

Process of developing a mesh network with Raspberry Pi in wooded areas



A Final year project Submitted Towards Consideration
for a Bachelor of Engineering

Author

Liam Hogan

Supervisor

Philip Creevy

South East Technological University
Department Of Engineering Technology
School of Engineering
Ireland.
May 20, 2024

Contents

| | |
|---|-----------|
| List of Figures | 3 |
| List of Tables | 4 |
| 1 Introduction | 1 |
| 1.1 Introduction | 1 |
| 1.2 Motivation | 1 |
| 2 Literature Review | 2 |
| 2.1 Introduction | 2 |
| 2.1.1 Overview | 2 |
| 2.1.2 Mesh network | 4 |
| 2.2 Hardware Consideration | 5 |
| 2.2.1 Sensor considerations | 6 |
| 2.2.2 Radio Module | 13 |
| 2.2.3 ADC Considerations | 13 |
| 2.2.4 Camera | 14 |
| 2.2.5 Memory module | 15 |
| 2.2.6 Battery | 16 |
| 2.2.7 Arduino vs PI Consideration | 16 |
| 2.2.8 Conclusion | 17 |
| 2.3 Software considerations | 19 |
| 2.3.1 Raspberry Pi OS | 19 |
| 2.3.2 Sensor code | 19 |
| 2.3.3 MCP3008 | 22 |
| 2.3.4 Raspberry Pi VR 220 Camera | 23 |
| 2.3.5 MM2 Series 900 MHz | 23 |
| 2.3.6 code structure | 23 |
| 2.3.7 File structure | 23 |
| 2.3.8 Test Driven development | 25 |
| 2.4 Attenuation | 25 |
| 2.4.1 Absorption of water | 27 |
| 2.5 mesh network considerations | 28 |
| 2.6 Review key of research Papers | 28 |
| 2.7 Summary | 29 |
| 3 Methodology | 30 |
| 3.1 Introduction | 30 |
| 3.2 Procedure | 30 |
| 3.3 Setup of raspberry pi | 31 |
| 3.4 Additional Research | 31 |

| | | |
|----------|---|-----------|
| 3.4.1 | ADC | 31 |
| 3.4.2 | Radio module | 32 |
| 3.4.3 | What is the difference between a port and a channel | 33 |
| 3.4.4 | Why the MM2 Series 900 MHz wasn't picked | 35 |
| 3.4.5 | SB Components LoRa HAT | 35 |
| 3.5 | Software Module Development | 36 |
| 3.5.1 | Sensors | 36 |
| 3.5.2 | Camera | 39 |
| 3.5.3 | Memory Management | 40 |
| 3.5.4 | Radio module | 41 |
| 3.6 | Data Analysis Methods | 50 |
| 3.6.1 | Met Eireann weather forecast API | 50 |
| 3.6.2 | Matplotlib | 50 |
| 3.7 | Timeline | 50 |
| 4 | Results | 52 |
| 4.1 | Recorded data from sensors | 52 |
| 4.1.1 | DHT22 | 52 |
| 4.1.2 | AS312 | 53 |
| 4.1.3 | DFR0026 | 53 |
| 4.1.4 | Raspberry Pi VR 220 | 54 |
| 4.2 | Recorded data from transceiver | 54 |
| 5 | Discussion | 55 |
| 5.1 | Discussion of results of Sensor data | 55 |
| 5.2 | Discussion of Results of camera data | 55 |
| 5.3 | Discussion of Lora module | 55 |
| 6 | Conclusion & Future Work | 57 |
| 6.1 | Conclusion | 57 |
| 6.1.1 | Key Contributions | 57 |
| 6.2 | Sources of Error | 58 |
| 6.2.1 | Radio Module | 58 |
| 6.2.2 | MCP3008 | 59 |
| 6.2.3 | Lack of knowledge of Linux os | 59 |
| 6.2.4 | Camera | 59 |
| 6.2.5 | Lack of hat for sensors | 59 |
| 6.3 | Future Work | 59 |
| 6.3.1 | Final Remarks | 60 |
| 7 | Appendix | 61 |
| | Bibliography | 61 |
| A | Python Scripts | 62 |
| A.1 | Python Scripts | 62 |
| A.1.1 | DHT22 | 62 |
| A.1.2 | AS312 | 64 |
| A.1.3 | DFR0026 | 65 |
| A.1.4 | Camera | 66 |
| A.1.5 | Memory management | 68 |
| A.1.6 | Radio module | 69 |

| | | |
|----------|-----------------------------|-----------|
| B | TDD Script | 72 |
| B.1 | TDD scripts | 72 |
| B.1.1 | DHT22 | 72 |
| B.1.2 | AS312 | 74 |
| B.1.3 | DFR0026 | 75 |
| B.1.4 | Memory Management | 76 |
| B.1.5 | Radio Module | 77 |
| C | Bash scripts | 78 |
| C.1 | Bashscripts | 78 |
| C.1.1 | Camerea | 78 |
| C.1.2 | Main | 79 |
| C.1.3 | Radio Module | 80 |

List of Figures

| | | |
|------|---|----|
| 2.1 | Basic block diagram of a mesh network | 4 |
| 2.2 | Rough circuit diagram for project | 6 |
| 2.4 | Interface for DFR0026 | 11 |
| 2.5 | Interface for AS312 | 12 |
| 2.6 | Schematic for MCP3008 | 14 |
| 2.7 | Silver Maple in-leaf excess attenuation for the line of trees geometry (receiver antenna height: 3.5 m, SAVAGE ET AL.pg.7 | 26 |
| 2.8 | Specific attenuation due to woodland (Recommendation ITU-R P.833-7 (02/2012) Attenuation in vegetation pg.5 | 26 |
| 2.9 | Predicted attenuation due to rain for the region, which is measured by using the ITU standards,(Source: Hindawi(2014)) | 27 |
| 2.10 | absorption of water | 27 |
| 3.1 | protocol Wu used(wu_lie et.al,2023:16705) | 33 |
| 3.2 | sample graph of a FIR response | 33 |
| 3.3 | Visualized timeline of project files versions | 51 |
| 3.4 | Visualized timeline of project programming timeline | 51 |
| 4.1 | Temperature and Humidity plotted overtime | 53 |
| 4.2 | unit test message for DHT22 module | 53 |
| 4.3 | lux values overtime | 54 |
| 4.4 | A photo from 25th of march 2024 | 54 |
| 5.1 | environment error message | 56 |
| 6.1 | Issue when trying to write to serial | 58 |

List of Tables

| | | |
|------|--|----|
| 2.1 | Highest shader air Met Eireann(13 th June 2023) | 7 |
| 2.2 | Lowest shader air Met Eireann(13 th June 2023) | 7 |
| 2.3 | Realtive Humidity(%) according to met eirrean | 8 |
| 2.4 | Comparing of temperature sensors | 8 |
| 2.5 | Comparing DHT22 and DHT11 | 8 |
| 2.6 | Illuminates values | 10 |
| 2.7 | table of light sensors | 10 |
| 2.8 | Motion sensor components | 11 |
| 2.9 | Radio modules found in research | 13 |
| 2.10 | Camera module | 15 |
| 2.11 | Mirco SDs in consideration | 15 |
| 2.12 | Memory usb to consider | 15 |
| 2.13 | battery considerations | 16 |
| 2.14 | Advantages /Disadvantages of Arduino vs pi | 16 |
| 2.15 | Table of Raspberry Pi's | 17 |
| 2.16 | Advantages and Disadvantages of LoRa and ZigBee | 28 |
| 3.1 | Comparing New Radio modules | 35 |
| 4.1 | Recorded data from DHT22 on the 5 th of march | 52 |
| 4.2 | Recorded data from AS312 on the May 20, 2024 | 53 |
| 4.3 | Recorded data from DFR0026 on the 25th of march 2024 | 53 |

Listings

| | | |
|-----|---|----|
| 2.1 | Example code for DHT2 | 21 |
| 2.2 | Example code for AS312 | 22 |
| 2.3 | ADC code | 22 |
| 2.4 | example code for camera | 23 |
| 2.5 | sample code for turning sensor data into a data | 24 |
| 2.6 | example code for storing directory | 24 |
| 3.1 | sample test intial code | 46 |
| 3.2 | sample test for DHT22 | 47 |
| 3.3 | unit test for DFR0026 and MCP3008 | 47 |
| 3.4 | unit test for AS312 | 48 |
| 3.5 | camera unit test | 48 |
| A.1 | DHT22code | 62 |
| A.2 | code for AS312 | 64 |
| A.3 | Code for DFR00026 | 65 |
| A.4 | Code for Camera | 66 |
| A.5 | Code for alternaive code for Camera | 67 |
| A.6 | Code for memory mangement | 68 |
| A.7 | Code for Radio module | 69 |
| B.1 | DHT22 unit test | 72 |
| B.2 | Code for unit test of AS312 | 74 |
| B.3 | Code for unit test of DFR0026 | 75 |
| B.4 | Code for unit test of memory module | 76 |
| B.5 | unit test code for Radio module | 77 |
| C.1 | Code for triggering the camerea | 78 |
| C.2 | Code for runing the main function | 79 |
| C.3 | Code for the testing serial of the radio module | 80 |

Glossary

| | | | |
|--------------|--|--------------|--|
| APD | Avalanche PhotoDiode | MC | Multiple-Carrier |
| API | Application Programming Interface | MIMO | Multiple Input Multiple Output |
| ASK | Amplitude Shift Keying | MLSE | Maximum Likelihood Sequence Estimation |
| AWG | Agile Waveform Generator | MMF | Multi Mode Fiber |
| B2B | Back-2-Back | MSK | Minimum Shift Keying |
| BBP | Baseband Processor | MSO | Mixed Signal Oscilloscope |
| BER | Bit Error Ratio | MZI | Mach-Zehnder Interferometer |
| BL | Bandwidth-Length | MZM | Mach-Zehnder Modulator |
| BLAST | Bell Labs <u>L</u> ayered <u>S</u> pace <u>T</u> ime | NGPON | Next Generation Passive Optical Network |
| BT | Time Bandwidth Product | NLSE | Non-Linear Schrödinger Equation |
| CD | Chromatic Dispersion | NRZ | Non-Return to Zero |
| CDMA | Code Division Multiple Access | ODN | Optical Distribution Network |
| CPM | Continuous Phase Modulation | OS | operating system (OS) |
| CSI | Channel State Information | OFDM | Orthogonal Frequency Division Multiplexing |
| D | Dispersion Coefficient | OOK | On Off Keying |
| DD | Direct Detection | OSA | Optical Spectrum Analyzer |
| DECT | Digital Enhanced Cordless Telecommunications | OSNR | Optical Signal to Noise Ratio |
| DPO | Digital Phosphorous Oscilloscope | PAPR | Peak to Average Power Ratio |
| DPM | Digital Phase Modulation | PD | Photo Diode |
| DSP | Digital Signal Processing | P-i-N | P-doped Intrinsic N-doped Photodiode |
| EDFA | Eridium Doped Fiber Amplifier | PON | Passive Optical Network |
| FBMC | Filter Bank Multi-Carrier | PRS | Partial Response Signalling |
| FDM | Frequency Division Multiplex | QMDD | Quadrature Modulation Direct Dectection |
| FDMA | Frequency Division Multiple Access | RF | Radio Frequency |
| FEA | Finite Element Analysis | RIN | Relative Intensity Noise |
| FEC | Forward Error Correction | SCPI | Standard Commands for Programmable Instruments |
| FFT | Fast Fourier Transform | SISO | Single Input Single Output |
| FIR | Finite Impulse Response | SMF | Single Mode Fiber |
| FRS | Full Response Signalling | SNR | Signal to Noise Ratio |
| FTTx | Fiber To The x | SOA | Semiconductor Optical Amplifier |
| GASK | Gaussian Amplitude Shift Keying | SPM | Self Phase Modulation |
| GFDM | Generalised Frequency Division Multiplexing | SS | Spread Spectrum |
| GIPO | General Purpose Input/Output | SSFM | Split-Step Fourier Method |
| GLPF | Gaussian Low-Pass Filter | SSSFM | Symmetricised Split Step Fourier Method |
| GMSK | Gaussian Minimum Shift Keying | TCM | Trellis Coded Modulation |
| GSM | Global System for Mobile Communications | TDM | Time Division Multiplex |
| GVD | Group Velocity Dispersion | TDMA | Time Division Multiple Access |
| IFFT | Inverse Fast Fourier Transform | TFM | Tamed Frequency Modulation |
| IIR | Infinite Impulse Response | TIA | TransImpedance Amplifier |
| IMDD | Intensity Modulation Direct Detection | TDD | Test Driven Development |
| ISI | InterSymbol Interference | UFMC | Universal Filtered Multiple Carrier |
| IVI | Interchangeable Virtual Intruments | USB | Universal Serial Bus |
| LAN | Local Area Network | VISA | Virtual Instrument Software Architecture |
| LD | Dispersion Length | WDM | Wave Division Multiplex |
| LD | Laser Diode | | |
| LUT | Look-Up Table | | |

Abstract

In this project we aim to transmit data in a forest across a wireless channel,we will look at reference to learn about the technology and why it is used ,We consider using the Pi to transmit sensor data by using a serial hat,this report focusing on the process of development in aiming to achieve the nature of a mesh network

Chapter 1

Introduction

1.1 Introduction

This project discusses the process of developing a mesh network from the sensor to serial level Communication, Communication is a key infrastructure in electronics and technology, as this can depict data between different devices, The nature of this is not a simple one due to several factors such as when an object obstructs the path of the signal, other signals interfering with the information that is being sent across the communication channel, There are multiple ways of avoiding this interaction one way on a high level is to use a mesh network like Lora, a mesh network is a type of network where no node in the network acts as a master meaning no device controls the each other. Imagine a group of friends playing a game of telephone. In a regular game, the message goes from one person to the next in a line. But what if, instead of a line, the friends formed a circle, and everyone could whisper the message to the person next to them? That's kind of how a mesh network works. this is possible thanks to the routing of each device. these will select what device talks to each other this project will look at the nature of this interaction

1.2 Motivation

The motives for looking at this topic are the following:

1. To familiarize with Linux os environment
2. To familiarize with bash scripting
3. To Showcase knowledge in the programming language python
4. To Showcase knowledge of embedded systems like the Raspberry Pi and Arduino Uno
5. To Showcase the process of selecting a sensor in the purposed area
6. To Familiarize with communication standards in the purposed area

Chapter 2

Literature Review

2.1 Introduction

The following literature review explores mesh networks in a wooded area, When communicating from two devices across a network there are many of issues associated with this communication such as signal loss due to:

- Environmental conditions such as rain .lighting etc
- Whether the device's antenna are in line of sight with each other
- If the devices are in the line of sight with each other. We can still reflections from a multi-path environment
- Possibility of falling trees obstructing the path of the signal causing more attenuation in the signal strength

In this project aims explore mesh networks and transmit data across them, a mesh network is a type of network where no node in the network acts as a master. A node is a device which has a transceiver. As we look at the environment in which this project will be carried out, we can expect different phenomena to occur such as Attenuation According to ITU ? "Attenuation due to vegetation varies widely due to the irregular Nature of the medium and the wide range of species, densities and water content obtained in practice" Transmitting any radio wave takes energy ,Another factor to consider is whether wind will cause a delay in the signal. This report aims to show my findings and try to account for environmental conditions

2.1.1 Overview

The following section provides a brief overview of this project on mesh networks in a forest the following question is:

1. What frequencies can transmit in a forest
 - What are the disadvantages of transmitting at this range
 - What are the effects of the multi-path environment when there is a line of sight
 - What happens to bon-line of sight
2. What sensors /senor modules should be used
 - What sensors will give a good range in an Irish forest
 - What are the limitations on the board used
 - Is there any need for any additional hardware to accommodate a specific board

3. What microprocessor/hardware should be used?

- The advantages/disadvantages of Arduino vs Raspberry Pi
- What is the major factor in the choice
- How are the sensors wired to the processor
- How to read the data
- What is the effective resolution needed for each application

2.1.2 Mesh network

A mesh network is a type of network that uses multiple devices to relay data between each other, making a decentralized network. The mesh to be used is a wireless mesh network which is created through the connection of wireless access point(WAP) nodes. Wireless mesh networks work through mesh nodes, mesh clients and gateways:

1. Mesh node
nodes act as mesh routers and endpoints
2. Mesh clients
these are end devices
3. Gateways
Data passes through the gateway as it enters or exits a network

The following is a block diagram of a mesh network:

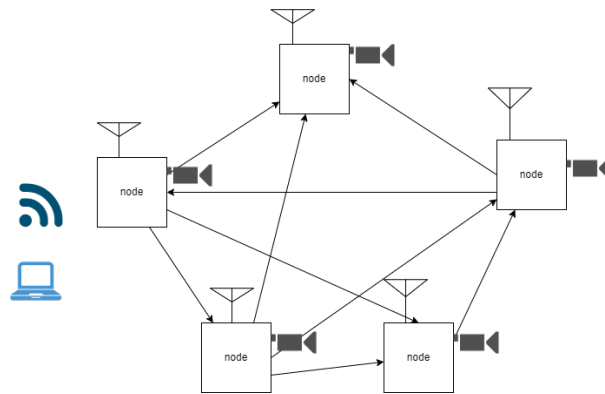


Figure 2.1: Basic block diagram of a mesh network

Each node will be attached to a tree, each having a transceiver

2.2 Hardware Consideration

In this project there needs to be data to transmit, The network needs to be able to:

1. Transmit data for example temperature, humidity, light and camera images.
2. Read data every hour and store it as a CSV file, The image file will depend on the module chosen
3. Detect any animal that passes the node with a motion sensor.

The following is a rough circuit diagram for the project:

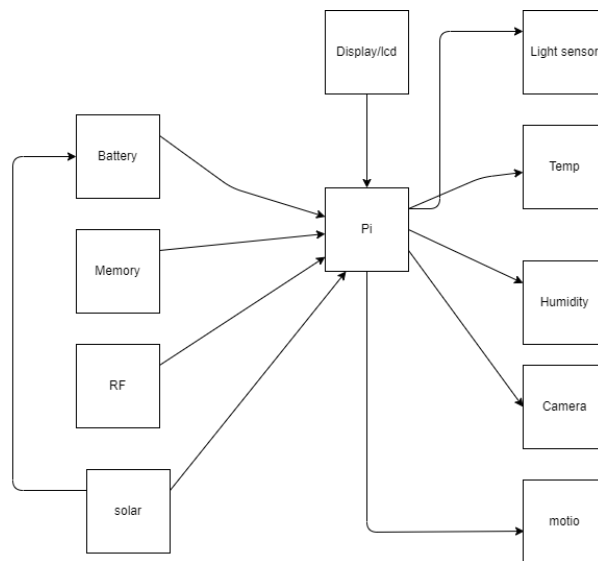


Figure 2.2: Rough circuit diagram for project

1. A PCB cannot be used due to the ordering process taking too long for the time frame of this project
2. Any type of board like wire wrap would take too long and would be outside of the goals of this project
3. A choice of either the Arduino or Raspberry Pi is left

This section discusses the following:

1. The sensors to be used in the project
2. The ADC we will have to will have to consider
3. The camera chosen will have to be considered for this project
4. The memory module considerations
5. The battery chosen
6. A choice made between Arduino or Raspberry PI

2.2.1 Sensor considerations

In this section, the process of considering each component of the sensors will be discussed. These components are:

1. Temperature
2. Humidity
3. Light
4. Motion

Temperature & Humidity sensor

The sensor needs to be able to work in the following conditions:

1. The mesh node will be outside
2. The device is in Ireland
3. The device is in a forest

Taking these requirements into consideration, the temperature in Ireland was researched.

The following table was obtained from Met Eireann ?. which shows the highest air temperature in a shaded area

| Highest Shaded Air (°C) | Station | Date |
|-------------------------|----------------------------------|-----------|
| 18.5°C | Dublin (Glasnevin) | 10th 1998 |
| 18.1°C | Dublin (Phoenix Park) | 23rd 1891 |
| 23.6°C | Dublin (Trinity College) | 28th 1965 |
| 25.8°C | Donegal (Glenties) | 26th 1984 |
| 28.4°C | Kerry (Ardfert Liscahane) | 31st 1997 |
| 33.3°C | Kilkenny (Kilkenny Castle) | 26th 1887 |
| 33.0°C | Dublin (Phoenix Park) | 18th 2022 |
| 31.7°C | Carlow (Oak Park) | 12th 2022 |
| 29.1°C | Kildare (Clongowes Wood College) | 1st 1906 |
| 25.2°C | Kildare (Clongowes Wood College) | 3rd 1908 |
| 20.1°C | Kerry (Dooks) | 1st 2015 |
| 18.1°C | Dublin (Peamount) | 2nd 1948 |

Table 2.1: Highest shader air Met Eireann(13th June 2023)

According to the table, the highest temperature is 33.3. The other extreme of the Lowest temperature was then considered:

| Lowest Shaded Air (°C) | Station | Date |
|------------------------|---------------------|-----------|
| -19.1°C | Sligo (Markree) | 16th 1881 |
| -17.8°C | Longford (Mostrim) | 7th 1895 |
| -17.2°C | Sligo (Markree) | 3rd 1947 |
| -7.7°C | Sligo (Markree) | 15th 1892 |
| -5.6°C | Donegal (Glenties) | 4th 1979 |
| -3.3°C | Offaly (Clonsast) | 1st 1962 |
| -0.3°C | Longford (Mostrim) | 8th 1889 |
| -2.7°C | Wicklow (Rathdrum) | 30th 1964 |
| -3.5°C | Offaly (Clonsast) | 8th 1972 |
| -8.3°C | Sligo (Markree) | 31st 1926 |
| -11.5°C | Wexford (Clonroche) | 29th 2010 |
| -17.5°C | Mayo (Straide) | 25th 2010 |

Table 2.2: Lowest shader air Met Eireann(13th June 2023)

According to the table above the lowest temp is -19.1 In consideration of where the project the condition is a range of -19.1°C to 33.3°C.

Humidity was also researched, Humidity refers to the amount of water vapor in the air. The following table was obtained from Met Eireann ?:

| | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec | Year |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Mean at 0900UTC | 87.0 | 86.4 | 84.0 | 79.5 | 76.9 | 76.7 | 78.5 | 81.0 | 83.4 | 85.5 | 88.5 | 88.0 | 83.0 |
| Mean at 1500UTC | 80.6 | 75.7 | 71.0 | 68.3 | 68.0 | 68.3 | 69.0 | 69.3 | 71.5 | 75.1 | 80.3 | 83.1 | 73.3 |

Table 2.3: Realtive Humidity(%) according to met eirrean

The ranges are from 68.3% to 88 % Taking these considerations into account. Here are the different components:

| Components | Voltage Range | temperature range | Accuracy | Analogue /Digital outputs | Current in | additional information |
|------------|------------------|-------------------|-----------------------------------|---------------------------|------------|-------------------------|
| DHT22 | 3-6 volts | -40 to 80 ^o C | +/-0.5°C | Digital | 1.5mA | sample period 2 seconds |
| LM35D2 | 4 -30 Volts | -55 to 150 | +/-0.5°C (at 25°C) | Analogue | 10mA | None |
| TMP36 | 2.7 to 5.5 Volts | -40 to 125 | +/-1°C (at 25°C) | Analogue | 250 µA | NONE |
| LM75 | 3.0 to 5.5V | -55 to 125°C | +/-2.0°C (at -55 to 125°C range)) | Analogue | 100 µA | NONE |
| DHT111 | 3-5.5V | 0-50 °C | ±2°C | Digital | 1mA | sample period 1 second |

Table 2.4: Comparing of temperature sensors

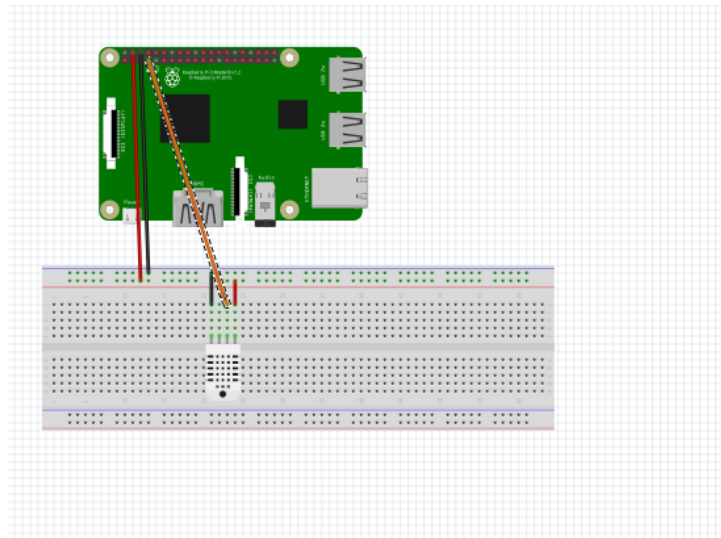
After this, the choice between the two sensors was narrowed down to DHT22 and DHT11. The following are the advantages and disadvantages of the DHT22 and DHT11:

| Device | Advantages | Disadvantages |
|--------|--|---|
| DHT22 | good accuracy has temp and humidity, falls in our temp range | sample period 2 seconds |
| DHT11 | OK voltage,better sample period | draws a lot of current , and our of range |

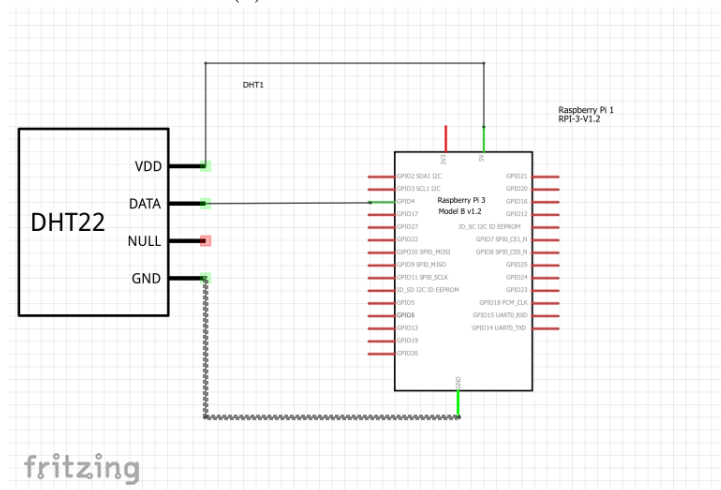
Table 2.5: Comparing DHT22 and DHT11

DHT22 was chosen. Which has a Digital output. See a wiring diagram next page:

This will have an interface of the following:



(a) Interface for DHT22



(b) Schematic for DHT22

From above the schematic is seen. DHT22 connections are the following:

- VDD is connected to 5v of the Pi
- the Data pin is connected to GPIO 3
- Gnd pin of the Pi is connected to the ground of DHT22

? The following is the link to the datasheet of this module when reading from this component. There is a delay of 2 seconds due to the sampling period.

Light sensor

In this section, the following will be considered:

1. What region the project is in
2. What light levels are expected in this country
3. What sensor will accommodate this range

For this sensor, the outside aspect of the project must also be considered. The following table was found on ? : This table is the associated lux level indicating when the values are. From

| Imminence | Example |
|---------------------------|---|
| **0.002 lux** | Moonless clear night sky |
| **0.2 lux** | Design minimum for emergency lighting (AS2293). |
| **0.27 & 1 lux** | Full moon on a clear night |
| **3.4 lux** | Dark limit of civil twilight under a clear sky |
| **50 lux** | Family living room |
| **80 lux** | Hallway/toilet |
| **100 lux** | Very dark overcast day |
| **300 to 500 lux** | Sunrise or Sunset on a clear day. Well-lit office area. |
| **1,000 lux** | Overcast day; typical TV studio lighting |
| **10,000 to 25,000 lux** | Full daylight (not direct sun) |
| **32,000 to 130,000 lux** | Direct sunlight |

Table 2.6: Illuminates values

above the sensor should ideally be 0.002 to 25000 lux, Bearing this in mind these components were researched:

| Modules | Voltage Range | Analogue /Digital Outputs | illumination range | Current rating |
|-----------------------|---------------|---------------------------|--------------------|----------------|
| LM393 with GL5528 | 3.3v to 5v | Analogue | 0 lux to 100lux | 250nA |
| DFR0026 | 3.3v to 5v | Analogue | 1 Lux to 6000 Lux | 120uA |
| LM393 with n5ac501085 | max 150V | Analogue | 10 lux to 100lux | 1mW |
| LM393 with NSL-06S53 | max 100v | analogue | 1 to 100 | 50mw |

Table 2.7: table of light sensors

After research DFR0026 ? is the option proposed for use as it is the best for this application, which will have an analog output to see the interface (see below):

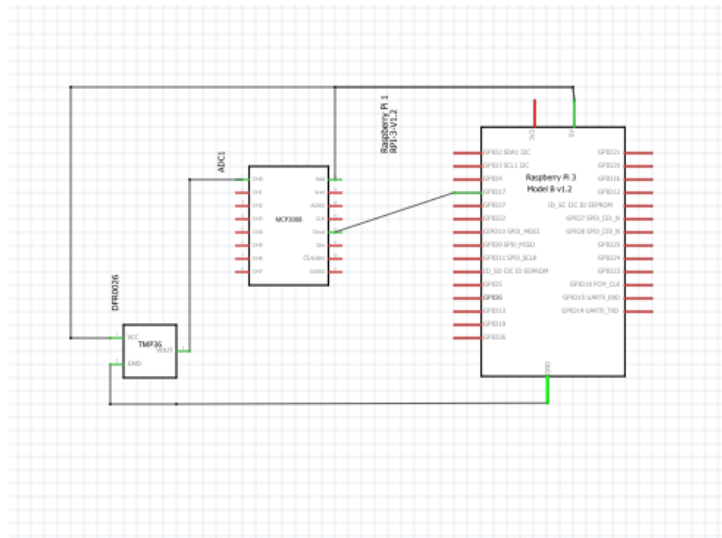


Figure 2.4: Interface for DFR0026

The following are the connections:

1. VCC pin is connected to 5v
2. Gnd of the sensor is connected to Gnd of the Pi
3. The output is connected to ch 0
4. the output ranges from 0 to 5 v

The component relies on the ADC which is on page13

Motion sensor

For this section, the following must be considered:

1. The range of the sensor
2. The degree of the sensor
3. How long of a delay is the sensor

The following are the components considered:

| Modules | Voltage Range | Distance | Max angle | Analogue /Digital Outputs | Power |
|----------|---------------|----------|-----------|---------------------------|-------|
| HC-SR501 | 5-20V | 3 to 7m | 110 | Digital | 50uA |
| AM312 | 4.5-20v | 3m | 130 | Digital | 60uA |
| AS312 | -0.3 - 3.6V | 12m | 130 | Digital | 100mA |

Table 2.8: Motion sensor components

The sensor chosen is AS312?(which has a delay time of 2 seconds) which is a digital interface to see the wiring, See below:

The following is the interface for the device:

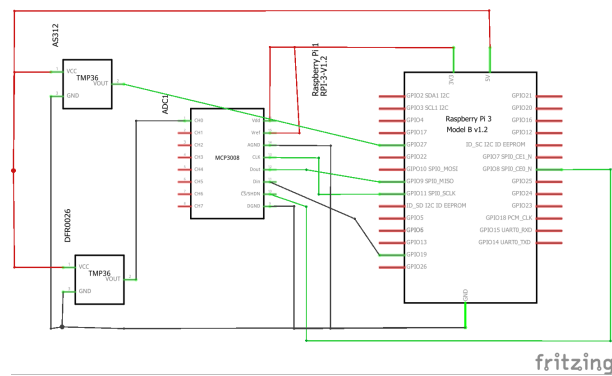
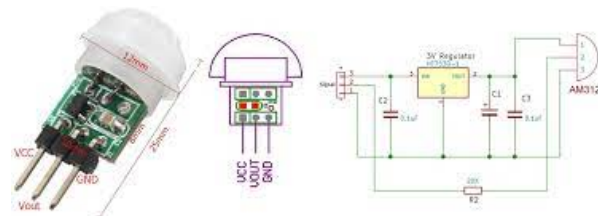


Figure 2.5: Interface for AS312

The connections are the following:



Pinout of AS312

1. VCC is connected to 5v pin of the Pi
2. GND is connected to the GND of the Pi
3. Vout is connected to GPIO 17

This component has the following:

1. Range of 12 meters
2. An angle of 65° degree
3. A Delay of 15μ Seconds

2.2.2 Radio Module

For this section, there are the following considerations:

- The devices are in a forest
- Meaning Gigahertz isn't a desirable frequency
- A module that has low-power
- A model that will have a high throughput

Through research, I found the following table:

| Modules | Tx/Rx Voltage | Frequency | Range | Tx/Rx power | Through put | Error detection | Rx sensitivity | Hopping channel |
|---------------------------------------|----------------|----------------|---------------|--------------------|-----------------|-------------------------------|---|-------------------------------------|
| Gravity 315MHz RF Receiver Module | 3.3v/5v | 315Mhz | 50 m | -10dbm -95dbm | 9.8kbps (max) | none | -108 dbi | no |
| MM2 Series 900 MHz OEM Radio | 3.5 5.0 | 902-928 Mhz | 32km | 1175 ma tx rx 15ma | 80 - 115.2 kbps | 32-bit CRC | -108dbm @ 115.2kbps for BER 10 ⁻⁴ -109 dbm @153.6 for BER 10 ⁻⁴ | 50 to 112, user-selectable |
| RF 433MHz Transmitter/Receiver Module | 5V 3-12v | 433.92 Mhz | 20-300 meters | 10 mW | 2kbps | ASK modulation no error check | -105 db | no |
| Digi XBee-PRO 900HP RF Module | 2.1 to 3.6v dc | 902 to 928 Mhz | 14km - 6.5km | 24dbm | 10kbps -200kbps | NONE | -107dbm | FHSS (Software Selectable Channels) |

Table 2.9: Radio modules found in research

Out of these, the MM2 Series 900 MHz? was chosen. Note that the seller of this radio module has limited the documentation of this module. This makes it hard to draw an interface for this module.

2.2.3 ADC Considerations

For the ADC there are the following considerations:

1. Low power
2. High-bit resolution
3. Low number of channels
4. High sample rate

The two things needed for this is a high bit Resolution and a high sample rate

| Device | Resolution | Sample rate | Input range | Power consumption |
|-------------|------------|-------------|-------------|-------------------|
| ADC pi Zero | 17 bits | 100KHz | 0-5.06v | 10mA |
| MCP3008 | 10 bits | 200 ksps | 2.7v- 5.5v | 500uA |
| DFR0553 | 16 bits | 1.7Mhz | 0 5.0V | 10mA |

Above are the components to choose from for this project, MCP3008 was chosen due to its resolution and sample rate

The following is the schematic for the MCP3008?

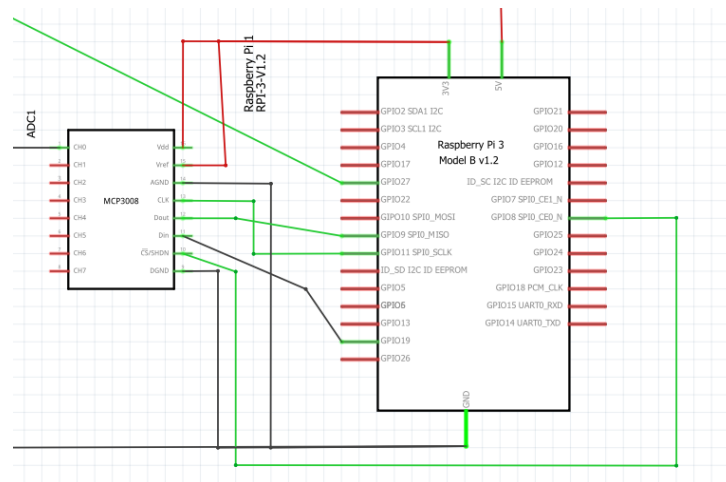


Figure 2.6: Schematic for MCP3008

The following are connections:

1. VDD is connected to 3v3 pin of the Pi
2. VRef is also connected to the 3v3 pin of the Pi
3. AGND is connected to the GND pin
4. CLK pin is connected to GPIO port 11
5. Dout pin is connected to GPIO port 9
6. Din pin is connected to GPIO port 19
7. CS pin is connected to GPIO port 8
8. DGND ping is connected to the GND pin

This component has the following:

1. A 10-bit resolution
2. Seen as the reference will be 3.3v
3. 200ksps, meaning the delay to read is 5μ seconds

2.2.4 Camera

For the camera, the following has to be considered:

1. Focal length
2. Resolution
3. Power
4. The lux values it operates at

The camera chosen is a Raspberry Pi VR 220? Camera To see how to connect look at the following link.

| Modules | Voltage range | lens size | Image Resolution | Video Resolution | Frame Rate | Type of Output | Preferred condition | Power |
|----------------------------|-----------------|-------------------------|------------------|------------------|------------|------------------|---------------------|-------|
| Raspberry Pi VR 220 Camera | 3.3V ac | can change with lens | 3280 X 2462 | 1920 x 1080 | 30 FPS | Need to research | Daytime | 38mA |
| DIGILENT 410-358 | 3.6v | optical size 1/4 inches | 2592 x 1944 | ? | ? | Digital | ? | 200mA |
| The Raspberry Pi NoIR | 3.3v | 1/4 inches | 3280 x 2464 | 1080 or 720 | 30 60 fps | need to research | house | 38mA |
| OV7670 VGA | 2.45 to 3.0v ac | 1/6 inches | 2.36mm x 3.6um | ? | 30 fps | analogue | need to research | 60mW |

Table 2.10: Camera module

2.2.5 Memory module

For this section, we consider the following:

1. The file formatting of the sensor data
2. The file formatting of the camera data
3. What are the possible sizes of data
4. What is the memory size of the Raspberry Pi OS

In my project, the following is used:

1. The sensor data is stored in a CSV file with the following heading: (timestamp, heat, humidity, light level, anything detected), Which can be around 25KB
2. For the camera using 10 MB is the largest file size
3. For the Raspberry Pi download the Raspberry Imager, This has lots of options such as the following on page 19

After has been we must consider a mircoSD, The following are considerations:

| Product Name | Capacity | Speed class | Read speed | V |
|--|----------|-------------------|---------------|----|
| SAMSUNG EVO Plus Class 10 microSDXC | 256 GB | U3 | up to 130MB/s | up |
| SANDISK Ultra Performance Class 10 microSDXC | 128 GB | class 10 u1 | up to 80 MB/s | up |
| Silicon Power 32GB 3D | 32GB | class 10 | Up to 100MB/s | Up |
| SanDisk 64GB Extreme PRO | 64GB | UHS Speed Class 3 | Up to 100MB/s | up |

Table 2.11: Mirco SDs in consideration

The best here is the silicon power 32GB due to its temperature range and read and write times. we can consider extra memory:

1. DHTT22 has a sample period of 2 seconds
2. AS312 which has a delay time of 15 μ seconds
3. MCP3008 5 μ seconds

Through research, the following was discovered :

| Brand | Product Name | Storage Capacity | USB Version | Data Transfer Sp | Read/Write Spec | Durability | Encryption | Form Factor | Compatibility | Price |
|----------|--------------------------------------|------------------|-------------|------------------|-----------------|---------------------------|------------|-------------|---------------------|--------|
| SanDisk | Cruzer Glide 1GB USB 3.0 Flash Drive | 1 GB | USB 3.0 | 100 MB/s | 80 MB/s | Water and shock resistant | No | Standard | Windows, Mac, Linux | \$5.99 |
| PNY | Turbo 1GB USB 2.0 Flash Drive | 1 GB | USB 2.0 | 480 Mbps | 300 Mbps | Water and shock resistant | No | Standard | Windows, Mac, Linux | \$3.99 |
| Verbatim | Store 'n' Go 1GB USB 2.0 Flash Drive | 1 GB | USB 2.0 | 480 Mbps | 300 Mbps | Not specified | No | Standard | Windows, Mac, Linux | \$2.99 |

Table 2.12: Memory usb to consider

The Raspberry Pi 4 supports USB 2 and USB 3. For this, the Turbo 1GB USB 2 Flash Drive will be chosen.

2.2.6 Battery

In this section, the following must be considered:

1. Enough power for all sensors and radio module
2. Storage of the battery
3. Discharge rate of the battery (how many operating hours can be got out of the battery)

Here are the following Devices I found :

| Modules | Voltage | Interface | Power | Chemistry | Supply time |
|------------------------|---------|-----------|-------|-----------------|-------------|
| Li-polymer Battery HAT | 5v | Micro USB | 1.8A | lithium battery | 5 hours |

Table 2.13: battery considerations

The battery chosen is the li-polymer which has a micro USB How to charge:

- Step 1: Insert the Li-polymer battery into a 2.0mm battery socket
- Step 2: Connect the power adapter to a micro USB or Type-C interface by USB cable.

Aside: this component has the following:

1. A battery that is 3.7v 3000mAh
2. Output voltage of 5 volts
3. an estimated Power supply time of 5 hours

2.2.7 Arduino vs PI Consideration

In this project, we will have to choose between what microprocessor we will use. There are 3 options:

1. PCB (printed circuit board) where the circuit is designed in a program like Fusion 360. The major issue is due to the current state of silicon chips which will slow down the progress of the implementation stage
2. Arduino
3. Raspberry Pi

The advantages and disadvantages of the Arduino and the Raspberry Pi are the following:

| Arduino | Pi |
|---|--|
| Advantages | Advantages |
| 1. Arduino has a 10-bit ADC | 1. Pi can compile Python (easier to write) |
| Disadvantages | Disadvantages |
| 1. Arduino has a supper set of C++ 2. Arduino only has 6 Analogue pins | 1. Pi is a technically a small CPU 2. The Pi needs an ADC circuit to deal with inputs that are analog |

Table 2.14: Advantages /Disadvantages of Arduino vs pi

However, the Arduino would be more efficient than the Raspberry Pi due to Raspberry Pi having an Operating System. The Pi was picked due to familiarity with Python and Linux. Linux can be used to handle the networking side of the project At the cost of some efficiency in power for an easier time writing the code for this project

Picking a Raspberry Pi

Now that a device was chosen to use the following needs to be defined:

1. The amount of GPIO PORTS we need
2. Nature of the output of the sensor
3. Speed of the clock

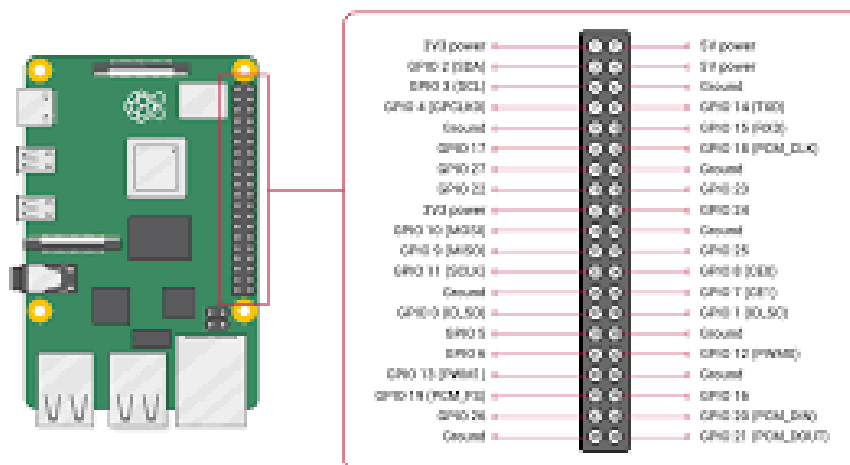
GPIO(General purpose input/output) is used to select the input/output. The Pi can take in digital signals only. Seen as the components are chosen that require A GPIO port (temperature/humidity, on page 8, Light on page 10, motion on page 11)

At least 3 GPIO ports to be available to us as the light sensor and the motion will need an adc as looking through the documentation First, let's look at the different models:

| Raspberry Pi Model | Internal Clock Speed | Power (Watts) | GPIO Features | Type of Connectors | SRAM |
|-------------------------|----------------------|---------------|---------------|--|---------|
| Raspberry Pi 1 Model B+ | 700 MHz | 5.5 | 26 GPIO pins | 1 HDMI, 1 micro USB, 1 USB 2.0, 1 audio jack | 512 MB |
| Raspberry Pi 2 Model B | 900 MHz | 7.5 | 40 GPIO pins | 1 HDMI, 1 micro USB, 4 USB 2.0, 1 audio jack | 1 GB |
| Raspberry Pi 3 Model B+ | 1.4 GHz | 8 | 40 GPIO pins | 1 HDMI, 1 micro USB, 4 USB 2.0, 1 audio jack, 1 Gigabit Ethernet, 1 PoE header | 1 GB |
| Raspberry Pi 3 Model A+ | 1.4 GHz | 5 | 26 GPIO pins | 1 HDMI, 1 micro USB, 2 USB 2.0, 1 audio jack | 512 MB |
| Raspberry Pi Zero | 1 GHz | 1.2 | 40 GPIO pins | 1 mini HDMI, 1 micro USB, 1 micro-USB OTG | 512 MB |
| Raspberry Pi Zero W | 1 GHz | 1.3 | 40 GPIO pins | 1 mini HDMI, 1 micro USB, 1 micro-USB OTG, 1 Wi-Fi/Bluetooth module | 512 MB |
| Raspberry Pi Zero 2 W | 1 GHz | 0.8 | 40 GPIO pins | 1 mini HDMI, 1 micro USB, 1 micro-USB OTG, 1 Wi-Fi/Bluetooth module | 512 MB |
| Raspberry Pi 4 Model B | 1.5 GHz | 7 | 40 GPIO pins | 2 HDMI, 2 USB 3.0, 2 USB 2.0, 1 Gigabit Ethernet, 1 audio jack | 1 GB, 2 |

Table 2.15: Table of Raspberry Pi's

The above table displays the modules, As our radio module is 900Mhz we want 1.5GHZ which is the Raspberry Pi 4. This needs a USB c -charger and an HDMI mini cable. For wiring our Pi here is the GPIO pin layout:



Pinout for the pi

2.2.8 Conclusion

In this project, the hardware needed is the following components:

1. 1 x Raspberry Pi 4 Model B
2. 1 x HDMI cable

3. 1 x USBC cable
4. 1 x USB C charging head
5. 1 x DHT22
6. 1 x DFR0026
7. 1 x AS312
8. 1 x MM2 Series 900 MHz
9. 1 x MCP3008
10. 1 x Raspberry Pi VR 220 Camera
11. 1 x Li-polymer Battery HAT
12. 1 x Turbo 1GB
13. 1 x silicon power 32GB microSD

2.3 Software considerations

Having established the essential hardware needed for this project. Next, consider the following for the software of the project:

1. How to structure code
2. Linux set up of sever and nodes
3. How will data be sent
4. Will this be an OOP or functional approach?
5. How to program each device?

2.3.1 Raspberry Pi OS

In this section, it must be kept in mind that each OS is heavyweight the following needs to be considered:

1. If the SD is formatted the data on the SD is lost. Does it corrupt the card?
2. An os that is low in capacity
3. Is a desktop needed or can we use the terminal?
4. How does the OS respond to USB drives?

According to the ? the imager will erase all the data while installing the os. From research, the suggestion of backing up the data is a good suggestion. for now, the recommended OS is used. and strip down as the project progresses. which will be discussed in the methodology section of this report.

2.3.2 Sensor code

In this section, the following will be discussed :

1. DHT22
2. AS312
3. MCP3008
4. DFR0026
5. Kuman for Raspberry Pi 3B+ TFT LCD Display
6. Raspberry Pi VR 220 Camera

the project code will mainly be object-oriented. so the goal is to first test it with my laptop and Create A bash file full of commands to install the libraries, making the code split up into different parts so that all that is needed is the libraries used and code that won't all have to be compiled in one file.

DHT22

In this section we have to consider the following:

1. The GPIO port as on page 9 This is connected to port 3
2. The type of output is digital so no extra hardware/code is needed

The following is a rough guide on how to read from the DHT22 from the following link. Firstly open the terminal in the Pi and type the following commands:

```
1      git clone https://github.com/adafruit/  
      ↪ Adafruit_Python_DHT.git  
2      cd Adafruit_Python_DHT  
3      sudo apt-get update  
4      sudo apt-get install build-essential python-dev  
5      sudo python setup.py install
```

the code does the following:

1. firstly git clone will clone the repository onto to device
2. Then change dirertorys a
3. update linux
4. install dev kit for python
5. and install the setup

this will then lead to the following code:

Listing 2.1: Example code for DHT2

```
1      #Libraries
2      import Adafruit_DHT as dht
3      from time import sleep
4      def setup_DHT22(Gpioport:int):
5          humidityy,temp=dht.read_retry(dht.DHT22, Gpioport)
6              sleep(5)
7              return humidity, temp
8      h,t=setup_DHT22(3)
9      print('Temp={0:0.1f}*C□□Humidity={1:0.1f}%'.format(t,h))
```

this code will do the following:

1. Import DHT from the adafruit library
2. in the function which takes the gpioport as an integer this will read the data on the pin and print it out

AS312

for this section i followed this link we also want to keep in mind the following:

1. This has a digital interface and is connected to GPIO 27

Here are the rough steps firstly type the following into the terminal

```
sudo apt-get install python-rpi.gpio
```

which will install a gpio python module

Then type this into an IDE of your choosing

Listing 2.2: Example code for AS312

```
1 import RPi.GPIO as GPIO
2 import time
3
4 pir_sensor = 27
5 GPIO.setmode(GPIO.BOARD)
6
7 GPIO.setup(pir_sensor, GPIO.IN)
8 current_state = 0
9
10 time.sleep(0.1)
11 current_state = GPIO.input(pir_sensor)
12 if current_state == 1:
13     print("GPIO pin %s is %s" % (pir_sensor,
14                                   current_state))
15     # trigger camera
16     # must look up this
17 GPIO.cleanup()
```

this code does the following:

1. it will look at the pin for a pulse
2. once it senses a pulse it will trigger the camera

DFR0026

from the last example, nothing has changed from the last component an example code for this can be found on page 22

2.3.3 MCP3008

for this section, we want to consider the following:

1. The MCP3008 data out is GPIO 9

this section follows this link firstly try the following in command in the terminal

```
sudo raspi-config nonint do_spi 0
```

Listing 2.3: ADC code

```
1 from gpiozero import MCP3008
2 from time import sleep
3 DFR0026 = MCP3008(channel=0, device=0, port=9)
4
5 print('raw: {:.5f}'.format(DFR0026.value))
6 sleep(0.1)
```

this code will select a channel and device, port and print the values of the ADC's

2.3.4 Raspberry Pi VR 220 Camera

to get started with this simply look at the following link here is an example of the code of this module :

Listing 2.4: example code for camera

```
1      from picamera import PiCamera
2      from time import sleep
3
4      camera = PiCamera()
5
6      camera.start_preview()
7      sleep(5)
8      camera.stop_preview()
```

this will take a photo of what is in front of the camera

2.3.5 MM2 Series 900 MHz

for this section, the seller of this module has no public documentation so it is hard to come up with an make interface for this section

2.3.6 code structure

The code structure for this will be an object-oriented program all the individual sensors and hardware for the pi will be as displayed above the code in this section will be formatted into objects for example I will have an object called proj_sensor and a method of this would be DHT22 while an attribute of this would be the sample rate the following is a rough breakdown of the structure of the code

- Sensor object
 - Temperature and humidity method
 - light method
 - Motion method which triggers the camera
 - Battery method which is a constructor method
 - Memory method which links with the radio
- radio object which reads from Memory and transmits the data

2.3.7 File structure

For the File structure, we want our sensor data to be stored every hour in a CSV file with the following column headings:

1. timestamp
2. Heat
3. Humidity
4. light level
5. motion detected (True/False)

for the writing to Date, we will use Pandas to write to the CSV file for file sorting, I will use the Python Library glob which I can use to look for files the following is an example of how to make a CSV file: firstly let's make a data frame:

Listing 2.5: sample code for turning sensor data into a data

```

1      import pandas as PD
2      import numpy as np
3      from datetime import datetime
4      cols_name=["Timestamp", "Temperature", "Hummidty", "
               Light_level", "Motion_dected"]
5
6      #assume that being recorded now
7      data=[]
8      timestamp=datetime.now()
9      timestamp=timestamp.strftime("%d/%m/%Y_%H:%M:%S")
10     Current_state=1
11     Heat=0.40
12     Hummidty=1.0
13     Light_level=0.23
14     data=np.array([[timestamp],[Heat],[Hummidty],[Light_level
               ],[Current_state]])
15     data=data.T
16     df= pd.DataFrame(data,columns=cols_name)

```

Next, use the.To_csv method from pandas another Libraries that could be useful is the Tkinter here is a sample of how to store where the file is gonna be:

Listing 2.6: example code for storing directory

```

1      import tkinter as tk
2      from tkinter import filedialog
3      import json
4      import os
5
6      root = tk.Tk()
7      root.withdraw()
8      selected_dir = filedialog.askdirectory()
9
10     if not os. path.exists('selected_dir.json'):
11         # Write the selected directory to a JSON file
12         with open('selected_dir.json', 'w') as f:
13             json.dump(selected_dir, f)
14             print("Successfully saved selected directory to
               JSON file.")
15     else:
16         print("File 'selected_dir.json' already exists. Not
               saving the directory.")
17
18     root.quit()

```

Other useful Libraries allow you to select all .csv, png called glob for our TDD Section, we will have use the following command:

```
# !/bin/bash
```

```

dir_name=$1

size=$(du -sh "$dir_name" | cut -f1)

echo "Directory size: $size"

```

This is a script that will look at a director this can be a home directory that will call the space 13K the "— cut -f1" will only focus on the size string message and then print out the size. this is just a sample script

2.3.8 Test Driven development

In this project we will be using Test Driven Development (TDD) is a software development approach where tests are written before the actual code the following are the advantages of TDD:

1. Advantages
 - (a) TDD forces you to consider potential failure points and edge cases upfront, leading to earlier detection and resolution of bugs.
 - (b) TDD encourages you to think about the desired behaviour and interfaces of your code
 - (c) TDD provides immediate feedback on whether your code works as intended,

2.4 Attenuation

Attenuation refers to a reduction in the strength of a signal. Attenuation occurs with any signal, whether digital or analogue. Seen the aim of making a network the first step is to look into what frequencies can be transmitted and received.

In the environment in which we want our project to take place, we want the following:

1. An antenna that is high so we can affect the data rate of the signal
2. A frequency range at which Attenuation is not present

Through research, I found the following plots:

1. First Plot The first plot I got for Savage e.t al pg. 7 ?

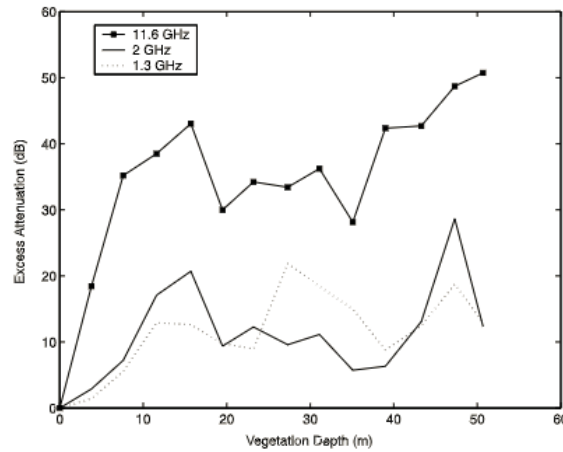


Figure 2.7: Silver Maple in-leaf excess attenuation for the line of trees geometry (receiver antenna height: 3.5 m, SAVAGE ET AL.pg.7

This graph displays as vegetation depth increases Attenuation rises. The problem with this graph is that it doesn't give an in-depth view of which attenuation occurs. This then led me to look up the International Telecommunication Union ? recommendations for Attenuation in wooded areas

2. Second Plot V is the vertical polarization H is the horizontal polarization

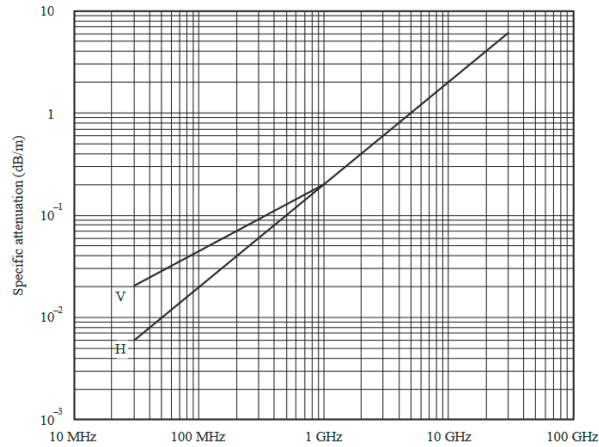


Figure 2.8: Specific attenuation due to woodland (Recommendation ITU-R P.833-7 (02/2012) Attenuation in vegetation pg.5

From this graph we can assume the following:

- (a) From a frequency $\geq 15\text{GHz}$ we can assume Attenuation is more components
- (b) Around the 1 GHz range we get low values of Attenuation
- (c) in the MHz range we get the best response

from this, I selected the range which is 10^6hz

so now that we established our range let us consider what happens when it rains?

| Frequency MHz | Attenuation dB/m |
|---------------|------------------|
| 106 | 0.04 |
| 466 | 0.12 |
| 949 | 0.17 |
| 1852 | 0.3 |
| 2118 | 0.34 |

Figure 2.9: Predicted attenuation due to rain for the region, which is measured by using the ITU standards,(Source: Hindawi(2014))

Ideally, we want a low MHz but we want speed and this is dictated by what we choose let's further see how radio waves are affected by water/rain

2.4.1 Absorption of water

for this, I found this graph from Lunken Heimer ?

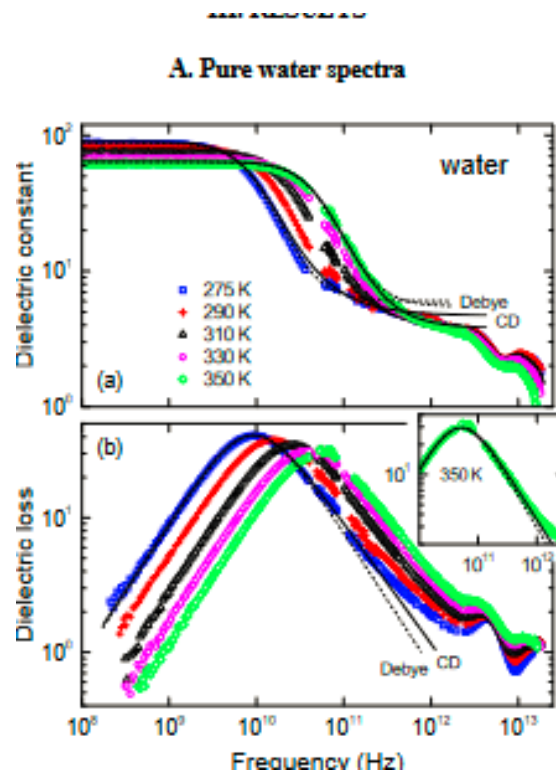


Figure 2.10: absorption of water

According to the graph, Water absorbs MHz frequencies which will affect the transmission in the transmission and in some cases, we might have to consider non-line-of-sight communication when it rains or we might also consider another node to route to receive the node.

2.5 mesh network considerations

For this section, we have to consider the following:

1. How are we setting up the network
2. What framework are we using to set this Up
3. What are the advantages/disadvantages

In my research I found two main frameworks that this project could use to achieve the mesh network these are the following:

1. LORA
2. Zigbee

According to Chen (2023)? "LoRa, as one of Low Power Wide Area Networks (LP-WANs) technologies, aims to enable IoT devices to perform long-range communications with lower power consumption [18]. LoRa makes use of the chirp spread spectrum (CSS) modulation to improve the transmission distance up to kilometres and also be resistant to multi-path effects."

According to Vlad?, "ZigBee is an LP-WPAN (Low-Power-Wireless Personal Area Network) with short range and low power consumption, as mentioned before. The range for ZigBee devices is up to fifty meters and it is characterized by a low data rate, having a maximum value of 250 kbps. The protocol is suitable for sensors and IoT applications because of the low data rate and low power consumption"

the following are the differences between the two: from research, these are very similar but

| LoRa | | ZigBee | |
|--------------------------------------|-------------------------|-----------------------|---------------------|
| Advantages | Disadvantages | Advantages | Disadvantages |
| Long transmission distance | Low transmission rate | Low power consumption | Low data rate |
| Low power consumption | Slow data transfer rate | Long range | Limited range |
| Multi-channel information procession | Small payload | Scalability | Signal interference |
| Strong anti-interference ability | Low bandwidth | — | High-sensitivity |
| High-sensitivity levels | Spectrum interference | | |

Table 2.16: Advantages and Disadvantages of LoRa and ZigBee

it seems if I plan on adding lots of Zigbee is the best for this challenge

2.6 Review key of research Papers

The following are the research papers I used

1. zhao

In my research, I found multiple projects that are similar to mine In Zhao(2023)(?, zhao) used LORA to track light sensitivity, air pressure one of the challenges Zhao came across was Attenuation as stated above and also the author came across the problem of not having sufficient solar panels

2. Daniel

Another paper I found in my research is by Daniel ? In this, Daniel discusses modeling radio wave propagation in a forest environment which isn't in the scope of the project Daniel's work shows that a better approximation for transmission loss was a key read to under what happens on a more in-depth scale in my project

3. Anna

? in Anna's paper she mainly used LORA where she compared line of sight and the non-line line of sight environments in urban and forested areas this paper aims to study the effects of signal propagation in different environments.

4. ITU

? in ITU in most research papers I found it referred back to this document this document was very helpful in terms of understanding Attenuation and challenges that face

2.7 Summary

This report highlights the challenges at come from transmitting data in a wooded area these challenges are the following:

1. Attenuation
2. Absorption

In a wooded area, we established that Attenuation occurs due to the reflection, and penetration of radio through any type of medium. We established that our antenna will have to be in the Mhz range but will still have signal loss /errors due to Absorption of the signal received due to rain or water being in the signal path we have yet to consider the non-line of sight environment but this is to be discussed when prototyping, this report mainly focuses on the hardware where the focus is on sensors such as:

- Temperature
- Light
- Motion
- Humidity

The report focuses on how to read this data from a Software perspective the code will be an object-oriented program where the code will be separated into different blocks of code so the file size is minimized and leads to a faster compile time.

Chapter 3

Methodology

3.1 Introduction

In this Section the proposed methodology of this project will be discussed this will cover the following:

1. The Procedure of the project
2. The Additional research
3. The setup of the Raspberry Pi
4. The Software Model Development
5. The Data Analysis Methods
6. The validity and reliability
7. The Limitations and Delimitation
8. The timeline

3.2 Procedure

The following are the steps of this project:

1. Consider the environment in which we commute across
2. Determine the desired range for the sensor to operate in.
3. Find the wide range of components available
4. Limit the base hardware based on the constraints of the project
5. Select hardware based on these constraints
6. State the software needed for the project
7. State how to setup the software
8. State the software needed to drive this sensor
9. Write unit test to develop the software
10. Discuss how to track the sensor data
11. Discuss the timeline of the project
12. Discuss the limitations

3.3 Setup of raspberry pi

Firstly once you have your pi heres a quick guide to setup the pi are the following:

1. Unpack the pi be sure to connect keyboard mouse and hdmi cable
2. Download the raspberry pi imager and select the 32 bit recommend os
3. Put the micro_SD card into the pi once the pi is setup you can make sub directories for this project type the following:

```
git clone https://github.com/mistaherd/meshnetwork_in_forest.git
```

This will download the essentially environment for setting up the Pi initial this will have to built out through the process of the project look at the timeline Section

4. Next configure the legacy camera options

```
sudo raspi-config
```

5. Download the following tools from linux

```
sudo apt update
sudo apt install raspistill
sudo apt install tmux
```

6. set working directory and activate virtual environment

```
cd <path/to/of/your/own>/meshnetwork_in_forest
source env/bin/activate
```

3.4 Additional Research

In this section will discuss any extra research done on the project. in this section we will discuss the following:

1. ADC
2. Radio module

3.4.1 ADC

The MCP3008 was not available when ordering parts, Another part for this was choosen which is the DFR0553 which has the following:

1. a supply voltages(VCC) of 3.3 to 5 v
2. Analog signal detection 0 to 5v
3. 4 analog chanel's
4. resolution of 16 bits
5. Operating current of 3mA

3.4.2 Radio module

for this section we want to keep the following in mind :

1. We want a module that will send and received data
2. we don't want an expensive solution due to wanting to have multiple nodes
3. must we pick a standard?
4. what module has an open source project on it
5. how do we set up a mesh network with this

Do we need a radio standard?

Lets assume we communicate with two pi via wires we know that an interference will occur when we commutation that is wireless we can have multiple cases where interference can occur these are the following:

1. the signal being reflected of objects such as trees
2. the signal can reach the receiver due to an object blocking the antenna
3. the signal isn't power to be picked up by the receiver

one essential part of this project is the ability to have our nodes have an address to set this up from a communication preceptive we could develop this when there is open source project that has sorted out the routeing for you. only issue with this approach is if there is any issues that come from the open source project we will inherit the bugs with this in mind the following standards were found

1. LoRa

LoRa

In ? lora is used that will organize sensor data from all nodes in the spanning tree toward the root(laptop /PC) this can be show by the following: this proves it possible to make a mesh

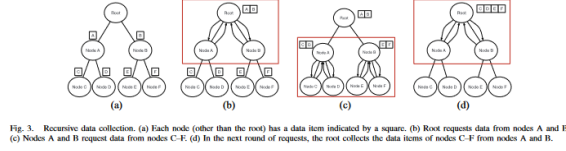


Figure 3.1: protocol Wu used(wu.lie et.al,2023:16705)

network using Lora.

from looking online Lora has more projects that are open source meaning we can use it.freely for example

Lora is uses spread spectrum modulation, In ? spread spectrum is apparent in Shannon's theorem which states the channel capacity C the upper limit on the information rate of data that can be communciated at a lower error rate through the received signal power S :

$$C = B \log_2(1 + \frac{S}{N})$$

Where B is the is the bandwidth of the channel in hertz.Where the bandwidth is:

$$B = F_{max} - F_{min}$$

spread spectrum creates a pseudo-random code sequence that modulates the data signal which will determine the how the signal is spread out.

To simulate the system we can use the following FIR response as an example in a given

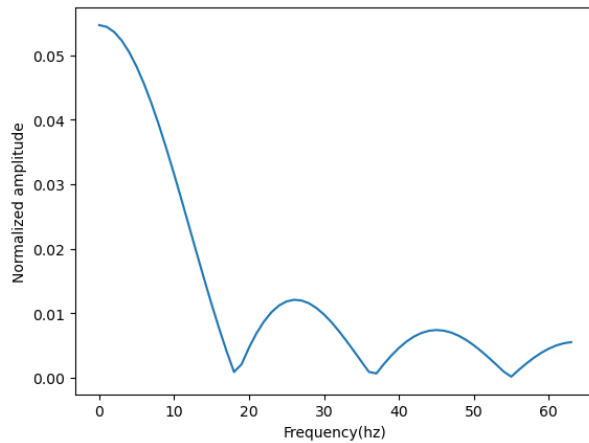


Figure 3.2: sample graph of a FIR response

medium of transmit each bandwidth is the length the of the sinc-roll-off which degrade depends on the impulse response in this given bandwidth channels are separated in the same fashion .

3.4.3 What is the difference between a port and a channel

In ? "A port is a virtual point where network connections start and end. Ports are software-based and managed by a computer's operating system. Each port is associated with a specific

process or service. Ports allow computers to easily differentiate between different kinds of traffic: emails go to a different port than webpages, for instance, even though both reach a computer over the same Internet connection.”

3.4.4 Why the MM2 Series 900 MHz wasn't picked

When ordering the parts for this module issues where due to company not selling the product to enterprise-level businesses so then two alternative radio modules were found:

1. SB Components LoRa HAT for Raspberry Pi
2. RPIZ SHD LORA433 Raspberry Pi Shield - LoRa, 433 MHz, SX1268

when we compare these we get the following table:

| Modules | Tx/RX Voltage | Frequency | Range | TX/RX power | Through put | Error detection | Rx sensitivity | Hopping channel |
|---|---------------|--|--|-------------|-------------|-----------------|--------------------------|-----------------|
| SX1268 433M LoRa HAT | 5v | 410.125~493.125MHz or 850.125~930.125MHz | 5KM(Sunny day; open area; Antenna: AUX 5dBi, Height 2.5m; Air Speed: 2.4kbps) | 11ma /100ma | 0.3Kbps | None | -147dBm@0.3Kbps (On air) | None |
| SB Components LoRa HAT for Raspberry Pi | 5v | 915/868/433 MHz | 5km | 22dBm | 0.3Kbps | None | N/A | None |

Table 3.1: Comparing New Radio modules

3.4.5 SB Components LoRa HAT

Which has a E22-900T22S on the board which has a throughput rate of 0.3kbps-62.5kbps so the maximum time it will take to get to a node will be around 16 seconds depending on distance ,the module has two was of interacting with the board:

1. Pi
2. USB to windows desktop in this configuration we can test single node from our laptop this is just to test sending message across the serial

This hat supports three frequencies:

1. 868 Mhz
2. 433 Mhz
3. 915 Mhz

E22-900T22S

E22-900T22S is a wireless serial port module (UART) based on SEMTECH's SX1262 RF chip. It has multiple transmission modes, working in the 850.125MHz 930.125MHz, (default 900.125MHz).which has the following functions:

1. LoRa spread spectrum
2. Listen before talk(LBT):
The module will monitor the channel before transmitting data , if the environment exceeds the threshold. it will be delayed .
3. RSSI(Received Signal Strength Indicator):
It's a measurement of how strong a radio signal is when it's received by a device,This is used in tandem with the LBT function
4. Networking function:
The module can implement multi-level repeater networking, as discuss above section when a single is sent over long distance it gets weaker, walls floors and other objects can block or distort this signal , A repeater reive these signal's and simply amplified and retransmits the data,Multi-level have secondary repeaters will boost the signal further in order to avoid signal loss.

5. Ultra-low power consumption:
6. Broadcast monitoring: Set the module address to 0xFFFF, which can monitor the data transmission of the module

3.5 Software Module Development

This section is here to discuss the method we took for developing software for the following:

1. Sensors
2. ADC
3. Camera
4. Radio module
5. Memory management
6. TDD

3.5.1 Sensors

This Section will discuss the following:

1. DHT22
2. AS312
3. DFR0026

To see the light sensor look on page 37

DHT22

For this section we used the following libraries:

```
1 #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/  
  python3.11  
2 import adafruit_dht  
3 import board  
4 import pandas as pd
```

This uses the library from this link

1. we define the our class

```
1 class DHT22:  
2     ##Set DATA pin to pin 4  
3     def __init__(self):  
4         """this will setup the data pin for DHT2"""  
5         # self.dhtDevice =adafruit_dht.DHT22(board.D4)  
6         self.dhtDevice =adafruit_dht.DHT11(board.D4)  
7         self.humidity=self.dhtDevice.humidity  
8         self.temperature=self.dhtDevice.temperature
```

In this class we have define our DhT device as 11 seen as the DHT22 was broken so we set our gpio pin 4 and setup the variables that read the sensor data

2. Next we read the data from the following function.

```
1     def Read_DHT22_data(self)-> tuple[float,float,str]:
2         """This will setup a DHT instance and return the
3           data from the sensor"""
4         try:
5             return self.temperature,self.humidity
6         except RuntimeError as e:
7             print(f"Error reading sensor:{e}")
8             return None, None
```

this will return out the temperature and humidity if the sensor is not connected this will return nothing . next use the following:

```
1     if __name__ == "__main__":
2         DHT22()
```

AS312

1. For this we import the following libraries:

```
1     #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/
2     env/lib/python3.11
3     import RPi.GPIO as GPIO
4     import time
```

2. next we set up our variables in the class

```
1     class AS312:
2     def __init__(self):
3         "connect the AS312 to pin 17"
4         self.pin_number=17
5         self.GPIO=GPIO
6         self.GPIO.setmode(GPIO.BCM)
7         self.GPIO.setup(self.pin_number,GPIO.IN)
8         self.current_state=0
```

This sets current state as 0

3. next we detect movement

```
1     def read_state(self)->bool:
2         time.sleep(0.1)
3         self.current_state =bool(self.GPIO.input(self.
4           pin_number))
5         return self.current_state
```

DFR0026

From the repository DFRobot_ADS1115 the following is considered:

1. Import the libraries:

```

1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/
   lib/python3.11
2  from DFRobot_ADS1115 import ADS1115
3  import time

```

2. Define our variables:

```

1  class DFR0026():
2      def __init__(self):
3          self.ADS1115_REG_CONFIG_PGA_6_144V = 0x00  # 6.144V
4              range = Gain 2/3
5          self.ADS1115_REG_CONFIG_PGA_4_096V = 0x02  # 4.096V
6              range = Gain 1
7          self.ADS1115_REG_CONFIG_PGA_2_048V = 0x04  # 2.048V
8              range = Gain 2 (default)
9          self.ADS1115_REG_CONFIG_PGA_1_024V = 0x06  # 1.024V
10             range = Gain 4
11          self.ADS1115_REG_CONFIG_PGA_0_512V = 0x08  # 0.512V
12             range = Gain 8
13          self.ADS1115_REG_CONFIG_PGA_0_256V = 0x0A  # 0.256V
14             range = Gain 16
15          self.ads1115 = ADS1115()
16          self.ads1115.set_addr_ADS1115(0x48)
17          self.ads1115.set_gain(self.
18              ADS1115_REG_CONFIG_PGA_6_144V)
19          self.adc_channel = 0

```

This configures all the pins and sets the associative gain.

3. Read the analog channel:

```

1      def read_voltage(self):
2          return self.ads1115.read_voltage(self.adc_channel)

```

3.5.2 Camera

Here are the steps for module development of the Camera:

1. install the following libraries:

```
1      #!/home/mistaherd/Documents/Github/  
      meshnetwork_in_forest/env/lib/python3.11  
2      from picamera2 import Picamera2 ,Preview  
3      from time import sleep  
4      from datetime import datetime
```

2. we define our class variables

```
1      class Raspberry_Pi_VR_220:  
2          def __init__(self):  
3              """setup an instance for the camera"""  
4              self.timestamp=datetime.now().strftime("%Y-%m-%d_  
              %H-%M-%S")  
5              self.fname = '/home/mistaherd/Documents/Github/  
              meshnetwork_in_forest/Images_camera/{}.png'.  
              format(self.timestamp)  
6              self.camera=Picamera2()  
7              self.camera_config=self.camera.  
              create_preview_configuration()  
8              self.timeamount=2
```

3. make the function for taking a picture

```
1      def take_pic(self)-> str:  
2          """this will take a picture from camera"""  
3          self.camera.configure(self.camera_config)  
4          self.camera.start_preview(Preview.QTGL)  
5          self.camera.start()  
6          sleep(self.timeamount)  
7          self.camera.capture_file(self.fname)  
8          return self.fname
```


3.5.3 Memory Management

For this we want to read data and append and check it the memory size. Here are the following steps:

1. import the following libraries:

```
1      #!/home/mistaherd/Documents/Github/  
      meshnetwork_in_forest/env/lib/python3.11  
2      import pandas as pd  
3      from DHT22 import DHT22  
4      from AS312 import AS312  
5      from DFR0026 import DFR0026  
6      import glob  
7      import re  
8      import subprocess
```

2. define our class sensors

```
1      class sensor_data:  
2          def __init__(self):  
3              self.dht22 = DHT22()  
4              self.humidity, self.temperature = self.dht22.  
                  Read_DHT22_data()  
5              self.AS312 = AS312(17)  
6              self.motion_detected = AS312.read_state()  
7              self.DF0026 = DFR0026()  
8              self.light_value = self.DF0026.Read_data()  
9              self.fname = "sensor_data.csv"
```

3. We write and append our data to the csv file

```
1      def write_append_csv(self):  
2          data = { "Timestamp" : self.timestamp,  
3                  "Temperature(oc)" : self.Temperature,  
4                  "Humidity(%)" : self.humidity,  
5                  "Light(lux)" : self.light_value,  
6                  "Motion_Detected": self.motion_detected  
7                  }  
8          df = pd.DataFrame(data)  
9          if glob.glob(self.fname):  
10             df.to_csv(self.fname, mode='a' , index=False, header  
11                        =False)  
12          else:  
13             df.to_csv(self.fname, mode='w' , index=False)
```

4. Next we define our variables for testing memory

```
1 class Memory_tester():
2     def __init__(self):
3         self.units={"K":10e3,"M": 10e6,"G":10e9}
4         self.regex ="\d{4}\.\.[0-9]{1,3}[K,M,G]"
5         self.fname=" ../bash_scrpits/memorytest.sh"
6         self.output_bash=subprocess.check_output(["bash",
            self.fname],universal_newlines=True)
```

5. next we check our memory

```
1     def check_memory(self):
2         try:
3             if re.search(self.regex,self.output_bash):
4                 value,unit=match.group(0).split()
5                 try:
6                     return float(value)*self.units[unit]
7                 except KeyError:
8                     raise ValueError(f"unknown unit:{unit}")
9
10        except subprocess.CalledProcessError as e:
11            raise ValueError(f"Error running script:{e.output}")
```

6. we then make an error if its using 20 percent memory

```
1     def error_check(self):
2         mem=self.check_memory()
3         max=32*10e9
4         if mem >= 0.2* max:
5             raise MemoryError("memory on pi is about to be
                used up")
```

7. to make sure our class run from another python file

```
1     if __name__=="__main__":
2         sensor_data()
3         Memory_tester()
```

3.5.4 Radio module

This section is based off the github repository: <https://github.com/sbcshop/Lora-HAT-for-Raspberry-Pi>
here are the following approach for this module

1. First import the following libraries:

```
1     #!/home/mistaherd/Documents/Github/
2     meshnetwork_in_forest/env/lib/python3.11
3     import time
4     import serial
```

```

4         import pandas as pd
5         import numpy as np
6         import threading
7         import base64
8         from memory_mangment import sensor_data

```

2. we define our class and its constants

```

1         class Transciever:
2             def __init__(self):
3                 self.transceive_ser=serial.Serial(port='/dev/
                    ttyS0',baudrate=9600,parity=serial.
                        PARITY_NONE,stopbits=serial.STOPBITS_ONE,
                            bytesize=serial.EIGHTBITS,timeout=1)
4                 self.message="Hello_world!"
5                 self.chunk_size=240
6                 self.txt_fname="/home/mistaherd/Documents/
                    Github/meshnetwork_in_forest/Tests/
                        transmitted_text.txt"
7                 self.png_fname="/home/mistaherd/Documents/
                    Github/meshnetwork_in_forest/Images_camera
                        /camera_output_2024-05-19_13_25_18.png"
8                 self.csv_fname=sensor_data().fname
9                 self.timelimit=time.time()+6
10                self.recived=self.transceive_ser.in_waiting
11                self.event=threading.Event()

```

Where "self.transceive_ser" set our serial port up which in linux is '/dev/ttyS0' we can control timeout to be 6 seconds but for this section well keep it at 1,"self.event" it seeing a threing event to occur in the code "

3. Setup a interrupt

```

1         def serial_interrupt(self):
2             if self.recived:
3                 self.event.set()

```

If there is any information to be sent on the wireless channel this will stop all operations

4. Make a process that will calculated the bytes before sending the data

```

1         def cal_bytes(self)-> int:
2             return len([bytes(self.data[i], 'utf-8').hex() for
                    i in range(len(self.data))])

```

5. Test sending and receiving hello world!

```

1         def transceive_test_message(self,transceive:bool):
2             """send /recive a hello world"""
3             if transceive:
4                 # self.message
5                 #transmite

```

```

6         self.transceive_ser.write(bytes(self.message,
7             'utf-8'))
8         time.sleep(0.2)
9         if not transceive:
10             while time.time() < self.reive_timelimit:
11                 self.transceive_ser.attachInterrupt(self.
12                     serial_interrupt)
13                 if self.event.is_set():
14                     data_read=self.transceive_ser.
15                         readline()
16                     data=data_read.decode("utf-8")
17                     print("message_ received:",data)
18                     self.event.clear()

```

6. test send/receiving a txt file

```

1         def transceive_test_txt_file(self,transceive:bool):
2             """send /revive a txt file"""
3             if transceive:
4                 with open(self.txt_fname,'r') as f:
5                     data=f.read()
6
7                 self.transceive_ser.write(bytes(data,'utf-8')
8                     )
9                 time.sleep(0.2)
10            if not transceive:
11                while time.time() < self.timelimit:
12                    self.transceive_ser.attachInterrupt(self.
13                        serial_interrupt)
14                    if self.event.is_set():
15                        data_read=self.transceive_ser.
16                            readline()
17                        data=data_read.decode("utf-8")
18                        print("message_ received:",data)
19                        self.event.clear()
20                        return data

```

7. Test sending and revecing csv file

```

1         def transceive_test_csv(self,transceive:bool):
2             if transceive:
3                 with open('/home/mistaherd/Documents/Github/
4                     meshnetwork_in_forest/main/sensor_data.csv
5                     ','r') as f:
6                     data=f.readlines()
7                     data=''.join(data)
8                     lora.write(bytes(data,'utf-8'))
9                     time.sleep(0.2)
10            if not transceive:
11                while time.time() < self.timelimit:
12                    self.transceive_ser.attachInterrupt(self.
13                        serial_interrupt)

```

```

11         if self.event.is_set():
12             data=self.transceive_ser.readlines()
13             output=[data[i].decode()[:-1].split("
",) for i in range(len(data))]
14             df=pd.DataFrame(output)
15             self.event.clear()
16             return df

```

8. Test sending and receiving an image file

```

1     def Transceive_png_file(self,transceive:bool):
2         """Transmit a PNG file"""
3         if transceive:
4             with open(self.png_fname, 'rb') as f:
5                 self.data = f.read()
6                 if self.cal_bytes()>self.chunk_size:
7                     chunks=[data[i:i+self.chunk_size] for i
8                             in range(0,len(self.data),self.
9                                 chunk_size)]
10                    for chunk in range(len(chunks)):
11                        encoded_chunk=base64.b64encode(chunk)
12                        self.transceive_ser.write(
13                            encoded_chunk)
14                else:
15                    raise ValueError("Image_file_must_be_
16                        corrupted")
17            if not transceive:
18                output=[]
19                self.transceive_ser.attachInterrupt(self.
20                    serial_interrupt)
21                if self.event.is_set():
22                    while(self.transceive_ser.read() != b''):
23                        data_read = self.transceive_ser.read
24                            ()
25                        print("bytes_received_%a"%data_read)
26                        output.append(base64.b64decode(
27                            data_read))
28                output=b"".join(output)
29                self.event.clear()
30                return output

```

9. For demo make sure to define the following:

```

1     def transive_choice(self,arugement):
2         """ run this for demo"""
3         if not self.event.is_set():
4             #transmit something
5             self.transmit=True
6             choice ={
7                 1:lambda :self.transceive_test_message(
8                     self.transmit),

```

```

8             2:lambda :self.transceive_test_txt_file(
                self.transmit),
9             3:lambda :self.transceive_test_csv(self.
                transmit),
10            4:lambda :self.Transceiveie_png_file(self.
                transmit)}
11        choice[arugement]()
12        #revived somthing
13        self.transmit=False
14        choice[self.user_message]()

```

10. have file as a module

```

1        if __name__=='__main__':
2            Transciever()

```

The following is the process of develop:

- To make test that will be there for the coding section of the project

this section will discuss the following for testing:

1. 1 x DHT22
2. 1 x DFR0026
3. 1 x AS312
4. 1 x SB Components LoRa HAT for Raspberry Pi
5. 1 x MCP3008
6. 1 x Raspberry Pi VR 220 Camera
7. 1 x Li-polymer Battery HAT

DHT22

According to the data sheet ? seen as the data is 8 bits and the range at which this operates at -40 to 80°C for temperature meaning we have at least 7 bit in the exponent to represent the measured value. to represent the high end of this sensor i used the following calculation:

$$2^6 + 2^4 = 80$$

which mean we have a 2 bits dedicated to decimal place so the high temperature to be 80.3°C for the lowest temp we have 6 bits to represent - 40 due to 2s complement so lowest will be -40.3°C so with that that stablish we must make a unit that will do the following:

1. Test if the output is a float
2. Test the high end of the temp sensor so it reads 80.3 as the highest
3. Test for the lowest temp around

be sure to follow steps for folder setup follow instructions on page ?? . we get the following sample code:

Listing 3.1: sample test initial code

```
1 import unittest
2 from protest import Read_DHT22
3 class test_project_code(unittest.TestCase):
4     def test_DHT_22_temp_output_type(self):
5         self.assertIsInstance(Read_DHT22, float)
6     def test_DHT22_temp_range(self):
7         self.assertGreaterEqual(Read_DHT22, -30.3)
8         self.assertLessEqual(Read_DHT22, 80.3)
```

This code imports unittest . the from DHT22 is a python files we can install functions from other python files this can be useful for testing purposes then we initialized a test class call Unittest.test as our first function of the class we check if the number of the output is a float or not this is for testing temperature the next function we test for is the range look at the data sheet online for . This code is simply testing the limits of the DHT22 for humidity the Data sheet which ranges from 0 to 100 % we want to test for the following:

1. Test if the recorded output is a tuple
2. Test if the temperature recorded is a float or integer
3. Test if the temperature recorded is in the range from -30 to 80.3
4. Test if the humidity recorded is a float or integer
5. Test if the humidity recorded is between 0 and 100

this lead to the following code

Listing 3.2: sample test for DHT22

```

1 import unittest
2 from DHT22 import DHT22
3 dht22_instance=DHT22()
4 class test_project_code(unittest.TestCase):
5     hum,temp=Read_DHT22(2)
6     def test_DHT22_output_type(self):
7         self.assertIsInstance(dht22_instance.Read_DHT22_data,
8                                tuple)
9
10    def test_DHT_22_temp_output_type(self):
11        self.assertIsInstance(temp, (int,float) )
12
13    def test_DHT22_temp_range(self):
14        self.assertGreaterEqual(temp, -30.3)
15        self.assertLessEqual(temp, 80.3)

```

seen as we expect our sensor to print out a humidity and temp values we set the output to a tuple to test for this we use isInstacne which will test if its a tuple next we test for the limits of the humidity

DFR0026

According to the data sheet ? we must keep in mind that this component is connected to an ADC this will give me the following test conditions:

1. Test if the output is a dictionary with elements of a string and integer in it.
2. Test the range of this with the upper limit being 5v the analogue voltage meaning:

$$\begin{aligned}
 \text{output} &= \frac{2^n \cdot \text{Analogue Input value}}{\text{Reference voltage}} \\
 &= \frac{2^{16} \cdot 5}{5} = 65536
 \end{aligned}$$

3. test the lover limit being 3.3v as the analogue $\frac{2^{16} \cdot 3.3}{5} = 43253$

Listing 3.3: unit test for DFR0026 and MCP3008

```

1 import unittest
2 from DFR0026 import DFR0026
3 class test_project_code(unittest.TestCase):
4     def test_DFR0026_out_type(self):

```



```

5         self.assertIsInstance(DFR0026().read_voltage(),dict[str,
           int])
6     def test_DFR0026_out_range(self):
7         self.assertLessEqual(DFR0026().read_voltage(),65536)
8         self.assertGreaterEqual(DFR0026().read_voltage(),43253)

```

AS312

for this section we want our tests to be the following:

1. test if output type is boolean

we can now add to the snippet :

Listing 3.4: unit test for AS312

```

1     import unittest
2     from AS312 import AS312
3     AS312_instance=AS312()
4     class test_project_code(unittest.TestCase):
5     def test_AS312_out_type(self):
6         self.assertIsInstance(AS312_instance.read_state, bool)

```

seen as this is a motion sensor our output will be true or false.

Raspberry Pi VR 220 Camera

according to the data sheet ? The resolution to it uses is 1080p50 which is 1920x1080p so our tests will have to incorporate the following:

1. Test if the file can run

This would lead to the following code snippet.

Listing 3.5: camera unit test

```

1     def test_Raspberry_Pi_VR220_out_shape(self):
2         self.assertEqual(camera_obj.run.returncode, 0)

```

This function check the pixel count or resolution

memory module

in this section will discuss the following:

1. silicon power 32GB For this use a bash script(see this on page 24) to test the size in a certain range for the silicon SD card
2. which will be 0B to 32GB

```

1         def Test_memory_silicon_power_32GB(self):
2             self.assertLessEqual(memorytest_obj.check_memory
3                                   ,32e9)
3             self.assertGreaterEqual(memorytest_obj.
3                                     check_memory,0)

```

SB Components LoRa HAT

for this section we want to test the following:

1. Test the serial connection
2. Test the serial interrupt
3. Test the sending/receiving of a message like "hello world"
4. Test the sending/receiving of a text file
5. Test the sending/receiving of a csv file
6. Test the sending/receiving of a image file

This will give the following code:

```
1  import unittest
2  from Radiomodule import Transciever
3  Transciever_instance=Transciever()
4  class test_project_code(unittest.TestCase):
5      def test_serial_connection(self):
6          self.assertIsInstance(Transciever_instance.
              transceive_ser,serial.Serial)
7      def test_serial_interrupt(self):
8          self.assertEqual(Transciever_instance.event.is_set()
              ,(False,True))
9      def test_transciever_test_message(self):
10         message=Transciever_instance.message
11         Transciever_instance.transceive_test_message(True)
12         received_message=Transciever_instance.
              transceive_test_message(False)
13         self.assertEqual(message,received_message)
14     def test_transciever_test_txt_file(self):
15         txt_fname=Transciever_instance.txt_fname
16         with open(txt_file,'r') as f:
17             expected_txt=f.read()
18         Transciever_instance.transceive_test_txt_file(True)
19         received_txt_file=Transciever_instance.
              transceive_test_txt_file(False)
20         self.assertEqual(expected_txt,received_txt_file)
21     def test_transciever_test_csv(self):
22         csv_fname=Transciever_instance.csv_fname
23         expected_df=pd.read_csv(csv_fname)
24         Transciever_instance.transceive_test_csv(True)
25         reviced_df=Transciever_instance.transceive_test_csv(
              False)
26         self.assertEqual(expected_df,reviced_df)
27     def test_transciever_test_img_file(self):
28         img_fname=Transciever_instance.png_fname
29         with open(img_fname,'rb') as f:
30             expted_out=f.read()
31         Transciever_instance.Transceive_png_file(True)
```

```

32         received_bin=Transciever_instance.Transcevie_png_file
           (False)
33         self.assertEqual(expted_out,received_bin)
34     if __name__ == '__main__':
35         unittest.main()

```

Unit test iterations

This section will discuss the iterations of unit test the following are the iterations of our unit testing:

1. In the first iteration the following was worked on:

- DHT22
- DFR0026
- AS312
- Camera
- turbo 1GB
- silicon power 32Gb

This was formed in the when the lit review was written

2. most the iterations where coming out the code and testing each section

3.6 Data Analysis Methods

In this section we will Discuss the tool we use to analyze the sensor data the following will be discussed:

1. Met Eireann weather forecast API
2. Matplotlib

3.6.1 Met Eireann weather forecast API

we can compare our humidity and temperature and compare this will the output of this API

3.6.2 Matplotlib

we can use this libraries to plot our changes in the data.

3.7 Timeline

The research phase will be conducted from 20/09/2023 to May 20, 2024. The prosed impanation phase is from january till May 20, 2024. The project file timeline can be visualized as the following: This was gotten from github. The project code can be visualized as the following:

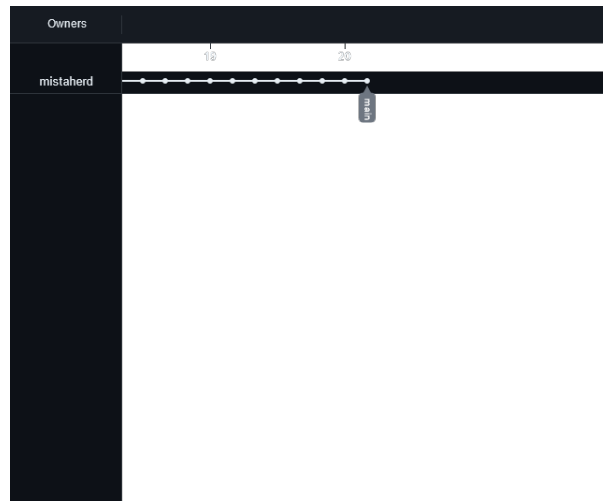


Figure 3.3: Visualized timeline of project files versions

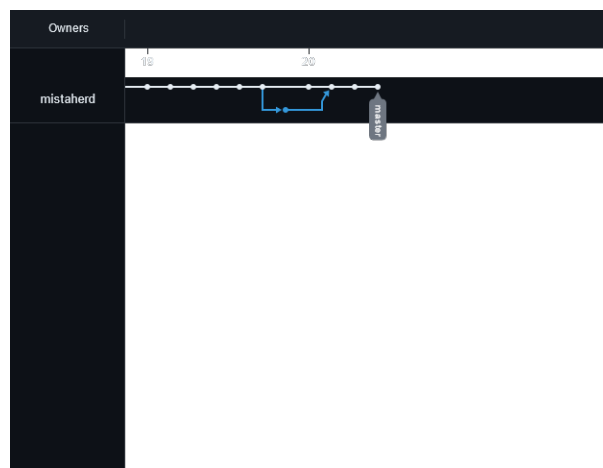


Figure 3.4: Visualized timeline of project programming timeline

Chapter 4

Results

In this section we will be showing results for different aspects of this project this will include the following:

1. Recorded data from sensors
2. Recorded data from transceiver
3. Recorded data from testing the mesh network

4.1 Recorded data from sensors

In this section will have tables from the following components:

1. DHT22 **heat and temp**
2. AS312 **Motion**
3. DFR0026 **Light**
4. Raspberry Pi VR 220 **Camera**

4.1.1 DHT22

Results during prototypeing

| date/time of record | Temperature | Humidity |
|---------------------|-------------|----------|
| 2024-05-03_17-31-52 | 20 | 49 |
| 2024-05-03_17-43-54 | 20 | 49 |
| 2024-05-03_17-31-52 | 20 | 49 |
| 2024-05-03_17-43-54 | 20 | 49 |

Table 4.1: Recorded data from DHT22 on the 5th of march

If this is plotted the following is gotten: last we tested if our code satisfies our python code after testing the unit test code we updated see the following message

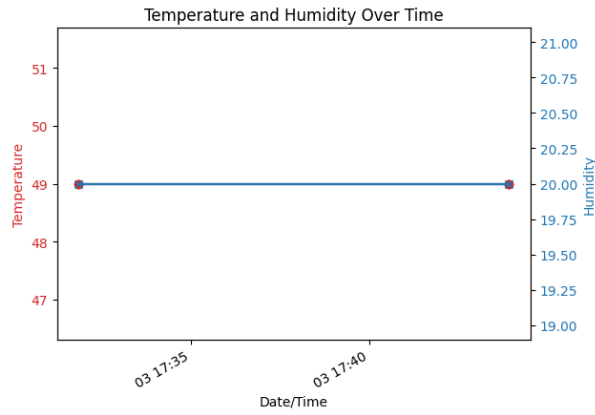


Figure 4.1: Temperature and Humidity plotted overtime

```
-----
Ran 5 tests in 0.002s
FAILED (failures=1)
Lost access to message queue
```

Figure 4.2: unit test message for DHT22 module

4.1.2 AS312

| date/time of record | motion detected(yes/no) |
|---------------------|-------------------------|
| 2024-03-25_15-02-57 | False |
| 2024-03-25_15-04-37 | True |
| 2024-05-03_18-07-51 | False |
| 2024-05-03_18-18-37 | True |

Table 4.2: Recorded data from AS312 on the May 20, 2024

4.1.3 DFR0026

For first test we got the following table: If this is plotted we get the following:

| Date/time of record | lux values(lux) |
|---------------------|-----------------|
| 2024-03-25_15-02-57 | 940 |
| 2024-03-25_15-03-13 | 945 |
| 2024-03-25_15-04-37 | 4963 |
| 2024-05-03_18-54-57 | 1284 |

Table 4.3: Recorded data from DFR0026 on the 25th of march 2024

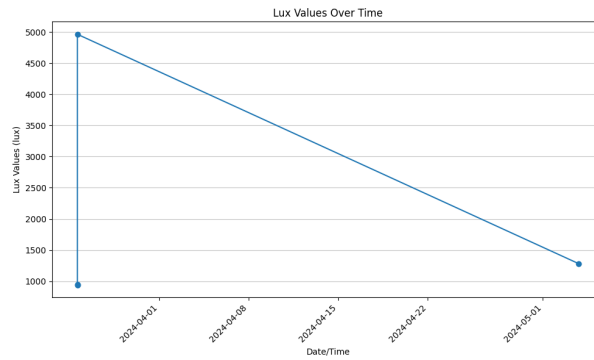


Figure 4.3: lux values overtime

4.1.4 Raspberry Pi VR 220

When testing the Raspberry Pi VR 220 the following picture was taken:

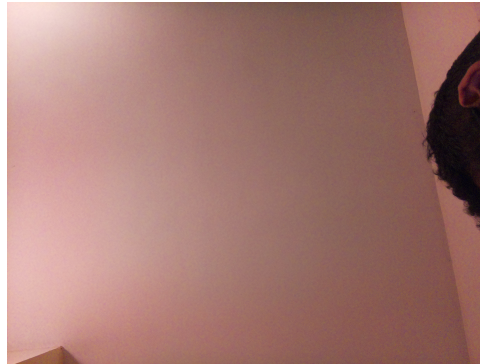


Figure 4.4: A photo from 25th of march 2024

4.2 Recorded data from transceiver

When testing the radio module the follow was tested:

1. Sending a message across the serial
 2. Sending a txt file across the serial
 3. Sending a csv file across the serial
 4. Sending a image file across the serial
- no results where obtained due to factor on 55

Chapter 5

Discussion

In this section we will discuss the following:

1. Results of the sensor data
2. Results of the camera
3. Results of the Lora module

5.1 Discussion of results of Sensor data

in this section, the data gathered from the sensor will be discussed Note:(**all tests here were conducted indoors**):

1. DHT22 (Temperature & Humidity):

On page ?? compared to the average room temperature which is 20°C, The range of humidity in a room is from 30% to 60% in the plot on page 53 note in the file sensor_data.csv there entries that are 0, this is due to testing different components and make these values as 0. during the development stage of this project we ran unit tests this module as seen on page 53.

2. AS312 (Motion):

As seen on page 53 this will output a table that is True when an object is detected and false when no object is detected

3. DFR0026 (Lux): As seen on page ?? is a table full of lux values according to this link we ideally want a lux value of 800 to 1700 lux which satisfies these conditions.

5.2 Discussion of Results of camera data

As seen on page ?? which shows an image that was taken by the camera attached to the pi

5.3 Discussion of Lora module

While updating the repository for this section the Pi's SD cards were corrupted due to a faulty binary file which was later fixed but this error led to the following error message:


```

(new_env) mistaherd@pc:~/Documents/Github/meshnetwork_in_forest/main$ (new_env) mistaherd@pc:~/Documents/Github/meshnetwork_in_forest/main$ python main.py
Traceback (most recent call last):
  File "/home/mistaherd/Documents/Github/meshnetwork_in_forest/main/main.py", line 2, in <module>
    from memory_mangement import sensor_data, Memory_tester
  File "/home/mistaherd/Documents/Github/meshnetwork_in_forest/main/memory_mangement.py", line 2, in <module>
    import pandas as pd
  File "/home/mistaherd/Documents/Github/meshnetwork_in_forest/new_env/lib/python3.9/site-packages/pandas/_init_.py", line 19, in <module>
    raise ImportError(
ImportError: Unable to import required dependencies:
numpy: Error importing numpy: you should not try to import numpy from

```

Figure 5.1: environment error message

Chapter 6

Conclusion & Future Work

6.1 Conclusion

In this final year project, We have explored the implementation and development of mesh networks within the challenging environment of a forest. Through:

1. The wireless commutation environment
2. The desired characteristics of the sensors
3. The available electronic devices for this limitation
4. The limited hardware available
5. The fundamental software needed
6. The Setup of the software
7. The examination of the programming software needed
8. The test before the writing of any code
9. The discussion of how to record data
10. The Limitations

6.1.1 Key Contributions

This project has contributed significantly to the understanding of the process of developing mesh networks in forest settings by:

- **Developing a Model:** In this we used serial communication to test the nature of which the network will send/ receive data from the network
- **Addressing Challenges:** in a line of sight environment the sending and receiving of images is a hard task due to how large the file is byte by byte
- **Performance Evaluation:** every method eventually worked via serial but image files had the most problems
- **Practical Implications:** Schedule the sending and receiving of data we can use this to extract the sensor data

- **Familiarity with different tools:** This project forces the project implementation to use the terminal in Linux, bash, In Linux the concept of Dev files are files where the port can be defined for example port 80 on the window would be `/dev/tty80`, also the addition of tools like ssh which allowed remote connection to the Pi board, batch files used to make these connections easier, A stronger familiarity with git was gained due to the keeping the project version controlled, A display of further knowledge of python was gained due to project such as using virtual environments and how to properly structure OPP program, An understanding of TDD for coding projects was gained where the implementor is forced to use these methods

6.2 Sources of Error

In this section, the following will be discussed:

1. Radio module
2. MCP3008
3. Time management
4. Lack of knowledge of Linux OS
5. Camera
6. Lack of hat for sensors

6.2.1 Radio Module

This section will discuss the problems noticed while working on this:

1. When trying to look for radio modules for the MM2 Series 900 MHz this part couldn't be ordered due to the vendor only accepting business customers this cost 4 weeks of research to look for another module
2. Once the appropriate modules were after looking and noticing that the documentation was very poor it did help as well.
3. One avenue not expected of this was enabling the serial. So once we enable via `sudo raspi-config`, and use the code from this link (use `lora_receiver.py` and `lora_transmitter.py`) when we get the following error:

```
sudo] password for mistaherd:
tho: write error: Input/output error
```

Figure 6.1: Issue when trying to write to serial

This issue took a lot of time to fix a simple `chmod rw` operation will fix the issue which should be indicated in the documentation but there isn't.

4. After getting a working demo on all use cases of the module the image file caused the most issues this was due to the nature of how large the large data in an image file further research should have been put into how an image is sent across a communication channel.
5. The lack of documentation meant time was spent looking at the components on board and finding how they worked.

6.2.2 MCP3008

In this section the Errors that occurred with this module are the following:

1. When it came to ordering this part, There were issues with the venter so another ADC had to be chosen this caused another few weeks to go by.

6.2.3 Lack of knowledge of Linux os

This section will discuss the errors that occurred theses are the following:

1. Since the lack of experience with Linux is evident this led to time spent learning about certain commands that will help with the project concepts such as piping commands nmap where foreign before this project a lot of time was spent learning the best tools for this project

6.2.4 Camera

This Section will discuss the errors that occurred with this module these are the following:

1. When using the python file "camera.py" this wouldn't comply due to how the old library is no longer supported this costs around 2 weeks due to finding different forms and tracking the right library down eventually. the decision was made to make a bash command to run the camera

6.2.5 Lack of hat for sensors

This Section will discuss the issues of this which are the following:

1. When wiring up all the sensors it took several minutes to wire the devices up which led to miss wiring the temp sensor which blew leading to an order for a replacement to be made
2. This issue along with the could have led to the code being simpler in nature them what is present in the repository

6.3 Future Work

While this project hasn't achieved its core objectives, several avenues for future research and development remain open:

- **Sockets:** The project could have ventured into socket programming, A socket is an endpoint of a two-way communication link between two programs running on the network. we picked serial communication for testing but this is where the main server and client can defined.
- **Helpful Tools:** Linux has a wide range of networking tools such as:
 - **Nmap** is a network scanner designed to discover hosts and services on a computer network. It sends packets and analyzes the responses to gather information
 - **ifconfig** Which provides extensive control over interfaces, addresses and routes
 - **traceroute** Traces the route packets take to reach a destination
 - **route** Displays or manipulates the kernel's IP routing table
 - **arp** Displays and manages the Address Resolution Protocol (ARP) cache, which maps IP addresses to MAC addresses

- **tcpdump** Captures and analyzes network traffic, useful for diagnosing network issues.
- **iftop** Displays a real-time bandwidth monitor for network interfaces.
- **Energy Efficiency:** Investigate further what can be done to make our system more energy effective i.e pipelining the sensor data
- **Security:** Investigate how to make our data secure via RSA and different protocols
- **PCB board :** After testing the board one problem was wiring up the sensor a potential for making a custom Pi hat that will provide an easy way of connecting our sensor

6.3.1 Final Remarks

The deployment of mesh networks in forest environments holds immense promise. The work provided here doesn't outline how to achieve this but can be viewed as a method to achieving this nature

Chapter 7

Appendix

Appendix A

Python Scripts

A.1 Python Scripts

A.1.1 DHT22

Listing A.1: DHT22code

```
1 #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/  
   python3.11  
2  import adafruit_dht  
3  import board  
4  import datetime  
5  import pandas as pd  
6  class DHT22:  
7  ##Set DATA pin to pin 4  
8      def __init__(self):  
9          # self.dhtDevice =adafruit_dht.DHT22(board.D4)  
10         self.dhtDevice =adafruit_dht.DHT11(board.D4)  
11     def Read_DHT22_data(self)-> tuple[float,float,str]:  
12         try:  
13             Humidity=self.dhtDevice.humidity  
14             Temperature=self.dhtDevice.temperature  
15             timestamp =datetime.datetime.now()  
16             timestamp = timestamp.strftime("%Y-%m-%d_%H:%M:%S")  
17             return Temperature, Humidity, timestamp  
18         except RuntimeError as e:  
19             print(f"Error reading sensor: {e}")  
20             return None, None  
21     def write_to_csv(self, filename: str):  
22         temperature, humidity, timestamp = self.Read_DHT22_data()  
23         if temperature is not None and humidity is not None and  
24             timestamp is not None:  
25             data = [(temperature, humidity, timestamp)]  
26             df = pd.DataFrame(data, columns=['Temperature', 'Humidity', 'Timestamp'])  
27             df.to_csv(filename, index=False)  
28         else:  
29             print("Failed to retrieve data from sensor. Data not  
                written to CSV.")  
30     dht_sensor = DHT22()
```

```
30 | dht_sensor.write_to_csv("sensor_data.csv")
```


A.1.2 AS312

Listing A.2: code for AS312

```
1 #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/  
   python3.11  
2  import RPi.GPIO as GPIO  
3  import time  
4  import datetime  
5  import pandas as pd  
6  #pin 17  
7  class AS312:  
8      def __init__(self, pin_number:int):  
9          self.pin_number=pin_number  
10         self.GPIO=GPIO  
11         self.GPIO.setmode(GPIO.BCM)  
12         self.GPIO.setup(self.pin_number, GPIO.IN)  
13         self.current_state=0  
14         self.timestamp=datetime.datetime.now().strftime("%Y-%m-%d  
            %H:%M:%S")  
15     def read_state(self)->int:  
16         self.current_state =self.GPIO.input(self.pin_number)  
17         return self.current_state  
18     def append_data(self):  
19         data={  
20             "Motion_Dectected": [self.current_state],  
21             "Timestamp": [self.timestamp]  
22         }  
23         df =pd.DataFrame(data)  
24         df.to_csv('sensor_data.csv',mode='a' ,index=False,header=  
            False)  
25  pir_sensor = AS312(17)  
26  try:  
27      time.sleep(0.1)  
28      current_state =pir_sensor.read_state()  
29      timestamp=pir_sensor.timestamp  
30      print("GPIO_pin%s_is%s" % (pir_sensor.pin_number,  
        current_state))  
31      if current_state == 1:  
32          print("Motion_dectected")  
33          pir_sensor.append_data()  
34  except KeyboardInterrupt:  
35      pass  
36  finally:  
37      GPIO.cleanup()
```

A.1.3 DFR0026

Listing A.3: Code for DFR00026

```
1 #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/  
   python3.11  
2  from DFRobot_ADS1115 import ADS1115  
3  import time  
4  class DFR0026():  
5      def __init__(self):  
6          self.ADS1115_REG_CONFIG_PGA_6_144V          = 0x00 # 6.144V  
              range = Gain 2/3  
7          self.ADS1115_REG_CONFIG_PGA_4_096V          = 0x02 # 4.096V  
              range = Gain 1  
8          self.ADS1115_REG_CONFIG_PGA_2_048V          = 0x04 # 2.048V  
              range = Gain 2 (default)  
9          self.ADS1115_REG_CONFIG_PGA_1_024V          = 0x06 # 1.024V  
              range = Gain 4  
10         self.ADS1115_REG_CONFIG_PGA_0_512V          = 0x08 # 0.512V  
              range = Gain 8  
11         self.ADS1115_REG_CONFIG_PGA_0_256V          = 0x0A # 0.256V  
              range = Gain 16  
12         self.ads1115 = ADS1115()  
13         self.ads1115.set_addr_ADS1115(0x48)  
14         self.ads1115.set_gain(self.ADS1115_REG_CONFIG_PGA_6_144V)  
15         self.adc_channel=0  
16         def read_voltage(self):  
17             return self.ads1115.read_voltage(self.adc_channel)  
18             #time.sleep(0.2) after read it  
19 light_vaule=DFR0026()  
20 print(light_vaule.read_voltage())
```

A.1.4 Camera

Listing A.4: Code for Camera

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/  
   python3.11  
2  from picamera import PiCamera  
3  from time import sleep  
4  from datetime import datetime  
5  class Raspberry_Pi_VR_220:  
6      def __init__(self):  
7          """setup an instan for the camera"""  
8          self.timestamp=datetime.now().strftime("%Y-%m-%d_%H:%M:%S  
          ")  
9          self.fname = '/home/mistaherd/Documents/Github/  
          meshnetwork_in_forest/{}.png'.format(self.timestamp)  
10         self.camera=PiCamera()  
11         self.timeamount=2  
12     def take_pic(self)-> str:  
13         """this will take a picture from camera"""  
14         self.camera.start_preview()  
15         sleep(self.timeamount)  
16         self.camera.capture(self.fname)  
17         self.stop_preview()  
18         return self.fname  
19 camera=Raspberry_Pi_VR_220()  
20 picture=camera.take_pic()
```

Camera alt

Listing A.5: Code for alternaive code for Camera

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/  
   lib/python3.11  
2  import subprocess  
3  class camera:  
4      def __init__(self):  
5          self.run=subprocess.run(["bash","/home/mistaherd/  
                                   Documents/Github/meshnetwork_in_forest/  
                                   bash_scrpits/camera.sh"])  
6  if __name__=="__main__":  
7      camera()
```

A.1.5 Memory management

Listing A.6: Code for memory mangement

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/lib/
   python3.11
2  import pandas as pd
3  from DHT22 import DHT22
4  from AS312 import AS312
5  from MCP3008 import DF0026
6  import glob
7  import re
8  import subprocess
9  class sensor_data:
10     def __init__(self):
11         self.dht22 = DHT22()
12         self.humidity,self.temperature,self.timestamp=self.dht22.
           Read_DHT22_data()
13         self.AS312=AS312(17)
14         self.motion_dected =AS312.read_state()
15         self.DF0026 =DF0026()
16         self.light_value=self.DF0026.Read_data()
17         self.fname="sensor_data.csv"
18     def write_append_csv(self):
19         data = { "Timestamp" : self.timestamp,
20                 "Temperature(oc)" : self.Temperature,
21                 "Humidity(%)" : self.humidity,
22                 "Light(lux)" :self.light_value,
23                 "Motion_Dected": self.motion_dected
24             }
25         df = pd.DataFrame(data)
26         if glob.glob(self.fname):
27             df.to_csv(self.fname,mode='a' ,index=False,header=
               False)
28         else:
29             df.to_csv(self.fname,mode='w' ,index=False)
30 class Memory_tester():
31     def __init__(self):
32         self.units={"K":10e3,"M": 10e6,"G":10e9}
33         self.regex = "\d{4}\.\.[0-9]{1,3}[K,M,G]"
34         self.fname="../bash_scrpits/memorytest.sh"
35         self.output_bash=subprocess.check_output(["bash",self.
           fname],universal_newlines=True)
36     def check_memory(self):
37         try:
38             if re.search(self.regex,self.output_bash):
39                 value,unit=match.group(0).split()
40                 try:
41                     return float(value)*self.units[unit]
42                 except KeyError:
43                     raise ValueError(f"unknown_unit:{unit}")
44
```

```

45         except subprocess.CalledProcessError as e:
46             raise ValueError(f"Error running script:{e.output}")
47     def error_check(self):
48         mem=self.check_memory()
49         max=32*10e9
50         if mem >= 0.2* max:
51             raise MemoryError("memory on pi is about to be used up")

```

A.1.6 Radio module

Listing A.7: Code for Radio module

```

1      #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/
      lib/python3.11
2      import time
3      import serial
4      import pandas as pd
5      import numpy as np
6      import threading
7      import subprocess
8      import base64
9      from memory_mangment import sensor_data
10     class Transciever:
11         def __init__(self):
12             self.transceive_ser=serial.Serial(port='/dev/ttyS0',
13                 baudrate=9600,parity=serial.PARITY_NONE,stopbits=
14                 serial.STOPBITS_ONE,bytesize=serial.EIGHTBITS,
15                 timeout=1)
16             self.message="Hello world!"
17             self.chunk_size=240
18             self.txt_fname="/home/mistaherd/Documents/Github/
19                 meshnetwork_in_forest/Tests/transmited_text.txt"
20             self.csv_fname=sensor_data().fname
21             self.timelimit=time.time()+6
22             self.recived=self.transceive_ser.in_waiting
23             self.event=threading.Event()
24         def serial_interrupt(self):
25             if self.recived:
26                 self.event.set()
27         def cal_bytes(self)-> int:
28             return len([bytes(self.data[i],'utf-8').hex() for i
29                 in range(len(self.data))])
30
31     # hello world
32     def transceive_test_message(self,transceive:bool):
33         """send /recive a hello world"""
34         if transceive:
35             # self.message
36             #transmite

```

```

32         self.transceive_ser.write(bytes(self.message,'utf
33         -8'))
34     time.sleep(0.2)
35     if not transceive:
36         while time.time()< self.reive_timelimit:
37             self.transceive_ser.attachInterrupt(self.
38             serial_interrupt)
39             if self.event.is_set():
40                 data_read=self.transceive_ser.readline()
41                 data=data_read.decode("utf-8")
42                 print("message received:",data)
43                 self.event.clear()
44
45     # Text file
46     def transceive_test_txt_file(self,transceive:bool):
47         """send /revive a txt file"""
48         if transceive:
49             with open(self.txt_fname,'r') as f:
50                 data=f.read()
51
52             self.transceive_ser.write(bytes(data,'utf-8'))
53             time.sleep(0.2)
54         if not transceive:
55             while time.time()< self.timelimit:
56                 self.transceive_ser.attachInterrupt(self.
57                 serial_interrupt)
58                 if self.event.is_set():
59                     data_read=self.transceive_ser.readline()
60                     data=data_read.decode("utf-8")
61                     print(data)
62
63     #test csv file
64     def transceive_test_csv(self,transceive:bool):
65         if transceive:
66             with open('/home/mistaherd/Documents/Github/
67             meshnetwork_in_forest/main/sensor_data.csv','r
68             ') as f:
69                 data=f.readlines()
70                 data=''.join(data)
71                 lora.write(bytes(data,'utf-8'))
72                 time.sleep(0.2)
73         if not transceive:
74             while time.time() <self.timelimit:
75                 self.transceive_ser.attachInterrupt(self.
76                 serial_interrupt)
77                 if self.event.is_set():
78                     data=self.transceive_ser.readlines()
79                     output=[data[i].decode()[:-1].split(",")
80                             for i in range(len(data))]
81                     df=pd.DataFrame(output)
82                     print(df)
83
84     #Test png,jpg

```

```

76     def Transcevie_png_file(self):
77         """Transmit a PNG file"""
78         if transceive:
79             with open(self.png_fname, 'rb') as f:
80                 data = f.read()
81                 chunks=[data[i:i+self.chunk_size] for i in range
82                     (0,len(data),self.chunk_size)]
83                 for chunk in range(len(chunks)):
84                     encoded_chunk=base64.b64encode(chunk)
85                     self.transceive_ser.write(encoded_chunk)
86         if not transceive:
87             output=[]
88             self.transceive_ser.attachInterrupt(self.
89                 serial_interrupt)
90             if self.event.is_set():
91                 while(self.transceive_ser.read() != b''):
92                     data_read = self.transceive_ser.read()
93                     print("bytes_ reviced_ %a"%data_read)
94                     output.append(base64.b64decode(data_read)
95                         )
96                 output=b"".join(output)
97                 with open("recived_img.png", 'wb') as f:
98                     f.write(output)
99     def transive_choice(self,arugement):
100         """ run this for demo"""
101         if not self.event.is_set():
102             #transmit something
103             self.transmit=True
104             choice ={
105                 1:lambda :self.transceive_test_message(self.
106                     transmit),
107                 2:lambda :self.transceive_test_txt_file(self.
108                     transmite),
109                 3:lambda :self.transceive_test_csv(self.
110                     transmit),
111                 4:lambda :self.Transcevie_png_file(self.
112                     transmit)}
113             choice[arugement]()
114             #revived somthing
115             self.transmit=False
116             choice[self.user_message]()
117 if __name__=='__main__':
118     Transciever()

```


Appendix B

TDD Script

B.1 TDD scripts

This section is for All the TDD sections of this report this section will share the TDD of the following:

1. DHT22
2. AS312
3. DFR0026
4. Raspberry Pi VR 220 Camera
5. Memory management
6. SB Components LoRa HAT for Raspberry Pi

B.1.1 DHT22

Listing B.1: DHT22 unit test

```
1      #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/  
      lib/python3.11  
2      from DHT22 import DHT22  
3      import unittest  
4      dht22_instance=DHT22()  
5      hum, temp,ts=dht22_instance.Read_DHT22_data()  
6      class test_project_code(unittest.TestCase):  
7          # DHT22  
8          def test_DHT22_output_type(self):  
9  
10             self.assertIsInstance(dht22_instance.Read_DHT22_data ,  
                                     tuple)  
11  
12             def test_DHT_22_temp_output_type(self):  
13                 self.assertIsInstance(temp, (int,float) )  
14  
15             def test_DHT22_temp_range(self):  
16                 self.assertGreaterEqual(temp,-30.3)  
17                 self.assertLessEqual(temp,80.3)  
18
```

```
19         def test_DHT22_hum_output_type(self):
20             self.assertIsInstance(hum,(int,float))
21
22         def test_DHT22_hum_range(self):
23             self.assertGreaterEqual(hum,0.0)
24             self.assertLessEqual(hum,100.0)
25     if __name__ == '__main__':
26         unittest.main()
```

B.1.2 AS312

Listing B.2: Code for unit test of AS312

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/  
   lib/python3.11  
2  import unittest  
3  from AS312 import AS312  
4  class test_project_code(unittest.TestCase):  
5      def test_AS312_out_type(self):  
6          self.assertIsInstance(AS312_instance.read_state, bool)  
7  if __name__ == '__main__':  
8      unittest.main()
```

B.1.3 DFR0026

Listing B.3: Code for unit test of DFR0026

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/  
   lib/python3.11  
2  import unittest  
3  from DFR0026 import DFR0026  
4  class test_project_code(unittest.TestCase):  
5      def test_DFR0026_out_type(self):  
6          self.assertIsInstance(DFR0026().read_voltage(),float)  
7      def test_DFR0026_out_range(self):  
8          self.assertLessEqual(DFR0026().read_voltage(),5)  
9          self.assertGreaterEqual(DFR0026().read_voltage(),0)  
10 if __name__ == '__main__':  
11     unittest.main()
```

B.1.4 Memory Management

Listing B.4: Code for unit test of memory module

```
1      #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/  
      lib/python3.11  
2      import unittest  
3      from memory_mangment import sensor_data,Memory_tester  
4      memorytest_obj=Memory_tester()  
5      class test_project_code(unittest.TestCase):  
6          def Test_memory_silicon_power_32GB(self):  
7              self.assertEqual(memorytest_obj.check_memory,32e9  
              )  
8              self.assertGreaterEqual(memorytest_obj.check_memory  
              ,0)  
9  
10     if __name__ == '__main__':  
11         unittest.main()
```

B.1.5 Radio Module

Listing B.5: unit test code for Radio module

```
1  #!/home/mistaherd/Documents/Github/meshnetwork_in_forest/env/
   lib/python3.11
2  import unittest
3  from Radiomodule import Transciever
4  Transciever_instance=Transciever()
5  class test_project_code(unittest.TestCase):
6      def test_serial_connection(self):
7          self.assertIsInstance(Transciever_instance.
                                transceive_ser,serial.Serial)
8      def test_serial_interrupt(self):
9          self.assertEqual(Transciever_instance.event.is_set()
                             ,(False,True))
10     def test_transciever_test_message(self):
11         message=Transciever_instance.message
12         Transciever_instance.transceive_test_message(True)
13         received_message=Transciever_instance.
            transceive_test_message(False)
14         self.assertEqual(message,received_message)
15     def test_transciever_test_txt_file(self):
16         txt_fname=Transciever_instance.txt_fname
17         with open(txt_file,'r') as f:
18             expected_txt=f.read()
19         Transciever_instance.transceive_test_txt_file(True)
20         received_txt_file=Transciever_instance.
            transceive_test_txt_file(False)
21         self.assertEqual(expected_txt,received_txt_file)
22     def test_transciever_test_csv(self):
23         csv_fname=Transciever_instance.csv_fname
24         expected_df=pd.read_csv(csv_fname)
25         Transciever_instance.transceive_test_csv(True)
26         reviced_df=Transciever_instance.transceive_test_csv(
            False)
27         self.assertEqual(expected_df,reviced_df)
28     def test_transciever_test_img_file(self):
29         img_fname=Transciever_instance.png_fname
30         with open(img_fname,'rb') as f:
31             expted_out=f.read()
32         Transciever_instance.Transceiveie_png_file(True)
33         received_bin=Transciever_instance.Transceiveie_png_file
            (False)
34         self.assertEqual(expted_out,received_bin)
35     if __name__ == '__main__':
36         unittest.main()
```

Appendix C

Bash scripts

C.1 Bashscripts

in this section we will have the following bash files:

1. Camerea
2. main
3. memorytest
4. radiomodule

C.1.1 Camerea

Listing C.1: Code for triggering the camerea

```
1  #!/bin/bash
2  timestamp=$(date +%Y-%m-%d_%H_%M_%S)
3  fname="camera_output_${timestamp}.png"
4  output_dir="Images_camera"
5  if [ ! -d "$output_dir" ]; then
6  # Create the directory if it doesn't exist
7  mkdir -p "$output_dir"
8  fi
9  rpicam-still --raw -o "$output_dir/$fname"
```

C.1.2 Main

Listing C.2: Code for running the main function

```
1  #!/bin/bash
2  is_root() {
3  if [[ $EUID -ne 0 ]]; then
4      echo "This script requires root privileges. Please run
      ↪ with sudo."
5      exit 1
6  fi
7  }
8  if [[ $1 -eq 0 ]]; then
9      echo "Error no arguments provided"
10     echo -e "enter what is transmitted:\n\r1:hello world \n\r2
      ↪ :text file \n\r3:csv file\n\r4:PNG\n\r"
11     exit 1
12 fi
13 # Call the is_root function to verify permissions
14 is_root
15 sudo chmod g+rw /dev/ttyS0
16 #get current time
17 current_time=$(date +%H:%M)
18 current_hour=$(echo $current_time | cut -d: -f 1)
19 previous_hour=$((current_hour-1))
20 while [ $current_time != "12:00" ]&&[ $current_time != "9:00"
      ↪ ]; do
21 if [ $current_time == "$current_hour:00" ]; then
22     # python Documents/Github/meshnetwork_in_forest/main/main
      ↪ .py $1
23     python main/main.py $1
24     echo "file ran successfully"
25 fi
26 break #because everyone needs a break sometime
27 done
```


C.1.3 Radio Module

Listing C.3: Code for the testing serial of the radio module

```
1  #!/bin/bash
2  #only use this for transceive module
3  # Function to check if the script is run with root privileges
4  is_root() {
5  if [[ $EUID -ne 0 ]]; then
6      echo "This script requires root privileges. Please run
7      ↪ with sudo."
8      exit 1
9  fi
10 }
11 # Call the is_root function to verify permissions
12 is_root
13 # Set appropriate permissions for /dev/ttyS0 (consider group
14 ↪ or user access)
15
16 sudo chmod g+rw /dev/ttyS0
17
18 if [[ "$1" == "1" ]]; then
19 python test_tranmitter.py
20 elif [[ "$1" == "0" ]]; then
21 python test_reciver.py
22 else
23 echo "Invalid argument. Please provide 1 (transmitter) or 0 (
24 ↪ receiver)."
25 exit 1
26 fi
```