Indonesian Journal of Computer Science
ISSN 2549-7286 (*online*)
Jln. Khatib Sulaiman Dalam No. 1, Padang, Indonesia
Website: ijcs.stmikindonesia.ac.id | E-mail: ijcs@stmikindonesia.ac.id

Building a Low-Cost Lora Gateway based on .Net Core for Fast Development and Easy Integration

Haryono
haryono@pradita.ac.id
Universitas Pradita

Article Information

Submitted : 18 Nov 2023
Reviewed: 20 Nov 2023
Accepted : 20 Dev 2023

Keywords

Lora, Gateway, C#, .Net Core, OrangePi, IoT

Abstract

In the world of IoT, it is involving many devices, each Node needs to send/receive the data from/to Datacenter. The Lora Gateway is usually used to achieve that communication. There are many ways to implement the Lora Gateway, it may use available Lora Gateway in the market. This method may be faster but need to have an extra fee, especially when many gateways are needed. Another method is by using the available GSM Network, this may lead fast but require managing Simcard at each Node. Some methods select to build the Lora Gateway by themself, but this requires some development and integration. To compensate those problem, this project is involved. The aim is to prove, that it is possible to build a Low-Cost Lora Gateway based on Time Division with the help of on .Net Core prgramming. As of today .Net Core can run in the Linux OS environment smoothly, hence building Lora Gateway based on Time Division using .Net Core is very possible to be done. By developing for all devices using .Net Core the communication data across the device is easy to achieve. It is because all projects in each device reference to the same library, in which contains the same data entities and services. This project proves that building a Low-Cost Lora Gateway based Time Division using .Net Core are relatively easy and fast. The fee to build the Lora Gateway is less compared to using available Gateway in the market or using GSM Gateway per node.

A. Introduction

There are many ways to build the Lora Gateway for IoT Communication. It can directly use available gateway hardware in the market, for example using RAK831 [1] [2], but the cost is relatively expensive. If many Node sensors are away from each other then need many gateways to be set up. The other way is by using an available IoT gateway provider, for example using GSM Network [3]. Building IoT using this type of network can be cheap if the Node sensor is not many, but, if many Nodes, it needs to place the Simcard in each Node, so maintenance of many sim cards is not easy. Besides, the Lora hardware and communication protocol should comply with the GSM provider. Another method of building the Gateway is from available open-source Arduino [4]. Another method other than the Gateway is creating a multi-hope device using Arduino [5]. Since both methods use Arduino code, as Arduino is designed for Embedded, not for the Server, so it will have a challenge in doing data exchange with the Server.

To compensate above, therefore this project is done. This project is to build a low-cost Lora Gateway DIY based Time Division using the .Net Core environment. Developing DIY Gateway is consuming time, especially when handling the communication data between devices across IoT networks. This project gives proof of concept that building Gateway based Time Division using the .Net Core environment can be done smoothly with fast development and easy integration. The cost can be suppressed as much as possible. In the [1] [2] it cost arround IDR 4 million **per gateway** [6]. Where as in [3], the 4G module arround IDR 500,000 **per node.** In the next session will be calculated how many fee will be used to implement this approach.

B. Research Method

The research method consists of 4 steps. The first is to survey the available Gateway that is used in the current phase. The second is analyzing and selecting methods that can be implemented successfully and efficiently. The third is designing the architecture of the selected method. Fourth is implementing the architecture (Procure hardware, Assembly the hardware, programming, and testing).

B.1. Survey of Available Lora Gateway for IoT Communication

The term Gateway means, it can handle data from/to multiple Clients. The client can connect via the gateway and the client can send/receive data to Data Center. The gateway act as a bridge, so that the data traffic can flow. There are many types of the gateway in terms of technology and market availability. The survey that has been conducted in the online market at least cost 2.8 million rupias per Gateway. The Gateway includes the 4G connection to connect to the internet, e.g Lora Gateway S281. Figure 1 is the architecture for Lora Gateway S281 [7]. It has a 4G and can be connected to multiple Lora Clients. The project to use this Gateway is available at [8].

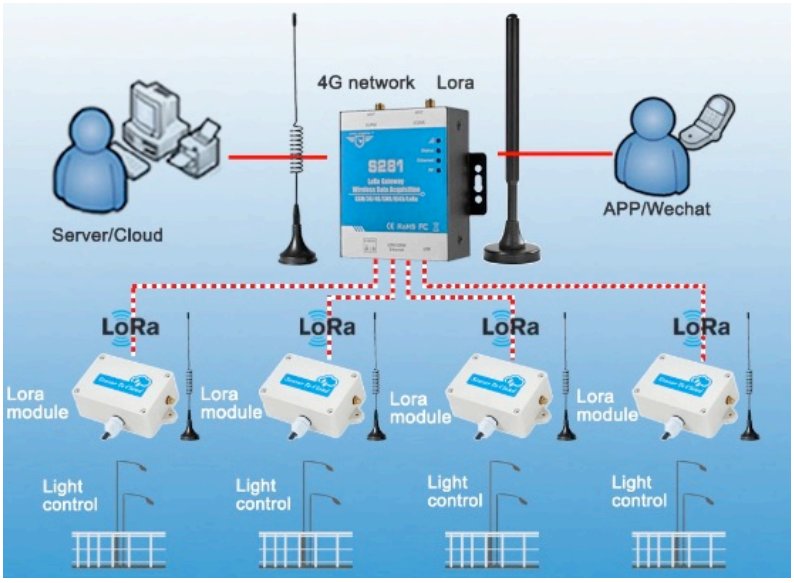

Figure 1. Architecture for Lora Gateway S281

Another product that was close to the Lora Gateway before is RAK831. The price is a bit more expensive than before. The advantages of this module are designed to work together with Raspberry Pi 3B+ [9], as shown in figure 2.

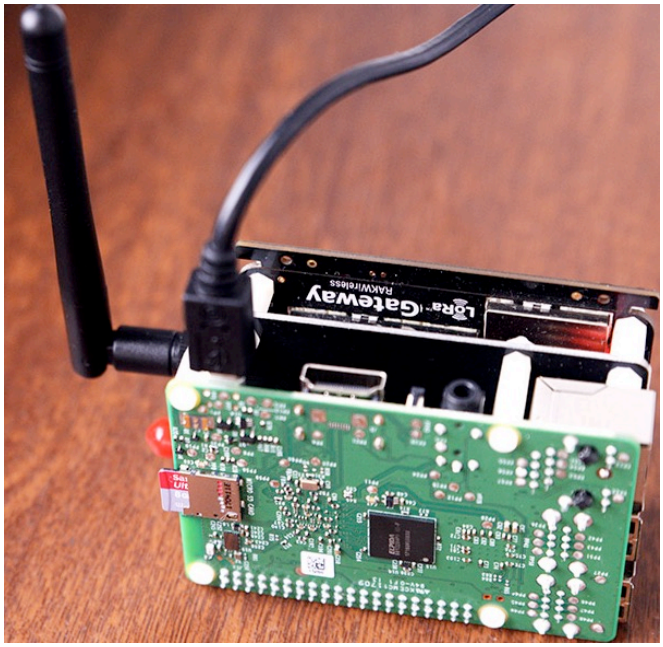

Figure 2. RAK831 with Raspberry Pi 3B+

The project which is using the RAK831 can be seen in [10] [11] [12] [13]. The RAK831 has not been equipped with an internet connection module, so the user should provide the internet connection. The data can be flown to Raspberry, then Raspberry will handle the next stage, which is sending the data to the Data Center.

RHF0M301 is another choice that can be implemented as Lora Gateway for IoT Communication [14]. Like RAK831, it does not have an Internet connection yet. The project that uses this product can be seen in [15] [16] [17]. So it still needs help from other boards to handle the connection to the internet. Usually, a user uses Raspberry, like Figure 3 [14].

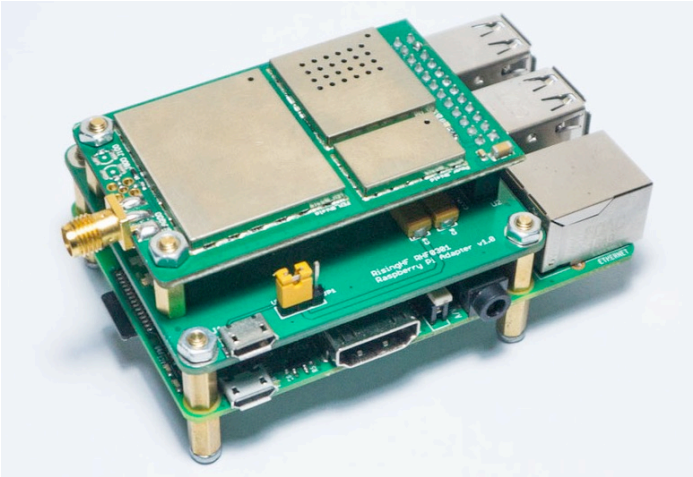

Figure 3. RHF0M301 with Raspberry Pi

Another method is by developing the gateway from scratch [18] [19] [20], this has advantages, it can decrease the fee and the protocol is flexible. Since it uses the Arduino framework integration is challenging.

B.2. Preliminary Design: Analyzing and Selecting Method

The requirement is the client can connect to the Data Center, a client can send/receive the data to/from Data Center. Client Node should be able to connect to the Gateway, the distance is at least 1 KM away. The gateway can serve the client at least 3 Nodes simultaneously.

The above requirement can be used as a scope to achieve the building of the Low-Cost Lora Gateway. Another aspect that needs to take into account is maintainability and flexibility in terms of hardware assembly and programming of the device.

To achieve the above requirement is as follows (Function and Hardware): 1. To connect all Nodes (Lora Module SX1278), 2. To handle data traffic (Orange pi), 3. To communicate with the Datacenter (Modem 4G). Those three items are the backbone or important aspect of building the Lora Gateway. The cost will be less than 1.1 Million Rupias. Based on that cost this can be implemented.

Maintainability is about how the hardware and software can be maintained efficiently. Hardware should be able to be changed easily at a low cost if the hardware is damaged, so modularity is an important aspect. Unlike available gateway in the market, some of them put all those three items in one PCB. Software should be developed in the best environment as much as possible, which is known as clean code and supports debugging environment. Code should be easily adapted

and refactored according to the sensor or command data. So to achieve this Dot Net environment is selected.

Flexible is about how the device can be replaced by other devices. For example, Lora Module can be changed by more strong signal if required more distance. It is by simply changing the Lora Module. Orange Pi can be replaced by another module that is based on Linux Environment. More, it can be changed by a custom ARM board by developing our own custom Linux board. Modem 4G can be replaced by another Modem, or even can be upgraded to use 5G modem or just use GSM network. Those such criteria fulfill the flexibility requirement.

B.3 Architecture Design

Figure 4 is the overview of the architectural design of this project. Data Center and UI are included to show how the Gateway is communicate with the data center via the internet through MQTT Broker. As shown the figure 4 Node is connected to the Gateway via transceivers of the LoRa long-range modem. To connect all nodes to the gateway, all modems should be set in the same parameter.

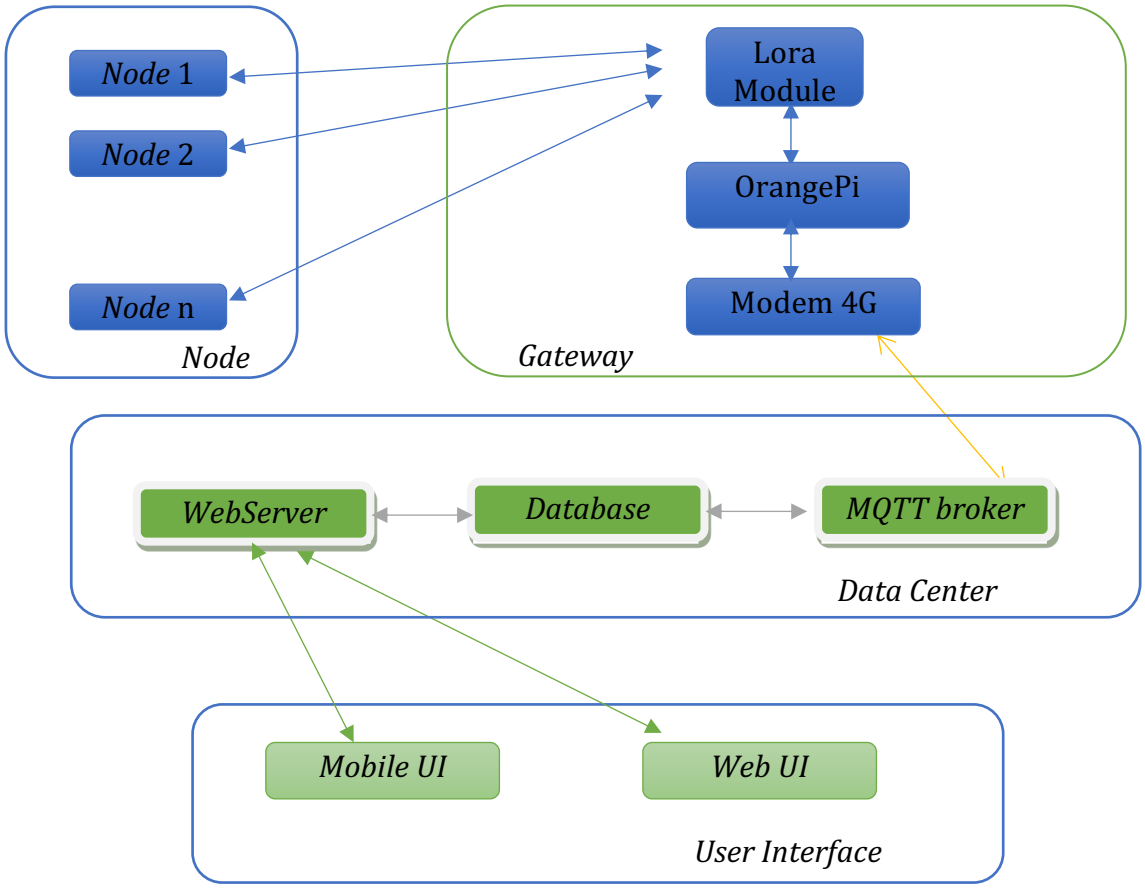

Figure 4. Big Design Overview of Lora Gateway in IoT System

The Gateway should manage the communication between them based on Time Division. To make sure the communication can be done, the Gateway is

managing the time slot to each Node. For example, the first 10 seconds are given to Node 1, then the next 10 seconds are given to Node 2, and so forth. The Gateway will know which device which is currently connected. After receiving data from Node the Gateway will forward the data to the Datacenter via Modem 4G. The Gateway is acting as MQTT Client to connect the MQTT Broker.

The hardware detail which is used to achieve the design is as follows: 1. Lora Module SX1278 in Figure 5, 2. Orange pi zero in Figure 6, 3. Modem 4G in Figure 7.

Figure 5. E32-433T30D Lora Module [20]

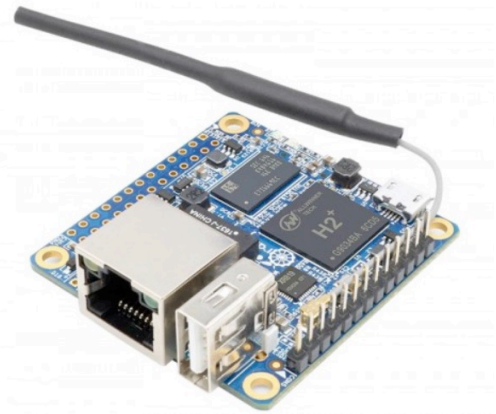

Figure 6. Orange pi Zero [21]

Figure 7. E3372 Megafone Modem

Lora Module is using SX1278 IC, frequency is about 410~441MHz, Power 21~30dBm, Distance 8.0km, UART Interface, E32-433T30D is adopted LoRa spread spectrum technology [20].

The software project is separated into 5 categories as shown in Figure 8. NodeLoraApp will be placed in each Node using OrangePi, GatewayLoraApp will be placed in the Gateway, AQSServer will be placed in DataCenter, and AQSClient will be placed in the User Interface.

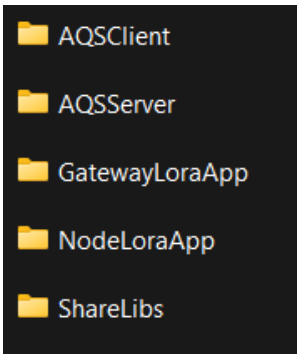

Figure 8. Software Project IoT

ShareLibs is the **Models/Entities** that is referenced by all project. To make it easy in maintain, debug and data exchange the Dot Net Core is selected. All projects are using the Dot Net Core environment so that all projects can reference the ShareLibs as shown in Figure 9. ShareLibs has the functionality to encode and decode the data. Each project can use the functionality (BinaryDataFormater) with the same code so that uniform and consistent data can be maintained easily.

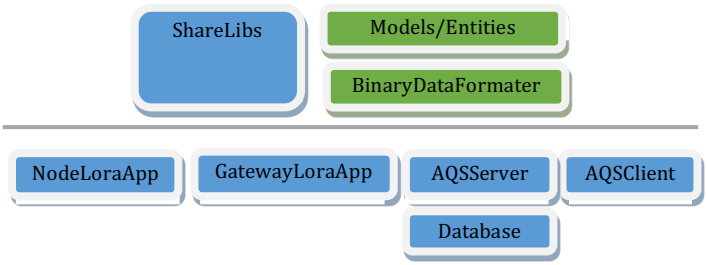

Figure 9. Software Architecture

C. Result and Discussion

In Figure 4, Gateway mainly consists of 3 components, they are Lora Module, OrangePi, and 4G Modem. Those components are connected via USB and UART. Lora Module and OrangePi are connected via Uart Port, Orange Pi, and 4G Modem is connected via USB port. To make it much more robust in terms of hardware wiring, Header Board is created, as shown in figure 10. To get the power 5 Volt from the solar panel, space for voltage step-down board from 12v to 5v is also reserved.

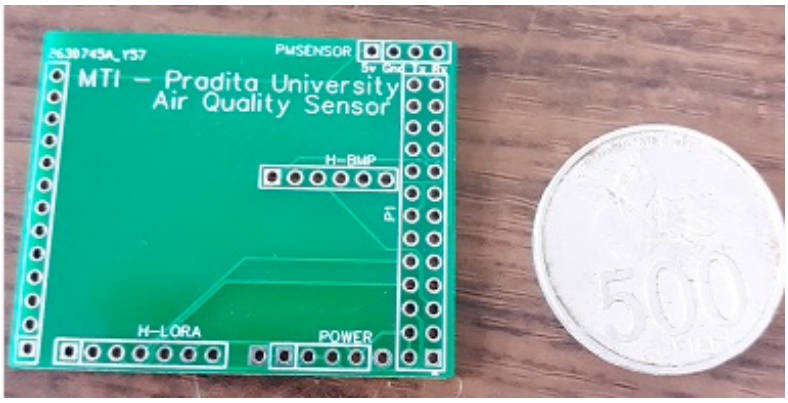

Figure 10. Header Board Gateway/Node

Figure 11 is shown Lora Gateway after assembly with the Header Board. Compare to the other gateway, it is smaller. OrangePi is placed under the Header Board, on top of the header there is Lora Module, and the 4G Modem is just plugged into the USB Port of OrangePi.

Figure 11. Lora Gateway (Lora Modem, OrangePi, 4G Modem)

Node is the main source to gather the data from various sensors or acts as an actuator to execute the command. For the testing purpose of this project, there are 5 different data type is gathered by the Node. They are CO, CO2, PM2.5,

Temperature, and Humidity. Figure 12 is the Node with the sensors which is mentioned. As Node is connected to the Gateway via Lora, the Node is also attached to the Lora Module with the same type as the Gateway has. So that communication between Node and the Gateway can be established.

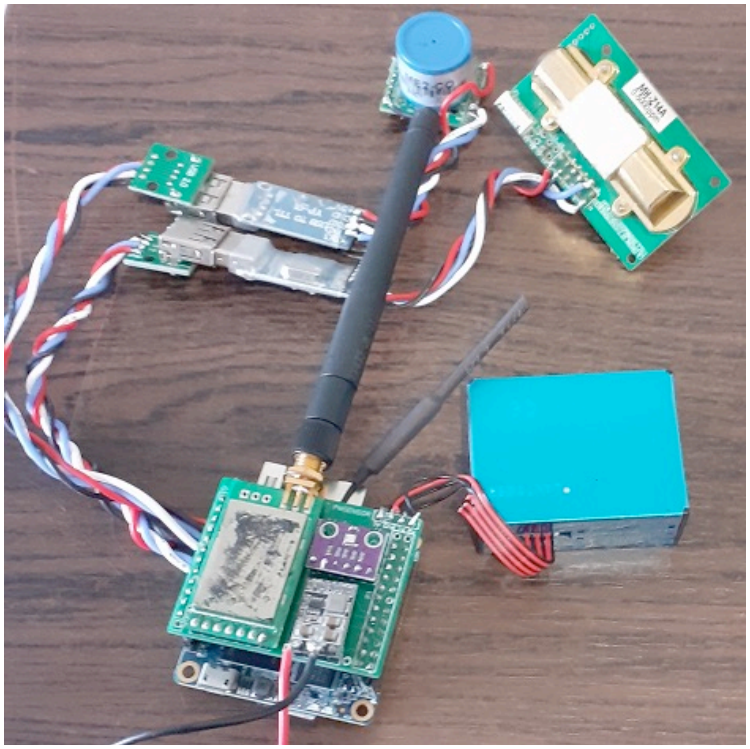

Figure 12. Node (Sensor CO, CO2, PM2.5, Temperature, Humidity)

The .Net Core is used to program all the devices. The same environment is used mainly because the project also has a focus on how to make the code **maintainable** and **reuse** the code easily. Figure 13 shows that from the bottom Linux OS as the base, followed by .Net Core and then the Application Software that is made for this project.

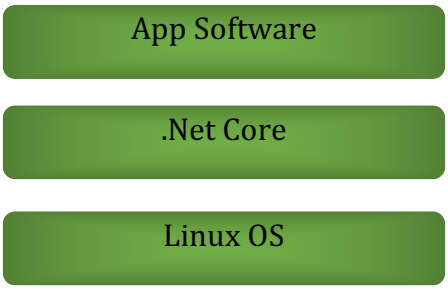

Figure 13. Software architecture

To transfer data from Node to Gateway is using binary format. A binary format is used to make sure the data is compact. Unlike ASCII format it is reserve

some more data to be transferred. Figure 14 is the method to format the data from the Model into the bitstream.

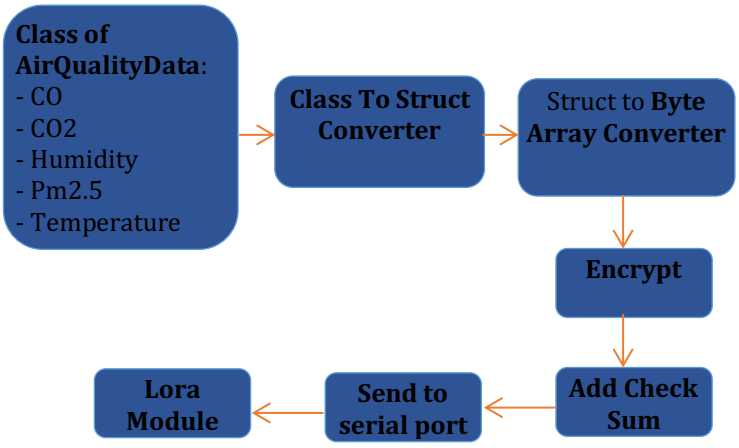

Figure 14. Formating the data in Node

Converting from object data of class is relatively easy in C#, just use **Marshal** class from structure data to convert into a byte array. To get back the actual data, it is done on the Server. The extractor class is created to extract the byte array into a class object, that can be called from all projects inside the ShareLibs library.

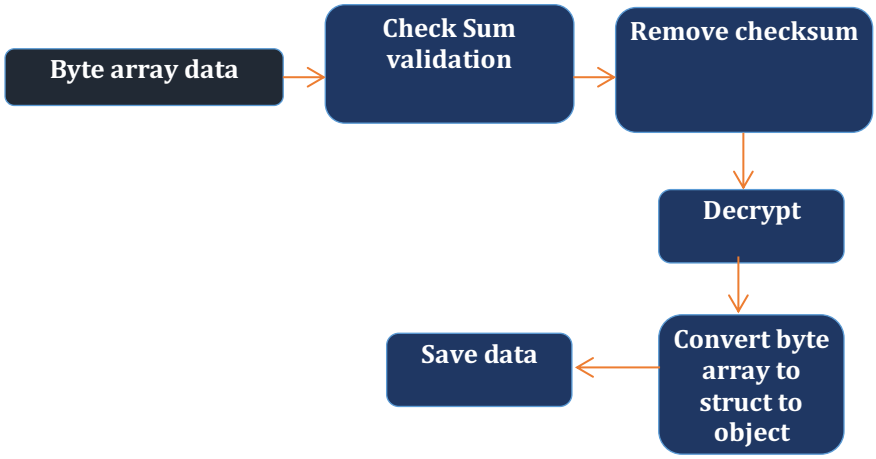

Figure 15. Extracting the byte stream to object

Testing is done by simulation and runs in real hardware. .Net core is multiplatform, code can be compiled into windows and run on it. The test is done by running the executable file in windows. Run about 3 Nodes and 1 Gateway. All get data can be received successfully. Besides, tests also are done by executing the Node Application and Gateway application in real hardware. As shown in figure 11 and figure 12 real hardware that has been installing the .Net Core Application run

on Linux OS. The test was conducted in a few days, non-stop for 24 hours, and the result can acquire the data successfully.

Far-field testing has been done. Figure 16 shows the position of Gateway and Node 1. The distance between Gateway and Node 1 is about 1 KM away, and the data can still be sent by the Node to be received by the Gateway safely. Based on the requirement this test can be concluded successfully.

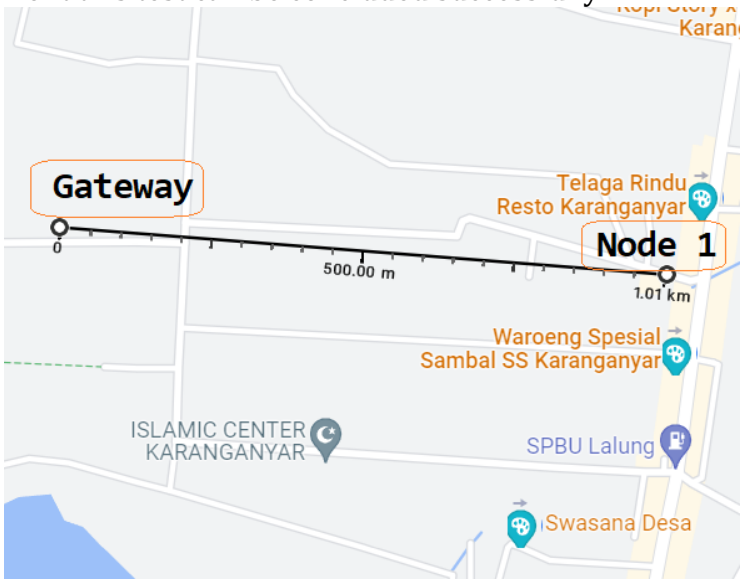

Figure 16. Far-field testing communication between Gateway and Node

The result of this project is about the software and hardware of Node, Gateway, and Server app for Datacenter. In terms of software, all programming is done using the same .Net Core environment with the C# language. As because all are using the same environment the integration and communication data between **Node, Gateway and Server** are easier to be done. This is because all Applications reference the same entity's Data in a Library Project.

The hardware consists of three main component modules, stacked modularly, which can be replaced or changed according to the need. For example, a 4G modem can be changed with a 5G modem or only a GSM modem. The Lora Module can be changed to the Higher gain Antenna, etc. Three commponentc that is used to build the gateway is E32-433T30D, Orangepi, and Modem. The fee can be calculated is as follows:

1. E32-433T30D = IDR 130,000
2. Orangepi = IDR 400,000
3. Modem 4G = IDR 140,000

Total would be arround IDR 670,000.

D. Conclusion

Building a Low-Cost Lora Gateway based on Time Division using .Net Core is successfully done. The development and integration are relatively easy and fast because all projects use the same environment. Duplication code can be avoided. All projects reference to the same data structures library so that communication data / data exchange is relatively easy to be done. The main modules contain 3 parts (LoraModule, OrangePi, and 4GModem), if the fee is sum-up, it is less than using available Gateway in the market or using GSM Gateway (in which each node need to have GSM device).

E. Acknowledgment

This project is sponsored by Pradita University.

F. References

[1] L. Parri, S. Parrino, G. Peruzzi, and A. Pozzebon, "Low Power Wide Area Networks (LPWAN) at Sea: Performance Analysis of Offshore Data Transmission by Means of LoRaWAN Connectivity for Marine Monitoring Applications," *Sensors*, vol. 19, no. 14, Art. no. 14, Jan. 2019, doi: 10.3390/s19143239.

[2] "Perencanaan Jaringan Long Range (LoRa) pada Frekuensi 920 MHz – 923 MHz di Kota Bandung - CORE." Accessed: Nov. 18, 2023. [Online]. Available: https://core.ac.uk/display/299936599

[3] A. Mohammad and Dr. K. K. Radha, "An IOT based Solar Integrated Home Security System by using GSM Module and Raspberry pi," *Int. J. Adv. Eng. Res. Sci.*, vol. 4, no. 12, pp. 195–199, 2017, doi: 10.22161/ijaers.4.12.28.

[4] D. Y. Tadeus, Yuniarto, and F. Mangkusasmito, "LoRa Gateway as Internet of Things (IoT) Infrastructure Components on Undip Vocational School," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 771, no. 1, p. 012009, Mar. 2020, doi: 10.1088/1757-899X/771/1/012009.

[5] Y. Triwidyastuti, M. Musayyanah, F. Ernawati, and C. D. Affandi, "Multi-hop Communication between LoRa End Devices," *Sci. J. Inform.*, vol. 7, no. 1, Art. no. 1, Jun. 2020, doi: 10.15294/sji.v7i1.21855.

[6] "New RAK831 LoRa/LoRaWAN Gateway Module 433/470/868/915MHz Base di Earlyta16 Shop," Tokopedia. Accessed: Nov. 21, 2023. [Online]. Available: https://www.tokopedia.com/earlyta16shop/new-rak831-lora-lorawan-gateway-module-433-470-868-915mhz-base

[7] admin, "Cellular Lora Gateway," Microthings. Accessed: Jul. 09, 2022. [Online]. Available: https://www.microthings.id/product/cellular-lora-gateway/

[8] O. Santiago, "Evaluation of criteria to encourage zero energy balance buildings in Colombia," Universidad Nacional de Colombia, 2022. [Online]. Available: https://repositorio.unal.edu.co/bitstream/handle/unal/80971/1053846978.2022.pdf?sequence=1&isAllowed=y

[9] "Designed for Developers from the IoT World: The Heavy-Duty RAK831 from RAKwireless - RAKwireless - IoT Made Easy." Accessed: Jul. 09, 2022. [Online]. Available: https://www.rakwireless.com/en-us/products/lpwan-gateways-and-concentrators/rak831

[10] C. I. Ndukwe, T. Iqbal, X. Liang, J. Khan, and L. Aghenta, "LoRa-based communication system for data transfer in microgrids," *AIMS Electron. Electr. Eng.*, vol. 4, pp. 303–325, Sep. 2020, doi: 10.3934/ElectrEng.2020.3.303.
[11] H. Soy and Y. Dilay, "A Conceptual Design of LoRa based Weather Monitoring System for Smart Farming," *Eur. J. Sci. Technol.*, Oct. 2021, doi: 10.31590/ejosat.1011947.
[12] C. Ndukwe, T. Iqbal, and J. Khan, "An Open Source LoRa Based, Low-Cost IoT Platform for Renewable Energy Generation Unit Monitoring and Supervisory Control," *J. Energy Power Technol.*, vol. 4, no. 1, Art. no. 1, Feb. 2022, doi: 10.21926/jept.2201007.
[13] "(PDF) Development of a Low-cost LoRa based SCADA system for Monitoring and Supervisory Control of Small Renewable Energy Generation Systems." Accessed: Aug. 09, 2022. [Online]. Available: https://www.researchgate.net/publication/347868716_Development_of_a_Low-cost_LoRa_based_SCADA_system_for_Monitoring_and_Supervisory_Control_of_Small_Renewable_Energy_Generation_Systems
[14] "RHF0M301 LoRaWAN Module - Marketplace - The Things Network." Accessed: Jul. 09, 2022. [Online]. Available: https://thethingsnetwork.org/marketplace/product/rhf0m301-lorawan-module
[15] P. Fraga-Lamas *et al.*, "Design and Experimental Validation of a LoRaWAN Fog Computing Based Architecture for IoT Enabled Smart Campus Applications," *Sensors*, vol. 19, no. 15, p. 3287, Jul. 2019, doi: 10.3390/s19153287.
[16] N. Raad, T. Hasan, and A. Chalak Shakir, "ECC Based Data Retrieval Using LoRaWAN Technology," *Int. J. Eng. Technol.*, vol. 7, p. 4918, Nov. 2018, doi: 10.14419/ijet.v7i4.19850.
[17] S. Barillaro, D. Anand, A. Gopstein, and J. Barillaro, "A Demonstration of Low Power Wide Area Networking for City-Scale Monitoring Applications," 2019, pp. 608–618. doi: 10.1007/978-3-030-31831-4_44.
[18] E. D. Widianto, A. A. Faizal, D. Eridani, R. D. O. Augustinus, and M. S. Pakpahan, "Simple LoRa Protocol: Protokol Komunikasi LoRa Untuk Sistem Pemantauan Multisensor," *TELKA - J. Telekomun. Elektron. Komputasi Dan Kontrol*, vol. 5, no. 2, Art. no. 2, Nov. 2019, doi: 10.15575/telka.v5n2.83-92.
[19] M. P. S. Simbolon, H. Wijanarko, F. Nakul, and R. Mahdaliza, "Penerapan Komunikasi Nirkabel LoRa pada Sistem Pencatat Kehadiran Portabel," *J. Appl. Electr. Eng.*, vol. 5, no. 2, Art. no. 2, Dec. 2021, doi: 10.30871/jaee.v5i2.3096.
[20] ebyte, "E32-433T30." Accessed: Jul. 09, 2022. [Online]. Available: https://www.ebyte.com/en/product-view-news.html?id=108
[21] "Orange Pi Zero LTS." Accessed: Jul. 09, 2022. [Online]. Available: https://orangepi.com/index.php?route=product/product&product_id=846