NodeMCU Based Mesh Controlled Network

Manish Bansal
Manish234@gmail.com
M. Tech Scholar, Department of Electrical & Electronics Engineering, BRCM CET, Bahal, (Haryana), India
Dr. Vivek Kumar
hodeee@brcm.edu.in
Professor & Head, Department of Electrical & Electronics Engineering, BRCM CET, Bahal, (Haryana), India
Mr. Sandeep Kumar
Sjakhar1989@gmail.com
Assistant Professor, Department of Electrical & Electronics Engineering, BRCM CET, Bahal, (Haryana), India

ABSTRACT

In this work the proposed system a mesh network around three IoT enabled NodeMCU development boards has been implemented. NodeMCU is an IoT enabled development board for rapid prototyping with on chip Wi-Fi chipset by Espressif. The hardware set-up along with the mesh networking protocol provides a distributed network and will collectively fulfil the purpose. This is an experimental work and the developed prototype has validated the research work reported here in this dissertation work. Solution has been provided for each problem statement that has been discussed and the research has been carried out in a phased manner.

Keywords: IoT, Web server, NodeMCU, Home Automation

INTRODUCTION

Here in this work efforts were made to design, develop and establish a Wireless-Local-Area-Network (WLAN) between different nodes connected in a Mesh network topology specifically for IoT applications. Here a hardware-based demonstration was made for the implementation of mesh network topology to establish connectivity between multiple Wi-Fi enabled NodeMCU devices for the transfer of sensors data without using the internet and without using any router in between. Here as this system was connected in a mesh network topology, there was no central node or parent node in this system instead here each node communicates with every other node available. It was observed that the system was capable of establishing connectivity between each node automatically by making use of same ssid, password and port address for each node just like a wireless ad-hoc network. Servers, clients, and the infrastructure that connects them are still the foundation of the Internet. The phrase "NodeMCU" is an abbreviation again for the term "node" and "microcontroller" (micro-controller unit). It is built by the ESP8266 Wi-Fi SoC from Express if Systems (System on Chip). The firmware as well as the experimental board designs are free to the source. The ESP-12F Wi-Fi device has an ESP8266 core processor in smaller module sizes. Tensilica L106 has an industry-leading 32-bit MCU micro with a 16-bit short form, clock frequencies of 80 MHz and 160 MHz, RTOS compatibility, as well as and an embedded Wi-Fi on-board antenna. The module complies the with IEEE802.11 b/g/n standard and the whole TCP/IP protocol stack. It provides unequalled versatility in incorporating Wi-Fi abilities into other systems or serving as a stand-alone application at the lowest possible cost and with the least amount of room necessary. The ESP8266EX provides a complete and ego Wi-Fi network of social; it may host the program or offload Wi-Fi networking activities from some other application processor. The ESP8266EX is connected to external sensors as well as other implementation devices through its GPIOs.

The Internet of Things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers

(UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. An IoT ecosystem consists of web-enabled smart devices that use embedded systems, such as processors, sensors and communication hardware, to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analysed or analysed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

EXPERIMENTAL SET-UP

As it could be observed that there were three different devices/ nodes in this system. Therefore, three separate hardware units centered on NodeMCU modules were designed and developed. As shown below in Figure-1 a NodeMCU module was interfaced to two push buttons and an OLED display module. The push buttons acted as system inputs and OLED acted as system output.

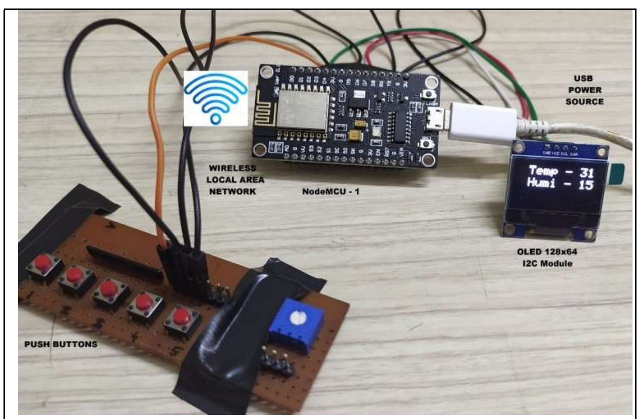

Figure-1 NodeMCU-1 Section

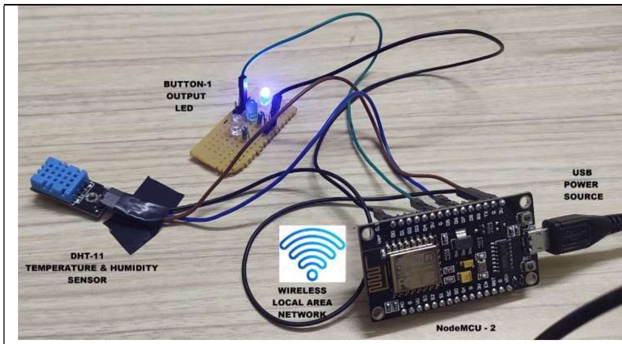

Figure 2. NodeMCU-2 Section

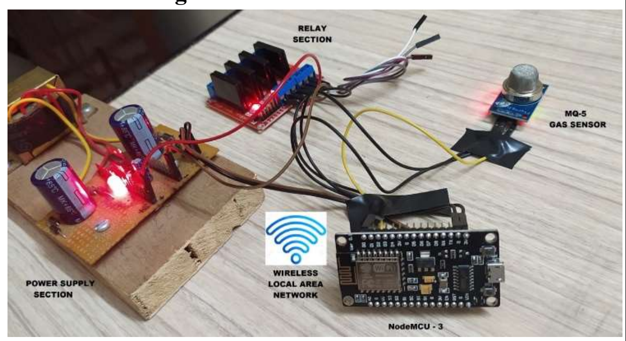

Figure 3. NodeMCU-3 Section

The NodeMCU along with the OLED and push buttons used 3.3V dc power supply for its operation. As OLED was an I2C device, only two pins SDA and SCL were required to interface it with the NodeMCU. Rest two pins were the power pins. Two push buttons were interfaced to the two digital I/O pins of the NodeMCU via two pull-up resistors of 10K in between. This was required to prevent any false triggering over these pins during high-impedance state. Node-2 architecture as shown below in Figure-2 was also implemented around a NodeMCU module. Here a commonly used temperature and humidity sensor module DHT-11 was interfaced to the NodeMCU. As DHT-11 module used here was a digital output type so instead of using an analog input channel it utilized a simple digital I/O channel. The sensor was made to operate on 3.3V dc and acted as an input for pin D7 of the NodeMCU. On the other side a SPDT relay was also interfaced to one of the digital I/O pins of the NodeMCU. The SPDT relay acted as an output device and was connected on pin D0 of the NodeMCU. The relay acted as an electromagnetic switch and was used here to switch 220V AC load connected, bulb in this case. Here as the relay power consumption was more so the system was fed with a higher power source 5V/ 1A. Node-3 architecture as shown below in Fig.-4 was implemented around another NodeMCU module. Here a commonly used gas sensor module MQ-135 was interfaced to the NodeMCU. As MQ-135 module used here was an analog output type so instead of using a digital input channel it utilized the only analog input channel A0 of NodeMCU. This channel fed internal ADC with the analog values and converted those values to their digital equivalents for further processing. The sensor was made to operate on 3.3V dc and acted as an input device here. On the other side another SPDT relay was interfaced to one of the digital I/O pins of the NodeMCU. The SPDT relay acted as an output device and was connected on pin D0 of the NodeMCU. The relay acted as an electromagnetic switch and was used here to switch 220V AC load connected, bulb in this case. Here as the relay power consumption was more so the system was fed with a higher power source 5V/ 1A.

FIRMWARE

First of all at the initial stage we downloaded and installed all the required libraries and also included those in the source code. These libraries were:

1. painlessMesh.h
2. ArduinoJson.h
3. SPI.h
4. Wire.h
5. Adafruit_GFX.h
6. Adafruit_SSD1306.h

Then we defined the Wi-Fi Credentials including the Mesh_SSID, Mesh_Password and Mesh_Port. These Wi-Fi credentials remained same for each and every board which we wanted to communicate between each other.

Sending Part of the Code:

First of all we read the button status and if the button status was low then we just toggled the button status variable. So, whenever the button was pressed the button status variable would change from one to zero or from zero to one according to the last variable saved in that variable. After saving the button status then was the time to send that and for sending that we used the JSON Format. JSON (Java Script Object Notation) is just an object notation or we can say a syntax in which a data is represented. It is very popular in sending and receiving data may be over the internet or may be between the hardware.

JSON Code Part:

First of all a JSON object was created in which there would be two values Relay1 and Relay2 and their keys were button1_status and button2_status respectively. So this was how JSON syntax looked like. Likewise we could create many values and assign them respective keys and this data was converted into string format by using the function in the code called as

serializeJson(doc,msg);

So then that JSON format data was saved in string called *msg.*

“Relay1”:true,
“Relay2”:false,
“Relay3”:true,
“Relay4”:true,
“Relay5”:false

Where “Relayx” were known as ‘Values’ and true/false were called ‘Keys’. Data transmitted in JSON Format appeared in the serial monitor window like this:

{“Relay1”:false, “Relay2”:false}
{“Relay1”:true, “Relay2”:false}

Then on the receiver side it was very easy to deserialize that string and just extract that particular data which we wanted out of that whole string.

Code Part at the Receiver Side:

On the receiver side if we wanted the data of Relay1 particularly so first of all we deserialized it

DeserializationError error = deserializeJson(doc, json);

and then

write relay1_status =doc[“Relay1”];

which was the name of our value which data we wanted and we got the key of this value easily

digitalWrite(Relay1, relay1_status);

and we could easily turn on and off the relay by using this variable. So this was the simplest and practical explanation of how we transmitted and received data using JSON. Here we just serialized the push button data sending it after every second.

Then that data was broadcasted to all the devices under same network under same port number. Then, talking about the receiving part of the code, after receiving the data we first deserialized the JSON and saved different data into different string.

```
String Temp = doc["TEMP"];
String Hum = doc["HUM"];
String Gas = doc["GAS"];
```

The proposed system has been implemented around the NodeMCU IoT enabled development board with on chip Wi-Fi chipset by Espressif. The hardware set-up along with the web-server dashboard will collectively fulfil the purpose. This is an experimental work and the developed prototype has validated the research work reported here in this dissertation work. Solution has been provided for each problem statement that has been discussed and the research has been carried out in a phased manner.

CONCLUSION

Here it could be easily observed and verified that to establish communication between multiple nodes in a wireless network could be through a router if it were a traditional Wi-Fi network which has its own set of limitations or it could be attained without using any router in between via a mesh network topology which has its own set of benefits. Here it was successfully demonstrated that a local wireless local area network could be established between multiple nodes without even using the internet. The number of nodes could increase to a significant level using this mesh network topology as here the nodes are mutually responsible for relaying each other's transmissions and these interconnected nodes resulted in a much larger coverage area.

REFERENCES

[1] Dilip Kumar Sharma; Neeraj Baghel; Siddhant Agarwal, "Multiple Degree Authentication in Sensible Homes based on IoT Device Vulnerability", 2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC), IEEE.

[2] Satyendra K. Vishwakarma; Prashant Upadhyaya; Babita Kumari; Arun Kumar Mishra, "Smart Energy Efficient Home Automation System Using IoT", 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), IEEE.

[3] Kabita Agarwal; Arun Agarwal; Gourav Misra, "Review and Performance Analysis on Wireless Smart Home and Home Automation using IoT", 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics, and Cloud) (I-SMAC), IEEE.

[4] Tushar Chaurasia; Prashant Kumar Jain, "Enhanced Smart Home Automation System based on Internet of Things", 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics, and Cloud) (I-SMAC), IEEE.

[5] Tui-Yi Yang; Chu-Sing Yang; Tien-Wen Sung, "A Dynamic Distributed Energy Management Algorithm of Home Sensor Network for Home Automation System", 2016 Third International Conference on Computing Measurement Control and Sensor Network (CMCSN), IEEE.

[6] Whether heated A. Jabbar; Mohammed Hayyan Alsibai; Nur Syaira S. Amran; Samiah K. Mahanadi, "Design and Implementation of IoT-Based Automation System for Smart Home", 2018 International Symposium on Networks, Computers, and Communications (ISNCC), IEEE.

[7] Shradha Somani; Parikshit Solunke; Shaunak Oke; Parth Medhi; P.P. Laturkar, "IoT Based Smart Security and Home Automation", 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), IEEE.

[8] Puneet Kumar Aggarwal; P. S. Grover; Laxmi Ahuja, "A Performance Evaluation Model for Mobile Applications", 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), IEEE.

[9] John Fox; Andrew Donnellan; Liam Doumen, "The deployment of an IoT network infrastructure, as a localized regional service", 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE.

[10] Alok Kumar Gupta; Rahul Johari, “IOT based Electrical Device Surveillance and Control System”, 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), IEEE.

[11] Fan Wu; Taiyang Wu; Mehmet Rasit Yuce, “Design and Implementation of a Wearable Sensor Network System for IoT-Connected Safety and Health Applications”, 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE.

[12] Arvind Arya; Akash Taliyan; Pradeep Chauhan; Anju Gautam, “Smart Kitchen with New Measurement, Web and Application Based with Affordable Design”, 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), IEEE.

[13] Sachin A. Goswami; Bhargav P. Padhya; Ketan D. Patel, “Internet of Things: Applications, Challenges, and Research Issues”, 2019 (IoT in Social, Mobile, Analytics, and Cloud) (I-SMAC), IEEE.