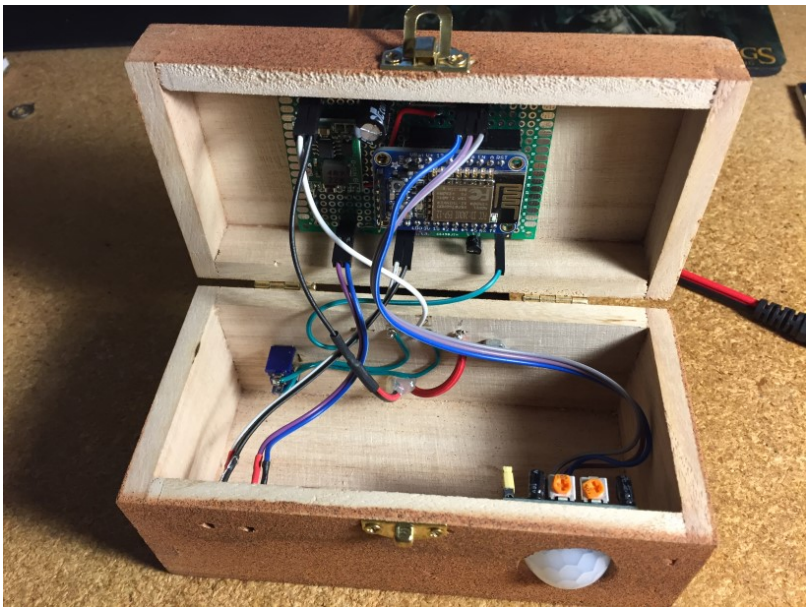
ESP8266 Motion Alarm

Version 2017-02-15 – R.Grokett



Overview

EEEOOO EEEOOO EEEOOO! You just triggered my motion detection burglar alarm system. But don't worry, here's how to build your own alarm.

I used the Adafruit HUZZAH ESP8266 as it has good tutorials for initially getting it set up. I also used the Arduino IDE with the ESP8266 library, since I was already quite familiar with using it with the Huzzah ESP8266. A PIR sensor is used to detect motion.

Note that a PIR motion detector alarm is not good in a location that may have pets, strong sunlight or motion from fans or wind, each of which could trigger false alarms. Mine is placed in a hallway that has no windows and no pets have access.

This simply alarms when motion is detected. It does have a delay so that you can enter and exit the room without triggering the alarm.

My Alarm horn uses 12v DC while the HUZZAH ESP can only use 4v-6v input (a bare ESP8266 can only work with 3.3v!). So I used a small step down inverter to convert 12v to 5v.

I also decided to interface this to IFTTT (www.ifttt.com) to send an SMS message each time motion was detected.

Note that IFTTT requires HTTPS SSL encryption. Thus this project includes code for that.

Parts List

- Adafruit HUZZAH ESP8266 https://www.adafruit.com/product/2471
- PIR Motion Detector such as https://www.adafruit.com/products/189
- FTDI or USB console cable https://www.adafruit.com/products/954 or equiv
- Alarm buzzer/horn – the louder the better! 6-12v DC
- 6-12V DC 1amp power supply (match voltage of your alarm horn)
- 6-12v to 5V DC-to-DC converter such as SeeedStudio https://goo.gl/eljYTu or https://www.adafruit.com/products/2190
- 1 – Red LED
- 1 – Green LED
- 2 – 220ohm resistors
- 1 – 1K resistor
- 1 – 2N2222 NPN transistor
- 2 – SPST toggle switches
- Breadboard, wire, box to put everything in
- Arduino IDE with ESP8266 extension package installed *(see Initial Setup below)*
- Download the ESP8266_PIR_ALARMsoftware from GitHub (https://github.com/rgrokett/ESP8266_PIR_Alarm/)

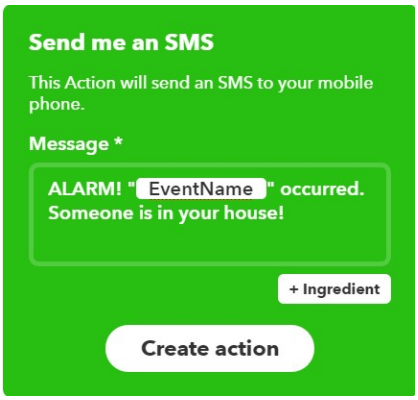
IFTTT Setup

1. Go to www.ifttt.com
2. Log in. If you don't have an account, you can sign up. It's free.
3. Once logged in, click on **My Applets**
4. Click on **New Applet**
5. Click the "**this**" of **ifthisthenthat**…
6. Type "**Maker**" into the Search services box
7. Click on the Maker icon

8. For Choose a Trigger, there is only one big gray box with "Receive a web Request". Click it
9. For Complete Trigger Fields, enter "**piralarm**" and click Create. This is the event name used in the ESP8266 .ino software. They must match.
10. Click the "**that**" of **ifthisthenthat**…
11. Type "sms" into the Search Actions box. You could change this to do other things, like send email messages, etc. But let's stick to sms. You can always edit later.

12. Click on the green SMS icon

13. Click Connect & enter your mobile phone number. Verify it is correct!
14. Click Send PIN & enter it.
15. Now click on the **Send me an SMS** icon.
16. You can edit the text or just leave it as is.



17. Click Create Action. You will see a screen that just describes what this recipe will do.
18. Click on the "**Receive notifications when this Applet runs**" box to activate it.
19. Click **Finish**
20. You now have an IFTTT Recipe.

IFTTT Maker URL

You will need Maker URL assigned by IFTTT in order to send from ESP8266 to IFTTT.

1. Click on your name in the upper right corner of the screen and select **Services.**
2. Again, click on the **"M" Maker** icon.
3. Click on **Settings** button.

4. On the Maker Settings screen, you need to copy the key portion of the "URL:" field.
   Example: `https://maker.ifttt.com/use/`**`aBc1fakekey2ab3cBA`**
   It is needed for the .ino program later on.

Any IFTTT Recipe that uses Maker channel can be used, as long as it is called "piralarm". (You can change the trigger name in the ESP8266_PIR_Alarm.ino program, if desired.)

Important Initial Setup of ESP8266

Before beginning the project, you should become familiar with the Adafruit HUZZAH board and programming it using the Arduino IDE. The best way is to use the excellent Adafruit tutorial:

https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout/using-arduino-ide

You must be able to program your ESP8266 and connect wirelessly to it via browser as described in their tutorial. Once completed, THEN continue below.

Software

You should program and test the ESP8266 before adding its hardware wiring.

1. **STOP**! Be sure you have already completed the Adafruit tutorial software setup of the Arduino IDE and tested the ESP8266 with your WiFi network as described in the Initial Setup section above!

2. Ok, download the ESP8266_PIR_ALARMsoftware from GitHub (https://github.com/rgrokett/ESP8266_PIR_Alarm/)

3. Copy the **ESP8266_PIR_Alarm** subdirectory into your Arduino IDE development directory. This folder has the 3 software files needed.
   `ESP8266_PIR_Alarm.ino`
   `HTTPSRedirect.h`
   `HTTPSRedirect.cpp`

4. Double-click the **ESP8266_PIR_Alarm.ino** program to load it into your Arduino IDE.

5. Using Arduino IDE, edit the ESP8266_PIR_Alarm.ino and insert your WiFi SSID and PASSWORD into the appropriate places.

6. Update the api_key with your IFTTT Maker URL key copied previously. You can look on https://ifttt.com/services/maker/settings, if needed.

7. You can also change a few variables, described here:

```
const char* ssid     = "{YOUR_WIFI_SSID}"; // Your WiFi SSID
const char* password = "{YOUR_WIFI_PWD}";  // Your WiFi Password

const char* api_key  = "aBc1fakekey2ab3cBA";   // Your URL Key from
https://ifttt.com/services/maker/settings
const char* event    = "piralarm";       // Your IFTTT Event Name

bool verifyCert = false; // Select true if you want SSL certificate
validation
```

IFTTT requires HTTPS SSL and HTTPS 302 Redirection. The ESP8266 Library (installed in the Adafruit Tutorial) contains the HTTPS SSL functions and an extension of that library was developed by https://github.com/electronicsguy/ESP8266/tree/master/HTTPSRedirect to handle the HTTPS 302 Redirection. Since this code wasn't in the ESP8266 Library, I have included a copy or you can get the latest version from the URL above and add the .cpp and .h files to the ESP8266_PIR_ALARMfolder.

The "WiFiClientSecure" provides SSL encryption, so the messages are always sent encrypted, but by default, the verification of the IFTTT's SSL Certificate is turned off. You can turn it on by changing `verifyCert = true;`
This requires the SHA1 Fingerprint of the IFTTT server to be used to verify the certificate.

```
const char* SHA1Fingerprint="A9 81 E1 35 B3 7F 81 B9 87 9D 11 DD 48 55
43 2C 8F C3 EC 87";
```

This fingerprint is initially retrieved from the IFTTT server using the Linux command:

```
$ openssl s_client -servername maker.ifttt.com -connect
maker.ifttt.com:443 | openssl x509 -fingerprint -noout
```

Replace colons with spaces in result and update the ESP8266_PIR_Alarm.ino as needed.
You should NOT have to change this unless IFTTT changes their SSL Certificate.
Again, you can bypass this check by setting `verifyCert = false;`

The IFTTT server initially returns a 302 Redirect message back, so the "HTTPSRedirect.cpp" software invisibly handles resending the request to the new host.

8. Compile and Upload the program using the FTDI or USB console cable just like shown in the Adafruit tutorial. Remember, you have to press the tiny GPIO0 and RESET buttons on the HUZZAH ESP8266 (aka *bootload* mode) to allow the upload to occur.

9. When the program finishes loading, open a Serial Monitor, set to 115,200 baud, and press the ESP8266 RESET button to restart the program running.

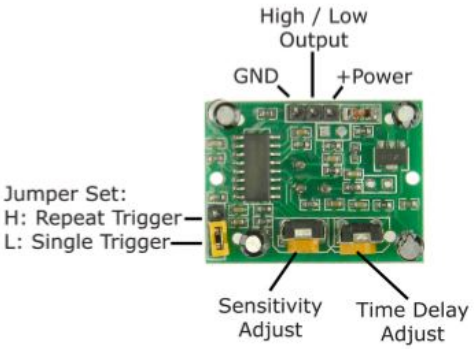
10. It should display the IP address in the Serial Monitor once connected to your Wifi.
    Also, the onboard red LED should blink 4 times signifying it's successfully connected.

11. Time for wiring everything…

Hardware – PIR

1. Temporarily unplug the FTDI/USB cable from the PC to power off the ESP8266.

2. Wire the PIR sensor as follows. Note that the PIR is powered by 5V, but its I/O line is 3.3v which makes it directly compatible with the ESP8266's 3.3v GPIO pins.



ESP8266 HUZZAH	PIR SENSOR
GND	GND
V+	VCC
GPIO 14	OUT

3. With the PIR now wired in, reconnect the FTDI/USB cable to the PC.
4. Again, start the Serial Monitor from the Arduino IDE.
5. Reset the ESP8266 and you should see the LED blink 4 times and the IP address displayed again.
6. The alarm has delays built-in that will allow you to activate/deactivate the alarm as you leave or enter the room.

```
int ALARM_DELAY = 60; // Delay in seconds when activating alarm/deactivating alarm
```

7. Once you see “Alarm now Active” message, if you move in front of the PIR, the Serial Monitor should register the event and send off to IFTTT. If IFTT trigger is successful, you should see a 200 OK HTTP response message and text and receive an email.

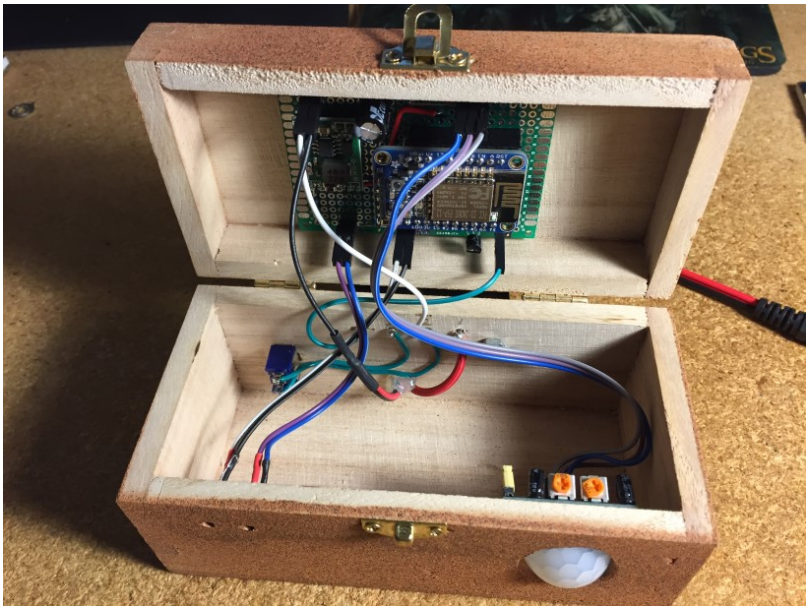
```
< HTTP/1.1 200 OK
< Server: Cowboy
< Connection: keep-alive
< X-Powered-By: Sad Unicorns
< X-Top-Secrettt: VG9vIGVhc3k/IElmIHlvdSBFK3.../NlY3JldEBlIHdnQgTWFrZXJzLg==
< Content-Type: text/html; charset=utf-8
< Content-Length: 50
< Etag: W/"32-44d0098f"
< Date: Wed, 29 Jun 2016 21:25:32 GMT
< Via: 1.1 vegur
<
* Connection #0 to host maker.ifttt.com left intact
* Closing connection #0
* SSLv3, TLS alert, Client hello (1):
```

8. Congratulations! You've fired the piralarm event

9. Again, the alarm delays a minute or two before resetting.

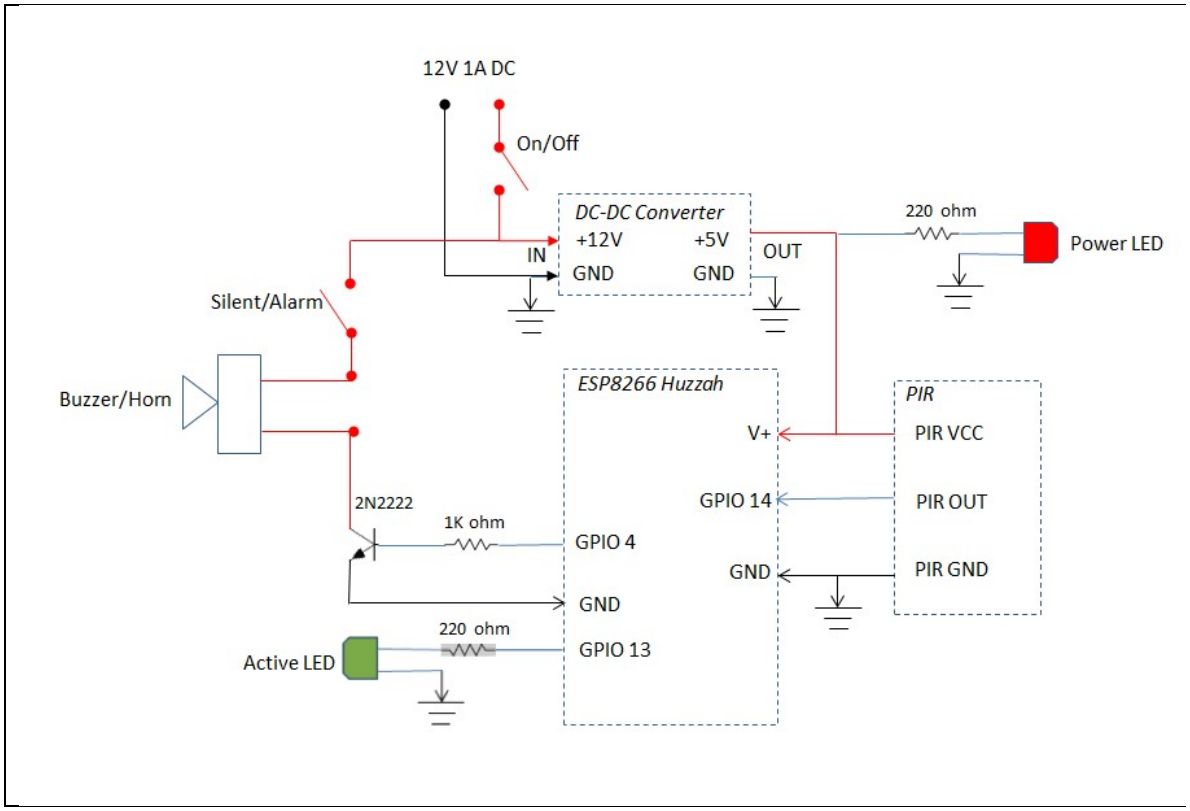
10. NOTE: if the PIR is too sensitive, turn down the Sensitivity adjustment on it (turn counter-clockwise). Otherwise, you might get the dreaded FALSE ALARM!

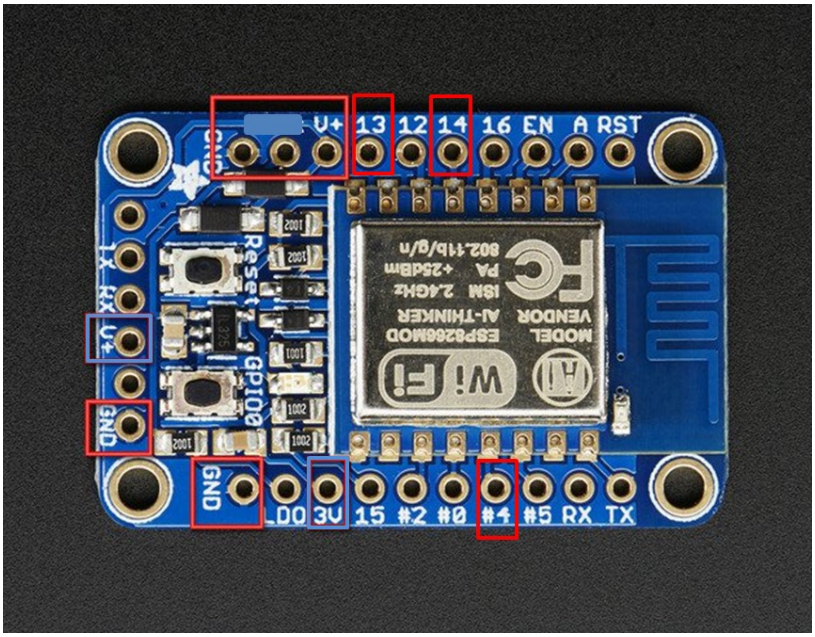
11. Unplug the FTDI/USB cable.



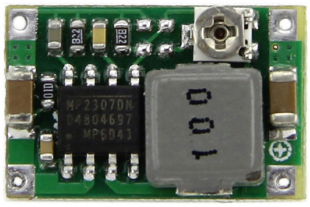
Hardware - Wiring

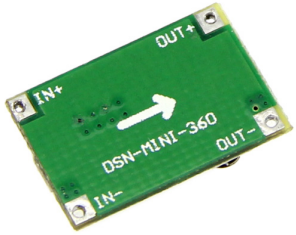
1. Using the wiring diagram, you should wire up all the pieces.





2. Wire the 12v to 5v DC-to-DC converter making sure of the 12V Input vs 5V Output side.

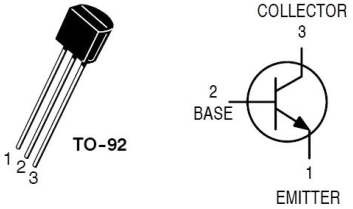




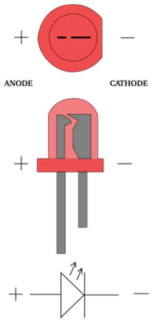
Note: there is a tiny potentiometer on the board that you will need to adjust to set the 5V output. You MUST connect 12v up to it and adjusting with a voltmeter BEFORE wiring it into the ESP8266 module. Set to slightly above 5V (such as 5.2v) to handle any sag once loaded.

3. Wire the 2N2222 NPN switching transistor to GPIO 4 via a 1K ohm resistor.

2N2222



4. There are two LEDs; one RED for Power light and the other Green for Active light. The longer lead goes to the + voltage (or GPIO repectively) side with the 220 ohm resistor. The shorter lead goes to ground.



5. There are two SPST switches, one for Power On/Off and the other for Audible/Silent Alarm.
6. The Buzzer/Horn should be located externally (Ideally, high up out of reach).
7. The rest can be installed into a box to be placed on a shelf.

Using Your Alarm

1. Once ready, plug in your 12v power supply and turn on the Power switch. This is how you will activate/deactivate your alarm.
2. After a few seconds, you should see the green LED blink four times to indicate it connected to your WiFi.
3. After another minute, during which the green LED will flash and buzzer will issue short beeps every few seconds letting you know it is about to activate. Leave the room else you will set it off!
4. By flipping off the Silent/Alarm switch, you can turn off the noisy buzzer, but you will still get the SMS message.
5. Do some motion, again the green LED will flash and buzzer will beep to give you time to deactivate it (i.e. turn off the Power switch). If you don't, the buzzer will sound full on and an SMS message will be sent. The alarm will reset itself after a couple minutes. If you move again, it will go thru the cycle again.

   Have Fun!