ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάλυση και εφαρμογές με το υλικό κόμβου LoRa

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΜΑΥΡΙΚΟΥ ΝΗΡΕΑ

(ΑΕΜ: 2415)

Επιβλέπων : Δόσης Μιχαήλ

Καθηγητής

Καστοριά Μήνας - Έτος (παρουσίασης της εργασίας)

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάλυση και εφαρμογές με το υλικό κόμβου LoRa

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΜΑΥΡΙΚΟΥ ΝΗΡΕΑ

(ΑΕΜ: 2415)

Επιβλέπων : Δόσης Μιχαήλ

Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ημερομηνία εξέτασης

.....	.....	.....
Ον/μο Μέλους	Ον/μο Μέλους	Ον/μο Μέλους
Ιδίοτητα Μέλους	Ιδίοτητα Μέλους	Ιδίοτητα Μέλους

Καστοριά Μήνας - Έτος (παρουσίασης της εργασίας

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την εταιρία Kudzu Technologies για την παροχή του αρχικού υλικού που χρησιμοποιήθηκε για την εκμάθηση και κατανόηση αυτής της εργασίας (SODAQ ONE).

Επίσης, θα ήθελα να ευχαριστήσω την εταιρία Kernel IT Solutions για την βοήθεια στην εκμάθηση της γλώσσας Python, χρήσεις των πλακετών υλικού και προτάσεις για το νέο υλικό που χρειάστηκε.

Τέλος, ευχαριστώ θερμά τον καθηγητή Μιχαήλ Δόση για την στήριξη και την βοήθεια του καθ’ όλη τη διάρκεια της εργασίας.

Περίληψη

Αυτή η διατριβή έχει ως στόχο τη διερεύνηση του υλικού κόμβου LoRa καθώς και την χρήση αυτού στην αποστολή πολυμέσων. Το LoRa χωρίζεται σε 2 πρωτόκολλά. Το πιο ευρέως γνωστό είναι το LoRaWAN, το οποίο χρησιμοποιεί 2 διαφορετικές πλακέτες, μία ως αποστολέα και την άλλη ως πύλη, για να στείλει δεδομένα μέσω διαδικτύου. Το άλλο πρωτόκολλο, με το οποίο και θα ασχοληθούμε, είναι το P2P, το οποίο χρησιμοποιεί 2 πλακέτες οι οποίες επικοινωνούν μεταξύ τους τοπικά, η μία ως αποστολέας και η άλλη ως δέκτης. Χρησιμοποιώντας 2 πλακέτες υλικού του μοντέλου Raspberry Pi 4 και προγραμματίζοντάς τες χρησιμοποιώντας τη γλώσσα προγραμματισμού Python, καταφέραμε να στείλουμε μια εικόνα από τη μία πλακέτα στην άλλη. Αυτή η αποστολή μπορεί να γίνει σε αποστάσεις που αγγίζουν τα 3 χιλιόμετρα σε αστικές περιοχές και τα 15 χιλιόμετρα σε πιο ανοιχτές περιοχές, όπου υπάρχει οπτικό πεδίο μεταξύ των πλακετών.

Λέξεις Κλειδιά:

Πλακέτα, Αποστολή, LoRa, LoRaWAN, P2P, Raspberry Pi, Εικόνα, Κόμβος, Πολυμέσα.

Abstract

This Thesis aims to investigate the LoRa node hardware protocol as well as its use in multimedia transmission. LoRa is divided into 2 protocols. The most widely known is LoRaWAN, which uses 2 different boards, one as a sender and the other as a gateway, to send data over a cloud on the internet. The other protocol, which we will personally examine, is P2P, which uses 2 boards that communicate with each other locally, one as a sender and the other as a receiver. Using 2 Raspberry Pi 4 model hardware boards and programming them using the Python programming language, we managed to send an image from one board to the other. This transmission can be done at distances of up to 3 kilometers in urban areas and 15 kilometers in more rural areas, where there is a line of sight between the plates.

Key Words:

Boards, Transmission, LoRa, P2P, Raspberry Pi, Multimedia, Picture, Noda,

Πίνακας Περιεχομένων

Λίστα Εικόνων

Εισαγωγή

Εν έτη 2023, ζούμε σε μία κοινωνία που θεωρείται τεχνολογικά ανεπτυγμένη. Οι υπολογιστές είναι παντού, ακόμα και στις τσέπες μας. Το διαδίκτυο είναι πλέον απαραίτητο για όλους, σχεδόν σε επίπεδο που ήταν κάποτε ο ηλεκτρισμός. Σε μερικά χρόνια, πιθανώς να είναι απαραίτητο στην ανθρώπινη διαβίωση. Αν όμως βρεθούμε σε ένα μέρος όπου οι τεχνολογικές υποδομές δεν αρκούν. Δεν υπάρχει κάλυψη σήματος. Δεν υπάρχουν ηλεκτρικοί πυλώνες να μας μεταφέρουν ρεύμα.

Ακόμα και σε τέτοια μέρη, η τεχνολογία υλικού κόμβου LoRa λειτουργεί άψογα. Βασισμένη σε κατανάλωση ρεύματος αρκετά μικρή, όπου μια μπαταρία αρκεί για να την τροφοδοτήσει για μέρες, με τη χρήση μίας κεραίας μπορεί να μεταδώσει δεδομένα σε τεράστιές χιλιομετρικές αποστάσεις και σε περιπτώσεις, ακόμα και στο διαδίκτυο. Σε αυτήν την εργασία θα ερευνήσουμε μία τέτοια χρήση του.

LoRa, ταξίδι και εφαρμογές.

Τι είναι το πρωτόκολλο υλικού κόμβου LoRa και ποιες οι χρήσεις του.

Τι είναι το LoRa, οι υποκατηγορίες του και χρήσεις αυτών.

Το ταξίδι στον κόσμο του LoRa.

Ο δρόμος και το ταξίδι από την αρχή αυτής της εργασίας μέχρι το τέλος της.

Κώδικας και λειτουργία.

Επεξήγηση του κώδικα και πως αυτός λειτουργεί.

Χρήση του μηχανήματος.

Λεπτομερής επεξήγηση για ανακατασκευή χρήση.

Ενδεικτικές Εφαρμογές και μελλοντική χρήση.

Τρόποι χρήσης αυτής της εργασίας σε καθημερινό περιβάλλον.

Υλικό και λογισμικό που χρησιμοποιήθηκε.

Λίστα υλικού και λογισμικού.

Εικόνα 1. LoRa Logo

Πηγή: https://devopedia.org/lora#Semtech-2017

1.1 Τί είναι το πρωτόκολλο υλικού κόμβου LoRa και ποιες οι χρήσεις του.

Το πρωτόκολλο υλικού κόμβου LoRa είναι μία μέθοδος αποστολής μέσω ράδιο η οποία βασίζεται στο Φάσμα Εξάπλωσης Chirp (Chirp Spread Spectrum (CSS)) [1]. Έχει δημιουργηθεί από την Γαλλική εταιρία Cycleo την οποία αγόρασε αργότερα η Semtech, η οποία έχει τα δικαιώματα της πατέντας μέχρι και σήμερα. Το όνομα του πρωτόκολλου είναι ακρωνύμιο και προέρχεται από τις Αγγλικές λέξεις Long Range που σημαίνει Μακριά Εμβέλεια. [2] [3]

Η βασική χρήση μίας πλακέτας υλικού που χρησιμοποιεί LoRa είναι η επικοινωνία με κάποια άλλη παρόμοια πλακέτα που χρησιμοποιεί το ίδιο πρωτόκολλο (Peer to Peer (P2P)). Φτάνοντας αποστάσεις που αγγίζουν τα 5 χιλιόμετρα σε αστικές περιοχές και τα 15 χιλιόμετρα σε ανοιχτές περιοχές με ορατότητα μεταξύ των πλακετών και έχοντας ελάχιστο ενεργειακό κόστος, το πρωτόκολλο LoRa χρησιμοποιείται για την ανταλλαγή δεδομένων ή τον έλεγχο συσκευών σε μεγάλες αποστάσεις.

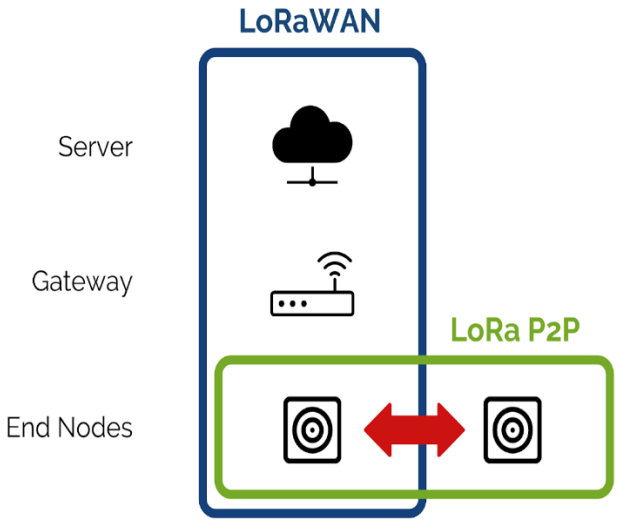
Το υπό-πρωτόκολλο LoRaWAN (Long Range Wide Area Network) είναι το κύριο πρωτόκολλο χρήσης του LoRa. Το LoRaWAN είναι αναγνωρισμένο από την Παγκόσμια Ένωση Τηλεπικοινωνιών (ITU) και προτείνεται για χρήσης μεγάλης εμβέλειας με μικρό ενεργειακό κόστος. Το πρωτόκολλο διατηρείται από την LoRa Alliance [4], μία ανοιχτή, μη-κερδοσκοπική ομάδα, της οποίας ιδρυτικό μέλος είναι η Semtech. [3]

Το Πρωτόκολλο LoRa είναι ο πρωτοπόρος στο δίκτυο της αποστολής μεγάλης εμβέλειας με μικρό ενεργειακό κόστος (Low Power, Wide Area (LPWA). Η βασική διαφορά που το ξεχωρίζει είναι η χρήση μίας πύλης (Gateway) σαν δέκτης, η οποία επικοινωνεί διαδικτυακά με ένα δίκτυο συσκευών, επιτρέποντας έτσι την υπέρβαση του ορίου εμβέλειας, καθώς και την ευκολία χρήσης. Χρησιμοποιώντας το Διαδίκτυο των Πραγμάτων (Internet of Things (IoT)) σαν βάση και έχοντας πολλά πλεονεκτήματα, όπως το χαμηλό ενεργειακό κόστος λειτουργίας, αμφίδρομη επικοινωνία, εσωτερική ασφάλεια, ευκολία μεταφοράς και μεγάλη εμβέλεια. Αυτά, μαζί

με το μικρό ρυθμό bit το διαχωρίζουν από άλλες πιο σπάταλες μορφές αποστολής δεδομένων. Σήμερα, πλακέτες υλικού LoRa χρησιμοποιούνται σε πολλά μέρη γύρο μας, όπως αισθητήρες, μετρητές, συσκευές ελέγχου μηχανημάτων και πολλά άλλα. [3]

Τα πράγματα που κρατάνε ακόμα πίσω το πρωτόκολλο είναι το σχετικά μικρό εύρος ζώνης του. Παρ' όλες τις λειτουργίες που προσφέρει στο μέγεθος και κόστος του, η αποστολή μεγάλου όγκου πληροφοριών δεν είναι ιδιαίτερα εφικτή, γι' αυτό και χρησιμοποιείται σε πιο απλές μορφές υλικού, όπως αισθητήρες και κλιματιστικά.

Σχετικά με την αρχιτεκτονική που χρησιμοποιείται, χρειάζεται τουλάχιστον μία πλακέτα υλικού που να έχει υποδοχή για κεραία. Μετά, ανάλογα τη μέθοδο αποστολής, χρειάζεται είτε ένα δεύτερο παρόμοιο μηχάνημα στην περίπτωση αποστολής P2P ή στην περίπτωση WAN ένα δεύτερο μηχάνημα το οποίο θα δρα σαν πύλη (Gateway), το οποίο θα λαμβάνει τις πληροφορίες που θα στέλνονται από το πρώτο μηχάνημα και θα τις εκπέμπει σε κάποιου είδους server για να υπάρχει πρόσβαση σε αυτές σε μακρινές αποστάσεις μέσο του διαδικτύου.



Εικόνα 2. Τα πρωτόκολλα LoRa και η αρχιτεκτονική τους.

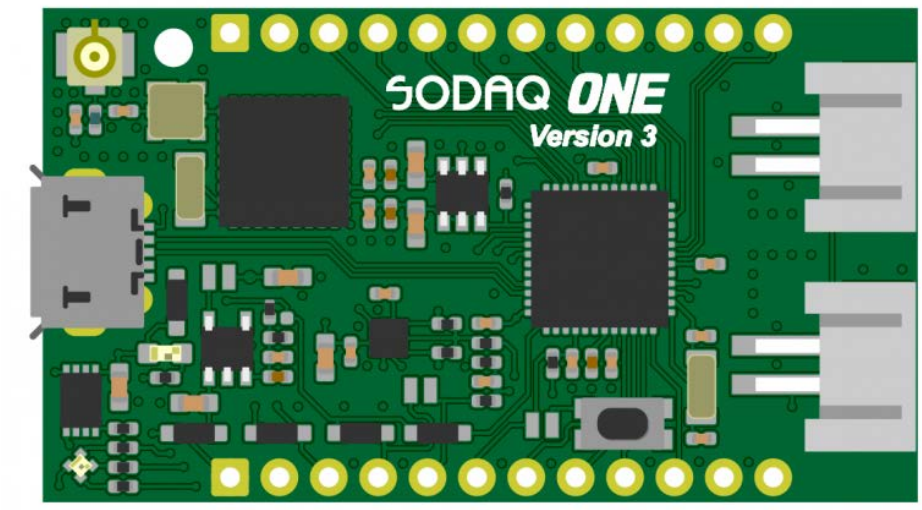
Πηγή: https://www.seeedstudio.com/blog/2021/04/26/what-is-peer-to-peer-p2p-lora-communication/

1.2 Το ταξίδι στον κόσμο του LoRa.

Αυτή η διατριβή ξεκίνησε με στόχο την αποστολής κάποιου πολυμέσου, ειδικότερα μιας εικόνας, μέσω του πρωτοκόλλου LoRa. Μετά από σχετική έρευνα, έμαθα ότι μια τοπική εταιρία, η Kudzu Technologies [5], ασχολείται με παρόμοια πράγματα. Απευθύνθηκα λοιπών στον CEO της εταιρίας, ο οποίος ήταν ιδιαίτερα χαρούμενος να μου δανείσει μία πλακέτα υλικού με δυνατότητες LoRa για να χρησιμοποιήσω κατά τη διάρκεια της εργασίας μου. Έτσι, ήρθαν στην κατοχή μου 2 πλακέτες υλικού τύπου SODAQ ONE [6] καθώς και 2 κεραίες για ράδιο-αποστολές με LoRa.

Ξεκίνησα με το τί είναι το πρωτόκολλο LoRa. Δεν είχα ιδέα. Δεν είναι κάτι που το ακούς καθημερινά, παρ' όλο που υπάρχει παντού γύρο μας. Επίσης, πως θα μπορούσα να προγραμματίσω αυτές τις πλακέτες; Είχα μια ιδέα για το πρόγραμμα Xilinx αλλά ήθελα κάτι πιο απλό. Έτσι κατέληξα στην Γλώσσα C++, την οποία γνώριζα καλά/

Οι πρώτες βδομάδες περνούσαν μαθαίνοντας για το πρωτόκολλο LoRa, τις εφαρμογές του. Βλέποντας πολλές εφαρμογές στο διαδίκτυο, αποφάσισα να χρησιμοποιήσω το Android Studio για να υλοποιήσω την εργασία μου. Πρώτα όμως πρέπει να καταφέρω να κάνω τις πλακέτες να επικοινωνήσουν.

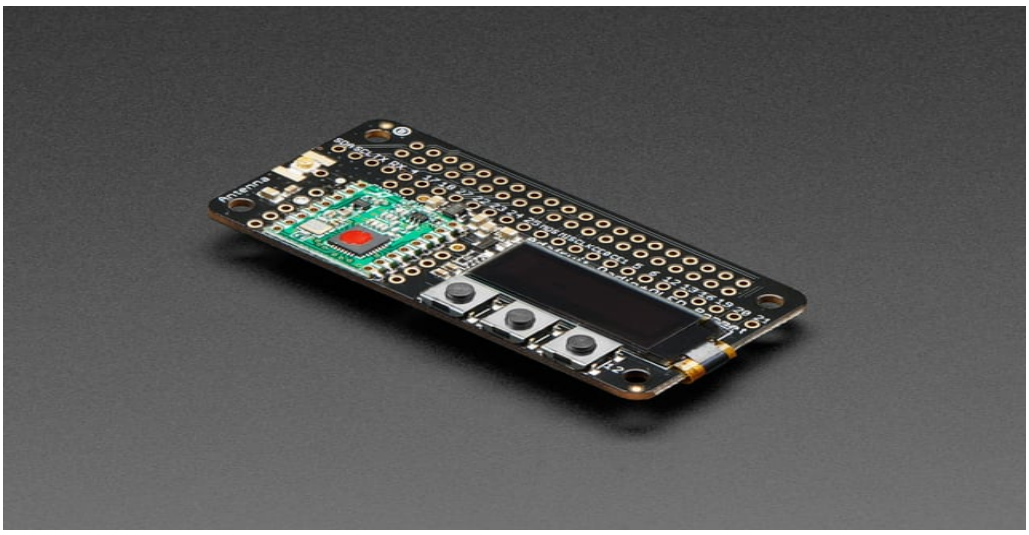


Εικόνα 3. Κάτοψη πλακέτας τύπου SODAQ ONE (Model RN2483)

Πηγή: https://support.sodaq.com/Boards/One/

Τέτοιου είδους πλακέτες υλικού συνήθως λειτουργούν με έναν ατέρμονα βρόγχο ο οποίος ενεργοποιείται μόλις ξεκινήσει το πρόγραμμα να τρέχει [7]. Ευτυχώς, οι πλακέτες SODAQ ONE είχαν ένα μικρό RGB LED φωτάκι το οποίο και χρησιμοποίησα για να καταφέρω την επικοινωνία μεταξύ τους, να στείλω δηλαδή εντολή από τη μία πλακέτα στην άλλη για να ανάψει το λαμπάκι και να αλλάζει χρώμα σε τακτά χρονικά διαστήματα. Δούλεψε!

Μετά από διορθώσεις κάποιων bugs στον κώδικα, συνειδητοποίησα ότι η αποστολή μίας ολόκληρης εικόνας είναι ίσως αδύνατη, λόγο του μικρού εύρους ζώνης του πρωτοκόλλου LoRa. Επικοινώνησα με τον καθηγητή μου για να με καθοδηγήσει. Μου πρότεινε να τεμαχίσω την εικόνα σε bytes και να την στείλω σε κομμάτια τα οποία θα συναρμολογούνταν στην πλακέτα-δέκτη. Ήρθε η ώρα για την πραγματική αποστολή. Υπήρξε όμως ένα πρόβλημα. Οι πλακέτες SODAQ ONE δεν έχουν ενσωματωμένη μνήμη, το οποίο τα καθιστά ανίκανα να κρατήσουν δεδομένα, όπως μια φορτωμένη εικόνα. Επικοινώνησα με μία άλλη τοπική εταιρία η οποία μεταξύ άλλων, εμπορευόταν υλικό. Αφού τους εξήγησα το πρόβλημά μου, μου πρότειναν τα Raspberry Pi, τα οποία επίσης δεν γνώριζα, παρά μόνο ονομαστικά. Αποφάσισα επίσης να προμηθευτώ με 2 επιπρόσθετα εξαρτήματα για τα Raspberry, τα RFM9X LoRa Bonnet της Adafruit, τα οποία συμπεριλαμβάνουν μία οθόνη και 3 μικρά κουμπάκια, τα οποία θα με διευκόλυναν, καθώς και μία μικρή οθόνη OLED.

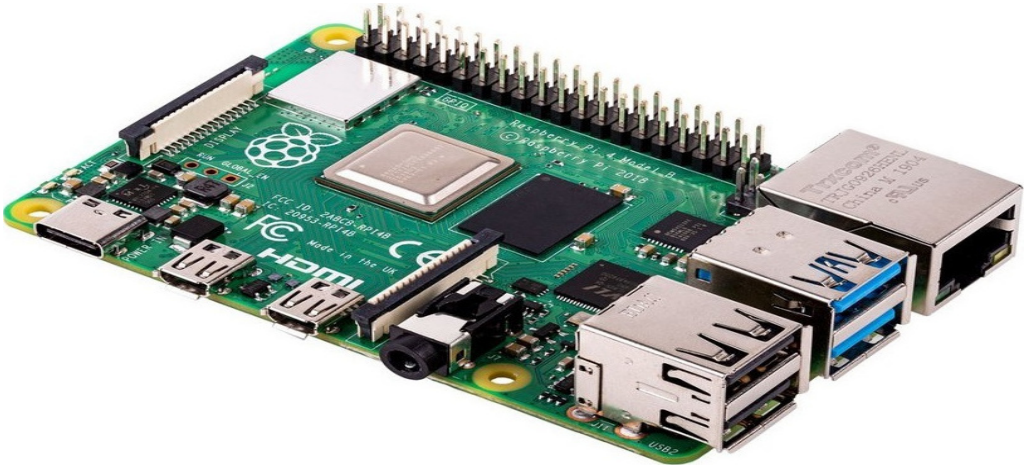


Εικόνα 4. Adafruit RFM9X LoRa Bonnet

Πηγή: **https://learn.adafruit.com/adafruit-radio-bonnets/**

Η τιμή του συνολικού πακέτου με ανάγκασε να διακόψω την πτυχιακή για ένα καλοκαίρι για να μαζέψω χρήματα για να τα προμηθευτώ. Αφού ήρθαν στην κατοχή μου, τα συναρμολόγησα και τα συνέδεσα κατ' ευθείαν. Τα Raspberry Pi είναι ένας μικροϋπολογιστής, με τις δυνατότητες ενός κανονικού Η/Υ. Λειτουργούν με το λειτουργικό λογισμικό Raspbian το οποίο είναι βασισμένο σε Unix [8]. Μη γνωρίζοντας για τα Raspberry στράφηκα στο διαδίκτυο για ακόμη μια φορά. Τα πιο πολλά παραδείγματα στο διαδίκτυο ήταν γραμμένα σε γλώσσα Python, την οποία γνώριζα μόνο ονομαστικά. Διαβάζοντας διάφορα Forum, πείστηκα ότι ίσως η αλλαγή γλώσσας να ήταν προς όφελός μου, έτσι άλλαξα από τη γλώσσα C++ σε Python και προγραμματιστικό περιβάλλον από το Android Studio στο Thonny. Ο επόμενος καιρός ήταν κυρίως εκμάθηση της γλώσσας Python. Ευτυχώς, ή βάση της ήταν παρόμοια με άλλες γλώσσες που γνώριζα και αρκετά πιο απλή.

Αρχικά εγκατέστησα το λογισμικό στα Raspberry μέσω της μικρής ΣΔ κάρτας που παίρνουν. Χρησιμοποίησα το λογισμικό PuTTY για να έχω πρόσβαση στην κονσόλα σε επίπεδο διαχειριστή καθώς και το λογισμικό XRDP μαζί με τη χρήση απομακρυσμένης επιφάνειας εργασίας των Windows για να έχω οπτική πρόσβαση μέσω του δικού μου Η/Υ. Στη συνέχεια, τα συνέδεσα στο δίκτυο μου μέσω Wi-Fi ώστε να επικοινωνώ μαζί τους ασύρματα και ξεκίνησα.



Εικόνα 5. Πλακέτα Raspberry Pi 4 (Model B)

Πηγή: **https://www.raspberrypi.com/products/raspberry-pi-4-model-b/**

Έχοντας κατανοήσει την γλώσσα σε ένα καλό επίπεδο, πρώτο μου μέλημα ήταν να αναπαράγω τον κώδικα που έτρεχα στα SODAQ ONE στα καινούρια Raspberry. Χρησιμοποιώντας τον οδηγό χρήσης της Adafruit, απέστειλα μηνύματα μέσω της μικρής οθόνης που είχα συνδέσει στα Raspberry. Στον ίδιο οδηγό, υπήρχαν και μικρά Demo για τον προγραμματισμό των κουμπιών, και τότε μου ήρθε η ιδέα. Θα προγραμματίσω κάθε ένα κουμπί έτσι ώστε να κάνει κάτι διαφορετικό στον κώδικα, αντί να το φορτώσω όλο στον κλασικό ατέρμονο βρόγχο. Έτσι, θα μπορούσα να χρησιμοποιήσω τον βρόγχο μόνο για το κομμάτι του παραλήπτη, ενώ τα κουμπιά θα αναλάμβαναν όλα τα υπόλοιπα, από τη φόρτωση της εικόνας στην αποστολή.

Αρχικά εστίασα στο κομμάτι της αποστολής πίνακα. Κάνοντας πολλών ειδών πίνακες και στέλνοντάς τους με το πάτημα ενός από τα κουμπιά. Ο προγραμματισμός του κουμπιού ήταν αρκετά απλός καθώς λειτουργούν με έναν βρόγχο “if - else”, συνεπώς είναι δυνατό να προστεθούν αρκετές εντολές μετά το πάτημα του κουμπιού, οι οποίες θα εκτελεστούν σε ακολουθία. Κατέληξα να φτιάχνω τυχαία στοιχεία, να τα βάζω μέσα σε έναν πίνακα και να τα στέλνω με το πάτημα ενός κουμπιού. Μετά όμως ήρθε το κομμάτι της εικόνας. Πως μπορώ να την μετατρέψω σε κάτι που να μπορώ να στείλω.

Ψάχνοντας το διαδίκτυο, βρήκα μια ιδέα που υλοποιούσε την κωδικοποίηση μιας εικόνας με base-64 φτιάχνοντας ένα αρχείο κειμένου με τα δεδομένα της. Στη συνέχεια, μετέτρεπε αυτό το αρχείο πίσω σε εικόνα. Άρα η αρχική ιδέα ήταν να στείλω τα περιεχόμενα του αρχείου και μετά να το αποθηκεύσω ξανά και να το μετατρέψω σε εικόνα. Δυστυχώς, αυτό αποδείχτηκε πιο δύσκολο απ’ ότι περίμενα. Λόγο της κωδικοποίησης, είχα πολλά προβλήματα με την αποθήκευση του αρχείου στον πίνακα καθώς και την αποκωδικοποίηση.

Έπειτα από αρκετή αναζήτηση, έμαθα για την ύπαρξη της PIL (Python Imaging Library) η οποία περιέχει διάφορες λειτουργίες για την χρήση εικόνων σε προγράμματα Python, κάτι που μου έλυσε τα χέρια. Φόρτωσα λοιπών την εικόνα κατ’ ευθείαν σε έναν πίνακα, τον χώρισα σε κομμάτια, και την έστειλα, ένα κομμάτι τη φορά μέσω ενός βρόγχου αποστολής. Μετά από μικροδιορθώσεις, ο κώδικας ήταν έτοιμος.

1.3 Κώδικας και λειτουργία.

Παρ’ όλο που έχουμε δύο μηχανήματα, ο κώδικας είναι ο ίδιος και στα 2. Τα κουμπιά λειτουργούν ως ο αποστολέας, και ο ατέρμον βρόγχος δρα ως ο παραλήπτης, με το να «ακούει» συνεχώς για τυχών αποστολές στη συχνότητά του.

Όπως οι περισσότεροι κώδικές, έτσι και ο δικός μας ξεκινάει με την συμπερίληψη των βιβλιοθηκών. Εφ’ όσων χρησιμοποιούμε το bonnet της Adafruit, χρειαζόμαστε και κατάλληλες βιβλιοθήκες ώστε να έχουμε έλεγχο στα κουμπιά και την μικρή OLED οθόνη του. Η ρύθμιση των κουμπιών γίνεται ορίζοντας το πάτημα καθώς και το σήκωμα του κουμπιού. Στη ρύθμιση της οθόνης, ορίζουμε τις διαστάσεις καθώς και το πρωτόκολλο επικοινωνίας με αυτήν.

Έπειτα, την καθαρίζουμε ώστε να μην δείχνει τίποτα. Μετά σειρά έχει το κομμάτι του LoRa. Η συχνότητα ενός LoRa διαφέρει από Ήπειρο σε Ήπειρο. Σε άλλα μέρη του κόσμου όπως είναι η Αμερική, η συχνότητα που χρησιμοποιείται είναι στα 915 MHz, ενώ στην Ασία είναι τα 169 και τα 433 MHz. Εδώ, στην Ευρώπη, χρησιμοποιείται η συχνότητα των 868 MHz, όπως και το ρυθμίζουμε μέσα στο πρόγραμμα.

Ακολουθούν οι διεργασίες που θα χρησιμοποιηθούν αργότερα στον κώδικά μας, οι οποίες κυρίως ανοίγουν μία εικόνα μέσα σε έναν πίνακα στο πρόγραμμα, κόβουν τον πίνακα και τέλος ξανά συναρμολογούν την εικόνα και την αποθηκεύουν. Έπειτα, δηλώνουμε μεταβλητές που θα χρησιμοποιήσει το πρόγραμμά μας. Συνήθως, αυτές δηλώνονται στην αρχή του κώδικα αλλά επέλεξα να της βάλω λίγο πιο κοντά στα κομμάτια που τις χρησιμοποιούν.

Ακολουθεί ο ατέρμον βρόγχος. Όταν δεν συμβαίνει τίποτα, ένα μήνυμα αναβοσβήνει στην οθόνη και το μηχάνημα βρίσκεται σε λειτουργεία παραλαβής πακέτου. Όσο δεν λαμβάνει κάτι, παραμένει στην κατάσταση που αναφέρθηκε.

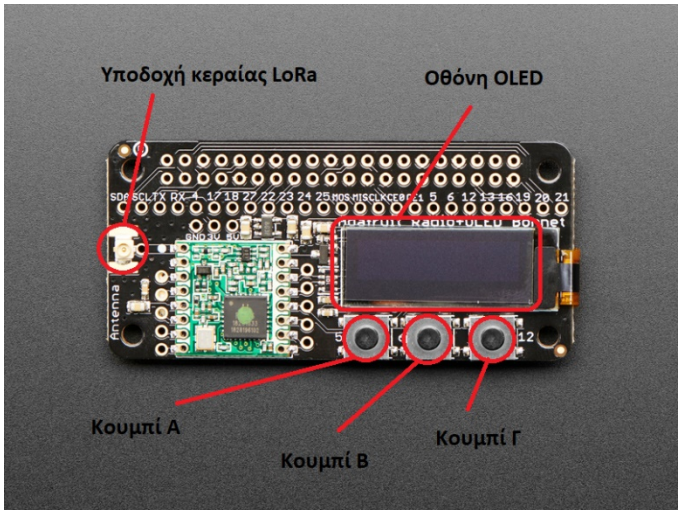
Εάν λάβει κάποιο πακέτο, το «κολλάει» σε έναν πίνακα και κρατάει τον αριθμό πακέτων που έχει λάβει μέχρι στιγμής με έναν μετρητή. Επίσης,

εμφανίζει και ανάλογο μήνυμα στην οθόνη πάνω στο μηχάνημα. Μέσα στο πρόγραμμα, μας δείχνει από τι αποτελείται το πακέτο το οποίο λάβαμε. Όλη αυτή η διαδικασία λαμβάνει τέλος όταν λάβει ένα συγκεκριμένο πακέτο το οποίο έχουμε ορίσει. Μόλις συμβεί αυτό, μας δίνει ένα μήνυμα τέλους, μηδενίζει τον μετρητή πακέτων και αποθηκεύει τον κολλημένο πίνακα σε έναν δεύτερο πίνακα, πριν διαγράψει τα περιεχόμενα του πρώτου.

Τέλος, ακολουθούν τα 3 κουμπιά, A, B και C. Πατώντας το κουμπί A, το πρόγραμμα φορτώνει μία εικόνα στη θέση αρχείου του προγράμματος με όνομα που έχουμε ορίσει από πριν και την βάζει σε έναν πίνακα από bytes, ο οποίος χωρίζεται σε μικρότερα κομμάτια ανά 200 χαρακτήρες για να αποσταλούν ομαλά. Μας λέει τον αριθμό κομματιών που προέκυψαν και βγάζει μήνυμα ετοιμότητας στην μικρή οθόνη.

Το κουμπί B ξεκινάει να στέλνει ένα-ένα τα κομμάτια του προηγούμενου πίνακα ανά τακτά χρονικά διαστήματα μόλις πατηθεί και μας ενημερώνει για κάθε ένα που στέλνει. Μόλις στείλει και το τελευταίο κομμάτι, στέλνει ένα πακέτο «κλειδί», το οποίο σημαίνει το τέλος της παραλαβής στη μηχανή λήψης.

Το τρίτο και τελευταίο κουμπί C, Περνάει τον πίνακα που έχει προκύψει από τον δέκτη, ενώνει τα στοιχεία του και τα μετατρέπει σε μία εικόνα, η οποία αποθηκεύεται στη θέση αρχείου του κώδικα. Έπειτα, μας δίνει ένα μήνυμα τέλους στην οθόνη.



Εικόνα 6. Επεξήγηση του Adafruit RFM9X

1.4 Χρήση του μηχανήματος

Αρχικά χρειάζεται να συναρμολογήσουμε το μηχάνημα. Εφ’ όσων εγκαταστήσουμε το Raspbian λογισμικό στο Raspberry μας, ακολουθώντας τις οδηγίες της που έρχονται μαζί με αυτό, κουμπώνουμε το Bonnet της Adafruit πάνω του, επάνω στο οποίο υπάρχει η θύρα σύνδεσης της κεραίας. Συνδέουμε μια πηγή τροφοδοσίας, (στην δικιά μας περίπτωση, τον φορτιστή-μετασχηματιστή που περιλαμβάνεται με το Raspberry) και ένα καλώδιο ethernet. Σε περίπτωση ανοιχτού δικτύου Wi-Fi, δεν χρειάζεται καλώδιο, εφ’ όσων υπάρχει ενσωματωμένη κεραία Wi-Fi στο μηχάνημα. Χρησιμοποιούμε την εφαρμογή PuTTY για να έχουμε πρόσβαση σε επίπεδο root, εάν χρειαστεί να δοθεί κάποιο δικαίωμα στον χρήστη μέσα στα Raspberry. Με τη σύνδεση απομακρυσμένης επιφάνειας εργασίας των Windows και τη χρήση του προγράμματος XRDP έχουμε πρόσβαση στην επιφάνεια εργασίας του κάθε μηχανήματος, όσο είναι συνδεδεμένο στο ίδιο δίκτυο με τον υπολογιστή στον οποίο το τρέχουμε και έχουμε γνώση της IP που έχει λάβει το κάθε μηχάνημα (ή του έχουμε αναθέσει, αν είναι στατική).

Αφού έχουμε πρόσβαση στο εσωτερικό του μηχανήματος, ξεκινάμε τον κώδικά μας. Επίσης, καλό θα είναι να έχουμε μέσα στον ίδιο φάκελο με το αρχείο του προγράμματος την εικόνα που επιθυμούμε να στείλουμε μέσω αυτού και να την έχουμε ονομάσει κατάλληλα, όπως ορίζουμε μέσα στον

κώδικα. Ξεκινώντας το πρόγραμμα και στα δύο μηχανήματα (αποστολέας και δέκτης) τα βάζουμε και τα δύο σε έναν ατέρμον βρόγχο σας παραλήπτες, εφ' όσον ο κώδικας λειτουργεί αμφίδρομα. Ξεκινάμε με το 1^{ο} μηχάνημα. Πατώντας το 1^{ο} κουμπί (αριστερά), ο κώδικας ψάχνει για μία εικόνα με συγκεκριμένο όνομα στην θέση αρχείου όπου βρίσκεται (φάκελος), την φορτώνει στη μνήμη του σαν πίνακα από bytes και την «κόβει» σε μικρότερους πίνακες για να μπορέσει να την στείλει παρ' όλου του μικρού εύρους μεγέθους αποστολής. Με το 2^{ο} κουμπί (μεσαίο) ο κώδικας ξεκινάει να στέλνει τους μικρούς «κομμένους» πίνακες που έχει στην μνήμη του έναν τη φορά, ανά τακτά χρονικά διαστήματα. Μετά την αποστολή του τελευταίου πίνακα, στέλνει μία λέξη κλειδί, η οποία σηματοδοτεί το τέλος της αποστολής. Καθ' όλη αυτή την ώρα, το 2^{ο} μηχάνημα συνεχώς ακούει για τυχών μηνύματα στη συχνότητα που του έχουμε ορίσει. Όταν ξεκινάει η αποστολή από το 1^{ο} μηχάνημα, το 2^{ο} λαμβάνει κάθε πίνακα και αρχίζει να τον προσθέτει στον προηγούμενο που έχει λάβει, δημιουργώντας έτσι τον αρχικό πίνακα. Όταν λάβει τη λέξη κλειδί, αποθηκεύει τον πίνακα στην μνήμη του και καθαρίζει τους προηγούμενους πίνακες που έχει λάβει ώστε να μην υπάρχει πρόβλημα σε περίπτωση που λάβει κάτι άλλο. Πατώντας τώρα το 3^{ο} κουμπί (δεξιά) στο 2^{ο} μηχάνημα, κάνοντας κάποιες τελευταίες αλλαγές στον πίνακα που έχουμε λάβει, τον μετατρέπει σε εικόνα, την οποία αποθηκεύει στην θέση αρχείου που βρίσκεται το αρχείο του κώδικα. Αυτή η λειτουργεία μπορεί να πραγματοποιηθεί και αντίστροφα, από το μηχάνημα 2 στο μηχάνημα 1, εφ' όσον ο κώδικας που τρέχει καθώς και το υλικό τους είναι το ίδιο.

Εικόνα 7. Επεξήγηση Raspberry Pi 4

1.5 Ενδεικτικές εφαρμογές και μελλοντική χρήση.

Ένα τέτοιου είδους πρόγραμμα μπορεί να έχει αρκετές εφαρμογές ανάλογα με τις περιστάσεις. Για παράδειγμα, με τη χρήση ενός Gateway και το πρωτόκολλο LoRaWAN, θα μπορούσε να χρησιμοποιηθεί σε ένα πεδίο παρατήρησης με μια ενσωματωμένη κάμερα η οποία τραβάει φωτογραφίες ανά ένα χρονικό διάστημα. Μόλις η φωτογραφία τραβηχτεί, ξεκινάει η αποστολή. Αυτό θα βοηθούσε σε περιβαλλοντικές μελέτες σε μέρη όπου δεν υπάρχουν αρκετές ενεργειακές υποδομές ή είναι σχετικά απρόσιτες. Επίσης, με την ίδια τεχνολογία, θα μπορούσε να χρησιμοποιηθεί σε συστήματα ασφαλείας ως έξτρα βήμα. Μόλις ενεργοποιηθεί το σύστημα, μία φωτογραφία να παίρνεται και να στέλνεται μέσω δικτύου. Επίσης, σε περίπτωση απώλειας δικτύου, επιτρέπεται η αποστολή εικόνων από μέρος σε μέρος εάν είναι εντός της εμβέλειας των

κεραιών. Αν κάποιος θέλει να έχει οπτική πρόσβαση σε ένα μέρος κοντά στο σπίτι του ανά τακτά χρονικά διαστήματα, θα μπορούσε να χρησιμοποιήσει μία πλακέτα συνδεδεμένη σε μία κάμερα για να έχει εικόνες του μέρους αυτού χωρίς να χρειάζεται να μεταβεί εκεί. Με χρήση δικτύου, αυτό θα μπορούσε να γίνει και σε ακόμα μεγαλύτερες αποστάσεις.

Εικόνα 8. Κεραία LoRa Reach σε βουνοκορφή

Πηγή: **https://blog.emlid.com/pushing-the-limits-of-lora/**

1.6 Υλικό και λογισμικό που χρησιμοποιήθηκε.

Η τελική μορφή αυτής της διατριβής αποτελείται από 2 ίδια μηχανήματα, τα οποία έχουν το κάθε ένα τα εξής κομμάτια υλικού:

1. 1 πλακέτα Raspberry Pi 4 μοντέλου B μνήμης 2GB της Raspberry Pi Foundation.
2. 1 επιπρόσθετο LoRa Radio Bonnet με οθόνη OLED μοντέλου RFM95W της Adafruit.
3. 1 κεραία LoRa συχνότητας 868 Mhz.

Η χρήση των μηχανημάτων έγινε με τη χρήση:

1. Λειτουργικό σύστημα Raspberry Pi OS (πρώην Raspbian).

2. Το λογισμικό XRDP της neutrinolabs για τη χρήση τερματικού και δικαιώματα SUDO.
3. Τη σύνδεση απομακρυσμένης επιφάνειας εργασίας των Windows 10 (Με σύνδεση σε κοινό δίκτυο με τα μηχανήματα).

Οι κώδικες γράφτηκαν, εκτελέστηκαν και διορθώθηκαν στο προγραμματιστικό περιβάλλον Thonny της Cybernetica AS.

Συμπεράσματα

Το πρωτόκολλο LoRa αν και φαίνεται ανούσιο και απαρχαιωμένο στην σημερινή τεχνολογικά εξελιγμένη εποχή που διανύουμε δεν είναι τελικά και τόσο άσχημο. Ακούμε συχνά για ενεργειακές κρίσεις ανά τον κόσμο και μία τέτοιου είδους τεχνολογία θα μπορούσε, αν όχι να αποτρέψει, να ελαττώσει αυτά τα πιθανά σενάρια κρίσης.

Ακόμα και σήμερα, τα «έξυπνα σπίτια», οι μετρητές ρύπανσης, συστήματα πυρασφάλειας και πολλά άλλα βρίσκονται ήδη γύρο μας, πολλά εκ των οποίων χρησιμοποιούν το πρωτόκολλο αυτό. Σε μέρη όπως η Αμερική, η αγορά των «Έξυπνων σπιτιών» προβάλει χρήση 57% των νοικοκυριών, με κέρδη έως και 44 δισεκατομμύρια δολάρια μέχρι το 2025. [9] Σήμερα, υπάρχουν πάνω από 260 εκατομμύρια τέτοια σπίτια, με

ραγδαίες ανόδους στους αριθμούς αυτούς. Στην Ευρώπη, ο αριθμός ανέρχεται στα 63.1 εκατομμύρια σπίτια μέσα στο 2022 [10].

Δεν είναι μακριά ένα μέλλον όπου τέτοιου είδους συστήματα θα βρίσκονται παντού, σε κάθε σπίτι, κάθε κτήριο, κάθε γωνιά του πλανήτη, κάνοντας τη ζωή μας ευκολότερη. Οπουδήποτε δεν μπορεί να φτάσει ένα καλώδιο αλλά μπορεί να φτάσει ένας άνθρωπος, μία μικρή πλακέτα, μια μικρή κεραία και μία μπαταρία, θα μπορεί να υπάρχει χρήση του πρωτοκόλλου αυτού.

Βιβλιογραφία

1] S. MONTAGNY, «Savoie Mont Blanc University,» July 2022. [Ηλεκτρονικό]. Available: https://www.univ-smb.fr/lorawan/en/free-book/.

2] Semtech, «LoRa Developer Portal,» Semtech, [Ηλεκτρονικό]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/.

3] Semtech, [Ηλεκτρονικό]. Available: https://www.semtech.com/lora/what-is-lora.

4] «LoRa Alliance,» [Ηλεκτρονικό]. Available: https://lora-alliance.org/.

5] «Kudzu Technologies,» [Ηλεκτρονικό]. Available: https://www.kudzu.gr/.

6] Sodaq, «Overview - ONE Board,» [Ηλεκτρονικό]. Available: https://support.sodaq.com/Boards/One/.

7] Rebound EU, «Embedded Systems Explained,» [Ηλεκτρονικό]. Available: https://reboundeu.com/insights/blog/embedded-systems-explained/.

8] Raspberry Pi Foundation, «What is a Raspberry Pi,» [Ηλεκτρονικό]. Available: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/.

9] B. Thormundsson, «Smart home in the United States - statistics & facts,» statista.com, 2023.

10] Berg Insight, «EU smart home market not as mature as US, but growing fast,» 2022.

Παράρτημα Κώδικα

```
# Import Python System Libraries
import time
# Import Blinka Libraries
import busio
from digitalio import DigitalInOut, Direction, Pull
import board
# Import the SSD1306 module.
import adafruit_ssd1306
# Import RFM9x
import adafruit_rfm9x
# Import base64 to encode image file
import base64
# Import PIL
from PIL import Image
import io

# Button A Configuration
btnA = DigitalInOut(board.D5)
btnA.direction = Direction.INPUT
btnA.pull = Pull.UP

# Button B Configuration
btnB = DigitalInOut(board.D6)
btnB.direction = Direction.INPUT
btnB.pull = Pull.UP
```

```
# Button C Configuration
btnC = DigitalInOut(board.D12)
btnC.direction = Direction.INPUT
btnC.pull = Pull.UP

# Create the I2C interface
i2c = busio.I2C(board.SCL, board.SDA)

# 128x32 OLED Display Configuration
reset_pin = DigitalInOut(board.D4)
display = adafruit_ssd1306.SSD1306_I2C(128, 32, i2c, reset=reset_pin)

# Clear the display
display.fill(0)
display.show()
width = display.width
height = display.height

# Configure LoRa Radio
CS = DigitalInOut(board.CE1)
RESET = DigitalInOut(board.D25)
spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
rfm9x = adafruit_rfm9x.RFM9x(spi, CS, RESET, 868.0)      #EU LoRa
frequency#
rfm9x.tx_power = 23
prev_packet = None
```

```
##################################################################
####################

# Function that turns a byte array to an image and saves it.
def bytearraytoimage(byte_array):
    stream = io.BytesIO(byte_array)
    image = Image.open(stream)
    image.save('LastReceivedImage.jpg')

# Function that reads a saved image into the code.
def openimagefile():
    image_path = 'nireasjpg2.jpg' #Change depending on image#
    with open(image_path, 'rb') as file:
        image_data = file.read()
    return image_data
# Function that takes a byte array and splits it into smaller arrays.
def splitbytearray(byte_array, chunk_size):
    return [byte_array[i:i + chunk_size] for i in range (0,
len(byte_array),chunk_size)]

##################################################################
# Variables
receivecounter = 0
receivedarray = []
savedarray = []
mytxend = "txend"
mytxend_bytes = bytes(mytxend, 'utf-8')
```

```
##################################################################
###############

# Receiver Loop
while True:
    packet = None
    display.fill(0)
    display.text('RasPi LoRa', 35, 0, 1)
    packet = rfm9x.receive()
    if packet is None:
        display.show()
        display.text('- Waiting for PKT -', 15, 20, 1)
    else:
        display.fill(0)
        if packet != mytxend_bytes:
            receivedarray.extend(packet)
            receivecounter = receivecounter + 1
            print ("\nReceived Chunk: "+str(receivecounter))
            prev_packet = packet
            packet_text = str(prev_packet)
            display.text('RX: ', 0, 0, 1)
            display.text(str(receivecounter), 25, 0, 1)
            print (prev_packet)
        elif packet == mytxend_bytes:
            savedarray = receivedarray
            receivedarray = []
            print ("\nReceived "+str(receivecounter)+" Chunks.")
```

```
            receivecounter = 0
            print ("Transmition is over.")
            display.text('Transmition is over.', 25, 0, 1)

##################################################################
#####
#time.sleep(1)

# Button Configuration
    if not btnA.value:
        transmition = []
        display.fill(0)
        display.text('Image loaded!', 25, 15, 1)
        mybytearray = openimagefile()
        transmition = splitbytearray(mybytearray, 200)
        print(mybytearray)
        print("Array size = " + str(len(mybytearray)))
        print(transmition)
        print("\nChunks to transmit = " + str(len(transmition)))
    elif not btnB.value:
        display.fill(0)
        txcount = 0
        while txcount < len(transmition):
            button_b_data = transmition[txcount]
            rfm9x.send(button_b_data)
            print('\nChunk: ' + str(txcount+1) + ' of ' + str(len(transmition)) + '\n'
+ str(transmition[txcount]))
            txcount = txcount + 1
```

```
            time.sleep(1)
        rfm9x.send(mytxend_bytes)
        display.text('\nImage Sent!\r\n', 25, 15, 1)
    elif not btnC.value:
        display.fill(0)
        convertedarray = bytearray()
        convertedarray.extend(savedarray)
        finalarray = bytes(convertedarray)
        bytearraytoimage(finalarray)
        display.text("Image Saved", 25, 15, 1)

    display.show()
    time.sleep(0.1)
```