

Guide for Fitting a Pi Zero W into a PS2 Slim

By Danial (Reddit: 67isd)



Change Log

Version	Date	Description
1.0	June 26, 2017	Initial version
1.1	June 28, 2017	Added safe reset and safe shutdown functionality to front button Added notes on setting aspect ratio
1.2	July 6, 2017	Changed formatting of document Added steps to support PS2 controllers (not fully completed)
1.3	September 10, 2022	Fixed wrong shell script filename extension in page 27 (for shutdown button)

Table of contents

Change Log	2
Introduction	4
Interior Layout	6
Materials required	7
Opening the case	9
Creating the common ground point	10
Getting the Sony A/V composite output to work	11
PWM Audio circuit	12
Configure the PWM audio in the operating system	15
Video circuit	16
Wire up the two front USB ports	18
Wire up the two PS2 controller ports (TO DO)	21
PS2 power LED and reset switch	23
Add code for the switch	25
Fit the power cable into the casing	29
Put everything together	30
Make sure everything works	32
Adjust the screen aspect ratio	33
Finished	34

Introduction

The following guide will show you how to fit a Raspberry Pi Zero W inside a PS2 Slim (preferably one that is already dead so that you don't waste a perfectly working one) to make it run RetroPie, utilizing some of the existing ports on the PS2.

Please note - this guide is based on the PS2 SCPH-70000 Series. I used an SCPH-77001. I'm not sure how the other versions of PS2 Slim will work out, but most of the steps should work, such as those for the A/V output and USB ports.

Throughout the guide I might refer to the Pi Zero W as the Pi or Pi Zero, just for shorthand.

DISCLAIMER: I am not responsible for any damage you do to your PS2 or other parts, or for an unsuccessful assembly. I've tried to get into as much detail as possible for the methods I used.

Features of this implementation:

- A low-power Pi Zero W (you can use a Pi Zero as well).
- This will use the PS2 proprietary A/V cable and jack to do composite audio and video.
- The 2 USB ports at the front will be used for controllers, keyboard, mouse, etc. (Bluetooth peripherals should also work with the Pi Zero W if you want to use wireless controllers)
- The PS2's reset button will power on/off or reset the Pi. The power LED on this button also works - it shows up as red but turns to green temporarily when pressed.
- The Pi's activity LED can be viewed inside the DVD tray. The micro SD can also be accessed from here.
- The PS2 power jack is removed to allow a Pi's micro USB cable for power.
- (TO DO: The 2 PS2 controller ports are supported)

Experience required:

- This guide assumes you understand how to use SSH, the command line and text editing on the Raspberry Pi.
- Competency with soldering and desoldering
- Able to check for connectivity with a multimeter

What it doesn't support:

- HDMI - I suppose this can be done if you use the Pi's mini HDMI jack and make a hole big enough at the back for a cable, but this project was intended for my CRT TV and its composite video.
- Rear ethernet adapter - I already have Wi-Fi on the Pi Zero W, and a USB ethernet adaptor would be too big to add to this. Perhaps using a Pi 3 with included ethernet port would've made it easier.
- The optical audio connector at the back of the PS2.

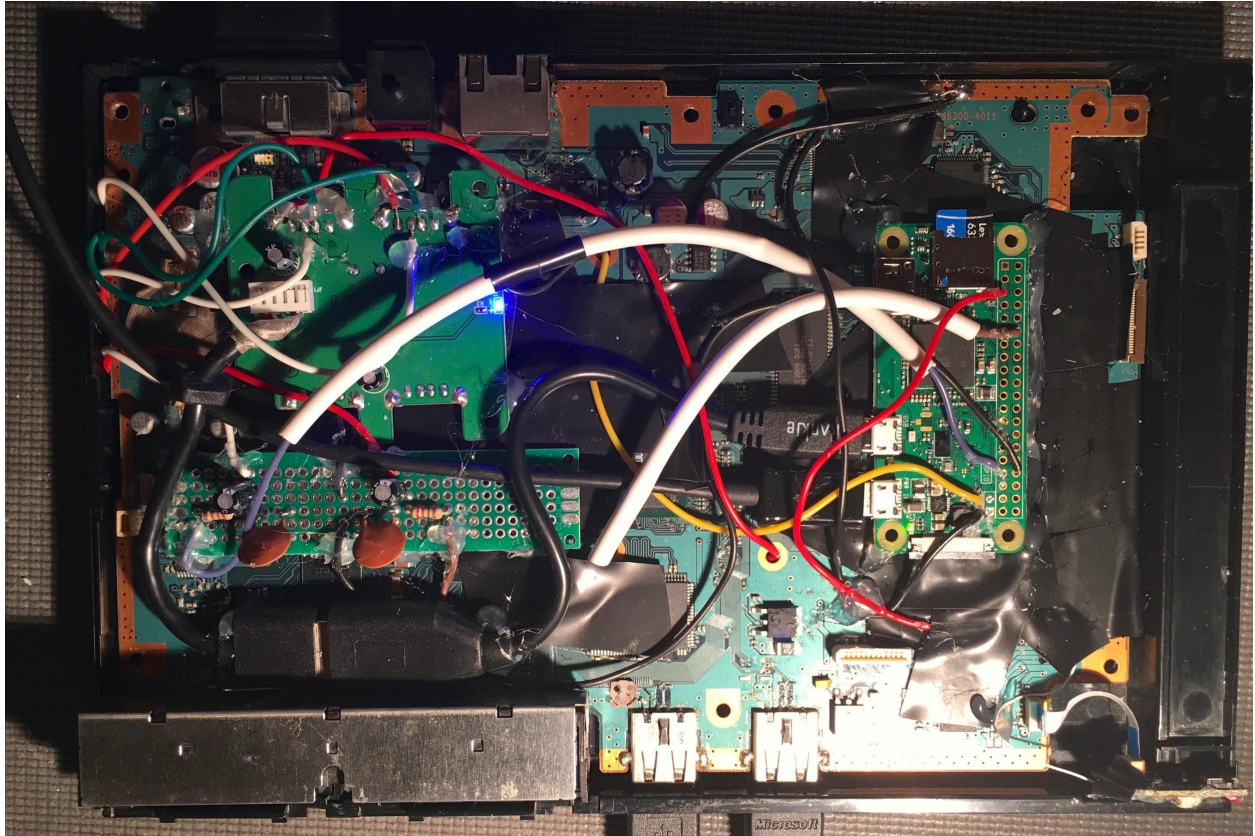
When compared to other mods where the Pi is using the casing of an old game console, this method does not discard the PS2 motherboard inside. The reasons I chose this route are:

- Every panel jack I needed was already mounted on the motherboard and so I didn't need to buy or solder any connectors.
- I wired the Pi's ground to the PS2 motherboard's ground to create a common ground point. This saved time and wiring material on the jacks as I didn't need to manually wire the ground for all of them to the Pi.
- The PS2 power LED and switch PCB (which is socketed to the motherboard) works automatically without extra work. No need to make your own LED and reset switch combo.

Of course, this method caused some issues, but I'll go over how I fixed them in the guide.

Interior Layout

We will refer to this layout throughout the guide:



The Raspberry Pi Zero W is on the right side. On the left is the USB hub (top), the audio circuit board (middle) and the USB cabling (bottom) that connects the hub to the Pi.

The back of the PS2 motherboard (not shown) contains wiring to the various ports, such as USB and audio/video, which will be detailed in the later sections.

Materials required

- A PS2 Slim you're willing to sacrifice, I strongly recommend an SCPH-70000 series if you want to follow this guide completely
- A Sony PS1/PS2/PS3 composite A/V cable. This is the one with red/white/yellow jacks to the TV.
- PS2 and/or USB controllers
- Pi Zero W with a RetroPie image installed on a micro SD card of sufficient size (I used 16GB)
 - RetroPie image download: <https://retropie.org.uk/download/>
- USB OTG adapter
 - <https://www.amazon.ca/gp/product/B00YOX4JU6>
- 2 x 10 uF radial electrolytic capacitors (mine had 50V rating, but at least 20V should be safe)
- 2 x 33 nF ceramic capacitors (mine had 50V rating but at least 20V should be safe)
 - <http://www.ebay.ca/itm/112056423295>
- 2 x 150 Ohm and 2 x 270 Ohm resistors - you can get a set of assorted sizes from a local electronics shop or eBay
- PCB prototyping board - I used 2x8 cm. The smaller the better.
 - <https://www.amazon.ca/gp/product/B06XC38GXZ>
- Micro USB cable with power adapter
- Small USB hub with at least three ports. First confirm that it works with a Pi Zero (I used an Insignia 4-port USB hub, model NS-PCH5420-C. There is also a similar NS-PCH5421-C.)
 - <http://www.bestbuy.ca/en-ca/product/insignia-insignia-4-port-usb-2-0-travel-hub-n-s-pch5420-c-ns-pch5420-c/10290275.aspx?>
- A two-port PS1/PS2 to PS3/USB controller converter (blue transparent model)
 - <http://www.ebay.ca/itm/331503112257>
 - NOTE: I have not implemented this yet, but the guide will explain the steps to a small extent.
- As well as the usual electronics modding tools, such as:
 - Lots of wire
 - Wire strippers
 - Solder
 - Solder wick or sucker/pump
 - Soldering iron
 - Screwdriver kit
 - Hot glue gun and some refill sticks

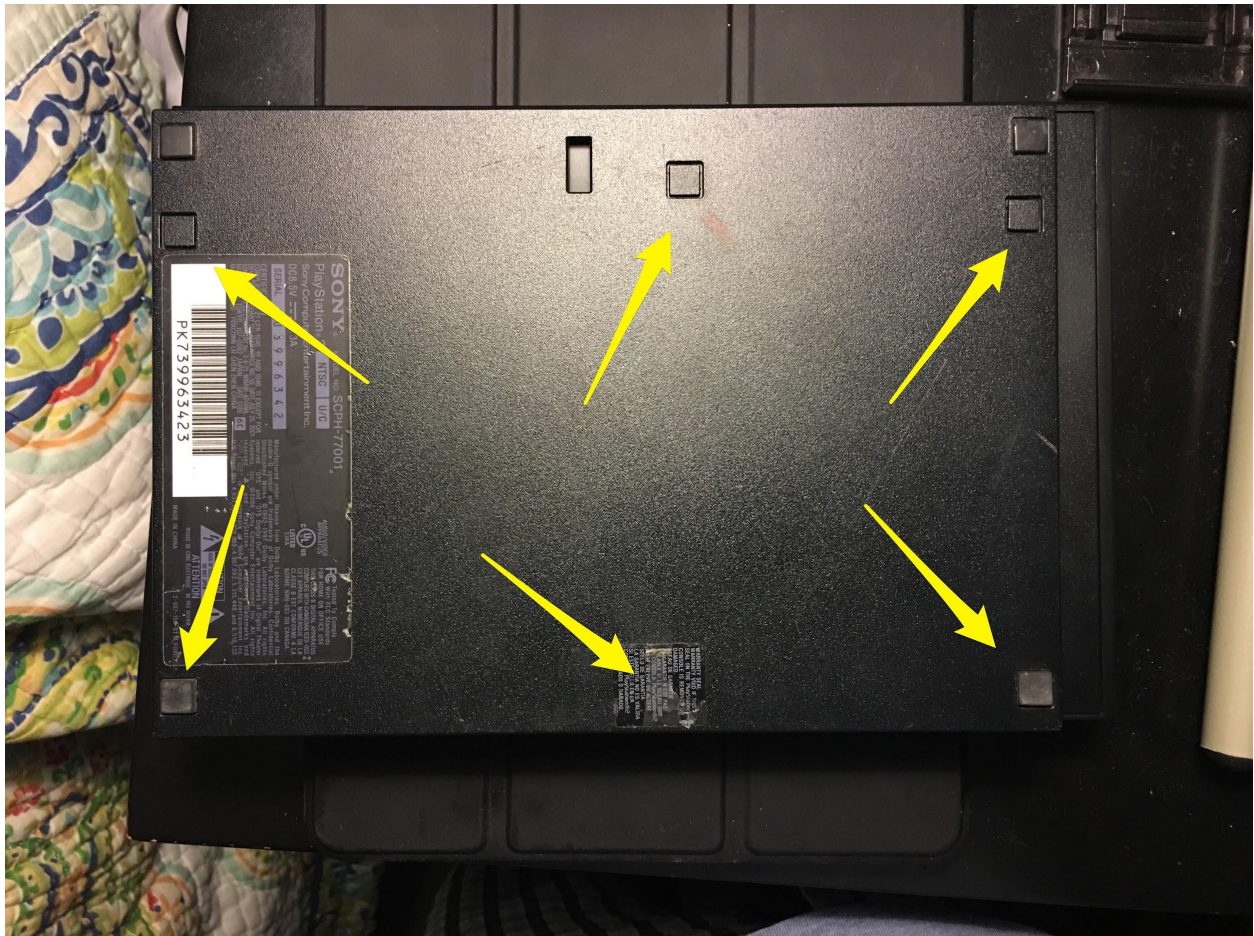
- I am not a soldering expert, so I used hot glue for strain relief wherever I felt necessary. Stranded wire may be less reliant on this, though I only used solid-core wire. Hot glue however will be required to hold down various components to the PS2.
- Electric tape
- Multimeter or connectivity tester to verify your connections
- Optional:
 - Hammer - to help open the USB hub if necessary, if it is not screwed together
 - Tweezers for picking up small pieces
 - X-Acto knife / box cutter / utility knife
 - Mini HDMI connector plus HDMI cable - for debugging and configuring your Pi, if you don't want to SSH.

Now let's get started...

Opening the case

Remove the pads on the bottom to access the screws. You can use your fingernails or a plastic tool. Specifically, you must remove:

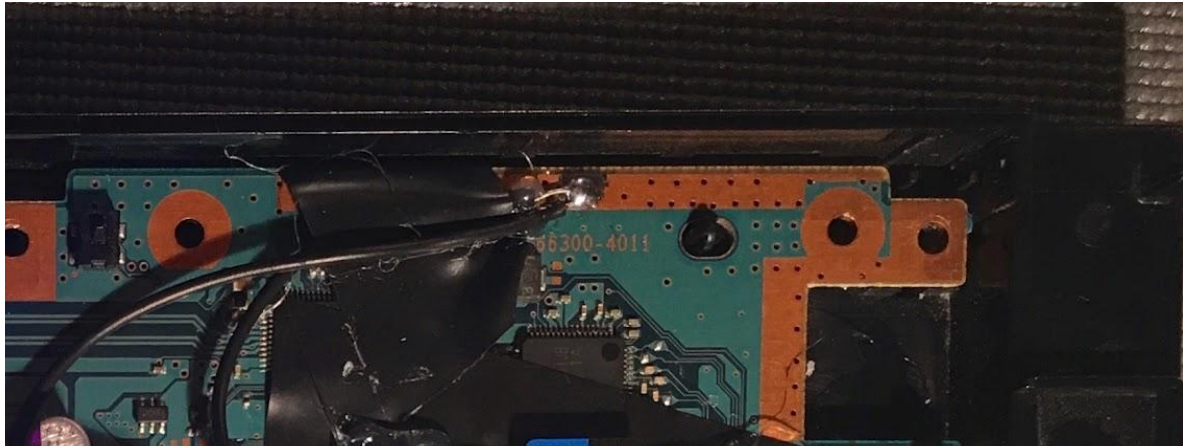
- At the front: Three plastic pads
- At the rear: Two rubberized pads and one plastic pad



Unscrew the DVD laser followed by the top and bottom metal shielding. Remove the transparent plastic sheet at the bottom as well. We won't need any of these parts anymore; just keep the top and bottom black halves.

Creating the common ground point

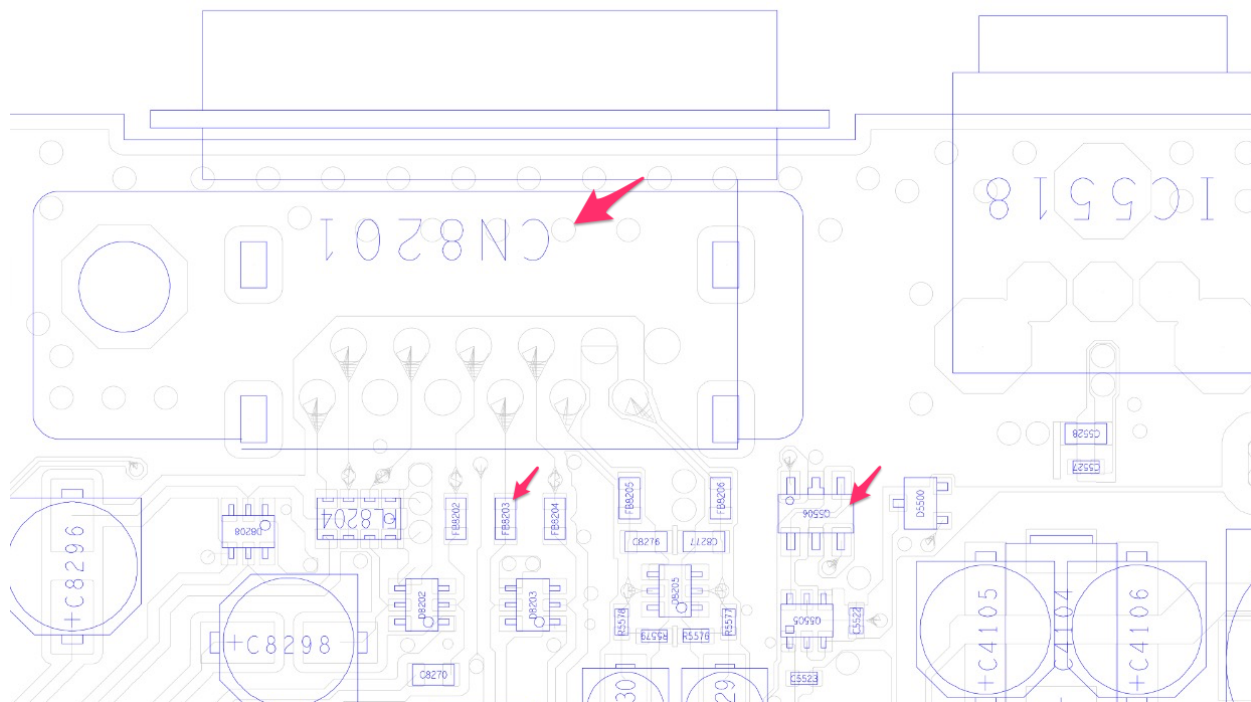
Solder one of the ground pins on the Pi to the PS2 motherboard's ground so that they share a common ground. You can use a metal spot on the motherboard's border if you are sure it is the ground. In the following picture I used the spot at the top right of the PS2 motherboard for ground (black wire).



Getting the Sony A/V composite output to work

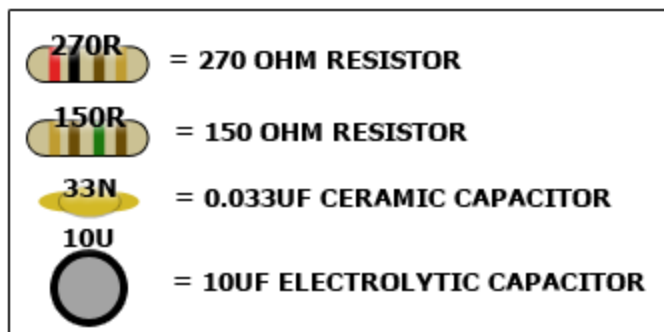
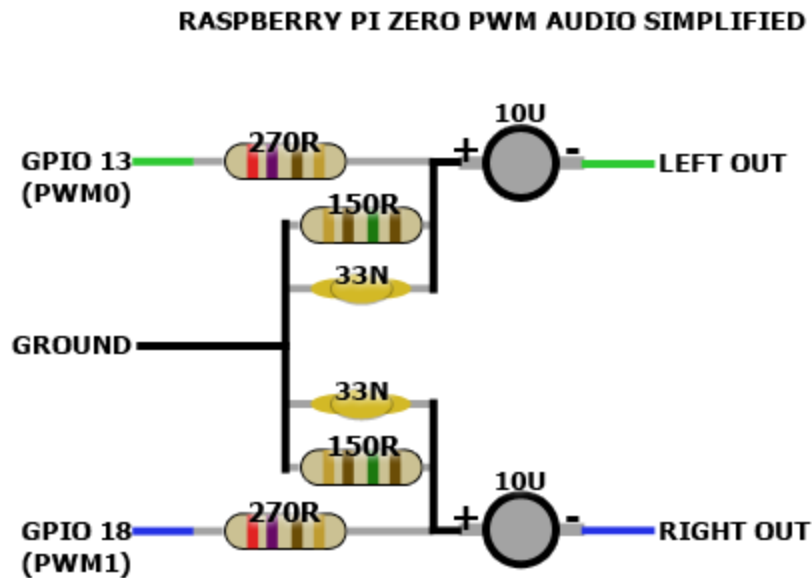
Here is a diagram from the PS2 SCPH-70000 Series service manual, which we will refer to later. The A/V connector is labeled CN8201, and two other components of note are Q5506 and FB8203:

BITED/GH-035 BOARD(SIDE A)



PWM Audio circuit

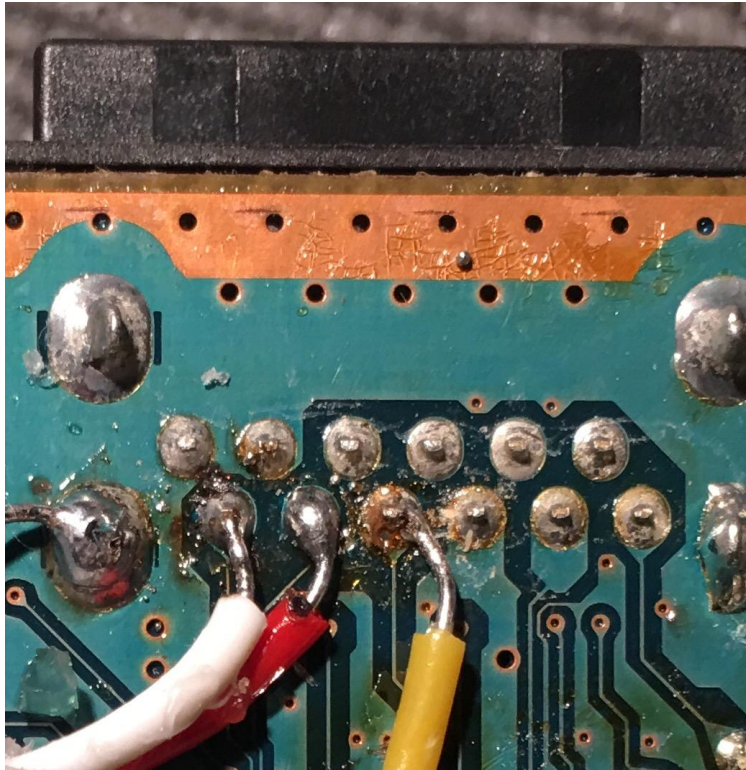
First create the PWM audio circuit on the PCB board you've provided. The following diagram is taken from <http://sudomod.com/forum/viewtopic.php?t=480>.



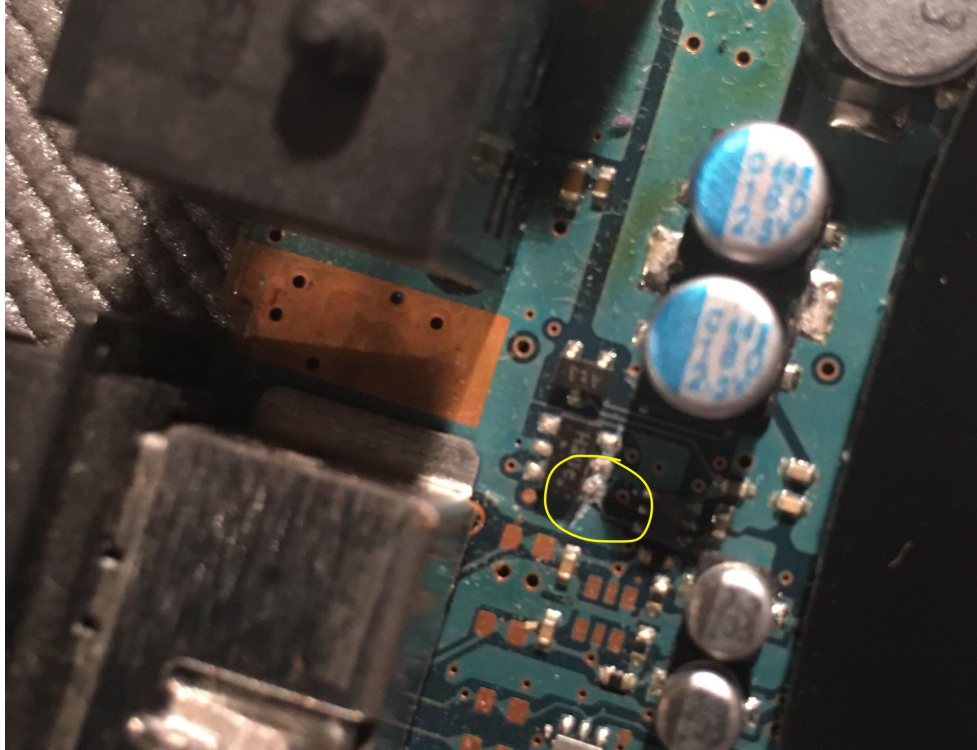
(As an alternative to the PWM circuit, you could use a USB sound card, but only if it will fit. I haven't tested this.)

Connect the left side of the PWM audio board to the the Pi's ground and GPIO 13 and 18.

On the right side of your audio board, connect the left channel output (white wire) and right channel output (red wire) to the AV connector at the back of the motherboard, as shown below. You don't need to wire the audio board's ground to the motherboard's A/V grounds, as we already have a common ground connection between PS2 and Pi. You can verify the connection with your multimeter. (The yellow wire shown is part of the video connection, which we will visit later.)



Turn the motherboard over and cut the legs off of transistor Q5506 from the schematic shown earlier and in the picture below. It's next to the A/V connector. Use an X-Acto knife or desolder it. Specifically, you need to cut at least the lower left leg from the traces, otherwise the sound will come out scratchy after the USB portion gets connected. Thus by doing this we cut our custom audio circuit from the rest of the PS2 motherboard:



Configure the PWM audio in the operating system

Restart the Pi, go into `/boot/config.txt` and add these two lines to initiate the PWM audio:

```
dtoverlay=pwm-2chan,pin=18,func=2,pin2=13,func2=4
disable_audio_dither=1
```

The second line removes interference in the video when sound is started.

NOTE: If you used a USB sound card instead of the above PWM audio, you'd need to skip the above lines and instead do this:

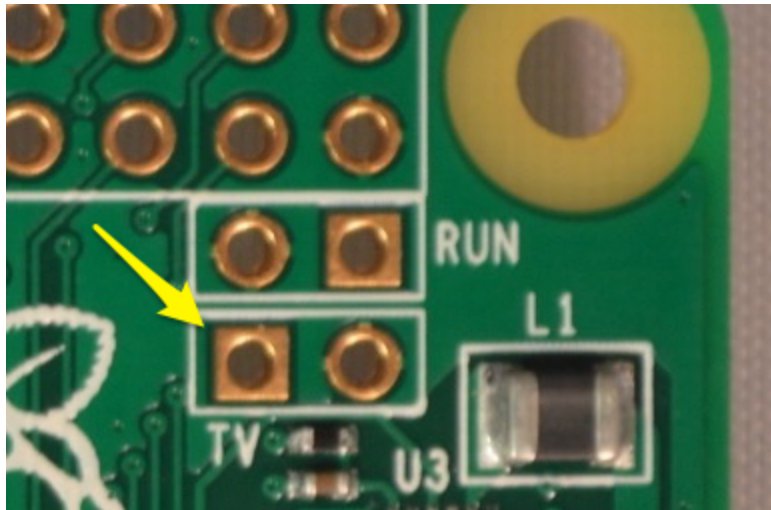
1. Create a file called `/etc/asound.conf`
2. Add these three lines to it:

```
defaults.ctl.card 1
defaults.pcm.card 1
defaults.timer.card 1
```

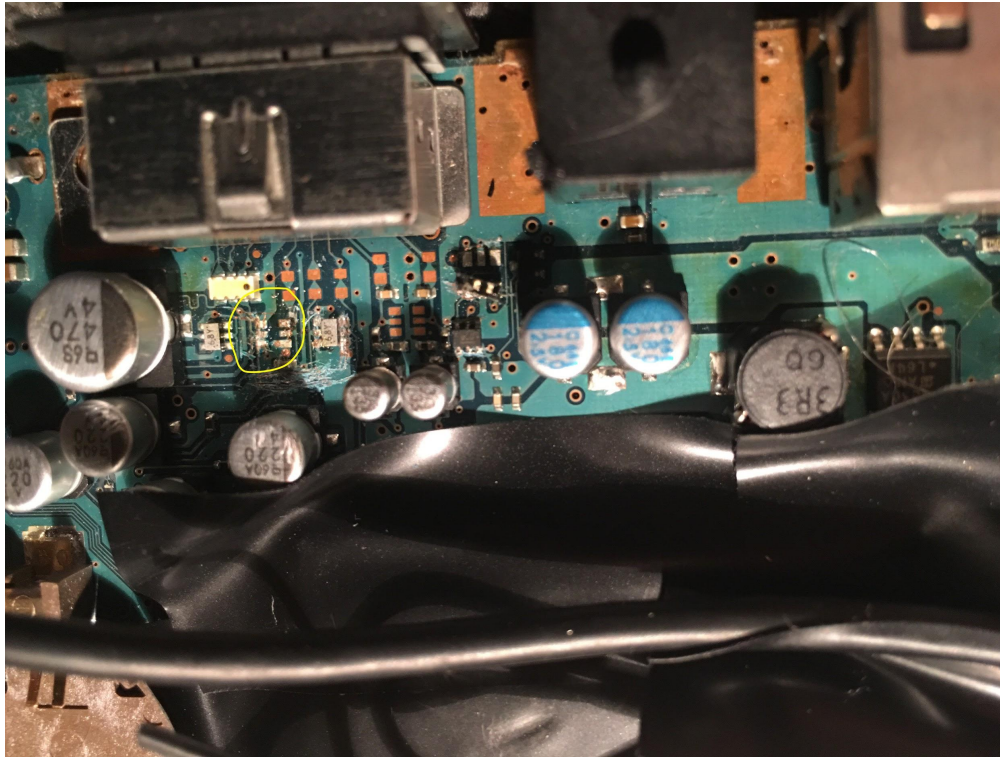
Skip those these two steps if you're not going to use a USB sound card.

Video circuit

Now wire the video connection from the Pi's TV pin to the A/V connector on the PS2 motherboard (the yellow wire shown in the earlier pic. Again, we don't need to wire a ground. The TV pin is shown below:



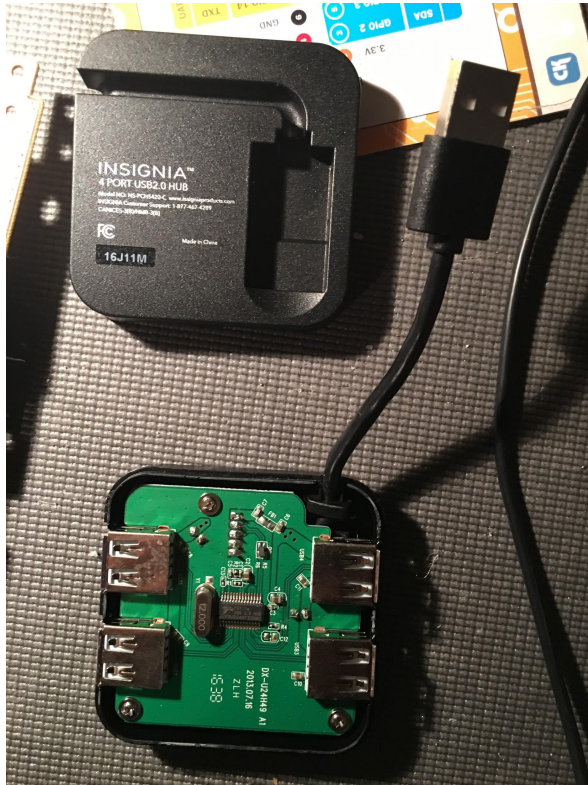
Next, cut/desolder the ferrite bead component FB8203 shown in the schematic earlier and circled in the photo below. It is below the video connection. This prevents awful static in the video that will happen after connecting the USB portion later. This will separate the video trace from any connections from the rest of the motherboard.



Now connect the Sony A/V component cable, turn on the Pi and test the video output to ensure it shows up.

Wire up the two front USB ports

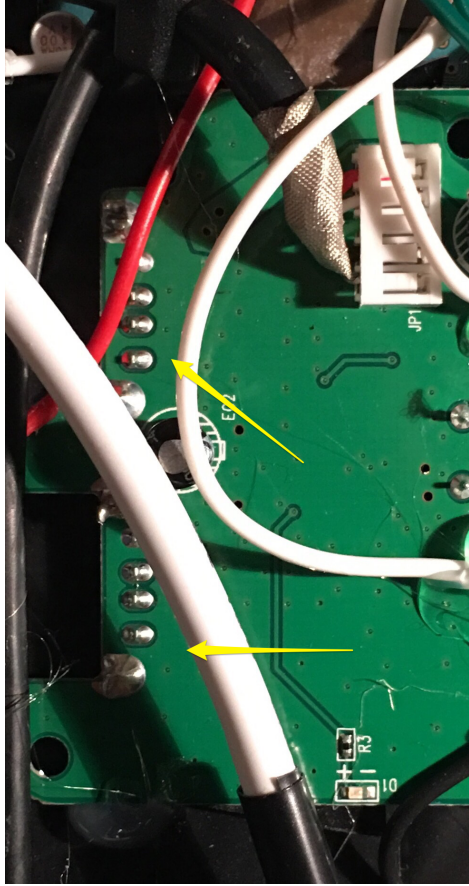
You will need to take apart a USB hub. In my case, I gently tapped the side with a hammer to unseal the plastic halves. Then I unscrewed the PCB from its casing.



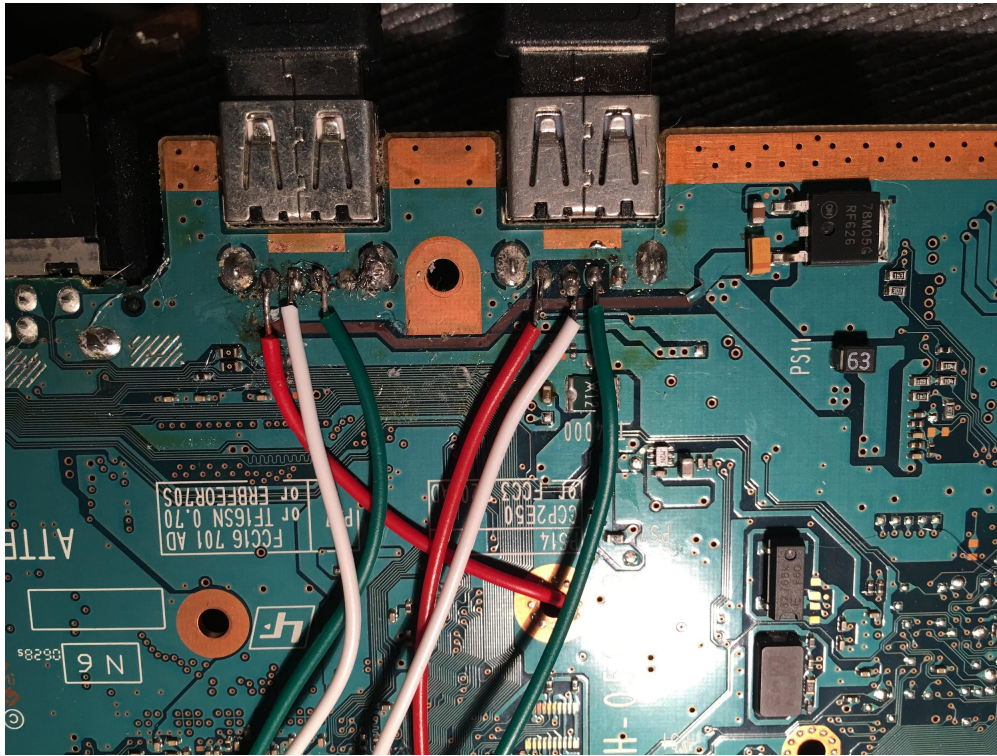
I gently tore off the USB sockets from the hub, but it would be safer to desolder them. We will only need to use the sockets' solder points on the hub.

Plug the hub into the OTG adapter.

Now make sure you use the correct side of the hub for the wiring - you can tell by comparing the orientation of the USB jacks on the hub to those on the PS2, or by finding the side of the hub that has the solder points/pins, as shown in the following image:



Pick two USB ports on the hub. For each port, wire the 5V, D+ and D- solder points (pins 1, 2 and 3 respectively) to the equivalent solder points on a USB port on the PS2:



Again, you don't need to wire the ground on the PS2's USB jacks, since they already connect to the common ground. If you're wondering, the 5V comes from the Pi's OTG USB connection.

Next, you need to ground your hub's PCB. Pick one of the hub's USB ports, and connect its ground pin (which is pin 4, or the fat pin next to pin 1 or 4) to the common ground you created on the PS2. Confirm with your multimeter that that the ground on the hub's other USB sockets are also grounded to the PS2 (they shouldn't require extra wiring).

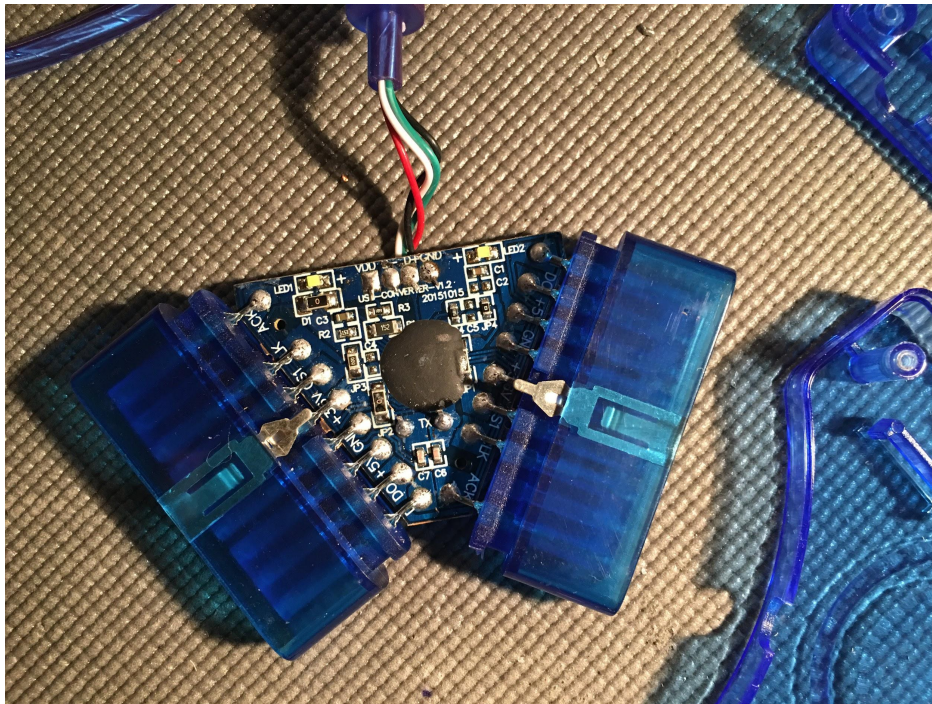
If you turn on the Pi, the USB ports should now work with your peripherals.

Wire up the two PS2 controller ports (TO DO)

This section of the guide has not been completed. I actually have not implemented this yet because I had damaged the converter PCB. I will update this section with photos as soon as I receive a new PCB and I complete the work.

NOTE: This section could be optional if you do not wish to use PS2 controllers or if you think it will be too difficult. The controller pins for the blue converter's PCB are soldered next to traces that can be **easily torn off** if you are not careful enough.

To start, unscrew the halves of the converter's casing so that you can access the PCB:

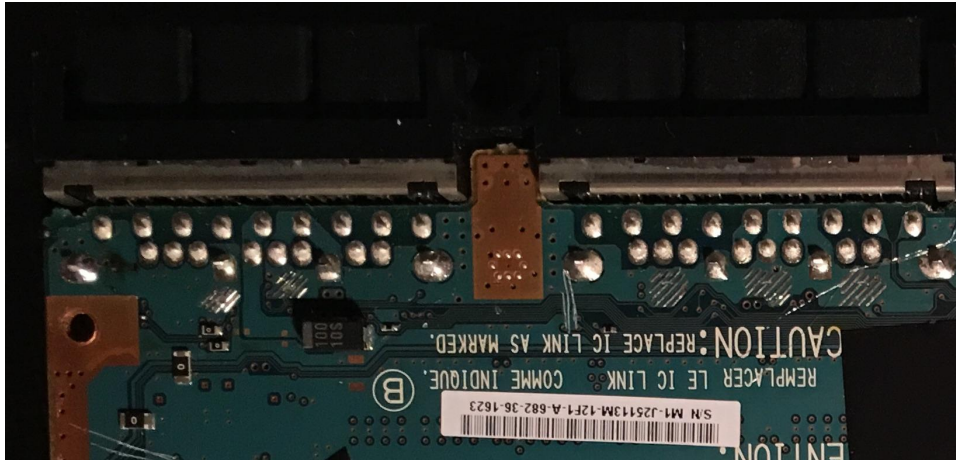


Now the hard part. Desolder the pins for the blue sockets from the PCB, or instead you could try to cut the pins with a small wire cutter. There should be 16 total pins to remove (8 for each controller).

Then cut the converter's USB connector so that there remains a USB cord with the 4 wires coming out of it (black, red, white, green).

Wire the black wire of the USB cord to the common ground on the PS2, so that the converter PCB is grounded. Wire the other three wires on the USB cord to the pins on one of the available USB ports on the hub (red = pin 1, white = pin 2, green = pin 3).

Turn the PS2 motherboard over and wire the top row of pins of the two controller ports to the corresponding pins on the converter:

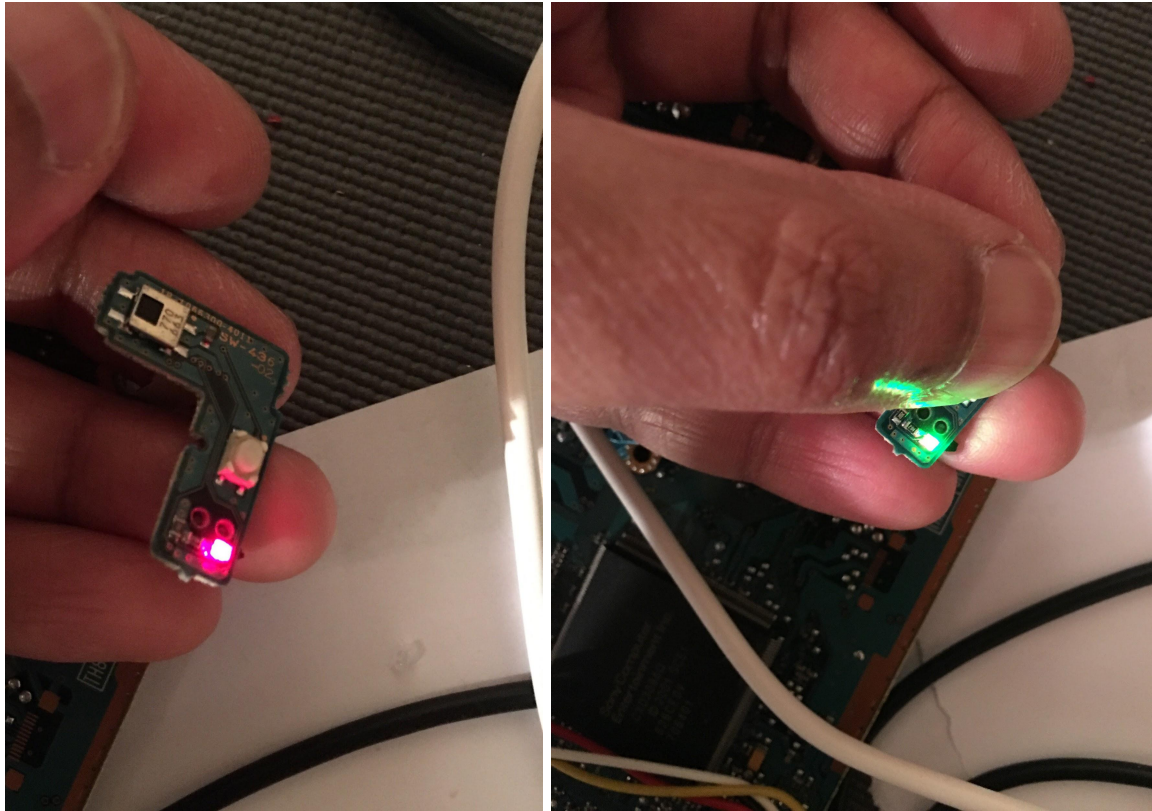


Plug in your PS2 controllers and turn on the Pi. Use your USB controller or keyboard to re-configure the controllers from the menu (press Start on the USB controller to see the menu). It should detect the PS2 controllers and allow you to configure them.

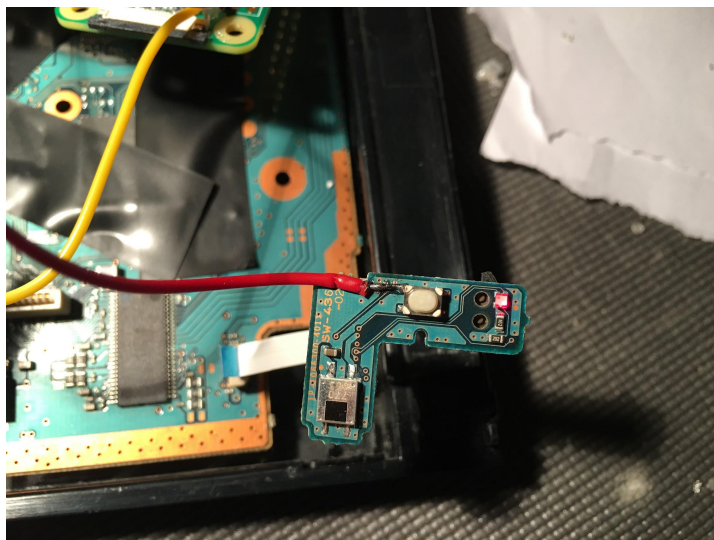
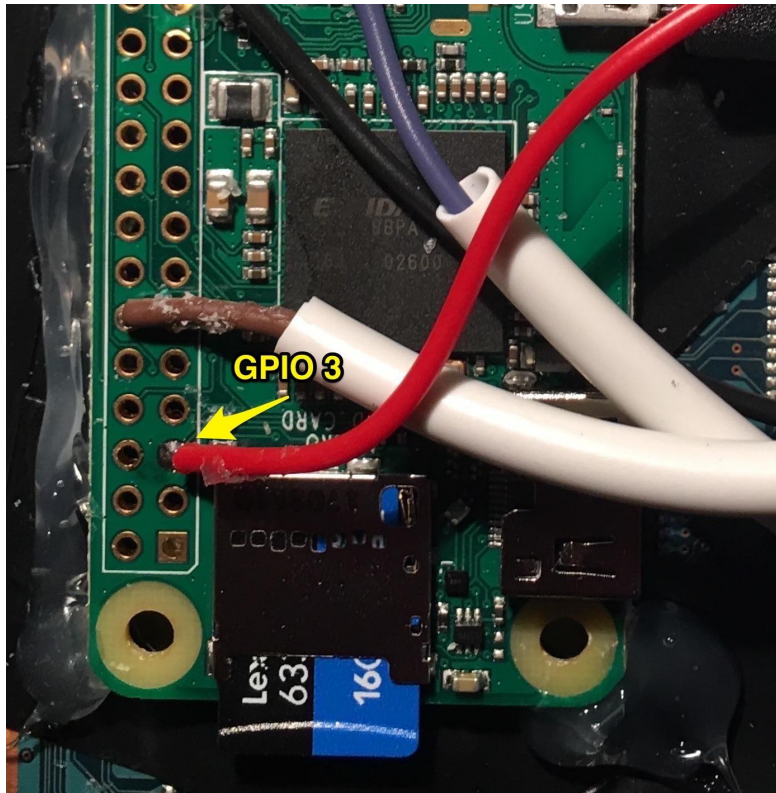
Test the PS2 controller on both ports to ensure the ports are working.

PS2 power LED and reset switch

After wiring the USB and turning on the Pi, the PS2 red status LED should automatically start working due to power received from the 5V wire in the USB. The LED PCB actually receives 3.5V. It turns green for a second when you press the tactile switch next to it:



Now we want to wire GPIO 3 (Pin 5 on the Pi Zero) to this tactile switch. Wiring the switch will require a very careful hand. Connect GPIO 3 on the Pi to one of the top two pins (left or right) on the tactile switch. Be careful you do not damage the delicate switch! You might need to solder a thinner wire to the switch instead and then add some hot glue to ensure it stays put.



Add code for the switch

We want this switch to support these three functions:

1. Safe reset - closes all processes before doing a reset (short press)
2. Safe shutdown - closes all processes before shutting down (long press)
3. Wake - turn on the Pi after it has been shut down

On the Pi Zero, GPIO 3 has an alternate/special function, where if you ground it, it will wake the Pi. There is no extra code required to do this, as the switch will connect GPIO 3 to Ground. However, for any GPIO pin to have reset or shutdown functionality, we need to create a python script that performs this task.

We will base our reset/shutdown implementation from this website:

<https://howchoo.com/g/mwnlytk3zmm/how-to-add-a-power-button-to-your-raspberry-pi>. The only difference from his instructions is that I modified the python script to check for both a regular/short press (safe reset) and a long press (safe shutdown). A long press will be 1.5 seconds. The shutdown initiates while the the finger is still pushing down on the button. I will include all of the instructions in the following page.

So, start an SSH session to the Pi or connect a keyboard and monitor/TV to it. Next we'll add some code to the device:

1. Create the Python script that waits for a button press and takes action:

```
sudo nano /usr/local/bin/listen-for-shutdown.py
```

NOTE: You must use the TAB key for valid indentation in Python:

```
#!/usr/bin/env python

import RPi.GPIO as GPIO
import subprocess
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(3, GPIO.IN)

# Wait for the button to be pressed
GPIO.wait_for_edge(3, GPIO.FALLING)

# Start timer to check for how long it was held down
startTime = time.time()
time.sleep(0.2)

# Set how long we want a long press to be
longPressSec = 1.5

# If held down for a long press, then shut down
while GPIO.input(3) == GPIO.LOW:
    time.sleep(0.01)
    elapsedTime = time.time() - startTime
    if elapsedTime >= longPressSec:
        subprocess.call(['shutdown', '-h', 'now'], shell=False)

# Otherwise, wait till GPIO comes HIGH and then reboot
subprocess.call(['init', '6'], shell=False)
```

Then, make this script executable:

```
sudo chmod +x /usr/local/bin/listen-for-shutdown.py
```

2. Create a shell script that will start/stop the python script as a service:

```
sudo nano /etc/init.d/listen-for-shutdown.sh

#!/bin/sh

### BEGIN INIT INFO
# Provides:          listen-for-shutdown.py
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
### END INIT INFO

# If you want a command to always run, put it here

# Carry out specific functions when asked to by the system
case "$1" in
  start)
    echo "Starting listen-for-shutdown.py"
    /usr/local/bin/listen-for-shutdown.py &
    ;;
  stop)
    echo "Stopping listen-for-shutdown.py"
    pkill -f /usr/local/bin/listen-for-shutdown.py
    ;;
  *)
    echo "Usage: /etc/init.d/listen-for-shutdown.sh {start|stop}"
    exit 1
    ;;
esac

exit 0
```

Then, make this script executable:

```
sudo chmod +x /etc/init.d/listen-for-shutdown.sh
```

3. Register the shell script to run on boot:

```
sudo update-rc.d listen-for-shutdown.sh defaults
```

4. Reboot the Pi, or if you want to try it right away, start your new service now like this:

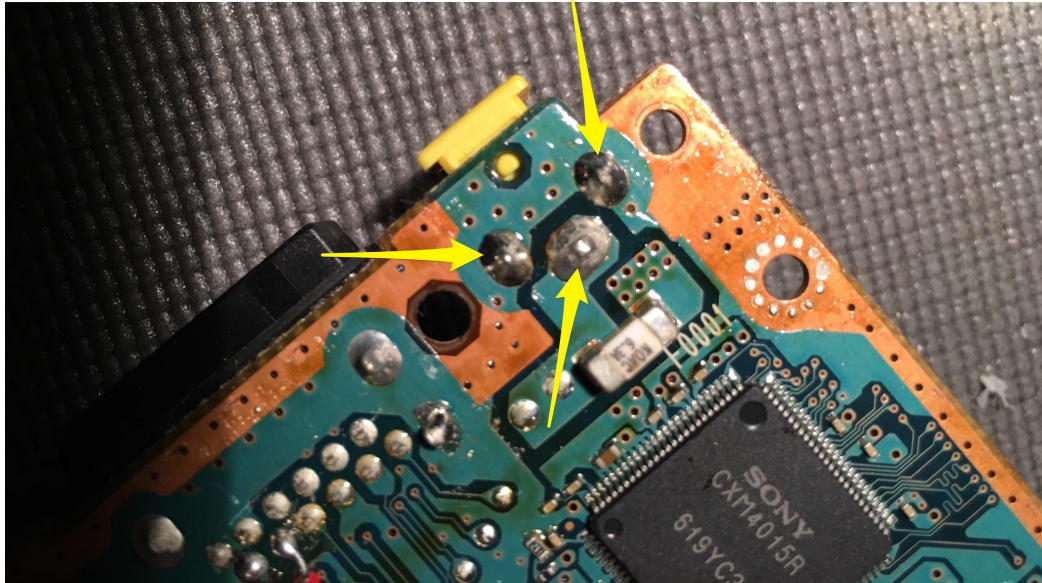
```
sudo /etc/init.d/listen-for-shutdown.sh start
```

Next, test what you have done:

1. Try a normal press/release of the front button and ensure a safe reset occurs.
2. After the Pi has finished restarting, hold down the front button for a little while to test the safe shutdown.
3. After the Pi is shut down, press the front button to wake up the Pi.

Fit the power cable into the casing

Desolder the PS2's power jack at the back. When you put the black PS2 shell together you can now use the vacant hole for the Pi's USB power cable:



Put everything together

Arrange all the pieces internally the way it is shown at the beginning of the guide. You'll need some electric tape padded under the Pi as well as under other components you added to the PS2 motherboard.

Place the Pi where the DVD laser used to be, so that when you open the DVD you can see the Pi activity light and can access the SD card:



Close everything up and put the six screws and pads back on the bottom.

This is what the back looks like, where the power jack used to be:



Make sure everything works

Test the DVD tray eject button and make sure the wires inside do not get in the way of the latch when you close the tray.

Plug in the A/V cable and the USB controllers. Turn on the Pi. Ensure the video works.

Ensure the PS2 front LED lights up red. When you press the PS2 reset button, make sure the LED changes to green for a second and the Pi resets.

Test the USB controllers on ports 1 and 2.

Ensure a game comes on and sound plays. If volume is too low, you can go into the Emulation Station sound settings (press Start) to increase the maximum volume to 100%.

Adjust the screen aspect ratio

We now want to change the global aspect ratio of the classic consoles (NES, SNES, Genesis, Master System, Atari, etc.) to be 4:3 to scale perfectly onto the CRT screen. To do so, navigate to the RetroPie tab in Emulation Station and start up RetroPie Setup. Then select the following:

1. Configuration / Tools
2. Configure basic libretto emulator options
3. Configure default options for all libretto emulators
4. Aspect Ratio (0 - 4:3)
5. OK

Test a game and ensure it fills the screen without stretching. For example, Golden Axe on the Sega Genesis:



Finished

Hopefully you're done!

