ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Σύστημα Προσαρμοστικής Ενεργειακής Διαχείρισης Κτηρίου με χρήση δικτύου αισθητήρων βασισμένο στα πρωτόκολλα ZigBee και RS485

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σάββας Γ. Λεβεντικίδης

Επιβλέπων: Δημήτριος Σούντρης, Καθηγητής Ε.Μ.Π.
Συνεπιβλέπων: Σωτήριος Κοκόσης, Μέλος Ε.ΔΙ.Π

Αθήνα, Οκτώβριος 2024

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Σύστημα Προσαρμοστικής Ενεργειακής Διαχείρισης Κτηρίου με χρήση δικτύου αισθητήρων βασισμένο στα πρωτόκολλα ZigBee και RS485

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σάββας Γ. Λεβεντικίδης

Επιβλέπων: Δημήτριος Σούντρης, Καθηγητής Ε.Μ.Π.
Συνεπιβλέπων: Σωτήριος Κοκόσης, Μέλος Ε.ΔΙ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15η Οκτωβρίου 2024.

.....	.....	.....
Δημήτριος Σούντρης	Παναγιώτης Τσανάκας	Σωτήριος Ξύδης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Επ. Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2024

.....
Σάββας Γ. Λεβεντικίδης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Η παρούσα εργασία εξετάζει το σχεδιασμό και την υλοποίηση ενός συστήματος που στόχο έχει την προσαρμοστική διαχείριση της ενέργειας κτηρίων. Το σύστημα ενσωματώνει συνδυασμό των πρωτοκόλλων ZigBee (για ασύρματη επικοινωνία) και RS485 (για ενσύρματη επικοινωνία) για να δημιουργηθεί ένα δίκτυο κόμβων συνδεδεμένων με διάφορους αισθητήρες. Σημαντικό στοιχείο του δικτύου είναι η «γεφύρωση» του ασύρματου και ενσύρματου υποδικτύου. Οι κόμβοι έχουν δυνατότητα να συλλέγουν δεδομένα μέσω αναλογικών και ψηφιακών εισόδων και να ανταλλάσσουν πληροφορίες μεταξύ τους ώστε να αλληλεπιδρούν κατάλληλα με το περιβάλλον μέσω ψηφιακών και αναλογικών εξόδων. Το έργο περιλαμβάνει τόσο το σχεδιασμό του υλικού (πλακέτα τυπωμένου κυκλώματος - PCB) όσο και την ανάπτυξη λογισμικού (προγραμματισμός μικροελεγκτή). Το σύστημα μπορεί να ενσωματωθεί σε διαφορετικές και εξατομικευμένες εφαρμογές ρυθμίζοντας κατάλληλα τις εισόδους τις εξόδους και τους ελέγχους που πραγματοποιεί, μέσω Software. Η εργασία υπογραμμίζει τη σημασία της αποδοτικής διαχείρισης της ενέργειας στα κτήρια και τις δυνατότητες των τεχνολογιών IoT να επιτύχουν σημαντική εξοικονόμηση ενέργειας και μείωση των λειτουργικών δαπανών. Τέλος επιδεικνύει την λειτουργία του συστήματος μέσα από μια εξατομικευμένη εφαρμογή.

Λέξεις Κλειδιά:
ZigBee, RS-485, μικροελεγκτής, διαχείριση ενέργειας, ενέργεια κτηρίων, πλακέτα, συ-σχεδιασμός Hardware-Software, εφαρμογή πραγματικού χρόνου, IoT

Abstract

This work examines the design and implementation of a system aimed at the adaptive energy management of buildings. The system incorporates a combination of the ZigBee protocol (for wireless communication) and the RS485 protocol (for wired communication) to create a network of nodes connected to various sensors. A key feature of the network is the "bridging" of the wireless and wired subnetworks. The nodes are capable of collecting data through analog and digital inputs and exchanging information among themselves to appropriately interact with the environment via digital and analog outputs. The project involves both the design of the hardware (Printed Circuit Board - PCB) and the development of software (microcontroller programming). The system can be integrated into different and customized applications by appropriately configuring the inputs, outputs, and controls performed through the software. The work emphasizes the importance of efficient energy management in buildings and the potential of IoT technologies to achieve significant energy savings and reduce operational costs. Finally, it demonstrates the functionality of the system through a customized application.

Keywords:
ZigBee, RS-485, μικροελεγκτής, energy management, buildings energy, PCB, Hardware-Software Co-Design, real-time application, IoT

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Δημήτριο Σούντρη για την υποστήριξη του καθ' όλη την διάρκεια των σπουδών μου καθώς και για την συμβολή του στην χάραξη της ακαδημαϊκής μου πορείας.

Επιπλέον, θα ήθελα να ευχαριστήσω τον συνεπιβλέποντα της εργασίας Σωτήριο Κοκόση, μέλος του Εργαστηριακού Διδακτικού Προσωπικού.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια και τους φίλους μου για την διαρκώς ενεργή υποστήριξη τους.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

1 Εισαγωγή

Τα κτήρια σήμερα τροφοδοτούνται με ηλεκτρική ενέργεια, η οποία έχει γίνει απαραίτητο στοιχείο της σύγχρονης ζωής, καλύπτοντας ανάγκες όπως θέρμανση, ψύξη, φωτισμό και τη λειτουργία συσκευών και υπολογιστών των ενοίκων τους. Η διαχείριση της ενέργειας στα κτήρια έχει εξελιχθεί σε μια απαιτητική διαδικασία, καθώς η συνολική κατανάλωση ενέργειας, μαζί με την κατανάλωση ενέργειας στα κτήρια, έχει αυξηθεί σημαντικά τις τελευταίες δεκαετίες.

Οι αισθητήρες IoT προσφέρουν λύσεις για την καλύτερη διαχείριση της ενεργειακής κατανάλωσης, εντοπίζοντας δραστηριότητες που μπορούν να μειώσουν την κατανάλωση ενέργειας και να περιορίσουν τα λειτουργικά κόστη. Μέσω των τεχνολογιών IoT, η χρήση αισθητήρων επιτρέπει την παρακολούθηση σε πραγματικό χρόνο της ενέργειας που καταναλώνεται σε ένα κτήριο. Παράλληλα, οι αισθητήρες συμβάλλουν στην ανάλυση των εξωτερικών κλιματικών συνθηκών και στην αυτόματη ρύθμιση των συστημάτων θέρμανσης και ψύξης, προκειμένου να εξασφαλιστεί η βέλτιστη κατανάλωση ενέργειας. Η τεχνολογία IoT έχει επεκταθεί ταχύτατα σε πολλούς τομείς, χάρη στην εύκολη πρόσβαση σε χαμηλού κόστους αισθητήρες, που αυξάνουν την αποδοτικότητα των ενεργειακών συστημάτων σε κτίρια σε παγκόσμιο επίπεδο, συμβάλλοντας έτσι και στην προστασία του περιβάλλοντος.

Σκοπός αυτής της εργασίας είναι η δημιουργία ενός συστήματος που μπορεί να χρησιμοποιηθεί είτε ως διακομιστής είτε ως κόμβος σε ένα δίκτυο που αποτελείται τόσο από κόμβους ασύρματα συνδεδεμένους μέσω του πρωτοκόλλου ZigBee όσο και από κόμβους ενσύρματα συνδεδεμένους μέσω πρωτοκόλλου RS-485. Βασικό χαρακτηριστικό του δικτύου είναι η «γέφυρα» επικοινωνίας μεταξύ ενός ασύρματου πρωτοκόλλου (ZigBee) και ενός ενσύρματου πρωτοκόλλου (RS-485). Κάθε κόμβος-πλακέτα του δικτύου έχει την δυνατότητα να λάβει πληροφορίες από διαφορετικά είδη αισθητήρων και να μεταδώσει τις πληροφορίες αυτές σε όλους τους κόμβους. Ακόμη μπορεί να επενεργήσει με το περιβάλλον μέσω ψηφιακών και αναλογικών εξόδων οι οποίες μπορούν να χρησιμοποιηθούν για την επίτευξη της ενεργειακής διαχείρισης ενός κτηρίου. Οι εφαρμογές του δικτύου αυτού είναι πολλαπλές καθώς υπάρχουν

πολλαπλά είδη αισθητήρων που παρέχουν όλων των ειδών τις πληροφορίες αλλά και η αλληλεπίδραση του δικτύου με το περιβάλλον μπορεί να αποτελεί εξατομικευμένη εφαρμογή για διαφορετικά είδη κτηρίων, διαφορετικές περιστάσεις και προσωπικές επιλογές.

Στην δημιουργία των πλακετών αυτών εξίσου σημαντικό ρόλο παίζει το hardware όσο και το software. Για να επιτευχθεί το τελικό αποτέλεσμα χρειάστηκε συστηματικός συν-σχεδιασμός (co-design) Hardware με Software αφού συνδυάζονται πρωτόκολλα επικοινωνίας τόσο ασύρματα, τα οποία βασίζονται σε πολύ μεγάλο βαθμό στην στοίβα πρωτοκόλλων τους όσο και σε ενσύρματα που ρίχνουν περισσότερο βάρος στην κυκλωματική τους υπόσταση.

Η παρούσα εργασία είναι διαρθρωμένη σε 6 κεφάλαια ως εξής:

- Στο κεφάλαιο 1 εισάγεται το γενικότερο θέμα που αφορά η εργασία.
- Στο κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο πίσω από το αντικείμενο της εργασίας, δηλαδή τα πρωτόκολλα ZigBee και RS485, περιγράφεται ο μικροελεγκτής και το λογισμικό της ST Microelectronics που χρησιμοποιήθηκε για την υλοποίηση του συστήματος και εφαρμογές ενεργειακής διαχείρισης κτηρίων.
- Στο κεφάλαιο 3 περιγράφεται η δομή και η λειτουργία του hardware που σχεδιάστηκε υλοποιήθηκε στο πλαίσιο της εργασίας.
- Στο κεφάλαιο 4 αναλύεται η δομή και η λειτουργία του software που αναπτύχθηκε για το σύστημα.
- Στο κεφάλαιο 5 παρουσιάζεται η υλοποιημένη εφαρμογή ενεργειακής διαχείρισης κτηρίου με το σύστημα που αναπτύχθηκε.
- Το κεφάλαιο 6 συνοψίζει την παρούσα εργασία

2 Θεωρητικό Υπόβαθρο

2.1 Πρωτόκολλο ZigBee

Γενικά Χαρακτηριστικά

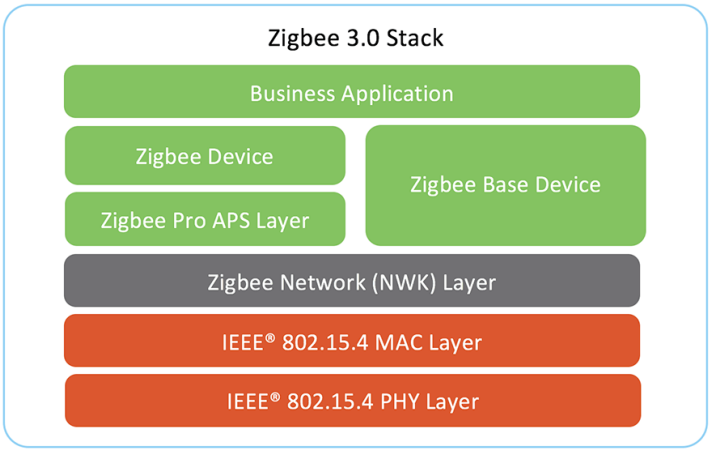
Το ZigBee είναι μια ασύρματη τεχνολογία που αναπτύχθηκε ως ανοιχτό παγκόσμιο πρότυπο συνδεσιμότητας για να ανταποκριθεί στις μοναδικές ανάγκες των ασύρματων δικτύων δεδομένων IoT χαμηλού κόστους και χαμηλής κατανάλωσης ενέργειας. Είναι σχεδιασμένη να υποστηρίζει ασύρματες επικοινωνίες, παρακολούθηση και έλεγχο συσκευών που λειτουργούν με μπαταρία και δικτύων αισθητήρων. Το ZigBee βασίζεται στην προδιαγραφή IEEE 802.15.4 και χρησιμοποιείται ευρέως στους οικιακούς αυτοματισμούς, στη συλλογή ιατρικών δεδομένων και στα βιομηχανικά συστήματα ελέγχου. [1]

Το πρωτόκολλο ZigBee είναι μια ανοιχτή ασύρματη προδιαγραφή που δημιουργήθηκε από το ZigBee Alliance και τυποποιεί τον τρόπο με τον οποίο τα δίκτυα IoT χαμηλής κατανάλωσης ενέργειας μπορούν να επικοινωνούν με ασφάλεια και αξιοπιστία [2]

- Πρότυπο ZigBee: Το ZigBee Alliance ανέπτυξε το πρότυπο ZigBee προσθέτοντας ένα επίπεδο δικτύου, ένα επίπεδο ασφαλείας και ένα πλαίσιο εφαρμογών πάνω στο πρότυπο IEEE 802.15.4.
- ZigBee 3.0: Η προδιαγραφή ZigBee Pro προσθέτει νέες λειτουργίες, όπως τη διαχείριση συσκευών-παιδιών, βελτιωμένη ασφάλεια και νέες επιλογές τοπολογίας δικτύου.

Οι συσκευές ZigBee δημιουργούν αδιάλειπτα ένα mesh δίκτυο, επιτρέποντας την αποδοτική μεταφορά δεδομένων μέσω ενός κεντρικού κόμβου συνδεδεμένου σε μια πύλη (gateway) για απομακρυσμένη πρόσβαση στο διαδίκτυο. Μια εφαρμογή ZigBee επιτρέπει στον χρήστη να ελέγχει έξυπνες συσκευές από οπουδήποτε. Το ZigBee βασίζεται στο φυσικό επίπεδο (Physical Layer) και στο υποεπίπεδο ελέγχου πρόσβασης μέσων (Medium Access Control), όπως ορίζεται στο πρότυπο IEEE

802.15.4, το οποίο διαχειρίζεται τις βασικές λειτουργίες του δικτύου. Το Επίπεδο Δικτύου (Network Layer) του ZigBee διαχειρίζεται τη δομή του δικτύου, τη δρομολόγηση και την ασφάλεια, ενώ το επίπεδο εφαρμογών (Application Layer) περιλαμβάνει το υποεπίπεδο υποστήριξης εφαρμογών, τα αντικείμενα συσκευών ZigBee και εφαρμογές καθορισμένες από τον χρήστη. [3]



Εικόνα 1:Στοίβα Πρωτοκόλλου ZigBee

Η τυποποίηση του δικτύου ZigBee διασφαλίζει ότι όλα τα προϊόντα και οι υπηρεσίες θα συνεργάζονται μεταξύ τους. Το πιο πρόσφατο πρωτόκολλο ZigBee, το ZigBee Pro 2023, προσθέτει βελτιώσεις στην ασφάλεια και υποστήριξη για ζώνες συχνοτήτων πέραν των 2.4 GHz. Η συχνότητα των 800 MHz για την Ευρώπη και η συχνότητα των 900 MHz για τη Βόρεια Αμερική και την Αυστραλία βελτιώνει την ισχύ και την εμβέλεια του σήματος για ευρύτερη χρήση. Τα βασικά χαρακτηριστικά του ZigBee περιλαμβάνουν [4]:

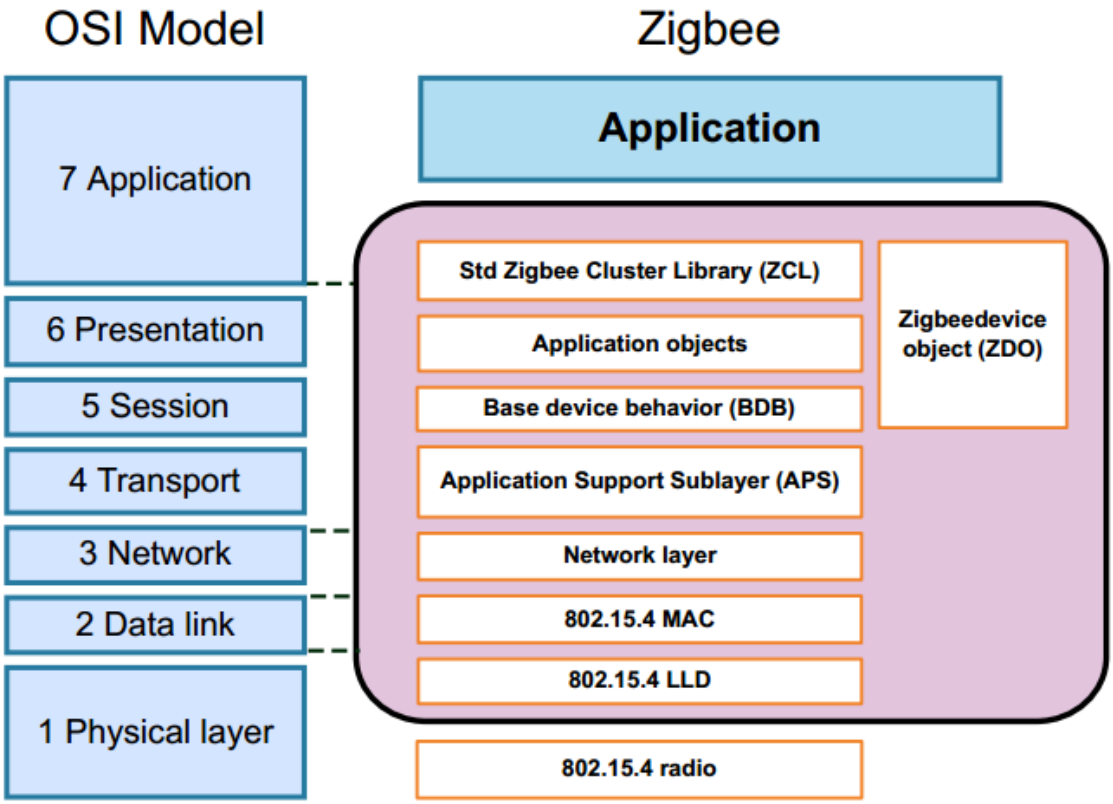
- Ευελιξία: Υποστηρίζει πολλαπλές τοπολογίες δικτύων, όπως σημείο-προς-σημείο (point-to-point), σημείο προς πολλαπλά σημεία και mesh δίκτυα.
- Χαμηλό duty cycle: Παρέχει μακρά διάρκεια ζωής της μπαταρίας.
- Χαμηλή καθυστέρηση: Μεταφέρει εύκολα δεδομένα με ελάχιστη καθυστέρηση
- Κλιμακωσιμότητα: Περιλαμβάνει το Direct Sequence Spread Spectrum (DSSS) με δυνατότητα υποστήριξης έως 65.000 κόμβων ανά δίκτυο.
- Ευρωστία: Χρησιμοποιεί collision avoidance, retries and acknowledgments.
- Χαμηλή κατανάλωση ενέργειας: Οι συσκευές ZigBee μπορούν να λειτουργούν για αρκετά χρόνια με μια απλή και φθηνή μπαταρία χάρη στη λειτουργία εξοικονόμησης ενέργειας (sleep mode).

• Χαμηλός ρυθμός μετάδοσης δεδομένων: Με ρυθμό έως 250 kbit/s, το ZigBee είναι κατάλληλο για διαλείπουσες μεταδόσεις δεδομένων αισθητήρων ή συσκευών.

• Ασφάλεια: Το ZigBee χρησιμοποιεί κρυπτογράφηση AES 128-bit καθώς και πολλές επιπλέον τεχνικές ασφαλείας.

Επίπεδα στοίβας Zigbee

Τα επίπεδα της στοίβας, όπως ορίζονται από την προδιαγραφή Zigbee, βασίζονται στο μοντέλο των 7 επιπέδων OSI. Στην περίπτωση του Zigbee, αφορούν τα επίπεδα δικτύου και το πλαίσιο εφαρμογών. Η στοίβα Zigbee είναι χωρισμένη σε πολλά συστατικά, όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 2:Περιγραφή Στοίβας Πρωτοκόλλου ZigBee

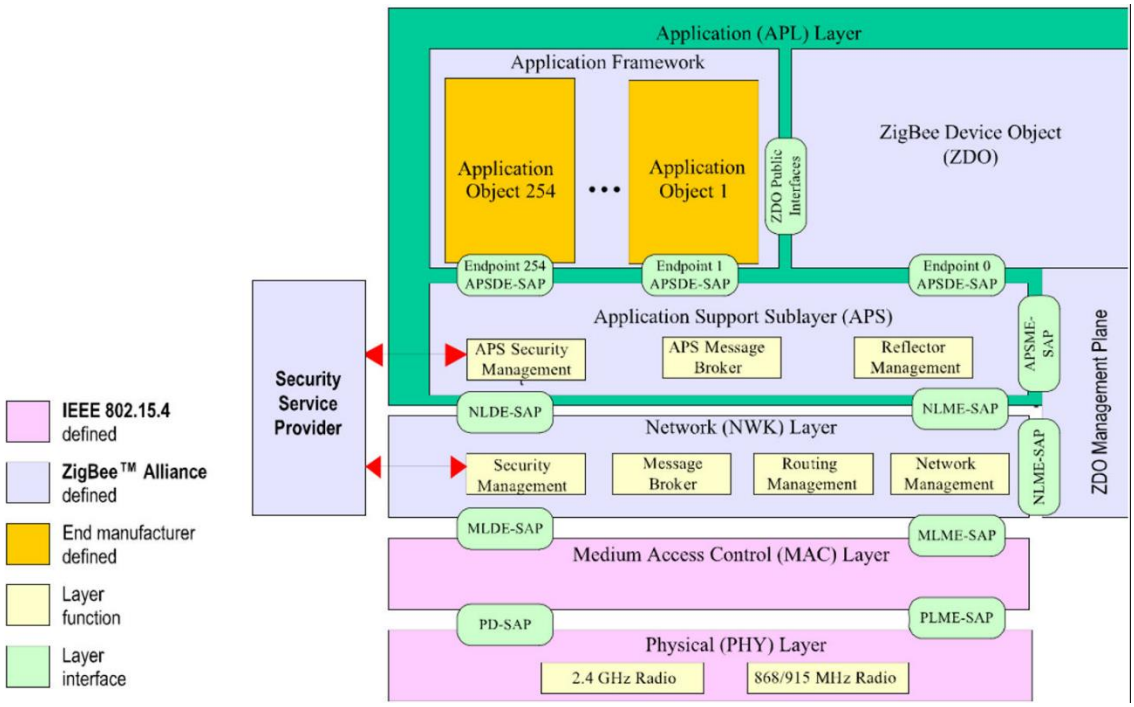
Επίπεδο Δικτύου (NWK)

Το επίπεδο δικτύου απαιτείται για να παρέχει λειτουργίες που εξασφαλίζουν τη σωστή λειτουργία του υποεπιπέδου MAC του προτύπου IEEE 802.15.4 και να παρέχει μια κατάλληλη διεπαφή υπηρεσίας προς το επίπεδο εφαρμογής. Μεταξύ

άλλων, αυτό είναι το επίπεδο στο οποίο τα δίκτυα ξεκινούν, συνδέονται, αποσυνδέονται και ανακαλύπτονται [5].

Επίπεδο Εφαρμογής (APL)

Το επίπεδο εφαρμογής αποτελείται από διάφορα υποεπίπεδα. Τα συστατικά του επιπέδου εφαρμογής παρουσιάζονται παρακάτω:



Εικόνα 3:Επίπεδο Εφαρμογής και υποεπίπεδα

Υποεπίπεδο Υποστήριξης Εφαρμογών (APS)

APS σημαίνει υποεπίπεδο υποστήριξης εφαρμογών. Παρέχει μια διεπαφή μεταξύ του επιπέδου δικτύου (NWK) και του επιπέδου εφαρμογής μέσω ενός γενικού συνόλου υπηρεσιών που χρησιμοποιούνται τόσο από το ZDO όσο και από τα αντικείμενα εφαρμογής που ορίζονται από τον κατασκευαστή. Το APS είναι υπεύθυνο για:

- διαχείριση σύνδεσης (binding)
- προώθηση μηνυμάτων μεταξύ συνδεδεμένων συσκευών
- ορισμό και διαχείριση ομαδικών διευθύνσεων
- αντιστοίχιση διευθύνσεων από 64-bit επεκταμένες διευθύνσεις σε 16-bit NWK
- διευθύνσεις (ειδικός πίνακας)
- κατακερματισμό και επανασυναρμολόγηση πακέτων

– αξιόπιστη μεταφορά δεδομένων

Η σύνδεση (binding) στο Zigbee επιτρέπει σε ένα σημείο τερματισμού σε έναν κόμβο να συνδέεται ή να "συνδέεται" με ένα ή περισσότερα σημεία τερματισμού σε έναν άλλο κόμβο [5].

Ο πίνακας σύνδεσης αντιστοιχίζει μια πηγή διεύθυνσης και ένα σημείο τερματισμού σε μία ή περισσότερες διευθύνσεις και σημεία τερματισμού προορισμού. Αυτός ο πίνακας είναι διαθέσιμος και διατηρείται σε όλες τις συσκευές του δικτύου.

Αντικείμενο Συσκευής Zigbee (ZDO)

Το στοιχείο ZDO χειρίζεται τις λειτουργίες διαχείρισης συσκευών και επικοινωνίας. Περιλαμβάνει:

– αρχικοποίηση του υποεπιπέδου APS και του επιπέδου NWK
– ανακάλυψη συσκευών
– ανακάλυψη υπηρεσιών
– διαχείριση δικτύου, συμπεριλαμβανομένου του ορισμού της λειτουργικής κατάστασης της συσκευής (ZC, ZR ή ZED)
– διαχείριση ασφάλειας
– εκκίνηση και/ή απόκριση σε αιτήματα σύνδεσης από απόσταση (remote binding).

Βασική Συμπεριφορά Συσκευής (BDB)

Πρόκειται για ένα τυπικό λογισμικό στοιχείο που χειρίζεται βασικές λειτουργίες, όπως η ανάθεση (commissioning), η ασφάλεια δικτύου και η διαχείριση επίμονων δεδομένων. Αυτή η συσκευή δεν χρειάζεται σημείο τερματισμού (endpoint) [5].

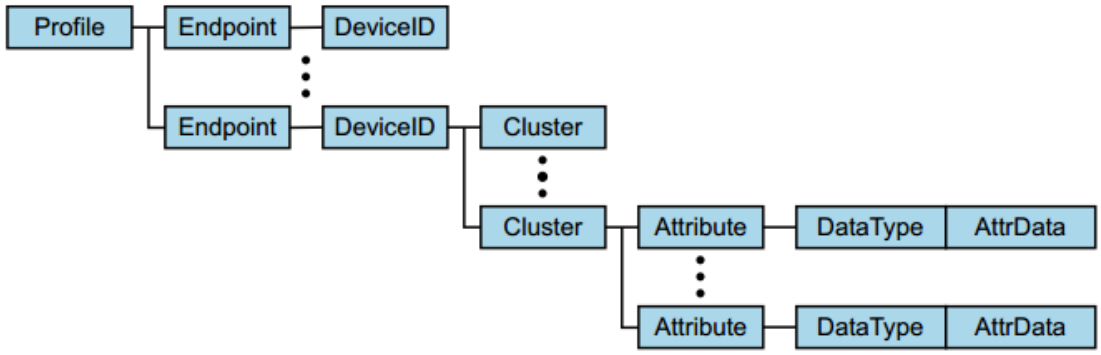
Πλαίσιο Εφαρμογής

Το πλαίσιο εφαρμογής στο Zigbee είναι πολύ πλούσιο και ορίζει το περιβάλλον στο οποίο φιλοξενούνται τα αντικείμενα εφαρμογής στις συσκευές.

Η ανταλλαγή δεδομένων μεταξύ συσκευών Zigbee πραγματοποιείται μέσω ενός μοντέλου πελάτη-διακομιστή. Βασίζεται σε ένα Προφίλ Εφαρμογής,Clusters και Attributes.

Το Προφίλ Εφαρμογής είναι μια συλλογή περιγραφών συσκευών, οι οποίες μαζί σχηματίζουν μια συνεργατική εφαρμογή. Το Προφίλ ορίζει τη μορφή ανταλλαγής δεδομένων για τις λειτουργίες εφαρμογών μιας φυσικής συσκευής Zigbee. Ένα

Προφίλ αποτελείται από ένα ή περισσότερα Σημεία Τερματισμού (Endpoints), το καθένα με μία ή περισσότερες συσχετισμένες Συστάδες (Clusters).



Εικόνα 4:Οργάνωση Προφίλ Εφαρμογής ZigBee

Οι συστάδες αποτελούν μια ομάδα εντολών και χαρακτηριστικών που καθορίζουν τις δυνατότητες μιας συσκευής. Η διαχείριση των συστάδων γίνεται από τη Βιβλιοθήκη Συστάδων Zigbee (ZCL - Zigbee Cluster Library).
Ο αριθμός των Σημείων Τερματισμού (Endpoints) που μπορούν να χρησιμοποιηθούν για μια εφαρμογή Zigbee κυμαίνεται μεταξύ 1 και 240 [5].

Βιβλιοθήκη Συστάδων(Cluster) Zigbee (ZCL)

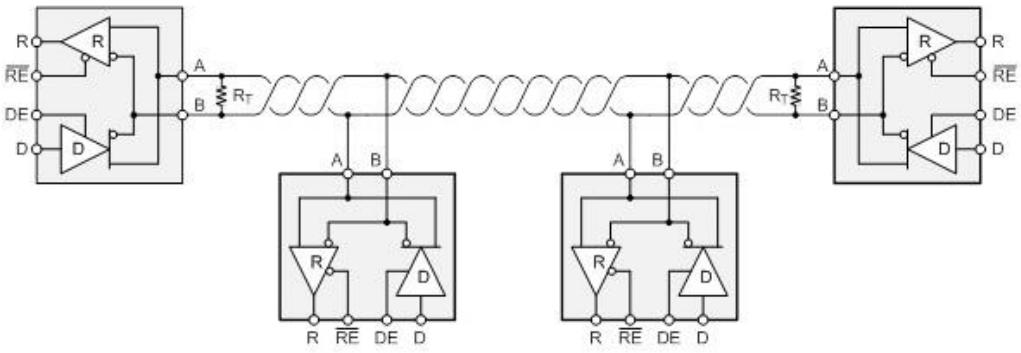
Η ZCL είναι η βιβλιοθήκη που διαχειρίζεται τα Clusters. Τα custers μπορούν να θεωρηθούν ως μια ομάδα εντολών και χαρακτηριστικών που είναι συγκεκριμένα για μια αφιερωμένη εφαρμογή (π.χ. Κλείδωμα Πόρτας, Αναμονή/Ενεργοποίηση κ.λπ.).
Η ZCL ορίζεται από την Zigbee Alliance με σκοπό την επιτάχυνση της ανάπτυξης και τυποποίησης των δημόσιων προφίλ. Με την ZCL, οι κατασκευαστές μπορούν να δημιουργούν γρήγορα προϊόντα Zigbee με συνέπεια και συμβατότητα.
Ένα cluster είναι μια σχετική συλλογή εντολών και χαρακτηριστικών, η οποία μαζί ορίζει μια διεπαφή σε συγκεκριμένη λειτουργικότητα. Οι εντολές είναι ενέργειες(attributes) που μπορεί να αναλάβει ένα cluster. Τα Attributes είναι δεδομένα ή καταστάσεις εντός ενός cluster [5].

2.2 Σειριακή Επικοινωνία RS485

Το RS-485 (γνωστό και ως TIA/EIA-485) είναι ένα βιομηχανικό πρότυπο που ορίζει τη φυσική διεπαφή και το φυσικό επίπεδο για την επικοινωνία σημείο-προς-σημείο (point-to-point) μεταξύ ηλεκτρικών συσκευών. Το πρότυπο RS-485 επιτρέπει

μεγάλες αποστάσεις καλωδίωσης σε ηλεκτρικά θορυβώδη περιβάλλοντα και μπορεί να υποστηρίξει πολλαπλές συσκευές στον ίδιο δίαυλο. [6]

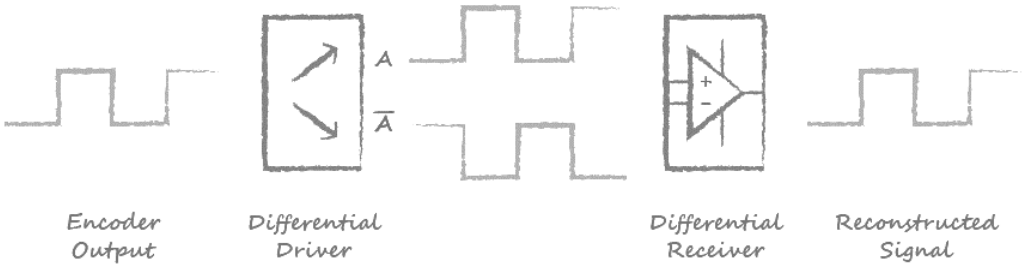
Το RS-485 χρησιμοποιεί δύο γραμμές σήματος, "A" και "B", οι οποίες πρέπει να είναι ισορροπημένες και διαφορικές, ώστε να καταστέλλει τις παρεμβολές κοινού σήματος (common mode interference). Οι ισορροπημένες γραμμές είναι δύο αγωγοί που μοιράζονται ένα ζεύγος σε ένα καλώδιο συνεστραμμένου ζεύγους, με την ίδια σύνθετη αντίσταση σε κάθε γραμμή. Εκτός από την προσαρμοσμένη σύνθετη αντίσταση των γραμμών, πρέπει να υπάρχει επίσης προσαρμογή σύνθετης αντίστασης στον δέκτη και τον πομπό. Ιδανικά, τα δύο άκρα του καλωδίου πρέπει να έχουν μια τερματική αντίσταση συνδεδεμένη ανάμεσα στις δύο γραμμές σήματος. Χωρίς τις τερματικές αντιστάσεις, οι ανακλάσεις του σήματος από τα μη τερματισμένα άκρα του καλωδίου μπορούν να προκαλέσουν αλλοίωση των δεδομένων. Οι τερματικές αντιστάσεις επίσης μειώνουν την ευαισθησία στον ηλεκτρικό θόρυβο λόγω της χαμηλότερης σύνθετης αντίστασης. Η τερματική αντίσταση περιλαμβάνει επίσης αντιστάσεις pull-up και pull-down για πολώσουν κάθε γραμμή στην περίπτωση που καμία συσκευή δεν οδηγεί τις γραμμές. Αυτές οι αντιστάσεις εξασφαλίζουν ότι οι γραμμές δεν "αιωρούνται" (floating) όταν δεν υπάρχει ενεργή συσκευή που να στέλνει δεδομένα, αποτρέποντας την εμφάνιση αβέβαιων λογικών καταστάσεων στο δίκτυο RS-485. [7,8]



Εικόνα 5:Δίκτυο RS-485

Σε μια διεπαφή μονής τερματικής γραμμής (single-ended), το λαμβανόμενο σήμα αναφέρεται στη στάθμη τάσης της γης και η κατάστασή του καθορίζεται με βάση προκαθορισμένα επίπεδα τάσης (γνωστά ως λογικά επίπεδα, τα οποία καθορίζουν αν το σήμα είναι λογικό υψηλό ή χαμηλό). Ωστόσο, σε μεγάλες αποστάσεις καλωδίωσης, όπου οι τάσεις τείνουν να μειώνονται συχνά εμφανίζονται σφάλματα. Σε μια διαφορική εφαρμογή, η συσκευή host δημιουργεί το αρχικό single-ended σήμα, το οποίο στη συνέχεια κατευθύνεται σε έναν διαφορικό πομπό. Αυτός ο

πομπός δημιουργεί το διαφορικό σήμα που αποστέλλεται μέσω του καλωδίου. Με δύο παραγόμενα σήματα, ο δέκτης πλέον δεν αναφέρεται στο επίπεδο τάσης σε σχέση με το έδαφος, αλλά αναφέρεται στα ίδια τα σήματα μεταξύ τους. Αυτό σημαίνει ότι αντί να ψάχνει συγκεκριμένα επίπεδα τάσης, ο δέκτης εξετάζει πάντα τη διαφορά μεταξύ των δύο σημάτων. Ο διαφορικός δέκτης ανακατασκευάζει το ζεύγος σημάτων σε ένα single-ended σήμα που μπορεί να ερμηνευθεί από τη συσκευή host, χρησιμοποιώντας τα σωστά λογικά επίπεδα που απαιτούνται από τη συσκευή. Αυτού του τύπου η διεπαφή επιτρέπει επίσης τη λειτουργία συσκευών με διαφορετικά επίπεδα τάσης, μέσω της επικοινωνίας των διαφορικών πομποδεκτών. Όλα αυτά συνεργάζονται για να ξεπεράσουν την υποβάθμιση του σήματος που θα συνέβαινε με μια εφαρμογή μονής τερματικής γραμμής σε μεγάλες αποστάσεις καλωδίωσης. [8,9]



Εικόνα 6:Διαμόρφωση και Αποδιαμόρφωση σήματος RS-485

Ένα από τα σημαντικά πλεονεκτήματα του φυσικού επιπέδου του RS-485 είναι η προδιαγραφή τάσης σήματος. Το RS-485 δεν απαιτεί τη χρήση συγκεκριμένης τάσης στον δίαυλο, αλλά καθορίζει τη minimum απαιτούμενη διαφορική τάση, δηλαδή τη διαφορά μεταξύ των τάσεων των σημάτων A και B. Ο δίαυλος απαιτεί ελάχιστη διαφορική τάση ±200 mV στον δέκτη, και γενικά όλες οι συσκευές RS-485 θα έχουν το ίδιο εύρος εισόδου τάσης, παρά το γεγονός ότι μεταδίδουν με διάφορες τάσεις. Αυτό σημαίνει ότι οποιαδήποτε συσκευή RS-485 μπορεί να λάβει σήματα στην κλίμακα τάσης από -7 έως 12 V, επιτρέποντας στους μηχανικούς να σχεδιάσουν το σύστημα της συσκευής host με οποιαδήποτε τάση μετάδοσης εντός αυτής της κλίμακας. [9]

2.3 Μικροελεγκτής STM32WB55RGV6 [10]

Περιγραφή

Οι συσκευές STM32WB55xx είναι σχεδιασμένες για ασύρματη επικοινωνία και εξαιρετικά χαμηλή κατανάλωση ενέργειας, με υποστήριξη πολλαπλών πρωτοκόλλων, όπως Bluetooth® Low Energy και IEEE 802.15.4. Διαθέτουν δύο πυρήνες: έναν Arm® Cortex®-M4 για εφαρμογές υψηλής απόδοσης και έναν Arm® Cortex®-M0+ για διαχείριση των λειτουργιών πραγματικού χρόνου στα χαμηλότερα επιπέδα.

Ο πυρήνας Cortex®-M4 προσφέρει υψηλή απόδοση λειτουργώντας σε συχνότητα έως 64 MHz και ενσωματώνει μονάδα κινητής υποδιαστολής (FPU), η οποία υποστηρίζει υπολογισμούς μονής ακριβείας, καθώς και εντολές DSP για επεξεργασία ψηφιακών σημάτων. Επίσης, περιλαμβάνει μονάδα προστασίας μνήμης (MPU) για ενισχυμένη ασφάλεια εφαρμογών.

Τα μοντέλα αυτά προσφέρουν ευρύ φάσμα δυνατοτήτων αποθήκευσης, όπως Flash μνήμη έως 1 MB καθώς και SRAM με χωρητικότητα έως 256 KB Διαθέτουν Quad-SPI interface για γρήγορη αποθήκευση και ανάγνωση δεδομένων.

Η επικοινωνία μεταξύ των δύο πυρήνων διευκολύνεται από το IPCC με έξι αμφίδρομα κανάλια και τα hardware semaphores του HSEM, που επιτρέπουν κοινή χρήση πόρων με ασφάλεια.

Στο θέμα της ασφάλειας, οι συσκευές προσφέρουν λειτουργίες προστασίας ανάγνωσης και εγγραφής της μνήμης, καθώς και κρυπτογραφικές δυνατότητες μέσω δύο AES μηχανών κρυπτογράφησης, PKA (Public Key Accelerator), και RNG (Random/True Random Number Generator). Η λειτουργία Customer Key Storage διασφαλίζει την προστασία των κρυπτογραφικών κλειδιών.

Για επεξεργασία σημάτων, περιλαμβάνουν 12-bit ADC και συγκριτές χαμηλής κατανάλωσης, μαζί με γεννήτρια τάσης αναφοράς υψηλής ακρίβειας. Περιλαμβάνουν επίσης πλήρες σύνολο χρονομέτρων, όπως 16-bit και 32-bit timers, καθώς και RTC για λειτουργία πραγματικού χρόνου, η οποία μπορεί να διατηρείται ακόμη και όταν η κύρια τροφοδοσία δεν είναι διαθέσιμη μέσω της παροχής VBAT.

Οι συσκευές υποστηρίζουν ευρύ φάσμα επικοινωνιακών διεπαφών, όπως USART, LPUART, I2C, SPI, SAI, USB 2.0 FS, και περιλαμβάνουν DMA με 14 κανάλια για αποδοτική μεταφορά δεδομένων. Η υποστήριξη του USB περιλαμβάνει

ενσωματωμένο κρύσταλλο και δυνατότητες BCD (Battery fallback detection) και LPM (Low Power Mode).

Η ενεργειακή αποδοτικότητα ενισχύεται με έναν SMPS step-down μετατροπέα που προσαρμόζεται αυτόματα ανάλογα με την τάση του συστήματος, και αρκετές λειτουργίες εξοικονόμησης ενέργειας για χαμηλής κατανάλωσης εφαρμογές. Διατίθενται σε διάφορες διαμορφώσεις και συσκευασίες με διαφορετικό αριθμό pins.

Συνολικά, οι συσκευές αυτές είναι ιδανικές για εφαρμογές που απαιτούν χαμηλή κατανάλωση ενέργειας, ισχυρή επεξεργαστική απόδοση και ασφαλή ασύρματη επικοινωνία.

Χαρακτηριστικά

- **Ενσωματώνει την πατενταρισμένη τεχνολογία αιχμής ST**

- **PF πομποδέκτης**
  - 2.4 GHz
  - RF πομποδέκτης που υποστηρίζει την προδιαγραφή Bluetooth® 5.4, IEEE 802.15.4-2011 PHY και MAC, υποστηρίζοντας Thread 1.3 και Zigbee® 3.0
  - Ευαισθησία RX: -96 dBm (Bluetooth® Low Energy στα 1 Mbps), -100 dBm (802.15.4)
  - Προγραμματιζόμενη ισχύς εξόδου έως +6 dBm με βήματα 1 dB
  - Ενσωματωμένο balun για μείωση του BOM
  - Υποστήριξη για 2 Mbps
  - Υποστήριξη GATT caching
  - Υποστήριξη EATT (enhanced ATT)
  - Υποστήριξη advertising extension
  - Αποκλειστικός CPU Arm® 32-bit Cortex® M0+ για το επίπεδο RF σε πραγματικό χρόνο
  - Ακριβές RSSI για ενεργοποίηση του ελέγχου ισχύος
  - Κατάλληλο για συστήματα που απαιτούν συμμόρφωση με τους κανονισμούς ραδιοσυχνοτήτων ETSI EN 300 328, EN 300 440, FCC CFR47 Part 15 και ARIB STD-T66
  - Υποστήριξη για εξωτερική PA
  - Διαθέσιμο ενσωματωμένο παθητικό τσιπ συσκευής (IPD) για βελτιστοποιημένη

λύση αντιστοίχισης (MLPF-WB-01E3, ή MLPF-WB55-02E3, ή MLPF-WB-02D3)

- **Πλατφόρμα εξαιρετικά χαμηλής κατανάλωσης**
  - Τροφοδοσία 1.71 έως 3.6 V
  - Εύρος θερμοκρασίας -40 °C έως 85 / 105 °C
  - Λειτουργία απενεργοποίησης 13 nA
  - Λειτουργία αναμονής 600 nA + RTC + 32 KB RAM
  - Λειτουργία διακοπής 2.1 μA + RTC + 256 KB RAM
  - Λειτουργία ενεργού MCU: < 53 μA / MHz όταν το RF και το SMPS είναι ενεργοποιημένα
  - Ραδιόφωνο: Rx 4.5 mA / Tx στα 0 dBm 5.2 mA

- **Πυρήνας:** CPU Arm® 32-bit Cortex®-M4 με FPU, προσαρμοστικό επιταχυντή πραγματικού χρόνου (ART™ Accelerator) που επιτρέπει εκτέλεση χωρίς καθυστέρηση από τη μνήμη flash, συχνότητα έως 64 MHz, MPU, 80 DMIPS και οδηγίες DSP

- **Δείκτης απόδοσης**
  - 1.25 DMIPS/MHz (Drystone 2.1)
  - 219.48 CoreMark® (3.43 CoreMark/MHz στα 64 MHz)

- **Δείκτης ενέργειας**
  - 303 ULPMark™ CP score

- **Διαχείριση τροφοδοσίας και επαναφοράς**
  - Ενσωματωμένος μετατροπέας SMPS υψηλής απόδοσης με έξυπνη λειτουργία παράκαμψης
  - Εξαιρετικά ασφαλής, χαμηλής κατανάλωσης BOR (brownout reset) με πέντε επιλεγόμενα όρια
  - Εξαιρετικά χαμηλής κατανάλωσης POR/PDR
  - Προγραμματιζόμενος ανιχνευτής τάσης (PVD)
  - Λειτουργία VBAT με RTC και εφεδρικά μητρώα

- **Πηγές ρολογιού**
  - Κρυσταλλικός ταλαντωτής 32 MHz με ενσωματωμένους πυκνωτές ρύθμισης (ρολόι ραδιοφώνου και CPU)
  - Κρυσταλλικός ταλαντωτής 32 kHz για RTC (LSE)
  - Εσωτερικός χαμηλής κατανάλωσης ταλαντωτής 32 kHz (±5%) RC (LSI1)
  - Εσωτερικός χαμηλής κατανάλωσης ταλαντωτής 32 kHz (σταθερότητα ±500 ppm) RC (LSI2)
  - Εσωτερικός ταλαντωτής με δυνατότητα ρύθμισης ταχύτητας από 100 kHz έως 48 MHz, αυτόματης ρύθμισης από LSE (ακρίβεια καλύτερη από ±0.25%)
  - Εσωτερικός ταλαντωτής υψηλής ταχύτητας 16 MHz εργοστασιακής ρύθμισης RC (±1%)
  - 2x PLL για σύστημα ρολογιού, USB, SAI, ADC

- **Μνήμες**
  - Έως 1 MB μνήμη flash με προστασία τομέα (PCROP) κατά των λειτουργιών R/W, επιτρέποντας στοίβα ραδιοφώνου και εφαρμογή
  - Έως 256 KB SRAM, συμπεριλαμβανομένων 64 KB με hardware parity check
  - 20x 32-bit backup καταχωρητές
  - Boot loader που υποστηρίζει διεπαφές USART, SPI, I2C και USB
  - Ενημέρωση OTA (over the air) Bluetooth® Low Energy και 802.15.4
  - Διεπαφή μνήμης Quad SPI με XIP
  - 1 Kbyte (128 διπλές λέξεις) OTP

- **«Πλούσια» αναλογικά περιφερειακά (έως 1.62 V)**
  - 12-bit ADC 4.26 Msps, έως 16-bit με υπερδειγματοληψία υλικού, 200 µA/Msps
  - 2x συγκριτές εξαιρετικά χαμηλής κατανάλωσης
  - Έξοδος τάσης αναφοράς με μεγάλη ακρίβεια στα 2.5 V ή 2.048 V (buffered output).

- **Περιφερειακά συστήματος**
  - Inter processor communication controller (IPCC) για επικοινωνία με Bluetooth® Low Energy και 802.15.4
  - HW σεμαφόροι για κοινή χρήση πόρων μεταξύ των CPU
  - 2x ελεγκτές DMA (7x κανάλια ο καθένας) που υποστηρίζουν ADC, SPI, I2C,

USART, QSPI, SAI, AES, timers

- 1x USART (ISO 7816, IrDA, SPI Master, Modbus και λειτουργία Smartcard)
- 1x LPUART (χαμηλής κατανάλωσης)
- 2x SPI 32 Mbit/s
- 2x I2C (SMBus/PMBus®)
- 1x SAI (διπλού καναλιού υψηλής ποιότητας ήχου)
- 1x USB 2.0 FS συσκευή, χωρίς κρύσταλλο, BCD και LPM
- Ελεγκτής αφής, έως 18 αισθητήρες
- LCD 8x40 με μετατροπέα step-up
- 1x 16-bit, τετρακάναλος προηγμένος timer
- 2x 16-bit, δικάναλος timer
- 1x 32-bit, τετρακάναλος timer
- 2x 16-bit εξαιρετικά χαμηλής κατανάλωσης timer
- 1x ανεξάρτητο Systick
- 1x ανεξάρτητο watchdog
- 1x window watchdog

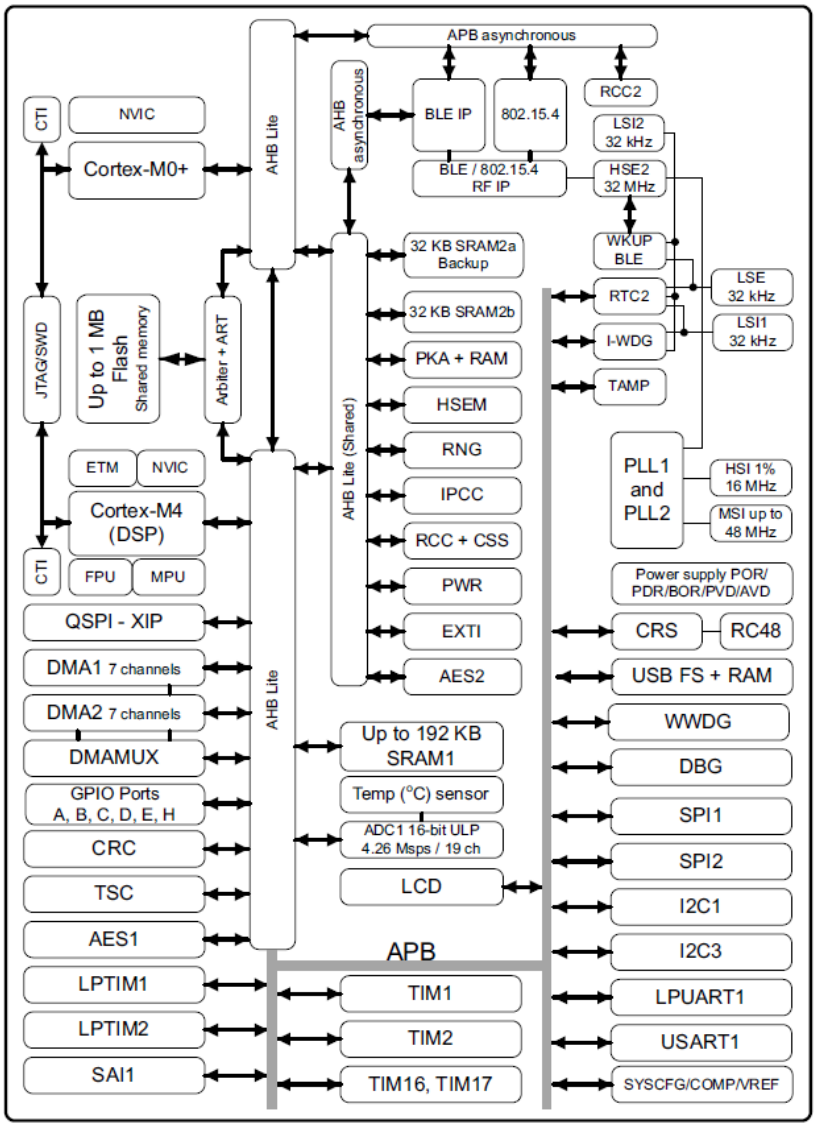
- **Ασφάλεια και Ταυτότητα**
  - Ασφαλής εγκατάσταση υλικολογισμικού (SFI) για το Bluetooth® Low Energy και το 802.15.4 SW stack
  - 3x υλικολογισμικό κρυπτογράφησης AES μέγιστο 256-bit για την εφαρμογή, το Bluetooth® Low Energy και το IEEE802.15.4
  - Υπηρεσίες αποθήκευσης/διαχείρισης κλειδιών πελάτη
  - HW public key authority (PKA)
  - Κρυπτογραφικοί αλγόριθμοι: RSA, Diffie-Helman, ECC over GF(p)
  - Γεννήτρια τυχαίων αριθμών (RNG) (RNG)
  - Τμηματική προστασία κατά των λειτουργιών R/W (PCROP)
  - Μονάδα υπολογισμού CRC
  - Πληροφορίες die: μοναδικό ID 96-bit
  - Μοναδικό ID IEEE 64-bit, δυνατότητα εξαγωγής 802.15.4 64-bit και Bluetooth® Low Energy 48-bit EUI

- **Έως 72 γρήγορες θύρες εισόδου/εξόδου, 70 από αυτές ανθεκτικές στα 5 V**

- **Υποστήριξη ανάπτυξης**
  - Σειριακή αποσφαλμάτωση (SWD), JTAG για τον επεξεργαστή εφαρμογών
  - Αλληλεπίδραση εφαρμογών μέσω εισόδου και εξόδου
  - Ενσωματωμένο Trace Macrocell™ για εφαρμογή

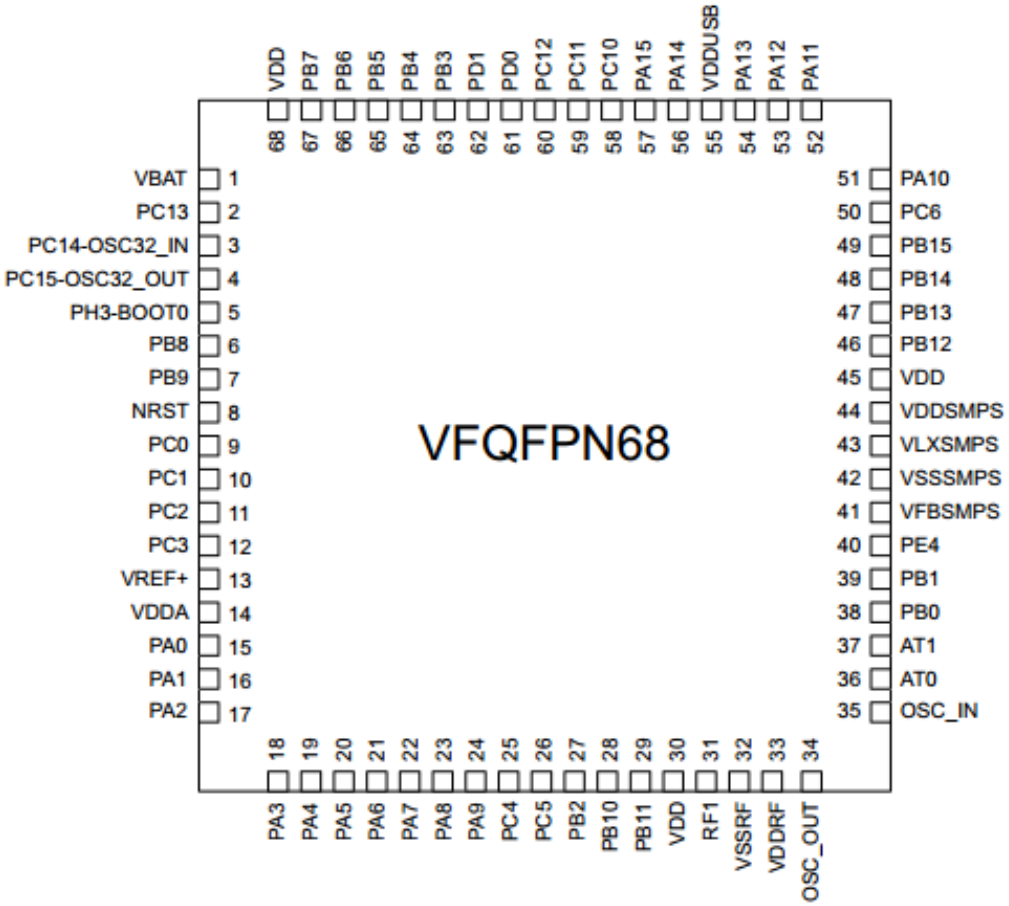
- **Συμβατές συσκευασίες ECOPACK2**

Block diagram



Εικόνα 7:STM32WB55RGV6 Block Diagram

Pinout



Εικόνα 8:VFQFPN68 package pinout

Βασικά Περιφερειακά

Analog to Digital Converted(ADC)

Η συσκευή ενσωματώνει έναν αναλογικό σε ψηφιακό μετατροπέα διαδοχικής προσέγγισης με τα εξής χαρακτηριστικά:

- Ανάλυση 12-bit (native), με ενσωματωμένη βαθμονόμηση
- Έως 16-bit ανάλυση με λόγο υπερδειγματοληψίας 256
- Μέγιστος ρυθμός μετατροπής 4.26 Msps με πλήρη ανάλυση
  - Χρόνος δειγματοληψίας έως 39 ns
  - Αυξημένος ρυθμός μετατροπής για χαμηλότερη ανάλυση (έως 7.11 Msps για ανάλυση 6-bit)
- Έως δεκαέξι εξωτερικά κανάλια και τρία εσωτερικά κανάλια: εσωτερικές τάσεις αναφοράς, αισθητήρας θερμοκρασίας

- Απλές και διαφορικές είσοδους
- Σχεδιασμός χαμηλής κατανάλωσης
  - Ικανότητα λειτουργίας με χαμηλό ρεύμα σε χαμηλό ρυθμό μετατροπής (η κατανάλωση μειώνεται γραμμικά με την ταχύτητα)
  - Αρχιτεκτονική διπλού χρονισμού: η ταχύτητα του ADC είναι ανεξάρτητη από τη συχνότητα της CPU
- Πολύ ευέλικτη ψηφιακή διεπαφή
  - Λειτουργία σάρωσης, συνεχής ή διακοπτόμενη: μπορούν να προγραμματιστούν δύο ομάδες μετατροπών αναλογικών σημάτων για να διαφοροποιηθούν οι μετατροπές φόντου και υψηλής προτεραιότητας σε πραγματικό χρόνο
  - Ο ADC υποστηρίζει πολλαπλές εισόδους ενεργοποίησης για συγχρονισμό με ενσωματωμένους χρονομετρητές και εξωτερικά σήματα
  - Τα αποτελέσματα αποθηκεύονται σε τρεις καταχωρητές δεδομένων ή στη SRAM με υποστήριξη ελεγκτή DMA
  - Προεπεξεργασία δεδομένων: ευθυγράμμιση αριστερά/δεξιά και αντιστάθμιση μετατόπισης ανά κανάλι
  - Ενσωματωμένη μονάδα υπερδειγματοληψίας για βελτιωμένο SNR
  - Προγραμματιζόμενος χρόνος δειγματοληψίας ανά κανάλι
  - Τρία αναλογικά watchdog για αυτόματη παρακολούθηση τάσης, δημιουργία διακοπών και ενεργοποίηση επιλεγμένων χρονομετρητών
  - Βοηθός υλικού για γρήγορη εναλλαγή μεταξύ απλών και Injected καναλιών.

Timers and Watchdogs

Οι STM32WB55xx περιλαμβάνουν έναν προηγμένο χρονομετρητή 16-bit, έναν γενικής χρήσης χρονομετρητή 32-bit, δύο βασικούς χρονομετρητές 16-bit, δύο χρονομετρητές χαμηλής κατανάλωσης, δύο χρονομετρητές watchdog και έναν χρονομετρητή SysTick. Ο παρακάτω πίνακας συγκρίνει τα χαρακτηριστικά των προηγμένων χρονομετρητών ελέγχου, γενικής χρήσης και χαμηλής κατανάλωσης.

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/ compare channels	Complementary outputs
Advanced control	TIM1	16-bits	Up, down, Up/down	Any integer between 1 and 65536	Yes	4	3
General purpose	TIM2	32-bits	Up, down, Up/down			4	No
General purpose	TIM16	16-bits	Up			2	1
General purpose	TIM17	16-bits	Up			2	1
Low power	LPTIM1 LPTIM2	16-bits	Up			1	1

Πίνακας 1:Χαρακτηριστικά των Timers του STM32WB55RGV6

Universal synchronous/asynchronous receiver transmitter (USART)

Οι συσκευές ενσωματώνουν έναν καθολικό σύγχρονο δέκτη-πομπό (USART). Αυτή η διεπαφή παρέχει ασύγχρονη επικοινωνία, υποστήριξη IrDA SIR ENDEC, λειτουργία επικοινωνίας πολλαπλών επεξεργαστών, λειτουργία επικοινωνίας μισής αμφίδρομης μονής γραμμής και έχει δυνατότητα LIN master/slave. Παρέχει υλική διαχείριση των σημάτων CTS και RTS, και ενεργοποίηση οδηγού RS485.

Η USART είναι ικανή να επικοινωνεί με ταχύτητες έως 4 Mbit/s, και παρέχει επίσης λειτουργία Smart Card (συμβατή με ISO 7816) και δυνατότητα επικοινωνίας τύπου SPI.

Η USART υποστηρίζει σύγχρονη λειτουργία (λειτουργία SPI), και μπορεί να χρησιμοποιηθεί ως SPI master.

Η USART έχει χρονικό τομέα ανεξάρτητο από το ρολόι της CPU, επιτρέποντάς της να ξυπνά τον MCU από τη λειτουργία Stop χρησιμοποιώντας baudrates έως 200 kbaud. Τα γεγονότα αφύπνισης από τη λειτουργία Stop είναι προγραμματιζόμενα και μπορεί να είναι:

- η ανίχνευση του bit εκκίνησης
- οποιοδήποτε ληφθέν πλαίσιο δεδομένων
- ένα συγκεκριμένο προγραμματισμένο πλαίσιο δεδομένων.

Η διεπαφή USART μπορεί να εξυπηρετηθεί από τον ελεγκτή DMA.

General-purpose inputs/outputs (GPIOs)

Κάθε ένα από τα GPIO pins μπορεί να διαμορφωθεί μέσω λογισμικού ως έξοδος (push-pull ή open-drain), ως είσοδος (με ή χωρίς pull-up ή pull-down) ή ως εναλλακτική λειτουργία περιφερειακού. Τα περισσότερες από τα GPIO pins μοιράζονται με ψηφιακές ή αναλογικές εναλλακτικές λειτουργίες. Η γρήγορη εναλλαγή I/O μπορεί να επιτευχθεί χάρη στη χαρτογράφησή τους στο δίαυλο AHB2.

Η διαμόρφωση της εναλλακτικής λειτουργίας των I/Os μπορεί να κλειδωθεί, εάν χρειαστεί, ακολουθώντας μια συγκεκριμένη ακολουθία για να αποφευχθεί η τυχαία εγγραφή στα μητρώα των I/Os.

Direct memory access controller (DMA)

Η συσκευή ενσωματώνει δύο DMAs. Η άμεση πρόσβαση στη μνήμη (DMA) χρησιμοποιείται για την παροχή υψηλής ταχύτητας μεταφοράς δεδομένων μεταξύ περιφερειακών και μνήμης καθώς και μεταξύ μνημών. Τα δεδομένα μπορούν να μετακινηθούν γρήγορα από το DMA χωρίς καμία ενέργεια της CPU. Αυτό διατηρεί τους πόρους της CPU ελεύθερους για άλλες λειτουργίες.

Οι δύο ελεγκτές DMA έχουν συνολικά δεκατέσσερα κανάλια, ένας πλήρης συσχετιζόμενος πίνακας επιτρέπει σε οποιοδήποτε περιφερειακό να αντιστοιχιστεί σε οποιοδήποτε από τα διαθέσιμα κανάλια DMA. Κάθε DMA έχει έναν διαχειριστή για τον χειρισμό της προτεραιότητας μεταξύ των αιτημάτων DMA.

Η DMA υποστηρίζει:

- δεκατέσσερα ανεξάρτητα διαμορφώσιμα κανάλια (αιτήματα)
- Έναν πλήρη συσχετιζόμενο πίνακα μεταξύ περιφερειακών και όλων των καναλιών DMA υπάρχει. Υπάρχει επίσης δυνατότητα ενεργοποίησης υλικού μέσω του DMAMUX. •
- Οι προτεραιότητες μεταξύ των αιτημάτων από τα κανάλια DMA είναι προγραμματιζόμενες μέσω λογισμικού (τέσσερα επίπεδα που αποτελούνται από πολύ υψηλή, υψηλή, μεσαία και χαμηλή) ή υλικού σε περίπτωση ισότητας (το αίτημα 1 έχει προτεραιότητα έναντι του αιτήματος 2, κ.λπ.).
- Ανεξάρτητο μέγεθος μεταφοράς πηγής και προορισμού (byte, half word, word), προσομοιώνοντας το πακετάρισμα και ξεπακετάρισμα. Οι διευθύνσεις πηγής/προορισμού πρέπει να είναι ευθυγραμμισμένες με το μέγεθος των δεδομένων.

- Υποστήριξη για διαχείριση κυκλικού buffer.
- Τρεις σημαίες γεγονότων (DMA half transfer, DMA transfer complete και DMA transfer error) λογικά OR-ed μαζί σε ένα μόνο αίτημα διακοπής για κάθε κανάλι.
- Μεταφορά από μνήμη σε μνήμη.
- Μεταφορές από περιφερειακό σε μνήμη και από μνήμη σε περιφερειακό, και από περιφερειακό σε περιφερειακό.
- Πρόσβαση στη μνήμη flash, SRAM, περιφερειακά APB και AHB ως πηγή και προορισμό.
- Προγραμματιζόμενος αριθμός δεδομένων προς μεταφορά: έως 65536.

2.4 Λογισμικό STM32CubeIDE

Περιγραφή[11]

Το STM32CubeIDE είναι ένα εργαλείο ανάπτυξης πολλαπλών λειτουργικών συστημάτων, το οποίο αποτελεί μέρος του οικοσυστήματος λογισμικού STM32Cube. Είναι μια προηγμένη πλατφόρμα ανάπτυξης C/C++ με διαμόρφωση περιφερειακών, δημιουργία κώδικα, μεταγλώττιση κώδικα, και δυνατότητες αποσφαλμάτωσης για μικροελεγκτές και μικροεπεξεργαστές STM32. Βασίζεται στο πλαίσιο Eclipse®/CDT™ και το εργαλείο GCC για την ανάπτυξη, και το GDB για την αποσφαλμάτωση. Επιτρέπει την ενσωμάτωση εκατοντάδων υπαρχόντων plugins που ολοκληρώνουν τις δυνατότητες του Eclipse® IDE.

Το STM32CubeIDE ενσωματώνει τις λειτουργίες διαμόρφωσης και δημιουργίας έργων STM32 από το STM32CubeMX για να προσφέρει μια ολοκληρωμένη εμπειρία εργαλείου και να εξοικονομήσει χρόνο εγκατάστασης και ανάπτυξης. Αφού επιλεγεί ένας «κενός» STM32 MCU ή MPU, ή ένας προδιαμορφωμένος μικροελεγκτής ή μικροεπεξεργαστής από κάποια συγκεκριμένη πλακέτα ή ένα συγκεκριμένο παράδειγμα, το project δημιουργείται και μαζί και ο αρχικός κώδικας. Ανά πάσα στιγμή κατά την ανάπτυξη, ο χρήστης μπορεί να επιστρέψει στην αρχικοποίηση και διαμόρφωση των περιφερειακών ή του middleware και να αναδημιουργήσει τον αρχικό κώδικα χωρίς καμία επίπτωση στον κώδικα του χρήστη.

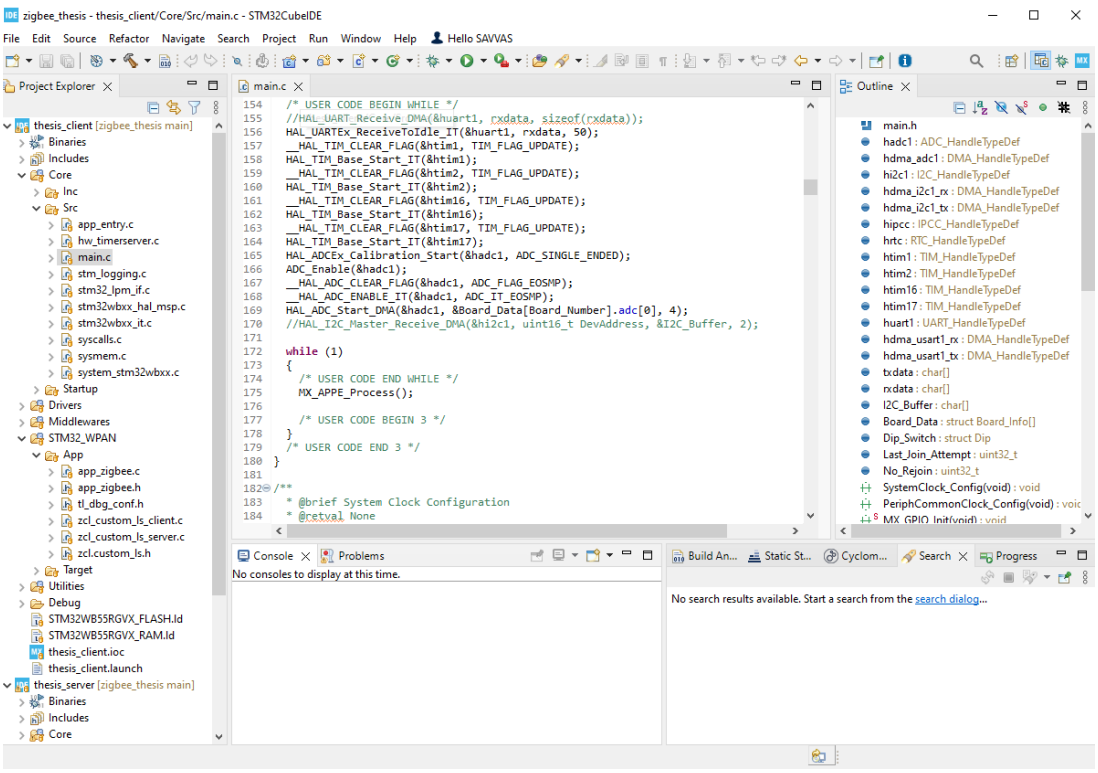
Το STM32CubeIDE περιλαμβάνει αναλυτές κατασκευής και στοίβας που παρέχουν στον χρήστη χρήσιμες πληροφορίες για την κατάσταση του έργου και τις

απαιτήσεις μνήμης. Επίσης παρέχει τυπικές και προηγμένες δυνατότητες αποσφαλμάτωσης, συμπεριλαμβανομένων προβολών των καταχωρητών του πυρήνα της CPU, των μνημών, και των καταχωρητών περιφερειακών, καθώς και παρακολούθηση ζωντανών μεταβλητών, διεπαφή Serial Wire Viewer, ή αναλυτή σφαλμάτων.

Χαρακτηριστικά[11]

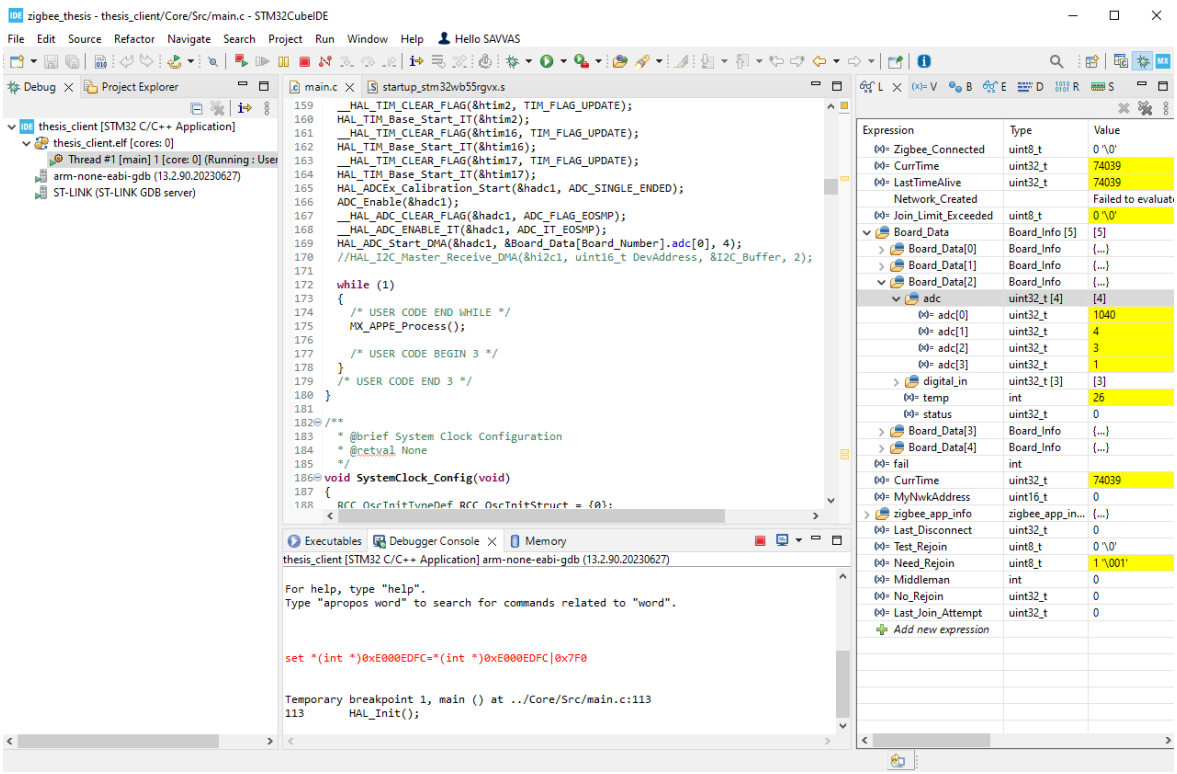
- Ενσωμάτωση υπηρεσιών από το STM32CubeMX:
  - Επιλογή μικροελεγκτή STM32, μικροεπεξεργαστή, πλατφόρμας ανάπτυξης και παραδειγματικού έργου
  - Διαμόρφωση εξόδων ακίδων, ρολογιού, περιφερειακών και middleware
  - Δημιουργία έργου και παραγωγή κώδικα αρχικοποίησης
  - Ολοκληρωμένο λογισμικό και middleware με αναβαθμισμένα STM32Cube Expansion Packages
- Βασισμένο στο Eclipse®/CDT™, με υποστήριξη για πρόσθετα του Eclipse®, GNU C/C++ για το εργαλείο Arm® και αποσφαλματωτή GDB
- Σειρά STM32MP1:
  - Υποστήριξη για έργα OpenSTLinux: Linux®, U-Boot, TF-A και OP-TEE, συμπεριλαμβανομένου του Device Tree από το STM32CubeMX
  - Υποστήριξη για εφαρμογή Linux® User Space, κοινή ή στατική βιβλιοθήκη
- Πρόσθετες προηγμένες λειτουργίες αποσφαλμάτωσης, συμπεριλαμβανομένων:
  - Προβολές πυρήνα CPU, καταχωρητών περιφερειακών και μνήμης
  - Προβολή μεταβλητών σε πραγματικό χρόνο
  - Ανάλυση συστήματος και ανίχνευση σε πραγματικό χρόνο (SWV)
  - Εργαλείο ανάλυσης σφαλμάτων CPU
  - Υποστήριξη αποσφαλμάτωσης με RTOS, συμπεριλαμβανομένων των Azure® RTOS ThreadX και FreeRTOS™ kernel
- Υποστήριξη για τις διεπαφές αποσφαλμάτωσης ST-LINK (STMicroelectronics) και J-Link (SEGGER)
- Εισαγωγή έργου από το Atollic® TrueSTUDIO® και το AC6 System Workbench for STM32 (SW4STM32)
- Υποστήριξη πολλαπλών λειτουργικών συστημάτων: Windows®, Linux® και macOS®, μόνο εκδόσεις 64-bit

C/C++ Developer View



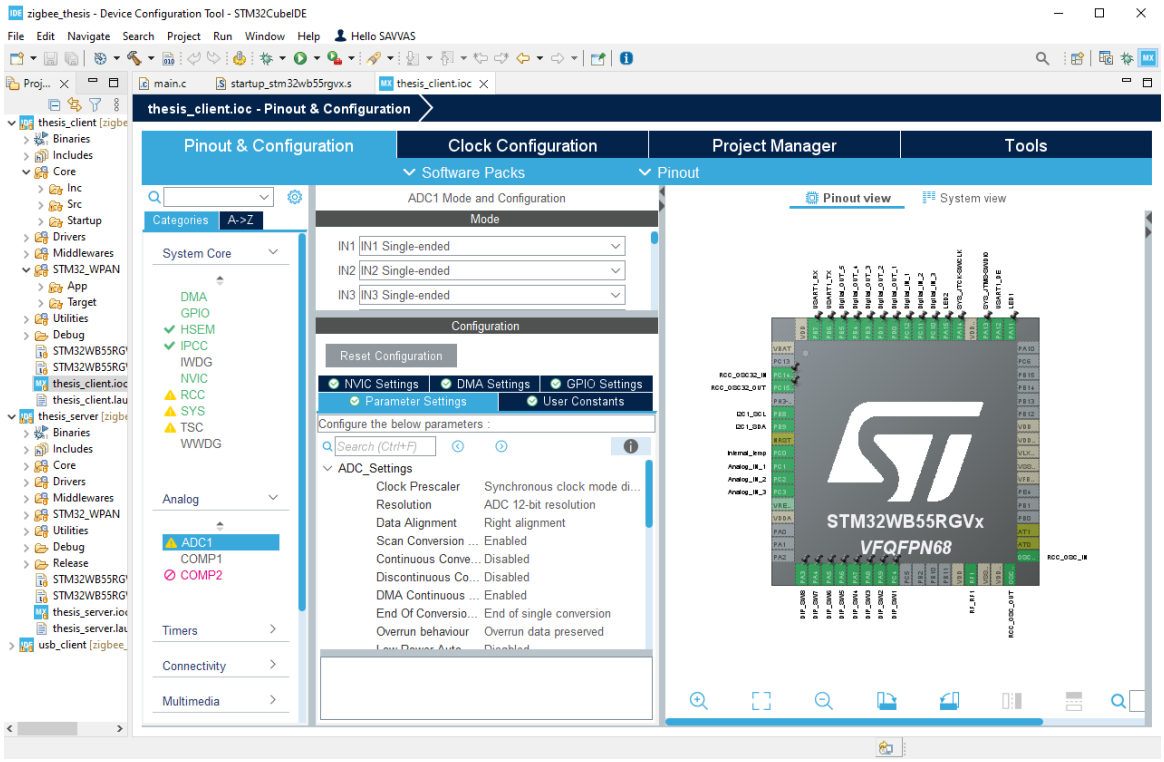
Εικόνα 9:C/C++ Οπτική του Λογισμικού STM32CubeIDE

Debug Viewer



Εικόνα 10:Οπτική Αποσφαλμάτωσης του λογισμικού STM32CubeIDE

Cube MX Tool



Εικόνα 11:Λογισμικό STM32CubeMX

2.5 Ενεργειακή Διαχείριση Κτηρίου και IoT

Σε αυτήν την ενότητα παρουσιάζονται τεχνικές που χρησιμοποιούνται σε Συστήματα Διαχείρισης Ενέργειας Κτηρίων (Building Energy Management System – BEMS) και το Διαδίκτυο των Πραγμάτων (IoT). Τα κτήρια γενικά τροφοδοτούνται με ηλεκτρική ενέργεια, η οποία έχει καταστεί αναπόσπαστο μέρος της σύγχρονης ζωής, επιτελώντας λειτουργίες όπως θέρμανση, ψύξη, φωτισμός και λειτουργία των συσκευών και των υπολογιστών των χρηστών του κτηρίου. Η διαχείριση της ενέργειας στα κτήρια έχει, επομένως, καταστεί ένα απαιτητικό ζήτημα, καθώς η συνολική κατανάλωση ενέργειας και η κατανάλωση ενέργειας των κτηρίων έχουν αυξηθεί σημαντικά τις τελευταίες δεκαετίες. Σε αυτό το πλαίσιο, θα περιγράψουμε ορισμένες μεθόδους διαχείρισης και μείωσης της κατανάλωσης ενέργειας στα κτήρια χρησιμοποιώντας το Διαδίκτυο των Πραγμάτων (IoT). [12]

Οι αισθητήρες IoT συμβάλλουν παρέχοντας λύσεις για τη διαχείριση της κατανάλωσης ενέργειας και ανιχνεύοντας ενέργειες που οδηγούν σε χαμηλότερη κατανάλωση ενέργειας και μείωση κόστους. Οι τεχνολογίες IoT υποστηρίζουν τη

χρήση αισθητήρων για την παρακολούθηση της κατανάλωσης ενέργειας ενός κτηρίου. Επίσης, συμβάλλουν στην αναγνώριση των κλιματικών συνθηκών και στην προσαρμογή των συστημάτων ρύθμισης του κλίματος του κτηρίου ανάλογα, ώστε να καταναλώνεται η ελάχιστη ποσότητα ηλεκτρικής ενέργειας. Η οικονομία που συνδέεται με το IoT αναπτύσσεται σε όλους τους τομείς λόγω της εύκολης προσβασιμότητας των χαμηλού κόστους αισθητήρων που ενισχύουν την αποδοτικότητα των συστημάτων διαχείρισης ενέργειας των κτηρίων παγκοσμίως. [13]

Στο [14] εξετάζεται η εφαρμογή του IoT στα ενεργειακά συστήματα, ειδικά στα έξυπνα δίκτυα. Αναλύουν τα πλεονεκτήματα και τα μειονεκτήματα των τεχνολογιών IoT, συμπεριλαμβανομένων του cloud computing και των διαφόρων πλατφορμών για ανάλυση δεδομένων. Ανασκοπούν τις προκλήσεις της ανάπτυξης του IoT στον ενεργειακό τομέα, συμπεριλαμβανομένων θεμάτων ιδιωτικότητας και ασφάλειας, και προτείνουν κάποιες λύσεις σε αυτές τις προκλήσεις. Αυτή η έρευνα ήταν χρήσιμη για τους ερευνητές στον τομέα, καθώς τους επέτρεψε να αποκτήσουν μια ιδέα για τη χρήση εργαλείων cloud computing, ανάλυσης δεδομένων και οπτικοποίησης σε διάφορους τομείς.

Το cloud computing και το Διαδίκτυο των Πραγμάτων (IoT) είναι δύο πολύ διαφορετικές τεχνολογίες που ήδη αποτελούν μέρος της ζωής μας. Στο [15] επικεντρώνονται στην ενσωμάτωση του Cloud και του IoT σε ένα παράδειγμα που ονομάζεται CloudIoT. Συζητούν τις κύριες ιδιότητες, τα χαρακτηριστικά, τις υποκείμενες τεχνολογίες και τα ζητήματα του παραδείγματος. Το άρθρο επιτρέπει στους ερευνητές να δουν και να κατανοήσουν νέα σενάρια εφαρμογής, καθώς και τις κύριες προκλήσεις που σχετίζονται με το καθένα από αυτά.

Ένα έξυπνο κτήριο είναι εξοπλισμένο με σύγχρονες τεχνολογίες για τον αυτοματισμό διαδικασιών, όπως ο έλεγχος φωτισμού, θέρμανσης, αερισμού και κλιματισμού (HVAC), και την τοπική χρήση ενέργειας μέσω έξυπνων τεχνολογιών [16].

Ένα έξυπνο δίκτυο αναφέρεται στην ενσωμάτωση προηγμένων ηλεκτρικών πληροφοριών σε ένα σύστημα που έχει τη δυνατότητα επικοινωνίας με τον χρήστη και τον πάροχο [17].

Ένα σύστημα διαχείρισης ενέργειας κατοικίας (HEMS) ορίζεται ως το βέλτιστο σύστημα που παρέχει υπηρεσίες διαχείρισης ενέργειας για την αποτελεσματική

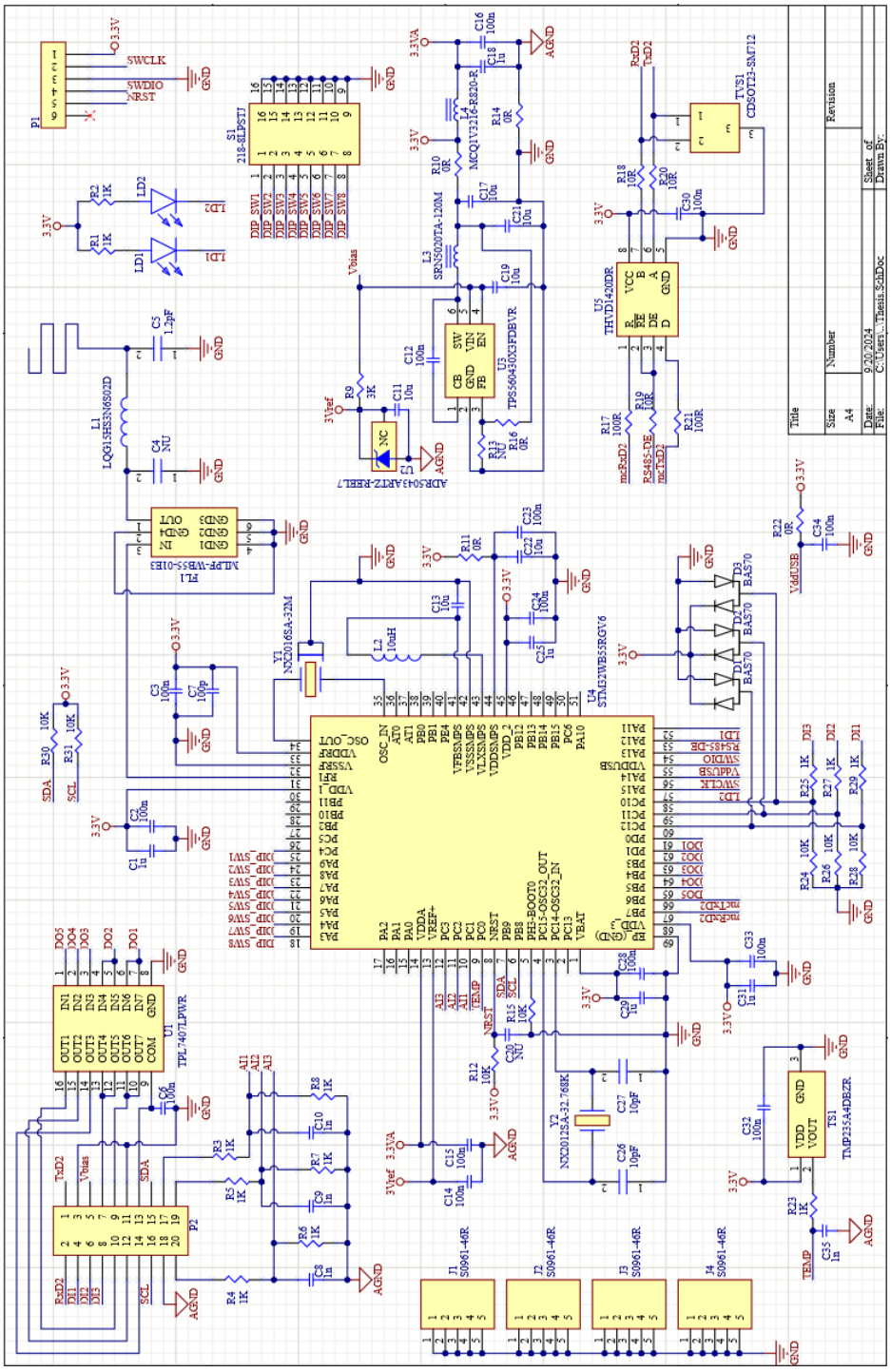
παρακολούθηση και διαχείριση της παραγωγής, αποθήκευσης και κατανάλωσης ηλεκτρικής ενέργειας σε έξυπνα σπίτια [18].

Ένα σύστημα διαχείρισης ενέργειας κτηρίων (BEMS) περιλαμβάνει πολλά διαφορετικά πρωτόκολλα και εφαρμογές, καθιστώντας το ανεξάρτητο από το πρωτόκολλο της συσκευής [19].

Στο [20] παρουσιάζεται μια Συστηματική Βιβλιογραφική Ανασκόπηση (Systematic Literature Review – SLR) με τη μέθοδο Kitchenham and Charters για τις διάφορες μεθόδους που χρησιμοποιούν το Διαδίκτυο των Πραγμάτων (IoT) για τη μείωση της κατανάλωσης ενέργειας των κτηρίων, συγκρίνοντας και αναλύοντας τα πλεονεκτήματα και τα μειονεκτήματα των υφιστάμενων μεθόδων και των προτεινόμενων σχεδίων.

3 Σχεδιασμός πλακέτας – Hardware

3.1 Σχηματικό διάγραμμα

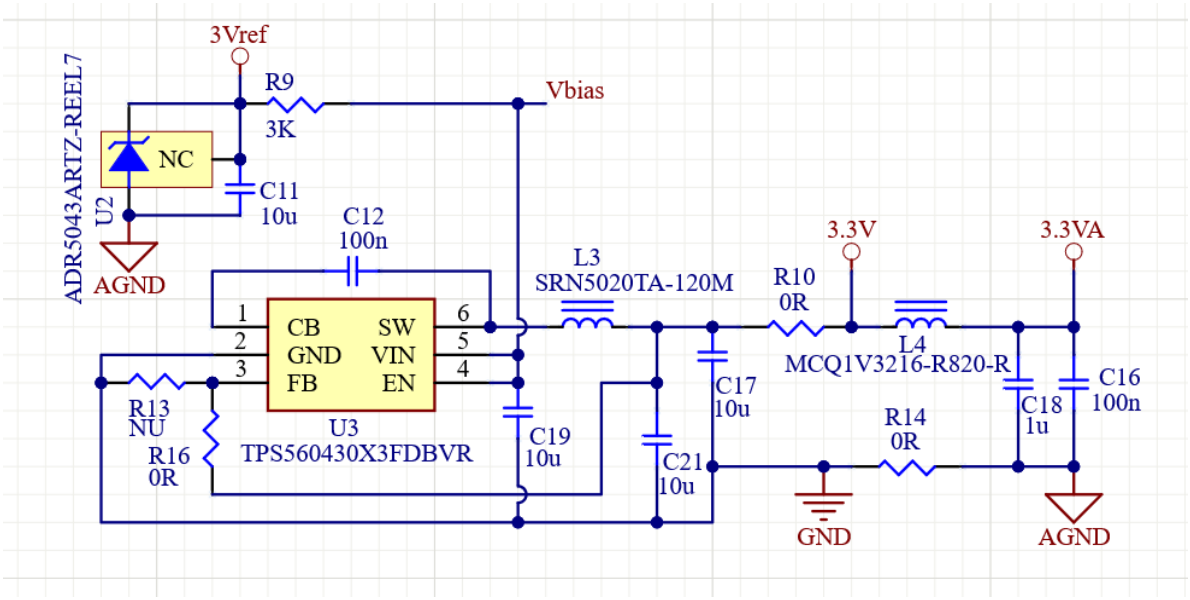


Εικόνα 12:Σχηματικό Διάγραμμα Συστήματος

3.2 Περιγραφή – Λειτουργικότητα

Τροφοδοσία και Κρύσταλλοι

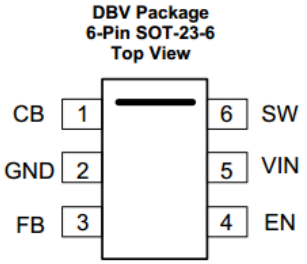
Το κύκλωμα της τροφοδοσίας τόσο του μικροελεγκτή όσο και των υπόλοιπων περιφερειακών κυκλωμάτων είναι το εξής:



Εικόνα 13:Κύκλωμα Τροφοδοσίας

Το βασικό ολοκληρωμένο που ρυθμίζει την τροφοδοσία είναι το TPS560430 της Texas Instruments.

Το TPS560430 είναι ένας εύχρηστος συγχρονος μετατροπέας DC-DC step-down, ικανός να παρέχει ρεύμα φορτίου έως 600 mA. Με ένα ευρύ φάσμα εισόδου από 4 V έως 36 V, η συσκευή είναι κατάλληλη για μια ποικιλία εφαρμογών, από βιομηχανικές έως αυτοκινητιστικές, για την τροφοδοσία από μια μη ρυθμιζόμενη πηγή. Το TPS560430 διαθέτει εκδόσεις λειτουργικής συχνότητας 1.1 MHz και 2.1 MHz για είτε υψηλή απόδοση είτε μικρό μέγεθος λύσης. Διαθέτει επίσης έκδοση FPWM (forced PWM) για επίτευξη σταθερής συχνότητας και μικρού κυματισμού τάσης εξόδου σε όλο το εύρος φορτίου. Τα κυκλώματα soft-start και αντιστάθμισης έχουν ενσωματωθεί εσωτερικά, επιτρέποντας τη χρήση της συσκευής με ελάχιστα εξωτερικά εξαρτήματα. Η συσκευή διαθέτει ενσωματωμένες δυνατότητες προστασίας, όπως περιορισμό ρεύματος ανά κύκλο, προστασία βραχυκυκλώματος με λειτουργία hiccup και θερμική απενεργοποίηση σε περίπτωση υπερβολικής διασποράς ισχύος.[21] Το Pin Configuration του ολοκληρωμένου παρουσιάζεται στον πίνακα:



Pin Functions

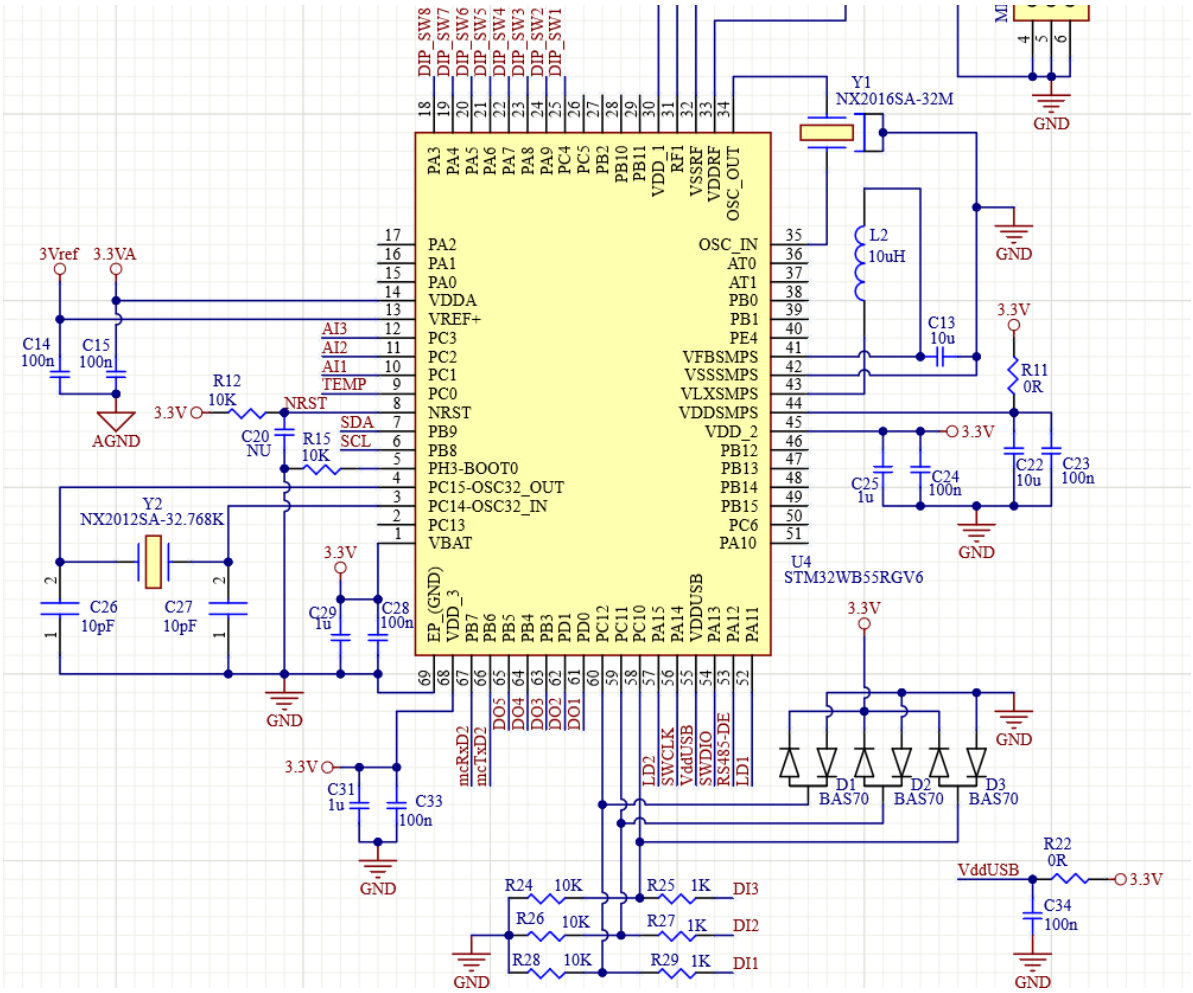
PIN		TYPE (1)	DESCRIPTION
NO.	NAME		
1	CB	P	Bootstrap capacitor connection for high-side FET driver. Connect a high quality 100-nF capacitor from this pin to the SW pin.
2	GND	G	Power ground terminals, connected to the source of low-side FET internally. Connect to system ground, ground side of C_{IN} and C_{OUT}. Path to C_{IN} must be as short as possible.
3	FB	A	Feedback input to the convertor. Connect a resistor divider to set the output voltage. Never short this terminal to ground during operation.
4	EN	A	Precision enable input to the convertor. Do not float. High = on, Low = off. Can be tied to VIN. Precision enable input allows adjustable UVLO by external resistor divider.
5	VIN	P	Supply input terminal to internal bias LDO and high-side FET. Connect to input supply and input bypass capacitors C_{IN}. Input bypass capacitors must be directly connected to this pin and GND.
6	SW	P	Switching output of the convertor. Internally connected to source of the high-side FET and drain of the low-side FET. Connect to power inductor.

(1) A = Analog, P = Power, G = Ground.

Πίνακας 2:Pin Configuration του ολοκληρωμένου TPS560430

Εκτός από τον Step Down converter που δίνει τροφοδοσία τόσο στον μικροελεγκτή όσο και στα περιφερειακά κυκλώματα, υπάρχει και ένα ολοκληρωμένο που παρέχει την τάση αναφοράς για τον adc του μικροελεγκτή. Σχεδιασμένο για εφαρμογές με περιορισμένο χώρο, το ADR5043 είναι εξαιρετικά ακριβής shunt voltage reference (αναφορά τάσης), τοποθετημένο σε εξαιρετικά μικρό πακέτο SC70 και SOT-23. Αυτή η αναφορά τάσης είναι πολυχρηστική και εύχρηστη, κατάλληλη για πληθώρα εφαρμογών. Διαθέτει χαμηλή θερμική απόκλιση, αρχική ακρίβεια καλύτερη από 0,1% και γρήγορο χρόνο σταθεροποίησης. Η τάση εξόδου του συγκεκριμένου ολοκληρωμένου είναι 3V Το ελάχιστο ρεύμα λειτουργίας κυμαίνεται από 50 μA σε μέγιστο 15 mA. Αυτό το χαμηλό ρεύμα λειτουργίας και η ευκολία χρήσης καθιστούν αυτή την αναφορά ιδανική για φορητές συσκευές που λειτουργούν με μπαταρία. Έχει διευρυμένο θερμοκρασιακό εύρος από −40°C έως +125°C.[22]

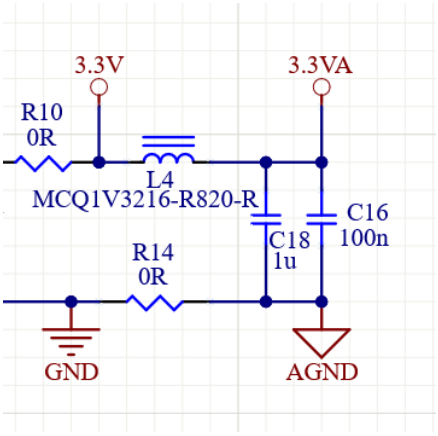
Η σύνδεση της τροφοδοσίας καθώς και των κρυστάλλων στον μικροελεγκτή είναι η εξής:



Εικόνα 14:Διασύνδεση περιφερειακών κυκλωμάτων με τον μικροελεγκτή

Οι δύο κρύσταλλοι που συνδέονται με τον μικροελεγκτή είναι ο NX2012SA-32.768K που είναι χρονισμένος στα 32.768kHz και ο NX2016SA-32M που είναι χρονισμένος στα 32MHz. Οι κρύσταλλοι συνεισφέρουν στην ρύθμιση και την λειτουργία των ρολογιών του μικροελεγκτή. Η ρύθμιση των ρολογιών παρουσιάζεται στην ενότητα *4.1 Διαμόρφωση του Μικροελεγκτή* και συγκεκριμένα στην υποενότητα *Διαμόρφωση των Ρολογιών*.

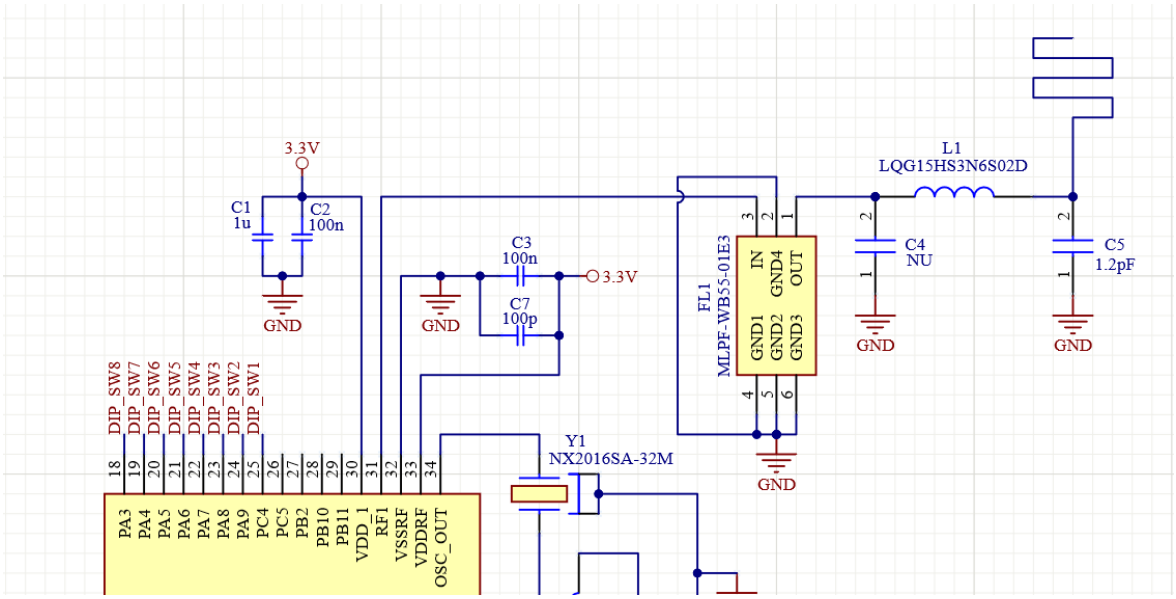
Οι τάσεις τροφοδοσίας παρέχονται απευθείας από το κύκλωμα με τον step-down μετατροπέα που αναλύθηκε παραπάνω, σημειώνοντας πως στον ελεγκτή παρέχεται και μια αναλογική τάση τροφοδοσίας, η οποία είναι η τάση τροφοδοσίας 3.3V που τροφοδοτεί τα περισσότερα κυκλώματα αλλά επιπλέον φιλτραρισμένη όπως φαίνεται στην παρακάτω εικόνα:



Εικόνα 15:Τάση τροφοδοσίας αναλογικών κυκλωμάτων

Κύκλωμα RF

Το κύκλωμα RF που συνδέεται με το RF pin του μικροελεκτή είναι το εξής:



Εικόνα 16:Κύκλωμα RF του συστήματος

Το pin του μικροελεκτή που σχετίζεται με την RF επικοινωνία συνδέεται απευθείας με το ολοκληρωμένο MLPF-WB55-01e3. Το MLPF-WB55-01E3 ενσωματώνει ένα δίκτυο προσαρμογής σύνθετης αντίστασης και φίλτρο αρμονικών. Το δίκτυο προσαρμογής της σύνθετης αντίστασης έχει σχεδιαστεί ώστε να μεγιστοποιεί την απόδοση RF του STM32WB. Αυτή η συσκευή χρησιμοποιεί την τεχνολογία IPD της STMicroelectronics σε μη αγώγιμο γυάλινο υπόστρωμα, η οποία βελτιστοποιεί την απόδοση RF. Είναι κατάλληλο για εφαρμογές Zigbee καθώς και Bluetooth και είναι βελτιστοποιημένο για χρήση σε σύνδεση με τον μικροελεγκτή που ενσωματώνει η συγκεκριμένη πλακέτα.[23]

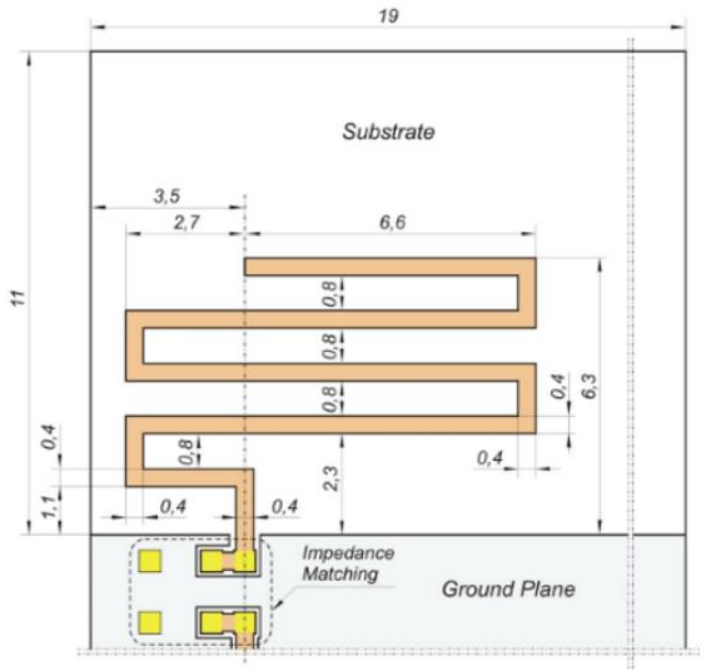
Η έξοδος του ολοκληρωμένου που είναι υπεύθυνο για την προσαρμογής της σύνθετης αντίστασης συνδέεται με ένα LC φίλτρο. Σκοπός του φίλτρου αυτού είναι τόσο η καταστολή των αρμονικών που παράγονται από τα ψηφιακά κυκλώματα και τα κυκλώματα RF όσο και την βελτίωση της συμφωνίας της σύνθετης αντίστασης μεταξύ της εξόδου του ολοκληρωμένου και της κεραίας. Με το φίλτρο αυτό σύμφωνα και με τους προηγούμενους λόγους ύπαρξης του βελτιώνεται η μετάδοση ισχύος του κυκλώματος και μειώνονται οι απώλειες αντανάκλασης. Η έξοδος του LC φίλτρου είναι συνδεδεμένη με μια PCB κεραία.

PCB κεραία[24]

Οι PCB κεραίες (Printed Circuit Board antennas) είναι κεραίες που κατασκευάζονται απευθείας πάνω στο τυπωμένο κύκλωμα (PCB) μιας συσκευής. Αυτές οι κεραίες είναι ιδιαίτερα δημοφιλείς σε εφαρμογές όπως οι συσκευές ασύρματης επικοινωνίας, τα κινητά τηλέφωνα και τα IoT συστήματα, λόγω του χαμηλού κόστους κατασκευής και της μικρής τους διάστασης. Η συγκεκριμένη κεραία είναι τύπου Meander Line Antenna. Τέτοιου τύπου κεραίες χρησιμοποιούνται ευρέως σε ασύρματες εφαρμογές, όπως Bluetooth, Wi-Fi, και Zigbee.

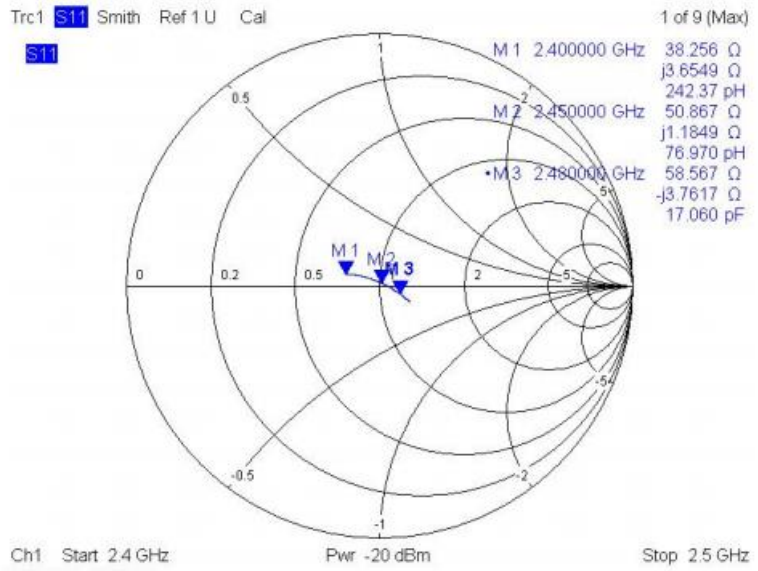
Η κεραία που χρησιμοποιείται βασίζεται στο μοντέλο που παρουσιάζεται στο [25]. Το μήκος της γραμμής meander καθώς και το σχήμα είναι βελτιστοποιημένα ώστε να επιτυγχάνουν καλή προσαρμογή σύνθετης αντίστασης γύρω από την συχνότητα των 2.4Ghz [26].

Οι κεραίες PCB τύπου meander line μπορούν να ρυθμιστούν στην απαιτούμενη αντίσταση των 50 Ω μέσω της προσαρμογής της κυκλωματικής αντίστασης με την τοπολογία π. Στην εικόνα 17, η περιοχή προσαρμογής της αντίστασης επισημαίνεται με διακεκομμένη γραμμή. Υπό ονομαστικές συνθήκες, αυτή η κεραία παρουσιάζει αντίσταση πολύ κοντά στην απαιτούμενη ονομαστική αντίσταση (50 Ω).



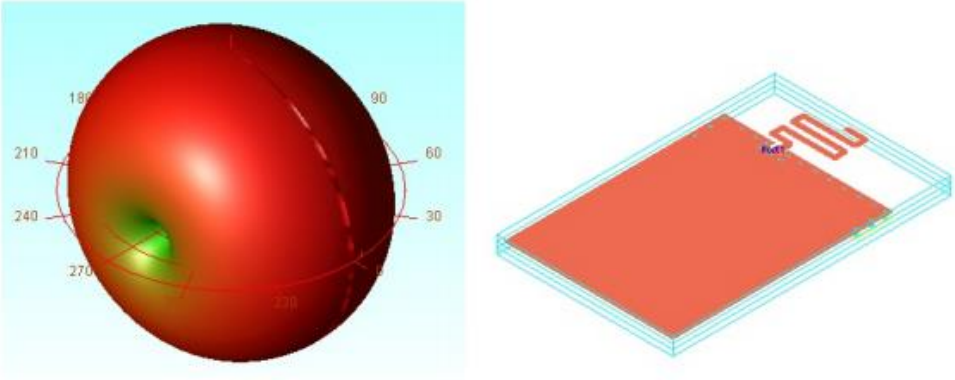
Εικόνα 17:Διαστάσεις της κεραίας (σε mm)

Το παρακάτω διάγραμμα Smith δείχνει τις μετρηθείσες τιμές της σύνθετης αντίστασης της κεραίας (παράμετρος σκέδασης S_{11}) σε τρεις βασικές συχνότητες: 2.4GHz (M1), 2.45GHz (M2) και 2.48GHz (M3).[24] Οι μετρήσεις αυτές πραγματοποιήθηκαν για να αξιολογηθεί η προσαρμογή σύνθετης αντίστασης της κεραίας στην περιοχή συχνοτήτων ISM των 2.4GHz, όπου η προσαρμογή της αντίστασης είναι κρίσιμη για τη βέλτιστη μετάδοση και λήψη σήματος. Παρατηρούμε ότι η συχνότητα που επιτυγχάνει την πλησιέστερη στα 50Ω σύνθετη αντίσταση είναι τα 2.45GHz



Εικόνα 18:Σύνθετη αντίστασης της κεραίας (Smith Chart)

Μια τρισδιάστατη (3-D) απεικόνιση του διαγράμματος ακτινοβολίας (radiation pattern), δηλαδή του μέτρου του ηλεκτρικού πεδίου |E| για την κεντρική συχνότητα της ISM ζώνης, 2.44175 GHz, είναι η εξής:



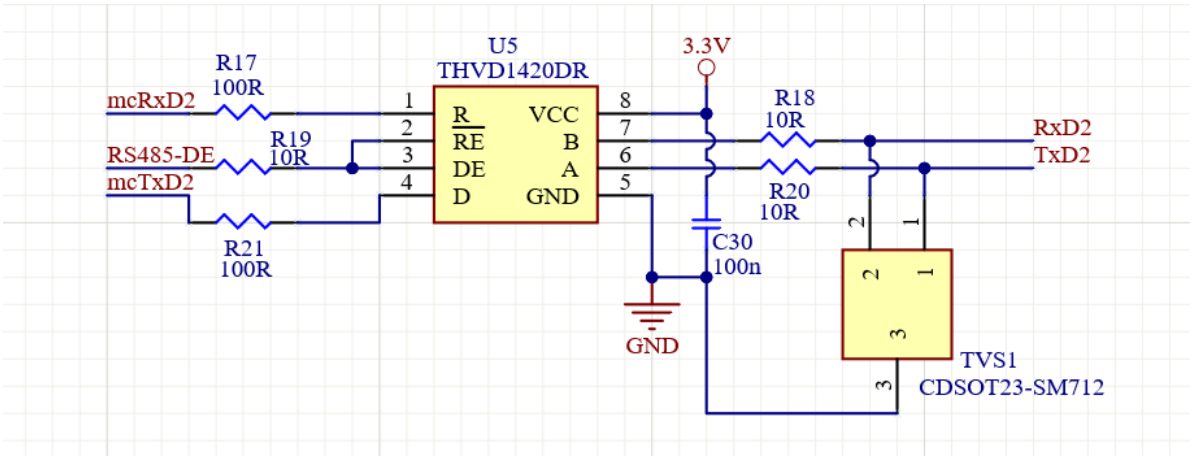
Εικόνα 19:Τρισδιάστατο διάγραμμα ακτινοβολίας

Οι παρακάτω αλλαγές επηρεάζουν την ακτινοβολούμενη ισχύ της κεραίας PCB:

- Μεταβολή του μεγέθους της πλακέτας
- Μεταλλική θωράκιση
- Χρήση πλαστικού καλύμματος
- Παρουσία άλλων εξαρτημάτων κοντά στην κεραία

Κύκλωμα μετατροπής UART σε RS-485

Το κύκλωμα μετατροπής της πληροφορίας που βγάζει η έξοδος της UART από τον μικροελεκγτή σε πρωτόκολλο RS-485 είναι το εξής:



Εικόνα 20:Κύκλωμα μετατροπής UART σε RS-485

Το βασικό ολοκληρωμένο του συγκεκριμένου κυκλώματος είναι το THVD1420DR της Texas Instruments. Το THVD1420 είναι ανθεκτικός πομποδέκτης

RS-485 σε ημί-αμφίδρομη λειτουργία για βιομηχανικές εφαρμογές. Η συσκευή λειτουργεί με τροφοδοσία από 3 έως 5,5 V. Το ευρύ εύρος κοινής τάσης (common-mode) και η χαμηλή διαρροή εισόδου στις ακίδες του διαύλου καθιστούν τις συσκευές κατάλληλες για πολυσημειακές εφαρμογές σε μακριά καλώδια. Το THVD1420 διατίθεται σε τυποποιημένη βιομηχανική συσκευασία SOIC 8 ακίδων για άμεση συμβατότητα, καθώς και σε μικρή συσκευασία SOT, η οποία πρωτοπορεί στον κλάδο. Οι συσκευές έχουν χαρακτηριστεί για λειτουργία σε θερμοκρασίες περιβάλλοντος από -40°C έως 125°C. Το Pin Configuration του ολοκληρωμένου είναι το εξής:

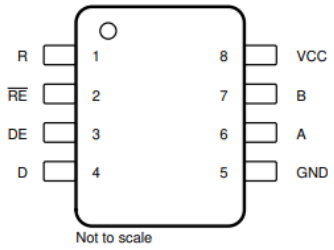


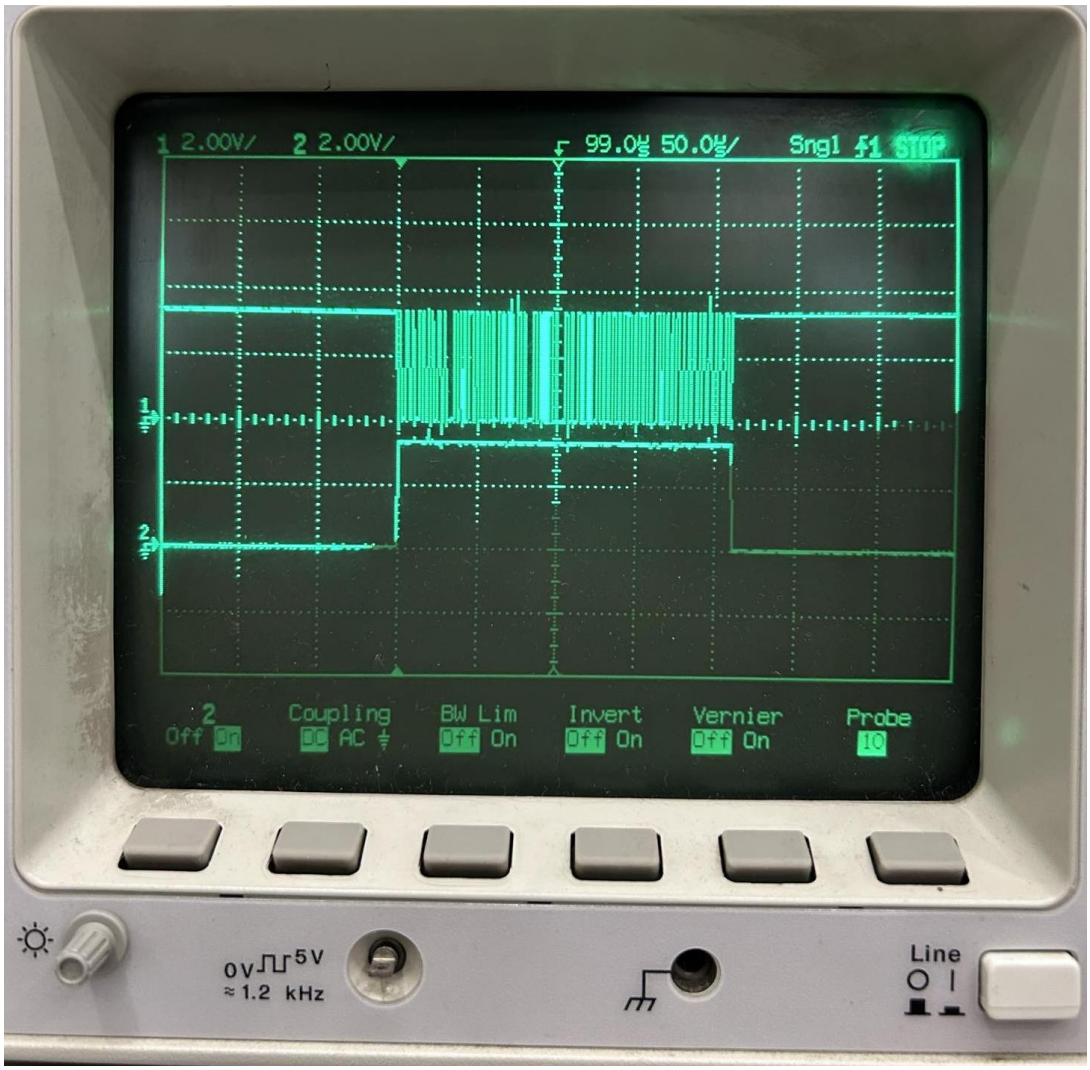
Figure 4-1. SOIC-8 (D), SOT-8 (DRL) Package, Top View

Table 4-1. Pin Functions

PIN		I/O	DESCRIPTION
NAME	NO.		
R	1	Digital output	Receive data output
$\overline{RE}$	2	Digital input	Receiver enable, active low (internal 2-MΩ pull-up)
DE	3	Digital input	Driver enable, active high (internal 2-MΩ pull-down)
D	4	Digital input	Driver data input
GND	5	Ground	Device ground
A	6	Bus input/output	Bus I/O port, A (complementary to B)
B	7	Bus input/output	Bus I/O port, B (complementary to A)
V_{CC}	8	Power	3.3-V to 5-V supply

Πίνακας 3:Pin Configuration του ολοκληρωμένου THVD1420

Τα pins RE’ και DE συνδέονται στο ίδιο pin του μικροελεγκτή αφού το RE’ είναι αρνητικής λογικής. Όταν το pin το μικροελεγκτή είναι high τότε το ολοκληρωμένο λαμβάνει από την UART και βγάζει σήμα RS-485, ενώ όταν είναι low το ολοκληρωμένο λαμβάνει σήμα RS-485 και βγάζει έξοδο UART η οποία μπαίνει ως είσοδος στον μικροελεγκτή, Στην εικόνα φαίνεται το Transmit pin του μικροελεγκτή καθώς και το σήμα RS485-DE.(Κανάλι 1 το Transmit Pin και Κανάλι 2 το Enable σήμα). Η μέτρηση έγινε με τον παλμογράφο 54645A OSCILLOSCOPE της HP.

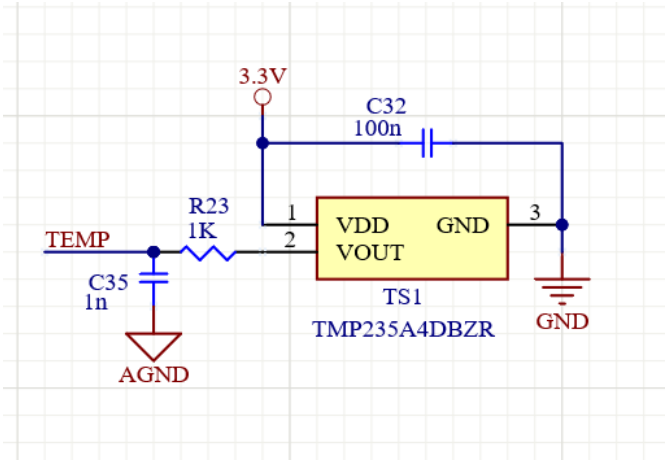


Εικόνα 21:Σήμα Μετάδοσης UART και σήμα επίτρεψης

Στα pin A και B του ολοκληρωμένου έχει συνδεθεί παράλληλα και ένα ολοκληρωμένο με δύο διόδους προστασίας. Η συσκευή CDSOT23-SM712 παρέχει προστασία από ηλεκτροστατικές εκφορτίσεις (ESD), ηλεκτρικά ταχέα μεταβατικά φαινόμενα (EFT) και υπερτάσεις (Surge) για θύρες δεδομένων, ικανοποιώντας τις απαιτήσεις των προτύπων IEC 61000-4-2 (ESD), IEC 61000-4-4 (EFT) και IEC 61000-4-5 (Surge). Η συστοιχία καταστολέων υπερτάσεων (TVS) προσφέρει δύο διόδους TVS με ανώτατη αντίστροφη τάση λειτουργίας 7 V ή 12 V και ελάχιστη τάση διάσπασης 7,5 V ή 13,3 V αντίστοιχα.[27]

Αισθητήρας Θερμοκρασίας

Ο αισθητήρας θερμοκρασίας που υπάρχει στην πλακέτα είναι ο εξής :



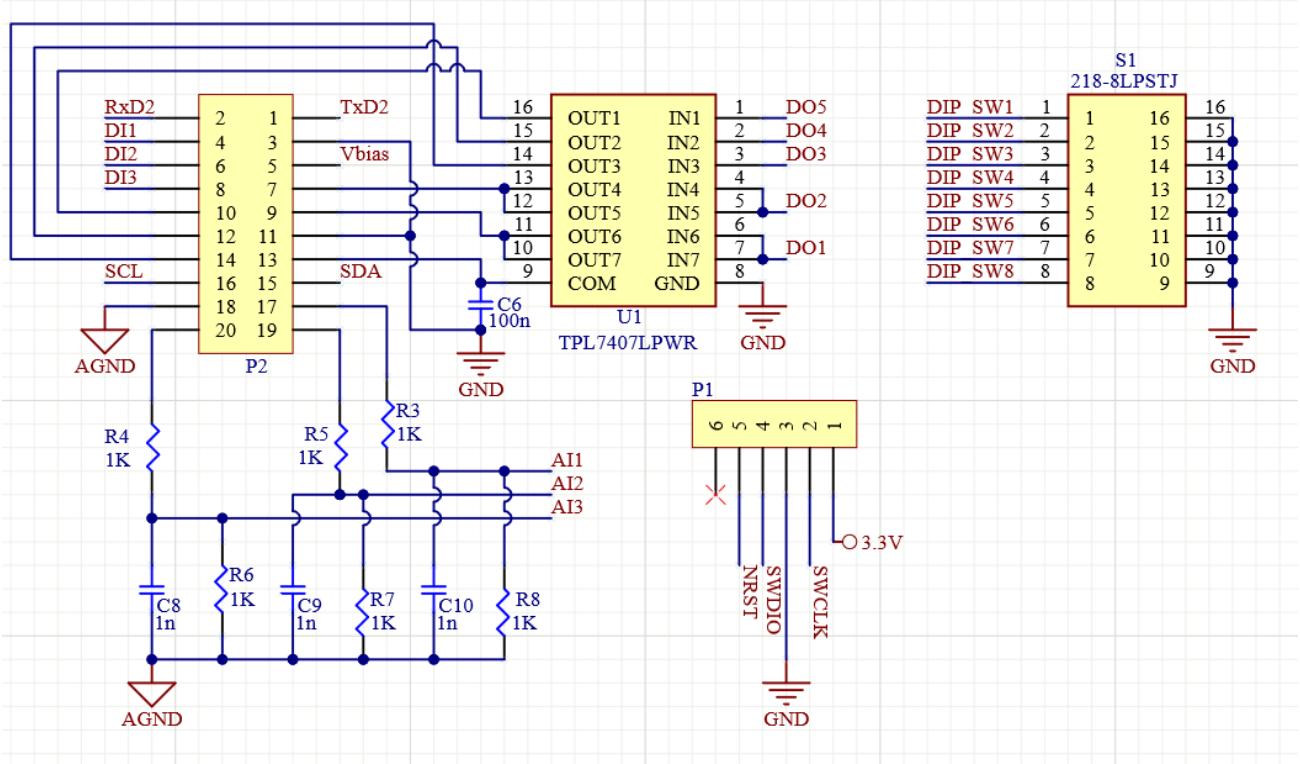
Εικόνα 22:Κύκλωμα αισθητήρα θερμοκρασίας TMP235A4DBZR

Οι συσκευές TMP23x είναι μια οικογένεια ακριβών CMOS ολοκληρωμένων κυκλωμάτων γραμμικών αναλογικών αισθητήρων θερμοκρασίας με τάση εξόδου που είναι ανάλογη της θερμοκρασίας, και μπορούν να χρησιμοποιηθούν σε πολλές αναλογικές εφαρμογές μέτρησης θερμοκρασίας. Αυτοί οι αισθητήρες θερμοκρασίας είναι πιο ακριβείς από παρόμοιες συσκευές με συμβατούς ακροδέκτες που κυκλοφορούν στην αγορά, με τυπική ακρίβεια από 0°C έως +70°C της τάξης του ±0,5°C. Η αυξημένη ακρίβεια της σειράς έχει σχεδιαστεί για πολλές αναλογικές εφαρμογές μέτρησης θερμοκρασίας. Η συσκευή TMP235 παρέχει θετική κλίση εξόδου 10 mV/°C σε όλο το εύρος θερμοκρασίας από -40°C έως +150°C και τροφοδοσία από 2,3 V έως 5,5 V.[28]

Η έξοδος του ολοκληρωμένου συνδέεται απευθείας με το pin του μικροελεγκτή που λαμβάνει δεδομένα στο πρώτο κανάλι του Adc.

Είσοδοι και έξοδοι

Οι είσοδοι και οι έξοδοι της πλακέτας σαν σύστημα φαίνονται στην φωτογραφία:



Εικόνα 23:Είσοδοι και έξοδοι του Συστήματος

Αρχικά το σύστημα έχει μια συστοιχία επαφών (P1) για σύνδεση του μικροελεγκτή με τον Debugger και δυνατότητα αποσφαλμάτωσης, Το debugging πραγματοποιείται μέσω πρωτοκόλλου SWD (Serial Wire Debug) και τα pins του μικροελεγκτή που είναι απαραίτητα για αυτή την λειτουργία είναι τα εξής :

- SWDIO και SWCLK είναι οι βασικές γραμμές για τη σειριακή επικοινωνία του SWD, όπου το SWDIO χρησιμοποιείται για αμφίδρομη μεταφορά δεδομένων και το SWCLK για συγχρονισμό.
- Το NRST επιτρέπει την επανεκκίνηση του μικροελεγκτή κατά τη διαδικασία του debugging.
- Τα GND και 3.3V χρησιμοποιούνται για την παροχή γείωσης και τάσης λειτουργίας, εξασφαλίζοντας σωστή λειτουργία του διαύλου και των εργαλείων debugging.

Αυτή η συστοιχία επαφών επιτρέπει τη σύνδεση με εξωτερικό εξοπλισμό debugging, όπως JTAG/SWD adapters ή programmers, διευκολύνοντας τη φόρτωση κώδικα, την ανάλυση σφαλμάτων και την παρακολούθηση της εκτέλεσης του προγράμματος σε

πραγματικό χρόνο.

Το σύστημα διαθέτει 8 Dip Switches οι οποίοι είναι απευθείας συνδεδεμένοι με τα Pins του μικροελεγκτή τα οποία είναι σε ρύθμιση Pull-Up αφού οι διακόπτες έχουν την μια επαφή τους συνδεδεμένη στα Pins του uC και την άλλη στην γείωση.

Η δεύτερη συστοιχία επαφών του συστήματος (P2) ενσωματώνει 3 αναλογικές εισόδους , 3 ψηφιακές εισόδους , SDA και SCL για χρήση I2C πρωτοκόλλου, RS-485 Rx και Tx , τάση τροφοδοσίας καθώς και 5 ψηφιακές εξόδους οι οποίες προκύπτουν από την έξοδο του ολοκληρωμένου TPL7407LPWR.

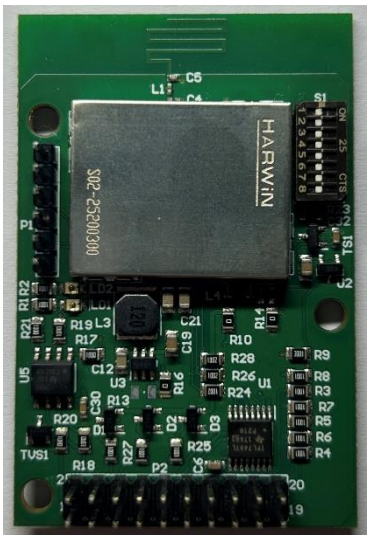
Η συσκευή TPL7407L είναι μια συστοιχία NMOS τρανζίστορ υψηλής τάσης και υψηλού ρεύματος. Αποτελείται από επτά NMOS τρανζίστορ που διαθέτουν εξόδους υψηλής τάσης με διόδους προστασίας κοινής καθόδου για την εναλλαγή επαγωγικών φορτίων. Το μέγιστο ρεύμα (drain current) ενός μεμονωμένου καναλιού NMOS είναι 600 mA. Έχει προστεθεί νέα ρύθμιση και κύκλωμα οδήγησης για μέγιστη ισχύ οδήγησης σε όλα τα εύρη GPIO (1.8 V – 5.0 V). Τα τρανζίστορ μπορούν να συνδεθούν παράλληλα για μεγαλύτερη ικανότητα ρεύματος. Το κύριο πλεονέκτημα του TPL7407L είναι η βελτιωμένη απόδοση ισχύος και η χαμηλότερη διαρροή σε σύγκριση με την υλοποίηση Bipolar Darlington. Με χαμηλότερη τάση VOL, η συσκευή διαχέει λιγότερη από τη μισή ισχύ σε σχέση με τους παραδοσιακούς οδηγούς ρελέ με ρεύματα μικρότερα από 250 mA ανά κανάλι.[29]

Επομένως οι ψηφιακές έξοδοι 3,4,5 έχουν δυνατότητα απόδοσης ρεύματος έως 600mA ενώ οι έξοδοι 1 και 2 έως 1,2A για φορτία ή ρελέ οπλισμού πιο απαιτητικά σε κατανάλωση ρεύματος.

Τέλος, η πλακέτα ενσωματώνει και δυο LED λυχνίες για ενδείξεις λειτουργίας.

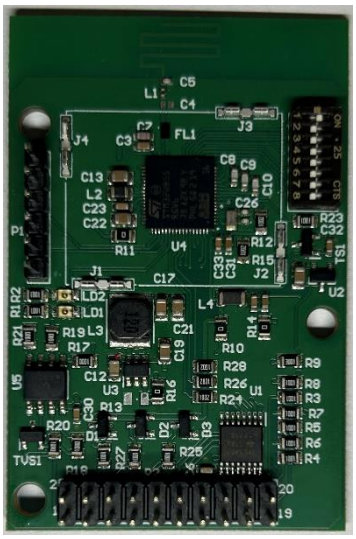
3.3 Πλακέτα Τυπωμένου Κυκλώματος (Printed Circuit Board – PCB)

Το PCB που υλοποιεί το παραπάνω σχηματικό αποτελείται από 4 στρώματα, έχει διαστάσεις 39.6mm x 60.1mm και είναι το εξής:



Εικόνα 24:Πρόσοψη Πλακέτας

Εικόνα 25:Πίσω όψη Πλακέτας



Εικόνα 26:Πρόσοψη Πλακέτας Χωρίς την Προστασία EMI

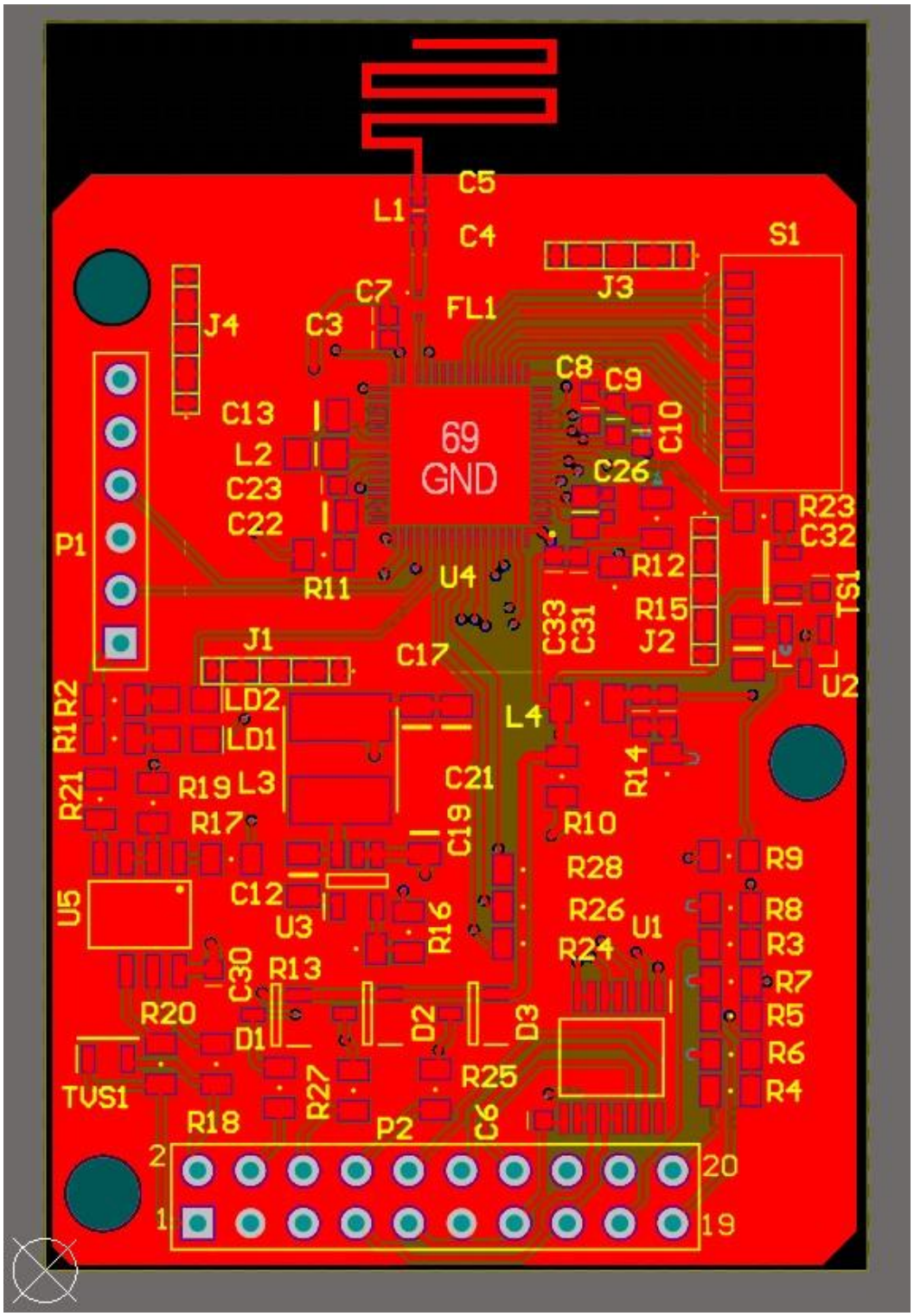
Το μεταλλικό καπάκι είναι EMI gasket. Τα EMI gaskets (gaskets ηλεκτρομαγνητικής παρεμβολής) είναι εξειδικευμένα εξαρτήματα που χρησιμοποιούνται για την προστασία ηλεκτρονικών συσκευών από την ηλεκτρομαγνητική παρεμβολή (EMI). Αυτά τα καπάκια δημιουργούν μια αγώγιμη σφράγιση μεταξύ μεταλλικών επιφανειών, εμποδίζοντας τη μετάδοση ανεπιθύμητων ηλεκτρομαγνητικών σημάτων που θα μπορούσαν να επηρεάσουν τη σωστή λειτουργία των ηλεκτρονικών στοιχείων.[30]

Η δομή και τα χαρακτηριστικά των Layers παρουσιάζονται στον Πίνακα:

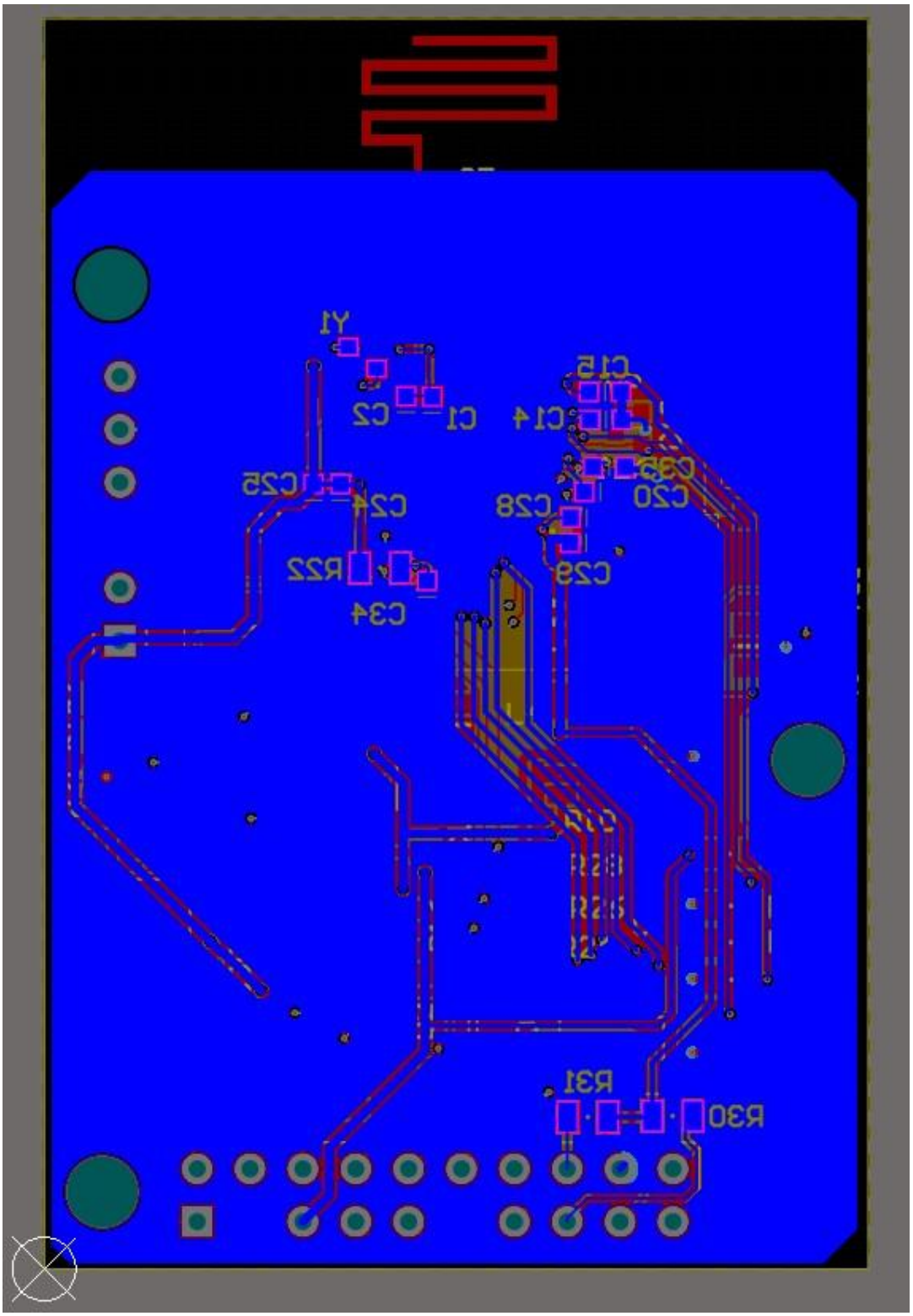
#	Name	Material	Type	Weight	Thickness	Dk	Df
	Top Overlay		Overlay				
	Top Solder	Solder Resist	Solder Mask		0.01mm	3.5	
1	Top Layer		Signal	1oz	0.036mm		
	Dielectric 2	PP-006	Prepreg		0.071mm	4.1	0.02
2	Layer 1	CF-004	Signal	1/2oz	0.018mm		
	Dielectric 1	FR-4	Dielectric		0.32mm	4.8	
3	Layer 2	CF-004	Signal	1/2oz	0.018mm		
	Dielectric 3	PP-006	Prepreg		0.071mm	4.1	0.02
4	Bottom Layer		Signal	1oz	0.036mm		
	Bottom Solder	Solder Resist	Solder Mask		0.01mm	3.5	
	Bottom Overlay		Overlay				

Πίνακας 4:Στρώματα και τα χαρακτηριστικά τους

Το ανώτερο και το κατώτερο στρώμα φαίνονται στις παρακάτω εικόνες.



Εικόνα 27:Ανώτερο Στρώμα Πλακέτας



Εικόνα 28:Κατώτερο Στρώμα Πλακέτας

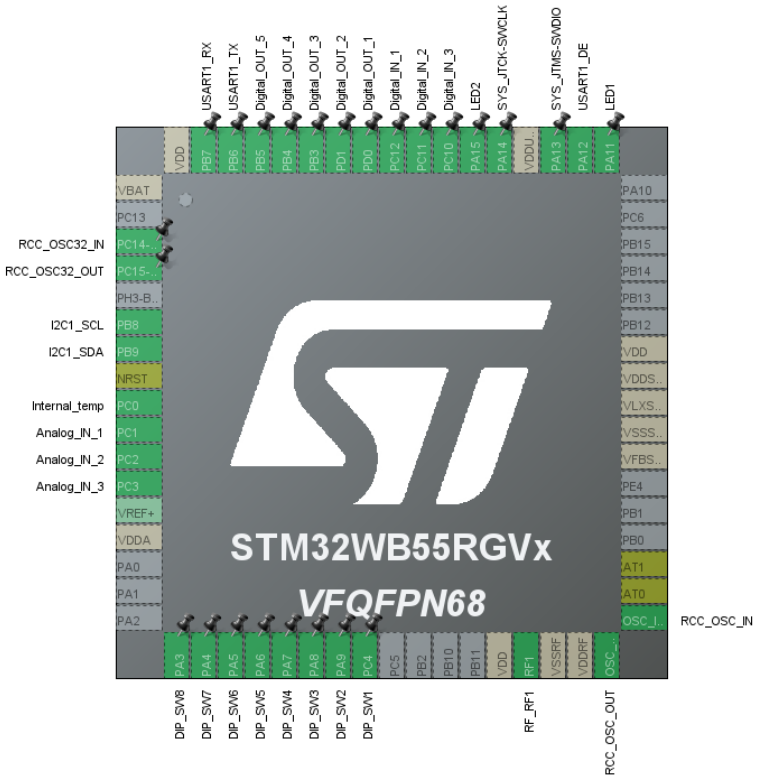
4 Σχεδιασμός εφαρμογής – Software

4.1 Διαμόρφωση του Μικροελεγκτή

Firmware του συνεπεξεργαστή Arm® Cortex®-M0+

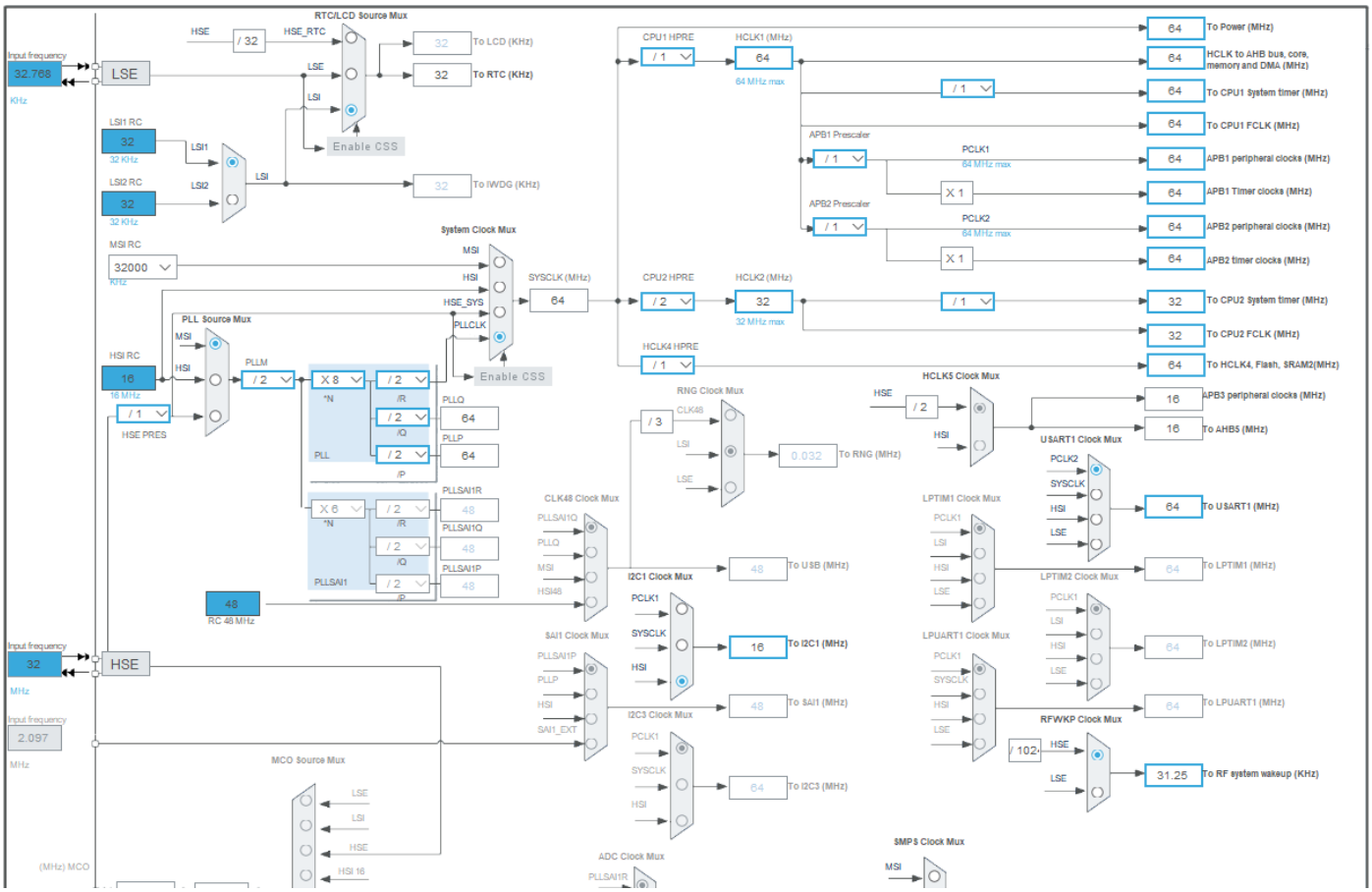
Η εφαρμογή αυτή απαιτεί να έχει φορτωθεί το δυαδικό αρχείο stm32wb5x_Zigbee_FFD_fw.bin στον Ασύρματο Συνεπεξεργαστή. Αυτό είναι εφικτό με χρήση του λογισμικού STM32CubeProgrammer και συγκεκριμένα της λειτουργίας Firmware Upgrade Services. Όλα τα διαθέσιμα δυαδικά αρχεία βρίσκονται στον κατάλογο /Projects/STM32_Copro_Wireless_Binaries, εντός του πακέτου λογισμικού που παρέχει η st για τους συγκεκριμένους μικροελεγκτές.

Διαμόρφωση των pins



Εικόνα 29:Διαμόρφωση των Pins του μικροελεγκτή

Διαμόρφωση των Ρολογιών



Εικόνα 30:Διαμόρφωση των ρολογιών του μικροελεγκτή

Περιφερειακά

Η διαμόρφωση του **ADC** περιλαμβάνει ανάλυση 12-bit με δεξιά στοίχιση δεδομένων, ενεργοποιημένη λειτουργία σάρωσης, ενώ η συνεχής και η διακεκομμένη λειτουργία μετατροπής είναι απενεργοποιημένες. Τα αιτήματα DMA είναι ενεργοποιημένα με το τέλος της μετατροπής να γίνεται μετά από μεμονωμένη μετατροπή και η υπερχείλιση να διατηρεί τα δεδομένα. Ο αριθμός των κανονικών μετατροπών είναι 4, εκτελούμενες στα κανάλια 1, 2, 3 και 4, με χρόνο δειγματοληψίας 6,5 κύκλων για το καθένα. Η εκκίνηση της μετατροπής γίνεται μέσω λογισμικού, ενώ η λειτουργία DMA έχει ρυθμιστεί σε κυκλική λειτουργία, μεταφέροντας δεδομένα από το περιφερειακό στη μνήμη.

Η διαμόρφωση των **Timers** καθώς και η λειτουργία τους παρουσιάζεται εκτενώς στο κεφάλαιο 4.3 και ενότητα: *Χρονισμοί και Δρομολόγηση*.

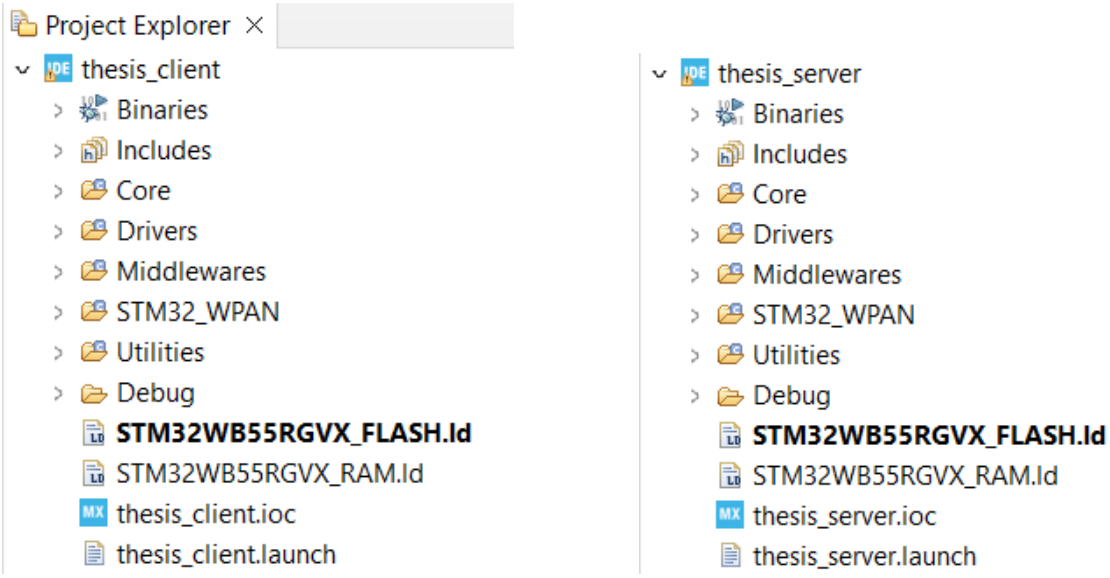
Η διαμόρφωση της **USART** περιλαμβάνει βασικές παραμέτρους όπως baud rate 1.000.000 bits/s, μήκος λέξης 8 bits, χωρίς χρήση ισοτιμίας (parity) και 1 bit διακοπής. Η κατεύθυνση των δεδομένων είναι τόσο για αποστολή όσο και για λήψη, με υπερδειγματοληψία 8 δειγμάτων και απενεργοποιημένη τη λειτουργία του single sample. Ο διαιρέτης του ρολογιού (ClockPrescaler) είναι ρυθμισμένος στο 1, και οι λειτουργίες FIFO, καθώς και τα Tx/Rx FIFO thresholds, είναι ρυθμισμένες σε 1/8 πλήρους διαμόρφωσης. Η πόλωση (polarity) είναι ρυθμισμένη σε υψηλή, ενώ οι χρόνοι assertion και deassertion είναι και οι δύο στο 0 sample time unit.Τα δεδομένα λαμβάνονται με χρήση διακοπής.

Η διαμόρφωση του **I2C** περιλαμβάνει τη χρήση Fast Mode με ταχύτητα επικοινωνίας 400 KHz. Το Custom Timing είναι απενεργοποιημένο, ενώ ο χρόνος ανόδου (Rise Time) έχει ρυθμιστεί στα 250 ns και ο χρόνος καθόδου (Fall Time) στα 30 ns. Το ψηφιακό φίλτρο έχει συντελεστή 0, ενώ το αναλογικό φίλτρο είναι ενεργοποιημένο. Η ρύθμιση του χρονισμού (Timing) έχει τιμή 0x00500616. Σχετικά με τα χαρακτηριστικά του slave, το Clock No Stretch Mode και η ανίχνευση γενικής διεύθυνσης (General Call Address Detection) είναι απενεργοποιημένα. Η επιλογή μήκους της κύριας διεύθυνσης είναι 7-bit, ενώ η αναγνώριση διπλής διεύθυνσης (Dual Address Acknowledged) είναι απενεργοποιημένη, με την κύρια διεύθυνση του slave ρυθμισμένη στο 0. Για την ανάγνωση από κάποιο αισθητήρα η συσκευή πρέπει να λειτουργήσει σε Master Receive mode.

4.2 Δομή της Εφαρμογής

Γενική Δομή

Για την υλοποίηση της εφαρμογής χρειάζονται δύο διαφορετικές υλοποιήσεις κώδικα. Η πρώτη θα ενσωματωθεί στην πλακέτα η οποία θα χρησιμοποιηθεί ως διακομιστής του δικτύου. Η δεύτερη θα ενσωματωθεί στις πλακέτες που θα χρησιμοποιηθούν ως δρομολογητές ή τελικές συσκευές εντός του δικτύου που έχει δημιουργήσει ο κεντρικός διακομιστής.

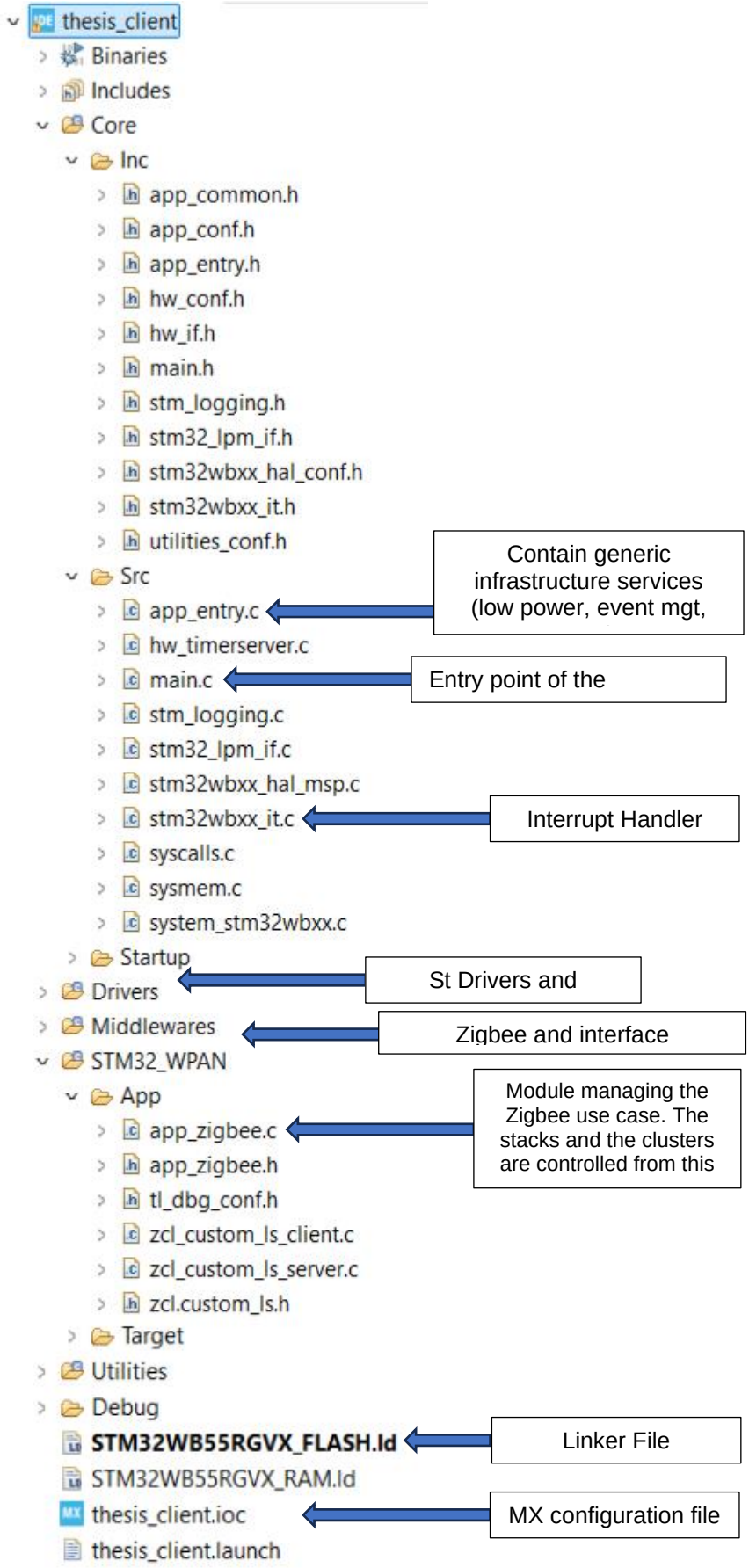


Εικόνα 31:Δομή των projects για τον δρομολογητή και τον διακομιστή

Τα δύο project έχουν ίδια δομή και αρχιτεκτονική. Μικρές Διαφορές υπάρχουν μόνο στις λειτουργίες των δυο Project οι οποίες θα αναφερθούν λεπτομερώς παρακάτω.

Δομή του Project

Η δομή του Project που αφορά τον client/router φαίνεται παρακάτω. Οι βασικές αρχικοποιήσεις των περιφερειακών του μικροελεγκτή βρίσκονται εντός του main.c το οποίο είναι και το αρχείο εκκίνησης της εφαρμογής και συγκεκριμένα η συνάρτηση main() εντός του ομώνυμο αρχείου. Οι βασικές λειτουργίες της εφαρμογής υλοποιούνται εντός του app_zigbee.c αρχείου το οποίο είναι και υπεύθυνο για την αρχικοποίηση της στοίβας του zigbee πρωτοκόλλου. Στο ίδιο αρχείο μπορούν να υλοποιηθούν και οι διαφορετικές λειτουργίες της εκάστοτε εξατομικευμένης εφαρμογής (π.χ. έλεγχοι από διαφορετικούς αισθητήρες , διαφορετικές λειτουργίες ανάλογα με τους ελέγχους κλπ.). Εντός του φακέλου Drivers βρίσκονται πολύ χρήσιμοι οδηγοί και βιβλιοθήκες της ST Microelectronics οι οποίες χρησιμοποιούνται για τον συγκεκριμένο μικροελεγκτή. Εντός του φακέλου Middlewares βρίσκονται τα βασικά αρχεία και βιβλιοθήκες για την αξιοποίηση και χρήση της στοίβας του zigbee πρωτοκόλλου. Στα βασικά αρχεία συγκαταλέγεται και το thesis_client.ioc, το οποίο είναι το αρχείο διαμόρφωσης του Hardware του μικροελεγκτή. Όπως αναφέρθηκε και προηγουμένως τα δύο Project έχουν την ίδια δομή. [5]



Εικόνα 32:Δομή και περιεχόμενα του project

4.3 Κώδικας της εφαρμογής – Ανάλυση Λειτουργικότητας

Εισαγωγή

Ο κώδικάς που παρέχεται από την ST εκτός από τις βασικές βιβλιοθήκες HAL, είναι το ZigBee Stack που τρέχει στον συνεπεξεργαστή M0+, οι συναρτήσεις που υλοποιούν την επικοινωνία του επεξεργαστή M4 με το stack, καθώς και ο Sequencer. Τα υπόλοιπα κομμάτια κώδικα, οι συναρτήσεις καθώς και το custom long string cluster υλοποιήθηκαν κατά την διάρκεια της παρούσας εργασίας και αποτελούν custom κώδικα για την συγκεκριμένη εφαρμογή. [31]

Main Συνάρτηση

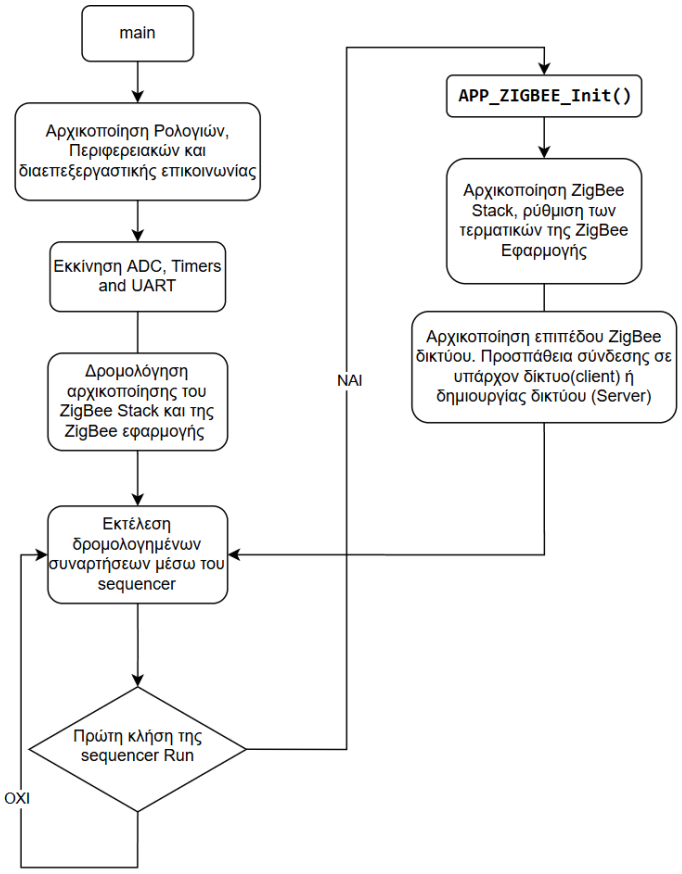
Κατά την εκκίνηση της εφαρμογής αρχικοποιούνται και διαμορφώνονται τα περιφερειακά του μικροελεγκτή με την αντίστοιχη _Init() συνάρτηση του κάθε περιφερειακού. Πριν την είσοδο στο βασικό loop της εφαρμογής αρχικοποιούνται κάποια flags και ενεργοποιούνται κάποια από τα περιφερειακά τα οποία χρησιμοποιούνται μετέπειτα είτε στο βασικό loop είτε σε ρουτίνες εξυπηρέτησης διακοπών.

Το βασικό loop καλεί συνέχεια την συνάρτηση MX_APPE_Process() η οποία είναι η εξής:

```
void MX_APPE_Process(void)
{
  /* USER CODE BEGIN MX_APPE_Process_1 */

  /* USER CODE END MX_APPE_Process_1 */
  UTIL_SEQ_Run(UTIL_SEQ_DEFAULT);
  /* USER CODE BEGIN MX_APPE_Process_2 */

  /* USER CODE END MX_APPE_Process_2 */
}
```

Η μοναδική λειτουργία της συνάρτησης αυτής είναι να καλεί την συνάρτηση UTIL_SEQ_Run(UTIL_SEQ_DEFAULT) η οποία εκτελεί τις συναρτήσεις που έχουν οριστεί στον sequencer για εκτέλεση με βάση την προτεραιότητα τους. Αν δεν υπάρχει κάποια συνάρτηση προς εκτέλεση στην ουρά αναμονής τότε ο sequencer εκτελεί την UTIL_SEQ_Idle() η οποία ουσιαστικά θέτει τον ελεγκτή σε Idle State.

Η λειτουργία της main συνάρτησης παρουσιάζεται συνοπτικά με το παρακάτω διάγραμμα:



Εικόνα 33:Διάγραμμα ροής main()

Η υλοποίηση της συνάρτησης main() τόσο για τον server όσο και για τον router/client είναι η εξής:

```
int main(void)
{

  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------
----*/

  /* Reset of all peripherals, Initializes the Flash interface and the
Systick. */
  HAL_Init();
  /* Config code for STM32_WPAN (HSE Tuning must be done before system
clock configuration) */
  MX_APPE_Config();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();
```

```c
/* Configure the peripherals common clocks */
  PeriphCommonClock_Config();

  /* IPCC initialisation */
  MX_IPCC_Init();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_USART1_UART_Init();
  MX_RTC_Init();
  MX_ADC1_Init();
  MX_TIM1_Init();
  MX_TIM2_Init();
  MX_TIM16_Init();
  MX_TIM17_Init();
  MX_I2C1_Init();
  MX_RF_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Init code for STM32_WPAN */
  MX_APPE_Init();

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  //HAL_UART_Receive_DMA(&huart1, rxdata, sizeof(rxdata));
  HAL_UARTEx_ReceiveToIdle_IT(&huart1, rxdata, 50);
  __HAL_TIM_CLEAR_FLAG(&htim1, TIM_FLAG_UPDATE);
  HAL_TIM_Base_Start_IT(&htim1);
  __HAL_TIM_CLEAR_FLAG(&htim2, TIM_FLAG_UPDATE);
  HAL_TIM_Base_Start_IT(&htim2);
  __HAL_TIM_CLEAR_FLAG(&htim16, TIM_FLAG_UPDATE);
  HAL_TIM_Base_Start_IT(&htim16);
  __HAL_TIM_CLEAR_FLAG(&htim17, TIM_FLAG_UPDATE);
  HAL_TIM_Base_Start_IT(&htim17);
  HAL_ADCEx_Calibration_Start(&hadc1, ADC_SINGLE_ENDED);
  ADC_Enable(&hadc1);
  __HAL_ADC_CLEAR_FLAG(&hadc1, ADC_FLAG_EOSMP);
  __HAL_ADC_ENABLE_IT(&hadc1, ADC_IT_EOSMP);
  HAL_ADC_Start_DMA(&hadc1, &Board_Data[Board_Number].adc[0], 4);

  while (1)
  {
    /* USER CODE END WHILE */
    MX_APPE_Process();

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

ST sequencer [32,33]

Ο sequencer είναι κώδικας υλοποιημένος από την ST Microelectronics ο οποίος είναι ένα σημαντικό εργαλείο που χρησιμοποιείται για τον προγραμματισμό και τη διαχείριση των εργασιών (tasks) σε πραγματικό χρόνο σε έναν μικροελεγκτή. Λειτουργεί ως προγραμματιστής εργασιών, διαχειρίζεται πότε και πώς θα εκτελεστούν συγκεκριμένες εργασίες. Μπορεί να προγραμματίσει τη σειρά και τη χρονική στιγμή εκτέλεσης των διαφόρων εργασιών με βάση προτεραιότητες ή συγκεκριμένα συμβάντα. Είναι κρίσιμος για συστήματα πραγματικού χρόνου, όπου η έγκαιρη εκτέλεση των εργασιών είναι ζωτικής σημασίας. Ο sequencer εξασφαλίζει ότι οι εργασίες εκτελούνται με την ακριβή σειρά και στον σωστό χρόνο, διασφαλίζοντας την ομαλή λειτουργία του συστήματος. Μπορεί να συνεργάζεται με άλλους ελεγκτές ή modules εντός του μικροελεγκτή, όπως DMA controllers, timers, και άλλα περιφερειακά, για να διασφαλίσει ότι οι εργασίες εκτελούνται με τον πιο αποδοτικό τρόπο.

Ο sequencer δεν είναι λειτουργικό σύστημα. Δεν σκοπεύει να ανταγωνιστεί τα τυπικά λειτουργικά συστήματα, αλλά αντίθετα τις τυπικές bare metal υλοποιήσεις. Είναι μια βελτιστοποιημένη συσκευασία μιας κλασικής υλοποίησης bare metal με while loop.και επιτρέπει την αποφυγή συνθηκών συναγωνισμού (race conditions) που συχνά αντιμετωπίζονται σε τέτοιου είδους υλοποιήσεις, ειδικά όταν εφαρμόζονται λειτουργίες χαμηλής κατανάλωσης ενέργειας.

Οι βασικές συναρτήσεις του sequencer είναι οι εξής:

UTIL_SEQ_Run()

Αυτή η συνάρτηση εκκινεί την εκτέλεση των προγραμματισμένων εργασιών (tasks) στον sequencer. Ο sequencer εκτελεί όλες τις εργασίες που έχουν προγραμματιστεί να εκτελεστούν σε αυτήν την ουρά, μέχρι να μην υπάρχουν άλλες εκκρεμείς. Στην ουσία, η UTIL_SEQ_Run() συνεχίζει την εκτέλεση των διαθέσιμων tasks και διασφαλίζει ότι εκτελούνται σε καθορισμένη σειρά.

UTIL_SEQ_Idle()

Αυτή η συνάρτηση καθορίζει τη συμπεριφορά του συστήματος όταν δεν υπάρχουν άλλες εργασίες για εκτέλεση. Είναι η συνάρτηση που καλείται όταν το σύστημα μπαίνει σε κατάσταση αδράνειας (idle). Μπορεί να χρησιμοποιηθεί για εξοικονόμηση ενέργειας ή για να εισάγει το σύστημα σε κατάσταση αναμονής μέχρι να εμφανιστεί κάποιο νέο γεγονός ή task.

UTIL_SEQ_RegTask()

Αυτή η συνάρτηση χρησιμοποιείται για να καταχωρήσει μια νέα εργασία (task) στον sequencer. Η εργασία που καταχωρείται συνήθως περιλαμβάνει μια συνάρτηση που θα εκτελεί συγκεκριμένες λειτουργίες.

UTIL_SEQ_SetTask()

Αυτή η συνάρτηση χρησιμοποιείται για να προσθέσει μια καταχωρημένη εργασία στην ουρά εκτέλεσης του sequencer. Αυτό σημαίνει ότι όταν καλείται, η εργασία θα προγραμματιστεί για εκτέλεση από τον sequencer κατά την επόμενη φάση εκτέλεσης με βάση την προτεραιότητα της.

UTIL_SEQ_PauseTask()

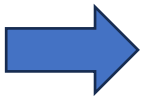
Αυτή η συνάρτηση χρησιμοποιείται για να σταματήσει προσωρινά την εκτέλεση μιας συγκεκριμένης καταχωρημένης εργασίας. Με την παύση, η εργασία παραμένει στην ουρά, αλλά δεν εκτελείται μέχρι να επανέλθει.

UTIL_SEQ_ResumeTask ()

Αυτή η συνάρτηση χρησιμοποιείται για να επαναφέρει μια εργασία που είχε προηγουμένως σταματήσει με την UTIL_SEQ_PauseTask(). Όταν καλείται, η εργασία επανέρχεται στην ουρά εκτέλεσης και θα εκτελεστεί όταν ο sequencer εκκινήσει την εκτέλεση των tasks.

Συνοπτικά ο sequencer θα μπορούσε να προσομοιωθεί με ένα while loop ώς εξής

```
While(1)
{
        UTIL_SEQ_Run();
}
```



```
While(1)
{
        if(flag1)
        {
                flag1 = 0;
                Fct1();
        }

        if (flag2)
        {
                flag2 = 0;
                Fct2();
        }

        __disable_irq();
        If (! (flag1 || flag2))
        {
                UTIL_SEQ_Idle();
        }
        __enable_irq();
}
```

Η ***UTIL_SEQ_RegTask()*** προσθέτει στο βρόχο while την υποστήριξη μιας νέας συνάρτησης

```
UTIL_SEQ_RegTask( flag1, Fct1());
UTIL_SEQ_RegTask( flag2, Fct2());
```

```
While(1)
{
        if(flag1)
        {
                flag1 = 0;
                Fct1();
        }

        if (flag2)
        {
                flag2 = 0;
                Fct2();
        }

        __disable_irq();
        If (! (flag1 || flag2))
        {
                UTIL_SEQ_Idle();
        }
        __enable_irq();
}
```

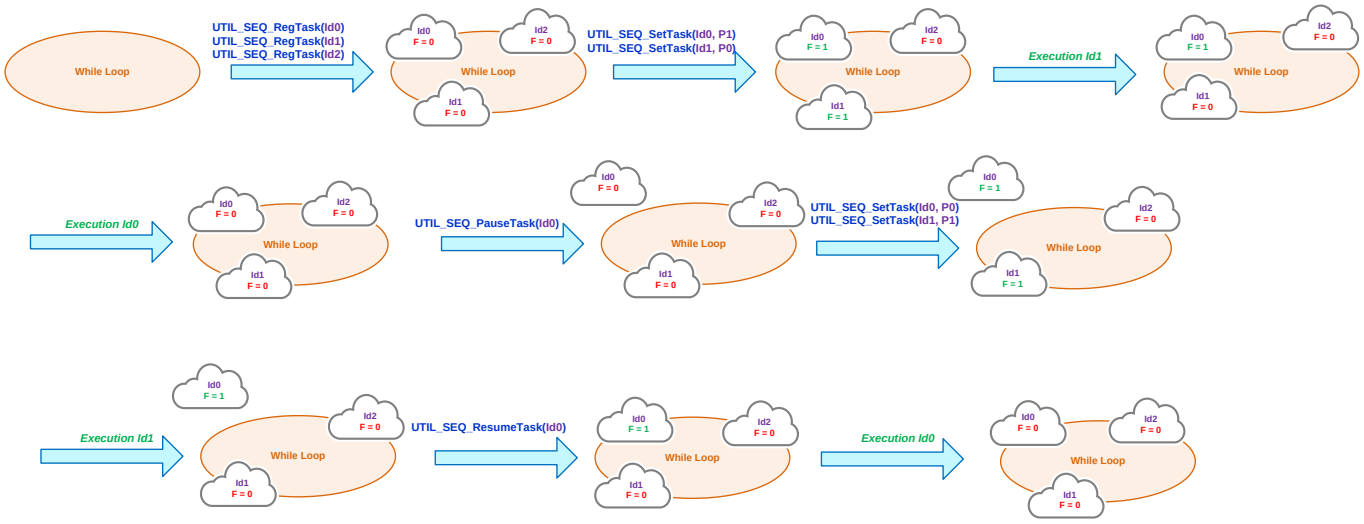
Η ***UTIL_SEQ_SetTask()*** θέτει το flag που αντιστοιχεί στη συνάρτηση που θα εκτελεστεί στο βρόχο while

```
UTIL_SEQ_SetTask( flag1);      =>     flag1 = 1
```

Η ***UTIL_SEQ_PauseTask()*** αποτρέπει την εκτέλεση της συνάρτησης στη βρόχο while ανεξαρτήτως της τιμής του αντίστοιχου flag. επιτρέπει ξανά την εκτέλεση της συνάρτησης αν το σχετικό flag έχει οριστεί.

Η ***UTIL_SEQ_ResumeTask ()*** επιτρέπει ξανά την εκτέλεση της συνάρτησης αν το σχετικό flag έχει οριστεί.

Ένα παράδειγμα δρομολόγησης του sequencer είναι το εξής:



Εικόνα 34:Παράδειγμα δρομολόγησης sequencer

When F = 1, the sequencer executes the task Prior execution, the sequencer sets back F to '0' Priority P0 is the highest

Οι συναρτήσεις που χειρίζεται ο Sequencer στην εφαρμογή

Οι συναρτήσεις που υποστηρίζονται από τον sequencer στην εφαρμογή αυτή είναι οι εξής:

Για τον Server::

```
typedef enum
{
  CFG_TASK_NOTIFY_FROM_M0_TO_M4,
  CFG_TASK_REQUEST_FROM_M0_TO_M4,
  CFG_TASK_ZIGBEE_NETWORK_FORM,
  CFG_TASK_SYSTEM_HCI_ASYNCH_EVT,
#if (CFG_USB_INTERFACE_ENABLE !=
0)
  CFG_TASK_VCP_SEND_DATA,
#endif /*
(CFG_USB_INTERFACE_ENABLE != 0)
*/
  /* USER CODE BEGIN
CFG_IdleTask_Id_t */
  CFG_RECEIVE_UART,
  CFG_TRANSMIT_UART,
  CFG_TRANSMIT_ZIGBEE,
  CFG_TRANSMIT_HEARTBEAT,
  CFG_ALLOW_JOIN,
  /* USER CODE END
CFG_IdleTask_Id_t */
  CFG_TASK_NBR  /**< Shall be
last in the list */
} CFG_IdleTask_Id_t;
```

Για τον Router/Client

```
typedef enum
{
  CFG_TASK_NOTIFY_FROM_M0_TO_M4,
  CFG_TASK_REQUEST_FROM_M0_TO_M4,
  CFG_TASK_ZIGBEE_NETWORK_FORM,
  CFG_TASK_SYSTEM_HCI_ASYNCH_EVT,
#if (CFG_USB_INTERFACE_ENABLE !=
0)
  CFG_TASK_VCP_SEND_DATA,
#endif /*
(CFG_USB_INTERFACE_ENABLE != 0)
*/
  /* USER CODE BEGIN
CFG_IdleTask_Id_t */
  CFG_RECEIVE_UART,
  CFG_TRANSMIT_UART,
  CFG_TRANSMIT_ZIGBEE,
  CFG_HEARTBEAT_CHECK,
  /* USER CODE END
CFG_IdleTask_Id_t */
  CFG_TASK_NBR  /**< Shall be
last in the list */
} CFG_IdleTask_Id_t;
```

Οι συναρτήσεις :`CFG_TASK_NOTIFY_FROM_M0_TO_M4`, `CFG_TASK_REQUEST_FROM_M0_TO_M4`, `CFG_TASK_SYSTEM_HCI_ASYNCH_EVT` καταχωρούνται προς εκτέλεση από την zigbee στοίβα με βάση τον κώδικα που δίνει η ST Microelectronics ως Middleware για την χρήση της στοίβας και αφορούν την επικοινωνία μεταξύ των δύο πυρήνων του μικροελεγκτή η οποία είναι αναγκαία για την χρήση του πρωτοκόλλου. Οι υπόλοιπες συναρτήσεις καταχωρούνται προς εκτέλεση κατά βάση από τις διακοπές των timers που έχουν οριστεί για την συγκεκριμένη εφαρμογή σύμφωνα με την υλοποίηση που παρουσιάζεται στην ενότητα *Χρονισμοί και Δρομολόγηση.*

Αρχικοποίηση Zigbee Εφαρμογής

Κατά την εκκίνηση της εφαρμογής έπειτα από εντολή του ασύρματου συνεπεξεργαστή η πρώτη συνάρτηση που εκτελείται από τον sequencer είναι η **void APP_ZIGBEE_Init(void)**

Η συνάρτηση APP_ZIGBEE_Init() είναι υπεύθυνη για την αρχικοποίηση του πρωτοκόλλου Zigbee σε ένα σύστημα που χρησιμοποιεί συνεπεξεργαστή για ασύρματη επικοινωνία. Ξεκινά με έναν έλεγχο συμβατότητας του firmware που είναι φορτωμένο στον συνεπεξεργαστή, εξασφαλίζοντας ότι το λογισμικό είναι κατάλληλο για τη λειτουργία του Zigbee. Στη συνέχεια, εγγράφει έναν buffer εντολών που θα χρησιμοποιηθεί για την επικοινωνία με το πρωτόκολλο, και αρχικοποιεί τη διαχείριση δεδομένων μέσω της συνάρτησης TL_ZIGBEE_Init.

Ακολουθεί η καταχώρηση (sequencer) διαφόρων εργασιών (tasks) που διαχειρίζονται τις ειδοποιήσεις και τα αιτήματα μεταξύ του επεξεργαστή M0 και του M4, όπως η δημιουργία δικτύου Zigbee. Έπειτα καταχωρεί τις υπόλοιπες εργασίες που αφορούν την επικοινωνία μέσω UART, την αποστολή δεδομένων μέσω Zigbee, καθώς και η παρακολούθηση της σταθερότητας του συστήματος με την εργασία Heartbeat_Check. Αφού ολοκληρωθούν αυτές οι ρυθμίσεις, η συνάρτηση ξεκινά τη λειτουργία του Zigbee στον επεξεργαστή CPU2 και αρχικοποιεί τα επίπεδα του Zigbee stack που είναι απαραίτητα για την επικοινωνία.

Τα επίπεδα του ZigBee stack αρχικοποιούνται μέσω της συνάρτησης **void APP_ZIGBEE_StackLayersInit**(**void**). Η συγκεκριμένη συνάρτηση εκτελεί τρεις βασικές κλήσεις.

Η πρώτη είναι η συνάρτηση **ZbInit()** η οποία είναι υπεύθυνη για την αρχικοποίηση μιας δομής Zigbee, επιτρέποντας την έναρξη και ρύθμιση των παραμέτρων επικοινωνίας για το πρωτόκολλο Zigbee. Μέσω της εντολής MSG_M4TOM0_ZB_INIT η συγκεκριμένη συνάρτηση λαμβάνει ως επιστροφή από

τον συνεπεξεργαστή, έπειτα από επεξεργασία του ZigBee stack, δείκτη στη δομή Zigbee (zigbee_app_info.zb).

Η δεύτερη κλήση είναι η συνάρτηση **APP_ZIGBEE_ConfigEndpoints()** η οποία είναι υπεύθυνη για τη διαμόρφωση των endpoints στο Zigbee δίκτυο, δηλαδή των σημείων επικοινωνίας της συσκευής που επιτρέπουν την αλληλεπίδραση με άλλες Zigbee συσκευές. Αρχικά, δημιουργείται ένα αίτημα προσθήκης ενός endpoint με τα κατάλληλα χαρακτηριστικά, όπως το profileId που καθορίζει το προφίλ επικοινωνίας (στην προκειμένη περίπτωση το ZCL_PROFILE_HOME_AUTOMATION) και το deviceId, το οποίο αντιστοιχεί σε έναν διακόπτη on/off. Το αίτημα προστίθεται στο Zigbee stack μέσω της συνάρτησης ZbZclAddEndpoint, και ελέγχεται αν η προσθήκη ήταν επιτυχής.

Στη συνέχεια, προστίθενται επιπλέον clusters. [34] Ειδικότερα, δημιουργούνται και καταχωρούνται ένας server και ένας client για το custom long string cluster, το οποίο μπορεί να υποστηρίζει συγκεκριμένες λειτουργίες και δεδομένα που σχετίζονται με την αποστολή και λήψη δεδομένων μέσω string. Αυτά τα clusters επιτρέπουν στην εφαρμογή να ανταλλάσσει εξειδικευμένα μηνύματα και να αλληλεπιδρά με άλλες συσκευές στο δίκτυο Zigbee μέσω του καθορισμένου endpoint.

Η τρίτη κλήση είναι η συνάρτηση **APP_ZIGBEE_NwkForm.** [35,36] Η συνάρτηση είναι υπεύθυνη για τη διαδικασία δημιουργίας ή συμμετοχής σε ένα Zigbee δίκτυο. Κατά την εκτέλεσή της, ελέγχει αν η συσκευή δεν έχει ήδη συνδεθεί με επιτυχία στο δίκτυο (ελέγχει το join_status) και αν έχει παρέλθει ο απαιτούμενος χρόνος αναμονής για την επόμενη προσπάθεια σύνδεσης (χρησιμοποιώντας τη συνάρτηση HAL_GetTick()).

Αν αυτές οι συνθήκες πληρούνται, διαμορφώνεται η δομή ZbStartupT που περιέχει τις παραμέτρους εκκίνησης του Zigbee δικτύου, όπως ο τύπος σύνδεσης (ZbStartTypeJoin) και το προεπιλεγμένο κλειδί ασφαλείας (link key) για το προφίλ Home Automation. Η συσκευή προσπαθεί να συνδεθεί στο δίκτυο μέσω της ZbStartupWait(), και αν η διαδικασία ολοκληρωθεί με επιτυχία (status ZB_STATUS_SUCCESS), η συνάρτηση ολοκληρώνει την αρχικοποίηση των παραμέτρων του δικτύου, ενημερώνοντας τις εσωτερικές καταστάσεις και ενεργοποιώντας τις κατάλληλες ενδείξεις (όπως LEDs για επιβεβαίωση σύνδεσης).

Σε περίπτωση αποτυχίας σύνδεσης, η συνάρτηση προγραμματίζει μια νέα προσπάθεια μετά από ένα καθορισμένο χρονικό διάστημα καθυστέρησης

(join_delay). Επιπλέον, διαχειρίζεται τις προσπάθειες επανένταξης (rejoin) και διαμορφώνει τις ομαδικές διευθύνσεις για την υποστήριξη της επικοινωνίας μέσω broadcast αν η σύνδεση ήταν επιτυχής. Αν αποτύχει μετά από πολλαπλές προσπάθειες, ενημερώνεται η κατάσταση του συστήματος για να δείξει ότι η σύνδεση δεν επετεύχθη.

Η διαμόρφωση και η προσθήκη της συσκευής στην ομαδική διεύθυνση γίνεται μέσω της συνάρτησης **APP_ZIGBEE_ConfigGroupAddr()** [36] η οποία είναι υπεύθυνη για τη διαμόρφωση των ομαδικών διευθύνσεων (group addresses) στο Zigbee δίκτυο. Οι ομαδικές διευθύνσεις χρησιμοποιούνται για την επικοινωνία με πολλαπλές συσκευές ταυτόχρονα μέσω μηνυμάτων broadcast.

Η συνάρτηση ρυθμίζει το endpoint στο οποίο θα αντιστοιχιστεί η ομαδική διεύθυνση (στην προκειμένη περίπτωση το SW1_ENDPOINT) και ορίζει τη διεύθυνση της ομάδας. Τέλος, η αίτηση αποστέλλεται στο Zigbee stack μέσω της ZbApsmeAddGroupReq(), η οποία προσθέτει τη συσκευή στην καθορισμένη ομάδα και επιστρέφει μια επιβεβαίωση μέσω της δομής ZbApsmeAddGroupConfT. Αυτή η διαδικασία επιτρέπει στη συσκευή να λαμβάνει μηνύματα που απευθύνονται σε αυτήν την ομάδα.

Ενδεικτικά η συνάρτηση **APP_ZIGBEE_NwkForm** για τον Client είναι η εξής:

```
static void APP_ZIGBEE_NwkForm(void)
{
  if((zigbee_app_info.join_status != ZB_STATUS_SUCCESS) && (HAL_GetTick()
>= zigbee_app_info.join_delay))
  {
    struct ZbStartupT config;
    enum ZbStatusCodeT status;

    /* Configure Zigbee Logging (only need to do this once, but this is a good place to put it) */
    ZbSetLogging(zigbee_app_info.zb, ZB_LOG_MASK_LEVEL_5, NULL);

    /* Attempt to join a zigbee network */
    ZbStartupConfigGetProDefaults(&config);

    APP_DBG("Network config : APP_STARTUP_CENTRALIZED_ROUTER");
    zigbee_app_info.startupControl = ZbStartTypeJoin;
    config.startupControl = zigbee_app_info.startupControl;



    /* Using the default HA preconfigured Link Key */
    memcpy(config.security.preconfiguredLinkKey, sec_key_ha,
ZB_SEC_KEYSIZE);
    config.channelList.count = 1;
    config.channelList.list[0].page = 0;
```

```
    config.channelList.list[0].channelMask = 1 << CHANNEL; /* Channel in use*/

    /* Using ZbStartupWait (blocking) here instead of ZbStartup, in order to demonstrate how to do
          * a blocking call on the M4. */
    status = ZbStartupWait(zigbee_app_info.zb, &config);

    APP_DBG("ZbStartup Callback (status = 0x%02x)", status);
    zigbee_app_info.join_status = status;

    if (status == ZB_STATUS_SUCCESS) {
      zigbee_app_info.join_delay = 0U;
      zigbee_app_info.init_after_join = true;
      APP_DBG("Startup done !\n");
      /* USER CODE BEGIN 0 */
      MyNwkAddress = ZbShortAddress(zigbee_app_info.zb);
      MyExtAddress = ZbExtendedAddress(zigbee_app_info.zb);
      Zigbee_Connected = 1;
      Board_Data[Board_Number].status = 1;
      Need_Rejoin = 0;
      Last_Join_Attempt = 0;
      Last_Successfull_Join = HAL_GetTick();
      HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_SET);
      /* USER CODE END 0 */
    }
    else
    {
      APP_DBG("Startup failed, attempting again after a short delay (%d
ms)", APP_ZIGBEE_STARTUP_FAIL_DELAY);
      zigbee_app_info.join_delay = HAL_GetTick() +
APP_ZIGBEE_STARTUP_FAIL_DELAY;
      /* USER CODE BEGIN 1 */

      /* USER CODE END 1 */
    }
  }
  if((zigbee_app_info.join_delay - Last_Join_Attempt) < 15000){
    /* If Network forming/joining was not successful reschedule the current task to retry the process */
    if (zigbee_app_info.join_status != ZB_STATUS_SUCCESS)
    {
      UTIL_SEQ_SetTask(1U << CFG_TASK_ZIGBEE_NETWORK_FORM,
CFG_SCH_PRIO_0);
      //strcpy(my_board.address, "uart");
    }
  /* USER CODE BEGIN NW_FORM */
    else
    {
      zigbee_app_info.init_after_join = false;

      /* Assign ourselves to the group addresses */
      APP_ZIGBEE_ConfigGroupAddr();

      /* Since we're using group addressing (broadcast), shorten the broadcast timeout */
      uint32_t bcast_timeout = 3;
      ZbNwkSet(zigbee_app_info.zb,
ZB_NWK_NIB_ID_NetworkBroadcastDeliveryTime, &bcast_timeout,
sizeof(bcast_timeout));
```

```
    }
  }
  else
  {
    Zigbee_Connected = 0;
    Board_Data[Board_Number].status = 0;
    Join_Limit_Exceeded = 1;
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_SET);
    /* USER CODE END NW_FORM */
  }
}
```

Χρονισμοί και Δρομολόγηση

Server

Ο Server αφήνει αρχικά ένα χρονικό παράθυρο 5000ms στο οποίο δεν εκτελεί καμία συνάρτηση ώστε να αρχικοποιηθεί το ασύρματο δίκτυο χωρίς παρεμβολές. Αφού περάσει το χρονικό πλαίσιο αρχικοποίησης εκτελεί τις εξής λειτουργίες :

Σε κάθε διακοπή του Timer1:

– Υπολογίζει την θερμοκρασία της πλακέτας η οποία προέρχεται από το πρώτο κανάλι του ADC.
– Ξεκινάει την μέτρηση του ADC για τα 4 κανάλια που είναι συνδεδεμένα με τις αναλογικές εισόδους της πλακέτας και αποθηκεύει τις μετρήσεις αυτόματα μέσω DMA.
– «Διαβάζει» και αποθηκεύει την τιμή των ψηφιακών εισόδων της πλακέτας.
– Εκτελεί την Transmit_Uart() (sequencer), η οποία στέλνει τις πληροφορίες της πλακέτας μέσω UART σε οποιαδήποτε άλλη πλακέτα είναι συνδεδεμένη ενσύρματα.

Σε κάθε διακοπή του Timer2:

– Εκτελεί την Transmit_Heartbeat() (sequencer), η οποία στέλνει ,μέσω ZigBee, σε όλους τους router/clients ένα μήνυμα ώστε να επιβεβαιώνουν την σύνδεση τους στο ασύρματο δίκτυο.

Σε κάθε διακοπή του Timer16:

– Αυξάνει έναν counter κάθε 30sec. Όταν ο counter φτάνει 5 δηλαδή κάθε δυόμιση λεπτά εκτελεί την APP_ZIGBEE_AllowJoining() (sequencer), η οποία ανανεώνει τον χρόνο εντός του οποίου μπορούν οι routers/clients να συνδεθούν στο δίκτυο

που δημιούργησε ο server.

Σε κάθε διακοπή του Timer17:

– Εκτελεί την Transmit_Zigbee() (sequencer), η οποία στέλνει τις πληροφορίες της πλακέτας μέσω ZigBee.

Router/Client

Σε κάθε διακοπή του Timer1:

– Υπολογίζει την θερμοκρασία της πλακέτας η οποία προέρχεται από το πρώτο κανάλι του ADC.
– Ξεκινάει την μέτρηση του ADC για τα 4 κανάλια που είναι συνδεδεμένα με τις αναλογικές εισόδους της πλακέτας και αποθηκεύει τις μετρήσεις αυτόματα μέσω DMA.
– «Διαβάζει» και αποθηκεύει την τιμή των ψηφιακών εισόδων της πλακέτας.
– Ανάλογα με την λογική στάθμη των DIP switches ενεργοποιεί ορισμένες λειτουργίες της πλακέτας όπως Middleman mode ή No Rejoin.
– Εκτελεί την Transmit_Uart() (sequencer), η οποία στέλνει τις πληροφορίες της πλακέτας μέσω UART σε οποιαδήποτε άλλη πλακέτα είναι συνδεδεμένη ενσύρματα.
– Σε περίπτωση που η πλακέτα είναι σε λειτουργία Middleman δηλαδή λειτουργεί ως ενδιάμεσος κόμβος του ασύρματου και ενσύρματου υποδικτύου εκτελεί την Transmit_Zigbee() (sequencer), η οποία στέλνει τις πληροφορίες της πλακέτας μέσω ZigBee.

Σε κάθε διακοπή του Timer2:

– Εκτελεί την Heartbeat()_Check (sequencer), η οποία ελέγχει αν η πλακέτα έχει λάβει το Heartbeat μήνυμα από τον server. Αν δεν λάβει μήνυμα για έναν συγκεκριμένο χρόνο , τότε αποσυνδέεται από το δίκτυο και αρχικοποιεί το network stack ώστε να προσπαθήσει να ξανασυνδεθεί. Ακόμη φροντίζει για την προσπάθεια επανασύνδεσης σε περίπτωση που δεν υπάρχει διαθέσιμο δίκτυο αλλά και στην περίπτωση που έχει προκύψει κάποιο security fail.

Σε κάθε διακοπή του Timer16:

– Εφόσον η πλακέτα είναι συνδεδεμένη στο δίκτυο και δεν ε middleman, εκτελεί την Transmit_Zigbee() (sequencer), η οποία στέλνει τις πληροφορίες της πλακέτας με zigbee.

Σε κάθε διακοπή του Timer17:

– Εφόσον η πλακέτα δεν είναι συνδεδεμένη στο δίκτυο αυξάνει έναν counter κάθε 60sec. Όταν ο counter φτάνει 2, δηλαδή κάθε δυό λεπτά εκτελεί την APP_ZIGBEE_NwkForm() (sequencer), η οποία προσπαθεί να συνδεθεί στο ασύρματο δίκτυο.

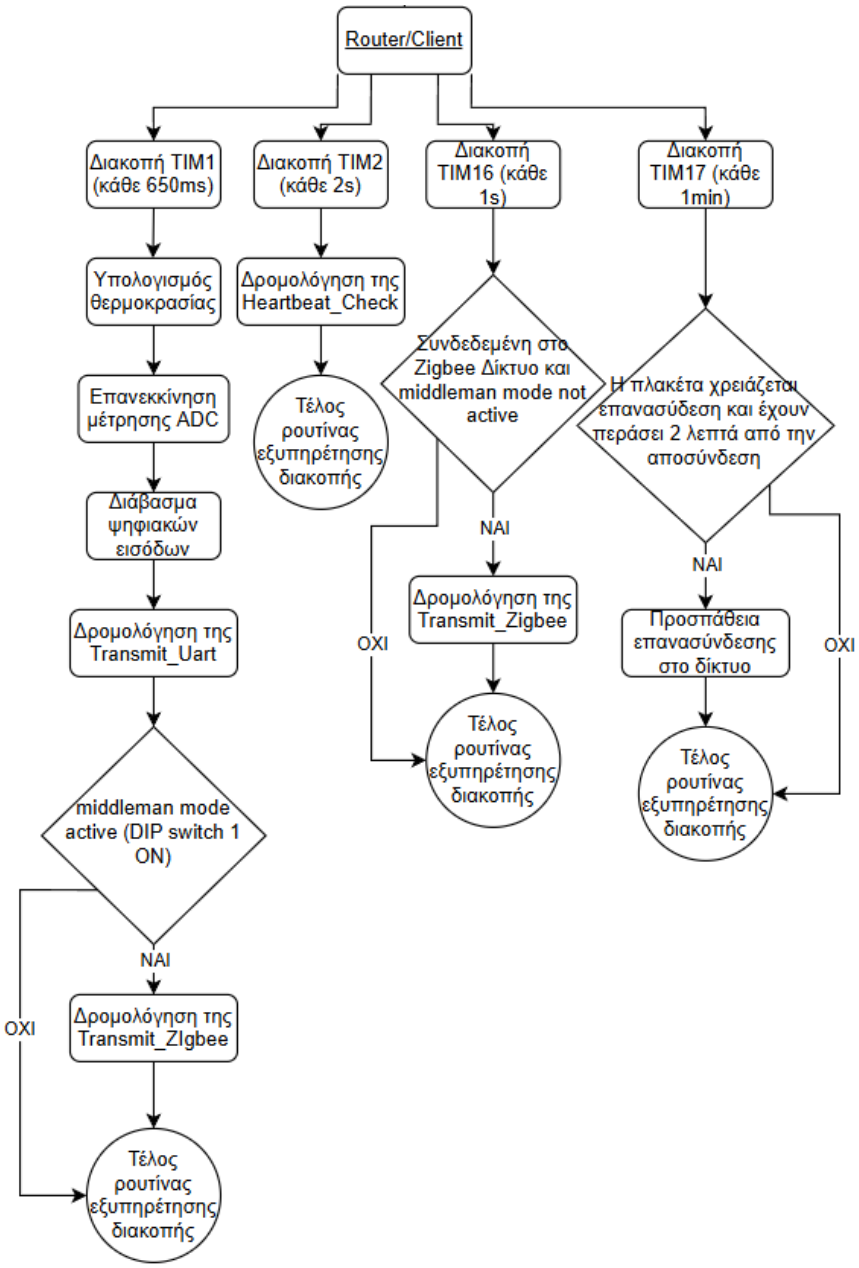
Τόσο ο Server όσο και ο router/client χρησιμοποιούν τέσσερις timers οι οποίοι έχουν ρυθμιστεί με τον ίδιο τρόπο ώστε να προκαλούν μια διακοπή όταν φτάνουν στο άνω όριο της μέτρησης τους. Η διαφοροποίηση μεταξύ των δύο υλοποιήσεων κώδικα είναι αφενός οι χρονισμοί των timers και αφετέρου οι ρουτίνες εξυπηρέτησης των διακοπών τους, οι οποίες καθορίζουν και ποιες συναρτήσεις θα καταχωρηθούν προς εκτέλεση από τον sequencer.

Όλοι οι timers έχουν Prescaler 64000. Ο prescaler διαιρεί την συχνότητα του ρολογιού ώστε να προκύψει η συχνότητα του timer, επομένως όλοι οι timers έχουν συχνότητα 1kHz αφού το ρολόι είναι χρονισμένο στα 64MHz. Ακόμη όλοι προκαλούν διακοπές όταν φτάσουν στο άνω όριο τους που ορίζει ο καταχωρητής περιόδου. Για παράδειγμα ένας timer με περίοδο 500 προκαλεί διακοπή κάθε 500*1ms = 500ms. Οι ακριβείς χρονισμοί για τον Server είναι οι εξής: TIM1->650ms, TIM2 -> 2100ms, TIM16 -> 30000ms, TIM17 -> 1000ms ενώ γα τον Client είναι οι εξής: TIM1->650ms, TIM2 -> 2000ms, TIM16 -> 1000ms, TIM17 -> 60000ms. Οι timers ενεργοποιούνται και ξεκινάνε την μέτρηση με λειτουργία διακοπής με την εξής εντολή που βρίσκεται εντός της main συνάρτησης πριν την είσοδο στο αέναο while loop: `HAL_TIM_Base_Start_IT(&htim1);`. Οι timers αρχικοποιούνται αυτόματα όταν φτάσουν στην τιμή του καταχωρητή περιόδου αφού στον ίδιο προκαλείται υπερχείλιση.

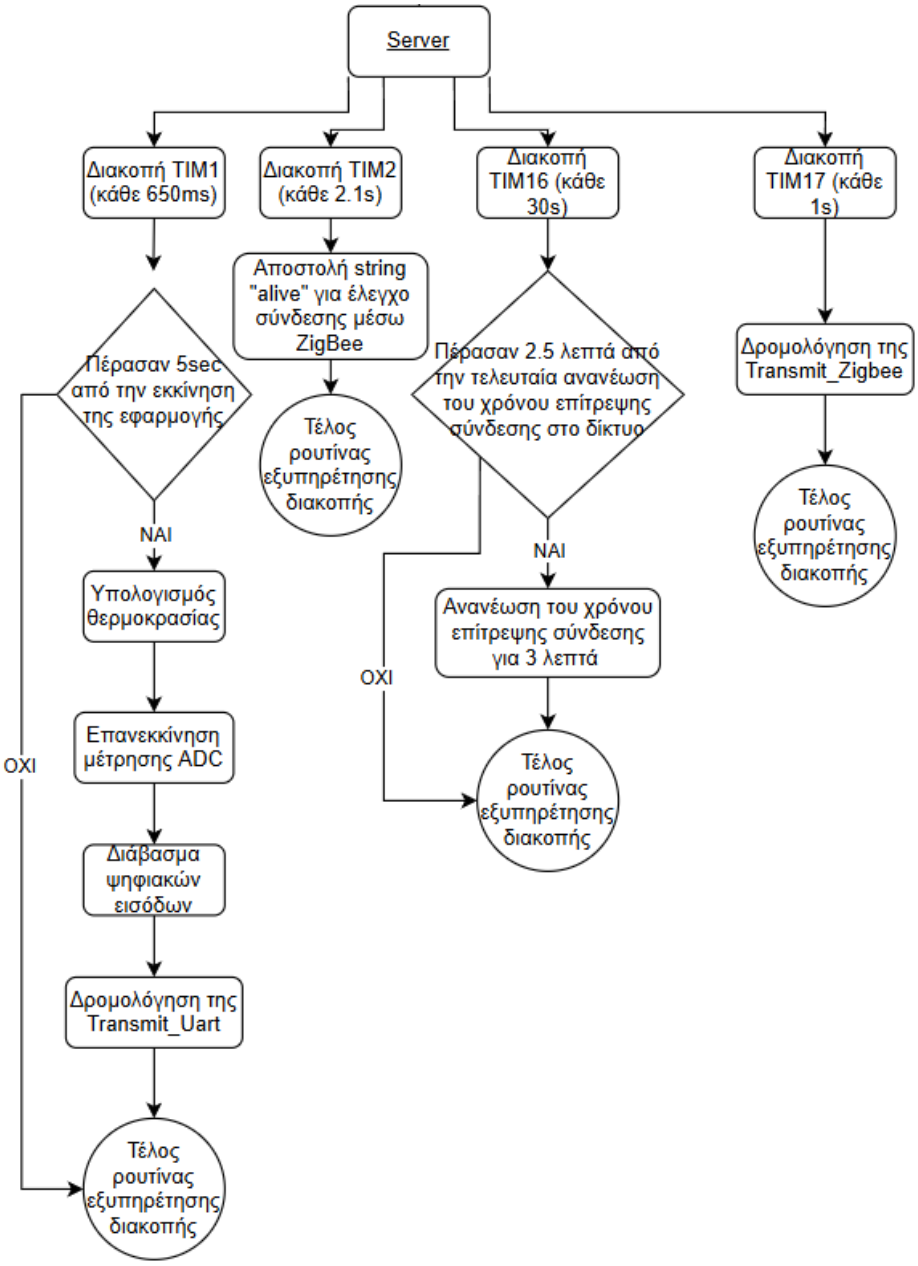
Εκτός από τις διακοπές των Timers στο σύστημα προκαλείται επίσης διακοπή **στο τέλος λήψης ενός πακέτου δεδομένων μέσω της σειριακής επικοινωνίας UART**. Η ρουτίνα εξυπηρέτησης της διακοπής αυτής καλεί την συνάρτηση Receive_Uart και επανεκκινεί την UART ώστε να είναι έτοιμη για λήψη.

Τέλος διακοπή προκαλείται και **στο τέλος της λήψης δεδομένων μέσω ZigBee.** Στην συγκεκριμένη διακοπή η Callback συνάρτηση που καλείται όπως έχει οριστεί εντός του custom long string cluster είναι η Receive_ZigBee_Cb.

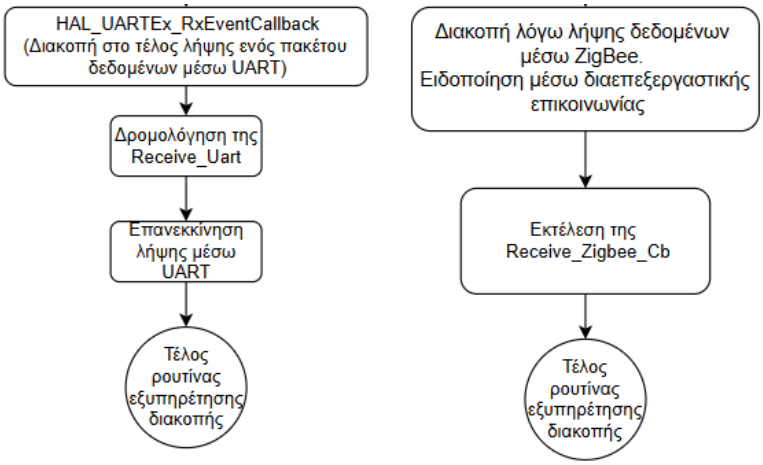
Η λειτουργίες των ρουτινών εξυπηρέτησης διακοπής παρουσιάζεται συνοπτικά με τα παρακάτω διαγράμματα :



Εικόνα 35:Διάγραμμα Ροής ρουτινών εξυπηρέτησης διακοπών router/client



Εικόνα 36:Διάγραμμα Ροής ρουτινών εξυπηρέτησης διακοπών server



Εικόνα 37:Διάγραμμα Ροής ρουτινών εξυπηρέτησης διακοπών

Η υλοποίηση των ρουτινών εξυπηρέτησης των διακοπών εξαιτίας των Timers είναι η εξής:

Για τον Server:

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  /* Prevent unused argument(s) compilation warning */
  UNUSED(htim);
  static int half_minutes_counter = 0;
  CurrTime = HAL_GetTick();
  if(CurrTime > 5000){
    if(htim == &htim1){
      Calculate_Temp();
      HAL_ADC_Start_DMA(&hadc1, &Board_Data[Board_Number].adc[0], 4);
      HAL_UARTEx_ReceiveToIdle_IT(&huart1, rxdata, 50);
      Board_Data[Board_Number].digital_in[0] =
HAL_GPIO_ReadPin(Digital_IN_1_GPIO_Port, Digital_IN_1_Pin);
      Board_Data[Board_Number].digital_in[1] =
HAL_GPIO_ReadPin(Digital_IN_2_GPIO_Port, Digital_IN_2_Pin);
      Board_Data[Board_Number].digital_in[2] =
HAL_GPIO_ReadPin(Digital_IN_3_GPIO_Port, Digital_IN_3_Pin);
      UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_UART,CFG_SCH_PRIO_0);
    }
    if(htim == &htim17){
      UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_ZIGBEE,CFG_SCH_PRIO_0);
    }
    if(htim == &htim2){
      UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_HEARTBEAT,CFG_SCH_PRIO_0);
    }
    if(htim == &htim16){
      half_minutes_counter ++;
      if(half_minutes_counter == 5){
        UTIL_SEQ_SetTask(1U << CFG_ALLOW_JOIN, CFG_SCH_PRIO_0);
        half_minutes_counter = 0;
      }
    }
  }
}
```

Για τον Router/Client:

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  /* Prevent unused argument(s) compilation warning */
  UNUSED(htim);
  static int minutes_counter = 0;
  if(htim == &htim1){
    Calculate_Temp();
    HAL_ADC_Start_DMA(&hadc1, &Board_Data[Board_Number].adc[0], 4);
    HAL_UARTEx_ReceiveToIdle_IT(&huart1, rxdata, 50);
    Board_Data[Board_Number].digital_in[0] =
HAL_GPIO_ReadPin(Digital_IN_1_GPIO_Port, Digital_IN_1_Pin);
    Board_Data[Board_Number].digital_in[1] =
HAL_GPIO_ReadPin(Digital_IN_2_GPIO_Port, Digital_IN_2_Pin);
```

```
    Board_Data[Board_Number].digital_in[2] =
HAL_GPIO_ReadPin(Digital_IN_3_GPIO_Port, Digital_IN_3_Pin);
    Middleman = !(HAL_GPIO_ReadPin(DIP_SW1_GPIO_Port, DIP_SW1_Pin));
    Test_Rejoin = !(HAL_GPIO_ReadPin(DIP_SW2_GPIO_Port, DIP_SW2_Pin));
    No_Rejoin = !(HAL_GPIO_ReadPin(DIP_SW3_GPIO_Port, DIP_SW3_Pin));
    UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_UART,CFG_SCH_PRIO_0);
    if(Middleman){
      UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_ZIGBEE,CFG_SCH_PRIO_0);
    }
  }
  if(htim == &htim2){
    UTIL_SEQ_SetTask(1U << CFG_HEARTBEAT_CHECK,CFG_SCH_PRIO_0);
  }
  if(htim == &htim16){
    if(Zigbee_Connected & (!Middleman)){
      UTIL_SEQ_SetTask(1U << CFG_TRANSMIT_ZIGBEE,CFG_SCH_PRIO_0);
    }
  }
  if(htim == &htim17){
    if(Need_Rejoin & !No_Rejoin){
      minutes_counter ++;
      if(minutes_counter == 2){
        Last_Join_Attempt = HAL_GetTick();
        UTIL_SEQ_SetTask(1U << CFG_TASK_ZIGBEE_NETWORK_FORM,
CFG_SCH_PRIO_0);
        Need_Rejoin  = 0;
        minutes_counter = 0;
      }
    }
  }
}
```

Ανάλυση Συναρτήσεων Βασικής Λειτουργίας

Κοινές Συναρτήσεις για Server και Client

Receive_Uart & Transmit_Uart

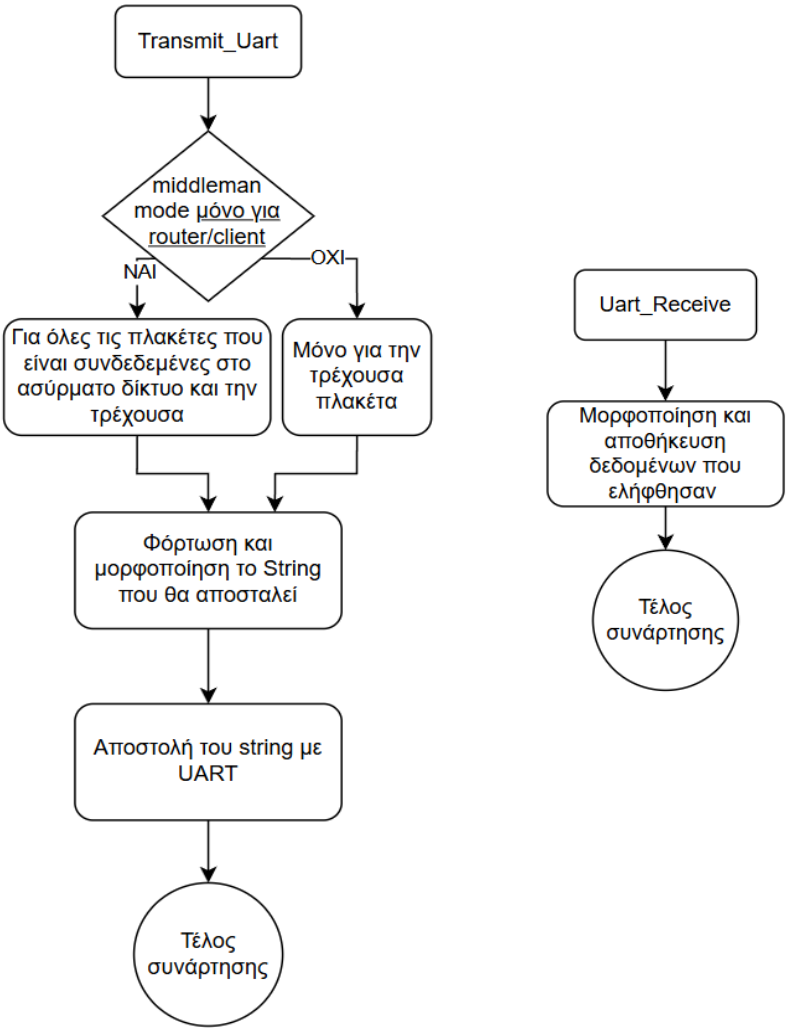
Η **void Receive_Uart(void)** και η **void Transmit_Uart(void)** είναι υπεύθυνες για την αποστολή και λήψη δεδομένων με χρήση της Uart η οποία έπειτα με την χρήση του ολοκληρωμένου THVD1420DR μετατρέπεται σε RS485.

Η Receive Uart «σετάρεται» για εκτέλεση μέσω του sequencer κάθε φορά λαβάνεται ένα πακέτο δεδομένων αφού χρησιμοποιούμε την συνάρτηση **HAL_UARTEx_ReceiveToIdle_IT** για την λήψη δεδομένων, η οποία λαμβάνει μια ποσότητα δεδομένων σε λειτουργία διακοπής μέχρι είτε να ληφθεί ο αναμενόμενος αριθμός δεδομένων είτε να συμβεί ένα γεγονός IDLE (αδράνειας). Αφού μεταφέρει

τα δεδομένα σε έναν προσωρινό buffer καλεί την Format_Transmit_Data και αδειάζει τους buffers.

Η **void Transmit_Uart**(**void**) «σετάρεται» για εκτέλεση μέσω του sequencer από τους Timers. Καλεί την Format_Transmit_Data χρησιμοποιώντας έναν προσωρινό buffer και έπειτα στέλνει τα δεδομένα μέσω της uart. Στο τέλος αδειάζει τον προσωρινό buffer. Σε περίπτωση που η συγκεκριμένη πλακέτα βρίσκεται σε middleman mode τότε η **Transmit_Uart** είναι υπεύθυνη να στείλει με Uart τα δεδομένα όλων των πλακετών που είναι συνδεδεμένες στο ασύρματο δίκτυο. Με την μέθοδο αυτή όλες οι πλακέτες που είναι συνδεδεμένες μόνο ενσύρματα θα έχουν τα δεδομένα τόσο των ενσύρματα συνδεδεμένων πλακετών όσο και των ασύρματα συνδεδεμένων πλακετών. Η διαφορά μεταξύ του Server και του Client είναι πως ο Server δεν έχει την λειτουργία του middleman αφού είναι αυτός που δημιουργεί το δίκτυο.

Η λειτουργία των συναρτήσεων παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 38:Διάγραμμα Ροής συναρτήσεων UART

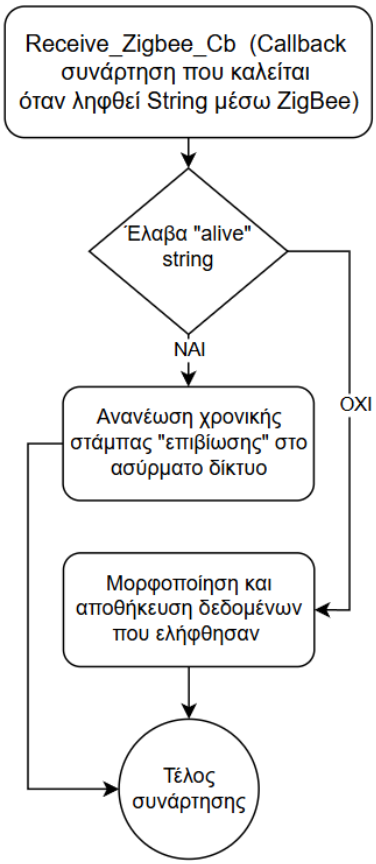
Η υλοποίηση των συναρτήσεων αυτών είναι η εξής:

```
void Receive_Uart(void)
{
  memset(received_uart_mess, 0, sizeof(received_uart_mess));
  strcpy(received_uart_mess, rxdata);
  Format_Receive_Data(received_uart_mess);
  memset(rxdata, 0, sizeof(rxdata));


}
void Transmit_Uart(void)
{
  if(!Middleman){
    memset(transmit_uart_mess, 0, sizeof(transmit_uart_mess));
    Format_Transmit_Data(transmit_uart_mess, Board_Number);
    HAL_UART_Transmit(&huart1, transmit_uart_mess,
strlen(transmit_uart_mess),100);
  }
  else{
    for(int i = 0; i < N; i++){
      memset(transmit_uart_mess, 0, sizeof(transmit_uart_mess));
      if(Board_Data[i].status == 1 | i == Board_Number){
        Format_Transmit_Data(transmit_uart_mess, i);
        HAL_UART_Transmit(&huart1, transmit_uart_mess,
strlen(transmit_uart_mess),100);
      }
    }
  }
}
```

Receive_Zigbee_Cb

Η **Receive_Zigbee_Cb** είναι η Callback συνάρτηση που καλείται μέσω του Custom Long string cluster κάθε φορά που το stack λαμβάνει δεδομένα μέσω ZigBee. Παρόμοια με την Receive_Uart αφού αντιγράψει τα δεδομένα σε έναν προσωρινό Buffer καλεί την Format_Receive_Data ώστε να επεξεργαστεί το string των δεδομένων. Σε περίπτωση που ο Client λάβει το string “alive” το οποίο στέλνει ο Server κάθε 2.1 sec, ανανεώνει μια μεταβλητή με την τρέχουσα χρονική στιγμή λειτουργίας. Αυτή τη λειτουργία δεν την χρειάζεται ο Server.

Η λειτουργία της συνάρτησης παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 39:Διάγραμμα Ροής συνάρτησης Receive Zigbee

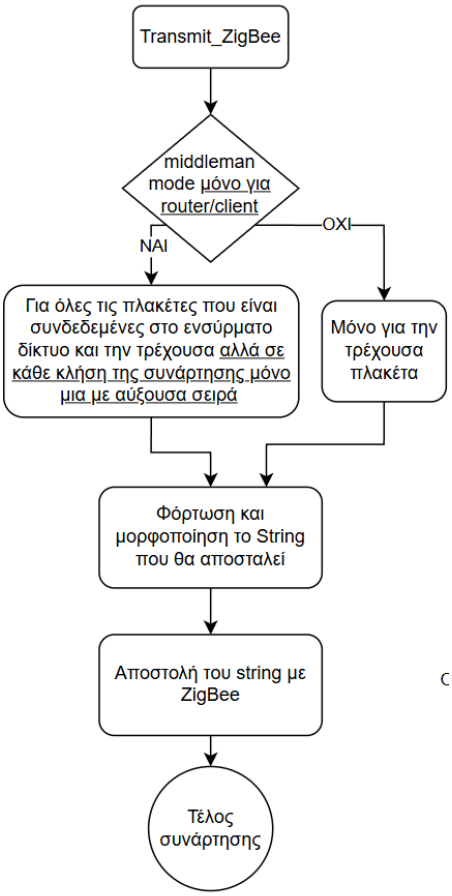
Η υλοποίηση της συνάρτησης είναι η εξής :

```
static enum ZclStatusCodeT Receive_Zigbee_Cb(struct ZbZclClusterT
*clusterPtr, struct set_custom_ls_command_req_t *cmd_req, \
        struct ZbZclAddrInfoT *src_info, void *arg)
{
  char *msg = "alive";
  memset(received_zigbee_mess, 0, sizeof(received_zigbee_mess));

  if(src_info->addr.nwkAddr != MyNwkAddress)
  {
    if(memcmp((char*)&cmd_req->string[0],(char*)&msg[0],5)!=0){
      strcpy(received_zigbee_mess, cmd_req->string);
      Format_Receive_Data(received_zigbee_mess);
    }
    else
    {
      if(!Test_Rejoin){
        LastTimeAlive = HAL_GetTick();
      }
    }
  }
  return ZCL_STATUS_SUCCESS;
}

static struct zcl_custom_ls_server_callbacks_t custom_ls_server_cb = {
               Receive_Zigbee_Cb,
};
```

Transmit_Zigbee

Η **void Transmit_Zigbee**(**void**) είναι υπεύθυνη για την αποστολή των δεδομένων ασύρματα μέσω ZigBee. Καλεί την Format_Transmit_Data χρησιμοποιώντας έναν προσωρινό buffer και έπειτα έχοντας «σετάρει» την αποστολή σε group addressing mode, καλεί την **ZbZclSet_custom_ls_ClientCommand** η οποία είναι η συνάρτηση του custom long string cluster , υπεύθυνη για την επικοινωνία με το stack του ασύρματου συνεπεξεργαστή και αποστολή των δεδομένων. Σε περίπτωση που η πλακέτα είναι σε middleman mode τότε σέ κάθε κλήση της συνάρτησης στέλνονται τα δεδομένα μιας από τις πλακέτες που είναι συνδεδεμένες μόνο στο ενσύρματο δίκτυο. Σε κάθε επόμενη κλήση της συνάρτησης στέλνονται τα δεδομένα της επόμενης πλακέτας συνδεδεμένης μόνο στο ενσύρματο δίκτυο έως ότου έχουν σταλεί τα δεδομένα όλων των πλακετών του ενσύρματου δικτύου και να εκκινήσει η αποστολή δεδομένων από την πρώτη πλακέτα κυκλικά στην επόμενη κλήση της συνάρτησης. Αυτό γίνεται διότι λόγω της ασύρματης φύσης της αποστολής υπάρχει μια καθυστέρηση της τάξης των δεκάδων ms μεταξύ των αποστολών δεδομένων. Στον Server η λειτουργία του middleman δεν υποστηρίζεται. Η διαδικασία αποστολής και λήψης string με την χρήση του εξατομικευμένου cluster παρουσιάζεται στην ενότητα *Custom Long String Cluster*

Η λειτουργία αυτής της συνάρτησης παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 40:Διάγραμμα Ροής συνάρτησης Transmit Zigbee

Η υλοποίηση της συνάρτησης είναι η εξής:

```
static void Transmit_Zigbee(void)
{
  struct ZbApsAddrT dst;
  static int i = 1;

  memset(&dst, 0, sizeof(dst));
  dst.mode = ZB_APSDE_ADDRMODE_GROUP;
  dst.endpoint = SW1_ENDPOINT;
  dst.nwkAddr = 0x0001;

  memset(transmit_zigbee_mess, 0, sizeof(transmit_zigbee_mess));
  if(Middleman){
    while(Board_Data[i].status == 1){
      if(i != Board_Number){
        i = (i + 1) % N;
        if(i == 0) i = 1;
      }
      else break;
    }
    if(Board_Data[i].status == 0 | i == Board_Number){
      Format_Transmit_Data(transmit_zigbee_mess, i);
      i = (i + 1) % N;
      if(i == 0) i = 1;
    }
  }
```

```
  else{
    Format_Transmit_Data(transmit_zigbee_mess, Board_Number);
  }

  MyNwkAddress = ZbShortAddress(zigbee_app_info.zb);

  if (ZbZclSet_custom_ls_ClientCommand(zigbee_app_info.custom_ls_client,
&dst, transmit_zigbee_mess, custom_ls_client_cb, NULL) !=
ZCL_STATUS_SUCCESS)
    {
      APP_DBG("Error, ZbZclSet_custom_ls_ClientCommand failed
(SW1_ENDPOINT)");
    }
}
```

Format_Transmit_Data & Format_Receive_Data

Η **Format_Transmit_Data** και η **Format_Receive_Data** είναι υπεύθυνες για την μορφοποίηση των δεδομένων πριν την αποστολή και μετά την λήψη. Στην πράξη λειτουργούν σαν ένα μικρό custom πρωτόκολλο επικοινωνίας. Τα δεδομένα που στέλνονται είναι οι τιμές των 4 adcs οι λογικές στάθμες των 3 ψηφιακών εισόδων, η θερμοκρασία κάθε πλακέτας καθώς και το status της (1 αν είναι συνδεδεμένη στο ασύρματο δίκτυο αλλιώς 0). Τα δεδομένα στέλνονται σε ένα string και χωρίζονται με κόμμα. Οι συναρτήσεις αυτές είναι υπεύθυνες να φτιάξουν το string αυτό βασιζόμενες στις τιμές των μεταβλητών της δομής Board_Data και να ανανεώσουν τις τιμές των μεταβλητών αυτών βασιζόμενες στο string που λαμβάνουν από μια άλλη πλακέτα.

Η υλοποίηση των συναρτήσεων είναι η εξής:

```
void Format_Transmit_Data(char* transmit, int Board)
{
  char temp_transmit[256] = {0};
  char adc1[6] = {0};
  char adc2[6] = {0};
  char adc3[6] = {0};
  char adc4[6] = {0};
  char di1[2] = {0};
  char di2[2] = {0};
  char di3[2] = {0};
  char temp[4] = {0};
  char stat[2] = {0};

  memset(transmit ,0, 256);
  memset(Board_Address, 0, sizeof(Board_Address));
  sprintf(Board_Address, "%u" , Board);

  sprintf(adc1, "%ld" ,Board_Data[Board].adc[0]);
  sprintf(adc2, "%ld" ,Board_Data[Board].adc[1]);
  sprintf(adc3, "%ld" ,Board_Data[Board].adc[2]);
  sprintf(adc4, "%ld" ,Board_Data[Board].adc[3]);
```

```
  sprintf(di1, "%ld" ,Board_Data[Board].digital_in[0]);
  sprintf(di2, "%ld" ,Board_Data[Board].digital_in[1]);
  sprintf(di3, "%ld" ,Board_Data[Board].digital_in[2]);
  sprintf(temp, "%d" ,Board_Data[Board].temp);
  sprintf(stat, "%ld" ,Board_Data[Board].status);

  strcat(temp_transmit, Board_Address);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, adc1);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, adc2);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, adc3);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, adc4);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, di1);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, di2);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, di3);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, temp);
  strcat(temp_transmit, ",");
  strcat(temp_transmit, stat);
  strcpy(transmit, temp_transmit);

  memset(temp_transmit ,0, sizeof(temp_transmit));
}

void Format_Receive_Data(char* receive)
{
  char* address = NULL;
  char* adc1 = NULL;
  char* adc2 = NULL;
  char* adc3 = NULL;
  char* adc4 = NULL;
  char* di1 = NULL;
  char* di2 = NULL;
  char* di3 = NULL;
  char* temp = NULL;
  char* stat = NULL;

  int addr = 0;
  int adc_temp[4] = {0};

  address =strtok(receive, ",");
  if(address != NULL)
  {
    adc1 = strtok(NULL, ",");
  }
  if(adc1 != NULL)
  {
    adc2 = strtok(NULL, ",");
  }
  if(adc2 != NULL)
  {
```

```
    adc3 = strtok(NULL, ",");
  }
  if(adc3 != NULL)
  {
    adc4 = strtok(NULL, ",");
  }
  if(adc4 != NULL)
  {
    di1 = strtok(NULL, ",");
  }
  if(di1 != NULL)
  {
    di2 = strtok(NULL, ",");
  }
  if(di2 != NULL)
  {
    di3 = strtok(NULL, ",");
  }
  if(di3 != NULL)
  {
    temp = strtok(NULL, ",");
  }
  if(temp != NULL)
  {
    stat = strtok(NULL, ",");
  }
  addr = atoi(address);
  adc_temp[0] = atoi(adc1);
  adc_temp[1] = atoi(adc2);
  adc_temp[2] = atoi(adc3);
  adc_temp[3] = atoi(adc4);
  if(!((addr == Board_Number) | ((addr == 0) & (adc_temp[0] == 0) &
(adc_temp[1] == 0) & (adc_temp[2] == 0) & (adc_temp[3] == 0)))){
    Board_Data[addr].adc[0] = adc_temp[0];
    Board_Data[addr].adc[1] = adc_temp[1];
    Board_Data[addr].adc[2] = adc_temp[2];
    Board_Data[addr].adc[3] = adc_temp[3];
    Board_Data[addr].digital_in[0] = atoi(di1);
    Board_Data[addr].digital_in[1] = atoi(di2);
    Board_Data[addr].digital_in[2] = atoi(di3);
    Board_Data[addr].temp = atoi(temp);
    Board_Data[addr].status = atoi(stat);
  }
  CurrTime = HAL_GetTick();
}
```

Συναρτήσεις αποκλειστικά του Server

Transmit_Heartbeat

Η **void Transmit_Heartbeat(void)** είναι υπεύθυνη για την αποστολή του “alive” string σε όλες τις συνδεδεμένες στο δίκτυο. Καλείται μέσω των Timers κάθε 2.1 sec όπως αναφέρθηκε και παραπάνω. Κάνει χρήση της

ZbZclSet_custom_ls_ClientCommand όπως ακριβώς και η Transmit_ZigBee, η οποία ορίζεται εντός του Custom Long String cluster.

Η υλοποίηση της συνάρτησης είναι η εξής:

```
static void Transmit_Heartbeat(void)
{
  if (zigbee_app_info.join_status == ZB_STATUS_SUCCESS)
  {
    struct ZbApsAddrT dst;

    ZbNwkGet(zigbee_app_info.zb, ZB_NWK_NIB_ID_PermitJoinCounter,
&remaining_join_permit_seconds, 1);

    memset(&dst, 0, sizeof(dst));
    dst.mode = ZB_APSDE_ADDRMODE_GROUP;
    dst.endpoint = SW1_ENDPOINT;
    dst.nwkAddr = 0x0001;

    if (ZbZclSet_custom_ls_ClientCommand(zigbee_app_info.custom_ls_client,
&dst, heartbeat_mess, custom_ls_client_cb, NULL) != ZCL_STATUS_SUCCESS)
    {
      APP_DBG("Error, ZbZclSet_custom_ls_ClientCommand failed
(SW1_ENDPOINT)");
    }
  }
}
```

APP_ZIGBEE_AllowJoining

Η **void APP_ZIGBEE_AllowJoining(void)** είναι υπεύθυνη για την ανανέωση του χρόνου επίτρεψης σύνδεσης στο δίκτυο από κάποιον client. Καλείται μέσω των timers κάθε 2.5min και ανανεώνει τον χρόνο για APP_ZIGBEE_PERMIT_JOIN_DURATION = 180ms. Αυτό πρακτικά σημαίνει πως ο Server επιτρέπει μόνιμα την σύνδεση μιας πλακέτας στο δίκτυο του. Ο λόγος που χρειάζεται αυτή η συνάρτηση είναι γιατί ο καταχωρητής που αποθηκεύει τον χρόνο επίτρεψης είναι 8-bit με αποτέλεσα να χρειάζεται ανανέωση.[36]

Η υλοποίηση της συνάρτησης είναι η εξής:

```
void APP_ZIGBEE_AllowJoining(void)
{
  struct ZbNlmePermitJoinReqT req = { .permitDuration =
APP_ZIGBEE_PERMIT_JOIN_DURATION };
  struct ZbNlmePermitJoinConfT conf;
  ZbBdbSetEndpointStatus(zigbee_app_info.zb,
ZB_BDB_COMMISS_STATUS_IN_PROGRESS, ZB_ENDPOINT_BCAST);
  ZbNlmePermitJoinReq(zigbee_app_info.zb, &req, &conf);
  if (conf.status != ZB_STATUS_SUCCESS) {
    APP_DBG("ZbNlmePermitJoinReq failed: %d", conf.status);
  } else {
```

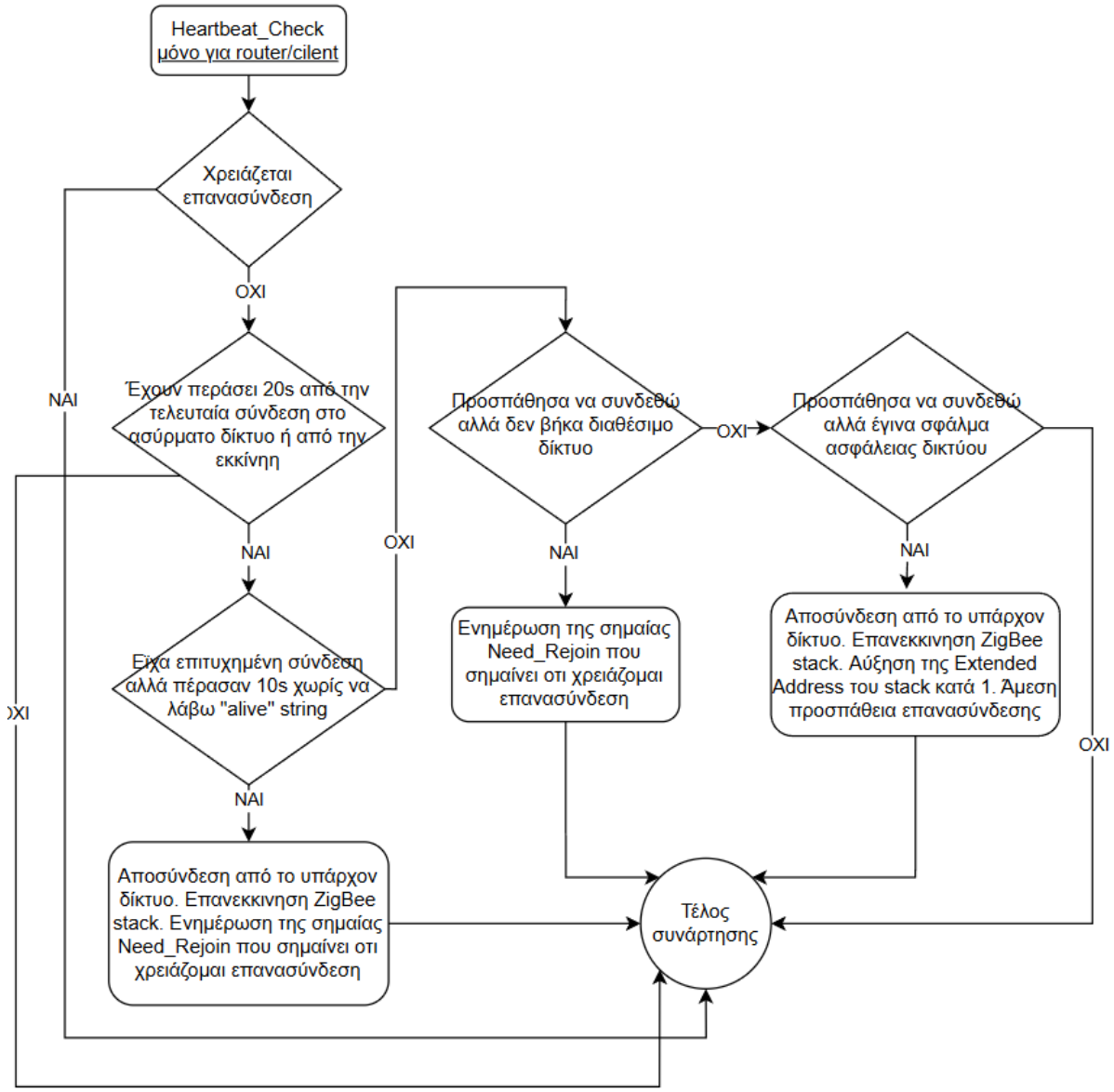
```
    APP_DBG("ZbNlmePermitJoinReq succeeded");
    ZbBdbSetEndpointStatus(zigbee_app_info.zb,
ZB_BDB_COMMISS_STATUS_SUCCESS, ZB_ENDPOINT_BCAST);
  }
}
```

Συναρτήσεις αποκλειστικά του Client

Heartbeat_Check

Η **void Heartbeat_Check** (**void**) είναι υπεύθυνη για την υλοποίηση του Heartbeat μηχανισμού. Καλείται μέσω των timers και πρακτικά ελέγχει κατά πόσο είναι ακόμα συνδεδεμένη η πλακέτα στο ασύρματο δίκτυο η όχι. Η συγκεκριμένη συνάρτηση έχει επίδραση αφού πρώτα περάσουν 20sec από την σύνδεση στο δίκτυο. Σε περίπτωση που η πλακέτα δεν λάβει το “alive” string για πάνω από 10sec τότε θεωρεί ότι έχει αποσυνδεθεί από το δίκτυο και επανεκκινεί το ZigBee stack «σετάρωντας» και το Need_Rejoin flag ώστε να προσπαθήσει να συνδεθεί σε κάποιο δίκτυο εκ νέου αυτόματα. Την προσπάθεια επανασύνδεσης κάνει η APP_ZIGBEE_NwkForm η οποία εκτελείται μέσω των Timers κάθε 2 min. Σε περίπτωση που η πλακέτα δεν εντοπίσει κάποιο διαθέσιμο δίκτυο εκτελεί την ίδια ενέργεια χωρίς όμως να χρειάζεται να επανεκκινήσει το stack. Σε περίπτωση που αντιμετωπίσει κάποιο Security Fail τότε αλλάζει το extended address της συσκευής εφόσον το προηγούμενο δεν μπορεί πλέον να συνδεθεί με τον Server, επανεκκινεί το stack και προσπαθεί να ξανασυνδεθεί στο δίκτυο. [36]

Η λειτουργία αυτής της συνάρτησης παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 41:Διάγραμμα Ροής συνάρτησης Heartbeat_Check

Η υλοποίηση της συνάρτησης είναι η εξής:

```
void Heartbeat_Check (void)
{
  static int retry_limit = 6300;
  static enum ZbStatusCodeT status;
  CurrTime = HAL_GetTick();
  if(Need_Rejoin){
    LastTimeAlive = HAL_GetTick();
  }
  if((CurrTime - Last_Successfull_Join) > 20000){
    if(zigbee_app_info.join_status == ZB_STATUS_SUCCESS){
      if(Middleman){
        retry_limit = 15000;
      }
      else retry_limit = 10000;
      if((CurrTime - LastTimeAlive) > retry_limit){
        status = ZbLeaveReq(zigbee_app_info.zb, leaveCallback, NULL);
        if (status == ZB_STATUS_SUCCESS){
```

```
            Last_Disconnect = HAL_GetTick();
            Zigbee_Connected = 0;
            Board_Data[Board_Number].status = 0;
            zigbee_app_info.join_status = 1;
            Need_Rejoin = 1;
            ZbReset(zigbee_app_info.zb);
          }
          HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_SET);
        }
      }
      if(zigbee_app_info.join_status == ZB_NWK_STATUS_NO_NETWORKS){
        if(Join_Limit_Exceeded){
          zigbee_app_info.join_delay = 0;
          Join_Limit_Exceeded = 0;
          Need_Rejoin = 1;
        }
      }
      if(zigbee_app_info.join_status == ZB_APS_STATUS_SECURITY_FAIL){
        zigbee_app_info.join_status = 1;
        zigbee_app_info.join_delay = 0;
        Need_Rejoin = 1;
        Join_Limit_Exceeded = 0;
        ZbReset(zigbee_app_info.zb);
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_RESET);
        MyExtAddress = ZbExtendedAddress(zigbee_app_info.zb);
        ZbLeaveReq(zigbee_app_info.zb, leaveCallback, NULL);
        ZbReset(zigbee_app_info.zb);
        ZbChangeExtAddr(zigbee_app_info.zb, MyExtAddress + 1);
        MyExtAddress = ZbExtendedAddress(zigbee_app_info.zb);
        UTIL_SEQ_SetTask(1U << CFG_TASK_ZIGBEE_NETWORK_FORM,
CFG_SCH_PRIO_0);
      }
    }
}
```

Custom Long String Cluster

Το Custom Long String Cluster είναι ένα προσαρμοσμένο ZigBee cluster που υλοποιήθηκε με στόχο την αποστολή και λήψη μεγάλων συμβολοσειρών (strings) μεταξύ συσκευών. [37] Αποτελείται από δύο κύριες λειτουργίες: την αποστολή εντολών από τον client και την επεξεργασία αυτών των εντολών από τον server. Ο client στέλνει εντολές χρησιμοποιώντας την ZbZclSet_custom_ls_ClientCommand. Η συνάρτηση αυτή δημιουργεί και αποστέλλει ένα μήνυμα που περιέχει τη συμβολοσειρά. Η **Transmit_Zigbee** όπως αναφέραμε και παραπάνω χρησιμοποιεί την συνάρτηση αυτή αφού πρώτα έχει μορφοποιήσει το string προς αποστολή.

Τα μηνύματα που αποστέλλονται από τον client περιλαμβάνουν το μήκος της συμβολοσειράς και τη συμβολοσειρά ως payload. Το πλαίσιο του μηνύματος

περιλαμβάνει τα εξής στοιχεία. Το Frame Control, το οποίο περιέχει σημαίες οι οποίες καθορίζουν τον τύπο του frame(Cluster-specific frame), την κατεύθυνση του frame , αν είναι απαραίτητη η απάντηση από τον παραλήπτη κλπ. Το Sequence Number, το οποίο είναι ένας μοναδικός αριθμός που χρησιμοποιείται για την αναγνώριση και παρακολούθηση του μηνύματος. Το Command ID το οποίο είναι αναγνωριστικό της εντολής που καθορίζει ποια λειτουργία θα εκτελεστεί από τον παραλήπτη.

Για παράδειγμα, ένα μήνυμα μπορεί να έχει την εξής δομή:

Frame Control	Sequence Number	Command ID	Payload Length	Payload
0x18	0x01	0x00	0x000A	“Hello ZigBee”

Πίνακας 5:Δομή μηνύματος του Custom Long String Cluster

Η ασφάλεια των μηνυμάτων εξασφαλίζεται μέσω της κρυπτογράφησης AES με 128-bit κλειδί. Αυτό επιτυγχάνεται με την ενεργοποίηση της επιλογής ZB_APSDE_DATAREQ_TXOPTIONS_SECURITY στις επιλογές μετάδοσης (txOptions). Έτσι, τα μηνύματα είναι κρυπτογραφημένα και προστατευμένα από μη εξουσιοδοτημένη πρόσβαση.

Ο server λαμβάνει και επεξεργάζεται τις εντολές που στέλνει ο client μέσω της custom_ls_command_handler συνάρτησης. Όταν ο server λαμβάνει ένα μήνυμα, περνάει από τα διάφορα επίπεδα του ZigBee stack (PHY, MAC, NWK, APS) και αποκρυπτογραφείται αν είναι κρυπτογραφημένο. Το ZCL επίπεδο αναγνωρίζει το cluster και καλεί τον κατάλληλο χειριστή εντολών (custom_ls_command_handler). Ο χειριστής εντολών επεξεργάζεται το μήνυμα και καλεί την αντίστοιχη callback συνάρτηση που έχει οριστεί στον server. Στην περίπτωση της συγκεκριμένης εφαρμογής η συνάρτηση callback που καλείται είναι η **Receive_Zigbee_Cb**

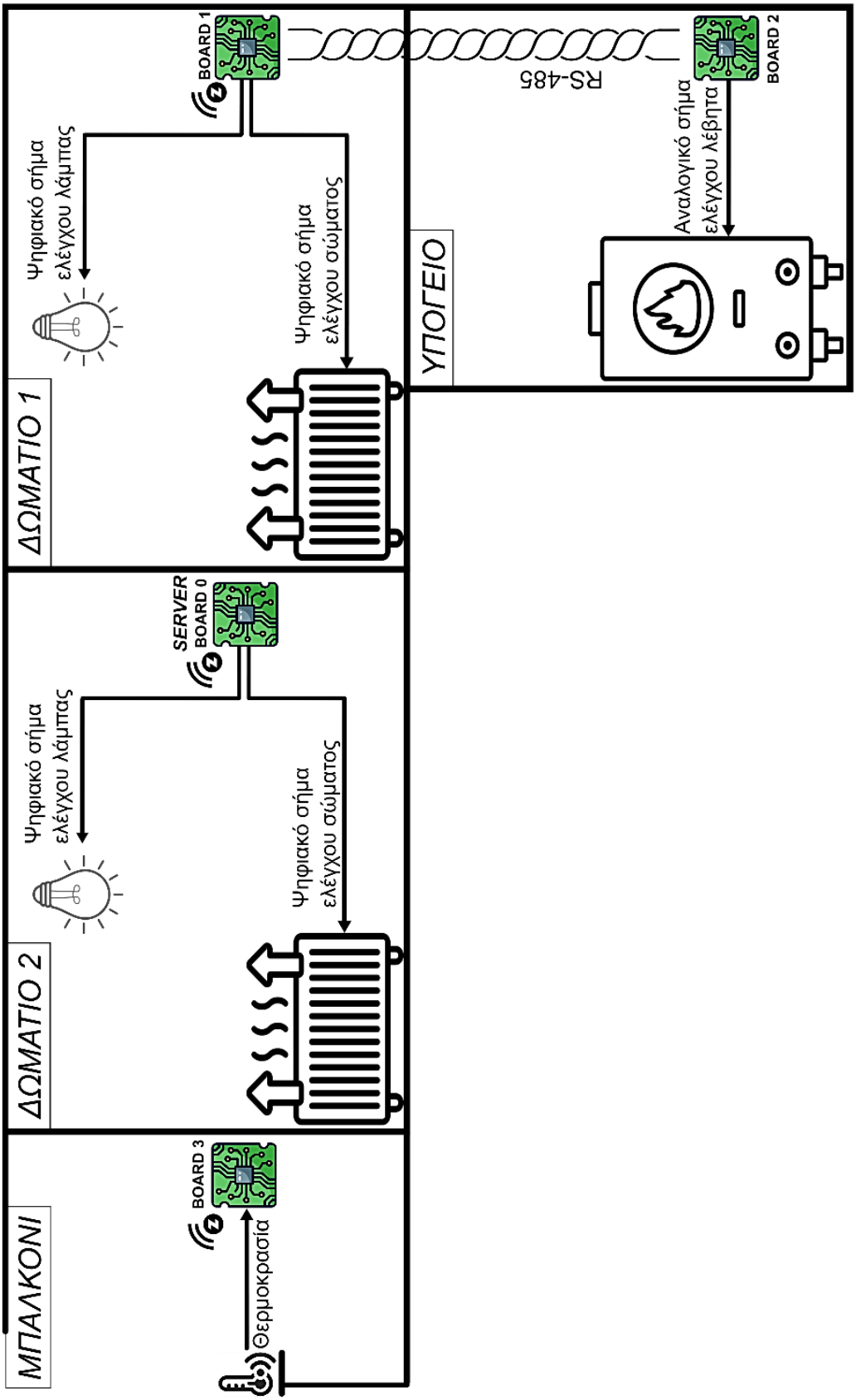
Συνοπτικά αυτό το cluster επιτρέπει την ασφαλή και αξιόπιστη επικοινωνία μεγάλων συμβολοσειρών μεταξύ ZigBee συσκευών, με δυνατότητα κρυπτογράφησης και επιβεβαίωσης παραλαβής.

Ο κώδικας του custom long string cluster βρίσκεται στον ίδιο φάκελο που βρίσκεται και ο κώδικας της Zigbee εφαρμογής, δηλαδή /Project_Name/STM32_WPAN/App. Το αρχείο zcl_custom_ls_client.c περιέχει τις βασικές λειτουργίες αποστολής δεδομένων ενώ το αρχείο zcl_custom_ls_server.c περιέχει τις βασικές λειτουργίες λήψης δεδομένων.

5 Σύστημα Προσαρμοστικής Ενεργειακής Διαχείρισης Κτηρίου

5.1 Δίκτυο Συστήματος

Το Δίκτυο Συστήματος αποτελείται από 4 πλακέτες εκ των οποίων οι 3 ανήκουν στο ασύρματο υποδίκτυο ZigBee και η τέταρτη είναι καλωδιακά συνδεδεμένη με μια από τις υπόλοιπες η οποία στο συγκεκριμένο δίκτυο λειτουργεί και ως middleman, διαμεσολαβητής δηλαδή για την μετάδοση πληροφοριών του ασύρματου υποδικτύου στο ενσύρματο και το αντίστροφο. Το δίκτυο συστήματος καθώς και εποπτικά η εφαρμογή παρουσιάζεται στην παρακάτω εικόνα:



Εικόνα 42:Δίκτυο συστήματος εφαρμογής ενεργειακής διαχείρισης κτηρίου

5.2 Περιγραφή Εφαρμογής

Για την συγκεκριμένη εφαρμογή χρησιμοποιούμε 4 πλακέτες συνδεδεμένες όπως παρουσιάζεται παραπάνω. Η πλακέτα νούμερο 1 βρίσκεται στο δωμάτιο νούμερο 1 και συνδέεται ενσύρματα με την πλακέτα με το νούμερο 2 που τοποθετείται στο υπόγειο. Η πλακέτα του υπογείου παράγει μια αναλογική τάση η οποία δίνεται ως είσοδος σε έναν λέβητα που βρίσκεται επίσης στο υπόγειο και θερμαίνει το νερό. Η αναλογική τάση που παράγει είναι αντιστρόφως ανάλογη με την θερμοκρασία που μετράει η πλακέτα νούμερο 3 που βρίσκεται στο μπαλκόνι του σπιτιού και επηρεάζει την απόδοση του λέβητα, ώστε είτε να ζεστάνει περισσότερο το νερό είτε λιγότερο. Όσο πιο μεγάλη είναι η τάση τόσο πιο πολύ ζεσταίνει το νερό ο λέβητας.

Αυτή η τεχνολογία είναι γνωστή και συνήθως αναφέρεται ως **αντιστάθμιση θέρμανσης** (ή **καιρική αντιστάθμιση**). Είναι μια εφαρμογή που χρησιμοποιεί έναν εξωτερικό αισθητήρα θερμοκρασίας για να προσαρμόσει την απόδοση του λέβητα ή του συστήματος θέρμανσης ανάλογα με τις εξωτερικές καιρικές συνθήκες.

Ο τρόπος λειτουργίας είναι ο εξής:

- Ο εξωτερικός αισθητήρας μετρά τη θερμοκρασία του περιβάλλοντος.
- Αν η εξωτερική θερμοκρασία είναι χαμηλή, το σύστημα θέρμανσης αυξάνει την απόδοση του λέβητα για να θερμάνει περισσότερο το νερό.
- Αν η εξωτερική θερμοκρασία είναι υψηλότερη, το σύστημα ρίχνει την ισχύ του, ώστε να μην υπερθερμανθεί το νερό.

Ένα παράδειγμα με πραγματικές τιμές θα ήταν το εξής :

- Εξωτερική θερμοκρασία: -5°C

Όταν η εξωτερική θερμοκρασία είναι πολύ χαμηλή, όπως στους -5°C, το σύστημα θέρμανσης αντιλαμβάνεται ότι χρειάζεται περισσότερη θερμότητα για να κρατήσει το κτίριο ζεστό. Σε αυτή την περίπτωση, ο λέβητας μπορεί να ρυθμίσει τη θερμοκρασία του νερού στους **70-75°C** ώστε να παράγει περισσότερη θέρμανση.

- Εξωτερική θερμοκρασία: 0°C

Σε πιο ήπιες συνθήκες, όπως στους 0°C, το σύστημα καταλαβαίνει ότι δεν χρειάζεται τόσο υψηλή θερμοκρασία νερού. Ο λέβητας μπορεί να ρυθμίσει τη θερμοκρασία του νερού στους **65°C**, εξοικονομώντας ενέργεια αλλά διατηρώντας επαρκή θέρμανση.

- Εξωτερική θερμοκρασία: 10°C

Όταν η εξωτερική θερμοκρασία είναι σχετικά ήπια, όπως στους 10°C, το σύστημα μειώνει σημαντικά τη θερμοκρασία του νερού, πιθανώς στους **45-50°C**, καθώς το κτίριο χρειάζεται λιγότερη θέρμανση για να παραμείνει ζεστό.

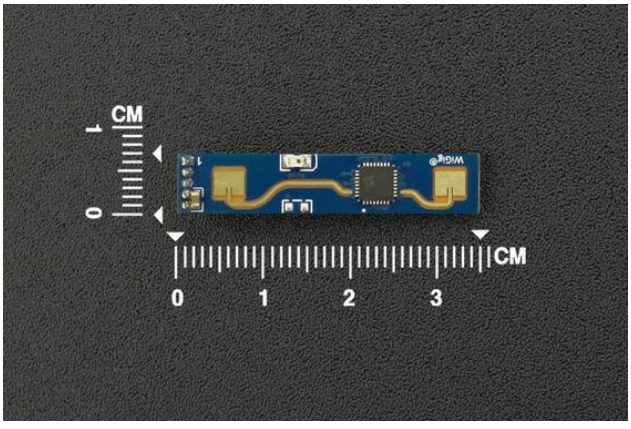
- Εξωτερική θερμοκρασία: 15°C

Σε ακόμα υψηλότερες θερμοκρασίες, όπως στους 15°C, το σύστημα μπορεί να ρυθμίσει τη θερμοκρασία του νερού στους **35-40°C** ή ακόμα και να απενεργοποιήσει τη θέρμανση αν δεν είναι απαραίτητη. Με αυτόν τον τρόπο, το σύστημα προσαρμόζεται αυτόματα στις ανάγκες θέρμανσης του κτιρίου, εξοικονομώντας ενέργεια και μειώνοντας το κόστος λειτουργίας.

Το σημαντικό στην συγκεκριμένη εφαρμογή είναι πως η πλακέτα νούμερο 3 που βρίσκεται στο μπαλκόνι επικοινωνεί ασύρματα με τις υπόλοιπες πλακέτες εντός του σπιτιού. Η μεταφορά των δεδομένων της και συγκεκριμένα της θερμοκρασίας γίνεται μέσω της πλακέτας νούμερο 1 η οποία λαμβάνει τα δεδομένα από την πλακέτα νούμερο 3 ασύρματα και λειτουργεί ως middleman ώστε να μεταφέρει τις πληροφορίες του ασύρματου υποδικτύου στο ενσύρματο.

Η πλακέτα 0 είναι ο server του δικτύου και εκτελεί τις ίδιες λειτουργίες με την πλακέτα 1 (middleman) όσο αφορά την αλληλεπίδραση με το περιβάλλον, οι οποίες είναι οι εξής:

Μέσω ενός αισθητήρα ανίχνευσης παρουσίας ανθρώπου καταλαβαίνει την εάν βρίσκεται κάποιος εντός του δωματίου που είναι τοποθετημένη, με σκοπό να χειριστεί τον φωτισμό αλλά και την θέρμανση του δωματίου. Ο αισθητήρας παράγει μια ψηφιακή έξοδο η οποία δίνεται ως είσοδος στην πλακέτα. Έτσι αν υπάρχει άνθρωπος στο δωμάτιο η πλακέτα ανιχνεύει σε μια από τις ψηφιακές εισόδους της HIGH τάση ενώ αν δεν υπάρχει ανιχνεύει LOW.[38]



Εικόνα 43:Ο Αισθητήρας Ανίχνευσης ανθρώπινης παρουσίας

Οι πλακέτες επικοινωνούν μεταξύ τους ασύρματα επομένως γνωρίζουν τόσο την ύπαρξη ανθρώπου τόσο στο δωμάτιο που βρίσκονται όσο και στο διπλανό, επομένως γνωρίζουν και την ύπαρξη ανθρώπου η όχι εντός του δωματίου αλλά και εντός του σπιτιού.

Αν υπάρχει άνθρωπος στο δωμάτιο τότε ανάβουν αυτόματα τα φώτα ενώ αν δεν υπάρχει, περιμένουν 20 δευτερόλεπτα από την στιγμή που θα σταματήσουν να εντοπίζουν ανθρώπινη παρουσία και έπειτα κλείνουν τα φώτα. Έτσι ο φωτισμός του σπιτιού ρυθμίζεται αυτόματα χωρίς να υπάρχει ενδεχόμενο να ξεχαστούν ανοιχτά αλλά ακόμα και για μικρές περιόδους απουσίας ανθρώπου, σταματούν να λειτουργούν εξοικονομώντας ενέργεια.

Οι πλακέτες επενεργούν επίσης στην θέρμανση του σπιτιού με τον εξής τρόπο:

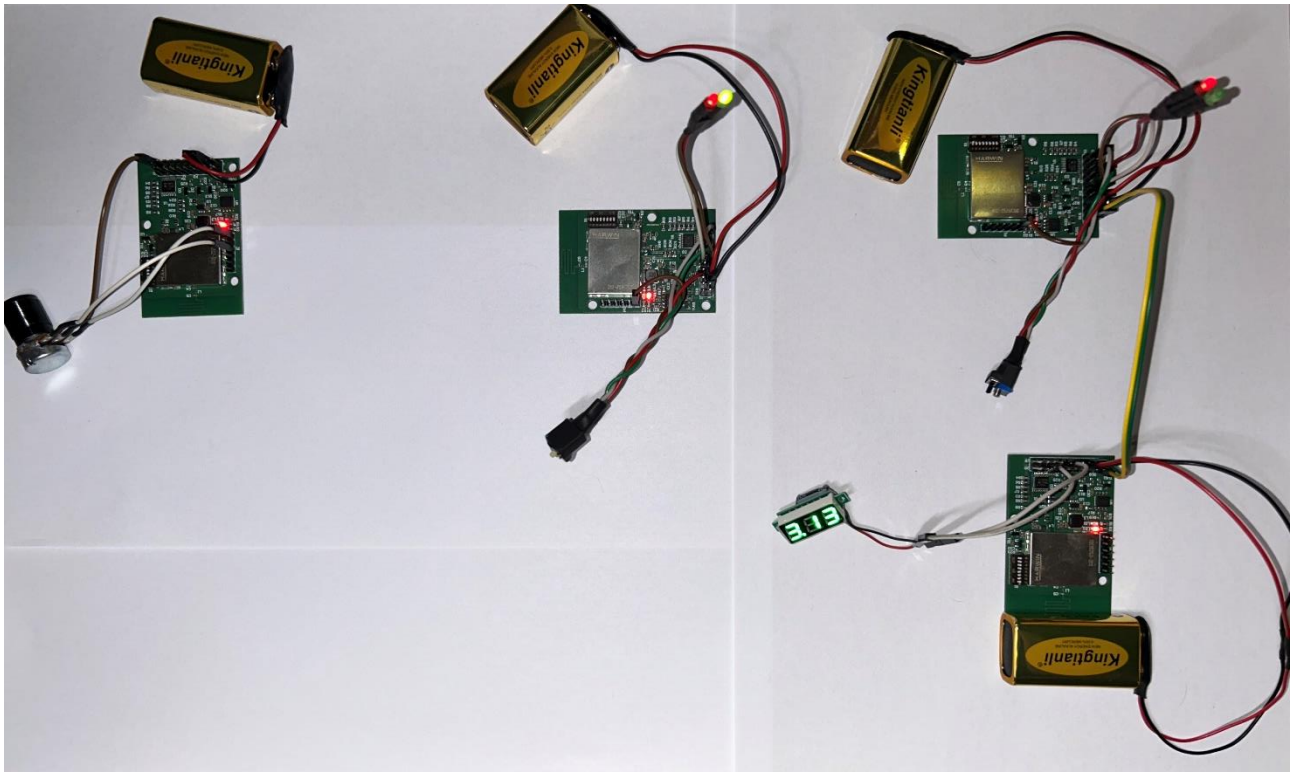
Αν υπάρχει άνθρωπος στο δωμάτιο τότε ανάβουν την θέρμανση έως ότου η θερμοκρασία φτάσει τους 25°C όπου και την κλείνουν. Όταν η θερμοκρασία ξαναπέσει στους 23°C τότε ξανανάβουν την θέρμανση. Με τον τρόπο αυτό αν υπάρχει άνθρωπος εντός του δωματίου η θερμοκρασία του δωματίου διατηρείται από 23°C έως 25°C.

Αν δεν υπάρχει άνθρωπος στο δωμάτιο αλλά υπάρχει άνθρωπος στο διπλανό δωμάτιο, δηλαδή το σπίτι δεν είναι άδειο, τότε διατηρούν την θερμοκρασία του δωματίου από 21°C έως 23°C, ώστε να εξοικονομείται ενέργεια αλλά να είναι έτοιμο το δωμάτιο να δεχθεί τον άνθρωπο χωρίς να χρειάζεται να θερμάνει από χαμηλότερη θερμοκρασία και να δαπανήσει πολύ μεγαλύτερο ποσό ενέργειας.

Αν δεν υπάρχει άνθρωπος σε κανένα δωμάτιο από τα δύο του σπιτιού τότε αναμένει 5 λεπτά από την τελευταία ανίχνευση ανθρώπινης παρουσίας εντός οποιουδήποτε δωματίου και κλείνει εντελώς την θέρμανση. Με τον τρόπο αυτό η θέρμανση του σπιτιού ρυθμίζεται αυτόματα αποκλείοντας το ενδεχόμενο να παραμείνει ανοιχτή από απερισκεψία. Απώτερος σκοπός φυσικά είναι η εξοικονόμηση ενέργειας αφού άλλωστε η θέρμανση είναι μια από τις βασικότερες ενεργειακές δαπάνες.

5.3 Επίδειξη Λειτουργικότητας

Για την επίδειξη της λειτουργίας των πλακετών και του δικτύου πραγματοποιούμε την συνδεσμολογία ακριβώς όπως και στην εφαρμογή και προσομοιώνουμε την λειτουργία του αισθητήρα ανίχνευσης ανθρώπινης παρουσίας με έναν διακόπτη ο οποίος συνδέεται από την μια μεριά με HIGH τάση και από την άλλη με μία από τις ψηφιακές εισόδους της πλακέτας. Ακόμη προσομοιώνομε τον φωτισμό με ένα πράσινο LED και την θέρμανση με ένα κόκκινο LED τα οποία ενεργοποιούνται μέσω των ψηφιακών εξόδων της πλακέτας, οι οποίες έχουν την δυνατότητα οδήγησης ρεύματος έως 1.2A ώστε να ενεργοποιήσουν κάποιο ρελέ. Τέλος προσομοιώνουμε τον αισθητήρα θερμοκρασίας που είναι συνδεδεμένος με την πλακέτα που βρίσκεται στο μπαλκόνι με μια αναλογική τάση που προέρχεται από τον μεσαίο ακροδέκτη ενός ποτενσιόμετρου ώστε να μπορούμε να προκαλέσουμε διαφοροποιήσεις της θερμοκρασίας εύκολα και να παρατηρήσουμε την μεταβαλλόμενη τάση που παράγει η πλακέτα που συνδέεται με τον λέβητα στο υπόγειο. Η πλακέτες κατά την επίδειξη λειτουργικότητας είναι οι εξής:



Εικόνα 44: Επίδειξη λειτουργικότητας συστήματος

Παρατηρούμε το αναμμένο πράσινο και κόκκινο LED στο δωμάτιο 2 που σημαίνει πως στο συγκεκριμένο δωμάτιο υπάρχει άνθρωπος αλλά και το αναμμένο κόκκινο LED στο δωμάτιο 1 χωρίς όμως το πράσινο να είναι ανοιχτό. Αυτό σημαίνει πως δεν υπάρχει άνθρωπος στο δωμάτιο αλλά διατηρεί την θέρμανση ανοιχτή σε χαμηλότερη στάθμη αφού γνωρίζει την ύπαρξη ανθρώπου στο διπλανό δωμάτιο. Αυτό επιβεβαιώνει την ασύρματη επικοινωνία μεταξύ τους. Τέλος παρατηρούμε και την αναλογική τάση που παράγει η πλακέτα στο υπόγειο η οποία φαίνεται στο ψηφιακό βολτόμετρο που είναι συνδεδεμένο με την συγκεκριμένη πλακέτα.

6 *Συμπεράσματα και Μελλοντική Δουλειά*

Συμπεράσματα

Συμπερασματικά, η εργασία καταδεικνύει την αξία της συνδυασμένης χρήσης πρωτοκόλλων ZigBee και RS-485 για την υλοποίηση ενός δικτύου αισθητήρων με δυνατότητες ενεργειακής διαχείρισης κτηρίων. Το σύστημα που αναπτύχθηκε αξιοποιεί τη διαλειτουργικότητα μεταξύ ασύρματων και ενσύρματων κόμβων, επιτρέποντας την παρακολούθηση και τον έλεγχο της κατανάλωσης ενέργειας σε πραγματικό χρόνο. Η προσέγγιση αυτή παρέχει τη δυνατότητα βελτιστοποίησης της ενεργειακής αποδοτικότητας μέσω της αλληλεπίδρασης με το περιβάλλον και των δεδομένων που συλλέγονται από τους αισθητήρες.

Η ανάπτυξη τόσο του hardware όσο και του software αποτέλεσε κρίσιμη παράμετρο για την επίτευξη του στόχου, και ο συστηματικός συν-σχεδιασμός τους ήταν καθοριστικός για την επιτυχή ολοκλήρωση του συστήματος. Οι εφαρμογές της τεχνολογίας αυτής είναι πολλαπλές και μπορούν να προσαρμοστούν σε διάφορες απαιτήσεις κτηρίων, βελτιώνοντας τόσο την ενεργειακή αποδοτικότητα όσο και τη βιωσιμότητα, με θετικά αποτελέσματα για την προστασία του περιβάλλοντος.

Μελλοντική Δουλεία

Στα μελλοντικά σχέδια του συστήματος θα μπορούσε να είναι μια πιθανή επέκταση του που να περιλαμβάνει την ανάπτυξη μιας εφαρμογής κινητού που θα επικοινωνεί με το δίκτυο ZigBee και θα επιτρέπει την παρακολούθηση και τον έλεγχο των αισθητήρων και των πλακετών σε πραγματικό χρόνο. Η εφαρμογή αυτή θα μπορούσε να λαμβάνει δεδομένα από το σύστημα, όπως την κατανάλωση ενέργειας και τις περιβαλλοντικές συνθήκες, και να επιτρέπει στους χρήστες να αλληλεπιδρούν με τις πλακέτες, ρυθμίζοντας παραμέτρους όπως η θερμοκρασία, ο φωτισμός ή άλλα στοιχεία που επηρεάζουν την ενεργειακή διαχείριση.

Η συλλογή δεδομένων σε πραγματικό χρόνο θα μπορούσε να αποθηκεύεται σε μια κεντρική βάση δεδομένων, επιτρέποντας την ανάλυση μακροχρόνιων τάσεων και

την εξαγωγή χρήσιμων συμπερασμάτων για τη βελτιστοποίηση της ενεργειακής απόδοσης του κτηρίου. Τα δεδομένα αυτά θα μπορούσαν να είναι προσβάσιμα τόσο μέσω της εφαρμογής όσο και μέσω web interface, δίνοντας τη δυνατότητα στους διαχειριστές να παρακολουθούν και να βελτιώνουν την απόδοση του συστήματος με βάση ιστορικά και ζωντανά δεδομένα.

Η ενσωμάτωση τεχνητής νοημοσύνης (AI) θα μπορούσε να αποτελέσει σημαντική προσθήκη για την αυτοματοποίηση της διαχείρισης ενέργειας. Με τη χρήση αλγορίθμων μηχανικής μάθησης (machine learning), το σύστημα θα μπορούσε να αναλύει τα δεδομένα που συλλέγονται από τους αισθητήρες και να προβλέπει την κατανάλωση ενέργειας με βάση τις συνήθειες των χρηστών, τις εξωτερικές κλιματικές συνθήκες ή άλλους παράγοντες. Επιπλέον, η AI θα μπορούσε να βοηθήσει στη λήψη αυτόματων αποφάσεων για τη βελτίωση της αποδοτικότητας, όπως η αυτόματη προσαρμογή των συστημάτων θέρμανσης και ψύξης ή η ενεργοποίηση/απενεργοποίηση συσκευών όταν δεν είναι απαραίτητες.

Η εφαρμογή AI σε συνδυασμό με μια κινητή πλατφόρμα και τη βάση δεδομένων θα επιτρέψει την πλήρη αυτοματοποίηση της ενεργειακής διαχείρισης, εξασφαλίζοντας όχι μόνο την εξοικονόμηση ενέργειας, αλλά και την παροχή μιας πιο έξυπνης και βιώσιμης λύσης για τη διαχείριση των κτηρίων.

Βιβλιογραφία

[1] Lee, J. S., Su, Y. W., & Shen, C. C. (2007, November). A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. In *IECON 2007-33rd Annual Conference of the IEEE Industrial Electronics Society* (pp. 46-51). IEEE.

[2] Craig, W. C. (2004). Zigbee: Wireless control that simply works. *Zigbee Alliance ZigBee Alliance*.

[3] Gislason, D. (2008). *Zigbee wireless networking*. Newnes.

[4] www.digi.com/solutions/by-technology/zigbee-wireless-standard

[5] AN5506, Getting Started with ZigBee on STM32WB series, *ST Microelectronics*

[6] Shi, L., & Guo, B. (2009, October). RS485/422 solution in embedded access control system. In *2009 2nd International Conference on Biomedical Engineering and Informatics* (pp. 1-4). IEEE.

[7] Zhao, L., Liang, R., & Zhang, J. (2014). The solving of bias resistor and its effect on the RS485 fieldbus. *J. Adv. Comput. Networks*, *2*(1), 71-75.

[8] THVD1420DR Datasheet, *Texas Instruments*

[9] www.sameskydevices.com/blog/rs-485-serial-interface-explained

[10] STM32WB55RGV6 Datasheet, *ST Microelectronics*

[11] STM32CubeIDE Data bried, *ST Microelectronics*

[12] Mataloto, B., Ferreira, J. C., & Cruz, N. (2019). LoBEMS—IoT for building and energy management systems. *Electronics*, *8*(7), 763.

[13] Minoli, D., Sohraby, K., & Occhiogrosso, B. (2017). IoT considerations, requirements, and architectures for smart buildings—Energy optimization and next-generation building management systems. *IEEE Internet of Things Journal*, *4*(1), 269-283.

[14] Hossein Motlagh, N., Mohammadrezaei, M., Hunt, J., & Zakeri, B. (2020). Internet of Things (IoT) and the energy sector. *Energies*, *13*(2), 494.

[15] Botta, A., De Donato, W., Persico, V., & Pescapé, A. (2016). Integration of cloud computing and internet of things: a survey. *Future generation computer systems*, *56*, 684-700.

[16] Yaïci, W., Krishnamurthy, K., Entchev, E., & Longo, M. (2020, June). Survey of internet of things (IoT) infrastructures for building energy systems. In *2020 Global Internet of Things Summit (GIoTS)* (pp. 1-6). IEEE.

[17] Sidid, S., & Gaur, S. (2017, April). Smart grid building automation based on Internet of Things. In *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)* (pp. 1-4). IEEE.

[18] Zhou, B., Li, W., Chan, K. W., Cao, Y., Kuang, Y., Liu, X., & Wang, X. (2016). Smart home energy management systems: Concept, configurations, and scheduling strategies. *Renewable and Sustainable Energy Reviews*, *61*, 30-40.

[19] Nugur, A., Pipattanasomporn, M., Kuzlu, M., & Rahman, S. (2018). Design and development of an iot gateway for smart building applications. *IEEE Internet of Things Journal*.

[20] Al Naqbi, A., Alyieliely, S. S., Talib, M. A., Nasir, Q., Bettayeb, M., & Ghenai, C. (2021, October). Energy Reduction in Building Energy Management Systems Using the Internet of Things: Systematic Literature Review. In *2021 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1-7). IEEE.

[21] TPS560430 SIMPLE SWITCHER® 4-V to 36-V, 600-mA Synchronous Step-Down Converter Data Sheet, *Texas Instruments*

[22] ADR5043, Precision, Micropower Shunt Mode Voltage References Data Sheet, *Analog Devices*

[23] MLPF-WB55-01E3 Datasheet, *ST Microelectronics*

[24] AN5129, Guidelines for meander design using low-cost PCB antennae with 2.4 GHz radio for STM32WB/WB0 MCUs, *ST Microelectronics*

[25] Misman, D., Aziz, M. A., Husain, M. N., & Soh, P. J. (2009, March). Design of planar meander line antenna. In *2009 3rd European Conference on Antennas and Propagation* (pp. 2420-2424). IEEE.

[26] AN5434, On-board antennas reference design for the STM32WB Series MCUs, *ST Microelectronics*

[27] CDSOT23-SM712 - Surface Mount TVS Diode Data Sheet, *Bourns*

[28] TMP23x Low-Power, High-Accuracy Analog Output Temperature Sensors Data Sheet, *Texas Instruments*

[29] TPL7407L 40-V 7-Channel Low Side Driver Data Sheet, *Texas Instruments*

[30] PD005-Product-Datasheet-SHIELDING-EMC-Shielding-range-EZi, *Harwin*

[31] https://www.st.com/en/embedded-software/stm32cubewb.html

[32] AN5289, How to build wireless applications with STM32WB MCUs, Chapter 4.4 *ST Microelectronics*

[33] https://wiki.stmicroelectronics.cn/stm32mcu/wiki/Utility:Sequencer

[34] AN5498 How to use Zigbee® clusters templates on STM32WB Series, *ST Microelectronics*

[35] AN5627, STM32WB Series ZigBee commissioning guide, *ST Microelectronics*

[36] AN5500, ZSDK API implementation for Zigbee® on STM32WB Series, *ST Microelectronics*

[37] AN5491, Creating manufacture specific clusters on STM32WB Series *ST Microelectronics*

[38] https://www.dfrobot.com/product-2648.html