

**Υλοποίηση δικτύου ασύρματης ραδιοεπικοινωνίας
μεταξύ δύο ενσωματωμένων κόμβων (κόμβος
ταυτοποίησης και κόμβος ελέγχου) για τον
απομακρυσμένο έλεγχο συσκευών μέσω ψηφιακής
ταυτοποίησης και αναγνώρισης φυσικών
αντικειμένων με χρήση της πλατφόρμας Arduino**

Πτυχιακή Εργασία

Ευστάθιος Χατζηκυριακίδης

Επιβλέπων : Ιωάννης Μαδεμλής, Εργαστηριακός Συνεργάτης

ΣΕΡΡΕΣ, ΜΑΪΟΣ 2011

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε υποστήριξη ή βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται μέσα σε αυτήν. Επίσης, σε αυτή την πτυχιακή εργασία αναφέρονται όλες οι πηγές από τις οποίες έκανα χρήση δεδομένων και πληροφοριών. Τέλος, βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά και ειδικά για τις αναγκαίες απαιτήσεις του προγράμματος σπουδών του Τμήματος Πληροφορικής & Επικοινωνιών του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Σερρών.

Ευστάθιος Χατζηκυριακίδης

Περίληψη

Σε αυτή την πτυχιακή εργασία αναπτύχθηκε ένα ιδιωτικό σύστημα ασύρματου κέντρου ελέγχου μικρής εμβέλειας για τον απομακρυσμένο έλεγχο συσκευών μέσω ψηφιακής ραδιοσυχνικής ταυτοποίησης (RFID). Επίσης, το σύστημα αυτό, αποτελείται από δύο ενσωματωμένους κόμβους (τον κόμβο ταυτοποίησης και τον κόμβο ελέγχου) οι οποίοι διασυνδέονται άμεσα και επικοινωνούν χρησιμοποιώντας ειδικές κάρτες ασύρματης ραδιοεπικοινωνίας.

Στον κόμβο ταυτοποίησης πραγματοποιείται τόσο η εκπαίδευση διαφόρων παθητικών ετικετών RFID (με την βοήθεια μίας κάρτας μνήμης microSD για την μόνιμη αποθήκευση τους) και η συσχέτιση αυτών με διάφορες απομακρυσμένες συσκευές (με την βοήθεια ενός απλού ανιχνευτή υπερύθρων και ενός τυπικού τηλεχειριστηρίου για την επιλογή της συσκευής) όσο και η σηματοδότηση του ανοίγματος ή του κλεισμάτος μιας συσκευής μετά την ψηφιακή ταυτοποίηση μίας γνωστής παθητικής ετικέτας RFID. Επίσης, ο κόμβος εμφανίζει διάφορα βοηθητικά μηνύματα στον τελικό χρήστη μέσω μίας τυπικής οθόνης υγρών κρυστάλλων.

Στον κόμβο ελέγχου διασυνδέονται οι διάφορες συσκευές, τις οποίες κυρίως μπορεί είτε να ανοίγει, είτε να κλείνει με την βοήθεια κατάλληλων ηλεκτρομηχανικών ρελέ μετά από την λήψη του κατάλληλου σήματος από το κόμβο ταυτοποίησης. Τέλος, αξίζει να αναφερθεί πως ο κόμβος ελέγχου μπορεί να ελέγξει τόσο συσκευές χαμηλών όσο και υψηλών ρευμάτων ή τάσεων (π.χ. μπορεί να ελέγξει είτε τυπικά ψηφιακά κυκλώματα είτε οικιακές συσκευές).

Περιεχόμενα

Περίληψη.....	3
Εισαγωγή.....	5
Ευχαριστίες.....	6
Περιγραφή Κεφαλαίων.....	7
Κεφάλαιο 1 - Πλατφόρμα Arduino.....	8
Κεφάλαιο 2 - Υλικό Πλατφόρμας Arduino.....	10
2.1 - Γενικά.....	10
2.2 - Arduino Duemilanove.....	14
Κεφάλαιο 3 - Προγραμματισμός Πλατφόρμας Arduino.....	20
3.1 - Γενικά.....	20
3.2 - Γλώσσα Προγραμματισμού.....	20
3.3 - Ολοκληρωμένο Περιβάλλον Ανάπτυξης.....	26
3.4 - Βιβλιοθήκες.....	30
3.5 - Παραδείγματα.....	31
Κεφάλαιο 4 - Επιμέρους Συστατικά Συστήματος.....	33
4.01 - Ράστερ & Καλώδια.....	33
4.02 - Κάρτα Επέκτασης / Κάρτα Μνήμης microSD.....	36
4.03 - Ανιχνευτής Υπερύθρων & Τηλεχειριστήριο.....	38
4.04 - Κάρτα Επέκτασης Αισθητήρων.....	40
4.05 - Ηλεκτρομηχανικά Ρελέ.....	41
4.06 - Κάρτες Ασύρματης Ραδιοεπικοινωνίας.....	42
4.07 - Προσαρμογέις Καρτών Ασύρματης Ραδιοεπικοινωνίας.....	43
4.08 - Οθόνη Υγρών Κρυστάλλων.....	45
4.09 - Καταχωρητής Ολίσθησης.....	47
4.10 - USB Καλώδια Πλατφορμών Arduino.....	49
4.11 - USB Τροφοδοτικά Τοίχου.....	50
4.12 - Συσκευή Ανάγνωσης RFID.....	51
4.13 - Προσαρμογέας Συσκευής Ανάγνωσης RFID.....	52
4.14 - Παθητικές Ετικέτες RFID.....	53
4.15 - Ηλεκτρική Δίοδος.....	55
4.16 - Ηλεκτρομηχανικό Ποτενσιόμετρο.....	56
4.17 - Ηλεκτρικοί Αντιστάτες.....	57
4.18 - Ηλεκτρομηχανικό Κουμπί.....	58
4.19 - Δίοδοι Εκπομπής Φωτός.....	59
Κεφάλαιο 5 - Διασύνδεση Συστατικών Συστήματος.....	60
Κεφάλαιο 6 - Προγραμματισμός Συστήματος.....	63
Ευρετήριο Εικόνων.....	91
Ευρετήριο Πινάκων.....	92
Ευρετήριο Προγραμμάτων.....	93
Βιβλιογραφία.....	94
Παράρτημα 1 - Κυκλώματα Συστατικών.....	95
Παράρτημα 2 - Φωτογραφίες Συστήματος.....	103
Παράρτημα 3 - Διαγράμματα Ροής.....	110
Παράρτημα 4 - Φύλλα Δεδομένων.....	133

Εισαγωγή

Αρχικά, τα πρώτα βήματα στην πορεία αυτής της πτυχιακής εργασίας ήταν η επιλογή και η μελέτη των κατάλληλων πλατφορμών Arduino που θα χρησιμοποιούνταν για την ανάπτυξη των βασικών ενσωματωμένων κόμβων του συστήματος. Πιο συγκεκριμένα, μελετήθηκε τόσο το υλικό τους μέρος όσο και η γλώσσα προγραμματισμού με την οποία αυτοί θα προγραμματίζονταν.

Παράλληλα, κρίθηκε απαραίτητη τόσο η επιλογή όσο και η μελέτη των κατάλληλων περιφερειακών εξαρτημάτων των ενσωματωμένων κόμβων του συστήματος. Επιπλέον, αξίζει να αναφερθεί πως μερικά από τα περιφερειακά εξαρτήματα χρειάστηκαν ειδικές κολλήσεις, κάτι που πραγματοποιήθηκε σε κατάλληλο εργαστηριακό χώρο με τον κατάλληλο ηλεκτρονικό εξοπλισμό.

Πιο συγκεκριμένα, μελετήσαμε ασύρματες κάρτες ραδιοεπικοινωνίας XBee, παθητικές ετικέτες RFID, συσκευές ανάγνωσης RFID, κάρτες επέκτασης αισθητήρων, ηλεκτρομηχανικά ρελέ, οθόνες υγρών κρυστάλλων, καταχωρητές ολίσθησης, ανιχνευτές υπερύθρων, κάρτες μνήμης microSD, κάρτες επέκτασης microSD, διόδους εκπομπής φωτός, ηλεκτρομηχανικά κουμπιά, κ.α.

Έτσι, αφού λοιπόν πραγματοποιήσαμε διεξοδική ανάλυση σχετικά με τα παραπάνω, καταλήξαμε στο υλικό που θα χρησιμοποιούσαμε για την ανάπτυξη των βασικών ενσωματωμένων κόμβων του συστήματος. Τέλος, μεταβήκαμε φυσιολογικά στην κατασκευαστική φάση όπου μέσα από μία επαναληπτική μεθοδολογία ανάπτυξης υλοποιήθηκαν σταδιακά οι ενσωματωμένοι κόμβοι του συστήματος αφού κάθε φορά προσθέταμε σε αυτούς τα διάφορα περιφερειακά εξαρτήματα και στην συνέχεια πραγματοποιούσαμε τις απαραίτητες αλλαγές στο υλικολογισμικό τους.

Ευχαριστίες

Η υλοποίηση αυτής της πτυχιακής εργασίας πραγματοποιήθηκε κυρίως στους εργαστηριακούς χώρους του Τμήματος Πληροφορικής & Επικοινωνιών του Τεχνολογικού Εκπαιδευτικού Ιδρύματος Σερρών με την υποστήριξη και την επίβλεψη του κ. Ιωάννη Μαδεμλή, εργαστηριακού συνεργάτη του τμήματος τον οποίο ευχαριστώ θερμά για την βοήθεια που μου παρείχε.

Ευστάθιος Χατζηκυριακίδης

Περιγραφή Κεφαλαίων

Στο **πρώτο** κεφάλαιο θα γίνει μία μικρή εισαγωγή όσο αφορά την πλατφόρμα Arduino. Συγκεκριμένα, σε αυτό το κεφάλαιο θα απαντηθούν ερωτήματα όπως : “Τι είναι μία πλατφόρμα Arduino;”, “Ποιος ξεκίνησε την δημιουργία της πλατφόρμας Arduino;”, “Ποια είναι η άδεια της πλατφόρμας Arduino;” και “Ποια είναι τα πιο βασικά πλεονεκτήματα της πλατφόρμας Arduino;”.

Στο **δεύτερο** κεφάλαιο θα μελετηθεί γενικά το υλικό μέρος μιας τυπικής πλατφόρμας Arduino και στην συνέχεια θα γίνει συγκεκριμένα η διεξοδική περιγραφή των επιμέρους συστατικών της πλατφόρμας Arduino Duemilanove.

Στο **τρίτο** κεφάλαιο θα αναλυθεί η γλώσσα προγραμματισμού και το ολοκληρωμένο περιβάλλον ανάπτυξης που είναι απαραίτητα για τον προγραμματισμό μίας πλατφόρμας Arduino. Στην συνέχεια, θα γίνει μία σύντομη αναφορά για τις διάφορες βιβλιοθήκες της πλατφόρμας Arduino και τελικά το κεφάλαιο θα ολοκληρωθεί μαζί με δύο απλά προγράμματα Arduino όπου ο αναγνώστης καλείται να εξοικειωθεί μέσω της ανάγνωσης αυτών με τις βασικές δομές της γλώσσας προγραμματισμού της πλατφόρμας Arduino.

Στο **τέταρτο** κεφάλαιο θα γίνει η ανάλυση των συστατικών του συστήματος, δηλαδή θα επεξηγηθούν ανεξάρτητα όλα τα περιφερειακά εξαρτήματα που χρησιμοποιούνται στους ενσωματωμένους κόμβους του συστήματος.

Στο **πέμπτο** κεφάλαιο θα παρουσιαστεί ο τρόπος διασύνδεσης όλων των συστατικών του συστήματος, δηλαδή των περιφερειακών εξαρτημάτων με τους ενσωματωμένους κόμβους του συστήματος.

Στο **έκτο** κεφάλαιο θα δοθεί ο πηγαίος κώδικας τόσο των βιβλιοθηκών όσο και των υλικολογισμικών που χρησιμοποιούνται στους ενσωματωμένους κόμβους του συστήματος.

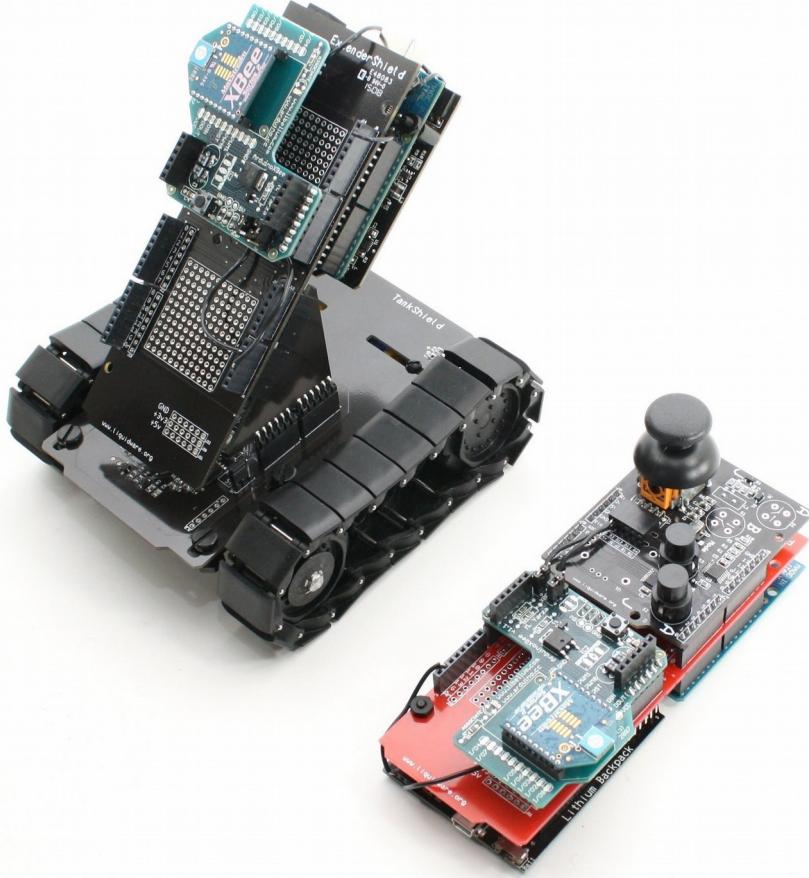
Αμέσως μετά, ακολουθούν αναλυτικά μερικά ευρετήρια για τις εικόνες, τους πίνακες, καθώς και για τα προγράμματα που βρίσκονται διασκορπισμένα στα κεφάλαια που αναφέρθηκαν παραπάνω.

Τέλος, ακολουθούν οι βιβλιογραφικές αναφορές της πτυχιακής εργασίας καθώς και τα παραρτήματα στα οποία υπάρχουν διάφορα σχεδιαγράμματα κυκλωμάτων, φωτογραφίες, φύλλα δεδομένων και διαγράμματα ροής που σχετίζονται με τα συστατικά υλικού και υλικολογισμικού του συστήματος.

Κεφάλαιο 1 - Πλατφόρμα Arduino

Το Arduino είναι μία φυσική υπολογιστική πλατφόρμα ανοιχτού λογισμικού, η οποία σχεδιάστηκε κυρίως για να καταστήσει πιο προσιτή, οικονομική και διασκεδαστική την ανάπτυξη ενσωματωμένων συστημάτων.

Τόσο το υλικό όσο και το λογισμικό της πλατφόρμας Arduino είναι ελεύθερα και ανοιχτά για όλους. Αυτό σημαίνει πως τα αρχιτεκτονικά σχέδια της πλατφόρμας είναι άμεσα προσβάσιμα με αποτέλεσμα να μπορεί οποιοσδήποτε να επιθυμεί να αναπτύξει μία δική του πλατφόρμα Arduino.



Εικόνα 01 : Ασύρματα ελεγχόμενο επίγειο όχημα υλοποιημένο με Arduino

Επιπλέον, η γλώσσα προγραμματισμού, οι διάφορες βιβλιοθήκες και το ολοκληρωμένο περιβάλλον ανάπτυξης που υπάρχουν για τον προγραμματισμό της πλατφόρμας Arduino αποτελούν ανοιχτό λογισμικό προσφέροντας έτσι ανεκτίμητη γνώση σε όλους.

Πιο συγκεκριμένα, τα αρχιτεκτονικά σχέδια της πλατφόρμας Arduino είναι υπό την άδεια "Creative Commons Attribution Share - Alike", ο πηγαίος κώδικας του ολοκληρωμένου περιβάλλοντος ανάπτυξης είναι υπό την άδεια "GNU General Public License" και οι βιβλιοθήκες για τον προγραμματισμό της πλατφόρμας είναι υπό την άδεια "GNU Lesser General Public License".

Οι πλατφόρμες Arduino κατασκευάζονται κυρίως από την εταιρία Smart Projects. Ωστόσο, το Arduino ξεκίνησε ως έργο προς ανάπτυξη το 2005 στην Ιταλία, στο Ινστιτούτο Αλληλεπιδραστικής Σχεδίασης Icrea ώστε οι φοιτητές του Ινστιτούτου να αναπτύσσουν ενσωματωμένα συστήματα οικονομικά και αποδοτικά αξιοποιώντας τις δυνατότητες και τις ευκαιρίες που μπορεί να προσφέρει το ελεύθερο λογισμικό.



Εικόνα 02 : Η βασική ομάδα ανάπτυξης της πλατφόρμας Arduino

Ακολουθούν μερικά βασικά πλεονεκτήματα της πλατφόρμας Arduino :

➤ **Οικονομική**

Σε σχέση με τις υπάρχουσες παρόμοιες πλατφόρμες στο εμπόριο η πλατφόρμα Arduino αποτελεί πολύ καλύτερη οικονομική λύση διότι είναι η φθηνότερη. Επίσης, εφόσον η πλατφόρμα Arduino είναι αρχιτεκτονικά ανοιχτή, μπορείτε να την αναπτύξετε και μόνοι σας. Σε μία τέτοια περίπτωση το κόστος ίσως είναι μηδαμινό.

➤ **Μεταφέρσιμη**

Σε σχέση με τις υπάρχουσες παρόμοιες πλατφόρμες στο εμπόριο όπου η εκτέλεση του ολοκληρωμένου περιβάλλοντος ανάπτυξης τους περιορίζεται σε συγκεκριμένα λειτουργικά συστήματα η πλατφόρμα Arduino παρέχει πλήρη μεταφερσιμότητα με αποτέλεσμα να μπορεί να προγραμματιστεί στα περισσότερα λειτουργικά συστήματα.

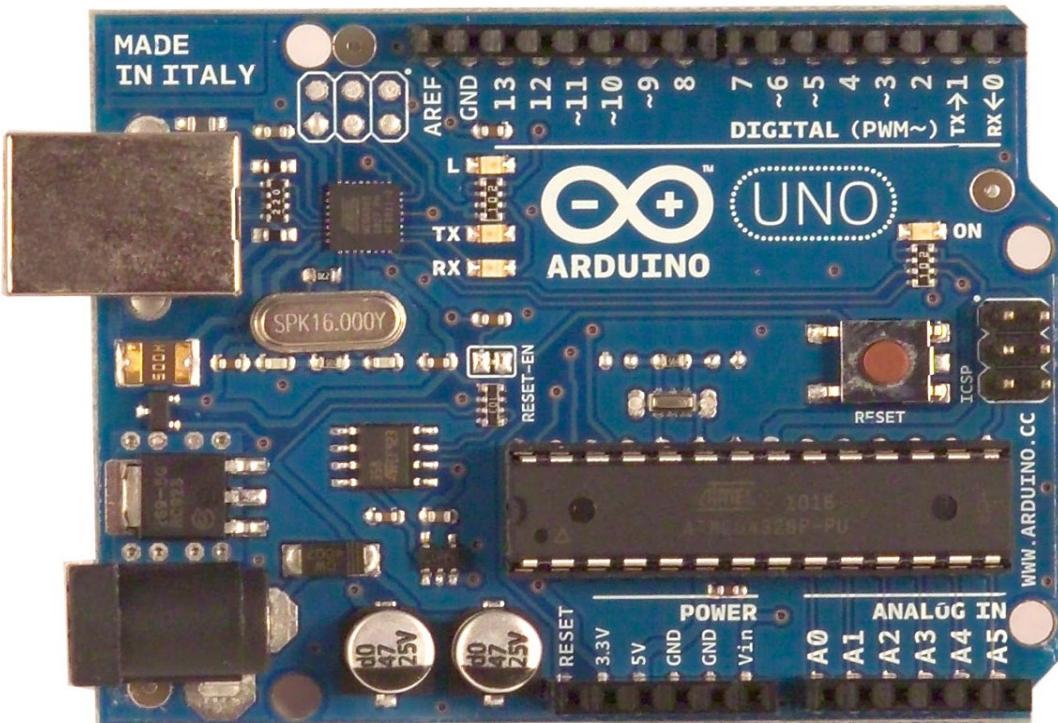
➤ **Επεκτάσιμη**

Τόσο το υλικό όσο και το λογισμικό της πλατφόρμας Arduino είναι ανοιχτά και ελεύθερα για όλους. Καθημερινά, χιλιάδες υποστηρικτές του ελεύθερου λογισμικού αναπτύσσουν διάφορες βιβλιοθήκες για την υποστήριξη της πλατφόρμας. Παράλληλα, τόσο η αρχιτεκτονική όσο και το υλικό της πλατφόρμας εξελίσσονται συνεχώς.

Κεφάλαιο 2 - Υλικό Πλατφόρμας Arduino

2.1 - Γενικά

Όλες οι πλατφόρμες Arduino αποτελούνται κυρίως από έναν CMOS 8 ψηφίων Atmel AVR μικροελεγκτή (της σειράς megaAVR) και συμπληρωματικά εξαρτήματα για τον εύκολο προγραμματισμό τους.



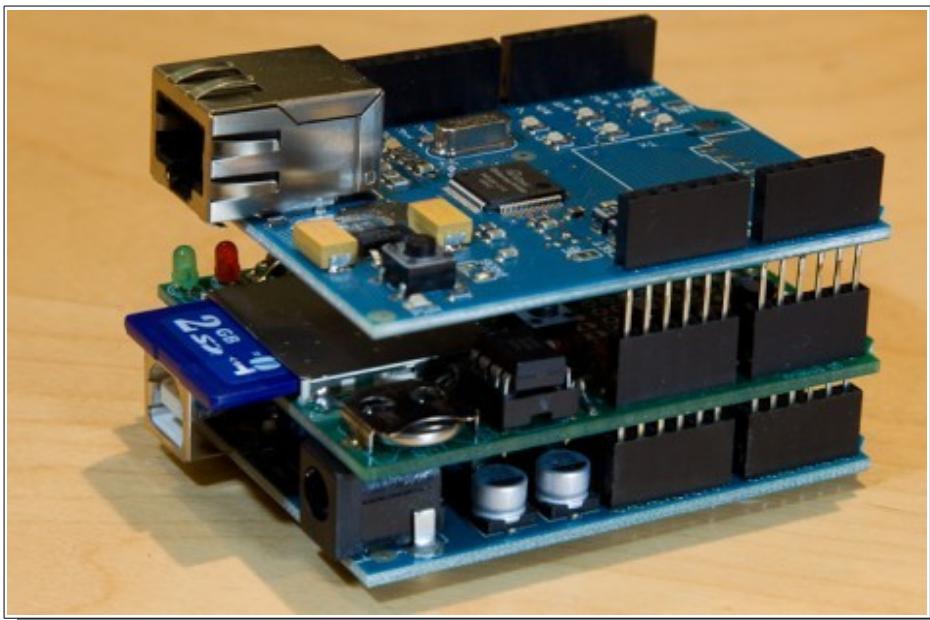
Εικόνα 03 : Η κάτοψη της πλατφόρμας Arduino Uno

Επίσης, οι πλατφόρμες Arduino διαθέτουν τρεις βασικές μνήμες :

- Μνήμη Flash, στην οποία τοποθετείται κάθε φορά το πρόγραμμα που πρόκειται να εκτελεστεί καθώς και ο φορτωτής εκκίνησης (συνήθως τοποθετείται από τον κατασκευαστή) που διευκολύνει την διαδικασία του προγραμματισμού της πλατφόρμας.
- Μνήμη EEPROM, για την κατά βούληση αποθήκευση μόνιμων δεδομένων.
- Μνήμη SRAM, η οποία χρησιμοποιείται για την προσωρινή αποθήκευση των στατικών και των μεταβλητών δεδομένων του προγράμματος που εκτελείται.

Ένα βασικό στοιχείο των πλατφορμών Arduino είναι ο πρότυπος τρόπος με τον οποίο εξάγονται οι επαφές εισόδου και εξόδου που διαθέτουν, επιτρέποντας έτσι την σύνδεση τους με άλλες εξωτερικές πλακέτες ή κάρτες επέκτασης.

Τις περισσότερες φορές, οι εξωτερικές κάρτες επέκτασης ακολουθούν το ίδιο πρότυπο, με αποτέλεσμα να μπορούν να διασυνδεθούν μεταξύ τους καταστρώνοντας στρώματα όπου κάθε ένα αυτά παρέχει μία συγκεκριμένη δυνατότητα (π.χ. σύνδεση σε ασύρματο WiFi ή ενσύρματο Ethernet δίκτυο, μόνιμη αποθήκευση δεδομένων σε SD μνήμες, κ.α.).



Εικόνα 04 : Μία πλατφόρμα Arduino με κάρτες επέκτασης (Ethernet και SD)

Παράλληλα, οι πλατφόρμες Arduino έχουν εκτεθειμένες τις περισσότερες επαφές εισόδου και εξόδου για χρήση με άλλα κυκλώματα. Για παράδειγμα, η πλατφόρμα Diecimila, παρέχει 14 ψηφιακές επαφές εισόδου και εξόδου (από τις οποίες οι 6 μπορούν να παράξουν σήματα PWM) και 6 αναλογικές εισόδους.

Όσο αφορά τις αναλογικές εισόδους χρησιμοποιούνται κατάλληλα κανάλια μετατροπής του αναλογικού σήματος σε ψηφιακό με ανάλυση 10 ψηφίων. Όλες οι επαφές μίας πλατφόρμας Arduino είναι διαθέσιμες στην κορυφή αυτής μέσω θηλυκών συνδέσεων μεγέθους 0.1 ιντσών.

Επίσης, αξίζει να αναφερθεί πως κάθε μία ψηφιακή επαφή της πλατφόρμας Arduino διαθέτει έναν εσωτερικό PULL-UP αντιστάτη (ο οποίος μπορεί να ενεργοποιηθεί προγραμματιστικά). Παράλληλα, το ρεύμα DC που διαρρέεται και αντιστοιχεί σε κάθε ψηφιακή επαφή της πλατφόρμας Arduino είναι 40mA.

Επιπλέον, στις περισσότερες πλατφόρμες Arduino υπάρχουν και διάφορες άλλες επαφές. Για παράδειγμα, μία επαφή αναλογικής αναφοράς (όσο αφορά την τάση), τρεις επαφές γείωσης (η μία είναι ψηφιακή), δύο επαφές διαφορετικής τάσης τροφοδοσίας, επαφή λήψης (RX) και αποστολής (TX) δεδομένων για την σειριακή επικοινωνία. Τις περισσότερες φορές, κάποιες από τις ψηφιακές επαφές μπορούν να χρησιμοποιηθούν για SPI ή I²C επικοινωνία ενώ κάποιες άλλες για την υποστήριξη εξωτερικών σημάτων διακοπής.

Παρακάτω, ακολουθεί ένας πίνακας όπου περιέχει για τις πιο τυπικές πλατφόρμες Arduino τα πιο βασικά χαρακτηριστικά όσο αφορά το υλικό τους μέρος.

Πλατφόρμα Arduino	Μικροελεγκτής Atmel AVR	Flash KiB	EEPROM KiB	SRAM KiB	Ψηφιακές Επαφές Ε / Ε	PWM	Αναλογικές Επαφές Εισόδου
Diecimila	ATmega168	16	0.5	1	14	6	6
Duemilanove	ATmega168/328	16	0.5	1	14	6	6
Uno	ATmega328	32	1	2	14	6	6
Mega	ATmega1280	128	4	8	54	14	16
Fio	ATmega328P	32	1	2	14	6	8
Mega2560	ATmega2560	256	4	8	54	14	16

Πίνακας 1 : Βασικά χαρακτηριστικά υλικού τυπικών πλατφορμών Arduino

Επίσης, οι πλατφόρμες Arduino έχουν μία δίοδο εκπομπής φωτός για την ένδειξη της τροφοδοσίας καθώς και ένα κουμπί επαναρχικοποίησης όπου με το πάτημα του ξεκινάει ξανά η εκτέλεση του προγράμματος που βρίσκεται αποθηκευμένο μέσα στην μνήμη Flash της πλατφόρμας.

Αξίζει να σημειωθεί πως βασικό και αναπόσπαστο εξάρτημα κάθε πλατφόρμας Arduino είναι ένας κρυσταλλικός ταλαντωτής 8MHz ή 16MHz (ανάλογα με την έκδοση της πλατφόρμας) ο οποίος παράγει παλμούς που χρονίζουν τον μικροελεγκτή.

Ο μικροελεγκτής σε κάθε πλατφόρμα Arduino είναι προγραμματισμένος και περιέχει έναν ειδικό φορτωτή εκκίνησης ο οποίος απλοποιεί κυρίως την διαδικασία της τοποθέτησης προγραμμάτων στην ενσωματωμένη μνήμη Flash της πλατφόρμας, με αποτέλεσμα να μην είναι απαραίτητη κάποια εξωτερική συσκευή προγραμματισμού μικροελεγκτών AVR.

Σε περίπτωση όπου επιθυμούμε να χρησιμοποιήσουμε κάποια εξωτερική συσκευή προγραμματισμού AVR μικροελεγκτών θα πρέπει να αξιοποιήσουμε τις επαφές ICSP που υπάρχουν επάνω στην πλατφόρμα Arduino.

Γενικά, οι περισσότερες πλατφόρμες Arduino είναι προγραμματισμένες μέσω μιας σειριακής σύνδεσης RS-232, αλλά ο τρόπος με τον οποίο αυτό υλοποιείται ποικίλει ανάλογα με την έκδοση. Οι σειριακές πλατφόρμες Arduino περιέχουν ένα απλό κύκλωμα αντιστροφής για την μετατροπή των σημάτων ανάμεσα στα επίπεδα RS-232 και TTL.

Οι πιο πρόσφατες πλατφόρμες Arduino μπορούν να προγραμματιστούν μέσω της θύρας USB που διαθέτουν, με την βοήθεια όμως ενός κυκλώματος μετατροπής από USB σε σειριακή όπως το FTDI FT232. Ωστόσο, υπάρχουν πλατφόρμες (π.χ. Arduino Uno) όπου δεν βασίζονται στο κύκλωμα FTDI για την μετατροπή από USB σε σειριακή αλλά αξιοποιούν έναν τυπικό μικροελεγκτή ATmega8U2, ο οποίος προγραμματίζεται κατάλληλα ώστε να λειτουργεί ως μετατροπέας από USB σε σειριακή.

Επιπλέον, οι πλατφόρμες Arduino συνήθως διαθέτουν δύο μικρές διόδους εκπομπής φωτός, μία για την λήψη και μία για την αποστολή δεδομένων μέσω των γραμμών RX, TX της σειριακής επικοινωνίας καθώς και μία δίοδο εκπομπής φωτός (με ενσωματωμένο αντιστάτη) συνδεδεμένη με τη ψηφιακή επαφή 13.

Αξίζει να αναφερθεί πως οι πλατφόρμες Arduino διαθέτουν και υποδοχή για εξωτερική τροφοδοσία όπου το οριακό εύρος της τάσης εισόδου πρέπει να είναι 6-20V. Το προτεινόμενο εύρος της τάσης εισόδου ώστε η πλατφόρμα να είναι λειτουργική είναι 7-12V. Επίσης, η τάση λειτουργίας της πλατφόρμας Arduino είναι 5V (φυσικά η πλατφόρμα διαθέτει έναν ενσωματωμένο γραμμικό ρυθμιστή τάσης ο οποίος μετατρέπει την τάση εισόδου κατάλληλα).

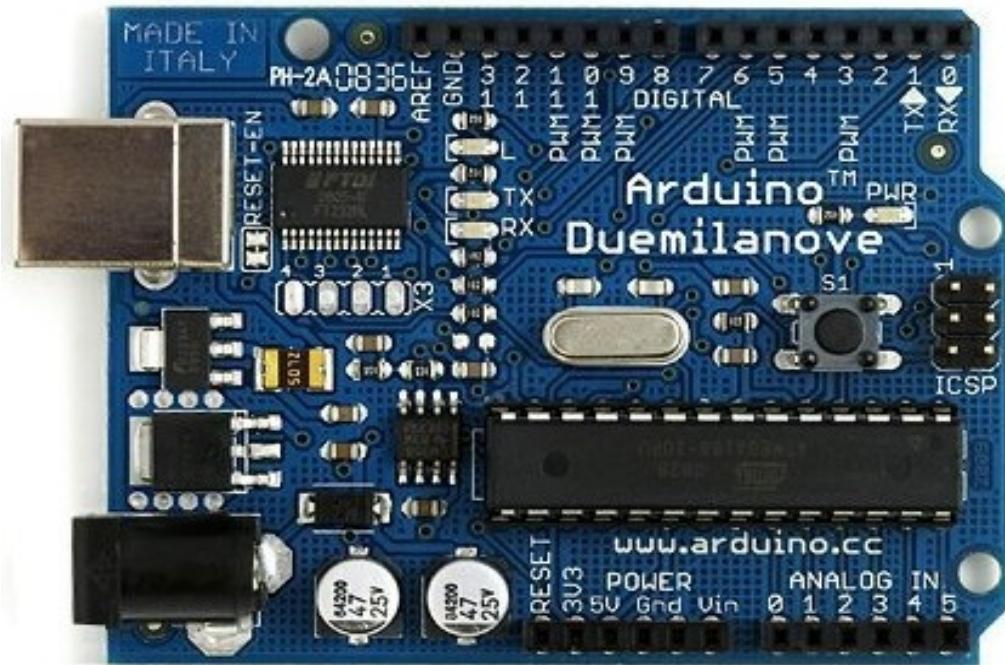
Μία πλατφόρμα Arduino μπορεί να τροφοδοτηθεί τόσο από την θύρα USB όσο και από την υποδοχή εξωτερικής τροφοδοσίας. Σε παλιότερες εκδόσεις η επιλογή της πηγής τροφοδοσίας γινόταν με ειδικό βραχυκυκλωτήρα που βρισκόταν επάνω στην πλατφόρμα. Παρόλα αυτά, στις τελευταίες πλατφόρμες Arduino η επιλογή της πηγής τροφοδοσίας γίνεται αυτόματα από την ίδια την πλατφόρμα.

Παρακάτω ακολουθούν σε τυχαία σειρά μερικές από τις πλατφόρμες Arduino που έχουν αναπτυχθεί και όπου η κάθε μία είτε αποτελεί εξέλιξη κάποιας άλλης, είτε έχει αναπτυχθεί για κάποιο συγκεκριμένο σκοπό :

- Serial Arduino
- Arduino Extreme
- Arduino USB
- Arduino Fio
- Arduino Mini
- Arduino Nano
- LilyPad Arduino
- Arduino Stamp
- Arduino NG
- Arduino NG+
- Arduino Bluetooth
- Arduino Diecimila
- Arduino Duemilanove
- Arduino Mega1280
- Arduino Uno
- Arduino Mega2560

2.2 - Arduino Duemilanove

Η πλατφόρμα Arduino Duemilanove (2009) αποτελεί το θεμέλιο των ενσωματωμένων κόμβων του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία. Η μελέτη που πραγματοποιήθηκε μας οδήγησε στην επιλογή αυτής της πλατφόρμας για την κατασκευή των ενσωματωμένων κόμβων του συστήματος.



Εικόνα 05 : Η κάτωψη της πλατφόρμας Arduino Duemilanove

Μέχρι στιγμής, έχει γίνει αναφορά στα πιο βασικά χαρακτηριστικά υλικού που συνηθίζουν να έχουν οι περισσότερες τυπικές πλατφόρμες Arduino. Μία από αυτές τις πλατφόρμες είναι και η Arduino Duemilanove. Για τον λόγο αυτό παρακάτω δεν θα καταγραφούν ξανά αναλυτικά όλα τα χαρακτηριστικά της πλατφόρμας αλλά θα αναφερθούν μόνο τα πιο βασικά.

➤ Συνοπτικά

Η πλατφόρμα Arduino Duemilanove διαθέτει εγκατεστημένο είτε τον μικροελεγκτή ATmega168, είτε τον ATmega328. Επίσης, εξάγει στην κορυφή της 14 ψηφιακές επαφές εισόδου και εξόδου (από τις οποίες οι 6 μπορούν να παράξουν σήματα PWM) και 6 αναλογικές εισόδους.

Επιπλέον, η πλατφόρμα παρέχει USB συνδεσιμότητα (η σύνδεση σε Η/Υ μπορεί να γίνει με ένα καλώδιο USB τύπου A-B), έναν κρυσταλλικό ταλαντωτή 16MHz, μία υποδοχή για εξωτερική τροφοδοσία και ένα κουμπί επαναρχικοποίησης.

Για την τροφοδοσία της πλατφόρμας μπορεί να χρησιμοποιηθεί τόσο η θύρα USB με την κατάλληλη σύνδεση σε H/Y (σταθερό ή φορητό) όσο και η υποδοχή της εξωτερικής τροφοδοσίας με κάποιο μετατροπέα AC σε DC (εναλλακτικά μπορεί να χρησιμοποιηθεί μπαταρία).

➤ Περιληπτικά

- Μικροελεγκτής : ATmega168 ή ATmega328
- Λειτουργική τάση : 5V
- Τάση εισόδου (προτεινόμενη) : 7-12V
- Τάση εισόδου (όρια) : 6-20V
- Ψηφιακές επαφές εισόδου και εξόδου : 14 (οι 6 παρέχουν PWM έξοδο)
- Αναλογικές επαφές εισόδου : 6
- Ρεύμα DC ανά επαφή εισόδου και εξόδου : 40mA
- Ρεύμα DC για την επαφή 3.3V : 50mA
- Μνήμη SRAM : 1KB (ATmega168) ή 2KB (ATmega328)
- Μνήμη Flash : 16KB (ATmega168) ή 32KB (ATmega328)
- Μνήμη EEPROM : 0.5KB (ATmega168) ή 1KB (ATmega328)
- Μνήμη σε χρήση : 2KB της μνήμης Flash για τον φορτωτή εκκίνησης
- Ταχύτητα ρολογιού : 16MHz

➤ Σχεδιάγραμμα

Το κύκλωμα της πλατφόρμας βρίσκεται στο παράρτημα 1.

➤ Τροφοδοσία

Η πλατφόρμα μπορεί να τροφοδοτηθεί είτε μέσω της θύρας USB, είτε μέσω κάποιας εξωτερικής τροφοδοσίας από την ειδική υποδοχή που διαθέτει. Η επιλογή της πηγής τροφοδοσίας γίνεται αυτόματα από την ίδια την πλατφόρμα.

Το οριακό εύρος της τάσης εισόδου είναι 6-20V. Το προτεινόμενο εύρος της τάσης εισόδου ώστε η πλατφόρμα να είναι λειτουργική είναι 7-12V. Σε περίπτωση που η τάση εισόδου είναι λιγότερη από 7V τότε η επαφή που σημαίνεται με "5V" δεν εξάγει τάση 5V με αποτέλεσμα η πλατφόρμα να είναι αναξιόπιστη.

Επίσης, σε περίπτωση που η τάση εισόδου είναι περισσότερη από 12V τότε ο ενσωματωμένος γραμμικός ρυθμιστής τάσης που υπάρχει στην πλατφόρμα πιθανόν να υπερθερμανθεί με αποτέλεσμα η πλατφόρμα να υποστεί βλάβες.

Ακολουθούν οι διάφορες επαφές τάσης / γείωσης της πλατφόρμας :

- **VIN**

Σε αυτή την επαφή η τάση είναι ίση με την τάση εισόδου από την οποία τροφοδοτείται η πλατφόρμα όταν χρησιμοποιείται μία πηγή τροφοδοσίας.

- **5V**

Σε αυτή την επαφή η τάση είναι ίση με την τάση εξόδου του γραμμικού ρυθμιστή τάσης της πλατφόρμας. Επίσης, η τάση αυτή (5V) τροφοδοτεί τα επιμέρους κυκλώματα της πλατφόρμας.

- **3V3**

Σε αυτή την επαφή η τάση είναι ίση με 3.3V και παράγεται από το ειδικό κύκλωμα FTDI που βρίσκεται ενσωματωμένο μέσα στην πλατφόρμα.

- **GND**

Η πλατφόρμα διαθέτει συνολικά 3 επαφές γείωσης.

➤ **Μνήμη**

Ο μικροελεγκτής ATmega168 διαθέτει μνήμη Flash χωρητικότητας 16KB, μνήμη SRAM χωρητικότητας 1KB καθώς και μνήμη EEPROM χωρητικότητας 0.5KB. Παράλληλα, ο ATmega328 διαθέτει μνήμη Flash χωρητικότητας 32KB, μνήμη SRAM χωρητικότητας 2KB καθώς και μνήμη EEPROM χωρητικότητας 1KB.

➤ Είσοδος και Έξοδος

Κάθε μία από τις 14 ψηφιακές επαφές της πλατφόρμας μπορεί να χρησιμοποιηθεί τόσο για είσοδο όσο και για έξοδο. Αυτές οι επαφές λειτουργούν στα 5V και μπορούν να παρέχουν ή να λάβουν μέγιστο ρεύμα 40mA. Επίσης, οι επαφές διαθέτουν και ενσωματωμένο PULL-UP αντιστάτη 20-50KΩ. Εξορισμού οι αντιστάτες είναι αποσυνδεδεμένοι και μπορούν να ενεργοποιηθούν προγραμματιστικά.

Επίσης, αξίζει να αναφερθεί πως σε κάθε πλατφόρμα Arduino οι διάφορες επαφές εισόδου και εξόδου που υπάρχουν είναι πιθανό να παρέχουν και δευτερεύουσες λειτουργίες :

- **Σειριακή επικοινωνία - επαφές 0 (RX), 1 (TX)**

Οι επαφές αυτές αναγράφονται ως TX, RX και χρησιμοποιούνται για την αποστολή ή την λήψη TTL σειριακών δεδομένων. Αυτές οι επαφές συνδέονται εσωτερικά στην πλατφόρμα με τις κατάλληλες επαφές του ενσωματωμένου FTDI κυκλώματος μετατροπής USB σε TTL.

- **Εξωτερικά σήματα διακοπής - επαφές 2, 3**

Αυτές οι επαφές μπορούν να χρησιμοποιηθούν για την υποστήριξη εξωτερικών σημάτων διακοπής. Επίσης, η εξυπηρέτηση ενός σήματος διακοπής μπορεί να πραγματοποιηθεί όταν η στάθμη της τάσης αντιστοιχεί στο λογικό 0, όταν αλλάζει τιμή καθώς και στο θετικό ή αρνητικό μέτωπο των σημάτων.

- **PWM - επαφές 3, 5, 6, 9, 10, 11**

Οι επαφές αυτές παρέχουν ψηφιακή PWM έξοδο (ανάλυσης 8 ψηφίων).

- **SPI - επαφές 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)**

Οι επαφές αυτές χρησιμοποιούνται για SPI επικοινωνία.

- **Αναφορά τάσης - επαφή AREF**

Επαφή για την αναφορά τάσης των αναλογικών επαφών εισόδου.

- **Αναλογικές επαφές εισόδου - επαφές 0, 1, 2, 3, 4, 5**

Η πλατφόρμα διαθέτει 6 αναλογικές επαφές εισόδου και κατάλληλα κανάλια για την μετατροπή του σήματος εισόδου από αναλογικό σε ψηφιακό με ανάλυση 10 ψηφίων.

- **Δίοδος εκπομπής φωτός - επαφή 13**

Στην πλατφόρμα υπάρχει μία ενσωματωμένη δίοδο εκπομπής φωτός την οποία μπορούμε να χρησιμοποιούμε όποτε το επιθυμούμε (αυτή η δίοδος είναι συνδεδεμένη με την επαφή 13).

- **Επαναρχικοποίηση - επαφή Reset**

Η επαφή αυτή μπορεί να χρησιμοποιηθεί για να επαναρχικοποιήσουμε τον μικροελεγκτή της πλατφόρμας με αποτέλεσμα να ξεκινήσει η εκτέλεση του προγράμματος ξανά από την αρχή.

➤ Αντιστοίχηση μεταξύ επαφών ATmega168/328 και Arduino

Αειτουργία Arduino	Επαφές ATmega168/328	Αειτουργία Arduino	
Επαναρχικοποίηση	01	28	Αναλογική είσοδος 5
Ψηφιακή επαφή 0 (RX)	02	27	Αναλογική είσοδος 4
Ψηφιακή επαφή 1 (TX)	03	26	Αναλογική είσοδος 3
Ψηφιακή επαφή 2	04	25	Αναλογική είσοδος 2
Ψηφιακή επαφή 3 (PWM)	05	24	Αναλογική είσοδος 1
Ψηφιακή επαφή 4	06	23	Αναλογική είσοδος 0
Τάση (VCC)	07	22	Γείωση (GND)
Γείωση (GND)	08	21	Αναλογική αναφορά
Κρυσταλλικός ταλαντωτής	09	20	Τάση (VCC)
Κρυσταλλικός ταλαντωτής	10	19	Ψηφιακή επαφή 13
Ψηφιακή επαφή 5 (PWM)	11	18	Ψηφιακή επαφή 12
Ψηφιακή επαφή 6 (PWM)	12	17	Ψηφιακή επαφή 11 (PWM)
Ψηφιακή επαφή 7	13	16	Ψηφιακή επαφή 10 (PWM)
Ψηφιακή επαφή 8	14	15	Ψηφιακή επαφή 9 (PWM)

Πίνακας 2 : Αντιστοίχηση μεταξύ επαφών ATmega168/328 και Arduino

➤ Επικοινωνία

Η πλατφόρμα μπορεί να συνδεθεί και να επικοινωνήσει ενσύρματα με H/Y (σταθερό ή φορητό), με πλατφόρμα Arduino ίδιας ή διαφορετικής έκδοσης και με άλλους μικροελεγκτές.

Τόσο, ο μικροελεγκτής ATmega168, όσο και ο ATmega328 παρέχουν σειριακή TTL (5V) επικοινωνία. Η επικοινωνία αυτή μπορεί να επιτευχθεί με την χρήση των ψηφιακών επαφών 0 (RX), 1 (TX).

Επίσης, πάνω στην πλατφόρμα υπάρχει ένα κύκλωμα προσαρμογής FTDI FT232RL το οποίο επιτρέπει την επικοινωνία μεταξύ USB και σειριακής.

Επιπλέον, οι μικροελεγκτές ATmega168 και ATmega328 μπορούν να υποστηρίζουν και SPI ή I²C επικοινωνία με την βοήθεια των κατάλληλων βιβλιοθηκών λογισμικού και των κατάλληλων επαφών εισόδου και εξόδου.

Κεφάλαιο 3 - Προγραμματισμός Πλατφόρμας Arduino

3.1 - Γενικά

Ο προγραμματισμός μιας πλατφόρμας Arduino μπορεί να γίνει με την χρήση της κατάλληλης γλώσσας προγραμματισμού καθώς και του κατάλληλου ολοκληρωμένου περιβάλλοντος ανάπτυξης. Προαιρετικά, μπορούν επίσης να χρησιμοποιηθούν και διάφορες βιβλιοθήκες που παρέχουν δυνατότητες τόσο σε επίπεδο υλικού όσο και σε επίπεδο λογισμικού.

Παράλληλα, ίσως να χρειάζονται και οι κατάλληλοι οδηγοί FTDI για την αναγνώριση και τον προγραμματισμό της πλατφόρμας Arduino από τον Η/Υ μέσω USB. Σε κάθε περίπτωση είναι απαραίτητο ο μικροελεγκτής AVR της πλατφόρμας να είναι προγραμματισμένος ώστε να περιέχει τον κατάλληλο φορτωτή εκκίνησης.

Επίσης, η πλατφόρμα Arduino βελτιώνεται και εξελίσσεται καθημερινά. Αυτό συμβαίνει διότι τόσο το υλικό, όσο και το λογισμικό της πλατφόρμας είναι ανοιχτά και ελεύθερα. Έτσι, αναπτύσσονται καθημερινά νέες βιβλιοθήκες και πραγματοποιούνται βελτιώσεις τόσο στην γλώσσα προγραμματισμού όσο και στο ολοκληρωμένο περιβάλλον ανάπτυξης της πλατφόρμας.

3.2 - Γλώσσα Προγραμματισμού

Η πλατφόρμα Arduino διαθέτει επάνω της έναν Atmel AVR μικροελεγκτή. Αυτό σημαίνει πως σε περίπτωση που επιθυμούμε να την προγραμματίσουμε μπορούμε να χρησιμοποιήσουμε την συμβολική γλώσσα του μικροελεγκτή AVR που χρησιμοποιεί.

Ωστόσο, η πλατφόρμα Arduino μπορεί να προγραμματιστεί και με τις γλώσσες προγραμματισμού C, C++. Οι μεταγλωττιστές που χρησιμοποιούνται για την μετατροπή του πηγαίου κώδικα C, C++ σε γλώσσα μηχανής του ανάλογου μικροελεγκτή AVR είναι οι avr-gcc και avr-g++ (οι δύο αυτοί μεταγλωττιστές αποτελούν ελεύθερο λογισμικό).

Έχοντας λοιπόν τόσο τους παραπάνω μεταγλωττιστές όσο και τις γλώσσες προγραμματισμού C, C++ μπορούμε να προγραμματίσουμε την πλατφόρμα Arduino. Παρόλα αυτά, με αυτό το τρόπο κάθε φορά που θα χρειαζόταν να χειριστούμε την πλατφόρμα (π.χ. να εργαστούμε με τα σήματα διακοπής ή με τις επαφές εισόδου και εξόδου) θα έπρεπε να ενσωματώνουμε συμβολική γλώσσα AVR μέσα στο πηγαίο κώδικα C, C++.

Επειδή όμως, η πλατφόρμα Arduino αναπτύχθηκε για να προσφέρει τόσο απλότητα όσο και μεταφερσιμότητα, συνοδεύεται μαζί με κατάλληλες C, C++ βιβλιοθήκες οι οποίες αποκρύπτουν όλες τις χαμηλού επιπέδου λεπτομέρειες της πλατφόρμας και περιέχουν συναρτήσεις για τον χειρισμό του υλικού της σε υψηλότερο επίπεδο.

Επίσης, αξίζει να αναφερθεί πως είναι δυνατή η χρήση διαδικαστικού, δομημένου και αντικειμενοστραφούς προγραμματισμού για την ανάπτυξη ενός προγράμματος Arduino. Παρακάτω, ακολουθούν μερικές από τις πιο βασικές δομές και λειτουργίες που μπορείτε να αξιοποιήσετε ως εργαλεία κατά την συγγραφή ενός προγράμματος Arduino :

➤ Δομή τυπικού προγράμματος Arduino

Η βασική δομή ενός προγράμματος Arduino αναλύεται τουλάχιστον σε δύο θεμελιώδεις συναρτήσεις. Οι συναρτήσεις αυτές είναι η `setup()` και η `loop()`. Η συνάρτηση `setup()` εκτελείται μόνο μία φορά και συνήθως χρησιμοποιείται για την αρχικοποίηση τόσο του προγράμματος όσο και της πλατφόρμας Arduino. Ενώ, η συνάρτηση `loop()` επανεκτελείται συνεχώς πραγματοποιώντας την βασική εργασία του προγράμματος.

➤ Δομές ελέγχου ροής

- `if` (δομή ελέγχου μίας συνθήκης)
- `if ... else` (δομή ελέγχου πολλαπλών συνθηκών)
- `for` (δομή επαναληπτικού ελέγχου συνθήκης)
- `while` (δομή επαναληπτικού ελέγχου συνθήκης)
- `do ... while` (δομή επαναληπτικού ελέγχου συνθήκης)
- `switch ... case` (δομή ελέγχου περιπτώσεων)
- `break` (εντολή διακοπής μιας επαναληπτικής δομής)
- `continue` (εντολή παράλειψης της τρέχουσας επανάληψης)
- `return` (εντολή επιστροφής από μία συνάρτηση)
- `goto` (εντολή μετάβασης σε κάποιο σημείο του κώδικα)

➤ Αριθμητικοί τελεστές

- `=` (τελεστής εκχώρησης)
- `+` (τελεστής πρόσθεσης)
- `-` (τελεστής αφαίρεσης)
- `*` (τελεστής πολλαπλασιασμού)
- `/` (τελεστής διαίρεσης)
- `%` (τελεστής υπόλοιπου ακέραιας διαίρεσης)

➤ Λογικοί τελεστές

- `&&` (λογική σύζευξη)
- `||` (λογική διάζευξη)
- `!` (λογική άρνηση)

➤ Δυαδικοί τελεστές

- & (δυαδική σύζευξη)
- | (δυαδική διάζευξη)
- ^ (δυαδική αποκλειστική διάζευξη)
- ~ (δυαδική άρνηση)
- << (δυαδική αριστερή ολίσθηση)
- >> (δυαδική δεξιά ολίσθηση)

➤ Τελεστές αύξησης και μείωσης

- ++ (αύξηση κατά μία ακέραιη μονάδα)
- -- (μείωση κατά μία ακέραιη μονάδα)

➤ Σύνθετοι τελεστές

- +=, -=, *=, /=, %= (σύνθετοι αριθμητικοί τελεστές)
- &=, |=, ^=, ~=, <<=, >>= (σύνθετοι δυαδικοί τελεστές)

➤ Τελεστές σύγκρισης

- == (ισότητα)
- != (ανισότητα)
- < (μικρότερο)
- > (μεγαλύτερο)
- <= (μικρότερο ή ίσο)
- >= (μεγαλύτερο ή ίσο)

➤ Τελεστές δεικτών

- * (τελεστής απόκτησης περιεχομένου)
- & (τελεστής απόκτησης διεύθυνσης)

➤ Σταθερές

- HIGH (τιμή υψηλής στάθμης για μία επαφή εισόδου ή εξόδου)
- LOW (τιμή χαμηλής στάθμης για μία επαφή εισόδου ή εξόδου)
- false (λογικό επίπεδο ψεύδους σε μία συνθήκη)
- true (λογικό επίπεδο αλήθειας σε μία συνθήκη)
- INPUT (χρησιμοποιείται για τον ορισμό μίας επαφής ως είσοδο)
- OUTPUT (χρησιμοποιείται για τον ορισμό μίας επαφής ως έξοδο)
- A0, ..., A5 (συμβολοσταθερές για τις αναλογικές επαφές εισόδου)

➤ Τύποι δεδομένων

- boolean (λογική δυαδική τιμή)
 - char (προσημασμένος χαρακτήρας 8 ψηφίων)
 - unsigned char (μη προσημασμένος χαρακτήρας 8 ψηφίων)
 - byte (μη προσημασμένος χαρακτήρας 8 ψηφίων)
 - int (προσημασμένος ακέραιος αριθμός 16 ψηφίων)
 - unsigned int (μη προσημασμένος ακέραιος αριθμός 16 ψηφίων)
 - word (μη προσημασμένος ακέραιος αριθμός 16 ψηφίων)
 - long (προσημασμένος ακέραιος αριθμός 32 ψηφίων)
 - unsigned long (μη προσημασμένος ακέραιος αριθμός 32 ψηφίων)
 - float, double (αριθμός κινητής υποδιαστολής απλής ακρίβειας)
 - String (αντικείμενο αλφαριθμητικού με χρήσιμες μεθόδους)
- Ως αλφαριθμητικό μπορεί να θεωρηθεί και ο πίνακας χαρακτήρων

➤ Συναρτήσεις μετατροπής τύπων

- char(), byte()
- int(), word(), long()
- float(), double()

➤ Συναρτήσεις εισόδου και εξόδου

- pinMode() (ορίζει μια επαφή ως είσοδο ή έξοδο)

➤ Συναρτήσεις ψηφιακής εισόδου και εξόδου

- digitalWrite() (γράφει σε μία ψηφιακή επαφή εξόδου)
- digitalRead() (διαβάζει από μία ψηφιακή επαφή εισόδου)

➤ Συναρτήσεις αναλογικής εισόδου και εξόδου

- analogReference() (ορίζει την τάση αναλογικής αναφοράς)
- analogWrite() (γράφει PWM σήματα σε μία επαφή εξόδου)
- analogRead() (διαβάζει από μία αναλογική επαφή εισόδου)

➤ Προηγμένες συναρτήσεις εισόδου και εξόδου

- tone() (παράγει ένα τετραγωνικό σήμα ορισμένης συχνότητας)
- noTone() (διακόπτει την παραγωγή τετραγωνικών σημάτων)
- shiftOut() (ολισθαίνει τα ψηφία μιας τιμής σε μία επαφή εξόδου)
- pulseIn() (επιστρέφει την διάρκεια σε μια ενός παλμού HIGH ή LOW)

➤ Συναρτήσεις χρόνου

- millis() (διάρκεια εκτέλεσης του προγράμματος σε ms)
- micros() (διάρκεια εκτέλεσης του προγράμματος σε μs)
- delay() (παύση προγράμματος - η διάρκεια δίδεται σε ms)
- delayMicroseconds() (παύση προγράμματος - η διάρκεια δίδεται σε μs)

➤ Μαθηματικές και Τριγωνομετρικές συναρτήσεις

- max() (βρίσκει τον μεγαλύτερο ανάμεσα σε δύο αριθμούς)
- min() (βρίσκει τον μικρότερο ανάμεσα σε δύο αριθμούς)
- abs() (επιστρέφει την απόλυτη τιμή ενός αριθμού)
- constrain() (ελέγχει για υπερχείλιση ή υποχείλιση ορίων)
- map() (πραγματοποιεί γραμμικό μετασχηματισμό ορίων)
- pow() (επιστρέφει το αποτέλεσμα μίας δύναμης)
- sqrt() (επιστρέφει την ρίζα ενός αριθμού)
- sin() (υπολογίζει το ημίτονο ενός αριθμού)
- cos() (υπολογίζει το συνημίτονο ενός αριθμού)
- tan() (υπολογίζει την εφαπτομένη ενός αριθμού)

➤ **Συναρτήσεις γεννήτριας ψευδοτυχαίων αριθμών**

- random() (δίδεται ένας νέος αριθμός από την γεννήτρια)
- randomSeed() (θέτει τον σπόρο της γεννήτριας παραγωγής)

➤ **Συναρτήσεις επεξεργασίας δυαδικών αριθμών**

- lowByte() (επιστρέφει το δεξιότερο byte μίας μεταβλητής)
- highByte() (επιστρέφει το αριστερότερο byte μίας μεταβλητής)
- bitRead() (διαβάζει ένα συγκεκριμένο ψηφίο μίας μεταβλητής)
- bitWrite() (γράφει σε ένα συγκεκριμένο ψηφίο μίας μεταβλητής)
- bitSet() (γράφει την τιμή 1 σε κάποιο ψηφίο μίας μεταβλητής)
- bitClear() (γράφει την τιμή 0 σε κάποιο ψηφίο μίας μεταβλητής)
- bit() (υπολογίζει μία συγκεκριμένη δύναμη με βάση το 2)

➤ **Συναρτήσεις χρήσης ρουτινών εξυπηρέτησης διακοπών**

- attachInterrupt() (ενεργοποιεί μία ρουτίνα εξυπηρέτησης διακοπής)
- detachInterrupt() (απενεργοποιεί μία ρουτίνα εξυπηρέτησης διακοπής)

➤ **Συναρτήσεις ενεργοποίησης και απενεργοποίησης διακοπών**

- interrupts() (ενεργοποιεί τα σήματα διακοπής)
- noInterrupts() (απενεργοποιεί τα σήματα διακοπής)

➤ **Υποστήριξη σειριακής επικοινωνίας**

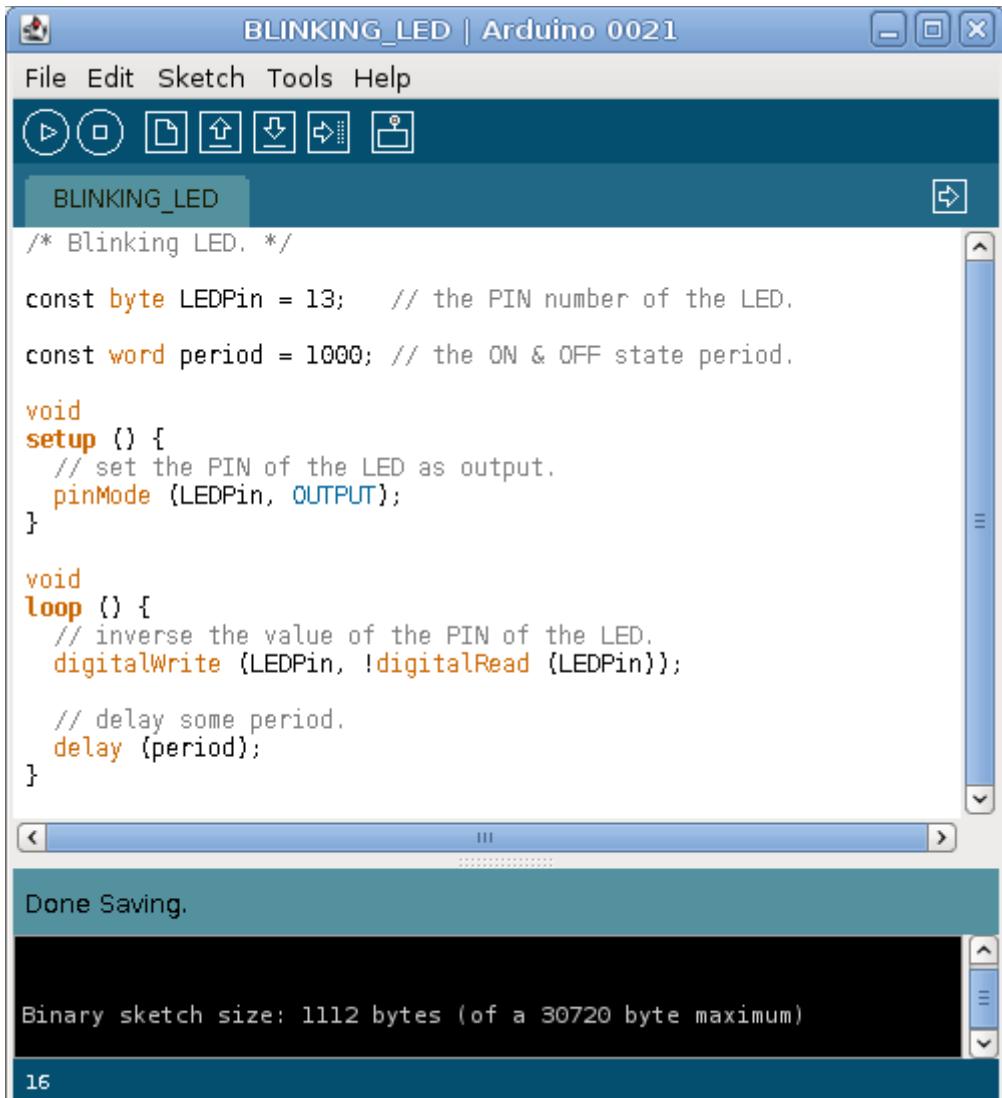
- Serial (αντικείμενο σειριακής επικοινωνίας με χρήσιμες μεθόδους)

3.3 - Ολοκληρωμένο Περιβάλλον Ανάπτυξης

Την πλατφόρμα Arduino συνοδεύει και ένα ανοιχτού λογισμικού ολοκληρωμένο περιβάλλον ανάπτυξης (βασισμένο στο περιβάλλον της Processing). Αυτό το περιβάλλον μπορεί να χρησιμοποιηθεί για τον αποδοτικό προγραμματισμό της πλατφόρμας Arduino.

Πιο συγκεκριμένα, το ολοκληρωμένο περιβάλλον ανάπτυξης μπορεί να χρησιμοποιηθεί για την συγγραφή και την μεταγλώττιση προγραμμάτων Arduino, τον προγραμματισμό μίας πλατφόρμας Arduino, την σειριακή επικοινωνία με μία πλατφόρμα Arduino και την χρήση ή ενσωμάτωση έτοιμων βιβλιοθηκών μέσα σε ένα πρόγραμμα Arduino.

Επίσης, το ολοκληρωμένο περιβάλλον ανάπτυξης έχει αναπτυχθεί με την γλώσσα προγραμματισμού JAVA. Το γεγονός αυτό επιτρέπει την εκτέλεση του περιβάλλοντος στα περισσότερα λειτουργικά συστήματα χωρίς προβλήματα συμβατότητας.



```
BLINKING_LED | Arduino 0021
File Edit Sketch Tools Help
(▶) (○) (D) (U) (D) (F) (S)
BLINKING_LED
/*
 * Blinking LED.
 */
const byte LEDPin = 13; // the PIN number of the LED.
const word period = 1000; // the ON & OFF state period.

void
setup () {
    // set the PIN of the LED as output.
    pinMode (LEDPin, OUTPUT);
}

void
loop () {
    // inverse the value of the PIN of the LED.
    digitalWrite (LEDPin, !digitalRead (LEDPin));

    // delay some period.
    delay (period);
}

Done Saving.

Binary sketch size: 1112 bytes (of a 30720 byte maximum)
```

Εικόνα 06 : Το ολοκληρωμένο περιβάλλον ανάπτυξης Arduino

Αξίζει να αναφερθεί πως το ολοκληρωμένο περιβάλλον ανάπτυξης αξιοποιεί μια σειρά από εργαλεία και βιβλιοθήκες για την επίτευξη των σκοπών του. Πιο συγκεκριμένα, χρησιμοποιεί τους μεταγλωττιστές avr-gcc ή avr-g++ για την μεταγλώττιση των προγραμμάτων Arduino, την βιβλιοθήκη avr-libc για την χρήση των συναρτήσεων χειρισμού της πλατφόρμας Arduino, το εργαλείο avrdude για την τοποθέτηση του προγράμματος στην πλατφόρμα Arduino και το GNU toolchain γενικότερα.

Το ολοκληρωμένο περιβάλλον ανάπτυξης αποτελείται από έναν συντάκτη κειμένου για την συγγραφή κώδικα, μία περιοχή μηνυμάτων, μία κονσόλα κειμένου, μία απλή εργαλειοθήκη με τις πιο βασικές λειτουργίες, ένα τερματικό για την σειριακή επικοινωνία με την πλατφόρμα Arduino και ένα βασικό μενού επιλογών με διάφορες χρήσιμες επιλογές.



Εικόνα 07 : Σειριακή επικοινωνία με μία πλατφόρμα Arduino

Ο συντάκτης κειμένου του ολοκληρωμένου περιβάλλοντος ανάπτυξης είναι εφοπλισμένος με βασικές λειτουργίες επεξεργασίας κειμένου. Επίσης, η περιοχή μηνυμάτων χρησιμοποιείται για την εμφάνιση βοηθητικών μηνυμάτων. Παράλληλα, η κονσόλα κειμένου χρησιμοποιείται για την εμφάνιση μηνυμάτων και σφαλμάτων που εξάγονται από τα διάφορα εργαλεία που χρησιμοποιούνται από το περιβάλλον.

Πολλές φορές, είναι πιθανόν κατά την διάρκεια του προγραμματισμού μιας πλατφόρμας Arduino ή κατά την συγγραφή ενός προγράμματος Arduino να απορήσουμε σκεπτόμενοι : “Πως είναι δυνατόν ένα πρόγραμμα το οποίο αποτελείται μόνο από τις συναρτήσεις setup() και loop() να αποτελεί πρότυπο πηγαίο κώδικα C ή C++;”.

Στην πραγματικότητα, το ολοκληρωμένο περιβάλλον ανάπτυξης είναι αυτό το οποίο κατά την μεταγλώττιση αποθηκεύει σε ένα προσωρινό αρχείο τον πηγαίο κώδικα του προγράμματος που έχουμε συγγράψει και στην συνέχεια προσθέτει σε αυτό όλα όσα είναι απαραίτητα (πρωτότυπα συναρτήσεων, υλοποίηση συνάρτησης main(), ενσωμάτωση επικεφαλίδων) ώστε να είναι δυνατή η μεταγλώττιση του.

Επίσης, μετά την μεταγλώττιση του πηγαίου κώδικα προκύπτει ο ισοδύναμος αντικειμενικός κώδικας του προγράμματος. Ο αντικειμενικός κώδικας διασυνδέεται με τις βιβλιοθήκες του Arduino όπου τελικά προκύπτει ένα αρχείο με μορφότυπο Intel Hex. Το αρχείο αυτό περιέχει όλα τα δεδομένα που χρειάζεται να τοποθετηθούν στην μνήμη Flash της πλατφόρμας Arduino ώστε να ολοκληρωθεί ο προγραμματισμός της.

Η μεταφορά των δεδομένων του τελικού αρχείου στην πλατφόρμα Arduino γίνεται μέσω USB. Για την επίτευξη αυτής της διαδικασίας χρησιμοποιείται από το ολοκληρωμένο περιβάλλον ανάπτυξης το εργαλείο avrdude και ο φορτωτής εκκίνησης της πλατφόρμας Arduino.

Η βασική εργαλειοθήκη περιέχει τις παρακάτω λειτουργίες :

- ▶ Έναρξη της διαδικασίας μεταγλώττισης ενός προγράμματος Arduino
- ▣ Απενεργοποίηση κάθε ενέργειας που συντελείται στο περιβάλλον
- ☒ Δημιουργία ενός νέου προγράμματος Arduino
- ⬆ Άνοιγμα ενός προγράμματος Arduino
- ⬇ Αποθήκευση του τρέχοντος προγράμματος Arduino
- ▷ Άμεσος προγραμματισμός της πλατφόρμας Arduino
- ⌚ Χρήση τερματικού σειριακής επικοινωνίας με την πλατφόρμα Arduino

Εκτός από τις παραπάνω στοιχειώδεις ενέργειες το ολοκληρωμένο περιβάλλον ανάπτυξης διαθέτει και άλλες σημαντικές προαιρετικές λειτουργίες. Μερικές από τις πιο ενδιαφέρουσες είναι οι παρακάτω :

➤ Μενού “Επεξεργασία”

- Λειτουργία “Αντιγραφή ως HTML”

Χρησιμοποιείται όταν επιθυμούμε να δημοσιεύσουμε στο Διαδίκτυο τον πηγαίο κώδικα του προγράμματος Arduino που έχουμε συγγράψει. Με την εκτέλεση αυτής της λειτουργίας μετατρέπεται ο κώδικας του προγράμματος σε κατάλληλο κώδικα XHTML. Ο κώδικας XHTML αντιγράφεται στην συνέχεια στην προσωρινή μνήμη και μπορεί να επικολληθεί σε οποιοδήποτε έγγραφο XHTML.

- **Λειτουργία “Αντιγραφή για το FORUM”**

Χρησιμοποιείται όταν επιθυμούμε να δημοσιεύσουμε στο FORUM της διαδικτυακής κοινότητας του Arduino το πρόγραμμα που έχουμε συγγράψει. Με την εκτέλεση αυτής της λειτουργίας μετατρέπεται ο πηγαίος κώδικας του προγράμματος σε κατάλληλο κώδικα και αντιγράφεται στην προσωρινή μνήμη ώστε να επικολληθεί άμεσα σε κάποια σελίδα του FORUM.

➤ **Μενού “Σκίτσο”**

- **Λειτουργία “Ενσωμάτωση Βιβλιοθήκης”**

Με αυτή την λειτουργία μπορεί να γίνει η επιλογή από μία εκτεταμένη λίστα μιας συγκεκριμένης βιβλιοθήκης ώστε να ενσωματωθεί στο πρόγραμμα Arduino.

➤ **Μενού “Εργαλεία”**

- **Λειτουργία “Αυτόματη Μορφοποίηση”**

Η λειτουργία αυτή επιτελεί μία αυτόματη διαδικασία μορφοποίησης του πηγαίου κώδικα του προγράμματος Arduino ώστε να είναι πιο αναγνώσιμος και σωστά δομημένος.

- **Λειτουργία “Πλακέτα”**

Η λειτουργία αυτή χρησιμοποιείται για την επιλογή της κατάλληλης έκδοσης της πλατφόρμας Arduino που θέλουμε να χρησιμοποιήσουμε. Επίσης, το ολοκληρωμένο περιβάλλον ανάπτυξης διαθέτει μία πλήρη λίστα με όλες τις πλατφόρμες Arduino που έχουν κατασκευαστεί και κυκλοφορούν στο εμπόριο.

- **Λειτουργία “Σειριακή Θύρα”**

Η λειτουργία αυτή χρησιμοποιείται για την επιλογή μιας σειριακής θύρας στην οποία διασυνδέεται κάποια πλατφόρμα Arduino ώστε να είναι δυνατή η διαδικασία του προγραμματισμού της και γενικότερα η σειριακή επικοινωνία με αυτήν μέσω του Η/Υ.

- **Λειτουργία “Τοποθέτηση Φορτωτή Εκκίνησης”**

Η λειτουργία αυτή εκτελείται σπανίως και χρησιμοποιείται για την μεταφορά του κατάλληλου φορτωτή εκκίνησης στην μνήμη Flash ενός μικροελεγκτή AVR μιας πλατφόρμας Arduino.

3.4 - Βιβλιοθήκες

Η πλατφόρμα Arduino συνοδεύεται μαζί με μερικές χρήσιμες βιβλιοθήκες όπου κάθε μία από αυτές παρέχει κάποια υπηρεσία είτε σε επίπεδο λογισμικού είτε σε επίπεδο υλικού. Επίσης, στο Διαδίκτυο μπορούν να βρεθούν και αρκετές βιβλιοθήκες όπου αναπτύσσονται κυρίως από προγραμματιστές που στηρίζουν το Arduino. Παρακάτω, ακολουθούν μερικές βασικές βιβλιοθήκες :

➤ **PS2Keyboard**

Ανάγνωση δεδομένων από πληκτρολόγια τύπου PS/2

➤ **EEPROM**

Διαχείριση της μνήμης EEPROM μιας πλατφόρμας Arduino

➤ **Webduino**

Επεκτάσιμος διαδικτυακός εξυπηρετητής ιστοσελίδων

➤ **Ethernet**

Ενσύρματη δικτυακή σύνδεση με την κάρτα επέκτασης Ethernet

➤ **XBee**

Ασύρματη δικτυακή σύνδεση με την κάρτα επέκτασης XBee

➤ **LiquidCrystal**

Διαχείριση τυπικών οθονών υγρών κρυστάλλων

➤ **SD**

Διαχείριση καρτών μνήμης SD

➤ **Servo / Stepper**

Διαχείριση και έλεγχος σερβοκινητήρων / βηματικών κινητήρων

➤ **Finite State Machine**

Υλοποίηση μηχανών πεπερασμένων καταστάσεων

➤ **NewSoftSerial**

Υλοποίηση σειριακής επικοινωνίας σε επίπεδο λογισμικού

➤ **Flash**

Αποθήκευση δεδομένων στην μνήμη Flash μιας πλατφόρμας Arduino

3.5 - Παραδείγματα

Παρακάτω, παραθέτονται δύο προγράμματα Arduino με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) ώστε με την ανάγνωση τους να εξοικειωθείτε τόσο με την γλώσσα προγραμματισμού όσο και με τη χρήση των θεμελιωδών συναρτήσεων των βιβλιοθηκών της πλατφόρμας Arduino.

Ακολουθεί ένα πρόγραμμα Arduino το οποίο εκτελεί επανειλημμένα το αναλογικό αναβόσβησμα μίας τυπικής διόδου εκπομπής φωτός η οποία είναι συνδεδεμένη στην ψηφιακή επαφή 9 μιας τυπικής πλατφόρμας Arduino :

```
/*
 * Fading LED.
 *
 * Copyright (C) 2011 Efstathios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

const byte ledPin = 9;      // PIN number (PWM) for the LED.
const word delayTime = 100; // LED delay time (milliseconds).

// startup point entry (runs once).
void
setup () {
    pinMode (ledPin, OUTPUT); // set fading LED as an output.
}

// loop the main sketch.
void
loop () {
    // fade in from min to max.
    for (byte fadeValue = 0; fadeValue <= 255; fadeValue +=5) {
        analogWrite (ledPin, fadeValue);
        delay (delayTime);
    }

    // fade out from max to min.
    for (byte fadeValue = 255; fadeValue >= 0; fadeValue -=5) {
        analogWrite (ledPin, fadeValue);
        delay (delayTime);
    }
}
```

Πρόγραμμα 1 : Αναλογικό αναβόσβησμα μίας διόδου εκπομπής φωτός

Ακολουθεί ένα πρόγραμμα Arduino το οποίο πραγματοποιεί μόνο μία φορά την εκτύπωση (στην σειριακή) ενός μικρού αλφαριθμητικού που έχει αντιστραφεί με την βοήθεια μιας δυναμικής στοίβας :

```
/*
 * Reverse a string by using a generic, dynamic stack data structure.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

// include stack library header.
#include <StackList.h>

const String msg = "Happy Hacking!"; // declare a string message.
StackList <char> stack; // create a stack of characters.

// startup point entry (runs once).
void
setup () {
    Serial.begin (9600); // start serial communication.
    stack.setPrinter (Serial); // set the printer of the stack.

    // print the message before reversing.
    Serial.println ("Normal String: " + msg);

    // push all the message's characters to the stack.
    for (int i = 0; i < msg.length (); i++)
        stack.push (msg.charAt (i));

    // print the message after reversing.
    Serial.print ("Reversed String: ");

    // pop all the message's characters from the stack.
    while (!stack.isEmpty ())
        Serial.print (stack.pop ());

    // print end of line character.
    Serial.println ();
}

// loop the main sketch (empty).
void loop () {}
```

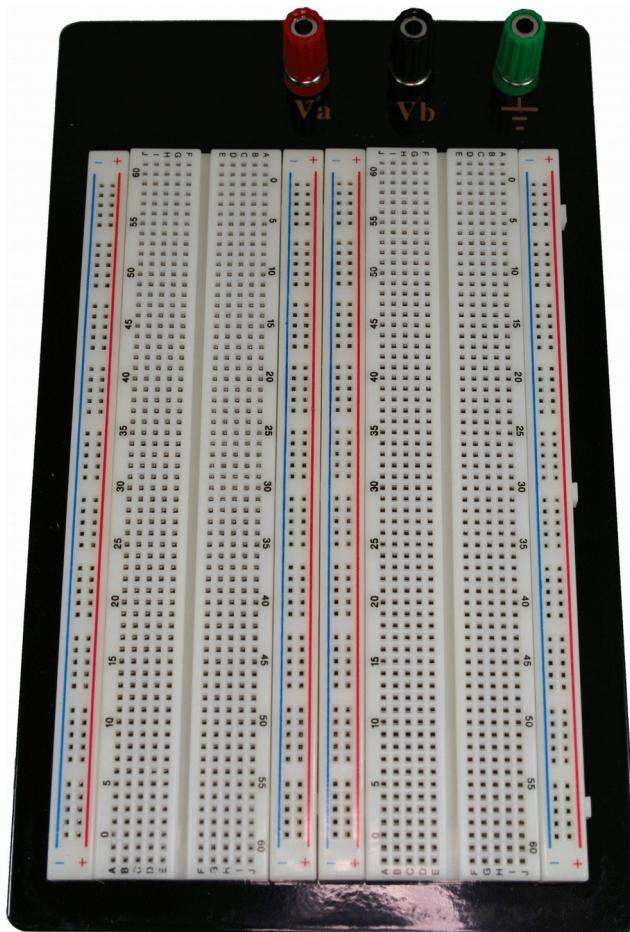
Πρόγραμμα 2 : Αντιστροφή αλφαριθμητικού με χρήση στοίβας

Κεφάλαιο 4 - Επιμέρους Συστατικά Συστήματος

4.01 - Ράστερ & Καλώδια

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκαν απαραίτητα για την ανάπτυξη τόσο του κόμβου ταυτοποίησης όσο και του κόμβου ελέγχου μερικά τυπικά ράστερ καθώς και ειδικά καλώδια μεγέθους 22 AWG.

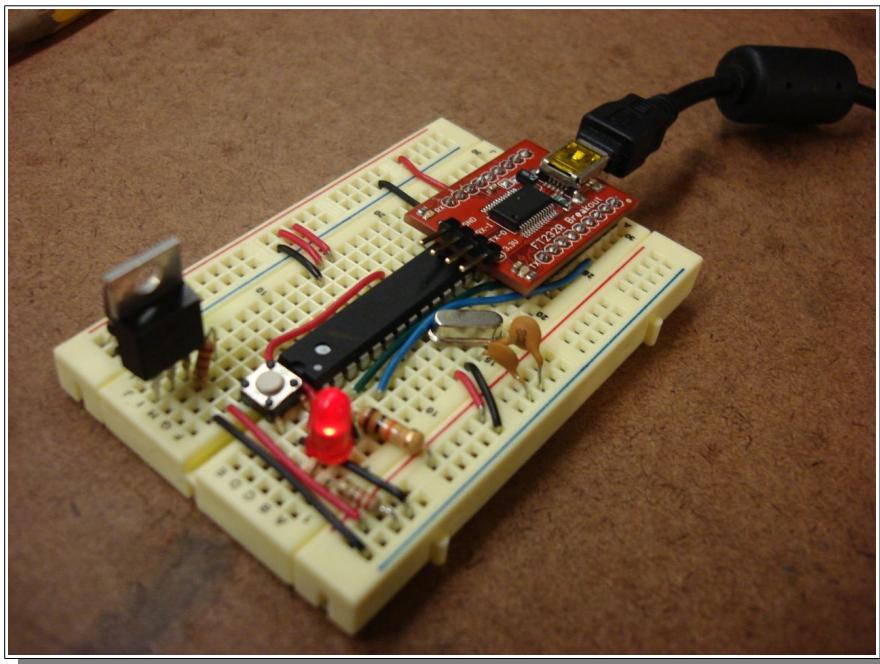
Το ράστερ αποτελεί βασικό και αναγκαίο εργαλείο για την προσωρινή κατασκευή τόσο ψηφιακών όσο και αναλογικών πρωτότυπων ηλεκτρονικών κυκλωμάτων. Πιο συγκεκριμένα, το ράστερ είναι μια πλακέτα με πλαστικό εξωτερικό περίβλημα και διαθέτει αρκετά σημεία επαφής τα οποία απέχουν μεταξύ τους συνήθως 2.54mm. Επίσης, σε αυτά τα σημεία επαφής μπορούν να τερματιστούν τόσο απλά καλώδια όσο και διάφορα ηλεκτρονικά εξαρτήματα όπως για παράδειγμα αντιστάτες, πυκνωτές, ολοκληρωμένα κυκλώματα σε συσκευασία τύπου DIP, δίοδοι εκπομπής φωτός, κ.α.



Εικόνα 08 : Τυπικό ράστερ για την κατασκευή ηλεκτρονικών κυκλωμάτων

Αξίζει να αναφερθεί πως τα διάφορα σημεία επαφής που υπάρχουν σε ένα ράστερ είναι οργανωμένα οριζόντια και κάθετα σε λωρίδες. Αυτές οι λωρίδες διακρίνονται είτε σε λωρίδες τροφοδοσίας, είτε σε λωρίδες τερματισμού. Επίσης, κάθε ράστερ διαθέτει τουλάχιστον μία ή περισσότερες λωρίδες τερματισμού καθώς και μία ή περισσότερες λωρίδες τροφοδοσίας.

Ο σκοπός των λωρίδων τερματισμού σε ένα ράστερ είναι η σύνδεση των διαφόρων ηλεκτρονικών εξαρτημάτων. Τα σημεία επαφής σε μία λωρίδα τερματισμού είναι οργανωμένα και διατεταγμένα σε γραμμές και στήλες. Οι γραμμές σημαίνονται με αριθμούς και το πλήθος τους εξαρτάται από τις διαστάσεις του ράστερ. Επίσης, οι στήλες σημαίνονται με τα γράμματα A-J του Αγγλικού αλφαριθμητού και είναι πάντα 10 σε πλήθος. Έτσι, κάθε σημείο επαφής έχει απόλυτες και μοναδικές συντεταγμένες αναφοράς.

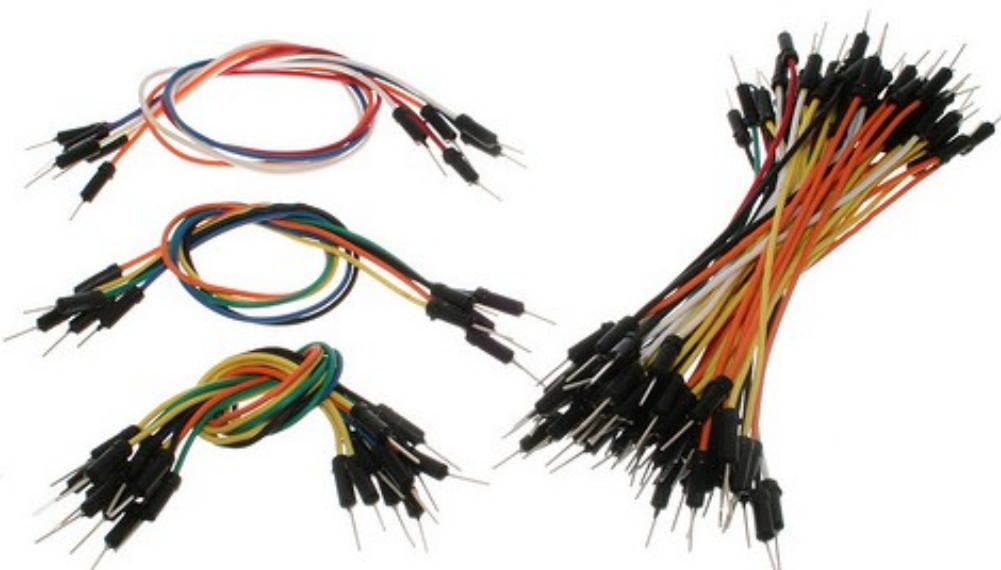


Εικόνα 09 : Δημιουργία μιας τυπικής πλατφόρμας Arduino στο ράστερ

Οι πέντε πρώτες στήλες A-E χωρίζονται από τις πέντε επόμενες στήλες F-J από μία φαρδιά εσοχή χωρίς σημεία επαφής, η οποία ονομάζεται κεντρική γραμμή. Η βασική ιδιότητα της λωρίδας τερματισμού είναι ότι για κάθε γραμμή της τα σημεία επαφής που βρίσκονται αριστερά ή δεξιά της κεντρικής γραμμής είναι βραχυκυκλωμένα μεταξύ τους. Έτσι, λόγου χάρη, αν ένας ακροδέκτης κάποιου εξαρτήματος συνδεθεί στο σημείο επαφής E4 της λωρίδας, τότε ουσιαστικά είναι συνδεδεμένος και με τα σημεία επαφής A4, B4, C4 και D4 αυτής.

Οι λωρίδες τροφοδοσίας αποτελούνται μόνο από δύο παράλληλες στήλες με σημεία επαφής. Η μια στήλη σημαίνεται συνήθως με κόκκινο χρώμα και το σύμβολο “+” ή το γράμμα “V”, ενώ η άλλη στήλη σημαίνεται με μαύρο ή μπλε χρώμα και το σύμβολο “-” ή το γράμμα “G”. Η βασική ιδιότητα αυτής της λωρίδας είναι ότι όλα τα σημεία επαφής καθεμιάς από τις δύο γραμμές της είναι βραχυκυκλωμένα. Ο ρόλος μιας λωρίδας τροφοδοσίας, λοιπόν, είναι να συνδεθεί σε αυτή μια τροφοδοσία και στη συνέχεια να τροφοδοτηθούν από αυτή τα διάφορα εξαρτήματα που είναι συνδεδεμένα στη λωρίδα τερματισμού.

Αφού τοποθετηθούν τα εξαρτήματα στη λωρίδα τερματισμού, η σύνδεση μεταξύ τους και με τη λωρίδα τροφοδοσίας μπορεί να γίνει με απλά καλώδια, οι γυμνές άκρες των οποίων μπορούν και αυτές να τερματιστούν στα σημεία επαφής του ράστερ. Για ευκολία μπορούμε να προμηθευτούμε έτοιμα καλώδια μεγέθους 22 AWG, διαφορετικών μηκών και χρωμάτων που διαθέτουν στα άκρα τους εκλεπτυσμένους ακροδέκτες για την εύκολη σύνδεση τους.



Εικόνα 10 : Πολύχρωμα καλώδια μεγέθους 22 AWG

Τα ράστερ μπορούν να χρησιμοποιηθούν για την κατασκευή προσωρινών και πρωτότυπων ηλεκτρονικών κυκλωμάτων αφού επιτρέπουν την γρήγορη σύνδεση ή αποσύνδεση εξαρτημάτων καθώς και αλλαγές στη συνδεσμολογία των κυκλωμάτων χωρίς κολλήσεις. Έτσι, εάν κάποιο πρωτότυπο κύκλωμα ολοκληρωθεί και δοκιμαστεί επιτυχώς στο ράστερ μπορεί στην συνέχεια να αποτυπωθεί σε μια πλακέτα και να αποκτήσει μόνιμη μορφή.

4.02 - Κάρτα Επέκτασης / Κάρτα Μνήμης microSD

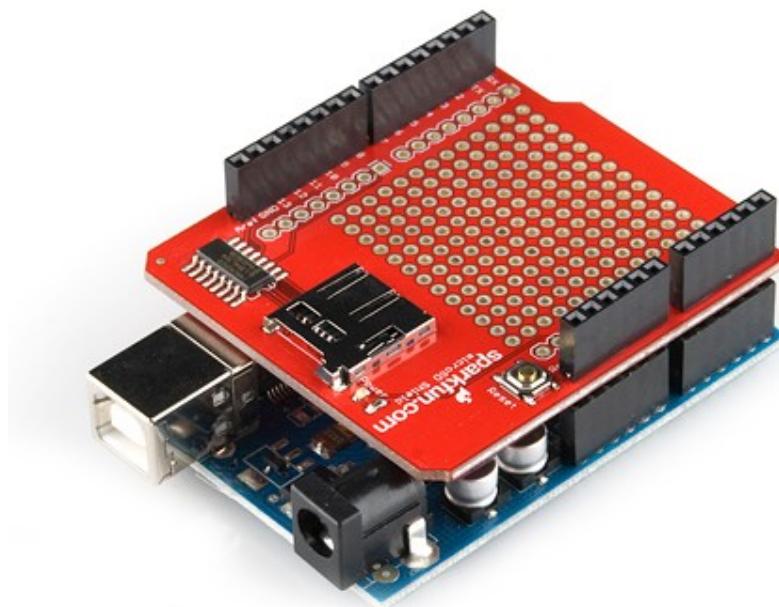
Οι πλατφόρμες Arduino με την περιορισμένη σε χωρητικότητα μνήμη EEPROM που διαθέτουν δεν μπορούν να χρησιμοποιηθούν σε εφαρμογές στις οποίες είναι επιθυμητή η μόνιμη αποθήκευση μεγάλου όγκου δεδομένων. Έτσι λοιπόν, εάν είναι απαραίτητο να αναπτυχθεί ένα αυτόνομο ενσωματωμένο σύστημα όπου θα πρέπει να διαθέτει μεγάλη αποθηκευτική ικανότητα μπορεί να χρησιμοποιηθεί κάποια εξωτερική κάρτα επέκτασης για την υποστήριξη καρτών μνήμης (π.χ. SD, microSD) μεγάλης χωρητικότητας.



Εικόνα 11 : Μία κάρτα μνήμης microSD χωρητικότητας 2GB

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση κάποιας εξωτερικής κάρτας επέκτασης και κάποιας τυπικής κάρτας μνήμης για την αποθήκευση των μοναδικών σειριακών αριθμών των παθητικών ετικετών ραδιοσυχνικής ταυτοποίησης (RFID). Συγκεκριμένα, το υλικό που χρησιμοποιήθηκε ήταν μια κάρτα επέκτασης microSD (SparkFun) και μία κάρτα μνήμης (2GB) microSD.

Η σειριακή επικοινωνία με την κάρτα μνήμης microSD πραγματοποιείται μέσω της κάρτας επέκτασης microSD καθώς και της διεπαφής SPI την οποία υποστηρίζει η πλατφόρμα Arduino μέσω των ειδικών επαφών εισόδου και εξόδου που διαθέτει. Επίσης, το σύστημα αρχείων που χρησιμοποιείται για την κάρτα μνήμης microSD είναι το FAT32. Συγκεκριμένα, η υποστήριξη του συστήματος αρχείων FAT32 παρέχεται μέσω της βιβλιοθήκης SdFat (η οποία αποτελεί ελεύθερο λογισμικό).



Εικόνα 12 : Η κάρτα επέκτασης (κόκκινη) microSD (SparkFun)

Αξίζει να αναφερθεί πως η κάρτα επέκτασης microSD διαθέτει μία σειρά από διάφορα μηχανικά και ηλεκτρονικά εξαρτήματα όπως : μία υποδοχή για κάρτες μνήμης microSD, ένα ενσωματωμένο κύκλωμα για την μετατροπή της τάσης από 5V σε 3.3V για την ορθή λειτουργία των καρτών μνήμης microSD, μία δίοδο εκπομπής φωτός κόκκινου χρώματος (ένδειξη τροφοδοσίας) καθώς και ένα μικρό κουμπί για την επαναρχικοποίηση του μικροελεγκτή AVR της πλατφόρμας Arduino. Επιπλέον, στο παράρτημα 1 θα βρείτε το σχεδιάγραμμα του κυκλώματος της κάρτας επέκτασης microSD.

4.03 - Ανιχνευτής Υπερύθρων & Τηλεχειριστήριο

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση κάποιου ανιχνευτή υπερύθρων (μαζί με ένα τυπικό τηλεχειριστήριο) για την άμεση συσχέτιση των διαφόρων παθητικών ετικετών ραδιοσυχνικής ταυτοποίησης (RFID) με τις απομακρυσμένες συσκευές του κόμβου ελέγχου του συστήματος.

Ένας ανιχνευτής υπερύθρων μπορεί να χρησιμοποιηθεί σε τηλεοράσεις, σε συσκευές DVD, σε στερεοφωνικά ή ακόμα και σε ειδικά ενσωματωμένα συστήματα για την ανίχνευση των διαφόρων υπέρυθρων σημάτων που πιθανόν να αποστέλλονται από τα τηλεχειριστήρια των αναφερθέντων συσκευών. Πιο συγκεκριμένα, ένας ανιχνευτής υπερύθρων συνήθως αποτελείται από ένα ειδικό μικροτσίπ καθώς και από ένα φωτοκύτταρο το οποίο είναι συντονισμένο κατάλληλα ώστε να είναι ικανό να αντιλαμβάνεται μόνο το υπέρυθρο φως.



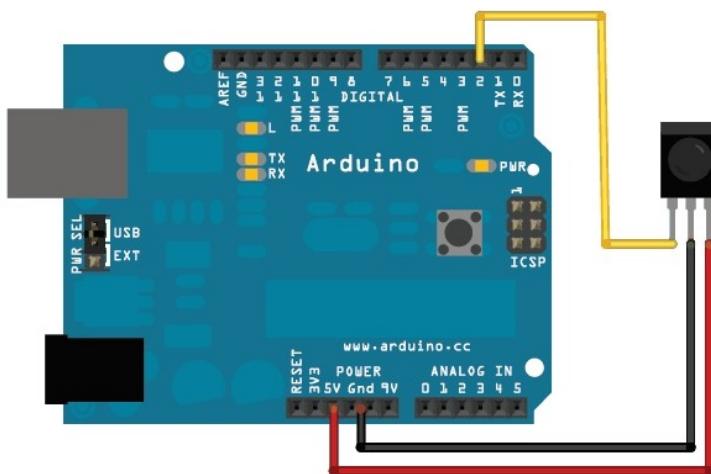
Εικόνα 13 : Ο ανιχνευτής υπερύθρων PNA4602 (Panasonic)

Αξίζει να αναφερθεί πως η παραγωγή των υπέρυθρων σημάτων από ένα τηλεχειριστήριο συνήθως πραγματοποιείται από μία ειδική δίοδο εκπομπής υπέρυθρου φωτός που βρίσκεται σε αυτό. Το υπέρυθρο φως είναι γνωστό πως δεν είναι αντιληπτό από τον ανθρώπινο οφθαλμό. Παρόλα αυτά, μπορούμε ικανοποιητικά και χωρίς ιδιαίτερα προβλήματα με δευτερεύουσες τεχνικές να εξετάσουμε την ορθότητα της λειτουργίας ενός ανιχνευτή υπερύθρων ή μίας διόδου εκπομπής υπέρυθρου φωτός.

Επίσης, γνωρίζουμε από την επιστήμη της Φυσικής πως κάθε σώμα που θερμαίνεται εκπέμπει ακτινοβολία κάποιας μορφής. Έτσι και η Γη μας όταν θερμαίνεται από την ακτινοβολία του Ήλιου, εκπέμπει υπέρυθρη ακτινοβολία. Ακριβώς για αυτόν τον λόγο οι ανιχνευτές υπερύθρων δεν κατασκευάζονται ώστε να αντιλαμβάνονται κάθε υπέρυθρο σήμα που μπορεί να παρουσιαστεί εντός της εμβέλειας που καλύπτουν διότι σε μία τέτοια περίπτωση δεν θα ήταν δυνατή από τους ανιχνευτές υπερύθρων η διάκριση μεταξύ των υπέρυθρων σημάτων που εκπέμπουν τα σώματα του περιβάλλοντος και των υπέρυθρων σημάτων που εκπέμπει κάποια δίοδο εκπομπής υπέρυθρου φωτός.

Στην πραγματικότητα, οι ανιχνευτές υπερύθρου φωτός διαθέτουν έναν ενσωματωμένο αποδιαμορφωτή ο οποίος λαμβάνει υπόψη μόνο κατάλληλα διαμορφωμένα υπέρυθρα σήματα. Αντίστοιχα, μία τυπική δίοδος εκπομπής υπέρυθρου φωτός εκπέμπει τα διάφορα υπέρυθρα σήματα διαμορφωμένα με τέτοιο τρόπο ώστε να είναι άμεσα αντιληπτά από τους ανιχνευτές.

Συγκεκριμένα, για τον κόμβο ταυτοποίησης που αναφέρθηκε παραπάνω χρησιμοποιήθηκε ο ανιχνευτής υπερύθρων PNA4602 (Panasonic). Επίσης, στο παράρτημα 1 θα βρείτε το σχεδιάγραμμα του κυκλώματος του ανιχνευτή. Αξίζει να αναφερθεί πως ο ανιχνευτής διαθέτει τρεις ακροδέκτες για την σύνδεση του σε ηλεκτρονικά κυκλώματα. Πιο συγκεκριμένα, διαθέτει δύο ακροδέκτες για την τροφοδοσία του (γείωση, τάση) και έναν ακροδέκτη ψηφιακής εξόδου.



Εικόνα 14 : Συνδεσμολογία του ανιχνευτή υπερύθρων PNA4602 (Panasonic)

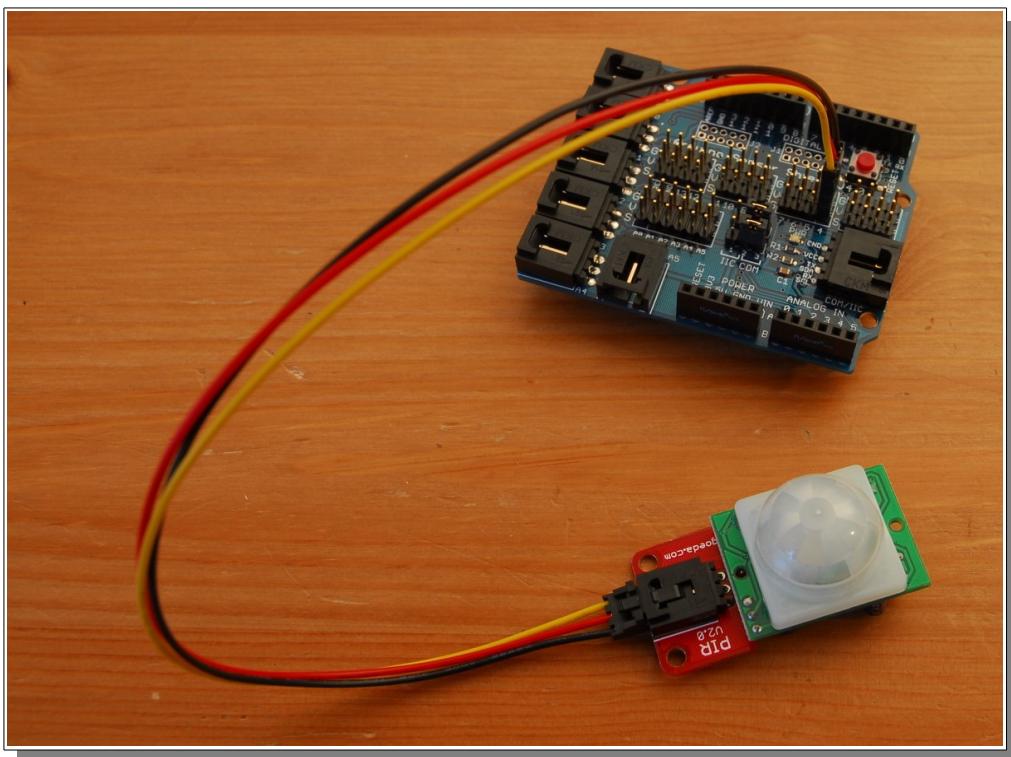
Το εύρος της τάσης τροφοδοσίας για την ορθή λειτουργία του ανιχνευτή υπερύθρων PNA4602 είναι από 3.3 έως 5V. Επιπλέον, το εύρος ευαίσθησίας για την ανίχνευση των σημάτων είναι από 800 έως 1100nm με μέγιστη απόλυτη τιμή 940nm και το εύρος συχνοτήτων για την ανίχνευση του φέροντος σήματος είναι από 35 έως 41KHz με μέση απόλυτη τιμή 38Khz.

Παράλληλα, αξιοποιήθηκε και η βιβλιοθήκη IRremote (η οποία αποτελεί ελεύθερο λογισμικό) για την υποστήριξη του ανιχνευτή υπερύθρων PNA4602. Η βιβλιοθήκη αυτή υποστηρίζει αρκετά πρωτόκολλα κωδικοποίησης δεδομένων. Συγκεκριμένα, μερικά από αυτά είναι τα ακόλουθα : NEC, Sony SIRC, Philips RC5 και Philips RC6. Επομένως, ο κόμβος ταυτοποίησης μπορεί να υποστηρίξει αρκετά τηλεχειριστήρια διότι τα περισσότερα βασίζονται σε κάποιο από τα παραπάνω πρωτόκολλα.

4.04 - Κάρτα Επέκτασης Αισθητήρων

Μία κάρτα επέκτασης αισθητήρων μπορεί να συνδεθεί με κάποια τυπική πλατφόρμα Arduino επιτρέποντας έτσι την γρήγορη και εύκολη διασύνδεση διαφόρων περιφερειακών εξαρτημάτων ως αυτόνομων μονάδων υλικού με την πλατφόρμα.

Μία αυτόνομη μονάδα υλικού συνήθως είναι κατασκευασμένη επάνω σε ανεξάρτητη πλακέτα και διαθέτει κατάλληλη υποδοχή για την σύνδεση της με την κάρτα επέκτασης αισθητήρων μέσω ενός τριπλού καλωδίου (γείωση, τάση, έξοδος). Έτσι λοιπόν, μπορούμε να συνδέσουμε στην κάρτα επέκτασης αισθητήρων διάφορα περιφερειακά εξαρτήματα ως αυτόνομες μονάδες υλικού και να τις χειριστούμε μέσω κάποιας τυπικής πλατφόρμας Arduino.



Εικόνα 15 : Η κάρτα επέκτασης αισθητήρων μαζί με έναν αισθητήρα PIR

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ελέγχου του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μιας κάρτας επέκτασης αισθητήρων για την σύνδεση μερικών ηλεκτρομηχανικών ρελέ (FlamingoEDA) ως αυτόνομες μονάδες υλικού που θα ελέγχουν τις διάφορες συσκευές του κόμβου.

4.05 - Ηλεκτρομηχανικά Ρελέ

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ελέγχου του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μερικών ηλεκτρομηχανικών ρελέ (FlamingoEDA) ως αυτόνομες μονάδες υλικού τόσο για το άνοιγμα όσο και για το κλείσιμο των διαφόρων συσκευών του κόμβου.



Εικόνα 16 : Ένας ηλεκτρομηχανικός ρελές (FlamingoEDA)

Ένας ηλεκτρομηχανικός ρελές είναι ουσιαστικά ένας ηλεκτρομηχανικός διακόπτης που είτε ανοίγει, είτε κλείνει ένα ηλεκτρικό κύκλωμα κάτω από τον έλεγχο ενός άλλου ηλεκτρικού κυκλώματος. Ιστορικά, στην αρχική του μορφή, ο ηλεκτρομηχανικός ρελές διέθετε έναν ειδικό ηλεκτρομαγνήτη ο οποίος ενεργοποιούσε το διακόπτη είτε με το άνοιγμα, είτε με το κλείσιμο μιας ή περισσότερων επαφών. Λόγω του ότι ένα ηλεκτρομηχανικό ρελέ είναι ικανό να ελέγξει ένα κύκλωμα εξόδου υψηλότερης ισχύος από το κύκλωμα εισόδου μπορεί να θεωρηθεί και μια μορφή ηλεκτρικού ενισχυτή.

Συνήθως, ένα ηλεκτρομηχανικό ρελέ διαθέτει μία ή περισσότερες επαφές όπου κάθε μία από αυτές μπορεί να είναι είτε "Κανονικά Ανοικτή" (Normally Open, NO), είτε "Κανονικά Κλειστή" (Normally Closed, NC). Επίσης, μια επαφή "Κανονικά Ανοικτή" συνδέει το κύκλωμα όταν το ρελέ είναι ενεργοποιημένο και αποσυνδέει το κύκλωμα όταν είναι απενεργοποιημένο. Ενώ, μια επαφή "Κανονικά Κλειστή" συνδέει το κύκλωμα όταν το ρελέ είναι απενεργοποιημένο και αποσυνδέει το κύκλωμα όταν είναι ενεργοποιημένο.

4.06 - Κάρτες Ασύρματης Ραδιοεπικοινωνίας

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκαν απαραίτητες μερικές ειδικές κάρτες ασύρματης ραδιοεπικοινωνίας για την υποστήριξη της σειριακής επικοινωνίας ανάμεσα στον κόμβο ελέγχου και στον κόμβο ταυτοποίησης.

Πιο συγκεκριμένα, τόσο στον κόμβο ταυτοποίησης όσο και στον κόμβο ελέγχου χρησιμοποιήθηκαν απλές κάρτες ασύρματης ραδιοεπικοινωνίας XBee (Digi International). Οι κάρτες αυτές χρησιμοποιούνται συνήθως σε δικτυακές εφαρμογές όπου απαιτούνται χαμηλοί ρυθμοί μετάδοσης δεδομένων, δικτυακή ασφάλεια και μεγάλη εξοικονόμηση ενέργειας.



Εικόνα 17 : Κάρτα ασύρματης ραδιοεπικοινωνίας XBee (Digi International)

Επίσης, οι κάρτες XBee βασίζονται στο δικτυακό πρότυπο IEEE 802.15.4 και λειτουργούν στη ζώνη συχνοτήτων 2.4GHz για την μετάδοση δεδομένων. Παράλληλα, παρέχουν αξιόπιστη μετάδοση δεδομένων μεταξύ μικροελεγκτών ή Η/Υ και μπορούν να αξιοποιηθούν για την υλοποίηση ασύρματων ιδιωτικών δικτύων περιορισμένης εμβέλειας. Επίσης, παρακάτω ακολουθούν μερικά από τα πιο βασικά χαρακτηριστικά των καρτών XBee του συστήματος :

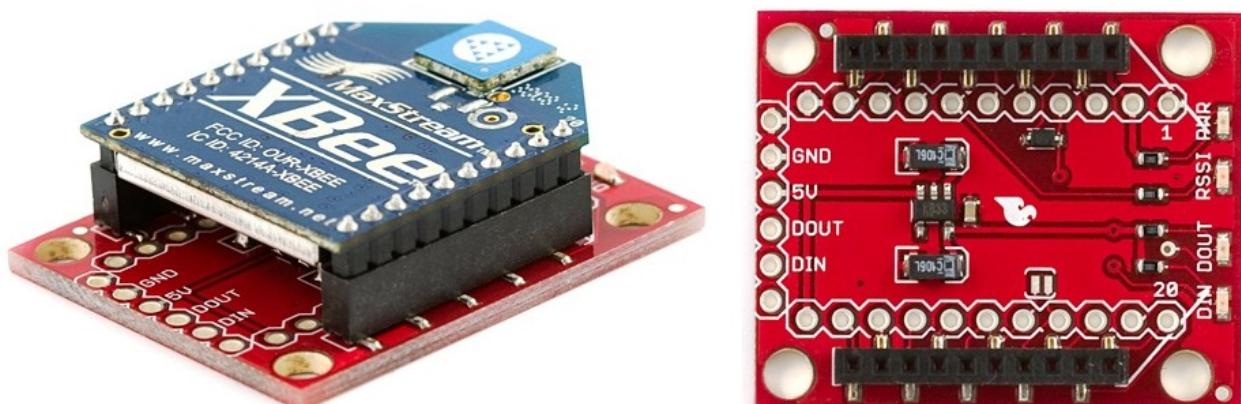
- Εύρος λειτουργικής τάσης τροφοδοσίας : 2.8-3.4V
- Μέγιστος ρυθμός μετάδοσης δεδομένων : 250Kbps
- Μέγιστη εμβέλεια επικοινωνίας : 100m
- Κεραία πομποδέκτη : ενσωματωμένη στην κάρτα
- Ψηφιακές επαφές εισόδου και εξόδου : 8
- Ασφάλεια δεδομένων : κρυπτογράφηση (128 ψηφίων)
- Δικτυακή τοπολογία : σημείο προς σημείο
- Υποστήριξη συνόλου εντολών : AT ή API
- Αναλογικές επαφές εισόδου : 6 (με κανάλια ADC ακρίβειας 10 ψηφίων)
- Ισχύς σήματος εξόδου : 1mW (+0dBm)

4.07 - Προσαρμογείς Καρτών Ασύρματης Ραδιοεπικοινωνίας

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκαν αναγκαίοι μερικοί ειδικοί προσαρμογείς (ειδικές κάρτες) τόσο για την αξιοποίηση όσο και για την αναβάθμιση των καρτών ασύρματης ραδιοεπικοινωνίας των κόμβων.

Γνωρίζουμε ήδη πως τόσο στον κόμβο ταυτοποίησης όσο και στον κόμβο ελέγχου χρησιμοποιήθηκαν απλές κάρτες ασύρματης ραδιοεπικοινωνίας XBee. Ωστόσο, οι κάρτες XBee δεν μπορούν να συνδεθούν απευθείας επάνω στα ράστερ των κόμβων ώστε να λειτουργήσουν άμεσα διότι η λειτουργική τάση τροφοδοσίας που απαιτούν είναι διαφορετική από αυτή που προσφέρουν οι πλατφόρμες Arduino στα ράστερ. Πιο συγκεκριμένα, οι πλατφόρμες Arduino των κόμβων παρέχουν στα ράστερ τάση 5V ενώ οι κάρτες XBee απαιτούν τάση 3.3V για την ορθή λειτουργία τους.

Έτσι, για να γεφυρώσουμε την παραπάνω ασυμβατότητα αξιοποιήσαμε δύο κάρτες προσαρμογής (μία για κάθε κόμβο του συστήματος) XBee Explorer Regulated (SparkFun). Συγκεκριμένα, μία κάρτα προσαρμογής XBee Explorer Regulated μπορεί αρχικά να τροφοδοτηθεί με τάση 5V και στην συνέχεια να υποστηρίξει με τάση 3.3V μία τυπική κάρτα XBee μέσω ενός ενσωματωμένου ρυθμιστή τάσης που διαθέτει.



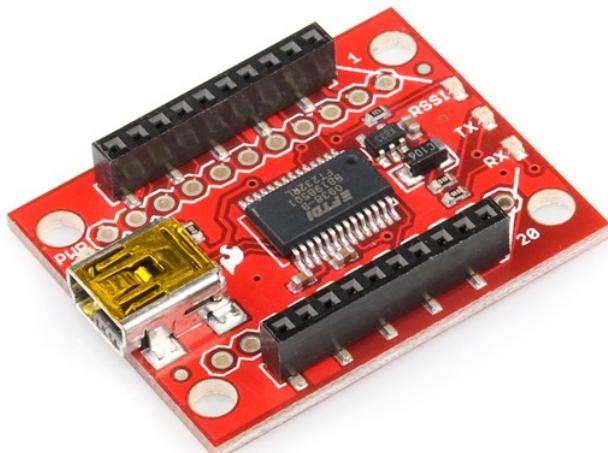
Εικόνα 18 : Η κάρτα προσαρμογής XBee Explorer Regulated (SparkFun)

Επίσης, μία κάρτα προσαρμογής XBee Explorer Regulated διαθέτει μία ειδική υποδοχή για την σύνδεση μιας κάρτας XBee, μερικές επαφές για την τροφοδοσία (5V, GND) και την σειριακή επικοινωνία (DIN, DOUT) καθώς και 4 διόδους εκπομπής φωτός που λειτουργούν ως ενδείκτες κατάστασης για την μεταφορά των σειριακών δεδομένων (DIN, DOUT), την τροφοδοσία (PWR) της κάρτας προσαρμογής και την ισχύ του λαμβανόμενου σήματος (RSSI).

Λόγο του ότι οι σειριακές θύρες των πλατφορμών Arduino των κόμβων του συστήματος χρησιμοποιήθηκαν για αποσφαλμάτωση, αξιοποιήθηκε η βιβλιοθήκη NewSoftSerial (η οποία αποτελεί ελεύθερο λογισμικό) για την δημιουργία νέων σειριακών θυρών σε επίπεδο λογισμικού και την υποστήριξη της επικοινωνίας μεταξύ των πλατφορμών Arduino και των καρτών XBee.

Επίσης, αξίζει να αναφερθεί πως πριν την σύνδεση των καρτών XBee με τις κάρτες προσαρμογής XBee Explorer Regulated κρίθηκε απαραίτητη η αναβάθμιση του υλικολογισμικού (στη νεώτερη έκδοση) των καρτών XBee με την βοήθεια του προγράμματος X-CTU (Digi International). Πιο συγκεκριμένα, για την αναβάθμιση αυτή χρειάστηκαν μία κάρτα προσαρμογής XBee Explorer USB (SparkFun) καθώς και ένα mini USB καλώδιο για την σύνδεση της κάρτας με τον Η/Y.

Η κάρτα XBee Explorer USB έχει αρκετές ομοιότητες με την κάρτα XBee Explorer Regulated. Παρόλα αυτά, παρέχει την σειριακή επικοινωνία μέσω USB όπως και η πλατφόρμα Arduino. Επιπλέον, στο παράρτημα 1 θα βρείτε το σχεδιάγραμμα του κυκλώματος τόσο της κάρτας XBee Explorer Regulated όσο και της κάρτας XBee Explorer USB.



Εικόνα 19 : Η κάρτα προσαρμογής XBee Explorer USB (SparkFun)

4.08 - Οθόνη Υγρών Κρυστάλλων

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε αναγκαία η χρήση μίας οθόνης υγρών κρυστάλλων για την εμφάνιση διαφόρων βιοηθητικών μηνυμάτων.

Συγκεκριμένα, στον κόμβο ταυτοποίησης του συστήματος αξιοποιήθηκε μία οθόνη υγρών κρυστάλλων GDM1602K (Xiamen Ocular). Η οθόνη αυτή μπορεί να εμφανίσει στον χρήστη (σε 2 γραμμές των 16 χαρακτήρων) 32 χαρακτήρες κειμένου. Επίσης, το χρώμα της οθόνης στο παρασκήνιο είναι μπλε ενώ ο οπίσθιος φωτισμός της είναι κίτρινος.

Παράλληλα, μία οθόνη υγρών κρυστάλλων GDM1602K μπορεί να συνδεθεί απευθείας σε κάποιο ράστερ και να λειτουργήσει με τάση 5V. Αξίζει να αναφερθεί πως το βασικότερο χαρακτηριστικό της οθόνης είναι ότι χρησιμοποιεί το κοινό σετ ολοκληρωμένων κυκλωμάτων της παράλληλης διεπαφής HD44780.



Εικόνα 20 : Η οθόνη υγρών κρυστάλλων GDM1602K (Xiamen Ocular)

Ακολουθούν οι επαφές της οθόνης υγρών κρυστάλλων GDM1602K :

Επαφή	Σύμβολο	Λειτουργία
1	VSS	Γείωση τροφοδοσίας (GND)
2	VDD	Τάση τροφοδοσίας (+5V)
3	V0	Ρύθμιση αντίθεσης οθόνης
4	RS	Σήμα επιλογής καταχωρητή (H / L)
5	R / W	Σήμα ανάγνωσης / εγγραφής (H / L)
6	E	Σήμα ενεργοποίησης (H / L)
7	DB0	Γραμμή διαδρόμου δεδομένων (H / L)
8	DB1	Γραμμή διαδρόμου δεδομένων (H / L)
9	DB2	Γραμμή διαδρόμου δεδομένων (H / L)
10	DB3	Γραμμή διαδρόμου δεδομένων (H / L)
11	DB4	Γραμμή διαδρόμου δεδομένων (H / L)
12	DB5	Γραμμή διαδρόμου δεδομένων (H / L)
13	DB6	Γραμμή διαδρόμου δεδομένων (H / L)
14	DB7	Γραμμή διαδρόμου δεδομένων (H / L)
15	A	Τάση διόδου εκπομπής φωτός (+4.2V)
16	K	Γείωση οπίσθιου φωτισμού (GND)

Πίνακας 3 : Επαφές οθόνης υγρών κρυστάλλων GDM1602K (*Xiamen Ocular*)

Επίσης, αξιοποιήθηκε και η βιβλιοθήκη ShiftRegLCD (η οποία αποτελεί ελεύθερο λογισμικό) για την υποστήριξη της οθόνης υγρών κρυστάλλων GDM1602K. Η βιβλιοθήκη αυτή μπορεί να χρησιμοποιηθεί για τον χειρισμό μίας HD44780 συμβατής οθόνης υγρών κρυστάλλων μέσω 2 ή 3 καλωδίων και ενός καταχωρητή ολίσθησης τύπου 74LS164 ή 74HC595.

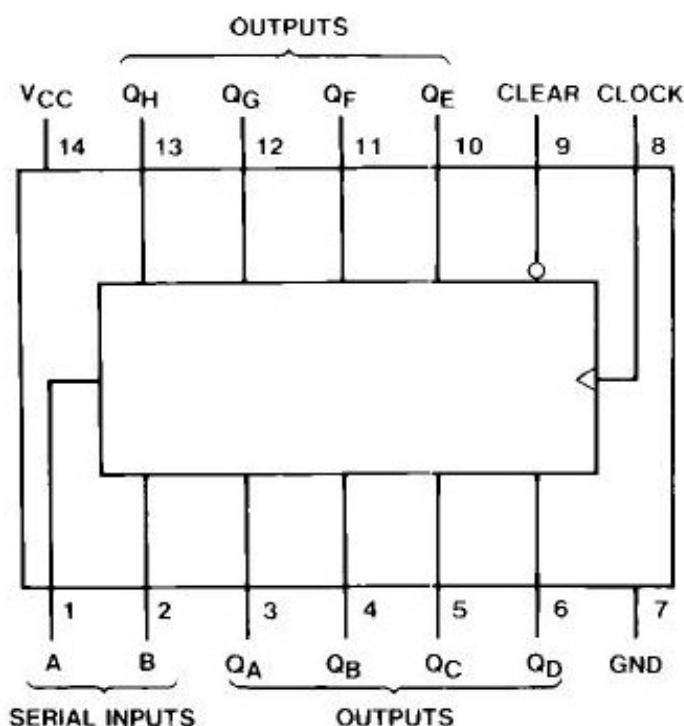
Παράλληλα, στο παράρτημα 1 θα βρείτε ένα σχεδιάγραμμα σχετικά με την συνδεσμολογία μεταξύ της οθόνης υγρών κρυστάλλων GDM1602K και του καταχωρητή ολίσθησης DM74LS164N που χρησιμοποιήθηκε για την οδήγηση της οθόνης.

4.09 - Καταχωρητής Ολίσθησης

Γενικότερα, ένας καταχωρητής είναι απλά μία ομάδα από Flip-Flop που χρησιμοποιούνται για την αποθήκευση ενός δυαδικού αριθμού. Το Flip-Flop αποτελεί στοιχειώδες “κύτταρο μνήμης” στα ηλεκτρονικά κυκλώματα και μπορεί να χρησιμοποιηθεί για την αποθήκευση ενός ψηφίου. Έτσι, για κάθε ένα από τα ψηφία ενός δυαδικού αριθμού είναι απαραίτητο να υπάρχει και ένα ξεχωριστό Flip-Flop.

Φυσικά, τα Flip-Flop θα πρέπει να είναι συνδεδεμένα κατάλληλα ώστε ο δυαδικός αριθμός να μπορεί είτε να εισαχθεί μέσα στον καταχωρητή, είτε να εξαχθεί από αυτόν. Σε κάθε περίπτωση μπορεί να υπάρχει από τον καταχωρητή και σχετική μετατόπιση των ψηφίων του δυαδικού αριθμού. Επίσης, μία ομάδα από Flip-Flop που προσφέρει είτε τη μία, είτε την άλλη, είτε και τις δύο λειτουργίες, συμπεριλαμβανομένου της μετατόπισης που αναφέρθηκε πιο πάνω ονομάζεται καταχωρητής ολίσθησης.

Τα ψηφία σε ένα δυαδικό αριθμό (θα τα ονομάσουμε δεδομένα) μπορούν να μετακινηθούν (να αλλάξουν θέσεις) με δύο μεθόδους αφορά στη μετατόπιση των δεδομένων ανά 1 ψηφίο κάθε φορά στη σειρά, αρχίζοντας είτε με το περισσότερο σημαντικό ψηφίο, είτε με το λιγότερο σημαντικό ψηφίο. Η τεχνική αυτή ονομάζεται ολίσθηση σε σειρά. Επίσης, η δεύτερη μέθοδος αφορά στη ταυτόχρονη μετατόπιση όλων των ψηφίων και ονομάζεται παράλληλη ολίσθηση.

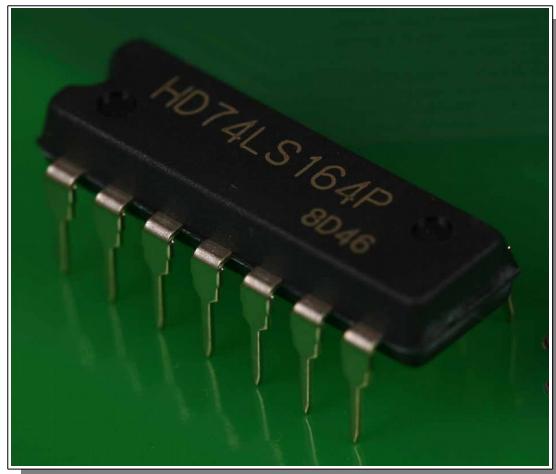


Εικόνα 21 : Οι επαφές του καταχωρητή ολίσθησης DM74LS164N

Υπάρχουν δύο τρόποι μετατόπισης δεδομένων μέσα σε έναν καταχωρητή (σειράς ή παράλληλο) καθώς και, παρόμοια, δύο τρόποι εξαγωγής δεδομένων από τον καταχωρητή. Το γεγονός αυτό οδηγεί στην κατασκευή τεσσάρων βασικών ειδών καταχωρητών :

- Σειριακή Είσοδος – Σειριακή Έξοδος
- Σειριακή Είσοδος – Παράλληλη Έξοδος
- Παράλληλη Είσοδος – Σειριακή Έξοδος
- Παράλληλη Είσοδος – Παράλληλη Έξοδος

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση ενός καταχωρητή ολίσθησης DM74LS164N (σειριακής εισόδου - παράλληλης εξόδου) για την οδήγηση της οθόνης υγρών κρυστάλλων GDM1602K (Xiamen Ocular) του κόμβου. Επίσης, στο παράρτημα 1 θα βρείτε το σχεδιάγραμμα του κυκλώματος του καταχωρητή ολίσθησης DM74LS164N.



Εικόνα 22 : Ο καταχωρητής ολίσθησης HD74LS164P

Παράλληλα, η βιβλιοθήκη που χρησιμοποιήθηκε για την υποστήριξη της οθόνης υγρών κρυστάλλων GDM1602K σε συνεργασία με τον καταχωρητή ολίσθησης DM74LS164N είναι η ShiftRegLCD (η οποία αποτελεί ελεύθερο λογισμικό). Πιο συγκεκριμένα, η βιβλιοθήκη αυτή μπορεί να χρησιμοποιηθεί για τον χειρισμό μίας HD44780 συμβατής οθόνης υγρών κρυστάλλων μέσω 2 ή 3 καλωδίων και ενός καταχωρητή ολίσθησης τύπου 74LS164 ή 74HC595.

Στην αρχική έκδοση του κόμβου ταυτοποίησης είχε χρησιμοποιηθεί μόνο ο καταχωρητής ολίσθησης DM74LS164N και η οθόνη υγρών κρυστάλλων GDM1602K, επιτρέποντας έτσι την οδήγηση της οθόνης μέσω 3 καλωδίων. Στη συνέχεια, το κύκλωμα του κόμβου εξελίχθηκε ώστε να είναι δυνατή η οδήγηση της οθόνης μόνο μέσω 2 καλωδίων. Ωστόσο, κρίθηκε απαραίτητη και η χρήση μίας διόδου 1N4001 σε συνεργασία με έναν αντιστάτη 1ΚΩ για την υλοποίηση της λογικής πράξης της σύζευξης στο κύκλωμα του κόμβου.

4.10 - USB Καλώδια Πλατφορμών Arduino

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μερικών ειδικών καλωδίων USB για τον προγραμματισμό και την τροφοδοσία των πλατφορμών Arduino των κόμβων. Πιο συγκεκριμένα, χρησιμοποιήθηκαν δύο λευκά καλώδια USB 2.0 (ένα για κάθε πλατφόρμα Arduino) τύπου A-B (αρσενικό σε αρσενικό).



Εικόνα 23 : Μαύρο καλώδιο USB 2.0 τύπου A-B (αρσενικό σε αρσενικό)

4.11 - USB Τροφοδοτικά Τοίχου

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση δύο USB τροφοδοτικών τοίχου για την άμεση τροφοδοσία των πλατφορμών Arduino (ένα για κάθε μία πλατφόρμα) των κόμβων. Πιο συγκεκριμένα, τα USB τροφοδοτικά τοίχου που χρησιμοποιήθηκαν διαθέτουν μία USB θηλυκή υποδοχή τύπου A και μπορούν να μετατρέψουν ένα αναλογικό σήμα εισόδου τάσης από 100 έως 240V σε ένα συνεχές σήμα εξόδου τάσης 5V.



Εικόνα 24 : Τροφοδοτικό τοίχου 5V με USB υποδοχή τύπου A

4.12 - Συσκευή Ανάγνωσης RFID

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μίας συσκευής για την ανάγνωση μερικών παθητικών ετικετών ραδιοσυχνικής ταυτοποίησης (RFID).

Συγκεκριμένα, στο κόμβο ταυτοποίησης του συστήματος αξιοποιήθηκε η συσκευή ανάγνωσης RFID ID-12 (SparkFun) η οποία αποτελεί αυτόνομη μονάδα υλικού και διαθέτει ειδική ενσωματωμένη κεραία. Οι επαφές της συσκευής αυτής έχουν απόσταση 2mm. Επίσης, η διασύνδεση και η χρήση της συσκευής είναι αρκετά απλή. Εφόσον τροφοδοτηθεί η συσκευή με κατάλληλη τάση και μία παθητική ετικέτα RFID βρεθεί εντός της υποστηριζόμενης εμβέλειας τότε η συσκευή εξάγει στην επαφή εξόδου της τον μοναδικό σειριακό αριθμό της ετικέτας.



Εικόνα 25 : Η συσκευή ανάγνωσης RFID ID-12 (SparkFun)

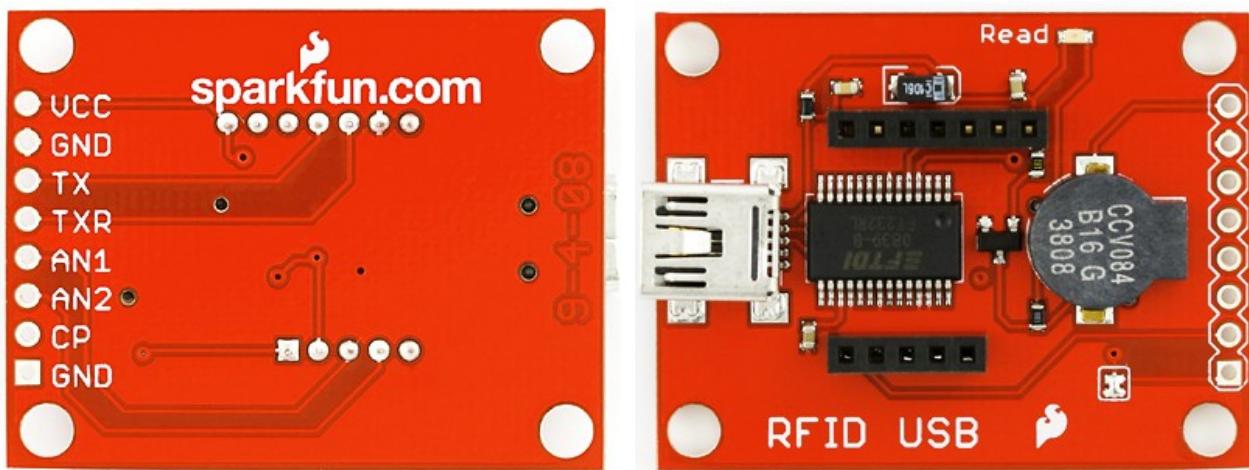
Μερικά βασικά χαρακτηριστικά της συσκευής ανάγνωσης RFID ID-12 :

- Λειτουργική τάση τροφοδοσίας : 5V
- Συχνότητα λειτουργίας του πομποδέκτη : 125KHz
- Πρωτόκολλο ετικετών RFID : EM4001 (ISO)
- Ρυθμός μετάδοσης : 9600bps
- Εμβέλεια ανάγνωσης : 100mm
- Μορφότυπα δεδομένων : ASCII, Wiegand26, Magnetic ABA Track2

Επιπλέον, επειδή η σειριακή θύρα της πλατφόρμας Arduino του κόμβου ταυτοποίησης του συστήματος χρησιμοποιήθηκε για την αποστολή μηνυμάτων αποσφαλμάτωσης, αξιοποιήθηκε η βιβλιοθήκη NewSoftSerial (η οποία αποτελεί ελεύθερο λογισμικό) για την δημιουργία μιας νέας σειριακής θύρας σε επίπεδο λογισμικού και την υποστήριξη της επικοινωνίας μεταξύ της πλατφόρμας Arduino και της συσκευής ανάγνωσης RFID ID-12.

4.13 - Προσαρμογέας Συσκευής Ανάγνωσης RFID

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη μία κάρτα προσαρμογής για την χρήση της συσκευής ανάγνωσης ραδιοσυχνικής ταυτοποίησης (RFID) ID-12 (SparkFun) του κόμβου.



Εικόνα 26 : Η USB συσκευή ανάγνωσης RFID (SparkFun)

Συγκεκριμένα, ως κάρτα προσαρμογής αξιοποιήθηκε μία USB συσκευή ανάγνωσης RFID (SparkFun). Η σύνδεση αυτής της συσκευής είναι απλή και πραγματοποιείται μέσω των επαφών VCC (τάση), GND (γείωση), TX (έξοδος). Επίσης, ο προσαρμογέας διαθέτει κατάλληλη υποδοχή για τους αναγνώστες RFID ID-2, ID-12 και ID-20, ένα κύκλωμα προσαρμογής FTDI για την σειριακή επικοινωνία μεταξύ USB και TTL, μία δίοδο εκπομπής φωτός καθώς και ένα μικρό μεγάφωνο ως ενδείκτες ανάγνωσης. Παράλληλα, στο παράρτημα 1 θα βρείτε το σχεδιάγραμμα του κυκλώματος της USB συσκευής ανάγνωσης RFID.

4.14 - Παθητικές Ετικέτες RFID

Οι ετικέτες ραδιοσυχνικής ταυτοποίησης (RFID) αποτελούν πομποδέκτες που ενσωματώνουν ένα ηλεκτρονικό κύκλωμα με μνήμη και κεραία. Σε κάθε ετικέτα RFID είναι αποθηκευμένος ένας μοναδικός σειριακός αριθμός ο οποίος προσδιορίζει μοναδικά ένα αντικείμενο. Πιο συγκεκριμένα, οι ετικέτες RFID αξιοποιούνται σε εφαρμογές όπου είναι απαραίτητη η ασύρματη αναγνώριση αντικειμένων μέσω ταυτοποίησης ραδιοσυχνοτήτων.



Εικόνα 27 : Παθητική ετικέτα RFID (πλαστική λευκή κάρτα)

Μία ετικέτα RFID μπορεί να ανιχνευθεί με την βοήθεια ενός αναγνώστη RFID. Επίσης, η κεραία μίας ετικέτας RFID επιτρέπει στο ηλεκτρονικό της κύκλωμα να μεταδώσει τις πληροφορίες αναγνώρισης σε ένα αναγνώστη RFID, ο οποίος με τη σειρά του μπορεί να μετατρέψει τα διάφορα ραδιοκύματα που “αντανακλώνται” από την ετικέτα σε χρήσιμες ψηφιακές πληροφορίες.



Εικόνα 28 : Παθητική ετικέτα RFID (πλαστικός μαύρος δίσκος)

Αξίζει να αναφερθεί πως οι ετικέτες RFID που δεν διαθέτουν κάποια αποκλειστική μπαταρία για την λειτουργία τους, αλλά η τροφοδοσία τους γίνεται μέσω ενός αναγνώστη RFID χαρακτηρίζονται ως “παθητικές”. Πιο συγκεκριμένα, όταν τα διάφορα ραδιοκύματα αποστέλλονται από έναν αναγνώστη RFID σε μία ετικέτα RFID και λαμβάνονται από την κεραία της, τότε παρουσιάζεται ηλεκτρομαγνητική επαγωγή που ως αποτέλεσμα έχει την άμεση τροφοδοσία του ηλεκτρονικού κυκλώματος της ετικέτας.

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μερικών παθητικών ετικετών RFID. Πιο συγκεκριμένα, ως ετικέτες RFID χρησιμοποιήθηκαν τρεις λευκές και ορθογώνιες πλαστικές κάρτες, δύο μαύροι και πλαστικοί δίσκοι καθώς και μία γυάλινη διαφανή κάψουλα.



Εικόνα 29 : Παθητική ετικέτα RFID (γυάλινη διαφανή κάψουλα)

Το πακέτο μετάδοσης των ετικετών RFID του συστήματος ξεκινάει με το πεδίο STX που περιέχει την τιμή 02H και σηματοδοτεί την αρχή του πακέτου. Στην συνέχεια, ακολουθεί το πεδίο DATA που περιέχει τον μοναδικό σειριακό αριθμό της ετικέτας RFID και έχει μέγεθος 10 bytes στον κώδικα χαρακτήρων ASCII, ενώ 5 bytes στο δεκαεξαδικό σύστημα. Αμέσως μετά, ακολουθεί το πεδίο CHECKSUM που περιέχει δεδομένα πλεονασμού και χρησιμοποιείται για την ανίχνευση πιθανών σφαλμάτων. Επίσης, το πεδίο CHECKSUM έχει μέγεθος 2 bytes στον κώδικα χαρακτήρων ASCII, ενώ 1 byte στο δεκαεξαδικό σύστημα.

Μετέπειτα, ακολουθούν ένας χαρακτήρας “επιστροφής στην αρχή της γραμμής” (Carriage Return – CR) και ένας χαρακτήρας “αλλαγής γραμμής” (Line Feed – LF). Το πεδίο CR περιέχει την τιμή 0DH, ενώ το πεδίο LF περιέχει την τιμή 0AH. Τέλος, ακολουθεί το πεδίο ETX που περιέχει την τιμή 03H και σηματοδοτεί τον τερματισμό του πακέτου μετάδοσης. Οι τιμές των πεδίων STX, CR, LF και ETX είναι ίσες η κάθε μία στο δεκαεξαδικό σύστημα με 1 byte, ενώ στον κώδικα χαρακτήρων ASCII με 2 bytes.

Η δομή του πακέτου μετάδοσης των παθητικών ετικετών RFID :

STX (02H)	DATA (10 ASCII)	CHECKSUM (2 ASCII)	CR (0DH)	LF (0AH)	ETX (03H)
----------------------	----------------------------	-------------------------------	---------------------	---------------------	----------------------

Πίνακας 4 : Η δομή του πακέτου μετάδοσης των παθητικών ετικετών RFID

Μερικά βασικά χαρακτηριστικά των παθητικών ετικετών RFID ID-12 :

- Μήκος (σε ψηφία) του συνολικού πακέτου μετάδοσης : 64 ψηφία
- Μήκος (σε ψηφία) του σειριακού αριθμού της ετικέτας : 32 ψηφία
- Πρωτόκολλο ετικετών RFID : EM4001 (ISO)
- Συχνότητα λειτουργίας του πομποδέκτη : 125KHz
- Μέθοδος κωδικοποίησης δεδομένων : Manchester

4.15 - Ηλεκτρική Δίοδος

Η δίοδος είναι ένα ηλεκτρικό εξάρτημα που περιορίζει τη κατευθυντήρια ροή των φορέων αγωγιμότητας. Πιο συγκεκριμένα, η δίοδος επιτρέπει το ηλεκτρικό ρεύμα να περάσει από τη μια διεύθυνση, αλλά εμποδίζει την κίνηση του από την αντίθετη διεύθυνση. Έτσι, η δίοδος μπορεί να θεωρηθεί και ως η ηλεκτρονική εκδοχή της βαλβίδας.

Γενικότερα, τα κυκλώματα που απαιτούν ροή προς μία μόνο κατεύθυνση συνήθως περιλαμβάνουν μία ή και περισσότερες διόδους. Επίσης, οι δίοδοι τις περισσότερες φορές κατασκευάζονται από υλικά ημιαγωγών όπως το πυρίτιο ή το γερμάνιο (και όχι μόνο).



Εικόνα 30 : Η ηλεκτρική δίοδος 1N4001

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μίας διόδου 1N4001 σε συνεργασία με έναν αντιστάτη 1ΚΩ για την υλοποίηση της λογικής πράξης της σύζευξης στο κύκλωμα του κόμβου.

4.16 - Ηλεκτρομηχανικό Ποτενσιόμετρο

Το ποτενσιόμετρο αποτελεί αναλογικό ηλεκτρομηχανικό εξάρτημα και μπορεί να χρησιμοποιηθεί σε διάφορα κυκλώματα ως μεταβλητός ηλεκτρικός αντιστάτης. Πιο συγκεκριμένα, αποτελείται από μία αγώγιμη πλάκα σχήματος “Ω”, πάνω στην οποία γυρίζει μια επαφή με την βοήθεια ενός στροφέα. Έτσι, ανάλογα με την απόσταση της επαφής από την είσοδο του ρεύματος στο ποτενσιόμετρο, δημιουργείται και η ανάλογη αντίσταση.



Εικόνα 31 : Ένα τυπικό ηλεκτρομηχανικό ποτενσιόμετρο $10K\Omega$

Συνήθως, τα περισσότερα ποτενσιόμετρα διαθέτουν τρεις επαφές. Πιο συγκεκριμένα, χρησιμοποιούνται δύο επαφές για την γείωση και την τάση καθώς και μία επαφή για την έξοδο της νέας τάσης η οποία εξαρτάται από την αντίσταση του ποτενσιόμετρου.

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση ενός ποτενσιόμετρου $470K\Omega$ για τον έλεγχο της αντίθεσης της τυπικής οθόνης υγρών κρυστάλλων GDM1602K (Xiamen Ocular) του κόμβου.

4.17 - Ηλεκτρικοί Αντιστάτες

Ο αντιστάτης είναι ένα εξάρτημα το οποίο χρησιμοποιείται σε διάφορα κυκλώματα για τον έλεγχο της ροής του ηλεκτρικού ρεύματος. Οι πιο κοινές συνδεσμολογίες αντιστατών που συναντά κανείς στα ηλεκτρονικά κυκλώματα είναι η σύνδεση αντιστατών σε σειρά και η σύνδεση αντιστατών παράλληλα.

Επίσης, ο αντιστάτης μερικές φορές λέγεται και ηλεκτρική αντίσταση. Ωστόσο η ηλεκτρική αντίσταση είναι απλώς φαινόμενο και όχι εξάρτημα. Πιο συγκεκριμένα, είναι το μέγεθος με το οποίο προσμετράται η δυσχέρεια στην έλευση ηλεκτρικού ρεύματος μέσα από ένα υλικό, όταν το υλικό αυτό δε φέρει ιδιάζον σχήμα έτσι ώστε να μην αναπτύσσονται επιπλέον ηλεκτρικές ιδιότητες οφειλόμενες σε χωρητικά ή επαγωγικά φαινόμενα.



Εικόνα 32 : Ένας ηλεκτρικός αντιστάτης $1\text{K}\Omega$

Η αντίσταση θεωρείται η αντίθετη έννοια της αγωγιμότητας. Υλικά που είναι μονωτές έχουν μεγάλη αντίσταση, ενώ υλικά που είναι αγωγοί έχουν μικρή αντίσταση. Η αντίσταση μπορεί να μην είναι σταθερή, αλλά να αλλάζει ανάλογα με τις εξωτερικές συνθήκες ή να εξαρτάται από το ηλεκτρικό ρεύμα.

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση διαφόρων αντιστατών. Πιο συγκεκριμένα, χρησιμοποιήθηκε για κάθε μία δίοδο εκπομπής φωτός ένας αντιστάτης 560Ω . Επίσης, στον κόμβο ταυτοποίησης χρησιμοποιήθηκε ένας αντιστάτης $10\text{K}\Omega$ για το κουμπί επιλογής των λειτουργιών του κόμβου καθώς καθώς και ένας αντιστάτης $1\text{K}\Omega$ σε συνεργασία με μία δίοδο $1N4001$ για την υλοποίηση της λογικής πράξης της σύζευξης στο κύκλωμα του κόμβου.

4.18 - Ηλεκτρομηχανικό Κουμπί

Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τον ενσωματωμένο κόμβο ταυτοποίησης του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητο ένα ηλεκτρομηχανικό κουμπί για την επιλογή των λειτουργιών που υποστηρίζει ο κόμβος. Πιο συγκεκριμένα, με το πάτημα του κουμπιού ο κόμβος εισέρχεται στην επόμενη λειτουργία που παρέχει. Με το επαναληπτικό πάτημα του κουμπιού ο κόμβος εισέρχεται στις διάφορες λειτουργίες του κυκλικά.



Εικόνα 33 : Ένα ηλεκτρομηχανικό κουμπί

Επίσης, αξίζει να αναφερθεί πως με το πάτημα του κουμπιού του κόμβου ταυτοποίησης αποστέλλεται ένα ασύγχρονο σήμα διακοπής στον μικροελεγκτή AVR του κόμβου. Σε περίπτωση που το σήμα διακοπής παραχθεί τη στιγμή που ο μικροελεγκτής AVR εκτελεί κάποιο κρίσιμο τμήμα του υλικολογισμικού του κόμβου τότε αυτό αγνοείται. Σε κάθε άλλη περίπτωση ο κόμβος σπεύδει να εξυπηρετήσει το σήμα διακοπής με την κατάλληλη ρουτίνα εξυπηρέτησης.

4.19 - Δίοδοι Εκπομπής Φωτός

Δίοδος εκπομπής φωτός αποκαλείται ένας ημιαγωγός ο οποίος εκπέμπει φωτεινή ακτινοβολία στενού φάσματος όταν του παρέχεται μία ηλεκτρική τάση κατά τη φορά ορθής πόλωσης. Επίσης, το χρώμα του φωτός που εκπέμπεται εξαρτάται από την χημική σύσταση του ημιαγώγιμου υλικού που χρησιμοποιείται, και μπορεί να είναι υπεριώδες, ορατό ή υπέρυθρο.

Το μήκος κύματος του φωτός που εκπέμπεται και κατά συνέπεια το χρώμα του, εξαρτάται από το χάσμα ενέργειας των υλικών, τα οποία χρησιμοποιούνται για την δημιουργία του περάσματος ρ-η, (όπου ρ : υλικό νοθευμένο με αποδέκτες και η : υλικό νοθευμένο με δότες) της διόδου. Επίσης, μια δίοδος εκπομπής φωτός είναι στην ουσία μια ένωση ρ-η που έχει κατασκευαστεί από ένα ημιαγωγό άμεσου ενεργειακού χάσματος και στην οποία η επανασύνδεση των ζευγών ηλεκτρονίων - οπών έχει ως αποτέλεσμα την εκπομπή φωτονίων.



Εικόνα 34 : Πέντε δίοδοι εκπομπής φωτός κόκκινου χρώματος

Η βασική αρχή μίας διόδου εκπομπής φωτός είναι μια επαφή ρ-η η οποία πολώνεται ορθά για να εγχέει ηλεκτρόνια και οπές μέσα στις ρ- και η- πλευρές αντίστοιχα. Το εγχεόμενο φορτίο μειονότητας επανασυνδέεται με το φορτίο πλειονότητας στην περιοχή απογύμνωσης ή στην ουδέτερη περιοχή. Σε ημιαγωγούς άμεσου διάκενου η επανασύνδεση οδηγεί σε εκπομπή φωτός αφού η ακτινοβόλα επανασύνδεση κυριαρχεί σε υλικά υψηλής ποιότητας. Σε υλικά έμμεσου χάσματος, η απόδοση εκπομπής φωτός είναι αρκετά φτωχή και οι περισσότερες από τις διαδρομές επανασύνδεσης είναι μη ακτινοβόλες με παραγωγή θερμότητας μάλλον παρά φωτός.

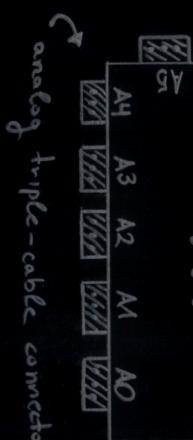
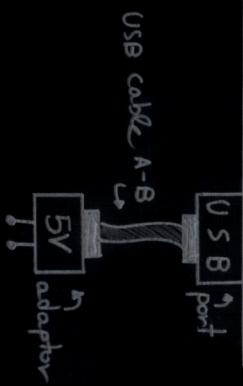
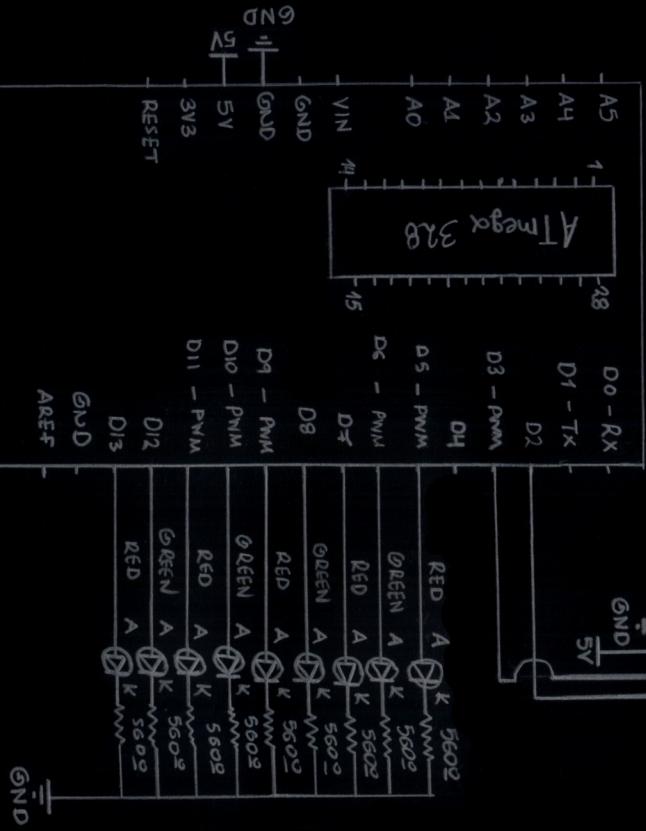
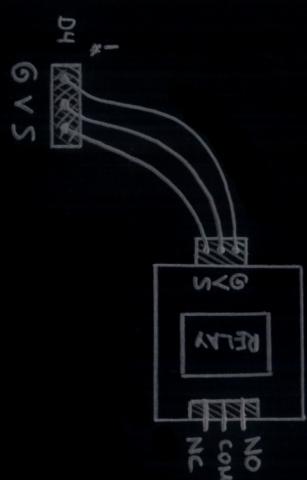
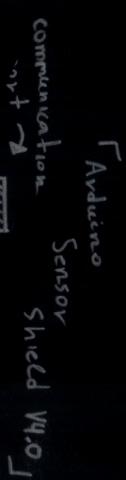
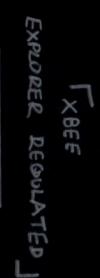
Κατά την διάρκεια της ανάλυσης που πραγματοποιήθηκε σχετικά με τους ενσωματωμένους κόμβους του συστήματος που πραγματεύεται αυτή η πτυχιακή εργασία, κρίθηκε απαραίτητη η χρήση μερικών διόδων εκπομπής φωτός κόκκινου και πράσινου χρώματος.

Κεφάλαιο 5 - Διασύνδεση Συστατικών Συστήματος

Layer φ - Bottom -

Layer 1 - Top.

Relay Module



Layer ϕ - Bottom.

۷۰۰

Layer 1 - Top.

Κεφάλαιο 6 - Προγραμματισμός Συστήματος

Παρακάτω, ακολουθεί ο πηγαίος κώδικας με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ταυτοποίησης :

```
/*
 * Remote Systems Control With Wireless Ad-Hoc Radio Network And RFID.
 *
 * This sketch implements the firmware of the identification node.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

/*
 * include external libraries headers.
 */

// library for creating software serial ports.
#include <NewSoftSerial.h>

// library for using LCDs with shift registers.
#include <ShiftRegLCD.h>

// library for using output streaming operators.
#include <Streaming.h>

// library for handling IR remote control sensors.
#include <IRremote.h>

// library for handling EEPROM memory of Arduino.
#include <EEPROM.h>

// library for performing non-local (long) jumps.
#include <setjmp.h>

// library for using SD storage card devices.
#include <SdFat.h>

// library for handling flash memory of Arduino.
#include <Flash.h>

/*
 * include internal libraries headers.
 */

// library for generic handling of Arduino EEPROM.
```

```

#include "eeprom_generic.h"

/*
 * define preprocessor constants.
 */

// set true / false for debug support.
#define DEBUG_STATE true

// define controllers if debug state is on.
#if defined (DEBUG_STATE) && true == DEBUG_STATE
    // for hardware serial verbose printing.
    #define SERIAL_ENABLED
#endif

/*
 * define function-like macros.
 */

// macro for performing a "kernel" panic state and
// printing an error from the Arduino flash memory.
#define error(s) systemError (PSTR (s))

/*
 * define Arduino I/O PINS constants.
 */

// the PIN number for the LCD's shift register data.
const byte LCD_SR_DTA_PIN = 7;

// the PIN number for the LCD's shift register clock.
const byte LCD_SR_CLK_PIN = 9;

// the PIN number for the RFID reader serial RX line.
const byte RFID_RLINE_PIN = 4;

// the PIN number for the RFID reader serial TX line (dumb).
const byte RFID_TLINE_PIN = 5;

// the PIN number for the RMIR sensor RX line.
const byte RMIR_RLINE_PIN = 6;

// the PIN number for the external status led.
const byte LED_STATUS_PIN = 3;

// the PIN number for the external state button.
const byte BUT_STATES_PIN = 2;

// the PIN number for the default hardware chip select.
const byte DEFAULT_CS_PIN = 10;

// the PIN number for the XBEE reader serial TX line.
const byte XBEE_TLINE_PIN = 5;

// the PIN number for the XBEE reader serial RX line.
const byte XBEE_RLINE_PIN = 8;

/*
 * define Arduino I/O IRQ constants.
 */

// the IRQ number for the external state button.

```

```

const byte BUT_STATES_IRQ = 0;

/*
 * define Arduino I/O serial port constants.
 */

#if defined (SERIAL_ENABLED)
// hardware serial port baud rate.
const unsigned long USB_BAUD_RATE = 9600;
#endif

// RFID software serial port baud rate.
const unsigned long RFID_BAUD_RATE = 9600;

// XBEE software serial port baud rate.
const unsigned long XBEE_BAUD_RATE = 9600;

/*
 * define FSM-related variables.
 */

// define FSM state function type definition name.
typedef void (*FSMStateFunctionT) ();

// array of FSM states functions' pointers.
const FSMStateFunctionT FSMStatesFunctions[][][2] = {
    // FSM state for doing nothing - idling.
    { idleP, idleV },

    // FSM state for handling an RFID tag.
    { RFIDHandleTagP, RFIDHandleTagV },

    // FSM state for learning an RFID tag.
    { RFIDLearnTagP, RFIDLearnTagV },

    // FSM state for learning RMIR keys.
    { RMIRLearnKeysP, RMIRLearnKeysV }
};

// calculate the number of total FSM states in the array.
const byte FSM_STATES_NUM =
    sizeof (FSMStatesFunctions) / (sizeof (const FSMStateFunctionT) * 2);

// current FSM state number (zero-first, index-based).
volatile byte FSMCurrentState = 0;

/*
 * define LCD-related variables.
 */

// number of total LCD lines.
const byte LCD_LINES_NUM = 2;

// object to control the LCD with a shift register.
ShiftRegLCD SYS_LCD (LCD_SR_DTA_PIN,
                      LCD_SR_CLK_PIN,
                      TWO_WIRE,
                      LCD_LINES_NUM);

/*
 * define XBEE-related variables.
 */

```

```

// object to control the XBEE software serial port.
NewSoftSerial SYS_XBEE = NewSoftSerial (XBEE_RLINE_PIN, XBEE_TLINE_PIN);

// define XBEE data separator token.
const char XBEEDataSeparator = '!';

/*
 * define RFID-related variables.
 */

// object to control the RFID software serial port.
NewSoftSerial SYS_RFID = NewSoftSerial (RFID_RLINE_PIN, RFID_TLINE_PIN);

// RFID knowledge filename (8.3 filename format).
const char RFID_KNOWLEDGE_FILENAME[] = "rfids.txt";

// the code size (digits) of the RFID tags.
const byte RFID_CODE_SIZE = 10;

// temporary RFID code (+1 for string termination).
char RFIDCode[RFID_CODE_SIZE + 1] = { '\0' };

/*
 * define RMIR-related variables.
 */

// object to control the RMIR sensor.
IRrecv SYS_RMIR (RMIR_RLINE_PIN);

// object to handle decoded RMIR codes.
decode_results RMIRCodesDecoded;

// define bounce time for RMIR codes.
const word RMIR_CODE_DURATION = 1000;

// size of the RMIR codes arrays.
const byte RMIR_ARRAY_SIZE = 60;

// array for the current RMIR codes.
uint32_t RMIRCurrentCodes[RMIR_ARRAY_SIZE];
// array for the new RMIR codes.
uint32_t RMIRNewCodes[RMIR_ARRAY_SIZE];

// number of the real RMIR codes.
byte RMIRCodesNumber = 0;

// temporary RMIR code index.
int8_t RMIRCodeIndex = -1;

/*
 * define microSD-related variables.
 */

// object to control the microSD card of the microSD shield.
Sd2Card SDCard;

// object to control the volume of the microSD card.
SdVolume SDVolume;

// object to control the root directory of the microSD card.
SdFile SDRoot;

// object to control the RFID knowledge file of the microSD card.

```

```

SdFile RFIDFile;

/*
 * define general variables.
 */

// information to restore calling environment.
jmp_buf bufEnv;

// define bounce duration time for buttons.
const word BUTTON_KEY_DURATION = 500;

// millis count to debounce pressed buttons.
volatile unsigned long buttonKeyBounceTime = 0;

/*
 * define ISR-related functions.
 */

// ISR function for the external state button IRQ.
void stateButtonISR ()
{
    // get the time since board began running the current program.
    unsigned long time = millis ();

    // check both if there is an overflow in millis() and ignore
    // presses intervals less than the bounce duration time (ms).

    if ((time < buttonKeyBounceTime) ||
        ((time - buttonKeyBounceTime) > BUTTON_KEY_DURATION)) {
        // change the current FSM state to the next one (cyclic behaviour).
        FSMCurrentState = ++FSMCurrentState % FSM_STATES_NUM;

        // flush any possible data coming from RFID reader.
        SYS_RFID.flush ();

        // flush any possible data coming from RMIR sensor (hack).
        RMIRFlush ();

        // clear the LCD display.
        SYS_LCD.clear ();

        // print a message to LCD according to the FSM.
        FSMStatesFunctions[FSMCurrentState][1] ();

        // set whatever bounce time in ms is appropriate.
        buttonKeyBounceTime = millis ();

        // go to the main loop.
        longjmp (bufEnv, 0);
    }
}

// try to attach ISRs for external interrupts.
void attachInterruptsISRs ()
{
    // attach ISR for the IRQ (for state button presses).
    attachInterrupt (BUT_STATES_IRQ, stateButtonISR, RISING);
}

// try to detach ISRs for external interrupts.

```

```

void
detachInterruptsISRs ()
{
    // detach ISR for the IRQ (for state button presses).
    detachInterrupt (BUT_STATES_IRQ);
}

/*
 * define RMIR-related functions.
 */

// try to check if an RMIR code exists.
int8_t
RMIRCodeExists ()
{
    // get the type of the RMIR code.
    int16_t cType = RMIRCodesDecoded.decode_type;

    // if the RMIR code type is not unknown.
    if (cType != UNKNOWN) {
        // get the value of the RMIR code.
        uint32_t cValue = RMIRCodesDecoded.value;

        // if the type of the RMIR code is RC5/6.
        if (cType == RC5 || cType == RC6) {
            // get the length (in bits) of the RMIR code.
            int16_t cLength = RMIRCodesDecoded.bits;

            // calculate the toggle value of the RMIR code.
            uint32_t tValue = cValue ^ (1 << (cLength - 1));

            // search the RMIR codes array.
            for (byte i = 0; i < RMIRCodesNumber; i++) {
                // if the RMIR code or the toggle value of it exists.
                if (RMIRCurrentCodes[i] == cValue ||
                    RMIRCurrentCodes[i] == tValue)
                    // return the index of it.
                    return i;
            }
        }
        else
            // search the RMIR codes array.
            for (byte i = 0; i < RMIRCodesNumber; i++)
                // if the RMIR code exists.
                if (RMIRCurrentCodes[i] == cValue)
                    // return the index of it.
                    return i;
    }

    // RMIR code does not exist.
    return -1;
}

// try to get an RMIR code index.
int8_t
getRMIRCodeIndex ()
{
    // the index of the RMIR code.
    int8_t index;

    do {
        // until an RMIR code is received.
        while (!SYS_RMIR.decode (&RMIRCodesDecoded));
}

```

```

    // disable all external interrupts.
    detachInterruptsISRs ();

    // try to get the index of the RMIR code.
    index = RMIRCodeExists ();

    // now accept the next RMIR code.
    SYS_RMIR.resume ();

    // enable all external interrupts.
    attachInterruptsISRs ();
}

// while the RMIR code does not exist.
while (index < 0);

// return the index of the RMIR code.
return index;
}

// try to debounce the RMIR sensor.
void
debounceRMIR ()
{
    // light the status led.
    digitalWrite (LED_STATUS_PIN, HIGH);

    // wait some time.
    delay (RMIR_CODE_DURATION);

    // dark the status led.
    digitalWrite (LED_STATUS_PIN, LOW);
}

// try to flush any possible data coming from RMIR sensor (hack).
void
RMIRFlush ()
{
    // init / start the RMIR sensor.
    SYS_RMIR.enableIRIn ();
}

// try to learn RMIR keys (printing LCD information).
void
RMIRLearnKeysV ()
{
    SYS_LCD << F ("3) Learning RMIR");
}

// try to learn RMIR keys.
void
RMIRLearnKeysP ()
{
    byte i = 0;           // RMIR codes array index.
    uint32_t oValue = 0;   // previous RMIR code read.
    bool learning = true; // learning process status.

    // while there are unlearned RMIR codes.
    while (i < RMIR_ARRAY_SIZE && true == learning) {
        // if RMIR code is received.
        if (SYS_RMIR.decode (&RMIRCodesDecoded)) {

```

```

// disable all external interrupts.
detachInterruptsISRs ();

// get the type of the RMIR code.
int16_t cType = RMIRCodesDecoded.decode_type;

// if the RMIR code type is not unknown.
if (cType != UNKNOWN) {
    // get the value of the RMIR code.
    uint32_t cValue = RMIRCodesDecoded.value;

    // if this is the first RMIR code.
    if (0 == i) {
        // print information to LCD & hardware serial.

        #if defined (SERIAL_ENABLED)
        Serial << F ("Learning Codes") << endl;
        #endif

        SYS_LCD.clear ();
        SYS_LCD << F (" Learning Codes ");

        // check if the first RMIR code is
        // the previous initialized value.
        if (cValue == oValue)
            // change the previous RMIR code value.
            oValue = ~cValue;
    }

    // if the type of the RMIR code is RC5/6.
    if (cType == RC5 || cType == RC6) {
        // get the length (in bits) of the RMIR code.
        int16_t cLength = RMIRCodesDecoded.bits;

        // toggle value of the previous RMIR code.
        uint32_t tValue = oValue ^ (1 << (cLength - 1));

        // if the current RMIR code is not simular to
        // the previous RMIR code (and toggle value).
        if (cValue != oValue && cValue != tValue) {
            // store the RMIR code.
            RMIRNewCodes[i++] = cValue;

            // refresh the previous RMIR code.
            oValue = cValue;

            // debounce RMIR sensor.
            debounceRMIR ();
        }
        else
            learning = false; // stop learning process.
    }
    else {
        // if the RMIR code is different from the previous RMIR code.
        if (cValue != oValue) {
            // store the RMIR code.
            RMIRNewCodes[i++] = cValue;

```

```

        // refresh the previous RMIR code.
        oValue = cValue;

        // debounce RMIR sensor.
        debounceRMIR ();
    }
    else
        learning = false; // stop learning process.
    }
}
// now accept the next RMIR code.
SYS_RMIR.resume ();

// enable all external interrupts.
attachInterruptsISRs ();
}

// disable all external interrupts.
detachInterruptsISRs ();

// print information to LCD & hardware serial.

#if defined (SERIAL_ENABLED)
Serial << F ("RMIR Codes Saved") << endl;
#endif

SYS_LCD.setCursor (0, 1);
SYS_LCD << F ("RMIR Codes Saved");

// store the number of RMIR codes read.
RMIRCodesNumber = i;

// save the RMIR codes to EEPROM of Arduino.
saveRMIRCodes ();

// load RMIR codes from EEPROM of Arduino.
loadRMIRCodes ();

// flush any possible data coming from RMIR sensor (hack).
RMIRFlush ();

// enable all external interrupts.
attachInterruptsISRs ();
}

// load RMIR codes from EEPROM of Arduino.
void
loadRMIRCodes ()
{
    // read number of real RMIR codes from EEPROM.
    EEPROMGenericRead (0, RMIRCodesNumber);

    // read RMIR codes from EEPROM to current array.
    EEPROMGenericRead (1, RMIRCurrentCodes);
}

// save RMIR codes to EEPROM of Arduino.
void
saveRMIRCodes ()
{
    // write number of real RMIR codes to EEPROM.
    EEPROMGenericWrite (0, RMIRCodesNumber);
}

```

```

// write RMIR codes from new array to EEPROM.
EEPROMGenericWrite (1, RMIRNewCodes);
}
/*
 * define RFID-related functions.
 */

// try to handle / parse an RFID tag.
bool
RFIDTagHandled (bool verbose = true)
{
    byte value = 0;           // temporary data received from RFID reader.
    byte code[6];            // code + checksum data of RFID tag received.
    byte checksum = 0;        // checksum data of RFID tag received.
    byte bytesRead = 0;       // number of received data from RFID reader.
    byte tempByte = 0;        // temporary value for checksum calculation.
    bool handled = false;    // flag indicating if an RFID tag was handled.

    // if there are any data coming from the RFID reader.
    if (SYS_RFID.available () > 0) {
        // disable all external interrupts.
        detachInterruptsISRs ();

        // check for the STX header (0x02 ASCII value).
        if (0x02 == (value = SYS_RFID.read ())) {
            // read the RFID 10-digit code & the 2 digit checksum.
            while (bytesRead < (RFID_CODE_SIZE + 2)) {
                // if there are any data coming from the RFID reader.
                if (SYS_RFID.available () > 0) {
                    // get a byte from the RFID reader.
                    value = SYS_RFID.read ();

                    // check for ETX | STX | CR | LF.
                    if ((0x0D == value) ||
                        (0x0A == value) ||
                        (0x03 == value) ||
                        (0x02 == value)) {
                        // stop reading - there is an error.
                        break;
                    }
                }

                // store the RFID code digits to an array.
                if (bytesRead < RFID_CODE_SIZE)
                    RFIDCode[bytesRead] = value;

                // convert hex tag ID.
                if ((value >= '0') && (value <= '9'))
                    value = value - '0';
                else if ((value >= 'A') && (value <= 'F'))
                    value = 10 + value - 'A';

                // every two hex-digits, add byte to code.
                if (bytesRead & 1 == 1) {
                    // make some space for this hex-digit by shifting
                    // the previous hex-digit with 4 bits to the left.
                    code[bytesRead >> 1] = (value | (tempByte << 4));

                    if (bytesRead >> 1 != 5)
                        // if we're at the checksum byte, calculate the checksum.

```

```

        checksum ^= code[bytesRead >> 1];
    }
    else
        tempByte = value;

        // ready to read next digit.
        bytesRead++;
    }

// handle the RFID 10-digit code & the 2 digit checksum data.
if (bytesRead == (RFID_CODE_SIZE + 2)) {
    // if the caller wants output information.
    if (verbose) {
        // print information to LCD & hardware serial.

        #if defined (SERIAL_ENABLED)
        Serial << F ("RFID: ")
            << RFIDCode
            << F (" CSUM: ")
            << _HEX (code[5])
            << (code[5] == checksum ? F (" (PASS)") :
                F (" (ERROR)")) << endl;
    #endif

    SYS_LCD.clear ();
    SYS_LCD << F ("RFID: ") << RFIDCode;
    SYS_LCD.setCursor (0, 1);
    SYS_LCD << F ("CSUM: ")
        << _HEX (code[5])
        << (code[5] == checksum ? F (" (PASS)") :
            F (" (ERROR)"));
    }
}

// check if the RFID code is correct.
if (code[5] == checksum)
    // set that the tag was handled.
    handled = true;
}

// enable all external interrupts.
attachInterruptsISRs ();
}

return handled;
}

// try to learn an RFID tag (printing LCD information).
void
RFIDLearnTagV ()
{
    SYS_LCD << F ("2) Learning RFID");
}

// try to learn an RFID tag.
void
RFIDLearnTagP ()
{
    // if an RFID tag was handled.
    if (RFIDTagHandled ()) {
        // flush any possible data coming from RMIR sensor (hack).
        RMIRFlush ();
}

```

```

// try to get an RMIR code index.
RMIRCodeIndex = getRMIRCodeIndex ();

// if the RFID code was stored.
if (RFIDCodeStored ()) {
    // print information to LCD & hardware serial.

    #if defined (SERIAL_ENABLED)
    Serial << F ("RFID: ")
        << RFIDCode
        << F (" , IDEV: ")
        << RMIRCodeIndex
        << endl;
    #endif

    SYS_LCD.clear ();
    SYS_LCD << F ("RFID: ") << RFIDCode;
    SYS_LCD.setCursor (0, 1);
    SYS_LCD << F ("IDEV: ") << RMIRCodeIndex;
}

// flush any possible data coming from RFID reader.
SYS_RFID.flush ();
}

// try to store an RFID code to the knowledge file.
bool
RFIDCodeStored ()
{
    // disable all external interrupts.
    detachInterruptsISRs ();

    // flag indicating if the RFID code was stored.
    bool stored = false;

    // try to open the knowledge file for appending.
    if (!RFIDFile.open (&SDRoot,
                       RFID_KNOWLEDGE_FILENAME,
                       O_CREAT | O_APPEND | O_WRITE))
        error ("RFIDFile.open ()");
    else {
        // print information to hardware serial.

        #if defined (SERIAL_ENABLED)
        Serial << F ("Appending: ") << RFID_KNOWLEDGE_FILENAME << endl;
        #endif

        // clear write error.
        RFIDFile.writeError = false;

        // write data to the file.
        RFIDFile << RFIDCode << F (" ");
        if (RMIRCodeIndex < 10) RFIDFile << F ("0");
        RFIDFile << RMIRCodeIndex << endl;

        // check if there was a write error.
        if (RFIDFile.writeError)
            error ("RFIDFile.write ()");

        // try to close the file.
        if (!RFIDFile.close ())

```

```

        error("RFIDFile.close ()");

    // set that the code was stored.
    stored = true;
}

// enable all external interrupts.
attachInterruptsISRs ();

return stored;
}

// try to get the device number of an RFID code.
bool
getRFIDCodeDevice (int8_t & iDev)
{
    // disable all external interrupts.
    detachInterruptsISRs ();

    const byte bufSize = 13; // the size of the data buffer.
    byte bytesRead = 0;      // the number of received data.

    // the buffer of received data (+1 for string termination).
    char bufData[bufSize + 1] = { '\0' };

    // assume that the RFID code does not exist.
    iDev = -1;

    // try to open the RFID knowledge file for reading.
    if (!RFIDFile.open (&SDRoot, RFID_KNOWLEDGE_FILENAME, O_READ))
        return false; // file cannot open or does not exist.
    else {
        // keep reading records from the RFID knowledge file.
        while (true) {
            // try to read a record from the file.
            bytesRead = RFIDFile.read (bufData, bufSize);

            // if we reached EOF break the loop.
            if (0 == bytesRead) break;

            // if there was a read error.
            if (bytesRead != bufSize)
                error ("RFIDFile.read ()");

            // check if the RFID code is the one in the current record.
            if (strncmp (RFIDCode, bufData, RFID_CODE_SIZE) == 0)
                // get the device number of the RFID code.
                iDev = atoi (bufData + RFID_CODE_SIZE + 1);

            // try to ignore `endl` characters.

            RFIDFile.read (); // '\r'.
            RFIDFile.read (); // '\n'.
        }

        // try to close the file.
        if (!RFIDFile.close ())
            error ("RFIDFile.close ()");
    }
}

```

```

// enable all external interrupts.
attachInterruptsISRs ();

// the RFID knowledge file exists.
return true;
}

// try to handle an RFID tag (printing LCD information).
void
RFIDHandleTagV ()
{
    SYS_LCD << F ("1) Handling RFID");
}

// try to handle an RFID tag.
void
RFIDHandleTagP ()
{
    // the device number of the RFID code.
    int8_t iDev;

    // if an RFID tag was handled (work silently).
    if (RFIDTagHandled (false)) {
        // clear the LCD display.
        SYS_LCD.clear ();

        // try to get the device number of the RFID code.
        if (getRFIDCodeDevice (iDev)) {
            // if the RFID code exists (checking the device number).
            if (iDev >= 0) {

                // print information to LCD & hardware serial.
                #if defined (SERIAL_ENABLED)
                Serial << F ("RFID: ")
                    << RFIDCode
                    << F (", IDEV: ")
                    << iDev
                    << endl;
                #endif

                SYS_LCD << F ("RFID: ") << RFIDCode;
                SYS_LCD.setCursor (0, 1);
                SYS_LCD << F ("IDEV: ") << iDev;

                // try to send the device number with
                // a separator to remote network node.
                SYS_XBEE << iDev << XBEEDataSeparator;
            }
            else {
                // print information to LCD & hardware serial.

                #if defined (SERIAL_ENABLED)
                Serial << F ("Wrong RFID Tag") << endl;
                #endif

                SYS_LCD << F (" Wrong RFID Tag ");
            }
        }
    }
}

```

```

        }
    else {
        // print information to LCD & hardware serial.

        #if defined (SERIAL_ENABLED)
        Serial << F ("No RFID File") << endl;
        #endif

        SYS_LCD << F ("  No RFID File  ");
    }
}

// do nothing - idling function (printing LCD information).
void
idleV ()
{
    SYS_LCD << F ("Wi-Remote & RFID");
    SYS_LCD.setCursor (0, 1);
    SYS_LCD << F ("V1.0 (GNU / GPL)");
}

// do nothing - idling function.
void idleP () {}

/*
 * define utilities functions.
 */

// try to print to hardware serial a
// string from Arduino flash memory.
void
printFString (PGM_P str)
{
    #if defined (SERIAL_ENABLED)

        // read the characters of the string from flash memory.
        for (byte c; c = pgm_read_byte (str); str++)
            // print the current character to hardware serial.
            Serial << c;

    #endif
}

// try to perform a "kernel" panic where the system needs reset.
void
systemError (const char *str)
{
    // disable all external interrupts.
    detachInterruptsISRs ();

    // clear the LCD display.
    SYS_LCD.clear ();

    // print error message to LCD.
    SYS_LCD << F ("  System Error  ");
}

```

```

#ifndef SERIAL_ENABLED
// print error message to hardware serial.
Serial << F ("System Error: ");
printFString (str);
Serial << endl;

// if there was an error with microSD.
if (SDCard.errorCode ())
    // print microSD error message to hardware serial.
    Serial << F ("MicroSD Error: (")
        << _HEX (SDCard.errorCode ())
        << F (", ")
        << _HEX (SDCard.errorData ())
        << F ("")
        << endl;
#endif

// system needs hardware reset.
while (true);

// code never reaches here.
}

/*
 * define setup & loop functions.
 */

// startup point entry (runs once).
void
setup ()
{
    #if defined (SERIAL_ENABLED)
    // set hardware serial port data rate.
    Serial.begin (USB_BAUD_RATE);
    #endif

    // set RFID serial port data rate.
    SYS_RFID.begin (RFID_BAUD_RATE);

    // set XBEE serial port data rate.
    SYS_XBEE.begin (XBEE_BAUD_RATE);

    // set state button PIN as input.
    pinMode (BUT_STATES_PIN, INPUT);

    // set status led PIN as output.
    pinMode (LED_STATUS_PIN, OUTPUT);

    // set default chip select PIN as output.
    pinMode (DEFAULT_CS_PIN, OUTPUT);

    // start RMIR sensor.
    SYS_RMIR.enableIRIn ();

    // initialize the SD card at SPI_HALF_SPEED to avoid bus errors
    // with breadboards. use SPI_FULL_SPEED for better performance.
    if (!SDCard.init (SPI_HALF_SPEED))
        error ("SDCard.init ()");

    // initialize a FAT volume.
    if (!SDVolume.init (&SDCard))

```

```

    error ("SDVolume.init ()");

    // open the root directory.
    if (!SDRoot.openRoot (&SDVolume))
        error ("SDRoot.openRoot ()");

    // load RMIR codes from EEPROM of Arduino.
    loadRMIRCodes ();

    // display application information.
    idleV ();

    // enable all external ISR.
    attachInterruptsISRs ();
}

```

```

// loop the main sketch.
void
loop ()
{
    // save the environment of the calling function.
    setjmp (bufEnv);

    // perform a state action according to the FSM.
    FSMStatesFunctions[FSMCurrentState][0] ();
}

```

Πρόγραμμα 3 : Βασικό υλικολογισμικό του κόμβου ταυτοποίησης

Παρακάτω, ακολουθεί ο πηγαίος κώδικας με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ελέγχου :

```

/*
 * Remote Systems Control With Wireless Ad-Hoc Radio Network And RFID.
 *
 * This sketch implements the firmware of the control node.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

/*
 * include external libraries headers.
 */

```

```

// library for creating software serial ports.
#include <NewSoftSerial.h>

/*
 * define Arduino I/O PINS constants.
 */

// the PIN number for the XBEE reader serial TX line.
const byte XBEE_TLINE_PIN = 2;

// the PIN number for the XBEE reader serial RX line.
const byte XBEE_RLINE_PIN = 3;

// start Arduino I/O PINS (Board: Duemilanove).

// array for the digital I/O standard logic PINS.
const byte IODigitalLogicPINS[] = {
  4, 5, 6, 7, 8, 9, 10, 11, 12, 13 // ignore: 0/1 (Serial), 2/3 (XBEE) .
};

// end Arduino I/O PINS (Board: Duemilanove).

// number of the digital I/O standard logic PINS.
const byte IO_DIGITAL_LOGIC_PINS_NUM =
  sizeof (IODigitalLogicPINS) / sizeof (const byte);

/*
 * define Arduino I/O serial port constants.
 */

// XBEE software serial port baud rate.
const unsigned long XBEE_BAUD_RATE = 9600;

/*
 * define XBEE-related variables.
 */

// object to control the XBEE software serial port.
NewSoftSerial SYS_XBEE = NewSoftSerial (XBEE_RLINE_PIN, XBEE_TLINE_PIN);

// define XBEE data separator token.
const char XBEEDataSeparator = '!';

/*
 * define XBEE-related functions.
 */

// try to handle / parse a decimal number from XBEE.
int8_t
getXBEEDecimalNumber () {
  int8_t value = -1; // the decimal number fetched from XBEE.
  char ch;           // the temporary character read from XBEE.

  // if there are any data coming from the XBEE.
  if (SYS_XBEE.available () > 0) {
    // keep reading characters until
    // the separator token is found.
    do {
      // fetch a new character from XBEE.
      ch = SYS_XBEE.read ();

      // if the character is a decimal digit.

```

```

if (ch >= '0' && ch <= '9') {
    // if the first decimal digit is fetched now.
    if (value == -1) value = 0;

        // recalculate the value of the decimal number.
        value = value * 10 + ch - '0';
    }
} while (ch != XBEEDataSeparator);

// return the decimal number.
return value;
}

/*
 * define setup & loop functions.
 */

// startup point entry (runs once).
void
setup ()
{
    // set XBEE serial port data rate.
    SYS_XBEE.begin (XBEE_BAUD_RATE);

    // set digital standard logic PINS as output.
    for (byte p = 0; p < IO_DIGITAL_LOGIC_PINS_NUM; ++p)
        pinMode (IODigitalLogicPINS[p], OUTPUT);
}

// loop the main sketch.
void
loop ()
{
    // try to get a device number from the remote network node.
    int8_t idev = getXBEEDecimalNumber ();

    // if the device number is supported in the system.
    if (idev >= 0 && idev < IO_DIGITAL_LOGIC_PINS_NUM) {
        // try to get the value of the appropriate pin of the device.
        byte value = digitalRead (IODigitalLogicPINS[idev]);

        // write on the pin of the device the inversed value.
        digitalWrite (IODigitalLogicPINS[idev], !value);
    }
}

```

Πρόγραμμα 4 : Βασικό υλικολογισμικό του κόμβου ελέγχου

Παρακάτω, ακολουθεί ο πηγαίος κώδικας με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) της βιβλιοθήκης που χρησιμοποιείται για την διαχείριση της μνήμης EEPROM μιας πλατφόρμας Arduino :

```
/*
 * Remote Systems Control With Wireless Ad-Hoc Radio Network And RFID.
 *
 * This file implements generic tools for handling the Arduino EEPROM.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

#ifndef EEPROM_GENERIC_H
#define EEPROM_GENERIC_H

/*
 * include external libraries headers.
 */

// include necessary header for libraries.
#include <WProgram.h>

/*
 * define Arduino EEPROM read / write functions.
 */

// write to Arduino EEPROM a data structure.
template <class T> int
EEPROMGenericWrite (int addr, const T & value)
{
    const byte *p = (const byte *) (const void *) &value;

    int i;
    for (i = 0; i < sizeof (value); i++)
        EEPROM.write (addr++, *p++);

    return i;
}

// read from Arduino EEPROM a data structure.
template <class T> int
EEPROMGenericRead (int addr, T & value)
{
    byte *p = (byte *) (void *) &value;
```

```

int i;
for (i = 0; i < sizeof (value); i++)
    *p++ = EEPROM.read (addr++);

return i;
}

#endif // EEPROM_GENERIC_H

```

Πρόγραμμα 5 : Βιβλιοθήκη χειρισμού μνήμης EEPROM

Παρακάτω, ακολουθεί ο πηγαίος κώδικας με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) του υλικολογισμικού που χρησιμοποιείται για την λειτουργία εκκαθάρισης της μνήμης microSD του ενσωματωμένου κόμβου ταυτοποίησης :

```

/*
 * Remote Systems Control With Wireless Ad-Hoc Radio Network And RFID.
 *
 * This sketch removes the RFID knowledge of the identification node.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

/*
 * include external libraries headers.
 */

// library for using output streaming operators.
#include <Streaming.h>

// library for handling flash memory of Arduino.
#include <Flash.h>

// library for using SD storage card devices.
#include <SdFat.h>

/*
 * define preprocessor constants.
 */

// set true / false for debug support.
#define DEBUG_STATE true

```

```

// define controllers if debug state is on.
#if defined (DEBUG_STATE) && true == DEBUG_STATE
    // for hardware serial verbose printing.
    #define SERIAL_ENABLED
#endif

/*
 * define function-like macros.
 */

// macro for performing a "kernel" panic state and
// printing an error from the Arduino flash memory.
#define error(s) systemError (PSTR (s))

/*
 * define Arduino I/O PINS constants.
 */

// the default hardware chip select PIN number.
const byte DEFAULT_CS_PIN = 10;

/*
 * define Arduino I/O serial port constants.
 */

#if defined (SERIAL_ENABLED)
// hardware serial port baud rate.
const unsigned long USB_BAUD_RATE = 9600;
#endif

/*
 * define microSD-related variables.
 */

// object to control the microSD card of the microSD shield.
Sd2Card SDCard;

// object to control the volume of the microSD card.
SdVolume SDVolume;

// object to control the root directory of the microSD card.
SdFile SDRoot;

// object to control the RFID knowledge file of the microSD card.
SdFile RFIDFile;

/*
 * define RFID-related variables.
 */

// RFID knowledge filename (8.3 filename format).
const char RFID_KNOWLEDGE_FILENAME[] = "rfids.txt";

/*
 * define utilities functions.
 */

// try to print to hardware serial a
// string from Arduino flash memory.
void
printFString (PGM_P str)
{

```

```

#if defined (SERIAL_ENABLED)

// read the characters of the string from flash memory.
for (uint8_t c; c = pgm_read_byte (str); str++)
    // print the current character to hardware serial.
    Serial << c;

#endif
}

// try to perform a "kernel" panic where the system needs reset.
void
systemError (const char *str)
{
    #if defined (SERIAL_ENABLED)
    // print error message to hardware serial.
    Serial << F ("System Error: ");
    printFString (str);
    Serial << endl;

    // if there was an error with microSD.
    if (SDCard.errorCode ())
        // print microSD error message to hardware serial.
        Serial << F ("MicroSD Error: () << _HEX (SDCard.errorCode ())
            << F (", ") << _HEX (SDCard.errorData ())
            << F (")") << endl;
    #endif

    // system needs hardware reset.
    while (true);

    // code never reaches here.
}

```

```

/*
 * define setup & loop functions.
 */

// startup point entry (runs once).
void
setup ()
{
    #if defined (SERIAL_ENABLED)
    // set hardware serial port data rate.
    Serial.begin (USB_BAUD_RATE);
    #endif

    // set the default chip select PIN as output.
    pinMode (DEFAULT_CS_PIN, OUTPUT);

    // initialize the SD card at SPI_HALF_SPEED to avoid bus errors
    // with breadboards. use SPI_FULL_SPEED for better performance.
    if (!SDCard.init (SPI_HALF_SPEED))
        error ("SDCard.init ()");

    // initialize a FAT volume.

```

```

if (!SDVolume.init (&SDCard))
    error ("SDVolume.init ()");

// open the root directory.
if (!SDRoot.openRoot (&SDVolume))
    error ("SDRoot.openRoot ()");

// open the RFID knowledge file.
if (!RFIDFile.open (&SDRoot, RFID_KNOWLEDGE_FILENAME, O_WRITE))
    error ("RFIDFile.open ()");

// remove the RFID knowledge file.
if (!RFIDFile.remove ())
    error ("RFIDFile.remove ()");

#if defined (SERIAL_ENABLED)
// print remove information to hardware serial.
Serial << F ("RFID knowledge removed.") << endl;
#endif
}

// loop the main sketch.
void
loop ()
{
    // do nothing - idling function.
}

```

Πρόγραμμα 6 : Υλικολογισμικό καθαρισμού μνήμης microSD

Παρακάτω, ακολουθεί ο πηγαίος κώδικας με ικανοποιητικά σχόλια (στην Αγγλική γλώσσα) του υλικολογισμικού που χρησιμοποιείται για την λειτουργία εκκαθάρισης της μνήμης EEPROM του ενσωματωμένου κόμβου ταυτοποίησης :

```
/*
 * Remote Systems Control With Wireless Ad-Hoc Radio Network And RFID.
 *
 * This sketch clears the EEPROM memory of an Arduino.
 *
 * Copyright (C) 2011 Efstatios Chatzikyriakidis (contact@efxa.org)
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

/*
 * include external libraries headers.
 */

// library for handling EEPROM memory of Arduino.
#include <EEPROM.h>

/*
 * define Arduino I/O PINS constants.
 */

// the PIN number for the status led.
const byte LED_STATUS_PIN = 13;

/*
 * define Arduino EEPROM constants.
 */

// the capacity (in bytes) of the EEPROM.
const word EEPROM_SIZE = 1024;

/*
 * define setup & loop functions.
 */

// startup point entry (runs once).
void
setup ()
{
    // set the status led PIN as output.
    pinMode (LED_STATUS_PIN, OUTPUT);

    // dark the status led.
    digitalWrite (LED_STATUS_PIN, LOW);

    // clear the EEPROM memory.
    for (word i = 0; i < EEPROM_SIZE; i++)
}
```

```
EEPROM.write (i, 0);

// light the status led.
digitalWrite (LED_STATUS_PIN, HIGH);
}

// loop the main sketch.
void
loop ()
{
    // do nothing - idling function.
}
```

Πρόγραμμα 7 : Υλικολογισμικό καθαρισμού μνήμης EEPROM

Ευρετήριο Εικόνων

Εικόνα 01 : Ασύρματα ελεγχόμενο επίγειο όχημα υλοποιημένο με Arduino.....	8
Εικόνα 02 : Η βασική ομάδα ανάπτυξης της πλατφόρμας Arduino.....	9
Εικόνα 03 : Η κάτωψη της πλατφόρμας Arduino Uno.....	10
Εικόνα 04 : Μία πλατφόρμα Arduino με κάρτες επέκτασης (Ethernet και SD)	11
Εικόνα 05 : Η κάτωψη της πλατφόρμας Arduino DueMilanove.....	14
Εικόνα 06 : Το ολοκληρωμένο περιβάλλον ανάπτυξης Arduino.....	26
Εικόνα 07 : Σειριακή επικοινωνία με μία πλατφόρμα Arduino.....	27
Εικόνα 08 : Τυπικό ράστερ για την κατασκευή ηλεκτρονικών κυκλωμάτων.....	33
Εικόνα 09 : Δημιουργία μιας τυπικής πλατφόρμας Arduino στο ράστερ.....	34
Εικόνα 10 : Πολύχρωμα καλώδια μεγέθους 22 AWG.....	35
Εικόνα 11 : Μία κάρτα μνήμης microSD χωρητικότητας 2GB.....	36
Εικόνα 12 : Η κάρτα επέκτασης (κόκκινη) microSD (SparkFun).....	37
Εικόνα 13 : Ο ανιχνευτής υπερύθρων PNA4602 (Panasonic).....	38
Εικόνα 14 : Συνδεσμολογία του ανιχνευτή υπερύθρων PNA4602 (Panasonic)	39
Εικόνα 15 : Η κάρτα επέκτασης αισθητήρων μαζί με έναν αισθητήρα PIR.....	40
Εικόνα 16 : Ένας ηλεκτρομηχανικός ρελές (FlamingoEDA).....	41
Εικόνα 17 : Κάρτα ασύρματης ραδιοεπικοινωνίας XBee (Digi International)....	42
Εικόνα 18 : Η κάρτα προσαρμογής XBee Explorer Regulated (SparkFun).....	43
Εικόνα 19 : Η κάρτα προσαρμογής XBee Explorer USB (SparkFun).....	44
Εικόνα 20 : Η οθόνη υγρών κρυστάλλων GDM1602K (Xiamen Ocular).....	45
Εικόνα 21 : Οι επαφές του καταχωρητή ολίσθησης DM74LS164N.....	47
Εικόνα 22 : Ο καταχωρητής ολίσθησης HD74LS164P.....	48
Εικόνα 23 : Μαύρο καλώδιο USB 2.0 τύπου A-B (αρσενικό σε αρσενικό).....	49
Εικόνα 24 : Τροφοδοτικό τοίχου 5V με USB υποδοχή τύπου A.....	50
Εικόνα 25 : Η συσκευή ανάγνωσης RFID ID-12 (SparkFun).....	51
Εικόνα 26 : Η USB συσκευή ανάγνωσης RFID (SparkFun).....	52
Εικόνα 27 : Παθητική ετικέτα RFID (πλαστική λευκή κάρτα).....	53
Εικόνα 28 : Παθητική ετικέτα RFID (πλαστικός μαύρος δίσκος).....	53
Εικόνα 29 : Παθητική ετικέτα RFID (γυάλινη διαφανή κάψουλα).....	54
Εικόνα 30 : Η ηλεκτρική δίοδος 1N4001.....	55
Εικόνα 31 : Ένα τυπικό ηλεκτρομηχανικό ποτενσιόμετρο 10ΚΩ.....	56
Εικόνα 32 : Ένας ηλεκτρικός αντιστάτης 1ΚΩ.....	57
Εικόνα 33 : Ένα ηλεκτρομηχανικό κουμπί.....	58
Εικόνα 34 : Πέντε δίοδοι εκπομπής φωτός κόκκινου χρώματος.....	59

Ευρετήριο Πινάκων

Πίνακας 1 : Βασικά χαρακτηριστικά υλικού τυπικών πλατφορμών Arduino.....	12
Πίνακας 2 : Αντιστοίχηση μεταξύ επαφών ATmega168/328 και Arduino.....	18
Πίνακας 3 : Επαφές οθόνης υγρών κρυστάλλων GDM1602K (Xiamen Ocular)..	46
Πίνακας 4 : Η δομή του πακέτου μετάδοσης των παθητικών ετικετών RFID...	54

Ευρετήριο Προγραμμάτων

Πρόγραμμα 1 : Αναλογικό αναβόσβησμα μίας διόδου εκπομπής φωτός.....	31
Πρόγραμμα 2 : Αντιστροφή αλφαριθμητικού με χρήση στοίβας.....	32
Πρόγραμμα 3 : Βασικό υλικολογισμικό του κόμβου ταυτοποίησης.....	79
Πρόγραμμα 4 : Βασικό υλικολογισμικό του κόμβου ελέγχου.....	81
Πρόγραμμα 5 : Βιβλιοθήκη χειρισμού μνήμης EEPROM.....	83
Πρόγραμμα 6 : Υλικολογισμικό καθαρισμού μνήμης microSD.....	86
Πρόγραμμα 7 : Υλικολογισμικό καθαρισμού μνήμης EEPROM.....	88

Βιβλιογραφία

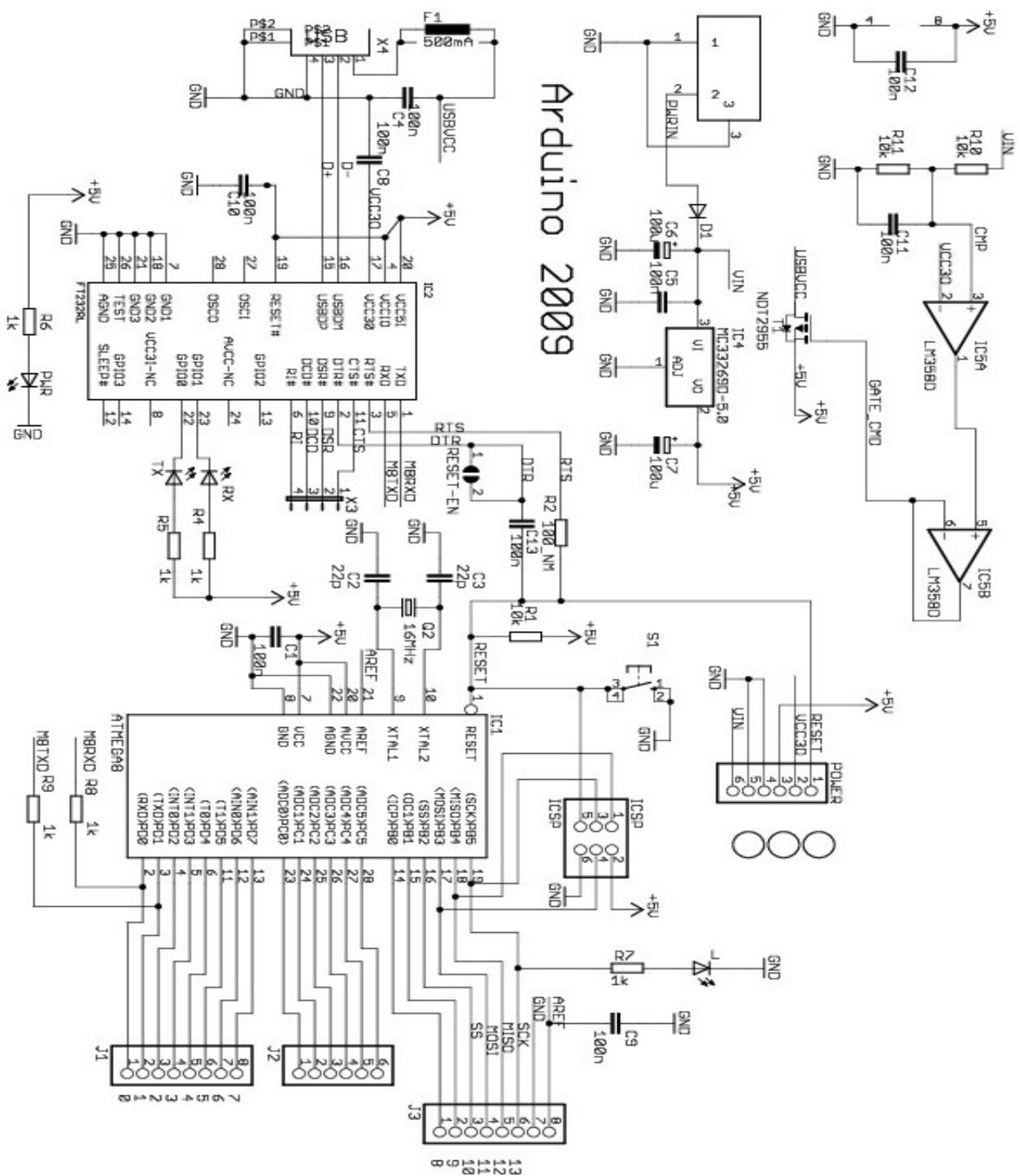
Βιβλία

- 1)** Michael Margolis - Arduino Cookbook
O'Reilly Media, 2011
- 2)** Massimo Banzi - Getting Started with Arduino
O'Reilly Media / Make, 2008
- 3)** Brian W. Evans - Arduino Programming Notebook
Second Edition, 2008
- 4)** Tom Igoe - Making Things Talk
O'Reilly Media / Make, 2007
- 5)** Donald P. Leach, Albert Paul Malvino - ΨΗΦΙΑΚΑ ΗΛΕΚΤΡΟΝΙΚΑ
Πέμπτη Έκδοση, ΤΖΙΟΛΑ, 2006
- 6)** Brian W. Kernighan, Rob Pike - The Practice of Programming
Addison-Wesley, 1999
- 7)** Bjarne Stroustrup - The C++ Programming Language
Third Edition, Addison-Wesley, 1997
- 8)** Brian W. Kernighan, Dennis M. Ritchie - The C Programming Language
Second Edition, Prentice Hall Software Series, 1988

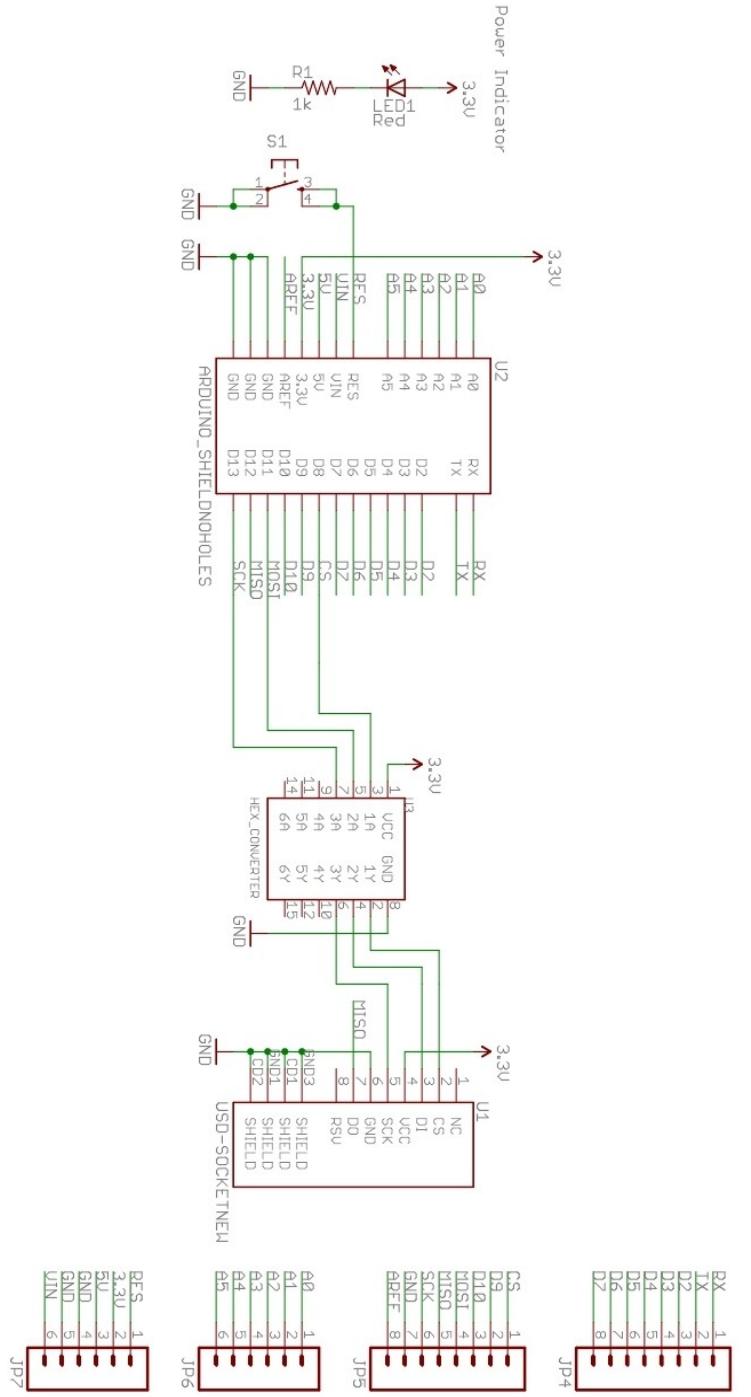
Ιστοσελίδες

- 1)** <http://www.arduino.cc/>
- 2)** <http://www.sparkfun.com/>
- 3)** <http://www.adafruit.com/>
- 4)** <http://www.makershed.com/>
- 5)** <http://www.instructables.com/>
- 6)** <http://www.wikipedia.org/>
- 7)** <http://directory.fsf.org/>
- 8)** <http://www.ieee.org/>

Παράρτημα 1 - Κυκλώματα Συστατικών



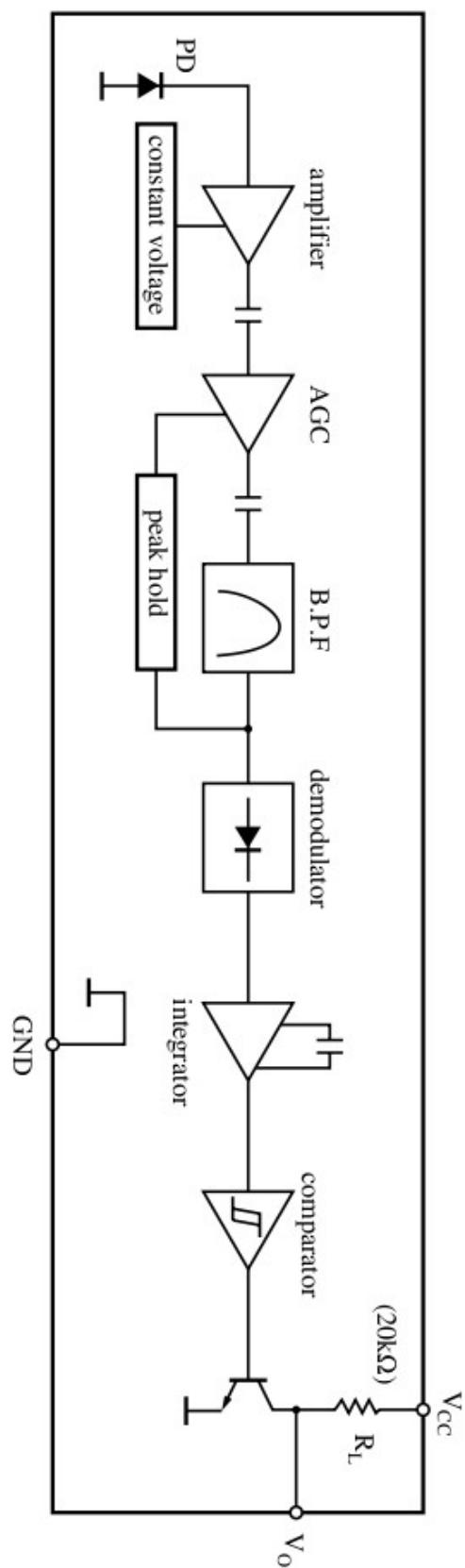
Κύκλωμα 1 : Το κύκλωμα της πλατφόρμας Arduino Duemilanove



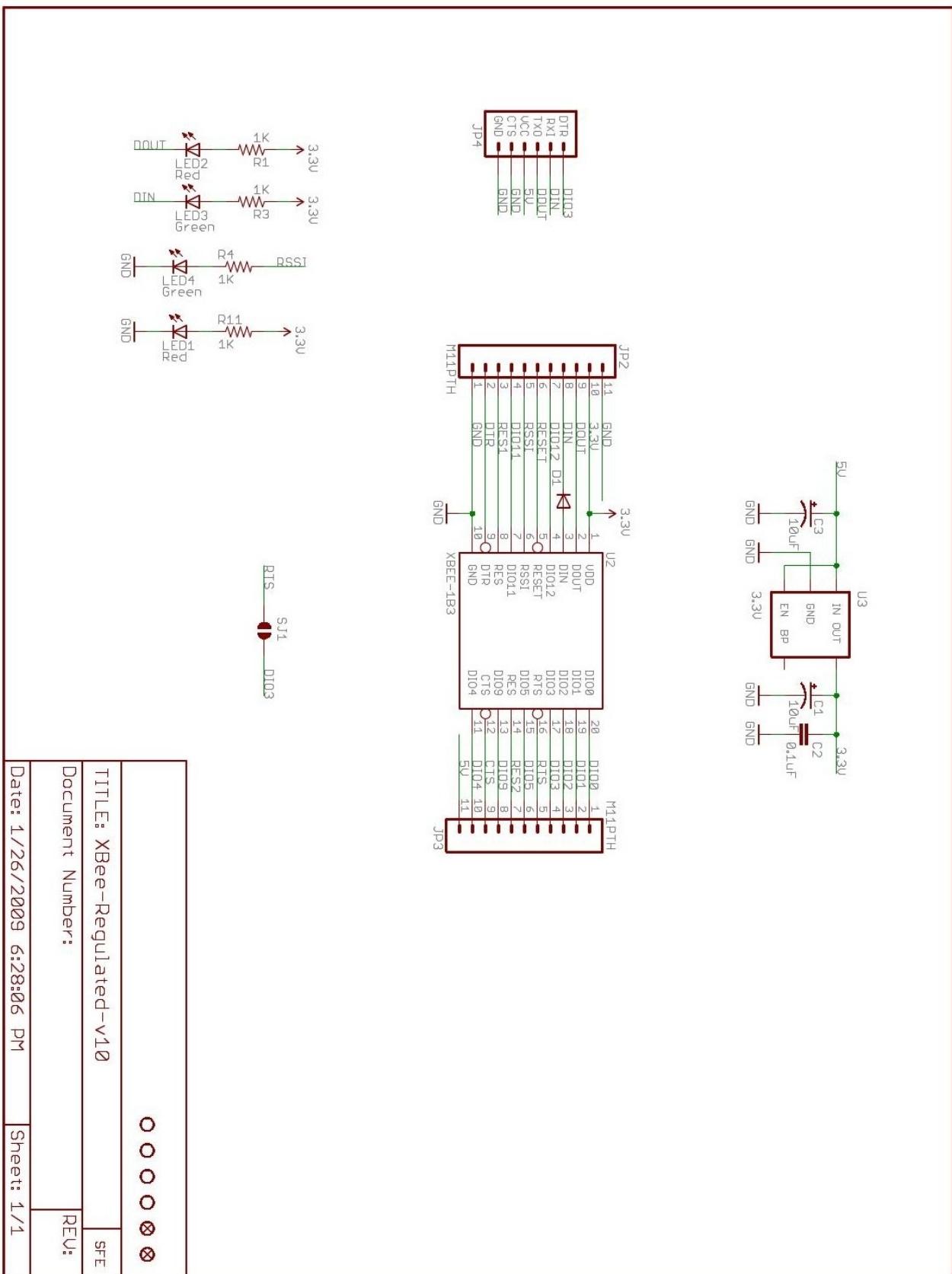
Κύκλωμα 2 : Το κύκλωμα της κάρτας επέκτασης microSD (SparkFun)

Released under the Creative Commons Attribution Share-Alike 3.0 License
<http://creativecommons.org/licenses/by-sa/3.0>
Design by Ryan Quens

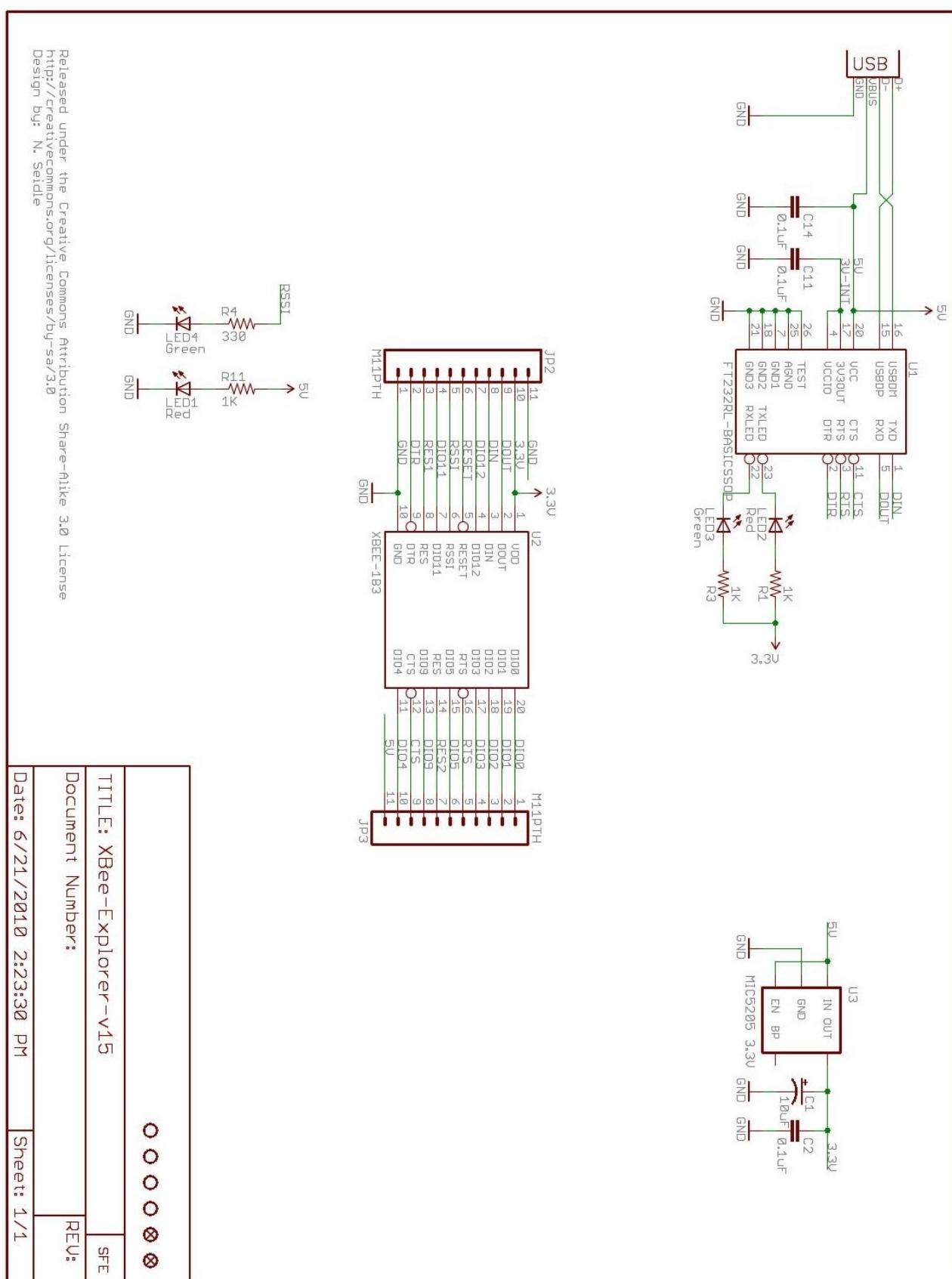
TITLE: microSD_Shield-v13	
Document Number:	REV:
Date: not saved!	Sheet: 1/1

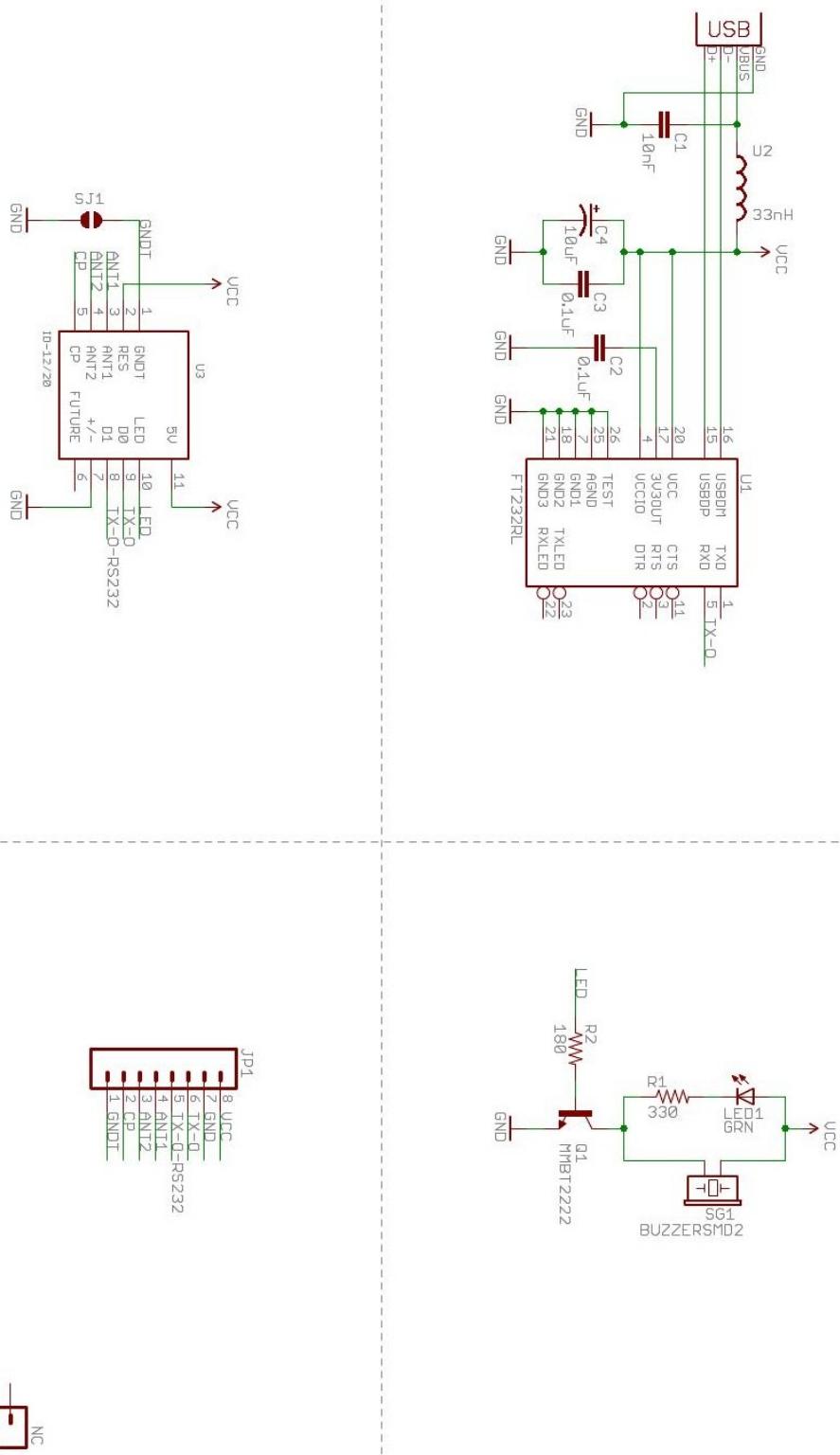


Κύκλωμα 3 : Το κύκλωμα του ανιχνευτή υπερύθρων PNA4602 (Panasonic)



Κύκλωμα 4 : Το κύκλωμα της κάρτας XBee Explorer Regulated (SparkFun)

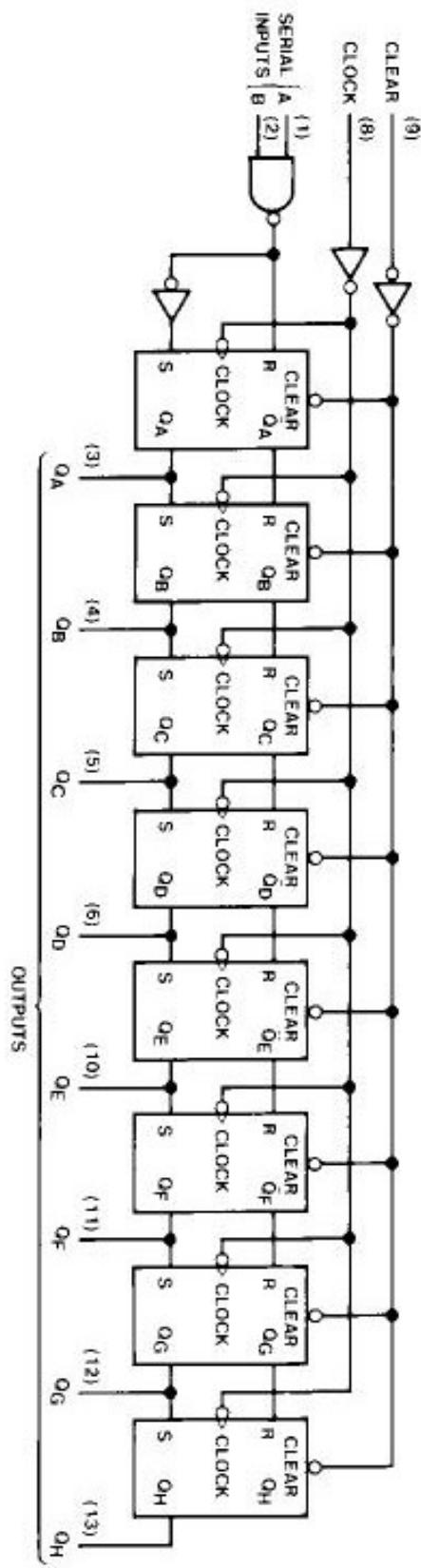




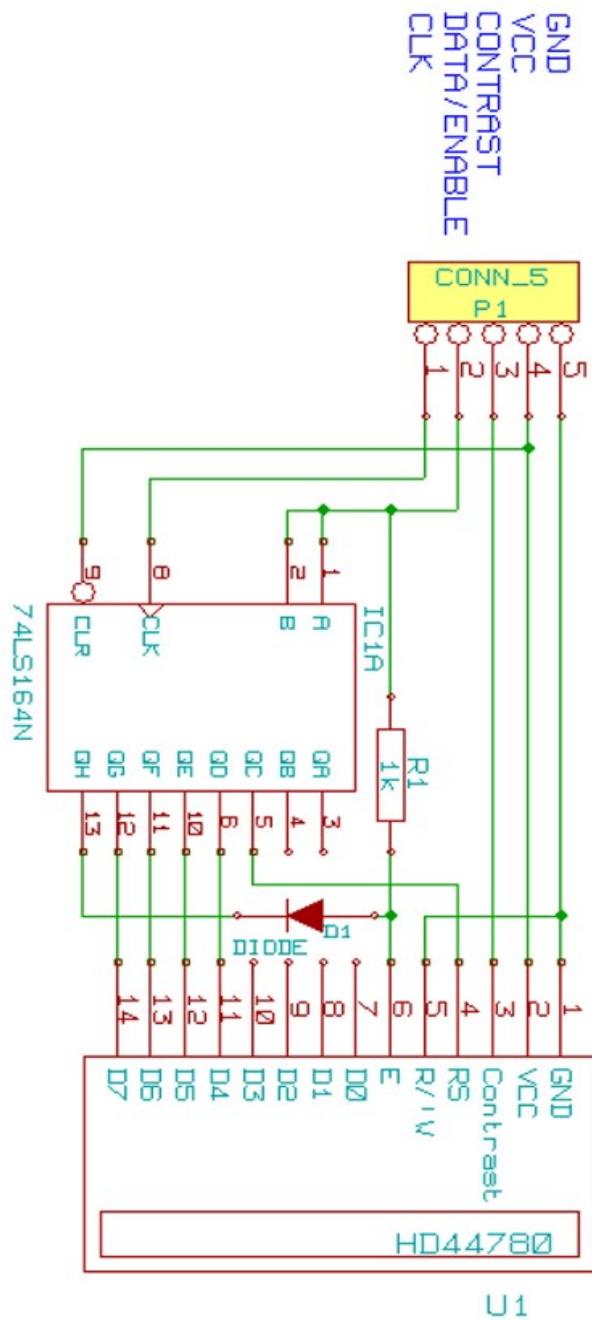
Released under the Creative Commons Attribution Share-alike 3.0 License
<http://creativecommons.org/licenses/by-sa/3.0/>
Design by: [Fluweiss](#)

○○○○
TITLE: RFID_USB_Reader-v14 SFE SFE
Document Number: REU:
Date: 7/14/2009 2:46:49 PM Sheet: 1/1

Κύκλωμα 6 : Το κύκλωμα της USB συσκευής ανάγνωσης RFID (SparkFun)



Κύκλωμα 7 : Το κύκλωμα του καταχωρητή ολίσθησης DM74LS164N

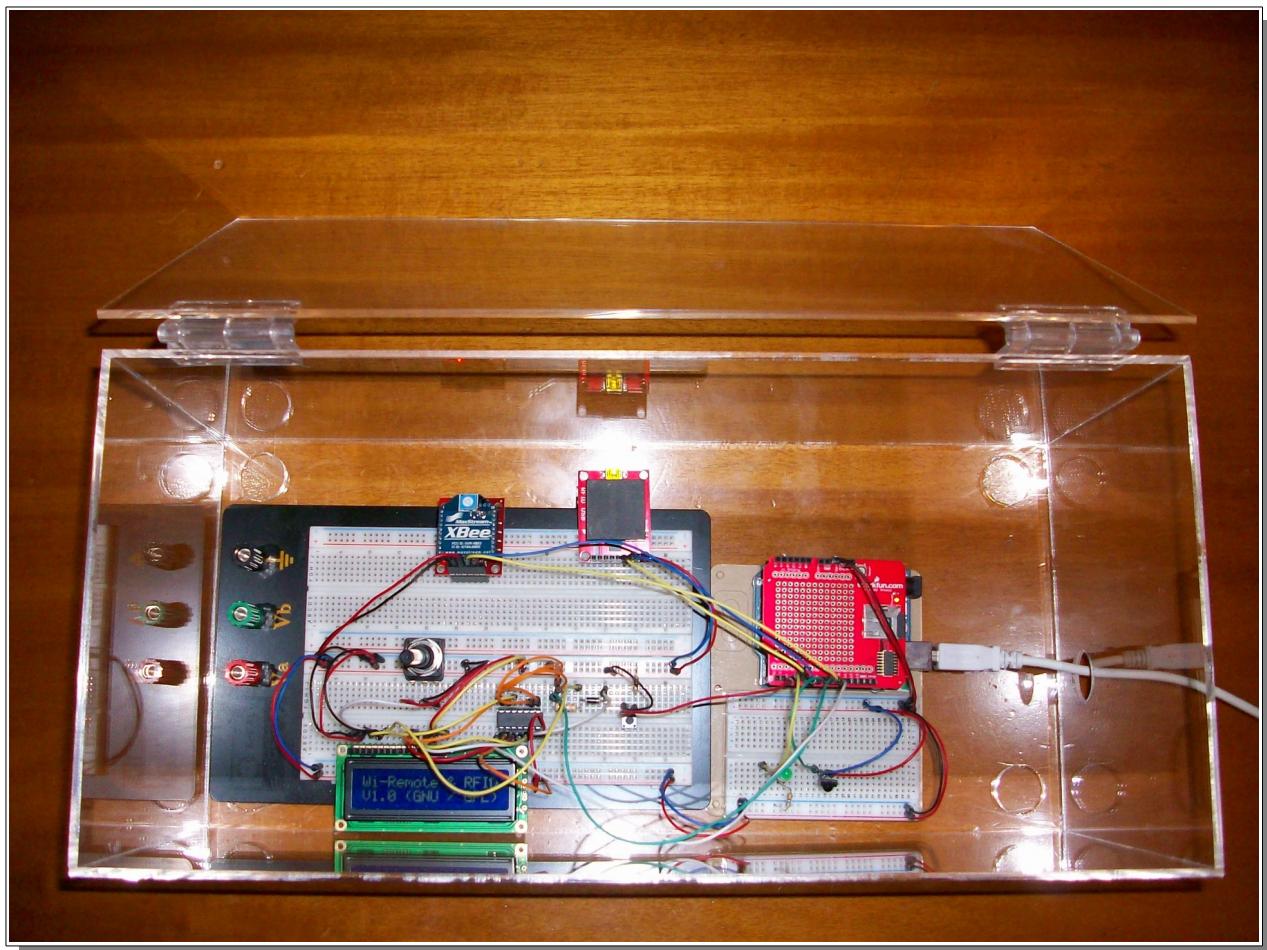


Κύκλωμα 8 : Σύνδεση οθόνης GDM1602K και καταχωρητή DM74LS164N

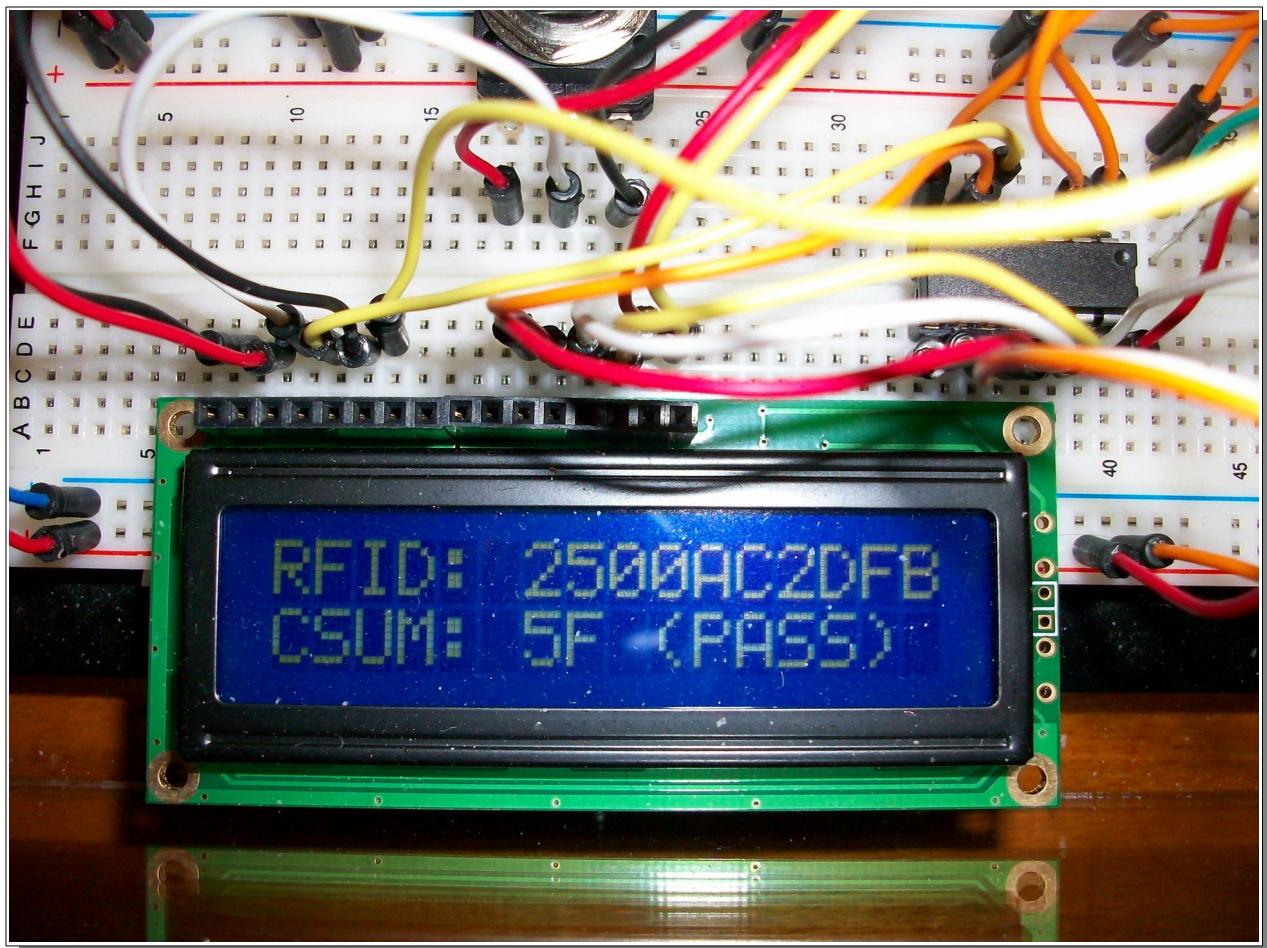
Παράρτημα 2 - Φωτογραφίες Συστήματος



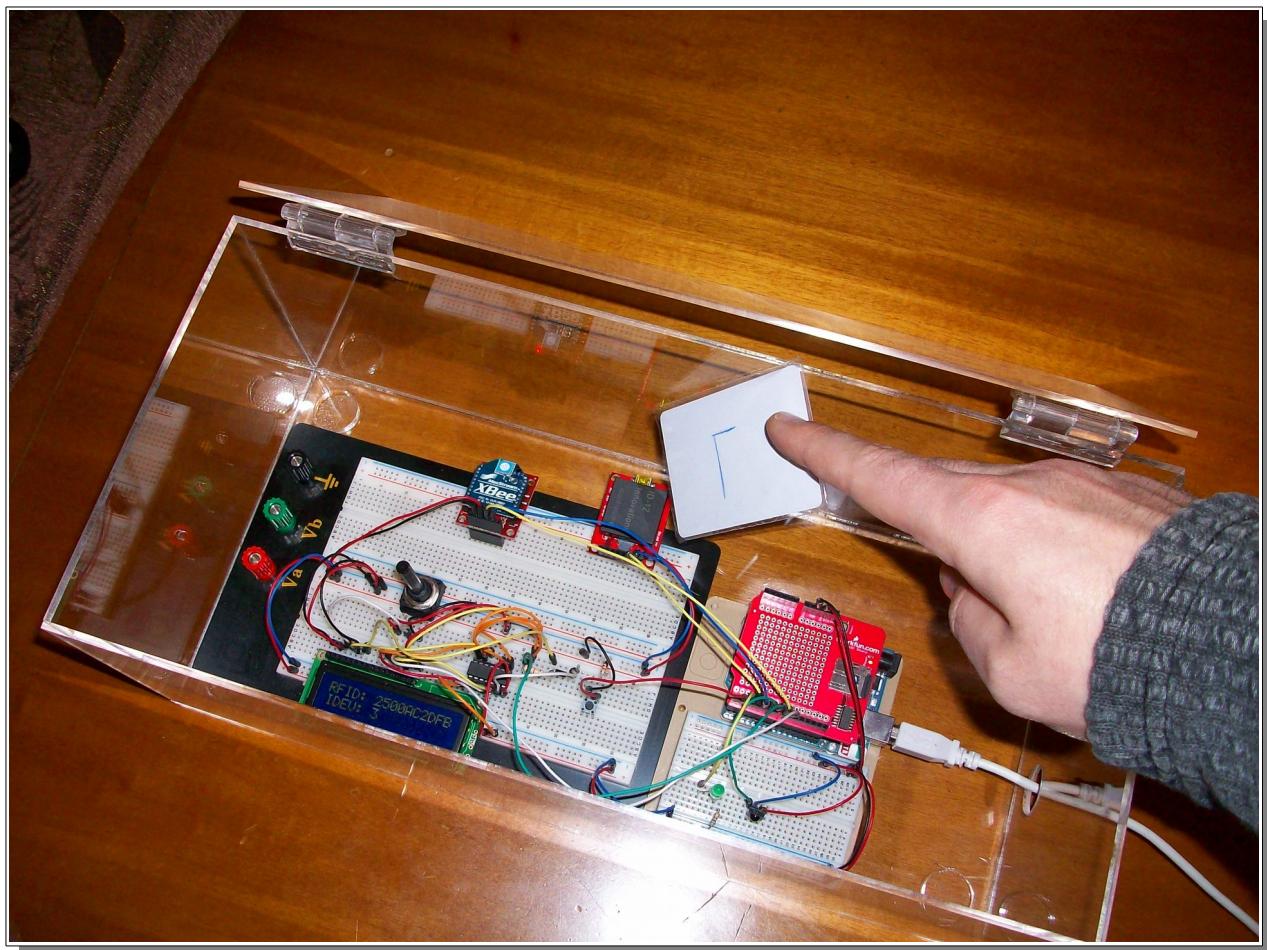
Φωτογραφία 1 : Ο κόμβος ταυτοποίησης με τα λειτουργικά του μέρη



Φωτογραφία 2 : Η κάτοψη του κόμβου ταυτοποίησης



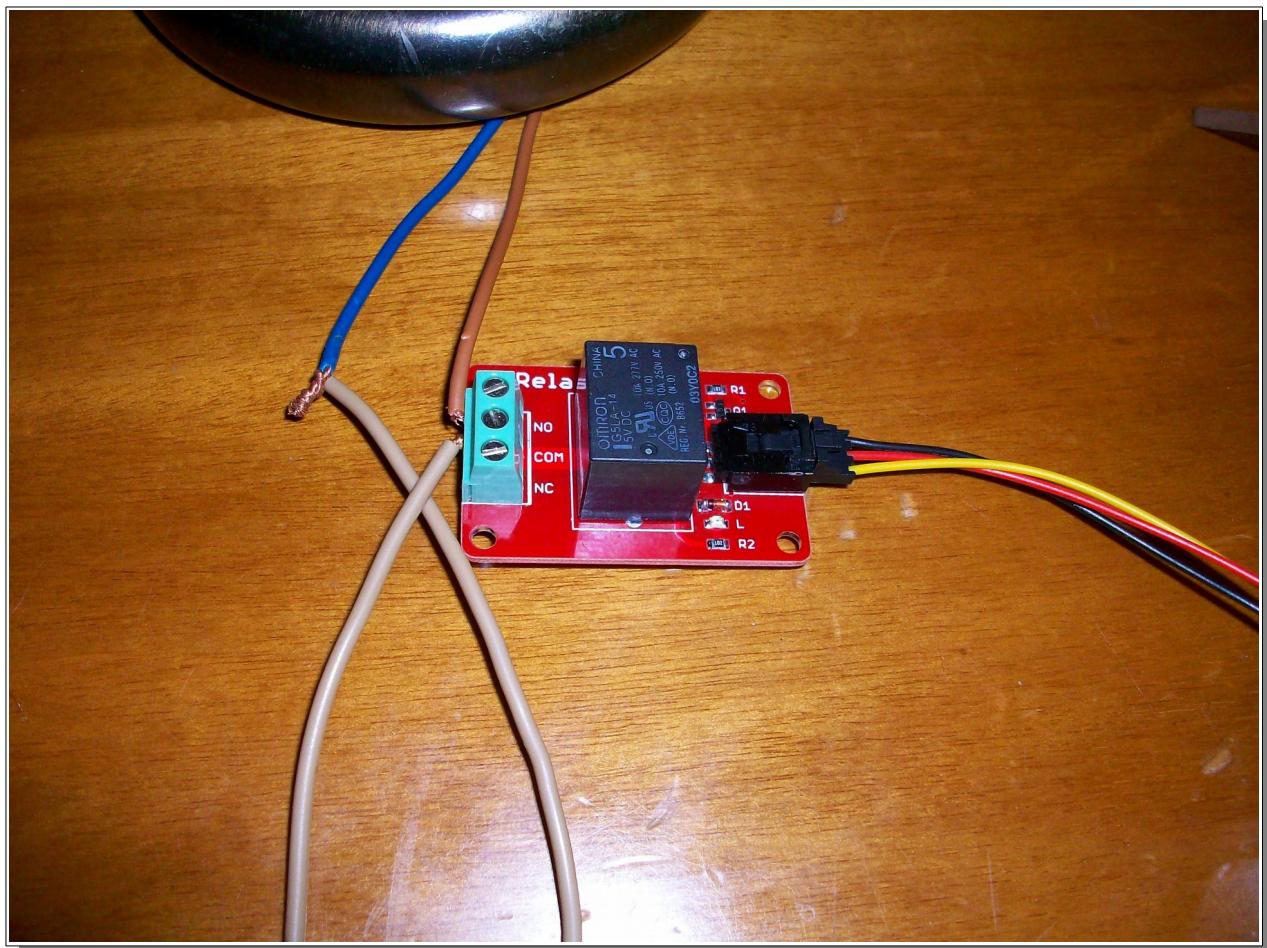
Φωτογραφία 3 : Η οθόνη υγρών κρυστάλλων του κόμβου ταυτοποίησης



Φωτογραφία 4 : Ψηφιακή ταυτοποίηση μιας παθητικής ετικέτας RFID



Φωτογραφία 5 : Ο κόμβος ελέγχου με τα λειτουργικά του μέρη



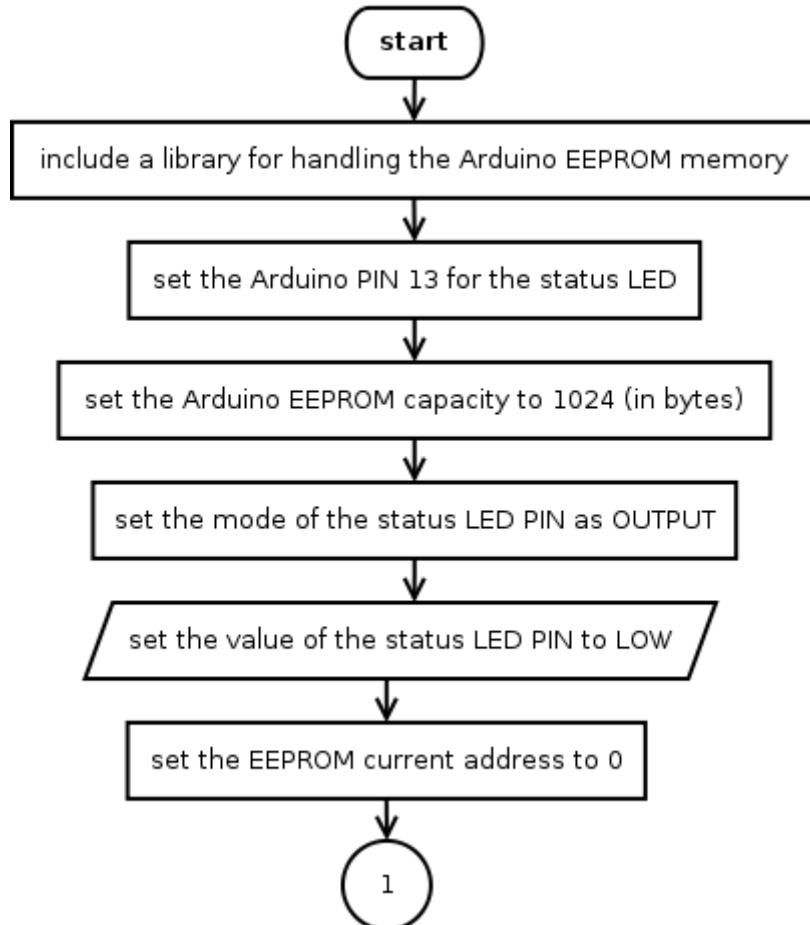
Φωτογραφία 6 : Συνδεσμολογία ηλεκτρομηχανικού ρελέ και φωτιστικού

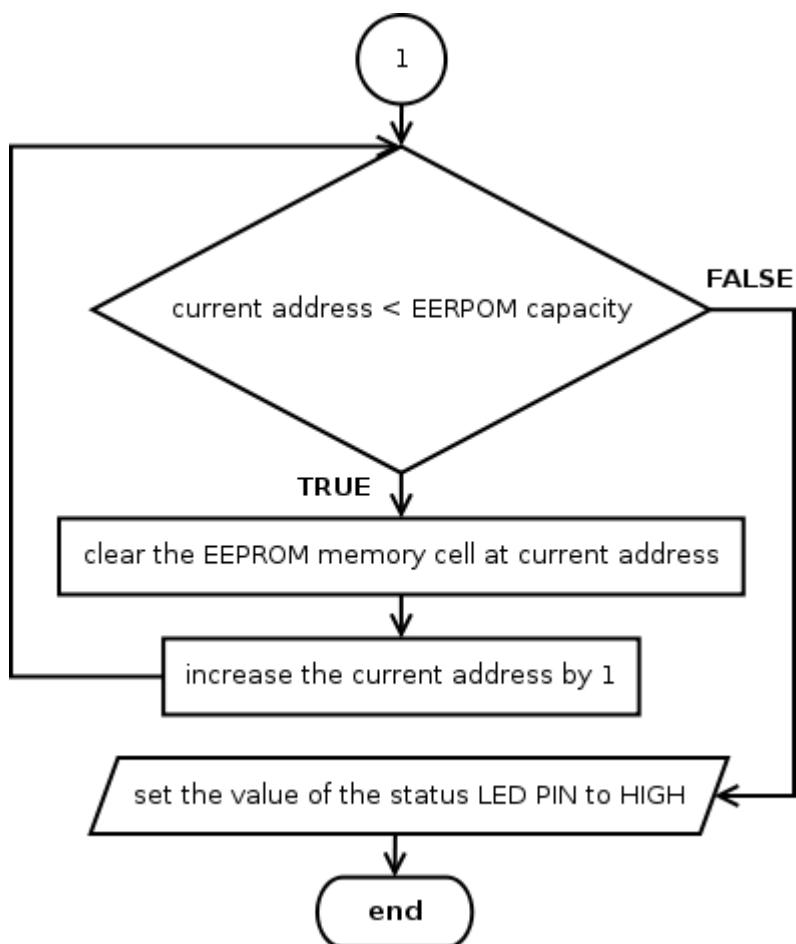


Φωτογραφία 7 : Οι διάφορες παθητικές ετικέτες *RFID* του συστήματος

Παράρτημα 3 - Διαγράμματα Ροής

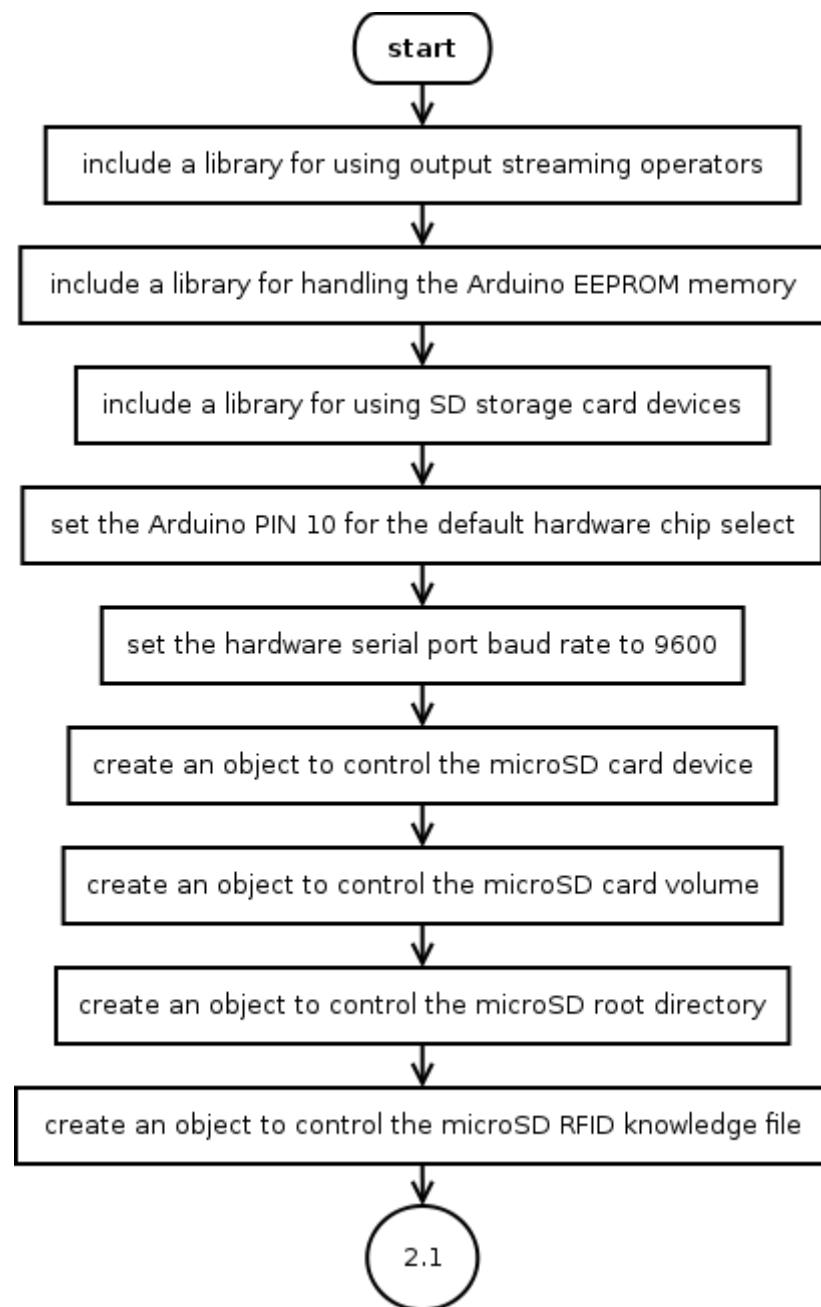
Παρακάτω, ακολουθεί το διάγραμμα ροής του υλικολογισμικού που χρησιμοποιείται για την λειτουργία εκκαθάρισης της μνήμης EEPROM του ενσωματωμένου κόμβου ταυτοποίησης :

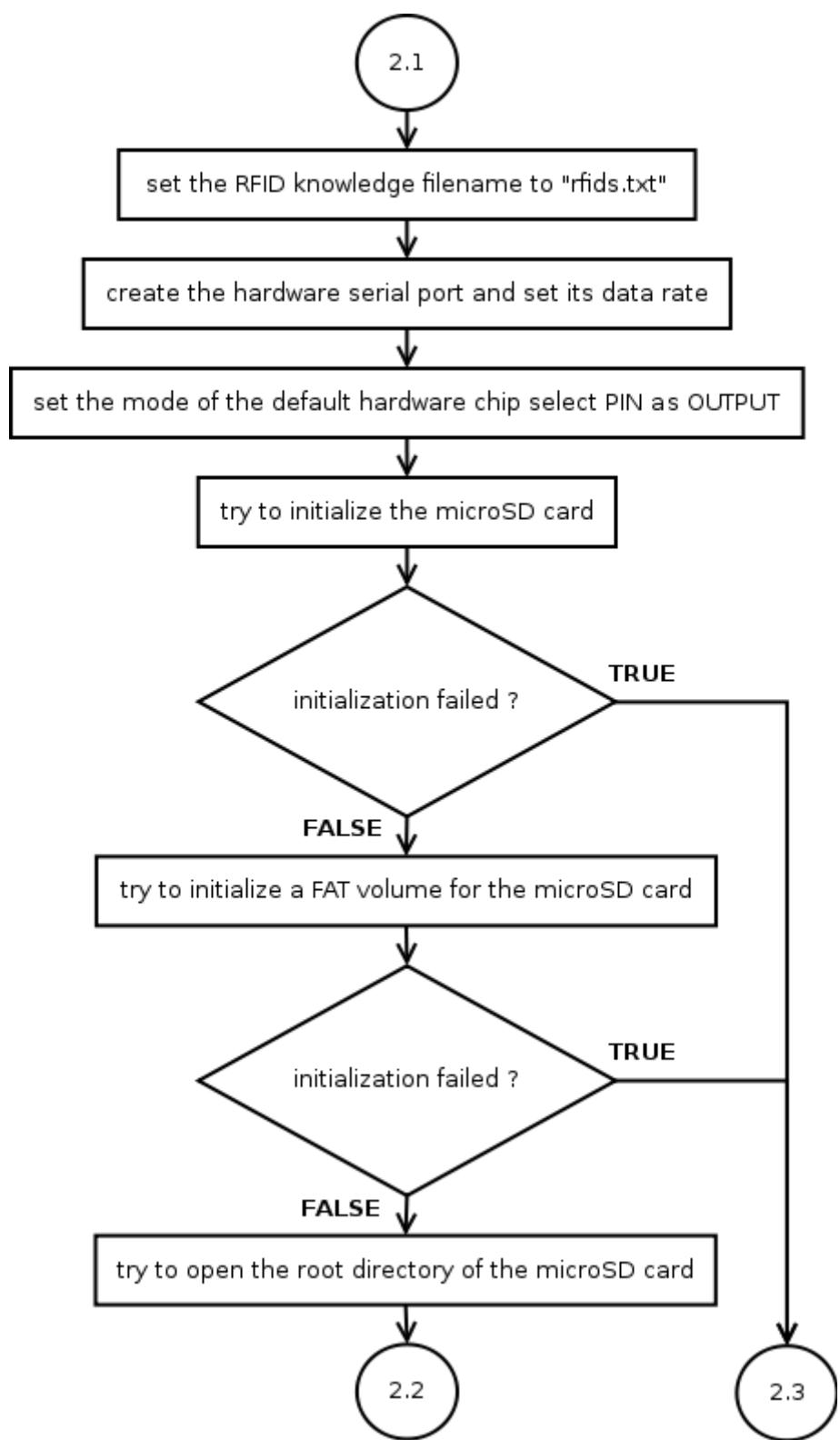


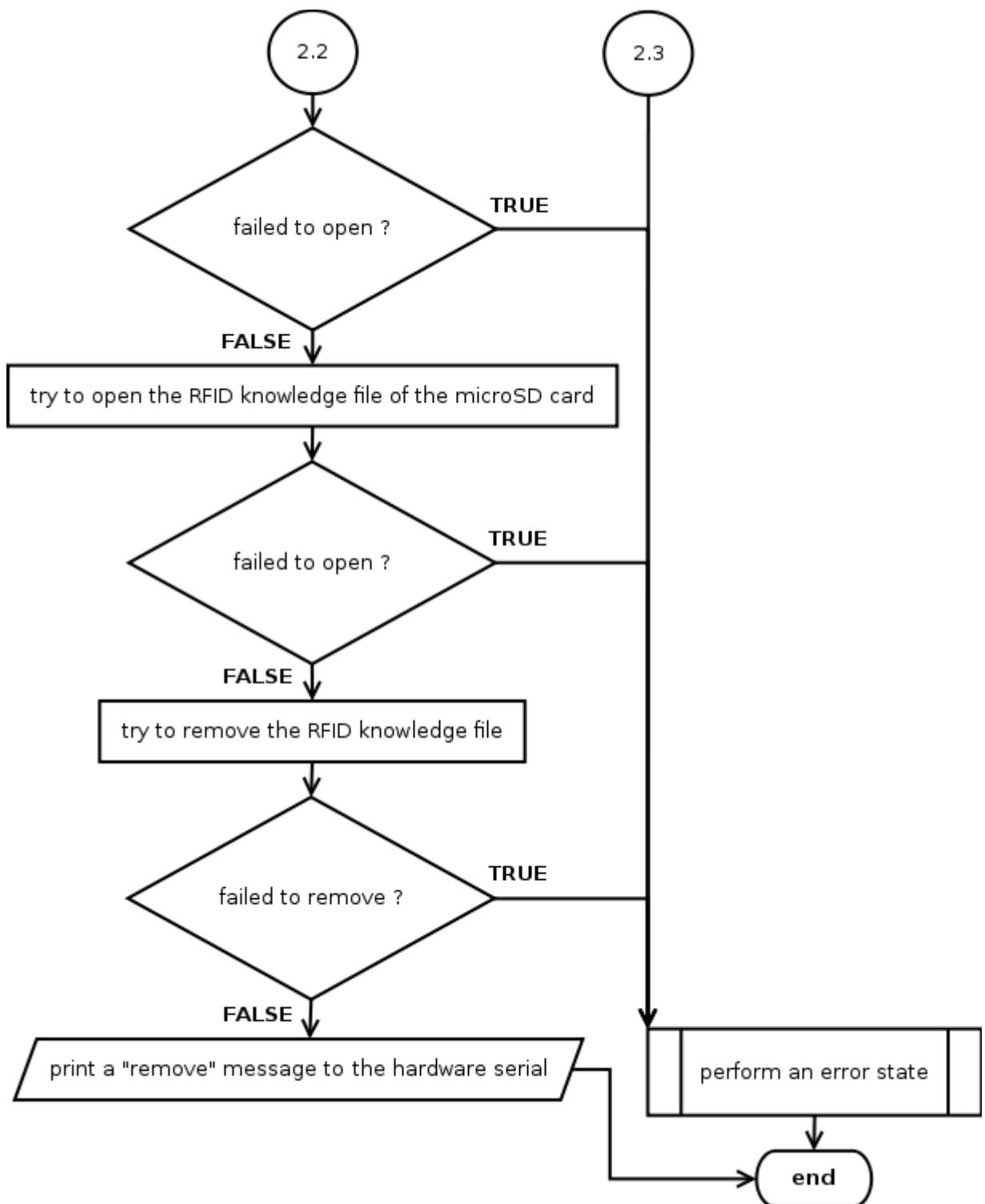


Διάγραμμα 01 : Διάγραμμα ροής υλικολογισμικού καθαρισμού μνήμης EEPROM

Παρακάτω, ακολουθεί το διάγραμμα ροής του υλικολογισμικού που χρησιμοποιείται για την λειτουργία εκκαθάρισης της μνήμης microSD του ενσωματωμένου κόμβου ταυτοποίησης :

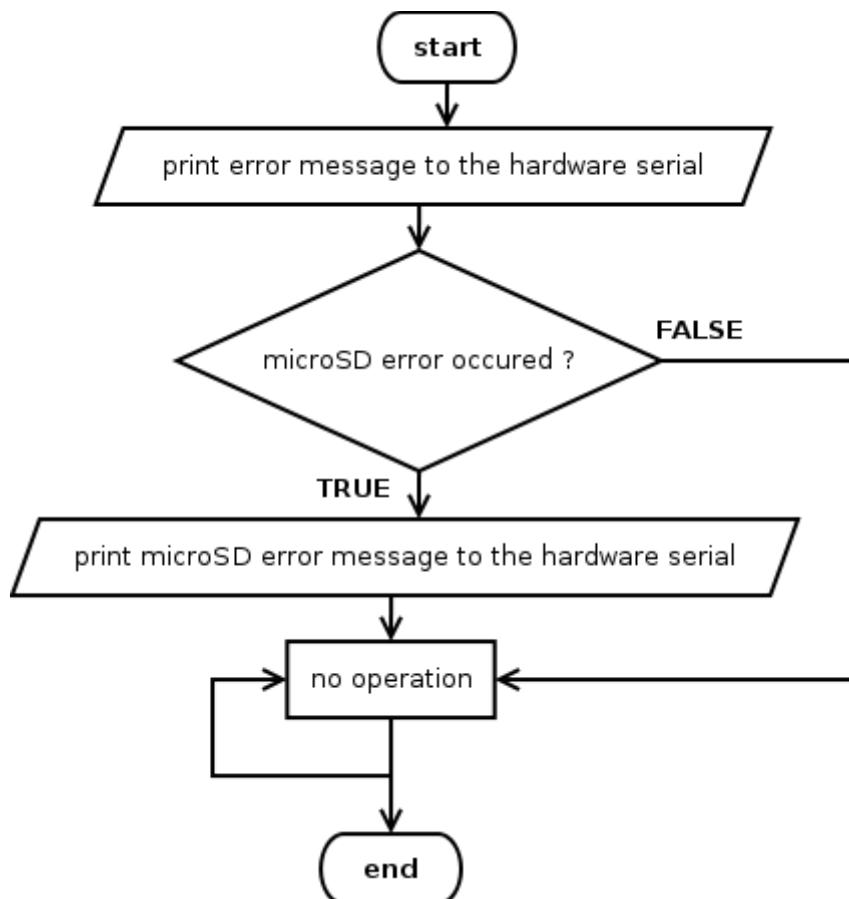






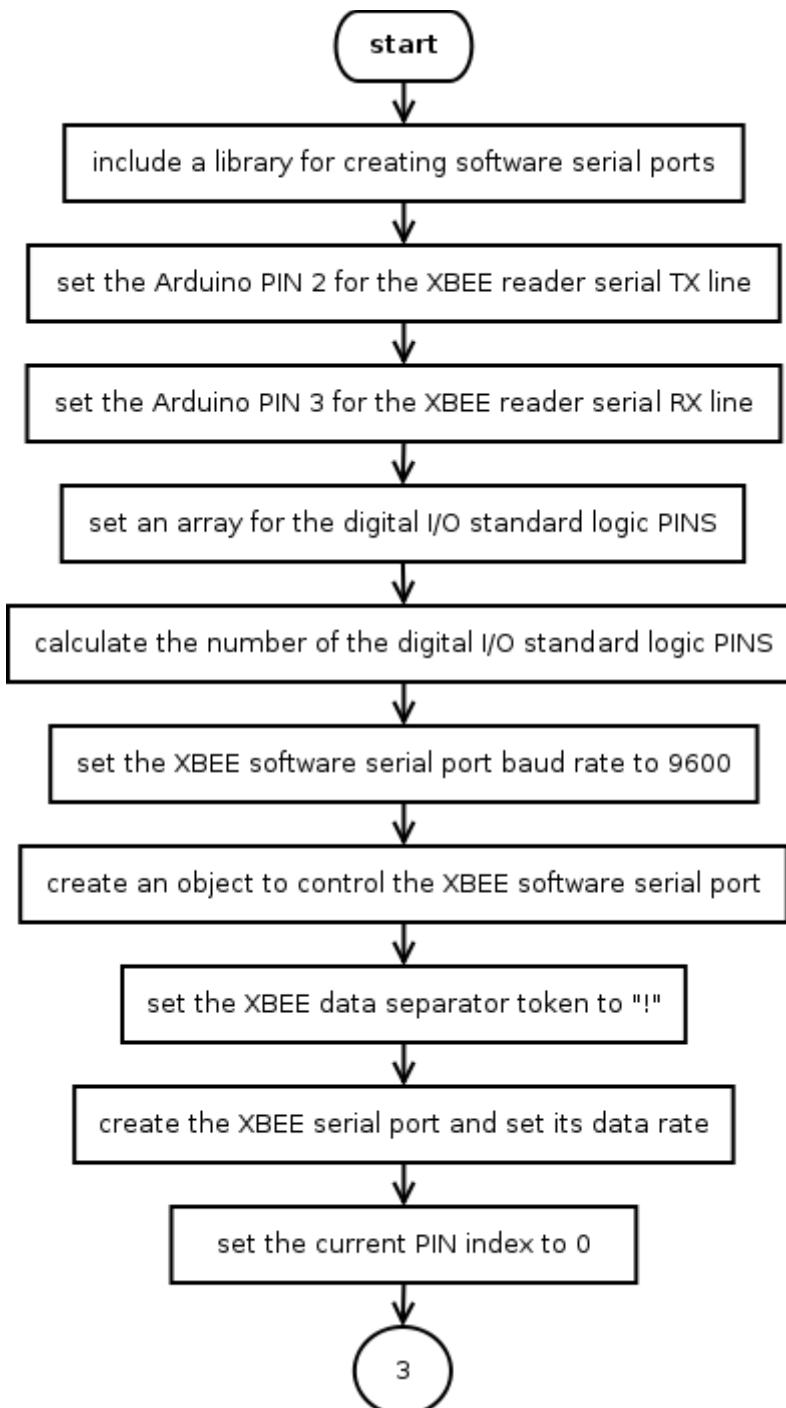
Διάγραμμα 02 : Διάγραμμα ροής υλικολογισμικού καθαρισμού μνήμης microSD

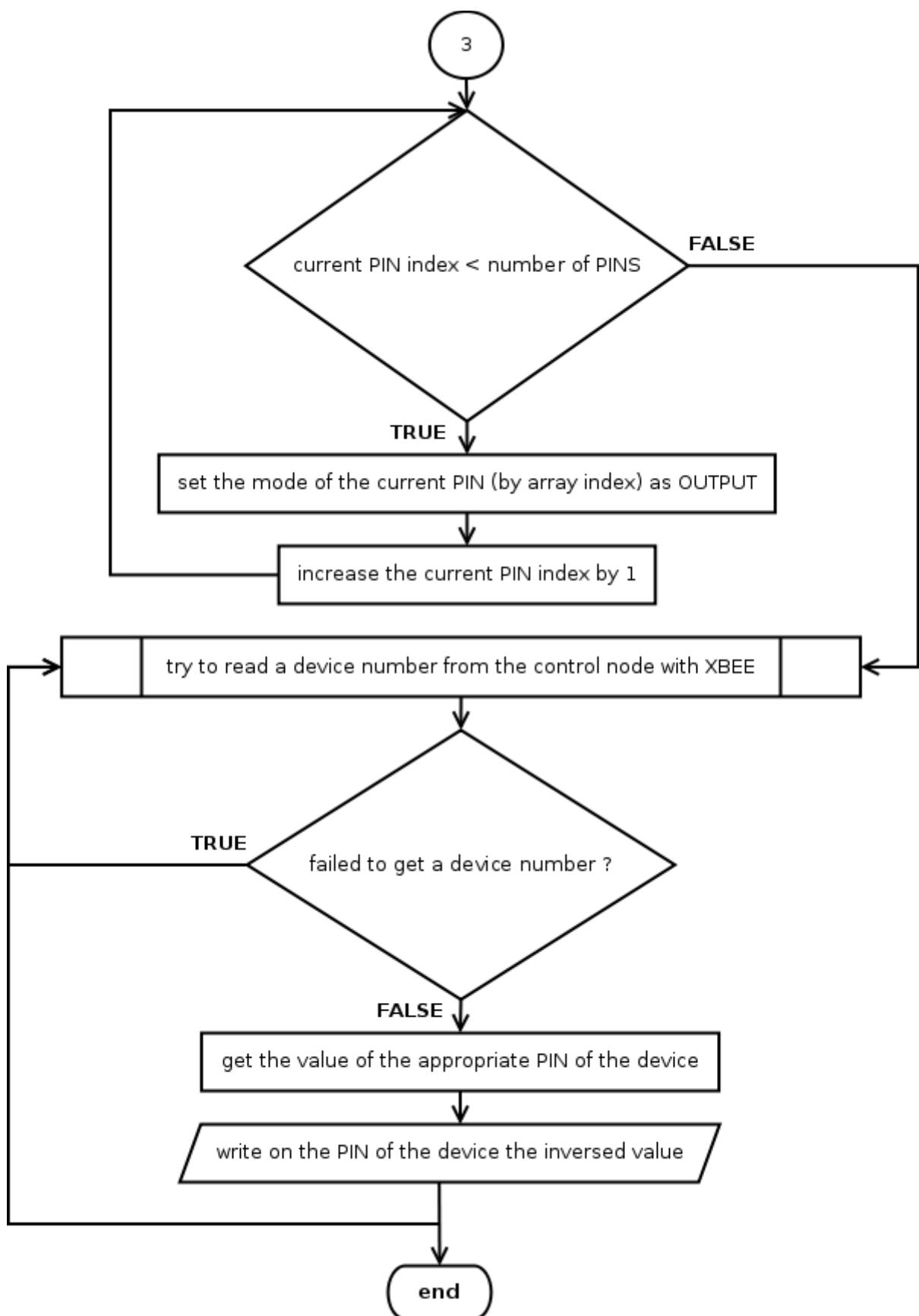
Παρακάτω, ακολουθεί το διάγραμμα ροής της υπορουτίνας “**perform an error state**” του υλικολογισμικού που χρησιμοποιείται για την λειτουργία εκκαθάρισης της μνήμης microSD του ενσωματωμένου κόμβου ταυτοποίησης :



Διάγραμμα 03 : Διάγραμμα ροής της υπορουτίνας “perform an error state”

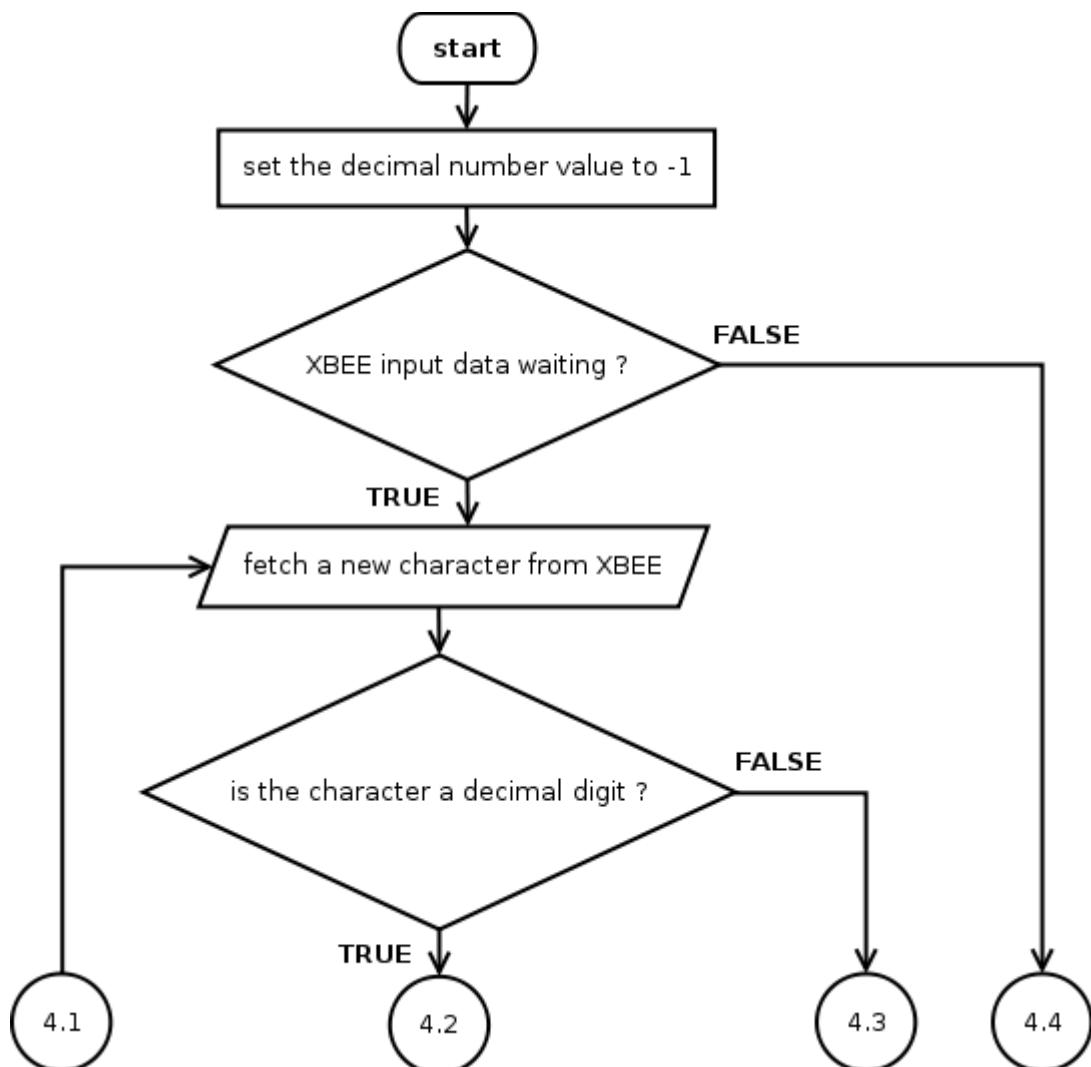
Παρακάτω, ακολουθεί το διάγραμμα ροής του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ελέγχου :

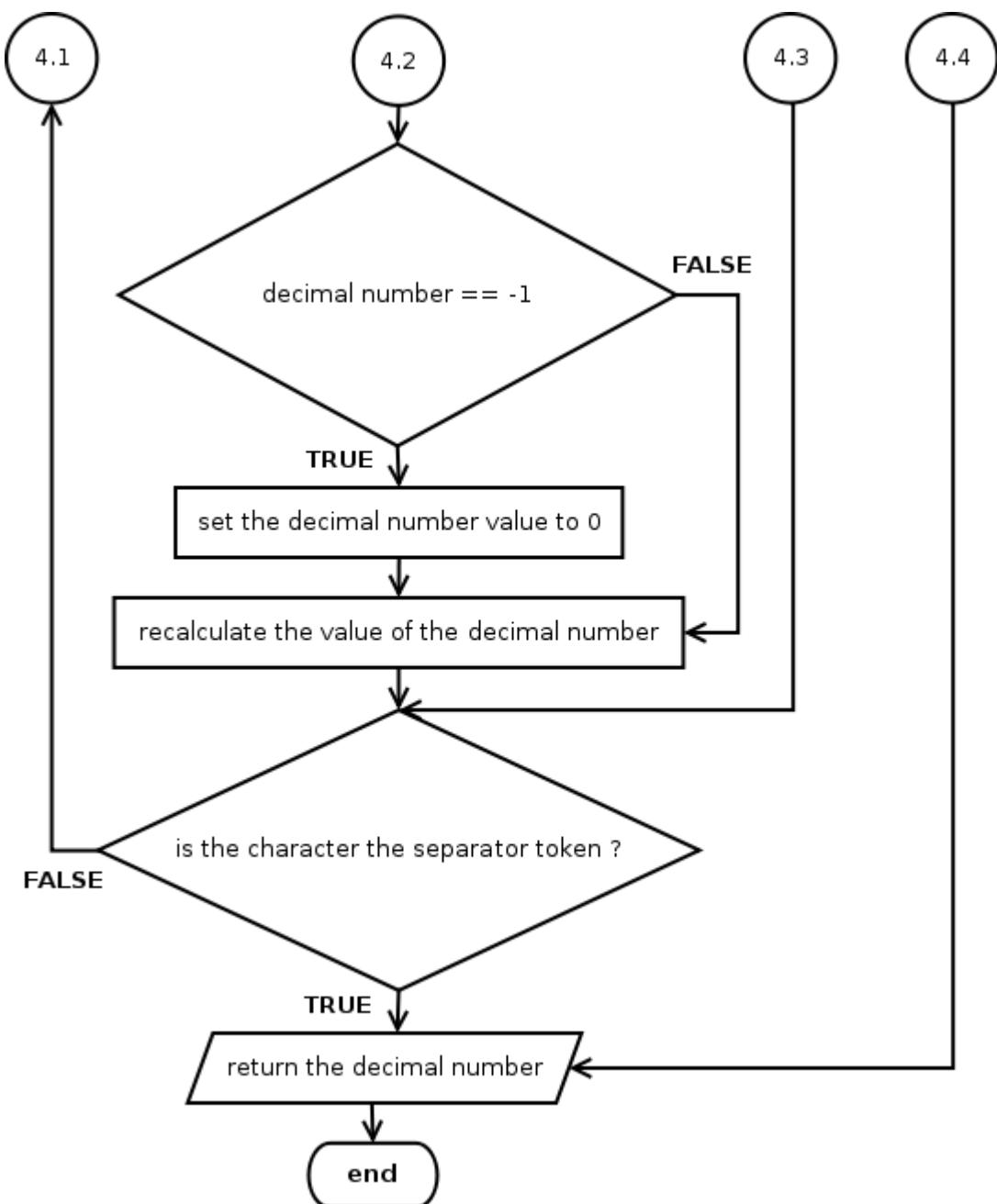




Διάγραμμα 04 : Διάγραμμα ροής υλικολογισμικού κόμβου ελέγχου

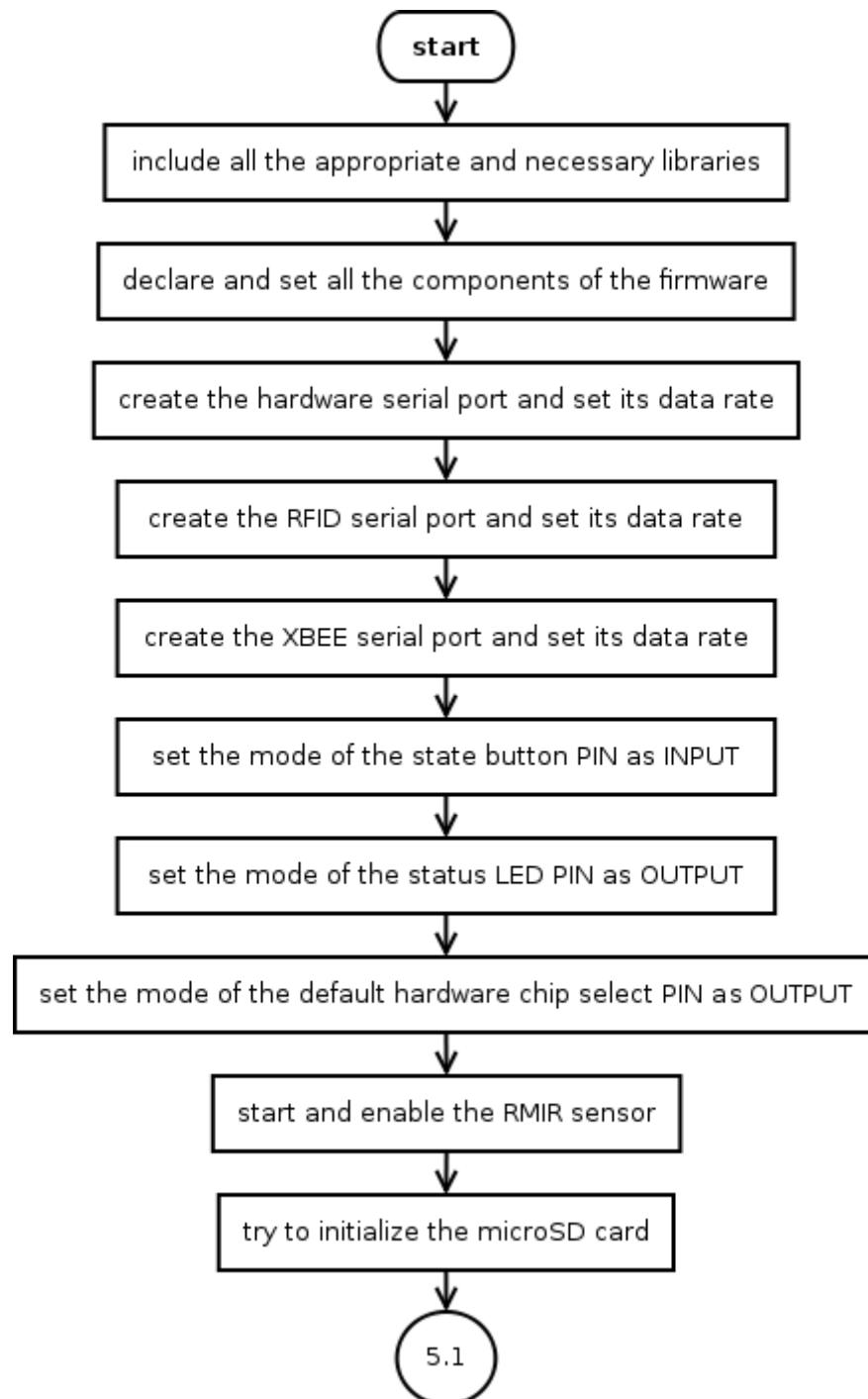
Παρακάτω, ακολουθεί το διάγραμμα ροής της υπορουτίνας “**read number from XBEE**” του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ελέγχου :

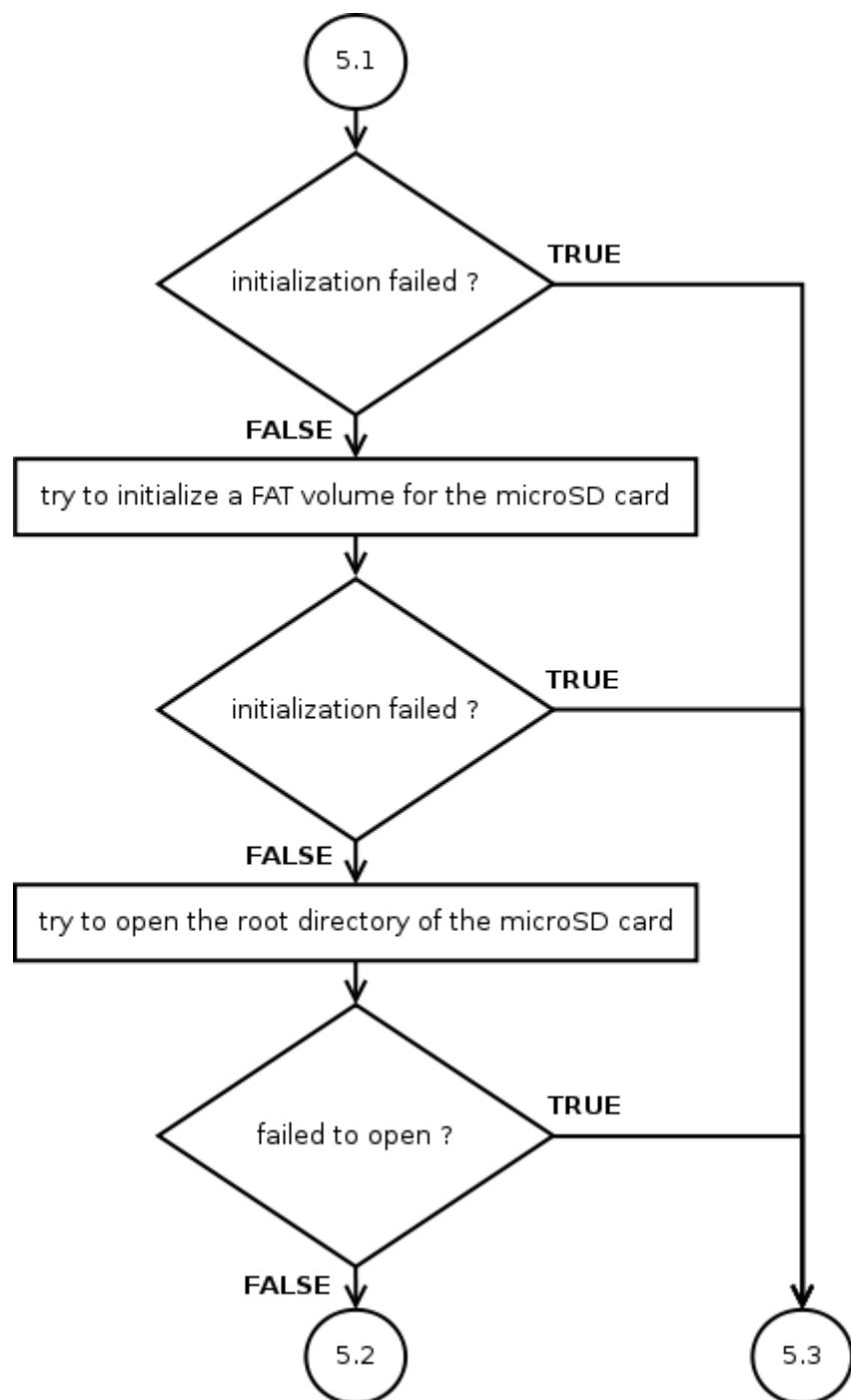


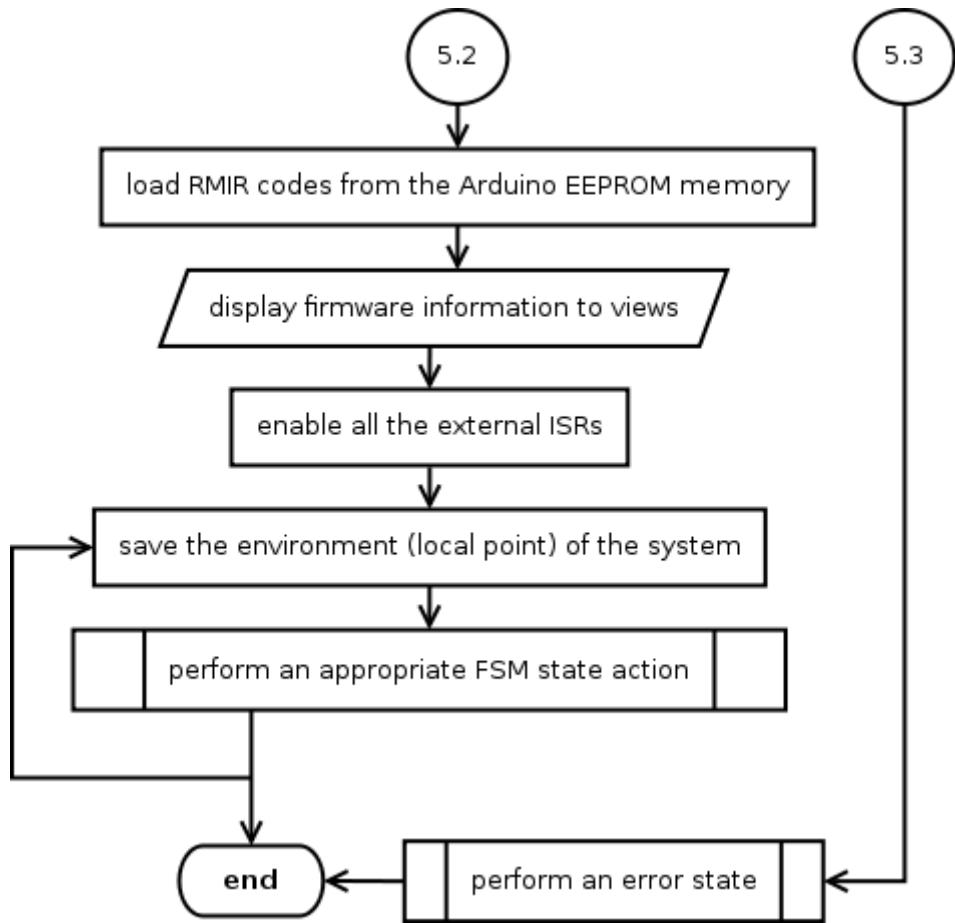


Διάγραμμα 05 : Διάγραμμα ροής της υπορουτίνας “read number from XBEE”

Παρακάτω, ακολουθεί το διάγραμμα ροής του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ταυτοποίησης :

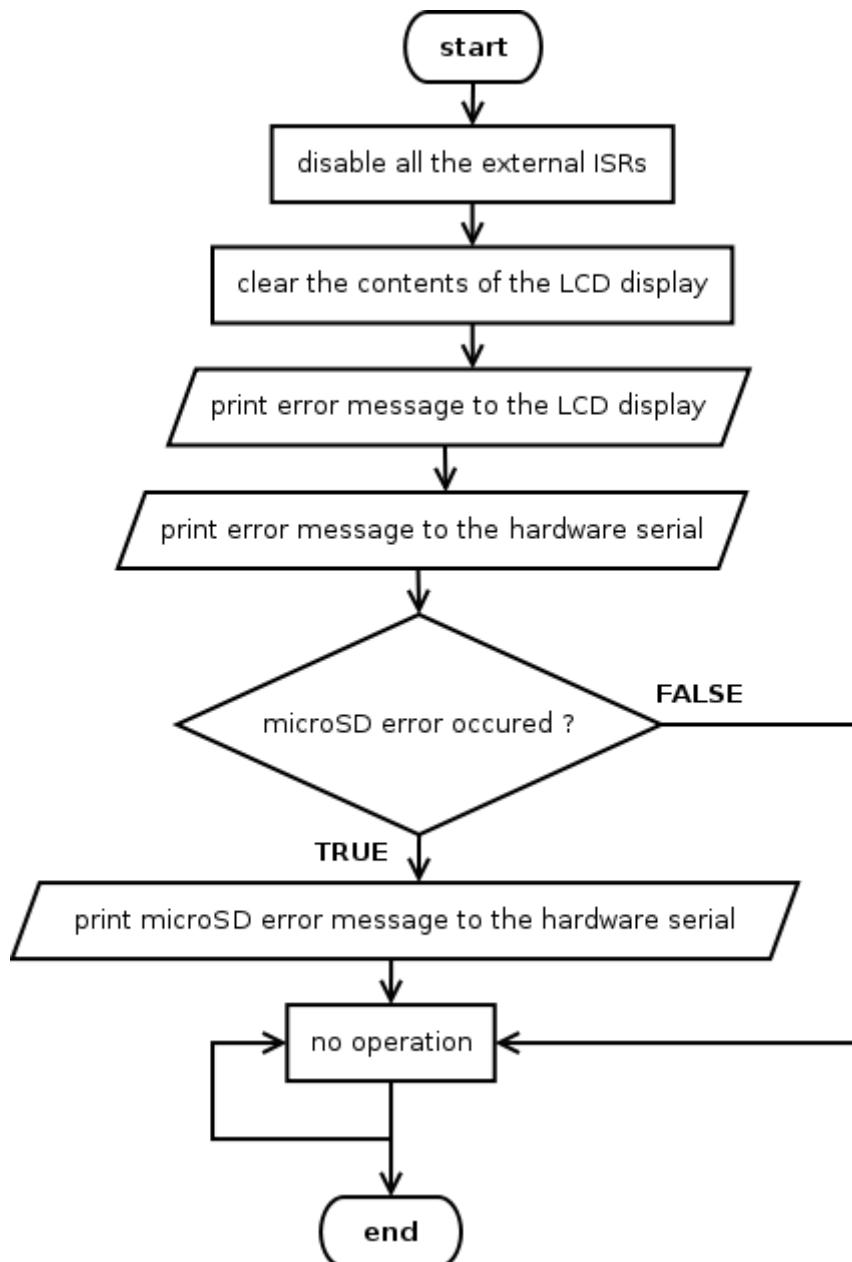






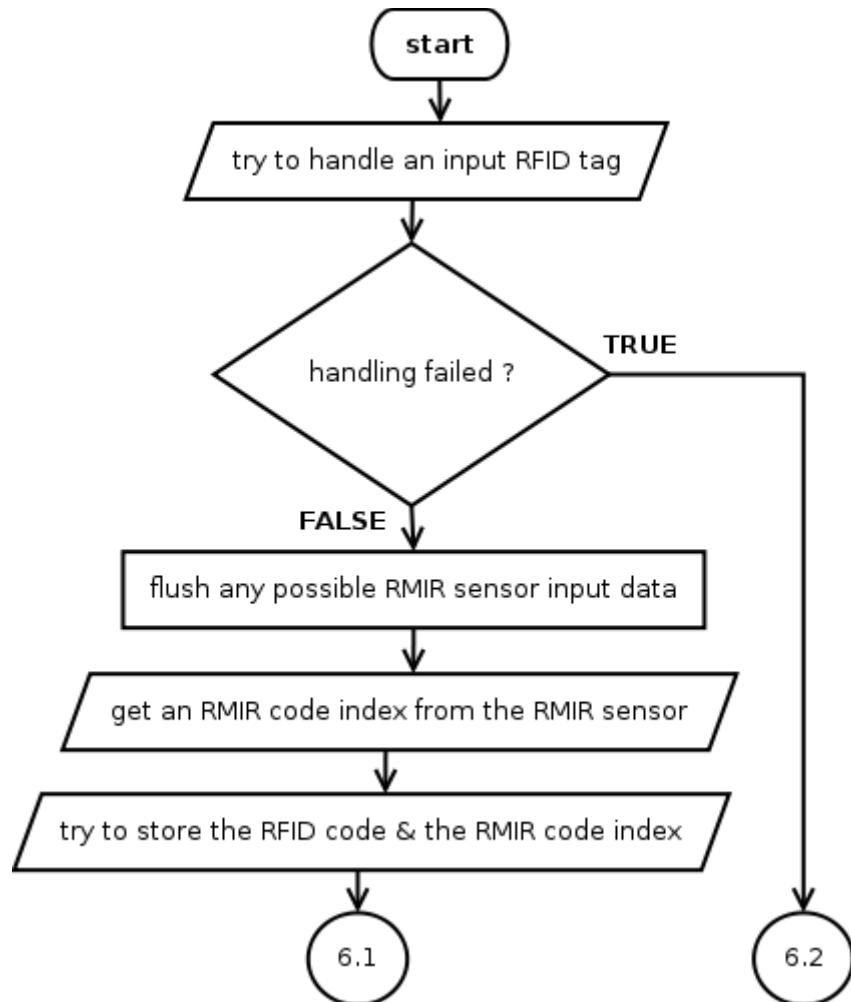
Διάγραμμα 06 : Διάγραμμα ροής υλικολογισμικού κόμβου ταυτοποίησης

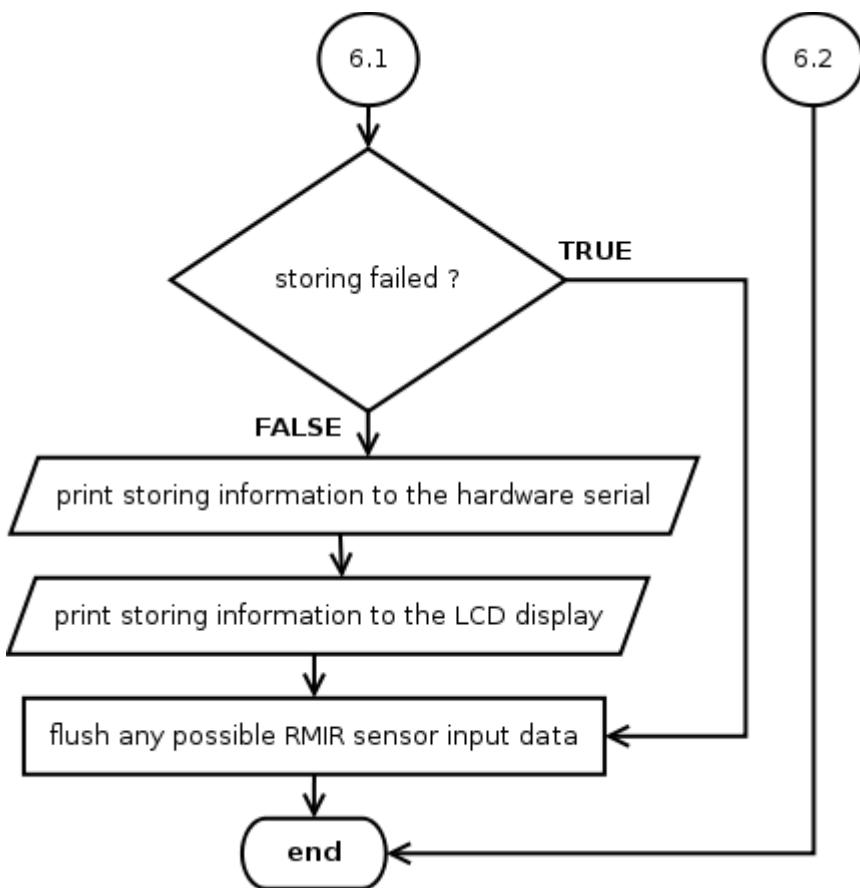
Παρακάτω, ακολουθεί το διάγραμμα ροής της υπορουτίνας “**perform an error state**” του υλικολογισμικού που χρησιμοποιείται για την βασική λειτουργία του ενσωματωμένου κόμβου ταυτοποίησης :



Διάγραμμα 07 : Διάγραμμα ροής της υπορουτίνας “*perform an error state*”

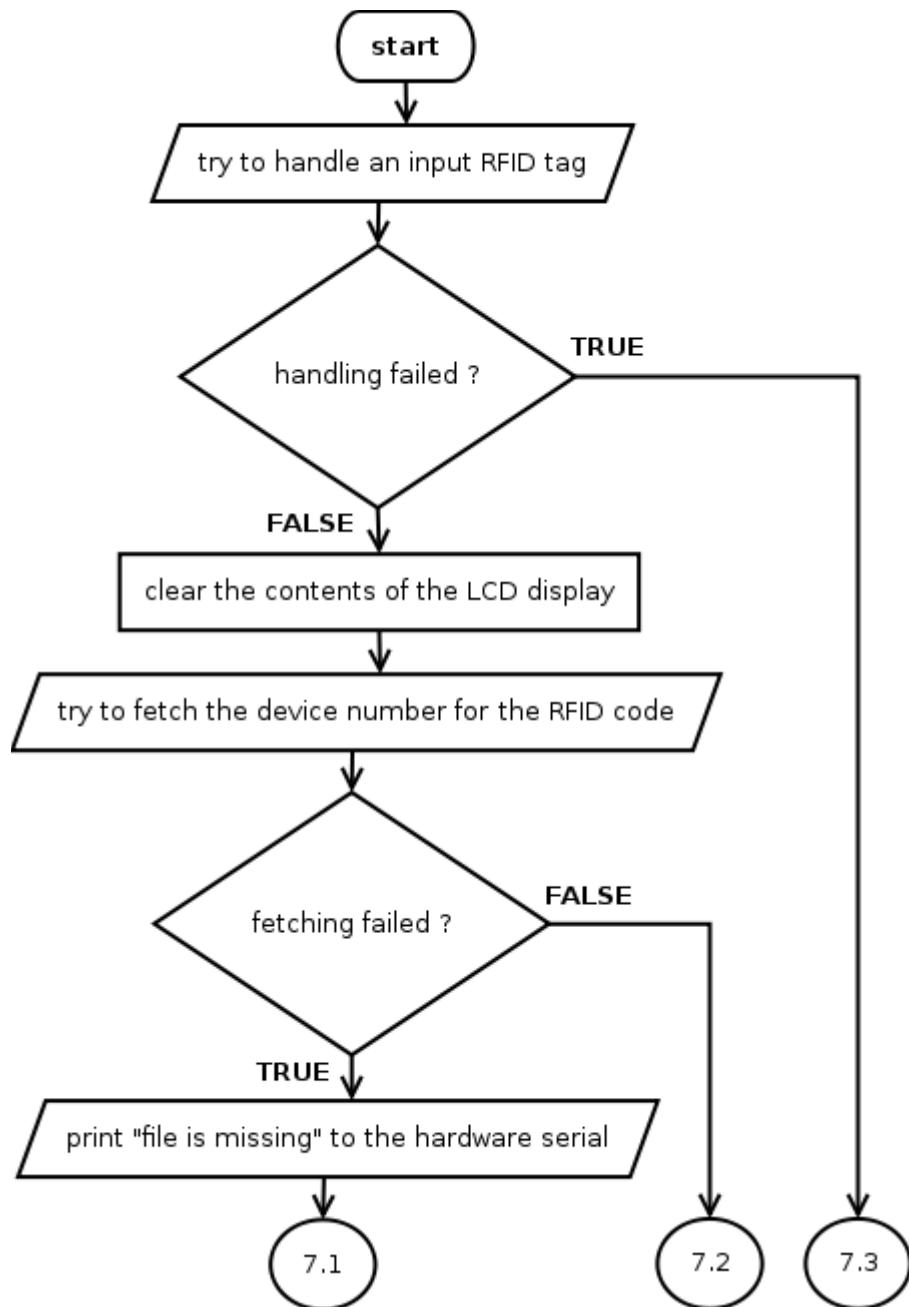
Παρακάτω, ακολουθεί το διάγραμμα ροής εκείνης της υπορουτίνας του ενσωματωμένου κόμβου ταυτοποίησης που χρησιμοποιείται για την εκμάθηση μιας παθητικής ετικέτας ραδιοσυχνικής ταυτοποίησης (RFID) :

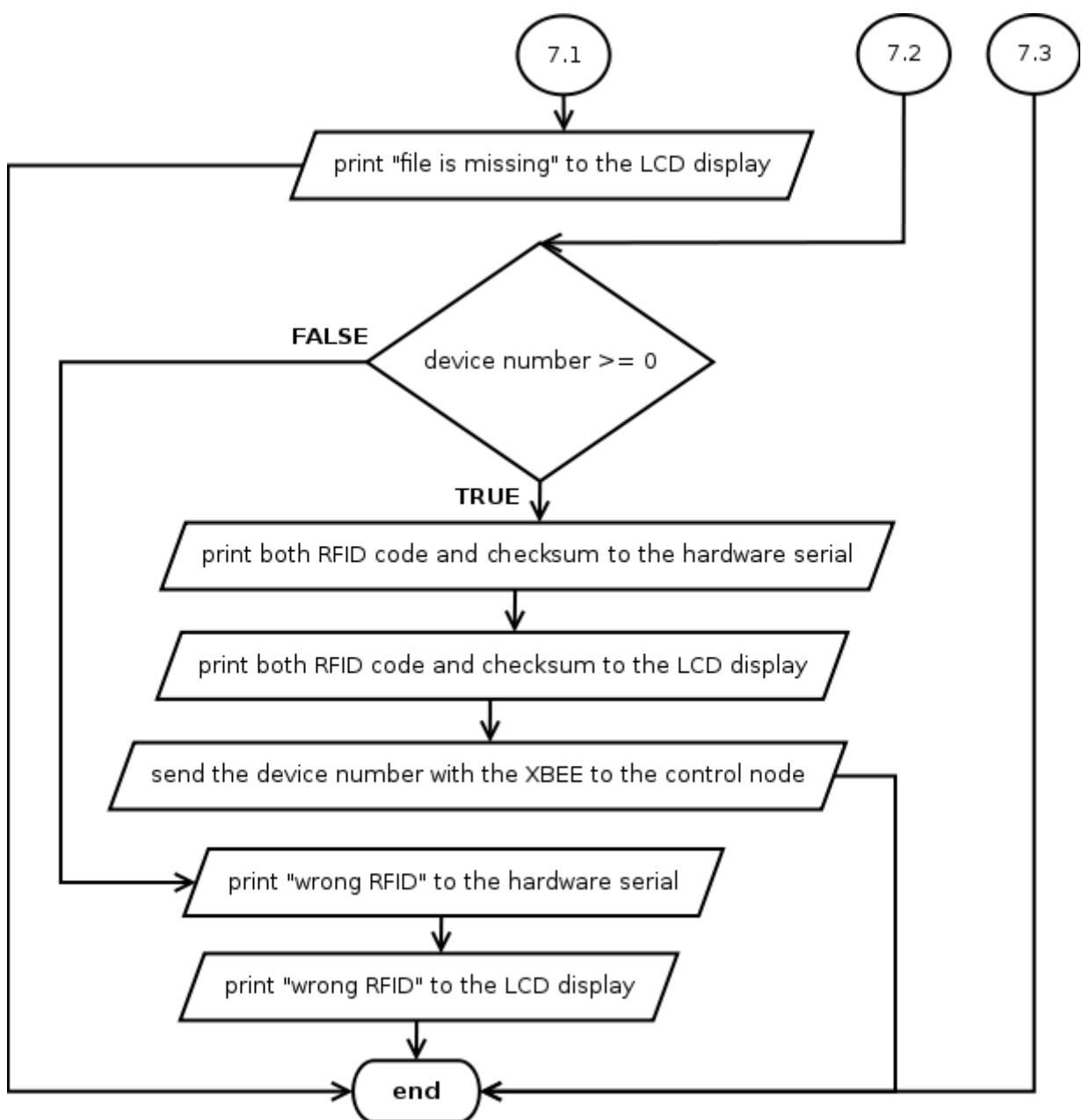




Διάγραμμα 08 : Διάγραμμα ροής υπορουτίνας εκμάθησης ετικέτας RFID

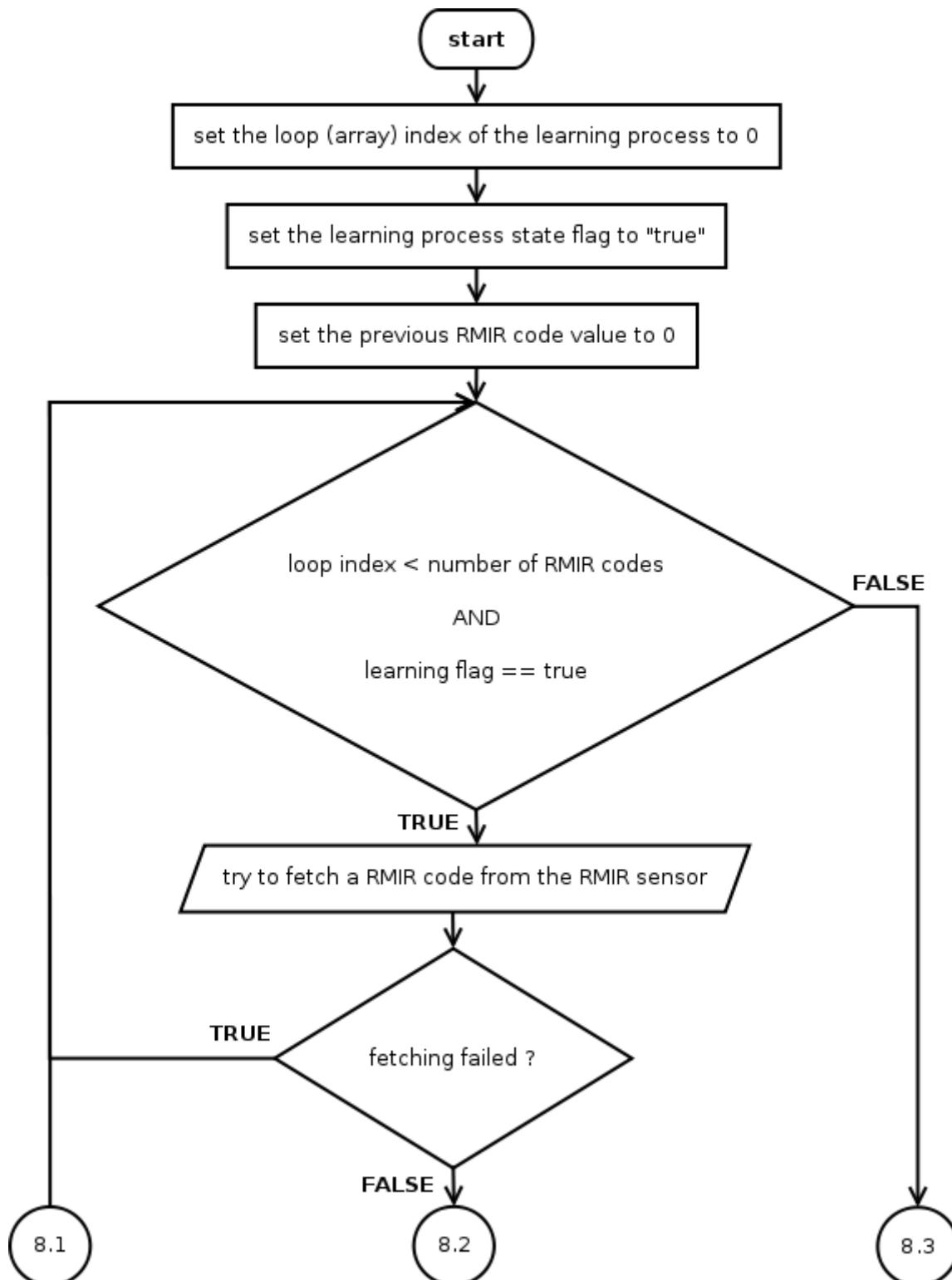
Παρακάτω, ακολουθεί το διάγραμμα ροής εκείνης της υπορουτίνας του ενσωματωμένου κόμβου ταυτοποίησης που χρησιμοποιείται για την άμεση εξυπηρέτηση μίας παθητικής ετικέτας ραδιοσυχνικής ταυτοποίησης (RFID) :

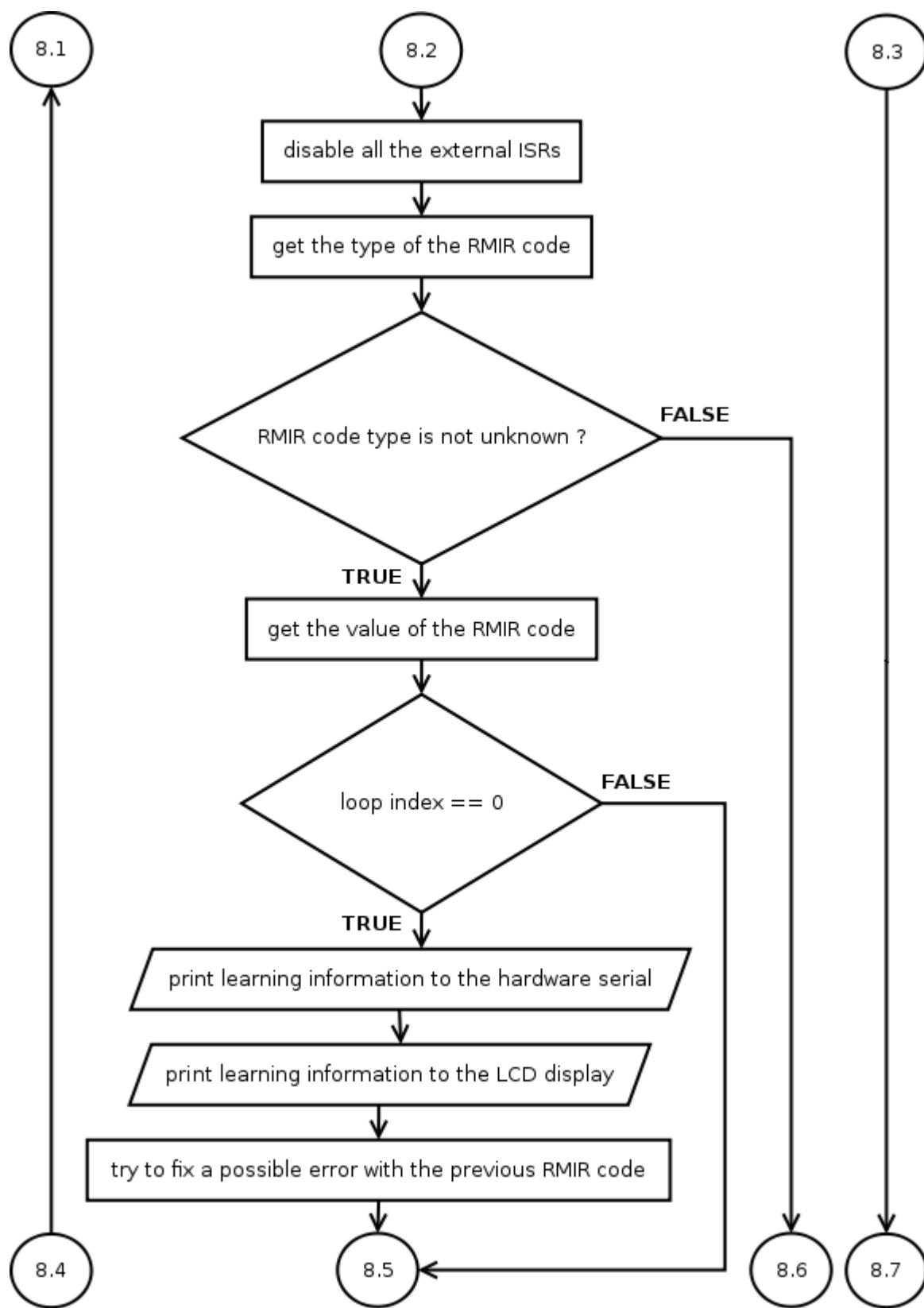


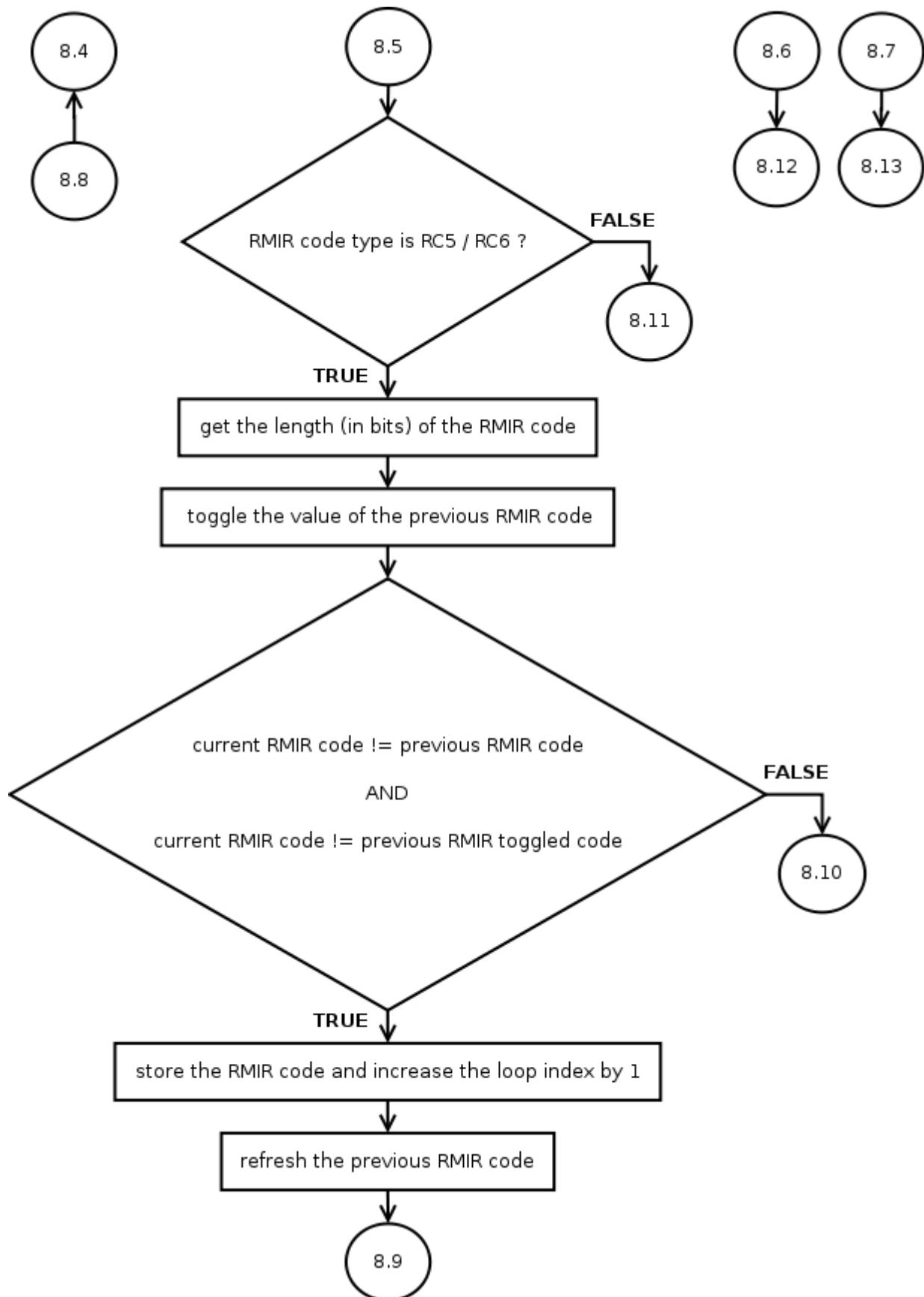


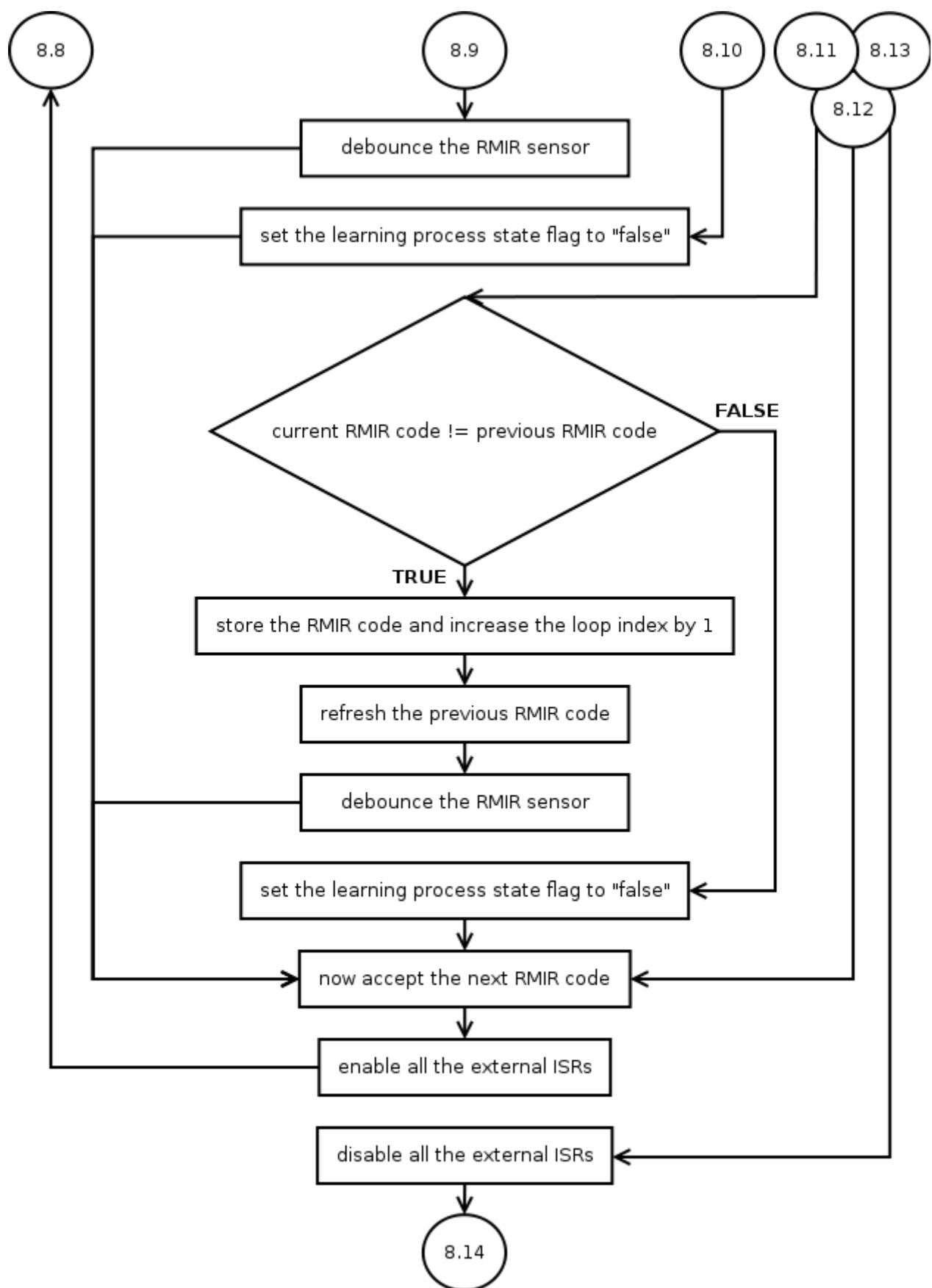
Διάγραμμα 09 : Διάγραμμα ροής υπορουτίνας εξυπηρέτησης ετικέτας RFID

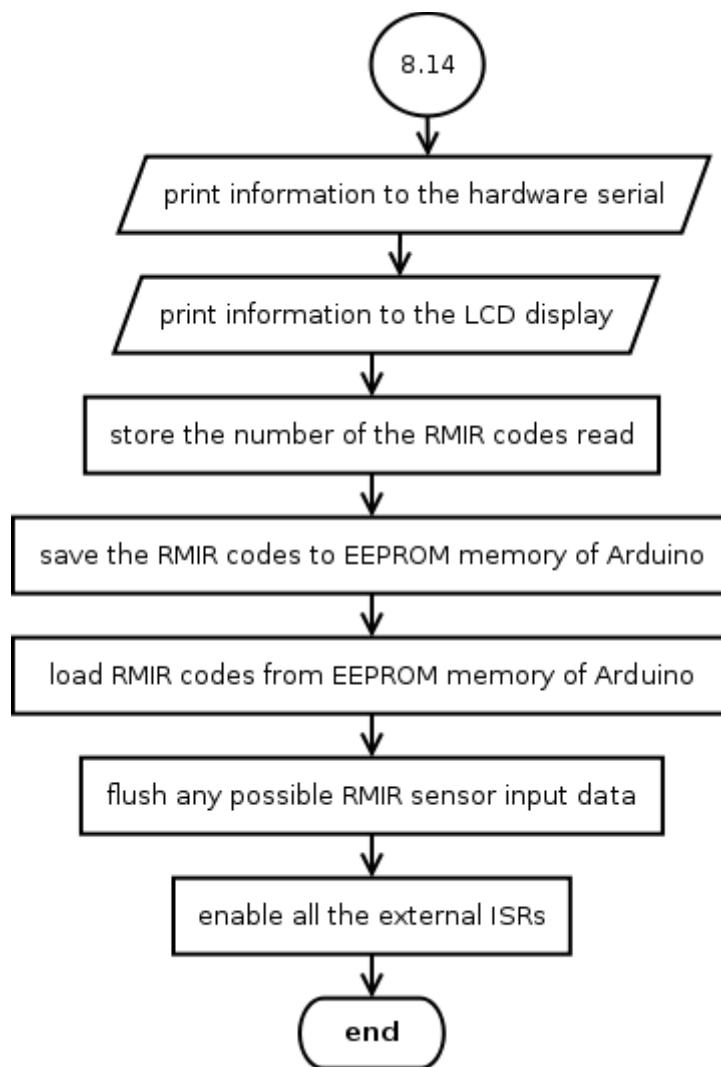
Παρακάτω, ακολουθεί το διάγραμμα ροής εκείνης της υπορουτίνας του ενσωματωμένου κόμβου ταυτοποίησης που χρησιμοποιείται για την άμεση εκμάθηση των πλήκτρων ενός ή περισσοτέρων τηλεχειριστηρίων :











Διάγραμμα 10 : Διάγραμμα ροής υπορουτίνας εκμάθησης τηλεχειριστηρίου

Παράρτημα 4 - Φύλλα Δεδομένων

DM74LS164

8-Bit Serial In/Parallel Out Shift Register

General Description

These 8-bit shift registers feature gated serial inputs and an asynchronous clear. A low logic level at either input inhibits entry of the new data, and resets the first flip-flop to the low level at the next clock pulse, thus providing complete control over incoming data. A high logic level on either input enables the other input, which will then determine the state of the first flip-flop. Data at the serial inputs may be changed while the clock is HIGH or LOW, but only information meeting the setup and hold time requirements will be entered. Clocking occurs on the LOW-to-HIGH level transition of the clock input. All inputs are diode-clamped to minimize transmission-line effects.

Features

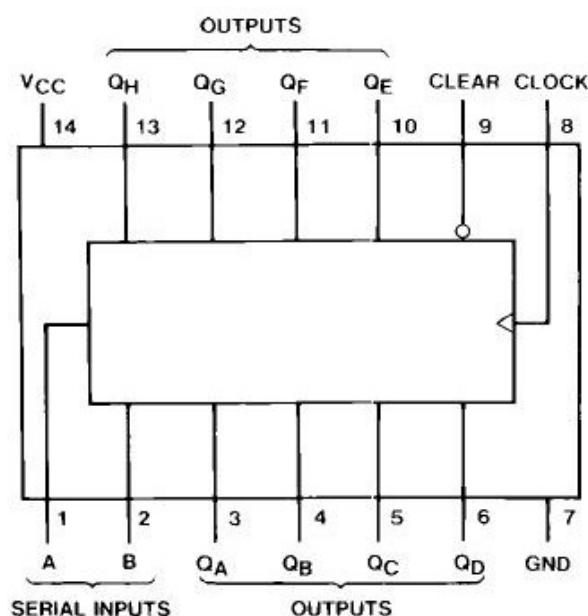
- Gated (enable/disable) serial inputs
- Fully buffered clock and serial inputs
- Asynchronous clear
- Typical clock frequency 36 MHz
- Typical power dissipation 80 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS164M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS164N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

Inputs			Outputs				
Clear	Clock	A B	QA	QB	...	QH	
L	X	X X	L	L	...	L	
H	L	X X	QA ₀	QB ₀	...	QH ₀	
H	↑	H H	H	QA _n	...	QG _n	
H	↑	L X	L	QA _n	...	QG _n	
H	↑	X L	L	QA _n	...	QG _n	

H = HIGH Level (steady state)

L = LOW Level (steady state)

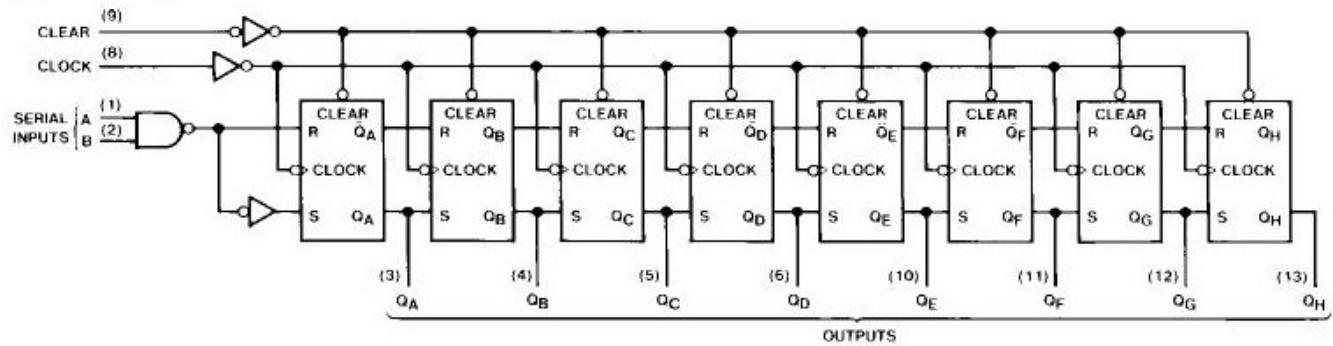
X = Don't Care (any input, including transitions)

↑ = Transition from LOW-to-HIGH level

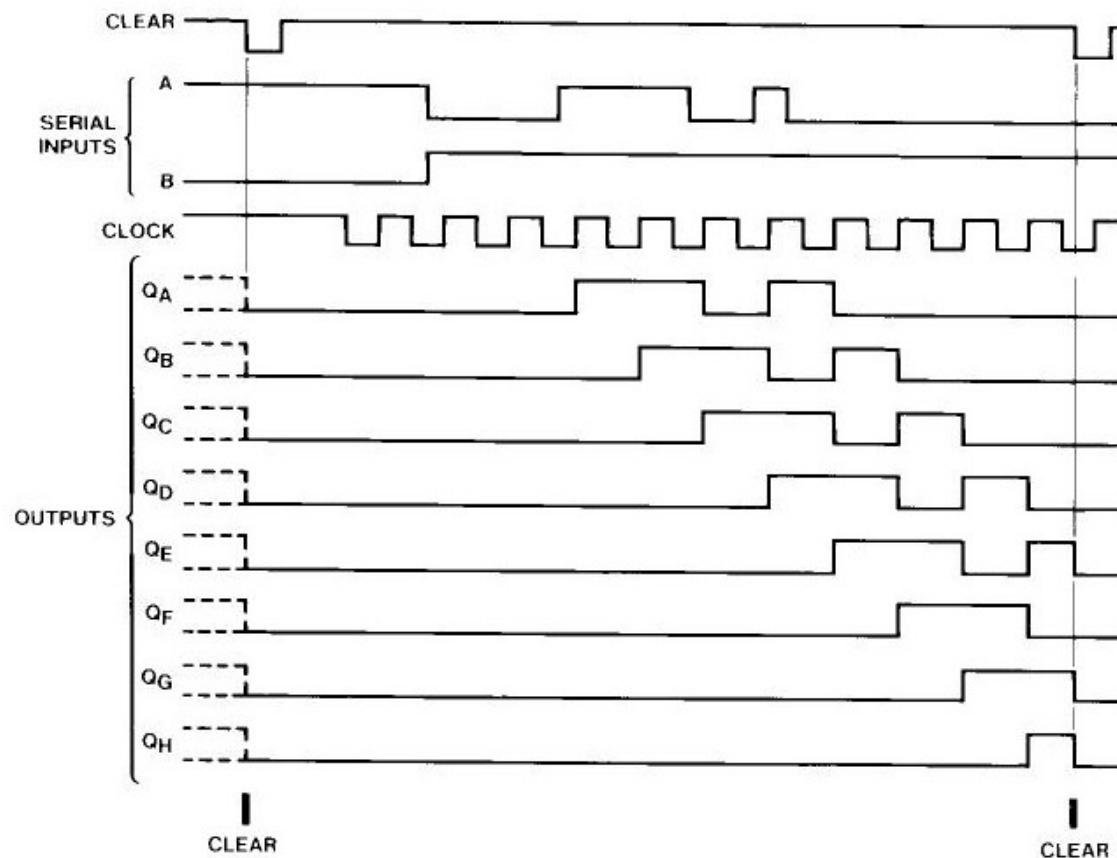
QA₀, QB₀, QH₀ = The level of QA, QB, or QH, respectively, before the indicated steady-state input conditions were established.

QA_n, QG_n = The level of QA or QG before the most recent ↑ transition of the clock; indicates a one-bit shift.

Logic Diagram



Timing Diagram



Absolute Maximum Ratings(Note 1)

Supply Voltage	7V
Input Voltage	7V
Operating Free Air Temperature Range	0°C to +70°C
Storage Temperature Range	-65°C to +150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" tables will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter	Min	Nom	Max	Units
V _{CC}	Supply Voltage	4.75	5	5.25	V
V _{IH}	HIGH Level Input Voltage	2			V
V _{IL}	LOW Level Input Voltage			0.8	V
I _{OH}	HIGH Level Output Current			-0.4	mA
I _{OL}	LOW Level Output Current			8	mA
f _{CLK}	Clock Frequency (Note 2)	0		25	MHz
t _W	Pulse Width (Note 2)	Clock	20		ns
		Clear	20		
t _{SU}	Data Setup Time (Note 2)	17			ns
t _H	Data Hold Time (Note 2)	5			ns
t _{REL}	Clear Release Time (Note 2)	30			ns
T _A	Free Air Operating Temperature	0		70	°C

Note 2: T_A = 25°C and V_{CC} = 5V.

Electrical Characteristics

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 3)	Max	Units
V _I	Input Clamp Voltage	V _{CC} = Min, I _I = -18 mA			-1.5	V
V _{OH}	HIGH Level Output Voltage	V _{CC} = Min, I _{OH} = Max V _{IL} = Max, V _{IH} = Min	2.7	3.4		V
V _{OL}	LOW Level Output Voltage	V _{CC} = Min, I _{OL} = Max V _{IL} = Max, V _{IH} = Min		0.35	0.5	V
		I _{OL} = 4 mA, V _{CC} = Min		0.25	0.4	
I _I	Input Current @ Max Input Voltage	V _{CC} = Max, V _I = 7V			0.1	mA
I _{IH}	HIGH Level Input Current	V _{CC} = Max, V _I = 2.7V			20	µA
I _{IL}	LOW Level Input Current	V _{CC} = Max, V _I = 0.4V			-0.4	mA
I _{OS}	Short Circuit Output Current	V _{CC} = Max (Note 4)	-20		-100	mA
I _{CC}	Supply Current	V _{CC} = Max (Note 5)		16	27	mA

Note 3: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 4: Not more than one output should be shorted at a time, and the duration should not exceed one second.

Note 5: I_{CC} is measured with all outputs OPEN, the SERIAL input grounded, the CLOCK input at 2.4V, and a momentary ground, then 4.5V, applied to the CLEAR input.

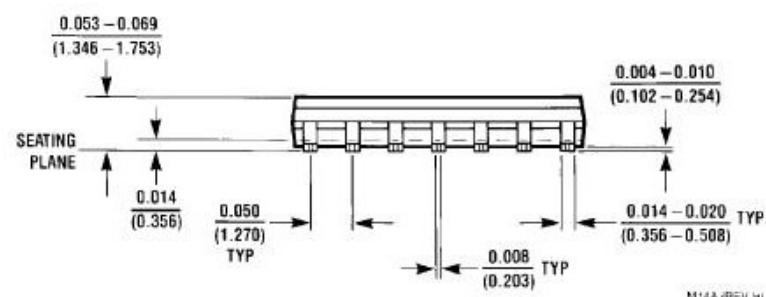
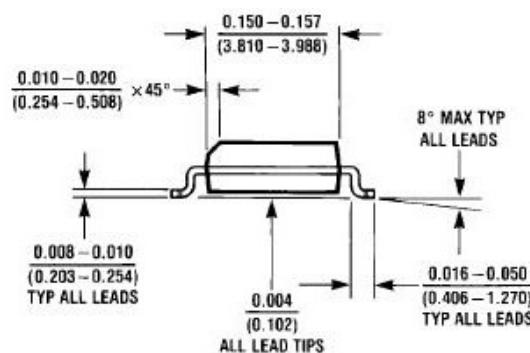
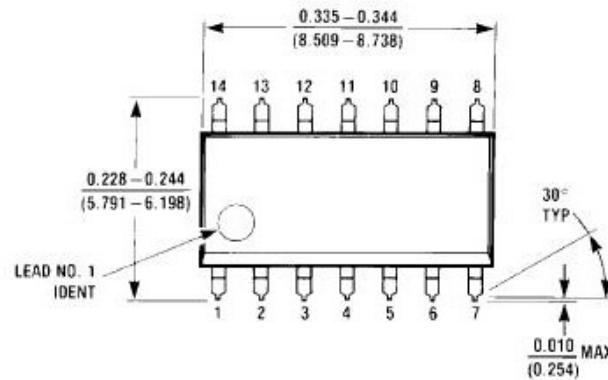
Switching Characteristics

at V_{CC} = 5V and T_A = 25°C

Symbol	Parameter	From (Input) To (Output)	R _L = 2 kΩ				Units	
			C _L = 15 pF		C _L = 50 pF			
			Min	Max	Min	Max		
f _{MAX}	Maximum Clock Frequency		25				MHz	
t _{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Clock to Output		27		30	ns	
t _{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clock to Output		32		40	ns	
t _{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clear to Output		36		45	ns	

Physical Dimensions

inches (millimeters) unless otherwise noted

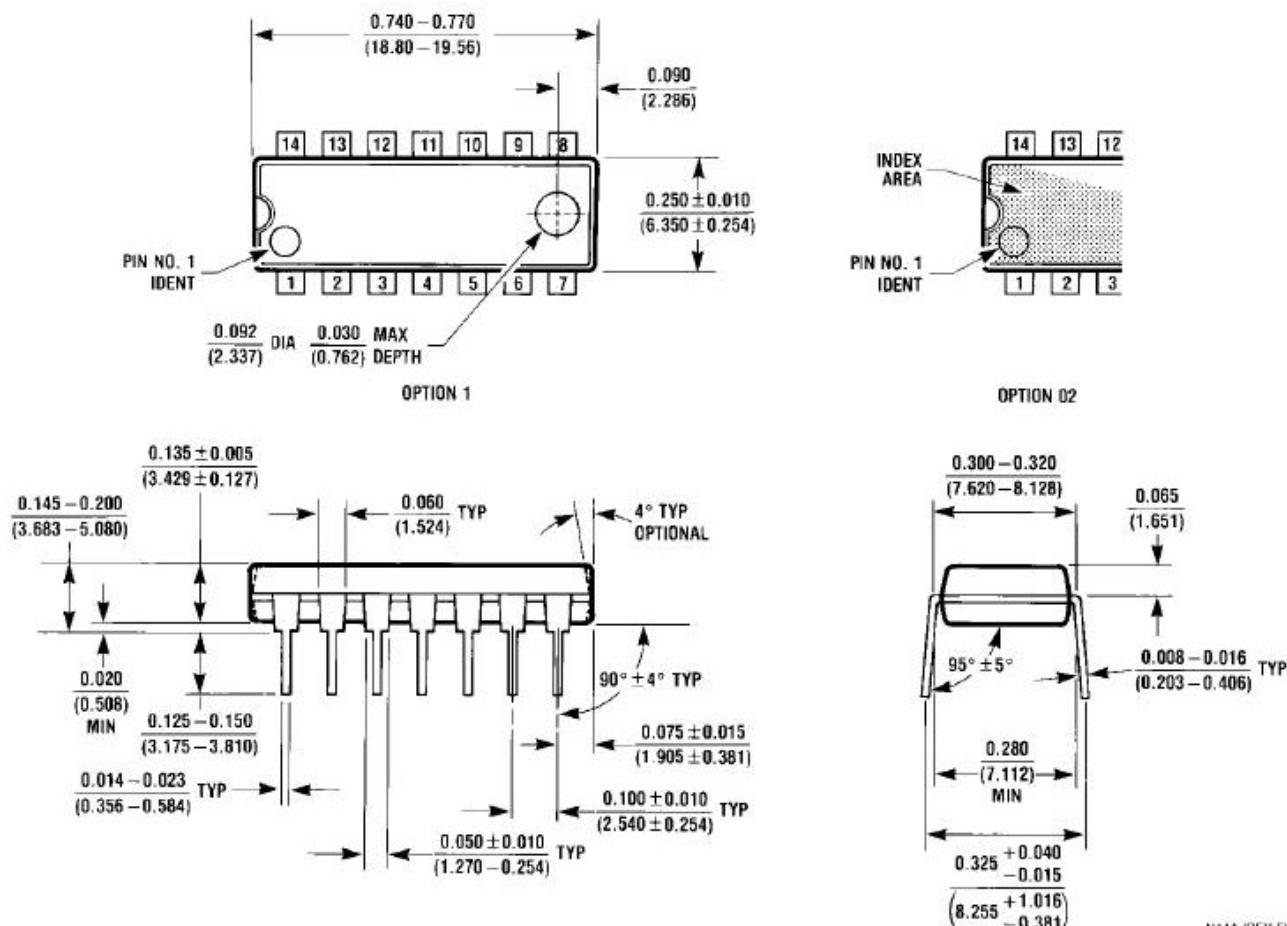


M14A (REV H)

**14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
Package Number M14A**

Physical Dimensions

inches (millimeters) unless otherwise noted (Continued)



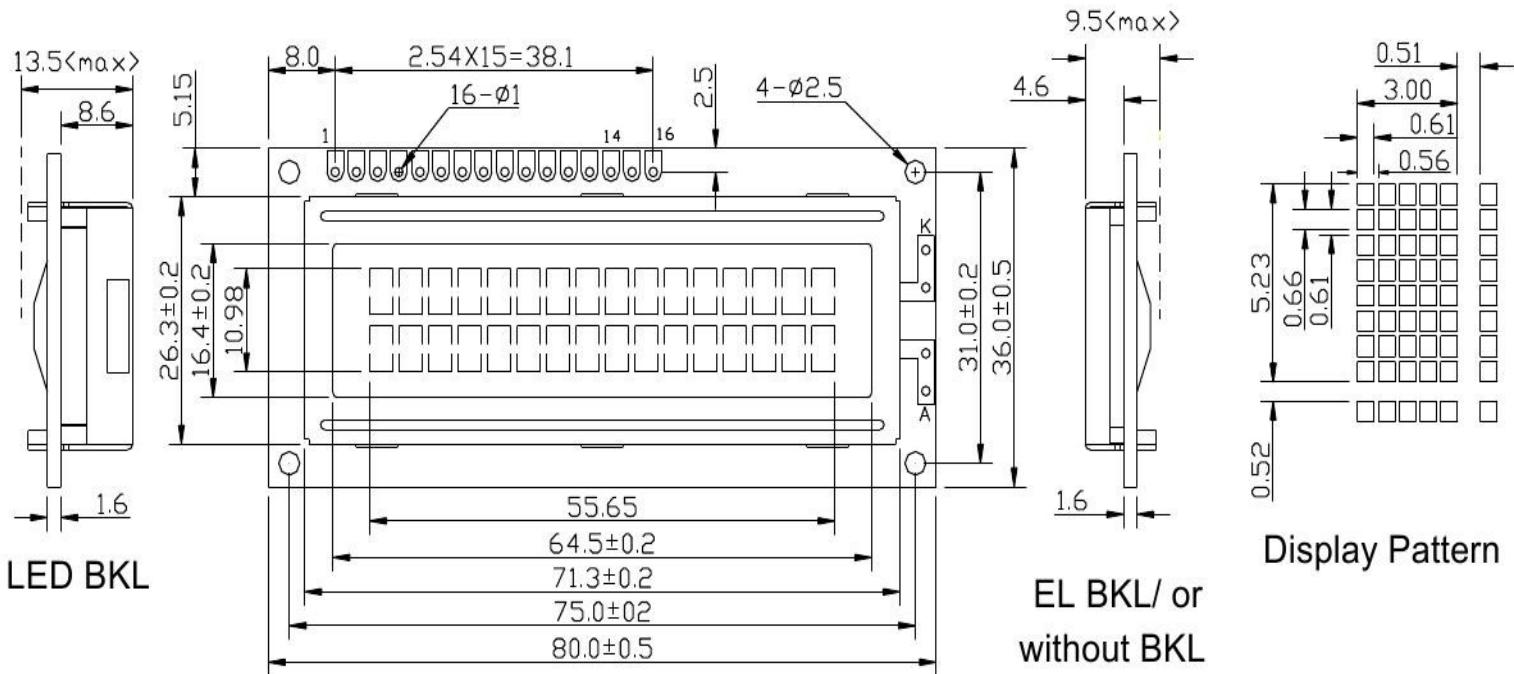
14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide
Package Number N14A

Fairchild does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and Fairchild reserves the right at any time without notice to change said circuitry and specifications.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.



Feature

1. 5X8 dots with cursor
 2. Built-in controller (KS0066U or Equivalent)
 3. +5V power supply(Also available for +3.0V)
 4. 1/16 duty
 5. BKL to be driven by pin1, pin2, or pin15, pin16, or A, K
 6. N.V.optional

PIN NO	Symbol	Function
1	VSS	GND
2	VDD	+5V
3	V0	Contrast adjustment
4	RS	H/L Register select signal
5	R/W	H/L Read/Write signal
6	E	H/L Enable signal
7	DB0	H/L Data bus line
8	DB1	H/L Data bus line
9	DB2	H/L Data bus line
10	DB3	H/L Data bus line
11	DB4	H/L Data bus line
12	DB5	H/L Data bus line
13	DB6	H/L Data bus line
14	DB7	H/L Data bus line
15	A	+4.2V for LED
16	K	Power supply for BKL(0V)

Mechanical Data

Item	Standard	Unit
Module dimension	80.0x36.0	mm
Viewing area	64.5x16.4	mm
Dot size	0.56x0.61	mm
Character size	3.00x5.23	mm

Absolute Maximum Rating

Item	Symbol	Standard			Unit
		Min	Typ	Max	
Power supply	VDD-VSS	-0.3	-----	5.5	V
Input voltage	VI	-0.3	-----	VDD	

Electronical characteristics

Item	Symbol	Condition	Standard			Unit
			Min	Typ	Max	
<i>Input voltage</i>	VDD	+5V	4.7	5.0	5.5	V
		+3.3V	2.7	3.0	5.3	V
<i>Supply current</i>	I _{DD}	VDD=5V	-----	1.5	4	mA
<i>Recommended LCD riling voltage for normal temp version module</i>	VDD-V0	-20°C	-----	-----	-----	V
		0 °C	4.7	5.0	5.5	
		25°C	4.3	4.5	4.7	
		50°C	4.1	4.3	4.5	
		70°C	-----	-----	-----	
<i>LED forward voltage</i>	V _F	25°C	-----	4.2	4.6	V
<i>LED forward current</i>	I _F	25°C	-----	120	160	mA
<i>EL power supply current</i>	I _{EL}	V _{EL} =110V AC 400Hz	-----	-----	-----	mA

Display character address code:

Display position

PNA4601M Series (PNA4601M/4602M/4608M/4610M)

Bipolar Integrated Circuit with Photodetection Function

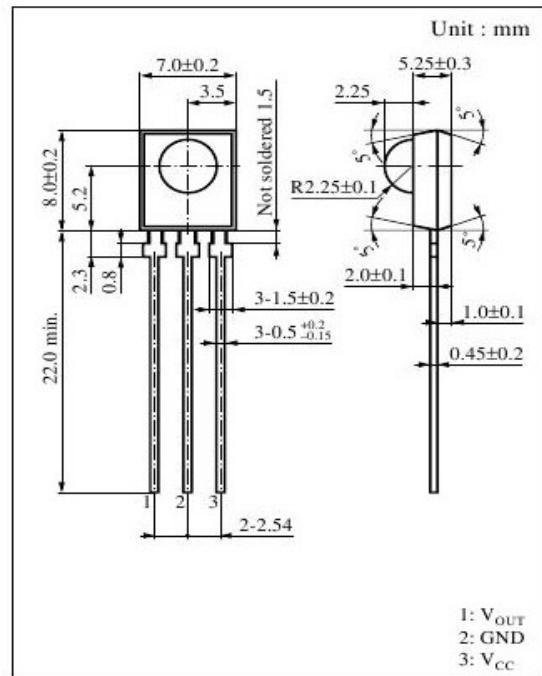
For infrared remote control systems

■ Features

- Extension distance is 8 m or more
- External parts not required
- Adoption of visible light cutoff resin

■ Absolute Maximum Ratings (Ta = 25°C)

Parameter	Symbol	Ratings	Unit
Power supply voltage	V _{CC}	-0.5 to +7	V
Power dissipation	P _D	200	mW
Operating ambient temperature	T _{opr}	-20 to +75	°C
Storage temperature	T _{stg}	-40 to +100	°C



■ Main Characteristics (Ta = 25°C V_{CC} = 5V)

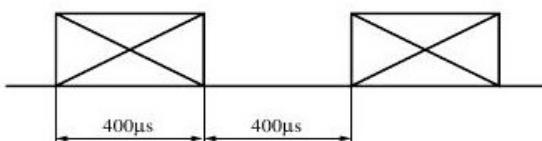
Parameter	Symbol	Conditions	min	typ	max	Unit
Operating supply voltage	V _{CC}		4.7	5.0	5.3	V
Current consumption	I _{CC}	Note 3	1.8	2.4	3.0	mA
Maximum reception distance	L _{max}	Note 1	8	10		m
Low-level output voltage	V _{OL}	Note 2		0.35	0.5	V
High-level output voltage	V _{OH}	Note 3	4.8	5.0	V _{CC}	V
Low-level pulse width	T _{WL}	Note 1	200	400	600	μs
High-level pulse width	T _{WH}	Note 1	200	400	600	μs
Carrier frequency	PNA4601M PNA4602M PNA4608M PNA4610M	f ₀		36.7 38.0 56.9 33.3		kHz

Note 1) Fig. 1 burst wave, L = L_{max}, 16 pulses

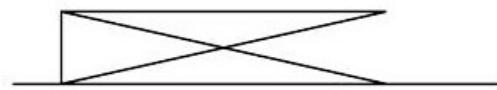
Note 2) Fig. 2 continuous wave, L ≤ L_{max}

Note 3) Light shut off condition

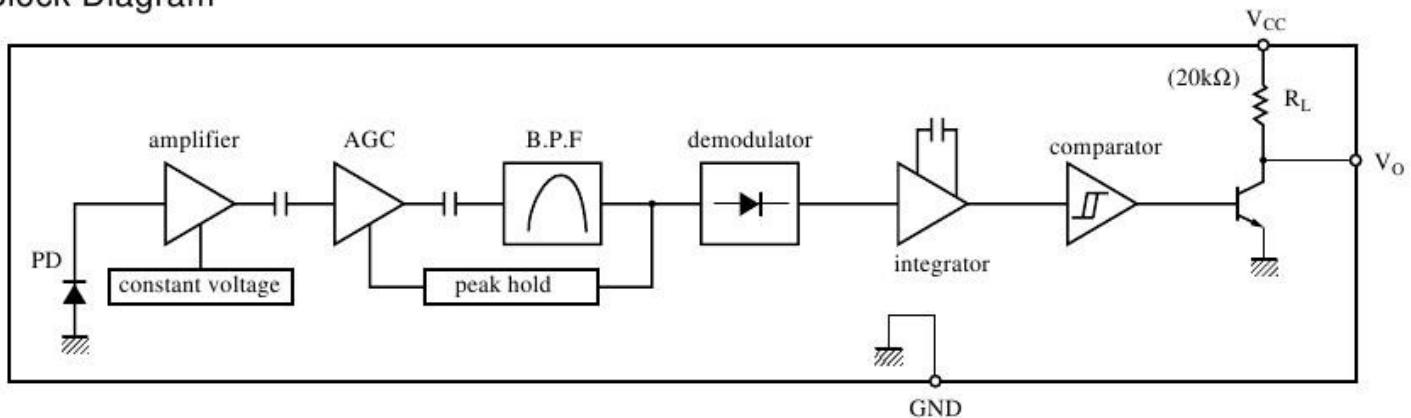
Carrier wave : f₀



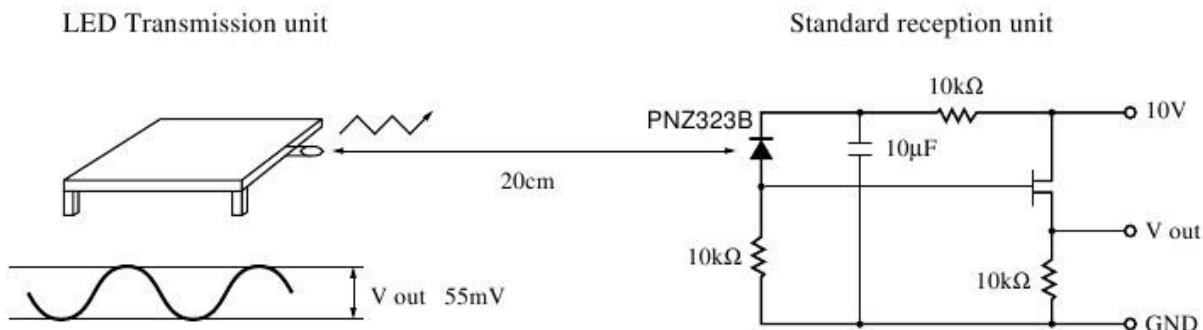
Carrier wave : f₀



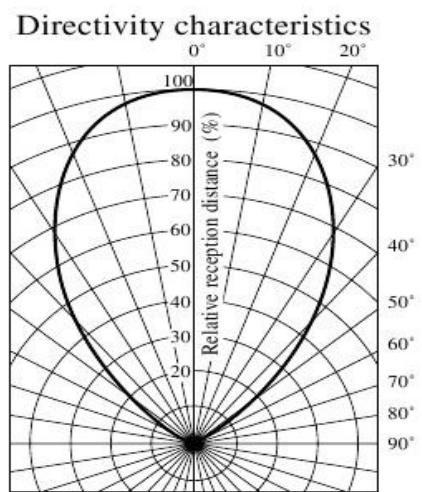
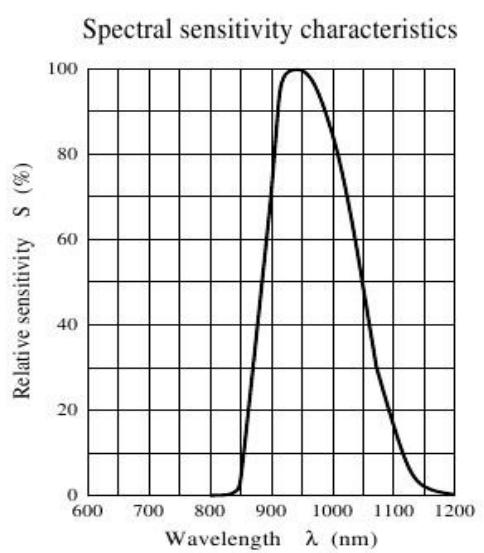
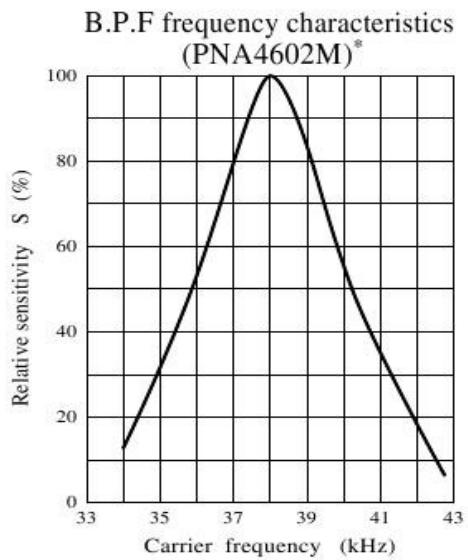
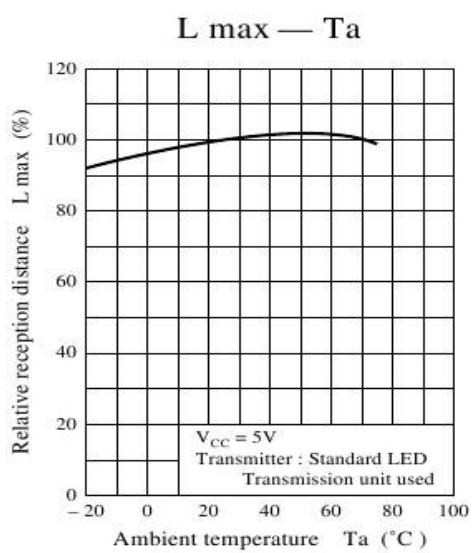
■ Block Diagram



■ Panasonic Transmitter Specifications



- The light output of the LED transmission unit is adjusted so that the transmission output (V_{out}) of the standard reception unit will be 55 mV when the transmission waveform (duty = 50%) is output from the LED transmission unit. Here, infrared sensitivity (SIR) of PNZ323B is 0.53 μA when emission illuminance (H) is 12.45 $\mu\text{W}/\text{cm}^2$.
- The maximum reception distance under these specifications is an assurance that T_{WH} and T_{WL} values will be within the tolerance ranges when 16 consecutive pulses of an optical output equivalent to the maximum reception distance are transmitted by the above transmission unit (The maximum reception distance is measured in the dark without external disturbance noise.)

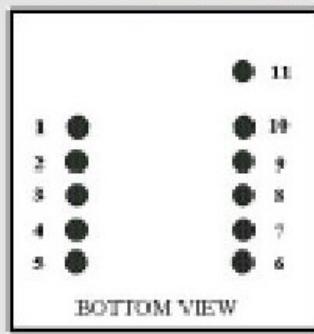


ID SERIES DATASHEET Mar 01, 2005

ID-2/ID-12 Brief Data

The ID2, ID12 and ID20 are similar to the obsolete ID0, ID10 and ID15 MK(ii) series devices, but they have extra pins that allow Magnetic Emulation output to be included in the functionality. The ID-12 and ID-20 come with internal antennas, and have read ranges of 12+ cm and 16+ cm, respectively. With an external antenna, the ID-2 can deliver read ranges of up to 25 cm. All three readers support ASCII, Wiegand26 and Magnetic ABA Track2 data formats.

ID2 / ID12 / ID20 PIN-OUT



1. GND
2. RES (Reset Bar)
3. ANT (Antenna)
4. ANT (Antenna)
5. CP
6. Future
7. +/- (Format Selector)
8. D1 (Data Pin 1)
9. D0 (Data Pin 0)
10. LED (LED / Beeper)
11. +5V



Operational and Physical Characteristics

Parameters	ID-2	ID-12	ID-20
Read Range	N/A (no internal antenna)	12+ cm	16+ cm
Dimensions	21 mm x 19 mm x 6 mm	26 mm x 25 mm x 7 mm	40 mm x 40 mm x 9 mm
Frequency	125 kHz	125 kHz	125 kHz
Card Format	EM 4001 or compatible	EM 4001 or compatible	EM 4001 or compatible
Encoding	Manchester 64-bit, modulus 64	Manchester 64-bit, modulus 64	Manchester 64-bit, modulus 64
Power Requirement	5 VDC @ 13mA nominal	5 VDC @ 30mA nominal	5 VDC @ 65mA nominal
I/O Output Current	+/-200mA PK	-	-
Voltage Supply Range	+4.6V through +5.4V	+4.6V through +5.4V	+4.6V through +5.4V

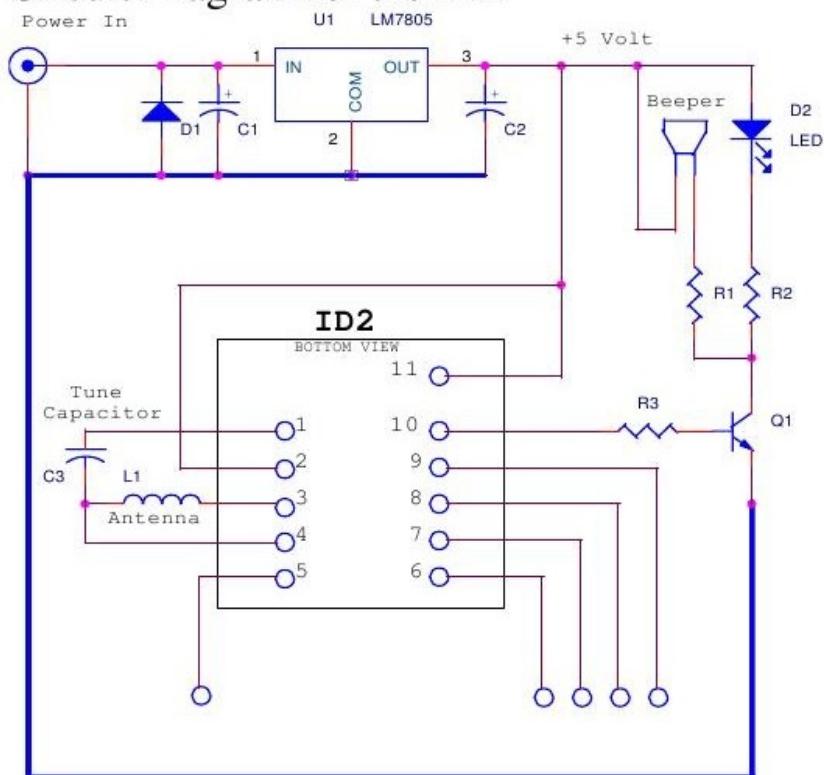
Pin Description & Output Data Formats

Pin No.	Description	ASCII	Magnet Emulation	Wiegand26
Pin 1	Zero Volts and Tuning Capacitor Ground	GND 0V	GND 0V	GND 0V
Pin 2	Strap to +5V	Reset Bar	Reset Bar	Reset Bar
Pin 3	To External Antenna and Tuning Capacitor	Antenna	Antenna	Antenna
Pin 4	To External Antenna	Antenna	Antenna	Antenna
Pin 5	Card Present	No function	Card Present *	No function

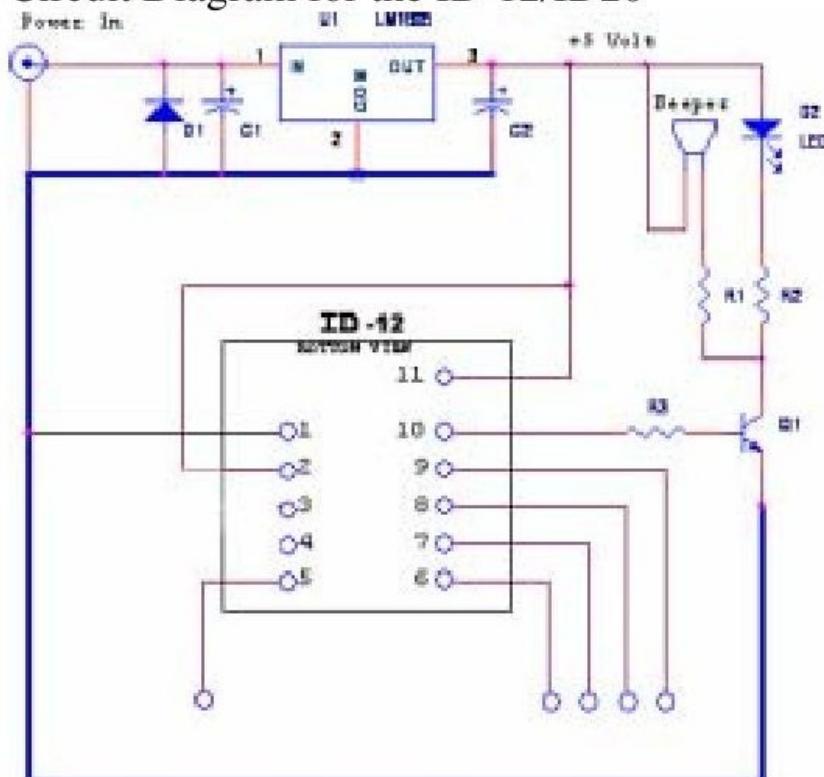
Pin 6	Future	Future	Future	Future
Pin 7	Format Selector (+/-)	Strap to GND	Strap to Pin 10	Strap to +5V
Pin 8	Data 1	CMOS	Clock *	One Output *
Pin 9	Data 0	TTL Data (inverted)	Data *	Zero Output *
Pin 10	3.1 kHz Logic	Beeper / LED	Beeper / LED	Beeper / LED
Pin 11	DC Voltage Supply	+5V	+5V	+5V

* Requires 4K7 Pull-up resistor to +5V

Circuit Diagram for the ID2



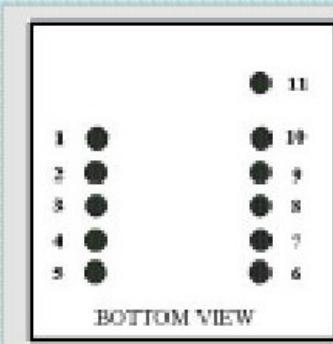
Circuit Diagram for the ID-12/ID20



ID-2RW/ID-12RW Brief Data

The ID2-RW, ID12-RW and ID15-RW are a new series of Read/Write modules for the Temec Q5 tag. It has full functionality including password. They contain built-in algorithms to assist customers programming the popular Sokymat Unique type tag. Password protection is allowed. Control is via a host computer using a simple terminal program such as hyper terminal or Qmodem.

ID2 / ID12 / ID20 PIN-OUT



- | | |
|----|---------------------|
| 1 | GND |
| 2 | RES (Reset Bar) |
| 3 | ANT (Antenna) |
| 4 | ANT (Antenna) |
| 5 | Future |
| 6 | Program LED |
| 7 | ASCII in |
| 8 | Future |
| 9 | ASCII Out |
| 10 | Read (LED / Beeper) |
| 11 | +5V |



Operational and Physical Characteristics

Parameters	ID-2RW	ID-12RW	ID-20RW
Read Range	N/A (no internal antenna)	12+ cm (Unique Format)	15+ cm (Unique Format)
Dimensions	21 mm x 19 mm x 6 mm	26 mm x 25 mm x 7 mm	40 mm x 40 mm x 9 mm
Frequency	125 kHz	125 kHz	125 kHz
Card Format	Temec Q5555	Temec Q5555	Temec Q5555
Read Encoding	Manchester modulus 64	Manchester modulus 64	Manchester modulus 64
Power Requirement	5 VDC @ 13mA nominal	5 VDC @ 30mA nominal	5 VDC @ 50mA nominal
I/O Output Current	+/-200mA PK	-	-
Voltage Supply Range	+4.6V through +5.4V	+4.6V through +5.4V	+4.6V through +5.4V
Coil Detail	L = 0.6mH - 1.5mH, Q = 15-30	-	-

Description

A simple terminal program such as Qmodem or Hyper-terminal can be used to send commands to the module. The blocks are individually programmable. The command interface is simple to use and easily understood. The programmer also has two types of internal reader. One of these is provided to read Sokymat 'Unique' type tag configuration. The module does not require a MAX232 type chip interface. The module does **not** need an RS232 interface such as a MAX232 IC. The input pin7 goes to the computer through a 4k7 resistor and the output goes to the computer through a 100R resistor.

DATA FORMATS

Output Data Structure – ASCII

STX (02h)	DATA (10 ASCII)	CHECK SUM (2 ASCII)	CR	LF	ETX (03h)
-----------	-----------------	---------------------	----	----	-----------

[The 1byte (2 ASCII characters) Check sum is the “Exclusive OR” of the 5 hex bytes (10 ASCII) Data characters.]

Output Data Structure – Wiegand26

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
P	E	E	E	E	E	E	E	E	E	E	E	O	O	O	O	O	O	O	O	O	O	O	O	O	P
Even parity (E)													Odd parity (O)												

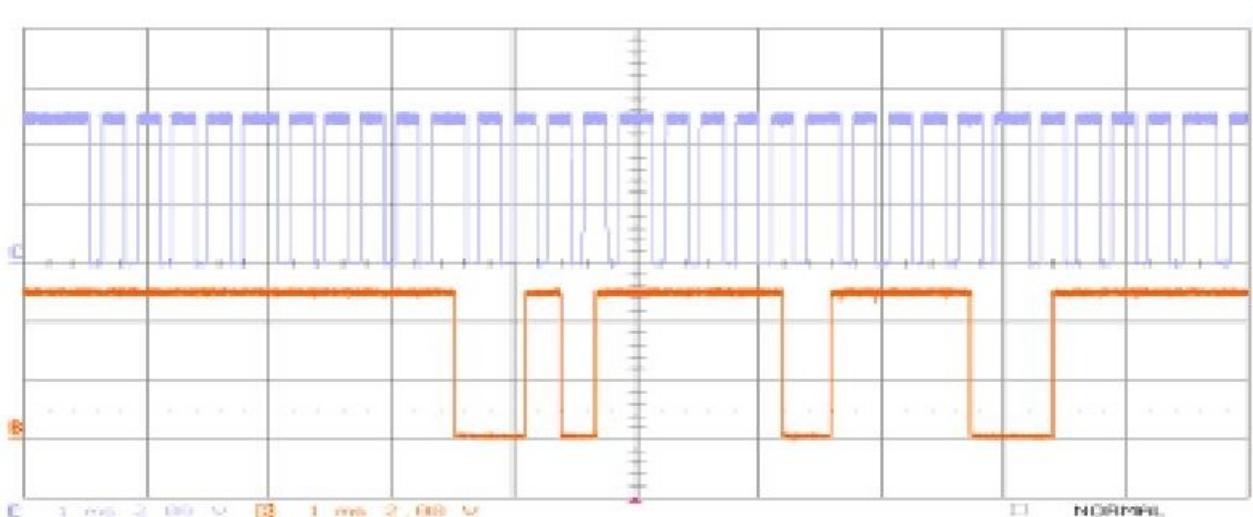
P = Parity start bit and stop bit

Output Data Magnetic ABA Track2

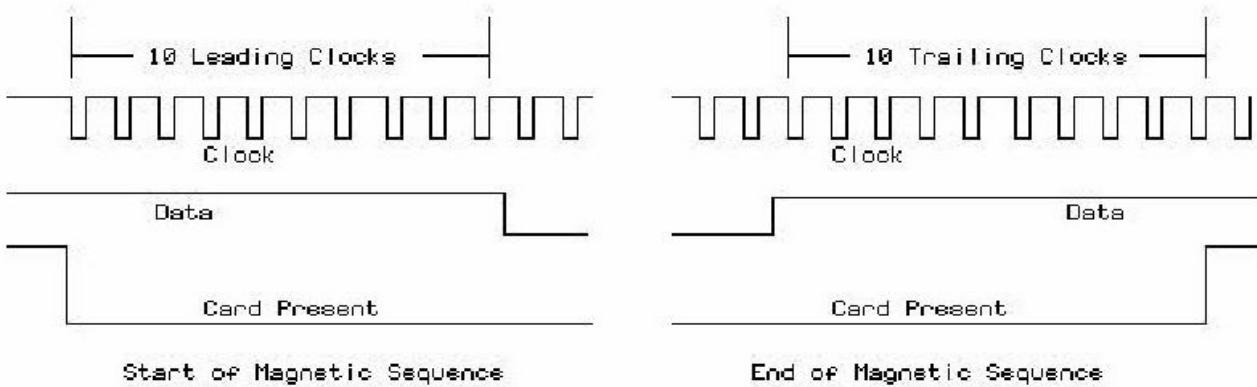
10 Leading Zeros	SS	Data	ES	LCR	10 Ending Zeros
------------------	----	------	----	-----	-----------------

[SS is the Start Character of 11010, ES is the end character of 11111, LRC is the Longitudinal Redundancy Check.]

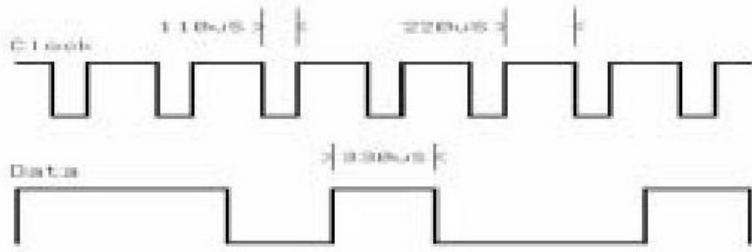
Magnetic Emulation Waveforms



Start and End Sequences For Magnetic Timing



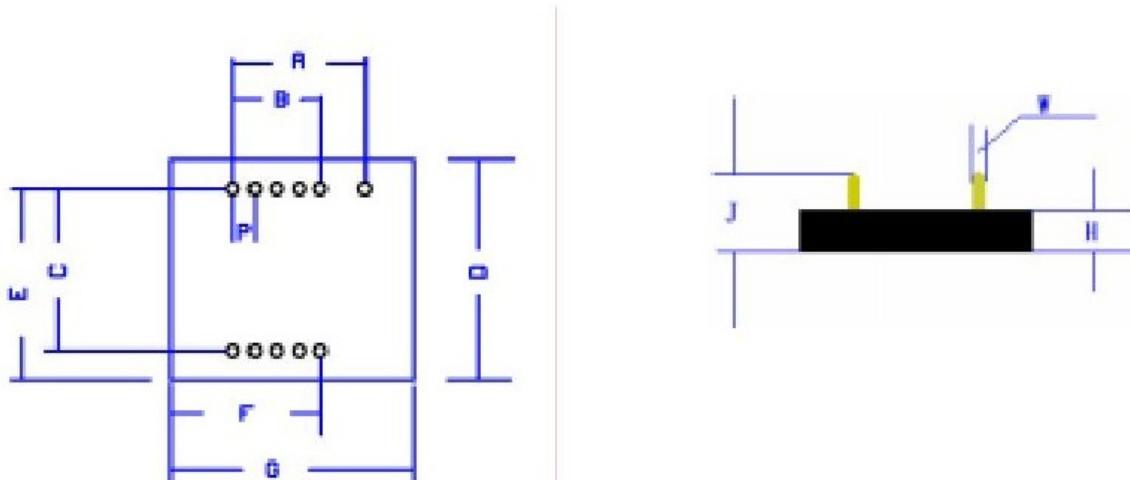
DATA TIMINGS FOR MAGNETIC EMULATION



The magnetic Emulation Sequence starts with the Card Present Line going active (down). There next follows 10 clocks with Zero '0' data. At the end of the 10 leading clocks the start character (11010) is sent and this is followed by the data. At the end of the data the end character is sent followed by the LCR. Finally 10 trailing clocks are sent and the card present line is raised.

The data bit duration is approximately 330uS. The approximate clock duration is 110uS. Because of the symmetry data can be clocked off either the rising or falling edge of the clock.

Dimensions (Top View) (mm)



	ID-0/ID-2wr			ID-10/ID-12wr			ID-15/ID-20wr		
	Nom.	Min.	Max.	Nom.	Min.	Max.	Nom.	Min.	Max.
A	12.0	11.6	12.4	12.0	11.6	12.4	12.0	11.6	12.4
B	8.0	7.6	8.4	8.0	7.6	8.4	8.0	7.6	8.4
C	15.0	14.6	15.4	15.0	14.6	15.4	15.0	14.6	15.4
D	20.5	20.0	21.5	25.3	24.9	25.9	40.3	40.0	41.0
E	18.5	18.0	19.2	20.3	19.8	20.9	27.8	27.5	28.5
F	14.0	13.0	14.8	16.3	15.8	16.9	22.2	21.9	23.1
G	22.0	21.6	22.4	26.4	26.1	27.1	38.5	38.2	39.2
P	2.0	1.8	2.2	2.0	1.8	2.2	2.0	1.8	2.2
H	5.92	5.85	6.6	6.0	5.8	6.6	6.8	6.7	7.0
J	9.85	9.0	10.5	9.9	9.40	10.5	9.85	9.4	10.6
W	0.66	0.62	0.67	0.66	0.62	0.67	0.66	0.62	0.67

Note – measurements do not include any burring of edges.

NOTICE - Innovative Devices reserve the right to change these specifications without prior notice.

Designing Coils for ID2

The recommended Inductance is 1.08mH to be used with an internal tuning capacitor of 1n5. In general the bigger the antenna the better, provided the reader is generating enough field strength to excite the tag. The ID-2 is relatively low power so a maximum coil size of 15x15cm is recommended if it is intended to read ISO cards. If the reader is intended to read glass tags the maximum coil size should be smaller, say 10x10cm.

There is a science to determine the exact size of an antenna but there are so many variables that in general it is best to get a general idea after which a degree of ‘Try it and see’ is unavoidable.

If the reader is located in a position where there is a lot of heavy interference then less range cannot be avoided. In this situation the coil should be made smaller to increase the field strength and coupling.

It is difficult to give actual examples of coils for hand winding because the closeness and tightness of the winding will significantly change the inductance. A professionally wound coil will have much more inductance than a similar hand wound coil.

For those who want a starting point into practical antenna winding it was found that 63 turns on a 120mm diameter former gave an inductance of 1.08mH. For those contemplating adding an additional tuning capacitor it was found that 50 turns on a 120mm diameter former gave 700uH. The wire diameter is not important.

Anybody who wishes to be more theoretical we recommend a trip to the Microchip Website where we found an application sheet for Loop Antennas.

<http://ww1.microchip.com/downloads/en/AppNotes/00831b.pdf>

The Tuning Capacitor

It is recommended that the internal 1n% capacitor is used for tuning, however a capacitor may be also be added externally. The combined capacitance should not exceed 2n7. Do not forget that the choice of tuning capacitor can also substantially affect the quality of your system. The Id12 is basically an ID2 with an internal antenna. The loss in an ID12 series antenna is required to be fairly high to limit the series current. A low Q will hide a lot of the shortcomings of the capacitor, but for quality and reliability and repeatability the following capacitors are recommend.

Polypropylene	Good Readily available. Ensure AC voltage at 125kHz is sufficient.
COG/NPO	Excellent. Best Choice
Silver Mica	Excellent but expensive
Polycarbonate	Good Readily available. Ensure AC voltage at 125kHz is sufficient.

Voltage Working.

A capacitor capable of withstanding the RMS voltage at 125KHz MUST be chosen. The working voltage will depend on the coil design. I suggest the designer start with rugged 1n5 Polypropylene 630v capacitor to do his experiments and the come down to a suitable size/value. The capacitor manufacturer will supply information on their capacitors. Do not simply go by the DC voltage. This means little. A tolerance of 2% is preferable. A tolerance of 5% is acceptable.

Fine Tuning

We recommend using an oscilloscope for fine-tuning. Connect the oscilloscope to observe the 125KHz AC voltage across the coil. Get a sizeable piece of ferrite and bring it up to the antenna loop. If the voltage increases then you need more inductance (or more capacitance). If the voltage decreases as you bring the ferrite up to the antenna then the inductance is too great. If you have no ferrite then a piece of aluminum

sheet may be used for testing in a slightly different way. Opposing currents will flow in the aluminum and it will act as a negative inductance. If the 125kH AC voltage increases as the aluminum sheet approaches the antenna then the inductance is too high. Note it may be possible that the voltage will first maximize then decrease. This simply means that you are near optimum tuning. If you are using ferrite then the coil is a little under value and if you are using an aluminum sheet then the coil is over value.

ID Innovations

Advanced Digital Reader Technology

----Better by Design