Experiment 03
Smart Light Bulb

In this experiment, we will learn how to control a light bulb remotely. We will use the Wi-Fi-enabled microcontroller to connect the light bulb to a local network through which we can send control commands. The control commands should enable the user to remotely switch the light ON and OFF.

Background

The Internet of Things (IoT) enables multiple devices to connect and transfer data within a network. As a consequence of the design, it is required for a device to have the capability of connecting to an available network. Household devices such as light bulbs might not necessarily have the hardware requirements to connect and share data through a network. "Edge" devices can help negate this problem by forming a bridge between household devices and the network. The Node MCU in Fig1 acts as an Edge device connecting a lightbulb to the Local Area Network (LAN). With the LAN connected to the internet, the Edge device can now be accessed from outside the LAN.

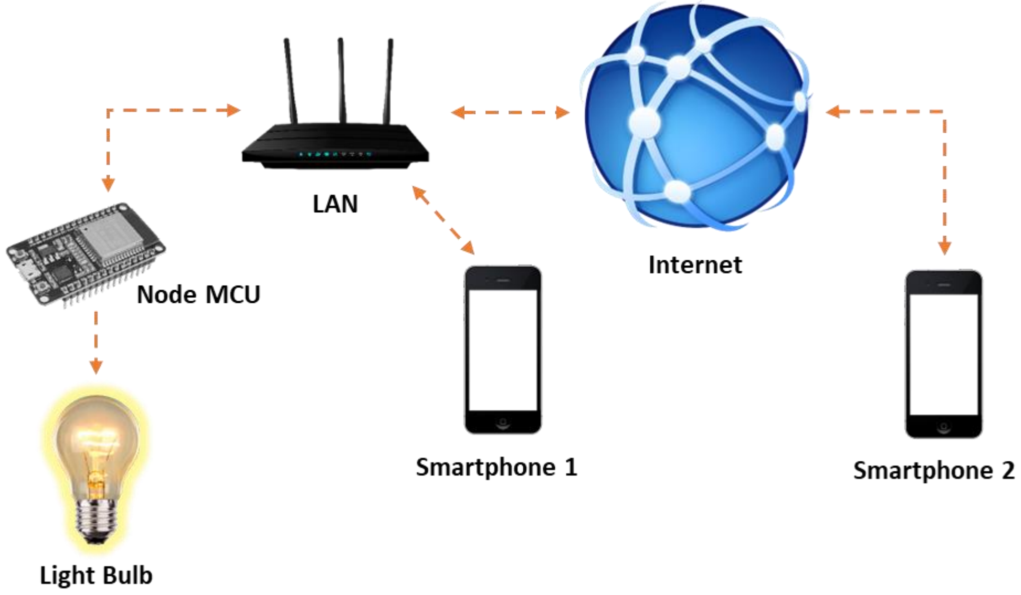

Fig 1.1. Network map of connected IoT devices.

The Node MCU can be programmed to process instructions such as switching ON/OFF the light. Connected devices in the network can now send preprogrammed instructions to control the light bulb. For example, in Fig 1.1, consider the LAN to be disconnected from the Internet. Now Smartphone 1 can send a command to turn the light bulb ON while Smartphone 2 cannot since the Edge device is no longer connected to the internet.

💻Experiment Set-up: Configuration

In this experiment, we will be using ESPAsyncWebServer and ESPAsyncTCP libraries to create a web server that runs on ESP 8266. The ESPAsyncTCP library allows the microcontroller to send and receive TCP data packets. The WebServer library is designed on top of the TCP library and allows the Node MCU to host a web server. This web server functionality enables the Node MCU to display content on the website that can be accessed through the network.

When the Node MCU connects to the router, it is assigned an IP address (Dynamic IP allocation). This IP address is the web address of the website hosted on the microcontroller. Any user connected to the network can enter this IP address in their web browsers to access the website. The website is the interface through which a user can interact with the Node MCU and control a connected device remotely.

In this module, we will create a simple HTML website that displays a button. When a user clicks on the button, the microcontroller will switch the light bulb ON/OFF. This website will be hosted on the microcontroller. The Node MCU connects to a wireless network via an access point. Students can create an access point by enabling either tethering or hotspots on their mobile devices and then configuring the Node MCU to connect with the created access point. Most Android-based mobile phones and Windows/Linux-based PCs will allow users to create access points without an internet connection. For the scope of this experiment, internet connectivity is not required.

Instructions

1. Please refer to the IoT Board Manual for installing the Arduino software and setting up Node MCU. Generally, the libraries can be installed from the Library Manager available within the Arduino IDE. But the ESPAsyncWebServer and ESPAsyncTCP libraries cannot be installed through the Library Manager.
2. The links to the libraries is on Canvas. Download the Zip files of the libraries.
3. Open Arduino and click on "Sketch" from the Menu bar. From the dropdown options, click on "Include Library" and then click on "Add .zip library." Now select the downloaded library files. This will copy the libraries into the Arduino Installation Libraries folder.
4. Click on "File" from the Menu bar and click on "Open" to open the "smartbulb.ino" file. The "smartbulb.ino" file can be found in the course material for this module. Now the IDE displays the code that needs to be flashed onto the microcontroller.
5. Find the Network Credentials section in the code. In this section, you have to enter the name and password of the Wi-Fi network you want the Node MCU connected to. If you do not have access to a Wi-Fi network, please refer to the IoT Board Manual for creating a mobile hotspot with a capable device (Laptop/Smart Phone).
6. Now upload the code to the microcontroller. Refer to the IoT Board Manual for flashing instructions.
7. Open the Serial Monitor from the dropdown menu under "Tools" on the Menu bar. Refer to the IoT Board Manual for setting up the Serial Monitor window.
8. In the Serial Monitor Window, the Node MCU will now display the IP address assigned to it. Note down this IP address as this needs to be entered into a browser to access the website running on the microcontroller.
9. For example, consider the IP address looks like "192.168.1.28." Now open a web browser on any capable device connected to the same Wi-Fi network. In the browser search bar, type "http://192.168.1.28/" and press enter. The web page with the LED control button should now be visible.
10. Click on the button and verify if the LED is switched ON/OFF.

☞ Deliverables

Demonstration:

1. Record a video demonstration explaining the outcome of the experiment. Refer to the title page for a brief description of the expected outcome. Make sure you talk over all observations and the video is presentable.
2. Address the following items in your recording or add it as text in the submission:
   a. What is latency? What is the latency experienced when the LED button is pressed on the website? What would a possible bottleneck be in improving the overall latency?
   b. Does the IP address shown in the Serial Monitor change if you disconnect the Node MCU from the router? Also, provide a reason as to why the IP address changes or remains the same.
   c. Let's consider that the network the Node MCU is connected to, has a connection to the Internet while performing the experiment. Would the same setup work when trying to remotely connect with the light bulb via Smart Phone 2, as shown in Fig 1.1? Please provide a brief explanation.
   d. Connect and access the website from multiple devices and try to click on the button simultaneously on all the devices. What outcome do you observe?

🗎References and Further Reading

[1] https://github.com/me-no-dev/ESPAsyncWebServer

[2] https://github.com/me-no-dev/ESPAsyncTCP

[3] https://tttapa.github.io/ESP8266/Chap07%20-%20Wi-Fi%20Connections.html

[4] https://tttapa.github.io/ESP8266/Chap05%20-%20Network%20Protocols.html

[5] https://www.youtube.com/watch?v=O2W8uyC-Gio (Explains some of the limitations of the ESPAsync libraries)