Batteryless Sensor System for Breakage Detection in Electric Fence

T.M.A. Dabare

Batteryless Sensor System for Breakage Detection in Electric Fence

T.M.A. Dabare
Index No: 14000148

Supervisor: Dr. Kasun de Zoysa

December 2018

Submitted in partial fulfillment of the requirements of the
B.Sc in Computer Science Final Year Project (SCS4124)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: T.M.A. Dabare

………………………………………………

Signature of Candidate Date:

This is to certify that this dissertation is based on the work of Mr. T.M.A. Dabare under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor's Name: Dr. Kasun de Zoysa

………………………………………………

Signature of Supervisor Date:

This is to certify that this dissertation is based on the work of Mr. T.M.A. Dabare under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Co-Supervisor’s Name: Mr. K.V.D.J. Prabhash Kumarasinghe

………………………………………………

Signature of Co-Supervisor Date:

Abstract

The human-elephant conflict is an ongoing tragedy in the everyday life of rural villagers and farmers in many parts of Sri Lanka. Approximately 80 people and 200 elephants die every year due to the fights between them, when elephants enter the farm lands and human habitats in search of food and water [10]. While many illegal methods are in use to prevent elephants from entering human habitats, such as poisoning and shooting them, the only viable solution that has been tried so far is electric fencing. Maintaining these fences is not an easy task. Locating breakages in these fences is a critical challenge. Electric fences installed in Sri Lanka are 2-40 km in length [2]. At present, breakages are located by walking along the fence and inspecting it. This requires a considerable amount of manpower and as a consequence, the breakages are occasionally left unattended, which allows the elephants to cross over and roam into nearby villages.

In this dissertation, a novel approach is proposed to address the problem of locating breakages in electric fences. For this, cheap batteryless nodes will be setup along the fence which detect the breakages in fence and communicate wirelessly to a base station. This focuses on two main aspects, one is to make batteryless energy efficient low cost nodes and second is to design an energy efficient wireless communication protocol to be installed in the designed nodes.

With this novel idea we can detect breakages in electric fences automatically without man-power. Wildlife officers fixes breakages during day-time since night-time is dangerous. Therefore if a breakage happens in the electric fence it is detected and the location of the breakage will be reported to the authority within a day and wildlife officers can look into that quickly.

Preface

A novel approach to address the problem of locating breakages in electric fences is introduced in this dissertation. For this cheap batteryless nodes will be setup along the fence which detect the breakages in fence and communicate wirelessly to a base station. This focuses on two main aspects, one is to make batteryless nodes and the second is to communicate reliably in between nodes. I propose to make batteryless energy efficient nodes, and to design a wireless energy efficient multihop communication protocol which will be installed in the designed nodes. For this I modify the existing gossip protocol and T-MAC protocol. These two were developed separately in the beginning. And in the final phase of the research I integrate the both and made a single system. The testing was carried out by me. I used an off the shelf energizer to power the fence.

Acknowledgement

I dedicate thesis to my loving family who has been an immense support to me throughout this journey of life. I appreciate the feedback and motivation provided by my friends to achieve my research goals. It is a great pleasure for me to acknowledge the assistance and contribution of all the people who helped me to successfully complete my research.

Personal Statement

Nature is the hidden power of mankind. The man is highly dependable on nature that we cannot live without it. However today we have been misled to believe that technology is the most important thing of our lives; that we have to grab every chance that we have with technology. But we have forgotten that the essence of all living beings is nature. We take very much from nature and instead of giving back, we destroy it. Sometimes, we see the damage we have done, yet we do not take the responsibility, because we are not so called "environmentalists" or "ecologists". I think that, whatever the field you are in, whatever the job you do, even if it is not related to nature, there is a responsibility for each and every one of us to take care of nature as much as we can. And if we all can contribute to save the nature with the knowledge we have gained from all the years of learning and the experiences, it is my opinion that it will be one step forward in protecting nature.

Through this research, I will be able to put the knowledge, which I have gained during my undergraduate years, into reducing the human elephant conflict by making the existing electric fence stable, reliable and robust. I am very much honored to be able to conduct this research personally, because I hope that, by doing this research I will be able to save both lives of elephants and human. And if I succeed in this research, the findings can be used in other countries too. This is my chance to give back to the nature for what I have been receiving from it.

Table of Contents

List of Figures

List of Tables

List of Acronyms

IOT	Internet Of Things
MAC	Media Access Control
WSN	Wireless Sensor Network
I/O	Input/Output
LDR	Light Dependent Resistor
IDE	Integrated Development Environment
EEPROM	Electrically Erasable Programmable Read-Only Memory
GUI	Graphical User Interface

Chapter 1 - Introduction

The widely used method identified so far to reduce human-elephant conflict is electric fencing. These fences break due to many reasons, and locating these breakages is a tedious task. As solutions there are many systems designed to locate these breakages, but most of them are not reliable while some costs lot. Almost all of them have high maintenance cost because they use batteries in their systems. Therefore, through this research addresses this gap and thereby contribute to reduce the human-elephant conflict by making a battery less reliable energy efficient electric fence breakage detection system.

1.1 Background to the Research

The human-elephant conflict is an ongoing tragedy in the everyday life of rural villagers and farmers in many parts of Sri Lanka. Approximately 80 people and 200 elephants die every year due to the fights between them, when elephants enter the farm lands and human habitats in search of food and water [10]. While many illegal methods are in use to prevent elephants from entering human habitats, such as poisoning and shooting them, the only viable solution that has been tried so far is electric fencing.

People successfully use electric fences to protect human habitats and farm lands from elephants in various parts of the world [1, 2]. Fences deployed in Sri Lanka are 2-40 km in length [3]. In Sri Lanka, over 2500 km of electric fences are deployed in affected areas with support of the government and nongovernment organizations [3]. The total area covered by electric fences can be more than that since various communities and individuals build electric fences by themselves to protect their premises from elephants without the support of the government.

1.2 Research Problem

Building an electric fence is not an easy task. The financial cost associated with building an electric fence mostly consists of buying an energizer, electric wires long enough to cover the targeted area and wooden or concrete posts as a support to the electric wires. Additionally, human labor is necessary to lay the wires and to build a housing for the energizer. Also maintaining these fences is not an easy task. Locating breakages in these fences is a critical challenge faced by the rangers of the wildlife parks. Breakages in the fence are usually caused by several factors that include falling branches, breaching by animals such as buffaloes and elephants, and also occasional interference from human activity. For example, villagers sometimes cut the electric fence to allow their cattle into the wildlife sanctuary to feed them. At present, breakages are located by walking along the fence and inspecting it. This requires a considerable amount of manpower and as a consequence, the breakages are occasionally left unattended, which allows the elephants to cross over and roam into nearby villages. Also these breakages can be fixed during the day time because during night time, going near the forest is dangerous. So breakages should be detected during daytime. There are electric fences with breakage detection functionality, but these systems need batteries to power their nodes which is set up along the fence. These systems [7, 8, 4] are suitable for short fences, but for electric fences which uses to cover a village, these systems are not suitable because when the batteries die they should be replaced. This is very costly and time consuming. Even we use rechargeable batteries, after sometime they also will stop working, again we have to replace all the batteries. Because of these problems there is a big need for a batteryless electric fence breakage detection system. Due to these reasons, many human-elephant conflict affected areas still lack reliable electric fences. Figure 1.1 represent a high level diagram of an electric fence.

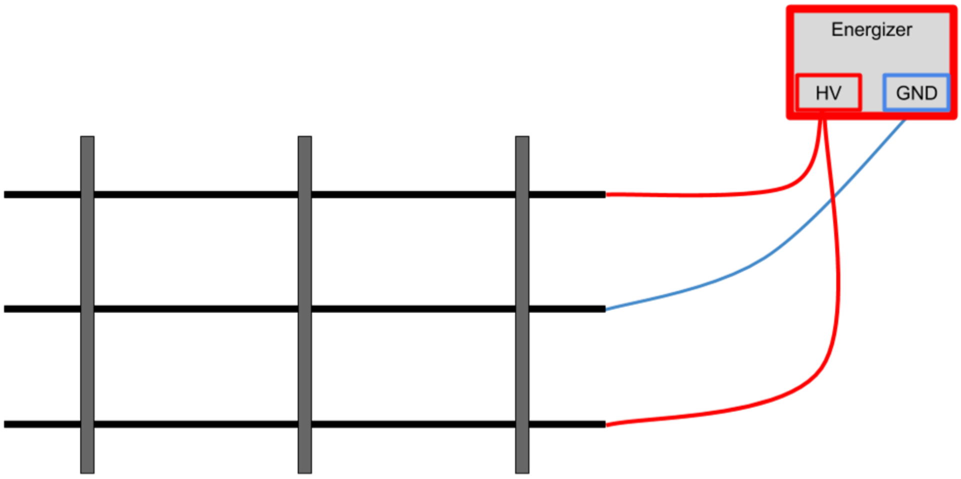


Figure 1.1: High level architecture of an electric fence

1.3 Significance of the Problem

When a fence is down, a person has to walk along the fence to find the broken location. Therefore identifying breakages need a big labor cost and it is time consuming. These fences are repaired during the day time. The present breakage detection systems [7, 8, 4] need costly maintenance because their nodes have batteries to power the nodes up. Therefore this research will help for authorities to find the exact location where the fence is broken and thus reduce the labor cost and time. Also the proposed system is batteryless hence the maintenance cost of the system is very low.

It will contribute to WSN by designing an energy efficient multihop wireless communication protocol.

1.4 Research Questions

- How to design an energy efficient batteryless sensor system for breakage detection in electric fence?

1.5 Aims and Objectives

The aim is to reduce the duration at which the fence is down, by sending the exact location where the fence is damaged, to the relevant authorities, and speeding up the process of search and repair, thus reducing the danger of elephants getting through the fence. The maintenance cost of the fence needs to be reduced by making batteryless nodes.

Objective of this research is to design a reliable robust less maintenance batteryless electric fence breakage detection system.

1.6 Justification for the Research

Locating breakages in electrical fences is a critical task. Currently there is a solution proposed by the [4] where they setup cheap nodes with sensors along the fence thereby they can find the broken location of the fence. But there are three drawbacks in this system which makes it unreliable, first during the charging mode elephants can enter through the fence. Second the communication protocol designed for it is not reliable. Third the nodes are not energy efficient which drains the battery fast and this system uses rechargeable batteries which will stop working after some time and causing costly maintenance.

Therefore this research will focus on proposing and evaluating a batteryless electric fence breakage detection system.

1.7 Methodology

At the first step of the research a literature review is done to find the most suitable WSN communication protocol for inter communication between nodes. If a protocol could not be found a new protocol will be designed. In next step batteryless sensor nodes will be designed. Finally the designed communication protocol will be implemented in the designed batteryless nodes and will be evaluated.

1.8 Outline of the Dissertation

The dissertation is structured as follows. Chapter two explores the existing approaches related to this research. It gives a brief explanation of the existing electric fence breakage detection systems and existing WSN communication protocols. Chapter three describes the proposed research design and methodology. Potential ways of addressing the research problem is discussed in this chapter. Chapter four demonstrates the implementation details of the proposed methodology. Chapter five presents the evaluation of the proposed system and the evaluation results of the proposed approaches. The last chapter, chapter six demonstrates the conclusion of the thesis and outlines the future work.

1.9 Delimitations of Scope

In this research a batteryless reliable robust less maintenance electric fence breakage system and an energy efficient wireless communication protocol to communicate between the sensor nodes of the system will be designed. The assumption and the limitation, of this research is stated below.

The assumptions would be,

- The fence would mainly consist of 3 wires, the top and the bottom wires would be live wires whereas the middle wire would be the ground wire as shown in figure 1.1.
- The fence is located in sunny area where the nodes are uncovered from sunlight after setting up to the fence posts.
- Information about the status of the fence is needed in the day-time for the authorities to repair the fence. Because during night-time repairing is not done.

The limitations would be,

- The proposed protocol can have only maximum of 240 nodes.

1.10 Conclusion

This chapter laid the foundations for the dissertation. It introduced the research problem and research questions and hypotheses. Then the research was justified, definitions were presented, the methodology was briefly described and justified, the dissertation was outlined, and the limitations were given. On these foundations, the dissertation can proceed with a detailed description of the research.

Chapter 2 - Literature Review

2.1 Introduction

In this chapter, a review of related work on electric fence breakage detection systems and wireless MAC protocols is presented.

2.2 Electric Fence Breakage Detection Systems

Electric fence breakage detection systems are systems which use to detect breakages in electric fences automatically.

2.2.1 Fence Alarm System

Eastwood has invented the fence alarm system [5] which comprises an alarm transmission element and an alarm receiving element. The alarm transmission element detects the breakages in the fence and transmits a unique wireless signal to the alarm receiving element and thereby capable of detecting the breakage location. This method is suitable for small areas because the transmitter elements communicate directly with the receiver elements so they should be in the same range. For long range, expensive transmitters should be used. Figure 2.1 shows the high level architecture of the fence alarm system.

2.2.2 'eleAlert' System

The system described in [6] uses a separate communication line running alongside the fence wire, to detect faults. The system identifies the breakages in the communication line rather than the wire itself. This system consist of a RTU (Remote Transmitting Unit), Bridges separating the fence into segments and a number of LITs (Location Identification Tags) within each segment. The RTU is the gateway between the system and the mobile network. The devices are connected by a pair of thin copper wires, which perform multiple functions of intrusion detection, power distribution and communication. Figure 2.2 shows the high level architecture of the eleAlert system.

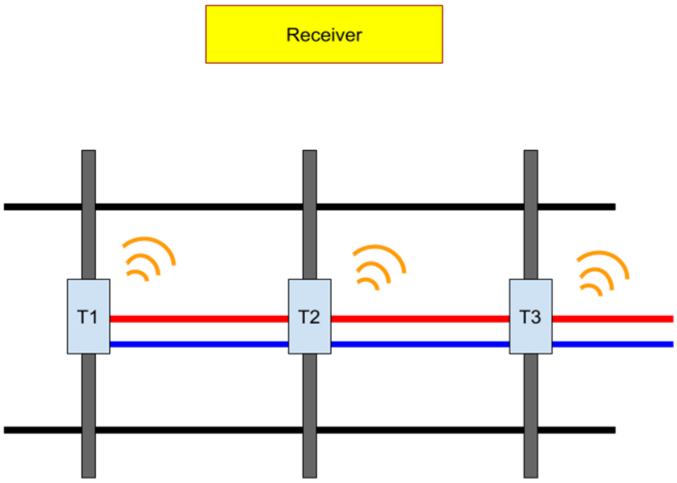


Figure 2.1: High level architecture of fence alarm system

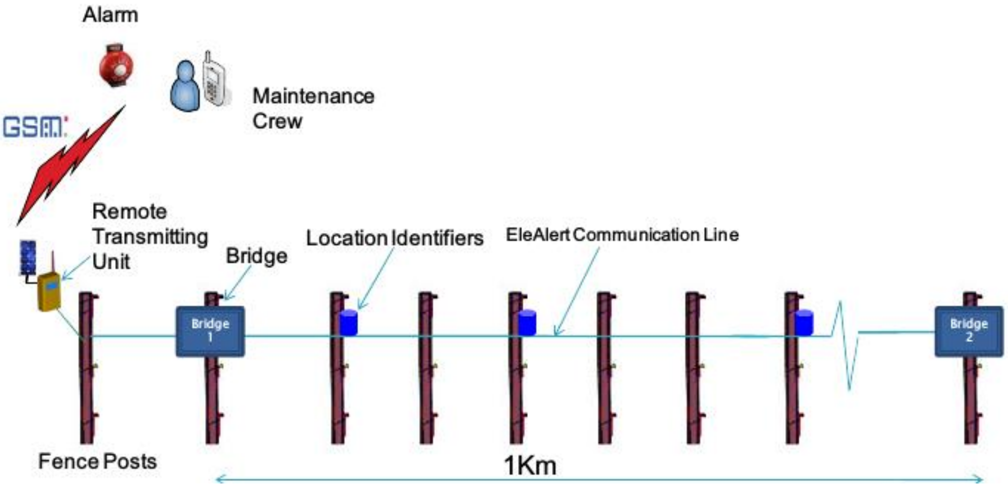


Figure 2.2: High level architecture of eleAlert system

2.2.3 Sensor-based Breakage Detection for Electric Fences

The system described in [7], has presented a low-cost electric fence that is able to communicate through the fence wire itself as the communication medium. It consists of two modes: the fence pulse mode and the communication pulse mode which the fence periodically switching between the two. During the fence pulse mode, the fence operates with high-voltage pulses in order to deter elephants from the covered area. The communication pulse mode uses low-voltage pulses where a central controller sends node IDs through the fence wire to nodes installed in wooden posts at different locations of the fence. The reception of a node ID indicates that the fence is working up to the node’s location. Figure 2.3 shows the high level architecture of the Sensor-based breakage detection for electric fences system.

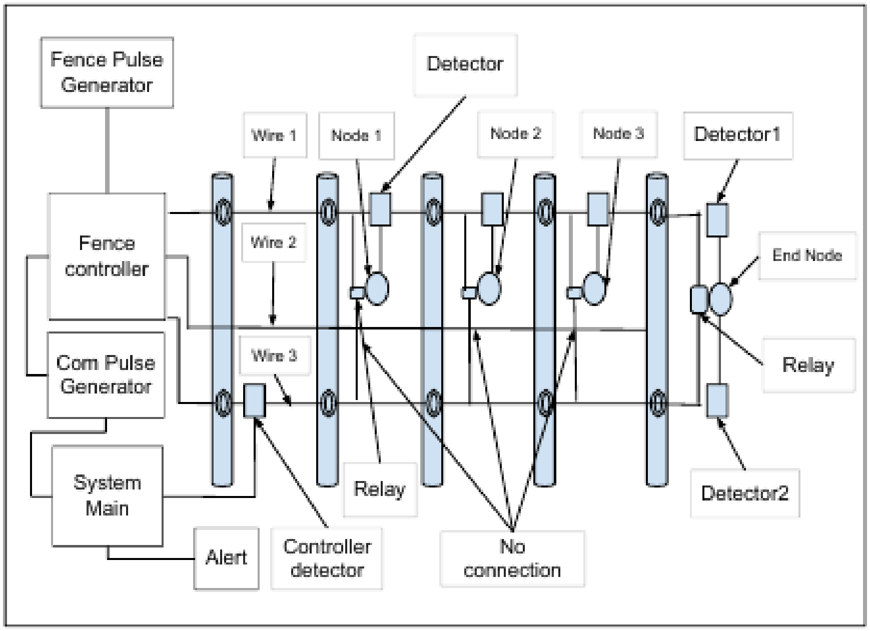


Figure 2.3: High level architecture of Sensor-based breakage detection for electric fences system

2.2.4 Time-domain Reflectometry Method

Time-domain reflectometry is a method that can be used for detecting breakages in conductors such as coaxial and twisted pair cables [9]. In order to use this methods, the cable should be properly terminated at the end points and should have a uniform impedance. The wires used in low cost electric fences does not fulfill these requirements so this method cannot be used to detect breakages in electric fences.

2.2.5 Wire is Not Dead

In this approach [8] the system consists of three main components (energizer, pulse detector and fence nodes). The energizer continuously generates high-voltage pulses. The pulse detector is connected to the fence wire in the same end of the fence where the energizer is located. It is designed to count the absence of pulses on fence wire. The fence nodes are placed along the fence wire on wooden posts. Nodes short the fence when high voltage is transmitting and the pulse detector listen to number of pulses shorted and say up to which node is working. Figure 2.4 shows the high level architecture of the wire is not dead system.

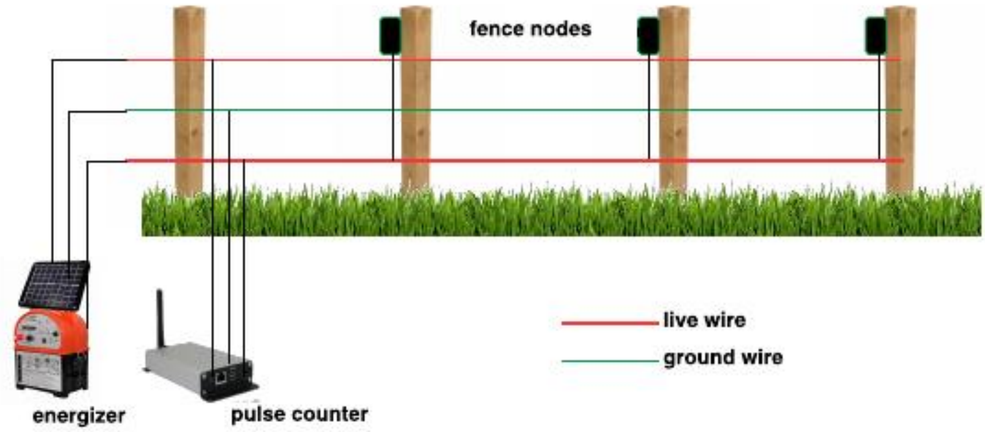


Figure 2.4: High level architecture of wire is not dead system

2.2.6 Sensor Based Fence Breakage Detection System

In this system [4] there are nodes and a controller. Nodes are placed along the fence wire on wooden posts. The nodes are connected to the live lines in the fence. Nodes are charged using this wire. While the fence is in high voltage mode, nodes collect the neon status (used to check ground line breakage) and when the fence is changed to DC mode, the communication process begins. The numbering of nodes is done from the controller side. All the nodes are given a waiting time, in the waiting time nodes are put to receive mode and they are listening whether there is a packet sent by the next node.

Waiting time for a node is defined by waitTime = (totalNodes - nodeID) * 10.

Figure 2.5 shows the high level architecture of the sensor based fence breakage detection system.

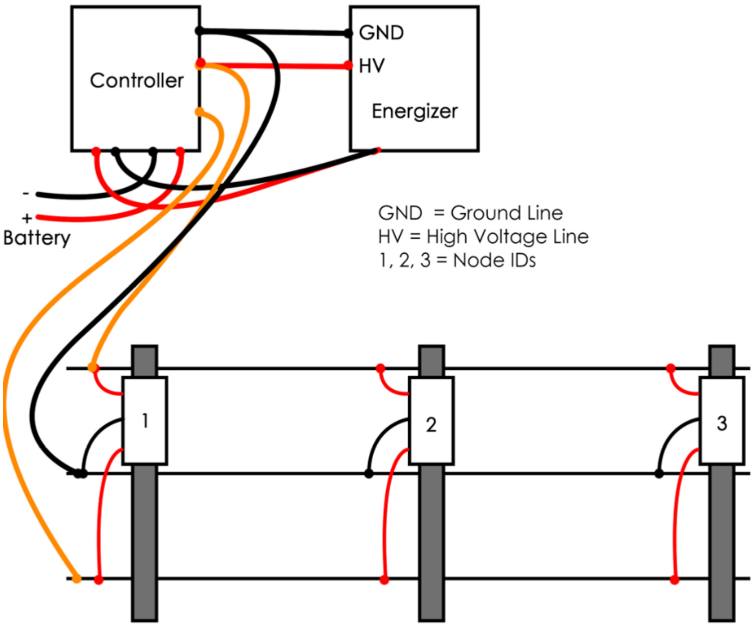


Figure 2.5: High level architecture of Sensor based fence breakage detection system

2.3 Wireless MAC Protocols

2.3.1 Gossip

Gossip is a protocol that lets large number of nodes exchange key value data with no central management. Each node gossip with a random small subset of the nodes, updating of its knowledge on the key/value data of other nodes [14].

Figure 2.6 shows the high level architecture of gossip protocol.

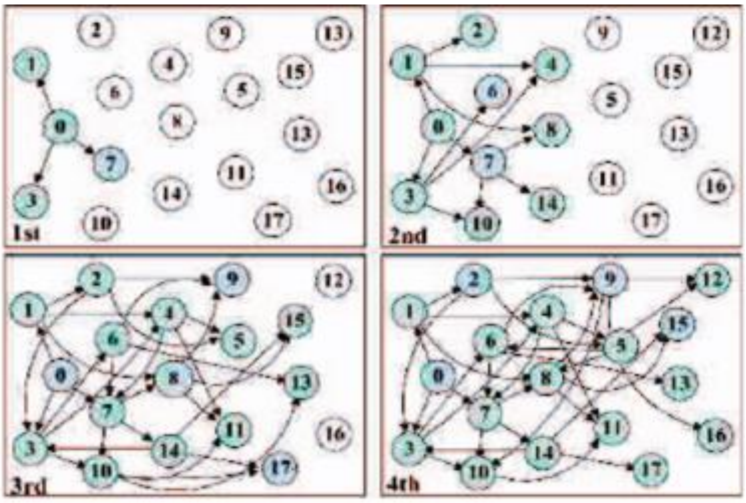


Figure 2.6: High level architecture of gossip protocol

2.3.2 Survey of MAC Protocol for Wireless Sensor Networks

In [11] , They present a survey on some of the popular MAC layer protocols. It also provides a brief analysis of these protocols which could be helpful in future work in this direction. This paper also provides a reference for further research in this area giving an insight on energy conservation at MAC layer.

2.3.3 The Evolution of MAC Protocols in WSNs

This [12] details the evolution of WSN MAC protocols in four categories: asynchronous, synchronous, frame-slotted, and multichannel. These designs are evaluated in terms of energy efficiency, data delivery performance, and overhead needed to maintain a protocol's mechanisms. With extensive analysis of the protocols many future directions are stated at the end of this survey. The performance of different classes of protocols could be substantially improved in future designs by taking into consideration the recent advances in technologies and application demands.

2.3.4 S-MAC

Sensor-MAC [13] (S-MAC). Basic concept of S-MAC is periodic sleep listen schedules which are handled locally by the sensor network. Nodes which are adjacent form clusters virtually and they share common schedule. This means that if two nodes are side by side and fall in two different clusters they wake up at listen schedule of both clusters. This also results in more energy consumption as nodes wake up to two different schedules. The schedules are also needed to be communicated to different nodes of virtual cluster which is accomplished by SYNC packets and time in which it is sent is known as synchronization period. Carrier sense helps in collision avoidance. In addition to it, unicast data packets transmission is done using RTS/CTS. A new and innovative feature of S-MAC is message passing through which a long message is sent in burst by dividing it into small messages. This helps in energy saving by using common overhead. However, this concept of sleeping schedule may also result in high delay termed as latency which will be significant in case of multi-hop routing algorithms, as each node in between will have their own sleep schedules. This is known as sleep delay. Figure 2.7 shows the high level architecture of S-MAC protocol.

2.3.5 T-MAC

T-MAC [15] (Timeout MAC) is the protocol which is derived from S-MAC protocol in which the non-sleep and sleep periods are fixed. In T-MAC the sensor node deviates to sleep period if no event has occurred for a time ‘Tact’.

There are many such events like data receiving, start of listen or sleep period etc. Minimum idle listening period is the time ‘Tact’. The interval Ta is greater than sum of the contention time, length of an RTS packet, turnaround time and the length of the CTS packet. This whole scenario results in energy consumption which less in TMAC as compared to Sensor S-MAC protocol. However, this is adjusted against high delay or latency which T-MAC protocol has as compared to the S-MAC protocol.

Advantages Of this is, T-MAC can easily handle variable load due to dynamic sleeping schedule. Disadvantages of this is, T-MAC's major disadvantage is early sleeping problem in which nodes may sleep as per their activation time and data may get lost especially for long messages.

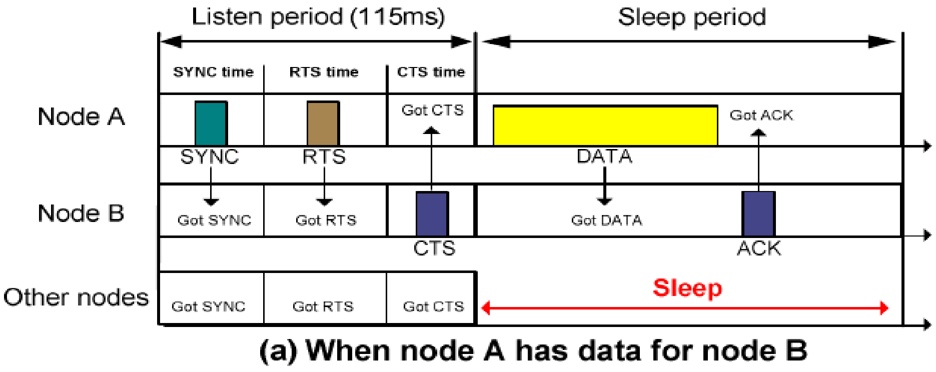


Figure 2.7: High level architecture of S-MAC protocol

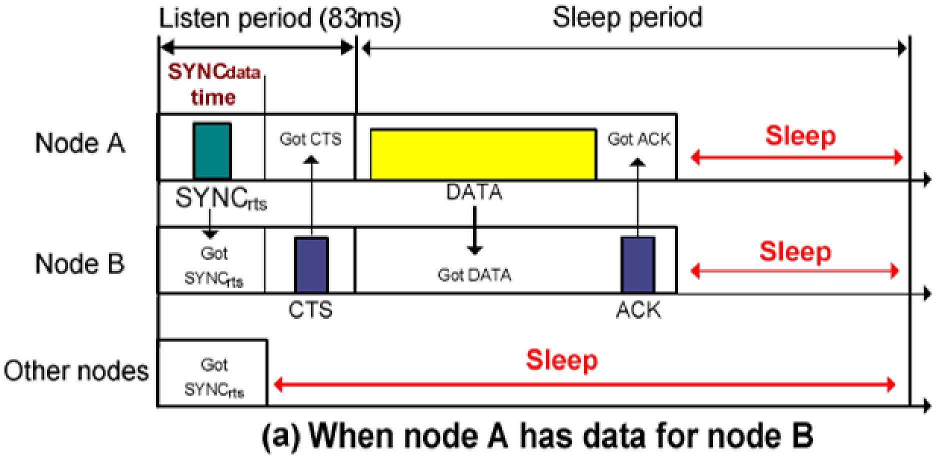


Figure 2.8: High level architecture of TEEM protocol

2.3.6 TEEM

TEEM (Traffic aware Energy Efficient MAC) protocol [16] is a variation of S-MAC. Unlike S-MAC, which uses fixed duty cycle it uses an adaptive duty cycle which utilizes the traffic information of each node. Reduction in listen time is achieved by making the node sleep prior to the specified time when there is no data traffic expected. TEEM modifies its structure in two ways, first it makes all nodes switch off their radios when no data packets is expected in network, and second is, it uses separate RTS packet which is different from traditional RTS. TEEM uses a small listen interval which is divided into two parts, the first part is for sending SYNC data packets which is for the nodes who have any data for transfer in their queue. Next part is for sending SYNC no data packets, for those nodes who have no data packets to transmit in its buffer. It uses an approach of piggybacking a RTS packet onto its SYNC packet, which reduces the listening time. It is feasible as well as efficient to combine these two control packets as naturally if a node wins the channel for sending its SYNC packet in the SYNC data period then it means it will definitely send its data messages. This combination of SYNC and RTS is called a SYNC/RTS packet. TEEM is a very efficient protocol that makes sleep and listen durations adaptive. Traffic information of each node helps in decreasing the power consumption significantly. The listen time of nodes reduces a lot by putting them into sleep state earlier when there is no probability of traffic. Figure 2.8 shows the high level architecture of TEEM protocol.

2.3.7 Wise-MAC

TDMA and CSMA with preamble sampling protocol is presented in [17] where sensor nodes have two communication, channels one data channel accessed via TDMA method and one control channel accessed via CSMA method. Wise-MAC protocol [18] utilizes preamble sampling approach to reduce idle listening. Here a preamble is present in the beginning of each data packet for alerting the receiving node. Nodes present in the network, samples the medium with a common period but their relative schedule offsets are independent. If after waking and sampling the medium a node finds it busy, it continues to listen until a packet is received or the medium becomes idle again. Initially the length of preamble is equal to the sampling period. To decrease the power consumption due to the fixed preamble length, Wise-MAC offers a method of dynamically determining the length of the preamble. It does this by utilizing the information of the sleep schedules of the transmitting nodes immediate neighbors. The neighbor’s schedule is learned and updated in every data exchange as a part of acknowledgement. In this way, a table is maintained at every node which has the sleep schedules of its neighbors. Depending on this table, Wise-MAC schedules transmissions. Wise-MAC's performance is better than the S-MAC and other similar protocols. The alteration in preamble length dynamically provides better performance under variable traffic loads. Decentralized sleep-scheduling is the major drawback of Wise-MAC, which results in each neighbor of node acquiring different sleep and wake-up times. This issue becomes significant in broadcasted communication, where the broadcasted packet is buffered for neighbors in sleep mode and delivered multiple times as each neighbor wakes up. In addition to this hidden terminal problem is also a considerable issue in Wise-MAC.

2.3.8 EM-MAC

EM-MAC (Efficient Multichannel MAC) [19] belongs to the category of asynchronous duty cycled MAC protocol which utilizes the available multiple radio channels. In contrast with existing multi-channel energy MAC protocols, EM-MAC dynamically selects the channels it switches among for receiving. EM-MAC avoids the use of heavily loaded individual channels by effectively exploiting various orthogonal channels. This high energy efficiency is achieved by accurately predicting the wake-up channel and wake-up time of the receiving node. Every time a node wakes up, it randomly selects its wake-up time and channel according to pseudorandom function, avoiding channels having heavy traffic loads which is undesirable (which it has detected high). A sender has knowledge of receiver's pseudorandom function which it uses to predict the receiver's wakeup channel and wake up time for communicating with that particular receiver.

EM-MAC is an asynchronous, multi-channel protocol that is receiver initiated. Major disadvantage of this protocol is that the periodic transmission of beacon may create additional interference when there is little or no traffic.

2.3.9 BEAM

The BEAM (Burst-aware Energy-efficient Adaptive MAC) [20] optimizes the sleep time of the receiver by using two modes of operation. These modes are based on the payload size of the intended data packet. An adaptation algorithm is also present for the adaption of listen and transmission cycles according to the load present in network. Its basic operation consists of a sender transmitting periodically short preamble frames including the payload with destination address given at the start of every preamble. After an intended receiver acknowledges a data frame by data ACK packet, sender stops sending the frame. This mode has two transmissions for a payload. It is useful when the size of payload is very small. On the other hand there is second mode which involves short preamble without payload. Here the sender periodically transmits a series of short preamble frame as an indication of a possible transmission. Destination address stored at the start of preambles helps the nodes to identify whether the transmission is intended for them or not. As a node wakes up it senses the channel and on hearing the preamble it checks its address part to see if it is an intended receiver or not. After this it starts sending an early ACK frame to indicate the sender to stop the preamble transmission. An early ACK frame indicates the transmitting node that intended receiver is awake, and it can now transmit data frame. On a successful transmission receiver sends an ACK frame. BEAM switches itself between the modes based on the size of the payload. Figure 2.9 shows the high level architecture of BEAM protocol.

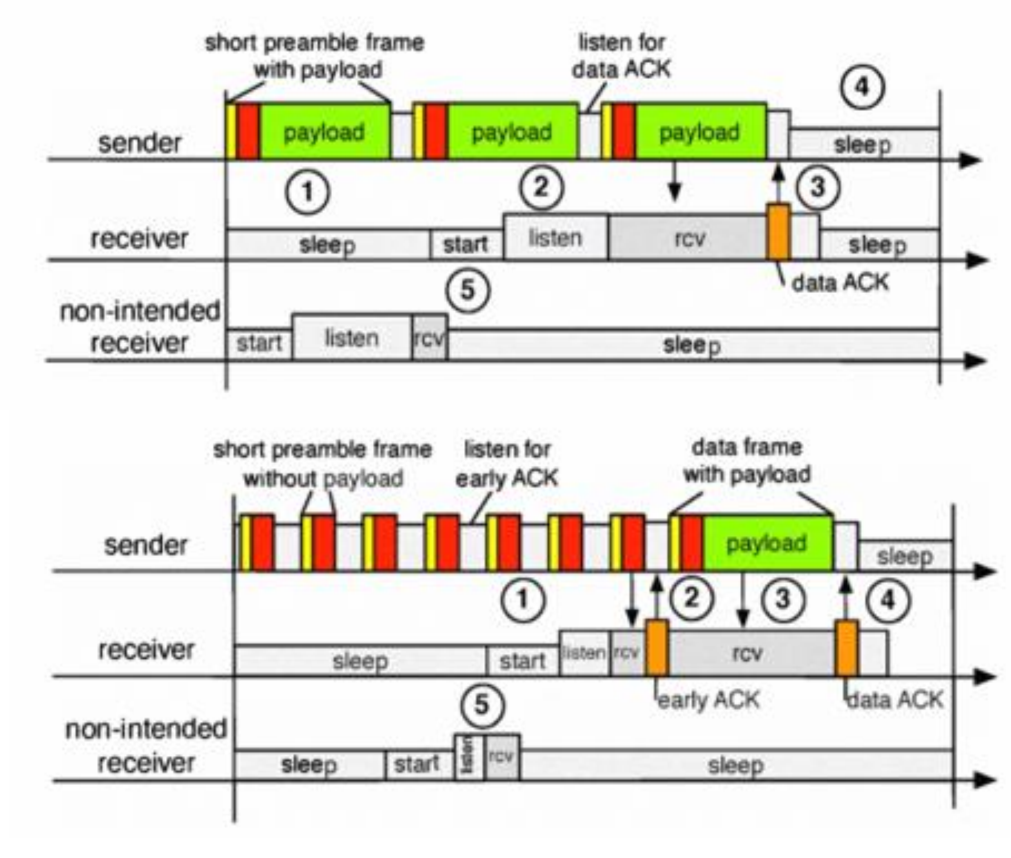


Figure 2.9: High level architecture of BEAM protocol

2.3.10 R-MAC

R-MAC (Reservation MAC) [21] protocol design concerns mainly a heavy loaded WSN in which there are many number of nodes. The main focus of this protocol is avoiding over hearing, so that immediate switching between sleep state and wake-up state of nodes is minimized which is considered a main cause of energy loss in such networks. According to the traffic load the listen-sleep duration is adjusted thus reducing collisions. Here a sensor node reserves the channel in advance for its next transmission which helps the neighboring nodes to know the involved transmitter/receiver for each time slot, thereby they can schedule their sleep and wake-up interval. This helps in avoiding over hearing of several transmissions and also reduces frequent switching between the two modes. This protocol divides the listen period into many different periods first one is a reservation period of duration R, a transmission period of N time slots, then there is sleep period and a synchronization period.

Design of R-MAC is favorable for the networks where the traffic conditions are very high, and where the main source of energy loss between the nodes is overhearing. It helps to reduce energy wastage due to over hearing and frequent switch between the sleep and listen modes by reserving time slots for transmission. R-MAC also modifies the duration of sleep of a node according to the traffic load present which is also one point where it excels than S-MAC or its variants. But the main concern is that it is only suitable for the applications which have heavy traffic and where the main source of energy wastage is the overhearing. The issue of idle listening which is a major energy loss factor in low data rate applications is not addressed by the R-MAC.

2.3.11 LPL Technique

LPL (Low Power Listening) technique [22] approach is simple, asynchronous and energy efficient. An accurate measure of whether the channel is active is a crucial component of a LPL-based MAC layer, and this may sometimes become a problem in environments like residential places or some other places where possibilities of other type of radiations is also there, in these environment LPL is susceptible to have false wake up problem that is a node may detect energy in environment but it may not be a channel activity and this increases the energy loss.

2.3.12 X-MAC

X-MAC [23] is a newer approach to LPL with shorter preamble and other modifications. X-MAC employs three strategy first approach is the inclusion of intended receiver's address in beginning of the preamble so that only intended receiver is awake and others can go to sleep, this avoids overhearing. This approach is also helpful in the case when more than one sender wants to transmit to one receiver. The other nodes sense the channel and if they sense the ACK packet of the receiver they wait, back off and then transmit their data after a random interval without any preamble as the receiver is awake. Receiver also waits for some time before going to sleep so that if any other sender is ready to send packet it could receive those data packets (note that there is no preamble attached in such packets as the receiver is already awake). Figure 2.10 shows the high level architecture of X-MAC protocol.

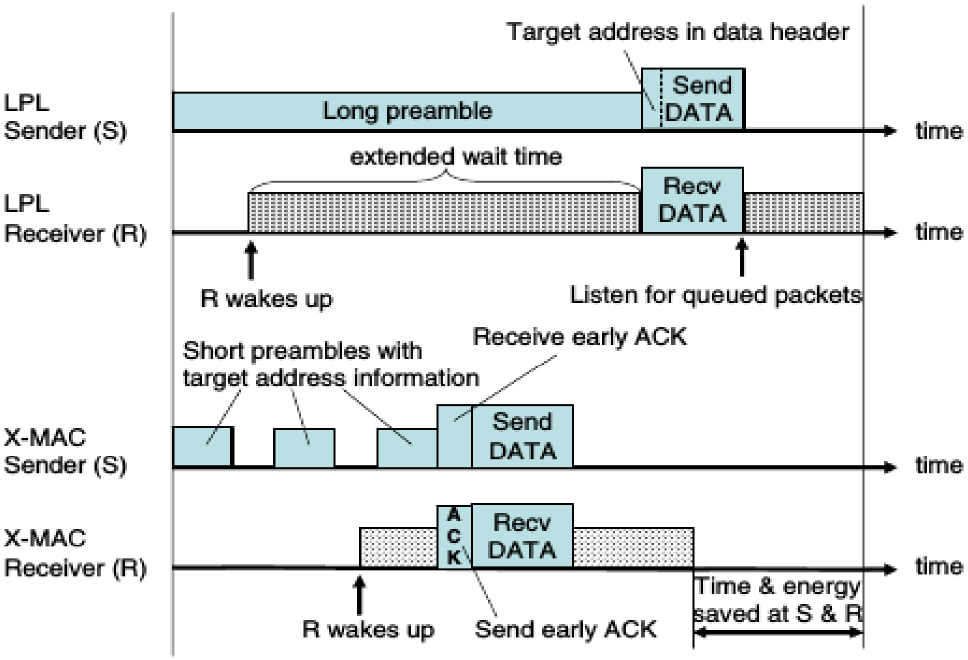


Figure 2.10: High level architecture of X-MAC protocol

2.4 Conclusion

This chapter is mainly focused on providing an extensive review on the existing electric fence breakage detection systems and WSN communication protocols. Initially, it discussed about the existing electric fence breakage detection systems and then about the WSN communication protocols with their advantages and disadvantages.

Chapter 3 - Design

3.1 Introduction

This chapter explicates the proposed solutions to the research problem. It consists of two main sections. Section 3.2 describes the research design of wireless communication protocol. Section 3.3 describes the research design of batteryless and energy efficient sensor node.

3.2 Research Design of Wireless Communication Protocol

The high-level architecture of designing the energy efficient, reliable wireless multihop communication protocol for intercommunication between nodes is given in figure 3.1.

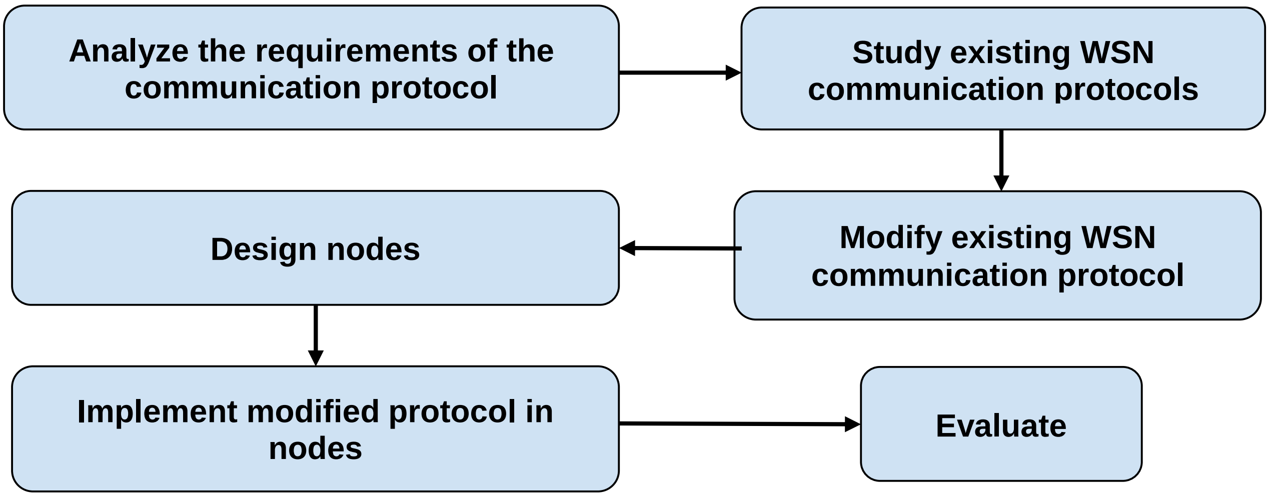

Analyze the requirements of the communication protocol
Study existing WSN communication protocols
Modify existing WSN communication protocol
Design nodes
Implement modified protocol in nodes
Evaluate

Figure 3.1: Research design of wireless communication protocol

First, the requirements of the wireless communication protocol were analyzed. The requirements are stated below:

- Energy efficient
- Multihop
- Reliable
- Simple
- Only need to send a single bit as the message
    - Message is the fence working status
    - Either yes or no
- Not time critical
    - Sufficient to know the fence breakage status within the day

Then a survey was done about existing WSN communication protocols. With the survey results a suitable WSN communication protocol could not founded which fulfills the above mentioned requirements. So we decided to design a new WSN communication protocol the 'Dead or Alive Protocol' by combining and modifying existing protocols to suite the above requirements.

For this, the existing Gossip protocol explained in section 2.3.1 and T-MAC protocol explained in section 2.3.5 was selected. The new WSN communication protocol is designed by combining some of their features with some extra features. The new wireless communication protocol works as same as the Gossip protocol with dynamic duty cycle feature by T-MAC protocol. The sleep time changes dynamically. When a node wakes up it broadcasts a message and then listen for some period and goes to sleep again. This repeats. There are more than one node in the same radio range of that node, this is called the *zone* of that node. The *zone radius* is defined with the number of nodes. In this 'Dead or Alive Protocol', sensor nodes should have an ID number, and the nodes should be arranged in an order. Figure 3.2 shows a row of nodes with zone radius 1 and figure 3.4 shows a row of nodes with zone radius 2. We propose to setup the sensor nodes for this research as in the figure 3.4. Since in case if a node fails, the whole system will not fail as shown in figure 3.5. Figure 3.3 shows how system fails when one node stops working.

Duty cycling feature was added to save the energy. This is an asynchronous protocol which all the nodes do not follow the same duty cycle, which means some nodes are listening while others are sleeping. With this non synchronized dynamic duty cycling technique, it is guaranteed that a node's broadcasting message is most probably received by some other node. The algorithm of the designed wireless communication protocol is stated below in pseudo code. Figure 3.6 shows the state diagram of the system.

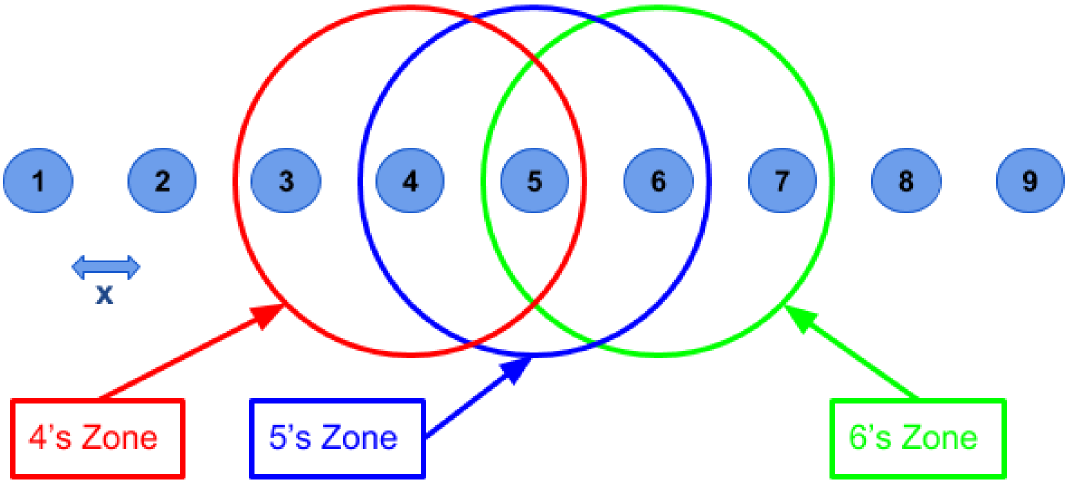


Figure 3.2: Nodes arranged in a row with zone radius 1

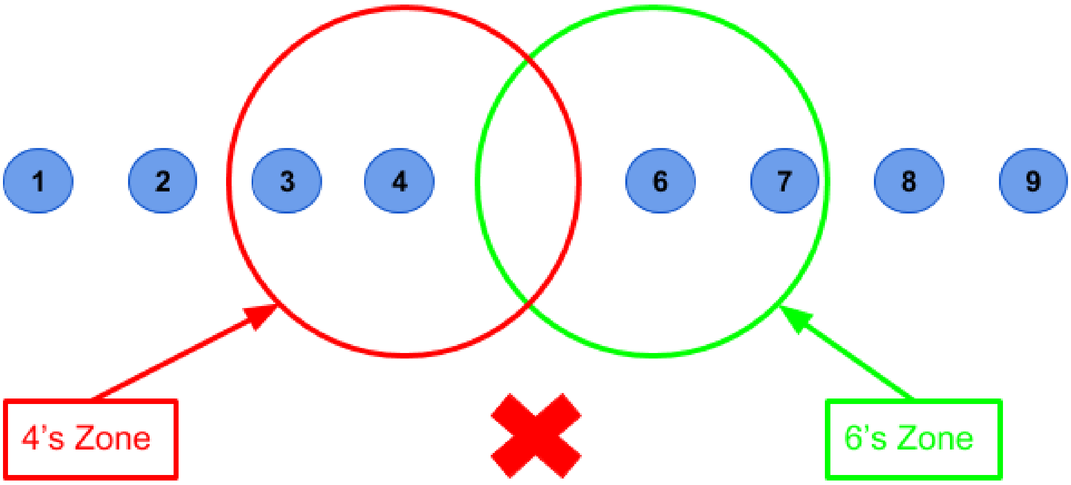


Figure 3.3: System breaks in the middle if one node fails

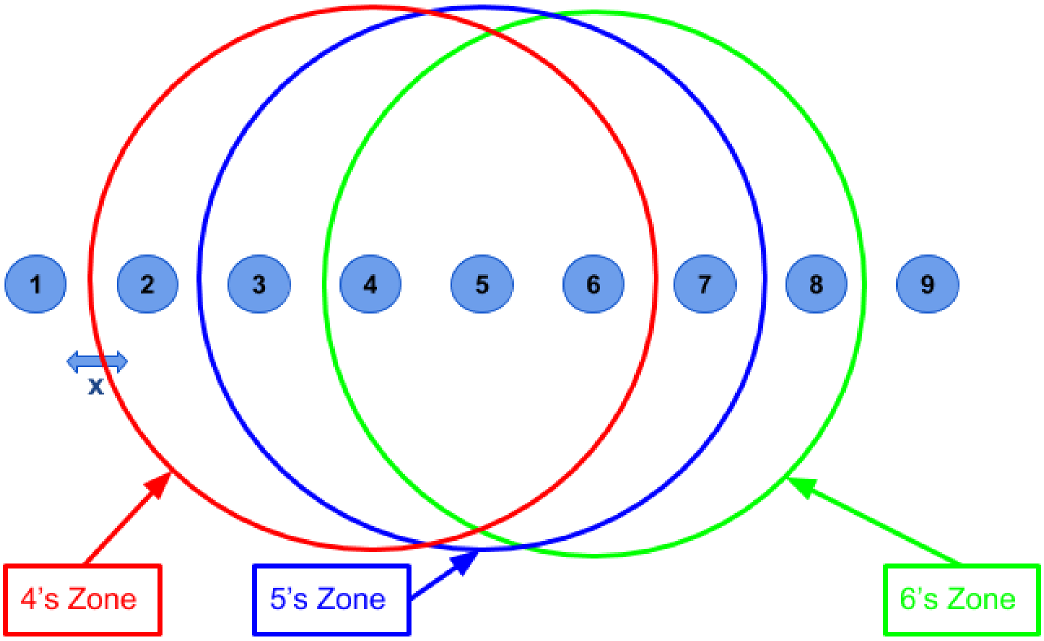


Figure 3.4: Nodes arranged in a row with zone radius 2

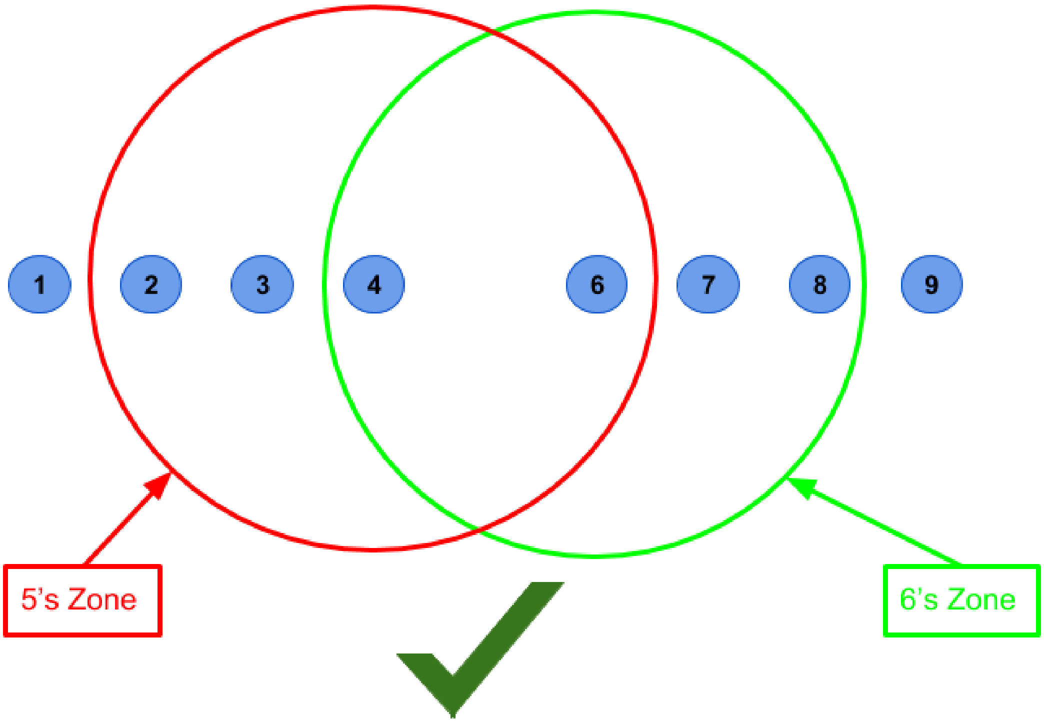


Figure 3.5: System does not break in the middle even if one node fails

Algorithm 1: Dead or Alive Protocol

```
BEGIN:
Boot_system()

WHILE(1)

    Wake_System()
    Cycle <- WAKE_TIME + LONG_DS * PADDING * (RADIO_ID % (ZONE_RADIUS +
    1))
    Add_My_Data()
    Calculate_Hash()
    Broadcast()
    Reduce_Neighbour_Life()

    WHILE (Cycle -- ):
        Check_Fence_Status()
        IF(Got_Message()):
            Check_Hash()
            Merge_Data()
            Update_Neighbour_Table()

    Deep_Sleep()
```

State Diagram

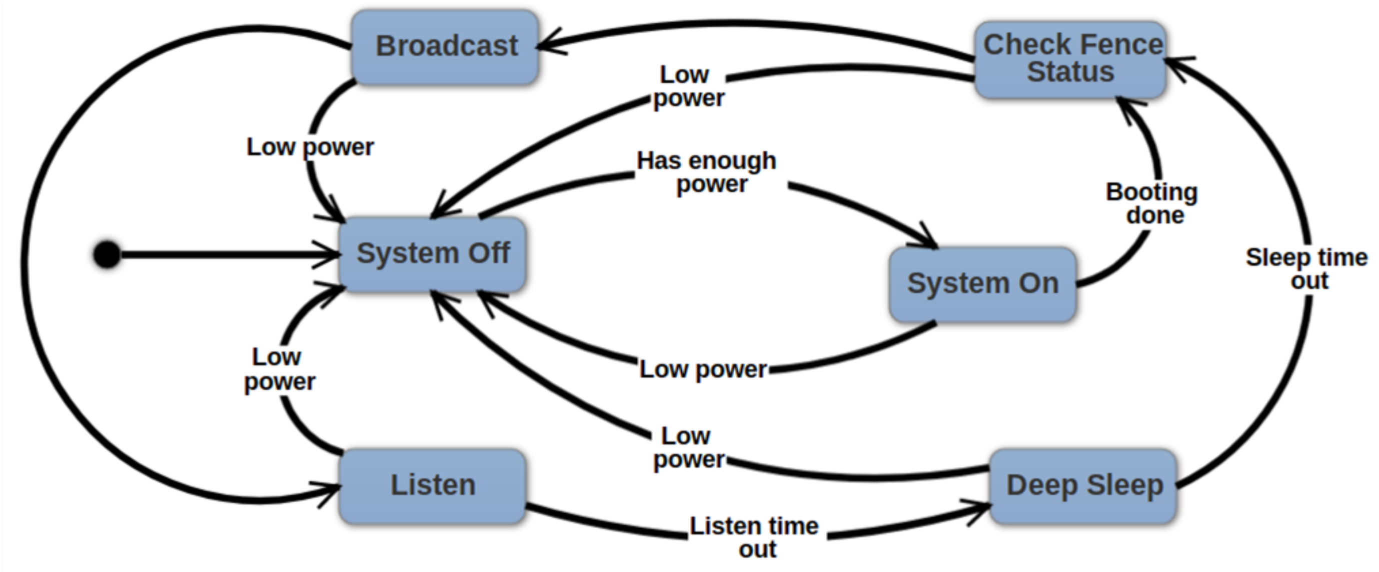


Figure 3.6: State diagram of the system

3.2.1 Dead or Alive Protocol

Duty cycles

Each node has its own duty cycle. As mentioned in algorithm 1, these duty cycles changes dynamically. In their first cycle the nodes follow the default duty cycle. In the second cycle they add a small time to their working period. Then the default cycle and the modified cycle, and so on. The custom duty cycle is calculated by WORK_TIME + PADDING_TIME * (RADIO_ID % (ZONE_RADIUS + 1)). With this, each node gets its own duty cycle. In average all the nodes have the same duty cycle. Figure 3.7 shows an example of how the duty cycles of the nodes are calculated (with dummy values).

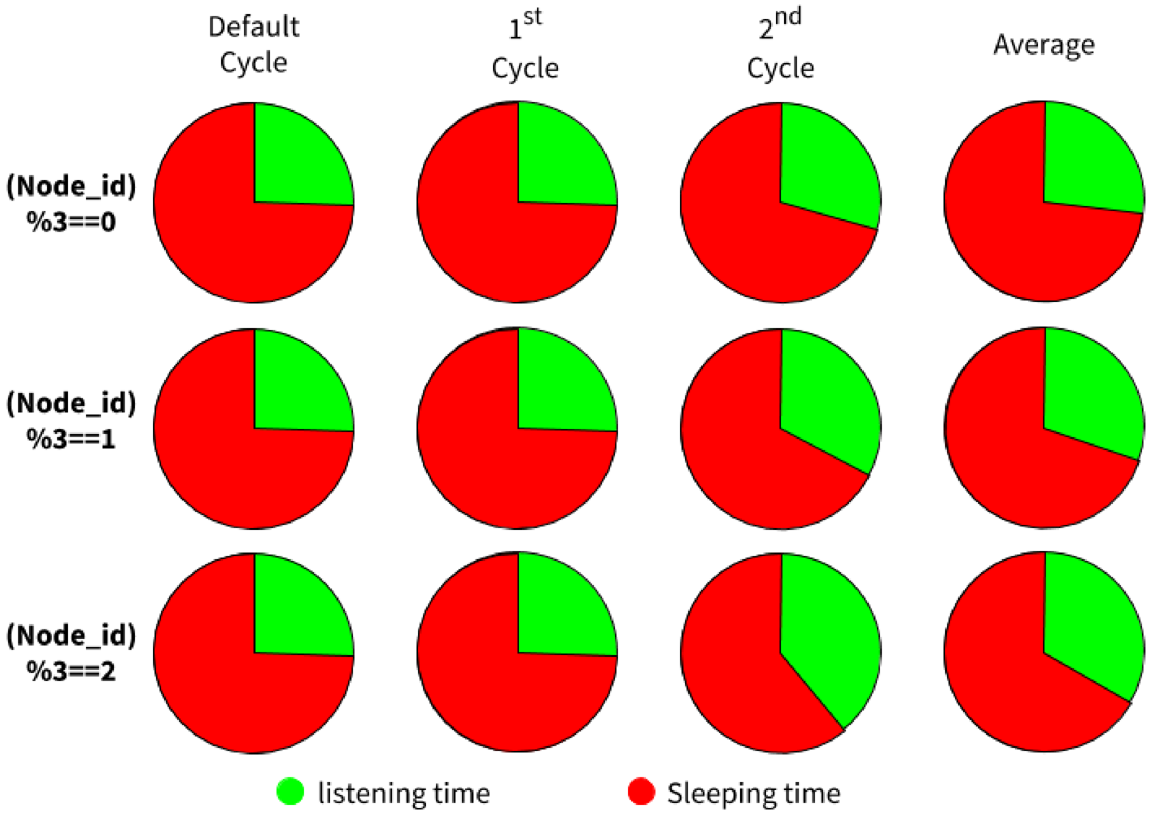


Figure 3.7: Duty cycles of nodes

Broadcasting Message Frame

Every broadcasting packet has a fixed length and it is 32 bits. The first byte is allocated to hold the senders node ID. The next 30 bytes are allocated for the message, each bit in this partition denotes a separate node. If a node works its corresponding bit is set to one. If a node does not work its corresponding bit is set to zero. Within a single frame, statuses of 240 nodes can be sent. The final byte is to hold the checksum of the frame. A checksum is added because all the nodes broadcasts and collisions can occur thus the integrity of the message can be preserved. Therefore the checksum of each message is checked before processing. Figure 3.8 shows the broadcasting message frame structure of the protocol.

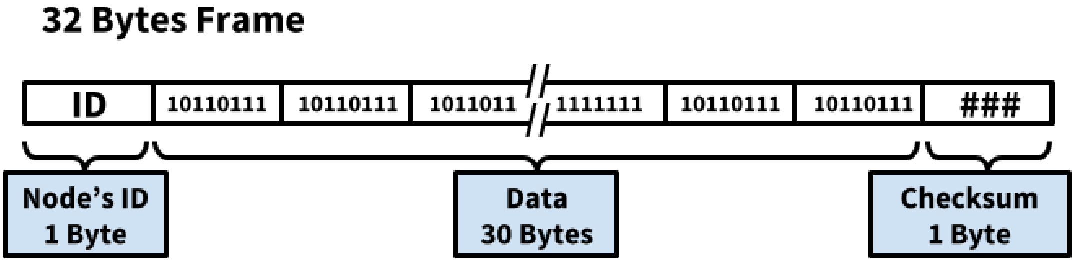


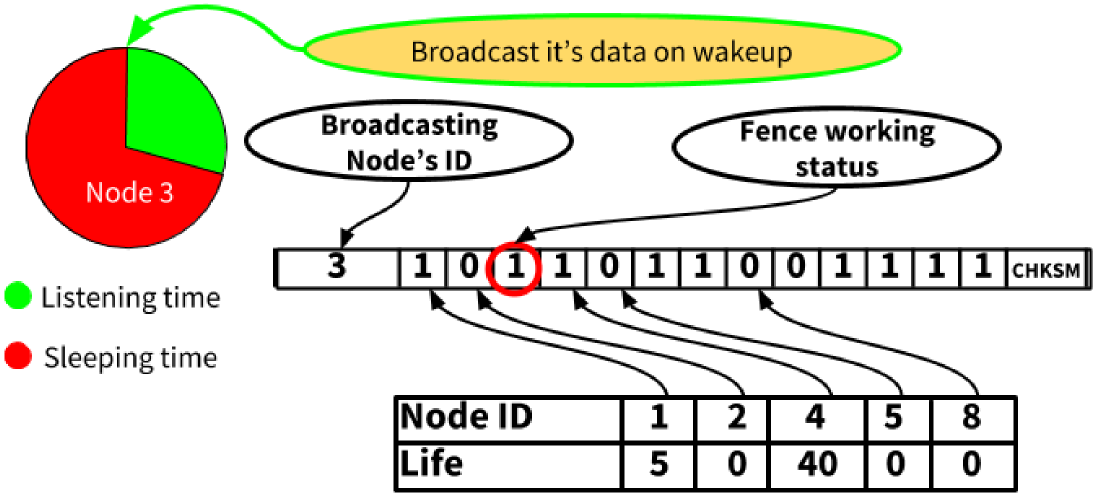
Figure 3.8: Broadcasting message frame structure

Neighbor Table

Every node maintains a neighbor table. When node-01 receives a broadcasting message from node-02, first, node-01 looks in its neighbor table whether there is a record for node-02. If a record does not exists, node-01 creates a new record in its neighbor table with a LIFE value. The life value is set to MAX_LIFE_VALUE if the corresponding bit (which is 2nd bit in this case) value is 1. If the corresponding bit (which is 2nd bit in this case) value is 0, the LIFE value is set to 0. LIFE value is the value which says how long this record is valid. LIFE value is in duty cycles, which means up to how many duty cycles this record is valid. In each new duty cycle all the LIFE values in the neighbor table gets reduced by 1 until it reaches 0. If LIFE value equals 0, it means, this record is no longer valid and treated as broken node. If the LIFE value is greater than 0 it is a valid record and that node is treated as a working node. In this research we use MAX_LIFE_VALUE as 40, which is 40 duty cycles (about 60 seconds' life time).

Broadcasting

Every node broadcasts its data on each wake up. When a node receives a broadcasting message it merges the data with the locally saved data and the neighbor table is updated. When broadcasting it sets its node ID to the first byte of the message then it checks the fence status. If the fence is working, it sets its corresponding bit in the data partition to 1. If the fence is not working it sets its corresponding bit in the data partition to 0. In below figure 3.9, node 3 broadcasts. First it puts 3 in the first byte, then it checks fence status and put in the 3rd bit. Then it looks its neighbor table. If the LIFE values of the neighbors are greater than 0, the corresponding bits are also set to 1. If the LIFE values of the neighbors are 0, the corresponding bits are set to 0. In the below example neighbors 2, 5 and 8 are expired so their corresponding bit values are set to 0 and neighbors 1 and 4 have LIFE values greater than 0, so their corresponding bits are set to 1.



Node ID	1	2	4	5	8
Life	5	0	40	0	0

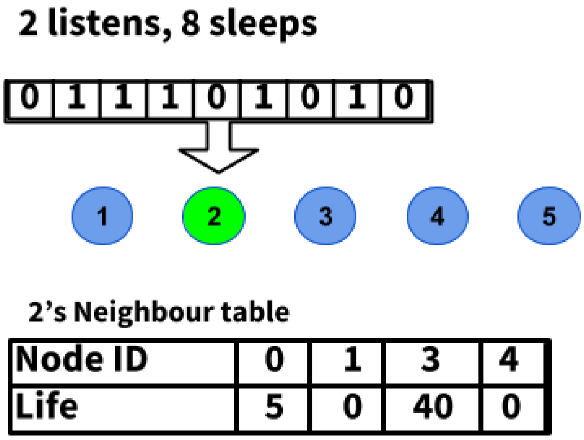
Figure 3.9: Broadcasting message

Data Merging

When a node receives a broadcasting message it merges data it has with the new data it receives from the message and updates the neighbor table. The protocol does not blindly merge the data which will circulate bogus data, instead it uses a special mechanism. Nodes believe that the data they share about their neighbors who are inside their zone are true. They replace this data without doubt.

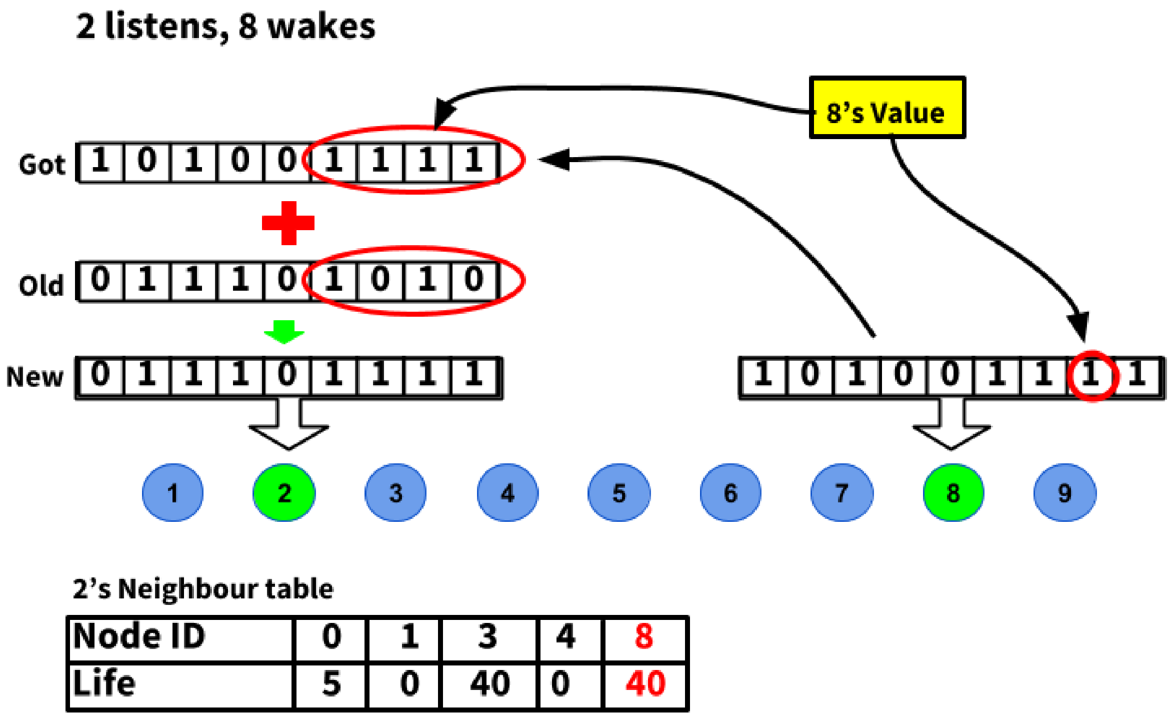
If node-A gets a message from node-B whose ID is greater than node-As ID, node-A takes only the data which is after node-B. Figure 3.10 and figure 3.11 clarifies this. First node-02 listens and node-08 sleeps, and when node-08 wakes it broadcasts a message, and node-02 listens it, then node-02 updates its neighbor table. It does not have a record for node-08, so it adds a new record. Since the 8th bit is 1 which says that the fence is working up to node-08, node-02 sets the LIFE value of node-08 to MAX_LIFE_VALUE. Then from the data partition node-02 takes all the bits after 6th bit including 6th bit. Node-02 takes 6th bit and 7th bit because they are in the zone of node-08 because the whole network knows the zone radius.

When node-A gets a message from node-B whose ID is less than node-As ID, node-A takes only the data which is before node-B. The example in figure 3.12 and figure 3.13 clarifies this. First node-08 is listening and node-02 is sleeping then node-02 wakes and broadcasts a message and node-08 receives it. Node-08 does not have a record for node-02, so it adds a new record. Since the second bit is 0 which says that the fence is not working in node-02, node-08 sets the LIFE value of node-02 to 0. Node-08 takes only the data where the bit position is less than 2. Also it takes the 3rd and 4th bit values since node-08 knows node-03 and node-04 is in node-02s zone and node-02 knows these values.



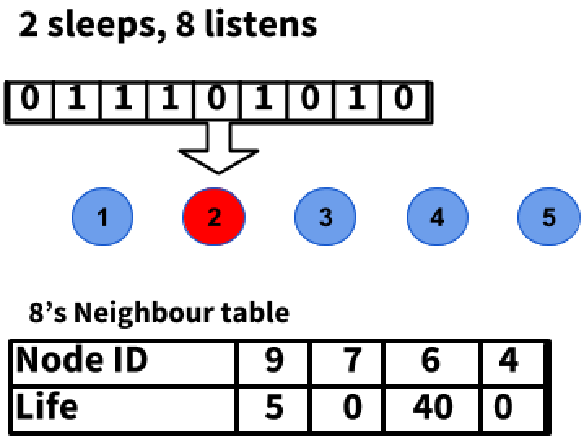
Node ID	0	1	3	4
Life	5	0	40	0

Figure 3.10: Node-02 is about to receive a message from node-08



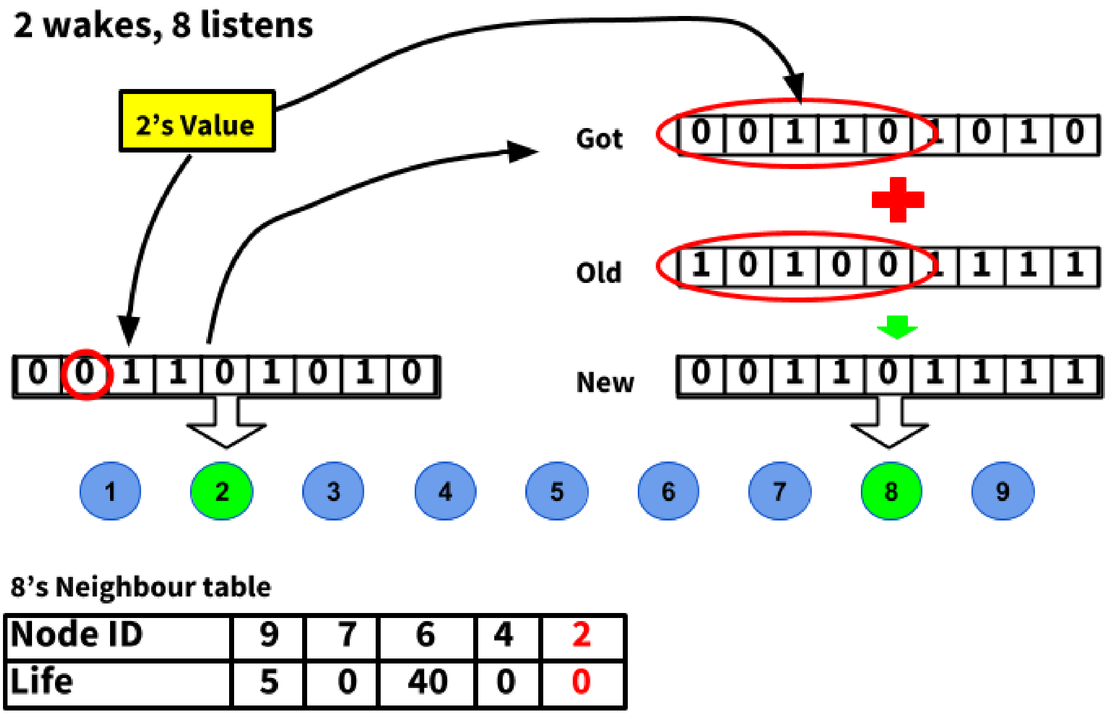
Node ID	0	1	3	4	8
Life	5	0	40	0	40

Figure 3.11: Node-02 receives a message from node-08



Node ID	9	7	6	4
Life	5	0	40	0

Figure 3.12: Node-08 is about to receive a message from node-02



Node ID	9	7	6	4	2
Life	5	0	40	0	0

Figure 3.13: Node-08 receives a message from node-02

3.3 Research Design of the Batteryless Sensor Node

The high-level architecture of designing the batteryless and energy efficient sensor node for electric fence to detect breakage is given in figure 3.14. Figure 3.15 shows the high level diagram of the batteryless breakage detection system.

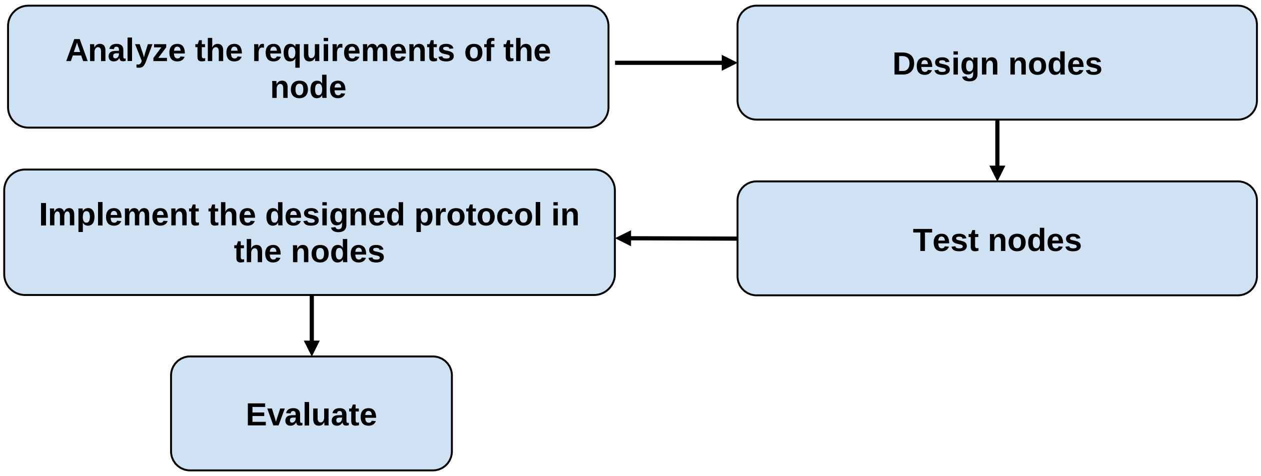

Analyze the requirements of the node
Design nodes
Test nodes
Implement the designed protocol in the nodes
Evaluate

Figure 3.14: Research design of the batteryless sensor node

First, we analyzed the requirements of the batteryless and energy efficient sensor node and came up with the below requirements.

- Maintenance free
  - Batteryless
- Energy efficient
- Low cost
- Detect fence breakages

Next we searched for a method to scavenge energy from the electric fence and store it in a capacitor to power the nodes. But this was unsuccessful because the energizer which powers the fence up does not generate a considerable amount of current to charge a capacitor. Then we came up with the idea of using a small low cost solar cell to power the nodes. The solar cell charges a 47 µf capacitor. This power is enough to power the nodes. Once the capacitor is charged, the node wakes up and listens for some period and goes back to sleep. The capacitor charges during the deep sleep mode. And this is repeated.

Requirements of the Microcontroller of the Node

- Should have enough I/O pins to connect a radio module, and to connect two LDRs which is used to detect the breakages of the fence as shown in figure 3.15.
- Should support deep sleep modes.
- Should have a watchdog timer to wake up from sleep after a given time is completed.
- Should be energy efficient.

Requirements of the Radio Module of the Node

- Should be energy efficient.
- Should be long range.
- Low data rate. Since our Dead or Alive protocol does not need high data rate, low data rate is enough which conserves power.

In figure 3.15, the radio module is shown in as component number 02.

Mechanism to Detect Breakages in Electric Fence

To detect breakages in the fence we use the same method which is used in system [4] explained in section 2.2.6. That is we attach two neon lights to the fence. When high voltage goes through the fence these neon lights emits a light. There are two LDRs connected to the microcontroller and these LDRs are glued to the neon lights. When neon light emits light the resistance of the LDR reduces. The resistance of the LDR can be detected by the microcontroller and can know the state of the fence, whether the fence is working or not working. As shown in figure 3.15, 8 and 9 are neon lights which is connected to the high voltage lines and 6 and 7 are LDRs connected to the microcontroller. Neon-08 and LDR-06 are glued tightly and insulated and Neon-09 and LDR-07 are glued tightly and insulated, so that LDRs does not respond to the outside light. It gets light only from the neon light. This method separates the electric fence from the breakage detection system. Therefore this method protects the system if a lightning strikes to the fence or any other harm which can cause by high voltage.

Mechanism to Power the Nodes without Batteries

To make the system easy to maintain the system should be batteryless, because all types of batteries need replacement after sometime. In order to make out system batteryless, we use a small capacitor. As shown in figure 3.6, once the capacitor (shown in number 03) is charged and has enough power the node starts working then after some time nodes goes to deep sleep. During deep sleep time the capacitor start charging again. Then it goes back to the working state again. As the energy source a small solar panel is used. There are systems [6] using solar panels but they also use a rechargeable battery. In these systems battery charges with the solar panel and the system works from the battery. But again the problem with these systems are, their batteries die after some time so need costly maintenance after sometime. Our novel idea replaces these systems. Our whole system powers with this small capacitor and consumes less power. The Dead or Alive protocol helps the system to work energy efficiently. Figure 3.15 shows how the system is powered. Solar panel (4) charges the capacitor (3) and it powers the microcontroller (1).

Deep Sleep

To make the system energy saving duty cycling method is introduced. And during deep sleep the radio module and the microcontroller is put in to sleep. Only the watchdog timer in the microcontroller works during this period. This does not consume much power. The purpose of the watchdog timer is to interrupt the microcontroller to wake up and work again. During deep sleep period the capacitor is charged.

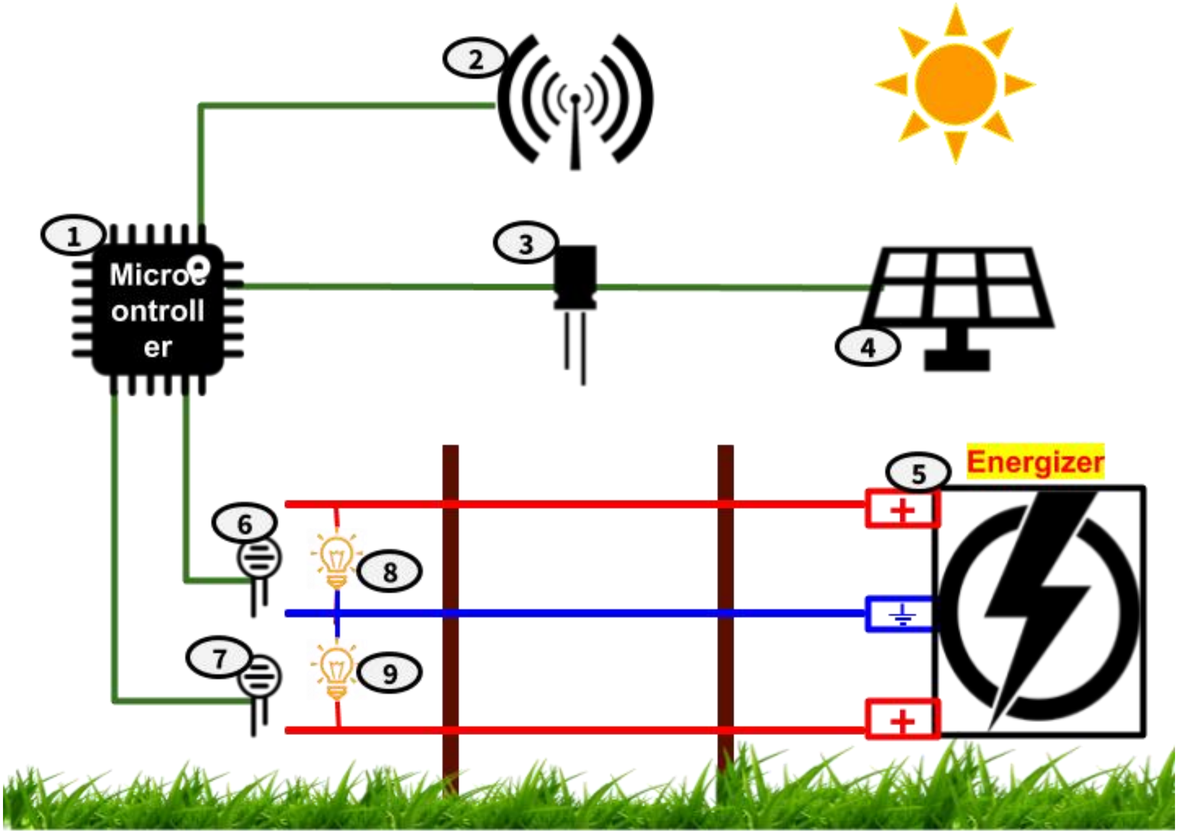


Figure 3.15: High level diagram of the batteryless breakage detection system

States of the System

As shown in figure 3.6 first the system is in ‘system off’ state. Once the capacitor is charged to power the node the system goes to the ‘system on’ state. Until the power supply goes down the system will not got to the ‘system off’ state, once there is no enough power the system goes to the ‘system off’ state from any state. After the booting process is completed it will go to the ‘check fence status’ state where the fence status is checked by checking the neon light and then the system will go to the ‘broadcast’ state where it broadcasts its data to others and goes to the ‘listen’ state where it listens for others broadcasts. After the time out, the system will go to the ‘deep sleep’ state where it does nothing and lets the capacitor charge. After it completes the sleeping time, it again goes to the ‘check fence status’ state. And this process repeats until the power source gets down.

3.4 Conclusion

This chapter provided a detailed description on the research design. The research design encompasses two main steps which are designing of the wireless communication protocol and designing of batteryless sensor node. Research design of the wireless communication protocol is a software approach while the research design of the batteryless sensor node is a hardware approach.

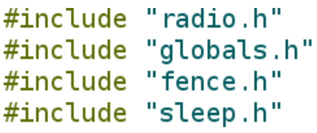
Chapter 4 - Implementation

4.1 Introduction

This chapter explains the steps taken in implementing the batteryless electric fence breakage detection system. Proposed in Chapter 3 and describes the various tools used. Implementation was carried out in two phases; implementing the Dead or Alive protocol, implementing the batteryless sensor node. Section 4.2 describes the software tools utilized for implementation, Section 4.3 presents the implementation details of the "Dead or Alive" protocol and Section 4.4 presents the implementation details of the batteryless sensor node.

4.2 Software Tools

The proposed wireless communication protocol was implemented using C/C++, with Arduino IDE. D. Parson's NRFLite [26] library was used in microcontroller's code to communicate with the radio module. To design the circuits and schematics the open source KiCad software was used.



```
#include "radio.h"
#include "globals.h"
#include "fence.h"
#include "sleep.h"
```

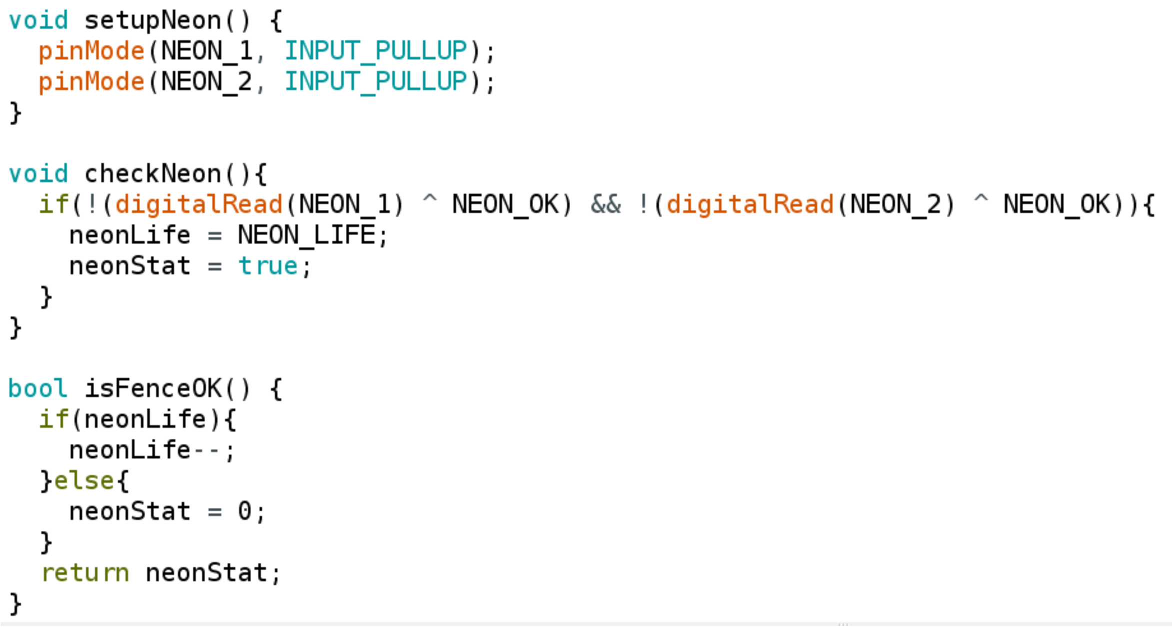
4.3 Implementation of the Dead or Alive Protocol

These are the included files in the main file. "radio.h" contains the methods to communicate with the radio module. "globals.h" contains the global variables and definitions. "fence.h" contains the methods used to check the fence status. "sleep.h" contains the methods which uses to put the system to sleep mode. Codes of "radio.h" and "sleep.h" are attached in the appendix section.

```
#define PACKET_SIZE 32

#define BCAST_ID 255// broadcast id
#define RADIO_ID BCAST_ID

#define DEFAULT_WORK_TIME 30//325

#define MAX_NEIGHBOURS 10
#define NEIGHBOUR_LIFE 15 //in duty cycles

#define PAD 50//padding for working time

int WORK_TIME = DEFAULT_WORK_TIME;

uint8_t NODE_ID = 255;//radio id.

#define ZONE_RADIUS 2//number of nodes

#define NEON_LIFE 5 //in duty cycles
```

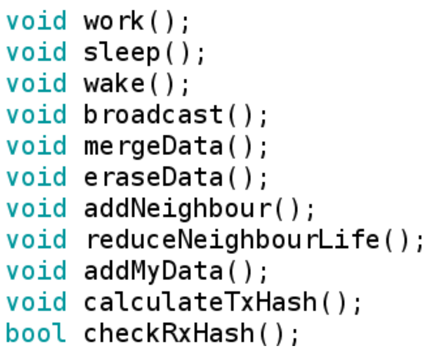
The above code is in the “globals.h” file. PACKET_SIZE is the size of the packet which nodes broadcast. BCAST_ID is the broadcasting ID. All the nodes set their radio ID to this value which is RADIO_ID. And all the radio modules broadcast to this value. This value is not the node ID. Node ID is a unique ID to identify the nodes it is defined as NODE_ID. It is stored in the microcontrollers EEPROM. By default it is set to 255. This value is read from the EEPROM on the booting process. DEFAULT_WORK_TIME is the default period which all nodes work. MAX_NEIGHBOURS is the maximum array size of the neighbors. NEIGHBOUR_LIFE is the maximum life a neighbor gets. This is in duty cycles. As described in section 3 PAD is the padding time which every node use to calculate their dynamic working time. ZONE_RADIUS is the zone radius by number of nodes. NEON_LIFE is the maximum number of duty cycles a neon status read is valid.

```
#define NEON_1 0
#define NEON_2 4

#define NEON_OK LOW

int neonLife = 0;
bool neonStat = false;
```



```
void setupNeon() {
  pinMode(NEON_1, INPUT_PULLUP);
  pinMode(NEON_2, INPUT_PULLUP);
}

void checkNeon(){
  if(!(digitalRead(NEON_1) ^ NEON_OK) && !(digitalRead(NEON_2) ^ NEON_OK)){
    neonLife = NEON_LIFE;
    neonStat = true;
  }
}

bool isFenceOK() {
  if(neonLife){
    neonLife--;
  }else{
    neonStat = 0;
  }
  return neonStat;
}
```

The above code is in the “fence.h” file. LDR-01 which is attached to neon-01 is connected to the pin 0 of the microcontroller and LDR-02 which is attached to neon-02 is connected to the pin 4. In setupNeon() method the pins are initialized. In checkNeon() method checks whether both the neons lights are lighting means both the high voltage lines and the ground line is working. isFenceOK() method returns the status of the fence.



```
void work();
void sleep();
void wake();
void broadcast();
void mergeData();
void eraseData();
void addNeighbour();
void reduceNeighbourLife();
void addMyData();
void calculateTxHash();
bool checkRxHash();
```

Above are the methods in the main file. work() is the function which is executed during working time. sleep() method is called when node wants to sleep. Broadcast method is called when the node wants t broadcast. mergeData() method is called when a message is received to extract the data from the message. eraseData() is called when the node wants to erase the data it has. addNeighbour() method is called when the node receives a message, to update the neighbor table. reduceNeighbourLife() is called in each cycle to reduce the life of the neighbor. addMyData() method is called when broadcasting a message, and is used to add local data to the broadcasting message. calculateTxHash() is called when broadcasting to calculate the hash of the transmitted packet. checkRxHash() is called to check the hash value is valid in a received packet.

```
byte rxData[PACKET_SIZE];
byte txData[PACKET_SIZE];

typedef struct {
  byte from;
  byte data[PACKET_SIZE - 2];// two bytes for 'from' and 'hash'
  byte hash;
}  * Packet;

byte neighbourCount = 0;
byte neighbours[MAX_NEIGHBOURS];
byte neighboursLife[MAX_NEIGHBOURS];


Packet rxPacket =  (Packet)rxData;
Packet txPacket =  (Packet)txData;
```

Received data is stored in the rxData[] array. txData[] array is to holds the data to be broadcasted. Also it contains the local data. As mentioned in section 3 the structure of the packet is first byte is to hold the senders node-ID and 30 bytes for data and the last byte is for the checksum. neighbourCount holds the number of neighbors currently the node has. Initially it is set to 0. neighbours[] array holds the node-IDs of the neighbors while the neighboursLife[] array holds the lives of the corresponding neighbors who are in the neighbour[] array.

```
void setup() {
  setupMemory();

  txPacket->from = NODE_ID;
  eraseData();

  setupRadio(RADIO_ID);

  setupNeon();
  setupSleep();
}
```

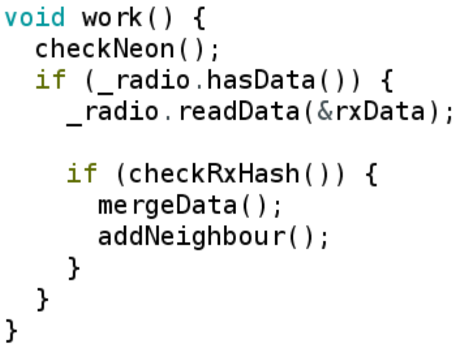
First of all this method is called. It setups the memory, sets the node ID to the source field of the broadcasting packet, setups the radio with the radio ID, setups the LDRs which is attached to the neon bulbs, setups the watchdog timer and sleeping parameters of the system.

```
int cycle;
bool LONG_DS = false;//long dutycycle
void loop() {
  wake();

  cycle = WORK_TIME + LONG_DS * PAD * (NODE_ID % (ZONE_RADIUS + 1));

  broadcast();
  reduceNeighbourLife();

  while (cycle--) {
    work();
  }

  sleep();//sleeps for one second

  LONG_DS = !LONG_DS;
}
```

After setup system enters to this loop and never goes out until power downs. LONG_DS is to check whether the cycle is a long duty cycle or the default duty cycle. Here the node wakes, calculates it’s working time then broadcasts then reduces neighbor life and works until the calculated time ends and goes to sleep for a second. Then it changes the default cycle to long duty cycle and long duty cycle to default duty cycle.

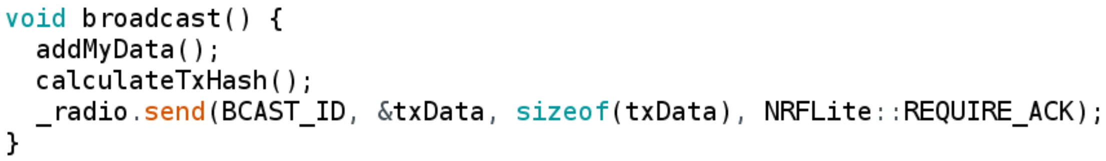
```
void sleep() {
  _radio.powerDown();
  watchdogSleep1s();
}

void wake() {
  setupRadio(RADIO_ID);
  _radio.hasData();
}
```

In sleep method, it puts the radio to power down mode then goes to sleep by telling the watchdog timer to interrupt after a one second. In wake method the radio is re initialized and hasData() method is called to wake the radio up again.



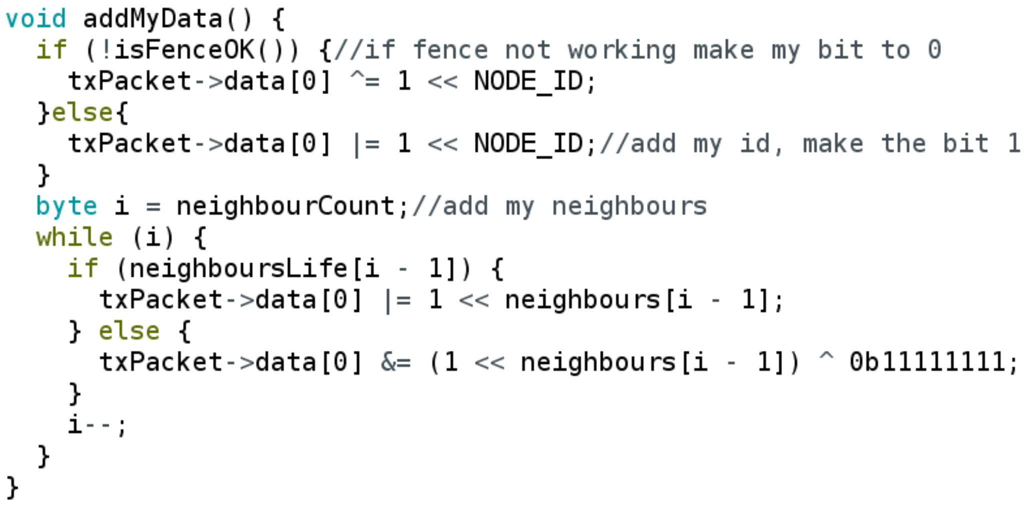
```
void work() {
  checkNeon();
  if (_radio.hasData()) {
    _radio.readData(&rxData);

    if (checkRxHash()) {
      mergeData();
      addNeighbour();
    }
  }
}
```

In work() method it checks the neon status and listens for messages. If the radio gets a message it checks the hash of the message. If the hash is valid it merges the data and updates the neighbor table.



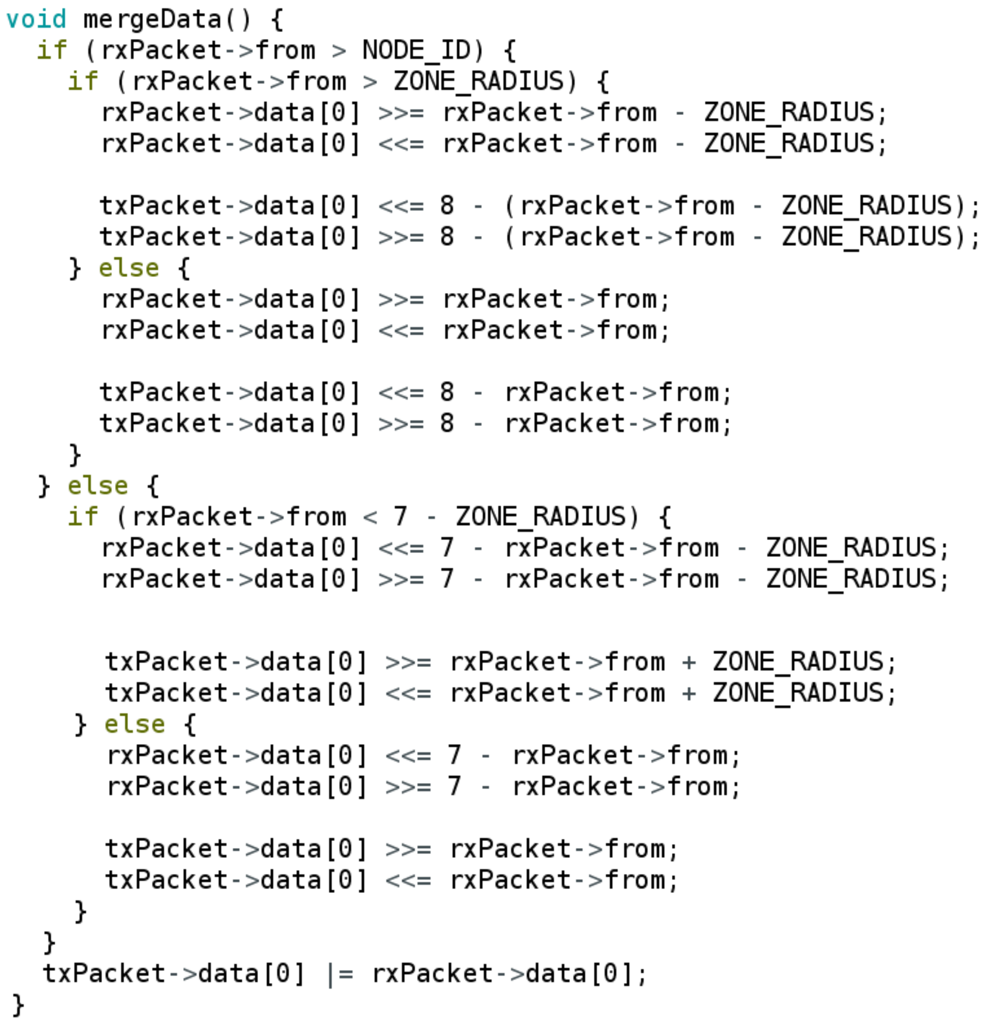
```
void broadcast() {
  addMyData();
  calculateTxHash();
  _radio.send(BCAST_ID, &txData, sizeof(txData), NRFLite::REQUIRE_ACK);
}
```

In broadcast() method the node adds its data to the txPacket and calculates the hash and broadcasts the message.

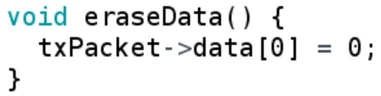


```
void addMyData() {
  if (!isFenceOK()) {//if fence not working make my bit to 0
    txPacket->data[0] ^= 1 << NODE_ID;
  }else{
    txPacket->data[0] |= 1 << NODE_ID;//add my id, make the bit 1
  }
  byte i = neighbourCount;//add my neighbours
  while (i) {
    if (neighboursLife[i - 1]) {
      txPacket->data[0] |= 1 << neighbours[i - 1];
    } else {
      txPacket->data[0] &= (1 << neighbours[i - 1]) ^ 0b11111111;
    }
    i--;
  }
}
```

This method adds local data to the broadcasting message. First it checks the local fence status. If fence is working it sets its bit to 1 else to 0. Then it goes through the neighbor table and adds neighbor working status to the message.

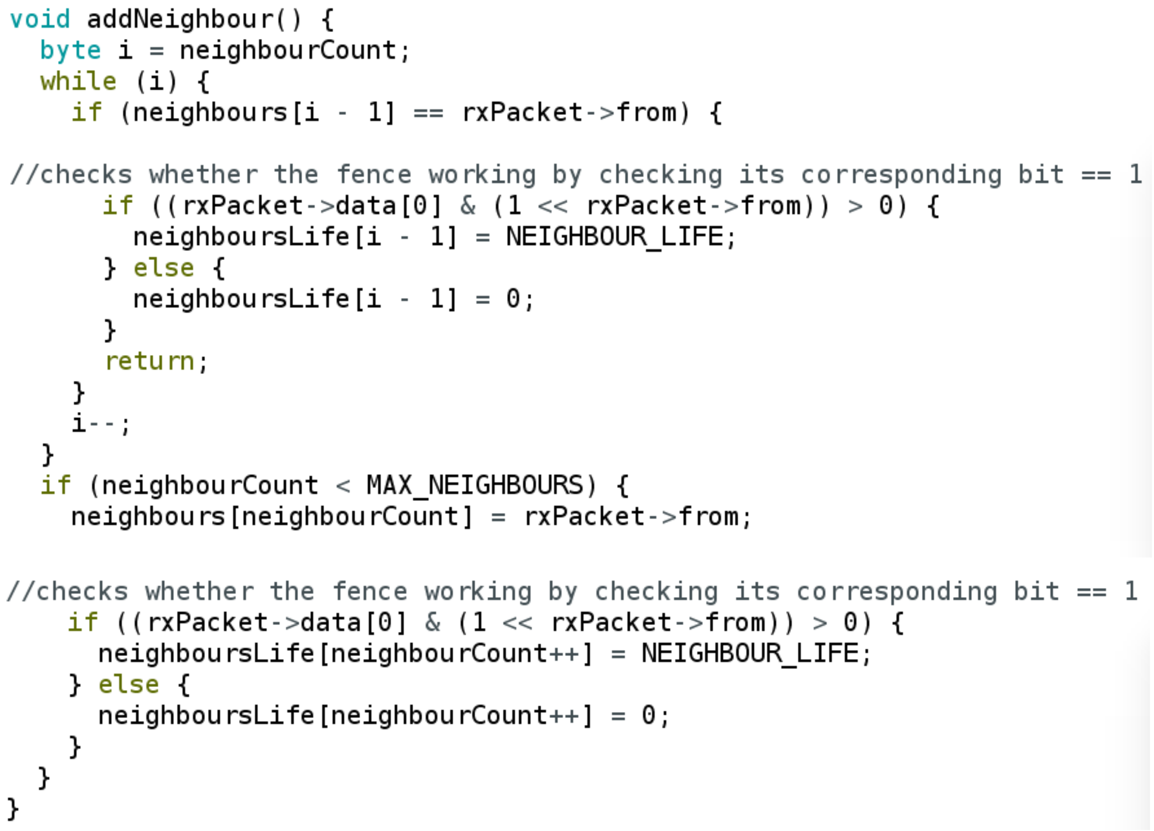


```
void mergeData() {
  if (rxPacket->from > NODE_ID) {
    if (rxPacket->from > ZONE_RADIUS) {
      rxPacket->data[0] >>= rxPacket->from - ZONE_RADIUS;
      rxPacket->data[0] <<= rxPacket->from - ZONE_RADIUS;

      txPacket->data[0] <<= 8 - (rxPacket->from - ZONE_RADIUS);
      txPacket->data[0] >>= 8 - (rxPacket->from - ZONE_RADIUS);
    } else {
      rxPacket->data[0] >>= rxPacket->from;
      rxPacket->data[0] <<= rxPacket->from;

      txPacket->data[0] <<= 8 - rxPacket->from;
      txPacket->data[0] >>= 8 - rxPacket->from;
    }
  } else {
    if (rxPacket->from < 7 - ZONE_RADIUS) {
      rxPacket->data[0] <<= 7 - rxPacket->from - ZONE_RADIUS;
      rxPacket->data[0] >>= 7 - rxPacket->from - ZONE_RADIUS;


      txPacket->data[0] >>= rxPacket->from + ZONE_RADIUS;
      txPacket->data[0] <<= rxPacket->from + ZONE_RADIUS;
    } else {
      rxPacket->data[0] <<= 7 - rxPacket->from;
      rxPacket->data[0] >>= 7 - rxPacket->from;

      txPacket->data[0] >>= rxPacket->from;
      txPacket->data[0] <<= rxPacket->from;
    }
  }
  txPacket->data[0] |= rxPacket->data[0];
}
```



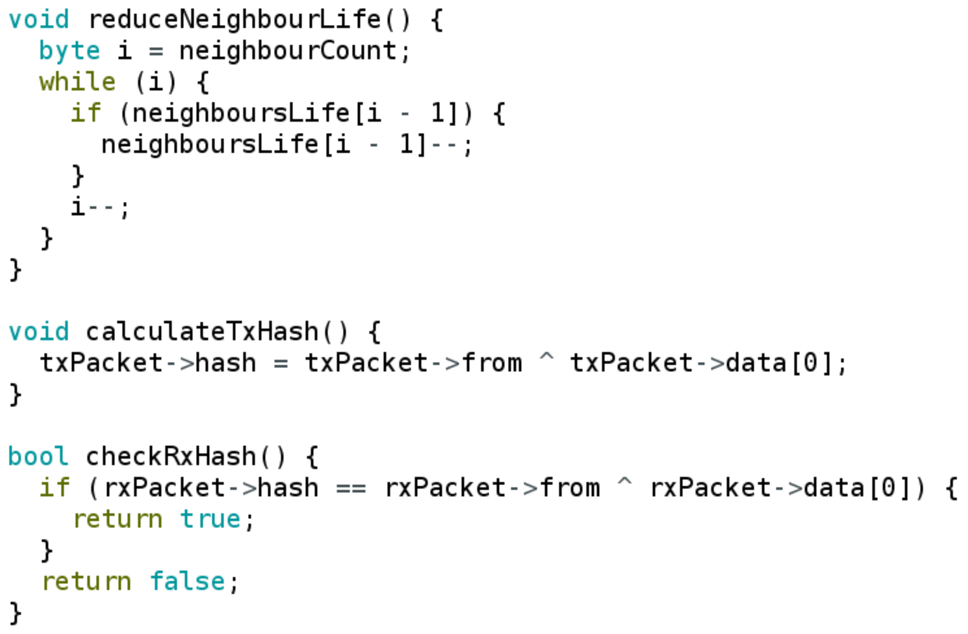
```
void eraseData() {
  txPacket->data[0] = 0;
}
```

mergeData() method extract valid data from a received packet and merges with local data. As described in section 3 first it checks whether the received message is from a previous node or a next node and applies the algorithm as mentioned in the section 3. eraseData() method erases the data in the data portion of the txPacket which is local data.



```
void addNeighbour() {
  byte i = neighbourCount;
  while (i) {
    if (neighbours[i - 1] == rxPacket->from) {

//checks whether the fence working by checking its corresponding bit == 1
      if ((rxPacket->data[0] & (1 << rxPacket->from)) > 0) {
        neighboursLife[i - 1] = NEIGHBOUR_LIFE;
      } else {
        neighboursLife[i - 1] = 0;
      }
      return;
    }
    i--;
  }
  if (neighbourCount < MAX_NEIGHBOURS) {
    neighbours[neighbourCount] = rxPacket->from;

//checks whether the fence working by checking its corresponding bit == 1
    if ((rxPacket->data[0] & (1 << rxPacket->from)) > 0) {
      neighboursLife[neighbourCount++] = NEIGHBOUR_LIFE;
    } else {
      neighboursLife[neighbourCount++] = 0;
    }
  }
}
```

This method updates the neighbor table when a message is received. First it checks whether the node is already in the table is not it adds the node to the table. Then it checks the status of the fence of that node which is by checking the corresponding bit value in the data partition. If it is 1, the life of that neighbor is set to the maximum life else it is set to 0.



```
void reduceNeighbourLife() {
  byte i = neighbourCount;
  while (i) {
    if (neighboursLife[i - 1]) {
      neighboursLife[i - 1]--;
    }
    i--;
  }
}

void calculateTxHash() {
  txPacket->hash = txPacket->from ^ txPacket->data[0];
}

bool checkRxHash() {
  if (rxPacket->hash == rxPacket->from ^ rxPacket->data[0]) {
    return true;
  }
  return false;
}
```

In reduceNeighbourLife() method, it goes through all the records of the neighbor table and reduces the life of all neighbors by 1. In calculateTxHash() method it calculates the hash of the transmitting packet by XORing the first byte and the first byte of the data partition. In checkRxHash() method it checks whether the hash of the received packet is valid.

4.4 Implementation of the Batteryless Sensor Node

Microcontroller of the Node

Out of the microcontrollers which are available in the current market. ATtiny85 [25] is the most suitable microcontroller, which has 6 I/O pins, 8Kb flash memory, and a watchdog timer. This was chosen because it has enough number of I/O pins to connect the radio module and to connect 2 LDRs which is used to check the fence status and it has enough flash memory to store the protocol which designed in the previous step. Table 4.1 shows the comparison of the available microcontrollers in the market. Figure 4.1 is the selected ATtiny85 microcontroller.

Table 4.1: Comparison of the available microcontrollers

Microcontroller	***ATtiny2313***	***ATtiny84***	***ATtiny85***	***ATmega328***	***ATmega2560***
Cost	$3.13	$3.53	$2.65	$3.70	$16.75
Pin Count	20	14	8	32	100
Max IO pins	18	12	6	23	86
Flash Memory	2 Kb	8 Kb	8 Kb	32 Kb	256 Kb
EEPROM	128 bytes	512 bytes	512 bytes	1 Kb	1 Kb
PWM channels	4	4	6	6	15
Timers	1 8 bit 1 16 bit	1 8 bit 1 16 bit	2 8 bit	2 8 bit 1 16 bit	2 8 bit 4 16 bit

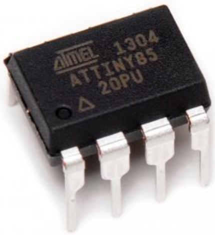


Figure 4.1: ATtiny58

Figure 4.2: NRF24l01

Radio Module of the Node

Out Of the radio technologies available in the market the most suitable technology is NRF24 technology. This is low cost and low power consumption with long range transmission. So for this research NRF24l01 (1Mbps version) module is selected to implement the nodes. Table 4.2 shows the comparison of the available radio technologies in the market. Figure 4.2 is the selected NRF24l01 radio module.

Table 4.2: Comparison of the available radio technologies

	BLE	*Bluetooth*	*LORA*	*WiFi*	*XBee*	*NRF24l01*
Range	< 50m	< 100m	< 20km	< 100m	< 1km	< 100m
Cost	< $5	< $5	< $5	< $3	> $25	< $1.5
Power Consumption	LOW	HIGH	LOW	HIGH	LOW	LOW

Circuit Diagram

To implement a node below electric components were used. The designed schematic diagram is shown in figure 4.3. The final nodes are shown in the appendix section.

- ATtiny85 x1
- NRF24l01 x1
- Solar cell (5.40 cm x 5.80 cm, 5V) x1
- Small neon bulb x2
- LDR x2
- 1K resistor x1
- 4.7K resistor x1
- 0.1µf capacitor x1
- 47µf capacitor x1

In addition to these a special node for the base station is also designed. And a user interface to see and locate the broken nodes is also implemented. These are explained in the appendix section.

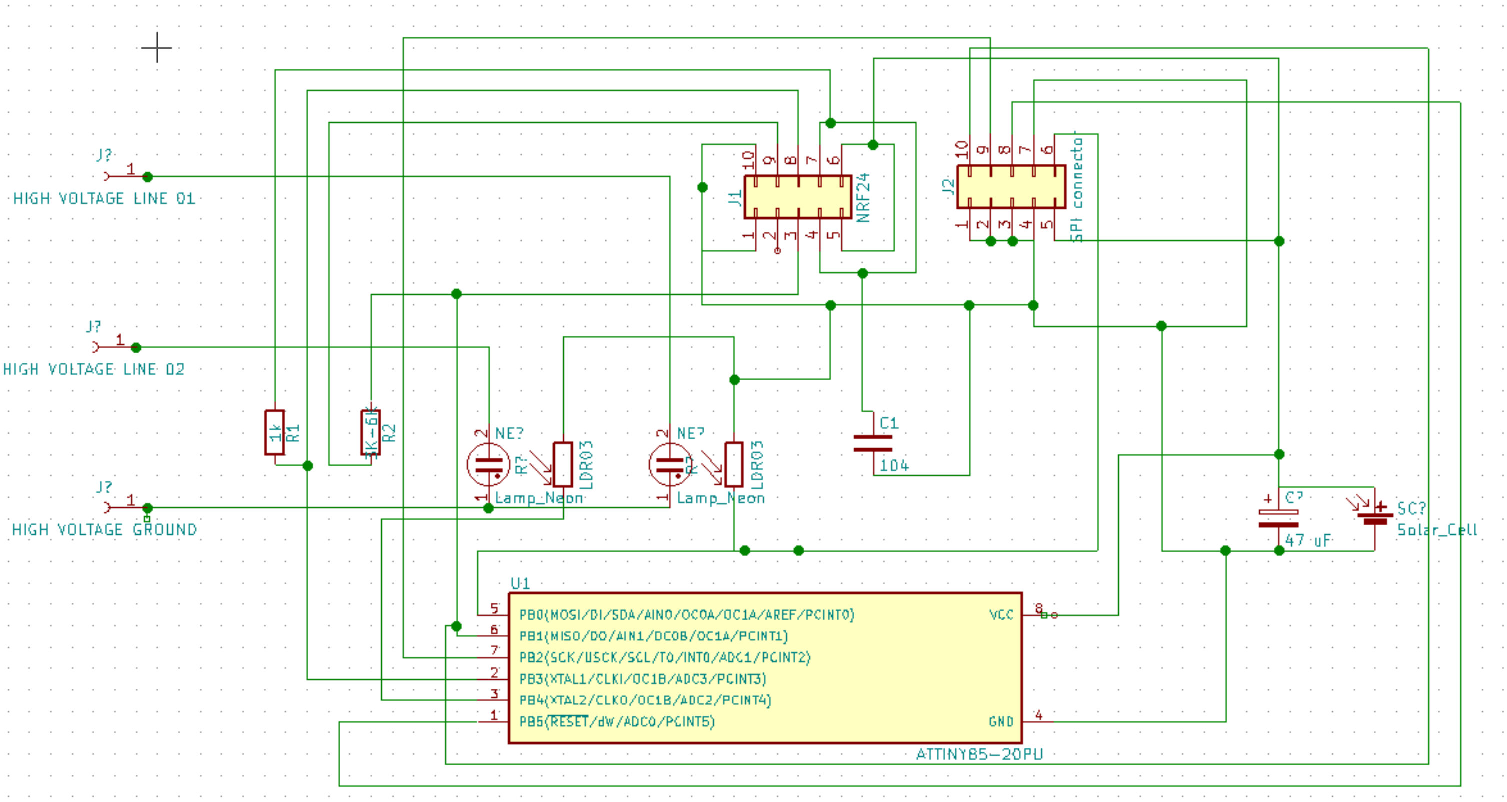

J?
1
HIGH VOLTAGE LINE 01
J?
1
HIGH VOLTAGE LINE 02
J?
1
HIGH VOLTAGE GROUND
1k
R1
1K-6K
R2
NE?
R?
LDR03
Lamp_Neon
NE?
R?
LDR03
Lamp_Neon
J1
NRF24
J2
SPI connector
C1
104
C?
47 uF
SC?
Solar_Cell
U1
PB0(MOSI/DI/SDA/AIN0/OC0A/OC1A/AREF/PCINT0)
PB1(MISO/DO/AIN1/DC0B/OC1A/PCINT1)
PB2(SCK/USCK/SCL/T0/INT0/ADC1/PCINT2)
PB3(XTAL1/CLKI/OC1B/ADC3/PCINT3)
PB4(XTAL2/CLKO/OC1B/ADC2/PCINT4)
PB5(RESET/dW/ADC0/PCINT5)
VCC
GND
ATTINY85-20PU

Figure 4.3: Schematic diagram of the node

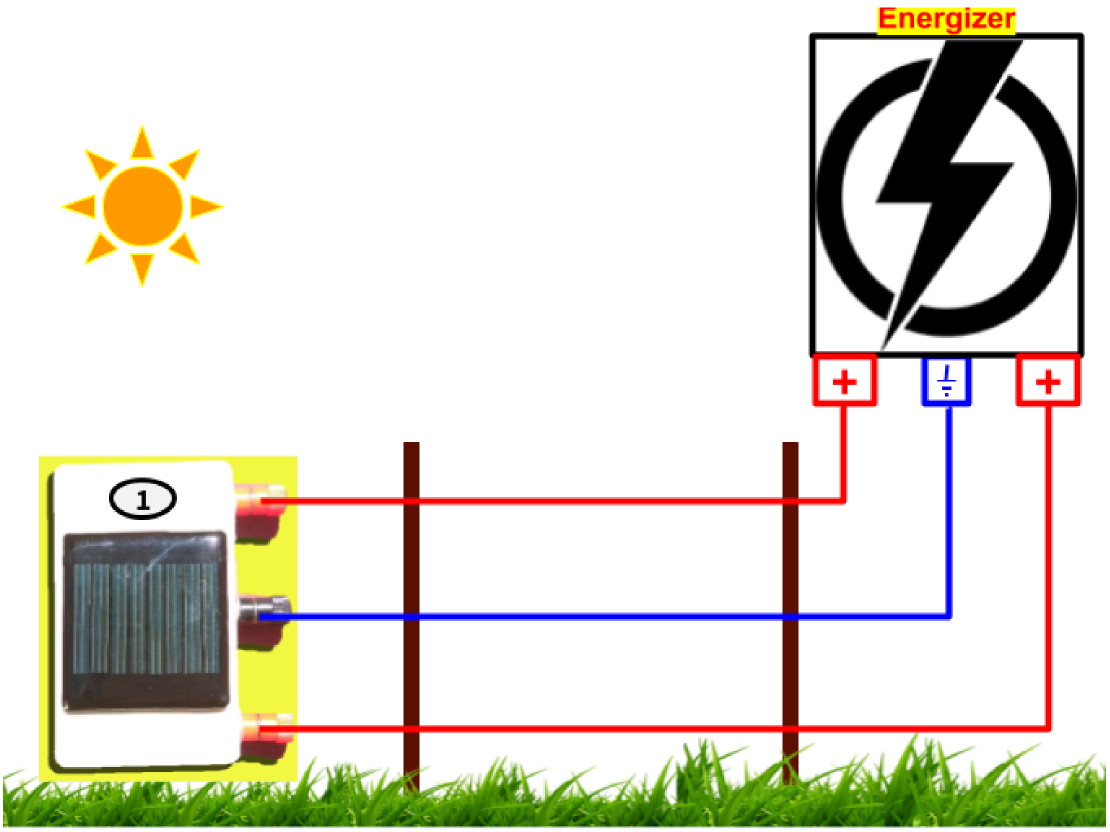


Figure 4.4: Prototype connected to the electric fence

Figure 4.4 shows how the prototype is connected to the fence. The node is denoted in number 1. In the node, there are 3 connector ports in red black and red. These are to connect the high voltage wires to the node. The high voltage wires are connected to a red connectors and the ground wire of the high voltage which comes from the energizer is connected to the black connector. The connection of the nde to an actual electric fence is shown in the appendix section.

4.5 Conclusion

In the first phase the main software's used for the implementation was explained. Then the implementation is divided into two main phases which are, the implementation of the "Dead or Alive" protocol was elaborated and the implementation of the energy efficient batteryless node was elaborated. Finally the implemented wireless communication protocol was installed in the prototype node and was evaluated. The evaluation of the batteryless sensor system for breakage detection in electric fence is elaborated in the next chapter.

Chapter 5 - Results and Evaluation

5.1 Introduction

This chapter elaborates the experimental results to evaluate the proposed solution. Section 5.2 describes the results and the evaluation of the proposed Dead or Alive protocol the proposed batteryless electric fence breakage detection system. To evaluate the system, 7 nodes were made. All the experiments are conducted by using these seven (7) nodes.

5.2 Results and the Evaluation of the Proposed System

There are four parameters that changes the behavior of the protocol. They are: (1)NEIGHBOUR_LIFE - Defines how long a neighbor record in the neighbor table is valid in cycles.
(2)MAX_NEIGHBOURS - Defines the size of the neighbor table.
(3)DEFAULT_WORK_TIME - Defines the default working time.
(4)ZONE_RADIUS - Define the zone radius in number of nodes.

For the experiments, NEIGHBOUR_LIFE is set to 40 which is about 60 seconds, MAX_NEIGHBOURS=10, DEFAULT_WORK_TIME with 600 milliseconds and ZONE_RADIUS=2 which means the nodes are arranged in a way that four nodes are in a nodes zone when arranged in order as shown in figure 3.4.

Since a big area is needed to set up 7 nodes as shown in figure 3.4 with zone radius 2, the experiments were conducted in a room with multicasting without broadcasting. So the node-N sends the message to node-(N-2), node-(N-1), node-(N+1) and node-(N+2). For example node 5 will multicast to node-03, node-04, node-06 and node-07. First, the network was set up and started. Then one hour traffic was captured and was analyzed.

Table 5.1: Broadcasted packets

Node ID	Broadcasts	Successful Broadcasts	Failed Broadcasts	Success Rate
1	2246	684	1562	30.5
2	2044	1018	1026	49.8
3	2480	1450	1030	58.5
4	2280	1186	1094	52.0
5	2158	1082	1076	50.1
6	2638	1092	1546	41.4
7	2274	658	1616	28.9

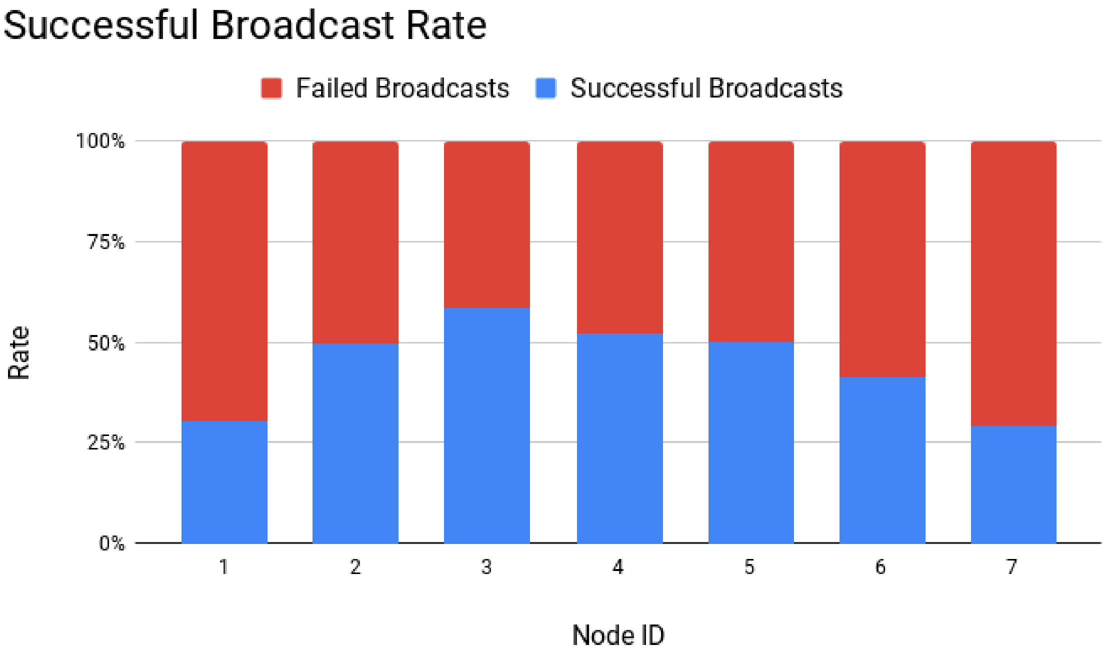


Figure 5.1: Success rate of broadcasted messages

Table 5.1 shows the total number of broadcasts by each node for one hour and the number of successful broadcasts out of them plus the success rate. Figure 5.1 shows the diagram of the corresponding success broadcast rate. Success rate of node 1, 2, 6 and 7 are low because they do not have four neighbor. All nodes 3, 4, and 5 have 4 neighbors each, hence their success rate is high of the broadcasted messages. Information shows average 50% success rate of the broadcasted messages to neighbors.

Table 5.2: Distribution of the broadcasted packets

Node	Previous of Previous	Previous	Next	Next of Next
1	0	0	368	316
2	0	316	260	442
3	258	390	312	490
4	354	186	402	244
5	138	412	174	358
6	344	482	266	0
7	402	256	0	0

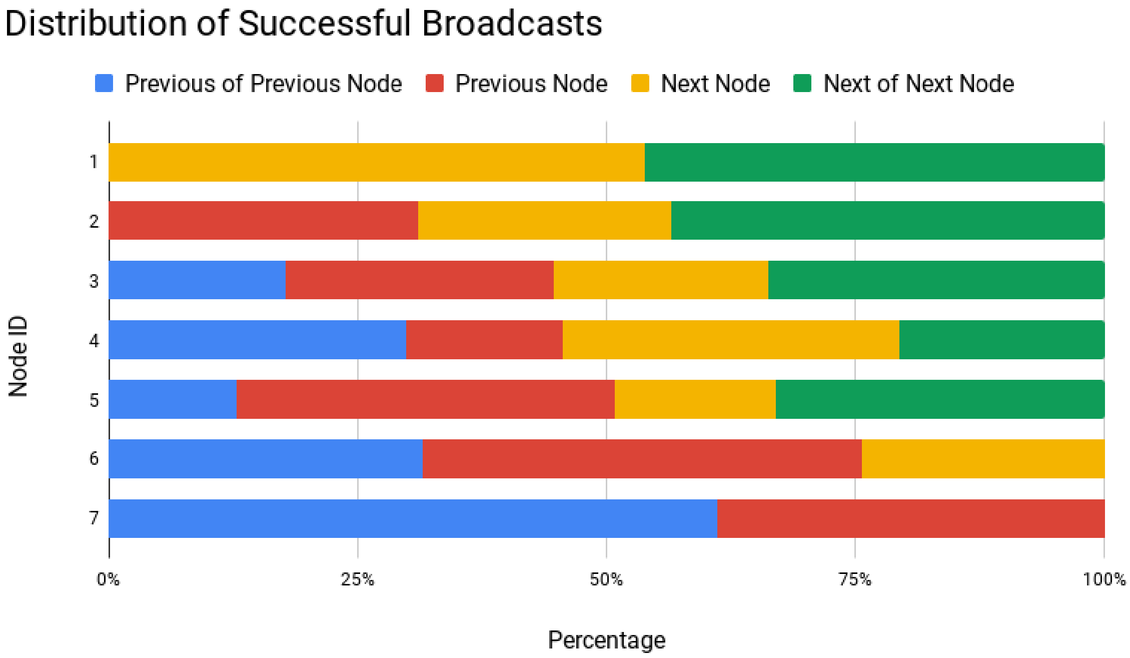


Figure 5.2: Percentages of the distribution of the success broadcasts among neighbors

Table 5.2 shows who received the successful broadcasts. It shows how much is received by the previous of previous node, previous node, next node, and the next of next node. Figure 5.2 shows the data in table 5.2 in percentage. In average it shows that the successful broadcasts are equally distributed among the neighbors.

Table 5.3: Average time gap of successful broadcasts (in seconds)

	TO						
FROM	1	2	3	4	5	6	7
1		12.8	21.3				
2	15.7		28.8	10.7			
3	14.8	9.5		13.9	9.5		
4		11.7	23.9		11.4	19.9	
5			29.4	10.7		26.5	11.0
6				12.6	8.8		12.5
7					15.0	20.5	

Table 5.4: Maximum time gap of successful broadcasts (in seconds)

	TO						
FROM	1	2	3	4	5	6	7
1		47.0	84.0				
2	73.0		160.0	47.0			
3	68.0	29.0		66.0	39.0		
4		43.0	144.0		51.0	113.0	
5			140.0	59.0		125.0	48.0
6				38.0	34.0		42.0
7					53.0	86.0	

Table 5.3 shows the average time of successful broadcast from every node to its every neighbor in seconds. Table 5.5 shows the maximum time of successful broadcast from every node to its every neighbor in seconds. Mean value is 14.00, median value is 11, mode value is 3, standard deviation is 12.41 , maximum value is 160 and the minimum value is 1 of all the time gaps of successful broadcasts. Figure 5.3 shows the normal distribution of successful broadcasting time gaps of all nodes. The peak of figure 5.3 is 14, which means most of the nodes have a 14 second time gap between successful broadcasts.

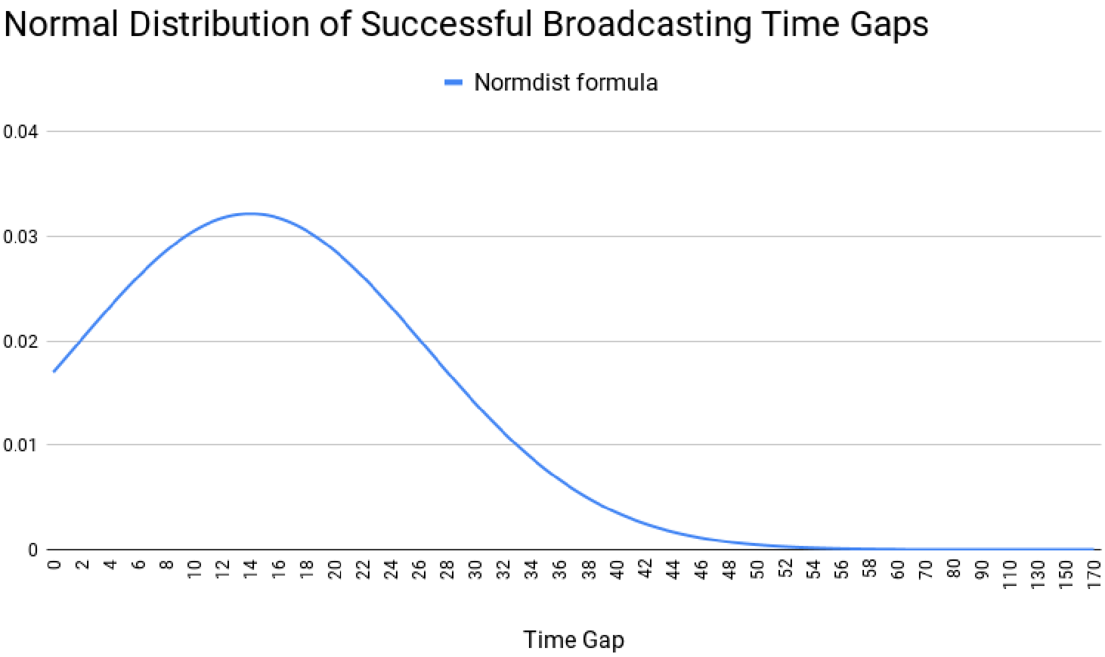


Figure 5.3: Normal distribution of successful broadcasting time gaps

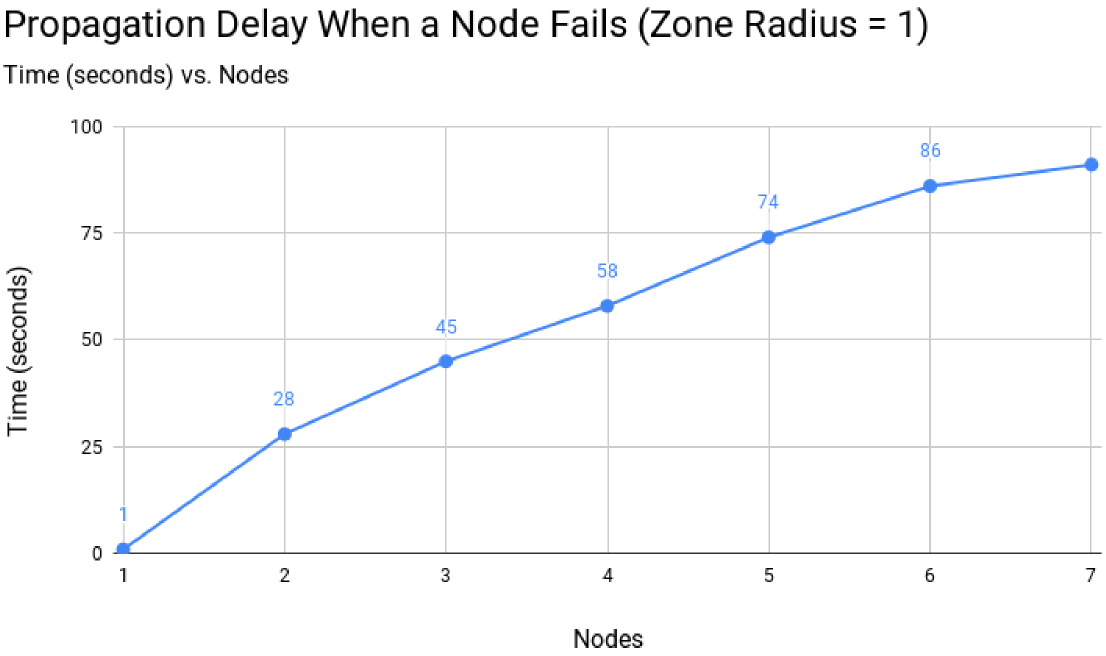


Figure 5.5: Message propagation delay when a node fails (Zone radius = 1)

When calculating the message propagation delay, for each test, 20 trials were carried out and the average is taken. Figure 5.5 shows the time taken to propagate the message when a node fails in case like the solar panel stops working in a node arrangement with zone radius 1. It shows that for node-01 it takes 1 second, for node-02 it takes 45 seconds, so on. According to the results the equation for the n^{th} node is 15.3*n - 5.71. So approximately if there are 200 nodes and if the node-200 fails it takes 50 minutes to get informed. Figure 5.6 shows the time taken to propagate the message when a node fails when the zone radius is 2. According to the graph, the time taken for the n^{th} node is given by the below equation. Approximately if there are 200 nodes and if the node-200 fails, it takes 25 minutes to get informed.

$$\text{Delay for } n^{th} \text{ node} \begin{cases} 1 \\ 15.3*(n/2)-5.71 \\ 15.3*((n-1)/2)-5.71 \end{cases}$$

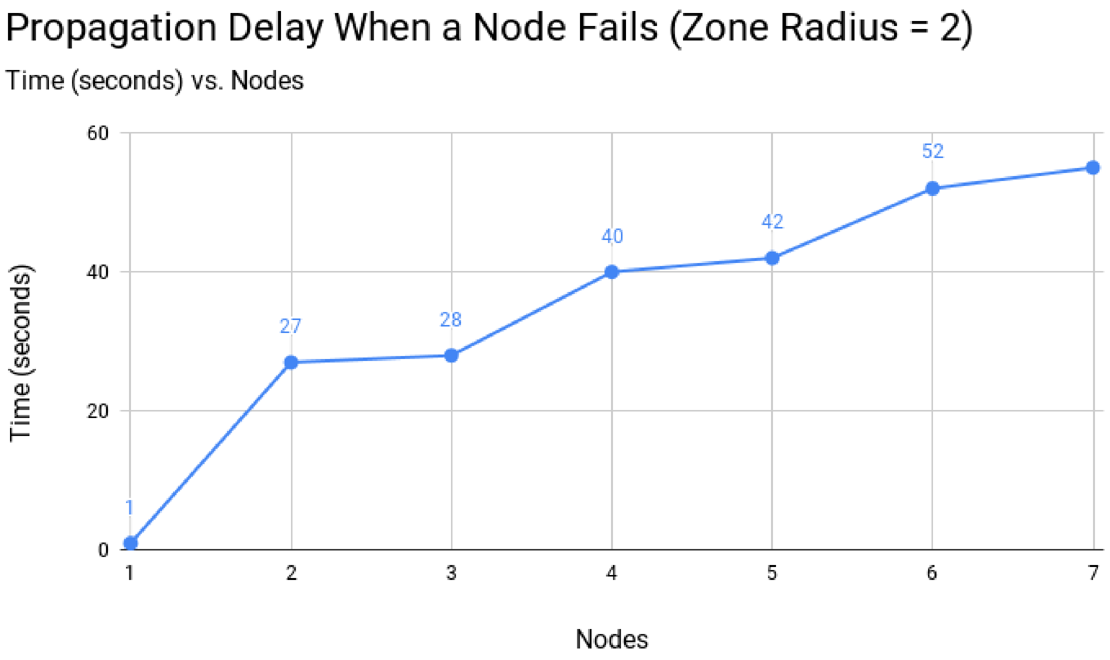


Figure 5.6: Message propagation delay when a node fails (Zone radius = 2)

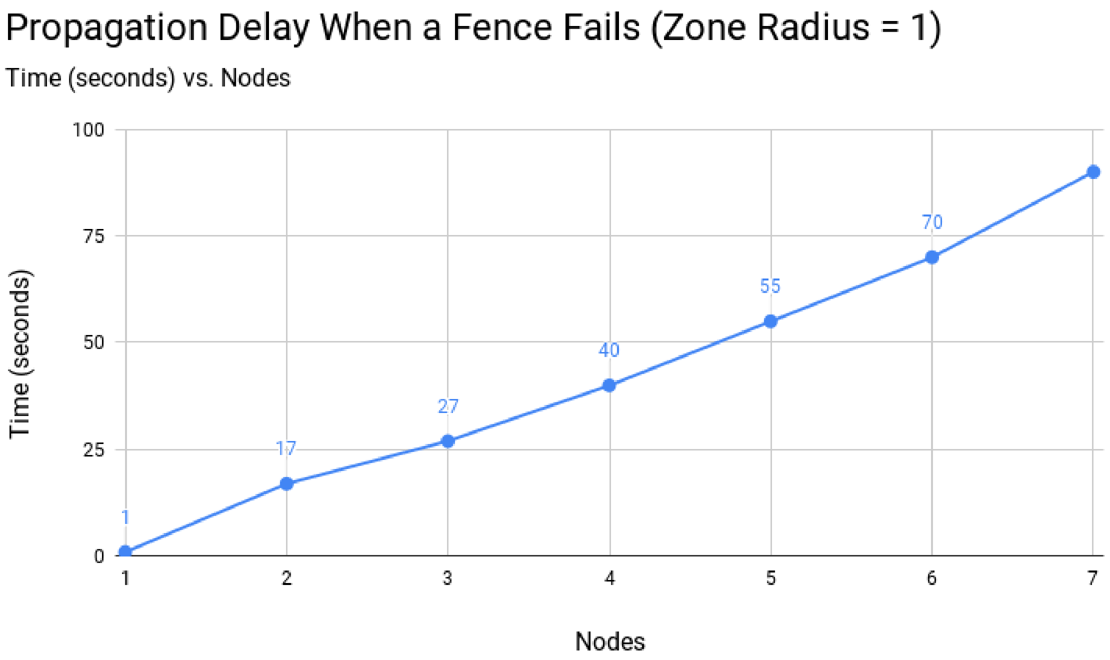


Figure 5.7: Message propagation delay when a fence fails (Zone radius = 1)

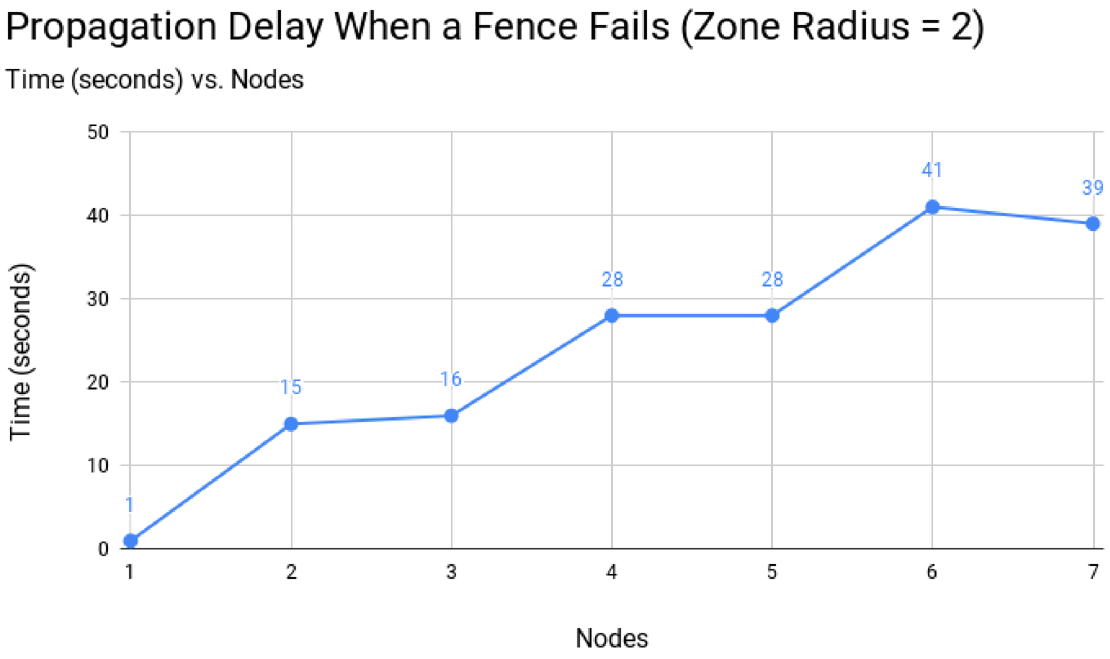


Figure 5.8: Message propagation delay when a fence fails (Zone radius = 2)

Figure 5.7 and figure 5.8 show the time taken to propagate the message when a fence breakage is identified at a node. Not like the above mentioned scenario, in this case the node is working fine and the node detects the fence is broken at its point. Figure 5.7 shows results when the nodes are arranged in a way with zone radius 1 and figure 5.8 shows the results when the nodes are arranged with zone radius 2. For zone radius 1, the equation to calculate the propagation delay of n^{th} node is 14.3*n - 14.4. When zone radius is 2 the delay of n^{th} node is given by the below equation. For a sensor network with 200 nodes with zone radius 1, it will take about 47 minutes to identify when the fence is broken at node-200. For a network with zone radius 2 it will take about 24 minutes to identify the breakage at node-200.

$$\text{Delay for } n^{th} \text{ node} \begin{cases} 1 \\ 14.3*(n/2)-14.4 \\ 14.3*((n-1)/2)-14.4 \end{cases}$$

Working time of the nodes were evaluated by placing the nodes in outside and by measuring the maximum distance two nodes can be kept when their radio ranges overlap in a way that they can communicate. Figure 5.9 shows the average time of 20 days which node works. The nodes start working at about 7.30 AM when the sun rises and stops working at about 5.30 PM when the sun goes down. These results were taken by testing the nodes for 10 days in November 2018 and 10 days in December 2018. The time does not highly affect the distance of the nodes. It's always like 135 meters. Figure 5.10 shows the maximum distance of the nodes with the intensity level of light. To measure intensity of light a light intensity meter is needed. A light intensity meter could not be found and it cost a lot to buy one, so the light intensity was measured using a 7mm LDR. When the light intensity is high the resistance of the LDR goes down and when the light intensity is low the resistance of the LDR goes up. The graph in figure 5.10 is drawn with the light intensity measured with the resistance value of LDR. According to this graph a light intensity level of 120 ohm is needed to power the node up and the maximum distance which two nodes can be kept is 135 meters.

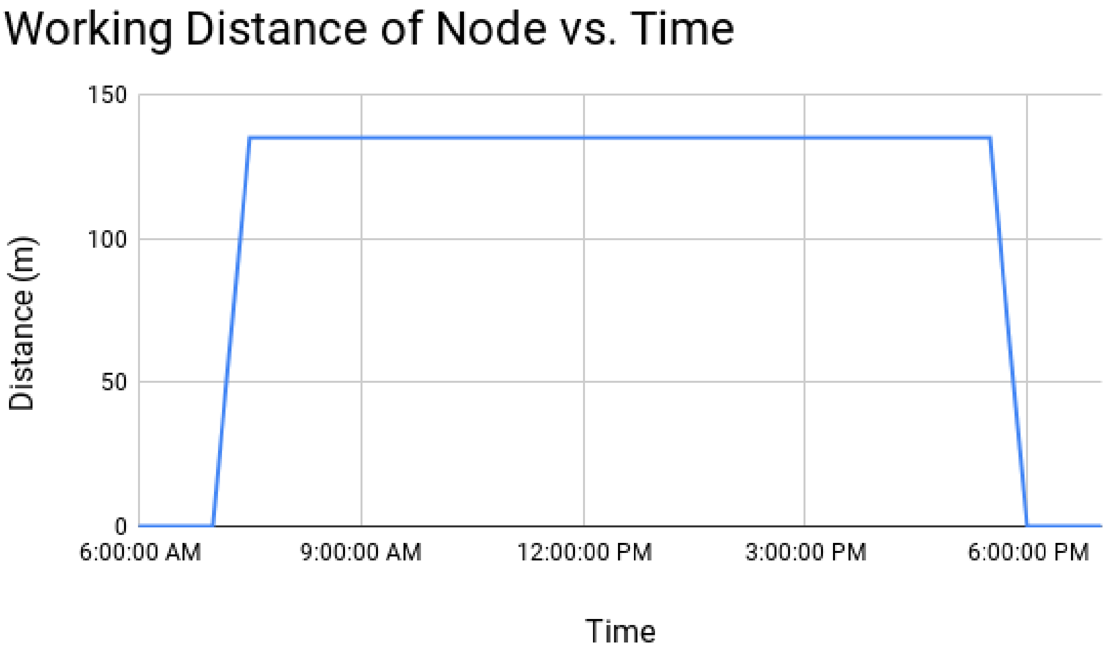


Figure 5.9: Working distance of node with time

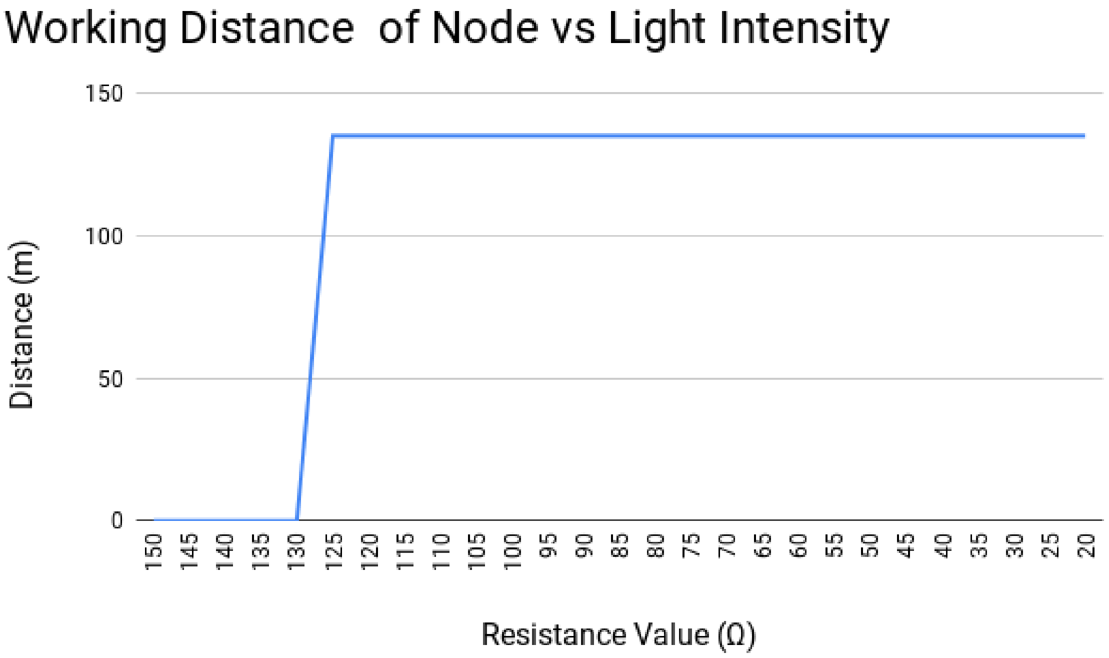


Figure 5.10: Working distance of node with light intensity

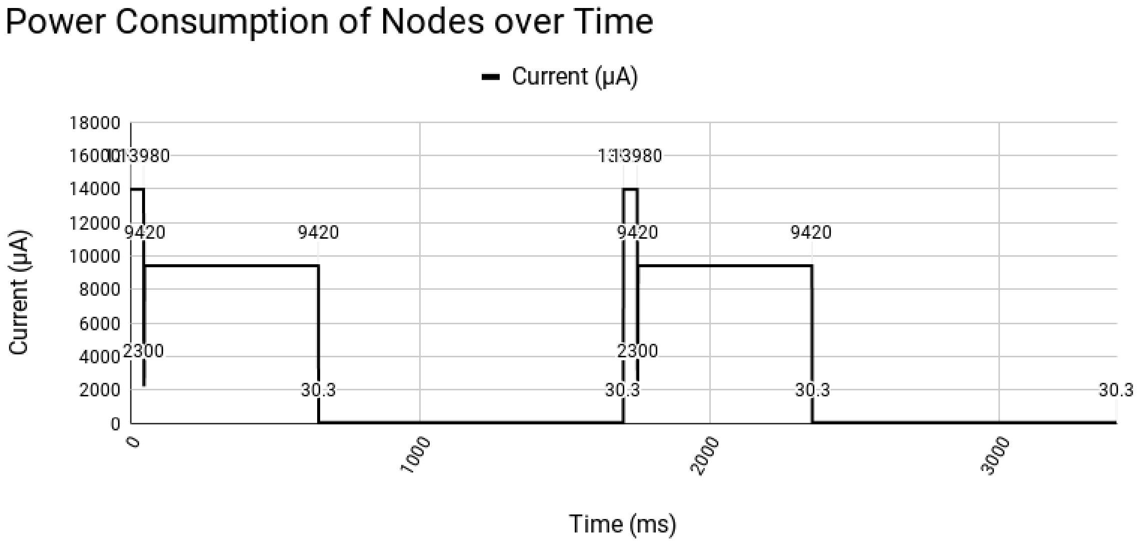
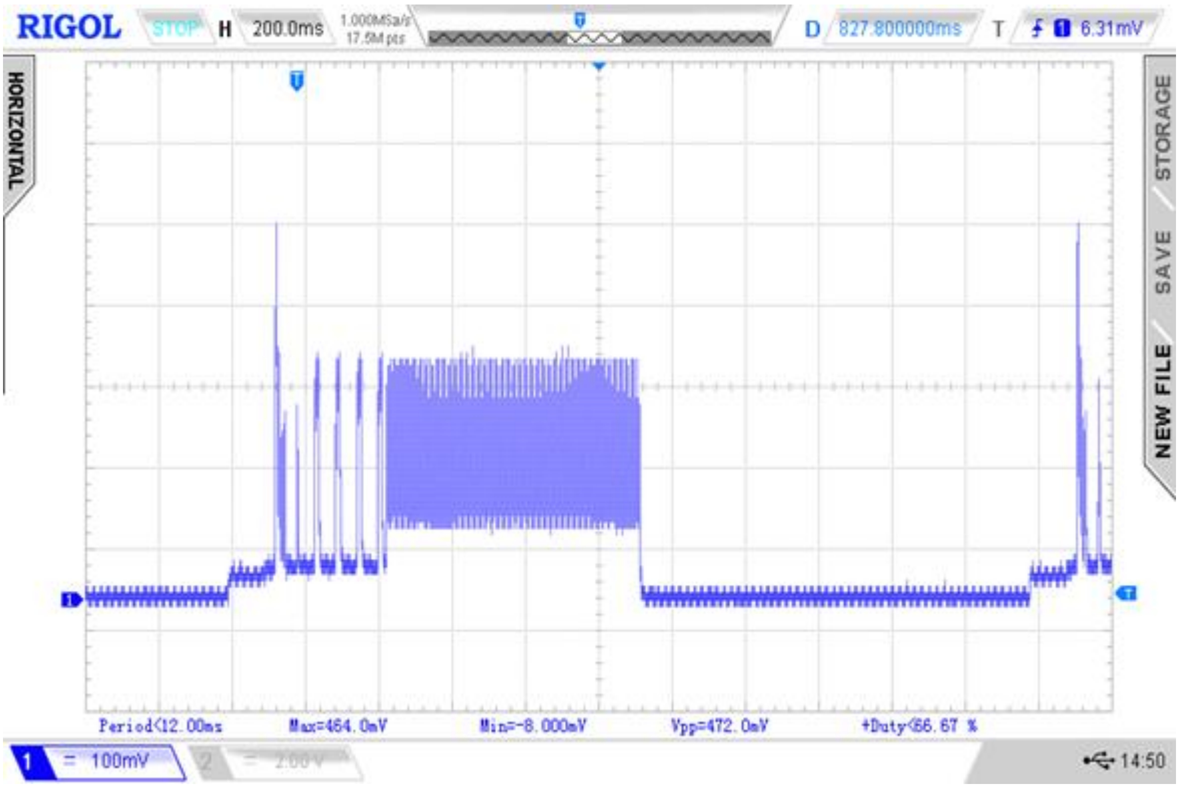


Figure 5.11: Power analysis of the node



Period<12.00ms Max=464.0mV Min=-8.000mV Vpp=472.0mV +Duty<66.67 %

Figure 5.12: Oscilloscope readings of the node

Finally a power analysis was done using an oscilloscope with a shunt resistor of 33 ohms. The oscilloscope reading is shown in figure 5.12. In this figure the high peak is when the node broadcasts a message, lower area shows power consumption when node is in the deep sleep state, and the rest shows the power consumption when the node is working which is listening for other nodes messages. The recalculated power analysis for the figure 5.12 is shown in figure 5.11. According to the above graph the node works for 600 milliseconds and sleeps for 1050 milliseconds. During deep sleep it consumes 30.3µA, during working period it consumes 9.42mA and during broadcasting it consumes 13.98mA. According to the above graph the duty cycle of the node is 37.5%.

Final Results

With the above evaluation, the best configuration for the nodes to arrange is with zone radius 2, since if a node fails to work, the whole system will not go down. If the nodes are arranged with zone radius 1, long fence coverage can be achieved but the reliability is low since, if a node fail to work the system will not identify breakages of the fence from the failed node onwards. Hence node configuration with zone radius 2 is better than zone node configuration with zone radius 1. Even though the configuration of nodes with zone radius 3 is more reliable than the configuration with zone radius 2, zone radius 3 is not appropriate since it costs a lot because it requires more nodes to cover an area than the nodes needed to cover the area with zone radius 2. With the node configuration of zone radius 2 and by using the maximum supported nodes by the ‘Dead or Alive’ protocol which is 240, we can cover an electric fence with length 14.5 Km. In a sensor system with 240 nodes with the above mentioned calculation we can get to know the status of the fence at node-240 within 30 minutes. Unlike state-of-the-art systems which is described in section 2, the proposed system does not have any kind of battery built in, also the system is separated from the high voltage electric fence in hardware level as elaborated in the section 3. Hence the maintenance cost of the proposed system is very low compared to other state-of-the-art systems.

5.3 Conclusion

This chapter elaborated on the experimental results and the evaluation of the proposed system. First it explained the parameters which changes the behavior of the Dead or Alive protocol. Then evaluated the protocol by analyzing the captured traffic of the proposed system. Then different propagation delays of messages of the protocol is elaborated. Then the working time and the working conditions of the prototype nodes were evaluated. Then a power consumption analysis of the prototype node was done. Finally the best configuration for the designed protocol was identified.

Chapter 6 - Conclusions

6.1 Introduction

This chapter includes a review of the research aims and objectives, research problem, limitations of the current work and implications for further research.

6.2 Conclusions about Research Questions (Aims/Objectives)

This research has addressed the question "how to design a batteryless sensor system for breakage detection in electric fence?" To get an answer for this question the research came up with a batteryless wireless sensor network with less maintenance. This is focused on maintenance purposes and warning is not a main consideration. Wildlife officers are alerted of possible breakages with location details. The wildlife offices maintain the fences during the day time. The aim of this research is to reduce the time period at which the fence is down, by sending the exact location where the fence is damaged during day-time, to the relevant authorities, and speeding up the process of search and repair, thus reducing the danger of elephants getting through the fence. Reducing the maintenance cost of the fence is also an aim. Objective of this research is to design a reliable, robust, less-costly batteryless electric fence breakage detection system without the need for battery replacement.

Existing breakage detection systems need batteries to power their systems up, which needs costly maintenance after the system is deployed. Some systems [7, 8, 4] use rechargeable batteries, these rechargeable batteries also die after some time, and again they need a battery replacement. For the proposed system, a multihop, simple message, energy efficient protocol is needed, but out of the WSN communication protocols stated in the section 2.3, there is no suitable protocol which satisfies all the requirements.

6.3 Conclusions about Research Problem

The proposed system in this research is a batteryless sensor system which addresses the above question. It is made energy efficient by designing a novel wireless communication protocol known as “Dead or Alive Protocol” which runs on the node. It is designed by combining two existing wireless communication protocols which are Gossip protocol and the T-MAC protocol with some additional features (see section 3.2). This protocol helps the designed hardware to operate without batteries, but with a small. As the energy source, a small solar panel is used stated in section 4.

The proposed system in this research can be easily installed to an existing electric fence without any modification to the fence. Once it is installed, it starts working in daytime approximately from (7.30 AM to 5.30 PM) and checks for breakages. If a breakage is detected by a node in the system, system informs the wildlife offices the location where the breakage has happened. With the results of the chapter 5 the proposed system can cover 12 km electric fence with zone radius 2 node arrangement and with 100 meters node distance. And the system will report the statuses of all nodes within 30 minutes.

The proposed system does not have any kind of battery built in. The system is separated from the high voltage electric fence in the hardware level as elaborated in the section 3.3. Hence the maintenance cost of the proposed system is very low compared to other state-of-the-art systems.

6.4 Limitations

As outlined in section 1.9, the limitation of this system is in the designed “Dead or Alive” protocol, it can only support up to 240 nodes due to packet size of the message. In the evaluation part in chapter 5, the maximum distance of two nodes is identified as 135 m. With 240 nodes a maximum distance of 14.5 km. As stated in [2], the maximum length of currently deployed electric fence is about 40 km. Therefore, to cover an electric fence which is longer than 12 km, a single system is not sufficient. Either three systems should be used, or a modification to the WSN communication protocol should be done to support more than 240 nodes. Therefore this is a limitation of the proposed system.

6.5 Implications for Further Research

As mentioned in the section 6.4, the number of nodes which the protocol supports is limited to 240. Investigations should be carried out to increase this limit by changing the proposed “Dead or Alive” protocol.

The proposed sensor system works only during the day time. If authorities wants to get the status of the electric fence during night-time, a power source to work during night-time should be found. This can be achieved by scavenging energy from the fence which goes through the high voltage lines in the electric fence by induction or by some other mean.

References

[1] L. Gunaratne, P. Premarathne, et al. The effectiveness of electric fencing in mitigating human-elephant conflict in sri lanka. Technical report, Economy and Environment Program for Southeast Asia (EEPSEA), 2006.

[2] J. Kioko, P. Muruthi, P. Omondi, and P. I. Chiyo. The performance of electric fences as elephant barriers in amboseli, kenya. South African Journal of Wildlife Research, 38(1):52–58, 2008.

[3] P. Fernando. Managing elephants in sri lanka: where we are and where we need to be. Ceylon Journal of Science (Biological Sciences), 44(1), 2015.

[4] SCORE lab University of Colombo School of Computing: (SENSOR BASED FENCE BREAKAGE SYSTEM) “http://www.scorelab.org/ASSET/” , “https://goo.gl/D4bn2u”

[5] J. Eastwood, (Fence Alarm) Mar. 9 2010. US Patent 7,675,417.

[6] L. Wijesinghe, P. Siriwardena, S. Dahanayake, D. Kasthurirathne, R. Corea, and D. Dias. Electric fence intrusion alert system (eleAlert). In 2011 IEEE Global Humanitarian Technology Conference, 2011.

[7] E. Tennakoon, C. Madusanka, K. De Zoysa, C. Keppitiyagama, V. Iyer, K. Hewage, and T. Voigt. Sensor-based breakage detection for electric fences. In Sensors Applications Symposium (SAS), 2015 IEEE, pages 1–4. IEEE, 2015.

[8] Namal Jayasuriya, Asanka P. Sayakkara, Chathura Suduwella, Chamath Keppitiyagama, Kasun De Zoysa, Kasun Hewage. Wired-backscatter Communication for Breakage Detection in Electric Fences “Wire is not dead”. In EWSN ’17 Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks. Pages 300-304

[9] Journal of Wildlife Research, 38(1):52–58, 2008. Q. Shi and O. Kanoun. A new algorithm for wire fault location using time-domain reflectometry. IEEE Sensors Journal, 14(4):1171–1178, 2014.

[10] K. Jehan Silva, Article “Elephant deaths in Sri Lanka” DOI: http://elephantcare.org/humanele.htm#Humans%20deaths

[11] "Survey of MAC Protocol for Wireless Sensor Networks - IEEE Conference Publication", Ieeexplore.ieee.org, 2015. [Online]. DOI:

https://ieeexplore.ieee.org/document/7306657/. [Accessed: 30- Jun- 2018].

[12] "The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey - IEEE Journals & Magazine", Ieeexplore.ieee.org, 2013. [Online]. DOI: https://ieeexplore.ieee.org/document/6188353/?arnumber=6188353. [Accessed: 30- Jun- 2018].

[13] Wei Ye, John Heidemann, Deborah Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", Information Science Institute (ISI), University of Southern California (USC).2002, pp.1-10.

[14] Xie Hong, Zhou Jie, "Research of A New Messages Disseminatin Algorithm Based on Structured Gossip protocol", 2010 International Conference on Electrical and Control Engineering

[15] Tijs van Dam, Koen Langendoen, "An Adaptive Energy Efficient MAC Protocol for Wireless Sensor Networks", SenSys03 Los Angeles, California USA, 2003 pp.1-6.

[16] Changsu Suh, Young-Bae Ko, "A Traffic Aware, Energy Efficient MAC Protocol for Wireless Sensor Networks", Korea Science & Engineering Foundation, University IT Research Centre Project and a grant of the 21C Frontier R&D Program, pp. 1-4.

[17] A. El-Hoiydi, "Spatial TDMA and CSMA with preamble sampling for low power ad hoc wireless sensor networks", Proceedings of ISCC 2002, Seventh International Symposium on Computers and Communications,1- 4 July 2002, pp.685 - 692.

[18] C. C. Enz, A. El-Hoiydi, J-D. Decotignie, V. Peiris, "WiseNET: An Ultralow-Power Wireless Sensor Network Solution", IEEE Computer, Volume: 37, Issue: 8, August 2004, pp.244-251.

[19] L. Tang, Y. Sun, O. Gurewitz, and D. Johnson. "EM-MAC: A Dynamic Multichannel Energy- Efficient MAC Protocol for Wireless Sensor Networks", In The Twelfth ACM International Sympo- sium on Mobile Ad-Hoc Networking and Computing (MobiHoc11), Paris, May 2011, pp.1-11.

[20] M. Anwander, G. Wagenknecht, T. Braun, and K. Dolfus. "BEAM: A Burst-Aware Energy-Efficient Adaptive MAC Protocol for Wireless Sensor Networks", In Seventh International Conference on Networked Sensing Systems (INSS10) Kassel, Germany, June 2010, pp.195-202.

[21] Samira Yessad,Farid Nait-Abdesselam, Tarik Taleb ,Brahmin Bensaou, "R-MAC: Reservation Medium Access Control Protocol for Wireless Sensor Networks", 0742-1303/07 DOI 10.1109/LCN.2007.159.

[22] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks”, In SenSys, 2004, pp 1-6.

[23] M. Buettner, G. V. Yee, E. Anderson, and R. Han. “X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks”, In SenSys, 2006, pp.307-320.

[24] Nordic Semiconductor ASA - Vestre Rosten 81, N-7075 Tiller, Norway, “Single chip 2.4 GHz Transceiver” nRF24L01 datasheet, Mar. 2006.

[25] Atmel, “Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash” ATtiny85 datasheet [Rev. 2586Q–AVR–08/2013]

[26] D. Parson, P 2013. “A nRF24L01+ library with AVR 2 pin support”. DOI: https://github.com/dparson55/NRFLite [Release 2.0.9 13/10/2018]

Appendix A: Publications

Appendix B: Diagrams

Prototype nodes of the designed system

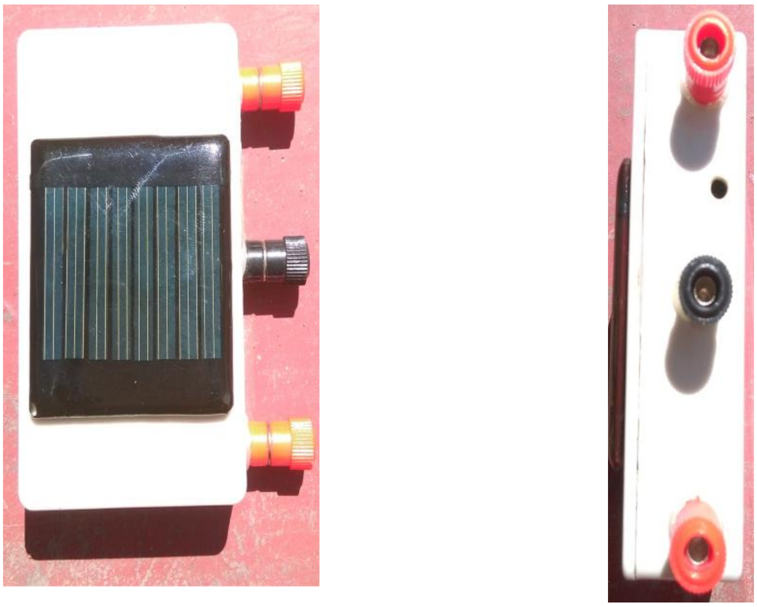


Figure A.1 Top view and side view of the designed prototype node.

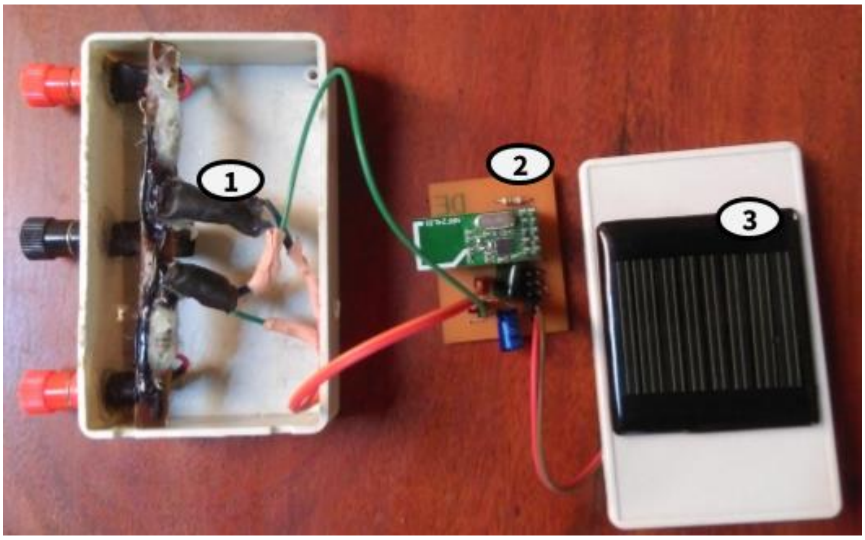


Figure A.2 Inside of the designed prototype node

Figure A.1 shows the top and side view of the designed prototype of the batteryless sensor node. In the top view the solar panel is mounted in the top of the node. Figure A.2 shows the inside of the prototype node. From number 1 it shows the way the LDR and the neon light is glued and insulated so no outside light can change the resistance of the LDR, only the light from the neon bulb can change the resistance of the LDR. Number 2 shows the circuit of the node. The green color module is the radio module the blue color component is the capacitor which charges from the solar panel denoted by number 3.

System for the base station

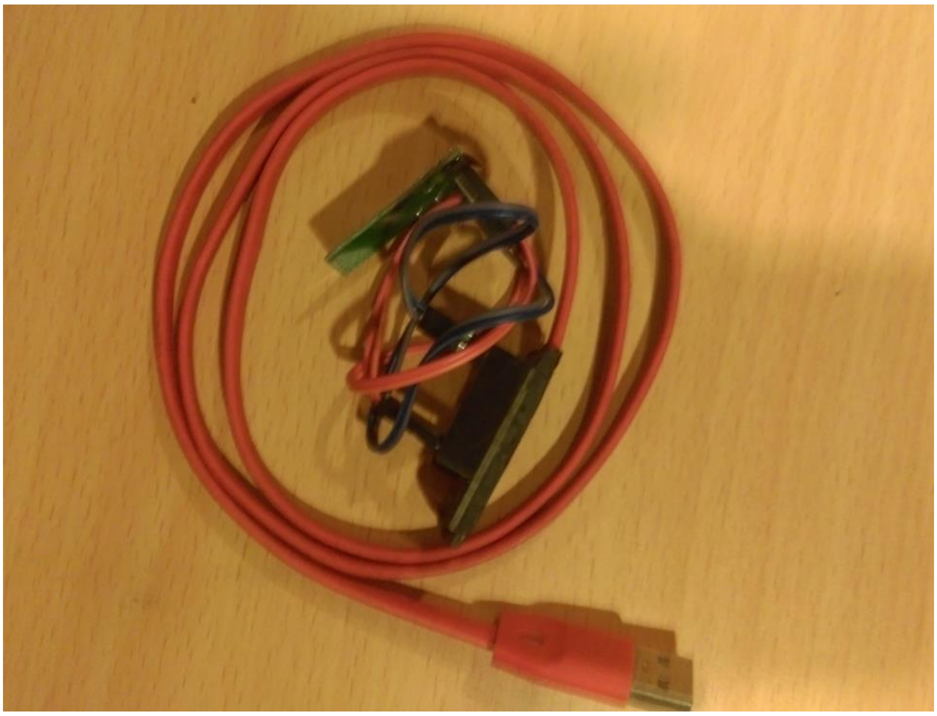
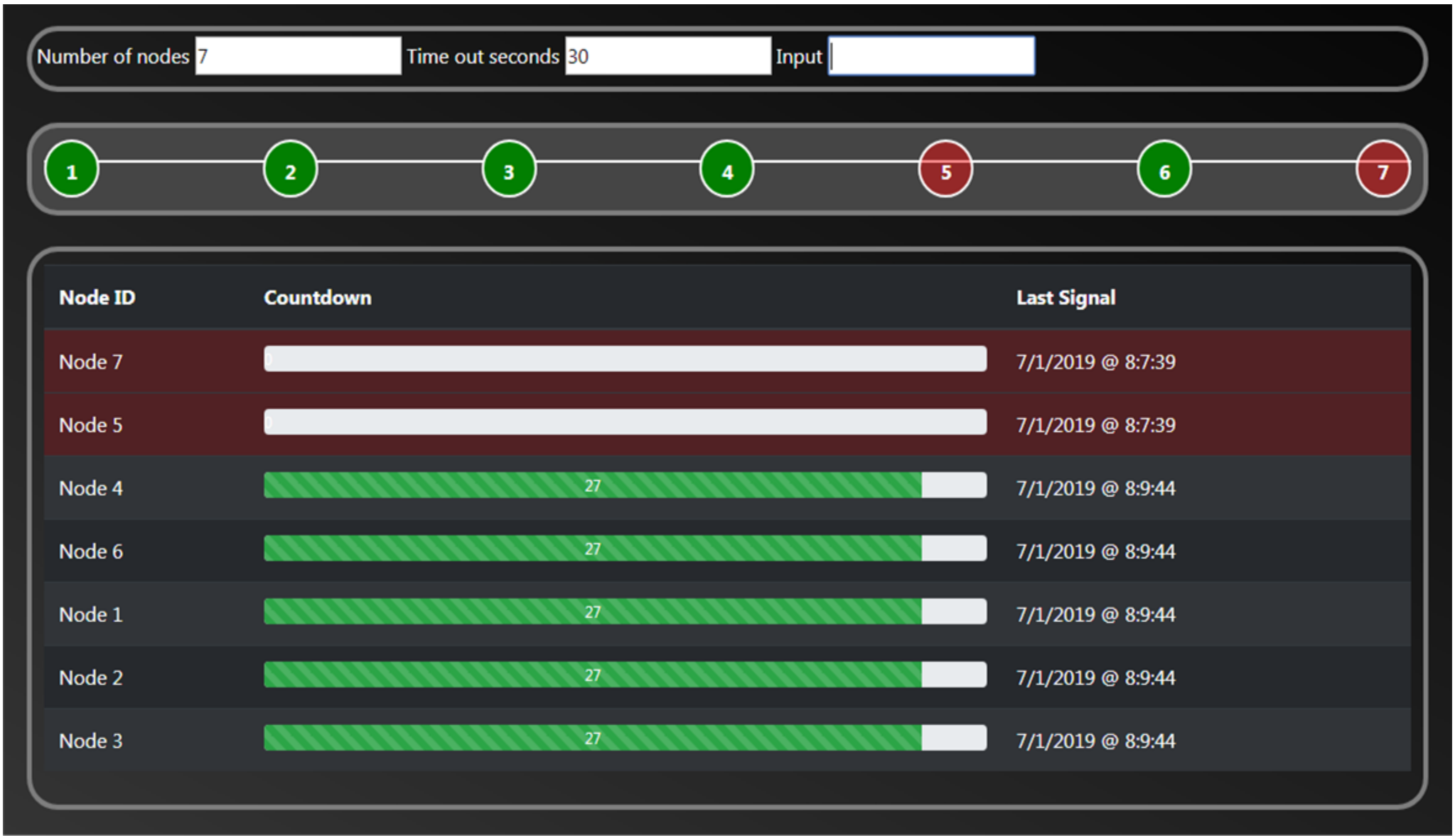


Figure A.3 Hardware component of the base station system

Figure A.3 shows the hardware component designed to install in the base station. It captures the broadcasted packets form the nodes and give the output as a data string to the computer. This device works as a keyboard. After connecting the device to the computer user has to open the web GUI shown in the figure A.4 and point the cursor to the input field named in ‘Input’. Number of nodes in the system is set in the ‘Number of nodes’ field and the time out is set in seconds in the ‘Time out seconds’ field. When the device detects broadcasts from the nodes it is typed in the ‘Input’ field. The node positions are shown in circles and node data is shown down it. The working nodes are shown in green color while the nodes which are not working blinks in red. In the ‘Last Signal’ column it shows the last time which a signal got from that node and the table is sorted in this column where the not working nodes come to top, therefore it is easy to identify the nodes which are not working.

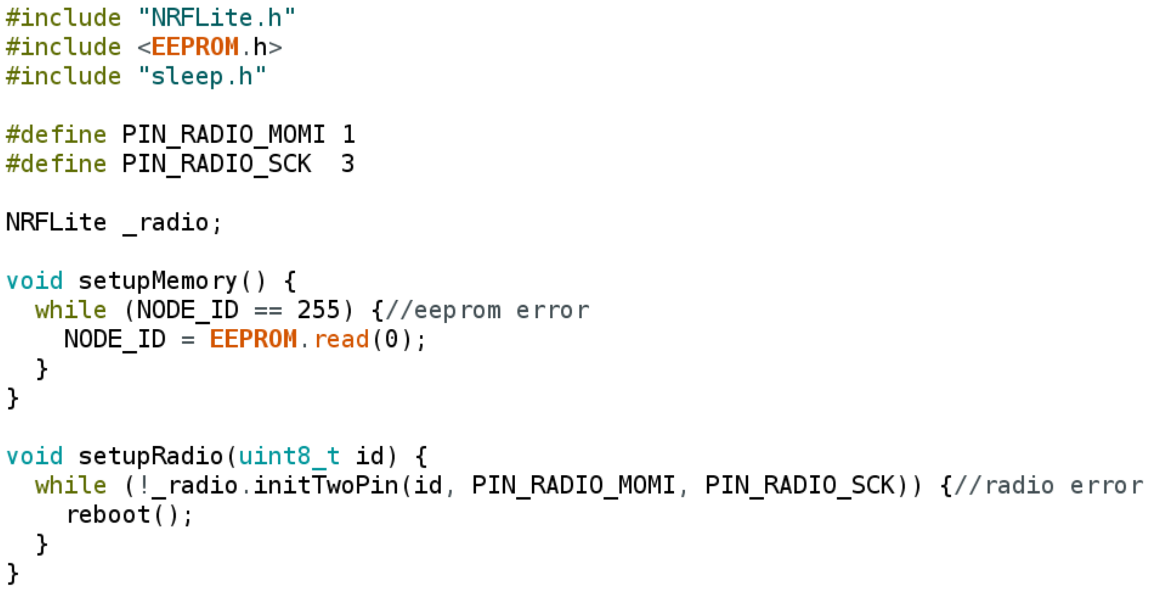


Number of nodes 7 Time out seconds 30 Input

Node ID	Countdown	Last Signal
Node 7		7/1/2019 @ 8:7:39
Node 5		7/1/2019 @ 8:7:39
Node 4	27	7/1/2019 @ 8:9:44
Node 6	27	7/1/2019 @ 8:9:44
Node 1	27	7/1/2019 @ 8:9:44
Node 2	27	7/1/2019 @ 8:9:44
Node 3	27	7/1/2019 @ 8:9:44

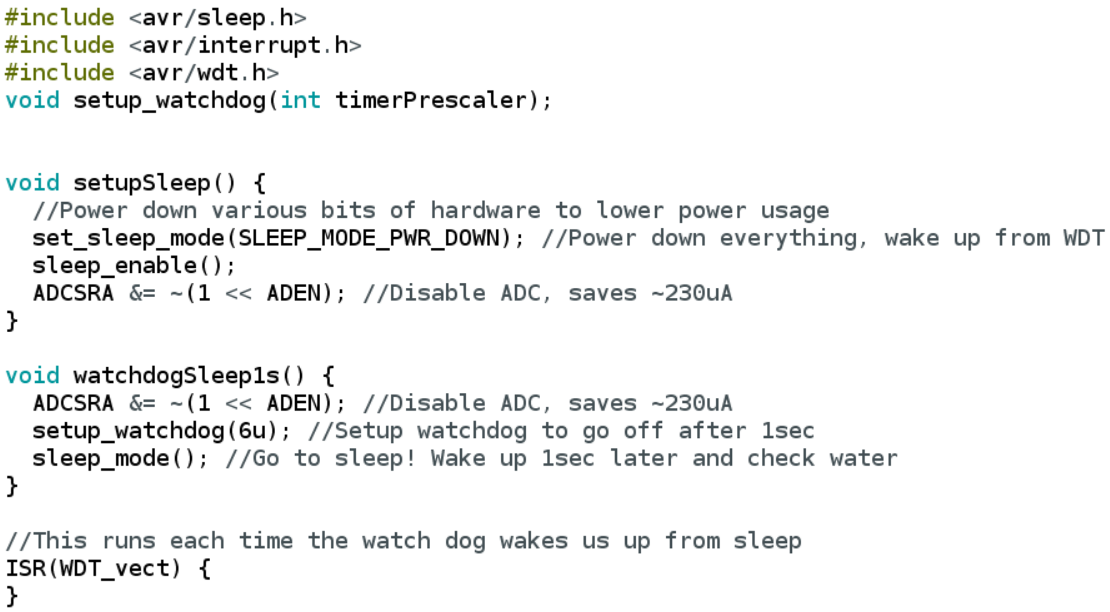
Figure A.4 Web GUI of the base station system

Appendix C: Code Listings

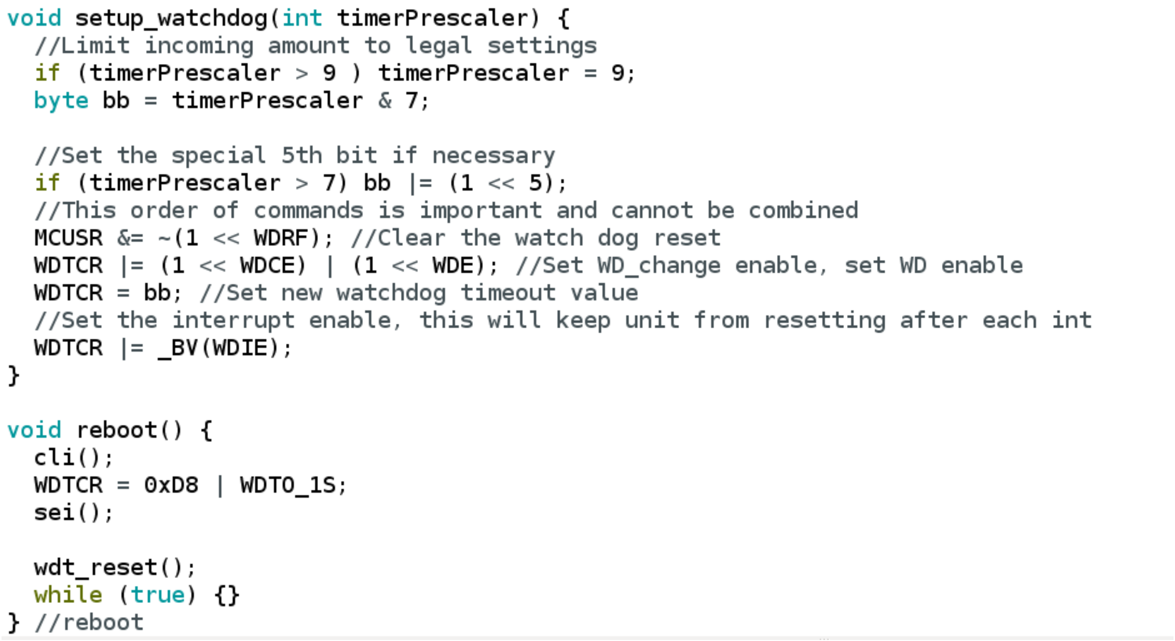


```
#include "NRFLite.h"
#include <EEPROM.h>
#include "sleep.h"

#define PIN_RADIO_MOMI 1
#define PIN_RADIO_SCK  3

NRFLite _radio;

void setupMemory() {
  while (NODE_ID == 255) {//eeprom error
    NODE_ID = EEPROM.read(0);
  }
}

void setupRadio(uint8_t id) {
  while (!_radio.initTwoPin(id, PIN_RADIO_MOMI, PIN_RADIO_SCK)) {//radio error
    reboot();
  }
}
```

Above code is in the "radio.h". setupMemory() method initializes the EEPROM of the microcontroller and retrieve the node ID from it and assigns it to the NODE_ID global variable. setupRadio() method configure the radio and assigns the radio ID to the radio. After assigning a radio ID, the radio only listens for messages which comes for that specific radio ID.



```
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <avr/wdt.h>
void setup_watchdog(int timerPrescaler);


void setupSleep() {
  //Power down various bits of hardware to lower power usage
  set_sleep_mode(SLEEP_MODE_PWR_DOWN); //Power down everything, wake up from WDT
  sleep_enable();
  ADCSRA &= ~(1 << ADEN); //Disable ADC, saves ~230uA
}

void watchdogSleep1s() {
  ADCSRA &= ~(1 << ADEN); //Disable ADC, saves ~230uA
  setup_watchdog(6u); //Setup watchdog to go off after 1sec
  sleep_mode(); //Go to sleep! Wake up 1sec later and check water
}

//This runs each time the watch dog wakes us up from sleep
ISR(WDT_vect) {
}
```



```
void setup_watchdog(int timerPrescaler) {
  //Limit incoming amount to legal settings
  if (timerPrescaler > 9 ) timerPrescaler = 9;
  byte bb = timerPrescaler & 7;

  //Set the special 5th bit if necessary
  if (timerPrescaler > 7) bb |= (1 << 5);
  //This order of commands is important and cannot be combined
  MCUSR &= ~(1 << WDRF); //Clear the watch dog reset
  WDTCR |= (1 << WDCE) | (1 << WDE); //Set WD_change enable, set WD enable
  WDTCR = bb; //Set new watchdog timeout value
  //Set the interrupt enable, this will keep unit from resetting after each int
  WDTCR |= _BV(WDIE);
}

void reboot() {
  cli();
  WDTCR = 0xD8 | WDTO_1S;
  sei();

  wdt_reset();
  while (true) {}
} //reboot
```

Above code is in the “sleep.h”. In setupSleep() method it setups the watchdog timer to trigger after 1 second. In watchdogSleep1s() method the system sleeps for 1 second. reboot() method reboots the system.