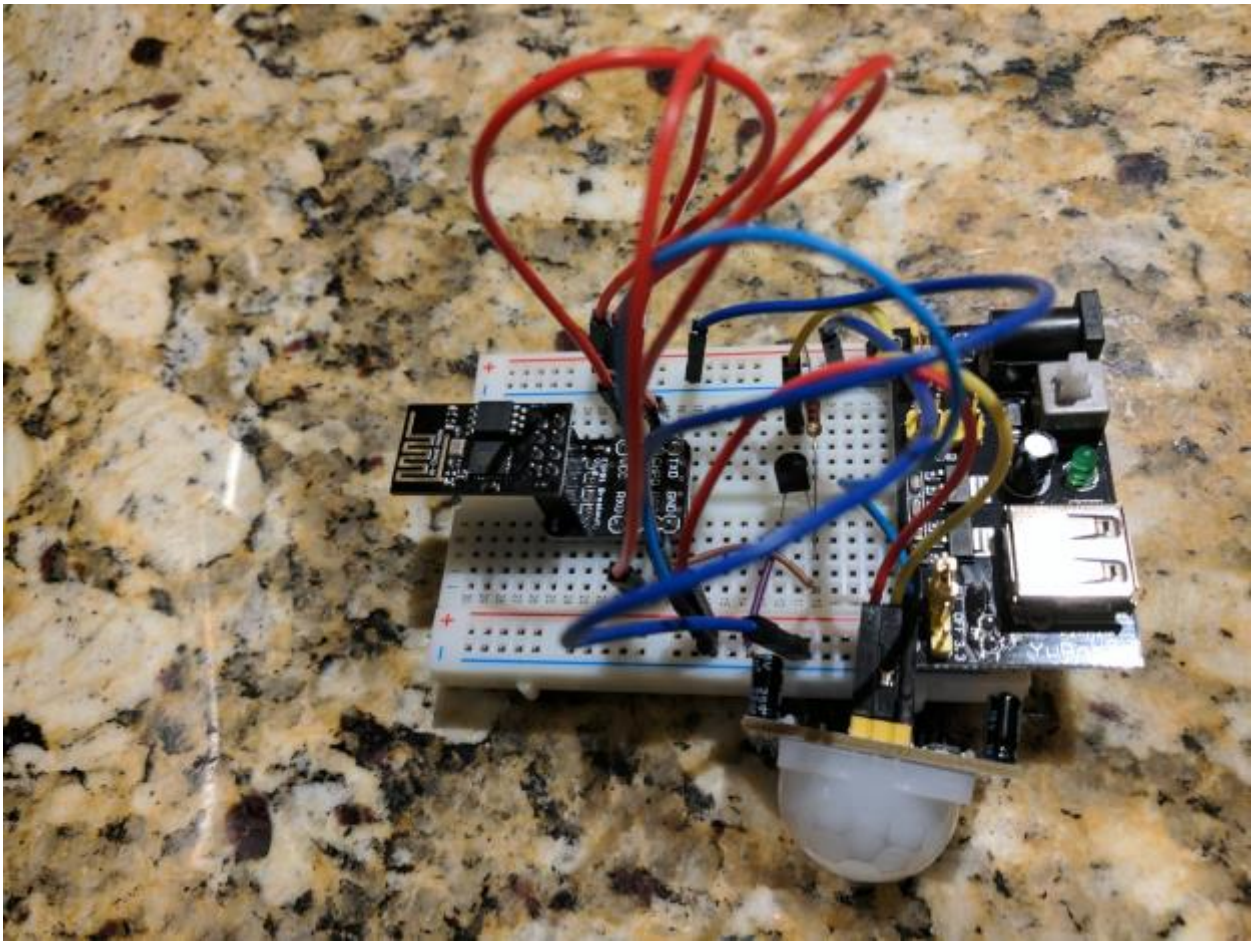
ESP8266-based Motion Detector for IFTTT Webhook Maker Integration
HARDWARE BUILD



Simply put, this project takes the output of a motion detector (I used Velleman's VMA314) and uses an inexpensive ESP8266 module (which has an embedded microcontroller compatible with the Arduino IDE) and the Webhook Maker module from IFTTT.com to trigger a web event (for me, a message into my personal Slack channel) when it senses motion.

AFTER loading the Sketch into the ESP8266 (see tutorials like this for help), the programmed module can be added to the circuit.

The materials required are:

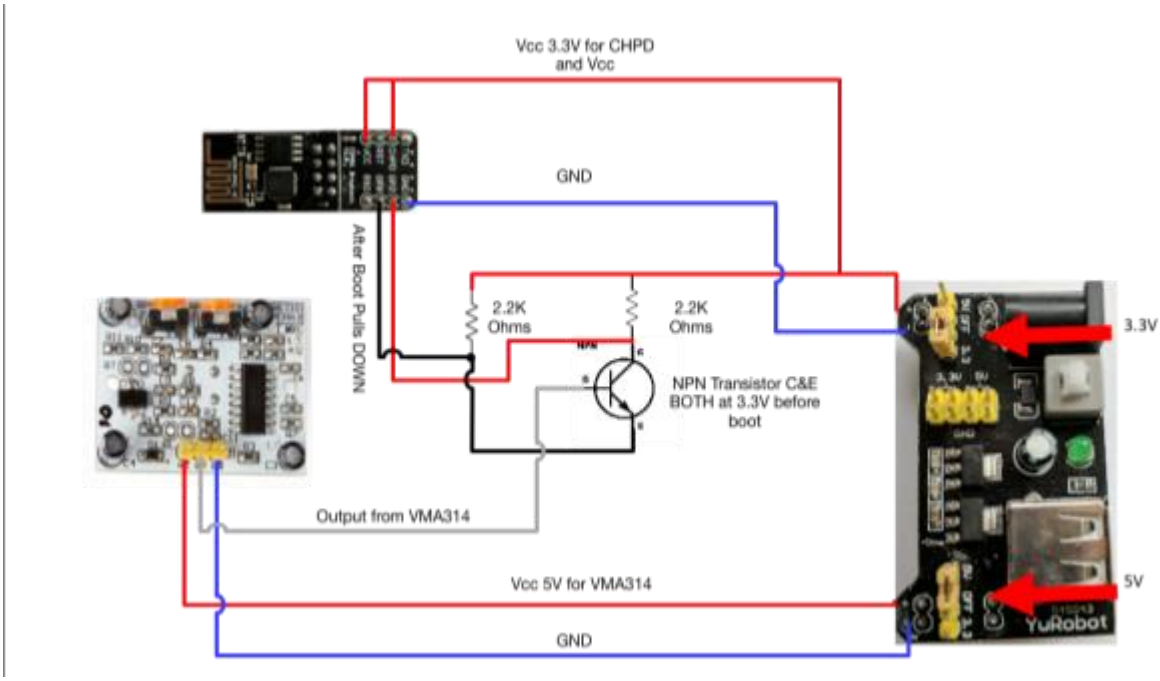
- An ESP8266 module (such as this one)
- A motion sensor (VMA314 or similar)
- A 3.3V and 5V (simultaneous) power supply (like this one)

- Two 2.2k Ohm resistors (I found these values by experiment – the documentation on the ESP8266 that I have found is inconsistent on the internal pull-up/down values used internally to the module, so I experimented with values until it worked).
- An N-P-N transistor (from any electronics supply shop)

Optionally:

- A breadboard
- A breakout board for the ESP8266
- Wires; OR
- A soldering iron, board etching kit, etc.

The circuit is described below:



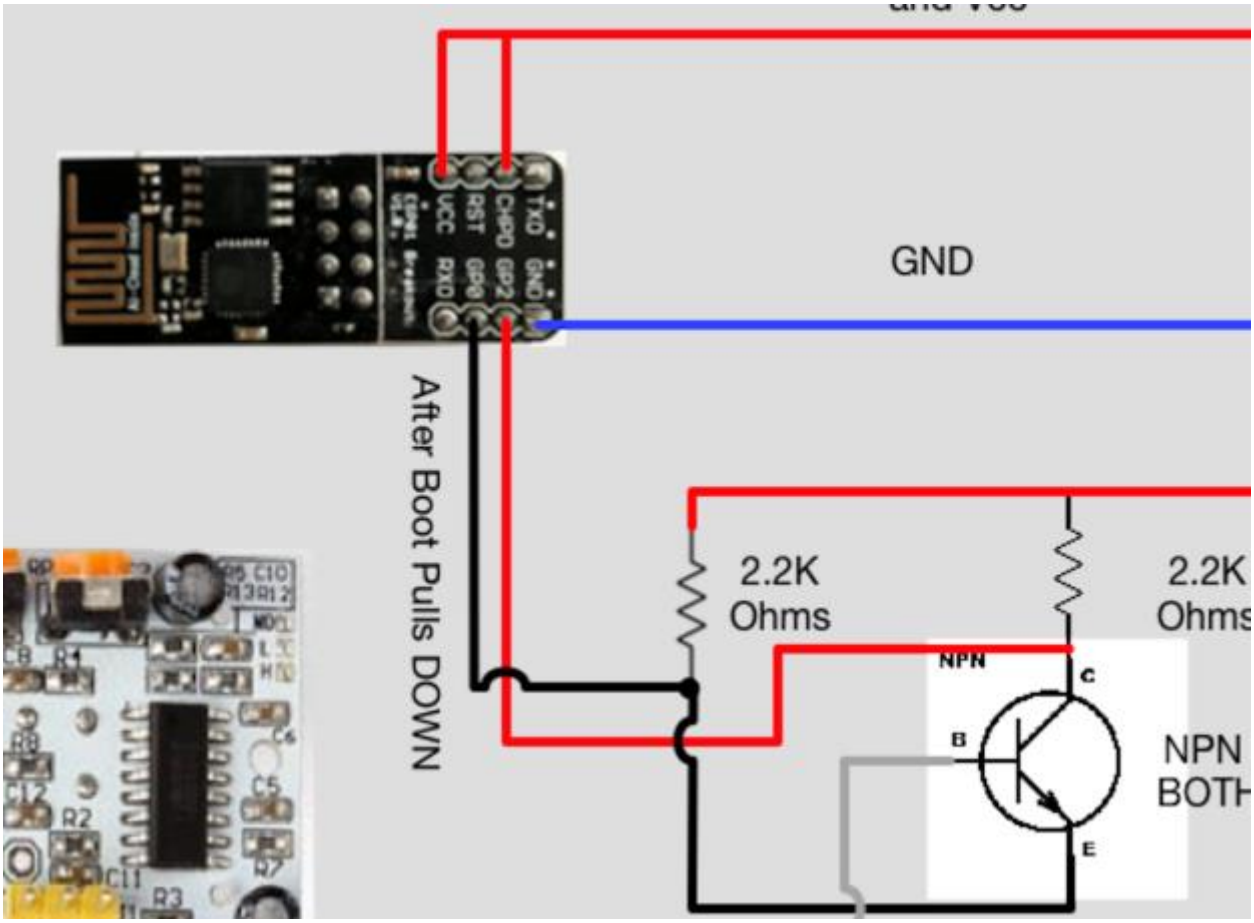
Of note: the VMA314 takes a voltage of 5V for Vcc, but outputs 3.3V on its signal line (HIGH Means DETECTION). When it powers up, it seems to asset DETECTION immediately for a second or so. These properties create two problems:

First problem – the VMA314 needs 5V, but the ESP8266 cannot tolerate 5V. This is why I used the power supply I used, but you may have other options available.

The second problem is that the ESP8266 needs its GPIO pins (and CHP0) HIGH in order to boot to its programmed firmware, so if you merely use an NPN transistor to invert the output signal of the detector, it will drive the GPIO LOW – which will prevent a boot.

My solution was to use the 2.2k Ohm pull-up to 3.3V on GP0, which is also tied to the Emitter of the transistor. The collector of the transistor is pulled up to 3.3V by a 2.2K Ohm resistor and to GP2. Finally, the output line of the VMA3145 connects to the base.

A blow-up of this is shown below:



The upshot of this is that on power-up, GP0 and GP2 are pulled HIGH, as needed, and the transistor has 3.3V on all of its pins (so it does nothing, having no potential difference across the gate).

When the ESP8266 finishes booting up, it drives GP0 LOW (Ground), and the emitter of the transistor is now connected to Ground, happily inverting signals from the VMA314.

Again, I came to these resistor values by experiment (using what I already had available), so Your Mileage May Vary…