Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Μεταπτυχιακή Διατριβή
στα Πληροφοριακά Συστήματα

Ανάπτυξη Συστήματος καταγραφής ίχνους (GPS Logger) με χρήση της Πλατφόρμας Υλικού Ανοικτού Κώδικα «Arduino»

Πέτρος Μπίρμπας

Επιβλέπων Καθηγητής
Μιχαήλ Βασιλακόπουλος

Μάιος 2012

Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Ανάπτυξη Συστήματος καταγραφής ίχνους (GPS Logger) με χρήση της Πλατφόρμας Υλικού Ανοικτού Κώδικα «Arduino»

Πέτρος Μπίρμπας

Επιβλέπων Καθηγητής
Μιχαήλ Βασιλακόπουλος

Η παρούσα μεταπτυχιακή διατριβή υποβλήθηκε
προς μερική εκπλήρωση των απαιτήσεων για απόκτηση

μεταπτυχιακού τίτλου σπουδών
στα Πληροφοριακά Συστήματα

από τη Σχολή Θετικών και Εφαρμοσμένων Επιστημών
του Ανοικτού Πανεπιστημίου Κύπρου

Μάιος 2012

Περίληψη

Η εργασία αφορά τη σχεδίαση, συναρμολόγηση υλικού (hardware) και την ανάπτυξη εφαρμογής λογισμικού, για την καταγραφή ίχνους με χρήση gps, πάνω από την πλατφόρμα υλικού ανοικτού κώδικα "arduino". Η συγκεκριμένη πλατφόρμα είναι μια ανοικτή πλατφόρμα υλικού, για την οποία διατίθεται ένα περιβάλλον ανάπτυξης εφαρμογών σε γλώσσα υψηλού επιπέδου, καθώς και πλήθος από στοιχεία υλικού (hardware), όπως αισθητήρες, δικτυακές διεπαφές, κ.ά., τα οποία διατάσσονται κατάλληλα ώστε να επιτελούν μια ή περισσότερες λειτουργίες.

Στο πλαίσιο της εργασίας υλοποιήθηκε η προδιαγραφή, η σχεδίαση και η ανάπτυξη ολοκληρωμένου συστήματος υλικού και λογισμικού (η ανάπτυξη υλικού αφορά σύνδεση των απαραίτητων modules υλικού πάνω στην πλατφόρμα arduino) για την καταγραφή ίχνους κινούμενου οχήματος ή πεζού, με χρήση της τεχνολογίας GPS.

Είναι χαρακτηριστικό ότι μολονότι η συγκεκριμένη πλατφόρμα γίνεται ολοένα και πιο δημοφιλής, η τεκμηρίωση του λογισμικού της συσκευής (π.χ. βιβλιοθήκες που υποστηρίζουν κάποιο στοιχείο υλικού και εφαρμογές που συνδέουν τέτοιες βιβλιοθήκες με εξειδικευμένο λογισμικό) με βάση τους προτεινόμενους από την Τεχνολογία Λογισμικού τρόπους, είναι πολύ περιορισμένη.

Στην εργασία αυτή επιχειρήθηκε μια συνθετική δουλειά που θεματικά εκτείνεται σε μεγάλο εύρος των γνωστικών αντικειμένων του προγράμματος σπουδών συνθέτοντας και αξιοποιώντας γνώσεις που αποκτήθηκαν σε αυτό, για να κατασκευάσει μια νέα συσκευή με το ανάλογο λογισμικό. Η συσκευή που κατασκευάστηκε αποτελεί μια ιδιαίτερα οικονομική εναλλακτική πρόταση σε σύγκριση με υπάρχουσες λύσεις, για συγκεκριμένες εφαρμογές, με σημαντικές δυνατότητες προσαρμογής στο επίπεδο του υλικού και λογισμικού.

Summary

This work focuses on the design, assembly of hardware and the development of application software, for the recording of tracks using gps technology, over the platform of open source hardware “arduino”, which is widely used for prototyping. This particular platform is an open hardware platform, supported by a high-level software development environment, as well as by a wealth of hardware elements such as sensors, network interfaces, etc, which are available in the market and can be deployed in a suitable way to perform one or more desired operations.

In this work the tasks of specification, design, and implementation of an integrated system consisting of arduino-based hardware and software, have been undertaken. The hardware implementation involved high-level design and assembly of readily-available hardware elements around an arduino platform. On top of this, a software application for recording tracks of pedestrians or vehicles on the move using gps technology has been implemented.

Although this particular hardware platform becomes more popular, especially as a prototyping platform, the documentation of application software (eg. libraries that support specific hardware modules and applications required to interconnect such libraries with specialized software modules) is relatively poor and usually not compliant to the Software Engineering principles.

In this context a synthetic work has been undertaken, that is thematically extended to a wide area of this post-grad program, by putting together knowledge acquired in different modules to create a working hardware prototype of a new data logger device, along with the corresponding software. The prototype device that has been developed is a particularly economic alternative compared to existing similar devices, especially considering the wide customization options both in software and hardware.

.

Ευχαριστίες

Καταρχάς θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής μου κ. Μιχάλη Βασιλακόπουλο, πρώτα για την ευκαιρία που μου έδωσε να ασχοληθώ με το πολύ ενδιαφέρον αυτό θέμα, για την καθοδήγηση και την βοήθειά του τόσο στη συγγραφή όσο και στη διόρθωση της κάθε φάσης δημιουργίας της διατριβής.

Πολλές ευχαριστίες οφείλω επίσης στον Επίκουρο καθηγητή κ. Βασίλειο Βεσκούκη για τις πολύτιμες συμβουλές και το ακούραστο ενδιαφέρον και συμβολή του κυρίως για την πραγματοποίηση του πειραματικού μέρους της παρούσας πτυχιακής εργασίας.

Πιο πολύ όμως θα ήθελα να ευχαριστήσω την οικογένεια μου, τα παιδιά μου Στάθη και Αμαλία για το χρόνο μου που τους στέρησα και ιδιαίτερα την σύζυγό μου Μάγδα για την υπομονή, την κατανόηση αλλά και το ενδιαφέρον που έδειξε όλα αυτά τα χρόνια παροτρύνοντας και ενθαρρύνοντας ηθικά όλη αυτή την προσπάθεια.

Πέτρος Ε. Μπίρμπας

Θεσσαλονίκη 7 Μαΐου 2012

Περιεχόμενα

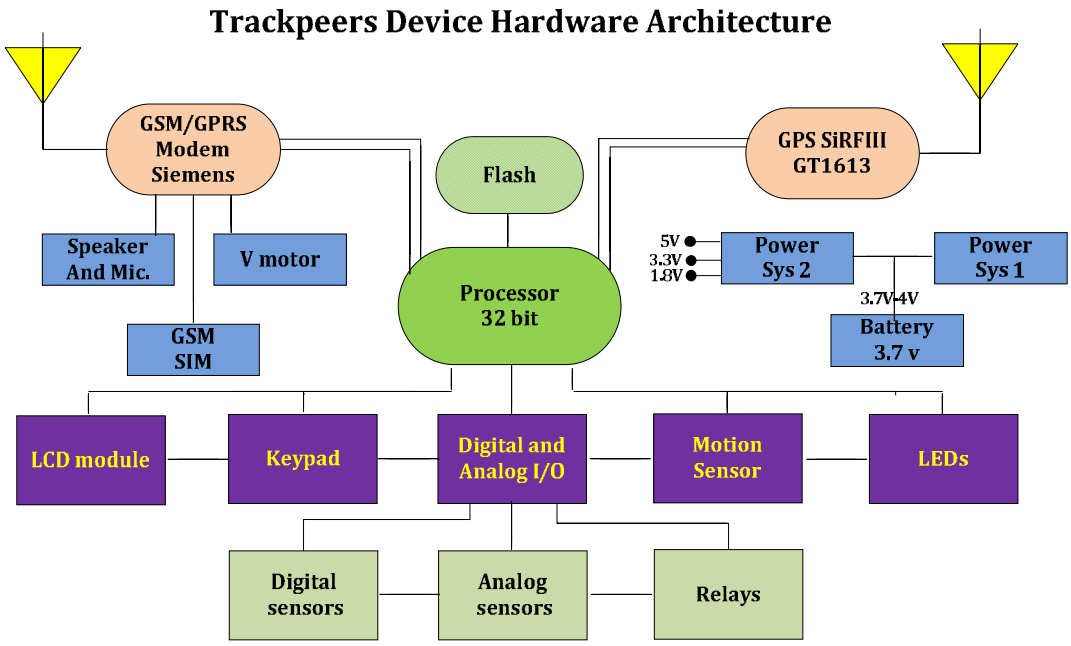
Κεφάλαιο 1
Ψηφιακά συστήματα καταγραφής τροχιάς GPS

1.1 Ψηφιακά Συστήματα Καταγραφής Τροχιάς GPS

Ένα ψηφιακό σύστημα καταγραφής τροχιάς GPS, είναι μία συσκευή που προσδιορίζει και αποθηκεύει την ακριβή γεωγραφική θέση ενός ατόμου, οχήματος, ή οποιουδήποτε αντικειμένου στο οποίο έχει προσαρτηθεί η συσκευή. Τα καταγεγραμμένα δεδομένα είναι δυνατόν να αποθηκευτούν στην ίδια την μονάδα καταγραφής ή να μεταφερθούν σε έναν υπολογιστή με πρόσβαση στο διαδίκτυο, χρησιμοποιώντας κινητό τηλέφωνο, ράδιο ή μόντεμ προσαρτημένο στην συσκευή. Αυτό επιτρέπει, με την χρήση κατάλληλου λογισμικού, την προβολή των δεδομένων σε έναν χάρτη σε πραγματικό χρόνο ή ύστερα από ανάλυση της τροχιάς.

Μία μονάδα καταγραφής τροχιάς με GPS περιλαμβάνει τον μετρητή GPS, ο οποίος λαμβάνει το σήμα και υπολογίζει τις συντεταγμένες. Αν πρόκειται για GPS data logger τότε διατίθεται μεγάλη μνήμη για να αποθηκεύσει τις συντεταγμένες, ενώ αν είναι data pusher περιλαμβάνει επιπλέον

ένα μόντεμ GSM/GPRS για να μεταδώσει τις πληροφορίες στον κεντρικό υπολογιστή σε μορφή πακέτων IP. Το Σχήμα 1.1 απεικονίζει την αρχιτεκτονική μιας πιο σύνθετης συσκευής καταγραφής τροχιάς με GPS [01].



Σχήμα 1.1: Σύνθετος GPS tracker.

1.2 Κατηγορίες συσκευών καταγραφής τροχιάς

Συνήθως ένα ψηφιακό σύστημα καταγραφής τροχιάς με GPS θα ανήκει σε μία από τις παρακάτω κατηγορίες:

Data loggers

Ένας καταγραφέας GPS αυτού του είδους απλά αποθηκεύει τη θέση της συσκευής ανά γνωστά χρονικά διαστήματα που ορίζονται από τον κατασκευαστή της συσκευής. Η αποθήκευση των δεδομένων γίνεται στην εσωτερική μνήμη της συσκευής.

Οι σύγχρονοι GPS data loggers διαθέτουν κάρτα μνήμης ή εσωτερική μνήμη και θύρα USB (Εικόνα 1.1). Η λειτουργία και η μορφή μερικών data loggers μοιάζει με αυτή μιας κάρτας μνήμης USB (Εικόνα 1.2) . Η κάρτα μνήμης επιτρέπει την ανάκτηση των δεδομένων προς ανάλυση σε έναν υπολογιστή. Τα σημεία συντεταγμένων της συσκευής που αποθηκεύονται μπορούν να είναι τύπου GPX, KML, NMEA κλπ.

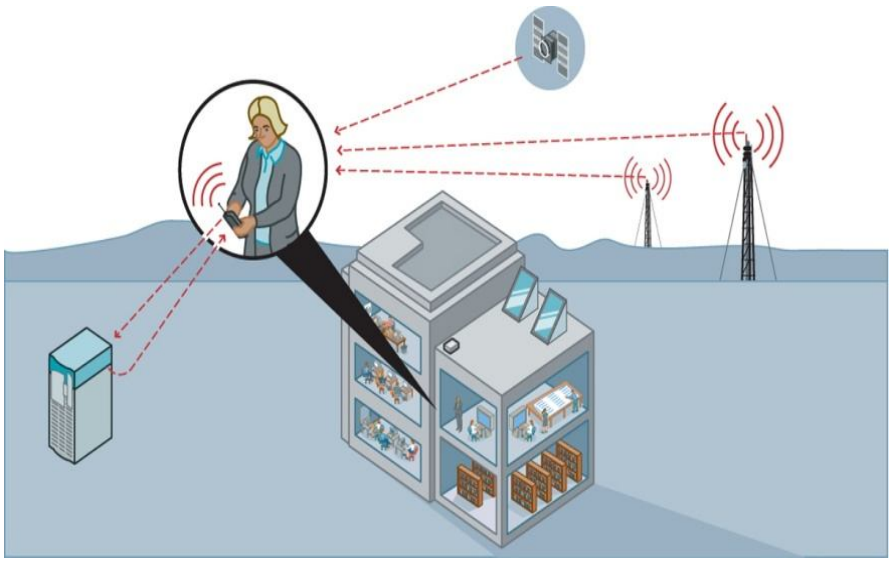
Εικόνα 1.1: GPS data logger.

Εικόνα 1.2: GPS data logger USB.

Data pushers

Οι data pushers είναι η πιο συνηθισμένη κατηγορία συσκευής καταγραφής τροχιάς με GPS. Αυτού του είδους οι συσκευές στέλνουν (push) την θέση καθώς και άλλες πληροφορίες, όπως ταχύτητα και υψόμετρο, σε έναν εξυπηρέτη (server) που μπορεί να αποθηκεύσει και να αναλύσει δεδομένα.

Σε αυτή την κατηγορία ανήκουν συσκευές που αποτελούνται από έναν δέκτη GPS και ένα κινητό τηλέφωνο, συνδεδεμένα μεταξύ τους, χρησιμοποιώντας την ίδια μπαταρία. Ανά χρονικά διαστήματα, το κινητό στέλνει μήνυμα SMS ή GPRS, το οποίο περιέχει τα δεδομένα που έλαβε το GPS (Εικόνα 1.3). Οι τελευταίες τεχνολογίες smartphone διαθέτουν λογισμικό καταγραφής τροχιάς με GPS που μετατρέπουν το κινητό τηλέφωνο σε έναν GPS data logger ή pusher (Εικόνα 1.4). Επιπροσθέτως, από το 2009 άρχισαν να αναπτύσσονται εφαρμογές ανοιχτού κώδικα, προορισμένες για τηλέφωνα που χρησιμοποιούν Java ME, iPhones, Androids, Windows mobiles και Symbian.



Εικόνα 1.3: GPS με κινητό τηλέφωνο.

Εικόνα 1.4: BlackBerry smartphone με λογισμικό για GPS.

Data pullers

Αυτή η κατηγορία ψηφιακών συσκευών καταγραφής τροχιάς με GPS είναι παρόμοια με αυτή των data pushers με τη μόνη διαφορά ότι δεν λαμβάνουν δεδομένα ανά σταθερά χρονικά διαστήματα αλλά συνέχεια. Αυτή η τεχνολογία δεν χρησιμοποιείται τόσο συχνά. Ένα παράδειγμα αυτού του είδους συσκευών είναι ένας υπολογιστής που είναι συνδεδεμένος στο διαδίκτυο και έχει εγκατεστημένο ένα gpsd. Το gpsd αποτελεί εφαρμογή για υπολογιστές με λειτουργικό σύστημα Linux, *BSD και Mac OS X.

Συσκευές με data pullers χρησιμοποιούνται σε περιπτώσεις όπου δεν επιβάλλεται να είναι γνωστή η θέση της συσκευής συνέχεια, αλλά μόνο όταν αυτό είναι επιθυμητό. Ευρεία χρήση data puller γίνεται σε συσκευές που περιλαμβάνεται ένα κινητό τηλέφωνο και δέκτης GPS, οπότε όταν στέλνεται ένα ειδικό μήνυμα SMS, η απάντηση με SMS είναι η θέση της συσκευής.

1.3 Πρότυπα συστημάτων καταγραφής θέσης

1.3.1 GPS (Global Positioning System)

Το Παγκόσμιο Σύστημα Εντοπισμού θέσης (GPS) βασίζεται στο Παγκόσμιο Δορυφορικό Σύστημα Πλοήγησης (GNSS) και παρέχει πληροφορίες για την θέση και τον χρόνο σε οποιεσδήποτε καιρικές συνθήκες, πάνω ή κοντά στη Γη, οπουδήποτε υπάρχει ανοιχτός ορίζοντας για επικοινωνία με 4 ή περισσότερους δορυφόρους GPS [10]. Υποστηρίζεται από την κυβέρνηση των Ηνωμένων Πολιτειών Αμερικής και μπορεί κάποιος να έχει πρόσβαση ελεύθερα, με κάποιους τεχνικούς περιορισμούς που υπάρχουν για στρατιωτικούς σκοπούς, αρκεί να διαθέτει έναν δέκτη GPS.

Το πρόγραμμα GPS αναπτύχθηκε το 1973 για να ξεπεράσει τους περιορισμούς των προηγούμενων συστημάτων πλοήγησης, ολοκληρώνοντας τις ιδέες των προκατόχων του και υλοποιώντας απόρρητες μελέτες μηχανικού σχεδιασμού της δεκαετίας του 1960. Το GPS δημιουργήθηκε από το Υπουργείο Άμυνας των Η.Π.Α. και λειτούργησε αρχικά με 24 δορυφόρους. Μπήκε σε πλήρη λειτουργία το 1994.

Πέρα από το GPS, άλλα συστήματα χρησιμοποιούνται ή είναι υπό κατασκευή. Το Ρωσικό Παγκόσμιο Δορυφορικό Σύστημα Πλοήγησης GLONASS το οποίο χρησιμοποιούσε μόνο ο Ρωσικό στρατός μέχρι το 2007, οπότε έγινε διαθέσιμο και για τους πολίτες. Ετοιμάζονται ακόμα, το Κινέζικο Σύστημα Πλοήγησης (Chinese Compass navigation System) και το Ευρωπαϊκό Galileo.

1.3.2 Λειτουργία GPS

Ένας δέκτης GPS υπολογίζει την θέση του συντονίζοντας με ακρίβεια τα σήματα που στέλνονται από τους δορυφόρους GPS. Κάθε δορυφόρος μεταδίδει συνεχώς μηνύματα που περιλαμβάνουν:

- την ώρα που μεταδόθηκε το μήνυμα
- ακριβείς πληροφορίες για την τροχιά του δορυφόρου (ephimeris = εφημερίς)

- την γενική κατάσταση όλων των δορυφόρων και των τροχιών που ακολουθούν (almanac)

Ο δέκτης χρησιμοποιεί το μήνυμα που λαμβάνει για να προσδιορίσει τον χρόνο που μεταδόθηκε το μήνυμα και υπολογίζει την απόσταση του από κάθε δορυφόρο. Οι αποστάσεις σε συνδυασμό με τις θέσεις των δορυφόρων, χρησιμοποιούνται για τον υπολογισμό της θέσης του δέκτη, με την βοήθεια κατάλληλων αλγορίθμων. Στη συνέχεια, η θέση του δέκτη προβάλλεται σε κάποιον χάρτη ή αριθμητικώς σε συντεταγμένες γεωγραφικούς μήκους και πλάτους. Πολλές μονάδες GPS εμφανίζουν προσανατολισμό και ταχύτητα.

Τρεις δορυφόροι μπορεί να είναι αρκετοί για να γίνει επίλυση, αφού ο χώρος έχει τρεις διαστάσεις. Όμως, ακόμα και ένα ελάχιστο σφάλμα στο ρολόι πολλαπλασιαζόμενο με την ταχύτητα του φωτός – με την οποία μεταδίδονται τα σήματα των δορυφόρων – οδηγεί σε μεγάλο σφάλμα στον προσδιορισμό θέσης. Έτσι, οι δέκτες χρησιμοποιούν τέσσερις ή παραπάνω δορυφόρους για δεδομένη θέση και ώρα του δέκτη. Ο χρόνος μετάδοσης του μηνύματος καθώς και οι υπολογισμοί που ακολουθούν δεν φαίνονται στις εφαρμογές GPS που χρησιμοποιούν μόνο τα αποτελέσματα θέσης του δέκτη.

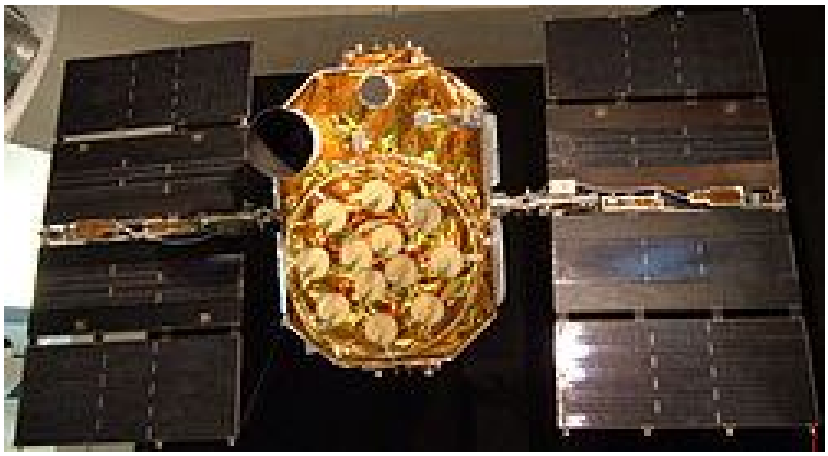
Αν και για συνηθισμένες εργασίες απαιτούνται τουλάχιστον τέσσερις δορυφόροι, σε ειδικές περιπτώσεις μπορεί να χρησιμοποιηθούν και λιγότεροι. Αν μία μεταβλητή είναι ήδη γνωστή, ο δέκτης μπορεί να προσδιορίσει τη θέση του χρησιμοποιώντας μόνο τρεις δορυφόρους. Για παράδειγμα, ένα πλοίο ή ένα αεροσκάφος μπορεί να βρίσκεται σε γνωστό υψόμετρο. Μερικοί δέκτες GPS, όταν λιγότεροι από τέσσερις δορυφόροι είναι ορατοί, μπορεί να χρησιμοποιούν περισσότερα στοιχεία ή να κάνουν υποθέσεις (ξαναχρησιμοποιώντας το τελευταίο γνωστό γεωγραφικό πλάτος, στίγμα κατ' εκτίμηση, αδρανειακή πλοήγηση ή πληροφορίες του υπολογιστή του οχήματος), υπολογίζοντας συντεταγμένες με μικρότερη ακρίβεια.

1.3.3 Δομή GPS

Το GPS αποτελείται από τρία βασικά τμήματα. Αυτά είναι το Τμήμα Διαστήματος (S.S. : Space Segment), το Τμήμα Ελέγχου (C.S. : Control Segment) και το Τμήμα Χρήστη (U.S. : User Segment). Το Τμήμα Διαστήματος και το Τμήμα Ελέγχου αναπτύσσονται, διατηρούνται και λειτουργούν υπό την εποπτεία της αεροπορίας των Ηνωμένων Πολιτειών. Οι δορυφόροι GPS εκπέμπουν σήματα από το διάστημα και κάθε δέκτης GPS χρησιμοποιεί αυτά τα σήματα για να υπολογίσει

τις συντεταγμένες των τριών διαστάσεων (γεωγραφικό μήκος, γεωγραφικό πλάτος και υψόμετρο) και τον χρόνο.

Το Τμήμα Διαστήματος (Εικόνα 1.5) συγκροτείται από 24 έως 32 δορυφόρους στην Μέτρια Γήινη Τροχιά (2000km έως 35786km υψόμετρο πάνω από τη Γη) και περιλαμβάνει επίσης τους ενισχυτές που απαιτούνται για να τους εκτοξεύουν σε τροχιά. Το Τμήμα Ελέγχου (Εικόνα 1.6) αποτελείται από έναν κύριο σταθμό ελέγχου, έναν εναλλακτικό κύριο σταθμό ελέγχου και ένα πλήθος κεραιών εδάφους και σταθμών ελέγχου εκπομπών. Το Τμήμα Χρήστη (Εικόνα 1.7) αποτελείται από τους χιλιάδες στρατιωτικούς, χρήστες της ασφαλούς, περιορισμένης χρήσεως Υπηρεσίας Προσδιορισμού Ακριβούς Θέσης GPS, καθώς και από εκατομμύρια πολιτών, εταιριών και επιστημόνων, χρήστες της Πρότυπης Υπηρεσίας Προσδιορισμού Θέσης GPS.



Εικόνα 1.5: Δορυφόρος GPS (Μουσείο Διαστήματος, San Diego)

Εικόνα 1.6: Σταθμός Ελέγχου (Μουσείου Πυραύλων και Διαστήματος)

Εικόνα 1.7: Δέκτες GPS (Trimble, Garmin, Leica)

1.3.4 Galileo

Το Galileo είναι ένα Παγκόσμιο Δορυφορικό Σύστημα Πλοήγησης που κατασκευάζεται από την Ευρωπαϊκή Ένωση και από την Ευρωπαϊκή Υπηρεσία Διαστήματος. Το κόστος υπολογίζεται στα 20 δισεκατομμύρια Ευρώ και η ονομασία του προέρχεται από τον γνωστό Ιταλό αστρονόμο Γαλιλέο Γαλιλέι [11]. Ένας από τους στόχους του Galileo είναι να παρέχει ένα σύστημα υψηλής ακρίβειας προσδιορισμού θέσης, τέτοιο ώστε οι χώρες της Ευρώπης να είναι εντελώς ανεξάρτητες από το Ρωσικό GLONASS και το Αμερικανικό GPS. Όταν το Galileo θα τεθεί σε

λειτουργία θα χρησιμοποιεί τα δύο κέντρα λειτουργίας στο έδαφος, ένα δίπλα από το Μόναχο στη Γερμανία και το άλλο στο Φουτσίνο στην Ιταλία. Αρχικά προγραμματιζόταν να λειτουργεί το 2012 αλλά η ημερομηνία μεταφέρθηκε αρκετές φορές και δύσκολα θα είναι έτοιμο προς χρήση πριν το 2018.

Το σύστημα πλοήγησης Galileo πρόκειται να παίρνει μετρήσεις υψομέτρου από το Μέσο Ύψος Θάλασσας και να παρέχει καλύτερα αποτελέσματα σε υψηλά γεωγραφικά πλάτη σε σχέση με το GPS και το GLONASS. Επιπλέον, το Galileo θα παρέχει μία λειτουργία Έυρεσης και Διάσωσης (Search And Rescue - SAR). Για αυτό το λόγο, κάθε δορυφόρος θα διαθέτει ένα σύστημα εκπομπής του σήματος σε ένα Βοηθητικό Κέντρο Διάσωσης, όπου θα οργανώνει την διαδικασία διάσωσης όποτε αυτή προκύψει. Ταυτόχρονα, θα εκπέμπεται ένα σήμα στον χρήστη, ενημερώνοντάς τον ότι η κατάστασή του εντοπίστηκε και ότι καταφθάνει κάποιο συνεργείο βοήθειας. Αυτό το τελευταίο χαρακτηριστικό θεωρείται καινούργιο και πρωτοποριακό σε σχέση με τα συστήματα πλοήγησης GLONASS και GPS, τα οποία δεν παρέχουν τέτοιου είδους βοήθεια στον χρήστη. Η χρήση της βασικής (χαμηλής ακρίβειας) υπηρεσίας Galileo θα είναι ελεύθερη και ανοιχτή για όλους. Οι υψηλής ακριβείας ιδιότητες του Galileo θα είναι διαθέσιμες για συνδρομητές και για στρατιωτική χρήση.

1.3.5 Glonass

Το Ρωσικό GLONASS (GLobal NAvigation Satellite) σύστημα εντοπισμού ξεκίνησε το 1976 από την Ρωσία η οποία θέλοντας να αντιπαρατεθεί σε αυτά που έκανε η Αμερική με το GPS, ξεκίνησε ένα δικό της project εντοπισμού θέσης μέσω δορυφόρου που το ονόμασε GLONASS [18]. Ολοκληρώθηκε το 1995. Αναβαθμίστηκε τη δεκαετία του 2000 και σήμερα απορροφά το ένα τρίτο του προϋπολογισμού της Ρωσικής Ομοσπονδίας διαστήματος. Αποτελείται από ένα σύστημα 24 δορυφόρων που περιστρέφονται σε 3 τροχιές με 8 δορυφόρους σε κάθε τροχιά. Καλύπτει όλη την υδρόγειο [19].

Εικόνα 1.8: Ένα μοντέλο του GLONASS-K δορυφόρου έτσι όπως εμφανίστηκε στην έκθεση CeBIT 2011

1.3.6 Compass (Beidu-2)

Το Κινεζικό Compass Navigation System (CP2) ή BEIDOU-2 είναι ένα παγκόσμιο σύστημα πλοήγησης που αποτελείται και βασίζεται σε ένα σύστημα από 35 δορυφόρους στην πλήρη ανάπτυξή του, εκ των οποίων οι 5 είναι γεωστατικής τροχιάς (GEO) για να παρέχουν συμβατότητα με το παλαιότερο κινεζικό σύστημα, το Compass 1 (BEIDOU-1) και οι 30 μέσης τροχιάς (MEO) , με απώτερο σκοπό να καλύψουν όλη την Γη [20].

Η Κίνα λειτούργησε το σύστημα με 5 δορυφόρους κατά τη διάρκεια των Ολυμπιακών αγώνων του Πεκίνου το 2008 σε δοκιμαστική μορφή με πολύ καλά αποτελέσματα από ότι ανακοίνωσε. Την Τρίτη 26 Δεκεμβρίου 2011 επίσης ξεκίνησε να παρέχει υπηρεσίες στους Κινέζους πολίτες σε καθεστώς δοκιμαστικής αξιολόγησης [21]. Αναμένεται να έχει τεθεί σε πλήρη λειτουργία μέχρι το 2020. Θα παρέχει 2 επίπεδα υπηρεσιών: α) Ελεύθερη χρήση προς τους πολίτες. και β) Εξουσιοδοτημένη χρήση για στρατιωτικούς σκοπούς.

Εικόνα 1.9: Χρονική μετεξέλιξη του Compass Navigation System (Beidou)

1.4 Πρότυπα παράστασης δεδομένων θέσης

1.4.1 NMEA 0183

Το NMEA 0183 (ή απλούστερα NMEA) είναι ένα πρωτόκολλο επικοινωνίας μεταξύ στρατιωτικών μηχανημάτων όπως, ανεμόμετρα, αυτόματοι πιλότοι, δέκτες GPS και πολλά άλλα [15]. Έχει οριστεί και ελέγχεται από την Εθνική Στρατιωτική Εταιρία Ηλεκτρονικών των Η.Π.Α.

Το πρότυπο NMEA χρησιμοποιεί ένα απλό πρωτόκολλο σειριακής επικοινωνίας ASCII που καθορίζει τον τρόπο δόμησης των δεδομένων που μεταδίδονται σε μηνύματα μεταξύ ενός «ομιλητή» και πολλών «ακροατών». Με ειδικούς διαστολείς (expanders), ένας ομιλητής μπορεί να έχει μονόδρομη συνομιλία με σχεδόν απεριόριστο αριθμό ακροατών, και χρησιμοποιώντας πολλαπλούς αισθητήρες μπορεί να επικοινωνήσει με μία θύρα υπολογιστή.

Επίσης, στο χώρο των εφαρμογών, το πρότυπο NMEA καθορίζει τα περιεχόμενα κάθε τύπου μηνύματος έτσι ώστε όλοι οι ακροατές να αναλύουν τα μηνύματα ορθά. Εκτενέστερη αναφορά για το πρότυπο NMEA 0183 θα γίνει στο κεφάλαιο 2, παράγραφος 2.4.

1.4.2 GPX (GPS eXchanged format)

Το GPX είναι ένα πρότυπο τύπου XML, σχεδιασμένο σαν ένα απλό πρότυπο δεδομένων GPS για εφαρμογές.

Περιγράφει σημεία προορισμών, τροχιές και διαδρομές. Το πρότυπο χρησιμοποιεί ετικέτες για να αποθηκεύει τοποθεσίες, υψώματα και χρόνο και μπορεί με αυτό τον τρόπο να αλληλεπιδρά με δεδομένα μεταξύ συσκευών GPS και πακέτα εφαρμογών. Τέτοια πακέτα εφαρμογών, για παράδειγμα, επιτρέπουν σε χρήστες να δουν διαδρομές που ακολούθησαν σε δορυφορικές εικόνες (στο Google Earth), να σχολιάσουν σε χάρτες και να προσθέσουν ετικέτες σε φωτογραφίες με γεωαναφορά στο πρότυπο δεδομένων Exif.

1.4.3 KML (Keyhole Markup Language)

Το KML είναι ένα πρότυπο XML που χρησιμοποιείται για την προβολή θέσεων γνωστών γεωγραφικών συντεταγμένων, σε δυσδιάστατους ή τρισδιάστατους χάρτες στο διαδίκτυο. Δημιουργήθηκε από την Keyhole Inc. η οποία ανήκει στην Google από το 2004. Το αρχείο KML, το οποίο ονομαζόταν αρχικά Keyhole Earth Viewer, αναπτύχθηκε από την Google Earth. Το KML πλέον είναι ένα παγκόσμιο πρότυπο του Ανοιχτού Γεωδιαστημικού Συνεταιρισμού (Open Geospatial Consortium). Το Google Earth ήταν το πρώτο πρόγραμμα που πρόβαλλε και επεξεργάστηκε με γραφικό τρόπο τα αρχεία KML. Άλλες εφαρμογές, όπως το Marble, έχουν αρχίσει να δημιουργούνται και να υποστηρίζουν τα πρότυπα KML.

1.5 Σκοπός και αντικείμενο της εργασίας

Σκοπός της εργασίας είναι η επίδειξη της δυνατότητας να αξιοποιήσουμε απλά συστατικά υλικού ανοικτού κώδικα για να κατασκευάσουμε ένα track logger και να διερευνήσουμε τις δυνατότητές του.

Δεδομένου του πλήθους των track loggers αναγκαστικά έγιναν κάποιοι περιορισμοί στην επιλογή των κατάλληλων. Καθοριστικό ρόλο έπαιξε το κόστος .

Λαμβάνοντας υπόψη την φύση του αντικειμένου της εργασίας, η δυνατότητα παρέμβασης επί των track loggers αναγκαστικά επηρέασε την επιλογή του κατάλληλου υλικού και λογισμικού.

Αντικείμενο της εργασίας είναι η επιλογή των υλικών, η συναρμολόγηση-κατασκευή και ο προγραμματισμός του track logger.

Κεφάλαιο 2
Τεχνολογία που χρησιμοποιείται

2.1 Arduino

2.1.1 Γενικές πληροφορίες της πλατφόρμας

Το Arduino [14] είναι μια πρωτότυπη ηλεκτρονική πλατφόρμα διαμόρφωσης ανοικτού λογισμικού βασισμένη στο εύκαμπτο και εύχρηστο υλικό, καθώς και στο λογισμικό. Η χρήση του προορίζεται για καλλιτέχνες, σχεδιαστές, ανθρώπους που ασχολούνται με ηλεκτρονικά στον ελεύθερο χρόνο τους και καθένα ενδιαφερόμενο στη δημιουργία διαλογικών αντικειμένων ή των περιβαλλόντων.

Το Arduino μπορεί να «αισθανθεί» το περιβάλλον προσλαμβάνοντας δεδομένα από μεγάλη ποικιλία αισθητήρων και μπορεί να επηρεάσει το χώρο με τον έλεγχο των φώτων, των μηχανών, και άλλων ηλεκτρονικών επεξεργαστών. Ο μικροεπεξεργαστής στον πίνακα της πλατφόρμας είναι προγραμματισμένος στη χρήση της γλώσσας προγραμματισμού Arduino και στο

περιβάλλον ανάπτυξης Arduino. Τα σχέδια Arduino μπορούν να είναι αυτόνομα ή μπορούν να επικοινωνήσουν με οποιοδήποτε λογισμικό ενός υπολογιστή.

Οι πίνακες της πλατφόρμας μπορούν να δημιουργηθούν χειρονακτικά ή να αγοραστούν προσυναρμολογημένοι. Το λογισμικό λειτουργίας του Arduino προσφέρεται δωρεάν μέσω του διαδικτύου (www.arduino.cc). Τα σχέδια αναφοράς υλικού (αρχεία CAD) είναι διαθέσιμα στο κοινό με άδεια ανοικτού λογισμικού και παρέχεται ελευθερία προσαρμογής για οποιαδήποτε ανάγκη. Το Arduino έλαβε μια τιμητική αναφορά στο ψηφιακό τμήμα Κοινοτήτων των 2006 ARS Electronica Prix.

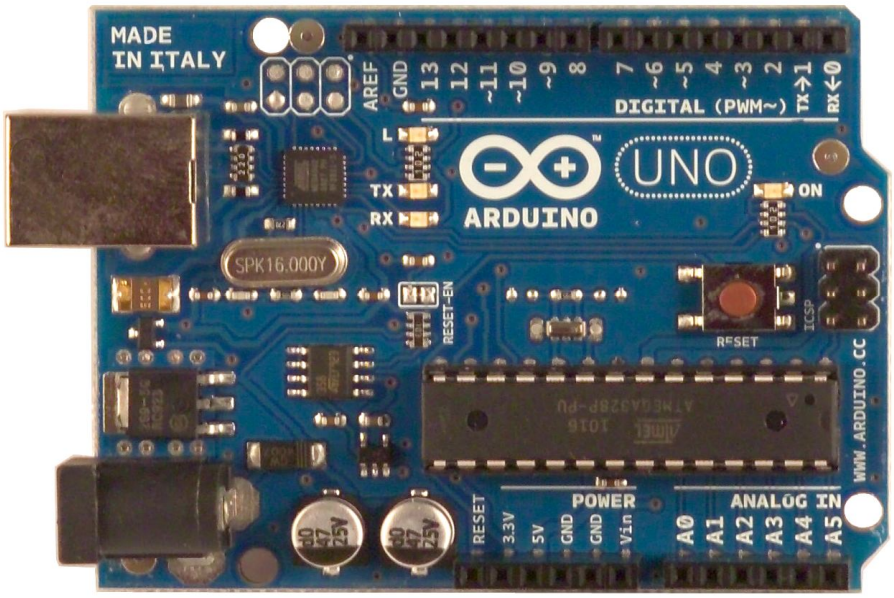
2.1.2. Arduino Uno

Το Arduino Uno (Εικόνα 2.1, 2.2) είναι μία πλατφόρμα βασισμένη στον μικροεπεξεργαστή Atmega328. Έχει 14 ψηφιακές εισόδους και εξόδους (εκ των οποίων οι έξι μπορούν να χρησιμοποιηθούν ως αποτελέσματα PWM), 6 αναλογικές εισόδους, ένα ταλαντωτής κρυστάλλου 16 MHZ, μια σύνδεση USB, μία είσοδο παροχής ενέργειας, ένα διασυνδετικό αγωγό ICSP, και ένα κουμπί αναστοιχειοθέτησης. Περιέχει όλα όσα απαιτούνται για να υποστηρίξουν τη λειτουργία του μικροεπεξεργαστή. Μπορεί να συνδεθεί με έναν υπολογιστή διαμέσου ενός καλωδίου USB ή να τροφοδοτηθεί με έναν προσαρμογέα ρεύμα-συνεχές ρεύμα ή με μία μπαταρία.

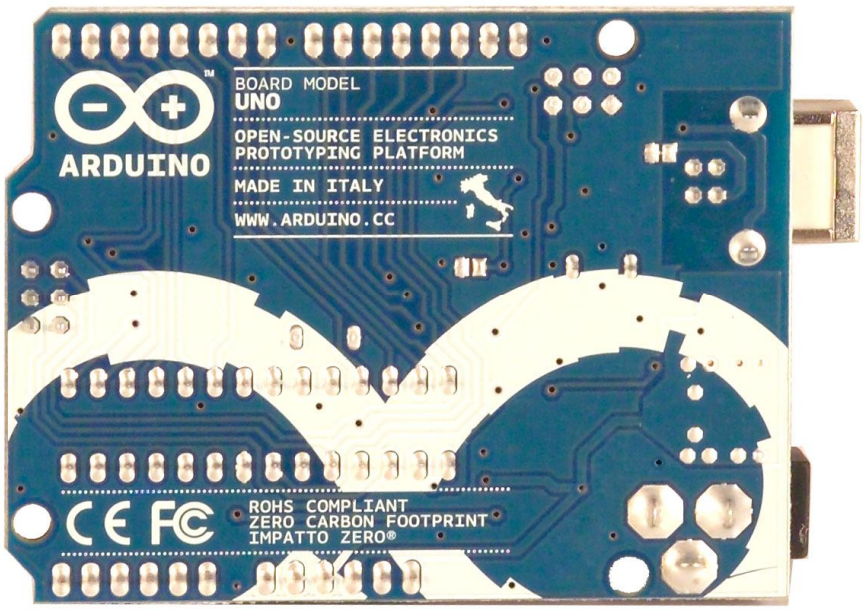
Το Arduino Uno διαφέρει από όλες τις προηγούμενες πλακέτες στο ότι δεν χρησιμοποιεί το τσιπ FTDI USB-to-serial. Αντ' αυτού, χρησιμοποιεί το AtmegaU2 προγραμματισμένο σαν έναν μετατροπέα USB-to-serial. Το Arduino Uno διαθέτει έναν αντιστάτη που γειώνει την γραμμή 8U2 HWB, διευκολύνοντας έτσι την είσοδο σε λειτουργία DFU (Εικόνα 2.3).

«Uno» σημαίνει «Ένα» στα Ιταλικά και ονομάστηκε έτσι προετοιμάζοντας την επερχόμενη νέα έκδοση Arduino 1.0. Το Uno είναι η τελευταία έκδοση σε μία σειρά εκδόσεων Arduino με USB.

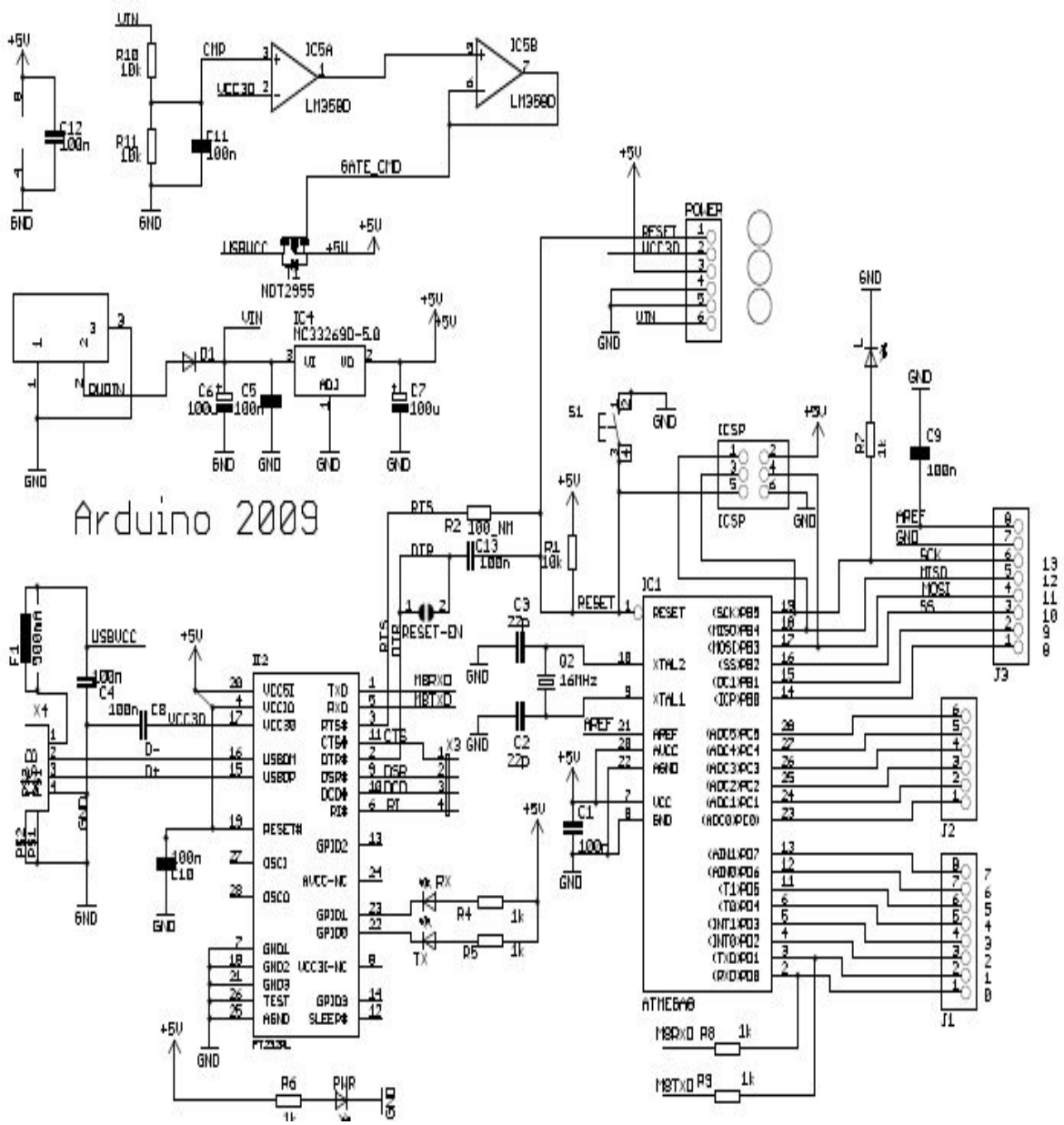
Πιο κάτω φαίνονται στις Εικόνες 2.1 & 2.2 οι κατόψεις της πάνω και κάτω πλευράς της ηλεκτρονικής πλακέτας Arduino Uno.



Εικόνα 2.1 Κάτοψη Arduino Uno (πάνω)



Εικόνα 2.2 Κάτοψη Arduino Uno (κάτω)



Εικόνα 2.3 Συνδεσμολογία Arduino

2.1.3. Φυσικά χαρακτηριστικά

Το μέγιστο μήκος και το πλάτος του Arduino Uno είναι 2.7 και 2.1 ίντσες αντίστοιχα, με το καλώδιο USB και το εξωτερική τροφοδότηση εκτείνεται σε μεγαλύτερη διάσταση. Τέσσερις τρύπες βιδών επιτρέπουν στην πλατφόρμα την σύνδεση με μια επιφάνεια ή μια θήκη. Να σημειωθεί ότι η απόσταση μεταξύ των ψηφιακών εισόδων 7 και 8 είναι 160 mil (0.16 ") , ούτε ένα πολλαπλάσιο 100 mil του διαστήματος των άλλων εισόδων.

2.1.4. Ειδικά χαρακτηριστικά Arduino

Μικροεπεξεργαστής	Atmega328
Λειτουργούσα τάση	5V
Τάση εισαγωγής (που συνιστάται)	7-12V
Τάση εισαγωγής (όρια)	6-20V
Ψηφιακές I/O είσοδοι	14 (εκ των οποίων οι 6 παρέχουν την παραγωγή PWM)
Αναλογικές είσοδοι	6
ΣΥΝΕΧΕΣ ρεύμα ανά I/O είσοδο	40 μA
ΣΥΝΕΧΕΣ ρεύμα για 3.3V είσοδο	50 μA
Μνήμη	32 KB (Atmega328) εκ των οποίων τα 0.5 KB χρησιμοποιούνται από bootloader
SRAM	2 KB (Atmega328)
EEPROM	1 KB (Atmega328)
Ταχύτητα ρολογιών	16 MHZ

Πίνακας 2.1 Φυσικά χαρακτηριστικά της πλατφόρμας

2.1.5. Τροφοδότηση

Το Arduino Uno μπορεί να τροφοδοτηθεί μέσω της σύνδεσης USB ή με μια εξωτερική παροχή ηλεκτρικού ρεύματος. Η πηγή ενέργειας επιλέγεται αυτόματα.

Η εξωτερική παροχή ηλεκτρικού ρεύματος (εκτός της σύνδεσης USB) μπορεί να προέλθει είτε από έναν προσαρμογέα ρεύμα-συνεχές ρεύμα (τοίχος-ακροχόρδωνας) είτε από μπαταρία. Ο προσαρμογέας μπορεί να συνδεθεί στην είσοδο που βρίσκεται πάνω στην πλατφόρμα με διαστάσεις 2.1mm. Κάποια καλώδια από μια μπαταρία μπορούν να συνδεθούν στις Gnd και Vin εισόδους που βρίσκονται στην τροφοδότηση μπαταρίας.

Η πλατφόρμα μπορεί να λειτουργήσει σε μία εξωτερική τροφοδότηση από 6 έως 20 βολτ. Εάν παρέχεται λιγότερο από 7 βολτ, εντούτοις, η είσοδος των 5 βολτ μπορεί να παρέχει λιγότερο από 5 βολτ και η πλατφόρμα μπορεί να είναι ασταθής. Εάν παρέχονται περισσότερα από 12 βολτ, ο ρυθμιστής τάσης μπορεί να υπερθερμάνει και να βλάψει την πλατφόρμα. Το προτεινόμενο φάσμα τροφοδότησης είναι από 7 έως 12 βολτ.

2.1.6 Είσοδοι τροφοδότησης

Οι είσοδοι τροφοδότησης είναι οι ακόλουθες:

- **VIN** Η τάση εισαγωγής στον πίνακα Arduino όταν χρησιμοποιεί μια εξωτερική πηγή ενέργειας (σε αντιδιαστολή με 5 βολτ από τη σύνδεση USB ή άλλη ρυθμισμένη πηγή ενέργειας). Είναι δυνατόν να παρέχεται η τάση μέσω αυτής της εισόδου, ή, εάν παρέχοντας την τάση μέσω του προσαρμογέα εισόδου της μπαταρίας, εξασφαλίζεται πρόσβαση σε αυτή την είσοδο.

- **5V** Η ρυθμισμένη παροχή ηλεκτρικού ρεύματος χρησιμοποιείται για να τροφοδοτήσει το μικροεπεξεργαστή και άλλα συστατικά στον πίνακα. Αυτό μπορεί να προέλθει από VIN δια μέσω ενός ρυθμιστή της πλατφόρμας, είτε να παρασχεθεί από USB είτε από έναν άλλο ρυθμισμένο 5V ανεφοδιασμό.

- **3V3** Ένας ανεφοδιασμός 3.3 βολτ παράγεται στην πλατφόρμα. Το μέγιστο ρεύμα που μπορεί να μεταδώσει είναι 50 μA.

- **GND** Είσοδοι εδάφους.

2.1.7 Προστασία Overcurrent USB

Το Arduino Uno μπορεί να επαναριθμήσει τις πολλές ασφάλειες, ώστε να προστατέψει τις θύρες USB του υπολογιστή από τη χαμηλή είτε υψηλή τάση. Αν και οι περισσότεροι υπολογιστές παρέχουν την δική τους εσωτερική προστασία, η ασφάλεια παρέχει ένα πρόσθετο στρώμα προστασίας. Εάν περισσότερα από 500 μA εφαρμόζονται στη θύρα USB, η ασφάλεια διακόπτει αυτόματα τη σύνδεση έως ότου διορθωθεί το πρόβλημα λόγω υψηλής ή χαμηλής τάσης.

2.1.8 Μνήμη

Ο μικροεπεξεργαστής Atmega 328 έχει 32 KB της μνήμης για την αποθήκευση του κώδικα (εκ του οποίου 0.5 KB χρησιμοποιούνται για bootloader). Επίσης έχει 2 KB SRAM και 1 KB EEPROM (που μπορούν να διαβαστούν και να γραφτούν με Βιβλιοθήκη EEPROM).

2.1.9 Είσοδοι και έξοδοι

Κάθε μια από τις 14 ψηφιακές εισόδους στο Uno μπορεί να χρησιμοποιηθεί ως είσοδος ή έξοδος, χρησιμοποιώντας τις συναρτήσεις pinMode (), digitalWrite (), και digitalRead (). Λειτουργούν σε 5 βολτ. Κάθε είσοδος μπορεί να παρέχει ή να λάβει ένα μέγιστο 40 mA και έχει έναν εσωτερικό pull-up αντιστάτη (αποσυνδεδεμένο εξ ορισμού) των 20-50 kOhms. Επιπλέον, μερικές είσοδοι έχουν ειδικευμένες λειτουργίες:

Σειριακές: 0 (RX) και 1 (TX). Χρησιμοποιούνται για την παραλαβή (RX) και την διαβίβαση (TX) των σειριακών στοιχείων TTL (Transistor-Transistor Logic). Αυτές οι είσοδοι συνδέονται με τις αντίστοιχες εισόδους του τμηματικού τσιπ ATmega8U2 USB-to-TTL.

Εξωτερικοί διακόπτες: 2 και 3. Αυτές οι είσοδοι μπορούν να διαμορφωθούν για να προκαλέσουν διακοπή σε μια χαμηλή τιμή, μια αύξηση ή μια μειωμένη άκρη, ή μια αλλαγή στην τιμή.

PWM: 3, 5, 6, 9, 10, και 11. Παρέχουν οκτάμπιτη παραγωγή PWM μαζί με την συνάρτηση analogWrite ().

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Αυτές οι είσοδοι υποστηρίζουν την επικοινωνία SPI, η οποία χρησιμοποιεί την βιβλιοθήκη SPI.

LED: 13. Υπάρχει ενσωματωμένο led που συνδέεται με την ψηφιακή είσοδο 13. Όταν η τιμή της εισόδου είναι HIGH, το led είναι αναμμένο, ενώ όταν η τιμή της εισόδου είναι LOW, το led δεν είναι αναμμένο.

Το Arduino Uno έχει 6 αναλογικές εισόδους, με ονομασία A0 εως A5, κάθε μια από τις οποίες παρέχει 10 bit της ανάλυσης (δηλ. 1024 διαφορετικές τιμές). Εξ ορισμού μετρούν από το έδαφος έως 5 βολτ, ενώ είναι εν τούτοις πιθανό να αλλαχτεί το ανώτερο όριο της εμβέλειάς τους χρησιμοποιώντας την AREF είσοδο και την συνάρτηση analogReference(). Επιπλέον, μερικές είσοδοι έχουν ειδικές λειτουργίες:

TWI: A4 (SDA) και A5 (SCL). Υποστήριξη TWI επικοινωνίας, χρησιμοποιώντας την Wire library.

2.1.10 Δευτερεύουσες είσοδοι και έξοδοι

Υπάρχουν μερικές άλλες είσοδοι στην πλατφόρμα:

AREF. Αναφορά τάσης για τις αναλογικές εισόδους, χρησιμοποιώντας την συνάρτηση analogReference().

Reset. Η επαναφορά της γραμμής αυτής σε LOW τιμή, επαναρυθμίζει το μικροεπεξεργαστή. Συνήθως χρησιμοποιείται για την πρόσθεση ενός κουμπιού αναστοιχειοθέτησης στις ασπίδες που εμποδίζουν αυτό της πλατφόρμας.

2.1.11 Επικοινωνία

Το Arduino Uno έχει διάφορες εγκαταστάσεις για την επικοινωνία με έναν υπολογιστή, ένα άλλο Arduino, ή άλλους μικροεπεξεργαστές. Οι Atmega328 μικροεπεξεργαστές παρέχουν UART TLL (5V) σειριακή επικοινωνία, η οποία είναι διαθέσιμη στις ψηφιακές εισόδους 0 (RX) και 1 (TX). Ένα τσιπ Atmega8U2 στην πλατφόρμα επιτρέπει την σειριακή επικοινωνία με USB παρέχοντας

μία εικονική είσοδο COM του λογισμικού στον υπολογιστή. Το firmware 8U2 χρησιμοποιεί τους πρότυπους USB COM drivers κι έτσι δεν χρειάζεται κανένας εξωτερικός driver. Ωστόσο, στα Windows, απαιτείται ένα αρχείο .inf. Το λογισμικό Arduino περιλαμβάνει ένα σειριακό όργανο ελέγχου, που επιτρέπει στα απλά δεδομένα κειμένου να αποσταλούν προς αλλά και από την πλατφόρμα Arduino. Τα RX και TX LEDs στην πλατφόρμα θα αναβοσβήσουν όταν διαβιβάζεται το στοιχείο μέσω του τσιπ USB-to-serial και της σύνδεσης USB στον υπολογιστή (αλλά όχι για την σειριακή επικοινωνία των εισόδων 0 και 1). Η Software Serial βιβλιοθήκη επιτρέπει την σειριακή επικοινωνία σχετικά με οποιαδήποτε ψηφιακή είσοδο του Arduino Uno.

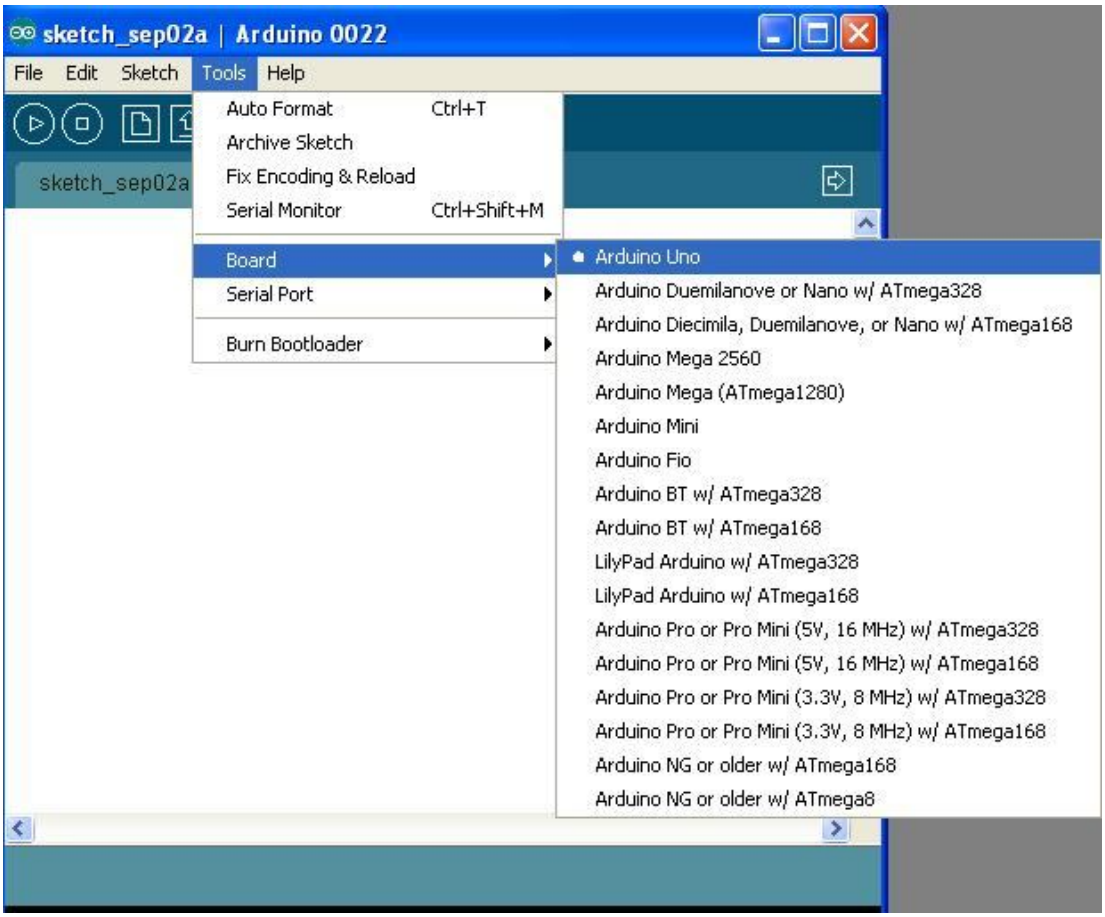
Ο ATmega328 μικροεπεξεργαστής επίσης υποστηρίζει την I2C (TWI) και SPI επικοινωνία. Το λογισμικό Arduino περιλαμβάνει μια βιβλιοθήκη Wire για να απλοποιήσει τη χρήση της I2C επικοινωνίας. Για να χρησιμοποιηθεί η SPI επικοινωνία, χρειάζεται η βιβλιοθήκη SPI.

2.1.12 Προγραμματισμός

Το Arduino Uno μπορεί να προγραμματιστεί με το λογισμικό Arduino. Στο λογισμικό Arduino (http://arduino.cc/en/Main/Software) επιλέγετε στο μενού Tools > Board > Arduino Uno (Εικόνα 2.4). Ο μικροεπεξεργαστής ATmega328 στο Arduino Uno παρέχεται με bootloader, ο οποίος επιτρέπει την μεταφόρτωση νέου κώδικα σε αυτό χωρίς τη χρήση ενός εξωτερικού προγραμματιστή υλικού. Επικοινωνεί με τη χρήση τον αρχικού STK500 πρωτοκόλλου.

Μπορεί, επίσης, να παρακαμφθεί το bootloader και να προγραμματιστεί ο μικροεπεξεργαστής μέσω της επιγραφής ICSP (προγραμματισμός σε σειριακό κύκλωμα).

Ο πηγαίος κώδικας της Atmega8U2 firmware είναι διαθέσιμος. Ο Atmega8U2 φορτώνεται με έναν DFU bootloader, ο οποίος μπορεί να ενεργοποιηθεί συνδέοντας το jumper στο πίσω μέρος της πλακέτας και ύστερα κάνοντας επανεκκίνηση στον 8U2 . Ύστερα, για να φορτωθεί το καινούργιο firmware, χρησιμοποιείται το λογισμικό FLIP της Atmel (Windows) ή το DFU programmer (Mac OS X και Linux). Ή, πιο απλά, χρησιμοποιούμε τον διασυνδετικό αγωγό ICSP με έναν εξωτερικό προγραμματιστή, επαναπρογραμματίζοντας τον DFU bootloader.



Εικόνα 2.4 Ορίζοντας τον τύπο της πλατφόρμας

2.1.13 Αυτόματη επαναρύθμιση (reset από το λογισμικό)

Το Arduino Uno έχει σχεδιαστεί με τέτοιο τρόπο ώστε να επιτρέπεται η επαναρύθμιση από το λογισμικό που τρέχει σε έναν συνδεδεμένο υπολογιστή, παρά από το πάτημα του κουμπιού πριν από κάθε μεταφόρτωση προγράμματος. Μια από τις γραμμές ελέγχου ροής του υλικού (DTR) του FT232RL συνδέεται με τη γραμμή αναστοιχειοθέτησης ATmega328 διαμέσου ενός πυκνωτή των 100 nanofarad. Όταν αυτή η γραμμή παίρνει την τιμή LOW, η γραμμή αναστοιχειοθέτησης μειώνεται αρκετά ώστε να επαναρυθμήσει το τσιπ. Το λογισμικό Arduino επιτρέπει την μεταφόρτωση του κώδικα με το πάτημα ενός κουμπιού στο περιβάλλον του Arduino. Αυτό σημαίνει ότι το bootloader μπορεί να έχει ένα πιο σύντομο διάλειμμα, καθώς την στιγμή που χαμηλώνει το DTR μπορεί να συντονιστεί καλά η έναρξη της μεταφόρτωσης.

Αυτή η οργάνωση έχει κι άλλες επιπτώσεις. Όταν το Uno είναι συνδεδεμένο με έναν υπολογιστή που τρέχει MAC OS X ή Linux, επαναρυθμίζεται κάθε φορά που γίνεται μια σύνδεση από το

λογισμικό (μέσω USB). Για το επόμενο μισό-δευτερόλεπτο περίπου, το bootloader τρέχει στο Uno. Ενώ προγραμματίζεται για να αγνοήσει τα δύσμορφα στοιχεία (δηλαδή οτιδήποτε εκτός από τη μεταφόρτωση του νέου κώδικα), θα παρεμποδίσει τα πρώτα bytes των δεδομένων που στέλνονται στην πλατφόρμα μετά την πραγματοποίηση της σύνδεσης. Εάν ένας τρέχον κώδικας στην πλατφόρμα λαμβάνει μία επιβεβαίωση ή άλλα δεδομένα όταν ξεκινά για πρώτη φορά, πρέπει να είναι σίγουρο ότι το λογισμικό με το οποίο επικοινωνεί, περιμένει ένα δευτερόλεπτο μετά από την διάνοιξη της σύνδεσης και πριν την αποστολή των δεδομένων αυτών.

Το Arduino Uno περιέχει ένα ίχνος που μπορεί να κοπεί για να θέσει εκτός λειτουργίας την αυτόματη αναστοιχειοθέτηση / επαναρύθμιση. Τα μαξιλάρια από κάθε πλευρά του ίχνους μπορούν να συγκολληθούν μαζί για να το επιτρέψουν πάλι. Το ίχνος αυτό ονομάζεται "RESET-EN". Μπορεί επίσης να τεθεί εκτός λειτουργίας η αυτόματη επαναρύθμιση, με τη σύνδεση ενός αντιστάτη των 110 ohm από 5 Volt με τη γραμμή επαναρύθμισης.

2.1.14 Αναφορά στο μοντέλο προγραμματισμού του Arduino, τις ενσωματωμένες βιβλιοθήκες, τις βιβλιοθήκες που συνοδεύουν κάθε πρόσθετη συσκευή.

Το προγραμματιστικό περιβάλλον που χρησιμοποιείται συνήθως στα παραδείγματα της πλατφόρμας ονομάζεται «επεξεργασία» ή Processing. Είναι βασισμένο στην προγραμματιστική γλώσσα Java και έχει σχεδιαστεί για τη χρήση από σχεδιαστές, καλλιτέχνες και οποιονδήποτε άλλο που δεν χρειάζεται να γνωρίζει όλες τις λεπτομέρειες του προγραμματισμού, αλλά επιθυμούν τη δημιουργία κάποιου «έργου». Είναι χρήσιμο εργαλείο για την πραγματοποίηση ιδεών προγραμματισμού, επειδή απαιτείται σχετικά μικρός κώδικας επεξεργασίας για αρκετά σημαντικά πράγματα, όπως η δημιουργία σύνδεσης δικτύων, η σύνδεση μιας εξωτερικής συσκευής μέσω μίας σειριακής θύρας ή ο έλεγχος ψηφιακής κάμερας μέσω Fire Wire. Είναι ελεύθερο και ανοιχτό προς το κοινό, διαθέσιμο μέσα από το διαδίκτυο. Επειδή βασίζεται στην προγραμματιστική γλώσσα Java, μπορεί να χρησιμοποιηθεί σε οποιοδήποτε λειτουργικό σύστημα όπως σε MAC OS X, Windows και Linux. Σε περίπτωση που η διαδικασία «επεξεργασίας» (Processing) δεν είναι επιθυμητή, ο χρήστης οφείλει να είναι σε θέση να χρησιμοποιήσει δείγματα του κώδικα και τα κατάλληλα σχόλια, με τη χρήση ψευδογλώσσας για οποιοδήποτε περιβάλλον πολυμέσων είναι επιθυμητό.

Η πλατφόρμα (Εικόνα 2.5) διαθέτει οκτώ κουμπιά επιλογών, τα οποία είναι : βεβαίωση / σύνταξη, διακοπή, νέο, άνοιγμα, αποθήκευση, μεταφόρτωση στην I/O πλατφόρμα, σειριακή οθόνη και αναγνωριστικός κατάλογος. Το πρώτο ελέγχει αν ο κώδικας του χρήστη περιέχει λάθη και τα επισημαίνει, ενώ το δεύτερο διακόπτει τη σειριακή επικοινωνία. Το τρίτο δημιουργεί νέα καρτέλα για την καταγραφή κώδικα, το τέταρτο παρουσιάζει όλους τους κώδικες που έχουν δημιουργηθεί και το πέμπτο αποθηκεύει τον κώδικα. Το έκτο μεταβιβάζει τον κώδικα στην I/O πλατφόρμα, αφού πρώτα έχει γίνει αποθήκευση και έλεγχος της σωστής σύνταξης του κώδικα. Το έβδομο κουμπί εμφανίζει τα σειριακά δεδομένα στην οθόνη, τα οποία αποστέλλονται από την πλατφόρμα. Πρέπει να γίνει επιλογή της μεταβλητής ταχύτητας μεταβίβασης των δεδομένων του ηλεκτρονικού υπολογιστή ή αλλιώς των τηλεγραφικών σημάτων, σύμφωνα με τον κώδικα που έχει συνταχθεί. Το τελευταίο κουμπί επιτρέπει τη διαχείριση κωδικών με περισσότερα από ένα αρχεία. Τα αρχεία αυτά μπορεί να είναι αρχεία της προγραμματιστική γλώσσα Arduino, της C, της C++ ή των διασυνδετικών προσαγωγών.

Υπάρχει η δυνατότητα χρήσης προϋπαρχόντων βιβλιοθηκών σε οποιονδήποτε κώδικα με τη χρήση της επιλογής “Import library” ή χειρονακτικά προσθέτοντας στον κώδικα την εντολή #include(όνομα βιβλιοθήκης). Παρέχονται κάποιες βιβλιοθήκες με την εγκατάσταση του προγράμματος οδήγησης της πλατφόρμας και αυτές είναι:

EEPROM – ανάγνωση και εγγραφή σε μόνιμη αποθήκευση (εγγραφή στη μνήμη της πλατφόρμας). Ακολουθούν κάποιες από τις βιβλιοθήκες :

- Ethernet – εφαρμόζεται για τη σύνδεση με το Διαδίκτυο χρησιμοποιώντας το Arduino Ethernet Shield.

- Firmata – χρησιμοποιείται την επικοινωνία με τις εφαρμογές του υπολογιστή που χρησιμοποιείται ένα τυποποιημένο τμηματικό πρωτόκολλο.

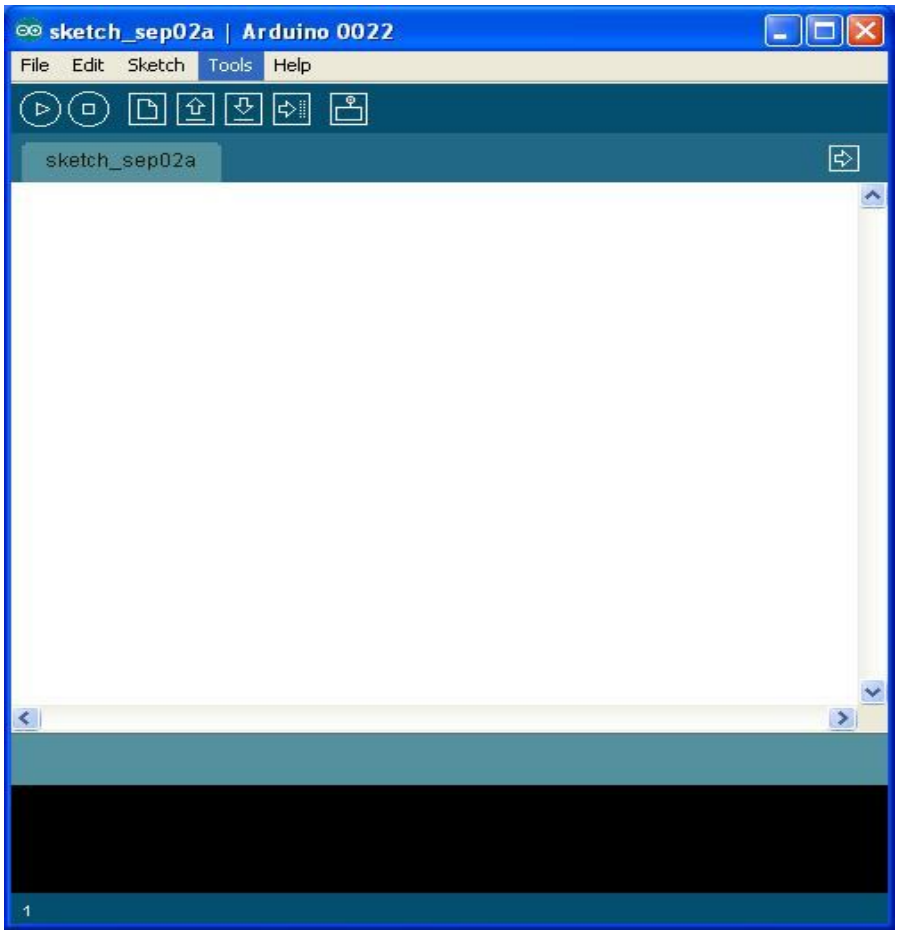
- Liquid Crystal – εφαρμόζεται για τον έλεγχο των επιδείξεων υγρού κρυστάλλου (LCDs).

- Servo – εφαρμόζεται για τον έλεγχο των servo μηχανών.

- Software Serial – εφαρμόζεται για την τμηματική ανακοίνωση σχετικά με οποιασδήποτε ψηφιακές εισόδους.

- Stepper – εφαρμόζεται για τον έλεγχο των stepper μηχανών.

- Wire – η διεπαφή δύο καλωδίων (TWI/I2C) έχει την δυνατότητα αποστολής και λήψης των δεδομένων εκτός των συσκευών και των αισθητήρων.



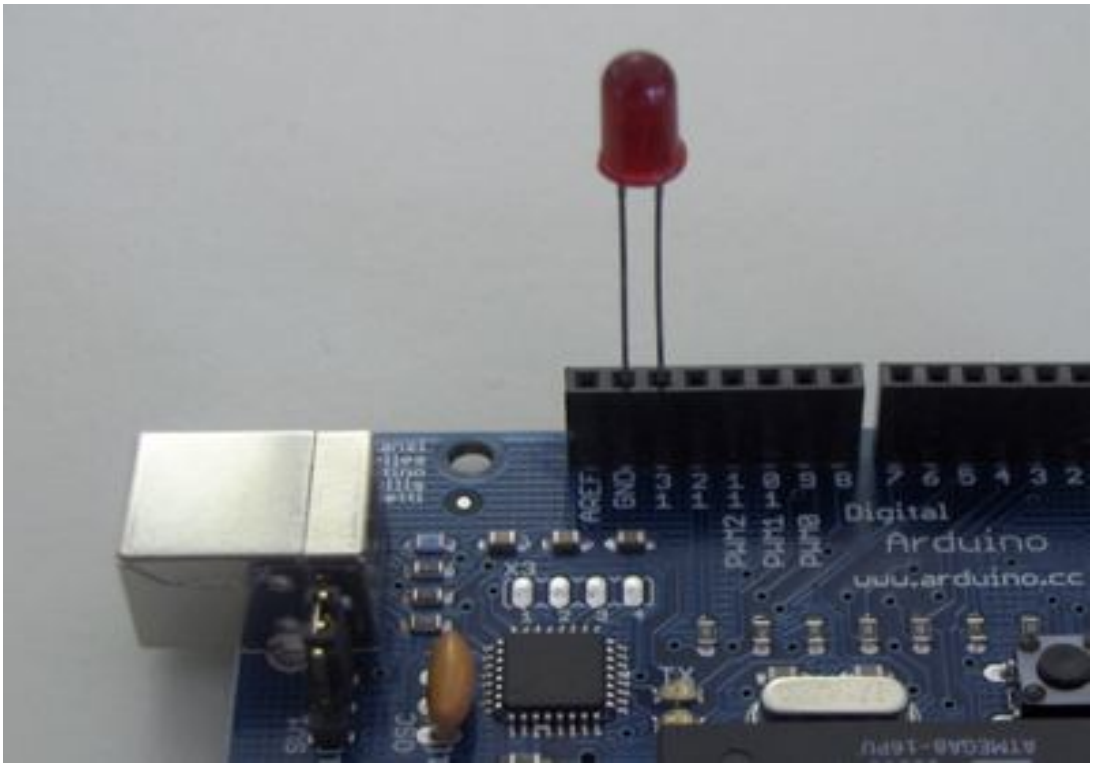
Εικόνα 2.5 Προγραμματιστικό περιβάλλον Arduino

2.1.15 Παράδειγμα προγραμματισμού 1 (εφαρμογή blink)

Στις περισσότερες γλώσσες προγραμματισμού, το πρώτο παράδειγμα προγράμματος που χρησιμοποιείται είναι αυτό που εμφανίζει στην οθόνη το μήνυμα «hello world». Δεδομένου ότι η πλατφόρμα Arduino δε διαθέτει οθόνη, το πρώτο πρόγραμμα είναι αυτό που μας βοηθά να αναβοσβήνουμε ένα μικρό λαμπάκι. Οι πλατφόρμες σχεδιάζονται έτσι ώστε να καθιστάται εύκολο να αναβοσβήνουν το λαμπάκι, χρησιμοποιώντας την ψηφιακή είσοδο 13. Αξίζει να

σημειωθεί, ότι κάποια μοντέλα Arduino, όπως και το Uno, διαθέτουν ενσωματωμένο λαμπάκι. Στα μοντέλα, τα οποία δεν διαθέτουν ενσωματωμένο λαμπάκι, υπάρχει ένας αντιστάτης 1 KB στην είσοδο, επιτρέποντας τη άμεση σύνδεση με ένα λαμπάκι. (Αν χρειάζεται η σύνδεση με ένα λαμπάκι και με άλλη ψηφιακή είσοδο, πρέπει να χρησιμοποιηθεί εξωτερικός αντιστάτης).

Τα συγκεκριμένα λαμπάκια έχουν πολικότητα, γεγονός που σημαίνει μόνο αν γίνει σωστή σύνδεση και με σωστό προσανατολισμό των καλωδίων τους. Το 'κοντό' καλώδιο, συνδέεται με την ψηφιακή GND είσοδο. Ο βολβός, που ανήκει στο λαμπάκι, τυπικά διαθέτει μία επίπεδη άκρη στην συγκεκριμένη πλευρά. Εάν το λαμπάκι δεν αναβοσβήσει, πρέπει να γίνει αντιστροφή των καλωδίων (δε θα καταστραφούν τα λαμπάκια εάν γίνει αντίστροφη συνδεσμολογία για μικρή χρονική περίοδο).



Εικόνα 2.6 Σύνδεση led με την πλατφόρμα

Το παράδειγμα του κώδικα είναι πολύ απλό και περισσότερες λεπτομέρειες παρατίθενται στον κώδικα σαν σχόλια.

```
******* ΚΩΔΙΚΑΣ *******
```

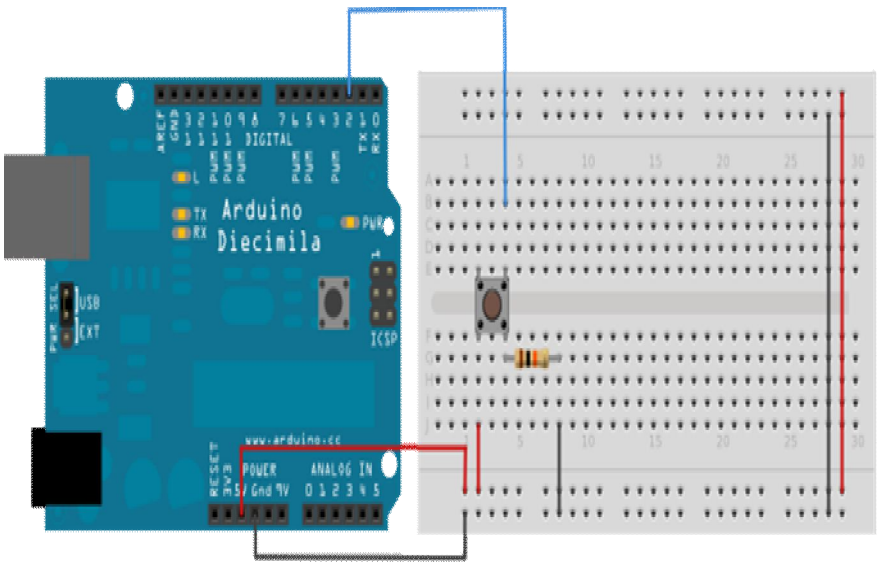
```
/* Blinking LED
 * ------------
 *
 * turns on and off a light emitting diode(LED) connected to a digital
 * pin, in intervals of 2 seconds. Ideally we use pin 13 on the Arduino
 * board because it has a resistor attached to it, needing only an LED
 *
 * Created 1 June 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 * based on an orginal by H. Barragan for the Wiring i/o board
 */
int ledPin = 13;                 // LED connected to digital pin 13
void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}
void loop()
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(1000);                  // waits for a second
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(1000);                  // waits for a second
}
```

2.1.16 Παράδειγμα προγραμματισμού 2 (εφαρμογή Button)

Τα κουμπιά ή διακόπτες συνδέουν δύο σημεία ενός κυκλώματος όταν τα πατάς. Σε αυτό το παράδειγμα όταν πατηθεί το κουμπί ανάβει το λαμπάκι LED που έχει συνδεθεί στο pin 13 της πλατφόρμας.

Υλικά:

- ✓ Arduino
- ✓ κουμπί
- ✓ αντιστάτης 10Kohm
- ✓ breadboard
- ✓ καλώδια

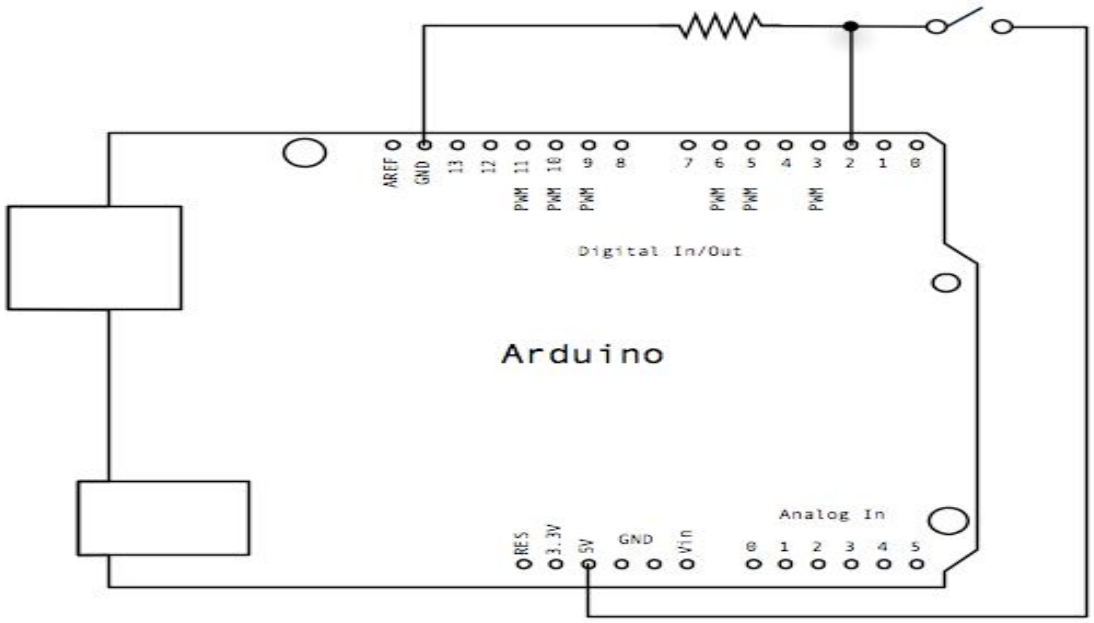


Εικόνα 2.7 Κύκλωμα σύνδεσης Arduino με breadboard και Button

Συνδέουμε τρία καλώδια στην πλατφόρμα Arduino. Τα πρώτα δύο, το κόκκινο και το μαύρο, τα συνδέουμε στις δύο κατακόρυφες στήλες του breadboard για να υπάρχει πρόσβαση στην παροχή 5 Volt και την γείωση. Το τρίτο καλώδιο συνδέεται από την ψηφιακή είσοδο 2 στο ένα πόδι του κουμπιού. Το ίδιο ακριβώς απέναντι πόδι του κουμπιού συνδέεται με έναν αντιστάτη 10Kohms στην γείωση. Το άλλο πόδι συνδέεται στην παροχή ρεύματος 5 Volt του Arduino.

Όταν το κύκλωμα του κουμπιού είναι ανοιχτό ,δηλαδή δεν είναι πατημένο, δεν υπάρχει σύνδεση μεταξύ των δύο ποδιών του κουμπιού και έτσι το pin είναι γειωμένο και παίρνει την τιμή LOW. Όταν το κουμπί πατιέται, το κύκλωμα κλείνει συνδέοντας έτσι το pin στα 5 Volt και παίρνει την τιμή HIGH, ανάβοντας το LED.

Αν αποσυνδέσουμε το pin ψηφιακής εισόδου/εξόδου, το φωτάκι LED μπορεί να αναβοσβήνει άτακτα. Αυτό συμβαίνει επειδή, θα αλλάζει τιμές τυχαία μόνο του, είτε σε HIGH είτε σε LOW. Γι' αυτό το λόγο χρησιμοποιείται ο αντιστάτης στο κύκλωμα.



Εικόνα 2.8 Ηλεκτρονικό σχεδιάγραμμα Συνδεσμολογίας Arduino με χρήση αντιστάτη

******* ΚΩΔΙΚΑΣ *******

```
/*
  Button

 Turns on and off a light emitting diode(LED) connected to digital
 pin 13, when pressing a pushbutton attached to pin 2.


 The circuit:
 * LED attached from pin 13 to ground
 * pushbutton attached to pin 2 from +5V
 * 10K resistor attached to pin 2 from ground

 * Note: on most Arduinos there is already an LED on the board
 attached to pin 13.


 created 2005
 by DojoDave <http://www.0j0.org>
 modified 28 Oct 2010
 by Tom Igoe

 This example code is in the public domain.

 http://www.arduino.cc/en/Tutorial/Button
 */

// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
```

```
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

2.2 Parallax GPS Receiver Module

Ο δέκτης GPS της εταιρίας Parallax (Εικόνα 2.9) [16] είναι μία ολοκληρωμένη, χαμηλού κόστους μονάδα GPS με προσαρτημένη κεραία. Ο δέκτης παρέχει πρότυπα δεδομένα με βάση το πρωτόκολλο NMEA0183 ή συγκεκριμένα δεδομένα που ζητάει ο χρήστης μέσω της σειριακής μονάδας εντολών. Ο δέκτης μπορεί να συγχρονιστεί με μέχρι 12 δορυφόρους ενώ ταυτόχρονα παρέχει υποστήριξη από το WAAS/EGNOS (Wide Area Augmentation System/European Geostationary Navigation Overlay Service) για πιο ακριβή αποτελέσματα.

Τα αποτελέσματα που δίνει ο δέκτης είναι η ώρα, ημερομηνία, γεωγραφικό πλάτος, γεωγραφικό μήκος, υψόμετρο, ταχύτητα και προσανατολισμός / κατεύθυνση πλοήγησης. Μπορεί να χρησιμοποιηθεί για επαγγελματικές εφαρμογές πλοήγησης, τροχιακών συστημάτων, χαρτογράφησης, αυτόματου πιλότου και ρομποτικής.

Βασικά στοιχεία του δέκτη:

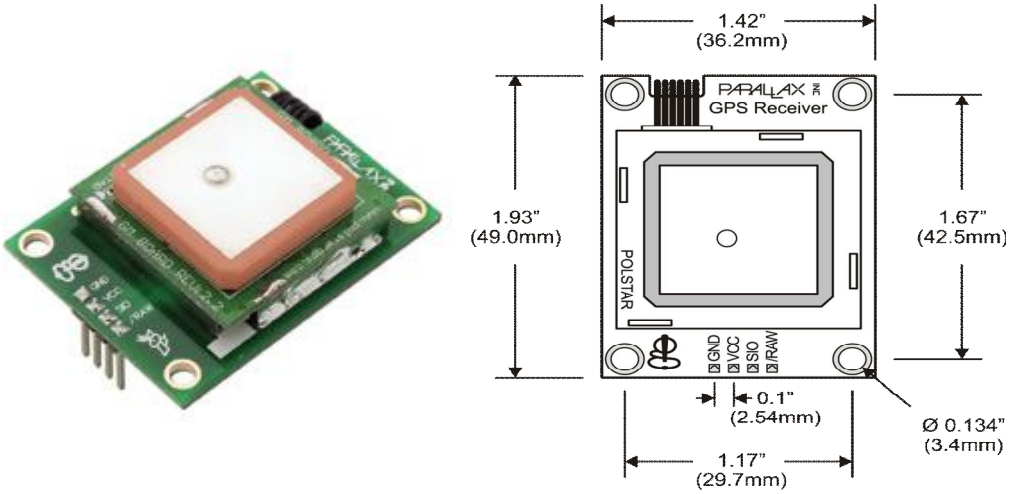
- ταχύτητα σειριακής επικοινωνίας 4800 baud (σύμβολα ανά δευτερόλεπτο), διασύνδεση TTL (Transistor-Transistor Logic), πολυπύρηνους παράλληλους μικροελεγκτές τύπου Propeller

- αλφαριθμητικά NMEA0183 ή συγκεκριμένα δεδομένα που ζητά ο χρήστης μέσω της διασύνδεσης εντολών

- χρειάζεται παροχή ρεύματος τάσης +5VDC έντασης 115mA

- 0.100’’ απόσταση μεταξύ των ακίδων (pins)

- προγραμματιζόμενο μικροελεγκτή Parallax SX/B και ανοιχτό λογισμικό για προχωρημένους χρήστες (δεν υποστηρίζεται από την Parallax αλλά υπάρχει στην σελίδα της εταιρίας)



Εικόνα 2.9 Parallax GPS Receiver Module

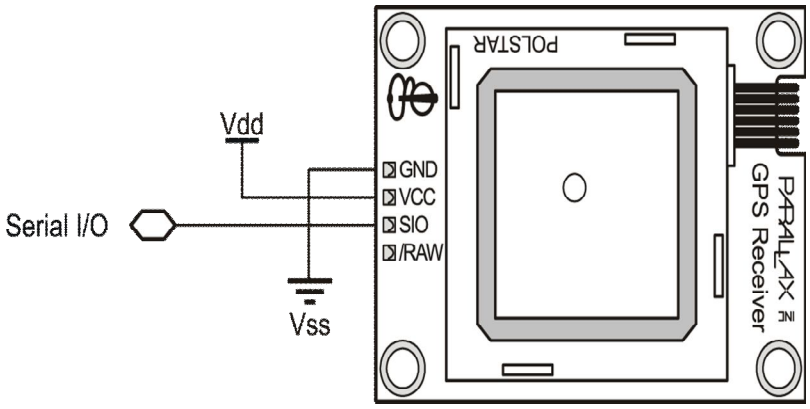
2.2.1 Ηλεκτρονική συνδεσμολογία αισθητήρα

Pin	Ονομασία Pin	Τύπος	Λειτουργία
1	GND	G	Γείωση του συστήματος. Συνδέεται στην γείωση του παροχέα ρεύματος της συσκευής
2	VCC	P	Τροφοδοσία +5VDC
3	SIO	I/O	Σειριακή επικοινωνία (εντολές που στέλνονται ΠΡΟΣ τον δέκτη και δεδομένα που λαμβάνονται ΑΠΟ τον δέκτη). Ασύγχρονη, διασύνδεση επιπέδου TTL, 4800bps (bits per second), 8 bits δεδομένων, no parity, 1 stop bit, μη-αναστρέψιμη.
4	/RAW	I	Pin επιλογής τρόπου λειτουργίας. Προεπιλεγμένη κατάσταση : HIGH. Όταν το /RAW pin δεν είναι συνδεδεμένο, η προεπιλεγμένη κατάσταση λειτουργίας

			«Smart mode» ενεργοποιείται, με ειδικές εντολές ζητούνται τα κατάλληλα δεδομένα GPS και τα αποτελέσματα επιστρέφονται στον δέκτη. Όταν το /RAW pin ενεργοποιείται (LOW), ο δέκτης εισέρχεται σε κατάσταση λειτουργίας «Raw Mode» και μεταδίδει συγκεκριμένα αλφαριθμητικά, επιτρέποντας στον χρήστη να χρησιμοποιήσει άμεσα τα δεδομένα του GPS.

Πίνακας 2.2 Περιγραφή λειτουργιών αισθητήρα (Σημείωση : Τύπος : I = Input, O = Output, P = Power, G = Ground)

Ο δέκτης GPS μπορεί να αλληλεπιδράσει με οποιαδήποτε συσκευή, αρκεί να συνδεθούν, τουλάχιστον, τα πρώτα τρία pins. Η παρακάτω συνδεσμολογία (Εικόνα 2.2.1) χρησιμοποιεί την BASIC γλώσσα μικροελεγκτή αφού το /RAW pin είναι HIGH, δηλαδή δεν έχει συνδεθεί καθόλου και λειτουργεί σε «Smart Mode».



Εικόνα 2.10 Συνδεσμολογία για BASIC

Ο δέκτης είναι κατασκευασμένος ώστε να στέκεται οριζόντιος και η κεραία να είναι στραμμένη προς τον ουρανό. Στις γωνίες του δέκτη υπάρχουν τρύπες για βίδες, αν ο χρήστης επιθυμεί πιο σταθερό στήσιμο.

Ο δέκτης GPS πρέπει να βρίσκεται σε εξωτερικό υπαίθριο χώρο και να «βλέπει» καθαρό ουρανό για να συνδεθεί με τους δορυφόρους. Εφόσον υπάρχει καθαρός ουρανός, το GPS λειτουργεί οπουδήποτε, 24 ώρες τη μέρα, 7 μέρες τη βδομάδα. Αξίζει να σημειωθεί ότι κάποια προϊόντα,

όπως υπολογιστές και ασύρματες/RF συσκευές που εκπέμπουν μαγνητικά πεδία και ηχοπαράσιτα, μπορεί να εμποδίζουν τον δέκτη να λαμβάνει τα σήματα από τους δορυφόρους GPS μειώνοντας η απόδοση του. Επιπροσθέτως, όταν η εφαρμογή χρησιμοποιείται σε κάποιο αυτοκινούμενο όχημα, τότε ο δέκτης πρέπει να βρίσκεται στην οροφή του οχήματος. Αν ο δέκτης βρίσκεται μέσα στο αυτοκίνητο, η κεραία πρέπει να έχει καθαρή θέα προς τον ουρανό και να μην εμποδίζεται το σήμα της από μεταλλικά αντικείμενα μέσα στο όχημα.

2.2.2 Ηλεκτρολογικά χαρακτηριστικά αισθητήρα

Κατάσταση	Τιμή
Θερμοκρασία λειτουργίας	-40°C εως +85°C
Θερμοκρασία αποθήκευσης	-55°C εως +100°C
Ρεύμα παροχής (V_{cc})	+4.5V εως +5.5V
Ρεύμα γείωσης (V_{ss})	0V
Ρεύμα pin σε σχέση με το V_{ss}	-0.6V εως +(Vcc+0.6)V

Πίνακας 2.3 Ηλεκτρολογικά χαρακτηριστικά του αισθητήρα

Χαρακτηριστικά Συνεχούς Ρεύματος

Στα Vcc = +5.0V και TA = 25°C

Παράμετρος	Σύμβολο	Ελάχιστο	Μέσο	Μέγιστο	Μονάδες
Παροχή Ρεύματος	V_{cc}	4.5	5.0	5.5	V
Παροχή Τάσης	I_{cc}	80	115	135	mA

Πίνακας 2.4 Ηλεκτρολογικά χαρακτηριστικά συνεχούς ρεύματος

2.2.3 Δείκτες αναγνώρισης κατάστασης

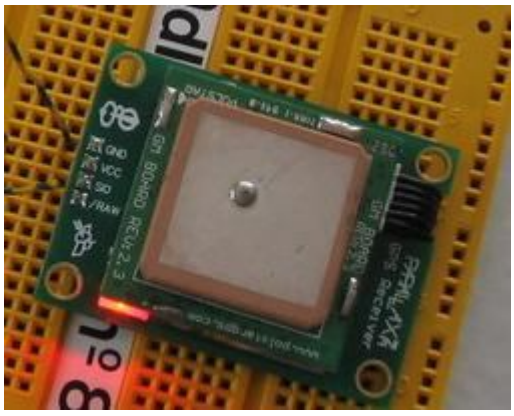
Ο δέκτης GPS έχει ένα απλό κόκκινο φωτάκι τύπου LED (Light-Emitting Diode) που υποδηλώνει την κατάσταση του συστήματος. Το LED βρίσκεται στην κάτω δεξιά γωνία του δέκτη (Εικόνα 2.11). Ένα άσπρο επικάλυμμα στην περιφέρεια του δέκτη χρησιμεύει στην αντανάκλαση του κόκκινου φωτός ώστε να γίνεται αντιληπτό από τον χρήστη. Το LED υποδηλώνει τις τρεις καταστάσεις του δέκτη :

Αναβοσβήνει (γρήγορα ή αργά) : Ψάχνει για δορυφόρους ή δεν βρέθηκαν δορυφόροι

Αναμμένο: Διασφαλίστηκε επιτυχώς σύνδεση με δορυφόρους (απαιτούνται τουλάχιστον 4 ώστε ο δέκτης να λαμβάνει έγκυρα δεδομένα)

Σβησμένο: Ο δέκτης βρίσκεται εκτός λειτουργίας – Δεν υπάρχει τροφοδοσία

Ανάλογα την τοποθεσία, ο δέκτης μπορεί να χρειαστεί 5 λεπτά ή και περισσότερα για να συγχρονιστεί με τον απαιτούμενο αριθμό δορυφόρων. Σε αυτό το χρονικό διάστημα το κόκκινο φωτάκι θα αναβοσβήνει. Όταν βρεθούν οι απαραίτητοι δορυφόροι για την λήψη έγκυρων δεδομένων, το κόκκινο φως θα μείνει μόνιμα αναμμένο. Ο αριθμός των δορυφόρων μπορεί να μην είναι σταθερός αλλά να ποικίλει για διάφορα χρονικά διαστήματα.



Εικόνα 2.11 Parallax GPS σε λειτουργία

2.2.4 Επιλογές Λειτουργίας

Το /RAW pin επιτρέπει στον χρήστη να επιλέξει δύο τρόπους λειτουργίας :

- Smart Mode : Όταν το /RAW pin δεν είναι συνδεδεμένο (ορίζεται ως HIGH), η προεπιλεγμένη κατάσταση λειτουργίας Smart Mode ενεργοποιείται και ο δέκτης πρέπει να πάρει συγκεκριμένες εντολές από τον χρήστη και να λάβει έτσι συγκεκριμένα δεδομένα. Στην επόμενη παράγραφο, «Πρωτόκολλο Επικοινωνίας», αναγράφονται περισσότερες πληροφορίες.

- Raw Mode : Όταν το /RAW pin έχει συνδεθεί (ορίζεται ως LOW), η κατάσταση λειτουργίας Raw Mode ενεργοποιείται και ο δέκτης τότε λαμβάνει πρότυπα αλφαριθμητικών NMEA0183 v2.2 (GGA, GSV, GSA και RMC). Έτσι οι χρήστες μπορούν να χρησιμοποιήσουν αυτά τα δεδομένα άμεσα όπως επιθυμούν.

Και στους δύο τρόπους λειτουργίας, τα δεδομένα μεταδίδονται στα 4800bps, 8 data bits, no parity, 1 stop bit, non-inverted, TTL.

2.2.5 Πρωτόκολλο επικοινωνίας

Smart mode

Η δομή των εντολών που δέχεται ο δέκτης, όταν βρίσκεται σε κατάσταση λειτουργίας Smart Mode, δηλαδή το pin 4 του δέκτη δεν έχει συνδεθεί, ακολουθεί το σειριακό πρωτόκολλο της Parallax, AppMod. Το pin 3, SIO, μεταφέρει τις εντολές που στέλνονται ΠΡΟΣ στον δέκτη και λαμβάνει δεδομένα ΑΠΟ τον δέκτη.

Η εντολή που θα στείλει ο χρήστης στον δέκτη πρέπει να ξεκινάει με τα αρχικά «!GPS», χωρίς τα εισαγωγικά, και στη συνέχεια να προσθέτει την εντολή της επιλογής του. Κάθε εντολή αποτελείται από ένα byte στο δεκαεξαδικό αριθμητικό σύστημα. Ανάλογα με την εντολή, επιστρέφεται ένας συγκεκριμένος αριθμός δεδομένων byte. Παρακάτω παρατίθεται η λίστα με όλες τις εντολές και τις τιμές που επιστρέφουν.

Εντολή	Σταθερά	Περιγραφή	Επιστρεφόμενα bytes	Τιμή
0x00	GetInfo	Έκδοση του δέκτη	2	Hardware, Firmware
0x01	GetValid	Έλεγχος εγκυρότητας δεδομένων	1	0 = Μη Έγκυρα 1=Έγκυρα
0x02	GetSats	Αριθμός δορυφόρων που βρέθηκαν	1	Αριθμός δορυφόρων
0x03	GetTime	Ώρα (UTC / Greenwich)	3	Ώρες, Λεπτά, Δεύτερα
0x04	GetDate	Ημερομηνία (UTC / Greenwich)	3	Μήνας, Μέρα, Χρονιά
0x05	GetLat	Γεωγραφικό Πλάτος	5	Μοίρες, Λεπτά, Δεύτερα, Κλασματικά λεπτά, Κατεύθυνση (0=Βορράς, 1=Νότος)
0x06	GetLong	Γεωγραφικό Μήκος	5	Μοίρες, Λεπτά, Δεύτερα, Κλασματικά λεπτά, Κατεύθυνση (0=Ανατολή, 1=Δύση)
0x07	GetAlt	Υψόμετρο από τη Μέση Στάθμη Θάλασσας (σε δεκάδες μέτρα, 65535 max)	2	Υψόμετρο
0x08	GetSpeed	Ταχύτητα πλοήγησης (σε δεκάδες knots)	2	Ταχύτητα
0x09	GetHead	Προσανατολισμός / Κατεύθυνση πλοήγησης (σε δεκάδες μοίρες)	2	Κατεύθυνση

Πίνακας 2.5 Λίστα εντολών και αντίστοιχων τιμών που επιστρέφουν

Στο παράρτημα υπάρχει ένα παράδειγμα κώδικα σε BASIC 2 Stamp, που μπορεί να χρησιμοποιηθεί με το κατάλληλο λειτουργικό, όταν ο δέκτης είναι σε κατάσταση Smart mode.

Κώδικας BASIC 2 Stamp για λειτουργία Smard Mode

```
' =====================================================================
'
' File..... GPSDemoV1.1.BS2
' Purpose... Demonstrates features of the Parallax GPS Receiver Module
```

```
' Author.... (c) Grand Idea Studio, Inc. [www.grandideastudio.com]
' E-mail.... support@parallax.com
' Updated... 04 Oct 2006 (by Parallax Tech Support)
'
' {$STAMP BS2}
' {$PBASIC 2.5}
'
' ============================================================================
' -----[ Program Description ]---------------------------------------------
'
' This program demonstrates the capabilities of the Parallax GPS Receiver
' Module.
'
' Before running this demo, ensure that the /RAW pin is left unconnected
' or pulled HIGH to enable "smart" mode, in which the GPS Receiver Module
' will accept commands and return the requested GPS data.
'
' For an application that requires constant monitoring of multiple GPS
' data components, it is recommended to use the "raw" mode of the GPS
' Receiver Module by pulling the /RAW Pin LOW. In this mode, the module
' will transmit a constant stream of raw NMEA0183 data strings, which can
' then be parsed by the host application.
' -----[ I/O Definitions ]--------------------------------------------------
Sio PIN 15 ' connects to GPS Module SIO pin
' -----[ Constants ]-------------------------------------------------------
T4800 CON 188
Open CON $8000
Baud CON Open | T4800 ' Open mode to allow daisy chaining
MoveTo CON 2 ' DEBUG positioning command
ClrRt CON 11 ' clear line right of cursor
FieldLen CON 22 ' length of debug text
EST CON -5 ' Eastern Standard Time
CST CON -6 ' Central Standard Time
MST CON -7 ' Mountain Standard Time
```

```
PST CON -8 ' Pacific Standard Time
EDT CON -4 ' Eastern Daylight Time
CDT CON -5 ' Central Daylight Time
MDT CON -6 ' Mountain Daylight Time
PDT CON -7 ' Pacific Daylight Time
UTCfix CON PST ' for San Diego, California
DegSym CON 176 ' degrees symbol for report
MinSym CON 39 ' minutes symbol
SecSym CON 34 ' seconds symbol
' GPS Module Commands
GetInfo CON $00
GetValid CON $01
GetSats CON $02
GetTime CON $03
GetDate CON $04
GetLat CON $05
GetLong CON $06
GetAlt CON $07
GetSpeed CON $08
GetHead CON $09
' -----[ Variables ]----------------------------------------------------
char VAR Byte
workVal VAR Word ' for numeric conversions
eeAddr VAR workVal ' pointer to EE data
ver_hw VAR Byte
ver_fw VAR Byte
valid VAR Byte ' signal valid? 0 = not valid, 1 = valid
sats VAR Byte ' number of satellites used in positioning calculations
tmHrs VAR Byte ' time fields
tmMins VAR Byte
tmSecs VAR Byte
day VAR Byte ' day of month, 1-31
month VAR Byte ' month, 1-12
```

```
year VAR Byte ' year, 00-99

degrees VAR Byte ' latitude/longitude degrees

minutes VAR Byte ' latitude/longitude minutes

minutesD VAR Word ' latitude/longitude decimal minutes

dir VAR Byte ' direction (latitude: 0 = N, 1 = S, longitude: 0 = E, 1 = W)

heading VAR Word ' heading in 0.1 degrees

alt VAR Word ' altitude in 0.1 meters

speed VAR Word ' speed in 0.1 knots

' -----[ EEPROM Data ]----------------------------------------------------

NotValid DATA "No", 0

IsValid DATA "Yes", 0

DaysInMon DATA 31,28,31,30,31,30,31,31,30,31,30,31

MonNames DATA "JAN",0,"FEB",0,"MAR",0,"APR",0,"MAY",0,"JUN",0
DATA "JUL",0,"AUG",0,"SEP",0,"OCT",0,"NOV",0,"DEC",0

' -----[ Initialization ]--------------------------------------------------

Initialize:

PAUSE 250 ' let DEBUG open

DEBUG CLS ' CLEAR THE SCREEN
DEBUG "Parallax GPS Receiver Module Test Application", CR,

"-------------------------------------------"

Draw_Data_Labels:

DEBUG MoveTo, 0, 3, " Hardware Version: "

DEBUG MoveTo, 0, 4, " Firmware Version: "

DEBUG MoveTo, 0, 6, " Signal Valid: "

DEBUG MoveTo, 0, 7, " Acquired Satellites: "

DEBUG MoveTo, 0, 9, " Local Time: "

DEBUG MoveTo, 0, 10, " Local Date: "

DEBUG MoveTo, 0, 12, " Latitude: "

DEBUG MoveTo, 0, 13, " Longitude: "

DEBUG MoveTo, 0, 14, " Altitude: "

DEBUG MoveTo, 0, 15, " Speed: "

DEBUG MoveTo, 0, 16, " Direction of Travel: "

' -----[ Program Code ]----------------------------------------------------

Main:
```

```
GOSUB Get_Info
GOSUB Get_Valid
GOSUB Get_Sats
GOSUB Get_TimeDate
GOSUB Get_Lat
GOSUB Get_Long
GOSUB Get_Alt
GOSUB Get_Speed
GOSUB Get_Head
GOTO Main
' -----[ Subroutines ]------------------------------------------------------
' --------------------------------------------------
Get_Info:
SEROUT Sio, Baud, ["!GPS", GetInfo]
SERIN Sio, Baud, 3000, No_Response, [ver_hw, ver_fw]
DEBUG MoveTo, FieldLen, 3, HEX ver_hw.HIGHNIB, ".", HEX ver_hw.LOWNIB
DEBUG MoveTo, FieldLen, 4, HEX ver_fw.HIGHNIB, ".", HEX ver_fw.LOWNIB
RETURN
' --------------------------------------------------
Get_Valid:
SEROUT Sio, Baud, ["!GPS", GetValid]
SERIN Sio, Baud, 3000, No_Response, [valid]
DEBUG MoveTo, FieldLen, 6 ' was the signal valid?
LOOKUP valid, [NotValid, IsValid], eeAddr ' get answer from EE
GOSUB Print_Z_String ' print it
DEBUG ClrRt ' clear end of line
IF (valid = 0) THEN Signal_Not_Valid
RETURN
' --------------------------------------------------
Get_Sats:
SEROUT Sio, Baud, ["!GPS", GetSats]
SERIN Sio, Baud, 3000, No_Response, [sats]
DEBUG MoveTo, FieldLen, 7, DEC sats
```

```
RETURN

' -----------------------------------------------------

Get_TimeDate:

SEROUT Sio, Baud, ["!GPS", GetTime]

SERIN Sio, Baud, 3000, No_Response, [tmHrs, tmMins, tmSecs]

SEROUT Sio, Baud, ["!GPS", GetDate]
SERIN Sio, Baud, 3000, No_Response, [day, month, year]

GOSUB Correct_Local_Time_Date

DEBUG MoveTo, FieldLen, 9, DEC2 tmHrs, ":", DEC2 tmMins, ":", DEC2 tmSecs

DEBUG MoveTo, FieldLen, 10, DEC2 day, " "

eeAddr = (month - 1) * 4 + MonNames ' get address of month name

GOSUB Print_Z_String ' print it

DEBUG " 20", DEC2 year

RETURN

' -----------------------------------------------------

Get_Lat:

SEROUT Sio, Baud, ["!GPS", GetLat]

SERIN Sio, Baud, 3000, No_Response, [degrees, minutes, minutesD.HIGHBYTE,
minutesD.LOWBYTE, dir]

' convert decimal minutes to tenths of seconds

workVal = minutesD ** $0F5C ' minutesD * 0.06

DEBUG MoveTo, FieldLen, 12, DEC3 degrees, DegSym, " ", DEC2 minutes, MinSym, " "

DEBUG DEC2 (workVal / 10), ".", DEC1 (workVal // 10), SecSym, " "

DEBUG "N" + (dir * 5)

' convert to decimal format, too

workVal = (minutes * 1000 / 6) + (minutesD / 60)

DEBUG " (", " " + (dir * 13), DEC degrees, ".", DEC4 workVal, " ) "

RETURN

' -----------------------------------------------------

Get_Long:

SEROUT Sio, Baud, ["!GPS", GetLong]

SERIN Sio, Baud, 3000, No_Response, [degrees, minutes, minutesD.HIGHBYTE,
minutesD.LOWBYTE, dir]

' convert decimal minutes to tenths of seconds
```

```
workVal = minutesD ** $0F5C ' minutesD * 0.06
DEBUG MoveTo, FieldLen, 13, DEC3 degrees, DegSym, " ", DEC2 minutes, MinSym, " "
DEBUG DEC2 (workVal / 10), ".", DEC1 (workVal // 10), SecSym, " "
DEBUG "E" + (dir * 18)
' convert to decimal format, too
workVal = (minutes * 1000 / 6) + (minutesD / 60)
DEBUG " (", " " + (dir * 13), DEC degrees, ".", DEC4 workVal, " ) "
RETURN
' -----------------------------------------------------
Get_Alt:
SEROUT Sio, Baud, ["!GPS", GetAlt]
SERIN Sio, Baud, 3000, No_Response, [alt.HIGHBYTE, alt.LOWBYTE]
DEBUG MoveTo, FieldLen, 14, DEC (alt / 10), ".", DEC1 (alt // 10), " meters "
workVal = alt / 10 ' remove tenths from altitude
' convert altitude from meters to feet
workVal = (workVal * 3) + (workVal ** $47E5) ' 1 meter = 3.2808399 feet
DEBUG " ( ", DEC workVal, " feet ) "
RETURN
' -----------------------------------------------------
Get_Speed:
SEROUT Sio, Baud, ["!GPS", GetSpeed]
SERIN Sio, Baud, 3000, No_Response, [speed.HIGHBYTE, speed.LOWBYTE]
DEBUG MoveTo, FieldLen, 15, DEC (speed / 10), ".", DEC1 (speed // 10), " Knots "
' convert speed from knots to MPH
workVal = speed + (speed ** $2699) ' 1 knot = 1.1507771555 MPH
DEBUG " ( ", DEC (workVal / 10), ".", DEC1 (workVal // 10), " MPH ) "
RETURN
' -----------------------------------------------------
Get_Head:
SEROUT Sio, Baud, ["!GPS", GetHead]
SERIN Sio, Baud, 3000, No_Response, [heading.HIGHBYTE, heading.LOWBYTE]
IF speed = 0 THEN
DEBUG MoveTo, FieldLen, 16, "N/A "
```

```
ELSE
DEBUG MoveTo, FieldLen, 16, DEC (heading / 10), ".", DEC1 (heading // 10), DegSym,
" "
ENDIF
RETURN
' -----------------------------------------------------
No_Response:
DEBUG MoveTo, 0, 18, "Error: No response from GPS Receiver Module"
PAUSE 5000
GOTO Initialize
'
' -----------------------------------------------------
Signal_Not_Valid:
DEBUG MoveTo, FieldLen, 7, "?", ClrRt ' clear all fields
DEBUG MoveTo, FieldLen, 9, "?", ClrRt
DEBUG MoveTo, FieldLen, 10, "?", ClrRt
DEBUG MoveTo, FieldLen, 12, "?", ClrRt
DEBUG MoveTo, FieldLen, 13, "?", ClrRt
DEBUG MoveTo, FieldLen, 14, "?", ClrRt
DEBUG MoveTo, FieldLen, 15, "?", ClrRt
DEBUG MoveTo, FieldLen, 16, "?", ClrRt
GOTO Main
' -----------------------------------------------------
' adjust date for local position
Correct_Local_Time_Date:
workVal = tmHrs + UTCfix ' add UTC offset
IF (workVal < 24) THEN Adjust_Time ' midnight crossed?
workVal = UTCfix ' yes, so adjust date
BRANCH workVal.BIT15, [Location_Leads, Location_Lags]
Location_Leads: ' east of Greenwich
day = day + 1 ' no, move to next day
eeAddr = DaysInMon * (month - 1) ' get days in month
READ eeAddr, char
```

```
IF (day <= char) THEN Adjust_Time ' in same month?
month = month + 1 ' no, move to next month
day = 1 ' first day
IF (month < 13) THEN Adjust_Time ' in same year?
month = 1 ' no, set to January
year = year + 1 // 100 ' add one to year
GOTO Adjust_Time
Location_Lags: ' west of Greenwich
day = day - 1 ' adjust day
IF (day > 0) THEN Adjust_Time ' same month?
month = month - 1
IF (month > 0) THEN Adjust_Time ' same year?
month = 1 ' no, set to January
eeAddr = DaysInMon * (month - 1)
READ eeAddr, day ' get new day
year = year + 99 // 100 ' set to previous year
Adjust_Time:
tmHrs = tmHrs + (24 + UTCfix) // 24 ' localize hours
RETURN
' -----------------------------------------------------

' Print Zero-terminated string stored in EEPROM
' -- eeAddr - starting character of string
Print_Z_String:
READ eeAddr, char ' get char from EE
IF (char = 0) THEN Print_Z_String_Done ' if zero, we're done
DEBUG char ' print the char
eeAddr = eeAddr + 1 ' point to the next one
GOTO Print_Z_String
Print_Z_String_Done:
RETURN
' -----------------------------------------------------
```

Raw Mode

Όταν ο δέκτης εισέρχεται σε λειτουργία /Raw mode, ο δέκτης λαμβάνει συνέχεια δεδομένα, ανεξαρτήτως αν έχει βρεθεί ο απαραίτητος αριθμός δορυφόρων. Αυτό σημαίνει ότι όλα τα δεδομένα που λαμβάνει δεν μπορεί να είναι οποιαδήποτε χρονική στιγμή έγκυρα. Σε αντίθεση με την λειτουργία Smart mode, ο χρήστης δεν χρειάζεται να ζητήσει με κάποια εντολή, μία συγκεκριμένη πληροφορία. Όλες οι πληροφορίες λαμβάνονται συνεχώς από τον δέκτη και είναι έτοιμες για επεξεργασία.

Στην παρούσα εργασία, η επικοινωνία με τον δέκτη GPS έγινε σε /RAW mode, αφού η δυνατότητα παρέμβασης και επεξεργασίας επί των δεδομένων που λαμβάνει ο χρήστης από το GPS είναι άμεση. Μόνο έτσι θα ήταν δυνατή η υλοποίηση της εργασίας, δεδομένου του σκοπού της.

Ο προγραμματισμός για την υλοποίηση του πειράματος έγινε με βάση την κεντρική ιδέα, ότι δηλαδή δεδομένα λαμβάνονται συνέχεια. Επίσης, απαραίτητη είναι η γνώση διαμόρφωσης των δεδομένων που λαμβάνει ο δέκτης. Τα δεδομένα ακολουθούν το πρωτόκολλο NMEA0183 και άρα κάθε αλφαριθμητικό έχει συγκεκριμένη μορφή. Ο αλγόριθμος λειτουργεί, περιμένοντας το κατάλληλο byte που ορίζει ότι ξεκινάει η επιθυμητή πληροφορία ($). Εν συνεχεία, κάθε byte που λαμβάνεται από τον δέκτη αποθηκεύεται σε μία μεταβλητή. Τέλος, με την λήψη των δύο bytes που ακολουθούν τον αστερίσκο, τελειώνει η σειρά δεδομένων, ο αλγόριθμος τελειώνει και ξεκινάει πάλι από την αρχή.

2.2.6 Άλλες προδιαγραφές

- Η συχνότητα ανανέωσης πλοήγησης του δέκτη GPS Parallax είναι μία κάθε δευτερόλεπτο.

- Υψηλή ευαισθησία (-152 dBm για ευθυγράμμιση τροχιών και –139 dBm ανάκτηση).

- Ο δέκτης περιέχει επαναφορτιζόμενη μπαταρία για μνήμη και ρολόι.

- Ο δέκτης παρέχει μέσος όρο ακρίβειας προσδιορισμού θέσης +/- 5 μέτρα και +/- 0.1 μέτρα ανά δευτερόλεπτο ακρίβεια προσδιορισμού ταχύτητας.

2.2.7 Περισσότερα για την τεχνολογία GPS

Όπως αναφέρθηκε στο Κεφάλαιο 1 το GPS (Global Positioning System) αποτελείται από τρία μέρη : το τμήμα Διαστήματος, το τμήμα Ελέγχου και το τμήμα Χρήστη. Το τμήμα Διαστήματος αποτελείται από ένα σύνολο 24 ενεργών δορυφόρων και έναν ή περισσότερους σε ελεύθερες τροχιές, κινούμενοι γύρω από τη γη κάθε 12 ώρες. Τέσσερις δορυφόροι βρίσκονται σε κάθε μία από τις έξι τροχιές και είναι ορατοί από οποιαδήποτε τοποθεσία. Οι τροχιές είναι διαθέσιμες περιμετρικά της Γης και έχουν κλίση 55 μοίρες από τον ισημερινό. Οι δορυφόροι βρίσκονται σε τροχιά υψομέτρου 11 χιλιάδες ναυτικά μίλια.

Κάθε δορυφόρος εκπέμπει δύο σήματα : το L1, σε συχνότητα 1575.42 MHz, και το L2 στα 1227.60 MHz. Το σήμα L1 διαμορφώνεται με δύο ψευδο-τυχαία ηχητικά σήματα – το σήμα προστασίας P (Protected) και το σήμα πορείας/πρόσκτησης C/A (Course/Acquisition). Το σήμα L2 φέρει μόνο το σήμα προστασίας P. Οι δέκτες GPS που προορίζονται για πολίτες, λαμβάνουν μόνο το σήμα C/A στην συχνότητα L1. Κάθε σήμα από κάθε δορυφόρο περιέχει ένα επαναλαμβανόμενο μήνυμα το οποίο προσδιορίζει, την θέση και τις τροχιακές παραμέτρους, τόσο του ίδιου του δορυφόρου όσο και των υπολοίπων (αλμανάκο), την κατάσταση «υγείας» των δορυφόρων (health bit) και την ακριβή ατομική ώρα.

Ο δέκτης μετράει τον χρόνο που χρειάζεται το σήμα για να φτάσει από τον δορυφόρο στον δέκτη, γνωρίζοντας την ώρα που ξεκίνησε το σήμα από τον δορυφόρο, και αφαιρώντας το από την ώρα που έφτασε στο δέκτη, με βάση το εσωτερικό του ρολόι. Αν ο δέκτης είχε τέλειο ρολόι, συγχρονισμένο με ακρίβεια με τους δορυφόρους, οι τρεις μετρήσεις από τους τρεις δορυφόρους θα ήταν αρκετές για να προσδιοριστεί η θέση στις τρεις διαστάσεις (X, Y, Z), μέσω τριγωνισμού. Όμως, αυτό αποτελεί ιδανική περίπτωση, οπότε χρειάζεται και ένας τέταρτος δορυφόρος για την επίλυση του σφάλματος του ρολογιού. Με τέσσερις δορυφόρους, ένας δέκτης GPS παρέχει πληροφορίες ακριβείας, για την ώρα, την ημερομηνία και την θέση (γεωγραφικό πλάτος, γεωγραφικό μήκος, υψόμετρο, ταχύτητα, προσανατολισμός πλοήγησης).

Τα δεδομένα θέσης και η ακρίβεια εξαρτώνται από την γεωμετρία του δορυφόρου, την ηλεκτρομαγνητική παρεμβολή και από απρόβλεπτες αντανακλάσεις και κύματα - κάθε ένα με τη δική του εξασθένηση και καθυστέρηση. Κυρίως, λόγω της γεωμετρίας του δορυφόρου, η μέτρηση του υψομέτρου με GPS μπορεί να περιέχει σφάλμα 1.5 φορές μεγαλύτερο από την ακρίβεια προσδιορισμού θέσης. Ο δέκτης GPS της Parallax παρέχει μέτρηση υψομέτρου με ακρίβεια +/- 20 μέτρα.

Τα σήματα GPS λειτουργούν σε συχνότητες ράδιου με μικροκύματα. Μπορούν να διαπεράσουν το γυαλί αλλά απορροφούνται από τα μόρια νερού, ξύλο και πυκνές φυλλωσιές, και αντανακλούνται στο σκυρόδεμα, το μέταλλο και την πέτρα. Αυτό σημαίνει ότι οι μονάδες GPS θα έχουν πρόβλημα στην λειτουργία τους σε δάση, ζούγκλες, φαράγγια, μέσα σε αμάξια, καράβια ή σε πυκνή χιονόπτωση. Αυτές οι περιβαλλοντικές συνθήκες μειώνουν την ακρίβεια ή ακόμα και καθιστούν αδύνατη τη λειτουργία του δέκτη.

2.3 Libelium MicroSD Module

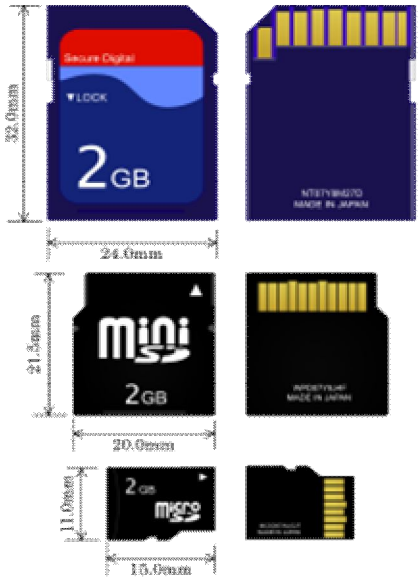
SD είναι η κάρτα μνήμης που κατασκευάστηκε για φορητές συσκευές από την SD Card Association. Η τεχνολογία SD χρησιμοποιείται από περισσότερες από 400 εταιρίες σε δεκάδες κατηγορίες προϊόντων και σε περισσότερα από 8000 μοντέλα.

Οι κάρτες μνήμης SD μπορούν να διαφοροποιηθούν με βάση την χωρητικότητά τους (Εικόνα 2.12α). Υπάρχουν οι κάρτες κατηγορίας SDSC (Standard Capacity) οι οποίες έχουν μέγιστη χωρητικότητα 2 GB, αλλά υπάρχουν και κάποιες μέχρι και 4 GB. Οι SDHC (High Capacity) έχουν χωρητικότητα από 4 εως 32 GB. Επιπλέον, υπάρχουν οι κάρτες SD που ανήκουν στις SDXC (eXtended Capacity), οι οποίες ξεκινούν από 32 GB και φτάνουν έως και 2 TB. Η διαθεσιμότητα καρτών μνήμης SD των 4 GB, στην γκάμα προϊόντων SDSC αλλά και στην SDHC, προκαλεί αρκετή σύγχυση στους χρήστες, καθώς κάθε κατηγορία έχει διαφορετικό πρωτόκολλο επικοινωνίας.

Επίσης, η διαφοροποίηση μεταξύ των καρτών μνήμης SD [13] γίνεται με βάση την συσκευή για την οποία προορίζονται (Εικόνα 2.12β). Οι SD/miniSD/microSD ανήκουν στην κατηγορία χωρητικότητας SDSC. Οι SDHC/miniSDHC/microSDHC ανήκουν στις κάρτες χωρητικότητας SDHC και αντίστοιχα οι SDXC/microSDXC στην SDXC. Οι

προσαρμογείς (adapters) SD διευκολύνουν την χρήση των μικροσκοπικών καρτών μνήμης SD. Στους προσαρμογείς, μεγαλύτερου σχετικά μεγέθους από τις ίδιες τις κάρτες SD, εισέρχεται η κάρτα SD. Ουσιαστικά, πρόκειται για συσκευές που συνδέουν τα pins από τις κάρτες SD στα pins του μεγαλύτερου SD προσαρμογέα. Παρόλες τις διαφορές στην χωρητικότητα των καρτών μνήμης SD, το μέγεθός τους μπορεί να είναι ίδιο.

Το πρωτόκολλο επικοινωνίας για τις SDHC και SDXC είναι ελάχιστα διαφορετικό σε σχέση με τις SDSC, κάτι που προκαλεί προβλήματα στην λειτουργία σε παλαιότερες συσκευές, οι οποίες δεν αναγνωρίζουν τις νέες κατηγορίες καρτών μεγαλύτερης χωρητικότητας. Όταν μία κάρτα SDHC ή SDXC εισέλθει σε μία παλιά συσκευή SDSC, δεν προκαλεί κάποια φυσική ή ηλεκτρική ζημιά στην κάρτα ή στην συσκευή, αλλά η συσκευή δεν θα αναγνωρίσει την κάρτα. Τα περισσότερα προβλήματα τέτοιου είδους μπορούν να επιλυθούν εύκολα από μία ανανέωση στην έκδοση firmware του προϊόντος της συσκευής, αλλά δυστυχώς οι εταιρίες σπάνια διορθώνουν προβλήματα σε παλαιότερες συσκευές.



Εικόνα 2.12α SD, miniSD, microSD (από πάνω προς τα κάτω)

Εικόνα 2.12β Σύγκριση μεγέθους SD, miniSD, microSD

2.3.1 Συνδεσμολογία MicroSD module

Η κύρια ιδέα της χρήσης μίας μικροσυσκευής MicroSD module (Εικόνα 2.13) είναι να παρέχεται η δυνατότητα αποθήκευσης ενός μεγάλου ποσού δεδομένων από τους αισθητήρες, όταν δεν υπάρχει διαθέσιμη ασύρματη επικοινωνία με το ασύρματο δίκτυο των αισθητήρων. Συγκεκριμένα, ένα microSD module με μια μικροκάρτα SD μπορεί να συνδεθεί στο Arduino μέσω της σειριακής θύρας USB και έπειτα να γίνει ανάγνωση των δεδομένων απευθείας από οποιονδήποτε υπολογιστή.

Το microSD module μπορεί να συνδεθεί με την πλατφόρμα μέσω της ICSP θύρας και μέσω των ψηφιακών εισόδων 8-13 (Εικόνα 2.3β), έτσι ώστε να τροφοδοτείται εύκολα και χωρίς προβλήματα. Όταν η μικροσυσκευή συνδεθεί στις ψηφιακές εξόδους 8-13, παρέχεται τροφοδότηση της τάξεως 5V συνήθως από την έξοδο pin 8, η οποία τίθεται σε λειτουργία HIGH. Ανάλογα με το που θα συνδεθεί η microSD module στο Arduino, πρέπει να αλλάξει η θέση του jumper παροχής ρεύματος. Αν η μικροκάρτα συνδεθεί στις ψηφιακές εξόδους 8-13 τότε το jumper πρέπει να τοποθετηθεί στα 3V3 και GND. Αλλιώς, αν συνδεθεί στον διασυνδετικό αγωγό ICSP, το jumper τοποθετείται στο 5V και 3V3 (Εικόνα 2.3γ).

Συγκεκριμένα, στο πείραμα χρησιμοποιήθηκε μία κάρτα χωρητικότητας 1 gigabyte. Επιλέχθηκε η μικροσυσκευή της εταιρίας Libelium [17] διότι παρουσιάζει το πλεονέκτημα της άμεσης σύνδεσής της με την πλατφόρμα χωρίς τη χρήση καλωδίων.

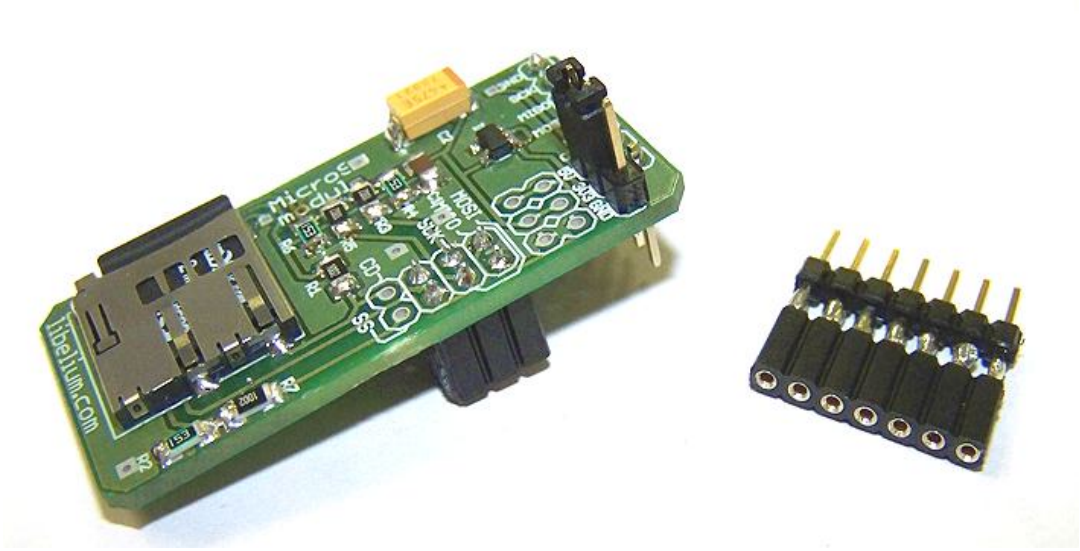
Οι microSD κάρτες έχουν την δυνατότητα επικοινωνίας με ένα τμηματικό πρωτόκολλο, το οποίο αποκαλείται SPI που υποστηρίζεται από το υλικό των μικροεπεξεργαστών. Απαιτούνται μόνο τέσσερις γραμμές I/O για την διαβίβαση των εντολών και των δεδομένων προς την κάρτα, καθώς και την παραλαβή δεδομένων από την κάρτα (Εικόνα 2.3δ). Οι συνδέσεις μπορούν να γίνουν με πληθώρα συνδυασμών, αλλά ενδιαφέρον παρουσιάζουν τα εξής χαρακτηριστικά που παρέχονται σε όλες τις κάρτες, τα οποία είναι:

- /CS (δίχως επιλογή κάρτας)
- CLK (ρολόι)

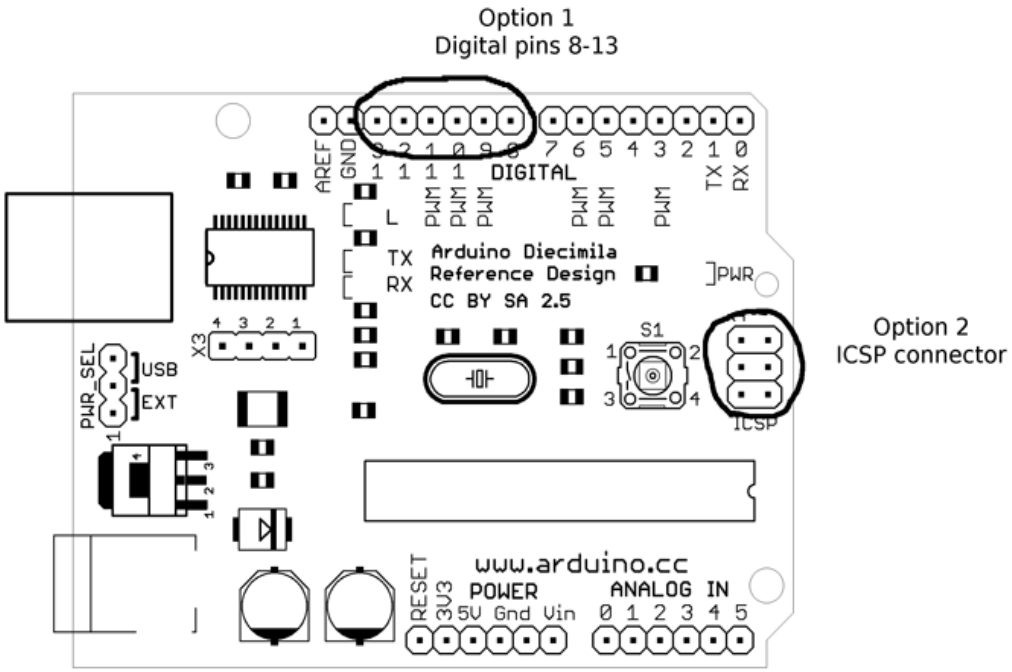
- MOSI (master out, slave in)

- MISO (master in, slave out)

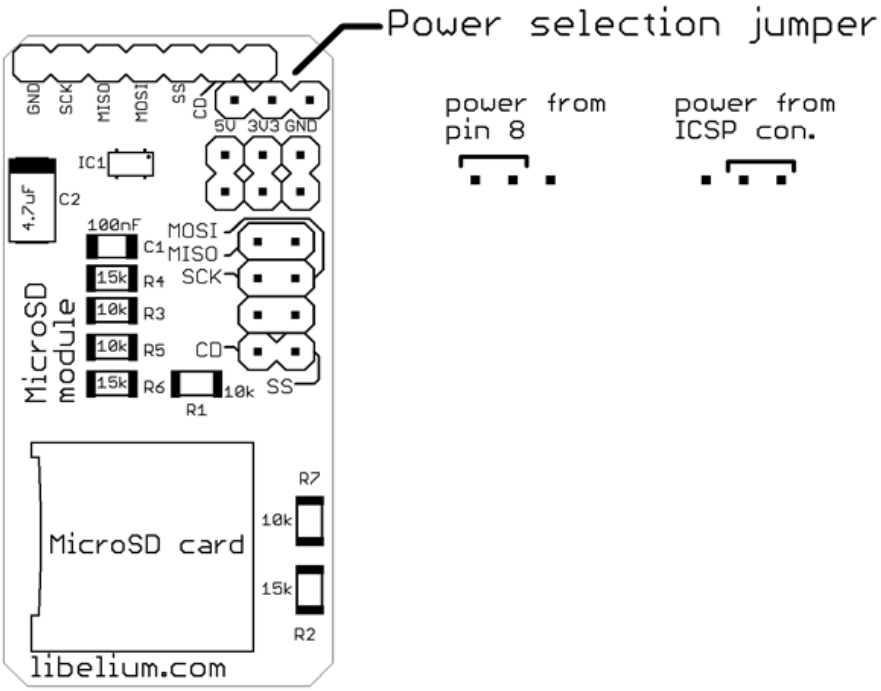
Οι microSD κάρτες απαιτούν μεταξύ 2.7 και 3.6 V για την ορθή λειτουργία τους. Χαμηλής ισχύος κάρτες που ανέρχονται στα 1.8 V έχουν τη δυνατότητα να ανταπεξέλθουν, αν και θεωρούνται σπάνιες και προτείνονται για ειδικές εφαρμογές.



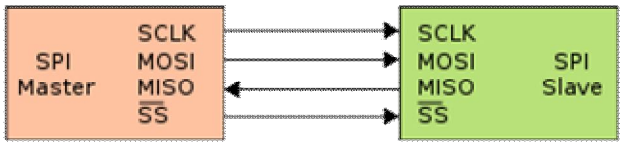
Εικόνα 2.13 MicroSD Module



Εικόνα 2.14 Σύνδεση MicroSD module σε ICSP ή στα pins 8-13



Εικόνα 2.15 Τοποθέτηση jumper ανάλογα με τη σύνδεση



Εικόνα 2.16 Σύνδεση με πρωτόκολλο επικοινωνίας SPI

2.4 NMEA (National Marine Electronics Association)

Η NMEA έχει αναπτύξει ένα σύστημα το οποίο προσδιορίζει τον τρόπο σύνδεσης μεταξύ των στρατιωτικών ηλεκτρονικών συσκευών. Το πρότυπο NMEA επιτρέπει στις στρατιωτικές ηλεκτρονικές συσκευές να στέλνουν πληροφορίες σε υπολογιστές και άλλους στρατιωτικούς εξοπλισμούς.

Τα περισσότερα προγράμματα υπολογιστών που παρέχουν δεδομένα θέσης, επεξεργάζονται και αναμένουν, γενικότερα, δεδομένα σε μορφή NMEA. Αυτή η μορφή δεδομένων περιλαμβάνει τρεις παραμέτρους που λαμβάνονται από τον δέκτη GPS : Θέση, Ταχύτητα, Ώρα. Η βασική ιδέα υλοποίησης του NMEA είναι μία γραμμή δεδομένων, που καλείται «παράγραφος». Κάθε παράγραφος είναι τελείως ανεξάρτητη από άλλες παραγράφους. Υπάρχουν πρότυπα παραγράφων για κάθε κατηγορία συσκευής. Όλα τα πρότυπα παραγράφων ξεκινούν με δύο γράμματα που προσδιορίζουν τον τύπο συσκευής που χρησιμοποιεί το συγκεκριμένο είδος παραγράφου, και ακολουθούν τρία γράμματα που προσδιορίζουν το περιεχόμενο της παραγράφου. Για τους δέκτες GPS, τα δύο πρώτα γράμματα είναι GP.

Κάθε παράγραφος ξεκινάει με το αλφαριθμητικό «$» και τελειώνει με ένα carriage return/line feed και δεν μπορεί να περιέχει πάνω από 80 χαρακτήρες (συμπεριλαμβανομένου του τέλους της παραγράφου). Τα δεδομένα διαχωρίζονται μεταξύ τους με ένα κόμμα («,»). Από μόνα τους τα δεδομένα είναι απλά χαρακτήρες ASCII και μπορούν να επεκταθούν σε περισσότερες παραγράφους με συγκεκριμένους εξειδικευμένους τρόπους, αλλά κανονικά περιέχονται όλα σε μία γραμμή δεδομένων. Η μονάδα μέτρησης των δεδομένων που περιέχονται σε μία παράγραφο NMEA, ποικιλεί ανάλογα με την ακρίβεια τους. Για παράδειγμα, η ώρα μπορεί να υποδηλώνεται σε

δέκατα του δευτερολέπτου και η θέση με ακρίβεια 3 ή 4 δεκαδικών ψηφίων. Οι αλγόριθμοι που διαβάζουν τα δεδομένα πρέπει να χρησιμοποιούν τα κόμματα για να προσδιορίσουν τις τιμές τους. Στο τέλος κάθε παραγράφου υπάρχει ένας αριθμός checksum που μπορεί να ελέγχεται από την μονάδα που διαβάζει τα δεδομένα. Ο αριθμός checksum ξεκινάει με έναν αστερίσκο («*») που ακολουθείται από δύο δεκαεξαδικά ψηφία, τα οποία αναπαριστούν την τιμή που προκύπτει αν σε όλους τους χαρακτήρες μεταξύ του «δολλαρίου» $ και του αστερίσκου * εφαρμοστεί η πράξη XOR (eXclusive OR). Μερικές παράγραφοι δεν περιέχουν checksum.

Κάθε παράγραφος ερμηνεύεται με μοναδικό τρόπο που ορίζεται από το πρότυπο NMEA. Κάποια δεδομένα μπορεί να παρέχουν ίδιες πληροφορίες με άλλα, όμως θα περιέχουν σίγουρα κάτι νέο. Οι συσκευές ή τα προγράμματα που διαχειρίζονται δεδομένα NMEA μπορούν να επεξεργάζονται συγκεκριμένες παραγράφους και να αγνοούν όποιες δεν τους είναι χρήσιμες.

Το πρότυπο NMEA δεν περιέχει εντολές για GPS. Κάποιοι δέκτες έχουν εντολές με τις οποίες μπορεί ένας χρήστης να επιλέξει να ληφθούν όλες ή ένα σύνολο παραγράφων που επιθυμεί. Δεν υπάρχει τρόπος να σταλεί πάλι κάτι στον δέκτη, αν για παράδειγμα κάποια δεδομένα δεν τα λάβαμε ή δεν διαβάστηκαν σωστά. Ωστόσο, κάποιοι δέκτες GPS ελέγχουν το checksum και αγνοούν όποια δεδομένα έχουν λάθος, οπότε τα δεδομένα θα σταλούν πάλι αμέσως μετά.

2.4.1 Παραγράφοι GGA, GSA, GSV, RMC

- **GGA – Απαραίτητα δεδομένα προσδιορισμού θέσης 3D και δεδομένα για την ακρίβεια**

Παράδειγμα : $GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47

Όπου :

GGA Global Positioning System Fix Data

123519 Ώρα λήψης 12:35:19 UTC

4807.038,N Γεωγραφικό πλάτος 48° και 07.038’ προς Βορρά (North)

01131.000,E Γεωγραφικό μήκος 11° και 31.000’ προς Ανατολάς (East)

1 Τύπος δεδομένων : 0 = Εσφαλμένο

1 = GPS (SPS)

2 = DGPS

3 = PPS

4 = Κινηματικός πραγματικού χρόνου

5 = RTK

6 = Dead reckoning

7 = Λειτουργία Manual

8 = Λειτουργία Simulation

08 Αριθμός δορυφόρων που εντοπίστηκαν

0.9 Οριζόντια παραμόρφωση θέσης

545.4,M Υψόμετρο, μέτρα πάνω από την Μέση Στάθμη Θάλασσας

46.9M Υψόμετρο γεωειδούς (Μέση Στάθμη Θάλασσας) πάνω από το ελληψοειδές WSG84

(Κενό) Χρόνος σε δευτερόλεπτα από την τελευταία ανανέωση DGPS

(Κενό) Αριθμός ID του σταθμού DGPS

*47 Checksum (πάντα ξεκινάει με *)

Αν το υψόμετρο του γεωειδούς λείπει τότε το υψόμετρο μπορεί να είναι λανθασμένο. Κάποια μη-σταθερά πρότυπα περιλαμβάνουν το υψόμετρο ελλειψοειδούς αντί του γεωειδούς. Επίσης, κάποιες μονάδες δεν αναφέρουν καθόλου αρνητικά υψόμετρα.

Αυτή είναι η μόνη παράγραφος στο σύστημα NMEA με αναφορά σε υψόμετρο.

• GSA – DOP (Dilution Of Precision) και ενεργοί δορυφόροι

Αυτή η παράγραφος περιέχει πληροφορίες για την φύση των δεδομένων θέσης. Τα δεδομένα αυτής της παραγράφου περιλαμβάνουν τον αριθμό των δορυφόρων που χρησιμοποιούνται καθώς και τον αριθμό DOP. Ο αριθμός DOP προσδιορίζει το κατά πόσο η γεωμετρία των δορυφόρων επηρεάζει την ακρίβεια προσδιορισμού θέσης. Είναι ένας μοναδικός αριθμός, ο οποίος όσο πιο πολύ πλησιάζει τη μονάδα τόσο καλύτερα είναι τα αποτελέσματα. Για συντεταγμένες προσδιορισμένες στο τρισδιάστατο σύστημα αναφοράς (3D), με χρήση 4 δορυφόρων, για DOP = 1.0 θεωρείται τέλειο αποτέλεσμα.

Στην παράγραφο GSA υπάρχουν διαφορές στον τρόπο που παρουσιάζονται τα δεδομένα, κάτι που μπορεί να επηρεάσει την λειτουργία των προγραμμάτων που τα επεξεργάζονται. Στο παράδειγμα παρακάτω, υπάρχουν 5 δορυφόροι στην επίλυση και στη θέση των κενών θα έπρεπε να υπάρχουν πληροφορίες για τους δορυφόρους οι οποίοι δεν φαίνονται. Κάποιοι άλλοι δέκτες εξάγουν τα όλα δεδομένα στην αρχή και στο τέλος προσθέτουν τα κενά. Αυτή η διαφορά παίζει ρόλο σε κάποια προγράμματα που εμφανίζουν τους ενεργούς δορυφόρους και ενδεχομένως να εμφανίζουν λάθος πληροφορίες.

Παράδειγμα: $GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39

Όπου :

GSA Κατάσταση Δορυφόρων

A Αυτοματοποιημένη επιλογή 2D ή 3D επίλυσης (M = Manual)

3 3D επίλυση 1 = Καμία επίλυση

2 = Επίλυση σε 2D

3 = Επίλυση σε 3D

04,05… PRNs [...] των δορυφόρων (χώρος για 12)

2.5 DOP (Dilution Of Precision)

Οριζοντιακή ακρίβεια (Horizontal DOP)

Κατακόρυφη ακρίβεια (Vertical DOP)

*39 Checksum (πάντα ξεκινάει με *)

- **GSV – Satellites in View**

Περιέχει δεδομένα για τους δορυφόρους που μπορεί να βρει ο δέκτης. Μία παράγραφος GSV μπορεί να περιέχει δεδομένα για μέχρι 4 δορυφόρους και άρα μπορεί να χρειάζονται έως και 3 τέτοιες παράγραφοι για να ληφθεί πλήρης πληροφορία για τους δορυφόρους. Είναι λογικό η παράγραφος GSV να περιέχει περισσότερους δορυφόρους από την GGA αφού η πρώτη περιέχει και τους δορυφόρους που δεν χρειάστηκαν για την επίλυση. Τα περιεχόμενα της παραγράφου GSV μπορεί να μην εμφανίζονται σε σειρά.

Το πεδίο με την ονομασία SNR (Signal to Noise Ratio) στο πρότυπο NMEA έχει να κάνει με την δύναμη του σήματος. Το SNR είναι μία τιμή που προσδιορίζει έμμεσα την ποιότητα του σήματος με το οποίο οι δορυφόροι εκπέμπουν τα δεδομένα. Μπορεί να πάρει τιμές από 0 έως 99 και οι μονάδες του είναι σε dB (decibel). Τα όρια των έγκυρων τιμών σε ένα δεδομένο GPS θα έχουν συχνά διαφορά 25 μέχρι 35 μονάδες μεταξύ χαμηλότερης και ψηλότερης τιμής. Η τιμή 0 αποτελεί ειδική περίπτωση και εμφανίζεται όταν ο δέκτης «βλέπει» τους δορυφόρους αλλά δεν μπορεί να τους χρησιμοποιήσει για να προσδιοριστεί η θέση.

Παράδειγμα : $GPGSV,2,1,08,01,40,083,46,02,17,308,41,12,07,344,39,14,22,228,45*75

Όπου :

GSV Satellites in View

Αριθμός παραγράφων για πλήρη δεδομένα

1 Παράγραφος 1 από τις 2

08 Αριθμός δορυφόρων που βρέθηκαν

01 PRN του δορυφόρου

40 Ύψος, σε μοίρες

083 Αζιμούθιο, σε μοίρες

46 SNR – όσο πιο μεγάλο τόσο καλύτερο

*75 Checksum (πάντα ξεκινάει με *)

- **RMC – Recommended Minimum**

Περιέχει τα απαραίτητα δεδομένα Θέσης, Ταχύτητας και Ώρας.

Παράδειγμα :

$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A

Όπου :

RMC Recommended Minimum C

123519 Ώρα 12:35:19 UTC

A Κατάσταση A = Active ή V = Void

4807.038,N Γεωγραφικό Πλάτος 48° και 07.038’ προς Βορρά

01131.000,E Γεωγραφικό Μήκος 11° και 31.000’ προς Ανατολάς

022.4 Ταχύτητα στο έδαφος σε μονάδες knots

084.4 Γωνία τροχιάς σε μοίρες

230394 Ημερομηνία, 23 Μαρτίου 1994

003.1,W Μαγνητική απόκλιση

*6A Checksum (πάντα ξεκινάει με *)

Για την έκδοση 2.3 του NMEA, υπάρχει ένα καινούργιο πεδίο στην παράγραφο RMC ακριβώς πριν το checksum.

Κεφάλαιο 3
Προδιαγραφές και κατασκευή του GPS Logger

3.1 Λειτουργίες

Αντικείμενο του λογισμικού που θα κατασκευάσουμε είναι η λήψη των βασικών παραμέτρων θέσης (θέση, ταχύτητα, ώρα) σημείων μιας διαδρομής, με δεδομένα που λαμβάνονται από το GPS module, και η αποθήκευσή τους στην κάρτα microSD σε κατάλληλη μορφή. Παράλληλα, θα υλοποιήσουμε μια λειτουργία ώστε με το πάτημα ενός κουμπιού να αποθηκεύεται η τρέχουσα θέση μας, η οποία μπορεί να είναι ένα σημείο ενδιαφέροντος ή γενικά μια θέση που επιθυμούμε να σημειώσουμε. Επίσης, ο χρήστης, αν δεν επιθυμεί να λαμβάνονται μετρήσεις συνέχεια από το GPS, μπορεί να ορίσει ο ίδιος την συχνότητα με την οποία θέλει να γίνονται λήψεις δεδομένων. Αφού ληφθούν τα σημεία της διαδρομής θα δημιουργείται ένα αρχείο KML στην μικρο-κάρτα,

το οποίο αφού ανακτηθεί αργότερα από την κάρτα, θα παρουσιάζει στο Google Earth την αποτύπωση της διαδρομής με γραφικά στοιχεία.

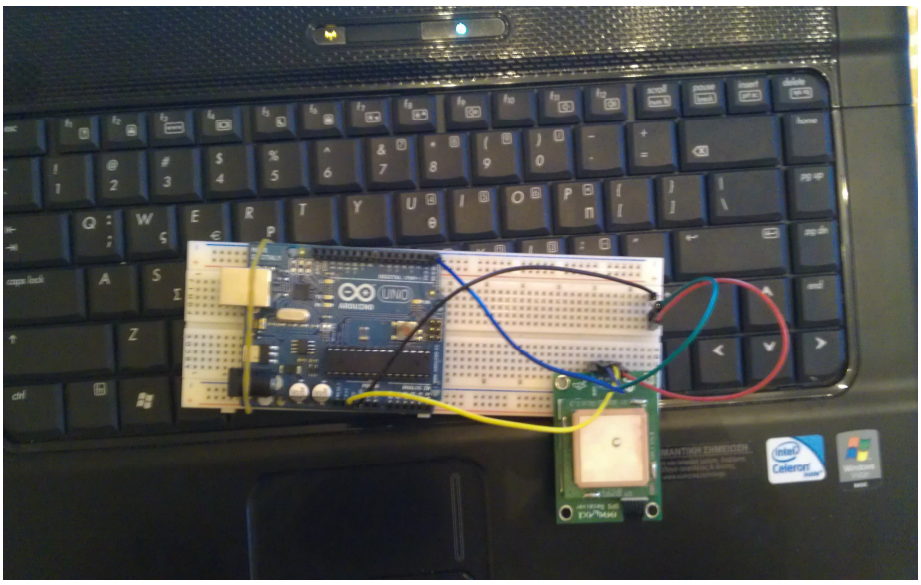
Οι παράμετροι των σημείων (Θέση, Ταχύτητα, Ώρα) λαμβάνονται από το αλφαριθμητικό GPRMC προτύπου NMEA0183, με κατάλληλη επεξεργασία. Τα αλφαριθμητικά GPRMC αποθηκεύονται και αυτά σε αρχείο στην αρχική τους μορφή.

3.2 Δομή και κατασκευή της συσκευής

Η συσκευή αποτελείται από το Arduino UNO, την microSD module, το GPS module, ένα κουμπί και το απαιτούμενο breadboard, μέσω του οποίου γίνεται η συνδεσμολογία στο Arduino.

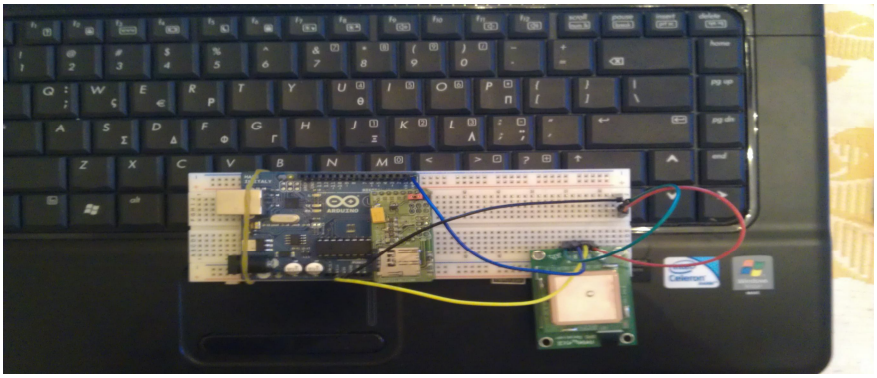
Η συνδεσμολογία της πλατφόρμας αφορά τόσο τον δέκτη GPS module της Parallax, όσο το μικροεπεξεργαστή της κάρτας. Αξίζει να αναφερθεί ότι η πλατφόρμα επεξεργάζεται και εκμεταλλεύεται τις πληροφορίες σε βαθμό «χαμηλού επιπέδου» νοητικών διεργασιών (επεξεργασία μόνο αριθμών και ονομάτων) και εισέρχεται σε συγκεκριμένες περιπτώσεις στη φάση των πολύπλοκων συνδυασμών στοιχείων και εξειδικευμένων εφαρμογών για μεγάλους επιστημονικούς κλάδους, κάτι που αντιστοιχεί σε «υψηλού επιπέδου» νοητικές διεργασίες.

Το GPS module τοποθετήθηκε, αρχικά, στο breadboard και συνδέθηκε με τα κατάλληλα καλώδια με την πλατφόρμα (Εικόνα 3.1). Συγκεκριμένα, έγινε σύνδεση των ψηφιακές εξόδους /RAW και GND του αισθητήρα με τις ψηφιακές εξόδους pin GND. Οι ψηφιακές είσοδοι VCC και SIO του αισθητήρα συνδέθηκαν με τις εξόδους 5V power και 0 αντίστοιχα, που βρίσκονται στο Arduino, ώστε να παρέχεται η απαραίτητη τροφοδοσία για τη λειτουργία και χρήση και των δύο.



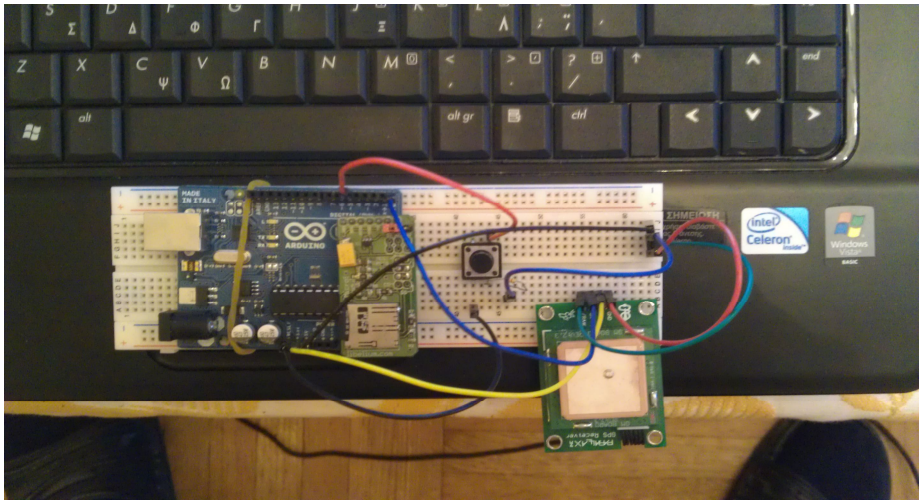
Εικόνα 3.1 Arduino UNO με GPS module

Ο επεξεργαστής της μικροκάρτας έχει τη δυνατότητα σύνδεσης με δύο διαφορετικούς τρόπους. Άξιο αναφοράς είναι ότι ο συγκεκριμένος επεξεργαστής μπορεί να συνδεθεί απευθείας με το arduino και με τους δύο τρόπους. Συγκεκριμένα, μπορεί να συνδεθεί απευθείας είτε με τις ψηφιακές εξόδους της πλατφόρμας pin 8 έως pin 13, είτε με τις ψηφιακές εξόδους ICSP. Στην πρώτη περίπτωση τροφοδοτείται εξ ολοκλήρου από το pin 8. Η μοναδική διαφορά της συνδεσμολογίας των προαναφερθέντων τρόπων είναι η μετακίνηση ενός jumper που βρίσκεται στην εξωτερική πλευρά. Επιλέχθηκε η σύνδεση στην ψηφιακή έξοδο ICSP ώστε να παραμένουν ελεύθερα περισσότερα pins σε περίπτωση που στο μέλλον χρησιμοποιηθούν με κάποιο τρόπο για να βελτιωθεί η εφαρμογή (Εικόνα 3.2).



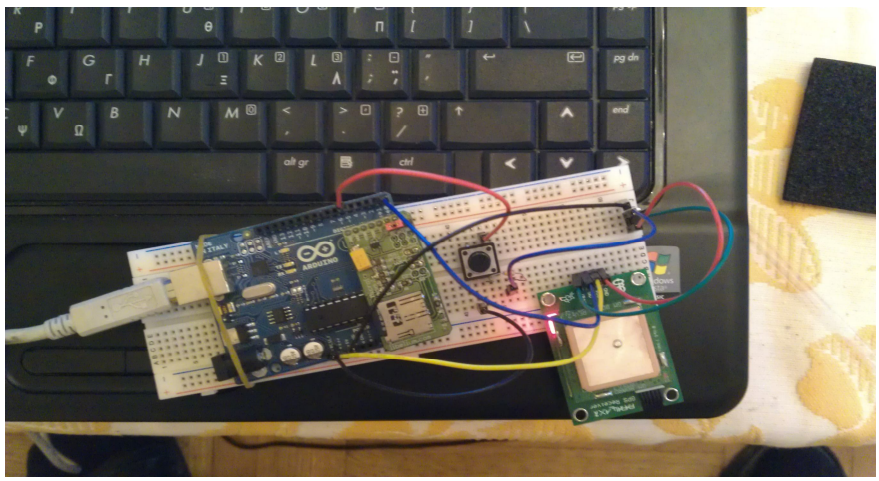
Εικόνα 3.2 Arduino UNO με GPS module και microSD module

Το κουμπί συνδέεται με 3 καλώδια προς την πλατφόρμα, που συνδέουν το κουμπί από το pin 2 με την τροφοδοσία 3.3V και τον αντιστάτη με την γείωση GND (Εικόνα 3.3). Η λειτουργία του κουμπιού, ως παράδειγμα προγραμματισμού, αναλύθηκε με χρήση και του κατάλληλου πηγαίου κώδικα στο Κεφάλαιο 2.1.14. Η διαφορά είναι ότι το κουμπί τροφοδοτείται με 3.3V του Arduino UNO, αντί 5V στο παράδειγμα, επειδή η τροφοδοσία 5V χρησιμοποιείται από το GPS module.



Εικόνα 3.3 Arduino UNO με GPS module, microSD module και button

Τέλος, συνδέεται το Arduino UNO με μια θύρα USB του υπολογιστή (Εικόνα 3.4).



Εικόνα 3.4 Arduino UNO με GPS module, microSD module και button με τροφοδοσία

3.3 Παράμετροι λειτουργίας και ροή εργασιών

Αρχικά, όταν το gps logger έχει κατασκευαστεί, πρέπει να βεβαιωθούμε ότι μέσα στην κάρτα microSD βρίσκονται τα απαραίτητα αρχεία. Το πρώτο και βασικότερο αρχείο είναι το params.txt το οποίο περιέχει τις παραμέτρους με τις οποίες θα λειτουργήσει ο gps logger. Αυτές είναι δύο. Η πρώτη έχει την ονομασία «delay» και η τιμή της προσδιορίζει τον αριθμό των milliseconds (1s = 1000ms) που θα γίνεται προσωρινή παύση μετά από κάθε μέτρηση. Η αρχικοποιημένη τιμή αυτής της παραμέτρου είναι 0 (delay = 0). Η δεύτερη παράμετρος, στην δεύτερη γραμμή του αρχείου ονομάζεται «logname» και η τιμή της προσδιορίζει το όνομα του αρχείο στο οποίο θέλουμε να αποθηκευτούν τα σημεία της διαδρομής. Η τιμή της παραμέτρου logname αν δεν την αλλάξει ο χρήστης είναι 0 (logname = 0) και το αρχείο θα έχει το όνομα logs.txt. Η επέκταση «.txt» του αρχείου προστίθεται από το πρόγραμμα οπότε ο χρήστης αρκεί να γράψει στη θέση του 0 ένα όνομα πχ. «mylogs». Το όνομα του αρχείου μπορεί να έχει εως και 20 χαρακτήρες. Να σημειωθεί ότι πρέπει να δοθεί ιδιαίτερη προσοχή στον τρόπο που είναι δομημένοι οι παράμετροι και οι τιμές τους στο αρχείο ώστε να διαβαστούν σωστά από το πρόγραμμα. Για παράδειγμα, ακριβώς πριν και μετά το σύμβολο «=» υπάρχει ένα κενό (space).

Ένα παράδειγμα του αρχείου καταγραφής params.txt δίνεται στο παρακάτω σχήμα 3.1:

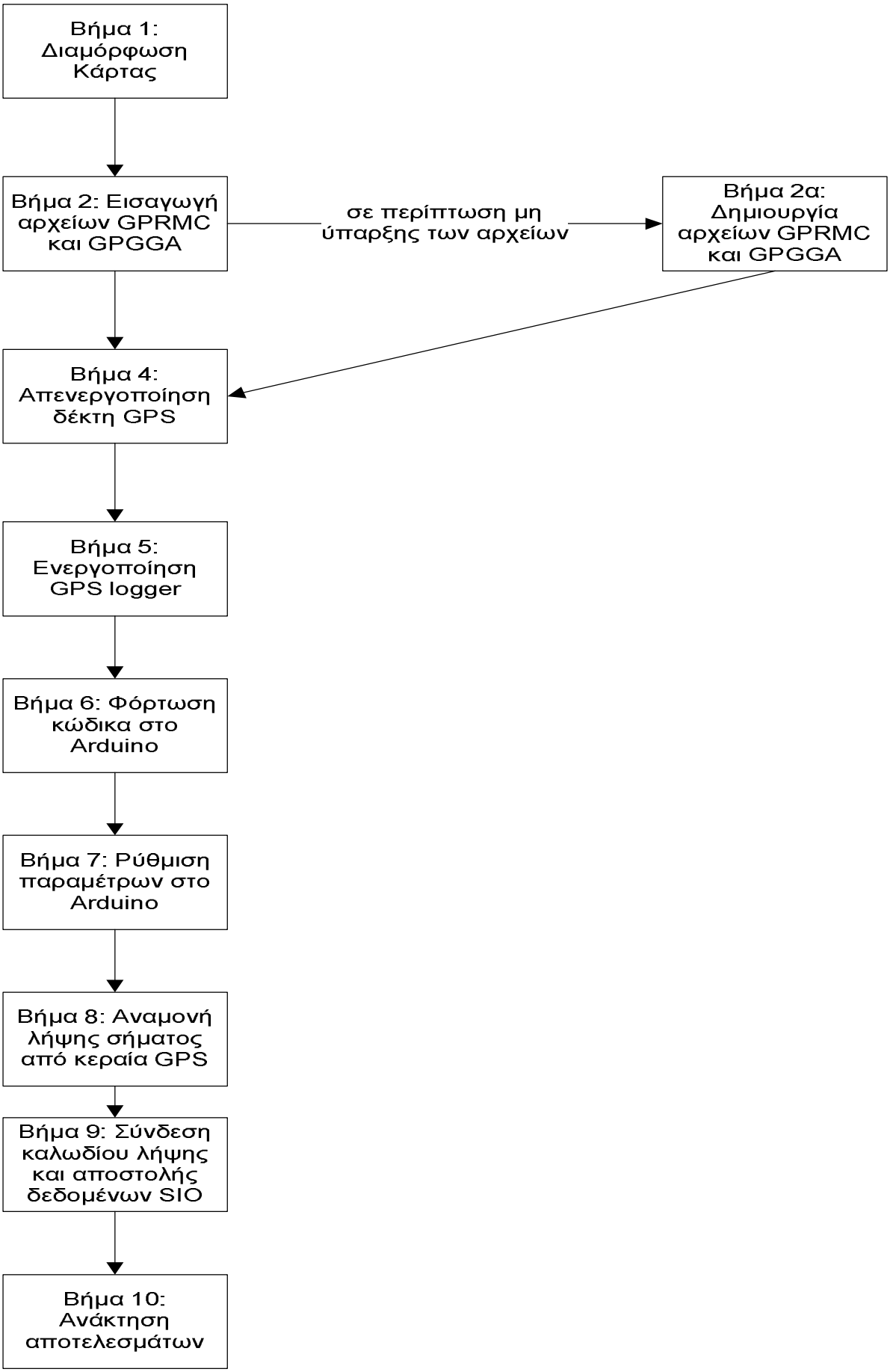
```
1000
logname
```

Σχήμα 3.1 Ενδεικτικά περιεχόμενα αρχείου params.txt

Πριν την εισαγωγή οποιουδήποτε αρχείου μέσα στην μικρο-κάρτα, πρέπει να γίνει διαμόρφωση (format) της κάρτας. Ύστερα εισάγονται τα αρχεία GPRMC.txt και GPGGA.txt, μέσα στα οποία αποθηκεύονται τα αλφαριθμητικά των αντίστοιχων προτύπων NMEA0183, RMC και GGA. Αν αυτά τα αρχεία δεν υπάρχουν τότε το πρόγραμμα τα δημιουργεί. Αν τα αρχεία .txt δεν είναι κενά, τα νέα δεδομένα αποθηκεύονται χωρίς να διαγραφούν τα προηγούμενα.

Όταν βεβαιωθούμε ότι όλα τα παραπάνω έχουν γίνει σωστά μπορούμε να θέσουμε τον gps logger σε λειτουργία. Στην αρχή απενεργοποιούμε προσωρινά τον δέκτη GPS, προκειμένου να μην λαμβάνει άχρηστα δεδομένα μέχρι να ξεκινήσουμε την διαδρομή. Αυτό γίνεται απλά, αποσυνδέοντας το καλώδιο της ψηφιακής εξόδου SIO (pin 3) του GPS module από την ψηφιακή έξοδο pin 0 (RX) του Arduino Uno. Ανοίγουμε το περιβάλλον του Arduino (arduino.exe) και φορτώνουμε τον κώδικα που παρατίθεται στην επόμενη παράγραφο (File -> Upload to I/O board ή πατώντας το κουμπί Upload). Πατάμε στο κουμπί με την αναγραφή «Serial Monitor» δεξιά, επιλέγουμε baud rate 4800bps και περιμένουμε να φορτωθεί η κάρτα (initialize) microSD στην πλατφόρμα. Εμφανίζεται το σχετικό μήνυμα στην οθόνη σειριακής επικοινωνίας του περιβάλλοντος Arduino, Serial Monitor (card initialized) καθώς και η τιμή της παραμέτρου “delay” (Params.txt delay =). Ύστερα βγαίνουμε σε εξωτερικό χώρο, όπου η κεραία του GPS module θα μπορεί να λάβει σήμα από δορυφόρους, και περιμένουμε μέχρι το κόκκινο λαμπάκι του δέκτη να σταματήσει να αναβοσβήνει και να μείνει μόνιμα κόκκινο. Μόλις γίνει αυτό, συνδέουμε το καλώδιο λήψης και αποστολής δεδομένων SIO από το GPS module στο Arduino και ξεκινάμε την διαδρομή που θέλουμε να κάνουμε. Στην οθόνη της εφαρμογής Arduino θα βλέπουμε παράλληλα όλα τα αλφαριθμητικά (GPGGA, GPGSV, GPGSA, GPRMC) του προτύπου NMEA0183, ενώ μόλις αποθηκεύεται κάποιο σημείο θα γίνεται η σχετική ενημέρωση με ένα μήνυμα. Στα σημεία διαδρομής που μας ενδιαφέρουν ιδιαίτερα και θέλουμε να σημειώσουμε, πατάμε το κουμπάκι συνεχόμενα για λίγα δευτερόλεπτα ώστε να αναγραφεί σίγουρα η σχετική ένδειξη «1» στο αρχείο.

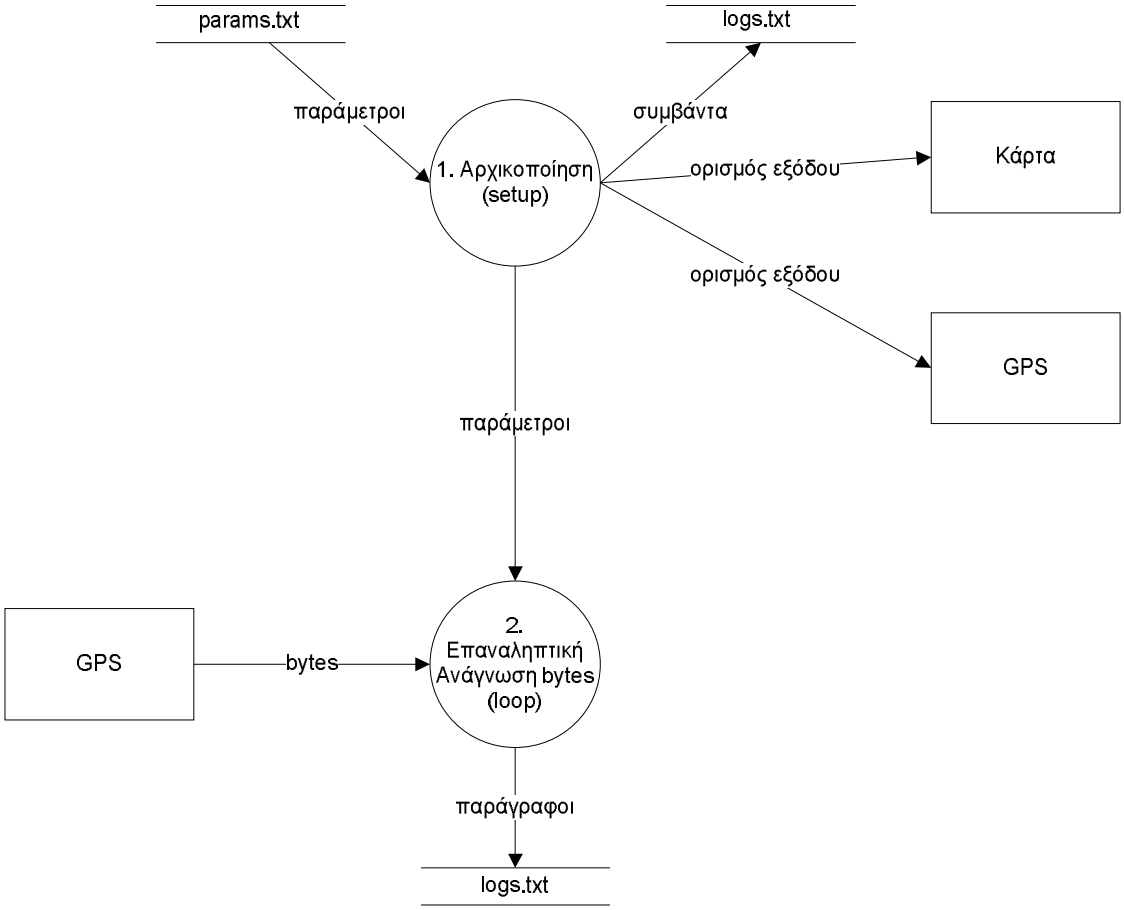
Η παραπάνω διαδικασία αποτυπώνεται στο παρακάτω σχήμα: 3.2:



Σχήμα 3.2 Ροή εργασιών

3.4 Διάγραμμα ροής δεδομένων

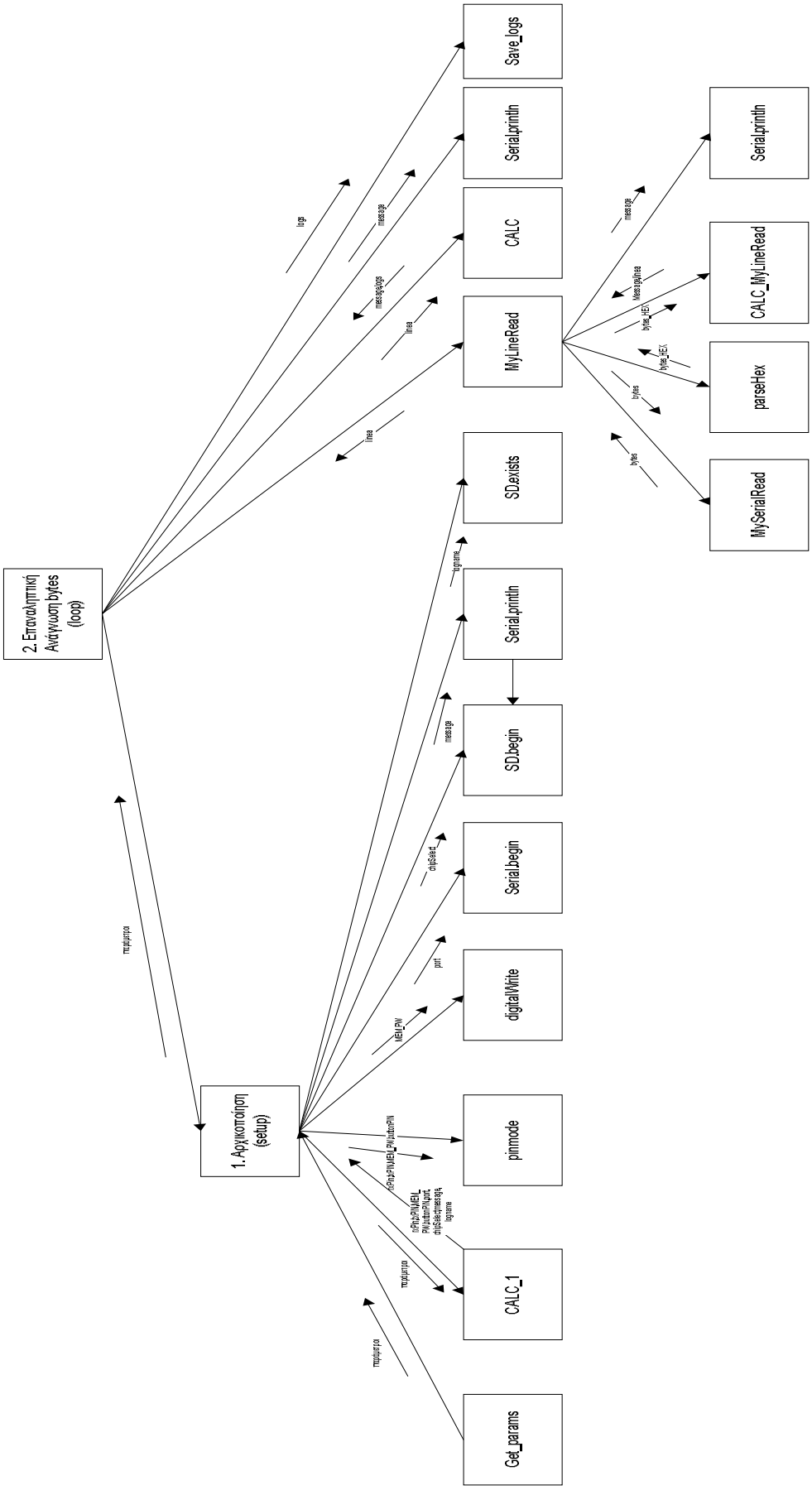
Το διάγραμμα ροής δεδομένων της εφαρμογής φαίνεται στο παρακάτω σχήμα 3.3:



Σχήμα 3.3 Διάγραμμα Ροής Δεδομένων

3.5 Διάγραμμα δομής προγράμματος

Το διάγραμμα δομής προγράμματος της εφαρμογής φαίνεται στο παρακάτω σχήμα 3.5:



Σχήμα 3.4 Διάγραμμα Δομής Προγράμματος

3.6 Λογισμικό της συσκευής

```
#include <string.h>
#include <ctype.h>
#include <stdlib.h>
#include <SD.h>

#define MEM_PW 8

#define BUFFSIZE 300    // SIZE OF INPUT BUFFER IN CHARS

#define TIMEWAIT 100    // TIME TO WAIT IF NO DATA ON SERIAL PIN (ms)

char num[10];
char simeiostr[8]="0";
char bstate[0];
int delta = 0;        // Pointer for file reading
//int ledPin = 13;                  // LED test pin
int rxPin = 0;                    // RX PIN [SIO from GPS module]
int txPin = 1;                    // TX TX

int GPS_rx=2;
int GPS_tx=3;

char mydel[10];
long p_delay=0;      // Delay initialized by the user in params.txt
char p_logname[20]="0";    // Logs filename stored in params.txt
char templog[20];      // Just a temporary storage array

char byteGPS=-1;
```

```
char linea[BUFFSIZE];
char mylog[50];

int giota=0;
int bien=0;

char GPRMCstr[6] = "GPRMC";
char GPGGAstr[6] = "GPGGA";
char Whatstr[6];
char WhatFile[10];
int runner=0;

int komma[13];    // Position of seperator ',' in string
int buttonState=0;    // If button is pushed = 1
int gprmcn=0;

long sum;

const int chipSelect = 10;    // SS microSD's pin [ works if microsd is connected to ICSP too]
const int buttonPin = 5;      // Button Pin

const int NAV_RATE_POS = 8;    // Position of delay in params.txt (delay = )

File dataFile;    // Where all logs will be loged
File paramFile;   // Doing params.txt operations
File specialFile; // GPGGA.txt or GPRMC.txt (depends on string received)

// ----------- Setup start ------------
void setup() {
  pinMode(rxPin, INPUT);
```

```
pinMode(txPin, OUTPUT);

pinMode(MEM_PW, OUTPUT);
pinMode,(buttonPin, INPUT);
digitalWrite(MEM_PW, HIGH);
Serial.begin(4800);           // with main computer


for (int i=0;i<BUFFSIZE;i++){       // Initialize a buffer for received data
  linea[i]=' ';
}
for (int i=0;i<50;i++){
  mylog[i]=' ';
}

Serial.print("Initializing SD card...");
// see if the card is present and can be initialized:
if (!SD.begin(chipSelect)) {
  Serial.println("Card failed, or not present");
  // don't do anything more:
  return;
}
Serial.println("card initialized.");

paramFile = SD.open("params.txt", FILE_READ);

// Find what we want in params.txt
if (paramFile){

  // First take delay (ms)
  if (paramFile.seek(NAV_RATE_POS)==1)
```

```
  {
    Serial.print("Params.txt delay = ");
    mydel[delta] = paramFile.read();
    while (mydel[delta]!=' ')
    {
      delta++;
      mydel[delta] = paramFile.read();
    }
    p_delay = atol(mydel);
    Serial.print(p_delay);
  }
  Serial.println();

  // Read two bytes ("= ") to reach filename (8+2)

  paramFile.read();
  paramFile.read();

  Serial.println();
  int deiktis=0;
  Serial.print("Params.txt logname = ");
  templog[deiktis] = paramFile.read();
  while (templog[deiktis]!=EOF)
  {
   p_logname[deiktis] = templog[deiktis];
   deiktis++;
   templog[deiktis] = paramFile.read();
  }
  Serial.println(p_logname);
  paramFile.close();
}
```

```
else
{
  Serial.println("error opening params.txt");
}

// If "logname = 0" in params.txt then use default [logs.txt]
if (strcmp(p_logname,"0") == 0)
{
  strcpy(p_logname,"logs.txt");
}
else
{
  strcat(p_logname,".txt");
  Serial.println();
  Serial.print(p_logname);

  if (SD.exists(p_logname)){
    Serial.println(" already exists.");
  }
  else
  {
  Serial.println(" doesn't exist.");
  Serial.print("Creating ");
  Serial.println(p_logname);
  dataFile = SD.open(p_logname,FILE_WRITE);
  Serial.print("dataFile = ");
  Serial.println(dataFile);
  if (dataFile)
  {
    Serial.print(p_logname);
    Serial.print(" created successfully!");
```

```
      dataFile.close();
    }
    else
    {
      Serial.print("Couldnt create ");
      Serial.print(p_logname);
      Serial.print(" file.");
    }
  }
  }
  Serial.print("The logs filename is : ");
  Serial.print(p_logname);
  Serial.println();
}

// ---------- Setup ends -----------

// ---------- Functions -----------

// read a Hex value and return the decimal equivalent
// ======================================================
uint8_t parseHex(char c) {
  if (c < '0')
    return 0;
  if (c <= '9')
    return c - '0';
  if (c < 'A')
    return 0;
  if (c <= 'F')
    return (c - 'A')+10;
}
```

```
// ========================================================
// Reads a byte from gps receiver

char MySerialRead(int waittime) {
  char x;

  do {
    x=Serial.read();
    if (x==-1) delay(waittime);
  } while (x==-1);

  return x;
}

// ========================================================
// Reads and stores the full NMEA string (GGA,GSA,GSV or RMC)

int MyLineRead(char* b) {

  for (int j=0;j<BUFFSIZE;j++) b[j]=' ';

  int i=0;
  long tsum=0;

  char a, n1, n2;

  while (MySerialRead(TIMEWAIT) != '$');  // Wait for a dollar...


  do {
    if (i>=BUFFSIZE) return -9;   // should NEVER occur
```

```
    a=MySerialRead(TIMEWAIT);
    if (a=='*')
    {  // ready to read checksum

      n1=MySerialRead(TIMEWAIT);
      if (((n1<'0') || (n1>'9')) && ((n1<'A') || (n1>'F'))) return -99;   //
shound not occur
      n2=MySerialRead(TIMEWAIT);
      if (((n2<'0') || (n2>'9')) && ((n2<'A') || (n2>'F'))) return -99;   //
shound not occur
      // Serial.print(">?n1>"); Serial.print(n1); Serial.print("<  >?n2>");
Serial.print(n2);Serial.print("< ");
      tsum = parseHex(n1) * 16;
      tsum += parseHex(n2);

      for (int t=0; t < i; t++)
      {
        tsum ^= b[t];
      }

      if (tsum!=0)
      {
        Serial.println();
        Serial.println("Checksum mismatch!!\n");
        b[i]='\0';
        return -999;
      }
      else
      {
        b[i]='\0';
        break;
```

```
      }
    }
    else
    {
      b[i++]=a;
    }

  } while(a!=13);

  return 0;
}

// ========================================================

void loop() {

  // DELAY HERE?
  //delay(p_delay);

  int ReadAttemptResult=MyLineRead(linea);   // Read and check...

  if (ReadAttemptResult) {
    Serial.print("\nERROR: MyLineRead returns
");Serial.print(ReadAttemptResult);Serial.println();
    return;
  }
  else
  {
    // NOW do whatever it takes with linea...

    Serial.print(linea);
```

```
    Serial.println();
    // Define string type
    for (int j=0;j<5;j++) Whatstr[j] = linea[j];
    giota = 0;
    runner = 0;

    // Is this the right string? (GPRMC)
    if (strcmp(Whatstr,GPRMCstr) == 0) {
      gprmcn++;
      // If yes, then make our log : [Number of
point,Time,Button,X,direction,Y,direction,velocity]

      // First find positions of separators (,) in string
      for (int k=0;k<BUFFSIZE;k++)
      {
        if (linea[k]==',') {
          komma[giota]=k;
          giota++;
        }
      }

      // Copy number of point
      ltoa(gprmcn,simeiostr,10);
      while (runner<strlen(simeiostr))
      {
        mylog[runner]=simeiostr[runner];
        runner++;
      }
      mylog[runner] = ',';
      runner++;
```

```
// Copy Time in UTC (HhMmSs)
for (int j=komma[0];j<komma[1];j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}

// See if button is pushed and copy
// 1 = Pushed, 0 = Not Pushed
buttonState = digitalRead(buttonPin);
ltoa(buttonState,bstate,10);
mylog[runner] = bstate[0];

runner++;
mylog[runner] = ',';
runner++;

// Copy Longitude (X)
for (int j=komma[4];j<komma[5];j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}

// Copy Direction W/E
for (int j=komma[5];j<komma[6];j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}
```

```
// Copy Latitude (Y)
for (int j=komma[2];j<komma[3];j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}

// Copy Direction N/S
for (int j=komma[3];j<komma[4];j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}

// Copy Velocity in knots
for (int j=komma[6];j<(komma[7]-1);j++)
{
  mylog[runner] = linea[j+1];
  runner++;
}

// Save the log
dataFile = SD.open(p_logname, FILE_WRITE);
if (dataFile){
  dataFile.println(mylog);
  dataFile.close();
}
else
{
  Serial.print("error opening ");
  Serial.println(p_logname);
```

```
      }


      for (int i=0;i<runner;i++)
      {
        mylog[i]=' ';
      }


    }


    // Finally if it was GPRMC or GPGGA
    // save it in the appropriate file
    if (strcmp(Whatstr,GPRMCstr) == 0 || strcmp(Whatstr,GPGGAstr) == 0)
    {
      strcpy(WhatFile,Whatstr);
      strcat(WhatFile,".txt");
      specialFile = SD.open(WhatFile,FILE_WRITE);
      if (specialFile)
      {
        specialFile.println(linea);
        specialFile.close();
      }
    }


  }
}
```

3.7 Ανάλυση Πηγαίου Κώδικα

Ο Πηγαίος Κώδικας αποτελείται από μια σειρά από συναρτήσεις, η λειτουργία των οποίων δίνεται αναλυτικά πιο κάτω ως εξής:

3.7.1 Συνάρτηση setup

Η συνάρτηση setup χρησιμοποιείται για τον καθορισμό όλων των επιμέρους παραμέτρων που σχετίζονται με το Arduino και είναι υποχρεωτική σύμφωνα με το μοντέλο προγραμματισμού του Arduino. Πιο συγκεκριμένα, ορίζονται όλοι οι απαραίτητοι ηλεκτρολογικοί παράμετροι που καθορίζουν την λειτουργία των στοιχείων που συνδέονται στο Arduino. Στις παραμέτρους αυτούς περιλαμβάνονται οι ψηφιακές έξοδοι (pins) των:

- GPS module
- Κάρτας και
- Κουμπιού.

Αρχικά, ορίζονται οι είσοδοι και οι έξοδοι και επιχειρείται σύνδεση με την κάρτα. Στη συνέχεια διαβάζονται οι παράμετροι που είναι αποθηκευμένοι στο αρχείο params.txt. Οι τιμές αυτές επηρεάζουν την λειτουργία του προγράμματος, ανάλογα με τις επιλογές του χρήστη. Η σημαντικότερη από τις παραμέτρους αυτές είναι η καθυστέρηση σε ms. Επίσης, διαβάζεται από το αρχείο το όνομα του αρχείου καταγραφής (logfile), το οποίο αν δε βρεθεί ορίζεται ότι είναι το logs.txt, που είναι το προκαθορισμένο αρχείο καταγραφής.

3.7.2 Συνάρτηση parseHex

Η συνάρτηση parseHex είναι μία βοηθητική συνάρτηση και χρησιμοποιείται για την μετατροπή μίας δεκαεξαδικής τιμής ενός χαρακτήρα στο δεκαδικό αριθμητικό σύστημα. Ως όρισμα παίρνει το χαρακτήρα σε δεκαεξαδική τιμή και επιστρέφει το δεκαδικό αντίστοιχο αριθμό.

Ένα παράδειγμα κλήσης της συνάρτησης είναι το εξής:

tsum = parseHex(n1) * 16;

όπου:

n1: ο χαρακτήρας σε 16 μορφή

tsum: η αντίστοιχη δεκαδική τιμή

3.7.3 Συνάρτηση MySerialRead

Η συνάρτηση MySerialRead χρησιμοποιείται για την ανάγνωση ενός byte που έλαβε ο δέκτης GPS. Δέχεται ως είσοδος το χρόνο καθυστέρησης και επιστρέφει ένα χαρακτήρα. Αν το byte που θα διαβαστεί έχει την τιμή –1 που είναι λάθος, τότε μετά από ένα χρονικό διάστημα ίσο με την καθυστέρηση που δόθηκε ως είσοδος, διαβάζεται αμέσως μετά κι άλλο byte. Η διαδικασία επαναλαμβάνεται μέχρι να ληφθεί ένα σωστό byte από το GPS.

Ένα παράδειγμα κλήσης της συνάρτησης είναι το εξής:

a=MySerialRead(TIMEWAIT);

όπου:

TIMEWAIT: ο χρόνος καθυστέρησης

a: ο χαρακτήρας που επιστρέφει

3.7.4 Συνάρτηση MyLineRead

Η συνάρτηση MyLineRead διαβάζει και αποθηκεύει όλα τα bytes που λήφθηκαν από το δέκτη GPS στην πάροδο του χρόνου, τα οποία ορίζουν μία παράγραφο του προτύπου NMEA0183 και τα αποθηκεύει σε μία μεταβλητή. Το τέλος κάθε παραγράφου του προτύπου NMEA0183 προσδιορίζεται από το checksum, δηλαδή τα δύο τελευταία bytes μετά τον αστερίσκο *. Εφόσον έχει ληφθεί μία ολοκληρωμένη παράγραφος NMEA0183

από το δέκτη GPS, τότε γίνεται έλεγχος του checksum κάνοντας την πράξη XOR (eXclusive OR) όλων των αλφαριθμητικών που παρεμβάλλονται μεταξύ του συμβόλου $ (αρχή παραγράφου) και του * (τέλος δεδομένων - αρχή checksum). Αν ο έλεγχος είναι σωστός, αυτό σημαίνει ότι η παράγραφος είναι έγκυρη και η συνάρτηση επιστρέφει την ολοκληρωμένη παράγραφο.

Ένα παράδειγμα κλήσης της συνάρτησης είναι το εξής:

ReadAttemptResult=MyLineRead(linea);

όπου:

linea: οι χαρακτήρες που θα διαβαστούν

ReadAttemptResult: αποτέλεσμα της ανάγνωσης

3.7.5 Συνάρτηση loop

Στη συνέχεια του αλγορίθμου, υπάρχει η συνάρτηση loop, η οποία είναι και η βασική επαναληπτική διαδικασία του αλγορίθμου και είναι υποχρεωτική σύμφωνα με το μοντέλο προγραμματισμού του Arduino. Η συνάρτηση loop, η οποία εκτελείται συνέχεια όσο τροφοδοτείται το arduino, καθορίζει την λειτουργία της εφαρμογής. Στην αρχή της συνάρτησης γίνεται προσπάθεια να διαβαστεί μία ολοκληρωμένη και έγκυρη παράγραφο NMEA0183 μέσω της συνάρτησης MyLineRead. Γίνεται δηλαδή κλήση της συνάρτησης MyLineRead και αν αυτή επιστρέψει τη τιμή 0, τότε σημαίνει ότι διαβάστηκε μία ολοκληρωμένη και έγκυρη παράγραφο NMEA0183. Σε αυτήν την περίπτωση, η παράγραφος αυτή αποθηκεύεται στην μεταβλητή linea.

Στη συνέχεια γίνεται ο έλεγχος αν η παράγραφος που ελέγχθηκε είναι η GPRMC. Σε περίπτωση σωστού ελέγχου γίνεται επεξεργασία της παραγράφου RMC. Η συνάρτηση αγνοεί τις υπόλοιπες παραγράφους (GGA, GSA, GSV).

Επίσης, η συνάρτηση ελέγχει την κατάσταση του κουμπιού για να διαπιστώσει αν το κουμπί έχει πατηθεί ή όχι. Η συνάρτηση αυτή χρησιμοποιεί το κόμμα (,), το οποίο

διαχωρίζει τα δεδομένα της παραγράφου RMC του προτύπου NME, για να πάρει τις απαραίτητες τιμές που μας ενδιαφέρουν και είναι οι εξής:

Θέση, Ώρα και Ταχύτητα.

Στη συνέχεια αποθηκεύει τις παραπάνω τιμές σε μία νέα μεταβλητή με την μορφή που φαίνεται στο παρακάτω παράδειγμα:

Παράγραφος: 110,070656,1,02346.8633,E,3758.4931,N,001.9

Όπου:

110: Αριθμός σημείου (ξεκινάει από 1)

070656: Ώρα (07:06:56 UTC)

1: Button mark (1 = Πατήθηκε, 0 = Δεν πατήθηκε)

02346.8633: X - Γεωγραφικό μήκος (23 μοίρες και 46.8633 πρώτα λεπτά)

E: Προσανατολισμός μήκους (E = East, W = West)

3758.4931: Y - Γεωγραφικό πλάτος (37 μοίρες και 58.4931 πρώτα λεπτά)

N: Προσανατολισμός πλάτους (N = North, S = South)

001.9: Ταχύτητα σε μονάδες knots (1 ναυτικό μίλι ανά ώρα => 1 knot = 1.852km/h)

Η νέα αυτή παράγραφος αποθηκεύεται στο αρχείο καταγραφής (default: log.txt).

Στη συνέχεια, παρατίθεται ένα δείγμα 50 σημείων που μετρήθηκαν και αποθηκεύτηκαν στο αρχείο καταγραφής όπως φαίνεται παρακάτω:

1,070503,0,02346.8279,E,3758.5302,N,000.0

2,070504,0,02346.8284,E,3758.5298,N,000.0

3,070505,0,02346.8287,E,3758.5296,N,000.0

4,070506,0,02346.8289,E,3758.5294,N,000.0

5,070507,1,02346.8292,E,3758.5292,N,000.0

6,070508,1,02346.8295,E,3758.5289,N,000.0

7,070509,1,02346.8297,E,3758.5288,N,000.0

8,070510,1,02346.8297,E,3758.5288,N,000.0

9,070511,0,02346.8297,E,3758.5288,N,000.0

10,070512,0,02346.8297,E,3758.5288,N,000.0

11,070513,0,02346.8273,E,3758.5297,N,000.0

12,070514,0,02346.8257,E,3758.5303,N,000.0

13,070515,0,02346.8244,E,3758.5309,N,000.0

14,070516,0,02346.8236,E,3758.5313,N,000.0

15,070517,0,02346.8230,E,3758.5316,N,000.0

16,070518,0,02346.8149,E,3758.5350,N,004.3

17,070519,0,02346.8168,E,3758.5346,N,001.7

18,070520,0,02346.8182,E,3758.5342,N,001.0

19,070521,0,02346.8203,E,3758.5336,N,004.4

20,070522,0,02346.8218,E,3758.5332,N,003.5

21,070523,0,02346.8233,E,3758.5327,N,003.3

22,070524,0,02346.8233,E,3758.5327,N,000.0

23,070525,0,02346.8233,E,3758.5327,N,000.0

24,070526,0,02346.8233,E,3758.5327,N,000.0

25,070527,0,02346.8234,E,3758.5327,N,000.0

26,070528,0,02346.8254,E,3758.5321,N,002.9

27,070529,0,02346.8254,E,3758.5321,N,000.0

28,070531,0,02346.8255,E,3758.5321,N,000.0

29,070532,0,02346.8273,E,3758.5316,N,002.0

30,070533,0,02346.8274,E,3758.5315,N,000.0

31,070534,0,02346.8274,E,3758.5315,N,000.0

32,070535,0,02346.8274,E,3758.5315,N,000.0

33,070536,0,02346.8275,E,3758.5315,N,000.0

34,070537,0,02346.8275,E,3758.5315,N,000.0

35,070538,0,02346.8275,E,3758.5315,N,000.0

36,070539,0,02346.8275,E,3758.5315,N,000.0

37,070540,0,02346.8276,E,3758.5315,N,000.0

38,070541,0,02346.8297,E,3758.5308,N,003.3

39,070542,0,02346.8297,E,3758.5308,N,000.0

40,070543,0,02346.8318,E,3758.5303,N,003.6

41,070544,0,02346.8334,E,3758.5299,N,003.0

42,070545,0,02346.8351,E,3758.5295,N,003.0

43,070546,0,02346.8368,E,3758.5291,N,003.0

44,070547,0,02346.8362,E,3758.5297,N,001.7

45,070548,0,02346.8377,E,3758.5292,N,002.8

46,070549,0,02346.8394,E,3758.5288,N,002.5

47,070550,0,02346.8408,E,3758.5284,N,002.5

48,070552,0,02346.8423,E,3758.5274,N,002.3

49,070553,0,02346.8441,E,3758.5267,N,003.2

50,070554,0,02346.8460,E,3758.5258,N,004.2

3.7.6 Δεσμευμένες Συναρτήσεις Arduino

Στην ενότητα αυτή παρουσιάζονται οι δεσμευμένες συναρτήσεις Arduino που χρησιμοποιήθηκαν:

pinMode

Περιγραφή

Η συνάρτηση αυτή διαμορφώνει το καθορισμένο pin, έτσι ώστε να συμπεριφέρεται σαν pin εισόδου ή σαν pin εξόδου.

Σύνταξη

pinMode(pin, mode)

Παράμετροι

pin: ο αριθμός του pin του οποίου καθορίζεται η μορφή.

mode: παίρνει τις τιμές INPUT ή OUTPUT. Αλλάζοντας ένα pin από INPUT σε OUTPUT, μεταβάλλεται δραστικά η ηλεκτρική συμπεριφορά του pin.

Pins ορισμένα ως Input

Τα pins του Arduino που ορίζονται με την τιμή INPUT στην συνάρτηση pinMode() ηλεκτρολογικά αναφέρονται ως ευρισκόμενα σε κατάσταση υψηλής αντίστασης. Αυτό σημαίνει ότι τα pins αυτά δεν επηρεάζουν άμεσα την λειτουργία του ηλεκτρικού κυκλώματος. Αυτού του είδους τα pins είναι χρήσιμα για την ανάγνωση δεδομένων ενός αισθητήρα, αλλά όχι και για την τροφοδοσία μιας διόδου LED.

Pins ορισμένα ως Output

Τα pins του Arduino που ορίζονται με την τιμή OUTPUT στην συνάρτηση pinMode() λέγεται ότι βρίσκονται σε κατάσταση χαμηλής αντίστασης. Δηλαδή μπορούν να παρέχουν μία ουσιώδη ποσότητα ρεύματος στα άλλα κυκλώματα. Τα Atmega pins παρέχουν θετική ή αρνητική τάση ύψους έως και 40mA σε άλλα κυκλώματα ή συσκευές. Αυτό τα καθιστά χρήσιμα για να ανάβουν ένα LED αλλά όχι και για την ανάγνωση δεδομένων από αισθητήρες. Τα pins που διαμορφώνονται σαν OUTPUT κινδυνεύουν να καταστραφούν αν βραχυκυκλωθεί η σύνδεση με τη γείωση ή την τροφοδοσία 5V. Η ποσότητα ρεύματος που παρέχεται από ένα pin Atmega, σε συνδυασμό με κάποιο συνδετικό εξάρτημα στο ηλεκτρικό κύκλωμα, είναι αρκετή για να τροφοδοτήσει τις περισσότερες μηχανές ή κινητήρες.

Επιστροφή τιμής

Καμία.

digitalWrite

Περιγραφή

Αποθηκεύει την τιμή HIGH ή LOW σε ένα ψηφιακό pin. Αν το pin έχει προκαθοριστεί ως OUTPUT μέσω της συνάρτησης pinMode(), η τάση του θα διαμορφωθεί σε 5V (ή 3V σε πλακέτες 3.3V) για την τιμή HIGH και 0V (γείωση) για την τιμή LOW.

Αν το pin έχει προκαθοριστεί ως INPUT, γράφοντας την τιμή HIGH, επιτρέπεται η ενεργοποίηση αντίστασης ύψους 20K. Γράφοντας την τιμή LOW απενεργοποιείται η αντίσταση.

Σύνταξη

digitalWrite(pin, value)

Παράμετροι

pin: ο αριθμός του pin

value: HIGH ή LOW

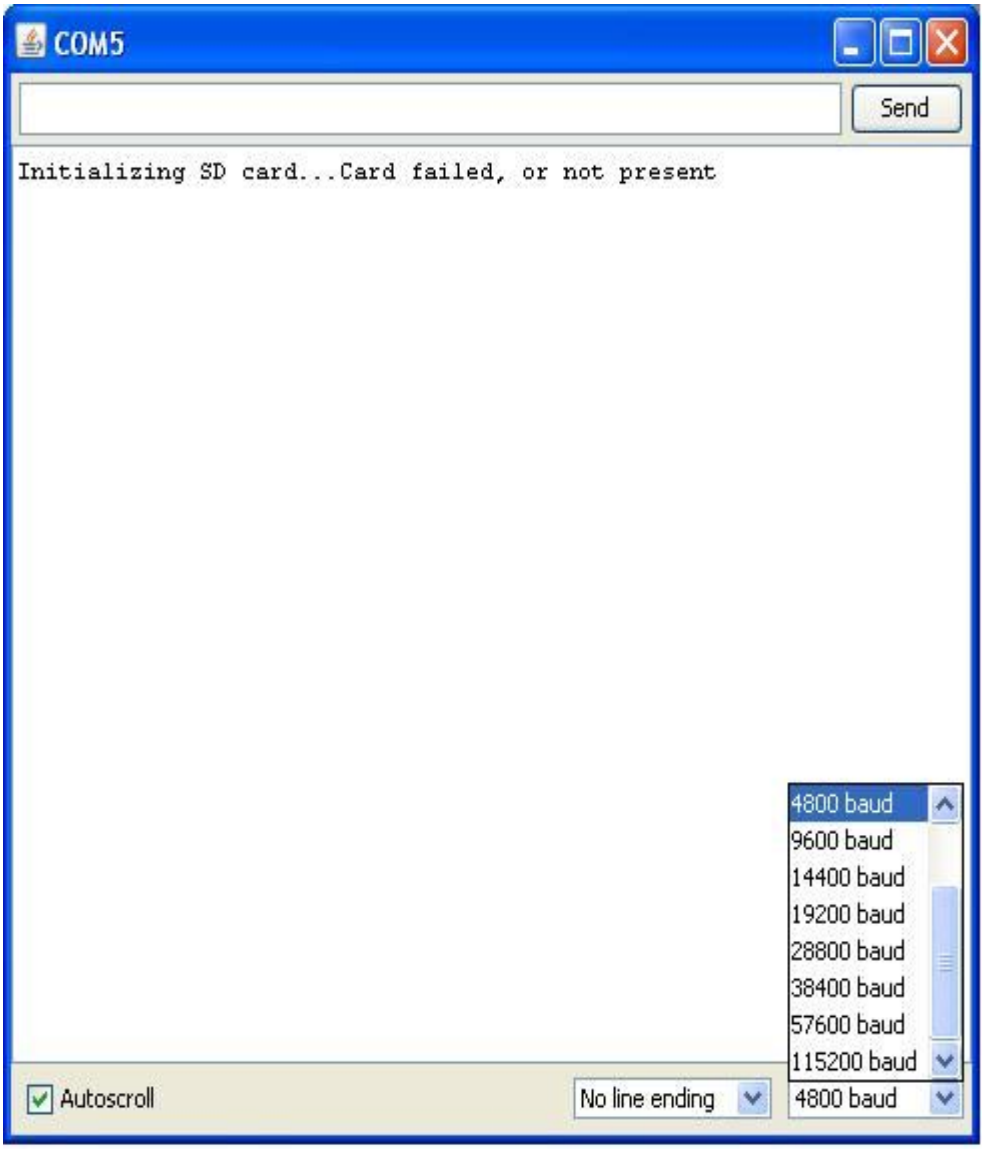
Επιστροφή τιμής

Καμία.

Serial

Η συνάρτηση Serial χρησιμοποιείται για την επικοινωνία μεταξύ Arduino και ενός υπολογιστή ή άλλης συσκευής. Όλες οι πλακέτες Arduino έχουν τουλάχιστον μία σειριακή θύρα (γνωστή ως UART ή USART), που λέγεται πιο απλά Serial (Σειριακό). Επικοινωνεί μέσω των ψηφιακών pin 0 (RX) και 1 (TX) αλλά και με έναν ηλεκτρονικό υπολογιστή μέσω της θύρας USB. Συνεπώς, χρησιμοποιώντας την συνάρτηση Serial, δεν επιτρέπεται να χρησιμοποιηθούν τα pins 0 και 1 σαν ψηφιακές είσοδοι ή έξοδοι.

Στο περιβάλλον Arduino υπάρχει μία σειριακή οθόνη που επιτρέπει την επικοινωνία με την πλακέτα Arduino. Η σειριακή οθόνη ανοίγει πατώντας στο κουμπί Serial monitor στη γραμμή εργαλείων (Εικόνα 3.4), όπου και επιλέγεται η κατάλληλη συχνότητα (baud rate) που χρησιμοποιήθηκε στην συνάρτηση Serial.begin().



Εικόνα 3.4 Serial monitor και επιλογή baud rate

Το Arduino Mega διαθέτει τρεις επιπλέον σειριακές θύρες:

την Serial1 στα pin 19 (RX) και 18 (TX),

την Serial2 στα pin 16 (RX) και 17 (TX) και

την Serial3 στα pin 15 (RX) και 14 (TX).

Serial.begin(datarate)

Περιγραφή

Η συνάρτηση begin θέτει την συχνότητα μεταφοράς δεδομένων για την σειριακή επικοινωνία, σε μορφή bits per second (baud). Για επικοινωνία με τον ηλεκτρονικό υπολογιστή, χρησιμοποιείται μία από τις παρακάτω συχνότητες : 300, 1200, 2400, 4800, 9600, 14400, 28800, 38400, 57600 ή 115200. Μπορούν, επίσης, να καθοριστούν άλλες τιμές, όπως για παράδειγμα, για την επικοινωνία μέσω των pin 0 και 1 με ένα εξάρτημα που επιτρέπει συγκεκριμένες συχνότητες baud.

Παράμετροι

datarate: bits ανα δευτερόλεπτο (baud) σε ακέραια μορφή

Επιστροφή

Καμία

Serial.print(data)

Περιγραφή

Η συνάρτηση αυτή εκτυπώνει δεδομένα στην σειριακή θύρα.

Παράμετροι

data: ακέραιοι τύποι, συμπεριλαμβανομένων χαρακτήρων και floats.

Η εκτύπωση δεκαδικών floats υποστηρίζεται με ακρίβεια δύο δεκαδικών ψηφίων στα δεξιά του δεκαδικού μέρους. Αυτό μπορεί να αλλάξει σύντομα.

Σύνταξη

Η εντολή αυτή μπορεί να πάρει πολλές μορφές.

Serial.print(b) χωρίς καθορισμό του τύπου δεδομένων, εκτυπώνει το b σαν έναν δεκαδικό αριθμό σε ένα αλφαριθμητικό ASCII.

Serial.print(b,DEC) εκτυπώνει τον b σαν έναν δεκαδικό αριθμό στο αλφαριθμητικό σύστημα ASCII (ακριβώς όπως στην παραπάνω εντολή)

Serial.print(b,HEX) εκτυπώνει το b σαν έναν δεκαεξαδικό αριθμό

Serial.print(b,OCT) εκτυπώνει τον b σαν έναν οκταδικό αριθμό.

Serial.print(b,BIN) εκτυπώνει τον σαν έναν δυαδικό αριθμό.

Serial.print(b,BYTE) εκτυπώνει τον b σαν ένα byte, τον χαρακτήρα ASCII δηλαδή που αναπαριστά η τιμή που έχει ο b

Serial.print(str) αν το str είναι ένα string ή ένας πίνακας χαρακτήρων, εκτυπώνεται το str σαν ένα ASCII string

Παράμετροι

b: το byte προς εκτύπωση ή

str: το string προς εκτύπωση

Επιστροφή

Καμία

Serial.println(data)

Περιγραφή

Η συνάρτηση αυτή εκτυπώνει δεδομένα στην σειρακή θύρα, συνοδευόμενα από ένα χαρακτήρα carriage return (ASCII 13 ή «\r») και έναν χαρακτήρα νέας γραμμής (new line feed ASCII 10 ή «\n»). Η εντολή αυτή έχει την ίδια μορφή με την Serial.print().

Παράμετροι

data: ακέραιοι τύποι, συμπεριλαμβανομένων χαρακτήρων και floats.

Η εκτύπωση δεκαδικών floats υποστηρίζεται με ακρίβεια δύο δεκαδικών ψηφίων στα δεξιά του δεκαδικού μέρους. Αυτό μπορεί να αλλάξει σύντομα.

Επιστροφή

Καμία

3.7.7 Απαραίτητες συναρτήσεις βιβλιοθηκών που χρησιμοποιήθηκαν

SD

Περιγραφή

Περιέχεται στην βιβλιοθήκη SD.h του λογισμικού. Χρησιμοποιείται για την επικοινωνία μεταξύ του Arduino και της συσκευής αποθήκευσης microSD module.

SD.begin(int pin)

Περιγραφή

Ο τύπος της συνάρτησης είναι boolean. Καθορίζει την τροφοδοσία μέσω του SS pin της microSD module.

Παράμετροι

pin: ακέραιος αριθμός του pin του Arduino στο οποίο συνδέεται το SS pin της microSD module. Αν η microSD module συνδέεται στα pin 8 εως 13 τότε το SS pin αντιστοιχεί στο pin 10 του Arduino UNO. Αν η σύνδεση της κάρτας γίνει στο ICSP τότε η τιμή του δεν χρειάζεται να αλλάξει, αλλά και πάλι επιβάλλεται η χρήση της παραμέτρου ως έχει.

Επιστροφή

Αν επιτευχθεί η τροφοδοσία της κάρτας τότε επιστρέφεται η τιμή FALSE αλλιώς η τιμή TRUE.

SD.exists(filepath)

Περιγραφή

Ο τύπος της συνάρτησης είναι Boolean. Προσδιορίζει αν το προκαθορισμένο αρχείο ή φάκελος αρχείου, υπάρχει.

Παράμετροι

filepath: το όνομα του αρχείου με την επέκταση

Επιστροφή

TRUE αν υπάρχει το αρχείο και FALSE αν δεν υπάρχει

SD.open(filename, mode=FILE_READ)

Περιγραφή

Ανοίγει το αρχείο ή τον φάκελο προς ανάγνωση.

Παράμετροι

filename: το όνομα του αρχείου ή το μονοπάτι αρχείου σε μορφή «directory/file»

mode: παίρνει την τιμή FILE_READ κάτι που σημαίνει ότι το αρχείο ανοίγει με σκοπό να αναγνωστεί και όχι να επεξεργαστεί το περιεχόμενό του.

Επιστροφή

Επιστρέφεται μία δομή αντικειμένου του αρχείου το οποίο ανοίγεται. Η δομή αυτή (class) στην βιβλιοθήκη SD ονομάζεται File και περιέχει όλες τις απαραίτητες πληροφορίες που αφορούν το αρχείο. Κάθε δομή τύπου File μπορεί να αλληλεπιδρά με το αρχείο που αφορά, μέσω κατάλληλων συναρτήσεων. Μόνο ένα αρχείο μπορεί να ανοιχτεί κάθε φορά.

Βιβλιοθήκη File

Περιγραφή

Περιέχει όλες τις απαραίτητες πληροφορίες ενός ανοιγμένου αρχείου. Η επεξεργασία των πληροφοριών που περιέχει γίνεται μέσω ενός συνόλου συναρτήσεων της βιβλιοθήκης SD.

File.read(pointer)

Περιγραφή

Διαβάζει ένα byte προκαθορισμένης θέσης από το ανοιγμένο αρχείο.

Παράμετροι

pointer : ακέραιος αριθμός που προσδιορίζει την θέση του byte μέσα στο αρχείο. Για παράδειγμα αν ο pointer είναι 3 σε ένα αρχείο txt που περιέχει την λέξη «FILE», τότε το byte που θα διαβαστεί είναι το «L».

Επιστροφή

Το byte που διαβάστηκε.

File.seek(position)

Περιγραφή

Μεταφέρει τον pointer-δρομέα του ανοιγμένου αρχείου.

Παράμετροι

position : ακέραιος αριθμός που προσδιορίζει πόσες θέσεις θα μεταφερθεί ο δρομέας μέσα στο αρχείο από την θέση την οποία ήδη βρίσκεται. Για παράδειγμα αν έχουν διαβαστεί 5 bytes τότε ο δρομέας-pointer βρίσκεται στην θέση 6. Αν τότε χρησιμοποιηθεί η συνάρτηση File.seek(3) τότε ο pointer θα πάει στην θέση 9 του αρχείου και όχι στην 3.

Επιστροφή

Καμία

File.print(data)

Περιγραφή

Εκτυπώνει δεδομένα στο αρχείο που έχει ανοιχθεί.

Παράμετροι

data : ακέραιοι τύποι, συμπεριλαμβανομένων χαρακτήρων και floats.

Επιστροφή

Ένας αριθμός που αντιστοιχεί στο ανοιγμένο αρχείο και καλείται instance ή σε περίπτωση σφάλματος, ο κώδικας που προσδιορίζει το είδος του σφάλματος.

File.close()

Περιγραφή

Κλείνει το αρχείο που είναι ανοιγμένο.

Παράμετροι

Καμία. Προϋποτίθεται το instance του αρχείου το οποίο είναι ήδη ανοιγμένο και πρέπει να κλείσει. Έτσι, αφού δεν γίνεται να είναι ανοιχτά παραπάνω από ένα αρχείο, η παράμετρος αυτή έχει αφεθεί κενή.

Επιστροφή

Καμία

Κεφάλαιο 4
Συμπεράσματα και Βελτιώσεις

4.1 Συμπεράσματα

Με την υλοποίηση του πειράματος επιτεύχθηκε η λειτουργία ενός ψηφιακού συστήματος καταγραφής τροχιάς. Το υλικό που χρησιμοποιήθηκε βρίσκεται εύκολα και σε προσιτές τιμές στο διαδίκτυο. Η αποκατάσταση της επικοινωνίας των υλικών μεταξύ τους και με το Arduino έγινε με την κατάλληλη συνδεσμολογία ενώ μέσω του φιλικού και εύχρηστου περιβάλλοντος Arduino ο χρήστης επεμβαίνει στις λειτουργικές ρυθμίσεις της συσκευής. Παράλληλα, στο διαδίκτυο υπάρχουν έτοιμες βιβλιοθήκες που με την χρήση τους διευκολύνεται κατά πολύ η επεξεργασία των ρυθμίσεων και των λειτουργιών του υλικού σε σχέση με το Arduino. Με τις βιβλιοθήκες αυτές απαλλάσσεται ο χρήστης από την επακριβής λειτουργία του υλικού που χρησιμοποιεί και το μόνο που χρειάζεται είναι η γνώση παραμετροποίησης κάθε συνάρτησης.

Συμπερασματικά διαπιστώνουμε ότι η κατασκευή ενός ψηφιακού συστήματος καταγραφής τροχιάς είναι εφικτή με χρήση υλικού χαμηλού κόστους και με στοιχειώδης γνώσεις προγραμματισμού.

4.2 Προβλήματα που αντιμετωπίστηκαν

Τα προβλήματα που παρουσιάστηκαν κατά την υλοποίηση του πειράματος, ήταν αρχικά η αποθήκευση κάθε παραγράφου NMEA σε σωστή μορφή. Παρατηρήθηκε ότι πολλές παραγράφοι αποθηκεύονταν αλλοιωμένες με «παραμορφωμένα» δεδομένα. Αυτό παρουσιάστηκε λόγω της διαδικασίας αποθήκευσης που ακολουθήθηκε αρχικά. Ωστόσο, παρόλο που επιλύθηκε το πρόβλημα, δεν μπορεί να γίνει κάποια διευκρίνηση ως προς την ακριβή αιτία του. Επίσης, παρουσιάστηκαν αρκετές ατέλειες στην χρήση της βιβλιοθήκης της μικροκάρτας, που χρησιμοποιήθηκε αρχικά. Η αποθήκευση δεδομένων στην κάρτα SD απαιτούσε το «γέμισμα» του αρχείου με χαρακτήρες τύπου HTML (ETX = ALT+3). Συνεπώς, η αποθήκευση σε κενό αρχείο δεν ήταν δυνατή και άρα η δημιουργία νέου αρχείου μέσα στην μικρο-κάρτα, αυτόματα μέσω του λογισμικού και όχι από τον χρήστη, θα ήταν ανούσια. Το πρόβλημα λύθηκε με την χρήση ανανεωμένης βιβλιοθήκης (SD), η οποία είναι πιο απλή, εύχρηστη και αποτελεσματική, χωρίς να προϋποθέτει κάτι ιδιαίτερο για την μορφή των αρχείων όπου θα αποθηκεύονταν τα δεδομένα.

Έπειτα, αντιμετωπίστηκε πρόβλημα σχετικά με την λειτουργία του δέκτη GPS σε συνδυασμό με την microSD module. Ήταν αδύνατη η αποθήκευση δεδομένων του δέκτη στην μικρο-κάρτα όταν αυτή ήταν συνδεδεμένη στα pins 8 εως 13. Το πρόβλημα αντιμετωπίστηκε με την σύνδεση της μικρο-κάρτας στη θύρα διασύνδεσης ICSP, εξασφαλίζοντας έτσι και περισσότερα ελεύθερα pins προς χρήση για μελλοντικές βελτιώσεις της εφαρμογής.

Τέλος, ένας από τους στόχους του πειράματος ήταν η γραφική αναπαράστασης της διαδρομής που κατέγραψε ο δέκτης GPS στο Google Earth. Αυτό για να γίνει έπρεπε να δημιουργηθεί ένα αρχείο KML, κάτι που δεν έγινε εφικτό λόγω της περιορισμένης μνήμης που διαθέτει η πλατφόρμα Arduino Uno. Έτσι, αναφέρεται ενδεικτικά ότι, η γραφική αναπαράσταση των διαδρομών σε αυτή την εργασία έγινε με την εισαγωγή δεδομένων στο Google Earth αναλογικά από τον χρήστη χωρίς τη χρήση αλγορίθμου.

4.2.1 Δημιουργία αρχείου KML

Η δομή του αρχείου KML για την γραφική αναπαράσταση μιας διαδρομής, αποτελεί ένα πρότυπο εντολών xml. Οι συντεταγμένες των σημείων που καθορίζουν την μορφή της διαδρομής, είναι οι βασικές παράμετροι που πρέπει να εισαχθούν σε ένα αρχείο KML ώστε να αναπαρασταθεί η διαδρομή στο Google Earth. Άλλες παράμετροι που μπορούν να εισαχθούν αλλά δεν αφορούν την εργασία είναι το χρώμα των γραμμών, το πάχος κλπ. Έτσι για την παραγωγή του αρχείου, με την πλατφόρμα Arduino Uno, δημιουργήθηκε ένα αρχείο KML στην μικρο-κάρτα SD. Το αρχείο αυτό περιλαμβάνει τα αμετάβλητα συστατικά του αρχείου, με στόχο να συμπληρωθούν οι συντεταγμένες των σημείων μέσω του αλγορίθμου.

Αφού ληφθούν τα δεδομένα από το GPS module και αποθηκευτούν στην κάρτα, κάθε φορά που πατιέται το κουμπί, αποθηκεύονται στο αρχείο KML οι συντεταγμένες του σημείου ενδιαφέροντος. Οι συντεταγμένες σημείου στο KML εισάγονται σε μία παράγραφο. Για παράδειγμα ένα σημείο με συντεταγμένες γεωγραφικού μήκους 23.78100858963766 και γεωγραφικού πλάτους 37.97624536015139 αποθηκεύεται στο αρχείο KML όπως φαίνεται παρακάτω :

```
<Placemark>
	<name>point3</name>
	<LookAt>
		<longitude>23.78100858963766</longitude>
		<latitude>37.97624536015139</latitude>
		<altitude>0</altitude>
		<heading>0.9901300027679391</heading>
		<tilt>0</tilt>
		<range>253.7359952335571</range>
		<altitudeMode>relativeToGround</altitudeMode>
		<gx:altitudeMode>relativeToSeaFloor</gx:altitudeMode>
		</LookAt>
		<styleUrl>#msn_ylw-pushpin0</styleUrl>
		<Point>
			<altitudeMode>clampToGround</altitudeMode>
			<gx:altitudeMode>clampToSeaFloor</gx:altitudeMode>
	<coordinates>23.7816395852935,37.97620216923205,0</coordinates>
		</Point>
```

```
</Placemark>
```

Συνεπώς, μετά από την λήψη κάθε σημείου ενδιαφέροντος πρέπει να αποθηκευτεί μία παράγραφος παρόμοια της παραπάνω. Το πρόβλημα που αντιμετωπίστηκε σε αυτό το εγχείρημα έχει να κάνει με την μνήμη την οποία διαχειρίζεται το Arduino. Κι αυτό επειδή το μέγεθος της παραγράφου, δεδομένου ότι κάθε γράμμα της παραγράφου καταλαμβάνει μνήμη ενός byte, είναι μεγάλο και δημιουργεί προβλήματα στην μεταγλώττιση (compile) του αλγορίθμου.

4.3 Βελτιώσεις του GPS Logger

Η κατασκευή της συσκευής με την οποία ασχοληθήκαμε μπορεί να βελτιωθεί με αρκετούς τρόπους. Αρχικά, μία παράμετρος που επιδέχεται βελτίωση είναι η τροφοδότηση του Arduino, το οποίο πρέπει να βρίσκεται συνεχώς σε σύνδεση με έναν ηλεκτρονικό υπολογιστή μέσω θύρας USB. Κάλλιστα θα μπορούσε να υπάρχει αυτόνομη παροχή ισχύος από ανεξάρτητη πηγή, όπως για παράδειγμα από ένα αυτοκινούμενο όχημα ή μία μπαταρία.

Επίσης, μία συσκευή καταγραφής τροχιάς μπορεί να εφοδιαστεί με ένα GSM module με σκοπό να στέλνει σε πραγματικό χρόνο δεδομένα μέσω δικτύου κινητής τηλεφωνίας.

Έπειτα, υπάρχει η δυνατότητα παραγωγής αρχείου KML ώστε να υπάρχει άμεση επεξεργασία και εισαγωγή δεδομένων στο Google Earth για γραφική αναπαράσταση των λήψεων της συσκευής. Αυτό προϋποθέτει την ύπαρξη μνήμης κάτι που μπορεί να επιτευχθεί με σύνδεση 2 ή και παραπάνω Arduino μεταξύ τους ή με χρήση κατάλληλων εξαρτημάτων.

Τέλος, η εφαρμογή της παρούσας εργασίας μπορεί να υποστηρίξει την ενσωμάτωση εξαρτημάτων που θα παρέχουν δυνατότητα καταγραφής και άλλων μεγεθών από το όχημα στο οποίο βρίσκεται ενσωματωμένο. Τέτοια μεγέθη, όπως η ταχύτητα, επιτάχυνση, απόσταση που διανύθηκε, μπορούν να ενταχθούν σε ψηφιακά συστήματα ελέγχου του οχήματος και να παρέχουν στον χρήστη περισσότερες πληροφορίες. Ένα τέτοιο παράδειγμα καθημερινής χρήσης είναι τα GPS αυτοκινήτων.

Βιβλιογραφία

[01] http://en.wikipedia.org/wiki/GPS_tracking_unit

[02] Banzi, M., (2009). Getting Started with Arduino, 1st edition, Italy, ISBN – 0596155514

[03] Evans, B., (2007), Arduino Programming Notebook, 2st edition, California

[04] Greenberg, I., (2007), Processing: Creative Coding and Computational Art, 1st edition, New York , ISBN–10:1-59059–617-X

[05] Igoe, T., (2007). Making Things Talk, 1st edition, California: O" Reilly Media, ISBN–10: 0-596-51051-9

[06] Noble, J., (2009), Programming interactivity: A Designers Guide to Processing, Arduino and open Framework, 1st edition, California: O" Reilly Media, ISBN – 0596154143

[07] Reas, K., (2007), Processing: A Programming Handbook for Visual Designers and Artists, 1st edition, California: O" Reilly Media

[08] Stroustrup, B., (2003), Η Γλώσσα προγραμματισμού C ++, 3rd edition, (μτφρ, Τ. Άλβας). Αθήνα: Κλειδάριθμος

[09] Shiffman, D., (2008) Learning Processing: A Beginner"s Guide to Programming Images, Animation and Interaction, 1st edition, USA , ISBN: 978 – 0 - 12 -373602 - 4

[10] http://en.wikipedia.org/wiki/Gps

[11] http://en.wikipedia.org/wiki/Galileo_satellite_navigation

[12] http://en.wikipedia.org/wiki/Kml

[13] http://en.wikipedia.org/wiki/Secure_Digital

[14] http://arduino.cc/

[15] http://www.gpsinformation.org/dale/nmea.htm

[16] http://www.parallax.com/

[17] http://www.libelium.com/

[18] http://en.wikipedia.org/wiki/GLONASS, http://www.glonass-ianc.rsa.ru/en/

[19] http://ilrs.gsfc.nasa.gov/satellite_missions/list_of_satellites/g120_general.html

[20] http://www.sinodefence.com/satellites/compass-beidou.asp

[21] http://en.wikipedia.org/wiki/Beidou_navigation_system