TENET TECHNETRONICS

2014

Controlling LED's using Mobile Hotspot through Wi-Fi module (ESP8266) with Arduino.

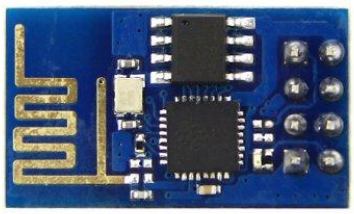
"Simplifying

Author: Mr. Prajwal R

Introduction

ESP8266 is one of the low cost Wi-Fi modules as a highly integrated chip designed for the needs of a new connected world. It offers a complete and self-contained Wi-Fi networking solution, which can carry software applications, or through another application processor uninstall all Wi-Fi networking capabilities.

ESP8266 has powerful on-board processing and storage capabilities that allow it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.



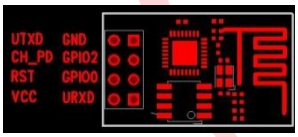
Here is a quick demo on how to control LED's using mobile hotspot through Wi-Fi module ESP8266 with Arduino.

HardwareRequired:

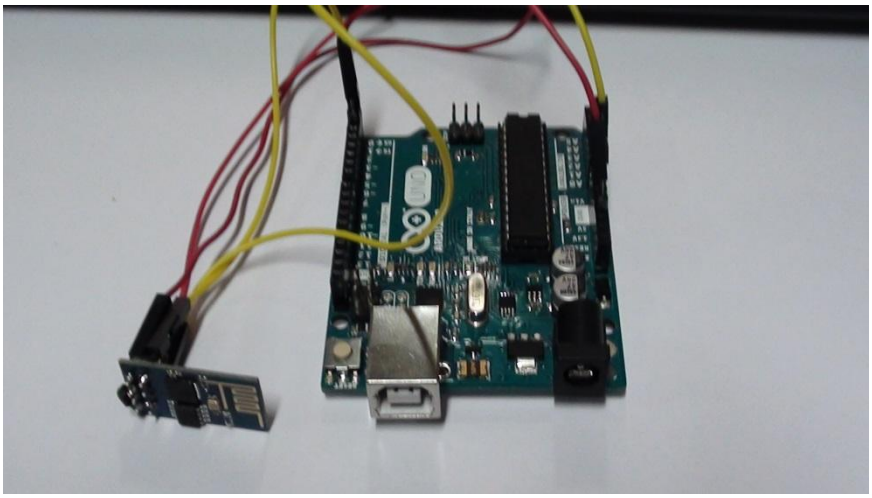
- Arduino board.
- Arduino cable.
- Wi-Fi module ESP8266.
- LED's.
- Smart Phone.
- USB to TTL serial cable.

Step one: Connections

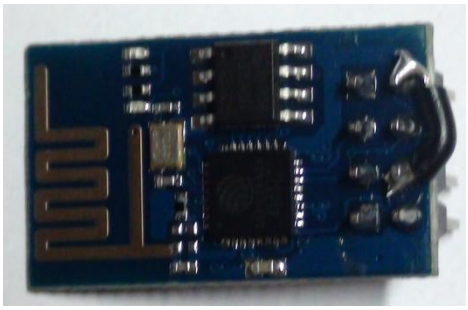
a) Connection of Arduino with Wi-Fi module ESP8266 is as follows



Arduino board	Wi-Fi module ESP8266
GND	GND
3.3v	VCC
Rx	UTXD
Tx	URXD
3.3v	CH_PD



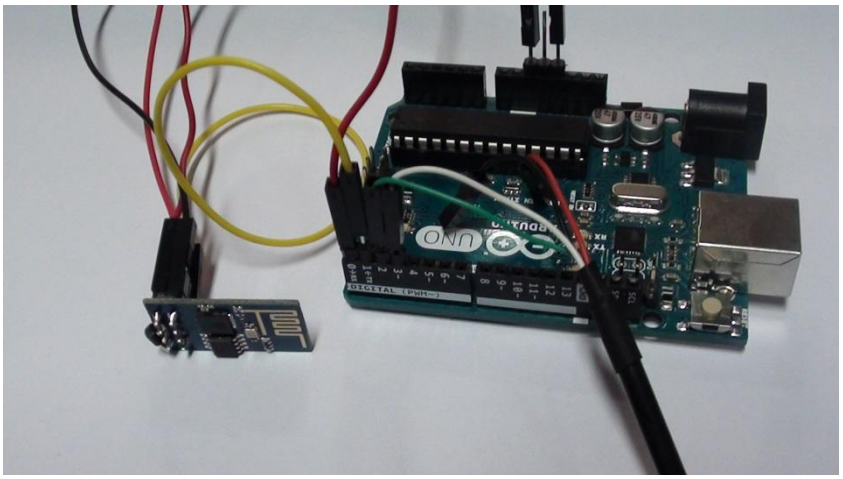
Note:CH_PD and Vcc pins of Wi-Fi module need to be given to 3.3V. Since Boot mode must be 1 to enable Wi-Fi.



b) Connection of Arduino with USB to TTL serial cable is as follows.

Arduino board	USB to TTL Serial cable
D2	TX(Green)
D3	Rx(White)
GND	GND

Note : We used a software serial to print some debugging information as there isone hardwareserial on Arduino board.



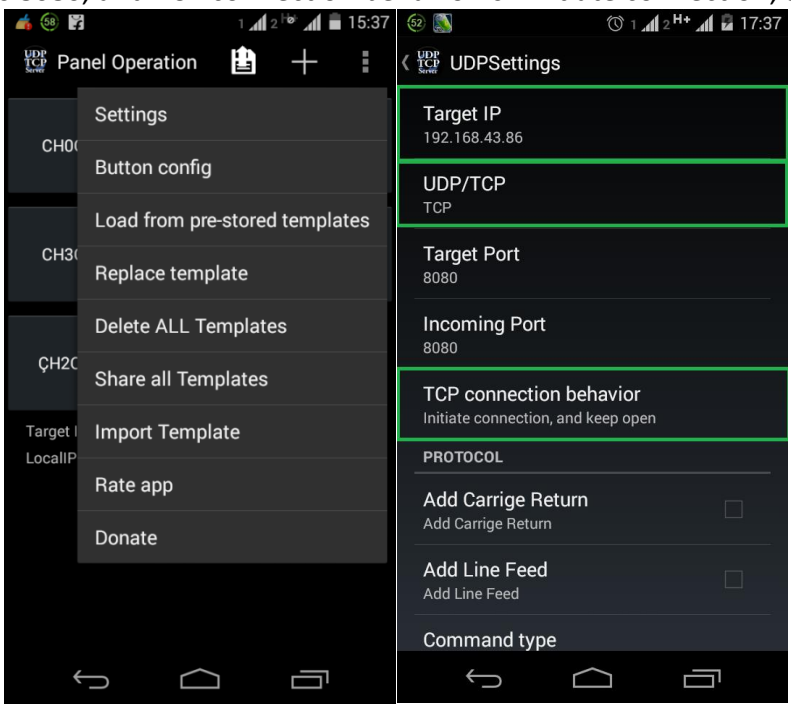
c) Connection of Arduino with LED's is as follows.

Arduino board	LED's
D4	LED1
D5	LED2
D6	LED3
D7	LED4

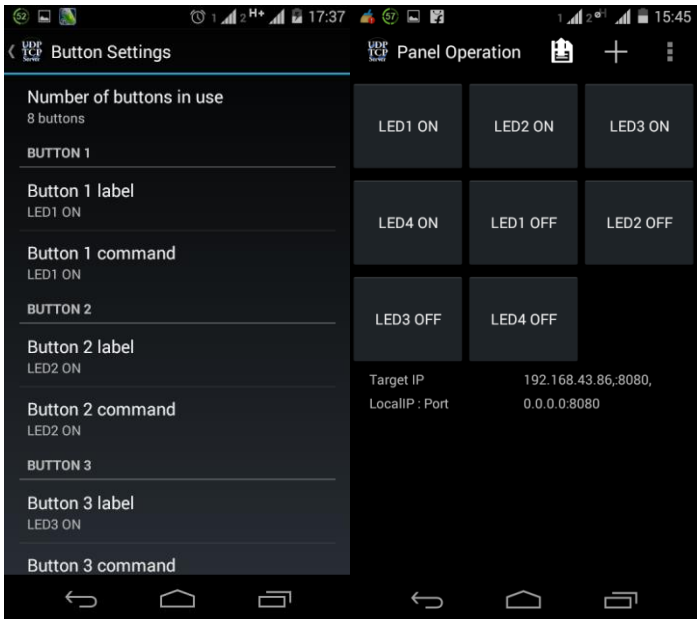
Step two:

Install “udp/tcp server” app to the android phone and make following settings.

Open this app after installation. Click Settings in the up right corner. Set the Target IP as the Wi-Fi IP that has shown on the serial monitor. We are 172.16.1.12 in this blog. UPD/TCP option is TCP, Target Port is 8080, and TCP connection behavior is “Initiate connection, and keep open”.



Then do the Button Settings. Set 8 buttons in this app, they are “LED1 ON”, “LED2 ON”, “LED3 ON”, “LED4 ON”, “LED1 OFF”, “LED2 OFF”, “LED3 OFF”, “LED4 OFF”, which represent the on and off status for each LED’s.



Step Three:

Connect the Arduino board to the computer using Arduino USB cable. Launch the Arduino IDE and select the appropriate serial port and board.

Program:

/*

ESP8266 library

When you use with UNO board, uncomment the follow line in uartWIFI.h.

#define UNO

When you use with MEGA board, uncomment the follow line in uartWIFI.h.

#define MEGA

Connection:

When you use it with UNO board, the connection should be like these:

ESP8266_TX->D0

ESP8266_RX->D1

ESP8266_CH_PD->3.3V

ESP8266_VCC->3.3V

ESP8266_GND->GND

FTDI_RX->D3 //The baud rate of software serial can't be higher that 19200, so we use software serial as a debug port

FTDI_TX->D2

When you want to output the debug information, please use DebugSerial. For example,

DebugSerial.println("hello");

Note: The size of message from ESP8266 is too big for arduino sometimes, so the library can't receive the whole buffer because

the size of the hardware serial buffer which is defined in HardwareSerial.h is too small.

Open the file from \arduino\hardware\arduino\avr\cores\arduino\HardwareSerial.h.

See the follow line in the HardwareSerial.h file.

#define SERIAL_BUFFER_SIZE 64

The default size of the buffer is 64. Change it into a bigger number, like 256 or more.

*/

```
#include <SoftwareSerial.h>
#define SSID      "TENET"                  //type your own SSID name
#define PASSWORD   "1234567890"                              //type your own WIFI password
```

```
#include "uartWIFI.h"
#include <SoftwareSerial.h>
WIFI wifi;

externintchlID;          //client id(0-4)
```

```
void setup()
```

```
{
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(7,OUTPUT);
wifi.begin();
bool b = wifi.Initialize(STA, SSID, PASSWORD);
if(!b)
 {
DebugSerial.println("Init error");
 }
delay(8000);  //make sure the module can have enough time to get an IP address
 String ipstring  =wifi.showIP();
DebugSerial.println(ipstring);            //show the ip address of module

delay(2000);
wifi.confMux(1);
delay(100);
if(wifi.confServer(1,8080))
        DebugSerial.println("Server is set up");
}
```

```
void loop()

{

charbuf[100];

intiLen = wifi.ReceiveMessage(buf);

if(iLen> 0)

 {


if (strcmp(buf, "LED1 ON") == 0)

   {

digitalWrite(4,HIGH);

        DebugSerial.println("LED1 ON");

   }

if (strcmp(buf, "LED1 OFF") == 0)

   {

digitalWrite(4,LOW);

        DebugSerial.println("LED1 OFF");

   }

if (strcmp(buf, "LED2 ON") == 0)

   {

digitalWrite(5,HIGH);

        DebugSerial.println("LED2 ON");
```

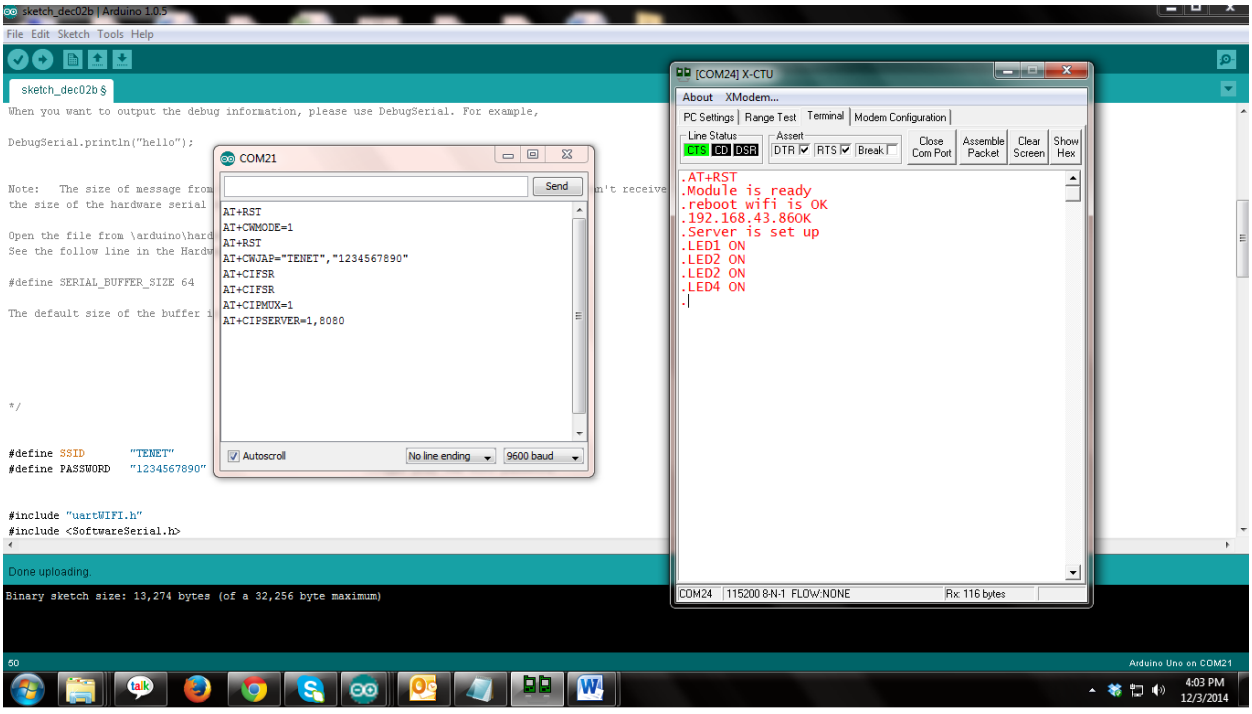
```
  }
if (strcmp(buf, "LED2 OFF") == 0)
  {
digitalWrite(5,LOW);
        DebugSerial.println("LED2 OFF");
  }
if (strcmp(buf, "LED3 ON") == 0)
  {
digitalWrite(6,HIGH);
        DebugSerial.println("LED2 ON");
  }
if (strcmp(buf, "LED3 OFF") == 0)
  {
digitalWrite(6,LOW);
        DebugSerial.println("LED3 OFF");
  }
if (strcmp(buf, "LED4 ON") == 0)
  {
digitalWrite(7,HIGH);
        DebugSerial.println("LED4 ON");
  }
```

```
if (strcmp(buf, "LED4 OFF") == 0)
   {
digitalWrite(7,LOW);
        DebugSerial.println("LED4 OFF");
   }
 }
}
```

Note: before uploading this program make sure uartWifi library has been installed.

Step three:

Finally to see the output, click the serial monitor icon in the Arduino IDE(newer module works only in 9600 baud rate, so you need change the baud rate in uartWifi library). And also to see the response from the module, open any serial terminal software (we have used x-ctu) with appropriate serial port which is connected to D2 and D3 of your Arduino.



TENET
TECHNETRONICS

"Simplifying Technology For Life"