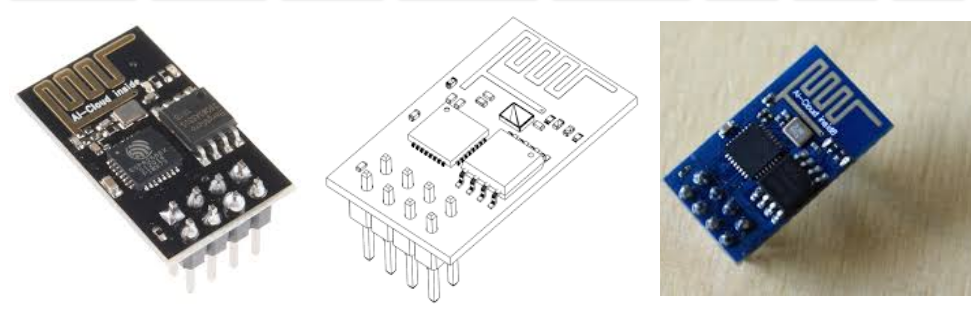
**iot based home automation**

**introduction**

In this project, we will be using the ESP8266 – 01 Wi-Fi Development board to make a small IoT home automation that has Wi-Fi featured in it. The system operates on a local web server and is easy to use for the novice. With this project, we can control at most two AC appliances which suit best for your small IoT projects.



**Component**

* ESP8266 – 01: It will serve as the brain of our project.
* FTDI Board (USB to TTL): To upload code in ESP8266 – 01
* 2 AC Appliances
* One(1) 2-Channel Relay Module
* Breadboard
* Connecting wires

Application

1. Traffic monitoring.

### **2. Fleet management.**

### **3. Agriculture.**

### **4. Smart grid and energy saving.**

### **5.Hospitality.**

Objective

During this activity ,you will help students to achieve following objectives

1. Understanding the principle and operation of ESp8266-01 module
2. Design algorithm and flowchart to home automation system
3. Programming ESp8266-01 module
4. Interfacing ESp8266-01 module

program

delay(10);

pinMode(0, OUTPUT); // set GPIO 0 as OUTPUT

pinMode(2, OUTPUT); // set GPIO 2 as OUTPUT

digitalWrite(0, 0); // initially set GPIO 0 as Low

digitalWrite(2, 0); // initially set GPIO 2 as Low

Serial.println();

Serial.println();

Serial.print("Connecting to "); // Connect to WiFi network

Serial.println(ssid);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

server.begin(); // Start the server

Serial.println("Server started");

Serial.println(WiFi.localIP()); // Print the IP address

}

void loop()

{

WiFiClient client = server.available(); // Check if a client has connected

if (!client)

{

return; // Wait until the client sends some data

}

Serial.println("new client");

while(!client.available())

{

delay(1);

}

String req = client.readStringUntil('\r'); // Read the first line of the request

Serial.println(req);

client.flush();

int val1;

int val2;

if (req.indexOf("/gpio1/0") != -1) // Match the request

val1 = 1;

else if (req.indexOf("/gpio1/1") != -1)

val1 = 0;

else if (req.indexOf("/gpio2/0") != -1)

val2 = 1;

else if (req.indexOf("/gpio2/1") != -1)

val2 = 0;

else

{

Serial.println("invalid request");

client.stop();

return;

}

digitalWrite(2, val1); // Set GPIO 2 according to the request

digitalWrite(0, val2); // Set GPIO 0 according to the request

client.flush();

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>"; // Prepare the response

s += "<body><h1>ESP8266 Smallest Home Automation</h1>\r\nGPIO1 is now ";

s += (val1)?"high":"low";

s += "\n GPIO2 is now ";

s += (val2)?"high":"low";

s += "</html>\n";

client.print(s); // Send the response to the client

delay(1);

Serial.println("Client disonnected"); // The client will actually be disconnected and when the function returns and 'client' object is detroyed

}

#include <ESP8266WiFi.h> // This library helps to provide the functions of the ESP8266

const char\* ssid = "xxxx"; // Declare your Wi-Fi name here within the double quotes

const char\* password = "xxxx"; // Declare your Wi-Fi password here within the double quotes

WiFiServer server(80); // Create an instance of the server and specify the port to listen on as an argument

void setup()

{

Serial.begin(9600); // set buad rate

delay(10);

pinMode(0, OUTPUT); // set GPIO 0 as OUTPUT

pinMode(2, OUTPUT); // set GPIO 2 as OUTPUT

digitalWrite(0, 0); // initially set GPIO 0 as Low

digitalWrite(2, 0); // initially set GPIO 2 as Low

Serial.println();

Serial.println();

Serial.print("Connecting to "); // Connect to WiFi network

Serial.println(ssid);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

server.begin(); // Start the server

Serial.println("Server started");

Serial.println(WiFi.localIP()); // Print the IP address

}

void loop()

{

WiFiClient client = server.available(); // Check if a client has connected

if (!client)

{

return; // Wait until the client sends some data

}

Serial.println("new client");

while(!client.available())

{

delay(1);

}

String req = client.readStringUntil('\r'); // Read the first line of the request

Serial.println(req);

client.flush();

int val1;

int val2;

if (req.indexOf("/gpio1/0") != -1) // Match the request

val1 = 1;

else if (req.indexOf("/gpio1/1") != -1)

val1 = 0;

else if (req.indexOf("/gpio2/0") != -1)

val2 = 1;

else if (req.indexOf("/gpio2/1") != -1)

val2 = 0;

else

{

Serial.println("invalid request");

client.stop();

return;

}

digitalWrite(2, val1); // Set GPIO 2 according to the request

digitalWrite(0, val2); // Set GPIO 0 according to the request

client.flush();

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>"; // Prepare the response

s += "<body><h1>ESP8266 Smallest Home Automation</h1>\r\nGPIO1 is now ";

s += (val1)?"high":"low";

s += "\n GPIO2 is now ";

s += (val2)?"high":"low";

s += "</html>\n";

client.print(s); // Send the response to the client

delay(1);

Serial.println("Client disonnected"); // The client will actually be disconnected and when the function returns and 'client' object is detroyed

}

#include <ESP8266WiFi.h> // This library helps to provide the functions of the ESP8266

const char\* ssid = "xxxx"; // Declare your Wi-Fi name here within the double quotes

const char\* password = "xxxx"; // Declare your Wi-Fi password here within the double quotes

WiFiServer server(80); // Create an instance of the server and specify the port to listen on as an argument

void setup()

{

Serial.begin(9600); // set buad rate

delay(10);

pinMode(0, OUTPUT); // set GPIO 0 as OUTPUT

pinMode(2, OUTPUT); // set GPIO 2 as OUTPUT

digitalWrite(0, 0); // initially set GPIO 0 as Low

digitalWrite(2, 0); // initially set GPIO 2 as Low

Serial.println();

Serial.println();

Serial.print("Connecting to "); // Connect to WiFi network

Serial.println(ssid);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

server.begin(); // Start the server

Serial.println("Server started");

Serial.println(WiFi.localIP()); // Print the IP address

}

void loop()

{

WiFiClient client = server.available(); // Check if a client has connected

if (!client)

{

return; // Wait until the client sends some data

}

Serial.println("new client");

while(!client.available())

{

delay(1);

}

String req = client.readStringUntil('\r'); // Read the first line of the request

Serial.println(req);

client.flush();

int val1;

int val2;

if (req.indexOf("/gpio1/0") != -1) // Match the request

val1 = 1;

else if (req.indexOf("/gpio1/1") != -1)

val1 = 0;

else if (req.indexOf("/gpio2/0") != -1)

val2 = 1;

else if (req.indexOf("/gpio2/1") != -1)

val2 = 0;

else

{

Serial.println("invalid request");

client.stop();

return;

}

digitalWrite(2, val1); // Set GPIO 2 according to the request

digitalWrite(0, val2); // Set GPIO 0 according to the request

client.flush();

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>"; // Prepare the response

s += "<body><h1>ESP8266 Smallest Home Automation</h1>\r\nGPIO1 is now ";

s += (val1)?"high":"low";

s += "\n GPIO2 is now ";

s += (val2)?"high":"low";

s += "</html>\n";

client.print(s); // Send the response to the client

delay(1);

Serial.println("Client disonnected"); // The client will actually be disconnected and when the function returns and 'client' object is detroyed

}

#include <ESP8266WiFi.h> // This library helps to provide the functions of the ESP8266

const char\* ssid = "xxxx"; // Declare your Wi-Fi name here within the double quotes

const char\* password = "xxxx"; // Declare your Wi-Fi password here within the double quotes

WiFiServer server(80); // Create an instance of the server and specify the port to listen on as an argument

void setup()

{

Serial.begin(9600); // set buad rate

delay(10);

pinMode(0, OUTPUT); // set GPIO 0 as OUTPUT

pinMode(2, OUTPUT); // set GPIO 2 as OUTPUT

digitalWrite(0, 0); // initially set GPIO 0 as Low

digitalWrite(2, 0); // initially set GPIO 2 as Low

Serial.println();

Serial.println();

Serial.print("Connecting to "); // Connect to WiFi network

Serial.println(ssid);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

server.begin(); // Start the server

Serial.println("Server started");

Serial.println(WiFi.localIP()); // Print the IP address

}

void loop()

{

WiFiClient client = server.available(); // Check if a client has connected

if (!client)

{

return; // Wait until the client sends some data

}

Serial.println("new client");

while(!client.available())

{

delay(1);

}

String req = client.readStringUntil('\r'); // Read the first line of the request

Serial.println(req);

client.flush();

int val1;

int val2;

if (req.indexOf("/gpio1/0") != -1) // Match the request

val1 = 1;

else if (req.indexOf("/gpio1/1") != -1)

val1 = 0;

else if (req.indexOf("/gpio2/0") != -1)

val2 = 1;

else if (req.indexOf("/gpio2/1") != -1)

val2 = 0;

else

{

Serial.println("invalid request");

client.stop();

return;

}

digitalWrite(2, val1); // Set GPIO 2 according to the request

digitalWrite(0, val2); // Set GPIO 0 according to the request

client.flush();

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>"; // Prepare the response

s += "<body><h1>ESP8266 Smallest Home Automation</h1>\r\nGPIO1 is now ";

s += (val1)?"high":"low";

s += "\n GPIO2 is now ";

s += (val2)?"high":"low";

s += "</html>\n";

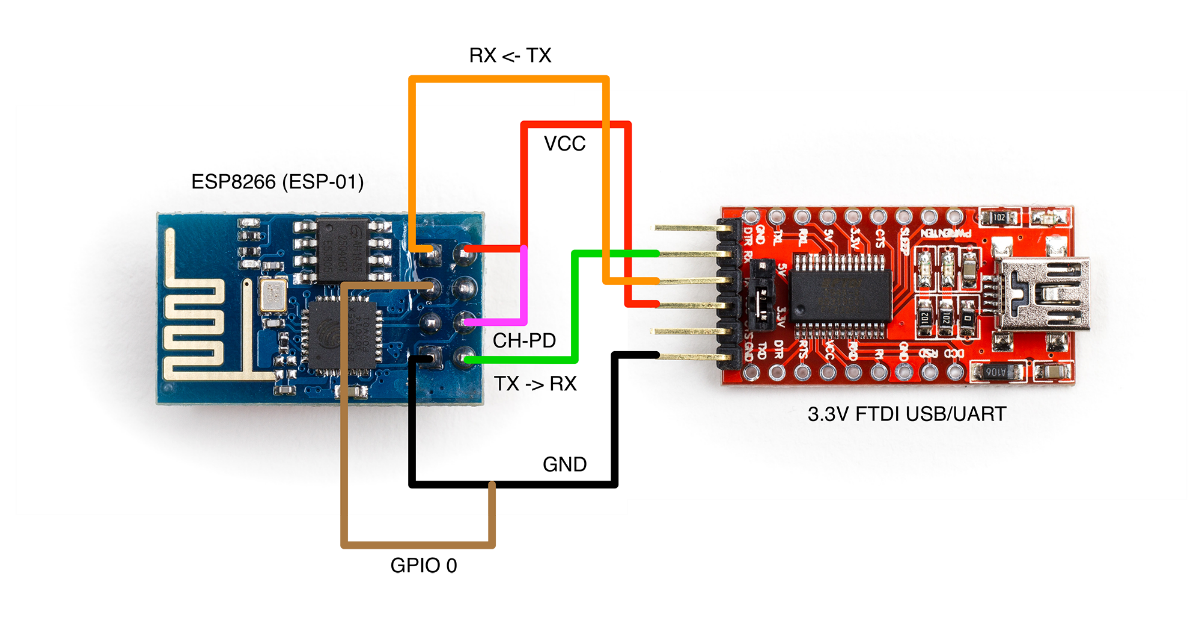
client.print(s); // Send the response to the client

delay(1);

Serial.println("Client disonnected"); // The client will actually be disconnected and when the function returns and 'client' object is detroyed

}

Hardware

1. Connect 2 Ac bulb at relay module output and connect VCC and GND supply pin of relay module to %v input jack to powered the circuit
2. Connect ESP8266-01 module to relay such that GPIO 0 and GPIO 2 PIN to IN1 and IN2 input pin of relay
3. Connect VCC and GND pin to 3.3 volt jack input to powered module
4. Connect the FTDI pins to ESP8266 as shown below.Fig. 12: Flashing ESP8266 – 01
5. Select your COM Port from Tools > Port after connecting FTDI USB to your PC or Laptop.
6. Also, check if your ESP8266 supports 9600 or any other Baud Rate.
7. Now Click Upload.

