



VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Linh Nguyen

CARPUTER WITH LORA

Car Location Monitoring System Based on LoRa Network

Technology and Communication

2021

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

ABSTRACT

Author	Linh Nguyen
Title	Carputer with LoRa
Year	2021
Language	English
Pages	82
Name of Supervisor	Jukka Matila

Over the past few years, the Internet of Things (IoT) has become one of the most promising topics in the technical, social and economic area and its market has been growing rapidly since then. At Vaasa University of Applied Sciences, there are many IoT-related projects which are being conducted for school or for enterprises collaborations.

The primary objective of this project was researching the solid background and creating an understandable concept of LoRa and LoRaWAN for anyone who would want to get started with LoRa modulation technology and explore how they enable the potential of IoT worldwide with a very large-scale network. The secondary aim was to demonstrate how LoRa is applied to our real-life application. A GPS tracking system will be installed in a car for monitoring the vehicle's location on a website application.

Throughout the theory parts of this thesis, LoRa and LoRaWAN were exploring from the roots. The basis, as well as the most important key parameters, were discussed with examples and different comparisons with other technologies. The final GPS system was an example of LoRa devices and a server network forming a LoRa Wide Area Network. It was successful to demonstrate how data was sent

from the GPS node to the gateway and finally displayed on a visualized web application while consuming low power.

Keywords LoRa, LoRaWAN, IoT, low-power, modulation technology

CONTENTS

ABSTRACT

1	INTRODUCTION.....	11
2	THEORETICAL BACKGROUND	13
	2.1 Wireless Communication Technology	13
	2.2 The Internet of Things	14
	2.2.1 The IoT Explained	14
	2.2.2 The IoT's Characteristics	15
	2.2.3 Challenges for the IoT	17
	2.3 Low-Power Wide Area Networks (LPWAN)	17
3	LORA	19
	3.1 Overview of LoRa.....	19
	3.2 Features of LoRa Technology.....	20
	3.2.1 Wide Area versus Power.....	21
	3.2.2 Network.....	24
	3.3 Operative Rules and Regulations.....	24
	3.3.1 Radio Bands	24
	3.3.2 Duty Cycle – Time on Air.....	25
	3.4 Fresnel Zone.....	26
	3.5 Radio Frequency Modulation	29
	3.6 Link Budget	30
	3.7 EIRP and ERP.....	31
	3.7.1 EIRP	31
	3.7.2 ERP	32
	3.8 RSSI and SNR.....	33
	3.8.1 RSSI	33
	3.8.2 SNR.....	33
4	LORAWAN.....	35
	4.1 Overview of LoRaWAN	35
	4.1.1 Main Components	35
	4.1.2 Frequency Bands and Regulations.....	36

4.1.3	Data Rates	36
4.2	Spreading Factor	37
4.3	Advantages and Disadvantages.....	38
4.3.1	Advantages	38
4.3.2	Disadvantages	39
4.4	Security	39
4.5	Device Classes	40
4.5.1	Class A Device	41
4.5.2	Class B Device	43
4.5.3	Class C Device	44
4.6	Annexes.....	45
4.6.1	DeciBel.....	45
4.6.2	dBi and dBd	46
5	GPS TRACKER	48
5.1	The Things Network – TTN	49
5.2	Dragino Gateway	50
5.2.1	Overview	50
5.2.2	Configuration of the Dragino Gateway	50
5.2.3	Creating an Application on TTN	56
5.3	TTGO T-Beam ESP32.....	56
5.3.1	Overview	56
5.3.2	Arduino IDE and TTGO T-Beam Setup and Installation of the Arduino IDE.....	58
5.3.3	Programming the GPS Tracker	65
5.4	Testing Result	74
6	CONCLUSIONS	80
	REFERENCES.....	81

LIST OF FIGURES AND TABLES

Figure 1. How the IoT works /2/	15
Figure 2. Networking connections /5/	19
Figure 3. OSI Model /5/	20
Figure 4. Example of LoRa end node /7/	21
Figure 5. Example of LoRa gateway /7/	21
Figure 6. Bandwidth and range graph for networks /8/.....	22
Figure 7. LoRa networks /5/.....	24
Figure 8. ISM frequencies band /10/.....	25
Figure 9. Fresnel zone /6/.....	27
Figure 10. Example of interference in Fresnel zone /6/	27
Figure 11. Example of obstacles within the Fresnel zone /6/.....	27
Figure 12. Curvature consideration /6/.....	28
Figure 13. Fresnel zone with 40% blockage based on flat Earth /6/.....	29
Figure 14. Fresnel zone with 40% blockage considering curvature of the Earth /6/	29
Figure 15. Signal modulation /6/.....	29
Figure 16. RF modulation /12/	30
Figure 17. Link budget and link margin /6/	30
Figure 18. EIRP demonstration /6/.....	31
Figure 19. ERP demonstration /6/	32
Figure 20. RSSI /6/.....	33
Figure 21. RSSI and SNR relation /6/	33
Figure 22. SNR of LoRa /6/	34
Figure 23. Security Session Keys are shared within LoRaWAN /5/	40
Figure 24. Data packets securely transmitted from end to end /5/	40
Figure 25. LoRa protocol stack /5/.....	40
Figure 26. How class A devices operate /5/	41
Figure 27. End node after data transmitting /5/.....	42
Figure 28. Data is received in window 1 (Rx1) /5/	42
Figure 29. Data is received in window 2 (Rx2) /5/	42
Figure 30. How class B(Beacon) devices operate /5/.....	43

Figure 31. Beacon intervals /5/	43
Figure 32. Beacon ping periods /5/	44
Figure 33. How class C devices operate /5/	44
Figure 34. Isotropic antenna radiation pattern /15/	46
Figure 35. Half-wave dipole antenna /15/	46
Figure 36. Dipole antenna radiation pattern /15/.....	47
Figure 37. dBi and dBd relationship /15/	47
Figure 38. TTN server in LoRaWAN /16/	49
Figure 39. TTN Console dashboard	50
Figure 40. LG01N Dragino gateway /17/.....	50
Figure 41. Dragino wiring	51
Figure 42. WiFi network scanning.....	51
Figure 43. Join WiFi network 1	52
Figure 44. Joining WiFi network 2	52
Figure 45. Save & Apply joining network	52
Figure 46. Disable the default Dragino AP	53
Figure 47. Finding the Gateway EUI	53
Figure 48. Register gateway on TTN 1	54
Figure 49. Register gateway on TTN 2	54
Figure 50. Gateway Overview	55
Figure 51. Finish registering gateway 1	55
Figure 52. Finish registering gateway 2	56
Figure 53. Register application on TTN.....	56
Figure 54. T-Beam pin map /18/	57
Figure 55. Arduino IDE working platform	59
Figure 56. TTGO and sensor wiring	60
Figure 57. Selecting the board	60
Figure 58. Set the frequency plan.....	60
Figure 59. Choose the serial port	61
Figure 60. Arduino → Preferences → Additional Boards Manager URLs	63
Figure 61. Tools → Boards → Boards Manager	63
Figure 62. Register device in TTN.....	64

Figure 63. Change to ABP method	64
Figure 64. Securities keys	65
Figure 65. Result of successfully uploading code to the board.....	74
Figure 66. Serial monitor result	75
Figure 67. Gateway traffic in TTN.....	76
Figure 68. Application Data traffic in TTN	76
Figure 69. Detailed data traffic in TTN 1.....	76
Figure 70. Detailed data traffic in TTN 2.....	77
Figure 71. Cayenne visualized tracking web app.....	77
Figure 72. Cayenne's data recording	78
Figure 73. Data integrity secured through gateway	78
Table 1. Wireless communications comparison /6/.....	23
Table 2. ToA and duty cycle examples	26
Table 3. Data rate comparison.....	37
Table 4. Spreading factor comparison.....	38
Table 5. Arduino libraries	62

LIST OF ABBREVIATIONS

LoRa	Long Range data links
LoRaWAN	Long Range Wide Area Network
IoT	Internet of Things
RF	Radio Frequency
NB-IoT	Narrowband Internet of Things
LPWANs	Low-Power Wide Area Networks
GPS	Global Positioning System
LTE	Long-Term Evolution for wireless broadband communications
MAC	Media Access Control
AES	Advanced Encryption Standard
M2M	Machine-to-Machine connectivity
PLC	Power Line Communication
EM	Electromagnetism
IP	Internet Protocol address
URI	Uniform Resource Identifier, domain: www. (World Wide Web)
OSI	Open Systems Interconnection model is a standard framework used to describe the communications functions of a network system

PHY	Physical layer
ASK	Amplitude Shift Keying modulation
FSK	Frequency Shift Keying modulation
PSK	Phase Shift Keying modulation
CSS	Chirp Spread Spectrum
ISM	Industrial, Scientific and Medical
SF	Spreading Factor

1 INTRODUCTION

Since the first wireless signal and the radio were invented at the end of the 19th century, the structure of human civilization has been changed significantly. These inventions transformed all the traditional wired infrastructure to wireless communication and led to the explosion of telecommunication and the Internet of Things later.

The Internet of Things or IoT has been widely known these days as a network of massive interconnecting smart devices, sensors, and actuators from home to workplaces and government authority agencies with smarter functionality, more efficient through the Internet. It is predicted to be the next big digital trend for the upcoming decades. The IoT's true value lies in its disruptive potential for reimagining business processes and, ultimately, rewiring business, government, and society. /1/

Within the IoT ecosystems, a low-power, wide-area network (LPWAN) was considered as one of the most popular parts of the IoT community because it was designed to simultaneously allow long range data transmission with low power consumption and to support a large number of users. With the effort to standardize LPWANs, LoRa developed by Semtech is becoming the most adopted LPWANs technology for the IoT paradigm, alongside Sigfox and NB-IoT. The key features of LoRa based solution are long range, low power, and low-cost installation.

LoRaWAN is a network software protocol based on the LoRa physical layer developed and maintained by a strong industrial standard - LoRa Alliance. LoRaWAN is deployed in a star topology which is perfect for long range or deep-in building communication operations. It can be scaled to an extremely large and flexible network for a huge number of devices with low power requirements.

Integration of smart objects, LoRa-based devices to the IoT, coupled with the LoRaWAN protocol has been applied around the world in supporting different fields such as:

- Smart Cities, Buildings & Homes
- Smart Agriculture
- Smart Supply Chain & Logistic

This thesis included six sections that carry out a deep research with the purpose to provide comprehensive knowledge about LoRa technology, LoRaWAN protocol in the IoT world. Throughout this thesis development, a final project was built, called a real-time LoRa-based GPS tracking system which can send data to the gateway by using only LoRa wireless.

2 THEORETICAL BACKGROUND

This section covers the foundations of wireless communication development as well as the evolution of the IoT. General principles behind the LoRa and LoRaWAN technology are presented as they are the most potential solution for LPWANs and the IoT ecosystem.

2.1 Wireless Communication Technology

Communication has always been an essential part of human civilization. Since ancient times, the loud voice was used to communicate in distances, then smoke was used as a signal to alert or notify people in different situations. In 1844, with the invention of a semaphore telegraph system based on electromagnetism, the very first famous Morse code of dots and dashes was sent over a wire. With this invention, Samuel Morse and his partner Vail took America and the rest of the world to an information era.

Later on, at the beginning of the 19th century, radio waves were discovered and studied by physicist Hertz using electromagnetism radiation theory. Then, an Italian inventor Marconi invented the wireless signal with radio transmitters and receivers. In 1901, the first coded message was sent across the Atlantic Ocean, this was a huge milestone in modern radio development as well as in communication technologies.

In the communication field, there are three main groups of methods on how IoT devices and sensors connect to the Internet and communicate, which are:

- **Device to Device:** data is transmitted and received directly between two devices using Bluetooth, Z-Wave, or Zigbee. This method is often used in a short range for small data packages, such as home automation, for example, Philips HUE light bulbs, thermostats, door locks, cameras.
- **Device to Cloud:** data is transmitted and received directly between devices and the Internet Cloud service. The device usually uses traditional communications, such as wired Ethernet or Wi-Fi, for example, smart TV.

- **Device to Gateway:** devices do not connect directly to the server network, but through a based station called the gateway, which has a user application software. Examples include a smartwatch without e-SIM (embedded-SIM) connected to a smartphone via Bluetooth, in this case, the phone is the gateway and it has WiFi or a 4G/5G connection.

As this thesis goes on, LoRaWAN devices will be studied how they are being used and how they will be developed in the near future.

2.2 The Internet of Things

The Internet of Things has been around for a long time, during the last ten years, it has gained public attention by the term IoT for its wide-range development in manufacturing. The IoT refers to physical things in a network which can talk to each other over the Internet. They can transmit, receive and analyse sensing data depending on purposes from homemade projects to industrial productions and become one of the hottest technology topics. IoT solutions can help to reduce installation and maintenance costs while simultaneously increase productivity with the use of intelligence.

2.2.1 The IoT Explained

The IoT provides the invisible bridge of humans with the physical world and virtual world in which every object in it connects to a network where the network can be the Internet, the company's intranet, or industrial communication protocol in a secure way. IoT objects can be any embedded electronics, vehicles, appliances, sensors, actuators, software, for example, a phone that controls a coffee machine at a desired time or a GPS tracking system in a car sending locations to a smartphone. The interconnected devices transmit data to cloud storage or a data center for automation processes or opening new services. These collected data can be used to deploy behaviors upon requests or can be used for analysis to improve the feature of a product or service.

Speaking of the Internet of Things, the most important part is the "things". "Things" are uniquely addressable endpoints or objects, using an IP address or URI.

The briefing concept of how the IoT works was demonstrated in Figure 2 below:

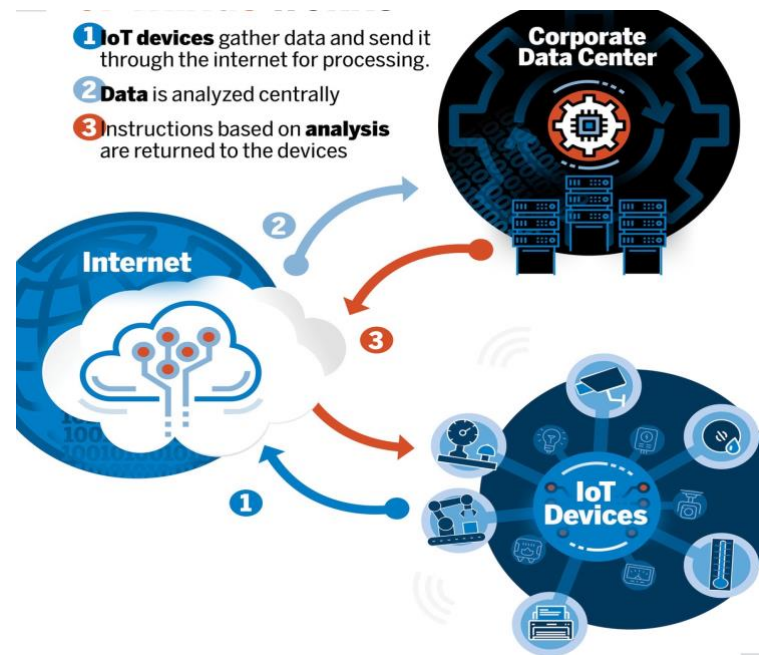


Figure 1. How the IoT works /2/

2.2.2 The IoT's Characteristics

The potential and availability of the IoT can lean on these four elements: /3/

1. Things: Intelligent devices and objects connected
2. Processes: Real-time information transmission
3. Analytics: Data processed and analyzed with algorithms
4. People: Users gaining valuable insights about their Things

Expanding four elements above, the fundamental of IoT are listed as follows:

- **Intelligence:** Various algorithms, software, and hardware are working under the surface to make smarter products and smoother services while humans interact with devices through a graphical user interface.
- **Connectivity:** An access point can provide the Internet connection or within the device itself. The IoT gateways are base stations that collect all the data from devices and sensors, then pass this information to the cloud.
- **Sensing:** The sensing monitors, actuators are the analog input sources for cloud storage. It can be said that the cloud server is like a human brain,

then these sensors are arms and legs. These sensors track and measure from different sources of data, for example, the change in temperature and humidity, physical activity of a person, or the usage of household appliances.

- **Communication:** connected devices can exchange data, so the data can be analyzed on the server. Communication in the IoT is varied, it can be short range, such as Wi-Fi, Bluetooth, or long range to extreme long range for over 15 kilometers, for example, LPWANs technology such as LoRa or NB-IoT.
- **Energy:** Energy consumption is the essential element for any working system. The sensing appliances of the IoT are functioning in harsh environments such as deep underground infrastructure, complex buildings among a huge number of devices running or outer space, these special conditions required ultra-low power to reduce the cost and efficiency of battery changing.
- **Cybersecurity:** This is a significant problem that IoT has been facing since the hackers can aim to attack the connected objects, which causes the loss of connection and disruptive data transmission or damage to the physical equipment themselves, systems, or products.

IoT is growing and this growth will continue and gain more success in the future while many organizations are forecasted to triple the number of connected devices in their systems for the next seven years, according to Forbes reports /4/.

Top industries using IoT are:

- Manufacturing
- Logistics and transportation
- Energy, oil, and gas
- Healthcare monitoring
- Industrial security systems
- Smart agriculture and environment solutions

2.2.3 Challenges for the IoT

Challenges at the Edge technology for connecting the Things to the Internet:

- Long battery life: the IoT devices must use low power consumption because it would cost a lot of money and a task force to replace a battery on a regular basis for a hundred or a thousand devices.
- Wide range and deep in-building penetration: There are monitoring sensors in hazardous places such as underground mining or in a complex of buildings which will reduce the signal.
- Ease of programming and intelligence: the IoT devices should be easy to program and control remotely, and the program that runs the application should be intelligent enough to make decisions right at the edge.
- Security: this is a must for every technology
- Remote management: Real-time data and controlling are key for IoT tasks.

2.3 Low-Power Wide Area Networks (LPWAN)

There are many wireless communications that can connect the devices to the Internet, one of them is Low-Power Wide Area Networks (LPWAN). LPWAN technology comes with a number of elements that magnify the unique and disruptive potential of the IoT world.

LPWANs initially was created for industrial IoT applications to send a small amount of data over a great distance for long battery life. The following names are the most outstanding competing technologies in the field: Narrowband IoT (NB-IoT), Sigfox, Lora.

LPWANs provide low cost, low power, and wide-range coverage areas.

- **Wide range:** The operating area of LPWAN technology can vary from a few kilometers in urban areas to over 15 km in rural settings. It can be enabled even in underground locations.

- **Low power:** LPWAN technology is optimized for low-power consumptions, one LPWAN transceiver can run on small, inexpensive batteries for 10 to 15 years, the IoT devices can switch off while they are not listening to any signals, which reduces significantly the maintenance costs.
- **Low cost:** Since the cost of hardware installations and maintenance is reduced, and the battery consumption is running at an ultra-low rate, LPWANs is optimized to the simplest protocol for long range data links while working efficiently with a star topology. With inexpensive infrastructure requirements and the use of license-free or already owned licensed bands, the network cost is reduced, therefore, the other aspects of LPWANs, such as security or quality of service can have more budget to spend.

3 LORA

3.1 Overview of LoRa

LoRa is Semtech’s RF modulation technology which is the LPWAN standard. The name LoRa is the reference for its extreme long-range data links while consuming a little power of batteries – one of the most important features of this technology. LoRa is meant for wireless, remote locations and usually, battery-operated things and LoRa transceivers can transmit data in the range up to 5 kilometers in urban areas and up to 15 kilometers in rural areas.

LoRa has mastered the long-distance connectivity challenges to power the industrial IoT. Within the vast area of the IoT, there is no single technology that can fulfill all of the IoT capabilities. Before the rising of LPWANs, monitoring or automation systems, such as wildlife tracking or measuring the conditions of plants for automated irrigation and nurturing systems, seemed to be impossible. With LoRa, this mission is accomplished, small sensor data packages in hazardous areas can be sent to central servers over the long distance, and the actuators or devices can be switched off to deep sleep while they are not listening to any signals to preserve power. LoRa is by far the most suited technology to conquer the difficulties that IoT had faced for a long time in the past.

Figure 2 gives a comparison between LoRa and other IoT technologies as well as M2M connectivity:


<u>Traditional Cellular</u> Long Range High Data Rates Low Battery Life High Cost	 LPWAN (3-5B in 2022) Long Range Low Data Rates Long Battery Life Low Cost High Capacity Potential	<u>Cat-M1</u> Long Range High Data Rates Low Battery Life Medium Cost
<u>Local Area Network</u> (Wi-Fi) Short Range High Data Rates Low Battery Life Medium Cost	<u>Narrow-Band IoT</u> (NB-IoT) Stationary Devices Short Range (indoor coverage) Low Data Rates Good Battery Life Low Cost	<u>Personal Area Network</u> (Bluetooth®) Very Short Range Low data rates Good Battery Life Low Cost

Figure 2. Networking connections /5/

LoRa's usages in industry and home life according to Robert Lie include/6/:

- Smart applications: Fuel monitoring for car or for heating systems tanks
- Health and hygiene: Heart rate monitor, waste management, temperature measurements
- Safety: Gas detection, the radioactivity level
- Efficiency: Asset and fleet management (for example, pallets and containers in Amazon warehouses)
- Agriculture: Automated hydroponic system, plants monitoring

3.2 Features of LoRa Technology

LoRa is a PHY - a physical layer protocol lying at the bottom of the OSI Model. The PHY is the part that runs an algorithm to modulate bits into electromagnetic waves which define radio signal, frequency, and modulation.

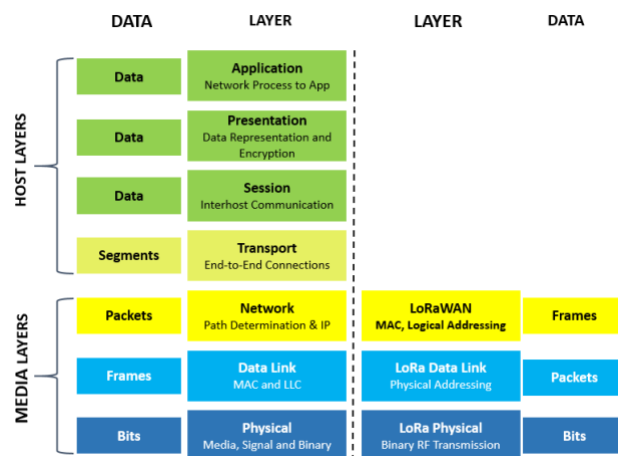


Figure 3. OSI Model /5/

LoRa consists of two main parts which are the end node and gateway.

- **End node:** The end node often runs on a battery that has a radio module with a printed antenna and a microprocessor to process the in and out data. If an end node has a wireless transceiver and sensors, it is called a remote sensor or a “mote”.



Figure 4. Example of LoRa end node /7/

- **Gateway:** A gateway also has a radio module with an antenna and a microprocessor to process all in and out data. The gateway is connected to mains power and the Internet. Multiple gateways can receive data from one end node.



Figure 5. Example of LoRa gateway /7/

3.2.1 Wide Area versus Power

Within dilemmas of the IoT applications, the “range” is the biggest one, since the range will lead to other problems and for small IoT devices, this problem becomes worse. Here are the reasons why the range is so important:

- **The distance versus transceiver power:** The further away a device from the gateway is, the more power it is required to operate, even in the line-of-sight condition. If the power is increased, the battery life is decreased. This is one of the reasons that the IoT devices are on sleep mode most of the time to save energy for transmission.
- **Bandwidth versus frequency:** The higher the bandwidth, the higher the frequency, meanwhile, the shorter the range is. This fact came from this formula for calculating the wavelength:

$$\lambda = \frac{v}{f} (m) \quad (1)$$

Where:

- λ (lambda) (m): the wavelength is the distance between two identical points in the adjacent cycle of a waveform signal
- v (m/s): velocity (speed) of a wave
- f (Hz): frequency of the wave
- **Interference:** The risk of interference always appears along with the travel distance. Sometimes, when the receiver and the transmitter are not in line-of-sight connection, they have to work with reflected signals which will increase the data loss or reduce the bandwidth.

The illustration of the bandwidth and range for networking is depicted in Figure 6.

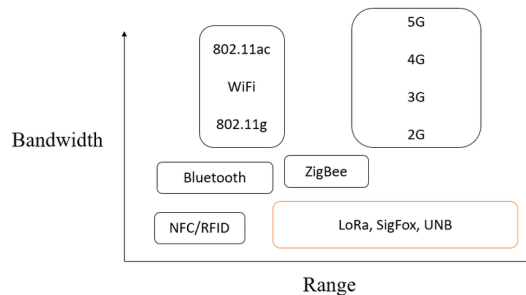


Figure 6. Bandwidth and range graph for networks /8/

In the 20th century, transmission or power towers helped people to transmit data over a long distance easily by using high voltage electrical power lines, and these towers were emitting a number of kilowatts of energy. To save energy and the environment, LPWAN and LoRa came to the feature of low power.

For a better understanding, wireless communication range and power comparison was combined in Table 1.

Table 1. Wireless communications comparison /6/

Wireless Technology	Wireless Communication	Range (m)	Tx Power (mW - milliWatts)
Bluetooth	Short range	~ 10	~ 2.5
LoRa	LPWAN	Urban area: ~2000 – 5000 Rural area: ~5000 – 1500 Direct line of sight: > 15000	~ 20
WIFI	Short range	~ 50	~ 80
3G/4G	Cellular	~ 5000	~ 500

The first three wireless technologies, all have short range operation compared to cellular, however, cellular technology uses a huge amount of energy compared to LoRa. Therefore, to operate along with these features, LoRa cannot send a big load of data such as video streaming or audio messages, only small data packages such as sensor data can be sent.

Sensor data packages are small, about 0.3 kbps to 5.5 kbps, and they only change a few times a day, given that the required energy is relatively low, for example, the humidity and temperature of a room, or a car position at the parking lot. When the devices are not transferring or listening to any signal, the power consumption measured is only in milliwatts (mW), allowing the battery to last for years and years.

Depending on the surrounding environment, the distance between the LoRa sender and receiver can vary in rural areas or in urban areas, the building material where the devices operate indoors is also one of the factors. For the direct line of sight which has no obstacles within the transmission path, the LoRa operative range is even higher.

3.2.2 Network

The difference between normal devices and IoT devices is that the IoT devices are capable of connecting to the Internet, and the number is expected to rise to billions of IoT devices in the next few years.

When billions of devices connect to a network, this network needs to be a standard because each IoT device will come from different manufactures and even the network itself.

As for LoRa devices, the LoRaWAN network is standardized and maintained by LoRa Alliance, following the international IoT standards from the IEEE Standard Association (IEEE SA) which are accepted by everybody /9/.

LoRa gateway can listen to multiple frequencies simultaneously, which means a lot more than one device (about a hundred devices) can communicate with the gateway at the same time. This network can be seen in Figure 8 below.

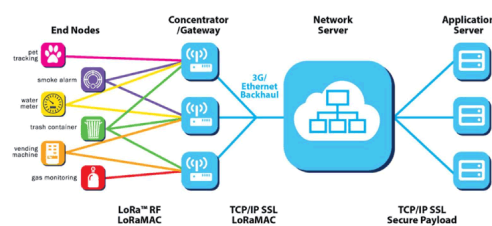


Figure 7. LoRa networks /5/

The communication between the gateway and end node is bidirectional, which means that the end nodes can send the signal to the network and application server through the gateway as well as receive the signal from the server. Transmission from end node to gateway is an uplink and transmission from gateway to end node is a downlink.

3.3 Operative Rules and Regulations

3.3.1 Radio Bands

LoRa operates in the international defining unlicensed ISM radio band. Because it is unlicensed, there is no license fee required and anyone can use frequencies in this

band. The local regulatory authorities in most countries will use standard created by the ETSI (European Telecommunications Standard Institute) or FCC (Federal Communications Commission), except for Japan and South Korea, they have their own standards.

Here are the ISM frequencies range for each region based on the information from “The Things Network”

Region	Frequency (MHz)	Region	Frequency (MHz)
Asia	433	Australia	915-928
Europe, Russia, India, Africa (parts)	863-870	Canada	779-787
US	902-928	China	779-787, 470-510

Figure 8. ISM frequencies band /10/

Despite all the upsides, there are some outweighs when using the ISM band which are low data rate and a lot of interference signals.

In Europe and in Finland, the frequency band is 863 MHz – 870 MHz complies with the following rules:

- The maximum transmission power for uplink is 25mW (14 dBm) and for downlink (869.525 MHz), is 0.5W (27 dBm).
- There are 0.1% and 1.0% duty cycle per day based on the channel.
- Maximum allowed antenna gain is +2.15 dBi.

In this thesis, The Things Network – a free public LoRaWAN will be used for demonstration, and here are their policies /10/:

- Uplink time on-air (airtime) is limited to 30 seconds for every 24-hour per node.
- The downlink message is limited to 10 messages for every 24-hour per node.

3.3.2 Duty Cycle – Time on Air

Time on Air (ToA) is the amount of time that the receiver has to wait until it receives the signal from the sender.

The duty cycle is the ratio of time a load or circuit is ON compared to the time the load or circuit is OFF. /11/ The duty cycle is expressed as a ratio or as a percentage of ON time.

Time on Air and duty cycle are directly connected to each other, examples in table 2 below:

Table 2. ToA and duty cycle examples

ToA (ms)	Duty cycle (%)	Waiting time (s)
530	1	$99 \times 530 = 52470 \text{ ms} = 52.47 \text{ s}$ (equivalent to 99%)
400	0.1	$999 \times 400 = 399600 \text{ ms} = 399.6 \text{ s}$ (equivalent to 99.9%)

After sending the first signal with the ToA of 530ms and duty cycle 1%, the sender must wait for 52.47s to resend the same size signal.

The purpose of the duty cycle is to manage the broadcasting sequence, without a duty cycle, anyone or any device can use the broadband for an unlimited time which will crash the whole communication system.

3.4 Fresnel Zone

There are always interference and signal loss in the path between the transmitter and the receiver. To ensure the least interference and lost signal, transmission paths are designed with a certain clearance distance determined by a Fresnel zone analysis.

The boundaries of these zones are elliptically shaped around the direct line of sight from the gateway to the end node. Anything within the Fresnel zone such as trees, buildings, the ground can reflect or bounce back the signal, which makes the power of the signal will be decreased. If the original signal and the reflected one are in the

same phase, the signal will be increased, otherwise, it will be decreased or dropped out.

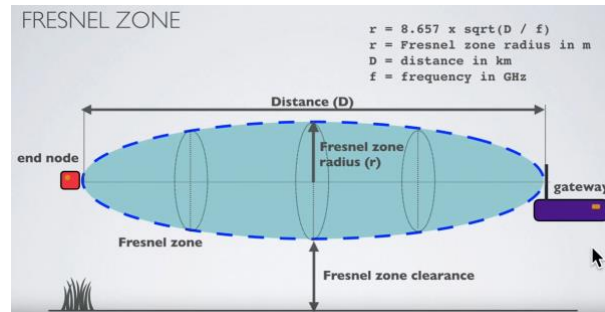


Figure 9. Fresnel zone /6/

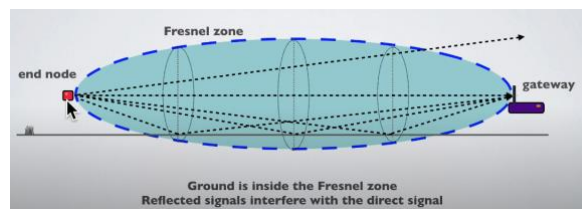


Figure 10. Example of interference in Fresnel zone /6/

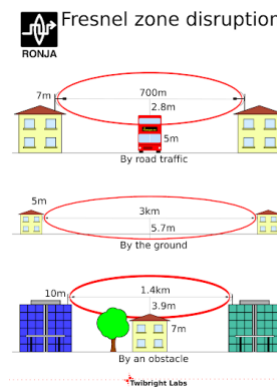


Figure 11. Example of obstacles within the Fresnel zone /6/

The maximum radius of this zone, based on the flat Earth is calculated using the following formula:

$$r = 8.657 \times \sqrt{\frac{D}{f}} \text{ (m)} \quad (2)$$

Where:

r: the Fresnel zone radius in meter (m)

D: the distance between end node and gateway in kilometer (km)

F: the frequency in gigahertz (GHz)

This formula does not take the curvature of the Earth into consideration, however, if the distance between the end node and the gateway is larger than 2km, then the height would be taken into account. Below is the picture describing this situation:

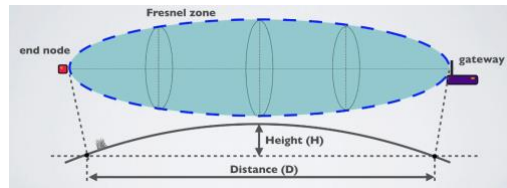


Figure 12. Curvature consideration /6/

The height H is calculated using the following formula:

$$H = \frac{1000 \times D^2}{8 \times R_{Earth}} \text{ (m)} \quad (3)$$

Where:

H is the height or the earth curvature allowance in meter (m)

D is the distance between gateway and end node in kilometer (km)

$R_{earth} = 8504$ km is the Earth radius in kilometer (km)

It is always better to avoid any construction or object within the Fresnel zone but in real life, this zone cannot be completely clear. Therefore, it is denoted as the Fresnel zone will work at the maximum of 40% blockage, out of this range, the loss will be too significant to handle.

Following is the formula for 60% clear of the Fresnel zone (or 40% blockage) based on flat Earth:

$$r = 8.657 \times \sqrt{\frac{0.6D}{f}} \text{ (m)} \quad (4)$$

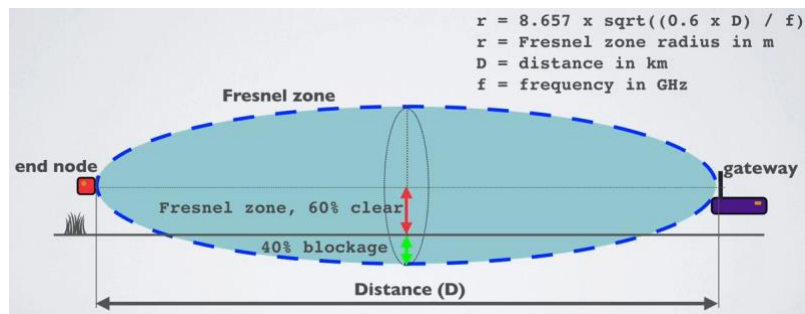


Figure 13. Fresnel zone with 40% blockage based on flat Earth /6/

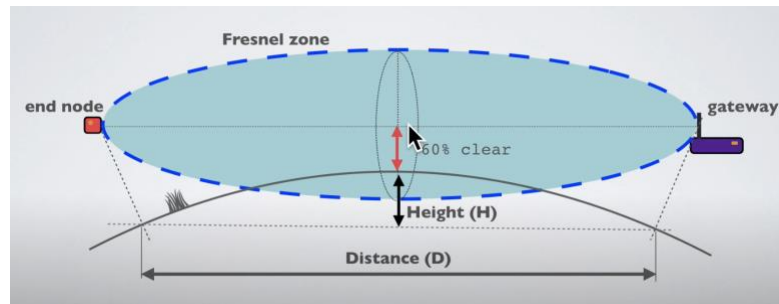


Figure 14. Fresnel zone with 40% blockage considering curvature of the Earth /6/

3.5 Radio Frequency Modulation

Modulation is the definition of encoding analog or digital information into a carrier signal. Then, the modulated signal is broadcasted to the receiver.

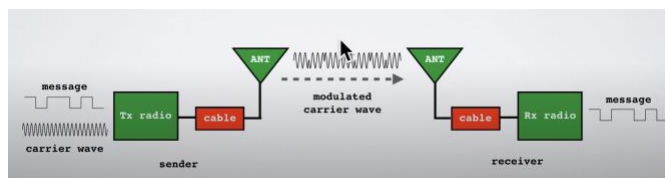


Figure 15. Signal modulation /6/

For analog signals, three encoded modulation types are used: Amplitude Modulation (AM), Frequency Modulation (FM), and Phase Modulation (PM).

For digital signals, three basic encoded modulation types are used: Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK), and Phase Shift Keying (PSK).

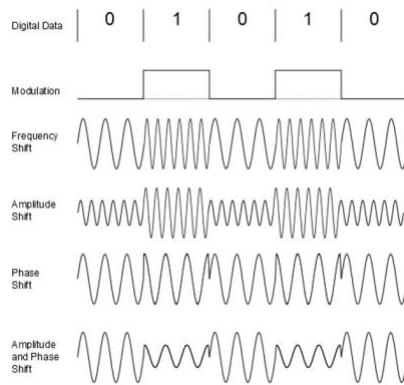


Figure 16. RF modulation /12/

Other modulation methods can be formed by combining these three basics techniques, however, LoRa is a proprietary spread-spectrum modulation technique that is similar to FSK based on the existing Chirp Spread Spectrum (CSS).

3.6 Link Budget

The link budget is a way to measure the transmission performance between the transmitter and the receiver. It is the sum of all of the gains and losses via mediums. It is specified that:

- The radio transmitter value must be specified in dBm.
- Gain: through an antenna (dBi).
- Loss: by cables, connectors, the signal propagating via the medium (dB).
- Path loss of path attenuation: when the signal loses its strength during propagation via the medium.

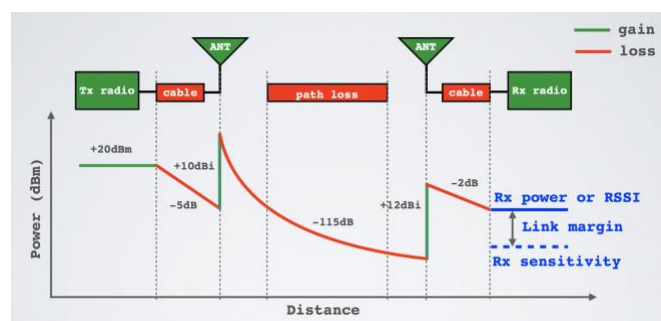


Figure 17. Link budget and link margin /6/

Following is the simple equation of the link budget:

$$P_{received} = P_{transmitted} + Gains - Losses \quad (5)$$

(dBm)

Where: P is power

The receiver's sensitivity (Rx) is the lowest power level at which the receiver can get or demodulate the signal.

The link margin is the difference between the receiver power and the receiver sensitivity. All is defined with the unit of dBm.

3.7 EIRP and ERP

3.7.1 EIRP

EIRP stands for Effective Isotropic Radiated Power. It is the total power radiated by a hypothetical isotropic antenna (which will be explained later) in a single direction.

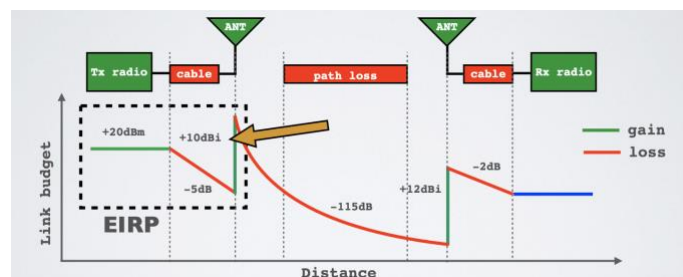


Figure 18. EIRP demonstration /6/

EIRP calculation:

$$EIRP = P_{Tx} + antenna\ gain - cable\ loss \quad (6)$$

(dBm)

Where:

P_{Tx} is transmitted power (dBm)

Antenna gain (dBi)

Cable loss (dBm)

3.7.2 ERP

ERP stands for Effective Radiated Power. It is the total power radiated by a real antenna related to a half-wave dipole antenna (which will be explained later).

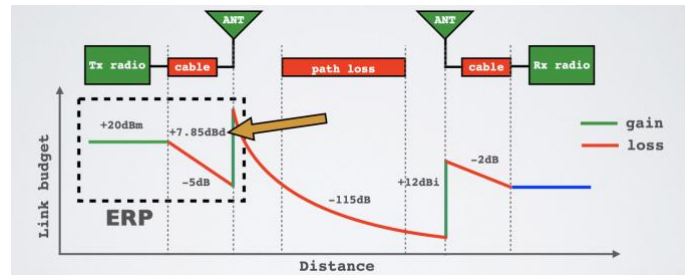


Figure 19. ERP demonstration /6/

ERP calculation:

$$ERP = P_{Tx} + \text{antenna gain} - \text{cable loss} \quad (7)$$

(dBm)

Where:

P_{Tx} is transmitted power (dBm)

Antenna gain (dBd)

Cable loss (dBm)

EIRP and ERP relation:

$$EIRP = ERP + 2.15 \text{ (dBm)} \quad (8)$$

EIRP and ERP are used to set some of the radio frequency rules.

3.8 RSSI and SNR

3.8.1 RSSI

RSSI stands for the Received Signal Strength Indication. It is received signal power in mW which is measured in dBm. This index is used to measure the performance of a receiver if it can get the signal from the transmitter well.

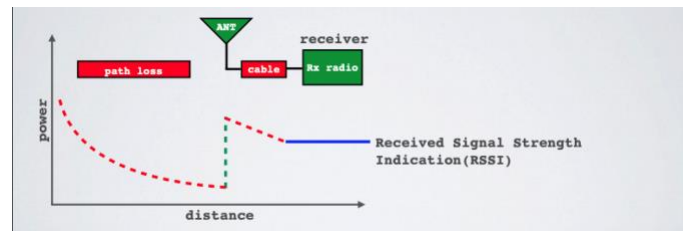


Figure 20. RSSI /6/

RSSI is measured in dBm as a negative value, therefore, the closer the value getting to 0, the better the signal is at the receiver.

It is denoted as if:

- RSSI = -30 dBm, this is a strong signal
- RSSI = -120 dBm, this is a weak signal

3.8.2 SNR

SNR stands for Signal-to-Noise Ratio, it is the ratio between the desired receiving power signal and the power of the noise in the background.

SNR unit is often in decibels (dB). Figure 21 below describes how the received signal operates regarding the SNR:

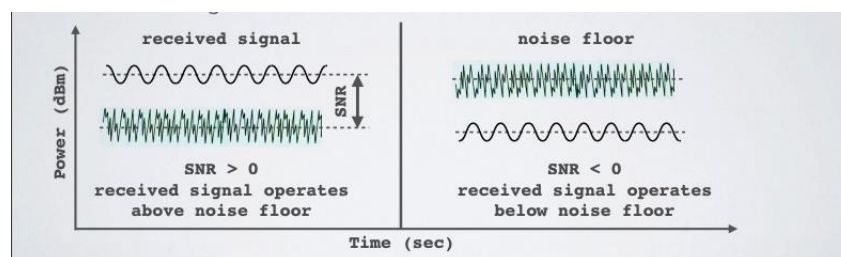


Figure 21. RSSI and SNR relation /6/

LoRa operates below the noise level. LoRa devices can demodulate signals in the range of -20 dB and -7.5 dB below the background noise.

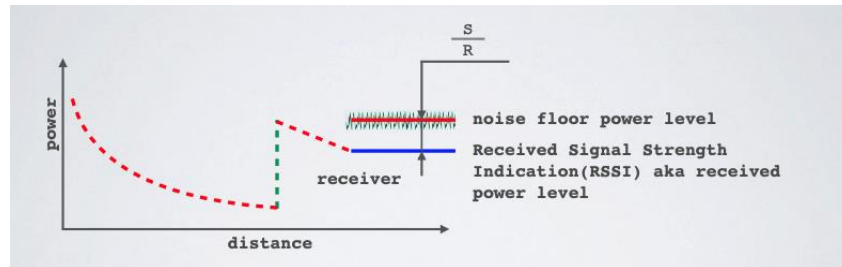


Figure 22. SNR of LoRa /6/

The typical LoRa's SNR indexes are from -20 dB to 10 dB, the more the value is near 10 dB, the less signal is interfered.

4 LORAWAN

4.1 Overview of LoRaWAN

LoRaWAN is a global Software Protocol that uses the LoRa PHY to define the format of the data packet and the way they are processed by the network and devices. LoRaWAN is specifically designed for wireless LPWAN and to fulfill the requirements for IoT devices. It is developed and managed by LoRa Alliance – a non-profit association that includes hundreds of organizations and enterprises such as Amazon, Semtech, Cisco, and Digita Oy.

LoRaWAN is described by LoRa Alliance as:

“The LoRaWAN specification is a Low Power, Wide Area (LPWA) networking protocol designed to wirelessly connect battery operated ‘things’ to the Internet in regional, national or global networks, and targets key Internet of Things (IoT) requirements, such as bi-directional communication, end-to-end security, mobility and localization services.” /13/

LoRaWAN makes the IoT solutions are more affordable and long-lasting, reducing maintenance to a minimum.

LoRaWAN propagates better and offers better redundancy than other radio protocols.

4.1.1 Main Components

The main components of the LoRaWAN are:

- Network Server
- Application Server
- Join Server

These three components carry mechanisms such as configuring radio parameters (for example, data rates and channels, time settings), message integrity checking and application payload encryption.

4.1.2 Frequency Bands and Regulations

LoRaWAN operates in the unlicensed spectrum ISM bands, such as WiFi and Bluetooth, at a low frequency band (below 1 GHz), given that low frequencies allow wide range communication following formula (1). In different regions, different LoRaWAN frequencies can be used.

In Europe, the allowance frequencies are between 863 and 870 MHz, called EU868 and 10 frequency channels can be used in this band with the following specifications:

- 8 channels can transfer data up to 5.5 kbps.
- 1 channel has a double bandwidth with 11 kbps.
- For FSK modulation, 1 channel can transfer up to 50 kbps.
- There are 3 defined fixed channels that must be supported by all devices, which are 868.1, 868.3, and 868.5 MHz. This feature will allow any device to join any network.

LoRaWAN is also subjected to regional and national regulations. In Europe, EU868 follows the duty cycle of transmission to limit the amount of time for broadcasting signals. The general duty cycle of EU868 is 0.1%, 1%, and 10%.

Power per channel band limitation on transmission is fixed worldwide.

4.1.3 Data Rates

The allowed baud rate is between 0.3 kbps and 11 kbps for LoRa modulation, it can be up to 50 kbps for FSK modulation.

Table 3. below shows how the difference of data rates impacts the performance of LoRa's devices.

Table 3. Data rate comparison

High data rate	Low data rate
<ul style="list-style-type: none">- Small spreading factor- High chances of packet loss	<ul style="list-style-type: none">- High spreading factor- More ToA- More power consumption- Longer range- Reduce gateway capacity- Duty cycle is affected

4.2 Spreading Factor

The spreading factor is the most important feature in LoRaWAN that determines the long-term performance of LoRaWAN.

Spreading Factor (SF) decides on how many chirps, the carrier of the data, are sent per second. /14/

Based on the environmental conditions between the device and the gateway, this SF is graded on a scale from 7 to 12. The differences between low and high SF in theory are in Table 4 below.

Table 4. Spreading factor comparison

Low SF	High SF
<ul style="list-style-type: none">- More chirps are sent per second- Encode more data per second	<ul style="list-style-type: none">- Fewer chirps are sent per second- Encode fewer data per second
For the same amount of data	For the same amount of data
<ul style="list-style-type: none">- Less ToA- Less running time- Less energy consumption	<ul style="list-style-type: none">- More ToA- The end will be up and running longer- More energy consumption

Even though high SF seems to have many disadvantages, it has its own benefit for the extended ToA, which means that the receiver has more chances to sample the signal power and has a better sensitivity and better coverage area.

More importantly, to be able to examine the energy consumption, the spreading factor is not the only feature to measure.

4.3 Advantages and Disadvantages

4.3.1 Advantages

The advantages of LoRaWAN include:

- LoRaWAN is a standardized protocol for LPWAN, it has long range, low power operating, and low cost for installing and maintenance.
- Low bandwidth, depending on the SF.
- Easy for everyone anywhere to start with, one installed gateway to connect to the network.
- 128bit end-to-end encryption for payload security.
- LoRaWAN works great for vehicles in mobility, it does not have to handle cell-to-cell work like cellular.

4.3.2 Disadvantages

The disadvantages of LoRaWAN are:

- LoRaWAN is not suitable for real-time data, only small packets of data can be sent over a couple of minutes.
- LoRaWAN cannot do phone calls or controlling lights house like ZigBee or Bluetooth.
- Since only small data packets are allowed, LoRaWAN cannot deal with a huge load of data, such as streaming live video, sending photos or social media web surfing.

4.4 Security

Security is the top priority for IoT implementation, especially for industrial deployments.

There are two phases of the security at LoRaWAN:

- Join procedure: the stage to establish mutual connections between the end node and the network that it aims to connect.
- Message authentication: to ensure that only authorized devices can join the network.

The LoRaWAN security cryptography is defined as two layers:

- 128-bit AES Network Session Key (NwSKey)
- 28-bit Application Session Key (AppSKey)

These steps and cryptography techniques will ensure:

- Network traffic stable
- Only authenticated devices can access the network
- No eavesdropping on the network
- Network traffic cannot be captured or replayed

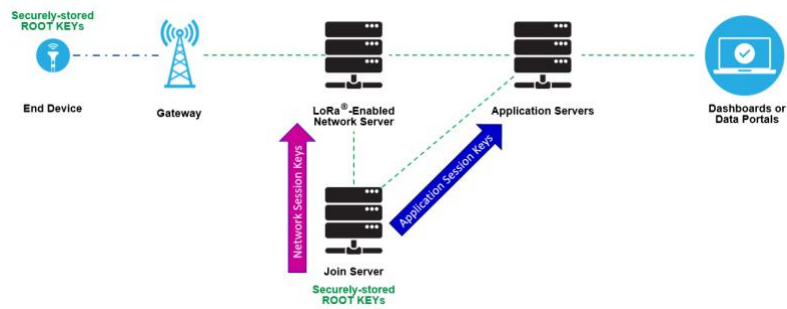


Figure 23. Security Session Keys are shared within LoRaWAN /5/

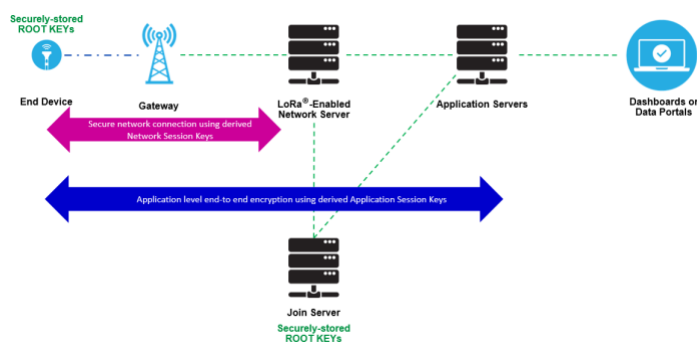


Figure 24. Data packets securely transmitted from end to end /5/

4.5 Device Classes

LoRaWAN is a MAC protocol configured based gateway and end nodes, sometimes the star topology connection network of gateways is required to work with multiple end nodes.

From the OSI model (Figure 4), LoRaWAN is spanning three layers on top of the LoRa PHY which are the session layer, network layer, data link layer.

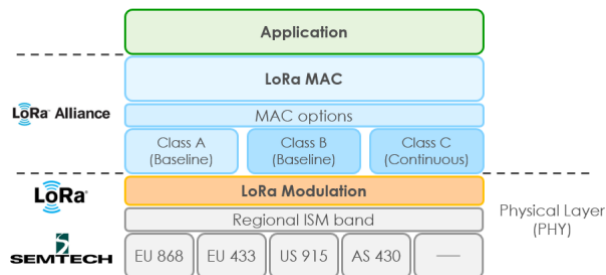


Figure 25. LoRa protocol stack /5/

LoRa devices run three modes according to their device classes.

- Class A (All): devices are battery-powered and supported by all classes.
- Class B (Beacon): devices also work on battery and have extra receiving windows on schedule.
- Class C (Continuous): devices also work on battery, but often work on mains powered because they consume much power due to continuously listening.

The devices in these classes have to follow these rules:

- All other classes must support class A devices.
- Class B must support class A and B devices.
- Class C support all other class devices.

4.5.1 Class A Device

Class A devices can commute bidirectionally. At any time, when a change takes place in the monitored environment of the end node, an uplink will be initiated and transmitted to the network (Tx).

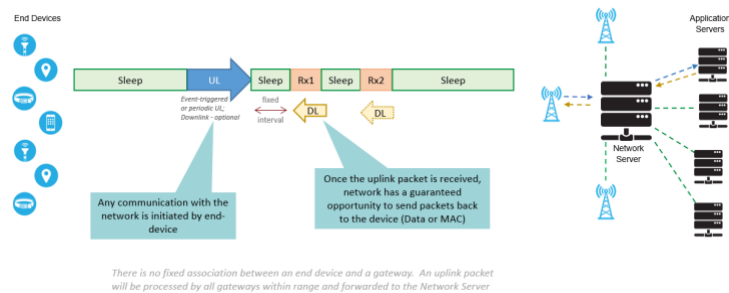


Figure 26. How class A devices operate /5/

The whole mode A operation will be described through Figures 27, 28 and 29.

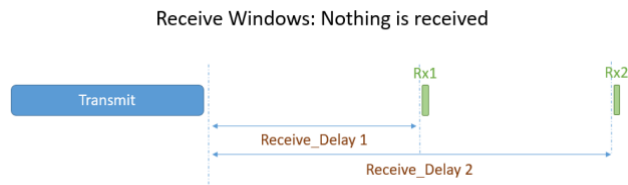


Figure 27. End node after data transmitting /5/

The end node will open two windows at *Receive_Delay 1* and *Receive_Delay 2* to listen for the response from the network server (gateway), typically for 1 second, this duration can be configured.

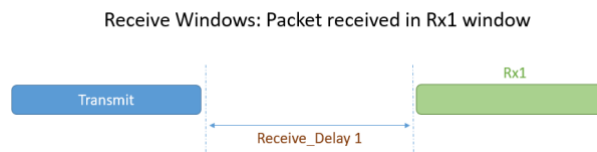


Figure 28. Data is received in window 1 (Rx1) /5/

The end device is up to receive a downlink in the first window (Rx1), if it does not, it will go back to the idle state and wake up a moment later for response at window Rx2.

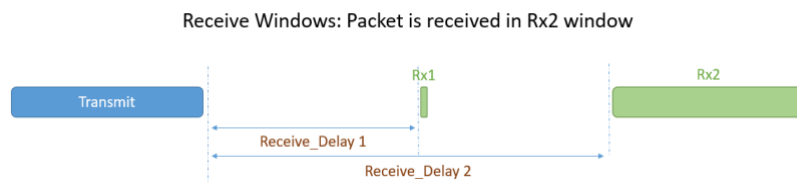


Figure 29. Data is received in window 2 (Rx2) /5/

If there is no downlink during the Rx2 window as well, the end device will go back to sleep mode until there is new data to report.

The gateway can only respond within either *Receive_Delay 1* or *Receive_Delay 2*, not in both of them, and until either of these durations get a downlink or the whole transmission is completed, no uplink can be sent out by an end device.

Class A devices do not fit actuators.

4.5.2 Class B Device

Class B mode is a mixture of classes A and C, it can commute bidirectionally and has extra scheduled receive windows.

A synchronized beacon is periodically broadcasted from the server through a gateway, then the end node can periodically use these beacons to derive and align its internal time with the network.

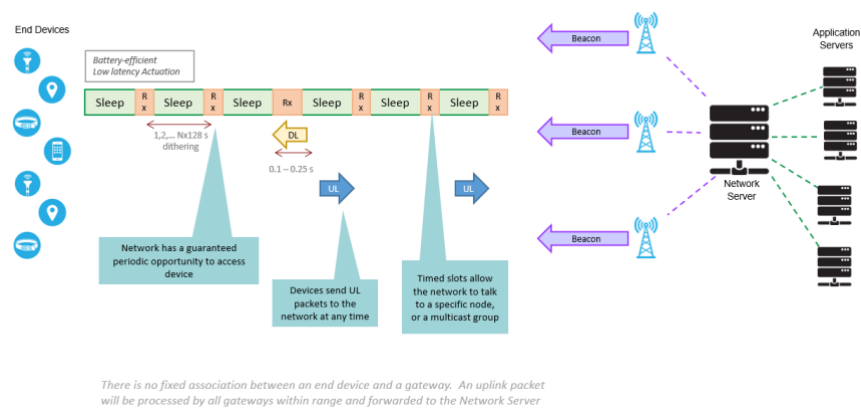


Figure 30. How class B(Beacon) devices operate /5/

The detailed process is described through Figure 31 and 32:

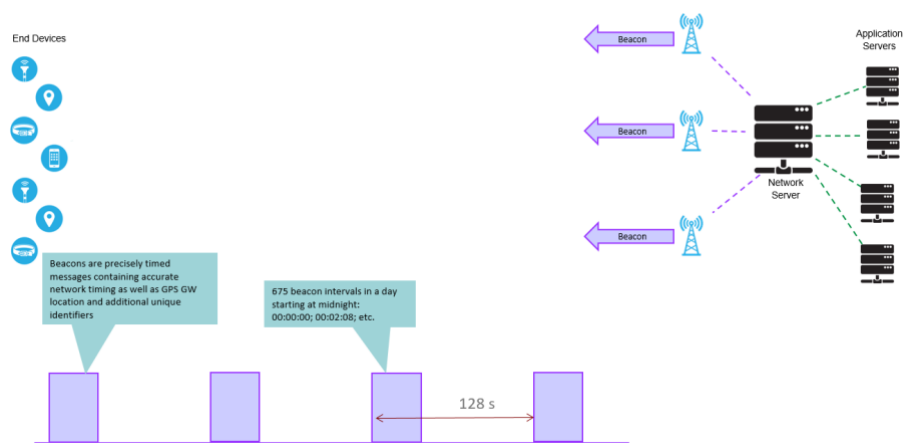


Figure 31. Beacon intervals /5/

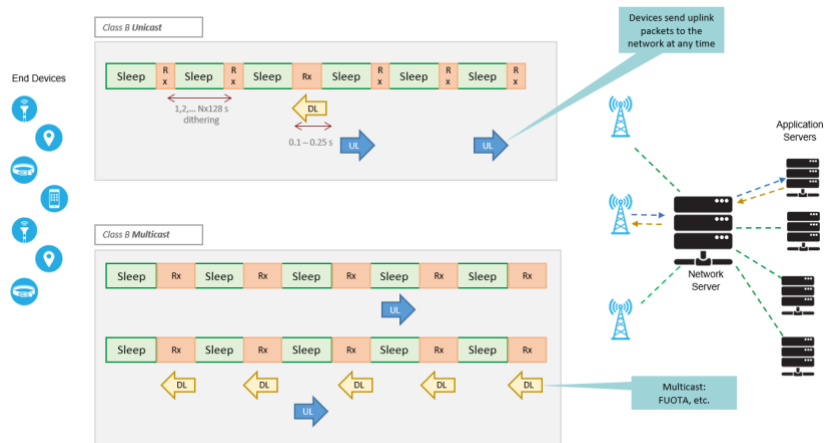


Figure 32. Beacon ping periods /5/

With the beacon timing, the gateway knows that the end device is up for listening and it will send the downlink in any receive window (ping period).

All the devices in the network must have a built-in GPS timing function in order to support class B devices.

Class B devices fit both actuators and sensors.

4.5.3 Class C Device

C stands for continuously, which means that class C devices stay up for listening to the downlink until there is a next uplink ready, this advantage brings the latency to the lowest and also consumes so much power that the end device cannot live on battery.

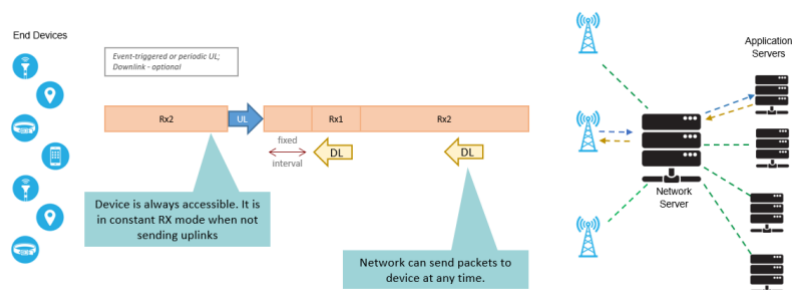


Figure 33. How class C devices operate /5/

Class C devices also have two receive windows like class A, but they keep the second window (Rx2) open to receive downlinks at any time as described in Figure 33.

Class C devices fit for streetlights and electrical meters.

4.6 Annexes

4.6.1 DeciBel

DeciBel is the unit to express the ratio of physics quantities such as electrical powers or sounds on a logarithmic scale.

The logarithmic scale is easier to work with when expressing an extremely big or extremely small number.

DeciBel is denoted as dB and dB is not an absolute value, dB means the only ratio.

Here is the formula to calculate the dB value:

$$A = 10 \times \log_{10} \left(\frac{P_o}{P_i} \right) \quad \text{dB} \quad (9)$$

Where:

- P_o is the output power
- P_i is the input power

To convert dB to power ratio, use the following formula:

$$\frac{P_o}{P_i} = 10^{(A/10)} \quad (10)$$

When using the reference of fixed value 1mW for P_i , the deciBel unit is expressed as dBm, this dBm expresses an absolute value.

4.6.2 dBi and dBd

These two units are used to measure the performance of the antenna and the comparing purposes.

dBi is an isotropic antenna gain, this is a hypothetical antenna used for measuring real antenna characteristics. This hypothetical point source antenna has a sphere pattern and radiates its power in all directions equally.

Isotropic Radiation Pattern

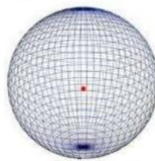


Figure 34. Isotropic antenna radiation pattern /15/

dBd is dipole antenna gain. “Dipole” means “two poles”, hence, a dipole antenna is two antenna sections, such as metal wire or rod connected to the same feeder or coaxial cable (insulator). It is also called a half-wave dipole antenna and this is the simplest and most widely used antenna. The dipole antenna radiates in a pattern of a donut.

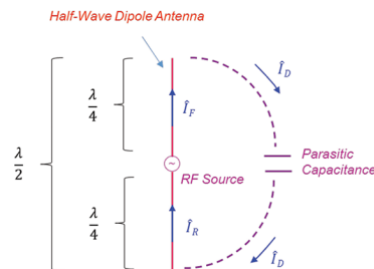


Figure 35. Half-wave dipole antenna /15/

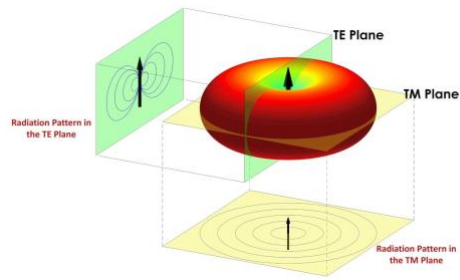


Figure 36. Dipole antenna radiation pattern /15/

dBd and dBi connect to each other following the formula:

$$dBi = dBd + 2.15 \quad (11)$$

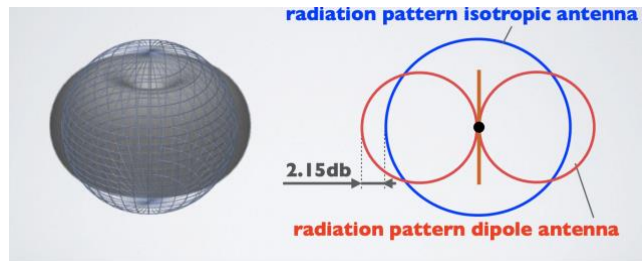


Figure 37. dBi and dBd relationship /15/

5 GPS TRACKER

GPS stands for Global Position System which is a satellite-based navigation system. This system provides continuous real-time positioning information anywhere in the world and under any weather condition.

In a GPS tracking system, the GPS module gets a satellite signal and sends out raw GPS NMEA sentences, these signals are processed to filter the information, such as longitude, latitude, altitude, date, time and so on. Finally, these coordinate data are sent to the tracking application to display the current location of the device. At this point, an IP network connection is needed in order to send the data from the GPS module to the application.

With a LoRa based GPS device, only an Internet connection is needed for the gateway. The LoRa GPS end node can send data to the gateway only through LoRa wireless.

This project demonstrates how to build a LoRa based GPS system, using a Dragino gateway and TTGO T-Beam ESP32 with LoRa chip and GPS chip on board.

The T-Beam board will be put in a car as a LoRa GPS tracking node and it will send the GPS data to the gateway using LoRa wireless. The Dragino gateway will encode the data and send it to The Things Network server and transfer it to the tracking application Cayenne backend service. This system is an example of LoRaWAN.

The hardware components used in the project:

- Dragino gateway LG01N single channel
- TTGO T-Beam ESP32 board v1.1
- Battery 3.7V 2950 mAh
- Micro USB cable
- Jumper wires
- Si7021 temperature and humidity sensor
- RJ45 cable

The software components used in the project:

- Arduino Integrated Development Environment (IDE) 1.8.13
- The Things Network (TTN)
- myDevices Cayenne service

User accounts:

- Create a free account on myDevices
- Create a free account on The Things Network

5.1 The Things Network – TTN

The Things Network (TTN) is a free LoRaWAN server open to all. A gateway registered to TTN can be reached by potential nearby sensor nodes.

TTN stays between the gateways and the third-party applications

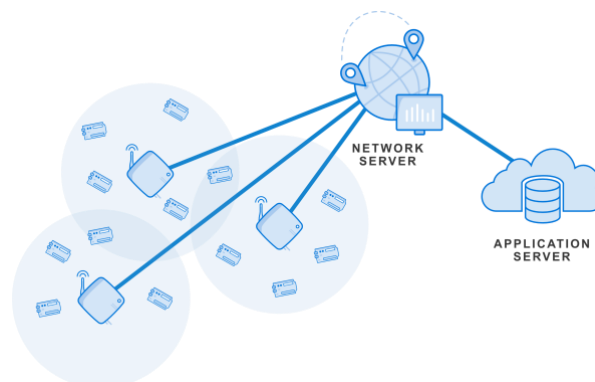


Figure 38. TTN server in LoRaWAN /16/

There are two ways to work with TTN, using the CLI or a UI console. In this project, the TTN Console was used to monitor the use of applications, devices, and network.

The figures below demonstrate simple steps to start working with TTN.

1. After creating an account, log in to the TTN platform using this links:
<https://account.thethingsnetwork.org/users/login>
2. After successfully logging in, go to the Console from the right top of the page and choose either Applications or Gateways to start with.

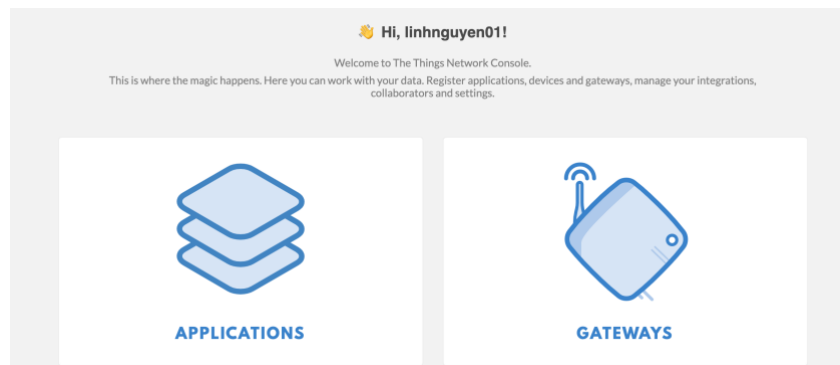


Figure 39. TTN Console dashboard

5.2 Dragino Gateway

5.2.1 Overview

This project used a LG01N Dragino gateway which is an open-source single channel LoRa gateway. The GPS data was sent through this gateway to TTN. This gateway has a long range spread spectrum and high interference immunity.



Figure 40. LG01N Dragino gateway /17/

The Dragino gateway has its own built-in LoRa chip and the WiFi/Internet chip, the sensor nodes connect to this gateway using LoRa frequency (868MHz in EU).

/17/

5.2.2 Configuration of the Dragino Gateway

The configuration of the Dragino gateway was done as follows:

- 1) Connect the antenna to the gateway, power up the gateway through the 12V-1A port with the adapter. Using RJ45 cable connect the LAN port of the gateway to the PC or laptop being used to set up.



Figure 41. Dragino wiring

2) Open the web browser and go to the Dragino login page using the link <http://10.130.1.1/cgi-bin/luci/admin> or just simply type the IP address of “10.130.1.1”.

3) The account for login page:

Username: root

Password: dragino

4) Access the Internet as a WiFi client by the following steps:

Step 1: Go to Network → Wireless, select radio0, and press Scan

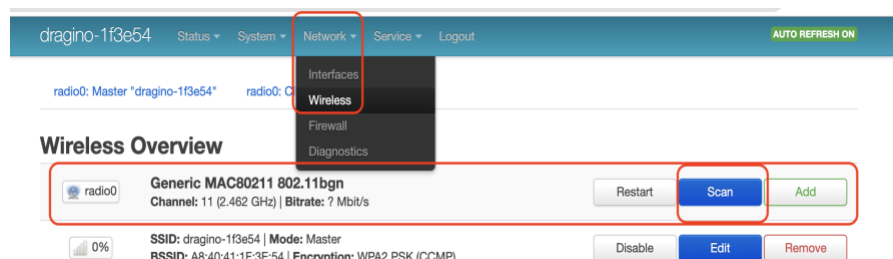


Figure 42. WiFi network scanning

Step 2: Select correct WiFi SSID and join, filling in the password into WPA passphrase and place any name for new network

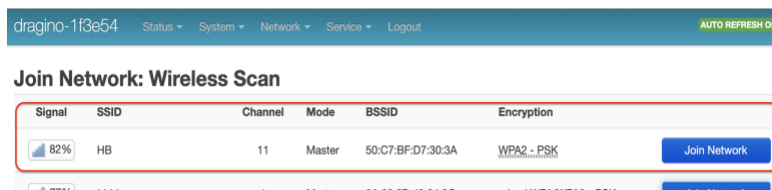


Figure 43. Join WiFi network 1

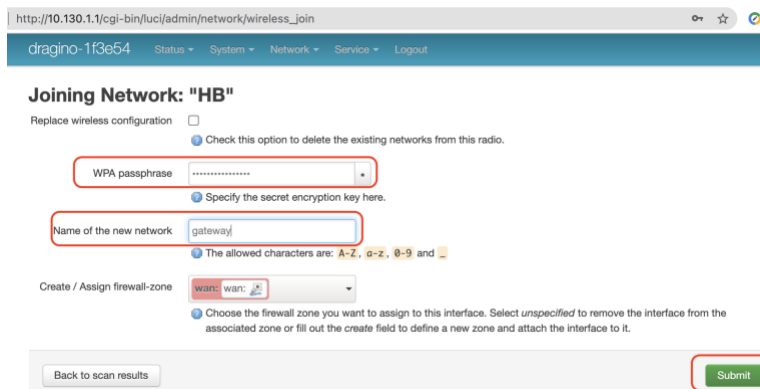


Figure 44. Joining WiFi network 2

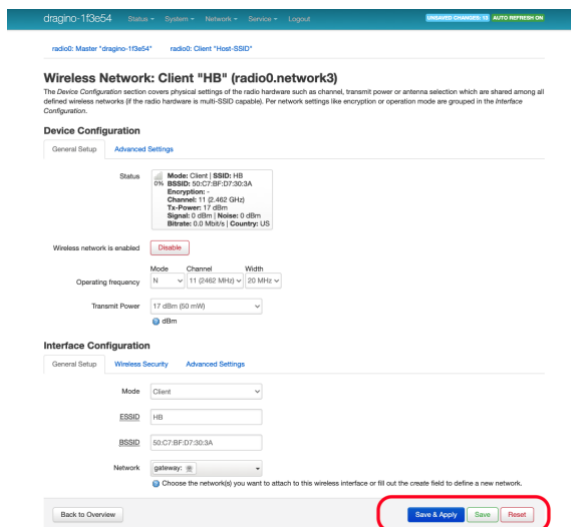


Figure 45. Save & Apply joining network

Step 3: Default Dragino SSID makes the gateway as an Access Point (AP), go back to the Wireless Overview and disable the default SSID to connect to the Internet



Figure 46. Disable the default Dragino AP

5) Open Gateways from TTN's Console to register a new gateway.

Copy the Gateway ID from Service → LoRaWAN GateWay and paste it to the Gateway EUI

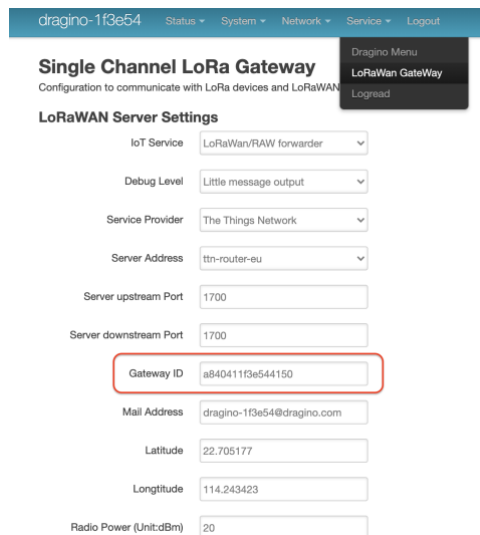


Figure 47. Finding the Gateway EUI

Select "I'm using the legacy packet forwarder"

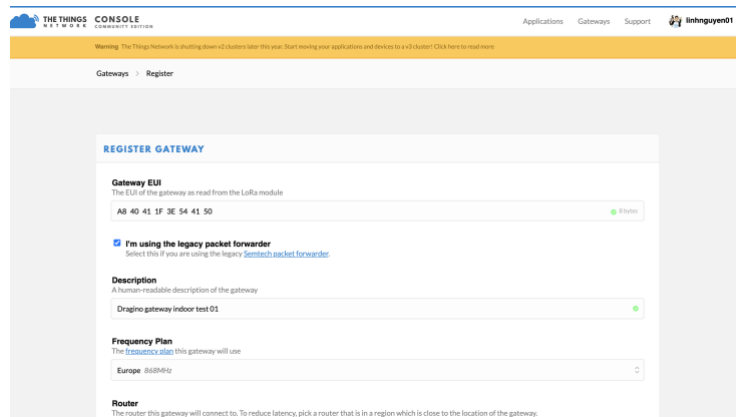


Figure 48. Register gateway on TTN 1

Choose the correct Frequency Plan (Europe 868 MHz) and modify the position on the map, choose one of the Antenna Placement then press Register Gateway.

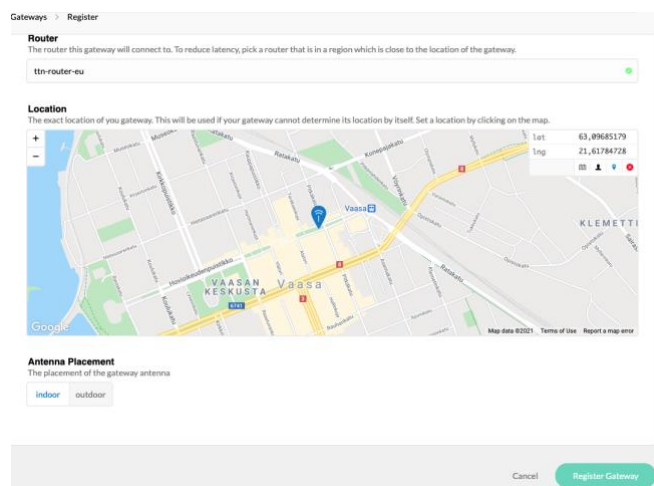


Figure 49. Register gateway on TTN 2

Verify that the gateway is connected.

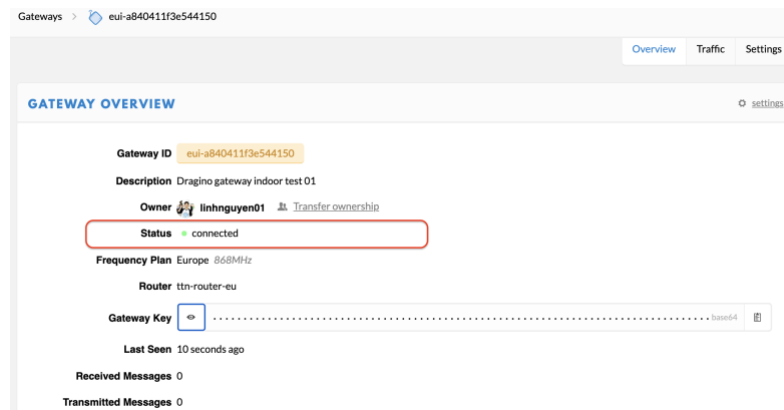


Figure 50. Gateway Overview

6) Go back to the Dragino setup page to check and modify some values:

Service Provider: The Things Network

Radio Power: 20

Frequency: 8681000000

The other fields can be left as default and click Save & Apply button as in figure 54 and 55 below:

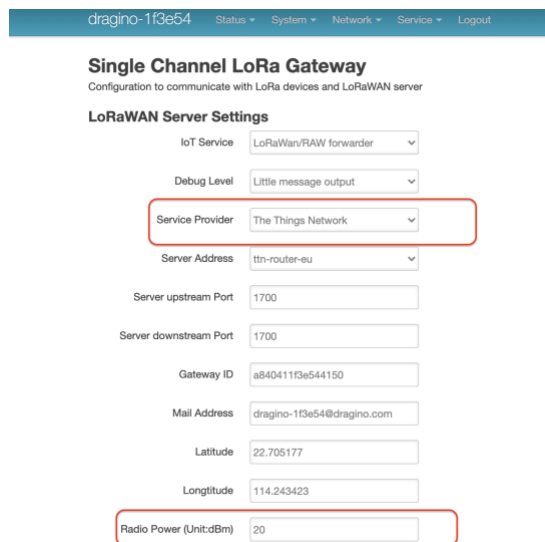


Figure 51. Finish registering gateway 1

Radio Settings
Radio settings for Channel

Frequency (Unit:Hz)

Spreading Factor

Coding Rate

Signal Bandwidth

Preamble Length
 Length range: 6 ~ 65536

LoRa Sync Word
 Value 52(0x34) for LoRaWAN

Figure 52. Finish registering gateway 2

5.2.3 Creating an Application on TTN

An application is created on TTN as follows:

Open the Application as in figure and click Add Application, fill the name and choose “ttn-handler-eu” for Handler registration, click Add application to create it.

Applications > Add Application

ADD APPLICATION

Application ID
The unique identifier of your application on the network

Description
A human readable description of your new app

Application EUI
An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page.

Handler registration
Select the handler you want to register this application to

Figure 53. Register application on TTN

The next target is to register a device which is the TTGO T-Beam board on the next chapter.

5.3 TTGO T-Beam ESP32

5.3.1 Overview

TTGO T-Beam v1.1 from LilyGo brand is an ESP32 board which has a LoRa transceiver and GPS chip on board, and it comes as pre-flash firmware board with an external 868 MHz antenna.

Below is a picture of T-Beam pin map with its following specifications

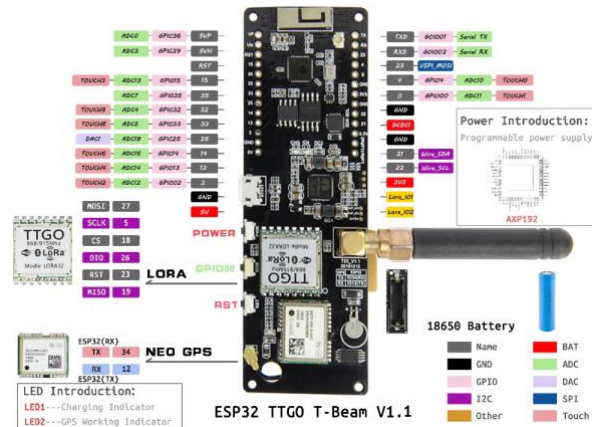


Figure 54. T-Beam pin map /18/

Hardware of main board:

- ESP latest version Rev1
- Wifi
- Bluetooth
- 4 MB flash
- 8 MB PSRAM
- 3D antenna

Lora module:

- Operating voltages: 1.8V ~ 3.7V
- Accepted current: 10 ~ 14 mA
- Transmit current:
 - 120 mA at +20 dBm
 - 90 mA at +17 dBm
 - 29 mA at +13 dBm
- Sleep current: 0.2 μ A at sleep and 1.5 μ A while idle
- Transmit power: +20 dBm
- Frequency: 868 MHz
- Frequency error: +/- 15 KHz
- FIFO space: 64 Byte

- Data rate: 1.2 Kbps ~ 300 Kbps at FSK
0.018 Kbps ~ 37.5 Kbps at LoRa
- Modulation mode: FSK, GFSK, MSK, GMSK, LoRa TM, OOK
- Interface: SPI
- Temperature: -40°C - +85°C

GPS:

- Module: NEO – M8N
- Ceramic antenna
- Default Baud rate: 9600

Power:

- Power supply input: USB 5V/1A
- Charging current: 500 mA
- Battery input: 3.7V – 4.2V

5.3.2 Arduino IDE and TTGO T-Beam Setup and Installation of the Arduino IDE

The Arduino IDE is used to write, compile and upload code to the Arduino board. The Arduino IDE (1.8.13) for Mac OS X is used in this project to program the T-Beam board. It can be downloaded with the appropriated operating system from this page: <https://www.arduino.cc/en/software>

The following steps are needed to install the IDE for Mac OS X:

1. Download the installer by clicking “Mac OS X 10.10 or newer”
2. The installer is downloaded as ZIP file, double click to extract it
3. Open the Arduino IDE with the “.app” extension

Figure 47. below is the main working platform of the Arduino IDE with number labeled.

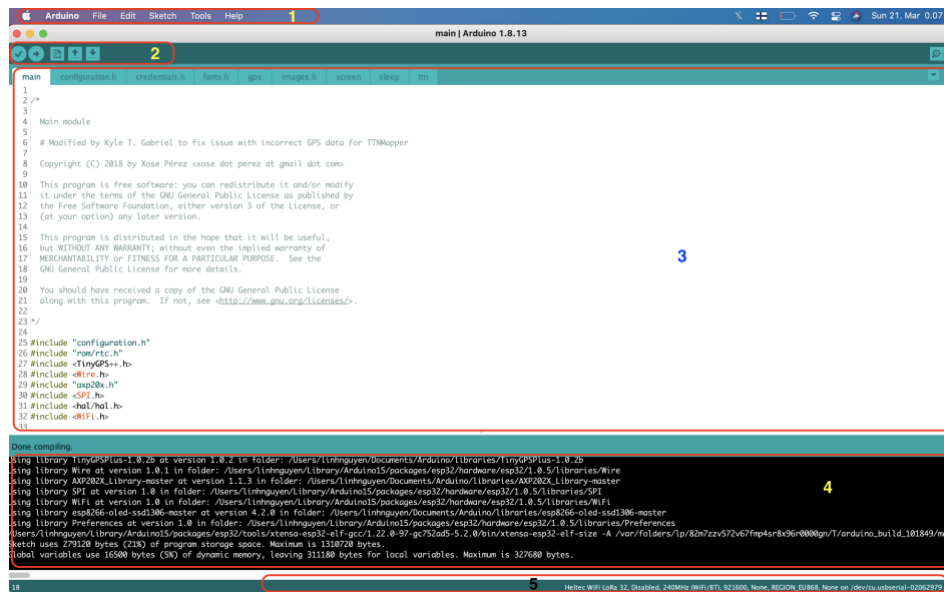


Figure 55. Arduino IDE working platform

Labeled numbers in Figure 56 are:

1. Menu bar: Access to editing, settings, and different functionalities
2. Toolbar: Quick access to the most frequent commands
3. Editor environment: Coding area
4. Status console: Running status and errors messages
5. Status bar: Displaying the configured information of the board with the port, on Mac OS X, it is a serial port, on Windows, it is a COM port.

The configure the T-Beam board takes place as follows:

Connect pin LoRa1 to pin 33 on the board so that the ESP32 module can receive and process data from LoRa module.

Connect the VCC to 3.3V pin and GND, SCL, SDA of the Si7021 sensor to the corresponding pin GND, 21, 22 respectively in the T-Beam board.

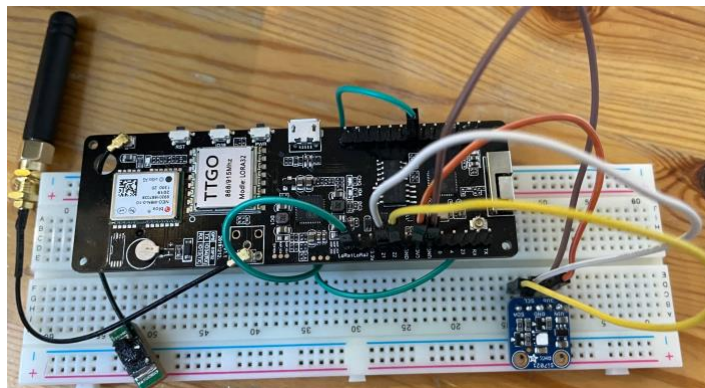


Figure 56. TTGO and sensor wiring

Use a micro-SB cable to connect the T-Beam board to the computer and get back to the Arduino IDE to configure the board.

1. Click Tools, choose Board → ESP32 Arduino → choose either Heltec WiFi LoRa 32 or T-Beam

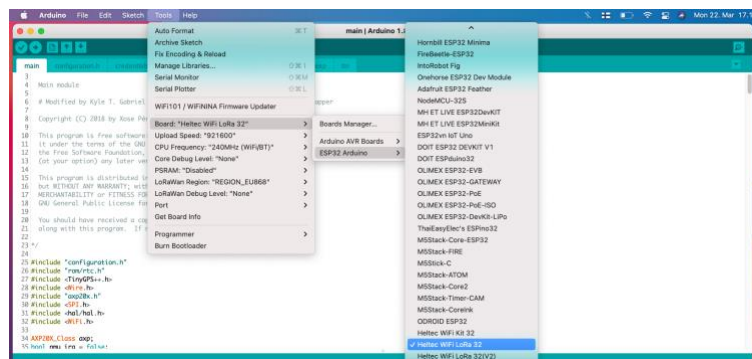


Figure 57. Selecting the board

2. Choose the correct frequency for the board: REGION_EU868

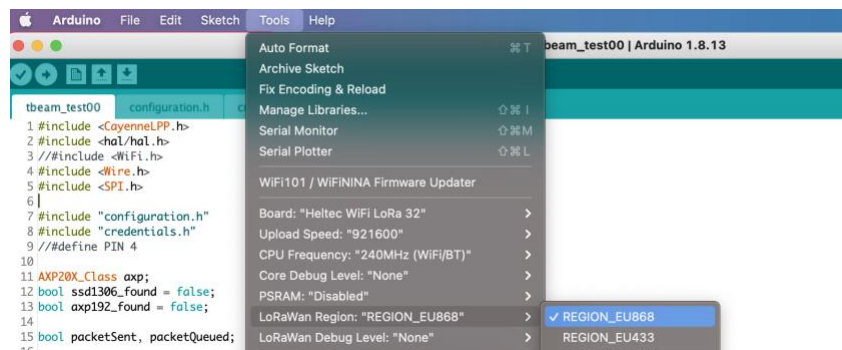


Figure 58. Set the frequency plan

3. Choose the serial port as in Figure 50 below for the connection

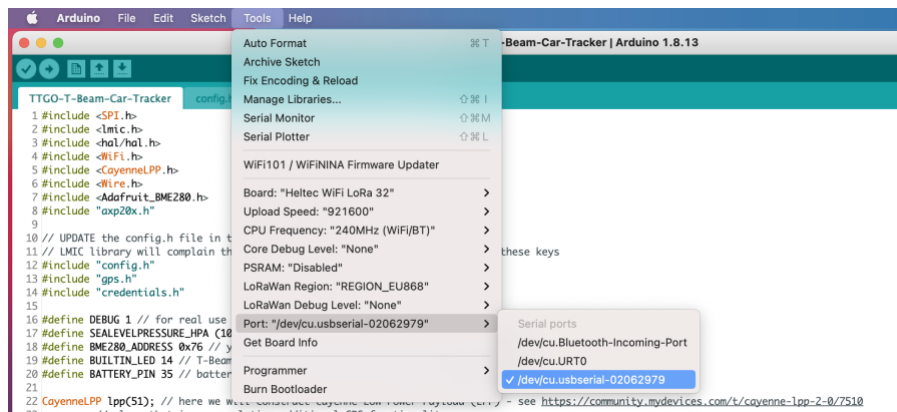


Figure 59. Choose the serial port

A table of necessary libraries is shown below.

Table 5. Arduino libraries

Libraries	Usage	Source
ESP32 Core for Arduino	To program ESP32 board	https://github.com/espressif/arduino-esp32#installation-instructions
TinyGPS++	Read and parse the NMEA sentences to get the GPS data	http://arduiniana.org/libraries/tinygpsplus/
CayenneLPP	Integrate the LoRa node to the Cayenne platform using Low Power Payload	https://github.com/ElectronicCats/CayenneLPP
Arduino-LMIC	To use the SX1272 and SX1276 transceivers modules in the Arduino environment	https://github.com/mcci-catena/arduino-lmic
AXP202X_Library	To power on the GPS chip on board	https://github.com/lewisxhe/AXP202X_Library
Adafruit_Si7021	To work with Si7021 sensor	Install from the IDE's Libraries Manager

There are two methods to install the libraries:

- Manually put the libraries root folder or the downloaded ZIP file to the *libraries* folder (folder *Arduino/libraries* mostly will be created by default when installing the IDE, if not, create a folder named *libraries* inside the folder *Arduino* – a folder where the Arduino sketches are saved).
- Search, install, and update libraries from the IDE's Libraries Manager

Most of these libraries can be installed using either one of the above methods, except for the ESP32 Core for Arduino. Below are the installation steps from the GitHub page of the developer of this library: /19/

1. Open Arduino → Preferences, then insert one of the URLs below into Additional Boards Manager URLs:

- https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
- https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json

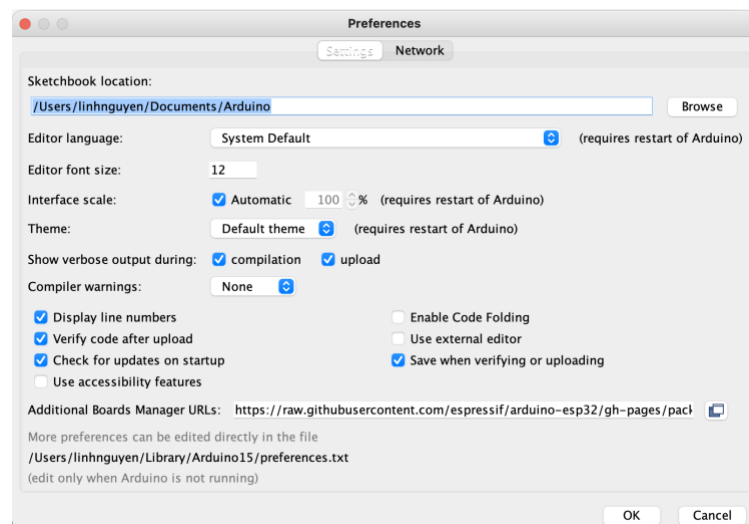


Figure 60. Arduino → Preferences → Additional Boards Manager URLs

2. Open Tools → Board → Boards Manager and search for “esp32” to install

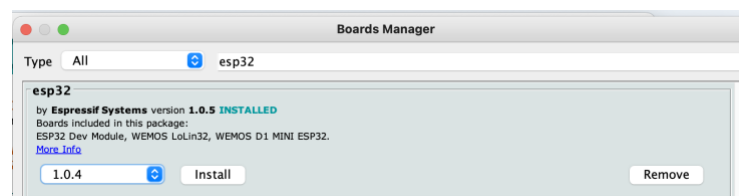
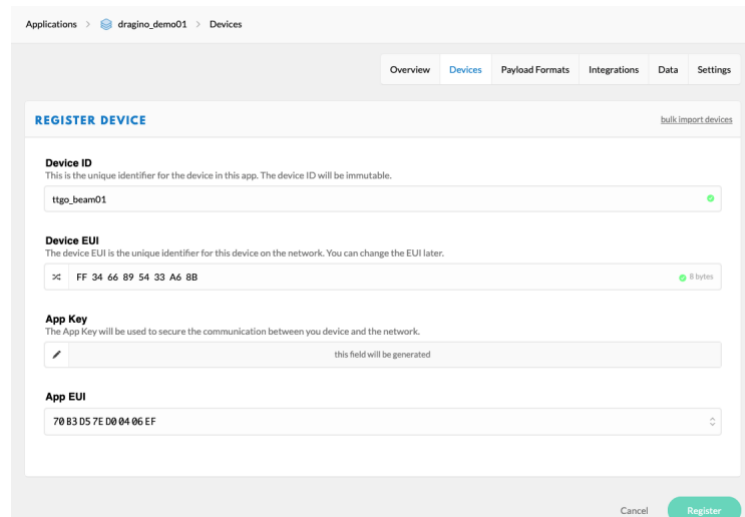


Figure 61. Tools → Boards → Boards Manager

The device needs to be registered in TTN application as follows:

In the Application Overview page, there is Devices section, click onto register device

Enter a Device ID and Device EUI as random or by choices click to Register button.



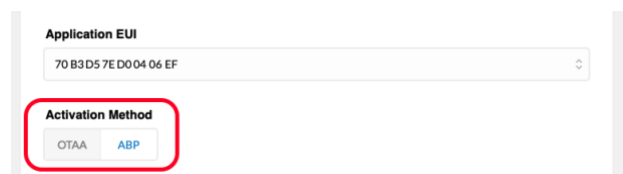
The screenshot shows the 'REGISTER DEVICE' form in the TTN interface. The form is titled 'REGISTER DEVICE' and has a 'bulk import devices' link. It contains the following fields:

- Device ID:** This is the unique identifier for the device in this app. The device ID will be immutable. Value: ttgo_beam01.
- Device EUI:** The device EUI is the unique identifier for this device on the network. You can change the EUI later. Value: FF 34 66 89 54 33 A6 8B.
- App Key:** The App Key will be used to secure the communication between you device and the network. Value: this field will be generated.
- App EUI:** Value: 70 B3 D5 7E D0 04 06 EF.

At the bottom right, there are 'Cancel' and 'Register' buttons.

Figure 62. Register device in TTN

When a new device is registered on TTN, it is set by default using OTAA (Over-the-Air Activation), change it to ABP (Activation by Personalization) by clicking Settings from the Device Overview page, choose ABP in Activation Method and then Save.



The screenshot shows the 'Activation Method' section in the TTN interface. The 'Application EUI' field is visible above, with the value 70 B3 D5 7E D0 04 06 EF. Below it, the 'Activation Method' section shows two options: 'OTAA' and 'ABP'. The 'ABP' option is selected and highlighted by a red box.

Figure 63. Change to ABP method

With the ABP method, in the Device Overview page, there are three essential security keys to put into code to upload to the TTGO board in the next step.

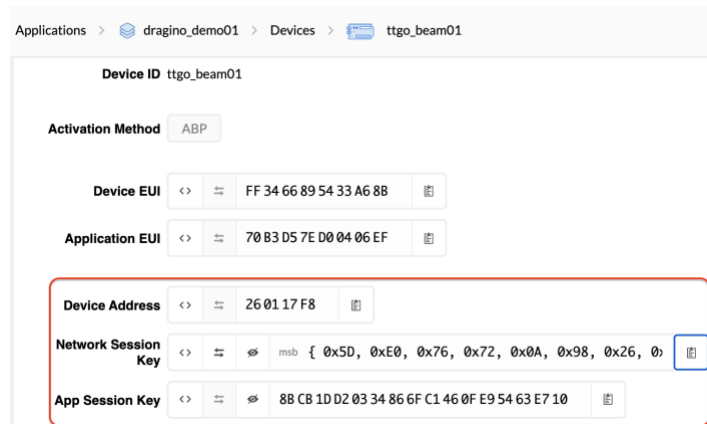


Figure 64. Security keys

5.3.3 Programming the GPS Tracker

The GPS tracker is programmed by opening a new sketch in Arduino, inserting all the necessary libraries, and defining variables for function and function declarations in the beginning of the file.

```

1. #include <Adafruit_Si7021.h>
2. #include <lmic.h>
3. #include <hal/hal.h>
4. #include <SPI.h>
5. #include <WiFi.h>
6. #include <CayenneLPP.h>
7. #include <Wire.h>
8. #include <TinyGPS++.h>
9.
10. #include <SparkFun_u-blox_GNSS_Arduino_Library.h>
11. #include <u-blox_config_keys.h>
12. #include <u-blox_structs.h>
13. #include "axp20x.h"
14. #include "credentials.h"
15.
16. #define I2C_SDA      21
17. #define I2C_SCL      22
18. #define GPS_RX_PIN   34
19. #define GPS_TX_PIN   12
20. #define NSS_GPIO     18
21. #define RESET_GPIO   14
22.
23. #define DIO0_GPIO    26
24. #define DIO1_GPIO    33
25. #define DIO2_GPIO    32
26.
27. #define BATTERY_PIN  35
28. // #define SEALEVELPRESSURE_HPA (1013.25) // this should be set according to the weather
    forecast
29.
30. AXP20X_Class axp;
31. CayenneLPP lpp(51); // here we will construct Cayenne Low Power Payload (LPP) - see
    https://community.mydevices.com/t/cayenne-lpp-2-0/7510

```

```

32. HardwareSerial GPS(1);
33. SFE_UBLOX_GNSS myGPS;
34. TinyGPSPlus tGps;
35. Adafruit_Si7021 sensor = Adafruit_Si7021();
36.
37. double latitude, longitude, altitude;
38. int sats; // GPS satellite coun
39. char s[32]; // used to sprintf for Serial output
40. float vBat; // battery voltage
41. //long nextPacketTime;
42. int state = 0; // steps through auto-baud, reset, etc states
43. float tmp, hum; // Si7021 data are saved here
44.
45. // These callbacks are only used in over-the-air activation, so they are
46. // left empty here (we cannot leave them out completely unless
47. // DISABLE_JOIN is set in config.h, otherwise the linker will complain).
48. void os_getArtEui (u1_t* buf) { }
49. void os_getDevEui (u1_t* buf) { }
50. void os_getDevKey (u1_t* buf) { }
51.
52. static osjob_t sendjob; // callback to LoRa send packet
53. void do_send(osjob_t* j);
54.
55. // Schedule TX every this many seconds (might become longer due to duty
56. // cycle limitations).
57. const unsigned TX_INTERVAL = 60;
58.
59. const unsigned int GPS_FIX_RETRY_DELAY = 10; // wait this many seconds when no GPS fix is
received to retry
60. const unsigned int SHORT_TX_INTERVAL = 20; // when driving, send packets every
SHORT_TX_INTERVAL seconds
61. const double MOVING_KMPH = 10.0; // if speed in km/h is higher than MOVING_HMPH, we assume
that car is moving
62.
63.

```

Set the GPIO pin mapping according to the pin map of the board

```

1. // Pin mapping
2. const lm32_pinmap lm32_pins = {
3.     .nss = NSS_GPIO,
4.     .rxtx = LM32_UNUSED_PIN,
5.     .rst = RESET_GPIO,
6.     .dio = {DIO0_GPIO, DIO1_GPIO, DIO2_GPIO},
7. };

```

Copy Network Session Key and paste to variable NWKSKEY, doing the same for App Session Key to variable APPSKEY and Device Address to DEVADDR.

```

1. static PROGMEM u1_t NWKSKEY[16] = { 0x5D, 0xE0, 0x76, 0x72, 0x0A, 0x98, 0x26, 0xBD, 0xF2,
0x96, 0x94, 0x5C, 0xF6, 0xF6, 0x26, 0x61 }; // LoRaWAN NwksKey, network session key
2. static u1_t PROGMEM APPSKEY[16] = { 0x8B, 0xCB, 0x1D, 0xD2, 0x03, 0x34, 0x86, 0x6F, 0xC1,
0x46, 0x0F, 0xE9, 0x54, 0x63, 0xE7, 0x10 }; // LoRaWAN AppKey, application session key
3. static const u4_t DEVADDR = 0x260117F8 ; // LoRaWAN end-device address (DevAddr)

```

The transmission interval is 20 seconds by default, it can be changed by modifying the value of TX_INTERVAL

```
1. // Schedule TX every this many seconds (might become longer due to duty cycle limitations).
2. const unsigned TX_INTERVAL = 60;
```

From the new version of T-beam T22 V1.1, to get the GPS signal, the GPS chip needs to be powered on by the following code from axp192 library

```
1. void axp192Init() {
2.     Wire.begin(I2C_SDA, I2C_SCL);
3.     if (!axp.begin(Wire, AXP192_SLAVE_ADDRESS)) {
4.         Serial.println("AXP192 Begin PASS");
5.     } else
6.     {
7.         Serial.println("AXP192 Begin FAIL");
8.     }
9.     axp.setPowerOutPut (AXP192_LDO2, AXP202_ON); // LORA radio
10.    axp.setPowerOutPut (AXP192_LDO3, AXP202_ON); // GPS main power
11.    axp.setPowerOutPut (AXP192_DCDC2, AXP202_ON);
12.    axp.setPowerOutPut (AXP192_EXTEN, AXP202_ON);
13.    axp.setPowerOutPut (AXP192_DCDC1, AXP202_ON);
14.    axp.setDCDC1Voltage (3300); // for the OLED power or Si7021 sensor
15.    delay(100);
16. }
```

Sometimes the GPS chip does not get the expected signal or it can take too long to get any signal from the satellite. The following function is applied to reset GPS chip to its original setting of Ublox manufacture and solve this problem.

```
1. void GPS_loop()
2. {
3.     Serial.print("==== STATE ");
4.     Serial.println(state);
5.     switch (state) {
6.     case 0: // auto-baud connection, then switch to 38400 and save config
7.         do {
8.             Serial.println("GPS: trying 38400 baud");
9.             GPS.begin(38400, SERIAL_8N1, GPS_RX_PIN, GPS_TX_PIN);
10.            if (myGPS.begin(GPS)) break;
11.
12.            delay(100);
13.            Serial.println("GPS: trying 9600 baud");
14.            GPS.begin(9600, SERIAL_8N1, GPS_RX_PIN, GPS_TX_PIN);
15.            if (myGPS.begin(GPS)) {
16.                Serial.println("GPS: connected at 9600 baud, switching to 38400");
17.                myGPS.setSerialRate(38400);
18.                delay(100);
19.            } else {
20.                delay(2000); //Wait a bit before trying again to limit the Serial output
                flood
21.            }
22.        }
23.    }
```

```

22.     } while(1);
23.     myGPS.setUART1Output(COM_TYPE_UBX); //Set the UART port to output UBX only
24.     myGPS.saveConfiguration(); //Save the current settings to flash and BBR
25.     Serial.println("GPS serial connected, saved config");
26.     state++;
27.     break;
28.
29.     case 1: // hardReset, expect to see GPS back at 38400 baud
30.         Serial.println("Issuing hardReset (cold start)");
31.         myGPS.hardReset();
32.         delay(3000);
33.         GPS.begin(38400, SERIAL_8N1, GPS_RX_PIN, GPS_TX_PIN);
34.         if (myGPS.begin(GPS)) {
35.             Serial.println("Success.");
36.             state++;
37.         } else {
38.             Serial.println("*** GPS did not respond at 38400 baud, starting over.");
39.             state = 0;
40.         }
41.         break;
42.
43.     case 2: // factoryReset, expect to see GPS back at 9600 baud
44.         Serial.println("Issuing factoryReset");
45.         myGPS.factoryReset();
46.         delay(3000); // takes more than one second... a loop to resync would be best
47.         GPS.begin(9600, SERIAL_8N1, GPS_RX_PIN, GPS_TX_PIN);
48.         if (myGPS.begin(GPS)) {
49.             Serial.println("Success.");
50.             state++;
51.         } else {
52.             Serial.println("*** GPS did not come back at 9600 baud, starting over.");
53.             state = 0;
54.         }
55.         break;
56.
57.     case 3: // print version info
58.         Serial.print("GPS protocol version: ");
59.         Serial.print(myGPS.getProtocolVersionHigh());
60.         Serial.print('.');
61.         Serial.println(myGPS.getProtocolVersionLow());
62.         Serial.println();
63.     }
64.     delay(3000);
65. }

```

Using separated functions to acquire information and later these data are forwarded and displayed on the Cayenne dashboard.

Battery voltage function

```

1. void getBatteryVoltage ()
2. {
3.     // we've set 10-bit ADC resolution 2^10=1024 and voltage divider makes it half of maximum
   readable value (which is 3.3V)
4.     vBat = analogRead(BATTERY_PIN) * 2.0 * (3.3 / 1024.0);
5.     Serial.print("Battery voltage: ");
6.     Serial.print(vBat);

```

```
7.     Serial.println("V");
8. }
```

Temperature and humidity function

```
1. void getSi7021Values()
2. {
3.     hum = sensor.readHumidity();
4.     tmp = sensor.readTemperature();
5.     Serial.print("Humidity:   ");
6.     Serial.print(hum, 2);
7.     Serial.print("\tTemperature: ");
8.     Serial.println(tmp, 2);
9.     delay(100);
10. }
```

There are separate GPS functions to acquire the latitude, longitude, altitude and satellites. The encode () function is to make sure that the data is correctly filtered when the board receives a valid signal

```
11.
12. void gps_encode(){
13.     int data;
14.     int previousMillis = millis();
15.
16.     while((previousMillis + 1000) > millis())
17.     {
18.         while (GPS.available() )
19.         {
20.             char data = GPS.read();
21.             tGps.encode(data);
22.         }
23.     }
24. }
25.
26. double gps_latitude() {
27.     double lat_ublox = myGPS.getLatitude();
28.     lat_ublox = lat_ublox / 10000000.;
29.     double lat = tGps.location.lat();
30.     sprintf(s, "Lat: %f - Lat_Ublox: %f", lat, lat_ublox);
31.     Serial.println(s);
32.     return lat;
33. }
34.
35. double gps_longitude() {
36.     double lon_ublox = myGPS.getLongitude();
37.     lon_ublox = lon_ublox / 10000000.;
38.     double lon = tGps.location.lng();
39.     sprintf(s, "Long: %f - Long_Ublox: %f", lon, lon_ublox);
40.     Serial.println(s);
41.     return lon;
42. }
43.
44. double gps_altitude() {
45.     double alt_ublox = myGPS.getAltitude();
```

```

46.     alt_ublox = alt_ublox / 1000.;
47.     double alt = tGps.altitude.meters();
48.     sprintf(s, "Alt: %f meters - Alt_Ublox: %f meters", alt, alt_ublox);
49.     Serial.println(s);
50.     return alt;
51. }
52.
53. int gps_sats() {
54.     int sats = tGps.satellites.value();
55.     sprintf(s, "Sats: %d", sats);
56.     Serial.println(s);
57.     byte SIV = myGPS.getSIV(); //Satellite-in-View
58.     Serial.print(F(" SiV: "));
59.     Serial.print(SIV);
60.     return sats;
61. }

```

Two main functions needed to make the device join LoRa network.

onEvent() function handles the joining process. This function is pre code snippet and it can be applied to different boards.

```

1. void onEvent (ev_t ev) {
2.     Serial.print(os_getTime());
3.     Serial.print(": ");
4.     switch(ev) {
5.         case EV_SCAN_TIMEOUT:
6.             Serial.println(F("EV_SCAN_TIMEOUT"));
7.             break;
8.         case EV_BEACON_FOUND:
9.             Serial.println(F("EV_BEACON_FOUND"));
10.            break;
11.         case EV_BEACON_MISSED:
12.            Serial.println(F("EV_BEACON_MISSED"));
13.            break;
14.         case EV_BEACON_TRACKED:
15.            Serial.println(F("EV_BEACON_TRACKED"));
16.            break;
17.         case EV_JOINING:
18.            Serial.println(F("EV_JOINING"));
19.            break;
20.         case EV_JOINED:
21.            Serial.println(F("EV_JOINED"));
22.            break;
23.         case EV_RFU1:
24.            Serial.println(F("EV_RFU1"));
25.            break;
26.         case EV_JOIN_FAILED:
27.            Serial.println(F("EV_JOIN_FAILED"));
28.            break;
29.         case EV_REJOIN_FAILED:
30.            Serial.println(F("EV_REJOIN_FAILED"));
31.            break;
32.         case EV_TXCOMPLETE:
33.            Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
34.            digitalWrite(BUILTIN_LED, LOW);
35.            if (LMIC.txrxFlags & TXRX_ACK)

```

```

36.         Serial.println(F("Received ack"));
37.     if (LMIC.dataLen) {
38.         Serial.println(F("Received "));
39.         Serial.println(LMIC.dataLen);
40.         Serial.println(F(" bytes of payload"));
41.         int y = LMIC.dataLen;
42.         sprintf(s, "RSSI %d SNR %.1d", LMIC.rssi, LMIC.snr);
43.         Serial.println(s);
44.     }
45.     os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL), do_send);
46.     Serial.println(F("-----"));
47.     break;
48. case EV_LOST_TSYNC:
49.     Serial.println(F("EV_LOST_TSYNC"));
50.     break;
51. case EV_RESET:
52.     Serial.println(F("EV_RESET"));
53.     break;
54. case EV_RXCOMPLETE:
55.     // data received in ping slot
56.     Serial.println(F("EV_RXCOMPLETE"));
57.     break;
58. case EV_LINK_DEAD:
59.     Serial.println(F("EV_LINK_DEAD"));
60.     break;
61. case EV_LINK_ALIVE:
62.     Serial.println(F("EV_LINK_ALIVE"));
63.     break;
64. default:
65.     Serial.println(F("Unknown event"));
66.     break;
67. }
68. }
69.

```

do_send() function transmits the data to TTN and Cayenne service after calling all the required functions for GPS signal and sensor data.

```

1. void do_send(osjob_t* j){
2.     getBatteryVoltage();
3.     getSi7021Values();
4.     GPS_loop();
5.     // Check if there is not a current TX/RX job running
6.     if (LMIC.opmode & OP_TXRXPEND)
7.     {
8.         Serial.println(F("OP_TXRXPEND, not sending"));
9.     }
10.    else
11.    {
12.        gps_encode();
13.        latitude = gps_latitude();
14.        longitude = gps_longitude();
15.        altitude = gps_altitude();
16.        sats = gps_sats();
17.
18.        lpp.reset();
19.        lpp.addGPS(1, latitude, longitude, altitude);

```

```

20.     lpp.addTemperature(2, tmp);
21.     lpp.addRelativeHumidity(3, hum);
22.     //lpp.addBarometricPressure(4, pressure);
23.     lpp.addAnalogInput(5, vBat);
24.     lpp.addAnalogInput(7, sats);
25.
26.     // Prepare upstream data transmission at the next possible time.
27.     LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
28.
29.     Serial.print(lpp.getSize());
30.     Serial.println(F(" bytes long LPP packet queued.));
31.     Serial.println("-----");
32. }
33. // try again in a few 'GPS_FIX_RETRY_DELAY' seconds...
34.     os_setTimedCallback(&sendjob, os_getTime() + sec2osticks(GPS_FIX_RETRY_DELAY),
do_send);
35. }

```

Two main functions which are always needed to control the board in Arduino IDE are `setup()` and `loop()` functions.

Inside `setup()` function, values and necessary functions are called to initiate value and turn on the appropriate functionalities

```

1. void setup() {
2.     Serial.begin(115200);
3.     Serial.println(F("LoRa based GPS tracker"));
4.
5.     axp192Init();
6.
7.     // set battery measurement pin
8.     adcAttachPin(BATTERY_PIN);
9.     analogReadResolution(10);
10.
11.    //Si7021 sensor
12.    if (!sensor.begin())
13.    {
14.        Serial.println("Did not find Si7021 sensor!");
15.    }
16.
17.    Serial.print("Found model ");
18.    switch(sensor.getModel()) {
19.        case SI_Engineering_Samples:
20.            Serial.print("SI engineering samples"); break;
21.        case SI_7013:
22.            Serial.print("Si7013"); break;
23.        case SI_7020:
24.            Serial.print("Si7020"); break;
25.        case SI_7021:
26.            Serial.print("Si7021"); break;
27.        case SI_UNKNOWN:
28.            default:
29.                Serial.print("Unknown");
30.    }
31.    Serial.print(" Rev(");
32.    Serial.print(sensor.getRevision());

```



```

33.     Serial.print("");
34.     Serial.print(" Serial #"); Serial.print(sensor.sernum_a, HEX);
        Serial.println(sensor.sernum_b, HEX);
35.
36.     //Turn off WiFi and Bluetooth
37.     WiFi.mode(WIFI_OFF);
38.     btStop();

```

Every board needs a pre code snippet to setup the frequency and establish the connection to LoRa network. The codes used in this thesis were put inside setup() function

```

39.     // LMIC init
40.     os_init();
41.     // Reset the MAC state. Session and pending data transfers will be discarded.
42.     LMIC_reset();
43.     // Set static session parameters. Instead of dynamically establishing a session
44.     // by joining the network, precomputed session parameters are provided.
45.     LMIC_setSession (0x1, DEVADDR, NWKKEY, APPSKEY);
46.
47.     #if defined(CFG_eu868)
48.     // Set up the channels used by the Things Network, which corresponds
49.     // to the defaults of most gateways. Without this, only three base
50.     // channels from the LoRaWAN specification are used, which certainly
51.     // works, so it is good for debugging, but can overload those
52.     // frequencies, so be sure to configure the full frequency range of
53.     // your network here (unless your network autoconfigures them).
54.     // Setting up channels should happen after LMIC_setSession, as that
55.     // configures the minimal channel set.
56.     // NA-US channels 0-71 are configured automatically
57.     LMIC_setupChannel(0, 868100000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
58.     LMIC_setupChannel(1, 868300000, DR_RANGE_MAP(DR_SF12, DR_SF7B), BAND_CENTI); // g-
band
59.     LMIC_setupChannel(2, 868500000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
60.     LMIC_setupChannel(3, 867100000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
61.     LMIC_setupChannel(4, 867300000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
62.     LMIC_setupChannel(5, 867500000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
63.     LMIC_setupChannel(6, 867700000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
64.     LMIC_setupChannel(7, 867900000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI); // g-
band
65.     LMIC_setupChannel(8, 868800000, DR_RANGE_MAP(DR_FSK, DR_FSK), BAND_MILLI); // g2-
band
66.
67.     LMIC_disableChannel(1);
68.     LMIC_disableChannel(2);
69.     // TTN defines an additional channel at 869.525Mhz using SF9 for class B
70.     // devices' ping slots. LMIC does not have an easy way to define set this
71.     // frequency and support for class B is spotty and untested, so this
72.     // frequency is not configured here.
73.     #elif defined(CFG_us915)
74.     // NA-US channels 0-71 are configured automatically

```

```

75. // but only one group of 8 should (a subband) should be active
76. // TTN recommends the second sub band, 1 in a zero based count.
77. // https://github.com/TheThingsNetwork/gateway-conf/blob/master/US-global_conf.json
78. LMIC_selectSubBand(1);
79. #endif
80.
81. LMIC_setLinkCheckMode(0);
82.
83. // TTN uses SF9 for its RX2 window.
84. LMIC.dn2Dr = DR_SF9;
85.
86. // Set data rate and transmit power for uplink (note: txpow seems to be ignored by the
library)
87. LMIC_setDrTxpow(DR_SF7,14);
88.
89. Serial.println(F("Ready to track"));
90. Serial.println("-----");
91.
92. // Start job
93. do_send(&sendjob);
94. }
95.
96. void loop() {
97.   os_runloop_once();
98. }

```

Click Verify and Upload the code to the board. The figure below shows the result in the terminal when uploading to the board is successful.

```

Done uploading.
Writing at 0x00060000... (73 %)
Writing at 0x00060000... (76 %)
Writing at 0x0006c000... (80 %)
Writing at 0x00070000... (83 %)
Writing at 0x00074000... (86 %)
Writing at 0x00078000... (89 %)
Writing at 0x0007c000... (93 %)
Writing at 0x00080000... (96 %)
Writing at 0x00084000... (100 %)
Wrote 789136 bytes (489804 compressed) at 0x00010000 in 7.8 seconds (effective 813.6 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1998.0 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...

```

Figure 65. Result of successfully uploading code to the board

5.4 Testing Result

After compiling and uploading the code to the TTGO board, the Serial Monitor was opened while the board was still connected to the computer to check the result.

Result: It showed the notification of enabling and resetting the GPS chip successfully in each boot time. Depending on several factors such as where the board is placed, the quality of the GPS chip, it took some time to catch the GPS signal, it could be a few minutes or even hours to lock the GPS signal. Using different libraries for the GPS chip might affect the speed of getting GPS location as well. The result in serial monitor is always available when the program works properly, it is not affected by the result on TTN's console.

```
/dev/cu.usbserial-02062979
23:31:57.318 -> {}LoRa & GSM based TTN car tracker
23:31:57.414 -> I2C device Found at address 0x34 !
23:31:57.414 -> axp192 PMU Found
23:31:57.414 -> done
23:31:57.414 ->
23:31:57.414 -> AXP192 Begin PASS
23:31:57.414 -> DCDC1: ENABLE
23:31:57.414 -> DCDC2: ENABLE
23:31:57.414 -> LDO2: ENABLE
23:31:57.414 -> LDO3: ENABLE
23:31:57.414 -> DCDC3: ENABLE
23:31:57.414 -> Exten: ENABLE
-----
23:31:57.448 -> DCDC1: ENABLE
23:31:57.448 -> DCDC2: ENABLE
23:31:57.448 -> LDO2: ENABLE
23:31:57.448 -> LDO3: ENABLE
23:31:57.448 -> DCDC3: ENABLE
23:31:57.448 -> Exten: ENABLE
23:31:57.448 -> Initializing LoRa module
23:31:57.448 -> Ready to track
23:31:57.448 -> Battery voltage: 6.59V
23:31:58.459 -> Lat: 0.000000
23:31:58.459 -> Lng: 0.000000
23:31:58.459 -> Alt: 0.000000 meters
23:31:58.459 -> Speed: 0.000000 km/h
23:31:58.459 -> Sats: 0
23:31:58.459 -> Unknown event
23:31:58.459 -> 30 bytes long LPP packet queued.
23:32:00.572 -> EV_TXCOMPLETE (includes waiting for RX windows)
23:32:00.572 -> Next LoRa packet scheduled in 480 seconds!
-----
```

Figure 66. Serial monitor result

To test the Gateway traffic in TTN, the following steps were taken:

Going back to the TTN's console, open Application Overview → Devices → Data, and from Gateway Overview, click Traffic to see if any data going through the gateway to the server on TTN.

Result: Not all of the messages went completely through the gateway. In order to make a successful transmission, the TTGO board needed to be in the range of the gateway and not too close, for example, next to the gateway. It was about 10% to 15% that the messages went through the gateway, then dropped out due to the network connection from the gateway to the Internet. All the data were encrypted for security, from the TTN's application console, the data could be seen as already decrypted by TTN and be represented in hexadecimal, and to decode the message into a readable format, payload function is required, in this case, Cayenne LPP was used for the purpose of visualization in Cayenne service. The data could be

displayed on the Cayenne dashboard only if it was transmitted to the application server which showed in the application data tab.

These results are displayed in Figures 67 - 72 below:

Only data traffic without useful information was seen from the gateway traffic.

time	frequency	mod.	CR	data rate	airtime (ms)	cnt	dev addr:	payload size:
10:39:17	868.1	lora	4/5	SF 7 BW 125	87.3	9	26 01 17 F8	43 bytes
10:29:06	868.1	lora	4/5	SF 7 BW 125	87.3	3	26 01 17 F8	43 bytes
10:21:56	868.1	lora	4/5	SF 7 BW 125	87.3	0	26 01 17 F8	43 bytes
10:16:59	868.1	lora	4/5	SF 7 BW 125	87.3	0	26 01 17 F8	43 bytes
09:55:00	868.1	lora	4/5	SF 7 BW 125	87.3	0	26 01 17 F8	43 bytes
02:06:42	868.1	lora	4/5	SF 7 BW 125	87.3	4	26 01 17 F8	43 bytes

Figure 67. Gateway traffic in TTN

The useful information could be only decrypted and seen on the owner’s application tab. Therefore, all the security key parameters needed to be kept secure, otherwise, anyone who has those keys can decrypt the data messages.

time	counter	port	payload:	anal
24:29:08	0	1	retry	01 88 00 00 00 03 4C 72 00 00 00 02 67 00 00 03 68 00 04 73 00 00 05 02 02 93 07 02 00 00
23:23:10	0	1	retry	01 88 00 00 00 00 00 00 00 00 00 00 02 67 00 00 03 68 00 04 73 00 00 05 02 02 93 07 02 00 00
23:19:53	0	1	retry	01 88 00 00 00 00 00 00 00 00 00 00 02 67 00 00 03 68 00 04 73 00 00 05 02 02 93 07 02 00 00

Figure 68. Application Data traffic in TTN

Uplink

Payload

```
01 88 00 00 00 03 4C 72 00 00 00 02 67 00 00 03 68 00 04 73 00 00 05 02 02 93 07 02 00 00
```

Fields

```
{
  "analog_in_5": 6.59,
  "analog_in_7": 0,
  "barometric_pressure_4": 0,
  "gps_1": {
    "altitude": 0,
    "latitude": 0,
    "longitude": 21.6378
  },
  "relative_humidity_3": 0,
  "temperature_2": 0
}
```

Figure 69. Detailed data traffic in TTN 1

```
Metadata
{
  "time": "2021-04-01T20:19:53.436299678Z",
  "frequency": 868.1,
  "modulation": "LORA",
  "data_rate": "SF7BW125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtw_id": "eui-a840411f3e544150",
      "timestamp": 4098358105,
      "time": "2021-04-01T20:19:53.405212Z",
      "channel": 0,
      "rssi": -31,
      "snr": 9,
      "latitude": 22.70518,
      "longitude": 114.24342
    }
  ]
}

Estimated Airtime
71.936 ms
```

Figure 70. Detailed data traffic in TTN 2

GPS position was showed in a map with another sensor data

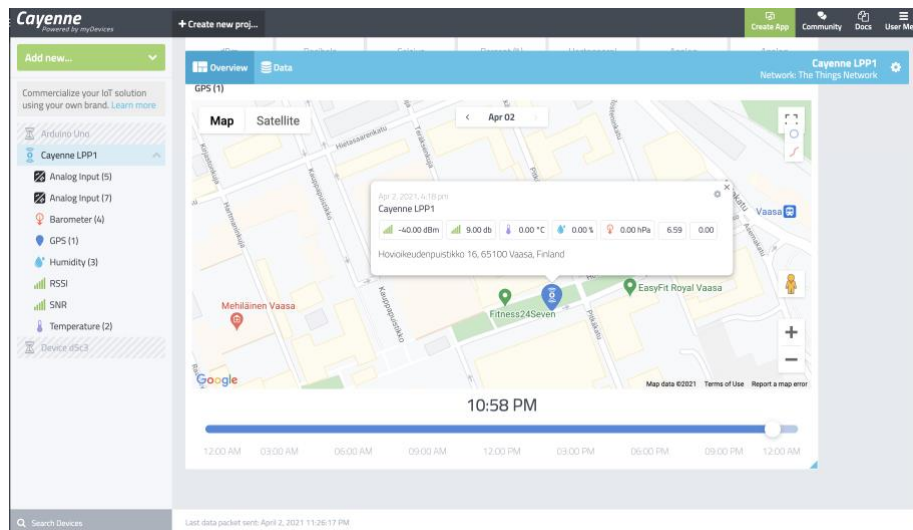


Figure 71. Cayenne visualized tracking web app

Cayenne kept the recording of captured data for downloading

The screenshot shows the Cayenne data recording interface. The table displays the following data:

Timestamp	Device	Channel	Sensor Name	Sensor ID	Data Type	Unit	Values
2021-04-07 4:29:30	Cayenne LPP1	7	Satellite	b046af10-947c-11eb-b767...	analog_sen...	null	
2021-04-07 4:29:30	Cayenne LPP1	2	Temperature (2)	23255da0-9320-11eb-883c...	temp	c	18.299999237061
2021-04-07 4:29:30	Cayenne LPP1	1	GPS (1)	22ff3800-9320-11eb-a2e4...	gps	m	63.0999,0
2021-04-07 4:29:30	Cayenne LPP1	101	SNR	22df53f0-9320-11eb-b767...	snr	db	9
2021-04-07 4:29:30	Cayenne LPP1	100	RSSI	22d20d80-9320-11eb-877...	rssi	dbm	-60
2021-04-07 4:29:30	Cayenne LPP1	5	Battery	238a6100-9320-11eb-a2e...	analog_sen...	null	
2021-04-07 4:29:30	Cayenne LPP1	3	Humidity (3)	233fea80-9320-11eb-8779...	rel_hum	p	34
2021-04-07 4:22:48	Cayenne LPP1	1	GPS (1)	22ff3800-9320-11eb-a2e4...	gps	m	63.0999,0
2021-04-07 4:22:48	Cayenne LPP1	100	RSSI	22d20d80-9320-11eb-877...	rssi	dbm	-59
2021-04-07 4:22:48	Cayenne LPP1	2	Temperature (2)	23255da0-9320-11eb-883c...	temp	c	18.39999961853
2021-04-07 4:22:48	Cayenne LPP1	7	Satellite	b046af10-947c-11eb-b767...	analog_sen...	null	
2021-04-07 4:22:48	Cayenne LPP1	101	SNR	22df53f0-9320-11eb-b767...	snr	db	9
2021-04-07 4:22:48	Cayenne LPP1	5	Battery	238a6100-9320-11eb-a2e...	analog_sen...	null	
2021-04-07 4:22:48	Cayenne LPP1	3	Humidity (3)	233fea80-9320-11eb-8779...	rel_hum	p	34,5
2021-04-07 4:16:06	Cayenne LPP1	7	Satellite	b046af10-947c-11eb-b767...	analog_sen...	null	
2021-04-07 4:16:06	Cayenne LPP1	5	Battery	238a6100-9320-11eb-a2e...	analog_sen...	null	
2021-04-07 4:16:06	Cayenne LPP1	3	Humidity (3)	233fea80-9320-11eb-8779...	rel_hum	p	35
2021-04-07 4:16:06	Cayenne LPP1	1	GPS (1)	22ff3800-9320-11eb-a2e4...	gps	m	63.0999,0
2021-04-07 4:16:06	Cayenne LPP1	100	RSSI	22d20d80-9320-11eb-877...	rssi	dbm	-60
2021-04-07 4:16:06	Cayenne LPP1	2	Temperature (2)	23255da0-9320-11eb-883c...	temp	c	18.10000038147
2021-04-07 4:16:06	Cayenne LPP1	101	SNR	22df53f0-9320-11eb-b767...	snr	db	9
2021-04-07 4:09:24	Cayenne LPP1	3	Humidity (3)	233fea80-9320-11eb-8779...	rel_hum	p	35,5
2021-04-07 4:09:24	Cayenne LPP1	7	Satellite	b046af10-947c-11eb-b767...	analog_sen...	null	

Figure 72. Cayenne's data recording

During the testing time, the gateway also caught data traffic from any LoRa module in range. Without the necessary keys and payload decoder function, the data cannot be decrypted.

The screenshot shows the Gateway Traffic interface. The table displays the following data:

time	frequency	mod.	CR	data rate	airtime (ms)	cnt	dev addr	payload size
13:27:42	868.1	lor	4/5	SF 7 BW 125	82.2	31	26 01 17 F8	39 bytes
13:23:58	868.1	lor	4/5	SF 7 BW 125	118	30765	11 02 1A EF	64 bytes

Figure 73. Data integrity secured through gateway

The board was initially programmed without deep sleep mode to test the power feature.

Result: With a 3.7V Li-ion battery, the TTGO board run over 2 days without any saving power mode. With the deep sleep mode, the board lasted for 4 days. However, this result could not reflect the full picture of the low power feature of LoRa module based on many factors, for example, the board itself had to power

both the GPS chip and Si7021 temperature and humidity sensor, the quality of the board and the GPS chip.

To test how far the board can reach to the gateway, it was placed on a car, a bike, and carry on by walking around Vaasa city.

It was also tested by changing different settings of the gateway, for example, set the SF to a higher level.

Result: Within the urban area of Vaasa, the GPS signal was recorded just about 1 km away from the gateway. This result was reasonable for the position of the indoor gateway and it was placed just about 3 meters above the ground with surrounding residential buildings, other obstacles, and the speed of a moving vehicle. The combination of them increased the loss of signal significantly based on the Fresnel zone rule. The range also got slightly better when the SF was increased; however, SF was not the only reason that could completely make a longer range transmission and high SF was considered to have more disadvantages than advantages.

6 CONCLUSIONS

This thesis was considered as successfully providing a clear understanding knowledge of LoRa/LoRaWAN and how they would change the picture of the IoT in the next decades. With the final implementation of a GPS tracking system, a LoRaWAN was built and tested on basics features as a smart application for daily usage.

The development and implementation phases of the GPS system were successful. In these phases, fundamental operations of LoRaWAN and GPS signal transmission worked properly. GPS signals were sent from the LoRa node to the gateway and the gateway forwarded them to the central server where they were later transferred and visualized on a monitoring web application. The result could be improved by several methods, for example, changing the components with better quality from different vendors and the place to install the gateway to reduce the loss signal within the Fresnel zone.

The initial idea of a GPS system was to migrate into a new radio head to replace the original equipment manufacturer of a car and upgrade the old car into a smart one with a GPS system and more. However, due to the limits of resources, this ambitious step is still on hold and it is also a potential development in the future to provide a reasonable update for self-usage or for a business idea.

REFERENCES

- /1/ Internet of Things: From sensing to doing. Accessed 18.10.2020
https://www2.deloitte.com/us/en/insights/focus/tech-trends/2016/internet-of-things-iot-applications-sensing-to-doing.html?_ga=2.196194788.915445100.1509372760-1419471899.1508865832
- /2/ What is IoT? The internet of things explained. Accessed 20.10.2020
[https://www.networkworld.com/article/3207535/what-is-iot-the-internet-of-things-explained.html#:~:text=The%20internet%20of%20things%20\(IoT\)%20is%20a%20catch%2Dall,data%2C%20receive%20instructions%20or%20both.](https://www.networkworld.com/article/3207535/what-is-iot-the-internet-of-things-explained.html#:~:text=The%20internet%20of%20things%20(IoT)%20is%20a%20catch%2Dall,data%2C%20receive%20instructions%20or%20both.)
- /3/ IoT explained ... in under 100 words. Accessed 20.10.2020
<https://www2.deloitte.com/ch/en/pages/innovation/articles/iot-explained.html>
- /4/ The 2018 SANS Industrial IoT Security Survey: Shaping IoT Security Concerns. Accessed 28.10.2020 <https://forescout-wpengine.netdna-ssl.com/wp-content/uploads/2018/07/2018-SANS-Industrial-IoT-Security-Survey.pdf>
- /5/ What are LoRa and LoRaWAN? Accessed 02.11.2020 <https://lora-developers.semtech.com/library/tech-papers-and-guides/lora-and-lorawan/>
- /6/ LoRa/LoRaWAN tutorial. Accessed 01.12.2020
https://www.mobilefish.com/developer/lorawan/lorawan_quickguide_tutorial.html
- /7/ The Thing Node. Accessed 01.12.2020
<https://www.thethingsnetwork.org/docs/devices/node/index.html>
- /8/ Internet of Things related standards. Accessed 01.12.2020
<https://standards.ieee.org/initiatives/iot/stds.html>
- /9/ Frequencies by country. Accessed 03.12.2020
<https://www.thethingsnetwork.org/docs/lorawan/frequencies-by-country.html>

/10/ Duty Cycle. Accessed 07.12.2020

<https://www.thethingsnetwork.org/docs/lorawan/duty-cycle.html>

/11/ What is duty cycle. Accessed 07.12.2020

<https://www.fluke.com/en/learn/blog/electrical/what-is-duty-cycle#:~:text=Duty%20cycle%20is%20the%20ratio,a%20percentage%20of%20ON%20time>

/12/Digital modulation basics, part 1. Accessed 07.12.2020

<https://www.5gtechnologyworld.com/digital-modulation-basics-part-1/>

/13/ About LoRaWAN. Accessed 15.12.2020 <https://lora-alliance.org/about-lorawan/>

/14/ How spreading factor affects LoRaWAN device's battery life. Accessed 15.12.2020 <https://www.thethingsnetwork.org/article/how-spreading-factor-affects-lorawan-device-battery-life>

/15/ Decibel, dBm, dBi, dBd. Accessed 28.01.2021

https://www.mobilefish.com/download/lora/lora_part5.pdf

/16/ Network Architecture. Accessed 28.01.2021

<https://www.thethingsnetwork.org/docs/network/architecture/index.html>

/17/ LoRa Gateway User Manual. Accessed 01.04.2021

http://www.dragino.com/downloads/downloads/LoRa_Gateway/LG01N/LG01N_LoRa_Gateway_User_Manual_v1.1.pdf

/18/ TTGO T-Beam. Accessed 01.04.2021

<https://www.aliexpress.com/item/32889583204.html?spm=a2g0s.9042311.0.0.27424c4dk0rki3>

/19/ Arduino ESP32 installation instruction. Accessed 01.04.2021

<https://github.com/espressif/arduino-esp32#installation-instructions>