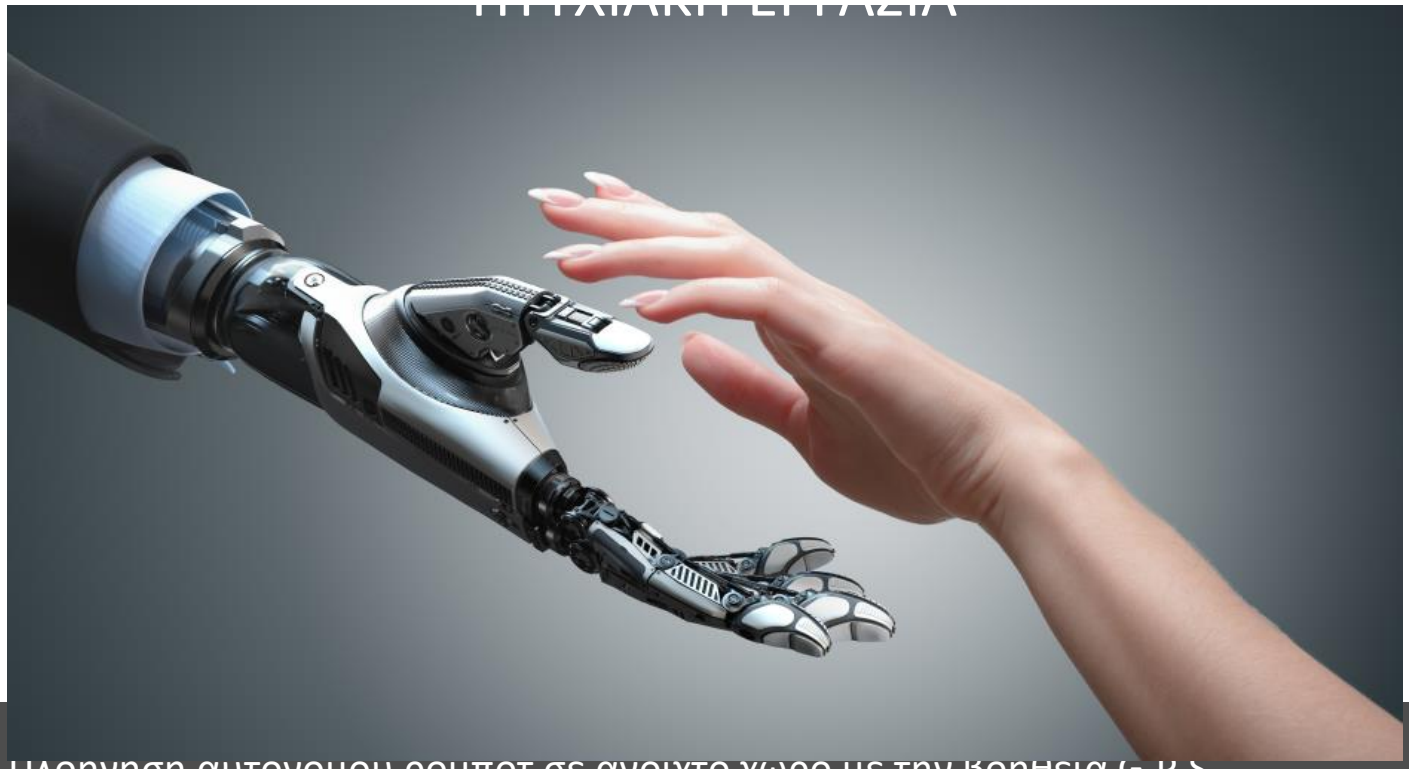
ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ



Πλοήγηση αυτονομου ρομποτ σε ανοιχτο χωρο με την βοηθεια G.P.S.

Σπουδαστής
Καραμπατζάκης Ματθαίος Α.Μ 5916

Επιβλέπων
Δρ. Καββουσανός Εμμανουήλ
Καθηγητής Σχολής Μηχανικών
Τμήματος Μηχανολόγων Μηχανικών

Ηράκλειο 2021

Ελληνικό Μεσογειακό Πανεπιστήμιο

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

1.1 Περιγραφή

Η παρακάτω εργασία εκπονήθηκε στα πλαίσια της πτυχιακής εργασίας του φοιτητή Καραμπατζάκη Ματθαίου, για το τμήμα Μηχανολόγων Μηχανικών Τ.Ε. του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Κρήτης.

Η ταχεία εξέλιξη της τεχνολογίας και τα προηγμένα συστήματα ρομποτικής και αυτοματισμού, στοχεύουν στην αφαίρεση της ανθρώπινης παρέμβασης από σχεδόν κάθε εφαρμογή, έτσι ώστε να εκμηδενιστούν τα σφάλματα λόγω του ανθρώπινου παράγοντα. Ο αυτοματισμός της πλοήγησης οχημάτων αποκτά εξαιρετικό ενδιαφέρον στην εποχή της ρομποτικής καθώς μπορεί εγγυηθεί μεγαλύτερη ασφάλεια στις μεταφορές καθώς η ανθρώπινη παρέμβαση κατά την πλοήγηση ελαχιστοποιείται, ενώ ταυτόχρονα μειώνεται και η καύσιμη ύλη που απαιτείται λόγω της αυτόματης εύρεσης βέλτιστων διαδρομών, όσο και λόγω της βέλτιστης χρήσης των δυνατοτήτων του οχήματος. Τέλος, ο ελεύθερος χρόνος του οδηγού αυξάνεται, με αποτέλεσμα ο οδηγός να μπορεί να ασχοληθεί με άλλες δραστηριότητες.

Προκειμένου να γίνει η μετάβαση από τη χειροκίνητη στην αυτοματοποιημένη οδήγηση, απαιτείται η διασύνδεση αρκετών διαφορετικών εργαλείων και τεχνολογιών, τα οποία ποικίλλουν από τη διαχείριση της τεχνολογίας GPS, τη χρήση μικροελεγκτών, τη χρήση αισθητήρων και την ολοκλήρωσή τους με το όχημα πλοήγησης κάνοντας χρήση της κατάλληλης συνδεσμολογίας. Κάτι τέτοιο βεβαίως απαιτεί σύνθετη γνώση τόσο στους τομείς της ηλεκτρονικής, των τηλεπικοινωνιών καθώς και του προγραμματισμού. Αυτό είναι και το αντικείμενο της παρούσας πτυχιακής εργασίας, το οποίο αφορά την πλοήγηση οχήματος πάνω σε δρόμους και προσβάσιμες επίγειες περιοχές κάνοντας χρήση όλων των προαναφερόμενων εργαλείων και μεθόδων.

Λέξεις κλειδιά: Ρομποτική, αυτοματισμοί, πλοήγηση, παγκόσμιο δορυφορικό σύστημα, εντοπισμός θέσης, προγραμματισμός.

1.2 Abstract

The purpose of this bachelor thesis and its research is based on the rapid technological development, advanced robotics and modern automation systems which aim to remove human intervention from almost any application, so as to eliminate errors due to the human factor.

Vehicle navigation automation is of great interest in the age of robotics as it can guarantee greater safety in transport as the human intervention during navigation is minimized. At the same time the fuel required is reduced due to the automatic finding of optimal routes, as well as due to optimal use of the capabilities of the vehicle.

Finally, the driver's free time increases, so that the driver can engage in other activities while moving.

In order to make the transition from manual to automated driving, several different tools and technologies are required, which vary from managing GPS technology, using microcontrollers, using sensors and integrating them with the navigation vehicle using the appropriate wiring. This of course requires complex knowledge in the fields of electronics, telecommunications and programming.

This is the subject of this bachelor thesis, which concerns the navigation of a vehicle on roads and accessible terrestrial areas using all the aforementioned tools and methods.

Keywords: Robotics, automation, navigation, global satellite system, positioning, programming.

1.3 Στόχοι-Μεθοδολογία

Ο βασικός στόχος της εργασίας είναι η διασύνδεση διαφόρων εξαρτημάτων και τεχνολογιών για την επίτευξη του αυτόματου ελέγχου και καθοδήγησης ενός μη επανδρωμένου οχήματος, το οποίο να μπορεί να ολοκληρώσει μια οποιαδήποτε διαδρομή, χωρίς την ανθρώπινη καθοδήγηση.

Για την πραγματοποίηση αυτού του στόχου αρχικά απαιτείται η απόκτηση δεδομένων από το παγκόσμιο δορυφορικό σύστημα εντοπισμού θέσης (GPS) με τη χρήση της κατάλληλης συσκευής-δέκτη GPS. Για τη συγκεκριμένη εργασία επιλέχθηκε η συσκευή EVK-7P της εταιρίας Ublox. Η συσκευή αυτή θα πρέπει να συνδεθεί με την μητρική πλακέτα, για την οποία επιλέχθηκε η υπολογιστική πλατφόρμα Arduino και πιο συγκεκριμένα ο μικροελεγκτής Arduino Mega 2560 έτσι ώστε να είναι δυνατή η λήψη δεδομένων από την πλατφόρμα EVK-7P και ταυτόχρονα, να μπορεί να μας παράσχει και έλεγχο του οχήματος.

Το όχημα πλοήγησης το οποίο χρησιμοποιήθηκε είναι το συναρμολογούμενο μοντέλο Rover 4WD1 Robot, λόγω του ότι παρέχει μεγάλη ευελιξία προσθήκης διαφόρων εξαρτημάτων πάνω του. Για την καθοδήγηση των κινητήρων του οχήματος επιλέχθηκε η πλακέτα Arduino Motor Shield Rev3 η οποία μπορεί να κινήσει 2 κινητήρες ταυτόχρονα και ανεξάρτητα μεταξύ τους. Για τη λειτουργία των ηλεκτροκινητήρων, τοποθετήθηκε η πλακέτα Motor Shield Rev3 πάνω στον Arduino MEGA 2560. Επιπλέον, πάνω στο όχημα τοποθετήθηκε μια οθόνη LCD Display 16X2 η οποία δίνει τη δυνατότητα παρακολούθησης των συντεταγμένων του οχήματος καθώς και εμφάνισης διαφόρων μηνυμάτων σφάλματος τα οποία μπορεί να προκύψουν κατά τη διάρκεια της πλοήγησης.

Τέλος, πάνω στο όχημα τοποθετήθηκε ο αισθητήρας BNO055 της εταιρίας Adafruit με στόχο την εμφάνιση διαφόρων μετρούμενων μεγεθών, σημαντικών για την πλοήγηση του οχήματος, όπως η γωνία περιστροφής σε τρεις άξονες καθώς και η επιτάχυνση πάνω σε αυτούς τους τρείς άξονες.

1.4 Δομή της εργασίας

Σε κάθε κεφάλαιο της παρούσας εργασίας παρουσιάζεται και η διαφορετική τεχνολογία που χρησιμοποιήθηκε για την επίτευξη του στόχου της πτυχιακής καθώς και η διασύνδεσή της με το την πλατφόρμα και το όχημα. Πιο συγκεκριμένα, στο κεφάλαιο 2 της παρούσας εργασίας πραγματοποιείται μία εκτεταμένη ανάλυση του συστήματος GPS το οποίο είναι το σύστημα που θα χρησιμοποιηθεί για τον προσδιορισμό της θέσης του οχήματος. Στο κεφάλαιο 3 αναλύεται η οικογένεια πλατφορμών Arduino με τα πλεονεκτήματα και τα μειονεκτήματα της κάθε έκδοσης με στόχο την επιλογή της πιο κατάλληλης για την ανάπτυξη του δικού μας έργου. Εν συνεχεία, στο κεφάλαιο 4 παρουσιάζεται η συσκευή GPS που χρησιμοποιήθηκε για τη λήψη των απαραίτητων δεδομένων προσανατολισμού του οχήματος στο δρόμο ενώ στο κεφάλαιο 5 παρουσιάζεται το συναρμολογούμενο όχημα πάνω στο οποίο θα γίνει η ολοκλήρωση όλων των διαφορετικών τεχνολογιών. Ακολούθως, στο κεφάλαιο 6 αναλύεται η συνδεσμολογία των εξαρτημάτων που χρησιμοποιήθηκαν προκειμένου το όχημα να έχει τις λειτουργίες και το εύρος κίνησης που θέλουμε ενώ παρουσιάζεται και ο κώδικας που αναπτύχθηκε. Στο κεφάλαιο 7 παρουσιάζεται ο αισθητήρας που χρησιμοποιήθηκε έτσι ώστε οι κινητήρες του οχήματος να έχουν αισθητήρες γωνιακής ταχύτητας. Ο αισθητήρας αυτός είναι ικανός να μετρήσει την ταχύτητα και την επιτάχυνση σε τρεις άξονες καθώς και να μας παράσχει τα δεδομένα τα οποία χρειαζόμαστε για τις μετρήσεις μας. Στο κεφάλαιο 8 γίνεται μία σύνοψη σε σχέση με το αυτοματοποιημένο όχημα που υλοποιήσαμε ενώ τέλος, στο Παράρτημα παρατίθεται ο κώδικας που χρησιμοποιήθηκε για τη διασύνδεση όλων των εξαρτημάτων και τη λειτουργία του οχήματος.

2. Το GPS

2.1 Τι είναι το GPS

Το NAVSTAR/G.P.S. (**NAV**IGATION **S**ATELLITE **T**IMING **A**ND **R**ANGING / **G**LOBAL **P**OSITIONING **S**YSTEM) η απλά **GPS** είναι ένα παγκόσμιο δορυφορικό σύστημα προσδιορισμού θέσης, χρόνου και ταχύτητας, για οποιοδήποτε σημείο στην επιφάνεια της γης η και κάτω από αυτήν, σε οποιαδήποτε χρονική στιγμή το οποίο είναι ανεξάρτητο από τις καιρικές συνθήκες. Το σύστημα σχεδιάστηκε στη δεκαετία του 1970, αναπτύχθηκε στη δεκαετία του 1980 και βρίσκεται συνεχώς υπό τον έλεγχο του Υπουργείου Άμυνας των ΗΠΑ. Αρχικά, σχεδιάστηκε για τόσο για την κάλυψη των αναγκών της ναυσιπλοΐας όσο και για στρατιωτικούς σκοπούς με στόχο να είναι δυνατός ο προσδιορισμός θέσης ενός αντικειμένου σε πραγματικό χρόνο με ακρίβεια ±10-15 μέτρα. Γρήγορα έγινε αντιληπτή η δυνατότητα χρήσης του συστήματος και για την κάλυψη πολιτικών αναγκών πλοήγησης. Η πολιτική χρήση του GPS, όπως είναι οι τοπογραφικές και γεωδαιτικές εφαρμογές υψηλής ακρίβειας ή οι χαμηλότερης ακρίβειας εφαρμογές GIS, οι εφαρμογές πλοήγησης στόλου οχημάτων, έγινε δυνατή ύστερα από απόφαση των ΗΠΑ (1983, με αφορμή κάποιο αεροπορικό δυστύχημα), σχεδόν από τα πρώτα βήματα, με πρόβλεψη για περαιτέρω βελτίωση.

Το GPS ανήκει στην κατηγορία των συστημάτων GNSS (Global Navigation Satellite Systems), δηλαδή των παγκόσμιων δορυφορικών συστημάτων πλοήγησης, όπως είναι το παρόμοιο Ρωσικό σύστημα GLONASS (GLOBAL NAVIGATION SATELLITE SYSTEMS) και το πολλά υποσχόμενο καθαρά πολιτικό Ευρωπαϊκό σύστημα GALILEO. Το GLONASS μέχρι το 2010, είχε επιτύχει κάλυψη 100% του εδάφους της Ρωσίας και τον Οκτώβριο του 2011 έγινε η αποκατάσταση 24 δορυφόρων, επιτρέποντας την πλήρη παγκόσμια κάλυψη. Σήμερα, η Ευρωπαϊκή Ένωση προχωράει στην ανάπτυξη του πρώτου πολιτικού συστήματος προσδιορισμού θέσης και πλοήγησης, του GALILEO, του οποίου η πλήρης ολοκλήρωση του συστήματος 30 δορυφόρων GALILEO (27 υπό λειτουργία και τρεις ενεργοί ανταλλακτικοί) προβλέπεται ως το 2019. Ένας από τους στόχους του GALILEO είναι η παροχή ενός συστήματος εντοπισμού θέσης υψηλής ακρίβειας στο οποίο θα μπορούν να βασιστούν τα ευρωπαϊκά κράτη, ανεξαρτητοποιώντας τα έτσι από τα

αντίστοιχα συστήματα GLONASS (Ρωσία) και GPS (ΗΠΑ), τα οποία μπορούν να απενεργοποιηθούν εν καιρώ πολέμου ή συρράξεων.

Η εποχή της δορυφορικής και διαστημικής γεωδαισίας αρχίζει ουσιαστικά στην δεκαετία του 1960. Τα τελευταία 25 χρόνια περίπου, το GPS φαίνεται να έχει επικρατήσει σε πολλές εφαρμογές.

Οι τοπογραφικές και υδρογραφικές αποτυπώσεις, οι απλοί τριγωνισμοί και τα δίκτυα πύκνωσης, τα εθνικά, ηπειρωτικά και παγκόσμια δίκτυα, οι συνδέσεις διαφορετικών συστημάτων αναφοράς, οι φωτογραμμετρικές και κτηματογραφικές αποτυπώσεις, οι χαράξεις στην οδοποιία και τα τεχνικά έργα, η μελέτη μικρομετακινήσεων κρίσιμων τεχνικών έργων καθώς επίσης και οι γεωδυναμικές εφαρμογές, όπως είναι η παρακολούθηση μικρομετακινήσεων του φλοιού της γης, αποτελούν μερικές χαρακτηριστικές εφαρμογές του GPS κυρίως όσον αφορά τον κλάδο των επιστήμων του Μηχανικού που σχετίζεται με αυτά ή παρόμοια αντικείμενα.

Εκτός από τις παραπάνω εφαρμογές υψηλής ακρίβειας, οπού η απαίτηση σε ακρίβεια κυμαίνεται από μερικά χιλιοστά του μέτρου έως και μερικά εκατοστά, αρκετές ακόμα εφαρμογές με χαμηλότερες απαιτήσεις σε ακρίβεια, από μερικές δεκάδες εκατοστά έως και μερικά μέτρα, καλύπτονται από τις δυνατότητες του GPS, όπως για παράδειγμα, η ενημέρωση χαρτών, οι εφαρμογές GIS, η πλοήγηση και ο εντοπισμός προεπιλεγμένων θέσεων.

Το σύστημα GPS αποτελείται ουσιαστικά από πομπούς σε τροχιά που είναι οι δορυφόροι GPS και από δέκτες GPS στη γήινη επιφάνεια. Ο δέκτης μπορεί να βρίσκεται σε ένα κινούμενο όχημα η ένα τοπογραφικό όργανο η ακόμα και να κρατιέται στην παλάμη του χεριού λαμβάνοντας ηλεκτρομαγνητικά σήματα που εκπέμπονται από τους ορατούς ως προς το δέκτη δορυφόρους.

Τα δορυφορικά σήματα χρησιμοποιούνται για την εκτέλεση μετρήσεων από το δέκτη, που ισοδυναμούν σε αποστάσεις μεταξύ δέκτη και δορυφόρων σε κάθε χρονική στιγμή. Οι παρατηρήσεις και άλλες πληροφορίες καταγράφονται στην μνήμη του δέκτη και επεξεργάζονται είτε εσωτερικά από το λογισμικό του δέκτη σε πραγματικό χρόνο είτε εκ των υστέρων, παρέχοντας τη θέση, η την ταχύτητα και τον χρόνο. Κατά την διάρκεια των μετρήσεων, ο δέκτης διαβάζει και ένα μήνυμα δεδομένων πλοήγησης που περιλαμβάνει απαραίτητες πληροφορίες για τον υπολογισμό της θέσης σε πραγματικό χρόνο, όπως είναι τα στοιχεία τροχιάς των δορυφόρων, από τα οποία υπολογίζονται οι

συντεταγμένες των δορυφόρων, οι παράμετροι διόρθωσης χρόνου και άλλα συστηματικά σφάλματα.

2.2 Αρχή λειτουργίας του GPS

Στον δέκτη GPS γίνεται η λήψη και η ανάλυση του λαμβανόμενου σήματος και μέσω μετρήσεων αποστάσεων μεταξύ δορυφόρου και δέκτη, προσδιορίζεται η θέση του δέκτη. Επειδή οι δέκτες GPS διαθέτουν κατά κανόνα χρονόμετρα (ρουβιδίου η καισίου) όπως οι δορυφόροι του συστήματος εκτός των ατμοσφαιρικών χρονικών καθυστερήσεων έχουμε και τις χρονικές καθυστερήσεις που οφείλονται κυρίως στο χρονόμετρο του δέκτη, αλλά και δευτερευόντως, του δορυφόρου. Έτσι κατά τον προσδιορισμό θέσης ενός δέκτη (X,Y,Z ή φ,λ,h) προστίθεται και ένας επιπλέον άγνωστος Δt, που αντιπροσωπεύει τη χρονική καθυστέρηση του χρονομέτρου του δέκτη σε σχέση με το χρόνο αναφοράς του GPS. Ο χρόνος αναφοράς του GPS έχει έναρξη την 0^hU.T.C της $5^{ης}$ Ιανουαρίου του 1980. Η προσδιοριζόμενη θέση (X, Y, Z) αναφέρεται στο Παγκόσμιο Γεωκεντρικό Σύστημα Αναφοράς 1984κ γνωστό ως WGS84.

Οι μετρήσεις του GPS διακρίνονται σε δύο βασικές κατηγορίες: σε μετρήσεις ψευδοαποστάσεων (pseudoranges) και σε μετρήσεις φάσεων (phase measurements). Ως ακριβέστερη μέθοδος λογίζεται αυτή με τις μετρήσεις φάσεων.

Υπάρχουν γενικά δυο μέθοδοι προσδιορισμού θέσης: η στατική και η κινηματική. Στο στατικό προσδιορισμό ο δέκτης είναι στάσιμος και οι παρατηρήσεις διαρκούν από λίγα λεπτά μέχρι και μερικές ώρες, ενώ στον κινηματικό ο δέκτης βρίσκεται σε κίνηση λαμβάνοντας συνεχώς το δορυφορικό σήμα.

Ο τρόπος προσδιορισμού με GPS μπορεί να είναι απόλυτος (absolute positioning), ή σχετικός (relative positioning). Στον απόλυτο εντοπισμό η θέση του δέκτη (X, Y, Z) υπολογίζεται ως προς το γεωκεντρικό σύστημα αναφοράς ενώ στο σχετικό η θέση του δέκτη καθορίζεται σε σχέση με κάποιον άλλο δέκτη (ΔX, ΔY, ΔZ). Στο σχετικό εντοπισμό αντί των πρωτογενών παρατηρήσεων, είναι δυνατό να χρησιμοποιηθούν οι λεγόμενες διαφορές (ψευδοαποστάσεων, φάσεων) συνήθως απλές, διπλές ή και τριπλές διαφορές.

Στις εφαρμογές με απαιτήσεις μεγάλης ακρίβειας, παραδείγματος χάρη αποτυπώσεις σε μεγάλες κλίμακες, γεωδαιτικά δίκτυα κάθε είδους, χρησιμοποιούνται οι τεχνικές του σχετικού προσδιορισμού (διαφορικός εντοπισμός – differential positioning).

2.3 Το δορυφορικό τμήμα και η δομή του

Το δορυφορικό σύστημα GPS περιλαμβάνει μια ολόκληρη δομή συνεχούς λειτουργίας, παρακολούθησης, ελέγχου και συντήρησης των δορυφόρων, με την ευθύνη του Υπουργείου Άμυνας των ΗΠΑ.

Συγκεκριμένα αποτελείται από τρία τμήματα:

- το δορυφορικό τμήμα
- το τμήμα ελέγχου
- το τμήμα των χρηστών

Το δορυφορικό τμήμα αποτελείται από 24-32 δορυφόρους. Οι δορυφόροι αυτοί σκεπάζουν ομοιόμορφα με το σήμα τους ολόκληρο τον πλανήτη, γεγονός που αποδεικνύει τη φιλοσοφία που κρύβεται πίσω από τη λειτουργία του συστήματος GPS, δηλαδή τη διαθεσιμότητα του σε κάθε σημείο της Γης, ώστε να μην υπάρχει κίνδυνος να αποπροσανατολιστεί ποτέ κανείς και πουθενά.

Όλοι οι δορυφόροι βρίσκονται σε ύψος 20.200 χιλιομέτρων πάνω από την επιφάνεια της θάλασσας και εκτελούν δύο περιστροφές γύρω από την Γη κάθε 24 ώρες. Η κατασκευάστρια εταιρία είναι η Rockwell International, η εκτόξευση τους πραγματοποιήθηκε από το ακρωτήριο Canaveral, ενώ η τροφοδοσία τους με ηλεκτρική ενέργεια πραγματοποιήθηκε μέσω των φωτοβολταϊκών συστημάτων που διαθέτουν.

Κάθε σειρά δορυφόρων συμπληρώνει η και αντικαθιστά σταδιακά τις προηγούμενες σειρές επειδή οι δορυφόροι έχουν ορισμένη διάρκεια ζωής. Οι δορυφόροι του συστήματος ταξινομούνται με τους εξής τρόπους:

- Σύμφωνα με την σειρά εκτόξευσης
- Σύμφωνα με τη θέση στην τροχιά
- Σύμφωνα με ένα κωδικό της NASA
- Με βάση ένα διεθνή κώδικα
- Με βάση ένα αριθμό ο οποίος ορίζει ποια εβδομάδα του P-κώδικα εκπέμπει ο δορυφόρος, που είναι και ο πιο συνηθισμένος τρόπος καταχώρησης.

Για παράδειγμα, ο δορυφόρος με ημερομηνία εκτόξευσης 23/7/1997 έχει τον κωδικό IIR-2 ως προς τη σειρά εκτόξευσης (Launch Order) φέρει την ονομασία BLOCKIIR, ως προς το

διαστημικό όχημα που φέρει τον κωδικό SVN 43 (Space Vehicle 43) και ως το μοναδικό εβδομαδιαίο τμήμα του κώδικα P που εκπέμπει, φέρει τον κωδικό PRN 13 (Pseudo Random Noise 13). Οι ονομασίες με βάση τον κωδικό SVN ή PRN είναι αυτές που χρησιμοποιούνται συνήθως.

Ο αρχικός σχεδιασμός προέβλεπε 21 δορυφόρους ενώ από τα τέλη του 1993 ο αριθμός τους είναι σταθερά πάνω από 24. Ο αριθμός των 24 δορυφόρων αποτελεί τον απαραίτητο αριθμό για την πλήρη λειτουργία του συστήματος. Με αυτόν τον τρόπο, μπορούν να παρατηρούνται ταυτόχρονα έξι έως οκτώ δορυφόροι από οποιοδήποτε σημείο της γήινης επιφάνειας με καλό ορίζοντα. Οι νεότεροι δορυφόροι παρουσιάζουν ολοένα και μεγαλύτερη αυτονομία από το τμήμα ελέγχου, αφού μπορούν με παρατηρήσεις μεταξύ τους, να προσδιορίζουν μόνοι τους τα στοιχεία τροχιάς και άλλες παραμέτρους που απαιτούνται.

Οι δορυφόροι είναι ομοιόμορφα κατανεμημένοι σε έξι τροχιακά επίπεδα, ανά 60^{o} στο ισημερινό επίπεδο και γωνία κλίσης 55^{o} ως προς το ισημερινό επίπεδο. Ο δορυφορικός σχηματισμός έχει τέτοια διάταξη ώστε από κάθε σημείο της γήινης επιφάνειας να λαμβάνεται δορυφορικό σήμα τουλάχιστον από τέσσερις δορυφόρους, θεωρώντας ότι δεν παρεμβάλλονται εμπόδια μεταξύ δέκτη και δορυφόρων. Ο μέγιστος αριθμός δορυφόρων που μπορεί να λαμβάνει ένας δέκτης, μπορεί να φτάσει και τους δώδεκα με πολύ καλό ορίζοντα. Αξίζει να σημειωθεί ότι στο πλάτος των 35^{o} και για μικρά χρονικά διαστήματα η δορυφορική κάλυψη μειονεκτεί σε σχέση με την υπόλοιπη γη.

Η περίοδος κάθε δορυφόρου είναι μισή αστρική ημέρα (μέση αστρική ημέρα = $23^{h}56^{m}4.09^{s}$ σε ηλιακό χρόνο), δηλαδή 12 ώρες σε αστρικό χρόνο ή $T=11^{h}58^{m}2.05^{s}$. Συνεπώς, η θέση κάθε δορυφόρου ή δορυφορικού σχηματισμού θα είναι ίδια στον ίδιο αστρικό χρόνο (μία φορά την ημέρα) και οι δορυφόροι θα εμφανίζονται στον ορίζοντα ενός τόπου $3^{m}55.91^{s}$ νωρίτερα κάθε ημέρα ή περίπου 4 min, όσο είναι μικρότερη η αστρική από την μέση αστρική ημέρα.

Η διάρκεια ζωής των δορυφόρων είναι περίπου 10 έτη, το βάρος τους είναι της τάξης του ενός τόνου, ενώ το μέγεθος του βασικού κορμού είναι όσο ένα πολύ μικρό δωμάτιο με τα πλαίσια των συσσωρευτών ενέργειας να εκτείνονται στα μερικά μέτρα.

Η ελλειπτική τροχιά τους παρουσιάζει πολύ μικρή εκκεντρότητα και είναι σχεδόν κυκλική με ακτίνα R=26.560 Km. Αν λάβουμε υπόψη μια αντιπροσωπευτική τιμή της γήινης σφαίρας, Rγ=6.371 Km, τότε το μέσο υψόμετρο ενός δορυφόρου είναι Rμ=20.189 Km, τιμή που αντιστοιχεί στην πλησιέστερη απόσταση δορυφόρου και χρήστη GPS που βρίσκεται στην επιφάνεια της γης.

Με βάση τα στοιχεία, εύκολα υπολογίζεται η απόσταση ενός δορυφόρου που βρίσκεται στον ορίζοντα, SA $=\sqrt{r^2 - R^2}$ = 25.785 Km και οι γωνίες β=13.87° και η συμπληρωματική της γ=76.13°. Έτσι, το δορυφορικό σήμα που ταξιδεύει με την ταχύτητα του φωτός χρειάζεται περίπου 67ms έως 86ms για να φτάσει στην επιφάνεια της γης. Η γωνιακή ταχύτητα του δορυφόρου είναι περίπου διπλάσια από την γωνιακή ταχύτητα την γης και πιο συγκεκριμένα 1.458×10^{-4}rad/sec, οπότε η ταχύτητα του δορυφόρου κατά μήκος της τροχιάς του είναι u_s= 3874m/s (1394.4 Km/h). Σε $3^m55.91^s$ ο δορυφόρος διανύει περίπου 914 Km, οπότε για ένα σταθερό σημείο στη γήινη επιφάνεια και για δορυφόρο κοντά στον ορίζοντα η γωνία που αντιστοιχεί είναι περίπου 2° ενώ για δορυφόρο κοντά στο ζενίθ η γωνία είναι περίπου 2.6°. Κατά συνέπεια, στον ίδιο χρόνο από μέρα σε μέρα ο δορυφόρος θα διαγράφει γωνία 2-2.6 μοιρών σε σχέση με ένα σταθερό σημείο στη γη.

Κάθε δορυφόρος φέρει ως βασικό εξοπλισμό ταλαντωτές η ατομικά χρονόμετρα, υπολογιστές και κεραίες τηλεπικοινωνίας. Μεταφέρει συνήθως τρία η τέσσερα ατομικά ρολόγια καισίου η και ρουβιδίου εκ των οποίων το ένα χρησιμοποιείται ως βασικό για την παραγωγή μιας θεμελιώδους συχνότητας για το δορυφορικό σήμα και την διατήρηση της κλίμακας του χρόνου ενώ τα υπόλοιπα ως εφεδρικά. Οι δορυφόροι τείνουν να αποκλίνουν από τις σχεδιασμένες τροχιές τους και υπόκεινται κατά διαστήματα σε διορθώσεις από το σύστημα ελέγχου.

2.4 Το τμήμα ελέγχου

Το τμήμα ελέγχου αποτελείται από:

- Πέντε επίγειους μόνιμους σταθμούς παρακολούθησης, συμπεριλαμβανομένου και του κεντρικού σταθμού, με γνωστές συντεταγμένες ως προς το WGS84, κατανεμημένους σε όλη την γη και συγκεκριμένα στις θέσεις Hawaii, Colorado Springs (MCS), Ascension Island, Diego Garcia, Kwajalein.
- Τρεις σταθμούς τηλεπικοινωνιών (upload station, Ground Antennas). Οι σταθμοί αυτοί βρίσκονται στις θέσεις των μόνιμων σταθμών εκτός από τους σταθμούς MCS και Hawaii. Μια ακόμα κεραία, που λειτουργεί και φροντίζει για τον έλεγχο των δορυφόρων πριν την εκτόξευση τους αλλά και ως εφεδρική, βρίσκεται στο Cape Canaveral, Florida. Κάθε σταθμός βλέπει όλους τους δορυφόρους στη διάρκεια μιας ημέρας, οπότε ο κάθε δορυφόρος είναι σε επικοινωνία τρεις φορές την ημέρα για να λάβει τα δεδομένα του μηνύματος πλοήγησης.
- Έναν κεντρικό σταθμό ελέγχου (MCS: Master Control Station) που βρίσκεται στην αεροπορική βάση Falcon στο Colorado Springs, ο οποίος είναι υπεύθυνος για τη συνολική κατάσταση και λειτουργία του δορυφορικού σχηματισμού. Ένας ακόμα εφεδρικός σταθμός ελέγχου (BUMCS: Backup MCS) βρίσκεται στο Gaithersburg, Maryland.

Οι σταθμοί παρακολούθησης είναι δέκτες GPS που φέρουν ατομικά χρονόμετρα και εκτελούν συνεχώς μετρήσεις ψευδοαποστάσεων, μεγαλύτερης ακρίβειας από ένα κοινό δέκτη. Σκοπός των σταθμών ελέγχου είναι να κάνουν πρόβλεψη για τις δορυφορικές θέσεις και για τις παραμέτρους των δορυφορικών χρονομέτρων. Η πρόβλεψη αυτή γίνεται σε δυο στάδια. Στο πρώτο στάδιο δημιουργείται η εφημερίδα αναφοράς προσεγγίζοντας τη δορυφορική τροχιά χρησιμοποιώντας τα δεδομένα των 7 προηγούμενων ημερών. Στο δεύτερο στάδιο γίνεται πρόβλεψη εφημερίδων με εκτίμηση και διόρθωση των δορυφορικών διαταραχών.

Σε περίπτωση βλάβης των σταθμών ελέγχου, οι δορυφόροι μπορούν από μόνοι τους να προβλέψουν την τροχιά τους με πιθανή όμως μείωση της ακρίβειας. Οι δορυφόροι της σειράς IIR έχουν αυτονομία 180 ημερών στο μήνυμα ναυσιπλοΐας και φέρουν δέκτες GPS

που εκτελούν μετρήσεις μεταξύ τους ώστε η εξάρτηση από το σύστημα ελέγχου να περιορισθεί στο ελάχιστο.

Η τροφοδότηση των δορυφόρων από τους σταθμούς ελέγχου γίνεται κάθε 8 ώρες. Τα δεδομένα αυτά εκπέμπονται τελικά από τους δορυφόρους στους χρήστες GPS. Λόγω του περιορισμένου αριθμού των σταθμών παρακολούθησης η ακρίβεια που παρέχει το δίκτυο είναι επαρκής για τη ναυτιλία και την πλοήγηση, αλλά ανεπαρκής για γεωδαιτικές εφαρμογές. Για τον λόγο αυτό δημιουργήθηκαν και ανεξάρτητα δίκτυα παρακολούθησης με σκοπό να εμπλουτίσουν με ακριβέστερες παρατηρήσεις το δίκτυο GPS. Τον σημαντικότερο ρόλο στον τομέα διαχείρισης και παρακολούθησης ανέλαβε από το 1994 η διεθνής υπηρεσία GPS (IGPS, International GPS Service) με ένα εκτεταμένο δίκτυο σε όλο τον κόσμο.

2.5 Το τμήμα των χρηστών

Το τμήμα των χρηστών περιλαμβάνει τους δέκτες GPS οι οποίοι λαμβάνουν, επεξεργάζονται τα σήματα και καταγράφουν τις μετρήσεις. Ο δέκτης αποτελείται από την κεραία, τον κυρίως δέκτη και τον υπολογιστή (χειριστήριο-καταγραφικό). Μέσω της κεραίας μπορεί να κεντρώνεται σε σημεία για τον προσδιορισμό της θέσης του όπως ακριβώς ένας κλασσικός θεοδόλιχος. Ο σχεδιασμός των τροχιών είναι τέτοιος, ώστε ανά πάσα χρονική στιγμή και σε οποιοδήποτε σημείο της γης, να υπάρχουν τουλάχιστον 4 ορατοί δορυφόροι που να λαμβάνονται ταυτόχρονα. Αυτός ο αριθμός δορυφόρων είναι απαραίτητος για να καταστεί δυνατός ο προσδιορισμός θέσης (X,Y,Z) ενός σημείου χρησιμοποιώντας έναν δέκτη.

2.6 Τα ηλεκτρομαγνητικά κύματα

Το ηλεκτρομαγνητικό κύμα (ηλεκτρομαγνητική ακτινοβολία) είναι η ταυτόχρονη διάδοση ενός ηλεκτρικού και ενός μαγνητικού πεδίου. Τα ηλεκτρομαγνητικά κύματα παράγονται από την επιταχυνόμενη κίνηση ηλεκτρικών φορτίων που ταλαντώνονται. Αν συνδέσουμε για παράδειγμα μια πηγή εναλλασσόμενης τάσης σε δύο μεταλλικούς αγωγούς τότε τα θετικά και αρνητικά φορτία που σχηματίζονται στα άκρα των αγωγών μεταβάλλονται ημιτονοειδώς με το χρόνο και κατά συνέπεια οι δύο αγωγοί διαρρέονται από εναλλασσόμενο ηλεκτρικό ρεύμα (ταλαντευόμενο ηλεκτρικό δίπολο, κεραία εκπομπής ηλεκτρομαγνητικών κυμάτων). Με αυτόν τον τρόπο δημιουργείται συνεχώς ηλεκτρικό και μαγνητικό πεδίο (ηλεκτρομαγνητικές διαταραχές των εντάσεων) τα οποία απομακρύνονται

στο χώρο με την ταχύτητα του φωτός. Τα ηλεκτρομαγνητικά κύματα διαδίδονται και στο κενό αλλά με σταθερή ταχύτητα c=2997924458 m/s η περίπου 300.000 km/s. Τα διανύσματα του ηλεκτρικού και μαγνητικού πεδίου είναι κάθετα μεταξύ τους και κάθετα στην διεύθυνση τους (εγκάρσιο κύμα). Τα ηλεκτρομαγνητικά κύματα προσφέρονται για ένα πλήθος εφαρμογών. Ιδιαίτερα στην μεταφορά πληροφοριών σε μεγάλες αποστάσεις (τηλεπικοινωνίες), μεταξύ των οποίων και με το GPS, όπου χρησιμοποιείται το φάσμα των ραδιοκυμάτων και μικροκυμάτων. Η μετάδοση μιας πληροφορίας και η λήψη της απαιτούν τη χρήση ενός πομπού και ενός δέκτη. Σε γενικές γραμμές η πληροφορία που πρόκειται να μεταφερθεί παράγεται στον πομπό από ένα μηχανισμό και μετατρέπεται σε ηλεκτρικό ρεύμα το οποίο στη συνέχεια προστίθεται σε μια υψίσυχνη συχνότητα (ηλεκτρικό ρεύμα) που καλείται φέρουσα συχνότητα η φέρων κύμα. Το ηλεκτρικό ρεύμα-διαμορφωμένο κύμα, οδηγείται στην κεραία εκπομπής η οποία εκπέμπει ηλεκτρομαγνητικά κύματα της ίδιας μορφής. Η διαδικασία της πρόσθεσης είναι πολύπλοκη, γίνεται με ψηφιακό τρόπο και λέγεται διαμόρφωση. Ένας δέκτης, μέσω της κεραίας του, λαμβάνει το διαμορφωμένο σήμα και το επεξεργάζεται κατάλληλα, ώστε να αντλήσει την πληροφορία που μεταφέρεται.

Ο όρος σήμα σαν μια γενικότερη έννοια είναι μια ροή πληροφοριών που περιγράφουν τη συμπεριφορά η τη φύση πολλών φαινομένων. Ένα μήνυμα από γράμματα και αριθμούς, ο ήχος, μια εικόνα, ένα video, ένα ηλεκτροκαρδιογράφημα αποτελούν μερικά παραδείγματα σημάτων.

Μαθηματικά τα σήματα μπορούν να περιγράφουν χρησιμοποιώντας συναρτήσεις ως προς μια ή περισσότερες ανεξάρτητες μεταβλητές, συνήθως με συναρτήσεις εξαρτώμενες από το χρόνο ή τη θέση. Ένα απλό ημιτονοειδές σήμα δεν μπορεί στη μορφή που είναι να μεταφέρει καμία πληροφορία. Εάν με κάποιο συγκεκριμένο τρόπο μεταβάλλουμε μια βασική του παράμετρο, δηλαδή το πλάτος ή τη συχνότητα ή τη φάση όπως συμβαίνει στην περίπτωση παραγωγής των δορυφορικών σημάτων του GPS, τότε το διαμορφωμένο σήμα μεταφέρει μια πληροφορία.

Η ταχύτητα και η διεύθυνση του δορυφορικού σήματος μεταβάλλονται κατά τη διάδοση τους στην ατμόσφαιρα λόγο διασποράς και διάθλασης. Οι ατμοσφαιρικές επιδράσεις στο σήμα του GPS είναι σημαντικές και αντιμετωπίζονται με ιδιαίτερη προσοχή.

2.7 Διαμόρφωση δορυφορικού σήματος

Το ατομικό ρολόι του δορυφόρου, εκτός από τη διατήρηση της κλίμακας του χρόνου, παράγει μια θεμελιώδη συχνότητα f=10.23MHz, από την οποία προκύπτουν οι δύο βασικές συμφασικές φέρουσες συχνότητες ή κύματα.

Η συχνότητα L_1=154 με f=154×10.23MHz=1575.42MHz με μήκος κύματος λ= 19.03cm

Η συχνότητα L_2=120 με f=120×10.23MHz=1227.60MHz με μήκος κύματος λ= 24.42cm

Ο προσδιορισμός θέσης σε πραγματικό χρόνο απαιτεί τη μέτρηση αποστάσεων μεταξύ δεκτών και δορυφόρων. Επειδή οι μετρήσεις είναι ‘’μιας κατεύθυνσης’’, δηλαδή τα σήματα εκπέμπονται μόνο κατά τη φορά ‘’δορυφόροι -> δέκτες’’, χρησιμοποιούνται δυο εκπεμπόμενοι μετρητικοί κώδικες, ο C/A (Coarse Acquisition Code) και ο P (Precision Code) που είναι δυαδικές ακολουθίες από κάποιους αλγόριθμους.

Οι κώδικες αυτοί δεν μπορούν όμως να μεταδοθούν σε μεγάλες αποστάσεις και για αυτό το λόγο προστίθενται πάνω τα δυο σήματα της δέσμης L και L_1 (διαμόρφωση από C/A και P(Y)) και την L_2 (διαμόρφωση μόνο από P(Y)). Η διαμόρφωση είναι τέτοια, ώστε να είναι δυνατή η μέτρηση του χρόνου διάδοσης ή του χρόνου ταξιδιού του σήματος (travel time) από το δορυφόρο στο δέκτη. Ο C/A και ο P είναι κώδικες ψευδοτυχαίου θορύβου (PRNcodes, Pseudo Random Noise). Ταυτόχρονα, οι δυο φορείς διαμορφώνονται και από το μήνυμα πλοήγησης ή μήνυμα ναυσιπλοΐας (navigation message) ή μήνυμα δεδομένων (Data) για την παροχή πρόσθετων πληροφοριών, όπως είναι τα στοιχεία της τροχιάς των δορυφόρων (δορυφορική εφημερίδα) και παράμετροι για το συγχρονισμό των ρολογιών.

Κάθε δορυφόρος GPS εκπέμπει ένα εξαιρετικά σύνθετο σήμα. Οι υψηλές φέρουσες συχνότητες, της τάξης του 1.5GHz της δέσμης L (1-2 GHz), δεν έχουν επιλεγεί τυχαία, ούτε το είδος των συγκεκριμένων κωδικών και ο τρόπος διαμόρφωσης. Παράμετροι που ελήφθησαν υπόψη στο σχεδιασμό του σήματος ήταν η ακρίβεια του προσδιορισμού θέσης και ταχύτητας σε πραγματικό χρόνο, η παγκόσμια κάλυψη, η ταυτόχρονη εξυπηρέτηση πολλών χρηστών, η μη απαίτηση ατομικών χρονομέτρων από τους δέκτες GPS, η ανθεκτικότητα των δορυφορικών σημάτων σε παρεμβολές, καθώς και η μείωση της επίδρασης των σφαλμάτων της ιονόσφαιρας. Ας σημειωθεί ότι η χρήση δυο συχνοτήτων (L_1 και L_2) επιτρέπει την εξάλειψη της ιονοσφαιρικής επίδρασης.

Για τη διαμόρφωση των φορέων L_1 και L_2, χρησιμοποιούνται δυο διακριτές τεχνικές/μέθοδοι: η μέθοδος της δυαδικής διφασικής διαμόρφωσης (binary biphase modulation, binary phase-shift keying - BSPK), που αφορά την διαμόρφωση ενός ημιτονοειδούς φορέα από μια δυαδική/ψηφιακή ακολουθία, και η μέθοδος 'modulo 2' πρόσθεσης που αφορά την πρόσθεση δυο δυαδικών ακολουθιών.

Σύμφωνα με τη μέθοδο BSPK ένας φορέας σταθερής συχνότητας, για παράδειγμα ο L_1, διαμορφώνεται ως προς τη φάση (αλλάζει η φάση για να μεταφέρει πληροφορία) από μια ακολουθία δυαδικών ψηφίων 0 και 1, έτσι ώστε κάθε αλλαγή στην ακολουθία των ψηφίων από 0 σε 1 ή από 1 σε 0, να επιφέρει αλλαγή στη φάση του φορέα κατά 180^o, διαφορετικά η φάση του φορέα παραμένει αμετάβλητη. Έτσι, η διαμόρφωση της φάσης του φορέα έχει δυο καταστάσεις: την κανονική κατάσταση φάσης (normal state) όπου η φάση δεν έχει αλλάξει και η οποία αντιστοιχεί στον αριθμό +1 και την αντεστραμμένη/κατοπτρική κατάσταση (mirror image state), η οποία αντιστοιχεί στον αριθμό -1 όπου η φάση του φορέα έχει αλλάξει κατά 180^o. Η διαμόρφωση των δυαδικών 0 και 1 μιας ακολουθίας πάνω στο φορέα υλοποιείται με πολλαπλασιασμό του σήματος του φορέα είτε με +1 είτε με -1.

Επειδή ο φορέας L_1 διαμορφώνεται από δυο κώδικες, τον C/A και τον P, η διαδοχική υπέρθεση δεν είναι μοναδική και έτσι ο φορέας L_1 διαχωρίζεται πριν διαμορφωθεί, με έναν διαχωριστή τάσης, σε δυο συνιστώσες εκ των οποίων η μια μετατίθεται ηλεκτρονικά ώστε να προηγείται κατά 90^o ως προς τη φάση σε σχέση με άλλη συνιστώσα (Phase quadrature, Quasi Phase Shift Keying). Η συνιστώσα που προηγείται κατά 90^o διαμορφώνεται από το αποτέλεσμα της άθροισης (D+C/A), ενώ η άλλη από το αποτέλεσμα (D+P). Τα σήματα των διαμορφωμένων φορέων συντίθενται ηλεκτρονικά και το τελικό σήμα εκπέμπεται από την κεραία του δορυφόρου.

Ο δέκτης GPS λαμβάνει τα σήματα από όλους τους ορατούς δορυφόρους για επεξεργασία-ανάκτηση των επιμέρους συνιστωσών και την εκτέλεση των μετρήσεων. Όταν το λαμβανόμενο σήμα πολλαπλασιαστεί με το αντίγραφο του κώδικα (αφού γίνει η συσχέτιση) ή και ακόμα αν το λαμβανόμενο σήμα πολλαπλασιαστεί με τον εαυτό του (τετραγωνισμός-squaring), τότε θα προκύψει το σήμα χωρίς τον συγκεκριμένο κώδικα.

Το εκπεμπόμενο δορυφορικό σήμα, έτσι όπως διαμορφώνεται, έχει χαρακτηριστικά διευρυμένου φάσματος στο πεδίο των συχνοτήτων (spread spectrum). Ο μηχανισμός

διευρυμένου φάσματος είναι μια μορφή ασύρματης επικοινωνίας όπου το εύρος ζώνης των συχνοτήτων (bandwidth) που χρησιμοποιείται, μεταβάλλεται σκόπιμα και είναι μεγαλύτερο από το ελάχιστο που απαιτείται για να μεταδοθεί η πληροφορία. Η τεχνική αυτή προσφέρεται για να υπάρχει ισχυρή ανθεκτικότητα σε παρεμβολές από σήματα ίδιας συχνότητας και για λόγους ασφάλειας της επικοινωνίας από το σύστημα ελέγχου (σκόπιμη παρεμβολή σφαλμάτων).

2.8 Οι κώδικες PRN

Οι κώδικες PRN είναι σήματα που χρησιμοποιούνται για τη μέτρηση των ψευδοαποστάσεων. Είναι στην ουσία ακολουθίες των δυαδικών αριθμών 0 και 1 (binary bits ή chips) και μια πρώτη εικόνα τους δείχνει ότι η ακολουθία τους είναι τυχαία. Έχουν τα χαρακτηριστικά του τυχαίου θορύβου αλλά δεν είναι καθόλου τυχαίοι. Οι ακολουθίες των κωδικών παράγονται από συγκεκριμένους μαθηματικούς αλγόριθμους σε ειδικές διατάξεις (feedback shift registers, FBSR) και επαναλαμβάνονται ύστερα από κάποιο χρονικό διάστημα. Ας σημειωθεί ότι δεν θα πρέπει να συγχέεται η συχνότητα, η περίοδος και το μήκος κύματος του συνολικού κώδικα με τα αντίστοιχα του κάθε ψηφίου του. Η συχνότητα παραγωγής των ψηφίων, λέγεται ψηφιακός ρυθμός (chipping rate) και εκφράζεται συνήθως σε ψηφία ανά δευτερόλεπτο αντί σε Hz. Κάθε στοιχείο του κώδικα έχει συγκεκριμένη διάρκεια και μήκος (chipping length) που προκύπτει αν η διάρκεια του πολλαπλασιαστεί με την ταχύτητα του φωτός. Ο όρος chip αντί του όρου bit δίνει έμφαση στο γεγονός ότι τα ψηφία 0 και 1 δεν είναι δεδομένα (data) όπως στην περίπτωση του μηνύματος πλοήγησης.

Οι κώδικες είναι μοναδικοί για κάθε δορυφόρο και μπορούν να αναπαράγονται ακριβώς οι ίδιοι και στον δέκτη ανεξάρτητα από την παραγωγή τους στο δορυφόρο. Ο δέκτης μπορεί να συγκρίνει το λαμβανόμενο κώδικα με ένα αντίγραφο που παράγει ο ίδιος. Ο χρόνος που απαιτείται για την ταύτιση αντιστοιχεί στο χρόνο ταξιδιού του σήματος και συνεπώς στην ψευδοαπόσταση. Η ταύτιση είναι μια διαδικασία χρονικής μετατόπισης του αντιγράφου ως προς τον αντίστοιχο λαμβανόμενο κώδικα. Στην ουσία η συσχέτιση είναι μια σύγκριση ψηφίο με ψηφίο στα τμήματα του κώδικα. Το αποτέλεσμα κάθε φορά οδηγεί σε υψηλή η χαμηλή συσχέτιση με ανάλογη μετατόπιση. Το ζητούμενο είναι να επιτευχθεί η μέγιστη δυνατή ταύτιση.

Οι πρωτογενείς μετρήσεις στους κώδικες PRN είναι μετρήσεις χρονικών διαστημάτων, χρόνος λήψης στον δέκτη – χρόνος εκπομπής στον δορυφόρο, που πολλαπλασιάζονται με την ταχύτητα του φωτός και μετατρέπονται σε ψευδοαπόσταση.

Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια, οι κώδικες PRN είναι δυο ειδών: Ο κώδικας C/A (Coarse/Acquisition Code ή Clear Access) που είναι σε ελεύθερη χρήση και ο κώδικας P (Precise ή Protected) που είναι διαθέσιμος μόνο σε εξουσιοδοτημένους χρήστες. Ο C/A προστίθεται μόνο στον φορέα L1 ενώ ο P και στους δυο φορείς (L1 και L2). Το μήνυμα πλοήγησης μεταδίδεται και με τους δυο φορείς. Κάθε δορυφόρος παράγει και εκπέμπει μια μοναδική ακολουθία, διαφορετική από τους άλλους, τόσο για τον C/A όσο και για τον P. Αν υποθέσουμε ότι έχουμε 32 δορυφόρους GPS, τότε θα έχουμε 32 διαφορετικούς κώδικες C/A και 32 διαφορετικούς κώδικες P.

Η ακρίβεια μέτρησης των ψευδοαποστάσεων εξαρτάται από το μήκος του κύματος και είναι ενδεικτικά της τάξης του 0.5% έως 1% του μήκους κύματος, δηλαδή στην περίπτωση του κώδικα C/A, που έχει μήκος κύματος περίπου 300m, περίπου 1,5 έως 3m ή και καλύτερη καθώς εξελίσσεται η τεχνολογία των δεκτών και οι τεχνικές μετρήσεων. Αντίστοιχα το μήκος κύματος του κώδικα P είναι περίπου 30m, συνεπώς ο P προσφέρεται για μετρήσεις ψευδοαποστάσεων με ακρίβεια 10 φορές μεγαλύτερη σε σχέση με το C/A, δηλαδή ακρίβεια ψευδοαποστάσεων της τάξης των 15 έως 30cm.

Λόγω του μικρότερου ρυθμού του κώδικα P που είναι ίσος με το 1/10 του αντίστοιχου ρυθμού του κώδικα C/A (10,23MHz-1,023MHz), έχει και 10 φορές μικρότερη διάρκεια (περίπου 0,1μs =100ns ή ακριβέστερα 1/10,23 = 97,8 ns). Έτσι ο κώδικας P θα χρειαζόταν περίπου 266,4 ημέρες για να επαναληφθεί. Προφανώς κάτι τέτοιο δεν θα μπορούσε να συμβαίνει, καθώς θα χρειαζόταν πολύς χρόνος για την συσχέτιση και αυτό που γίνεται τελικά είναι και ο διαχωρισμός της συνολικής ακολουθίας σε έναν αριθμό 38 διαφορετικών εβδομαδιαίων τμημάτων. Από αυτά, τα 322 τμήματα αντιστοιχούν σε συγκεκριμένους δορυφόρους ενώ τα υπόλοιπα διατίθενται για άλλες χρήσεις του συστήματος. Ο αριθμός του εβδομαδιαίου τμήματος χαρακτηρίζει τον αριθμό PRN του κάθε δορυφόρου.

Τα μεσάνυχτα του Σαββάτου προς την Κυριακή κάθε εβδομάδα όλα τα τμήματα των κωδικών καθώς και το μήνυμα ναυσιπλοΐας επαναλαμβάνονται από την αρχή, ανεξάρτητα από την κατάσταση μετάδοσης στην οποία βρίσκονται. Συνεπώς, ο κώδικας P

επαναλαμβάνεται και αυτός κάθε βδομάδα. Επίσης χρησιμοποιείται αντίστοιχος αριθμός διαφορετικών C/A κωδίκων.

Ο κώδικας C/A είναι απαραίτητος επειδή εκτός του ότι παρέχει τη δυνατότητα για γρήγορη πρόσβαση στον κώδικα P, επιτρέπει και τον ικανοποιητικό συγχρονισμό μεταξύ όλων των δεκτών που παρατηρούν ταυτόχρονα, έτσι ώστε οι μετρήσεις που καταγράφονται από τους δέκτες να είναι με ικανοποιητική ακρίβεια ταυτόχρονες.

Ο νέος κώδικας L2C θα προσφέρει τη δυνατότητα διόρθωσης της ιονόσφαιρας σε πραγματικό χρόνο αυξάνοντας έτσι την ακρίβεια του απόλυτου προσδιορισμού θέσης κατά μερικά μέτρα και ακόμα αναμένεται να αυξήσει την ποιότητα επανάκτησης των συνιστωσών του φορέα L2 σε σχέση με τις έως τώρα τεχνικές συσχέτισης. Ας σημειωθεί ότι η συσχέτιση στον φορέα L1 (code correlation) λόγω του γνωστού κώδικα C/A είναι ποιοτικά καλύτερη σε σχέση με άλλες τεχνικές που εφαρμόζονται για τον φορέα L2, λόγω του P(Y).

Η διόρθωση του δορυφορικού χρόνου ως προς την κλίμακα χρόνου του GPS γίνεται με βάση τις εκπεμπόμενες χρονικές παραμέτρους του μηνύματος πλοήγησης. Υπάρχει, πάντως και η δυνατότητα, σε μια εκ των υστέρων επεξεργασία, να χρησιμοποιηθούν χρονικές παράμετροι υψηλής ακρίβειας, όπως οι παράμετροι διόρθωσης IGS.

2.9 Σκόπιμη μείωση της ακρίβειας

Η παραμόρφωση του σήματος του GPS εφαρμόζεται από τις στρατιωτικές υπηρεσίες των ΗΠΑ, με σκοπό τη μείωση της ακρίβειας του συστήματος GPS, όταν αυτό χρησιμοποιείται από μη εξουσιοδοτημένους χρήστες.

Η παραμόρφωση του σήματος επιτυγχάνεται με δυο τρόπους. Ο ένας τρόπος είναι με την μέθοδο αντί-εξαπάτησης AS (Anti-Spoofing) η οποία αποκρύπτει τον κώδικα P, πολλαπλασιάζοντας τον με κάποιο μυστικό κώδικα W, με σκοπό την προστασία του από εχθρική εξαπάτηση. Ο κώδικας P δεν είναι σε ελεύθερη χρήση από το σύστημα, αλλά μεταδίδεται ως κώδικας Y (Y1, Y2) μέσω μιας διαδικασίας απόκρυψης που προκύπτει από τον συνδυασμό του P με τον μυστικό κώδικα W.

Η άμεση πρόσβαση στον P είναι δυνατή μόνο σε εξουσιοδοτημένους χρήστες στους οποίους είναι γνωστός ο μυστικός αλγόριθμος. Παρόλα αυτά, η εξέλιξη των τεχνικών είναι τέτοια που καθιστά δυνατή και τη μέτρηση του P χωρίς να είναι γνωστός ο Y, με κάπως

μικρότερη ακρίβεια. Οι γεωδαιτικοί δέκτες που προσαρμόζονται στην κατάσταση AS, δηλαδή αναγνωρίζουν αν είναι ενεργή ή όχι και ανάλογα εφαρμόζουν τις κατάλληλες τεχνικές συσχέτισης για την επανάκτηση των συνιστωσών του δορυφορικού σήματος.

Η σημασία της κατάστασης AS για τον προσδιορισμό θέσης υψηλής ακρίβειας, όπου χρησιμοποιούνται οι γεωδαιτικοί δέκτες, είναι μικρή, επειδή για τις υψηλές ακρίβειες που απαιτούνται, χρησιμοποιούνται οι παρατηρήσεις φάσης και όχι οι ψευδοαποστάσεις από τον κώδικα P. Παρόλα αυτά, για κάποιες εφαρμογές, η μέτρηση ψευδοαποστάσεων βοηθά και διευκολύνει σημαντικά τον προσδιορισμό της θέσης μέσω της εφαρμογής κατάλληλων αλγοριθμικών τεχνικών.

Ο δεύτερος τρόπος είναι η μέθοδος της επιλεκτικής διαθεσιμότητας SA (Selective Availability). Η μέθοδος αυτή ίσχυσε στο διάστημα Ιούλιος 1991-Μάιος 2000. Με τον μηχανισμό αυτό η μείωση της ακρίβειας επιβάλλεται με δυο διαφορετικές τεχνικές:

Α) Με την τεχνική epsilon (ε) μέσω της μείωσης της ακρίβειας των παραμέτρων της εκπεμπόμενης δορυφορικής εφημερίδας, με αποτέλεσμα οι συντεταγμένες των δορυφόρων να υπολογίζονται με μεγάλη αβεβαιότητα (σφάλματα στις παραμέτρους της τροχιάς του δορυφόρου).

Β) Με την τεχνική delta (δ) μέσω της μεταβολής της συχνότητας εξόδου του δορυφορικού χρονομέτρου, επιβάλλοντας στην ουσία ένα σφάλμα ρολογιού με άμεση αντανάκλαση στις παρατηρήσεις των ψευδοαποστάσεων.

Τα σφάλματα της κατάστασης SA στις παρατηρήσεις εμφανίζουν μια περιοδικότητα της τάξης των μερικών ωρών και μερικών λεπτών αντιστοίχως και επηρεάζουν τις ψευδοαποστάσεις και τις φάσεις με σφάλματα της τάξης των μερικών δεκάδων μέτρων.

Ο απόλυτος προσδιορισμός θέσης με ένα δέκτη σε πραγματικό χρόνο, επηρεάζεται άμεσα από την κατάσταση SA. Για να ξεπεράσει το πρόβλημα αυτό θα πρέπει είτε ο χρήστης να είναι εξουσιοδοτημένος και άρα να διαβάζει και τις κρυπτογραφημένες διορθώσεις είτε να χρησιμοποιεί την μέθοδο του σχετικού προσδιορισμού θέσης όπου τα σφάλματα αυτά σχεδόν απαλείφονται, ιδιαίτερα με χρήση των εφημερίδων ακριβείας και όχι των εκπεμπόμενων εφημερίδων.

Με ανενεργή την κατάσταση SA, η ακρίβεια του απόλυτου προσδιορισμού θέσης στη λύση πλοήγησης, είναι της τάξης των μερικών μέτρων, ενδεικτικά 5-15m. Με ενεργή την κατάσταση SA, η ακρίβεια υποβιβάζεται από το τμήμα ελέγχου στα 100m για την οριζόντια θέση (φ, λ), στα 156m στο γεωμετρικό υψόμετρο (h) και στα 340 ns στο χρόνο για το επίπεδο εμπιστοσύνης 95%.

3. Ο μικροελεγκτής Arduino

3.1 Εισαγωγή στο μικροελεγκτή Arduino

Εικόνα 3.1.1 Το λογότυπο της εταιρίας

Το Arduino είναι ένας μικροελεγκτής μονής πλακέτας ανοικτού κώδικα, δηλαδή μια απλή μητρική πλακέτα, με ενσωματωμένο μικροελεγκτή, η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring. Η Wiring ουσιαστικά πρόκειται για την γλώσσα προγραμματισμού C++ με κάποιες απλοποιήσεις και αλλαγές μαζί με ένα σύνολο από βιβλιοθήκες υλοποιημένες επίσης στην γλώσσα προγραμματισμού C++. Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με ηλεκτρονικό υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, PureData, Supercollider. Επιπλέον, το Arduino επιτρέπει στον χρήστη την πρωτοτυποποίηση ηλεκτρονικών κυκλωμάτων γρήγορα και εύκολα, αρκεί να έχει στοιχειώδεις γνώσεις ηλεκτρονικών και μια μικρή εμπειρία με τον προγραμματισμό. Οι περισσότερες εκδώσεις του Arduino μπορούν να αγοραστούν προ-συναρμολογημένες, αλλά επιπλέον δίνεται η δυνατότητα σε όσους θέλουν να το συναρμολογήσουν μόνοι τους, εφόσον το διάγραμμα και οι πληροφορίες είναι ελεύθερες για τους χρήστες.

3.2 Ιστορική αναδρομή

Το 2005, στην κωμόπολη Ιβρέα της επαρχίας Τορίνο στην περιοχή Πεδεμόντιο της βορειοδυτικής Ιταλίας, ξεκίνησε ένα σχέδιο από μαθητές προκειμένου να φτιαχτεί μια συσκευή για τον έλεγχο προγραμμάτων διαδραστικών σχεδίων, η οποία θα ήταν πιο φθηνή από άλλα πρωτότυπα συστήματα τα οποία ήταν διαθέσιμα εκείνη την περίοδο. Οι ιδρυτές Massimo Banzi και David Cueartielles ονόμασαν το σχέδιο Arduino of Ivrea και ξεκίνησαν να παράγουν πλακέτες σε ένα μικρό εργοστάσιο της περιοχής.

3.3 Περιγραφή του μικροελεγκτή Arduino

Το ηλεκτρονικό κύκλωμα της πλατφόρμας βασίζεται στον μικροελεγκτή ATmega της Atmel και του οποίου όλα τα σχέδια, καθώς και το software που χρειάζεται για την λειτουργία του, διανέμονται ελεύθερα και δωρεάν έτσι ώστε να μπορεί να χρησιμοποιηθούν για κατασκευές από τον καθένα. Όταν κατασκευαστεί, μπορεί να συμπεριφερθεί σαν ένας μικροσκοπικός ηλεκτρονικός υπολογιστής, αφού ο χρήστης μπορεί να συνδέσει επάνω του πολλαπλές μονάδες εισόδου/εξόδου και να προγραμματίσει τον μικροελεγκτή να δέχεται δεδομένα από τις μονάδες εισόδου, να τα επεξεργάζεται και να στέλνει κατάλληλες εντολές στις μονάδες εξόδου. Όλες οι πλακέτες περιλαμβάνουν ένα γραμμικό ρυθμιστή τάσης 5volt και ένα κρυσταλλικό ταλαντωτή 16MHz (η κεραμικό αντηχητή σε κάποιες παραλλαγές). Ο μικροελεγκτής είναι από κατασκευής προγραμματισμένος με ένα bootloader, έτσι ώστε να μην χρειάζεται εξωτερικός προγραμματιστής.

Το κύριο πλεονέκτημα του Arduino είναι ότι γύρω του υπάρχει μια πολύ μεγάλη κοινότητα που το υποστηρίζει και η οποία έχει δημιουργήσει, συντηρεί και επεκτείνει μια ανάλογου μεγέθους γνωσιακή βάση online στο internet.

3.4 Η ενσωματωμένη μνήμη του Arduino

Το Arduino βασίζεται στον ATmega328, έναν 8-bit RISC μικροελεγκτή, τον οποίο χρονίζει στα 16MHz. Ο ATmega 328 διαθέτει ενσωματωμένη μνήμη τριών τύπων:

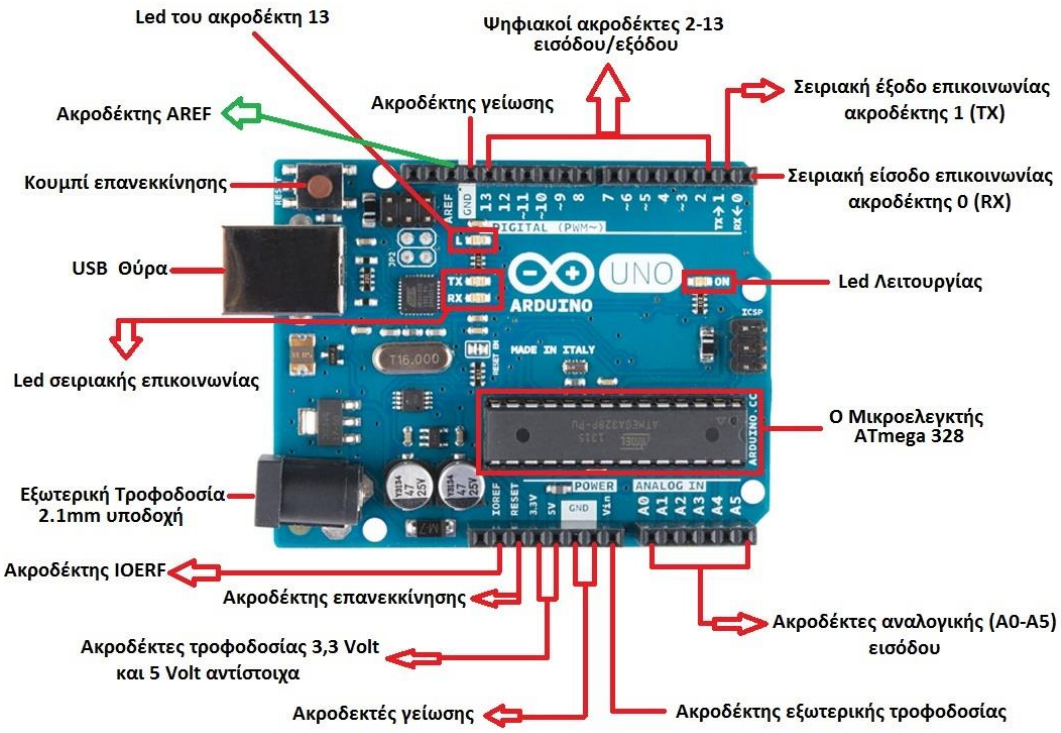
- 2Kb μνήμης SRAM που είναι η ωφέλιμη μνήμη που χρησιμοποιούν τα προγράμματα για να αποθηκεύσουν μεταβλητές και πίνακες κατά το runtime. Όπως και σε έναν υπολογιστή, αυτή η μνήμη χάνει τα δεδομένα της όταν η παροχή του ρεύματος στο Arduino σταματήσει ή αν γίνει επανεκκίνηση.

- 1Kb μνήμης EEPROM η οποία μπορεί να χρησιμοποιηθεί για εγγραφή/ανάγνωση δεδομένων ανά byte από τα προγράμματα κατά το runtime. Σε αντίθεση με την SRAM, η EEPROM δεν χάνει τα περιεχόμενα της με την απώλεια τροφοδοσίας η την επανεκκίνηση, οπότε είναι σαν το ανάλογο του σκληρού δίσκου.

- 32Kb μνήμης Flash, από τα οποία τα 2Kb χρησιμοποιούνται από το firmware του Arduino που έχει εγκαταστήσει ήδη ο κατασκευαστής του. Το firmware, που στην ορολογία του Arduino ονομάζεται bootloader, είναι αναγκαίο για την εγκατάσταση των προγραμμάτων στον μικροελεγκτή μέσω της θύρας USB, χωρίς δηλαδή να χρειάζεται εξωτερικός hardware programmer. Τα υπόλοιπα 30Kb της μνήμης Flash χρησιμοποιούνται για την αποθήκευση αυτών ακριβώς των προγραμμάτων, αφού πρώτα μεταγλωττιστούν στον υπολογιστή. Η μνήμη Flash, όπως και η EEPROM δεν χάνει τα περιεχόμενα της με την απώλεια τροφοδοσίας η επανεκκίνησης.

3.5 Είσοδοι-Έξοδοι της πλακέτας

Καταρχήν το Arduino διαθέτει σειριακό πρωτόκολλο επικοινωνίας. Ο μικροελεγκτής ATmega υποστηρίζει σειριακή επικοινωνία, την οποία το Arduino προωθεί μέσα από έναν ελεγκτή Serial-over-USB ώστε να μπορεί να συνδέεται με τον υπολογιστή μέσω USB θύρας. Η σύνδεση αυτή χρησιμοποιείται για τη μεταφορά των υλοποιημένων προγραμμάτων από τον υπολογιστή στο Arduino αλλά και για αμφίδρομη επικοινωνία του Arduino με τον υπολογιστή μέσα από το πρόγραμμα την ώρα που εκτελείται.



Εικόνα 3.5.1 Arduino pin

Επιπλέον, στην πάνω πλευρά του Arduino βρίσκονται 14 θηλυκά pin, αριθμημένα από το 0 έως το 13, που μπορούν να λειτουργήσουν ως ψηφιακές είσοδοι και έξοδοι. Τα pin αυτά λειτουργούν με τάση 5 Volt και το μέγιστο ρεύμα που μπορούν να παρέχουν ή να δεχτούν το κάθε ένα από αυτά είναι 40mA. Ως ψηφιακή έξοδος, ένα από τα pin μπορεί να τεθεί από το πρόγραμμα σε κατάσταση HIGH ή LOW, οπότε το Arduino θα ξέρει αν πρέπει να διοχετεύσει ρεύμα ή όχι στο συγκεκριμένο pin. Με αυτό τον τρόπο μπορεί λόγου χάρη να ανάψει και να σβήσει ένα μικρό LED που έχει συνδεθεί στο συγκεκριμένο pin. Αν πάλι ρυθμιστεί ένα από αυτά τα pin ως ψηφιακή είσοδος μέσα από το πρόγραμμα, μπορεί με την κατάλληλη εντολή να διαβαστεί τη κατάστασή του ανάλογα με το αν η εξωτερική συσκευή που είναι συνδεμένη σε αυτό το pin διοχετεύει ρεύμα ή όχι. Με αυτό τον τρόπο για παράδειγμα ο χρήστης μπορεί να ελέγξει την κατάσταση ενός διακόπτη. Μερικά από τα 14 pin, εκτός από ψηφιακές είσοδοι και έξοδοι έχουν και δεύτερη λειτουργία. Συγκεκριμένα:

- Τα pin 0 και 1 λειτουργούν ως RX και TX της σειριακής όταν το πρόγραμμα ενεργοποιεί τη σειριακή θύρα. Έτσι, όταν για παράδειγμα το πρόγραμμα στέλνει δεδομένα στην σειριακή, αυτά προωθούνται και στην θύρα USB μέσω του ελεγκτή Serial-over-USB αλλά

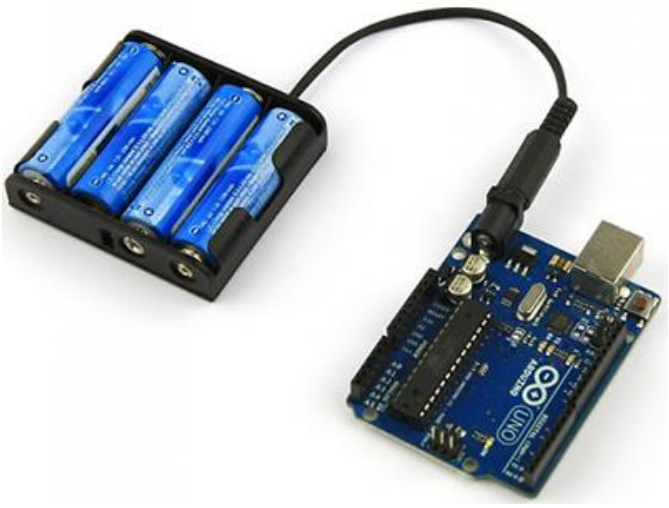
και στο pin 0 για να διαβάσει ενδεχομένως μια άλλη συσκευή. Αυτό φυσικά σημαίνει ότι αν στο πρόγραμμα ενεργοποιηθεί το σειριακό πρωτόκολλο επικοινωνίας, χάνονται 2 ψηφιακές εισόδους/εξόδους.

- Τα pin 2 και 3 λειτουργούν και ως εξωτερικά interrupt 0 και 1 αντίστοιχα. Με άλλα λόγια, μπορούν να ρυθμιστούν μέσα από το πρόγραμμα ώστε να λειτουργούν αποκλειστικά ως ψηφιακές είσοδοι στις οποίες όταν συμβαίνουν συγκεκριμένες αλλαγές, η κανονική ροή του προγράμματος σταματάει άμεσα και εκτελείται μια συγκεκριμένη συνάρτηση. Τα εξωτερικά interrupt είναι ιδιαίτερα χρήσιμα σε εφαρμογές που απαιτούν συγχρονισμό μεγάλης ακρίβειας.

- Τα pin 3, 5, 6, 9, 10 και 11 μπορούν να λειτουργήσουν και ως ψευδοαναλογικές έξοδοι με το σύστημα PWM (Pulse Width Modulation), δηλαδή το ίδιο σύστημα που διαθέτουν οι μητρικές κάρτες των ηλεκτρονικών υπολογιστών για να ελέγχουν τις ταχύτητες των ανεμιστήρων. Έτσι, μπορεί να συνδεθεί για παράδειγμα ένα LED σε κάποιο από αυτά τα pin και ο χρήστης να ελέγξει πλήρως τη φωτεινότητα του με ανάλυση 8bit (256 καταστάσεις από 0-σβηστό έως 255-πλήρως αναμμένο) αντί να δίνεται απλά η δυνατότητα αναμμένο-σβηστό που παρέχουν οι υπόλοιπες ψηφιακές έξοδοι. Είναι σημαντικό να τονιστεί ότι το PWM δεν είναι πραγματικά αναλογικό σύστημα και ότι θέτοντας στην έξοδο την τιμή 127, δεν σημαίνει ότι η έξοδος θα δίνει 2,5 Volt αντί της κανονικής τιμής των 5 Volt, αλλά ότι θα δίνει ένα παλμό που θα εναλλάσσεται με μεγάλη συχνότητα και για ίσους χρόνους μεταξύ των τιμών 0 και 5 Volt.

Στην κάτω πλευρά του Arduino, υπάρχει μια ακόμα σειρά από 6 pin, με τη σήμανση ANALOGIN, αριθμημένα από το 0 έως το 5. Το κάθε ένα από αυτά λειτουργεί ως αναλογική είσοδος χρησιμοποιώντας το ADC (Analog to Digital Converter) που είναι ενσωματωμένο στον μικροελεγκτή. Για παράδειγμα, ο χρήστης μπορεί να τροφοδοτήσει ένα από αυτά με μια τάση η οποία μπορεί να κυμαίνεται με ένα ποτενσιόμετρο από 0 volt ως μια τάση αναφοράς Vref η οποία, αν δεν υποστεί κάποια αλλαγή είναι προ ρυθμισμένη στα 5 Volt. Στην περίπτωση αυτή, μέσα από το πρόγραμμα μπορεί να διαβαστεί η τιμή του pin ως ένας ακέραιος αριθμό ανάλυσης 10-bit, από 0 όταν η τάση στο pin είναι 0 Volt, μέχρι και 1023 όταν η τάση στο pin είναι 5 Volt. Η τάση αναφοράς μπορεί να ρυθμιστεί με μια εντολή στο

1,1 Volt, ή σε όποια τάση επιθυμεί ο χρήστης μεταξύ 2 και 5 Volt τροφοδοτώντας εξωτερικά με αυτή την τάση το pin με την σήμανση AREF που βρίσκεται στην απέναντι πλευρά της πλακέτας. Έτσι, αν τροφοδοτηθεί το pinAREF με 3,3 Volt και στη συνέχεια δοκιμάσει ο χρήστης να διαβάσει κάποιο pin αναλογικής εισόδου στο οποίο εφαρμόζεται τάση 1,65 Volt, το Arduino θα επιστρέψει την τιμή 512. Τέλος, καθένα από τα 6 αυτά pin, με κατάλληλη εντολή μέσα από το πρόγραμμα, μπορεί να μετατρέψει σε ψηφιακό pin εισόδου/εξόδου όπως τα 14 pin που βρίσκονται στην απέναντι πλευρά και τα οποία περιγράφηκαν πριν. Σε αυτή την περίπτωση, τα pin μετονομάζονται από 0-5 σε 14-19 αντίστοιχα.



3.6 Η τροφοδοσία της πλακέτας

Το Arduino μπορεί να τροφοδοτηθεί με ρεύμα είτε από τον υπολογιστή μέσω της σύνδεσης USB, είτε από εξωτερική τροφοδοσία που παρέχεται μέσω μιας υποδοχής φις των 2,1mm με τον θετικό πόλο στο κέντρο και ο οποίος βρίσκεται στην κάτω αριστερή γωνία του Arduino. Για να μην υπάρχουν προβλήματα, η εξωτερική τροφοδοσία πρέπει να είναι από 7 έως 12 Volt και μπορεί να προέρχεται από ένα κοινό μετασχηματιστή του εμπορίου, από μπαταρίες ή από οποιαδήποτε άλλη πηγή συνεχούς ρεύματος.

Εικόνα 3.6.1 Τροφοδοσία του Arduino

Δίπλα από τα pin των αναλογικών εισόδων, υπάρχει μια ακόμα συστοιχία από 6 pin με την σήμανση POWER. Η λειτουργία του καθενός έχει ως εξής:

- Το πρώτο, με την ένδειξη RESET, όταν γειωθεί σε ένα από τα 3 pin με την ένδειξη GND που υπάρχουν στο Arduino, έχει ως αποτέλεσμα την επανεκκίνηση του Arduino.

- Το δεύτερο, με την ένδειξη 3.3V, μπορεί να τροφοδοτήσει τα εξαρτήματα με τάση 3,3Volt. Η τάση αυτή δεν προέρχεται από την εξωτερική τροφοδοσία αλλά παράγεται από τον ελεγκτή Serial-over-USB και έτσι η μέγιστη ένταση που μπορεί να παράσχει είναι 50mA.

- Το τρίτο, με την ένδειξη 5V, μπορεί να τροφοδοτήσει τα εξαρτήματα με τάση 5 Volt. Ανάλογα με τον τρόπο τροφοδοσίας του ίδιου του Arduino, η τάση αυτή προέρχεται είτε άμεσα από την θύρα USB οπού λειτουργεί στα 5 Volt, είτε από εξωτερική τροφοδοσία αφού αυτή περάσει πρώτα από ένα ρυθμιστή τάσης για να την τροποποιήσει στα 5 Volt.

- Το τέταρτο και πέμπτο pin με την ένδειξη GND, είναι η γείωση.

- Το έκτο και τελευταίο pin, με την ένδειξη Vin έχει διπλό ρόλο. Σε συνδυασμό με το pin γείωσης δίπλα του, μπορεί να λειτουργήσει ως μέθοδος εξωτερικής τροφοδοσίας του Arduino, στην περίπτωση που δεν βολεύει να χρησιμοποιηθεί η υποδοχή του φις των 2,1mm. Αν όμως είναι ήδη συνδεμένη η εξωτερική τροφοδοσία μέσω του φις, μπορεί να χρησιμοποιηθεί αυτό το pin για να τροφοδοτήσει εξαρτήματα με την πλήρη τάση της

εξωτερικής τροφοδοσίας (7-12Volt), πριν αυτή περάσει από τον ρυθμιστή τάσης όπως γίνεται με το pin των 5 Volt.

3.7 Ενσωματωμένα κουμπιά και LED πάνω στην πλακέτα

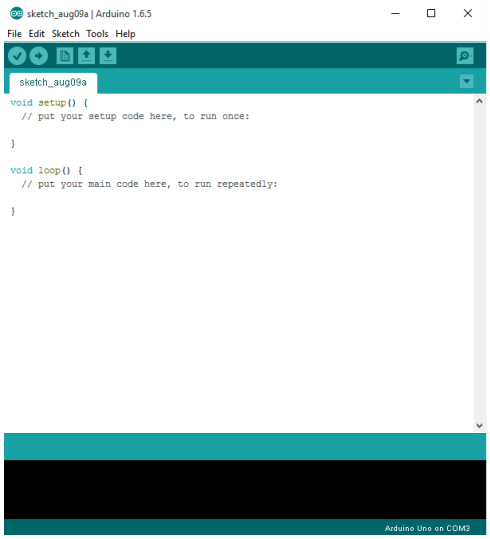
Πάνω στην πλακέτα του Arduino υπάρχει ένας διακόπτης micro-switch καθώς και 4 μικροσκοπικά LED επιφανειακής στήριξης. Πατώντας τον διακόπτη, που έχει την σήμανση RESET, γίνεται επανεκκίνηση στην πλακέτα του Arduino.

Το LED με την σήμανση POWER μας πιστοποιεί ότι το Arduino έχει τεθεί σε λειτουργία.

Τα δύο LED με τις σημάνσεις RX και TX, χρησιμοποιούνται ως ένδειξη λειτουργίας του σειριακού πρωτόκολλου επικοινωνίας, καθώς ανάβουν όταν το Arduino στέλνει ή αντίστοιχα λαμβάνει δεδομένα μέσω της USB θύρας. Τα LED αυτά ελέγχονται από τον ελεγκτή Serial-over-USB και συνεπώς δεν λειτουργούν όταν η σειριακή επικοινωνία γίνεται αποκλειστικά μέσω των ψηφιακών pin 0 και 1.

Τέλος υπάρχει το LED με την σήμανση L. Η βασική δοκιμή λειτουργίας του Arduino είναι να του ανατεθεί να αναβοσβήσει ένα LED. Για να μπορεί να γίνει αυτό από την πρώτη στιγμή, χωρίς να συνδεθεί τίποτα πάνω στο Arduino, οι κατασκευαστές σκέφτηκαν να ενσωματώσουν ένα LED στην πλακέτα, το οποίο το συνέδεσαν στο ψηφιακό pin 13. Έτσι, ακόμα και αν δεν έχει συνδεθεί τίποτα πάνω στο φυσικό pin 13, αναθέτοντας του την τιμή HIGH μέσα από το πρόγραμμα, θα ανάψει αυτό το ενσωματωμένο LED.

3.8 Σύνδεση του Arduino με τον υπολογιστή



Εικόνα 3.8.1 Περιβάλλον IDE

Το ολοκληρωμένο περιβάλλον ανάπτυξης του Arduino είναι το ArduinoIDE, το οποίο είναι μια εφαρμογή γραμμένη σε γλώσσα προγραμματισμού Java, που λειτουργεί σε πολλές πλατφόρμες, και προέρχεται από το IDE για την γλώσσα προγραμματισμού Processing και το σχέδιο Wiring. Έχει σχεδιαστεί για να εισαγάγει τον προγραμματισμό στους καλλιτέχνες και στους νέους που δεν είναι εξοικειωμένοι με την ανάπτυξη λογισμικού. Περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα με χαρακτηριστικά όπως είναι η επισήμανση σύνταξης και ο συνδυασμός αγκυλών και είναι επίσης σε θέση να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με πολύ εύκολο τρόπο. Ένα πρόγραμμα ή κώδικας που γράφεται για τον Arduino ονομάζεται σκίτσο (sketch). Τα προγράμματα του Arduino είναι γραμμένα σε γλώσσα C ή C++. Το ArduinoIDE έρχεται μαζί με μια βιβλιοθήκη λογισμικού που ονομάζεται Wiring από το πρωτότυπο σχέδιο Wiring, γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου/εξόδου πολύ πιο εύκολες. Οι χρήστες πρέπει μόνο να ορίσουν δυο λειτουργίες για να κάνουν ένα πρόγραμμα κυκλικής εκτέλεσης. Οι λειτουργίες αυτές είναι η setup() η οποία είναι μια συνάρτηση που τρέχει μια φορά στην αρχή του προγράμματος και στην οποία γίνεται η αρχικοποίηση των ρυθμίσεων, επιπλέον η loop(), που είναι μια συνάρτηση η οποία εκτελείται συνέχεια μέχρι η πλακέτα να απενεργοποιηθεί.

3.9 Διαφορές και χαρακτηριστικά στις βασικές εκδόσεις του Arduino

Το Arduino Diecimila έχει ουσιαστικά δύο βασικές διαφορές με το Arduino Duemilanove. Η μια διαφορά τους είναι ότι το Arduino Diecimila βασίζεται στον μικροελεγκτή ATmega128, ο οποίος διαθέτει τη μισή μνήμη από τον ATmega328, δηλαδή 1Kb SRAM, 512 bytes EEPROM και 16Kb Flash όπου τα 14 Kb από αυτά είναι ελεύθερα λόγω του bootloader. Η άλλη διαφορά τους είναι ότι το Arduino Diecimila δεν επιλέγει αυτόματα μεταξύ της εξωτερικής τροφοδοσίας και της τροφοδοσίας μέσω της θύρας USB, αλλά διαθέτει ειδικό jumper με το οποίο μπορεί να επιλεχθεί χειροκίνητα η πηγή τροφοδοσίας.

Εικόνα 3.9.1 Arduino Diecimila

Εικόνα 3.9.2 Arduino Duemilanove

Εικόνα 3.9.3 Arduino UNO

Το Arduino Uno χρησιμοποιεί την ίδια τεχνολογία ATmega328 όπως και το τελευταίο μοντέλο Arduino Duemilanove, αλλά ενώ το Arduino Duemilanove χρησιμοποιεί ένα FTDIchipset για το USB, το Arduino Uno χρησιμοποιεί τεχνολογία ATmega8U2 προγραμματιζόμενο ως σειριακός μετατροπέας.

Το Arduino Mega είναι η πιο εξελιγμένη έκδοση με τον μικροελεγκτή ATmega1280 και αρκετά μεγαλύτερο μέγεθος. Οι διαφορές που έχει με τον Arduino Duemilanove είναι:

Εικόνα 2.9.4 Arduino MEGA

- Τετραπλάσια μνήμη (8Kb SRAM, 4Kb EEPROM, 128Kb Flash)
- 40 επιπλέον ψηφιακά pin εισόδου/εξόδου (σύνολο 54)

- 10 επιπλέον pin αναλογικής εισόδου (σύνολο 16)
- Ψευδοαναλογικές εξόδους PWM σε 8 ακόμα ψηφιακά pin (σύνολο 14)
- Εξωτερικά interrupt σε 4 ακόμα ψηφιακά pin (σύνολο 6)
- Επιπλέον 3 σειριακά Interface (σύνολο 4)

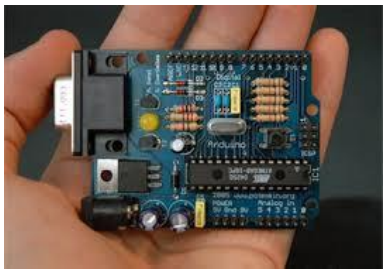
Το Arduino Mega είναι συμβατό με τα περισσότερα shield που έχουν κυκλοφορήσει για το Arduino αλλά όχι με το Ethernet Shield, το οποίο είναι ένα αρκετά σημαντικό μειονέκτημα για όσους θέλουν να φτιάξουν εφαρμογές με πρόσβαση στο Internet η σε κάποιο άλλο δίκτυο.

Το Arduino Mega 2560, είναι μια νέα έκδοση του Arduino Mega η οποία χρησιμοποιεί τεχνολογία surface-mounted ATmega2560 φέροντας την ολική μνήμη στα 256Kb, από τα οποία τα 8Kb χρησιμοποιούνται από το bootloader. Επίσης ενσωματώνει την νέα τεχνολογία ATmega8U2 USBchipset.

Εικόνα 3.9.5 Arduino MEGA 2560

3.10 Διάφορες επιπλέον εκδόσεις Arduino

Το Serial Arduino, είναι μια βασική πλακέτα που χρησιμοποιεί την θύρα RS232 ως μέσο σύνδεσης με τον υπολογιστή για τον προγραμματισμό και την επικοινωνία. Αυτή η πλακέτα είναι πολύ εύκολο να την συναρμολογήσει οποιοσδήποτε, η ακόμα να χρησιμοποιηθεί ως άσκηση μάθησης.

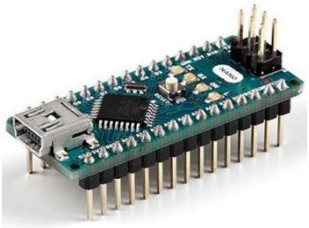


Εικόνα 3.10.1 Serial Arduino

Το Arduino Mini, είναι μια έκδοση μινιατούρας που αρχικά βασιζόταν στο μικροελεγκτή ATmega168 αλλά τώρα παρέχεται και με τον ATmega328. Η πλακέτα αυτή προορίζεται για καταστάσεις όπου ο διαθέσιμος χώρος είναι περιορισμένος. Έχει 14 ψηφιακούς ακροδέκτες εισόδου/εξόδου εκ των οποίων 6

μπορούν να χρησιμοποιηθούν ως έξοδοι PWM, έχει επίσης 8 αναλογικές εισόδους και ένα κρυσταλλικό ταλαντωτή στα 16MHz. Μπορεί να συνδεθεί με τον υπολογιστή και προγραμματιστεί μέσω USB θύρας. Επίσης να σημειωθεί ότι η τάση τροφοδοσίας για το Arduino Mini δεν πρέπει να ξεπεράσει τα 9 Volt.

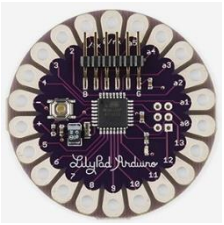
Εικόνα 3.10.2 Arduino Mini



Εικόνα 3.10.3 Arduino Nano

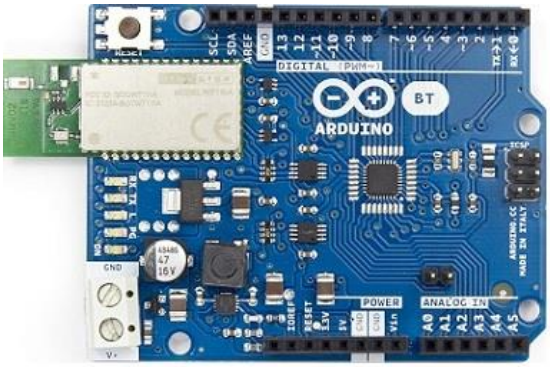
Το Arduino Nano, είναι μια ακόμα πιο μικρή ηλεκτρονική πλακέτα με βάση τον μικροελεγκτή ATmega328. Έχει περίπου την ίδια λειτουργία με το Arduino Duemilanove, αλλά είναι σε μικρότερη διάσταση. Στερείται μόνο μιας υποδοχή ρεύματος DC και μπορεί να συνδεθεί με τον υπολογιστή και προγραμματιστεί μέσω θύρας USBMini-B.

Το Lily Pad Arduino, βασίζεται στον μικροελεγκτή ATmega168V το οποίο είναι μια έκδοση χαμηλής ισχύος του ATmega168. Έχει 14 ψηφιακές εισόδους/εξόδους εκ των οποίων 6 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM, ένω έχει επίσης 6 αναλογικές εισόδους και ένα κρυσταλλικό ταλαντωτή στα 8MHz.



Εικόνα 3.10.4 LilyPad

Το Arduino Bluetooth, είναι μια πλακέτα που αρχικά βασίστηκε στο μικροελεγκτή ATmega168, αλλά τώρα διατίθεται με τον μικροελεγκτή ATmega328 και με την μονάδα bluegigaWT11 Bluetooth. Υποστηρίζει ασύρματη σειριακή επικοινωνία μέσω Bluetooth, αλλά να επισημάνουμε ότι δεν είναι συμβατό με Bluetooth ακουστικά η άλλες συσκευές ήχου. Έχει 14 ψηφιακές θύρες εισόδου/εξόδου εκ των οποίων 6 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM και 1 μπορεί να χρησιμοποιηθεί για την επανεκκίνηση της μονάδας WT11 Bluetooth. Οι αναλογικές είσοδοι του είναι 6 και ο κρυσταλλικός του ταλαντωτής είναι στα 16MHz.

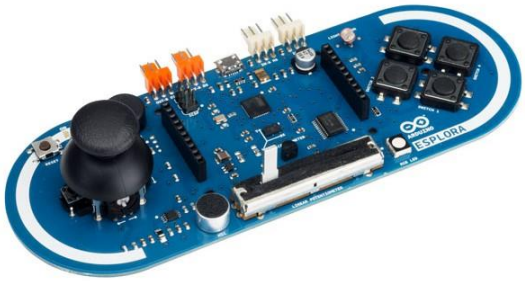


Εικόνα 3.10.5 Arduino Bluetooth

Το Arduino Leonardo, είναι μια πλακέτα που βασίζεται στο μικροελεγκτή ATmega32u4. Έχει 20 ψηφιακές θύρες εισόδου/εξόδου εκ των οποίων 7 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM και 12 ως αναλογικές είσοδοι. Ο κρυσταλλικός του ταλαντωτής είναι στα 16MHz. Διαθέτει μια υποδοχή ρεύματος και μια υποδοχή ICSP και μπορεί να συνδεθεί με τον υπολογιστή και προγραμματιστεί μέσω USB Mini-B θύρας. Το Arduino Leonardo διαφέρει από όλες τις προηγούμενες πλακέτες από το γεγονός ότι ο μικροελεγκτής ATmega32u4 έχει ενσωματωμένη επικοινωνία USB, εξαλείφοντας έτσι την ανάγκη για ένα δευτερεύον επεξεργαστή. Αυτό επιτρέπει στο Arduino Leonardo να εμφανίζεται στον συνδεμένο υπολογιστή όχι μόνο σαν μια εικονική σειριακή θύρα αλλά και ως πληκτρολόγιο η ποντίκι.

Εικόνα 3.10.6 Arduino Leonardo

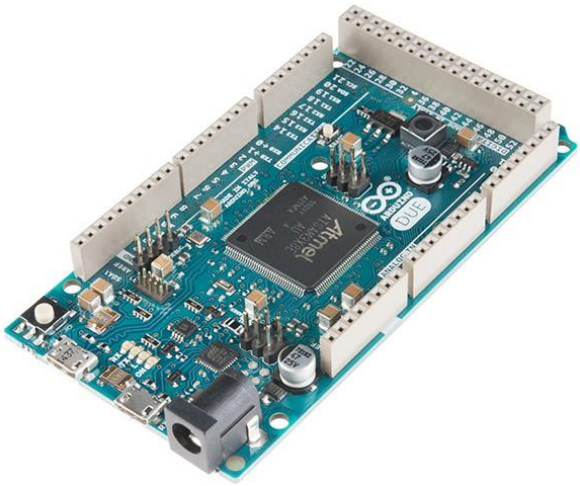
Το Arduino Esplora, είναι μια πλακέτα που προέρχεται από το Arduino Leonardo. Arduino Esplora διαφέρει πολύ από όλες τις προηγούμενες εκδόσεις του Arduino, τόσο στη γεωμετρία και στο σχήμα του αλλά και στα περιφερειακά εξαρτήματα, τα οποία βρίσκονται ενσωματωμένα πάνω στην πλακέτα και είναι έτοιμα προς χρήση. Είναι σχεδιασμένο για ανθρώπους που θέλουν να χρησιμοποιήσουν το Arduino χωρίς να χρειάζεται να μάθουν πρώτα το ηλεκτρονικό μέρος. Το Arduino Esplora έχει ενσωματωμένο σύστημα ήχου και διάφορες φωτεινές ενδείξεις. Επίσης διαθέτει διάφορους αισθητήρες εισόδου, συμπεριλαμβανομένου ενός joystick, ενός γραμμικού ποτενσιομέτρου, ενός αισθητήρα θερμοκρασίας, ενός επιταχυνσιομέτρου, ενός μικροφώνου, ενός αισθητήρα φωτός και τεσσάρων διακοπτών. Επιπλέον, έχει την δυνατότητα να επεκτείνει τις λειτουργίες του με δυο υποδοχές Tinkerkit εισόδου και εξόδου, καθώς και με υποδοχή για μία οθόνη TFTLCD έγχρωμη. Όπως το Arduino Leonardo έτσι και το Arduino Esplora χρησιμοποιεί ένα μικροελεγκτή ATmega32U4 με 16MHz κρυσταλλικό ταλαντωτή και ένα USBMini για την διασύνδεση του με τον ηλεκτρονικό υπολογιστή και αναγνωρίζεται ως συσκευή USB, όπως ένα ποντίκι η ένα πληκτρολόγιο.



Εικόνα 3.10.7 Arduino Esplora

Εικόνα 3.10.8 Arduino Esplora με οθόνη TFT LCD

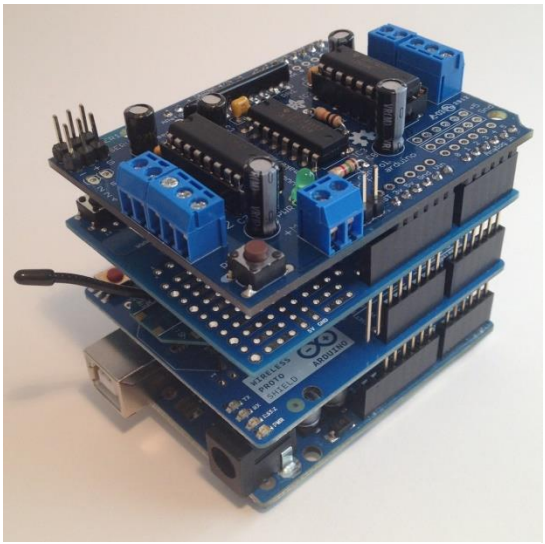
Το Arduino Due, είναι μια πλακέτα με βάση τον επεξεργαστή AtmelSAM3X8EARMCortex-M3. Είναι η πρώτη πλακέτα Arduino που βασίζεται σε ένα ARM μικροελεγκτή με 32-bit πυρήνα. Έχει 54 ψηφιακές εισόδους/εξόδους εκ των οποίων 12 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM, διαθέτει 12 αναλογικές εξόδους, 4 σειριακές θύρες UARTs, ένα ρολόι 84MHz, ένα USBOTG ως μέσο σύνδεσης, 2 TWI, 2 DAC



Εικόνα 3.10.9 Arduino DUE

(μετατροπέας ψηφιακού σήματος σε αναλογικό), μια υποδοχή ρεύματος, μια υποδοχή SPI, μια υποδοχή JTAG, ένα κουμπί επανεκκίνησης και ένα πλήκτρο διαγραφής. Σε αντίθεση με τις άλλες πλακέτες Arduino, το Arduino Due μπορεί να δεχτεί μέγιστη τάση 3.3 Volt, ενώ μια μεγαλύτερη τάση μπορεί να προκαλέσει ζημία στο ηλεκτρικό κύκλωμα της πλακέτας.

3.11 Arduino shields



Εικόνα 3.11.1 Arduino Shields

Τα Arduino Shields στην ουσία είναι τυπωμένες ηλεκτρονικές πλακέτες που χρησιμοποιούνται ως επέκταση κυκλωμάτων. Οι πλακέτες αυτές συνδέονται στα κανονικά παρεχόμενα Arduino pin-headers και η τοποθέτησή τους είναι πολύ εύκολη. Αυτές οι πλακέτες μπορούν να παρέχουν για παράδειγμα έλεγχο σε ηλεκτρικούς κινητήρες, GPS, Ethernet, Wi-Fi, LCD οθόνες εικόνας η πλακέτες πρωτοτυποποίησης. επίσης, αξίζει να αναφέρουμε ότι πολλές τέτοιες πλακέτες μπορούν να κατασκευαστούν και από τους χρήστες. Παρακάτω θα δούμε πιο αναλυτικά κάποια από τα βασικά shields του Arduino.

Εικόνα 3.11.2 Arduino Motor Shield

Μια πολύ βασική πλακέτα επέκτασης του Arduino είναι το Arduino Motor Shield. Η πλακέτα αυτή βασίζεται στο ολοκληρωμένο L298 το οποίο είναι μια γέφυρα διπλής οδήγησης που έχει σχεδιαστεί για να οδηγεί επαγωγικά φορτία όπως ρελέ, ηλεκτρικούς κινητήρες συνεχούς ρεύματος καθώς και βηματικούς. Επιτρέπει την οδήγηση δυο ηλεκτρικών κινητήρων, ελέγχοντας την ταχύτητα και την κατεύθυνση του καθενός χωριστά και ένα βηματικό κινητήρα. Όσον αφορά την τροφοδοσία του, υπάρχει μόνο μία υποδοχή και άρα πρέπει να τροφοδοτείται μόνο από εκεί. Η μέγιστη τάση που μπορεί να δεχτεί είναι 12 Volt και το μέγιστο ρεύμα που μπορεί να διαρρέει την πλακέτα ανά κανάλι είναι 2A δηλαδή 4A συνολικά.

Μια επιπλέον γνωστή πλακέτα επέκτασης είναι το Arduino Wireless SD Shield. Η πλακέτα αυτή επιτρέπει σε μια βασική πλακέτα Arduino να επικοινωνεί ασύρματα με μια ασύρματη μονάδα. Βασίζεται στις ενότητες του Xbee από την Digi, αλλά μπορεί να χρησιμοποιεί οποιαδήποτε μονάδα με το ίδιο αποτύπωμα. Η μονάδα αυτή μπορεί να επικοινωνεί μέχρι και 30 μέτρα σε εσωτερικούς χώρους η 90 μέτρα σε εξωτερικούς χώρους. Μπορεί να χρησιμοποιηθεί ως υποκατάστατο της σειριακής επικοινωνίας.

4. Η συσκευή GPS που χρησιμοποιήθηκε

4.1 Η συσκευή

Η συσκευή που χρησιμοποιήθηκε στα πλαίσια της πτυχιακής εργασίας είναι η πλατφόρμα EVK-7P (Evaluation Kit Precise Point Positioning) της εταιρίας Ublox. Η συγκεκριμένη συσκευή GPS έχει ενσωματωμένη θύρα USB που παρέχει τόσο τροφοδοσία ρεύματος όσο και μεταφορά δεδομένων σε αρκετά υψηλές ταχύτητες και με αυτό τον τρόπο εξαλείφεται η ανάγκη για εξωτερική παροχή ρεύματος. Οι συσκευές u-blox είναι φιλικές προς τον χρήστη και λόγω της τροφοδοσίας τους καθίστανται ιδανικές για χρήση σε εργαστήρια, οχήματα και υπαίθριους χώρους. Επιπλέον, μπορούν να χρησιμοποιηθούν με PDA ή φορητό υπολογιστή, καθιστώντας τες, τον ιδανικό σύντροφο σε όλα τα στάδια σχεδιασμού σε έργα.

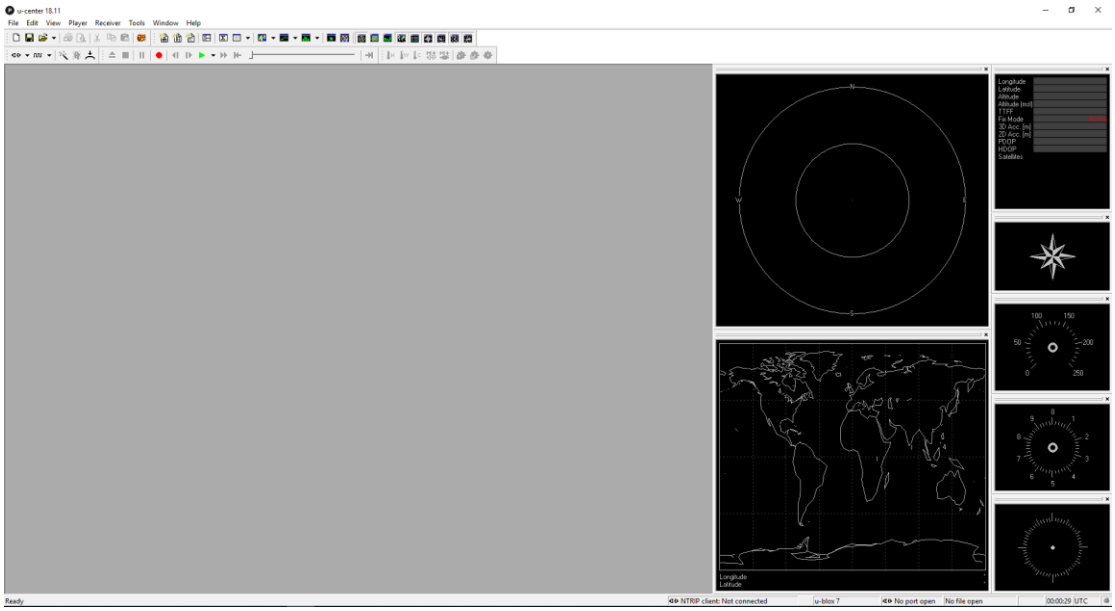
Εικόνα 4.1.1 Ublox EVK-7P

4.2 Σύνδεση με τον υπολογιστή

Αρχικά για να χρησιμοποιήσουμε την πλατφόρμα EVK-7P, έπρεπε να κατεβάσουμε από την σελίδα www.u-blox.com, το αρχείο u-center_v18.11.exe και να το εγκαταστήσουμε στον υπολογιστή μας. Στην συνέχεια συνδέσαμε την πλατφόρμα EVK-7P στον υπολογιστή μας, μέσω της σειριακής θύρας USB.

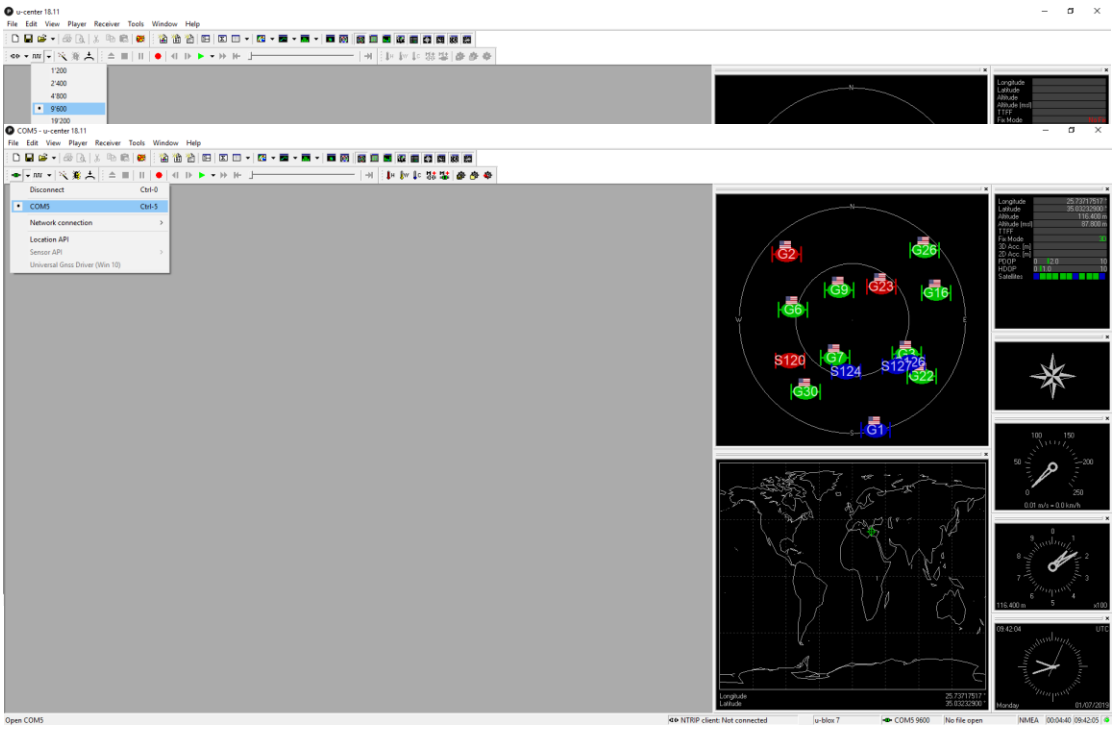
Εικόνα 4.2.1 Πίσω όψη του Ublox EVK-7P

Ανοίγοντας το εγκατεστημένο πλέον πρόγραμμα μας εμφανίστηκε το παρακάτω περιβάλλον.

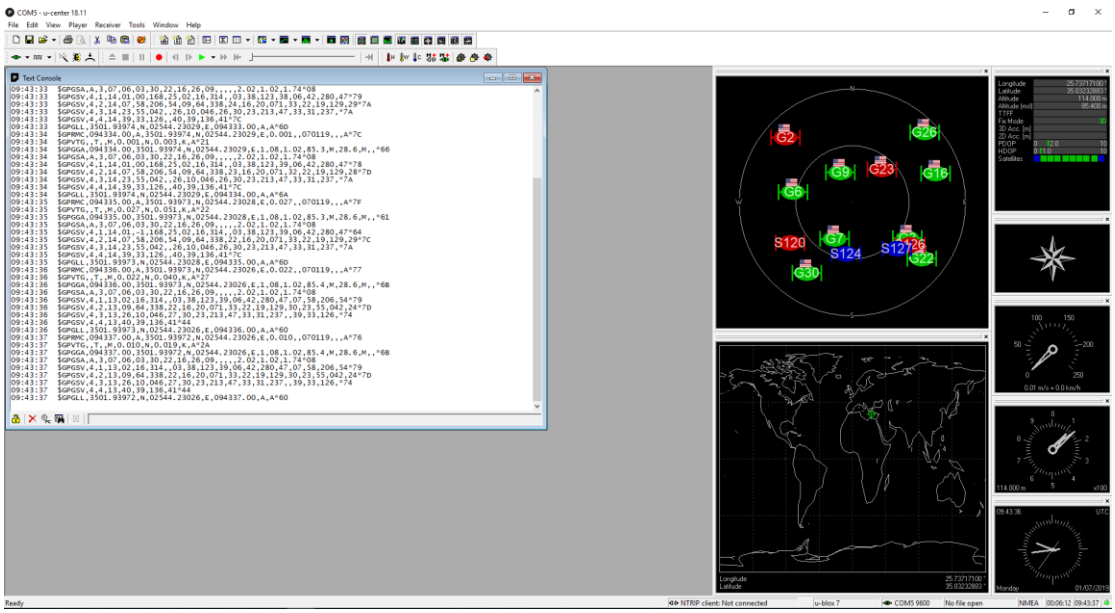


Για να αρχίσουμε να παίρνουμε δεδομένα από την πλατφόρμα EVK-7P έπρεπε να πραγματοποιήσουμε κάποιες ρυθμίσεις. Δηλαδή έπρεπε να επιλέξουμε, σε ποια θύρα του υπολογιστή μας ήταν συνδεμένη.

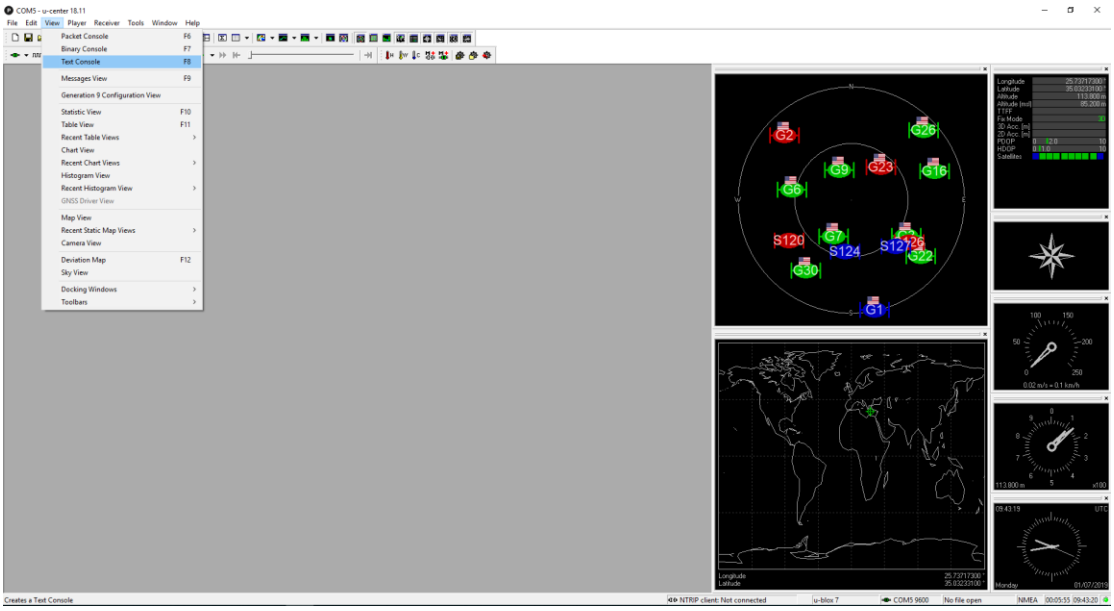
Καθώς επίσης ποια θα ήταν η ταχύτητα επικοινωνίας.

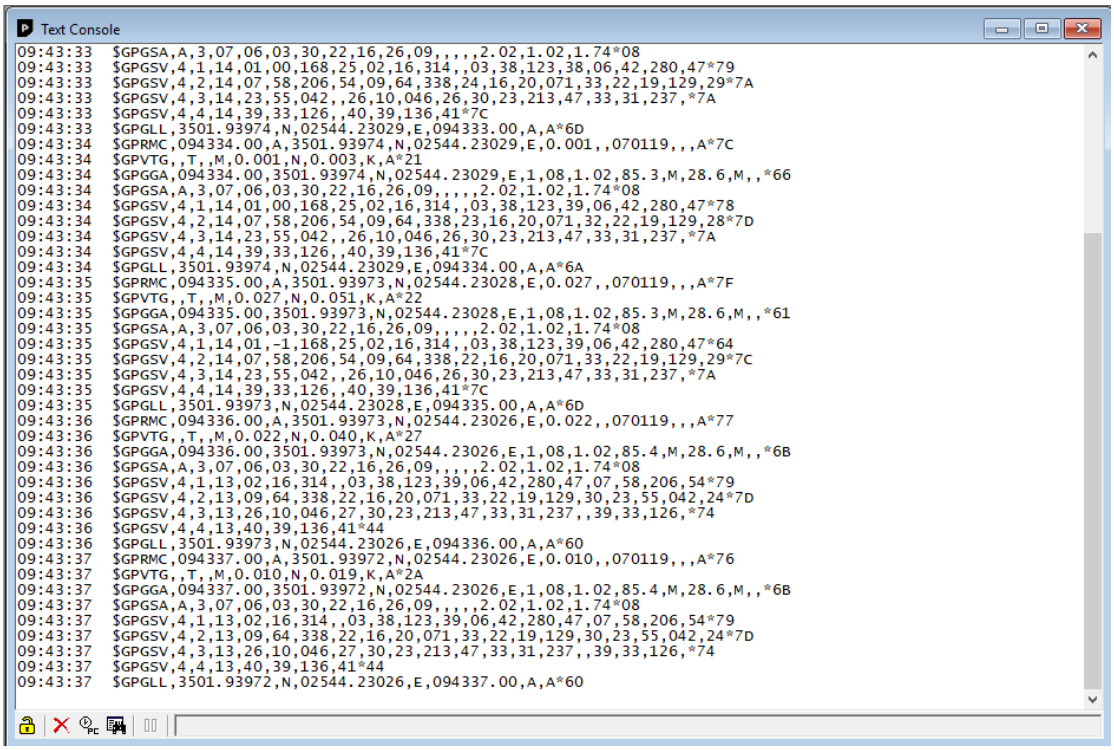


Μόλις ήταν έτοιμες οι ρυθμίσεις, ξεκινούσε η επικοινωνία μεταξύ της πλατφόρμας EVK-7P και του υπολογιστή μας. Αν επιλέξουμε view -> text console, εμφανίζεται ένα παράθυρο στην οθόνη μας, με κάποια πακέτα δεδομένων που μας στέλνει η πλατφόρμα EVK-7P.

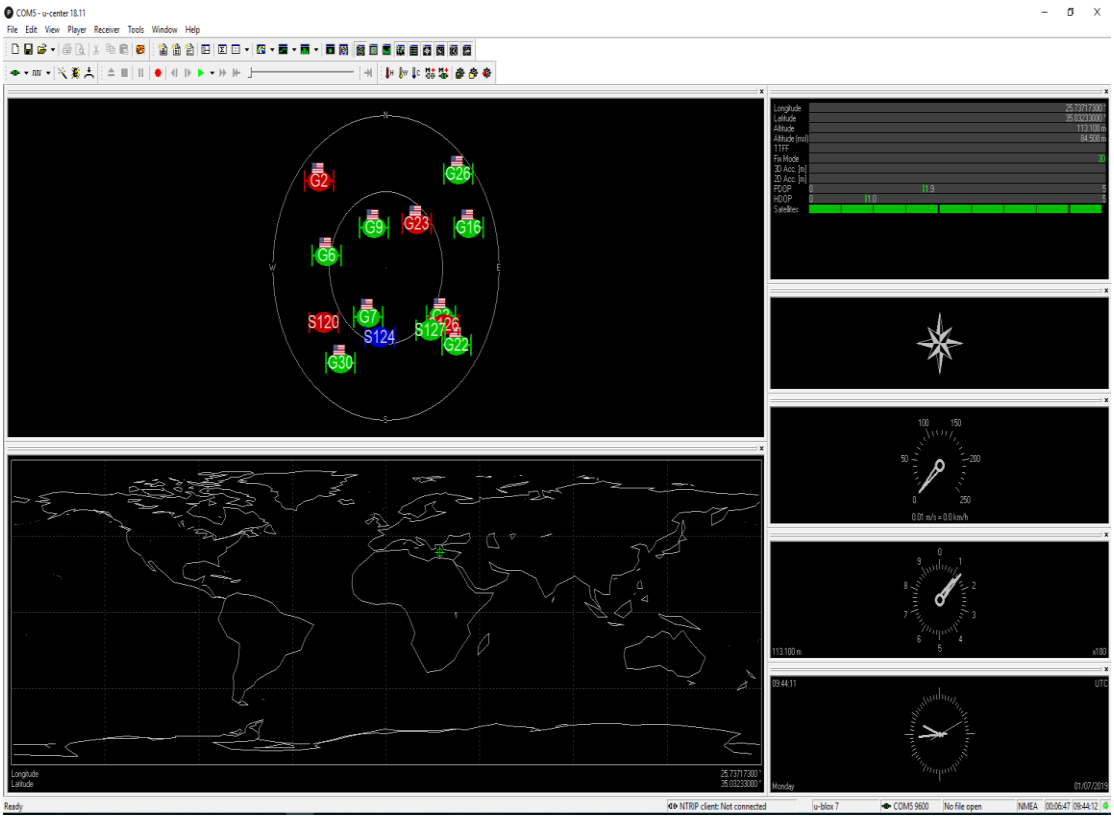


Τα δεδομένα αυτά είναι στην ουσία ο χώρος και ο χρόνος στον οποίο βρισκόμαστε.


COM5 - u-center 18.11
File Edit View Player Receiver Tools Window Help
Packet Console F6
Binary Console F7
Text Console F8
Messages View F9
Generation 9 Configuration View
Statistic View F10
Table View F11
Recent Table Views
Chart View
Recent Chart Views
Histogram View
Recent Histogram View
GNSS Driver View
Map View
Recent Static Map Views
Camera View
Deviation Map F12
Sky View
Docking Windows
Toolbars
G2
G26
G9
G23
G16
G6
S120
G7
S124
S122
G30
G1
Creates a Text Console
NTRIP client: Not connected
u-blox 7
COM5 9600
No file open
NMEA


Text Console
09:43:33 $GPGSA,A,3,07,06,03,30,22,16,26,09,,,,,2.02,1.02,1.74*08
09:43:33 $GPGSV,4,1,14,01,00,168,25,02,16,314,,03,38,123,38,06,42,280,47*79
09:43:33 $GPGSV,4,2,14,07,58,206,54,09,64,338,24,16,20,071,33,22,19,129,29*7A
09:43:33 $GPGSV,4,3,14,23,55,042,,26,10,046,26,30,23,213,47,33,31,237,*7A
09:43:33 $GPGSV,4,4,14,39,33,126,,40,39,136,41*7C
09:43:33 $GPGLL,3501.93974,N,02544.23029,E,094333.00,A,A*6D
09:43:34 $GPRMC,094334.00,A,3501.93974,N,02544.23029,E,0.001,,070119,,,A*7C
09:43:34 $GPVTG,,T,,M,0.001,N,0.003,K,A*21
09:43:34 $GPGGA,094334.00,3501.93974,N,02544.23029,E,1,08,1.02,85.3,M,28.6,M,,*66
09:43:34 $GPGSA,A,3,07,06,03,30,22,16,26,09,,,,,2.02,1.02,1.74*08
09:43:34 $GPGSV,4,1,14,01,00,168,25,02,16,314,,03,38,123,39,06,42,280,47*78
09:43:34 $GPGSV,4,2,14,07,58,206,54,09,64,338,23,16,20,071,32,22,19,129,28*7D
09:43:34 $GPGSV,4,3,14,23,55,042,,26,10,046,26,30,23,213,47,33,31,237,*7A
09:43:34 $GPGSV,4,4,14,39,33,126,,40,39,136,41*7C
09:43:34 $GPGLL,3501.93974,N,02544.23029,E,094334.00,A,A*6A
09:43:35 $GPRMC,094335.00,A,3501.93973,N,02544.23028,E,0.027,,070119,,,A*7F
09:43:35 $GPVTG,,T,,M,0.027,N,0.051,K,A*22
09:43:35 $GPGGA,094335.00,3501.93973,N,02544.23028,E,1,08,1.02,85.3,M,28.6,M,,*61
09:43:35 $GPGSA,A,3,07,06,03,30,22,16,26,09,,,,,2.02,1.02,1.74*08
09:43:35 $GPGSV,4,1,14,01,-1,168,25,02,16,314,,03,38,123,39,06,42,280,47*64
09:43:35 $GPGSV,4,2,14,07,58,206,54,09,64,338,22,16,20,071,33,22,19,129,29*7C
09:43:35 $GPGSV,4,3,14,23,55,042,,26,10,046,26,30,23,213,47,33,31,237,*7A
09:43:35 $GPGSV,4,4,14,39,33,126,,40,39,136,41*7C
09:43:35 $GPGLL,3501.93973,N,02544.23028,E,094335.00,A,A*6D
09:43:36 $GPRMC,094336.00,A,3501.93973,N,02544.23026,E,0.022,,070119,,,A*77
09:43:36 $GPVTG,,T,,M,0.022,N,0.040,K,A*27
09:43:36 $GPGGA,094336.00,3501.93973,N,02544.23026,E,1,08,1.02,85.4,M,28.6,M,,*6B
09:43:36 $GPGSA,A,3,07,06,03,30,22,16,26,09,,,,,2.02,1.02,1.74*08
09:43:36 $GPGSV,4,1,13,02,16,314,,03,38,123,39,06,42,280,47,07,58,206,54*79
09:43:36 $GPGSV,4,2,13,09,64,338,22,16,20,071,33,22,19,129,30,23,55,042,24*7D
09:43:36 $GPGSV,4,3,13,26,10,046,27,30,23,213,47,33,31,237,,39,33,126,*74
09:43:36 $GPGSV,4,4,13,40,39,136,41*44
09:43:36 $GPGLL,3501.93973,N,02544.23026,E,094336.00,A,A*60
09:43:37 $GPRMC,094337.00,A,3501.93972,N,02544.23026,E,0.010,,070119,,,A*76
09:43:37 $GPVTG,,T,,M,0.010,N,0.019,K,A*2A
09:43:37 $GPGGA,094337.00,3501.93972,N,02544.23026,E,1,08,1.02,85.4,M,28.6,M,,*6B
09:43:37 $GPGSA,A,3,07,06,03,30,22,16,26,09,,,,,2.02,1.02,1.74*08
09:43:37 $GPGSV,4,1,13,02,16,314,,03,38,123,39,06,42,280,47,07,58,206,54*79
09:43:37 $GPGSV,4,2,13,09,64,338,22,16,20,071,33,22,19,129,30,23,55,042,24*7D
09:43:37 $GPGSV,4,3,13,26,10,046,27,30,23,213,47,33,31,237,,39,33,126,*74
09:43:37 $GPGSV,4,4,13,40,39,136,41*44
09:43:37 $GPGLL,3501.93972,N,02544.23026,E,094337.00,A,A*60

Αν κοιτάξουμε επίσης δεξιά στην οθόνη μας, τότε θα δούμε ότι υπάρχουν κάποια παράθυρα με διάφορα δεδομένα, τα οποία απεικονίζονται πιο αναλυτικά. Μεγαλώνοντας τα εικονίδια, βλέπουμε πάνω αριστερά τους δορυφόρους, από τους οποίους λαμβάνουμε το σήμα αυτή την χρονική στιγμή. Κάτω αριστερά παρατηρούμε το σημείο που βρισκόμαστε πάνω στη γη.



Στην δεξιά μεριά της οθόνης στο πρώτο εικονίδιο φαίνεται το γεωγραφικό πλάτος και μήκος της τοποθεσίας μας, σε δεκαδικές μοίρες καθώς και το υψόμετρο που βρισκόμαστε πάνω από την θάλασσα σε μέτρα. Αν ανοίξουμε το πρόγραμμα του google Earth και πάμε με τον κέρσορα μας στο σημείο που έχουμε την συσκευή εντοπισμού, θα παρατηρήσουμε ότι τα δεδομένα που διαβάζουμε από την συσκευή αντιστοιχούν σε αυτά που μας δείχνει και το google Earth. Άρα, με αυτό τον τρόπο επιβεβαιώσαμε την ορθή λειτουργία της συσκευής μας.

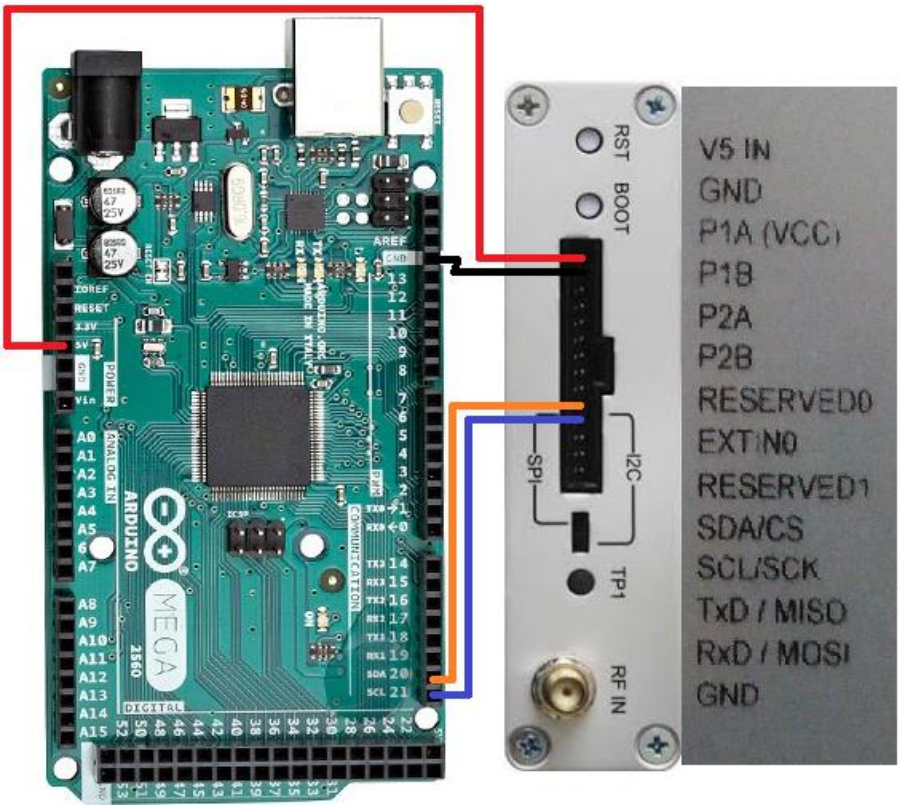
4.3 Σύνδεση με τον arduino

Arduino MEGA 2560	EVK-7P
GND	GND
5V	V5 IN
SDA 20	SDA/CS
SCL 21	SCL/SCK

Αφού ελέγξαμε ότι η συσκευή μας λειτουργεί σωστά στον υπολογιστή μας, έπρεπε να βρούμε ένα μικρότερο υπολογιστή, λόγω έλλειψης χώρου πάνω στο όχημα, έτσι ώστε να λαμβάνουμε τα δεδομένα από την πλατφόρμα EVK-7P και ταυτόχρονα, να μπορούμε να ελέγχουμε και το όχημα μας. Αυτό το ρόλο τον έπαιξε ο μικροελεγκτής Arduino Mega 2560.

Στην ιστοσελίδα www.arduino.cc μπορούμε να βρούμε αρκετή βοήθεια για οποιοδήποτε σχέδιο ή πρόγραμμα αποφασίσουμε να υλοποιήσουμε. Επίσης, υπάρχουν αρκετές πληροφορίες για την γλώσσα και τις εντολές, που χρειάζεται να ξέρουμε, για να μπορέσουμε να φέρουμε σε πέρας κάθε μας εργασία. Έτσι και εμείς με βάση την ιστοσελίδα του arduino ξεκινήσαμε την εργασία μας.

Μετά από αρκετό διάβασμα και ψάξιμο, βρήκαμε αρκετές πληροφορίες και κώδικες για να διαχειριστούμε την πλατφόρμα EVK-7P. Παρακάτω θα δείξουμε το πρώτο μας πρόγραμμα, το οποίο θα μας επαληθεύσει την λειτουργία του EVK-7P από τον arduino. Ο συγκεκριμένος κώδικας επικοινωνεί με το EVK-7P με το πρωτόκολλο επικοινωνίας I2C. Για να αρχίσουμε να λαμβάνουμε τα δεδομένα από το EVK-7P αρκεί να το συνδέσουμε στον Arduino Mega 2560. Η συνδεσμολογία είναι η εξής:



Εικόνα 4.3.1 Συνδεσμολογία Arduino με το EVK-7P

Αυτό που πρέπει να προσέξουμε σε αυτό το σημείο είναι να έχουμε γυρίσει τον διακόπτη στην I2C επικοινωνία.

Εφόσον συνδέσαμε το EVK-7P με τον arduino mega 2560, πρέπει να συνδέσουμε και τον arduino με τον υπολογιστή μας. Στη συνέχεια, ανοίγουμε το πρόγραμμα διαχείρισης του arduino, κάνουμε τις απαραίτητες ρυθμίσεις και έπειτα εισάγουμε τον παρακάτω κώδικα στον arduino mega 2560.

```
// How to get data from u-blox Evaluation kit EVK-7 through I2C interface
// example for Arduino
#include <recoonI2C.h>
#define BUFFER_SIZE  190
uint8_t gpsAddress = 0;
uint8_t slaves = 0;
uint8_t buf[BUFFER_SIZE];
```

```
void setup() {
Serial.begin(115200);
Serial.println("Discover TWI slaves...");
twi_init(400000); // or 100000 Hz (standard mode)
twiDiscoverSlaves();
  slaves = getSlavesCount();
Serial.print("Total ");
Serial.print(slaves, DEC);
Serial.println(" i2c slave devices found");
  if (slaves) {
    for (uint8_t i = 0; i< slaves; i++) {
Serial.print("\t0x");
gpsAddress = getSlaveAddress(i);
      if (gpsAddress< 0x10) Serial.print("0");
Serial.print(gpsAddress, HEX);
Serial.println(" found");
    }
  }
}
void loop() {
  if (!slaves) return; // No TWI slave devices found. Return
delay(200);
  uint16_t bytes = twiReadBytes(gpsAddress, 0xFD, (uint8_t *) buf, 2);
  if (!bytes) return; // got some TWI error. Return
  uint16_t totalBytes = ((uint16_t) buf[0] << 8) | buf[1];
  if (!totalBytes) return; // GPS not ready to send data. Return
Serial.print("GPS is ready to transfer ");
Serial.print(totalBytes, DEC);
Serial.println(" bytes");
  while (totalBytes) {
    uint16_t bytes2Read;
```

```
    if (totalBytes> BUFFER_SIZE) {
      bytes2Read = BUFFER_SIZE;
    } else {
      bytes2Read = totalBytes;
    }
    bytes = twiReadBytes(gpsAddress, 0xFF, (uint8_t *) buf, bytes2Read);
    for (uint8_t i = 0; i< bytes; i++)
    {

Serial.print(char(buf[i]));
}
totalBytes -= bytes2Read;
 }
Serial.println();
}
```

Όταν ο κώδικας εισαχθεί με επιτυχία, τότε ανοίγουμε τη σειριακή οθόνη από το πρόγραμμα, για να δούμε αν λειτουργεί το EVK-7P. Το μήνυμα που αντικρίζουμε είναι το παρακάτω.

```
Discover TWI slaves...
Total 1 i2c slave devices found
        0x42 found
GPS is ready to transfer 582 bytes
$GPRMGC,081259.00,A,3501.94109,N,02544.23128,E,0.084,,120119,,,A*7E
$GPVTG,,T,,M,0.084,N,0.155,K,A*2E
$GPGGA,081259.00,3501.94109,N,02544.23128,E,1,05,2.17,83.4,M,28.6,M,,*66
$GPGSA,A,3,23,22,01,07,16,,,,,,,3.72,2.17,3.03*00
$GPGSV,5,1,17,01,28,171,42,03,61,081,23,06,28,314,,07,25,196,37*7D
$GPGSV,5,2,17,09,52,277,32,11,06,176,28,16,09,102,34,17,11,252,15*74
$GPGSV,5,3,17,18,03,159,36,19,13,275,,22,43,105,36,23,70,341,34*70
```

$GPGSV,5,4,17,26,12,075,29,31,08,038,,33,31,237,,39,33,126,33*7A

$GPGSV,5,5,17,40,39,135,*46

$GPGLL,3501.94109,N,02544.23128,E,081259.00,A,A*61

Αρχικά, το arduino με αυτόν τον κώδικα ψάχνει για τυχόν συνδεμένες συσκευές στην I2C επικοινωνία και έπειτα μας ενημερώνει για το πλήθος των συσκευών που βρέθηκαν συνδεμένες. Στην περίπτωση μας μόνο το EVK-7P είχαμε συνδεδεμένο, άρα μας αναφέρει ότι μια συσκευή βρέθηκε και στην συνέχεια μας δείχνει και την διεύθυνση επικοινωνίας του EVK-7P. Στην συνέχεια στέλνει τις απαραίτητες ‘’εντολές-ερωτήσεις’’ στην διεύθυνση αυτή. Αμέσως μετά μας ενημερώνει, ότι η συσκευή μας είναι έτοιμη για να μας μεταφέρει ένα αριθμό δεδομένων. Πιο συγκεκριμένα τα δεδομένα αυτά μας δίνουν τις παρακάτω πληροφορίες.

$ GPGGA - Καθορισμός δεδομένων καθολικού συστήματος εντοπισμού θέσης

$ GPGLL - Γεωγραφική θέση, γεωγραφικό πλάτος / γεωγραφικό μήκος

$ GPGSA - GPS DOP και ενεργοί δορυφόροι

$ GPGSV – θεατοί δορυφόροι GPS

$ GPVTG - Η διαδρομή που πραγματοποιείται και η ταχύτητα εδάφους

Αφού συγκρίναμε πάλι τα δεδομένα που πήραμε από το GPS, παρατηρούμε ότι οι συντεταγμένες μας δείχνουν την τοποθεσία που βρισκόμαστε. Άρα έτσι είμαστε σίγουροι ότι και με τον arduino δουλεύει το EVK-7P.

Εμείς όμως τα δεδομένα που χρειαζόμαστε στην συγκεκριμένη φάση είναι μόνο η γεωγραφική θέση, δηλαδή το γεωγραφικό πλάτος και μήκος του σημείου που βρισκόμαστε. Γι’ αυτό το λόγο χρειαζόμαστε ένα διαφορετικό κώδικα, που να είναι λίγο πιο απλός και ταυτόχρονα να μπορεί να εξυπηρετήσει τις ανάγκες μας. Μετά από αρκετό ψάξιμο και διάβασμα, βρήκαμε ένα πολύ απλό κώδικα και έπειτα από κάποιες δικές μας μετατροπές εξυπηρετούσε αυτό που ακριβώς θέλαμε. Ο κώδικας είναι ο παρακάτω:

```
#include <SoftwareSerial.h>
#include "TinyGPS++.h"
```

```
int fixed = 0;
int led = 13;
TinyGPSPlus gps;
SoftwareSerial SoftSerial(2, 3); // RX, TX
void setup()
{
pinMode(led, OUTPUT);
SoftSerial.begin(9600);  //
Serial.begin(115200);     //
}
void loop()
{
  while (SoftSerial.available() > 0)
gps.encode(SoftSerial.read());
  if (gps.location.isUpdated())
  {
    fixed = 1;
Serial.print("LAT="); Serial.print(gps.location.lat(), 6);
Serial.print("  LNG="); Serial.println(gps.location.lng(), 6);
  }
if(fixed ==0)
  {
digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
delay(300);              // wait for a second
digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
delay(300);
  }
  }
```

5. Το όχημα

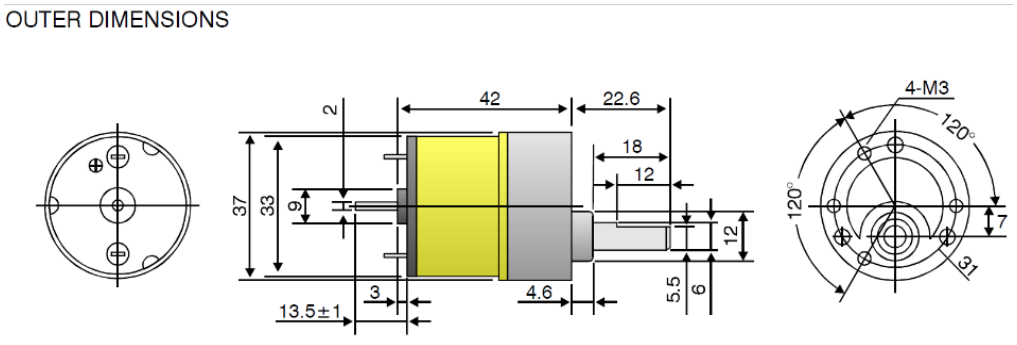
Παρακάτω θα δούμε το όχημα που χρησιμοποιήθηκε στην εργασία, καθώς και τα υπόλοιπα ηλεκτρονικά εξαρτήματα και την συνδεσμολογία μεταξύ τους.

Στα πλαίσια της εργασίας πήραμε το συναρμολογούμενο όχημα Rover 4WD1 Robot, το οποίο μας έδωσε την δυνατότητα να προσθέσουμε τα εξαρτήματα μας πάνω του κατά τη δική μας κρίση.

Εικόνα 5.1Rover 4WD1 Robot

Εικόνα 5.2 Ηλεκτροκινητήρας

Το όχημα αποτελούνταν από το Κυρίως μέρος, δηλαδή το σκελετό του και από 4 ηλεκτροκινητήρες συνεχούς ρεύματος μόνιμου μαγνήτη. Πιο συγκεκριμένα, η ονομαστική τάση των κινητήρων είναι 7.2 volt, ενώ σε αυτήν την τάση, χωρίς φορτίο ο ηλεκτροκινητήρας παρήγαγε 175 RPM στην έξοδο του μειωτήρα. Να αναφέρουμε επίσης ότι ο μειωτήρας είναι 50:1. Στις παρακάτω εικόνες θα δούμε πιο αναλυτικά τα χαρακτηριστικά τους.



Εικόνα 5.3 Διαστάσεις του ηλεκτροκινητήρα

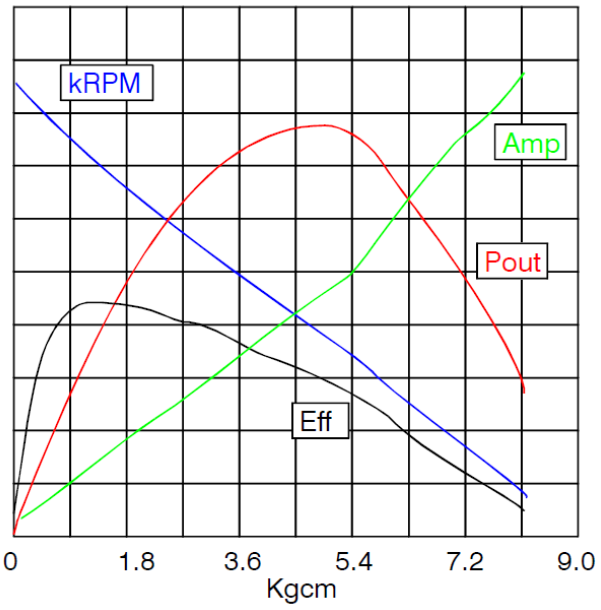
Type: HN-GH35GMB
Model: HN-GH7.2-2414T - 50:1

1. Testing Conditions:
   Temp: 25° Celsius
   Humidity: 60%
   Motor Orientation: Horizontal
2. Rated Voltage: 7.2vdc
3. Voltage Operating Range: 6-7.2vdc
4. Rated Load at 7.2vdc: 1.0Kg-cm
   Do not exceed rated load. Damage may occur!
5. No Load Speed at 7.2vdc: 175 RPM +/- 10%
6. Speed at Rated Load (1.0Kg-cm): 146 RPM +/- 10%
7. No Load Current at 7.2vdc: < 257mA
8. Current at Rated Load (1.0Kg-cm): < 556mA
9. Shaft End-Play: Maximum 0.8m/m
10. Insulation Resistance: 10M ohm at 300vdc
11. Withstand Voltage: 300vdc for 1 Second
12. The gear motor is not intended for instant reverse. The gear motor must be stopped before reversing.
13. The gear motor does not include protection from water or dust etc.

Εικόνα 5.4 Χαρακτηριστικά του ηλεκτροκινητήρα

DRAWING OF CURVES

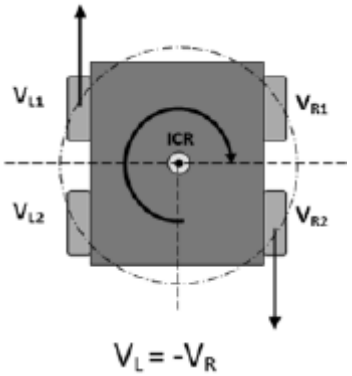
Pout	Amp	Eff	kRPM
5.0	4.0	1.0	0.20
4.5	3.6	0.9	0.18
4.0	3.2	0.8	0.16
3.5	2.8	0.7	0.14
3.0	2.4	0.6	0.12
2.5	2.0	0.5	0.10
2.0	1.6	0.4	0.08
1.5	1.2	0.3	0.06
1.0	0.8	0.2	0.04
0.5	0.4	0.1	0.02
0.0	0.0	0.0	0.00



Εικόνα 5.5 Χαρακτηριστικές καμπύλες του ηλεκτροκινητήρα

6. Συνδεσμολογία εξαρτημάτων

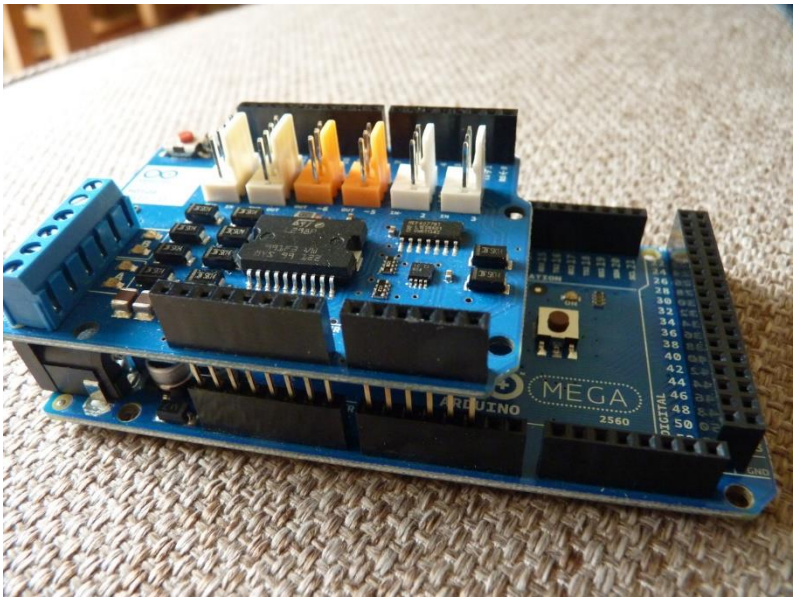
Συνδέσαμε τους δυο αριστερά κινητήρες παράλληλα και το ίδιο κάναμε και για τους δυο δεξιά. Με αυτό τον τρόπο σύνδεσης οι κινητήρες έχουν την ίδια τάση στην είσοδο τους άρα και τις ίδιες στροφές. Έτσι δεν θα χρειαζόταν ξεχωριστός έλεγχος για κάθε ένα κινητήρα χωριστά, αφού έτσι κι αλλιώς, οι δυο δεξιά κινητήρες, θέλαμε να έχουν την ίδια ταχύτητα περιστροφής, όπως θέλαμε και για τους δύο κινητήρες αριστερά. Η συνδεσμολογία αυτή μας βοηθούσε στο να καθοδηγήσουμε το όχημα, ανάλογα με το που έπρεπε να πάει. Για παράδειγμα, αν θέλαμε να στρίψει επιτόπου, δηλαδή ο άξονας γύρω από τον οποίο θα περιστρεφόταν να ήταν στο κέντρο του οχήματος, θα έπρεπε οι δυο κινητήρες δεξιά να έχουν την ίδια γωνιακή ταχύτητα με τους δυο κινητήρες αριστερά και αντίθετη φορά.



Εικόνα 6 Περιστροφή οχήματος

6.1 Σύνδεση ηλεκτροκινητήρων

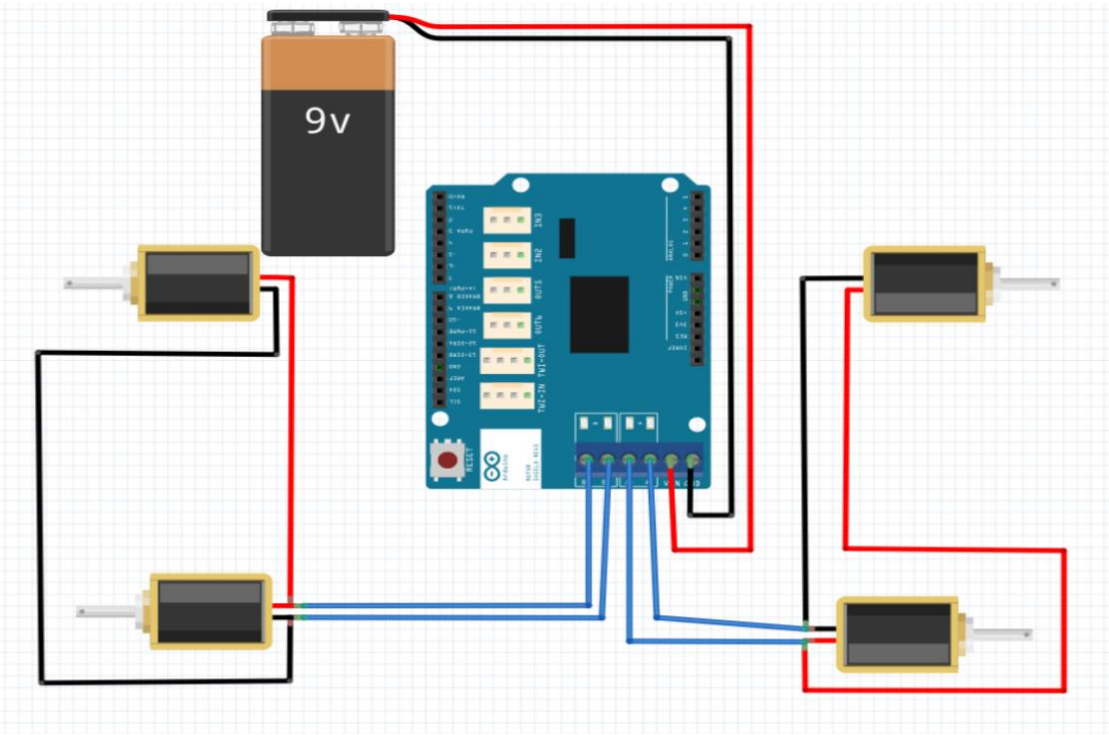
Για την καθοδήγηση των κινητήρων επιλέξαμε την πλακέτα Arduino Motor Shield Rev3. Η πλακέτα αυτή είναι μια πλακέτα που επεκτείνει τις δυνατότητες του arduino. Προσαρμόζεται γρήγορα και εύκολα πάνω στις ηλεκτρονικές πλακέτες του arduino. Μπορεί να κινήσει 2 κινητήρες ταυτόχρονα και ανεξάρτητα και να δώσει στον κάθε ένα, μέχρι και 2 Ampere ρεύμα. Η μέγιστη τάση που μπορούμε να βάλουμε στην είσοδο είναι 12 volts και η μικρότερη είναι 5 volts. Με την ηλεκτρονική πλακέτα Motor Shield Rev3 μπορούμε να ελέγξουμε την ταχύτητα



Εικόνα 6.1.1 Arduino Motor Shield Rev3 καιArduino MEGA

περιστροφής των κινητήρων, καθώς και την κατεύθυνση ανεξάρτητα του ενός από τον άλλο. Αυτό γίνεται χάρη στο ολοκληρωμένο L298, το οποίο είναι μια διπλή γέφυρα σχήματος «ήττα», η οποία έχει σχεδιαστεί για να χειρίζεται επαγωγικά φορτία, όπως ηλεκτρομαγνητικές βαλβίδες, ηλεκτρικούς κινητήρες συνεχούς ρεύματος και βηματικούς κινητήρες.

Για να λειτουργήσουν οι ηλεκτροκινητήρες, χρειάζεται να τοποθετήσουμε την πλακέτα Motor Shield Rev3 πάνω στον arduino. Εμείς επιλέξαμε να την τοποθετήσουμε στον Arduino MEGA 2560, διότι έχει μεγαλύτερες δυνατότητες, τις οποίες ο Arduino UNO δεν θα μας τις έδινε. Μετά από την σύνδεση του Arduino MEGA 2560 με την πλακέτα Motor Shield Rev3, η σύνδεση που ακολουθεί για τους ηλεκτροκινητήρες και τη μπαταρία φαίνεται στην παρακάτω εικόνα.



Εικόνα 6.1.2 Συνδεσμολογία κινητήρων

6.2 Ο κώδικας για την καθοδήγηση των κινητήρων

Μετά από τη συνδεσμολογία, απαραίτητος είναι ο κώδικας που θα γράψουμε, για να καταφέρουμε να κινήσουμε το όχημα μας προς την κατεύθυνση που θέλουμε και με την ταχύτητα που κρίνουμε. Τα Pin που δεσμεύει η πλακέτα Motor Shield Rev3 από τον arduino Mega 2560, για την καθοδήγηση και των δυο κινητήρων είναι τα 3, 11, 12, 13, τα οποία τα δηλώνουμε ως εξόδους για τον arduino. Στις εξόδους 3 και 11, θα εξάγουμε ένα σήμα PWM

για να μπορούμε να παράγουμε ένα μεγάλο εύρος στροφών. Άρα, δηλώνουμε στο πρόγραμμα μας ποιοι θα είναι οι έξοδοι για τους κινητήρες.

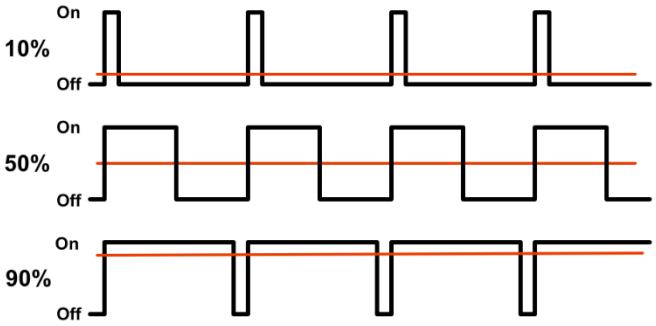
```
voidsetup () {
        pinMode (12, OUTPUT); // κατεύθυνση κινητήρα στο κανάλι A
        pinMode (13, OUTPUT); // κατεύθυνση κινητήρα στο κανάλι B
        pinMode (9, OUTPUT); // Φρένο για το κανάλι A
        pinMode (8, OUTPUT); // Φρένο για το κανάλι B
              }
Void loop () {
 //εκκίνηση και των δυο κινητήρων με τη μέγιστη ταχύτητα και αντίθετη
//φορά για να κατευθυνθεί το όχημα μας προς τα μπροστά.
digitalWrite (9, LOW); // απενεργοποίηση φρένου για τον κινητήρα A
digitalWrite (8, LOW); // απενεργοποίηση φρένου για τον κινητήρα B
digitalWrite (12, LOW); // A
                digitalWrite (13, HIGH); //B
                analogWrite (3,255); // A
                analogWrite (11,255); // B
                digitalWrite (9, HIGH); // φρένο για τον κινητήρα A
                digitalWrite (8, HIGH); // φρένο για τον κινητήρα B
              }
```

6.3 Το σήμα PWM

Το σήμα Pulse Width Modulation (PWM), είναι η διαμόρφωση του εύρους μιας παλμοσειράς. Είναι ένας τρόπος περιγραφής ενός ψηφιακού σήματος, που δημιουργείται μέσω μιας τεχνικής διαμόρφωσης, η οποία περιλαμβάνει την κωδικοποίηση ενός μηνύματος σε ένα παλμικό σήμα. Αν και αυτή η τεχνική διαμόρφωσης μπορεί να χρησιμοποιηθεί για την κωδικοποίηση πληροφοριών για μετάδοση, η κύρια χρήση της είναι να επιτρέπει τον έλεγχο της ισχύος που τροφοδοτείται σε ηλεκτρικές συσκευές, ιδιαίτερα σε αδρανειακά φορτία όπως οι ηλεκτρικοί κινητήρες. Επιπλέον, ο

Εικόνα 6.3.1 Παλμοσειρά PWM

PWM είναι ένας από τους δύο κύριους αλγόριθμους που χρησιμοποιούνται στους φωτοβολταϊκούς φορτιστές ηλιακών συσσωρευτών, ενώ ο άλλος είναι ο εντοπισμός μέγιστων σημείων ισχύος.



Η μέση τιμή της τάσης και του ρεύματος που τροφοδοτείται στο φορτίο ελέγχεται με την ενεργοποίηση και απενεργοποίηση ενός διακόπτη μεταξύ τροφοδοσίας και φορτίου με γρήγορο ρυθμό. Όσο μεγαλύτερο χρόνο είναι ο διακόπτης κλειστός σε σχέση με τις περιόδους που είναι ανοιχτός, τόσο μεγαλύτερη είναι η συνολική ισχύς που παρέχεται στο φορτίο.

Η συχνότητα μεταγωγής PWM πρέπει να είναι πολύ υψηλότερη από αυτή που θα επηρέαζε το φορτίο, αυτό το οποίο σημαίνει ότι η προκύπτουσα κυματομορφή που αντιλαμβάνεται το φορτίο, πρέπει να είναι όσο το δυνατόν πιο ομαλή. Ο ρυθμός στον οποίο πρέπει να ρυθμίζεται η τροφοδοσία μπορεί να ποικίλει σημαντικά ανάλογα με το φορτίο και την εφαρμογή.

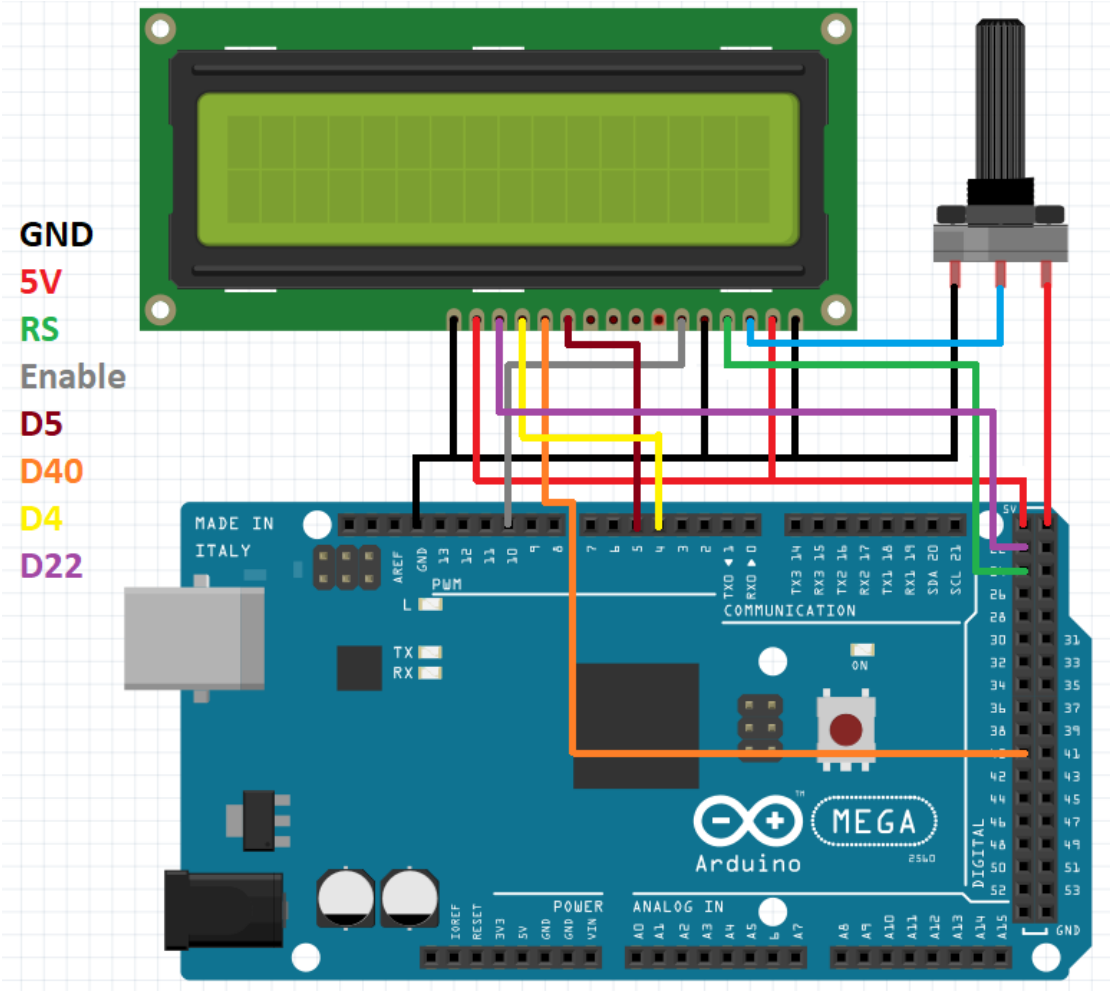
6.4 Περιφερειακά εξαρτήματα

Για τη διευκόλυνση μας, επειδή θα έπρεπε να έχουμε μια εικόνα των αποτελεσμάτων, κατά τη διάρκεια της πορείας του οχήματος, τοποθετήσαμε πάνω σε αυτό μια οθόνη LCD Display 16X2. Στην οθόνη αυτή μπορούμε να παρακολουθούμε τις συντεταγμένες που βρίσκεται το όχημα και κάποια μηνύματα για τυχόν σφάλματα που μπορεί να προκύψουν.

Εικόνα 6.4.1LCD Display 16X2

Μαζί με την LCD οθόνη χρησιμοποιήσαμε ένα ποτενσιόμετρο με εσωτερική αντίσταση 10ΚΩ. Το ποτενσιόμετρο αυτό το τοποθετήσαμε, για να μπορέσουμε να αλλάξουμε τη φωτεινότητα των χαρακτήρων της LCD οθόνης.

Για να πετύχουμε όλα τα παραπάνω, θα πρέπει να συνδέσουμε την οθόνη με τον arduino. Η συνδεσμολογία φαίνεται στην παρακάτω εικόνα.



Εικόνα 6.4.2 Συνδεσμολογία LCD οθόνης

Μετά από την συνδεσμολογία, σειρά έχει ο κώδικας τον οποίο θα πρέπει να γράψουμε για να μπορέσουμε να βάλουμε σε λειτουργία την οθόνη μας. Οι εντολές είναι οι παρακάτω. Με την εντολή **#include<LiquidCrystal.h>** καλούμε την βιβλιοθήκη της LCD οθόνης ενώ στην συνέχεια με την εντολή **LiquidCrystallcd (24, 10, 5, 40, 4, 22);** δηλώνουμε τις θύρες που την έχουμε συνδέσει με τον arduino.

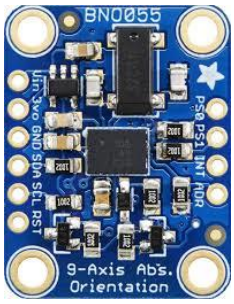
Στο void setup καλούμε την συνάρτηση που θα ενεργοποιήσει την επικοινωνία ανάμεσα στον arduino και την LCD οθόνη. Αυτό γίνεται με την εντολή **lcd.begin(16,2);** Έπειτα μπορούμε να τυπώσουμε πάνω στην οθόνη με την εντολή **lcd.print(" hello, World! ");** Και ο κώδικας ολοκληρώνεται στην συνέχεια.

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(24, 10, 5, 40, 4, 22);

void setup() {
  // set up the LCD's number of columns and rows:
lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
lcd.setCursor(0, 1);
  // print the number of seconds since reset:
lcd.print(millis()/1000);
}
```

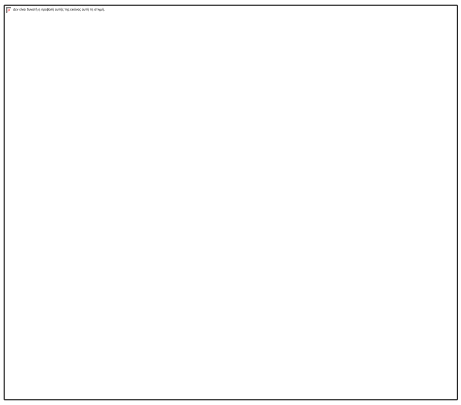
7. Ο αισθητήρας BNO055

7.1 Ανάλυση του αισθητήρα

Οι κινητήρες που ήταν τοποθετημένοι πάνω στο όχημα δεν είχαν αισθητήρες γωνιακής ταχύτητας. Αυτό ήταν ένα μεγάλο μειονέκτημα που έπρεπε να το αντιμετωπίσουμε με κάποιο τρόπο. Το συγκεκριμένο πρόβλημα το λύσαμε με ένα αισθητήρα που μετράει την ταχύτητα και την επιτάχυνση σε τρεις άξονες. Ο αισθητήρας αυτός είναι ο BNO055 της εταιρίας Adafruit. Χρησιμοποιώντας τον αισθητήρα αυτό, έχουμε την δυνατότητα να μετρήσουμε την γωνία περιστροφής τριών αξόνων. Επίσης, μπορούμε να μετρήσουμε την επιτάχυνση πάνω σε αυτούς τους τρείς άξονες.



Εικόνα 7.1.1 Ολοκληρωμένο κύκλωμα αισθητήρας BNO055



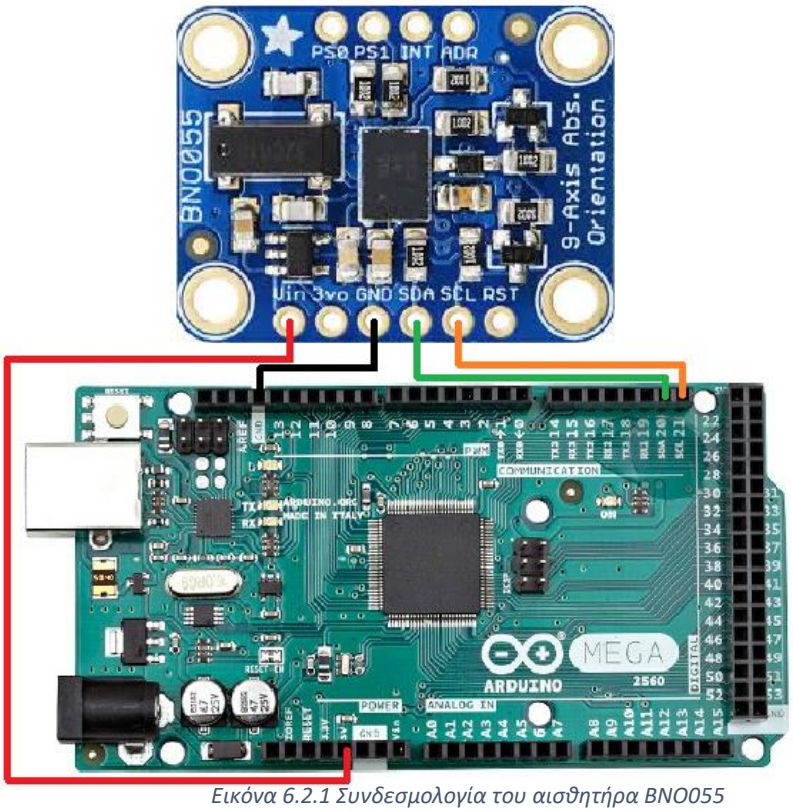
Εικόνα 7.1.2 BNO055

Το BNO055 είναι ένας έξυπνος αισθητήρας απόλυτου προσανατολισμού εννέα αξόνων, τον οποίο χρησιμοποιήσαμε για την πτυχιακή μας εργασία. Ο αισθητήρας BNO055 είναι μια ηλεκτρονική συσκευή η οποία έχει ενσωματωμένο ένα τριαξονικό επιταχυνσιόμετρο 14 bit, ένα τριαξονικό γυροσκόπιο 16 bit με εύρος ± 2000 μοίρες ανά δευτερόλεπτο, ένα τριαξονικό γεωμαγνητικό αισθητήρα και έναν επεξεργαστή υψηλής ταχύτητας ARM Cortex M0 32bit μαζί με έναν μικροελεγκτή που εκτελεί το λογισμικό σύντηξης του αισθητήρα. Για τη βέλτιστη ολοκλήρωση του συστήματος, το BNO055 είναι εξοπλισμένο με ψηφιακές αμφίδρομες διεπαφές I2C και UART.

Ο αισθητήρας BNO055 μπορεί να μας δώσει τα ακόλουθα δεδομένα. Δεδομένα προσανατολισμού τριών αξόνων που βασίζονται σε σφαίρα 360^{o}, τετραγωνική έξοδο τεσσάρων σημείων για ακριβέστερο χειρισμό δεδομένων, δεδομένα γωνιακής επιτάχυνσης τριών αξόνων, τρείς άξονες γεωμαγνητικού πεδίου, δεδομένα τριών αξόνων γραμμικής επιτάχυνσης και θερμοκρασία περιβάλλοντος σε βαθμούς κελσίου.

7.2 Η συνδεσμολογία του αισθητήρα

Για να εξάγουμε τα δεδομένα που μας δίνει ο αισθητήρας BNO055 χρειαζόταν να τον συνδέσουμε στον arduino. Η συνδεσμολογία είναι αρκετά απλή. Αρχικά συνδέουμε το GND και το Vin του BNO055 με το GND και το 5V του Arduino αντίστοιχα. Έπειτα συνδέουμε το SDA και το SCS του BNO055 με το SDA και το SCS του Arduino αντίστοιχα και αυτή ήταν η συνδεσμολογία, η οποία φαίνεται και στην διπλανή εικόνα.



Εικόνα 6.2.1 Συνδεσμολογία του αισθητήρα BNO055

7.3 Βιβλιοθήκη για τον αισθητήρα

Μετά από τη συνδεσμολογία, για να μπορούμε να εμφανίσουμε τα δεδομένα από το BNO055 και να τα επεξεργαστούμε, χρειάζεται ο απαραίτητος κώδικας και κάποιες ενέργειες, έτσι ώστε να βαθμονομήσουμε σωστά την συσκευή μας. Αρχικά, βρήκαμε τις απαραίτητες βιβλιοθήκες, οι οποίες ήταν ικανές να διαχειριστούν τα δεδομένα και να μας προσφέρουν χρήσιμες πληροφορίες που θα μπορούσαμε να επεξεργαστούμε με ευκολία.

Η συγκεκριμένη βιβλιοθήκη υποστηρίζει τον αισθητήρα BNO055 και ορίζει κάποιες βασικές πληροφορίες του αισθητήρα, όπως για παράδειγμα τα όρια των επιμέρους αισθητηρίων και επιστρέφει τυποποιημένες μονάδες με βάση το Διεθνές σύστημα μονάδων (S.I.), συγκεκριμένου τύπου και κλίμακας για κάθε τύπο αισθητήριου που υποστηρίζεται. Αυτό είναι ένα πολύ σημαντικό κομμάτι, διότι μπορούμε να χρησιμοποιήσουμε τα δεδομένα αμέσως αφού έχουν ήδη μετατραπεί σε μονάδες που γνωρίζουμε και μπορούμε να τις

διαχειριστούμε εύκολα και γρήγορα σε σχέση με τιμές από 0-1023, που θα πρέπει πρώτα να κατανοήσουμε τι ακριβώς σημαίνουν.

Για να αποκτήσουμε τη βιβλιοθήκη του αισθητήρα BNO055 και να την κάνουμε χρήσιμη και συνεργάσιμη, πρέπει πρώτα να βρούμε ένα πάροχο, ο οποίος να μπορεί να μας παράσχει την βιβλιοθήκη και έπειτα να την εντάξουμε στο λογισμικό arduinoIDE. Πιο συγκεκριμένα, εμείς κατεβάσαμε την βιβλιοθήκη στον υπολογιστή μας, από την σελίδα github.com. Η ιστοσελίδα αυτή είναι μία πολύ γνωστή σελίδα στο διαδίκτυο, που παρέχει αρκετές πληροφορίες, ελεύθερο κώδικα, βιβλιοθήκες και άλλα.

Έπειτα από την ανάκτηση και την εγκατάσταση της βιβλιοθήκης, ανοίγοντας τον φάκελο της, παρατηρούμε ότι έχει μέσα κάποια παραδείγματα για την πρώτη επαφή με τον αισθητήρα BNO055, τα οποία χρησιμοποιήσαμε για να πάρουμε μια πρώτη εικόνα των δεδομένων. Ο κώδικας του παραδείγματος μας έδωσε τα εξής δεδομένα. Καθώς περιστρέφαμε τον αισθητήρα ως προς τους τρεις άξονες , στην οθόνη μας παρουσιαζόταν η τιμή της κάθε γωνίας του κάθε άξονα όσες μοίρες είχαν περιστραφεί, με μεγάλη ακρίβεια έως και δυο δεκαδικά ψηφία.

7.4 Βαθμονόμηση του αισθητήρα

Επειδή υπάρχουν χιλιάδες αισθητήρες και ο κώδικας δεν μπορεί να είναι διαφορετικός για τον κάθε ένα από αυτούς, διότι οι τιμές των δεδομένων που θα λάβουμε μπορεί να είναι διαφορετικές και επηρεασμένες από εξωτερικούς παράγοντες, όπως κλιματολογικές αλλαγές και διαφορά υψόμετρου. Γι’ αυτό το λόγο χρειάζεται να γίνει μια βαθμονόμηση του αισθητήρα, για να μπορέσουμε να πάρουμε σωστά και απόλυτα αποτελέσματα, από όλα τα επιμέρους αισθητήρια. Για να γίνει αυτό, πρέπει να εισάγουμε ένα συγκεκριμένο κώδικα και να κάνουμε κάποιες απαραίτητες ενέργειες, έτσι ώστε να καταφέρουμε να ανακτήσουμε τα νούμερα τις βαθμονόμησης.

Για να βαθμονομήσουμε τον αισθητήρα μας, χρειάστηκε να εισάγουμε στον arduino ένα κώδικα βαθμονόμησης του αισθητήρα BNO055 και να τον εκτελέσουμε έχοντας συνδεμένο τον αισθητήρα μας. Καθώς εκτελείται ο κώδικας, βλέπουμε στη σειριακή μας οθόνη τρείς μεταβλητές μηδενισμένες. Οι μεταβλητές αυτές αντιπροσωπεύουν τους 3 επιμέρους αισθητήρες που είναι το επιταχυνσιόμετρο, το γυροσκόπιο και το μαγνητόμετρο.

Καθώς μετακινούμε τον αισθητήρα προς διάφορες κατευθύνεις, παρατηρούμε ότι οι τιμές των αισθητήρων αλλάζουν από τις τιμές 0 μέχρι 3. Όταν και οι τρείς αισθητήρες συγχρονιστούν στον αριθμό 3 σημαίνει ότι είναι πλήρως βαθμονομημένοι ως προς τους τρεις άξονες του κάθε ενός. Στην συνέχεια σταματάει να εκτελείται ο κώδικας και παρουσιάζονται τα δεδομένα βαθμονόμησης, τα οποία είναι τα εξής:

```
Accelerometer: 65535 65535 65535
Gyro: 65535 65535 65535
Mag: 65535 65535 65535
Accel Radius: 65535
Mag Radius: 65535
```

7.5 Ο κώδικας βαθμονόμησης

Το πρόγραμμα αποθηκεύει αυτόματα τα δεδομένα βαθμονόμησης και έτσι δεν χρειάζεται να επαναλαμβάνουμε αυτή την ενέργεια κάθε φορά που χρησιμοποιούμε τον αισθητήρα αυτόν. Παρακάτω θα δούμε τον κώδικα βαθμονόμησης του αισθητήρα BNO055.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <EEPROM.h>
#define BNO055_SAMPLERATE_DELAY_MS (100)
Adafruit_BNO055 bno = Adafruit_BNO055(55);

void displaySensorDetails(void)
{
sensor_t sensor;
bno.getSensor(&sensor);
Serial.println("------------------------------------");
Serial.print("Sensor:       "); Serial.println(sensor.name);
```

```
Serial.print("Driver Ver:   "); Serial.println(sensor.version);
Serial.print("Unique ID:    "); Serial.println(sensor.sensor_id);
Serial.print("Max Value:    "); Serial.print(sensor.max_value); Serial.println(" xxx");
Serial.print("Min Value:    "); Serial.print(sensor.min_value); Serial.println(" xxx");
Serial.print("Resolution:   "); Serial.print(sensor.resolution); Serial.println(" xxx");
Serial.println("------------------------------------");
Serial.println("");
delay(500);
}

void displaySensorStatus(void)
{
/* Get the system status values (mostly for debugging purposes) */
uint8_t system_status, self_test_results, system_error;
system_status = self_test_results = system_error = 0;
bno.getSystemStatus(&system_status, &self_test_results, &system_error);

/* Display the results in the Serial Monitor */
Serial.println("");
Serial.print("System Status: 0x");
Serial.println(system_status, HEX);
Serial.print("Self Test:     0x");
Serial.println(self_test_results, HEX);
Serial.print("System Error:  0x");
Serial.println(system_error, HEX);
Serial.println("");
delay(500);
}

void displayCalStatus(void)
{
```

```
/* Get the four calibration values (0..3) */
/* Any sensor data reporting 0 should be ignored, */
/* 3 means 'fully calibrated" */
uint8_t system, gyro, accel, mag;
system = gyro = accel = mag = 0;
bno.getCalibration(&system, &gyro, &accel, &mag);

/* The data should be ignored until the system calibration is > 0 */
Serial.print("\t");
if (!system)
{
Serial.print("! ");
}

/* Display the individual values */
Serial.print("Sys:");
Serial.print(system, DEC);
Serial.print(" G:");
Serial.print(gyro, DEC);
Serial.print(" A:");
Serial.print(accel, DEC);
Serial.print(" M:");
Serial.print(mag, DEC);
}

//  Display the raw calibration offset and radius data
void displaySensorOffsets(const adafruit_bno055_offsets_t &calibData)
{
Serial.print("Accelerometer: ");
Serial.print(calibData.accel_offset_x); Serial.print(" ");
Serial.print(calibData.accel_offset_y); Serial.print(" ");
```

```
Serial.print(calibData.accel_offset_z); Serial.print(" ");

Serial.print("\nGyro: ");
Serial.print(calibData.gyro_offset_x); Serial.print(" ");
Serial.print(calibData.gyro_offset_y); Serial.print(" ");
Serial.print(calibData.gyro_offset_z); Serial.print(" ");

Serial.print("\nMag: ");
Serial.print(calibData.mag_offset_x); Serial.print(" ");
Serial.print(calibData.mag_offset_y); Serial.print(" ");
Serial.print(calibData.mag_offset_z); Serial.print(" ");

Serial.print("\nAccel Radius: ");
Serial.print(calibData.accel_radius);

Serial.print("\nMag Radius: ");
Serial.print(calibData.mag_radius);
}

//    Arduino setup function (automatically called at startup)
void setup(void)
{
Serial.begin(115200);
delay(1000);
Serial.println("Orientation Sensor Test"); Serial.println("");

/* Initialise the sensor */
if (!bno.begin())
{
/* There was a problem detecting the BNO055 ... check your connections */
Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
```

```
while (1);
}

int eeAddress = 0;
long bnoID;
bool foundCalib = false;

EEPROM.get(eeAddress, bnoID);

adafruit_bno055_offsets_t calibrationData;
sensor_t sensor;

/*
*  Look for the sensor's unique ID at the beginning oF EEPROM.
*  This isn't foolproof, but it's better than nothing.
*/
bno.getSensor(&sensor);
if (bnoID != sensor.sensor_id)
{
Serial.println("\nNo Calibration Data for this sensor exists in EEPROM");
delay(500);
}
else
{
Serial.println("\nFound Calibration for this sensor in EEPROM.");
eeAddress += sizeof(long);
EEPROM.get(eeAddress, calibrationData);

displaySensorOffsets(calibrationData);

Serial.println("\n\nRestoring Calibration data to the BNO055...");
```

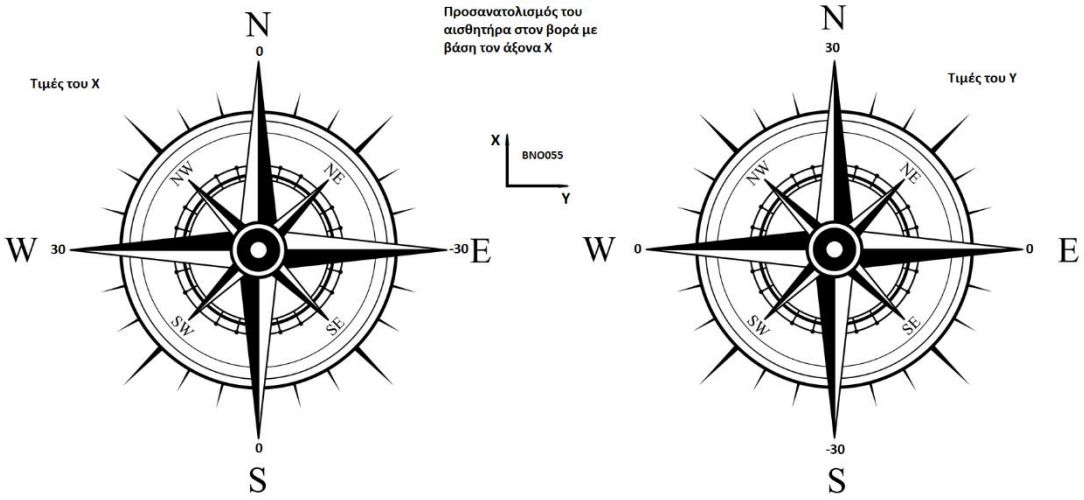
```
bno.setSensorOffsets(calibrationData);

Serial.println("\n\nCalibration data loaded into BNO055");
foundCalib = true;
}

delay(1000);

/* Display some basic information on this sensor */
displaySensorDetails();

/* Optional: Display current status */
displaySensorStatus();

bno.setExtCrystalUse(true);

sensors_event_t event;
bno.getEvent(&event);
if (foundCalib){
Serial.println("Move sensor slightly to calibrate magnetometers");
while (!bno.isFullyCalibrated())
{
bno.getEvent(&event);
delay(BNO055_SAMPLERATE_DELAY_MS);
}
}
else
{
Serial.println("Please Calibrate Sensor: ");
while (!bno.isFullyCalibrated())
{
```

```
bno.getEvent(&event);

Serial.print("X: ");
Serial.print(event.orientation.x, 4);
Serial.print("\tY: ");
Serial.print(event.orientation.y, 4);
Serial.print("\tZ: ");
Serial.print(event.orientation.z, 4);

/* Optional: Display calibration status */
displayCalStatus();

/* New line for the next sample */
Serial.println("");

/* Wait the specified delay before requesting new data */
delay(BNO055_SAMPLERATE_DELAY_MS);
}
}

Serial.println("\nFully calibrated!");
Serial.println("--------------------------------");
Serial.println("Calibration Results: ");
adafruit_bno055_offsets_t newCalib;
bno.getSensorOffsets(newCalib);
displaySensorOffsets(newCalib);

Serial.println("\n\nStoring calibration data to EEPROM...");

eeAddress = 0;
bno.getSensor(&sensor);
```

```
bnoID = sensor.sensor_id;

EEPROM.put(eeAddress, bnoID);

eeAddress += sizeof(long);
EEPROM.put(eeAddress, newCalib);
Serial.println("Data stored to EEPROM.");

Serial.println("\n--------------------------------\n");
delay(500);
}

void loop() {
/* Get a new sensor event */
sensors_event_t event;
bno.getEvent(&event);

/* Display the floating point data */
Serial.print("X: ");
Serial.print(event.orientation.x, 4);
Serial.print("\tY: ");
Serial.print(event.orientation.y, 4);
Serial.print("\tZ: ");
Serial.print(event.orientation.z, 4);

/* Optional: Display calibration status */
displayCalStatus();

/* Optional: Display sensor status (debug only) */
//displaySensorStatus();
```

```
/* New line for the next sample */
Serial.println("");

/* Wait the specified delay before requesting new data */
delay(BNO055_SAMPLERATE_DELAY_MS);
}
```

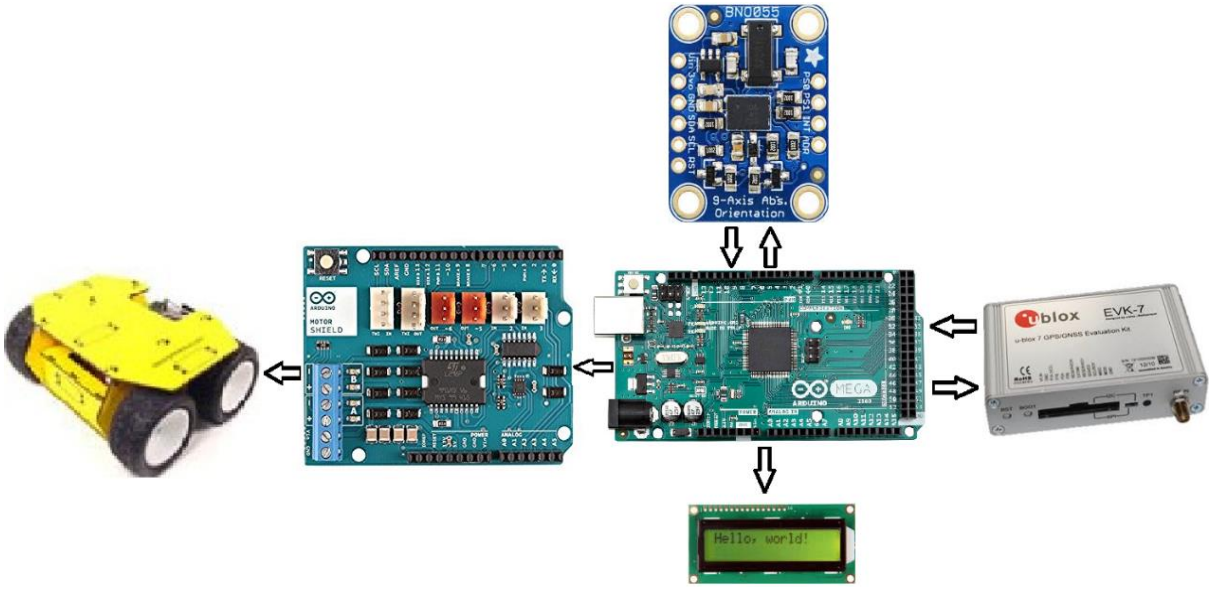
Μετά από τη βαθμονόμηση του αισθητήρα, για να πάρουμε τα δεδομένα που χρειαζόμαστε, φτιάξαμε έναν άλλο κώδικα ο οποίος θα συμβάδιζε με την πτυχιακή μας εργασία. Στον κώδικα αυτό, καταφέραμε να μετατρέψουμε τα δεδομένα που μας έδινε το μαγνητόμετρο σε μοίρες, έτσι ώστε να είναι πιο κατανοητά και χρήσιμα για εμάς. Το μαγνητόμετρο μας έδινε τις τιμές για τον προσανατολισμό του άξονα X, από 0 μέχρι -30 για προσανατολισμό από τον βορά μέχρι την ανατολή, από -30 μέχρι 0 από την ανατολή μέχρι τον νότο, από 0 μέχρι 30 από τον νότο μέχρι την δύση και από 30 μέχρι 0 από την δύση μέχρι τον βορρά. Παρατηρήσαμε ότι ο προσανατολισμός του άξονα X και του άξονα Y διαφέρουν κατά ένα τεταρτημόριο, το οποίο μας βοηθάει στο να μετατρέψουμε τις τιμές που μας δίνει ο αισθητήρας μας σε μοίρες πολύ πιο εύκολα.



Εικόνα 6.5.1 Προσανατολισμός του αισθητήρα

7.6 Η λογική του συστήματος

Η πληροφορίες που εισάγονται στην μικροελεγκτή arduino είναι η θέση η οποία βρίσκετε το όχημα και ο προσανατολισμός του, ως δεδομένο έχουμε βάλει την θέση την οποία θέλουμε να πάει το όχημα μας. Την θέση που βρίσκεται το όχημα μας μας την δίνει το GPS μέσο της σειριακής επικοινωνίας, η θέση του οχήματος βασίζετε σε δύο αριθμούς, το γεωγραφικό μήκος και το γεωγραφικό πλάτος. Για την απόλυτη ακρίβεια θα μπορούσαμε να συμπεριλάβουμε και το ύψος το οποίο βρίσκεται το όχημα αλλά στην περίπτωση μας δεν έχει νόημα διότι το όχημα πατάει πάνω στο έδαφος. Στην περίπτωση που είχαμε το ίδιο σύστημα ελέγχου θέσης για ένα ιπτάμενο όχημα, τότε το ύψος θα ήταν υποχρεωτικό.

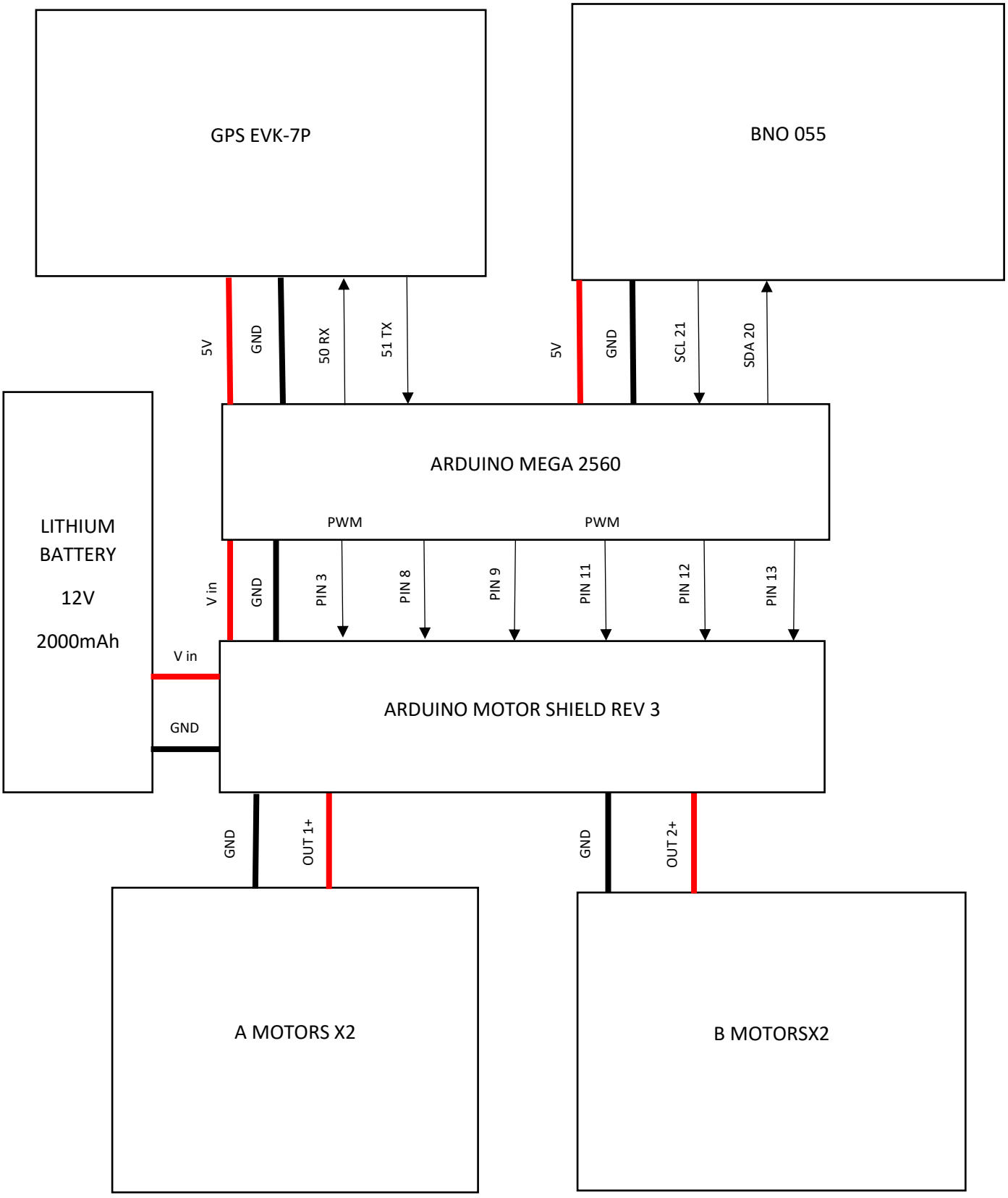


Όταν θέτουμε σε λειτουργία το όχημα, εκτελούντες αναλυτικά οι εξής ενέργειες. Ανοίγει η σειριακή επικοινωνία με την LCD οθόνη, για να μας ενημερώνει σε ποια φάση της διαδικασίας βρισκόμαστε. Στην συνέχεια ανοίγει την σειριακή επικοινωνία με τον αισθητήρα προσανατολισμού BNO 055 και εκτελεί μια συνθήκη βαθμονόμησης για τον προσανατολισμό του αισθητήρα και κατά συνέπεια και του οχήματος.

Μετά από τις παραπάνω ενέργειες εκτελείτε ο κύριος κώδικας, ο οποίος επαναλαμβάνετε συνεχώς, μέχρι το όχημα μας να φτάσει στον προορισμό του. Ο Arduino αρχίζει να επικοινωνεί με το GPS, το οποίο στέλνει συνεχώς συντεταγμένες για την θέση την οποία βρίσκετε το όχημα μας. Καθώς ο Arduino διαβάζει την θέση που βρίσκεται από το GPS

και ξέρει ποια είναι η θέση που πρέπει να πάει, υπολογίζει την διαδρομή που πρέπει να ακολουθήσει με βάση τον προσανατολισμό. Στην συνέχεια, διαβάζει τον προσανατολισμό του οχήματος από τον αισθητήρα BNO 055 και δίνει τις κατάλληλες εντολές στην βοηθητική πλατφόρμα καθοδήγησης ηλεκτροκινητήρων ARDUINO MOTOR SHIELD, η οποία κινεί τους κινητήρες του οχήματος κατάλληλα, έτσι ώστε να ταυτίζετε ο προσανατολισμός του οχήματος με τον προσανατολισμό του σημείου που πρέπει να πάει.

7.7 Ηλεκτρολογικό διάγραμμα συνδέσεων



8. Σύνοψη

Στην παρούσα πτυχιακή εργασία κατασκευάσαμε ένα μη επανδρωμένο όχημα, το οποίο μπορεί να πλοηγηθεί σε οποιαδήποτε διαδρομή, με χρήση του συστήματος GPS, χωρίς την ανθρώπινη καθοδήγηση. Για την κατασκευή χρησιμοποιήθηκαν διάφορα εμπορικώς διαθέσιμα εργαλεία τα οποία συνοπτικά ήταν: η συσκευή EVK-7P για τη λήψη του σήματος GPS, ο μικροελεγκτής Arduino Mega 2560, το μοντέλο Rover 4WD1 Robot, η πλακέτα Arduino Motor Shield Rev3, η οθόνη LCD Display 16X2 καθώς και ο αισθητήρας BNO055.

Για την ολοκλήρωση διασύνδεσης των διαφορετικών στοιχείων με το όχημα εφαρμόστηκε ο κατάλληλος προγραμματιστικός κώδικας ο οποίος παρατίθεται με τα κατάλληλα σχόλια στο Παράρτημα, θέτοντας την αναπαραγωγή του πολύ εύκολη. Τελικά, μπορούμε να πούμε ότι ο στόχος της αυτοματοποιημένης πλοήγησης του οχήματος, χωρίς την ανθρώπινη παρέμβαση, εστέφθη με πλήρη επιτυχία.

Παράρτημα: Ο κώδικας που χρησιμοποιήθηκε

```
/* ο παρακάτω κώδικας γράφτηκε στα πλαίσια της πτυχιακής εργασίας και έχει ως αποτέλεσμα να καθοδηγήσει το όχημα Rover 4WD1 Robot διαβάζονταν τις συντεταγμένες από μια συσκευή GPSτην EVK-7Pτης εταιρίας Ublox */

/* Η παρακάτω βιβλιοθήκη μας επιτρέπει να επιλέξουμε διαφορετικές θύρες σειριακής επικοινωνίας στην ηλεκτρονική πλακέτα arduino. Έχουμε επιλέξει την θύρα 50 για την λήψη και την θύρα 51 για την αποστολή δεδομένων.*/
#include <SoftwareSerial.h>
SoftwareSerial SoftSerial(50, 51); // RX, TX

/* Ηπαρακάτω βιβλιοθήκη μας βοηθάει να προγραμματίσουμε την οθόνη υγρών κρυστάλλων που έχουμε τοποθετήσει πάνω στο όχημα για να μας εμφανίζει διάφορα μηνύματα. */
#include<LiquidCrystal.h>
LiquidCrystallcd (24, 10, 5, 40, 4, 22);/* Μετηνεντολή αυτή ορίζουμε ποιες θύρες θα χρησιμοποιήσει η οθόνη από τον arduino. */
lcd.begin(16,2);
lcd.print(" hello, World! ");
Delay(1000);
lcd.clear();/* Μετηνεντολή αυτή καθαρίζει η οθόνη από ότι έχει γραμμένο.

//βιβλιοθήκεςπουχρησιμοποιείτοGPS.
#include "TinyGPS++.h"
TinyGPSPlus gps;
```

```
// Βιβλιοθήκες και μεταβλητές που χρησιμοποιεί ο αισθητήρας BNO055.
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#define BNO055_SAMPLERATE_DELAY_MS (100)
Adafruit_BNO055 bno = Adafruit_BNO055();
floatMag = 0.0;

//θύρες εξόδου για φωτεινές διόδους LED.
const int Gled = 48; // πράσινο LED.
const int Rled = 49; //κόκκινο LED.
const int Yled = 46;// κίτρινο LED.

// Έναρξη αρχικών ρυθμίσεων.
voidsetup()
{
/*Έναρξη σειριακής επικοινωνίας σε περίπτωση που έχουμε συνδεμένο τον υπολογιστή μας*/
Serial.begin(115200);
while(!Serial)
  {
   ; // περιμένουμε να συνδεθεί η σειριακή θύρα. Απαιτείται μόνο για τη θύρα USB.
}
Serial.println("HELO!");
  // set the data rate for the SoftwareSerial port
SoftSerial.begin(9600);

  if (!bno.begin())
{
```

```
/* Η συνθήκη αυτή εκτελείτε αν Παρουσιαστή κάποιο πρόβλημα κατά την ανίχνευση του BNO055 και μας λέει να ελέγξουμε τις συνδέσεις μας */
Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
lcd.setCursor(0,0);
lcd.print(("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
while (1);
  }
lcd.clear();
//δήλωσηκατάστασης για τις θύρες των κινητήρων.
pinMode(3, OUTPUT);
pinMode(12, OUTPUT);
pinMode(11, OUTPUT);
pinMode(13, OUTPUT);
// δήλωση κατάστασης για τις θύρες των LED.
pinMode(Gled, OUTPUT);
pinMode(Rled, OUTPUT);
pinMode(Yled, OUTPUT);
// ανοίγουμε και κλείνουμε τα LED με την σειρά.
digitalWrite(Rled, HIGH);
delay(500);
digitalWrite(Yled, HIGH);
delay(500);
digitalWrite(Gled, HIGH);
delay(500);
digitalWrite(Rled, LOW);
delay(500);
digitalWrite(Yled, LOW);
delay(500);
digitalWrite(Gled, LOW);
delay(500);
```

```
// Καλούμε την παρακάτω συνάρτηση για να γίνει βαθμονόμηση της ηλεκτρονικής μας πυξίδας.
Calibration_magnetometer();
}
//Έναρξη κυρίως προγράμματος που εκτελείτε συνέχεια.
voidloop()
{
/* Αν η σειριακή θύρα που έχουμε ορίσει είναι διαθέσιμη και έτοιμη να μας μεταφέρει τα δεδομένα τότε εκτελείτε η παρακάτω συνθήκη "if". */
if (SoftSerial.available())
  {
// Διαβάζουμε τα δεδομένα από την σειριακή θύρα που έχουμε ορίσει.
gps.encode(SoftSerial.read());
  }
/* Αν τα δεδομένα που μας στέλνει το GPS είναι ανανεωμένα τότε εκτελείτε η συνθήκη "if". */
if (gps.location.isUpdated())
{
digitalWrite(Gled, HIGH);// Ανάβειτοπράσινο LED.
digitalWrite(Rled, LOW);// Σβήνειτοκόκκινο LED.
floatLat = gps.location.lat();/* Στην μεταβλητή Lat καταχωρείτε το γεωγραφικό πλάτος που διαβάζουμε από το GPS. */
floatLng = gps.location.lng();/* Στην μεταβλητή Lng καταχωρείτε το γεωγραφικό μήκος που διαβάζουμε από το GPS. */

/* Εκτυπώνουμε τις τιμές του γεωγραφικού πλάτους και μήκους στη σειριακή μας οθόνη με έξι δεκαδικά ψηφία σε περίπτωση που έχουμε συνδέσει τον arduino με τον υπολογιστή μας. */
Serial.print("LAT=");
Serial.print(Lat, 6);
Serial.print("  LNG=");
```

```
Serial.print(Lng, 6);
/* Αν έχει καταχωρηθεί τιμή για την τοποθεσία που βρισκόμαστε, τοτε εκτελείτε η παρακάτω συνθήκη “if”. */
if (gps.location.isValid())
  {
/* Παρακάτω δηλώνουμε τις συντεταγμένες του σημείου που θέλουμε να πάει το όχημα μας. */
static const double Next_LAT = 35.032283, Next_LON = 25.737298;
/* Η παρακάτω συνάρτηση παίρνει σαν δεδομένα τις συντεταγμένες που βρίσκετε το όχημα και τις συντεταγμένες που θέλουμε να πάει και υπολογίζει την απόσταση μεταξύ αυτόν των δυο σημείων. */
double distanceToNext = TinyGPSPlus::distanceBetween(gps.location.lat(), gps.location.lng(), Next_LAT,  Next_LON);
/* Η παρακάτω συνάρτηση παίρνει σαν δεδομένα τις συντεταγμένες που βρίσκετε το όχημα και τις συντεταγμένες που θέλουμε να πάει και υπολογίζει τον προσανατολισμό του σημείου που θέλουμε να πάμε σε μοίρες σε σχέση με τον προσανατολισμό του βορα. */
double courseToNext = TinyGPSPlus::courseTo(gps.location.lat(), gps.location.lng(), Next_LAT, Next_LON);
Serial.print(F("  To next point    Distance="));
Serial.print(distanceToNext, 2);
Serial.print(F(" m Course-to="));
Serial.print(courseToNext, 2);
Serial.print(F(" degrees "));
lcd.setCursor(0,0);
lcd.print(distanceToNext, 2);
    lcd.print(“  m to next”);
lcd.setCursor(0,1);
lcd.print(courseToNext, 2);
lcd.print(“  deg to next”);
```

```
/* Η μεταβλητή Mag παίρνει τιμές από την συνάρτηση Magnetometer() η οποία υπολογίζει τον προσανατολισμό του οχήματος σε μοίρες. */
Mag = Magnetometer();
Serial.print("  Mag=");
    Serial.print(Mag, 2);
/* Η παρακάτω συνθήκη if ελέγχει αν η απόσταση του σημείου που θέλουμε να πάμε είναι μικρότερη του ενός μέτρου και μόλις εκτελεστεί ακινητοποιεί το όχημα. */
if (distanceToNext< 1)
{
digitalWrite(Gled, LOW);
digitalWrite(Rled, HIGH);
Serial.println("  stop");
digitalWrite(12, LOW); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 0);  // A
analogWrite(11, 0); // B
delay(1000);
digitalWrite(Rled, LOW);
    }
/* Αλλιώς αν είναι μεγαλύτερη η απόσταση από ένα μέτρο τότε το όχημα θα συνεχίσει να κινείτε μέχρι να βρει το σημείο που του ορίσαμε. */
else
{
/* Γιανα κινηθεί το όχημα προς το σημείο που του έχουμε ορίσει, φτιάξαμε τρείς συνθήκες ΄΄ if -elseif-else΄΄ έτσι ώστε αν η ηλεκτρονική πυξίδα μας δώσει μεγαλύτερη τιμή από την τιμή που έχουμε υπολογίσει ότι είναι ο προσανατολισμός του σημείου που θέλουμε να πάμε, τότε θα κατευθυνθεί το όχημα αριστερά, αλλιώς αν είναι μικρότερη θα κατευθυνθεί δεξιά,  αλλιώς θα πάει ευθεία. */
if ( Mag> (courseToNext + 2))
```

```
      {
digitalWrite(Yled, HIGH);
Serial.println("  turnleftt");
digitalWrite(12, LOW); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 40);  // A
analogWrite(11, 120); // B
digitalWrite(Yled, LOW);
      }
      else if ( Mag< (courseToNext - 2))
      {
digitalWrite(Yled, HIGH);
Serial.println("  turn right");
digitalWrite(12, LOW); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 120);  // A
analogWrite(11, 40); // B
digitalWrite(Yled, LOW);
      }
      else
      {
digitalWrite(Gled, HIGH);
Serial.println("  gostrait");
digitalWrite(12, LOW); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 120);  // A
analogWrite(11, 120); // B
digitalWrite(Gled, LOW);
}
    }
   }
```

```
  }
}
/* Η παρακάτω συνάρτηση μας δίνει τον προσανατολισμό του οχήματος μας σε μοίρες. */
floatMagnetometer()
{
imu::Vector<3> magnetometer = bno.getVector(Adafruit_BNO055::VECTOR_MAGNETOMETER);
  float X = magnetometer.x();
  float Y = magnetometer.y();
  float Z = magnetometer.z();
  float Xm = 0.0;
  if (X > 31 or Y > 31 or X < -31 or Y < -31 )
  {
//    Calibration_magnetometer();
  }
  if (X <= 0 and X >= -30 and Y >= 0 and Y <= 30)
  {
Xm = map(X, 0, -30, 0, 90);
  }
  if (X <= 0 and X >= -30 and Y <= 0 and Y >= -30)
  {
Xm = map(X, -30, 0, 90, 180);
  }
  if (X >= 0 and X <= 30 and Y >= -30 and Y <= 0)
  {
Xm = map(X, 0, 30, 180, 270);
  }
  if (X >= 0 and X <= 30 and Y >= 0 and Y <= 30)
  {
Xm = map(X, 30, 0, 270, 360);
```

```
  }
  return Xm;
}
/* Η παρακάτω συνάρτηση κινεί το όχημα περιστροφικά έτσι ώστε να βαθμονομηστεί ο αισθητήρας μας. */
voidCalibration_magnetometer()
{
digitalWrite(12, LOW); // A
digitalWrite(13, LOW); //B
analogWrite(3, 120);  // A
analogWrite(11, 120); // B
delay(2000);
digitalWrite(12, HIGH); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 120);  // A
analogWrite(11, 120); // B
delay(2000);
digitalWrite(12, LOW); // A
digitalWrite(13, HIGH); //B
analogWrite(3, 0);  // A
analogWrite(11, 0); // B
delay(200);
}
```

Βιβλιογραφία

- https://www.u-blox.com/en
- https://www.arduino.cc/
- https://learn.adafruit.com