Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Jon Andreas Kornberg

Optimization of LoRaWAN Buoy Controller for Marine Monitoring

Master's thesis in Cybernetics and Robotics
Supervisor: Jo Arve Alfredsen
June 2022

NTNU
Norwegian University of
Science and Technology

Jon Andreas Kornberg

Optimization of LoRaWAN Buoy Controller for Marine Monitoring

Master's thesis in Cybernetics and Robotics
Supervisor: Jo Arve Alfredsen
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

Observing marine phenomena is complex and require advanced technology, but is crucial to monitor marine life and to optimize aquaculture industry. Acoustic telemetry is a technology used to acquire data from subsea to surface, enabling observation of marine life. It consists of transmitting tags implanted in to fish and surface receivers. To optimize the value of telemetry data and distribute it to the user, it should be accessible in real-time. This thesis elaborates the continued work on a telemetry data relaying service called Internet of Fish (IoF). It is a system that consists of surface buoys acquiring acoustic telemetry data from fish below the surface and relaying it to a shore-based station over radio communication above the surface. The station further relays the telemetry data to a cloud service that can store and present it.

Previous IoF studies have enabled fish positioning in aquaculture farm cages using data from multiple acoustic receivers and accurate time synchronization. This thesis pivots the use-case towards single buoys monitoring wild fish in their natural fauna, including optimization of quality of service (QoS) and battery life by minimizing functionality and removing vigorous time synchronization. Here, QoS is a measure of successfully relayed data. This is realized with software redesign, while reusing existing hardware. Moreover, the buoy will be deployed in strategic marine areas where tagged fish are staying or passing, to observe their behavior.

The buoy relaying system is realized with a Synchronization and LoRa Interface Module (SLIM) that consist of a printed circuit board (PCB) inside a waterproof box with external connectors and antennas. With tailored embedded software and low-power long-range radio communication the SLIM provides a QoS of 100 % with a battery life of around 7 months. In this project the system has been deployed with a distance up to 3.2 km between the buoy and the station, but promising results shows that the communication range is not a limit in likely deployment scenarios, if flexible in gateway placement.

Samandrag

Studering av fenomen i sjøen og havet er vanskeleg og krev avansert teknologi. Likevel er det nødvendig for å overvake akvakultur og optimalisere fiskeoppdrett. Akustisk telemetri er ein teknologi som realiserer datainnsamling frå object under vatn. Dette gjer det mogleg å observere liv i havet. Teknologien består av sendarar som er implantert i fisk, og overflatemottakarar. For å optimalisere nytteverdien av telemetri dataen og distribuere den til brukargrensesnitt, må den vere tilgjengeleg i sanntid. Denne avhandlinga utgreier det vidare arbeidet med ei vidarasendingstenesta kalla Internet of Fish (IoF) (Fisk på internett). Det er eit system som består av bøyer som samlar inn fisketelemetri-data med akustiske mottakarar under vassoverflata og sender den vidare til ein landstasjon trådlaust gjennom lufta med radio kommunikasjon. Landstasjonen sender dataen vidare til ei skyteneste som kan lagre og presentere den.

Tidlegare IoF arbeid har lagt til rette for posisjonering av fisk i oppdrettsmerder ved å tidssynkronisere fleire akustiske mottakar og slå saman dataane deira. Denne avhandlinga endrar fokuset mot eit anna bruksområde der enkle bøyer skal kunne overvake fisk i naturlig fauna, herunder optimalisering av kvaliteten og batterilevetida til tenesta ved å minimalisere funksjonaliteten og fjerne unødvendig tidssynkronisering. Kvalitet blir målt på grad av vellukka vidaresendingar. Dette er realisert ved å gjenbruka eksisterande maskinvare, men redesign av programvare. Bøyene vil blir plassert på strategiske lokasjonar i sjøen der merka fisk passerer eller oppheld seg, for å observere dei.

Vidaresendingstenesta er realisert med ein synkronisering- og grensesnittsmodul som er kalla SLIM. Denne består av eit kretskort inni ein vasstett boks med eksterne koblingmoglegheiter. Med skreddarsydd programvare og energieffektiv langdistanse radio kommunikasjon oppnår SLIMen ei kvalitetsgrad på 100 % med ei batterilevetid på omtrent 7 månadar. I løpet av dette prosjektet har tenesta blitt verifisert funksjonell med ein distanse mellom bøye og landstasjon på opp til 3.2 km, men lovande resultat viser at rekkevidda ikkje er ei begrensing i sannsynlege bruksområde, viss ein er fleksibel på lokasjon til landstasjon.

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology and
Electrical Engineering
Department of Engineering Cybernetics

MASTER'S THESIS ASSIGNMENT

Name: Jon Andreas Kornberg
Program: Engineering Cybernetics
Title: Performance evaluation and optimisation of LPWAN acoustic telemetry buoy
Credits: 30 SP

Project description:

This project targets development and performance evaluation of a marine sensor buoy concept with LPWAN connectivity. The buoy is designed for carrying a specific acoustic telemetry receiver and relaying data received underwater from acoustic sensor transmitters to an Internet backend. The objective of the buoy is to enable remote near real-time acquisition of acoustic telemetry and other buoy data, while preserving the favourable properties of currently used off-line stand-alone logging receivers such as ease of deployment, long operational life, and low cost. The buoy should thus be lightweight and allow deployment for several months in a remote fjord or coastal site consuming minimal amounts of power, while still being capable of transferring receiver detections wirelessly to a shore-based gateway/base station in near real time. The goal of this project is to bring the current version of the LoRa-based buoy controller (SLIM – Synchronisation and Lora Interface Module) into a robust operational state, and evaluate, optimise and document its performance through properly designed lab and field tests. The project includes the following tasks:

- Technical survey of LoRa wireless communication and the LoRaWAN network protocol
- Using the existing hardware and firmware implementation of the SLIM buoy controller, develop and implement an optimised solution for the single buoy monitoring scenario:
    - Ultra-low power near real-time remote access to acoustic telemetry data
    - Buoy position and time awareness (no precision acoustic signal timing)
    - Optimisation of operational life and communication range
- Plan and conduct realistic field tests to explore the function and performance of the buoy system
- Discuss field test results and work out detailed documentation and project report

Project start: 24 January 2022
Project due: 20 Juni 2022
Host institution: NTNU, Department of Engineering Cybernetics
Supervisor: Jo Arve Alfredsen, NTNU/DEC

Trondheim, 24 January 2022
Jo Arve Alfredsen

Preface

This master's thesis concludes my degree in Master of Technology at the Cybernetics and Robotics programme at Norwegian University of Science and Technology (NTNU). It covers my work contributing to the Internet of Fish (IoF) project the last year, including the foundation acquired in my specialization project. Working with the IoF project has been very rewarding in terms of knowledge, but also experiences during field work. Being allowed to test my work in real life with fjord deployments has been really motivating.

In conjunction with this I want to thank some important persons who made this possible. First of all, I want to thank my supervisor Jo Arve Alfredsen for great enthusiasm, commitment and knowledge in the field. He has spent a lot of his valuable time with me spreading wisdom and guidance during discussions and field work, and I am grateful. He has also provided all the equipment I needed. Furthermore, I want to thank my fellow student Vetle Berg Abrahamsen for important discussions and hints. He has been working in parallel with me this semester, struggling with similar IoF issues, though on another approach. Then, I want to thank Nikolai Lauvås for help setting up the *otter01* server for me and Vetle. Finally, I want to thank my family and my friends for all the support.

Jon Andreas Kornberg
June 2022

Contents

List of Abbreviations

ADR adaptive data rate. 6, 17, 36, 39, 51, 71, 73, 75

ARM Advanced RISC Machine. 31

ASK Amplitude Shift Keying. 10

ASL above sea level. 28, 49

BURTC backup real time counter. 31, 36, 68

CPU central processing unit. 31

CR coding rate. 17

CSS Chirp Spread Spectrum. 14

DPPM Differential Pulse Position Modulation. 27

DR data rates. 16–18

EIRP effective isotropic radiated power. 20

ETSI European Telecommunications Standards Institute. 11

FCC Federal Communications Commission. 11

FSK Frequency Shift Keying. 10, 14

GNSS Global Navigation Satellite System. xix, 21, 22, 26, 30, 33, 34, 38, 42, 48–50, 63, 67–70

GPIO general purpose input/output. 20, 31, 33–35, 68

GPS Global Positioning System. 21

HAL hardware abstraction layer. 20, 72

HFXO high frequency crystal oscillator. 31

IDE integrated development environment. 45

IoF Internet of Fish. iii, v, ix, xix, 3–5, 26–28, 36, 38, 40, 41, 44, 47–49, 58, 72–75

IoT Internet of Things. 3, 4, 12, 23, 38

IP Internet Protocol. 39

ISM Industrial, Scientific and Medical. 11, 19

ISR interrupt service routine. 36, 37, 62

LCD liquid crystal display. 34, 37, 63, 69

LED light emitting diodes. 34, 37, 63, 69

LEUART low-energy universal asynchronous receiver-transmitter. 34

LFXO low frequency crystal oscillator. 31

LMiC LoRaMAC in C. xix, 20, 21, 33, 36, 38, 62, 70, 72, 75

LoRa Long-Range. xviii, xix, 6, 13–20, 27, 28, 30, 33, 39, 44, 48, 49, 63, 73, 74

LoRaWAN Long Range Wide Area Network. xix, 4–6, 12, 13, 17–20, 27, 30, 33–42, 44, 48, 49, 53, 58, 60, 70–75

M2M machine-to-machine. 23

MCU microcontroller unit. 30, 31, 33–37, 45, 62, 68, 69

MQTT Message Queuing Telemetry Transport. xix, 23, 39, 40

NTNU Norwegian University of Science and Technology. ix

OASIS Organization for the Advancement of Structured Information Standards. 23

OS operating system. 36, 38, 39, 62

PCB printed circuit board. iii, 30, 31, 34, 35, 69

PDOP position dilution of precision. 44, 49

PER packet error rate. 17, 51–53

PPS pulse-per-second. 68

PSK Phase Shift Keying. 10

QoS quality of service. iii, 5, 6, 67, 70, 71, 73, 75

RAM random access memory. 31, 33

RISC Reduced Instruction Set Computer. 31

RSSI received signal strength indicator. xx, 8–10, 39, 52, 53, 58, 60, 73

RTC real-time counter. 33

SDK software development kit. 35

SeqN sequence number. 58, 70, 72

SF spreading factor. 15–19, 51, 58, 71, 73, 74

SLIM Synchronization and LoRa Interface Module. iii, v, xviii, xx, 4, 5, 26–28, 30, 31, 33–37, 41, 44, 45, 48–53, 58, 61–63, 65, 67–72, 74, 75, 81, 88

SNR signal-to-noise ratio. xx, 8, 14, 15, 18, 39, 44, 52, 53, 58, 60, 71, 73

SPI Serial Peripheral Interface. 20, 33, 34, 68

TBR Thelma Biotel Receiver. 26, 28–30, 34, 37, 38, 41, 44, 45, 49, 58, 63, 67, 69, 70, 72

TLS Transport Layer Security. 24, 40

ToA time on air. 17, 20, 74

TTFF time-to-first-fix. 22, 33, 49, 53, 68, 69

UART universal asynchronous receiver-transmitter. 34

USB Universal Serial Bus. 35, 61

List of Tables

List of Figures

Chapter 1

Introduction & Previous Work

1.1 Marine Observations

Monitoring the patterns and behavior of fish in the ocean is essential for understanding aquaculture and fauna. Observation and data acquisition is key to unveil new knowledge and require evolving technology. Scientist have studied the Earth, stars and neighbor galaxies for centuries, using observation technology like satellites, telescopes and sensors. However, marine phenomena are hard to observe due to rough and complex environments. Still, they are important to our lives regarding food, industry and climate.

1.2 Fish Observation

Aquaculture is one important phenomenon regarding food and industry. Specially fish is a major part of our society and have been since day one, but what do we know about their lives under the surface? Evolving technologies within fish observation let us study their behavior and travel patterns. The methods used need to encounter challenges concerning wireless signal propagation, limited space, battery life and more. Another challenge is to develop subsea technology onshore since it will act differently when put under the surface. One strategy is to attach measurement devices to captured fish, releasing them back to their habitat, and hope to capture it again. However, it is not certain the data will ever be retrieved. Another strategy increases the probability to retrieve data by using pop-up satellite archival tags. This is a tag that detach from the fish after some time in sea. It floats to the surface, where satellite signals can be received, and reveal its position and sends telemetry data. These pop-up tags have helped biological scientist to track the journey of Atlantic salmon according to an article from [1].

A third strategy is to use acoustic telemetry tags. These tags use advanced technology regarding low-power and underwater communication. They offer telemetry data acquisition and relaying while under water. The tags send digital data to surface receivers using modulated ultrasonic sound pulses. This enables observation and monitoring of fish in the ocean[2], [3].

1.3 Internet of Fish

Internet of Fish (IoF) is a concept introduced by Hassan, etal[4] in 2019. The name is based on the term Internet of Things (IoT) which describes devices that connects "things" to the Internet. The state-of-the-art acoustic receivers are stand-alone and only collect and save data in memory. Usually the user have to retrieve the receiver to get the data. IoF is a data relaying service for acoustic telemetry receivers that enables real-time observation of fish. Real-time data is favourable for operations requiring responding actions. Data from stand-alone receivers is old and tedious to fetch. The IoF system is also used to time-synchronize multiple acoustic receivers. Data from time-synchronized acoustic receivers can be merged and used in multilateration, enabling positioning of fish. This has been tested by Hassan[5] in a floating fish farm.

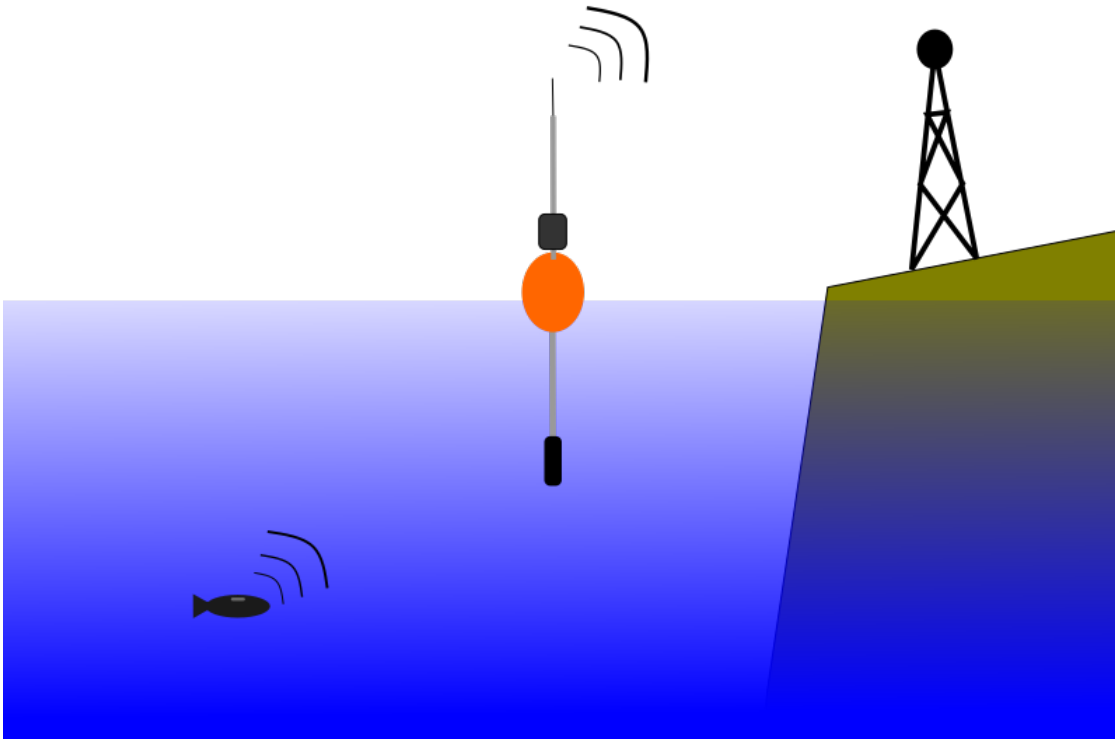


Figure 1.1: Internet of Fish concept sketch.

1.3.1 Data Relaying Service

Data received by acoustic receivers are relayed to a gateway that in turn uploads it to a database over the Internet, seen in Figure 1.1. The central unit in this system is called a Synchronization and LoRa Interface Module (SLIM) that connects acoustic receivers to a gateway using a low-power wireless radio communication protocol called Long Range Wide Area Network (LoRaWAN). The SLIM is battery powered to be versatile and remove demands for power infrastructure. Hence, a low-power design is necessary to increase lifetime and reduce maintenance regarding battery replacements. It should also support long range communication to be able to send data from deployments in sea to shore-based gateways.

1.3.2 Previous IoF Work

In addition to Hassan, several others have also contributed to the project. Kjelsvik[6] has optimized the communication with a new IoF protocol and created a system to save and present the data from a central server. Rundhovde[7] has power optimized the SLIM and created a more sophisticated time-synchronization algorithm. Joelsgaard[8] has designed a new version of SLIM called cSLIM, which should remove hardware issues on the original SLIM and add a NB-IoT modem. NB-IoT (NarrowBand - IoT) is a communication alternative to LoRaWAN. cSLIM is under further development by Abrahamsen in parallel with this project.

1.4 Single Monitoring Buoy

The previous IoF work previously focused on fish observation in floating fish farms, including positioning[5], [7]. In this thesis the use-case is to observe behavior of wild fish in their natural fauna. This can be realized by deploying acoustic receivers and a SLIMs on surface buoys. This use-case has been tested using Rundhovde's[7] software. The results is an unstable service with showers of data and with long periods of time without any data. From earlier tests Rundhovde[7] has experienced problems communicating with the acoustic receiver, apparently

due to heavy time-synchronization work added by the new algorithm.

1.4.1 Scope of This Thesis

The goal of this project has been to develop and implement an optimized solution for a single buoy monitoring scenario. I.e optimizing the SLIM for low power to increase battery life while retaining the basic functionality of relaying data from the acoustic receiver to a shore-based gateway. Moreover, it should provide telemetry data in near real-time and with a high quality of service (QoS) in terms of successfully relayed data. This includes a reliable communication link over an extensive distance and simple time-synchronization of the acoustic receiver. Multilateration of fish will not be possible in a single buoy scenario, hence, offering time awareness is sufficient.

The entire IoF system will be presented, but the focus of this thesis is the SLIM and its communication link with the gateway.

1.4.2 Approach

A thorough survey of the existing solution, previous work and the LoRaWAN communication protocol is necessary to optimize the IoF system for the single monitoring buoy scenario. The existing software and hardware solution used and optimized by Rundhovde[7] is used as a basis for this project. His software is immensely influenced by the time-synchronization algorithm, but serve as a great basis for understanding the SLIM functionality. However, the software modules has been disassembled completely to be understood thoroughly. Then it has been reassembled and rewritten to meet the goals described above. The rest of the IoF system is also implemented from scratch, including data transport, storage and presentation. Furthermore, desk tests, field tests and deployments are conducted to test and verify functionality and performance.

1.4.3 Related Work

A LoRaWAN communication network have been tested in marine environments by Parri[9], etal. They have realized a communication range of 8.33 km, from off-shore breeding cages to shore. Pensieri[10] etal., have stress tested the LoRaWAN communication range, also in a marine environment. They have accomplished a range of over 110 km. Hassan[5] realized LoRaWAN links over distances up to 450 m with a QoS of greater than 98 %, and over a distance of 2.5 km with a QoS of 92.8 %.

LoRaWAN uses a modulation technique called Long-Range (LoRa). One of its advantages is dynamic transmit parameters. LoRaWAN offer an adaptive data rate (ADR) scheme which attempts to choose parameters that optimize power consumption and data rate. However, several enhanced algorithms have been implemented to increase performance in both static and dynamic links. E.g Farhad[11] etal., have implemented two algorithms that one of them reduce convergence period with up to 68 % in a static link. Farhad's results are based on simulation results.

1.5 Outline

This thesis describes and evaluate the work done in this project. In Chapter 2, theory about the low-power communication protocol and other relevant technology is presented. Further in Chapter 3 the implemented system is presented and described. Moreover, its components' properties, functionality, and interaction are described with focus on what makes the system low-power. Chapter 4 describes the conducted deployments and tests in terms of methods and results. The tests include realistic functionality field tests and power consumption tests. Finally, all these aspects are discussed and compared with previous work in Chapter 5.

An earlier specialization project by the author of this thesis is added in Appendix C. However, this thesis is independent, hence, some content and figures are repeated.

Chapter 2

Theory

2.1 Wireless Communication

Wireless communication techniques use electromagnetic waves or sound to transfer information. A wireless network consists of transmitters and receivers. Their benefits are mobility and low demands for wired infrastructure. In addition, transmitters have become so small that they fit into small gadgets like smart watches. Hence, wireless communication has taken over wired communication in many applications. For the rest of this section (2.1) the derivation will focus on wireless communication using electromagnetic waves realized with antennas.

2.1.1 Signal Propagation

Wireless transmitters propagate signals with sinusoidal electromagnetic waves. As opposed to in wired communication, wireless signal power is spread out in the atmosphere and only a tiny part of the signal power is absorbed in the receiving antenna[12]. Hence, one drawback with wireless signals is power efficiency related to range. When the distance increase, the signal power decrease. If signal power is below the ambient noise power it is hard to resolve the signal. The ratio between signal power and noise power is called signal-to-noise ratio (SNR). The signal power is often described with received signal strength indicator (RSSI) denoted in decibel relative to 1 mW (dBm).

There are different types of transmitters with different propagation patterns. Some of them propagate almost isotropic, i.e same amount of power in all directions, while others propagate more directional. Transmitters and receivers are characterized with a gain relative to theoretical isotropic propagation, often denoted in decibel (dBi), with respect to the direction.

Between a transmitting antenna and a receiving antenna a wireless signal encounters several power losses in both supply wires and obstacles. In addition, the signal encounters free space loss. Free space loss is not actually a loss, but it is a representation of the fact the signal propagates isotropically. The free space loss is derived from area of a sphere with radius equal to the distance between the

antennas. It can be calculated from the Friis'[13] transmission equation, see eq 2.1.

$$\begin{aligned} \frac{P_r}{P_t} &= G_t G_r \left(\frac{\lambda}{4\pi d}\right)^2 \\ &= G_t G_r \left(\frac{c}{4\pi df}\right)^2 \end{aligned} \tag{2.1}$$

P_t and P_r are the transmitted and received power, G_t and G_r are the antenna gains, λ is the signal's wavelenght and d is the distance between the antennas. In the second line, wavelength λ is swapped with c/f from the wave speed equation $c = \lambda f$ where c is the speed of light and f is the signal's frequency. Eq 2.1 is based on absolute values, but can also be expressed in logarithmic scales in decibel, see eq 2.2.

$$P_{r[dBm]} = P_{t[dBm]} + G_{t[dBi]} + G_{r[dBi]} + 20\log\left(\frac{c}{4\pi df}\right) \tag{2.2}$$

The derivations above assumes no obstacles or loss in the transmission. Obstacles blocking the direct line of sight between sending and receiving antenna, like walls or mountains, can reduce the RSSI drastically. In addition, due to multipathing the signal also suffer from interference from delayed signal taking other paths. Hence, obstacles outside line of sight can also reduce the RSSI.

When considering the attenuation of the signal due to multipathing The Fresnel Zone can be used[12]. It is a mathematical derivation of where obstacles in the path will interfere the signal. The Fresnel Zone is an ellipsoid around the transmitter and receiver, and its shape is determined by the frequency of the signal and the distance between the antennas. The radius, R_n, of the n-th Fresnel Zone is given in eq 2.3. The Fresnel Zone order, n, is the number of half wave periods a signal is delayed compared to the direct line of sight signal ($n = 0$). Obstacles in the first order zone ($n = 1$) adds reflected signals delayed with a half wave period. These signals attenuate the direct signal due to destructive interference. However,

obstacles in the second order zone ($n = 2$) delays signals with a whole wave period, and increases RSSI due to constructive interference.

$$R_n = \sqrt{\frac{nd_1d_2\lambda}{d_1 + d_2}} \tag{2.3}$$

d_1 and d_2 is the distance to the obstacle, along the direct line of sight, from the transmitter and receiver respectively. In radio communication the first order of The Fresnel Zone is used when considering channel attenuation. Moreover, the largest radius is half way between the transmitter and receiver, and $\lambda = c/f$. Eq2.3 can be simplified to calculate the maximum clearance from the direct line of sight, see eq2.4.

$$R_{1,max} = \frac{1}{2}\sqrt{\frac{cD}{f}} \tag{2.4}$$

c is the speed of light, f is the frequency of the radio signal and D is the total distance between the antennas.

2.1.2 Modulation

To send information over a radio link a modulation technique is needed. A sinusoidal carrier radio wave can be modulated in terms of amplitude, frequency or phase. When the information is digital the radio signal is modulated to represent 1's and 0's or series of them in a symbol. Conventional modulation techniques like Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK) and Phase Shift Keying (PSK) simply alter amplitude, frequency or phase respectively, to represent different symbols. Using more complex modulation techniques can increase receiver sensitivity, i.e at how low RSSI symbols are successfully demodulated.

Spread spectrum modulation is a collection of modulation techniques utilizing the entire allocated frequency band. The transmit power is smeared out over the frequency band, see Figure 2.1. This makes the link less affected by noise, multipathing and other transmissions on the same channel. This is done by adding

redundant information to the transmission. Hence, the information data rate is decreased. This is the price to pay to increase receiver sensitivity.

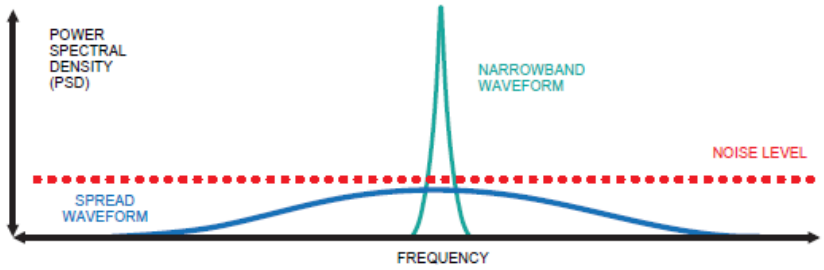


Figure 2.1: Spectrogram of spread spectrum versus narrow spectrum modulation. Figure from [14].

2.1.3 Band Regulations

The electromagnetic frequency specter is divided into radio bands. There are licensed bands where only certain users can utilize and there are unlicensed bands, like the Industrial, Scientific and Medical (ISM) band, which everyone can use for free. Specially the unlicensed band require regulations to prevent overuse. E.g in Europe the European Telecommunications Standards Institute (ETSI) regulates the ISM band, and Federal Communications Commission (FCC) in the United States. In addition there can be national regulations on top of these.

2.2 LoRaWAN

Long Range Wide Area Network (LoRaWAN) is a communication protocol developed by the LoRa Alliance[15], optimized for long range and low power. The protocol is popular among distributed Internet of Things (IoT) devices requiring battery powered operation. LoRaWAN achieves low power by optimizing bandwidth utilization and sacrificing data rate over the communication link. In addition to low power and long range, LoRaWAN implements a data link layer ensuring link reliability.

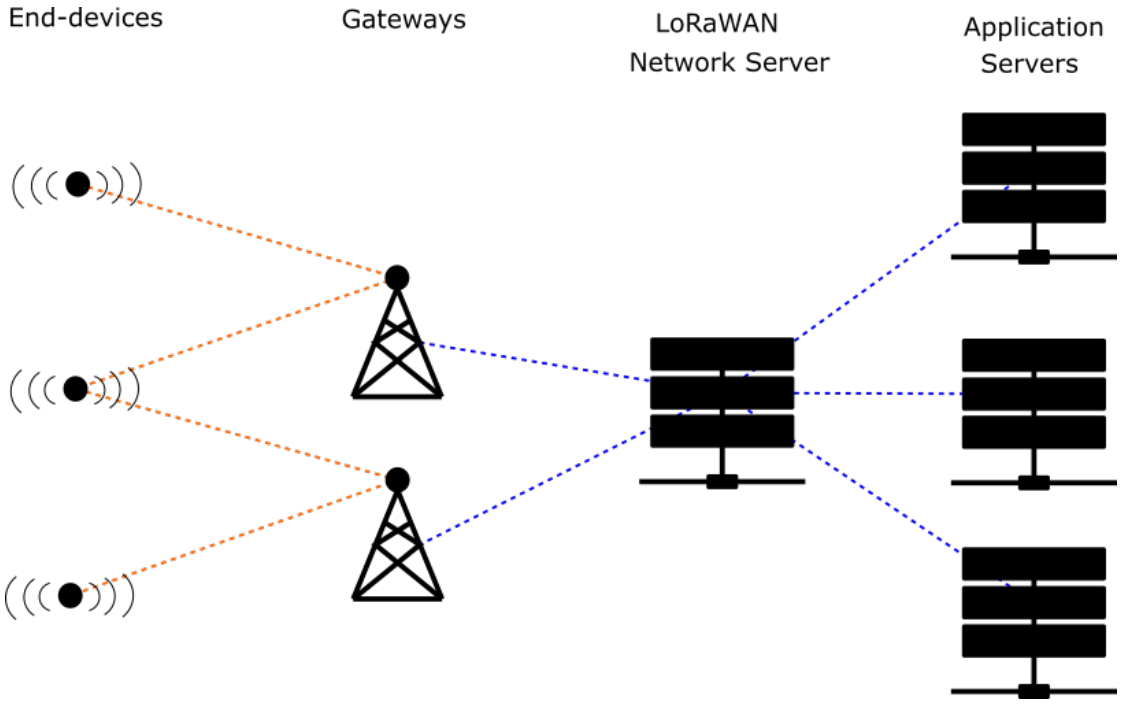


Figure 2.2: LoRaWAN network architecture.

2.2.1 Network Architecture

LoRaWAN is a centralized star of stars network shown in Figure 2.2. Its basic components are end-devices, gateways and a network server. End-devices are battery powered distributed devices that serve a purpose to report information to the user. A typical end-device is an IoT sensor that report sensor data. The

data is broadcasted to gateways within range. The gateways forwards the data to the network server which removes duplicates and logs it. Users can implement applications that receives the data from the network server. This direction of data propagation is called uplink. Furthermore, the user can send downlinks to end-devices. The downlink path is opposite of the uplink path, but the network server chooses the most suitable gateway to forward the data.

End-device and gateways communicate with radio signals and elsewhere the Internet protocol is used. The gateways either use a cellular or an Ethernet backhaul to connect with central network server.

LoRaWAN networks can be implemented and deployed privately, where all the components of the network have to be realized by the user. There are also communities, like The Things Network[16], sharing gateways and network servers.

2.2.2 End-device Classes

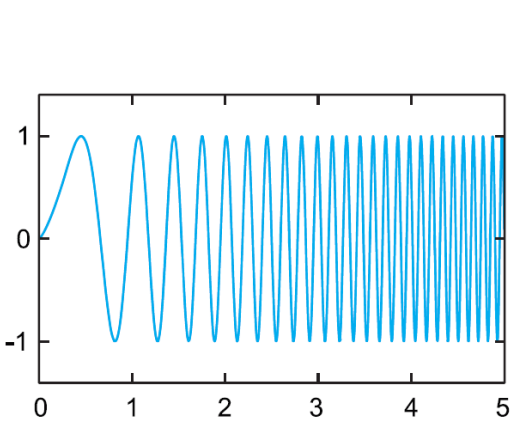
The end-devices in a LoRaWAN network is characterized by classes. The classes differ in the end-device's availability and power consumption. Since turning on the antenna for receiving increases power consumption, it is a trade-off between these. Class A end-devices open two receive windows after an uplink. If the gateway wants to send a downlink, it needs to do it one of these. Hence, all communication is initiated by the end-device. Class B end-devices open the same receive windows as Class A, but also at fixed intervals after them. E.g a Class B end-device can open receive windows every 10 seconds. Class C end-devices open a continuous receive window after the two as in Class A.

2.2.3 LoRa Modulation

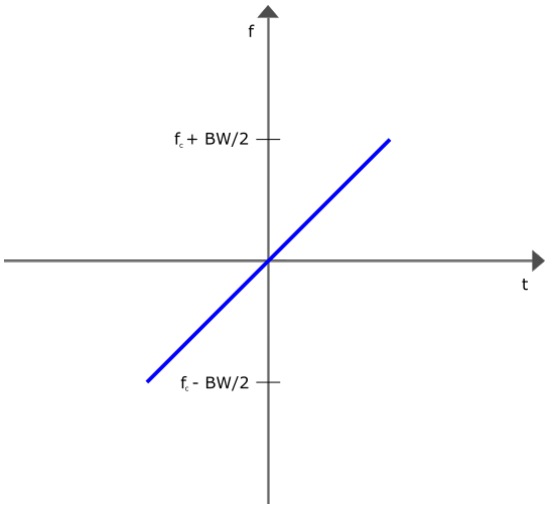
Long-Range (LoRa) is the radio modulation technique used by LoRaWAN. It is a spread spectrum modulation technique by Semtech[17] offering long-range, low data rate radio communication. The modulation is described in one of Semtech's application notes[18] and on their website[14]. With a receiver sensitivity of -137

dBm compared to -122 dBm for FSK, LoRa offer a large link budget. A link budget is the maximum signal attenuation between the transmitter and the receiver, which also correlates with communication range, disregarding transmit power.

LoRa uses Chirp Spread Spectrum (CSS) modulation to spread the signal in the frequency domain. A chirp, shown in Figure 2.3, is a constant power sweep signal with linearly increasing or decreasing frequency over time, within the band. Respectively, there are up-chirps and down-chirps. LoRa uses these chirps and phase shifted versions of them, shown in Figure 2.4 to represent information in symbols. Notice that when an up-chirp reaches the highest frequency of the channel, it continues from the lowest frequency, and vice-versa for a down-chirp.



(a) In time domain[14].



(b) Frequency with respect to time.

Figure 2.3: LoRa up-chirp.

The CSS is resistant against noise, interference from other devices transmitting on the same channel, and jamming. The key is to spread the symbol energy over frequency as well as over time. This adds redundancy to the symbol and it can be demodulated successfully at the receiver. Increasing the symbol time decreases line rate, i.e the nominal bit rate (bit/s). Looking at the Shannon-Hartley theorem (eq and ref) one can see that a channel information rate capacity C is determined by bandwidth B and SNR($\frac{S}{N}$) at the receiver, see eq2.5. The theorem defines the

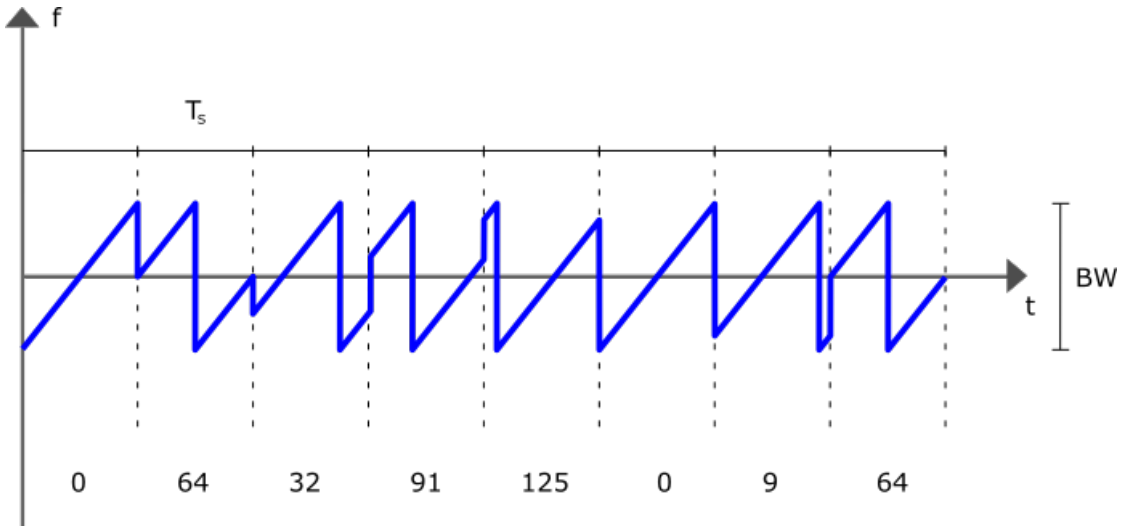


Figure 2.4: Eight example LoRa symbols, each representing a 7-bit (SF=7) number [0 127]. Symbol decimal value is denoted below each symbol and is an approximation for the purpose of illustration.

upper limit of information rate (bit/s) possible at an arbitrary communication link with negligible bit error rate. Information rate is the rate of bits carrying actual data, i.e all bits exclusive redundant bits.

$$C = B \log_2 \left(1 + \frac{S}{N}\right) \tag{2.5}$$

The distance between the transmitting and receiving antennas will affect the SNR. As the distance increases the signal power decreases due to free space loss and other losses in the signal path. In LoRa applications the SNR-values will typically be low due to long distances and low sending power. In low SNR conditions increasing bandwidth also increases noise, and will not increase channel capacity noticeable. The only opportunity to realize a link at longer distances, i.e lower SNR, without increasing sending power is to decrease channel information rate. Hence, it is a trade-off between information rate, and the distance between the transmitter and the receiver.

LoRa implements a dynamic sending parameter called spreading factor (SF).

This parameter can be manually or automatically tuned to optimize data rate at different distances. The SF defines the amount bits in each symbol, and also the symbol time T_s. LoRa defines SF7 - SF12 corresponding to defined data rates (DR), see table 2.1. The symbol time T_s is defined in eq2.6.

$$T_s = \frac{2^{SF}}{BW} \tag{2.6}$$

BW is the bandwidth of the channel. The SF is also the number of bits in each symbol. Hence, the bit time T_b can be calculated, see eq2.7.

$$T_b = \frac{T_s}{SF} = \frac{2^{SF}}{SF \cdot BW} \tag{2.7}$$

$$R_b = \frac{1}{T_b} = \frac{SF \cdot BW}{2^{SF}} \tag{2.8}$$

R_b in eq2.8 is the nominal data rate.

Table 2.1: LoRa data rates with nominal bit rates based on $BW = 125\,000$ Hz.

DR	**SF**	**Nominal bit rate [bit/s]**
DR0	SF12	366
DR1	SF11	671
DR2	SF10	1221
DR3	SF9	2197
DR4	SF8	3906
DR5	SF7	6836

E_b/N_0 is a ratio used to compare digital modulation techniques, but can also be used to compare the different SFs in LoRa. E_b is energy per bit sent and N_0 is the noise spectral density [W/Hz]. For LoRa E_b can be calculated as shown in eq2.9.

$$E_b = S \cdot T_b = S \cdot \frac{T_s}{SF} = \frac{2^{SF}}{SF \cdot BW} \tag{2.9}$$

S is the average signal power. Assuming constant S the energy per bit E_b is proportional to bit time T_b. E.g when SF increases from SF7 to SF8, energy per bit increases from $E_b = \frac{S}{BW} \cdot \frac{2^7}{7}$ to $E_b = \frac{S}{BW} \cdot \frac{2^8}{8}$. This is an increase of approx 2,29. Although the signal power is held constant the energy per bit is increased. Hence, increasing distance between transmitter and receiver, and increasing SF, will use more energy per message sent, assuming constant message size. Another drawback of increasing SF is increased time on air (ToA), i.e time spent transmitting. This affect how often a message can be sent if the band is regulated. LoRaWAN regulations si explained in section 2.2.6.

Another transmit parameter is coding rate (CR). LoRa uses forward error correction coding to ensure a reliable link. It has four defined CRs which describes the ratio between user data bits and total bits, see table 2.2. The error correction scheme used is proprietary and not revealed by Semtech.

Table 2.2: LoRa coding rates.

CR	**Coding Rate**
1	$\frac{4}{4+1}$
2	$\frac{4}{4+2}$
3	$\frac{4}{4+3}$
4	$\frac{4}{4+4}$

LoRa limits the payload size in a frame and is specific for each DR[19]. Table 2.3 shows maximum payload sizes in Europe.

2.2.4 Adaptive Data Rate

To optimize the data rate and power consumption LoRaWAN offer adaptive data rate (ADR). The goal is to decrease the SF, i.e increase data rate and decrease power consumption, while retaining an acceptable packet error rate (PER), see Figure 2.5. If an end-device want this service it sets a flag in its uplinks.

The standard ADR algorithm implemented for LoRaWAN by The Things

Table 2.3: Maximum LoRa payload in Europe.

Data Rate	**Max Size (bytes)**
DR0	51
DR1	51
DR2	51
DR3	115
DR4	222
DR5	222

Network[20] looks at the SNR values of the last uplinks from the end-device in question. The highest SNR value from these uplinks, with a margin, is compared with the theoretical SNR at the current SF. If the actual value is higher than the theoretical value, the SF can be adjusted down and DR increased. Opposite, if the value is lower or violating the margin, the SF is increased. More specifically the algorithm is based on eq2.10 and eq2.11.

$$SNR_{margin} = SNR_{max} - SNR_{DRx} - margin_{dB} \tag{2.10}$$

$$N_{step} = int(SNR_{margin}/2.5) \tag{2.11}$$

SNR_{margin} is the difference between highest SNR value, SNR_{max}, and the theoretical for the current DR, SNR_{DRx}, with the margin $margin_{dB}$. N_{step} is the number of DR steps to move, where a positive number is an increase in DR and vice-versa. If N_{step} is 0 the DR is correct. The $int()$ operator rounds the argument down to the nearest integer. The value 2.5 is the step size.

2.2.5 Link Reliability

To ensure that the recipient receives the message from the sender, LoRaWAN has implemented confirmed uplinks and downlinks[22]. When either an end-device or a gateway sends a confirmed message it expects an acknowledge from the recipient. If no acknowledge is received a resending procedure will be initiated. This usually

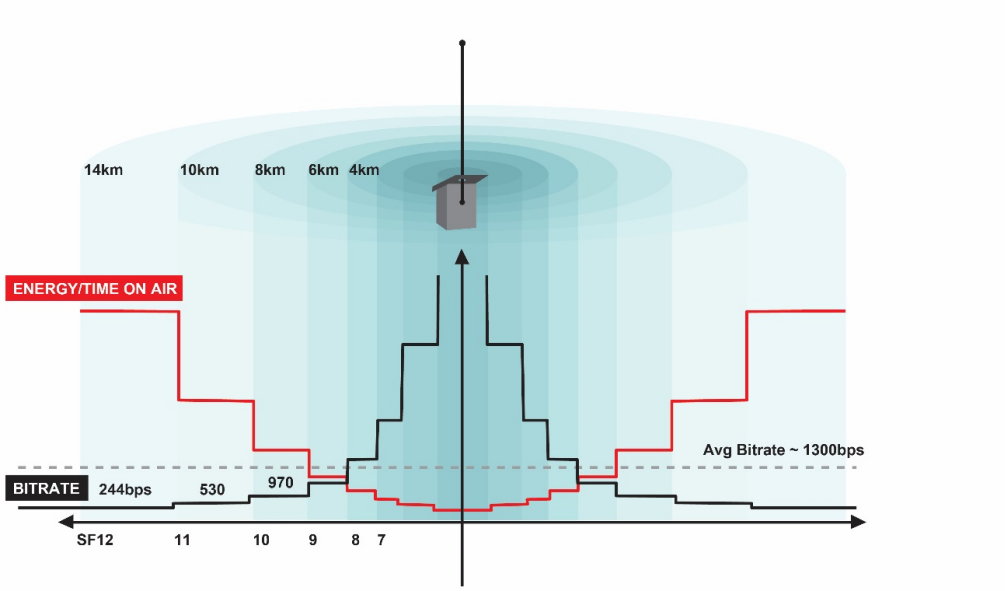


Figure 2.5: LoRa energy consumption and time on air versus distance, spreading factor and data rate. Figure from [21].

conclude in resending with a higher SF. Though link reliability is advantageous it comes with a price. For end-devices resending can potentially be expensive in terms of power consumption and duty-cycle. For a gateway it can potentially be expensive in terms of duty-cycle, if demanded to acknowledge uplink for several end-devices.

LoRaWAN defines four message types for normal operation after join: Unconfirmed uplink and downlink, confirmed uplink and downlink. The two latter is expecting acknowledgement from recipient.

2.2.6 Regional Regulation Parameters

As mentioned in section 2.1.3, the ISM radio frequency band is regulated. Based on these regulations, there are defined a set of regional parameters for each region[23]. These parameters define channel carrier frequencies, bandwidth, transmission power, duty-cycle, etc., used by the participants in a LoRaWAN network. The

transmission power is restricted by effective isotropic radiated power (EIRP). This is the total power radiated from a theoretical isotropic antenna and measured in Watts (W). I.e the transmission power amplified with the antenna gain (dBi). The duty-cycle restriction limits the ToA, i.e access time of the frequency band. It is the transmission time over a time interval ratio. E.g after an end-device has sent an uplink it has to wait for amount of time to avoid duty-cycle violation. This further decrease the possible data throughput in a LoRaWAN network.

2.2.7 Data Throughput

Though LoRaWAN have a defined nominal data rate, the actual data throughput is much lower[14]. First, a LoRa frame adds some overhead including sync symbols, a header, a checksum, and error correction coding. Second, the LoRaWAN protocol adds some overhead including addresses, flags and more. However, the most limiting factor is the duty cycle preventing continuous transmission.

2.3 LoRaMAC in C

LoRaMAC in C (LMiC) is a library implementing a LoRaWAN device. It was originally developed by IBM, but is today maintained by MCCI[24]. The original version by IBM has ported to Arduino, an easy-to-learn development platform, and is the version continued by MCCI. Behind the hood the library is based the C programming language and it can be ported to other hardware than Arduino, though with a supported radio module and interface. An example component structure is shown in Figure 2.6.

A hardware abstraction layer (HAL) is needed to access the needed resources on a host. The HAL is platform dependent and the developer must tailor it for the specific hardware. The LMiC needs access to system time, GPIO, interrupts and SPI.

The library version used in this project is v4.1.0[24] and it supports Class A and Class B devices in LoRaWAN version 1.0.2/1.0.3 in several regions.

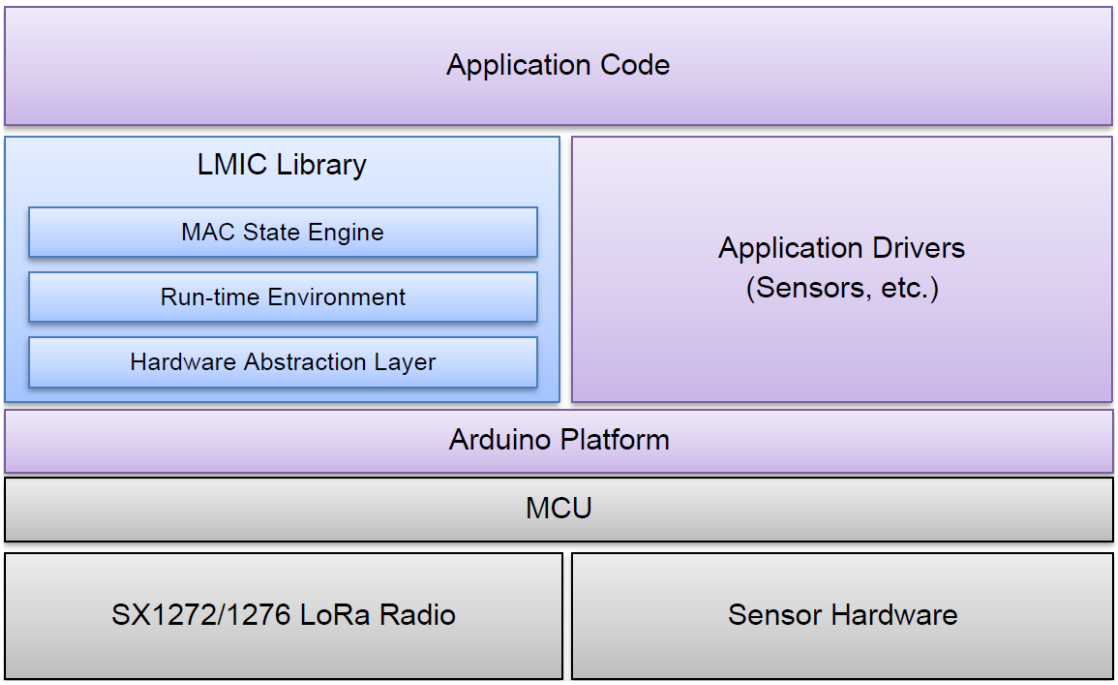


Figure 2.6: Example component structure in an application using LMiC. Figure from LMiC documentation[24].

2.4 Global Navigation Satellite System

A Global Navigation Satellite System (GNSS) is a common concept for all satellite based navigation systems, e.g the American Global Positioning System (GPS) or the Russian GLONASS. The systems were initially deployed for military purposes, but have later been opened for civilians. They consists of multiple satellites circling the Earth to serve GNSS receivers with navigation information such as position and velocity. This section will introduce the basic principles in GNSS operation, and will not go into technical details on the matter.

2.4.1 Multilateration

In fact it is not the navigation satellites who provides the position of a GNSS receiver. The satellites basically broadcast its own position with an accurate timestamp. A GNSS receiver uses multiple of these broadcasting signals to

determine its position and speed. The receiver measures the distance to each satellite using the timestamp included in the signal. Using three satellites the GNSS receiver can calculate its position. However, due to imperfect clock oscillators in the GNSS receiver, it can't measure time accurately enough. Hence, it needs four satellites to solve for the time offset, i.e clock drift.

2.4.2 GNSS Modes

When an GNSS receiver starts up without any knowledge of visible navigation satellites and their location, its called a cold start. At a cold start the receiver uses relatively long time to find the first fix. A fix is a finished calculation of position and time. The cold start is tedious because the receiver have to look through a wide band of frequencies to find signals from the satellites. The received signal frequency is shifted from the original signal frequency used by the satellite transmitter. The shift is a result of the Doppler effect and clock drift in the receiver. During the operation of searching for satellites signals, called acquisition, the GNSS receiver is working at full power and it's advantageous that the searching time is short, concerning battery powered devices. The searching time is called time-to-first-fix (TTFF) and is often in the magnitude of minutes.

When the GNSS receiver has calculated the first fix, the consecutive calculations will be faster and often in the magnitude of seconds. They are faster because the receiver knows the previous fix and has a narrower frequency band to search in. Depending on the application the interval between fixes vary, but the fix calculation duty cycle is usually less than during acquisition. Hence, the power consumption is less during further operation. This stage is called tracking. When a satellite disappears behind the horizon it becomes out of range. Then the receiver needs to step into acquisition mode for a period of time to find new satellites to track.

Concerning battery powered devices with GNSS receivers, these operations contribute to a higher power consumption. This can be handled with low power modes. Cyclic tracking is one option where the receiver cycles between tracking, fix calculation and idle. The quality of the fixes will not be reduced, but the interval

between each fix will increase to a magnitude of seconds. On/Off operation is an other option. In this mode the receiver cycles between acquisition and sleeping. When a fix is acquired the receiver goes to sleep. After some time it wakes up and start acquisition again. To make On/Off operation power efficient, the interval between each acquisition needs to be longer than in cyclic tracking operation and is in the magnitude of minutes, hours or days.

2.5 Message Queuing Telemetry Transport

Message Queuing Telemetry Transport (MQTT) is a messaging protocol used for machine-to-machine (M2M) and IoT communication. It was initially developed by IBM and is today an open OASIS standard[25]. The protocol is lightweight and it can be implemented on embedded devices with limited resources and network bandwidth. It uses a publish/subscribe (pub/sub) architecture where clients publish and subscribe to topics. Subscribing clients receive messages from clients publishing on a common topic.

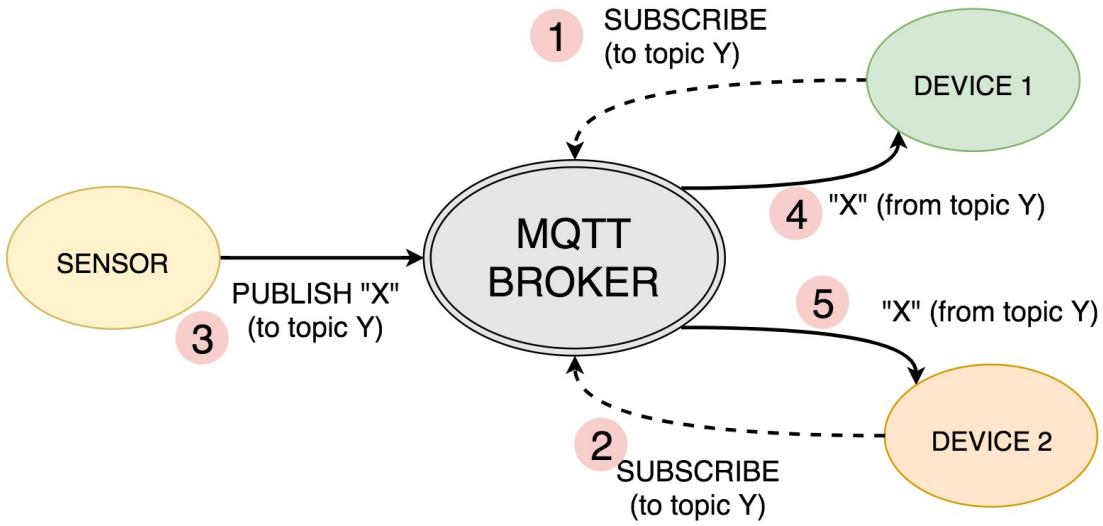


Figure 2.7: Example MQTT device structure. Figure from [26].

The MQTT protocol creates a star-topology network with a central broker, see Figure 2.7. The broker is the manager between all clients, and forward published

messages to the correct receivers. Clients only know the broker and not each other. Though clients communicate with a common topic, it does not guarantee that other malicious clients intervene by subscribing and publishing on the same topic. Hence, the broker can create password protected users that can connect to prevent unwanted clients to intervene. Moreover, the broker can set up Transport Layer Security (TLS) to encrypt message content and password.

Chapter 3

Internet of Fish System

3.1 System Overview

The goal of the IoF system is to monitor fish in real time. Fish have implanted a tag in their stomachs which periodically broadcasts data signals. These signals are picked up by surface receivers. The surface receivers logs the data in its memory and forward the data in real time. In addition to fish tag data, the surface receivers provide sensor data measuring water temperature and noise levels around it. This part of the IoF system is developed by Thelma Biotel. They manufacture fish tags[2] and receivers[3].

The rest of the system is developed by former students on the IoF project and the author of this thesis. The IoF smart buoy consists of a SLIM and an antenna above the surface in addition to a Thelma Biotel Receiver (TBR) below the surface. The SLIM receives the forwarded data from the TBR and in turn forwards it to shore. In addition, it has a GNSS receiver and can provide position data to the user and time to the TBR.

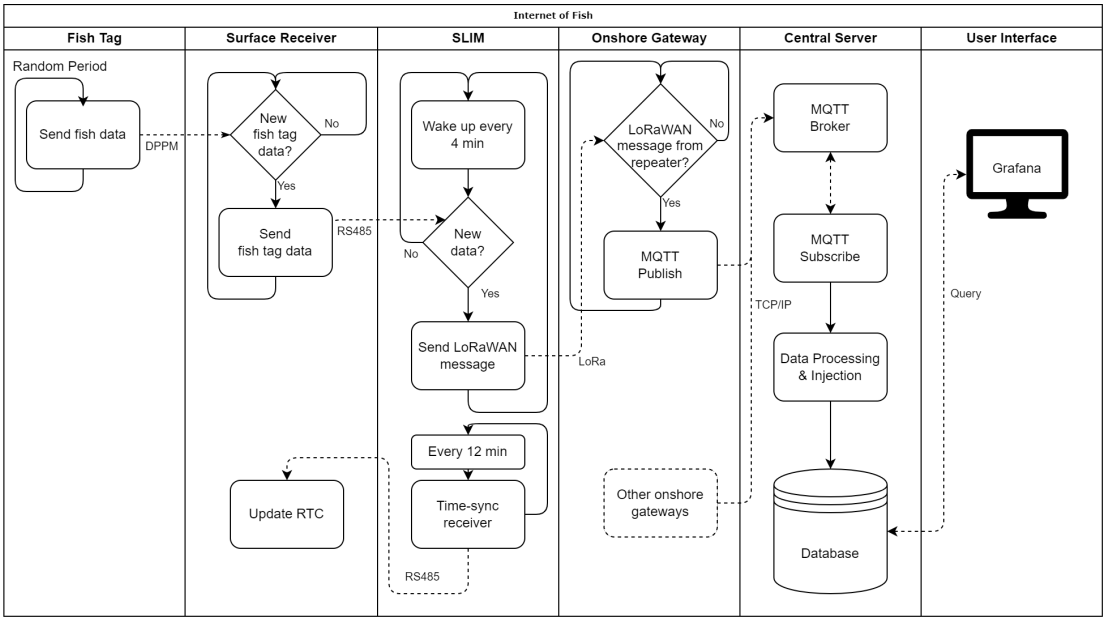


Figure 3.1: Flow chart showing IoF data flow. Differential Pulse Position Modulation (DPPM) is the protocol used by the acoustic fish tags. RS485 is the serial protocol used between the surface receiver and the SLIM. LoRaWAN and LoRa is used between the SLIM to the onshore gateway. Communication between gateway, central server and user interface is done over the Internet using TCP/IP.

3.2 IoF Buoy Construction and Assembly

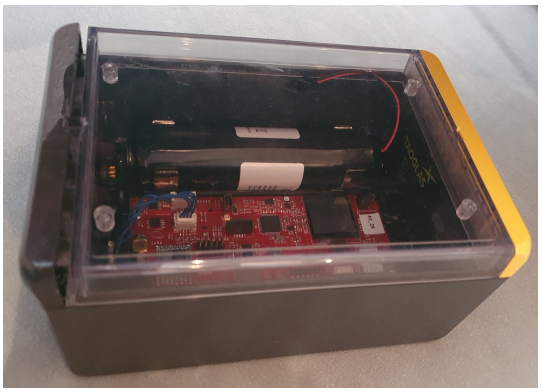
The IoF buoy consists of the buoy itself, a floating device, and a pole ranging into the water and up in the air. The pole above the surface elevates the LoRa antenna, approximately 2 m above sea level (ASL), to increase link quality and is the mounting point for the SLIM. Below the surface is a stabilizing weight at the pole end and a bracket to hold the TBR. From the stabilizing weight the buoy is anchored to the bottom of the sea to prevent it from drifting away. A cable between the TBR and the SLIM is fitted inside the hollow pole. Similarly, an antenna cable between the SLIM and the LoRa antenna is fitted inside the top pole. The entire pole length can be divided into three pieces to ease transportation. The assembled buoy without anchoring and with two SLIMs (normally one) can be seen in Figure 3.2b. Figure 3.2a shows the TBR bracket and the lower entrance hole for the TBR cable.

(a) TBR bracket and lower cable entrance hole, taken by the author of this thesis.

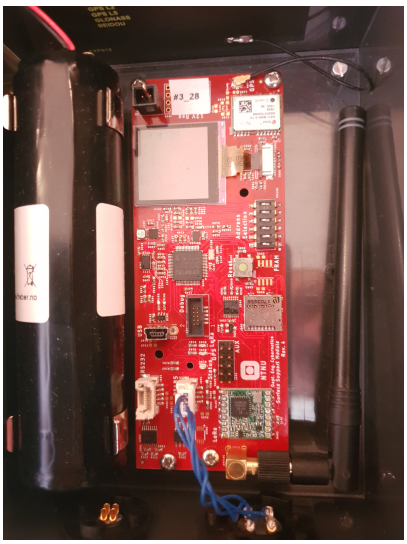
(b) The IoF buoy. Picture from field test in Gaulosen, taken by the author of this thesis.

3.3 Synchronization and LoRa Interface Module

The SLIM has as mentioned the purpose of sending data to shore and synchronizing the TBR with time. It is an end-device in a LoRaWAN network connecting to the onshore gateway. It is mounted to the pole above the floating element, see figure 3.2b. The SLIM consists of a waterproof box with a printed circuit board (PCB) inside, see Figure 3.3. In addition, it has exterior connectors to the TBR and the external LoRa antenna, a GNSS receiver antenna and a battery mount.



(a) Waterproof box.



(b) SLIM printed circuit board.

Figure 3.3: Synchronization and LoRa Interface Module. Pictures taken by the author of this thesis.

3.3.1 SLIM PCB

On the PCB sits a Silicon Labs EFM32 Giant Gecko[27] microcontroller unit (MCU), along with several external peripherals. Their function and interconnectivity is explained in this section and showed in Appendix A.

Microcontroller Unit

The central brain on the SLIM PCB is the EFM32™ Giant Gecko 32-bit MCU by Silicon Labs[27]. This is were the software, explained in section 3.3.2, runs and it directly or indirectly control all the other external peripherals on the PCB.

The EFM32 is a 32-bit Reduced Instruction Set Computer (RISC) based on the ®Advanced RISC Machine (ARM) architecture. It is low-power with low power consumption using energy-saving modes. Moreover, it has short wake-up times from energy-saving modes to active mode to respond on events and get back into energy-saving. Figure 3.4 shows a block diagram of the MCU and the energy-saving modes.

The MCU is configured with two external oscillators, a 32 768 Hz low frequency crystal oscillator (LFXO) and a 48 MHz high frequency crystal oscillator (HFXO). The LFXO drives two low frequency clock and the HFXO drives a high frequency clock. Together the clocks drive the central processing unit (CPU) and different embedded peripherals in the MCU. The HFXO frequency is divided by 8, hence the high frequency clock run at 6 MHz. This will reduce

It offers 5 energy modes from EM0 to EM4. The first energy mode, EM0, is the normal run mode with everything active. In EM1 the CPU is sleeping, but all other peripherals are available. In EM2 the HFXO is turned off and only selected low energy peripherals are available. In EM3 the LFXO is turned off as well, leaving only a few peripherals able to wake-up the device. In EM4 everything is off except a reset pin, general purpose input/output (GPIO) pins and a backup real time counter (BURTC). The device can retain random access memory (RAM) in this state. The SLIM uses the MCU in EM1 when not active and running in EM0.

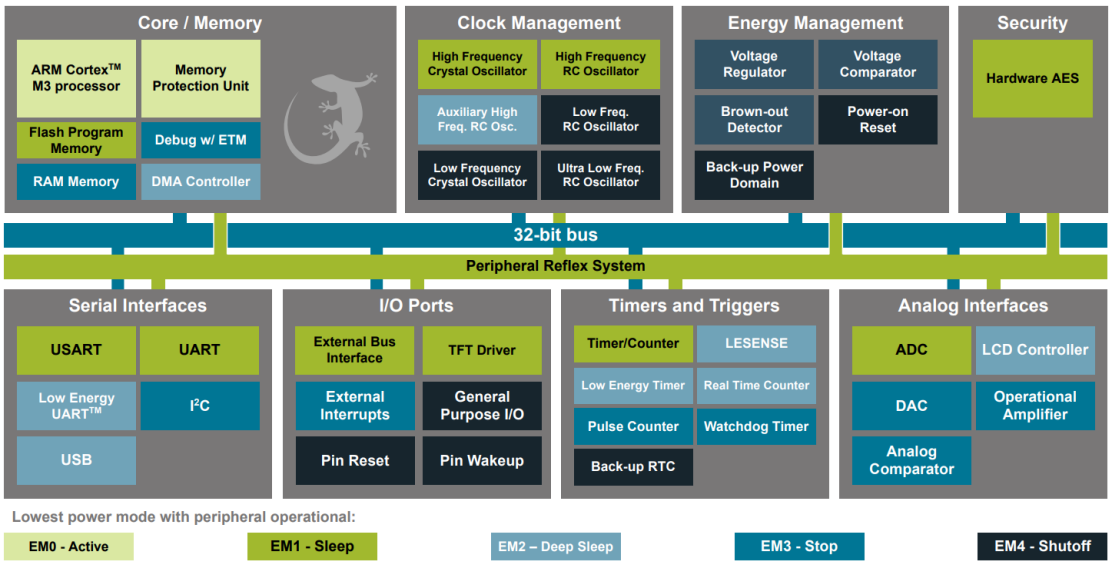


Figure 3.4: EFM32GG block diagram showing all peripherals. Figure taken from the datasheet[28].

GNSS Receiver

Figure 3.5: NEO-M8N GNSS receiver by u-blox. Picture taken from their website[29].

The GNSS receiver provides geolocation service on the SLIM. It is a NEO-M8N module, see Figure 3.5, from u-blox[30] providing reception of signal from GNSS satellites. The MCU configures and fetch geolocation data through the provided Serial Peripheral Interface (SPI) using a protocol called UBX. A TAOGLAS GNSS antenna is connected to the receiver and glued to the SLIM's interior box wall.

The receiver is configured to be "manually" controlled by the MCU. It is configured with an external pin that forces it in and out of back-up mode. In back-up mode the receiver is sleeping and only retaining a RTC and RAM. When the MCU sets this pin though GPIO the receiver starts up and starts searching for GNSS signals. When the MCU reset the pin the receiver enter back-up mode. The receiver remember valuable data from previous searches to speed up the TTFF next search.

LoRa Radio

The LoRa radio module offer transmitting and receiving of LoRa frames for the LoRaWAN protocol. It is a RF96 module by Hope Microelectronics[31]. The radio is configured and controlled by the LMiC library using SPI and GPIO. It can either be connected to an internal LoRa antenna inside the SLIM box or to the external LoRa antenna on top of the buoy.

RS-232 Interface

A RS-232 transceiver is available on the SLIM is used as a serial interface. RS-232 is a serial communication standard. The interface is used for debug printing, e.g sending text status messages from the MCU. The transceiver is connected to a UART interface and GPIO on the MCU. The MCU can modify the transceiver's state with GPIO pins.

RS-485 Interface

A RS-485 transceiver is available on the SLIM is used as a serial interface to the TBR. RS-485 is a serial communication standard. The transceiver is connected to a low energy UART (LEUART) interface and GPIO on the MCU. The MCU can modify the transceiver's state with GPIO pins. The transceiver is half-duplex and can either receive or transmit. It is in the receive mode whenever not transmitting, to reduce power-consumption and receive asynchronous serial data from the TBR.

Status Display

The status display offers the user/developer a status interface. It is a liquid crystal display (LCD) and can print text that inform the user/developer about the SLIM's status in run time during debugging or deployment. The display configured and controlled by MCU using SPI.

Status Lights

The PCB is equipped with four light emitting diodes (LED). They are also used to provide status to the user/developer. With a quick glans they can say something about the SLIM's state. There is one dedicated to GNSS status, one dedicated to LoRaWAN status and two for general status. The LED controlled by MCU using GPIO.

Address Switches

SLIMs deployed with identical software can get a unique ID with the address switches. The address switches provide a 6-bit ID. The ID is used as a part of the LoRaWAN ID when joining a LoRaWAN network. The address is read by the MCU using GPIO.

Power Supply

The SLIM PCB can be powered either through a Universal Serial Bus (USB) power or battery. There is a power multiplexer that chooses USB power if available or battery power elsewhen. When deployed the SLIM uses the battery. It is a double D cell thionyl chloride lithium battery with 3.6 V nominal voltage and 35 A h nominal capacity, see Figure 3.6.

Figure 3.6: Double D cell thionyl chloride lithium battery. Picture taken by the author of this thesis.

3.3.2 Software Implementation

The software running on the SLIM, i.e the MCU, is based on drivers from Silicon Labs SDK. This includes low level drivers for the MCU's peripherals. The SLIM software source code, except SDK, is provided in Appendix B.

Scheduler

The LMiC runs the LoRaWAN stack and require control over the MCU's resources to guarantee protocol compliance. To allow other software to run on the MCU it implements an OS scheduler. All software shall run through this OS after the LMiC library is initiated. However there is a small exception to this; interrupt service routine (ISR) can run in the event of an interrupt. As always, ISR should be short and only update states and do not perform heavy tasks.

The OS allow application software to run through so-called jobs. A job is linked with a function call, and can be scheduled to execute as soon as possible or in the future. The scheduler is non-premptive, i.e tasks run until completion. LMiC documentation[24] stress that jobs must not be long-running in order to ensure seamless operation. They should only update states and schedule new actions.

The MCU implements a heartbeat ISR from the BURTC. The heartbeat ISR is executed every second where local time is updated and application jobs are scheduled using the LMiC OS. The jobs are described in Table 3.1.

LoRaWAN Manager

The LoRaWAN manager is the applications interface to the LoRaWAN network using the LMiC library. The LMiC version is updated since used by former IoF contributors and in the specialization project (Appendix C). It allows the application to queue uplinks, which are sent when possible. In addition, it offers network status information. If the application queues a confirmed uplink the LoRaWAN manager adds it to a first-in-first-out queue with a capacity of 10 uplinks. On the other hand, if it queues an unconfirmed uplink, it is added to a buffer with the capacity of 1 uplink. The confirmed uplinks are prioritized and will be sent first, and then the unconfirmed uplink is sent. Any potential new queue attempts for unconfirmed uplink will overwrite unsent uplink in buffer, which will be lost. This is acceptable since it is a unconfirmed uplink. Any potential new queue attempts for confirmed uplinks, exceeding the queue capacity, will also be lost. The LMiC sets the SLIM up as a Class A end-device with ADR.

Table 3.1: IoF Jobs description and run interval.

Job	Description	Execution Interval
Blink job	Turns on status LEDs. Schedules job to turn them back off after 100 ms.	5 s
Display job	Updates the LCD display with potential new information.	5 s
Poll Navdata job	Initiate navigation data polling. Time period offset in relation to App job.	12 min
Sync job	Time synchronize the TBR. Follows after navigation data retrieval.	12 min
App job	Fetches any received TBR messages, queues an uplink to the LoRaWAN network if any new tag, TBR or buoy data are to be sent	4 min

TBR Interface

To receive data from the TBR and time synchronize it the SLIM MCU needs a TBR interface. Data and commands are exchanged using character strings. The command interface is described in the TBR documentation[32].

The interface communicate with the TBR over the RS-485 interface. To send characters from a string they are added to a sending buffer. The buffer is processed through an ISR which sends the next character in the buffer when the previous has been sent, until the buffer is empty. On the other hand, when characters are received from the TBR they trigger an ISR which adds them to a receive buffer. The receive buffer can later be read and the characters can be interpreted as a string.

The interface also parse incoming TBR messages and creates data structures that can be passed to the application.

GNSS Interface

The GNSS interface provide configuration of the u-blox receiver, control of the external control pin and navigation data polling.

IoF Software Application

The IoF software application uses the interfaces above and implements the IoF buoy functionality. It is implemented as multiple jobs which are scheduled periodically, see Table 3.1. The App job process IoF data and sends it on the LoRaWAN network. First, it fetches new TBR data, i.e tag detection and TBR sensor data, if any. Second, it builds an IoF frame (explained in section 3.6) to be sent on the LoRaWAN network. If requested the application also add buoy data like position, battery status and air temperature. Last, an LoRaWAN uplink is queued if any data are to be sent.

The nature of the LMiC OS scheduler does not let it implement any kind of task sleep functionality. Hence, a job needs to add busy-wait delay if it needs to wait for something. However, it is also possible to divide a job into smaller sub-jobs, where a sub-job schedule the next one to execute in the future. This will work like a sleep functionality, but is cumbersome to implement if a job requires many of them. However, the Blink job, the Poll Navdata job and the Sync job are divided into two parts for this reason.

3.4 Onshore Gateway

The onshore gateway is a Multitech Conduit IP67 Base Station[33]. It is a outdoor IoT gateway designed to deploy LoRaWAN networks, see Figure 3.7. In the IoF project the gateway is configured both as a gateway and a network server in terms of a LoRaWAN network explained in section 2.2.1. The network server can be

modified with respect to regional parameters and other desired behavior. What ADR algorithm that is used is not revealed, but probably similar to the one described in section 2.2.4. The gateway needs a power connection and either a cellular or a Ethernet connection to the Internet.

Figure 3.7: Multitech Conduit IP67 Base Station[33]. Here without LoRa and cellular antennas.

Moreover, the gateway implements a MQTT-client. All received LoRaWAN uplinks are published to a designated MQTT broker. Metadata about the LoRa link, like RSSI, SNR and sequence number, is also published. This makes it possible to monitor the LoRa link performance.

3.5 Cloud Server

The cloud server is a virtual machine running the Linux OS Ubuntu 18.04[34]. It is responsible to receive data from deployed gateways, process it, store it in databases and present it to the user. The server has an reachable global IP address.

3.5.1 MQTT Broker

The server implements a MQTT broker as explained in section 2.5. It is an open source MQTT broker by Eclipse called Mosquitto[35]. This is the broker that the

MQTT clients in the gateways connect to. The broker is configured with password protection and TLS.

3.5.2 Data Processing Instance

To process the received data, the server implements a Node-RED[36] instance. Node-RED is a graphical programming language that makes it easy to process data in so-called flows. In a flow there are input and output nodes, and function nodes in between that manipulate data.

In the IoF system, the flow has an MQTT input node. This node acts as a MQTT client and subscribes to the broker mentioned above. The data coming from the gateways via the broker is then parsed. The parsing involves bit shifts and combining raw bytes into distinct data fields. These data fields are injected into the database using an output node. Hence, the Node-RED instance works as a parser and a transition between two other technologies, see Figure 3.8.

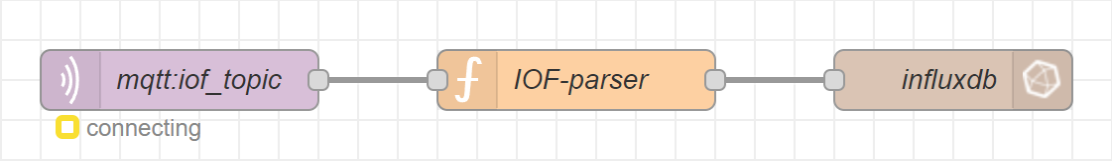


Figure 3.8: A simplified version of the Node-RED flow running on the cloud server. Figure from the Node-RED graphical programming interface.

3.5.3 Time Series Database

A database is used to store historical data from the IoF system and make it available in the future. In this system an InfluxDB database is used. InfluxDB is a database platform by InfluxData[37], optimized for time-series data. It fits this application for all data that have a timestamp like tag data and LoRaWAN link performance data.

3.5.4 User and Developer Interface

The IoF data is presented to the user using Grafana. Grafana is an open-source observability platform developed by Grafana Labs[38]. It visualizes data from databases in user defined dashboards with graphs, tables and diagrams, and support database queries to many databases. In this system it sends queries to InfluxDB and present the data in adequate graphs and tables. There are three main dashboard created for the user: TAG detections, TBR data, Buoy data (Figure 3.9, 3.10, 3.11). In addition there is a dashboard monitoring the LoRaWAN link, to measure its performance.

TAG detections

This dashboard present tag detections within a user defined time period. They are presented in a table with tag ID, TBR serial number, tag data value, metadata and a timestamp. Next to the table, there is a map with markers where the buoys are located. This helps the user understand where the a fish has been detected, see Figure 3.9. Data and metadata of selected tag IDs are plotted in time series graphs.

TBR data

This dashboard present TBR sensor data within a user defines time period. Acoustic noise in the water is plotted as two graphs in a time series diagram. One graph represent the average noise and the other represent peak noise, both measured within the last TBR measuring period. The water temperature is plotted as graph in another time series diagram, along with air temperature. Though air temperature is measured by the SLIM and not the TBR, it is a convenient comparison. See Figure 3.10.

Buoy data

This dashboard present information data about a selected buoy within a user defined time period. A buoy is identified with the connected TBR's ID. The

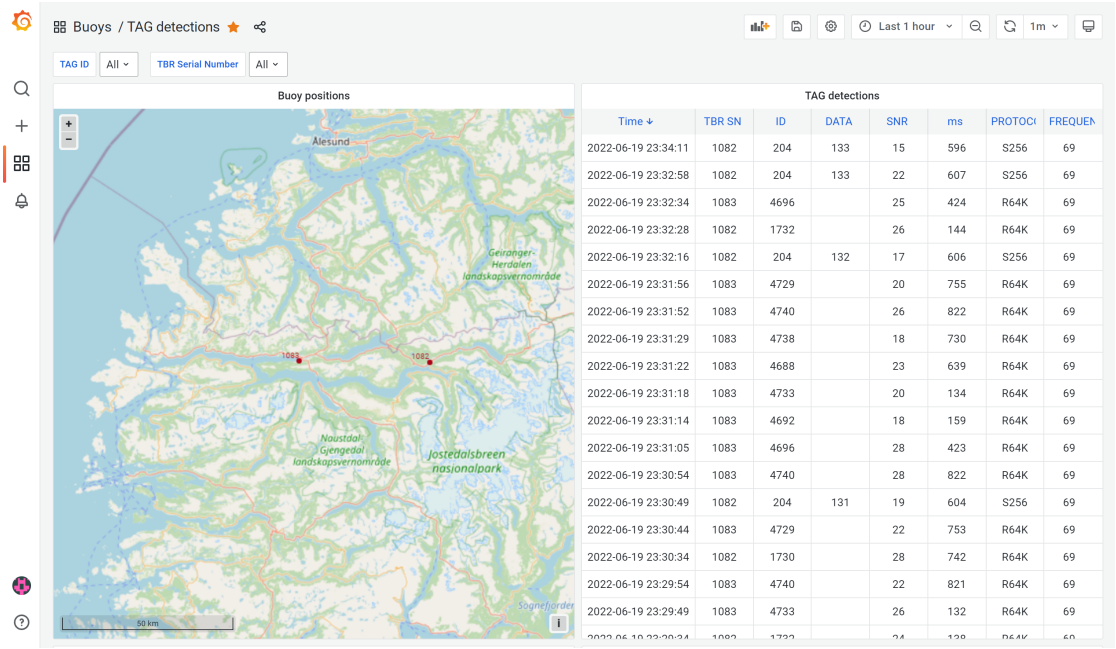


Figure 3.9: TAG detection dashboard in Grafana®. Map from ©*OpenStreetMap* contributors [39].

buoy's location is presented with a marker on a map. Furthermore, GNSS metadata, temperature data and battery voltage is presented in graphs in time series diagrams. See Figure 3.11.

LoRaWAN Link

This dashboard present metadata about the LoRaWAN link provided by the gateway that receives uplinks from a selected buoy. Every uplink within a user defined time period is presented in a table and metadata plotted in graphs in time series diagrams, like in the TAG detection dashboard.

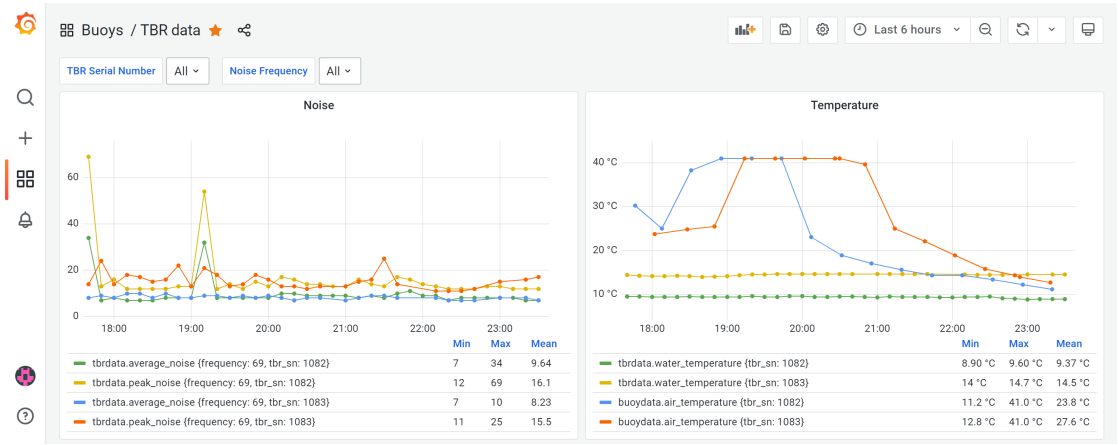


Figure 3.10: TBR data dashboard in Grafana®.

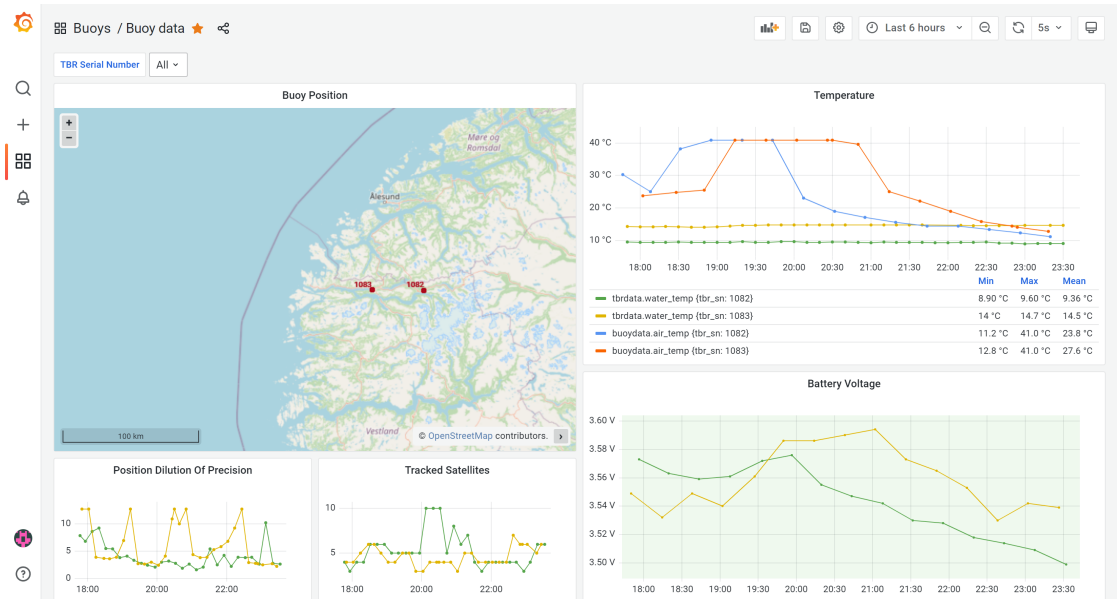


Figure 3.11: Buoy data dashboard in Grafana®. Map from ©*OpenStreetMap* contributors [39].

3.6 IoF Protocol

The IoF protocol is designed for the IoF system and is used between the SLIM and the cloud server. It is optimized for size to reduce access time on LoRa frequencies. To reduce size the data fields are reduced to a minimum resolution and merged together in a bit field. This happens in the SLIM software before sending a LoRaWAN uplink. The bit field is parsed when arriving the data processing instance in the cloud server. Data fields are extracted and prepared for database injection.

The protocol has been developing through different students and the version used here are by Kjelsvik. It is thoroughly described in page 41 to 48 in his thesis[6]. He defines a dynamically sized frame that consists of different building blocks, here called packets. The frame always contains a header. The header size is constant (6 bytes) and includes the connected TBR's serial number, header flags and a reference timestamp. Then there are three possible packets: tag detection packet, TBR sensor packet, and SLIM packet.

Tag detection packets includes fish tag ID and data, tag protocol and tag carrier frequency, a detection timestamp relative to the reference timestamp, a detection millisecond timestamp, and a SNR value. The tag protocol defines the size (5-7 bytes) of the tag detection packet, which depend on the tag ID and data resolution.

TBR sensor packet includes sensor data from the TBR and a detection timestamp relative to the reference timestamp. The packet size is constant (7 bytes) and it identifies as a TBR sensor packet with a field similar to the tag protocol field in a tag detection packet.

SLIM packets include data about the buoy and the SLIM in question. These are measured SLIM battery voltage, latitude, longitude, position dilution of precision (PDOP), number of tracked satellites, fix type. The SLIM packet size is constant and is 10 bytes.

The IoF protocol is slightly modified in this project. The first modification is in the header flag section where there was a free flag spot. Here it is added a

timesync flag. The flag is set in uplinks before valid time has been provided and synchronized to the TBR, or if the SLIM fail to synchronize the TBR. The flag will notify the user that the packet timestamp is not to be trusted. The second modification is in the measured battery voltage field in the SLIM packet. The field implementation is modified to alternate between containing a measured battery voltage value and a air temperature value. The first bit in the field is used as a flag to tell whether the value is voltage or temperature.

3.7 Developing Tools

3.7.1 Integrated Development Environment

Simplicity Studio 5 is Silicon Labs' IDE[40]. It is used to develop, program and debug Silicon Labs' MCUs and development kits. The SLIM's EFM32 MCU is connected to Simplicity Studio with a J-Link Debug Probe by Segger. The probe allow programming and debugging the MCU. In debug runs, one can stop the program at breakpoints and look at memory and variable values. This is helpful while looking for bugs.

3.7.2 Power Debugger

Power Debugger[41] is a development tool by Microchip that can measure an MCU's power consumption. The tool is designed for Microchip's AVR and SAM devices, but also comes with two current sensing channels. Power Debugger can measure the current consumption of any device by intercepting the power supply and route it through one of the current sensing channels. It has a software interface to configure it and monitor current. The software interface is integrated in Microchip Studio, which is Microchip's IDE.

Chapter 4

Tests & Deployments

4.1 Test Overview

To verify the functionality and measure performance of the IoF system, several tests were conducted. Initially during the development, the system was tested indoors to validate that sub-modules and the total system of the system worked as intended. Tests results from these tests are not directly described in this thesis, but was the basis for several of the design choices described in Chapter 3. When the system seemed to work fine, it was field tested in realistic marine environment. There functionality and communication range were tested. Finally, two buoys were deployed in actual scenarios with real fish. The deployments were coordinated with a biology project where 196 salmon were tagged with acoustic transmitters in two rivers. To estimate and evaluate power consumption and battery life, power consumption tests were conducted indoors on a desk.

4.1.1 Atlantic Salmon

Atlantic salmon is a fish that spend most of its life in The Atlantic Ocean, but also in fresh water and rivers. It is very popular for recreational fishing, food and especially export through aquaculture industry, according to The Great Norwegian Encyclopedia[42]. Hence, biological scientist study them to learn their behavior and health. The wild Atlantic salmon is anadromous and starts its life in fresh water rivers before it moves to salt water in fjords and oceans. After 1-5 years in a river it migrates downstream the river to the ocean where it grows and matures. After becoming mature it returns to a river to spawn new fish eggs. According to the Institute of Marine Research[43], the salmon usually return to the exact same river that it was spawned itself. Biological scientist yet don't know for sure how they do it, but evolving technology offering fish tracking has become available the last couple of decades.

4.2 Field Test in Gaulosen

The IoF buoy was tested in realistic environments in a fjord called Gaulosen in Trøndelag County, Norway. The SLIM was tested for overall functionality due to new software and software optimizations, as well as the LoRaWAN link was tested for performance. The buoy was equipped with two SLIMs to run side-by-side tests to differentiate LoRaWAN link performance with respect to LoRa antenna placement. At the same time, two different passive GNSS receiving antennas could be differentiated by performance.

Figure 4.1: Buoy (yellow diamond) and gateway (red triangle) location in Gaulosen. First and second buoy location marked with "1" and "2", respectively. Figure from [44].

4.2.1 Buoy Setup

The buoy was set up like in a deployment, see Figure 3.2b in section 3.2, with two SLIMs mounted to the pole. SLIM A was connected to the TBR and the external LoRa antenna on top of the pole. SLIM B was not connected to anything and used the internal LoRa antenna. Also the two SLIMs used two differently sized GNSS receiver antennas. A fish tag was fastened under water close to the TBR to simulate fish data. It sent tag messages at random intervals, but on average every minute. The "fish data" was a value incrementing with 1 up to 255, then wrapping to 0, and so on.

The external LoRa antenna's elevation was 200 cm ASL, and the internal LoRa antenna in SLIM B had an elevation of 65 cm. The initial distance between the buoy and the gateway was approximately 5.5 km. Later in the test, the buoy was moved and the distance reduced to 2.9 km, see map in Figure 4.1.

The SLIM software was slightly changed for the purpose of this test. The two SLIM boxes was equipped with two different passive GNSS receiver antennas. To compare these two, the battery voltage measurement field in the IoF protocol SLIM packet was exchanged with a TTFF and air temperature field. I.e the SLIM did not send battery voltage measurement byte, but TTFF and air temperature measurement combined in one byte. The TTFF measurement, in combination with PDOP and number of tracked satellites, offered a basis for antenna comparison. See the two antennas in Figure 4.2.

Moreover, the IoF application interval was set to 1 minute, i.e the SLIM software will check for tag detection messages every minute and send if any. Since SLIM B was not connected to any TBR it would not send very often. Hence, it was set up to send SLIM packet every minute to increase test result data.

4.2.2 Gateway Setup

The onshore LoRaWAN gateway was mounted to a rack on a floating jetty offering an onshore power supply. The LoRa antenna's elevation was 345 cm ASL.

(a) Small passive GNSS antenna.

(b) Big passive GNSS antenna.

Figure 4.2: Two TAOGLAS® passive GNSS antennas. Pictures from [45].

4.2.3 Test Results

The test lasted for 20 days. However, some modifications to software, hardware and location were done during this time period. This divided the test into smaller time periods with different circumstances, see Table 4.1. Note that SLIM A would receive a tag detection approximately every minute, which led to sending a confirmed uplink every minute. In contrast, SLIM B sent an unconfirmed uplink containing a SLIM packet every minute.

Test results was retrieved from the InfluxDB database through the Grafana dashboard.

Table 4.1: Test time sub-periods in chronological order.

Test ID	**Time Period [days]**	**Description**
GAUL1	5	Initial setup.
GAUL2	14	SLIM A: reset and switch to internal LoRa antenna.
GAUL3	1	Software update and relocation. SLIM A: switch back to external LoRa antenna.

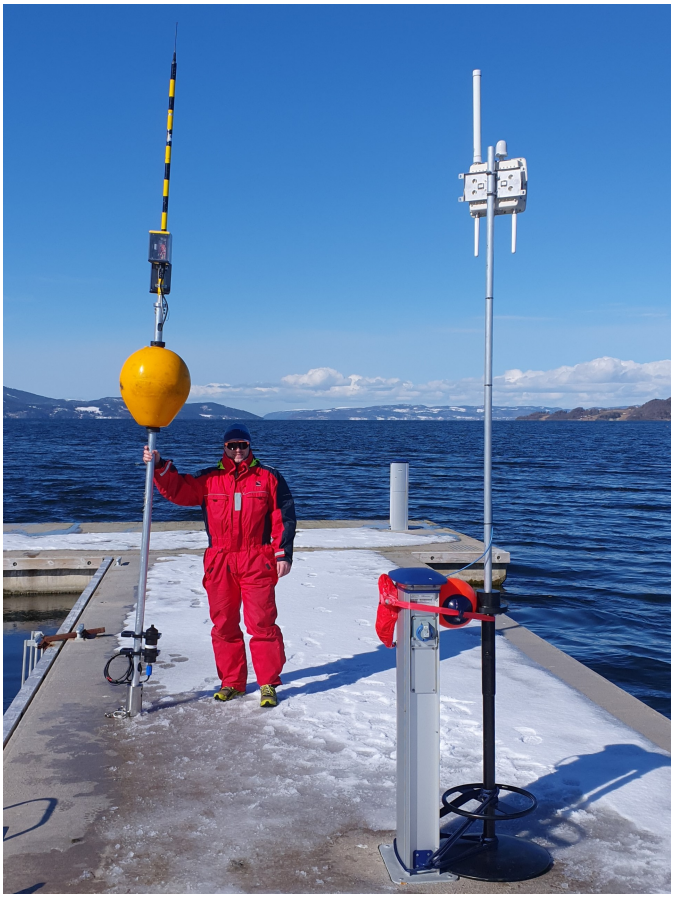


Figure 4.3: Buoy and gateway setup at field test in Gaulosen. Picture taken by author of this thesis.

GAUL1

SLIM A behaved very strangely from the start. It joined with SF8 and began to increase to SF12 after a few minutes, due to lost uplinks. Then it started to send the same message approximately every 20 minutes for a day, before it apparently returned to normal operation. That lasted for 2-3 hours before it went silent for days.

SLIM B seemed to behave good. It kept sending uplinks throughout the period and jumped between different SFs. Looking at the received uplinks one could see that the ADR tried to optimize the link. However, the PER was unacceptably

high; 44.6 %. It was calculated based on expected and received uplinks. Moreover, it had 6 re-joins during this test period.

GAUL2

Though, SLIM B had a bad PER it had a plausible continuous link. For the second test period GAUL2, the external antenna used by SLIM A was exchanged with an internal antenna to resemble SLIM B. No changes were done to SLIM B.

SLIM A continued the strange behaviour and went silent after 6 hours. Moreover, the average SNR value from the received uplinks was -5.2 dB.

SLIM B also continued with the same behaviour, but with an even worse PER; 55.5 %. It had 9 re-joins in this period. Average SNR value was -4.4 dB.

GAUL3

A potential software bug was found, that could cause the strange behavior of SLIM A. Both SLIMs were updated with new software. The buoy was moved to a location closer to the gateway, see location "2" in Figure 4.1. The hypothesis about the external antenna failing was discarded and SLIM A used the external antenna for the final test period.

SLIM A started to behave adequate with next to no packet error and didn't miss any tag detection. Again it went silent during the next night, but restarted the next morning. Moreover, there was still some issues, but the behaviour while the SLIM was alive was better. Figure 4.4 shows the RSSI and SNR value results. The silent period during the night seemed to look like a battery issue, since battery voltage decrease with low air temperatures during the night. Hence, this battery was starting to reach the end of its lifetime and the SLIM was off during the night, but woke back up in the morning with higher air temperatures. Therefore, this silent period can not be compared with others in previous test periods, since those appeared more random.

SLIM B continued the same behaviour, though with better SNR values due to

shorter distance to the gateway. The average SNR value this final period was 0.8 dB and no re-joins. The PER was 25.7 %, which was better than previous test period at longer distance, but still bad.

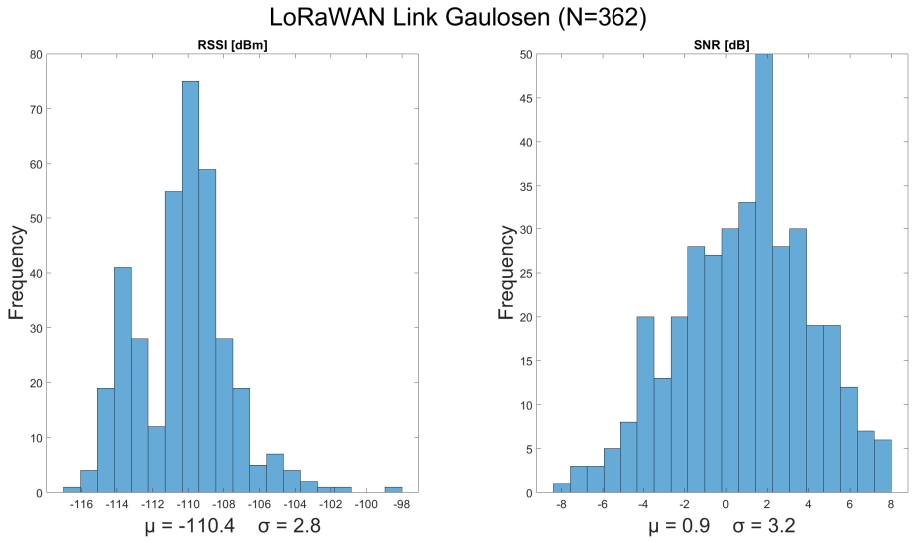


Figure 4.4: Histogram showing RSSI [dBm] and SNR [dB] value frequency from received LoRaWAN uplinks from SLIM A during test GAUL3 (described in section 4.2.3), measured by the gateway.

TTFF test results

The measured TTFF by both SLIM A and SLIM B was random and in a pattern hard to analyze. Both antenna types (Figure 4.2) performed similar in term of average TTFF around 8 s.

4.3 Deployment in Nordfjord

Figure 4.5: IoF buoy deployment in Stryn.

After the field test in Gaulosen began to show improving results, the focus turned to deployment in Nordfjord before the upcoming salmon migration. Nordfjord is one of the main fjords in Vestland County, Norway. More specifically, in two rivers in Nordfjordeid and Stryn (Figure 4.6), recently 196 salmons were tagged. One buoy was deployed at each location where the river flows into the fjord. This offered realistic deployment environment and results with real fish, see Figure 4.5.

The placement of the buoys was strategic to detect fish migrating from the river. However, there were no strategy behind the gateways location other than an available power outlet and a plausible radio link.

4.3.1 Buoy and Gateway Location

Nordfjordeid

In Nordfjordeid the gateway was placed close to the buoy, approximately 320 m away, onshore and a bit uphill. Figure 4.7 shows the location of the buoy and the gateway. Figure 4.8 shows the terrain profile and line of sight between them.

Stryn

In Stryn the gateway was placed further away from the buoy and at a greater elevation. The line of sight distance between them was approximately 3.2 km. Figure 4.9 shows the location of the buoy and the gateway. Figure 4.10 shows the terrain profile and line of sight between them.

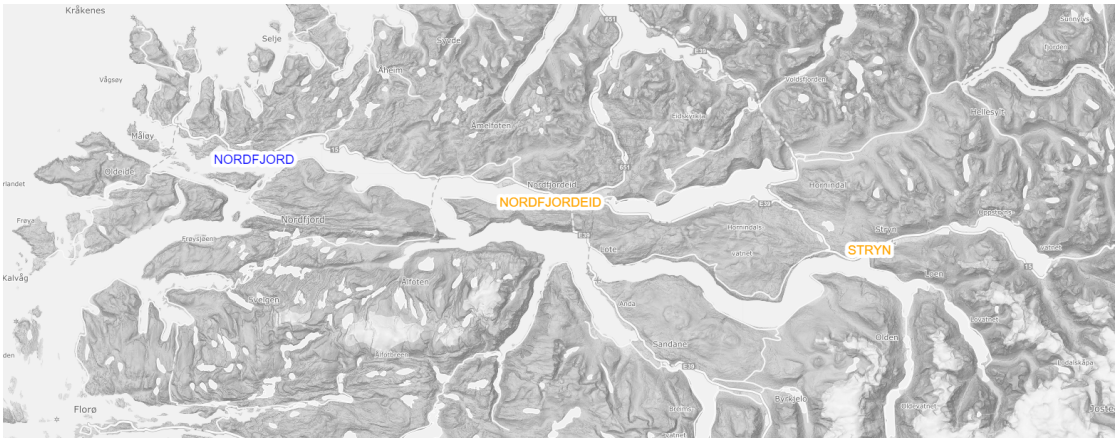


Figure 4.6: Map of the area around Nordfjord. "NORDFJORD" in blue is marking the start of the fjord near the sea. "NORDFJORDEID" and "STRYN" is marked with orange. Figure from [44].

Figure 4.7: Buoy (yellow diamond) and gateway (red triangle) location in Nordfjordeid. Figure from [44].

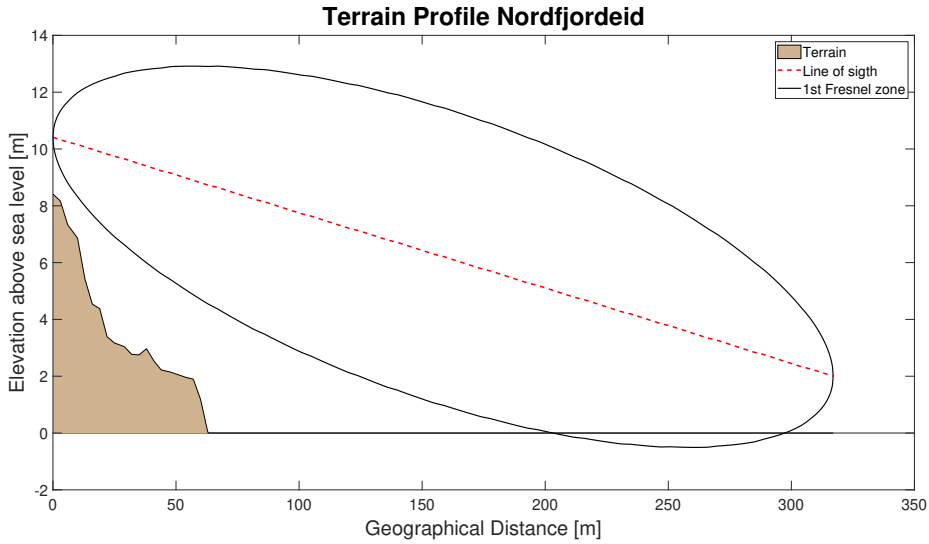


Figure 4.8: Terrain profile between gateway and buoy in Nordfjordeid. The red dashed line is the line of sight. The black ellipse is a two-dimensional representation of the 1st Fresnel Zone. Terrain data from *NVE Atlas*.

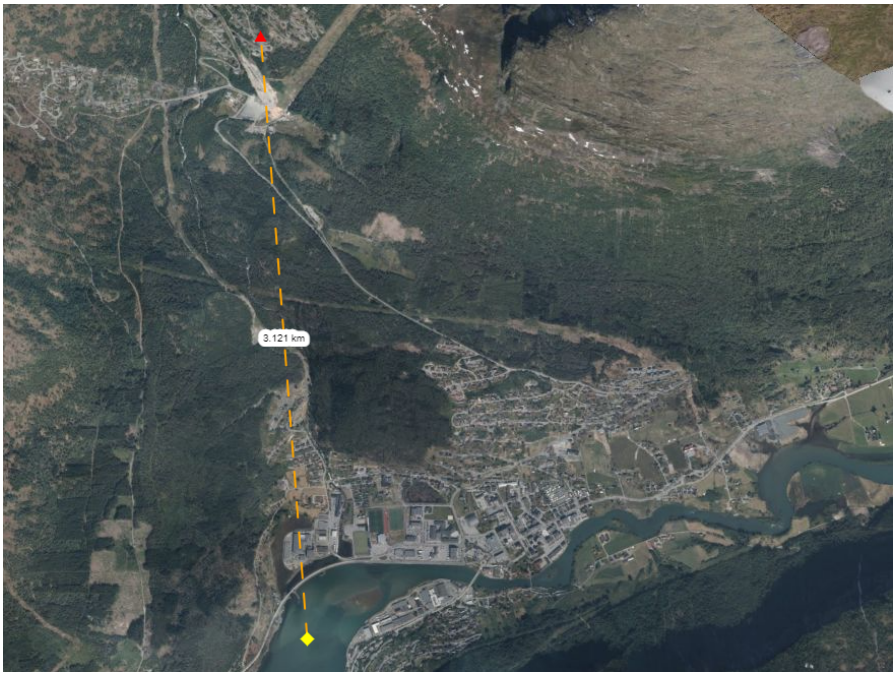


Figure 4.9: Buoy (yellow diamond) and gateway (red triangle) location in Stryn. Figure from [44].

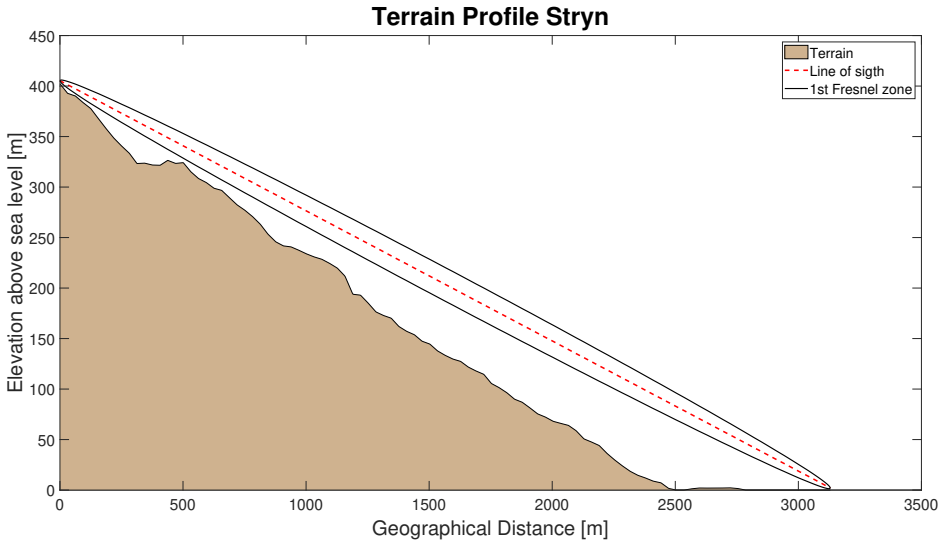


Figure 4.10: Terrain profile between gateway and buoy in Stryn. The red dashed line is the line of sight. The black ellipse is a two-dimensional representation of the 1st Fresnel Zone. Terrain data from *NVE Atlas*.

4.3.2 Test Results

The buoys were monitored from the Grafana dashboards and tag detections started to occur right after deployment. From the IoF data showing up in Grafana everything seemed to work as it should. It included tag detections, TBR sensor data and buoy data. The results presented in this section is based on data from deployment on April 29th until June 9th.

LoRaWAN Link

From the LoRaWAN Link dashboard in Grafana the LoRaWAN link was monitored. Both buoys joined immediately with SF7 and started sending uplinks every 4 minutes.

One way to evaluate the SLIMs LoRaWAN behaviour is to examine the uplinks' sequence number (SeqN). The SeqN is an incremental counter value added to each uplink. It starts from 0 after joining the network. The SeqN from the buoy in Stryn increased linearly over time and was never reset to 0, shown in Figure 4.11. It has skipped 8 SeqNs due to packet loss. The buoy in Nordfjordeid has a different behavior, shown in Figure 4.12. The SeqN follows a pattern looking like a saw-tooth wave. The SeqN was reset to 0, 197 times. This was due to re-joins or joins as a side-effect of a power-cycle on either the SLIM or the gateway. Another behaviour of the buoy on Nordfjordeid where silent period with no uplinks, barely visible in Figure 4.12. Three different events can be defined from the SeqN:

1. A SeqN reset with no silent period.
2. A silent period of no uplinks **and** a SeqN reset.
3. A silent period of no uplinks, but SeqN continue **without** reset after.

The radio link quality in terms of RSSI and SNR is presented in Figure 4.13 and Figure 4.14.

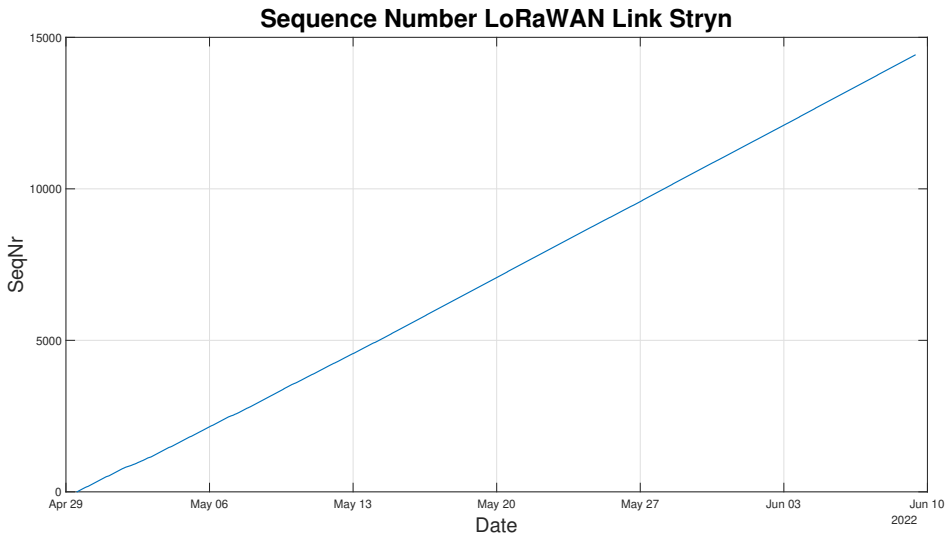


Figure 4.11: Sequence number in received uplinks from buoy in Stryn from April 29th until June 9th.

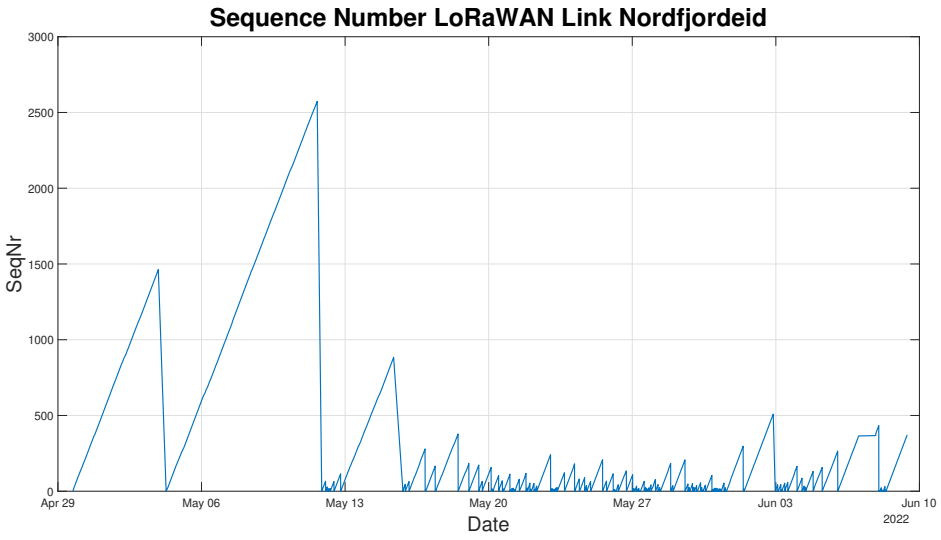


Figure 4.12: Sequence number in received uplinks from buoy in Nordfjordeid from April 29th until June 9th.

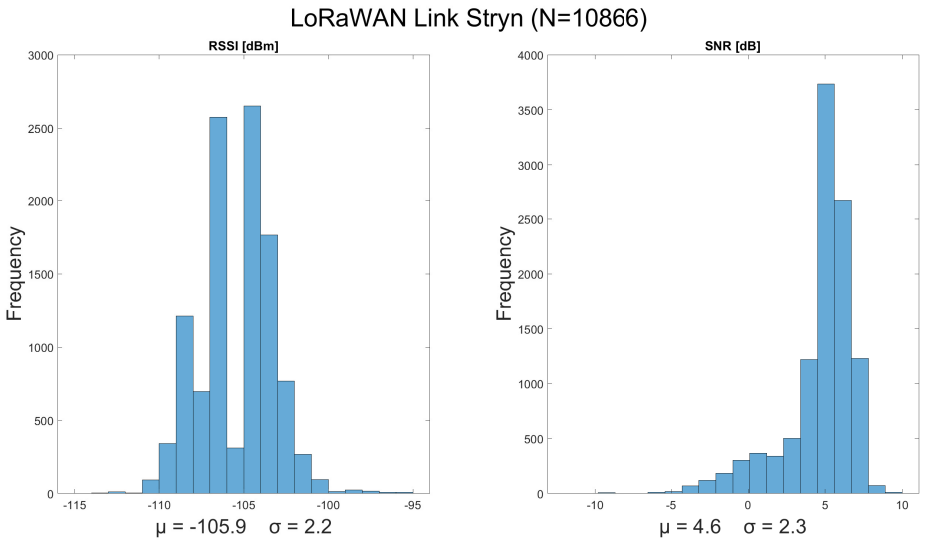


Figure 4.13: Histogram showing RSSI [dBm] and SNR [dB] value frequency from received LoRaWAN uplinks in Stryn, measured by the gateway during May 2022. μ is the average value and σ is the standard deviation.

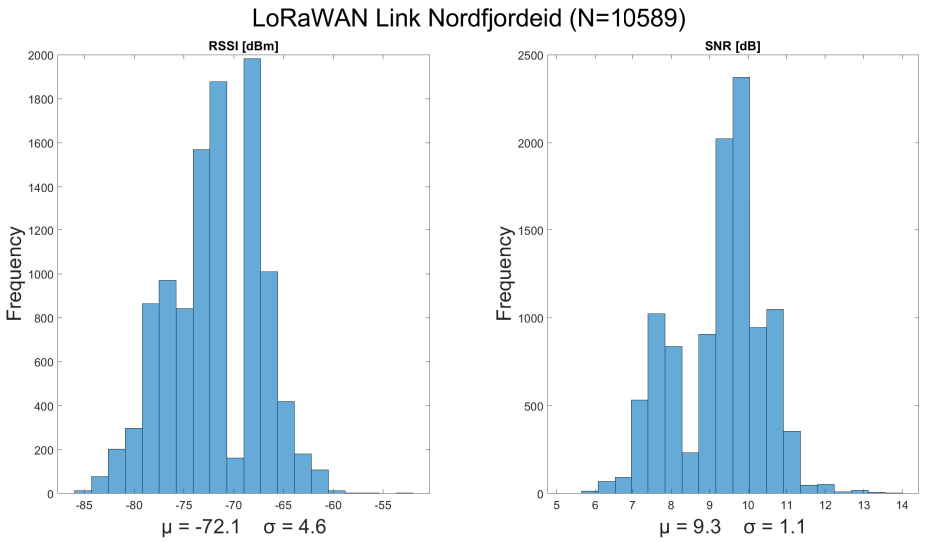


Figure 4.14: Histogram showing RSSI [dBm] and SNR [dB] value frequency from received LoRaWAN uplinks in Nordfjordeid, measured by the gateway during May 2022. μ is the average value and σ is the standard deviation.

4.4 Power Consumption Tests

To evaluate a battery powered device's lifetime, a power consumption estimate is necessary. A power consumption estimate can provide a lifetime estimate, i.e time from deployment to a necessary battery replacement, and also provide information to optimize the device setting to increase lifetime.

Several power consumption tests were conducted to measure a SLIM's average power consumption, but also to map what is causing it to increase or decrease. The test was done indoor on a desk in an unrealistic environment, but will give a clue to the power consumption of a SLIM deployed in the fjord. The tests measured current which can be translated into power by multiplying it with the voltage of the power source.

4.4.1 Power Debugger Setup

The Power Debugger was used to measure average current over a finite time period. Adequate time periods was needed to ensure periodic current contributions were added correctly. E.g if a current contribution is based on a current burst every second, the measuring time period should be 1 s or a multiple of that. The SLIM was powered with USB through Channel B on the Power Debugger, see Figure 4.15. Hence, the measured current is based on a 5 V power source.

4.4.2 Peripherals' Power Contribution

To map what causing power consumption to increase or decrease, the different SLIM peripherals was enabled one by one to measure their power contribution. First the idle power consumption was measured, i.e the power consumption with no peripheral enabled. Second, peripherals were enabled to measure their power contribution beyond the idle power consumption.

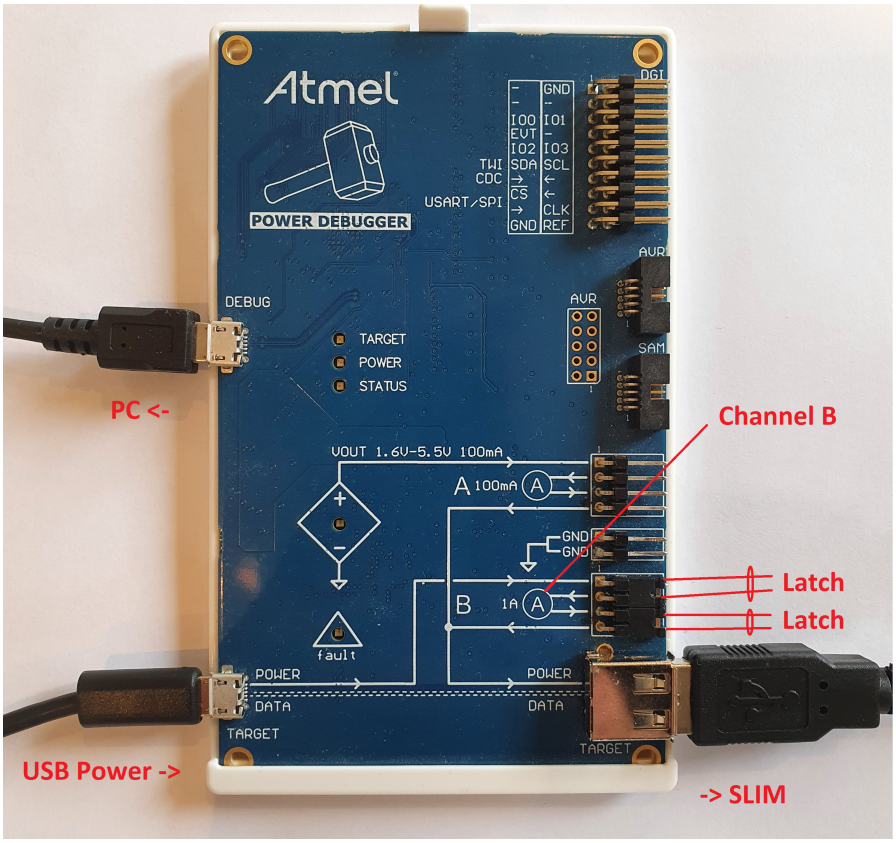


Figure 4.15: Power Debugger setup for current measurement on the SLIM.

Setup

The SLIM software was modified to disable all peripherals. Moreover, the peripherals were still initialized and put into normal configuration, but not used. From now this is called the idle state. Beyond this idle state, software was modified to enable peripherals with adequate operation settings, see Table 4.2. Any additional start-up current was ignored in this test, and was not added as a contribution to the average current.

Idle State

In the idle state the MCU ran the LMiC OS with the heartbeat ISR every second. In between jobs it slept in energy mode EM1. RS232 transceiver was initialized as

normal and not in shutdown mode. Debug printing was not disabled during the testing, but did not add any noticeable current contribution when not connected to a PC. The LCD display was initialized as normal, but with no updates. The LEDs were off. The GNSS receiver was initialized as normal, but constantly in back-up mode. The LoRa radio was initialized as normal and put into sleep mode. RS485 transceiver was initialized as normal and put into receive mode. During the tests it was beneficial to keep the J-Link debugger connected to ease consecutive software updates, and was also a part of the idle state in these tests.

Results

These results can be transformed to be more generic for arbitrary SLIM setup. The generic average current contribution is presented in Table 4.3.

4.4.3 Battery Life Estimation

The current consumption results above can be used to estimate the battery life of a SLIM, i.e deployment period, assuming the battery life of the TBR is greater. To get a realistic estimate calculation, the behavior of the buoy deployed in Stryn is used as a basis. The behavior and peripheral utilization, as well as power contribution, is showed in Table 4.4.

The battery has a nominal voltage of 3.6 V and a nominal capacity of 35 A h. As a best case scenario the battery has a capacity of 126 W h and a lifetime of approximately 206 days in Stryn, based average power consumption in Table 4.4. In a worst case scenario the battery has a voltage of 3.5 V and a capacity of 30 A h. This gives a lifetime of 172 days.

Table 4.2: Peripheral current contribution at 5 V. "+" contributions adds to idle state current and "-" contributions reduce current.

Peripheral	**Operation**	**Current contribution**
Idle state		3.79 mA
Energy mode EM2		- 0.91 mA
RS232 Transceiver	Connected to PC	+ 2.68 mA
	Shutdown mode	- 0.29 mA
RS485 Transceiver	Transmitting 10 bytes every 10 s	$\sim$ 0 mA
LCD display	Update with 0.2 Hz	+ 0.53 mA
	Update with 0.1 Hz	+ 0.27 mA
LEDs	1 with duty cycle 2 %	+ 0.07 mA
	1 with duty cycle 1 %	+ 0.04 mA
	2 with duty cycle 2 %	+ 0.13 mA
GNSS receiver	Acquisition (searching)	+ 23.2 mA
LoRa	LoRaWAN uplink SF7, 10 byte payload, every 10 s	+ 1.99 mA
	LoRaWAN uplink SF7, 10 byte payload, every 20 s	+ 1.01 mA
	LoRaWAN uplink SF8, 10 byte payload, every 20 s	+ 1.27 mA

Table 4.3: Generic peripheral power contribution. "+" contributions adds to idle state power and "-" contributions reduce power.

Peripheral	**Power contribution**	**Details**
Idle State	19.0 mW	
Energy mode EM2	- 4.55 mW	If activated during sleep
Debugger	- 0.90 mW	If not connected
RS232	+ 13.4 mW	If connected
LCD display	+ 13.3 $\frac{\text{mW}}{\text{Hz}}$	per Hz
LEDs	+ 17.5 mW	per LED per duty cycle
GNSS receiver	+ 116 mW	per duty cycle
LoRa SF7	+ 9.95 $\frac{\text{mW}}{\text{Hz}}$	per message per byte per second

Table 4.4: SLIM average power consumption in Stryn.

Peripheral	**Utilization**	**Power contribution**
Idle State	Continuously	19.0 mW
Debugger	Not connected	- 0.90 mW
LCD display	Update with 0.2 Hz	+ 2.65 mW
LEDs	1 with duty cycle 2 %	+ 0.65 mW
GNSS receiver	Assuming average TTFF (based on tests in Gaulosen): 10 s. Duty cycle: 1.4 %	+ 1.6 mW
LoRa	Uplink with SF7 every 4 min. Assuming average payload: 60 bytes.	+ 2.5 mW
Total Average Power		25.5 mW

Chapter 5

Discussion & Conclusion

5.1 Discussion

This section will discuss the software design choices based on test results, theory and previous work. The discussions will focus on the SLIMs performance in terms of battery life and QoS. The other components of the system have worked as expected. This includes the acoustic transmitter and receiver, the shore-based gateway and the backend server. These components are important for the system, but not adequate for discussion in this thesis.

5.1.1 SLIM Power Consumption

Reducing the SLIM's power consumption, thus increasing battery life, have been a major part of the main focus and the reason behind software design choices. Compared to Rundhovde's[7] results the battery lift has become longer. He accomplished a lifetime of around 4 months when the SLIM encounters several tag detections every minute. The power consumption estimate in section 4.4.3 shows a lifetime estimate of around 6 months. This section discuss some of the software choices done to increase battery life and weight them against functionality requirements.

Removing Time-Synchronization

When optimizing the SLIM for the single monitoring buoy scenario, only time awareness is necessary. That is, the opportunity to fetch time and date with an approximately 1-second accuracy, precision and resolution. Note that there is no attempt of verifying time accuracy in the software and the time fetched from the GNSS receiver is naively accepted. A time offset of greater than 1 s is not expected, but can occur. The main use-case for time awareness is after SLIM start-up when it has no memory of time and space. The other use-case is to update the SLIM's and TBR's system time regularly to remove large clock drifts.

In Rundhovde's software, time-synchronization consisted of a state-machine controlling the GNSS receiver, and a system time controller. The system time

controller measured the oscillator frequency based on a reference pulse-per-second (PPS) signal from the GNSS receiver, and temperature, to control the system time drift by modifying the BURTC top counter value. I.e it defined one second based on the PPS signal and used that second to update system time. Moreover, the state-machine woke the GNSS receiver every minute and monitored it while awake. It was awake until position and time had been found and sent to the SLIM. By removing time-synchronization the MCU can spend less time controlling the GNSS receiver and system time. If this has a noticeable effect on the power consumption depends on the time-synchronization's duty cycle, i.e how much of total time it uses for computations. Anyway, removing it will not make the power consumption worse.

GNSS Receiver Usage

The new software design uses a less sophisticated approach when using the GNSS receiver. The GNSS time and position is requested less frequently, every 12 min, and interact using low duty-cycle polling. The duty-cycle of the GNSS receiver depends on the TTFF which will vary for each GNSS search, but will be significantly reduced by increasing the GNSS search interval. The execution duty-cycle of the GNSS interface on the MCU is low. It checks for new data, and if no data it sends a new poll request. This action is scheduled every second until new data is received. Here it is important to note that the MCU use a pseudo sleep functionality between each poll request and not busy-wait delays. Busy-wait delays let the MCU processor spin in a loop until desired time is over, that would result in a duty-cycle of 100 % during GNSS search.

The GNSS interface uses a GPIO pin to manually control the sleep state of the GNSS receiver. The interface wakes it up when position and time data is requested and puts it back to sleep when this data is received. A similar behavior could also be implemented using the on/off feature. The on/off feature can schedule wake-up times and put the GNSS receiver back to sleep after a fix is acquired. This feature is abandoned because of a weird hardware design choice in the GNSS receiver. It wakes up whenever there is activity on the SPI line, also when the MCU is

communicating with other external peripherals. This will wake it up at unwanted times and override the time schedule. This issue is also discussed by Rundhovde[7].

Both the duty-cycle reduction in GNSS activity and in GNSS interface execution should reduce the power contribution. Furthermore, more testing and a more thorough survey of the u-blox could be conducted to reduce it even more. The issue with the on/off mode on the u-blox receiver is that a lot of data gets lost when not tracking the satellites, hence, the TTFF is long. However, it has a feature that can predict satellites path in the future and reduce the next TTFF. This will require more power testing to see if is worth spending energy on predicting. This is not done in this project.

Potential Hardware Optimizations

In this project the main focus has been software updates, but there are potential hardware changes shouldn't be unmentioned (PCB schematics in Appendix A). As also Rundhovde[7] discovered the current RS485 transceiver solution is very energy hungry. To support the high baud rate (115200 $\frac{\text{bit}}{\text{s}}$) supported by the TBR it requires a resistor bias network. This adds a noticeable power contribution to the SLIM. It would beneficial to reduce the TBR's baud rate to e.g 9600 $\frac{\text{bit}}{\text{s}}$. Moreover, a reduction in baud rate would also add the opportunity to let the MCU sleep in energy mode EM2. This will reduce the power by 4.55 mW. The MCU is currently sleeping in EM1 to keep high frequency oscillator running. If baud rate is reduced this oscillator is not necessary during sleep.

Minimalistic Functional Alternative

Though the above-mentioned power optimization increase lifetime compared to earlier version, it is possible to further strip down the system to further increase battery life. Power contribution from the user interface is a relative big part of the total consumption. In the battery life estimation based on the deployment in Stryn, in section 4.4.3, the LCD screen and LED contribute with 3.3 mW out of 25.5 mW. Though the user interface is very useful during deployment and

potential buoy maintenance, it can be reduced or removed completely to maximize battery life. Another minimalistic alternative is to remove the GNSS utilization completely. Its main functionality is to provide time awareness. According to the LMiC documentation[24], time can also be provided through the LoRaWAN network. Removing the GNSS removes the location service and data that can contribute to water current estimation and theft protection.

However, the TBR can be time synchronized before deployment and retain time awareness throughout the deployment period. It will suffer for some clock drift over time and tag detection timestamps may not be accurate. Nevertheless, this might be beneficial. When removing time updates from the SLIM, the TBR time might be offset absolute time, but relative time between tag detections is consistent. Relative time between tag detections can be more important in a monitoring scenario.

5.1.2 LoRaWAN Performance

Looking at the buoy deployed in Stryn one could say the system works as expected including the LoRaWAN link. The SeqN counter value have not yet been reset and only missed 8 uplinks. The missed uplinks is likely to be unconfirmed uplinks that does not include tag detections. However, looking at the LoRaWAN behavior at Nordfjordeid and during the test at Gaulosen it does not behave appropriately.

Quality of Service

While Hassan[5] and Rundhovde[7] focused on multilateration and time-synchronization in their software version, this version have focused more on QoS in terms of low data loss. In experiments by Hassan the communication system performed a QoS of 92.8 % over 2.5 km. These experiments used unconfirmed uplinks. By using confirmed uplinks the deployment in Stryn shows that a QoS of 100 % is possible. However, this require a stable link, not like during the tests at Gaulosen. Moreover, confirmed uplinks should be used with care. First, it limits scaling of the system. One gateway can not

send acknowledgement to unlimited end-devices without violating its duty-cycle. Second, the combination of a non-optimal ADR algorithm and confirmed uplinks can be bad, requiring the SLIM to send more uplinks than necessary. This is elaborated in the next paragraph. Assuming the ADR algorithm have chosen an adequate SF for transmission, confirmed uplinks increase the QoS.

ADR Weakness

Both the deployed SLIM at Stryn and at Nordfjordeid use SF7 the majority of the time, and this seems to be the correct choice. Hence, the ADR is barely adjusting anything. However, looking at the ADR activity of SLIM A during the first test at Gaulosen, GAUL1, the SF value is fluctuating. Due to packet loss, it needs to increase SF to receive an acknowledge. When the gateway receives the uplink it determines the SF to be too high and demand the SLIM to reduce it down to SF7. This happens both with SLIM A and SLIM B, but SLIM B is not sending confirmed uplinks and will increase SF more seldom. SLIM A does not suffer from packet loss due to resending, but uses a lot of time-on-air to resend uplinks, thus increasing power consumption. In addition, the uplink queue is growing faster than the SLIM manage to send uplinks causing tag detections to be lost. If this happens the QoS decreases. SLIM B suffer from unacceptably high package loss due to sending at incorrect SF. Hence, a better and more sophisticated ADR algorithm should be integrated. E.g integrate a Gaussian filter-based ADR algorithm implemented by Farad[11], etal. This algorithm uses statistics of measured SNR values to optimize the SF and transmit power. This will be much better approach than basing the choice on the best SNR value from the last uplinks. Farad[11], etal., have also implemented a exponential moving average based ADR algorithm.

Software Issue at Gaulosen

During GAUL1 and GAUL2 field tests, SLIM A also suffered from a software issue. A misunderstanding concerning LoRaWAN uplink resending strategy lead to long silent periods with no uplinks. A internal software flag was set whenever the SLIM had an ongoing LoRaWAN transmission. Somehow during failing transmissions

this flag was not reset and froze the entire LoRaWAN software interface. This was fixed in GAUL3 and gave better results.

Consecutive Resets at Nordfjordeid

Section 4.3 described LoRaWAN SeqN reset events occurring on the buoy in Nordfjordeid. These events does not seem to reduce its functionality performance noticeable, but comparing this behavior with the behavior of the buoy in Stryn can help understand the problem. Even though they both run the same software they behave very different when looking at the SeqN. However, what telling them apart is the amount of fish detections occuring over time. With this clue the focus turned to investigating buffer and packet sizes. In the LoRaWAN interface the payload size is defined to be maximum 120 bytes. This should be safe because the TBR interface have queue size of 10 on received packets from the TBR. 10 TBR packets is maximum 70 bytes, a SLIM packet is 10 bytes and the header is 6 bytes. This gives a total payload size of 86 bytes. However, a software issue in the TBR interface can let it use the queue size twice, in conjunction with sending a time-synchronization command and receiving acknowledgement. A potential payload size is then 156 bytes, which is over maximum payload size of 120 bytes. A hypothesis is that this make the system crash or freeze. The LoRaWAN uplink log shows that no uplink is bigger than 120 bytes, which strengthen the hypothesis.

However, a software fix for this issue is tested. This issue might have been partially the reason for all the resets, but even with the issue-fix the issue persisted. With the debugger functionality in Simplicity Studio another issue is revealed in the LMiC library. When the payload size is over 120 bytes the LMiC library calls the HAL failure function, which triggers a system reset. If this issue is caused by bad integration, hardware error or a library malfunction is hard to say, but it limits the potential data throughput in the IoF service. Former IoF developers and users may not have experienced this issue if present, since previous version have had shorter sending intervals and smaller payload sizes.

5.1.3 LoRa Modulation Performance

Removing the formalities and restrictions added by the LoRaWAN layer, LoRa modulation is working. Looking at the deployment in Stryn, the distance between the gateway and the buoy is 3.2 km and the uplinks are sent at SF7. Only one time during the deployment up until May 30th have a uplink been resent at SF8. Also, uplinks have been resent at SF7 sometimes. This never happens more than 10 times during a day giving a resend ratio of less than 0.3 %.

Hence, SF7 is the correct SF for this deployment. This means that LoRa has a much greater potential in terms of range by increasing SF, i.e energy per bit. The RSSI is on average -105.9 dBm which is far from the receiver sensitivity of -137 dBm stated by Semtech[18]. Also Pensieri[10] have experienced RSSI between -120 dBm and -130 dBm at 110 km. In this project the range have been tested to 5.5 km in GAUL1 and GAUL2 field tests. This setup introduce multipathing attenuation due to low gateway altitude, and the ocean block the major part the 1st Fresnel zone. The SFs that is suggested by the ADR algorithm is performing bad, but would probably work fine using higher static SFs. Hence, a link of over 5.5 km would be realizable given a high gateway elevation or a higher SF.

The signal attenuation due to blocking the Fresnel Zone can be evaluated based on the results from GAUL3 and Stryn. The communication distance is approximately the same in both scenarios. In the GAUL3 test the sea is block the major part of the 1st Fresnel Zone. Whereas in Stryn there is no blockage. The deployment in Stryn perform much better in terms of RSSI and SNR. The average RSSI from Stryn and from GAUL3 is -105.9 dBm and -110.9 dBm, respectively. The average SNR is 4.6 dB and 0.9 dB.

5.1.4 User Data Throughput

The IoF service is made to relay fish telemetry data, hence, it should also be evaluated based on data throughput or QoS. As mentioned in section 5.1.2 the buoy in Nordfjordeid behave strange and not allowing payload sizes over 120 bytes. As it is, this limits the possible data throughput. There are a few parameters in

software that can accommodate this. First, the uplink sending interval is 4 min (240 s), based on the dynamic range of the timestamp in the IoF protocol, described in section 3.6. The relative packet timestamp has a dynamic range of 255 s and defines the maximum time interval between uplinks. However, this interval can be reduced to send uplinks more often, optimizing the data throughput. This would work for the deployment in both Stryn and Nordfjordeid, since they both mainly use SF7. Though, increasing sending frequency would also increase overhead added by LoRaWAN protocol. Hence, the total amount of bytes sent increase, power consumption increase and ToA increase. Increasing ToA can lead to duty-cycle saturation and loss of data can occur. Moreover, the buffer sizes in the software is set naively. Before deployment, the expected tag detection rate was once in a while, and not several per minute. These parameters should be optimized for future deployments.

The data throughput should also be discussed in terms of communication range and SF used. The LoRa modulation, and hence the LoRaWAN protocol, decrease maximum payload size as the SF increase. Hence, a deployment utilizing SF12 will not have the same possible data throughput as with SF7. Selecting the appropriate payload size with respect to SF is hard and some time hard to guess prior to deployment.

That suggests an IoF interface, which allow users or developers change these parameters while deployed, either manually or automatically. In addition to functional parameters, the user can filter out unwanted tag IDs to reduce necessary data throughput. The LoRaWAN protocol fully support downlinks from the gateway, allowing this type of interface. Downlinks will nor increase battery consumption on the SLIM since Class A end-devices open two receive windows after every uplink anyways.

5.2 Conclusion

The deployment result from Stryn is a proof of the IoF concept. The system relays telemetry data from the acoustic receiver with a QoS of 100 % in near real-time, and the data is presented in a graphical dashboard for monitoring. The deployment results from Nordfjordeid shows that software errors limit the supported telemetry data throughput as is. Removing these errors is possible, and the throughput will only be limited by LoRaWAN duty-cycle restrictions. Moreover, the possible throughput will decrease as the distance between the buoy and the shore-based gateway increase. The IoF service is not limited in terms of LoRaWAN communication range in likely fish monitoring scenarios, but will require gateways on high altitudes for extensible distances.

Power consumption tests shows that the battery life of a deployed SLIM could last for 206 days (around 7 months) in nominal conditions. This will vary for arbitrary deployments based on demanded telemetry data throughput and communication distance.

5.2.1 Further Work

A software update with following tests should be conducted to remove the issues concerning LoRaWAN payload size. This includes a deeper test of the LMiC to verify its functionality. By removing these issues, the potential data throughput can be increased.

A generic buoy setup up is hard to accomplish. Hence, a IoF interface would be favorable. This interface could optimize the service in arbitrary deployment scenarios. This interface should include a sophisticated ADR algorithm, tag filtering, parameter setup and system status overview.

References

[1] Moe, Trude Haugseth. "Nå er laksens utrolige reise kartlagt." (), [Online]. Available: https://forskning.no/fisk-havet-havforskning/na-er-laksens-utrolige-reise-kartlagt/1876165. (accessed: 13.06.2022).

[2] Thelma Biotel. "Transmitters." (), [Online]. Available: https://www.thelmabiotel.com/transmitter/. (accessed: 19.06.2022).

[3] ——, "TBR 700." (), [Online]. Available: https://www.thelmabiotel.com/receivers/tbr-700/. (accessed: 19.06.2022).

[4] W. Hassan, M. Føre, J. B. Ulvund, and J. A. Alfredsen, "System for Real-time Positioning and Monitoring of Fish in Commercial Marine Farms Based on Acoustic Telemetry and Internet of Fish (IoF)," in *Proceedings of the Twenty-ninth (2019) International Ocean and Polar Engineering Conference*, ©2019 International Society of Offshore and Polar Engineers (ISOPE), 2019, pp. 1496–1503, ISBN: 978-1 880653 85-2. [Online]. Available: http://hdl.handle.net/11250/2611967.

[5] W. Hassan, "Fish on the net, Acoustic Doppler telemetry and remote monitoring of individual fish in aquaculture," Ph.D. dissertation, Norwegian University of Science and Technology (NTNU), 2021, ISBN: 978-82-326-5327-0. [Online]. Available: https://hdl.handle.net/11250/2771284.

[6] P. A. Kjelsvik, "Internet of Fish, Real-time monitoring of fish through LPWAN and Internet technologies," M.S. thesis, Norwegian University of Science and Technology (NTNU), 2019. [Online]. Available: http://hdl.handle.net/11250/2625681.

[7] M. Rundhovde, "Optimization of LoRa surface buoy for underwater acoustic receiver," M.S. thesis, Norwegian University of Science and Technology (NTNU), 2020. [Online]. Available: https://hdl.handle.net/11250/2780999.

[8] E. H. Jølsgard, "LPWAN connectivity and embedded solutions for smart ocean monitoring buoys," M.S. thesis, Norwegian University of Science and Technology (NTNU), 2021. [Online]. Available: https://hdl.handle.net/11250/2828788.

[9] L. Parri, S. Parrino, G. Peruzzi, and A. Pozzebon, "A LoRaWAN Network Infrastructure for the Remote Monitoring of Offshore Sea Farms," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2020, pp. 1–6. DOI: 10.1109/I2MTC43012.2020.9128370.

[10] S. Pensieri, F. Viti, G. Moser, *et al.*, "Evaluating LoRaWAN Connectivity in a Marine Scenario," *Journal of Marine Science and Engineering*, vol. 9, no. 11, 2021, ISSN: 2077-1312. DOI: 10.3390/jmse9111218. [Online]. Available: https://www.mdpi.com/2077-1312/9/11/1218.

[11] A. Farhad, D.-H. Kim, S. Subedi, and J.-Y. Pyun, "Enhanced LoRaWAN Adaptive Data Rate for Mobile Internet of Things Devices," *Sensors*, vol. 20, no. 22, 2020, ISSN: 1424-8220. DOI: 10.3390/s20226466. [Online]. Available: https://www.mdpi.com/1424-8220/20/22/6466.

[12] K. Bullington, "Radio propagation fundamentals," *The Bell System Technical Journal*, vol. 36, no. 3, pp. 593–626, 1957. DOI: 10.1002/j.1538-7305.1957.tb03855.x.

[13] H. Friis, "A Note on a Simple Transmission Formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, 1946. DOI: 10.1109/JRPROC.1946.234568.

[14] Semtech. "What are LoRa® and LoRaWAN®?" (), [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/. (accessed: 19.06.2022).

[15] LoRa Alliance®. "What is LoRaWAN® Specification." (), [Online]. Available: https://lora-alliance.org/about-lorawan/. (accessed: 19.06.2022).

[16] The Things Network. “The Things Network.” (), [Online]. Available: https://www.thethingsnetwork.org/. (accessed: 18.06.2022).

[17] Semtech. “What is LoRa®?” (), [Online]. Available: https://www.semtech.com/lora/what-is-lora. (accessed: 19.06.2022).

[18] ——, “AN1200.22, LoRa™ Modulation Basics,” version Rev. 2, 2015.

[19] ——, “Packet Size Considerations.” (), [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/the-book/packet-size-considerations/. (accessed: 19.06.2022).

[20] The Things Network. “Adaptive Data Rate.” (), [Online]. Available: https://www.thethingsindustries.com/docs/reference/adr/. (accessed: 18.06.2022).

[21] Semtech. “Bit Rate vs. Energy Downlink.” (), [Online]. Available: https://lora-developers.semtech.com/uploads/documents/images/Energy.png. (accessed: 18.06.2022).

[22] ——, “An In-depth look at LoRaWAN® Class A Devices.” (), [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lorawan-class-a-devices/. (accessed: 19.06.2022).

[23] The Things Network. “Regional Parameters.” (), [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/regional-parameters/. (accessed: 19.06.2022).

[24] MCCI®. “Arduino LoRaWAN MAC in C (LMIC).” (), [Online]. Available: https://github.com/mcci-catena/arduino-lmic/blob/8d378ea410887d3fb08ea2c9acce95cc4c047788/doc/LMIC-v4.1.0.pdf. (accessed: 16.05.2022).

[25] MQTT.org. “MQTT.” (), [Online]. Available: https://mqtt.org/. (accessed: 19.06.2022).

[26] Norwegian Creations. “MQTT-overview.” (), [Online]. Available: https://www.norwegiancreations.com/wp-content/uploads/2017/07/MQTT-overview.jpg. (accessed: 18.06.2022).

[27] Silicon Labs, “EFM32GG Reference Manual,” version Rev. 1, 2016.

[28] *EFM32GG Data Sheet*, version Rev 2.0, Silicon Labs®, 2018. [Online]. Available: http://www.silabs.com/.

[29] u-blox. "NEO-M8-top-bottom." (), [Online]. Available: https://content.u-blox.com/sites/default/files/products/NEO-M8-top-bottom.png. (accessed: 06.06.2022).

[30] *u-blox 8 / u-blox M8 Receiver description, Including protocol specification v15-20.30,22-23.01*, UBX-13003221, version R25, u-blox, Aug. 2021.

[31] *RF96/97/98, - 137 mhz to 1020 mhz low power long range transceiver*, Hope Microelectronics, 2006.

[32] Thelma Biotel, "TBR 700 RT - Protocol," Apparently not published by Thelma Biotel, 2018.

[33] Multitech®. "MultiTech Conduit® IP67 Base Station." (), [Online]. Available: https://www.multitech.com/brands/multiconnect-conduit-ip67. (accessed: 15.05.2022).

[34] ®Canonical Ltd. "Ubuntu 18.04.6 LTS (Bionic Beaver)." (), [Online]. Available: https://releases.ubuntu.com/18.04/. (accessed: 19.06.2022).

[35] Eclipse®Foundation. "Eclipse Mosquitto™." (), [Online]. Available: https://mosquitto.org/. (accessed: 26.05.2022).

[36] OpenJS Foundation® and Node-RED® contributors. "About Node-RED." (), [Online]. Available: https://nodered.org/about/. (accessed: 19.06.2022).

[37] InfluxData®. "InfluxDB." (), [Online]. Available: https://www.influxdata.com/products/influxdb/. (accessed: 16.05.2022).

[38] Grafana Labs. "Grafana®." (), [Online]. Available: https://grafana.com/grafana/. (accessed: 19.06.2022).

[39] ©OpenStreetMap contributors. "OpenStreetMap." (), [Online]. Available: https://www.openstreetmap.org/copyright. (accessed: 19.06.2022).

[40] Silicon Labs. "Simplicity Studio." (), [Online]. Available: https://www.silabs.com/developers/simplicity-studio. (accessed: 03.06.2022).

[41] *Power Debugger*, DS40002201A, version Rev. A, Microchip®, 2020, ISBN: 978-1-5224-5953-8.

[42] Vøllestad, Asbjørn. “laks.” (), [Online]. Available: https://snl.no/laks. (accessed: 13.06.2022).

[43] Institute of Marine Research (Norwegian: Havforskningsinstituttet (HI)). “Salmon – Atlantic.” (), [Online]. Available: https://www.hi.no/en/hi/temasider/species/salmon--atlantic. (accessed: 13.06.2022).

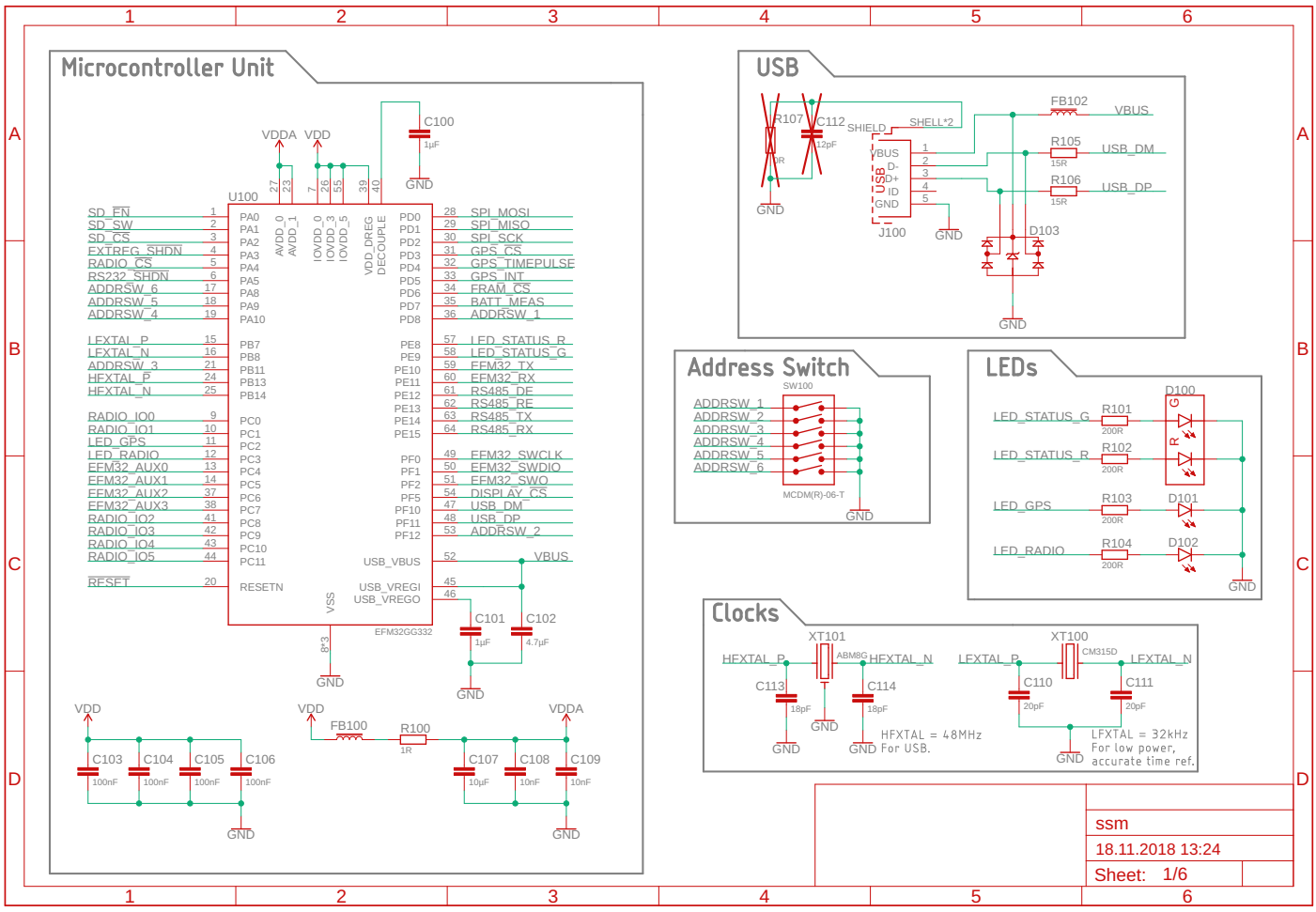
[44] Kartverket. “Norgeskart.” (), [Online]. Available: https://www.norgeskart.no/. (accessed: 19.06.2022).

[45] TAOGLAS®. “TAOGLAS.” (), [Online]. Available: https://www.taoglas.com/. (accessed: 19.06.2022).

Appendix A

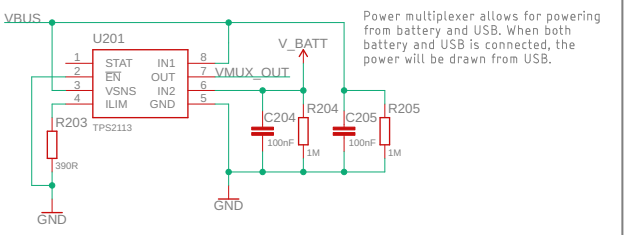
SLIM Schematics

Synchronization and LoRa Interface Module (SLIM) schematics rev. 4.

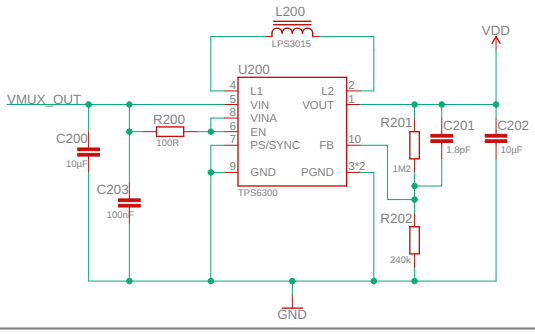


ssm	
18.11.2018 13:24	
Sheet: 1/6	

Power Multiplexer



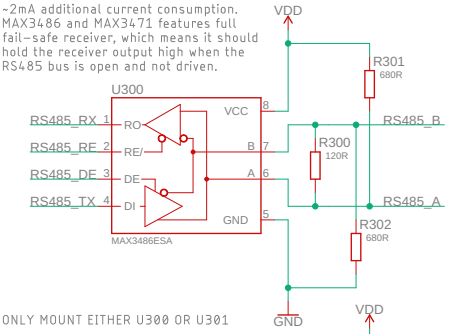
Switch-Mode 3V Regulator



	ssm
	18.11.2018 13:24
	Sheet: 2/6

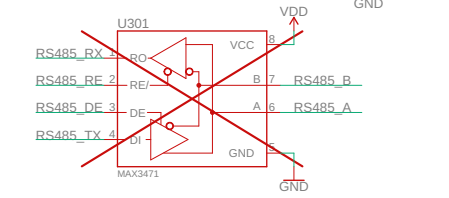
RS485

Consider leaving bias resistors, R301 and R302, unmounted. The bias resistors will cause ~2mA additional current consumption. MAX3486 and MAX3471 features full fail-safe receiver, which means it should hold the receiver output high when the RS485 bus is open and not driven.

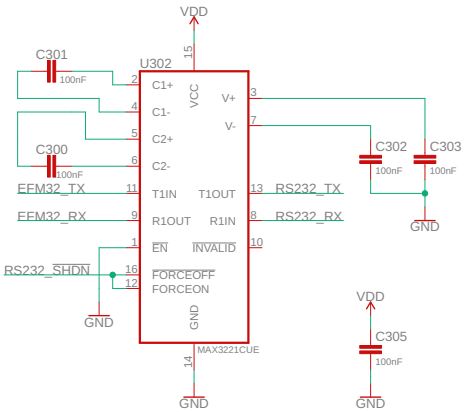


ONLY MOUNT EITHER U300 OR U301

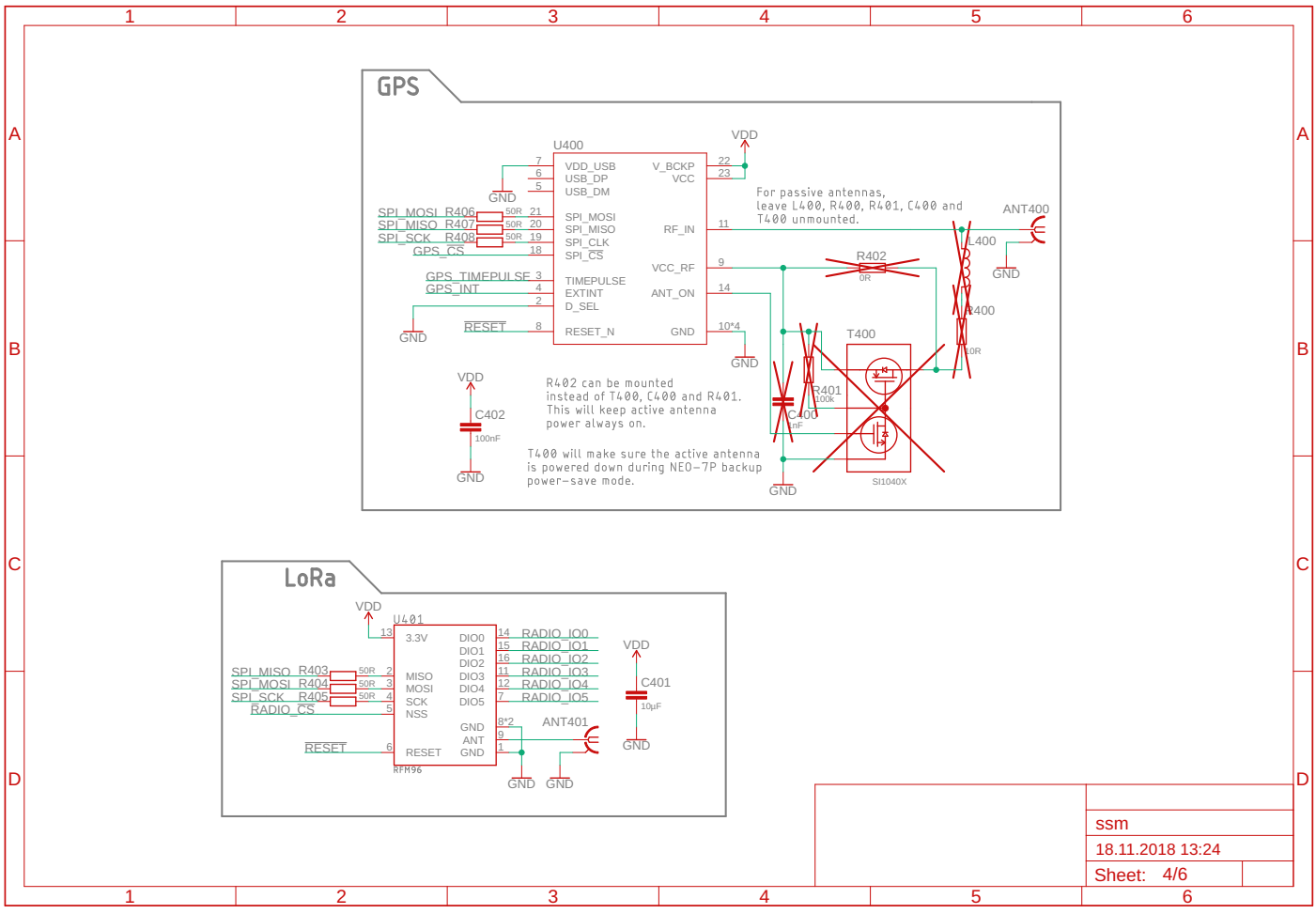
MAX3471 is a low-power RS485 transceiver, that can be powered by a 2.5V supply and has a low supply current. The downside is the maximum data rate of 64kbps. The default data rate of TBR-700 is 115.2kbps.



RS232

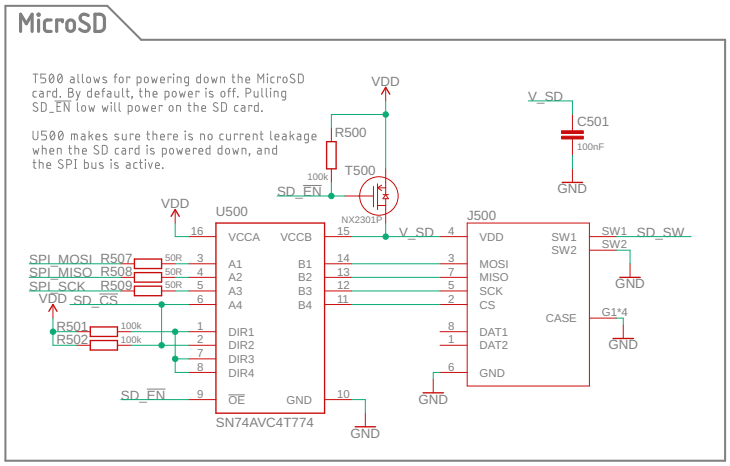


ssm	
18.11.2018 13:24	
Sheet: 3/6	



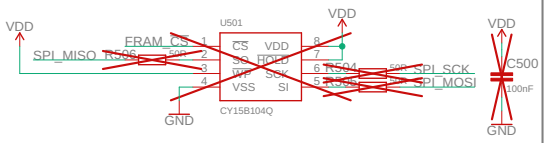
ssm	
18.11.2018 13:24	
Sheet: 4/6	

MicroSD



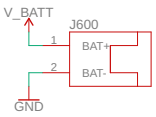
F-RAM

Ferro-electric random access memory device.

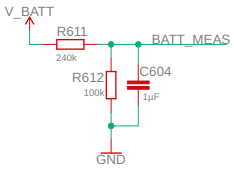


ssm	
18.11.2018 13:24	
Sheet: 5/6	

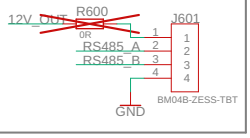
Battery Connector



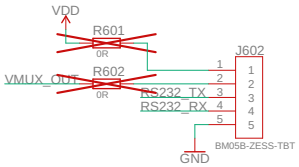
Battery Measurement



RS485 Connector

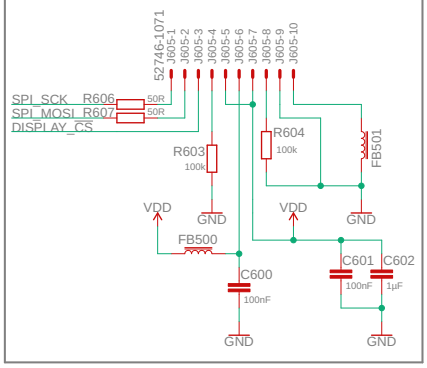


RS232 Connector

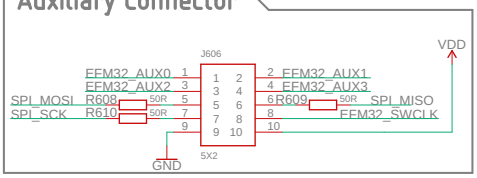


Memory LCD

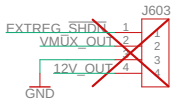
This connector will support connecting a Sharp Memory LCD (e.g. LS013B7DH03).



Auxiliary Connector

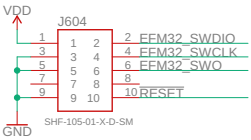


External Regulator

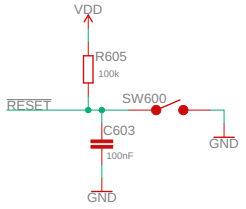


Fits external 12V regulator, e.g. S10V2F12. Can be used to power external equipment requiring 12V.

Debug Connector



Reset Button



ssm	
18.11.2018 13:24	
Sheet: 6/6	

Appendix B

SLIM Software Source Code

The SLIM source code is added as digital appendix.

Appendix C

Specialization Project Report

The specialization project "LoRaWAN Connectivity for Acoustic Telemetry Buoy" by the author of this thesis is added as digital appendix. Can also be supplied on request.

Jon Andreas Kornberg

Optimization of LoRaWAN Buoy Controller for Marine Monitoring

NTNU
Norwegian University of
Science and Technology