Meshed Network

By: Johnson Domacasse (#4471709)

22 March 2024

Contents

1. Introduction: 3
2. Methods: 3
2.1 Implementation: 3
3. Results: 3
3.1 Simple messaging: 3
3.2 Measurements: 4
3.3 Nodes in network: 5
4. Discussion: 5
5. Bibliography: 6

1. Introduction:

The purpose of this research is to get a better understanding of meshed networks by implementing a proof of concept (PoC). The concept of a meshed network allows us to connect multiple devices (nodes) to communicate with each other under a single wireless local area network. Each node can communicate and deliver connection independently with each other without needing some sort of access point to relay this communication. The ESP-MESH network protocol allows the creation of a mesh network by using ESP32 hardware. There are other protocol to implement this using Wi-Fi. For the sake of this research, we will be implementing a mesh network using the painless mesh library that makes use of the ESP-MESH protocol**[1]**.

2. Methods:

This research was performed by using a number of tools that are already available. These include the following:

- The painless mesh library
- 3x ESP32-C3-DevKitM-1 boards
- 3 DHT22 sensors (& appropriate library for control)
- Arduino JSON library

2.1 Implementation:

We begin by implementing a program that is able to send a message from each node to every other node available in the network. See the results section for the output. After confirming that the messages were being received by every node, we went a bit further with the implementation by adding some sensors to it. We connect one DHT22 sensor to each node. This node will be responsible for taking the temperature and humidity measurements from this sensor and relaying this data to others nodes as well. We confirm that the sensors work by blowing warm air into them as well to see if the measurements are being properly updated. See the results section for these measurements**[2]**. Finally we use the painless mesh API to determine how many nodes are in the network. This was done first by using the “getNodeList” function. Be advised, this function returns the amount of nodes in the form of an 32b unsigned integer list. So we want the size of this list so that it wont give us errors. We add the size function to determine the amount**[3]**. According to the documentation, this function simply gets a list of all known nodes that are directly or indirectly connected to the current node. Meaning it will not detect the current node as “part” of the network**[4]**. For a better overview the result of the function was incremented by two to include the current node.

3. Results:

The format of the messages include where the message has come from and what the message is. Each node has their own personal way of saying hello. From there each node has a sensor connected to it which it will read the temperature and humidity using an additional library.

3.1 Simple messaging:

Here you can see the result of all three nodes receiving messages.

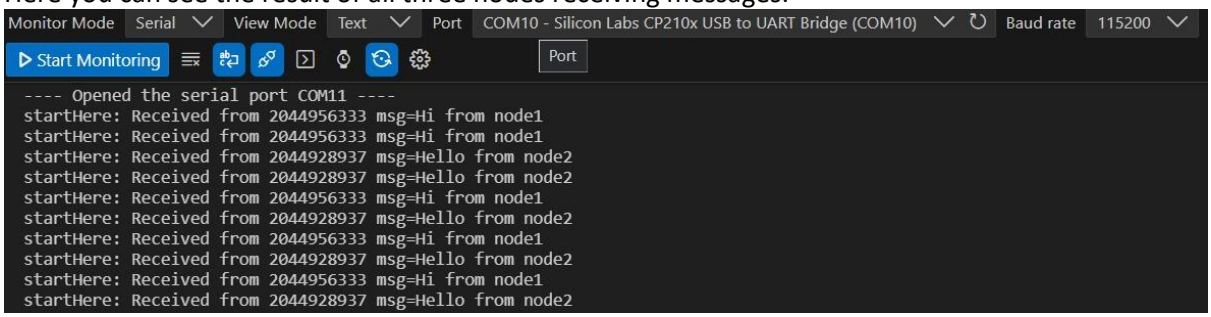


Figure 1: Node 3 receiving hello messages.

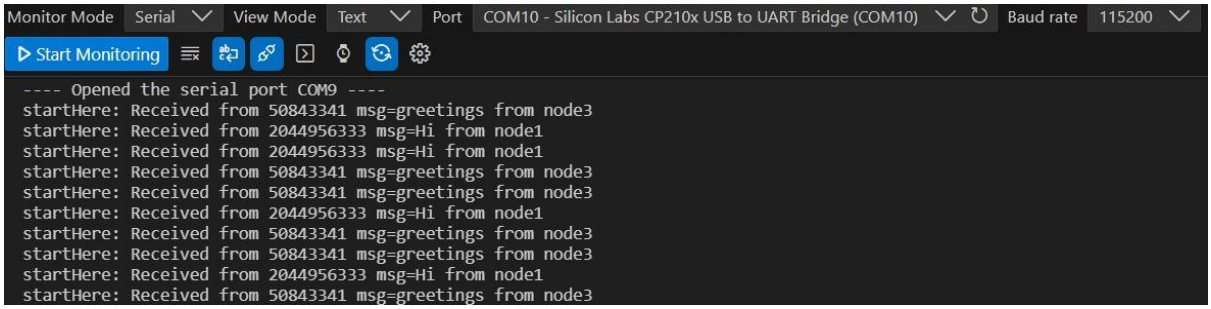


Figure 2: Node 2 receiving hello messages.

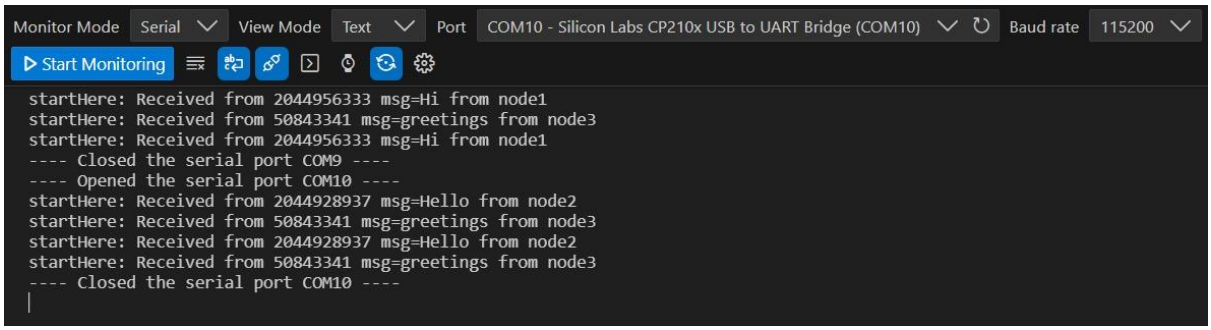


Figure 3: Node 1 receiving hello messages.

3.2Measurements:

Figure 4: Node 3 receiving measurements.

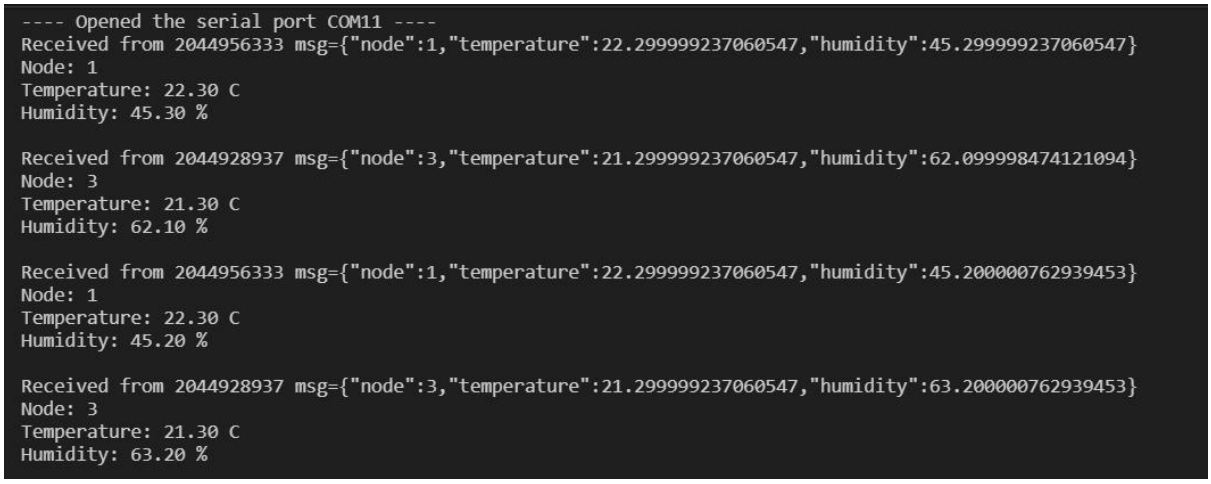
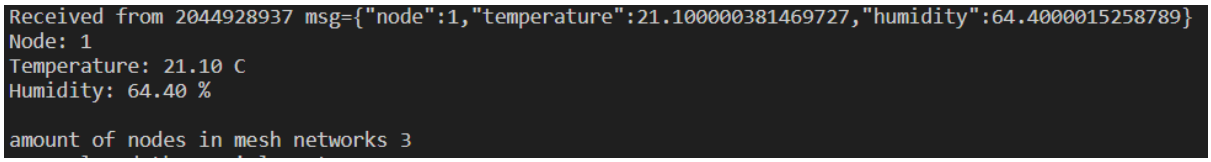


Figure 5: Node 2 receiving measurements.

Figure 6: Node 1 receiving measurements.

3.3 Nodes in network:



4. Discussion:

What this researched has proved is that multiple nodes can be connected into one meshed network so that the same message is sent over to every node. Some things that I realized while working was the library that was used for the actual mesh network is quite outdated. For example, on compile time, around 15 problems/warnings would pop up. All of these were from the painless mesh library and most of them had to do with a JSON function. For future purposes, I would check if there is an updated library to be used. Despite this the program did still run without any problems.

5. Bibliography:

[1] - Santos, S. (2020, November 23). *ESP-mesh with ESP32 and ESP8266: Getting started*. Random Nerd Tutorials. https://randomnerdtutorials.com/esp-mesh-esp32-esp8266-painlessmesh/

[2] - *Painlessmesh · GITLAB*. GitLab. (n.d.). https://gitlab.com/painlessMesh

[3] – *"no suitable conversion function from 'std::string' to 'const char *' exists."* Stack Overflow. (n.d.). https://stackoverflow.com/questions/59474515/no-suitable-conversion-function-from-stdstring-to-const-char-exists

[4] - *PainlessMesh/readme.md at master · GMAG11/Painlessmesh*. GitHub. (n.d.). https://github.com/gmag11/painlessMesh/blob/master/README.md