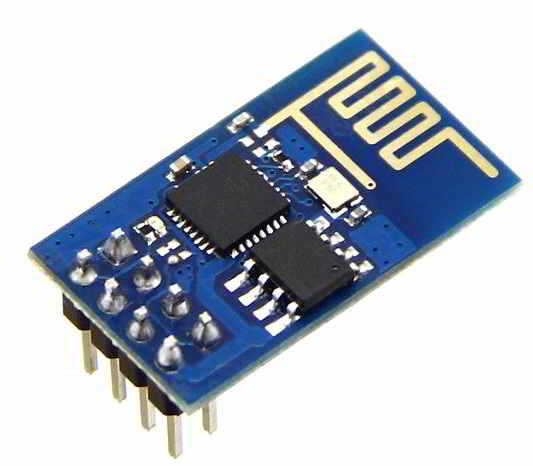
ESP8266 WiFi Module Interfacing with Arduino UNO

Introduction

ESP8266 wifi module is low cost standalone wireless transceiver that can be used for end-point IoT developments.

ESP8266 wifi module enables internet connectivity to embedded applications. It uses TCP/UDP communication protocol to connect with server/client.



To communicate with the ESP8266 wifi module, microcontroller needs to use set of AT commands. Microcontroller communicates with ESP8266-01 wifi module using UART having specified Baud rate (Default 115200).

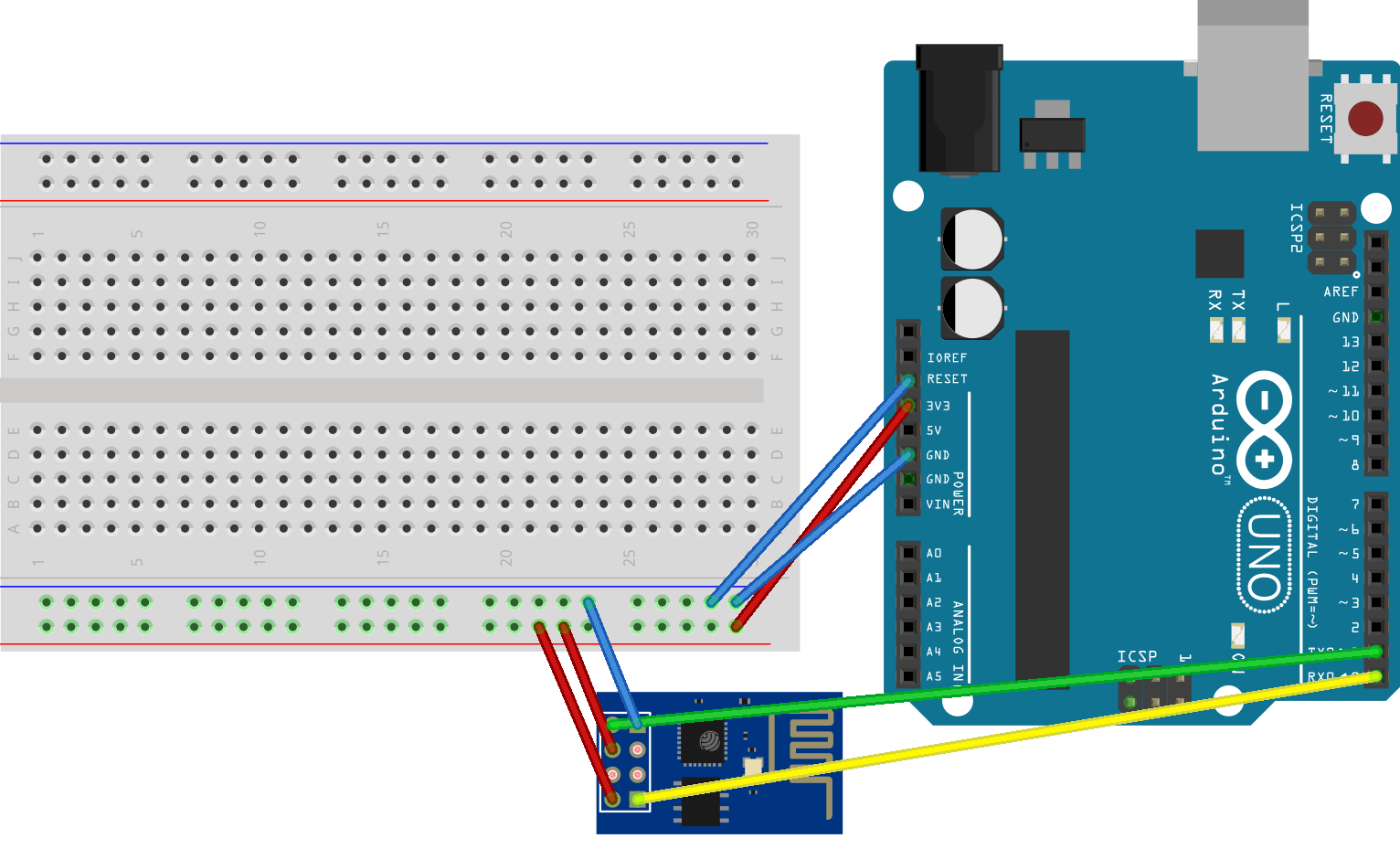
Now let’s interface ESP8266 wifi Module with Arduino UNO.

Connecting the ESP8266 to an Arduino:-

The steps you need to take are simple. This is written for the ESP8266-01 but you can find the pinout for other models easily and use the same pins. First we will connect the Arduino UNO to a breadboard:

1. Connect the Arduino’s **3v3** (3.3V) output to the red line on a breadboard. The ESP8266 works with 3.3V and not 5V, so this is necessary. If you want to connect other components that use 5V, you can connect the 5V output to the other red line of the breadboard, just make sure you don’t connect the two.
2. Connect **GND** (ground) to the blue line.
3. Connect the **RES** or **RESET** pin to the blue line. When you ground the reset pin, the Arduino works as a dumb USB to serial connector, which is what we want to talk to the ESP8266.
4. Connect the **RXD** pin of the Arduino to the **RX** pin of the ESP8266 (yellow color in the picture).
5. Connect the **TXD** pin of the Arduino to the **TX** pin of the ESP (green color in the picture). Usually, when we want two things to talk to each other over serial, we connect the **TX** pin of one to the **RX** of the other (send goes to receive and the opposite). Here we do not have the Arduino talk to the ESP8266 though, our computer is talking to it *via* the Arduino.
6. Connect the **GND** pin of the ESP to the blue line and the **VCC**pin to the red line.
7. Finally **CH\_PD** goes to the red line, supposedly it will not work if you do not connect this. According to a [discussion](http://www.eevblog.com/forum/microcontrollers/esp8266-ch_pd-useful-for-anything/msg681407/?PHPSESSID=66231e7f2b5b7525e3c455cd5eef1a36#msg681407) I found:

ESP-01 and ESP-03 were initially meant to be used as an Arduino WiFi module. In that aspect it made sense to break out CH\_PD so that the user could disable the device when not in use (to save power)



Using the Arduino IDE:-

In the Arduino IDE, you don’t need to choose a board, as we’re not uploading anything to the ESP8266. Just choose the right port in the **Tools** menu and go to **Tools** → **Serial Monitor**. Then simply set your baud rate to **115200** (the default ESP8266 firmware uses it) and your line endings to **Both NL & CR**.

If you type AT in the message field and press enter, it should respond with OK.

And since we talked about the CH\_PD pin, remember that if you want to flash the ESP8266 you should connect the GPIO0 pin to GND (blue line), which puts the ESP into flash mode.