



Raspberry Pi Garage Door Opener

Created by Lewis Callaway



Last updated on 2014-03-14 01:30:14 PM EDT

Guide Contents

Guide Contents	2
Overview	3
Parts Needed	4
Parts:	4
Tools:	4
Web IO Pi Installation	5
Web IO Pi Configuration	9
Motion Setup	12
Hardware	14
Hardware	14
How It Connects to the Relay	14
How it Connects to the Raspberry Pi	15
Soldering the Relay Board Time!	15
Step One:	16
Step Two:	17
Step Three:	18
Step Four	19
Step Five	20
Step Six	21
Network Configuration	22
Step One:	22
Step Two:	22
Step Three:	22
Step Four:	22
How to Control	24

Overview

Have you ever forgotten to close your garage door and remembered to shut it later? In this tutorial I will show you how to open and close your garage door over the internet using a Raspberry Pi, relay, and Web IO Pi. There's even a webcam attached to the Pi so you can rest assured that the door is really closed.



Before you start this guide you will want to follow a few others.

- Follow [this guide \(http://adafru.it/aUa\)](http://adafru.it/aUa) to set up your Raspberry Pi
- SSH is how we will be sending all the code in this tutorial to the Raspberry Pi so follow [this tutorial \(http://adafru.it/aWc\)](http://adafru.it/aWc) to find out how.
- When the Raspberry Pi is in the garage it will be communicating to the internet via WiFi. Use [this tutorial \(http://adafru.it/cg7\)](http://adafru.it/cg7) to set it up.

Parts Needed

Parts:

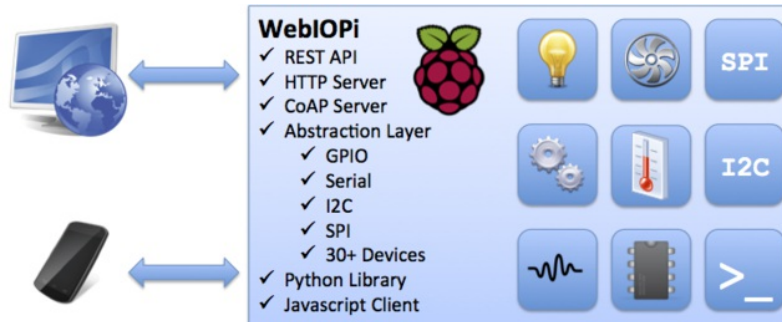
- [Raspberry Pi Model B \(http://adafru.it/998\)](http://adafru.it/998)
(The Model B is better for two reasons: One is that it has more power for streaming the webcam feed. Also it has an ethernet jack for prototyping and two USB ports for the webcam and WiFi dongle.)
- [Relay \(http://adafru.it/d6R\)](http://adafru.it/d6R) and [Perma-Proto PCB \(For Soldering Relay to\) \(http://adafru.it/589\)](http://adafru.it/589)
- OR a [PowerSwitch Tail \(http://adafru.it/268\)](http://adafru.it/268)- which is even better and much easier/safer to wire up
- [USB WiFi Dongle \(http://adafru.it/814\)](http://adafru.it/814)
- Two different types of wire:
[Female Jumper Wires \(http://adafru.it/266\)](http://adafru.it/266) (This is used to connect the GPIO wires to the relay)
Regular Hookup Wire (For hooking up the garage door to the relay)
- [5V USB Power Supply \(http://adafru.it/501\)](http://adafru.it/501) and [Micro USB Cable \(http://adafru.it/592\)](http://adafru.it/592)
- [Webcam \(http://adafru.it/d6S\)](http://adafru.it/d6S) (Any webcams at this link will work. I prefer ones that don't use a powered hub, but one that needs it would work.)
- [SD Card \(I recommend getting one preformatted\) \(http://adafru.it/1121\)](http://adafru.it/1121)
- [1N4001 Diode \(http://adafru.it/755\)](http://adafru.it/755)

Tools:

- Soldering Iron
- Screwdriver
- Wire Strippers
- Wire Cutters
- A computer!

Web IO Pi Installation

The framework we are going to use is called Web IO Pi. The Google code page for Web IO Pi is available [here \(http://adafru.it/d6T\)](http://adafru.it/d6T). Their website has a great overview image on what it can do.



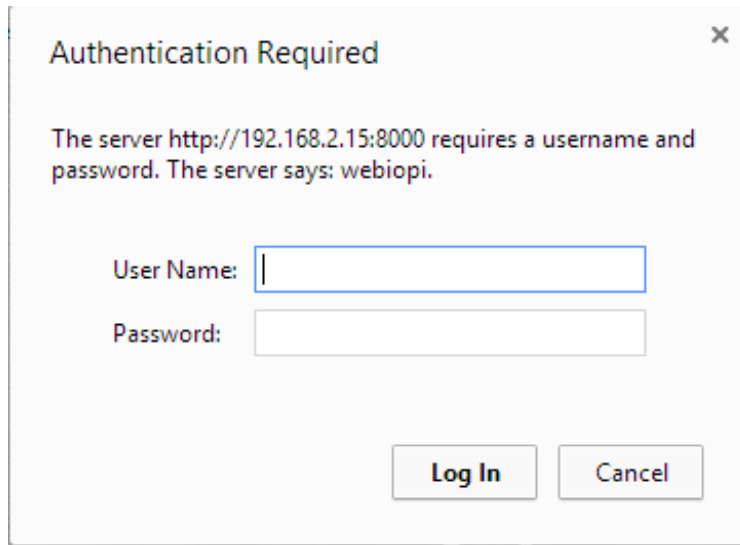
The first thing we need to do is install Web IO Pi. We can do it two ways. One way is to download it from the Pi Store, but if you do that you need to plug in a monitor, keyboard, and mouse. Also you need to have an account, and signing up/logging in didn't work for me. Downloading it through SSH worked wonders! Below is the code you need to install it.

```
$ wget http://webiopi.googlecode.com/files/WebIOPi-0.6.0.tar.gz
$ tar xvzf WebIOPi-0.6.0.tar.gz
$ cd WebIOPi-0.6.0
$ sudo ./setup.sh
```

Now we need to set it up so Web IO Pi runs when the Pi is booted up.

```
$ cd
$ sudo update-rc.d webiopi defaults
$ sudo reboot
```

About a minute after rebooting the Raspberry Pi, go to the IP address of the Raspberry Pi, followed by the port number of Web IO Pi which is 8000. For me it was 192.168.2.15:8000. The IP Address is the same as what you used for SSH. Once you arrive at the Web IO Pi page you are greeted with this:

A screenshot of a web browser's authentication dialog box. The title bar says "Authentication Required" with a close button (X) in the top right corner. The main text reads: "The server http://192.168.2.15:8000 requires a username and password. The server says: webiopi." Below this text are two input fields: "User Name:" followed by a text box containing a single vertical bar (cursor), and "Password:" followed by an empty text box. At the bottom right are two buttons: "Log In" and "Cancel".

Authentication Required

The server http://192.168.2.15:8000 requires a username and password. The server says: webiopi.

User Name:

Password:

Log In Cancel

The default login for this is:

User Name: webiopi

Password: raspberry

Instructions to change the login are available [here \(http://adafru.it/d6U\)](http://adafru.it/d6U).

Next for basic control click on GPIO Header.

WebIOPi Main Menu

[GPIO Header](#)

Control and Debug the Raspberry Pi GPIO with a display which looks like the physical header.

[GPIO List](#)

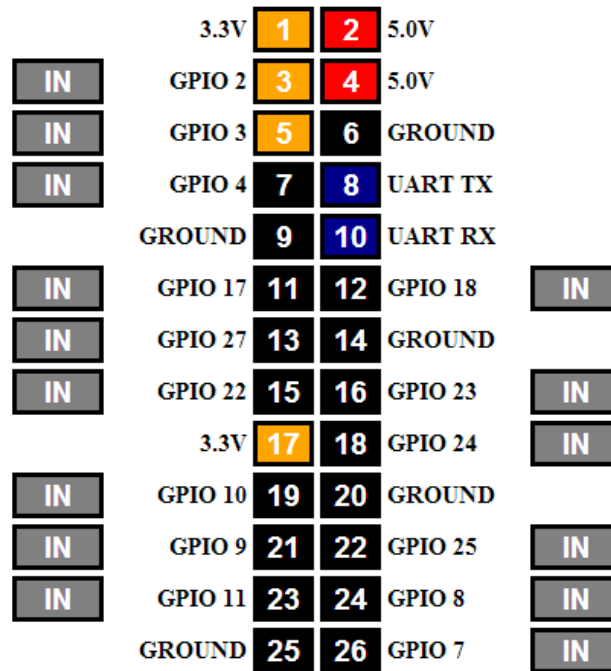
Control and Debug the Raspberry Pi GPIO ordered in a single column.

[Serial Monitor](#)

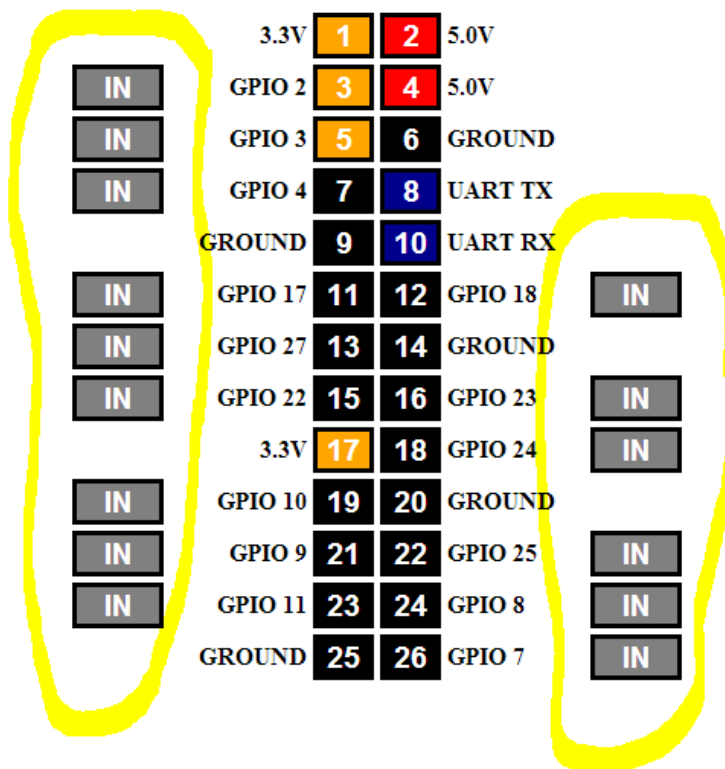
Use the browser to play with Serial interfaces configured in WebIOPi.

[Devices Monitor](#)

Control and Debug devices and circuits wired to your Pi and configured in WebIOPi.



Now you will see a virtual representation of the Raspberry Pi's GPIO pins. You can connect a relay between a ground pin and a GPIO pin and change the button (circled below) to the left or right of it, to out instead of in.



Now whenever you press one of the GPIO pins, it will send an amount of voltage to the relay

connected to it.

Web IO Pi Configuration

So far we have Web IO Pi setup, but to get it to open the garage door we needed to press one small button instead of one large button. Now we need to set it up so that we have one large button that opens it.

First we need to create a folder for the required files to do this:

```
sudo mkdir garage
cd garage
sudo mkdir html
sudo mkdir python
cd
```

Next we need to create the python file that helps the HTML file control the garage door opener.

```
sudo nano /home/pi/garage/python/script.py
```

Now copy the code below into the new Nano document.

```
import webiopi

GPIO = webiopi.GPIO

Garage = 17 # GPIO pin using BCM numbering

# setup function is automatically called at WebIOPi startup
def setup():
    # set the GPIO used by the light to output
    GPIO.setFunction(Garage, GPIO.OUT)
# loop function is repeatedly called by WebIOPi
def loop():

    # gives CPU some time before looping again
    webiopi.sleep(1)
```

Now ^X answer Y to the question of if you want to save and hit enter to save.

Next we need to create the HTML file that will control the garage.

```
sudo nano /home/pi/garage/html/index.html
```

Copy the code below into the new nano document.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.o$
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>Garage Control</title>
  <script type="text/javascript" src="/webiopi.js"></script>
  <script type="text/javascript">
    webiopi().ready(function() {
      // Create a "Light" labeled button for GPIO 17
      var button = webiopi().createGPIOButton(17, "Garage");

      // Append button to HTML element with ID="controls" using jQuery
      $("#controls").append(button);

      // Refresh GPIO buttons
      // pass true to refresh repeatedly of false to refresh once
      webiopi().refreshGPIO(true);
    });
  </script>
  <style type="text/css">
    button {
      display: block;
      margin: 5px 5px 5px 5px;
      width: 1280px;
      height: 720px;
      font-size: 100pt;
      font-weight: bold;
      color: white;
    }

    #gpio17.LOW {
      background-color: Black;
    }

    #gpio17.HIGH {
      background-color: Yellow;
    }
  </style>
</head>
<body>
  <div id="controls" align="center"></div>
</body>
</html>
```

Now ^X answer Y to the question of if you want to save and hit enter to save.

The file we just created won't help if we don't modify the Web IO Pi configuration file.

First edit the configuration file.

```
sudo nano /etc/webiopi/config
```

Now find the [SCRIPTS] section and add the following line:

```
garage = /home/pi/garage/python/script.py
```

Find the [HTML] line and add the following line:

```
doc-root = /home/pi/garage/html
```

Lastly find the [REST] line and add the following lines:

```
gpio-export = 17  
gpio-post-value = true  
gpio-post-function = false
```

To save the changes ^X answer Y to the question of if you want to save and hit enter to save.

To make the changes we need to reboot the Pi.

```
sudo reboot
```

Motion Setup

Now we are going to set up motion so you can check the garage door's status and monitor your garage using a webcam.

If you're using the Pi Camera instead of a generic webcam, check out this tutorial for setting up *motion* to work with it <http://rbnrpi.wordpress.com/project-list/setting-up-wireless-motion-detect-cam/> (<http://adafru.it/dcv>)

First, we need to give Raspbian the UVC camera support.

```
sudo apt-get install rpi-update  
sudo rpi-update
```

Now we need to install motion.

```
sudo apt-get install motion
```

and we need to configure it.

```
sudo nano /etc/motion/motion.conf
```

Once we are there we need to change a few things. First we need to change daemon to on so it starts when the Pi boots. To change this, change **daemon off**, to **daemon on**. Next thing we want to change is the resolution. You can leave it at 320 x 240 or change it to something higher - 320x240 is fine for most people. Next change **webcam localhost on** to **off**. This allows us to view the camera feed from another device instead of just the Raspberry Pi. Now ^X answer Y to the question of if you want to save and hit **enter**.

Lastly, we need to set up motion to start when the Pi boots.

```
sudo nano /etc/default/motion
```

In this file change **start_motion_daemon = no** to

```
start_motion_daemon=yes
```

Now we should reboot the Pi so it makes the changes.

```
sudo reboot
```

Now we have motion set up. Plug in a compatible webcam and it should stream video to your Pi's IP Address followed by **:8081**

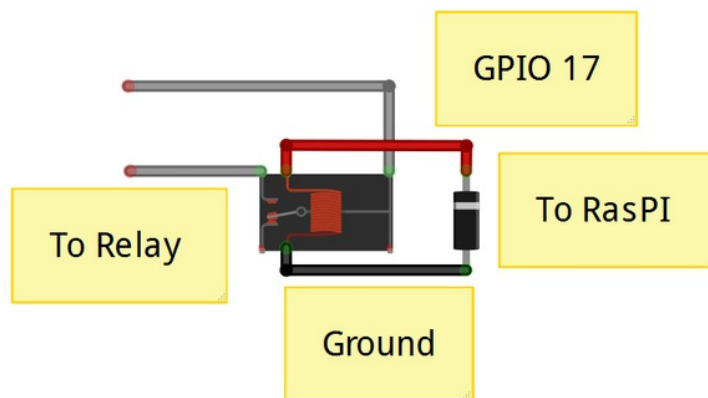
Hardware

Hardware

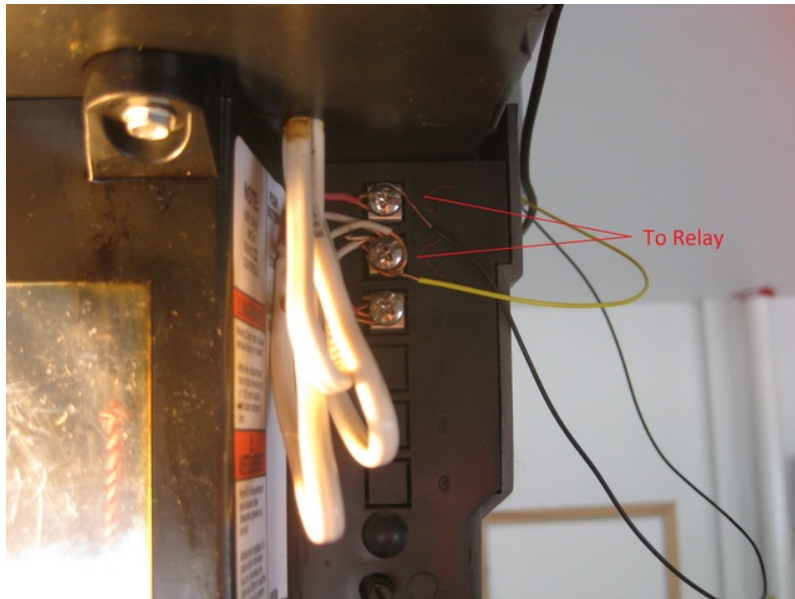
The hardware in this project consists of a relay that converts the voltage on the Raspberry Pi to a signal that opens and closes a garage door. To hookup the relay we need to solder the relay directly to a Perma-Proto PCB and solder all the wires onto the PCB. To make the PCB not short out we will cover the bottom with electrical tape. We will also place a 1N4001 Diode over the relay coil pins to avoid damaging the Pi GPIO pin. **This isn't optional if we don't do this it will eventually destroy the Pi!**

How It Connects to the Relay

Find a pinout for the relay. Usually they are on the datasheet for the relay. Below is an example of a relay and how to hook it up. The Raspberry Pi will connect to the coil with a diode between the control (red wire) and ground (black wire) pins. The garage door will connect to the normally open pin and relay switch pin (two gray wires).

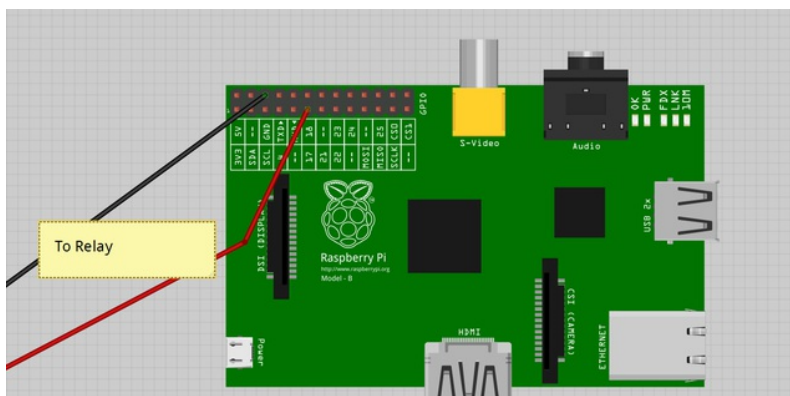


Below is an example of where the the relay connects to the garage door. Use a tool such as a screwdriver to press between the screws on the garage door opener. When you press the two that open the garage door, screw the wires attached to the relay onto it.



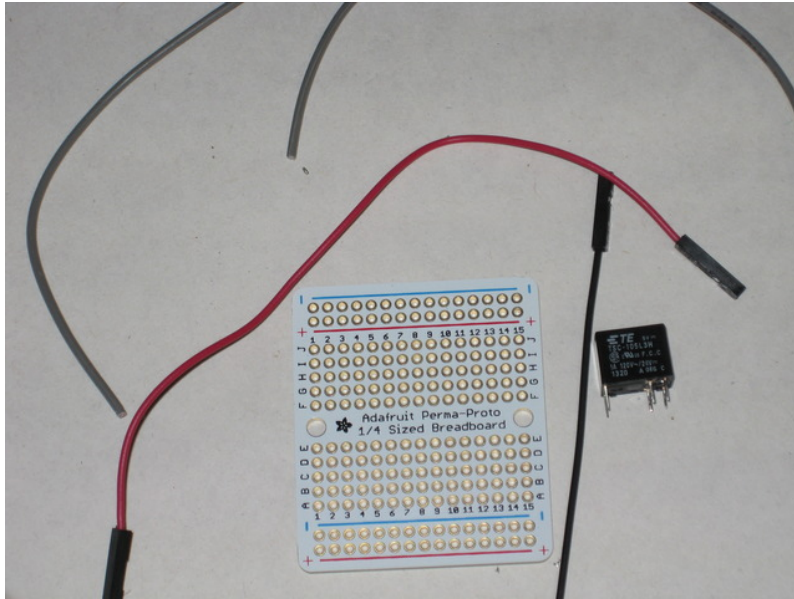
How it Connects to the Raspberry Pi

The coil on the relay will connect to Ground and GPIO 17 on the Raspberry Pi using the female jumper wires.



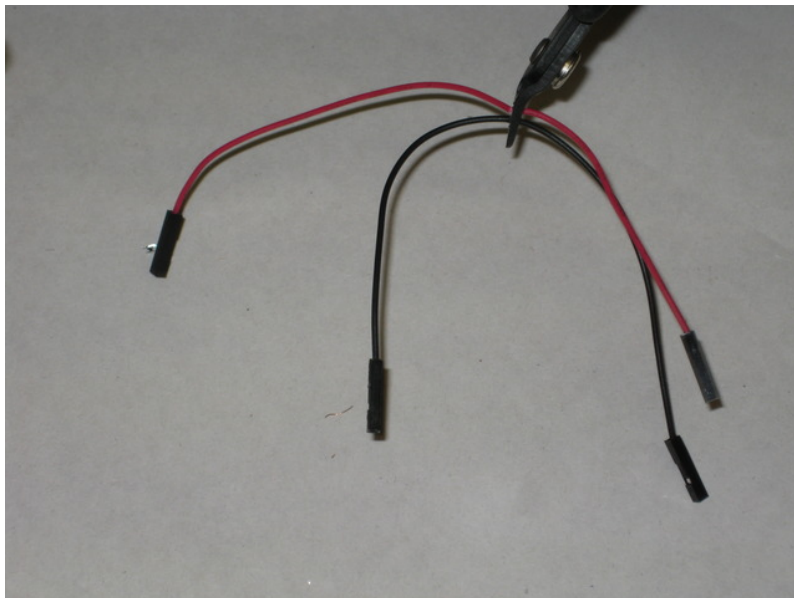
Soldering the Relay Board Time!

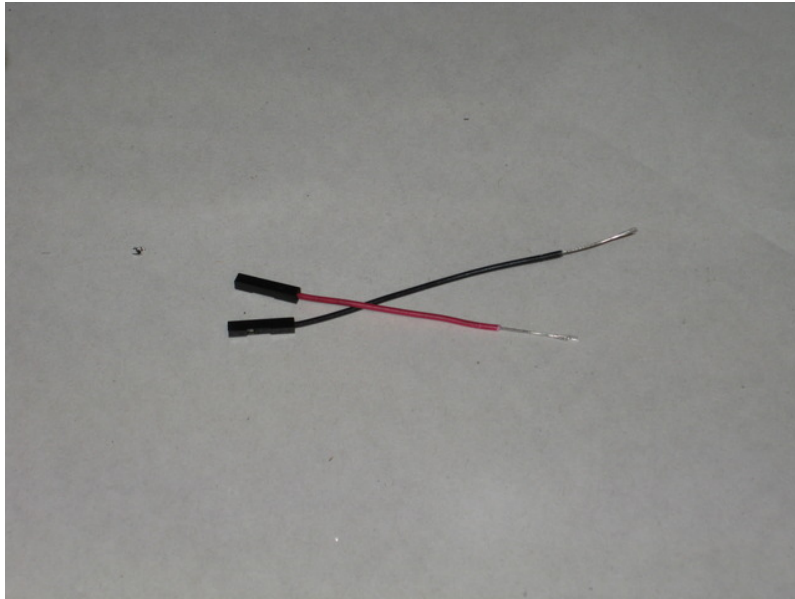
Below are the parts you will need:



Step One:

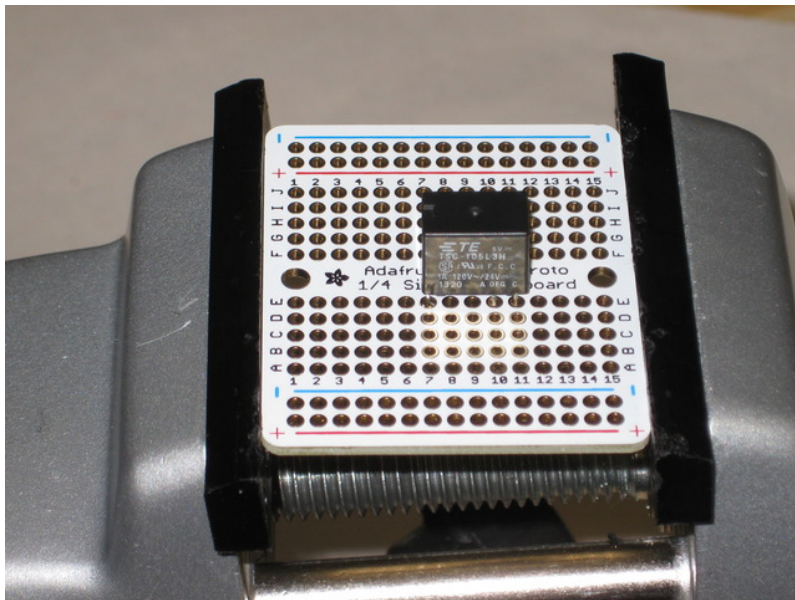
Cut and strip the two female jumper wires in half.

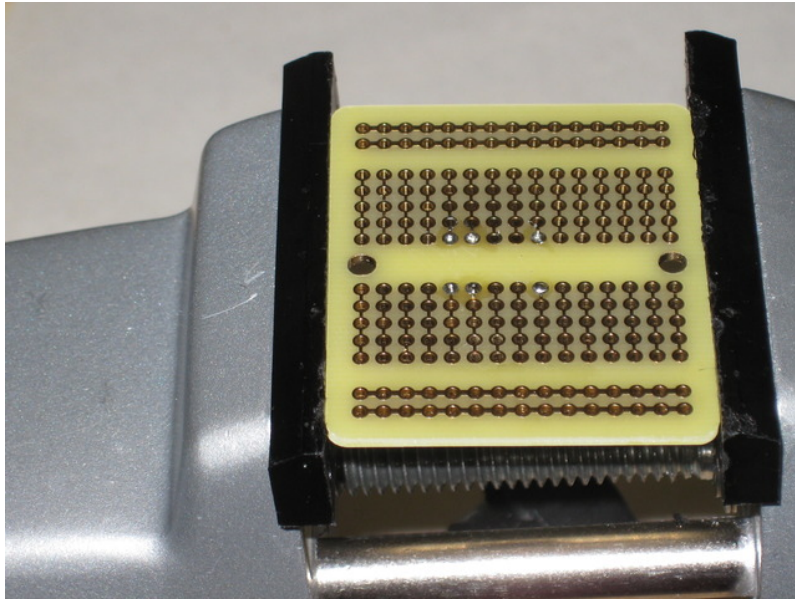




Step Two:

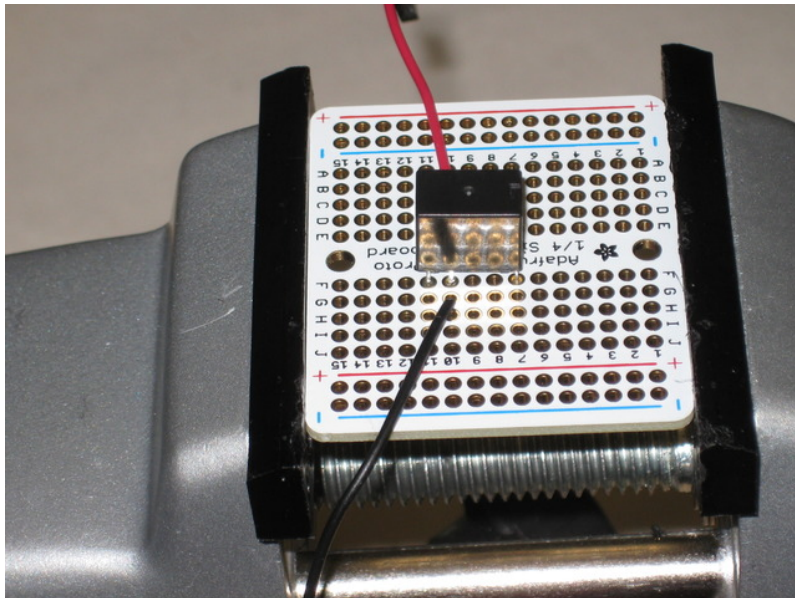
Solder the relay to the Perma-Proto.

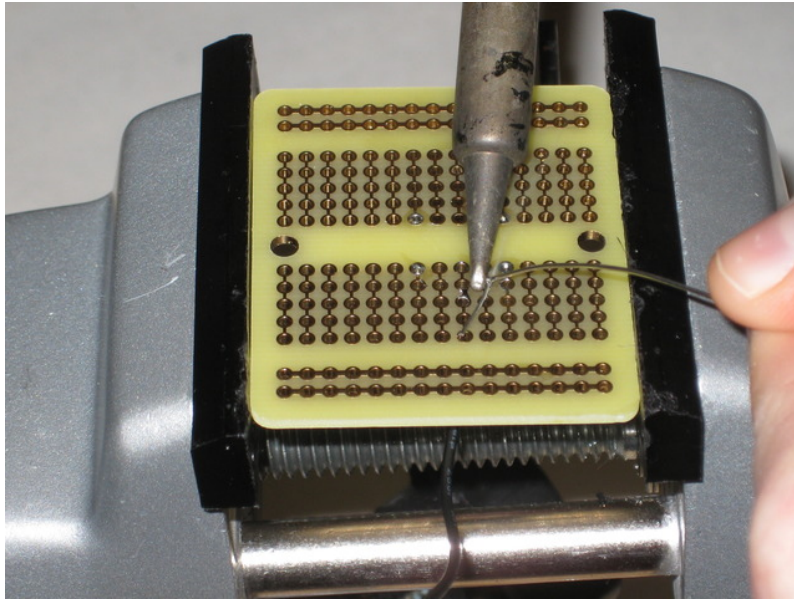




Step Three:

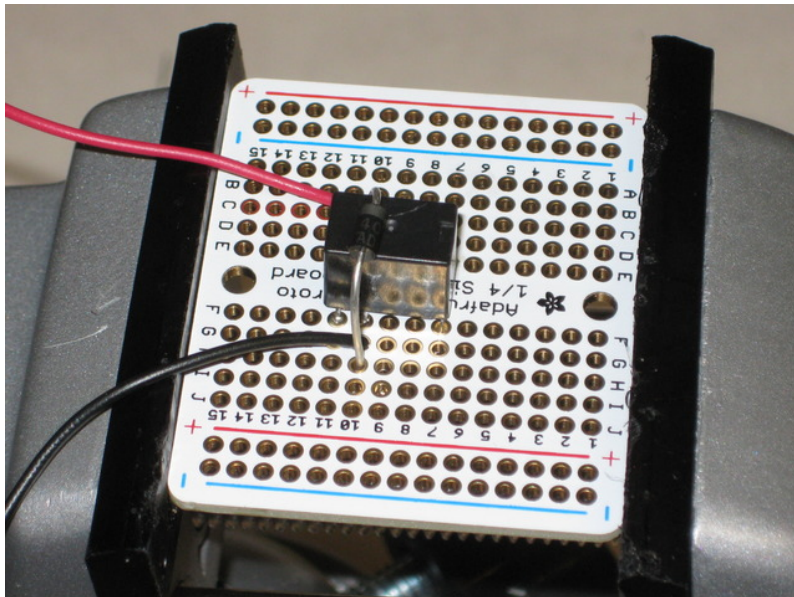
Now we need to solder the black female jumper wire to the coil on the relay and the red jumper wire to the coil on the relay. After that trim the excess wire on the bottom of the board.

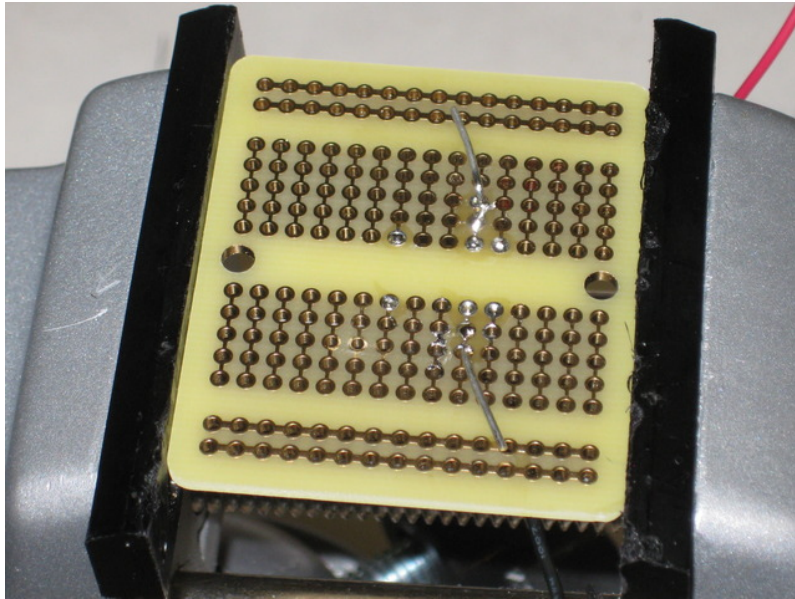




Step Four

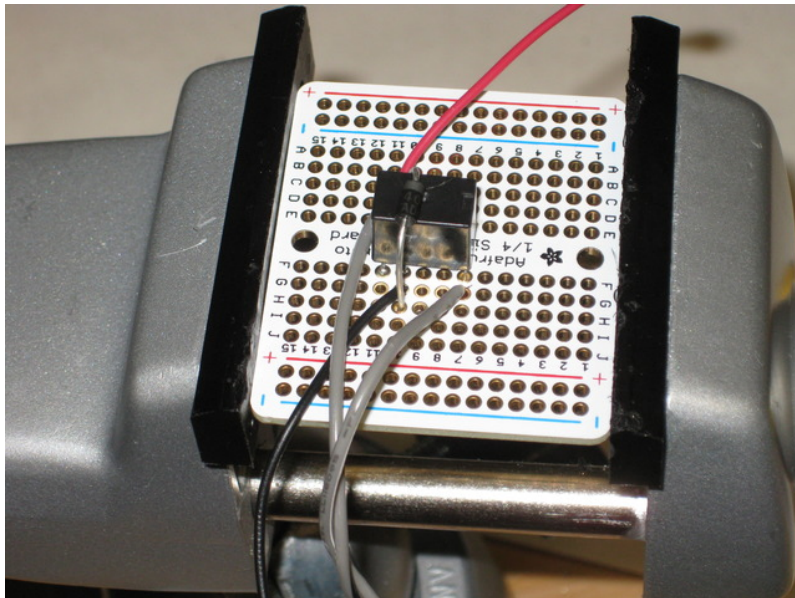
To prevent the Raspberry Pi from crashing, we need to place a 1N4001 diode between the ground and control pins on the relay. The silver band on the diode needs to be connected to the control wire not ground.

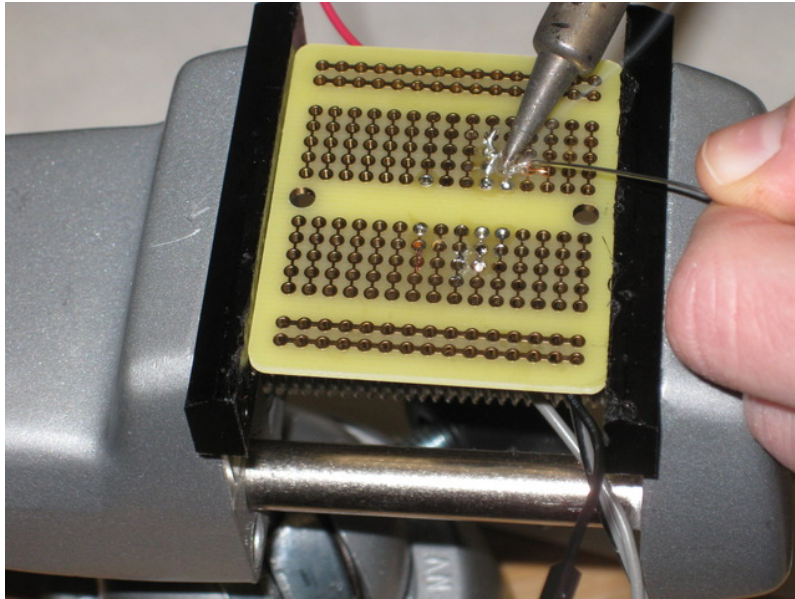




Step Five

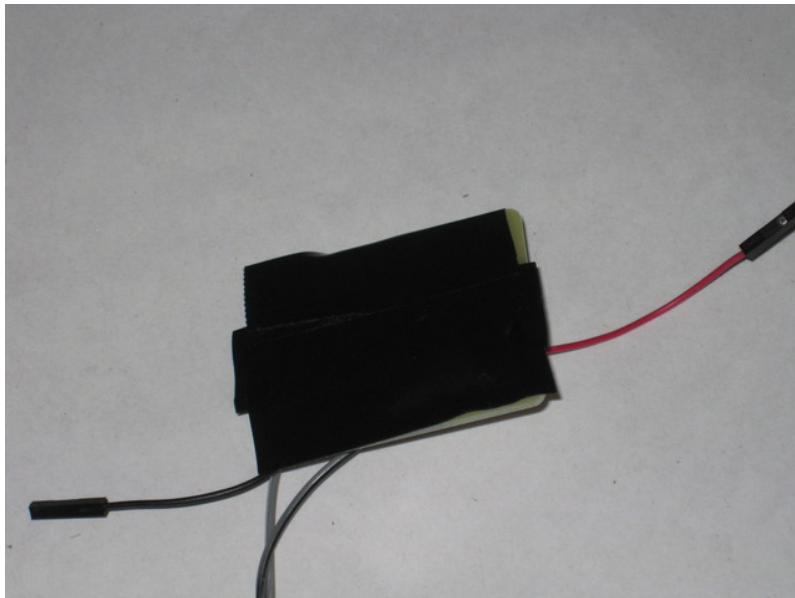
Solder the wires that connect to the garage door to the normally open pin and ground on the relay. Again trim the excess leads on the board.





Step Six

Lastly cover the bottom of the board we have just made with electrical tape. Once that is done you can connect it to the Raspberry Pi and Garage Door Opener.



Network Configuration

If we skip this step, then we will only be able to open the garage door from our WiFi network *inside the house*. What we need to do now is to port forward the Raspberry Pi's port 8000 and 8081 so that when we access our public IP address we can open and close the garage door from afar.

Step One:

Find out your router's configuration IP Address and visit it. For example my Belkin is 192.168.2.1.

Step Two:

Find the page on your router's web site that allows you to open up ports on the firewall. Some routers call it port forwarding and virtual servers.

Step Three:

We need to open up 8000 and 8081 for the IP address of the Raspberry Pi. Lots of information on port forwarding is available on the internet, and it depends a little on how your router is set up. If you don't have the exact same router as me, just google for your router name/model and "port forwarding" - chances are someone's done the work for you to figure out how! Below is how it looked for me.

Firewall > Virtual Servers

This function will allow you to route external (Internet) calls for services such as a web server (port 80), FTP server (Port 21), or other applications through your Router to your internal network.
[More Info](#)

Clear Changes Apply Changes

Add Active Worlds Add

Clear entry 1 Clear

	Enable	Description	Inbound port	Type	Private IP address	Private port
1.	<input type="checkbox"/>		-	TCP	192.168.2.	-
2.	<input checked="" type="checkbox"/>		8081 - 8081	TCP	192.168.2.16	8081 - 8081
3.	<input checked="" type="checkbox"/>		8000 - 8000	TCP	192.168.2.16	8000 - 8000
4.	<input type="checkbox"/>		-	TCP	192.168.2.	-
5.	<input type="checkbox"/>		-	TCP	192.168.2.	-
6.	<input type="checkbox"/>		-	TCP	192.168.2.	-
7.	<input type="checkbox"/>		-	TCP	192.168.2.	-
8.	<input type="checkbox"/>		-	TCP	192.168.2.	-

Step Four:

Lastly, find out what your public IP address is. This can be found with just a simple Google search of "What is my IP address?" or visiting **ipchicken.com** Once you find out the IP

address, you can type that followed by :8000 and you can control the garage door anywhere. If you do the IP address followed by 8081 you will view the garage camera. (I have found that you can only view it in Firefox.)

How to Control
