Design of a Multi-hop Ad-hoc Outdoor Wireless Sensor Network

by

Jonathan Bruce Wotherspoon

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering in the Department of Electrical and Electronic Engineering at the University of Stellenbosch

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Supervisor: Prof. T. Niesler
Dr. R. Wolhuter

April 2019

Declaration

I have read and understand the Stellenbosch University Policy on Plagiarism and the definitions of plagiarism and self-plagiarism contained in the Policy [Plagiarism: The use of the ideas or material of others without acknowledgement, or the re-use of one's own previously evaluated or published material without acknowledgement or indication thereof (self-plagiarism or text-recycling)]. I also understand that direct translations are plagiarism. Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism. I declare that the work contained in this assignment is my own work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

Signature: ......J.B. Wotherspoon

Date: ....April 2019

Abstract

Design of a Multi-hop Ad-hoc Outdoor Wireless Sensor Network

J.B. Wotherspoon

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MEng (EE)

April 2019

Currently the inability to monitor rhino and other endangered animal species in real time is hindering the effort to reduce poaching. This thesis proposes a multi-hop ad-hoc wireless sensor network (WSN) to deliver behavioural and positional data from an on-animal tag to a database in real-time. After practical testing, a LoRa capable radio frequency module operating at a frequency of 433 MHz was chosen. Each WSN node is power-aware and sustains its battery with energy harvested from solar radiation. A multi-hop ad-hoc routing protocol routes communication from an on-animal tag to the nearest server node. WSN nodes can detect breaks in a communication route and can request an alternative route on demand. During practical testing over a 45 km route with 3 WSN nodes, an on-animal tag mounted at rhino ear-height and ankle-height was able to successfully communicate 92.5% and 64.6% of messages respectively. A server node received 96% of communication from an on-animal tag, routed through 2 WSN nodes which were spread over 14 km. The report concludes by detailing improvements and future work to the research project before it could be commercially viable.

Uittreksel

Ontwerp van 'n Koordlose Veelsprongige Ad-hok Bewakingsnetwerk

J.B. Wotherspoon

Departement Elektries en Elektroniese Ingenieurswese,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.

Tesis: MIng (EE)

April 2019

Wildbewarings veldtogte word tans belemmer deur tegnologiese tekortkominge wat verhoed dat bedreigde spesies intyds bewaak en beskerm kan word. Hierdie tesis vervat ‘n omskrywing van die navorsing wat gedoen is om ‘n koordlose veelsprongige ad-hok bewakingsnetwerk (WSN) te ontwikkel en te implementeer wat ten doel is om liggingsinligting van ‘n op-dier-toestel intyds na ‘n databasis te versend. Na aanleiding van praktiese eksperimente is die stelsel met ‘n 433 MHz LoRa radiomodule bewerkstellig. Elke bewakingsnetwerknodus is kragstandbewus en onderhou sy eie battery met energie wat vanaf die son opgewek is. ‘n Veelsprongige ad-hok herleidingsprotokol herlei boodskappe vanaf die op-dier-toestel na die naaste bedieningsnodus. Indien ‘n bewakingsnetwerknodus ‘n onderbreking in versendingsroete identifiseer, kan ‘n alternatiewe roete aangevra word. Praktiese eksperimente is met 3 bewakingsnetwerknodusse en ‘n op-dier-toestel teen renosteroorhoogte en -enkelhoogte uit gevoer. Daar was bevind dat 92.5% van alle boodskappe teen renosteroorhoogte en 64.6% van alle boodskappe ten renosterenkelhoogte suksesvol versend is. ‘n Bedienernodus was in staat om 96% van die boodskappe wat deur twee bewakingsnetwerknodusse herlei was te ontvang. Die bewakingsnetwerknodusse was oor ‘n bereik van 14 km uit mekaar uit versprei. Die tesis sluit af met voorstelle oor toekomstige verbeteringe wat bewerkstellig sal moet word voordat die stelsel handelsgeregtig sal wees.

Acknowledgements

I would like to express my sincere gratitude to the following people:

- Prof. Thomas Niesler and Dr Riaan Wolhuter, for their continuous support, advice and going above and beyond as supervisors.
- Nathalie Mitton, for facilitating my internship at Inria, Lille in France.
- Bruce and Danielle Wotherspoon, for their love, encouragement and help with editing.
- Tracy De Allende, for her help during the practical tests and her support and encouragement throughout the writing process.
- The RhinoNet team, for supporting and sharing ideas with each other to reach a common goal.
- Nicholas Lawrenson, for many lunches where we bounced ideas around with each other.
- Arshad Vawda, for his support and enthusiasm at all the right moments.
- Philip Strydom, for helping me translate my abstract to Afrikaans.
- Wessel Croukamp and Wynand van Eeden, for their technical support and skills sharing.
- My other friends and family, for their input and feedback leading to this thesis.

Dedications

This report is dedicated to everyone who has been or still is involved in conserving the natural beauty of this earth.

Contents

List of Figures

List of Tables

Nomenclature

List of Abbreviations

ACK	Acknowledge
AODV	Ad-hoc On-demand Distance Vector
dBi	Decibel Isotropic
CSMA-CA	Collision Sense Multiple Access with Collision Avoidance
CRC	Cyclic Redundancy Check
CS	Carrier Sense
CTS	Clear To Send
DNI	Direct Normal Irradiance
GUI	Graphical User Interface
ICASA	Independent Communications Authority of South Africa
IEEE	Institute of Electrical and Electronics Engineers
INRIA	Institute for Research in Computer Science and Automation
FIT IOT-lab	Future Internet of Things Internet of Things Laboratory
ISM	Industrial, Scientific and Medical radio bands
LiPo	Lithium Polymer battery
LoRa	Long Range
MAC	Medium Access Control
MACAW	Multiple Access with Collision Avoidance for Wireless
MCU	Micro-Controller Unit
RF	Radio Frequency
RFID	Radio Frequency Identification
RSSI	Received Signal Strength Indicator
RTS	Request To Send
Rx	Receive
SPI	Serial Peripheral Interface
Tx	Transmit
UART	Universal Asynchronous Receiver-Transmitter
UHF	Ultra High Frequency
VHF	Very High Frequency
WSN	Wireless Sensor Network

Chapter 1

Introduction

South Africa is viewed as the primary custodian of Africa's rhinoceros. The Kruger National Park, South Africa's largest wildlife conservation reserve, holds the largest rhinoceros population in the world. From 2007 to 2014, the country experienced an exponential increase in rhinoceros poaching with the greatest number of cases reportedly occurring in the Kruger National Park. Covering an area of 19 485 km^2 and estimated to hold between 6649 and 7830 rhinos, monitoring these animals in the park is one of the main challenges to the prevention and mitigation of rhino poaching [1]. Current anti-poaching methods include militarising conservation through assistance from the South African National Defence force and installing counter-poaching activity teams [2]. Unfortunately, policing the park remains a challenge due to its large size.

The Predator Network (PREDNET), a research collaboration between the University of Stellenbosch and INRIA (a French public research body dedicated to digital science and technology) aims to bring engineering solutions to issues that involve the protection and monitoring of valuable natural resources and ecosystems. The collaboration is dedicated to joint research and development of wireless sensor networks as technical tools to monitor large, remote regions. RhinoNet, currently a key project of the PREDNET, aims to address rhino poaching. The project hopes to attach rhinos with electronic tags to monitor animal behaviour and position in real time. This data will allow authorities to pinpoint locations at which poaching may be about to occur and thereby shorten response time. The device, known as an on-animal tag, uses a vibration sensor, accelerometer and low-power algorithms to classify rhino behaviour. The data from the on-animal tag must be wirelessly communicated to a server. As part of RhinoNet, a wireless sensor network (WSN) is needed to deliver data from the on-animal tags to this server.

The work documented in thesis focused on the development of the first prototype of this WSN, with a main objective of delivering data from the on-animal tags to a database.

1.1 Project Objectives

The objectives of this project were set out as follows:

- Investigate previous work done in related fields.
- Propose and define an engineering solution for the development of a WSN applicable to rhinoceros and related conservation.
- Determine a suitable operating frequency and RF module for use in the WSN.
- Design and develop hardware for such a WSN node.
- Design and develop a power supply unit for the WSN node.
- Design and implement a multi-hop ad-hoc WSN routing protocol.
- Practically test the system and evaluate the acquired measurements and results.
- Suggest improvements to the hardware and routing protocol based on the results of the practical tests.

1.2 Project Outcomes and Contributions

The project realised the following outcomes and contributions:

- Experimental evaluation showed that a LoRa RF module operating at a frequency of 433 MHz was a suitable choice for this project.
- Hardware for a prototype WSN node was designed and built, which incorporates a low power MCU, the chosen RF module and other modules.
- A power supply for the WSN nodes was designed and built, which incorporates battery monitoring, solar charging and voltage regulation.
- An ad-hoc multi-hop wireless routing protocol was designed, and tested with all communication received from an on-animal tag routed to the nearest server node.
- A feature addition to the MAC protocol for the on-animal tags, called 'address saving', increased messages received by 6.71%, and decreased timeouts and overheard packets on the WSN nodes by 94.7% and 93.1% respectively.
- An addition to the MAC protocol for the WSN nodes, added a delay before a node replies to a broadcast RTS packet. This resulted in a 22.2% increase in received messages, and a 97.8% decrease in the number of retransmissions required.
- Practical tests showed that the maximum distance which two WSN nodes could be separated and still communicate was 16.6 km.
- Communication from an on-animal tag to three WSN nodes spread over 45 km was tested, and with the on-animal tags' antenna set to 1.6 m and 0.2 m above the ground, 92.5% and 64.6% of messages were successfully received respectively.
- A server node received 96% of messages sent from an on-animal tag and routed through two mesh nodes which were spread over 14 km.

1.3 Structure of this Thesis

The thesis is structured as follows: Chapter 2 presents previous work done in similar fields. Chapter 3 considers the requirements to develop the system of WSN nodes described in this chapter. Chapter 4 investigates an operating frequency and RF module for use in the WSN by means of practical tests. Chapter 5 covers the hardware selected for use in the WSN node as well as the power supply. Chapter 6 details the choices made for the WSN routing protocol. Chapter 7 describes additional software created to support the use of the WSN nodes by a user. Chapter 8 analyses measurements and results from field tests, and conclusions and recommendations are given in Chapter 9.

Chapter 2

Literature Study

In this chapter, previous studies which implemented wireless sensor networks (WSNs) to monitor the behaviour of animals will be discussed. The solutions these projects implemented will be described and problems described by their authors reviewed. Further literature review relating to the design of a specific function of the WSN, will be discussed where it is most relevant in this thesis.

A project which had similar ambitions to that of the RhinoNet is the ZebraNet [3]. In this project, a collar was placed around the neck of a zebra to allow the position of the animal to be tracked. The positional data from the animal was stored in flash memory until it came into contact with another animal also equipped with a networking collar. The two collars exchange their data in the hope that one of them will reach a base station and be able to offload data from both animals to the station. The flash memory on each collar was large enough to store 26 days worth of positional data for the host animal, and 52 days from other collars it may come into contact with. The designers of the hardware in this project noted the need to ensure that the power supplied to each component be as noise free as possible. They also noted that the radio module chosen, which had an output power of 21.5 dB m at an operating frequency of 900 MHz, was only capable of communicating a distance of 900 m when deployed during field tests.

A study performed in 2004 attempted to monitor the position of sheep when they were let out to graze in Norway [4]. The flock mentality of the animals was used to advantage by mounting a GSM modem (known as the mobile access point) to a leader sheep, and mounting ultra high frequency (UHF) radio tags to the other sheep in the flock. The radio tags, with an operating frequency of 433 MHz were able to communicate over a distance of 100 m, and would attempt to communicate their ID number to the mobile access point every minute. The mobile access point would receive communication every 3 hours from the UHF tags, and take a GPS measurement. The data was uploaded from the mobile access point every 12 hours to an online database. The study found that the lack of noise suppression circuitry on the power supplied to the RF modules resulted in very unstable links and reduced range. The study further noted that the batteries chosen were insufficient to power the nodes for extended periods of time.

A study carried out in 2009 attempted to combine data collected from on-animal nodes and satellite imagery to track the movement and grazing patterns of cattle in Australia [5]. The on-animal nodes communicated their GPS data via a UHF radio to a WSN. The UHF radio has an operating frequency of 915 MHz, and with the networking protocols selected for the project and the required sampling rate of 4 Hz, only 5 on-animal nodes were allowed per a network. The study also implemented a radio link check to ensure efficient data transmissions by only communicating if the link quality was above a certain threshold.

In 2010, a WSN was deployed to monitor European badger (Meles meles) in their natural environment [6]. The network was made up of 3 parts: the Radio Frequency Identification (RFID) collars placed around the neck of the badgers, a collection of fixed nodes to receive data from the collars, and a set of nodes which monitored micro-climatic conditions such as air and soil temperature. Certain of the fixed nodes implemented a 3G modem to upload data directly to an online database, while the others simply stored data locally until it was physically collected. The study made use of adaptive sensing, to learn when the animals were most active near certain nodes and to adjust the duty cycle of the RFID reader for those periods. Compared to a node which had a duty cycle of 100%, the adaptive sensing nodes were able to detect 73% of the sightings, and only expend 8.2% of the energy used by the always-on node. The study further assigned classes to each measure to classify the priority of the message. Priority 1 data (highest priority) was sent to a node with a 3G modem which uploaded the received messages immediately. Priority 2 data was sent to a 3G node and was only uploaded within a few days. Priority 3 data (lowest priority) was not communicated but rather saved to local storage on the node and could only be accessed directly from the memory of the node.

A study conducted in 2010, proposed a WSN to monitor whether the migration patterns of wild animals were affected by a newly built road [7]. The WSN consisted of two types of nodes, a camera node and a sensor node. All the nodes used the commercially available Imote2, with an infrared motion detector. The camera nodes incorporated extra flash storage and a camera module which could take still images. Three camera nodes were placed at the entrance of a tunnel underneath the newly built road, while the sensor nodes were placed in a 2.5 ha semicircular area around the tunnel entrance. When the infrared motion detector on a sensor node detected movement, the node would broadcast a message to the camera nodes. The neighbouring nodes, would repeat this message to the camera nodes. Once the message was received by the camera node, all future messages received about the same animal sighting would be ignored. The study noted that the IEEE 802.15.4 MAC protocol, which was preloaded on the Imote2 nodes, did not satisfy their requirements for ultra-low power consumption. This was due to the long active periods which the MAC protocol requires to resolve medium access. The authors therefore implemented the Collision Sense Multiple Access with Collision Avoidance (CSMA-CA) MAC protocol.

A further study carried out in 2010 had the aim of tracking endangered Eurasian Lynx (Lynx lynx) migration in Latvian forests [8]. The study used the commercially available TMote mini sensor nodes with a TI MSP430 micro-controller and a RF module operating at 433 MHz. The sensor nodes were placed on the animals, and would communicate directly to nodes set up as a base station. The project used a CSMA-based MAC protocol, which it justified by saying that the probability of a packet collision during communication is small due to the large spaces covered. The base stations were continuously in receive mode, with communication initiated from the animal-borne nodes. During practical tests, the study found that the chosen RF module and MAC layers were inefficient for communicating further than a distance of 250 m.

This study from 2011 deployed six wide-band directional antennas on top of 40 m towers to monitor animal behaviour in a forest [9]. The animals were fitted with small very high frequency (VHF) transmitters which periodically emitted frequencies between 148 MHz and 168 MHz. All received signals were captured from the directional antennas, and transmitted to a database using an RF module operating at 900 MHz. The recorded data was processed at a later stage to estimate where the animal had been, based on the received signal strength recorded by each directional antenna. If three or more towers recorded a signal from the same animal, triangulation could be used to accurately determine the location of the animal. With towers spaced 800 m apart, the system was able to accurately track the position of larger terrestrial animals. The authors indicated that by utilising solar charging at each tower, they were able to allow for uninterrupted use of the equipment.

A study in 2012 proposed to monitor the behaviour of sheep by attaching an animal borne device with an accelerometer to each sheep [10]. The WSN was made up of several commercially available Iris motes with a Zigbee RF module operating at 2.4 GHz. A base station was stationed at the edge of the field and it was the end destination for all communication. Two relay nodes, positioned at the far ends of the field, sent received data directly to the base station. The animal borne devices formed an ad-hoc mesh network and would route data until it reached a relay node or the base station. The network utilised an acknowledge packet, sent from the base station to the originating node to inform the node that its data had been successfully received at the base station. Acceleration data was transmitted from each animal borne device at one second intervals to the mesh network, which would send it to the base station. The study noted that the relay nodes were very useful in alleviating traffic from the animal borne devices, however it noted that traffic at these nodes was far higher than any other nodes, leading to 50% of messages needing to be retransmitted before arriving at the relay nodes.

A study produced in 2018 used an accelerometer and machine learning in an animal borne device to classify the behaviour of cows in real time [11]. The authors used a decision tree methodology to classify the measurements from the accelerometer into 3 behaviours: standing, walking and lying. Communication was only initiated from the animal-borne device to the server node when a change in behaviour was detected. The commercially available Zigbee RF module used in the animal borne devices and the server node made use of the IEEE 802.15.4 communication standard for the physical and MAC layers. The IEEE 802.15.4 standard was designed specifically for low-rate wireless personal area networks and made use of a contention period, during which nodes use CSMA-CA to communicate with a central node, and a contention free period, during which nodes requiring a higher bandwidth are assigned specific times to communicate to the server node [12]. The study from 2018 found that the range of communication of their chosen RF module was 500 m with line of sight. The study notes that the lack of extended operating time of the animal-borne device was due to poor power management and small battery capacity.

Much of the experience fom the authors discussed in this chapter can be used in our WSN. Below is listed five points which will be taken into account when designing our WSN.

- The importance of a noise free power supply to the RF module to increase communication range.
- The successful use of solar energy to maintain charge in the batteries.
- The node should be power aware, and be able to enter a sleep mode if the battery reaches a low level.
- The importance of choosing an operating frequency and RF module which is specific to the needs of the WSN.
- MAC and routing protocols should be chosen which are adapted specifically to the requirements of the network.

In this chapter several studies were reviewed in which a WSN was deployed to monitor animal behaviour. In the following chapter the design requirements for our system will be considered and an engineering solution proposed.

Chapter 3

Design Requirements and System Formulation

In this chapter, the project design requirements and proposed solution will be discussed.

3.1 Project Requirements

The following section details the communication specifications of the on-animal tag, as well as challenges expected due to the environment in which the proposed solution will be deployed.

An aim of the RhinoNet project is for the on-animal tag to monitor the behaviour and position of an animal in real time. It has strict power requirements and therefore, during periods of normal animal activity, the tag is expected to transmit data at intervals of between 90 seconds and 30 minutes. If the tag detects a crisis situation, it will communicate its current position immediately, and this message should be treated as time critical by the proposed solution.

The antenna of the RF module is situated within the on-animal tag enclosure, which is mounted on the ankle of the animal as shown in Figure 3.1. This mounting was chosen by the designer of the on-animal tag. This poses a significant challenge for the RF communications, as the position of the antenna is low and close to the ground which will result in increased obstruction and reflection of the RF signal.

The data transmitted by the on-animal tag will vary depending on the situation. However, a message length of between 1 and 100 bytes is expected. Furthermore, the tag will only send data and will not receive communication.

The proposed solution must also consider the challenges posed by the outdoor environment in which the system will be deployed. The animals are free to roam in large uninhabited areas, which in the case of the Kruger National Park, can be several thousand square kilometres. The proposed solution will therefore need to be able to cover extensive geographical areas.

Most wildlife habitat is situated in rural areas, where there is little to no GSM coverage. The proposed solution will require the capability of sending data from an area without

GSM coverage to an area where the data can be uploaded.

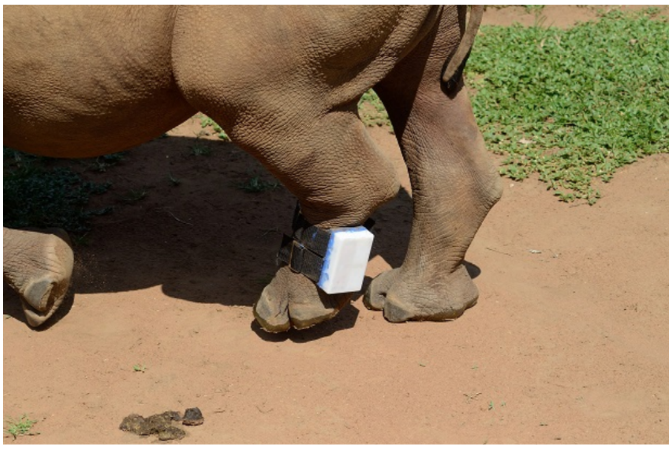


Figure 3.1: Animal tag mounted on the ankle of a rhino. Reproduced from [13].

In addition, there may be no formal electricity supply in most of the area to be considered. The proposed solution will therefore need to be energy self-sufficient .

With the requirements of the system set out, we can propose an engineering solution.

3.2 Proposed Solution

Due to the relatively low number of messages, and small amount of data sent during each message from an on-animal tag, the proposed system will not have to focus on ensuring high throughput. Furthermore, because the animals are able to roam over large distances, the communication from an on-animal tag may need to take place over extended ranges. Several of the studies discussed in Chapter 2 found that the choice of operating frequency and RF module had a large impact on the range over which their system could communicate [3, 4, 8, 11]. Therefore, an operating frequency and RF module will need to be chosen, with a focus on long range rather than high bandwidth.

Because GSM coverage is expected to be intermittent, the proposed solution will only be able to place nodes with a GSM modem in areas where there is GSM signal. These nodes will be called server nodes and all data received by these nodes will be uploaded to a database. Repeater nodes will be used to pass data from the on-animal tags to the server nodes. The repeater nodes (henceforth referred to as mesh nodes), will create a multi-hop network, through which data can be sent to the server nodes.

The probable lack of a formal electricity supply, dictates that the nodes must be battery powered. This is also important as the nodes should be fully portable. Several studies in Chapter 2 indicated that solar harvesting was effective in extending the working life of their system [3, 9]. Therefore, in order to be self sustaining, the nodes will utilise solar energy harvesting to recharge the batteries. Furthermore, several studies discussed in Chapter 2 indicated the need for the WSN nodes to be able to enter a power-saving mode if the battery charge falls too low [6, 7, 11]. Therefore, the proposed solution will be power aware and able to switch between several power-saving modes when required.

A node should also be able to communicate to its neighbours in the network if it must shut down completely in order to preserve battery capacity. Due to the possibility of a node leaving the network without informing its neighbours (for example if it is destroyed by a wild animal or sabotaged) the network must be able to detect a break in the communication route to a server, and to find an alternative route without user input.

A summary of the proposed solution is shown in Figure 3.2, and can be described as a multi-hop ad-hoc network with server nodes and mesh nodes. Any node is capable of receiving communication from an on-animal tag, and all data will be sent to the nearest server node. The server node will either upload the data to an online database via GSM, or send it to a local storage device via UART. The nodes will be battery powered, and will incorporate solar energy harvesting to extend battery life. Each node is capable of detecting a break in the network and able to find a new route to a server without user input.

While the proposed WSN uses similar building blocks to those discussed in Chapter 2, our proposed system is expected to be deployed over extended distances rather than in a confined space. Furthermore, while most of networks described in the previous studies used commercially available WSN nodes to form their networks, our project will design the hardware of the WSN nodes, with low-power usage and extended communication range as the main priorities.

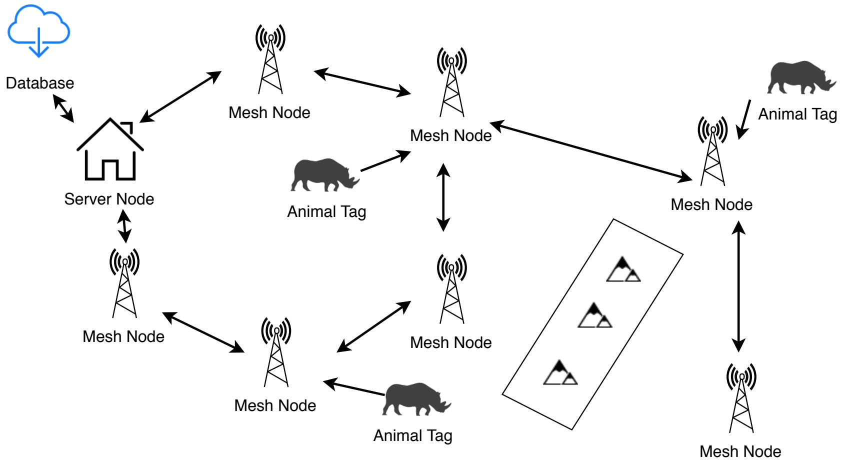


Figure 3.2: The proposed WSN to deliver data from the on-animal tags to a database. The network will route the data via the mesh nodes to a server node which can upload it to a database.

3.3 Summary and Conclusion

In this chapter, an ad-hoc wireless sensor network (WSN) was proposed as a way to deliver data from the on-animal tags to a server or database. The WSN consists of mesh nodes, which form the links in the multi-hop network, and server nodes, which are the endpoints and can upload the data to its desired destination.

In the next chapter practical tests will be done to determine the operating frequency and subsequent RF module which will form the basis for communications of the WSN.

Chapter 4

Choosing an Integrated RF Module

In order to facilitate the development of a large-scale outdoor WSN, a low-power radio transceiver is needed which is capable of long range communication while remaining small and utilising little power during transmission. The radio will not only need to accomplish WSN node-to-node communication, but also facilitate the extremely important on-animal tag to WSN node communication. It is accepted that communication with the on-animal tag will be the most challenging, as this device is expected to be mounted very close to the ground, have low-gain antennas due to packaging constraints and can find itself transmitting from difficult terrain such as deep shrubbery. The large expanses of park in which the animals are free to roam, will also mean that many WSN nodes will be needed to ensure sufficient coverage. Hence it is imperative for the nodes to have as extensive a communications range as possible.

Therefore the chosen radio should have the following attributes:

- It must provide a reliable link between nodes with minimal data transmission errors.
- It must communicate over long distances, typically more than 2 km, to limit the number of nodes needed by the network.
- It must offer a sleep mode with low current consumption.
- It must have an operating frequency in an ISM band.
- It must have a maximum transmit power of 100 mW or less and abide by ICASA regulations [14].

This chapter will discuss the process of choosing an integrated RF module for use in the WSN. First, the most appropriate operating frequency at which the radios should transmit will be evaluated by means of practical measurements. Then several radios will be evaluated, first in terms of the specifications listed on their datasheets and second in terms of practical tests done to measure communication distance, accuracy of communication and dropped packets for each device. Finally a radio module will be chosen for use in this project.

4.1 Choice of Operating Frequency

The first decision to be made will be with regards to the operating frequency at which the RF module will communicate. The operating frequency will influence the antenna size, power requirements for communication over a certain distance, and the cost of required materials. This section will firstly identify possible frequencies for use in the WSN, after which practical measurements will be taken to determine which frequency offers the best communication between the on-animal tag and WSN node and finally an operating frequency will be chosen.

4.1.1 Possible ISM Bands

There are a few operating frequencies which are available for public and amateur use within guidelines, meaning that no license is required to operate within those bands. These free bands are called the Industrial, Scientific and Medical research (ISM) frequency bands and are available in South Africa and all region 1 countries at the following frequencies: 40.66 MHz, 433 MHz, 2.4 GHz and 5.725 GHz. There are also plans to incorporate 169 MHz and 61 GHz in the near future [15]. See Appendix A for a map indicating all region 1 countries.

The reason for choosing one of the ISM frequencies for this project is so that the system will not be limited to operation in a single country by frequency licensing issues. Furthermore, there are a large number of commercially available RF modules which operate at these ISM frequencies.

Previous studies which implemented a WSN operating at 915 MHz and above indicated that they were not able to achieve the minimum 2 km communication range between modules which is required for this project [3, 16]. This lack of range is most likely due to the fact that higher operating frequencies are less able to penetrate or pass around obstacles such as vegetation [17].

The Friis equation, shown in Equation 4.1.1, indicates that better propagation properties may be expected at lower frequencies. The Friis equation states that the power received (P_r) at one antenna is equal to the transmitted power (P_t) multiplied by the gains of both antennas (G_r, G_t) and the wavelength (λ) of the transmitted signal divided by the distance (r) between antennas.

$$P_r = P_t \frac{G_r G_t \lambda^2}{(4\pi)^2 r^2} \tag{4.1.1}$$

Consider a transmitted power P_t at a wavelength λ_1 and a received power P_r at distance r_1 from the transmitter. For the same power to be received at a different wavelength λ_2 the distance of separation r_2 must satisfy Equation 4.1.2.

$$P_t\frac{G_rG_t(\lambda_1)^2}{(4\pi)^2(r_1)^2} = P_t\frac{G_rG_t(\lambda_2)^2}{(4\pi)^2(r_2)^2} \tag{4.1.2}$$

$$\frac{(\lambda_1)^2}{(r_1)^2} = \frac{(\lambda_2)^2}{(r_2)^2}$$

$$\frac{\lambda_1}{\lambda_2} = \frac{r_1}{r_2} \tag{4.1.3}$$

Here we have assumed that the gain of the transmitting antenna is the same at λ_1 and λ_2, and the same for the receiving antenna.

From Equation 4.1.3 we see that a signal with a lower frequency is less attenuated by distance in free space. Therefore the ISM frequencies at or above 915 MHz will not be considered for this project. The three ISM frequencies that fall below this threshold are 40.66 MHz, 169 MHz, and 433 MHz. Very little hardware is available for use in the 40.66 MHz band and the equipment that is available tends to be very large and cumbersome. Therefore, this frequency will also not be considered further. The following sections consider specifically the ISM frequencies 169 MHz and 433 MHz.

4.1.2 Antenna Size

Due to the robust environment in which wild animals live, the on-animal tag must fit within the device enclosure. A study which considered collars that can be attached to rhino found that the enclosure should not exceed 40 mm by 60 mm in size [18]. With this in mind, we will consider the length of a simple half wave antenna given by Equation 4.1.4.

$$\lambda = \frac{c}{2 * f} \tag{4.1.4}$$

From Equation 4.1.4, a half wave antenna for 169 MHz and 433 MHz will have a length of 0.88m and 0.34m respectively. Both exceed the dimensions of the envisaged enclosure and therefore more specialised designs will need to be considered in both cases. It will however remain true that, a corresponding antenna for 169 MHz will have a length more than double that of the 433 MHz design. On the other hand, according to Equation 4.1.3, the distance which the 169 MHz signal will be able to propagate in free space is 2.56 times greater than the distance of a 433 MHz signal. This benefit may not all be achievable in practice however, since signal propagation will not take place in free space, but close to the ground, among vegetation. As a next step we therefore perform practical measurements to determine whether the practically measured increased gain, due to the lower frequency is indeed so great.

4.1.3 Experimental Procedure

Practical measurements were done to verify whether using the lower frequency provides sufficient benefit in terms of signal propagation to justify the added complexity associated with a larger antenna in the confines of the permissible enclosure dimensions. The experiments were designed to test the following conditions:

- Whether there is an improvement in the received signal strength when operating at 169 MHz instead of 433 MHz.
- How raising the receiving antenna affects the received signal strength at each frequency.

For the experiments the following set up was used. A wide band log-periodic dipole array antenna, set up as the receiving station, was attached to a signal analyser. This antenna could be adjusted to heights of 0.63m, 1m, 2m, 3m and 3.5m above the ground. A second antenna, set up as the transmitting station, was attached to a portable signal generator, and was fixed at a height of 0.1 m above the ground. This antenna height corresponds to the estimated height of the on-animal antenna when attached to the ankle of an animal [13]. The transmitting station was placed at distances of 100m, 300m, 350m and 500m from the receiving station. Together the antennas had a gain of -2.361 dBmı at 169 MHz and -1.266 dBmı at 433 MHz.

At each separating distance a 25 dB m signal was generated, first at 169 MHz and then at 433 MHz, and the received signal strength measured using the signal analyser. This was repeated for each height of the receiving antenna. The measurements were adjusted for the gains of the antennas at each frequency to allow a direct comparison. A summary of the test setup is given in Table 4.1.

Table 4.1: Summary of the test setup used to compare signal propagation at 169 MHz and 433 MHz.

Properties	Values used for test
Tx antenna height	0.1[m]
Rx antenna height	0.63 ; 1 ; 2 ; 3 ; 3.5 [m]
Tx output power	25 [dB m]
Antenna gain (169 MHz)	-2.361 [dBi]
Antenna gain (433 MHz)	-1.266 [dBi]
Test distances	100 ; 300 ; 350 ; 500 [m]
Frequencies tested	169 ; 433 [MHz]

4.1.4 Testing Location

The Jan Marais park in Stellenbosch was chosen as the testing location. The park allows for a testing range up to 500 m without any man-made obstructions or elevation changes. The park contains approximately 1 m tall foliage growth, which approximates the habitat of wild animals in which the system will be used. Figure 4.1 shows images illustrating the foliage found in the Jan Marais Park.

4.1.5 Experimental Results

Table 4.2 shows the average, over all distances, of the received signal strength at each receiving antenna height. The table also shows the difference between the received signal strength at 169 MHz and 433 MHz. The total change is the difference in received signal strength between the received signal strength at a receiving antenna height of 0.63m and 3.5m.

Figure 4.1: Example of the foliage in the Jan Marais park in Stellenbosch.

Table 4.2: Average measured received signal strength for each frequency over all distances.

Rx Height [m]	169 MHz [dB m]	433 MHz [dB m]	Difference [dB]
0.63	-74.64	-78.11	3.47
1	-72.51	-76.36	3.85
2	-69.39	-72.61	3.22
3	-68.01	-70.48	2.47
3.5	-66.64	-68.48	1.85
Total Change	8	9.64	

Frequency vs Signal Strength

The results in Table 4.2 confirm that, for all receiving antenna heights, the measured signal is stronger at 169 MHz than at 433 MHz. However, as the receiving antenna height is raised from the ground, this benefit decreases. At a height of 0.63 m the difference is 3.47 dB, and at a height of 3.5 m the difference is only 1.85 dB. Figure 4.2 shows how the difference in received signal strength decreases as the height of the receiving antenna is increased. The figure also shows the change in received signal strength at an operating frequency of 433 MHz.

This decrease in the difference can be explained by considering the Fresnel zone at each frequency and the manner in which the surrounding environment affects the transmitted signal. According to the Fresnel equation, the radius (r in meters) at its widest point is given by Equation 4.1.5, with d equal to the distance between links (in km), and f the frequency (in GHz).

$$r = 17.32 * \sqrt{\frac{d}{4f}}\ meters \tag{4.1.5}$$

If the Fresnel zone between transmitter and receiver is clear of any objects such as shrubs, trees or hills then the signal will be unobstructed. However, if there are obstructions, the

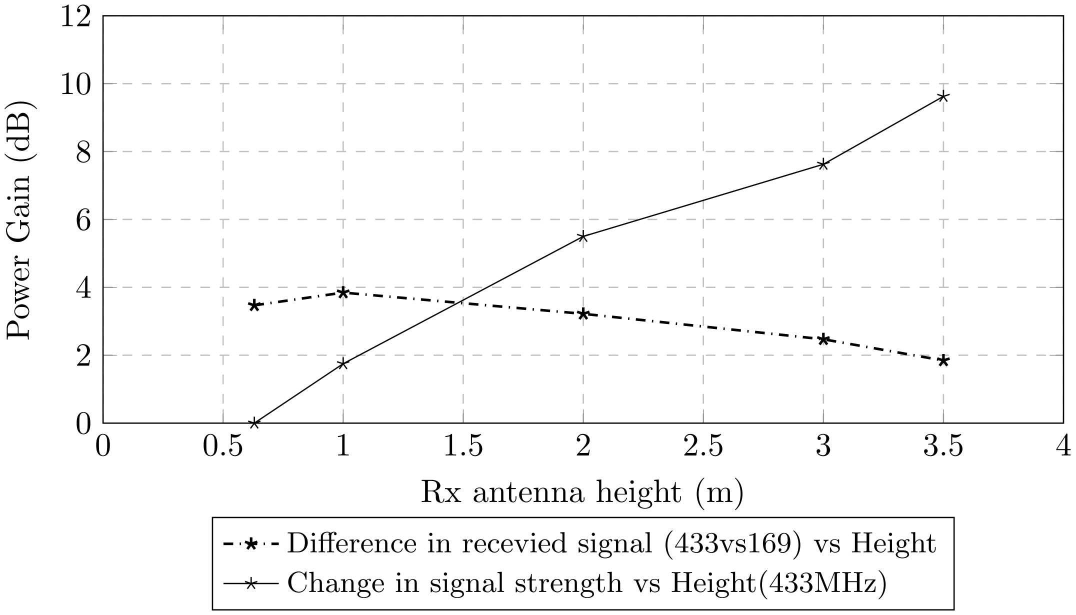


Figure 4.2: Received signal strength difference between 169 MHz and 433 MHz as a function of antenna height as well as the change in received signal strength for 433 MHz as a function of antenna height.

signal may be reflected by the obstruction and arrive at the receiving antenna with a shifted phase, reducing received signal strength.

As the distance between transmitter and receiver increases, or the frequency is lowered, the maximum Fresnel zone radius increases [19]. Figure 4.3 demonstrates this principle. In the figure, the distance between the receiving and transmitting antenna is kept constant, therefore, the only factor affecting the Fresnel zone size is the frequency. The Fresnel radius is smaller for 433 MHz than for 169 MHz. Therefore, it is possible that obstructions in the communication path will have a greater effect at 169 MHz than at 433 MHz.

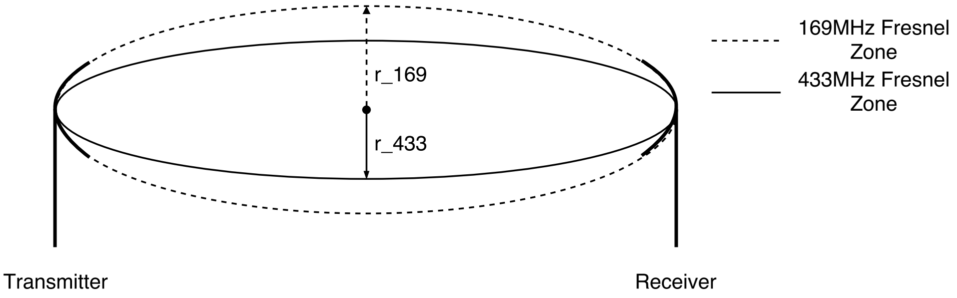


Figure 4.3: The Fresnel zones for 169 MHz and 433 MHz. The zone is larger at 169 MHz due to the longer wavelength. If there is an obstruction within the Fresnel zone then the received signal strength will be affected. The maximum Fresnel radius for the 169 MHz signal is denoted by r_169, while the maximum Fresnel radius for the 433 MHz signal is denoted by r_433.

Antenna Height vs Received Signal Strength

Table 4.2 also shows that, as the receiving antenna is raised from 0.63 m to 3.5 m, the measured signal strength increases by 8 dB m for the 169 MHz signal and 9.64 dB m for the 433 MHz signal.

This increase in received signal strength can be attributed to fewer obstructions in the Fresnel zone. As the receiving antenna is raised, a smaller proportion of the Fresnel clearance is obstructed by foliage for both frequencies. The 433 MHz signal experiences a larger increase in received signal strength as its Fresnel zone is narrower than the 169 MHz Fresnel zone, and as the antenna height increases the signal path becomes less obstructed for the 433 MHz signal. For the 169 MHz configuration, the antenna is possibly not raised high enough to experience the same improvement in received signal strength, as its clearance zone is still obstructed by objects which no longer obstruct the 433 MHz signal.

Therefore increasing the height of the receiving antenna, even by a few meters, makes a large improvement to the received signal strength for both frequencies. Figure 4.2 shows this increase in received signal strength as the antenna height is increased. This improvement overshadows the increase in signal strength achieved by lowering the communication frequency from 433 MHz to 169 MHz.

4.1.6 Conclusion

The results of our experiments indicate that the received signal strength is higher for 169 MHz than it is for 433 MHz, but that difference however, is small. At its maximum it is 3.85 dB, while when the receiving antenna height is increased, this difference decreases to 1.85 dB. The tests also indicate that raising the receiving antenna from 0.63 m to 3.5 m has a dramatic positive effect on the received signal strength. At 433 MHz a 9.64 dB improvement was measured when moving the receiving antenna from its lowest position to its highest.

We conclude that the gains achieved by using the lower 169 MHz frequency is very small, and are far outweighed by increases in received signal strength achieved by raising the height of the receiving antenna. Taking into account the higher antenna complexity needed at the lower frequency, it was judged that 433 MHz is the more appropriate operating frequency choice for use in our outdoor wireless sensor network.

4.2 Choice of Radio Device

After deciding on an operating frequency of 433 MHz, a suitable radio device to facilitate communication between nodes needed to be identified. The radio module will be used in both the WSN as well as the on-animal tags. This lowers the cost of the network and lessens the complexity of the nodes. The radio module must therefore be suitable for both communication between WSN nodes as well as between on-animal and WSN nodes.

4.2.1 Which Chips to Test

The on-animal tags are battery powered and therefore have a very limited supply of energy. Therefore, it is important for the RF module to use very little power during sleep mode as well as to minimise power during transmission. A compromise will need to be made between power usage and output power for the chosen RF module.

Table 4.3 lists 6 radio modules which were considered for use in this project. These modules are all commercially available, have an external SMA connector for connection to a 50 Ω antenna and have a minimum output power of at least 10 dB m. The chosen devices interface with a micro-controller (MCU) using either SPI or UART communication. They all have a specific instruction set written by the manufacturer which can be used to configure the radio.

Table 4.3: Key characteristics of the 6 considered radio devices

Device	CC1101	SI4463	RFM96W	RFM22B	RFM23BP	E31-TTL
Vendor	TI	SI	HopeRF	HopeRF	HopeRF	EByte
Tx power [dB m]	+10	+20	+20	+20	+30	+30
Sensitivity [dB m]	-112	-120	-148	-121	-114	-126
Current Tx [µA]	29.2	88	120	85	550	510
Current Rx [µA]	16	13.7	12.1	18.5	18.5	15.5
Supply [V]	3	3.3	3.3	3	5	5
Modulation	FSK	FSK	LoRa	FSK	FSK	FSK
Sleep mode current [µA]	0.12	1	0.2	1	1	1.7
Interface	SPI	SPI	SPI	SPI	SPI	UART

The TI CC1101 radio module was chosen for use in the first prototypes of the on-animal tags in the RhinoNet project [13]. This chip has a maximum transmit (Tx) power output of 10 dB m, uses 29.2 mA during a transmission and 0.12 µA during sleep mode. During previous field tests conducted by a member of the RhinoNet team, it was only possible to achieve a communication range of 1 km, when placing the receiving antenna approximately 6 m above ground and the transmitting antenna 2 m above the ground. This low-power radio, is ideal for an extremely low power WSN situated indoors, or where nodes are not far apart. For our application however, a radio with a longer range of communication is required, therefore the TI CC1101 was not investigated further.

Table 4.3 also indicates that a family of chips was considered, namely the RFM22B and the RFM23BP from HopeRF. These devices share the same communication protocol and are therefore able to communicate with each other [20]. In our application this pair would have the advantage that the higher power RFM23BP could be used in the stationary WSN nodes where power consumption is less critical, while the lower power radio could be used in the on-animal tags where power conservation is critical. The RFM22B is almost identical to the higher power RFM23BP, but the RFM23BP has a power amplifier on the antenna output which boosts the output power to 30 dB m. This boost however also results in a loss of sensitivity for the higher power module.

The SI4463 radio module has almost identical specifications in output power, sensitivity and Tx current draw to the RFM22B, while the E31-TTL-1W has Tx current draw and output power similar to the RFM23BP.

As a fifth option, a Long Range (LoRa) capable radio was included among the candidates. This was done after Mr Jaco Swart, a final year engineering student in 2016, did some research into the capabilities of this new type of network protocol, namely LoraWAN. LoRa has been claimed to achieve far greater communication ranges than conventional On-off keying (OOK) and Frequency-shift keying (FSK) based radios [21]. LoRa achieves this greater range by trading data rate for sensitivity and by increasing the transmitted signal bandwidth within a fixed channel bandwidth [22]. The LoRa signal is modulated by a patented stable chirp, created using a fractional-N phase locked loop [23]. It is claimed that LoRa radios are capable of efficiently demodulating signals 19.5 dB below the noise floor, while comparative FSK systems need a signal power of 8-10 dB above the noise floor for successful demodulation [21, 24]. The LoRA capable RFM96W was chosen, since it offers a very high sensitivity of -148 dB m, while using just 0.2 µA when in sleep mode.

Two of the radios which were included for testing, the RFM23BP and E31-TTL, have a Tx output power of 30 dB m even though this falls outside the Independent Communications Authority of South Africa (ICASA) regulations at this frequency [14]. These devices were included to give an indication of the range that would be possible at this higher output power. If it showed significant improvements over other options, then special permission could be requested from ICASA to utilise them.

4.2.2 The Testing Station and Antenna

A testing station was built which housed all 5 of the chosen radios and interfaced them with a single Arduino MCU. Experimental results were output by the Arduino as serial data to a Phone or PC to be examined at a later stage. A block diagram of the testing station is given in Figure 4.4.

Power was supplied by a 12 V battery to allow full portability. Voltage regulators are used to regulate the voltage to 5V and 3.3V, and supply these to each of the components as needed. Each radio can be selected individually. The MCU automatically determines which radio is currently selected and loads the corresponding software. The testing station was designed in a modular fashion so that all components can used in future projects if desired. Due to Arduino programming limitations, it was not possible to use more than one radio at the same time.

Each RF module is connected via a short 50 Ω cable to a female port grounded via a metallic box. The box acts as a ground plane and also ensures that no torque is experienced by the radio device SMA port when an antenna cable is connected.

Figure 4.5 shows a fully built testing station with each RF module connected to the

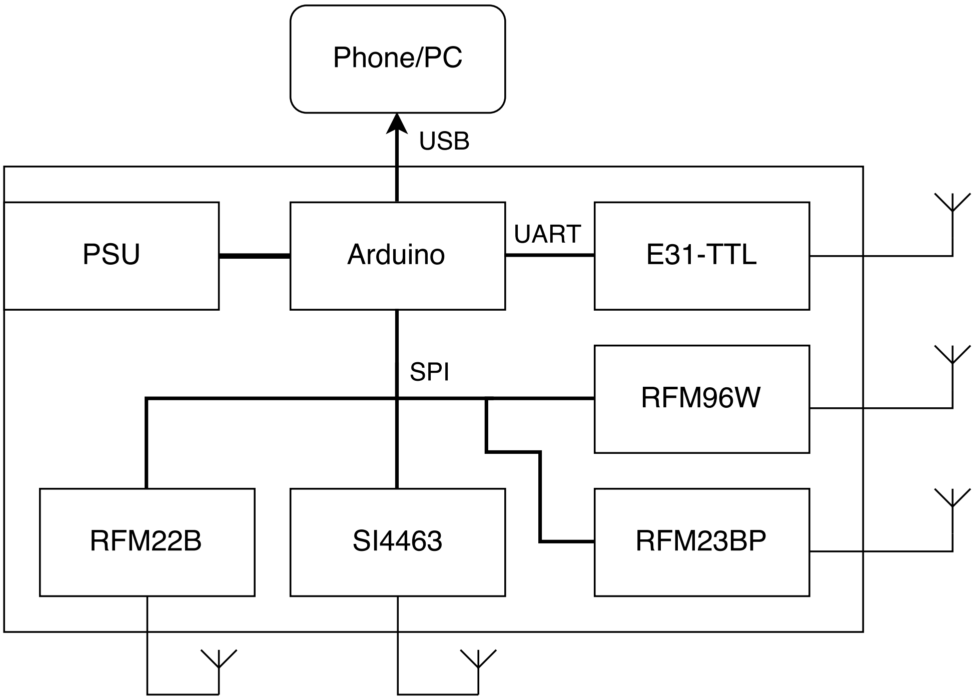


Figure 4.4: A block diagram of the testing station.

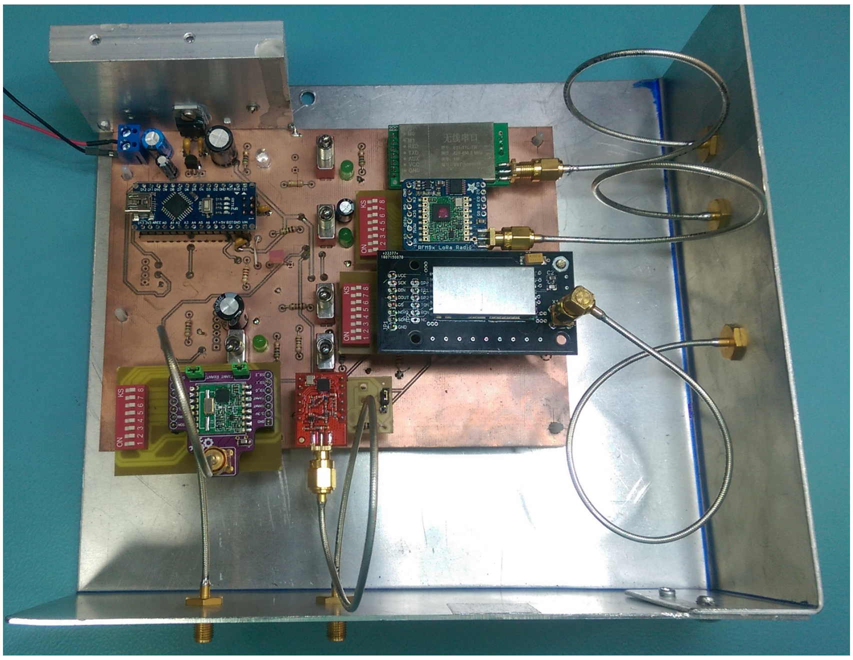


Figure 4.5: The fully assembled testing station.

Arduino MCU. The small extensions from the SMA ports on the radio devices themselves to the metal enclosure can be seen. The base station and mobile station are identical in design.

A 5dBi antenna with a S_{11} coefficient of -19 dB m at 433 MHz was used for the base station. This antenna was attached to a pole to elevate it approximately 4m above the ground. The antenna for the mobile station was attached to a fibreglass pole allowing its height to be adjusted to 0.1 m or 4 m above the ground. This antenna has an S_{11}

coefficient of -11.8 dB m at 433 MHz. While the antenna can play a role in determining which radio works best, because the same antennas are used for all the radios and all radios have outputs matched at 50 Ω, the antenna factors are mitigated.

4.2.3 The Terrain

A location in Cape Point nature reserve was chosen as the testing site as it resembles the environment in which the network is expected to be deployed. The terrain is quite flat, with very few trees and, during the rainy season, can develop very thick grass and shrub growth up to approximately 1 m in height. Measurements were possible over distances varying between 0 to 5500 m. Waypoints were placed at set intervals along an empty road, for the placement of the mobile antenna and testing station during testing. Figure 4.6 gives an indication of the type of terrain at the test site.

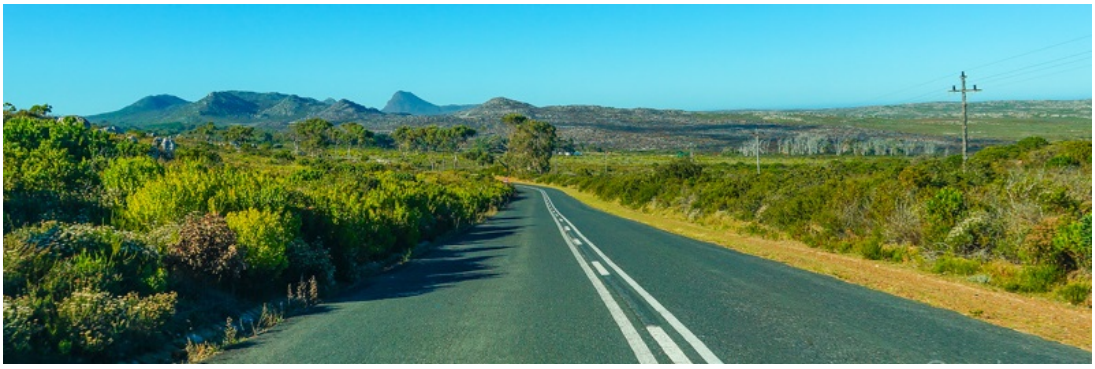


Figure 4.6: An example of the terrain and bush found in the chosen testing environment at Cape Point Nature reserve.

4.2.4 Testing Protocol

The objective of the test was to determine how the radios will function when used for the WSN node to node communication, as well as for on-animal tag to WSN node communication. Because this thesis focuses on the WSN communication, no tests were performed to determine the on-animal to on-animal tag communication.

One testing station, henceforth known as the base station, remained at a single location with its antenna fixed at a height of 4m above ground level. The other testing station, henceforth referred to as the mobile station, was moved to predetermined waypoints and tests conducted with its antenna at 10 cm (low) and 4 m (high) above ground. At each waypoint, each of the 5 radios sent 30 packets of data from the mobile station to the base station. Each packet received by the base station was acknowledged with a response packet from the base station to the mobile station. Each time a packet was received at either station the received signal strength indicator (RSSI), packet number as well as the data sent was stored. Therefore, for each particular configuration of waypoint and antenna height, 60 packets were sent between stations. Since the E31-TTL does not provide an RSSI measurement, only the success of the packets communicated could be analysed for this device.

4.2.5 RSSI Measurements

In wireless communications, the measurement of the power present in a received radio signal is known as the Received Signal Strength Indicator, RSSI [25]. The RSSI gives a measure of the link quality between two radios who are attempting to communicate with each other and is measured in dB m with values closer to 0 dB m indicating better link quality while more negative values refer to worse link quality. Each radio has a maximum RSSI which is given by the manufacturer and refers to the lowest value of RSSI which can be obtained before a link between radios can no longer be achieved.

The RSSI is not a definite measure of how successfully a radio receives packets at a certain distance. However, it does give an indication of how close a radio is to the limits of its sensitivity and thus the limits of its communication range.

RSSI measurements for the radios over the test range are shown in Figure 4.7. No results for the E31-TTL radio are given since this radio does not provide an RSSI measurement. From the figure it is possible to identify the distance when a radio was no longer able to establish communication, as the point where the plot terminates.

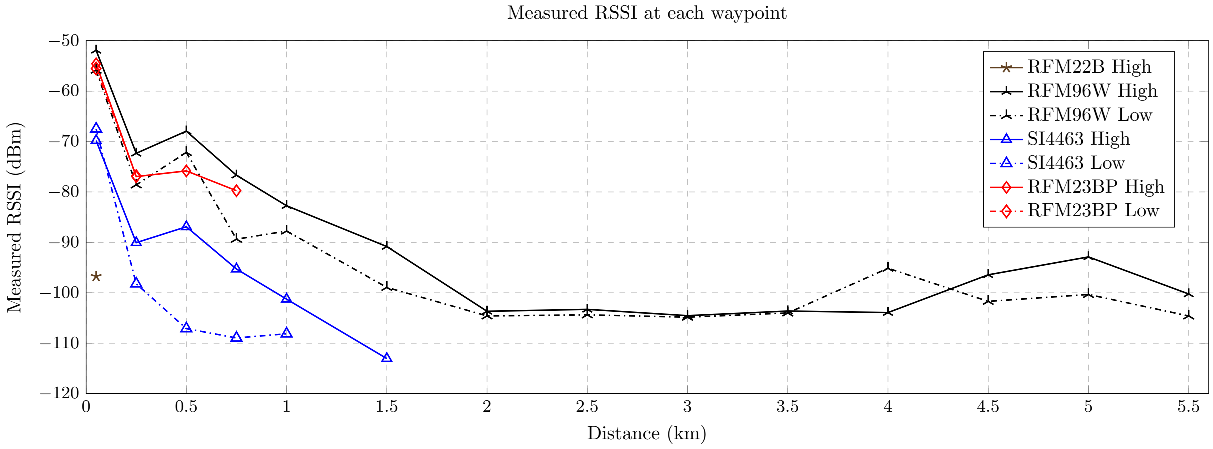


Figure 4.7: The RSSI measurements at each waypoint (from 50m to 5500m) from each radio when the mobile antenna was set to 4 m (high) as well when the antenna was set to 0.1 m (low).

The further the two radios are from one another, the lower the RSSI becomes. This is expected as the signals between the radios experience increased free space losses as the distance between the radios increases [26]. The signals are also progressively disrupted and deflected by objects in the environment or absorbed by the ground and foliage.

Another trend observed in the figure is that the RSSI is on average higher when the mobile antenna is set to a height of 4 m (high) above ground than when it is set to a height of 0.1 m (low). This is due to increased absorption by the ground as well as increased signal reflection by foliage due to the low antenna height. When the antenna is set to the high position, it is clear from most of the ground clutter and therefore the signal is stronger.

At a separation distance of 50 m, all radios show an RSSI above -70 dB m, except the RFM22B which has an RSSI of -96 dB m. No RSSI measurements were obtained from the RFM22B at 50 m when the mobile antenna was set to low, meaning that the signal had dropped below the sensitivity of the radio. The RFM23BP is able to determine RSSI measurements up to distances of 250 m with the mobile antenna low and 750 m with the antenna high.

The SI4463 is able to provide an RSSI up to a separating distance of 1000 m when the mobile antenna was set to low and 1500 m when the antenna was high. It is interesting to note that the SI4463 has a claimed sensitivity of -120 dB m, and at a distance of 1500 m the RSSI was -117 dB m. At greater distances, the radio was no longer able to provide an RSSI, meaning that it had moved out of range. The SI4463 performs much better than the RFM22B or RFM23BP. However, the LoRa capable RFM96W, was able to obtain an RSSI over the entire 5.5 km range whether the antenna was set to low or high.

The RSSI for the RFM96W initially decreases as the distance between stations increases. Unlike the other radios, which stop receiving an RSSI after 1500 m, the RFM96W reaches a minimum of -105 dB m at 2000 m after which the RSSI remains mostly constant. Furthermore, unlike the other radios which exhibit considerable differences in RSSI between the high and low antenna positions, the RFM96W does not show a large difference between these testing conditions.

From the RSSI measurements in Figure 4.7, the RFM96W appears to be the best radio by some margin for this project. However, the RSSI measurements by themselves are insufficient for a judgement on the radio. Because most manufacturers determine the RSSI by measuring the power at the input of the transceiver before any decoding of the signal has taken place, there is no indication of whether the data has been corrupted during transmission. To determine whether such corruption has occurred, the percentage of successful packet transmissions between stations must be considered.

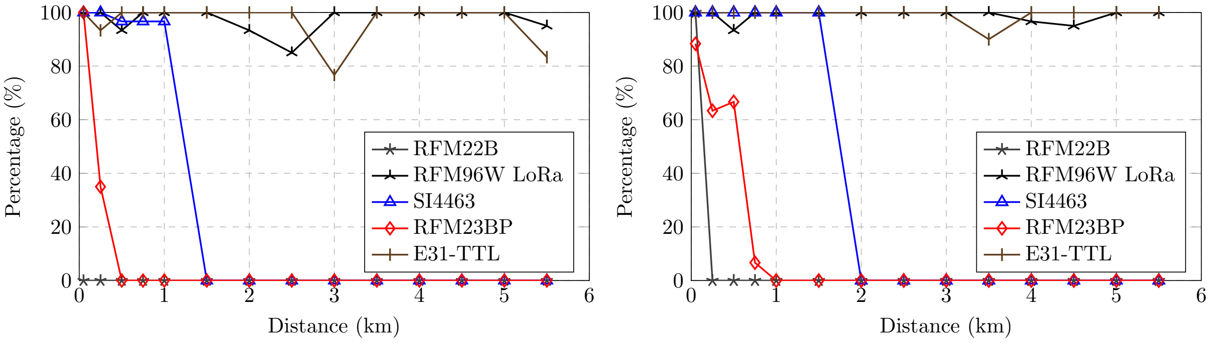
4.2.6 Packet Transmission

The RSSI measurement, together with the percentage of packets received, give an indication of how well a radio performs at a specific distance. The packets that were sent between radios have the following characteristics. The packets sent from the base station to the mobile station were made up of 12 bytes, and the packets sent from the mobile station to the base station were made up of 21 bytes. This packet length is on top of the built-in preamble and other data bits the radios may add to each transmission.

Figures 4.8a and 4.8b indicate the percentage of successful packet transmissions as a function of the distance for low and high antenna positions respectively. For a packet to be considered as successfully transmitted, it must be received with no errors.

In general packet transmission is worse in Figure 4.8a than in Figure 4.8b due to the lower antenna height. This can be ascribed to the reflections and absorption introduced by objects and foliage near the ground. This trend agrees with that observed in the RSSI measurements.

With the antenna set to the low position, the RFM22B was not able to correctly transmit or receive even a single packet while the RFM23BP was not able to transmit any packets successfully from 250 m onwards and the SI4463 was able to transmit all



(a) Antenna low

(b) Antenna high

Figure 4.8: The percentage of packets which were successfully transmitted between stations for (a) antenna low and (b) antenna high positions. Each marker indicates the percentage at a specific distance, where 100% corresponds to no packet loss, while 0% indicates that no packets were successfully transmitted.

packets without error over a distance of up to 1000 m. After this, packet transmission for this module rapidly approached 0%. The RFM96W and E31-TTL were similarly effective in terms of packet transmission, transmitting an average of 97.62% and 96.67% of packets respectively over the entire considered range of 5500 m.

When the antenna was set to the high position, the RFM22B and RFM23BP were not able to transmit successfully from 250 m and 750 m onwards respectively. The SI4463 transmitted all its packets successfully up to a distance of 1500 m, after which the success dropped to zero. The RFM96W and E31-TTL successfully received 98.93% and 99.29% of all their packets respectively.

A summary of the percentage of packets successfully transmitted over all considered distances is presented in Table 4.4. As noted from Figures 4.8a and 4.8b, the radios perform better when the mobile antenna is set to the high position. The table also shows that there are large differences in packet reception between the 5 radios.

Table 4.4: Summary of the percentage of successful packet transfers between stations, averaged over all distances.

Test/Radio	RFM22B	RFM23BP	SI4463	RFM96W	E31-TTL
Low [%]	0	9.64	35	97.62	96.67
High [%]	7.14	16.07	42.86	98.93	99.29
Average [%]	3.57	12.86	38.93	98.28	97.98

The RFM22B, and RFM23BP received fewer than 15% of all packets transmitted and showed only a slight improvement when the antenna was set to its high position. The SI4463 performed marginally better, able to receive 38.93% of all transmitted packets. Overall the RFM96W was able to successfully receive 98.28% of all packets while the E31-TTL was able to successfully receive 97.98% of all packets. The RFM96W was able to receive slightly more packets with the antenna low than the E31-TTL, with a success rate of 97.62% for the RFM96W compared to a success rate of 96.67% for the E31-TTL. With the antenna set high, the E31-TTL does marginally better, successfully transmitting 99.29% of all packets versus the success rate of 98.93% for the RFM96W.

4.2.7 Radio Module Choice

Only the RFM96W and the E31-TTL were able to achieve most of the requirements set out at the beginning of Chapter 4. Both were able to provide reliable links between nodes and communicate over distances of 2 km. They both operate within the 433 MHz ISM band and are able to provide consistent communication even when the antenna is placed in the low position. However, the E31-TTL does not abide by the ICASA regulations since its output power exceeds 100 mW.

The RFM96W was able to achieve a slightly higher successful transmission rate with the antenna low than the E31-TTL, despite having an output power 10 times lower. For these reasons the LoRa capable RFM96W will be the radio module chosen for the WSN for this project. The hardware design and integration of this radio module to the WSN node is described in Chapter 5.

4.3 Summary and Conclusion

This chapter considered the choice of a commercially available radio transceiver module for use in our wildlife monitoring wireless sensor network. The radio specifications were constrained by physical size, power output and communication range. It was first established through experimental evaluation that 433 MHz offers a more compact antenna while maintaining a received signal strength similar to that achieved at 169 MHz. Subsequently 6 currently available radio modules were compared first on the basis of the data provided in their datasheets and then by means of physical tests.

The LoRa capable RFM96W outperformed all other candidates of the same output power. This device was able to achieve a packet loss below 2% over a 5500 m range at a power of 20 dB m. Therefore it was concluded that this LoRa communication module offers a practical candidate for an outdoor wildlife sensor network and will be used further for this project.

The results discussed in this chapter have been published in a conference proceedings from the IEEE Africon 2017 under the name, *Choosing an integrated radio-frequency module for a wildlife monitoring wireless sensor network* [27]. In the next chapter the hardware selection and design choices for the WSN node will be discussed.

Chapter 5

Hardware Selection

This chapter will describe the design processes behind the hardware that was selected for each node. The WSN nodes, once deployed, will need to be self-powering and will rely solely on batteries for that power. Therefore, much emphasis was placed on choosing low power components for use in each node. Furthermore, it must be possible to put components that are not essential or that are not currently in use into a sleep mode or to disable them completely.

Two types of node will be deployed in the network, namely an access point (server) node and an ad-hoc (mesh) node. The server node is distinguishable by the inclusion of a GSM modem, which allows data received from the network to be uploaded to an online database. A high level description of the hardware in a server node is shown in Figure 5.1.

Each node will consist of a power supply, MCU, RF module and sensors. In this chapter the reasons for including each component will be given and detailed hardware design choices will be explained.

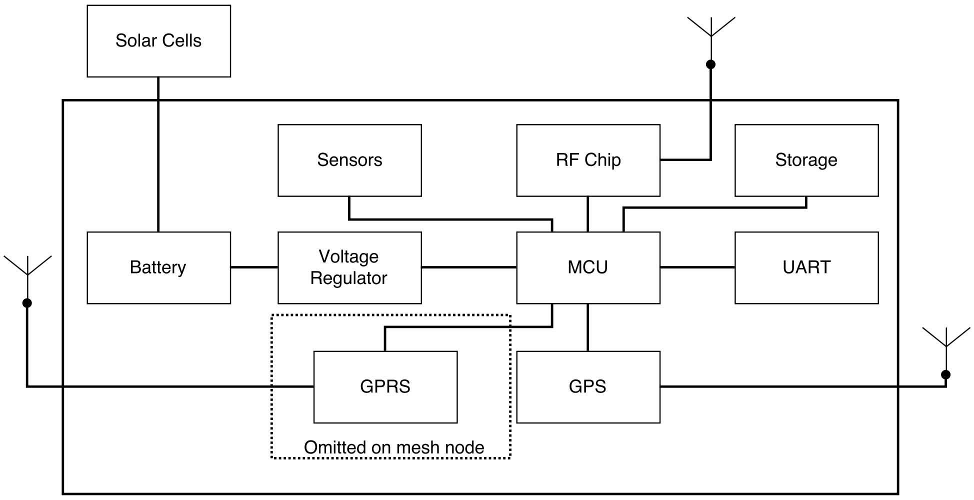


Figure 5.1: A high level description of the hardware in an access point (server) node. Ad-hoc (mesh) nodes have the same hardware components and architecture, but do not include the GSM modem.

5.1 Server Nodes and Mesh Nodes

The WSN network will consist of server nodes and mesh nodes. Both types of node form part of the wireless mesh network and must be able to receive and send messages via radio frequency (RF) communication. The server nodes and mesh nodes will share all hardware components, with the exception of the GSM modem, which is not present on the mesh nodes. Each node can configured as either a mesh node, or a server node, and the server capabilities are activated automatically by attaching a GSM modem to a mesh node. A node can also be configured as a server via a setting in the software.

Using the same hardware for both node types lowers the complexity of the design and therefore the cost of the network as a whole. It also simplifies longer-term maintainability since essentially only one design need be considered. Because of the use of server nodes and mesh nodes, the network can be seen as an non-heterogeneous network.

5.2 Choice of MCU

At the heart of each node is the micro controller unit (MCU). In choosing a suitable MCU, the following characteristics should be considered. The MCU should be designed for use in low power applications while having a processing ability which is on par with conventional MCUs. It is also important that the MCU chosen should have a sufficient number of peripheral ports to support the communication needs of the rest of the hardware that makes up the node. Table 5.1 compares 4 commonly used low power MCUs considered in this project.

Table 5.1: Comparison between 4 commonly used low power MCUs. *Represents the fact that the peripherals could be used either for SPI, UART or I2C.

Processor	ATmega328	MSP430	SAM L21	MSP432
Vendor	Atmel	TI	Atmel	TI
Clock Speed [MHz]	20	24	48	48
Flash memory [kB]	32	16	256	256
SRAM [kB]	2	1	40	64
Active mode current μA/MHz	200	81.4	35	94
Sleep mode current [nA]	2000	6300	200	840
Architecture	8-bit RISC	16-bit RISC	ARM M0+	32-bit ARM
Supply voltage [V]	4.5-5.5	1.8-3.6	1.62-3.63	1.62-3.7
Pins	32	40	48	64
Floating Point math unit	SW	SW	SW	HW
GPIO pins	23	30	37	48
A/D convertor	8x10-bit	14x10-bit	14x12-bit	12x14-bit
UARTs	1	2*	6*	4*
I2C	1	1*	6*	4*
SPI	2	3*	6*	8*
ULPBench scores	N/A	123.7	185.5	192.3

The processors which are compared in Table 5.1, are all well suited to low power applications. The MSP430 by Texas Instruments (TI) has been frequently used in wireless sensor networks over the past few years, for example to monitor the air quality in a mine [28] [29], to monitor of the height and flow of a river [30], and even to monitor badgers in their natural habitat [6].

The SAM L21 and MSP432 are both newer devices, and have been commercially available since 2015 and 2016 respectively. The MSP432, which is an upgraded version of the MSP430, uses an ARM M4 processor in place of the proprietary RISC processor used in the older device, and has a more accurate analog to digital convertor (ADC) as well as a larger number of peripheral ports. The MSP432 also has a lower sleep current of 840 nA compared to the 6300nA used by the MSP430.

The SAM L21 is similar to the MSP432 but does not have a hardware Floating-Point Unit. For our project this may become a disadvantage in future, should classification algorithms be implemented on the nodes. Such algorithms which extensively utilise floating point calculations could then result in higher power usage on the SAM L21.

The Atmel ATmega328 is very popular and is found in the Arduino platform. However, despite its smaller number of general purpose input and output (GPIO) pins and lower clock speed, it still consumes more current per MHz than the other devices in Table 5.1. The ATmega328 also has fewer peripheral buses and is therefore not a serious contender against the other MCUs for use in the network.

The Ultra Low Power Bench (ULPBench) scores are an industry accepted and standardised score of the MCU energy efficiency in ultra-low power applications. The scores are based on a combination of the power usage of a MCU during sleep modes, time to wake up, and power usage during active modes. The scores show how well a particular MCU is adapted to ultra-low power applications, with a higher score indicating a lower power usage profile. All the scores are standardised and can be directly compared to one another, with multiple vendors endorsing the scores on ULPBench, including Intel, TI, ARM, Silicon labs and others. From the ULPBench scores alone it can be seen that the MSP432 has the lowest power usage profile and is well suited for low power applications. The SAM L21 follows closely behind, followed by the older MSP430. The ATmega328 is not listed on the ULPBench score sheet as it does not meet the requirements of an ultra low power processor.

The MSP430 has multiple low power modes. In Low Power Mode 3, for example, it uses only 6 3µA. The MSP432 also has multiple low power modes, with a sleep mode current of 840 nA. With the wide acceptance of the TI MSP430 processor for the design of WSNs, it was decided to choose a processor from the same range. The final decision was to use the TI MSP432, as it has a large WSN online community, has a very high ULPBench score and has a sufficient number of peripheral ports to allow the attachment of additional hardware.

Figure 5.2 shows all the connections between the MSP432 breakout board and peripheral devices. Multiple IO pins have been used as well as 1 SPI port, and 2 UART ports. The 3V3 line is from the output of the TPS62742 voltage regulator.

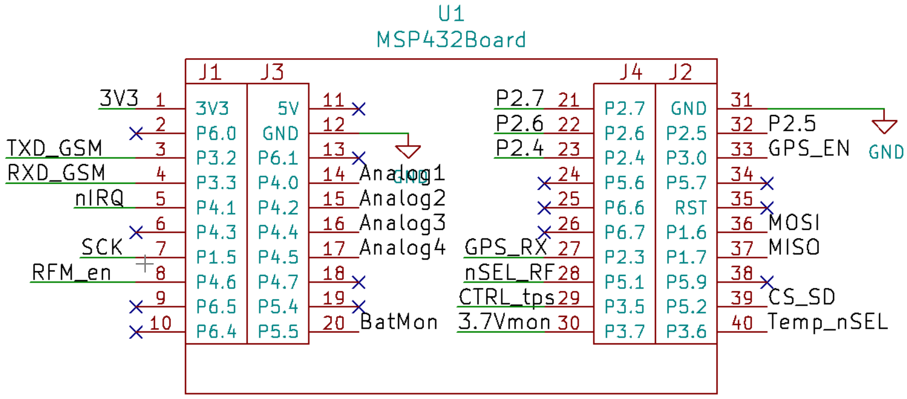


Figure 5.2: All connections between the chosen MSP432 Breakout Board and other hardware.

5.3 Choice of RF Module

The full process for choosing the RF module has already been fully described in Chapter 4. After conducting field tests, the conclusion was that the LoRa capable RFM96W by HopeRF was able to communicate 5.5 km with an accuracy of 98%. These results were better than any other RF module tested with the same output power and in certain tests the module outperformed radios with 10 times the output power in terms of distance and accuracy of communication.

Two of the studies in Chapter 2, emphasised that particular attention should be given to ensuring that the power supplied to the RF module is well regulated and has a low ripple voltage [3, 4]. To ensure this low ripple on the power supply voltage of the RF module in our system, a process of post regulation is used. This ensures that any frequency components related to the DC-DC convertor switching frequencies are filtered out. Post regulation is a process whereby a low dropout (LDO) linear regulator is placed between a switching DC-DC voltage regulator and the device to which power is being supplied [31]. The LDO filters and smooths ripple voltages which might be introduced by the switching regulator. Linear regulators do not include switching circuitry and therefore have the advantage of low noise on their regulated outputs. However, linear regulators are not very efficient and dissipate most of the excess power in the form of heat. To minimise this, the difference between input and output voltages must be kept small. In this case the input is 3.3 V therefore there are almost no losses as the LDO simply filters the input and regulates it to 3.3 V.

For this design, the AP2112K3-3 LDO is placed between the RFM96W and the output of the switching regulator of the power supply. According to the datasheet the APK2112K has a quiescent current of 55 µA and a standby current of just 10 nA when disabled [32]. A 10 µF capacitor is included to stabilise the output of the LDO. The dropout voltage of the regulator when the output current is 100 mA is approximately 50 mV, therefore the RFM96W will be supplied with a filtered voltage of 3.25 V by the LDO during operation.

The complete schematic for the RF module using the RFM96W is shown in Figure 5.3. The schematic shows the SPI connections between the RF chip and the MCU as well as the post regulation LDO, and the enable pin. An external antenna must be connected to the SMA port (X1) on the RF board. Digital input output pin zero (DIO0), which is the interrupt pin from the RFM96W, is shown to connect to pin P4.1 of the MSP432 in Figure 5.2, where it is labelled nIRQ. The other DIO pins are not used.

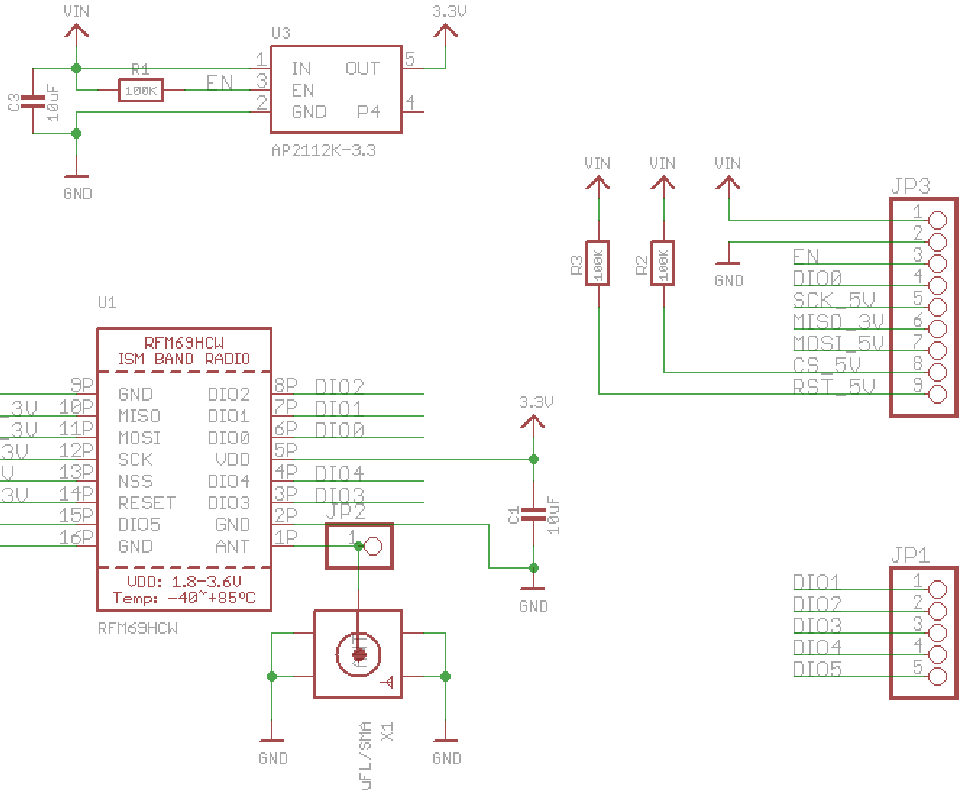


Figure 5.3: Circuit diagram showing connections to the RFM96W with post regulation.

5.3.1 RF Module Parameters Setup

The RF module was programmed with the following parameters during setup. The output power of the module was set to 20 dB m. The bandwidth was set to 31.25 kHz, and the spreading factor was set to 512 chips/symbol. The cyclic redundancy check (CRC) coding rate, which will be discussed further in Chapter 6, was set to 4/8.

5.4 Antenna

The antenna attached to the WSN node will be connected to the RFM96W LoRa module which operates at a frequency of 433 MHz. This antenna will need to facilitate communication with other WSN nodes as well as with on-animal nodes. We will begin by first discussing the different antenna polarities and their typical uses, and then selecting an antenna for use in our project.

An antenna is a transducer which converts a radio frequency electric current to electromagnetic waves that are radiated into space [33]. The orientation in space of the electric field determines the polarisation of these electromagnetic waves. Most antennas radiate either with linear or circular polarisation. Examples of the polarisations with which antennas can radiate are shown in Figure 5.4 and are expanded on further in this section.

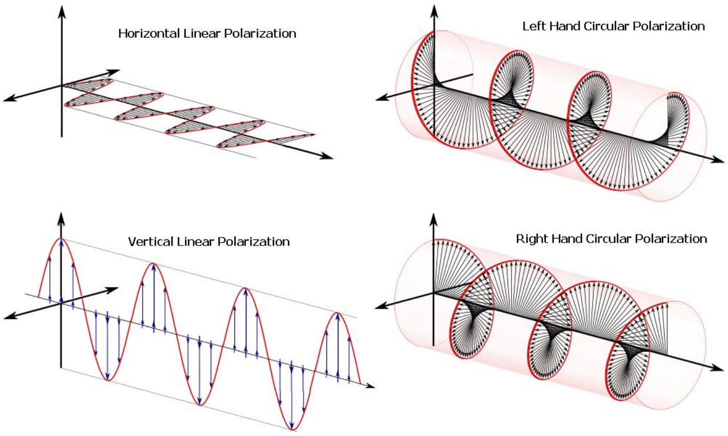

Horizontal Linear Polarization
Left Hand Circular Polarization
Vertical Linear Polarization
Right Hand Circular Polarization

Figure 5.4: Linear and circular polarised electromagnetic waves radiated from antennas [34].

5.4.1 Linear Polarisation

A linearly polarised antenna produces an electromagnetic wave whose electric field components lie in fitted planes. There are two forms of linear polarisation, vertical and horizontal. In a vertical orientation, the electric field remains perpendicular to the Earth's surface, while in the horizontal orientation the electric field produced is parallel to the Earth's surface. For maximum strength of the received signal both transmitting and receiving antenna should have the same polarisation. If the polarisations do not match, the received signal strength will decrease. The formula for polarisation mismatch loss between a linear transmitting and linear receiving antenna X_{LIN_LIN} is given by Equation 5.4.1, with θ the angle by which the antennas are rotated relative to each other [35].

$$X_{LIN\_LIN} = -20 \log \Big[cos(\theta) \Big] \text{ dB} \tag{5.4.1}$$

When both linear antennas share the same polarisations, $\theta = 0$ and the mismatch loss will be 0 dB. However, if the antennas are rotated $\theta = 90$ relative to each other, the mismatch loss will be infinite. In practice this loss is usually approximately -30 dB [36]. Linearly polarised antennas are usually easier to manufacture, as they can be as simple as a wire, or dipole antenna. Therefore they are very popular for low cost networks, and for FM radio reception in a car.

5.4.2 Circular Polarisation

In a circularly polarised antenna, the plane of polarisation rotates by one complete revolution during each period of the electromagnetic wave. Unlike linearly polarised antennas, a circularly polarised antenna can have a polarisation pattern that is right-hand-circular, which follows a clockwise pattern, or left-hand-circular which follows a counter clockwise pattern. Examples of these polarisation patterns can be seen in Figure 5.4. The polarisation mismatch loss for a circular to circular polarised antenna X_{CIR_CIR} is given by Equation 5.4.2 [35].

$$X_{CIR\_CIR} = -10\log\left[\frac{1}{2} \pm \frac{1}{2}\right] \text{ dB} \tag{5.4.2}$$

The plus sign in Equation 5.4.2 is used when both circularly polarised antennas have the same polarisation orientations. Therefore, if both antenna are right-hand-circular or left-hand-circular the mismatch loss will be 0 dB. However, if one antenna has an left-hand-circular orientation and the other right-hand-circular, the mismatch loss will be infinite, meaning that the antennas will not be able to communicate. This property can be exploited in radio networks by allowing two signals with differing circular polarisations to operate at the same frequency within the same area.

5.4.3 Linear to Circular Polarisation

Wherever possible, it is recommended to use either linear or circular polarised antennas within a single system. However, it is possible to mix the polarisations. When this happens the linearly polarised antenna will simply pick up the in-phase component of the circularly polarised wave and vice-versa. The polarisation mismatch loss for a linear to circularly polarised antenna X_{LIN_CIR} is given in Equation 5.4.3 [35].

$$X_{LIN\_CIR} = -10\log\left[\frac{1}{2}\right] \text{ dB} \tag{5.4.3}$$

Thus the polarisation mismatch loss will be -3 dB for a linear to circularly polarised antenna. The same is true for a circular to linear polarised antenna. Because the antennas are always able to pick up the in phase component of each others' signal, the antennas will facilitate communication regardless of their orientation.

5.4.4 WSN Antenna

The antenna for the WSN nodes will all be mounted securely, at similar heights and with the same orientation. Therefore, the antenna polarisation will remain constant. A circular or linear antenna polarisation can be used for the nodes, with both configurations yielding similar results in terms of received signal strength between stations.

The on-animal nodes make use of an antenna designed by Mr Sam Dodson specifically for this project. This antenna is linearly polarised, and is located within an enclosure attached to the animal's leg. When the animal moves, the antenna may change orientation, which would affect the polarisation of the antenna. If the WSN antenna is linearly polarised, and the polarisation orientation of the on-animal antenna does not match the WSN antenna closely, the received signal strength will appear weaker at the receiving antenna. As a worst case scenario, if the on-animal antenna is rotated by 90 degrees with respect to the orientation of the WSN node antenna, it may happen that no communication at all is possible between the on-animal node and the WSN node.

A solution to this, is to use circularly polarised antennas for the WSN node, and the specially designed linearly polarised antenna for the on-animal nodes. With this antenna setup, the received signal will always be 3 dB lower than it would be with a perfectly aligned linear to linear transmission. However, the orientation of the linearly polarised on-animal antenna will have no affect on the received signal strength at the WSN node. Therefore this compromise, will ensure that transmissions between an on-animal node and WSN node will always be possible, and not be affected by changes in the on-animal antenna orientation as the animal moves about. The use of circularly polarised antennas for the WSN nodes will also not affect communication between the WSN nodes themselves.

Due to the difficulty in obtaining a circularly polarised antenna, a decision was taken with the other members of the RhinoNet group to use a 5 dBi outdoor omni-directional linearly polarised antenna for the WSN nodes. This antenna will provide a large gain, while being able to pick up signals from any direction.

5.5 Storage

A 2 GB SD card is incorporated into each node for use as non-volatile storage. This allows for storage of data that exceeds the MCU's RAM or must be preserved even if the MCU loses power. Examples of data that can be stored here are messages from the network that could not be sent to another node, or temperature sensor data that is periodically measured to be sent at a later time. The size of the SD card is limited to 2 GB by the SD protocol implemented in the code[1]. This storage will also be used to record diagnostic information such as the number of retries needed to send a message and other network health related data which may help improve the network over time.

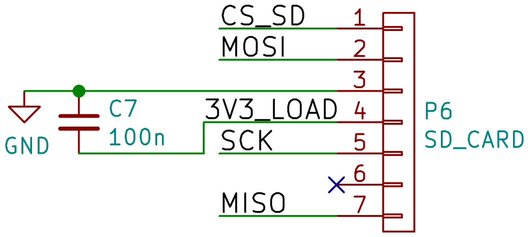


Figure 5.5: Connections between the SD card and MSP432.

The schematic showing the SD card interface is shown in Figure 5.5. The schematic shows the SPI connections between the SD card and the MCU, as well as a 100 nF capacitor between the 3V3 rail and the ground connection to ensure stable input power.

[1] Implemented FatFS - Generic FAT Filesystem Module by ChaN

5.6 GPS Module

Each node makes use of a Global Positioning System (GPS) module to update the real-time clock (RTC) on the MCU, and to store the positional data of the node. By synchronising clocks across the network in this way, the time that messages take to traverse the network can be determined with high accuracy. Furthermore, there is no requirement to set the clock of the individual nodes manually. The GPS information can be used to detect when the position of a node has changed so that it can send an alert that it has experienced some sort of tampering.

The GPS that was chosen is the GlobalTop MTK3339, which is able to track 22 satellites and has a receiving sensitivity of -165 dB m [37]. The module has an on-board patch antenna with the option to attach an external antenna if required. The operating current of the GPS is typically 25 mA during acquisition and 20 mA during tracking. The MTK3339 has a hot start time of 1 second and a cold start time of 35 seconds. A cold start is performed when the GPS has lost all data in its memory and needs to obtain the almanac and all tracking data from the satellites afresh. A hot start does not require this lengthy process by maintaining a copy of the almanac and RTC and simply updating the positional data from the satellite connection. In order to ensure that the GPS is able to perform a hot start, power to the Vbackup pin, which powers the volatile memory of the GPS, must not be interrupted.

A module incorporating the MTK3339 was obtained from Adafruit. This module has a built in connector for a CR1220 coin cell battery and also a header to connect the required IO pins to a MCU. The Adafruit GPS module also incorporates a MIC5225-3.3 LDO regulator which ensures a fixed supply voltage of 3.3 V. The MIC5225-3.3 LDO can be disabled when the GPS is not in use and this disables power to the GPS resulting in a total current draw of just 0.1 µA by the module. While powered down, the GPS will be able to maintain all satellite tracking information, its RTC and the contents of its volatile memory while a battery is attached to Vbackup.

The connections between the GPS module and the header connections to the MCU are shown in Figure 5.6. A small LED, used for debugging, is placed on the 3D-FIX pin which gives a visual indication of when the GPS is ready to output positional data received from the satellites. Once the GPS has located 3 satellites within its sight the LED will flash once per a second. The LED can be removed once all features are confirmed to be working successfully to lower power consumption.

5.7 GSM Module

For the WSN to connect to the internet, a Global System for Mobile Communications (GSM) modem is included in the server nodes. All data that is transmitted from a rhino tag will be routed by the network to the nearest server node, which will then upload the data to its final destination. The GSM modem facilitates an HTTP, or SSL connection via GPRS to an online web server. In this way data can be uploaded to this server. Once the data has been successfully uploaded, the HTTP connection is closed and the modem is put back into sleep mode.

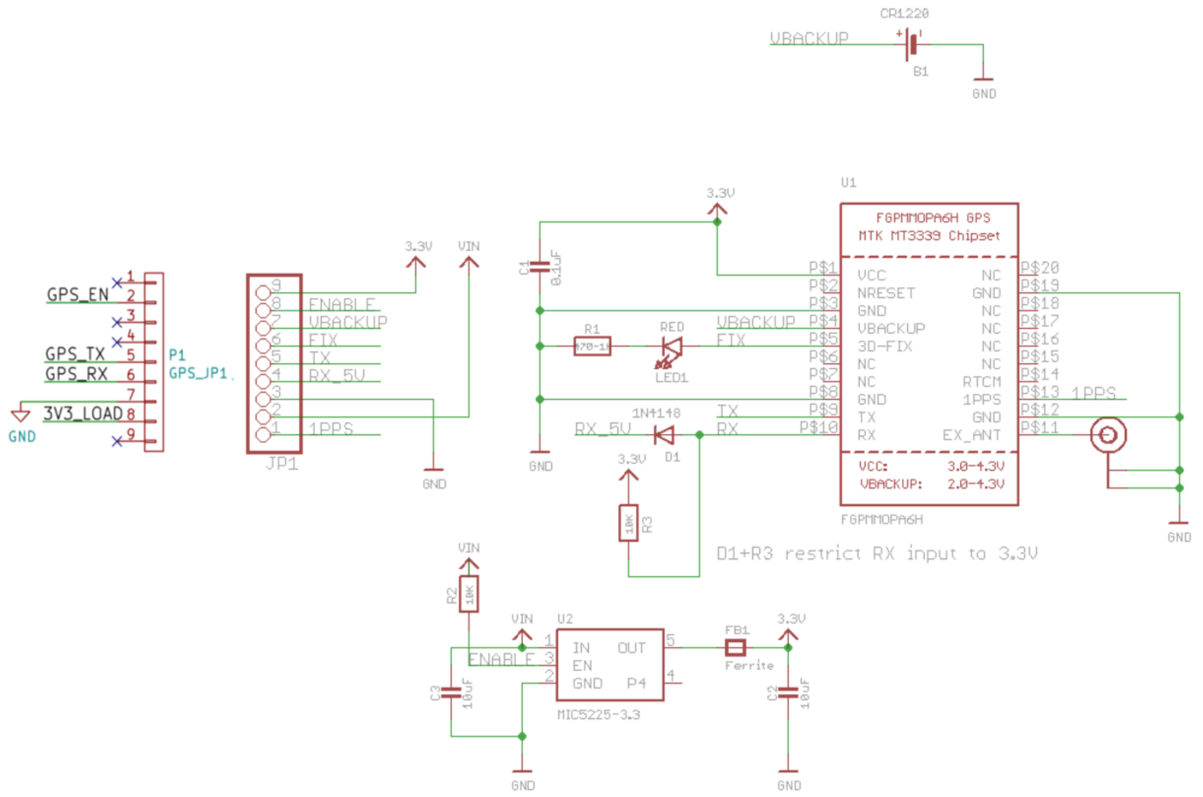


Figure 5.6: Complete schematic of the MTK3339 GPS chipset with post-regulation and connections between header and MSP432.

The GSM modem chosen for this project is the Telit GL865-QUAD. This is a quad-band GSM chip, capable of sending SMS, making and receiving calls, as well as creating a GPRS connection [38]. The GL865 requires an activated SIM-card to use these features. The GSM chip expects a supply voltage of between 3.2 V and 4.5 V, and when in an idle state, has a current usage of 1.5 mA. The current drawn during operation is 230 mA, and when powered off this reduces to 5 µA.

The Telit GL865 is supplied in a module from MikroElektronika called the GSM Click. The GSM click incorporates the GL865 chip, a SIM-card holder, an SMA antenna connection as well a header port to connect the chip to a MCU. The GSM click uses a UART to communicate with the MCU. Figure 5.7 shows the connections between the GSM Click and the MCU. Several capacitors are placed on the input of the board to ensure smooth current flow.

5.8 Temperature and Pressure Sensor

Every node includes a temperature and barometric pressure sensor. Not only can these measurements be used for future conservation projects, but the data can also be combined with a smoke detector to determine, in real time, whether there is a fire nearby and alert the authorities [39]. The data could also help debugging possible hardware failures during long term usage of the nodes.

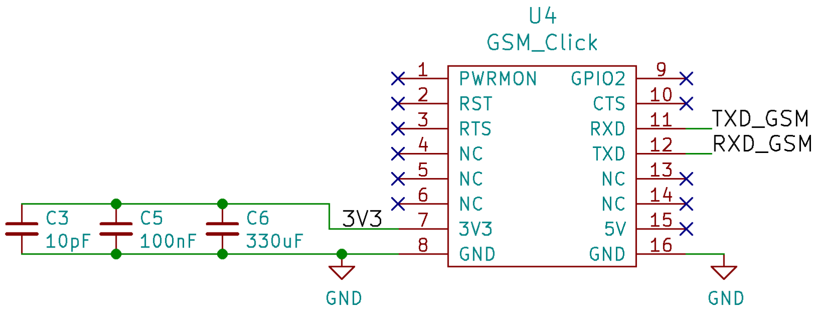


Figure 5.7: Schematic showing connections between the GSM click and the MSP432 as well as all supporting hardware.

The MS5607 temperature and pressure sensor from Measurement Specialities has a low-power usage of just 0 15µA in standby mode and 3 2µA in active mode. This device supports both SPI and I2C communication with an MCU. For this project, the SPI interface was chosen. When a temperature or pressure reading is requested by the MCU, the MS5607 will take the measurement and communicate this to the MCU. Once complete, it will automatically return to sleep until a new measurement is requested. Figure 5.8 shows the connections between the MS5607 and the MCU. The SPI bus communication is selected by pulling the PS pin of the MS5607 to ground.

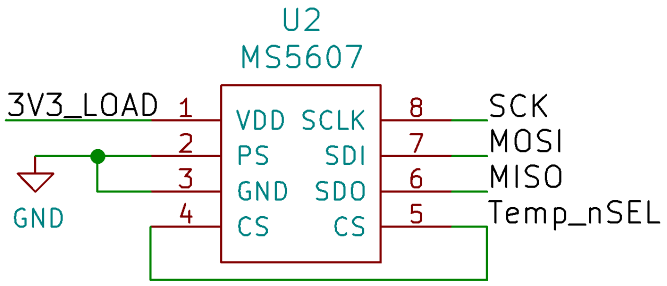


Figure 5.8: Connections between MS5607 temperature sensor and the MSP432

5.9 Power Supply Overview

Since the nodes will be located in the open veld, they must be self sufficient in terms of energy requirements. To achieve this, each node will include a battery, solar energy harvester and voltage regulator. The following sections describe the design of the various components of the power supply, shown diagrammatically in Figure 5.9. This section will describe the maximum power requirements of the node, and also give a summary of 4 power profiles which offer an estimation to the amount of power required by a mesh and server node in normal and extreme usage conditions. Section 5.10 details the design of components for the voltage regulator, Section 5.11 details the design choices for the battery. Section 5.13 details the design choice for the solar panels and finally Section 5.14 details the design choices for the energy harvesting module.

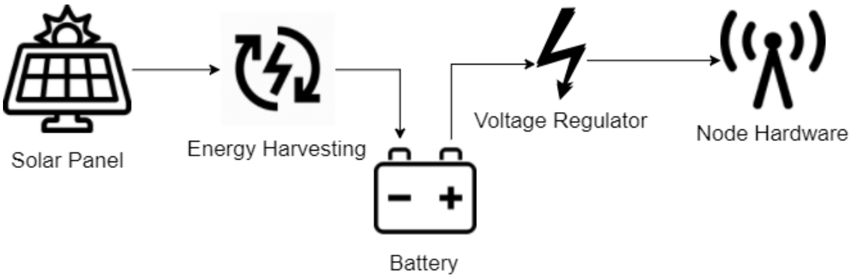


Figure 5.9: A high level outline of the power supply for a WSN node.

5.9.1 Power Requirements

Table 5.2 presents the current used by each component when powered down, in its sleep state and in an active state. If a usage profile for each node can be estimated, the values from Table 5.2 can be used to calculate the expected current drawn during various states. All the values presented in the table are taken from the component datasheets and may vary slightly during actual use. In the table, Powered down (PD) refers to the component being completely switched off and requiring re-initialisation by the MCU before use. Sleep (SLP) refers to the components' sleep mode, activated by the MCU. Active (ACT) refers to the component in its active state, which for the RFM96W radio is further broken up into receive (RX) and transmit (TX).

Table 5.2: Current used by each component while powered down, or sleep, and active states (according to the datasheet).

Component	Powered down [nA]	Sleep [µA]	Active [mA]
MCU	840	600	3
Radio	10	0.2	12.1 (RX)/120(TX)
GPS	100	0.1	25
GSM	5000	1500	230
Temp	150	0.15	0.0032
SD Card	0	700	1.47

5.9.2 Absolute Maximum Current

The absolute maximum current can be calculated by adding the current used by each component of a WSN node in its active state. It is highly unlikely that all components will be in their active state simultaneously. However, as a precaution, the power supply must be able to supply this maximum current, calculated in Equation 5.9.1.

$$\begin{aligned} I_{MAX} &= MCU_{ACT} + Radio_{TX} + GPS_{ACT} + GSM_{ACT} + Temp_{ACT} + SD_{ACT} \qquad (5.9.1)\\ I_{MAX} &= 3 + 120 + 25 + 230 + 0.0032 + 1.47\\ I_{MAX} &= 359.4732\text{mA} \end{aligned}$$

Therefore the maximum current that a node will draw from the battery, if all components are simultaneously in their active states and the RF module is transmitting, is approximately 360 mA.

5.9.3 Power Profiles

To understand the power requirements of the WSN nodes, four power profiles were used to estimate the node power usage. These power profiles describe the power usage of a server and mesh node during expected and extreme usage conditions. The specifications of each power profile will be summarised in this section, while the full calculations are available in Appendix B.

The states of the WSN node can be separated into various activities. In an activity, the power required by each hardware module can be calculated. For example, during an RF transmission state, the MCU will be active, the RF module will be transmitting, while all other hardware will be in sleep mode. Knowing this, an equation can be set up to calculate the current required for this activity. Table 5.2 can then be used to calculate the current and power required during this activity.

The current used by each activity is calculated using Equations B.1.1 to B.1.8. Each power profile will then specify how much time the node spends completing an activity, and the current required by the node during the power profile can be calculated.

A summary of the activity states most used by the node and the current required for each is shown in Table 5.3. The full table showing all the activity states as well as the current required by each is given in Table B.1 in Appendix B. Table 5.3 shows only the current required by the node when powered down, in sleep mode, and during a GSM upload. It also shows the current required by the node when transmitting (Tx) or receiving (Rx) with the RF module, as well as the current required when the node is set to the receive sleep mode ($RFRx_{SLP}$). This mode is used when the node is waiting for a new message.

Table 5.3: Summary of the activity states and the current used during each.

Activity State	Current Usage [mA]	Symbol
Power Down	6.1×10^{-3}	I_{PD}
Sleep	605.45×10^{-3}	I_{SLP}
GSM Com	233	I_{GSM}
RF Rx	15.36	I_{RF-RX}
RF Rx$_{SLP}$	12.71	$I_{RF-RX_{SLP}}$
RF Tx	123.26	I_{RF-TX}

When a node is sending or receiving a message, it follows a specific protocol, which is described in Chapter 6. Using this protocol, it is possible to determine the minimum and maximum amount of time which the RF module spends in Rx or Tx mode during a message reception or transmissions. A time, 40% above the minimum time to accommodate for retransmissions or other delays, is taken as a reference for the power profile calculations.

The time which the RF module spends in Rx and Tx mode during receiving or transmission of a message is described thoroughly in Appendix B and only a summary is given in Table 5.4. This table shows that when a message is transmitted, the RF module requires 1817.6 ms in Rx mode and 3652 ms in Tx mode. When a message is received by a node, the RF module requires 4566 ms in Rx mode and 1426 ms in Tx mode.

Table 5.4: Summary of the time a node spends in Rx or Tx mode when receiving and sending a message.

Node Completing	Rf Rx Mode [ms]	Rf Tx Mode [ms]
Sending	1817.6	3652
Receiving	4566	1426

5.9.3.1 Normal Mesh Power Profile

This power profile describes a mesh node which receives a message every 90 seconds. The mesh node will pass this message to the next hop en route to the server node. When the node is not receiving or sending messages, it will be set to RF Rx sleep mode. With these specifications, it is calculated that the node will require on average current of 19.13 mA and subsequent power usage of 63.13 mW.

5.9.3.2 Normal Server Power Profile

This power profile describes a server node where a message is received every 90 seconds. The server node will not pass the message along, but will instead upload the collected data via the GSM module to an online database every 5 minutes. A single upload takes 10 seconds to complete. In this power profile, the node will require an average current of 21.9 mA and subsequent power usage of 72.7 mW.

5.9.3.3 Extreme Mesh Power Profile

This power profile describes a mesh node which receives and passes along as many messages as possible. The mesh node is capable of receiving 314 messages per an hour and requires an average current of 63.16 mA to operate. The power used by the node in the extreme mesh power profile is 208.43 mW.

5.9.3.4 Extreme Server Power Profile

This power profile describes a server node which receives as many messages as possible, and uploads the data via the GSM module every 2 minutes. Under these conditions, the server node is able to receive 550 messages an hour, and requires an average current of 57 mA to operate. The power used by the node in this profile is 188.1 mW.

5.9.4 Power Profile: Summary

The mesh power profiles and their respective current and power usages are summarised in Table 5.5.

From the table, it is seen that the extreme mesh power profile has the highest current and power usage of all the profiles. Therefore, this power profile will be used as a worst case scenario when calculating anything regarding power usage.

Table 5.5: Summary of the current and power used by the nodes in each of the power profiles.

Power Profile	Current [mA]	Power [mW]
Normal Mesh	19.13	63.13
Normal Server	21.9	72.7
Extreme Mesh	63.16	208.43
Extreme Server	57	188.1

5.10 Voltage Regulator

The TPS62742 DC-DC switching voltage regulator by Texas Instruments is used to regulate the battery voltage for use by the MCU and other components. The regulator is a DC-DC buck converter, which converts voltages down to 3.3 V from input voltages between 3.3 V and 5.5 V. If the battery voltage nears 3.3 V, the device stops switching and the output is connected directly to the input. The TPS62742 has a quiescent current of 360 nA with a 90% efficiency at 10 µA output current, and an efficiency of 95% at an output of 100 mA or more. The regulator allows for a maximum output current of 400 mA which is above the absolute maximum current of 360 mA calculated in Section 5.9.2.

The device has a separate load output which can provide up to 100 mA, and is controlled by the CTRL I/O pin on the device. The LOAD output is enabled by pulling the CTRL pin high and when disabled, is internally pulled down to ground. This allows a subsystem of sensors to be powered directly from the LOAD output and disabled when they are not in use. The GPS, SD card and temperature sensor are all powered from the LOAD output of the TPS62742.

The schematic of the TPS62742 voltage regulator is shown in Figure 5.10. To select an output of 3.3 V, the four V_{SEL} lines (V_{SEL1}, V_{SEL2}, V_{SEL3}, V_{SEL4}) are all pulled up to the V_{IN} value. The datasheet recommends 10 µF capacitors for C_{in} and C_{out} which are on the input and output pins of the regulator respectively. A 2.2 µH inductor is used to minimise ripple current on the output line, and is connected between the switch output pin and the V_{OUT} feedback pin. The regulator is enabled when the BU45K33 3.3 V voltage detector detects a battery voltage of 3.3 V or above. This will be explained in more detail in Section 5.12.

5.11 Battery

The battery must power the node when no solar energy can be harvested, such as at night or when conditions are overcast. It must therefore be chosen to withstand continuous charge/discharge cycles, and should have a capacity that is sufficient to sustain the node in reasonable weather conditions. Since the node hardware requires a 3.3 V supply, and the voltage regulator can withstand a maximum input voltage of 5.5 V, the nominal battery voltage needs to be between these values. Batteries that were considered in this design were of the Nickel Metal Hydride (NiMH), and Lithium-Ion (Li-ion) type. The characteristics for these batteries are summarised in Table 5.6.

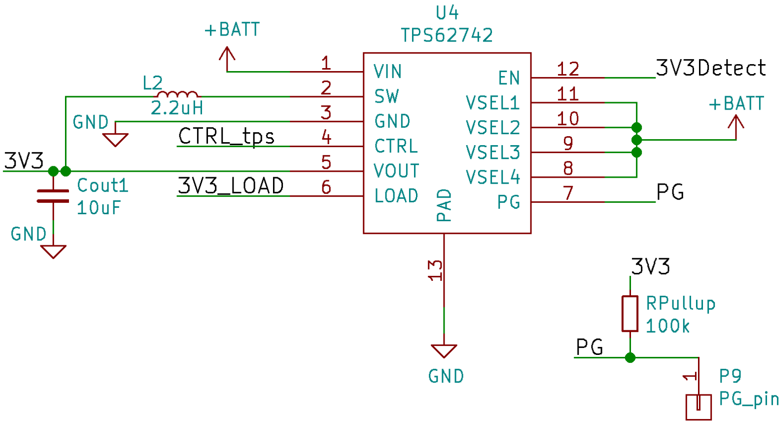


Figure 5.10: Circuit diagram of the TPS62742 DC-DC regulator. The 3v3_Load line can be enabled by pulling the CTRL pin high. The TPS62742 is enabled by the output of the 3.3 V voltage detection chip (3V3Detect).

Table 5.6: Characteristics of NiMH and Li-ion batteries

Specification	NiMH	Li-ion
Energy Density [mWh/cm^3]	260	416
Discharge rate [% per month]	30	10
Nominal cell voltage [V]	1.25	3.6
Temperature tolerance	Low	High
Memory effect	No	No

The NiMH and Li-ion batteries have energy densities of 260mWh/cm^3 and 416mWh/cm^3 respectively while the batteries have a self-discharge rate of about 30% and 10% per a month respectively [40]. The self-discharge rate for the NiMH battery is approximately 10-15% in the first 24h, after which it will lose another 10-15% per a month. The Li-ion battery has a self-discharge rate of 5% in the first 24h, after which is loses approximately another 5% per month. A single NiMH cell has a nominal cell voltage of 1.25 V while an Li-ion has a nominal cell voltage of 3.6 V. The NiMH cell is also more susceptible to overheating and loses voltage at higher temperatures, whereas the Li-ion battery is not affected much by heat. Both batteries do not suffer from memory effect, which is the need for the the battery to be fully discharged before it can be recharged, lest it remembers the lower charge. Therefore the batteries can be recharged at any time during the discharging process.

An Li-ion battery was chosen for use in this project as its specifications closely match the requirements of this project, and it is also widely accepted that this type of battery is used for projects which require longer lifetimes [41].

To calculate the required battery capacity it was assumed that the node must survive 3 days without any support from the solar charger. The battery capacity required can be calculated using Equation 5.11.1 which describes the battery capacity C needed to supply an average current I_{mean} to the node, for T_{dark} hours while taking into account the losses

C_{Loss} in the battery and power supply electronics.

$$C = \frac{I_{mean} \times T_{dark}}{1 - C_{Loss}} \tag{5.11.1}$$

The losses are a combination of the self discharge rate, which is approximately 10% for a Li-ion battery, a further 10% for the voltage regulator and another 5% for losses due to other power supply components. The expected losses are therefore estimated to be 25%. Assuming the highest average expected current from the extreme mesh power profile, I_{mean} will be taken to be 63.16 mA. The battery is expected to supply this current for 3 days, therefore T_{dark} is set to 72 hours.

$$T_{dark} = 72 hours \tag{5.11.2}$$

$$C_{Loss} = 25\% \; of \; capacity \tag{5.11.3}$$

$$I_{mean} = 63.16\text{mA} \tag{5.11.4}$$

Combining Equation 5.11.2, 5.11.3 and 5.11.4 into Equation 5.11.1 allows the capacity of the battery to be calculated for a worst case scenario.

$$\begin{aligned} C_{worst} &= \frac{63.16 \times 72}{1 - 0.25} \\ &= 6063.36\text{mA h} \end{aligned} \tag{5.11.5}$$

A battery with a capacity of 6063.36 mA h is able to supply the node for a 3 full days with no charging from the sun. However, after this time, the battery will have no remaining charge and will therefore be damaged. The battery enters a state of non-conformity at approximately 40% of its capacity after which the voltage begins to drop rapidly as seen in Figure 5.11. Therefore the battery capacity must be sufficient to ensure that at most 60% of its charge is lost during the 3 days of darkness. Equation 5.11.6 is used to calculate the capacity of the battery needed if the battery must only lose 60% of charge during 3 days of darkness.

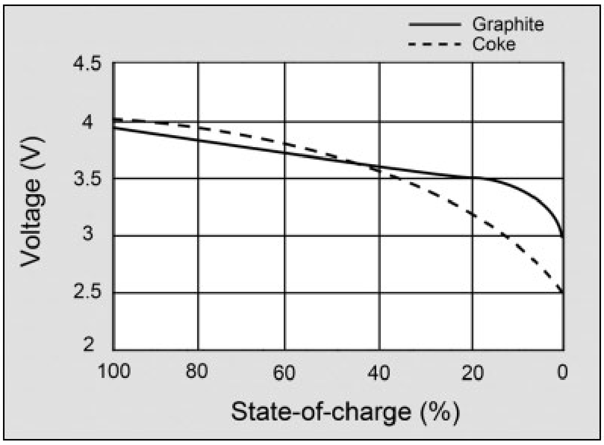


Figure 5.11: Li-ion battery discharge curve [42]. The curve for Graphite Li-ion is used for all calculations.

$$\begin{aligned} C_{bat} &= \frac{C_{worst}}{0.6} \\ &= 10105.6\text{mA h} \end{aligned} \tag{5.11.6}$$

The battery used in this project must have a capacity of at least 10105.6 mA h for it to lose 60% of its capacity over 3 days with no sunshine when the node is constantly using the maximum expected current. Therefore, a 10350 mA h Li-ion battery from Ansmann was chosen for this project. This battery has a maximum charge voltage of 4.2 V, a nominal voltage of 3.7 V and a cut-off voltage of 3 V [43].

5.12 Voltage Detection and Automatic Shutdown

Managing the battery voltage and ensuring that it does not experience excessive drain is important for maintaining long term battery health. Several systems have been put in place to manage the current drain on the battery depending on the measured battery voltage. These system are illustrated in Figure 5.12.

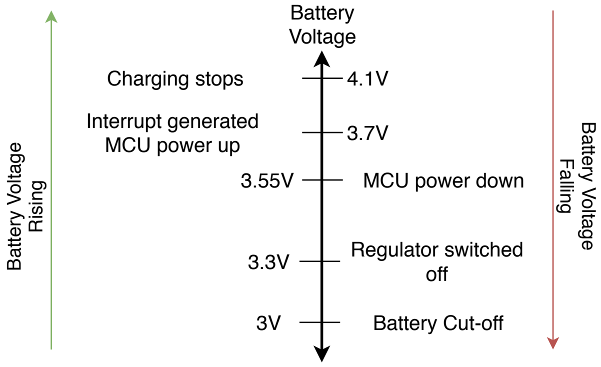


Figure 5.12: Summary of the battery threshold voltages and the actions taken when one of them is reached.

A maximum charging voltage of 4.1 V was selected to prolong the battery life and maximise the number of charging cycles that can be expected from the battery. The number of discharge cycles a Li-ion battery can withstand doubles every 100 mV that the battery voltage is below 4.2 V [44]. However, every 70 mV drop below 4.2 V also lowers the battery capacity by approximately 10%. A maximum battery voltage of 4.1 V represents a compromise between lower capacity and longer life expectancy. A battery voltage of 4.1 V ensures twice as many discharge cycles as a maximum voltage of 4.2 V, while reducing the available stored energy by only 10-15%. The Li-ion can expect between 600 and 1000 complete discharge cycles when the maximum battery voltage is set to 4.1 V, instead of 300 to 500 when it is set to 4.2 V [44].

The battery has an internal minimum voltage, which is indicated as 3 V. If the battery voltage falls below this value, a chemical imbalance can form which will prohibit further charging [45]. Therefore, an internal switch is built into the battery, to ensure that the voltage never falls below this level.

The discharge curve for an Li-ion battery is illustrated in Figure 5.11. The figure shows that the battery voltage decreases in a linear fashion until 3.5 V or as the battery charge nears 25% of its capacity. If the battery continues to discharge from this point, the battery voltage decreases exponentially until 3 V, at which stage the battery is completely discharged. For all the components on the node to function as expected, the input voltage must stay above 3.3 V. However, due to the exponential decrease in battery voltage as the voltage goes below 3.5 V, it is safer to avoid this area in its entirety.

Therefore, in order to avoid the effect of the exponential decrease once the battery voltage goes below 3.5 V, the MCU monitors the battery voltage directly through an ADC pin and once the battery voltage falls below 3.55 V, the MCU will put itself and all of the peripheral devices into sleep mode. Once in sleep mode the MCU waits to be woken up by an interrupt generated by the S-80837 voltage monitoring chip, which generates an interrupt once the battery voltage rises above 3.7 V. During this sleep mode the current drawn from the battery will be 15.45 µA by the node as calculated in B.1.2.

Considering Figure 5.11, at 3.55 V the battery should still have approximately 30% charge. The node will remain in this sleep state until the battery has charged up sufficiently for the S-80837 to trigger an interrupt to the MCU, alerting it that the voltage has again risen above 3.7 V. From a battery voltage of 3.55 V to 3.5 V, the battery will discharge approximately 10%. Equation 5.12.1 calculates the number of days that the node will be able to remain in its sleep state (T_{lef}) while waiting for the battery to charge again before the battery reaches the exponential decay in voltage.

$$\begin{aligned} T_{left} &= C_{BAT}(3.55\text{V}) \div I_{BAT}(Sleep) \\ &= 10\% \times 10350\text{mA h} \div 605\,45\text{e} - 3\text{mA} \\ &= 71.22 \text{ days} \end{aligned} \tag{5.12.1}$$

With the node in sleep mode, the battery will lose 10% of its charge in 71 days. Therefore, the battery-self discharge will have more of an effect on the battery voltage than the node. A further layer of protection for the battery is put in place for the case that the battery voltage falls below 3.3 V. This will also activate in the case of a short circuit or some other major fault with the hardware in a node. The BU45K33 voltage monitoring chip detects when the battery voltage falls below 3.3 V. If this occurs, the chip will disable the voltage regulator, disconnecting the MCU and all peripheral hardware from the battery completely. Disabling the voltage regulator also gives the solar charger more time to charge the battery before needing to power all the components again. With the node hardware completely disconnected, the only components drawing current are the voltage monitoring chips, the disabled voltage regulator and the solar charger. Equation 5.12.2 shows the current drawn from the battery once the voltage regulator is disabled.

$$\begin{aligned} I_{BAT}(Disconnect) &= I_{TPS62742} + 2 \times I_{Bat\ Monitor} + I_{LT3652} \\ &= 360\text{nA} + 2 \times 1500\text{nA} + 500\text{nA} \\ &= 3.86\text{µA} \end{aligned} \tag{5.12.2}$$

The current drawn of 3.86 µA is so low that the self-discharge rate of the battery will have a much greater effect on the charge than the components connected to it. The regulator is automatically re-enabled once the battery voltage rises above 3.3 V again.

The schematics for the BU45K33 3.3 V voltage detection chip and the S-80837 3.7 V detection chip are shown in Figure 5.13. Also shown are the voltage divider network used to monitor the battery voltage by the MSP432. The MSP432 ADC can only read voltages up to 3.3 V, therefore the battery voltage is divided by 2 and then read by the ADC.

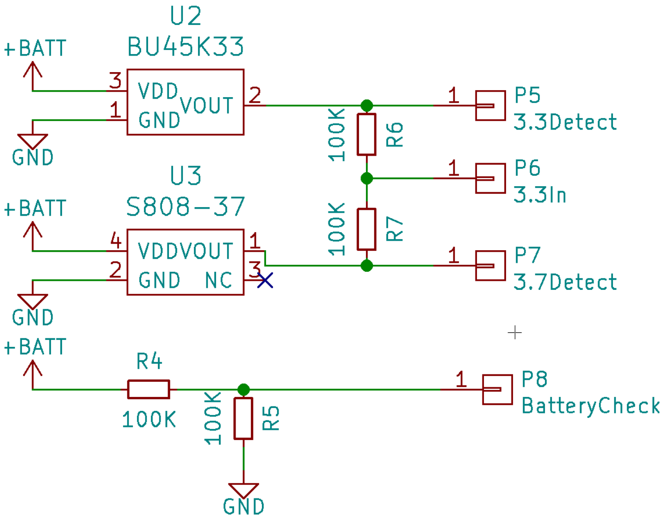


Figure 5.13: Schematics for the battery monitoring circuitry. The 3.3Detect output will disable the voltage regulator when the battery voltage falls below 3.3 V. The 3.7Detect output will be pulled high when the battery voltage is above 3.7 V. The BatteryCheck output is connected to the ADC pin A0 of the MSP432 so that it can directly monitor the battery voltage. The battery voltage is divided by 2 to ensure that it falls within the ADC's permissible input values.

5.13 Solar Power

In order to ensure long term usage of the WSN nodes, energy harvested from a solar panel will be used to recharge the battery. The solar panel should output enough power to replace any charge used by the node during normal usage, and also after extended periods of darkness. This section will describe the characteristics of solar panels and then estimate the power output of a solar panel needed to ensure the battery remains charged.

5.13.1 Solar Cells

A single solar cell has a voltage output of 0.5 V. The current and power output from a solar cell is approximately proportional to the amount of solar radiation it receives from the sun. This solar radiation is measured in watts per a square meter [$\mathrm{W\,m^{-2}}$]. The voltage and current output produced by the solar cell form an IV characteristic curve where changes in illumination, affect the current output from the cell. A solar cells' output current is proportional to the light intensity, while its output voltage is determined by the load. The IV characteristic curve for a solar cell is given in Figure 5.14. If a higher current or voltage output is needed, solar cells can be connected in parallel or series to supply either of those combinations.

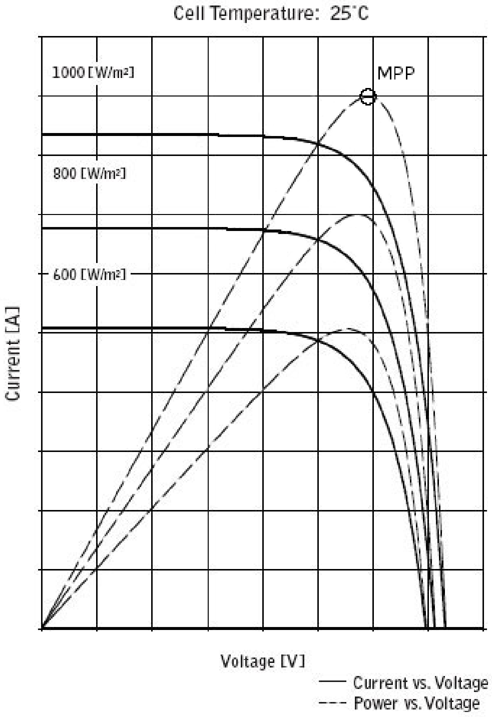


Figure 5.14: The IV characteristic curve of a Silicon Photovoltaic Solar Cell [46].

The IV characteristic curve in Figure 5.14 shows that lower illumination results in lower output current. The maximum output power, however, is produced at approximately the same output voltage, and is fairly independent of the illumination experienced by the panel. Higher surface temperatures also affect the maximum power output, with a higher cell temperature causing a reduction in the output power [47].

The maximum power point (MPP) output occurs at the 'knee' of each curve, where the solar cell transitions from a constant voltage device to a constant current device. If the voltage drawn from the solar cell is not properly managed, then the solar cell will not operate at its MPP. If a solar panel were to be directly connected to a battery, the solar panels output voltage would be governed by the battery's internal resistance and the solar panel would output voltage only up to the maximum voltage of the battery, which is very likely far from the maximum power point of the solar panel. The solar panel would also draw power from the battery when there is no light shining on the panel. The voltage at which the MPP of a solar panel occurs varies only by a small amount with changes in illumination. Therefore, to ensure maximum output from a panel to good approximation, the voltage at which the panel outputs can be set to the voltage at which the MPP occurs at 1000 $\mathrm{mW\,m^{-1}}$ illumination. As the illumination decreases the voltage will only have to be varied by a small amount to stay on the MPP of the panel.

To ensure maximum power output from a solar panel, a charger should be placed between the solar panel and the battery. A charger has the ability to steer the panels output voltage to the point of maximum power output, even when low illumination levels cannot support the full power requirements of the charger. This will ensure that the voltage which the panel must supply is always as near as possible to the MPP of the IV characteristic curve of the panel.

5.13.2 Solar Panel Combinations

Solar panels can be connected together in series or in parallel or a combination of both. If the solar panels are connected in series, then the combined output from the panels will have a higher voltage. If connected in parallel, the combined output will have a higher current [48]. Figure 5.15 demonstrates the effect on the IV characteristic curve of combining solar cells, or panels, in parallel or series combinations. The MPP for a parallel combination of solar cells will occur at the same voltage as a single cell, while the MPP for a series combination of solar cells will be at the combined voltage of all the cells.

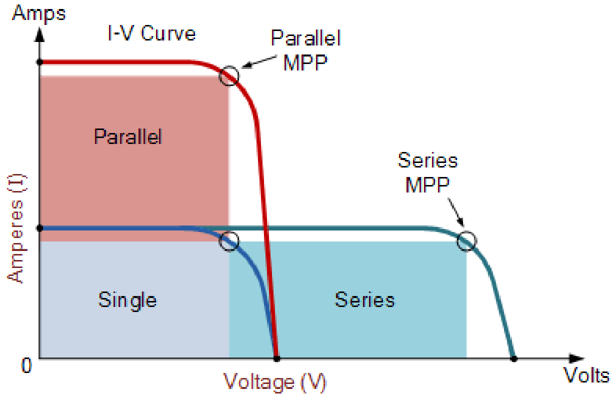


Figure 5.15: The effect on the IV characteristic curve of adding solar cells in either a parallel or series combination [49].

5.13.3 Shading

Solar panels function at their best when they are in full sunlight with no shadows cast upon them. The effects of shade falling on even a small part of the panel can cause large losses in output power. If a solar panel is made up of multiple cells in series, then the same current must flow through all cells. In the case of a slightly shaded single cell, the output current for the entire panel will drop. This occurs because the shaded cell is not able to produce as much current as its neighbours in the panel, and as a result all the cells in series with it will be forced to lower their output current to the current of the shaded cell [50]. Figure 5.16 shows the effect on the IV characteristic curve of shading on cells within a solar module.

There are a number of ways to mitigate the effect of shading on solar cells. One approach is to place all solar cells in parallel rather than in series, so that the current limiting effect of a shaded cell does not affect the others. If all the solar cells are in parallel, however, then the output voltage from the panel will be the same as a single cell, which is too low for most systems to operate on. A power optimiser on each individual panel can also be used to find the MPP for each panel, despite possible shading. The individually optimised outputs will be combined later for energy harvesting. These techniques are usually applied to larger systems and not to smaller systems such as ours. Its very important therefore for our system to be fixed in an unshaded area where the solar panels can receive direct sunlight throughout the day.

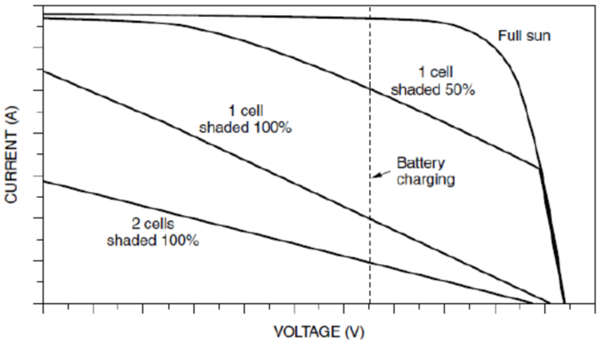


Figure 5.16: The effects of shading on the IV characteristic curve of a photovoltaic solar panel consisting of multiple cells. The dashed line shows the typical voltage at which the module would operate when charging a battery. Shading causes the MPP of the solar panel to shift drastically and the power output can no longer be achieved using the constant voltage assumption [48].

5.13.4 Illumination in South Africa

When evaluating the IV curve for a solar panel, it is important to know the illumination levels that can be expected in the area of deployment. There are 2 measurements which are relevant for illumination levels, namely the Direct Normal Irradiation (DNI) and Global Horizontal Irradiation (GHI). DNI is the amount of solar radiation in [$\mathrm{W\,m^{-2}}$] received per a unit area on a surface which is perpendicular to the incoming rays, and has not been reflected by any other surface. GHI is a combination of the DNI and light reflected by other surfaces. South Africa receives an average of 2500 sun hours per year, which equals to 6.84 hours per day. DNI levels of between 1.2 $\mathrm{kW\,m^{-2}}$ and 3.2 $\mathrm{kW\,m^{-2}}$ are experienced by South Africa annually as shown in Figure 5.17 [51]. The sun intensity during summer is not an issue for our solar charger, however, sun intensity drops during winter months. Therefore, our system should be designed to operate during the winter months.

5.13.5 Solar Panel Power Estimation

Once the required battery capacity is known, the required solar panel capacity can be calculated to ensure the battery can always be recharged. The capacity of the solar panel is based directly on the power requirements of a WSN node, the battery capacity and amount of solar energy expected in the region of deployment.

As described in Section 5.11, a battery with a capacity of 10350 mA h was chosen because it can operate for 3 full days with no input from a solar charger. After these three days, the battery charge will have significantly decreased and the solar charger should be able to charge the battery to its full capacity within 3 days. From Equation 5.11.5, the battery charge required by the node during 3 days with maximum expected power usage is 6013.44 mA h. Initially it was thought that the battery could be recharged from this state to full capacity in a single day. However, it was determined that this would require an impractically large solar panel and solar charger. Therefore, in this design, a specification of 3 days to charge the battery to full capacity after 3 days of darkness was deemed sufficient.

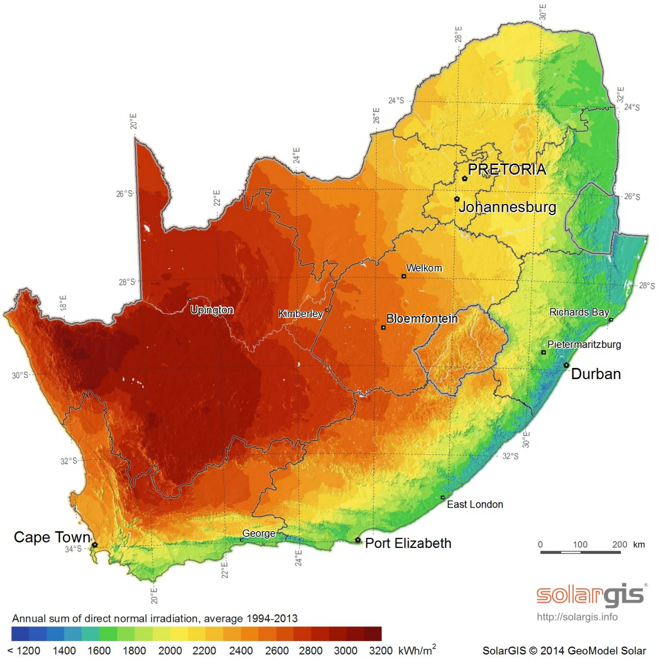


Figure 5.17: The annual sum of Direct Normal Irradiation (DNI) over South Africa [51].

The power consumed (P_{Dark}) during 3 days without input from the solar charger is calculated in Equation 5.13.1 by multiplying the capacity required to power the node (C_{Worst}), with the voltage at which the node operates (V_{Node}).

$$\begin{aligned} P_{Dark} &= C_{Worst} \times V_{Node} \\ &= 6013.44\text{mA h} \times 3.3\text{V} \\ &= 19.84\text{W h} \end{aligned} \tag{5.13.1}$$

Therefore 19.84 W h are used by the node over three days. The solar charger is expected to replenish the battery to full charge within 3 days. During the 3 days of charging, the node will continue to use power at the same rate. Therefore, the power which the solar charger must supply in order to completely recharge the battery within 3 days is that used during the 3 days of darkness (P_{Dark}), as well as the power used by the node normally during the 3 days of charging ($P_{ChargeDays}$). The power required from the solar charger ($P_{ChargerSupply}$), is calculated in Equation 5.13.2.

$$\begin{aligned} P_{ChargerSupply} &= P_{Dark} + P_{ChargeDays} \\ &= 19.84\text{W h} + 19.84\text{W h} \\ &= 39.68\text{W h} \end{aligned} \tag{5.13.2}$$

The charger must supply at least 39.68 W h during the 3 charging days. The power which the charging system should supply in a single day ($P_{SolarPerDay}$) is calculated in Equation 5.13.3.

$$\begin{aligned} P_{SolarPerDay} &= P_{ChargerSupply} \div T_{ChargeDays} \\ &= 39.68\text{W h} \div 3 \\ &= 13.23\text{W h} \end{aligned} \tag{5.13.3}$$

Therefore, the charging system must supply 13.23 W h of power per day. In Section 5.13 it was discussed that South Africa obtains an average of 2500 sun hours per year. This would mean that the average number of sun hours per a day is 6.84. A worst case scenario of 4.5 hours per a day of sunshine will be assumed in our case. Therefore, the solar panel will need to provide 13.23 W h during 4.5 hours of sunshine. The power which the solar charger must supply per an hour ($P_{SolarPanel}$) can now be calculated using Equation 5.13.4.

$$\begin{aligned} P_{SolarPanel} &= P_{SolarPerDay} \div T_{Sun} \\ &= 13.23\text{W h} \div 4.5 \\ &= 2.94\text{W} \end{aligned} \tag{5.13.4}$$

Therefore, the solar charger must provide at least 2.94 W in order to fully replenish the battery within 3 days. To calculate the required power rating of the solar panel, the charging efficiency of the LT3652 energy harvesting chip must be taken into account. This charging efficiency is related to the battery float voltage which is set by the user. With a chosen battery floating voltage of 4.1 V the charger has an efficiency of approximately 80%. The required solar panel power output ($P_{SolarPanelFinal}$) can therefore be determined using Equation 5.13.5.

$$\begin{aligned} P_{SolarPanelFinal} &= P_{SolarPanel} \div ChargerEfficiency \\ &= 2.94\text{W} \div 0.8 \\ &= 3.675\text{W} \end{aligned} \tag{5.13.5}$$

Therefore, a solar panel with an output power of 3.675 W will be able to fully recharge the battery within 3 days after 3 days of complete darkness while the node is requiring power according to its worst case power usage.

The choice of solar panel is limited by the specifications of the LT3652 energy harvesting chip. The LT3652 allows a peak input power of 5 W, input voltages up to 40 V, and a maximum input current of 3 A. The minimum input voltage $V_{in}(start)$ for the LT3652 to begin harvesting energy is 3.3 V.

The solar panel chosen for this project is the DFRobot FIT0330 2 W solar panel. It has an open circuit voltage V_{OC} of 9.6 V and the voltage at the MPP is 9 V. Since an individual panel only supplies 2 W, two panels will be placed in parallel with each other to ensure an input power to the LT2653 energy harvesting chip of 4 W.

5.14 Energy Harvesting

To maximise the energy obtained from the solar panel, an energy harvesting module is placed between the solar panel and the battery.

The LT3652 by Linear Technology is an energy harvesting integrated circuit which can charge a battery to a set voltage from a variety of input sources. The device functions as a step-down battery charger which is capable of operating over a input voltage range of 4.95 V to 32 V. It offers features such as an input supply voltage regulation loop for peak Power Point Tracking for optimal energy extraction from a solar source as well as programmable over-voltage protection. The battery charge rate can also be controlled, up to a maximum of 2 A.

5.14.1 Energy Harvesting Design Requirements

As discussed in Section 5.11, the battery float voltage is chosen to be 4.1 V. Furthermore, the maximum charge current must be below the rated charge current of the battery. The LT3652 can support a charging current up to 2 A and the battery chosen for this project allows for a charge current up to 3.3 A. Therefore, the LT3652 will be programmed to regulate the charge current to its maximum value, 2 A.

The LT3652 uses a voltage regulation loop to steer a solar panels output voltage towards the point of maximum power when illumination levels cannot support the chargers set charge current. The maximum input voltage $V_{in}(max)$ is chosen at 20 V.

5.14.2 Detailed Design Procedure

The LT3652 is configured to set the float battery voltage to 4.1 V. This floating voltage is determined by the RFB resistors attached to the VFB pin. The required values for the resistors R_{FB1}, R_{FB2} and R_{FB3} are calculated in Equations 5.14.1, 5.14.2 and 5.14.3. The V_{FB} pin expects an input resistance ($R_{\|}$) of 250 kΩ to compensate for input bias current, while the I_{FRB} current is used by the LT3652 to check the battery voltage during the charging process. This current is chosen to be 10 µA so that it does not add unnecessary drain on the battery and is close to the minimum allowed the LT3652.

$$\frac{R_{FB2}}{R_{FB1}} = \frac{3.3\text{V}}{V_{flt} - 3.3\text{V}} \tag{5.14.1}$$

$$I_{FRB} = \frac{3.3\text{V}}{R_{FB2}} \tag{5.14.2}$$

$$R_{\|} = R_{FB3} + R_{FB1} \parallel R_{FB2} \tag{5.14.3}$$

$$\therefore R_{FB1} = 80\text{k}\Omega,\ R_{FB2} = 330\text{k}\Omega,\ R_{FB3} = 186\text{k}\Omega$$

Using these equations the values for R_{FB1}, R_{FB2} and R_{FB3} are calculated to be 80 kΩ, 330 kΩ and 186 kΩ respectively.

The LT3652 uses an input voltage regulation loop, which reduces the current drawn from the solar panel when the input voltage falls below a predetermined value. This feature is enabled by connecting a resistor divider from the V_{IN} to V_{IN_REG}. The LT3652 then varies the output charge current to maintain at least 2.7 V at the V_{FB} pin, irrespective of the solar panel output current. The power P_{IN} versus voltage V_{IN} curve for the solar panel at illumination levels from 100 W m^{-2} to 1000 W m^{-2} is shown in Figure 5.18. The figure shows that the voltage band which the LT3652 will maintain, to provide maximum power output, varies between $V_{REG(MIN)} = 8.65\text{V}$ and $V_{REG(MAX)} = 9\text{V}$. When the illumination on the solar panel becomes so low that the panel can no longer supply the charger at all, the LT3652 will reach the V_{SHDN} voltage at which time it will stop drawing current from the solar panel.

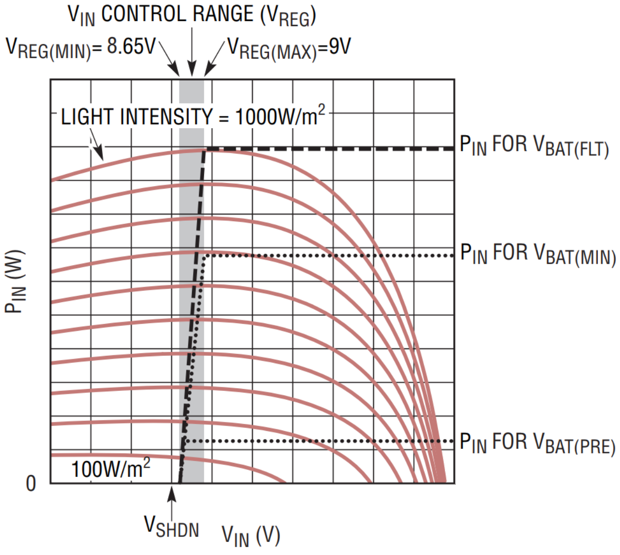


Figure 5.18: The LT3652 maintains the voltage input from the solar panel within the control range and lowers the output current depending on the light intensity on the solar panel [52].

The solar panel chosen for the project has a peak power output voltage $V_P(max)$ of 9 V. Choosing a value of 100 kΩ for R_{in2} and using Equation 5.14.4, the values for the the input resistor R_{in1} can be determined. The input to the charger usually requires a blocking diode between the input pin and the solar panel. However, for battery float voltages below 4.2 V, no diode is required.

$$\begin{aligned} \frac{R_{in1}}{R_{in2}} &= \frac{V_P(max)}{2.74} - 1 \\ \therefore R_{in1} &= 228\text{k}\Omega, \; R_{in2} = 100\text{k}\Omega \end{aligned} \tag{5.14.4}$$

Therefore R_{in1} will have a value of 228 kΩ, and R_{in2} will have a value of 100 kΩ. The resistor R_{sense} is chosen to limit the maximum charge current to the battery. The battery can withstand charge currents up to 3.5 A while the LT3652 can supply a charge current up to 2 A. Therefore the charge current I_{charge} will be set to the maximum 2 A. The R_{sense} resistor is chosen such that the desired maximum average current through the resistor creates a 100 mV drop between the sense pin and the battery. Equation 5.14.5 describes how the value of this resistor is determined.

$$R_{sense} = \frac{0.1}{I_{charge}} = 0.05\Omega \tag{5.14.5}$$

The inductor L1 has the primary purpose of minimising ripple current while charging is in progress. The ripple current is typically limited to a range of 25% to 35% of $I_{charge(MAX)}$. For these calculations the ripple current ratio $\triangle I_{ripple}$ will be limited to 25% of the maximum charge current. Equation 5.14.6 shows how the minimum inductance required to ensure this, is calculated.

$$\begin{aligned} L_{min} &= \frac{10 \times R_{sense}}{\triangle I_{ripple}} \times V_{flt} \times \left[1 - \frac{V_{flt}}{V_{in}(max)}\right] \\ &= \frac{10 \times 0.05}{0.25} \times 4.1 \times \left[1 - \frac{4.1}{20}\right] \\ &= 6.519\text{µH} \end{aligned} \tag{5.14.6}$$

The minimum value the inductor should be is 6.519 µH, therefore an inductor value for $L1$ of 8 µH was chosen.

The LT3652 supports a timer-based termination scheme where a charge cycle is terminated after a certain time period elapses. This feature is activated when a capacitor C_{TIMER} is placed between the timer pin 6 and ground. Once the preset time T_{EOC} has elapsed, the charging process will terminate if the battery voltage is within 2.5% of the float voltage. If the battery voltage has not yet reached this level, a new cycle will begin. If the charging cycle has completed and the battery voltage drops below 97.5% of the charge float voltage a new cycle will be started. A timing cycle (T_{EOC}) of 3 hours is recommended, and the capacitor value required to achieve this is calculated using Equation 5.14.7.

$$\begin{aligned} C_{TIMER} &= T_{EOC} \times 2.27e - 7 \\ &= 0.68\text{µF} \end{aligned} \tag{5.14.7}$$

All these calculated component values and the connections with the LT3652 are shown in Figure 5.19. Finally, the design permits either one or 2 solar panels to be connected in series. This provides extendability in the case that a single solar panel is not able to supply enough power to sufficiently recharge the battery in practice.

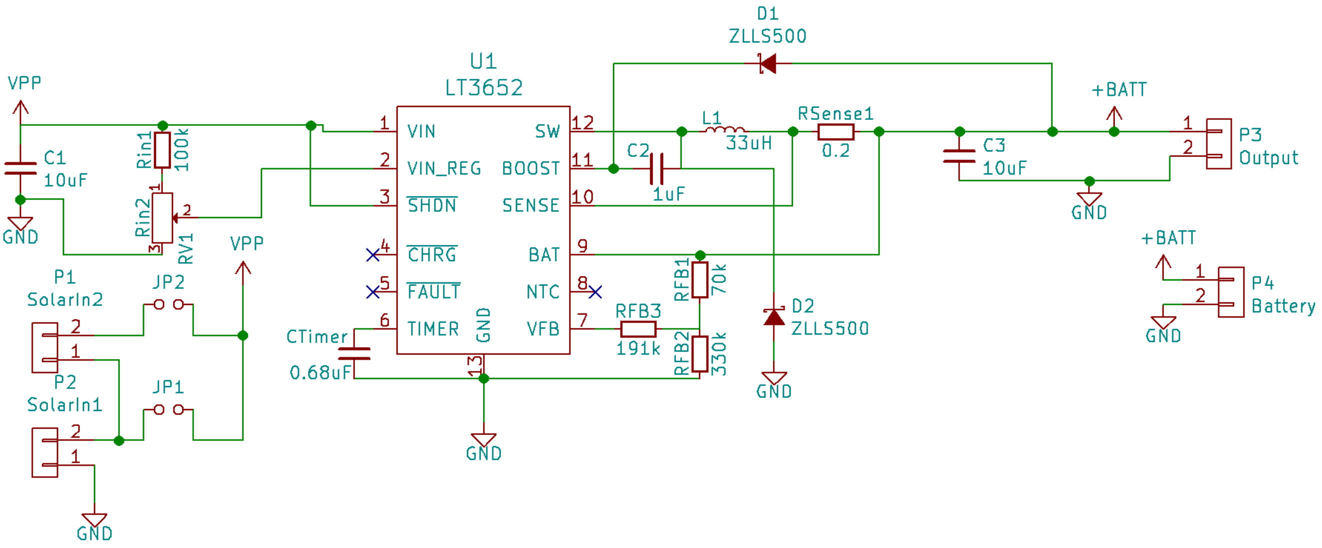


Figure 5.19: Schematic of the energy harvesting module using the LT3652 Energy harvesting integrated circuit.

5.15 PCB

Two printed circuit boards (PCBs) were designed for the WSN node and for the power supply. The PCB for the WSN node connects the MCU, RF module, GPS, GSM, temperature sensor, SD card and all supporting electronics. The power supply PCB connects the solar battery charger, voltage regulator and all supporting electronics. The output of the power supply PCB is connected to the inputs of the WSN node PCB through eight wires which are organised into a single port. This allows the WSN node to be isolated from the solar charger, solar panel inputs and the battery.

5.15.1 WSN Node PCB

This PCB was designed as a prototype. Therefore, all the modules are connected to the board with headers, and can be easily replaced if they break or if a new module is required. This will allow future configuration of the modules on the board to be done easily. A micro SD card holder is soldered to the PCB. An 8 port slot allows a connection from the power supply PCB to be connected to this PCB, which includes the power and ground lines as well as the monitoring lines. An illustration of the connections between the PCBs is shown in Figure 5.20.

For the case that the power supply PCB is not used, there is a dedicated 3.3 V input for the PCB. An image of the WSN node PCB is shown in Figure 5.21.

5.15.2 Power Supply PCB

All the electronic components for the solar charger and voltage regulator are soldered directly to the power supply PCB. The PCB allows two solar panels to be connected in series. There is also a dedicated input for the battery. Furthermore, a dedicated output from the 3.3 V regulator is also provided on the board.

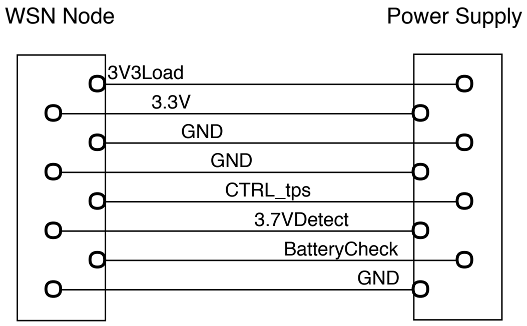


Figure 5.20: Connections between the WSN node and power supply. An 8 port slot allows the connections between the PCBs to be easily managed.

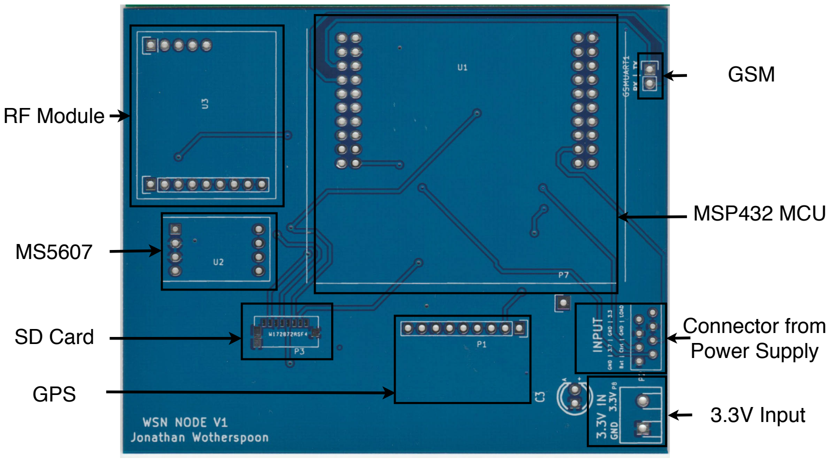


Figure 5.21: WSN node PCB.

The PCB has been laid out in such a way that the solar panel input, battery input, and regulated 3.3 V output are all separated. This PCB has a matching port connector to that of the WSN node PCB, through which the power, ground and monitoring lines can be connected to the WSN node PCB. The power supply PCB can be seen in Figure 5.22.

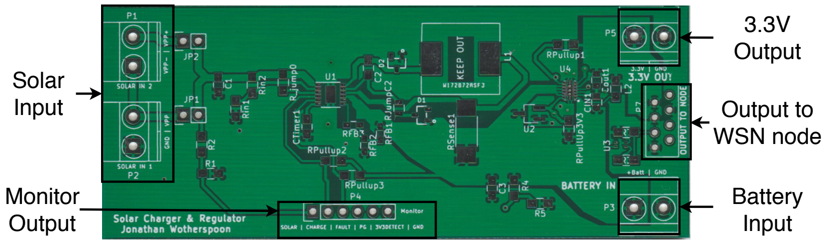


Figure 5.22: Power supply PCB. The PCB has inputs for 2 solar panels, and a battery. It has the eight port connection to the WSN node PCB and a dedicated regulated 3.3 V output. It also contains outputs for several monitoring pins.

This PCB also has output pins for several monitoring lines. These monitoring pins can be used to debug any future problems, or for future tests of the power supply. The outputs include the voltage from the solar panels, the charge and fault pins of the LT3652, the power good pin from the voltage regulator, the 3.3 V voltage detector and a dedicated ground pin.

5.16 Packaging

The two PCBs, battery and all electronic equipment are placed inside a waterproof plastic enclosure as shown in Figure 5.24. The enclosure has dimensions of 200 by 120 by 75 mm. The cables for the antenna and the solar panels are placed through waterproof connectors into the enclosure. The enclosure can be attached to a pole or placed on the ground, and in Figure 5.23 an assembled node is connected to the a pole and placed on a roof. If the enclosure is mounted to a metallic rod, then some form of lightning protection must be added to the system. A possible solution would be to connect the metallic rod to a large mesh grid mounted under the earth. This should divert energy away from the enclosure directly into the ground.

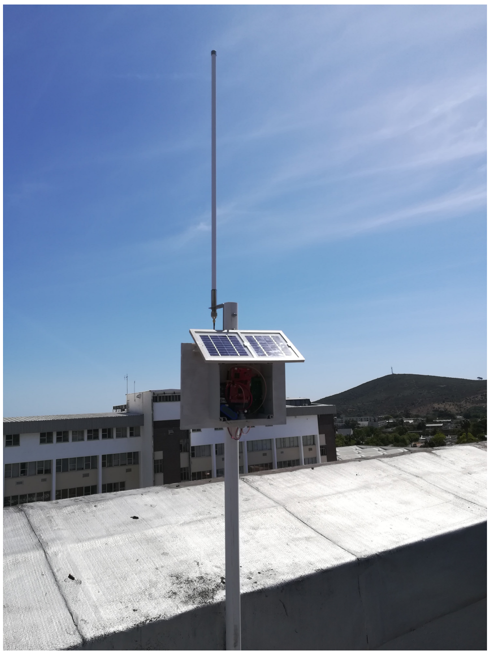


Figure 5.23: An assembled WSN node, with the antenna and solar panels attached.

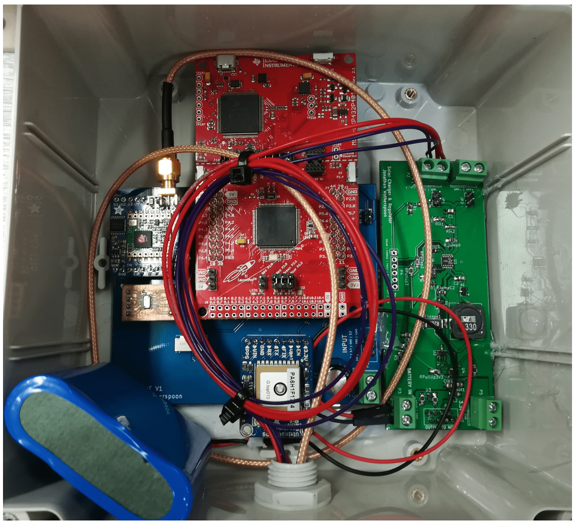


Figure 5.24: The components of the WSN node inside of an enclosure.

5.17 Summary and Conclusion

The hardware design of a WSN node and power supply was discussed in this chapter. The low power TI MSP432 was chosen as the MCU for the WSN node. The chapter described the design process for including several other components to each WSN node including storage, GPS, GSM and a temperature sensor.

The design of the power supply was discussed next. The power supply consisted of a battery, solar charger and voltage regulator. Four power profiles were created to estimate how much power a mesh and server node required during normal and extreme conditions. The power required by the extreme mesh power profile was then used to calculate a battery capacity which could power a WSN node for 3 days.

Battery monitoring hardware was designed to alert the WSN node when the battery falls below specific threshold voltages, and to disable the voltage regulator completely if the battery voltage falls too low. A 4 W solar panel was then chosen for its ability to recharge the chosen battery to full capacity after 3 days with the WSN node requiring maximum current.

Finally a PCB was designed for the WSN node and the power supply. The outputs from the power supply PCB can be connected to the WSN node with a single 8 port cable. This allows the inputs from the solar panels and battery to be separate from the regulated 3.3 V voltage which is supplied to the WSN node.

In the next chapter, the design of the multi-hop ad-hoc routing protocol for the WSN will be discussed.

Chapter 6

Ad-hoc Networking Protocol

This chapter will describe the design of the ad-hoc networking protocol for the outdoor wireless sensor network (WSN). In it, three layers of the ad-hoc network are discussed, namely, the physical layer, the Medium Access Control (MAC) layer and the routing layer. The requirements of each layer, implementations made during previous research, and the details of the chosen protocol will all be discussed.

An ad-hoc WSN is a decentralized wireless network, which forgoes the need for pre-existing infrastructure, such as the routers and access points associated with traditional wireless networks [53]. This type of network allows nodes to join the network on-the-fly, eliminating the need for complex network administration or infrastructure. Due to the ability of the nodes of an ad-hoc WSN to be mobile, the topology of the network can dynamically change over time and the routing protocol will need to compensate for this.

6.1 Network Requirements

The protocols and layers that are chosen for this network will need to comply with the following requirements:

- Nodes in the WSN can be configured as either a mesh node or a server node.
- All WSN nodes must be able to receive communication from the on-animal tags.
- The network must allow multi-hop communication, with messages able to pass through multiple mesh nodes before arriving at a destination (server) node.
- WSN nodes should always be ready to receive data from an on-animal tag.
- If a route breaks in the network, the routing protocol should be able to detect that break and find an alternative route to the destination.
- A node should be able to enter or exit the network without causing permanent disruption to the network.

With these requirements set out for the network, we will discuss the classification of nodes in the network in more detail. We will also discuss the challenges that will need to be overcome by the network due to the placement of the nodes in an outdoor environment. Finally, we will discuss the layers of the network.

6.1.1 Classification of Nodes

By definition an ad-hoc WSN considers its nodes to be randomly placed and therefore the network requires no fixed infrastructure to facilitate communication. Our network differs slightly from this definition in that it has two types of nodes: WSN nodes and on-animal tags.

The WSN nodes, can be configured as either a WSN server node or a WSN mesh node. WSN nodes, when deployed formally, will be implemented on fixed locations. While it is possible for them to be moved to different locations, due for example to losses accrued to an improperly placed antenna or the requirements of solar panel orientation, it is recommended that they are left in a fixed position on fixed infrastructure.

The WSN mesh nodes (hereinafter referred to as mesh nodes), are able to receive data from on-animal tags or from other WSN nodes, but will not store any data that they receive. Instead, upon receiving data, they will find the shortest route to a WSN server node and pass the information along that route.

The WSN server nodes (hereinafter referred to as server nodes), will receive data from on-animal tags, or other WSN nodes, and will either store this data for a user to retrieve later, upload the data via GSM to a database or via UART to a local storage device.

The second type of node is the on-animal tag, which is attached to the animal itself. An on-animal tag moves with the animal and cannot form part of any fixed WSN infrastructure. The on-animal tags will implement the same physical and MAC layers as the WSN nodes, and are therefore able to communicate with them. While it is possible that the on-animal tags be used to route packets in the network, this functionality was not considered for this project.

6.1.2 Node Addresses

Each node is given a one byte integer address ranging from 0 to 254. The address value 255 is reserved as this is used only for broadcast messages. Both WSN nodes and on-animal tags must have an address, and addresses cannot be repeated in the network. Therefore it is possible to have 255 WSN nodes, or on-animal tags in one network. It is possible to extend the node address to two bytes, but this will increase the overhead required during sending. A shortened list of possible addresses that can be assigned to nodes is given in Table 6.1.

Table 6.1: List of addresses which can be assigned to nodes. Address 255 is reserved for a broadcast message and therefore cannot be assigned to a node.

Address	Hex	Used For	Reservation
0	0x00	Node Address	Unreserved
1	0x01	Node Address	Unreserved
2	0x02	Node Address	Unreserved
...	...	...	...
254	0xFE	Node Address	Unreserved
255	0xFF	Broadcast Message	Reserved

6.1.3 Challenges Facing Our Ad-hoc WSN

In our ad-hoc WSN, there are several challenges which need to be addressed. These include the compromise between robust MAC and routing protocols, low power usage and free space propagation loss. A further challenge is the need to conserve power on both the WSN nodes and on-animal tags, as each are powered by batteries.

The following challenges were identified for our network and each of them will need to addressed for the network to function optimally.

1. Due to the free movement of the on-animal tags between WSN nodes, it is possible that a tag will be within range of multiple nodes at any time. In this situation, the network should provide priority to those nodes which are closest to a server.
2. The network is expected to be deployed in large outdoor areas which means it will generally not operate in free space, but will experience significant additional losses due to the diffraction and multi-path conditions caused by general or localised topography. External interference is also possible. Therefore, the MAC protocol should make provision for lost packets, and allow for packets to be retransmitted or re-acknowledged as required.
3. Because the WSN nodes use batteries as their only source of power, it is possible that a node will need to power down in order to conserve energy. When this occurs, the affected node should be able to inform its neighbours that it is powering down, so that they can disable all affected routes. Furthermore, once power has been restored, the node must alert its neighbours so that it can be reintegrated into the network.
4. It is also possible that a node fails due to physical damage (by some wild animal, or sabotage perhaps), and that the network consequently experiences a breakage in a route. The routing protocol must be able to detect the breakage, and to discover a new route for communication.
5. Although not a direct requirement by the WSN, the MAC protocol should allow the power usage of an on-animal tag to be minimised when it is not in range of a WSN node. Therefore, the MAC protocol should allow the on-animal tag to detect when it is not in range of any WSN nodes, without excessive energy expenditure.

All of these issues need to taken into consideration when choosing the layers of the ad-hoc WSN.

6.1.4 Layers of a WSN

Successful communication between nodes is aided by multiple protocol layers. The layers utilised in this network, shown in Figure 6.1 with their Open Systems Interconnection (OSI) numbers alongside, are the physical layer, MAC layer and network (routing) layer. An application layer (software operating system) is built on top of all those layers but is responsible for the high level workings of the node and not specifically the communication of the WSN. The layers are stacked on top of one another, so that each layer can be independently changed without affecting the layers below or above it.

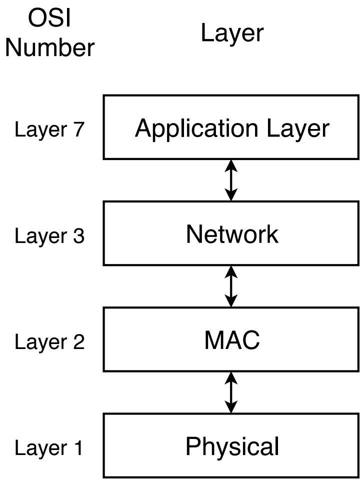


Figure 6.1: The networking layers of a WSN node, with the numbers assigned to them according to the OSI model. In order to ensure optimal performance and low power usage, the functions in each layer are optimised for our specific application. Image adapted from [54].

- The physical layer interacts with the hardware and software of the RF module directly. Therefore, if the physical layer is changed at a later stage, a new RF module could replace the current module.
- The MAC layer is a protocol which allows transactions to occur between two nodes, while attempting to interfere with the communication of other nodes around it as little as possible. The MAC layer is responsible for robust communication between two nodes, and does not have a view of the entire route which the message must take to reach it's destination.
- The network layer relates to the protocols put in place to successfully send a message across multiple nodes from a source to the desired destination. The routing layer can also discover new routes to a server or detect breaks in a route and fix them.

This chapter will describe each layer individually. Before the MAC and network layers are chosen, approaches proposed by other researchers will be discussed. An implementation, which satisfies the requirements set out in Section 6.1, will then be described.

The physical layer will be discussed in Section 6.2, the MAC layer in Section 6.3 and the network layer in Section 6.4. The application layer will be discussed separately in Chapter 7.

6.2 Physical Layer

The components of the physical layer are determined by the LoRa capable RFM96W radio module chosen in Chapter 4. While some of the basic components may be similar to other radio modules, the RFM96W has some which are specific to itself.

6.2.1 Basic Functionality Components of the RFM96W

When receiving a signal, the RFM96W transceiver module first amplifies the received RF signal using an low-noise amplifier (LNA). Following the LNA, the conversion to differential is made to improve the second order linearity and harmonic rejection. The resulting signal is down converted to in-phase and quadrature components at the intermediate frequency by the mixer stage. A pair of sigma delta ADCs then perform data conversion, with all subsequent signal processing and demodulation performed in the digital domain [55]. An image showing the stages from antenna input till the storage of the message in the data buffer is shown in Figure 6.2.

The RFM96W utilises a half duplex communication system whereby the module is able to send and receive signals on the same medium but not at the same time. The RFM96W has 2 oscillators which produce the local oscillator frequency for both the receiver and the transmitter respectively. One generator covers the lower UHF bands (up to 525 MHz) while the other covers the upper UHF bands (from 860 MHz onwards) [55]. The Phase Lock Loops (PLL) are optimised for low lock time, and fast auto-calibration.

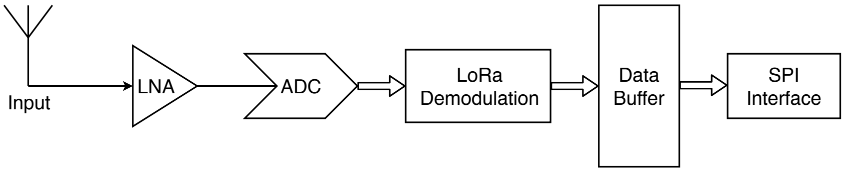


Figure 6.2: The basic components that make up the receiving for the RFM96W module. All analogue lines are shown as thin lines, while everything completed in the digital domain is shown with big arrows. Image adapted from [55].

Three distinct RF power amplifiers are featured on the RFM96W module. Two of these deliver up to +14 dBm output power and are unregulated for high power efficiency. They are connected directly to their respective RF receiver inputs via a pair of passive components to form a single antenna port high efficiency transceiver. One of these amplifiers is designed specifically for the lower UHF bands while the other is intended for the upper UHF bands. The third power amplifier, which can deliver up to +20 dBm output power, requires a dedicated matching network on the output. Unlike the two high efficiency amplifiers, this third high stability amplifier covers all frequency bands that the frequency synthesizer can address. In our case, the output power of the RF module is set to 20dBm, therefore, the high stability amplifier is used.

6.2.2 Spread Spectrum

LoRa utilises a technique to encode data before transmission known as chirp spread spectrum [56]. Spread spectrum uses wideband noise-like signals, which are difficult to detect or demodulate, to spread a narrowband signal over a wider range of the spectrum [57]. This spread signal can be transmitted at a lower spectral power density, and the spread energy makes it less prone to interference from narrow band signals. Similarly, a spread spectrum signal is much more difficult to jam, as the jammer will have to know the exact sequence with which the signal is spread to block it thoroughly.

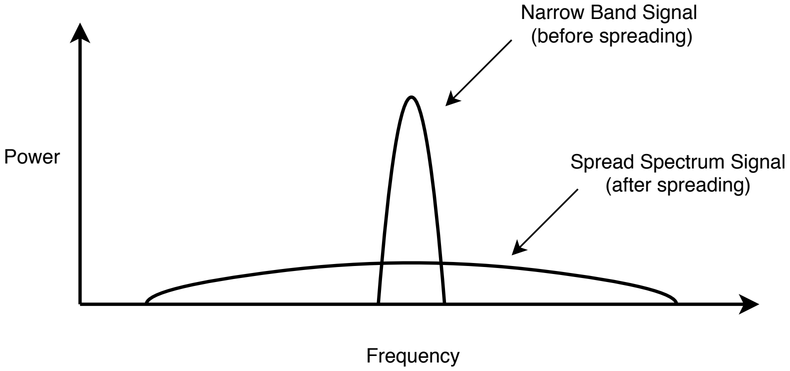


Figure 6.3: An example of a narrowband signal and a spread spectrum signal. Image adapted from [57].

LoRa is based on the chirp spread spectrum, which uses a wideband modulated chirp pulse to encode the information. The chirp is a sinusoidal signal whose frequency increases or decreases over time, usually with a polynomial expression for the relationship between time and frequency. The chirp spread spectrum is immune to the Doppler effect as well as being very difficult to detect or intercept at low power [58].

6.2.3 CRC

The Cyclic Redundancy Check (CRC) is an error-detecting code used for digital communication to detect accidental changes to raw data during transmission or storage. A block of data will be followed by a short check value, which is based on a polynomial division of the data itself. On receiving the data, the calculation is repeated and if the values do not match, corrective action can be taken.

The RFM96W has a built-in CRC error checking process, which both appends the polynomial check to the end of a payload before transmission and removes it to perform the data integrity check after reception. This built-in mechanism could be used to determine whether the current transmission link requires some form of error correction. Currently however, the CRC check is not used by our implementation.

6.2.4 Forward Error Correction

Forward error correction (FEC) is a mathematical technique for controlling the number of errors in a transmission over a noisy communication medium. The sender encodes the message in a redundant way by using error-correcting code. This enables the receiver to identify and correct errors without the need for retransmission. The RFM96W has a built in forward error correction technique used for transmissions. The code rate for the FEC is set to a user defined value of 4/8. This means that for every 4 bits of data sent by the user, 4 bits are added for error correction making a total of 8 bits sent. The user is capable of setting code rates that vary from 4/8 to 7/8.

6.3 MAC Layer

Next we discuss the Medium Access Control (MAC) layer. The MAC protocols are used to control access to a shared communication medium for transferring data. MAC protocol determines who is allowed to utilise the shared medium when there is a contention during communication. It is responsible for resolving conflicts for channel access and data collisions that may occur on the shared medium. In our case the shared medium is the free space through which the RF signals propagate. Therefore, if nodes are placed densely together, the chance that collisions will occur is higher than when they are sparsely distributed. The choice of MAC protocol directly affects both the reliability of the network as well as the energy consumption.

The MAC protocol chosen for this project must be suitable for use in the WSN nodes as well as the on-animal tags. Therefore, the MAC protocol must facilitate communication from WSN node to WSN node as well as communication from the on-animal tag to WSN node. Since energy efficiency on the on-animal tag is of particular importance, the MAC protocol should prioritise communication from the on-animal tag to a WSN node, and should be as computationally efficient as possible. To allow further power saving, the MAC protocol should allow the on-animal tag to check for the presence of WSN nodes within its range before transmitting the complete DATA packet.

This section will give examples of typical challenges which a MAC protocol should overcome, as well as describe several popular MAC protocols used in WSNs. The suitability of each protocol for use in this project will be considered and finally a MAC protocol will be chosen. The complete description of the MAC protocol as well as the timing between packets and all required control variables will be discussed. The section will conclude with improvements made to the MAC protocol during testing.

6.3.1 Challenges Faced by WSNs

Some of the common challenges which a MAC protocol must overcome are: the hidden node problem, the exposed node problem and ensuring communication despite node mobility.

6.3.1.1 The Hidden Node Problem

The hidden node problem is illustrated in Figure 6.4. Node A and node C can both communicate with node B. However node A and node C cannot communicate with each other as they are not within range. When node A and node C try to communicate with node B simultaneously, their transmissions will interfere, causing a conflict at node B. The MAC layer must resolve such conflicts when they occur, and ensure that communication with node B becomes possible with minimal delay.

6.3.1.2 The Exposed Node Problem

The exposed node problem is illustrated in Figure 6.5. In this case, there are four nodes, A, B, C and D. Node B is currently communicating with node A, while node C would like to initiate communication with node D. Node C can detect that node B is communicating and will therefore wait until this transmission is complete, even though the communication from node C to node D would not be affected if C initiated communication immediately. This is known as the exposed node problem and should be addressed by the MAC protocol. However, the exposed node problem is not as important as the hidden node problem,

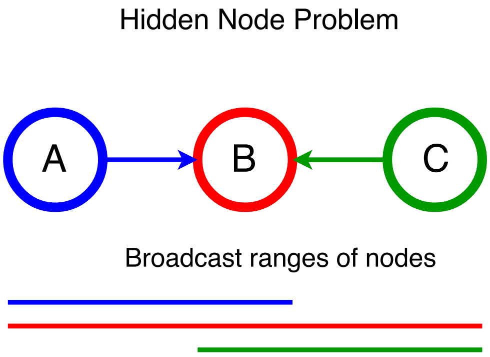


Figure 6.4: A visual representation of the hidden node problem. Node A and node C are both within communication range of node B, but not each other.

because it leads only to a small delay in the network and not a collision, which is associated with greater delays.

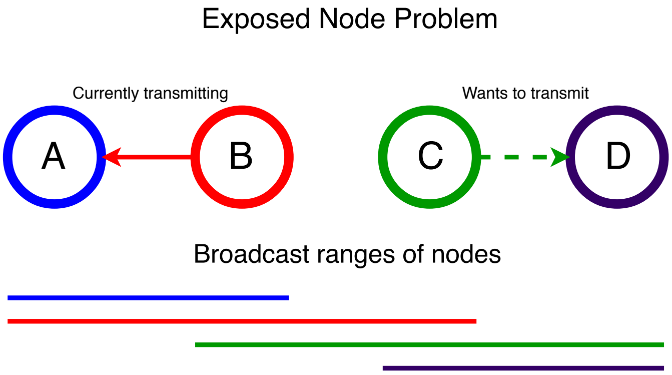


Figure 6.5: A visual representation of the exposed node problem. Node B is communicating with node A, and node C intends to communicate with node D.

6.3.1.3 Mobility of Nodes

The mobility of nodes can pose a further challenge to the MAC protocol. If a node moves during transmission, some packets can become corrupted. In this situation the MAC protocol must determine which packets have been corrupted and how to resend them. Node mobility can also lead to route changes, which the MAC protocol must detect and the routing algorithm must correct [59].

6.3.2 Choosing a MAC Protocol

There are many ways to classify MAC protocols for wireless networks. For simplicity, we will consider two MAC protocol families commonly used for ad-hoc WSNs. These are contention-based, and contention-free MAC protocols. A contention-based protocol always attempts to send a message immediately, and only when a collision occurs does it take corrective action. Contention-free protocols, on the other hand, assign time slots within which each node is permitted to communicate. The method of time slot assignment differs between protocols. Figure 6.6 shows a classification tree of these two MAC protocol families, including a few sub classifications and examples of each. This section will discuss the difference between these protocols as well as their suitability for the outdoor WSN that we consider. Finally, a MAC protocol will be chosen.

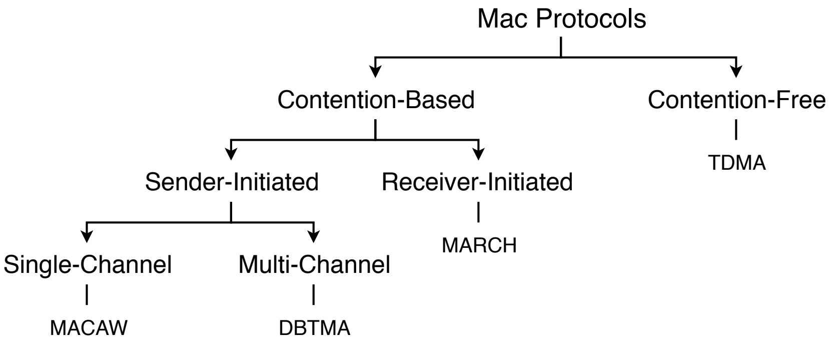


Figure 6.6: A classification of MAC protocols for ad-hoc WSNs into contention-based and contention-free families, showing examples of each.

6.3.2.1 Contention-based Protocols

A contention-based MAC protocol allows nodes to initiate communication at will. If a collision occurs during communication, the protocol prescribes a method of recovery. Contention-based protocols can be divided into two distinct sub-types: sender-initiated and receiver-initiated. Sender-initiated protocols can be further divided into single-channel and multi-channel variants. We will discuss each contention-based MAC protocol shown in Figure 6.6 in this section, starting with the sender-initiated MACAW and DBTMA and then the receiver-initiated MARCH protocols. These protocols are widely used in networks where there is no way of synchronising time between nodes.

Multiple Access with Collision Avoidance for Wireless (MACAW) is an example of a single-channel sender-initiated contention-based MAC protocol [60]. In the MACAW protocol there are two nodes, A and B, and communication from node A to node B proceeds as follows.

- A Request to send (RTS) packet is sent from node A to node B.
- A Clear to Send (CTS) packet is sent from node B to node A.
- A DATA segment specifying the length of the data that will be sent from node A to node B is sent from node A to node B.
- The DATA packet itself is sent from node A to node B.
- An acknowledgement (ACK) that the DATA packet has been successfully received is sent from node B to node A.

If a collision occurs during this communication, each node backs off for a predetermined time period, after which the same communication procedure is attempted again. MACAW has several shortcomings, one of which is that it does not solve the hidden node problem. However, MACAW is easy to use and its RTS-CTS packets allow sender nodes to easily determine whether the node with which it is attempting to communicate is busy or not.

This protocol is suitable for an outdoor WSN as it requires only a single-channel for communication, and because the protocol can be designed to always prioritise time critical communication in the event of a contention.

Dual Busy Tone Multiple Access (DBTMA) is an example of a multi-channel sender-initiated MAC protocol. With DBTMA, the single-channel is split into two sub channels, a data sub-channel and control sub-channel. Data is transmitted on the data sub-channel while busy tones are transmitted on the control sub-channel. Two busy tones are used: a transmit busy tone, which indicates that the node is transmitting, and a receive busy tone, which indicates that a node is receiving [61]. The two narrow-bandwidth tones are implemented on the shared channel by separating them slightly on the RF spectrum. The protocol operates as follows between node A and node B.

- Node A senses the control channel to check whether a receive busy tone is active on the medium or not.
- If no busy tone is active, node A transmits a busy transmit tone on the control sub-channel and transmits an RTS on the data sub-channel from node A to node B.
- Once the RTS is sent by node A, it ceases transmission of the transmit busy tone and enters a waiting state.
- When Node B receives the RTS packet it transmits a receive busy tone on the control sub-channel, announcing to node A it is ready for the DATA packet.
- When node A senses the receive busy tone from node B, node A waits a mandatory time before sending the DATA packet from node A to node B on the data sub-channel. The time which node A waits before sending is to allow all other nodes within communication distance to abort sending an RTS.
- Once the message is received by node B, it ceases transmission of its busy tone.
- Any node within communication range which senses a busy tone on the control channel desists from initiating communication for a predetermined period.

The DBTMA MAC protocol would allow an animal-borne tag to quickly and easily determine whether any nodes are communicating, and therefore would not have to rely on sending a RTS. However, this protocol requires two sub-channels, and for the sending node the control channel needs to transmit a tone during the RTS transmission. This is impractical for an outdoor WSN, as the control channel will consume an unnecessary amount of energy for the on-animal tag.

An example of a receiver-initiated protocol is Multiple Access with ReduCed Handshake (MARCH). This protocol makes use of the broadcast characteristics of an omni-directional antenna to reduce the number of control messages required to transmit a DATA packet across multiple hops [62]. The proposed handshake mechanism is illustrated in Figure 6.7. The figure shows that in this protocol a RTS packet is only used for the first DATA packet of the stream from node A to node B. Node B replies to node A's RTS with a CTS packet, which is overheard by node C. Once the data transmission from node A to node B is complete, node C will send a CTS packet to node B. Node B will respond by sending the data to node C and this will continue throughout the network until the data arrives at the destination.

The RTS/CTS packets in the MARCH protocol include the following information: (a) the MAC address of the sender and receiver and (b) the route identification number. It is assumed that each node will sense the channel and not transmit until the channel is free.

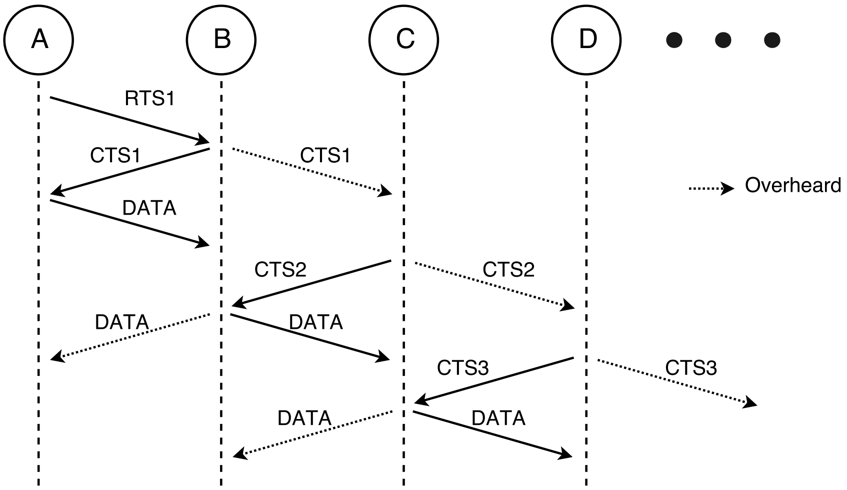


Figure 6.7: Handshake mechanism employed by the MARCH MAC protocol.

The MARCH protocol allows for a high throughput and low control overhead. However, it requires access to the routing information for the entire network. It would therefore only be possible to implement this protocol on the WSN nodes, as the on-animal tags do not have access to the routing information for the entire network. This means that separate protocols would be needed for the on-animal tag to WSN node communication, and for the WSN node to WSN node communication, which adds further complexity to the system.

6.3.2.2 Contention-free Protocols

Contention-free MAC protocols differ from the contention-based protocols in that there is no contest for bandwidth except during the reservation stage. In this way each node receives a fair amount of bandwidth. Once bandwidth is reserved, a node receives exclusive access to the medium for a set time period.

An example of this type of protocol is Time-Division Multiple Access (TDMA). In TDMA, data transmission is divided into frames [63]. A receiving node divides the time during which it receives transmissions into slots. The node communicating with the receiving node may transmit a frame only in its assigned slot. This protocol is reserved mostly for voice communication, such as mobile telephony.

In order for this MAC protocol to be utilised in our WSN, all nodes must be synchronised in time. This protocol would also need to be adapted to allow urgent messages to be assigned multiple slots in preference to regular messages. The TDMA protocol is also most suited for networks which need to continuously communicate data. The outdoor

WSN, however, only communicates data when a message is received from an on-animal tag.

6.3.2.3 Choice of MAC Protocol

The MAC protocol chosen for this project is a modified version of the MACAW contention-based protocol. The chosen protocol uses the same handshaking technique between nodes used by the MACAW protocol, but also includes a carrier sense check before transmission, in addition to the ability to send broadcast messages to multiple nodes. The DATA segment which specifies the length of the data packet is not used in our implementation in order to save energy, and reduce the number of packets per a message. If the ability to know the size of the DATA packet is required by future implementations, it is recommended that a single byte in the RTS packet be dedicated toward this.

A contention-based protocol was chosen as it allows the WSN nodes to receive communications from an on-animal tag at any time, without needing to wait for a scheduled slot. This is relevant in the scenario that the on-animal tag records an emergency condition and needs to alert the network immediately. A scheduled MAC protocol was not used as it requires very stable time synchronisation in order to properly coordinate the communication between nodes, and this is simply not possible due to the low power requirements of the on-animal tags.

Protocols requiring multiple simultaneous channels were not considered because the radio communication hardware chosen for the project does not allow multi-channel communication. Furthermore, multi-channel communication is associated with higher power usage, which is unfavourable for the on-animal tag or WSN nodes. The chosen protocol is sender-initiated, as this allows the on-animal tags to communicate emergencies or anomalies as soon as they occur, rather than requiring them to wait until data is requested from them.

6.3.3 Implemented MAC Protocol

The chosen MAC protocol is a slightly modified version of the MACAW protocol. This section will discuss the features of the chosen MAC protocol. It will give an overview of the communication between nodes, explain the control variables required during communication and describe examples of the MAC protocol during use. Finally, features which were added to increase message throughput, are described.

6.3.3.1 Overview

When a message is sent between nodes, whether it is between WSN nodes or from an on-animal tag to a WSN node, the handshaking protocol given in Figure 6.8 is followed.

In Figure 6.8, node A sends a message to node B. First, node A performs a carrier sense (CS) to determine whether the medium is busy, because another node is currently transmitting in its vicinity. Once the CS check is complete, node A sends a request to send (RTS) packet to node B. This RTS packet can be broadcast or sent directly to node B. If node B receives the RTS successfully, it replies to node A with a clear to send (CTS) packet. Once the CTS is received by node A, it sends its DATA packet to node B. If

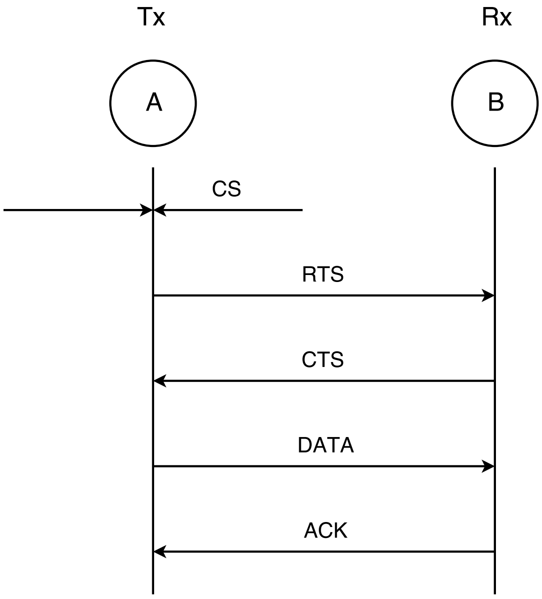


Figure 6.8: An overview of the handshaking that occurs between two nodes as prescribed by the chosen MAC protocol. Node A is sending a message to node B.

node B successfully receives the DATA packet from node A, it sends an acknowledgement (ACK) to node A.

If, during the CS, the sending node detects that the medium is busy, it will enter a waiting period before it checks the medium again. Likewise, if no CTS is received after a RTS has been sent, or no ACK is received after the DATA packet has been sent, then the sending node will wait for a set period of time and retry to send the packet. The timing of the MAC protocol will be explained in Section 6.5.

Hereafter in this chapter, a message will be referred to as data which has been sent from one node to another after completing the entire MAC protocol handshaking sequence. A packet will be referred to as the individual parts of the handshaking procedure. Therefore, only after the RTS, CTS, DATA and ACK packets have been successfully sent between node A and node B, will the message successfully be sent from node A to node B.

6.3.3.2 Timeouts, Attention and Overheard Packets

When a RTS is received by a node and it replies with a CTS, it enters a waiting state for the DATA packet. This state is known as 'attention', because the node is waiting to receive information specifically from the node who sent the RTS.

When a node is in attention and it receives a packet from a node other than the node which sent the RTS, it will discard this packet. This is known as an overheard packet.

Finally, if the node in attention does not receive a DATA packet from the node which sent the RTS before the waiting time expires, the MAC protocol will reset, and the node will only respond to a new RTS. This is known as a timeout, and is set to a value of 8 s.

6.3.3.3 MAC Control Variables

The MAC protocol requires specific control variables to be sent as part of each packet to allow it to determine the correct response. These variables form part of the control overhead of a packet. The control variables required by the MAC protocol are shown in Figure 6.9. Each variable is an 8 bit integer value.

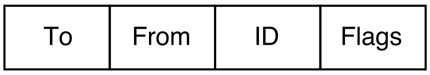


Figure 6.9: The control variables required by the MAC protocol.

Table 6.2 gives a short description of each variable shown in Figure 6.9. Complete descriptions follow below.

Table 6.2: MAC control variables and short descriptions.

Variable	Description
To	The address of the node to which the message is addressed.
From	The address of the node from which the message is being sent.
ID	Unique identification number of the message sent.
Flags	Special flags reserved by the MAC protocol.

To: this variable is used by a node to determine whether the packet is intended for it. A node will only accept incoming packets which have been broadcast or addressed to itself.

From: the address of the node from which the packet is being sent. When a node receives a RTS or DATA packet, it will use this variable to determine to which node the CTS or ACK packet should be sent. This variable is also checked by the receiving node, to ensure that the DATA packet it receives is from the node which sent the RTS.

ID: an unique integer identifying the packet sent. The ID is used by the receiving node to determine whether the packet has been received already. Each node maintains a record of the ID of the most recent packet it received from a node. The sending node increments the ID each time a packet is sent. Therefore, when a receiving node receives a packet it will check the ID of the received packet against that of the previously received packet from that node, to determine whether or not the packet has already been received.

The ID is also used by the transmitting node to determine whether a CTS or ACK packet is associated with the most recent RTS or DATA packet. When a node sends a RTS, the receiving node must use the ID of the received RTS as the ID for the CTS. Similarly, once a DATA packet has been successfully received by a node, the node must reply with an ACK which has the same ID as the received DATA packet.

Flags: used by the MAC protocol to determine what type of packet has been sent and therefore how it should be replied to. A list of MAC protocol flags is given in Table 6.3.

6.3.3.4 MAC Protocol Flags

Each packet which is sent is accompanied by a specific flag set by the MAC protocol to determine the type of packet. The flags, their values and descriptions are given in Table 6.3.

Table 6.3: Flags used by the MAC protocol.

Flag	Value	Description
RTS-Flag	0x40	Request to Send.
CTS-Flag	0x0F	Acknowledge receipt of a RTS packet.
ACK-Flag	0x80	Acknowledge receipt of a DATA packet.
Tag-Node-Flag	0x08	DATA packet originating from an on-animal tag.
Node-Node-Flag	0x04	DATA originating from a WSN node.

The Request To Send flag (RTS-Flag) indicates a RTS packet, and is sent by a node when it would like to check whether a node is available for data transfer.

The Clear To Send flag (CTS-Flag) indicates a CTS packet and is used to acknowledge that a RTS packet was correctly received by a node.

The Acknowledgement flag (ACK-Flag) is used to acknowledge that a DATA packet was correctly received by a node.

The Tag-Node-Flag and the Node-Node-Flag are used by the MAC protocol to identify the type of node the DATA packet is being sent from, and how the routing protocol should respond to it. All DATA packets from an on-animal tag must have the Tag-Node-Flag set as the WSN node can then route this DATA packet to the nearest server. The Node-Node-Flag is used to identify messages sent between nodes.

6.3.3.5 MAC Communication Overview

To illustrate a generic transmission between node A and node B, all the exchanged packets are illustrated in Figure 6.10. In the figure, node A is sending a message to node B.

The RTS packet is addressed to node B from node A, has an ID of x and the RTS-Flag set. The CTS packet is addressed to node A from node B, also has an ID of x and the CTS-Flag set. The DATA packet is addressed to node B from node A, has an ID of x+1, flag set to y and the payload is attached. The ID of the DATA packet has incremented from the previous transmission to x+1, and the payload can be up to 251 bytes in length. The flag is set to y, because the flag for this packet depends on the type of node the message originates from. If the message originates from an on-animal tag, y will be set to the Tag-Node-Flag, but if the message originates from a WSN node, y will be set to the Node-Node-Flag. Finally, an ACK packet is addressed to node A from node B, with an ID of x+1 and the ACK-Flag set.

	To	From	ID	Flags	
RTS	B	A	x	0x40	
CTS	A	B	x	0x0F	
DATA	B	A	x+1	y	Payload
ACK	A	B	x+1	0x80	

Figure 6.10: The MAC packets used during a generic transmission between node A and node B. In this example, node A is sending a message to node B.

6.3.3.6 Example: Node-Node Communication

A complete example of the packet transfer between two WSN nodes, A and B, is shown in Figure 6.11. The figure shows all the information contained within the packets during each transmission, as well as the checks that are performed by each node when a packet is received.

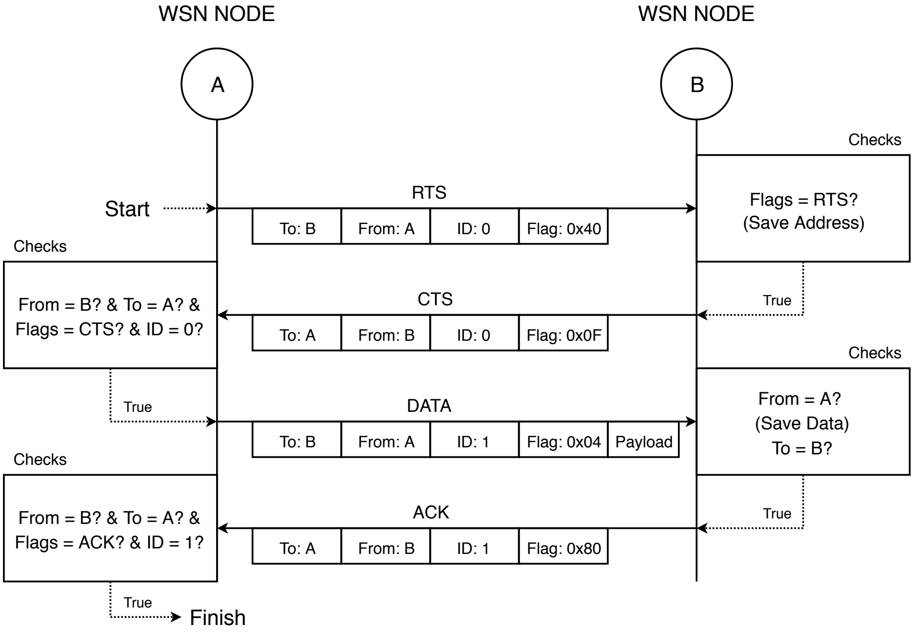


Figure 6.11: An illustration of a complete exchange from node A to node B, indicating all checks performed by the MAC protocol during the exchange. In this example, WSN node A is sending a message to WSN node B.

Node A begins by sending a RTS packet to node B. When this RTS packet is received by node B, it checks that the flag is set to 0x40, the hex value reserved for the RTS-Flag. If this is true, the node saves the address from which the packet was sent for later use and sends a CTS packet to node A. Once node A receives the CTS packet from node B, it checks that the sender of the packet has the same address as the address to which the RTS packet was addressed. It also checks that the CTS packet was addressed to itself, in this case address 'A'. Furthermore, it verifies that the CTS-Flag is set, and that the packet ID matches the ID of the RTS packet it sent. If all these checks succeed then node A sends the DATA packet to node B, with the payload attached.

When node B has received the DATA packet it checks that it is from node A. If it is, it saves the payload portion of the packet. If the DATA packet was addressed to node B, it responds with an acknowledgement to node A. If the checks fail, no acknowledgement is sent.

Finally, Node A checks that the ACK packet is addressed to itself, that it has been sent from node B, that the ACK-Flag is set, and that it has the same ID as the DATA packet. If all these conditions are true then the message has successfully been sent from node A to node B, and both nodes will continue with their respective application cycles.

6.3.3.7 Example: Tag-Node Communication

An example of a message transfer from an on-animal tag to a WSN node is shown in Figure 6.12. The figure shows all the information contained within the packets during each transmission, as well as the checks that are performed by each node when a packet is received. Differences from the WSN node to WSN node communication shown in Figure 6.11 are indicated in bold. In this example the on-animal tag, who is initiating communication, has the address 'C', while the WSN node, which is receiving, has an address 'D'.

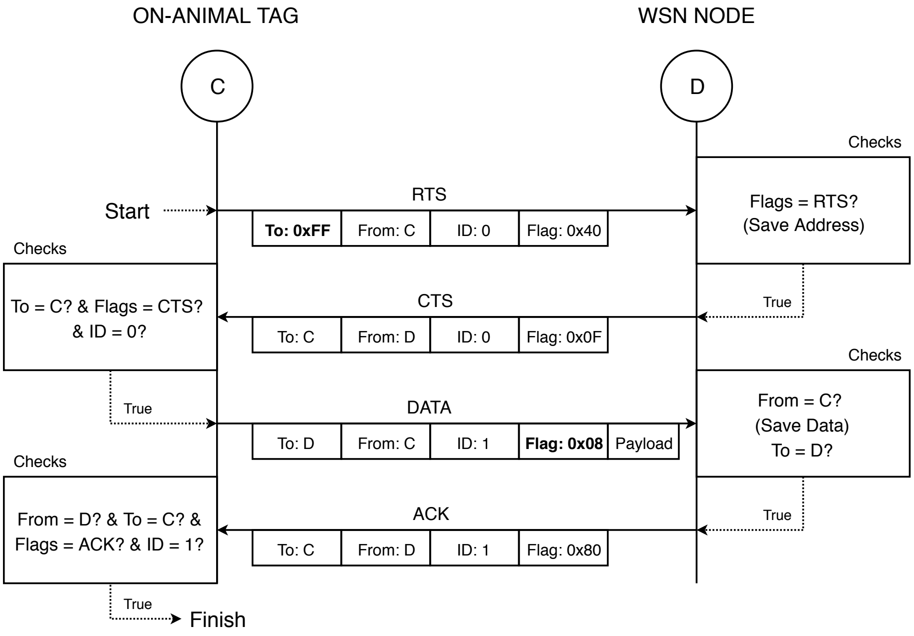


Figure 6.12: An illustration of a complete exchange from node C to node D, indicating all checks performed by the MAC protocol during the exchange. In this example, on-animal tag node C is sending a message to WSN node D. Differences to the WSN node to WSN node example in Figure 6.11 are indicated in bold.

The packet information in this example is very similar to that for communication between two WSN nodes. The only differences are that the on-animal tag sends the RTS to the broadcast address, and that it sets the Tag-Node-Flag for the DATA packet. The reason for broadcasting the RTS is because, at transmit time, the on-animal tag does not know the identity of the nearest nodes and simply wants to pass its message to any node within listening distance.

When WSN node D receives the RTS sent by the tag, it replies with a CTS packet addressed to the tag. The tag receives the CTS packet and, after passing the checks required by the MAC protocol, sends its DATA packet to node D. However, unlike the RTS which was broadcast, this packet is addressed specifically to node D. The Tag-Node-Flag flag in this DATA packet is set to a hex value of 0x08. The address of node D is saved for later use.

When node D receives the DATA packet and it has verified that it was transmitted by node C, it saves the payload. If node D was the address of the DATA packet, it sends an acknowledgement packet to the on-animal tag. Once the ACK packet is received by the tag, and it has passed all the checks, the message has been successfully exchanged between the two nodes.

The reason that the tag broadcasts the RTS and then sends the DATA packet to the node which replies first with a CTS, is to mitigate the possibility of duplicating data in the network. By sending the DATA packet to the first responding node, it is ensured that the data is sent only to one node in the network. How that node is determined will be explained in Section 6.3.4.1.

6.3.3.8 Example: Node Broadcast

A node (on-animal tag or a WSN node) may broadcast a message to any WSN node within listening distance. In the example in Figure 6.13, node E will be broadcast a message which both node F and node G receive. The figure illustrates all packet information and checks performed by the MAC protocol. Differences to the WSN node to WSN node communication shown in Figure 6.11 are indicated in bold.

Node E sends an RTS packet to the broadcast address. WSN nodes F and G are within receiving distance and both receive this RTS. Both reply with a CTS packet addressed to node E. It does not matter which CTS packet node E receives first, but in this example it receives the CTS packet first from node F and ignores the later CTS sent by node G.

Once Node E receives the CTS packet, it checks that it is addressed to itself, that the CTS-Flag is set and that the ID corresponds to that used in the RTS packet. Even though the CTS packet from node G is discarded by node E, node G enters a receiving state, waiting to receive the DATA packet from node E.

Node E then transmits the DATA packet, again addressed to the broadcast address. Both node F and node G receive the DATA packet and check that it was sent by node E. If it was, both nodes save the payload sent in the packet. Both nodes check if the packet was addressed to them, and because it was addressed to the broadcast address, neither node replies with an acknowledge.

Unlike in the previous examples (Sections 6.3.3.6 and 6.3.3.7), where the initiating node would usually wait for an acknowledge packet, in this case the DATA packet was

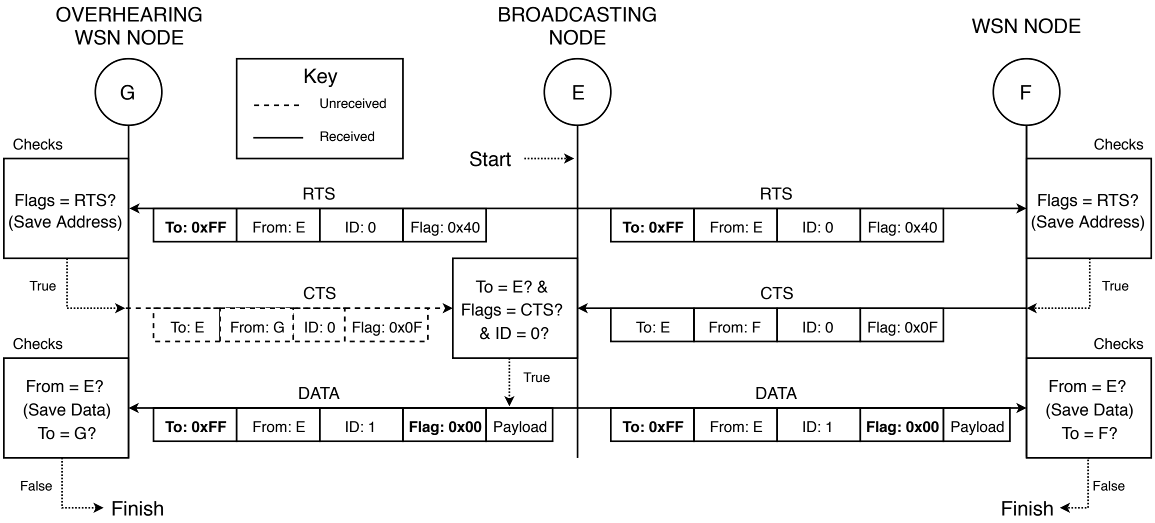


Figure 6.13: An illustration of a complete exchange from node E to node F and node G, indicating all checks performed by the MAC protocol during the exchange. In this example, node E is broadcasting a message which both node F and node G receive. Differences to the WSN node to WSN node example in Figure 6.11 are indicated in bold.

broadcast and node E assumes that the packet has been received immediately after sending the DATA packet. Nodes F and G both save the payload that was broadcast to them and then continue with their application cycles without sending an acknowledge.

6.3.3.9 Retransmission and Re-acknowledgement

After a RTS or DATA packet has been sent by a node, it waits for a CTS or ACK from the receiving node. If none is received within a certain period, the sending node will retransmit the packet. This is known as a retransmission, and a single packet can be transmitted a total of 4 times before the communication is considered to have failed and the message is discarded.

When a receiving node has successfully received a packet, and has replied with a CTS or ACK to indicate this, but the sending node does not receive this confirmation, the sending node will retransmit the packet. When the receiving node receives the retransmission, it will detect that this packet has already been received send another CTS or ACK to the sending node. This is known as a re-acknowledgement. A node will only re-acknowledge packets addressed to itself.

6.3.4 Improvements to MAC Protocol

The following are improvements that were made to the MAC protocol during testing. An improvement is made to the MAC protocol of the WSN nodes, and the on-animal tags. Each improvement will be described in this section and results showing their positive effect on message throughput are discussed in Chapter 8.

6.3.4.1 Broadcast Reply Delay

When a broadcast RTS is received by a node, that node will delay for a certain time period before replying with a CTS. This delay, T_{delay}, is dependant on the RSSI of the received RTS, whether the node is a server or not, and on how many hops the node is from the nearest server node. T_{delay}, in milliseconds, is calculated in Equation 6.3.1.

$$\begin{aligned} T_{delay} &= T_{RSSI} + T_{rand} + B_{mesh} \times (N_{hops} + 1) \times (T_{RSSI}) \\ if\ T_{delay} &> 500,\ T_{delay} = 500\text{ms} \end{aligned} \tag{6.3.1}$$

T_{RSSI} is the absolute value of the measured RSSI of the received message, and can have a value which ranges from 0 to 137. T_{rand} is a random number between 0 and 137, while B_{mesh} is a boolean value which is 1 if the node is configured as a mesh node, and 0 if it is configured as a server. N_{hops} is the number of hops from the node to the nearest server. Hops will be explained further in Section 6.4.

According to Equation 6.3.1, if the node is a server, then the delay will be the absolute value of the measured RSSI plus a random number between 0 and 137. If the node is a mesh node, then the number of hops the node is from a server node plus 2, will be multiplied by the absolute value of the measured RSSI. A random number between 0 and 137 will be added to this. If after this the value of T_{delay} is greater than 500, it will be set to 500.

The reason that a mesh node has a longer T_{delay} than a server node, is that the network will always want to route messages to a server. Therefore, if a message is broadcast and is from an on-animal tag, the network would expend less energy if the message was sent directly to a server node. If there is no server within range of the transmitting node, then the mesh node which is nearest to a server will reply first and the message can be sent to that node. If two or more nodes are the same number of hops from a server node, then the node which is nearest to the transmitting tag will reply first.

T_{rand} ensures that nodes which have a very similar measured RSSI, but are hidden from each other, will not reply at the same time. Therefore, their replies will not interfere with each other at the node requesting the RTS, thereby solving the hidden node problem in the MAC protocol. This is true for both server and mesh nodes.

6.3.4.2 On-animal Tag Address Saving

A further improvement to the MAC protocol is the use of address saving on the on-animal tags. When an on-animal tag communicates for the first time, it will broadcast the RTS to all WSN nodes. Once it receives a CTS, it will save the address of the originating node and send the DATA packet directly to that node. The next time the on-animal tag sends a message, it will first attempt to send a RTS to the WSN node with which it last communicated. If this fails, a subsequent RTS will be broadcast.

Due to the ability of the LoRa RF modules chosen in the project to communicate over long distances, as demonstrated in Chapter 4, the possibility of an animal-tag being in range of a WSN node for multiple transmissions is high. The introduction of address saving should reduce the number of timeouts and overheard messages on the remaining WSN nodes.

6.3.5 Conclusion

The functionality supported by the described MAC protocol allows the on-animal tags to move freely and to communicate their data to any idle WSN node within range. The MAC protocol also solves the hidden node problem by addressing most communication to a specific node and, in the case of broadcast messages, adding a delay before a node replies. Finally, the MAC protocol requires only a small control overhead of 4 variables in each packet and a total of 4 packets for reliable node-to-node communication. The developed MAC protocol is therefore well suited to the particular requirements of our outdoor WSN.

6.4 Ad-hoc Routing (Network) Layer

In a network, the routing protocol is responsible for the delivery of messages from a source node to the message destination across multiple hops as quickly as possible and without failures. An ad-hoc network dictates that the nodes will not know the topology and layout of the network before being introduced into the network. Therefore, when a node is switched on for the first time, it will attempt to determine who its neighbours are and how it is situated within the network.

This section will explain the popular protocols used for ad-hoc networking and highlight the suitability of each for use in our WSN. A routing protocol will then be chosen and described in detail, including the routing packet, routing table and route discovery mechanism.

6.4.1 Choosing a Routing Protocol

There are three main types of ad-hoc networking protocols, each with its individual advantages and disadvantages. The three protocol types are table driven (proactive) protocols, on-demand (reactive) protocols, and hybrid protocols which combine aspects of proactive and reactive together.

This section will describe each type of protocol and provide an example of each. Finally a protocol will be chosen based on its suitability for our network.

6.4.1.1 Table-Driven (Proactive) Protocols

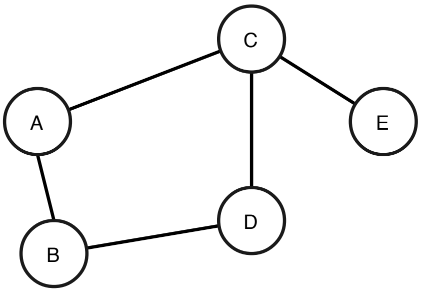
In a proactive protocol, each node maintains a comprehensive list of all network destinations as well as the routes to those destinations. This list is known as a routing table, and is duplicated throughout the network. The routing tables are updated periodically, by distributing the routes to all nodes throughout the network.

Advantages of this type of protocol are that they provide a low delay to finding a route to any destination, and the routes are immediately available to all nodes. A key disadvantage is that they have a low scalability because the control overhead is proportional to the number of nodes in the network. These protocols also require a larger storage capacity than some others due to the comprehensive routing tables that must be stored in memory. Furthermore, the network reaction time can fall sharply when there is link loss or a node fails to reply.

An example of a proactive routing protocol is the Destination-Sequenced Distance-Vector (DSDV) table driven protocol.

6.4.1.2 Example: Destination-Sequenced Distance-Vector (DSDV)

In the DSDV protocol, each node maintains a routing table which contains the shortest path to a destination as well as the first node on this shortest path [64]. An example of such a table is given in Figure 6.14. The routing table also contains a sequence number for each route, which is even if the route is working and odd if the link is broken. When a node receives a route update and the sequence number of the route update matches that of the current route in its table it simply discards the route with the most hops.

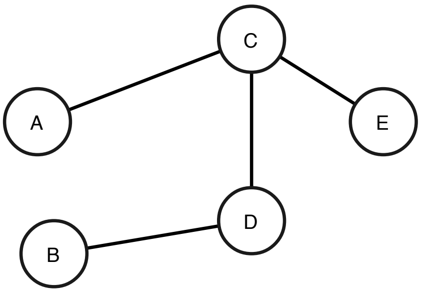


Dest	Next	Hops
C	C	0
E	C	1
D	B	1
B	B	0

Figure 6.14: An example of a routing table in the DSDV protocol for node A of the shown network.

Nodes exchange updated portions of the routing tables at regular intervals. Full updates are exchanged when significant changes are observed in the topology of the network.

When a link break is detected (such as between node A and node B), node A will set the number of hops for all routes through the broken link to infinity and broadcast its table through the network. Node C detects that the link between node A and node B is broken and that it has a connection to node B, and will therefore inform all its neighbours of the shortest distance to node B. This information is propagated through the network. Node A may now send its data to node B via the new route A-C-D-B instead of the old route A-B. An example of the updated routing table is shown in Figure 6.15.



Dest	Next	Hops
C	C	0
E	C	1
D	C	1
B	C	2

Figure 6.15: When the link between node A and node B breaks, the routing table for node A is updated as shown.

For a single link break multiple tables must be transferred through the network, first describing the link break and then proposing updated routes to circumvent the break.

This is a very bandwidth intensive process and not suitable to our network which needs to keep as much bandwidth open for communication from an on-animal tag as possible.

6.4.1.3 On-Demand (Reactive) Protocols

In an on-demand protocol, each node has the ability to discover routes on-demand. This is achieved by flooding the network with a route request. When a node receives a request and is also the destination in question, it will reply with a route reply which is addressed (unicast) to the requesting node. In the process, all the nodes along the unicast route learn about the position of the destination node as well as the requesting node. Each node maintains a routing table which contains destinations with which it has communicated before. If it becomes necessary to communicate with a node which is not listed in this routing table, a route discovery is performed for that node.

An advantage of a reactive routing protocol is that it requires less bandwidth in the network than its proactive counterpart, because routes are only found on demand. This frees up bandwidth for data messages or, in our case, for messages from on-animal tags. Reactive protocols also require far less storage than a proactive protocol, since only routes to requested nodes are stored in memory, and not the route to each possible node. However, because routes must be discovered on demand, there may be a delay between receiving a message and transferring it to its destination. Therefore, if a reactive protocol were to be used in the WSN, the delay to finding a route to a server would need to be minimized.

An example of a reactive protocol is the Ad-hoc On-demand Distance Vector (AODV) routing protocol.

6.4.1.4 Example: Ad-hoc On-demand Distance Vector (AODV)

For this protocol, nodes store only the next hop information for a route in their routing table. If a node does not have routing information for a particular destination, it will send out a route request. This request will be multicast (flooded) through the network in a controlled way [65].

If a node receives a route request, it can either forward the request or, if it is the intended destination, respond with a route reply. If a route request is received multiple times, the repeats will be discarded limiting the possibility of a loop.

When a node responds with a route reply, this is unicast back to the source. This means that the route reply is addressed specifically to the requesting node and will follow the same route it followed to arrive at the destination node. This is possible because the route request accumulates the addresses of each node which forwarded the request. An example of a route request multicast through the network to find a route from A to J and the subsequent route reply which is unicast back to the originating node, is shown in Figure 6.16.

If a link break is detected along a route, the node which detected the break will send an unsolicited route reply to the source with the number of hops to the (now broken) destination set to infinity. The nodes along the path that receive this message will delete the route from their table and the source node will initiate a new route request to determine an alternative route to its destination. This approach to routing causes control

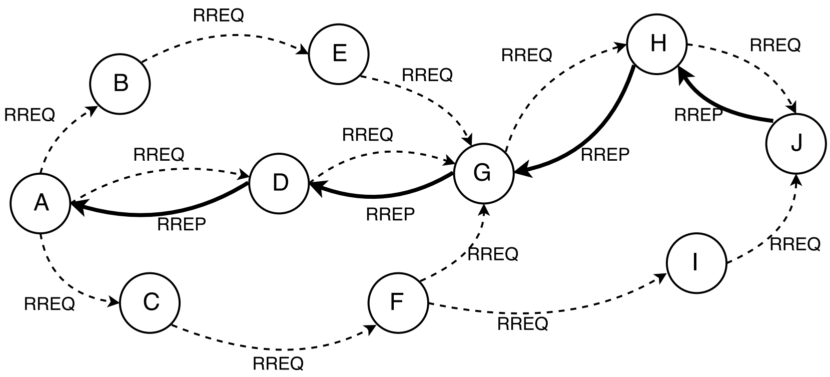


Figure 6.16: An illustration of AODV route finding. Node A broadcasts a Route Request (RREQ) to find a route to node J. When the RREQ reaches node J, it responds with a unicast Route Reply (RREP) to node A. The final route for a message to be sent from node A to node J will be A-D-G-H-J.

traffic on the network to be dynamic, and as long as links remain valid between nodes, AODV remains passive.

6.4.1.5 Hybrid Protocols

For networks where neither a reactive nor a proactive protocol is suitable, a hybrid protocol which combines the characteristics of both can be more appropriate. Hybrid protocols have an advantage over reactive protocols, because they can establish a link quickly due to the proactive routing tables stored within the network. However, the storage and processing time required exceeds that of a pure reactive protocol.

An example of a hybrid protocol is the Zone Routing Protocol (ZRP).

6.4.1.6 Example: Zone Routing Protocol (ZRP)

The Zone Routing Protocol uses a combination of proactive and reactive routing procedures to send data over a network [66]. A proactive routing protocol, known as the intra-zone routing protocol, is used if the destination of the packet is within the zone of the sender. A node is considered to be within the zone if it is within r hops of the origin. If the destination is within the zone, a route is determined using the proactive routing table, and the packet is sent immediately. If the destination is not within the intra-zone, a reactive protocol is used to determine the route.

The reactive protocol, known as the inter-zone routing protocol, checks each individual zone outside the intra-zone to determine if it includes the destination. Once the matching zone is found, the proactive protocol is used to route the packet to the destination. An example with $r = 1$ is shown in Figure 6.17, with the intra-zones for nodes 9 and 10 highlighted.

If node 10 would like to communicate with node 4, it will use the reactive inter-zone routing protocol to determine which zone node 4 is in. In this case, node 4 is in the intra-zone of node 9, therefore the message can be routed from node 10 to node 9 which will then use the proactive intra-zone routing protocol to route the message to node 4.

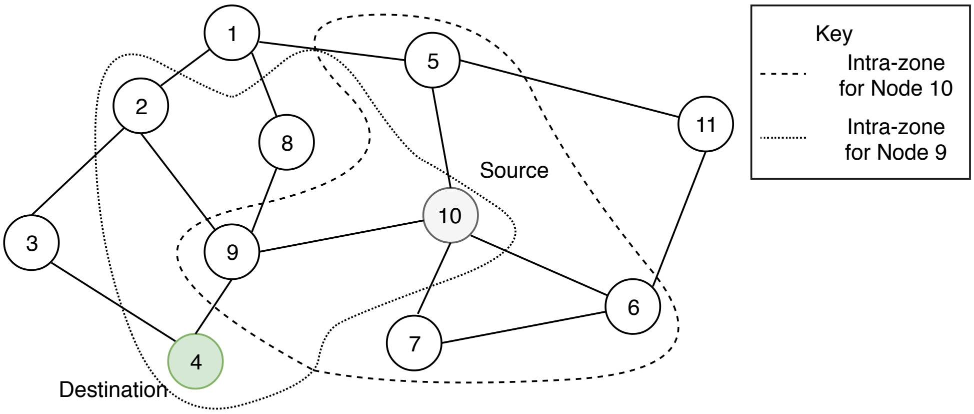


Figure 6.17: An example of a ZRP network. The intra-zones for node 9 and node 10 are shown.

ZRP reduces the control overhead of routing a packet over multiple-hops by using the reactive protocol to determine which zone the destination node is in, rather than needing to determine the complete route to the node. It also reduces the time needed to route packets within its own zone by using table-based proactive routing. This protocol is very useful in situations where nodes need to send most messages to nodes in their immediate vicinity. In our WSN however, nodes need only to know the routes to the server nodes, therefore the proactive intra-zone routing table is not required. Furthermore, it is desirable for a node to maintain the path to a server node, rather than needing to re-discover it repeatedly. This protocol is therefore not suited to the needs of our WSN.

6.4.1.7 Chosen Routing Protocol

The routing protocol chosen for our WSN is a reactive protocol based on the AODV protocol. A reactive routing protocol was chosen as it requires a low control overhead, freeing up bandwidth for the on-animal communications to be routed through the network.

It is also not necessary for each node to know about every other node in the network, because the mesh nodes will pass data to a server node and not be destinations themselves. The MSP432 MCU has only 64 kB of flash memory, which restricts the size of a routing table, therefore, the low storage requirements of the reactive protocol make it a favourable choice for our WSN network.

The chosen reactive routing protocol will discover a route to a server node on start-up, and become dormant after that.

6.4.2 Implemented Routing Protocol

A reactive protocol based on the AODV protocol was chosen for our WSN. This section will provide an overview of how the protocol was implemented, including the control variables required, the routing table and how a route discovery is performed. LoRa is not natively a mesh network, and the ability of the nodes to for a mesh network was adapted from the open source RadioHead library.

6.4.2.1 Overview

The implemented AODV routing protocol is a multi-hop protocol which passes a message from node to node until the message reaches its final destination. Each node does not know the entire route which the message will traverse to arrive at the final destination. However, it knows the address of the next node to which the message should be sent on its path to the destination. If a node does not know the route to a destination, it can initiate a route discovery by sending a broadcast message requesting a route to the destination. Any node which knows a route to the destination can unicast a reply to the requesting node. Once the requesting node has discovered a route to the destination, the next hop along this route is stored in its routing table for future use.

If a node can no longer make contact with the next hop on a route it will mark that route as invalid and attempt to find a new route to the destination. In this way the routing protocol is only active when a node does not have a route to a destination in its routing table. Figure 6.18 shows an example of a network of mesh and server nodes. The on-animal tags are able to move freely between WSN nodes, and communicate with any node. The arrows on the links show the direction in which communication is possible. With the current set up, WSN nodes are able to communicate with each other, but on-animal tags can only send data to a WSN node and not receive data from them. On-animal tags can also not communicate with each other.

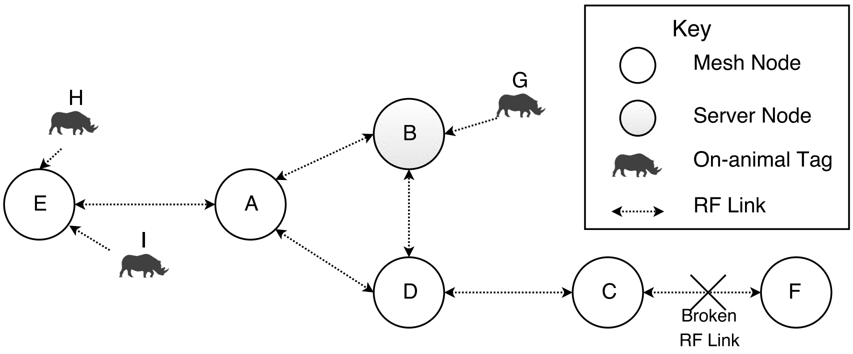


Figure 6.18: An example of a network of the 3 different type of nodes.

6.4.2.2 Routing Packet

The routing protocol requires that specific variables are sent during each transmission. These variables allow the protocol to decide how to react to a message, and are arranged in a routing packet as shown in Figure 6.19. The routing packet makes up part of the payload in the DATA packet sent by the MAC protocol.

Dest	Source	Hops	ID_R	Flags_R	Message Type

Figure 6.19: The six variables which make up the routing packet. All are 8-bit unsigned integers.

Dest: the address of the node which is the final destination of the communication. The destination can be any WSN node, but never an on-animal tag.

Source: the address of the node from which the message originated. This will always be a WSN node, as an on-animal tag does not have the ability to route messages through the network. The on-animal tag can only pass its data to a WSN node, which will then find the nearest server in its routing table and send the data received from the tag to the server.

Hops: the number of nodes the routed message has already passed. After a message has been received by a node and is sent on to another node, the number of hops is incremented by 1. A message will continue through the network until the number of hops reaches a user defined threshold of 30. After this it will be discarded, preventing a message from being routed through the network infinitely.

ID_R: the identification number of the routed packet. This ID is assigned to the routing packet by the source node. After each packet is sent, the ID is incremented by 1. Each WSN node maintains a list of the most recent ID from all source nodes which have routed a packet through it. The ID of the received packet is checked against the ID of the most recent packet from that source node to determine if the packet has been received already. If it has, the received packet is ignored.

Flags_R: routing specific flags. These flags currently do not affect the routing of a message, they are simply saved at the destination. They can in future be used to indicate the priority of a message.

Message Type: used by the routing protocol to identify the type of message which is been routed and how to react to it. A table listing the types of messages that can be specified is found in Table 6.4.

6.4.2.3 Message Type

When any message is routed in the network, it is imperative to know the actions which are required by that message. This allows the nodes to react in accordance with the received message. There are currently 8 message types which can be specified by a sending node as listed in Table 6.4. Each message type is assigned an 8 bit integer and forms part of the routing packet.

Table 6.4: Message types used by the routing protocol, and the integer value assigned to each.

Message Type	Integer Value
Application Message	0
Discovery Request	1
Discovery Response	2
Route Failure	3
Server Discovery Request	4
Server Discovery Response	5
Battery Low	6
Battery High	7

Application Message: when the message contains only data routed from one node to another, it is an application message. This includes messages which are received by a mesh node and then routed to a server node. The routing protocol does not react in any specific manner to this message type, but will route the message to its desired destination.

Discovery Request: is used by a node when a route to a specified node needs to be discovered. The node will send a broadcast message requesting a route to the specific node. If any node in the vicinity hears this request, it will pass it on until a route is found. A route is found either when the route request reaches the destination node, or when a node already has a route to the specified node in its routing table.

Discover Response: once a route to the destination has been found the node will send a Discovery Response message. This message will be unicast back to the node from which the Discovery Request originated.

Route Failure: when a node is not able to pass 3 consecutive messages to the next hop on a route, that route is deemed to have failed. In this case, a Route Failure message is unicast to the source node of the most recent message to notify them of the route failure. When the nodes along the unicast route receive this message, they mark the current route as invalid.

Server Discovery Request: while nodes can specify the address of a node they would like to discover, it is also possible to request a route to the nearest server. In this case the node will broadcast a Server Discovery Request message.

Server Discovery Response: when a node receives a Server Discovery Request it will first check whether it is a server, or whether it has a server in its routing table. If either case is true it will unicast a Server Discovery Response to the source node, updating it to the location of a server. If the node does not know the location of a server, it will broadcast the Server Discovery Request to its neighbours.

Battery Low: if a node detects that its battery is low and it will no longer be able to send and receive messages, it broadcasts a Battery Low message to the surrounding nodes. The nodes which receive this message will then mark all routes which pass through that node as invalid.

Battery High: once the battery of the affected node is recharged or replaced, it will broadcast a Battery High message. The nodes which receive this message will then mark all routes passing through that node as valid once again.

6.4.2.4 Routing Table

When a node would like to send a message to a particular destination, it must know both the destination and the route. With the chosen routing protocol, the node does not know the entire route to the destination, but only the next node along the route. A routing table stores known servers to which a message can be routed and the next node along the route to that server. The routing table also contains information about the route, such as the number of hops to the destination, whether the route is currently active, and whether the destination node is a server or not. The variables stored in the routing table are given in Table 6.5.

Table 6.5: Routing table variables and short descriptions.

Variable	Description
Dest	The address of the final destination of a message.
Next hop	The next node along the route to the destination.
State	State of the route, either valid or invalid.
Hops	Number of nodes between this node and the destination.
Server	Is the final destination a server or not.
Next hop failures	Number of messages which failed to be sent to the next hop.

Dest: the intended recipient of the message. As the message is passed along the network each node that receives the message will determine the next hop to the final destination from its routing table and then transmit the message to that node.

Next hop: identifies the next node to which the message should be transmitted on the path to the destination.

State: a marker indicating whether a route is currently valid or not, stored as a boolean type. If the route is marked as invalid then no messages will be sent along that route. A route will be marked as invalid if the number of Next hop failures reaches 3. It can also be marked as invalid if a node receives a Battery Low message from a neighbouring node, or if the node receives a route failure message.

Hops: the number of nodes on the route from the current node to the destination node. If the next hop is the destination node then the number of hops will be 0, but if there are one or more nodes between the current node and the destination, the number of hops will be the corresponding positive integer. The routing table is sorted by the number of hops in ascending order.

Server: indicates whether the destination node is a server or not. Stored as a boolean type, if the destination is a server then the variable will reflect as 'true' in the table.

Next hop failures: the number messages which have failed to be sent to the next hop. If the number of Next hop failures exceeds two, then the state of that route will be declared invalid and a new route must be found to the destination. The Next hop failures variable is zero initially. Each time a message fails to be sent to the next hop it is incremented by 1. If a message is successfully sent to the next hop, the number will be reduced by 1, reaching a minimum of 0.

For our network, it is only necessary for each node to know the route to server nodes. Therefore the routing table is limited to a user defined value of 10 entries. This number can easily be increased as required. An example of a routing table for node A in Figure 6.18 is given in Table 6.6. For this example, the routing table contains routes to each WSN node in the figure, and not just to the server node.

Destination B is a server and therefore its 'Server' variable in the routing table is set to True. Because it is a server node, it is placed at the beginning of the routing table. This is done to shorten the amount of time spent searching the routing table for the nearest server node. Node B, E, and D are all neighbouring nodes of node A and are therefore 0

Table 6.6: An example of a routing table for node A as shown in Figure 6.18.

Dest	Next hop	State	Hops	Server	Route Failures
B	B	True	0	True	0
E	E	True	0	False	0
D	D	True	0	False	0
C	D	True	1	False	0
F	D	False	2	False	3

hops away. Node C is 1 hop away from node A, and hence a message from node A will need to be routed through node D. Node F is 2 hops away and hence a message from node A will need to be routed through nodes D and C to reach node F. Node A does not know the entire route to node F, only that a message destined for node F must be sent to node D, which will route it to node F. In the example routing table, the link between node C and node F is currently broken, therefore the state of the route to F is marked as false (invalid).

6.4.2.5 Route Discovery

When a node would like to send a message to a destination but it does not yet know the route to that node, it can initiate a route discovery. If the node would like to find a specific destination it can use a Discovery Request message type. However, if the node does not know the destination but would only like to find a server, it can use a Server Discovery Request message type. The requests are broadcast to all neighbouring nodes in the network and if any of those nodes already have the destination (or a server) stored in their routing table they will respond with the corresponding message type, either Discovery Response or Server Discovery Response.

Once the requesting node receives the response it will store that route in its routing table. If the table is full, it will replace the route in its table with the most hops. All routes to destinations that are not servers will be replaced first, and only after that will routes to servers be replaced.

Recall the WSN example network in Figure 6.18, in which the route to node F was broken and hence node A requests a new route to node F. If node F comes back online and is ready to respond, the interactions between the nodes are shown in Figure 6.20, with the arrows indicating the direction of communication. All previous routes to node F were marked as invalid when the link between nodes C and F was broken. Therefore, the request will have to reach node F before a route to it can be re-established.

In the figure, node A broadcasts a Route Request packet to its neighbouring nodes E, B and D. Each of these nodes check their routing tables, discover they have no valid route to node F, and consequently broadcast the route request on to their neighbouring nodes. It is possible that nodes B and D might attempt to broadcast the route request to each other. However, once either node has received the route request once, they will ignore further route requests with the same routing packet ID. Once the route request reaches node F, it will unicast a route reply addressed to node A through the route which arrived first. In this example the route request arrived first from nodes C and D.

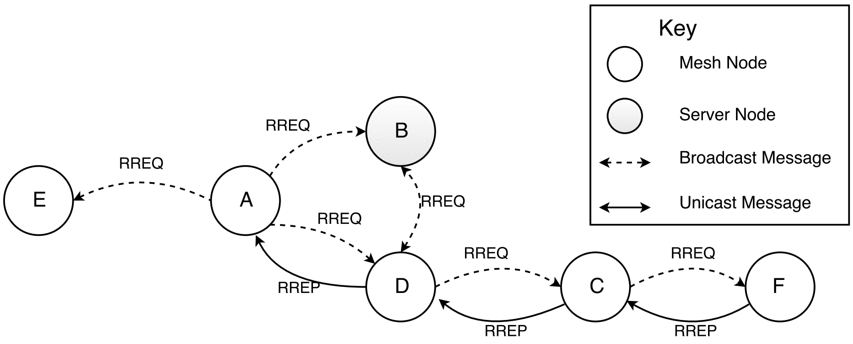


Figure 6.20: The communication for node A to discover a route to node F if no node has node F in its routing table.

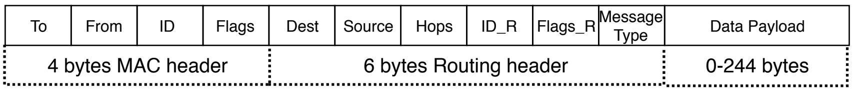
Once the requesting node (node A) has broadcast its route discovery request it will remain in receiving mode for a period of 30 seconds. This period can be configured by the user at a later stage as required. During this time the node will only receive messages addressed directly for it, and will not respond to a RTS which has been broadcast. This is done to ensure that the node does not broadcast while awaiting a route reply.

6.5 Network Data Characteristics

In this section the timing and payload of the protocols will be explained. The data payload allowed per a message will be described, the timing diagram of the full message exchange between nodes will be given and the number of messages sent per a minute will be considered.

6.5.1 Data Payload Per a Message

The RFM96W RF module is capable of sending a message of up to 254 bytes in length. A DATA packet is made up of 4 bytes for the MAC protocol header, 6 bytes for the routing protocol header, and consequently up to 244 bytes of data payload. Figure 6.21 shows the full DATA packet and how many bytes are required by each protocol.



To	From	ID	Flags	Dest	Source	Hops	ID_R	Flags_R	Message Type	Data Payload
4 bytes MAC header				6 bytes Routing header						0-244 bytes

Figure 6.21: The DATA packet showing the 4-byte MAC header, 6-byte routing header and the data payload which can be up to 244 bytes in length.

6.5.2 Timing Diagram

This section will describe the time taken for a message to be sent between 2 nodes. The number of retries per a packet, the time taken by the RF module to send each packet, as well as the time which the module will wait for a reply are all shown in Figure 6.22.

Node A begins with a CS to determine if the medium is busy. If the medium is busy, then the node enters a waiting state which lasts between 2 seconds to 4 seconds (randomly chosen), checking to see if the medium is free every 100 ms to 400 ms (randomly chosen).

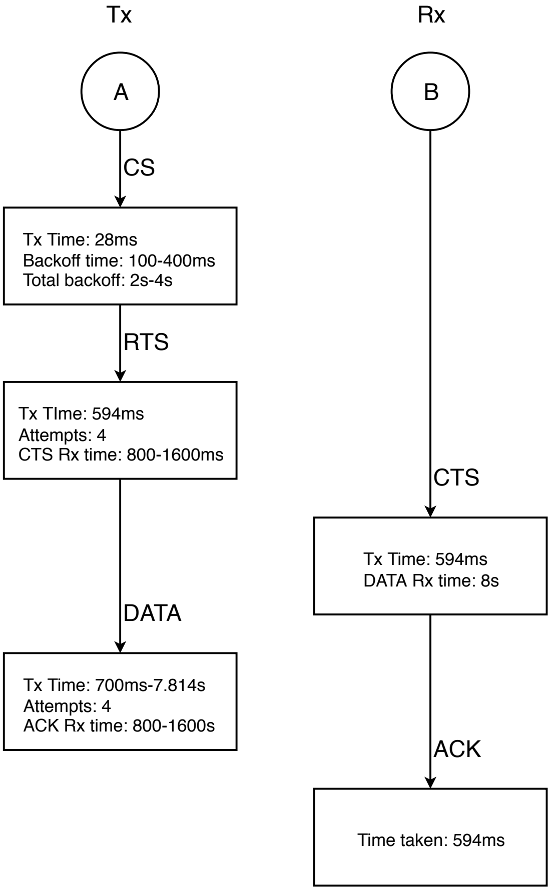


Figure 6.22: Detailed timing information for each action of the MAC protocol. In this example, node A is transmitting a message to node B.

A CS takes approximately 28 ms to complete and if the wait period expires and the medium is still not free then the message sending fails. Once the CS is completed and the medium is judged to be free, node A sends a RTS packet to node B. This packet takes approximately 594 ms to be sent. Node A then enters a receiving state for a random time period lasting between 800 ms and 1600 ms, while waiting to receive the CTS packet from Node B. If the waiting period on Node A expires without a CTS been received from node B, it is assumed that node B did not receive the RTS and therefore the RTS packet is retransmitted. The RTS will be transmitted a total of 4 times before message sending fails.

Upon successfully receiving a RTS, node B responds with a CTS packet sent to node A. The packet takes a total time of 594 ms to be sent and is only sent once. After sending the CTS packet, node B enters a receive state for 8 seconds to listen for the DATA packet from node A.

When node A has received the CTS packet it will send the DATA packet to node B. If the message is from a WSN node, the payload will include the routing protocol header. Transmission of the DATA packet will take between 700 ms and 7.814 s depending on the length of the payload. It is possible to transmit a payload of 244 bytes in one DATA packet, however the longer the payload the longer the time it takes for the RF module to transmit the packet. Once the DATA packet has been sent, node A enters a receive state for between 800 ms and 1600 ms (randomly chosen), during which it waits for node B to acknowledge receipt of the DATA packet. If node A does not receive the acknowledgement within the waiting time, it will retransmit the DATA packet. The DATA packet will be sent a total of 4 times before the communication is abandoned.

Once node B has successfully received the DATA packet from node A it replies with an acknowledgement. The ACK takes 594 ms to transmit.

The total time taken to transmit a message with a payload of zero bytes between two nodes, if there are no wait times, and no retransmissions, is known as the handshaking overhead. For our network, if the minimum times are used, this time is 2922 ms.

6.5.3 Message Throughput

Knowing that the minimum handshaking overhead is 2922 ms, it is possible to determine the time that it it would take to send a payload of a certain length. It takes approximately 29.16 ms to send a single byte. Equation 6.5.1 calculates the total message time, as a function of the number of bytes (N_{bytes}) in the payload.

$$T_{msg} = 2922 + 29.16 \times N_{bytes} \tag{6.5.1}$$

The time required to send a message between nodes is therefore, the number of bytes in the data payload (N_{bytes}) multiplied by the time which it takes to send a single byte plus the handshaking overhead of 2922 ms. The number of messages which can be sent per minute is therefore given in Equation 6.5.2.

$$\begin{aligned} Messages\ per\ minute &= \frac{1\ minute}{T_{msg}} \\ &= \frac{60000\text{ms}}{2922 + 29.16 \times N_{bytes}} \end{aligned} \tag{6.5.2}$$

Figure 6.23 shows that, as the number of bytes in the payload increases, the number of messages which can be sent through the network decreases exponentially. Therefore, before a message is sent through the network, the trade off between the power required to send a message, and the length of the message which is sent, should be considered.

6.6 Summary and Conclusion

This section considered the requirements of the ad-hoc WSN network, and the classification of nodes. The network was divided into several layers which stack on top of one another. The network protocols must be able to route messages across multiple WSN nodes and receive messages from WSN nodes and from on-animal tags.

The physical layer consisting of the RF module, was discussed first. Its use of spread spectrum modulation and forward error correction were discussed, as well as the basic components used by the module to send and receive messages.

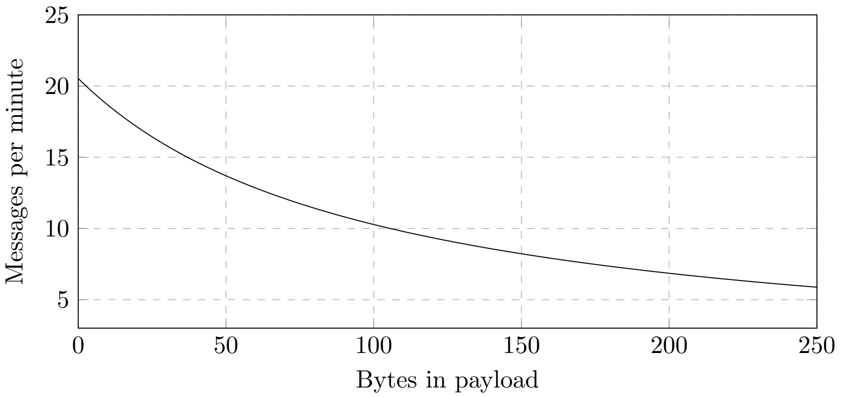


Figure 6.23: Messages which can be sent per minute in the network, as a function of the length of the data payload in bytes.

The MAC layer was considered next. Several MAC protocols were investigated, and a MAC protocol which suited the requirements of the WSN was chosen. This MAC protocol is based on the MACAW protocol and uses only 4 control variables per a packet. When sending a message, the protocol uses a RTS-CTS DATA-ACK handshake, to ensure that message throughput is reliable and consistent.

Several optimisations were made to the MAC protocol, including delay times for the WSN nodes before responding to a broadcast message, and address saving on the on-animal tags. Several examples were used to show how the MAC protocol was implemented for different situations.

Next, the routing layer was considered, and several protocols which have been used for multi-hop ad-hoc networking on previous projects were discussed. The chosen routing protocol is based on the AODV reactive routing protocol. This protocol remains dormant until it is needed to find a new node. During normal routing, the protocol requires only 6 control variables per a message. The routing table was discussed and an example of route finding with the protocol was given.

Finally, the overall network characteristics were considered. A node is capable of sending a message with a data payload of between 0 and 244 bytes with a handshaking overhead of 2922 ms. An equation was given for calculating the time required to send a message as a function of the length of the data payload.

In the following chapter, software written to support the functioning of the WSN nodes is described.

Chapter 7

Software Design

This chapter will discuss additional software that was written for the WSN nodes. This software is in addition to the C code implemented on the MSP432 to support the functioning of each hardware module. The chapter does not include the software implemented on the FIT IOT platform at Inria.

The software described here supports the use of non-volatile flash memory on the MSP432 in order to store variables, the use of UART commands to communicate with the MSP432 and a GUI to simplify the issuing of the UART commands. This chapter will also discuss a few key program routines of each node.

7.1 Flash Memory

Non-volatile flash memory on the MSP432 is used to store variables which monitor statistics during usage of the WSN nodes. Each such variable is assigned to a specific address in non-volatile memory and updated as required. In order to limit the number of writes to the flash memory, a complete copy is maintained in an array in volatile memory on the MSP. Variables are first updated to this array, while updates to the flash memory are performed at less frequent intervals. The flash memory is copied to the local array at start-up. In total the MSP432 has 256 kB of non-volatile flash memory and 64 kB of volatile memory.

The location of the stored variables in the local array as well as the flash memory are given in Table 7.1. The name of each variable as well as the variable type is also given. The non-volatile flash memory is made up of 8 bit integer slots. Therefore, if a 32 bit number is to be stored in the flash memory it must stored as 4 bytes. When it is read from memory, the bytes are combined to produce the 32 bit integer. This is all implemented by this project, and the memory storage uses Big-endian to store all memory.

The following three sections will explain how the variables are divided into sections and what each variable is used to store.

Table 7.1: List of addresses which are assigned to variables in non-volatile flash memory and the volatile array. A copy of these variables is maintained in an array in volatile memory on the MSP432 to allow for quick access and to lower the number of writes to flash memory.

Array Address	Flash Address	Name	Type
0	0x0003F000	Node Address	uint8_t
1	0x0003F001	Node Configuration	uint8_t
2	0x0003F002	Node Restarts	uint32_t
6	0x0003F006	Memory Updates	uint32_t
10	0x0003F00A	SD Card Log	uint32_t
14	0x0003F00E	Shutdown at 3.7 V	uint32_t
18	0x0003F012	Shutdown at 3.55 V	uint32_t
22	0x0003F016	Messages Received	uint32_t
26	0x0003F01A	Messages Received From Tag	uint32_t
30	0x0003F01E	Messages Passed On	uint32_t
34	0x0003F022	Re Acknowledgements	uint32_t
38	0x0003F026	Re Route Required	uint32_t
42	0x0003F02A	Re Route Received	uint32_t
46	0x0003F02E	Messages Sent	uint32_t
50	0x0003F032	Message Retries	uint32_t
54	0x0003F036	Failure to Send	uint32_t
58	0x0003F03A	Attention	uint32_t
62	0x0003F03E	Timeouts	uint32_t
66	0x0003F042	Routing Table 1st Entry	6 × uint8_t
72	0x0003F048	Routing Table 2nd Entry	6 × uint8_t
...	...	...	...
120	0x0003F079	Routing Table 10th Entry	6 × uint8_t

7.1.1 Node Identification

The node address and the server status are 2 variables which are required for a WSN Node to function within a network. These variables are stored as 8 bit unsigned integers (uint8_t) in flash memory and are stored at the start of the memory at address 0x0003F000 and 0x0003F001.

Node Address: The address of the node within the WSN network. Each node has a unique address and it is used by the MAC and routing protocol to send and receive messages.

Node Configuration: Stores the node configuration. The node can be configured as either a mesh node or server node. Because this is a boolean requirement, the variable is stored as 0x01 if the node is a server, and as 0x00 if the node is not a server.

7.1.2 Node Statistics

These variables track the usage of features of the WSN-Node. They are stored in memory from address 0x0003F002 to 0x0003F015.

Node Restarts: Tracks the number of times the start up procedure is run. This can occur after an error which causes the node to restart or if the node loses power and then regains it again at a later time.

Memory Updates: Tracks the number of times the flash memory is updated. The memory is updated at set intervals, when the user makes a direct change to memory or when the routing table is updated.

SD Card Log: Records the log number of the SD card. The log number is incremented after each restart as well as after every new day.

Shutdown at 3.7 V**:** Records the number of times the battery voltage falls below 3.7 V and the peripheral devices (SD Card, GPS, and temperature sensor) were disconnected from the power supply.

Shutdown at 3.55 V**:** Records the number of times the battery voltage fell below 3.55 V. At this voltage, the node sends a message to all the nodes in the vicinity alerting them that the node will be powering off, and then enters sleep mode until the battery voltage rises above 3.7 V again.

7.1.3 Network Statistics

The following variables are used to track statistics related to the networking protocol running on the node. This includes information concerning the number of messages received by the node, and other network events. These statistics are all stored from address 0x0003F016 to 0x0003F041.

Messages Received: Tracks the number of messages which are addressed directly to the node. This does not include messages that are received by a node as part of a route and then passed on to the next hop.

Messages Received From Tag: Tracks the number of messages that are sent from an on-animal tag to the WSN node.

Messages Passed on: This is the number of messages that are received by a node that is not the end destination of that message and are therefore forwarded to the next hop on the route.

Re-acknowledgements: Tracks the number of times that an acknowledgement must be re-sent because the sending node didn't receive it, and retransmits the packet.

Route Discovery Required: Records the number of times a route to a destination cannot be found and a route discovery message is sent to find a new route.

Route Discovery Received: Tracks the number of times a route discovery message is received with a route to the requested node.

Messages Sent: Records the total number of messages that are sent by a node.

Message Retries: Records the total number of times the RTS or DATA packet must be resent because an CTS or ACK was not received.

Failure to Send: Records the number of messages which could not be sent to a node. This includes messages for which there was no response after the RTS was sent, as well as messages for which the ACK was received for the RTS but never for the DATA packet.

Attention: Records the number of times an RTS is received by a node and the node enters a waiting state for the data message. This can be from a tag or another WSN node.

Timeouts: Records the number of times the waiting state entered after receiving an RTS expires.

7.1.4 Routing Table

The entire routing table is stored in flash memory. Each entry in the routing table contains 6 variables, of which 4 are 8 bit integers and 2 are boolean. All the variables are stored in memory using 8 bit integers. The boolean values are converted to integers by storing them as 0x01 when the boolean is true and 0x00 when it is false.

Each entry in the routing table therefore requires 6 bytes. The complete routing table of 10 routes requires 60 bytes to store in memory. The routing table is stored from address 0x0003F042 to 0x0003F080.

7.2 Program Routine

The following section discusses the program routines of the Node. The routines which are examined in this section are the start up routine as well as the continuous routine. Furthermore, the check for a server algorithm is also explained. The flow diagrams for each of these routines can be seen in Figure 7.1.

7.2.1 Node Start Up

When the node is switched on or restarted, it first sets the registers, enables the selected ports and checks the peripheral hardware before arriving at the main loop. The setting of the registers and ports is known as the node start up routine, and the flow diagram for this routine is shown in Figure 7.1 (a).

The node begins by initialising the ports and functions specific to the MSP432 MCU, including setting the clock speed to 12 MHz, connecting all interrupt triggers to their corresponding functions and enabling the ports which are required for the MCU to communicate with the connected hardware devices.

The node then checks which devices are connected and whether they are functioning. If they are not, the user is alerted via a message to the UART. The time of the RTC on the MCU is compared to the time from the GPS. If they differ, then the MCU RTC is updated to the GPS time.

After this, the variables stored in non-volatile flash memory are copied to an array in volatile memory. The radio is then set up according to the parameters given in subsection 5.3.1. During this stage, the node address is copied from the flash memory, as well

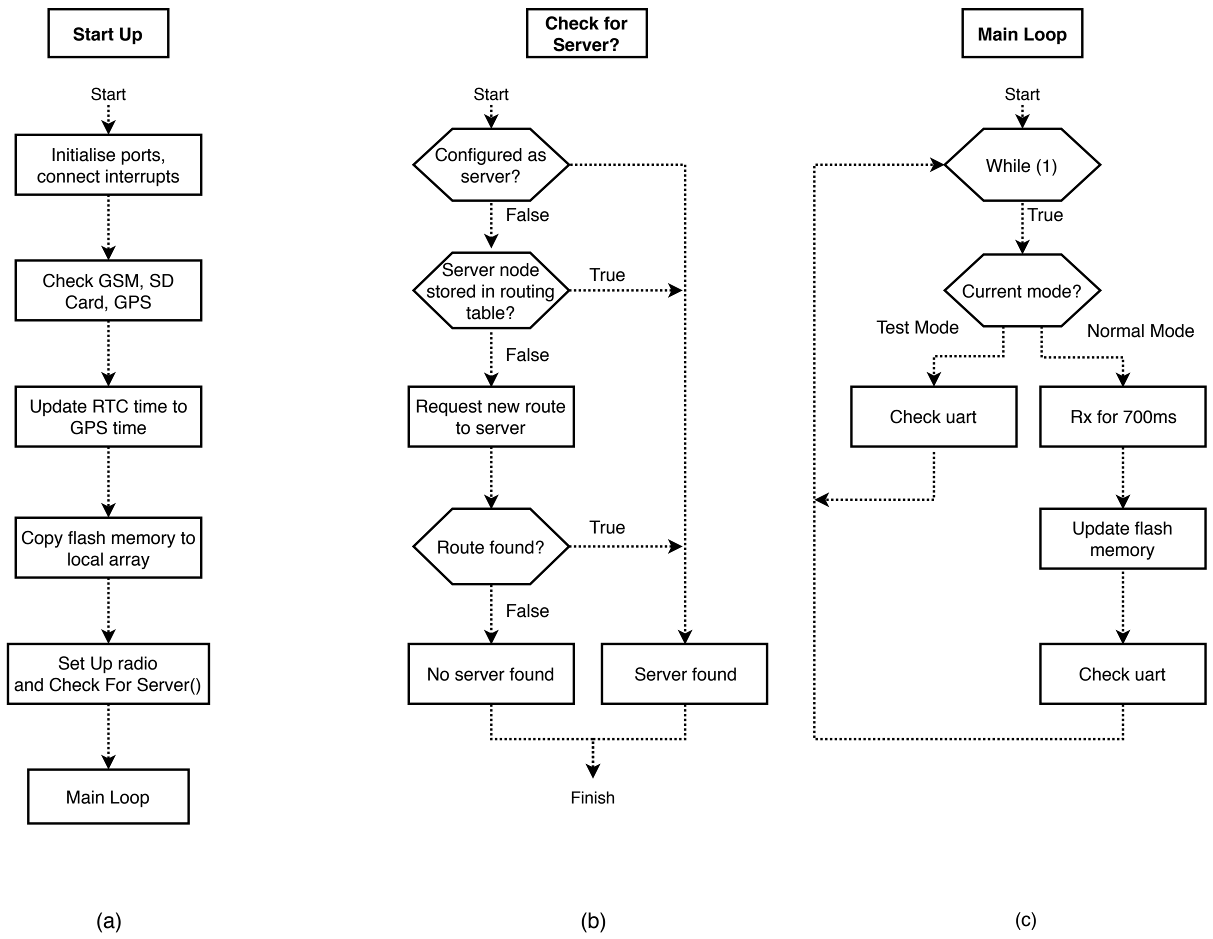


Figure 7.1: Flow diagrams for routines of the WSN node. (a) The flow diagram for the node on start up. (b) The flow diagram for the node during a check for server is shown, and (c) the flow diagram for the node in its continuous main loop.

as the node configuration. Once the RF module is fully functional, a check for server is done. Once the check for server is complete, the node start up routine is complete and the program continues to the main loop.

7.2.2 Check For Server

A check for a server is a routine which determines where the nearest server node is from the current node. A check for server is always done during node start up, but it can also be done by the user at any time via the UART commands. The flow diagram for this function is shown in Figure 7.1 (b).

The node first checks whether it has been configured as a server. If this is true, then the server check is complete, as the node itself is a server. If the node is configured as a mesh node, then it will check whether a route to a server node is stored in its routing table. If it has a route stored to a server, then the server check is complete.

If there is no route to a server node stored in the routing table, it will then send a Server Discovery Request. If a Server Discovery Response is received, server node will be added to the routing table and the server check is complete. If the node does not receive a response before it times out, the server check will fail. In this case there was no server in the vicinity of the node. The reasons for this could be that this node is out of range of any active servers or other nodes near servers. The node should be moved to a location where the server check can find a nearby server node.

7.2.3 Main Loop

The main loop takes place inside a continuous loop. If the mode is set to test mode, then the node only checks whether a new command has been issued via the UART. This will continue until the node is set to normal mode.

In normal mode, the RF module is set to receive mode and the MCU goes to sleep. After 700 ms the MCU checks whether the flash memory should be updated, and whether a command has been issued from the UART to change to test mode. This continues unless there is an interrupt from the RTC or the RF module.

7.3 UART Commands

It is possible for a user to interact with certain variables and functions on a WSN node via UART. The complete list of commands available to the user are given in Table 7.2. All UART commands must be followed by the variables 0x0A 0x0D to be accepted as a valid command by the node.

Table 7.2: List of commands that can be sent via UART to interact with variables and functions on a WSN node. All required arguments are sent as uint8_t data types. All commands should be followed by 0x0A 0x0D.

Command	Description	Required Arguments [uint8]
0x80	Change to Normal Mode	
0xE0	Change to Test Mode	
0xE1	Print all Stats	
0xE2	Print Routing table	
0xE3	Reset Node Stats	
0xE4	Reset Routing table	
0xE5	Print Node Address	
0xE6	Update Address	[address]
0xE7	Print Node Configuration	
0xE8	Set Node Configuration	[configuration]
0xE9	Add Route	[dest addr] [next hop addr] [n_hops] [dest configuration]
0xEA	Reset Network Stats	
0xEB	Print Temperature	
0xEC	Print Time	
0xED	Check for Server	
0xEE	Set Time from GPS	
0xEF	Set Time Manually	[sec] [min] [hour] [day] [month] [year1] [year2]

7.3.1 Test Mode and Normal Mode

A WSN node has 2 operating modes, namely test mode and normal mode. Switching between these is achieved by sending the UART commands 0xE0 and 0x80 respectively. Normal mode is the operating mode which a node is automatically set to when it is restarted, while test mode allows a user to interact directly with the stored variables on the node.

When in normal mode, the node will be able to send and receive messages from the network and the user cannot request information via the UART commands except to change to test mode. If a request to change to test mode is received by the node, the operating mode will only change once the node has completed sending and receiving any messages it may have been processing before the request was received.

In test mode the user can display the routing table, set the network address of the node, or configure the node to a server or mesh node. The node can be changed back to its normal operating mode by sending the normal mode UART command. The node can also enter normal mode by disconnecting and reconnecting power to the MCU, or by resetting the node using the hardware reset switch on the MSP432 breakout board.

7.3.2 UART Command Descriptions

The following section will describe each UART command, as well as the additional information which may be required by the command.

Change to Normal Mode: If in Test mode, the node will change its operating mode to Normal mode.

Change to Test Mode: If in Normal mode, the node will change its operating mode to Test Mode.

Print all Stats: Prints all the statistics that are stored in flash memory of the node. This will display the network and node statistics from address 0x0003F002 to 0x0003F041.

Print Routing table: Prints all 10 entries of the routing table currently stored on the node.

Reset Node Stats: Sets all the node statistics stored in flash memory to zero.

Reset Routing table: Sets every integer in the 10 routing table entries to 0 and every boolean to false.

Print Node Address: Prints the current address of the node.

Update Address: Sets the node address to the number which is sent via UART in the [address] field. For example, to set the node to address 0x01, the following command is sent: 0xE6 0x01.

Print Node Configuration: Prints the current node configuration. If the node is a server, the following message will be printed: ”I'm a server”. If the node is a mesh node, then it will print ”Not a server”.

Set Node Configuration: Sets the node to be a server or a mesh node. To set the node as a server, send 0xF0 in [configuration], and to set it as a mesh node send 0x0F.

Add Route: Add a route to the routing table, with the following specification:

- destination address
- next hop address
- number of hops to the destinations
- node configuration of the destination (0xF0 for server, 0x0F for mesh node)

As an example, the UART command to add a route directly to a server node with an address of 4 is: 0xE9 0x04 0x04 0x00 0xF0.

Reset Network Stats: Sets all the network stats in flash memory to zero.

Print Temperature: Reads and prints the temperature as measured by the internal sensor of the MS5607 on the node.

Print Time: Print the time as stored in the RTC of the MSP432.

Check for Server: Performs check for a server routine described in Section 7.2.

Set Time from GPS: Switches the GPS on, and reads the GPS time. If this GPS time differs from the time given by the RTC on the node, then the RTC time is set to the GPS time.

Set Time manually: Set the time from a manual input. The user must send the seconds, minutes, hours, day of the month, and the month. The year must be sent as two bytes corresponding to its first 2 digits and then the second two digits.

7.4 Python GUI Interface

In order to simplify the interaction with the WSN node using the UART commands, a graphical user interface (GUI) was created. A screenshot of the RhinoNet Node Analyser GUI is shown in Figure 7.2. The GUI was designed using PYQT5, and the functionality implemented with Python. The interface allows a user to click a button to send commands to the MSP432, instead of having to type the command and send it via a terminal emulator. All output from the node will be displayed in the textbox at the bottom right of the GUI. There are also dedicated fields where the user can enter the additional information required for commands.

7.4.1 Implemented Functionality

All the functions described in Section 7.3 were implemented into the GUI. The user is able to switch the node from test mode to normal mode, and back again. When in test mode, the user can use the 'print' buttons on the left of the GUI to print the statistics, the

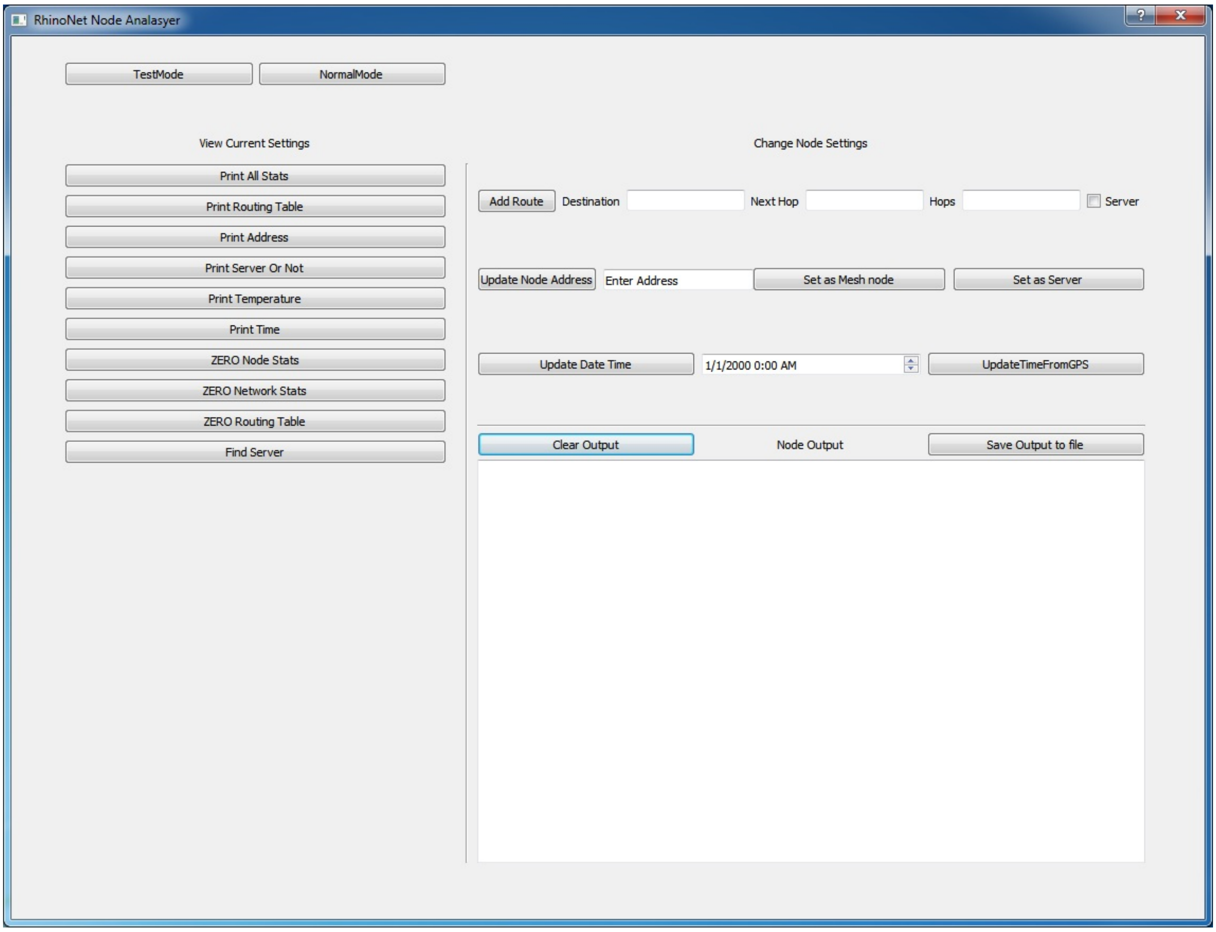


Figure 7.2: A simple GUI was created using Python to easily interact with the UART commands on the WSN node. The GUI allows the user to read all the variables stored on the device, as well as set parameters such as the node address, change its configuration and add routes to the routing table. The GUI is a stand-alone executable (.exe)

current node address, whether it is a server or not and the routing table. Furthermore, the user can set the network stats, node stats and the routing table to zero. Lastly the 'Find Server' button will begin the check for server routine.

The user can also change the node address, set its configuration as a server or not, add a route to a destination node or set the time of the RTC on the node. All output from the node will be displayed in the textbox, and the user can clear the output or save it to a file on the local computer.

7.4.2 Using The GUI With The WSN Node

In order to use the GUI with the WSN node, a few simple steps should be followed and the GUI can only be used to communicate with one node at a time.

To use the GUI, ensure the WSN node is connected to the computer by means of a USB cable. Double click the main.exe and the GUI will start up. On start up, the program will automatically connect to the serial port of the WSN node. The user is then able to interact with the node.

Once the user has completed the interaction, the program can be closed and the UART connection will terminate automatically.

7.5 Summary and Conclusion

This chapter described the additional software written to allow ease of access to the WSN node. Storage of variables in non-volatile memory was also implemented. Each stored variable was described and its address in non-volatile memory given.

Software was implemented on the node allowing the user to switch the node from normal mode to test mode with UART commands. When in test mode the user can interact with all the stored variables on the node, and display or reset them.

The start up and main loop program routines were described, and the check for server function was explained.

Finally, a GUI was designed which allows the user an easier way of interacting with the node. The GUI implements all of the UART commands described in this section, and can be used as a substitute for a serial emulator program.

In the following chapter, results and measurements acquired from practical tests will be described and explained.

Chapter 8

Results

This chapter describes the results from several tests which were performed on the hardware and communication of the WSN. A simulation and test was performed to ensure that the power supply and solar energy harvester works as designed. The communications protocol was programmed to the Future Internet of Things Internet Of Things (FIT IOT-LAB) at the French national institute for computer science and applied mathematics (INRIA), Lille in France and the communication between nodes and from the on-animal tag to the WSN nodes was investigated on that platform. Further investigations with the WSN nodes designed in this project, were performed to further validate the communications protocol. The results from all these tests and conclusions derived from each will be explained in this chapter.

8.1 Power Supply Testing

Simulations and tests were performed in order to ensure that the power supply and solar energy harvester function as designed. First, the voltage and charge of the battery with different solar panels and varying solar radiation conditions was simulated. Next, the power supply with integrated energy harvester, were built and a full test with a load and solar panels was completed.

In the following section, the simulation conditions and results are first described. Following this, the test set-up, solar intensity received during the test as well as the battery voltage throughout the test is described. Finally, the simulation will also be compared with the practical measurements and a conclusion will be given about the effectiveness of the power supply and solar charger.

8.1.1 Simulation

A simulation was set up to predict the behaviour of the energy harvesting system. The charge of the battery was monitored over a 20 day period with a load corresponding to a WSN node using maximum current. The simulation was repeated for 4 different solar panel power ratings.

During the simulation it was assumed that the node is continuously in the extreme mesh power profile, using 208.43 mW. The battery in the simulation has a capacity of 10350 mA h and will be charged to a voltage of 4.1 V. It is assumed for these simulations that when the battery charge reaches 30%, the node will be placed into deep sleep, during which it uses only 605 µA. The node only wakes once the battery charge rises above 50%.

During the 20 day period of the simulation, the first 2 days were assumed sunny, the next 3 dark, the next 3 sunny, the next 3 dark and finally the last 9 sunny. For a summary of the sun during the simulated days, see Figure 8.1.

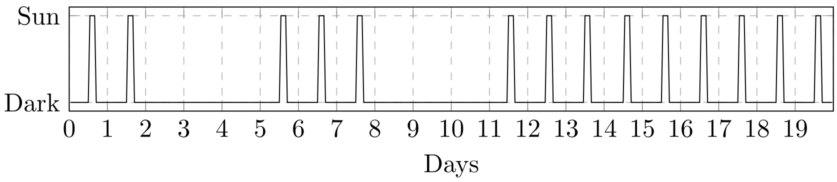


Figure 8.1: A summary of the days which the solar panel experienced sun during the simulation. Each sunny day consists of 4 hours of 1000 W m^{-2} illumination.

A sunny day in the simulation consists of 4 hours of 1000 W m^{-2} illumination. The efficiency of the LT3652-based charger is also considered by the simulation, by assuming that only 80% of the power output from the solar panel is available for charging the battery. Figures 8.2 to 8.5 show the battery charge over the 20 day period with solar panel sizes of 1 W, 2 W, 3 W and 4 W respectively. In this chapter, only the battery charge simulations are shown, Appendix C provides the simulation results showing the battery voltage during the simulations. Furthermore, the curve used to calculate the battery charge from the battery voltage is also given in the appendix.

8.1.1.1 1W Solar Panel

Figure 8.2 shows the state of charge of the battery for the duration of the simulation when a 1 W solar panel is connected to the solar charger.

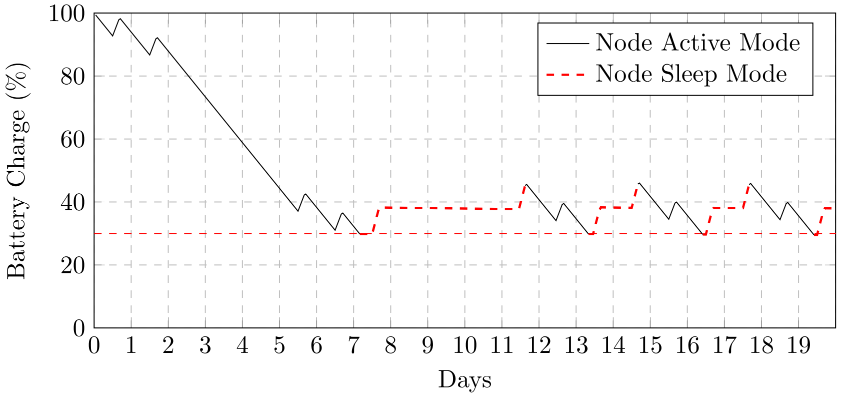


Figure 8.2: Simulated state of charge of the battery over a 20 day period with a 1 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

During the first 2 days of sun, the solar panel is not able to supply enough power to replace the battery charge used by the node. The battery charge falls consistently, with only small increases when the solar panel receives sun. On day 7 the battery charge falls below the 30% threshold (marked by the horizontal line on the figure), and the node enters deep sleep mode. It remains in the sleep state for 5 consecutive days, which includes the entire second period of darkness. Two days of sunshine are required before the battery charge reaches 50% and on day 11 the node can return to its active mode.

Even with the solar panels receiving sun, the node remains in active mode for only 2 days before the battery charge again falls below 30% and the node re-enters sleep mode. The battery requires a further two days of charging from the solar panels before it reaches 50% again. After 2 more days, it re-enters sleep mode. This cycle will continue indefinitely, unless the node reduces its power consumption or a larger solar panel is attached to the solar charger.

8.1.1.2 2W Solar Panel

Figure 8.3 shows the state of charge of the battery for the duration of the simulation when a 2 W solar panel is connected to the solar charger.

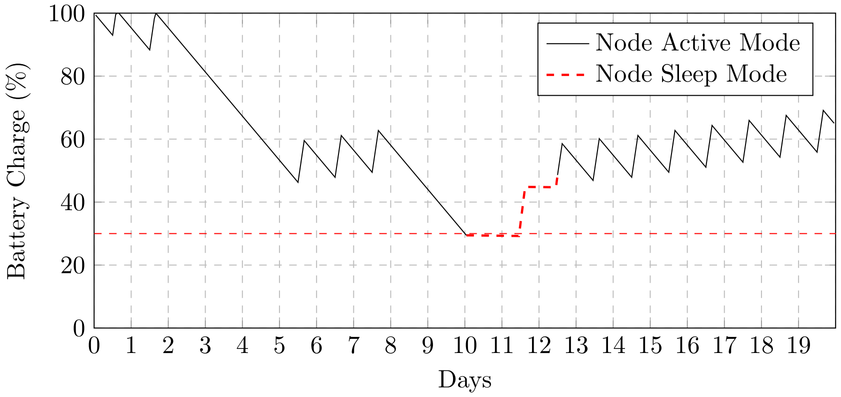


Figure 8.3: Simulated state of charge of the battery over a 20 day period with a 2 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

The 2 W solar panel is able to supply enough power to maintain the battery charge at 100% during the first 2 days of sunshine. However, after the initial 3 days of darkness, the solar panel does not provide enough power to recharge the battery to 100% within 3 days.

At the end of the first period of darkness, the battery charge is 7.85% higher than it was with the 1 W solar panel at the same time. After the darkness, with 3 days of sunshine, the battery charge is able to increase to 61% from 49%. During day 10, during the second 3 day period of darkness, the battery charge falls below the 30% threshold and the node enters sleep mode.

The battery requires 2 days of sun to restore charge to the 50% level, where the node can switch back to active mode. Once the node is in active mode again, the 2 W solar panel is able to supply enough power to avoid re-entering sleep mode.

After a full day with 4 hours of sun, the battery charge is increased by 315.377 mA h. Therefore, with this solar panel size it will take approximately 23 days to increase the battery charge from 30% to 100%. This is too slow, as our requirements are that the battery be fully charged within 3 days. Hence also the 2 W solar panel does not produce enough power to replace the charge used by the node during extreme usage.

8.1.1.3 3W Solar Panel

Figure 8.4 shows the state of charge of the battery for the duration of the simulation when a 3 W solar panel is connected to the solar charger.

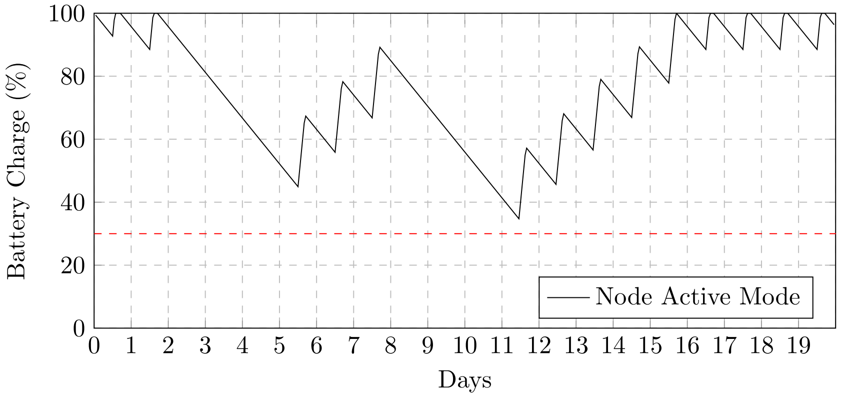


Figure 8.4: Simulated state of charge of the battery over a 20 day period with a 3 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

By comparing Figure 8.4 to Figures 8.2 and 8.3, we note that the node never enters sleep mode when a 3 W solar panel is connected to the charger. The solar panel is able to supply enough power during the sunny periods to maintain continuous node operation. After one full day of sun the battery charge is increased by 1193 mA h.

After the first 3 day period of darkness, the battery is re-charged to approximately 89% during 3 consecutive days of sunshine. During the second 3 day period of darkness, the battery charge falls to 34.7%, which is close to but still above the threshold at which the node would enter sleep mode. Thereafter, the battery is charged to full capacity within 5 days. Therefore the 3 W solar panel provides enough power to keep the node operational throughout the conditions postulated by this simulation, however, if the second period of darkness were increased in length by a single day, the node would have been forced into sleep mode. Furthermore, the 3 W solar panel is also not able to recharge the battery to full capacity within 3 days, as set out in the specifications requirements for a solar panel in Section 5.13.

8.1.1.4 4W Solar Panel

Figure 8.5 shows the state of charge of the battery for the duration of the simulation when a 4 W solar panel is connected to the solar charger.

With this solar panel, the battery is fully charged within a 3 day period after the first period of darkness. The battery capacity never falls below 44% despite the 2 periods of darkness, because it is recharged to full capacity within 3 days.

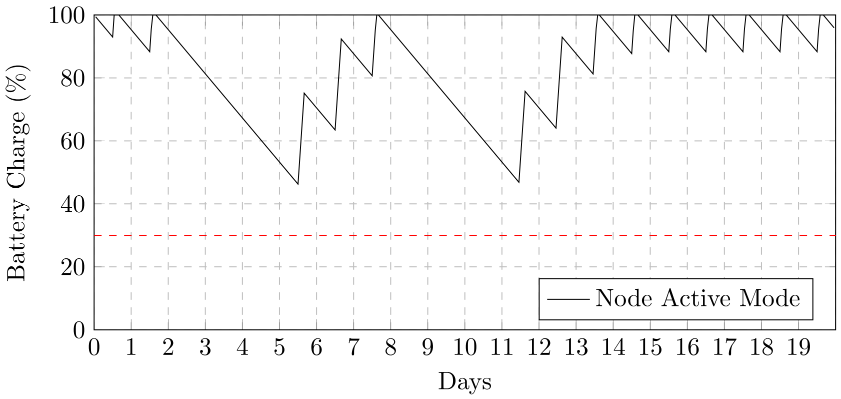


Figure 8.5: Simulated state of charge of the battery over a 20 day period with a 4 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

After one full day of sun the battery charge is increased by 2071 mA h. Therefore this solar panel will take 3 days to fully recharge the battery from 36.5% in the conditions stipulated in this simulation. While this does still not correspond to the specification of 3 days to recharge the battery from 30%, the conditions stipulated by this simulation are a worst case scenario and in practice the battery should charge much faster due to a higher number of sun hours than specified here.

Therefore, this simulation shows that the chosen combination of a battery capacity of 10350 mA h and 4 W solar panel size is suited for the energy demands from the node even during the most extreme power usage scenarios.

For figures showing the battery voltage during these simulations, please refer to Appendix C. The battery voltage was calculated using an approximation to the Li-ion discharge curve shown in Figure 5.11. The approximated battery charge to voltage curve can be found in Figure C.1.

8.1.2 Test Set-up

After our simulations indicated that a 4 W solar panel would maintain the charge of the WSN nodes, the power supply was built and practically tested to verify the validity of the simulations. The power supply was connected to the maximum expected load of 208.43 mW, a solar panel rated at 4 W provided charging power, and the battery had a capacity of 10350 mA h. The terminal voltage of the battery, the times during which the solar charger was charging the battery and the solar radiation received by the solar panels were monitored every minute for 10 consecutive days.

During the first two days, the solar panels were connected to the solar charger. They were subsequently disconnected for 3 days, and reconnected again for the final 5 days. The disconnection from the 3rd to 6th day, was done to simulate the days without sun on the solar panels. The same technique was used for the solar simulations.

The two 2 W solar panels, which were chosen for this project in Section 5.13, were placed on the roof of the Electronic and Electrical Engineering Department of Stellenbosch University. They were connected in parallel, to create an input of 4 W to the charger, with a maximum power point at approximately 9 V, and were placed at coordinates, 33°55'41.9"S 18°52'01.1"E. A photograph of this setup is shown in Figure 8.6.

The panels were secured to the roof with tape, and faced North-Northwest. Due to the slope in the roof, the panels were placed at an angle of 10 degrees relative to the ground, facing towards the sun. The test was started at 00:00 (midnight) on the 16 March and ended at 23:59 (midnight) on the 26 March. The solar panels received approximately 11.5 hours of sunlight per day.

The power supply will allow the batteries to constantly supply power to the voltage regulator, unless the battery voltage falls below 3.3 V. The load must dissipate the maximum expected power associated with the extreme mesh profile, calculated in Appendix B, to be 208.43 mW. Therefore, a resistive load of 53 Ω, was connected continuously to the output of the voltage regulator during the test. The resistors were rated at 500 mW, to ensure that they would not overheat during the test. For added safety, a small fan was used to ensure that they were kept cool. The batteries were fully charged to 4.1 V before the test began.

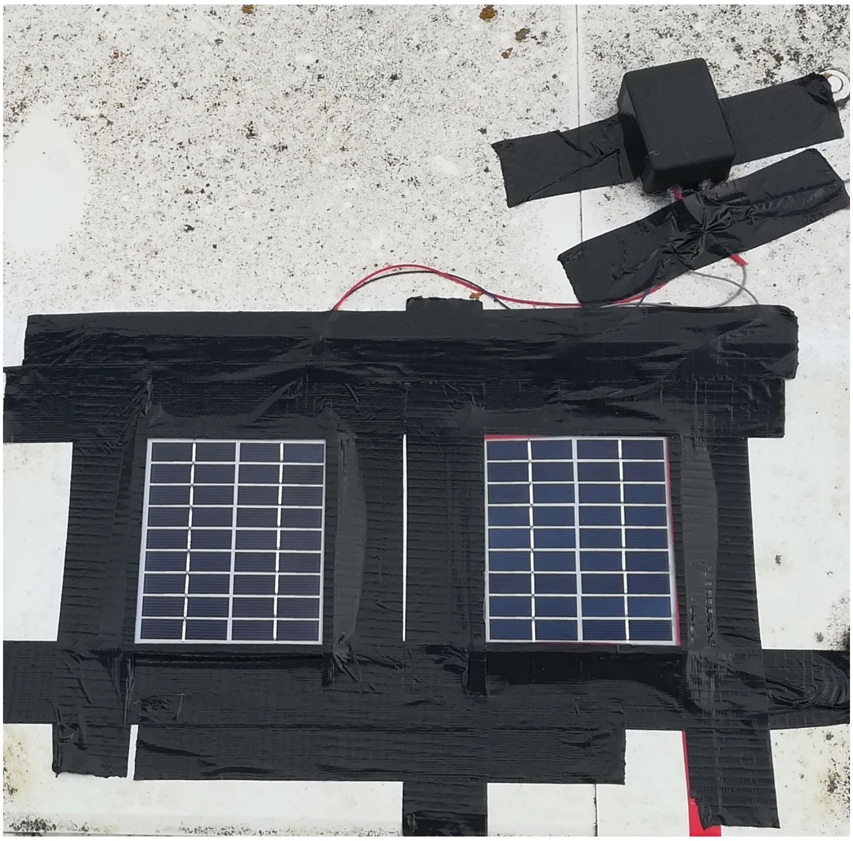


Figure 8.6: Two solar panels connected in parallel and placed on the roof of the Stellenbosch University Electronic and Electric Engineering building. Each solar panel is rated at 2 W, therefore supplying 4 W to the solar charger. The two solar panels are connected in parallel inside the enclosure on the top right.

8.1.3 Solar Radiation During The Test

The solar radiation, Direct Normal Irradiance (DNI), was measured by a weather station at the Engineering Faculty of Stellenbosch University. This weather station is located at coordinates 33°55’42.1”S 18°51’53.7”E and therefore in the immediate vicinity of the test setup. The DNI solar radiation for the duration of the test, measured each minute, is shown in Figure 8.7.

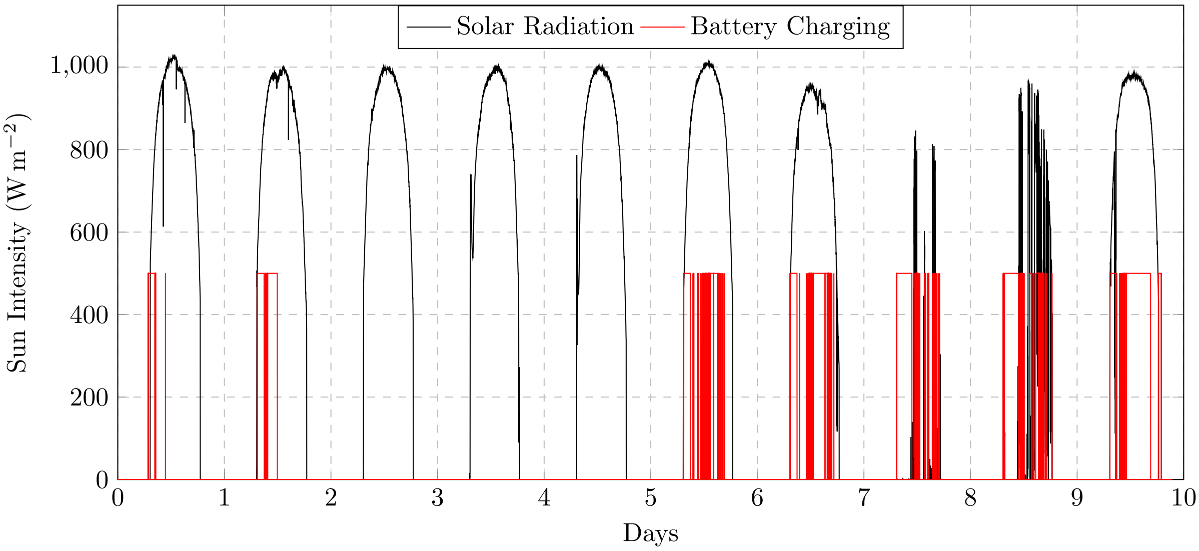


Figure 8.7: The measured solar radiation intensity (black) for the duration of the test, and the times during which the battery was charged by the solar charger (red).

The figure shows that the solar radiation increases sharply at 6:30 am, reaching a peak of 1000 $\mathrm{W\,m^{-2}}$ on some days, before dropping to zero by 19:15 pm. For the duration of the test, an average of 670 minutes of solar radiation was recorded, per day. The solar radiation on days 7 to 10 never reached 1000 $\mathrm{W\,m^{-2}}$, and during days 8 and 9 only 169 and 379 minutes with a DNI above 0 $\mathrm{W\,m^{-2}}$ were experienced respectively, .

8.1.4 Test Results

In this section the results from the test will be discussed. Specific attention will be given to the decline in battery voltage during the days when the solar panels were disconnected, as well as the ability of the solar panels to recharge the battery once the solar panels were reconnected.

8.1.4.1 Battery Voltage

The measured battery terminal voltage for the duration of the test is indicated by the black line in Figure 8.8. Corresponding simulated battery voltage is indicated by the blue line, and the red line indicates the battery voltage of the predictive simulation using the assumption of 4 hours of 1000 $\mathrm{W\,m^{-2}}$ per a sunny day. The battery voltage of the simulations will be discussed later in this chapter, therefore only the measured battery voltage (the black line) will be discussed in this section. The difference in measured battery voltage between the current and previous day, taken at midnight, is shown in Table 8.1.

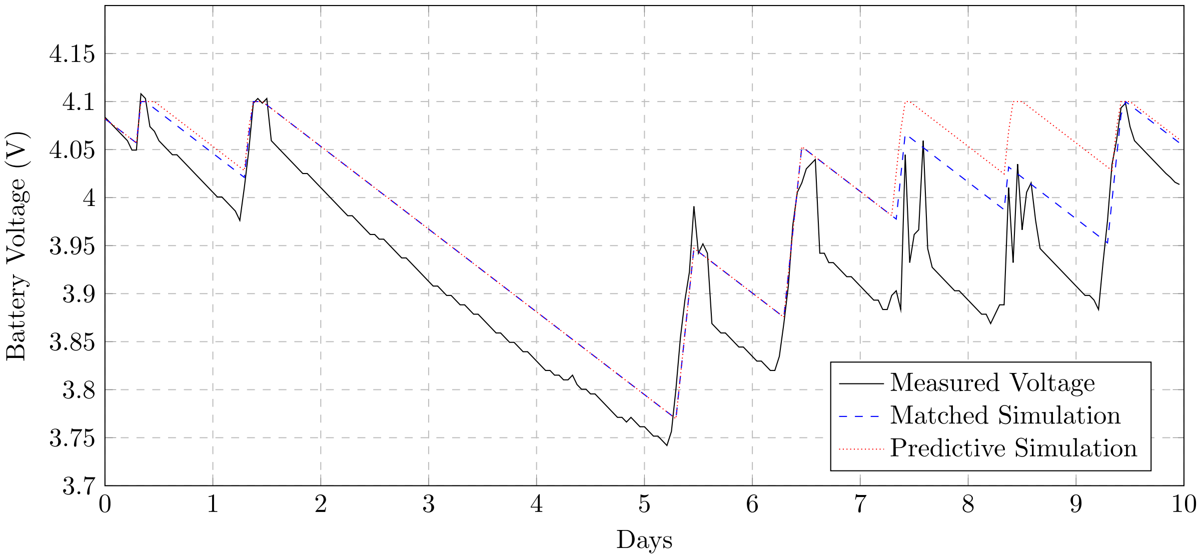


Figure 8.8: Battery voltage during simulation and practical tests. The measured terminal voltage of the battery is shown in black, while the corresponding (matched) simulation is shown in blue and the predictive simulation using the assumption of 4 hours of 1000 W m^{-2} per a sunny day is shown in red.

In Figure 8.8, during the first two days the battery voltage drops in the evening when there is no sun, and then increases to 4.1 V soon after the solar panels receive sunlight again. At midnight on the second day, the solar panel is disconnected from the charger, and the battery voltage drops for 3 consecutive days to its lowest point of 3.74 V. At midnight after the 5th day, the solar panel is reconnected to the charger, and the battery is able to recharge again.

During day 6 and 7, the sun shone throughout the day and the battery charges well, increasing by 78.13 mV on both days. If this trend were to continue, the battery would have been fully charged by the end of day 8. However, on the eighth and ninth day, the sun was mostly obscured and on the eighth day, the battery voltage decreased by 9.77 mV, while on the ninth day it increased by 19.53 mV. The tenth day, was cloudless, and the solar charger was able to charge the battery to its full 4.1 V once again. The measured battery voltage at the end of the 10-day test was 4.02 V.

Table 8.1: Daily 24-hour difference in battery voltage, measured at 24:00.

Test day	1	2	3	4	5	6	7	8	9	10
Difference [mV]	-73.24	9.77	-92.77	-78	-68.36	78.13	78.13	-9.77	19.53	103

In Figure 8.8 the fall in battery voltage immediately after the solar charger stops charging is clearly visible. During charging, the measured voltage is held high by the solar charger, but as soon as charging stops, the measured voltage returns to the terminal voltage of the battery. On days with intermittent solar charging, this is even more evident. For example, on day 8, the battery voltage peaks, and drops multiple times.

The more gradual drop in measured voltage after charging stops may be attributed to the self-discharge rate of the battery, which is determined by its physical characteristics.

For an Li-ion battery the expected self-discharge within the first 24 hours after charging is 5% and thereafter, the Li-ion battery is expected to lose 5% of its charge each month.

Self-discharge is evident during days 3, 4, and 5 of the practical test. During these days, the battery voltage drop decreases day to day. On day 3, the voltage drop is 92.77 mV, on day 4 the voltage drop is 78 mV, while on day 5 it is only 68.36 mV.

Considering the battery voltage during the test, we see that the solar charger would have been able to recharge the battery to 4.1 V after the 3 days of complete darkness, had it not been for the cloudy days on day 8 and 9. Further analysis of the time which the solar charger was actively charging the battery, and the solar radiation received during the charging is required to determine how successfully the charger and solar panels are able to convert solar energy to battery charge.

8.1.4.2 Charge Times

The time which the solar charger was actively charging the battery was identified by monitoring the n_charge and n_fault output lines of the solar charger. When the n_fault output is pulled low, and the n_charge output is pulled high, charging of the battery is in progress. These periods of active charging are indicated in Figure 8.7 by the red line. When the charger was active, the red line is pulled high, and when it was not, the line is low.

The number of minutes which the solar charger was active during each day is presented in Table 8.2. Furthermore, the percentage of time which the solar panel received certain levels of solar radiation while actively charging is also shown in the table. The solar radiation arriving at the solar panel are divided into three categories, DNI above 1000 $\mathrm{W\,m^{-2}}$, DNI between 800 $\mathrm{W\,m^{-2}}$ and 1000 $\mathrm{W\,m^{-2}}$, and DNI between 500 $\mathrm{W\,m^{-2}}$ and 800 $\mathrm{W\,m^{-2}}$. Day 3, 4 and 5 are not shown in the table as the solar panels were disconnected during these days.

Table 8.2: Summary of the number of minutes per a day which the solar charger was active, as well as the percentage of time during charging which the solar panels received solar radiation of three categories. Day 3, 4, and 5 are not included in this table, as the solar panel was disconnected during these days.

Test day	1	2	6	7	8	9	10	Total
Total Charge time [minutes]	101	234	258	354	277	348	498	2070
DNI >1000 $\mathrm{W\,m^{-2}}$ [%]	0	0	37.21	0	0	0	0	4.64
1000 $\mathrm{W\,m^{-2}}$ >DNI >800 $\mathrm{W\,m^{-2}}$ [%]	12.87	60.68	27.91	73.44	2.88	23.85	76.51	46.33
800 $\mathrm{W\,m^{-2}}$ >DNI >500 $\mathrm{W\,m^{-2}}$[%]	59.51	25.21	57.75	16.67	6.5	7.18	13.25	21.06

During day 1 the charger was only active for 101 minutes, with the majority of the charging accomplished while the DNI was between 500 and 800 $\mathrm{W\,m^{-2}}$. The test had only been running for 6 hours when the solar charger became active for the first time. Therefore, the battery had not lost much charge, and the solar charger was able to replace the expended charge quickly. During day 2, the solar charger was active for 234 minutes with the majority of the charging occuring when the DNI was between 800 and 1000 $\mathrm{W\,m^{-2}}$.

During day 6 and 7, the solar charger was active for 258 and 354 minutes respectively. Despite the charger operating for 96 minutes more during day 7 than day 6, the battery voltage increased by 78.13 mV on both days. The shorter charging time on day 6 can be explained by the charger receiving solar radiation above 1000 $\mathrm{W\,m^{-2}}$ for 37.21% of the time. On day 7, however, most of the charging occurred while the DNI was between 800 $\mathrm{W\,m^{-2}}$ and 1000 $\mathrm{W\,m^{-2}}$, with the DNI never rising above 1000 $\mathrm{W\,m^{-2}}$. Therefore, the amount of power which the charger received from the solar panels while the DNI was above 1000 $\mathrm{W\,m^{-2}}$, was so much that the charging time could be reduced by 96 minutes.

On day 8 of the test, the sun intensity rose above 500 $\mathrm{W\,m^{-2}}$ for only 25 minutes, which corresponded to 9.38% of the charging time. Despite this, by the end of the day the battery voltage dropped by only 9.77 mV. On day 9, the sun intensity rose above 500 $\mathrm{W\,m^{-2}}$ for 31.03% of the charging time, and the battery voltage increased by 19.53 mV. We conclude that, even with low solar radiation, the chosen solar panels and solar charger are able to maintain the battery charge under the most severe expected load conditions.

On day 10 of the test the battery charger is active for 498 minutes. The solar radiation during charging was between 800 and 1000 $\mathrm{W\,m^{-2}}$ for 76.51% of the charging time and the battery voltage increased by 103 mV over the course of the day, ensuring that the battery voltage reached 4.1 V before the sun set.

Day 8 of the test was characterised by low solar illumination. Despite this, by sunset, the battery voltage had dropped by only 9.77 mV relative to the previous day. This indicates that the original assumption of a day with low solar illumination resulting in no solar charging at all (such as on day 3, 4 and 5) was too conservative. The is positive for our project as it indicates that the actual loss of battery voltage on a cloudy or day with low illumination is closer to 20 mV, rather than the 68 to 93 mV measured when the solar panel was disconnected completely.

8.1.5 Results Versus Simulation

Two simulations were set up to predict the battery voltage during the test. The first simulation, hereafter referred to as the predictive simulation, was run before the test to provide a first prediction of what the voltage would be during the 10 day period. The second simulation, hereafter referred to as the matched simulation, was run after the test, to more closely match the measured voltage of the battery during the test. Both these simulations are shown in Figure 8.8, and are super imposed on the measured terminal voltage.

The predictive simulation assumed 4 hours of 1000 $\mathrm{W\,m^{-2}}$ illumination per a day, for days 1 and 2, and for days 6 to 10. During day 3,4 and 5 the solar panels were assumed to receive no sun. The matched simulation, on the other hand, assumed that 1000 $\mathrm{W\,m^{-2}}$ illumination was received for 101 minutes and 234 minutes on days 1 and 2 respectively, that no solar illumination was received during days 3, 4 and 5 and finally that 240, 240, 120, 60 and 240 minutes of 1000 $\mathrm{W\,m^{-2}}$ illumination for days 6 to 10 respectively. The times for the matched simulation were obtained by manually adjusting them until the output battery voltage closely matched the measured terminal voltage. The charge time per a day during the practical test and both simulations are summarised in Table 8.3.

Table 8.3: Summary of charge time during the practical test and simulations. The total charge time and charge time during which the DNI was above 800 $\mathrm{W\,m^{-2}}$ during the test is shown. Furthermore, the charge time for the rough simulation, and the matched simulation is shown. The total for each row is given in the last column.

Charge Time/ Day	1	2	6	7	8	9	10	Total
Total during test [minutes]	101	234	258	354	277	348	498	2070
Test with DNI >800 $\mathrm{W\,m^{-2}}$ [minutes]	13	142	168	260	8	83	381	1055
Matched Simulation [minutes]	101	234	240	240	120	60	240	1235
Predictive Simulation [minutes]	240	240	240	240	240	240	240	1680

In Figure 8.8 the most obvious difference between the battery voltage during the test and the simulations, is the voltage drop after the charger stops supplying power to the battery. As discussed earlier, this sudden drop is due to the solar charger maintaining a higher voltage at its output during charging and when the charger deactivates, the measured voltage reverts to the terminal voltage of the battery. This was not modelled in the simulation, as it was not yet known that the battery voltage would react in this way after charging. As a further improvement to the simulation, this factor can be taken into consideration.

A second observation that can be made from the figure is that the measured rate of battery discharge for an active node when the charger is inactive corresponds very closely to the simulations. During days 3, 4, and 5, the voltage of both simulations and the measured battery, decreased at the same rate. The measured battery voltage shows small deviations from the voltage predicted in the simulations, but this can be attributed to the imperfect chemical properties of the battery.

A further observation made from the figure, is that battery voltage at the end of the practical test and simulations are within 50 mV of each other, and all three are above 4 V. This shows that despite the measured and simulated battery voltages differing at times, the simulation is a useful and accurate longer-term predictor of battery capacity.

Table 8.3 shows that the total number of minutes which the solar charger spent charging the battery is more than what was predicted by both simulations. The longer charging time can be attributed to the majority of charging during the practical test occurring at a DNI between 800 $\mathrm{W\,m^{-2}}$ and 1000 $\mathrm{W\,m^{-2}}$, which is lower than the value of 1000 $\mathrm{W\,m^{-2}}$ assumed in the simulations. Therefore, the lower illumination received in the practical test, required the solar charger to operate for a longer period of time to charge to the same values.

In total, the matched simulation charged for 1235 minutes at the assumed illumination of 1000 $\mathrm{W\,m^{-2}}$, while during the practical test the solar charger charged for 1055 minutes at a DNI above 800 $\mathrm{W\,m^{-2}}$. During the practical test, the average charge time per a day was 295 minutes, compared to the expected 240 minutes of the predictive simulation and 176 minutes in the matched simulation. Once again this simply shows that during the practical test the solar illumination was lower than assumed for the simulations and therefore the solar charger required slightly longer to charge the batteries. Therefore, for a worst case scenario, the simulations should be updated to use illumination of 800 $\mathrm{W\,m^{-2}}$ instead of 1000 $\mathrm{W\,m^{-2}}$.

In conclusion, the practical test demonstrates that the simulations are a good way of predicting the general variation in battery charge over a period of several days. They provide a valuable design tool for assessing the resilience of the system to hypothesised climatic operating scenarios. For an even more accurate simulation, the drop on battery voltage after charging should be incorporated, and the expected solar illumination should be adjusted to seasonal levels.

8.1.6 Conclusion

A simulation tool was developed to assess the sufficiency of differently rated solar panels in realistic weather scenarios for our system. The simulations showed that, in conjunction with the chosen solar charger, a 4 W solar panel is able to fully recharge the chosen battery from its lowest point after 3 days with no sun. It was further shown that the simulation could accurately predict the long-term behaviour of the battery charge under an assumed weather scenario.

A 10 day practical test was undertaken to examine the solar charger and battery. It was found that the solar panel and solar charger were able to maintain the battery voltage at around 4.1 V even with the maximum expected load. After a 3 day period of darkness, the solar panels were able to fully charge the battery from a low of 3.74 V within 5 days. During the test, 2 days of cloudy weather were experienced, and the battery voltage increased by a combined value of 9.76 mV over both days. This was far better than the original assumption that the solar charger would not be able to supply any charge to the batteries on cloudy days.

Overall, the simulations and practical test confirmed that the solar panel, solar charger and other power supply electronics work as designed and are more than satisfactory to power the node for long term usage.

8.2 Network Implementation on FIT IOT-LAB Platform

The networking protocols designed during Chapter 6 were implemented on the Future Internet of Things Internet Of Things Laboratory (FIT IOT-LAB) platform, at INRIA, Lille in France. The FIT IOT-LAB platform consists of 256 nodes which have been placed inside a building. Each node includes a RF module and can be individually programmed and monitored during a test. This allows a user to monitor the flow of data traffic through the network easily and to discover points of interest during operation. Problems in the network can be quickly discovered and remedied using this system. The MAC and routing protocols, developed during this project, were implemented on nodes in the platform.

As part of the implementation, a user can choose whether a node will be an on-animal tag, a mesh node or a server node. In this way, interactions between all nodes can be tested. An example network which can be instantiated on the platform is shown in Figure 8.9.

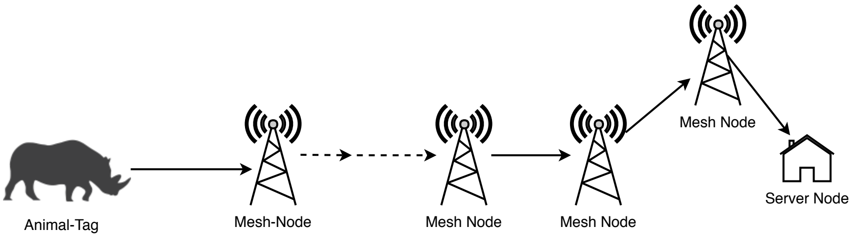


Figure 8.9: An example of a network which can be instantiated using the implementation on the FIT IOT-LAB platform. A node can be configured to be a on-animal tag, a mesh node or a server node.

8.2.1 Test Setup

The WSN430 nodes were chosen to implement the protocols on the platform at INRIA. The MCU of this node is a TI MSP430, which is similar to the MSP432 chosen for our nodes, and the radio chip is a TI CC1101, which operates at 868 MHz. This was the closest among the available options to the chosen operating frequency of 433 MHz. The MSP430 is a 16 bit processor, has 48 kB Flash and 10 kB RAM. The FIT IOT-LAB platform allows a user to program the nodes in several operating systems, including Contiki, Riot or TinyOS. For our implementation, the native MSP430 commands were used.

Each node can be accessed via the secure shell (SSH) server. Code is compiled on a server and then uploaded to each node via SSH. Hence it is never necessary to physically handle the nodes.

8.2.2 Communication Between On-animal Tag and WSN-node

It is vitally important to ensure that all communication from an on-animal tag is received by the WSN network. The following scenarios were investigated to determine how reliably the network was able to route the data from an on-animal tag by the WSN network. The test scenarios are visualised in Figure 8.10. Each scenario is summarised below:

(a) Communication from the on-animal tag directly to a server node.

(b) Communication through one mesh node to a server node.

(c) Communication through 2 mesh nodes to a server node.

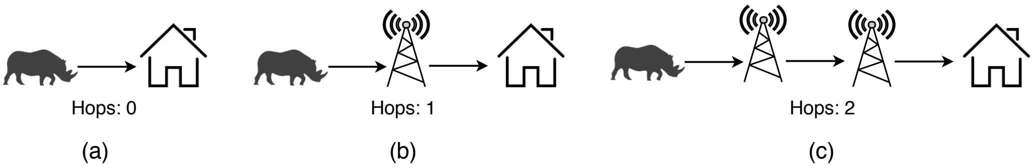


Figure 8.10: Three test scenarios used to test the on-animal tag to WSN-node communication. Scenario (a) tests communication directly from an on-animal tag to a server node. Scenario (b) tests communication from the on-animal tag to a mesh node which routes the data to the server node, and scenario (c) tests communication from the on-animal tag to a mesh node, which routes the data through a neighbouring mesh node to the server node.

In a single test of a scenario, the on-animal tag would transmit 500 packets of varying length and containing different data and the total number of packets received at the server node was recorded. Each test was repeated with two different sets of nodes to ensure that the results were not dependant on the specific hardware of any node, and in total 1000 messages were transmitted by the on-animal tag per a scenario. For the tests with 1 and 2 hops, the route to the server node from the mesh nodes was pre-configured. The on-animal tag sends its data to the node furthest from the server. The results of these tests are summarised in Table 8.4.

Table 8.4: Results of the on-animal tag to WSN-node tests, for each scenario.

Number of hops	Messages Sent	Messages Received
Zero	1000	1000
One	1000	1000
Two	1000	984

For the scenario with no hops (a) and with one hop (b), all of the data was received at the server node. For the scenario with 2 hops (c), 98.4% of the data was received at the server node. The 1.6% loss, could be due to system issues on the FIT IOT-LAB platform, or timing issues between the nodes.

8.2.3 Improvement to the On-animal Tag MAC Protocol

While completing the on-animal tag to WSN-node tests described in the previous section, it was noticed that if an on-animal tag was between 2 or more WSN nodes, the time which the mesh nodes spent negotiating which WSN node was going to receive the data from the on-animal tag was unreasonably long. This was because the MAC protocol used by the on-animal tag specified that, for every message sent, the RTS packet must always be broadcast. Each WSN-node within range of the on-animal tag would then respond to this RTS, but the DATA packet from the on-animal tag would only be sent to the WSN-node it had received a reply from first. This would leave the other WSN nodes in a waiting state for the DATA packet, which would never be delivered to them. Eventually they would time out and reset for the next message.

This timeout and reset is an unnecessary waste of time for the WSN nodes. Furthermore, while the WSN node is waiting for the DATA packet, it will receive every message that is being sent, whether it is intended for itself or not. A message that is received by a node, but is not directed for that node, is known as an overheard message. A proposed change to the MAC protocol of the on-animal tag, termed 'address saving' should reduce the number of timeouts as well as overheard messages on the WSN nodes.

8.2.3.1 Address Saving MAC Protocol

The MAC protocol of the on-animal tags was changed so that instead of broadcasting each RTS packet, the tag first attempts to send it to the address of the WSN node with which it had last communicated. Only when the RTS fails to be delivered to that previous node, is the subsequent RTS broadcast. This change to the MAC protocol will be referred to as *address saving*. A summary of the MAC protocol of the on-animal tag with and without address saving is given in Table 8.5.

Table 8.5: Summary of the on-animal tag MAC protocol, highlighting the changes made to the protocol, known as address saving.

MAC protocol (without address saving)	MAC protocol (with address saving)
On-animal tag broadcasts every RTS. WSN node which replies first is sent the data message.	On-animal tag broadcasts only first RTS. Thereafter future messages are directed to the last successful address. If the communication to this last address breaks, the RTS is re-broadcast.

8.2.3.2 Test Setup

Three cases were used to determine whether address saving in the on-animal tag MAC protocol increases the data throughput, and reduces timeouts and overheard messages by the WSN nodes. The test involved one, two and three on-animal tags placed between two mesh nodes, and the cases are summarised in Figure 8.11.

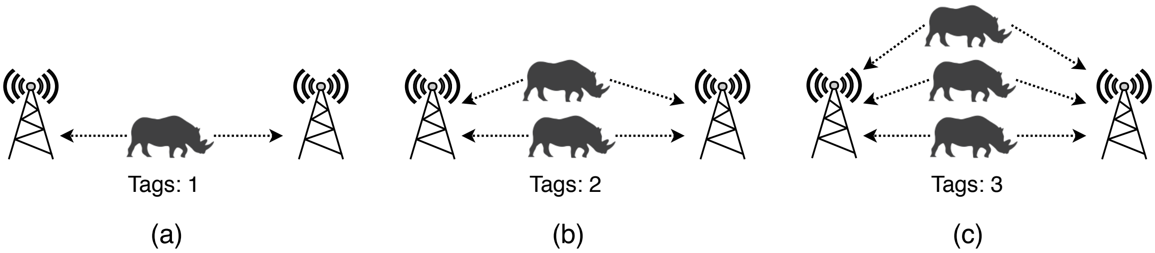


Figure 8.11: Three test cases used to determine whether the incorporation of address saving into the on-animal tag MAC protocol provides a benefit. (a) One on-animal tag placed between two WSN nodes. (b) Two on-animal tags placed between two WSN nodes. (c) Three on-animal tags placed between two WSN nodes.

For each case, each on-animal tag attempted to send 500 messages of varying data. The test was then repeated using different nodes on the FIT IOT-LAB platform to ensure the results were not dependant on the specific hardware chosen. For each test case, the original MAC protocol was evaluated first, followed by the on-animal tag MAC protocol with address saving.

8.2.3.3 Test Results

During the tests, the number of messages successfully received by the WSN nodes was recorded, as well as the total number of timeouts and overheard messages.

The percentage of messages successfully received by the WSN nodes during each test case are shown in Figure 8.12. The results for the tests without the address saving MAC protocol are shown in blue, while the results for the tests with address saving are shown in red.

During case (a) with one on-animal tag transmitting, 89.27% of messages were received by the WSN nodes without address saving, while 99.93% of messages were received by the WSN nodes with address saving enabled. This is an improvement of 10.66% in the number of successfully received messages.

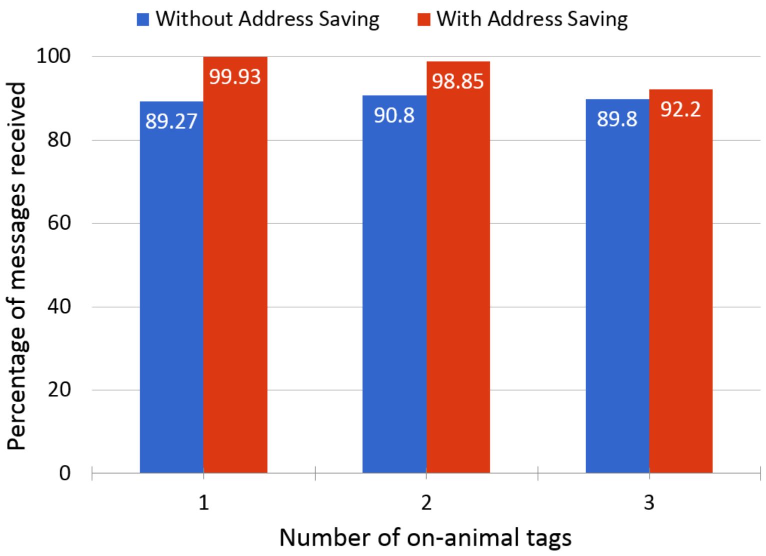


Figure 8.12: The total percentage of messages successfully received by the WSN nodes for the MAC protocol with and without address saving.

The number of successfully received messages for case (b), with two on-animal tags, and case (c), with three on-animal tags, also improved by 8.05% and 2.4% respectively when address saving was enabled. Overall, an increase of 6.71% was observed in the percentage of messages successfully received by the WSN nodes when address saving was enabled.

It was noticed that as the number of on-animal tags increased, the effectiveness of address saving diminished. It was hypothesised that this occurs due to clashes caused by the WSN nodes replying at the same time to the RTS. A solution to this issue would be to implement a delay before a WSN node responds to a broadcast RTS, so that the nodes do not reply at the same time. This delay was later implemented and results of the testing is discussed in Section 8.3.2.

The total number of packets overheard by the WSN nodes is shown in Figure 8.13. In case (a) there was only one on-animal tag between the two WSN nodes, and a total of 1625 packets were overheard by the WSN nodes. However, in this case the on-animal tag transmitted only 1000 messages. Therefore, the WSN nodes were not only overhearing packets from the on-animal tag, but also from the other WSN node. In this case when address saving was enabled, there was a decrease of 99.2% in the number of overheard packets (from 1625 to just 13).

For case (b) and (c), there was a decrease of 98.04% and 88.45% in the total number of overheard packets respectively. This is a large improvement, and frees up a significant amount of time for the WSN nodes.

The number of WSN node timeouts recorded during the tests are shown in Figure 8.14. In case (a), when one on-animal tag is present and there is no address saving, the total number of timeouts during the tests were 746. When the address saving MAC protocol was enabled, the number of timeouts decreased from 746 to just 7.

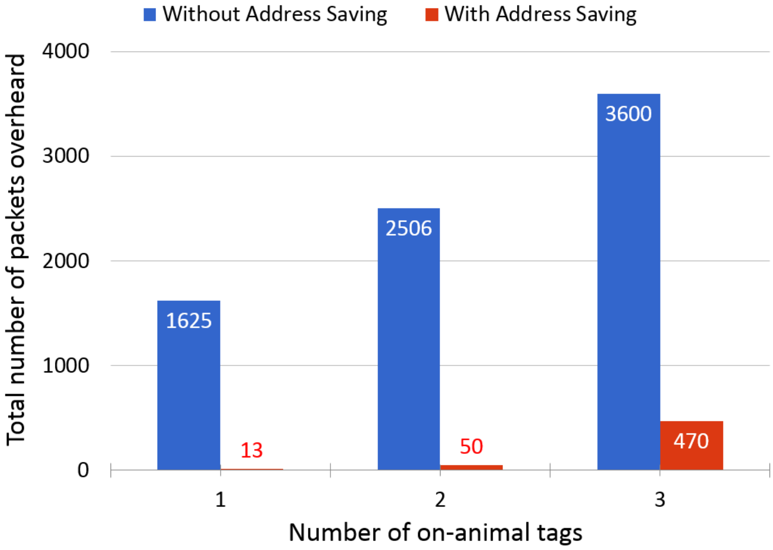


Figure 8.13: The total number of packets overheard by the WSN nodes for the MAC protocol with and without address saving. Overall there was a 93.1% decrease in the number of packets overheard by the WSN nodes.

The other test cases noticed similarly large reductions in the number of timeouts by the WSN nodes, and on average, a decrease in the number of timeouts experienced by the WSN nodes of 94.7% was seen.

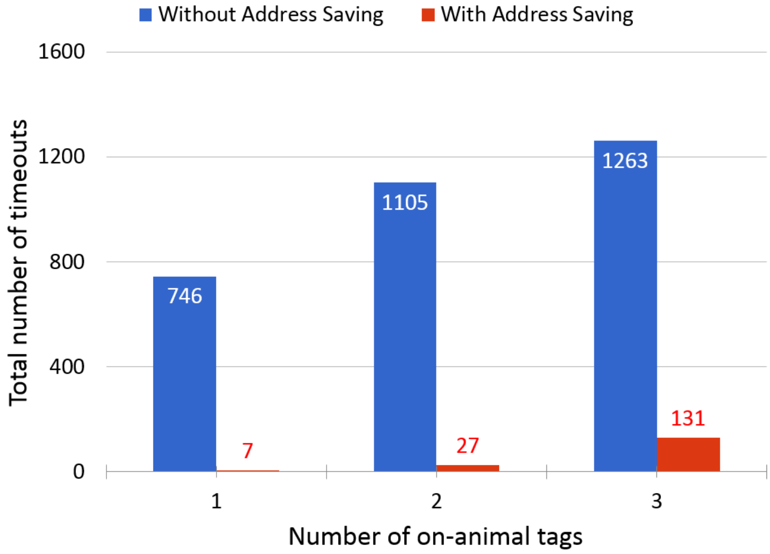


Figure 8.14: The total number of timeouts with and without address saving. Overall, 94.7% fewer timeouts are experienced by the WSN nodes over the course of the test when address saving is used by the on-animal tags.

The tests described above demonstrate that using the address saving MAC protocol on the on-animal tags can increase the number of successfully received messages while dramatically reducing the number of timeouts and overheard messages.

8.3 Laboratory Testing of the WSN Nodes

Five prototypes of the WSN nodes designed during this project were built and tested in a controlled indoor environment. Tests were run on these nodes to ensure that specific features of the MAC and routing protocol functioned as designed. Each of the five prototypes could be configured as an on-animal tag, mesh node or server node. In the laboratory setting, the nodes were placed on a flat surface approximately 30 cm from each other, with antennas pointing directly upwards. A USB cable from the computer supplied power to each node and transferred communication from the UART of the node.

The multi-hop routing from an on-animal tag through 2 and 3 mesh nodes to a server node as well as the broadcast reply delay from Section 6.3.4.1 were practically tested on this platform. The results of both these tests will be discussed in this section.

8.3.1 Multi-hop Routing

An experimental evaluation was performed in order to confirm the ability of the nodes to route data from an on-animal tag across multiple hops to a server node. The tests performed on the FIT IOT-LAB platform at INRIA (described in Section 8.2.2), already demonstrated the ability of the routing protocol to send messages across one and two hops. However, the experiments described here focus on demonstrating routing ability with two and three hops with the hardware designed for our WSN nodes. To verify this for more than four hops, further prototypes would need to be constructed.

8.3.1.1 Testing Protocol

Two multi-hop routing scenario were tested:

(a) Data from the on-animal tag is routed through 2 mesh nodes to the server node.

(b) Data from the on-animal tag is routed through 3 mesh nodes to the server node.

An illustration of both scenarios is shown in Figure 8.15. The routes to the destination server node was pre-loaded into the routing table of each node. Each scenario was tested twice, with the on-animal tag transmitting 1000 messages to the mesh node furtherest from the server node in each test. A different route to the server node is used during each test to ensure the results were not dependant on the hardware. In total, the on-animal tag transmitted 2000 messages per a scenario.

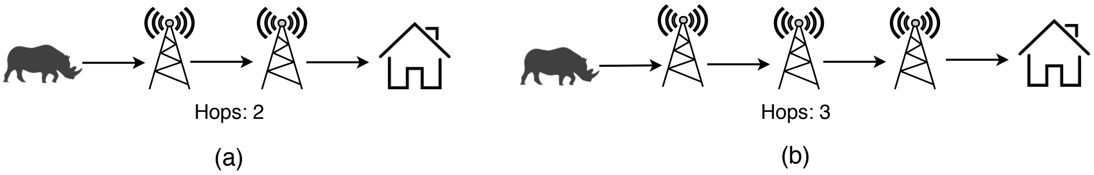


Figure 8.15: The two multi-hop routing scenarios. Scenario (a) tests communication from the on-animal tag through 2 mesh nodes to the server node, and scenario (b) tests communication from the on-animal tag through 3 mesh nodes to the server node.

8.3.1.2 Results

A summary of the number of messages sent by the on-animal tag and received by the server node for both scenarios is given in Table 8.6. For scenario (a) when the messages were routed through two mesh nodes to the server 2000 messages were sent, and all 2000 messages were received at the server node. In scenario (b) with three hops, 2000 messages were sent, and 1999 messages were received at the server node, with 1 failure.

Table 8.6: Summary of the number of messages sent by the on-animal tag, and received by the server node, after routing through two and three mesh nodes.

Number of hops	Tag Send	Server Received
Two	2000	2000
Three	2000	1999

A summary of the number of RTS packets to which the nodes replied, the total number of retransmissions as well as the number of re-acknowledgements are given in Table 8.7. In total there were 15 and 24 retransmissions for the two-hop and three-hop routes respectively. One re-acknowledgement was sent during the test with two hops, and two re-acknowledgements were sent during the test with three hops.

Table 8.7: A summary of the number of RTS packets to which the nodes replied, the total number of retransmissions and re-acknowledgements for both test scenarios.

Number of hops	RTS Replies	Retransmissions	Re-Acknowledgement
Two	6005	15	1
Three	8020	24	2

8.3.1.3 Conclusion

The routing protocol and hardware designed for our WSN is capable of routing messages across multiple hops from an on-animal tag to a server node. With the messages routed through two mesh nodes to the server, all sent messages were received, and when the messages were routed through three mesh nodes to the server node only 1 of the 2000 transmitted messages failed to be routed correctly.

8.3.2 Broadcast Reply Delay

While completing the address saving MAC protocol tests on the FIT IOT-LAB platform, it was noticed that when the RTS packet from an on-animal tag was broadcast within range of two WSN nodes, both nodes would often receive the RTS and reply simultaneously. The replies from the WSN nodes would cause a clash at the on-animal tag, resulting in a failed message. This effect would be further compounded in our WSN network, because the range of the LoRa RF module is so large, it is possible for an on-animal tag to be within communication range of multiple WSN nodes but for those nodes not to be in range of each other. In this case, when the on-animal tag transmits a broadcast RTS, all the nodes within range will reply at the same time, causing a clash at the tag.

A possible solution was to have each WSN node delay for a varying time period before issuing a reply to the broadcast RTS. In our network all messages are routed to a server node, so the delay should prioritise server nodes, secondly mesh nodes which are the least number of hops from a server node and following that, the delay should take the RSSI of the received RTS into account. A randomly generated number is incorporated into the delay to distinguish between two or more nodes who are equal in all the previous qualifications.

With all these issues in mind, Equation 6.3.1 was designed as a basis for the WSN nodes to reply to a broadcast RTS. The implementation of the delay was described in Section 6.3.4.1, and in this section practical tests will be discussed which sought to verify whether the implementation of the delay results in fewer missed messages.

8.3.2.1 Test Protocol

Only four nodes were used during this test, with one node configured as an on-animal tag, two as WSN mesh nodes and one as a WSN server node. All WSN nodes were in range of the on-animal tag. Both mesh nodes were pre-configured to route any messages they received directly to the server node. Tests were conducted first for the case with the delay disabled and then for the case when the delay was enabled on the WSN nodes.

The on-animal tag was set to send a message every 15 seconds, broadcasting the RTS packet for each message. These tests focused on verifying whether the delay improved the number of messages successfully sent when the RTS was broadcast, therefore, address saving was disabled during the tests. Each case was tested twice, with the position of the on-animal tag moved within the setup, to ensure the results were not hardware dependant. The on-animal tag transmitted 500 messages per a test, resulting in a total of 1000 messages transmitted by the on-animal tag for each case.

8.3.2.2 Results

The number of messages received at the server was recorded, as well as the number of failures by the tag to send a message. Furthermore, the number of times the WSN nodes replied to a RTS, timeouts by the WSN nodes, the number of retransmissions by any node, and the number of messages routed to the server node by the mesh nodes were also recorded.

The total number of messages sent by the on-animal tag and received by the server node are shown in Table 8.8. The table also shows the number of messages which failed to be sent by the on-animal tag and by the mesh nodes.

For the case with the broadcast reply delay enabled on the WSN nodes, all of the 1000 messages transmitted were received by the server node. For the case with the broadcast reply delay disabled, the server node only received 678 of the 1000 transmitted messages. With the broadcast reply disabled, the on-animal tag failed to send 16 messages, and the mesh nodes failed to route a total of 306 messages to the server node.

Table 8.8: Summary of the total number of messages sent by the on-animal tag and received by the server node. The number of messages which the on-animal tag failed to send, as well as the number of messages which the mesh nodes failed to route to the server node are also shown.

Delay type	Total Sent	Total Received	Tag Failures	Node Failures
Delay disabled	1000	678	16	306
Delay enabled	1000	1000	0	0

For the case with the delay disabled, the high number of messages which failed to be routed by the mesh nodes is due to the server node ignoring the majority of the RTS packets sent by the mesh nodes, because the server node has not reset after replying to the broadcast RTS from the tag. This is because when the RTS is broadcast by the tag, all three nodes reply with a CTS and enter a waiting state for the DATA packet to be sent directly to them from the tag. During this waiting state, they ignore all communication from other nodes, and only reset after 8 seconds. The on-animal tag sends the DATA packet to the address of the node which CTS it receives first and if a CTS from a mesh node is received first, it will be sent the DATA packet, and it would try to route that message to the server node. This causes an unnecessary step in the process and often the server node does not reset until after the mesh node has abandoned sending the message.

This effect is not seen when the delay is enabled, because the CTS from the server node is always sent first and therefore the DATA packet is sent directly to the server node.

To understand the full impact of the broadcast reply delay, further network statistics were inspected. For each test case, the total number of times the WSN nodes replied to a RTS, the number of messages routed to the server node by the mesh nodes and the number of timeouts by WSN nodes are given in Table 8.9.

Table 8.9: Summary of the network statistics during each test case. The number of times a node replied to an RTS and subsequent timeouts, the combined retransmissions by the tag and WSN nodes, and the number of messages passed on from the mesh nodes to the server node are all shown.

Delay type	RTS Replies	Retransmissions	Passed On	Timeouts
Delay disabled	3214	2272	594	2002
Delay enabled	3000	51	12	1992

During the tests with the delay disabled, 594 messages were successfully passed on by the mesh nodes to the server node. Considering that a total of 678 messages were successfully received by the server node during the tests with the delay disabled, and 306 messages failed to be sent by the mesh nodes to the server node, this implies that a total of 900 messages were therefore sent to the mesh nodes from the on-animal tag instead of directly to the server node. With the broadcast delay implemented, only 12 messages were routed through the mesh nodes to the server, indicating that the majority (988) of the messages were sent directly to the server node.

The number of times a WSN node replied to an RTS, and the number of timeouts was slightly higher when the delay was disabled. This can again be attributed to the large number of messages which were routed through a mesh node to the server node.

The number of retransmissions are also reduced significantly by implementing the delay. The retransmissions are separated into the retransmissions sent by the on-animal tag and those sent by the mesh nodes and shown in Table 8.10.

Table 8.10: The number of retransmissions sent by the tag and by the mesh nodes during testing.

Delay type	Tag Retransmissions	Node Retransmissions
Delay disabled	95	2177
Delay enabled	33	18

The number of on-animal tag retransmissions was reduced from 95 to 33 when the delay was enabled. The number of retransmissions by the mesh nodes was significantly reduced from 2177 to only 18 by implementing the delay. This reduction was due to the reduction in the number of messages passed on by the mesh nodes, and the consequent reduction in the total number of messages communicated in the network.

As discussed earlier in this section, without the delay, the mesh nodes received a total of 900 messages from the on-animal tag which they attempted to route to the server node. After replying to the broadcast RTS from the on-animal tag, the server node would first need to timeout before being able to receive the message from the mesh node. Because this timeout was currently set to 8 s, the mesh node usually attempted 2 or 3 retransmissions before the message was successfully sent, leading to the high number of retransmissions seen here.

8.3.2.3 Conclusion

Two conclusions can be drawn from this test. Firstly, the 8 s timeout after a node has replied to a RTS is impractically long, and further tests should be performed in order to optimise this value to reduce the number of retransmissions. Secondly, the implementation of the MAC RSSI delay before a node responds to a broadcast RTS was very successful, increasing the number of messages received from 67.8% without the delay to 100% with the delay enabled.

8.4 Outdoor Testing of the WSN Nodes

After the laboratory tests, experiments were conducted to determine how the parts of the WSN interact when placed in an outdoor environment. This was achieved in two phases. In the first phase, the maximum distance by which 3 WSN nodes could be separated while still being able to communicate with each other was determined. The WSN nodes were then fixed at those locations. In the second phase, the communication between the fixed WSN nodes and a moving on-animal tag was considered. The second phase was repeated twice, first with the antenna of the on-animal tag set to a height similar to that of a rhino's ear, and second set to a height corresponding to the height of a rhino's ankle.

8.4.1 Test Location

The tests were carried out in the areas surrounding Stellenbosch. This is primarily farmland, and has mostly small changes in elevation, similar to what might be expected during actual deployment. Much of the area around Stellenbosch is also accessible by road, allowing for the tests to be easily replicated. Furthermore, the Engineering building provided a convenient and secure place to set up a WSN node as a server station.

8.4.2 Test Hardware

Three WSN nodes and one on-animal tag were used during this test. A server node with a 5 dBi outdoor antenna was attached to a 1.5 m PVC pipe and placed on the roof of the 6th floor of the Electrical and Electronic Engineering building. A 15 m cable was used to connect the antenna to the server node. The server node was connected via UART to a computer, and a python script was executed on the computer to capture and store all output from the node.

Two WSN nodes were each placed inside waterproof plastic enclosures, and a Lithium-polymer battery was connected to the power supply. A short 50 Ω cable connected the RF module to an antenna connector on the outside of the enclosure. Because the nodes would only be installed temporarily for the test, 0 dBi monopole antennas were attached to the antenna connectors on the outside of the enclosure. The enclosures were attached to 2 m poles, which could be strapped to a longer pole if increased height was required. An image of the two mesh nodes inside their enclosures can be seen in Figure 8.16.

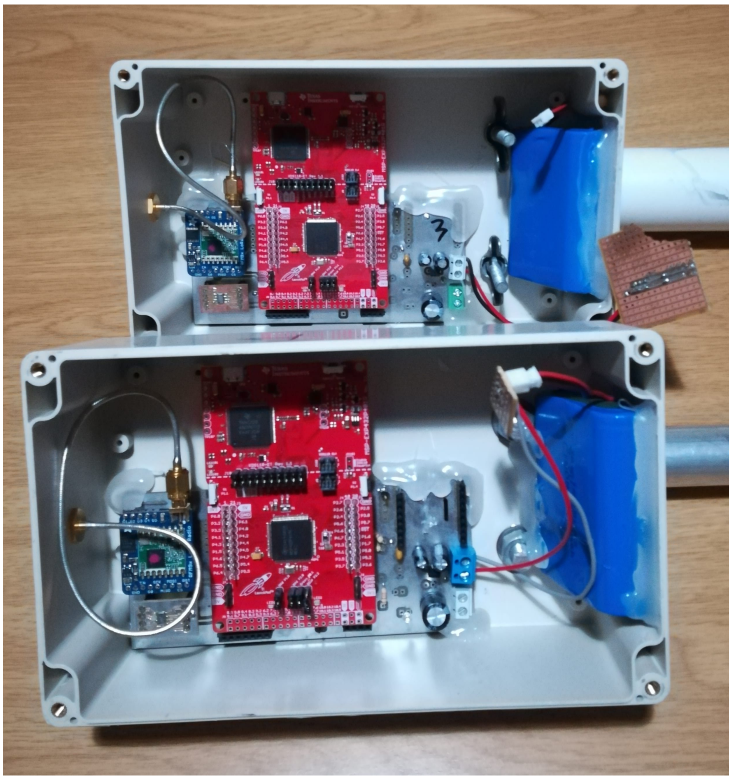


Figure 8.16: Two WSN mesh nodes inside waterproof plastic enclosures. A short 50 Ω cable was connected from the RF module to the outside of the box. The Lithium-polymer battery was connected to a 3.3 V voltage regulator.

An on-animal tag was placed in an enclosure inside a vehicle, and connected to an antenna strapped to the outside of the car. The antenna could be adjusted to two heights, 1.6 m above the ground (high), and 0.2 m above the ground (low). These antenna heights were chosen because they correspond to the approximate heights the antenna of the on-animal tag would be if attached to the ear (high) or the ankle (low) of a rhino. This antenna had a power reflection of -11.8 dB m at 433 MHz. The on-animal tag was configured with the address saving MAC protocol, while for the WSN nodes the broadcast reply delay was enabled.

8.4.3 Maximum Distance Between WSN Nodes

A practical test was conducted to determine the maximum distance which the WSN nodes could be separated, while still being able to communicate with each other. The location for this test was from the Engineering building at Stellenbosch University, along the R304 towards Malmesbury. Because we were not allowed to use private property, the nodes always had to be placed next to the road, which occasionally meant that they could not be placed on the highest point in that region. Before installing a network of WSN nodes it would be recommended to use a simulation software such as Pathloss to simulate the communication and connection range of the network.

8.4.3.1 Testing Protocol

A server node (node 1) was placed on the roof of the Electronic Engineering building, with its antenna on top of the roof and was approximately 20 m above the ground. A second node was placed inside a vehicle with its antenna attached to the outside of the vehicle at a height of 1.6 m above the ground. This node attempted to send a message to the server node every 10 s while the vehicle was driven along the R304 towards Malmesbury. Once the node in the vehicle could no longer successfully send a message to the server node, a mesh node (node 2) was placed along the roadside at a location where communication with the server node at the Engineering building was still possible.

The mesh node was made to request a route to the server node, and once it had received a reply from the server node, it was made to send 10 messages to the server node to ensure that its link was reliable. Once the link was determined to be reliable, the node was fixed at a height of 4 m above the ground, and configured as a server node. The test was then continued with the newly fixed node now acting as the server node, and continuing the journey along the R304. The node on the roof of the Engineering building (node 1) was still a server node, however, the antenna attached to the vehicle was now out of range of that node, and all communication would be sent to the newly placed server (node 2).

Once the maximum distance from node 2 was determined, a new node (node 3) would be fixed at a height of 4 m from the ground and configured as a server node.

8.4.3.2 Results

The results of the maximum distance tests are summarised in Table 8.11 and graphically illustrated on a map in Figure 8.17. The maximum distance which was achieved between node 1 and node 2 was 16.6 km. At this distance the road reached the top of a small hill and node 2 was attached to an old telephone pole, approximately 4 m above the ground.

Table 8.11: Summary of the maximum distances which separated the WSN nodes, while still being able to communicate.

Distance between nodes	1 & 2	2 & 3
Distance	16.6 km	13.35 km

The maximum distance between node 2 and node 3 was 13.35 km. In total the distance covered from node 1, to node 3 was 31.28 km.

The major limiting factor in the tests was the geographical features of the land around the road. It is hypothesised, that with the ability to place the nodes on the highest geographical points in the region (which in this case were all on private property), the distances between nodes could be further increased.

At the end of the test, node 1 was located on the roof of the Engineering building with its antenna approximately 20 m above the ground. Node 2 was located the furtherest distance from node 1 along the R304 still within range of node 1, with its antenna 4 m above the ground and node 3, was located the furtherest distance from node 2 along the R304 still within range of node 2, with its antenna 4 m above the ground.

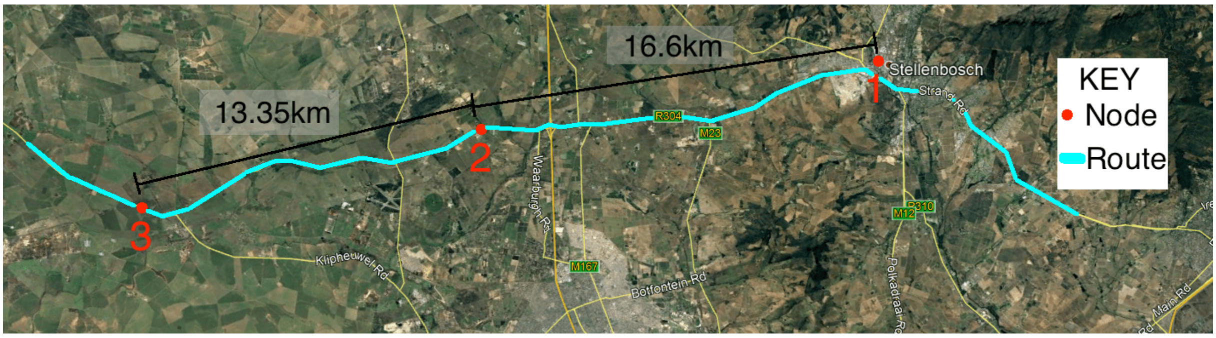


Figure 8.17: Map showing the placement of WSN nodes after the maximum distance finding test. The server node (1) was placed on the Engineering building, while node 2 (2) was placed 16.6 km away, and node 3 was placed 13.35 km away from node 2. The 46 km route which was driven during the on-animal to WSN node test is also shown on the map.

8.4.4 Message Transfer from On-animal Tag to WSN Nodes

A practical test was used to determine how successfully an on-animal tag would be able to communicate with the WSN nodes when it was moved between them. The test would also determine the maximum distance at which the on-animal tag could still communicate with the WSN nodes, how elevation and geographical features affected the transfer of messages, and whether the tag was able to successfully choose which node to communicate with when it was within range of 2 or more nodes. The goal of this test was not to test the routing ability of the WSN nodes, but rather the ability of the on-animal tag to successfully communicate with the WSN nodes over extended distances.

8.4.4.1 Testing Protocol

The 3 WSN nodes were fixed in the positions determined during the maximum distance finding test (Section 8.4.3) and were all configured as server nodes. Node 1 was positioned 20 m above the ground on the Engineering building, and nodes 2 and 3 were fixed at 4 m above the ground at their respective positions along the R304. It was not possible to place nodes 2 and 3 at the points of highest elevation in the region, as these points were on private land. Instead, they were placed on public property as close to those points as possible.

The on-animal tag was placed in a vehicle, with its antenna fixed to the outside of the vehicle. The antenna height could be adjusted to be 1.6 m (high) or 0.2 m (low) above the ground. The on-animal tag was programmed to send a message every 10 s to a WSN node. All improvements to the MAC protocol described in Section 6.3.4 were enabled on the on-animal tag as well as on the WSN nodes.

The vehicle carrying the on-animal tag, was then driven along a predetermined route, from the furthermost point from which it was possible to communicate with node 1, to the last point from which the tag could communicate with node 3. The route was approximately 46 km in length from beginning to end. The route and position of the nodes during this test are shown in Figure 8.17.

The test was first completed with the tag antenna set to the high position, and then with the antenna set to the low position.

8.4.4.2 Results: Distance Obtained

The GPS position of the vehicle during each message transmission was tracked, therefore it was possible to determine the furtherest distance over which the tag was able to communicate with each node. For the distance to be considered the maximum distance, the tag had to send 5 messages at this point without any failures or retransmissions. For both antenna height positions, the maximum distance over which the tag could communicate with each node is summarised in Table 8.12.

Table 8.12: The maximum distances over which the on-animal tag could communicate with the respective WSN node. At each of these distances, the on-animal tag was able to send 5 messages without any failures or retransmissions.

Antenna position	Node 1	Node 2	Node 3
High	17.4 km	13.4 km	8 km
Low	9 km	3 km	4 km

With the antenna of the on-animal tag set to the high position, the maximum distance between the tag and node 1 was 17.4 km. At this point, the tag was at the top of a small hill, and there was a decline in signal strength on the other side of the hill which prevented the tag from communicating to this node over a greater distance. The maximum distance over which the tag was able to communicate with nodes 2 and 3 was 13.4 km and 8 km respectively. At these distances the tag was not in a line of sight of the nodes, as both were not located at the highest elevation of the surroundings.

With the on-animal tag antenna set to the low position, the maximum distance over which communication with node 1 was possible was approximately 9 km and for nodes 2 and 3 these distances were 3 km and 4 km respectively. These distances are shorter than those achieved with the antenna in the high position, but this is to be expected as the signal from the antenna in the low position experiences far more reflection and absorption by objects close to the ground. Furthermore, the position of the antenna on the vehicle, behind the right rear wheel, could also have played a role. It is possible that the signal from the antenna was partly absorbed by the vehicle in this position.

8.4.4.3 Results: Message Data

The number of messages sent and successfully received, the number of retransmissions required, as well as the number of transmission failures are summarised for both antenna heights in Table 8.13. The total number of messages sent during the test with the antenna set to the high position was 146, and with the antenna set low, was 147. The number of messages differs slightly between the tests due to the vehicle travelling at slightly different speeds.

Table 8.13: Communication data recorded at the on-animal tag during testing.

Antenna position	Messages Sent	Messages Received	Failed	Retransmissions
High	146	135	11	78
Low	147	95	52	213

With the antenna set to the high position, of the 146 total messages sent, 135 were successfully received by a WSN node, while 11 message transmissions failed, and a total of 78 retransmissions were initiated. With the antenna set to the low position, of the 147 message sent, 95 messages were successfully received, while 52 message transmissions failed and a total of 213 retransmissions were initiated.

8.4.4.4 Results: WSN Node Switches

A summary of the number of times the tag switched between WSN nodes is given in Table 8.14. A switch between nodes is completed if a tag has successfully communicated with a particular node for one message, but fails to communicate with the same node for the next message. In this case the RTS of the subsequent message is broadcast, and if the RTS is answered by a different node, a switch is considered to have taken place. A switch between nodes could be prompted by geographical obstructions, or by the on-animal tag having reached the limits of its communication range with a particular node.

If a tag is able to switch between nodes without any further failures, it indicates that the WSN nodes are placed in positions which provide sufficient communication coverage of that area. If the tag is not able to switch nodes successfully, then it is an indication that more WSN nodes are required in that region to ensure better communication coverage, or that the current nodes must be placed in better positions.

Table 8.14 shows that the tag made 4 switches between WSN nodes with the antenna set to the high position. The switches made were from node 3, to node 2 and back again to node 3. The tag then switched back to node 2 again before the final switch to node 1,

Table 8.14: The number of times the tag switched between WSN nodes.

Antenna position	Switches between WSN nodes
High	4
Low	0

after which it only communicated with node 1 for the remainder of the test. Because the MAC protocol on the tag currently requires the communication to fail completely before the RTS is broadcast again, these switches account for 4 of the total 11 failures during the test with the antenna set to the high position.

The table also shows that the tag made no switches between WSN nodes when the antenna was set to the low position. This is an indication that for this antenna configuration, the WSN nodes are too far apart or are not placed at favourable locations to provide adequate coverage of that region.

8.4.4.5 Results: Retransmissions

To understand what impact the retransmissions had in ensuring a message is successfully received by a WSN node from the tag, the retransmissions sent by the tag were classified according to whether the message was eventually successfully received or not. The number of retransmissions which led to the successful sending of a message, and the number of retransmissions which failed to secure the success of the transmission, for both antenna configurations, is summarised in Table 8.15.

Table 8.15: The number of retransmissions associated with successful and unsuccessful message deliveries.

Antenna position	Successful retransmissions	Unsuccessful retransmissions
High	35	43
Low	25	188

With the antenna in the high position, 35 retransmissions were sent which resulted in the message being successfully received, while 43 retransmissions were sent but failed to ensure that the message was received. With the antenna in the low position, the number of retransmissions associated with successfully received messages was 25, while 188 retransmissions were associated with unsuccessful messages.

It is immediately apparent that, for both antenna configurations, the number of retransmissions sent that fail to secure message delivery was higher than those for which the message was successfully received. With the extremely low power budget of the on-animal tag, the high number of retransmissions which resulted in a failed message represents an unacceptable waste of power.

In order to understand how much of an affect the retransmissions had on the success of a message, successful messages were classified according to the number of retransmissions they required during sending. This is summarised for both antenna heights in Table 8.16.

Table 8.16: Summary of the percentage of successfully transmitted messages, according to the number of retransmissions required.

	Transmissions for successful send [%]			
Antenna position	None	1	2	3
High	82.96	10.38	4.44	2.22
Low	82.11	11.58	4.21	2.1

With the antenna in the high position, 82.96% of successfully sent messages required no retransmissions during sending. Similarly, with the antenna in the low position, 82.11% of the messages which were successfully sent required no retransmissions.

The percentage of successful messages which required 1 retransmission was 10.38% and 11.58% with the antenna in the high and low positions respectively. Furthermore, the percentage of messages successfully sent which required 2 or 3 retransmissions was 4.44% and 2.22% for the antenna in the high position and 4.21% and 2.1% for the antenna in the low position respectively.

Therefore, most successful messages required no retransmissions, irrespective of the antenna position. Furthermore, when allowing 1 retransmission, the percentage of successful messages increased by only 10.38% and 11.58% for high and low positions respectively. A single retransmission doubles the power required to send the RTS. Allowing 2 and 3 retransmissions triples and quadruples the power usage, while the percentage of successful messages only increases by 4.3% or 2.15% on average respectively.

In a system as power constrained as the on-animal tag, the amount of power expended for retransmissions may not be acceptable in view of the small increase in the chance of a message transmission success. Therefore, a change is recommended to the MAC protocol of the on-animal tag for future implementation. For non critical messages, no retransmissions should be used, and only if the message is critical should retransmissions be used.

8.4.5 WSN Routing Over Extended Distances

A practical test was conducted to determine the ability of the WSN nodes to successfully route messages from an on-animal tag through two mesh nodes to a server node over extended distances. Three WSN nodes were distributed over a total distance of 14 km with two of the nodes configured as mesh nodes and one as a server. A total of 50 messages were sent from an on-animal tag to the furtherest WSN node, and the messages were then routed from that node, through the next mesh node to the server. The same testing location and hardware as used in the maximum distance outdoor tests were employed.

8.4.5.1 Test Setup

With the server node placed on the roof of the Electronic Engineering building, mesh node 1 was fixed approximately 11.6 km away at a height of 4 m above the ground. Mesh node 2 was placed 3.5 km from mesh node 1, and its antenna was also placed 4 m above the ground. The on-animal tag was placed only a few meters away from mesh node 2.

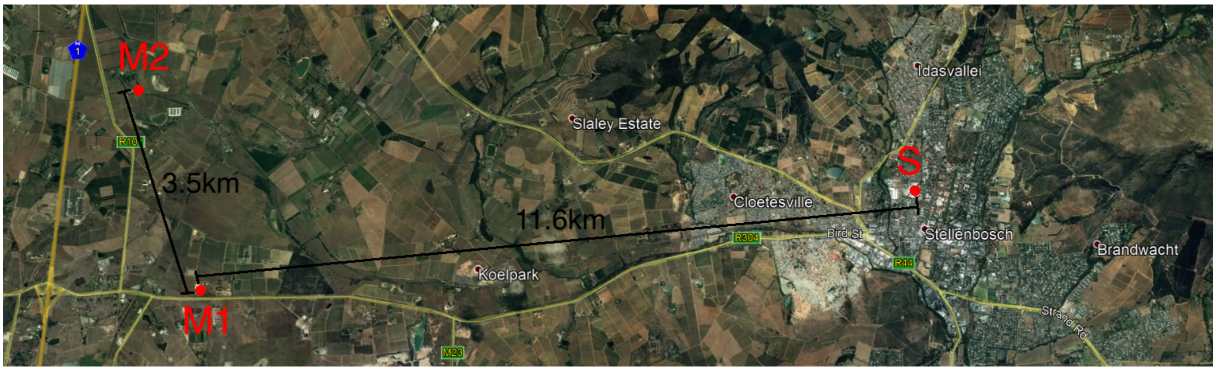


Figure 8.18: Map showing the placement of the WSN nodes for the routing over extended distance tests. The server node (S) was placed on the roof of the Engineering building, and mesh node 1 (M1) was placed 11.6 km away from the server node. Mesh node 2 (M2) was placed 3.5 km from mesh node 1. An on-animal tag sent 50 messages to mesh node 2, which was routed through mesh node 1 to the server node.

The on-animal tag sent 50 messages to mesh node 2, which passed them to mesh node 1 which in turn sent them to the server node. The test setup can be seen in Figure 8.18.

8.4.5.2 Results

A summary of the total messages sent by the on-animal tag and successfully received by the server node is shown in Table 8.17. Of the 50 messages which were sent by the tag, 48 were successfully received by the server node. Both unsuccessful messages were lost between mesh node 1 and mesh node 2.

Table 8.17: The number of messages sent by the on-animal tag, and the number of messages successfully received by the server node.

Total Sent	Total Received
50	48

For the first lost message, mesh node 2 had sent 2 RTS packets to the server node during the previous transmission. The server replied with a CTS after the second RTS, in reaction to which mesh node 2 sent the DATA packet, which the server received. However, mesh node 2 did not receive the ACK from the server and therefore resent the DATA packet to the server node. The server node retransmitted the ACK, which mesh node 2 again did not receive. Mesh node 2 retransmitted the DATA packet a third time, after which it recorded the message as a failure. While mesh node 2 was resending the DATA packets to the server node, mesh node 1 was trying to send the next message to mesh node 2. Mesh node 1 never received a reply from mesh node 2, and after 3 retransmissions it recorded the message as a failure.

For the second failed message, mesh node 1 sent an RTS unsuccessfully to mesh node 2. After one retransmission, mesh node 2 replied with a CTS. Mesh node 1 then sent the DATA packet, but received no reply from mesh node 2. After 3 retransmissions, the message was recorded as a failure.

A summary of the network activity of the on-animal tag and the WSN nodes is given in Table 8.18. The number of RTS packets to which the WSN nodes replied, the number of retransmissions and the number of message failures are all given in the table.

Table 8.18: Summary of the network activity of each node after the extended distance routing test.

Node	RTS replies	Retransmissions	Failures
Tag	N.A	1	0
Mesh node 1	50	9	2
Mesh node 2	49	8	1
Server node	48	4	0

In total, the tag performed 1 retransmission, and was able to send all its messages successfully to mesh node 1. Mesh node 1 performed 9 retransmissions and failed to send 2 messages to mesh node 2. Of the 9 retransmissions, 7 were due to the failed messages, and only 2 ultimately resulted in a successful send. Mesh node 2 performed 8 retransmissions, of which 4 were due to the ACK not received from the server node. Four more retransmissions resulted in successful messages. Of these, two were due to non-receipt of an ACK from the server node.

The server node received 48 RTS packets from mesh node 2, and sent 4 re-acknowledgements. Two re-acknowledgements were sent during the message which mesh node 2 recorded as a failure, and two more were sent when mesh node 2 failed to receive the first ACK, which eventually led to a successful send.

8.4.5.3 Conclusion

The server was able to successfully receive 48 of the 50 messages which were sent by the on-animal tag and routed through two mesh nodes. On two occasions the ACK from the server was not received by mesh node 2, and it is thought that this happened because the antenna had moved slightly causing a polarisation mismatch. Therefore, it is imperative that the antenna be attached to a solid base, so that polarisation mismatches can be minimised between WSN node antenna.

8.5 Summary and Conclusion

This chapter investigated the effectiveness of the WSN nodes and routing protocol to successfully route and receive messages from the on-animal tags. The ability of the power supply with solar energy harvester to maintain battery charge with the maximum expected load was also investigated.

A simulation tool was developed to accurately predict the long-term changes to the battery charge under specified weather and load conditions. Practical tests were performed to demonstrate that the power supply with solar energy harvester are able to power the maximum expected load for 10 days, without the battery voltage falling below 3.7 V.

The networking protocol was implemented on the FIT IOT-LAB platform at INRIA, and the ability of the protocol to successfully route messages over multiple hops was investigated. The tests found that with no hops and one hop, the server node was able to receive all messages sent. With 2 hops, the server node received 98.4% of all messages. The implementation on the platform was also used to investigate the effects of an improvement to the on-animal tag MAC protocol referred to as address saving. The tests showed that with 3 test cases, successful message reception increased by 6.7%, a 93.1% decrease in the number of overheard packets by the WSN nodes was observed, and 94.7% fewer timeouts were experienced by the WSN nodes.

The WSN nodes designed during this project were built and tested in an indoor laboratory environment. The ability of the networking protocol to successfully route messages through multiple hops was tested on these nodes. For the case with 2 hops, all messages were successfully received by the server node and with 3 hops, 99.95% of the messages sent were successfully received at the server node.

a delay before a WSN node replies to a broadcast RTS packet was tested in the laboratory environment. An increase in the number of received messages at the server node from 67.8% to 100% was observed. The tests also found that, with the delay implemented, there was a 97.7% decrease in total retransmissions by the on-animal tag and WSN nodes.

The WSN nodes were then tested in an outdoor environment to determine the maximum distance by which they could be separated and still communicate, how successful the communication was between the on-animal tag and WSN nodes, as well as how successful the nodes were at routing messages over long distances.

It was found that a WSN node positioned 20 m above the ground (on the roof of the Engineering building) and another WSN node 4 m above the ground, could communicate over a distance of 16.6 km. When both WSN nodes were positioned 4 meters above the ground, they were able to communicate over a distance of 13.35 km.

The communication between the WSN nodes and an on-animal tag was considered next. In this test the maximum distance over which communication was possible was 17.4 km when the tag antenna was 1.6 m above the ground (high) and 9 km when the antenna was 20 cm above the ground (low). Furthermore, over a 45 km route including 3 WSN nodes, the on-animal tag was able to successfully communicate 92.5% of messages with the antenna in the high position and 64.6% with the antenna in the low position.

The role of retransmissions in ensuring that a message was successfully received by a WSN node from an on-animal tag was explored next. It was discovered that 79.4% of all retransmissions by the on-animal tag during sending resulted in a failed message. Furthermore, of the messages successfully received, 82.5% were sent without the use of any retransmissions. With one and two retransmissions used during sending, the number of successfully received messages increased by only 10.9% and 4.3% respectively. Therefore, it was suggested that for the power constrained on-animal tag, the number of retransmissions should be limited to none for non-critical messages.

Finally the ability of the WSN nodes to route messages over long distances was tested. It was found that with three WSN nodes distributed over 14 km, the server node received 96% of all messages sent by the on-animal tag and routed through two mesh nodes.

In conclusion, these tests demonstrate the ability of the power supply and energy harvesting equipment to successfully power the node for long periods. Furthermore, the ability of the WSN nodes and networking protocol to successfully route messages over multiple hops has been shown, as has the ability of the on-animal tags to successfully send messages to the WSN nodes over extended distances and in difficult terrain.

Chapter 9

Conclusion and Recommendations

No single strategy to combat rhino-poaching in South Africa has been completely successful, for a number of reasons. One such reason is the difficulty of continuously monitoring the position and welfare of the animals, due to the large size of its habitat. This motivated the attempt to develop a wide area WSN based monitoring network in order to acquire real time animal position and behavioural data. The work set out in this thesis sought to design a multi-hop ad-hoc WSN to route behavioural and positional data from sensors attached to the animal to a database in real time. By identifying an operating frequency and RF module to communicate over extended distances, by designing the hardware for the WSN nodes and by choosing MAC and routing protocols, we were able to develop a wireless sensor network able to route data from the power constrained on-animal tag to a database over extended ranges.

Although each frequency presents pros and cons, it was experimentally evaluated that 433MHz offers a more compact antenna while maintaining a received signal strength similar to that achieved at 169MHz. After practically evaluating 6 commercially available radio modules, the LoRa capable RFM96W RF module was chosen. An ad-hoc multi-hop routing protocol was implemented on the FIT IOT-LAB platform at INRIA, Lille in France. It allows a user to monitor interactions between all types of nodes, with the user specifically able to monitor the flow of traffic on each node individually. The method termed "Address saving" was developed and implemented in the tag MAC protocol, and was shown to increase the messages received by WSN nodes while reducing the number of overheard packets and the number of timeouts. A further MAC protocol extension provided WSN nodes with an added delay before replying to a broadcast RTS packet. This was shown to allow all messages sent by an on-animal tag to be received, compared to only 67.8% with the delay disabled.

The on-animal tag to WSN node communication was tested using 3 WSN nodes over a 45km outdoor route. The tag antenna was configured at rhino ear-height and also at ankle-height, and it was shown that the on-animal tag was able to successfully communicate 92.5% and 64.6% of messages for these configurations respectively. The multi-hop routing ability of the WSN, tested in an outdoor environment, showed that a server node received 96% of messages routed through two hops from an on-animal tag. This indicates that the network could be successfully deployed in conditions which resemble the habitat of wild animals.

The design and development of the various hardware and software components of this research project have contributed to the current state of animal monitoring WSN research. The outcomes of this project will assist in improved monitoring of rhino behaviour and position in real time, thereby enabling quicker response times and advance warning of possible poaching situations.

9.1 Project Outcomes and Contributions

The project realised the following outcomes and contributions:

- A suitable operating frequency and commercially available radio transceiver module was selected. It was established through experimental evaluation that 433 MHz offers a more compact antenna while maintaining a received signal strength similar to that achieved by 169 MHz. Subsequently, 6 radio modules were practically compared and a LoRa capable RF module was chosen as the most suitable for the WSN.

- A prototype WSN node was designed, implemented and tested. Each node is battery powered, sustained by solar energy, and has power adaptive behaviour with multiple power-saving modes.

- An interface was developed allowing the user to set and read variables on the WSN nodes by means of UART commands. A GUI was developed to allow a user to use the UART commands without the need for a terminal emulator.

- An ad-hoc multi-hop wireless routing protocol was designed, implemented and tested. The network protocol is divided into several layers with the physical layer, MAC layer and routing layer stacked on top of one another. A WSN node can be configured as either a server node, which periodically uploads received data, or a mesh node, which routes data to the nearest server node. Any WSN node can receive messages from on-animal tags and other WSN nodes. A node can discover a route to a server, determine if a route has failed, and request an alternative route.

- The routing protocol was implemented on the FIT IOT-LAB platform at INRIA, Lille in France. The implementation allows a user to instantiate a network of up to 256 nodes, with the user choosing whether a node is an on-animal tag, a mesh node or a server node. In this way, interactions between all types of nodes can be monitored, with the user able to monitor the flow of traffic on each node individually.

- A feature addition to the MAC protocol for the on-animal tags, called address saving, was tested on the FIT IOT-LAB platform at INRIA. The feature was found to increase messages received by 6.71%, decrease timeouts on the WSN nodes by 94.7% and decrease the number of overheard packets by 93.1%.

- A further addition to the MAC protocol for the WSN nodes, added a delay before a WSN node replies to a broadcast RTS packet. It was found that WSN nodes with the delay enabled received 100% of messages sent by an on-animal tag, compared to only 67.8% with the delay disabled. Furthermore, the tests also showed that there was a 97.8% combined decrease in the number of retransmissions required by the on-animal tag and WSN nodes when the delay was enabled.

- Practical tests showed that the maximum distance which two WSN nodes could be separated and still communicate was 16.6 km.
- The on-animal tag to WSN node communication was tested over a 45 km route including 3 WSN nodes. The on-animal tag was able to successfully communicate 92.5% of messages with its antenna 1.6 m above the ground and 64.6% with its antenna 20 cm above the ground.
- The role of retransmissions in successfully sending a message from an on-animal tag to the WSN was considered during outdoor tests. It was determined that the on-animal tag was able to successfully send 82.5% of messages without the need for any retransmissions. Therefore, it was suggested that due to the constrained energy budget of the on-animal tags, retransmissions only be considered for critical messages.
- Three WSN nodes were distributed over a distance of 14 km, and messages routed from an on-animal tag through two mesh nodes to a server node. The server node received 96% of messages sent by the on-animal tag.

9.2 Recommendations and Future Work

The following are recommendations for future work by which to improve and continue the research undertaken during this project.

- Further testing is needed to determine the optimal timing of the MAC protocol features. It was determined that the 8 second timeout for a WSN node after replying to a RTS packet was impractically long, and further testing is required to determine an optimal value for this timeout.
- Future implementations of the MAC protocol on the on-animal tag, must take into consideration whether a message is critical or not in deciding whether to use retransmissions while sending.
- Future work should consider the mounting of a WSN node on a drone or air-borne device to extend the range of the WSN network during critical periods.
- The WSN should be tested with an on-animal tag attached to a rhino in its natural habitat. From these experiments the ability of the complete system designed by the RhinoNet team to monitor the animal and deliver data from the tag to a database in real time will be determined.
- The software on the WSN node should be expanded to maintain received data in memory until it has been successfully sent. With this update, if a message fails to send to a neighbouring node en route to a server, the data will be kept in memory until an alternative route to a server node is found.
- When the WSN is deployed, it is imperative that the antenna of the node be mounted to a stable platform. This will ensure that losses due to antenna polarisation mismatches between nodes will be minimised.

Bibliography

[1] Department Environmental Affairs of South AFrica, "Minister Molewa highlights progress on Integrated Strategic Management of Rhinoceros — Department of Environmental Affairs," Tech. Rep., 2017.

[2] B. Büscher and M. Ramutsindela, "GREEN VIOLENCE: RHINO POACHING AND THE WAR TO SAVE SOUTHERN AFRICA'S PEACE PARKS," *African Affairs*, vol. 115, no. 458, p. adv058, dec 2015.

[3] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in ZebraNet," in *Proceedings of the 2nd International Conference on Embedded networked Sensor Systems - SenSys '04*. New York, New York, USA: ACM Press, 2004, pp. 227–238.

[4] B. Thorstensen, T. Syversen, T.-A. Bjørnvold, and T. Walseth, "Electronic shepherd - a low-cost, low-bandwidth, wireless network system," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services - MobiSYS '04*. New York, New York, USA: ACM Press, 2004, p. 245.

[5] R. N. Handcock, D. L. Swain, G. J. Bishop-Hurley, K. P. Patison, T. Wark, P. Valencia, P. Corke, and C. J. O'Neill, "Monitoring Animal Behaviour and Environmental Interactions Using Wireless Sensor Networks, GPS Collars and Satellite Remote Sensing," *Sensors*, vol. 9, no. 5, pp. 3586–3603, may 2009.

[6] V. Dyo, S. A. Ellwood, D. W. Macdonald, A. Markham, C. Mascolo, B. Asztor, S. Scellato, N. Trigoni, R. Wohlers, and K. Yousef, "Evolution and Sustainability of a Wildlife Monitoring Sensor Network General," in *SenSys '10 Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, 2010, pp. 127–140.

[7] A. J. Garcia-Sanchez, F. Garcia-Sanchez, F. Losilla, P. Kulakowski, J. Garcia-Haro, A. Rodríguez, J. V. López-Bao, and F. Palomares, "Wireless sensor network deployment for monitoring wildlife passages," *Sensors*, vol. 10, no. 8, pp. 7236–7262, 2010.

[8] R. Zviedris, A. Elsts, G. Strazdins, A. Mednis, and L. Selavo, "LynxNet: Wild Animal Monitoring Using Sensor Networks," in *Real-World Wireless Sensor Networks*. Springer, Berlin, Heidelberg, dec 2010, pp. 170–173.

[9] R. Kays, S. Tilak, M. Crofoot, T. Fountain, D. Obando, A. Ortega, F. Kuemmeth, J. Mandel, G. Swenson, T. Lambert, B. Hirsch, and M. Wikelski, "Tracking Animal Location and Activity with an Automated Radio Telemetry System in a Tropical Rainforest," *The Computer Journal*, vol. 54, no. 12, pp. 1931–1948, nov 2011.

[10] E. Nadimi, R. Jørgensen, V. Blanes-Vidal, and S. Christensen, "Monitoring and classifying animal behavior using ZigBee-based mobile ad hoc wireless sensor networks and artificial neural networks," *Computers and Electronics in Agriculture*, vol. 82, pp. 44–54, mar 2012.

[11] F. Kiani, "Animal behavior management by energy-efficient wireless sensor networks," *Computers and Electronics in Agriculture*, vol. 151, pp. 478–484, aug 2018.

[12] Gang Lu, B. Krishnamachari, and C. Raghavendra, "Performance evaluation of the IEEE 802.15.4 MAC for low-rate low-power wireless networks," in *IEEE International Conference on Performance, Computing, and Communications, 2004*. IEEE, pp. 701–706.

[13] S. P. le Roux, "A prototype Animal Borne Behaviour Monitoring System," Stellenbosch University, Tech. Rep., 2015.

[14] Independent Communications Authority of South Africa, "The Radio Frequency Spectrum Regulations 2015," pp. 68–69, 2015.

[15] T. Rama Rao, D. Balachander, T. Nishesh, and M. V. S. N. Prasad, "Near ground path gain measurements at 433/868/915/2400 MHz in indoor corridor for wireless sensor networks," *Telecommunication Systems*, vol. 56, no. 3, pp. 347–355, jul 2014.

[16] L. Ruiz-Garcia, P. Barreiro, and J. Robla, "Performance of ZigBee-Based wireless sensor nodes for real-time monitoring of fruit logistics," *Journal of Food Engineering*, vol. 87, no. 3, pp. 405–415, aug 2008.

[17] "Range Calculation [APPLICATION NOTE]," 2015. [Online]. Available: http://www.atmel.com/Images/Atmel-9144-Range-Calculation{_}Application-Note.pdf

[18] R. Du Toit, "Modern technology for rhino management," *Pachyderm*, vol. 22, pp. 18–24, 1996.

[19] B. J. Henderson, P. Eng, R. Coudé, and P. V. Eng, "Radio Mobile Program Operating Guide," Calgary, Alberta, Canada, 2013. [Online]. Available: http://www3.telus.net/hendersb/documents/RadioMobile.pdf

[20] HopeRF Electronic, "RFM22B/23B ISM TRANSCEIVER MODULE," pp. 1–70. [Online]. Available: http://ee-classes.usc.edu/ee459/library/datasheets/RFM22B.pdf

[21] M. Bor, J. Vidler, and U. Roedig, "LoRa for the Internet of Things," in *International Conference on Embedded Wireless Systems and Networks*. Junction Publishing, 2016, pp. 361–366.

[22] Semtech, "AN1200.22 LoRa Modulation Basics," 2015. [Online]. Available: http://www.semtech.com/images/datasheet/an1200.22.pdf

[23] C. A. Hornbuckle, "Fractional-N synthesized chirp generator," 2010. [Online]. Available: https://www.google.com/patents/US7791415

[24] Semtech Corporation, "LoRa FAQ." [Online]. Available: http://www.semtech.com/wireless-rf/lora/LoRa-FAQs.pdf

[25] M. Sauter, *From GSM to LTE : an introduction to mobile networks and mobile broadband.* Wiley, 2011.

[26] A. Zogg, "Multipath delay spread in a hilly region at 210 MHz," *IEEE Transactions on Vehicular Technology*, vol. 36, no. 4, pp. 184–187, 1987.

[27] J. Wotherspoon, R. Wolhuter, and T. Niesler, "Choosing an Integrated Radio-frequency Module for a Wildlife Monitoring Wireless Sensor Network," in *AFRICON 2017.* IEEE, 2017.

[28] P. K. Daingade Shinde, "MSP430 Based mine monitoring and control using wireless sensor networks," *International Research Journal of Engineering and Technology*, vol. 02, no. 02, pp. 259–262, 2015.

[29] Y. Zhu, W. Zeng, and L. Xie, "Design of Monitoring System for Coal Mine Safety Based on MSP430 and nRF905," in *Intelligence Science and Information Engineering (ISIE), 2011 International Conference on.* IEEE, 2011, pp. 98–101.

[30] L. B. Hörmann, P. M. Glatz, C. Steger, and R. Weiss, "A wireless sensor node for river monitoring using MSP430{®} and energy harvesting," in *Education and Research Conference (EDERC), 2010 4th European.* IEEE, 2010, pp. 140–144.

[31] Steven Keeping, "Hybrid Power Supplies Deliver Noise-Free Voltages — DigiKey," 2012. [Online]. Available: https://www.digikey.com/en/articles/techzone/2012/may/hybrid-power-supplies-deliver-noise-free-voltages-for-sensitive-circuitry

[32] D. Incorporated, "AP2112 CMOS LDO REGULATOR," 2017. [Online]. Available: https://www.diodes.com/assets/Datasheets/AP2112.pdf

[33] Y. Singh, *Electromagnetic field theory.* Dorling Kindersley, 2011.

[34] F. Schut, "Geostationary Satellite," 2014. [Online]. Available: http://www.arnoldsat.com/

[35] B. W. Pike, "Power Transfer Between Two Antennas with Special Reference to Polarization," Range Systems Engineering, Tech. Rep., 1965.

[36] J. H. Reisert, "Antenna Polarization." [Online]. Available: http://www.astronwireless.com/topic-archives-antennas-polarization.asp

[37] GlobalTop Technology, "MKT3339," 2012. [Online]. Available: www.gtop-tech.com

[38] Telit Wireless Solutions, "GL865-DUAL/QUAD V3 Product Description," 2014. [Online]. Available: http://www.telit.com/fileadmin/user{_}upload/products/Downloads/2G/CL865/Telit{_}GL865-DUAL{_}QUAD{_}V3{_}Product{_}Description{_}r6.pdf

[39] W. Krüll, R. Tobera, I. Willms, H. Essen, and N. von Wahl, "Early Forest Fire Detection and Verification using Optical Smoke, Gas and Microwave Sensors," *Procedia Engineering*, vol. 45, pp. 584–594, 2012.

[40] M. Teresa Penella, J. Albesa, and M. Gasulla, "Powering wireless sensor nodes: Primary batteries versus energy harvesting," in *2009 IEEE Intrumentation and Measurement Technology Conference.* IEEE, may 2009, pp. 1625–1630.

[41] Battery University, "What's the Best Battery?" 2017. [Online]. Available: http://batteryuniversity.com/learn/archive/whats{_}the{_}best{_}battery

[42] "How do Lithium Batteries Work?" 2017. [Online]. Available: http://batteryuniversity.com/learn/article/lithium{_}based{_}batteries

[43] Olimex, "Rechargable LI-PO battery 3.7V 6600mAh." [Online]. Available: http://www.mantech.co.za/datasheets/products/LIPO6600mAh.pdf

[44] Battery University, "How to Prolong Lithium-based Batteries," 2017. [Online]. Available: http://batteryuniversity.com/learn/article/how{_}to{_}prolong{_}lithium{_}based{_}batteries

[45] B. Schneider, "A Guide to Understanding LiPo Batteries," 2017. [Online]. Available: https://rogershobbycenter.com/lipoguide/

[46] V. Quaschning, "Photovoltaic Systems," *Renewable Energy World*, pp. 81–84, 2004.

[47] "Power Characteristics of a Solar Panel." [Online]. Available: https://www.wholesalesolar.com/solar-information/solar-panel-efficiency

[48] G. M. Masters, *Renewable and Efficient Electric Power Systems.* Wiley-Interscience, 2004.

[49] "Solar Cell I-V Characteristic and Solar I-V Curves." [Online]. Available: http://www.alternative-energy-tutorials.com/energy-articles/solar-cell-i-v-characteristic.html

[50] C. Brown, "Shade Losses in PV Systems, and Techniques to Mitigate Them." [Online]. Available: http://blog.aurorasolar.com/shading-losses-for-pv-systems-and-techniques-to-mitigate-them/

[51] M. Suri, T. Cebecauer, A. Meyer, and J. van Nieker, "Accuracy-enhanced Solar Resource Maps of South Africa," in *Third Southern African Solar Energy Conference*, 2015, pp. 450–456.

[52] J. Drew, "Designing a Solar Cell Battery Charger," *Linear Technology Magazine*, pp. 12–15, dec 2009.

[53] A. El-Hoiydi, "Aloha with preamble sampling for sporadic traffic in ad hoc wireless sensor networks," in *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, vol. 5. IEEE, pp. 3418–3423.

[54] W. Charfi, M. Masmoudi, and F. Derbel, "A layered model for wireless sensor networks," in *2009 6th International Multi-Conference on Systems, Signals and Devices.* IEEE, mar 2009, pp. 1–5.

[55] HopeRF, "RFM95/96/97/98 Datasheet." [Online]. Available: http://www.hoperf.com/upload/rf/RFM95{_}96{_}97{_}98W.pdf

[56] B. Ray, "14 LoRa FAQs Answered," 2015. [Online]. Available: https://www.link-labs.com/blog/lora-faqs

[57] P. Prabakaran, "Tutorial on Spread Spectrum Technology," 2003. [Online]. Available: https://www.eetimes.com/document.asp?doc{_}id=1271899

[58] P. Zhang and H. Liu, "An Ultra-Wide Band System with Chirp Spread Spectrum Transmission Technique," in *2006 6th International Conference on ITS Telecommunications*. IEEE, jun 2006, pp. 294–297.

[59] A. Raja and X. Su, "Mobility handling in MAC for wireless ad hoc networks," *Commun. Mob. Comput*, vol. 9, pp. 303–311, 2009.

[60] V. Bharghavan, A. Demers, S. Shenker, L. Zhang, V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 4, pp. 212–225, oct 1994.

[61] Z. Haas and Jing Deng, "Dual busy tone multiple access (DBTMA)-a multiple access control scheme for ad hoc networks," *IEEE Transactions on Communications*, vol. 50, no. 6, pp. 975–985, jun 2002.

[62] C.-K. Toh, V. Vassiliou, G. Guichal, and C.-H. Shih, "MARCH: a medium access control protocol for multihop wireless ad hoc networks," in *MILCOM 2000 Proceedings. 21st Century Military Communications. Architectures and Technologies for Information Superiority (Cat. No.00CH37155)*, vol. 1. IEEE, pp. 512–516.

[63] R. Nelson and L. Kleinrock, "Spatial TDMA: A Collision-Free Multihop Channel Access Protocol," *IEEE Transactions on Communications*, vol. 33, no. 9, pp. 934–944, 1985.

[64] C. E. Perkins, P. Bhagwat, C. E. Perkins, and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," in *Proceedings of the conference on Communications architectures, protocols and applications - SIGCOMM '94*, vol. 24, no. 4. New York, New York, USA: ACM Press, 1994, pp. 234–244.

[65] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," Tech. Rep., jul 2003.

[66] Z. Haas, M. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," 2002.

[67] S. Kaijage, "Dunia Yetu: THE ITU," 2012. [Online]. Available: http://stanleykaijage.blogspot.com/2012/05/world.html

Appendices

Appendix A

Regions

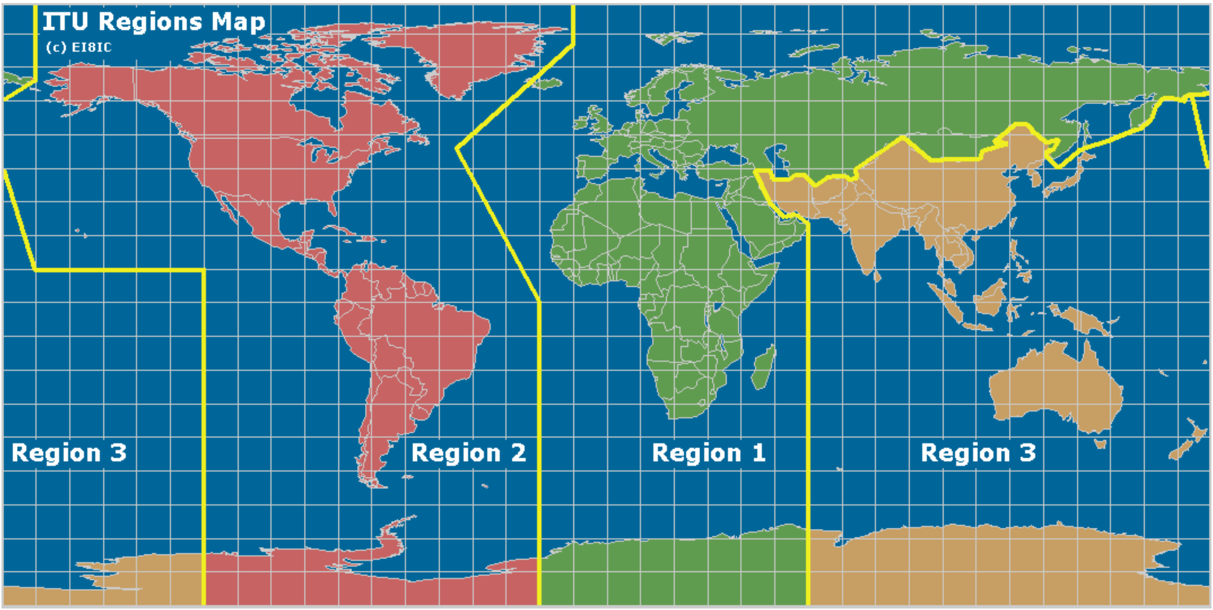


Figure A.1: This figure shows the regions in world in which standard ISM bands are the same. For region 1 the 433MHz band is reserved as an ISM band [67].

Appendix B

Node Power Usage Profiles

This appendix will discuss the calculations performed for the four power profiles, used to determine the power required by a node during normal and extreme usage. Firstly, the actions which a node can perform are defined and grouped into activity states after which the current required by the node for each state is then determined. The time taken for the node to perform each action required by the network protocol, described in Chapter 6 to send or receive a message is discussed.

Four power profiles are then proposed, namely the normal mesh, normal server, extreme mesh and extreme server power profiles. These power profiles describe the power required by a mesh and server node during expected and extreme usage conditions. The profiles can then be used to model the power usage by the node, and inform design decisions regarding the electronics making up the power supply.

B.1 Node Activities

The time which a node spends completing specific tasks can be separated into several activities. Each activity will consist of an action which the node will perform, such as transmitting using the RF module or obtaining GPS positional data. The current required by each hardware component during an activity can then be used to calculate the current used by the node when an activity is completed. The activities, current required by the node during an activity and an assigned symbol is given in Table B.1. Reading from the temperature sensor is not included as the power used by this component is negligible.

The currents used by the node during an activity is shown in Table B.1 and are calculated using Equations B.1.1 to B.1.8. During an activity when a component is not specifically needed, it will be powered down. The only exceptions are the RF module and MCU, as these are used frequently, and therefore will be placed in sleep mode when not in use. The current required for each component in each mode is given in Table 5.2.

For example, when a reading is taken from the GPS, the current can be calculated using Equation B.1.5. During this activity, the MCU (MCU_{ACT}) and GPS (GPS_{ACT}) will be active, while the RF module is in sleep mode (RF_{SLP}) and all other components are powered down.

$$I_{PD} = MCU_{PD} + RF_{PD} + GPS_{PD} + GSM_{PD} + Temp_{PD} \tag{B.1.1}$$

$$I_{SLP} = MCU_{SLP} + RF_{SLP} + GPS_{PD} + GSM_{PD} + SD_{PD} + Temp_{PD} \tag{B.1.2}$$

$$I_{SD} = MCU_{ACT} + RF_{SLP} + GPS_{PD} + GSM_{PD} + SD_{ACT} + Temp_{PD} \tag{B.1.3}$$

$$I_{GSM} = MCU_{ACT} + RF_{SLP} + GPS_{PD} + GSM_{ACT} + SD_{PD} + Temp_{PD} \tag{B.1.4}$$

$$I_{GPS} = MCU_{ACT} + RF_{SLP} + GPS_{ACT} + GSM_{PD} + SD_{PD} + Temp_{PD} \tag{B.1.5}$$

$$I_{RF-RX} = MCU_{ACT} + RF_{RX} + GPS_{PD} + GSM_{PD} + SD_{PD} + Temp_{PD} \tag{B.1.6}$$

$$I_{RF-RX_{SLP}} = MCU_{SLP} + RF_{RX} + GPS_{PD} + GSM_{PD} + SD_{PD} + Temp_{PD} \tag{B.1.7}$$

$$I_{RF-TX} = MCU_{ACT} + RF_{TX} + GPS_{PD} + GSM_{PD} + SD_{PD} + Temp_{PD} \tag{B.1.8}$$

Table B.1 shows that the activities which require the most current are the GSM communication and the RF transmission. Both of these activities require over 100 mA. Therefore, in order to minimise the amount of time which the node spends in these states by only utilising them when necessary.

Table B.1: Current used by the node in the following states.

Activity	Current Usage [mA]	Symbol	Description
Power Down	6.1×10^{-3}	I_{PD}	Node Powered down
Sleep	605.45×10^{-3}	I_{SLP}	Node Sleep mode
SD wr	4.48	I_{SD}	SD card write
GSM Com	233	I_{GSM}	GSM Communication
GPS Com	28	I_{GPS}	GPS Communication
RF-Rx	15.36	I_{RF-RX}	RF Receive
RF-Rx$_{SLP}$	12.71	$I_{RF-RX_{SLP}}$	RF Receive Sleep
RF-Tx	123.26	I_{RF-TX}	RF Transmit

Assumptions can be made regarding the amount of time a node will remain in a certain activity state under different circumstances. The amount of power required by the node under these circumstance can then be used to form a power profile. Four power profiles, describing a mesh and server node under expected and extreme conditions will be considered. For each of these power profiles, the average current required by the node will be calculated. These values can then be used to determine the battery capacity and other power related electronics to power the node. In all calculations, the time which the GPS and SD card remains in its active state is so small that it is regarded as negligible.

B.2 RF Communication States

The nodes are built to form part of an RF network, and therefore, the majority of a nodes time will be spent on network functions. It is therefore important to know exactly how long each action requires to complete. For example when sending a message, the node will follow its set protocol, setting the RF module to transmit (Tx) a Request To Send (RTS), then to receive (Rx) the Clear To Send (CTS), set the RF module to transmit the data and finally receive the acknowledgement. The MAC and routing protocol are explained in Chapter 6. Here we consider the minimum and maximum time required by the RF module to send and receive a message. For the power profile calculations, a duration between this minimum and maximum is used.

B.2.1 Sending

Table B.2 shows the time which the node spends completing each stage of the sending protocol. To send a message, the node will first do a CS check, then transmit (Tx) the RTS, then Rx the CTS, then Tx the data packet and finally Rx the Acknowledgement (ACK). The minimum time gives the time which each stage takes if there are no delays or clashes and the payload of the data packet is 0 bytes. The maximum time shows the longest time a node could take per a stage while sending a message. In this column, the RTS has been retransmitted 3 times, and the payload of the data packet is 244 bytes.

The adjusted time shows the times of each stage when the times are adjusted to account for retransmissions and delays and the data packet is 60 bytes. In this column, the time taken to transmit the RTS and complete the CS check has been increased by 20% from the minimum time to accommodate for possible retransmissions of packets or a delay during the CS. The Rx values are increased a further 25% above the Tx times to accommodate for the time which the power amplifier of the RF module takes to ramp up, additional waiting due to timeouts and missed packets caused by collisions or retransmissions. The minimum, maximum and adjusted time taken for a node to transmit a message is given in Table B.2.

Table B.2: Summary of the minimum and maximum time taken to send a message. An adjusted time is given which considers the possibility of a retransmissions, a data packet which is 60 bytes long, and delays during receiving a packet.

Activity State	Minimum time [ms]	Maximum time [ms]	Adjusted Time [ms]
CS	28	4000	33.6
Tx RTS	594	2376	713
Rx CTS	594	7794	892
Tx Data	700	7814	2939
Rx ACK	594	2394	892
Total	2510	21984	5173.6

From Table B.2, one can see that the minimum time taken to send a message with no data payload, is 2510 ms. The maximum time to send a message with a payload of 244 bytes is 21984 ms. This time is made up of the maximum CS delay time of 4000 ms, the RTS which is transmitted a total of 4 times, and the maximum waiting time of 1800 ms to wait for the CTS after each RTS is sent. After the RTS is sent for the fourth time, the CTS is received and the node then sends the data message with a payload of 244 bytes. This takes 7814 ms and the node finally receives the ACK after the longest waiting time.

The adjusted time considers that there is a retransmission of the RTS 20% of the time, 60 bytes are sent as the data payload and the waiting times for the receiving of a packet are 25% longer than the sending times. The total time to send a message with the adjusted time is 5173.6 ms.

B.2.2 Receiving

The time taken per a packet to receive a message is shown in Table B.3. The table gives the minimum time to receive a message, the maximum time and the adjusted time.

Table B.3: Summary of the minimum and maximum time taken to receive a message. An estimated extended time is given which considers the possibility of a retransmissions, a data packet which is 60 bytes long and other delays during receiving.

Activity State	Minimum time [ms]	Maximum time [ms]	Adjusted time [ms]
Rx RTS	594	2394	892
Tx CTS	594	2376	713
Rx Data	700	9414	3674
Tx ACK	594	2376	713
Total	2482	14778	5992

The minimum expected time to receive a message is 2482 ms, while the maximum expected time to receive a message is 14778 ms. The maximum time is made up of the longest wait time for the RTS, the CTS having to be retransmitted 4 times and the longest waiting time for the data packet with a payload of 244 bytes. The ACK is also transmitted 4 times.

The adjusted time considers a 20% chance of resending the CTS and ACK, and a 25% extra waiting time for the data packet with a payload of 60 bytes. The adjusted time to receive a message is 5992 ms.

B.2.3 Time Summary

The adjusted time will be used for the calculations of the power profiles, therefore it will only be considered for the rest of the chapter. The times that a node is in the RF-Rx state and RF-Tx state during sending and receiving are shown in Table B.4. It is assumed that during the CS, the node is in Rx mode.

Table B.4: Summary of the time a node spends in RF-Rx or RF-Tx activity state when receiving and sending a message. These values are taken from the adjust time.

Message	RF-Rx [ms]	RF-Tx [ms]
Sending	1817.6	3652
Receiving	4566	1426

When a message is sent, the node will be in Rx mode and Tx mode for 1817.6 ms and 3652 ms respectively. When a message is received, the node will be in Rx mode and Tx mode for 4566 ms and 1426 ms respectively.

B.3 Power Profiles

A power profile is used to estimate the current and power required by a mesh and server node under normal and extreme conditions. Four power profiles will be discussed, namely the normal mesh and normal server power profiles as well as the extreme mesh and extreme server power profiles. The normal power profiles describes nodes which receive a message from the RF network every 90 seconds, while the extreme mesh and extreme server power profiles, describe the power requirements of a node which is receiving messages continuously. The mesh node will always pass the received message on to the next hop en route to a server node, while the server node will store the received data and periodically upload via a GSM modem. Using these specifications, the time which the node spends completing each activity can be calculated, and this will be multiplied by the current required to complete that activity.

B.3.1 Power Profile: Normal Mesh

The normal mesh power profile describes a mesh node during expected usage conditions. In this profile it is assumed that the node will receive a message every 90 seconds, which will be passed to the next hop en route to a server. When the node is not actively receiving or sending a message it will remain in receive sleep mode.

The node will first receive a message which takes 5992 ms, after which the node will send that message to the next node which takes approximately 5173.6 ms. Equation B.3.1 is used to calculate the amount of time which the node is in the RF-Rx state for every message. The sum of the time which the node is in the RF-Rx state during sending $T_{RX_{SEND}}$ and receiving $T_{RX_{RECEIVE}}$, is 6383.6 ms.

$$\begin{aligned} T_{RX} &= T_{RX_{SEND}} + T_{RX_{RECEIVE}} \\ &= 1817.6 + 4566 \\ &= 6383.6\text{ms} \end{aligned} \tag{B.3.1}$$

The time which the node remains in the RF-Tx state during sending ($T_{TX_{SEND}}$) and receiving ($T_{TX_{RECEIVE}}$) of a message is calculated using Equation B.3.2 to be 5078 ms per a message.

$$\begin{aligned} T_{TX} &= T_{TX_{SEND}} + T_{TX_{RECEIVE}} \\ &= 3652 + 1426 \\ &= 5078\text{ms} \end{aligned} \tag{B.3.2}$$

If a message is received every 90 seconds, in one hour the number of messages a node will receive is $\frac{3600\text{s}}{90} = 40$. The percentage of time that the node is in the RF-Tx state, can be calculated by multiplying the time per a message which the node is in that state by the number of messages received during one hour. Using Equation B.3.3, the node will remain in the RF-Tx state for 5.64% of an hour.

$$\begin{aligned} R_{TX} &= \frac{TxTime \times 40}{1\ hour} \\ &= \frac{5078\text{ms} \times 40}{3600\text{s}} \\ &= 5.64\% \end{aligned} \tag{B.3.3}$$

A similar calculation is done to calculate the percentage of time the node is in the RF-Rx state. Equation B.3.4 shows that the node is in the RF-Rx state for 7.09% of an hour.

$$\begin{aligned} R_{RX} &= \frac{RxTime \times 40}{1\ hour} \\ &= \frac{6383.6\text{ms} \times 40}{3600\text{s}} \\ &= 7.09\% \end{aligned} \tag{B.3.4}$$

If the node is in the RF-Rx_sleep state for the rest of the hour, the percentage of time the node is in this state can be calculated using Equation B.3.5 to be 87.27%.

$$\begin{aligned} R_{RX_{SLP}} &= 100 - R_{TX} - R_{RX} \\ &= 87.27\% \end{aligned} \tag{B.3.5}$$

Therefore, under normal conditions a mesh node will be in the RF-Tx state R_{TX} for 5.64%, the RF-Rx state R_{RX} for 7.09% of time and in the RF-Rx_sleep $R_{RF-RX_{SLP}}$ for 87.27% of the time. Equation B.3.6 calculates the average current used by a mesh node in the normal mesh power profile (I_{MESH}), by multiplying the percentage of time the node is in each state by the current required during that state.

$$\begin{aligned} I_{MESH} &= R_{RF-RX_{SLP}} \times I_{RF-RX_{SLP}} + R_{RX} \times I_{RF-RX} + R_{TX} \times I_{RF-TX} \\ &= \frac{87.27}{100} \times 12.71 + \frac{7.09}{100} \times 15.36 + \frac{5.64}{100} \times 123.26 \\ &= 19.13\text{mA} \end{aligned} \tag{B.3.6}$$

Thus, the average current for a node in the normal mesh power profile will be 19.13 mA. Equation B.3.7 calculates the power required by the node, by multiplying the average current by the voltage at which it is supplied V_{SUPPLY}.

$$\begin{aligned} P_{MESH} &= I_{MESH} \times V_{SUPPLY} \\ &= 19.13 \times 10^{-3} \times 3.3 \\ &= 63.139\text{mW} \end{aligned} \tag{B.3.7}$$

Therefore, a mesh node in the normal mesh power profile will require and average current of 19.13 mA and 63.14 mW of power.

B.3.2 Power Profile: Normal Server

The normal server power profile describes a server node which receives a message from the RF network every 90 s and uploads the received data via the GSM every 5 minutes. The server node is the final destination of the message and therefore does not pass the message to the next node. An upload to an online database via the GSM modem takes 10 seconds in total. When the server node is not receiving a message or uploading via the GSM it will be in the receive sleep mode.

The total time which the GSM will be used T_{GSM} in one hour is calculated using Equation B.3.8 to be 120 s. The node is therefore uploading via the GSM for 3.33% of an hour.

$$\begin{aligned} T_{GSM} &= \frac{10\text{s} \times 60}{5} \\ &= 120\text{s} \end{aligned} \tag{B.3.8}$$

Using Table B.4, the time which the server node will be in RF-Tx state while receiving a message is 1426 ms, and it is in the RF-Rx state for 4566 ms per an incoming message. If a message is received every 90 seconds, then the total number of messages the node receives in 1 hour will be 40.

The percentage of time which the node is in the RF-Tx state (R_{TX}) per an hour is calculated using Equation B.3.3. With a Tx time of 1426 ms, the server node is in the RF-Tx state for 1.58%. Furthermore, the time which the node is in the RF-Rx state (R_{RX}) can be calculated using Equation B.3.4. With an Rx time of 4566 ms, the server node is in the RF-Rx state for 4.06%.

When the server node is not uploading via the GSM, or receiving messages it will be in the RF-Rx_sleep state. Equation B.3.9 calculates the percentage of time the node is in the RF-Rx_sleep state by subtracting the times the node is in the other states. The node is therefor in the RF-Rx_sleep state for 91.03%.

$$\begin{aligned} R_{RX_{SLP}} &= 100 - R_{TX} - R_{RX} - R_{GSM} \\ &= 91.03\% \end{aligned} \tag{B.3.9}$$

Now that the ratios of each state are know, Equation B.3.10 is used to calculate the current I_{SERVER} used by the server node in this power profile.

$$\begin{aligned} I_{SERVER} &= R_{RX_{SLP}} \times I_{RF-RX_{SLP}} + R_{RX} \times I_{RX} + R_{TX} \times I_{TX} + R_{GSM} \times I_{GSM} \\ &= \frac{91.03}{100} \times 12.71 + \frac{4.06}{100} \times 15.36 + \frac{1.58}{100} \times 123.26 + \frac{3.33}{100} \times 233 \\ &= 21.9\text{mA} \end{aligned} \tag{B.3.10}$$

The current used by a server node in the normal server power profile is 21.9 mA, and using Equation B.3.7 the power used in the normal server power profile is 72.27 mW.

B.3.3 Power Profile: Extreme Mesh

The extreme mesh power profile can be described as a mesh node which receives messages from the RF network continuously and routes those messages to the next hop en route to a server. The node will spend no time in any other activity states.

The time per a message which the node is in the RF-Rx state T_{RX} is 6383.6 ms and the time per a message which the node is in the RF-Tx state T_{TX}, is 5078 ms.

Equation B.3.11 gives the percentage of time that the node will remain in the receive state R_{RX} per an hour if it is continuously receiving and passing messages.

$$\begin{aligned} R_{RX} &= \frac{T_{RX}}{T_{RX} + T_{TX}} \\ &= \frac{6383.6\text{ms}}{6383.6\text{ms} + 5078\text{ms}} \\ &= 55.7\% \end{aligned} \tag{B.3.11}$$

The node will therefore spend 55.7% of the time in the RF-Rx state, and the remaining 44.3% in RF-Tx state. Equation B.3.12 calculates the current (I_{XMESH}) required by the node in this power profile, by multiplying the percentage of time the node spends in each state by the current required during that state.

$$\begin{aligned} I_{XMESH} &= R_{RX} \times I_{RX} + R_{TX} \times I_{TX} \\ &= \frac{55.7}{100} \times 15.36 + \frac{44.3}{100} \times 123.26 \\ &= 63.16\text{mA} \end{aligned} \tag{B.3.12}$$

In the extreme mesh power profile, the average current used by the node is 63.16 mA, while the power usage is calculated using Equation B.3.7 to be 208.43 mW.

B.3.4 Power Profile: Extreme Server

The extreme server profile describes a server which is receiving messages continuously and uploads the data from those messages to an online database every 2 minutes. A GSM upload takes 10 s and the total time per an hour which the node uploads via GSM is calculated using Equation B.3.13.

$$\begin{aligned} T_{GSM} &= \frac{10\text{s} \times 3600\text{s}}{2 \times 60\text{s}} \\ &= 300\text{s} \end{aligned} \tag{B.3.13}$$

Therefore, the node uploads via the GSM for a total of 300 s per an hour. Using Table B.4, the time which the server node will be in RF-Tx state T_{TX} while receiving a message is 1426 ms and the time spent in RF-Rx state T_{RX} per a message is 4566 ms. Equation B.3.14 can then be used to calculate the number messages received by the server node in an hour.

$$\begin{aligned} Messages &= \frac{Hour - T_{GSM}}{T_{TX} + T_{RX}} \\ &= \frac{3600 - 300}{1.426 + 4.566} \\ &= 550.73 \text{ messages} \end{aligned} \tag{B.3.14}$$

If the server node receives messages continuously, it will receive 550 messages. For these calculations, we assume that the 0.73 of a message calculated in Equation B.3.14 will be spent by the node in the RF-Rx state. Equation B.3.15 calculates the percentage of time the server node is in the RF-Tx state, and Equation B.3.16 calculates the percentage of time the server node is in the RF-Rx state.

$$\begin{aligned} R_{TX} &= \frac{1.426\text{s} \times 550}{3600\text{s}} \\ &= 21.79\% \end{aligned} \tag{B.3.15}$$

$$\begin{aligned} R_{RX} &= 100 - R_{TX} - R_{GSM} \\ &= 69.88\% \end{aligned} \tag{B.3.16}$$

The server node will be in the RF-Tx state for 21.79%, while 69.88% is spent in the RF-Rx state. Equation B.3.17 calculates the average current required by the node in the extreme server profile ($I_{XSERVER}$), by multiplying the percentage of time the node spends in each state by the current required during that state.

$$\begin{aligned} I_{XSERVER} &= R_{RX} \times I_{RX} + R_{TX} \times I_{TX} + R_{GSM} \times I_{GSM} \\ &= \frac{69.88}{100} \times 15.36 + \frac{21.79}{100} \times 123.26 + \frac{8.33}{100} \times 233 \\ &= 57\text{mA} \end{aligned} \tag{B.3.17}$$

The current used by the server node when in the extreme server power profile is 57 mA and the power required by a server node in this power profile is 188.1 mW.

B.3.5 Power Profile: Summary

The power profiles and their respective current and power usages are summarised in Table B.5.

Table B.5: Summary of the current and power used by the nodes in each power profile.

Power Profile	Current [mA]	Power [mW]
Normal Mesh	19.13	63.13
Normal Server	21.9	72.7
Extreme Mesh	63.16	208.43
Extreme Server	57	188.1

The Extreme mesh power profile has the highest power usage of all the profiles, requiring a current of 63.16 mA and a power of 208.43 mW. This power profile will be used as a worst case scenario during power electronics related calculations.

Appendix C

Battery Charging and Discharging graphs

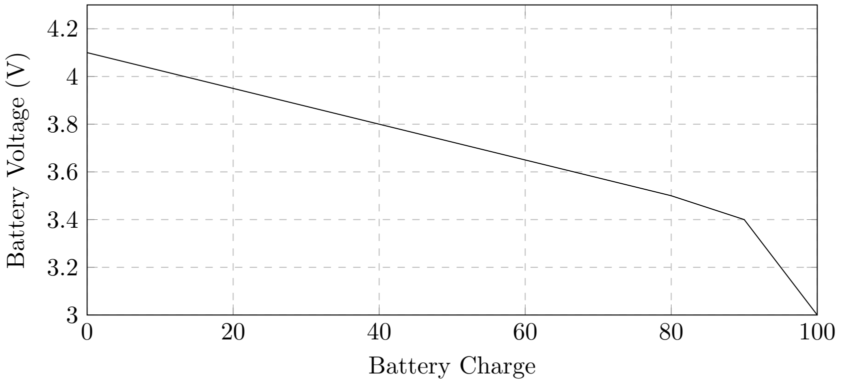


Figure C.1: Approximated battery charge to voltage curve. This curve was used to calculate battery voltage from charge. The curve makes use of 3 lines. For a battery discharge of between 0% and 80%, the line is $Voltage = -(3/400) * Discharge + 4.1$; For a battery discharge of between 80% and 90%, the line is $Voltage = -(1/100) * Discharge + 4.7$; and for a battery discharge of between 90& and 100%, the line is $Voltage = -(1/25) * Discharge + 7$

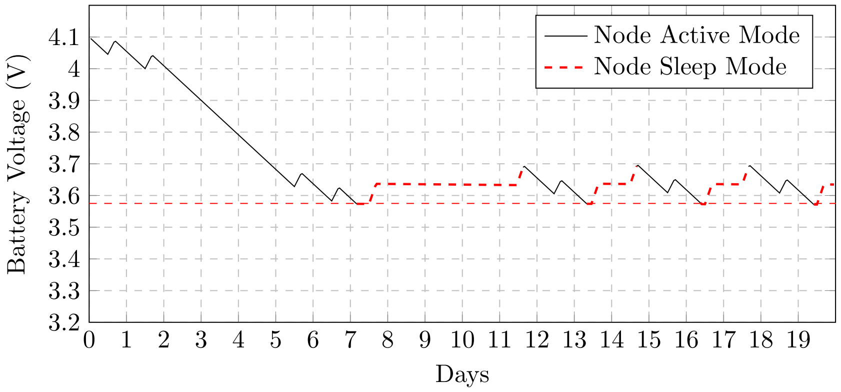


Figure C.2: Simulated battery voltage over a 20 day period with a 1 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

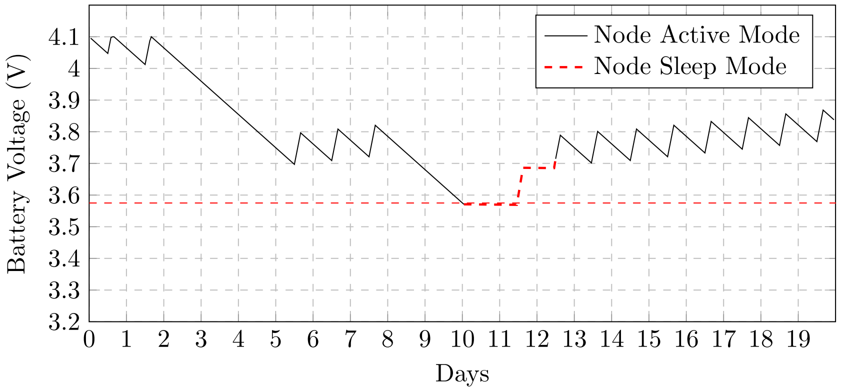


Figure C.3: Simulated battery voltage over a 20 day period with a 2 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

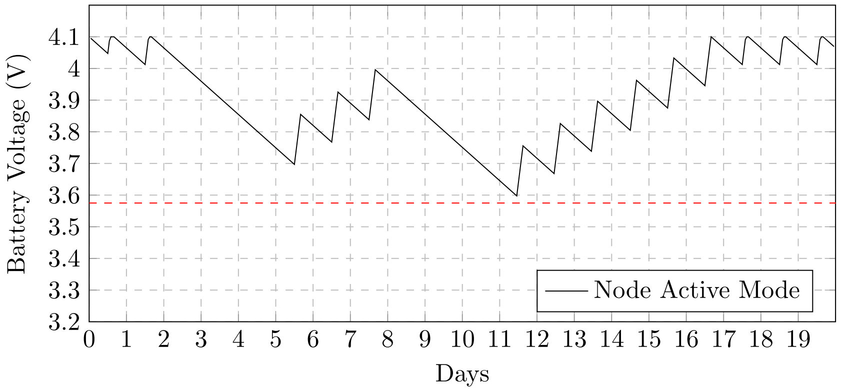


Figure C.4: Simulated battery voltage over a 20 day period with a 3 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

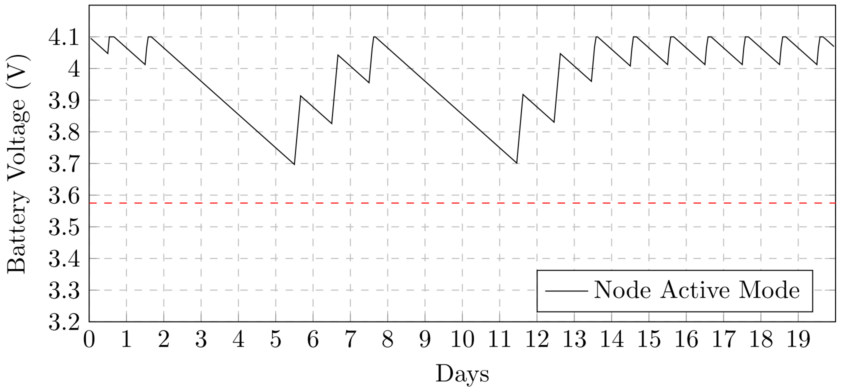


Figure C.5: Simulated battery voltage over a 20 day period with a 4 W solar panel connected to the charger. The load corresponds to a WSN node in the extreme mesh power profile.

Appendix D

Source Code and Hardware Design

To view the source code and hardware design including the design of the PCBs please visit the following link:

https://goo.gl/MBu5uP