ESP-01 Ins and Outs

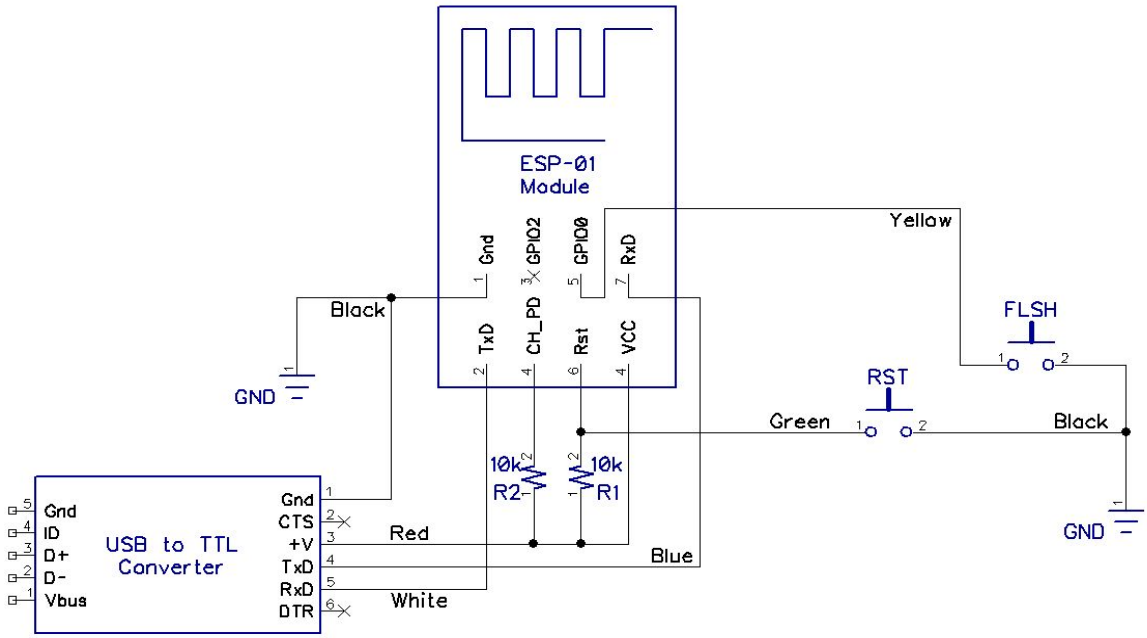
The ESP-01 module contains the ESP8266 MCU and a flash memory chip. There are two LED's: a red one which indicates power is connected to the module, and a blue one which indicates data flow, and can also be controlled by user programming. The Wi-Fi antenna is the PCB trace that covers the top of the module; it's called a Meandered Inverted-F Antenna (MIFA,) is surprisingly efficient, and only mildly directional.

There are eight connection pads near the bottom of the module; the figure above identifies their functions. Usually, two 4-pin male headers are inserted in the rear of the module and soldered on the front. This makes the I/Os accessible, but is not breadboard friendly, and requires flywires from the ESP-01 to a solderless breadboard.

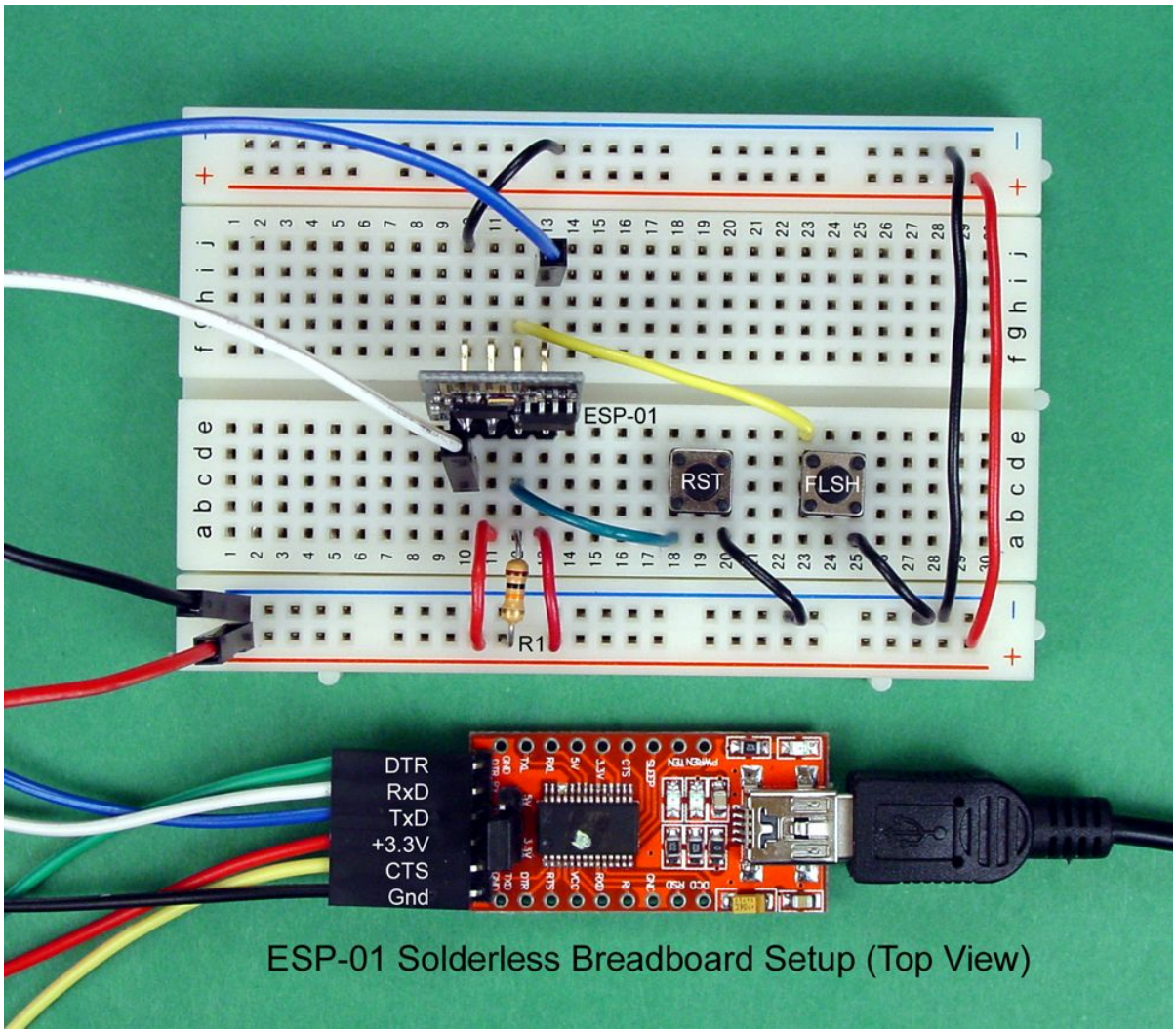
Connecting Things Together

The schematic diagram below shows the connections required to the ESP-01, and the photographs show the completed solderless breadboard assembly. The wire colors on the schematic correspond to the wire colors in the photographs.

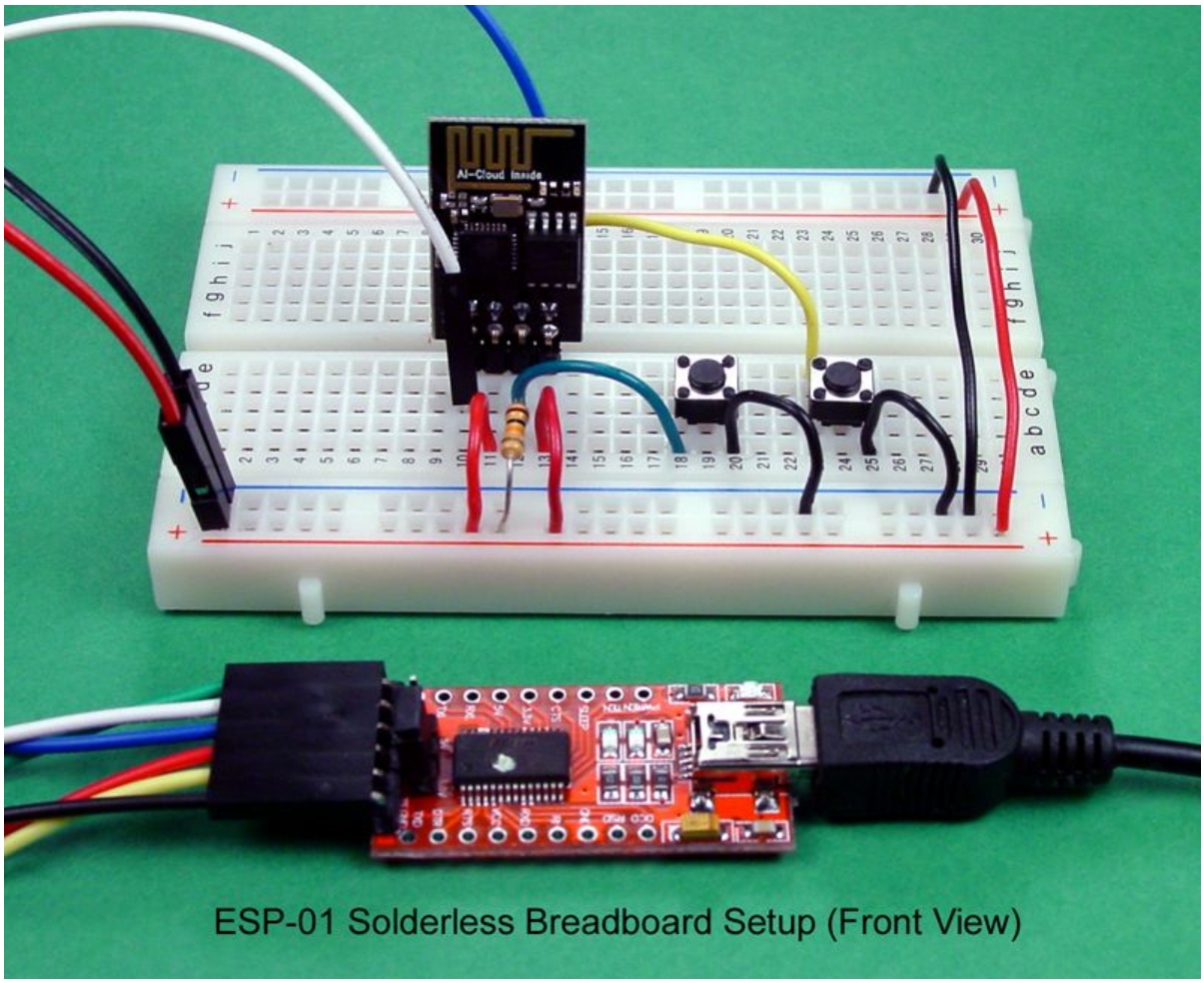
Construct the assembly as shown, but do not connect the cable from the USB to TTL converter to the PC until you have set the shunt on the converter PCB to the 3.3V position, and double checked all wiring. Using 5V to power the ESP-01 could damage it beyond repair.



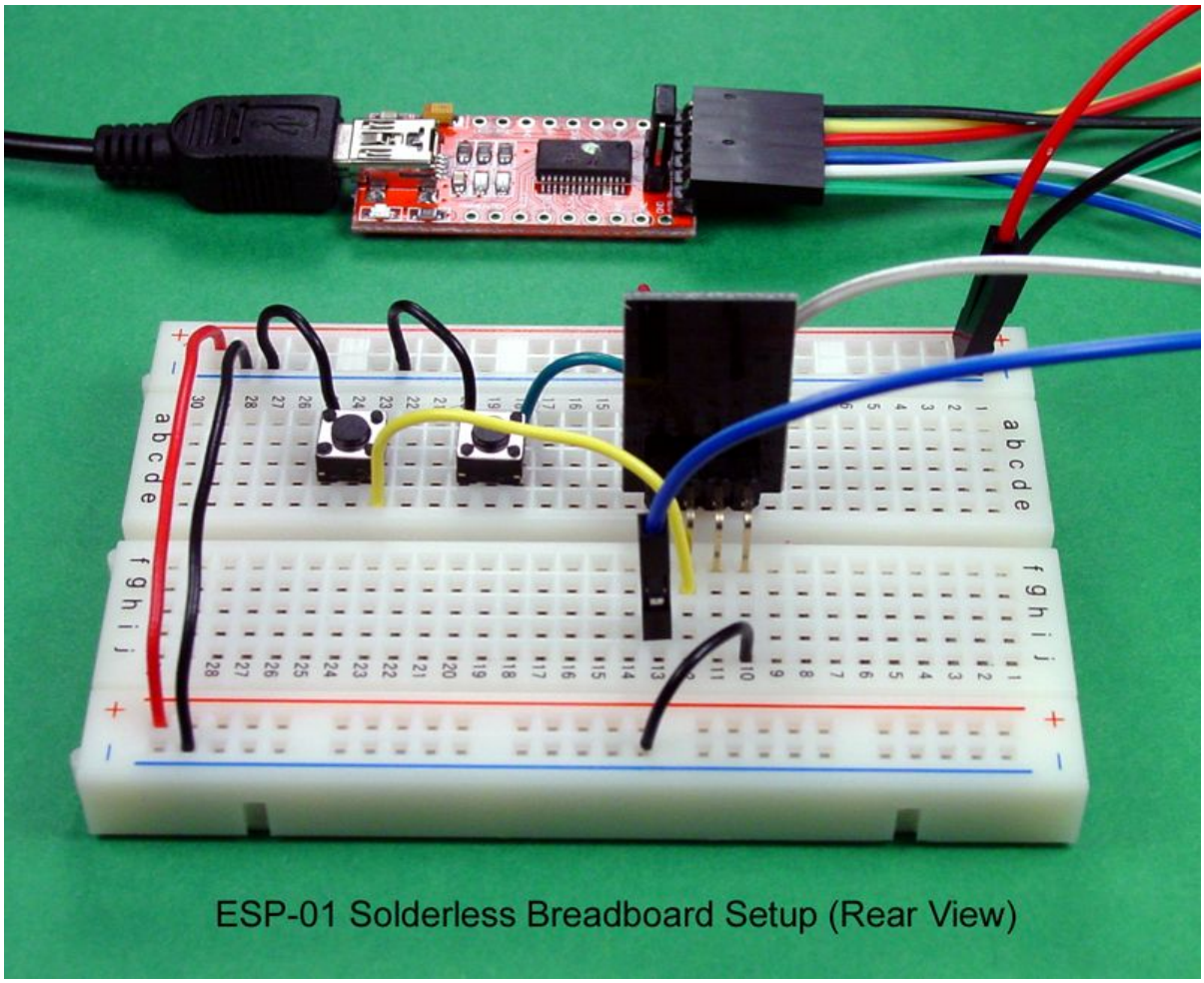
ESP-01 Connection Diagram



ESP-01 Solderless Breadboard Setup (Top View)



ESP-01 Solderless Breadboard Setup (Front View)



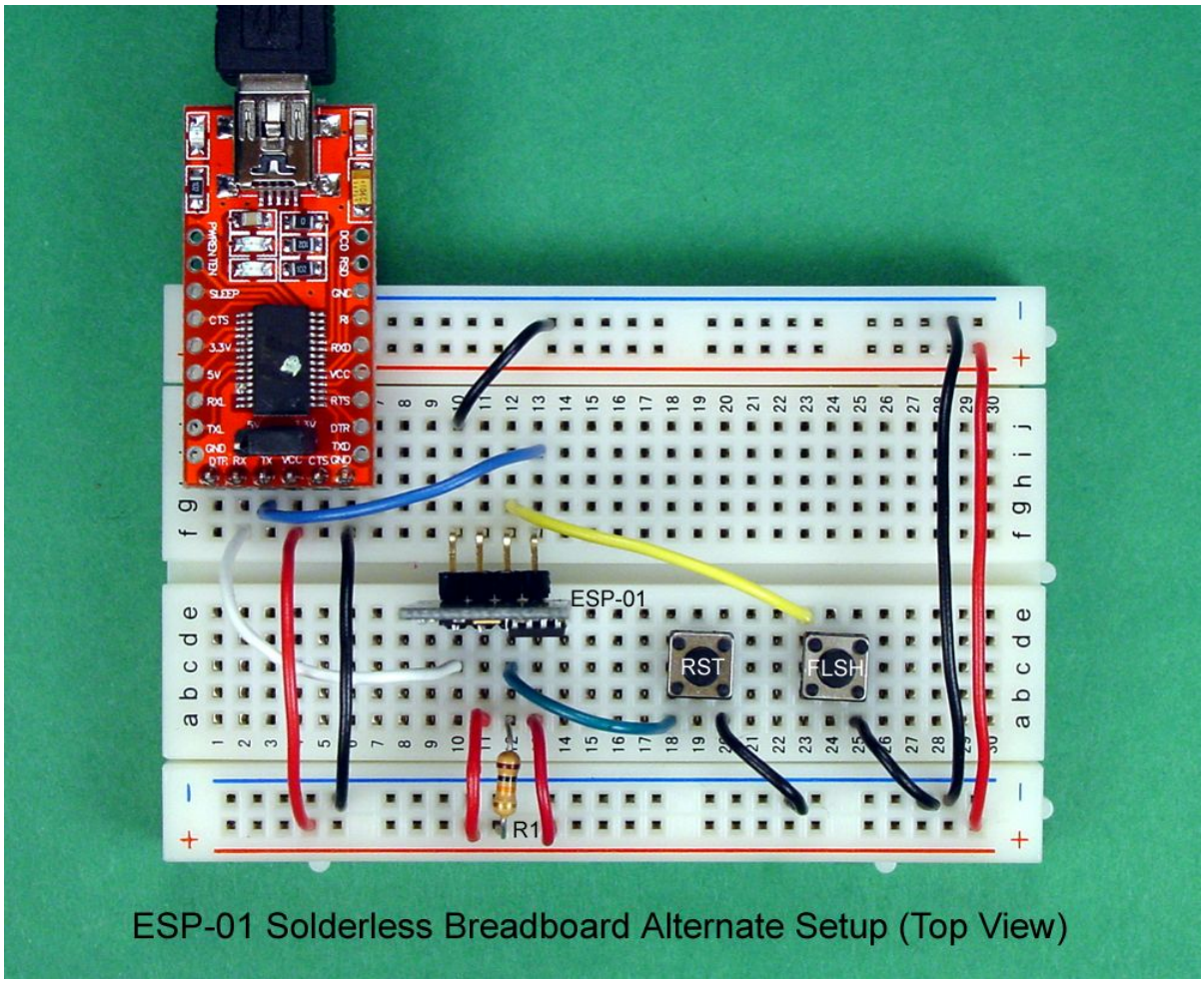
ESP-01 Solderless Breadboard Setup (Rear View)

Between the schematic diagram and the photographs, you should have most of the information needed to assemble the solderless breadboard setup. The notes below will also help.

- The USB to TTL converter shown in the photographs utilizes an FTDI 232 UART chip and works well with Windows, Mac, and Linux operating systems. It also provides a 3.3VDC power source for the ESP-01. Be certain that the shunt on the converter PCB is set to output 3.3V; that will ensure that both the supply voltage and the TxD signal will be at the correct voltage. Using a higher voltage could damage the ESP-01.
- Whatever USB to TTL converter you decide on should be installed and tested prior to using it with the ESP-01 breadboard setup. Drivers for FTDI devices are located at the FTDI web site.

- Estimates of the current required for the ESP-01 during Wi-Fi operation vary from 250mA to 750mA. The current supplied by the USB to TTL converter should suffice for programming the ESP-01, but may prove to be inadequate for long term use. A better choice is a filtered regulated 3.3VDC supply capable of 1A or more.
- The DTR and CTS leads from the USB to TTL converter are not required and are not connected.
- The two switches shown are normally open (NO) single pole momentary contact pushbuttons.
- One of the discrepancies in the information available for the ESP-01 is whether CH_PD should be connected directly to +3.3V, or should be connected through a 10k pull-up resistor. The author has tested both methods, and has found both to work. After you build and test the circuit as shown here (with CH_PD tied directly to +3.3V,) try using a 10k resistor instead of the direct connection. If the circuit works with the 10k pull-up resistor, leave it in the circuit.

As you see in the photographs above, the use of fly wires from the USB to TTL converter is not optimum. A better option is to replace the six right angle pins on the converter with six straight pins on the bottom of the PCB. The modification will allow the USB to TTL converter to be plugged into the solderless breadboard and will result in a much neater and less fragile assembly, as shown in the following photo.



ESP-01 Solderless Breadboard Alternate Setup (Top View)

Powering Up

Before connecting the USB to TTL converter to your PC, check to be sure that the voltage selection shunt is in the 3.3V position, and that all the wiring on the ESP-01 breadboard setup is correct and secure. Then, plug the USB cable in; the red LED on the ESP-01 should light and stay on, and the blue LED should flicker whenever there is signalling between the EXP-01 and the PC. Next, test the reset switch by pressing and and holding it down. Look at the ESP-01; when you release the reset switch, the blue LED should flash twice. If all is well at this point, disconnect the circuit from the PC and proceed to the next section.

Arduino IDE

The recommended Arduino IDE version for use with the ESP8266 modules is Version 1.6.5. If you have an earlier version, you can try it and see if it works, or you can upgrade to 1.6.5.

- Once the proper Arduino IDE is installed, start the program and click File, Preferences, and look for the entry box for Additional Board Manager URLs. Enter the following URL exactly as it is written and then click OK.
    - http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Next, click Tools, Board Manager, and scroll down the list to find "esp8266 by ESP8266 Community." Select this entry and click the Install button; downloading and installation will begin and continue for a few minutes. While it is installing, take a look at the different platforms supported. In addition to the generic ESP8266 Module, support is provided for the NodeMCU, the Huzzah, and the SweetPea. By the time you read this, probably more will have been added.
- When the installation is finished, click the Close button.
- Now, click Tools, highlight the Board selector, and choose "Generic ESP8266 Module."
- Click Tools again, and visually confirm that the board selected is Generic ESP8266 Module.
- Click File, Examples, and scroll down the list until you come to ESP8266WiFi, and then click WiFiScan. A new IDE window should open containing the WiFiScan sketch.

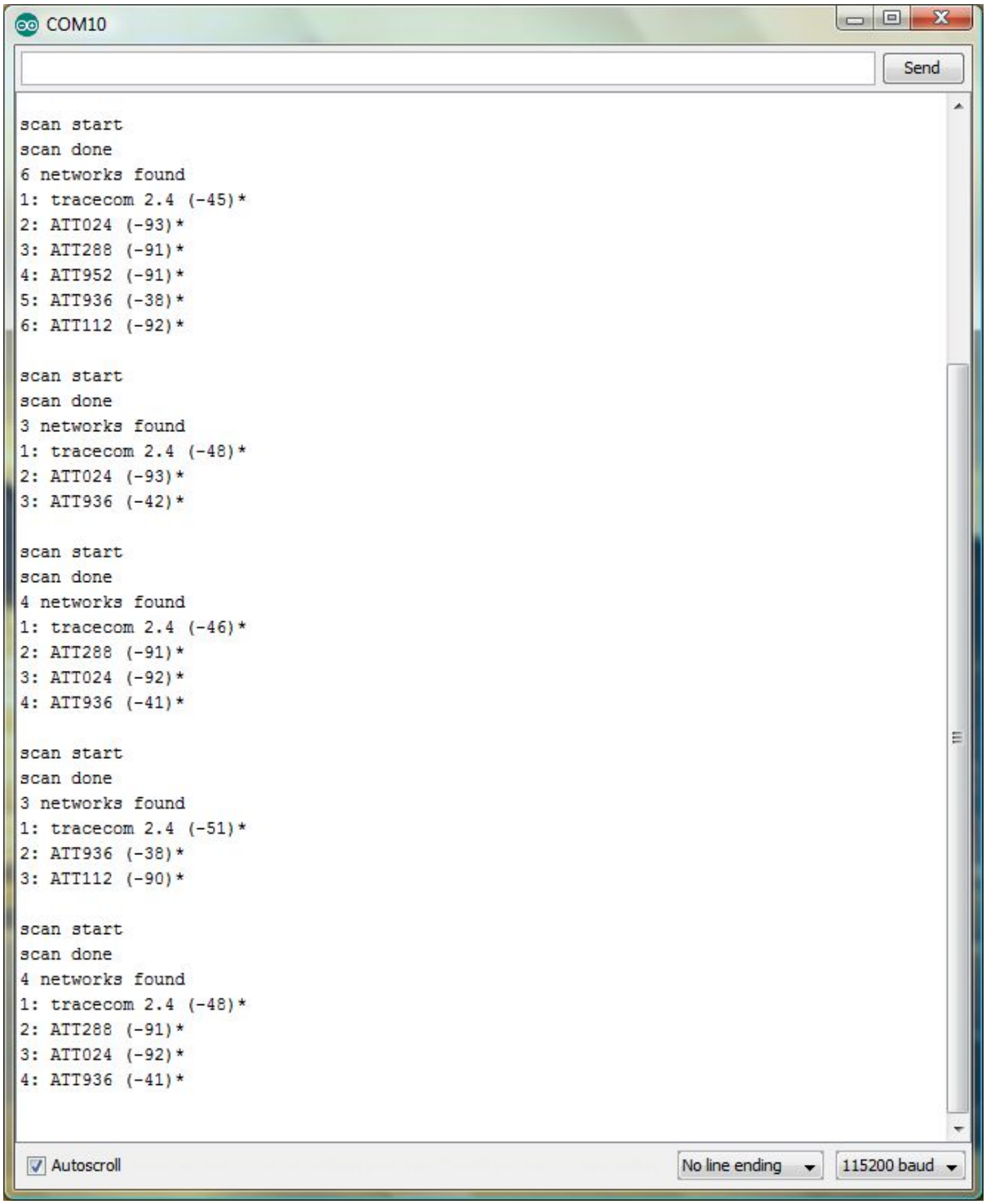
Reconnect the circuit to the PC and confirm that the red LED on the ESP-01 is lit. Click Tools, Port, and select the port where the ESP-01 is connected. Finally, you are ready to program the ESP-01.

Press and hold the Reset button, and then press and hold the Flash button. Release the Reset button, and while holding the Flash button pressed, click the

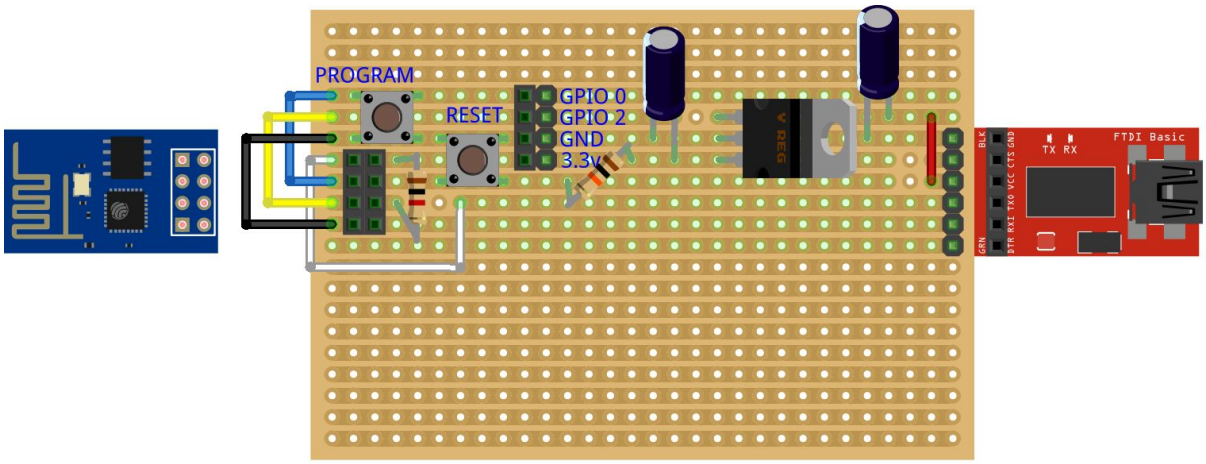
Upload arrow in the Arduino IDE. The sketch should compile in a minute or so, and when it is complete, release the Flash button. The compiled code will be sent to the ESP-01; as it is sent, the blue LED on ESP-01 will flicker.

To see the results of all this clicking and choosing, click Tools, Serial Monitor, and set the baud rate in the lower right corner of the Serial Monitor window to 115200. If you have an earlier version of the ESP-01 (probably built on a blue PCB,) the baud rate is most likely 9600.

The ESP-01 should be scanning for Wi-Fi networks and reporting the results in the Serial Monitor window, as shown in the example below.

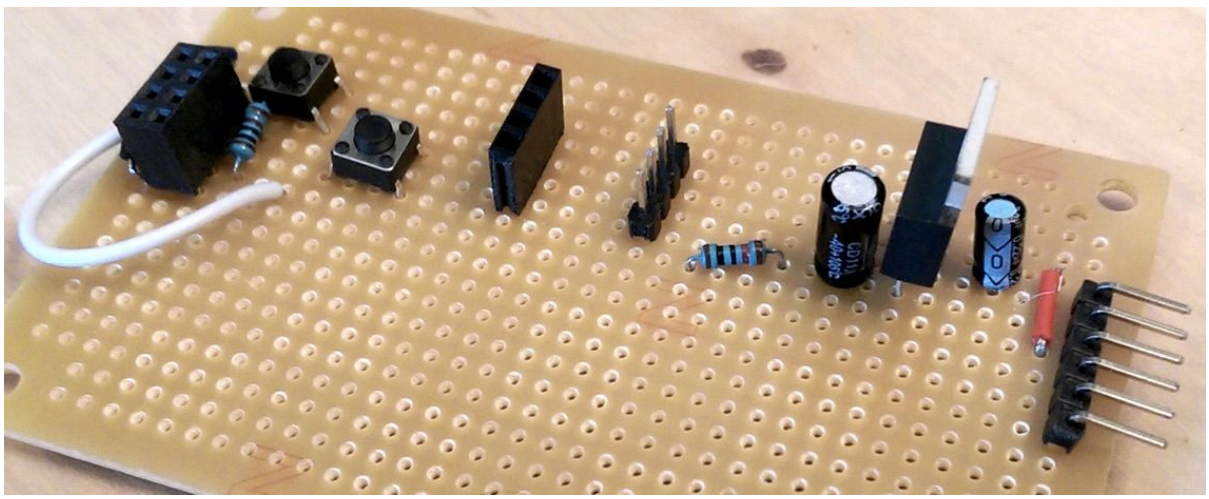


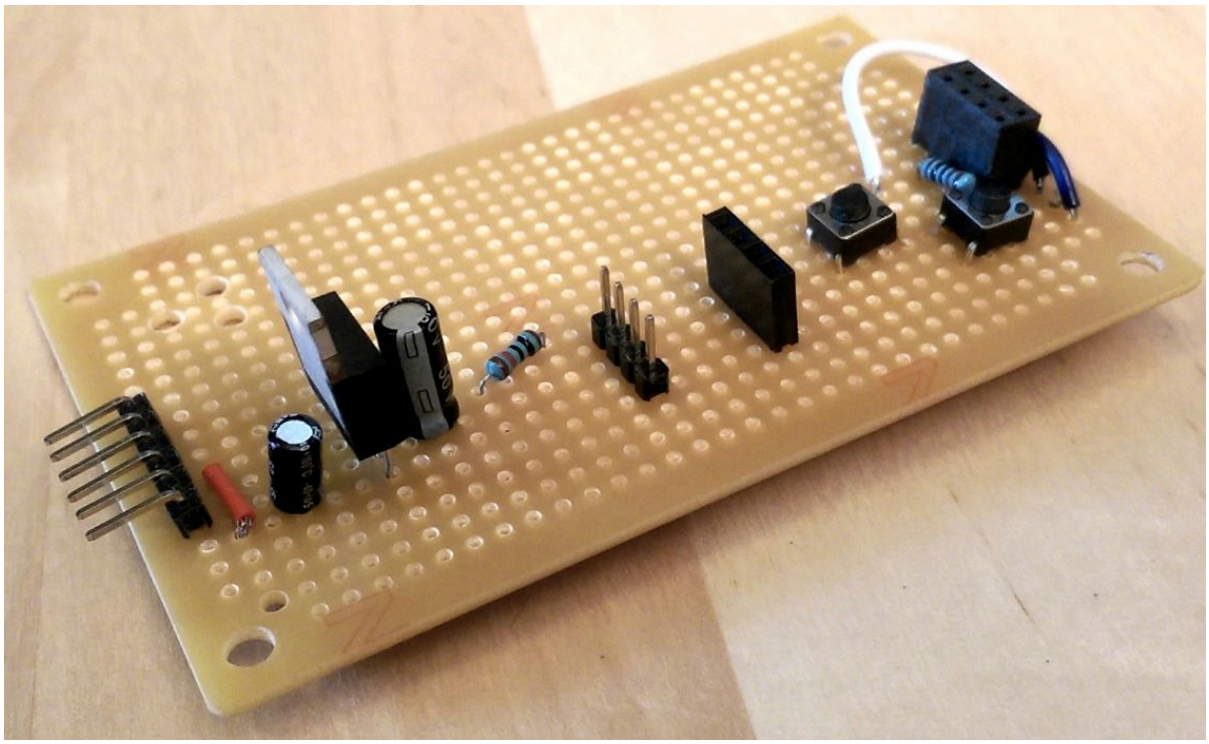
You should see your own network in the report, and all other networks that are within range of the ESP-01. The numbers in parentheses show each network's signal strength, and because the numbers are negative, lower numbers represent stronger signals.

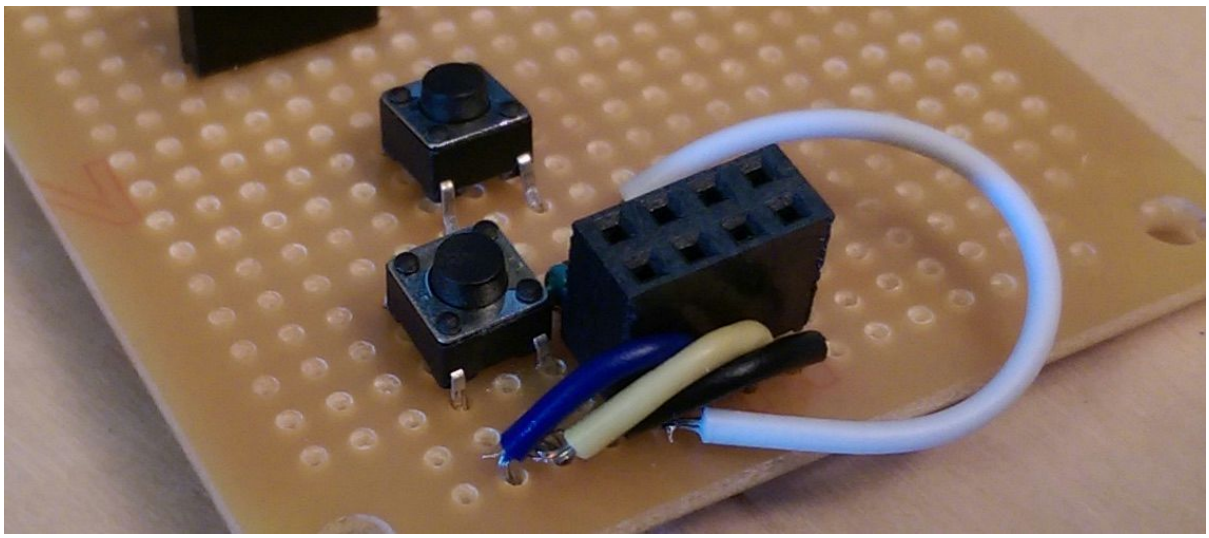


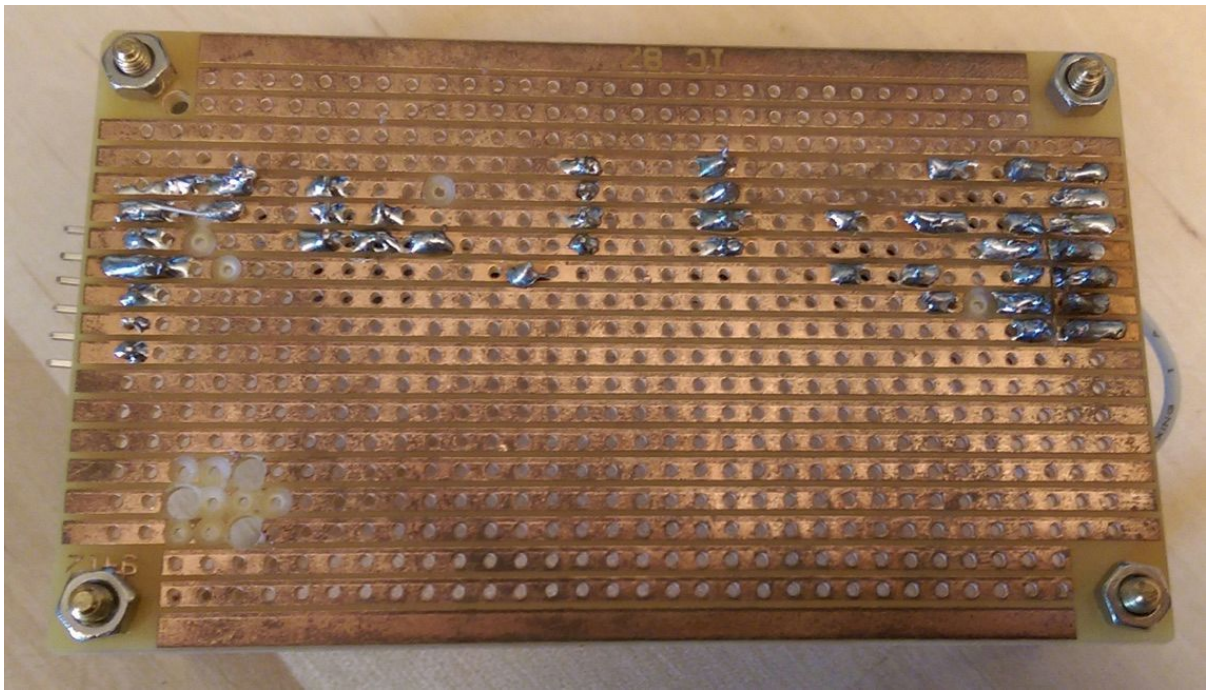
Connect everything as shown in the diagram. Be sure to cut the stripboard tracks between the two rows of pins where the ESP-01 connects. (And the other locations - the `CH_PD` track and a few on either side of the power tracks - but you can see each of those cuts in the graphic.)

I changed the spacing a bit when I assembled mine to give my fingers more room to connect and disconnect things from the pin headers.









Programing

There are a few ways to program the ESP-8266, but my favorite is to use the Arduino IDE with the ESP8266 board from esp8266.com. Adafruit has a nice guide for installing it.

When uploading your sketch, you have to be a little carefule about the order and timing of the button presses:

1. Click upload in the Arduino IDE
2. Quickly press and hold the RESET button (the one farther from the ESP-01)
3. Quickly press and hold the PROGRAM button
4. Watch the status line in the Arduino IDE. When it changes from "Compiling..." to "Uploading..." release the RESET button
5. After you see it starting to print the upload status in red text, release the PROGRAM button.

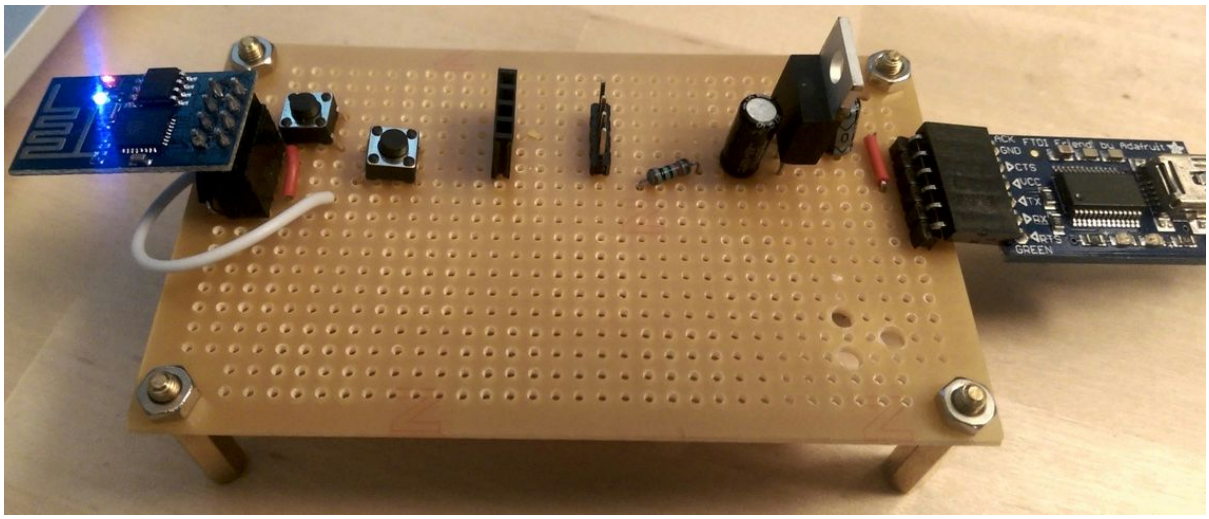
Here's an example sketch to use PROGRAM button on `GPIO 0` as a regular input:

```
const int buttonPin = 0;

void setup() {
 Serial.begin(9600);
 // Normally the button will take the pin from "floating" (not connected to anything) to grounded
 // This enables the internal pullup resistor so that the default state is high rather than floating.
 pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  Serial.print("Button is ");
  int status = digitalRead(buttonPin);
  // because of the pullup resistor, the pin state is reversed: button pressed = LOW, button released = HIGH
  if (status == LOW) {
    Serial.println("pressed");
  } else {
    Serial.println("released");
  }
  delay(500);
}
```

Happy hacking! Discuss on esp8266.com.


FTDI Friend by Adafruit
GND
CTS
VCC
TX
RX
RTS
GREEN