Α.Τ.Ε.Ι. Θεσσαλονίκης
Τμήμα Ηλεκτρονικής

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ:

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

ΚΑΙ

ΕΦΑΡΜΟΓΕΣ ΤΗΣ ΣΕΙΡΑΣ ROBO TECHNOLOGY ΤΗΣ LEGO

ΚΥΡΙΑΚΙΔΗΣ ΧΡΗΣΤΟΣ **ΝΤΕΛΗΣ ΓΕΩΡΓΙΟΣ**

Επιβλέπον Καθηγητής: ΣΑΠΟΥΝΙΔΗΣ ΘΕΟΔΟΣΙΟΣ

Θεσσαλονίκη 2009

ΠΡΟΛΟΓΟΣ

Ο ήλιος πιθανότατα έλαμπε επάνω από την πόλη της Θεσσαλονίκης στις 16 Αυγούστου του 2008, όταν έπεσε στα χέρια των δύο φοιτητών ένα φυλλάδιο που μιλούσε για το Lego Nxt Robot. Δεν ήταν τυχαίο που ο ένας από τους δύο φοιτητές ήταν ο Κυριακίδης Χρήστος, και φυσικά, η επιτυχία ήταν ότι δίπλα του βρισκόταν ο Ντελής Γεώργιος. Ξαφνικά, το μάτι του ενός άστραψε. Ευθείς, κοίταξε τον άλλον στα μάτια και του είπε: **αυτή θα είναι η Πτυχιακή μας**.

Μετά από την συναίνεση του καθηγητή τους και την έγκριση της πτυχιακής εργασίας από την επιτροπή αξιολόγησης, ξεκινήσανε την έρευνα για το Lego Nxt Robot και τον τρόπο ανταπόκρισης του στις γλώσσες προγραμματισμού. Αρχικά, μπήκανε στη διαδικασία εύρεσης του κατάλληλου προμηθευτή για την αγορά του ρομπότ. Μετά από έρευνα τιμών στο διαδίκτυο καταλήξανε στο συμπέρασμα ότι η τιμή αγοράς του δεν είχε σημαντική διαφορά από την αποκλειστική αντιπρόσωπο της Ελλάδας, η οποία είναι η:

Διερευνητική Μάθηση Α.Ε. και στεγάζεται στην Γ. Βλάχου 13, Αθήνα.
Τηλ. +30 210 6779 800.

Η ρομποτική είναι ένας δύσκολος τομέας της ηλεκτρονικής, παρόλα αυτά, τους φάνηκε σαν μία πρόκληση η οποία έχει να τους δώσει πολλά περισσότερα στο μέλλον, από την εκπόνηση μιας «Πτυχιακής Εργασίας», αφού η ενασχόληση με το συγκεκριμένο θέμα απαιτούσε τη διερεύνηση των γνώσεών τους πάνω σε θέματα αυτοματισμού και ρομποτικής.

Αξίζει να σημειωθεί ότι η λέξη Lego προέρχεται από τη Δανία και αναλύεται ως εξής:

Le Go = Leg Godt (Δανεικά) = Play Well (αγγλικά)

ΕΥΧΑΡΙΣΤΙΕΣ

Θα θέλαμε να ευχαριστήσουμε θερμά για τη βοήθεια που μας προσέφεραν κατά την εκπόνηση της παρούσας πτυχιακής εργασίας:

Τον επιβλέποντα καθηγητή Σαπουνίδη Θεοδόσιο για την επίβλεψη της συγκεκριμένης πτυχιακής εργασίας.

Τον K. Bram Fokke , K. Brian Bagnall και τον Niels K. Handest για την πολύτιμη βοήθεια τους και τις βιβλιοθήκες τους.

Τέλος, θα θέλαμε να ευχαριστήσουμε τους γονείς μας και τις οικογένειές μας για την ηθική και υλική υποστήριξη που μας παρείχαν σε όλα αυτά τα χρόνια των σπουδών μας.

Περι Πτυχιακής Εργασίας

Οι φοιτητές που ανέλαβαν την περάτωση της εργασίας είναι :

Κυριακίδης Χρήστος του Ιωσήφ

ΚΑΣ : 504536

Έτος εισαγωγής : Εαρινό εξάμηνο 2004-2005

Ντελής Γεώργιος του Ευαγγέλου

ΚΑΣ : 505041

Έτος εισαγωγής : Χειμερινό εξάμηνο 2005-06

Επιβλέπον καθηγητής :

Σαπουνίδης Θεοδόσιος εργαστηριακός συνεργάτης

Ημερομηνία Ανάληψης :17/2/09

Ημερομηνία Περάτωσης : 11/08/09

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 3ο - Εισαγωγή στις γλώσσες προγραμματισμού

ΚΕΦΑΛΑΙΟ 4ο - Η Java

ΚΕΦΑΛΑΙΟ 5^{ο} - Το Visual Studio με τις VB.net και C sharp(C#)

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΠΑΡΑΡΤΗΜΑ

Περίληψη

Ο σκοπός αυτής της πτυχιακής εργασίας είναι η αναζήτηση στον ιστοχώρο και η εύρεση των πιο συνηθισμένων γλωσσών προγραμματισμού για το Lego NXT Robot, αξιοποιώντας τες, για ασύρματο έλεγχο με την Τεχνολογία Bluetooth.

Η μελέτη που παρουσιάζεται στη συγκεκριμένη πτυχιακή εργασία σχετίζεται με εφαρμογές του Lego NXT Robot και την ασύρματη επικοινωνία Bluetooth. Οι εφαρμογές είναι γραμμένες αντίστοιχα στη Java, την VB.net και την C#. Αρχικά, κάνουμε μία ιστορική αναδρομή στη ρομποτική και στη συνέχεια αναλύουμε τα χαρακτηριστικά (software και hardware) του Lego NXT Robot μαζί με τους αισθητήρες του. Έπειτα αναφερόμαστε στις γλώσσες προγραμματισμού και τον τρόπο λειτουργίας τους εμβαθύνουμε όμως την επιρροή αυτών των τριών γλωσσών στην ασύρματη επικοινωνία του Lego NXT Robot με τον υπολογιστή.

Λέξεις Κλειδιά

Ρομπότ, Lego, NXT, Visual Studio, NetBeans, Net Framework, Java, VB.net, C sharp, Mindsqualls, NXT#, i-command, Bram Fokke, RCX, Bluetooth, αισθητήρας, Σερβοκινητήρας, Hardware, Input Port, Output Port, Επαναφορτιζόμενη μπαταρία,

Abstract

The aim of this dissertation is the search in the website and the finding of most usual languages of planning for Lego NXT Robot, developing, for wireless control with the Technology Bluetooth.

The study that is presented in the particular final work is related with applications of Lego NXT Robot and the wireless communication Bluetooth. The applications are written respectively in Java, VB.net and C#. Initially, we make a historical retrospection in the robotics and afterwards we analyze the characteristics (software and hardware) Lego NXT Robot with its sensors. Then we were reported in the languages of planning and the way of operation˙ we deepen however the influence of these three languages in the wireless communication of Lego NXT Robot with the computer.

Key Words

Robot, Lego, NXT, Visual Studio, NetBeans, Net Framework, Java, VB.net, C sharp, Mindsqualls, NXT#, i-command, Bram Fokke, RCX, Bluetooth, Sensors, Rotation Sensor, Hardware, Input Port, Output Port, Rechargeable battery.

Αφιερωμένο

Στις Μητέρες μας,

Ασημούλα και Στεριανή

ΚΕΦΑΛΑΙΟ 1^{Ο}

Οριοθέτηση της ρομποτικής

1.1) Εισαγωγή

Η απαλλαγή από μονότονες και χειρωνακτικές εργασίες αποτελούσε πάντα την ανθρώπινη επιθυμία. Στον αιώνα μας, η δυνατότητα πραγματοποίησης ενός τέτοιου στόχου άρχισε να φαίνετε εφικτή με την ανάπτυξη των αυτοματισμών και ειδικότερα της ρομποτικής. Σύμφωνα με το Ινστιτούτο Ρομποτικής στης Αμερικής, Ρομπότ ονομάζεται μια μηχανή η οποία έχει ανθρωπόμορφη συμπεριφορά και εκτελεί ανθρώπινες εργασίες σύμφωνα με προγραμματισμένες εντολές του ανθρώπου. Δηλαδή είναι ένας αναπρογραμματιζόμενος και πολυλειτουργικός χωρικός μηχανισμός σχεδιασμένος να μετακινεί υλικά, αντικείμενα, εργαλεία ή ειδικευόμενες συσκευές με κατάλληλες μεταβλητά προγραμματιζόμενες κινήσεις που στοχεύουν στην βελτίωση της απόδοσης μιας σειράς εργασιών.

Οι σύγχρονοι ρομποτικοί μηχανισμοί κατάγονται από δύο εντελώς διαφορετικούς κλάδους:

• Από τα πρώιμα αυτόματα, που ουσιαστικά δεν ήταν τίποτε άλλο παρά ψυχαγωγικά «παιχνίδια» για μεγάλους

• Από τις ραγδαίες τεχνολογικές εξελίξεις στο χώρο της βιομηχανικής παραγωγής που είχε συνεχώς αυξανόμενες ανάγκες για όλο και πιο «έξυπνες» μηχανές οι οποίες θα μπορούσαν να αντικαταστήσουν επάξια τον άνθρωπο στην παραγωγική διαδικασία.

Ο Ήρων ο Αλεξανδρινός, Έλληνας σοφός του 1ου αιώνα π.χ. θεωρείται ο πατέρας της σύγχρονης ρομποτικής. Δίδαξε στο μουσείο της Αλεξάνδρειας και τα αυτόματά του περιγράφονται στο βιβλίο του «Πνευματικά και Αυτομοτοποιητική». Κατασκεύασε μεγάλο αριθμό αυτοκίνητων μηχανών, που λειτουργούσαν και κινούνταν από μόνες τους σαν όντα αληθινά, αξιοποιώντας τις ιδιότητες των υγρών και των αερίων, διαθέτοντας πολύπλοκα μηχανικά συστήματα και έναν ιδιοφυή προγραμματισμό κινήσεων. Κατά την παράδοση, που ίσως να περιλαμβάνει και υπερβολές κατασκεύασε μηχανικά πουλιά που κελαηδούσαν

έπιναν νερό και πετούσαν. Τα σχέδια που έχουν σωθεί μας δείχνουν ότι είχε κατασκευάσει μια βρύση που έτρεχε αυτόματα νερό, πύλες ναού που άνοιγαν αυτόματα, βωμούς που μπορούσαν να κινούνται με κάποιο πρόγραμμα κλπ.

Οπωσδήποτε για πολλούς αιώνες δεν φαίνεται να υπήρξαν μιμητές του. Στην Ευρώπη του 18ου αιώνα εκδηλώθηκε ξαφνικό ενδιαφέρον για τα αυτόματα μεταξύ παλιών επιτήδειων τεχνιτών. Σε μουσείο της Βιέννης διατηρείται ένας αυτόματος «γραφέας» από το 1753, μηχανισμός που είχε την ικανότητα να γράφει και να σχεδιάζει. Γάλλοι ωρολογοποιοί κατασκεύασαν πολλούς μηχανικούς ανθρώπους που έγραφαν, σχεδίαζαν ή έπαιζαν μουσικά όργανα. Φωτογραφίες στο μουσείο Τεχνών και Επιτηδευμάτων μας δείχνουν ότι ο Ζακ Ντε Βωκανσόν είχε κατασκευάσει μηχανοκίνητη πάπια που κούναγε τα φτερά της, έπινε νερό, τσιμπολογούσε καλαμπόκι και ακόμη «χώνευε» ή τουλάχιστον διέλυε το καλαμπόκι. Πιο σύγχρονα δείγματα κλασικών αυτομάτων αποτελούν οι κούκλες που βαδίζουν και μιλούν.

Ο όρος ρομπότ παράγεται από την Τσέχικη λέξη «ρομπότε» που σημαίνει αγγαρεία και χρησιμοποιήθηκε για πρώτη φορά από τον Κ. Τσάπεκ στο θεατρικό έργο «RUR» το 1920, όπου ρομπότ ονομάζονταν μηχανικοί άνθρωποι. Η παλαιότερη ελληνική λέξη αυτόματο, χρησιμοποιείται πλέον περισσότερο για μηχανισμούς που μιμούνται τον άνθρωπο ή κάποιο ζώο, χωρίς αναγκαστικά να παράγουν ωφέλιμο έργο. Ο νέος όρος «ανδροειδές» αναφέρεται σε ανθρωπόμορφους αλλά όχι όμως σε ζωόμορφους μηχανισμούς.

Η Ρομποτική είναι ένας νεοσύστατος τεχνολογικός κλάδος, παράγωγος της τεχνολογίας του αυτοματισμού και ασχολείται με τη μελέτη και την ανάπτυξη των ρομπότ, προγραμματιζόμενων δηλαδή μηχανισμών που χρησιμοποιούνται σε επιστημονικές ή βιομηχανικές εφαρμογές ως υποκατάστατα του ανθρώπου. Ένα ρομπότ μπορεί να μοιάζει στην εξωτερική του εμφάνιση με τον άνθρωπο, μπορεί να κινείται και να ενεργεί όπως ο άνθρωπος, αλλά μπορεί και όχι, είναι δε αρκετά δύσκολο να οριστεί η διαχωριστική γραμμή μεταξύ των ρομπότ και των απλών αυτοματοποιημένων μηχανών. Κατά γενικό κανόνα, όσο πιο περίπλοκη και εξειδικευμένη είναι μια μηχανή, τόσο μεγαλύτερη είναι η πιθανότητα να χαρακτηριστεί σαν ρομπότ.

Με την ανάπτυξη της τεχνικής των ρομπότ χωρίστηκαν σε δύο βασικές κατηγορίες:

• Τα ρομπότ που κατευθύνονται από τον άνθρωπο

• Τα ρομπότ με τεχνητή νοημοσύνη (ολοκληρωτικά), τα οποία δρουν

κατά κάποιο τρόπο «λογικά» χωρίς την ανάμειξη του ανθρώπου.

Τα περισσότερα σύγχρονα ρομπότ είναι ρομπότ χειριστές αν και υπάρχουν και άλλα είδη όπως πληροφόρησης, κινούμενα κλπ. Το βιομηχανικό ρομπότ- χειριστής έχει μηχανικά χέρια (ένα ή περισσότερα) και πίνακα ελέγχου ή ενσωματωμένη διάταξη προγραμματισμένης λειτουργίας. Μπορεί να χειρίζεται εξαρτήματα που ζυγίζουν από λίγα γραμμάρια μέχρι αρκετά κιλά, έχει ακτίνα δράσης μέχρι περίπου δύο μέτρα και μπορεί να εκτελεί από 200 μέχρι 1000 εργασίες την ώρα. Τα αυτόματα βιομηχανικά ρομπότ έχουν το σοβαρό πλεονέκτημα σε σχέση με τον άνθρωπο, ότι εκτελούν με μεγαλύτερη ταχύτητα και μεγαλύτερη ακρίβεια επαναλαμβανόμενες εργασίες. Τα ρομπότ αποτελούν το χαρακτηριστικότερο παράδειγμα συσκευής ευρείας χρήσης. Τα πλεονεκτήματα των ρομπότ, στα οποία οφείλεται η ευρεία χρήση τους, είναι η ακρίβεια και η επαναληψιμότητα. Ταυτόχρονα είναι σημαντικό να σημειωθεί ότι η απόδοση των ρομπότ είναι γενικά ανεξάρτητη από τον αριθμό των επαναλήψεων εκτέλεσης μιας εργασίας. Τα μειονεκτήματα των ρομποτικών βραχιόνων αναδεικνύονται κυρίως σε εργασίες που απαιτούν ‘’ νοημοσύνη ‘’ και σε εργασίες που εκτελούνται σε αβέβαιο περιβάλλον.

1.2) Εφαρμογές

Μεγαλύτερη εφαρμογή έχουν βρει τα ρομπότ χειριστές που κατευθύνονται από απόσταση και με «μηχανικό χέρι», που στηρίζεται σε κινητή ή ακίνητη θέση. Ο χειριστής διευθύνει την κίνηση του χεριού, ενώ το παρακολουθεί άμεσα ή σε τηλεοπτική κάμερα. Συχνά τα ρομπότ εφοδιάζονται με εκπαιδευμένο σύστημα που τα κατευθύνει με βάση κάποιο συγκεκριμένο πλάνο για την εργασία τους. Όταν σε ένα ρομπότ αυτού του είδους υποδεικνύεται η σειρά των διαδικασιών που πρέπει να εκτελέσει, το σύστημα διεύθυνσης

αποθηκεύει αυτή τη σειρά στο πρόγραμμα διεύθυνσης και ύστερα την επαναλαμβάνει με ακρίβεια. Τα ρομπότ χειριστές χρησιμοποιούνται για εργασίες σε σημεία απροσπέλαστα για τον άνθρωπο ή σε συνθήκες επικίνδυνες ή βλαβερές γι αυτόν, όπως στην πυρηνική βιομηχανία, στη χημική βιομηχανία κλπ. Κατά τη διάρκεια της δεκαετίας του 60 εμφανίστηκαν υποβρύχια ρομπότ χειριστές που ήταν ικανά να χειριστούν συσκευές και να κάνουν εργασίες σε μεγάλα βάθη στους ωκεανούς.

Πριν από λίγα χρόνια ένα τέτοιο ρομπότ χειριστής έφτασε μέχρι τον πλανήτη Άρη και μας έστειλε θαυμάσιες εικόνες και πάρα πολλές επιστημονικές μετρήσεις από τα όργανα που ήταν εφοδιασμένο.

Στα τέλη της δεκαετίας του 60 εμφανίστηκε μια νέα τεχνολογική τάση που συνδέεται με τη δημιουργία «λογικών» ρομπότ. Αυτά έχουν αισθητήρες που συλλέγουν πληροφορίες για την κατάσταση που επικρατεί στο κοντινό τους περιβάλλον (κάμερες για εικόνες, μικρόφωνα για ήχους, θερμόμετρα για μέτρηση εξωτερικής θερμοκρασίας, αυτόματους μετρητές αποστάσεων κλπ), έναν ηλεκτρονικό υπολογιστή για την επεξεργασία των παραπάνω πληροφοριών και κινητήριο σύστημα για να εκτελεί τις απαραίτητες ενέργειες. Στη βάση αυτών των στοιχείων ο τεχνητός εγκέφαλος διαμορφώνει το μοντέλο του περιβάλλοντος και παίρνει απόφαση (τεχνητή νοημοσύνη) για τη σειρά των ενεργειών που θα πραγματοποιηθούν από τους μηχανισμούς κίνησης που διαθέτει. Οι ενέργειες του έξυπνου ρομπότ αν το επιθυμούμε έχουν ορισμένες ομοιότητες με την ανθρώπινη συμπεριφορά. Οι εφαρμογές της ρομποτικής συνεισφέρουν στη μείωση του κόστους, την αύξηση της παραγωγικότητας και την βελτίωση της ποιότητας των παραγομένων προϊόντων. Επιπλέον, οι εφαρμογές της ρομποτικής απαλλάσσουν τον άνθρωπο από πολλές επικίνδυνες και ανθυγιεινές εργασίες (π.χ. ορυχεία πυρηνικοί αντιδραστήρες).

Είναι γεγονός αναμφισβήτητο ότι η εφαρμογή των ρομπότ μείνει θέσεις εργασίας σε ανειδίκευτο και χαμηλά ειδικευμένο προσωπικό. Αντίστοιχα, δημιουργεί νέες θέσεις εργασίας για ειδικευμένο προσωπικό. Είναι προφανές ότι μέχρι σήμερα η μείωση των θέσεων εργασίας δεν αντισταθμίζεται από τη δημιουργία νέων θέσεων.

1.3) Ιστορική αναδρομή της Lego Mindstorms από το RCX στο NXT

Η Lego προώθησε το MINDSTORMS NXT το φθινόπωρο του 2006, αλλά η ιστορία μας αρχίζει πραγματικά οκτώ έτη νωρίτερα το φθινόπωρο 1998, όταν εισήγαγε η LEGO το MINDSTORMS kit: Το σύστημα εφευρέσεων ρομποτικής (RIS) με το Robotic Control Explorer (RCX) (δείτε το σχήμα 1-1). Αντιπροσώπευσε την εξέλιξη πέντε δεκαετιών μορφωτικών τεχνικών κατασκευής από τη LEGO. Τα έτη είναι βασισμένα σε υπολογιστικές μεθόδους εκπαίδευσης από το Τεχνολογικό Ινστιτούτο της Μασαχουσέτης (MIT).

Σχήμα 1-1. Robotic Command Explorer (RCX) του 1998

Η LEGO παρήγαγε τα παιχνίδια οικοδόμησης από τα τέλη δεκαετίας του 1940, αρχίζοντας από τα γνωστά τουβλάκια που έχουμε γνωρίσει οι περισσότεροι στην παιδική μας ηλικία, και εκτείνετε έως τα σημερινά ιδιαίτερα τεχνικά κομμάτια, τα οποία περιλαμβάνουν ακτίνες, μηχανές (moter), αισθητήρες κ.α. Αν και το ύφος της οικοδόμηση ενός LEGO έχει αλλάξει, τα νέα κομμάτια έχουν ως σκοπό να είναι συμβατά και λειτουργικά με τα παλαιά. Αυτό το χαρακτηριστικό γνώρισμα συμβατότητας, και ο τεράστιος κατάλογος των μερών που μπορούμε να προμηθευτούμε, είναι μερικοί από τους λόγους για τους οποίους η LEGO είναι μια μεγάλη ελκυστική επιλογή για την κατασκευή.

Το MIT ερευνούσε τη χρήση των υπολογιστών στην εκπαίδευση από τα τέλη της δεκαετίας του 1970. Το όνομα MINDSTORMS λήφθηκε πραγματικά από έναν τίτλο βιβλίων του MIT του επιστήμονα υπολογιστών Seymour Papert. Η εργασία περιελάμβανε όχι μόνο την έννοια να χρησιμοποιήσει τους υπολογιστές για να παραδώσει τις εργασίες και να αξιολογήσει τα παιδιά, αλλά για τα παιδιά να μάθουν δηλαδή να χρησιμοποιούν τους υπολογιστές οι ίδιοι για την επίλυση προβλημάτων και την εφεύρεση νέων τεχνολογικών επιτευγμάτων.

Υποστηριζόμενες Γλώσσες προγραμματισμού :

- RCX Code (περιέχεται στις Mindstorm εκδόσεις λιανικής)
- ROBOLAB (βασίζεται στο LabVIEW και αναπτύχθηκε στο Tufts University)

Δημοφιλείς Γλώσσες τρίτων κατασκευαστών:

- C and C++ under BrickOS (formerly LegOS)
- Java under leJOS or TinyVM
- NQC ("Not Quite C")
- pbFORTH (επεκτάσεις της Forth γλώσσας προγραμματισμού)
- Visual Basic (μέσω του COM+ interface παρεχόμενο με το CD)
- RobotC (νέα γλώσσα συμβατή με την έκδοση NXT)

1.4) Το RCX και το σύστημα εφευρέσεων ρομποτικής

Το RIS ήταν επαναστατικό επειδή, για πρώτη φορά οι άνθρωποι μπόρεσαν εύκολα να μοιραστούν τις σύνθετες εφευρέσεις. Ο καθένας θα μπορούσε να χτίσει ακριβώς την ίδια εφεύρεση χωρίς παραδοσιακές δεξιότητες όπως η ξυλουργική, τα μέταλλα, η ηλεκτρονική, και ο προγραμματισμός. Ήταν επίσης ένα όφελος για τα εκπαιδευτικά ιδρύματα που χρειάστηκαν μια ανθεκτική και επαναχρησιμοποιήσιμη πλατφόρμα για τα προγράμματα εργαστηρίων εφαρμοσμένης μηχανικής. Το σχήμα 1-2 παρουσιάζει περισσότερα από 700 μέρη που περιλαμβάνονται με RIS.

Σχήμα 1-2. Το αρχικό σύστημα εφευρέσεων ρομποτικής (RIS) έτους 1998

Το RCX έχει ένα οκτώ bit επεξεργαστή, τρεις εισόδους, τρεις εξόδους, την υπέρυθρη επικοινωνία, ένα ηχείο, και μια τετραψήφια LCD οθόνη. Περιέχει έναν

μικροελεγκτή Renesas H8/300 ως εσωτερική ΚΜΕ του. Το τούβλο προγραμματίζεται με τη μεταφόρτωση ενός προγράμματος (που γράφεται σε μια από διάφορες διαθέσιμες γλώσσες προγραμματισμού) από ένα PC ή MAC στη RAM το υ το ψβλο υ μέσω μιας ειδικής υπέρυθρης διεπαφής (IR).

Αφότου αρχίσει ο χρήστης να δημιουργεί ένα πρόγραμμα γρήγορα θα καταλάβει ότι μια δημιουργία RCX Mindstorms μπορεί να λειτουργήσει από μόνη της, ενεργώντας στα εσωτερικά και εξωτερικά ερεθίσματα σύμφωνα με τις προγραμματισμένες οδηγίες. Επίσης, δύο ή περισσότερα τούβλα RCX μπορούν να επικοινωνήσουν το ένα με το άλλο μέσω της διεπαφής IR, επιτρέποντας τη συνεργασία ή τον ανταγωνισμό μεταξύ των τούβλων. Εκτός από τη θύρα IR, υπάρχουν τρεις θύρες εισαγωγής αισθητήρων και τρεις θύρες σύνδεσης μηχανών (επίσης χρησιμοποιήσιμοι για τους λαμπτήρες, κ.λπ.). Υπάρχει επίσης μία LCD που μπορεί να επιδείξει το επίπεδο φόρτισης των μπαταριών, την κατάσταση των θυρών εισόδου-εξόδου, και ποιο πρόγραμμα εκτελείται, καθώς και άλλες πληροφορίες. Τα τούβλα RCX έκδοσης 1.0 διαθέτουν μία παροχή ρεύματος για να επιτρέπουν τη συνεχή λειτουργία αντί της λειτουργίας περιορισμένου χρόνου κατά τη χρησιμοποίηση μπαταριών. Στην έκδοση RCX 2.0, η παροχή ρεύματος αφαιρέθηκε. Τα τούβλα RCX με παροχή ρεύματος είναι δημοφιλή για τα στατικά προγράμματα ρομποτικής (όπως τα ρομπότ βραχίονες) ή για τα πρότυπα μοντέλα τρένων Lego. Το RCX έχει τρεις θύρες εισόδου για αισθητήρες (π.χ. αισθητήρα αφής ή αισθητήρα φωτός) και τρεις θύρες εξόδου (π.χ. για τους κινητήρες ή για τα λαμπάκια). Οι χρήστες χτίζουν αρχικά το ρομπότ τους χρησιμοποιώντας τα κομμάτια LEGO και το RCX. Κατόπιν δημιουργούν ένα πρόγραμμα της αρεσκείας τους χρησιμοποιώντας όποια διαθέσιμη γλώσσα θέλουν (Robolab, NQC ή LEJOS) και το φορτώνουν στο RCX χρησιμοποιώντας μια ειδική υπέρυθρη συσκευή αποστολής σημάτων. Η δημιουργία τους μπορεί πλέον να αλληλεπιδράσει με το περιβάλλον πλήρως αυτόνομα. Η επικοινωνία γίνεται με τη βοήθεια του υπέρυθρου φωτός. Ένας υπέρυθρος αισθητήρας συνδέεται σε σειριακή θύρα ή σε θύρα USB. Μια ασύρματη σύνδεση με το ίδιο το RCX του επιτρέπει να κινηθεί ελεύθερα, ειδικά ως τμήμα κίνησης των οχημάτων ρομπότ

Λόγω της σοβαρής υπεραπλοποίησης, η γλώσσα ήταν σχεδόν ανώφελη αλλά χρήσιμη για τα πιο παλιά σχέδια. Οι άνθρωποι θέλησαν αμέσως να αντιστρέψουν το προϊόν

και να το εξελιξουν. Το ελάχιστα σε απόδοση Hardwork ασκεί επίσης την πίεση στην προσπάθεια.

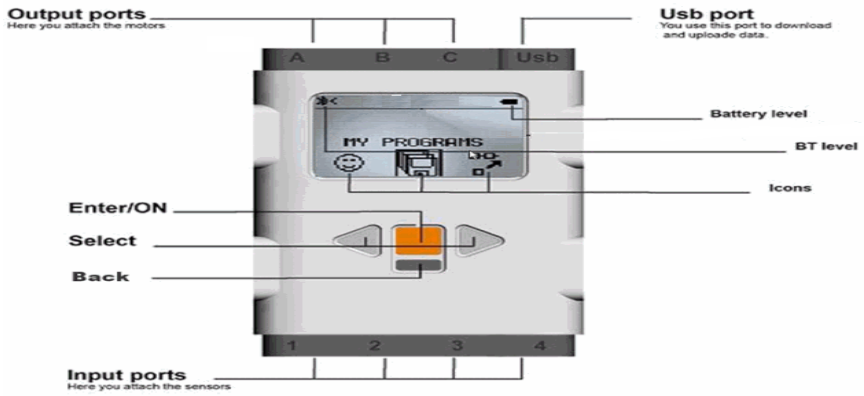
Μετά από το RCX, η LEGO εισήγαγε μια πιο περισσότερο προγραμματιστική κεντρική μονάδα η αποκαλούμενη Robotics Discovery System (RDS). Στόχευσε για ένα νεώτερο ακροατήριο από RIS, το οποίο είχε το λιγότερο ενδιαφέρον για τον προγραμματισμό. Επειδή το λειτουργικό σύστημά του ήταν στην αμετάβλητη μόνο ανάγνωσης μνήμη (ROM).

ΚΕΦΑΛΑΙΟ 2^{Ο}

Τι νέο υπάρχει στο Lego Mindstorms NXT

2.1) Εισαγωγή

Διάφορες αλλαγές εισάγονται στο NXT έναντι του προκατόχου του, το RCX. Με τη νέα γενιά NXT, η Lego προχωρεί ένα βήμα πιο πέρα από την επανάσταση των «οικιακής κατασκευής» ρομπότ που η ίδια είχε ξεκινήσει πριν από οκτώ χρόνια με τα Lego Mindstorms. Πολύ πιο εύκολα και γρήγορα στην κατασκευή τους, τα κιτ Mindstorms NXT δίνουν τη δυνατότητα στους ερασιτέχνες λάτρεις της ρομποτικής κάθε ηλικίας να φτιάξουν και να προγραμματίσουν το δικό τους μίνι ρομπότ. Η αναπροσαρμογή λογισμικού για το δημοφιλές σύστημα εφευρέσεων ρομποτικής LEGO MINDSTORMS NXT απελευθερώνεται. Η νέα έκδοση λογισμικού LEGO MINDSTORMS NXT1.1 τώρα παρέχει την υποστήριξη για Vista και Macintosh Windows. Με τη βελτιωμένη χρήση μνήμης του λογισμικού, το LEGO MINDSTORMS NXT περιλαμβάνει μικρότερα συνταγμένα προγράμματα και συμπιεσμένα αρχεία. Το NXT βασίζεται στο επιτυχημένο Robotics System Invention της εταιρείας, το οποίο έχει βελτιωθεί με την πρόσθεση νέων τεχνολογιών και αισθητήρων αυξημένων ικανοτήτων. Το «τουβλάκι» NXT που αποτελεί τον εγκέφαλο του ρομπότ είναι ένας αυτόνομος επεξεργαστής των 32 bit (σε αντίθεση με τα 16 bit της πρώτης γενιάς), ο οποίος μπορεί να προγραμματιστεί μέσω ηλεκτρονικού υπολογιστή.



Τα εξωτερικά χαρακτηριστικά του Mindstorms NX

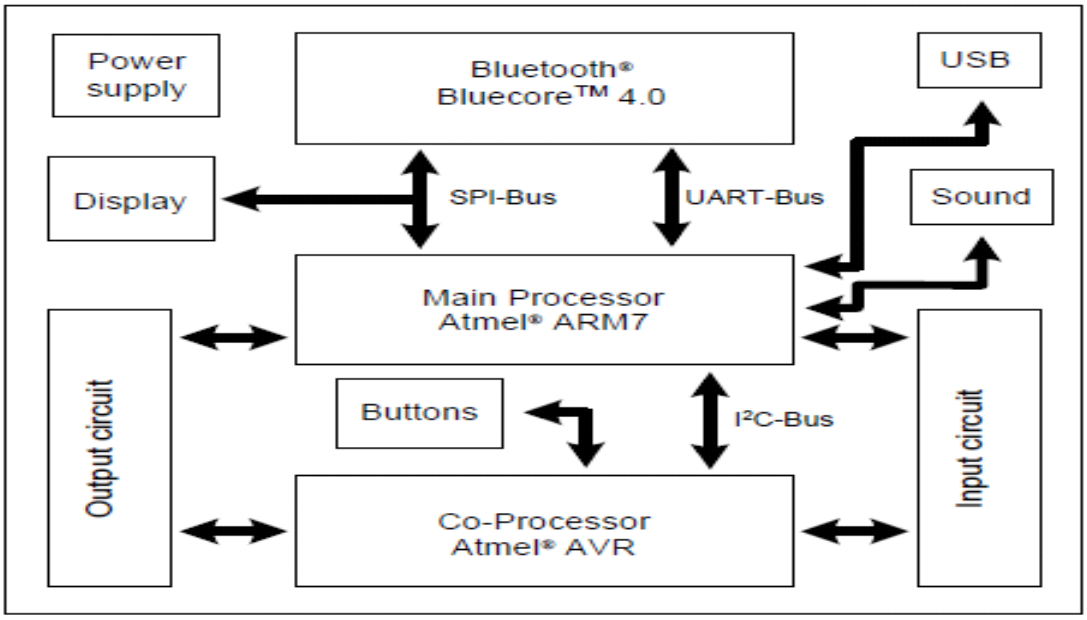
2.2) Σχετικά με τα κατασκευαστικά χαρακτηριστικά του Lego Mindstorms NXT

Κύρια μονάδα επεξεργασίας (MPU)	Atmel® 32-bit ARM® processor, AT91SAM7S256 - 256 KB FLASH - 64 KB RAM - 48 MHz
Μικροεπεξεργαστής	Atmel® 8-bit AVR processor, ATmega48 - 4 KB FLASH - 512 Byte RAM - 8 MHz
Bluetooth wireless communication	CSR BlueCoreTM 4 v2.0 +EDR System - Supporting the Serial Port Profile (SPP) - Internal 47 KByte RAM - External 8 MBit FLASH - 26 MHz
USB 2.0 communication	Full speed port (12 Mbit/s)
4 Εισόδους	Το καλώδιο εισόδου αποτελείτε από 6 συρμάτινες επαφές διασύνδεσης και υποστηρίζει αναλογική και ψηφιακή διασύνδεση - 1 high speed port, IEC 61158 Type 4/EN 50170 compliant
3 Εξόδους	Το καλώδιο εξόδου αποτελείτε από 6 συρμάτινες επαφές διασύνδεσης μέσα από αυτές εισέρχεται το σήμα από τους κωδικοποιητές

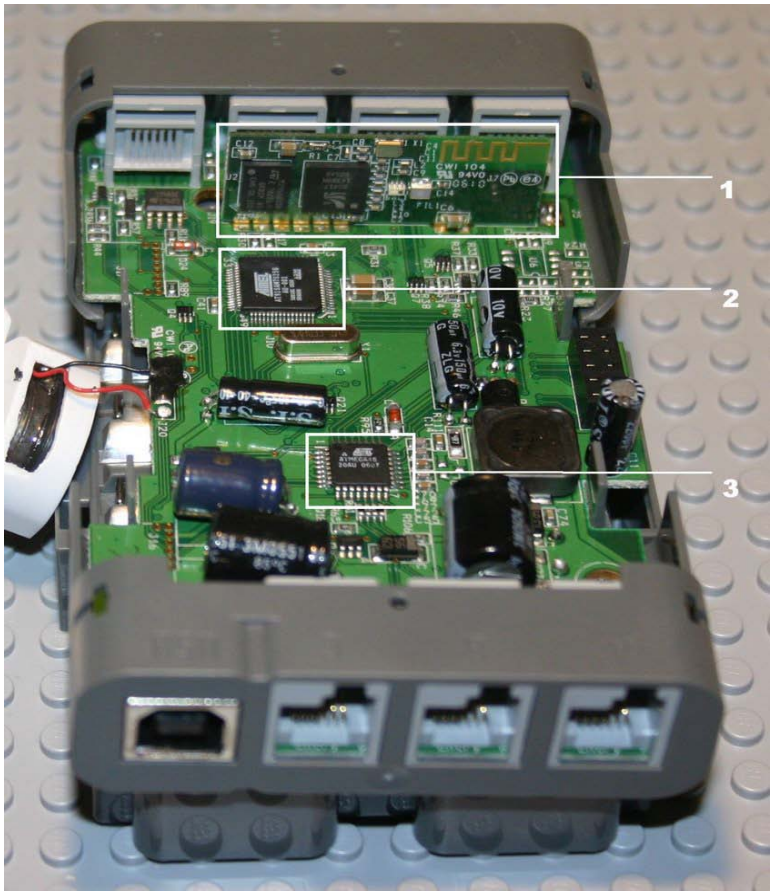
Μια οθόνη (Display)	100 x 64 pixel LCD black & white graphical display - View area: 26 X 40.6 mm
Μεγάφωνο	Κανάλι ήχου εξόδου με 8-bit ανάλυση -Υποστηρίζει ένα ποσοστό δειγμάτων 2-16 KHz
4 Κουμπιά (χειρισμού Menu)	Τα κουμπιά είναι Φτιαγμένα από καουτσούκ
Τροφοδοσία	6 AA μπαταριές - Οι αλκαλικές μπαταρίες συνιστάτε. - Η επαναφορτιζόμενη λιθίου μπαταρία 1400 mAH μπορεί να χρησιμοποιηθεί
Η διασύνδεση των εισόδων-εξόδων	Γίνετε με καλώδιο το οποίο προσαρμόζετε σε ένα συνδετήρα 6 επαφών, RJ12 και έχει ένα έλασμα στην δεξιά πλευρά του για να κουμπώνει στις εξόδους και εισόδους αντίστοιχα.

2.3) Γραφική απεικόνιση Hardware του Lego Mindstorms NXT

Στο παρακάτω γράφημα απεικονίζεται η γραφική αναπαράσταση για το πώς οι διαφορετικές λειτουργίες μονάδες που αναφέραμε παραπάνω συνδέονται και ελέγχονται από το ευφυές <<τουβλάκι>> Nxt



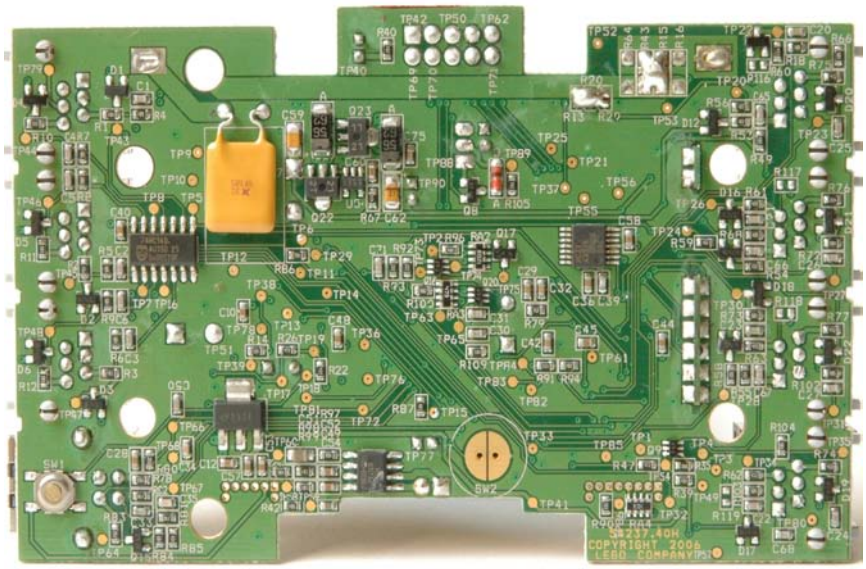
Όπως παρατηρήσαμε στο παραπάνω γράφημα το Nxt αποτελείται από 3(τρία) κύρια μέρη τα οποία τα εμφανίζουμε αριθμημένα και πάνω στην κύρια πλακέτα του Hardware.



(1)Το μικρό κύκλωμα -1 που παρουσιάζεται στην κορυφή είναι το Bluetooth .Το αριστερό ορθογώνιο είναι το Chip (M29W800DT) το οποίο δέχεται τροφοδοσία Vcc=3,6volt και κατά την εγγραφή και κατά την διαγραφή, είναι 8Mbit τέλος έχει 19 memory blocks της κάρτας μνήμης μαζί με το λογισμικό ελέγχου του Bluetooth από το Cambridge Silicon Radio. Το τετράγωνο chip είναι ο μικροελεγκτής CSR BlueCore 4 με Hardware όνομα BC417143BON606AJ όπου επικοινωνεί ασύρματα με τις συσκευές υποστηρίζοντας Bluetooth v2.0.Το ίχνος χρυσής λωρίδας ‘’ζιγκ ζαγκ’’ στην δεξιά γωνία είναι η κεραία του Bluetooth.

(2) Το περικυκλωμένο στο σχήμα παραπάνω με την ονομασία - 2 μας παρουσιάζει την Κύρια μονάδα επεξεργασίας την 32-bit ARM (AT91SAM7s256) στον οποίο τρέχουν όλα τα προγράμματα .Περιλαμβάνει την Flash memory, τα αρχεία του συστήματος, την μνήμη Ram και το πρωτόκολλο Usb.

(3) Το περικυκλωμένο στο σχήμα με την ονομασία – 3 παρουσιάζεται ο Μικροεπεξεργαστής 8-bit της Atmel Atmega-48 οδηγεί το PWM για τον έλεγχο των μοτέρ, οι αισθητήρες ανατροφοδοτούνται από τη κύρια μονάδα επεξεργασίας « T91SAM7s256 » .Δηλαδή ένα συνδέσουμε δυο μοτέρ πάνω στο Nxt το ένα στην αριστερή πλευρά του και το άλλο στην δεξιά και δώσουμε το ίδιο ποσοστό δύναμης κίνησης ο εκλεκτής θα προσπαθήσει και τα δυο μοτέρ να γυρίσουν με τον ίδιο ποσοστό δύναμης-ταχύτητας.



Αυτή είναι κάτω όψη της πλακέτας

Με μια προσεκτική μάτια θα παρατηρήσουμε RA4 και R90, είναι οι pull-up αντιστάσεις για το JTAG συνδετήρα J17(Δες διπλανό σχήμα.). Όπου χρησιμοποιείται στην περίπτωση που θέλουμε να προγραμματίσουμε την Κύρια μονάδα επεξεργασίας την 32-bit ARM (AT91SAM7s256).

Στη διπλανή εικόνα παρατηρούμε την συνδεσμολογία για τον προγραμματισμό του επεξεργαστή κάνουμε λοιπόν χρήση του Segger SAM-ICE JTAG που υποστηρίζει τον προγραμματισμό του επεξεργαστή μας, είναι της Atmel και συνδέεται με τον υπολογιστή μας μέσω Usb καλωδίου υποστηρίζει windows και Linux.

2.4) Οι Πόρτες (Ports) του Lego Mindstorms Nxt

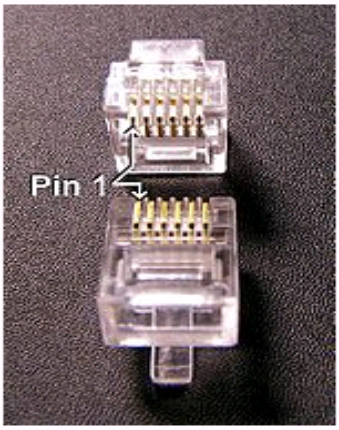
Προτού αρχίσουμε να μελετάμε του αισθητήρες και τα αποτελέσματα που παίρνουμε από αυτούς, πρέπει να καταλάβουμε πώς να τα συνδέσουμε με το NXT. Σε αυτό το κεφάλαιο, θα περιγράψουμε τον ασυνήθιστο συνδετήρα που χρησιμοποιείται στο NXT και θα καθορίσουμε τα ηλεκτρικά σήματα που βρίσκονται στις πόρτες (ports).

Οι τέσσερις πόρτες (ports) εισαγωγής αισθητήρων είναι στο κατώτατο σημείο του NXT, και είναι αριθμημένοι από 1 έως 4. Οι τρεις πόρτες (ports) σύνδεσης των μοτέρ στην έξοδο είναι στην κορυφή του NXT, και ονομάζονται A, B, και C. Εάν κοιτάξουμε στο τέλος ενός καλωδίου Nxt θα δούμε ότι υπάρχει ένα κλιψακι που αποτελείτε από έξι επαφές και τα καλώδια που αποτελούν την επαφή. Τα καλώδια είναι κωδικοποιημένα κατά χρώμα: άσπρο, μαύρο, κόκκινο, πράσινο, κίτρινο, και μπλε. Η λειτουργία των καλωδίων εξαρτάται

από το εάν χρησιμοποιούνται για εισαγωγή αισθητήρων ή για παραγωγή κίνησης από τα μοτέρ στην έξοδο. Η λειτουργία τους εξαρτάται ακόμη και από τον τύπο του αισθητήρα που συνδέεται.

2.5) Οι πόρτες εισόδου (input ports) του Lego Mindstorms Nxt

Στο παρακάτω Σχήμα -2 που παραθέτουμε, παριστάνουμε το όνομα του Pin στον ιδιαίτερο συνδετήρα που χρησιμοποιούμε για την διασύνδεση με το Nxt και το όνομα του κάθε καλωδίου ανάλογα με το χρώμα και το ρόλο που «διαδραματίζει» για την σωστή λειτουργία των αισθητήρων στις πόρτες εισόδου.

Pin	Name	Function	Color	Pin Numbering
1	ANA	Analog interface, +9V Supply	white	
2	GND	Ground	black	
3	GND	Ground	red	
4	IPOWERA	+4.3V Supply	green	
5	DIGIAI0	I^2C Clock (SCL), RS-485 A	yellow	
6	DIGIAI1	I^2C Data (SDA), RS-485 B	blue	

Σχήμα -2. Οι πόρτες εισόδου (input ports).

Pin 1- White (λευκό) → ANALOG

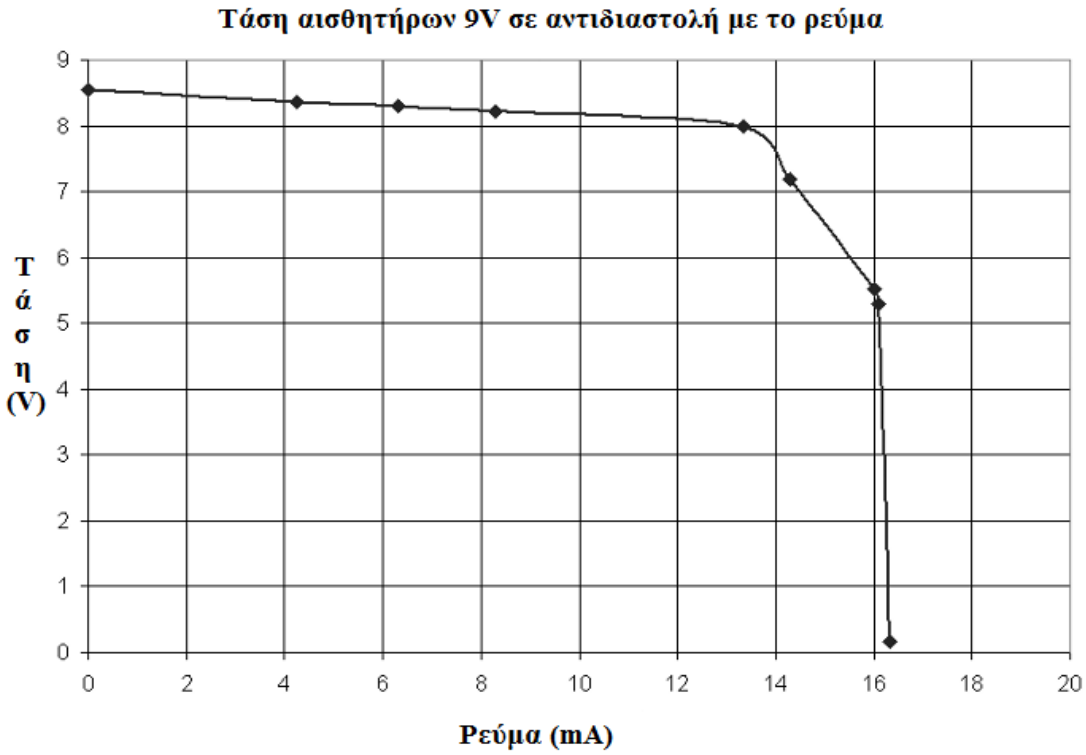
Μπορούμε να χρησιμοποιήσουμε το Pin 1 με δύο τρόπους: είτε ως αναλογική είσοδο, είτε ως 9V παροχή ηλεκτρικού ρεύματος που χρησιμοποιείται για τη συμβατότητα με τους παλαιούς αισθητήρες του RCX.

Όταν χρησιμοποιήσουμε το Pin 1 ως αναλογική είσοδο, το σήμα συνδέεται με έναν αναλογικό σε ψηφιακό μετατροπέα των 10 bit. Το σήμα στην είσοδο μπορεί να κυμανθεί από 0 ως 5V, και είναι αποκωδικοποιημένο σε μια ακατέργαστη ψηφιακή τιμή μεταξύ 0 και 1023. Η τιμή επιλέγεται κάθε 3 millisecond (msec). Το Pin 1 συνδέεται μόνιμα με 5V μέσω μιας pull-up αντίστασης 10KΩ, αυτό μας βοηθάει στην απλοποίηση πολλών σχεδίων αισθητηρίων.

Μπορούμε επίσης να χρησιμοποιήσουμε το Pin 1 ως 9V παροχή ηλεκτρικού ρεύματος. Η τάση είναι αυτή των μπαταριών που έχει για να λειτουργήσει το Nxt. Εάν χρησιμοποιήσουμε τις μπαταρίες NiMH, θα ήμαστε σε θέση να πάρουμε μόνο 7.2V. Μπορούμε να χρησιμοποιήσουμε αυτήν την παροχή ηλεκτρικού ρεύματος στους αισθητήρες που χρειάζονται τις υψηλότερες τάσεις. Παραδείγματος χάριν, ο αισθητήρας φωτός (light sensor) RCX χρησιμοποιεί αυτήν την παροχή ηλεκτρικού ρεύματος, και ο υπερηχητικός αισθητήρας NXT χρησιμοποιεί την ίδια παροχή για να πάρει περισσότερη δύναμη για τη συσκευή αποστολής σημάτων του.

Για αυτούς τους αισθητήρες, η τάση από το NXT προς τους αισθητήρες πηγαίνει για 3 ms και διαβάζουν την τιμή για 0.1 ms. Ο αισθητήρας χρειάζεται έναν πυκνωτή για να αποθηκεύσει τη τάση κατά τη διάρκεια της ανάγνωσης του διαστήματος.

Το **Σχήμα 3** παρουσιάζει τάση στην έξοδο στα διαφορετικά φορτία με καινούργιες μπαταρίες. Υπάρχει ένα τρέχον όριο περίπου 14 mA ανά πόρτα εισόδου. Εάν το ρεύμα είναι μεγαλύτερο από αυτήν την τιμή, η τάση μειώνεται γρήγορα.



Σχήμα-3. Τάση στην έξοδο

Pin-2 και Pin-3 ---Μαύρο και Κόκκινο → GND

Τα Pin-2 και Pin-3 είναι καλώδια που γειώνονται. Αυτά τα δυο Pins συνδέονται μαζί στο Nxt και στους αισθητήρες του Lego. Όλα τα σήματα έχουν αρχή μέτρησης και σημείο αναφοράς τα δυο Pins που πηγαίνουν στην γείωση. Ο αισθητήρας μπορεί να χρησιμοποιεί μόνο το ένα από τα δυο Pins, ή και τα δυο.

Pin-4---Πράσινο → 4,3Volt Τάση

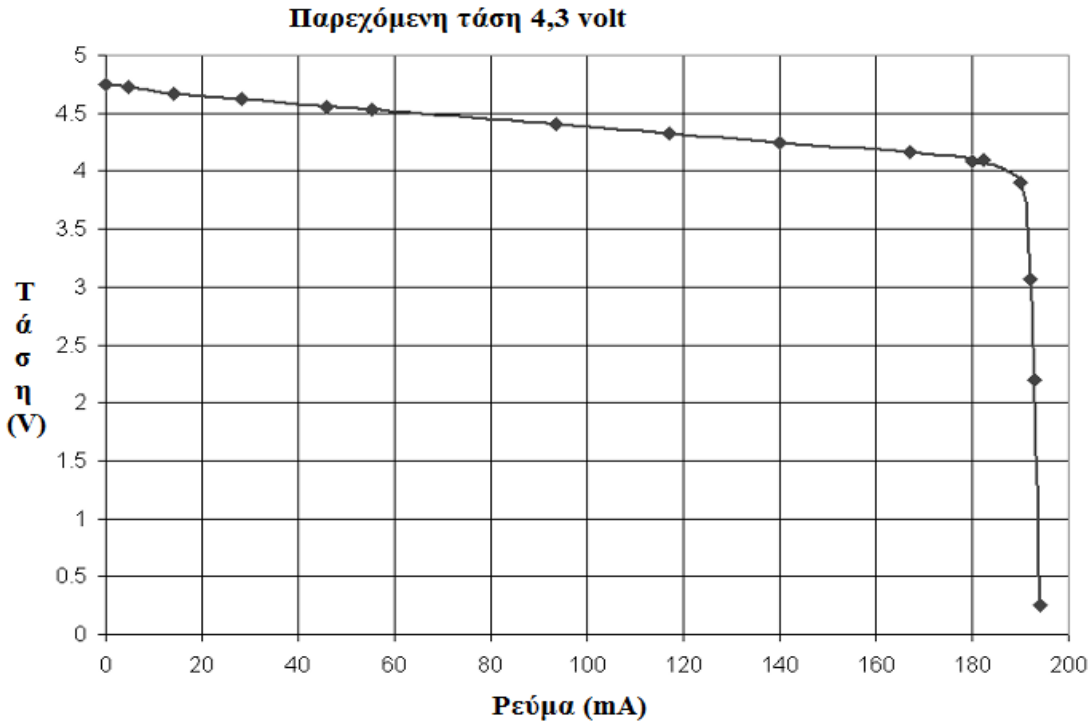
Αυτό το Pin είναι η παροχή κύριου ηλεκτρικού ρεύματος για όλους τους αισθητήρες NXT. Σε αντιδιαστολή με τη 9V παροχή ηλεκτρικού ρεύματος, αυτή η παροχή ηλεκτρικού ρεύματος έχει ένα συνολικό τρέχον όριο 180 mA και για τις επτά πόρτες (ports) εισόδου και εξόδου. Κάθε πόρτα (port) μπορεί να χρησιμοποιήσει 25 mA κατά μέσον όρο, αλλά είναι επίσης δυνατό να καταναλωθεί πιο πολύ ρεύμα σε μια πόρτα εάν κάποια άλλη καταναλώνει λιγότερο. Παραδείγματος χάρη, τα ρεύματα για τους τυποποιημένους αισθητήρες NXT και τους κωδικοποιητές μηχανών παρουσιάζονται στον πίνακα 2.

Τρέχουσα κατανάλωση ρεύματος απο τους ασθητήρες και απο τα Μοτέρ του Nxt

Touch	0
Sound	1.7 mA
Light Sensor (light off)	2.6 to 3 mA (depends on light)
Light Sensor (light on)	16.3 mA
Ultrasonic Sensor	4 mA
Motor position encoders	9 to 12 mA (depends on encoder position)
All RCX sensors and motors (via cable adapter)	0

Το Σχήμα-4 μας παρουσιάζει τις πτώσεις τάσης παροχής με το αυξανόμενο φορτίο. Πλησιάζει τα 5V χωρίς το φορτίο, και είναι μόνο 4.3V με ένα αρκετά δυνατό φορτίο. Αυτό δεν θα ήταν μια καλή παροχή ηλεκτρικού ρεύματος για τα ολοκληρωμένα κυκλώματα με 5V, αλλά τα νέα μέρη ψηφιακής-αναλογικής τεχνολογίας μπορούν να τροφοδοτηθούν από αυτήν. Ακόμα παρατηρούμε ότι μόλις ξεπεραστεί το συνολικό τρέχον

όριο 180 mA και για τις επτά πόρτες εισόδου-εξόδου η τάση πέφτει απότομα και δεν το αφήνει να ξεπεράσει το επιθυμητή όριο των 180mA .



Σχημα-4 Τάση στους αισθητήρες σε αντιδιαστολή με το ρεύμα.

Pin- 5 και Pin-6 είναι το Κίτρινο και το Μπλε αντίστοιχα → DIGIAI0 Και DIGIAI1

Μέσα από αυτά τα Pins περνάνε ψηφιακά σήματα των 3.3V και συνδέονται άμεσα με το μικροεπεξεργαστή του NXT. Χρησιμοποιούνται αρχικά για τις I^2C επικοινωνίες. Όταν τα Pins χρησιμοποιούνται ως έξοδοι, πρέπει να γνωρίζουμε ότι έχουμε ως όριο τα 3.3V. Όταν τα Pins χρησιμοποιούνται ως είσοδοι, το NXT έχει κύκλωμα προστασίας για να αποτρέψει τις υψηλές τάσεις έτσι ώστε να μην καταστραφεί τίποτα. Αυτό το κύκλωμα περιλαμβάνει μια αντίσταση 4.7kΩ που συνδέεται σε σειρά με την πόρτα εισόδου. Με αυτό το τρόπο, ακόμα κι αν η τάση του αισθητήρα είναι πάρα πολύ υψηλή, το αντίστοιχο ρεύμα θα είναι χαμηλό, έτσι ώστε τα ηλεκτρονικά κυκλώματα να μην χαλάσουν.

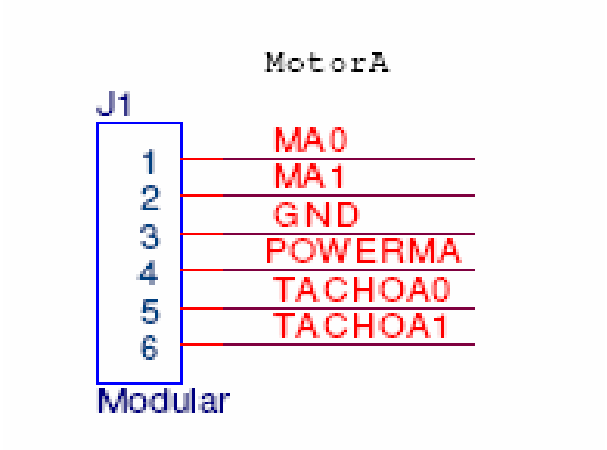
Εκτός από I^2C επικοινωνίες, το DIGIAI0 χρησιμοποιείται για τον αισθητήρα φωτός (light sensor) του NXT για να ελέγξει την κατάσταση της εκπέμπουσας φωτοδιόδου: όταν

είναι «ON» (κατάσταση αντανάκλασης) ή όταν είναι «OFF» (φως από περιβάλλον). Χρησιμοποιείται επίσης στον αισθητήρα ήχου (sound sensor) ως μεταγωγέας μεταξύ της dB κατάστασης (ακατέργαστο επίπεδο ήχου) και της dBA κατάστασης (φιλτραρισμένο επίπεδο ήχου για να το διασφαλίσει ψηλότερα από την ευαισθησία του ανθρώπινου αυτιού).

2.6) Οι πόρτες εξόδου (output ports) του Lego Mindstorms Nxt

Το Lego Mindstorms Nxt έχει τρεις πόρτες εξόδου που τις χρησιμοποιεί για να ελέγχει και να ενεργοποιεί την σύνδεση με το <<τουβλάκι>> Nxt. Ένα ψηφιακό περιβάλλον επικοινωνίας με την χρήση Έξι (6) καλωδίων εφαρμόζεται στις πόρτες εξόδου έτσι ώστε οι εξωτερικές συσκευές να μπορούν να παίρνουν πληροφορίες από το <<τουβλάκι>> Nxt

Στο σχήμα που ακλουθεί παρακάτω παρουσιάζουμε λεπτομερώς πληροφορίες της πόρτας εξόδου Α. Στην πόρτα Α το <<τουβλάκι>> Nxt στέλνει πληροφορίες σε ένα μοτέρ το οποίο το ονομάζουμε Moter Α γιατί το συνδέουμε στην πόρτα Α δηλαδή να έχουμε πλήρη έλεγχο της κίνησης των μοτέρ. Το σχηματικό της πόρτας Β και της πόρτας C είναι πανομοιότυπο.

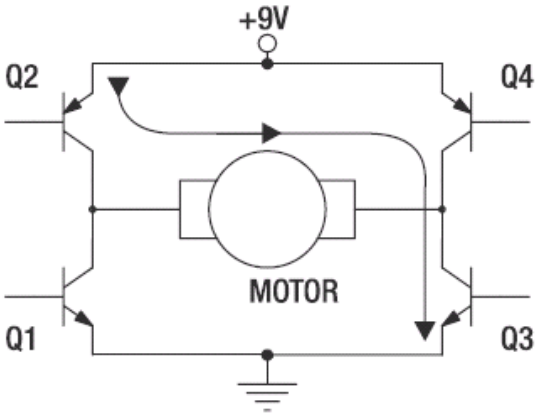


Σχήμα 5. Κύκλωμα πίσω από το Moter Α

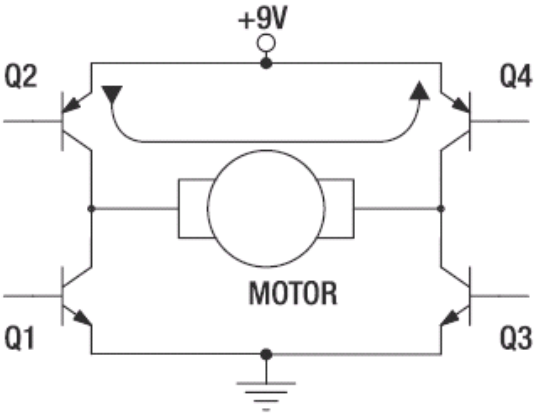
Στο σχημα-5 παρατηρούμε τα εξής :

Αριθμός Pin, Όνομα	**Χρώμα**	**Λειτουργία**
Pin 1, MA0	Λευκό	Μια παλμό-γεννήτρια PWM στέλνει σήμα για τον έλεγχο ενεργοποίηση
Pin 2, MA1	Μαύρο	Μια παλμό-γεννήτρια PWM στέλνει σήμα για τον έλεγχο ενεργοποίηση
Pin 3, GND	Κόκκινο	Μέσω του σήματος περνάει η γείωση στη έξοδο που είναι χρήσιμη για την τροφοδοσία
PIN 4, POWERMA	Πράσινο	Έχουμε 4,3volt τροφοδοσία στην έξοδο
PIN 5, TACHOA0	Κίτρινο	Δέχεται στη είσοδο μια τιμή που περιλαμβάνει αποδοχή και ενεργοποίηση της λειτουργιάς κίνησης του
PIN 6, TACHOA1	Μπλέ	Δέχεται στη είσοδο μια τιμή που περιλαμβάνει αποδοχή και ενεργοποίηση της λειτουργιάς κίνησης του

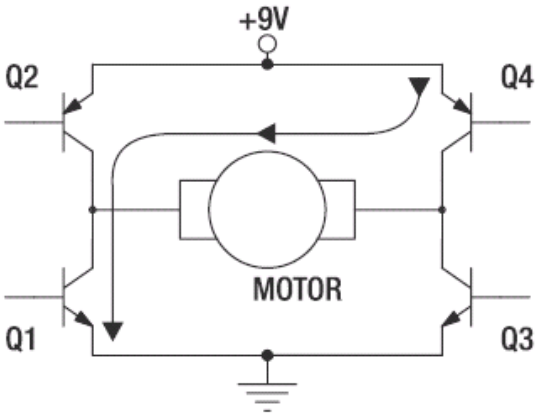
Πιο συγκεκριμένα, τα Pin 1, MA0 και Pin 2, MA1 είναι σήματα εξόδου για τον έλεγχο της ενεργοποίησης. Αυτά τα Pins προσφέρουν την τάση στα μοτέρ (motors). Η μέγιστη τάση είναι η τάση των μπαταριών (9V για τις τυποποιημένες μπαταρίες, 7.2V και 1500mAh για τις μπαταρίες NiMH [είναι επαναφορτιζόμενες μπαταριές οι κανονικές μπαταρίες έχουν ονομαστική τάση 1,5V ενώ οι επαναφορτιζόμενες 1,2V]).Το μοτέρ ελέγχεται από ένα κύκλωμα αποκαλούμενο Η-γέφυρα (με οδηγούς LB1836M και LB1930M του ολοκληρωμένου κυκλώματος). Μια Η-γέφυρα γίνεται από τέσσερα Transistors, τα οποία τα ονομάζουμε Q1 ,Q2, Q3, Q4. Το κύκλωμα ελέγχου σχεδιάζεται έτσι ώστε τα transistors Q1 και Q2 ναι είναι σε μια πλευρά και Q3 και Q4 στην άλλη πλευρά τα οποία δεν άγουν ταυτόχρονα, για να αποφύγουμε την ύπαρξη βραχυκυκλώματος. Το σχήμα-6 παρουσιάζει την κατάσταση των transistors για να «κινηθεί» προς τα εμπρός, με τα transistors Q1 και Q4 να είναι on. Το σχήμα-7 παρουσιάζει αντίστροφη κίνηση με τα transistors Q2 και Q3 να είναι on. Τα σχήματα-8 και 9 επεξηγούν την ροή του ρεύματος για το φρένο και την ελεύθερη κατάσταση .



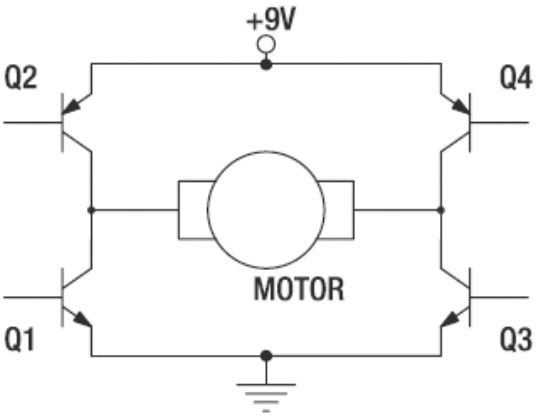
Σχήμα-6 Μπροστινή Κίνηση



Σχήμα-8 Φρενάρισμα

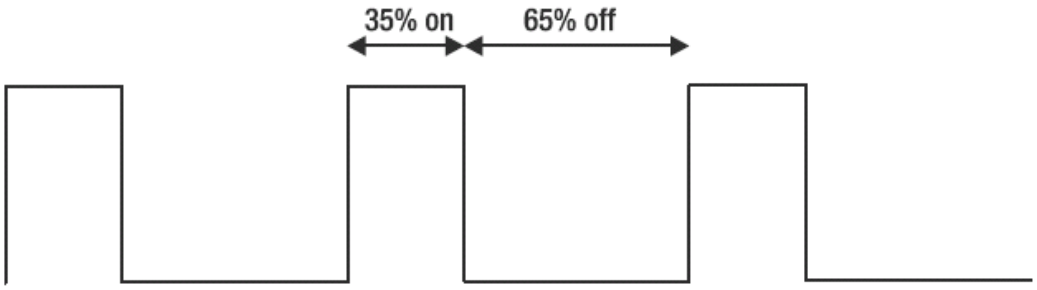


Σχήμα-7 Κίνηση προς τα πίσω



Σχήμα-9 Ελεύθερη Κίνηση

Η ταχύτητα των μοτέρ (moter) ελέγχεται από τη διαμόρφωση πλάτους του παλμού της (PWM), όπως παρουσιάζεται στο διάγραμμα σχήμα-9. Η ισχύς των μηχανών γίνεται on και off γρήγορα και πέρα από το χρονικό διάστημα. Η ταχύτητα ενός μοτέρ (moter) εξαρτάται από τη μέση τάση που εφαρμόζεται σε αυτό, και η χρήση της PWM είναι ένας τρόπος για τον έλεγχο αυτής της μέσης τάσης. Είναι αποδοτική μέθοδος επειδή τα transistors είναι εντελώς είτε κλειστά (off) είτε ανοικτά (on) .



Σχήμα-9. PWM για το μοτέρ (moter) για ποσοστό ισχύς 35%

Με το αρχικό firmware, το μήκος του κύκλου είναι 128μs. Αυτό αντιστοιχεί σε 7800Hz συχνότητα, η οποία είναι μια ευδιάκριτη συχνότητα που μπορεί μερικές φορές να ακουστεί ως υψηλό σφύριγμα. Το μήκος 128μs του κύκλου επιτρέπει μια πολύ καλύτερη γραμμικότητα της εντολής σε αντιδιαστολή με την χρήση της εντολής που θα χρησιμοποιούνταν από το RCX. Για το NXT, η σχέση μεταξύ της ταχύτητας και της τάσης που εφαρμόζουμε είναι γραμμική. Το διαθέσιμο ρεύμα στις πόρτες(ports) εξόδου είναι περίπου 700 mA, και μπορεί να επιτύχει 1A στο μέγιστο. Υπάρχει ένας οδηγός που έχει μια θερμική προστασία και περιορίζει ρεύμα όταν υπερθερμαίνει την συσκευή.

Pin-3 – Κόκκινο → GND

Αυτό το Pin είναι η γείωση. Αντίθετα από τα Pins των αισθητήρων, τα Pins-2 και Pins-3 για τα μοτέρ(moter) δεν συνδέονται μαζί. Εάν ένας αισθητήρας συνδεθεί τυχαία με μια πόρτα εξόδου προς τα μοτέρ, ο οδηγός βραχυκυκλώνεται.

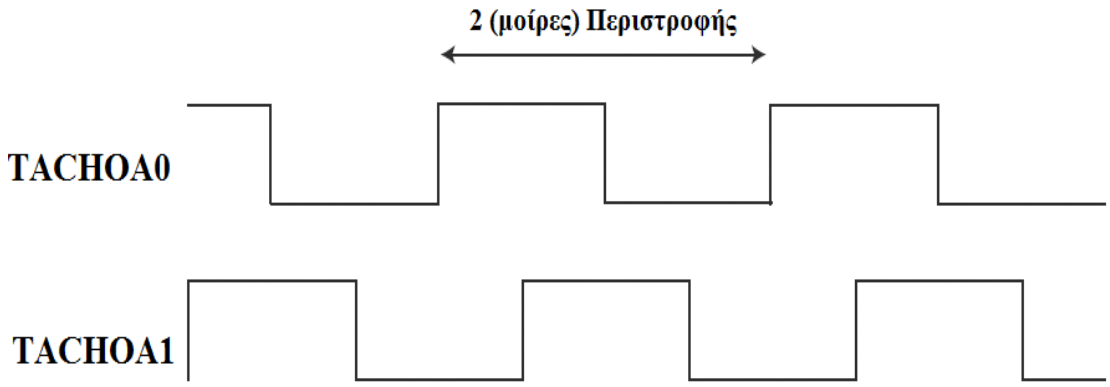
Pin-4 – Πράσινο → 4,3Volt Τάση

Αυτά το Pin συνδέεται με 4.3V παροχή ηλεκτρικού ρεύματος που μοιράζεται μεταξύ όλων των πορτών του NXT.

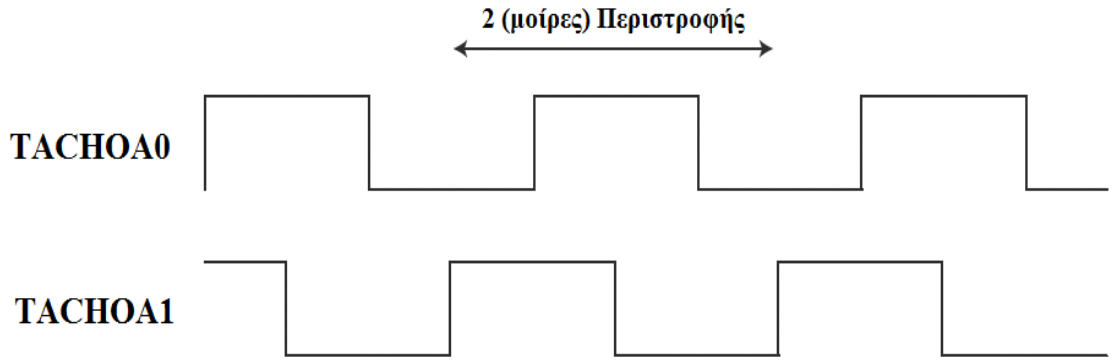
Pin-5 και Pin-6 → Κίτρινο και Μπλέ αντίστοιχα → TACHOA0 και TACHOA1

Οι δύο είσοδοι χρησιμοποιούνται για τον οπτικό κωδικοποιητή που είναι κατασκευασμένος μέσα στο μοτέρ του NXT. Ο κωδικοποιητής παράγει σήματα τετραγωνικού παλμού, και αυτός επιτρέπει στο NXT για να καθορίσει την κατεύθυνση και την ταχύτητα του μοτέρ. Τα δύο σήματα είναι μετατοπισμένα σε ορθογώνιους παλμούς, και

η μετατόπιση αντιπροσωπεύει το ένα τέταρτο της φάσης συνεπώς την χρονική περίοδο τετραγωνισμού. Το σχήμα-10 επεξηγεί τα σήματα για μοτέρ που πηγαίνει προς τα εμπρός, και το σχήμα-11 είναι για την αντίστροφη κίνηση «προς τα πίσω». Η συχνότητα του σήματος δίνει την περιστροφική ταχύτητα του μοτέρ. Ο μισό ς κύκλος του σήματος αντιστοιχεί σε μια μοίρα περιστροφής του μοτέρ.



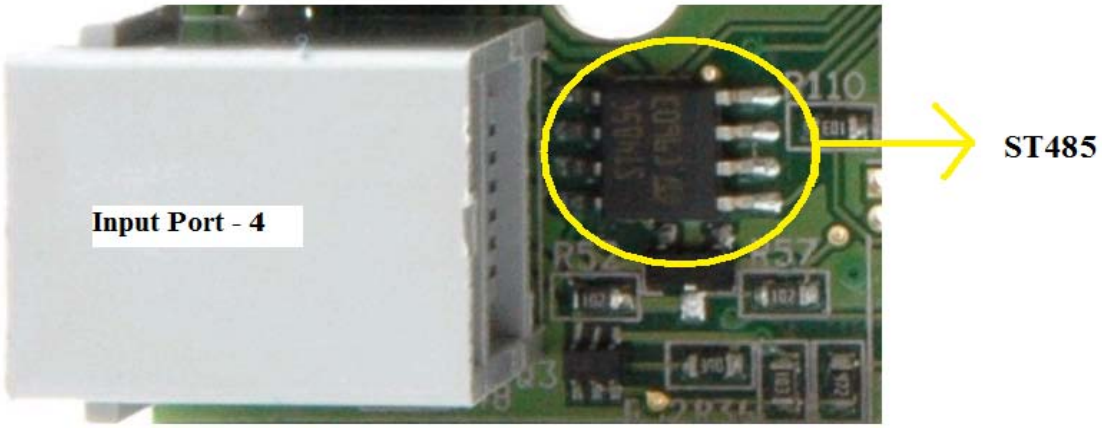
Σχήμα-10. Τετραγωνικοί παλμοί έτσι ωστε το μοτέρ να κινηθεί "μπροστά"



Σχήμα-11. Τετραγωνικοί παλμοί έτσι ωστε το μοτέρ να κινηθεί "πρός τα πίσω"

2.7) High-Speed Communication Port

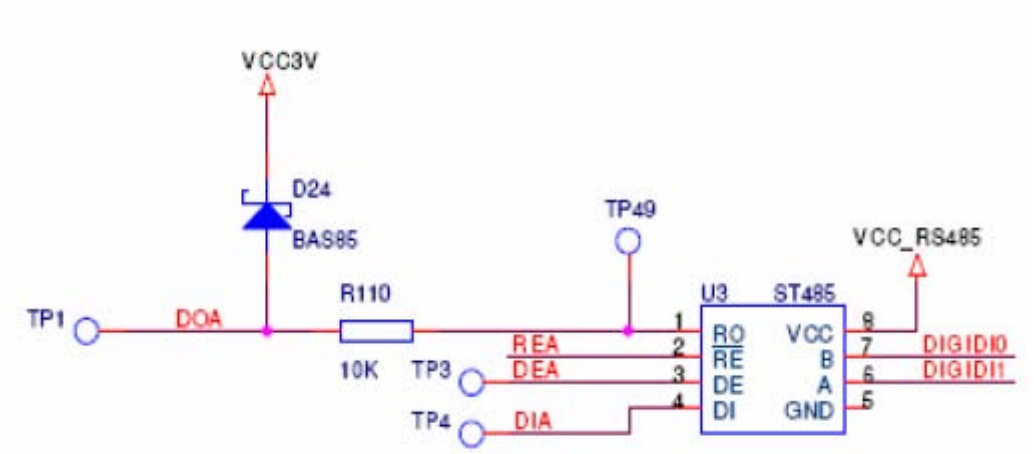
Η πόρτα (port input) 4 στο ευφυές «τούβλο» NXT μπορεί λειτουργήσει ως High Speed πόρτα επικοινωνίας. Επιτυγχάνεται με το τσιπ επικοινωνίας ST485 που εφαρμόζεται πίσω από το κανονικό κύκλωμα εισαγωγής (βλέπε σχημα-14). Αυτό το κύκλωμα επιτρέπει την εφαρμογή και μετάδοσης μεγάλων της αμφίδρομων πληροφοριών ανάμεσα σε δυο αλλά και περισσότερους σταθμούς διασύνδεσης σε μεγάλες αποστάσεις. Αυτήν την περίοδο LEGO δεν έχει αναπτύξει οποιεσδήποτε συσκευές που χρειάζονται αυτήν την λειτουργία επικοινωνίας. Εντούτοις, εάν οι μελλοντικές συσκευές που αναπτύσσονται απαιτούν τις υψηλότερες ταχύτητες επικοινωνίας, το «τούβλο» NXT είναι προετοιμασμένο.



Σχημα-14

Για τέτοιες μελλοντικές συσκευές, LEGO μπορεί να χρησιμοποιήσει το πρωτόκολλο επικοινωνίας P-Net το οποίο είναι βασισμένο στα πρότυπα του RS485 και κάνει χρήση ενός προστατευμένου καλωδίου. Αυτό επιτρέπει το μήκος του καλωδίου μέχρι 1200 μέτρα χωρίς επαναλήπτες. Η διάταξη διασύνδεσης έχει απομονωμένες και γαλβανισμένες επαφές ,μας δίνει την δυνατότητα να συνδεθούν 125 συσκευές ανά τμήμα και πάλι χωρίς τη χρήση επαναληπτών. Το P-Net είναι ένα αποδοτικό πρωτόκολλο μπορεί να χειριστεί και να ανταλλάξει μέχρι 300 δεδομένα-πληροφορίες το δευτερόλεπτο από 300 ανεξάρτητες διευθύνσεις. Τα στοιχεία μπορούν να μεταφερθούν υπό την μορφή

επεξεργάσιμων τιμών (θερμοκρασία, πίεση, ρεύμα, τάση κ.α.). Αυτό οδηγεί σε μια απόδοση μέχρι 9.600 δυαδικών σημάτων που προσεγγίζονται ανά δευτερόλεπτο από οπουδήποτε μέσα στο πλήρες σύστημα. Αυτό το υψηλό ποσοστό των πλήρως αναγνωρισμένων μεταδόσεων πληροφοριών μπορεί να επιτευχθεί, επειδή η P-Net Slaves χειρίζονται την επεξεργασία των στοιχείων της υποδοχής ή της μετάδοσης των πλαισίων, παράλληλα. Η επεξεργασία ενός αιτήματος από τα Slaves αρχίζει μόλις φτάσουν τα πρώτα bytes. Αυτό σε αντίθεση με την λειτουργία των άλλων των chips, όπου φτάνει ολόκληρο το frame πριν καλά καλά αρχίσει να το επεξεργαστεί.



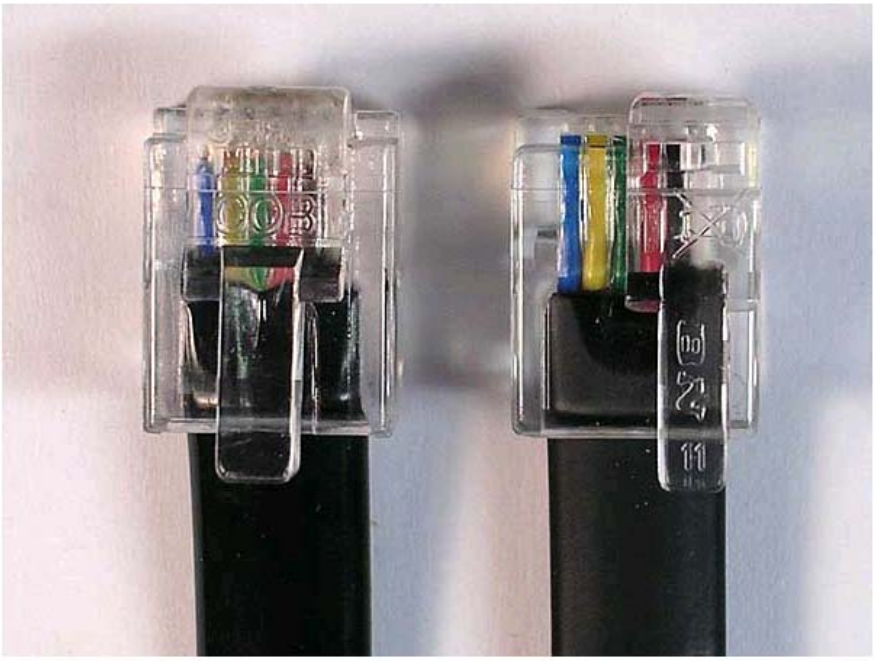
Σχήμα – 15. Το σχηματικό για το RS485 που βρίσκεται στην input port-4

Το chip ST485 χρησιμοποιεί 5 volt ως τάση λειτουργίας του και ο ARM7 επεξεργαστής χρησιμοποιεί 3.3 volt. Για αυτό έχει εφαρμοστεί μετατόπιση επιπέδων μεταξύ του chip ST485 και του ARM7 επεξεργαστή (δες σχημα-15). Οι ακόλουθες είναι παράμετροι επικοινωνίας:

- Ταχύτητα επικοινωνίας :921,6 Kbit/sec
- Δεδομένα bits : 8bit
- Σταματημένο bit : 1 bit
- Σφάλμα: 0 bi

2.8) Το καλώδιο σύνδεσης του Lego Mindstorms NXT με τις περιφερικές μονάδες

Η MINDSTORMS NXT εισάγει έναν νέο συνδετήρα για την σύνδεση με τα καλώδια αισθητήρων των μοτέρ. Ο συνδετήρας είναι παρόμοιος με έναν RJ-12 τηλεφώνου, αλλά εάν κοιτάξουμε προσεκτικά, θα παρατηρήσουμε ότι ο σύρτης δεν είναι στο κέντρο. Βρίσκετε τοποθετημένος στην αριστερή άκρη, και έτσι αποτρέπει τη χρήση των καλωδίων τηλεφώνων, είτε τυχαίων είτε σκόπιμων. Στο παρακάτω σχημα-12 παραθέτουμε τη διαφορά του καλωδίου NXT και του κλασικού καλωδίου RJ-12 τηλεφώνου.

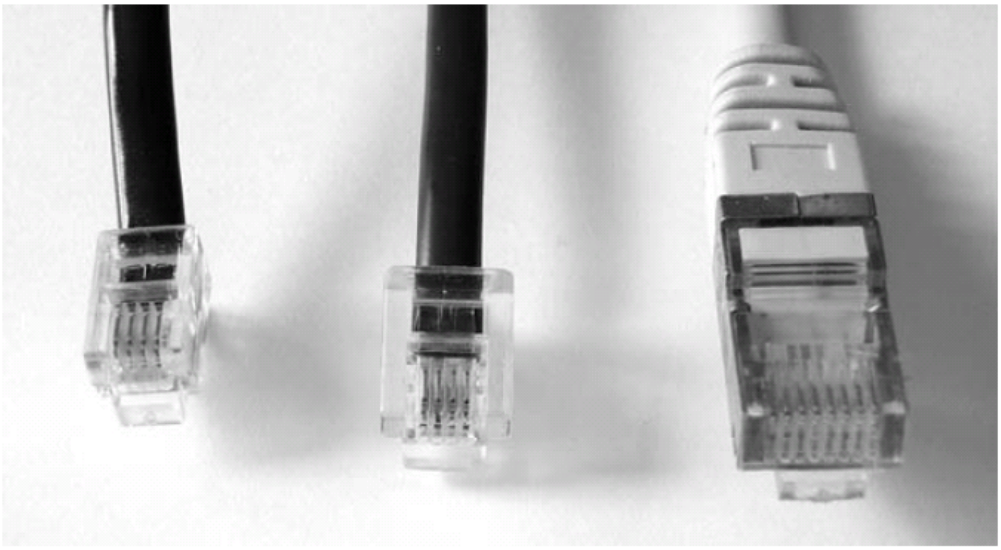


Σχήμα-12. Κλασικό καλώδιο RJ-12 και καλώδιο NXT

Πως θα φτιάξουμε το δικό μας καλώδιο διασύνδεσης

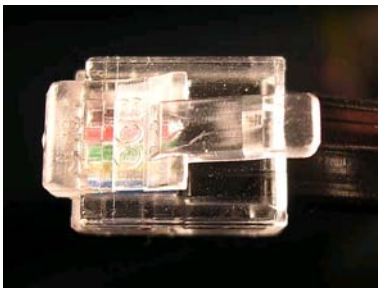
Εάν τα καλώδια LEGO δεν είναι διαθέσιμα ή εάν θέλετε να κάνετε δικά σας, μπορείτε να τροποποιήσετε RJ-12. Εντούτοις, να είστε προσεκτικοί κατά την επιλογή των καλωδίων που θα τροποποιηθούν. Τα RJ-12 καλώδια έχουν έξι αγωγούς σε έξι θέσεις, ενώ τα RJ-11 καλώδια (τυποποιημένα τηλεφωνικά καλώδια) έχουν τέσσερις αγωγούς σε έξι θέσεις, τα RJ-9 καλώδια έχουν τέσσερις αγωγούς σε τέσσερις θέσεις, και τα RJ-45 (τυποποιημένα καλώδια δικτύων) έχουν οκτώ αγωγούς σε οκτώ θέσεις (δείτε το σχήμα-13).

Δεν υπάρχει καμία ανάγκη να αγοράσουμε έτοιμα φτιαγμένα RJ-12 καλώδια. Μπορούμε να αγοράσουμε μερικά καλώδια και συνδετήρες, και να φτιάξουμε το δικό μας καλώδιο Nxt χρησιμοποιώντας ένα χαμηλού κόστους εργαλείο.

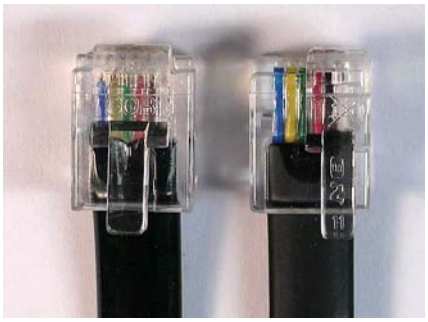


Σχήμα-13. Αυτοί οι συνδετήρες καλωδίων δεν μπορούν να υποστούν τροποποίηση για να λειτουργήσουν στο Nxt .

Η Lego χρησιμοποιεί ένα συνδετήρα όπως ανέφερα και πιο παραπάνω παρόμοιο με αυτό που χρησιμοποιούμε στα τηλέφωνα μας. Για λόγους ασφαλείας δεν θα μπορούσε να χρησιμοποίηση το ίδιο τηλεφωνικό συνδετήρα, έτσι επινόησαν και δημιούργησαν μια παραλλαγή. Βάλανε τον «συρτή», δηλαδή την ασφάλεια στην δεξιά πλευρά με αποτέλεσμα ο συνδετήρας Nxt δεν είναι διαθέσιμος οπουδήποτε. Μια έκδοση συνδετήρα αποκαλούμενη «DEC» μπορεί να βρεθεί εύκολα …αλλά με τον «σύρτη» σε λάθος πλευρά. Παρόλο την δυσκολία κατασκευής ενός καλωδίου Nxt σας παραθέτω την διαδικασία και τα βήματα που θα εξακολουθήσουμε για την δημιουργία του .

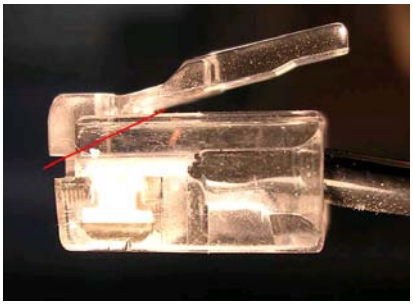


Αρχίζουμε με ένα κανονικό βούλωμα Rj-12, που εφαρμόστηκε στο καλώδιο. Το κλιψάκι πρέπει να έχει 6 υποδοχές διαθέσιμες το ίδιο και το καλώδιο αντίστοιχα για να μπορέσει να γίνει η τοποθέτηση των καλωδίων ανάλογα με τη σειρά χρώματος για να λειτουργήσει σωστά το καλώδιο. Ο χρωματικός κώδικας από τα αριστερά προς τα δεξιά είναι [Μπλε/ κίτρινο / πράσινο / κόκκινο / μαύρο / λευκό]. Ελέγχουμε τις πτυχές των καλωδίων και του συνδετήρα αν είναι καλές .

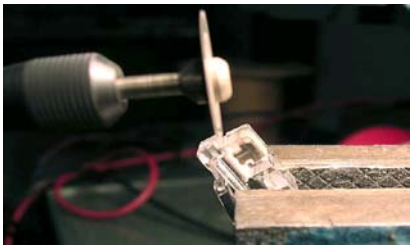


Συγκρίνοντας το κλιψάκι RJ-12 με ένα κλιψάκι Nxt παρατηρούμε δυο διαφορές :

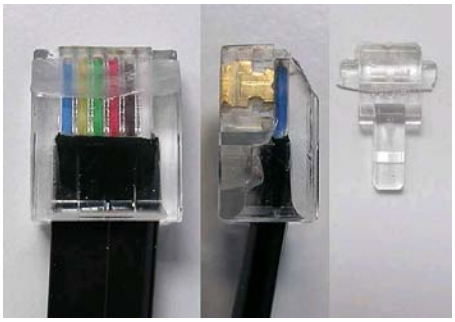
- Η ασφάλεια πρέπει να μετακινηθεί δεξιά στο κλιψάκι του RJ-12.
- Στο καλώδιο RJ-12 το μπροστινό μέρος είναι πιο ευρύτερο και θα πρέπει να συρρικνωθεί για να ταιριάζει στο μέγεθος των πορτών (Ports) του Nxt .



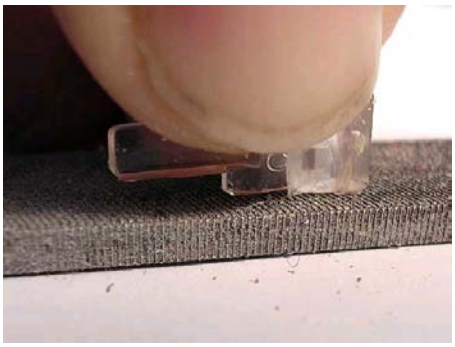
Το πρώτο βήμα είναι να κοπεί η ασφάλεια και πιο συγκεκριμένα να κοπεί μετά την κόκκινη γραμμή που παρουσιάζεται σε αυτήν την φωτογραφία.



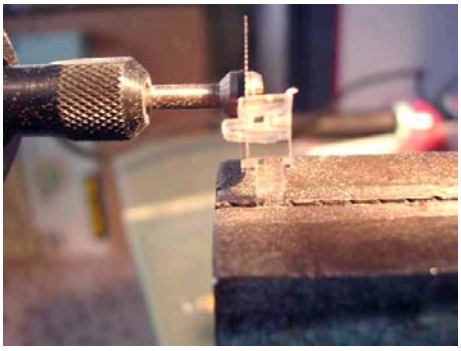
Η παραπάνω διαδικασία γίνεται με ένα πριόνι (με λεπίδα) όπως την παρουσιάζουμε στη διπλανή εικόνα.



Παρατηρούμε το αποτέλεσμα του κοψίματος όπου είναι φανερό το κλιψάκι χωρίς την ασφάλεια.



Ξύνουμε τις πλευρές τις ασφαλείας για να ταιριάζει με το πλάτος της ασφάλειας από το κλιψάκι του Nxt.

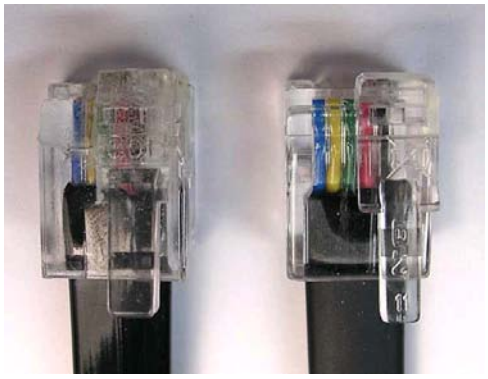


Εναλλακτικά, μπορούμε να χρησιμοποιήσουμε πριόνι για πιο γρήγορα αλλά πρέπει να ήμαστε πιο προσεκτικοί στο κόψιμο .

Ελέγχουμε την διαδικασία του κοψίματος βάζοντας την «ασφάλεια» στην υποδοχή ενός από τους αισθητήρες και πρέπει να ταιριάζει.

Τοποθετούμε τον συνδετήρα σε μια μέγγενη, βάζουμε διπλά ένα μικρό κομμάτι χαρτί για να μην επεκταθεί η κολλά στις επαφές του συνδετήρα και μετά βάζουμε την «ασφάλεια» πάνω στην κόλλα.

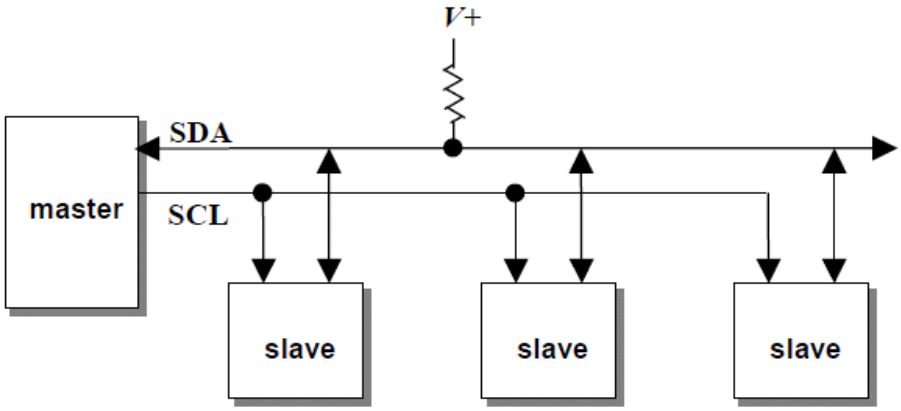
Θερμαίνουμε την κόλλα με ένα λαμπτήρα για ταχύτερο και πιο καλό κολλημα. δεν πρέπει να τοποθετήσουμε πολύ κοντά τον συνδετήρα γιατί μπορεί να λιώσει..



Τελικά, το αποτέλεσμα είναι εμφανές. Ο συνδετήρας που φτιάξαμε δεν έχει καμία διαφοροποίηση από τον συνδετήρα του Nxt.

2.9) I^2C Communication

Ο δίαυλος I^2C επινοήθηκε από τη Philips στις αρχές της δεκαετίας του 80. Από τότε έως σήμερα έχει χρησιμοποιηθεί σε αναρίθμητες συσκευές, από ενσωματωμένα συστήματα μέχρι την σύνδεση προσωπικών υπολογιστών με εξωτερικές ιδιοκατασκευές. Ο δίαυλος αυτός υποστηρίζει σειριακή επικοινωνία μεταξύ ολοκληρωμένων κυκλωμάτων κάθε τεχνολογίας (CMOS, NMOS, Bipolar). Στηρίζεται στην αρχιτεκτονική των δύο συρμάτων, με τη βοήθεια των οποίων τα επιμέρους ολοκληρωμένα κυκλώματα μιας συσκευής ανταλλάσουν σειριακά δεδομένα και σήματα συγχρονισμού μεταξύ τους. Οι δύο γραμμές το υ είναι η SCL και η SDA και είναι ανυψωμένες σε υψηλή τάση με αντιστάσεις ανύψωσης σε τάση (Pull Up Resistors), έχοντας τάση συνήθως από 3.3 V έως 5, αλλά αυτό ποικίλει από διάταξη σε διάταξη. Κάθε συσκευή που συνδέεται σε αυτές τις γραμμές, χαρακτηρίζεται από μια μοναδική διεύθυνση και μπορεί να λειτουργήσει ως πομπός ή ως δέκτης δεδομένων. Το εύρος διευθύνσεων του διαύλου είναι 7 bit, με 16 διευθύνσεις δεσμευμένες, έτσι ο συνολικός αριθμός ολοκληρωμένων(IC) που μπορούν να συνδεθούν στον δίαυλο είναι 112. Η πιο συνηθισμένη ταχύτητα που λαμβάνει χώρα σε έναν I^2C δίαυλο είναι αυτή των 100 kbit/sec και ονομάζεται standard mode. Υπάρχει ωστόσο και το low speed mode όπου οι ταχύτητες σε αυτό δεν ξεπερνούν τα 10 kbit/s. Οι συσκευές που συνδέονται στον I^2C διακρίνονται σε Master και Slave. Ο Master είναι αυτός που έχει την πρωτοβουλία στο κύκλωμα και στέλνει τους παλμούς χρονισμού στις συσκευές που είναι εξαρτώμενες (Slave). Η αρχή αποστολής δεδομένων γίνεται με τη συνθήκη εκκίνησης (Start). Αντίστοιχα η παύση αποστολής δεδομένων γίνεται με τη συνθήκη τερματισμού (Stop). Τα δεδομένα μεταφέρονται στον δίαυλο κατά bytes. Ανάμεσα στα bytes που αποστέλλονται από τον πομπό προς τον δέκτη, ο δέκτης παράγει παλμούς επιβεβαίωσης ACK (Acknowledge).



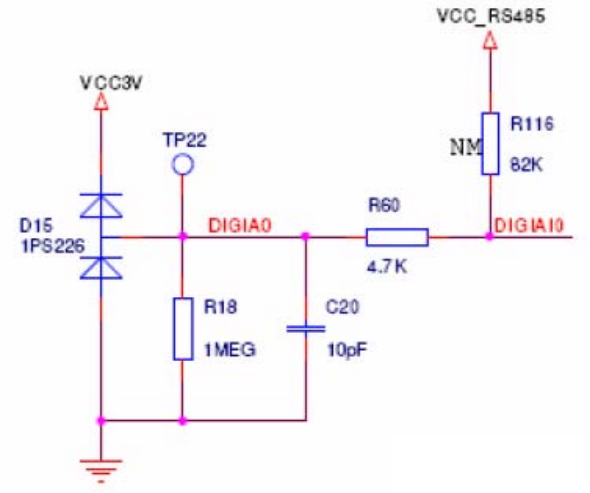
Σχήμα – 16. Ο δίαυλος I^2C διαθέτει 2 γραμμές

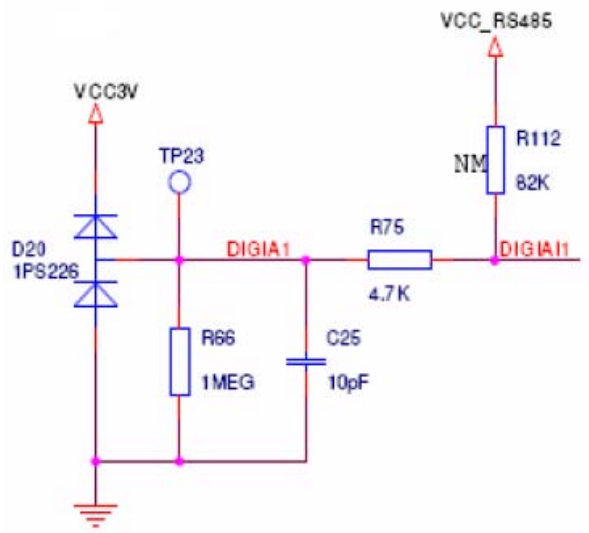
Στο σχήμα- 16 παρατηρούμε : Serial Data (SDA): η σειριακή γραμμή μεταφοράς δεδομένων, δύο κατευθύνσεων. Η γραμμή αυτή οδηγείται από εξόδους open-collector/drain, γεγονός που επιτρέπει την ταυτόχρονη πρόσβαση στη γραμμή. Όταν μία έξοδος είναι σε χαμηλή λογική κατάσταση, συνδέει τη γραμμή SDA στη γείωση. Όταν μία έξοδος είναι σε υψηλή λογική κατάσταση, αποσυνδέεται από τη γραμμή. Για να αποκτήσει η γραμμή SDA υψηλή κατάσταση απαιτείται αντίσταση pull up (βλ. σχήμα 16). Στην ταυτόχρονη πρόσβαση της γραμμής, αρκεί μία έξοδος να είναι '0', για να είναι η γραμμή SDA '0'. Η γραμμή είναι '1', μόνον όταν όλες οι έξοδοι είναι '1'. Εάν απαιτείται η μετάδοση δεδομένων από μία συγκεκριμένη έξοδο, όλες οι υπόλοιπες έξοδοι θα πρέπει να είναι '1', αλλιώς η γραμμή SDA θα είναι πάντα '0'. Serial Clock (SCL): το σειριακό ρολόι χρονισμού, οδηγείται από τον master. Δεν υπάρχουν σήματα επιλογής (chip select), διότι τα μεταδιδόμενα πακέτα δεδομένων περιέχουν τη διεύθυνση του παραλήπτη. Η διεύθυνση αυτή καθορίζεται εν μέρει από το είδος του slave και κατά δεύτερο λόγο από ορισμένους ακροδέκτες εισόδου του κάθε slave. Οι ακροδέκτες αυτοί συνδέονται σταθερά (hard-wired) στο VCC ή GND και καθορίζουν μέρος της διεύθυνσης του slave. Στον δίαυλο I^2C τα δεδομένα μεταφέρονται σε λέξεις των 8 bits (MSB πρώτα) και η λήψη κάθε λέξης επιβεβαιώνεται (acknowledge).

Το πρωτόκολλο επικοινωνίας I^2C λειτουργεί ως μια ψηφιακή διασύνδεση για τις εξωτερικές συσκευές που πρέπει να επικοινωνήσουν με το NXT. Έχει μια ψηφιακή διασύνδεση με τις εξωτερικές συσκευές για να εκτελεί της λειτουργιές χωριστά και μετά να στέλνει το αποτέλεσμα πίσω στο NXT ή να λαμβάνει τις νέες πληροφορίες από το Nxt.

Το Nxt έχει τέσσερα I^2C κανάλια επικοινωνίας, ένα για κάθε έναν από τους τέσσερις πόρτες εισόδου. Η I^2C ψηφιακή επικοινωνία εφαρμόζεται ως κύρια λειτουργία, έτσι αυτό σημαίνει ότι το NXT ελέγχει τη ροή πληροφοριών σε κάθε ένα από τα κανάλια επικοινωνίας.

Μια σημαντική πλευρά στην I^2C επικοινωνία μεταξύ δύο συσκευών είναι το Hardware μέσα σε κάθε μια από τις συσκευές. Τα παρακάτω σχήματα παρουσιάζουν το hardware - σχηματικό για την πόρτα εισόδου -1 στο NXT. Η σχηματική αναπαράσταση για τις υπόλοιπες πόρτες- 2, 3 και 4 είναι η ίδια όσον αφορά την I^2C επικοινωνία.





- Υπάρχει μια αντίσταση 4,7KΩ σε σειρά με την γραμμή σήματος
- Δεν υπάρχει κανένας pull-up αντιστάτης που να τοποθετείται εσωτερικά στο NXT. Αυτό σημαίνει ότι πρέπει να τοποθετηθεί στην εξωτερική συσκευή. Συστήνουμε τη χρησιμοποίηση αντιστάτες 82KΩ ως pull-up αντιστάτες και στο Serial Data (SDA) και στο Serial Clock (SCL).
- DIGIx1 το pin-5 είναι το σήμα του CLK και το DIGIx1 το pin-6 είναι το σήμα Data για την I^2C επικοινωνία.
- Τα ψηφιακά I/O pins στο NXT δεν μπορούν να σταλθούν στον αγωγό άμεσα. Επομένως το NXT θα οδηγήσει τα ψηφιακά I/O pins είτε ως master είτε ως slave ανάλογα με την κατάσταση, δηλ., το Nxt χρησιμοποιεί push-pull.

Όταν το NXT δεν πρέπει να ελέγξει τις I/O γραμμές, θα τεθεί ως είσοδος(π.χ., κατά το ανάγνωση των στοιχείων από μια συσκευή ή κατά την ανάγνωση της αναγνώρισης).

- Η I^2C έχει ταχύτητα μεταφοράς δεδομένων 9600 bit/sec

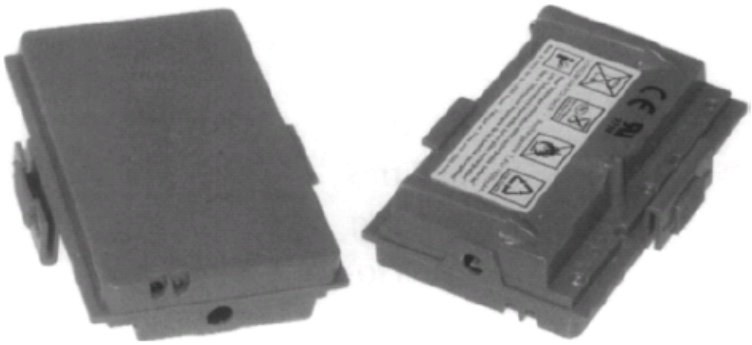
- Κάθε κανάλι δέχεται 16 byte στην είσοδο και 16 byte στην έξοδο. επομένως ένα μπορούν να σταλθούν 16 bytes μέγιστα και να παραλειφτούν κατά τη διάρκεια ενός κύκλου επικοινωνίας .

- Εάν οι πολλαπλοί αισθητήρες είναι συνδεμένοι σε σειρά στην ίδια πόρτα(port) η pull-up αντίσταση που θα προκύπτει πρέπει να είναι 82ΚΩ. Γι αυτό το λόγω θέλει κάποια προσοχή όταν πολλοί αισθητήρες συνδέονται στην ίδια πόρτα.

Οι ψηφιακές συσκευές έχουν μερικά πλεονεκτήματα έναντι των αναλογικών συσκευών. Οι ψηφιακές συσκευές μπορούν να περιλάβουν τα ονόματα συσκευών και να παραπέμψουν διάφορες μεμονωμένες παραμέτρους που είναι ειδικές για κάθε συσκευή (όπως τις τιμές βαθμολόγησης, τους χρόνους ξεκινήματος, και ούτω καθεξής). Για να είναι σε θέση να διακρίνει τις διαφορετικές ψηφιακές συσκευές μεταξύ τους, η LEGO έχει αρχίσει ένα σχέδιο εξέτασης για κάθε αισθητήρα που ως επιχείρηση αναπτύσσει τις νέες ψηφιακές συσκευές ή εγκρίνουν τις συσκευές τρίτων. Αυτήν την περίοδο, ο εφαρμόσιμος κατάλογος περιλαμβάνει μόνο τον υπερηχητικό αισθητήρα (Ultrasonic sensor), ο οποίος δίνει διεύθυνση 1 (μέσα σε ένα 7 bit πλαίσιο). Αυτή η διεύθυνση στέλνεται μαζί με τη κατεύθυνσης επικοινωνίας σε bit, και όπως με όλα τα υπόλοιπα data byte, η διεύθυνση μεταφέρεται με το σημαντικότερο bit πρώτα.

2.10) Επαναφορτιζόμενη μπαταρία

Η επαναφορτιζόμενη μπαταρία δεν περιλαμβάνεται στον κανονικό set αγοράς του Nxt αλλά είναι ένα επιπλέον αξεσουάρ που το αγοράζουμε εξτρά. Η επαναφορτιζόμενη έχει μια υποδοχή στην πλευρά για να συνδέσουμε τον φορτιστή, ο οποίος μπορεί βολικά να επαναφορτίσει την μπαταρία χωρίς να αναγκαστούμε να την αφαιρέσουμε από το NXT. Έναντι της αρχικής κάλυψης μπαταριών, το πάχος της είναι ελαφρώς μεγαλύτερο, και όταν συνδέεται με το NXT, προεξέχει στο κατώτατο σημείο ακόμα έχει δυο lads(κόκκινο και πράσινο) τα οποία μας δείχνουν το στάδιο φόρτισης της μπαταριάς με το κόκκινο η μπαταρία είναι αφόρτιστη και μόλις φόρτιση γίνει πλήρη ανάβει το πράσινο. Με τις κατάλληλες βιβλιοθήκες των γλωσσών προγραμματισμού (java, C#, VB.Net) μας δίνεται η δυνατότητα να ελέγχουμε την τάση της μπαταριάς σε real time .



Σχημα-17 επαναφορτιζόμενη μπαταρία

Η διαχείριση της τάσης μέσα στο Nxt αποτελείτε από 5 volt ανεφοδιασμού τα οποία παράγονται από την μπαταρία και τα υπόλοιπα 3,3volt χρησιμοποιούνται για την λειτουργία του επεξεργαστή ARM7 και του BlueCore chip για την Bluetooth επικοινωνία. Για να προστατεύσει την παροχή ηλεκτρικού ρεύματος μέσα στο NXT, βρίσκεται ένας διακόπτης που συνδέεται στην αρχή του κυκλώματος τάσης. Ο διακόπτης έχει ένα ρεύμα 1.85 A και θα διεγερθεί σε περίπου στα 3.3 A. παρακάτω στο σχημα-18 θα παρατηρήσουμε την τρέχουσα κατανάλωση ρεύματος για τάση μπαταριών 9volt.

Supply voltage	Current		Effect (Battery = 9 Volts)	
	Max [mA]	Normal [mA]	Max [mW]	Normal [mW]
No load on motors				
9 Volt	339	114	5184	1422
5 Volt	271	112	1744	448
3.3 Volt	72	38	410	216
Load on motors				
9 Volt	2901	848	26109	7632
5 Volt	271	112	1142	307
3.3 Volt	72	38	410	137
Standby	46 uA assumed standby current due to brown out detection			

Σχημα-18 παρουσίαση των τρέχων τιμών του ρεύματος στο Nxt

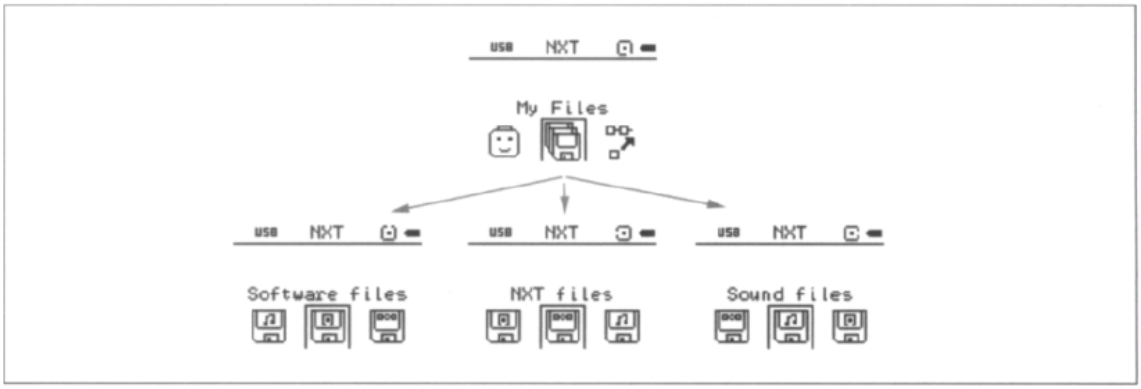
2.11) Το σύστημα αρχείων του Lego Mindstorms NXT (file system)

Το NXT έχει εισαγάγει ένα σύστημα αρχείων που αποθηκεύει μέχρι 64 αρχεία. Το Firmware επιτρέπει στο χρήστη να δημιουργήσουν και διαγράψουν τα αρχεία, να τα μετονομάσει, και τροποποιήσει το περιεχόμενό τους. Τα ονόματα αρχείων χωρίζουν μια επέκταση τριών χαρακτήρων, από το όνομα αρχείου μέχρι μια περίοδο. Το ίδιο το όνομα μπορεί να είναι μέχρι 15 χαρακτήρες πολύ. Οι επεκτάσεις αρχίζουν με το r, και από αυτήν την σύμβαση, τα εκτελέσιμα αρχεία σχετικά με το NXT έχουν μια επέκταση .rxe. Το σχημα-19 απαριθμεί τους αναγνωρισμένους τύπους αρχείων με τις επεκτάσεις τους.

File Type	Extension(s)
Data files	.rdt
Executable files and Try Me programs	.rxe, .rtm
Icon files	.ric
Hidden menu files	.rms
Program files	.rpg
Sound files	.rso
Hidden system files	.sys
Temporary hidden files	.tmp

Σχημα-19 Απεικόνιση επεκτάσεων αρχείων του Nxt

Ακόμα το Nxt παρουσιάζει αυτά τα αρχεία υπό μορφή έναρξης δομών επιλογών με " My files" και έπειτα σε " Software files", " NXT files", "Sound files" και ούτω καθεξής (δείτε το σχήμα-20). Μετά από κάθε επιλογή που παρουσιάσει τα αρχεία αποθηκεύονται στη μνήμη NXT. Η οθόνη με το Menu φαίνετε στο σχήμα-20 .

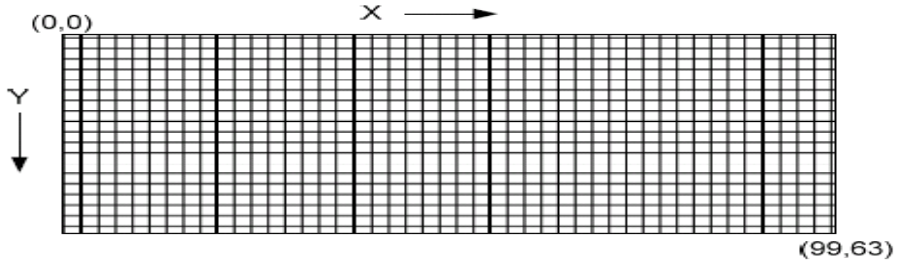


Σχημα-20. Menu επίλογων Nxt

2.12) Οθόνη απεικόνισης

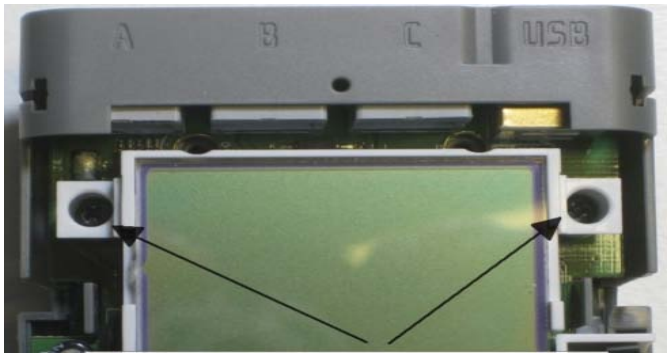
Μια matrix οθόνη έχει τοποθετηθεί στο Nxt για να βελτίωση την ενδιάμεση επικοινωνία του χρήστη με το περιβάλλον του. Η οθόνη απεικόνισης αποτελείτε από μια LCD με ασπρόμαυρα γραφικά με διαστάσεις 100 x 64 pixels. Η οθόνη έχει περιοχή απεικόνισης 26 x 40.6 mm. Ο LCD-ελεγκτής που χρησιμοποιείται για να ελέγξει την επίδειξη είναι ένα UltraChip 1601.

Υπάρχει ένα περιβάλλον επικοινωνίας SPI από το ARM7 μικροελεγκτή στον LCD-ελεγκτής UltraChip 1601. Το περιβάλλον επικοινωνίας SPI τρέχει σε 2 MHZ μέσα τυποποιημένο firmware LEGO® και δύο πλήρεις χάρτες μνήμης τίθενται κατά μέρος μέσα firmware για την ενημέρωση της οθόνης. Η οθόνη ενημερώνεται συνεχώς σε μια ακολουθία γραμμών ανά 17ms για μια συνολική ανανέωση απεικόνισης.



Τα pixels μέσα στην οθόνη LCD

Η πληροφορία της LCD διατίθεται μέσα στη μνήμη ως δισδιάστατη σειρά, κανονική [8] [100] (κανονικά [Y/8] [X]). Το στοιχείο στέλνεται στον ελεγκτή LCD στην ακόλουθη μορφή: το πρώτο byte ελέγχει την πρώτα 8-pixel κατακόρυφα (που αρχίζει (0.0)) και το δεύτερο byte ελέγχει τα επόμενα 8-pixel οριζόντια.



Στο παραπάνω σχήμα παρουσιάζουμε την LCD οθόνη του Nxt.

Ο παρακάτω πίνακας παρουσιάζει τα τεχνικά χαρακτηριστικά της LCD οθόνης.

Format	100 x 64 dots
LCD mode	STN / Positive Reflective Mode / Gray
Viewing direction	6 o'clock
Driving scheme	1/65 duty cycle, 1/9 bias
Power supply voltage (VDD)	3.0V
LCD driving voltage (VLCD)	9.0V (for best contrast)

2.13) Αναπαραγωγή ήχου

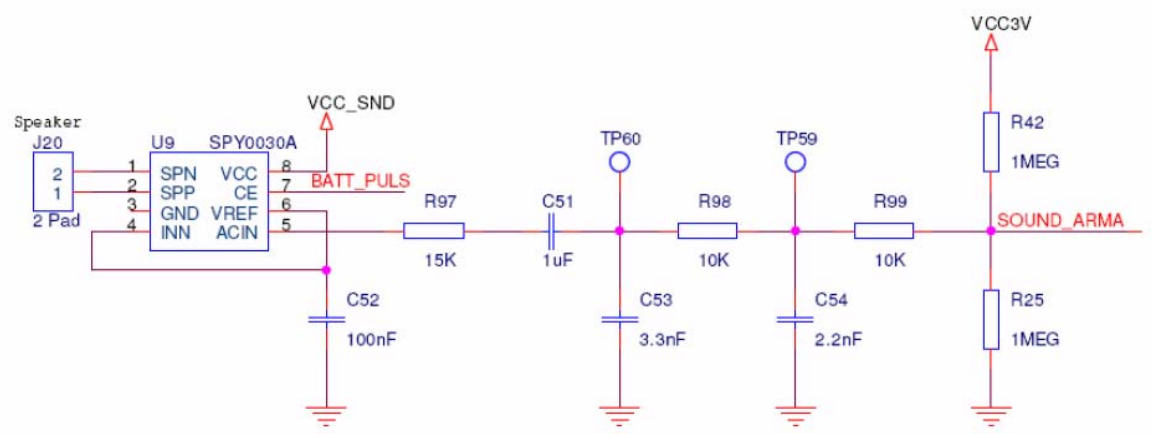
Το NXT περιλαμβάνει ένα τσιπ ενισχυτών για να βελτιώσει το επίπεδο ήχου και την ποιότητα παραγωγής. Ο ήχος στην έξοδο είναι ένα σήμα εξόδου μιας παλμογενητριας PWM που ελέγχεται από το ARM7 μικροελεγκτή. Τα φίλτρα που εισάγονται πριν από τον ενισχυτή θα μειώσουν το θόρυβο πέρα από τη δειγματοληψία στο σήμα.

Ο driver ήχου (SPY0030A) είναι ένα διαφορικό τσιπ ήχου ενισχυτών από SUNPLUS που μπορεί να έχει ένα μέγιστο κέρδος 20. Το Nxt έχει ένα μεγάφωνο με μια διάμετρο 21 χιλ. Ο

πίνακας παρουσιάζει κατωτέρω το ρεύμα και κατανάλωση ισχύος όταν οι ήχοι παίζονται σε δύο διαφορετικές συχνότητες.

Τρέχουσα κατανάλωση ρεύματος ήχου

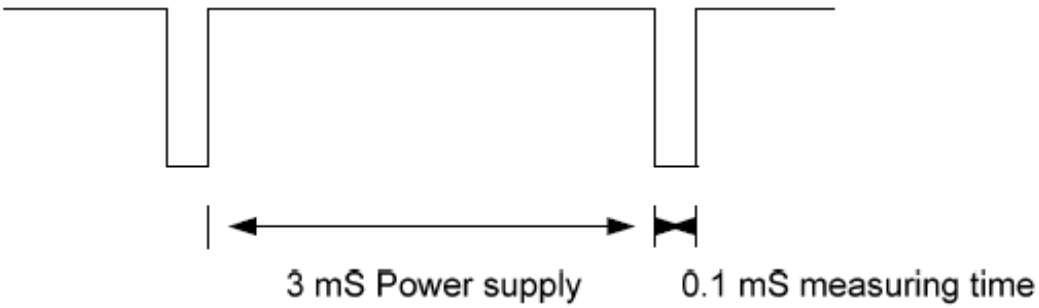
Συχνότητα	Ρεύμα mA	Ισχύς mW
440 Hz	102	169
4 KHz	78	97



Σχηματικό κυκλώματος ήχου εξόδου μέσα στο Nxt

2.14) Αισθητήρες

Για να εξασφαλίσει τη συμβατότητα με τους ρομποτικούς αισθητήρες συστημάτων εφευρέσεων LEGO MINDSTORMS (που αναπτύσσονται για το ευφυές τούβλο RCX), μια γεννήτρια ρεύματος έχει προστεθεί στο τούβλο NXT στα σωστά διαστήματα δύναμης και μέτρησης παραγωγής για αυτούς τους παλαιότερους αισθητήρες. Μαζί με τυποποιημένο firmware LEGO το ρεύμα που παράγετε από τη γεννήτρια κάνει την ίδια λειτουργία όπως είναι αυτή διαθέσιμη μέσα στο ευφυές τούβλο RCX. Το ρεύμα της γεννήτριας παρέχει περίπου 18 mA του ρεύματος εξόδου. Η γεννήτρια ελέγχει την δύναμη που πηγαίνει στους ενεργούς αισθητήρες. Παρέχει στον αισθητήρα τη ισχύ για 3ms και έπειτα την αναλογική τιμή που η διάρκεια της ακλουθεί για 0.1 ms. (βλέπε σχημα-21).



Σχήμα-21. Χρονικό διάγραμμα για το A/D σήμα εισόδου μόλις χρησιμοποιούμε αισθητήρες στην είσοδο.

Τους αισθητήρες τους χωρίζουμε σε δυο κατηγορίες τους παθητικούς (Passive sensors) όπου δεν χρειάζονται το πρόσθετο συγχρονισμό προαναφέραμε. Αυτοί οι αισθητήρες επιλέγονται επίσης κάθε 3ms επειδή η δειγματοληψία που χρησιμοποιεί το μετατροπέα A/D είναι ταυτόχρονη και επομένως, πρέπει να υποστηρίξει το συγχρονισμό που απαιτείται από τους αισθητήρες.

Στους αισθητήρες αυτούς συγκαταλέγονται ο ακόλουθοι :

- Touch sensor (both the RCX and NXT versions)
- Light sensor
- Sound sensor
- Temperature sensor

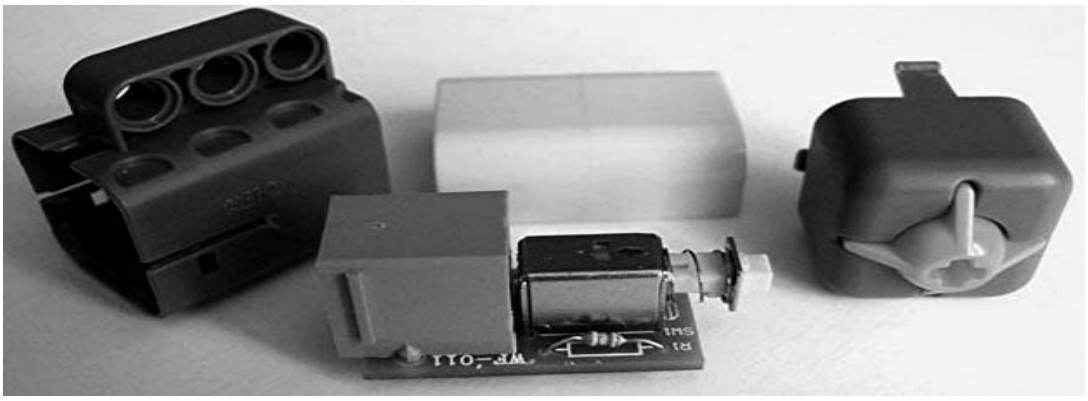
Η δεύτερη κατηγορία αισθητηρίων (sensors) είναι οι ψηφιακοί σε αυτούς συγκαταλέγονται όλοι οι αισθητήρες που χρησιμοποιούν το πρωτόκολλο επικοινωνίας I^2C και περιλαμβάνουν ένα μικροελεγκτή που επεξεργάζεται τις διάφορες καταστάσεις από το φυσικό περιβάλλον.

Σε αυτή την κατηγορία περιλαμβάνεται ο αισθητήρας υπερήχων (Ultrasonic sensor).

2.14.i) Αισθητήρας αφής (Touch sensor)

Ο αισθητήρας αφής NXT παρουσιάζεται στο σχήμα-22. Ένα χαρακτηριστικό γνώρισμα του αισθητήρα αφής είναι η διαγώνια τρύπα του που επιτρέπει σε μας να τον συνδέσουμε άμεσα με άλλα τουβλάκια με σκοπό να τον προσαρμόσουμε να επιτελεί

διάφορες λειτουργίες. Εσωτερικά ο αισθητήρας αφής είναι ένα τυπωμένο κύκλωμα με ένα κουμπί ώθησης τοποθετημένο και ένα ελατήριο, όπως βλέπετε στο σχήμα-24.



Σχήμα -24.Τα εσωτερικά μέρη του αισθητήρας Αφής

Σχήμα-22. Αισθητήρας Αφής

Υπάρχει επίσης ένας αντιστάτης σε σειρά με το διακόπτη έτσι δεν θα δημιουργήσει κανένα βραχυκύκλωμα εάν συνδεθεί τυχαία με κάτι στην πόρτα εξόδου. Με λίγα λόγια Ο Αισθητήρας Αφής δίνει στο ρομπότ την αίσθηση της αφής, ανιχνεύει όταν αυτός πιέζεται από κάτι άλλο και όταν απελευθερωθεί όπως φαίνετε στο σχήμα -23

Σχήμα-23. Αρχή λειτουργιάς αισθητήρας Αφής

Μπορούμε να χρησιμοποιήσουμε τον Αισθητήρας Αφής για να κάνουμε το ρομπότ να συλλέξει πράγματα: ένας ρομποτικός βραχίονας εφοδιασμένος με Αισθητήρας Αφής κάνει το ρομπότ να γνωρίζει αν υπάρχει ή δεν υπάρχει κάτι στο χέρι για να το αρπάξει. Ή μπορούμε να χρησιμοποιήσουμε ένα Αισθητήρα Αφής στο ρομπότ μας για να αποφασίσει σχετικά με μια εντολή. Για παράδειγμα, πιέζοντας τον Αισθητήρα Αφής μπορούμε να κάνουμε το ρομπότ να περπατά, να μιλά, να κλείνει μία πόρτα, ή να ανοίγει την τηλεόρασής μας.

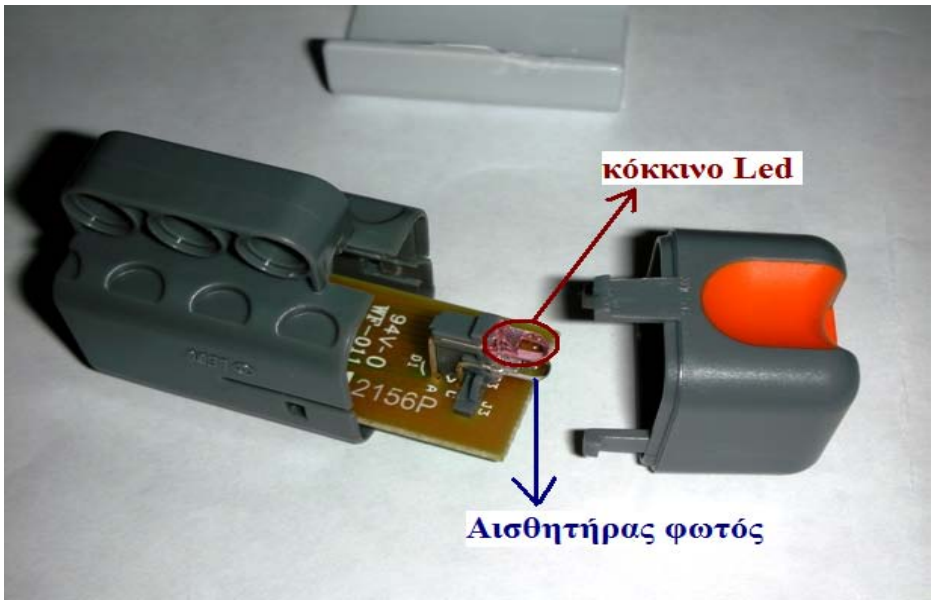
2.14.ii) Αισθητήρας Φωτός (Light Sensor)

Ο Αισθητήρας Φωτός (βλέπε σχήμα-25) είναι ένας από τους δύο αισθητήρες που δίνουν όραση στο ρομπότ σας (ο Αισθητήρας Υπερήχων είναι άλλος). Ο Αισθητήρας Φωτός επιτρέπει στο ρομπότ μας να διακρίνει μεταξύ φωτός και σκοταδιού. Μπορεί να διαβάσει την ένταση του φωτός σε ένα δωμάτιο και να μετρήσει την ένταση φωτός των χρωματισμένων επιφανειών.

Η σημαντικότερη διαφορά μεταξύ του αισθητήρα αφής και του αισθητήρα φωτός είναι ότι ο πρώτος παίρνει τιμές on και off ενώ ο δεύτερος επιστρέφει πολλές πιθανές τιμές. Αυτές οι τιμές εξαρτώνται από την ένταση του φωτός που χτύπα τον αισθητήρα την στιγμή που δέχεται τις τιμές. Οι τιμές στην γλώσσα προγραμματισμού VB.net και σύμφωνα με την βιβλιοθήκη που έχουμε εισάγει είναι από 0 έως 100. Δηλαδή κατά την ανάγνωση του αισθητήρα το περισσότερο φως μας αποδίδει και περισσότερο ποσοστό τιμών.

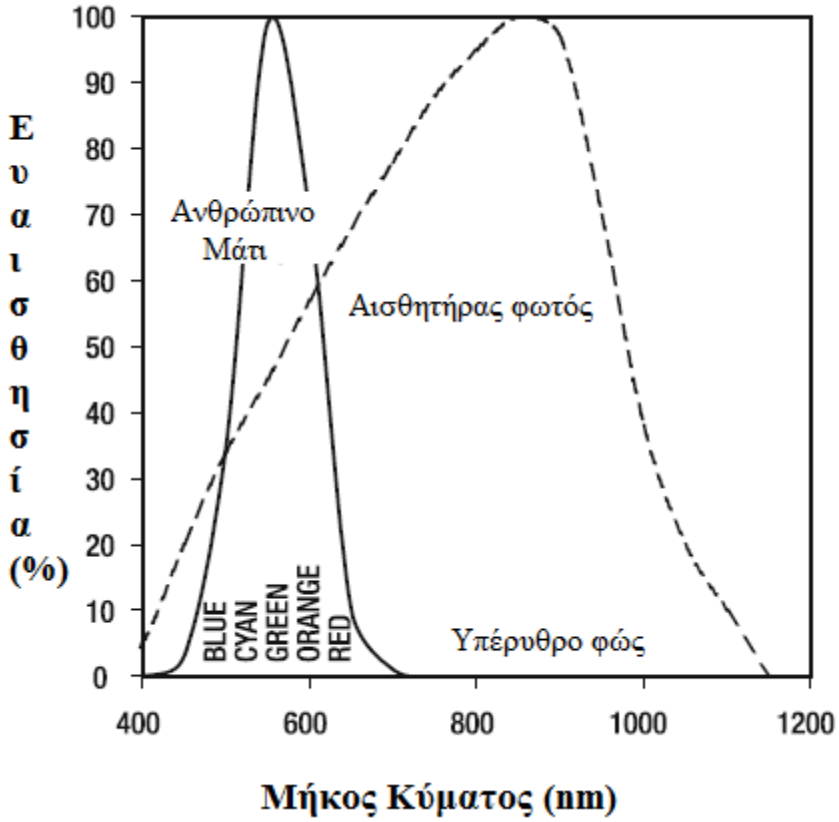
Σχήμα-25. Αισθητήρας φωτός

Πρέπει να δώσουμε προσοχή στο γεγονός ότι ο αισθητήρας δεν βλέπει χρώμα άλλα μπορεί μόνο να δει τις ποικίλες εντάσεις τιμών φωτός από ένα σύνολο συγκεκριμένων τιμών φωτός που μπορεί να λάβει ουσιαστικά ένα εύρη φάσμα μαύρης και άσπρης διακύμανσης. Εάν βάλουμε να διαβάσει χρώμα μαύρο και άσπρο θα μας επιστρέψει δυο διαφορετικές τιμές, αυτό γιατί το ποσοστό φωτός που θα επιστρέψει από το μαύρο θα είναι λιγότερο έντονο από το ποσοστό που θα επιστρέψει από το άσπρο χρώμα. Υπάρχει ένα άλλο χαρακτηριστικό γνώρισμα αυτής της συσκευής: Όχι μόνο ανιχνεύει το φως, αλλά εκπέμπει κάποιο φως επίσης. Ένα κόκκινο LED παρέχει μια σταθερή πηγή φωτός, επιτρέποντας σε μας να μετρήσουμε το απεικονισμένο φως που έρχεται πίσω στον αισθητήρα. (Βλέπε σχήμα-26. Το εσωτερικό του αισθητήρα).



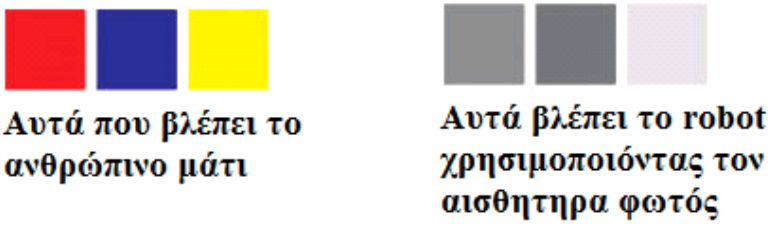
Σχήμα-26. Το εσωτερικό του αισθητήρα φωτός

Η νεώτερη έκδοση NXT έχει δει μερικές βελτιώσεις ως προς τον τρόπο που είναι δομημένα το led και ο αισθητήρας φωτός όπου υπάρχει μια πλαστική προέκταση που χωρίζει τα δύο. Όταν θέλουμε να μετρήσουμε το απεικονισμένο φως, πρέπει να είμαστε προσεκτικοί για να αποφύγουμε οποιαδήποτε πιθανή παρέμβαση από άλλες πηγές. Το ποσό φωτός που απεικονίζεται από μια επιφάνεια εξαρτάται από πολλούς παράγοντες κυρίως το χρώμα, η σύσταση, και η από φαση το υ από την πηγή. Ένα μαύρο αντικείμενο απεικονίζετε με το λιγότερο φως από ένα άσπρο, ενώ μια μαύρη επιφάνεια μεταλλική απεικονίζετε με το λιγότερο φως από μια μαύρη γυαλιστερή επιφάνεια. Συν, όσο μεγαλύτερη η απόσταση των αντικειμένων από τον αισθητήρα, τόσο λιγότερο φως επιστρέφεται στον αισθητήρα-ανιχνευτή φωτός. Αυτοί οι παράγοντες είναι αλληλοεξαρτώμενοι. Αλλά εάν κρατάμε όλους τους παράγοντες σταθερούς θα είμαστε ικανοί να συναγάγουμε πολλά πράγματα από τις αναγνώσεις.



Σχήμα- 27.Ευαισθησία αισθητήρα φωτός

Το φωτοτρανζίστορ στον αισθητήρα είναι πολύ πιο ευαίσθητο στα υπέρυθρα χρώματα του φωτός από το σχετικά στενό ορατό φάσμα βλέπουμε. Αυτό επειδή βλέπει θερμές πηγές φωτός όπως οι πυρακτωμένες λάμπες φωτός .Μας δειχνει πώς το φασμα του φωτοτραντζιστορ καλύπτει το ανθρώπινο μάτι.



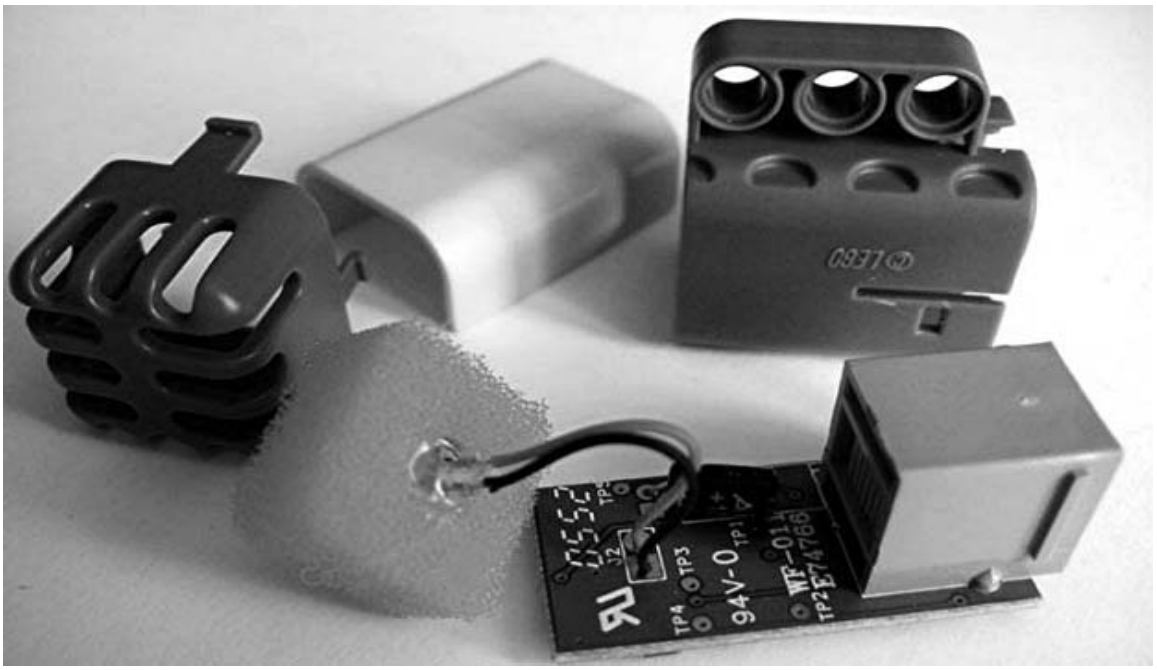
Σχημα-28. Αντιστοιχία ανθρώπινου ματιού και αισθητήρα φωτός.

2.14.iii) Αισθητήρας Ήχου (Sound Sensor)

Ο Αισθητήρας Ήχου κάνει ρομπότ μας να ακούσει! Ο Αισθητήρας Ήχου μπορεί να ανιχνεύσει τόσο ντεσιμπέλ [dB] όσο και προσαρμοσμένα ντεσιμπέλ [dBA]. Ένα ντεσιμπέλ είναι μια μονάδα μέτρησης της ηχητικής πίεσης dBA: για ανίχνευση προσαρμοσμένων ντεσιμπέλ, η ευαισθησία του αισθητήρα είναι προσαρμοσμένη στην ευαισθησία του ανθρώπινου αυτιού. Με άλλα λόγια, αυτοί είναι οι ήχοι που τα αυτιά σας μπορούν να ακούσουν. dB: για ανίχνευση πρότυπων [αδιόρθωτων] ντεσιμπέλ, όλοι οι ήχοι μετρούνται με την ίδια ευαισθησία. Έτσι, οι ήχοι μπορεί να περιλαμβάνουν ορισμένα κομμάτια που είναι πάρα πολύ υψηλά ή πολύ χαμηλά για να τα ακούσει το ανθρώπινο αυτί. Επειδή δέχεται στην είσοδο του φυσικές διεγέρσεις, ο αισθητήρας ήχου ήταν ένας από τους πρώτους αισθητήρες που είχε ευρύ χρήση. Ο αισθητήρας ήχου NXT (δείτε το σχήμα 29) κατασκευάζεται σαν τον αισθητήρα φωτός.

Σχήμα-29. Αισθητήρας Ήχου

Στο σχήμα-30 μπορούμε να δούμε το μικρόφωνο που τοποθετείται στον πλαστικό, έναν πυκνωτή, και το συνδετήρα (κλιψάκι) στην κορυφή τις πλακέτας.



Σχήμα-30 Το εσωτερικό του Αισθητήρα Ήχου

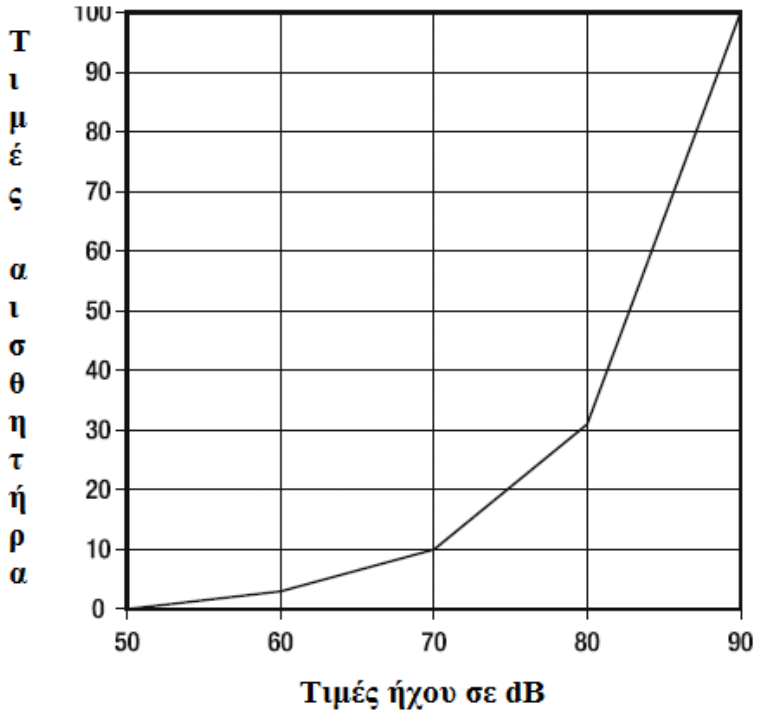
Ο Αισθητήρας Ήχου μπορεί να μέτρα στάθμη ηχητικής πίεσης έως 90 dB - το επίπεδο μιας χορτοκοπτικής μηχανής. Οι στάθμες ηχητικής πίεσης είναι εξαιρετικά περίπλοκες, έτσι η τιμή του Αισθητήρα Ήχου στο NXT εμφανίζεται σε ποσοστό [%]. Όσο χαμηλότερο είναι το ποσοστό, τόσο πιο χαμηλής έντασης είναι ο ήχος, για παράδειγμα:

- 4-5% είναι όπως ένα σιωπηλό σαλόνι
- 5-10% κάποιος που μιλάει από κάποια απόσταση
- 10-30% είναι μια φυσιολογική συνομιλία κοντά στην αισθητήρα ή μουσική σε κανονική ένταση
- 30-100% είναι άτομα που φωνάζουν ή μουσική που παίζεται σε μεγάλη ένταση

Τις παραπάνω τιμές μπορούμε να τις δούμε στον παρακάτω πίνακα υπό την μορφή υπολογισμού σε dB (decibels). Σε αυτή την περίπτωση 0dB είναι ο πιο εξασθενημένος ήχος που ένας μέσος άνθρωπος μπορεί να ακούσει. Στον παρακάτω πίνακα παραθέτουμε τις τιμές σε dB που αντιστοιχούν σε ποσότητες θορύβου (ήχου).

Τιμές σε dB	Πηγή δημιουργίας
90	Ηχηρός Θόρυβος
80	Άτομα που φωνάζουν
70	Ομιλία
60	Μέσα σε ένα γραφείο όπου υπάρχει απόσταση μεταξύ των ατόμων που ομιλούνε
50	Ένα ήσυχο δωμάτιο

Στο παρακάτω σχήμα-31 θα παραστήσουμε την ποσότητα του ήχου σε dB σε σχέση με τις τιμές που λαμβάνει ο αισθητήρας ήχου.



Σχήμα-31.Τιμές Ευαισθησία αισθητήρα ήχου

2.14.iv) Αισθητήρας Υπέρηχων (απόστασης) – Ultrasonic Sensor

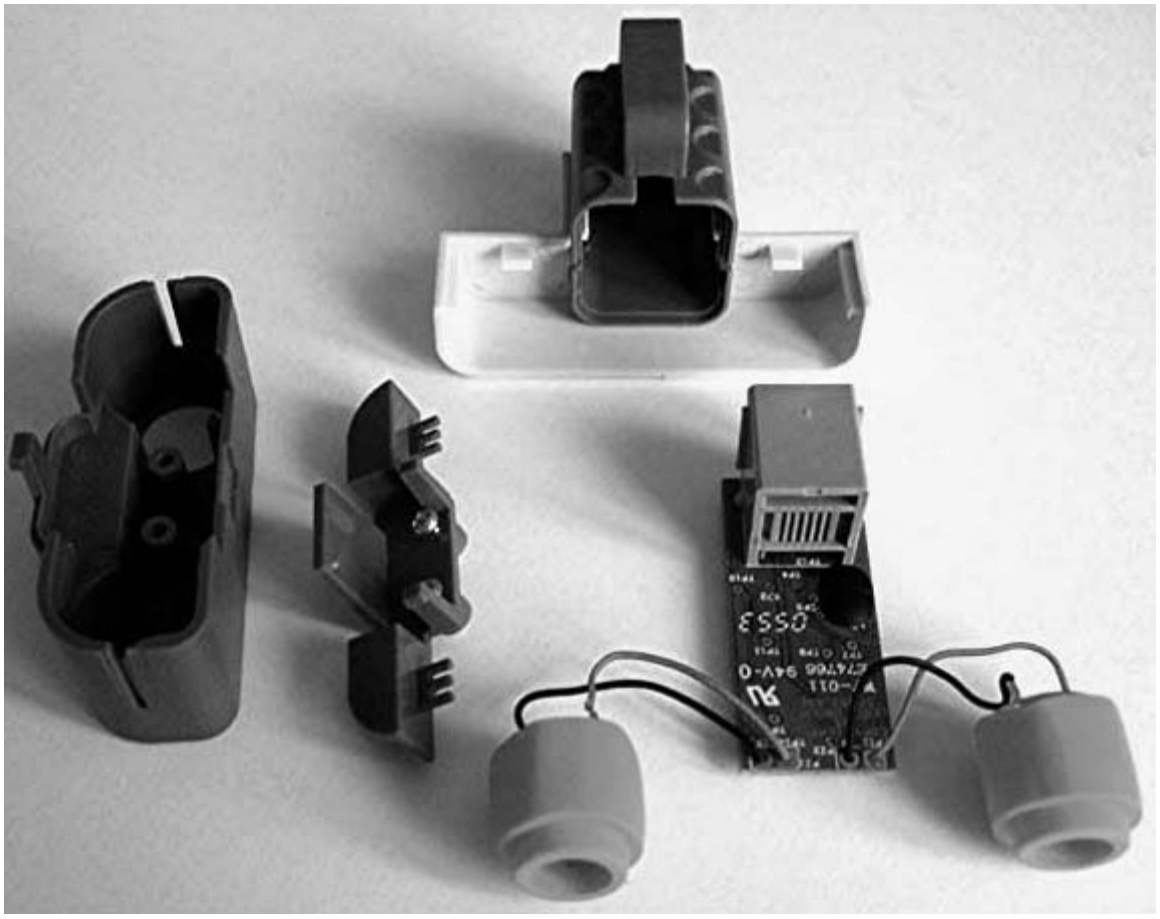
Ο Αισθητήρας Υπέρηχων είναι ένας από τους δύο αισθητήρες που δίνουν σας ρομπότ όραση [Ο Αισθητήρας Φωτός είναι ο άλλος]. ο Αισθητήρας Υπέρηχων (βλέπε σχήμα-32) επιτρέπει στο ρομπότ μας να βλέπει και να εντοπίζει αντικείμενα. Μπορούμε επίσης να το χρησιμοποιήσουμε για να κάνει το ρομπότ μας να αποφύγει τα εμπόδια, να καταλαβαίνει και να μετρά απόσταση, και να ανιχνεύει κίνηση. Ο Αισθητήρας Υπέρηχων μέτρα απόσταση σε εκατοστά και σε ίντσες. Είναι σε θέση να μετρά τις αποστάσεις από 0 έως 255 εκατοστά με ακρίβεια +/- 3 εκατοστών.

Σχήμα-32. Αισθητήρας Υπέρηχων

Η αρχή πίσω από την υπερηχητική ανίχνευση είναι ότι ο αισθητήρας εκπέμπει τον υψηλής συχνότητας υπέρηχο (πέρα από το όριο της ανθρώπινης ακρόασης), ο οποίος αναπηδά πίσω από τα αντικείμενα και διαβάζεται από τον αισθητήρα. Ο χρόνος που κάθε παλμός παίρνει στην αναπήδηση πίσω στον αισθητήρα καθορίζει την απόσταση από το αντικείμενο. Όσο πιο μακροχρόνιο το διάστημα, τόσο το αντικείμενο είναι πιο μακριά, και αντίστροφα. Οι νυχτερίδες χρησιμοποιούν την ίδια αρχή ως μέσο πλοήγησης και της εντόπισης του θηράματός τους. Ένας κοινός τρόπος να προγραμματιστεί ο αισθητήρας είναι να του δώσουμε κάποιες τιμές και να δούμε ποιες θα επιτρέψει στο ρομπότ μας. Οι

τιμές επιστροφής μεταξύ 0 και 100. Τότε η βιβλιοθήκη της Java μας επιτρέπει να τεθούν αριθμοί αυτοί σε ίντσες ή σε εκατοστόμετρα.

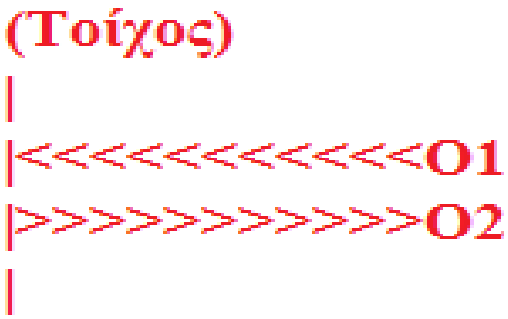
Το εσωτερικό του αισθητήρα Υπέρηχων παρουσιάζεται στο Σχήμα-33 παρατηρούμε τον εκπομπό και τον δέκτη των σημάτων ηχούς.



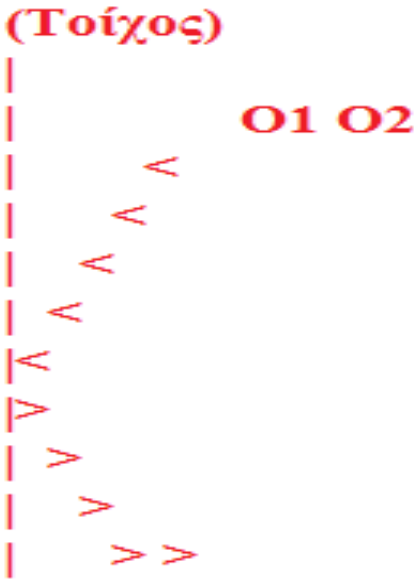
Σχήμα-33 Τα εσωτερικά μέρη του αισθητήρα Υπέρηχων

Ο αισθητήρας στέλνει το υπέρηχο σήμα του σε συχνότητα των 40KHz και μέτρα την απόσταση από τον υπολογισμό του χρόνου που χρειάζεται ένα ηχητικό κύμα για να χτυπήσει ένα αντικείμενο και να επιστρέψει πίσω - όπως η ηχώ. Αντικείμενα μεγάλου μεγέθους με σκληρές επιφάνειες επιστρέφουν τις καλύτερες αναγνώσεις. Αντικείμενα από μαλακό ύφασμα ή ότι είναι κυρτό [σαν μια μπάλα] ή είναι πολύ λεπτό ή μικρό μπορεί να είναι δύσκολο για τον αισθητήρα να τα ανιχνεύσει.

Πιο απλά ο αισθητήρας στέλνει και λαμβάνει pings και τα συμβολίζουμε όπως θα δούμε και στα παρακάτω σχήματα **O1**(παλμός εξόδου), **O2**(παλμός εισόδου) έστω ότι **μπροστά** από τον αισθητήρα υπάρχει ένας τοίχος :



Άρα ο **O2**(παλμός εισόδου) αναπηδά και φτάνει πάλι πίσω στον αισθητήρα οπότε λαμβάνουμε ικανοποιητικά την απόσταση του τοίχου.



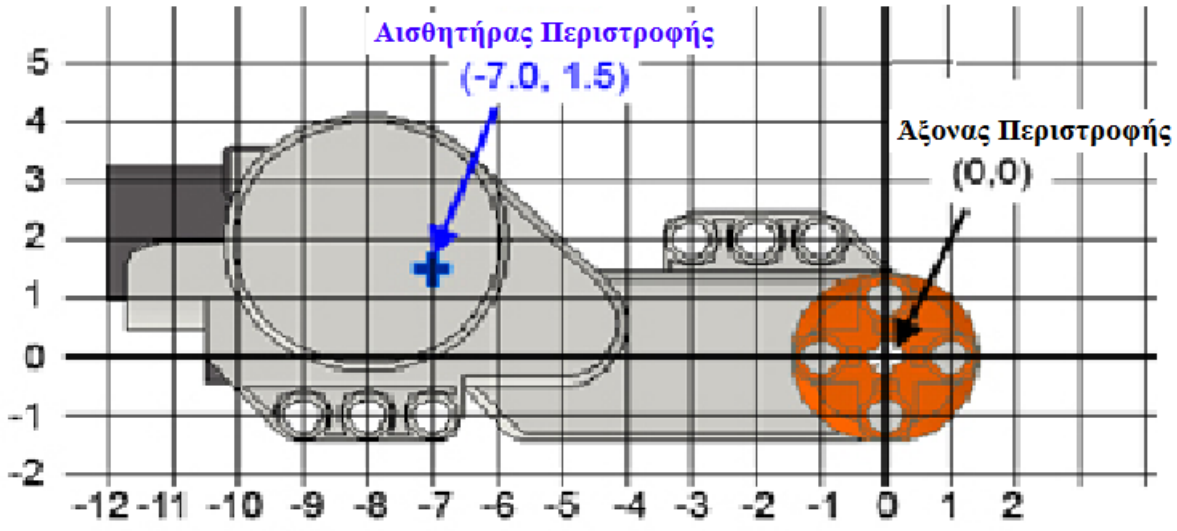
Έστω τώρα στην περίπτωση που ο αισθητήρας του robot μας κοιτάζει το αντικείμενο υπό γωνιά δείτε τη επιπτώσεις έχουμε στο διπλανό σχήμα .

Παρατηρούμε οι παλμός **O1**(παλμός εξόδου) προσπίπτει στο αντικείμενο αναπηδά αλλά δεν καταλήγει πίσω στο ν **O2**(παλμός εισόδου) όποτε η απόσταση- θέση του αντικειμένου δεν μπορεί να γίνει αντιληπτεί από τον αισθητήρα.

Τέλος, δύο ή περισσότεροι αισθητήρες υπερήχων που λειτουργούν στον ίδιο χώρο μπορεί να διακόπτουν ο ένας τις αναγνώσεις του άλλου.

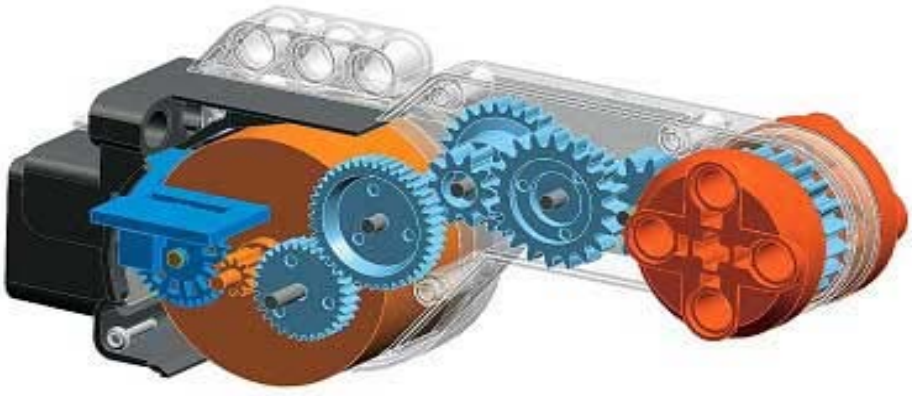
2.15) Σερβοκινητήρας Lego Mindstrom NXT– Rotation Sensor

Οι σερβομηχανές στις βιομηχανικές εφαρμογές είναι διαφορετικές από τις κανονικές μηχανές λόγω της ικανότητάς τους να περιστραφούν ακριβώς τον άξονα μηχανών. Αυτό επιτυγχάνεται από τη πρόσθετη ηλεκτρονική που χτίζεται στις μηχανές. Ομοίως, οι σερβομηχανές NXT (Βλέπε σχήμα-35) είναι προηγμένες στις ικανότητες και την ακρίβειά τους. Οι σερβομηχανές NXT έχουν έναν ενσωματωμένο οπτικό κωδικοποιητή που κρατά την αρίθμηση των περιστροφών άξονας μηχανών. Αυτός ο κωδικοποιητής είναι ακριβής μέχρι 1 μοίρα περιστροφής μηχανών. Μπορούμε να χρησιμοποιήσουμε αυτήν την ιδιότητα από το πρόγραμμά μας για την ακριβή μετακίνηση ή τον προσδιορισμό θέσης. Ακόμα μπορεί επίσης να κρατήσει δύο μηχανές συγχρονισμένες η μια με την άλλη και να κινήσει το ρομπότ μας σε μια ευθεία γραμμή.



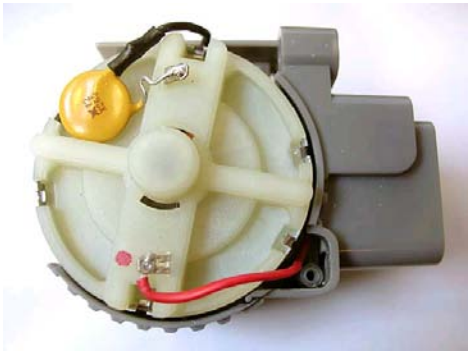
Σχήμα-35 Το Μοτέρ του Nxt

Παραδείγματος χάριν αν δώσουμε την εντολή στο ρομπότ μας να προχωρήσει ευθεία που σημαίνει ότι θα δώσουμε την ίδια τιμή ταχύτητας στα μοτέρ και κατά την διάρκεια που θα κινείται το ρομπότ μας θα κινείτε προς τα εμπρός εάν κρατήσουμε με το χέρι μας το ένα μοτέρ τότε θα παρατηρήσουμε ότι ο αισθητήρας κίνησης θα επιβραδύνει και το άλλο μοτέρ για να κινηθεί ευθεία το ρομπότ.



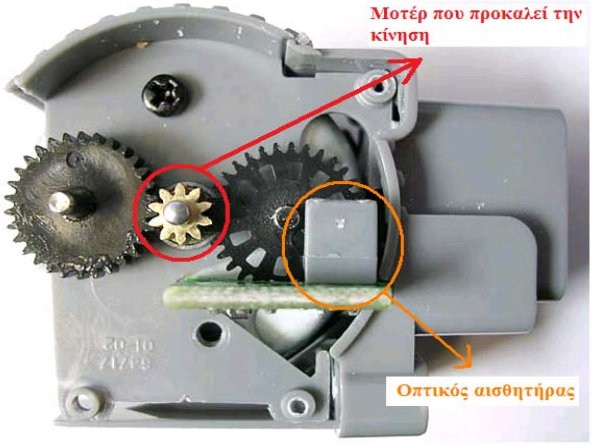
Σχήμα-36. Τα γρανάζια του Nxt Moter

Το σχήμα-36 παρουσιάζει την εσωτερική όψη με τα γρανάζια της με τον αισθητήρα κίνησης (με μπλε) τοποθετημένος αριστερά του μεγαλύτερου πορτοκαλιού τυμπάνου (η μηχανή). Στην πραγματικότητα, ο αισθητήρας κίνησης αποτελείται από μια μαύρη ρόδα που περιέχει 12 τρύπες που επιτρέπουν στον οπτικό αισθητήρα να διαβάσει 24 on/off καταστάσεις με κάθε πλήρη περιστροφή. Αυτό παρέχει στο NXT πολύ καλή δυνατότητα για να ανιχνεύσει τη θέση κάτω σε συνδυασμό με τις μοίρες. Από το σχήμα-36, μπορούμε επίσης να δούμε πώς η μηχανή NXT συνδέεται εσωτερικά. Υπάρχει αρκετή ροπή στις ρόδες κίνησης και τις κυκλικές διαδρομές ενδιάμεσα.

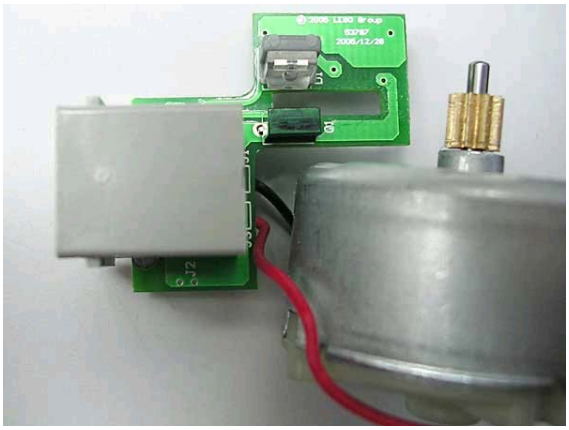


Εφόσον αφαιρέσουμε όλες τις βίδες θα βρεθούμε αντιμέτωποι με την διπλανή εικόνα. Το μοτέρ τροφοδοτείτε με 9 Volt DC τάση.

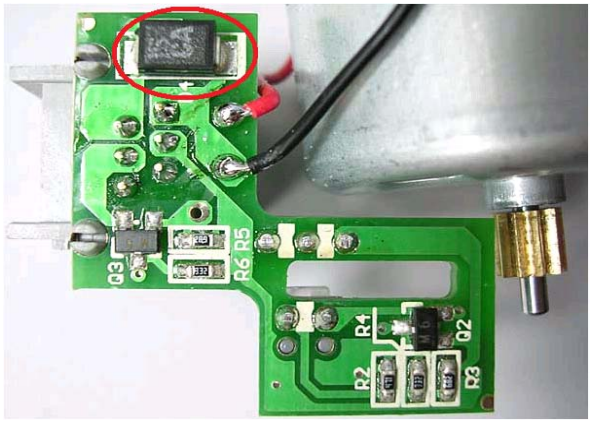
Το κίτρινο εξάρτημα είναι ένα Thermistor τύπου (RXE065 ή MF-R065) που προστατεύει το μοτέρ σε περιπτώσει υψηλού ρεύματος.



Από την άλλη πλευρά παρατηρούμε το μοτέρ που προκαλεί την κίνηση και τον οπτικό αισθητήρα που επιτρέπει την συγκεκριμένη λειτουργιά του μοτέρ.

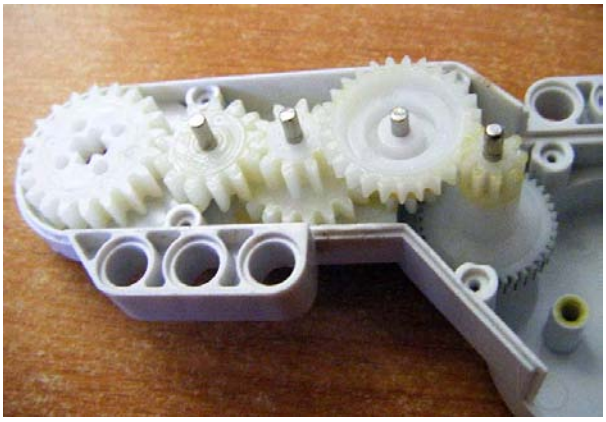


Παρατηρούμε το μοτέρ και το τυπωμένο κύκλωμα που συνδέεται το μοτέρ με το καλώδιο RJ12 και τον οπτικό αισθητήρα, που μας θυμίζει τον ίδιο αισθητήρα που είχαν μέσα τα πιο παλιά ποντίκια υπολογιστών και βεβαίος είναι η ίδια λειτουργία.



Η «μικρή» χάραξη που υπάρχει είναι για την είσοδο του τροχού και τον έλεγχο του από τον οπτικό αισθητήρα. Το μαύρο ορθογώνιο εξάρτημα που είναι κυκλωμένο με κόκκινο χρώμα είναι ένας καταστολέας υπέρτασης όπου κρατεί την τάση στα 15volt maximum.

Ο άξονας περιστροφής είναι οι τροχοί με το πορτοκαλί χρώμα και με το άσπρο είναι οι πλαστική βάση πάνω στην οποία στηρίζονται.



Βλέπουμε σε πρώτο πλάνο τους τροχούς και από το μοτέρ προς τον άξονα περιστροφής έχουμε τον λόγο οδόντων (gear ratio) :

10:30:40 = 1:4

9:27 = 1:3

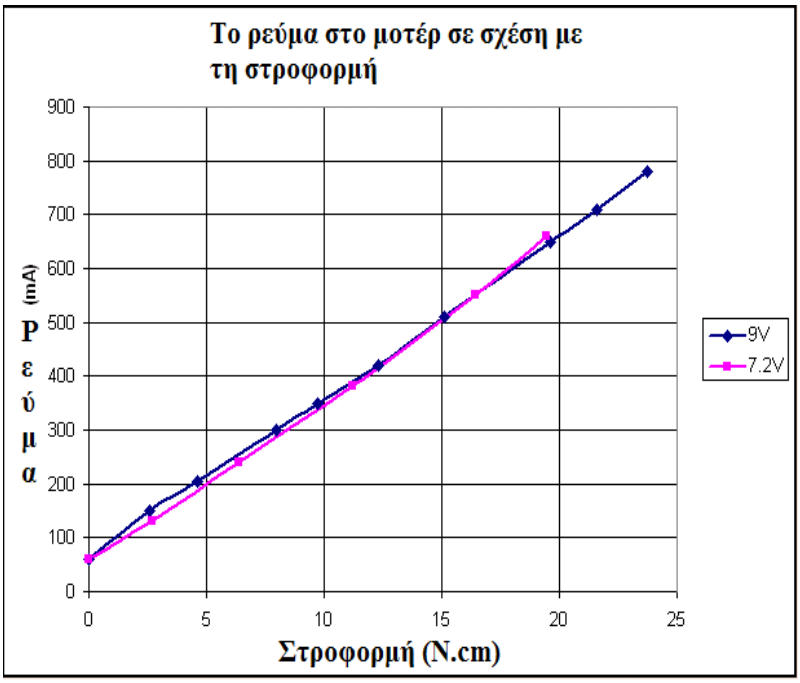
10:20 = 1:2

10:13:20 = 1:2

Σύνολο αν τα πολλαπλασιάσουμε έχουμε 1:48

Ο τροχός που ελέγχεται απο τον οπτικό αισθητήρα έχει 12 σχισμές.

Τα υπόλοιπα μέρη τροχών κινούνται σύμφωνα με τον τροχό του οπτικού αισθητήρα όπως φαίνετε και στο σχήμα αποτελείται από 12 σχισμές και λόγο οδόντων (gear ratio) 10:32. Για μια περιστροφή του άξονα το γρανάζι του οπτικού αισθητήρα γυρίζει 48*10/32=15 φόρες. Ο οπτικός αισθητήρας βλέπει 15*12=180 σχισμές. Χρησιμοποιώντας και τις δυο πλευρές των σχισμάτων μας δίνει κανονικά 360 χτυπήματα σε μια περιστροφή .



Στο παραπάνω σχεδιάγραμμα παρατηρούμε δυο γραφικές παραστάσεις η μπλε είναι η τάση που δέχεται το μοτέρ από αλκαλικές μπαταριές και είναι 9Volt και η ροζ που είναι η τάση που δέχεται το μοτέρ από την επαναφορτιζόμενη μπαταρία του. Παρατηρούμε ότι η ορμή σε σχέση με το ρεύμα παρουσιάζει γραμμική αύξηση με το φορτίο. Λόγω του υψηλού ρεύματος που περνάει μέσα από το Thermistor του μοτέρ και τον περιορισμό

τάσης που δέχεται το μοτέρ η καλύτερη λειτουργιά του θα είναι σε ροπή 15 N.cm για μεγάλες περιόδους. Για μεγαλύτερα φόρτια θα είναι καλύτερο να το χρησιμοποιούμε για μικρές περιόδους αλλά η προστασία θα μειώσει σύντομα την τρέχουσα και διαθέσιμη δύναμη.

2.16) Η λειτουργία Bluetooth στο Lego Mindstorms NXT

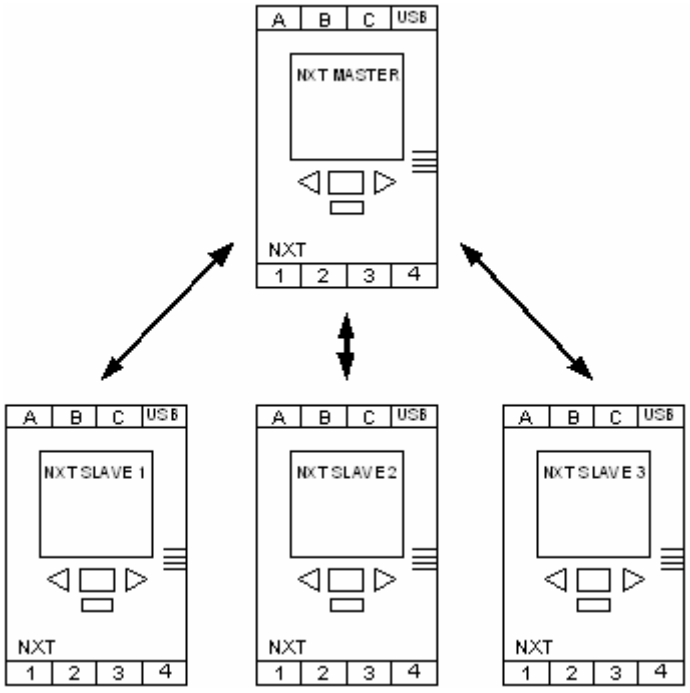
Το τούβλο NXT υποστηρίζει την ασύρματη επικοινωνία χρησιμοποιώντας Bluetooth® με τη συμπερίληψη ενός CSR BlueCore™ 4 έκδοση 2 τσιπ (βλέπε σχήμα-37). Το τούβλο NXT μπορεί να συνδεθεί ασύρματα με τρεις άλλες συσκευές συγχρόνως αλλά μπορεί μόνο να επικοινωνήσει με μια συσκευή τη φορά. Αυτή η λειτουργία έχει εφαρμοστεί χρησιμοποιώντας το serial Port Profile (SSP), το οποίο μπορεί να θεωρηθεί ασύρματη πόρτα επικοινωνίας.

Σχήμα-37. Η πλακέτα Bluetooth του NXT

Το τούβλο NXT μπορεί να επικοινωνήσει με τις συσκευές Bluetooth που μπορούν να προγραμματιστούν για να επικοινωνήσουν και τη χρησιμοποίηση των εντολών πρωτοκόλλου επικοινωνίας LEGO® MINDSTORMS® NXT και που υποστηρίζουν το serial Port Profile (SSP). Είναι πιθανό να στείλει τα προγράμματα και τα αρχεία μεταξύ

των υπόλοιπων NXT και να χρησιμοποιήσει την ασύρματη επικοινωνία για να στείλει και να λάβει τις πληροφορίες μεταξύ των υπόλοιπων Nxt κατά τη διάρκεια της εκτέλεσης ενός προγράμματος. Για να μειώσει την κατανάλωση ισχύος που χρησιμοποιείται από το Bluetooth, έχει εφαρμοστεί η τεχνολογία Bluetooth® κατηγορία II δεδομένου ότι μια συσκευή μπορεί να επικοινωνήσει μέχρι μια απόσταση περίπου 10 μέτρων.

Η λειτουργία Bluetooth μέσα στο τούβλο NXT είναι οργανωμένη ως κανάλι κύριας επικοινωνίας μεταξύ master και slave. Αυτό σημαίνει ότι ένα τούβλο NXT μέσα στο δίκτυο πρέπει να λειτουργήσει ως κύρια συσκευή(master) και τα άλλα τούβλα NXT να επικοινωνούν μέσω αυτού ως δευτερεύοντα(slave) εάν χρειάζονται. Το σχήμα-38 παρουσιάζει ποιες συσκευές NXT μπορούν να επικοινωνήσουν άμεσα μέσα σε ένα δίκτυο.



Σχήμα-38 Απεικόνιση επικοινωνίας Bluetooth τεσσάρων Nxt

Όπως φαίνεται στο σχημα-38, το master NXT μπορεί να συνδεθεί με τρεις άλλες συσκευές Bluetooth συγχρόνως. Το master NXT μπορεί μόνο να επικοινωνήσει με μια από τις Slave συσκευές κατά τη διάρκεια μιας δεδομένης στιγμής, που σημαίνει ότι εάν το master NXT επικοινωνεί με το NXT Slave 1 και το NXT Slave 3 αρχίζει τα στέλνοντας

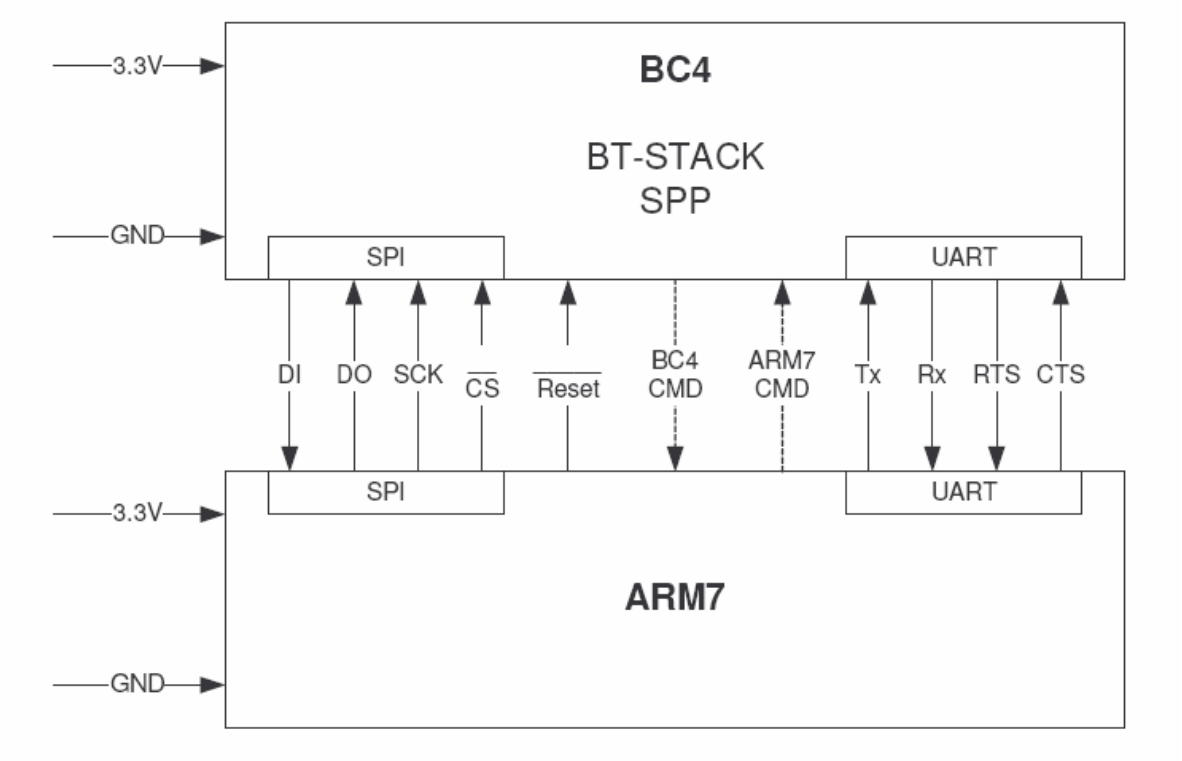
στοιχεία στο master NXT, το master NXT δεν θα αξιολογήσει τα λαμβανόμενα στοιχεία έως ότου επικοινωνήσει και με το NXT Slave 3. Ένα NXT δεν είναι ικανό να λειτουργήσει και ως Master συσκευή και ως Slave συγχρόνως επειδή αυτό μπορεί να προκαλέσει χαμένες πληροφορίες μεταξύ των συσκευών NXT. Αυτή η λειτουργία (δηλ., χρησιμεύοντας ως μια συσκευή Master και Slaves συγχρόνως) έχει τεθεί εκτός λειτουργίας τυποποιημένο firmware LEGO MINDSTORMS NXT. Οι συνδέσεις σε άλλες συσκευές Bluetooth εμφανίζονται μέσω των καναλιών. Το NXT χρησιμοποιεί τέσσερα κανάλια σύνδεσης για την επικοινωνία Bluetooth. Το κανάλι 0 χρησιμοποιείται πάντα από τις συσκευές NXT Slave στην επικοινωνία με το Master NXT (δηλ., προς το Master NXT) ενώ τα κανάλια 1, 2 και 3 χρησιμοποιούνται για την εξερχόμενη επικοινωνία από την Master συσκευή στις Slave συσκευές. Στο σχήμα-38 παραπάνω, το Master NXT θα χρησιμοποιήσει τα κανάλια επικοινωνίας 1, 2 και 3 κατά επικοινωνία αντίστοιχα με NXT SLAVE 1, NXT SLAVE 2 και NXT SLAVE 3. Όταν ένας από τους SLAVE NXT επικοινωνεί με τον Master NXT, θα χρησιμοποιήσει το κανάλι επικοινωνίας 0.

Διασύνδεση με το BlueCore Chip

Η λειτουργία Bluetooth μέσα στο NXT εφαρμόζεται χρησιμοποιώντας ένα αυτόνομο τσιπ, το CSR BlueCore™ 4 με μια Flash memory 8 Mbit. Το Bluetooth τσιπ από CSR περιέχει όλο το απαραίτητο υλικό για να τρέξει έναν ανεξάρτητο κόμβο Bluetooth. Ένας δεκαεξάμπιτος ενσωματωμένος επεξεργαστής τρέχει στο Bluetooth που εφαρμόζεται από CSR, αποκαλούμενο Bluelab. Το NXT τρέχει την έκδοση 3.2 Bluelab. Το Firmware μέσα στο BlueCore™ ενσωματώνει ένα επαναπρογραμματιζόμενο VM-task χρηστών που επιτρέπει σε μας για να ελέγχουμε και να τρέχουμε μέρη κώδικα για την εφαρμογή τους. Ένας διερμηνέας εντολής είναι ενσωματωμένος μέσα στο VM που είναι σε θέση να αποκωδικοποιήσει και να αποκριθεί στις εντολές που παραλαμβάνονται μέσω του interface UART από το ARM7 επεξεργαστή.

Το VM έχει μια πλήρη εφαρμογή και των Bluetooth profiles SSP-A και SSP-B. Το σχεδιάγραμμα SSP-A χρησιμοποιείται όταν το τοπικό BlueCore™ είναι σε λειτουργία σύνδεσης ενώ το SSP-B profile χρησιμοποιείται όταν αρχίζει μια άλλη συσκευή Bluetooth τη σύνδεση. Το BlueCore™ χρησιμοποιείται και αναφέρεται ως «stream mode» για να

ανταλλάξει τα στοιχεία σε ένα ποσοστό <= 220 K ταχύτητας μετάδοσης με την επιτυχία μιας σύνδεσης. Όταν BlueCore™ δεν είναι σε «stream mode» και είναι σε «command-mode» η οποία χρησιμοποιείται για να ελέγξει την εφαρμογή VM μέσα σε BlueCore™ και κατ' επέκταση, τη λειτουργία Bluetooth μέσα στο NXT. Η λειτουργία επικοινωνίας με το UART ελέγχεται από δύο σήματα (ARM7__CMD & BC4_CMD). Βλέπε σχήμα-39.



Σχήμα-39. Το Hardwork interface μεταξύ του ARM7 και BlueCore chip.

Το SPI interface επιτρέπει τη δυνατότητα συνεχούς ανανέωσης πληροφοριών από το τσιπ BlueCore™. Δεν χρησιμοποιείται κατά τη διάρκεια της κανονικής λειτουργίας του τούβλου NXT.

- Το reset χρησιμοποιείται για το σωστό ξεκίνημα του chip και για να απενεργοποιήσει το Bluetooth .
- BC4_CMD : Στέλνονται οι ενδείξεις από το BlueCore chip στον ARM7 και οι πληροφορίες data.
- ARM7__CMD : : Στέλνονται οι ενδείξεις από τον ARM7 στο BlueCore chip και οι πληροφορίες data.

- Η επικοινωνία UART χρησιμοποιείται για τα κοινά στοιχεία-data και για την επικοινωνία μεταξύ του BlueCore™ και του ARM7 επεξεργαστή. Με ταχύτητα επικοινωνίας : 460.8K bits/sec και Data bits:8 bits

Το τούβλο LEGO® MINDSTORMS® NXT μπορεί να επικοινωνήσει με τις εξωτερικές συσκευές Bluetooth που χρησιμοποιούν το profile (SSP) και μπορούν να προγραμματίσουν χρησιμοποιόντας το πρωτόκολλο επικοινωνίας LEGO® MINDSTORMS® NXT. Είναι επίσης πιθανό να στείλει άμεσες εντολές στο τούβλο LEGO® MINDSTORMS® NXT. Οι άμεσες εντολές ερμηνεύονται από το τούβλο NXT και μεταφράζονται σε συγκεκριμένες λειτουργίες χωρίς πρόσθετη αλληλεπίδραση χρηστών. Αυτό επιτρέπει τον άμεσο έλεγχο του τούβλου NXT από μια εξωτερική συσκευή Bluetooth® όπως ένα κινητό τηλέφωνο ή ένα PDA. Αυτό πετυχαίνεται χρησιμοποιώντας message commands, Message Read και Message Write. Όπως περιγράφεται νωρίτερα, η λειτουργία Bluetooth εφαρμόζεται το πρωτόκολλο Master - Slave που η επικοινωνία ελέγχεται από το Master Nxt. Αυτό επιτρέπει την αξιόπιστη επικοινωνία ειδικά όταν συνδέονται οι πολλαπλές συσκευές Slave με μια Master συσκευή. Όταν η Master συσκευή προγραμματίζεται για να ελέγξει εάν έχει λάβει τα στοιχεία από τη μια από τρεις πιθανές συσκευές Slave, θα στείλει την εντολή " Message Read " στη συγκεκριμένη συσκευή Slave για να ελέγξει ότι έχει τα ετα στοιχεία για να αρχίσει η επικοινωνία .

ΚΕΦΑΛΑΙΟ 3^{Ο}

Εισαγωγή στις γλώσσες προγραμματισμού

3.1) ΠΑΡΑΔΟΣΙΑΚΕΣ ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Στη δεκαετία του 50 φάνηκαν οι πρώτες γλώσσες προγραμματισμού, οι οποίες βασίζονταν σε ένα πρόγραμμα μεταγλώττισης (compiler) ο οποίος παρήγαγε προγράμματα (object). Εκείνη την εποχή τα mainframes που υπήρχαν εκτελούσαν άμεσα τα obj προγράμματα χωρίς βέβαια την ανάγκη ύπαρξης αντιστοιχών εκτελέσιμων προγραμμάτων (exe,com κτλ) όπως συμβαίνει στην μικροπληροφορική σήμερα.

Αρχικά εμφανίστηκαν 3 γλώσσες προγραμματισμού, η COBOL, η FORTRAN και η RPG. Η COBOL χρησιμοποιήθηκε αποκλειστικά για οικονομικές-λογιστικές εφαρμογές, όπου υπερτερούσε των άλλων σε αυτοματοποιημένες διαδικασίες που διέθετε για προσπέλαση των πρωτόγονων τότε βάσεων δεδομένων. Η FORTRAN κυριάρχησε σε επιστημονικές-τεχνικές εφαρμογές και έθεσε την αρχή μεταβλητών και εντολών που θα είχαν οι μετέπειτα γλώσσες προγραμματισμού. Η RPG ήταν μια μικρή, ευέλικτη γλώσσα που έβγαλε η IBM, αποκλειστικά για προγράμματα με λίγη απλή λογική και μεγάλο όγκο εξόδου εκτυπώσεων.

Τα βασικά ελαττώματα των πρώτων αυτών γλωσσών προγραμματισμού φάνηκαν από την δεκαετία του 70 και μετά με την κρίση του software (software crisis) του δομημένου προγραμματισμού. Τότε η COBOL μεταλλάχτηκε στη COBOL 74, η FORTRAN περνώντας από διάφορα στάδια, εκδόσεις π.χ. FORTRAN II, IV, V, 77), ενώ η RPG είχε αρχίσει ήδη να εξαφανίζεται. Στην εποχή του δομημένου προγραμματισμού υπήρχαν οι γλώσσες προγραμματισμού ALGOL 4, PASCAL, ADA, BASIC, COBOL 74, και FORTRAN 77. Από αυτές η PASCAL κατέληξε στη TURBO PASCAL, η ADA σαν επίσημη γλώσσα του NATO, η BASIC σε διάφορες εκδόσεις των Home computers της εποχής εκείνης.

Στις 10 Νοεμβρίου 1983 η Microsoft ανακοίνωσε τα Microsoft Windows, μια επέκταση του λειτουργικού συστήματος MS-DOS, το οποίο παρείχε γραφικό περιβάλλον στους χρήστες των PC. Στην συνέχεια ακολούθησαν οι εκδόσεις Windows 1.0 (1985), 2.0 (1987), 3.0 (1990), 3.11 (1993), NT (1993), 95 (1995), NT4 (1996), 98 (1998), ME (2000), 2000 (2000), XP (2001), 2003 (2003). Στο διάστημα 1990 – 2000 έγινε η επανάσταση στο προγραμματισμό με την εισαγωγή των visual programming εργαλείων και την υποστήριξη από την Visual Basic της Microsoft.

Η COBOL δεν τα πήγαινε γενικά καλά με τους μικροϋπολογιστές από την δεκαετία του 80. Αν και προσπάθησε στη δεκαετία του 90 με τη VISUAL COBOL σήμερα στη πρώτη δεκαετία του 2000 δεν μπορεί κανείς να ισχυριστεί ότι υπάρχει η γλώσσα αυτή. Η FORTRAN μεταλλάχτηκε από FORTRAN 77 στην FORTRAN 90 και στην συνέχεια στην VISUAL FORTRAN ++, κύρια με την συνεργασία της MICROSOFT και της DIGITAL. Είναι όμως τελικά μια δύσκολη γλώσσα, δεν τηρεί ορθά της αρχές του Object Oriented Programming και τελικά αφού κυριάρχησε για δεκαετίες έκλεισε τον κύκλο της αφήνοντας την θέση της σε νεότερες γλώσσες και σύγχρονα λογισμικά εργαλεία. Ωστόσο διδάσκεται ακόμα σε πολλές πολυτεχνικές κυρίως σχολές στη γη ενώ υπάρχουν πολλές βιβλιοθήκες γραμμένες σε FORTRAN. Η γλώσσα προγραμματισμού Visual Basic αναπτύχθηκε χωρίς την υποστήριξη object oriented αρχιτεκτονικής. Είχε σαν κύριο στόχο την παροχή ευκολιών στο προγραμματιστή και τον γρήγορο προγραμματισμό.Η καλύτερη και πιο δημοφιλής εφαρμογή της Visual Basic εμφανίζεται μέσω του προϊόντος Microsoft Access, το οποίο πρωτοεμφανίστηκε το 1992. Η ADA έγινε VISUAL ADA, την οποία όμως γνωρίζουν μόνο οι ιεροφάντες του προγραμματισμού της Εθνικής Άμυνας... Η TURBO PASCAL της BORLAND κατέληξε τελικώς στη DELPHI η οποία όμως δεν συμβιβάστηκε με την αντικειμενοστραφή τεχνολογία. Ωστόσο η DELPHI αφήνει καλές εντυπώσεις και σήμερα ακόμα και έχει σωστές επαφές με σχεσιακές βάσεις δεδομένων.

Συμπερασματικά οι παραδοσιακές γλώσσες προγραμματισμού υστερούσαν σε διάφορα επίπεδα. Όλες όμως υστερούσαν στην ορθή εφαρμογή της αντικειμενοστραφής τεχνολογίας, γεγονός αποφασιστικής σημασίας για την εξαφάνιση τους.

3.2) Γλώσσες αντικειμενοστραφούς τεχνολογίας

Από τη δεκαετία του 60 ήταν φανερή η έλλειψη σχεδιαστικού εργαλείου ευκολιών και ανάπτυξης προγραμμάτων σε low-level και καλή επαφή με το Master Control Program (MCP). Και βέβαια οι γλώσσες της εποχής εκείνης όπως η FORTRAN δεν ήταν καθόλου ευκίνητες για δόμηση λειτουργικών συστημάτων. Ωστόσο το 1965 στο Cambridge έγινε μια πρώτη απόπειρα με τη γλώσσα BCPL με κύριο στόχο την αντικατάσταση της ASSEMBLY.

Η BCPL δημιούργησε την γλώσσα B από όπου δομήθηκε το κλασσικό λειτουργικό σύστημα UNIX στις ιστορικές μηχανές PDP-7/11 και της DEC. Όμως η γλώσσα B δεν είχε I/O υποσύστημα, έλλειψη που θα γενικευτεί τα υπόλοιπα χρόνια και θα είναι ένα από τα βασικότερα ελαττώματα του μοντέρνου προγραμματισμού των ημερών μας. Μάλιστα ο Dennis Ritchie το 1972 έβγαλε από τη γλώσσα B τη γλώσσα C από όπου προήρθε η ANSI Standard C που παρέμεινε στενός συνεργάτης του UNIX.

Στη δεκαετία του 80 η MICROSOFT έκανε ανεξάρτητη τη C από το UNIX με την MS-C. Η MS-C αλλά και η C με UNIX ήταν εύκολη γλώσσα με μικρό πλήθος εντολών και μπορούσε να την μάθει κάποιος σχετικά εύκολα. Υστερούσε βέβαια σε πολλά θέματα, τα οποία σε παλαιότερες γλώσσες προγραμματισμού (π.χ. PASCAL, ADA, FORTRAN) ήταν απλές δυνατότητες ρουτίνας (π.χ. αυτόματες μετατροπές τύπων δεδομένων, ταχύτητες σε binary αρχεία, κτλ). Οπωσδήποτε η γλώσσα C δεν ήταν αυτή που άξιζε, απλά θα είναι ο γεννήτορας αυτών που πραγματικά θα αξίζουν, όπως η C++, JAVA, C#, κτλ.

Η εξέλιξη της C στη C++ και η τυποποίηση αυτής από το ινστιτούτο ANSI θεμελίωσε τις αρχές της αντικειμενοστραφούς τεχνολογίας ακολουθώντας τον αντικειμενοστραφή προγραμματισμό που παράλληλα αναπτυσσόταν μέσα στη δεκαετία του 90,(πχ UML). Η C++ είχε ισχυρή τυποποίηση, νέα ρεύματα input/output(iostream) αλλά πάνω από όλα θεμελίωσε τη χρήση της κλάσης. Μάλιστα από το 1993 και μετά η MICROSOFT βασιζόμενη στις εκδόσεις των studios υποστήριζε την MS VISUAL C++ η οποία κατά κανόνα επεκράτησε. Διακρίνεται σε άριστη συνεργασία με RDBMS, ορθή λειτουργία σε LAN(Local Area Network), Web και άριστη συνεργασία με διαδικτυακό προγραμματισμό, (π.χ. JAVA, C#, XML κτλ).

Συμπερασματικά η αντικειμενοστραφής τεχνολογία άρχισε με τη C++/VISUAL C++ που πήρε τη σκυτάλη από την VISUAL FORTRAN. Σήμερα θεωρείται το αλφάβητο του προγραμματισμού και κανονικά κάθε μηχανικός λογισμικού θα πρέπει άριστα να τη γνωρίζει. Ωστόσο από τη δεκαετία του 90 το μέλλον είχε αρχίσει να επικεντρώνεται στο Διαδίκτυο και ήταν φανερό πως έπρεπε να αναπτυχθούν νέα εργαλεία και γλώσσες προσαρμοσμένα για αυτό.

3.3) Διαδικτυακές γλώσσες προγραμματισμού

Οι νεότερες εκδόσεις του λειτουργικού εκμεταλλευόμενες τη ραγδαία εξέλιξη και επανάσταση του Internet και του World Wide Web (WWW) στήριξαν με εργαλεία και γλώσσες τον διαδικτυακό προγραμματισμό στα ακόλουθα χαρακτηριστικά:

- Μεταφερσιμότητα (portability) κατά κανόνα αντιστρόφως ανάλογη του αναμενόμενου υπολογιστικού χρόνου
- Αντικειμενοστραφής προγραμματισμός κύρια με κλάσεις και αντικείμενα
- Εύκολη συντήρηση, επέκταση και διαχρονικότητα

Το 1991 οι ερευνητές της SUN J. Gosling, E. Frank, M. Sheridan, C. Warth δημιούργησαν τη γλώσσα OAK για προγραμματισμό ηλεκτρονικών συσκευών σπιτιού (φούρνοι μικροκυμάτων, τηλεχειριστήρια...). Το 1995 η OAK μετονομάστηκε σε JAVA με λειτουργία σε διαφορετικά περιβάλλοντα, υψηλή απόδοση σε Internet και σε Multimedia εφαρμογές. Παράλληλα το 1990 ο Tim Berners-Lee, στο CERN, επέβαλε μια τάξη στο χάος που κυριαρχούσε στο Διαδίκτυο δημιουργώντας ένα απλό σύστημα κωδικοποίησης των πληροφοριών του σε Υπερκείμενα (Hypertext). Έτσι γεννήθηκε η ειδική γλώσσα περιγραφής υπερκείμενων HyperText Markup Language (HTML) καθώς και το πρωτόκολλο μεταφοράς δεδομένων HyperText Transport Protocol (HTTP).

Ωστόσο η JAVA δεν κατάφερε να καλύψει βασικές αδυναμίες. Η πρώτη από αυτές ήταν η σχεδόν αδυναμία της ανάμιξης της με πολλές άλλες γνωστές γλώσσες προγραμματισμού (mixed language programming). Βέβαια η ανάμειξη πολλών γλωσσών προγραμματισμού απαιτεί την ύπαρξη μεγάλων κατανεμημένων προγραμματιστικών

συστημάτων (Huge distributed software systems). Η JAVA υποστηρίζεται στις νεότερες εκδόσεις των Microsoft Windows μέσω του Virtual Java Machine της SUN. Η Microsoft έχει αναπτύξει το δικό της Virtual Machine για την JAVA το οποίο όμως δεν παρέχει την ίδια υποστήριξη και αξιοπιστία. Το Virtual Machine της Microsoft αναπτύχθηκε για να υποστηρίξει την Visual J++ γλώσσα η οποία ακολούθησε τις προδιαγραφές του JDK μέχρι την έκδοση 1.2.1. Με την εισαγωγή όμως της πλατφόρμας .net η J++ αντικαταστάθηκε από την J# (java syntax language) και την C# οι οποίες λειτουργούν με ανεξάρτητο περιβάλλον runtime από το VM.O δημιουργός της C# είναι ο Anders Hejlsberg, ένας από τους κορυφαίους προγραμματιστές με πληθώρα αξιοσημείωτων επιτευγμάτων στο ενεργητικό του.

Η C# είναι απευθείας συσχετιζόμενη με τη C, C++ και Java, και αυτό δεν είναι τυχαίο καθώς αυτές είναι οι τρεις ποιο διαδεδομένες και χρησιμοποιήσιμες γλώσσες προγραμματισμού στον κόσμο. Για αυτό η σύνταξη της C# έγινε με κατανοητή υποδομή παρεμφερής της C, C++ και Java έτσι ώστε να προσφέρει έναν βατό δρόμο μετανάστευσης από αυτές προς εκείνη. Τελικώς η C# εστίασε τις ελλείψεις που προϋπήρχαν στην γλώσσα προγραμματισμού Java και με συγκεκριμένες βελτιώσεις και καινοτομίες έλυσε τα προβλήματα αυτά.

Συνοψίζοντας αξίζει να σημειωθεί πως η C# προσπαθεί να συνθέσει τη C++ και τη JAVA σε αντίθεση με τη JAVA που ήταν απλά θυγατέρα της C++, γεγονός που καθιστά τη C# "συνδυαστικό" αντικαταστάτη των δυο αυτών σύγχρονων γλωσσών προγραμματισμού.

3.4) Πως δουλεύει η κοινή γλώσσα χρόνου εκτέλεσης

Το CLR διαχειρίζεται την εκτέλεση του .net κώδικα. Όταν μεταγλωττίζετε ένα C# πρόγραμμα, το εξερχόμενο από το μεταγλωττιστή αρχείο δεν είναι το εκτελέσιμο. Εν αντιθέσει, είναι ένα αρχείο που περιέχει ένα ειδικό τύπο ψευδοκώδικα που λέγεται ενδιάμεση γλώσσα της Microsoft (Microsoft Intermediate Language - MSIL), η οποία ορίζει ένα μέρος από εύχρηστες καθοδηγήσεις που είναι ανεξάρτητες από τις

υπολογιστικές μονάδες του επεξεργαστή (CPU). Κυρίως η MSIL δημιουργεί τη γλώσσα χαμηλού επιπέδου (assembly language).

Η ύπαρξη του CLR έγκειται στο να μετατρέπει το MSIL σε εκτελέσιμο κώδικα όταν το πρόγραμμα εκτελείται. Συνεπώς, οποιοδήποτε πρόγραμμα μεταγλωττίζεται στο MSIL μπορεί να εκτελείται σε κάθε περιβάλλον στο οποίο παρέχεται το CLR. Αυτό είναι το σημείο στο οποίο το .net Framework πετυχαίνει ευχρηστία σε όλα τα περιβάλλοντα.

Η ενδιάμεση γλώσσα της Microsoft (MSIL) μετατρέπεται σε εκτελέσιμο κώδικα χρησιμοποιώντας τον JIT μεταγλωττιστή (Just-In-Time). Η διαδικασία είναι η εξής: Όταν ένα .net πρόγραμμα εκτελείται, το CLR ενεργοποιεί τον JIT μεταγλωττιστή. Ο JIT μεταγλωττιστής μετασχηματίζει MSIL σε εκ γενετή κώδικα (native code) πάνω σε βασισμένη διαταγή καθώς κάθε μέρος του προγράμματος το χρειάζεται. Συνεπώς, το C# πρόγραμμα πραγματικά εκτελείται σαν εκ γενετή κώδικα αν και κατά αρχήν μεταγλωττίζεται σε MSIL. Αυτό μεταφράζεται ότι το πρόγραμμα τρέχει τόσο γρήγορα όσο εάν είχε μεταγλωττιστεί σε εκ γενετή κώδικα από την αρχή, αλλά κέρδισε τα

προτερήματα ευχρηστίας από το MSIL.

Εν κατακλείδι, το MSIL εξάγει και ένα άλλο πράγμα κατά τη μεταγλώττιση ενός C# προγράμματος: τα metadata. Ως metadata χαρακτηρίζονται τα δεδομένα που χρησιμοποιούνται από το πρόγραμμα και επιτρέπουν στο κώδικα να αλληλεπιδρά με διαφορετικό κώδικα. Τα Metadata συμπεριλαμβάνονται στο ίδιο αρχείο όπως το MSIL.

3.5) Διαχειριζόμενος και μη κώδικας

Γενικά, όταν γράφουμε ένα C# πρόγραμμα, δημιουργείται αυτό που αποκαλείται διαχειριζόμενος κώδικας (managed code). Ο διαχειριζόμενος κώδικας εκτελείται κάτω από τον έλεγχο του CLR με τη μεθοδολογία που παρουσιάστηκε. Από τη στιγμή που εκτελείται κάτω από τον έλεγχο του CLR, ο διαχειριζόμενος κώδικας είναι αντικείμενο στους προκαθορισμένους περιορισμούς και παράγει αρκετά πλεονεκτήματα. Οι περιορισμοί μπορούν εύκολα να περιγραφτούν και να συναντηθούν καθώς προγραμματίζετε σε C#. Οι

περιορισμοί αυτοί είναι ότι ο μεταγλωττιστής πρέπει να δημιουργήσει ένα MSIL αρχείο στοχεύοντας για το CLR και για να χρησιμοποιήσει τη βιβλιοθήκη κλάσεων του .NET Framework. Από την άλλη πλευρά τα πλεονεκτήματα είναι αξιοσημείωτα όπως παρέχεται ένας σύγχρονος διαχειριστής μνήμης, διαθέτει την ικανότητα να αναμιγνύει διαφορετικές γλώσσες προγραμματισμού, εξασφαλίζει μεγαλύτερη ασφάλεια, υποστηρίζει έλεγχο για νεότερες εκδόσεις και μια καθαρή οδό για να αλληλεπιδράσουν τα components του λογισμικού.

Από την αντίθετη πλευρά είναι ο μη διαχειριζόμενος κώδικας (unmanaged code) ο οποίος δεν εκτελείται κάτω από την έλεγχο του CLR. Συνεπώς όλα τα προγράμματα που υπήρχαν πριν της εκδόσεως του .NET Framework χρησιμοποιούσαν μη διαχειριζόμενο κώδικα. Είναι πιθανό ο διαχειριζόμενος και μη κώδικας να εκτελούνται μαζί, έτσι ώστε η γέννηση διαχειριζόμενου κώδικα από τη C# δεν περιορίζει την δυνατότητα να λειτουργούν σε συνδυασμό με τα προϋπάρχοντα προγράμματα.

3.6) Καθορισμός της κοινής γλώσσας

Το CLS (Common Language Specification) προσφέρει ανοικτές προδιαγραφές που θα πρέπει να τηρεί οποιαδήποτε γλώσσα προγραμματισμού θέλει να εκτελείται μέσω του CLR και να αποκαλείται .NET γλώσσα προγραμματισμού. Το CMS περιέχει το CTS (Common Type System) το οποίο αφορά κοινούς τύπους δεδομένων για όλες τις .NET γλώσσες προγραμματισμού ώστε να είναι εφικτή η δημιουργία εφαρμογών από source κώδικα διαφορετικών .NET γλωσσών προγραμματισμού.

ΚΕΦΑΛΑΙΟ 4^{Ο}

Η Java

4.1) Εισαγωγή

Στο Σαν Φρανσίσκο στις 23 Μαΐου 1995, όταν ο επικεφαλής της έρευνας στη Sun Microsystems, ο John Gage, και η διασημότερη ίσως φυσιογνωμία του διαδικτύου, ο δημιουργός του Netscape Marc Andreessen παρουσίαζαν επίσημα τη γλώσσα προγραμματισμού Java και τη σχετική τεχνολογία.

Δεν ήταν περίεργο που ο ένας από τους δύο άνδρες στη σκηνή ήταν ο Gage, αφού η νέα γλώσσα πρωτοείδε το φως σε ένα ερευνητικό πρόγραμμα της Sun που ονομαζόταν ‘’Green’’. Ο Andreessen συμμετείχε στο μικρό όμιλο φοιτητών που είχαν κατασκευάσει τον πρώτο φυλλομετρητή (browser) με τη βοήθεια γραφικών για τον παγκόσμιο ιστό (www), το Mosaic. Το πρόγραμμα αυτό, και ο διάδοχός του, το Netscape, είχαν προκαλέσει στην αρχή της δεκαετίας του 1990 μία επανάσταση στον κόσμο των υπολογιστών. Η νέα συνεργασία μεταξύ Netscape και Sun θα επέτρεπε τη σύνταξη μικρών προγραμμάτων σε Java, τα οποία θα μετέτρεπαν τις ιστοσελίδες σε αλληλεπιδραστικές και θα τις έκαναν πιο ζωντανές. Τα σχέδια αυτά γνώρισαν μεγάλη δημοσιότητα και υπήρχαν πάμπολλες ιδέες για τον προγραμματισμό μιας τοστιέρας μέσω του Ιστού – και άλλες πολλές.

Ακόμη και σήμερα, η Java συνδέεται στενά με το διαδίκτυο και υπάρχουν πολλοί άνθρωποι που νομίζουν ότι πρόκειται για μια γλώσσα αποκλειστικά για προγραμματισμό στον Ιστό. Πρόκειται για μια εντύπωση απολύτως εσφαλμένη. Η Java είναι μια πλήρης γλώσσα προγραμματισμού, με τις δικές της χαρακτηριστικές λειτουργίες και με περισσότερο ή λιγότερο τυπικά πεδία χρήσης.

Η Java είναι μια νεαρή γλώσσα προγραμματισμού. Οκτώ ή δέκα χρόνια δεν είναι ιδιαίτερα πολλά. Η συνέπεια είναι ότι οι επαγγελματίες της πληροφορικής και οι σπουδαστές της θα πρέπει μα παρακολουθούν τη διαρκή εξέλιξη της Java και των σχετικών

με αυτή προγραμμάτων. Η γραφή προγραμμάτων υπολογιστών σε μια γλώσσα που γνωρίζει διαρκώς νέες βελτιώσεις μοιάζει κάτι το ανέφικτο, στην περίπτωση της Java πάντως, είναι απολύτως εφικτό. Αυτό οφείλεται στη σχεδίαση και στη φιλοσοφία της γλώσσας.

Η ραγδαία εξάπλωση του Internet και του World-Wide Web δημιούργησαν την ανάγκη νέων τρόπων ανάπτυξης και διανομής του λογισμικού. Οι απαιτήσεις αυτές οδήγησαν στην δημιουργία της γλώσσας προγραμματισμού Java, από την εταιρία Sun microsystems TM. Η Java σχεδιάστηκε με σκοπό την ανάπτυξη εφαρμογών που θα τρέχουν σε ετερογενή δικτυακά περιβάλλοντα.

Η Java έχει τα ακόλουθα χαρακτηριστικά:

• **Αντικειμενοστραφής** (ομοιότητες εντολών με τη C++).

• **Δημιουργία ανεξάρτητων εφαρμογών και applets** (applet = προγράμματα που περιλαμβάνονται σε HTML σελίδες και εκτελούνται από τον Web Browser).

• **Είναι Interpreted γλώσσα.** Αυτό σημαίνει ότι ο java compiler δεν παράγει εκτελέσιμο κώδικα αλλά μια μορφή ψευδοκώδικα (bytecode) το οποίο από μόνο του δεν τρέχει σε καμία μηχανή. Προκειμένου λοιπόν να εκτελεστεί απαιτείται η χρήση ενός interpreter(=διερμηνέα) για να μετατρέψει το bytecode σε πραγματικό εκτελέσιμο κώδικα. Αυτό το χαρακτηριστικό δίνει τη δυνατότητα στα java bytecodes να μπορούν να τρέξουν σε οποιοδήποτε μηχάνημα, κάτω από οποιοδήποτε λειτουργικό, αρκεί να έχει εγκατασταθεί ένας java interpreter. Επίσης ένα άλλο χαρακτηριστικό του java bytecode είναι το μικρό του μέγεθος, (μόλις λίγα Kilobytes). Αυτό το κάνει ιδανικό για μετάδοση μέσω του δικτύου.

• **Κατανεμημένη** (distributed). Δηλαδή ένα πρόγραμμα σε Java είναι δυνατό να το φέρουμε από το δίκτυο και να το τρέξουμε. Επίσης είναι δυνατό διαφορετικά κομμάτια του προγράμματος να έρθουν από διαφορετικά sites.

• **Ασφαλής** (secure). Στο δίκτυο όμως ελλοχεύουν πολλοί κίνδυνοι για τον χρήστη - παραλήπτη μιας δικτυακής εφαρμογής, γι' αυτό η Java έχει σχεδιαστεί έτσι ώστε να ελαχιστοποιείται η πιθανότητα προσβολής του συστήματος του χρήστη από κάποιο applet γραμμένο για τέτοιο σκοπό.

• **Είναι multithreaded**. Η Java υποστηρίζει εγγενώς την χρήση πολλών threads.

Προκειμένου να το επιτύχει αυτό σε συστήματα με έναν επεξεργαστή, το Java runtime system (interpreter) υλοποιεί ένα δικό χρονοδρομολογητή (scheduler), ενώ σε συστήματα που υποστηρίζουν πολυεπεξεργασία η δημιουργία των threads ανατίθεται στο λειτουργικό σύστημα. Φυσικά όλα αυτά είναι αόρατα τόσο στον προγραμματιστή όσο και στον χρήστη.

• Υποστηρίζει multimedia εφαρμογές. Με αυτό εννοούμε ότι η Java παρέχει ευκολίες στη δημιουργία multimedia εφαρμογών. Αυτό επιτυγχάνεται τόσο με την ευελιξίας της σαν γλώσσα όσο και με τις πλούσιες και συνεχώς εμπλουτιζόμενες βιβλιοθήκες της.

4.2) Τα εργαλεία της Java

Ακολούθως παρουσιάζονται εν συντομία όλα τα εργαλεία που έρχονται με τη Java 2 Platform Standard Edition (J2SE), της Sun microsystems. Η παρούσα έκδοση της Java και του J2SE είναι η Java 2 έκδοση 5.0.

• **javac** Είναι ο compiler της Java. Η χρήση του στο command-line είναι : *javac <όνομα αρχείου>*. Εδώ να σημειώσουμε ότι το javac δεν παράγει ένα αρχείο με όλον τον κώδικα, αλλά χωριστό αρχείο για κάθε κλάση. Τα αρχεία των κλάσεων ονομάζονται : *<όνομα κλάσης>.class.*

• **java** Είναι ο interpreter της Java. Η χρήση του είναι η εξής : *java <κλάση>, πχ java myClass και όχι java myClass.class.*

• **javaw** (MONO στα Windows 95/NT) Είναι παρόμοιο με το java με μόνη την διαφορά ότι δεν χρειάζεται shell για να τρέξει.

• **jdb** Είναι ο Java debuger.

• **javah** Κατασκευάζει C files και stub files για κάποια κλάση. Αυτά τα αρχεία είναι απαραίτητα όταν θέλουμε να υλοποιήσουμε κάποιες από τις μεθόδους της κλάσης σε C,πράγμα πολύ σπάνιο.

• **javap** Είναι ο Java disassembler.

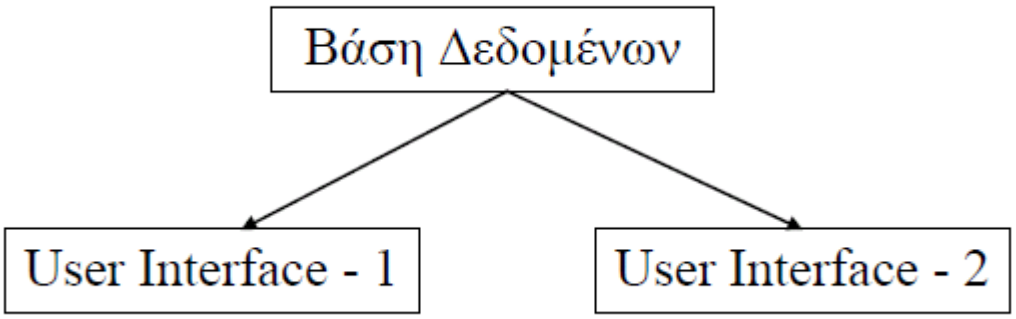
• **javadoc** Είναι ένα πρόγραμμα για αυτόματη κατασκευή documentation.

Είναι αρκετά χρήσιμο στην κατασκευή βοηθημάτων και τεχνικών αναφορών για εφαρμογές οποιουδήποτε μεγέθους.

• **appletviewer** Είναι ένα πρόγραμμα το οποίο μας επιτρέπει να τρέχουμε και να χρησιμοποιούμε τα διάφορα applets σε Java. Οι stand-alone εφαρμογές, ωστόσο, δεν τρέχουν στον appletviewer αλλά κατευθείαν στον java ή javaw.

4.3) Εισαγωγή στον αντικειμενοστραφή προγραμματισμό

Ο αντικειμενοστραφής προγραμματισμός, (OOP - Object Oriented Programming), είναι μια προγραμματιστική φιλοσοφία όπως και ο προστακτικός ή ο λογικός προγραμματισμός. Σύμφωνα με τον OOP ένα πρόγραμμα ΔΕΝ αποτελείται από τα δεδομένα και τον κώδικα που τα επεξεργάζεται αλλά από αντικείμενα (objects) τα οποία εμπεριέχουν τα δεδομένα και τα οποία (αντικείμενα) ανταλλάσσουν μεταξύ τους πληροφορίες και μηνύματα προκειμένου να επιτευχθεί ο στόχος του προγράμματος. Για παράδειγμα έστω ότι έχουμε ένα σύστημα που περιλαμβάνει μια βάση δεδομένων καθώς και user interfaces με τα οποία οι χρήστες επικοινωνούν και αλληλεπιδρούν με την Β.Δ..



Τα UIs στέλνουν τις εντολές των χρηστών στην ΒΔ και παρουσιάζουν στην οθόνη τις απαντήσεις που λαμβάνουν. Δηλαδή όλο το πρόγραμμα αποτελείται από ένα αντικείμενο ΒΔ και 2 αντικείμενα UI τα οποία είναι ολόιδια (αλλά εξυπηρετούν άλλους χρήστες). Συνεπώς ο κώδικας που θα γράφαμε θα περιείχε τον ορισμό ενός αντικειμένου ΒΔ, τον ορισμό ενός αντικειμένου UI και την κατασκευή ενός ΒΔ και 2 UI αντικειμένων.

Ο κώδικας που ορίζει ένα αντικείμενο λέγεται **κλάση** (class) του αντικειμένου αυτού. Η κλάση χρησιμοποιείται σαν *μήτρα* για την κατασκευή πανομοιότυπων αντιγράφων -αντικειμένων. Τα αντικείμενα - αντίγραφα, λέγονται στιγμιότυπα (instances) της κλάσης αυτής και η κατασκευή και αρχικοποίηση ενός από αυτά λέγεται instantiation. Το αντικείμενο αυτό καθ' εαυτό είναι μια οντότητα στη μνήμη η οποία περιέχει δεδομένα καθώς και μεθόδους μέσω των οποίων μπορούμε να αλλάξουμε τα δεδομένα ή να

επικοινωνήσουμε με το αντικείμενο. Αντίθετα η κλάση είναι απλώς ένα πρότυπο για την δημιουργία αντιγράφων του ίδιου αντικειμένου. Είναι δηλαδή ότι είναι ο Τύπος Δεδομένων για τις μεταβλητές (στον δομημένο προγραμματισμό).

4.4) Δομή ενός προγράμματος σε Java

< ΕΙΣΑΓΩΓΗ ΕΤΟΙΜΩΝ ΚΛΑΣΕΩΝ ΑΠΟ ΤΗ ΒΙΒΛΙΟΘΗΚΗ >

<ΟΡΙΣΜΟΣ ΝΕΑΣ ΚΛΑΣΗΣ>

<ΟΡΙΣΜΟΣ ΝΕΑΣ ΚΛΑΣΗΣ>

κ.ο.κ .

Μία από όλες τις κλάσεις θα παίξει το ρόλο της αφετηρίας του προγράμματος, με άλλα λόγια θα είναι σαν την main() της C. Αναλυτικότερα, θα αναλάβει να κατασκευάσει τα αντικείμενα που χρειάζεται το πρόγραμμα, με τη χρήση των κλάσεων που ορίσαμε ή φέραμε από τη βιβλιοθήκη. Φυσικά την αρχική κλάση θα την υποδείξουμε εμείς όταν θα ξεκινήσουμε τον interpreter της γλώσσας.

4.5) Μια συζήτηση σχετικά με τα αντικείμενα στη JAVA

Όπως ειπώθηκε και παραπάνω τα αντικείμενα περιέχουν δεδομένα και παρέχουν μεθόδους για την επεξεργασία των δεδομένων αυτών καθώς και για την επικοινωνία με άλλα αντικείμενα. Ας εμβαθύνουμε λίγο περισσότερο σε αυτές τις έννοιες. Κατ` αρχήν τα δεδομένα μπορεί να είναι άλλα αντικείμενα που περιέχονται μέσα σε ένα άλλο αντικείμενο ή μπορεί να είναι κοινές μεταβλητές όπως τις γνωρίζουμε από την C και την PASCAL. Τις μεταβλητές και τα αντικείμενα αυτά τα καλούμε *πεδία ή instance variables* του αντικειμένου.

Σε αυτό το σημείο να παρατηρήσουμε ότι μία κλάση μοιάζει με ένα structure της C το οποίο μπορεί να περιέχει κοινές μεταβλητές ή μεταβλητές που προέκυψαν από άλλα structures. Επιπλέον μία κλάση (ή ένα αντικείμενο) περιέχει και μεθόδους για την επικοινωνία του με τον *έξω κόσμο*, δηλαδή τα άλλα αντικείμενα. Οι μέθοδοι αυτοί υλοποιούνται σαν συναρτήσεις παρόμοιες με αυτές της C. Όταν λοιπόν κάποιος θέλει να ζητήσει κάτι από ένα αντικείμενο, (ή εναλλακτικά να στείλει ένα μήνυμα -αίτημα), δεν έχει παρά να καλέσει - εκτελέσει την αντίστοιχη μέθοδο του αντικειμένου. Αυτό γίνεται πολύ απλά ως εξής :

<Αντικείμενο>. <μέθοδος>(<Λίστα παραμέτρων>);

Π.χ.

mycar.startEngine();

Αυτό το σχήμα μας θυμίζει πολύ τον τρόπο πρόσβασης σε μία μεταβλητή που περιέχεται σε ένα structure της C. Επίσης να σημειώσουμε ότι μια μέθοδος μπορεί να καλεί άλλες μεθόδους του ίδιου αντικειμένου ή να επεξεργάζεται τα πεδία του. Φυσικά μπορεί να καλεί και μεθόδους ξένων αντικειμένων εφόσον έχει κάποιον δείκτη σε αυτά.

ΣΗΜΕΙΩΣΗ

Εδώ είδαμε ένα πολύ σπουδαίο χαρακτηριστικό του OOP. Αυτό είναι το να μπορεί να κρατά τα δεδομένα του κρυφά από τα άλλα αντικείμενα αλλά και να προσφέρει μεθόδους με τις οποίες μπορούν άλλα αντικείμενα να επικοινωνούν και να αλληλεπιδρούν μαζί του. Επιπλέον ο κώδικας που υλοποιεί αυτές τις μεθόδους είναι άγνωστος σε άλλες κλάσεις ή Αντικείμενα. Έτσι έχουμε την δημιουργία ενός interface επικοινωνίας ανεξάρτητου από την υλοποίηση και την εσωτερική δομή του αντικειμένου. Αυτό λέγεται implementation hiding που είναι μία μορφή data abstraction.

4.6) Τύποι δεδομένων στην Java

Οι τύποι δεδομένων στην JAVA είναι σαν αυτούς της C με μόνη διαφορά ότι το μέγεθός τους (σε bytes) είναι γνωστό και ίδιο σε όλες τις υλοποιήσεις της Java.

Τύπος	Όνομα	Μέγεθος
Byte	Byte	8-bit signed, -128..127
Short	Short	16-bit signed
Int	Ακέραιος	32-bit signed
Long	εκτεταμένος ακέραιος	64-bit signed
Float	πραγματικός (κινητής. υποδιαστολής)	32-bit signed
Double	διπλής ακρίβειας	64-bit signed
Char	unicode character	16-bit
Boolean	Boolean	true or false

4.7) Αναγνωριστικά, σταθερές,σχόλια και διαχωριστές

Τα αναγνωριστικά στην JAVA είναι οποιαδήποτε ακολουθία των χαρακτήρων A..Z, a..,z, 0..9, _, $, η οποία ξεκινά με χαρακτήρα γράμμα (κεφαλαίο ή μικρό) ή _ ή $, αλλά όχι με ψηφίο (0..9). Επίσης δεν θεωρούνται αναγνωριστικά οι δεσμευμένες λέξεις της JAVA. Οι σταθερές στην JAVA είναι πολλών τύπων, πχ αριθμητικά, λογικά, χαρακτήρων κλπ.

Μερικά παραδείγματα φαίνονται παρακάτω :

Integer literals (ακαίρεοι αριθμοί)

```
324             // decimal
0xABCD          // hexadecimal
032             // octal
2L              // long decimal integer
```

Boolean literals (λογικά δεδομένα)

```
true, false             // boolean
```

String literals (Συμβολοσειρές)

```
""                  \\ the empty string
"\""
"This is a string"
"This is a \
two-line string"
```

Τα σχόλια στην JAVA υποδηλώνονται με τρεις τρόπους.

1. Με ζεύγος από /* και */ όπως στην C.

2. Με // όπου το κείμενο από το // έως το τέλος της τρέχουσας γραμμής θεωρείται ως σχόλιο

3. Με ζεύγος /** και */. Αυτό είναι πανομοιότυπο με το /* και */ αλλά χρησιμοποιείται από το javadoc για δημιουργία documentation.

Διαχωριστικά (separators) στη JAVA είναι οι χαρακτήρες :

+ - ! % ^ & * | ~ / > <

( ) { } [ ] ; ? :,. =

και επίσης το κενό (SPACE), ο οριζόντιος στηλογνώμονας HT ή \t, η αλλαγή γραμμής LF ή \n καθώς και τα σχόλια.

4.8) Τελεστές

Οι αριθμητικοί τελεστές είναι οι ακόλουθοι:

Τελεστής	Περιγραφή
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
%	Υπόλοιπο
^	Ύψωση σε δύναμη

Οι συσχετιστικοί τελεστές είναι οι ακόλουθοι:

Τελεστής	Περιγραφή
<=	Μικρότερο ή ίσο
<	Μικρότερο
>=	Μεγαλύτερο ή ίσο
>	Μεγαλύτερο

Οι τελεστές ισότητας είναι οι ακόλουθοι:

Τελεστής	Περιγραφή
!=	Άνισο με
==	Ίσο με ==

Οι λογικοί τελεστές είναι οι ακόλουθοι:

Τελεστής	Περιγραφή
&&	ΚΑΙ (AND)
||	Η (OR)
!	ΟΧΙ (NOT)

Τους συσχετιστικούς τελεστές, τους τελεστές ισότητας και τους λογικούς τελεστές τους συναντάμε κυρίως στις δομές ελέγχου (βλέπε παραπάνω). Οι παραπάνω τελεστές χρησιμοποιούνται για συγκρίσεις μεταξύ αριθμών. Εάν η σύγκριση είναι αληθής τότε το αποτέλεσμα είναι η τιμή **true** διαφορετικά εάν είναι ψευδής τότε το αποτέλεσμα είναι η τιμή **false**.

Ο τελεστής αντιστοίχησης είναι ο:

Τελεστής	Περιγραφή
=	Τελεστής αντιστοίχησης

Ο τελεστής αύξησης και ο τελεστής μείωσης είναι οι ακόλουθοι:

Τελεστής	Περιγραφή
++	αύξηση κατά 1
--	μείωση κατά 1

4.9) Εντολές και Δομές ελέγχου

Στη JAVA μπορούμε να κάνουμε δηλώσεις μεταβλητών όχι μόνο στην αρχή ενός block αλλά και σε οποιοδήποτε σημείο του, αρκεί η δήλωση να γίνει πριν από τη χρήση της εν λόγω μεταβλητής. Όσον, τώρα, αφορά τις δομές ροής ελέγχου (if - else, for, while, do - while και switch), οι μόνες αλλαγές που έχουν γίνει σε σχέση με τη C είναι το ότι στο if-else, while, do-while οι εκφράσεις που αποτιμώνται προκειμένου να γίνει άλμα ή επανάληψη, θα πρέπει να είναι boolean και όχι αριθμητικές. Δηλαδή :

while (1) {...} // ΕΙΝΑΙ ΛΑΘΟΣ

while (true) {....} // ΕΙΝΑΙ ΣΩΣΤΟ

Υπάρχουν γενικά δύο ήδη δομών ελέγχου ροής (control flow):

- Οι δομές επιλογής και

• Οι δομές επανάληψης

Ο ακόλουθος πίνακας συνοψίζει τις δομές για τη Java

Είδος δομής ελέγχου ροής	Δομή ελέγχου ροής
Δομές επιλογής	if-else
	switch-case
Δομές επανάληψης	for
	while
	do-while

4.10) Threads

Πολλές γλώσσες προγραμματισμού όπως και η Java διαθέτουν εργαλεία για την υλοποίηση threads στα προγράμματά τους. Αυτές οι γλώσσες καλούνται multithreading languages. Στον παραδοσιακό προγραμματισμό όταν ένα πρόγραμμα εκτελείται ονομάζεται process (διεργασία) και οι εντολές του εκτελούνται σειριακά η μία μετά την άλλη μέχρι το τέλος του. Σ' αυτή την περίπτωση μπορούμε να πούμε ότι το συγκεκριμένο process διαθέτει μόνο ένα thread που εκτελεί τις εντολές του προγράμματος. Στις multithreading γλώσσες προγραμματισμού μπορούμε να ορίσουμε σε κάποιο process να περιέχει ένα ή και περισσότερα threads (ονομάζονται και lightweight processes) οι οποίες εκτελούν κάθε μια ξεχωριστά τον κώδικα του προγράμματος. Είναι δηλαδή σαν να έχουμε πολλά processes που εκτελούνται παράλληλα μέσα στο αρχικό process.

Υλοποίηση των Threads

Οι ενέργειες που λαμβάνουν χώρα κατά την εκτέλεση ενός Thread ορίζονται όπως αναφέραμε και παραπάνω στη μέθοδο run του Thread. Μετά από την δημιουργία και την αρχικοποίηση ενός Thread το σύστημα εκτελεί αυτόματα τον κώδικα που υπάρχει στη μέθοδο run. Συνήθως η μέθοδος αυτή περιέχει κάποια επαναληπτική δομή (πχ στην περίπτωση ενός animation θα έχουμε κάποιο loop που θα εμφανίζει διαδοχικά κάποιες εικόνες).

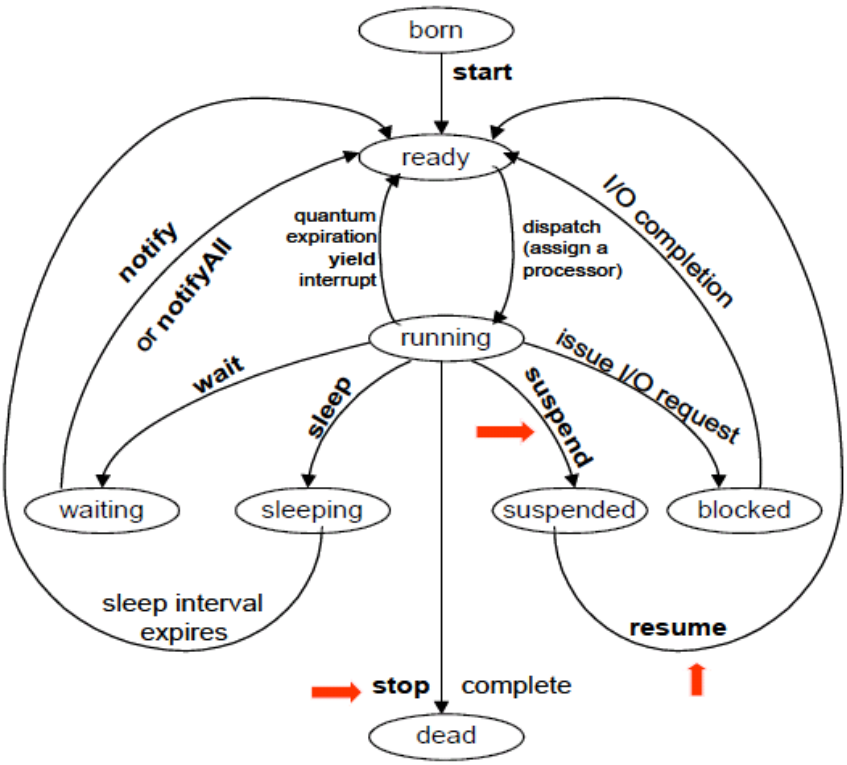
Υπάρχουν δυο τρόποι για τον ορισμό της run μεθόδου κάποιας Thread που δημιουργούμε:

• Δημιουργία υποκλάσης της κλάσης Thread και υπερκάλυψη (overriding) της run μεθόδου που υπάρχει στην Thread κλάση. Παράδειγμα η κλάση simpleThread που δώθηκε στο προηγούμενο παράδειγμα.

• Δημιουργία κλάσης που υλοποιεί το Runnable interface. Σ΄ αυτή την περίπτωση όταν δημιουργούμε την Thread πρέπει να της δώσουμε και μια αναφορά-δείκτη σε στιγμιότυπο της κλάσης που υλοποιεί το Runnable interface.

Οι καταστάσεις των Threads

Το παρακάτω παράδειγμα δείχνει τις καταστάσεις στις οποίες μπορεί να βρίσκεται κάποιο Thread και τις μεθόδους με τις οποίες μπορεί να πηγαίνει από μια κατάσταση σε μια άλλη. Οι μέθοδοι stop, suspend και resume είναι μέθοδοι που υπάρχουν αλλά δεν θα πρέπει να Χρησιμοποιούνται (deprecated) γιατί είναι δυνατόν να οδηγήσουν σε προβλήματα. Στη θέση αυτών των τριών μεθόδων θα πρέπει να υλοποιούμε δικές μας οι οποίες να αλλάζουν κάποια μεταβλητή σημαία (flag) που να δηλώνει την κατάσταση του Thread και χειριζόμαστε κατόπιν αυτές τις καταστάσεις ανάλογα με την περίσταση.

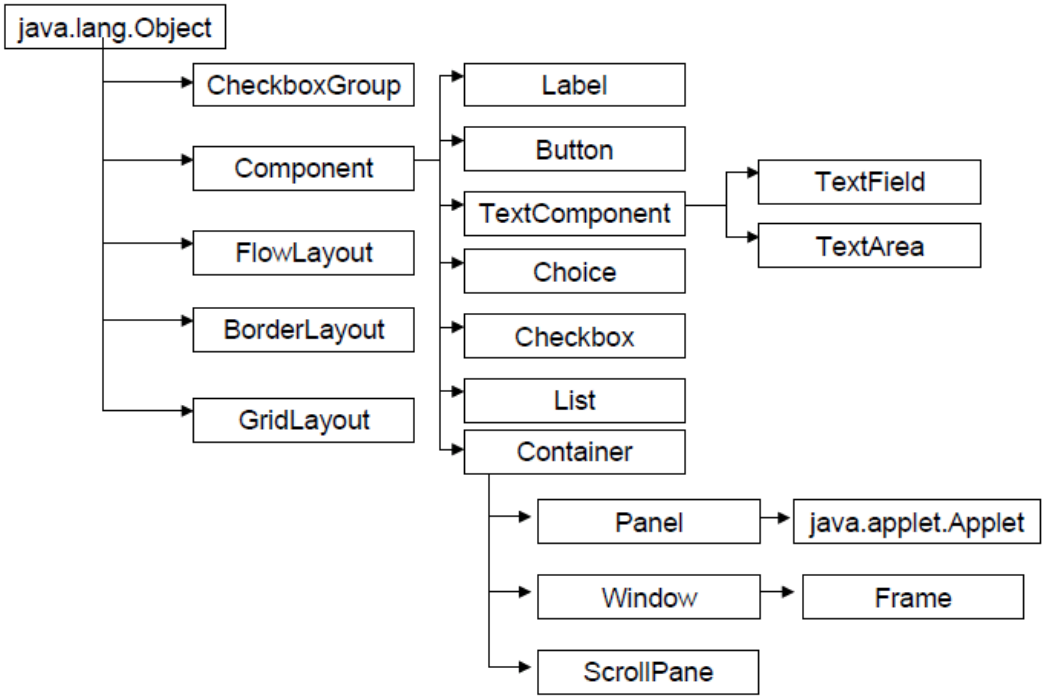


4.11) Βασικά στοιχεία ενός GUI

Ένα Graphical User Interface-GUI (στα ελληνικά Γραφική Διεπαφή με το Χρήστη) είναι το μέρος του προγράμματος, που φροντίζει για τον τρόπο εμφάνισης και χειρισμού του προγράμματος από τον χρήστη. Ένα GUI αποτελείται από GUI-components. Οι κλάσεις που χρησιμοποιούνται για την κατασκευή αντικειμένων τύπου GUI-components ανήκουν στο java.awt (Abstract Windowing Toolkit) package. Οι βασικότερες από αυτές είναι η κλάση Component και η κλάση Container. Κάθε κλάση που κληρονομεί την κλάση Component είναι και αυτή ένα Component. Επίσης κάθε κλάση που κληρονομεί την κλάση Container είναι και αυτή ένα Container. Οι κλάσεις που συνηθέστερα χρησιμοποιούνται για την κατασκευή GUI-components περιγράφονται παρακάτω:

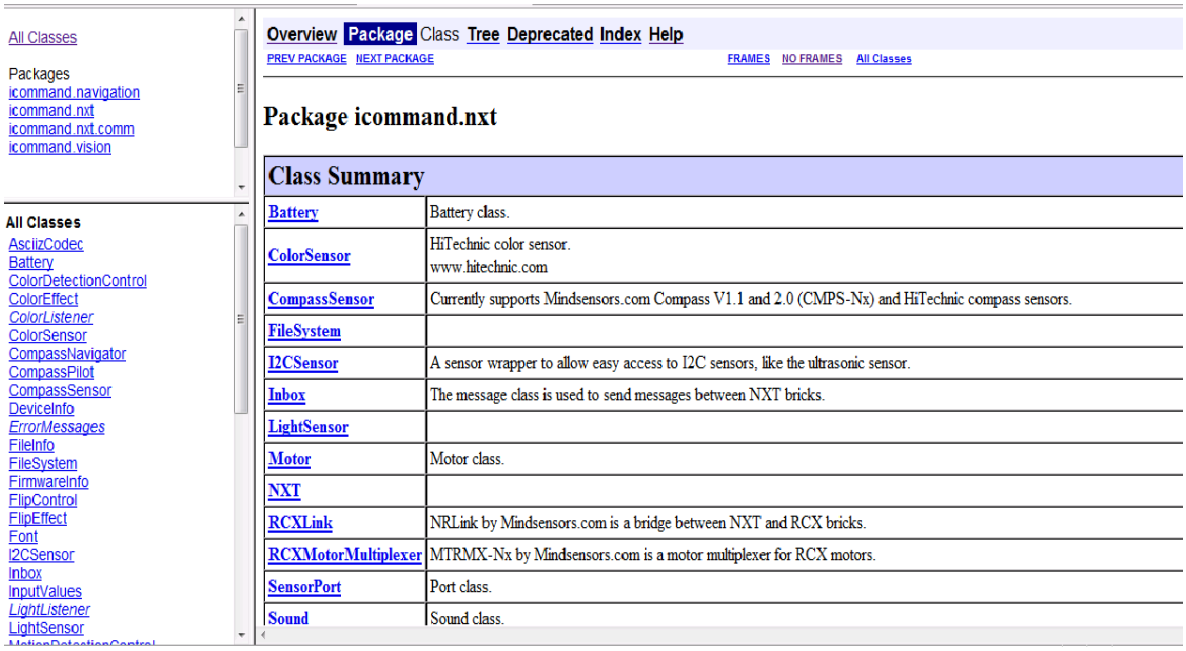
Label	Εμφανίζει κείμενο που δεν μπορεί να τροποποιηθεί από τον χρήστη.
Button	Περιοχή που προξενεί ένα γεγονός (event) όταν επιλέγεται με το ποντίκι.
TextField	Περιοχή όπου ο χρήστης εισάγει δεδομένα από το πληκτρολόγιο. Σε ένα TextField μπορούμε επίσης να εμφανίζουμε πληροφορίες.
TextArea	Περιοχή όπου ο χρήστης εισάγει δεδομένα από το πληκτρολόγιο. Σε ένα TextArea μπορούμε να έχουμε πολλές γραμμές κειμένου σε αντίθεση με το TextField που μπορούμε να έχουμε μόνο μία.
Choice	Μια λίστα στοιχείων από τα οποία ο χρήστης μπορεί να επιλέξει ένα από αυτά με το ποντίκι.
Checkbox	Ένα boolean component που είναι επιλεγμένο ή μή επιλεγμένο. Με αυτό υλοποιούνται και τα Radio buttons (αμοιβαίως αποκλειόμενες επιλογές).
List	Λίστα στοιχείων από τα οποία ο χρήστης μπορεί να επιλέξει ένα από αυτά με το ποντίκι. Διπλό κλικ σε ένα στοιχείο της λίστας προξενεί ένα action event.
Panel	Είναι ένα container αντικείμενο στο οποίο μπορούν να τοποθετηθούν component αντικείμενα.
ScrollPane	Είναι ένα container αντικείμενο στο οποίο μπορεί να τοποθετηθεί ένα component, συνήθως ένα Panel, το οποίο θα εμφανίζεται στον χρήστη αλλά κι όταν αυτό δεν είναι εφικτό θα εμφανίζονται Scrollbars που θα επιτρέψουν την διολίσθηση του component έτσι ώστε να γίνεται ορατό και το τμήμα του που δεν φαίνεται.

Το επόμενο σχήμα εμφανίζει την ιεραρχία των GUI- components κλάσεων μαζί με κάποιες επιπλέον συμπληρωματικές κλάσεις που διαχειρίζονται ένα GUI. Στην κλάση Component ορίζονται οι κοινές μέθοδοι όλων των υποκλάσεων της.



4.12) Η Βιβλιοθήκη I-Command

Η βιβλιοθήκη icommand εμπεριέχει εντολές οι οποίες είναι βασισμένες στην γλώσσα προγραμματισμού Java άρα λοιπόν ξέροντας τις εντολές τις βιβλιοθήκης μπορούμε να εποικοινωνήσουμε με το ρομπότ και να ελέγχουμε τις εισόδους του και τις εξόδους του. Δηλαδή έχουμε την δυνατότητα να ελέγχουμε τους αισθητήρες (φωτός, τον υπερηχητικό, ήχου και τον επαφής) και τα μοτέρ δίνοντας σε αυτά συγκριμένες εντολές και αποκτώντας έτσι τον έλεγχος κίνησης του ρομπότ.



Σχήμα-40 Η βιβλιοθήκη i-command

Τώρα για να μπορέσει η βιβλιοθήκη i-command να επικοινωνήσει με το Bluetooth του υπολογιστή και μετά με το αντίστοιχο του Nxt ρομπότ θα πρέπει να εισάγουμε ακόμα μια βιβλιοθήκη την BlueCove (με την εφαρμογή JSR-82). Υποστηρίζει Mac OS, BlueSoleil και Microsoft Bluetooth το οποίο το βρίσκουμε στα Windows XP SP2 ή Windows Vista. Τέλος θα πρέπει να εισάγουμε το JDK (java development kit) όπου αποτελείται από τον java μεταγλωττιστή και σχετικών εργαλείων που επιτρέπουν στον χρήστη την δυνατότητα να δημιουργεί εφαρμογές σε java.

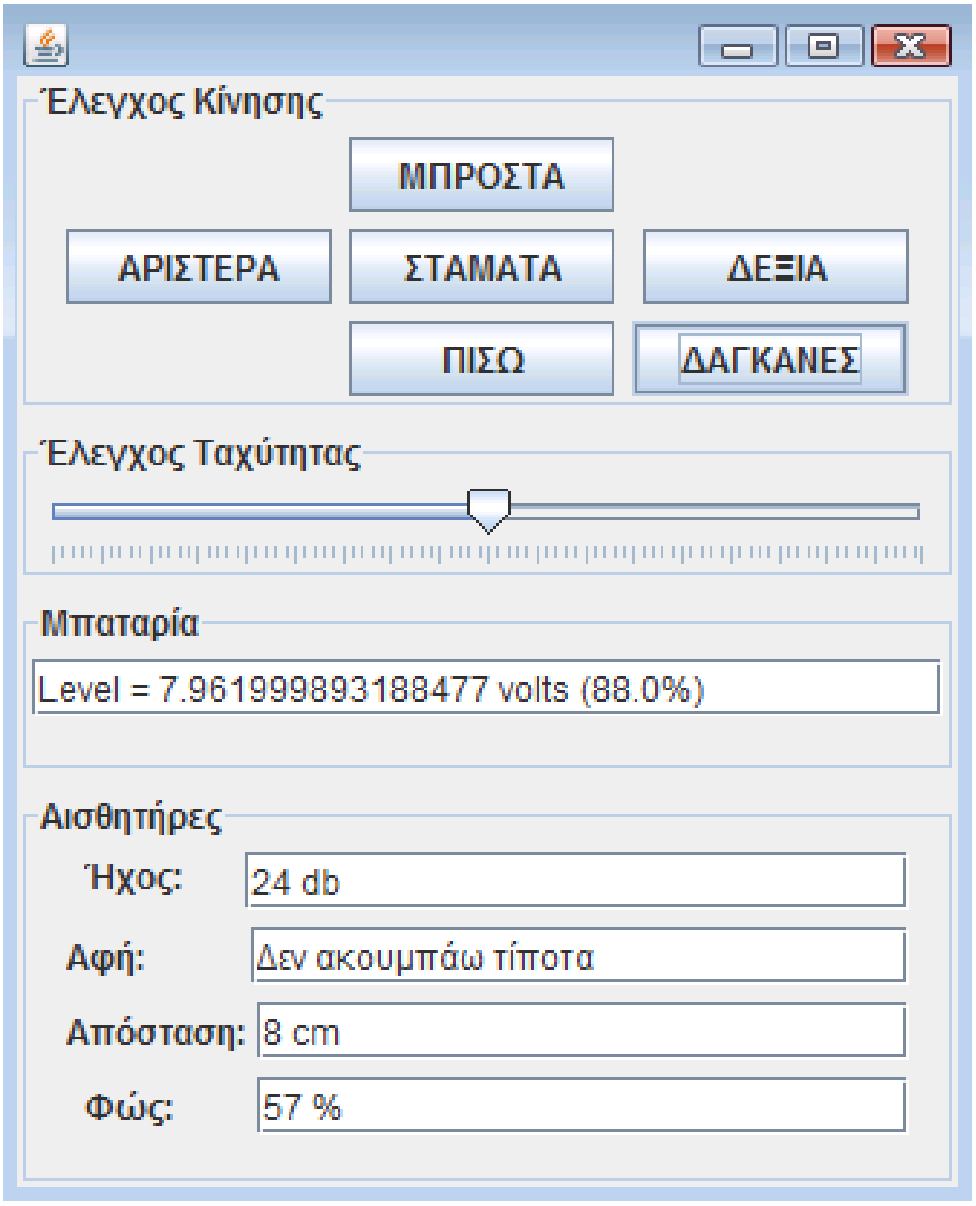
4.13) Αναφορά και διαδικασία υλοποίησης Προγράμματος στην Java

Θα κατασκευάσουμε ένα Panel μέσω του οποίου σε πρώτο στάδιο θα έχουμε πλήρη έλεγχο των 4 εισόδων του ρομπότ και σε Real Time απεικόνιση τους.

- Δηλαδή απεικονίζουμε τις τιμές ήχου σε dB στο εκάστοτε περιβάλλοντος που βρίσκεται το ρομπότ με την βοήθεια του αισθητήρα ήχου .
- Απεικονίζουμε τις τιμές τις μπαταριάς του ρομπότ NXT σε volts και επί της (%)
- Απεικονίζουμε την τιμή % του φωτός που «βλέπει» ο αισθητήρας φωτός όταν παίρνει το NXT ρομπότ πάνω από χρώματα .
- Απεικονίζουμε την απόσταση που τυγχάνει να βρίσκεται ένα αντικείμενο από το ρομπότ σε « cm » με την βοήθεια του υπερηχητικού αισθητήρα .
- Ακόμα με την βοήθεια του αισθητήρα αφής δίνουμε σαν έξοδο στο πρόγραμμα μας αν ακουμπήσει κάτι το ρομπότ μας ή όχι έτσι καταλαβαίνουμε αν έχει βρει κάποιο εμπόδιο μπροστά του. Με την βοήθεια του αισθητήρα αφής δημιουργήσαμε μια εφαρμογή συμφώνα με την οποία ελέγχουμε τους δυο κινητήρες που μας δημιουργούν την κίνηση και την ταχύτητα των κινητήρων αυτών του ρομπότ NXT ο τρίτος κινητήρας είναι συνδεδεμένος με δυο δαγκάνες όποτε αρχικά οι δαγκάνες είναι ανοικτές. Το καθοδηγούμε λυπών και το κατευθύνουμε πάνω σε μια μπάλα η οποία βρίσκεται σε οποιοδήποτε σημείο στο χώρο μόλις το ρομποτάκι πλησιάσει την μπάλα θα πατηθεί ο αισθητήρα αφής θα δώσει την λογική τιμή «1» και στο Panel θα εμφανιστεί στον αισθητήρα αφής ότι το ρομποτάκι μας «ακουμπάει κάτι» εκείνη την στιγμή θα γυρίσει το μοτέρ που είναι συνδεμένες οι δαγκάνες αντίστροφα για να κλείσουν και να αιχμαλωτίσουν το μπαλάκι. Εφόσον τώρα έχουμε πιάσει το μπαλάκι το μεταφέρουμε σε οποίο σημείο θέλουμε εμείς ελέγχοντας από το Panel τα μοτέρ κινήσεις του ρομπότ και πατώντας το κουμπί δαγκάνες που έχουμε στο Panel το μοτέρ που χειρίζεται τις δαγκάνες γυρίζει μπρος τα μπροστά και απελευθερώνεται το μπαλάκι.
- Όπως έγινε κατανοητό και με παραπάνω λεγόμενα μπορούμε να ελίξουμε την κίνηση του ρομπότ μας .Αυτό το καταφέρνουμε έχοντας τοποθετήσει τα κουμπιά «Μπροστά», «Πίσω», «Δεξιά», «Αριστερά» και «Σταματά». Τώρα για πετύχουμε πιο συγκεκριμένη και πιο λεπτομερή μετακίνηση του ρομπότ έπρεπε να ελέγχουμε

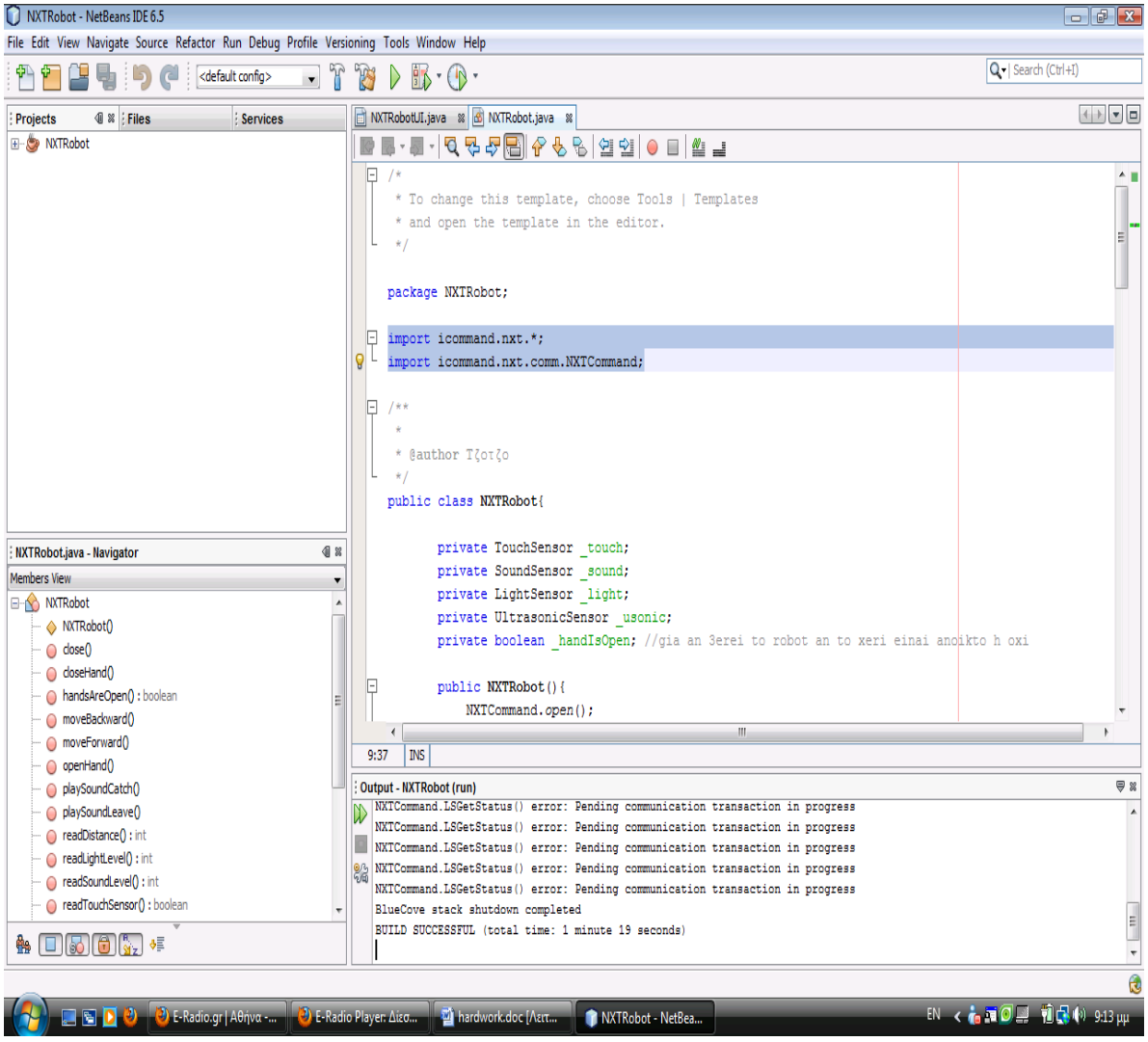
την ταχύτητα σε Real Time, έτσι λοιπόν κατασκευάσαμε μια μπάρα κύλισης που μας επιτρέπει την ρύθμιση της ταχύτητας στα μοτέρ κατά την κίνηση τους.

Στο παρακάτω σχήμα-41 σας παρουσιάζουμε το Panel ελέγχου του ρομπότ Nxt.



Σχήμα-41

Ας δούμε τώρα την διαδικασία υλοποίησης της εφαρμογής από την σκοπιά του κώδικα. Τον κώδικα το γραφούμε με την βοήθεια του προγράμματος NetBeans. Μας επιτρέπει να εισάγουμε τις βιβλιοθήκες που αναφέραμε λεπτομερώς πιο πάνω και να συντάξουμε κώδικα στην Java.



Σχήμα-42. Το interface του προγράμματος NetBeans.

(Η γραμματοσειρά με το πράσινο χρώμα είναι σχολιασμός του προγράμματος, στην ουσία μας βοηθάει να κατανοήσουμε τη λειτουργία των εντολών που προηγούνται)

Import icommand.nxt.*;

```
Import icommand.nxt.comm.NXTCommand;

//Με της δυο παραπάνω εντολές εισάγουμε την βιβλιοθήκη icommand στο NetBeans

 public class NXTRobot{         // ορίζουμε την κλάση   NXTRobot

private TouchSensor _touch;   //  ορίζουμε σαν αντικείμενο  τον αισθητήρα αφής

 private SoundSensor _sound;  // ορίζουμε σαν αντικείμενο  τον αισθητήρα ήχου

private LightSensor _light;                // ορίζουμε σαν αντικείμενο  τον αισθητήρα  φωτός

private UltrasonicSensor _usonic; // ορίζουμε σαν αντικείμενο  τον αισθητήρα υπέρηχων.

 private boolean _handIsOpen;           // ορίζουμε την μεταβλητή για να ξέρει το ρομπότ αν το χέρι είναι ανοικτό η όχι .

public NXTRobot(){

        NXTCommand.open();

        _touch = new TouchSensor(SensorPort.S1); //σύνδεση με τον αισθητήρα αφής στην πόρτας εισόδου 1

        _sound = new SoundSensor(SensorPort.S2); // σύνδεση με τον αισθητήρα ήχου στην πόρτας εισόδου 2

_light = new LightSensor(SensorPort.S3); // σύνδεση με τον αισθητήρα φωτός στην πόρτας εισόδου 3

   _usonic = new UltrasonicSensor(SensorPort.S4); // σύνδεση με τον αισθητήρα υπέρηχων στην πορτας εισοδου 4

    _handIsOpen = true; // θεωρούμε ότι αρχικά τα «χέρια» είναι ανοικτά

public void setSpeed(int s){   // θέτουμε μια μεταβλητή   S για ελέγχουμε την ταχύτητα με την μπάρα κύλισης που έχουμε στο interface του προγράμματος
```

```
        Motor.B.setSpeed(s);  // στο ρομπότ έχουμε συνδέσει  το μοτέρ στην έξοδο «B» για να κινηθεί και του οριοθετούμε την μεταβλητή S ως τιμή κίνησης του

        Motor.C.setSpeed(s); //στο ρομπότ έχουμε συνδέσει  το μοτέρ στην έξοδο «C» για να κινηθεί και του οριοθετούμε την μεταβλητή S ως τιμή κίνησης του

    }

  public void moveForward(){ // Για να κινηθεί το ρομπότ προς τα εμπρός

        Motor.B.forward(); // θέτουμε στο μοτέρ «B» να κινηθεί μπροστά

        Motor.C.forward(); // θέτουμε στο μοτέρ «C» να κινηθεί μπροστά

    }

public void moveBackward(){ // Για να μπορέσει το ρομπότ να κινηθεί προς τα πίσω

        Motor.B.backward();  // Θέτουμε στο μοτέρ «B» να κινηθεί προς τα πίσω

        Motor.C.backward(); // Θέτουμε στο μοτέρ «C» να κινηθεί προς τα πίσω

    }

    public void turnLeft(){ // Για να μπορέσει το ρομπότ να κινηθεί προς τα αριστερά

        Motor.B.forward();  //Θέτουμε στο μοτέρ «B» να κινηθεί προς τα μπροστά

       Motor.C.backward();  // Θέτουμε στο μοτέρ «C» να κινηθεί προς τα πίσω

    }

    public void turnRight(){ // Για να μπορέσει το ρομπότ να κινηθεί προς τα δεξιά

        Motor.B.backward();  //Θέτουμε στο μοτέρ «B» να κινηθεί προς τα πίσω

        Motor.C.forward();  // Θέτουμε στο μοτέρ «C» να κινηθεί προς τα μπροστά

    }

    public void stop(){  //Για να μπορέσει το ρομπότ να σταματήσει
```

```
        Motor.B.stop(); //Θέτουμε στο μοτέρ «B» να σταματήσει να κινείτε
        Motor.C.stop(); //Θέτουμε στο μοτέρ «C» σταματήσει να κινείτε
    }
public void openHand(){ //Εντολή για το να ανοίξει το «χέρι» το ρομπότ
        Motor.A.setSpeed(200);  // Το μοτέρ Α είναι συνδεμένο με τα δυο «χέρια» και του δίνουμε την τιμή 200 για την ταχύτητα που θα ανοίξει τα χεριά
        Motor.A.forward();          // Το μοτέρ πρέπει να κινηθεί προς τα μπροστά για να ξεκινήσουν ανοίξουν τα «χέρια»
        try{
           Thread.sleep(800);      //Του λεμέ να περιμένει για  (0.8 ms)
        }catch(Exception e){
           System.out.println(e);
        }
        Motor.A.stop();          // Του λεμέ να σταματήσει να ανοίγει τα «χέρια» 0.8 ms
        _handIsOpen = true;     // Τα  «χέρια» είναι πλέον ανοικτά
         }
    public void closeHand(){ //Εντολή για το να κλείνει το «χέρι» το ρομπότ
         Motor.A.setSpeed(200); // Το μοτέρ Α είναι συνδεμένο με τα δυο «χέρια» και του δίνουμε την τιμή 200 για την ταχύτητα που θα κλείσει τα χεριά
        Motor.A.backward();  // Το μοτέρ πρέπει να κινηθεί προς τα πίσω για να ξεκινήσουν κλείσουν τα «χέρια»
        try{
```

```
            Thread.sleep(800);   // Του λεμέ να περιμένει για  (0.8 ms)
        }catch(Exception e){
            System.out.println(e);
        }
        Motor.A.stop();     // Του λέμε να σταματήσει να ανοίγει τα «χέρια» 0.8 ms
        _handIsOpen = false;    // Τα  «χέρια» είναι πλέον κλειστά
    }
    public void close(){
        NXTCommand.close();
    }
    public boolean readTouchSensor(){  //Μέθοδος για να διαβάζουμε τον αισθητήρα αφής true or false
        return _touch.isPressed();
    }
    public int readSoundLevel(){      //Μέθοδος για να διαβάζουμε τον αισθητήρα ήχου
return _sound.getdB();                  // Μας γυρίζει τον ήχο που υπάρχει στον χώρο σε dB
    }
    public int readLightLevel(){       //Μέθοδος για να διαβάζουμε τον αισθητήρα αφής
        return _light.getLightPercent();
    }
    public int readDistance(){  //Μέθοδος για να διαβάζουμε τον αισθητήρα υπέρηχων
        return _usonic.getDistance();
```

```
    }
    public void playSoundCatch(){              //Μέθοδος για να παίζει ήχο το ρομπότ μας
       Sound.playSoundFile("woops.rso"); // βρες τον συγκεκριμένο ήχο woops
    }
    public void playSoundLeave(){   //Μέθοδος για να παίξει τον ήχο το ρομπότ μας
       Sound.playSoundFile("woops.rso");
    }
}
public NXTRobotUI() {
    movement = STOP;       // Αρχικά με το που τρεχουμε το πανελ με τις εντολες μας λεμε στο ρομποτ να μην κινειται
    lego = new NXTRobot();
    initComponents();          //Οριοθέτηση των αντικειμένων μέσα στη Java
    rb = new ReadRobot(this);
 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {    // Μέθοδος που πατώντας το κουμπί Μπροστά, πάει μπροστά το ρομπότ
    lego.moveForward();
    movement = FORWARD; // Εντολή για να κινηθεί μπροστά το ρομπότ
}
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  // Μέθοδος που πατώντας το κουμπί Σταμάτα, το ρομπότ σταματάει
   lego.stop();
```

```
        movement = STOP; // Εντολή για να σταματήσει το ρομπότ
}
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) { //  Μέθοδος που πατώντας το κουμπί Δεξιά, το ρομπότ πάει προς τα δεξιά
        lego.turnRight();
        movement = RIGHT; // Εντολή για να κινηθεί δεξιά το ρομπότ
  }
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) { //  Μέθοδος που πατώντας το κουμπί Αριστερά, το ρομπότ πάει προς τα Αριστερά
      lego.turnLeft();
      movement = LEFT; // Εντολή για να κινηθεί αριστερά το ρομπότ
    }
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) { //Μέθοδος που πατώντας το κουμπί Πίσω, το ρομπότ πάει προς τα Πίσω
      lego.moveBackward();
      movement = BACKWARD; // Εντολή για να κινηθεί πίσω το ρομπότ
    }
    private void speedControlStateChanged(javax.swing.event.ChangeEventevt){ //Μέθοδος για να ελέγχουμε την ταχύτητα από την μπάρα κύλισης του Panel
        lego.setSpeed((int)speedControl.getValue()); // Εντολή για τον ελεγχο ταχυτητας με μεταβλητες τιμές
        if(movement == FORWARD){  // Συνθήκη εκεινη την στιγμη που το ρομποτ πηγαινει μπροστα να αυξηθει-μειωθει η ταχύτητα
```

```
            lego.moveForward(); //εντολη για να παει μπροστα το ρομποτ
        }else if(movement == BACKWARD){ ){   // Συνθήκη εκεινη την στιγμη που το ρομποτ πηγαινει  πισω να αυξηθει-μειωθει η ταχύτητα
            lego.moveBackward();  //εντολή για να πάει πίσω το ρομπότ
        }else if(movement == LEFT){ // Συνθήκη εκείνη την στιγμή που το ρομπότ πηγαίνει αριστερά να αυξηθεί-μειωθεί η ταχύτητα
            lego.turnLeft(); //εντολή για να πάει αριστερά το ρομπότ
        }else if(movement == RIGHT){   ){ // Συνθήκη εκείνη την στιγμή που το ρομπότ πηγαίνει  δεξιά να αυξηθεί-μειωθεί η ταχύτητα
            lego.turnRight(); //εντολή για να πάει δεξιά το ρομπότ
        }
    private void OpenHandButtonActionPerformed(java.awt.event.ActionEvent evt) {
// Κουμπι για αν ανοιγει τα «χέρια» του το ρομποτ
        if(!lego.handsAreOpen()){  //Αν τα «χεριια» Δεν είναι ανοικτα
            lego.openHand();  // του λεμε να ανοίξει τα χεριια
            lego.playSoundLeave(); // και μολις ανοιξει τα χεριια παιξε τον ηχο .
        }
}
    public void WriteBatteryLevel(String s){   //Μέθοδος για να εμφανιζουμε τις τιμες της μπαταριας
        batteryTextField.setText(s);
    }
```

```
    public void WriteSoundLevel(String s){ // Μέθοδος για να εμφανιζουμε στο παραθυρο τις τιμές του ηχου
        soundTextField.setText(s);
    }
    public void WriteLightLevel(String s){ // Μέθοδος για να εμφανίζουμε στο παράθυρο τις τιμές της αντανάκλασης του φωτός
        lightTextField.setText(s);
    }
    public void WriteTouchLevel(String s){ //Μέθοδος για να εμφανίζουμε στο παράθυρο τα αποτελέσματα του αισθητήρα αφής
        touchTextField.setText(s);
    }
    public void WriteDistanceLevel(String s){ // Μέθοδος για να εμφανίζουμε στο παράθυρο τις τιμές του αισθητήρα υπέρηχων
        distanceTextField.setText(s);
    }
private NXTRobotUI window;
        public ReadRobot(NXTRobotUI w){ // εισάγουμε το ρομπότ με το που τρέχουμε το Panel
            window = w;
            isRunning = true;
            Thread t = new Thread(this); // Αρχικοποιούμε- δημιουργούμε το Thread
            t.start();  // Εκκίνηση του Thread
```

```
    }
    public void close(){
       isRunning = false;
    }
    public void run(){
       double batteryLevel, x; // Διάβασε την τιμή της μπαταρίας και θέση την ως x
       int soundLevel, lightLevel, distance; // διάβασε τις τιμές στάθμης ήχου, φωτός, απόστασης
       boolean touch; //διάβασε την τιμή του αισθητήρα αφής
        while(isRunning){ // Ενώ υπάρχει επικοινωνία με το ρομπότ
          batteryLevel = Battery.getVoltage(); // Διάβασε την μπαταρία από το ρομπότ(LEGO)
          soundLevel = lego.readSoundLevel(); // Διάβασε τον ήχο από το ρομπότ(LEGO)
          lightLevel = lego.readLightLevel(); // Διάβασε το φως από το ρομπότ(LEGO)
          distance = lego.readDistance(); // Διάβασε την απόσταση από το ρομπότ(LEGO)
          touch = lego.readTouchSensor();  // Διάβασε τον αισθητήρα αφής από το ρομπότ(LEGO)
          if(touch && lego.handsAreOpen()){ //Εάν ακουμπήσω κάτι και τα «χέρια» είναι ανοικτα
             lego.stop();  //Σταμάτα να κινείσαι
             lego.closeHand();  //Κλείσε τα <<χέρια>>
```

```
            lego.playSoundCatch(); //Παίξε τον ήχο
        }
        x = (int)((batteryLevel *100)/9); // Υπολογισμός στάθμης μπαταρίας
        //Εμφάνιση πληροφοριών
        window.WriteBatteryLevel( "Level = "+batteryLevel+" volts ("+x+"%)");
        window.WriteDistanceLevel(distance +" cm");
        window.WriteLightLevel(lightLevel + " %");
        window.WriteSoundLevel(soundLevel + " db");
        if(touch){ //Αν είναι πατημένο το pushbutton
            window.WriteTouchLevel("Ακουμπάω κάτι");
        }else{ //αλλιώς
            window.WriteTouchLevel("Δεν ακουμπάω τίποτα");
        }
        try{
            Thread.sleep(100); // περιμένουμε 100 ms για να ανανεωθούν οι πληροφορίες
        }catch (Exception e){
 System.out.println(e); // Εάν υπάρξει πρόβλημα με το Thread.sleep εμφάνισέ το
}
```

ΚΕΦΑΛΑΙΟ 5^{Ο}

Το Visual Studio με τις VB.net και C sharp(C#)

5.1) Εισαγωγή

Το Visual Studio 2008 είναι ένα εργαλείο ανάπτυξης που μπορούμε να χρησιμοποιήσουμε για να δημιουργήσουμε εφαρμογές οι οποίες θα εκτελούν χρήσιμες εργασίες και θα δείχνουν εντυπωσιακές με την εφαρμογή διάφορων ρυθμίσεων. Χρησιμοποιώντας το Visual Studio 2008, μπορούμε να δημιουργήσουμε εφαρμογές για το λειτουργικό σύστημα Windows, τον Ιστό, φορητές συσκευές και ένα πλήθος άλλων συσκευών και ρυθμίσεων. Το πιο σημαντικό πλεονέκτημα του Visual Studio 2008 είναι ότι έχει σχεδιασθεί για να κάνει ακόμη πιο παραγωγική την εργασία μας ειδικά αν χρησιμοποιούμε πληροφορίες σε βάσεις δεδομένων.

5.2.1) Το .NET Framework

Για να κατανοήσουμε τον όρο Visual Basic.NET θα πρέπει αρχικά να αναλύσουμε τον όρο <<.NET>>. Από μόνος του αυτός ο όρος, δεν σημαίνει και πολλά. Θα μπορούσαμε να ρωτήσουμε δέκα διαφορετικούς ανθρώπους και να πάρουμε δέκα διαφορετικές απαντήσεις. Ο όρος χρησιμοποιείται ευρέως και με πολλές διαφορετικές σημασίες. Μάλιστα, το .NET έχει χρησιμοποιηθεί και έχει διαφημιστεί υπερβολικά, περίπου όπως και ο όρος mp3. Έτσι, στην πραγματικότητα, όταν ακούμε ή διαβάζουμε τον όρο .NET θα πρέπει να καταλαβαίνουμε ότι εννοούμε το πλαίσιο ανάπτυξης εφαρμογών .NET (.NET Framework).

Ο επίσημος ορισμός του πλαισίου ανάπτυξης εφαρμογών .NET είναι ο εξής:

Το πλαίσιο εφαρμογών .NET είναι μία πλατφόρμα που μας επιτρέπει να δημιουργούμε εφαρμογές λογισμικού και βιβλιοθήκες που ονομάζονται ‘επιβλεπόμενες

βιβλιοθήκες'(managed applications)˙ μας παρέχει μεταγλωττιστή και εργαλεία τα οποία μας δίνουν τη δυνατότητα να κατασκευάζουμε, να απασφαλματώνουμε και να εκτελούμε επιβλεπόμενες εφαρμογές.

Για τους σκοπούς μας μπορούμε να πούμε ότι το .NET είναι η πλατφόρμα που διαθέτει ότι χρειαζόμαστε για να δημιουργούμε και να εκτελούμε επιβλεπόμενες εφαρμογές για τα Windows.

Λέμε ότι οι εφαρμογές είναι επιβλεπόμενες επειδή την εκτέλεσή τους επιβλέπει και διαχειρίζεται το πλαίσιο εφαρμογών .NET. Μάλιστα, το πλαίσιο εφαρμογών .NET διαχειρίζεται την εκτέλεση παρέχοντας ένα ελεγχόμενο περιβάλλον χρόνου εκτέλεσης (runtime environment) το οποίο διαθέτει μια μεγάλη ποικιλία υπηρεσιών όπως φόρτωση των εφαρμογών μας, διαχείριση της μνήμης και τέλος, παρακολούθηση και τήρηση της ασφαλείας και της ακεραιότητας κατά την εκτέλεση της εφαρμογής. Πριν το .NET (και την Java), δε γινόταν επίβλεψη των εφαρμογών επειδή αυτές δεν εκτελούνταν σε ελεγχόμενο περιβάλλον χρόνου εκτέλεσης. Κανένα άλλο συστατικό του συστήματος δεν παρείχε τις υπηρεσίες που προσφέρει το .NET. οι εφαρμογές έπρεπε να διαχειρίζονται τις δικές τους υπηρεσίες, πράγμα που πολλές φορές οδηγούσε σε εσφαλμένο κώδικα, κενά ασφαλείας και αλλοίωση δεδομένων. Λόγω αυτών των προβλημάτων, η συντήρηση και η απασφαλμάτωση των εφαρμογών ήταν δύσκολη.

Το πλαίσιο εφαρμογών .NET μας παρέχει μια μεγάλη ποικιλία εργαλείων όπως μεταγλωττιστές (), αποσφαλματωτές (), γλώσσες προγραμματισμού, μια μηχανή εκτέλεσης CLR (), εργαλεία για τον προγραμματιστή και ένα μεγάλο αριθμό προκαθορισμένων βιβλιοθηκών με δομικά στοιχεία. Αυτές οι βιβλιοθήκες ονομάζονται FCL (Framework Class Libraries/Βιβλιοθήκες Κλάσεων Πλαισίου Εφαρμογών).

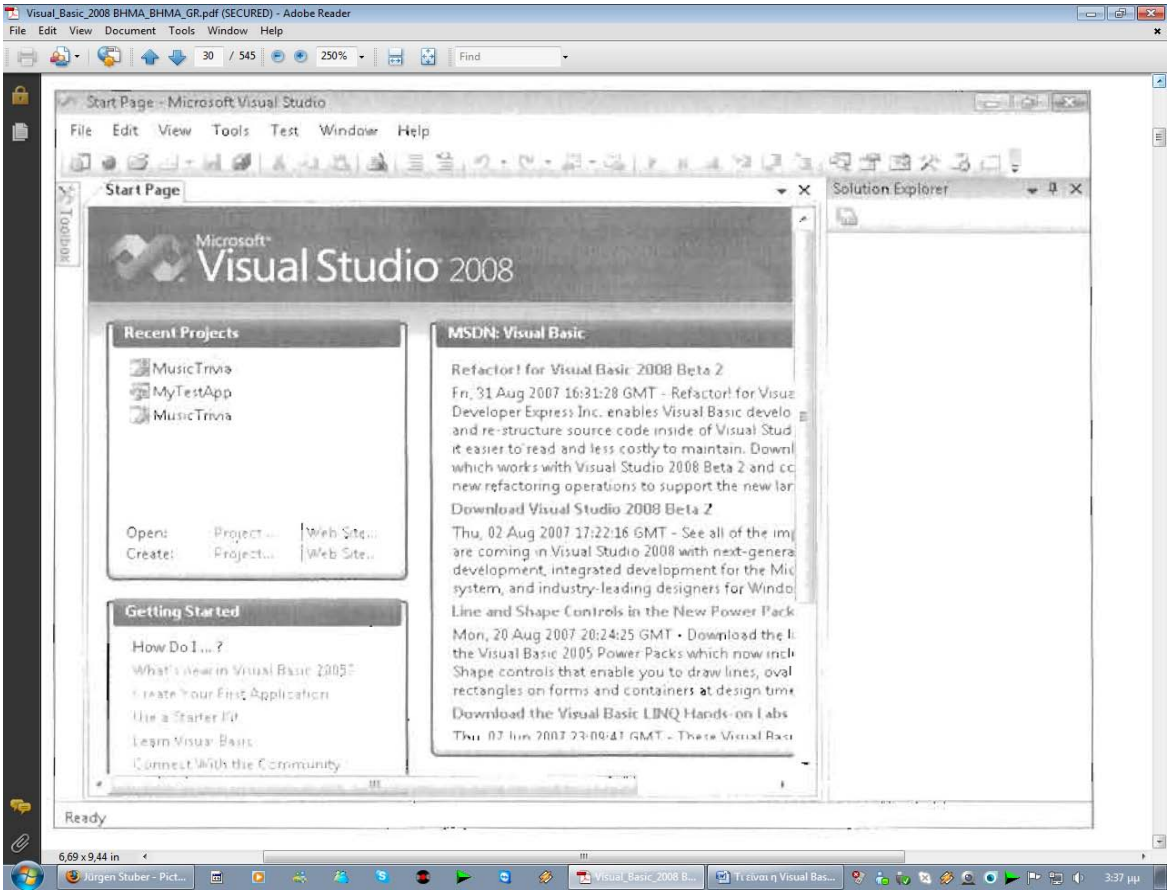
5.2.2) Εκδόσεις της Visual Basic .NET

Η πρώτη έκδοση της VB.net (Microsoft Visual Basic .NET 2002) κυκλοφόρησε το Φεβρουάριο του 2002. Η δεύτερη έκδοση (Microsoft Visual Basic .NET 2003) ήταν ευρέως διαθέσιμη τον Μάρτιο του 2003. Στα τέλη του 2005 κυκλοφόρησε η Visual Basic 2005, ενώ μετά από μία μεγάλη περίοδο εργασιών ανάπτυξης και συνεργασίας η Microsoft κυκλοφόρησε τη Visual Basic 2008 στις αρχές του 2008. Η Visual Basic 2008 εμπεριέχεται στο Visual studio 2008 το οποίο περιέχει εκτός των άλλων τη Visual C#, τη Visual C++, το Visual Web Developer και άλλα εργαλεία ανάπτυξης .NET της Microsoft.

5.2.3) Εξερεύνηση του Visual studio 2008

Στη συνέχεια θα περιγράψουμε τον τρόπο λειτουργίας του Visual Studio 2008 διότι μέσω αυτού θα γράψουμε τα προγράμματά μας στη VB.net και στη C#.

Με το ξεκίνημα του Visual Studio, βλέπουμε στην οθόνη μας το περιβάλλον ανάπτυξης με τα μενού, τα εργαλεία και τα παράθυρα των στοιχείων του. Αρχικά, βλέπουμε μία εισαγωγική σελίδα(start page) με μία ομάδα συνδέσμων, άρθρων του MSDN και επιλογές έργων. Η εισαγωγική σελίδα είναι μία περιεκτική πηγή πληροφοριών σχετικά με το έργο μας, ενώ περιλαμβάνει και πόρους της κοινότητας ανάπτυξης της Visual Basic. Αποτελεί ένα δρόμο για τη λήψη νέων πληροφοριών σχετικά με το Visual Studio



Αυτό το παράθυρο το χρησιμοποιούμε εάν θέλουμε να ανοίξουμε ένα ήδη υπάρχον project. Σε περίπτωση που θέλουμε να δημιουργήσουμε ένα καινούριο project, κλείνουμε το ‘start page’ και ξεκινάμε από την αρχή το καινούριο μας project.

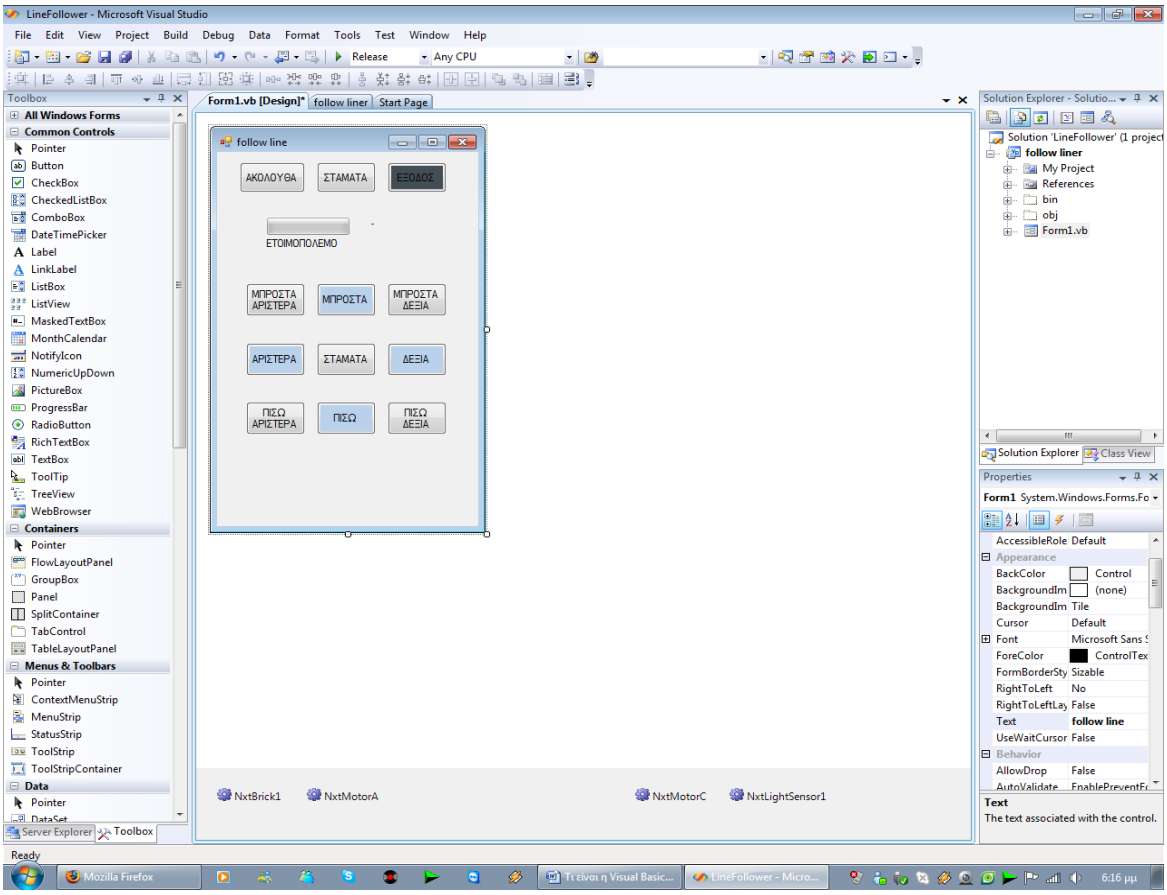
Στη συνέχεια θα αναφερθούμε σε μερικά χρήσιμα προγραμματιστικά εργαλεία του Visual Studio.

Η γραμμή μενού(menu bar) παρέχει πρόσβαση στις περισσότερες από τις διαταγές που ελέγχουν το περιβάλλον ανάπτυξης. Τα μενού και οι διαταγές λειτουργούν όπως και στα υπόλοιπα προγράμματα των Windows, και μπορούμε να τα προσπελαύνουμε χρησιμοποιώντας το πληκτρολόγιο ή το ποντίκι μας. Ακριβώς από κάτω υπάρχει η βασική γραμμή αργαλειών(standard toolbar), μια συλλογή κουμπιών τα οποία χρησιμοποιούνται ως συντομεύσεις για την εκτέλεση διαταγών και τον έλεγχο του περιβάλλοντος ανάπτυξης του Visual Studio.(Είναι παρόμοια με αυτή του Word και του excel όμως με περισσότερες

και πιο εντυπωσιακές δυνατότητες).μπορούμε να δούμε την πλήρη λίστα με τις γραμμές εργαλείων πατώντας δεξί κλικ σε οποιαδήποτε γραμμή εργαλείων.

Κατά μήκος της βάσης της οθόνης υπάρχει η γραμμή εργασιών(taskbar) των Windows. Με την βοήθεια της γραμμής εργασιών μπορούμε να εναλλασσόμαστε μεταξύ διαφόρων στοιχείων του Visual Studio και να ενεργοποιούμε άλλα προγράμματα των Windows. Επίσης, στη γραμμή εργασιών ίσως δείτε και εικονίδια για τον Internet Explorer, για βοηθητικά προγράμματα προστασίας από ιούς και για άλλα προγράμματα που έχουν εγκατασταθεί στο σύστημά μας

Η φόρμα μας (form1) στον ακολουθητή γραμμής (follow liner) στη VB.net:



Τα κύρια εργαλεία που είναι ορατά σε αυτό το περιβάλλον ανάπτυξης του Visual Basic είναι ο Σχεδιαστής (Designer), η Εξερεύνηση Λύσεων (Solution Explorer), το Παράθυρο Ιδιοτήτων (Properties) και η Εργαλειοθήκη (Toolbox). Επίσης, ίσως δούμε και

πιο εξειδικευμένα εργαλεία όπως η Εξερεύνηση Διακοσμητών (Server Explorer) και ο Φυλλομετρητής Αντικειμένων (Object Browser) τα οποία μπορεί να έχουν τη μορφή καρτελών στο περιβάλλον ανάπτυξης. Επειδή οι προτιμήσεις των προγραμματιστών δεν είναι ποτέ ίδιες, είναι δύσκολο να προβληθεί η εμφάνιση του Visual Studio αν αυτό έχει ήδη χρησιμοποιηθεί.

5.3.1) Η βιβλιοθήκη του Bram Fokke στη VB.net

Για να ελέγξουμε το ρομπότ μέσω της τεχνολογίας Bluetooth θα χρησιμοποιήσουμε τη βιβλιοθήκη NXT# του Bram Fokke.

Η NXT# είναι μία βιβλιοθήκη για την πλατφόρμα .NET και μας επιτρέπει να ελέγχουμε το LEGO NXT ρομπότ από προγράμματα γραμμένα σε VB.net και C# μέσω της τεχνολογίας σύνδεσης Bluetooth.

Τα κύρια χαρακτηριστικά της NXT#:

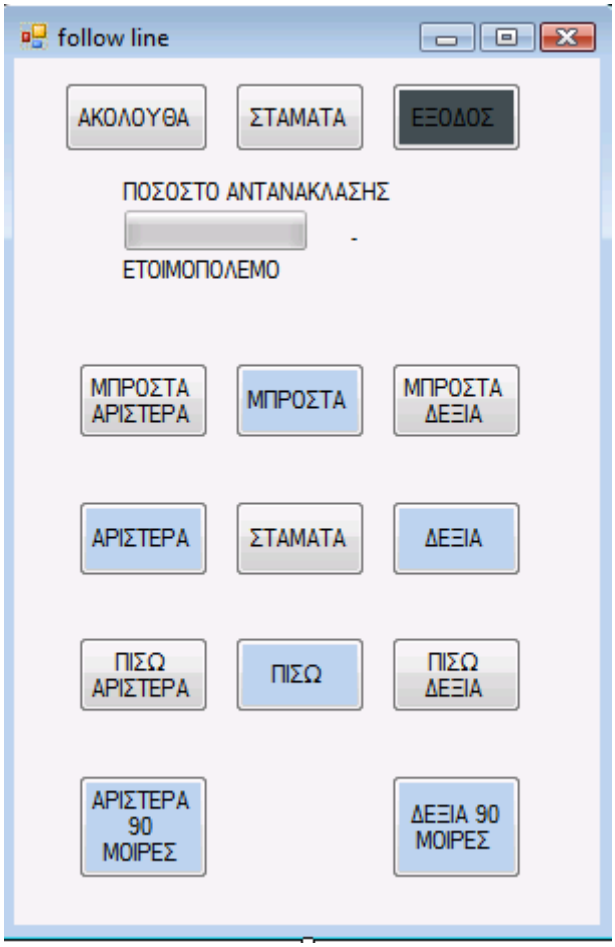
- Λειτουργούμε το πρόγραμμα του LEGO NXT και ελέγχουμε το NXT ρομπότ μέσω Bluetooth άμεσα από την εφαρμογή μας .NET χωρίς περεταίρω ενέργειες
- Περιέχει τα συστατικά NxtBrick, NxtMotor, NxtPressureSensor, NxtLightSensor, NxtSoundSensor και NxtSonar και μας παρέχει ολικό έλεγχο αυτών με απλές εντολές.
- Μας παρέχει το αντικείμενο NxtCommunicator το οποίο μας εκθέτει το άμεσο σύνολο εντολών του LEGO NXT.
- Επίσης, μας παρέχει και το LegoNxtMotorControl το οποίο μας βοηθάει να ελέγχουμε με ευκολία τους σερβοκινητήρες μας

Πως λειτουργεί;

Όταν συνδέεται με τον υπολογιστή, η λειτουργία Bluetooth δημιουργεί μία εικονική COM θύρα που μπορεί να χρησιμοποιηθεί για την επικοινωνία. Το πρωτόκολλο που χρησιμοποιείται για να επικοινωνήσει το NXT ρομπότ είναι τεκμηριωμένο στο Lego Bluetooth SDK. Η NXT# έχει δικές της εντολές με τις οποίες μπορούμε να ελέγξουμε άμεσα τους διάφορους σένσορες και τους σερβοκινητήρες μας.

5.3.2) Ακολουθητής γραμμής στη VB.net

Το πλαίσιο ελέγχου στον ακολουθητή γραμμής είναι το παρακάτω:



Το ρομπότ μας, το ονομάζουμε NxtBrick1 και από εδώ και στο εξής, όταν θα αναφερόμαστε στο ρομπότ, θα το καλούμε έτσι. Το NxtBrick1 το συνδέουμε με τον υπολογιστή με την παρακάτω εντολή:

```
NxtBrick1.Connect()
```

Το πρόγραμμά μας χωρίζεται σε δύο υποπρογράμματα ανεξάρτητα μεταξύ τους.

Το πρώτο αποτελείται από εννέα εντολές, με τις οποίες καθοδηγούμε το NxtBrick1 για να το φέρουμε στη σωστή θέση έτσι ώστε να ξεκινήσει το δεύτερο πρόγραμμα. Στην ουσία, με κάθε εντολή, αυξάνουμε, μειώνουμε ή μηδενίζουμε τις στροφές κάθε

σερβοκινητήρα για να πετύχουμε την εκαστοτε κίνηση. Δηλαδή, για να προχωρήσει εμπρός, με τις δύο παρακάτω εντολές:

```
NxtMotorA.Turn(10, 0)

NxtMotorC.Turn(10, 0)
```

Θέτουμε τους σερβοκινητήρες A και C την ταχύτητα στροφών με τιμή ίση με 10. Με θετικές τιμές, οι σερβοκινητήρες στρίβουν προς τα εμπρός και αντίστοιχα με αρνητικές, προς τα πίσω. Με τιμή ίση με το μηδεν, όπως είναι αναμενόμενο, σταματάνε να στρίβουν χωρίς όμως να φρενάρουν.

Για να πετύχουμε την στροφή μπροστά δεξια, δίνουμε τιμές στους σερβοκινητήρες A και C, 10 και 20 αντίστοιχα:

```
NxtMotorA.Turn(10, 0)

NxtMotorC.Turn(20, 0)
```

Για την επιτόπου στροφή δεξιά, οι τιμές που δίνουμε είναι -7 και 7 για να πετύχουμε αντίστροφη κίνηση των σερβοκινητηρων:

```
NxtMotorA.Turn(-7, 0)

NxtMotorC.Turn(7, 0)
```

Παρομοίως, για πίσω δεξιά:

```
NxtMotorA.Turn(-10, 0)

NxtMotorC.Turn(-20, 0)
```

Πίσω:

```
NxtMotorA.Turn(-10, 0)

NxtMotorC.Turn(-10, 0)
```

Μπροστά αριστερά:

```
NxtMotorA.Turn(20, 0)

NxtMotorC.Turn(10, 0)
```

Επιτόπου αριστερά:

```
NxtMotorA.Turn(7, 0)

NxtMotorC.Turn(-7, 0)
```

Και πίσω αριστερά:

```
NxtMotorA.Turn(-20, 0)

NxtMotorC.Turn(-10, 0)
```

Τέλος, το ΣΤΑΜΑΤΑ, σταματάει την κίνηση στους δύο σερβοκινητήρες.

Το δεύτερο είναι το κυρίως πρόγραμμα, το οποίο χρησιμοποιεί τον light sensor (ανιχνευτής και μετρητής φωτεινότητας) και τους δύο σερβοκινητήρες.

Έχουμε ένα χαρτόνι με μία οβάλ γραμμή (θα μπορούσαμε να έχουμε οποιοδήποτε σχήμα). Ο light sensor μετράει ανα μικρά τακτά χρονικά διαστήματα την ένταση της φωτεινότητας του εδάφους.

Κάθε φορά που μετράται η ένταση της φωτεινότητας, η τιμή συγκρίνεται με τις μεταβλητές `colWhite` και `colBlack` και αναλόγος καταλαβαίνει σε ποια απόχρωση του γκρι είμαστε.

Το ρομπότ ακολουθεί είτε την εσωτερική είτε την εξωτερική πλευρά της γραμμής. Αυτό το καταφέρνει με το να ακολουθεί το γκρι (η περιοχή ανάμεσα στο μαύρο και στο άσπρο τη βλέπει σαν γκρι). Επομένος, μόλις ξεφεύγει από την πορεία του (βλέπει είτε άσπρο είτε μαύρο), στρίβει στην κατάληλη φορά αναλόγως αν προηγουμένος είχε βρει μαύρο.

Το πρόγραμμά μας ολοκληρωμένο είναι το εξής:

```
Imports Bram.Lego

Public Class Form1

    Const colWhite = 52
'Αν χρώμα > colWhite τότε το χρώμα είναι άσπρο
    Const colBlack = 41
'Αν χρώμα < colBlack τότε το χρώμα είναι μαύρο
'Το χρώμμα ανάμεσα στο άσπρο και στο μαύρο είναι γκρι

#Region "Events"

    Dim Active As Boolean
'Συνθήκη αν είναι ενεργό
    Dim BlackFound As Boolean
'Συνθήκη αν βρήκε τη γραμμή

  Private Sub Form1_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
    NxtBrick1.Connect()
  End Sub

  Private Sub Form1_FormClosed(ByVal sender As Object, ByVal
e As System.Windows.Forms.FormClosedEventArgs) Handles
Me.FormClosed
    NxtBrick1.Disconnect()
  End Sub

    'προχωράει
```

```
  Private Sub btnGo_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnGo.Click
    Active = True
  End Sub

    'Σταματάει
  Private Sub btnStop_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnStop.Click
    Active = False
    StopMotor()
    BlackFound = False
  End Sub

    'Έξοδος
  Private Sub btnQuit_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnQuit.Click
    Me.Close()
  End Sub

  Private Sub NxtLightSensor1_ValueChanged(ByVal sensor As
Bram.Lego.NxtSensor) Handles NxtLightSensor1.ValueChanged
        If Not Active Then Exit Sub
'Μην κάνεις τίποτα μέχρι ο χρήστης να πατήσει το ακολούθα
    Dim s As NxtLightSensor = CType(sensor, NxtLightSensor)
    SetLevel(ProgressBar1, s.Value)
    SetText(LightValue, s.Value.ToString)
    'Αρχικά προχώρα μπροστά μέχρι να βρεις μαύρο, μετά
ακολούθα την άκρη της μαύρης γραμμής
    'οπότε πρέπει να ξέρουμε αν βρήκαμε μαύρο
    If s.Value <= colBlack Then BlackFound = True
'Ναι, βρήκαμε, θέσε τον δείκτη
```

```
        'Ακολούθα την άκρη αν βρέθηκε μαύρο
    If BlackFound Then
      FollowEdge(s.Value)
    Else
      GoForward()
    End If
  End Sub

#End Region

#Region "Form Updating"

    'Ανανέωση των αντικειμένων (items) στη φόρτα
  Delegate Sub SetTextCallback(ByVal txtObject As Object,
ByVal txt As String)
  Delegate Sub SetLevelCallback(ByVal Ctrl As Object, ByVal
val As Integer)

    'Ανανέωση της μπάρας προόδου
  Private Sub SetLevel(ByVal Ctrl As Object, ByVal val As
Integer)
        If Ctrl.InvokeRequired Then
            Dim d As New SetLevelCallback(AddressOf SetLevel)
            Me.Invoke(d, Ctrl, val)
        Else
            If val < 0 Then val = 0 Else If val > 100 Then
val = 100
            Ctrl.Value = val
        End If
    End Sub
```

```
  'Ανανέωση του κειμένου(text) στη φόρμα
   Private Sub SetText(ByVal txtObject As Object, ByVal txt
As String)
        If txtObject.InvokeRequired Then
            Dim d As New SetTextCallback(AddressOf SetText)
            Me.Invoke(d, txtObject, txt)
        Else
            txtObject.Text = txt
        End If
    End Sub

#End Region

#Region "Algorithm"

  Sub FollowEdge(ByVal LightValue As Integer)
    Select Case LightValue
                'Αν άσπρο, τότε στρίψε δεξιά
      Case Is > colWhite
        TurnRight()
                SetText(lblStatus, "ΣΤΡΙΒΕΙ ΑΡΙΣΤΕΡΑ")
                'Αν μαύρο, τότε στρίωε αριστερά
      Case Is < colBlack
        TurnLeft()
                SetText(lblStatus, "ΣΤΡΙΒΕΙ ΔΕΞΙΑ")
                'Αν είναι ενδιάμεσο, τότε πήγγαινε ευθεία
      Case Else
        GoForward()
                SetText(lblStatus, "ΠΗΓΑΙΝΕΙ ΕΥΘΕΙΑ")
    End Select
  End Sub
```

```
#End Region

#Region "Nxt Stuff"

    Const Speed = 3 'Ταχύτητα κίνησης
    Const TurnDegrees = 2 'Πόσο απότομα στρίβει

  Sub GoForward(Optional ByVal degrees As Integer = 0)
    NxtMotorA.Turn(Speed, degrees)
    NxtMotorC.Turn(Speed, degrees)
  End Sub

  Sub GoBack(Optional ByVal degrees As Integer = 0)
    NxtMotorA.Turn(-Speed, degrees)
    NxtMotorC.Turn(-Speed, degrees)
  End Sub

  Sub TurnLeft()
    NxtMotorA.Turn(Speed, TurnDegrees)
  End Sub

  Sub TurnRight()
    NxtMotorC.Turn(Speed, TurnDegrees)
  End Sub

  Sub StopMotor()
    NxtMotorA.Coast()
    NxtMotorC.Coast()
  End Sub
```

```
#End Region

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click
        StopMotor()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        NxtMotorA.Turn(20, 0)
        NxtMotorC.Turn(10, 0)
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click
        NxtMotorA.Turn(7, 0)
        NxtMotorC.Turn(-7, 0)
    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button7.Click
        NxtMotorA.Turn(-20, 0)
        NxtMotorC.Turn(-10, 0)
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        NxtMotorA.Turn(10, 0)
        NxtMotorC.Turn(10, 0)
    End Sub
```

```
    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click
        NxtMotorA.Turn(-10, 0)
        NxtMotorC.Turn(-10, 0)
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        NxtMotorA.Turn(10, 0)
        NxtMotorC.Turn(20, 0)
    End Sub

    Private Sub Button6_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button6.Click
        NxtMotorA.Turn(-7, 0)
        NxtMotorC.Turn(7, 0)
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button9.Click
        NxtMotorA.Turn(-10, 0)
        NxtMotorC.Turn(-20, 0)
    End Sub

    Private Sub ProgressBar1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ProgressBar1.Click

    End Sub
```

```
    Private Sub lblStatus_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
lblStatus.Click

    End SubEnd Class
```

5.4.1) Μια συζήτηση σχετικά με τη C sharp(C#)

Η C# είναι νία ολοκληρωμένη αντικειμενοστραφείς γλώσσα προγραμματισμού. Αναλυτικότερα, ο αντικειμενοστραφής προγραμματισμός (ΑΣΠ) – Object Oriented Programming (OOP) – είναι ένα στυλ προγραμματισμού (ή υπόδειγμα προγραμματισμού). Υπάρχουν και άλλα υποδείγματα ππρογραμματισμού, όπως είναιι ο συναρτησιακός (functional) και ο διαδυκτιακός (procedural) προγραμματισμός. Αυτά τα υποδείγματα όμως εστιάζονται περισσότερο στις ενέργειες, ενώ ο αντικειμενοστραφείς προγραμματισμός εστιάζεται στα ίδια τα δεδομένα.

Η C# μας επιτρέπει να αναπτύσσουμε διαδικασιακές εφαρμογές, καθαρά αντικειμενοστραφείς, ή ένα μείγμα και των δύο. Οι εφαρμογές που χρησιμοποιούν το υπόδειγμα ΑΣΠ κατασκευάζονται με τη χρήση γλωσσών ΑΣΠ. Η πρώτη γλώσσα αντικειμενοστραφούς προγραμματισμού παρουσιάστηκε τη δεκαετία του 1960, αλλά αυτές οι γλώσσες έγιναν πραγματικά δημοφιλείς στο τέλος της δεκαετίας του 70. Σήμερα χρησιμοποιούνται ευρέως, επειδή οι περισσότεροι προγραμματιστές συμφωνούν πως είναι εύκολες στην εκμάθηση, τη χρήση, της απασφαλμάτωση και τη συντήρηση. Για παράδειγμα, οι γλώσσες ΑΣΠ μπορούν να αναπαραστήσουν με ευκολία αντικείμενα του πραγματικού κόσμου. Η C# είναι γλώσσα ΑΣΠ. Όπως είναι και η VB.net, η Java, η C++, η SmallTalk και η Lisp.

Οι προγραμματιστές χρησιμοποιούν τον αντικειμενοστραφή προγραμματισμό για να για να γράψουν προγράμματα που αντιπροσωπεύουν την ανάλυση προβλημάτων του πραγματικού κόσμου σε υπομονάδες (modules). Αυτές οι υπομονάδες αντιπροσωπεύουν αντικείμενα του πραγματικού κόσμου και ονομάζονται κλάσεις (classes) ή τύποι (types). Μπορούμε να θεωρήσουμε ένα πρόγραμμα ΑΣΠ ως μία ομάδα αντικειμένων που αλληλεπιδρούν μεταξύ τους. Χρησιμοποιώντας ΑΣΠ, ένας προγραμματιστής ορίζει νέους

τύπους για να αναπαραστήσει αντικείμενα του πραγματικού κόσμου, όπως ένα αεροπλάνο, ένας άνθρωπος, ένας πελάτης, ένας σκύλος, ένα αυτοκίνητο ή ένα ρομπότ.

Αυτοί οι τύποι ή κλάσεις δημιουργούν αντικείμενα ή παρουσίες (instances). Αντικείμενο είναι μια μονάδα που αντιπροσωπεύει μια παρουσία του πραγματικού κόσμου. Πρόκειται για μια αυτάρκη μονάδα, επειδή περιλαμβάνει όλα τα δεδομένα και τις λειτουργίες που έχουν σχέση με το συγκεκριμένο αντικείμενο. Αυτό σημαίνει ότι κάθε αντικείμενο που δημιουργείται σε μια εφαρμογή περιέχει όλες τις πληροφορίες που το χαρακτηρίζουν (μέλη δεδομένων – data members) και όλες ρις ενέργειες (μεθόδους) που μπορούν να προσπελάζουν ή να τροποποιούν τις πληροφορίες αυτές.

Παρακάτω βλέπουμε ένα παράδειγμα για να κατανοήσουμε τον ορισμό της κλάσης ενός προσώπου:

```
1   using system
2
3   public class Person
4   {
5       // Μέλη δεδομένων
6       public string Name;
7       public string Address;
8       public string City;
9       public string State;
10      public string Zip;
11      public string Country;
12
13      // Μέθοδοι
14      public Virtual void Display()
15      {
16          Console.WriteLine (Name);
17          Console.WriteLine (Address);
18          Console.WriteLine (City);
```

```
19            Console.WriteLine (State);
20            Console.WriteLine (Zip);
21            Console.WriteLine (Country);
22        }
23  }
```

Αυτή η κλάση περιέχει δημόσια (public) μέλη δεδομένων και μία μέθοδο απεικόνισης για την εκτύπωση του περιεχομένου του αντικειμένου στην κονσόλα. Η λέξη-κλειδί virtual (εικονικός) σημαίνει ότι μία νέα κλάση που παράγεται από αυτή την κλάση θα έχει τη δυνατότητα να γράψει τη δική της υλοποίηση της μεθόδου απεικόνισης.

Παρακάτω θα χρησιμοποιήσουμε ένα διαφορετικό παράδειγμα για να εξετάσουμε όλες αυτές τις έννοιες. Το άλογό μου ο Μάιλο είναι μία παρουσία της κλάσης άλογο και η κλάση άλογο είναι μία δευτερεύουσα κλάση (subclass) της κλάσης Ζώο. Επειδή ο Μάιλο είναι άλογο, έχει ορισμένες συμπεριφορές και δεδομένα που ισχύουν για ένα άλογο. Αλλά επειδή το άλογο είναι επίσης και Ζώο, ο Μάιλο κληρονομεί επίσης ορισμένα χαρακτηριστικά και συμπεριφορές της κλάσης Ζώο.

Αυτό σημαίνει ότι η παρουσία της κλάσης Άλογο έχει μέλη δεδομένων που τη χαρακτηρίζουν και μεθόδους τις οποίες μπορώ να καλέσω γι’ αυτό το ψηλό και καμαρωτό πλάσμα (το Μάιλο). Για παράδειγμα παρακάτω μπορούμε να δούμε τις πληροφορίες της παρουσίας του αντικειμένου Μάιλο:

Δεδομένα

- Ράτσα: Είναι Αραβικό-ταχύτητας
- Φύλλο: Είναι αρσενικό
- Βάρος: Ζυγίζει 270 κιλά
- Χρώμα: Είναι καστανόξανθο
- Όνομα: Το όνομά του είναι Μάιλο
- Ηλικία: Είναι 4,5 χρονών

Ενέργειες

- Μιλάει: Χλιμιντρίζει
- Τρώει
- Κινείται
- Κοιμάται

Όλα αυτά τα στοιχεία δεδομένων (ράτσα, φύλο, βάρος, χρώμα, όνομα και ηλικία) και ενέργειες (μιλάει, τρώει, κινείται και κοιμάται) τον χαρακτηρίζουν, αλλά μπορούν επίσης να χαρακτηρίσουν οποιοδήποτε άλλο άλογο, όπως το άλογο του γείτονά μου, τον Ρέιντζερ. Και αν το σκεφτούμε, αυτά τα στοιχεία μπορούν να χαρακτηρίσουν οποιοδήποτε ζώο. Αυτό σημαίνει ότι η κλάση Άλογο κληρονομεί μέλη και μεθόδους δεδομένων από την κλάση Ζώο.

Ας πούμε ότι θέλουμε να δημιουργήσουμε μία εφαρμογή για μία κτηνιατρική κλινική. Για να καλύψουμε τις γάτες που επισκέπτονται την κλινική μας, αρκεί να δημιουργήσουμε και μία κλάση Γάτα η οποία θα κληρονομεί επίσης από την κλάση ζώο. Τότε, κάθε κλάση (γάτα ή άλογο) θα μπορεί να υποσκελίζει (override) τις λειτουργίες της κλάσης Ζώο όποτε είναι απαραίτητο. Για παράδειγμα, για την κλάση Γάτα η μέθοδος ομιλίας είναι ‘νιαουρίζει’ αντί να ‘χλιμιντρίζει’.

Ας πάμε λίγο πίσω να ξαναδούμε το παράδειγμα της κλάσης Person (άτομο). Αυτή τη φορά, προσθέτουμε την κλάση Employee (υπάλληλος) η οποία παράγεται από την κλάση Person. Η κλάση Employee παράγεται από την κλάση Person με την χρήση του τελεστή : (άνω και κάτω τελεία) που ακολουθείται από το στοιχείο Person. Η λέξη κλειδί override (υποσκελισμός) αλλάζει την υλοποίηση της μεθόδου απεικόνισης Display()

Συνέχεια από το προηγούμενο παράδειγμα κώδικα:

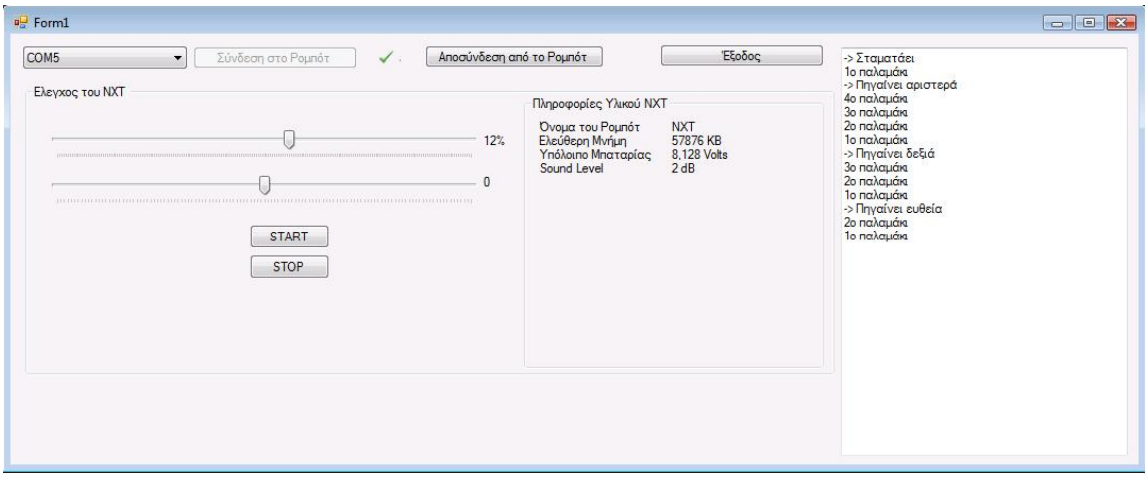
```
24  public class Employee : Person
25
26  {
27      public int Level;
28      public int Salary;
```

```
29
30      Public override void Display()
31      {
32          Console.WriteLine(Name + ¨έχει βαθμό¹ ¨ + Level.ToString() + ¨και παίρνει
        μισθό : ¨ + Salary.ToString() + ¨€¨)
33          Console.WriteLine(¨Η διεύθυνσή του είναι : );
34          Console.WriteLine(address);
35          Console.WriteLine(City + ¨,¨ + State + ¨ ¨ + Zip);
36          Console.WriteLine(Country);
37      }
38  }
```

Σε αυτή την περίπτωση, η κλάση Employee κληρονομεί από την κλάση Person και, επομένως, παίρνει όλα τα πεδία δεδομένων από αυτή τη βασική κλάση. Η κλάση Employee δε χρειάζεται να επανορίσει όλα τα πεδία στον ορισμό της επειδή τα παίρνει αυτόματα από την κλάση Person. Επομένως, για την κλάση Employee πρέπει να καθορίσουμε μόνο σε τι διαφέρει από μια παρουσία της κλάσης Person. Για παράδειγμα, μία παρουσία της κλάσης Employee θα μπορούσε να έχει θέση και μισθό, ενώ δεν ισχύει το ίδιο για όλες τις παρουσίες της κλάσης Person. Επιπλέον, όταν καλείται η μέθοδος Display για την κλάση Employee, θα μπορούσε να συμπεριλαμβάνει πληροφορίες θέσης και μισθού στο μήνυμα που εμφανίζει.

5.4.2) Η βιβλιοθήκη Mindsqualls και το πρόγραμμά στη C#

Σε αυτό το πρόγραμμα, θα μπορούσαμε να χρησιμοποιήσουμε επίσης τη βιβλιοθήκη του Fokke αλλά θεωρήσαμε ότι ήταν καλύτερο να δούμε μία ακόμη βιβλιοθήκη με εξίσου καλές δυνατότητες. Η βιβλιοθήκη που χρησιμοποιούμε στο συγκεκριμένο πρόγραμμα είναι η mindsqualls με την οποία μπορούμε να ελέγξουμε το NxtBrick1 με τις γλωσσες (C#, VB.net kai J#). Ο τρόπος λειτουργίας της βιβλιοθήκης είναι παρόμοιος με την βιβλιοθήκη του Fokke απλά αλλάζει λίγο ο τρόπος σύνταξης των εντολών.



Αρχικά διαλέγουμε την θύρα που συνδέεται το ρομπότ και πατάμε σύνδεση.

Με το πρώτο δρομέα ελέγχουμε την ταχύτητα των σερβοκινητήρων (με θετικές τιμές, προχωράει μπροστά και με αρνητικές, πίσω) και με το δεύτερο δρομέα, ελέγχουμε την κατεύθυνση πού θα στρίψει(εάν θέλουμε να το δουλέψουμε με τα κουμπιά).

Στη μέση βλέπουμε ένα πάνελ με διάφορες πληροφορίες του (το όνομά του, την ελεύθερη μνήμη, την τάση λειτουργίας της μπαταρίας του και την στάθμη του ήχου από τον Sound Sensor).

Στα δεξιά βλέπουμε τα παλαμάκια που δέχεται και τις κινήσεις που εκτελεί.

Με το παραπάνω πρόγραμμα θέλουμε να ελέγχουμε την κίνηση του ρομπότ με χτύπους των χεριών μας (παλαμάκια). Παλαμάκι καταλαβαίνει κάθε φορά που η στάθμη του ήχου ξεπερνάει τα 70 dB. Σε κάθε παλαμάκι, περιμένει 800 ms για να δεχθεί και το επόμενο παλαμάκι, αλλιώς εκτελεί τη λειτουργία για το εκάστοτε παλαμάκι. Με ένα παλαμάκι σταματάει να κινείται, με δύο προχωράει ευθεία, με τρία στρίβει δεξιά και με τέσσερα αριστερά. Εάν έχουμε τον δρομέα ταχύτητας στα αρνητικά, το ρομπότ εκτελεί τις παραπάνω εντολές αντίστροφα(αντί για μπροστά πηγαίνει πίσω, αντί για δεξιά στρίβει αριστερά και αντί για αριστερά στρίβει δεξιά).

Αρχικά ξεκινούμε με την αρχικοποίηση κάποιων μεταβλητών ελέγχου οι οποίες είναι:

```
bool oldstatus = false;
            NxtBrick brick=null;
//το ρομποτ
            NxtMotorSync motorPair = null;
//το ζευγος των κινητηρων
            NxtSoundSensor soundSensor = null;
//ο αισθητήρας ήχου
            DateTime lastClap;
//Χρόνος που δηλώνει πότε έγινε το τελευταίο παλαμάκι
        long timeInterval = 8000000;
//χρόνος αναμονής μεταξύ δύο παλαμακίων (σε nsec)
            int timesClapped = 0;
//ποσα παλαμάκια έχουν καταμετρηθεί
        byte oldsoundlevel = 0;
//παλιό επίπεδο ήχου
        bool timerSoundStarted = false, motorStartedFromButton = false;

//μεταβλητές ελέγχου του χρονόμετρου(για τα παλαμάκια) και του κινητήρα
//(απο το κουμπί Start)
```

Και συνεχίζουμε ως εξής:

```
//Συνάρτηση - δημιουργός του παραθύρου
            public Form1()
            {

            InitializeComponent();
//αρχικοποίηση των αντικειμένων (C#, αυτόματη συνάρτηση)
            timerCheckConnection_Tick(null, null);

            timerCheckConnection.Interval = 1500;
//θέτουμε κάθε πότε το πρόγραμμα θα ελέγχει την ύπαρξη σύνδεσης
```

```
//και θα ανανεώνει τις πληροφορίες του υλικού
            timerCheckConnection.Enabled = true;

//Αρχικά, δεν υπάρχει σύνδεση με το ρομπότ. Συνεπώς το πάνελ με τις
//πληροφορίες του ρομπότ είναι ανενεργό

                  interactionWithRobot.Enabled = false;
//Αρχικοποιούμε την θύρα σύμφωνα με μια επιλεγόμενη τιμή (αυτόματη
//επιλογή COM5 - για διευκόλυνση)
            cmbCOMport.SelectedIndex = 4;
//Αρχικοποιούμε τις μπάρες κύλισης (ισχύς κινητήρα και στροφές)
            motorPower_Scroll(null, null);
            trackBar1_Scroll(null, null);
        }
            private    void    timerCheckConnection_Tick(object    sender,
EventArgs e)
            {
//Η συνάρτηση αυτή καλείτε κάθε φορά που γίνεται ελεγχος ύπαρξη
//σύνδεσης (κάθε 1.5 sec)

            if (brick == null || brick.IsConnected == false)
//Αν δεν υπάρχει σύνδεση
                  {
//Θέτουμε το εικονίδιο της σύνδεσης να είναι κόκκινο X
                        picStatus.Image                                    =
nxtsensor.Properties.Resources.delete_16x;
                        if(oldstatus == true)
//Αν στον προηγούμενο έλεγχο υπήρχε σύνδεση
                        {
//Απενεργοποιούμε το πάνελ του ρομπότ
                              interactionWithRobot.Enabled = false;
                              oldstatus = false;
                              picStatus.Refresh();
                        }
                  }
                  else
                  {
//Αν υπάρχει σύνδεση με το ρομπότ
```

```
//θέτουμε το εικόνίδιο της σύνδεσης πράσινο √
                        picStatus.Image                                 =
nxtsensor.Properties.Resources.Verify;
                picStatus.Refresh();
                interactionWithRobot.Enabled = true;
//Ενεργοποιούμε το πάνελ του ρομπότ
                        oldstatus = true;
//Βαζουμε στο πάνελ τις πληροφορίες του ρομπότ

                        NxtGetDeviceInfoReply?          devinfo         =
brick.CommLink.GetDeviceInfo();
//Ζητούμε το ρομπότ να μας δώσει πληροφορίες για τον εαυτό του
                        if(devinfo != null)
                        {
                              lblRobotName.Text = devinfo.Value.nxtName;
//όνομα
                              lblFreeFlash.Text                         =
""+devinfo.Value.freeUserFlash +" KB";
//ελεύθερη μνήμη
                        }
                        lblBatteryLevel.Text                            =
""+brick.CommLink.GetBatteryLevel().Value/1000.0 + " Volts";
//Στάθμη μπαταρίας
                        grpHardwareInfo.Refresh();
//Ανανέώνουμε το πάνελ υλικού
                 }


            }
        private void soundSensorPoll(NxtPollable polledItem)
         {
//Η συνάρτηση αυτή καλείται κάθε φορά που γίνεται καταμέτρηση του
//επιπέδου του ήχου
//Εμφανίζει το επίπεδο ήχου στο παράθυρο
             label8.Text = "" +soundSensor.SoundLevel.Value+ " dB";
        }
           private void soundSensorOnAboveSoundLevel(NxtSensor sensor)
           {
```

```
//Η συνάρτηση αυτή καλείτε κάθε φορά που η ένταση του ήχου ξεπεράσει
καποια προκαθορισμένη τιμή.

                DateTime rightnow = DateTime.Now;        //Καταγράφουμε
την τρέχουσα ώρα με ακρίβεια nanosec
                if(rightnow.Ticks-lastClap.Ticks     >     timeInterval)
//Αν ο χρόνος μεταξύ των δύο τελευταίων χτυπημάτων
//ξεπερνάει την προκαθορισμένη τιμή, ξεκινάμε απο την άρχή
                {
               motorStartedFromButton = false;
               timesClapped = 1;
               textBox1.Text   =   timesClapped+   "o   παλαμάκι\r\n"   +
textBox1.Text;

                }
                else
                {
//Διαφορετικά ανεβάζουμε τον μετρητή για τα παλαμάκια
               timesClapped++;
               textBox1.Text   =   timesClapped   +   "o   παλαμάκι\r\n"   +
textBox1.Text;
                }
//Ορίζουμε τον χρόνο που έγινε το τελευταίου παλαμάκι --τώρα
                lastClap = rightnow;
//Ενεργοποιούμε τον αισθητήρα χρόνου παλαμακίων
            timerSoundStarted = true;
            }

            private void btnConnect_Click(object sender, EventArgs e)
            {
//Κουμπί Σύνδεσης
                if(cmbCOMport.SelectedIndex >= 0)
                {
//Δημιουργία σύνδεση με το ρομπότ
                        brick                    =                    new
NxtBrick((byte)(cmbCOMport.SelectedIndex+1));
                        try
                        {
```

```
//Πραγματοποίηση σύνδεσης
                                brick.Connect();
//Αντιστοίχιση κινητήρων
                                brick.MotorA = new NxtMotor();
                                brick.MotorB = new NxtMotor();
                                brick.MotorC = new NxtMotor();

                                motorPair = new NxtMotorSync(brick.MotorB,
brick.MotorC);
//Αντιστοίχιση Αισθητήρων
                                soundSensor = new NxtSoundSensor();

                                brick.Sensor2 = soundSensor;
                        soundSensor.dBA = true;

//Ορίζουμε κάθε πόσο χρόνο θα μετράται το επίπεδο του ήχου και τη
//συνάρτηση που θα καλείται
                                soundSensor.PollInterval = 25;
                        soundSensor.OnPolled                    +=                  new
NKH.MindSqualls.Polled(soundSensorPoll);
//Ορίζουμε την τιμή σύγκρισης του επιπέδου του ήχου
                        soundSensor.CompareSoundLevel=70;
//Κάθε φορά που το επίπεδο του ήχου ξεπεράσει την παραπάνω τιμή,
//καλείτε η συνάρτηση soundSensorOnAboveSoundLevel
                                soundSensor.OnAboveSoundLevel     +=     new
NKH.MindSqualls.NxtSensorEvent(soundSensorOnAboveSoundLevel);

//Ενεργοποιούμε το κουμπι αποσύνδεσης, και το πάνελ του ρομπότ, και
//απενεργοποιούμε το κουμπί της σύνδεσης
                        interactionWithRobot.Enabled = true;
                        btnDisconnect.Enabled = true;
                        btnConnect.Enabled = false;

                            }
                            catch (System.Exception ex)
                            {
//Σε περίπτωση αποτυχίας, εμφανίζουμε το απαραίτητο μήνυμα
```

```
                        MessageBox.Show(this, "The connection could
not be made. Check that the bluetooth is available and its port is " +
cmbCOMport.SelectedText+  ".",  "Not  Connected",  MessageBoxButtons.OK,
MessageBoxIcon.Error);

                    }
                }
                else
                {
                    MessageBox.Show(this,   "No   port   have   been
specified",     "Insufficient     parameters",     MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }
            }
            private void goStraight()
            {
//Συνάρτηση που προστάζει το ρομπότ να κινηθεί εμπρός
            sbyte v = (sbyte)motorPower.Value;
                Run(0, v);
        }

        private void turnLeft()
//Συνάρτηση που προστάζει το ρομπότ να κινηθεί αριστερά
        {
            sbyte v = (sbyte)motorPower.Value;
            Run(-70,v);
            }
            private void turnRight()
            {
//Συνάρτηση που προστάζει το ρομπότ να κινηθεί δεξιά

            sbyte v = (sbyte)motorPower.Value;
            Run(70, v);
            }
            private void Stop()
            {
//Συνάρτηση που προστάζει το ρομπότ να σταματήσει
```

```
            motorPair.ResetMotorPosition(true);
                  motorPair.Brake();
            }

            private void soundTimer_Tick(object sender, EventArgs e)
            {
//Η συνάρτηση αν δεν έχει χτυπηθεί παλαμάκι δεν κάνει τίποτε
            if (timerSoundStarted == true)
            {
//Αν έχουν χτυπηθεί τουλάχιστον 1 παλαμάκι και έχει περάσει το
//προκαθορισμένο χρονικό όριο, ή έχουν χτυπηθεί 4 παλαμάκια
//εκτελείτε η αντίστοιχη εντολή
                DateTime rightnow = DateTime.Now;
                if (rightnow.Ticks - lastClap.Ticks > timeInterval ||
timesClapped == 4)
                {
                    String msg = "";
                    switch (timesClapped)
                    {
                        case 1:
                            msg += "-> Σταματάει";
                            Stop();
                            break;
                        case 2:
                            msg += "-> Πηγαίνει ευθεία";
                            goStraight();
                            break;

                        case 3:
                            msg += "-> Πηγαίνει δεξιά";
                            turnRight();
                            break;
                        case 4:
                            msg += "-> Πηγαίνει αριστερά";
                            turnLeft();
                            break;
                    }
                    textBox1.Text = msg + "\r\n" + textBox1.Text;
```

```
                    timerSoundStarted = false;
                    timesClapped = 0;
                }
            }
            }

            private void motorPower_Scroll(object sender, EventArgs e)
            {
//Η συνάρτηση αυτή καλείτε κάθε φορά που αλλάζει η τιμή της ταχύτητας
//του ρομπότ
                  label7.Text = motorPower.Value + "%";
            if (motorStartedFromButton==true)
            {
//Αν έχει πατηθεί το κουμπί Start, αλλάζουμε την ταχύτητα του ρομπότ
                Run(trackBar1.Value, motorPower.Value);


            }
            }

        private void label7_Click(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
//Κουμπί Start. Προστάζουμε το ρομπότ να κουνηθεί με βάση ταχύτητα και
//γώνία που έχουν εκλεγεί απο το χρήστη
            Run(trackBar1.Value, motorPower.Value);
            motorStartedFromButton = true;
        }

        private void button2_Click(object sender, EventArgs e)
        {
//κουμπί Stop. Σταματάμε το ρομπότ
            motorStartedFromButton = false;
            Stop();
```

```
//Αρχικοποιούμε τις τιμές της γωνίας και της ταχύτητας
            trackBar1.Value = 0;
            motorPower.Value = 30;
            trackBar1_Scroll(sender, e);
            motorPower_Scroll(sender, e);
        }
        private void Form1_Load(object sender, EventArgs e)
        {
//Απενεργοποιούμε έναν έλεγχο για δια-νηματικές κλήσεις
            CheckForIllegalCrossThreadCalls = false;

        }

        private void Form1_FormClosed(object sender, FormClosedEventArgs
e)
        {
//Στο τέλος, απελευθερώνουμε το ρομπότ
            if(brick !=null && brick.IsConnected)
                brick.Disconnect();
            this.Dispose();

        }

        private void btnDisconnect_Click(object sender, EventArgs e)
        {
//κουμπί αποσύνδεσης
//απενεργοποιούμε τους αισθητήρες
            if(soundSensor != null)
            {
                soundSensor.PollInterval = 0;
                soundTimer.Enabled = false;
            }
//Αποσυνδεόμαστε απο το ρομπότ
            if (brick != null && brick.IsConnected == true)
                brick.Disconnect();
//απενεργοποιούμε το κουμπί αποσύνδεσης και ενεργοποιούμε το κουμπί
//σύνδεσης
            btnConnect.Enabled = true;
```

```
            btnDisconnect.Enabled = false;
        }

        private void label8_Click(object sender, EventArgs e)
        {

        }

        private void btnExit_Click(object sender, EventArgs e)
        {
//Κουμπί εξόδου. Πρώτα αποσυνδεόμαστε και μετα κλείνουμε την εφαρμογή
            btnDisconnect_Click(sender, e);
            Dispose();
        }

        private void trackBar1_Scroll(object sender, EventArgs e)
        {
//Η συνάρτηση αυτή καλείται κάθε φορά που αλλάζει η γωνία κίνησης
            label9.Text = ""+trackBar1.Value;
            if (motorStartedFromButton == true)
            {
//Αν το ρομπότ κουνιέται, αλλάζουμε την γωνία κίνησης με την νέα
                Run(trackBar1.Value, motorPower.Value);

            }

        }

        private void interactionWithRobot_Enter(object sender, EventArgs
e)
        {

        }

        private void Run(int degrees1, int power)
        {
//Μια δική μας συνάρτηση κίνησης του ρομπότ.
```

```
//Οι δύο μεταβλητές ορίζουν τον συντελεστή ταχύτητας του κάθε κινητήρα
//(σε σχέση με την ταχύτητα power)
            double bcoef, ccoef;
//Αλλάζουμε την γώνία σε ακτίνια
            double degrees = 3.141592653 * degrees1 / 180;
//degrees-> rad, degrees1-> μοίρες
            if(degrees < 0 )
            {
//Για αρνητικές γωνίες, κινούμαστε αριστερά.
//Αφήνουμε το δεξί κινητήρα με συντελεστή 1
//και ελαττώνουμε τον αριστερό σε συντελεστή cos(degrees)
                bcoef = Math.Cos(degrees);
                ccoef = 1;
            }
            else
            {
//Ενεργούμε αντίθετα στην άλλη περίπτωση
                ccoef = Math.Cos(degrees);
                bcoef = 1;
            }
//Ουσιαστικά μειώνουμε την ισχύ του ενός κινητήρα ανάλογα με τη γωνία
//στροφής
//Θέτουμε τις ταχύτητες των κινητήρων με βάση τους συντελεστές κίνησης
//και την επιλεγμένη, απο το χρήστη ταχύτητα
            sbyte powerC = (sbyte)(power * bcoef);
            sbyte powerB = (sbyte)(power * ccoef);
            brick.MotorB.Run(powerB, 0);
            brick.MotorC.Run(powerC, 0);
        }
        private void label3_Click(object sender, EventArgs e)
        {

        }
        private void grpHardwareInfo_Enter(object sender, EventArgs e)
      }
}
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλία

- Building Robots with Lego Mindstorms NXT (Dave Astolfo, Mario Ferrari, Giulio Ferrari)
- Extreme NXT Extending the LEGO MINDSTORMS NXT to the Next Level (Michael Gasperi and Philippe Hurbain with Isabelle Hurbain)
- LEGO Mindstorms NXT The Mayan Adventure (James Floyd Kelly)
- Creating Cool Mindstorms NXT Robots (Daniele Benedettelli)

Διαδίκτυο

- Lego Mindstorms (http://mindstorms.lego.com)

  Είναι η κεντρική σελίδα του Lego Mindstorms robot. Περιέχει τεχνικές πληροφορίες, διοργάνωνε διαγωνισμοί και έχει ένα forum που μπορούμε να λύσουμε τις διαφορές απορίες.

- http://www.nxtasy.org/

  Είναι μια πηγή πληροφοριών για γενικής φύσεως θέματα που αφορούν το Nxt Lego robot παραδείγματος χάριν τον προγραμματισμό του και την ανταπόκριση του μέσα από διαφορά projects.

- http://thenxtstep.blogspot.com/

  Ένα επίσημο Blog που αναφέρεται στην ρομποτική, δίνει όμως έμφαση στην λειτουργιά του Nxt Lego robot.

- http://www.philohome.com/

  Μια ιστοσελίδα του Philippe Hurbain μέσα από την όποια κάνει μια νύξη για τον τρόπο λειτουργιάς των μοτέρ και των αισθητήρων μέσα από τις εικόνες που παρουσιάζει.

- http://web.mac.com/ryo_watanabe/iWeb/Ryo%27s%20Holiday/NXT%20Motor.html

  Μας παρουσιάζει ο Ryo Watanable την μαθηματική ανάλυση του αισθητήρα κίνησης (moter).

- http://forum.mindstormsnxt.gr/jforum/forums/show/1.page

  Το ελληνικό Forum που ασχολείται αποκλειστικά με θέματα ρομποτικής σχετικά με το Lego robot.

- http://www.teamhassenplug.org/NXT/NXTSoftware.html

  Η ιστοσελίδα, μας παρουσιάζει τα χαρακτηριστικά κάθε γλώσσας προγραμματισμού που ανταποκρίνονται στο Lego robot και με ποιο τρόπο.

- http://www.extremenxt.com/vbpart1.htm

  Παίρνουμε πληροφορίες συμβατότητα της VB.net με το Nxt μέσω Bluetooth.

- http://www.jstuber.net/lego/nxt-programming/nxt-hardware.html

  Παρουσίαση υψηλής ευκρίνειας εικόνων του Hardware Nxt.

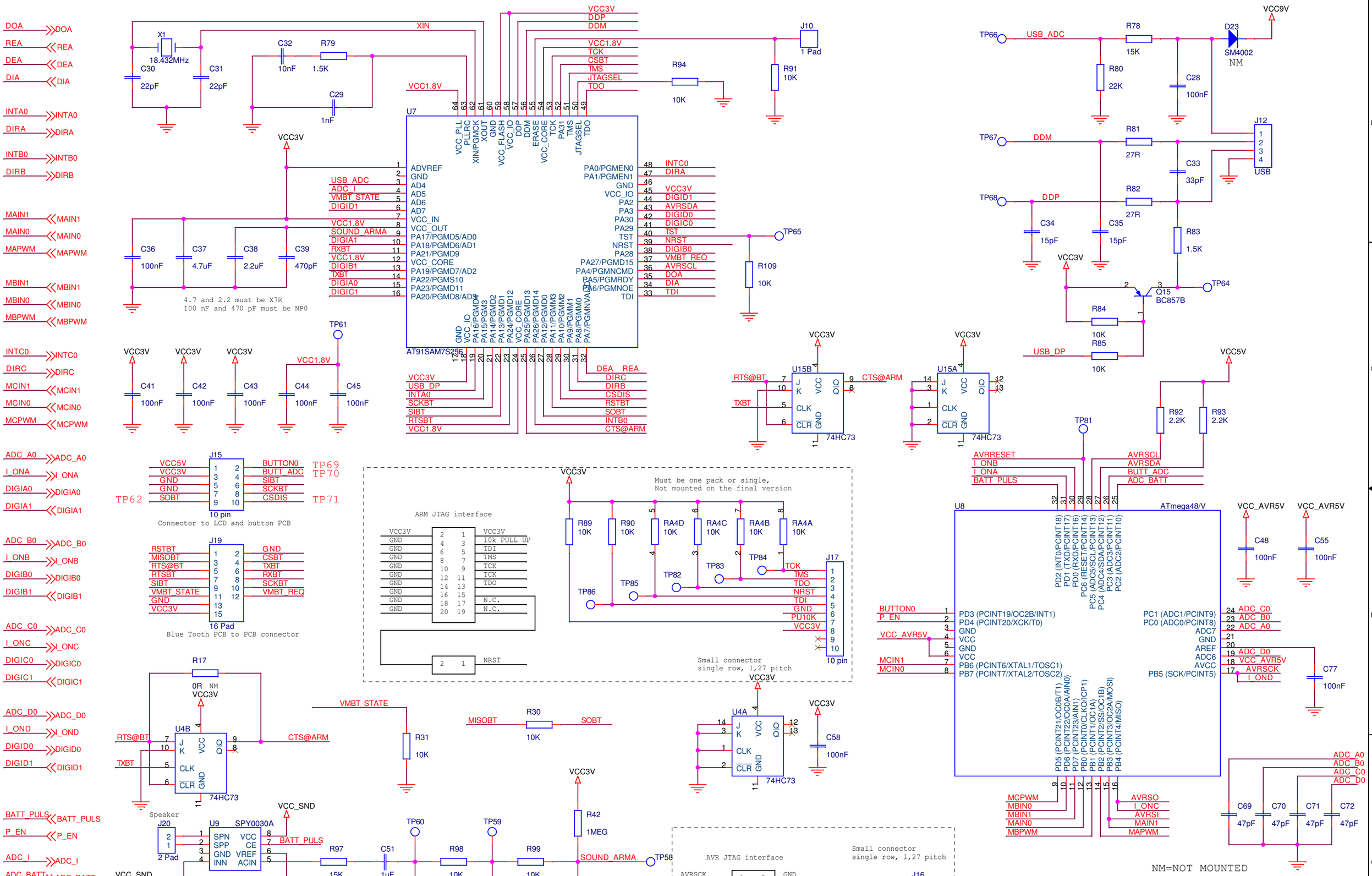
- http://www.mindsqualls.net/

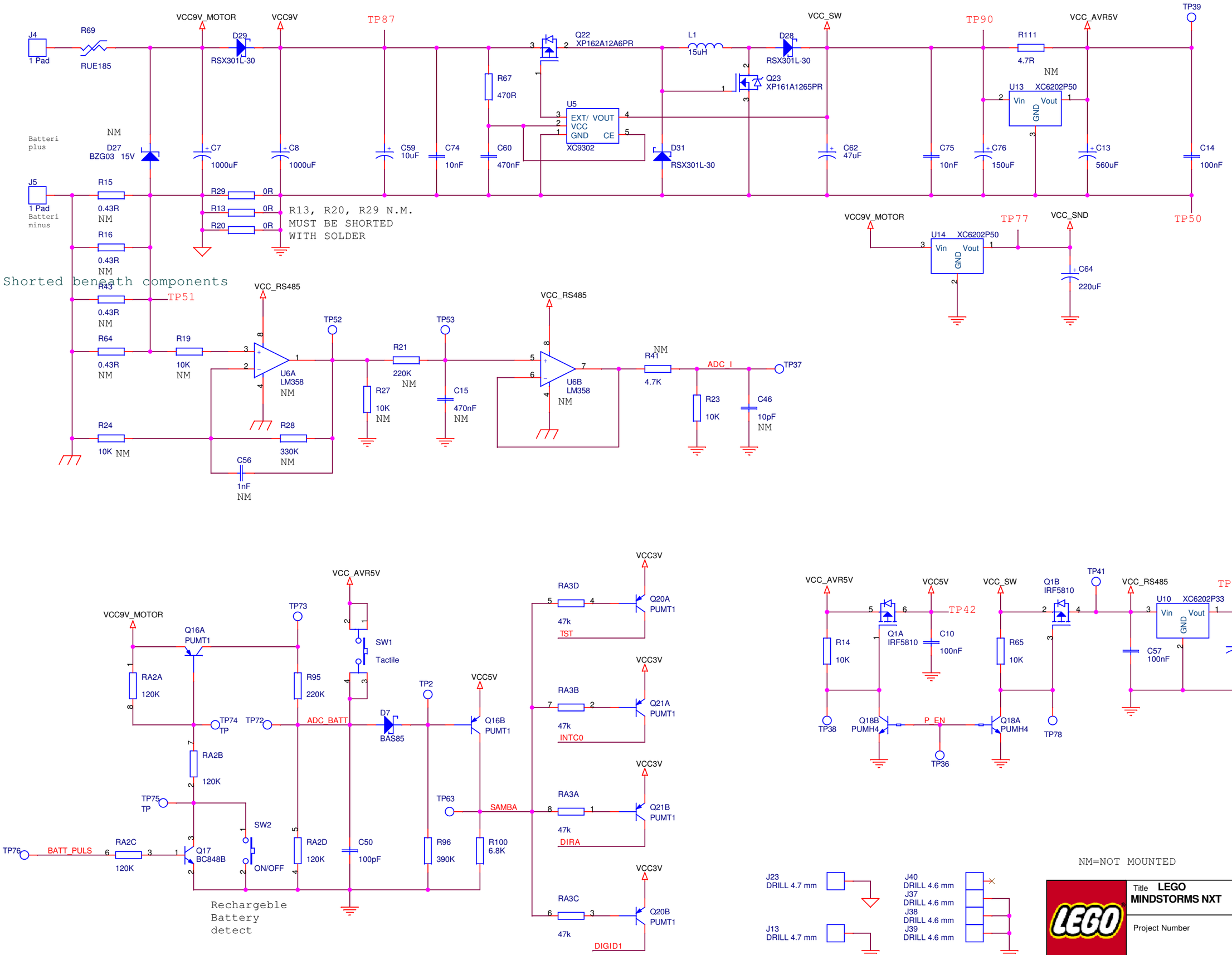
  Η βιβλιοθήκη της Mindsqualls

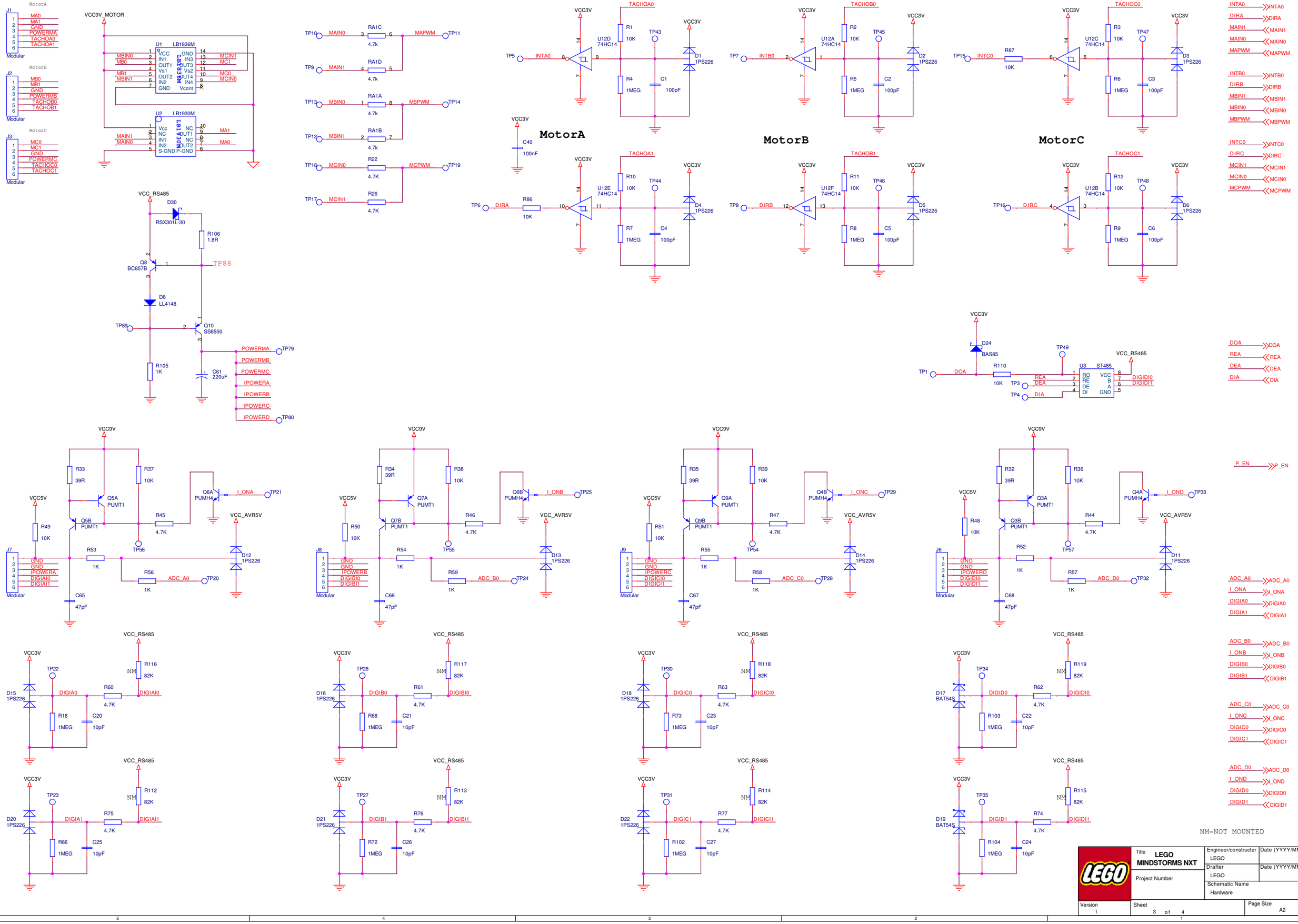
- http://lejos.sourceforge.net/p_technologies/nxt/icommand/api/index.html

  Η βιβλιοθήκη icommand

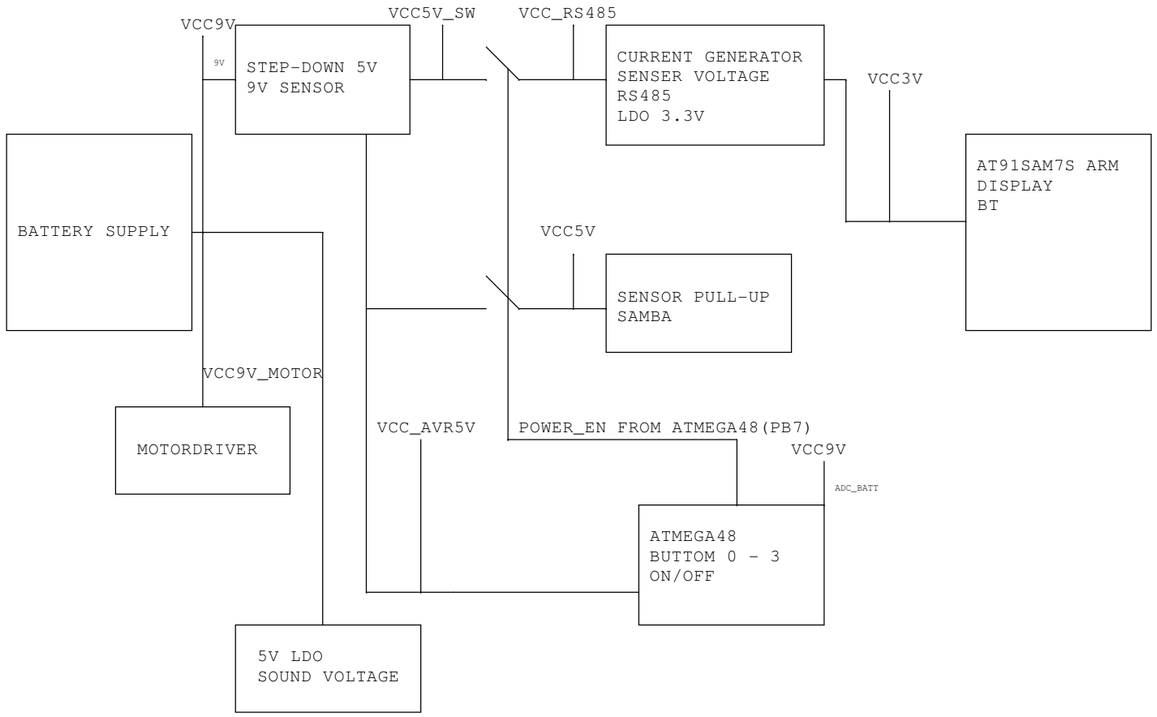
ΠΑΡΑΡΤΗΜΑ


DOA
REA
DEA
DIA
INTA0
DIRA
INTB0
DIRB
MAIN1
MAIN0
MAPWM
MBIN1
MBIN0
MBPWM
INTC0
DIRC
MCIN1
MCIN0
MCPWM
ADC_A0
I_ONA
DIGIA0
DIGIA1
ADC_B0
I_ONB
DIGIB0
DIGIB1
ADC_C0
I_ONC
DIGIC0
DIGIC1
ADC_D0
I_OND
DIGID0
DIGID1
BATT_PULS
P_EN
ADC_I
ADC_BATT
TST
X1
18.432MHz
C30
22pF
C31
22pF
C32
10nF
R79
1.5K
C29
1nF
XIN
VCC3V
DDP
DDM
VCC1.8V
TCK
CSBT
TMS
JTAGSEL
TDO
R94
10K
J10
1 Pad
R91
10K
U7
AT91SAM7S256
ADVREF
GND
AD4
AD5
AD6
AD7
VCC_IN
VCC_OUT
PA17/PGMD5/AD0
PA18/PGMD6/AD1
PA21/PGMD9
VCC_CORE
PA19/PGMD7/AD2
PA22/PGMS10
PA23/PGMD11
PA20/PGMD8/AD3
PA0/PGMEN0
PA1/PGMEN1
VCC_IO
PA2
PA3
PA30
PA29
TST
NRST
PA28
PA27/PGMD15
PA4/PGMNCMD
PA5/PGMRDY
PA6/PGMNOE
PA7/PGMNVALID
TDI
USB_ADC
ADC_1
VMBT_STATE
DIGID1
VCC1.8V
SOUND_ARMA
DIGIA1
RXBT
VCC1.8V
DIGIB1
TXBT
DIGIA0
DIGIC1
INTC0
DIRA
VCC3V
DIGID1
AVRSDA
DIGID0
DIGIC0
TST
NRST
DIGIB0
VMBT_REQ
AVRSCL
DOA
DIA
TDI
TP65
R109
10K
C36
100nF
C37
4.7uF
C38
2.2uF
C39
470pF
4.7 and 2.2 must be X7R
100 nF and 470 pF must be NP0
TP61
C41
100nF
C42
100nF
C43
100nF
C44
100nF
C45
100nF
VCC3V
USB_DP
INTA0
SCKBT
SIBT
RTSBT
VCC1.8V
DEA
REA
DIRC
DIRB
CSDIS
RSTBT
SOBT
INTB0
CTS@ARM
U15B
74HC73
RTS@BT
TXBT
CTS@ARM
U15A
74HC73
TP66
USB_ADC
R78
15K
D23
SM4002
NM
VCC9V
R80
22K
C28
100nF
TP67
DDM
R81
27R
C33
33pF
J12
USB
TP68
DDP
R82
27R
C34
15pF
C35
15pF
R83
1.5K
TP64
Q15
BC857B
R84
10K
R85
10K
USB_DP
VCC5V
R92
2.2K
R93
2.2K
TP81
AVRRESET
I_ONB
I_ONA
BATT_PULS
AVRSCL
AVRSDA
BUTT_ADC
ADC_BATT
U8
ATmega48/V
PD2 (INT0/PCINT18)
PD1 (TXD/PCINT17)
PD0 (RXD/PCINT16)
PC6 (RESET/PCINT14)
PC5 (ADC5/SCL/PCINT13)
PC4 (ADC4/SDA/PCINT12)
PC3 (ADC3/PCINT11)
PC2 (ADC2/PCINT10)
BUTTON0
P_EN
VCC_AVR5V
MCIN1
MCIN0
PD3 (PCINT19/OC2B/INT1)
PD4 (PCINT20/XCK/T0)
GND
VCC
GND
VCC
PB6 (PCINT6/XTAL1/TOSC1)
PB7 (PCINT7/XTAL2/TOSC2)
PC1 (ADC1/PCINT9)
PC0 (ADC0/PCINT8)
ADC7
GND
AREF
ADC6
AVCC
PB5 (SCK/PCINT5)
ADC_C0
ADC_B0
ADC_A0
ADC_D0
VCC_AVR5V
AVRSCK
I_OND
C48
100nF
C55
100nF
C77
100nF
PD5 (PCINT21/OC0B/T1)
PD6 (PCINT22/OC0A/AIN0)
PD7 (PCINT23/AIN1)
PB0 (PCINT0/CLKO/ICP1)
PB1 (PCINT1/OC1A)
PB2 (PCINT2/SS/OC1B)
PB3 (PCINT3/OC2A/MOSI)
PB4 (PCINT4/MISO)
MCPWM
MBIN0
MBIN1
MAIN0
MBPWM
AVRSO
I_ONC
AVRSI
MAIN1
MAPWM
C69
47pF
C70
47pF
C71
47pF
C72
47pF
ADC_A0
ADC_B0
ADC_C0
ADC_D0
NM=NOT MOUNTED
J15
10 pin
Connector to LCD and button PCB
VCC5V
VCC3V
GND
GND
SOBT
BUTTON0
BUTT_ADC
SIBT
SCKBT
CSDIS
TP62
TP69
TP70
TP71
J19
16 Pad
Blue Tooth PCB to PCB connector
RSTBT
MISOBT
RTS@BT
RTSBT
SIBT
VMBT_STATE
GND
VCC3V
GND
CSBT
TXBT
RXBT
SCKBT
VMBT_REQ
ARM JTAG interface
Must be one pack or single,
Not mounted on the final version
R89
10K
R90
10K
RA4D
10K
RA4C
10K
RA4B
10K
RA4A
10K
TP84
TP83
TP82
TP85
TP86
J17
10 pin
TCK
TMS
TDO
NRST
TDI
GND
PU10K
VCC3V
Small connector
single row, 1,27 pitch
R17
0R NM
VCC3V
U4B
74HC73
RTS@BT
TXBT
CTS@ARM
VMBT_STATE
R31
10K
MISOBT
R30
10K
SOBT
U4A
74HC73
C58
100nF
Speaker
J20
2 Pad
U9
SPY0030A
SPN
SPP
GND
INN
VCC
CE
VREF
ACIN
VCC_SND
BATT_PULS
R97
15K
C51
1uF
R98
10K
R99
10K
SOUND_ARMA
TP58
TP60
TP59
R42
1MEG
R25
1MEG
C52
100nF
C53
3.3nF
C54
2.2nF
C49
100nF
AVR JTAG interface
AVRSCK
AVRSO
N.C.
N.C.
AVRSI
GND
VCC_AVR5V
AVRRESET
N.C.
GND
Small connector
single row, 1,27 pitch
J16
8 pin
GND
VCC_AVR5V
AVRSCK
AVRSO
AVRSI
AVRRESET
LEGO
MINDSTORMS NXT
Project Number
Engineer/constructer
LEGO
Drafter
LEGO
Schematic Name
Hardware
Date (YYYY/MM/DD)
Date (YYYY/MM/DD)
Version
I
Sheet
1 of 4
Page Size
A3

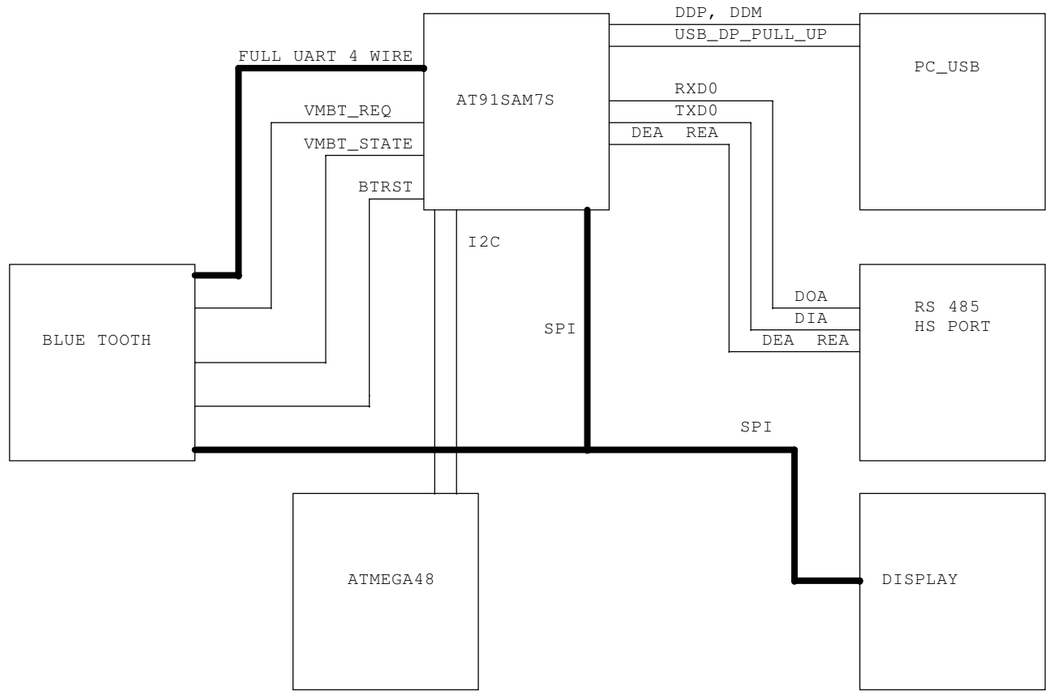

J4
1 Pad
R69
RUE185
VCC9V_MOTOR
D29
RSX301L-30
VCC9V
TP87
Q22
XP162A12A6PR
R67
470R
U5
EXT/ VCC
GND
VOUT
CE
XC9302
L1
15uH
D28
RSX301L-30
VCC_SW
Q23
XP161A1265PR
TP90
R111
4.7R
NM
U13 XC6202P50
Vin
Vout
GND
VCC_AVR5V
TP39
Batteri plus
NM
D27
BZG03 15V
C7
1000uF
C8
1000uF
C59
10uF
C74
10nF
C60
470nF
D31
RSX301L-30
C62
47uF
C75
10nF
C76
150uF
C13
560uF
C14
100nF
J5
1 Pad
Batteri minus
R15
0.43R
NM
R16
0.43R
NM
R29
0R
R13
0R
R20
0R
R13, R20, R29 N.M.
MUST BE SHORTED
WITH SOLDER
TP50
VCC9V_MOTOR
TP77
VCC_SND
U14 XC6202P50
Vin
Vout
GND
C64
220uF
Shorted beneath components
R43
0.43R
NM
TP51
R64
0.43R
NM
R19
10K
NM
VCC_RS485
U6A
LM358
NM
TP52
R21
220K
NM
TP53
R27
10K
NM
C15
470nF
NM
VCC_RS485
U6B
LM358
NM
R41
NM
4.7K
ADC_I
TP37
R23
10K
C46
10pF
NM
R24
10K NM
R28
330K
NM
C56
1nF
NM
DIGID1
DIRA
INTC0
TST
P_EN
BATT_PULS
ADC_BATT
ADC_I
VCC_AVR5V
VCC9V_MOTOR
TP73
Q16A
PUMT1
SW1
Tactile
RA2A
120K
R95
220K
TP74
TP
TP72
ADC_BATT
D7
BAS85
TP2
VCC5V
Q16B
PUMT1
RA2B
120K
TP75
TP
TP63
SAMBA
SW2
ON/OFF
TP76
BATT_PULS
RA2C
120K
Q17
BC848B
RA2D
120K
C50
100pF
R96
390K
R100
6.8K
Rechargeble
Battery
detect
VCC3V
RA3D
47k
TST
Q20A
PUMT1
RA3B
47k
INTC0
Q21A
PUMT1
RA3A
47k
DIRA
Q21B
PUMT1
RA3C
47k
DIGID1
Q20B
PUMT1
VCC_AVR5V
VCC5V
VCC_SW
Q1B
IRF5810
TP41
VCC_RS485
TP40
VCC3V
U10 XC6202P33
Vin
Vout
GND
TP42
Q1A
IRF5810
C10
100nF
R14
10K
R65
10K
C57
100nF
C11
220uF
C12
100nF
R40
1K
Bleeder
resistor
TP38
Q18B
PUMH4
P_EN
Q18A
PUMH4
TP36
TP78
NM=NOT MOUNTED
J23
DRILL 4.7 mm
J13
DRILL 4.7 mm
J40
DRILL 4.6 mm
J37
DRILL 4.6 mm
J38
DRILL 4.6 mm
J39
DRILL 4.6 mm
Steering for display PCB
Screwholes
Title
LEGO
MINDSTORMS NXT
Engineer/constructer
LEGO
Date (YYYY/MM/DD)
Drafter
LEGO
Date (YYYY/MM/DD)
Project Number
Schematic Name
Hardware
Version
I
Sheet
2 of 4
Page Size
A3


MotorA
MotorB
MotorC
J1
MA0
MA1
GND
POWERMA
TACHOA0
TACHOA1
Modular
J2
MB0
MB1
GND
POWERMB
TACHOB0
TACHOB1
J3
MC0
MC1
GND
POWERMC
TACHOC0
TACHOC1
VCC9V_MOTOR
U1 LB1836M
U2 LB1930M
MBIN0
MB0
MB1
MBIN1
MCIN1
MC1
MC0
MCIN0
MAIN1
MAIN0
MA1
MA0
RA1C
RA1D
RA1A
RA1B
4.7k
MAPWM
MBPWM
MCPWM
TP10
TP9
TP13
TP12
TP18
TP17
TP11
TP14
TP19
R22
R26
VCC3V
C40
100nF
TACHOA0
TACHOB0
TACHOC0
TACHOA1
TACHOB1
TACHOC1
U12D 74HC14
U12A 74HC14
U12C 74HC14
U12E 74HC14
U12F 74HC14
U12B 74HC14
R1
R2
R3
R10
R11
R12
10K
R4
R5
R6
R7
R8
R9
1MEG
C1
C2
C3
C4
C5
C6
100pF
TP43
TP45
TP47
TP44
TP46
TP48
D1 1PS226
D2 1PS226
D3 1PS226
D4 1PS226
D5 1PS226
D6 1PS226
TP5
TP7
TP15
TP6
TP8
TP16
INTA0
INTB0
INTC0
DIRA
DIRB
DIRC
R87
R86
VCC_RS485
D30
RSX301L-30
R106
1.8R
Q8
BC857B
TP88
D8
LL4148
TP89
Q10
SS8550
R105
1K
C61
220uF
POWERMA
POWERMB
POWERMC
IPOWERA
IPOWERB
IPOWERC
IPOWERD
TP79
TP80
D24
BAS85
TP49
TP1
DOA
R110
TP3
REA
TP4
DEA
DIA
U3 ST485
RO
RE
DE
DI
VCC
B
A
GND
DIGIDI0
DIGIDI1
VCC5V
VCC9V
R33
R37
R34
R38
R35
R39
R32
R36
39R
10K
R49
R50
R51
R48
Q5A PUMT1
Q5B PUMT1
Q7A PUMT1
Q7B PUMT1
Q9A PUMT1
Q9B PUMT1
Q3A PUMT1
Q3B PUMT1
Q6A PUMH4
Q6B PUMH4
Q4B PUMH4
Q4A PUMH4
I_ONA
I_ONB
I_ONC
I_OND
TP21
TP25
TP29
TP33
R45
R46
R47
R44
4.7K
TP56
TP55
TP54
TP57
VCC_AVR5V
D12 1PS226
D13 1PS226
D14 1PS226
D11 1PS226
J7
J8
J9
J6
GND
IPOWERA
DIGIAI0
DIGIAI1
IPOWERB
DIGIBI0
DIGIBI1
IPOWERC
DIGICI0
DIGICI1
IPOWERD
DIGIDI0
DIGIDI1
R53
R54
R55
R52
1K
R56
R59
R58
R57
ADC_A0
ADC_B0
ADC_C0
ADC_D0
TP20
TP24
TP28
TP32
C65
C66
C67
C68
47pF
R116
R117
R118
R119
82K
NM
R60
R61
R63
R62
DIGIAI0
DIGIBI0
DIGICI0
DIGIDI0
D15 1PS226
D16 1PS226
D18 1PS226
D17 BAT54S
R18
R68
R73
R103
C20
C21
C23
C22
10pF
R112
R113
R114
R115
R75
R76
R77
R74
DIGIAI1
DIGIBI1
DIGICI1
DIGIDI1
D20 1PS226
D21 1PS226
D22 1PS226
D19 BAT54S
R66
R72
R102
R104
C25
C26
C27
C24
TP23
TP27
TP31
TP35
TP22
TP26
TP30
TP34
P_EN
INTA0
DIRA
MAIN1
MAIN0
MAPWM
INTB0
DIRB
MBIN1
MBIN0
MBPWM
INTC0
DIRC
MCIN1
MCIN0
MCPWM
DOA
REA
DEA
DIA
NM=NOT MOUNTED
LEGO
Title
LEGO MINDSTORMS NXT
Project Number
Engineer/constructer
LEGO
Drafter
LEGO
Schematic Name
Hardware
Date (YYYY/MM/DD)
Version
I
Sheet
3 of 4
Page Size
A2

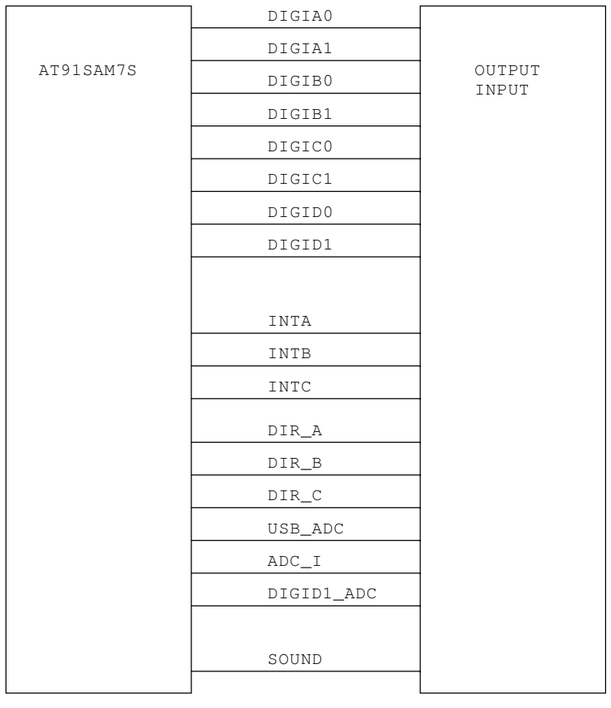
POWER BLOCK



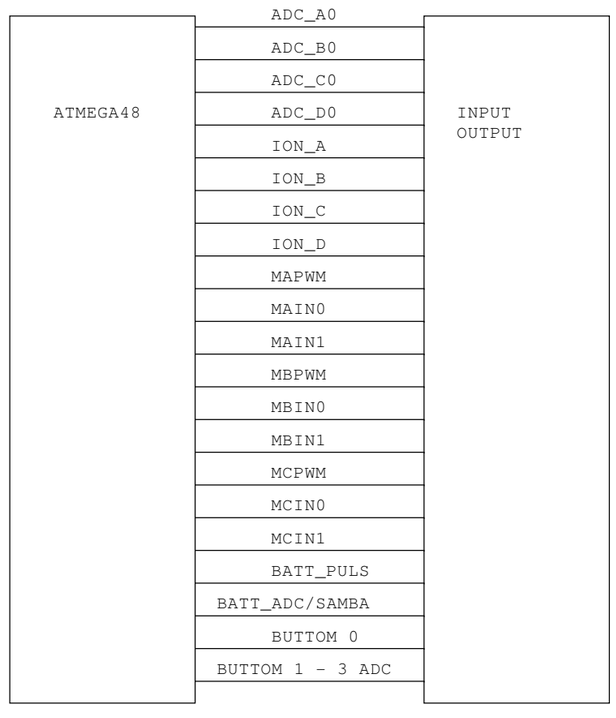
COMMUNICATION BLOCK



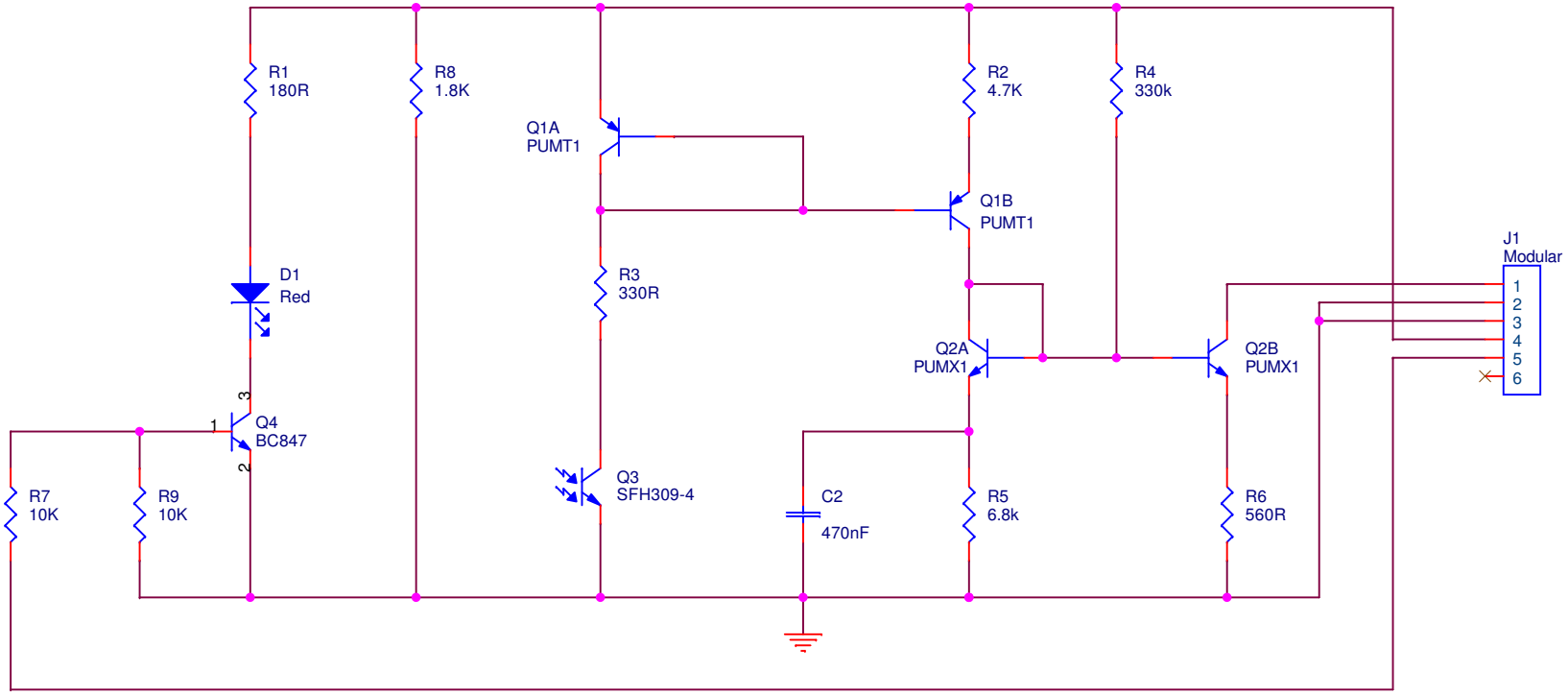
ARM BLOCK



AVR BLOCK



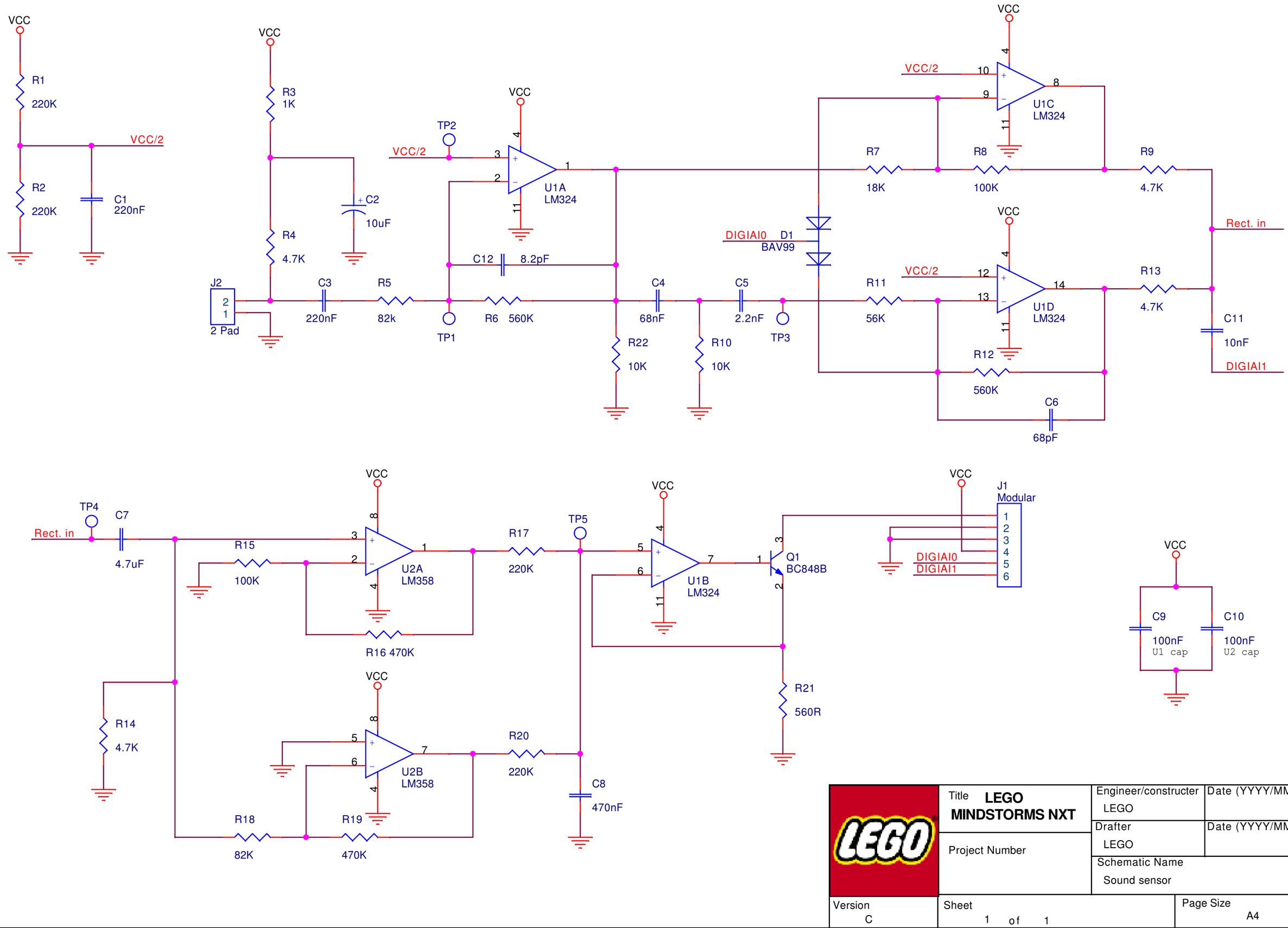
LEGO	Title: LEGO MINDSTORMS NXT	Engineer/constructer: LEGO	Date (YYYY/MM/DD)
	Project Number	Drafter: LEGO	Date (YYYY/MM/DD)
		Schematic Name: Hardware	
Version: 1	Sheet: 4 of 4	Page Size: A2	



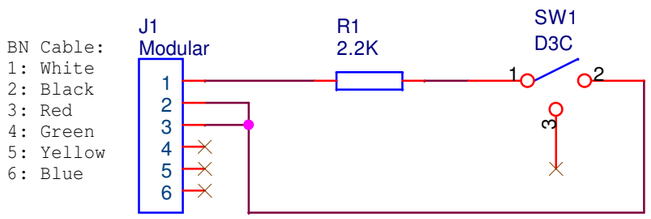
J2
DRILL 2.3 mm
J3
DRILL 2.3 mm

Drills for
LED fix

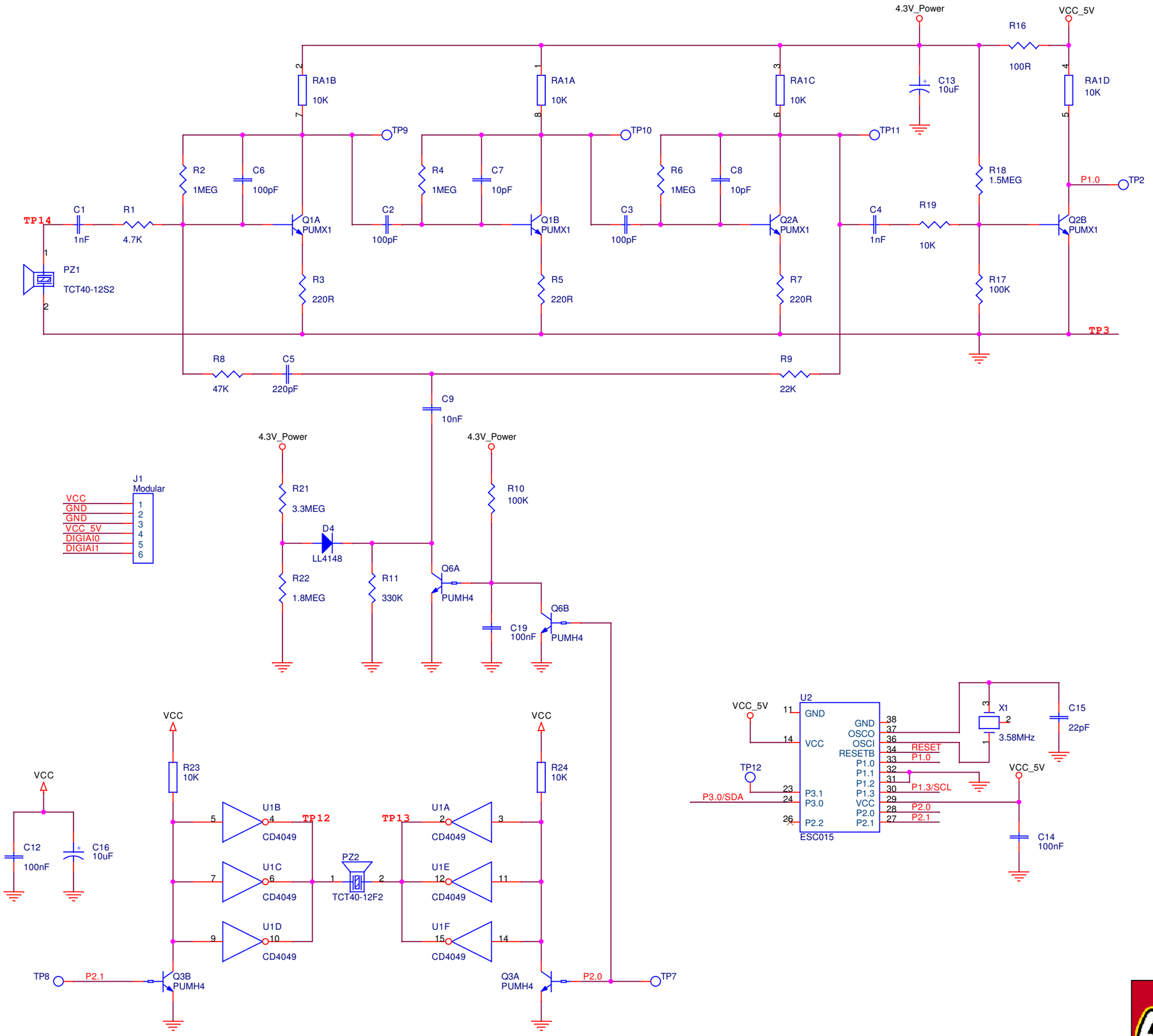
	Title **LEGO MINDSTORMS NXT**	Engineer/constructer: LEGO	Date (YYYY/MM/DD)
	Project Number	Drafter: LEGO	Date (YYYY/MM/DD)
		Schematic Name: Light sensor	
Version: F	Sheet: 1 of 1	Page Size: A4	



LEGO	Title **LEGO MINDSTORMS NXT**	Engineer/constructer LEGO	Date (YYYY/MM/DD)
	Project Number	Drafter LEGO	Date (YYYY/MM/DD)
		Schematic Name Sound sensor	
Version C	Sheet 1 of 1	Page Size A4	



	Title **LEGO MINDSTORMS NXT**	Engineer/constructer LEGO	Date (YYYY/MM/DD)
	Project Number	Drafter LEGO	Date (YYYY/MM/DD)
		Schematic Name Touch sensor	
Version A	Sheet 1 of 1	Page Size A4	



LEGO	Title **LEGO MINDSTORMS NXT**	Engineer/constructer LEGO	Date (YYYY/MM/DD)
	Project Number	Drafter LEGO	Date (YYYY/MM/DD)
		Schematic Name Ultrasonic sensor	
Version G	Sheet 1 of 1	Page Size A3	