KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

KESEPAKATAN BIMBINGAN TUGAS AKHIR (TA)

Kami yang bertanda tangan dibawah ini,

Pihak Pertama

Nama	: Redho Prasetya Uganda
NIM	: 061940352365
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pihak Kedua

Nama	: Mohammad Fadhli, S.Pd., M.T
NIP	: 199004032018031001
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pada hari ini SABTU ..... tanggal..... 06 MEI ..... 2023 telah sepakat untuk melakukan konsultasi bimbingan Tugas Akhir (TA).

Konsultasi bimbingan sekurang- kurangnya 1 (satu) kali dalam satu minggu. Pelaksanaan bimbingan pada setiap hari ..... KERJA ..... pukul..... 09-10 ..... tempat di Politeknik Negeri Sriwijaya.

Demikianlah kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas Akhir.

Palembang, 2023

Pihak Pertama	Pihak Kedua
(Redho Prasetya Uganda)	(Mohammad Fadhli, S.Pd., M.T)
NIM. 061940352365	NIP. 199004032018031001

Mengetahui
Ketua Jurusan Teknik Elektro

(Ir. Iskandar Lutfi, M.T)
NIP. 196501291991031002

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

KESEPAKATAN BIMBINGAN TUGAS AKHIR (TA)

Kami yang bertanda tangan dibawah ini,

Pihak Pertama

Nama	: Redho Prasetya Uganda
NIM	: 061940352365
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pihak Kedua

Nama	: Sopian Soim, S.T., M.T
NIP	: 197103142001121001
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pada hari ini SABTU tanggal 06 MEI 2023 telah sepakat untuk melakukan konsultasi bimbingan Tugas Akhir (TA).

Konsultasi bimbingan sekurang- kurangnya 1 (satu) kali dalam satu minggu. Pelaksanaan bimbingan pada setiap hari KERJA pukul 09 - 10 tempat di Politeknik Negeri Sriwijaya.

Demikianlah kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas Akhir.

Palembang, 2023

Pihak Pertama	Pihak Kedua
(Redho Prasetya Uganda)	(Sopian Soim, S.T., M.T)
NIM. 061940352365	NIP. 197103142001121001

Mengetahui
Ketua Jurusan Teknik Elektro

(Ir. Iskandar Lutfi, M.T)
NIP. 196501291991031002

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

LEMBAR BIMBINGAN TUGAS AKHIR

Lembar : 1

Nama : Redho Prasetya Uganda

NIM : 061940352365

Jurusan/Program Studi : Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi

Judul Tugas Akhir : Penerapan Protokol *CSMA-CA* Pada *Transceiver* Komunikasi LoRa Dalam Perancangan *Monitoring* Kualitas Udara

Pembimbing I : Mohammad Fadhli, S.Pd., M.T

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
1.	8/5 2023	Konsultasi program Arduino IDE Menggunakan Example LMIC	
2.	17/5 2023	Konsultasi program Arduino IDE Menggunakan Example sensor MQ7	
3.	29/5 2023	Konsultasi Program Arduino IDE Menggunakan Example LoRa Sender	
4.	31/5 2023	Konsultasi Program Arduino IDE Menggunakan Example LoRa Receiver	
5.	9/6 2023	Revisi Perancangan Alat Menggunakan Lora 433 MHz dirubah ke 915	
6.	14/6 2023	Konsultasi Antena Lora 433 MHz dirubah menjadi Antena Heliks	
7.	17/6 2023	Pengajuan BAB I	
8.	13./7 2023	Revisi BAB I Latar Belakang	

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
9.	24 / 7 2023	Bimbingan Jurnal Pengajuan BAB II	
10.	28 / 7 2023	Revisi BAB II Rumus perhitungan Packet Loss	
11.	29 / 7 2023	BAB III Metodologi Penelitian Bimbingan Jurnal revisi	
12.	31 / 7 2023	Revisi BAB III Kerangka Penelitian dan Skema Rangkaian Arduino UNO	
13.	1 / 8 2023	Revisi BAB III ACC BAB IV Hasil pengujian RSSI dan Packet Loss	
14.	10 / 8 2023	Revisi BAB V Kesimpulan dan Saran	
15.	10 / 8 2023	ACC	

Palembang, Agustus 2023
KPS Sarjana Terapan Teknik Telekomunikasi

(Lindawati, S.T., M.T.I)
NIP. 197105282006042001

Catatan:
*) melingkari angka yang sesuai.
Ketua Jurusan/Ketua Program Studi harus memeriksa jumlah pelaksanaan bimbingan sesuai yang dipersyaratkan dalam Pedoman Tugas Akhir sebelum menandatangani lembar bimbingan ini.
Lembar pembimbingan Tugas Akhir ini harus dilampirkan dalam Tugas Akhir.

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

LEMBAR BIMBINGAN TUGAS AKHIR

Lembar : 1

Nama : Redho Prasetya Uganda

NIM : 061940352365

Jurusan/Program Studi : Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi

Judul Tugas Akhir : Penerapan Protokol *CSMA-CA* Pada *Transceiver* Komunikasi LoRa Dalam Perancangan *Monitoring* Kualitas Udara

Pembimbing II : Sopian Soim, S.T., M.T

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
1.	10/5 2023	Diskusi dataset yang digunakan pada model	
2.	17/5 2023	Revisi Latar Belakang	
3.	23/5 2023	Revisi Rumusan Masalah	
4.	23/06 2023	Konsultasi Program Arduino IDE Menggunakan Example Lora Sender	
5.	26/6 2023	Acc BAB I	
6.	4/7 2023	Bimbingan BAB II	
7.	11/7 2023	Pengajuan BAB III	
8.	13/7 2023	Revisi diagram metodologi dan proses permodelan GNS3	

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
9.	18/7 2023	Pengajuan BAB IV & V	
10.	24/7 2023	Diskusi perancangan Alat.	
11.	28/7 2023	Revisi BAB IV hasil Packet Loss, RSSI	
12.	2/8 2023	Revisi Hasil Grafik, Packet Loss, RSSI	
13.	4/8 2023	ACC BAB IV	
14.	7/8 2023	ACC BAB V kesimpulan & saran.	
15.	11/8 2023	Acc Sidang Ujian.	

Palembang, Agustus 2023
KPS Sarjana Terapan Teknik Telekomunikasi

(Lindawati, S.T., M.T.I)
NIP. 197105282006042001

Catatan:
*) melingkari angka yang sesuai.
Ketua Jurusan/Ketua Program Studi harus memeriksa jumlah pelaksanaan bimbingan sesuai yang dipersyaratkan dalam Pedoman Tugas Akhir sebelum menandatangani lembar bimbingan ini.
Lembar pembimbingan Tugas Akhir ini harus dilampirkan dalam Tugas Akhir.

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

REKOMENDASI UJIAN TUGAS AKHIR

Pembimbing Tugas Akhir memberikan rekomendasi kepada,

Nama	:	Redho Prasetya Uganda
NIM	:	061940352365
Jurusan/Program Studi	:	Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi
Judul Tugas Akhir	:	Penerapan Protokol *CSMA-CA* Pada *Transceiver* Komunikasi LoRa Dalam Perancangan *Monitoring* Kualitas Udara

Mahasiswa tersebut telah memenuhi persyaratan dan dapat mengikuti Ujian Tugas Akhir (TA) pada Tahun Akademik 2022/2023

Palembang, Agustus 2023

Pembimbing I,

Mohammad Fadhli, S.Pd., M.T
NIP. 199004032018031001

Pembimbing II,

Sopian Soim, S.T., M.T
NIP. 197103142001121001

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

PELAKSANAAN REVISI TUGAS AKHIR

Mahasiswa berikut,

Nama	:	Redho Prasetya Uganda
NIM	:	061940352365
Jurusan/Program Studi	:	Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi
Judul Tugas Akhir	:	Penerapan Protokol *CSMA-CA* Pada *Transceiver* Komunikasi LoRa Dalam Perancangan *Monitoring* Kualitas Udara

Telah melaksanakan revisi terhadap Tugas Akhir yang diujikan pada hari Rabu tanggal 16 bulan Agustus tahun 2023. Pelaksanaan revisi terhadap Tugas Akhir tersebut telah disetujui oleh Dosen Penguji yang memberikan revisi:

No.	Komentar	Nama Dosen Penguji*)	Tanggal	Tanda Tangan
1.	Sudah revisi	Ir. Jon Endri, M.T NIP. 196201151993031001	24/08/2023	
2.	ACC	Hj. Adewasti, S.T., M.Kom NIP. 197201142001122001	24/08/2023	
3.	ACC	Sopian Soim, S.T., M.T NIP. 197103142001121001	29/08/2023	
4.	ACC	RA. Halimatussa'diyah, S.T., M.Kom NIP. 197406022005012002	29/08/23	

Palembang, Agustus 2023
Ketua Penguji **),

(Ir. Jon Endri, M.T)
NIP 196201151993031001

Catatan:
*) Dosen penguji yang memberikan revisi saat ujian Tugas Akhir.
**) Dosen penguji yang ditugaskan sebagai Ketua Penguji saat ujianTugas Akhir.
Lembaran pelaksanaan revisi ini harus dilampirkan dalam Tugas Akhir.

JTIP UNP
Jurnal Teknologi Informasi dan Pendidikan
Department of Electronics, Faculty of Engineering
Universitas Negeri Padang
Jln. Prof. Dr. Hamka Air Tawar Padang, 25132
E-mail. tip@ppj.unp.ac.id
Website. http://tip.ppj.unp.ac.id

LETTER OF ACCEPTANCE

No. 73/UN35/JTIP-LOA/Acc/2023

Dear Redho Prasetya Uganda,

No. 773

All Authors

Redho Prasetya Uganda, Mohammad Fadhli, Sopian Soim

Article Title

"Application of *CSMA-CA* Protocol on LoRa Communication *Transceiver* In Air Quality *Monitoring* Design"

Based on the recommendations from the peer review board, we are delighted to inform you that your following manuscript has been **ACCEPTED** for possible publication in **Jurnal Teknologi Informasi dan Pendidikan (JTIP) Vol. 16, No. 2, (2023)**.

Thank you for making the journal a vehicle for your research interests.

August 07, 2023

Best wishes,

Chief Editor

(Jurnal Teknologi Informasi dan Pendidikan)

Mobile Phone. +628 1363 609 995

This journal has been nationally **accredited Sinta 3** based on SK No. 200/M/KPT/2020 (23 Desember 2020).

Jurnal Teknologi Informasi dan Pendidikan
Volume 15, No. 1, Maret 2022
https://doi.org/10.24036/tip.v15i1

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

Application of *CSMA-CA* Protocol on LoRa Communication *Transceiver* In Air Quality *Monitoring* Design

Redho Prasetya Uganda[1*], Mohammad Fadhli[2], Sopian Soim[3]
[1]Electrical Engineering, Sriwijaya State Polytechnic, Palembang, Indonesia
[2]Electrical Engineering, Sriwijaya State Polytechnic, Palembang, Indonesia
[3]Electrical Engineering, Sriwijaya State Polytechnic, Palembang, Indonesia
**Corresponding Author: redhoprasetyauganda@gmail.com*

Article Information

Article history:

No. 773
Rec. August 5, 2023
Rev. August 5, 2023
Acc. August 7, 2023
Pub. Month dd, yyyy
Page. 1 – 11

Keywords:

- *CSMA-CA*
- *LoRa (Long Range)*
- *Air Quality Monitoring*
- *Wireless Communication technology*
- *Communication Efficiency*
- *Data Accuracy*

ABSTRACT

In this research, the application of the CSMA-CA protocol to the LoRa communication transceiver has been successfully applied in the design of an air quality monitoring system. The use of the CSMA-CA protocol helps optimize the communication process between devices in the LoRa network, reduce the potential for data collisions, and increase the efficiency of communication channel usage. The test results show that this system is able to transmit data with a high success rate at a relatively short to medium distance of 50-350m. However, at longer distances, there is a degradation in performance with packet loss and decreased RSSI values. Therefore, it is recommended to consider using repeaters or signal amplification at longer distances to ensure reliable data transmission. In addition, system integration can also improve monitoring efficiency and assist in decision-making based on the data generated by the monitoring system.

1. INTRODUCTION

In recent times, air pollution has become a more common environmental problem, especially in big cities filled with factories and vehicles. Since it can have adverse effects on public health, air pollution must be prevented and reduced. One way to do this is by monitoring the air quality at a particular location. This air quality monitoring system using wireless sensor networks is designed to remotely monitor air quality, and the ISPU value is displayed through a web application. This research uses a methodology consisting of four stages: identification of system requirements, system design, system implementation; testing; and analysis. The system configuration with Arduino Uno as the control center. In addition, this system uses the LoRa hardware communication method and the MQ-7 sensor to measure CO levels, with the web application of this system, users can know if air pollution is increasing in a place. The test results show that this system can read the sensor

according to the datasheet and send data to the web application using LoRa communication up to a distance of 300 meters with a packet loss ratio of 0% [1].

In computer networks, CSMA-CA (*Carrier Sense Multiple Access*) or *Collision Avoidance* is one of several access methods that use scheme sensing. When a *node* wants to transmit data, it must first look at the channel time for a predetermined amount to see if other *node* can transmit on the same channel within wireless range. If the channel is not working, the *node* is allowed to start transmission, but if the channel appears busy, the *node* will send data for an unspecified time delay. Once the transmission process is started, the actual data transmission of the application can still be delayed. With this method, a network *node* that is about to send data to the destination *node* first ensures that the network is not being used for data transfer by other *node*. If at the checking stage another data transmission is found and a (*collision*) occurs, then the *node* has to repeat the transmission (*request*) at the next interval, which is done randomly. In this way, the network can be effectively utilized in a randomized manner [2].

The technology known as "gateway" facilitates communication between the company's two main server devices. It directs traffic from workstations to the outside network, which can be found in almost every city in Indonesia. However, as *gateway* technology has limitations, it began to evolve rapidly with LoRa (*Long Range*) systems. Such limitations include *gateway* being complicated to design and implement, high cost for implementation, and requiring specialized system administration configurations [3].

The Low Power Wide Area Network (LPWAN) communication system called LoRa, developed by IBM, Semtech, Actility, and other companies in the LoRa Alliance, has long-distance transmission capabilities. LoRa is scientifically defined as the process of converting certain periodic waves into signals capable of carrying information. This repetitive and regular wave change has a gradual source of interference or vibration. The periodic wave change is called modulation [4]. In most cases, public networks using LoRa networks have greater signal coverage than typical cellular networks. Some of the devices that support LoRa technology are network adapters, USB ports, external antennas, routers, servers, network cables, computer devices, access points, and modems [5].

How to implement and run the analysis of the CSMA-CA method on the multinode LoRa MAC *Layer* communication protocol is one of the problems encountered during the development of *Long Range* technology. However, server-based and Android applications on Firebase can be easily accessed for implementation and analysis for the general public [6].

Many people in Indonesia are using LPWAN (*Low Power Wide Area Network*), which is a new trend in the rapidly growing field of Internet of Things (*IoT*). Due to the large number of uses, the government decided to limit the use of LPWAN to frequencies of 920-923 MHz. Currently, a more practical and cheaper NodeMCU hardware called ESP8266 is available. It is equipped with a chipset that can be used for programming and power supply usage. Previous studies found that the NodeMCU has many disadvantages, one of which is

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

the limited number of pin legs. This is because the NodeMCU is a microcontroller that is already integrated with the ESP8266. There are only 17 analog pins on the NodeMCU [7].

This implementation and analysis will require a Firebase server to display the 915 MHz content data of the target processed in CSMA-CA. The LoRa signal relay system will be made using the MAC *layer* and the addition of a helical antenna that functions as a receiver of signals that have been sent by the MAC *layer* [8].

In previous research on soil temperature and humidity monitoring, research was conducted using two types of LoRa frequencies, namely 433 MHz and 915 MHz, in 2016. Although the Minister of Communication and Information has determined that the 915 MHz frequency will be used for research, research was also conducted on the 433 MHz frequency as a performance comparison in 2017. In tests conducted at frequencies of 433 MHz and 915 MHz, it was seen that the RSSI value increased to -120 dB, SNR (Signal to Noise Ratio) increased to -20 dB, and PL decreased to 3%. These results show that LoRa technology can be used as a surveillance system communication network in Samarinda City in 2018. In addition, it has been found that LoRa performance at 433 MHz frequency is better than at 915 MHz frequency. Likewise with the results of the Yagi-Uda antenna. Previous studies have shown better and consistent results and progress compared to the rubber duck antenna [9].

Additional research is needed due to the need and prospect of LoRa research. Therefore, the author will conduct research by simulating data transmission through LoRa technology. The author analyzes the performance of LoRa applied to CO levels in air quality using predetermined test parameters. The data transmission process from NodeMCU ESP8266 LoRa Transmit to the microcontroller with LoRa receiver is done point-to-point. Furthermore, the data is sent serially to the microcontroller to be processed into PPM (Part Per Million) values. Furthermore, the data is sent in real-time to Firebase as a cloud server and website database as numerical data [10].

2. RESEARCH METHOD

This research uses the experimental method as a scientific approach to test and identify cause-and-effect relationships between variables in the designed air quality monitoring system. The research objectives are to validate the effectiveness and reliability of the system in measuring and monitoring air quality with high accuracy [11], as well as exploring the potential application of LoRa technology and CSMA-CA protocol in environmental monitoring at large.

Jurnal Teknologi Informasi dan Pendidikan
Volume 15, No. 1, Maret 2022
https://doi.org/10.24036/tip.v15i1

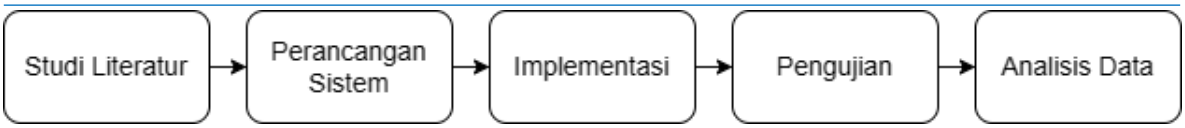


Figure 2.1 Experimental Research Methodology

Figure 2.1 depicts the steps of the experimental research methodology that reflects the systematic approach in this study. The first step is a literature study on LoRa technology, CSMA-CA protocol, and previous air quality monitoring applications to design a novel and efficient air quality monitoring system. The second step involves system design which includes the selection of suitable air quality sensors, configuration of LoRa communication devices, and adaptation of the CSMA-CA algorithm for this application.

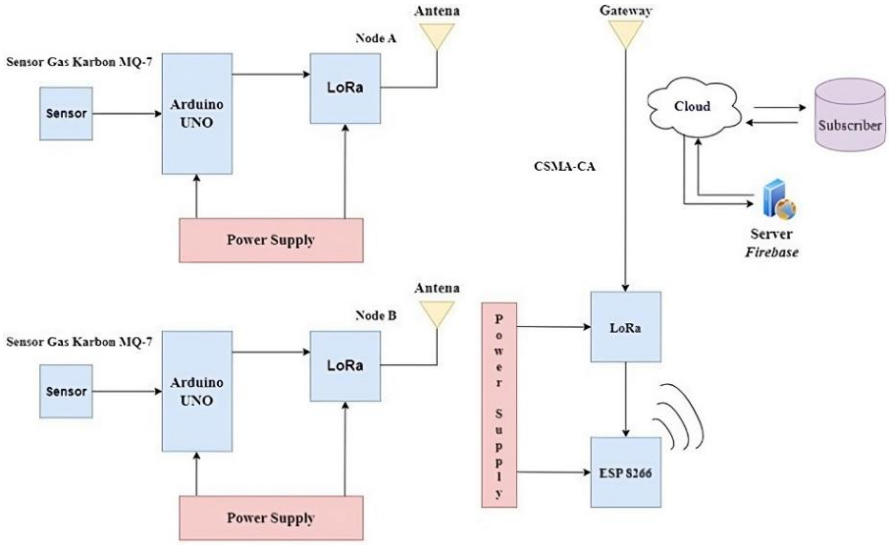
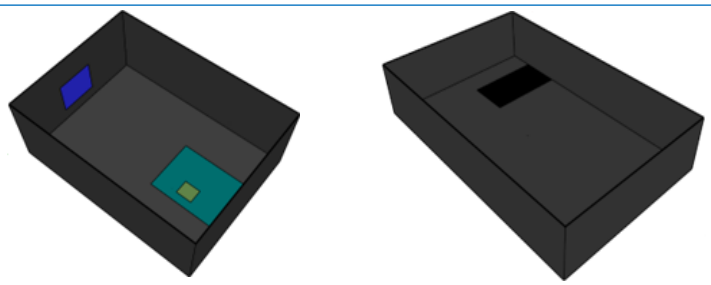


Figure 2.2 Hardware Circuit Block Diagram (*Hardware*)

The air quality monitoring system uses LoRa technology with main components such as MQ-7 gas sensor, Arduino UNO, LoRa, and antenna. LoRa acts as a transmitter to send sensor data to the Gateway, which then forwards the data to the Firebase server. The system implementation involves creating a hardware design using a 3D printer to print the hardcase. The goal is to improve the quality and creativity value of the air quality monitoring system using LoRa technology with the CSMA-CA protocol.

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id



(a) **(b)**

Figure 2.3 (a) Sketch Tool *Node* 1 & 2 **(b)** Sketch Tool *Gateway*

After system implementation, the next step is to conduct thorough testing. The goal is to ensure that the air quality monitoring system with LoRa technology and CSMA-CA protocol functions in accordance with the research objectives. Testing involves communication between LoRa transceivers, data collection from air quality sensors, and analysis of the resulting data. The reliability of the CSMA-CA protocol in managing network access and minimizing data collisions will be tested to ensure the efficiency and accuracy of the system in monitoring air quality. The test results will be the basis of evaluation for improvement and optimization of the system before it is applied in the field more widely.

Packet Loss is a condition in which some data packets are lost or fail to reach their destination while being sent over a communication network [12]. *Packet loss* can be calculated with the following equation :

$$Packet\ Loss = (\frac{(Packages\ Sent - Received\ Packages)}{Packages\ Shipped} \times 100\% \qquad (1)$$

Packet loss categories are shown in the following table:

Table 2.1 Criteria *Packet Loss*

Tiers	*Packet Loss (%)*	Index
Very Good	0 %	4
Good	3 – 14 %	3
Medium	15 – 24 %	2
Bad	> 25 %	1

The RSSI (*Received Signal Strength Indicator*) parameter is used to measure the strength of the radio signal received by the receiver from the sender. RSSI measures signal strength in units of decibel-milliwatts (dBm) or in some devices with a numerical scale. The higher the RSSI value, the stronger the received signal, and the better the quality of the wireless connection [13].

Table 2.2 Criteria *RSSI*

Criteria RSSI	Value Range RSSI (dBm)	Index
Very Good	> -50 dBm	4
Good	-50 dBm s/d -70 dBm	3
Simply	-70 dBm s/d -80 dBm	2
Weak	-80 dBm s/d -90 dBm	1
Very Weak	< -90 dBm	0

After data is collected from the monitoring system, the next step is to analyze the data to evaluate the measured air quality and the overall performance of the system. The results of this data analysis become the basis for drawing conclusions about the effectiveness and reliability of the air quality monitoring system using LoRa technology and the CSMA-CA protocol.

3. RESULTS

The results in this study are in the form of an application of the design that has been made before. The results of this research focus on implementing the MAC *Layer* method as a Collision Avoidance system with the CSMA-CA protocol to prevent data collisions during transmission. Tests were conducted to test the stability of transmission with a collision avoidance feature that is very vulnerable when data transmission involves more than one *node*.

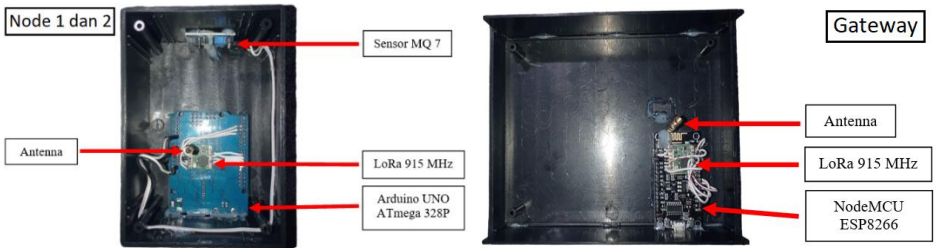


Figure 3.1 Hardware Design Results

The design of the air quality monitoring system based on the MAC *Layer* method in Figure 3.2 is generally divided into 2 (two), namely the transmitter and receiver. The transmitter (*node*) consists of an Arduino Uno microcontroller, sensors to measure air quality and a LoRa (*Long Range*) module to send measurement data, while the receiver (*gateway*) consists of a NodeMCU ESP8266 microcontroller connected to the internet network, a LoRa (*Long Range*) module to receive data from the transmitter, and firebase to display measurement data.

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

Table 3.1 *Packet Loss* Testing with CSMA-CA Implementation

Distance (m)	Data Transmission		Packet Loss (%)	Quality (Index)
	Tx	Rx		
50	47	47 = 0	0.00	4
100	59	59 = 0	0.00	4
150	70	70 = 0	0.00	4
200	80	79 = 1	1.25	4
250	90	89 = 1	1.11	4
300	100	99 = 1	1.00	4
350	110	108 = 2	1.82	4
400	120	111 = 9	7.50	3
450	130	120 = 10	7.69	3
500	140	121 = 19	13.57	3
550	150	125 = 25	16.67	2
600	160	130 = 30	18.75	2
650	170	136 = 34	20.00	2
700	180	146 = 34	18.89	2
750	190	153 = 37	19.47	2
800	200	160 = 40	20.00	2
850	210	170 = 40	19.05	2
900	240	181 = 59	24.58	1
950	270	191 = 79	29.26	1
1000	300	191 = 109	36.33	1

In table 3.1 the *Packet Loss* test results provide an overview of system performance at different distances. The longer the distance, the transmission (TX/RX) tends to decrease, accompanied by an increase in packet loss and a decrease in the quality index. At distances of 200 meters and 1000 meters, the transmission and quality index reached 1, indicating that the system was having problems transmitting data at these distances. It needs to be addressed with solutions such as the use of repeaters or signal amplification to maintain the quality and reliability of the system at longer distances.

Table 3.2 *RSSI* Testing with CSMA-CA Implementation

Distance (m)	RSSI (dBm)			Quality (Index)
	Max	*Min*	*Mean*	
50	-60	-71	-65,5	3
100	-76	-71	-73,5	2
150	-73	-84	-78,5	2
200	-73	-76	-74,5	2
250	-79	-76	-77,5	2
300	-82	-83	-82,5	1
350	-82	-84	-83	1
400	-81	-80	-80.5	1

450	-85	-80	-82.5	1
500	-84	-81	-82.5	1
550	-83	-82	-82.5	1
600	-80	-85	-82.5	1
650	-86	-82	-83.5	1
700	-88	-84	-85.5	1
750	-90	-85	-87.5	1
800	-87	-88	-87.5	1
850	-90	-88	-88.5	1
900	-88	-89	-88.5	1
950	-89	-89	-89	1
1000	-92	-89	-90.5	0

Table 3.2 contains the test results of RSSI (Received Signal Strength Indicator) and Quality (Index) at a distance of 50 to 1000 meters. At a distance of 50 meters, RSSI reaches a maximum value of -60 dBm and Quality (Index) reaches 3, indicating good signal quality. However, as the distance increases, the RSSI value tends to decrease. At distances of 100 to 850 meters, RSSI values fluctuate with Quality (Index) stabilizing at 1 or 2, indicating relatively good signal quality. However, at a distance of 900 to 1000 meters, the RSSI value decreases significantly and the Quality (Index) becomes 0, indicating a weak signal and potential problems in data transmission. The analysis shows that the communication system performs well at short to medium distances, but requires solutions such as the use of repeaters or signal amplification to ensure reliable data transmission at longer distances.

Tabel 3.3 Carbon Monoxide air quality sensor response testing

Pollutant Testing	MQ-7 Sensor Reading (ppm)	Reference Sensor Reading CO (ppm)
Uji 1	50	48
Uji 2	100	97
Uji 3	150	145
Uji 4	200	196
Uji 5	250	240
Uji 6	300	290
Uji 7	350	336
Uji 8	400	388
Uji 9	450	436
Uji 10	500	485

Table 3.3 shows the results of data readings from testing the response of carbon monoxide (CO) gas air levels using the MQ-7 sensor and the CO reference sensor at a distance of 50 meters with reading variations from 50 to 500 ppm. The readings of the MQ-7 sensor and the CO reference sensor are relatively close to each other, indicating that the

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

MQ-7 sensor provides fairly accurate results in detecting CO levels at a distance of 50 meters. Further testing at different distances and various environmental conditions can provide further understanding of the performance of the MQ-7 sensor in detecting CO levels in different scenarios.

4. DISCUSSION

In this research, the design and implementation of an air quality monitoring system using the CSMA-CA protocol in communication using LoRa (*Long Range*) technology is carried out. This research aims to develop an efficient and reliable solution in air quality monitoring, by utilizing the advantages of wide range and energy saving offered by LoRa technology. The research methodology involves the design of hardware and software to integrate air quality sensors with the LoRa network [14].

The results of this study show that the air quality monitoring system with CSMA-CA protocol on LoRa communication provides good performance in collecting real-time air quality data with sufficient accuracy. Analysis of the signal quality and efficiency of the CSMA-CA protocol in using the LoRa frequency spectrum has been evaluated and produced positive results [15]. The success of this system can make a significant contribution in supporting efficient air quality monitoring, addressing air pollution challenges, and promoting environmental protection and public health efforts. However, it should be noted that this research may also present some constraints and limitations, which can serve as a basis for further development to improve the performance and flexibility of the air quality monitoring system using LoRa technology and the CSMA-CA protocol [16].

The novelty of this research is to combine LoRa technology with CSMA-CA protocol for air quality monitoring. This combination provides frequency efficiency, reduces signal collisions, and enables real-time data collection. Its potential in supporting rapid action to address air pollution and public health impacts [17]. In addition, this research contributes to the development of LoRa-based monitoring technology and CSMA-CA protocol, opening up opportunities for similar applications in other monitoring fields. A comparison with previous findings shows that the approach of using CSMA-CA protocol in LoRa communication for air quality monitoring provides a more efficient, reliable, and dependable solution compared to other conventional approaches [18]. By utilizing the potential of LoRa in coverage and energy saving as well as the efficiency of CSMA-CA protocol in managing frequency access, this research brings a new innovation in air quality monitoring technology that can have a positive impact on the environment and public health [19].

5. CONCLUSION

The MAC *Layer* method-based air quality monitoring system is a solution that helps users to access air quality data in rural areas with peat characteristics without a fixed air quality monitoring station [20]. By adopting LoRa RFM95 communication and applying the MAC *Layer* method, this monitoring system is able to send data from sender to receiver within a range of more than 1000 meters without obstacles. The implementation of *Carrier Sense Multiple Access* (CSMA-CA) in this system uses NodeMCU ESP8266 as a microcontroller at the receiver by utilizing the Arduino IDE platform as a program and the Firebase website as a storage medium for recording measurement data to the database. Thus, this system is an efficient solution for monitoring air quality in rural areas that were previously difficult to access and provides important data for decision making and environmental improvement efforts.

REFERENCES

[1] M. Abbasi, A. Shahraki, M. Jalil Piran, and A. Taherkordi, "Deep Reinforcement Learning for QoS provisioning at the MAC layer: A Survey," *Eng. Appl. Artif. Intell.*, vol. 102, no. January, p. 104234, 2021, doi: 10.1016/j.engappai.2021.104234.

[2] S. Herrería-Alonso, A. Suárez-González, M. Rodríguez-Pérez, and C. López-García, "Enhancing LoRaWAN scalability with Longest First Slotted CSMA," *Comput. Networks*, vol. 216, no. June, p. 109252, 2022, doi: 10.1016/j.comnet.2022.109252.

[3] S. G. Spicer, C. Fullwood, J. Close, L. L. Nicklin, J. Lloyd, and H. Lloyd, "Loot boxes and problem gambling: Investigating the 'gateway hypothesis,'" *Addict. Behav.*, vol. 131, no. December 2021, p. 107327, 2022, doi: 10.1016/j.addbeh.2022.107327.

[4] H. A. A. Al-Kashoash and A. H. Kemp, "Comparison of 6LoWPAN and LPWAN for the Internet of Things," *Aust. J. Electr. Electron. Eng.*, vol. 13, no. 4, pp. 268–274, 2016, doi: 10.1080/1448837X.2017.1409920.

[5] O. T. Sanchez *et al.*, "Green Bear - A LoRaWAN-based Human-in-the-Loop case-study for sustainable cities," *Pervasive Mob. Comput.*, vol. 87, p. 101701, 2022, doi: 10.1016/j.pmcj.2022.101701.

[6] M. Gamal, N. Sadek, M. R. M. Rizk, and M. A. E. A. Ahmed, "Optimization and modeling of modified unslotted CSMA/CA for wireless sensor networks," *Alexandria Eng. J.*, vol. 59, no. 2, pp. 681–691, 2020, doi: 10.1016/j.aej.2020.01.035.

[7] H. Jamali-Rad *et al.*, "IoT-based wireless seismic quality control," *Lead. Edge*, vol. 37, no. 3, pp. 214–221, 2018, doi: 10.1190/tle37030214.1.

[8] H. Shan, H. T. Cheng, S. Member, and W. Zhuang, "Cross-Layer Cooperative MAC Protocol in," pp. 1–13.

[9] C. Garrido-Hidalgo, L. Roda-Sanchez, F. J. Ramírez, A. Fernández-Caballero, and T. Olivares, "Efficient online resource allocation in large-scale LoRaWAN networks: A multi-agent approach," *Comput. Networks*, vol. 221, no. November 2022, p. 109525, 2023, doi: 10.1016/j.comnet.2022.109525.

[10] E. Didik Widianto, A. A. Faizal, D. Eridani, R. Dwi, O. Augustinus, and M. S. Pakpahan, "Simple LoRa Protocol: Protokol Komunikasi LoRa Untuk Sistem Pemantauan Multisensor Simple LoRa Protocol: LoRa Communication Protocol for Multisensor Monitoring Systems," *Telka*, vol. 5, no. 2, pp. 83–92, 2019.

[11] G. L. Patzer, "Understanding the causal relationship between physical attractiveness and self-esteem," *J. Esthet. Restor. Dent.*, vol. 8, no. 3, pp. 144–147, 1996, doi: 10.1111/j.1708-8240.1996.tb01008.x.

[12] ETSI, "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General aspects of Quality of Service (QoS)," *Etsi Tr 101 329 V2.1.1*, vol. 1, pp. 1–37, 1999.

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

[13] W. Rong-Hou, L. Yang-Han, T. Hsien-Wei, J. Yih-Guang, and C. Ming-Hsueh, "Study of characteristics of RSSI signal," *Proc. IEEE Int. Conf. Ind. Technol.*, pp. 3–5, 2008, doi: 10.1109/ICIT.2008.4608603.

[14] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: Throughput and delay analysis," *Comput. Commun.*, vol. 25, no. 3, pp. 313–321, 2002, doi: 10.1016/S0140-3664(01)00369-3.

[15] N. Kiyavash, F. Koushanfar, T. P. Coleman, and M. Rodrigues, "A timing channel spyware for the CSMA/CA protocol," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 3, pp. 477–487, 2013, doi: 10.1109/TIFS.2013.2238930.

[16] P. S. Republic, "Technical Report 技术报告 Technical Report 技术报告," no. 151, pp. 704062220701–03, 2022.

[17] R. Doost-Mohammady, M. Y. Naderi, and K. R. Chowdhury, "Performance analysis of CSMA/CA based medium access in full duplex wireless communications," *IEEE Trans. Mob. Comput.*, vol. 15, no. 6, pp. 1457–1470, 2016, doi: 10.1109/TMC.2015.2462832.

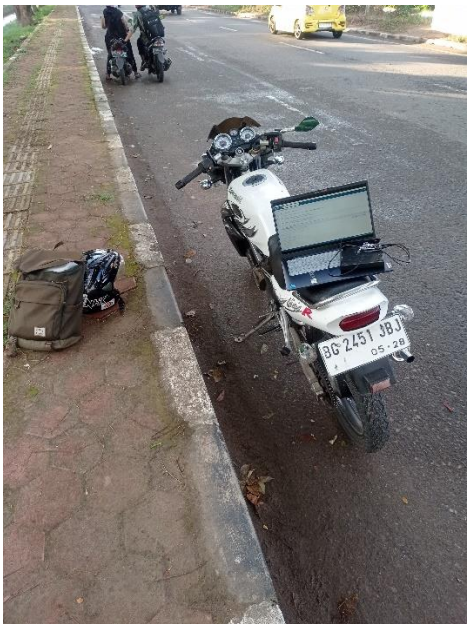
[18] N. Q. Dinh and D. S. Kim, "Performance evaluation of priority CSMA-CA mechanism on ISA100.11a wireless network," *Comput. Stand. Interfaces*, vol. 34, no. 1, pp. 117–123, 2012, doi: 10.1016/j.csi.2011.06.001.

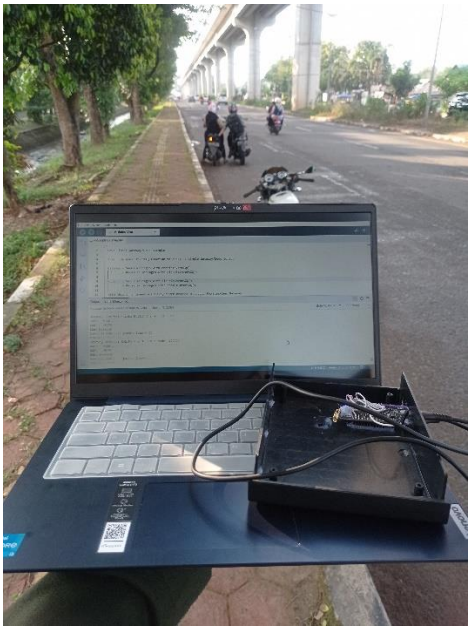
[19] C. Pham, "Investigating and experimenting CSMA channel access mechanisms for LoRa IoT networks," *IEEE Wirel. Commun. Netw. Conf. WCNC*, vol. 2018-April, pp. 1–6, 2018, doi: 10.1109/WCNC.2018.8376997.

[20] S. D. Purnamasari, E. Education, and S. Program, "Design of Gas Detector and Fire Detector Based Internet of Things Using Arduino Uno," no. 472, pp. 8–14, 2021.

LAMPIRAN

Dokumentasi





File Konfigurasi dan Hasil Konfigurasi (Program Arduino IDE)

LoRa Sender

```
#include <SPI.h>
#include <LoRa.h>

int counter = 0;

void setup() {
  Serial.begin(9600);
  while (!Serial);

  Serial.println("LoRa Sender");

  if (!LoRa.begin(915E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
}

void loop() {
  Serial.print("Sending packet: ");
  Serial.println(counter);

  // send packet
  LoRa.beginPacket();
  LoRa.print("hello ");
  LoRa.print(counter);
  LoRa.endPacket();

  counter++;

  delay(5000);
}
```

LoRa Receiver

```
#include <SPI.h>
#include <LoRa.h>

void setup() {
  Serial.begin(9600);
  while (!Serial);

  Serial.println("LoRa Receiver");
```

```
  if (!LoRa.begin(915E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
}

void loop() {
  // try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    // received a packet
    Serial.print("Received packet '");

    // read packet
    while (LoRa.available()) {
      Serial.print((char)LoRa.read());
    }

    // print RSSI of packet
    Serial.print("' with RSSI ");
    Serial.println(LoRa.packetRssi());
  }
}
```

Sensor MQ-7

```
#include "MQ7.h"

#define A_PIN 3
#define VOLTAGE 5

// init MQ7 device
MQ7 mq7(A_PIN, VOLTAGE);

void setup() {
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial connection
  }

  Serial.println("");   // blank new line

  Serial.println("Calibrating MQ7");
  mq7.calibrate();    // calculates R0
  Serial.println("Calibration done!");
    Serial.begin(9600);
  while (!Serial);
```

```
  Serial.println("LoRa Sender");
  if (!LoRa.begin(915E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
}
void loop() {

  read_sensor();
  Serial.print("Sending data sensor: ");
    Serial.println(mq7.readPpm());

  // send packet
  LoRa.beginPacket();
  LoRa.print("data sensor A ");
  LoRa.print(mq7.readPpm());
  LoRa.endPacket();


  delay(1000);
}


// the loop routine runs over and over again forever:
void read_sensor() {
    Serial.print("PPM = "); Serial.println(mq7.readPpm());

  Serial.println("");   // blank new line
  delay(1000);
  }
```

Node A

```
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

#define TX_INTERVAL 2000

// Pin mapping
const lmic_pinmap lmic_pins = {
    .nss = 10,//6,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 9,
    .dio = {2, 3, 4},
};

// left empty here (we cannot leave them out completely unless
```

```
// DISABLE_JOIN is set in arduino-
lmoc/project_config/lmic_project_config.h,
// otherwise the linker will complain).void os_getArtEui (u1_t* buf)
{ }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }

void onEvent (ev_t ev) {
}

osjob_t txjob;
osjob_t timeoutjob;
static void tx_func (osjob_t* job);

// Transmit the given string and call the given function afterwards
void tx(const char *str, osjobcb_t func) {
  os_radio(RADIO_RST); // Stop RX first
  delay(1); // Wait a bit, without this os_radio below asserts,
apparently because the state hasn't changed yet
  LMIC.dataLen = 0;
  while (*str)
    LMIC.frame[LMIC.dataLen++] = *str++;
  LMIC.osjob.func = func;
  os_radio(RADIO_TX);
  Serial.println("TX");
}

// Enable rx mode and call func when a packet is received
void rx(osjobcb_t func) {
  LMIC.osjob.func = func;
  LMIC.rxtime = os_getTime(); // RX _now_


  // receiving a packet)
  os_radio(RADIO_RXON);
  Serial.println("RX");
}

static void rxtimeout_func(osjob_t *job) {
  digitalWrite(LED_BUILTIN, LOW); // off
}

static void rx_func (osjob_t* job) {
  // Blink once to confirm reception and then keep the led on
  digitalWrite(LED_BUILTIN, LOW); // off
  delay(10);
  digitalWrite(LED_BUILTIN, HIGH); // on

  // Timeout RX (i.e. update led status) after 3 periods without RX
```

```
  os_setTimedCallback(&timeoutjob, os_getTime() +
ms2osticks(3*TX_INTERVAL), rxtimeout_func);
  // Reschedule TX so that it should not collide with the other
side's
  // next TX
  os_setTimedCallback(&txjob, os_getTime() +
ms2osticks(TX_INTERVAL/2), tx_func);

  Serial.print("Got ");
  Serial.print(LMIC.dataLen);
  Serial.println(" bytes");
  Serial.write(LMIC.frame, LMIC.dataLen);
  Serial.println();

  // Restart RX
  rx(rx_func);
}

static void txdone_func (osjob_t* job) {
  rx(rx_func);
}


#include "MQ7.h"

#define A_PIN 0
#define VOLTAGE 5

// init MQ7 device
MQ7 mq7(A_PIN, VOLTAGE);

void setup() {
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial connection
  }

  Serial.println("");   // blank new line

  Serial.println("Calibrating MQ7");
  mq7.calibrate();    // calculates R0
  Serial.println("Calibration done!");

  Serial.println("Starting");
  #ifdef VCC_ENABLE
  // For Pinoccio Scout boards
  pinMode(VCC_ENABLE, OUTPUT);
  digitalWrite(VCC_ENABLE, HIGH);
  delay(1000);
  #endif
```

```
  pinMode(LED_BUILTIN, OUTPUT);

  // initialize runtime env
  os_init();

#if defined(CFG_eu868)
  // Use a frequency in the g3 which allows 10% duty cycling.
  LMIC.freq = 869525000;
  // Use a medium spread factor. This can be increased up to SF12
for
  // better range, but then, the interval should be (significantly)
  // raised to comply with duty cycle limits as well.
  LMIC.datarate = DR_SF9;
  // Maximum TX power
  LMIC.txpow = 27;
#elif defined(CFG_us915)

  // set fDownlink true to use a downlink channel; false
  // to use an uplink channel. Generally speaking, uplink
  // is more interesting, because you can prove that gateways
  // *should* be able to hear you.
  const static bool fDownlink = false;

  // the downlink channel to be used.
  const static uint8_t kDownlinkChannel = 3;

  // the uplink channel to be used.
  const static uint8_t kUplinkChannel = 8 + 3;

  // this is automatically set to the proper bandwidth in kHz,
  // based on the selected channel.
  uint32_t uBandwidth;

  if (! fDownlink)
        {
        if (kUplinkChannel < 64)
                {
                LMIC.freq = US915_125kHz_UPFBASE +
                            kUplinkChannel * US915_125kHz_UPFSTEP;
                uBandwidth = 125;
                }
        else
                {
                LMIC.freq = US915_500kHz_UPFBASE +
                            (kUplinkChannel - 64) *
US915_500kHz_UPFSTEP;
                uBandwidth = 500;
                }
        }
  else
```

```
        {
        // downlink channel
        LMIC.freq = US915_500kHz_DNFBASE +
                    kDownlinkChannel * US915_500kHz_DNFSTEP;
        uBandwidth = 500;
        }

  // Use a suitable spreading factor
  if (uBandwidth < 500)
        LMIC.datarate = US915_DR_SF7;         // DR4
  else
        LMIC.datarate = US915_DR_SF12CR;      // DR8

  // default tx power for US: 21 dBm
  LMIC.txpow = 30;
#elif defined(CFG_au915)

  // set fDownlink true to use a downlink channel; false
  // to use an uplink channel. Generally speaking, uplink
  // is more interesting, because you can prove that gateways
  // *should* be able to hear you.
  const static bool fDownlink = false;

  // the downlink channel to be used.
  const static uint8_t kDownlinkChannel = 3;

  // the uplink channel to be used.
  const static uint8_t kUplinkChannel = 8 + 3;

  // this is automatically set to the proper bandwidth in kHz,
  // based on the selected channel.
  uint32_t uBandwidth;

  if (! fDownlink)
        {
        if (kUplinkChannel < 64)
                {
                LMIC.freq = AU915_125kHz_UPFBASE +
                            kUplinkChannel * AU915_125kHz_UPFSTEP;
                uBandwidth = 125;
                }
        else
                {
                LMIC.freq = AU915_500kHz_UPFBASE +
                            (kUplinkChannel - 64) *
AU915_500kHz_UPFSTEP;
                uBandwidth = 500;
                }
        }
  else
```

```
        {
        // downlink channel
        LMIC.freq = AU915_500kHz_DNFBASE +
                    kDownlinkChannel * AU915_500kHz_DNFSTEP;
        uBandwidth = 500;
        }

  // Use a suitable spreading factor
  if (uBandwidth < 500)
        LMIC.datarate = AU915_DR_SF7;         // DR4
  else
        LMIC.datarate = AU915_DR_SF12CR;      // DR8

  // default tx power for AU: 30 dBm
  LMIC.txpow = 30;
#elif defined(CFG_as923)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = AS923_F1 + kChannel * 200000;
        uBandwidth = 125;

        // Use a suitable spreading factor
        if (uBandwidth == 125)
                LMIC.datarate = AS923_DR_SF7;         // DR7
        else
                LMIC.datarate = AS923_DR_SF7B;        // DR8

        // default tx power for AS: 21 dBm
        LMIC.txpow = 16;

        if (LMIC_COUNTRY_CODE == LMIC_COUNTRY_CODE_JP)
                {
                LMIC.lbt_ticks = us2osticks(AS923JP_LBT_US);
                LMIC.lbt_dbmax = AS923JP_LBT_DB_MAX;
                }
#elif defined(CFG_kr920)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = KR920_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = KR920_DR_SF7;         // DR7
        // default tx power for KR: 14 dBm
        LMIC.txpow = KR920_TX_EIRP_MAX_DBM;
        if (LMIC.freq < KR920_F14DBM)
          LMIC.txpow = KR920_TX_EIRP_MAX_DBM_LOW;
```

```
        LMIC.lbt_ticks = us2osticks(KR920_LBT_US);
        LMIC.lbt_dbmax = KR920_LBT_DB_MAX;
#elif defined(CFG_in866)
        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = IN866_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = IN866_DR_SF7;         // DR7
        // default tx power for IN: 30 dBm
        LMIC.txpow = IN866_TX_EIRP_MAX_DBM;
#else
# error Unsupported LMIC regional configuration.
#endif


  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  // This sets CR 4/5, BW125 (except for EU/AS923 DR_SF7B, which
uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  Serial.print("Frequency: "); Serial.print(LMIC.freq / 1000000);
            Serial.print("."); Serial.print((LMIC.freq / 100000) %
10);
            Serial.print("MHz");
  Serial.print("  LMIC.datarate: "); Serial.print(LMIC.datarate);
  Serial.print("  LMIC.txpow: "); Serial.println(LMIC.txpow);

  // This sets CR 4/5, BW125 (except for DR_SF7B, which uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  Serial.println("Started");
  Serial.flush();

  // setup initial job
  os_setCallback(&txjob, tx_func);
}

void loop() {
  // execute scheduled jobs and events
  os_runloop_once();
}
// log text to USART and toggle LED
static void tx_func (osjob_t* job) {
```

```
//  Serial.print("PPM = "); Serial.println(mq7.readPpm());


int ppm=mq7.readPpm();


char cstr[16];
itoa(ppm, cstr, 10);


char aux_str[100];
  char sensorValue[4];
  dtostrf(ppm,4,0,sensorValue);

//      ////snprintf(aux_str, sizeof(aux_str),
"AT+HTTPPARA=\"URL\",\"%s\"\r\n", url);
  snprintf(aux_str, sizeof(aux_str), "MQ-7 sensor data from Node A
(ppm)=\%s\ ppm \n", sensorValue);

Serial.println(aux_str);
//cstr char sensor_data[] = "28.1|72||OK";

  tx(aux_str, txdone_func);  //  Send the data to gateway.


// will reschedule at half this time.
  os_setTimedCallback(job, os_getTime() + ms2osticks(TX_INTERVAL +
random(500)), tx_func);
}
```

Node B

```
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

#define TX_INTERVAL 2000

// Pin mapping
const lmic_pinmap lmic_pins = {
    .nss = 10,//6,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 9,
    .dio = {2, 3, 4},
};

// left empty here (we cannot leave them out completely unless
// DISABLE_JOIN is set in arduino-
lmoc/project_config/lmic_project_config.h,
// otherwise the linker will complain).
```

```
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }

void onEvent (ev_t ev) {
}

osjob_t txjob;
osjob_t timeoutjob;
static void tx_func (osjob_t* job);

// Transmit the given string and call the given function afterwards
void tx(const char *str, osjobcb_t func) {
  os_radio(RADIO_RST); // Stop RX first
  delay(1); // Wait a bit, without this os_radio below asserts,
apparently because the state hasn't changed yet
  LMIC.dataLen = 0;
  while (*str)
    LMIC.frame[LMIC.dataLen++] = *str++;
  LMIC.osjob.func = func;
  os_radio(RADIO_TX);
  Serial.println("TX");
}

// Enable rx mode and call func when a packet is received
void rx(osjobcb_t func) {
  LMIC.osjob.func = func;
  LMIC.rxtime = os_getTime(); // RX _now_


  // receiving a packet)
  os_radio(RADIO_RXON);
  Serial.println("RX");
}

static void rxtimeout_func(osjob_t *job) {
  digitalWrite(LED_BUILTIN, LOW); // off
}

static void rx_func (osjob_t* job) {
  // Blink once to confirm reception and then keep the led on
  digitalWrite(LED_BUILTIN, LOW); // off
  delay(10);
  digitalWrite(LED_BUILTIN, HIGH); // on

  // Timeout RX (i.e. update led status) after 3 periods without RX
  os_setTimedCallback(&timeoutjob, os_getTime() +
ms2osticks(3*TX_INTERVAL), rxtimeout_func);
```

```
  // Reschedule TX so that it should not collide with the other
side's
  // next TX
  os_setTimedCallback(&txjob, os_getTime() +
ms2osticks(TX_INTERVAL/2), tx_func);

  Serial.print("Got ");
  Serial.print(LMIC.dataLen);
  Serial.println(" bytes");
  Serial.write(LMIC.frame, LMIC.dataLen);
  Serial.println();

  // Restart RX
  rx(rx_func);
}

static void txdone_func (osjob_t* job) {
  rx(rx_func);
}


#include "MQ7.h"

#define A_PIN 0
#define VOLTAGE 5

// init MQ7 device
MQ7 mq7(A_PIN, VOLTAGE);

void setup() {
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial connection
  }

  Serial.println("");   // blank new line

  Serial.println("Calibrating MQ7");
  mq7.calibrate();    // calculates R0
  Serial.println("Calibration done!");

  Serial.println("Starting");
  #ifdef VCC_ENABLE
  // For Pinoccio Scout boards
  pinMode(VCC_ENABLE, OUTPUT);
  digitalWrite(VCC_ENABLE, HIGH);
  delay(1000);
  #endif
```

```
  pinMode(LED_BUILTIN, OUTPUT);

  // initialize runtime env
  os_init();

#if defined(CFG_eu868)
  // Use a frequency in the g3 which allows 10% duty cycling.
  LMIC.freq = 869525000;
  // Use a medium spread factor. This can be increased up to SF12
for
  // better range, but then, the interval should be (significantly)
  // raised to comply with duty cycle limits as well.
  LMIC.datarate = DR_SF9;
  // Maximum TX power
  LMIC.txpow = 27;
#elif defined(CFG_us915)

  // set fDownlink true to use a downlink channel; false
  // to use an uplink channel. Generally speaking, uplink
  // is more interesting, because you can prove that gateways
  // *should* be able to hear you.
  const static bool fDownlink = false;

  // the downlink channel to be used.
  const static uint8_t kDownlinkChannel = 3;

  // the uplink channel to be used.
  const static uint8_t kUplinkChannel = 8 + 3;

  // this is automatically set to the proper bandwidth in kHz,
  // based on the selected channel.
  uint32_t uBandwidth;

  if (! fDownlink)
        {
        if (kUplinkChannel < 64)
                {
                LMIC.freq = US915_125kHz_UPFBASE +
                            kUplinkChannel * US915_125kHz_UPFSTEP;
                uBandwidth = 125;
                }
        else
                {
                LMIC.freq = US915_500kHz_UPFBASE +
                            (kUplinkChannel - 64) *
US915_500kHz_UPFSTEP;
                uBandwidth = 500;
                }
        }
  else
```

```
        {
        // downlink channel
        LMIC.freq = US915_500kHz_DNFBASE +
                    kDownlinkChannel * US915_500kHz_DNFSTEP;
        uBandwidth = 500;
        }

  // Use a suitable spreading factor
  if (uBandwidth < 500)
        LMIC.datarate = US915_DR_SF7;         // DR4
  else
        LMIC.datarate = US915_DR_SF12CR;      // DR8

  // default tx power for US: 21 dBm
  LMIC.txpow = 30;
#elif defined(CFG_au915)

  // set fDownlink true to use a downlink channel; false
  // to use an uplink channel. Generally speaking, uplink
  // is more interesting, because you can prove that gateways
  // *should* be able to hear you.
  const static bool fDownlink = false;

  // the downlink channel to be used.
  const static uint8_t kDownlinkChannel = 3;

  // the uplink channel to be used.
  const static uint8_t kUplinkChannel = 8 + 3;

  // this is automatically set to the proper bandwidth in kHz,
  // based on the selected channel.
  uint32_t uBandwidth;

  if (! fDownlink)
        {
        if (kUplinkChannel < 64)
                {
                LMIC.freq = AU915_125kHz_UPFBASE +
                            kUplinkChannel * AU915_125kHz_UPFSTEP;
                uBandwidth = 125;
                }
        else
                {
                LMIC.freq = AU915_500kHz_UPFBASE +
                            (kUplinkChannel - 64) *
AU915_500kHz_UPFSTEP;
                uBandwidth = 500;
                }
        }
  else
```

```
        {
        // downlink channel
        LMIC.freq = AU915_500kHz_DNFBASE +
                    kDownlinkChannel * AU915_500kHz_DNFSTEP;
        uBandwidth = 500;
        }

  // Use a suitable spreading factor
  if (uBandwidth < 500)
        LMIC.datarate = AU915_DR_SF7;         // DR4
  else
        LMIC.datarate = AU915_DR_SF12CR;      // DR8

  // default tx power for AU: 30 dBm
  LMIC.txpow = 30;
#elif defined(CFG_as923)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = AS923_F1 + kChannel * 200000;
        uBandwidth = 125;

        // Use a suitable spreading factor
        if (uBandwidth == 125)
                LMIC.datarate = AS923_DR_SF7;         // DR7
        else
                LMIC.datarate = AS923_DR_SF7B;        // DR8

        // default tx power for AS: 21 dBm
        LMIC.txpow = 16;

        if (LMIC_COUNTRY_CODE == LMIC_COUNTRY_CODE_JP)
                {
                LMIC.lbt_ticks = us2osticks(AS923JP_LBT_US);
                LMIC.lbt_dbmax = AS923JP_LBT_DB_MAX;
                }
#elif defined(CFG_kr920)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = KR920_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = KR920_DR_SF7;         // DR7
        // default tx power for KR: 14 dBm
        LMIC.txpow = KR920_TX_EIRP_MAX_DBM;
        if (LMIC.freq < KR920_F14DBM)
          LMIC.txpow = KR920_TX_EIRP_MAX_DBM_LOW;
```

```
        LMIC.lbt_ticks = us2osticks(KR920_LBT_US);
        LMIC.lbt_dbmax = KR920_LBT_DB_MAX;
#elif defined(CFG_in866)
        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = IN866_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = IN866_DR_SF7;         // DR7
        // default tx power for IN: 30 dBm
        LMIC.txpow = IN866_TX_EIRP_MAX_DBM;
#else
# error Unsupported LMIC regional configuration.
#endif


  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  // This sets CR 4/5, BW125 (except for EU/AS923 DR_SF7B, which
uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  Serial.print("Frequency: "); Serial.print(LMIC.freq / 1000000);
           Serial.print("."); Serial.print((LMIC.freq / 100000) %
10);
           Serial.print("MHz");
  Serial.print("  LMIC.datarate: "); Serial.print(LMIC.datarate);
  Serial.print("  LMIC.txpow: "); Serial.println(LMIC.txpow);

  // This sets CR 4/5, BW125 (except for DR_SF7B, which uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  Serial.println("Started");
  Serial.flush();

  // setup initial job
  os_setCallback(&txjob, tx_func);
}

void loop() {
  // execute scheduled jobs and events
  os_runloop_once();
}
// log text to USART and toggle LED
static void tx_func (osjob_t* job) {
```

```
//  Serial.print("PPM = "); Serial.println(mq7.readPpm());


int ppm=mq7.readPpm();


char cstr[16];
itoa(ppm, cstr, 10);


char aux_str[100];
  char sensorValue[4];
  dtostrf(ppm,4,0,sensorValue);

//      ////snprintf(aux_str, sizeof(aux_str),
"AT+HTTPPARA=\"URL\",\"%s\"\r\n", url);
  snprintf(aux_str, sizeof(aux_str), "MQ-7 sensor data from Node B
(ppm)=\%s\ ppm \n", sensorValue);

Serial.println(aux_str);
//cstr char sensor_data[] = "28.1|72||OK";

  tx(aux_str, txdone_func);  //  Send the data to gateway.


// will reschedule at half this time.
  os_setTimedCallback(job, os_getTime() + ms2osticks(TX_INTERVAL +
random(500)), tx_func);
}
```

GATEWAY

```
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

#define TX_INTERVAL 2000

// Pin mapping
const lmic_pinmap lmic_pins = {
    .nss = D8,//6,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = D0,
    .dio = {D1, D2, 3},
};


// These callbacks are only used in over-the-air activation, so they
are
// left empty here (we cannot leave them out completely unless
```

```
// DISABLE_JOIN is set in arduino-
lmoc/project_config/lmic_project_config.h,
// otherwise the linker will complain).
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }

void onEvent (ev_t ev) {
}

osjob_t txjob;
osjob_t timeoutjob;
static void tx_func (osjob_t* job);

// Transmit the given string and call the given function afterwards
void tx(const char *str, osjobcb_t func) {
  os_radio(RADIO_RST); // Stop RX first
  delay(1); // Wait a bit, without this os_radio below asserts,
apparently because the state hasn't changed yet
  LMIC.dataLen = 0;
  while (*str)
    LMIC.frame[LMIC.dataLen++] = *str++;
  LMIC.osjob.func = func;
  os_radio(RADIO_TX);
  Serial.println("TX");
}

// Enable rx mode and call func when a packet is received
void rx(osjobcb_t func) {
  LMIC.osjob.func = func;
  LMIC.rxtime = os_getTime(); // RX _now_
  // Enable "continuous" RX (e.g. without a timeout, still stops
after
  // receiving a packet)
  os_radio(RADIO_RXON);
  Serial.println("RX");
}

static void rxtimeout_func(osjob_t *job) {
  digitalWrite(LED_BUILTIN, LOW); // off
}

static void rx_func (osjob_t* job) {
  // Blink once to confirm reception and then keep the led on
  digitalWrite(LED_BUILTIN, LOW); // off
  delay(10);
  digitalWrite(LED_BUILTIN, HIGH); // on
```

```
  // Timeout RX (i.e. update led status) after 3 periods without RX
  os_setTimedCallback(&timeoutjob, os_getTime() +
ms2osticks(3*TX_INTERVAL), rxtimeout_func);

  // Reschedule TX so that it should not collide with the other
side's
  // next TX

  //====================*********
//  os_setTimedCallback(&txjob, os_getTime() +
ms2osticks(TX_INTERVAL/2), tx_func);

  Serial.print("Got ");
  Serial.print(LMIC.dataLen);
  Serial.println(" bytes");
  Serial.write(LMIC.frame, LMIC.dataLen);
  Serial.println();

  // Restart RX
  rx(rx_func);
}

static void txdone_func (osjob_t* job) {
  rx(rx_func);
}

// log text to USART and toggle LED
static void tx_func (osjob_t* job) {
  // say hello
  tx("Hello, world!", txdone_func);
  // reschedule job every TX_INTERVAL (plus a bit of random to
prevent
  // systematic collisions), unless packets are received, then
rx_func
  // will reschedule at half this time.
  os_setTimedCallback(job, os_getTime() + ms2osticks(TX_INTERVAL +
random(500)), tx_func);
}

// application entry point
void setup() {
  Serial.begin(9600);
  Serial.println("Starting");
  #ifdef VCC_ENABLE
  // For Pinoccio Scout boards
  pinMode(VCC_ENABLE, OUTPUT);
  digitalWrite(VCC_ENABLE, HIGH);
  delay(1000);
  #endif
```

```
  pinMode(LED_BUILTIN, OUTPUT);

  // initialize runtime env
  os_init();

#if defined(CFG_eu868)
  // Use a frequency in the g3 which allows 10% duty cycling.
  LMIC.freq = 869525000;
  // Use a medium spread factor. This can be increased up to SF12
for
  // better range, but then, the interval should be (significantly)
  // raised to comply with duty cycle limits as well.
  LMIC.datarate = DR_SF9;
  // Maximum TX power
  LMIC.txpow = 27;
#elif defined(CFG_us915)

  // *should* be able to hear you.
  const static bool fDownlink = false;

  // the downlink channel to be used.
  const static uint8_t kDownlinkChannel = 3;

  // the uplink channel to be used.
  const static uint8_t kUplinkChannel = 8 + 3;

  // this is automatically set to the proper bandwidth in kHz,
  // based on the selected channel.
  uint32_t uBandwidth;

  if (! fDownlink)
        {
        if (kUplinkChannel < 64)
                {
                LMIC.freq = US915_125kHz_UPFBASE +
                            kUplinkChannel * US915_125kHz_UPFSTEP;
                uBandwidth = 125;
                }
        else
                {
                LMIC.freq = US915_500kHz_UPFBASE +
                            (kUplinkChannel - 64) *
US915_500kHz_UPFSTEP;
                uBandwidth = 500;
                }
        }
  else
        {
        // downlink channel
        LMIC.freq = US915_500kHz_DNFBASE +
```

```
                        kDownlinkChannel * US915_500kHz_DNFSTEP;
            uBandwidth = 500;
            }

    // Use a suitable spreading factor
    if (uBandwidth < 500)
            LMIC.datarate = US915_DR_SF7;         // DR4
    else
            LMIC.datarate = US915_DR_SF12CR;      // DR8

    // default tx power for US: 21 dBm
    LMIC.txpow = 21;
#elif defined(CFG_au915)

    // is more interesting, because you can prove that gateways
    // *should* be able to hear you.
    const static bool fDownlink = false;

    // the downlink channel to be used.
    const static uint8_t kDownlinkChannel = 3;

    // the uplink channel to be used.
    const static uint8_t kUplinkChannel = 8 + 3;

    // this is automatically set to the proper bandwidth in kHz,
    // based on the selected channel.
    uint32_t uBandwidth;

    if (! fDownlink)
            {
            if (kUplinkChannel < 64)
                    {
                    LMIC.freq = AU915_125kHz_UPFBASE +
                                kUplinkChannel * AU915_125kHz_UPFSTEP;
                    uBandwidth = 125;
                    }
            else
                    {
                    LMIC.freq = AU915_500kHz_UPFBASE +
                                (kUplinkChannel - 64) *
AU915_500kHz_UPFSTEP;
                    uBandwidth = 500;
                    }
            }
    else
            {
            // downlink channel
            LMIC.freq = AU915_500kHz_DNFBASE +
                        kDownlinkChannel * AU915_500kHz_DNFSTEP;
            uBandwidth = 500;
```

```
        }

  // Use a suitable spreading factor
  if (uBandwidth < 500)
        LMIC.datarate = AU915_DR_SF7;         // DR4
  else
        LMIC.datarate = AU915_DR_SF12CR;      // DR8

  // default tx power for AU: 30 dBm
  LMIC.txpow = 30;
#elif defined(CFG_as923)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = AS923_F1 + kChannel * 200000;
        uBandwidth = 125;

        // Use a suitable spreading factor
        if (uBandwidth == 125)
                LMIC.datarate = AS923_DR_SF7;         // DR7
        else
                LMIC.datarate = AS923_DR_SF7B;        // DR8

        // default tx power for AS: 21 dBm
        LMIC.txpow = 16;

        if (LMIC_COUNTRY_CODE == LMIC_COUNTRY_CODE_JP)
                {
                LMIC.lbt_ticks = us2osticks(AS923JP_LBT_US);
                LMIC.lbt_dbmax = AS923JP_LBT_DB_MAX;
                }
#elif defined(CFG_kr920)

        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = KR920_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = KR920_DR_SF7;         // DR7
        // default tx power for KR: 14 dBm
        LMIC.txpow = KR920_TX_EIRP_MAX_DBM;
        if (LMIC.freq < KR920_F14DBM)
          LMIC.txpow = KR920_TX_EIRP_MAX_DBM_LOW;

        LMIC.lbt_ticks = us2osticks(KR920_LBT_US);
        LMIC.lbt_dbmax = KR920_LBT_DB_MAX;
#elif defined(CFG_in866)
```

```
        const static uint8_t kChannel = 0;
        uint32_t uBandwidth;

        LMIC.freq = IN866_F1 + kChannel * 200000;
        uBandwidth = 125;

        LMIC.datarate = IN866_DR_SF7;          // DR7
        // default tx power for IN: 30 dBm
        LMIC.txpow = IN866_TX_EIRP_MAX_DBM;
#else
# error Unsupported LMIC regional configuration.
#endif


  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  // This sets CR 4/5, BW125 (except for EU/AS923 DR_SF7B, which
uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  Serial.print("Frequency: "); Serial.print(LMIC.freq / 1000000);
            Serial.print("."); Serial.print((LMIC.freq / 100000) %
10);
            Serial.print("MHz");
  Serial.print("  LMIC.datarate: "); Serial.print(LMIC.datarate);
  Serial.print("  LMIC.txpow: "); Serial.println(LMIC.txpow);

  // This sets CR 4/5, BW125 (except for DR_SF7B, which uses BW250)
  LMIC.rps = updr2rps(LMIC.datarate);

  // disable RX IQ inversion
  LMIC.noRXIQinversion = true;

  Serial.println("Started");
  Serial.flush();

  // setup initial job
  //==================**************
  os_setCallback(&txjob, tx_func);
}

void loop() {
  // execute scheduled jobs and events
  os_runloop_once();
}
```

CURRICULUM VITAE

NAMA LENGKAP : REDHO PRASETYA UGANDA

NPM : 0619 4035 2365

TEMPAT TANGGAL LAHIR : PALEMBANG, 18 JUNI 2001

ALAMAT : JLN. PANGERAN AYIN, KOMP. PUSPASARI BLOK E. NO 05. RT/34 RW/06, KEL. KENTEN, KEC. TALANG KELAPA, KAB. BANYUASIN, 30961.

TELEPON : 085758355554 / 085717166665

RIWAYAT PENDIDIKAN FORMAL

PENDIDIKAN	NAMA SEKOLAH	TAMAT TAHUN
SD	SD NEGERI 10 TALANG KELAPA	2013
SMP	SMP NEGERI 41 PALEMBANG	2016
SMA	SMA NEGERI 14 PALEMBANG	2019

PENGALAMAN INTERNSHIP

NO	PENGALAMAN	TAHUN
1	PT. PLN (PERSERO)	2022-2023

PENGALAMAN ORGANISASI

NO	PENGALAMAN ORGANISASI	TAHUN
1	STAFF ANGGOTA DEPARTEMEN HUMAS BADAN EKSEKUTIF MAHASISWA POLSRI	2019-2020
2	KETUA BIDANG VOKAL MUSIK DEPARTEMEN MINAT DAN BAKAT UKM SIMPONY POLSRI	2020-2021
3	PENGURUS FORUM HIMPUNAN MAHASISWA TALANG KELAPA KABUPATEN BANYUASIN (HIMATALPA)	2021-2022
4.	KEPALA DIVISI KESENIAN, SOSIAL, BUDAYA DAN OLAHRAGA IKATAN REMAJA PUSPASARI (IRAPARI)	2023 - SEKARANG

Semua data yang saya isikan dan tercantum dalam *curriculum vitae* ini adalah benar dan dapat dipertanggung jawabkan.

Palembang, Agustus 2023

(REDHO PRASETYA UGANDA)