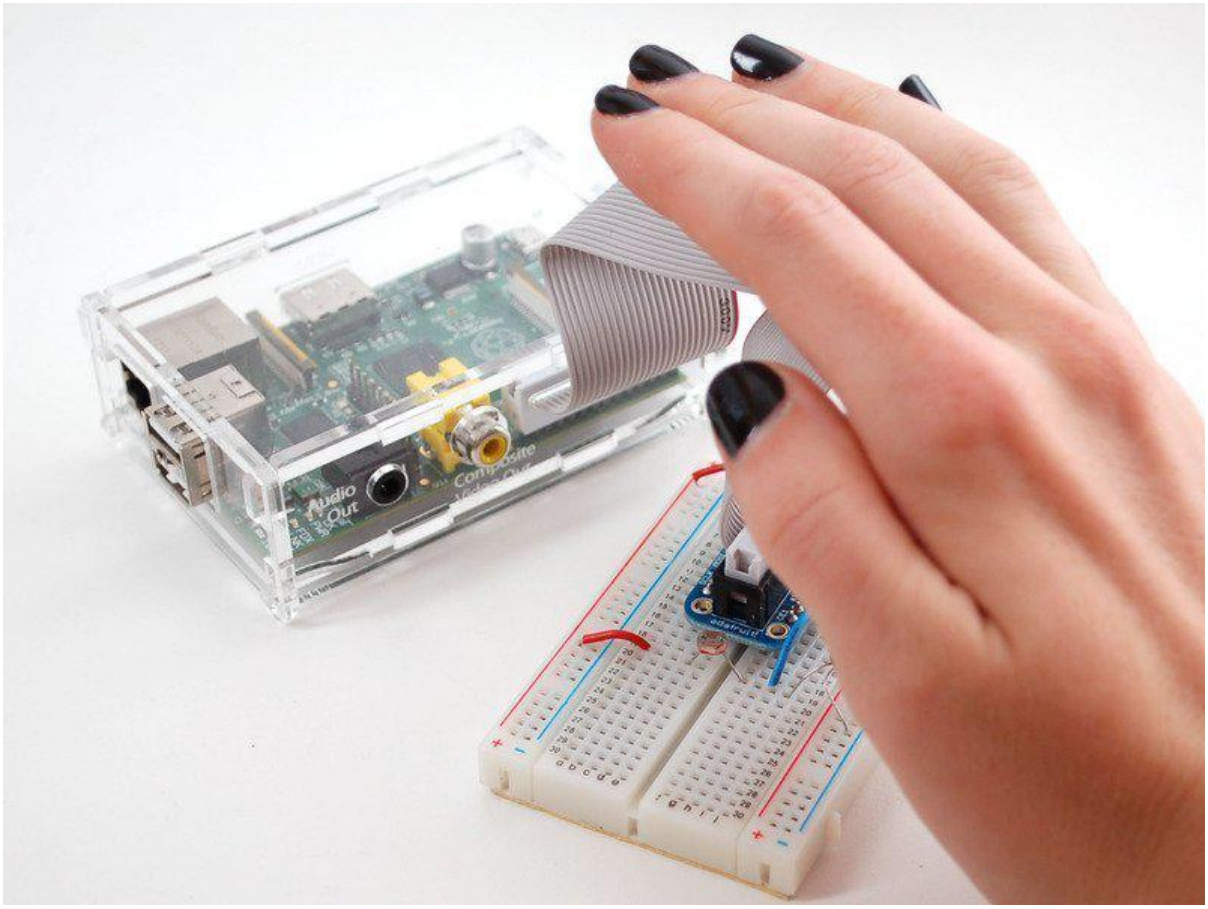
# Basic Resistor Sensor Reading on Raspberry Pi

Created by lady ada

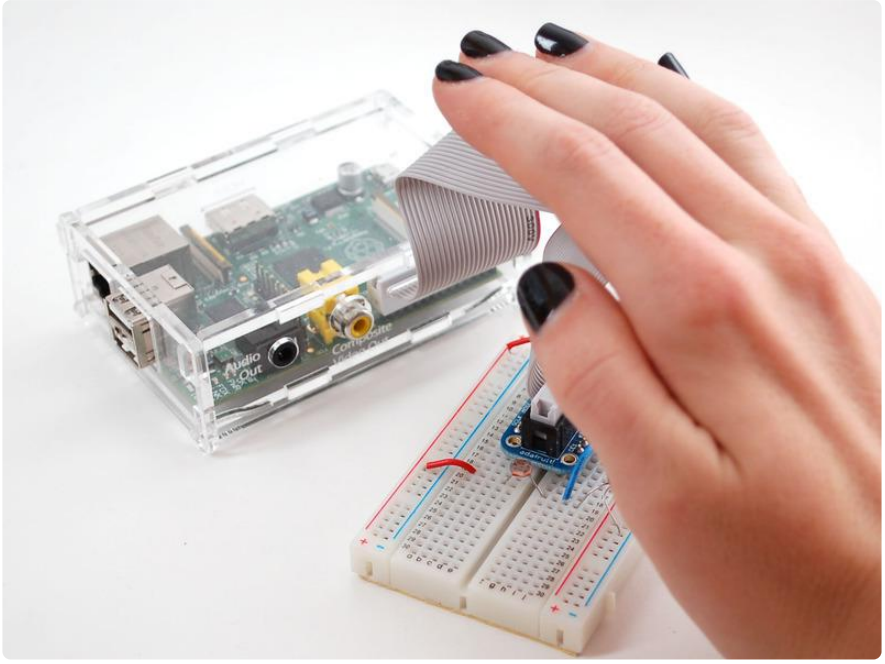


https://learn.adafruit.com/basic-resistor-sensor-reading-on-raspberry-pi

Last updated on 2023-08-29 02:10:14 PM EDT

# Table of Contents

# Overview



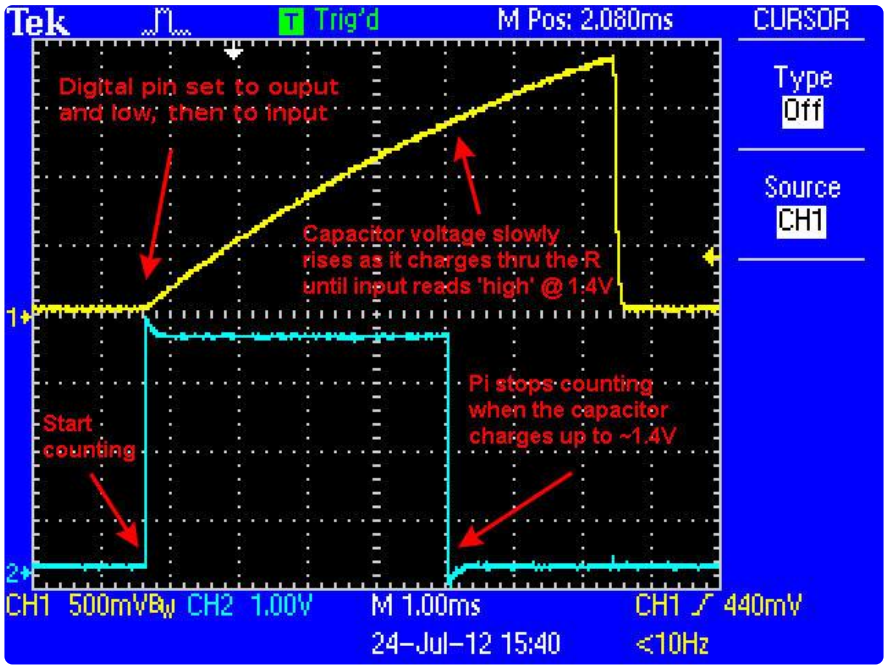
We've already covered how to use an Analog-to-Digital Converter chip with a Pi. These chips are the best way to read analog voltages from the Pi. However, there's a way to read many sensors without an ADC! By measuring the sensor as a resistor that is used to 'fill up' a capacitor, we can count how long it takes. It's not nearly as precise as an ADC and its a little flakey (since it depends on the Pi timing itself which can vary based on how 'busy' the computer is)

The way we do this is by taking advantage of a basic electronic property of resistors and capacitors. It turns out that if you take a capacitor that is initially storing no voltage, and then connect it to power (like 3.3V) through a resistor, it will charge up to the power voltage slowly. The bigger the resistor, the slower it is.

This technique only works with sensors that act like resistors. however, there are quite a few fun sensors that act this way: photocells (http://adafru.it/161), thermistors (temperature sensors) (http://adafru.it/372), flex sensors (http://adafru.it/182), force-sensitive resistors (http://adafru.it/166), and many more.

It cannot be used with sensors that have a pure analog output like IR distance sensors (http://adafru.it/164) or analog accelerometers (http://adafru.it/163).

# How it works




This capture from an oscilloscope shows whats happening on the digital pin (yellow). The blue line indicates when the Pi starts counting and when the counting is complete, about 4.5ms later.

This is because the capacitor acts like a bucket and the resistor is like a thin pipe. To fill a bucket up with a very thin pipe takes enough time that you can figure out how wide the pipe is by timing how long it takes to fill the bucket up halfway

In this case, our 'bucket' is a 1uF ceramic capacitor. You can change the capacitor nearly any way you want but the timing values will also change. 1uF seems to be an OK place to start for most sensors. If you want more range, use a bigger cap - but it will take longer to measure. For faster reads, go with a smaller capacitor

# Necessary Packages

# Update Your Pi to the Latest Raspbian

Your Pi will need to be running the latest version of Raspbian. This tutorial was written using Raspbian Stretch (Nov. 2018). Checkout our guide for Preparing an SD Card for your Raspberry Pi () if you have not done so already. After the installation is complete be sure and run the following commands to make sure your installation packages and firmware are up to date.

```
$ sudo apt-get update -y
$ sudo apt-get dist-upgrade -y
$ sudo apt-get upgrade -y
$ sudo rpi-update
$ sudo reboot
```

# Install pip3

pip3 is already installed with a full Raspbian installation, but the Raspbian Lite does not include pip3 so it needs to be installed as shown below.

```
$ sudo apt-get install python3-pip
```

# Install adafruit-blinka

The adafruit-blinka package works on all Raspberry Pi boards (except the compute nodes). It makes the CircuitPython libraries available on Raspberry Pi.

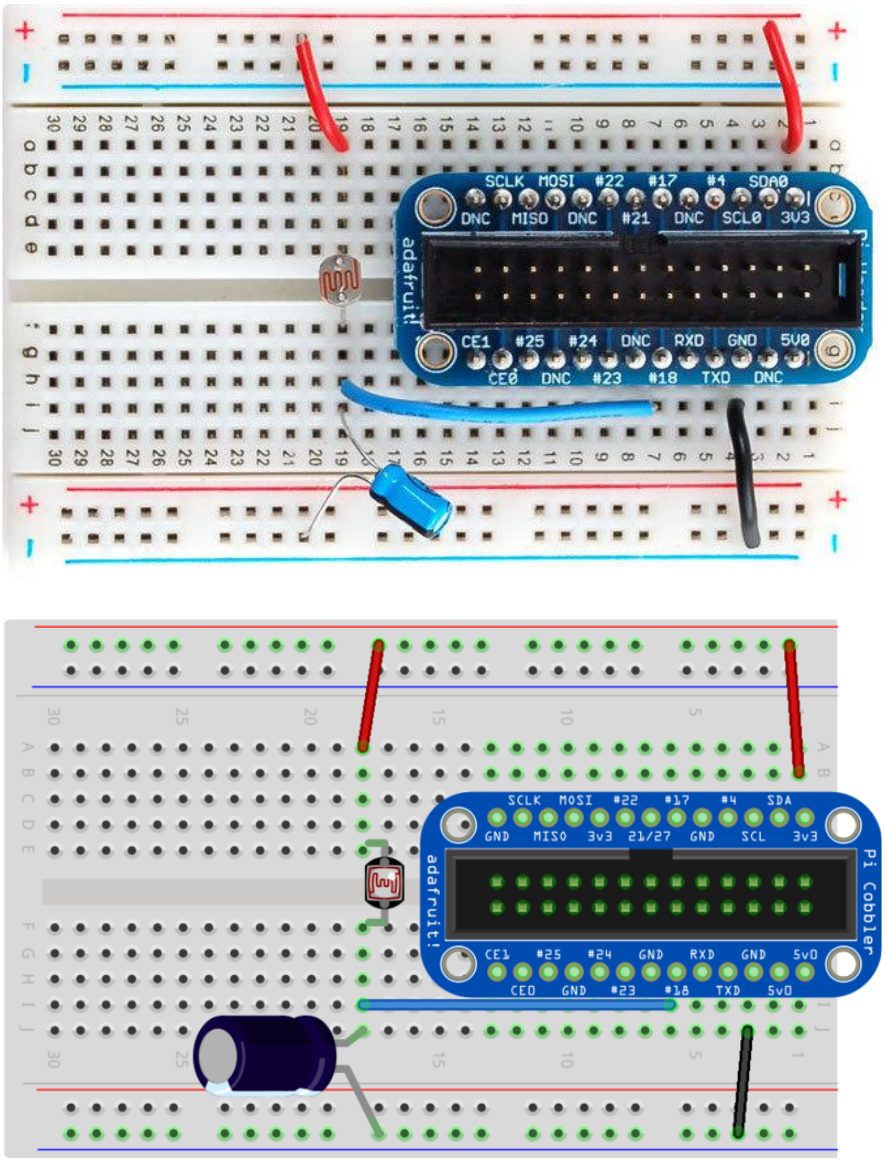
```
$ sudo pip3 install adafruit-blinka
```

---

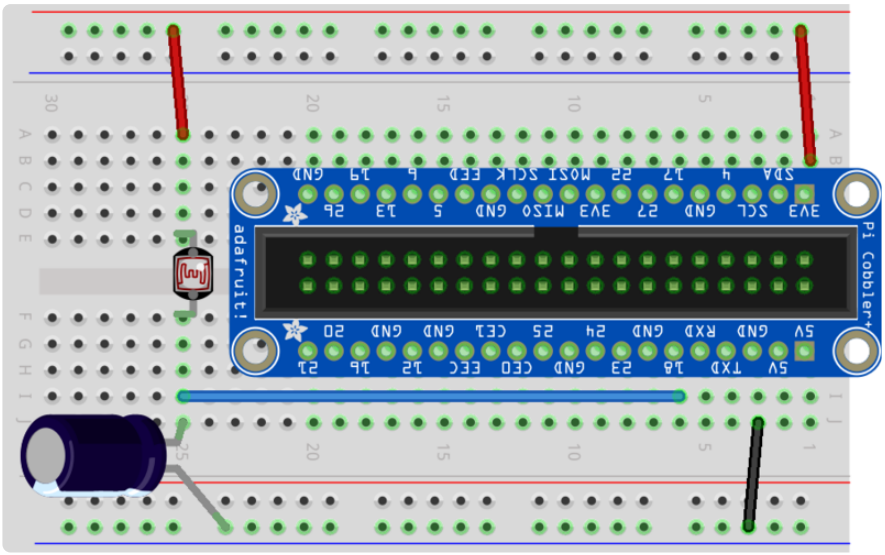
# Wiring

Wiring is simple using the Adafruit Pi Cobblers. Connect the blue right rail to ground and the red left rail to 3.3V. Then connect one side of the photocell to 3.3V and the other side to Pi GPIO #18 (you can use any pin but our example code is for #18). Then connect a 1uF capacitor from #18 to ground. Make sure the negative side of the capacitor (marked with a - down the side if its electrolytic) goes to ground. The capacitor just needs to be rated for 5V or greater, its really unlikely you'll find a 1uF that has less than 16V rating.

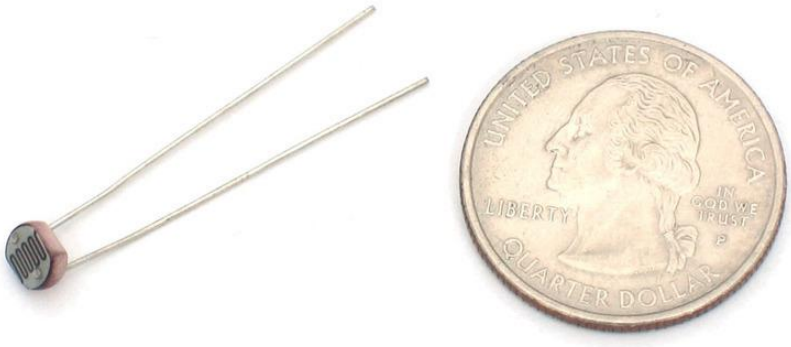
# Raspberry Pi 20-Pin Wiring (RPI rev. 1 and 2)



## Raspberry Pi 40-Pin Wiring (RPI rev. 3 - A+, B+ and Zero)




# Basic Photocell Reading




We'll start with a basic photocell. This is a resistor that changes resistance based on how bright the light is. You can read tons more about photocells in our tutorial () but basically we'll be able to measure how bright or dark the room is using the photocell. Note that photocells are not precision measurement devices, and this technique is also not very precise so its only good for basic measurements. For precision sensing, you'd want a digital lux sensor like this one (http://adafru.it/439) - we don't have a tutorial on connecting that to the Pi but we do have example code for Arduino.

## Download the Code

The following code can be downloaded to your raspberry pi

```
# SPDX-FileCopyrightText: 2019 Mikey Sklar for Adafruit Industries
#
# SPDX-License-Identifier: MIT

# Example for RC timing reading for Raspberry Pi
# using CircuitPython Libraries

import time
import board
from digitalio import DigitalInOut, Direction

RCpin = board.D18

while True:
    with DigitalInOut(RCpin) as rc:
        reading = 0

        # setup pin as output and direction low value
        rc.direction = Direction.OUTPUT
        rc.value = False

        time.sleep(0.1)

        # setup pin as input and wait for low value
        rc.direction = Direction.INPUT

        # This takes about 1 millisecond per loop cycle
        while rc.value is False:
            reading += 1
        print(reading)
```

Let's put this file right in your home directory for simplicity. The wget command makes things easy.

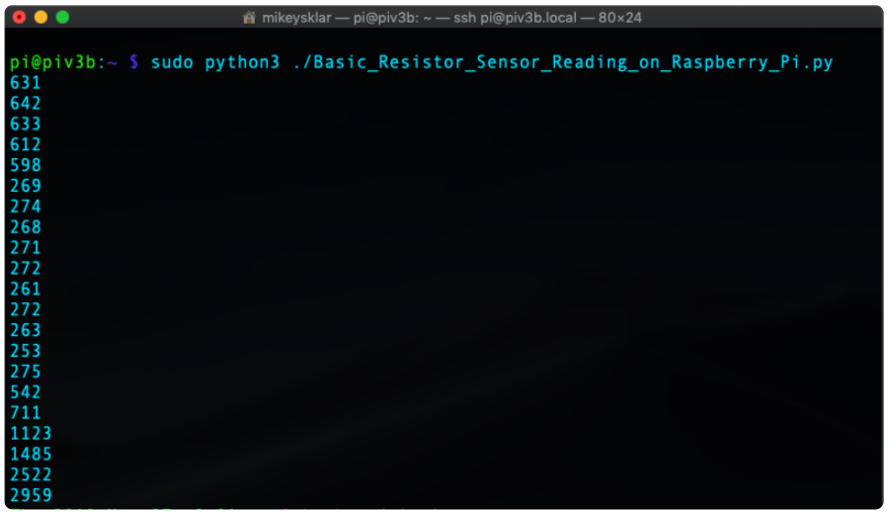
```
$ wget https://raw.githubusercontent.com/adafruit/Adafruit_Learning_System_Guides/
master/Basic_Resistor_Sensor_Reading_on_Raspberry_Pi/
Basic_Resistor_Sensor_Reading_on_Raspberry_Pi.py
```

# Running the Code

With the Pi connected to the Cobbler, run the script and shade your hand over the sensor to test it out!

The following command will start the program and you should see the ADC output on your screen.

```
$ sudo python3 ./Basic_Resistor_Sensor_Reading_on_Raspberry_Pi.py
```




Once you know it works you can change what pin you are using by changing the

RCpin = board.D18

to

RCpin = board.D(yourpinnumberhere)

any pin will work.