ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΤΕ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αριθμός 1417

ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΥΠΗΡΕΣΙΕΣ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

ΣΠΟΥΔΑΣΤΡΙΑ:
ΦΟΥΝΤΑ ΑΝΤΙΓΟΝΗ

ΕΙΣΗΓΗΤΗΣ:
ΓΙΑΝΝΟΥΛΗΣ ΣΠΗΛΙΟΣ

ΠΑΤΡΑ 2015

ΠΕΡΙΛΗΨΗ

Ανάπτυξη λογισμικού για υπηρεσίες εντοπισμού θέσης

Στόχος αυτής της διπλωματικής εργασίας είναι η ανάπτυξη λογισμικού, σε γλώσσα προγραμματισμού C, για την οικογένεια μικροελεγκτών AVR της εταιρείας Atmel.Το λογισμικό θα περιλαμβάνει βασικές λειτουργίες εισόδου/εξόδου και έλεγχο περιφερειακών συστημάτων όπως οθόνη LCD, GPS, GSM modem.Η τελική εφαρμογή θα ελέγχει μία ή περισσότερες εισόδους συναγερμού και θα ειδοποιεί τον χρήστη με ένα μήνυμα τύπου SMS σε περίπτωση που υπάρξει συναγερμός. Το μήνυμα SMS θα περιγράφει το είδος του συναγερμού και θα έχει τη δυνατότητα να δηλώσει και τις συντεταγμένες του συστήματος, οι οποίες θα λαμβάνονται από σύστημα GPS. Με τον τρόπο αυτό δημιουργείται μία εφαρμογή στον τομέα της ασφάλειας, η οποία μπορεί να χρησιμοποιηθεί για τον εντοπισμό ενός αντικειμένου π.χ. αυτοκίνητο ή και ανθρώπου.Το λογισμικό θα πρέπει να είναι δομημένο, εύκολα επεκτάσιμο και θα μπορεί εύκολα να μεταφερθεί σε άλλους μικροελεγκτές με παρόμοιες δυνατότητες.

Για την ανάπτυξη του λογισμικού θα χρησιμοποιηθεί το ATMEL Studio 6.0 που αποτελεί μία ολοκληρωμένη πλατφόρμα ανάπτυξης λογισμικού για μικροελεγκτές AVR της ATMEL και το αντίστοιχο evaluation board για τον μικροελεγκτή (π.χ Xplained Evaluation Board για τον ATxmega256A3BU). Προκειμένου να υποστηριχτούν όλες οι δυνατότητες που περιγράφονται παραπάνω, θα γίνει ενσωμάτωση και διασύνδεση σχετικού υλικού (hardware) στο υπάρχον evaluation board όπου απαιτείται π.χ. προσθήκη GPS.

SUMMARY

Location services development software

The scope of this dissertation is to develop firmware, in C programming language, for microcontrollers of the AVR family for ATMEL company. The firmware will include basic input/output functions and control of peripherals systems such as LCD monitor, GPS device, GSM modem. The final application will check one or more rules and will trigger alarms based on those rules.It will then notify the user with an SMS if an alarm has been activated. The SMS will describe the type of alarm and will also include the coordinates of the GPS device.This design will result in an application that will act as a typical application in the security field and could be used for locating an object (e.g. car or person).

The firmware should be structured, easily scalable and it should be easily transposed to another microcontroller with similar possibilities. The ATMEL studio and an evaluation board of the AVR family (e.g. Xplained) will be used as a software development platform. In order to support all the features described above, the overall hardware will be enchanced and peruprheals will be added to the main evaluation board where it is necessary (e.g. add GPS device).

Περιεχόμενα

1. ΕΙΣΑΓΩΓΗ

1.1. Γενικά

Οι μικροελεγκτές AVR χρησιμοποιούν τροποποιημένη Αρχιτεκτονική Χάρβαρντ 8-bit RISC και αναπτύχθηκαν από την Atmel για πρώτη φορά το 1996. Οι AVR ήταν μια από τις οικογένειες μικροελεγκτών που έκαναν χρήση της on-chip μνήμης flash για την αποθήκευση του προγράμματος,σε αντίθεση με επαναπρογραμματιζόμενες EPROM ή EEPROM που χρησιμοποιούνται από άλλους μικροελεγκτές.

Τα πλεονεκτήματα των μικροελεγκτών είναι η απόδοση, η αποτελεσματικότητα και η ευελιξία την οποία έχουν. Με την ευκολία στη χρήση τους, τη χαμηλή κατανάλωση ενέργειας και το υψηλό επίπεδο ολοκλήρωσης, οι 8-bit AVR μικροελεγκτές παρέχουν ένα μοναδικό συνδυασμό απόδοσης,εξοικονόμηση ενέργειας και ευελιξία σχεδιασμού.

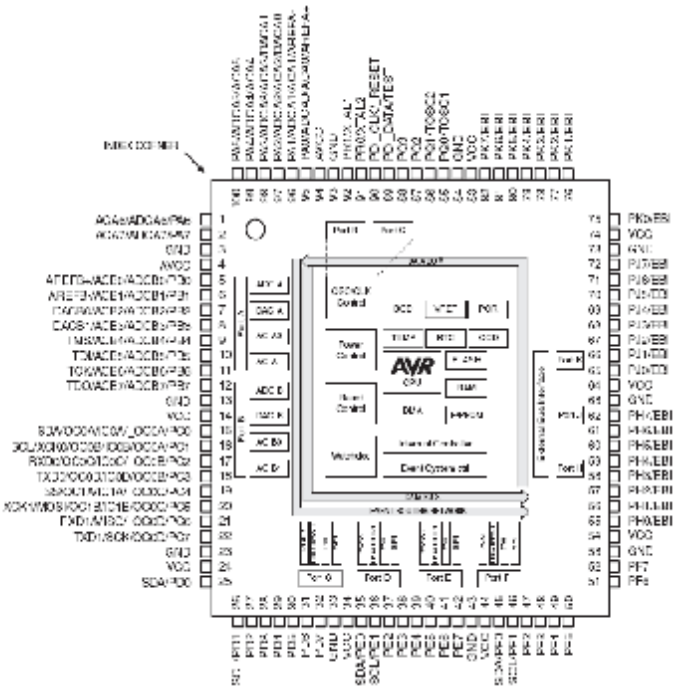
Στη παρούσα διπλωματική εργασία για την ανάπτυξη του λογισμικού χρησιμοποιήθηκε ο AVR ATxmega256A3BU.

Η δομή της εφαρμογής που σχεδιάστηκε και υλοποιήθηκε στα πλάισια της πτυχιακής εργασίας αποτελείται από ένα GPS το οποίο δέχεται το σήμα από τους δορυφόρους GPS και ανιχνεύει τις συντεταγμένες του αντικειμένου μας. Στη συνέχεια ο AVR επεξεργάζεται τις πληροφορίες μέσω του software που του έχουμε εγκαταστήσει.Ανάλογα με το αποτέλεσμα της επεξεργασίας και των παραμέτρων που έχουμε ορίσει,πραγματοποιεί η όχι επικοινωνία με το GSM Modem μέσω του οποίου αποστέλλει SMS στον χρήστη που έχουμε ορίσει σε περίπτωση που το αντικείμενο μας βρεθεί εκτός των συντεταγμένων που έχουμε θέσει.Σε αυτή την περίπτωση υπάρχει alarm και η σήμανση αποστέλλεται στο κινητό μας μέσω sms.Ο προγραμματισμός του συστήματος γίνεται μέσω της σειριακής θύρας του PC με χρήση του AVR Studio 6 και του HyperTerminal.

Εικόνα 1. Δομικό διάγραμμα εφαρμογής

1.2.Ο μικροελεγκτής ATxmega256A3BU

Πρόκειται για έναν υψηλής απόδοσης, χαμηλής ισχύος Atmel AVR 8/16-bit μικροελεγκτή.



Εικόνα2. Ο μικροελεγκτής ATxMEGA256A3BU

1.2.1. Χαρακτηριστικά

Μνήμη

Σταθερές (Nonvolatile) μνήμες προγράμματος και δεδομένων:

- 256 KBytes προγραμματιζόμενη μνήμη τύπου flash
- 8 Kbytes μνήμη εκκίνησης
- 4 KBytes EEPROM
- 16KBytes εσωτερική SRAM

Περιφερειακά

- DMA ελεγκτή τεσσάρων καναλιών.
- Σύστημα συμβάντων οκτώ καναλιών.
- Επτά χρονιστές /μετρητές των 16-bits.
- Τέσσερις χρονιστές /μετρητές με τέσσερα κανάλια σύγκρισης εξόδου ή καταγραφής εισόδου.
- Τρεις χρονιστές /μετρητές με δύο κανάλια σύγκρισης εξόδου ή καταγραφής εισόδου.
- Υψηλής ανάλυσης επέκταση σε όλους τους χρονιστές /μετρητές.
- Εξελιγμένη επέκταση κυματομορφής σε ένα χρονιστή / μετρητή.
- Ένα USB Interface.

- USB 2.00 συσκευή συμβατή με πλήρους ταχύτητας (12Mbps) και χαμηλής ταχύτητας (1.5Mbps)) λειτουργία.
- Έξι USARTs με υποστήριξη IrDA για ένα USART.
- Δύο Interface δύο καλωδίων με διπλό ταίριασμα διεύθυνση (I2C και SMBus συμβατή).
- Δύο σειριακά περιφερειακά Interface (SPIs).
- AES και DES μηχανές κρυπτογράφησης.
- CRC-16 (CRC-CCITT) και CRC-32 (IEEE 802.3) γεννήτρια.
- 32-bit μετρητή πραγματικού χρόνου (RTC) με ξεχωριστό ταλαντωτή και
- εφεδρικό σύστημα με μπαταρία.
- Δύο (δεκαέξι-καναλιών) 12-bit, 2msps μετατροπείς αναλογικού σε ψηφιακό.
- Ένα (δύο-καναλιών) 12-bit, 1msps ψηφιακό σε αναλογικό μετατροπέα.
- Τέσσερις Αναλογικοί Συγκριτές με λειτουργία σύγκρισης παραθύρου και πηγές ρεύματος.
- Εξωτερικές διακοπές σε όλα τα γενικού σκοπού I/O pins.
- Προγραμματιζόμενος χρονοδιακόπτης watchdog με ξεχωριστό on-chip εξαιρετικά χαμηλής ισχύος ταλαντωτή.
- QTouch υποστήριξη βιβλιοθήκης.

Ειδικά

- Power-on reset και προγραμματιζόμενο brown-out ανίχνευση.
- Εσωτερικές και εξωτερικές επιλογές ρολογιού με PLL και prescaler .
- Προγραμματιζόμενο πολυεπίπεδο ελεγκτή διακοπών.
- Πέντε τρόποι λειτουργίας χαμηλής κατανάλωσης.
- Interface προγραμματισμού και εντοπισμού σφαλμάτων.
- JTAG (IEEE 1149.1συμβατό) Interface, συμπεριλαμβανομένων των ορίων σάρωσης.
- PDI (Interface προγραμματισμού και εντοπισμού σφαλμάτων).

I/O και τα πακέτα

- 47 προγραμματιζόμενα I/O pins.
- 64-lead TQFP.
- 64-pad QFN.

Τάση λειτουργίας

- 1.6 - 3.6V

Συχνότητα λειτουργίας

- 0-12 MHz από 1.6V
- 0-32 MHz από 2.7V

Ακολουθεί η περιγραφή των βασικών λειτουργιών του AVR που χρησιμοποιήθηκαν σε αυτή τη διπλωματική εργασία.

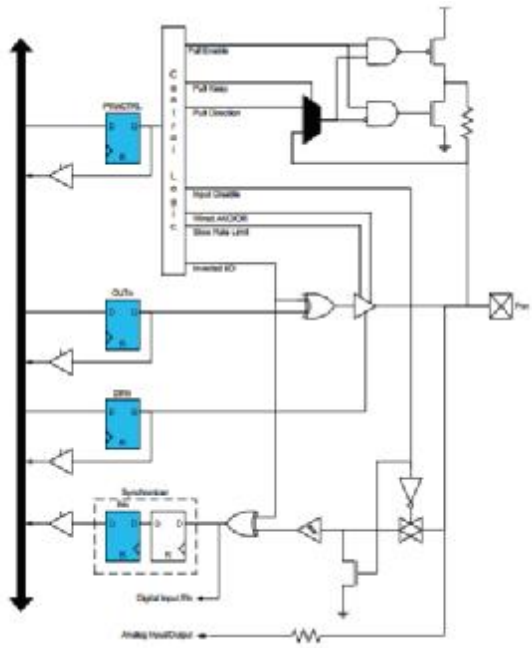
1.2.2. Θύρες Εισόδου/Εξόδου (I/O Ports)

Οι μικροελεγκτές AVR XMEGA έχουν ευέλικτες I/O πόρτες γενικού σκοπού. Μία πόρτα αποτελείται από 8 pins: pins από 0 μέχρι 7. Κάθε pin μπορεί να διαμορφωθεί ως είσοδος ή έξοδος με τις απαραίτητες τοποθετήσεις οδηγών και ρυθμίσεων.Επίσης υλοποιούν σύγχρονη και ασύγχρονη ανίχνευση εισόδου συνοδευόμενη από διακοπές και συμβάντα για επιλεγμένες αλλαγές κατάστασης pin.

Όλες οι λειτουργίες είναι μεμονωμένες και διαμορφώσιμες ανά pin αλλά διάφορα pins μπορούν να διαμορφωθούν σε μια μόνο λειτουργία.Τα pins έχουν λειτουργικότητα υλικού όπου διαβάζουν-τροποποιούν-γράφουν με ασφαλή και σωστό τρόπο την αλλαγή της τιμής του οδηγού ή/και την διαμόρφωση της αντίστασης εξόδου.Η κατεύθυνση ενός pin μπορεί να αλλάξει χωρίς να απαιτείται να αλλάξει η κατεύθυνση οποιουδήποτε άλλου pin.

Η διαμόρφωση των pins μιας θύρας επίσης ελέγχει την επιλογή εισόδου και εξόδου από άλλες λειτουργίες της συσκευής. Για παράδειγμα, είναι δυνατό να έχουμε και το περιφερειακό ρολόι και την έξοδο του ρολογιού πραγματικού χρόνου σε ένα pin της θύρας και διαθέσιμο για εξωτερική χρήση. Το ίδιο εφαρμόζεται και στα συμβάντα από το σύστημα διαχείρισης συμβάντων το οποίο μπορεί να χρησιμοποιηθεί για το συγχρονισμό και έλεγχο εξωτερικών λειτουργιών. Άλλα ψηφιακά περιφερειακά, όπως η USART, SPI, και οι χρονιστές/μετρητές, μπορούν να χαρτογραφηθούν σε επιλέξιμες μέσω pin θέσεων προκειμένου να βελτιστοποιηθεί το pin-out σε σχέση με τις ανάγκες της εφαρμογής.

Το παρακάτω σχήμα παρουσιάζει την λειτουργία I/O ενός pin καθώς και τους καταχωρητές που είναι διαθέσιμοι για τον έλεγχό του.



Εικόνα 3. Γενική λειτουργία I/O pin

Κάθε θύρα έχει έναν καταχωρητή κατεύθυνσης δεδομένων (DIR) και έναν καταχωρητή δεδομένων εξόδου (OUT) που χρησιμοποιούνται για έλεγχο των pinsτης θύρας. Ο καταχωρητής των δεδομένων εισόδου (IN) χρησιμοποιείται για την ανάγνωση των pinsτης θύρας. Επιπλέον κάθε pinέχει έναν καταχωρητή διαμόρφωσης (PINnCTRL) για επιπλέον πρόσθετη διαμόρφωση του pin. Η κατεύθυνση του pinορίζεται από το bit DIRn του καταχωρητή DIR.Εάν η DIRn είναι στο “1”, το pinn διαμορφώνεται ως pin εξόδου. Εάν η DIRn είναι στο “0”, το pinn διαμορφώνεται ως pin εισόδου. Όταν η κατεύθυνση του pin ορίζεται σαν έξοδο, το bit OUTn στον καταχωρητή OUT χρησιμοποιείται για να θέσει την τιμή του pin. Εάν η OUTn είναι στο “1”,το pin οδηγείται στο“1”. Εάν το OUTn είναι στο “0”, το pin οδηγείται στο“0”. Ο καταχωρητής IN χρησιμοποιείται για διάβασμα των τιμών των pins. Η τιμή ενός pin μπορεί πάντα να διαβαστεί ανεξαρτήτως αν το pin έχει διαμορφωθεί σαν είσοδο ή σαν έξοδο εκτός εάν η ψηφιακή είσοδο έχει απενεργοποιηθεί.

Τα I/O pins είναι σε κατάσταση υψηλής αντίστασης (tri-stated) όταν ένα γίνεται reset,ακόμα και αν το ρολόι δεν τρέχει. Όπως αναφέρθηκε, ο καταχωρητής διαμόρφωσης ενός pinn (PINnCTRL) χρησιμοποιείται για επιπλέον διαμόρφωση του I/O του pin. Ένα pin μπορεί να τεθεί σε totem-pole, wired-AND, ή wired-OR διαμόρφωση.Είναι επίσης πιθανόν να ενεργοποιηθεί αντιστροφή εισόδου και εξόδου για ένα pin.Μια totem-pole έχει 4 πιθανές διαμορφώσεις: totem-pole (push-pull), pull-down, pull-up, and bus-keeper. Το bus-keeper είναι ενεργό και στις δύο κατευθύνσεις.Αυτό γίνεται για να αποφευχθεί η ταλάντωση κατά την απενεργοποίηση της εξόδου. Το totem-pole διαμορφώνεται pull-up και pull-down με ενεργοποιημένες αντιστάσεις μόνο όταν το pin έχει οριστεί ως είσοδος.Αυτό το χαρακτηριστικό εξαλείφει την άσκοπη κατανάλωση ενέργειας. Για wired-AND και wired-OR διαμόρφωση, οι προαιρετικές pull-up και pull-down αντιστάσεις δραστηριοποιούνται και στις δύο κατευθύνσεις εισόδου και εξόδου.

Δεδομένου ότι η διαμόρφωση τραβήγματος (pull) διαμορφώνεται μέσω του καταχωρητή διαμόρφωσης pin, όλες οι ενδιάμεσες στάθμες των θυρών κατά τη διάρκεια του switching,της κατεύθυνσης των pins και των τιμών των pins αποφεύγονται.

Totem-pole

Στη totem-pole (push-pull) διαμόρφωση, το pin οδηγείται σε χαμηλή ή υψηλή στάθμη σύμφωνα με την κατάσταση του αντίστοιχου bit του καταχωρητή OUT. Σε αυτήν την διαμόρφωση, δεν υπάρχει κανένας περιορισμός για την ζήτηση ή παροχή ρεύματος εκτός από αυτό που το pin είναι ικανό να δώσει. Εάν το pin διαμορφώνεται για είσοδο, το pin θα είναι «στον αέρα» εάν δεν συνδεθεί καμία εξωτερική αντίσταση.

Totem-pole with Pull-down

Σε αυτόν τον τρόπο, η διαμόρφωση είναι η ίδια όπως για totem-pole λειτουργία, εκτός του ότι το pinδιαμορφώνεται με μια εσωτερική pull-downαντίσταση όταν τίθεται σαν είσοδο.

Totem-pole with Pull-up

Σε αυτόν τον τρόπο, η διαμόρφωση είναι όπως για totem-pole, εκτός του ότι το pin διαμορφώνεται με μια εσωτερική pull-up αντίσταση όταν τίθεται σαν είσοδο.

Bus-keeper

Στη διαμόρφωση Bus-keeper, παρέχεται έναν μηχανισμός (weakbus-keeper) που θα κρατήσει το pin στη λογική του στάθμη όταν το pin δεν οδηγείται πλέον σε υψηλή ή χαμηλή στάθμη. Εάν η τελευταία στάθμη στο pin/δίαυλο ήταν «1», η διαμόρφωση Bus-keeper θα χρησιμοποιήσει την εσωτερική pull αντίσταση για να κρατήσει το δίαυλο σε υψηλή στάθμη. Εάν η τελευταίο στάθμη στο pin/δίαυλο ήταν «0», η διαμόρφωση Bus-keeper θα χρησιμοποιήσει την εσωτερική pull αντίσταση για να κρατήσει το δίαυλο σε χαμηλή στάθμη.

Wired-OR

Στην wired-OR διαμόρφωση, το pin θα οδηγηθεί σε υψηλή στάθμη όταν τα αντίστοιχα bits στον OUT και DIR καταχωρητή τεθούν στο ‘’1’’. Όταν ο OUT καταχωρητής τίθεται μηδέν, το pin απελευθερώνεται, επιτρέποντάς του να γίνει pulled low με μια εσωτερική ή μία εξωτερική αντίσταση. Εάν χρησιμοποιηθεί εσωτερική pull-down, αυτό είναι επίσης ενεργό εάν το pin τεθεί ως είσοδος.

Wired-AND

Στην wired-AND διαμόρφωση, το pin θα οδηγηθεί χαμηλά όταν τα αντίστοιχα bits στους καταχωρητές OUT και DIR τεθούν στο μηδέν. Όταν ο OUT καταχωρητής τίθεται το ένα, το pin απελευθερώνεται επιτρέποντας στο pin να πάει σε υψηλή στάθμη με μια εσωτερική ή μία εξωτερική αντίσταση.Εάν χρησιμοποιείται η εσωτερική pull-up, αυτό είναι επίσης ενεργό όταν το pin τεθεί ως είσοδος .

Reading the PinValue

Ανεξάρτητα της κατεύθυνσης δεδομένων του pin, η τιμή του pin μπορεί να διαβαστεί από τον καταχωρητή IN. Εάν η ψηφιακή είσοδος είναι απενεργοποιημένη, η τιμή του pin δεν μπορεί να διαβαστεί.

Input Sense Configuration

Η ανίχνευση εισόδου χρησιμοποιείται για να ανιχνεύσει μια παρυφή ή ένα επίπεδο στο I/O pinεισόδου. Οι διαφορετικές διαμορφώσεις ανίχνευσης που είναι διαθέσιμες για κάθε pin είναι η ανίχνευση μιας ανερχόμενης παρυφής, πίπτουσας παρυφής, οποιασδήποτε παρυφής ή ανίχνευσης ενός χαμηλού επιπέδου. Το υψηλό επίπεδο μπορεί να ανιχνευτεί χρησιμοποιώντας διαμόρφωση ανάστροφης εισόδου. Η ανίχνευση εισόδου μπορεί να χρησιμοποιηθεί για να προκαλέσει αιτήματα διακοπής (IREQ) ή συμβάντα όταν υπάρχει μια αλλαγή στο pin.Τα I/O pins υποστηρίζουν τη σύγχρονη και ασύγχρονη ανίχνευση εισόδου. Η σύγχρονη ανίχνευση απαιτεί την παρουσία περιφερειακού ρολογιού, ενώ η ασύγχρονη ανίχνευση δεν απαιτεί.

Port Interrupt

Κάθε θύρα έχει δύο διανύσματα διακοπής, και είναι διαμορφώσιμο ποιο pin της θύρας θα προκαλέσει κάθε διακοπή. Οι διακοπές της θύρας θα πρέπει να ενεργοποιηθούν πριν χρησιμοποιηθούν. Ποιες διαμορφώσεις ανίχνευσης μπορούν να χρησιμοποιηθούν για να παραγάγουν διακοπές εξαρτάται από το αν η σύγχρονη ή ασύγχρονη ανίχνευση εισόδου είναι διαθέσιμη για το επιλεγμένο pin.

Για τη σύγχρονη ανίχνευση, όλες οι διαμορφώσεις ανίχνευσης μπορούν να χρησιμοποιηθούν για να παραγάγουν διακοπές. Για την ανίχνευση παρυφής, η τιμή του pin που αλλάζει πρέπει να δειγματοληπτηθεί μία φορά από το περιφερειακό ρολόι για να παραχθεί αίτημα διακοπής.

Για την ασύγχρονη ανίχνευση, μόνο το pin 2 σε κάθε θύρα έχει την πλήρη υποστήριξη σύγχρονης ανίχνευσης. Αυτό σημαίνει ότι για την ανίχνευση παρυφής, το pin 2 θα ανιχνεύσει και θα κλειδώσει οποιαδήποτε παρυφή και θα προκαλέσει πάντα ένα αίτημα διακοπής. Τα άλλα pin της θύρας έχουν περιορισμούς στην ασύγχρονη υποστήριξη ανίχνευσης. Αυτό σημαίνει ότι για την ανίχνευση παρυφής, η αλλαγμένη τιμή πρέπει να κρατηθεί μέχρι η συσκευή να ξεκινήσει και ένα ρολόι να είναι παρόν. Εάν η τιμή του pin επιστρέψει στην αρχική τιμή της πριν από το τέλος του χρόνου αφύπνισης της συσκευής, η συσκευή θα είναι ακόμη σε αφύπνιση, αλλά δεν θα παραχθεί αίτημα διακοπής.

Μια χαμηλή στάθμη μπορεί πάντα να ανιχνευθεί από όλα τα pins, ανεξάρτητα από ένα περιφερειακό ρολόι είναι παρόν ή όχι. Εάν ένα pin είναι διαμορφωμένο για χαμηλού επιπέδου ανίχνευση, η διακοπή θα προκληθεί όσο το pin κρατιέται σε χαμηλή στάθμη. Στον ενεργό τρόπο λειτουργίας, το χαμηλό επίπεδο πρέπει να κρατηθεί μέχρι την ολοκλήρωση της εκτέλεσης της τρέχουσας εντολής για να παραχθεί μια διακοπή. Σε όλους τους αδρανείς τρόπους λειτουργίας, η χαμηλή στάθμη πρέπει να κρατιέται μέχρι το τέλος του χρόνου αφύπνισης της συσκευής για να παραχθεί μια διακοπή. Εάν το χαμηλό επίπεδο εξαφανιστεί πριν το τέλος του χρόνου αφύπνισης, η συσκευή θα είναι ακόμα σε αφύπνιση αλλά δεν θα παραχθεί διακοπή.

Port Event

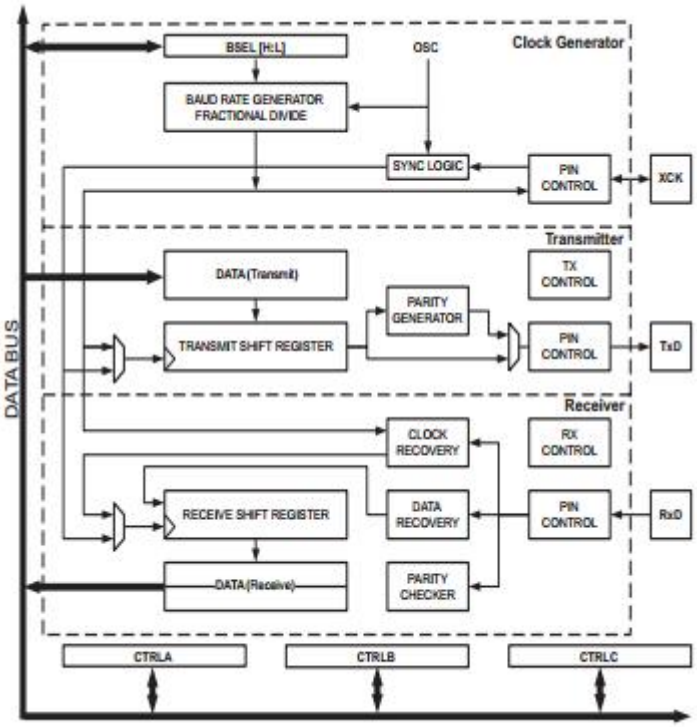
Τα pins των θυρών μπορούν να παραγάγουν ένα συμβάν όταν υπάρχει μια αλλαγή στο pin. Οι διαμορφώσεις ανίχνευσης αποφασίζουν τις συνθήκες για κάθε pin και πότε στην ουσία θα παραχθεί ένα συμβάν. Η παραγωγή συμβάντος απαιτεί την παρουσία ενός περιφερειακού ρολογιού και η ασύγχρονη δημιουργία συμβάντων δεν είναι δυνατή. Για ανίχνευση παρυφής, η αλλαγμένη τιμή του pin πρέπει να δειγματοληπτηθεί μία φορά από το περιφερειακό ρολόι για να παραχθεί ένα συμβάν. Για την ανίχνευση επιπέδου, μια χαμηλή στάθμη στο pin δεν θα παράγει συμβάντα και μία υψηλή στάθμη στο pin θα παράγει συνεχώς συμβάντα. Για τα γεγονότα που παράγονται σε ένα χαμηλό επίπεδο, η διαμόρφωση του pin πρέπει να τεθεί σε ανάστροφο I/O.

1.2.3. USART

Ο γενικός σύγχρονος και ασύγχρονος σειριακός δέκτης και πομπός (USART) είναι ένα γρήγορο και ευέλικτο υποσύστημα σειριακής επικοινωνίας. Η USART υποστηρίζει την πλήρη-διπλή επικοινωνία, όπως και την σύγχρονη και ασύγχρονη σειριακή λειτουργία.

Η επικοινωνία βασίζεται σε πλαίσια και η μορφή του πλαισίου μπορεί να διαμορφωθεί για να υποστηρίξει ένα ευρύ φάσμα προτύπων. Η USART διαθέτει απομονωτές και στις δύο κατευθύνσεις, επιτρέποντας τη συνεχή μετάδοση δεδομένων χωρίς οποιαδήποτε καθυστέρηση μεταξύ των πλαισίων. Ξεχωριστές διακοπές για λήψη και εκπομπή ελέγχουν την επικοινωνία. Το λάθος του πλαισίου και η υπερχείλιση του απομονωτή λήψης ανιχνεύονται στο υλικό και υποδεικνύονται με χωριστές σημαίες κατάστασης. Η δημιουργία bit ισοτιμίας άρτιου η περιττού και ο έλεγχος ισοτιμίας μπορεί επίσης να ενεργοποιείται.

Ένα διάγραμμα της USART παρουσιάζεται στο παρακάτω σχήμα.Τα κύρια λειτουργικά κομμάτια είναι η γεννήτρια ρολογιού, ο πομπός και ο δέκτης, τα οποία φαίνονται στα διακεκομμένα πλαίσια.



Εικόνα4. Γενική λειτουργία USART

Η γεννήτρια ρολογιών περιλαμβάνει την γεννήτρια του ρυθμού μετάδοσης που είναι σε θέση να παράγει ένα ευρύ φάσμα ρυθμών μετάδοσης της USART από οποιεσδήποτε συχνότητες που παράγει το ρολόι συστήματος.Αυτό αφαιρεί την ανάγκη να χρησιμοποιηθεί ένας εξωτερικός ταλαντωτής κρυστάλλου όπου με μία συγκεκριμένη συχνότητα επιτυγχάνει το απαραίτητο ρυθμό μετάδοσης.

Ο πομπός αποτελείται από έναν απομονωτή εγγραφής, έναν καταχωρητή ολίσθησης, και μια γεννήτρια ισοτιμίας.Ο απομονωτής εγγραφής επιτρέπει τη συνεχή μετάδοση δεδομένων χωρίς οποιαδήποτε καθυστέρηση μεταξύ των πλαισίων.

Ο δέκτης αποτελείται από έναν απομονωτή λήψης δύο επιπέδων και έναν καταχωρητή μετατόπισης.Μονάδες ανάκτησης ρολογιού και δεδομένων εξασφαλίζουν ισχυρό συγχρονισμό και φιλτράρισμα θορύβου κατά τη διάρκεια της ασύγχρονης λήψης δεδομένων. Περιλαμβάνει το λάθος πλαισίων, την υπερχείλιση απομονωτών και την ανίχνευση λάθους ισοτιμίας.

Παραγωγή ρολογιού

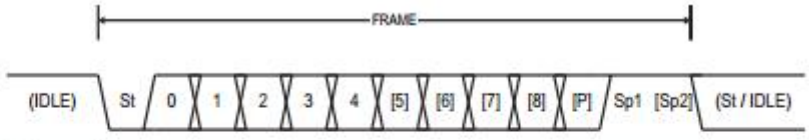
Το ρολόι που χρησιμοποιείται για την παραγωγή του ρυθμού μετάδοσης, την ολίσθηση και για την δειγματοληψία των bits δεδομένων παράγεται εσωτερικά από τη γεννήτρια κλασματικού ρυθμού μετάδοσης ή εξωτερικά από το pin μεταφοράς ρολογιού (XCK).

Μορφοποίηση πλαισίων

Η μεταφορά δεδομένων βασίζεται σε ένα πλαίσιο το οποίο αποτελείται από τα bits δεδομένων, τα bits συγχρονισμού (start and stop bits) και ένα προαιρετικό bit ισοτιμίας για τον έλεγχο λάθους. Η USART αποδέχεται όλους τους συνδυασμούς των ακόλουθων ως έγκυρες μορφές πλαισίου:

- 1 start bit
- 5, 6, 7, 8, η 9 bits δεδομένων
- άρτιο, η περιττό bit ισοτιμίας, όχι έλεγχος ισοτιμίας
- 1 ή 2 stop bits

Ένα πλαίσιο αρχίζει με το start bit, που ακολουθείται από όλα τα bits δεδομένων (το πρώτο bit είναι το ελάχιστα-σημαντικό και το τελευταίο bit το πιο πολύ-σημαντικό). Εάν είναι ενεργοποιημένο το bit ισοτιμίας προστίθεται μετά από τα bits δεδομένων, πριν από το πρώτο stop bit. Ένα πλαίσιο μπορεί να ακολουθηθεί άμεσα από ένα νέο bit έναρξης και ένα νέο πλαίσιο, ή η γραμμή επικοινωνίας μπορεί να επιστρέψει στην κατάσταση αδράνειας (high). Στο παρακάτω σχήμα απεικονίζονται πιθανοί συνδυασμοί σχημάτων πλαισίων.



St	:	Start bit, always low.
(n)	:	Data bits (0 to 8).
P	:	Parity bit, may be odd or even.
Sp	:	Stop bit, always high.
IDLE	:	No transfers on the communication line (RxD or TxD). The IDLE state is always high.

Εικόνα5. Μορφή πλαισίων USART

Υπολογισμός κομματιών ισοτιμίας

Άρτια ή περιττή ισοτιμία μπορεί να επιλεχτεί για τον έλεγχο λάθους. Εάν επιλέγεται η άρτια ισοτιμία,το bit ισοτιμίας τίθεται στο ‘‘1’’αν ο αριθμός της λογικής‘‘1’’των bits δεδομένων είναι περιττό (καθιστώντας ως άρτιο το συνολικό αριθμό αυτών). Εάν επιλέγεται η περιττή ισότητα, το bit ισοτιμίας τίθεται στο ‘‘1’’αν ο αριθμός της λογικής‘‘1’’των bits δεδομένων είναι άρτιος (καθιστώντας ως περιττό το συνολικό αριθμό αυτών).

Αρχικοποίηση USART

Η αρχικοποίηση της USART πρέπει να χρησιμοποιήσει την ακόλουθη αλληλουχία:

- Ρύθμιση του TxD pin στην τιμή high (‘‘1’’), και ρύθμιση προαιρετικά του XCK pin στο low(‘‘0’’).
- Ρύθμιση του TxD και προαιρετικά του XCK pin ως έξοδο.

- Ρύθμιση του ρυθμού μετάδοσης και της μορφοποίησης πλαισίων.
- Ρύθμιση του τρόπου λειτουργίας (ενεργοποιείται το XCK pin ως έξοδος στο σύγχρονο τρόπο λειτουργίας).
- Ενεργοποίηση του πομπού ή του δέκτη, ανάλογα με τη χρήση.

Για λειτουργία της USART οδηγούμενη από διακοπή (interrupt-driven), η γενική (global) διακοπή πρέπει να τεθεί εκτός λειτουργίας κατά τη διάρκεια της αρχικοποίησης. Πρίν γίνει μια επαναρχικοποίηση με αλλαγμένο ρυθμό μετάδοσης ή μορφοποίηση πλαισίου, πρέπει να είναι σίγουρο ότι δεν υπάρχει καμία τρέχουσα μετάδοση ενώ αλλάζουν οι καταχωρητές.

Πομπός δεδομένων - η USART ως πομπός

Όταν ο πομπός έχει ενεργοποιηθεί, η κανονική λειτουργία της θύρας του A pin αγνοείται από την USART και λαμβάνεται ως λειτουργία η σειριακή έξοδο του πομπού.Η κατεύθυνση του pin πρέπει να τεθεί ως έξοδο χρησιμοποιώντας τον καταχωρητή για την αντίστοιχη θύρα.

Αποστολή πλαισίων

Μια μετάδοση δεδομένων αρχίζει με τη φόρτωση του απομονωτή μετάδοσης (DATA) με τα δεδομένα που πρόκειται να σταλούν.Τα δεδομένα στον απομονωτή μετάδοσης μεταφέρονται στο καταχωρητή ολίσθησης όταν αυτός είναι άδειος και έτοιμος να στείλει ένα νέο πλαίσιο.Ο καταχωρητής ολίσθησης φορτώνεται αν είναι σε κατάσταση ηρεμίας (καμία τρέχουσα μετάδοση) ή αμέσως μετά τη μετάδοση το τελευταίου stop bit του προηγούμενου πλαισίου μετάδοσης. Όταν ο καταχωρητής ολίσθησης φορτωθεί με δεδομένα, θα μεταφέρει ένα πλήρες πλαίσιο.

Η σημαία διακοπής (TXCIF) ολοκλήρωσης της εκπομπής ενός πλαισίου τίθεται σε λογικό «1» και η προαιρετική διακοπή δημιουργείται όταν ολόκληρο το πλαίσιο του καταχωρητή ολίσθησης έχει μετατοπιστεί έξω και δεν υπάρχουν νέα δεδομένα στον απομονωτή μετάδοσης.

Ο καταχωρητής μετάδοσης δεδομένων (DATA) μπορεί να γραφτεί όταν η σημαία κενού του καταχωρητή δεδομένων (DREIF) είναι σε λογικό «1», δείχνοντας ότι ο καταχωρητής είναι κενός και έτοιμος για τα νέα δεδομένα.

Όταν χρησιμοποιούνται πλαίσια με λιγότερα από οκτώ bits,τα πιο σημαντικά bits που είναι γραμμένα στον καταχωρητή δεδομένων αγνοούνται.Αν χρησιμοποιούνται χαρακτήρες των 9-bit , το ένατο bit πρέπει να γραφτεί στο bit TXB8 πριν από το low byte του χαρακτήρα που είναι γραμμένο στα δεδομένα.

Απενεργοποίηση του πομπού

Η απενεργοποίηση του πομπού δεν θα τεθεί σε ισχύ εφόσον βρίσκονται σε εξέλιξη και εκκρεμούν μεταδόσεις που δεν έχουν ολοκληρωθεί.Δηλαδή, όταν ο καταχωρητής ολίσθησης και απομονωτής δεδομένων δεν περιέχουν δεδομένα για εκπομπή. Όταν ο πομπός απενεργοποιηθεί, θα παρακάμψει πλέον το pin TxDn, και η κατεύθυνση του pin θα οριστεί ως είσοδος αυτόματα από το υλικό, ακόμη και αν αυτό έχει διαμορφωθεί ως έξοδος από το χρήστη.

Λήψη Δεδομένων - Η USART ως Δέκτης

Όταν ο δέκτης ενεργοποιηθεί, το pin RxD λειτουργεί ως σειριακή είσοδο του δέκτη. Η κατεύθυνση του pin πρέπει να οριστεί ως είσοδος, η οποία είναι η προεπιλεγμένη ρύθμιση του pin.

Λήψη Πλαισίων

Ο δέκτης ξεκινά λήψη δεδομένων όταν ανιχνεύει ένα έγκυρο bit έναρξης. Κάθε bit που ακολουθεί το bit έναρξης θα δειγματοληφθεί στο ρυθμό μετάδοσης ή στο ρολόι XCK και μετατοπίζεται στη λήψη του καταχωρητή ολίσθησης μέχρις ότου ληφθεί το πρώτο bit διακοπής του πλαισίου.

Ένα δεύτερο bit διακοπής θα αγνοηθεί από τον δέκτη. Όταν το πρώτο bit διακοπής ληφθεί και ένα πλήρες σειριακό πλαίσιο είναι παρόν στη λήψη του καταχωρητή ολίσθησης, τα περιεχόμενα του καταχωρητή ολίσθησης θα μετακινηθούν στον buffer λήψης. Η διακοπή της σημαίας λήψης (RXCIF) τίθεται σε «1», και δημιουργείται μία προαιρετική διακοπή.

Τα δεδομένα δεν θα πρέπει να διαβαστούν αν δεν έχει τεθεί σε λογικό «1» η σημαία διακοπής πλήρης λήψης ενός πλαισίου. Όταν χρησιμοποιούνται πλαίσια με λιγότερα από οκτώ bits, τα αχρησιμοποίητα πιο σημαντικά bits διαβάζονται ως μηδέν. Αν χρησιμοποιούνται χαρακτήρες 9-bit, το ένατο bit θα πρέπει να διαβαστεί από το bit RXB8 πριν από το low byte του χαρακτήρα που διαβάζεται από τα δεδομένα.

Σημαίες σφάλματος δεκτών

Ο δέκτης USART έχει τρεις σημαίες σφάλματος. Οι σημαίες σφαλμάτων πλαισίου (FERR), η υπερχείλιση (BUFOVF) και ισοτιμίας (PERR) είναι προσβάσιμες από τον καταχωρητή κατάστασης. Οι σημαίες σφάλματος βρίσκονται στον απομονωτή (buffer) λήψης FIFO μαζί με το αντίστοιχο πλαίσιο τους. Λόγω της τεχνικής buffering των σημαιών σφάλματος, ο καταχωρητής κατάστασης πρέπει να διαβάσετε πριν από (DATA), δεδομένου ότι διαβάζοντας τον buffer λήψης αλλάζει ο απομονωτής FIFO.

Έλεγχος Ισοτιμίας

Όταν είναι ενεργοποιημένο, ο ελεγκτής ισοτιμίας υπολογίζει την ισοτιμία των bit δεδομένων στα εισερχόμενα πλαίσια και συγκρίνει το αποτέλεσμα με το bit ισοτιμίας του αντίστοιχου πλαισίου. Εάν ανιχνευθεί ένα σφάλμα ισοτιμίας, η σημαία σφάλματος ισοτιμίας τίθεται στο ‘‘1’’.

Απενεργοποίηση του δέκτη

Μία απενεργοποίηση του δέκτη θα είναι άμεση. Ο απομονωτής του δέκτη καθαρίζεται, και τα δεδομένα θα χαθούν από τις τρέχουσες υποδοχές.

Άδειασμα του buffer λήψης

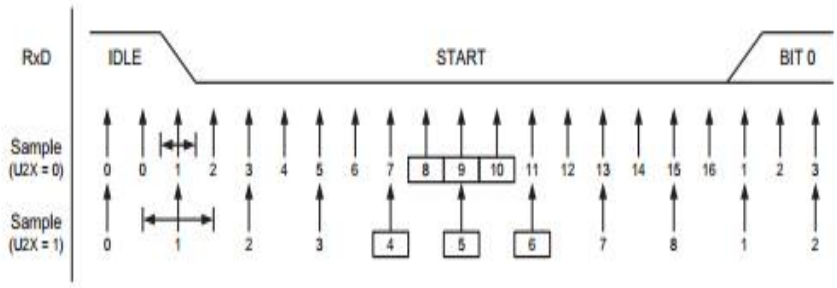
Αν ο απομονωτής λήψης πρέπει να αδειάσει κατά τη διάρκεια της κανονικής λειτουργίας, διαβάζεται η θέση δεδομένων μέχρι ότου να καθαριστεί σημαία διακοπής πλήρης λήψης.

Ασύγχρονη λήψη δεδομένων

Η USART περιλαμβάνει ανάκτηση ρολογιού και μια μονάδα ανάκτησης δεδομένων για το χειρισμό ασύγχρονης λήψης δεδομένων. Η μονάδα ανάκτησης ρολογιού χρησιμοποιείται για το συγχρονισμό των εισερχόμενων ασύγχρονων σειριακών πλαισίων στο pin RxD με το εσωτερικό ρολόι δημιουργίας του ρυθμού μετάδοσης. Δειγματοληπτεί και φιλτράρει τις χαμηλές συχνότητες κάθε εισερχόμενουbit, βελτιώνοντας έτσι την αντοχή του δέκτη στο θόρυβο. Το εύρος λειτουργίας της ασύγχρονης λήψης εξαρτάται από την ακρίβεια του εσωτερικού ρολογιού του ρυθμού μετάδοσης, το ρυθμό των εισερχόμενων πλαισίων, καθώς και το μέγεθος του πλαισίου σε αριθμό από bits.

Ανάκτηση ρολογιού

Η μονάδα ανάκτησης ρολογιού συγχρονίζει το εσωτερικό ρολόι με τα εισερχόμενα σειριακά πλαίσια. Η παρακάτω εικόνα παρουσιάζει τη διαδικασία δειγματοληψίας για το bit έναρξης ενός εισερχόμενου πλαισίου. Το ποσοστό του δείγματος είναι 16 φορές ο ρυθμός μετάδοσης για την κανονική λειτουργία, και οκτώ φορές ο ρυθμός μετάδοσης για τη λειτουργία διπλής ταχύτητας. Τα οριζόντια βέλη απεικονίζουν τη μεταβολή συγχρονισμού λόγω της διαδικασίας δειγματοληψίας. Σημειώστε την μεγαλύτερη μεταβολή χρόνου, όταν χρησιμοποιείτε η διπλή λειτουργία ταχύτητα λειτουργίας.



Εικόνα7. Δειγματοληψία start bit

Όταν η λογική ανάκτησης ρολογιού ανιχνεύει ένα high (αδράνεια) σε low (εκκίνηση) μετάβαση στη γραμμή RxD, η αλληλουχία ανίχνευσης bit έναρξης αρχίζει.Το δείγμα 1 υποδηλώνει το πρώτο μηδενικό δείγμα, όπως φαίνεται στην εικόνα. Η λογική ανάκτησης ρολογιού κατόπιν χρησιμοποιεί δείγματα 8, 9 και 10 για την κανονική λειτουργία και τα δείγματα 4, 5 και 6 για τη λειτουργία διπλής ταχύτητας για να αποφασίσει εάν έχει λάβει ένα έγκυρο bit έναρξης. Αν δύο ή τρία δείγματα έχουν low level, το bit έναρξης γίνεται δεκτό. Η μονάδα ανάκτησης ρολογιού συγχρονίζεται, και η ανάκτηση δεδομένων μπορεί να ξεκινήσει.Αν δύο ή τρία δείγματα έχουν ένα high level,το bit έναρξης απορρίπτεται ως θόρυβος, και ο δέκτης αναζητά την επόμενη μετάβαση high-low. Η διαδικασία αυτή επαναλαμβάνεται για κάθε bit έναρξης.

1.3. GSM Modem

Ένα μόντεμ GSM είναι ένα εξειδικευμένο τύπου μόντεμ που δέχεται μια κάρτα SIM, και λειτουργεί πάνω από μια συνδρομή σε εταιρεία κινητής τηλεφωνίας, ακριβώς όπως ένα κινητό τηλέφωνο. Από την οπτική γωνία κινητής τηλεφωνίας, ένα μόντεμ GSM μοιάζει ακριβώς όπως ένα κινητό τηλέφωνο.

Όταν ένα μόντεμ GSM είναι συνδεδεμένο με έναν υπολογιστή, αυτό επιτρέπει στον υπολογιστή να χρησιμοποιήσει το μόντεμ GSM για να επικοινωνεί μέσω του δικτύου κινητής τηλεφωνίας.Τα μόντεμ GSM χρησιμοποιούνται συχνά για την σύνδεση ενός συστήματος στο internet, ενώ σε άλλες περιπτώσεις μπορούν να χρησιμοποιηθούν για την αποστολή και λήψη μηνυμάτων SMS και MMS.

1.3.1. Υπηρεσία Σύντομων Γραπτών Μηνυμάτων

Μία από τις πιο επιτυχημένες υπηρεσίες που προσέφεραν τα δίκτυα κινητής τηλεφωνίας είναι αυτή των σύντομων γραπτών μηνυμάτων (SMS, Short Message Service). Όπως και οι AT εντολές έτσι και τα sms καθορίζονται με βάση τα πρότυπα του Ευρωπαϊκού Οργανισμού ETSI (πρότυπα GSM).

Είναι μία ασύμμετρη υπηρεσία και δεν χρειάζεται να υπάρχει δέσμευση φάσματος για τους κινητούς σταθμούς. Η αποστολή του μηνύματος από τον αποστολέα μπορεί να γίνει είτε είναι ο παραλήπτης ενεργός στο δίκτυο είτε όχι.Για αυτό και οι υπηρεσίες αποστολής και λήψης θεωρούνται ως ξεχωριστές υπηρεσίες. Βέβαια υπάρχει και η ανάγκη ενός κόμβου μεταξύ των δύο σταθμών.Έτσι ενώ ο αποστολέας στέλνει το μήνυμα που συντάσσει στον παραλήπτη αυτό φτάνει πρώτα στο κέντρο μεταγωγής της εταιρίας (SMSC enter ,SMSC) και από εκεί προωθείται στον πραγματικό παραλήπτη μαζί με κάποιες πρόσθετες πληροφορίες για την ορθή παράδοση του. Σύμφωνα με το πρότυπο του SMS και όπως προσδιορίστηκε από τον οργανισμό ETSI για το κάθε μήνυμα χρησιμοποιούνται 160 χαρακτήρες.

Υπάρχουν δύο τρόποι αποστολής και λήψης των sms μηνυμάτων. Διαφέρει η δομή τους,η σύνταξη κάποιων εντολών, η υποστήριξη κάποιων χαρακτηριστικών, ακόμη και ευκολία χρήσης της κάθε μμεθόδου:

1. Ο πρώτος είναι η αποστολή του μηνύματος σε μορφή κειμένου (text mode).Έχει πολύ απλή δομή. Στην αρχή του μηνύματος τοποθετείται ο αριθμός του παραλήπτη και ακολουθεί το μήνυμα.
2. Ο δεύτερος τρόπος του pdu mode (Protocol Data Unit) στέλνει μια ακολουθία δεκαεξαδικών χαρακτήρων ακολουθώντας συγκεκριμένη δομή. Εσωκλείει διάφορες πληροφορίες όπως το νούμερο του αποστολέα, το κέντρο μεταγωγής σύντομων μηνυμάτων (SMS Center), την ώρα αποστολής, την ημερομηνία καθώς και άλλες πληροφορίες.Τα δεδομένα κάθε τμήματος είναι γραμμένα σε αλφαριθμητική μορφή που ονομάζεται hexadecimal-octets και semidecimal-octets (οι ορολογίες αυτές εισάγονται από τον οργανισμό ETSI).

1.3.2. AT Εντολές

Οι συσκευές modem,όπως και τα κινητά τηλέφωνα,έχουν τη δυνατότητα εκτέλεσης διαφόρων λειτουργιών. Για παράδειγμα η δυνατότητα να διαβαστεί ένα γραπτό μήνυμα που υπάρχει σε κάποια θέση μνήμης,να συνταχθεί και να αποσταλεί κάποιο μήνυμα,η διαχείριση του τηλεφωνικού καταλόγου,ή η μέτρηση της έντασης του λαμβανόμενου σήματος είναι διαδικασίες που γίνονται εφικτές με τη χρήση AT εντολών (ATcommands).

Τα γράμματα AT, που προέρχονται από τη λέξη Attention που σημαίνει προσοχή,δηλώνουν στο modem ότι ακολουθεί το κύριο μέρος της εντολής που πρέπει να εκτελεσθεί. Είναι εντολές με συγκεκριμένη δομή αναγνωρίσιμες και εκτελέσιμες από το modem που αφορούν τις περισσότερες από τις λειτουργίες τις συσκευής.Η σύνταξη γίνεται γράφοντας πρώτα το πρόθεμα AT ακολουθούμενο από το σύμβολο της πρόσθεσης (+) και το όνομα κάθε εντολής. Στη συνέχεια ακολουθεί το σύμβολο της ισότητας (=) καθώς και τα ορίσματα που τυχόν δέχεται η εντολή. Το τέλος της εντολής σηματοδοτείται από τον χαρακτήρα [CR] (Carriage Return) που ταυτόχρονα σημαίνει την έναρξη εκτέλεσης της εντολής.

Ακολουθεί η μορφή και η λειτουργία κάθε εντολής που χρησιμοποιήθηκε κατά την σχεδίαση και υλοποίηση του συστήματος που υλοποιήθηκε στα πλαίσια της διπλωματικής εργασίας.

AT+CMGF	
Περιγραφή:	Η εντολή AT+CMGF θέτει μία παράμετρο η οποία προσδιορίζει τη μορφή των εισερχομένων και εξερχομένων μηνυμάτων που θα χρησιμοποιηθούν.
Σύνταξη:	*AT+CMGF =[<mode>] <cr>*
Παράμετροι:	<mode> = 0 ή 1
	0 = PDUmode. **1** = TEXT mode. (DEFAULT)

AT+CSMS	
Περιγραφή:	Η εντολή AT+CSMS επιλέγει την υπηρεσία μηνυμάτων.
Σύνταξη:	*AT+CSMS=<service><mt><mo><bm><cr>*

Παράμετροι: **<service>** = 0, 1, ή 128

0 = GSM 03.40 και 03.41 (η σύνταξη των AT εντολών ελέγχου των SMS είναι συμβατή με GSM 07.05 Phase 2 version 4.7.0). (DEFAULT)
1 = GSM 03.40 and 03.41 (η σύνταξη των AT εντολών ελέγχου των SMS είναι συμβατή με GSM 07.05 Phase 2+ version)
128 = συμβατότητα με Phase 1 και τύπο συσκευής M1 (συγκεκριμένου κατασκευαστή).

<mt>= mobile-terminated messages – 0 or 1

Τα μηνύματα αυτά αναφέρονται σε μηνύματα SMS που στέλνονται από μια κινητή συσκευή σε ένα κέντρο μηνυμάτων SMS, δηλαδή τα εξερχόμενα μηνύματα SMS.

0 = δεν υποστηρίζεται.
1 = υποστηρίζεται.

<mo> = mobile-originatedmessages – 0 or 1

Τα μηνύματα αυτά αναφέρονται σε μηνύματα SMS που στέλνονται σε μια κινητή συσκευή από ένα κέντρο μηνυμάτων SMS, δηλαδή τα εισερχόμενα μηνύματα SMS.

0 = δεν υποστηρίζεται.
1 = υποστηρίζεται.

<bm> = broadcast-type messages – 0 or 1

Τα μηνύματα αυτά στέλνονται από τον φορέα εκμετάλλευσης του δικτύου σε μία ομάδα συνδρομητών. Αυτά τα μηνύματα κειμένου ενδέχεται να περιέχουν ειδήσεις, πληροφορίες για τον καιρό, κ.λπ.

0 = δεν υποστηρίζεται.
1 = υποστηρίζεται.

AT+CSCS

Περιγραφή: Η εντολή αυτή επιλέγει το σετ χαρακτήρων του modem.

Σύνταξη: *AT+CSCS=<character set><cr>*

Παράμετροι: **<characterset>**: το σετ χαρακτήρων του modem. Πιθανές τιμές είναι: "GSM", "HEX", "IRA", "PCDN", "UCS2", "UTF-8" κλπ.

AT+CMGS

Περιγραφή: Η εντολή AT+CMGS εκπέμπει ένα μήνυμα SMS από το modem προς το δίκτυο ασύρματης επικοινωνίας. Η αριθμός της αναφοράς μηνύματος <mr> επιστρέφεται στο modem ως ένδειξη παράδοσης του μηνύματος στο κέντρο μηνυμάτων SMS.

Ο χαρακτήρας που παράγεται από το συνδυασμό πλήκτρων CTRL-Z χρησιμοποιείται για να οριοθετήσει το τέλος του μηνύματος. Όταν το modem λάβει το χαρακτήρα αυτόν, τερματίζει την εισαγωγή και αποστέλλει το μήνυμα. Σε περίπτωση επιτυχούς αποστολής επιστρέφει στο χρήστη "OK" διαφορετικά "ERROR" και τον αντίστοιχο κωδικό λάθους.

Σύνταξη: Text mode

(+CMGF=1):+CMGS="<da>"<cr><text>

<ctrl-Z/ESC>

PDU mode

(+CMGF=0):+CMGS=<length><cr>

<PDU>

<ctrl-Z/ESC>

Παράμετροι: **<da>:** Ο αριθμός του παραλήπτη.

<text>: Κείμενο που αποτελείται από χαρακτήρες ASCII.

Παράδειγμα:

AT+CMGS="+1234567890"<CR>

Hello world!

<Ctrl+z>

<length>:είναι η τιμή η οποία δείχνει το πλήθος των octets της ακολουθίας.

<PDU>: Το περιεχόμενο του μηνύματος σε δεκαεξαδική μορφή.

Παράδειγμα:

AT+CMGS=32

<CR>0011000B916407281553F80000AA0AE8329BFD4697D9EC37

<Ctrl+z>

AT+CSMP

Περιγραφή: Η εντολή AT+CSMP επιλέγει τιμές για τις επιπλέον παραμέτρους που απαιτούνται όταν ένα μήνυμα SMS στέλνεται στο δίκτυο ασύρματης επικοινωνίας ή αποθηκεύεται στη μνήμη του modem όταν επιλέγεται το μήνυμα σε μορφή κειμένου (text mode). Είναι δυνατόν να τεθεί η περίοδος εγκυρότητας <vp> ξεκινώντας από την στιγμή που το μήνυμα λαμβάνεται από το κέντρο ή να οριστεί ο απόλυτος χρόνος τερματισμού της περιόδου εγκυρότητας του.

Σύνταξη: *AT+CSMP=<fo>[<vp>[,pid>[,<dcs>]]]* <cr>

Παράμετροι: **<fo>:** first octet of GSM 03.40

SMS-DELIVER, SMS SUBMIT (DEFAULT=17), ή SMS COMMAND (DEFAULT=16)

<vp>: TP-Validity-Period

(DEFAULT=167).

<pid>: protocol-identifier

(DEFAULT=0).

<dcs>: SMS Data Coding Scheme

(DEFAULT=0).

AT+CPIN

Περιγραφή: Η εντολή αυτή εισάγει τους κωδικούς αριθμούς (CHV1/CHV2/PUK1/PUK2, κλπ.), στο GSM modem τα οποία είναι απαραίτητα πριν χρησιμοποιηθεί οποιαδήποτε λειτουργία του modem. Οι CHV1/CHV2 είναι συνήθως 4 ψηφία ενώ οι PUK1/PUK2 είναι 8 ψηφία.

Εάν η εφαρμογή προσπαθήσει να κάνει μια εξερχόμενη κλήση πριν επιβεβαιωθεί ο κωδικός PIN της κάρτας SIM (CHV1), τότε το modem θα αρνηθεί επιστρέφοντας "+CMEERROR: 11" (απαιτείται το PIN της κάρτας SIM).

Η εφαρμογή είναι υπεύθυνη για τον έλεγχο του PIN μετά από κάθε επανεκκίνηση ή αν το PIN είναι ενεργοποιημένο.

Σύνταξη: *AT+CPIN=<pin><cr>*

Παράδειγμα: AT+CPIN=1234

Μετά από 3 αποτυχημένες προσπάθειες εισαγωγής του κωδικού PIN (Personal Identification Number), θα απαιτηθεί το PUK (Personal Unblocking Key). Η επικύρωση του PUK αναγκάζει το χρήστη να εισάγει έναν νέο κωδικό ως δεύτερη παράμετρο και θα είναι το νέο PIN εφόσον η επικύρωση του PUK είναι επιτυχής. Για το λόγο αυτό χρησιμοποιείται η εντολή:

AT+CPIN=<Puk>,<NewPin><cr>

Παράδειγμα AT+CPIN=12345678,1234

Για να υπολογιστεί ποιος κωδικός θα πρέπει να εισαχθεί (ή όχι), πρέπει να χρησιμοποιηθεί η επόμενη εντολή:

AT+CPIN?

Οι πιθανές απαντήσεις είναι:

+CPIN: READY	Το modem δεν είναι σε εκκρεμότητα για κάθε κωδικό
+CPIN: SIM PIN	CHV1 απαιτείται
+CPIN: SIM PUK	PUK1 απαιτείται
+CPIN: SIM PIN2	CHV2απαιτείται
+CPIN: SIM PUK2	PUK2 απαιτείται
+CPIN: PH-SIM PIN	Κλείδωμα SIM (modem-SIM) απαιτείται
+CPIN: PH-NET PIN	Προσωπικοποίηση δικτύου απαιτείται
+CME ERROR: <err>	Αποτυχία SIM

ATE

Περιγραφή: Η εντολή αυτή χρησιμοποιείται για την αποφυγή του φαινομένου της επιστροφής χαρακτήρων (echo) κατά την διαδικασία αποστολής εντολών προς το modem.

Σύνταξη: *ATE<n><cr>*

Παράμετροι: <n> = 0 ή 1

0 = δεν επιστρέφονται οι χαρακτήρες. **1** = επιστρέφονται οι χαρακτήρες.

1.4. Global Positioning System (GPS)

Το GPS (Global Positioning System), είναι ένα παγκόσμιο σύστημα εντοπισμού θέσης, το οποίο βασίζεται σε ένα "πλέγμα" είκοσι τεσσάρων τεχνητών δορυφόρων της Γης, και ειδικές συσκευές, οι οποίες ονομάζονται "δέκτες GPS" Οι δέκτες αυτοί παρέχουν ακριβείς πληροφορίες για τη θέση ενός σημείου, το υψόμετρό του, την ταχύτητα και την κατεύθυνση της κίνησης του. Επιπλέον,σε συνδυασμό με ένα ειδικό λογισμικό χαρτογράφησης μπορούν να απεικονίσουν τις πληροφορίες αυτές.

Αυτό το σύστημα εντοπισμού συνδυάζει τρείς πρωτοποριακές τεχνολογίες: Το GSM, το GPS καθώς επίσης και το διαδίκτυο (internet).

Η τοποθεσία του αντικειμένου που φέρει τη συσκευή εντοπισμού μεταδίδεται σε πραγματικό χρόνο στο διακομιστή της (server), ο οποίος είναι υπεύθυνος να παρακολουθεί αν υπάρχουν νέες αιτήσεις εντοπισμού και αν υπάρχουν νέα μηνύματα με GPS περιεχόμενο(πχ συντεταγμένες,ημερομηνία). Ο server αναλαμβάνει όλη τη δουλειά διαχείρισης των μηνυμάτων SMS και των στιγμάτων εντοπισμού.

Σε γενικές γραμμές οι GPS δέκτες αποτελούνται από μια κεραία, συντονισμένη στις συχνότητες που μεταδίδουν οι δορυφόροι, από επεξεργαστές δεκτών και ένα εξαιρετικά σταθερό ρολόι (συνήθως ένας κρυσταλλικός ταλαντωτής).

1.4.1. Πρωτόκολλο NMEA

Το πρωτόκολλο NMEA είναι εύχρηστο,διότι προσφέρει πληροφορίες σε μορφή προτάσεων που περιέχουν χαρακτήρες κωδικοποιημένων σύμφωνα με το πρωτόκολλο ASCII και συνεπώς καθιστά ευκολότερη την επεξεργασία των πληροφοριών με τη βοήθεια ενός απλού parser που μπορούμε να δημιουργήσουμε στη γλώσσα προγραμματισμού C.

Οι προτάσεις του πρωτοκόλλου NMEA αποστέλλονται μέσω μιας σειριακής θύρας επικοινωνιών (serial port). Οι προτάσεις μεταφέρονται με διαχωριστικούς χαρακτήρες «LF» (Line Feed) και «CR» (Carriage Return) (0x0D και 0x0A, ή '\r\n' αντίστοιχα). Οι προτάσεις ξεκινούν με «$» ή με τον «!» και έχουν μέγιστο μήκος 80 χαρακτήρων και ελάχιστο 6. Το πρωτόκολλο ορίζει ότι οι πληροφορίες μέσα στις προτάσεις χωρίζονται από χαρακτήρες κόμμα «,». Στο τέλος κάθε πρότασης υπάρχει ένας χαρακτήρας «*» και μετά από αυτόν υπάρχει το άθροισμα ελέγχου (check sum). Το check sum χρησιμεύει για να πιστοποιηθεί η ορθότητα των δεδομένων από και προς το GPS module και είναι απαραίτητο σε ορισμένες προτάσεις.Κάθε πρόταση τερματίζει με τους χαρακτήρες <CR> και <LF>,οι οποίοι δεν είναι εκτυπώσιμοι, αλλά γίνονται αντιληπτοί από το GPS module και τον processor που θα επεξεργαστεί τα δεδομένα.Οι δύο πρώτοι χαρακτήρες ενημερώνουν για το είδος της συσκευής που στέλνει το GPS.

1.4.2. GPRMC

Το πρωτόκολλο NMEA υποστηρίζει μία σειρά από προτάσεις, κάθε μία εκ των οποίων παρέχει στο χρήστη πληροφορίες σχετικά με τη γεωγραφική θέση, το παγκόσμιο χρόνο, την ποιότητα του σήματος κλπ. Ωστόσο, η GPRMC (Recommended minimum specific GPS/Transitdata) είναι αυτή η οποία συνδυάζει όλες τις απαραίτητες πληροφορίες σε μία πρόταση:

```
$GPRMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,ddmmyy,x.x,a*hh

          1     2    3     4     5    6   7   8     9    10  11 12


1    = UTC of position fix
2    = Data status (V=navigation receiver warning)
3    = Latitude of fix
4    = (N)orth or (S)outh
5    = Longitude of fix
6    = (E)ast or (W)est
7    = Speed over ground in knots
8    = Track made good in degrees True
9    = UT date
10   = Magnetic variation degrees
11   = (E)ast or (W)est
12   = Checksum
```

Παράδειγμα

```
$GPRMC,220516,A,5133.82,N,00042.24,W,173.8,231.8,130694,004.2,W*70
          1   2    3    4     5     6   7     8      9     10  11 12
```

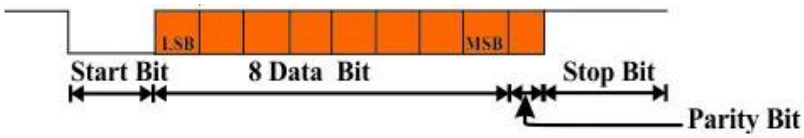
1	220516	Time Stamp
2	A	validity – A=ok, V=invalid
3	5133.82	Current Latitude
4	N	North/South
5	00042.24	Current Longitude
6	W	East/West
7	173.8	Speed in knots
8	231.8	True course
9	130694	Date Stamp
10	004.2	Variation
11	W	East/West
12	70	Checksum

1.5. Η Σειριακή διασύνδεση RS232

Η διασύνδεση με ένα GSM και GPS περιφερειακό υποσύστημα στηρίζεται συνήθως στη σειριακή διασύνδεση RS232 η οποία είναι ένας δημοφιλής τρόπος για να μεταφέρονται εντολές και δεδομένα μεταξύ ενός προσωπικού υπολογιστή και ενός μικροελεγκτή.Το πρωτόκολλο RS-232 καθορίζει τα σήματα που χρησιμοποιούνται στην επικοινωνία, και τον τρόπο για να μεταφέρονται τα σήματα μεταξύ των συσκευών. Το RS-232 είναι μία από τις πιο ευρέως χρησιμοποιούμενες τεχνικές που χρησιμοποιούνται για τη διασύνδεση των εξωτερικών υποσυστημάτων των υπολογιστών.

1.5.1. Το πλαίσιο RS232

Ένα πλαίσιο είναι ένα πλήρες πακέτο από bits. Στην ασύγχρονη σειριακή επικοινωνία του πρωτοκόλλου RS-232, το πλαίσιο αποτελείται από ένα ψηφίο έναρξης (start bit), επτά ή οκτώ δυαδικά ψηφία δεδομένων, ένα ψηφίο ισοτιμίας (parity bit), και ένα ή δύο ψηφία τερματισμού (stop bits). Ένα τυπικό διάγραμμα χρονισμού για ένα RS-232 πλαίσιο που αποτελείται από ένα ψηφίο έναρξης, 8 δυαδικά ψηφία δεδομένων, ένα ψηφίο ισοτιμίας και δύο δυαδικά ψηφία τερματισμού δείχνεται στην παρακάτω εικόνα. Η ακριβής δομή του πλαισίου θα πρέπει να συμφωνηθεί τόσο από τον πομπό όσο και από τον δέκτη πριν ανοίξει ο δίαυλος επικοινωνίας.



Εικόνα 8. RS-232 πλαίσιο

Το start bit χρησιμοποιείται για να σηματοδοτήσει την έναρξη ενός πλαισίου και το stop bit χρησιμοποιείται για να σηματοδοτήσει το τέλος του πλαισίου.

Το parity bit χρησιμοποιείται για την ανίχνευση των σφαλμάτων μετάδοσης. Υπάρχουν οι εξής ισοτιμίες: Even, Odd, Mark και Space. Για άρτια (Even) ή περιττή (Odd) ισοτιμία, ο πομπός θέτει το ψηφίο ισοτιμίας (το τελευταίο ψηφίο μετά τα δυαδικά ψηφία δεδομένων) σε μία δυαδική τιμή που θα εξασφαλίσει ότι το πλαίσιο περιέχει έναν ζυγό ή μονό αριθμό ψηφίων λογικού «1». Για παράδειγμα, εάν τα δεδομένα είναι 10010010, για άρτια ισοτιμία ο πομπός θα θέσει το ψηφίο ισοτιμίας στο λογικό «1» έτσι ώστε να κρατήσει τον αριθμό των ψηφίων λογικού «1» σε ζυγό αριθμό. Για περιττή ισοτιμία ο πομπός θα θέσει το ψηφίο ισοτιμίας στο λογικό «0» έτσι ώστε να κρατήσει τον αριθμό των ψηφίων λογικού «1» σε μονό αριθμό. Η ισοτιμία Mark θέτει το ψηφίο ισοτιμίας σε λογικό «1» ενώ η Space θέτει το ψηφίο ισοτιμίας σε λογικό «0», έτσι ώστε ο δέκτης να υπολογίσει εάν τα δεδομένα έχουν αλλοιωθεί.

Είναι φανερό ότι ο δέκτης και ο πομπός πρέπει να χρησιμοποιούν την ίδια ισοτιμία. Όταν το πλαίσιο ληφθεί, τότε ο δέκτης ελέγχει την ισοτιμία του λαμβανόμενου πλαισίου. Αν η ισοτιμία είναι λάθος, τότε ο δέκτης αναγνωρίζει

σφάλμα κατά τη μετάδοση και μπορεί να ζητήσει από τον πομπό εκ νέου την αποστολή του πλαισίου.

Το πρόβλημα με την ανίχνευση σφαλμάτων, χρησιμοποιώντας το ψηφίο ισοτιμίας είναι ότι αν δύο bits είναι λάθος τότε ο έλεγχος ισοτιμίας θα αποτύχει. Αυτό συμβαίνει επειδή κάθε σφάλμα ακυρώνει το αποτέλεσμα του άλλου (από την άποψη του υπολογισμού ισοτιμία). Κάθε άρτιος αριθμός σφαλμάτων προκαλεί αποτυχία στην ανίχνευση σφάλματος. Επειδή προσφέρει χαμηλή απόδοση στην ανίχνευση των σφαλμάτων, σε περιπτώσεις όπου η πιθανότητα σφάλματος είναι εξαιρετικά μικρή, είναι σύνηθες να μην χρησιμοποιείται ψηφίο ισοτιμίας.

1.5.2. Ρυθμός Μετάδοσης (Baud Rate)

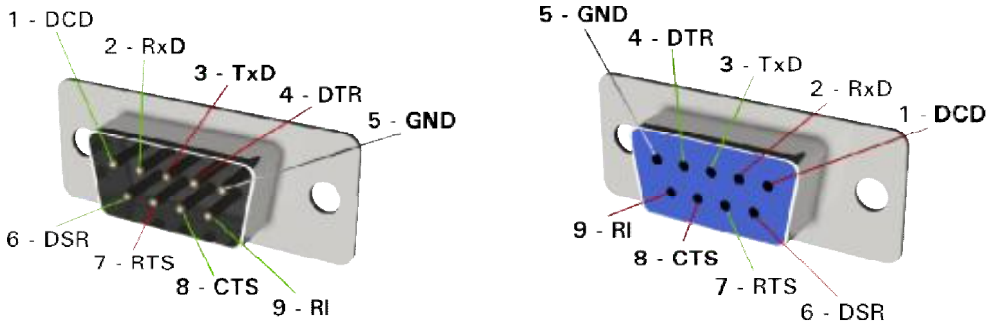
Ο ρυθμός μετάδοσης (baud rate) είναι η ταχύτητα επικοινωνίας που μετρά τον αριθμό των μεταβιβάσεων δυαδικών ψηφίων ανά δευτερόλεπτο. Για παράδειγμα, 19.200 baud είναι 19200 ψηφία ανά δευτερόλεπτο.

Το πρωτόκολλο RS-232 περιγράφεται ως ασύγχρονο, καθώς δεν υπάρχει ρολόι που να μεταδίδεται ταυτόχρονα με τα δεδομένα. Αντ 'αυτού μία διαφορετική μέθοδος ανάκτησης ρολογιού χρησιμοποιείται.

Κατά την έναρξη της αποστολής, μεταδίδεται το ψηφίο έναρξης υποδεικνύοντας στον δέκτη ότι ένα byte δεδομένων πρόκειται να ακολουθήσει. Δηλ. η μετάβαση του ψηφίου έναρξης (από λογικό «1» σε λογικό «0») χρησιμοποιείται για το συγχρονισμό του πομπού και του δέκτη. Κατόπιν, ο δέκτης γνωρίζοντας το ρυθμό μετάδοσης δηλ. το χρόνο t_{bit} που διαρκεί ένα ψηφίο του RS-232 πλαισίου, δειγματοληπτεί αρχικά την γραμμή μετάδοσης σε χρόνο $t_{bit}/2$ και στη συνέχεια κάθε t_{bit} μέχρι το ψηφίο τερματισμού. Το ψηφίο τερματισμού (από λογικό «0» σε λογικό «1») είναι το αντίθετο του ψηφίου έναρξης και επιτρέπει στον πομπό να στείλει ένα νέο πλαίσιο. Με τον τρόπο αυτό ο δέκτης δειγματοληπτεί τα δυαδικά ψηφία στο κέντρο της χρονικής τους διάρκειας δημιουργώντας μία αξιόπιστη ασύγχρονη σειριακή σύνδεση.

1.5.3. Συνδέσεις RS-232

Σε μία σειριακή επικοινωνία RS-232 συμμετέχουν δύο ειδών διαφορετικές συσκευές: Η τερματική συσκευή δεδομένων (Data Terminal Equipment, DTE) και η συσκευή ελέγχου δεδομένων (Data Control Equipment, DCE). Ο υπολογιστής είναι ένα DTE ενώ ένα μόντεμ είναι DCE. Συνήθως, η DTE έρχεται με ένα αρσενικό βύσμα, ενώ η DCE έρχεται με ένα θηλυκό βύσμα.



(α) **(β)**

Εικόνα 9. Connectors RS-232 τύπου DB9.(α) Αρσενικός και (β) Θηλυκός

Ωστόσο, αυτό δεν είναι πάντα αλήθεια. Μπορείτε να χρησιμοποιήσετε τον παρακάτω απλό τρόπο για να επιβεβαιώσετε αν μία συσκευή είναι DCEή DTE: Μετρήστε το pin 3 και το pin 5 της DB-9 με ένα βολτόμετρο, και αν πάρετε μια τάση -3Vέως -15V, τότε είναι μια συσκευή DTE. Εάν η τάση αυτή αντιστοιχεί στο pin 2, τότε είναι μια συσκευή DCE.

Ακολουθεί η περιγραφή των σημάτων RS-232, ανάλογα εάν μία συσκευή είναι DTE ή DCE:

PIN	DTE		DCE	
1	DCD	Data Carrier Detect	DCD	Data Carrier Detect
2	RxD	Receive Data	TxD	Transmit Data
3	TxD	Transmit Data	RxD	Receive Data
4	DTR	Data Terminal Ready	DSR	Data Set Ready
5	GND	Signal Ground	GND	Signal Ground
6	DSR	Data Set Ready	DTR	Data Terminal Ready
7	RTS	Request to Send	CTS	Clear to Send
8	CTS	Clear to Send	RTS	Request to Send
9	RI	Ring Indicator	RI	Ring Indicator

Πίνακας 1. Σήματα RS-232

Το μέγεθος του καλωδίου για σύνδεση RS-232 είναι περίπου 15 μέτρα αλλά στην πραγματικότητα εξαρτάται από το ρυθμό μετάδοσης.

Ρυθμός Μετάδοσης (bps)	Απόσταση (m)
2400	60
4800	30
9600	15
19200	7.6
38400	3.7
56000	2.6

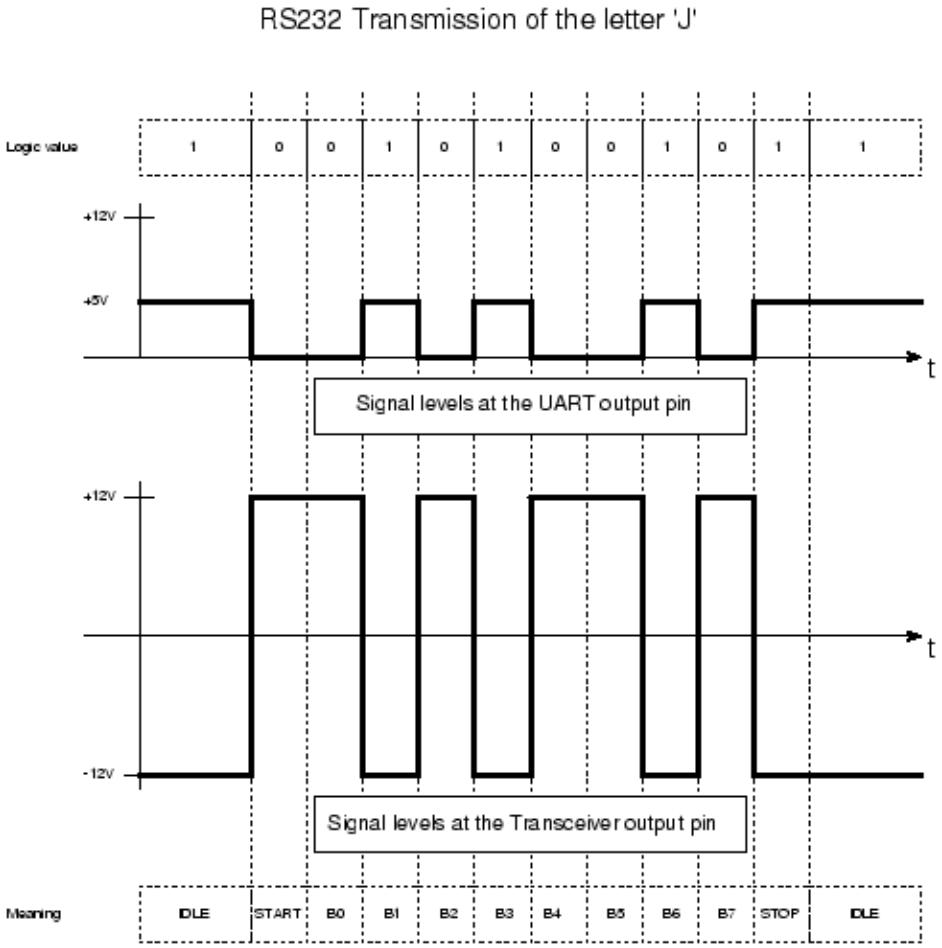
Πίνακας 2. Απόσταση μετάδοσης RS-232

1.5.4. Ηλεκτρικά χαρακτηριστικά

Τα ηλεκτρικά χαρακτηριστικά των RS-232 καθορίζουν το ελάχιστο και το μέγιστο της τάσης που αντιστοιχεί σε ένα δυαδικό ψηφίο δηλ. ποια τάση αντιπροσωπεύει το λογικό «0» και ποιο το λογικό «1». Ένα τυπικό ηλεκτρονικό κύκλωμα συνήθως λειτουργεί με τάση 5V ή 3.3V. Ωστόσο, αυτή η τάση δεν μπορεί να χρησιμοποιηθεί για μετάδοση δεδομένων μέσα από καλώδιο σε μεγάλες αποστάσεις, αφού είναι γνωστό ότι η τάση του σήματος, που αποστέλλεται μέσα από ένα καλώδιο, μειώνεται ανάλογα με την απόσταση λόγω της αντίστασης του καλωδίου.

Για το λόγο αυτό, το πρότυπο RS-232 χρησιμοποιεί τάσεις ±25V. Μια λογική «1» κυμαίνεται από -3V -25V, αλλά συνήθως είναι περίπου -12V. Μια λογική «0» κυμαίνεται από 3V έως 25V, αλλά συνήθως είναι περίπου +12V. Αν δεν υπάρχουν παλμοί στη γραμμή το επίπεδο τάσης είναι ισοδύναμο με τη λογική «1», δηλαδή -12 V. Ένα επίπεδο τάσης 0V στο δέκτη ερμηνεύεται ως αλλαγή γραμμής ή βραχυκύκλωμα.

Το παρακάτω διάγραμμα δείχνει την λειτουργία RS232 με την παραγωγή δυαδικών ψηφίων 0-5V στην ακίδα εξόδου του μικροελεγκτή που ακολουθείται από τη μεταφρασμένη τάση που μεταδίδεται με το σειριακό καλώδιο.



Εικόνα 10. Μετατροπή σημάτων TTL σε RS-232

Οι τάσεις του πρωτοκόλλου RS-232 είναι πολύ υψηλές για τη σύγχρονη λογική του υπολογιστή.Ως εκ τούτου, οι RS-232 διεπαφές τυπικά περιέχουν ειδικό υλικό, έναν οδηγό γραμμής και ένα δέκτη γραμμής.

Ο οδηγός της γραμμής είναι υπεύθυνος για τη μετατροπή της λογικής τάσης του υπολογιστή με στις υψηλές τάσεις που χρησιμοποιούνται από την RS-232 γραμμή και για την αντιστροφή της λογικής.Αυτό χρησιμοποιείται όταν το υλικό του υπολογιστή μεταδίδει σειριακά δεδομένα.Ο δέκτης γραμμής είναι υπεύθυνος για την αντίστροφη λειτουργία του οδηγού γραμμής.Μετατρέπει τα εισερχόμενα RS-232 σήματα σε τάσεις ασφαλή για τη λογική του υπολογιστή και φυσικά αντιστρέφει επίσης τη λογική.Συνήθως πολλαπλοί οδηγοί / δέκτες συνδυάζονται σε ένα ολοκληρωμένο. Το ολοκληρωμένο προστατεύει επίσης τη λογική του υπολογιστή από σφάλματα που θα μπορούσαν να συμβούν στο σειριακό καλώδιο.Μερικά ολοκληρωμένα οδηγού / δέκτη έχουν την απαραίτητη RS-232 τάση εντός του ολοκληρωμένου (on-chip), ενώ άλλα χρειάζονται εξωτερικές πηγές ενέργειας ή στοιχεία για την δημιουργία των τάσεων αυτών.

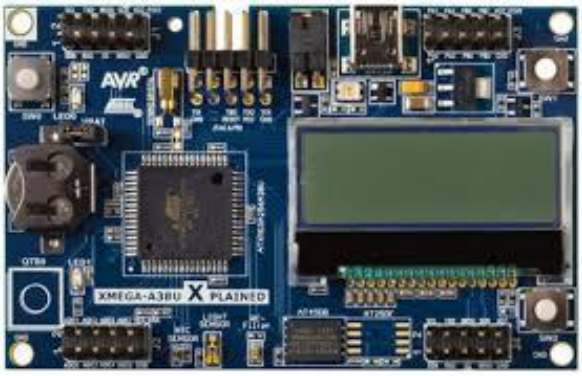
2. ΥΛΟΠΟΙΗΣΗ

Στο κεφάλαιο αυτό περιγράφονται το λογισμικό και το υλικό που αναπτύχθηκε για για προκύψει το τελικό λειτουργικό αποτέλεσμα αυτής της πτυχιακής. Συγκεκριμένα παρουσιάζεται το λογισμικό που τρέχει στον επεξεργαστή AVR καθώς και το υλικό διασύνδεσης όπου αυτό ήταν αναγκαίο, πχ στην διασύνδεση του GSM και του GPRS modem στις αντίστοιχες σειριακές θύρες του επεξεργαστή.

Για να αναπτυχθεί το λογισμικό της εφαρμογής χρησιμοποιήθηκε η πλατφόρμα ανάπτυξης XMEGA-A3BU της εταιρείας Atmel, το GSM modem και ο δέκτης GSP της εταιρείας Telit.

2.1. XMEGA-A3BU Xplained

Το kit Atmel AVRO Xplained αποτελεί μία πλατφόρμα για την έγκαιρη αξιολόγηση των δυνατοτήτων που προσφέρονται από τον μικροελεγκτή ATxMEGA256A3BU.



Εικόνα 11. Evaluation Board για τον μικροελεγκτή ATxMEGA256A3BU

Η μεγάλη συλλογή από χαρακτηριστικά που προσφέρει το kit επιτρέπουν στον χρήστη να ξεκινήσει αμέσως την χρήση των περιφεριακών του μικροελεγκτή και να κατανοήσει πώς να ενσωματώσει τον ATxmega256A3BUστο δικό του σχεδιασμό. Το kit διαθέτει τα παρακάτω βασικά χαρακτηριστικά:

- Atmel AVR ATxmega256A3BU μικροελεγκτή
- LCD οθόνη FSTN με 128x32 pixels
- Ρολόι πραγματικού χρόνου (Real Time Clock, RTC)
- Αισθητήρας φωτισμού περιβάλλοντος
- Αισθητήρας θερμοκρασίας
- Αναλογικό φίλτρο
- Ψηφιακές εισόδους/εξόδους
- Τρία μηχανικά πλήκτρα
- Δύο LEDs χρήστη
- Τέσσερις κεφαλίδες επέκτασης
- Ένα κουμπί Atmel AVR QTouch®
- Εξωτερική μνήμη DataFlash

Ο ATxmega256A3BU στο XMEGA-A3BU Xplained είναι προ-προγραμματισμένος με ένα φορτωτή εκκίνησης (boot loader). Ωστόσο, παρέχει δυνατότητες προγραμματισμού και αποσφαλμάτωσης μέσω μιας JTAG κεφαλίδας 10-pins. Για το λόγο αυτό χρησιμοποιήθηκε το εργαλείο AVR JTAGICE mkII από εξωτερικό ηλεκτρονικό υπολογιστή σε περιβάλλον προγραμματισμού Atmel Studio.

Εικόνα 12. Εργαλείο προγραμματισμού για τον μικροελεγκτή ATxMEGA256A3BU

2.2. Global System for Mobile (GSM)

Για να είναι εφικτή η αποστολή μηνυμάτων μέσω χρήσης της κινητής τηλεφωνίας χρησιμοποιήθηκε GSM modem τύπου GE863-QUAD το οποίο περιλαμβάνει GSM modem και GPS στο ίδιο ολοκληρωμένο, παρέχοντας τη δυνατότητα να χρησιμοποιηθεί σε σύνθετες εφαρμογές.

Τα βασικά χαρακτηριστικά του ολοκληρωμένου GE863-QUAD είναι τα παρακάτω:

- Quad band 850/900/1800/1900 MHz
- Ισχύς εξόδου:

  Κατηγορία 4 (2W) στο GSM 900 MHz

  Κλάση 1 (1W) στο GSM 1800/1900 MHz

- Έλεγχος στις εντολές (ITU, GSM, GPRS και Telit)
- Τροφοδοσία Τάσης: 3,4 V ÷ 4,2 V, Ονομαστικό: 3,8 V
- Κατανάλωση ισχύος: κατάσταση αναμονής: <3,5 mA λειτουργεί στα 250 mA
- Διαστάσεις (mm): 6 x 43.9 x 43.9
- Βάρος (g): 20 (συμπεριλαμβανομένης της θωράκισης)
- Εύρος θερμοκρασίας: -20 έως 70

Για τον έλεγχο του GE863-QUAD χρησιμοποιήθηκε το kit της εταιρεία telit το οποίο είναι μία πλατφόρμα για την αξιολόγηση των δυνατοτήτων των ολοκληρωμένων αυτού του τύπου. Το ολοκληρωμένο GE863-QUAD προσαρμόζεται στο α αξιολόγησης με τη βοήθεια ειδικού προσαρμογές (adapter) όπως φαίνεται στην εικόνα 13.

Το kit συνδέεται με εξωτερικό ηλεκτρονικό υπολογιστή μέσω σειριακής θύρας RS232. Ο έλεγχος του ολοκληρωμένου πραγματοποιήθηκε με χρήση AT εντολών με

τη βοήθεια του προγράμματος HyperTerminal σε λειτουργικό περιβάλλον WindowsXP.

(α) **(β)**

Εικόνα 13. (α) Evaluation board για ολοκληρωμένα τύπου GE863-QUAD

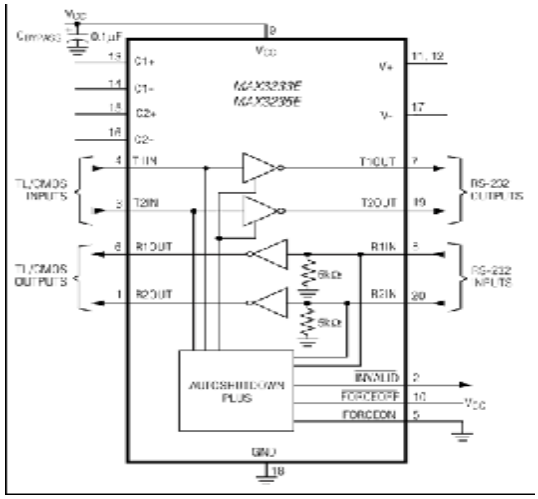
(β) GE863-QUAD adapter

2.3. Ηλεκτρικές συνδέσεις

Όπως φαίνεται στο σχήμα 1, η υλοποίηση της εφαρμογής απαιτεί τρεις διαφορετικές σειριακές συνδέσεις αμφίδρομης επικοινωνίας: GSM modem-AVR, GPS-AVR, AVR-PC. Το kit AVR Xplained διαθέτει μία USART τύπου RS232 και 4 USART τύπου TTL. Από την άλλη πλευρά το GSM modem και ο GPS δέκτης διαθέτουν USART τύπου RS232 και ο ηλεκτρονικός υπολογιστή (PC), θύρα τύπου USB ή RS232.

Το kit AVR Xplained συνδέθηκε στον φορητό υπολογιστή μέσω της USART τύπου RS232. Επειδή ο υπολογιστής που χρησιμοποιήθηκε δεν είχε ενσωματωμένη υποδοχή RS-232 χρησιμοποιήθηκε ένας μετατροπέας από USB σε RS-232.

Για να συνδεθεί το modem και το GPS στο AVR Xplained kit, θα έπρεπε οι σειριακές θύρες τύπου RS232 να μετατραπούν σε TTL και αντίστροφα. Για το λόγο αυτό χρησιμοποιήθηκε το ολοκληρωμένο MAX3235E το οποίο εξασφαλίζει την μετατροπή των σταθμών TTL του AVR στις στάθμες του προτύπου RS232 και προς τις δύο κατευθύνσεις. Το MAX3235E χρησιμοποιείται για τη μετατροπή λογικής TTL/CMOS σε RS232 χωρίς να απαιτούνται εξωτερικά στοιχεία (πυκνωτές).



Εικόνα 14. Το ολοκληρωμένο MAX3235

Στην υλοποίηση του κυκλώματος χρησιμοποιήσαμε την απλή συνδεσμολογία σειριακής διασύνδεσης με 3 ακροδέκτες. Ένα pin για την αποστολή δεδομένων (TX), ένα για την λήψη (RX) δεδομένων και το άλλο pin για την γείωση (GND). Έτσι συνδέοντας τους αντίστοιχους ακροδέκτες της κάθε συσκευής εξασφαλίζεται η επικοινωνία των συσκευών.

Ένα σημαντικό μειονέκτημα της λογικής TTL είναι ότι οι περισσότερες από τις συσκευές που λειτουργούν στην λογική TTL καταναλώνουν πολύ ρεύμα.Στο TTL Logic, ένα λογικό HIGH (ή 1) είναι +5 Volt, ενώ ένα λογικό LOW (ή 0) είναι 0 Volt. Αλλά η επίτευξη ακριβούς +5 Volt και 0 Volt δεν είναι πρακτικά δυνατόν, έτσι κάθε φορά, διάφορα IC κατασκευάζονται ώστε να καθορίζουν το εύρος στάθμης του TTL, το σύνηθες αποδεκτό εύρος για ένα HIGH είναι μέσα σε +3.5 ~ +5.0 Volt και για ένα LOW είναι 0 ~ +0.8 Volt.

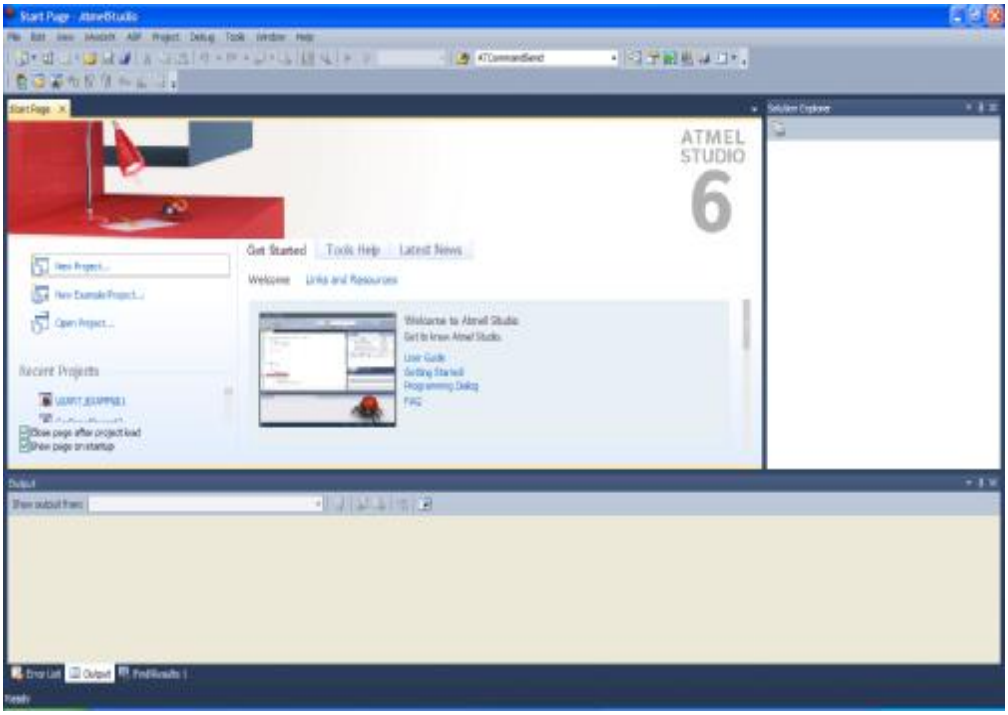
Για να διασυνδεθούν δύο οικογένειες διαφορετικού επιπέδου λογικής, οι αντίστοιχες αλλαγές τάσης πρέπει να είναι τέτοιες που να ταιριάζουν στην αντίστοιχη οικογένεια αλλιώς δε θα λειτουργήσει η διασύνδεση.Για παράδειγμα,για να διασυνδεθούν δύο συσκευές, μία από τις οποίες εργάζεται με TTL και η άλλη με RS232, θα πρέπει να μετατραπεί το HIGH του TTL (το οποία είναι 3.3v ~ 5v) στο HIGH του RS232 (που είναι -3v ~ -25v) και ομοίως, το LOW του TTL (0V ~ 0.8V) στο LOW του RS232 (που είναι + 3v ~ + 25v).

Αν δεν μετατραπούν τα επίπεδα λογικής τότε το LOW σήμα του TTL θα ερμηνευθεί ως HIGH στο RS232, κάνοντας όλη τη μεταφορά δεδομένων εσφαλμένη. Η καλύτερη λύση είναι η χρήση των ICs που μετατρέπουν άμεσα λογικά επίπεδα,όπως το MAX3233E

2.4. Προγραμματισμός

2.4.1. Atmel Studio 6.2

Το Atmel Studio 6.2 είναι μία ολοκληρωμένη πλατφόρμα ανάπτυξης (Integrated Development Platform, IDP) η οποία μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών για την οικογένεια μικροελεγκτών που βασίζονται στον Atmel AVR.Το Atmel Studio 6.2 παρέχει ένα εύκολο στη χρήση περιβάλλον για να γράψουμε, να δομήσουμε και να διορθώσουμε εφαρμογές σε κώδικα C / C ++ ή assembly.

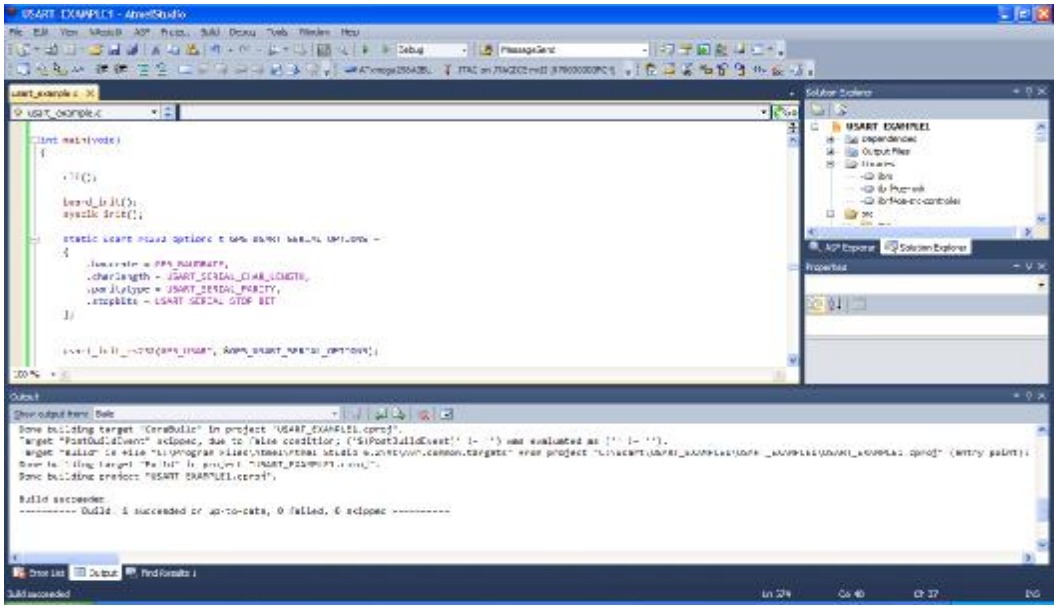


Εικόνα 15. Εργαλείο προγραμματισμού Atmel Studio

Το Atmel Studio 6.2 είναι δωρεάν και ολοκληρώνεται με το πλαίσιο λογισμικού της Atmel (ASF) το οποίο περιέχει μια μεγάλη βιβλιοθήκη ελεύθερου κώδικα με πάνω από 1.600 παραδείγματα πηγαίου κώδικα για την επιτάχυνση της ανάπτυξης νέων εφαρμογών. Έτσι ενισχύει το περιβάλλον γιατί υπάρχει πρόσβαση σε έτοιμο κώδικα προς χρήση ο οποίος ελαχιστοποιεί μεγάλο μέρος του σχεδιασμού χαμηλού επιπέδου που απαιτούνται για τις εφαρμογές.

Το Atmel Studio 6.2 είναι τώρα διαθέσιμο, προσθέτοντας προηγμένες δυνατότητες εντοπισμού σφαλμάτων και συνδέεται απόλυτα με όλα τα προγράμματα εντοπισμού σφαλμάτων της Atmel. Ένα από τα μεγαλύτερα πλεονεκτήματα των σύγχρονων μικροελεγκτών είναι η ικανότητά τους να στείλουν δεδομένα εντοπισμού σφαλμάτων σε έναν υπολογιστή, δίνοντάς σας μια τέλεια εικόνα για το τι συμβαίνει στο εσωτερικό.Αυτό δίνει τη δυνατότητα στο χρήστη να χρησιμοποιεί τις κανονικές εντολές debug όπως Run, Break, Reset, Single Step, Set Breakpoints και παρακολούθηση μεταβλητών.

Ο χρήστης σε κάθε βήμα της ανάπτυξης του κώδικα μπορεί να δει τα λάθη του, να τα διορθώσει και να παράγει το πηγαίο κώδικα, ο οποίος θα φορτωθεί στον μικροελεγκτή, μέσω της διαδικασίας «κτισίματος» (build)όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 16. Αποτέλεσμα διαδικασίας παραγωγής κώδικα

2.4.2. Το πρόγραμμα HyperTerminal

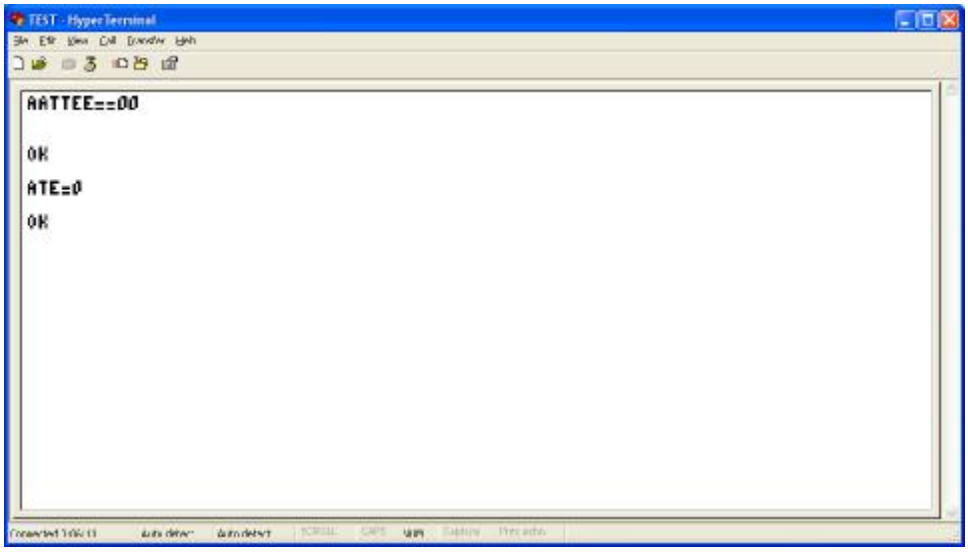
Πριν την διαδικασία ανάπτυξης της εφαρμογής και ειδικότερα πριν ξεκινήσει η διαδικασία προγραμματισμού του μικροελεγκτή κρίθηκε σκόπιμο η αποστολή προς το GSM μόντεμ των εντολών που αναφέρθηκαν, με σκοπό να εξομοιώσουμε την διαδικασία που θα αναλάμβανε να εκτελέσει στην συνέχεια ο μικροελεγκτής. Για τον λόγο αυτό θα έπρεπε να συνδέσουμε το GSM μόντεμ της εφαρμογής με ένα τερματικό και μέσω ενός διαύλου επικοινωνίας να αποστείλουμε τις εντολές AT προς αυτό. Το τερματικό που χρησιμοποιήθηκε ήταν ένας ηλεκτρονικός υπολογιστής ενώ η σύνδεση του με το GSM μόντεμ έγινε μέσω του καλωδίου σύνδεσης στην σειριακή θύρα του υπολογιστή. Για την πρόσβαση στη σειριακή θύρα κάναμε χρήση του προγράμματος *HyperTerminal* αφού πρώτα καθορίσαμε τις ρυθμίσεις της θύρας δεδομένων (Port Settings) ως εξής:

- Bits per second: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: None

Για παράδειγμα, συνδέουμε το GSM modem στην σειριακή θύρα του ηλεκτρονικού υπολογιστή. Το GSM modemκαι ο ηλεκτρονικός υπολογιστή έχουν τις ίδιες ρυθμίσεις.

Εικόνα 17. Επιλογή ρυθμίσεων σειριακής επικοινωνίας

Με το πάτημα ενός πλήκτρου από το πληκτρολόγιο του υπολογιστή αποστέλλεται μέσω της σειριακής θύρας μια αλληλουχία bitτα οποία αντιπροσωπεύουν τον ASCIIκωδικό του χαρακτήρα που πατήθηκε. Επειδή είναι ενεργοποιημένη η ηχώ από το modemo χαρακτήρας αυτός επιστρέφεται πίσω στο υπολογιστή τον οποίο λαμβάνει ο δέκτης της σειριακής θύρας και τον εκτυπώνει στην οθόνη. Η ίδια διαδικασία επαναλαμβάνεται για κάθε χαρακτήρα μέχρι το τέλος σύνταξης της εντολής και την αποστολή του χαρακτήρα [CR] (πάτημα πλήκτρου “Enter”). Αν η εντολή που στάλθηκε είναι σωστή το κινητό τηλέφωνο εκτελεί τις κατάλληλες διεργασίες και στέλνει ανάλογο μήνυμα προς τον υπολογιστή. Σε διαφορετική περίπτωση αν ανιχνευθεί λάθος τότε αποστέλλεται κατάλληλο μήνυμα.



Εικόνα 18. Παράδειγμα σειριακής επικοινωνίας

2.4.3. Ανάλυση λογισμικού

Modem Configuration

Ο μικροελεγκτής αρχικά στέλνει στο modem την εντολή απενεργοποίησης επιστροφής χαρακτήρων ATE0. Εφόσον η απάντηση του modem είναι “OK”, η εκτέλεση της εντολής θεωρείται επιτυχής και η υπορουτίνα οδηγείται στην επόμενη εντολή. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή εσφαλμένης εκτέλεσης (ψευδής, false) και η συνέχεια της λειτουργία της εγκαταλείπεται.

```
sprintf(modem_command_buffer, "ATE0");
flagResult=ATCommandResponse();
if(flagResult==false){
      return(false);
}

if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0){
      return(false);
```

Στην συνέχεια ο μικροελεγκτής στέλνει στο modem την εντολή επιλογής μηνυμάτων κειμένου AT+CMGF=1. Εφόσον η απάντηση του modem είναι “OK”, η εκτέλεση της εντολής θεωρείται επιτυχής και η υπορουτίνα οδηγείται στην επόμενη εντολή. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή εσφαλμένης εκτέλεσης (ψευδής, false) και η συνέχεια της λειτουργία της εγκαταλείπεται.

```
sprintf(modem_command_buffer, "AT+CMGF=1");
flagResult=ATCommandResponse();
if(flagResult==false) {
      return(false);
}

if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0){
      return(false);
```

Στην συνέχεια ο μικροελεγκτής στέλνει στο modem την εντολή επιλογής υπηρεσίας μηνυμάτων AT+CSMS=1. Μία απάντηση του modem που περιέχει τους χαρακτήρες +CSMS θεωρείται επιτυχής και η υπορουτίνα οδηγείται στην επόμενη εντολή. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή εσφαλμένης εκτέλεσης (ψευδής, false) και η συνέχεια της λειτουργία της εγκαταλείπεται.

```
sprintf(modem_command_buffer, "AT+CSMS=0");
flagResult=ATCommandResponse();
if(flagResult==false){
      return(false);
}

if(strncmp(modem_response_buffer, "\r\n+CSMS",7)!=0) {
      return(false);
}
```

Ακολούθως ο μικροελεγκτής στέλνει στο modem την εντολή επιλογής set χαρακτήρων AT+CSCS=“GSM”. Εφόσον η απάντηση του modem είναι “OK”, η εκτέλεση της εντολής θεωρείται επιτυχής και η υπορουτίνα οδηγείται στην επόμενη εντολή. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή εσφαλμένης εκτέλεσης (ψευδής, false) και η συνέχεια της λειτουργία της εγκαταλείπεται.

```
sprintf(modem_command_buffer, "AT+CSCS=\"GSM\"");
flagResult=ATCommandResponse();
if(flagResult==false) {
      return(false);
}

if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0){
      return(false);
}
```

Τέλος ο μικροελεγκτής στέλνει στο modem την εντολή επιλογής πρόσθετων παραμέτρων ενός μηνύματος SMS AT+CSMP=32, 167, 0, 0. Εφόσον η απάντηση του modem είναι “OK”, η εκτέλεση της εντολής θεωρείται επιτυχής και η υπορουτίνα οδηγείται στην επόμενη εντολή. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή εσφαλμένης εκτέλεσης (ψευδής, false) και η συνέχεια της λειτουργία της εγκαταλείπεται.

```
sprintf(modem_command_buffer, "AT+CSMP=32,167,0,0");
flagResult=ATCommandResponse();
if(flagResult==false){
      return(false);
}

if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0){
      return(false);
}
```

Εφόσον όλες οι AT εντολές εκτελεστούν σωστά, η υπορουτίνα επιστρέφει τιμή επιτυχούς εκτέλεσης (αληθής, true) και τερματίζει την λειτουργία της.

AT Command Send

Η υπορουτίνα αυτή χρησιμοποιείται για να στείλει ο μικροελεγκτής στο modem μία AT εντολή, μέσω της σειριακής πόρτας MODEM_USART που έχει οριστεί για αυτό το λόγο. Η AT εντολή θα πρέπει να τοποθετηθεί στην global μεταβλητή modem_command_buffer. Η υπορουτίνα δεν εξετάζει την απάντηση του modem και για αυτό το λόγο επιστρέφει τιμή επιτυχής εκτέλεσης (αληθής, true) πριν τερματιστεί η λειτουργία της.

```
boolATCommandSend(void)
{
  uint8_t       i;

  for(i=0;i<strlen(modem_command_buffer);i++){
        usart_putchar(MODEM_USART,modem_command_buffer[i]);
  }

  returntrue;
}
```

Get Response

Όπως αναφέρθηκε η υπορουτίνα AT Command Send δεν εξετάζει εάν το modem απάντησε σε μια AT εντολή ή όχι. Η υπορουτίνα Get Response χρησιμοποιείται για αυτό το λόγο. Η υπορουτίνα αυτή εξετάζει τη σημαία MODEM_CHANNEL.response_is_ready (η οποία γίνεται αληθής όταν η σειριακή θύρα επικοινωνίας MODEM_CHANNEL λάβει μία έγκυρη απάντηση από το modem) κάθε 1 δευτερόλεπτο, για 10 δευτερόλεπτα συνολικά. Εάν κατά τη διάρκεια του ελέγχου η τιμή αυτή γίνει αληθής (που σημαίνει ότι ο μικροελεγκτής έχει λάβει μία έγκυρη απάντηση από το modem και την έχει τοποθετήσει στην global μεταβλητή modem_response_buffer), η υπορουτίνα επιστρέφει την τιμή έγκυρης λήψης (αληθής, true) και τερματίζει τη λειτουργία της. Διαφορετικά, με την ολοκλήρωση του χρονικού διαστήματος των 10 δευτερολέπτων η υπορουτίνα επιστρέφει τιμή αποτυχίας κατά την λήψη (ψευδής, false) πριν τερματιστεί η λειτουργία της.

```
boolGetResponse()
{
   boolflagResponse=false;

   for(uint8_ti=0;i<10;i++){
        if(MODEM_CHANNEL.response_is_ready){
               flagResponse=true;
               break;
        }
        Delay(1000);
   }
   returnflagResponse;
```

Sim State

Για να είναι δυνατή η αποστολή μηνυμάτων sms θα πρέπει ο μικροελεγκτής να βεβαιωθεί ότι η SIM κάρτα του modem είναι έτοιμη (όχι κλειδωμένη). Η υπορουτίνα αυτή χρησιμοποιείται για να διαβάσει ο μικροελεγκτής την κατάσταση στην οποία βρίσκεται η SIM κάρτα του modem, με τη βοήθεια της AT εντολής AT+CPIN?.

```
boolflagResult=false;


sprintf(modem_command_buffer, "AT+CPIN?");
flagResult=ATCommandResponse();
if(flagResult==false){
    return(-1);
}
```

Εφόσον η AT εντολή εκτελεστεί με επιτυχία, η global μεταβλητή modem_response_buffer περιέχει την κατάσταση της κάρτας SIM του modem. Ανάλογα με την κατάσταση της κάρτας, η υπορουτίνα επιστρέφει έναν αντίστοιχο ακέραιο αριθμό και τερματίζει τη λειτουργία της. Σε περίπτωση λάθους, δηλ. εάν το modem επιστρέψει μία άγνωστη κατάσταση ή λάθος, η υπορουτίνα πριν τερματίσει την λειτουργία της στέλνει τον κωδικό «-1».

```
if(strncmp(modem_response_buffer, "\r\n+CPIN: READY\r\n",16)==0)
    return(1);
if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM PIN\r\n",18)==0)
    return(2);
if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM PUK\r\n",18)==0)
    return(3);
if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM PIN2\r\n",19)==0)
    return(4);
if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM PUK2\r\n",19)==0)
    return(5);
if(strncmp(modem_response_buffer, "\r\n+CPIN: PH-SIM PIN\r\n",21)==0)
    return(6);
if(strncmp(modem_response_buffer, "\r\n+CPIN: PH-NET PIN\r\n",21)==0)
    return(7);


return(-1);
```

Enter PIN

Η πιο συνηθισμένη κατάσταση μιας SIM κάρτας είναι «έτοιμη» ή να απαιτεί εισαγωγή PIN για να μπεί στην κατάσταση «έτοιμη». Η υπορουτίνα Enter PIN επιτρέπει την εισαγωγή (αποστολή) ενός PIN από τον μικροελεγκτή προς το modem. Η παράμετρος pin Number είναι ένας δείκτης ο οποίος δείχνει σε έναν πίνακα χαρακτήρων που περιέχει το PIN. Η παράμετρος pin Type προσδιορίζει τον τύπο του PIN που εισαχθεί έτσι ώστε να χρησιμοποιηθεί και η ανάλογη AT εντολή.

Υποστηρίζονται δύο τύποι PIN: ο τύπος PIN και ο τύπος PIN2 (ο οποίος ζητείται εφόσον εισαχθεί λάθος ο κωδικός του τύπου PIN 3 φορές συνεχόμενες).

Εφόσον ο κωδικός της κάρτας SIM γίνει δεκτός, το modem απαντά «OK», η υπορουτίνα επιστρέφει την τιμή έγκυρης εισαγωγής PIN (αληθής, true) και τερματίζει τη λειτουργία της. Διαφορετικά, η υπορουτίνα επιστρέφει τιμή αποτυχημένης εισαγωγής PIN (ψευδής, false) πριν τερματιστεί η λειτουργία της.

```
boolEnterPIN(char*pinNumber,intpinType)
{
 if(pinType==1){
       sprintf(modem_command_buffer, "%s%s", "AT+CPIN=",pinNumber);
 }
 else if(pinType==2){
       sprintf(modem_command_buffer, "%s%s", "AT+CPIN2=",pinNumber);
 }
 else{
       return(false);
 }

 ATCommandSend();

 if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)     {
       return(false);
 }

 return(true);
}
```

Message Send

Η αποστολή μηνύματος sms από το GSM modem κάτω από τον έλεγχο του μικροελεγκτή περιλαμβάνει δύο στάδια:

1ο στάδιο

Ο μικροελεγκτής στέλνει στο GSM modem την AT εντολή αποστολής μηνύματος sms (AT+SMSG) με παράμετρο τον αριθμό του παραλήπτη (AT+SMGS=<destinationaddress><CR>.Στη συνέχεια ο μικροελεγκτής αναμένει την απάντηση του GSM modem (η οποία τοποθετείται στον πίνακα modem_response_buffer. Εάν η απάντηση του modem αποτελείται από την ακολουθία χαρακτήρων “>” και “ “ (κενό) τότε η AT εντολή έχει γίνει αποδεκτή και το modem αναμένει το κείμενο που θα τοποθετήσει στο μήνυμα sms. Διαφορετικά θεωρείται ότι η AT εντολή απορρίφθηκε και η εκτέλεση της υπορουτίνας τερματίζεται επιστρέφοντας ένα κωδικό λάθους.

```
sprintf(modem_command_buffer, "%s%c%s%c", "AT+CMGS=",'"',DestinationAddr,'"');

flagResponse=ATCommandResponse();

if(flagResponse==false){
    return(-1);
}

if(memcmp(modem_response_buffer, "\r\n> ",4)!=0){
    return(-2);
}
```

2ο στάδιο

Ο μικροελεγκτής στέλνει στο modem το κείμενο του μηνύματος το οποίο τερματίζεται με τον ειδικό χαρακτήρα που προκύπτει από τον συνδυασμό πλήκτρων Control + Z (κωδικός ASCII = 26). Όταν το modem λάβει τον ειδικό χαρακτήρα, αποστέλλει το μήνυμα στον αποστολέα και επιστρέφει στον μικροελεγκτή την απάντηση: <CR><LF><+CMGS:><mr><CR><LF>. Ο κωδικός mr ονομάζεται αναφορά μηνύματος (message reference) και είναι ένας αριθμός ο οποίος αυξάνεται κυκλικά από 0-255 κάθε φορά που χρησιμοποιείται. Διαφορετικά θεωρείται ότι η αποστολή μηνύματος απορρίφθηκε και η εκτέλεση της υπορουτίνας τερματίζεται επιστρέφοντας ένα κωδικό λάθους.

```
char*CtrlZ=&buffer[0];
itoa(26,CtrlZ,10);

sprintf(modem_command_buffer, "%s%c", "sms",CtrlZ);

ATCommandSend();

boolflagResult=false;

flagResult=GetResponse();

if(flagResult==false){
      return(-4);
}

if(memcmp(modem_response_buffer, "\r\n+CMGS",7)!=0){
      return(-5);
}

memccpy(messageReference,modem_response_buffer+9,'\r',3);
mr=atoi(messageReference);

return(mr);
```

main

Περιλαμβάνει το βασικό πρόγραμμα της εφαρμογής. Ακολουθεί μία περιγραφή των λειτουργιών που εκτελεί:

Αρχικά, πριν από οποιαδήποτε άλλη διαδικασία αρχικοποίησης είναι απαραίτητο να απενεργοποιηθούν όλες οι διακοπές (interrupts) του μικροελεγκτή:

```
Disable_global_interrupt();
```

Κατόπιν αρχικοποιείται η κάρτα που φιλοξενεί τον μικροελεγκτή δηλ. το XBOARD. Επειδή η κάρτα αυτή εξαρτάται από τον κατασκευαστή, οι υπορουτίνες αρχικοποίησης (π.χ. ποιες ακίδες είναι είσοδοι ή έξοδοι, το ρολόι συστήματος κλπ) ορίζονται από τον κατασκευαστή.

```
board_init();
sysclk_init();
```

Στην συνέχεια αρχικοποιείται η σειριακή θύρα που θα χρησιμοποιηθεί για την επικοινωνία με τον δέκτη GPS και το GSM modem αντίστοιχα. Αυτό περιλαμβάνει τις παραμέτρους επικοινωνίας (ρυθμό μετάδοσης δεδομένων, μήκος χαρακτήρα, bit ισοτιμίας και bit τερματισμού), την θύρα (GPS_USART ή MODEM_USART) και τον τύπο επικοινωνίας (RS232). Τέλος ενεργοποιούνται οι σειριακοί δέκτες και οι αντίστοιχες διακοπές:

```
staticusart_rs232_options_tGPS_USART_SERIAL_OPTIONS=
{
	.baudrate=GPS_BAUDRATE,
	.charlength=USART_SERIAL_CHAR_LENGTH,
	.paritytype=USART_SERIAL_PARITY,
	.stopbits=USART_SERIAL_STOP_BIT
};
usart_init_rs232(GPS_USART,&GPS_USART_SERIAL_OPTIONS);
usart_rx_enable(GPS_USART);
USART_InterruptDriver_Initialize(&GPS_USART, &USART, USART_DREINTLVL_LO_gc);
USART_RxdInterruptLevel_Set(GPS_USART.usart, USART_RXCINTLVL_LO_gc);
```

```
staticusart_rs232_options_tMODEM_USART_SERIAL_OPTIONS=
{
	.baudrate=MODEM_BAUDRATE,
	.charlength=USART_SERIAL_CHAR_LENGTH,
	.paritytype=USART_SERIAL_PARITY,
	.stopbits=USART_SERIAL_STOP_BIT
};
usart_init_rs232(MODEM_USART,&MODEM_USART_SERIAL_OPTIONS);
usart_rx_enable(MODEM_USART);
USART_InterruptDriver_Initialize(&MODEM_USART,&USART, USART_DREINTLVL_LO_gc);
USART_RxdInterruptLevel_Set(MODEM_USART.usart, USART_RXCINTLVL_LO_gc);
```

Οι παράμετροι που ελέγχουν την επικοινωνία είτε του δέκτη GPS είτε του GSM modem έχουν ομαδοποιηθεί και έχουν τοποθετηθεί σε αντίστοιχες δομές (GPS_CHANNEL και MODEM_CHANNEL). Πριν την κανονική λειτουργία του προγράμματος είναι απαραίτητο οι δομές αυτές να τεθούν σε μία αρχική κατάσταση:

```
GPS_CHANNEL.byte_index=0;
memset(GPS_CHANNEL.rx_buffer,0,sizeof(GPS_CHANNEL.rx_buffer));
GPS_CHANNEL.response_is_ready=false;
GPS_CHANNEL.response_id=UNKNOWN;
```

```
MODEM_CHANNEL.byte_index=0;
memset(MODEM_CHANNEL.rx_buffer,0,sizeof(MODEM_CHANNEL.rx_buffer));
MODEM_CHANNEL.response_is_ready=false;
MODEM_CHANNEL.response_id=UNKNOWN;
```

Στη συνέχεια ο μικροελεγκτής στέλνει ένα δοκιμαστικό μήνυμα στον ηλεκτρονικό υπολογιστή που έχει συνδεθεί στη σειριακή θύρα επικοινωνίας USART_SERIAL_EXAMPLE. Η αρχικοποίηση της θύρας γίνεται από την υπορουτίνα αρχικοποίησης της κάρτας (board_init()).

```
for(i=0;i<tx_length;i++){
      usart_putchar(USART_SERIAL_EXAMPLE,tx_buf[i]);
      _XMEGA_GPIO_H_
}
```

Αφού ολοκληρωθεί η διαδικασία αρχικοποίησης ενεργοποιούνται όλες οι διακοπές (interrupts) του μικροελεγκτή:

```
      Enable_global_interrupt();
```

Το κύριο μέρος της εφαρμογής εξομοιώνει μία κατάσταση συναγερμού. Ως συναγερμός χρησιμοποιήθηκε ένα από τα πλήκτρα που διαθέτει η κάρτα XMEGA-A3BU Xplained. Επιλέχθηκε το πλήκτρο GPIO_PUSH_BUTTON_0. Το κύριο μέρος της εφαρμογής εξετάζει την κατάσταση του πλήκτρου με τη βοήθεια της εντολής gpio_pin_is_lowκαι όταν αυτό πατηθεί (δηλ. η κατάσταση του pin αλλάξει από high σε low), αποστέλλεται ένα μήνυμα “Alarm activated!!!” στον αριθμό τηλεφώνου 6907084535.

```
while(true)
{
     if(gpio_pin_is_low(GPIO_PUSH_BUTTON_0)){
          MessageSend("6907084535", "Alarm activated!!!");
     }
}
```

ISR(USARTC0_RXC_VECT)

Αποτελεί την υπορουτίνα εξυπηρέτησης διακοπής λήψης χαρακτήρα από την σειριακή θύρα επικοινωνίας στην οποία έχει συνδεθεί ο δέκτης GPS. Δηλ. η υπορουτίνα αυτή εκτελείται κάθε φορά που ο μικροελεγκτής λαμβάνει ένα χαρακτήρα από το δέκτη GPS, εφόσον η αντίστοιχη διακοπή έχει ενεργοποιηθεί κατά τη διάρκεια της αρχικοποίησης.

Αρχικά, η υπορουτίνα διαβάζει έγκυρους ASCII χαρακτήρες από σειριακή θύρα και τους αποθηκεύει στην περιοχή μνήμης που έχει δεσμευτεί από τον πίνακα rx_buffer της δομής GPS_CHANNEL. Ο δείκτης byte_index της ίδιας δομής ελέγχει σε ποιο σημείο του πίνακα θα γραφτεί ο χαρακτήρας που ελήφθει. Αυξάνει κατά 1 μετά από κάθε εγγραφή στον πίνακα προκειμένου ο επόμενος χαρακτήρας να αποθηκευτεί στην επόμενη ελεύθερη θέση του πίνακα. Η διαδικασία επαναλαμβάνεται μέχρι να ληφθεί ο χαρακτήρας <CR>, ο οποίος υποδηλώνει το τέλος μιας πρότασης NMEA. Με αυτόν τον τρόπο τοποθετείται μέσα στον πίνακα rx_buffer μία πλήρης πρόταση NMEA.

Στην διαδικασία αυτή για να μην υπάρξει υπερχείλιση στη μνήμη που δεσμεύει ο πίνακας rx_buffer (δηλ. να γραφτούν περισσότεροι χαρακτήρες από το μέγεθος του πίνακα, γεγονός που θα προκαλούσε απρόβλεπτες συνέπειες στην εκτέλεση του προγράμματος) πραγματοποιούνται δύο έλεγχοι:

Κάθε φορά που λαμβάνεται ο χαρακτήρας «$», ο δείκτης byte_index μηδενίζεται έτσι ώστε η πρόταση NMEA να αποθηκευτεί ξεκινώντας από την αρχή του πίνακα. Έτσι είναι σίγουρο ότι η πρόταση NMEA μπορεί να αποθηκευτεί πλήρως στον πίνακα rx_buffer.

Επιπλέον, γίνεται έλεγχος της τιμής του δείκτη byte_index έτσι ώστε να μην ξεπεράσει το μέγεθος του πίνακα rx_buffer. Αν για οποιοδήποτε λόγο συμβεί αυτό, τότε ο δείκτης μηδενίζεται.

```
        intUsartData;
        UsartData=usart_getchar(GPS_USART);

        if(UsartData==0) {
            return;
        }

        if(UsartData!='\n'){
            if(GPS_CHANNEL.byte_index>sizeof(GPS_CHANNEL.rx_buffer)) {
                GPS_CHANNEL.byte_index=0;
            }

            if(UsartData=='$'){
                GPS_CHANNEL.byte_index=0;
            }

            GPS_CHANNEL.rx_buffer[GPS_CHANNEL.byte_index++]=UsartData;
            return;
        }
```

Ακολουθεί μία εντολή η οποία δεν έχει προφανή σκοπό αλλά είναι πολύ σημαντική. Προκειμένου να διαχειριστούμε τον πίνακα rx_buffer ως string θα πρέπει αυτός να τερματίζεται με το «0». Για παράδειγμα εάν θέλουμε να βρούμε τον αριθμό των χαρακτήρων που περιέχει ο πίνακας χρησιμοποιώντας την συνάρτηση strlen. Η strlen θα μετρήσει τους χαρακτήρες του πίνακα από την αρχή μέχρι να βρει τη πρώτη μηδενική τιμή. Όπως γίνεται αντιληπτό εάν το μηδέν δεν τοποθετηθεί σωστά (δηλ. στο τέλος των χαρακτήρων) τότε το μήκος που θα επιστρέψει η συνάρτηση θα είναι τυχαίο (ανάλογα σε ποια θέση μνήμης θα βρει ένα μηδενικό).

```
        GPS_CHANNEL.rx_buffer[GPS_CHANNEL.byte_index]=0;
```

Στην συνέχεια εκτελείται η ανάλυση της πρότασης. Αρχικά ελέγχεται εάν η πρόταση είναι η “GPRMC”. Εφόσον αυτό συμβαίνει τότε η μεταβλητή response_id της δομής GPS_CHANNEL λαμβάνει την τιμή GPRMC_SENTENCE για να δηλώσει την έγκυρη λήψη μιας GPRMC πρότασης. Στην συνέχεια εξετάζεται ο rx_buffer (που περιέχει την πρόταση) και αποθηκεύονται στον πίνακα comma_pos οι θέσεις των κομμάτων μέσα στην πρόταση. Η μεταβλητή comma_cnt περιέχει το συνολικό αριθμό των κομμάτων. Αυτό γίνεται γιατί όλες οι πληροφορίες που περιλαμβάνει η GPRMC βρίσκονται μεταξύ κομμάτων και γνωρίζοντας σε ποια θέση είναι τα κόμματα μπορούμε να αντλήσουμε την αντίστοιχη πληροφορία.

```
         inti=0,j=0,k=0,comma_cnt=0;


         GPS_CHANNEL.byte_index=0;
         GPS_CHANNEL.response_id=UNKNOWN;


         if(GPS_CHANNEL.rx_buffer[0]=='$'&&GPS_CHANNEL.rx_buffer[1]=='G'&&GPS_CHANNEL.
rx_buffer[2]=='P'&&GPS_CHANNEL.rx_buffer[3]=='R'&&GPS_CHANNEL.rx_buffer[4]=='M'&&GPS_C
HANNEL.rx_buffer[5]=='C')
         {
              GPS_CHANNEL.response_id=GPRMC_SENTENCE;


              for(i=0;i<strlen(GPS_CHANNEL.rx_buffer);i++)
              {
                   if(GPS_CHANNEL.rx_buffer[i]==',')
                   {
                        comma_pos[comma_cnt]=i;
                        comma_cnt++;
                   }
              }
         }
```

Αυτό ακριβώς εκτελεί το επόμενο κομμάτι κώδικα αντλώντας πληροφορίες που αφορούν το χρόνο, την κατάσταση της πρότασης (έγκυρη ή όχι), το γεωγραφικό μήκος και το γεωγραφικό πλάτος. Οι πληροφορίες αυτές αποθηκεύονται σε αντίστοιχες μεταβλητές της δομής GPRMC.

```
        if(GPS_CHANNEL.response_id==GPRMC_SENTENCE)
        {
            k=0;
            for(j=comma_pos[0]+1;j<comma_pos[1];j++){
                GPRMC.time[k++]=GPS_CHANNEL.rx_buffer[j];
            }
            GPRMC.time[k++]=0;

            k=0;
            for(j=comma_pos[1]+1;j<comma_pos[2];j++){
                GPRMC.status[k++]=GPS_CHANNEL.rx_buffer[j];
            }
            GPRMC.status[k++]=0;

            k=0;
            for(j=comma_pos[2]+1;j<comma_pos[4];j++)      {
                GPRMC.latitude[k++]=GPS_CHANNEL.rx_buffer[j];
            }
            GPRMC.latitude[k++]=0;

            k=0;
            for(j=comma_pos[4]+1;j<comma_pos[6];j++)      {
                GPRMC.longitude[k++]=GPS_CHANNEL.rx_buffer[j];
            }
            GPRMC.longitude[k++]=0;

            GPS_CHANNEL.response_is_ready=true;
        }
```

ISR(USARTD0_RXC_VECT)

Αποτελεί την υπορουτίνα εξυπηρέτησης διακοπής λήψης χαρακτήρα από την σειριακή θύρα επικοινωνίας στην οποία έχει συνδεθεί το GSM modem. Δηλ. η υπορουτίνα αυτή εκτελείται κάθε φορά που ο μικροελεγκτής λαμβάνει ένα χαρακτήρα από το GSM modem, εφόσον η αντίστοιχη διακοπή έχει ενεργοποιηθεί κατά τη διάρκεια της αρχικοποίησης.

Αρχικά, η υπορουτίνα διαβάζει έγκυρους ASCII χαρακτήρες από σειριακή θύρα και τους αποθηκεύει στην περιοχή μνήμης που έχει δεσμευτεί από τον πίνακα rx_buffer της δομής MODEM_CHANNEL. Ο δείκτης byte_index της ίδιας δομής ελέγχει σε ποιο σημείο του πίνακα θα γραφτεί ο χαρακτήρας που ελήφθηκε. Αυξάνει κατά 1 μετά από κάθε εγγραφή στον πίνακα προκειμένου ο επόμενος χαρακτήρας να αποθηκευτεί στην επόμενη ελεύθερη θέση του πίνακα. Η διαδικασία επαναλαμβάνεται μέχρι να ληφθεί ο χαρακτήρας <LF>, ο οποίος υποδηλώνει το τέλος μιας απάντησης του modem. Με αυτόν τον τρόπο τοποθετείται μέσα στον πίνακα rx_buffer μία πλήρης απάντηση.

Στην διαδικασία αυτή για να μην υπάρξει υπερχείλιση στη μνήμη που δεσμεύει ο πίνακας rx_buffer (δηλ. να γραφτούν περισσότεροι χαρακτήρες από το μέγεθος του πίνακα, γεγονός που θα προκαλούσε απρόβλεπτες συνέπειες στην εκτέλεση του προγράμματος) γίνεται έλεγχος της τιμής του δείκτη byte_index έτσι ώστε να μην ξεπεράσει το μέγεθος του πίνακα rx_buffer. Αν για οποιοδήποτε λόγο συμβεί αυτό, τότε ο δείκτης μηδενίζεται.

```
intUsartData;
UsartData=usart_getchar(MODEM_USART);


if(UsartData==0) {
      return;
}


if(UsartData!='\n'||MODEM_CHANNEL.byte_index==1){
      if(MODEM_CHANNEL.byte_index>sizeof(MODEM_CHANNEL.rx_buffer)) {
            MODEM_CHANNEL.byte_index=0;
      }
}


MODEM_CHANNEL.rx_buffer[MODEM_CHANNEL.byte_index++]=UsartData;
```

Στην συνέχεια ο μικροελεγκτής ελέγχει το είδος της απάντησης που έλαβε ενημερώνοντας ταυτόχρονα αντίστοιχα την μεταβλητή response_id. Το πρόγραμμα υποστηρίζει την λήψη 3 διαφορετικών απαντήσεων:

- Απάντηση σε εκκίνηση αποστολής μηνύματος sms«> ». Το περιεχόμενο του πίνακα RxD_buffer της δομής MODEM_CHANNEL τερματίζεται με το μηδέν, έτσι ώστε να μπορεί να είναι διαχειρίσιμο ως string και στην συνέχεια το περιεχόμενό του μεταφέρεται στον global πίνακα modem_response_buffer. Ο δείκτης byte_index, που ελέγχει σε ποιο σημείο του πίνακα θα γραφτεί ο επόμενος χαρακτήρας που ληφθεί, μηδενίζεται έτσι ώστε η σειριακή θύρα να είναι σε θέση να λάβει την επόμενη απάντηση.

```
      if(MODEM_CHANNEL.byte_index==4&&MODEM_CHANNEL.rx_buffer[2]!='>'&&MODEM_
    CHANNEL.rx_buffer[3]!=' '){
    return;
}else{
    MODEM_CHANNEL.response_id=GREATER_THAN;
    return;
}
MODEM_CHANNEL.rx_buffer[MODEM_CHANNEL.byte_index++]=0;
memset(modem_response_buffer,0,sizeof(modem_response_buffer));

            memcpy(modem_response_buffer,MODEM_CHANNEL.rx_buffer,strlen(MODE
          M_CHANNEL.rx_buffer));
MODEM_CHANNEL.byte_index=0;


if(MODEM_CHANNEL.response_id==GREATER_THAN) {
```

```
            return;
        }
```

- Απρόκλητη απάντηση δηλ. μία απάντηση που στέλνει το modem χωρίς προηγουμένως να δεχτεί κάποια AT εντολή (+CDS, +CNMI).

```
        if((memcmp(MODEM_CHANNEL.rx_buffer,
"\r\n+CMTI",7)==0)||(memcmp(MODEM_CHANNEL.rx_buffer, "\r\n+CDS",6)==0))

        {
            MODEM_CHANNEL.response_id=UNSOLICITED_CODE;
        }
```

- Απάντηση λάθους (+ERROR).

```
        if((memcmp(MODEM_CHANNEL.rx_buffer,
"\r\nERROR",7)==0)||(memcmp(MODEM_CHANNEL.rx_buffer, "\r\n+CME ERROR",12)==0))
        {
            MODEM_CHANNEL.response_id=ERROR_CODE;
        }
```

Παράμετροι

Τιμές καταχωρούνται σε ονόματα με σκοπό η χρήση τους να είναι άμεσα κατανοητή.

```
#defineUNKNOWN                  0
#defineGPRMC_SENTENCE                   1
#defineGREATER_THAN                     2
#defineUNSOLICITED_CODE                 3
#defineERROR_CODE                       4
#definePIN_CODE                         5
```

Ορίζεται η σειριακή θύρα επικοινωνίας του μικροελεγκτή και του δέκτη GPS καθώς και ο ρυθμός μετάδοσης δεδομένων.

```
#defineGPS_USART                &USARTC0
#defineGPS_BAUDRATE             4800
```

Ορίζεται η σειριακή θύρα επικοινωνίας του μικροελεγκτή και του GSM modem καθώς και ο ρυθμός μετάδοσης δεδομένων.

```
#defineMODEM_USART              &USARTD0
#defineMODEM_BAUDRATE           4800
```

Ορίζεται μία δομή που περιλαμβάνει μεταβλητές που αναφέρονται στη σειριακή θύρα επικοινωνίας του μικροελεγκτή. Αυτές είναι ο πίνακας rx_bufferόπου αποθηκεύονται οι χαρακτήρες που λαμβάνονται από τη σειριακή θύρα, ο δείκτης byte_index ο οποίος δηλώνει την θέση στην οποία θα αποθηκευτεί ο χαρακτήρας που θα ληφθεί στη σειριακή θύρα, η μεταβλητή response_id όπου αποθηκεύεται το είδος

ενός πακέτου που έχει ληφθεί και τέλος η σημαία response_is_ready η οποία γίνεται αληθής κάθε φορά που η σειριακή θύρα έχει λάβει ένα ολοκληρωμένο πακέτο.

```
typedefstructUsartRxPacket{
        uint8_tbyte_index;
        boolresponse_is_ready;
        uint8_tresponse_id;
        uint8_trx_buffer[100];
}
usart_rx_packet_t;
```

Επειδή χρησιμοποιούνται δύο σειριακά, δημιουργούμε δύο αντίγραφα για τη δομή usart_rx_packet_t: ένα για τη σειριακή θύρα που συνδέεται ο δέκτης GPS (GPS_CHANNEL) και το δεύτερο για τη σειριακή θύρα που συνδέεται το GSM modem (MODEM_CHANNEL).

```
usart_rx_packet_tGPS_CHANNEL;
usart_rx_packet_tMODEM_CHANNEL;
```

Η δομή GPRMC_sentence_t ομαδοποιεί τα πιο σημαντικά στοιχεία που περιλαμβάνει η πρόταση GPRMC και περιλαμβάνει το χρόνο (time), την ημερομηνία (date), το γεωγραφικό μήκος (latitude), το γεωγραφικό πλάτος (longitude) και την εγκυρότητα της πρότασης (status). Η δομή GPRMC χρησιμοποιεί ένα αντίγραφο της δομής GPRMC_sentence_t.

```
typedefstructGPRMCxSentence{
        chartime[10];
        chardate[10];
        charlatitude[15];
        charlongitude[15];
        charstatus[2];
}
GPRMC_sentence_t;
GPRMC_sentence_tGPRMC;
```

Οι πίνακες modem_command_buffer και modem_response_buffer είναι global περιοχές μνήμης όπου αποθηκεύονται αντίστοιχα οι AT εντολές που στέλνονται από τον μικροελεγκτή προς το GSM modem και οι απαντήσεις που λαμβάνονται από αυτό.

```
charmodem_command_buffer[256];
charmodem_response_buffer[30];
```

Οι πίνακες command_pos είναι ένας βοηθητικός πίνακας που χρησιμοποιείται στην εξαγωγή των πληροφοριών που περιέχει μία πρόταση GPRMC.

```
        charcomma_pos[40];
```

Ο πίνακας tx_buff περιέχει ένα μήνυμα καλωσορίσματος, το οποίο στέλνει ο μικροελεγκτής στη σειριακή θύρα του ηλεκτρονικού υπολογιστή κάθε φορά που εκκινείται.

```
        uint8_ttx_buf[]= "\n Hello AVR world ! : ";
```

3. ΠΑΡΑΡΤΗΜΑ

3.1.Κώδικας Εφαρμογής

```
#include<conf_usart_example.h>
#include<asf.h>
#include<gpio.h>
#include<stdio.h>
#include<string.h>
#include<xmega_a3bu_xplained.h>
#include<xmega_gpio/xmega_gpio.h>

#defineUNKNOWN                      0
#defineGPRMC_SENTENCE                       1
#defineGREATER_THAN                 2
#defineUNSOLICITED_CODE             3
#defineERROR_CODE                           4
#definePIN_CODE                             5

#defineGPS_USART                    &USARTC0
#defineGPS_BAUDRATE         4800

#defineMODEM_USART                  &USARTD0
#defineMODEM_BAUDRATE               4800

charcomma_pos[40];
//char n[] ="213311 :\r\n";
uint8_ttx_buf[]= "\n Hello AVR world ! : ";
uint8_ttx_length=22;
uint8_treceived_byte;
uint8_ti;

void my_printf(char*message);
void gsm_printf(char*gsms);
void Delay(int);


typedefstructUsartRxPacket
{
       uint8_tbyte_index;
       boolresponse_is_ready;
       uint8_tresponse_id;
       charrx_buffer[100];
}
usart_rx_packet_t;

usart_rx_packet_tGPS_CHANNEL;
usart_rx_packet_tMODEM_CHANNEL;

typedefstructGPRMCxSentence
{
       chartime[10];
       chardate[10];
       charlatitude[15];
       charlongitude[15];
       charstatus[2];
}
GPRMC_sentence_t;

GPRMC_sentence_tGPRMC;
```

```
charmodem_command_buffer[256];
charmodem_response_buffer[30];

intMessageSend(char*,char*);


boolATCommandResponse(void);
boolATCommandSend(void);
boolGetResponse(void);
boolModemConfiguration(void);


boolEnterPIN(char*,int);
intSimState(void);

//USART_HOST_ISR_VECT()
ISR(USARTC0_RXC_vect)
{
	intUsartData;

	UsartData=usart_getchar(GPS_USART);

	if(UsartData==0)
		return;

	if(UsartData!='\n')
	{

		if(GPS_CHANNEL.byte_index>sizeof(GPS_CHANNEL.rx_buffer))
			GPS_CHANNEL.byte_index=0;

		if(UsartData=='$')
		{
			GPS_CHANNEL.byte_index=0;
		}

		GPS_CHANNEL.rx_buffer[GPS_CHANNEL.byte_index++]=UsartData;
		return;
	}


	GPS_CHANNEL.rx_buffer[GPS_CHANNEL.byte_index]=0;


	inti=0,j=0,k=0,comma_cnt=0;

	GPS_CHANNEL.byte_index=0;
	GPS_CHANNEL.response_id=UNKNOWN;


	if(GPS_CHANNEL.rx_buffer[0]=='$'&&GPS_CHANNEL.rx_buffer[1]=='G'&&GPS_CHA
NNEL.rx_buffer[2]=='P'&&GPS_CHANNEL.rx_buffer[3]=='R'&&GPS_CHANNEL.rx_buffer[4]
=='M'&&GPS_CHANNEL.rx_buffer[5]=='C')
	{
		GPS_CHANNEL.response_id=GPRMC_SENTENCE;

		for(i=0;i<strlen(GPS_CHANNEL.rx_buffer);i++)
		{
			if(GPS_CHANNEL.rx_buffer[i]==',')
			{
				comma_pos[comma_cnt]=i;
```

```
                        comma_cnt++;
                }
        }
}

if(GPS_CHANNEL.response_id==GPRMC_SENTENCE)
{

        k=0;
        for(j=comma_pos[0]+1;j<comma_pos[1];j++)
        {
                GPRMC.time[k++]=GPS_CHANNEL.rx_buffer[j];
        }
        GPRMC.time[k++]=0;

        k=0;
        for(j=comma_pos[1]+1;j<comma_pos[2];j++)
        {
                GPRMC.status[k++]=GPS_CHANNEL.rx_buffer[j];
        }
        GPRMC.status[k++]=0;

        k=0;
        for(j=comma_pos[2]+1;j<comma_pos[4];j++)
        {
                GPRMC.latitude[k++]=GPS_CHANNEL.rx_buffer[j];
        }
        GPRMC.latitude[k++]=0;

        k=0;
        for(j=comma_pos[4]+1;j<comma_pos[6];j++)
        {
                GPRMC.longitude[k++]=GPS_CHANNEL.rx_buffer[j];
        }
        GPRMC.longitude[k++]=0;

        GPS_CHANNEL.response_is_ready=true;
}
}


ISR(USARTD0_RXC_vect)
{
        intUsartData;

        UsartData=usart_getchar(GPS_USART);

        if(UsartData==0)
                return;

        if(UsartData!='\n'||MODEM_CHANNEL.byte_index==1)
        {

                if(MODEM_CHANNEL.byte_index>sizeof(MODEM_CHANNEL.rx_buffer))
                        MODEM_CHANNEL.byte_index=0;
        }

        MODEM_CHANNEL.rx_buffer[MODEM_CHANNEL.byte_index++]=UsartData;

        if(MODEM_CHANNEL.byte_index==4&&MODEM_CHANNEL.rx_buffer[2]!='>'&&MODEM_C
HANNEL.rx_buffer[3]!=' ')
        {
```

```
                return;
        }
        else
        {

                MODEM_CHANNEL.response_id=GREATER_THAN;
                return;
        }


        MODEM_CHANNEL.rx_buffer[MODEM_CHANNEL.byte_index++]=0;

        memset(modem_response_buffer,0,sizeof(modem_response_buffer));

        memcpy(modem_response_buffer,MODEM_CHANNEL.rx_buffer,strlen(MODEM_CHANNE
L.rx_buffer));

        MODEM_CHANNEL.byte_index=0;


        if(MODEM_CHANNEL.response_id==GREATER_THAN)
                return;

        if((memcmp(MODEM_CHANNEL.rx_buffer,
"\r\n+CMTI",7)==0)||(memcmp(MODEM_CHANNEL.rx_buffer, "\r\n+CDS",6)==0))
        {

                MODEM_CHANNEL.response_id=UNSOLICITED_CODE;
        }

        elseif((memcmp(MODEM_CHANNEL.rx_buffer,
"\r\nERROR",7)==0)||(memcmp(MODEM_CHANNEL.rx_buffer, "\r\n+CME ERROR",12)==0))
        {

                MODEM_CHANNEL.response_id=ERROR_CODE;
        }

        elseif((memcmp(MODEM_CHANNEL.rx_buffer, "\r\n+CPIN",7)==0))
        {

                MODEM_CHANNEL.response_id=PIN_CODE;
        }
        else
        {

                MODEM_CHANNEL.response_id=UNKNOWN;
        }

        MODEM_CHANNEL.response_is_ready=true;

}


intmain(void)
{

        Disable_global_interrupt();

        board_init();
        sysclk_init();
```

```
USART_InterruptDriver_Initialize(&GPS_USART, &USART,
       USART_DREINTLVL_LO_gc);
USART_RxdInterruptLevel_Set(GPS_USART.usart, USART_RXCINTLVL_LO_gc);



staticusart_rs232_options_tGPS_USART_SERIAL_OPTIONS=
{
       .baudrate=GPS_BAUDRATE,
       .charlength=USART_SERIAL_CHAR_LENGTH,
       .paritytype=USART_SERIAL_PARITY,
       .stopbits=USART_SERIAL_STOP_BIT
};


usart_init_rs232(GPS_USART,&GPS_USART_SERIAL_OPTIONS);
usart_rx_enable(GPS_USART);

USART_InterruptDriver_Initialize(&MODEM_USART, &USART,
       USART_DREINTLVL_LO_gc);
USART_RxdInterruptLevel_Set(MODEM_USART.usart, USART_RXCINTLVL_LO_gc);


staticusart_rs232_options_tMODEM_USART_SERIAL_OPTIONS=
{
       .baudrate=MODEM_BAUDRATE,
       .charlength=USART_SERIAL_CHAR_LENGTH,
       .paritytype=USART_SERIAL_PARITY,
       .stopbits=USART_SERIAL_STOP_BIT
};

usart_init_rs232(MODEM_USART,&MODEM_USART_SERIAL_OPTIONS);
usart_rx_enable(MODEM_USART);

GPS_CHANNEL.byte_index=0;
memset(GPS_CHANNEL.rx_buffer,0,sizeof(GPS_CHANNEL.rx_buffer));

GPS_CHANNEL.response_is_ready=false;
GPS_CHANNEL.response_id=UNKNOWN;

MODEM_CHANNEL.byte_index=0;
memset(MODEM_CHANNEL.rx_buffer,0,sizeof(MODEM_CHANNEL.rx_buffer));

MODEM_CHANNEL.response_is_ready=false;
MODEM_CHANNEL.response_id=UNKNOWN;


for(i=0;i<tx_length;i++)
{
       usart_putchar(USART_SERIAL_EXAMPLE,tx_buf[i]);
       _XMEGA_GPIO_H_

}

/* Enable PMIC interrupt level low. */
PMIC.CTRL |= PMIC_LOLVLEX_bm;

Enable_global_interrupt();


while(true)
{
```

```
                if(gpio_pin_is_low(GPIO_PUSH_BUTTON_0)){
                        MessageSend("6907084535", "Alarm activated!!!");
                }

        }
}


boolModemConfiguration()
{
        boolflagResult=false;

        sprintf(modem_command_buffer, "ATE0");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
        {
                return(false);
        }

        sprintf(modem_command_buffer, "AT+CMGF=1");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
        {
                return(false);
        }

        sprintf(modem_command_buffer, "AT+CSMS=0");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\n+CSMS",7)!=0)
        {
                return(false);
        }

        sprintf(modem_command_buffer, "AT+CSCS=\"GSM\"");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
        {
                return(false);
        }

        sprintf(modem_command_buffer, "AT+CNMI=2,1,0,1,0");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
        {
                return(false);
        }
```

```
        sprintf(modem_command_buffer, "AT+CSMP=32,167,0,0");
        flagResult=ATCommandResponse();
        if(flagResult==false)
                return(false);

        if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
        {
                return(false);
        }

        return(true);
}

boolATCommandResponse()
{
        boolflagResponse=false;

        ATCommandSend();

        flagResponse=GetResponse();

        return(flagResponse);
}



boolATCommandSend(void)
{
        uint8_t       i;

        for(i=0;i<strlen(modem_command_buffer);i++)
        {
                usart_putchar(MODEM_USART,modem_command_buffer[i]);
        }

        returntrue;
}


intMessageSend(char*DestinationAddr,char*MessageToSend)
{
        char    buffer[5];
        bool    flagResponse;
        char    messageReference[3];
        int             mr;

        char*CtrlZ=&buffer[0];

        itoa(26,CtrlZ,10);

        sprintf(modem_command_buffer, "%s%c%s%c",
"AT+CMGS=",'"',DestinationAddr,'"');

        flagResponse=ATCommandResponse();


        if(flagResponse==false)
        {
                return(-1);
        }

        if(memcmp(modem_response_buffer, "\r\n> ",4)!=0)
```

```
        {
                return(-2);
        }

        sprintf(modem_command_buffer, "%s%s", "sms",CtrlZ);

        ATCommandSend();

        boolflagResult=false;

        flagResult=GetResponse();

        if(flagResult==false)
        {
                return(-4);
        }

        if(memcmp(modem_response_buffer, "\r\n+CMGS",7)!=0)
        {
                return(-5);
        }

        memccpy(messageReference,modem_response_buffer+9,'\r',3);
        mr=atoi(messageReference);

        return(mr);

}


intSimState()
{
        boolflagResult=false;

        sprintf(modem_command_buffer, "AT+CPIN?");
        flagResult=ATCommandResponse();
        if(flagResult==false)
        {
                return(-1);
        }

        if(strncmp(modem_response_buffer, "\r\n+CPIN:
READY\r\n",16)==0)return(1);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM
PIN\r\n",18)==0)return(2);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM
PUK\r\n",18)==0)return(3);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM
PIN2\r\n",19)==0)return(4);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: SIM
PUK2\r\n",19)==0)return(5);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: PH-SIM
PIN\r\n",21)==0)return(6);
        if(strncmp(modem_response_buffer, "\r\n+CPIN: PH-NET
PIN\r\n",21)==0)return(7);

        return(-1);

}


boolEnterPIN(char*pinNumber,intpinType)
```

```
{
	if(pinType==1)
	{
		sprintf(modem_command_buffer, "%s%s", "AT+CPIN=",pinNumber);
	}
	elseif(pinType==2)
	{
		sprintf(modem_command_buffer, "%s%s", "AT+CPIN2=",pinNumber);
	}
	else
	{
		return(false);
	}

	ATCommandSend();

	if(strncmp(modem_response_buffer, "\r\nOK\r\n",6)!=0)
	{
		return(false);
	}

	return(true);
}


boolGetResponse()
{
	boolflagResponse=false;


	for(uint8_ti=0;i<10;i++){
		if(MODEM_CHANNEL.response_is_ready){
			flagResponse=true;
			break;
		}

		delay_ms(1000);
	}

	returnflagResponse;
}
```