ELIXIRCONF™ 2017

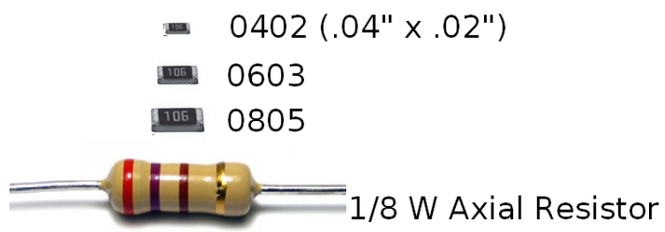
Nerves Training
Part 2: Basic I/O

Goals

- Basic electronics and circuits
- Introduction to the hardware we'll be using
- Hooking it up and trying it out

Resistors

- A device that resists the flow of current
  - Follows Ohms Law: V = IR
    - V = volts; I = current (Amperes) ; R = resistance (Ω)



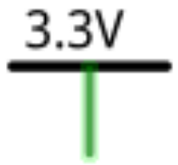
Diodes

- A device that allows current to flow one way
  - Usually very low resistance in the forward direction
  - Small voltage drop (0.2 V or more)

Light Emitting Diodes

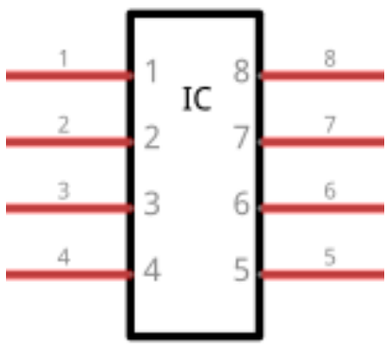
- Diodes intended for emitting light when current flows through them
  - Voltage drop depends on color
  - Too much current flowing through breaks them

Other symbols on schematics



Positive voltage from power supply

Ground, GND, 0 Volts



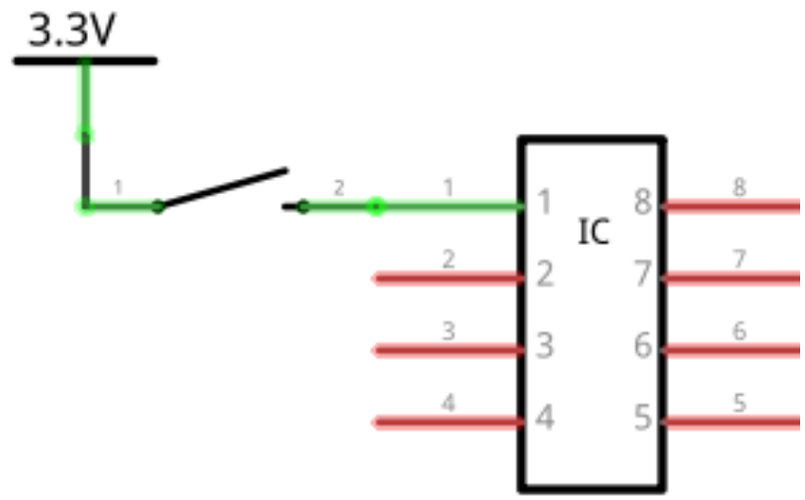
An integrated circuit or anything with lots of pins

A switch or button

Transistor

Basic circuits that you'll run into quickly

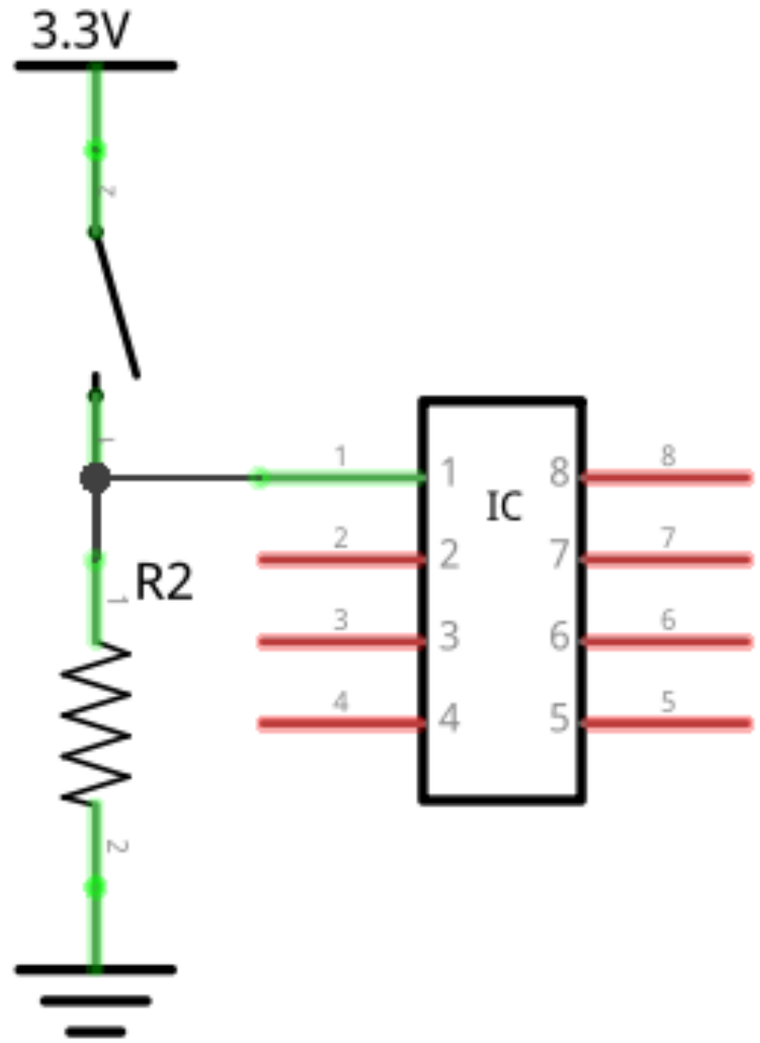
Connecting a button to a processor



- Assume pin 1 can detect 3.3 V (high) or 0 V (low)
- What happens when the button is pushed and released?

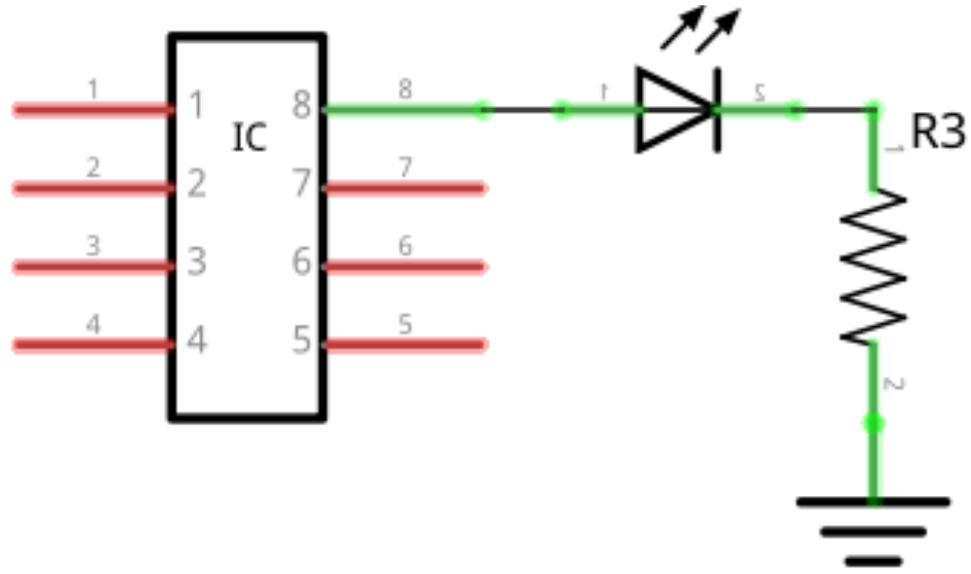
Fixing the button

- Add a pull down resistor
- Some processors can do this internally
- What happens when the button is pressed and released?
- For buttons, R2 can be set to 10 – 100 KΩ



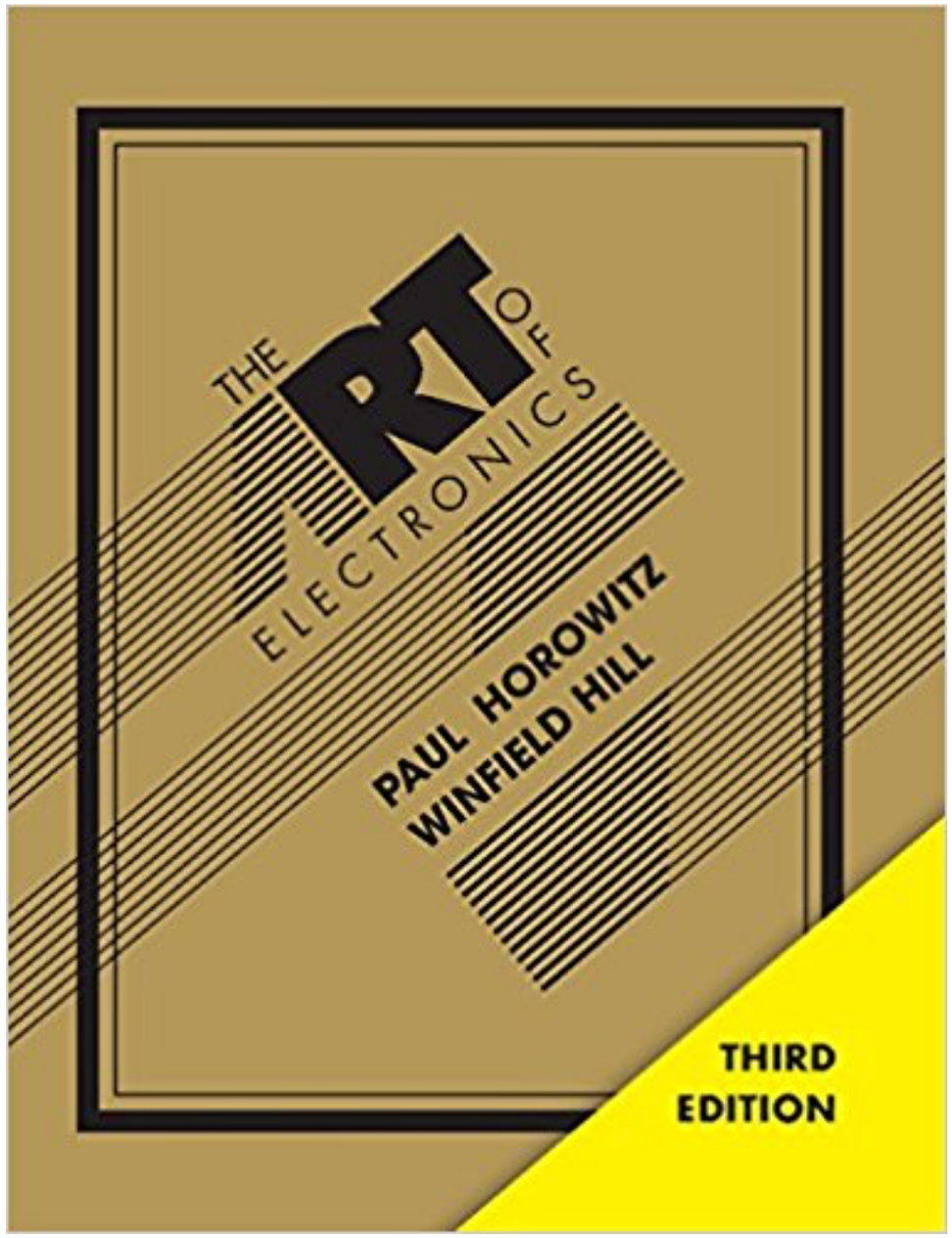
Connecting LEDs

- Too much current breaks processors and LEDs
- Resistors limit current (and dim the LED)
- Apply Ohm's law (V = IR) to calculate resistor
- Red LED voltage drop = 1.8 V
- Blue LED voltage drop = 3 V
- 3.3 V - 1.8 V = 1.5 V. If 10 mA is desired current, then 1.5 V / 0.01 A = 150 Ω resistor is needed.



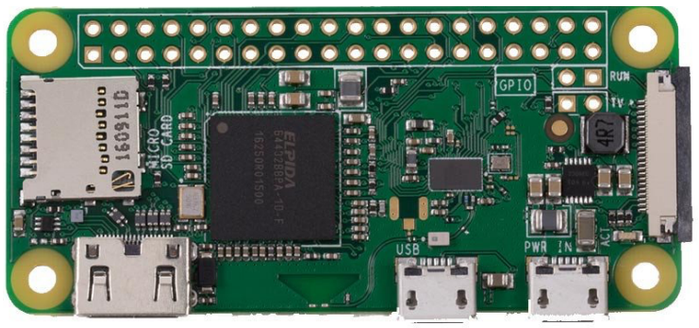
More electronics

- The Art of Electronics
- artofelectronics.net
- Contains way more info than you need but will last forever



GrovePi Zero

- Raspberry Pi hat for connecting up Grove sensors w/o soldering
- Isolates Pi somewhat from mistakes
- Has an ATmega processor
  - Supports analog sensors
  - 5 V digital I/O with more current
- I2C and UART passed through
- https://github.com/DexterInd/GrovePi/tree/master/Hardware



Grove Sensors

Grove Sensors for the Raspberry Pi: Officially supported and Tested

These Grove modules have been tested by Dexter Industries and the GrovePi community and have examples available to try them out in our Github repository.


Grove – Button


Grove – LED Socket Kit


Grove – Switch(P)


Grove – Light Sensor


Grove – Line Finder


Grove – Buzzer


Grove – Relay


Grove – Rotary Angle Sensor

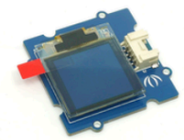

Grove – Temperature Sensor


Grove – Sound Sensor

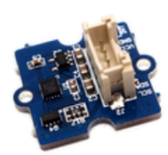

Grove – Vibration Motor


Grove – Ultrasonic Ranger


Grove – PIR Motion Sensor


Grove – OLED Display 96*96


Grove – Magnetic Switch



https://www.dexterindustries.com/GrovePi/supported-sensors/

Seeed Studio and Dexter Industries sell most of them, but other companies have added Grove connectors to interesting modules too.

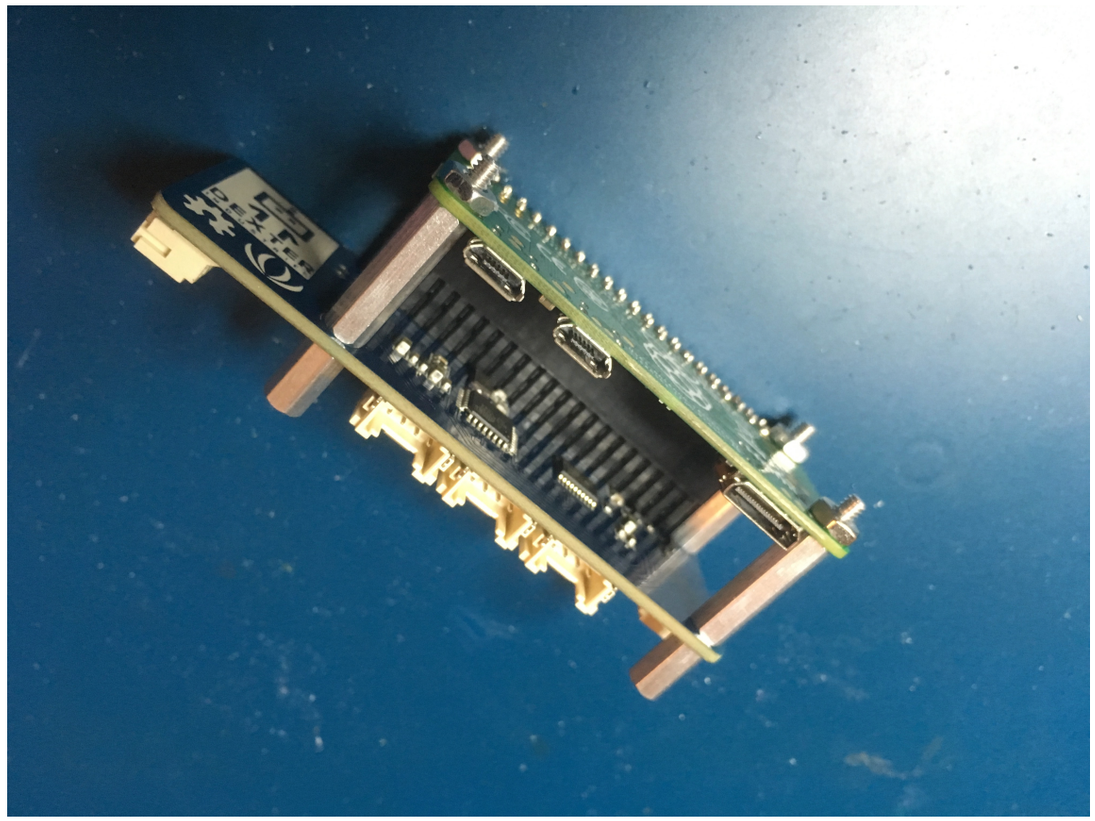
GrovePi Notes

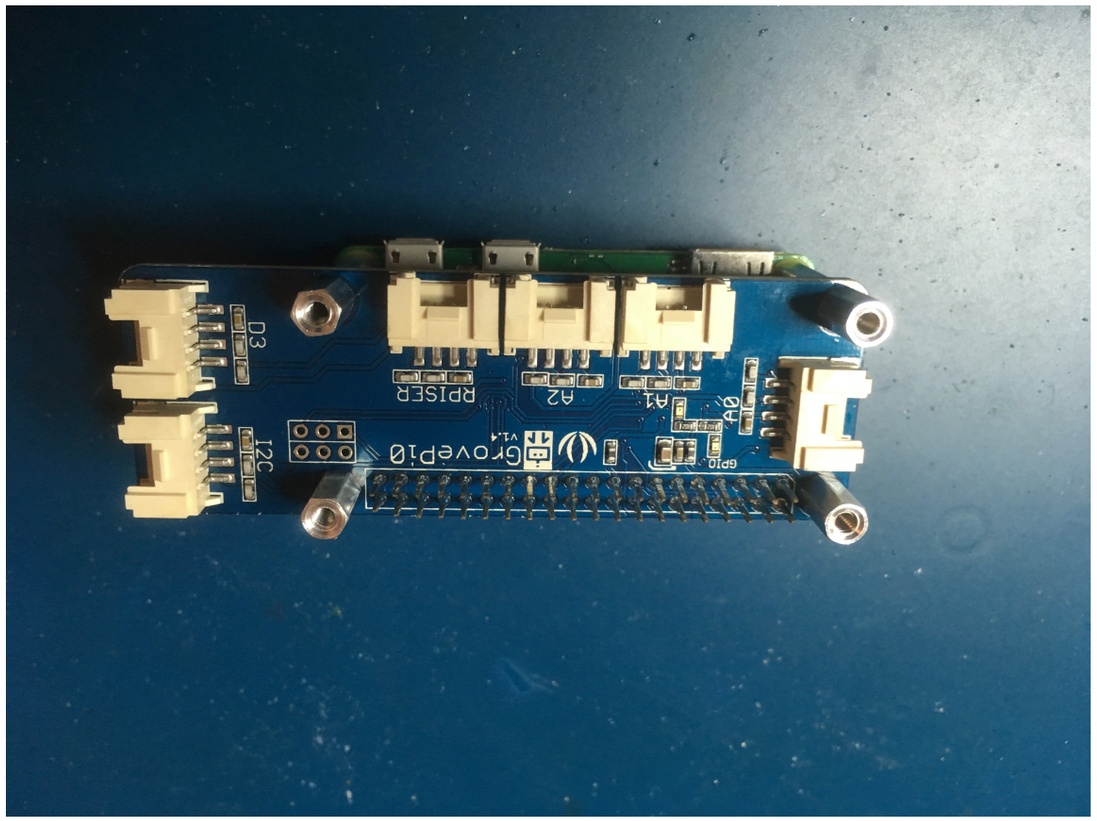
- Pros
  - Great for prototyping and learning electronics
  - Inexpensive and easy to buy
  - Lots of open source software and docs out there
  - Basically works
- Cons
  - Slow and power hungry compared to professional alternatives
  - Software quality varies

GrovePi Zero assembly

- GrovePi Zero
- Four 3/4” M-F 4-40 standoffs
- Four 3/8” M-F 4-40 standoffs
- Four 4-40 hex nuts

GrovePi Zero assembly

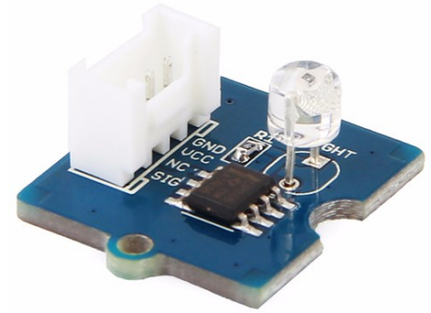


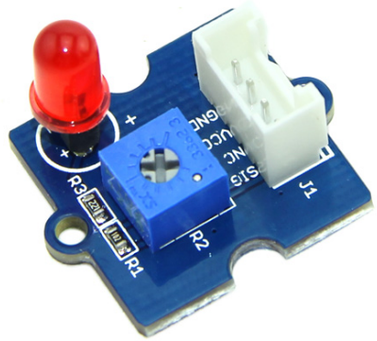


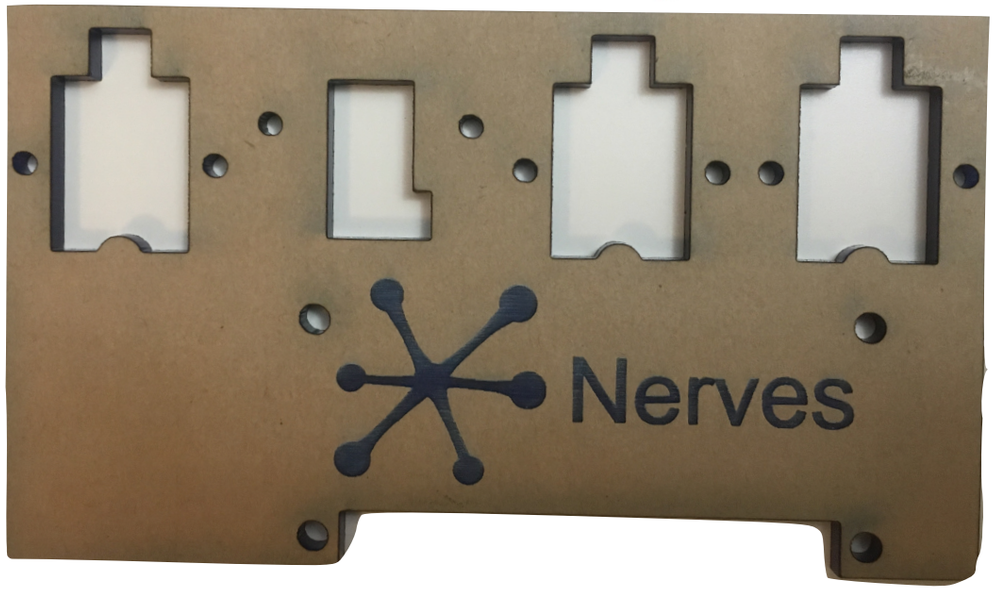
Don’t worry if there’s a small gap between the GrovePi Zero connector and the RPi Zero.

Adding Grove Sensors and the faceplate

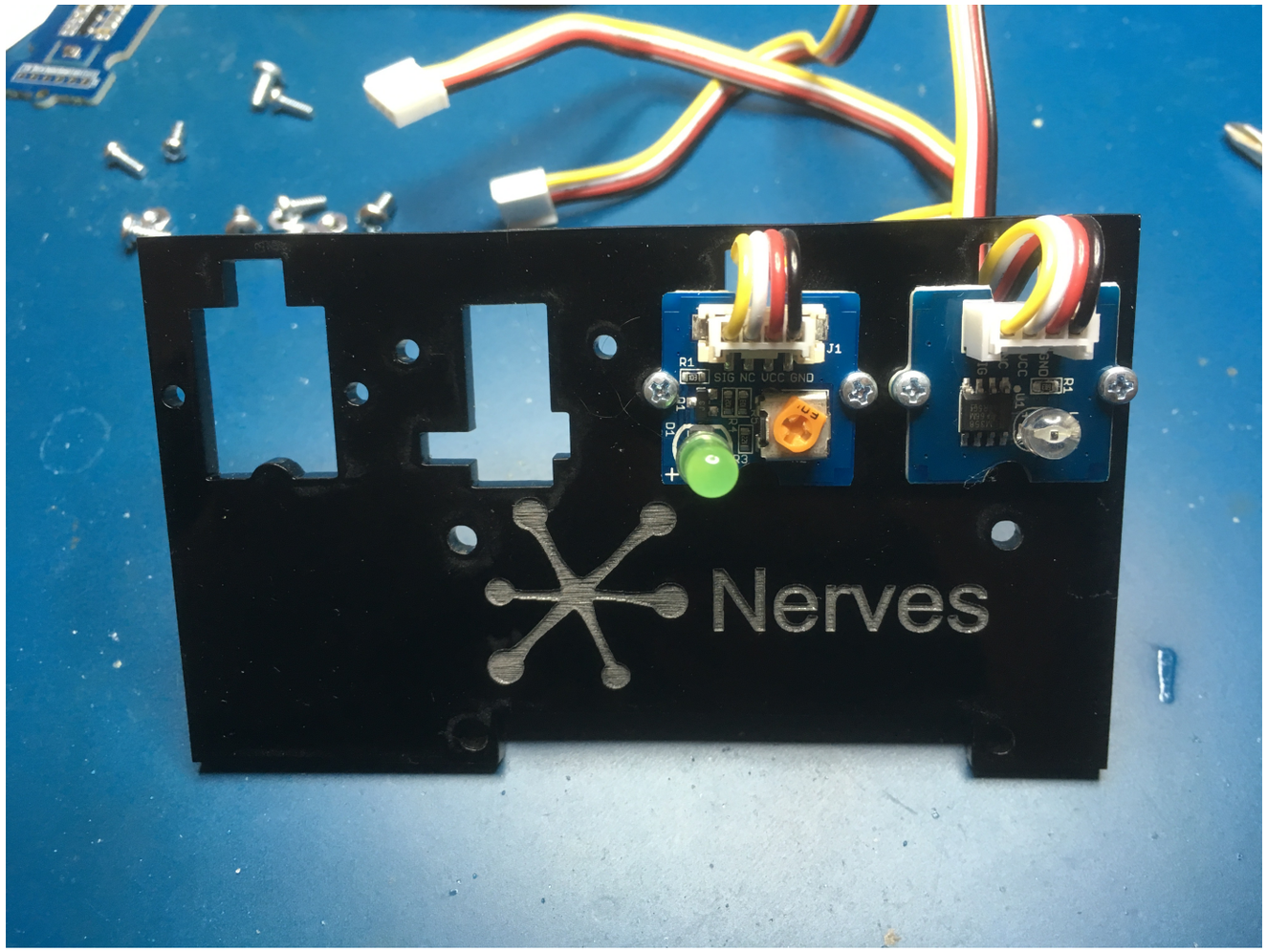
- Grove - Red LED (or other LED or buzzer)
- Grove - Light Sensor
- Acrylic faceplate
- Four 2-56 3/8" screws
- Four 2-56 hex nuts





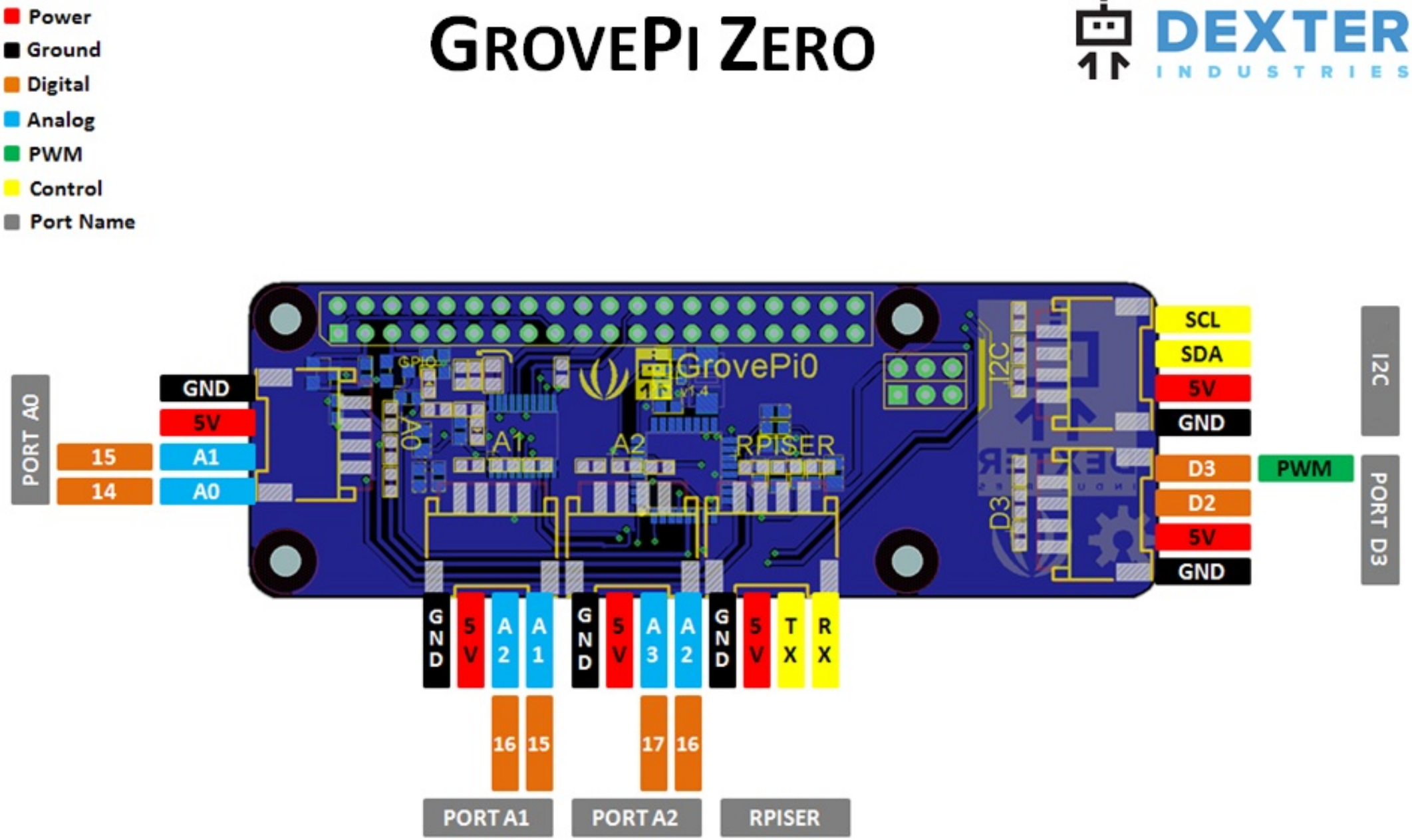


Attach sensors to faceplate



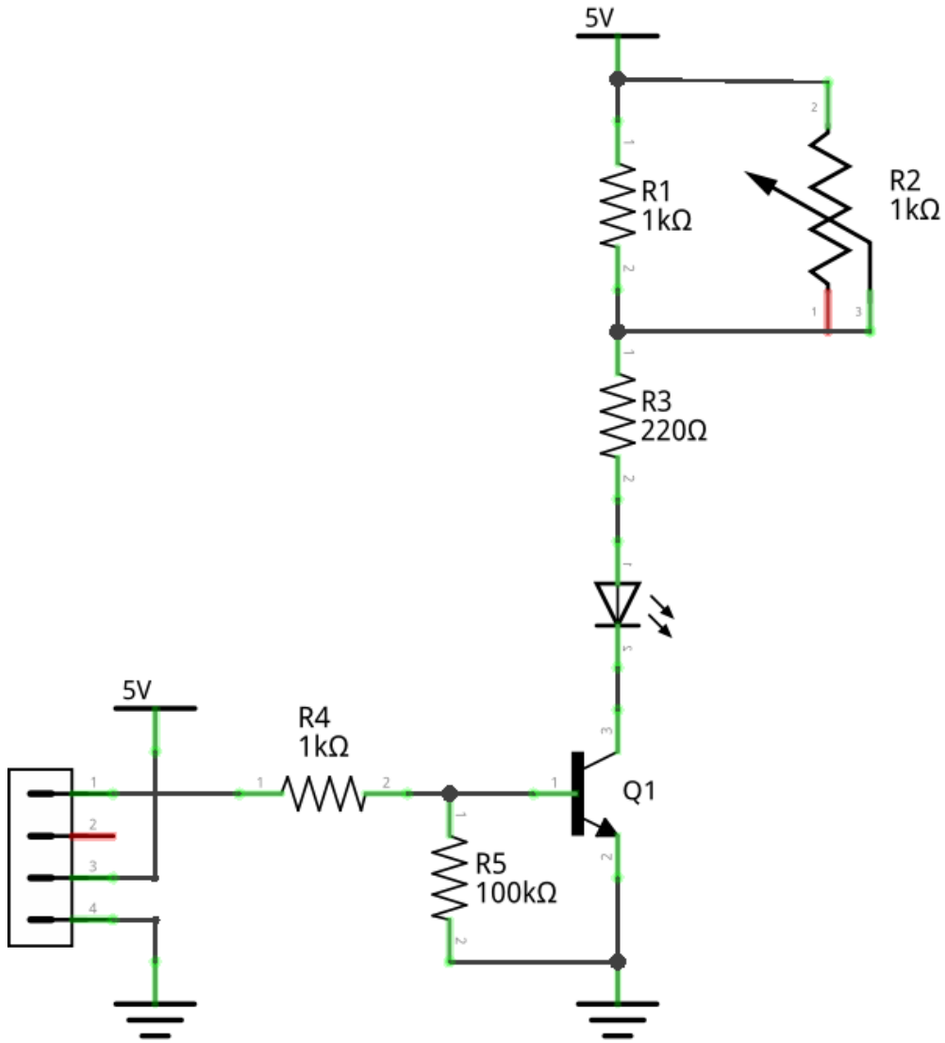
- Remove any protective paper off faceplate
- Plug the wires in to the sensors

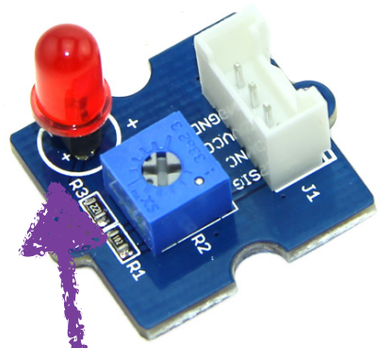
GrovePi Zero



Connect the light sensor to Port A1
Connect the LED or buzzer to Port D3

Grove LED Module

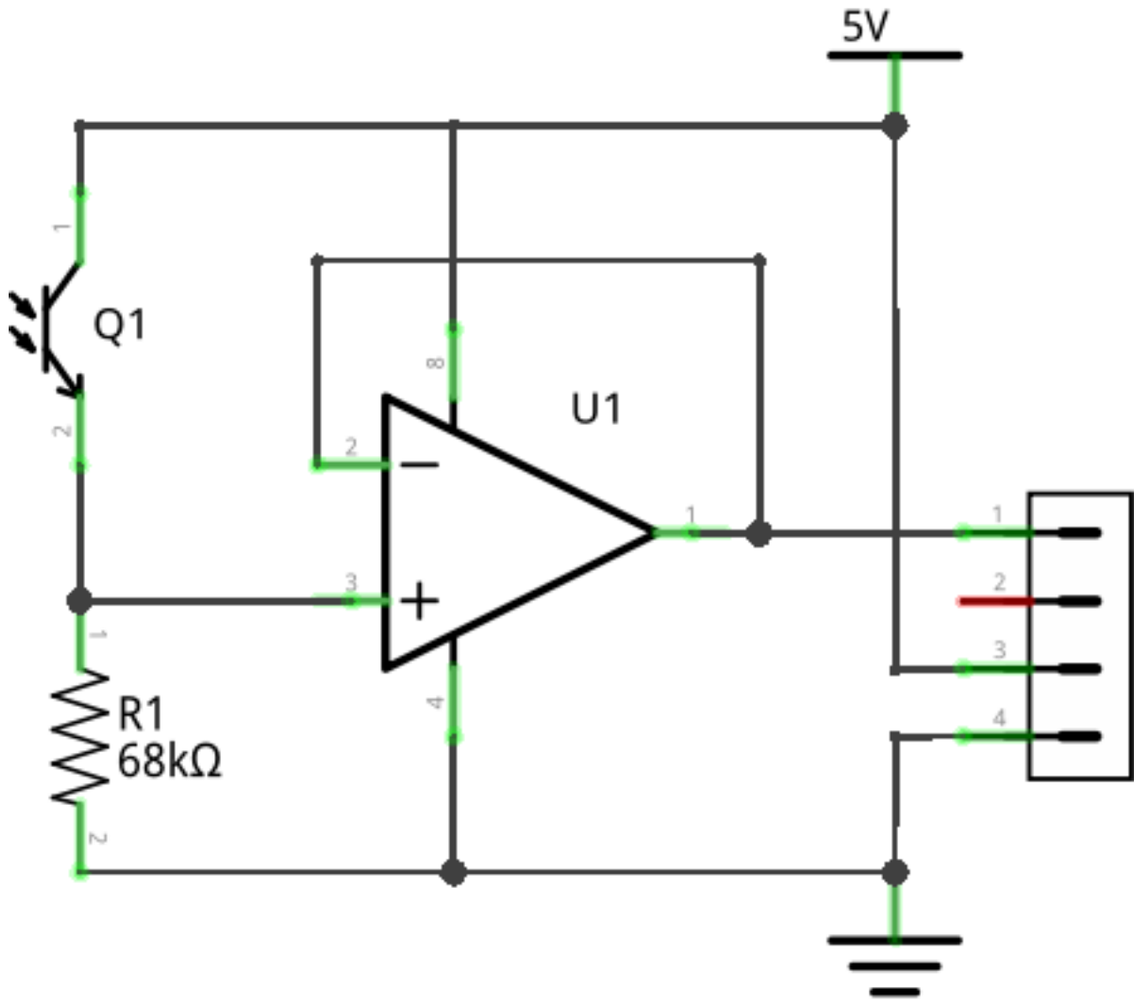


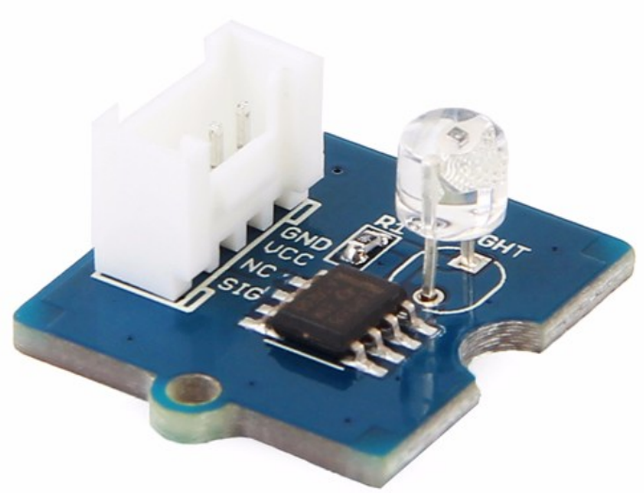


LED direction matters

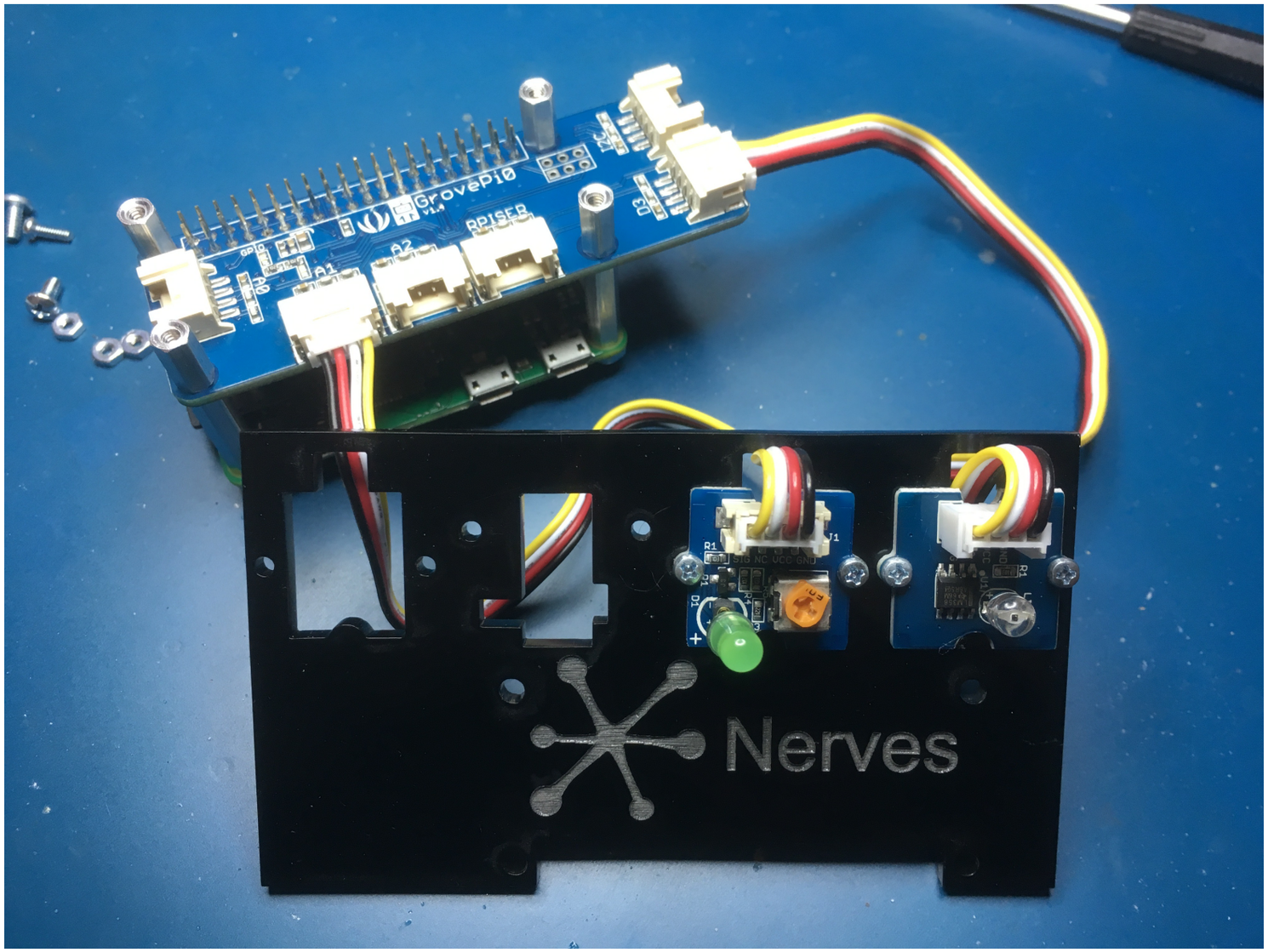
Flat side on base goes towards - side, but don't worry about it. If it doesn't work, rotate it.

Grove Light Sensor





Connect sensors



LED to D3 and Light sensor to A1

Create a new nerves_init_gadget project

```
$ cd <workspace>
$ mix nerves.new basic_io
$ cd basic_io

add nerves_init_gadget to mix.exs and config.exs
copy ssh key config to config/config.exs
```

Skip typing

```
$ git clone \
https://bitbucket.org/fhunleth/nervestraining-basic_io.git basic_io
$ cd basic_io
$ git checkout step1
```

Pull in the grovepi library

```
def deps(target) do
    [
      {:bootloader, "~> 0.1"},
      {:nerves_runtime, "~> 0.4"},
      {:nerves_init_gadget, "~> 0.2"},
      {:grovepi, “~> 0.3”}
    ] ++ system(target)
  end
```

Docs: https://hexdocs.pm/grovepi/readme.html

```
Skip typing
$ git clone \
https://bitbucket.org/fhunleth/nervestraining-basic_io.git basic_io
$ cd basic_io
$ git checkout step2
```

Try it out

```
$ mix firmware

$ mix firmware.push nerves.local
   -or-
$ ./upload.sh

$ picocom /dev/tty.usb*

iex> GrovePi.Board.firmware_version
"1.2.2"
```

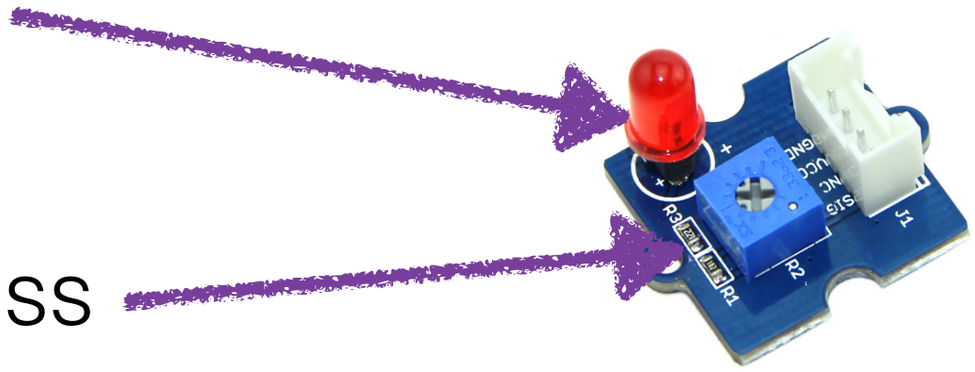
It is very important that this line works for everything that follows.

Turning on an LED

```
# LED board is connected to Port D3
iex> GrovePi.Digital.set_pin_mode(3, :output)
:ok
iex> GrovePi.Digital.write(3, 1)
:ok
# Turn the LED off
iex> GrovePi.Digital.write(3, 0)
:ok
```

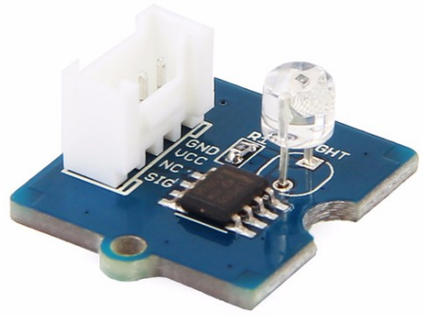
If not on, try removing the LED
and re-inserting the opposite way.

Use screwdriver to adjust brightness



Reading the light sensor

```
# Light sensor is connected to port A1
iex> GrovePi.Analog.read(1)
54
```



What are the maximum and minimum values returned?
What does this infer about the ADC being used?

Connecting it up

- Create a GenServer that polls the light level and turns on the LED if too dark (e.g., < 10)

One solution

lib/basic_io/application.ex

```
    # Define workers and child supervisors to be supervised
    children = [
      worker(BasicIo.Worker, []),
    ]
```

Skip typing

```
$ git checkout step3
```

One solution

```elixir
defmodule BasicIo.Worker do
  use GenServer

  @light_sensor 1  # Port A1
  @led 3           # Port D3

  def init(state) do
    GrovePi.Digital.set_pin_mode(@led, :output)
    send(self(), :check_sensor)
    {:ok, state}
  end

  def handle_info(:check_sensor, state) do
    light = GrovePi.Analog.read(@light_sensor)
    led_value = if light > 10, do: 0, else: 1
    GrovePi.Digital.write(@led, led_value)

    Process.send_after(self(), :check_sensor, 100)
    {:noreply, state}
  end
end
```

Skip typing

```
$ git checkout step3
```

Going farther with grovepi

- Check out how *grovepi* does unit testing w/o hardware
- Submit a PR for light sensor support
- Look at *grovepi* triggers to avoid polling in application
- https://github.com/adkron/grovepi

ELIXIRCONF™ 2017

Nerves Training
Part 2: Basic I/O