Στην πρώτη απόπειρα χρησιμοποιήθηκαν τα εργαλεία γραφικών της MATLAB, σε μία προσέγγιση παρόμοια με το παράδειγμα του [1]. Το πρόβλημα το οποίο παρουσιάστηκε και έγινε νωρίς αντιληπτό, είναι ότι παρόλο που η MATLAB είναι σε θέση να κάνει export μία εφαρμογή σε αρχείο εκτελέσιμου τύπου (.exe) , για να γίνει αυτό πρέπει σε κάθε υπολογιστή που εγκαθίσταται η εφαρμογή να είναι προεγκατεστημένο ένα πρόγραμμα της Mathworks, το MATLAB runtime compiler. Έτσι και αυτή η επιλογή απορρίφθηκε και η σχεδίαση του συστήματος έγινε όπως περιγράφηκε στο παρόν κεφάλαιο.

4.2.5 Στιγμιότυπο λειτουργίας



Εικόνα 8: Το πλήρες σύστημα σε λειτουργία, στιγμιότυπο από Windows 8

5. ΕΠΙΒΕΒΑΙΩΣΗ ΛΕΙΤΟΥΡΓΙΑΣ

Για την επιβεβαίωση λειτουργίας του συστήματος, τα τμήματα που το απαρτίζουν μελετήθηκαν ξεχωριστά.

5.1 Επιβεβαίωση λειτουργίας του Stepper

Στην αρχή της απόπειρας υλοποίησης του συστήματος, εξετάστηκε αν οι συναρτήσεις κίνησης που αναπτύχθηκαν λειτουργούν. Για τον έλεγχο των συναρτήσεων, το Stepper Motor συνδεδεμένο μόνο με το μικροελεγκτή και τον Stepper Motor Driver κλήθηκε να εκτελέσει clockwise και counter clockwise περιστροφή αλλάζοντας κάθε φορά την επιθυμητή φορά και γωνία με εκ νέου προγραμματισμό του μικροελεγκτή. Κατά τη διάρκεια των δοκιμών αυτών πιστοποιήθηκε και το γεγονός ότι χρειάζονται 512 συνεχόμενες ενεργοποιήσεις των $4^{ων}$ πόλων για μία πλήρη περιστροφή και κατόπιν έγιναν οι απαραίτητοι υπολογισμοί για το επιθυμητό βήμα των 5.625 μοιρών.

5.2 Επιβεβαίωση σειριακής επικοινωνίας

Για την επιβεβαίωση της σειριακής επικοινωνίας μέσω USART μεταξύ μικροελεγκτή και υπολογιστή, χρησιμοποιήθηκε το πρόγραμμα Bray Terminal, το οποίο προσφέρει τη λειτουργία μίας σειριακής οθόνης. Υλοποιήθηκε μία εφαρμογή επικοινωνίας μεταξύ μικροελεγκτή και υπολογιστή, κατά την οποία ο χρήστης στέλνει έναν χαρακτήρα ή μία συμβολοσειρά στο μικροελεγκτή μέσω του προγράμματος Bray Terminal, και ο μικροελεγκτής απαντάει στέλνοντας πίσω ακριβώς ότι πληροφορία δέχθηκε. Οι πληροφορίες που στέλνει ο μικροελεγκτής εμφανίζονται στη σειριακή οθόνη του προγράμματος Bray Terminal.

5.3 Επιβεβαίωση λειτουργίας συνολικού Hardware

Για να γίνει η επιβεβαίωση της λειτουργίας του hardware πριν τη σύνδεσή του με την εφαρμογή, χρησιμοποιήθηκε το Bray Terminal για την αποστολή των απαραίτητων χαρακτήρων και τον έλεγχο της ορθής λειτουργίας της κίνησης. Για να μπορέσει να γίνει αυτό, επειδή αρκετοί χαρακτήρες ASCII δεν είναι δυνατόν να πληκτρολογηθούν στο Bray Terminal, υλοποιήθηκε η εξής μέθοδος:

Οι χαρακτήρες που μπορούν να πληκτρολογηθούν στο Bray Terminal ξεκινάνε από τον χαρακτήρα space (032), ενώ ο χαρακτήρας '!' είναι το (033). Επιλέχθηκε να θεωρηθεί ο χαρακτήρας '!' ως η θέση 0 του stepper. Έτσι ο μικροελεγκτής προγραμματίστηκε με σκοπό την αποσφαλμάτωση, με τέτοιο τρόπο ώστε από κάθε χαρακτήρα που λάμβανε μέσω σειριακής θύρας να αφαιρεί την τιμή του χαρακτήρα '!'. Έτσι, αν για παράδειγμα έπρεπε να μεταφερθεί το stepper στη θέση 5, θα στελνόταν ο χαρακτήρας '&'. Στη συνέχεια, η τιμή του θα διαμορφωνόταν ως εξής: '&'-'!'=$038_{ASCIIdecimal}$-$033_{ASCIIdecimal}$= 5. Μετά τη σύνδεση της εφαρμογής όπου δε χρειάζεται να πληκτρολογηθούν χαρακτήρες χειροκίνητα, αφαιρέθηκε αυτή η διαδικασία.

5.4 Επιβεβαίωση λειτουργίας της web camera

Για την επιβεβαίωση λειτουργίας της βιβλιοθήκης WebcamControl.dll, υλοποιήθηκε εν μέρει μία από τις προτεινόμενες εφαρμογές προς πειραματισμό της αναφοράς [6]. Εφόσον η εφαρμογή λειτούργησε και πιστοποιήθηκε η δυνατότητα ενεργοποίησης των συσκευών λήψης εικόνας ενός υπολογιστή μέσω C# και της παραπάνω βιβλιοθήκης, τότε η βιβλιοθήκη

WebcamControl.dll και οι μέθοδοί της ενσωματώθηκαν στον κώδικα της υπόλοιπης εφαρμογής διαχείρισης του Virtual Museum.

6.
ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Με αφορμή την ιδέα για την κατασκευή του Ιδεατού Μουσείου, πολλές επιπλέον ιδέες εργασίας βασισμένες πάνω σ' αυτό μπορούν να προκύψουν.

Αρχικά, ως μελλοντική εργασία και επέκταση της παρούσας, θα μπορούσε να είναι η κατασκευή ενός ολοκληρωμένου συστήματος Ιδεατού Μουσείου, με ενσωματωμένη στη στρεφόμενη πλατφόρμα camera και υλοποιημένο πρωτόκολλο επικοινωνίας USB ώστε η συσκευή να χρησιμοποιεί ένα μόνο καλώδιο USB ταυτόχρονα για την camera και την επικοινωνία μεταξύ συστήματος-υπολογιστή.

Επίσης, στο παρόν σύστημα μπορεί να προστεθεί άλλη μία camera πάνω από την πλατφόρμα για φωτογραφίες από άλλες όψεις, με τελικό στόχο τη χρήση αυτών των φωτογραφιών για ψηφιακή αναπαράσταση των εκθεμάτων σε τρισδιάστατο μοντέλο, μέσω λογισμικού που ενδεχομένως να αναπτυχθεί μελλοντικά.

Άλλη πιθανή επέκταση, θα μπορούσε να είναι η προσθήκη μίας βάσης δεδομένων στην εφαρμογή, για αποθήκευση των φωτογραφιών και των στοιχείων των εκθεμάτων. Μαζί με την αναβάθμιση της εφαρμογής θα μπορέσει να δημιουργηθεί ένα πλήρες σύστημα πλοήγησης στο ιδεατό μουσείο του κάθε χρήστη.

Επίσης το σύστημα του Ιδεατού Μουσείου (σε μεγαλύτερο μέγεθος) θα μπορούσε να χρησιμοποιηθεί για την καταγραφή φθοράς και ζημιών σε πραγματικά μουσειακά εκθέματα.

Εκτός από την προσέγγιση που έχει παρουσιαστεί ως τώρα, ένα σύστημα με χρήση camera και stepper motor,(ή οποιοδήποτε είδος dc motor) θα μπορούσε να χρησιμοποιηθεί ως σύστημα μηχανικής όρασης σε drones, για χαρτογράφηση περιοχών κλπ.

7.
ΒΙΒΛΙΟΓΡΑΦΙΑ

[1]. http://www.nbcafe.in/interfacing-camera-with-pic-microcontroller-via-matlab-gui/

[2] .http://www.atmel.com/Images/compiler.pdf –summary της λειτουργίας του ATmega8515

[3] .http://www.atmel.com/Images/2512S.pdf – το πλήρες datasheet του ATmega8515

[4] .http://electronics.stackexchange.com/questions/122128/understanding-this-statement-in-atmega328ps-datasheet

[5]. http://avrprogrammers.com/atmega/atmega8515

[6]. http://www.codeproject.com/Articles/285964/WPF-Webcam-Control - συντάκτης Meshack Musundi

[7]. Pro C# 2010 and the .NET 4 Platform-Andrew Troelsen

[8]. C# 4.0 THE COMPLETE REFERENCE - Herbert Schildt

[9]. http://www.avr-tutorials.com/digital/about-avr-8-bit-microcontrollers-digital-io-ports

[10]. http://maxembedded.com/2011/06/port-operations-in-avr/

[11]. http://extremeelectronics.co.in/avr-tutorials/using-the-usart-of-avr-microcontrollers/

[12]. http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/

[13]. http://slideplayer.com/slide/1421990/ - διάλεξη και παρουσίαση πάνω στα DC motors του Λέκτορα Satnam Singh του Govt Polytechnic College - Mohali (Khunimajra)

[14]. http://reprap.org/wiki/Stepper_motor

[15]. http://homepage.cs.uiowa.edu/~jones/step/ tutorial on stepper motors by Douglas W. Jones ,THE UNIVERSITY OF IOWA Department of Computer Science

[16]. http://www.ams2000.com/stepping101/stepping101_Motors_1.html

[17]. http://www.orientalmotor.com/technology/articles/motor-sizing-calculations.html

[18]. http://www.nmbtc.com/step-motors/engineering/torque-and-speed-relationship/

[19]. http://www.eecs.berkeley.edu/~hodges/DarlingtonCircuit.pdf

[20]. https://blog.adafruit.com/2013/08/12/whats-inside-the-uln2003-darlington-driver-chip/

[21]. http://www.rutherfordjournal.org/article030103.html

[22]. The virtual museum: Interactive 3D navigation of a multimedia database -The Journal of Visualization and Computer Animation, Volume 3, Issue 3, pages 183–197, July/September 1992

[23].Production Thesis, Master of Professional Studies,-The Virtual Museum, Sally Ann Applin

[24]. http://web.archive.org/web/19970606010526/http://www.ucmp.berkeley.edu/

[25].Virtual museums, a survey and some issues for consideration, Journal of Cultural heritage, volume 10, issue 4, October-December 2009

[26]. Making Sense of Space, σελ. 31, 32, Iryn Duska, Mark Childe

[27].Web Based Global Virtual Museum Of Congenital Cardiac Pathology- Progress in Pediatric Cardiology, volume 33 issue 1

[28].www.neothemi.net

[29]. https://www.myminifactory.com/category/scan-the-world

[30].www.ipet.gr/momi

[31].www.iconasys.com

[32]. http://www.photogear360.com/content/8-diy-360-product-photography-turntable

[33]. http://www.instructables.com/id/360-Rotating-Platform-with-fixed-central-platform/

[34]. http://hackaday.com/2015/05/24/spin-diy-photography-turntable-system/

[35]. Atmel STK 500 user guide

ΠΑΡΑΡΤΗΜΑ Α

1 Stepper Motors

1.1 Γενικά για τα DC Motors

Εξ ορισμού, ένα DC motor είναι ένας ηλεκτρικός κινητήρας που μετατρέπει DC ηλεκτρική ενέργεια σε κίνηση. Στην πλειοψηφία τους, τα DC motors βασίζονται σε δυνάμεις παραγόμενες από μαγνητικά πεδία για την κίνησή τους.

Λόγω του γεγονότος ότι τα DC motors μπορούν να τροφοδοτηθούν απ' ευθείας από οποιαδήποτε πηγή συνεχούς τάσης έγιναν το πιο διαδεδομένο είδος ηλεκτρικού κινητήρα μέχρι σήμερα, με πάρα πολλές εφαρμογές στην καθημερινή ζωή. Η ταχύτητα ενός DC motor μπορεί να ελεγχθεί είτε με αυξομείωση της τάσης στα άκρα του είτε με μεταβολές της έντασης του ηλεκτρικού ρεύματος που περνάει από τα πηνία του. Σε πολλά σημεία της παρούσας εργασίας θα αναφερθεί η λέξη winding , η οποία αναφέρεται σε ένα πηνίο τυλιγμένο γύρω από ένα ηλεκτρομαγνητικό υλικό.

Αναλυτικότερα, τα τμήματα ενός dc motor όπως φαίνονται στην παρακάτω εικόνα είναι τα εξής:

α. Rotor –Το στρεφόμενο μέρος ενός ηλεκτρικού DC motor, βρίσκεται σε αλληλεπίδραση με το στάτορα, είναι το τμήμα του motor που παράγει την περιστροφική ροπή.

β. Stator –Το στάσιμο μέρος, το οποίο σε αλληλεπίδραση με το ρότορα δημιουργούν μαγνητικό πεδίο καθώς συνήθως αποτελείται από έναν στατικό μαγνήτη τυλιγμένο μέσα σε ένα πηνίο.

γ. Commutator (μεταγωγέας) –περιοδικά αντιστρέφει τη διεύθυνση του ρεύματος μεταξύ του ρότορα και του εξωτερικού κυκλώματος

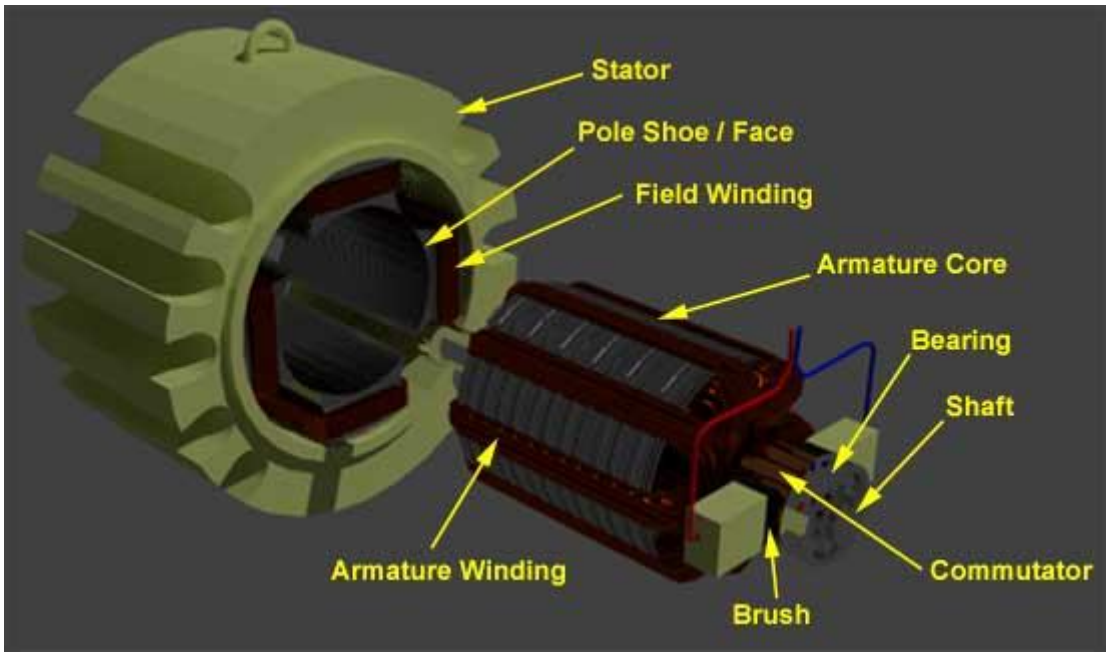
δ. Brush –στατική ηλεκτρική επαφή, από ελαφρύ αγώγιμο υλικό όπως ο άνθρακας, η οποία βρίσκεται σε επαφή με το μεταγωγέα καθώς αυτός περιστρέφεται. Ουσιαστικά άγει ρεύμα μεταξύ του στρεφόμενου και του στατικού μέρους του DC motor.

ε. Yoke –αποτελεί τμήμα του στάσιμου μέρους (stator) ενός DC motor, συνήθως κατασκευασμένο από σίδηρο ή χάλυβα. Ουσιαστικά είναι το κέλυφος ενός DC motor και η λειτουργία του είναι να προσφέρει προστασία στα εσωτερικά τμήματα του motor. Επίσης συγκρατεί σταθερούς τους μαγνητικούς πόλους και τα Field windings εντός του motor.

στ. Armature winding –είναι το προσαρτημένο στο ρότορα πηνίο, μεταβάλλει το μαγνητικό πεδίο στην τροχιά της κυκλικής του κίνησης, με αποτέλεσμα να μειώνονται οι μαγνητικές απώλειες και οι απώλειες ηλεκτρικού ρεύματος.

ζ. Field windings –αποτελούν πηνία τυλιγμένα γύρω από το πέταλο ενός πόλου. Η λειτουργία τους έχει ως αποτέλεσμα την αντίθετη πόλωση δύο γειτονικών πόλων του DC motor.

η. Poles –Οι μαγνητικοί πόλοι ενός DC motor αποτελούνται από δύο βασικά τμήματα, τον πυρήνα (core) και το πέταλο (shoe), συγκρατημένα σταθερά μεταξύ τους και προσαρτημένα στο πλαίσιο (yoke). Ο μόνος ρόλος του πυρήνα είναι να συγκρατεί το πέταλο του πόλου σταθερό και ενωμένο με το πλαίσιο. Το πέταλο με τη σειρά του διαχέει τη ροή αέρα μεταξύ του στάτορα και του ρότορα συντελώντας έτσι στη μείωση των απωλειών λόγω μαγνητικής αντίστασης.[12],[13]



Εικόνα 9: Τα τμήματα ενός DC motor[12]

1.2 Stepper Motors

Το stepper motor είναι ένα είδος DC motor χωρίς commutator (μεταγωγέα). Η μηχανική διαφορά από ένα απλό DC motor είναι ότι όλα τα τυλίγματα εντός είναι τμήματα του στάτορα και ο ρότορας είναι είτε ένας μόνιμος μαγνήτης είτε στην περίπτωση των variable reluctance motors μία οδοντωτή δομή από κάποιο μαγνητικό υλικό. Τα stepper motors χρησιμοποιούν πολλαπλούς οδοντωτούς ηλεκτρομαγνήτες τοποθετημένους γύρω από έναν μεταλλικό οδοντωτό άξονα. Οι ηλεκτρομαγνήτες ενεργοποιούνται από ένα εξωτερικό κύκλωμα οδήγησης ή έναν μικροελεγκτή. Για να περιστραφεί ο άξονας, οι ηλεκτρομαγνήτες ενεργοποιούνται σειριακά ο ένας μετά τον άλλο. Με την ενεργοποίηση ενός ηλεκτρομαγνήτη, παρουσιάζεται μαγνητική έλξη μεταξύ των δοντιών του άξονα και του ηλεκτρομαγνήτη. Μόλις τα δόντια του άξονα ευθυγραμμιστούν με τον ενεργό ηλεκτρομαγνήτη, η τοποθέτησή τους είναι τέτοια ώστε να μπορούν να ελκυσθούν από τον επόμενο σε σειρά. Έτσι, μόλις ο 1ος ηλεκτρομαγνήτης απενεργοποιηθεί και ενεργοποιηθεί ο επόμενος σε σειρά, ο άξονας θα περιστραφεί προς τον επόμενο ενεργό ηλεκτρομαγνήτη. Με επανάληψη ενεργοποιήσεων των γειτονικών ηλεκτρομαγνητών, επιτυγχάνεται η κυκλική κίνηση του stepper motor. Με αντίστροφη ενεργοποίηση των ηλεκτρομαγνητών, επιτυγχάνεται και κίνηση αντίθετης φοράς. Κάθε ενεργοποίηση γειτονικών ηλεκτρομαγνητών που επιφέρει ως αποτέλεσμα μία μικρή μετατόπιση ονομάζεται βήμα ή step, εξ ου και η ονομασία των stepper motors. Για μία πλήρη περιστροφή απαιτείται ακέραιος αριθμός βημάτων, και έτσι επιτυγχάνεται η περιστροφή ενός stepper motor σε συγκεκριμένες ακριβείς θέσεις. [14],[15]

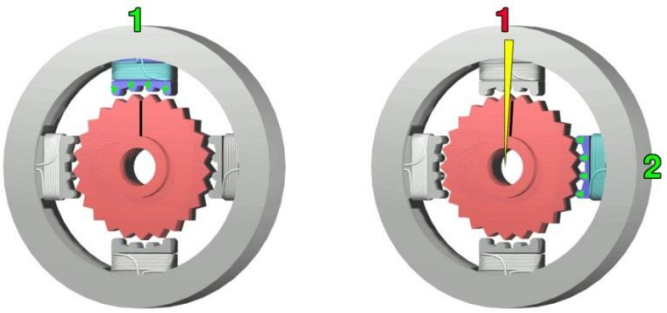
Τα stepper motors γενικά χωρίζονται σε δύο κατηγορίες, τα permanent motors και τα variable reluctance motors. Η διαφορά ανάμεσα στα δύο είδη μπορεί να γίνει αντιληπτή από μια απλή περιστροφή τους με το χέρι χωρίς την εφαρμογή ηλεκτρικής ισχύος. Τα permanent magnet motors με τη χειροκίνητη περιστροφή τους παράγουν ήχο και δεν κινούνται ελεύθερα, ενώ τα variable reluctance motors περιστρέφονται χωρίς πρόβλημα. Στην διπλωματική αυτή χρησιμοποιήθηκε το πρώτο είδος, για λόγους ροπής και ακρίβειας. Το δεύτερο είδος έχει μικρότερη ροπή αλλά είναι δυνατόν να περιστραφεί σε μεγαλύτερες ταχύτητες από το πρώτο. Έτσι μία από τις πιο βασικές χρήσεις των variable reluctance motors είναι σε σκληρούς δίσκους HDD, όπου τόσο η ταχύτητα όσο και η ακρίβεια στην κάθε πιθανή θέση είναι απαραίτητη.[14],[15]

Τη λειτουργία των stepper motors διαχειρίζεται εξ ολοκλήρου από ο stepper motor controller ενώ το βασικό χαρακτηριστικό και πλεονέκτημα της λειτουργίας τους, καθώς και ο λόγος για τον οποίο χρησιμοποιήθηκε αυτό το είδος μοτέρ στην πορεία αυτής της διπλωματικής είναι ότι το μοτέρ μπορεί να μείνει σταθερό σε οποιαδήποτε θέση από συγκεκριμένες γωνίες στην πορεία της κυκλικής του περιστροφής. Επίσης ένα άλλο πλεονέκτημα είναι ότι τα μοτέρ αυτά έχουν τη δυνατότητα να περιστραφούν και με τις δύο πιθανές φορές.[14],[15]

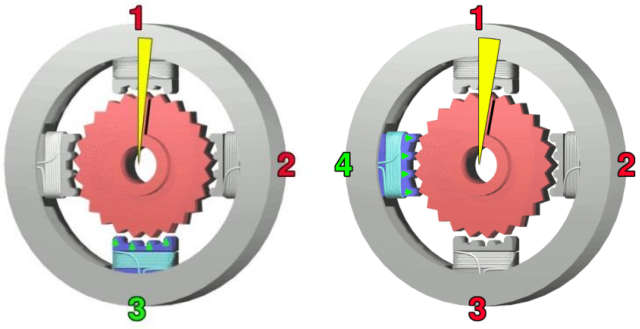
Τα stepper motors μπορούν να περιστραφούν με μεγάλη ακρίβεια σε μεγάλο εύρος βημάτων(steps) , από 90 μοίρες ανά step, ως και 0.72 μοίρες ανά step. Στα περισσότερα stepper motors και ανάλογα τον controller , υπάρχει η δυνατότητα του half stepping, να σταματήσει δηλαδή η κίνηση μεταξύ δύο ηλεκτρομαγνητών. Αυτό επιτυγχάνεται με ταυτόχρονη ενεργοποίηση δύο διπλανών ηλεκτρομαγνητών από τον controller, διπλασιάζοντας έτσι τα πιθανά steps per revolution και αυξάνοντας την ακρίβεια καθώς πλέον μπορεί να κινηθεί και στο μισό ενός βασικού step. Το αρνητικό του half stepping είναι ότι με τη διαδικασία αυτή μειώνεται η ροπή του stepper motor. Ένας τρόπος να αποφευχθεί αυτό, είναι η αύξηση του ρεύματος στον ενεργό ηλεκτρομαγνήτη, όπως θα περιγραφεί αναλυτικότερα παρακάτω.[15]

Παρακάτω απεικονίζεται η κίνηση ενός stepper motor με ακρίβεια 3.6 μοίρες ανά step για τα 4 πρώτα steps, με τη σειριακή ενεργοποίηση των $4^{ων}$ πόλων που διαθέτει. Πρακτικά σημαίνει ότι το συγκεκριμένο stepper θέλει 100 steps για μία περιστροφή.

Εικόνα 10 : Ο $1^{ος}$ ηλεκτρομαγνήτης είναι ενεργός, και ελκύει την πλησιέστερη ακμή του γραναζιού του άξονα.
Εικόνα 11 : Ο $1^{ος}$ ηλεκτρομαγνήτης είναι ανενεργός, ενεργοποιείται ο $2^{ος}$ και ελκύει με τη σειρά του την πλησιέστερη ακμή του γραναζιού του άξονα, με αποτέλεσμα την περιστροφική κίνηση του stepper κατά 3.6 μοίρες.



Εικόνες 12,13 : Ο 3^{ος} και ο 4^{ος} ηλεκτρομαγνήτης ενεργοποιούνται αντίστοιχα με τη σειρά τους, προκαλώντας έτσι στροφική κίνηση του stepper motor για άλλα δύο steps. Μ' αυτόν τον τρόπο, σε μία ενεργοποίηση και των 4^{ων} πόλων, το συγκεκριμένο stepper έχει εκτελέσει περιστροφή 14,4 μοίρες, το 1/25 δηλαδή μίας πλήρους περιστροφής.



1.3 Ροπή ενός stepper motor

Τα stepper motors δεν παρέχουν τόση ροπή όση είναι σε θέση να παρέχουν DC servo motors ή DC gear motors. Το πλεονέκτημά τους παρόλα αυτά είναι η ακρίβεια που παρέχουν στον έλεγχο της περιστροφής τους. Ενώ για τα stepper motors η θέση τους είναι δεδομένη την κάθε στιγμή, καθώς τα βήματά τους είναι απολύτως συγκεκριμένα, τα DC motors απαιτούν σε εξωτερικό κύκλωμα έναν αναλογικό μηχανισμό (πχ ποτενσιόμετρο) ώστε ο μικροελεγκτής που ελέγχει το κύκλωμα να διαβάζει τις μεταβολές ενός μεγέθους. Για να επιτευχθεί πλήρης ονομαστική ροπή σε ένα stepper motor, τα πηνία πρέπει να φτάνουν την πλήρη ονομαστική τιμή για το ρεύμα τους σε κάθε step. Η επαγωγή από το πηνίο και το αντίστροφο μαγνητικό πεδίο που παράγεται από την κίνηση του ρότορα, τείνουν να αντιστέκονται σε αλλαγές στο ρεύμα οδήγησης, με τελικό αποτέλεσμα όσο το stepper αυξάνει ταχύτητα περιστροφής, τόσο και λιγότερο χρόνο μένει σε πλήρες ρεύμα, ούτως ώστε να μειώνεται τελικά η ροπή του.

Ροπή συγχρονισμού –Pull in torque:

Αυτό είναι το μέτρο της ροπής που παράγεται από ένα stepper motor όταν αυτό λειτουργεί χωρίς να επιταχύνει. Σε χαμηλές ταχύτητες το stepper motor μπορεί να συγχρονιστεί με μία εφαρμοσμένη συχνότητα βήματος, και η ροπή αυτή πρέπει να ξεπεράσει την τριβή και την αδράνεια. Επίσης είναι σημαντικό το φορτίο του κινητήρα να είναι τριβής και όχι αδράνειας

για να αποφευχθούν ανεπιθύμητες ταλαντώσεις στον περιστροφή. Η καμπύλη συγχρονισμού (pull in curve) καθορίζει την περιοχή έναρξης-παύσης . Σε αυτήν την περιοχή το stepper μπορεί να ξεκινάει και να σταματάει στιγμιαία με ένα εφαρμοσμένο φορτίο και χωρίς να χάσει σε συγχρονισμό.

Ροπή αποσυγχρονισμού-Pull out torque :

Ως ροπή αποσυγχρονισμού καλείται η μέγιστη ροπή που μπορεί να παράγει ένα stepper motor χωρίς να χάσει σε ταχύτητα περιστροφής. Η ροπή αυτή μετράται αρχικά με την επιτάχυνση του stepper στην επιθυμητή ταχύτητα και έπειτα με την αύξηση του φορτίου του stepper ώσπου αυτό να αρχίσει να καθυστερεί ή να χάνει βήματα. Η μέτρηση αυτή λαμβάνεται υπ' όψιν ανάμεσα σε ένα μεγάλο εύρος ταχυτήτων και τα αποτελέσματα χρησιμοποιούνται για την παραγωγή της δυναμικής καμπύλης λειτουργίας του stepper. Η καμπύλη αυτή επηρεάζεται από την τάση, το ρεύμα καθώς και τις τεχνικές εναλλαγής της φοράς του ρεύματος.[17],[18]

Holding Torque

Η ροπή που παράγει το stepper όταν έχει ονομαστικό ρεύμα να περνάει μέσα από τους πόλους αλλά αυτό βρίσκεται σε ακινησία.

Συντελεστές αποτελεσματικότητας:

Η ροπή συμπεριφέρεται κατ αναλογία με το ρεύμα στο πηνίο καθώς και με τον αριθμό των δακτυλίων του σύρματος στο πηνίο. Δηλαδή για παράδειγμα, για να αυξήσουμε τη ροπή κατά 20% αρκεί να αυξήσουμε το ρεύμα κατά 20% κοκ. Λόγω του μαγνητικού κορεσμού, δεν υπάρχει κανένα πλεονέκτημα στο να αυξήσουμε το ρεύμα περισσότερο από δύο φορές στην τιμή του ονομαστικού, και κάνοντας το υπάρχει κίνδυνος βλάβης στο stepper.

Η επαγωγή επίσης μειώνει τις επιδόσεις ενός stepper motor από άποψη ροπής όταν αυτό κινείται σε υψηλές ταχύτητες. Κάθε τύλιγμα εντός του stepper έχει μία συγκεκριμένη τιμή επαγωγής και αντίστασης.

Η ροή ενεργειών σε ένα stepper motor και τα τμήματα τα οποία είναι απαραίτητα για τη λειτουργία του είναι τα εξής [16] :

- **Indexers** – ο indexer (ή controller) είναι ένας μικροεπεξεργαστής- μικροελεγκτής ικανός να παράγει βηματικούς παλμούς και σήματα κατεύθυνσης προς τον driver. Επιπλέον αυτός εκτελεί τις υπόλοιπες λειτουργίες του κυκλώματος.
- **Drivers** - Ο driver (ή amplifier) μετατρέπει τα σήματα ελέγχου που έρχονται από το μικροελεγκτή-indexer στην απαραίτητη ισχύ για τον ερεθισμό των πηνίων εντός του stepper motor. Στην αγορά υπάρχουν πολλά διαφορετικά είδη driver με διαφορετικές ονομαστικές τιμές τάσης και ρεύματος, καθώς επίσης και διαφορετική τεχνολογία κατασκευής. Στην παρούσα διπλωματική χρησιμοποιήθηκε το Darlington array chip uln2003 που θα περιγραφεί αναλυτικότερα παρακάτω.
- **Stepper motors** – όπως περιγράφηκαν παραπάνω.

1.4 Εφαρμογές Stepper motors

Συχνά τα ηλεκτρονικά ελεγχόμενα stepper motors αποτελούν μέρη συστημάτων ελέγχου κίνησης και θέσης. Στο πεδίο της οπτικής και των lasers χρησιμοποιούνται σε εξοπλισμούς ακριβείας θέσης , ενώ άλλες χρήσεις τους είναι για παράδειγμα ο έλεγχος της ροής υγρών ή αερίων από δεξαμενές, με την ενσωμάτωσή τους στη βαλβίδα της δεξαμενής. Πιο εμπορικές και καθημερινές χρήσεις των stepper motors αποτελούν τα απαρχαιωμένα πλέον floppy disk drives, τα scanner, τα CD-DVD drives, σκληροί δίσκοι, εκτυπωτές και 3D printers καθώς επίσης και συστήματα τύπωσης κυκλωμάτων (PCB).

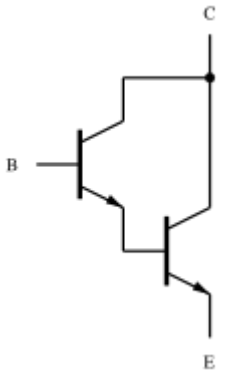
2 Stepper motor drivers

Η επίδοση ενός stepper motor εξαρτάται κατά πολύ από το κύκλωμα οδήγησης το οποίο χρησιμοποιεί. Οι καμπύλες ροπών μπορεί να επεκταθούν σε μεγαλύτερες ταχύτητες αν οι πόλοι του στάτορα κινηθούν πιο γρήγορα, με μόνο περιοριστικό στοιχείο η επαγωγή των πηνίων. Για την προσπέραση του προβλήματος αυτού και τη γρήγορη εναλλαγή μεταξύ των πόλων, πρέπει να αυξηθεί η τάση. Όμως αυτό οδηγεί στην ανάγκη να μειωθεί το ρεύμα προς αποφυγή βλαβών στο κύκλωμα. Για τα παραπάνω υπεύθυνο είναι το κύκλωμα του stepper motor driver.Για την παρούσα υλοποίηση χρησιμοποιήθηκε ο ULN2003 βασισμένος σε darlington transistor array.

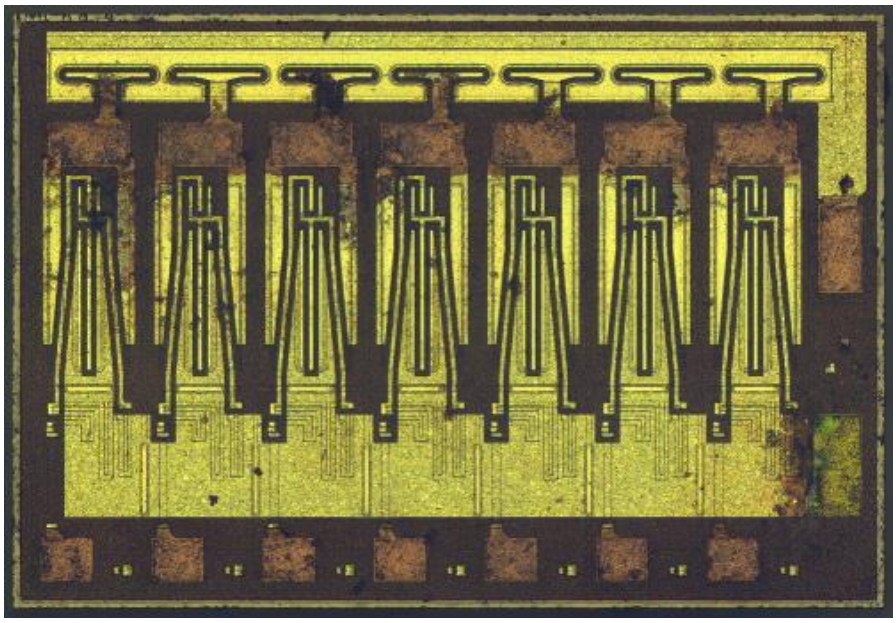
2.1 Darlington Transistor Array

Το Darlington transistor array που πάνω σ' αυτό βασίζεται η λειτουργία του ULN2003 , όπως αναφέρθηκε στο 3^{ο} κεφάλαιο, αποτελεί ένα σύστημα από δύο διπολικά transistor συνδεδεμένα με τέτοιο τρόπο ώστε το ρεύμα που ενισχύεται από το 1^{ο} transistor να ενισχύεται περεταίρω από το 2^{ο} . Αυτό το σύστημα δίνει μεγαλύτερο κέρδος ρεύματος μεταξύ common και emitter από ότι αν το κάθε transistor είχε χρησιμοποιηθεί ξεχωριστά , και στην περίπτωση των ενός ολοκληρωμένου όπως το uln2003 καταλαμβάνουν λιγότερο χώρο γιατί μπορούν να χρησιμοποιήσουν έναν κοινό συλλέκτη.[19]

Εικόνα 14 :Η διαμόρφωση Darlington που απεικονίζεται, ανακαλύφθηκε από τον μηχανικό των Bell Laboratories Sidney Darlington το 1953.[19]



Ο Stepper motor driver ULN2003, απεικονίζεται παρακάτω:



Εικόνα 15: Στην παραπάνω εικόνα απεικονίζεται το κύκλωμα ενός ολοκληρωμένου uln2003 ,το οποίο περιλαμβάνει 21 αντιστάσεις, 7 διόδους και 7 ζεύγη BJT transistor.[20]

3 Μικροελεγκτές

Οι μικροελεγκτές πρακτικά, είναι μικροεπεξεργαστές με ενσωματωμένη μνήμη (RAM, ROM, Flash) και γενικής χρήσης Inputs/Outputs (GPIOs) προγραμματιζόμενης λειτουργίας. Γενικά σε εφαρμογές αυτοματισμού χρησιμοποιούνται κατά κόρον μικροελεγκτές διαφορετικών προδιαγραφών και διαφορετικής αρχιτεκτονικής ανάλογα με την εκάστοτε εφαρμογή. Οι μικροελεγκτές βρίσκουν εφαρμογή σε συστήματα όπως δίκτυα αισθητήρων, home automation, καταμέτρηση ισχύος και ενέργειας κλπ. Συνήθως, για τον έλεγχο ενός συστήματος που περιέχει stepper motors είναι απαραίτητος ένας μικροελεγκτής με το ρόλο του Indexer , ικανός να παράγει τα κατάλληλα σήματα για την περιστροφή του stepper motor. Σε άλλες εφαρμογές που επεξεργάζονται αναλογικά δεδομένα, είναι απαραίτητο ο μικροελεγκτής να περιέχει ενσωματωμένο έναν Analog to Digital Converter (ADC), ο οποίος θα διαβάσει το αναλογικό σήμα και θα το μετατρέψει σε ψηφιακό. Στην περίπτωση που για την υλοποίηση της παρούσας διπλωματικής είχε χρησιμοποιηθεί άλλο είδος κινητήρα όπως servo motor ή απλό DC motor, θα ήταν απαραίτητο ο μικροελεγκτής που επιλέχθηκε για αυτήν τη διπλωματική να έχει ενσωματωμένο ADC ώστε να διαβάζει κάθε φορά τη θέση του κινητήρα από κάποιο εξωτερικό κύκλωμα όπως και προαναφέρθηκε.

Ένα παράδειγμα χρήσης μικροελεγκτή με stepper motor και web camera είναι το [1]. Στο συγκεκριμένο παράδειγμα χρησιμοποιείται ένας PIC μικροελεγκτής για τον έλεγχο ενός συστήματος όρασης σε ρομπότ, στο οποίο η camera βρίσκεται ενσωματωμένη με το stepper motor. Ο μικροελεγκτής περιστρέφοντας το stepper περιστρέφει και την κάμερα δίνοντας έτσι όραση 360 μοίρες στον χειριστή του συστήματος.

3.1 Μικροελεγκτές AVR

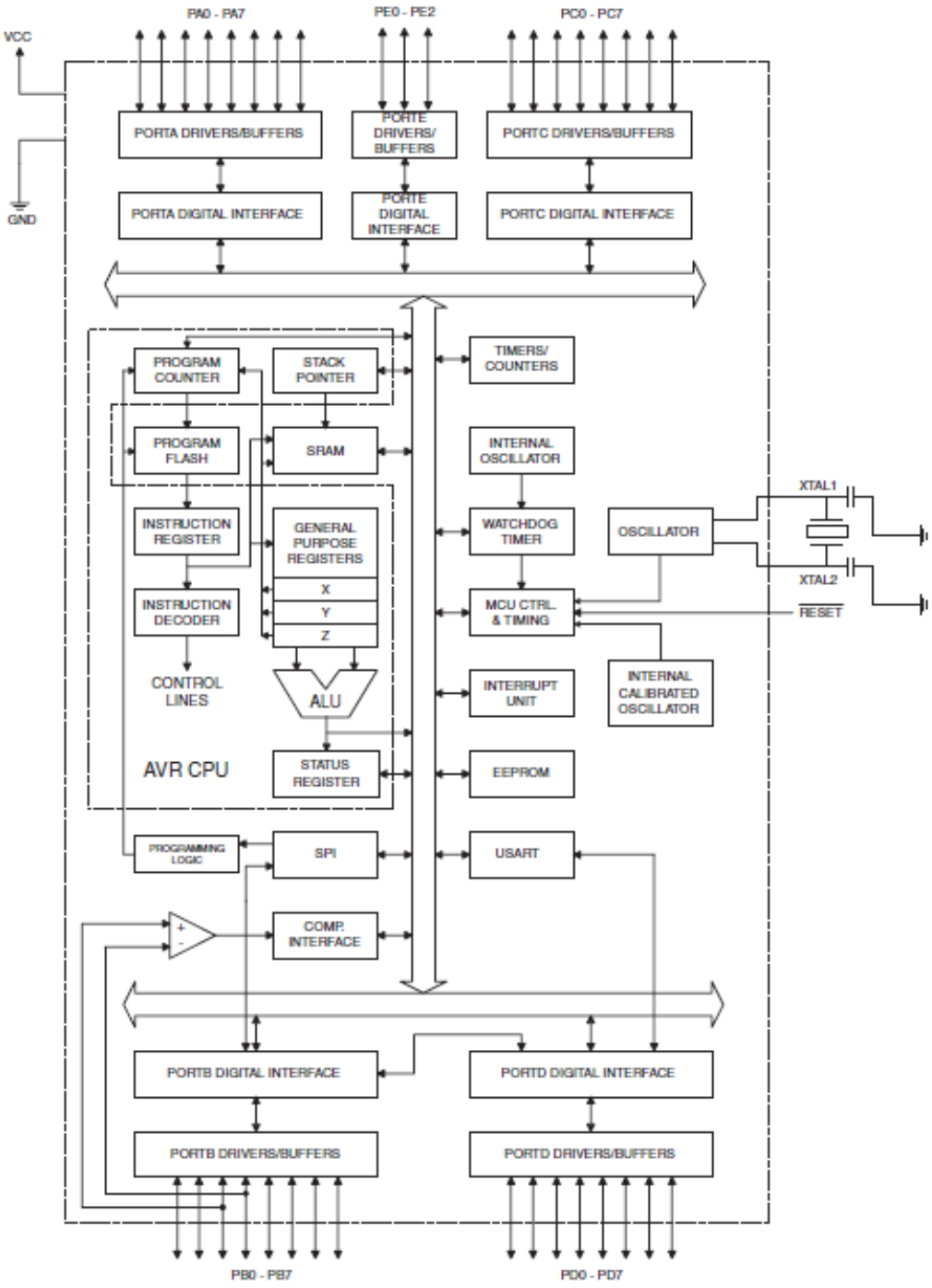
Ο τύπος μικροελεγκτών που μελετήθηκε και χρησιμοποιήθηκε στην υλοποίηση της παρούσας διπλωματικής είναι ο AVR. Οι μικροελεγκτές AVR αποτελούν μία σειρά ολοκληρωμένων κυκλωμάτων ανεπτυγμένων με βάση τη διαμορφωμένη αρχιτεκτονική υπολογιστικών συστημάτων του Harvard, στην οποία χρησιμοποιούνται διαφορετικά συστήματα μνήμης για την εγγραφή των δεδομένων και των εντολών. Οι AVR μικροελεγκτές αναπτύχθηκαν για πρώτη φορά από την εταιρία ATMEL το 1996 και η καινοτομία τους συγκριτικά με άλλους μικροελεγκτές της εποχής είναι ότι χρησιμοποιούν μνήμη Flash για τον προγραμματισμό τους αντί της μία μόνο φοράς προγραμματιζόμενης ROM που χρησιμοποιούταν ως τότε.

Η αρχιτεκτονική AVR υλοποιήθηκε για πρώτη φορά από δύο φοιτητές του Norwegian Institute of Technology (NTH), τους Alf-Egil Bogen και Vegard Wollan. Η αρχιτεκτονική ονομάστηκε αρχικά μRISC (Micro RISC). Ο επεξεργαστής των μικροελεγκτών AVR είναι σχεδιασμένος σε σύστημα RISC με 8 bit εντολές, αν και η εταιρία ATMEL παράγει και μικροελεγκτές αρχιτεκτονικής AVR με 32bit εντολές, τους AVR32.

Η στρατηγική σχεδίασης RISC (Reduced instruction set computing) είναι μία στρατηγική σχεδίασης που χρησιμοποιείται σε μικροεπεξεργαστές και χρησιμοποιεί απλοποιημένο instruction set προσφέροντας υψηλότερες επιδόσεις στην αρχιτεκτονική των μικροεπεξεργαστών, εκτελώντας εντολές σε λιγότερους κύκλους ρολογιού. Το σημαντικότερο κομμάτι της αρχιτεκτονικής AVR, είναι το RISC Register File ταχείας πρόσβασης, που αποτελείται από 32 8bit καταχωρητές γενικής χρήσης. Σε έναν κύκλο ρολογιού, ο AVR μπορεί να φορτώσει δεδομένα από δύο καταχωρητές του Register File στην ALU (Arithmetic and Logic Unit) , να πραγματοποιήσει την αντίστοιχη λειτουργία και να γράψει το αποτέλεσμα σε έναν άλλο καταχωρητή. Η ALU του AVR υποστηρίζει αριθμητικές και λογικές συναρτήσεις μεταξύ καταχωρητών, ή μεταξύ ενός καταχωρητή και μίας σταθεράς. Επίσης εκτελούνται και λειτουργίες που διαβάζουν δεδομένα από έναν μόνο καταχωρητή. [2]

Όσο αφορά τον προγραμματισμό ενός μικροελεγκτή, αυτός μπορεί να πραγματοποιηθεί είτε με κάποια γλώσσα υψηλού επιπέδου (HLL-High Level Language), είτε με γλώσσα Assembly. Η αρχιτεκτονική των AVR έχει αναπτυχθεί έτσι, ώστε να είναι δυνατόν να αναπτυχθεί ένας C compiler αργότερα για τον προγραμματισμό τους. Μειονέκτημα του προγραμματισμού ενός μικροελεγκτή σε C και όχι σε Assembly είναι το γεγονός ότι ο κώδικας καταλαμβάνει περισσότερο χώρο στη μνήμη, όμως με χρήση C ο προγραμματισμός γίνεται ευκολότερος.

Στην παρακάτω εικόνα παρουσιάζεται το block diagram του ATmega8515:



Εικόνα 16: Atmega8515 Block Diagram[3]

4 Πίνακας ASCII

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Εικόνα 17: ο πίνακας ASCII