Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Επικοινωνιων, Ηλεκτρονικης & Συστηματων Πληροφορικης

Σχεδίαση και υλοποίηση ασύρματου συστήματος μετάδοσης δεδομένων βασισμένο στο πρωτόκολλο **LoRa**

Διπλωματικη Εργασια

του

ΚΩΝΣΤΑΝΤΙΝΟΥ Κ. ΤΖΩΡΤΖΑΚΗ

Επιβλέπων: Παύλος-Πέτρος Σωτηριάδης
Αναπλ. Καθηγητής Ε.Μ.Π.

Εργαστηριο Ηλεκτρονικης
Αθήνα, Οκτώβριος 2017

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων Πληροφορικής
Εργαστήριο Ηλεκτρονικής

Σχεδίαση και υλοποίηση ασύρματου συστήματος μετάδοσης δεδομένων βασισμένο στο πρωτόκολλο **LoRa**

Διπλωματικη Εργασια

του

ΚΩΝΣΤΑΝΤΙΝΟΥ Κ. ΤΖΩΡΤΖΑΚΗ

Επιβλέπων: Παύλος-Πέτρος Σωτηριάδης
Αναπλ. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 19η Οκτωβρίου 2017.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Παύλος-Πέτρος Σωτηριάδης	Κιαμάλ Πεκμεστζή	Συμεών Παπαβασιλείου
Αναπλ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π

Αθήνα, Οκτώβριος 2017

(Υπογραφή)

.....

Κωνσταντινος Κ. Τζωρτζακης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων Πληροφορικής
Εργαστήριο Ηλεκτρονικής

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής μου εργασίας, κύριο Παύλο-Πέτρο Σωτηριάδη για την καθοδήγηση και υποστήριξη του καθώς και για την εμπιστοσύνη που μου έδειξε σε ενα τόσο απαιτητικό και πολυδιάστατο θέμα.

Φυσικά, θα ήθελα να ευχαριστήσω όλα τα παιδιά του Circuits and Systems Group για την βοήθεια τους αλλά και το φιλικό κλίμα που δημιούργησαν και κυρίως τον υποψήφιο διδάκτορα Κωνσταντίνο Παπαφώτη για την επίβλεψη και καθοδηγησή του.

Τέλος, ευχαριστώ την οικογένεια μου αλλά και τους φίλους μου για την στήριξη ώστε να ανταπεξέλθω στις δυσκολίες που είχε η εργασία αυτή.

Περίληψη

Στην παρόυσα εργασία γίνεται η σχεδίαση και υλοποίηση ενός ασύρματου δικτύου μετάδοσης δεδομένων βασισμένο στο πρωτόκολλο LoRa. Η πληροφορία αφορά δεδομένα απο αισθητήρες θερμοκρασίας , υγρασίας , φωτεινότητας, μονοξειδίου του άνθρακα , καπνού , μεθανίου και αλκοόλ. Τα δεδομένα αποθηκεύονται και παρουσιάζονται στο διαδίκτυο με χρήση GSM.

Λέξεις Κλειδιά

ασύρματο δίκτυο, επικοινωνία ,αισθητήρες αερίων, LoRa , GSM

Abstract

This work presents the design and implementation of a wireless data transmission system based on the LoRa protocol. The information concerns data from temperature, humidity, luminance, carbon monoxide, smoke, methane and alcohol sensors. The data is stored and presented on the Internet using GSM.

Keywords

LoRa, IoT, gas sensors ,wireless networks

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Στόχος της εργασίας

Στόχος της συγκεκριμένης εργασίας είναι η ασύρματη μετάδοση δεδομένων που συλλέγονται απο αισθητήρες σε ενα κεντρικό base station που θα κάνει την πληροφορία προσβάσιμη στους ενδιαφερόμενους χρήστες. Χρησιμοποιήθηκαν αισθητήρες θερμοκρασίας , υγρασίας , φωτεινότητας, μονοξειδίου του άνθρακα , καπνού , μεθανίου και αλκοόλ. Η μετάδοση της πληροφορίας πραγματοποιείται με το πρωτόκολλο επικοινωνίας LoRa και η αποθήκευση και παρουσίαση των δεδομένων βρίσκεται στο διαδίκτυο με την αποστολή μέσω GSM.

1.2 Η σημασία του Internet of Things (IoT)

Είναι η διασύνδεση αντικειμένων της καθημερινότητας δηλαδή συσκευών, οχημάτων , κτιρίων και άλλων αντικειμένων (επίσης αναφερόμενων ως ¨συνδεδεμένες συσκευές' και ¨έξυπνες συσκευές¨) που ενσωματώνονται με ηλεκτρονικά, λογισμικά, αισθητήρες, ενεργοποιητές και συνδεσιμότητα δικτύου με στόχο την αποθήκευση, επεξεργασία και ανταλλαγή δεδομένων.

Το IoT επιτρέπει στα αντικείμενα να ανιχνεύονται ή να ελέγχονται εξ αποστάσεως σε όλη την υπάρχουσα υποδομή δικτύου, δημιουργώντας ευκαιρίες για πιο άμεση ενσωμάτωση του φυσικού κόσμου σε συστήματα βασισμένα σε υπολογιστές και οδηγώντας σε βελτιωμένη αποτελεσματικότητα, ακρίβεια και οικονομικό όφελος. Όταν το IoT συμπληρώνεται με αισθητήρες και ενεργοποιητές, η τεχνολογία αποτελεί παράδειγμα γενικότερης κλάσης των κυβερνο-φυσικών συστημάτων, το οποίο περιλαμβάνει επίσης τεχνολογίες όπως τα έξυπνα δίκτυα, οι εικονικές μονάδες παραγωγής ενέργειας, τα έξυπνα σπίτια, οι έξυπνες μεταφορές και οι έξυπνες πόλεις. Κάθε αντικείμενο είναι μοναδικά αναγνωρίσιμο μέσω του ενσωματωμένου συστήματος πληροφορικής του, αλλά είναι σε θέση να λειτουργήσει μέσα στην υπάρχουσα υποδομή του Διαδικτύου. Οι εμπειρογνώμονες εκτιμούν ότι το IoT θα αποτελείται από περίπου 30 δισεκατομμύρια αντικείμενα μέχρι το 2020.

Το IoT προσφέρει προηγμένη συνδεσιμότητα συσκευών, συστημάτων και υπηρεσιών που υπερβαίνει τις επικοινωνίες μηχανής με μηχανή (M2M) και καλύπτει μια ποικιλία πρωτοκόλλων, τομέων και εφαρμογών. Η διασύνδεση αυτών των ενσωματωμένων συσκευών (συμπεριλαμ-

βανομένων των έξυπνων αντικειμένων) αναμένεται να οδηγήσει σε αυτοματοποίηση σε όλα σχεδόν τα πεδία, επιτρέποντας επίσης προηγμένες εφαρμογές όπως ένα έξυπνο δίκτυο και επεκτείνοντας σε τομείς όπως οι έξυπνες πόλεις.

"Things", κατά την έννοια του IoT, μπορούν να αναφέρονται σε μια ευρεία ποικιλία συσκευών, όπως εμφυτεύματα παρακολούθησης της καρδιάς, αναμεταδότες βιοκαυσίμων σε ζώα εκμετάλλευσης, ηλεκτρικά νερά σε παράκτια ύδατα, αυτοκίνητα με ενσωματωμένους αισθητήρες, / Παρακολούθηση τροφίμων / παθογόνων ή συσκευών πεδίου που βοηθούν τους πυροσβέστες σε επιχειρήσεις έρευνας και διάσωσης . Οι μελετητές υποδεικνύουν ότι τα"Things" είναι ένα ¨αναπόσπαστο μείγμα υλικού, λογισμικού, δεδομένων και υπηρεσιών¨.

Οι κλάδοι των εφαρμογών ποικίλουν , μερικοί από τους οποίους είναι οι εξής :

1. Υγειονομική Περίθαλψη
2. Τηλεπικοινωνίες
3. Βιομηχανική Παραγωγή
4. Μεταφορές
5. Ενέργεια

1.3 Συνεισφορά της εργασίας

Η συγκεκριμένη εργασία ως ένα κομμάτι του IoT , αποτελεί την λύση σε πολλά προβλήματα :

1.3.1 Πρόβλεψη και άμεση ανίχνευση πυρκαγιών

Ειδικά στην Ελλάδα , αλλά και στις περισσότερες χώρες, οι δασικές πυρκαγιές αποτελούν μείζον πρόβλημα. Συγκεκριμένα ,αρκετοί είναι οι λόγοι για τους οποίους τα ελληνικά δάση είναι ευάλωτα στις πυρκαγιές. Αρχικά ,τα μεγάλης διάρκειας θερμά και ξηρά καλοκαίρια, οι ήπιοι χειμώνες .Επιπρόσθετα, οι δυνατοί άνεμοι, το έντονο ανάγλυφο των δασικών εδαφών αλλά και η εύφλεκτη ξηροφυτική βλάστηση. Τελευταίο αλλά ίσως και πιο σημαντικό, η έντονη ανθρώπινη δραστηριότητα και η ελλιπής διαχείριση των εύφλεκτων αυτών δασών οδηγούν στην δημιουργία του παραπάνω προβλήματος.

1.3.2 Παρακολούθηση αγροτικών περιοχών σε πραγματικό χρόνο

Η χρήση του IoT και κατ επέκταση της συγκεκριμένης εργασίας εκσυγχρονίζει τη γεωργική βιομηχανία και επιτρέπει στους αγρότες να αντιμετωπίσουν τις τεράστιες προκλήσεις που αντιμετωπίζουν. Η βιομηχανία πρέπει να ξεπεράσει την αυξανόμενη έλλειψη νερού, την περιορισμένη διαθεσιμότητα των εδαφών, τη δυσκολία διαχείρισης του κόστους, ενώ ταυτόχρονα να ικανοποιεί τις αυξανόμενες ανάγκες κατανάλωσης ενός παγκόσμιου πληθυσμού που αναμένεται να αυξηθεί κατά 70/100 μέχρι το 2050. (Αναφορά: Οργανισμός Τροφίμων και Γεωργίας των Ηνωμένων Εθνών) Οι σημερινές αγροτικές βιομηχανίες μπορούν, για παράδειγμα, να

εκμεταλλευτούν το IoT για την απομακρυσμένη παρακολούθηση αισθητήρων που ανιχνεύουν την υγρασία του εδάφους, την ανάπτυξη των καλλιεργειών και τα επίπεδα ζωοτροφών, την εξ αποστάσεως διαχείριση και τον έλεγχο των έξυπνων συνδεδεμένων θεριζοαλωνιστικών μηχανημάτων και εξοπλισμού άρδευσης. Σε συνδυασμό με επιπλέον πληροφορίες , όπως οι υπηρεσίες καιρού, για την παροχή νέων στοιχείων και τη βελτίωση της διαδικασίας λήψης αποφάσεων.

1.3.3 Μείωση της ατμοσφαιρικής ρύπανσης

Η παρακολούθηση της εκπομπής αερίων σε αστικό περιβάλλον αλλά και η προβολή των δεδομένων στους πολίτες, θα οδηγήσει σε μείωση της ατμοσφαιρική ρύπανσης με τρόπους όπως η μείωση των οχημάτων στις ώρες αιχμής λόγω ευαισθητοποίησης των πολιτών. Σχετικά με τα αέρια που μας ενδιαφέρουν:

- Διοξείδιο του αζώτου (NO2): είναι ένα αέριο που παράγεται από την ταχεία οξείδωση του NO, το οποίο παράγεται με την καύση ορυκτών καυσίμων στα οχήματα και τη βιομηχανία. Είναι ένα τοξικό και ερεθιστικό αέριο που επηρεάζει το αναπνευστικό σύστημα και ενθαρρύνει επίσης την παραγωγή νιτρικού οξέος (HNO3) υπεύθυνης για όξινη βροχή.

- Διοξείδιο του άνθρακα (CO2): είναι φυσικό αέριο που υπάρχει στην ατμόσφαιρα μας. Μαζί με τους υδρατμούς και άλλα αέρια είναι ένα από τα αέρια θερμοκηπίου που ρυθμίζουν τη θερμοκρασία της Γης. Η υπερβολική παραγωγή ως αποτέλεσμα της αυξημένης χρήσης ορυκτών καυσίμων θα μπορούσε να έχει άμεσο αντίκτυπο στην αλλαγή του κλίματος.

- Μονοξείδιο του άνθρακα (CO): παράγεται σε ατελή καύση, δηλαδή όταν μέρος του καυσίμου δεν αντιδρά πλήρως λόγω έλλειψης οξυγόνου. Ο κίνδυνος για τον άνθρωπο και τα ζώα, όταν βρεθεί στην αιμοσφαιρίνη του αίματος, εμποδίζει τη μεταφορά οξυγόνου, η οποία μπορεί να είναι θανατηφόρα. Παρόλο που ο ανοικτός χώρος αραιώνεται εύκολα, η εκπομπή CO από τους κινητήρες των αυτοκινήτων που προκαλούνται από συμφόρηση μπορεί να έχει ρυθμούς 50-100ppm, οι οποίοι είναι επικίνδυνοι.

- Μεθάνιο (CH4): παράγεται όταν το οργανικό υλικό αποσυντίθεται σε περιβάλλον με χαμηλό οξυγόνο. Ως διοξείδιο του άνθρακα, είναι ένα αέριο θερμοκηπίου, οπότε η αύξηση του μπορεί να συμβάλει στην υπερθέρμανση του πλανήτη.

- Σουλφίδιο του υδρογόνου (H2S): εκπέμπεται στην ατμόσφαιρα από διάφορες βιομηχανίες, όπως το χαρτί. Είναι ιδιαίτερα επικίνδυνο επειδή είναι ένα πολύ τοξικό αέριο και είναι πρόδρομος του διοξειδίου του θείου, ένα από τα αέρια στις διαδικασίες σχηματισμού όξινης βροχής. Επιπλέον, αυτό το αέριο είναι ιδιαίτερα ενοχλητικό λόγω της άσχημης οσμής του.

- Υδρογονάνθρακες (αιθανόλη, προπάνιο, βουτάνιο, ισοβουτάνιο, τολουόλιο): προέρχονται από διάφορες πηγές, όπως η κακή καύση βενζίνης και ντίζελ ή ενδομυελικές διεργασίες. Είναι, μεταξύ άλλων, υπεύθυνες για το φαινόμενο του θερμοκηπίου και συμβάλλουν στην παραγωγή αναπνευστικών προβλημάτων.

- Όζον (O3): είναι ένα φυσικό συστατικό που μπορεί να βρεθεί σε επίπεδο θάλασσας με συγκέντρωση 0,01 mg / kg. Εντούτοις, με έντονη ηλιακή ακτινοβολία και υψηλή μόλυνση που προέρχεται από οχήματα, η συγκέντρωσή της μπορεί να φτάσει έως και 0,1 mg / kg επικίνδυνη. Σε αυτή την αναλογία, τα φυτά μπορεί να επηρεαστούν και ο άνθρωπος μπορεί να εμφανίσει ερεθισμό των ρινικών διόδων και του λαιμού και της ξηρότητας στην επένδυση των αναπνευστικών οδών.

.

Κεφάλαιο 2

Ασύρματες επικοινωνίες

2.1 Γιατί θέλουμε ασύρματες επικοινωνίες

Η επικοινωνία είναι η διαδικασία με την οποία ένας πομπός Α (άνθρωπος ή ομάδα) μεταβιβάζει πληροφορίες, σκέψεις, ιδέες ή συναισθήματα σε ένα δέκτη Β (άνθρωπος ή ομάδα) με στόχο να ενεργήσει πάνω του με τρόπο ώστε να προκαλέσει σε αυτόν την εμφάνιση ιδεών, πράξεων ή συναισθημάτων και σε τελική ανάλυση να επηρεάσει την κατάστασή του και τη συμπεριφορά του.

Η επικοινωνία είναι μια διαδικασία ανταλλαγής μηνυμάτων που δεν συμβαίνει απαραίτητα μεταξύ ανθρώπινων όντων, αλλά κάθε οργανισμού ή μηχανής που είναι σε θέση να λάβει και να στείλει μηνύματα ή σήματα που επενεργούν στην πνευματική ή φυσική του κατάσταση ή στη συμπεριφορά του. Η επικοινωνία μπορεί να είναι είτε αυθόρμητη και φυσική είτε (όταν αφορά ανθρώπινη κατασκευή) προσχεδιασμένη και κωδικοποιημένη συνειδητά και προσεκτικά.

Επομένως μέσω της ασύρματης επικοινωνίας διευκολύνεται σε μεγάλο βαθμό η διαδικασία αυτή. Αυτό συμβαίνει διότι αυξάνεται σημαντικά η απόσταση μεταξύ πομπού και δέκτη καθώς και η ταχύτητα της επικοινωνίας. Τέλος μειώνεται η ανάγκη χρήσης άλλων υλικών όπως τα καλώδια.

Συγκεκριμένα, ως ασύρματο δίκτυο χαρακτηρίζεται το τηλεπικοινωνιακό δίκτυο, συνήθως τηλεφωνικό ή δίκτυο υπολογιστών, το οποίο χρησιμοποιεί, ραδιοκύματα ως φορείς πληροφορίας. Τα δεδομένα μεταφέρονται μέσω ηλεκτρομαγνητικών κυμάτων, με συχνότητα φέροντος η οποία εξαρτάται κάθε φορά από τον ρυθμό μετάδοσης δεδομένων που απαιτείται να υποστηρίζει το δίκτυο. Η ασύρματη επικοινωνία, σε αντίθεση με την ενσύρματη, δεν χρησιμοποιεί ως μέσο μετάδοσης κάποιον τύπο καλωδίου. Σε παλαιότερες εποχές τα τηλεφωνικά δίκτυα ήταν αναλογικά, αλλά σήμερα όλα τα ασύρματα δίκτυα βασίζονται σε ψηφιακή τεχνολογία και, επομένως, κατά μία έννοια, είναι ουσιαστικώς δίκτυα υπολογιστών.

2.2 Ασύρματα πρωτόκολλα

Ως πρωτόκολλο επικοινωνίας ορίζεται ένα σύνολο κανόνων συμφωνημένων και από τα δυο επικοινωνούντα μέρη και που εξυπηρετούν την μεταξύ τους ανταλλαγή πληροφοριών. Το πρω-

τόκολλο επικοινωνίας είναι δηλαδή μια δέσμη κανόνων στους οποίους στηρίζεται η επικοινωνία των συσκευών (συνήθως, αλλά όχι πάντα, υπολογιστών) σε ένα δίκτυο. Οι κανόνες αυτοί καθορίζουν τη μορφή, το χρόνο και τη σειρά μετάδοσης των πληροφοριών στο δίκτυο. Εκτελούν, επίσης, έλεγχο και διόρθωση σφαλμάτων στη διάρκεια μετάδοσης των πληροφοριών. Μερικά από τα πιο γνωστά δίκτυα είναι:

- Wifi: Η ονομασία WiFi χρησιμοποιείται για να προσδιορίσει τις συσκευές WLAN που βασίζονται στην προδιαγραφή IEEE 802.11 b/g/n και εκπέμπουν σε συχνότητες 2.4GHz. Ωστόσο το WiFi έχει επικρατήσει και ως όρος αναφερόμενος συνολικά στα ασύρματα τοπικά δίκτυα. Συνήθεις εφαρμογές του είναι η παροχή ασύρματων δυνατοτήτων πρόσβασης στο Internet, τηλεφωνίας μέσω διαδικτύου (VoIP) και διασύνδεσης μεταξύ ηλεκτρονικών συσκευών όπως τηλεοράσεις, ψηφιακές κάμερες, DVD Player και ηλεκτρονικοί υπολογιστές. Σε φορητές ηλεκτρονικές συσκευές το 802.11 βρίσκει εφαρμογές ασύρματης μετάδοσης, όπως π.χ. στη μεταφορά φωτογραφιών από ψηφιακές κάμερες σε υπολογιστές για περαιτέρω επεξεργασία και εκτύπωση, αν και σε αυτόν τον τομέα έχει υποσκελιστεί από το πρωτόκολλο Bluetooth για τα πολύ μικρότερης εμβέλειας ασύρματα προσωπικά δίκτυα.

- Bluetooth: Το Bluetooth είναι ένα βιομηχανικό πρότυπο για ασύρματα προσωπικά δίκτυα υπολογιστών Wireless Personal Area Network ,WPAN. Πρόκειται για μια ασύρματη τηλεπικοινωνιακή τεχνολογία μικρών αποστάσεων, η οποία μπορεί να μεταδώσει σήματα μέσω μικροκυμάτων σε ψηφιακές συσκευές. Επομένως το Bluetooth είναι ένα πρωτόκολλο το οποίο παρέχει προτυποποιημένη, ασύρματη επικοινωνία ανάμεσα σε PDA, κινητά τηλέφωνα, φορητοί υπολογιστές, προσωπικοί υπολογιστές, εκτυπωτές, καθώς και ψηφιακές φωτογραφικές μηχανές ή ψηφιακές κάμερες, μέσω μιας ασφαλούς, φθηνής και παγκοσμίως διαθέσιμης χωρίς ειδική άδεια ραδιοσυχνότητας μικρής εμβέλειας. Από τεχνικής άποψης το Bluetooth είναι ένα πρωτόκολλο ασύρματης δικτύωσης σε φυσικό επίπεδο, υποεπίπεδο MAC και, προαιρετικά, υποεπίπεδο LLC.

- GSM:Το Global System for Mobile communications (Παγκόσμιο Σύστημα Κινητών Επικοινωνιών), συντμ. GSM είναι ένα κοινό Ευρωπαϊκό ψηφιακό σύστημα κινητής τηλεφωνίας. Το Ευρωπαϊκό Τηλεπικοινωνιακό Συμβούλιο (European Telecommunications Standards Institute) το 1982, άρχισε την μελέτη για την δημιουργία ενός κοινού Ευρωπαϊκού ψηφιακού συστήματος κινητής τηλεφωνίας δεύτερης γενιάς (2G). Αυτό το σύστημα ονομάστηκε αρχικά Group Special Mobile (GSM).ToGSM είναι ένα κυψελοειδές ψηφιακό σύστημα κινητής τηλεφωνίας δεύτερης γενιάς (2G), το οποίο χρησιμοποιεί ηλεκτρομαγνητικά σήματα και την τεχνική πολλαπλής πρόσβασης με διαχωρισμό του διαθέσιμου φάσματος συχνοτήτων σε ένα αριθμό καναλιών και την διαίρεση αυτών σε χρονοθυρίδες για την μετάδοση σημάτων.

Κεφάλαιο 3

LoRa

Το κύριο αντικείμενο της διπλωματικής εργασίας είναι η χρήση του πρωτοκόλλου LoRa για την δημιουργία συστήματος μεταφοράς δεδομένων.

3.1 Τι είναι το LoRa

- Μεγάλης εμβέλειας
- Μικρής κατανάλωσης ισχύος
- Ασύρματο πρωτόκολλο επικοινωνίας

Ιδανική τεχνλογία για κατασκευή IoT δικτύων σε παγκόσμιο επίπεδο. Δημιουργήθηκε το 2015.

- Μεγάλη εμβέλεια :

  Δυο κόμβοι LoRa διαθέτουν την δυνατότητα ασύρματης επικοινωνίας μέχρι και 15 km , ενω βρίσκονται σε ημιαστικό περιβάλλον.

- Μικρή κατανάλωση ισχύος:

  Το πρωτόκολλο είναι σχεδιασμένο για λειτουργίες χαμηλής κατανάλωσης, που επιτρέπουν την χρήση μπαταρίας με διάρκεια ζωής 10 έτη.

3.1.1 Τεχνικά χαρακτηριστικά

- Διαμόρφωση:

  Η διαμόρφωση που χρησιμοποιεί το LoRa είναι γνωστή ως CSS διαμόρφωση (Chirp Spread Spectrum modulaion).

  Στις ψηφιακές επικοινωνίες, το chirp spread spectrum(CSS) είναι μια τεχνική εξάπλωσης φάσματος που χρησιμοποιεί ευρυζωνικές γραμμικές συχνότητες που διαμορφώνονται με συχνότητα για να κωδικοποιήσουν τις πληροφορίες.

Στις τηλεπικοινωνίες και την ραδιοεπικοινωνία, οι τεχνικές διασποράς φάσματος είναι μέθοδοι με τις οποίες ένα σήμα (π.χ. ένα ηλεκτρικό, ηλεκτρομαγνητικό ή ακουστικό σήμα) που δημιουργείται με ένα συγκεκριμένο εύρος ζώνης διασκορπίζεται σκόπιμα στην περιοχή συχνοτήτων, με αποτέλεσμα ένα σήμα με μεγαλύτερο εύρος ζώνης. Αυτές οι τεχνικές χρησιμοποιούνται για διάφορους λόγους, όπως η καθιέρωση ασφαλών επικοινωνιών, η αύξηση της αντίστασης σε φυσικές παρεμβολές, ο θόρυβος και η εμπλοκή, η πρόληψη της ανίχνευσης και ο περιορισμός της πυκνότητας ροής ισχύος.

Ένα chirp είναι ένα σήμα στο οποίο η συχνότητα αυξάνεται (up-chirp) ή μειώνεται (down-chirp) με το χρόνο. Σε ορισμένες πηγές, ο όρος chirp χρησιμοποιείται εναλλακτικά με σήμα σάρωσης.

Όπως συμβαίνει και με άλλες μεθόδους διάχυσης φάσματος, το ραδιοφωνικό φάσμα που χρησιμοποιείται για τη μετάδοση σήματος χρησιμοποιεί όλο το διαθέσιμο εύρος ζώνης για να εκπέμψει ένα σήμα, καθιστώντας το ισχυρό απέναντι στον θόρυβο. Περαιτέρω, επειδή τα chirps χρησιμοποιούν μια ευρεία ζώνη του φάσματος, το φάσμα εξάπλωσης chirp είναι επίσης ανθεκτικό σε διαφυγή πολλαπλών διαδρομών ακόμα και όταν λειτουργεί σε πολύ χαμηλή ισχύ. Εντούτοις, διαφέρει από το φάσμα εξάπλωσης άμεσης ακολουθίας (DSSS) ή το φάσμα εξάπλωσης συχνότητας (FHSS) στο ότι δεν προσθέτει κανένα ψευδοτυχαίο στοιχείο στο σήμα για να το διακρίνει από το θόρυβο στο κανάλι, αντ άυτού εξαρτάται απο την γραμμική φύση του chirp pulse. Επιπρόσθετα, το φάσμα διασποράς είναι ανθεκτικό στο φαινόμενο Doppler, το οποίο είναι χαρακτηριστικό στις εφαρμογές κινητών ραδιοσυχνοτήτων.

- Κρυπτογράφηση: Έχει ενσωματομένη άκρη προς άκρη (end to end) κρυπτογράφηση AES128.

- Αμφίδρομη λειτουργία: Κάθε κόμβος LoRa εχει την δυνατότητα να λειτουργήσει σαν πομπός αλλά και σαν δέκτης.

3.1.2 Ρυθμίσεις λειτουργίας

- Συχνότητα πομπού και δέκτη : 169 MHz, 433 MHz, 868 MHz (Ευρώπη) και 915 MHz (Αμερική) ανάλογα την περιόχη εφαρμογής.

- Spread factor : αφορά στην ευαισθησία του δέκτη , η ρύθμιση sf12 είναι η πιο ευαίσθητη όπου το SNR ειναι στα – 20dB. Αυτό επίσης σημαίνει ότι η διάρκεια μετάδοσης αυξάνεται.

- Εύρος ζώνης : 125-500 kHz, the used bandwidth determine time on air and sensitivity. With 125KHz the sensitivity is better but time on air is longer.

- CR (Coding rate): είναι παράμετρος που καθορίζει τον μηχανισμό διόρθωσης σφάλματος, προσθέτοντας επιπλέον πληροφορία στο μήνυμα. 4/5 είναι η καλύτερη επιλογή, ειδικά όταν το σήμα είναι χαμηλό.

- prlen αναφέρεται στο μέγεθος του preamble σε σύμβολα. (Το preamble αποτελεί ένα είδος επικεφαλίδας και επιτρέπει στον δέκτη να αποκτήσει το σήμα και να συγχρονιστεί με τον πομπό)
- Power level περιγράφει την ισχύ του πομπού σε dB · παίρνει τιμές απο -3dB σε 15dB, στα 14dB λαμβάνουμε μεγαλύτερη ισχύ και καλύτερο link budget (κατανάλωση 44mA/h καθώς και 151dB link budget)
- Data rate: 250bps-50kbps

Από την εξίσωση :

$$R_b = SF \times \frac{BW}{2^{SF}} \times CR \tag{3.1}$$

μπορούμε να υπολογίσουμε το χρήσιμο bit rate.

Σχήμα 3.1: Πλαίσιο LoRa

Ένα πλαίσιο LoRa ξεκινάει με ενα preamble το οποίο ξεκινάει με μια σειρά σταθερών upchirps που καλύπτουν ολόκληρη τη ζώνη συχνοτήτων. Τα τελευταία δύο upchirps κωδικοποιούν τη λέξη συγχρονισμού. Η λέξη συγχρονισμού είναι μια μοναδική τιμή που χρησιμοποιείται για τη διαφοροποίηση των δικτύων LoRa που χρησιμοποιούν τις ίδιες ζώνες συχνοτήτων. Μια συσκευή με μια δεδομένη λέξη συγχρονισμού θα σταματήσει να ακούει μια μετάδοση εάν η λέξη συγχρονισμού που λάβει δεν αντιστοιχεί στην δεδομένη τιμή της.Η λέξη συγχρονισμού ακολουθείται από δύο και ένα τέταρτο downchirps για διάρκεια 2,25 συμβόλων.Η συνολική διάρκεια αυτού του preamble μπορεί να οριστεί μεταξύ 10,25 και 65.539,25 σύμβολα. Μετά το preamble, υπάρχει μια προαιρετική κεφαλίδα. Όταν είναι παρούσα, αυτή η επικεφαλίδα μεταδίδεται με ρυθμό κωδικού 4/8. Αυτό υποδεικνύει το μέγεθος του ωφέλιμου φορτίου (σε bytes), τον κωδικό συντελεστή που χρησιμοποιείται για το τέλος της μετάδοσης και εάν υπάρχει ένα CRC 16-bit για το ωφέλιμο φορτίο ή όχι στο τέλος του πλαισίου. Η κεφαλίδα περιλαμβάνει επίσης ένα CRC για να επιτρέψει στον δέκτη να απορρίψει πακέτα με μη έγκυρες κεφαλίδες.Το μέγεθος ωφέλιμου φορτίου αποθηκεύεται χρησιμοποιώντας ένα byte. Η κεφαλίδα είναι προαιρετική για να επιτρέψει την απενεργοποίηση της σε καταστάσεις όπου δεν είναι απαραίτητο, για παράδειγμα όταν το ωφέλιμο φορτίο είναι αρκετά μεγάλο.

3.1.3 Εφαρμογές

Το LoRa στοχεύει σε αναπτυξιακές εφαρμογές όπου οι τελικές συσκευές:

- έχουν περιορισμένη ανάγκη από ενέργεια (για παράδειγμα μια μπαταρία)

- δεν χρειάζεται να μεταδίδουν περισσότερα από μερικά bytes τη φορά
- Όπου η μεταφορά δεδομένων μπορεί να ξεκινήσει είτε από την τελική συσκευή (ένας αισθητήρας) είτε από μια εξωτερική συσκευή που επιθυμεί να επικοινωνήσει με την τελική συσκευή.

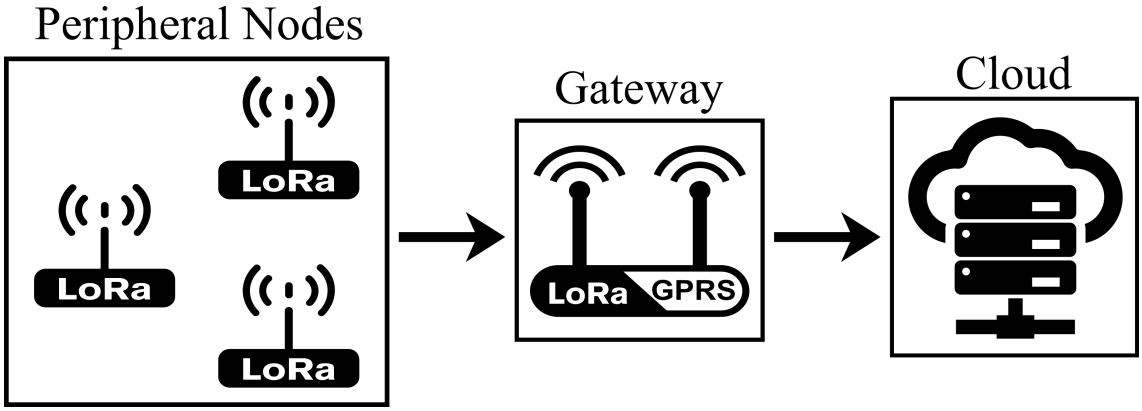
Η φύση του LoRa σε μεγάλη απόσταση και χαμηλής ισχύος το καθιστά έναν ενδιαφέροντα υποψήφιο για χρήση σε έξυπνη τεχνολογία ανίχνευσης σε αστικές υποδομές (όπως παρακολούθηση της υγείας, παρακολούθηση περιβάλλοντος)καθώς και σε βιομηχανικές εφαρμογές.

Κεφάλαιο 4

Το σύστημα

Στη συνέχεια ακολουθεί αναλυτική περιγραφή της υλοποίησης του συστήματος.

4.1 Αρχιτεκτονική του συστήματος



Σχήμα 4.1: Αρχιτεκτονική του συστήματος

Για το σύστημα αναπτύξαμε μια κεντρική αρχιτεκτονική δικτύου για την αποθήκευση και αποστολή των δεδομένων (σχήμα 4.1).

Πιο αναλυτικά, το σύστημα αποτελείται απο τους περιφερειακούς κόμβους και τον κεντρικό κόμβο. Η πληροφορία που συλλέγεται από τους αισθητήρες επεξεργάζεται στον μικροελεγκτή και στην συνέχεια διαβιβάζεται ασύρματα μέσω του πρωτοκόλλου LoRa στον κεντρικό κόμβο.

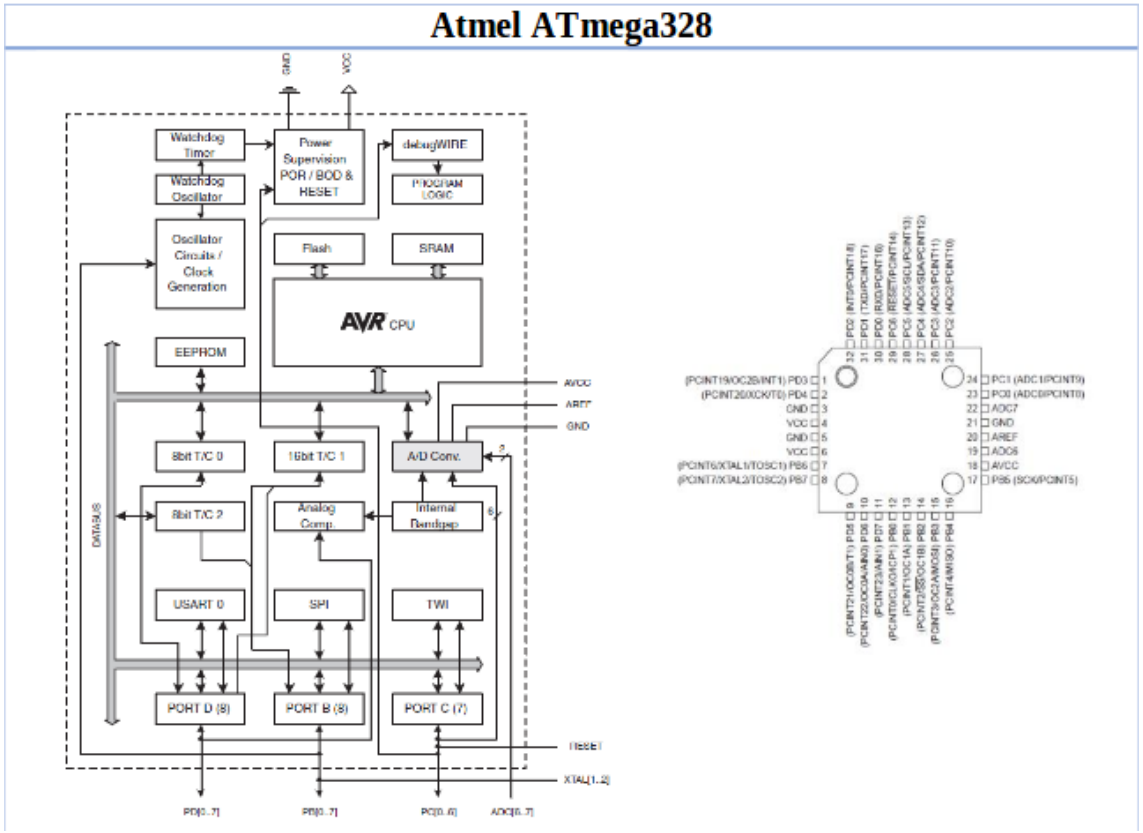
Έπειτα, ο μικροελεγκτής που βρίσκεται στον κεντρικό κόμβο επεξεργάζεται τα δεδομένα που λαμβάνει και κατόπιν μέσω GSM τα στέλνει στο διαδίκτυο.

Τέλος , οι αναλυτικές πληροφορίες που παρέχει κάθε περιφερειακός κόμβος είναι εύκολα προσβάσιμες στον κάθε ενδιαφερόμενο μέσω μιας ιστοσελίδας.

4.2 Περιγραφή Μονάδων του Συστήματος

4.2.1 Ο μικροελεγκτής ATmega328

Ο ATmega328 ανήκει στην οικογένεια των Atmega της Atmel. Είναι ένας μικροελεγκτής 8 bit αρχιτεκτονικής RISC. Ενσωματώνει πλήθος περιφερειακών και (UART, SPI, ADC , I2C) ενώ η ταχύτητα του μπορεί να φτάσει τα 20 Mhz με χρήση εξωτερικού κρυστάλλου. Είναι σχεδιασμένος με στόχο την μικρή κατανάλωση, συγκεκριμένα μπορεί να φτάσει ως 0.75 μA σε λειτουργία Power Save Mode. Στη συγκεκριμένη εργασία έγινε χρήση του ATmega328p και παρακάτω φαίνεται η δομή του.



Σχήμα 4.2: Ο επεξεργαστής ATmega

4.2.2 RN2483 LoRa-Integrated Module

Το RN2483 ανήκει στην εταιρία Microchip και αποτελεί μια πλήρως πιστοποιημένη ηλεκτρονική μονάδα για λειτουργία στα 433/868 MHz που βασίζεται στην ασύρματη τεχνολογία LoRa. Παρέχει πολλά πλεονεκτήματα όπως η πληθώρα λειτουργιών του αλλά κυρίως η χαμηλή κατανάλωση. Για το λόγο αυτό είναι η μονάδα που χρησιμοποιείται στην εργασία για την ασύρματη μεταφορά δεδομένων. Ακολουθεί πίνακας με τα τεχνικά χαρακτηριστικά του RN2483.

Table 4.1: LoRa module

Parameter Name	**Value**
Type	Sub-GHz
Output Power (dBm)	14.00
Host Interface	UART
Pin Count	47
Package	Surface mount module
RF Module	Yes
RF Transceiver	Yes
Operating Temperature Range	-40 to 85
Frequency Range	434, 868 MHz
Input Sensitivity (mVpp)	-148
Rx Input Sensitivity (dB)	-148
TX Current Consumption	40 mA (14dBm, 868MHz)
RX Current Consumption	14.2 mA

4.2.3 SIM 800 GSM Module

Το SIM800 είναι μια ολοκληρωμένη λύση Quad-band GSM / GPRS τύπου SMT που μπορεί να ενσωματωθεί στις εφαρμογές των πελατών τις εταιρίας. Υποστηρίζει Quad-band 850/900/1800 / 1900MHz, μπορεί να μεταδίδει πληροφορίες φωνής, SMS και δεδομένων με χαμηλή κατανάλωση ενέργειας (σχήμα 4.3).

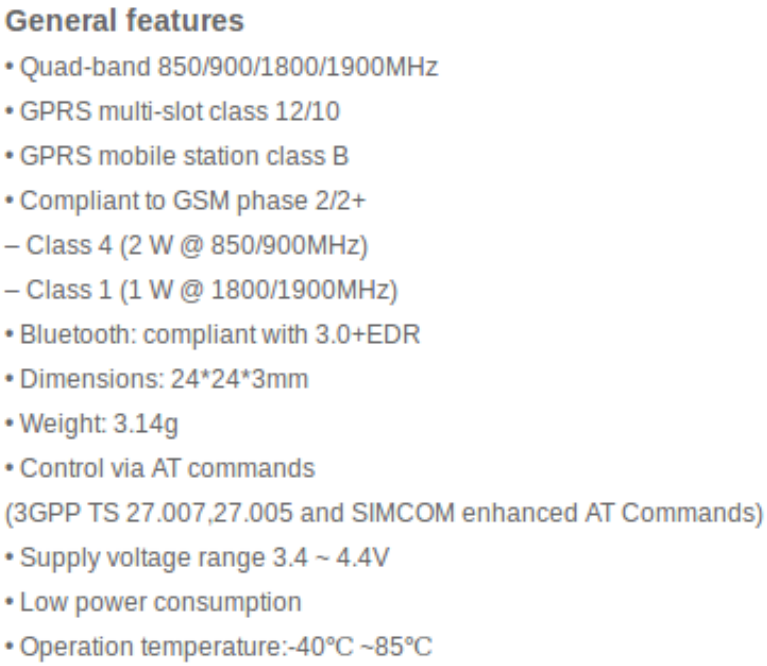
4.2.4 Το Real Time Clock (RTC)

Το RTC που χρησιμοποιήθηκε για τον χρονισμό του συστήματος είναι το DS1302 της εταιρίας Maxim Integrated.

4.2.5 Οι αισθητήρες

Οι αισθητήρες που χρησιμοποιήθηκαν είναι οι εξής :

1. Θερμοκρασία- Υγρασία : HDC1080 της εταιρίας Texas Instruments.
2. Φωτεινότητα : TEMT6000 Ambient Light Sensor
3. Αέρια : Το σύστημα είναι σχεδιασμένο κατά τέτοιο τρόπο ώστε να μπορεί να υποστηρίξει όλους τους αισθητήρες αερίων της σειράς MQ series Semiconductor Gas Sensor



Σχήμα 4.3: GPRS module

Table 4.2: Sensor's Characteristics

Sensor	**Range**	**Accuracy**
HDC1080 - Temperature	-40°C - 80°C	±0.2°C
HDC1080 - Humidity	0% - 100%	±2%
MQ2	10ppm - 500ppm	±5ppm
MQ7	300ppm - 10000ppm	±20ppm
MQ3	0.05mg/L - 10mg/L	±0.008mg/L
MQ4	200ppm - 10000ppm	±20ppm
TEMT6000	440-800nm (Spectral Bandwidth)	-

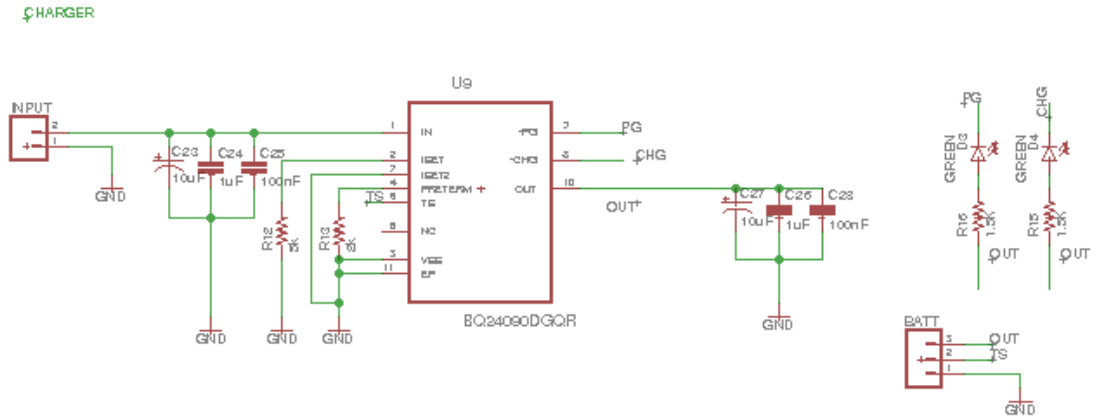
4.3 Κυκλωματική Υλοποίηση

Στη συγκεκριμένη ενότητα θα παρουσιαστεί η κυκλωματική υλοποίηση του συστήματος. Η παρουσίαση θα γίνει τμηματικά και χωρίς λεπτομέρειες για να διευκολύνει την κατανόηση. Όπως φαίνεται και στο Σχήμα 4.1 το σύστημα αποτελείται από δυο κύριες ενότητες:

1. Τους περιφερειακούς κόμβους
2. Το κεντρικό gateway

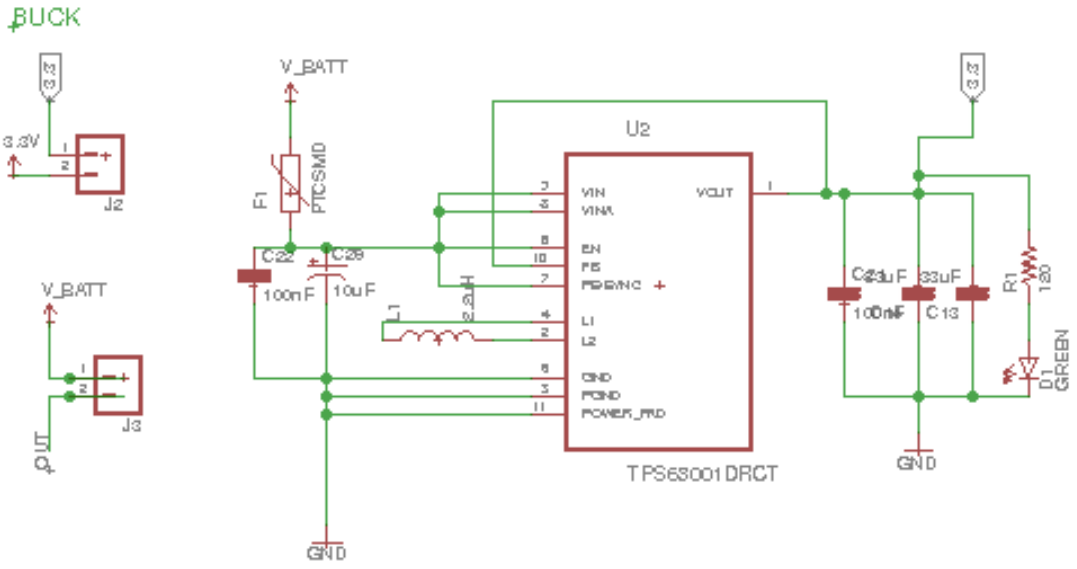
4.3.1 Η τροφοδοσία

Η τροφοδοσία και των δυο ενοτήτων είναι οι ίδια. Χρησιμοποιήθηκαν μπαταρίες λιθίου 3.7V, οι οποιές φορτίζουν μέσω της ηλιακής ενέργειας από φωτοβολταϊκά πάνελ 3.5W. Το κύκλωμα τροφοδοσίας αλλά και φόρτισης απεικονίζεται στο σχήμα 4.4.



Σχήμα 4.4: Σχηματικό τροφοδοσίας

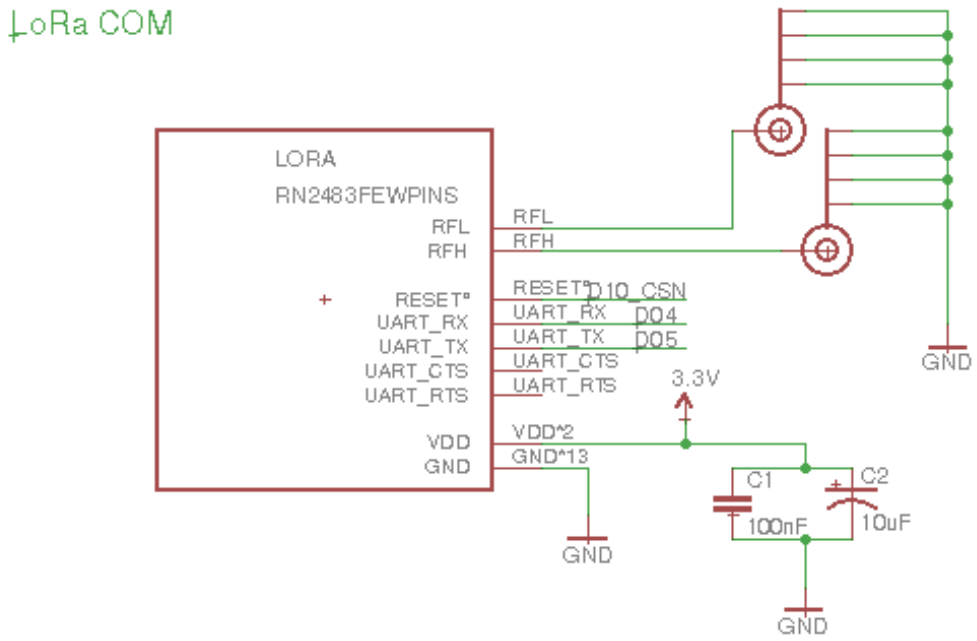
Η λειτουργία του ATmega328 και του RN2483 module απαιτεί σταθερή τάση 3.3V. Για το λόγο αυτό το και οι δυο ενότητες περιέχουν τον buck-boost converter TPS63001 που δέχεται ως είσοδο την τάση της μπαταρίας και έχει σταθερή έξοδο στα 3.3V.



Σχήμα 4.5: Σχηματικό buck-boost converter TPS63001

4.3.2 Διασύνδεση του RN2483 LoRa module

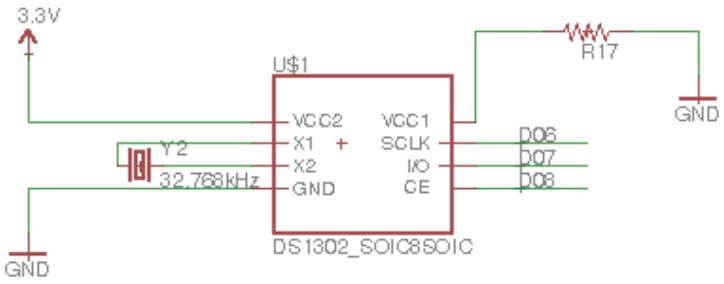
Στο παρακάτω σχηματικό παρουσιάζεται η διασύνδεση του RN2483 LoRa module με τον μικροελεγκτή και την τροφοδοσία για τους περιφερειακούς κόμβους και το gateway.



Σχήμα 4.6: Σχηματικό RN2483

4.3.3 Το Real Time Clock

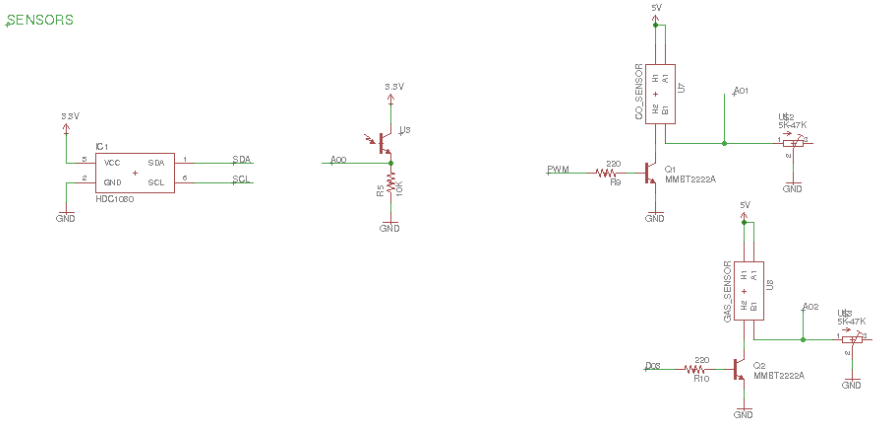
Η συνδεσμολογία του RTC φαίνεται στο σχήμα 4.7.



Σχήμα 4.7: Σχηματικό RTC DS1302

4.3.4 Οι αισθητήρες

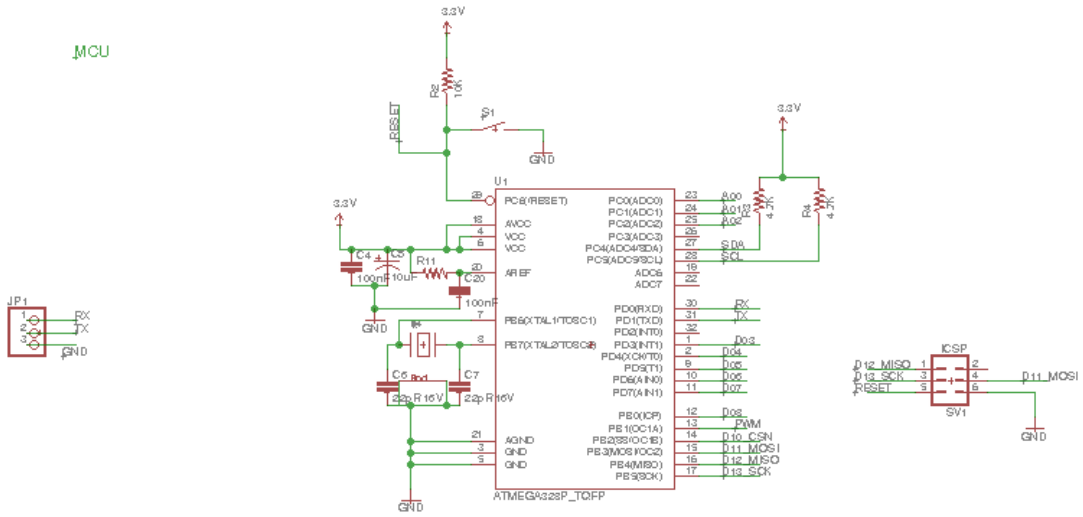
Στο σχήμα 4.8 φαίνεται η διασύνδεση των αισθητήρων του συστήματος στους περιφερειακούς κόμβους. Οι αισθητήρες των αερίων ενεργοποιούνται από τις επαφές του μικροελεγκτή μέσω BJT transistors.



Σχήμα 4.8: Σχηματικό αισθητήρων

4.3.5 Ο μικροελεγκτής ATmega328 των περιφερειακών κόμβων του συστήματος

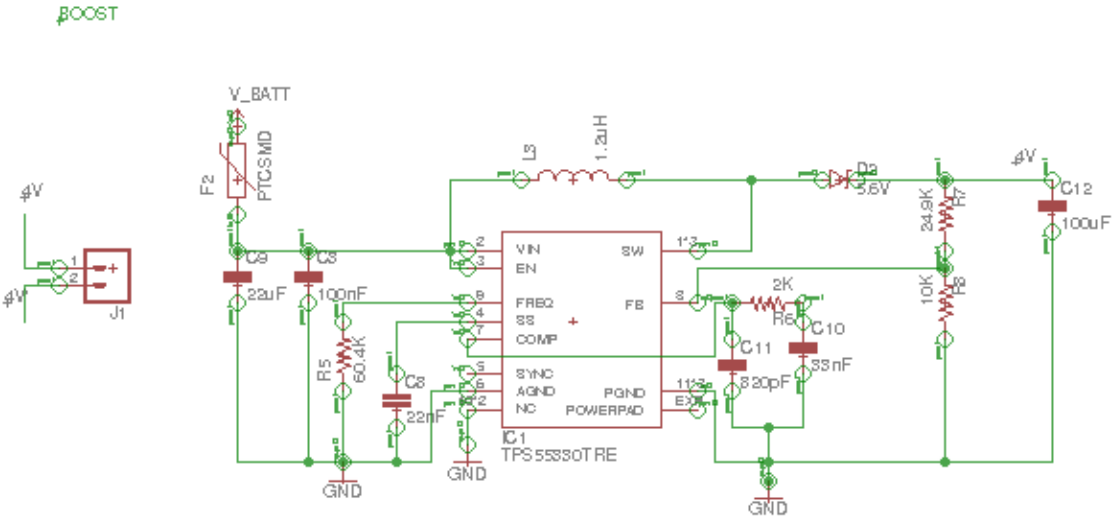
Όλες οι συνδέσεις του μικροελεγκτή των περιφερειακών κόμβων παρουσιάζονται στο σχήμα 4.9.



Σχήμα 4.9: Σχηματικό μικροελεγκτή περιφερειακών κόμβων

4.3.6 Buck-Boost converter στο gateway

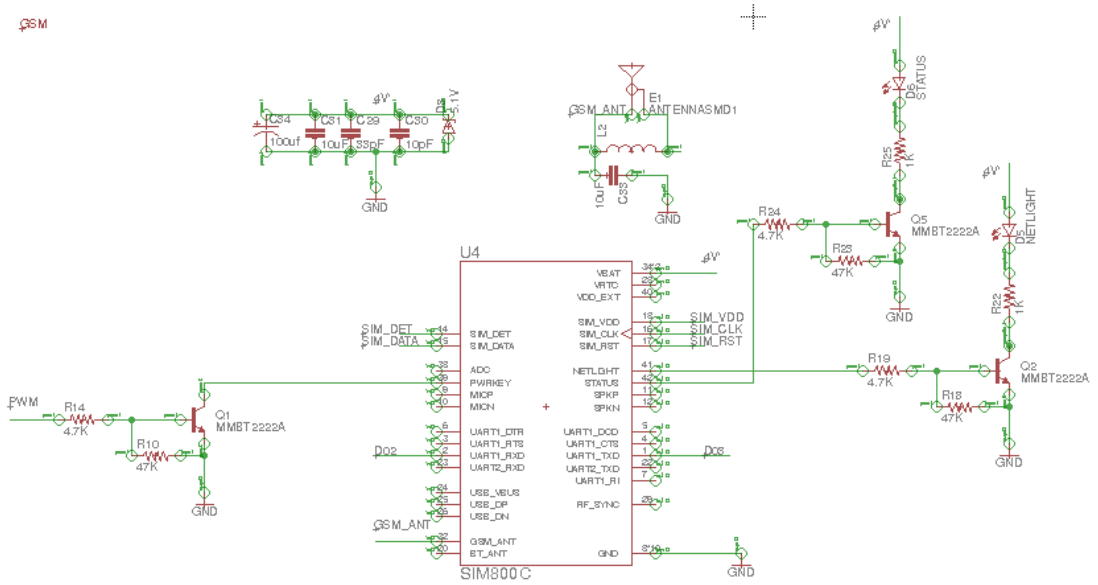
Η λειτουργία του SIM800 GSM module απαιτεί σταθερή τάση τροφοδοσίας στα 4V. Για το λόγο αυτό χρησιμοποιήθηκε ενας buck-boost converter TPS55330 με είσοδο την μπαταρία του gateway και σταθερή έξοδο στα 4 V για την τροφοδοσία του GSM module (σχήμα 4.10).



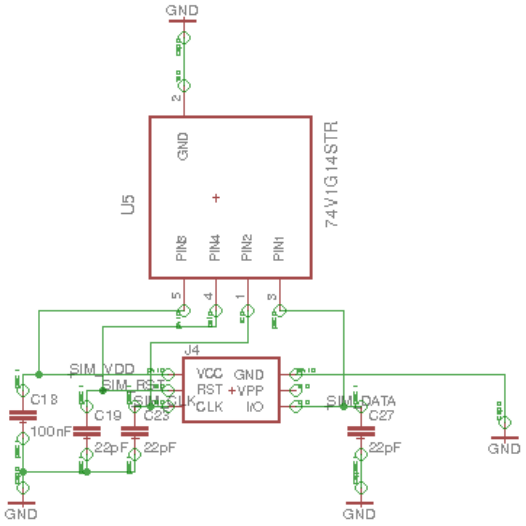
Σχήμα 4.10: Σχηματικό buck-boost converter TPS55330

4.3.7 GSM module

Στο σχήμα 4.11 φαίνεται η συνδεσμολογία του GSM module. Το κύκλωμα για την λειτουργία της κάρτας SIM φαίνεται στο σχήμα 4.12.



Σχήμα 4.11: Σχηματικό SIM800 GSM module

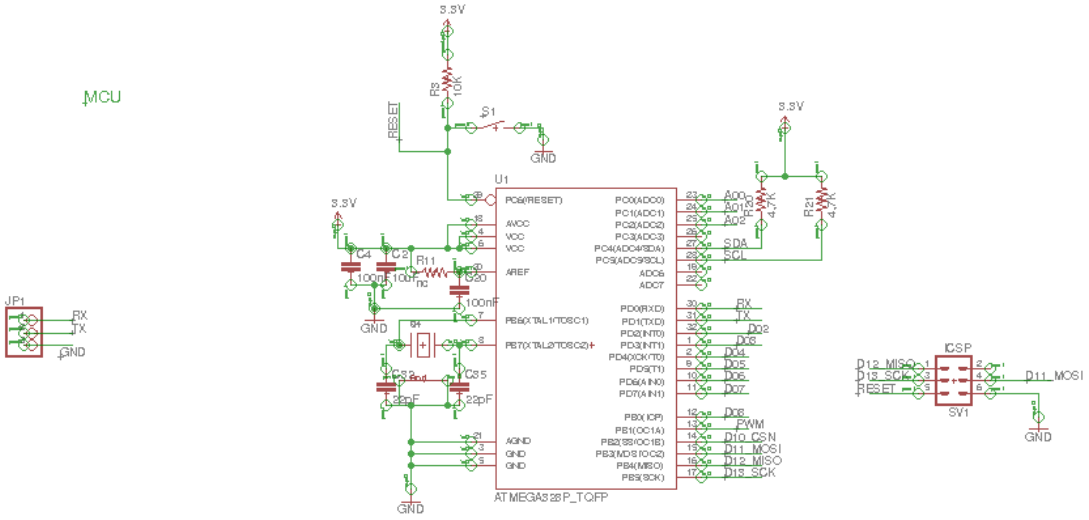


Σχήμα 4.12: Σχηματικό κυκλώματος λειτουργίας κάρτας SIM

Όπως φαίνεται στο σχήμα 4.12, για την ορθή λειτουργία του GSM module απαιτείται κύκλωμα που θα παρέχει ESD protection. Για τον συγκεκριμένο λόγο χρησιμοποιήθηκε το ESDA6V1SC5 της εταιρίας STMicroelectronics.

4.3.8 Ο μικροελεγκτής ATmega328 του gateway του συστήματος

Όλες οι συνδέσεις του μικροελεγκτή του gateway παρουσιάζονται στο σχήμα 4.13.



Σχήμα 4.13: Σχηματικό μικροελεγκτή gateway

4.4 Προγραμματισμός μικροεπεξεργαστών

Ένα από τα πιο σημαντικά κομμάτια της συγκεκριμένης εργασίας είναι ο προγραμματισμός του μικροελεγκτή ATmega328 των περιφερειακών κόμβων και του gateway. Κύριος στόχος είναι η αποδοτική λειτουργία του συστήματος με την μικρότερη δυνατή κατανάλωση ενέργειας.

4.4.1 Προγραμματισμός περιφερειακών κόμβων

Η λειτουργία των περιφερειακών κόμβων βασίζεται στην αποστολή των δεδομένων που λαμβάνουν από τους αισθητήρες. Οι απαιτήσεις σε ρεύμα για την λειτουργία των αισθητήρων είναι αρκετά μεγάλες και για το λόγο αυτό δεν είναι δυνατό να λειτουργούν μόνιμα.(πίνακας καταναλώσεων) Με χρήση του RTC για σωστό χρονισμό αλλά και του αισθητήρα φωτεινότητας πραγματοποιούνται οι παρακάτω διεργασίες με την ακόλουθη χρονική σειρά :

Θέρμανση αισθητήρων αερίου

- Οι αισθητήρες των αερίων θερμαίνονται για χρονική διάρκεια ενός λεπτού ανά ώρα.
- Αν η τιμή της τάσης της μπαταρίας πέσει κάτω από ένα επιτρεπτό όριο τότε η συγκεκριμένη διαδικασία παρακάμπτεται.

Λήψη δεδομένων απο τους αισθητήρες

- Η έναρξη της λήψης των δεδομένων πραγματοποιείται αμέσως μετά την ολοκλήρωση της παραπάνω διεργασίας.
- Αν η τιμή της τάσης της μπαταρίας πέσει κάτω από ένα επιτρεπτό όριο τότε η λήψη γίνεται από όλους τους αισθητήρες εκτός των αερίων.

Αποστολή δεδομένων.

Η ασύρματη αποστολή δεδομένων ξεκινάει σε προκαθορισμένη χρονική στιγμή και μαζί με την λήψη των δεδομένων έχει διάρκεια ενός λεπτού.

Sleep mode.

Στην υπόλοιπη διάρκεια δηλαδή 58 λεπτά ανά ώρα ο κάθε περιφερειακός κόμβος πηγαίνει σε κατάσταση sleep mode απενεργοποιώντας τους αισθητήρες και το RN2483 LoRa module με στόχο την χαμηλή κατανάλωση.

4.4.2 Προγραμματισμός gateway

Η λειτουργία του gateway είναι ο συγχρονισμός των περιφερειακών κόμβων, η λήψη όλων των δεδομένων και κατόπιν η αποστολή τους στο διαδίκτυο . Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο το gateway αποτελείται απο διαφορετικές βαθμίδες, επομένως εκτός από

την ορθή λειτουργία του συστήματος ο προγραμματισμός είναι αναγκαίος και για την διατήρηση της χαμηλής κατανάλωσης. Με χρήση και πάλι του Real Time Clock πραγματοποιούνται οι εξής διεργασίες:

Λήψη δεδομένων από τους περιφερειακούς κόμβους.

Η λειτουργία αυτή βασίζεται στην ασύρματη μετάδοση της πληροφορίας μέσω LoRa, επομένως απαιτείται η ενεργοποίηση του RN2483 LoRa module . Η χρονική διάρκεια είναι συνολικά 4 λεπτά ανά ώρα δηλαδή 2 λεπτά ανά κόμβο.

Επεξεργασία των δεδομένων από τον μικροελεγκτή.

Τα δεδομένα που λαμβάνει το gateway δεν είναι στην κατάλληλη μορφή για να μπορούν να τα αντιληφθούν οι χρήστες. Ο ATmega328 αναλαμβάνει αυτή την διεργασία όπου χωρίζει και ρυθμίζει με ορθό τρόπο την πληροφορία για ανέβει στο διαδίκτυο.

Αποστολή δεδομένων στο διαδίκτυο.

Η συγκεκριμένη διεργασία απαιτεί την ενεργοποίηση του SIM800 GSM module. Έχει χρονική διάρκεια 2 λεπτών ανά ώρα δηλαδή 1 λεπτού ανά κόμβο.

Sleep mode.

Στην υπόλοιπη διάρκεια δηλαδή 56 λεπτά ανά ώρα το γατεωαψ πηγαίνει σε κατάσταση sleep mode απενεργοποιώντας το SIM800 GSM module και το RN2483 LoRa module με στόχο την χαμηλή κατανάλωση.

Ο ολοκληρωμένος κώδικας του συστήματος βρίσκεται στο κεφάλαιο κώδικας.

Κεφάλαιο 5

Κατανάλωση ισχύος

Αρχικά, πραγματοποιήθηκε ανάλυση για την κατανάλωση ισχύος κάθε διεργασίας για τους περιφερειακούς κόμβους του συστήματος.

Table 5.1: Peripheral Node Power Consumption Distribution

Task	**Power Consumption (mW)**
Standby (Only MCU and RTC working)	8
LoRa Tranceiver	116
Temperature - Humidity Measurement	0.16
Smoke Sensor Measurement	400
CO Sensor Measurement	400
Methane Sensor Measurement	400
Alcohol Sensor Measurement	400
Ambient Light Sensor Measurement	1.6

Οι μετρήσεις των αισθητήρων καθώς και η ασύρματη μετάδοση τους πραγματοποιούνται σε προκαθορισμένα χρονικά διαστήματα. Η εξίσωση που περιγράφει την κατανάλωση ισχύος σε κάθε κόμβο :

$$P_{Total} = P_{Standby} + P_{Tranceiver} \times L + P_{Temp-Humid} \times O + P_{Gas} \times 2 \times R + P_{Light} \times A \quad (5.1)$$

Όπου οι σταθερές L,O,R,A αποτελούν τα duty cycles για 24 ώρες λειτουργίας. Απο συνδυασμό της κατανάλωσης και της διαθέσιμης ηλιακής ενέργειας σύμφωνα με το επόμενο κεφάλαιο, επιλέξαμε L=0.016, O=0.016, R=0.33, A=0.016. Η συνολική κατανάλωση ισχύος υπολογίζεται στις 870.81 mWh.

Στη συνέχεια, πραγματοποιήθηκε ανάλυση για την κατανάλωση ισχύος κάθε διεργασίας για το gateway του συστήματος.

Table 5.2: Gateway Power Consumption Distribution

Task	**Power Consumption (mW)**
Standby (MCU, RTC)	8
LoRa Tranceiver	116
GPRS Connection / Data Upload	1200

Δεδομένου ότι ο χρόνος για την μετάδοση των δεδομένων στο διαδίκτυο είναι προκαθορισμένος, είναι εύκολο να υπολογίσουμε την συνολική κατανάλωση βάση της επόμενης εξίσωσης.

$$P_{Total} = P_{Standby} + P_{Tranceiver} \times T \times N + P_{GPRS} \times G \times N \tag{5.2}$$

Στην περίπτωση αυτή παρατηρούμε πως η κατανάλωση ισχύος και επομένως και το duty cycle των διεργασιών εξαρτάται από τον αριθμό των περιφερειακών κόμβων. Για την συγκεκριμένη εργασία επιλέξαμε N= 2 , T = 0.008 και G = 0.008. Συνεπώς η κατανάλωση ισχύος του gateway σε δίαρκεια 24 ωρών υπολογίζεται στις 697.3 mWh.

Κεφάλαιο 6

Ανανεώσιμες Πηγές Ενέργειας

Όπως είναι ευρέως γνωστό, τα συστήματα που σχετίζονται με την μετάδοση πληροφορίας σχετικά με το περιβάλλον πρέπει να έχουν μεγάλη διάρκεια ζωής. Σε κάθε άλλη περίπτωση η πληροφορία θα ήταν ελλειπής και δεν θα μπορούσε να χρησιμοποιηθεί με αποτελεσματικό τρόπο από τους αρμόδιους. Στο συγκεκριμένο σύστημα χρησιμοποιούνται μπαταρίες λιθίου χωρητικότητας 26000 mAh για την τροφοδοσία του κάθε κόμβου αλλά και του gateway. Σύμφωνα με τους υπολογισμούς μας, με την αποκλειστική χρήση των συγκεκριμένων μπαταριών η διάρκεια ζωής του συστήματος θα ήταν μόνο 6 ημέρες. Η απάντηση στο πρόβλημα αυτό είναι η χρήση ανανεώσιμων πηγών ενέργειας για την φόρτιση των μπαταριών.

Με τον όρο ανανεώσιμες πηγές ενέργειας εννοούμε γενικά τις εναλλακτικές των παραδοσιακών πηγών ενέργειας (π.χ. του πετρελαίου ή του άνθρακα), όπως η ηλιακή και η αιολική. Παρουσιάζουν πολλά πλεονεκτήματα :

Είναι πολύ φιλικές προς το περιβάλλον, έχοντας ουσιαστικά μηδενικά κατάλοιπα και απόβλητα.

1. Δεν πρόκειται να εξαντληθούν ποτέ, σε αντίθεση με τα ορυκτά καύσιμα.

2. Μπορούν να βοηθήσουν την ενεργειακή αυτάρκεια μικρών και αναπτυσσόμενων χωρών, καθώς και να αποτελέσουν την εναλλακτική πρόταση σε σχέση με την οικονομία του πετρελαίου.

3. Είναι ευέλικτες εφαρμογές, που μπορούν να παράγουν ενέργεια ανάλογη με τις ανάγκες του επί τόπου πληθυσμού, καταργώντας την ανάγκη για τεράστιες μονάδες παραγωγής ενέργειας (καταρχήν για την ύπαιθρο) αλλά και για μεταφορά της ενέργειας σε μεγάλες αποστάσεις.

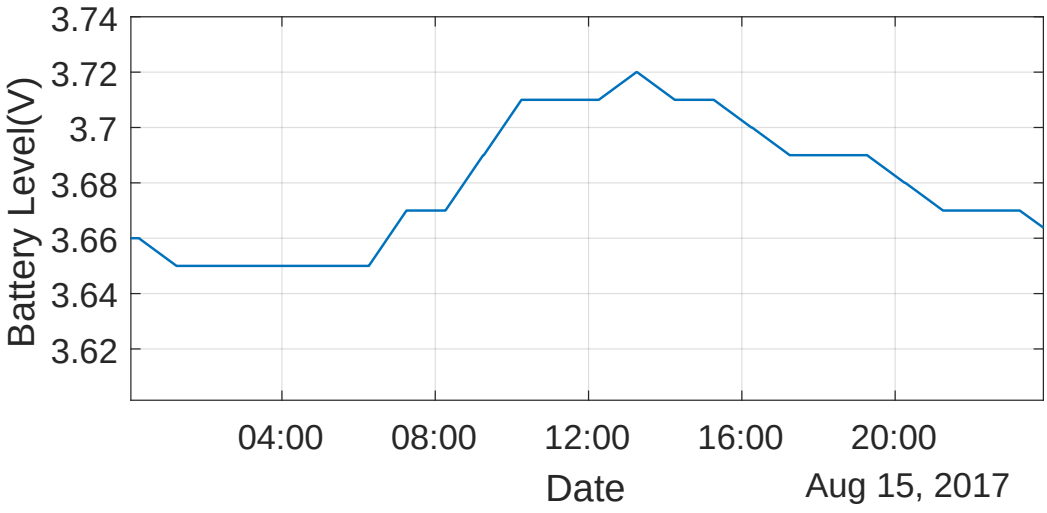
4. Ο εξοπλισμός είναι απλός στην κατασκευή και τη συντήρηση και έχει πολύ μεγάλο χρόνο ζωής.

5. Επιδοτούνται από τις περισσότερες κυβερνήσεις.

Στην συγκεκριμένη εργασία χρησιμοποιούμε ηλιακή ενέργεια. Χρησιμοποιείται περισσότερο για θερμικές εφαρμογές (ηλιακοί θερμοσίφωνες και φούρνοι) ενώ η χρήση της για την

παραγωγή ηλεκτρισμού έχει αρχίσει να κερδίζει έδαφος, με την βοήθεια της πολιτικής προώθησης των Ανανεώσιμων Πηγών Ενέργειας από το ελληνικό κράτος και την Ευρωπαϊκή Ένωση.

Η φόρτιση των μπαταριών του συστήματος πραγματοποιείται μέσω της χρήσης φωτοβολταϊκών πάνελ των 3.5 W. Για την Αθήνα η εκτιμώμενη παραγωγή ενέργειας στη διάρκεια 24 ωρών είναι 21 Wh. Ο τρόπος και η διάταξη του κυκλώματος φόρτισης παρουσιάζονται αναλυτικά στο προηγούμενο κεφάλαιο. Τα αποτελέσματα που επιβεβαιώνουν την αποδοτικότητα του συνδυασμού αυτού βρίσκονται στο παρακάτω σχήμα.



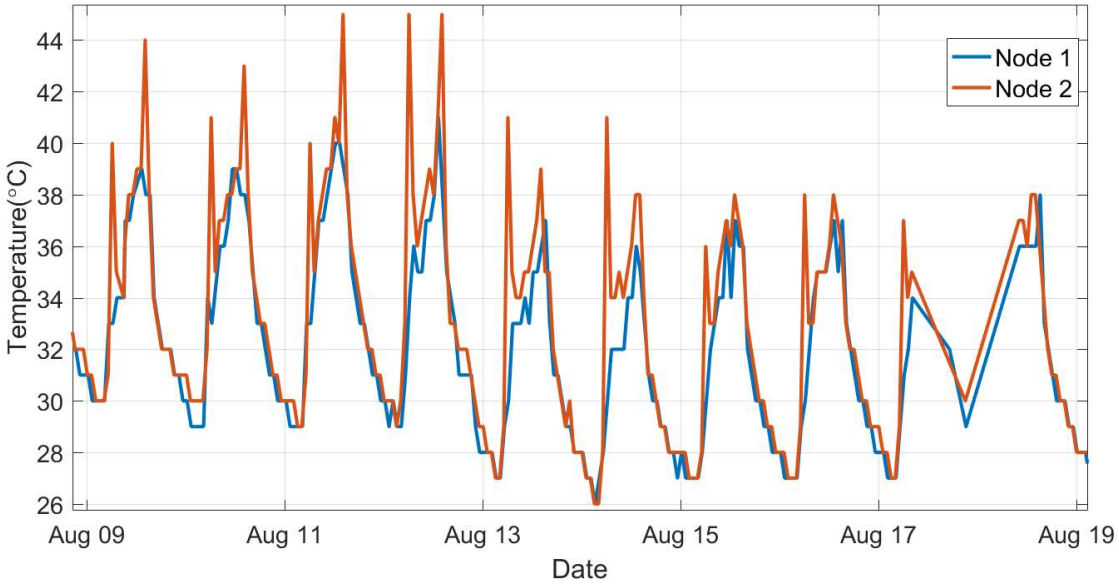
Σχήμα 6.1: Battery Level

Συγκεκριμένα κατά την διάρκεια μιας ηλιόλουστης ημέρας η φόρτιση παρέχει αρκετή ενέργεια στην μπαταρία για να καλύψει την ανάγκη τροφοδοσίας του συστήματος αλλά και επιπλέον απόθεμα ενέργειας σε περίπτωση που ακολουθήσουν μέρες με λιγότερη ηλιοφάνεια.

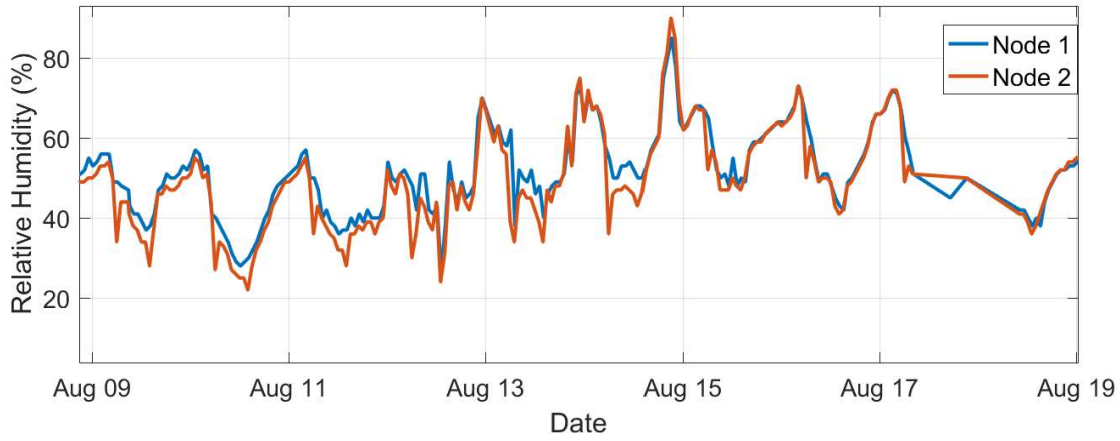
Κεφάλαιο 7

Αποτελέσματα

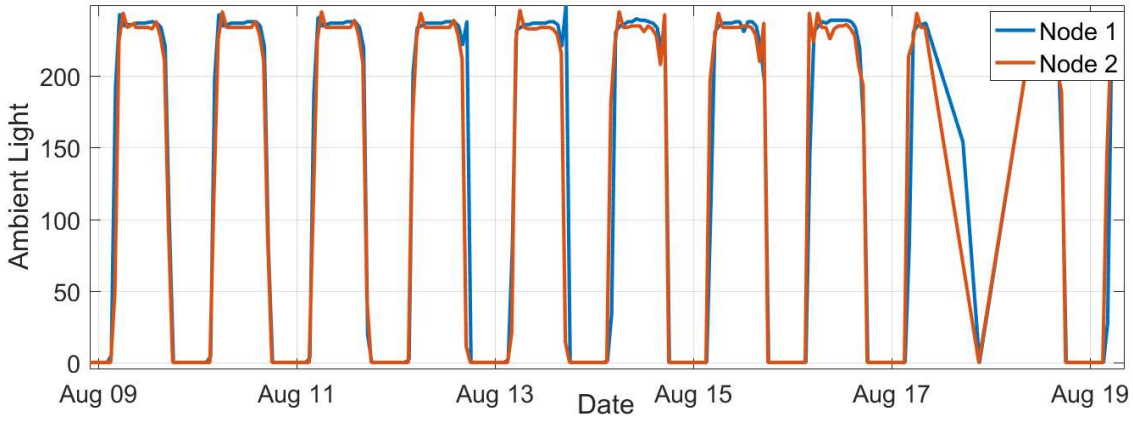
Μετά από 10 ημέρες συνεχόμενης λειτουργίας, οι ακόλουθες μετρήσεις καταγράφηκαν:



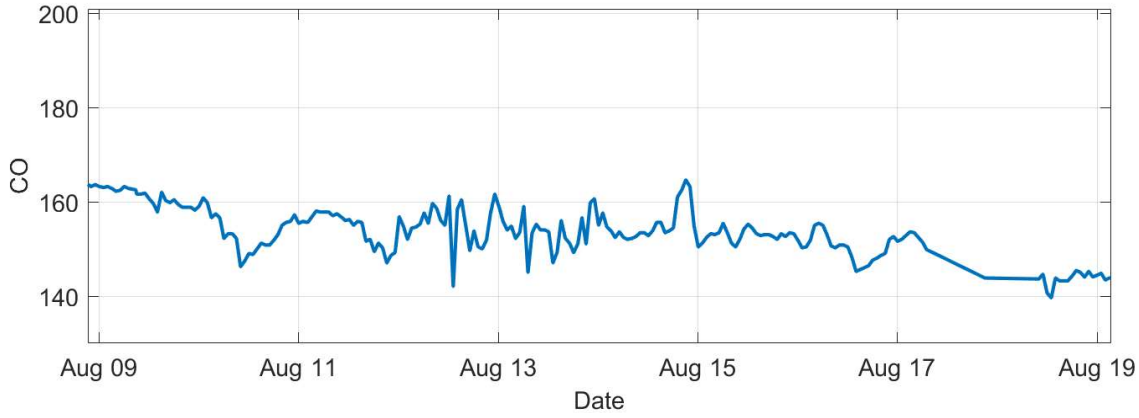
Σχήμα 7.1: Μετρήσεις θερμοκρασίας



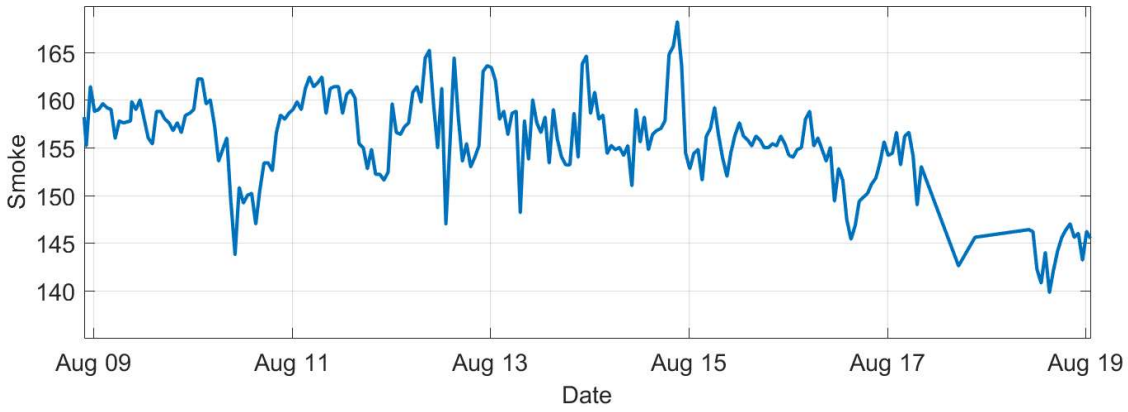
Σχήμα 7.2: Μετρήσεις Υγρασίας



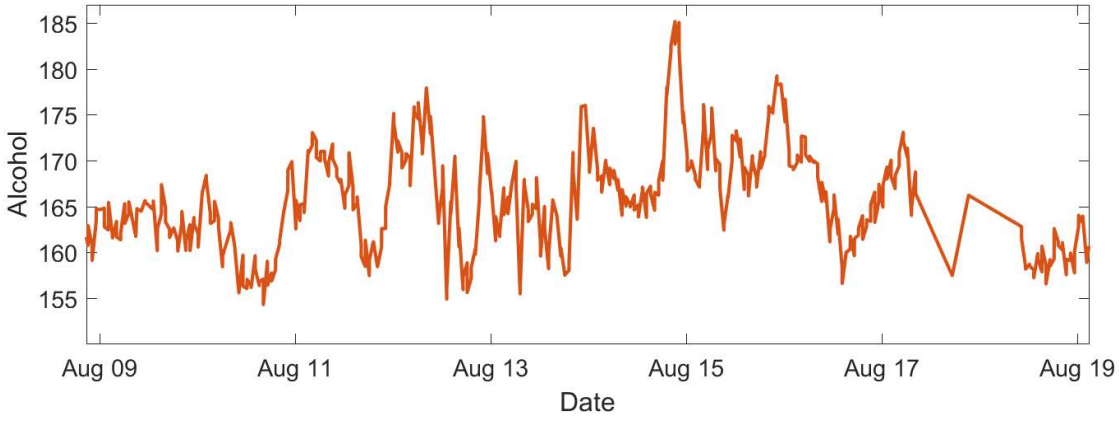
Σχήμα 7.3: Μετρήσεις Φωτεινότητας



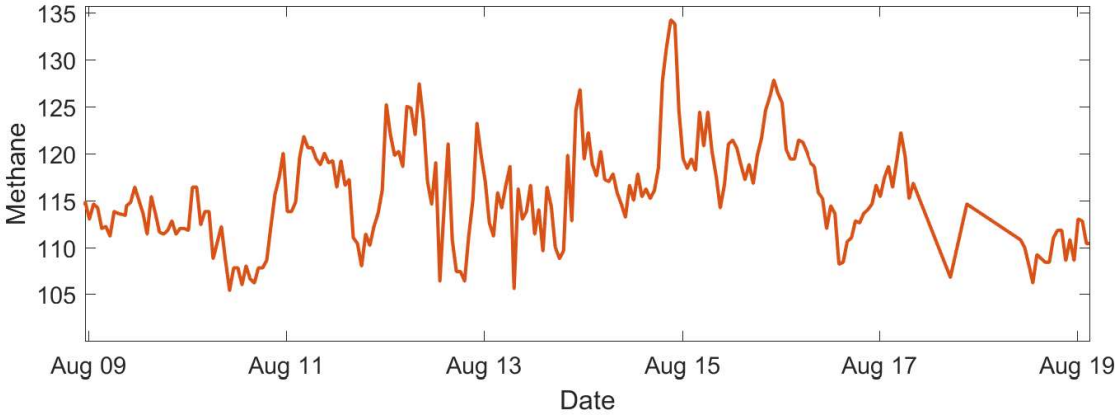
Σχήμα 7.4: Μετρήσεις CO κόμβου 1



Σχήμα 7.5: Μετρήσεις καπνού κόμβου 1



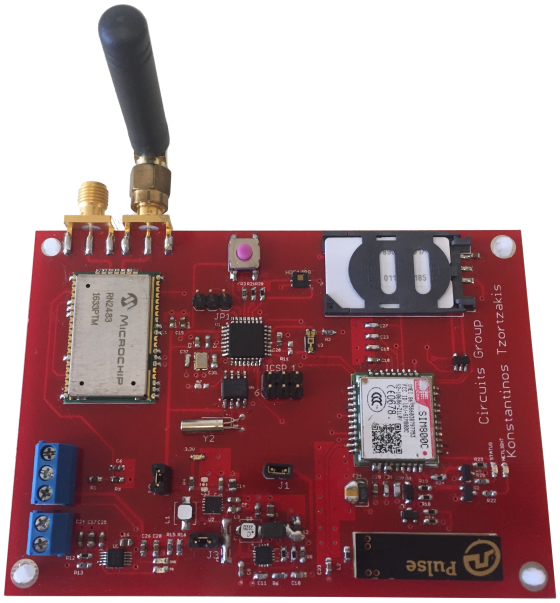
Σχήμα 7.6: Μετρήσεις αλκοόλ κόμβου 2



Σχήμα 7.7: Μετρήσεις CH4 κόμβου 2

Όσον αφορά τις μετρήσεις αερίων, η πληροφορία από τους αισθητήρες επεξεργάζονται και μεταφράζονται σε αναλογικές αναγνώσεις. Θα πρέπει να αναφέρουμε εδώ ότι η μετατροπή καθώς και οι ακριβείς υπολογισμοί από την αναλογική μορφή σε μονάδες στο S.I., είναι πέραν από το πεδίο αυτής της εργασίας και έτσι δεν είναι παρουσιάστηκε.

Οι περιφερειακοί κόμβοι και το gateway που περιγράψαμε υλοποιήθηκαν σε PCB, όπως φαίνονται στα παρακάτω σχήματα.



Σχήμα 7.8: Gateway

Σχήμα 7.9: Περιφερειακός κόμβος

Συμπερασματικά, τα αποτελέσματα αποδεικνύουν ότι το σύστημα είναι εντελώς αξιόπιστο και αυτό δικαιολογείται από την έλλειψη σημείων πτώσης στο μηδέν (drop points to zero) σε όλο το παράθυρο μέτρησης. Αυτό εξασφαλίζει επίσης ότι υπάρχει δεν υπάρχει απώλεια πακέτων μεταξύ της συνδεσιμότητας του συστήματος και ότι η παρεχόμενη ενέργεια είναι

επαρκής. Επομένως, αυτό το σύστημα που βασίζεται στο πρωτόκολλο LoRa μπορεί να βρει θέση ως μέρος μιας αρχιτεκτονικής Smart Cities .

Κεφάλαιο 8

Κώδικας

Στο συγκεκριμένο κεφάλαιο παρουσιάζεται αναλυτικά ο κώδικας που χρησιμοποιήθηκε για τον προγραμματισμό των μικροελεγκτών του συστήματος. Σε αρκετά σημεία υπάρχουν σχόλια που θα βοηθήσουν στην κατανόηση του.

Αρχικά παρουσιάζεται ο κώδικας του πρώτου περιφερειακού κόμβου. Ο κώδικας του δεύτερου περιφερειακού κόμβου δεν θα παρουσιαστεί καθώς δεν διαφέρει σημαντικά με εκείνο του πρώτου, παραμόνο στα χρονικά διαστήματα λειτουργίας.

```
#include <SoftwareSerial.h>
#include <LowPower.h>
#include <Wire.h>
#include "ClosedCube_HDC1080.h"
#include <virtuabotixRTC.h>
byte t6=0;
int smokeA0 = A2;
int gasCO = A1;
int detSmoke;
#define heater1       3
#define heater2        9
#define batt       A3

ClosedCube_HDC1080 hdc1080;
virtuabotixRTC myRTC(6, 7, 8);

#define DBG_LVL         2
#define LED_PIN         10
#define CMD_TIMEOUT     50
#define RN_RST         10 // RF module reset pin

#define SS_RX_PIN       5  // SoftSerial Rx pin
#define SS_TX_PIN       4  // SoftSerial Tx pin
```

```
24
25 //define LoRa parameters
26 #define RF_FREQ          867500000
27 #define RF_POW           14
28 #define RF_SF            "sf12"
29 #define RF_BW            125
30 #define RF_RATE          8
31
32 #define SERIAL_BAUD      57600
33 #define SLEEP_TIME      1 //33 Results in approximately 5 min
34 int sens;
35 int temt6000Pin = 0;
36 // Initialize software serial port for communications to RF module
37 SoftwareSerial mySerial(SS_RX_PIN, SS_TX_PIN);
38
39 // Function prototype for default value
40 boolean rn_cmd(String cmd, String resp, int timeout = CMD_TIMEOUT);
41
42 void setup()
43 {
44   // HW serial for debugging
45   if (DBG_LVL > 0) {
46     Serial.begin(SERIAL_BAUD);
47     Serial.println("Initializing...");
48 }
49 pinMode(smokeA0, INPUT);
50 pinMode(gasCO, INPUT);
51 pinMode(heater1, OUTPUT);
52 pinMode(smokeA0, INPUT);
53 pinMode(gasCO, INPUT);
54 pinMode(heater1, OUTPUT);
55 pinMode(heater2, OUTPUT);
56 // Set the current date, and time in the following format:
57 // seconds, minutes, hours, day of the week, day of the month, month, year
58 myRTC.setDS1302Time(00, 0, 15, 29, 29, 6, 2017);
59   // SW serial for LoRa communication
60   mySerial.begin(SERIAL_BAUD);
61
62   // Initialize pin modes
63   pinMode(LED_PIN, OUTPUT);
64   pinMode(RN_RST, OUTPUT);
```

```
65
66    // Blink LED once
67    Blink(LED_PIN, 1);
68
69    // Reset RF module
70    reset_rf();
71
72    // Set RF parameters
73    set_rf_params();
74    hdc1080.begin(0x40);
75
76    Serial.print("Manufacturer ID=0x");
77    Serial.println(hdc1080.readManufacturerId(), HEX); // 0x5449 ID of Texas Instruments
78    Serial.print("Device ID=0x");
79    Serial.println(hdc1080.readDeviceId(), HEX); // 0x1050 ID of the device
80
81  }
82
83  byte identity = 101;
84  byte light_sensor = 0;
85  byte hum_sensor = 0;
86  byte temp_sensor = 0;
87  String tmp_str;
88  void loop()
89  {
90
91    myRTC.updateTime();
92
93    digitalWrite(heater1, LOW);
94    digitalWrite(heater2, LOW);
95
96    int analogSensor = analogRead(smokeA0);
97    int analogSensor1 = analogRead(gasCO);
98    int battery = analogRead(batt);
99
100   Serial.print(battery);
101
102 t6= myRTC.minutes ;
103
104 hum_sensor= hdc1080.readHumidity();
105 light_sensor= analogRead(temt6000Pin);
```

```
106 emp_sensor=hdc1080.readTemperature();
107
108
109
110   //Smoke detection
111    if (analogSensor<=500)
112   detSmoke=0;
113
114    if (analogSensor>500)
115    detSmoke=1;
116
117   if (DBG_LVL > 1) get_rf_params();
118
119   if (DBG_LVL > 0) Serial.println("Sending data");
120
121  Serial.print("Current Date / Time: ");
122 Serial.print(myRTC.dayofmonth);
123 Serial.print("/");
124 Serial.print(myRTC.month);
125 Serial.print("/");
126 Serial.print(myRTC.year);
127 Serial.print(" ");
128 Serial.print(myRTC.hours);
129 Serial.print(":");
130 Serial.print(myRTC.minutes);
131 Serial.print(":");
132 Serial.println(myRTC.seconds);
133
134
135   unsigned long tmp_millis = millis();
136
137 //Start heating gas sensors
138 if(t6==2 && battery >=500)
139 {
140   digitalWrite(heater1, HIGH);
141   digitalWrite(heater2, HIGH);
142
143 }
144
145 //Transmit data
146 if(t6==3)
```

```
147 {
148   digitalWrite(heater1, HIGH);
149   digitalWrite(heater2, HIGH);
150 
151     if (DBG_LVL > 1) get_rf_params();
152 
153   if (DBG_LVL > 0) Serial.println("Sending data");
154   // Transmit data
155   String tmp_cmd = ("radio tx " + String (identity) + String(hum_sensor)
156   +String(temp_sensor)   + String(detSmoke) + String(analogSensor1)
157   + String(battery) + String(light_sensor) );
158 
159   boolean err = rn_cmd(tmp_cmd, "radio_tx_ok", 2000);}
160 
161 
162   if (DBG_LVL > 0) {
163     Serial.println("Going to sleep");
164     Serial.flush();
165 
166     delay(300);
167   }
168 
169 
170 
171 }
172 
173 
174 // Reset RF
175 void reset_rf() {
176 
177   digitalWrite(RN_RST, LOW);
178   LowPower.powerDown(SLEEP_250MS, ADC_OFF, BOD_OFF);
179   digitalWrite(RN_RST, HIGH);
180 
181   rn_cmd("", "", 200);
182 }
183 
184 // Configure RF parameters
185 boolean set_rf_params() {
186 
187   boolean err = false;
```

```
188
189   err |= rn_cmd("mac pause", "4294967245"); // Stop LoRaWAN MAC (allowing PHY-level Tx/Rx)
190   err |= rn_cmd("radio set freq " + String(RF_FREQ), "ok");
191   err |= rn_cmd("radio set pwr "  + String(RF_POW) , "ok");
192   err |= rn_cmd("radio set sf "   + String(RF_SF)  , "ok");
193   err |= rn_cmd("radio set bw "   + String(RF_BW)  , "ok");
194   err |= rn_cmd("radio set cr 4/" + String(RF_RATE), "ok");
195
196   return err;
197 }
198
199 // Get RF parameters
200 boolean get_rf_params() {
201
202   boolean err = false;
203   err |= rn_cmd("radio get freq",   String(RF_FREQ));
204   err |= rn_cmd("radio get pwr",  String(RF_POW));
205   rn_cmd("radio get sf", "");
206   err |= rn_cmd("radio get bw",   String(RF_BW));
207   err |= rn_cmd("radio get cr", "4/" + String(RF_RATE));
208
209   return err;
210 }
211
212 // Serial command to RF
213 boolean rn_cmd(String cmd, String resp, int timeout)
214 {
215   // If not empty, send command to Serial
216   if (cmd != "") mySerial.println(cmd);
217   boolean err = true;
218
219   unsigned long tmp_millis = millis();
220
221   // Check for timeout
222   String tmp_str = "";
223   while (millis() - tmp_millis < timeout) {
224     if (mySerial.available()) {
225       tmp_str = mySerial.readStringUntil('\n');
226       if (tmp_str.indexOf(resp) == 0) {
227         err = false;
228         break;
```

```
229         }
230       }
231     }
232
233     if (DBG_LVL > 0) {
234       if (resp != "") {
235         if (err) {
236           Serial.print("ERROR! ");
237         } else {
238           Serial.print("OK. ");
239         }
240       }
241       Serial.println("CMD: " + cmd + " RESP: " + tmp_str);
242       Serial.flush();
243     }
244     return err;
245   }
246
247   // Blink LED
248   void Blink(byte b_pin, int b_times)
249   {
250     if (DBG_LVL > 0) Serial.flush();
251
252     for (int i = 0; (i < b_times); i++) {
253       digitalWrite(b_pin, HIGH);
254       LowPower.powerDown(SLEEP_120MS, ADC_OFF, BOD_OFF);
255       digitalWrite(b_pin, LOW);
256       LowPower.powerDown(SLEEP_120MS, ADC_OFF, BOD_OFF);
257     }
258   }
```

Τέλος, παρουσιάζεται αναλυτικά ο κώδικας του κεντρικού κόμβου (gateway).

```
#include <SoftwareSerial.h>
#include <LowPower.h>
#include <virtuabotixRTC.h>

#define DBG_LVL         2
#define LED_PIN         10
#define CMD_TIMEOUT     50
#define RN_RST          10  // RF module reset pin

#define SS_RX_PIN       5  // SoftSerial Rx pin
#define SS_TX_PIN       4  // SoftSerial Rx pin

//define LoRa parameters
#define RF_FREQ         867500000
#define RF_POW          14
#define RF_SF           "sf12" //sf7,sf8,...,sf12
#define RF_BW           125
#define RF_RATE         8

#define SERIAL_BAUD     57600
#define SLEEP_TIME     1 //33 Results in approximately 5 min
int temt6000Pin = 0;
#define batter    A2

virtuabotixRTC myRTC(6, 7, 8);


// Initialize software serial port for communications to RF module
SoftwareSerial mySerial(SS_RX_PIN, SS_TX_PIN);
SoftwareSerial gsmSerial(3,2);

// Function prototype for default value
boolean rn_cmd(String cmd, String resp, int timeout = CMD_TIMEOUT);
int k = 1; unsigned long tmp_millis = millis();
void setup()
{
  // HW serial for debugging
  if (DBG_LVL > 0) {
    Serial.begin(SERIAL_BAUD);
```

```
40      Serial.println("Initializing...");
41        pinMode(9, OUTPUT);
42    digitalWrite(9,LOW);
43    delay(1000);
44    digitalWrite(9,HIGH);
45    delay(1000);
46    //keep GPRS module disabled
47    }
48  myRTC.setDS1302Time(00, 0, 15, 29, 29, 6, 2017);
49    // SW serial for RF module communications
50
51
52    mySerial.begin(SERIAL_BAUD);
53    // Initialize pin modes
54    pinMode(LED_PIN, OUTPUT);
55    pinMode(RN_RST, OUTPUT);
56
57    // Blink LED once
58    Blink(LED_PIN, 1);
59
60    // Reset RF module
61    reset_rf();
62
63    // Set RF parameters
64    set_rf_params();
65  }
66  String tmp_cmd = "";
67  String r1, h1, h2, h4;
68  String h3 = "radio_r";
69
70  String tmp_str = "";
71  String t0;
72  String t1;
73  String t2;
74  String t3;
75  String t4;
76  String t41;
77  String t412;
78  String batt;
79  String batt1;
80  String Co;
```

```
81 String Co1;
82 float batt12;
83 float Co12;
84 String light;
85 int light1;
86 int light12;
87 String light13;
88 int t6;
89 int oust=1;
90 ;
91
92 void loop()
93 {
94   myRTC.updateTime();
95
96    digitalWrite(9,LOW);
97
98   Serial.println(myRTC.minutes);
99   Serial.println(oust);
100
101  t6= myRTC.minutes ;
102  //Enable GPRS module for node101
103  if(t6==4&&oust==1)
104  {
105  digitalWrite(9,LOW);
106  delay(1000);
107  digitalWrite(9,HIGH);
108  delay(1000);
109  digitalWrite(9,LOW);
110  oust=0;
111  }
112
113  //Enable GPRS module for node202
114  if(t6==14&&oust==1)
115  {
116  digitalWrite(9,LOW);
117  delay(1000);
118  digitalWrite(9,HIGH);
119  delay(1000);
120  digitalWrite(9,LOW);
121  oust=0;
```

```
122   }
123   //Disable GPRS module
124     if(t6==5&&oust==1)
125   {
126   digitalWrite(9,LOW);
127   delay(1000);
128   digitalWrite(9,HIGH);
129   delay(3000);
130   oust=0;
131   }
132   if(t6==7)
133   {oust=1;}
134
135   //Disable GPRS module
136       if(t6==15&&oust==1)
137   {
138   digitalWrite(9,LOW);
139   delay(1000);
140   digitalWrite(9,HIGH);
141   delay(3000);
142   oust=0;
143   }
144   if(t6==17)
145   {oust=1;}
146
147   //Start LoRa receiver
148   if(t6==2||t6==12)
149   {
150
151   myRTC.updateTime();
152   Serial.println(myRTC.minutes);
153   mySerial.begin(SERIAL_BAUD);
154
155   // Start receiver
156   mySerial.println(F("radio rx 0"));
157
158   // Wait for ok (50ms)
159   tmp_millis = millis();
160   tmp_str = "";
161   while (millis() - tmp_millis < 50) {
162     while (mySerial.available()) {
```

```
163       tmp_str = mySerial.readStringUntil('\n');
164       if (tmp_str.indexOf("ok") == 0) {
165     Serial.print(F("RX Command: ")); Serial.println(tmp_str);
166         break;
167       }
168     }
169   }
170
171   // Wait for rx data (half minute)
172   tmp_millis = millis();
173   tmp_str = "";
174   while (millis() - tmp_millis < 30000) {
175     while (mySerial.available()) {
176       tmp_str = mySerial.readStringUntil('\n');
177       if (tmp_str.indexOf("radio_rx") == 0) {
178     Serial.print(F("RX: ")); Serial.println(tmp_str);
179       //tmp_millis = -70000 + millis();
180         break;
181       }
182     }}
183    mySerial.end();
184
185    gsmSerial.begin(SERIAL_BAUD);
186
187    //Processing data
188 t0=tmp_str.substring(10,13);
189 t1=tmp_str.substring(13,15);
190 t2=tmp_str.substring(15,17);
191 t4=tmp_str.substring(17,18);
192 String Co=tmp_str.substring(18,21);
193 String batt=tmp_str.substring(21,24);
194 t3=tmp_str.substring(24,27);
195
196 Serial.println(t0);
197 Serial.println(t1);
198 Serial.println(t2);
199 Serial.println(t4);
200 Serial.println(Co);
201 Serial.println(batt);
202 Serial.println(t3);
203
```

```
204 //Check battery for node101
205 if (t0=="101"){
206   float batt12=batt.toFloat() / 150;
207   if ( batt12 <= 3.4)
208   {
209     SendTextMessage101();
210   }
211
212 batt1=String(batt12);
213   Serial.println(batt1);
214   float Co12=Co.toFloat() / 5.0;
215 Co1=String(Co12);
216   Serial.println(Co1);
217   light12=  t3.toInt();
218   if (light12 >= 100)
219   {
220     light1=light12;
221   }
222   if (light12 < 100)
223   {
224     light13=tmp_str.substring(24,26);
225     light1=light13.toInt();
226   }
227  Send2Pachube();
228  oust=1;
229 }
230
231 if (t0=="010"){
232   t0=tmp_str.substring(11,14);
233 t1=tmp_str.substring(14,16);
234 t2=tmp_str.substring(16,18);
235 t4=tmp_str.substring(18,19);
236 String Co=tmp_str.substring(19,22);
237 String batt=tmp_str.substring(22,25);
238 t3=tmp_str.substring(25,28);
239
240 Serial.println(t0);
241 Serial.println(t1);
242 Serial.println(t2);
243 Serial.println(t4);
244 Serial.println(Co);
```

```
245 Serial.println(batt);
246 Serial.println(t3);
247 float batt12=batt.toFloat() / 150;
248  if ( batt12 <= 3.4)
249   {
250     SendTextMessage101();
251   }
252 batt1=String(batt12);
253   Serial.println(batt1);
254   float Co12=Co.toFloat() / 5.0;
255 Co1=String(Co12);
256   Serial.println(Co1);
257   light12=  t3.toInt();
258   if (light12 >= 100)
259   {
260     light1=light12;
261   }
262   if (light12 < 100)
263   {
264     light13=tmp_str.substring(25,27);
265     light1=light13.toInt();
266   }
267  Send2Pachube();
268  oust=1;
269 }
270
271 if (t0=="202"){
272   t0=tmp_str.substring(10,13);
273 t1=tmp_str.substring(13,15);
274 t2=tmp_str.substring(15,17);
275 t4=tmp_str.substring(17,20);
276 String Co=tmp_str.substring(20,23);
277 String batt=tmp_str.substring(23,26);
278 String light=tmp_str.substring(26,29);
279 //t3=tmp_str.substring(24);
280
281 Serial.println(t0);
282 Serial.println(t1);
283 Serial.println(t2);
284 Serial.println(t4);
285 Serial.println(Co);
```

```
286 Serial.println(batt);
287 Serial.println(light);
288
289 //Check battery for node202
290 float batt12=batt.toFloat() / 150;
291  if ( batt12 <= 3.4)
292   {
293     SendTextMessage202();
294   }
295 batt1=String(batt12);
296   Serial.println(batt1);
297   float Co12=Co.toFloat() / 5.0;
298 Co1=String(Co12);
299   Serial.println(Co1);
300       float t412=t4.toFloat() / 5.0;
301 t41=String(t412);
302   Serial.println(t41);
303    light12=  light.toInt();
304   if (light12 >= 100)
305   {
306     light1=light12;
307   }
308   if (light12 < 100)
309   {
310     light13=tmp_str.substring(26,28);
311     light1=light13.toInt();
312   }
313
314
315  Send2Pachube1();
316  oust=1;
317 }
318 if (t0=="020"){
319   t0=tmp_str.substring(11,14);
320 t1=tmp_str.substring(14,16);
321 t2=tmp_str.substring(16,18);
322 t4=tmp_str.substring(18,21);
323 String Co=tmp_str.substring(21,24);
324 String batt=tmp_str.substring(24,27);
325 String light=tmp_str.substring(27,30);
326 //t3=tmp_str.substring(24);
```

```
327
328 Serial.println(t0);
329 Serial.println(t1);
330 Serial.println(t2);
331 Serial.println(t4);
332 Serial.println(Co);
333 Serial.println(batt);
334 Serial.println(light);
335 float batt12=batt.toFloat() / 150;
336  if ( batt12 <= 3.4)
337   {
338   //  SendTextMessage202();
339   }
340 batt1=String(batt12);
341   Serial.println(batt1);
342   float Co12=Co.toFloat() / 5.0;
343 Co1=String(Co12);
344   Serial.println(Co1);
345     float t412=t4.toFloat() / 5.0;
346 t41=String(t412);
347   Serial.println(t41);
348   light12=  light.toInt();
349   if (light12 >= 100)
350   {
351     light1=light12;
352   }
353   if (light12 < 100)
354   {
355     light13=tmp_str.substring(27,29);
356     light1=light13.toInt();
357   }
358
359  Send2Pachube1();
360  oust=1;
361 }
362
363    delay(2000);
364    gsmSerial.end();
365
366
367
```

```
368   }
369 }
370
371
372
373 // GPRS MODULE for Node 101
374 void Send2Pachube()
375 {
376   gsmSerial.println(F("AT"));
377   delay(1000);
378
379   ShowSerialData();
380   gsmSerial.println(F("AT+CPIN?"));
381   delay(1000);
382
383   ShowSerialData();
384
385   gsmSerial.println(F("AT+CREG?"));
386   delay(1000);
387
388   gsmSerial.println(F("AT+CGATT?"));
389   delay(1000);
390
391   gsmSerial.println(F("AT+CIPSHUT"));
392   delay(1000);
393
394   gsmSerial.println(F("AT+CIPSTATUS"));
395   delay(2000);
396
397   gsmSerial.println(F("AT+CIPMUX=0"));
398   delay(2000);
399
400   ShowSerialData();
401
402   gsmSerial.println(F("AT+CSTT=\"internet\""));//start task and setting the APN,
403   delay(1000);
404
405   ShowSerialData();
406
407   gsmSerial.println(F("AT+CIICR"));//bring up wireless connection
408   delay(3000);
```

```
409
410   ShowSerialData();
411
412   gsmSerial.println(F("AT+CIFSR"));//get local IP adress
413   delay(2000);
414
415   ShowSerialData();
416
417   gsmSerial.println(F("AT+CIPSPRT=0"));
418   delay(3000);
419
420   ShowSerialData();
421
422   gsmSerial.println(F("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\""));
423
424   //start up the connection
425   delay(6000);
426
427   ShowSerialData();
428
429   gsmSerial.println(F("AT+CIPSEND"));
430
431   //begin send data to remote server
432   delay(4000);
433   ShowSerialData();
434
435    String str="GET /update?api_key=LILNAAE9RBQS8KK4&field3= " + String(t1)
436    + "&field2=" + String(t2) + "&field4=" + String(light1)
437    + "&field5=" + String(Co1) + "&field6=" + String(t4) + "&field7="
438    + String(batt1);
439
440
441   gsmSerial.println(str);
442   //begin send data to remote server
443   delay(4000);
444   ShowSerialData();
445
446   gsmSerial.println((char)26);//sending
447   delay(5000);//waitting for reply, important! the time is base on the condition of interne
448   gsmSerial.println();
449
```

```
450
451
452   ShowSerialData();
453
454
455
456   gsmSerial.println("AT+CIPSHUT");//close the connection
457   delay(100);
458   ShowSerialData();
459 }
460
461
462 //GPRS MODULE for Node 202
463 void Send2Pachube1()
464 {
465   gsmSerial.println("AT");
466   delay(1000);
467
468   ShowSerialData();
469   gsmSerial.println("AT+CPIN?");
470   delay(1000);
471
472   ShowSerialData();
473
474   gsmSerial.println("AT+CREG?");
475   delay(1000);
476
477   gsmSerial.println("AT+CGATT?");
478   delay(1000);
479
480   gsmSerial.println("AT+CIPSHUT");
481   delay(1000);
482
483   gsmSerial.println("AT+CIPSTATUS");
484   delay(2000);
485
486   gsmSerial.println("AT+CIPMUX=0");
487   delay(2000);
488
489   ShowSerialData();
490
```

```
491   gsmSerial.println(F("AT+CSTT=\"internet\""));//start task and setting the APN,
492   delay(1000);
493
494   ShowSerialData();
495
496   gsmSerial.println("AT+CIICR");//bring up wireless connection
497   delay(3000);
498
499   ShowSerialData();
500
501   gsmSerial.println("AT+CIFSR");//get local IP adress
502   delay(2000);
503
504   ShowSerialData();
505
506   gsmSerial.println("AT+CIPSPRT=0");
507   delay(3000);
508
509   ShowSerialData();
510
511   gsmSerial.println(F("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\""));
512
513   //start up the connection
514   delay(6000);
515
516   ShowSerialData();
517
518   gsmSerial.println("AT+CIPSEND");//begin send data to remote server
519   delay(4000);
520   ShowSerialData();
521
522     String str="GET /update?api_key=5SNUQRDGYVGKH2RY&field1= " + String(t1)
523     + "&field2=" + String(t2) + "&field6=" + String(batt1)
524     + "&field4=" + String(t41) + "&field5=" + String(Co1)
525     + "&field3=" + int(light1)  ;
526   gsmSerial.println(str);//begin send data to remote server
527   delay(4000);
528   ShowSerialData();
529
530   gsmSerial.println((char)26);//sending
531   delay(5000);//waitting for reply, important! the time is base on the condition of interne
```

```
532   gsmSerial.println();
533
534   ShowSerialData();
535
536   gsmSerial.println("AT+CIPSHUT");//close the connection
537   delay(100);
538   ShowSerialData();
539 }
540 void ShowSerialData()
541 {
542  gsmSerial.setTimeout(1000);
543   Serial.println(gsmSerial.readString());
544 }
545
546 //Gateway battery
547 void SendTextMessage()
548 {
549   Serial.println("Sending Text...");
550  gsmSerial.print("AT+CMGF=1\r"); // Set the shield to SMS mode
551   delay(100);
552
553  gsmSerial.print("AT+CMGS=\"+306939687763\"\r");
554   delay(200);
555
556  gsmSerial.print("Low battery - LoRa node 101");
557
558  gsmSerial.print("\r"); //the content of the message
559   delay(500);
560  gsmSerial.print((char)26);//the ASCII code of the ctrl+z is 26 (required according to t
561   delay(100);
562  gsmSerial.println();
563   Serial.println("Text Sent.");
564    delay(500);
565
566
567 }
568 //Node 101 battery
569 void SendTextMessage101()
570 {
571   Serial.println("Sending Text...");
572  gsmSerial.print("AT+CMGF=1\r"); // Set the shield to SMS mode
```

```
573   delay(100);
574
575  gsmSerial.print("AT+CMGS=\"+306945941178\"\r");
576   delay(200);
577
578  gsmSerial.print("Low battery node 101 ");
579
580  gsmSerial.print("\r"); //the content of the message
581   delay(500);
582  gsmSerial.print((char)26);//the ASCII code of the ctrl+z is 26 (required according to the
583   delay(100);
584  gsmSerial.println();
585   Serial.println("Text Sent.");
586    delay(500);
587
588
589 }
590 //Node 202 battery
591 void SendTextMessage202()
592 {
593   Serial.println("Sending Text...");
594  gsmSerial.print("AT+CMGF=1\r"); // Set the shield to SMS mode
595   delay(100);
596
597  gsmSerial.print("AT+CMGS=\"+306945941178\"\r");
598   delay(200);
599
600  gsmSerial.print("Low battery node 202 ");
601
602  gsmSerial.print("\r"); //the content of the message
603   delay(500);
604  gsmSerial.print((char)26);//the ASCII code of the ctrl+z is 26 (required according to the
605   delay(100);
606  gsmSerial.println();
607   Serial.println("Text Sent.");
608    delay(500);
609
610
611 }
612
613
```

```
614 void wakeup_rf() {
615
616   mySerial.end();
617   pinMode(SS_TX_PIN, OUTPUT);
618   digitalWrite(SS_TX_PIN, LOW);
619   delay(10);
620   digitalWrite(SS_TX_PIN, HIGH);
621   mySerial.begin(SERIAL_BAUD);
622   mySerial.write(0x55);
623 }
624 // Reset RF
625 void reset_rf() {
626
627   digitalWrite(RN_RST, LOW);
628   delay(250);
629   digitalWrite(RN_RST, HIGH);
630
631   rn_cmd("", "", 200);
632 }
633
634 // Configure RF parameters
635 boolean set_rf_params() {
636
637   boolean err = false;
638
639   err |= rn_cmd("mac pause", "4294967245"); // Stop LoRaWAN MAC (allowing PHY-level Tx/R
640   err |= rn_cmd("radio set freq " + String(RF_FREQ), "ok");
641   err |= rn_cmd("radio set sf "   + String(RF_SF)  , "ok");
642   err |= rn_cmd("radio set rxbw " + String(RF_BW)  , "ok");
643   err |= rn_cmd("radio set cr 4/" + String(RF_RATE), "ok");
644   err |= rn_cmd("radio set wdt 0", "ok");
645
646   return err;
647 }
648
649 // Get RF parameters
650 boolean get_rf_params() {
651
652   boolean err = false;
653   err |= rn_cmd("radio get freq",   String(RF_FREQ));
654   err |= rn_cmd("radio get sf",   String(RF_SF));
```

```
655   err |= rn_cmd("radio get rxbw",    String(RF_BW));
656   err |= rn_cmd("radio get cr", "4/" + String(RF_RATE));
657
658   return err;
659 }
660
661 // Serial command to RF
662 boolean rn_cmd(String cmd, String resp, int timeout)
663 {
664   // If not empty, send command to Serial
665   if (cmd != "") mySerial.println(cmd);
666   boolean err = true;
667
668   unsigned long tmp_millis = millis();
669
670   // Check for timeout
671   r1 = "";
672   String  tmp_str = "";
673   while (millis() - tmp_millis < timeout) {
674     while (mySerial.available()) {
675       tmp_str = mySerial.readStringUntil('\n');
676       if (tmp_str.indexOf(resp) == 0) {
677         err = false;
678         break;
679       }
680     }
681   }
682   r1 = tmp_str;
683   tmp_millis = millis();
684
685   if (DBG_LVL > 0) {
686     if (resp != "") {
687       if (err) {
688         Serial.print("ERROR! ");
689       } else {
690         Serial.print("OK. ");
691       }
692     }
693
694     char charBuf[50];
695     tmp_str.toCharArray(charBuf, 50);
```

```
696
697       if (charBuf[0] != 98) {
698         Serial.println("CMD: " + cmd + " RESP: " + tmp_str);
699         //Serial.println(tmp_str);
700       }
701       Serial.flush();
702     }
703     return err;
704   }
705
706   // Blink LED
707   void Blink(byte b_pin, int b_times)
708   {
709     if (DBG_LVL > 0) Serial.flush();
710
711     for (int i = 0; (i < b_times); i++) {
712       digitalWrite(b_pin, HIGH);
713       delay(120);
714       digitalWrite(b_pin, LOW);
715       delay(120);
716     }
717   }
```

Βιβλιογραφία

[1] J. Belissent, C. Mines, E. Radcliffe, and Y. Darashkevich, Getting clever about smart cities: New opportunities require new business models. Forrester, 2010.

[2] Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, John Anderson, Wireless Sensor Networks for Habitat Monitoring. .

[3] Cory Beard, High Reliability 4G and 5G Cellular Wireless Services for Smart Cities. IEEE Smart Cities Kansas City Kickoff Workshop, February 2016 in Kansas City, Missouri, USA.

[4] Andrea Molinari, Vincenzo Maltese, Lorenzino Vaccari, Andrea Almi, and Eleonora Bassi, White Paper on Big and Open Data for Trento Municipality. Forrester, 2010.

[5] M.Fazio, A.Celesti, A.Puliafito, M.Villari, Big Data Storage min the Cloud for Smart Environment Monitoring. Procedia Computer Science 52(2015) 500-506.

[6] URL: https://www.google.com.

[7] URL https://en.wikipedia.org/wiki/GSM: .

[8] URL: https://en.wikipedia.org/wiki/Bluetooth.

[9] URL: https://en.wikipedia.org/wiki/Wi-Fi.

[10] URL: http://www.semtech.com/images/datasheet/an1200.22.pdf.

[11] URL: http://ww1.microchip.com/downloads/en/DeviceDoc/50002346C.pdf.

[12] URL: http://www.microchip.com/wwwproducts/en/ATmega328.