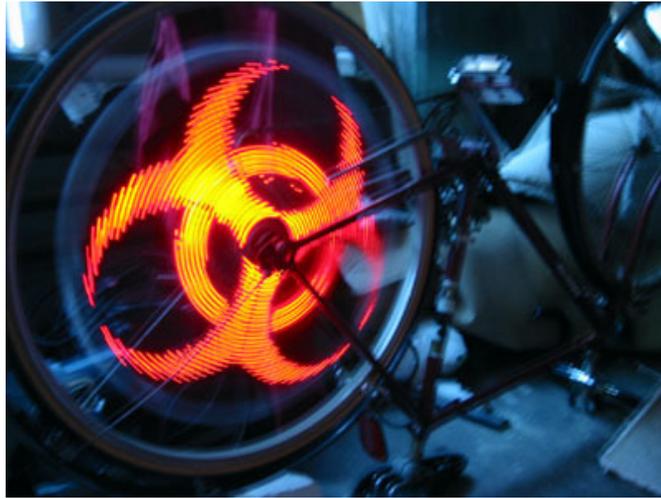


SpokePOV

Created by lady ada



Last updated on 2014-12-27 11:45:15 AM EST

Guide Contents

Guide Contents	2
Overview	4
Persistence of Vision for your Bike!	4
Introduction	4
Specification & Features	4
F.A.Q.	6
Frequently Asked Questions	6
Preparation	8
Prep	8
Tools	8
SpokePOV Assembly Prep	12
SpokePOV	15
Parts for the SpokePOV Kit	15
SpokePOV Kit Assembly Instructions	16
Parallel Port Dongle	34
Make it! - Dongle	34
SpokePOV Parallel Port Dongle Parts	34
SpokePOV Dongle Assembly Instructions	34
Serial Dongle	40
Make it! - Serial dongle	40
Serial Port SpokePOV Dongle Parts	40
Assembling the Serial Dongle	40
USB Dongle	46
Make it! - Serial dongle	46
USB Port SpokePOV Dongle Instructions	46
Using the USB dongle	46
Customize	48
Use it! - Upload images	48
Introduction	48

Configuring	48
Testing the connection	52
Connecting	54
Install	66
User Manual	66
Installing your SpokePOV	66
Using the SpokePOV	74
Design	76
Hardware design	76
Intro	76
Schematic	76
Microcontroller	77
Sensor	78
Button	78
EEPROM	78
Batteries	78
LEDs	78
Programming header	79
Programming dongle	79
Gallery	80
Here are some photos from other people!	80
Download	81
Example images	81
USB Driver (windows only!)	81
C++ Software	81
wxPython Software	81
Firmware	82
Hardware	82

Overview

Persistence of Vision for your Bike!



Introduction

Spoke POV is an easy-to-make electronic kit toy that turns your bicycle wheel into a customized display! The project includes a free schematic design, open software for uploading and editing stored bitmap images, and a high-quality kit with all the parts necessary to build your own.

[Kits & PCBs available for purchase now! \(http://adafru.it/dhC\)](http://adafru.it/dhC)

Specification & Features

- Can be used with road, mountain or BMX wheels!
- 30 LEDs on each side (22 for BMX) x 256 radial pixels

- Runs on 2-3 AA batteries for 10 hours or more, assuming 3000mAh alkalines and 50% image coverage.
- Can run on rechargeable NiMH AA's
- Comes with high-brightness red, yellow, green or blue LEDs
- Automatically shuts off after 3 minutes
- Software runs on any Mac, Windows or Linux computer with USB port
- Design is all through-hole parts, perfect for a first time kit builder.
- With one spoke, total persistence at 15mph. Two spokes, 10mph. Three spokes, 7mph. (Assuming a mountain bike wheel). Looks cool even if not completely persistent.
- Great for safe riding at night, provides excellent side visibility.
- Playa-tested!

F.A.Q.

Frequently Asked Questions

How does this thing work?

A SpokePOV is a 'stick' of LEDs with a magnetic (hall effect) sensor on the outer end. When a magnet passes by, the microcontroller takes note. By counting how long it takes between magnet passes, SpokePOV can tell how fast it's rotating and quickly blink the LEDs to present an image. Therefore, no matter how fast you are riding, the image will show up correctly!

Why is there a random LED lit on either side of the SpokePOV?

One LED is lit on both sides to indicate that the SpokePOV is on. The LED itself may vary, as it is also used to debug the SpokePOV.

What colors are available? Can I put my own colors in?

Right now kits are available in red, yellow, green and blue. You can use your own LEDs, just buy 60 5mm 2000mcd (or higher) LEDs and possibly vary the resistor packs to fit your LEDs and number of batteries - this may take some experimentation.

Why don't you have full color SpokePOVs?

While full-color POV is [really cool looking \(http://adafru.it/dhD\)](http://adafru.it/dhD), it's also fabulously expensive: to make a 60-LED SpokePOV full color it would cost 5x as much!

Will SpokePOV fit my wheel?

SpokePOV can fit either BMX (20") wheels or mountain/roadoad bike (700c/37") wheels. The short version is 6.5" (16.5cm) long, the longer version is 9" (23cm).

If you are using BMX wheels, [simply cut the PCB \(http://adafru.it/dhE\)](http://adafru.it/dhE) before soldering in parts and then don't solder in the extra LEDs or latches.

How fast do I have to pedal?

On a mountainbike wheel, I have found that with 1 SpokePOV you should bike 15mph/24kph. With 2 SpokePOV, bike 10mph/16kph. With 3 SpokePOVs, 7mph/11kph. This will give a nice persistant image. The faster the better, of course.

Larger wheels you might have to bike a little faster. Smaller wheels, you can go slower.

SpokePOV has been tested to work at 30mph (but can probably go even faster).

Can I use SpokePOV on a car or motorbike?

SpokePOV is not for motorbike or car use. It is specifically designed for bicycles and bicycle speeds. You could easily hurt someone very badly if the spokepov is badly attached to a motorcycle and then flies off at very high speeds.

Do you have pre-assembled SpokePOVs?

Sadly, no. I only sell kits: fun & easy to make and you'll learn a lot when you build them!
However, you can buy a similar product called 'Hokey Spokes' (<http://adafru.it/dhF>) - same idea but you can't customize the images.

Do you have any high-rez photos?

Try:

- http://farm1.static.flickr.com/46/109571161_c9a4fb7189_o.jpg (<http://adafru.it/dhG>)
- http://farm1.static.flickr.com/159/392464250_2836156f09_o.jpg (<http://adafru.it/dhH>)

Preparation

Prep

Learn how to solder with tons of tutorials! (<http://adafru.it/aTk>)

Don't forget to learn how to use your multimeter too! (<http://adafru.it/aZZ>)

Tools

There are a few tools that are required for assembly. None of these tools are included. If you don't have them, now would be a good time to borrow or purchase them. They are very very handy whenever assembling/fixing/modifying electronic devices! I provide links to buy them, but of course, you should get them wherever is most convenient/inexpensive. Many of these parts are available in a place like Radio Shack or other (higher quality) DIY electronics stores.



Soldering iron

Any entry level 'all-in-one' soldering iron that you might find at your local hardware store should work. As with most things in life, you get what you pay for.

Upgrading to a higher end soldering iron setup, like the [Hakko FX-888 that we stock in our store \(http://adafru.it/180\)](http://adafru.it/180), will make soldering fun and easy.

Do not use a "ColdHeat" soldering iron! They are not suitable for delicate electronics work and can damage the kit ([see here \(http://adafru.it/aOo\)](http://adafru.it/aOo)).



[Click here to buy our entry level adjustable 30W 110V soldering iron \(http://adafru.it/180\)](http://adafru.it/180).

[Click here to upgrade to a Genuine Hakko FX-888 adjustable temperature soldering iron. \(http://adafru.it/303\)](http://adafru.it/303)



Solder

You will want rosin core, 60/40 solder. Good solder is a good thing. Bad solder leads to bridging and cold solder joints which can be tough to find.

[Click here to buy a spool of leaded solder \(recommended for beginners\)](http://adafru.it/145) (<http://adafru.it/145>).

[Click here to buy a spool of lead-free solder](http://adafru.it/734) (<http://adafru.it/734>).



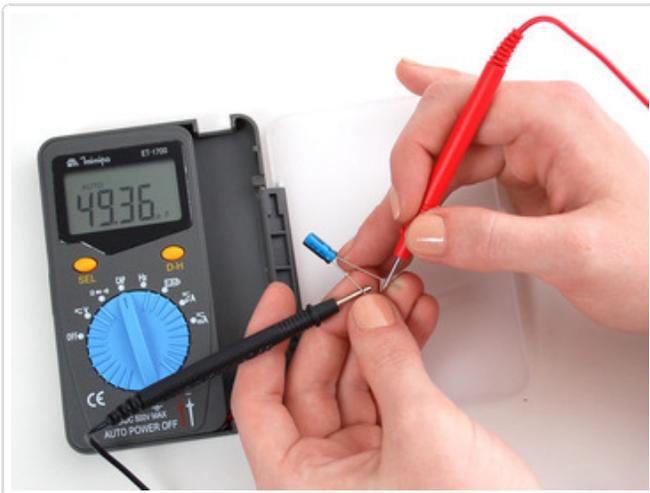
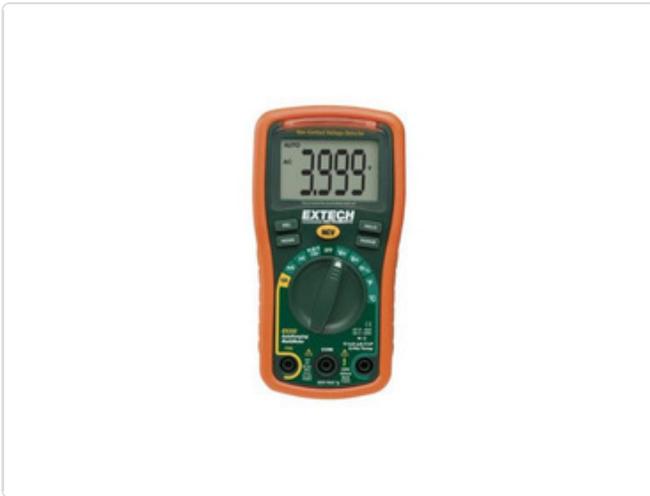
Multimeter

You will need a good quality basic multimeter that can measure voltage and continuity.

[Click here to buy a basic multimeter.](http://adafru.it/71) (<http://adafru.it/71>)

[Click here to buy a top of the line multimeter.](http://adafru.it/308) (<http://adafru.it/308>)

[Click here to buy a pocket multimeter.](http://adafru.it/850) (<http://adafru.it/850>)





Flush Diagonal Cutters

You will need flush diagonal cutters to trim the wires and leads off of components once you have soldered them in place.

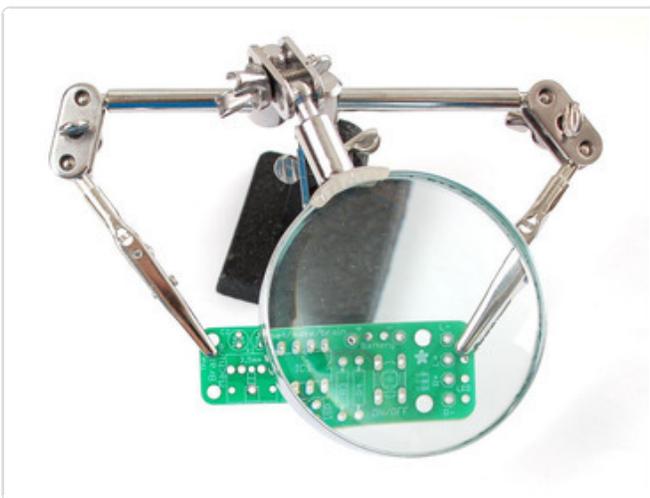
[Click here to buy our favorite cutters \(http://adafru.it/152\).](http://adafru.it/152)



Solder Sucker

Strangely enough, that's the technical term for this desoldering vacuum tool. Useful in cleaning up mistakes, every electrical engineer has one of these on their desk.

[Click here to buy a one \(http://adafru.it/148\).](http://adafru.it/148)



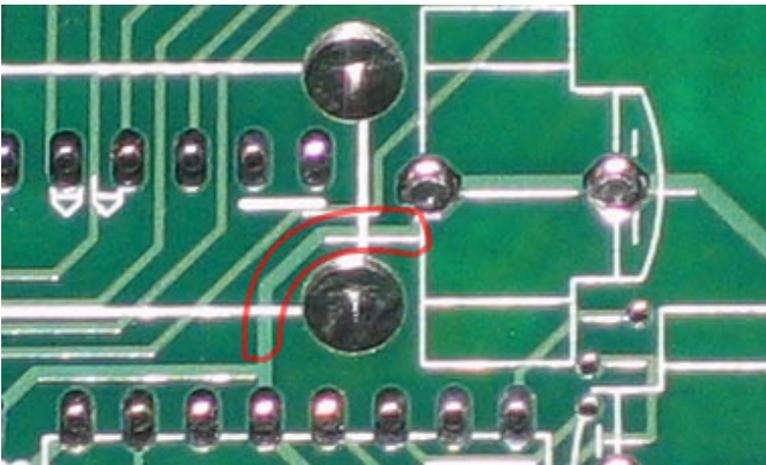
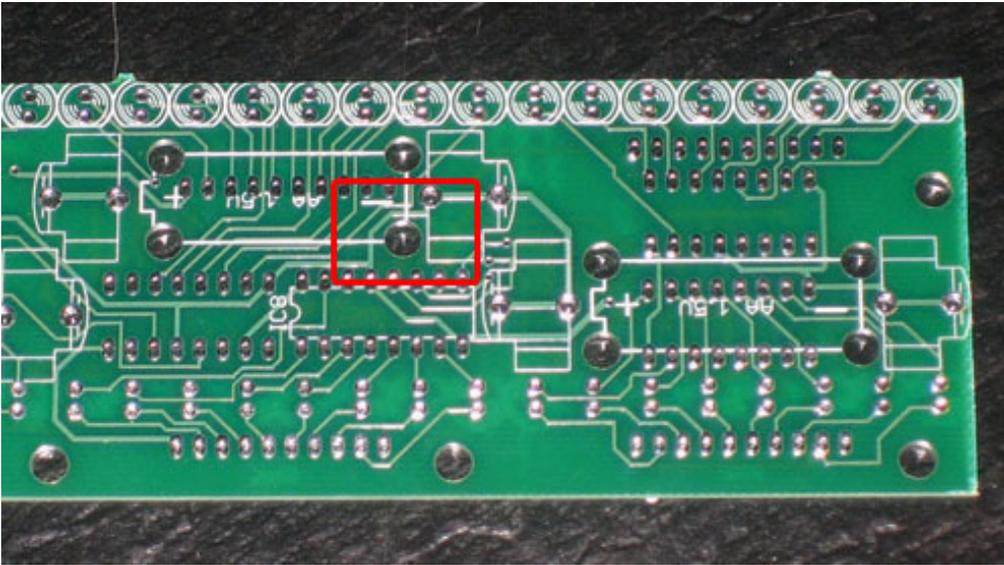
Helping Third Hand With Magnifier

Not absolutely necessary but will make things go much much faster, and it will make soldering much easier.

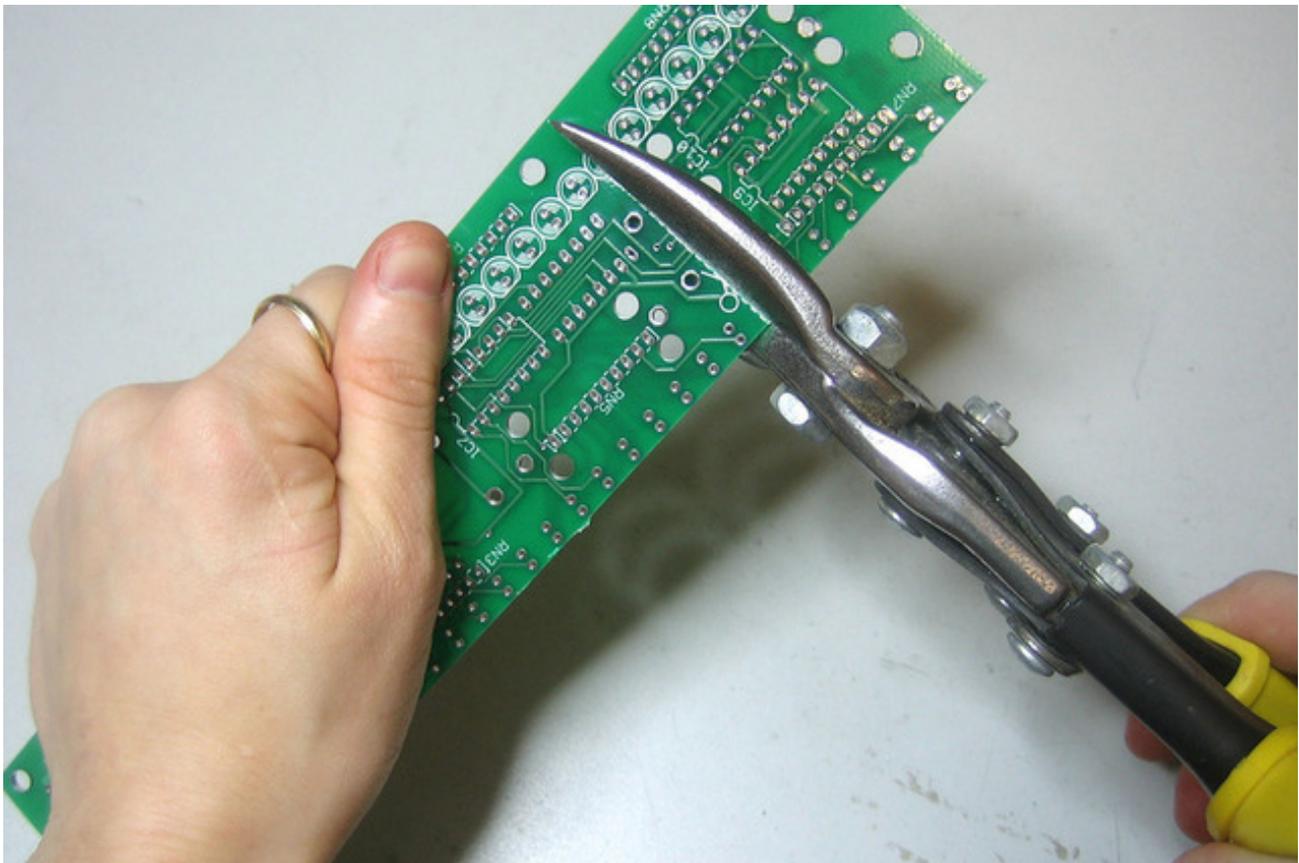
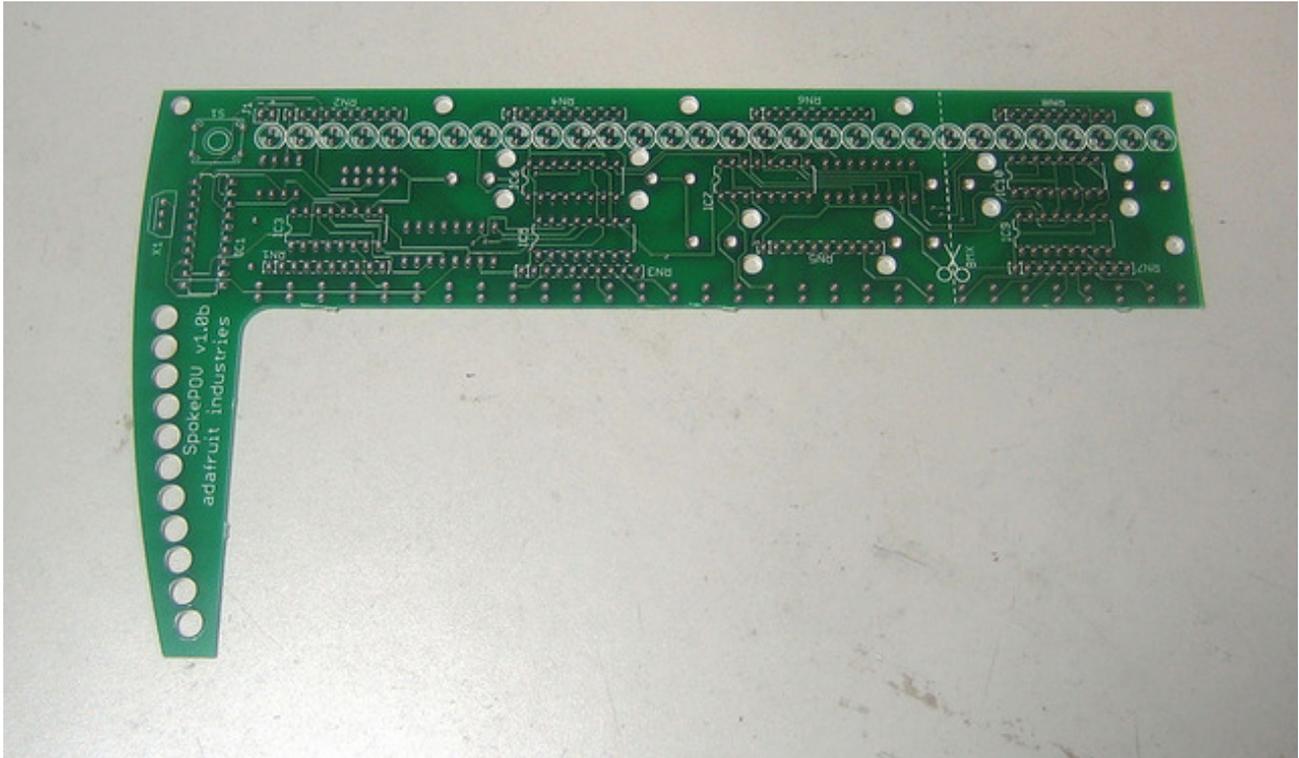
[Pick one up here \(http://adafru.it/291\).](http://adafru.it/291)

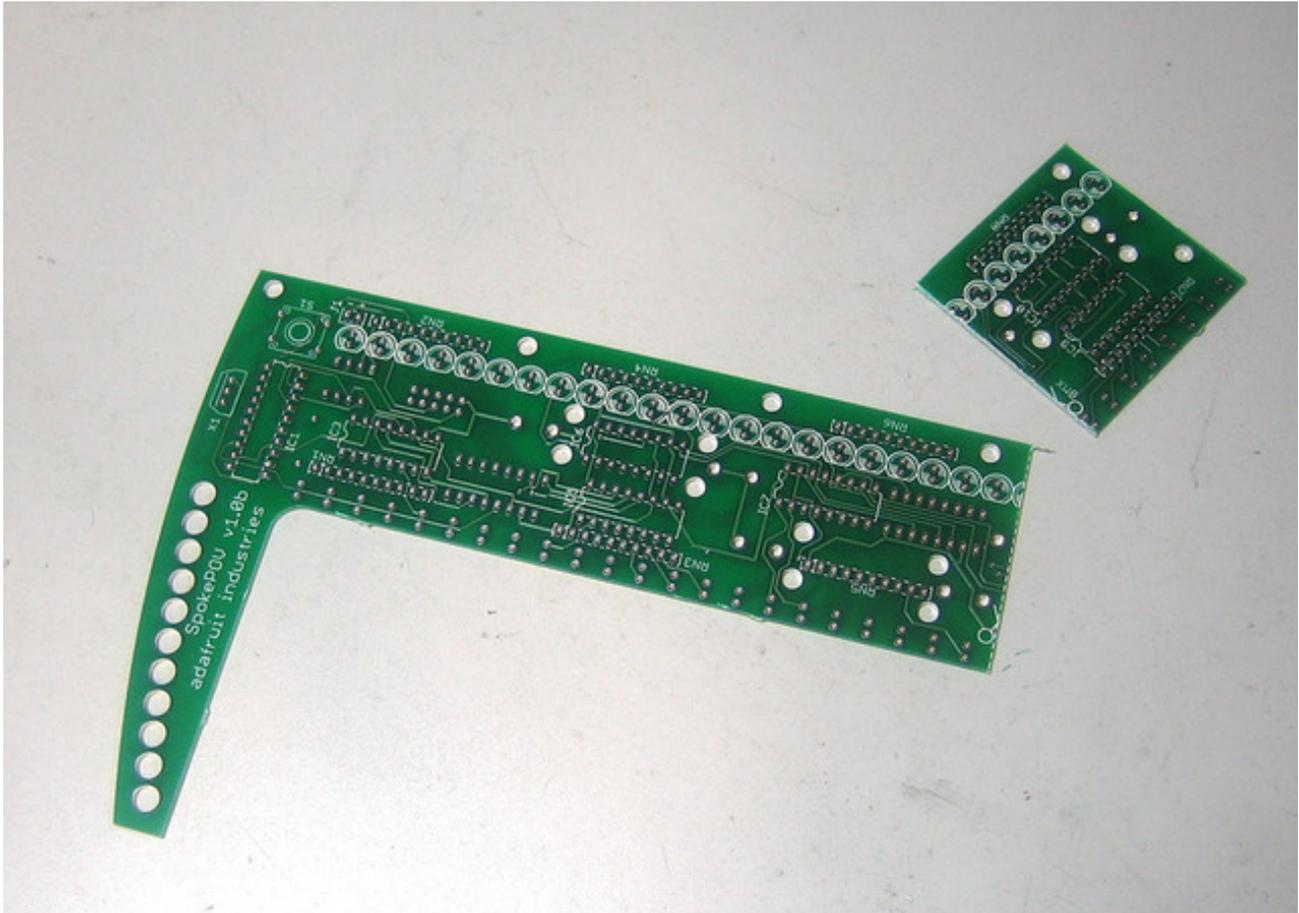
SpokePOV Assembly Prep

Before you start, if you have blue, green, white or purple LEDs you will have to add a third battery. To do that, purchase 2 more battery clips from Mouser (or if you have a kit they are included). Before you solder them in, you **MUST** cut the trace identified below. You can cut a trace using a razor blade, knife, sharp screwdriver, etc. The point is to make sure the printed wire is broken. If you don't cut it properly you will notice the 3rd battery gets **REALLY** hot when you put it in! (I thought I'd be all smart and put a note on the PCB about which trace to cut but on some early SpokePOVs, I put it on the wrong side of the board. If you hold it up to a bright light you'll see the trace, or use this handy guide.)



If you are planning to use the kit with BMX/Unicycle/kid's wheels, you'll need to cut down the PCB yourself. You can do this with tin snips (shown), a hacksaw, bandsaw, shear, etc.





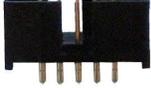
When assembling it, just solder all the parts you can. There will be 16 LEDs, 2 ICs and 2 resistor packs left over.

SpokePOV

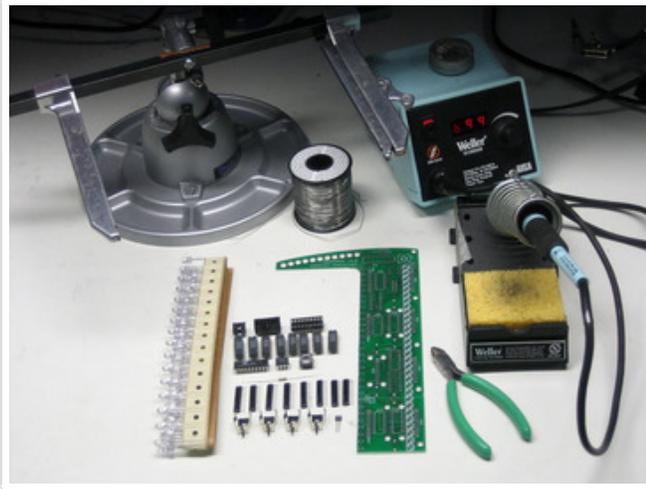
Parts for the SpokePOV Kit

Check to make sure your kit comes with the following parts. Sometimes we make mistakes so double check everything and email support@adafruit.com if you need replacements!

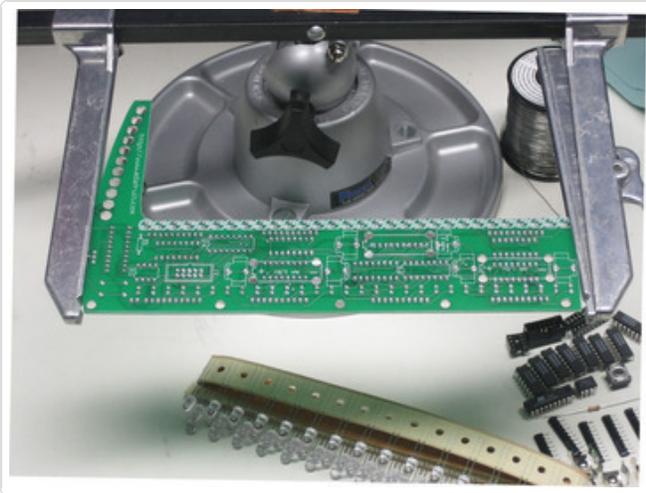
Picture	Name	Description	Part #	Distributor	Qty.
	IC1	Microcontroller Pre-programmed when purchased as part of a kit.	ATtiny2313V-10PU	Digikey Mouser or Adafruit (programmed)	1
	IC1*	20 Pin Socket	Generic	Mouser Digikey	1
	IC2	4 Kbyte EEPROM	25LC320, CAT25320 or AT25320 (4K SPI EEPROM)	Digikey Mouser	1
	IC2*	8 Pin Socket	Generic	Mouser Digikey	1
	IC3- IC10	Serial to Parallel latch	74HC595	Mouser Digikey	8
	X1	Hall effect sensor	DN6852 or US5881	Digikey	1
	S1	Waterproof switch	SKQBAKA010 or SKQBALA010	Mouser	1
					

	J2	Programming Header	30310-6002HB	Mouser Digikey	1
		AA battery clips	Keystone 92	Mouser Digikey	4 (red, yellow) or 6 (green, blue)
	C1	0.1uF ceramic capacitor	Generic		1
	C2	47uF to 100uF axial capacitor	Generic	Digikey	1
	R1	1/4W 5% 10K resistor (brown black orange)	Generic		1
	RN1- RN8	10-pin bussed 9 resistor network	33-39 ohm for yellow LEDs 33-39 ohm for red LEDs 56 or 68 ohm for blue LEDs 56 or 68 ohm for green LEDs	27 ohm - (Digikey , Mouser) 39 ohm - (Digikey , Mouser) 56 ohm - (Digikey , Mouser) 68 ohm - (Digikey , Mouser)	8
	LED1- LED60	LED	High Brightness 5mm LED	LEDshoppe ABCtronics	60
	PCB	Silkscreened PCB	Custom	Adafruit	1

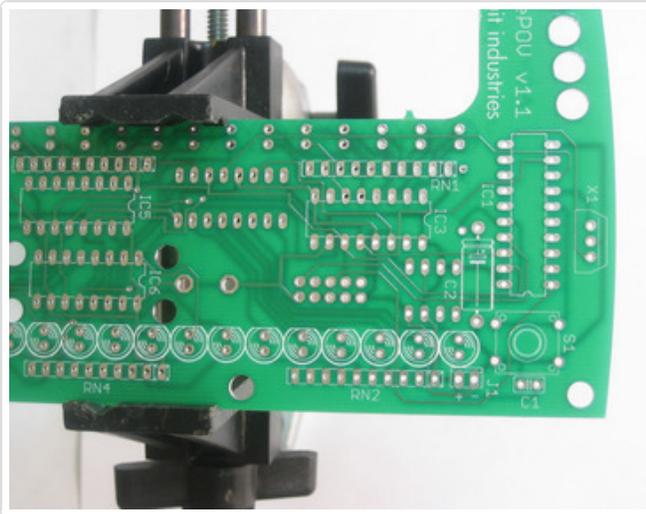
SpokePOV Kit Assembly Instructions



Get your bench set up for soldering.



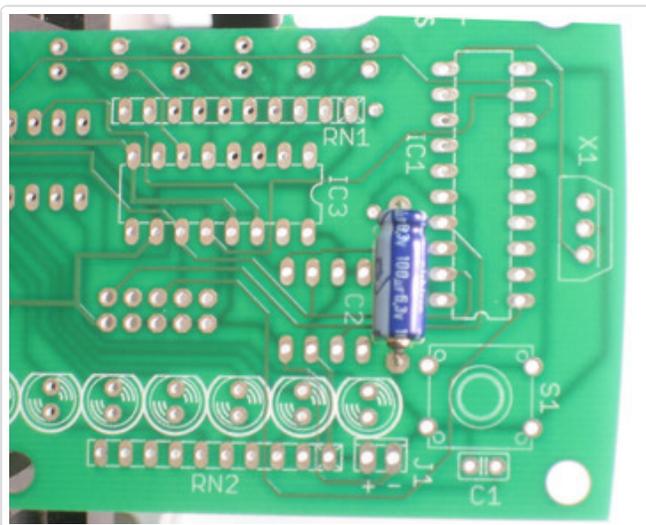
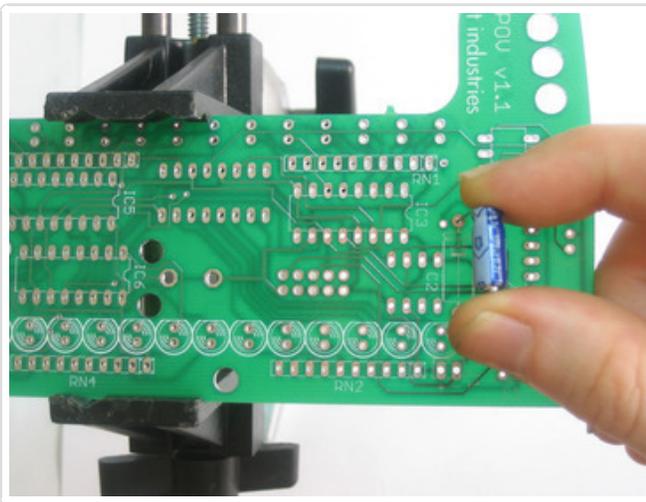
Place the PCB in a good board holder.

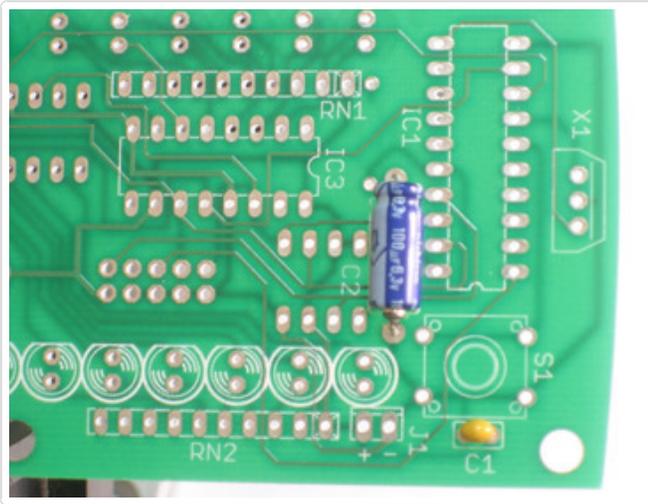


In v1.1 two components have been added, **C1** and **C2**. These parts add power supply stability!

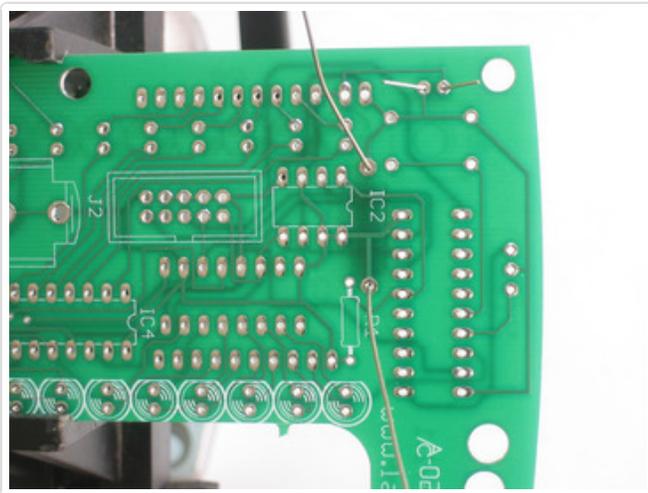
Start by inserting **C2** by bending the legs and placing it in the circuit board. **C2** is a *polarized capacitor*. This means it should be soldered in a certain way and if you put it in backwards it can be damaged or even explode!

If you look carefully you will see a bump on one end of the capacitor, and on the circuit board the printed image also has a bump. Check the photo if you are not sure.

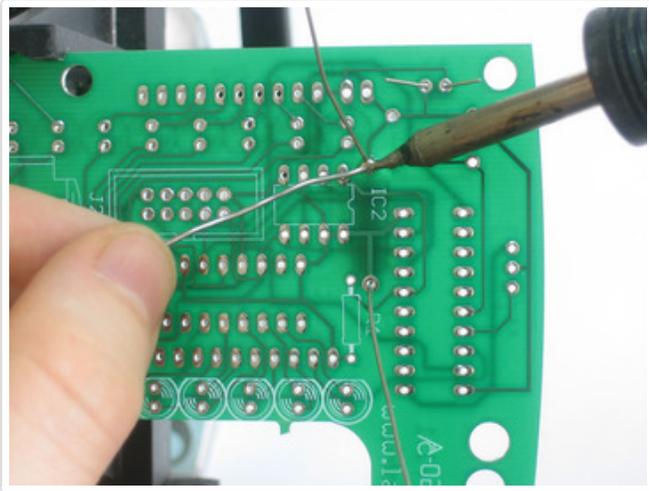




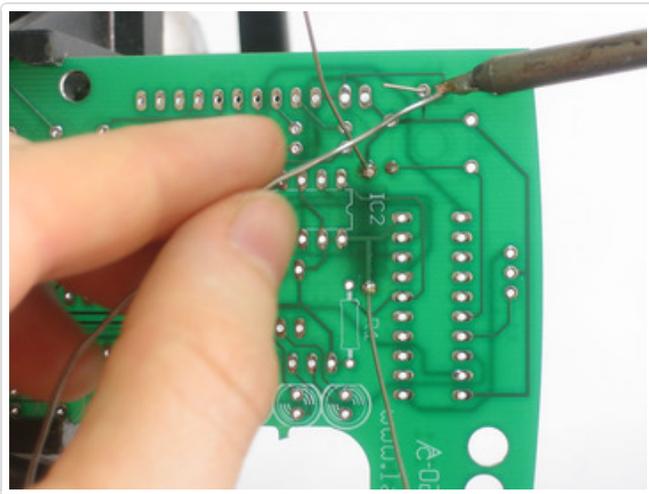
Next, insert **C1**, which is a non-polarized capacitor; it can go in either way.

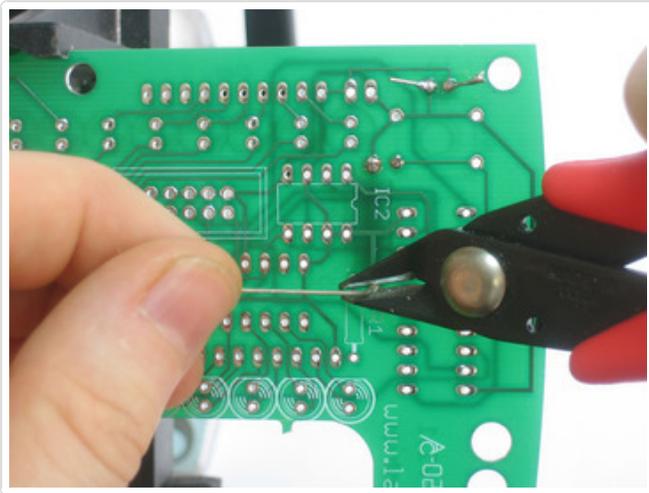


Bend the wire legs of the capacitors and turn the PCB over in the vice. Now you can solder the parts in.

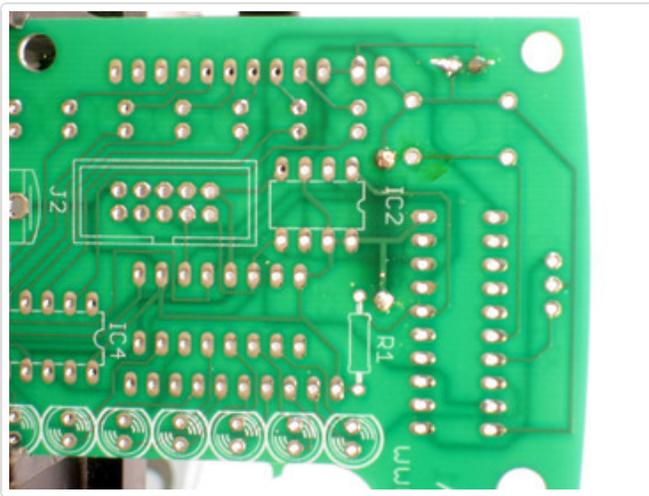


Use your trusty soldering iron to solder each of the 4 wires to the round pads.

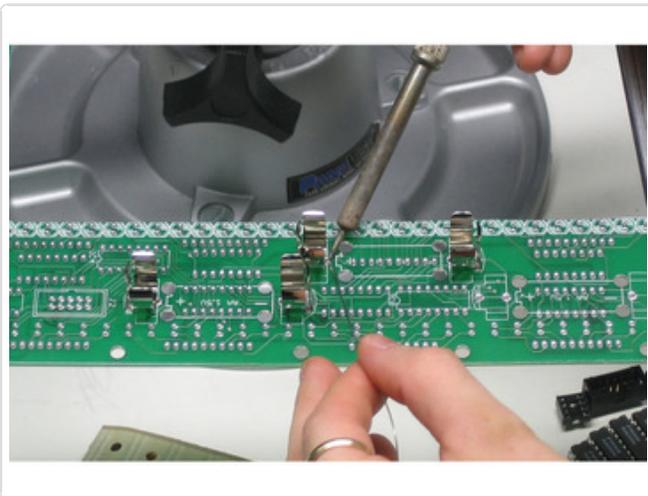
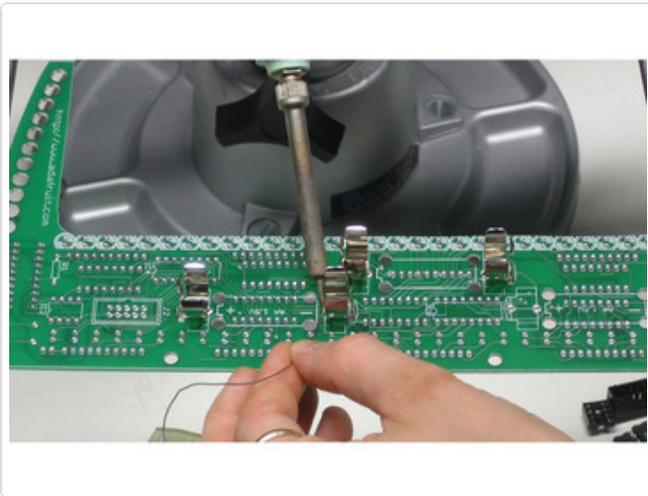
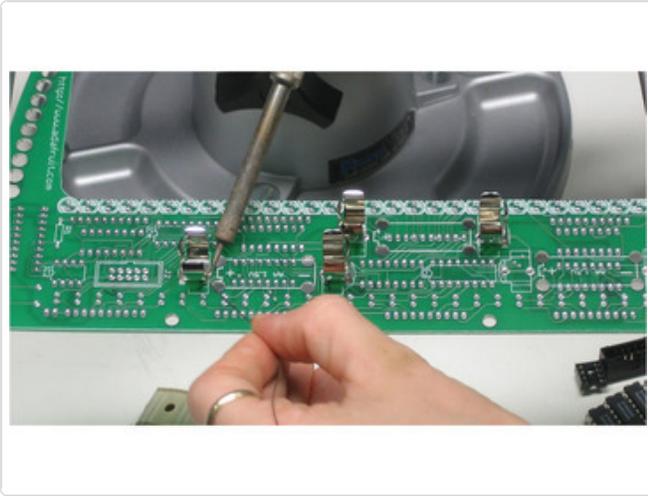


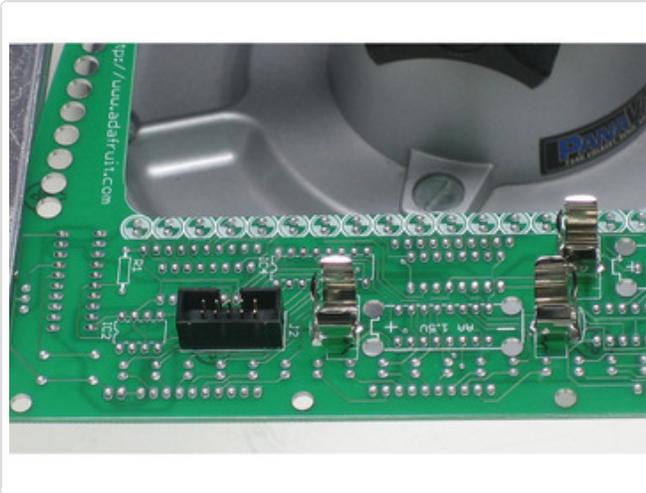


Use diagonal cutters to clip the excess wire, cut close to the end of the solder point.

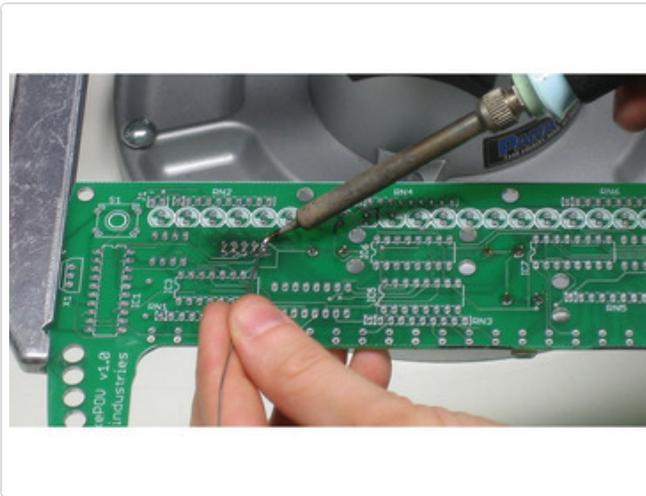


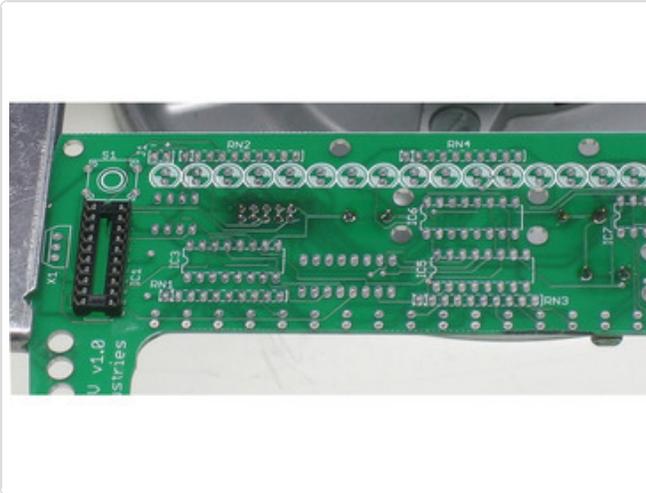
Place the 4 battery clips in the top two battery positions, as shown. Tack them in place with solder so that when you turn the board over they won't fall out!



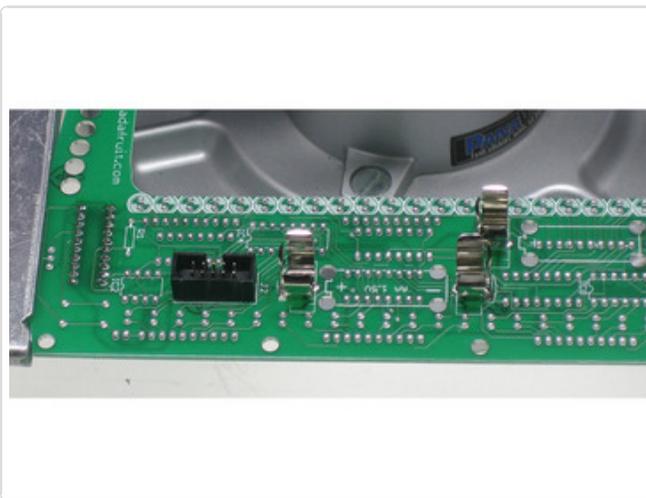
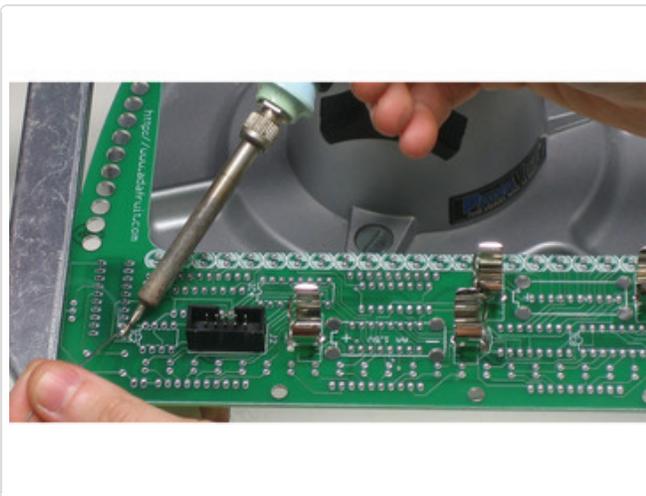


Put in the programming socket, make sure to align the notch with the picture on the PCB. Now turn the PCB over and hold the socket in place while you solder it in.

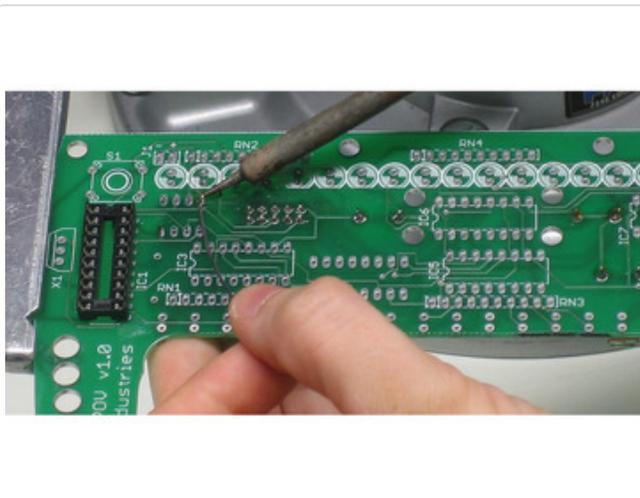
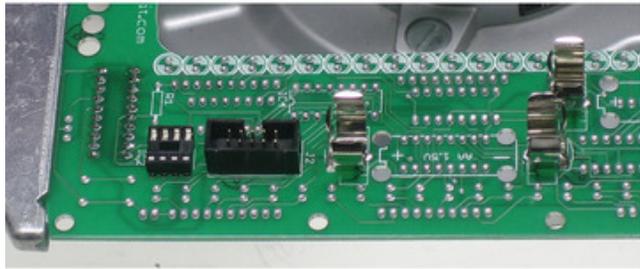


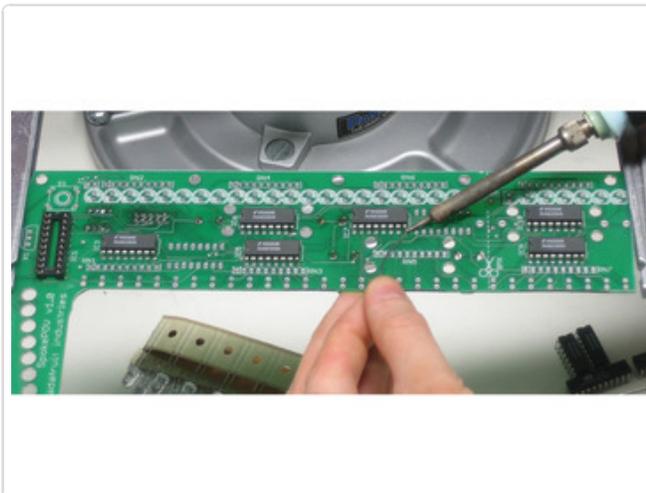


Place the large socket as shown. There's a notch in the socket which should line up with the notch in the image of the socket. The notch will help you align the microcontroller chip in properly. Turn the board over and hold the socket in with a finger, tack two corners pins to keep it in place then solder the rest of the pins.



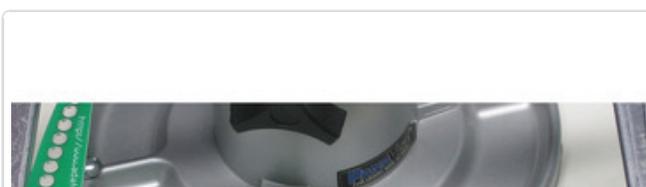
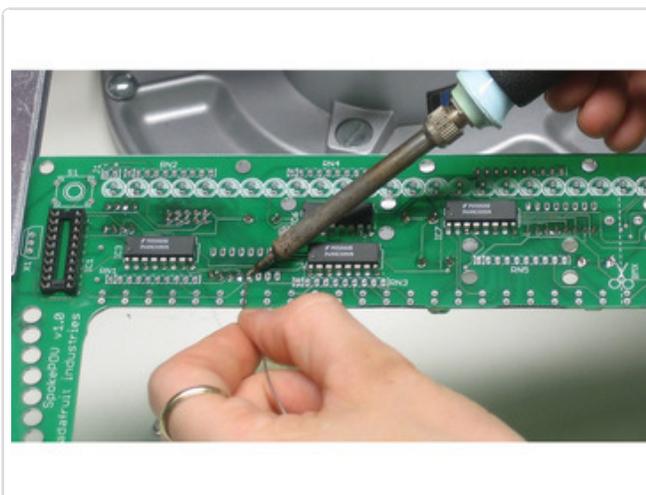
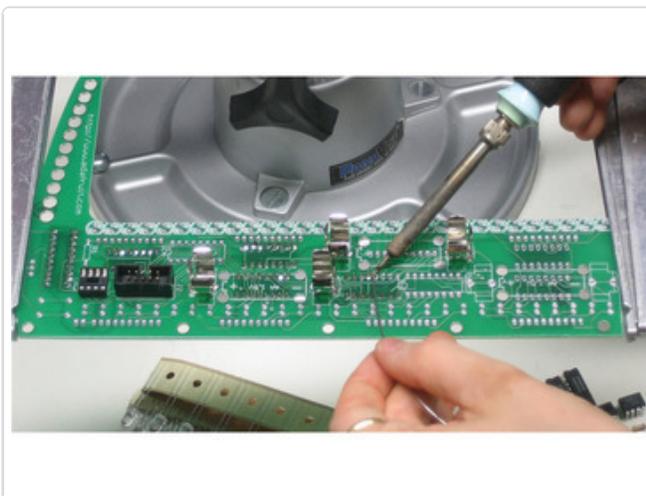
Solder in the smaller socket just like the larger one.

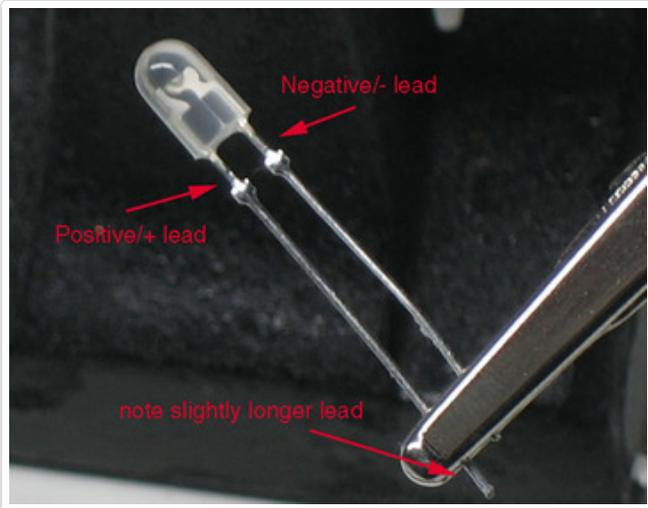
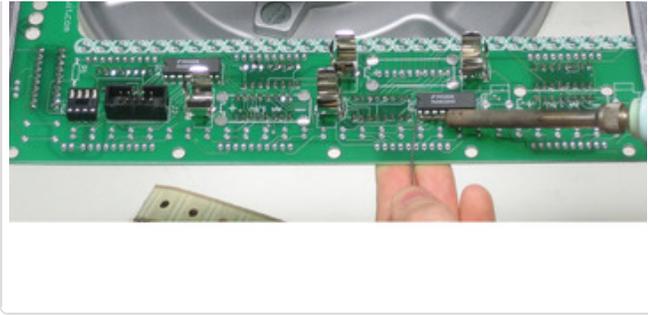




Place the 6 top 74HC595 latches in the proper spots, making sure to align the notches on the chip with the notches in the pictures. Tack them in place by soldering two corner pins of each one from the top. Then turn the board over and solder all the pins.

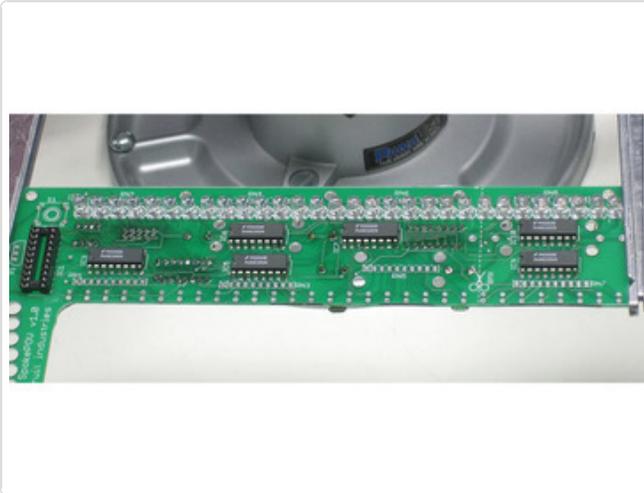
Next, do the same with the 2 latches on the reverse side.





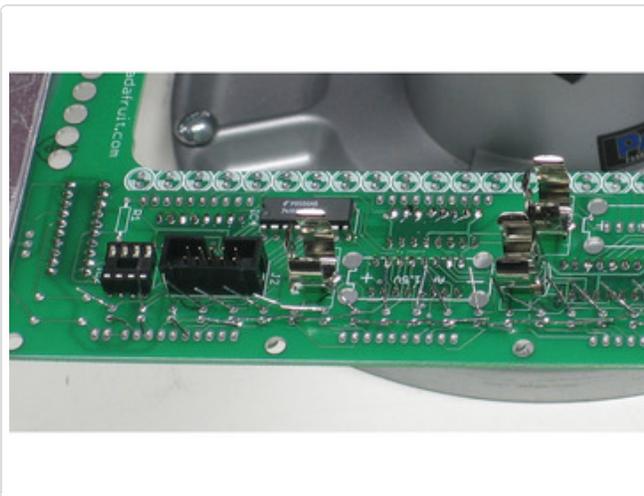
LEDs are diodes, that means they only work in one direction, unlike resistors. That means you **must** put the LED in right or it won't work!

How do you know which way is which? That's easy, the LEDs have one leg that is slightly longer! That leg is the "anode" **positive** leg. The shorter one is the "cathode" **negative** lead.

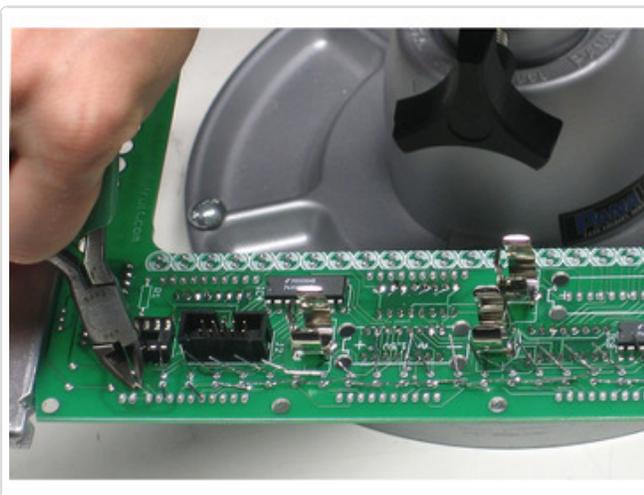


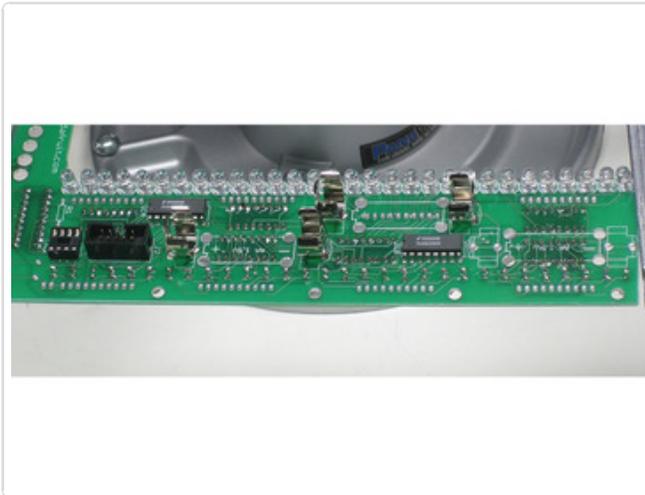
Grab the LED back and place the first 30 into the front row. **Make sure the longer lead/wire of the LED is in the hole closest to the edge of the board.** Otherwise the LED will not light up. This is a pretty common mistake so please work slowly and carefully :) Sometimes the LEDs have the flattened edge of the plastic wrong so go with the length of the leads. Longer lead goes in the hole closest to the edge of the board

When you place each LED, bend the leads out so that it won't fall out when you turn the board over.



Solder the LEDs and clip the leads, either one at a time or all at once, whichever is easiest for you.





Repeat for the other row of LEDs.

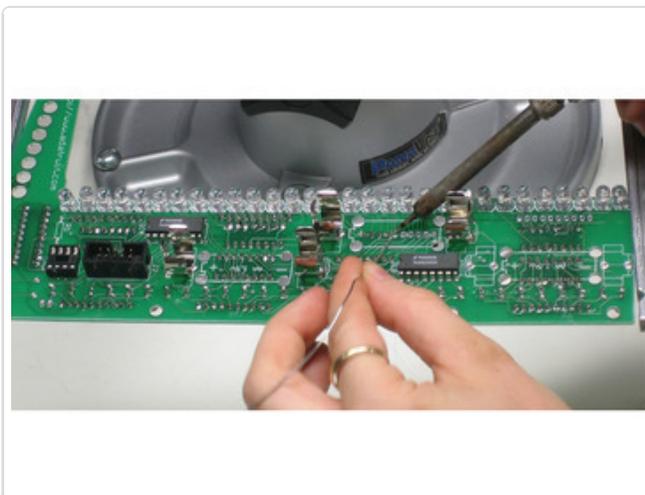
On this side, it can be a little tough to see which side is 'flattened'

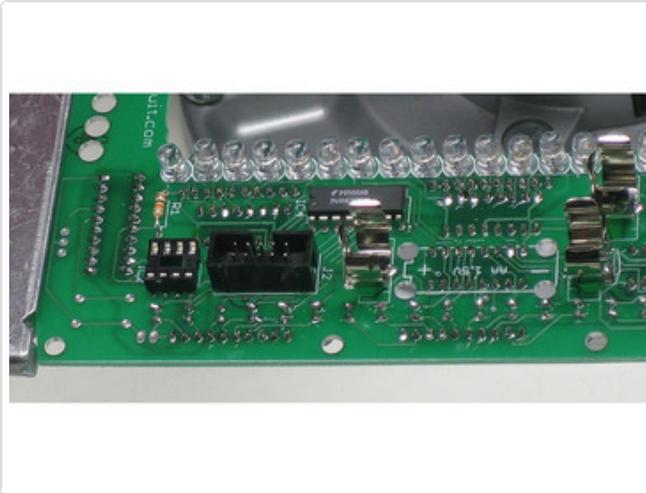
On this side of the kit (with the battery connectors as shown on the left) - the longer leg of the LED goes in the hole along the edge of the PCB.

Sometimes the LEDs have the flattened edge of the plastic wrong so go with the length of the leads. Longer lead goes in the hole closest to the edge of the board

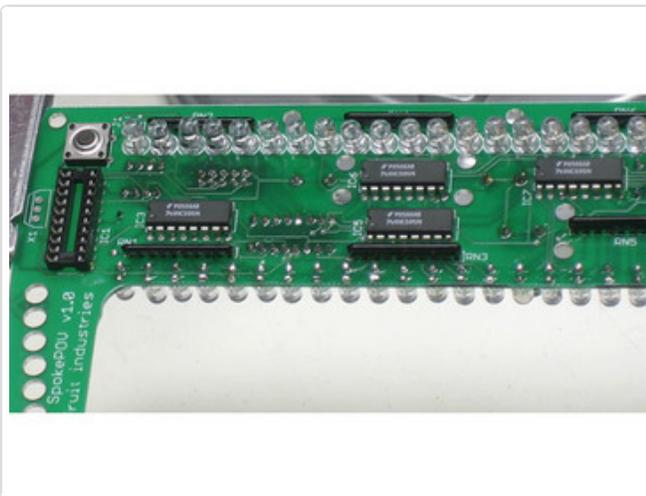


Solder in the 8 9-SIP resistor networks. Make sure you align them correctly: one side of the resistor pack has a dot which corresponds to pin 1 which is marked with an X in the picture on the circuit board.





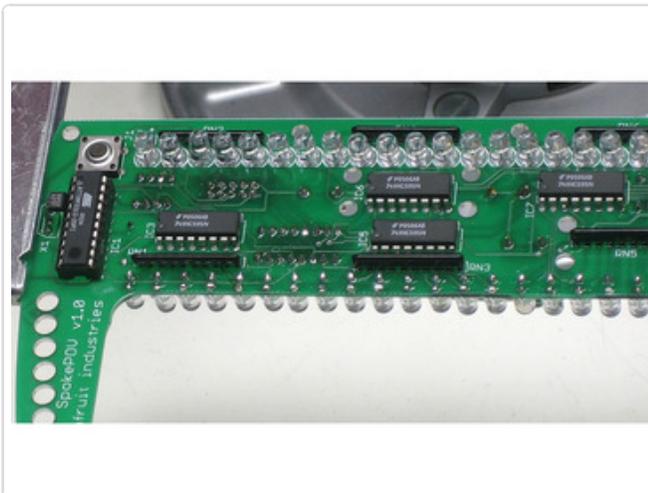
Place the one 10K resistor. Bend the leads like with the LEDs and solder it in, then clip the leads off.



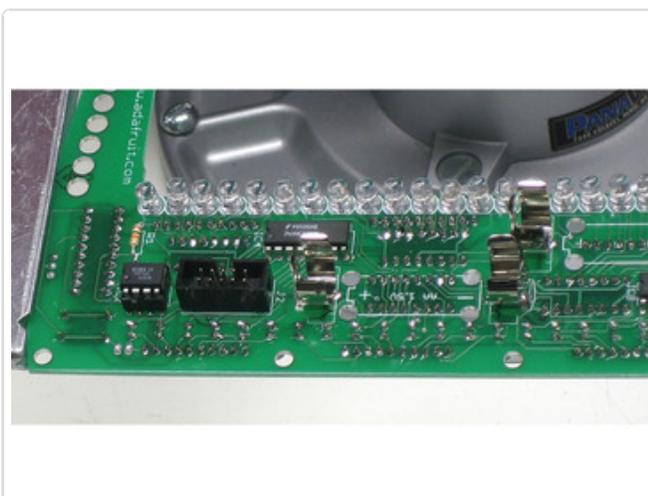
Place the button, which will snap into place. Solder it in.



Bend the sensor so that the face points out as shown. Since the sensor has to stick out but has short leads, solder it from the top, making sure that the leads don't stick too far through on the other side. See the picture to the left.



Put the microcontroller in the socket so that the notches match up. The sensor can 'lean' on the microcontroller. If you want, you can use a bit of glue to support it.



Place the EEPROM in the smaller socket so that the notches line up. You're done!

Now that you're done assembling it, you can test the board to make sure it's functioning. Put two (or three) good AA batteries into the clips. Whenever the microcontroller gets notice of a 'hard reset' it goes through a little test routine where it lights up all the LEDs in order, then it lights up one LED on each side of the SpokePOV to indicate that it's still on. The actual LED itself will vary as it is used for debugging.

If none of the LEDs are lighting up, go back and check to see if you put in any latches, batteries or the microcontroller in backwards. If just a few LEDs aren't lighting up, check if they're in backwards.

SpokePOVs now come pre-programmed with some sample images. You can trigger the images using a magnet, sweep it past the sensor a few times and then wave it in the air, you should see a pattern displayed.

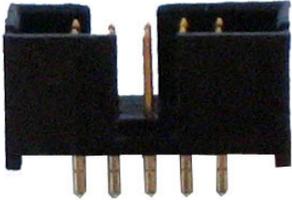
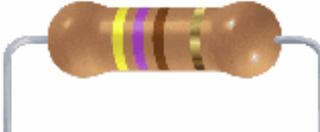
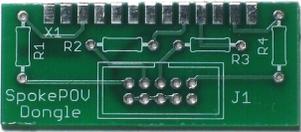
Can't get it working? Don't worry, help is available in [the forums \(http://adafru.it/forums\)](http://adafru.it/forums)!

Parallel Port Dongle

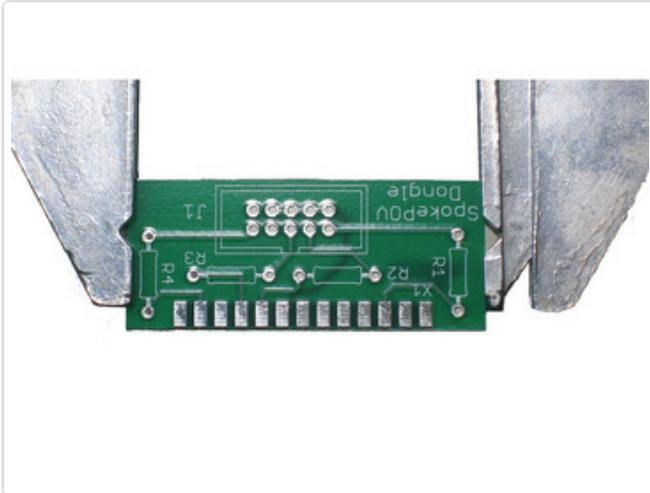
This kit isn't made anymore, but we're keeping the documentation up anyways! Enjoy but we don't have this for sale!

Make it! - Dongle

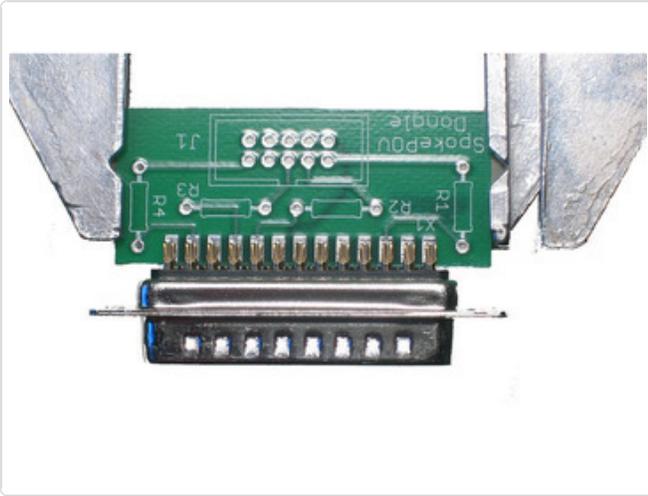
SpokePOV Parallel Port Dongle Parts

	J1	Programming Header	30310-6002HB	1
	X1	DB-25 male connector w/solder cup connectors		1
	R2-R4	1/4W 5% 1K resistors	Generic	3
	R1	1/4W 5% 47 ohm resistor	Generic	1
		PCB	Custom	1

SpokePOV Dongle Assembly Instructions

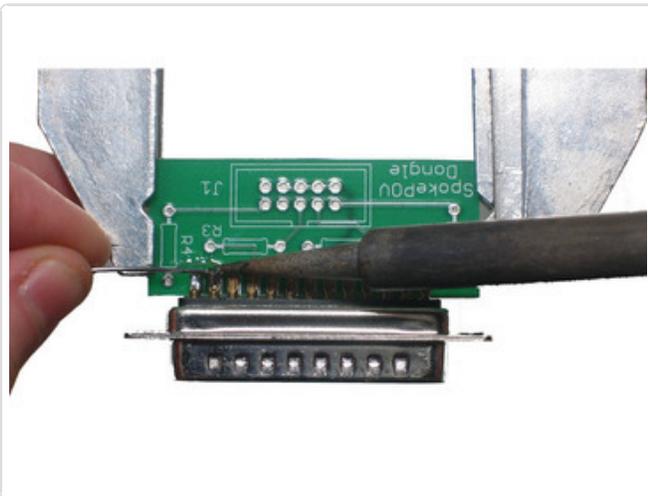
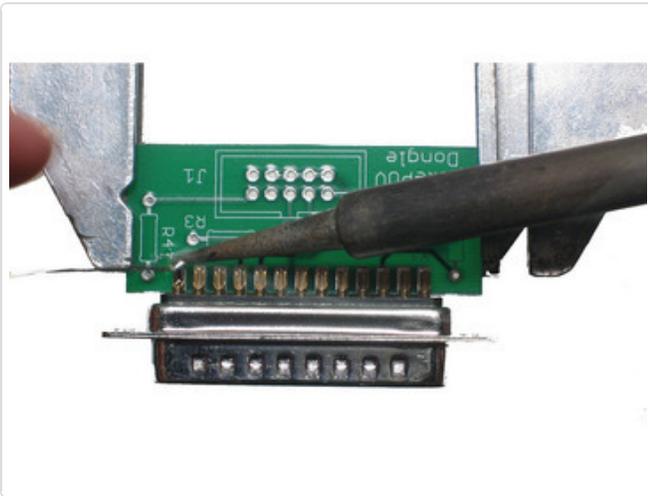


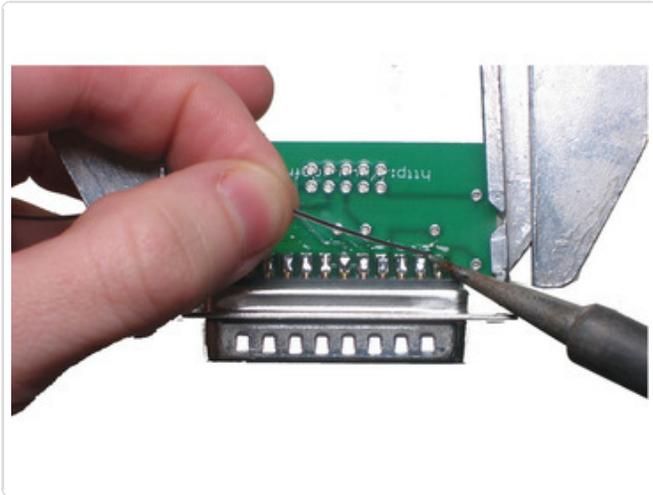
Get your soldering iron hot and put the small circuit board in a holder (if you have one).



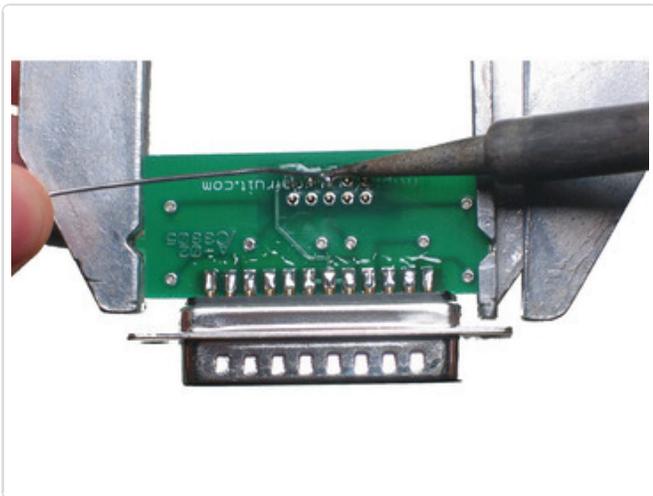
Slip the DB-25 socket onto the edge of the PCB so that the pads line up with the solder cups.

Now solder all the pads. Make sure you heat both the pad and cup and that they are soldered together, it's easy to not hit both.



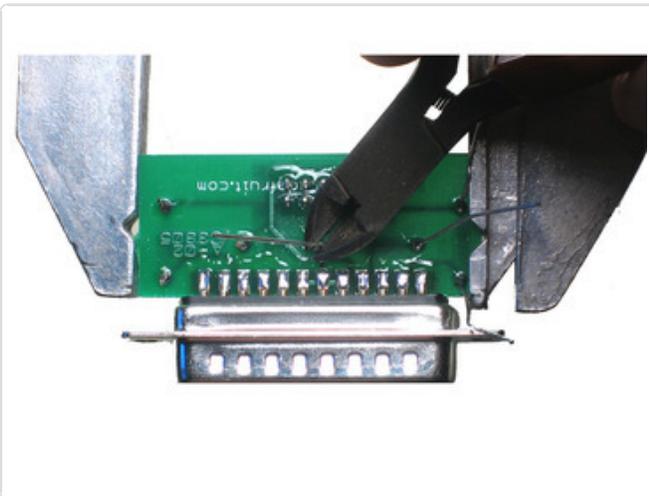
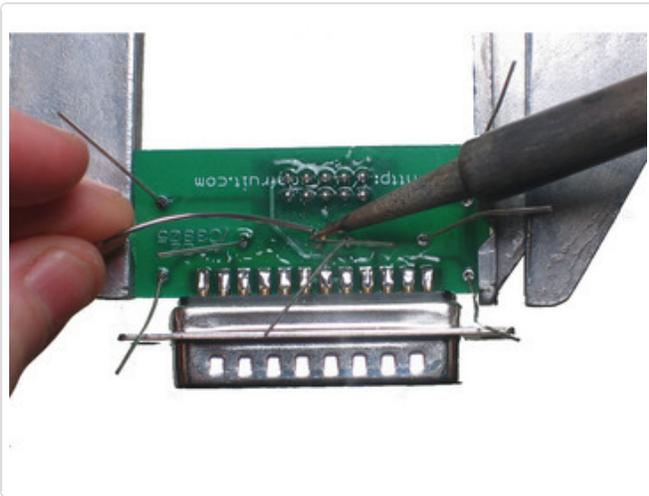


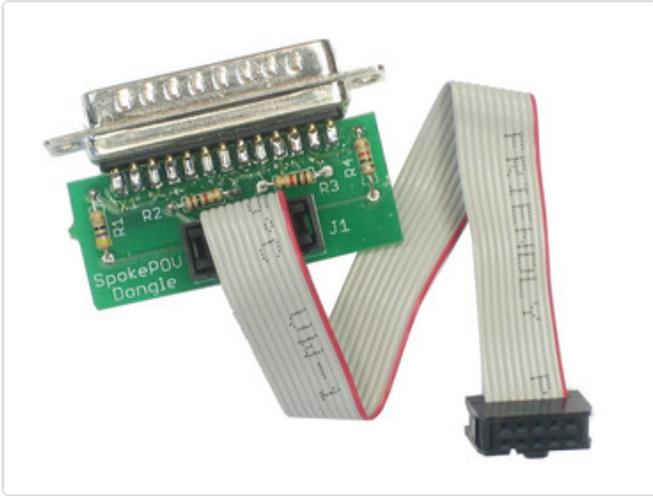
Flip the board over and do the same on the other side.



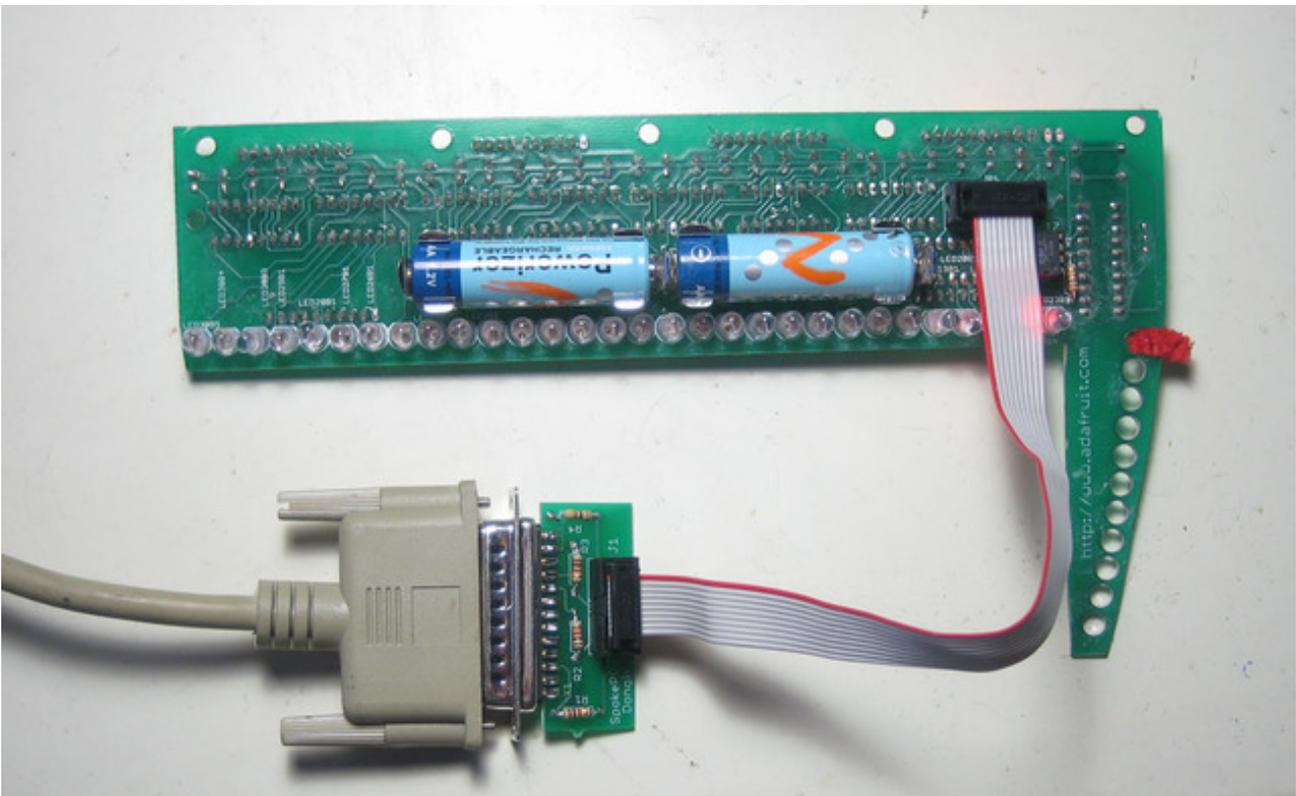
Put in the jumper header (make sure the notch matches in the picture) and solder it in.

Place the resistors in, solder and clip the leads.





Put the jumper into the header. OK, you're done!



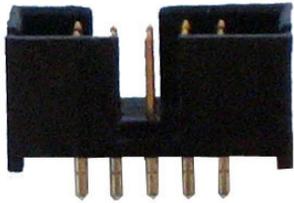
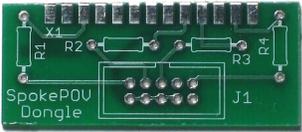
Connect the dongle to the SpokePOV and your computer parallel port to upload images. I bought a 6' parallel-port extension cable to make it easier.

Serial Dongle

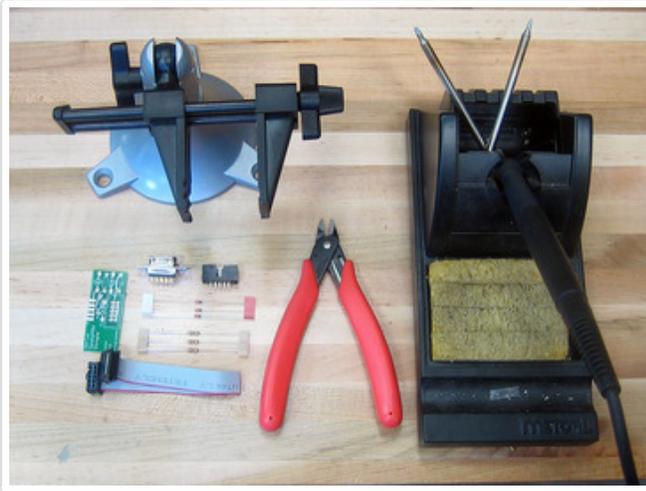
This kit isn't made anymore, but we're keeping the documentation up anyways! Enjoy but we don't have this for sale!

Make it! - Serial dongle

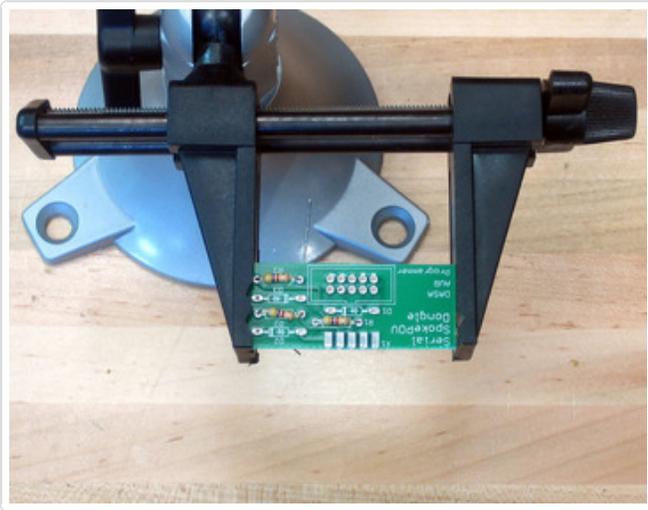
Serial Port SpokePOV Dongle Parts

	J1	Programming Header	30310-6002HB	1
	X1	DB-9 female connector w/solder cup connectors		1
	R1-R3	1/4W 5% 4.7K resistors	Generic	3
	D1-D3	5.1v Zener Diode	Generic	3
		PCB (looks a little different than shown here)	Custom	1

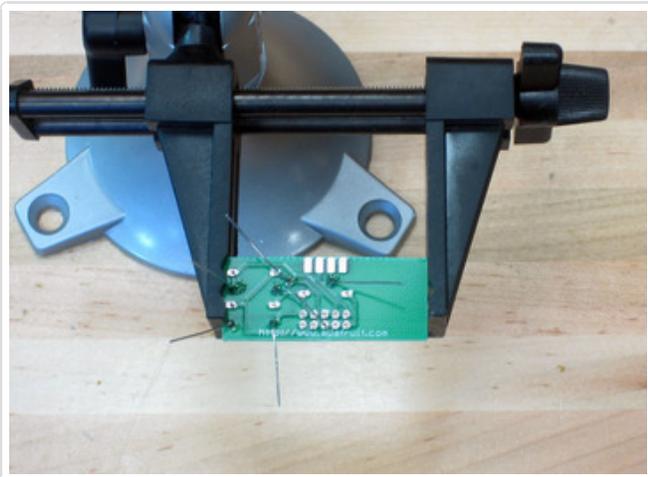
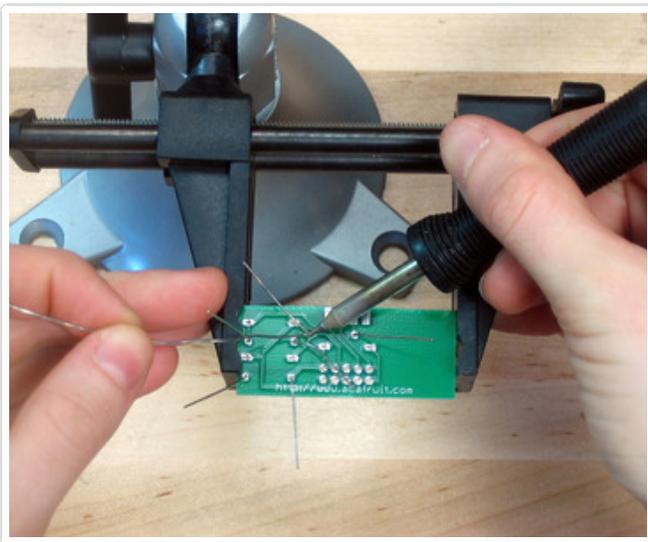
Assembling the Serial Dongle

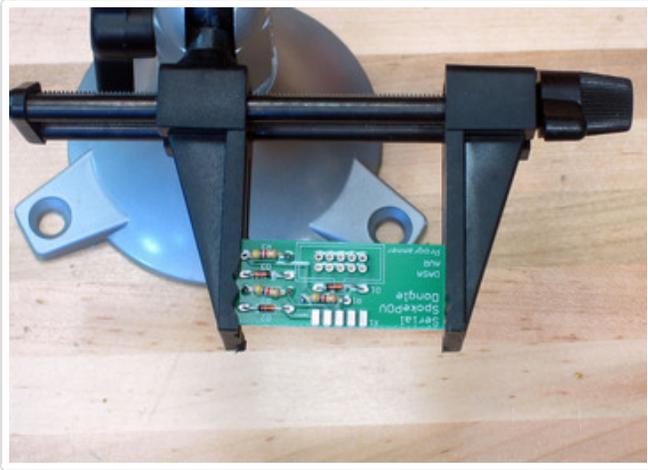


Get your bench set up for soldering.

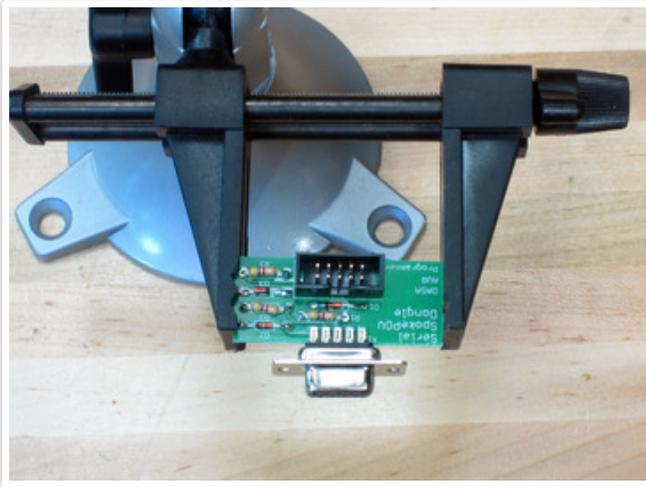


Place and solder in the 3 resistors, they don't have a specific orientation.

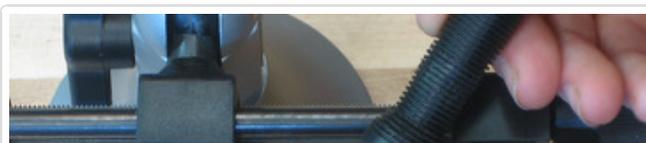
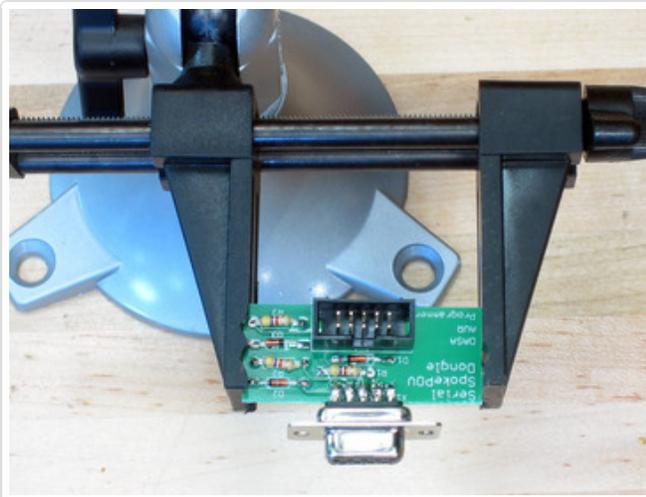
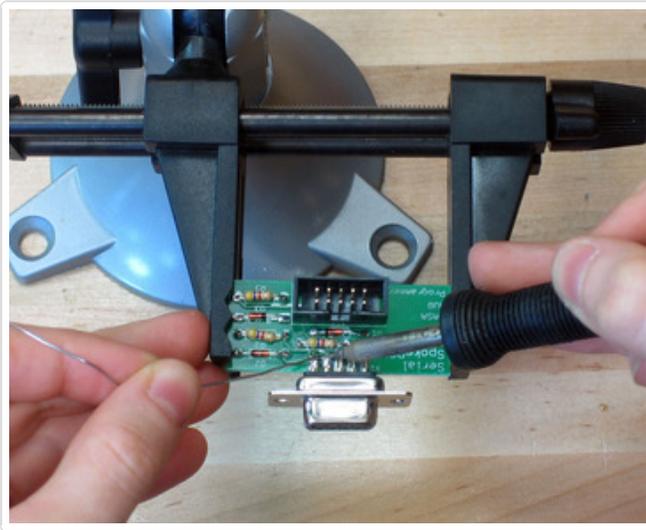


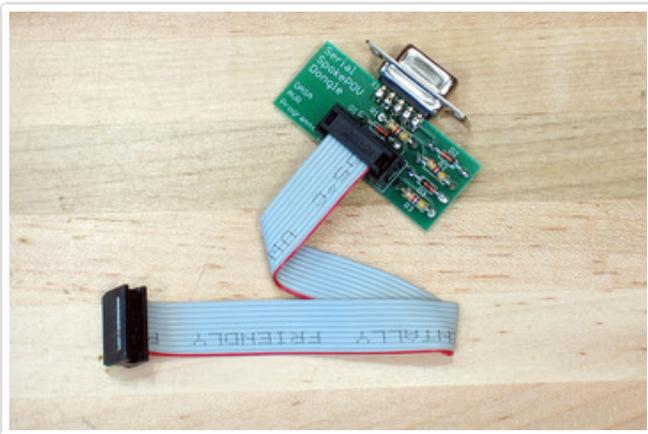
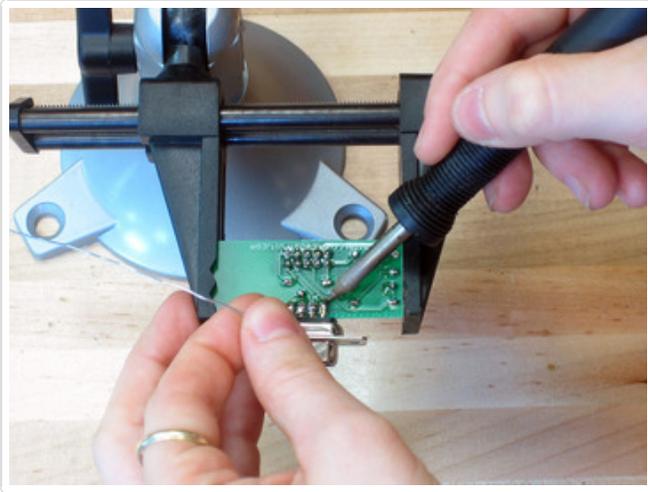


Place and solder in the zener diodes, making sure that you have the black stripe on the diode aligned with the white stripe on the PCB image.



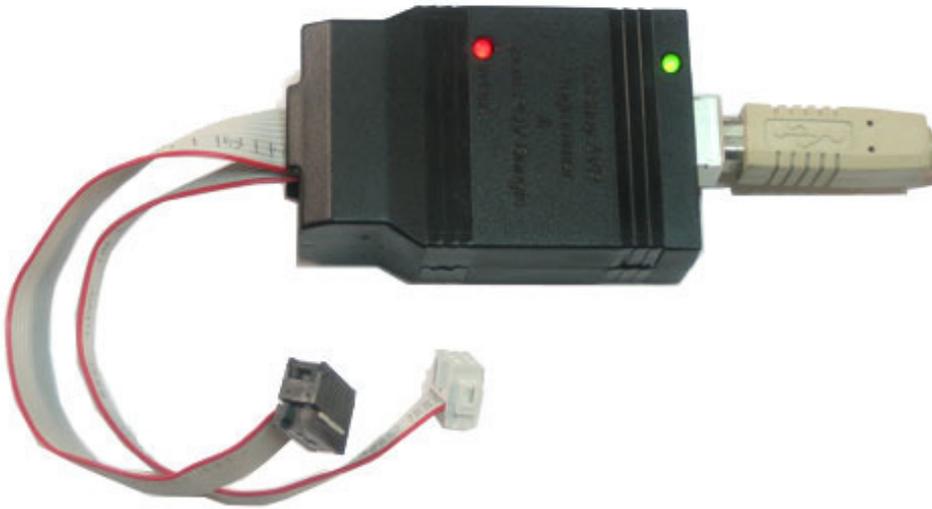
Place the box header, matching the notch up, and the DB9 connector on the edge of the PCB. Solder both, making sure that you get solder underneath the pins of the DB9.





Ok you are done!

USB Dongle



Make it! - Serial dongle

USB Port SpokePOV Dongle Instructions

The USB dongle is actually a dual-use kit, and the instructions live [over here at the USBtinyISP page \(http://adafruit.com/dhl\)](http://adafruit.com/dhl).

Follow those instructions to assemble it, but use the windows driver available on the [SpokePOV download page \(http://adafruit.com/dhj\)](http://adafruit.com/dhj) (it's actually an identical driver, only difference is that the name will appear as SpokePOV dongle instead of USBtinyISP).

You don't need to assemble or use the 6-pin cable as the SpokePOVs use the 10-pin cable only.

Using the USB dongle

Don't forget! Remove the jumper when attaching it to a SpokePOV dongle and put batteries into the SpokePOV to talk to it and upload images

Make sure you're running the SpokePOV software version 1.3 or higher, with USB support for Mac and Windows!

You can **Test Port...** on the USB dongle but the pictures show the older parallel/serial interfaces. Just pay attention to the box header and ignore that the PCB looks different.

Customize

Use it! - Upload images

Introduction

The SpokePOV software allows you to easily upload, download, verify, manipulate, fold, spindle, etc. your SpokePOVs! The software is written in C++ and is available as a Windows / MacOS X executable ([download here \(http://adafru.it/dhj\)](http://adafru.it/dhj)).

Note: Some people report it doesn't work on Intel Macs, will try to get this fixed soon and update the software.

The older, wxPython version documentation is [here \(http://adafru.it/dhL\)](http://adafru.it/dhL).

Don't forget: **Even if you only want to display 1 image, the SpokePOV will always go through all the banks. So if you have 4, you must upload the same image to all four banks!**

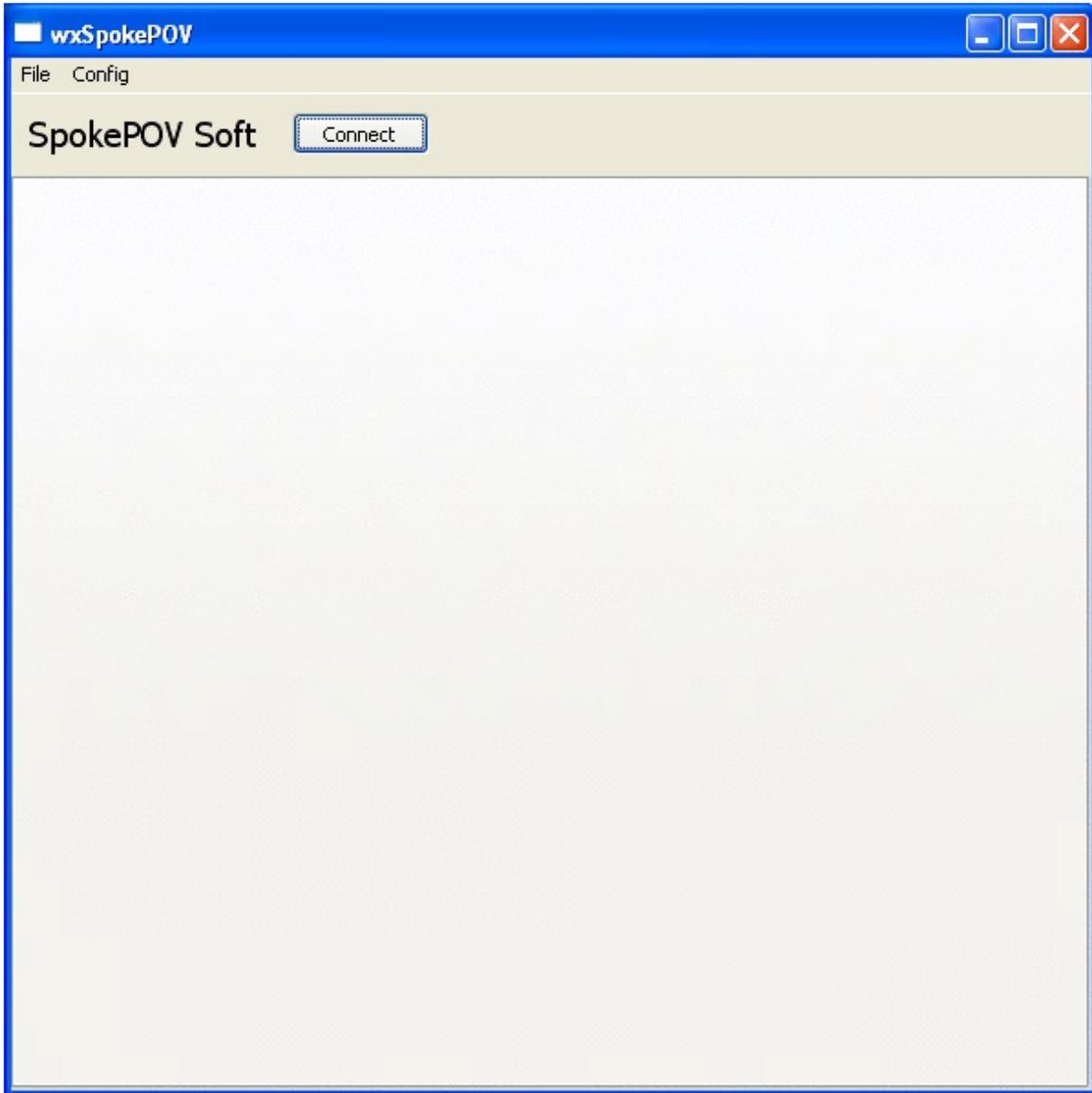
Don't forget...**take the jumper out of the USB dongle if you've got one! Otherwise your batteries will go pop!**

Configuring

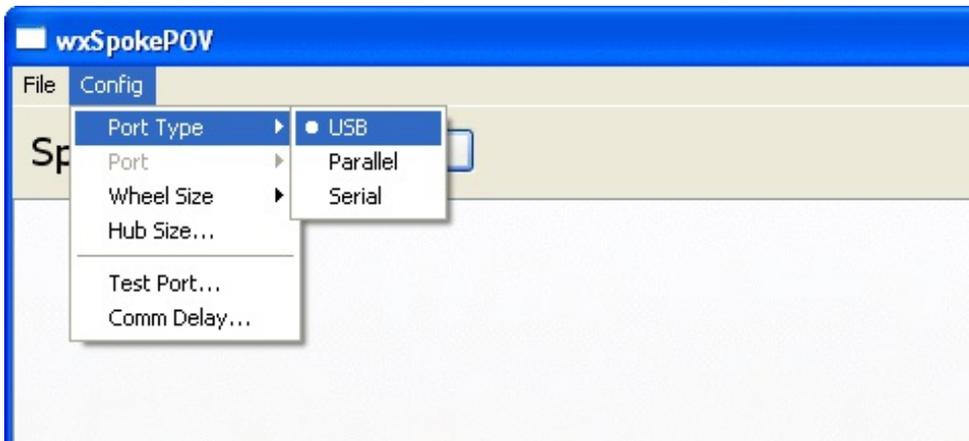
For windows users: download the zip file and uncompress it somewhere convenient. If you are using the parallel port, download and run/install **giveio** (from the [Download \(http://adafru.it/dhj\)](http://adafru.it/dhj) page) to let the software talk to the parallel port.

For MacOS X download the SpokePOV dmg and copy the application to somewhere convenient.

Now, start up the software! You will see a blank screen, there are menus and a connect button. Don't connect yet, you must first configure the port.

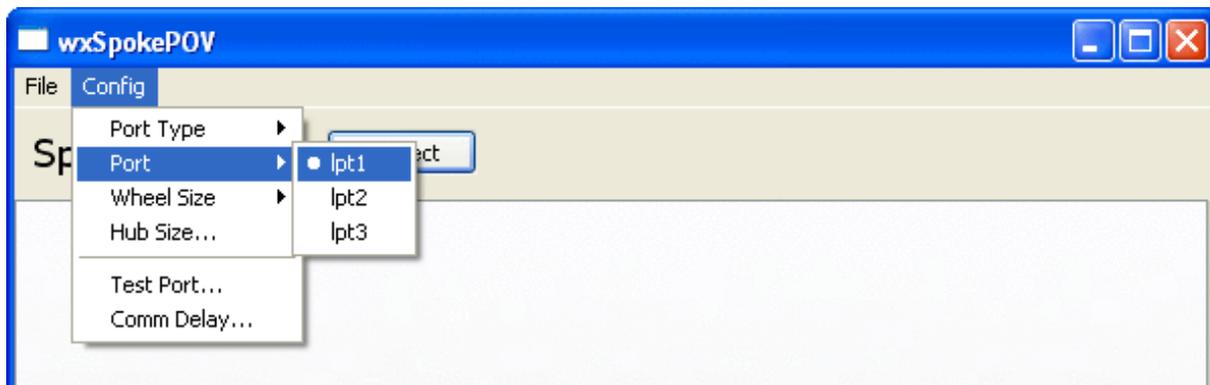
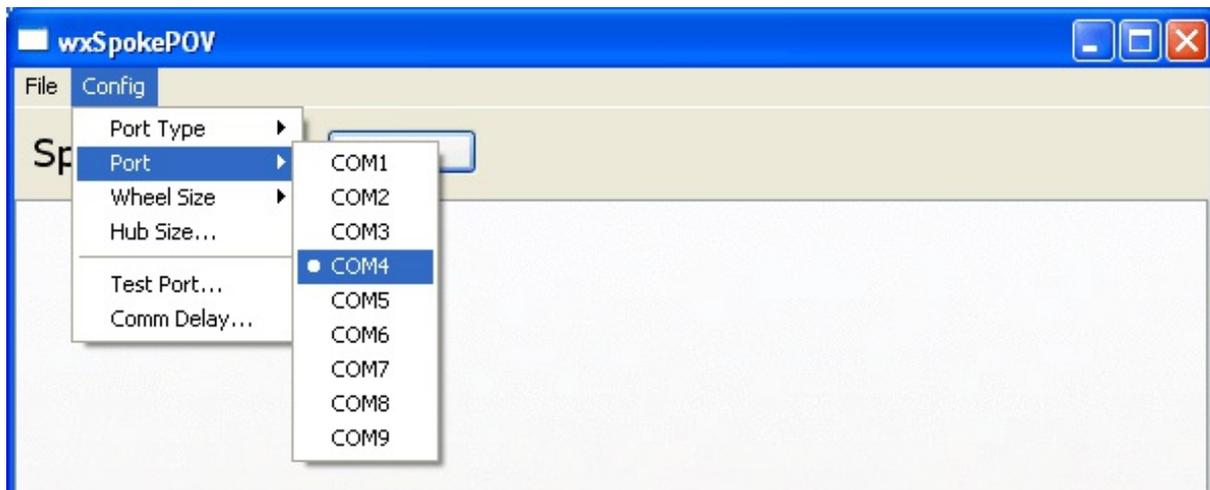


First select USB, serial or parallel port (on Mac, you can only choose USB or serial).

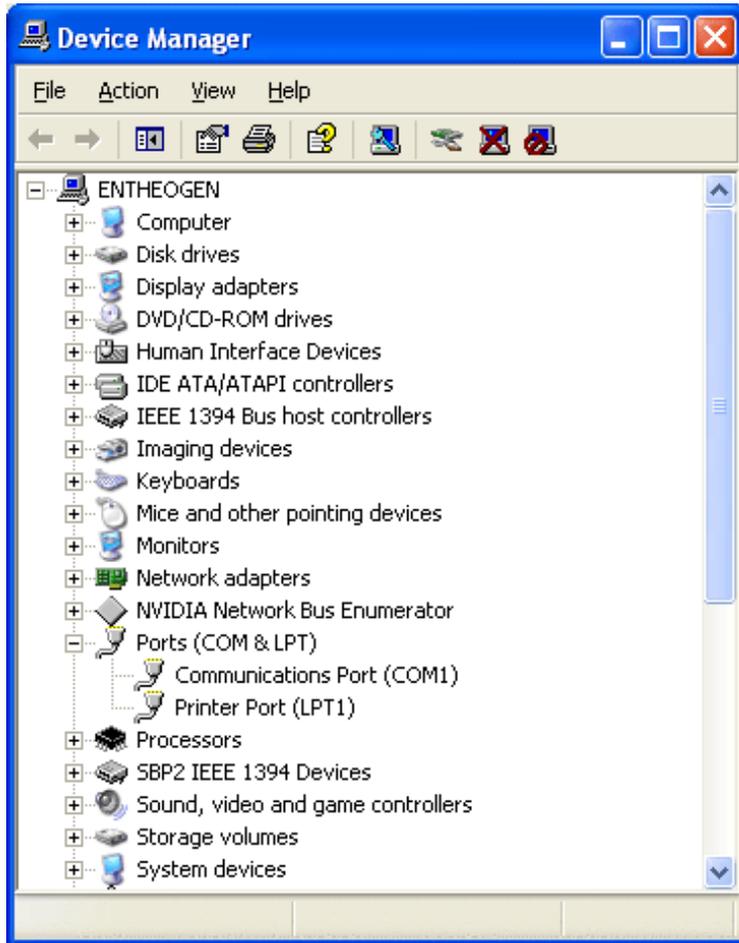


Next, for serial or parallel, select which port to use.

Important Note: under MacOS X, using a USB/serial adapter, sometimes there are multiple names for one port (`/dev/cu.KeySerial1` and `/dev/cu.USA19H1b1P1.1` for example). Try both! I've found they act differently and get better reliability with some over others!



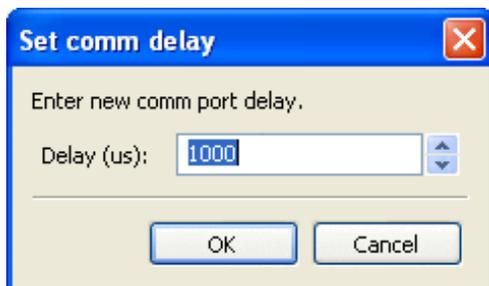
Check your Device Manager under windows, to verify what ports you have available, if you're not sure.



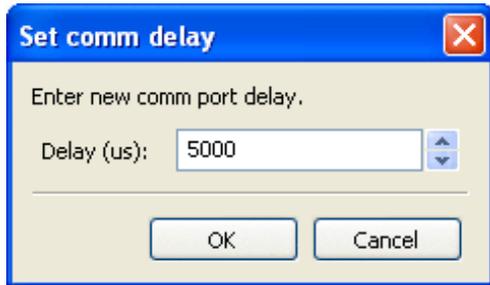
Next, select the Communication Delay. This is only really important if you're using a USB/Serial adaptor. These are very slow devices (despite being USB) so we need a higher delay.

For USB, the delay is ignored so it doesn't matter what it's set to.

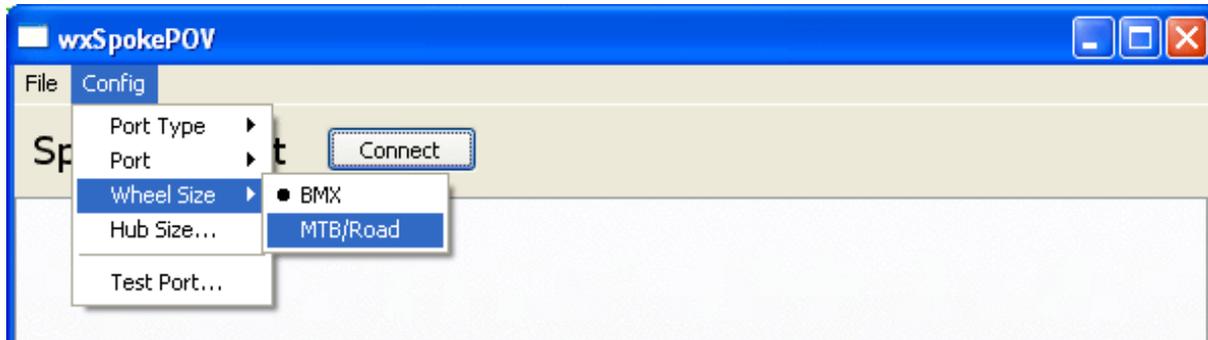
For parallel port as well as 'direct' serial (where the computer has a serial port right on the motherboard) use a delay of 1000 uS. Increase it up to 2000 uS if it's being flakey.



For USB/Serial adaptors, start at 4000 uS and increase it by 1000 until you get consistent results.



Lastly, select what size wheel you're using - BMX (22 LEDs) or MTB/Road (30 LEDs).



Finally, set the hub size. Measure your hub diameter. Better yet, measure how far the end of the attached SpokePOV is from the center of the wheel and then multiply by two to get the diameter. This will define how big the white space in the middle of the wheelpanel is and will give you the most precise images!



The configuration will be saved so you only have to do this once.

Testing the connection

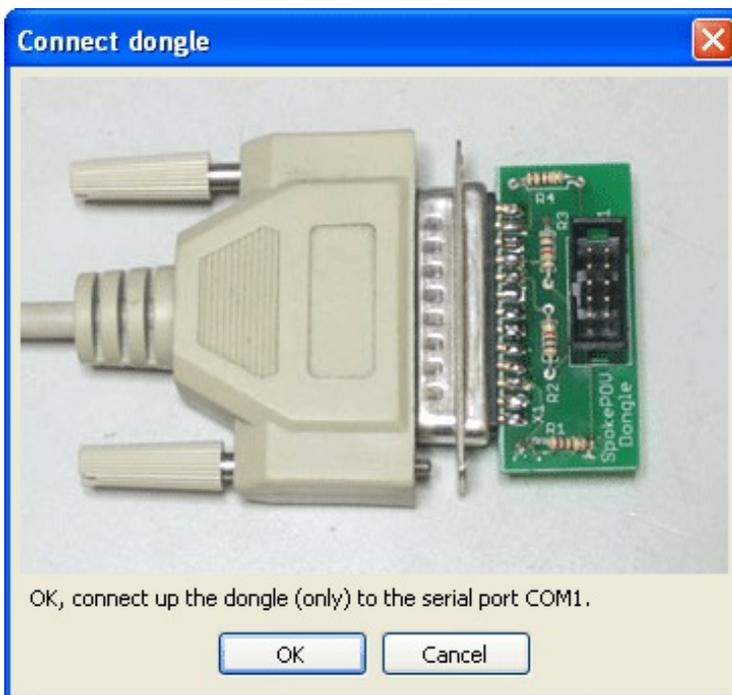
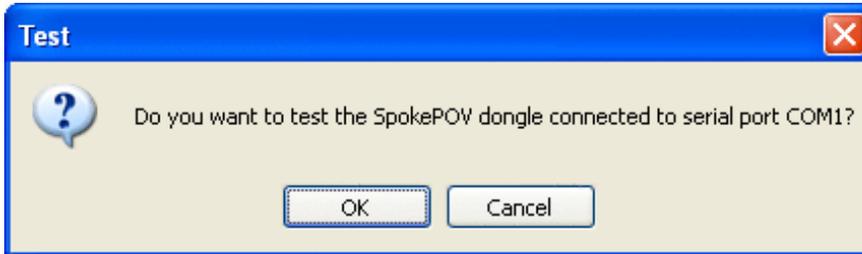
A big problem for people is figuring out whether there's a problem with their spokePOV dongle or software. Therefore there is now a port testing feature that will help you debug the communications system using only a multimeter!

Dont forget...take the jumper out of the USB dongle if you've got one! Otherwise your batteries will go pop!

Select the Test Port feature from the configuration window.



Make sure that you are testing the right *kind* of port and the correct connection.



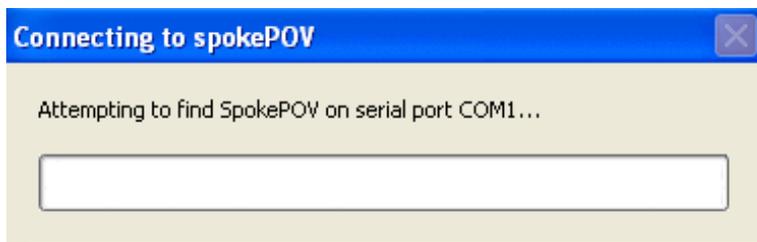
Now go thru all the steps, verifying the connection. You can **Test Port...** on the USB dongle but the pictures show the older parallel/serial interfaces. Just pay attention to the box header and ignore that the PCB looks different.

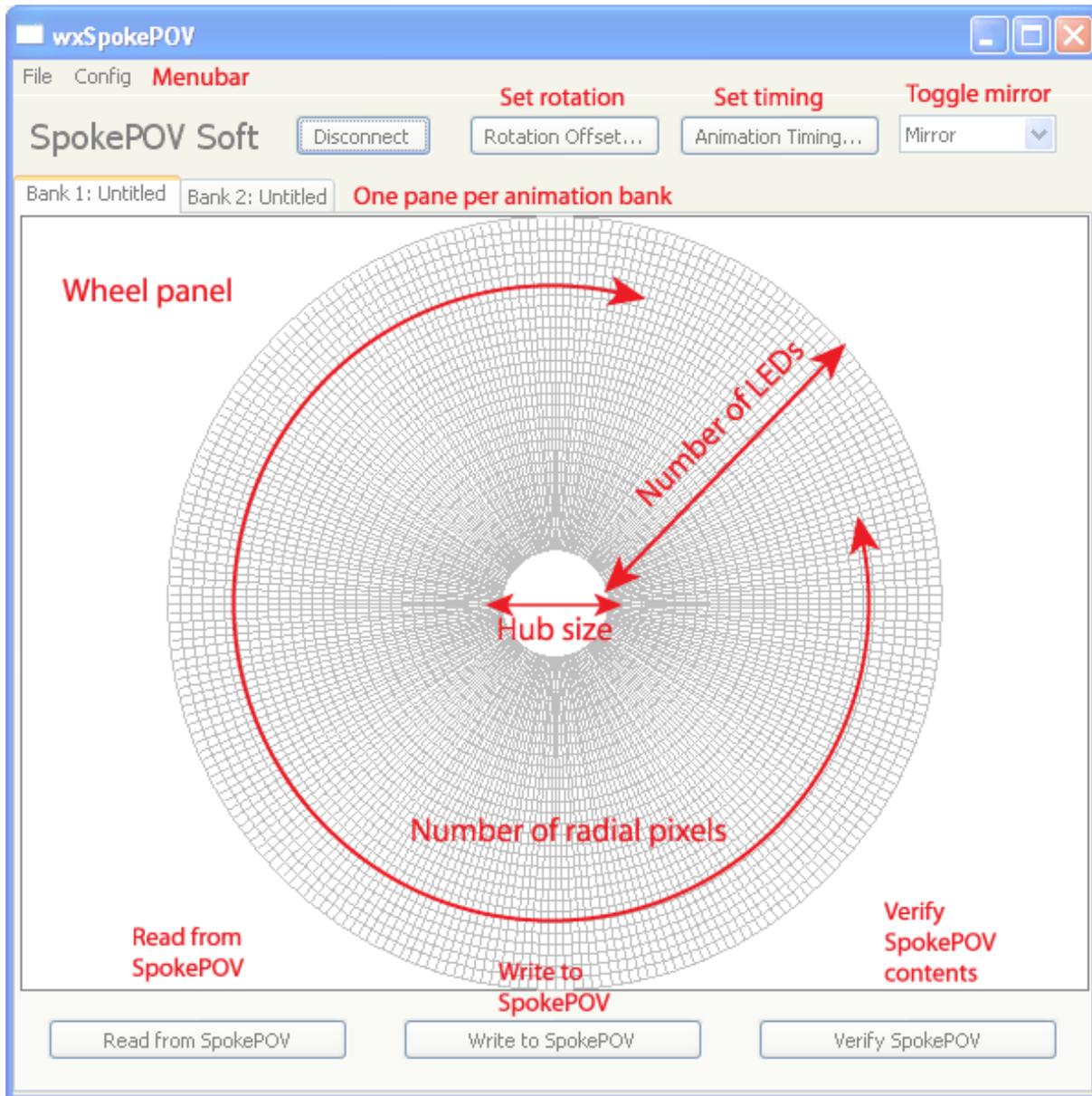


If it worked out, you should get this message!

Connecting

Turn on the SpokePOV, preferably by 'hard reset' - removing and replacing a battery. Make sure you have recharged/good batteries. Plug the dongle into the SpokePOV and click on the Connect button to connect to the SpokePOV.





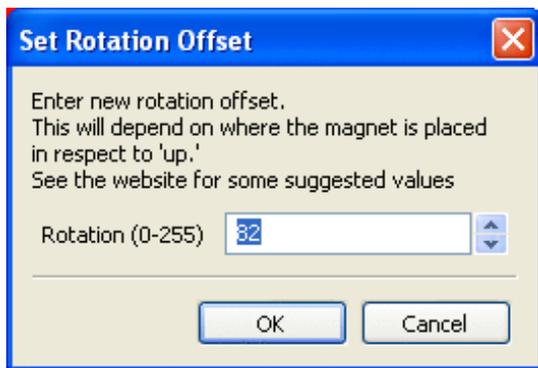
Once you connect, you can see the graphical interface for editing what image will be displayed. The **wheel panel** shows a simulated image of what is in the SpokePOV.

The main controls for the SpokePOV are in a panel below the wheel panel. The three buttons are used to 1) **write** the image currently in the wheel panel to the connected SpokePOV 2) **read** the image currently in the connected SpokePOV into the wheel panel and 3) **verify** that what's in the wheel panel is also what's in the SpokePOV.

There are also 3 controls up top, these are configurations for the entire SpokePOV. The **rotation offset** control allows you to change what the offset from 0 is. What this means is: when you upload an image, you'll do it right side up. That means that the SpokePOV starts displaying from the top, rotating clockwise. However, you're likely to put the magnet not at 12 o'clock but rather somewhere else, depending on whether you put the magnet on the fork, or chainstay, etc. [Here is a diagram of where these frame parts are](#)

located (<http://adafru.it/dhM>) and some rough numbers you can start with. The numbers range from 0 to 255 starting at the top and rotating so that 255 is right next to 0 again.

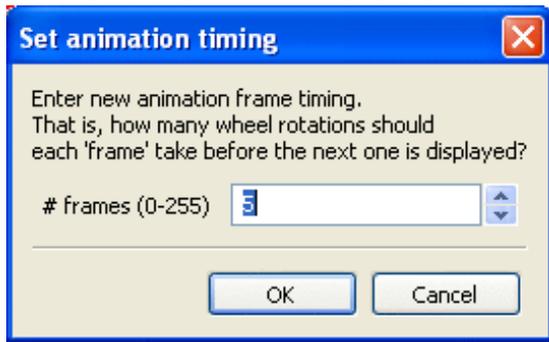
Location	Value
Fork	220
Chainstay	64
Seatstay	32



You can also set the **animation timing**, which is how many wheel rotations the spokePOV will spend at each animation frame before going to the next bank.

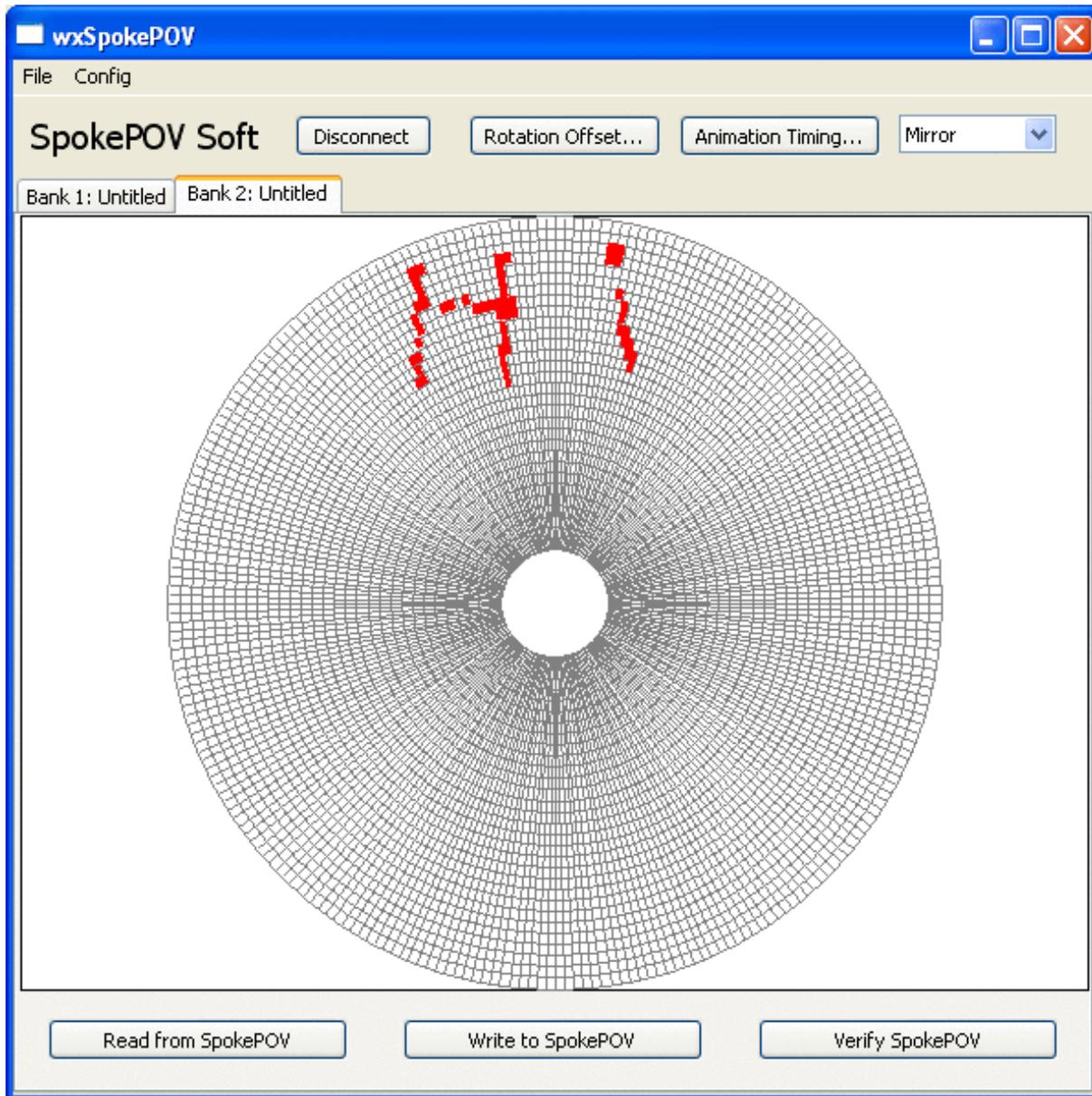
The images are stored in the 25Cxx EEPROM, each image takes 1Kbyte of storage. Therefore a 1K EEPROM (25C08) will store one image, a 2K EEPROM (25C16) will store two and a (25C32) will store four.

1. The firmware in the microcontroller automatically goes through the EEPROM showing each image, without knowing how large the EEPROM is. Therefore, you can swap out a different EEPROM at any time you want.
2. This also means that **you have to fill all the banks with something**. You can't just upload an image to the first bank and nothing to the other three, or the spokepov will display 'nothing' for 3 out of the 4 animation cycles.
3. You can sync all the spokepovs by quickly resetting them before you start.
4. Animation doesn't work great, but it does work. One way to improve it may be (assuming you have more than one spokepov) to leave out the hall effect sensor on all but one and wire the ground and sense wire (but not the VCC power wire) all together so that they are forced to sync together. You'll have to mess with the offset value though.



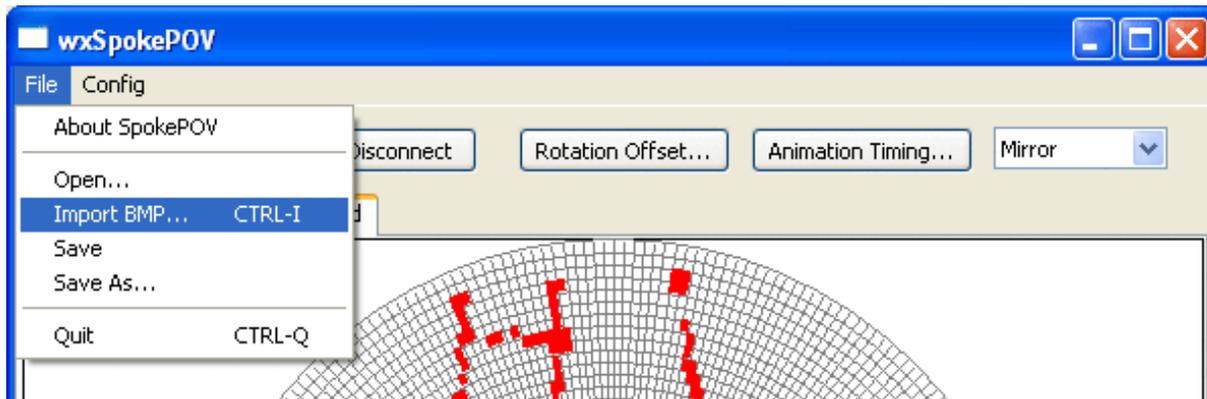
There is also a toggle for **mirroring the images**. Since SpokePOV is double sided, you can have the image on the opposite side either be mirrored (so that it looks backwards on the opposite side) or not (so it appears exactly the same on both sides). For example, if you have text you want to display, the image should not be mirrored. However, if you want an arrow that points in the direction you're biking, then you want it to be mirrored.

When you start up, and connect, the wheel panel is empty. You can click (and drag) in this window to 'draw' pixels.

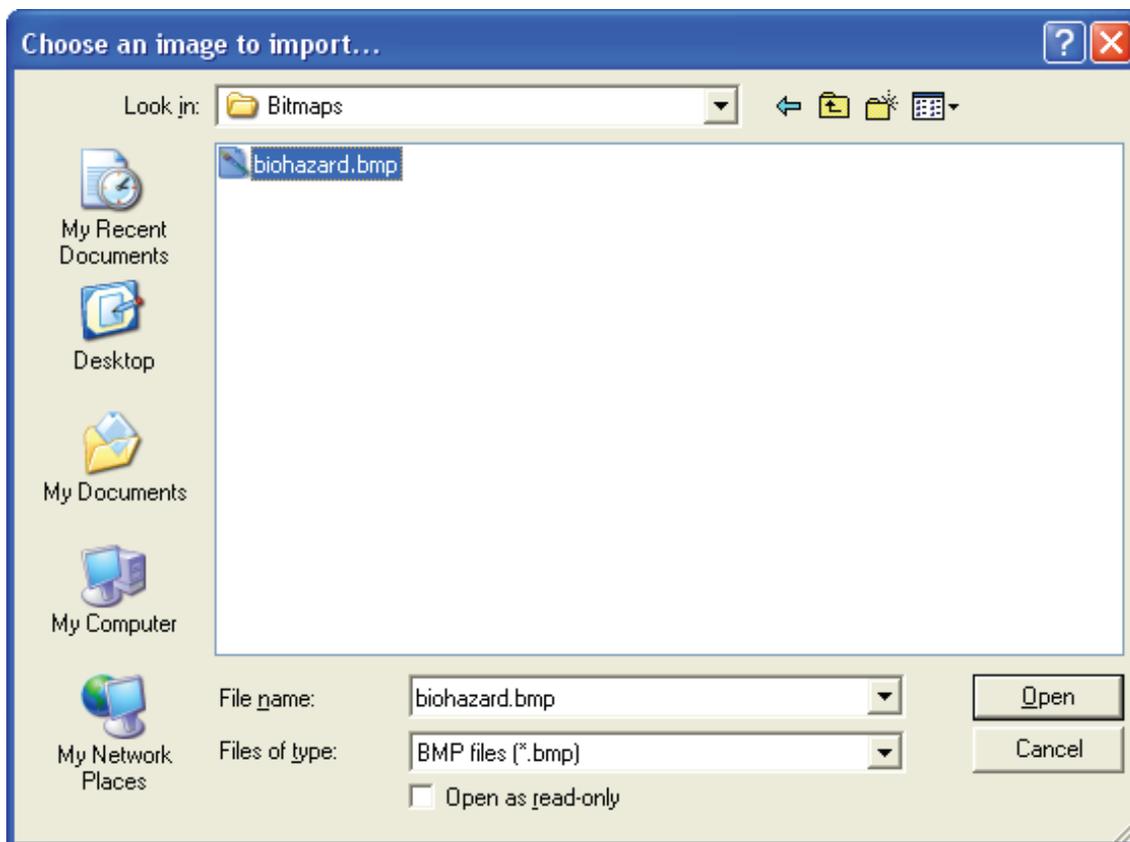


This is a very slow process as the software isn't very fast, it's better to import a bitmap (.bmp file) from a different program and then edit it. The wheel panel has one square for every pixel of resolution. If you're using this with a BMX configuration, the wheel will have fewer LEDs so it will appear a little different. The empty spot in the middle corresponds to the hub size, the space where no image appears.

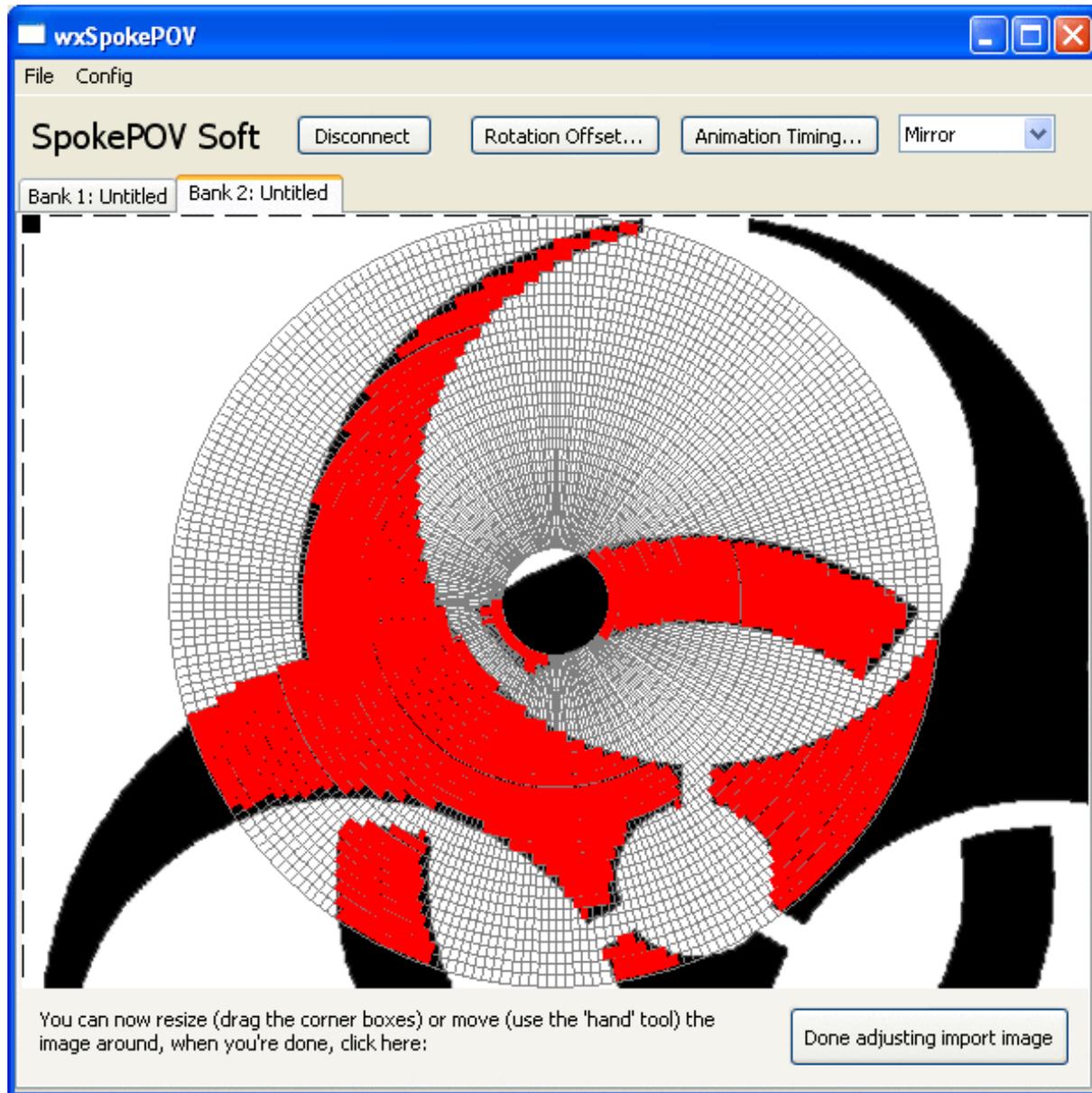
First thing you probably want to do is import a bitmap image. There are a couple example bitmaps distributed with the software.. Bitmaps must be monochrome or at least grayscale. The software assumes that 'white' pixels means no LED should be on and black pixel means a LED on. You'll get best performance if your bitmap is at least 600x600 pixels large.

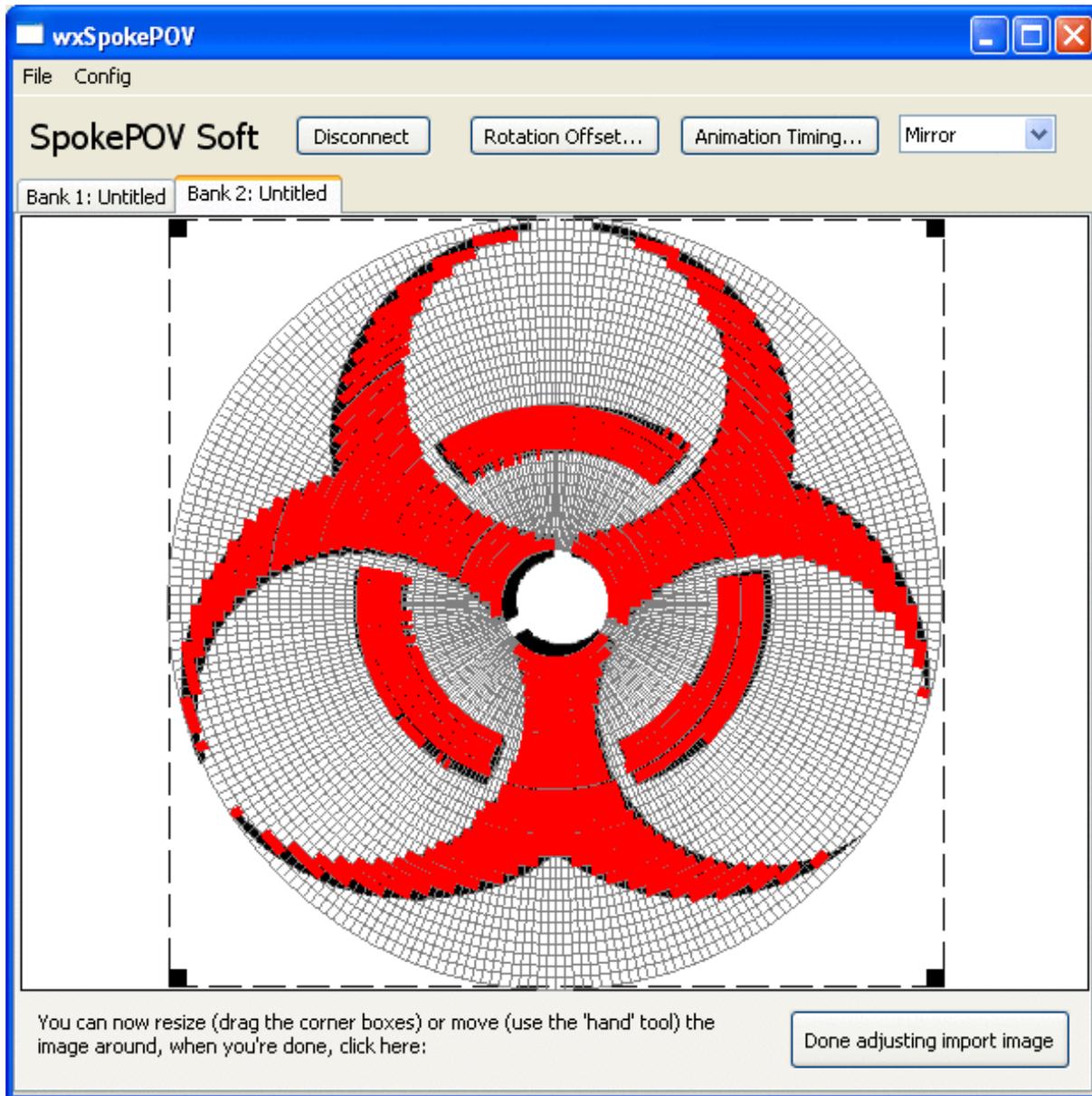


When you import a bitmap, the software will automatically figure out what the image should look like. The imported image will appear in the background, the red lines show where the LEDs will be lit.



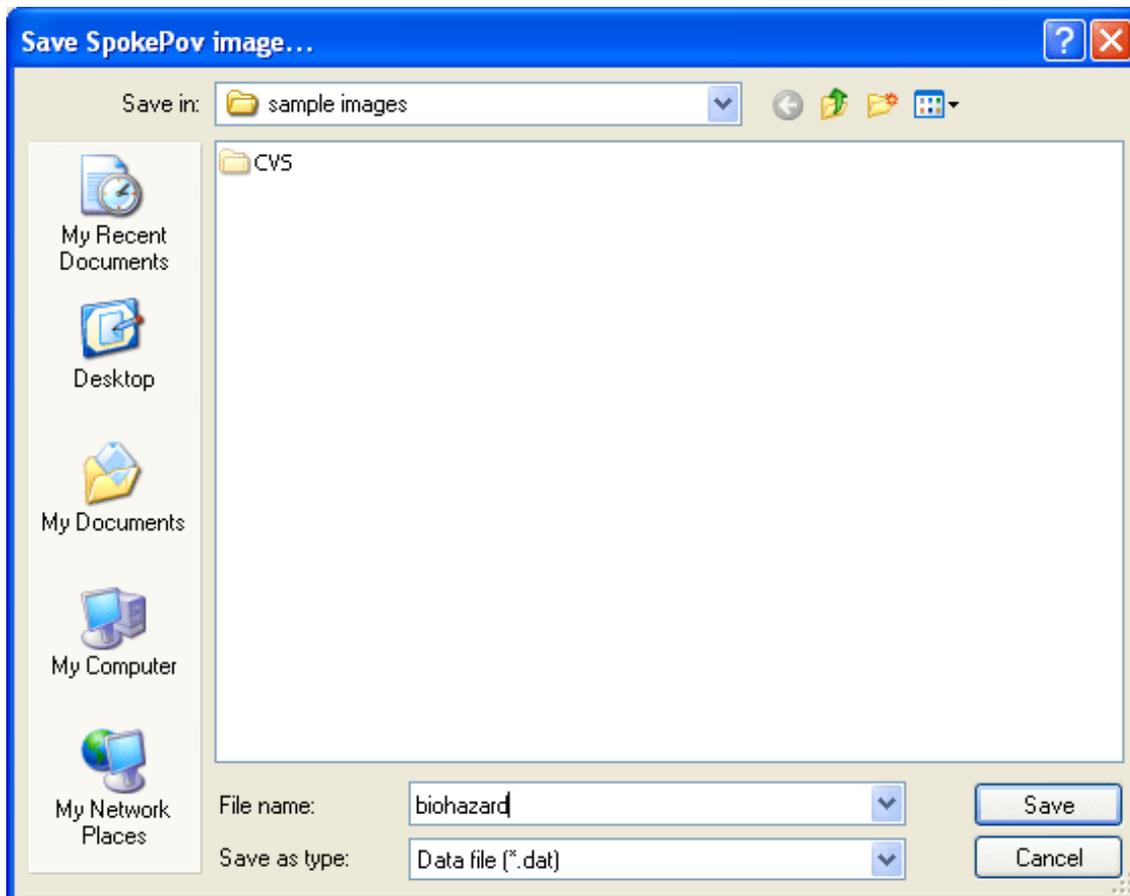
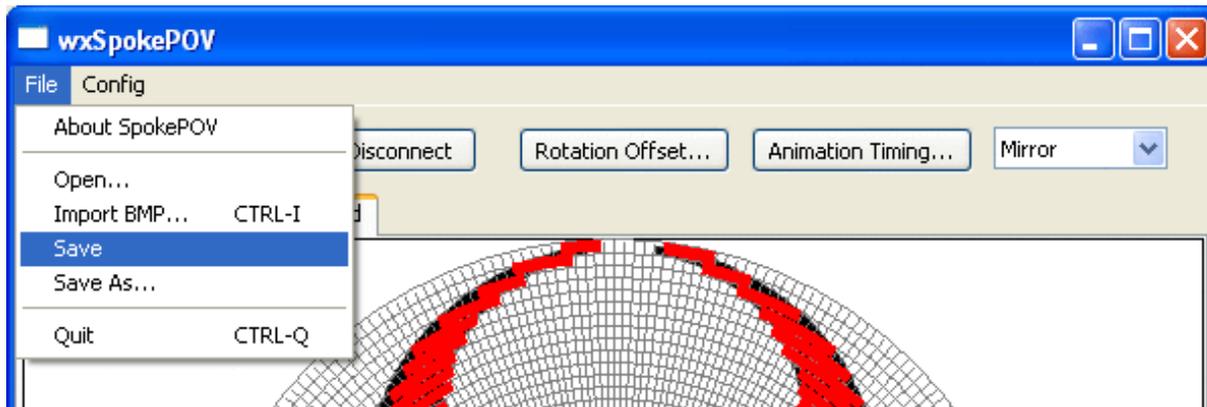
You can then move the image around (hand tool) and resize it (pull on the corner tabs, holding down shift will keep the image proportional).



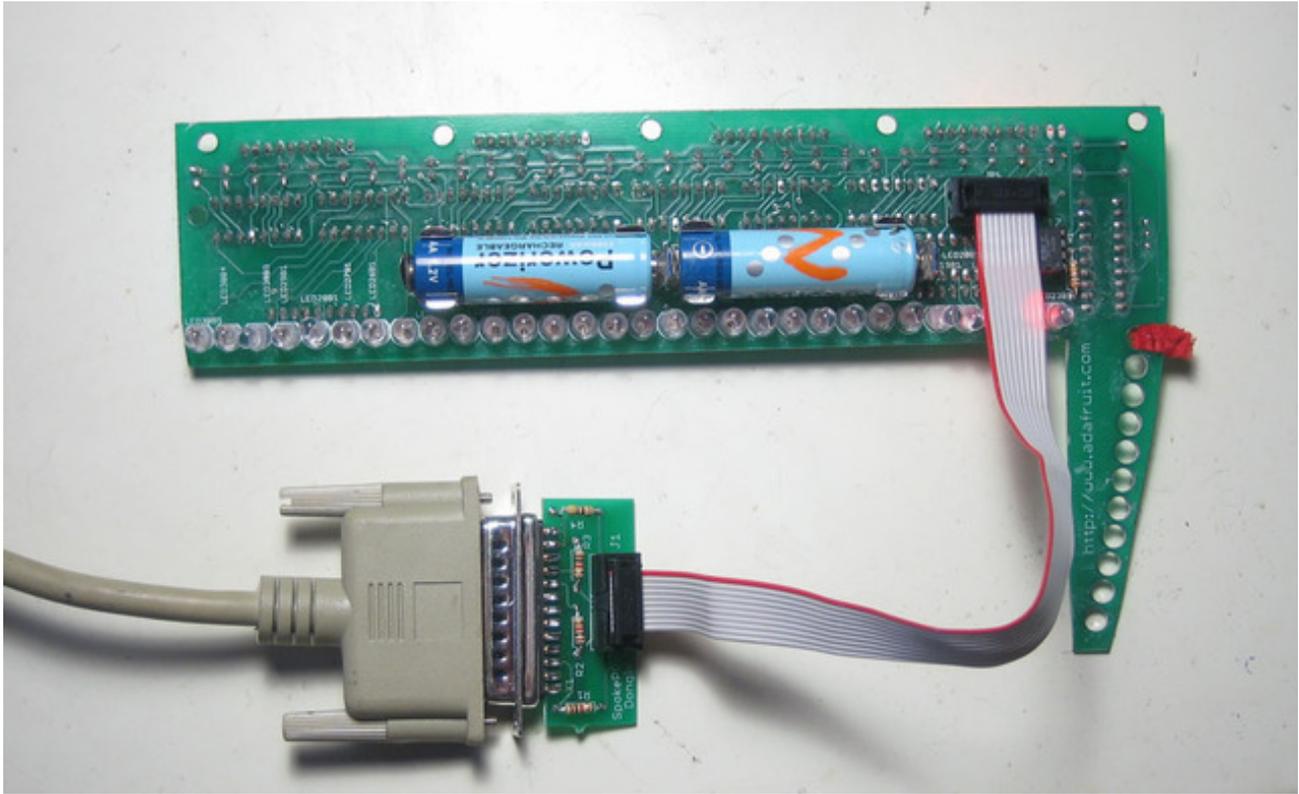


The software tries to figure out what the image should look like, but as seen above, the bitmap isn't aligned perfectly. To touch up the image, click on pixels (or click and drag to draw & erase).

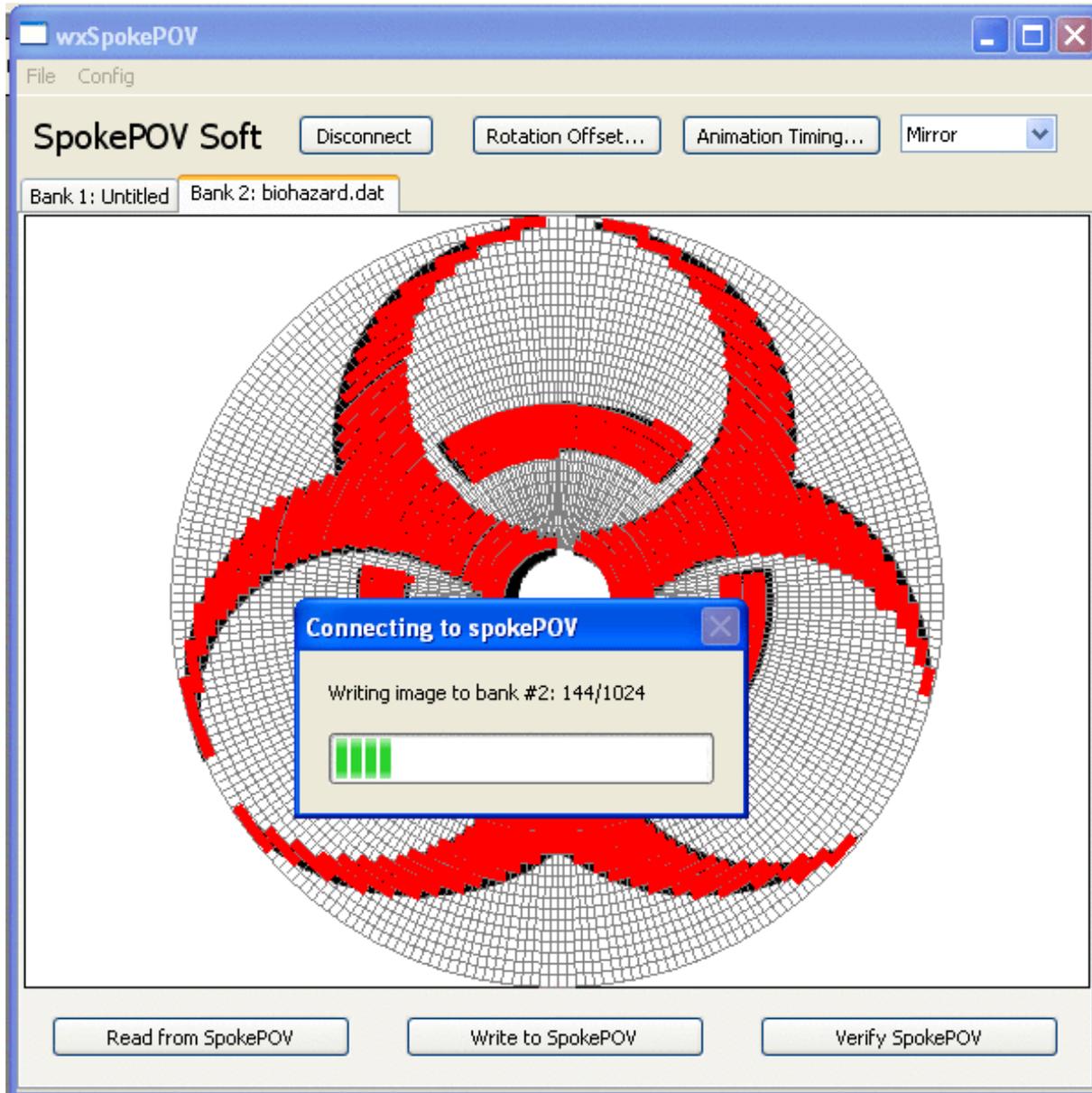
Now that you've done some edits, it might be good to save the image data file for later.



Now it's time to write the image data to your SpokePOV. Connect the dongle to your computer parallel/serial port. Perform a 'hard reset' on the SpokePOV (remove and replace a battery) and verify all the leds light up in order. Plug the programming cable into the SpokePOV. Press the reset button to make sure the SpokePOV is ready for upload.



Now click on the write button, and watch the ticker!



If all is well, the window will show the address incrementing and the LEDs on the SpokePOV will blink to show the progress. There are 1024 addresses, it should take 30-60s to write a full image. If you are using a USB/serial converter it may be much longer...You can manipulate the Comm delay to make it longer (less flakey) or shorter (faster).

You can also use the Read or Verify buttons to verify the contents, or read from the SpokePOV image bank into the panel.

Even if you only want to display 1 image, the SpokePOV will always go through all the banks. So if you have 4, you must upload the same image to all four banks!

Can't get it working? Don't worry, help is available in [the forums \(http://adafru.it/forums\)](http://adafru.it/forums)!

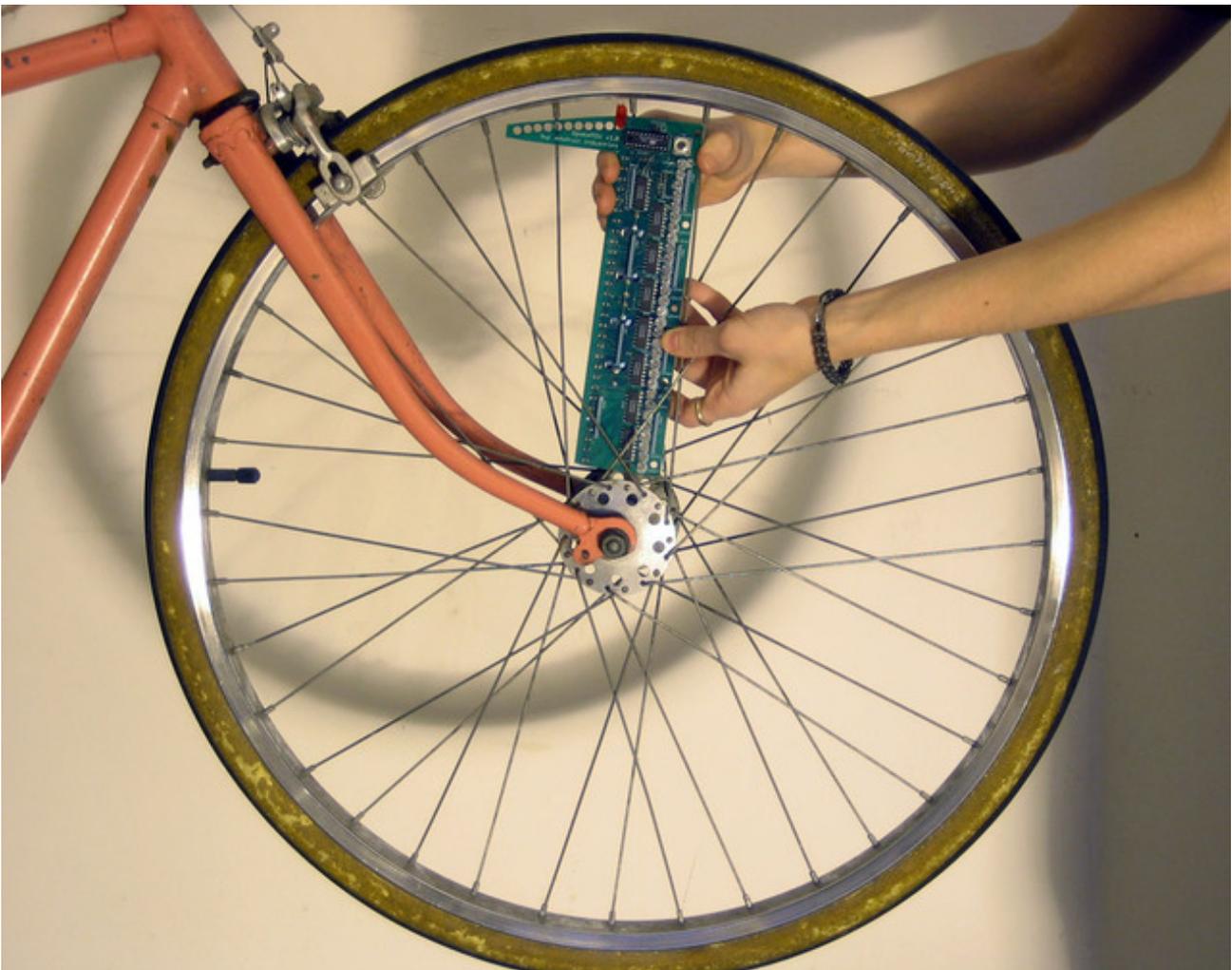
Install

User Manual

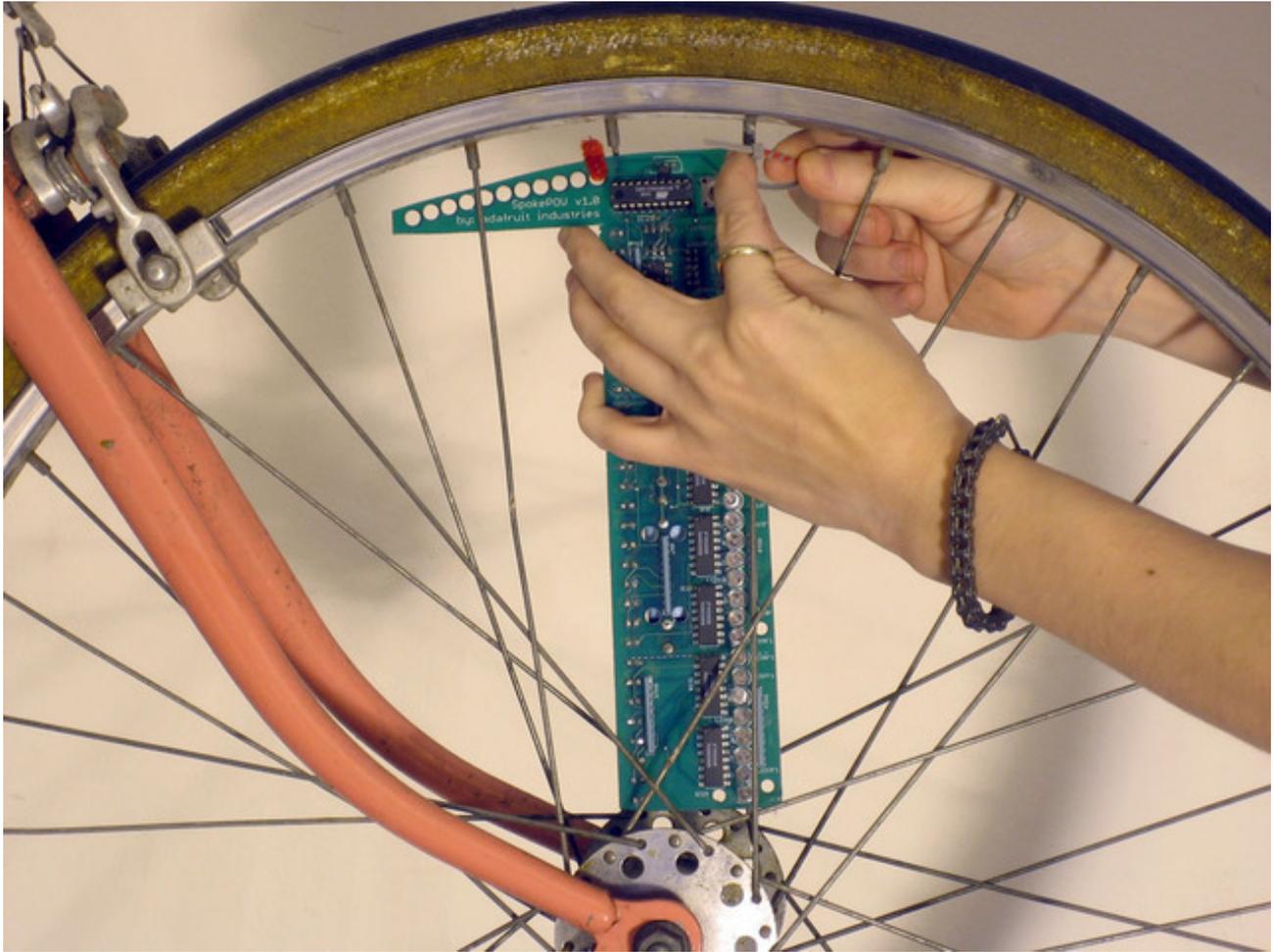
Installing your SpokePOV

Once you've [built and tested](http://adafruit.it/dhN) (<http://adafruit.it/dhN>) your SpokePOV, and [uploaded an image](http://adafruit.it/dhO) (<http://adafruit.it/dhO>), you can install it on your bike wheel.

The easiest place to install on a wheel is one of the large gaps between spokes. Your bike wheel may be a little different. The most important thing is that you don't install it so that the metal spokes scrape against any electronics so that they can short. As you can see in the picture below, I installed this SpokePOV so that it is inside the wheel: the spokes are all either to the side or in front where they only touch the LEDs (which is OK, they're plastic). When you clip the zipties, make sure to leave a little tail, if you cut them too short they are more likely to fail.



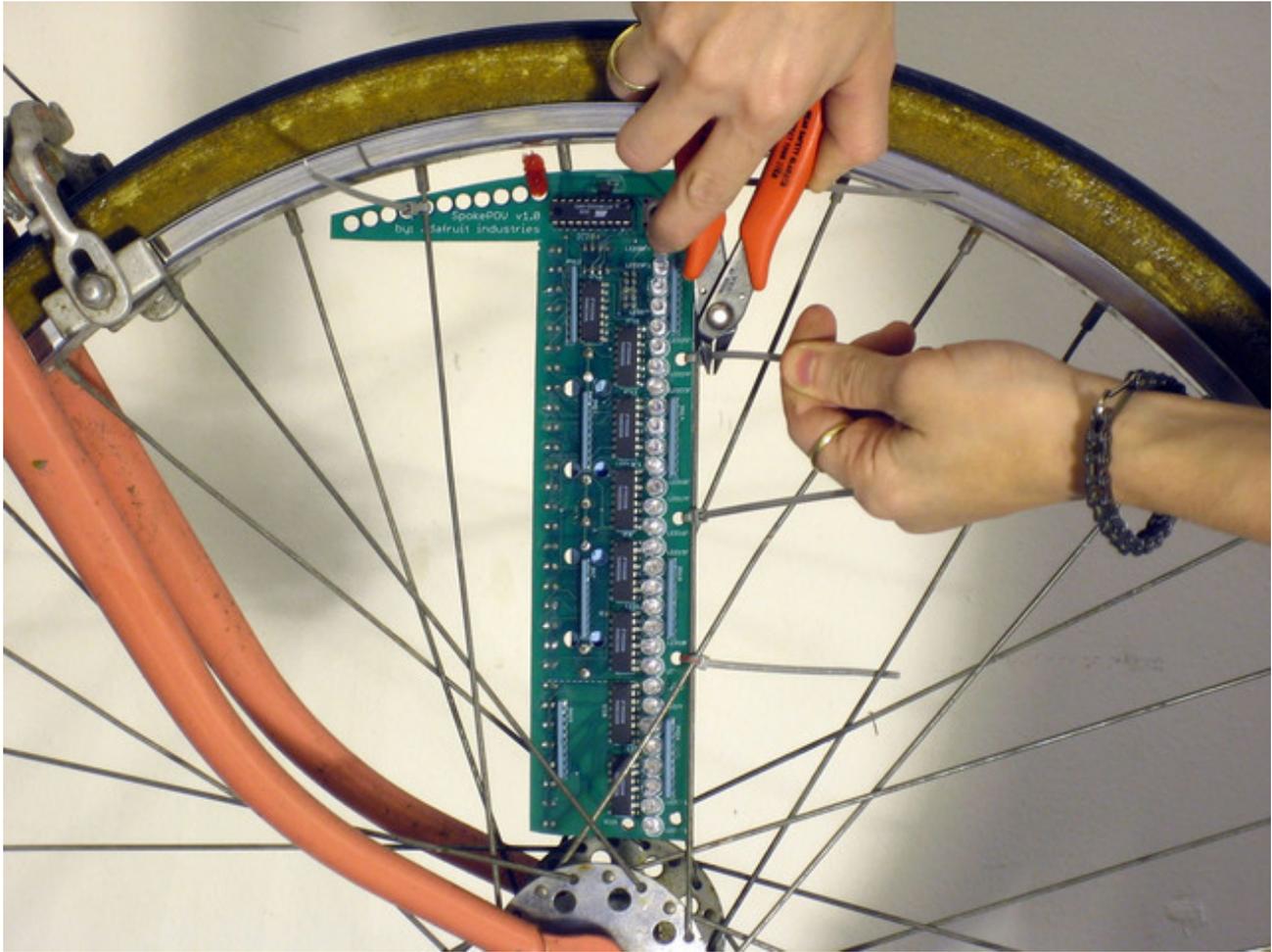
Use zip-ties (available at any hardware/crafts/office store) to attach the SpokePOV to the spokes.





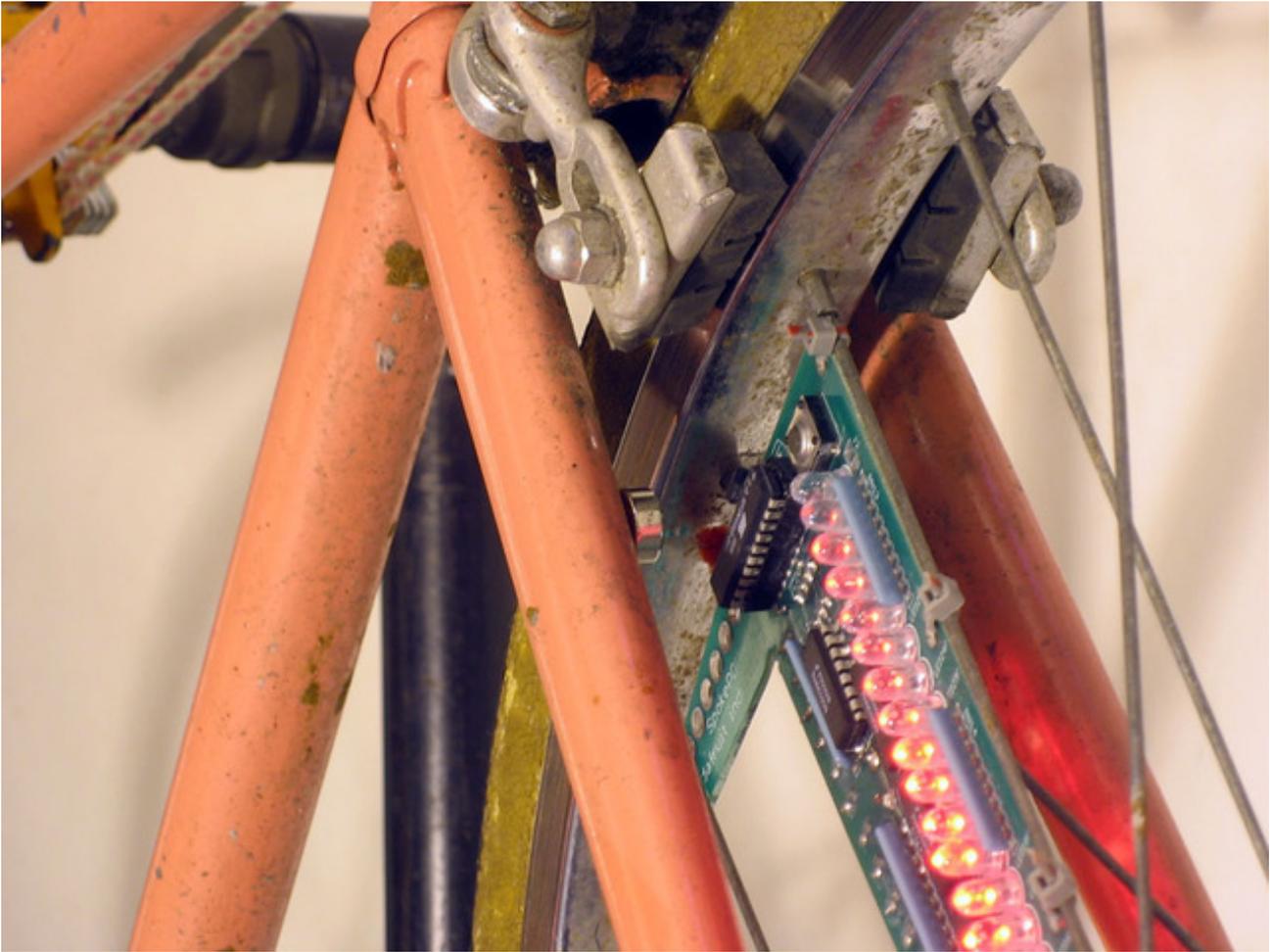








Attach the magnet to your frame. If you have a steel bike, this is really easy. You want to make sure that the south side of the magnet is facing out since the sensor is sensitive to the south side only. Try to put the magnet as close to the sensor as possible. Press the button to turn on the SpokePOV and try moving the wheel back and forth around the magnet, you should see the LEDs start to blink.



Now spin the wheel clockwise and stand back to see the image! (Turn off the lights for best results.)



Using the SpokePOV

There isn't much involved in using the SpokePOV. If you're worried about riding it in the rain, you can conformally coat the board (except for the button, which is waterproof, and the battery clips and header). Basically that means you coat the board in spray-on silicone which protects it against moisture. If you don't want to conformal coat, I'd suggest just keeping the SpokePOVs off while riding in the rain, and drying them off so that the batteries don't corrode. Unfortunately, there does not seem to be a simple, inexpensive way to make waterproof enclosures at DIY-level quantities.

The SpokePOV has three modes: off, idle, and display.

- **Off mode** is pretty much just off: the SpokePOV is consuming absolute minimal power. To turn it on, press the button & it will enter idle mode.

- **Idle mode** is a low power but active mode: the SpokePOV is waiting for the wheel to spin but it's not actually displaying any images. For example, it was just turned on. Or the wheel hasn't spun in the last few seconds. The SpokePOV will display one lit LED on each side to tell you it's still on. The LED itself may vary, as it is also used to debug the SpokePOV. If the SpokePOV doesn't receive any magnet signals in a few minutes, it will turn itself off to conserve power.
- **Display mode** is when the SpokePOV is receiving magnet signals and is displaying the stored image. This mode is probably what you bought the kit for. If it doesn't get any magnet signals for a few seconds it will go into idle mode.

The only interface to the SpokePOV is a single button. It works rather simply. Press the button quickly to reset the SpokePOV or turn it on if it's off. Press and hold the button for two seconds and then release to turn it off. If the SpokePOV is on there will be at least one LED lit. If the SpokePOV is off there will be no LEDs lit and it will not respond to the magnet until turned on.

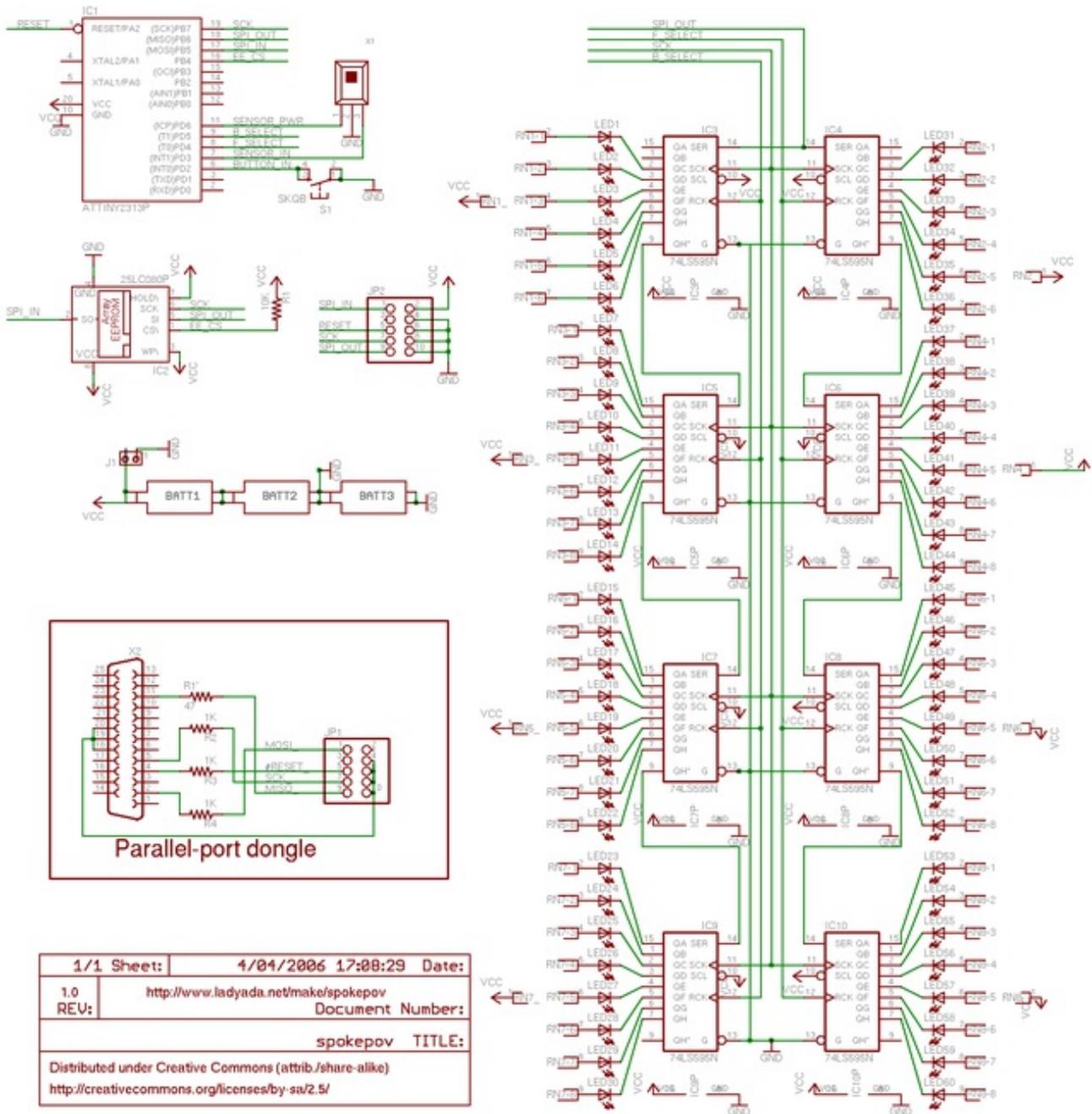
Design

Hardware design

Intro

SpokePOV works kinda like a cross between a LED ['propellor clock'](http://adafru.it/dhP) and a bike odometer. Like the odometer, there is a hall effect sensor and a magnet. Every time the magnet passes by the sensor, the sensor sends a pulse to a microcontroller. The microcontroller counts the time between pulses and uses that information to turn on the LEDs so that as the wheel rotates, the LEDs are always lit in the same location. Because the LEDs are really bright we see a streak when they move. Since they're moving really fast, the pattern of LEDs ends up looking like a persistent/enduring image. That's why it's called "persistence of vision", or POV for short.

Schematic



Microcontroller

The microcontroller used, the ATtiny2313, is one of my personal favorites. It's very low cost, has an 8MHz internal clock, runs as low as 1.8V, and has a bunch of pins. The microcontroller handles the button presses, sensor pulses, and reading data from the EEPROM into the latches.

When the microcontroller detects that there haven't been any sensor pulses in a few seconds, it powers down the LEDs. When there haven't been any pulses in a few minutes, the board goes into power-down mode: the sensor is turned off and the microcontroller is barely running. To turn it back on, a button press sends a signal to the microcontroller to let it know it's time to wake up and turn on the sensor.

Sensor

The sensor is a particularly sensitive unipolar hall-effect sensor. You can read about how these things work over at [sensorland \(http://adafru.it/dhQ\)](http://adafru.it/dhQ). When the **south**side of a magnet comes close to the front of the sensor, it sends a pulse to the microcontroller.

Button

The button is completely waterproof! Press the button for less than a second to reset the SpokePOV or turn it on if it's off. Hold down the button for 2+ seconds and release to turn the SpokePOV off.

EEPROM

The EEPROM (electrically erasable programmable read only memory) stores the image to be displayed. Since there are 32 LEDs (4 bytes) and 256 radial pixels, 1K of storage is needed. A larger eeprom could be used to display higher-resolution images or animations, etc.

To write the image data to the eeprom, a parallel-port dongle is connected to the board and the SpokePOV software is used.

Batteries

The default configuration is 2 x AA batteries. Alkaline batteries are best and will last a lot longer. However, I've found rechargeable NiMH batteries work pretty well. To figure out approximately how long your batteries will last use this equation: $(\# \text{ mAh in batteries}) / ((\% \text{ LEDs on}) * 60 \text{ LEDs} * 10\text{mA per LED})$. So for example, if you have 2000mAh AA rechargables, and about 1/4 of the LEDs are on in a picture, then it should last about 13.5 hours of constant (lit-up) use.

If you want to use some other sort of battery pack, or not use the clips, you can solder your wires to **J1**, which connects directly to power. You could also connect multiple SpokePOVs together this way and share one set of batteries. Note that your battery life will be a lot shorter and also there may be so much current draw that the batteries can't really handle it (like if all the LEDs are on). YMMV.

LEDs

The LEDs used are high-brightness (2000mcd) red 5mm LEDs. If you want to use green, blue, purple, white, etc. LEDs you'll have to install a third set battery clips and also cut a trace on the PCB. You may also have to use different value resistors. For example, if you want to use green LEDs and the datasheet says they have a forward voltage of 3.2V (it's usually 3-3.5V) and you're using 3 alkaline batteries with a nominal voltage of 1.5V each, the resistors should be: $((3 * 1.5V) - 3.2V - 0.5V) / .01 = 80 \text{ ohms}$. The .5V drop is from the internal latch offset and .01 is 10mA which is usually about how much you want going through the LEDs. Smaller resistance values = brighter LEDs but more power draw. If you're using 3 rechargables, you can try using the smallest resistor values you can get (10 ohms?)

If you want to use yellow, amber, orange LEDs, you won't need to put a third battery in since almost all of these have a forward voltage of 2.3 or less.

Programming header

The 10 pin header is used to upload images to the SpokePOV board and set variables (see the Software page for details) and can also be used to upload new firmware code to the microcontroller. The header pinout style is called "AVR ISP 10-pin", and is one of the prevailing standards for Atmel microcontrollers.

Programming dongle

When it's time to upload images, connect one end of the programming dongle to your SpokePOV and the other to the parallel port on your computer. The dongle can also be used as a general purpose AVR programmer (type "DT006") using the very popular AVRDUDE software (available for windows & linux/bsd/etc).

Gallery

Here are some photos from other people!

- [Flickr has a bunch \(http://adafru.it/dhR\)](http://adafru.it/dhR)
- [A foldingbike displaying a tree \(http://adafru.it/dhS\)](http://adafru.it/dhS)
- [Some abstract designs from Aneel \(http://adafru.it/dhT\)](http://adafru.it/dhT)
- [A hacked up version that displays RPM and Scrolling text on a rotating robot blade. \(http://adafru.it/dhU\)](http://adafru.it/dhU)
- [Some neat designs from theledlightingcompany \(http://adafru.it/dhV\)](http://adafru.it/dhV)
- [under's spinning stars \(http://adafru.it/dhW\)](http://adafru.it/dhW) ([here is the bmp file \(http://adafru.it/dhX\)](http://adafru.it/dhX))
- [Witchgers blue spokepov fun \(http://adafru.it/dhY\)](http://adafru.it/dhY)

Send me your photos or post them to flickr with the tag spokepov so I can link it up!

Download

Example images

In case you want them again, [here are the 'examples' used in the factory SpokePOVs](http://adafru.it/dhZ) (<http://adafru.it/dhZ>).

USB Driver (windows only!)

Here is a [driver for the USBtinyISP that says "SpokePOV Dongle"](http://adafru.it/di0) (<http://adafru.it/di0>) when its plugged in (otherwise identical)

C++ Software

Yay! I have rewritted the code to be much faster and much better with fewer errors and timeouts and nicer image handling. Please try it!

- **Version 1.4 with USB support for Windows** (<http://adafru.it/eje>)!
- Version 1.4 for MacOSX is [here](http://adafru.it/efb) (<http://adafru.it/efb>).
- v1.3 messed up serial support. let me know if you have any problems: should be fixed now. Thanks!
- [Version 1.2 for Serial/Parallel \(Windows/MacOS X\)](http://adafru.it/di1) (<http://adafru.it/di1>) (older version, not suggested)
- **giveio.exe** (<http://adafru.it/di2>), the windows software that allows spokepovsoft and other programs to access the parallel port (necessary if you use the parallel port dongle!), available for download from the pyParallel project on sourceforge.

One thing to note with the Mac version and serial dongle is that it may detect more than one 'port' for a single USB device. Say **/dev/cu.KeySerial1** as well as **/dev/cu.USA19H1b1P1** try both as I've gotten better results with one over the other.

wxPython Software

You can try this stuff, but it is old... it wont work nearly as well as the stuff above. I include it for your amusement only!

- [Zip file of v1.2](http://adafru.it/di3) (<http://adafru.it/di3>) for Windows users (now with serial support!)
- [DMG of v1.2](http://adafru.it/di4) (<http://adafru.it/di4>) for MacOS X users (for use with a usb/serial converter!) If you're using the PL2303-chipset usb adaptor on a MacOS X machine, [download this driver](http://adafru.it/aTU) (<http://adafru.it/aTU>) and install it, NOT the one that comes with the device!!!
- [giveio.exe](http://adafru.it/di2) (<http://adafru.it/di2>), the software that allows python and other programs to access the parallel port (necessary!), available for download from the pyParallel project on sourceforge.
- [Just the v1.0 python script, config file, and sample bitmap](http://adafru.it/di5) (<http://adafru.it/di5>) for Linux/FreeBSD/etc. users. You'll have to also have Python 2.4, wxPython and pyParallel installed!

Firmware

- V1.01 avr-gcc firmware and intel .hex file (<http://adafru.it/di6>)

A hacked version of the v1.01 firmware that blinks all the LEDs when the bike is 'stopped' (<http://adafru.it/di7>)

Hardware

- High rez [PNG of schematic](http://adafru.it/di8) (<http://adafru.it/di8>)
- Latest EagleCAD [schematic](http://adafru.it/di9) (<http://adafru.it/di9>) and [board file](http://adafru.it/dia) (<http://adafru.it/dia>)