Follow along!
www.sarahwithee.com/raspberrypi

Intro to Hacking with the Raspberry Pi

Sarah Withee (@geekygirlsarah)
University of Missouri-Kansas City

π

What is the Raspberry Pi?

“The Raspberry Pi is a credit-card sized computer that plugs into your TV and a keyboard. It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word-processing and games. It also plays high-definition video. We want to see it being used by kids all over the world to learn programming.”

-- raspberrypi.org/faqs

What is “hacking”?

“We are taking back the term ‘Hacking’ which has been soured in the public mind. Hacking is an art form that uses something in a way in which it was not originally intended. This highly creative activity can be highly technical, simply clever, or both. Hackers bask in the glory of building it instead of buying it, repairing it rather than trashing it, and raiding their junk bins for new projects every time they can steal a few moments away.”

-- hackaday.com/about/

Disclaimer

This IS a workshop…

- for integrating the Pi into fun hacking projects…
- for learning to program with sensors, LEDs, and more
- for learning to build sample projects to inspire you to build your own projects

Disclaimer

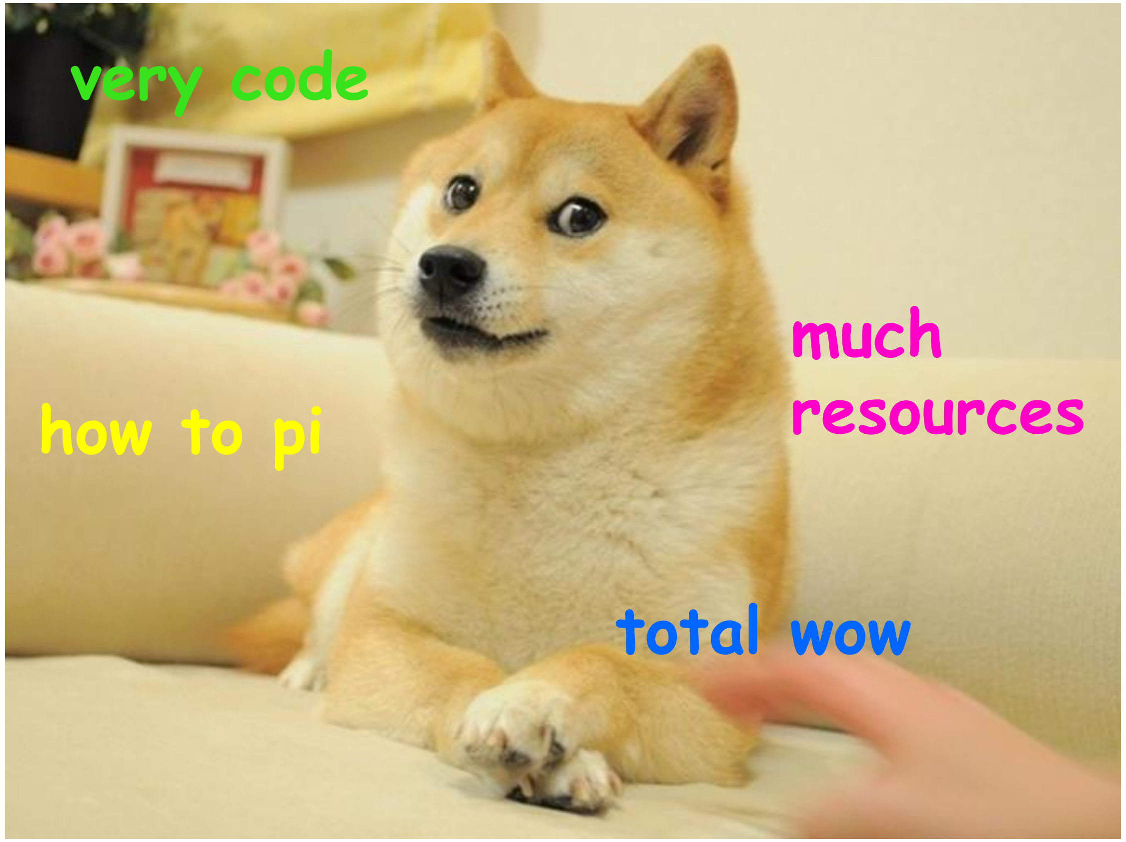
This is NOT a workshop…

- on how to use Linux
- on how to install a Raspberry Pi OS
- for advanced electronics
- on general programming. (It’s a regular computer, you can write code as normal)

I’ll include links at the end for these

Breakdown

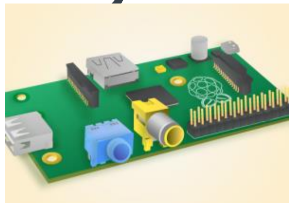
1. Raspberry Pi Information
2. Intro to Hardware
3. Project 1 (LEDs) – Cylon/Knight Rider Lights
4. Project 2 (LCD Screen) – Bouncy Ball
5. Project 3 (Sensors) – Distance Tracker
6. Other Ideas
7. More Information and Conclusion

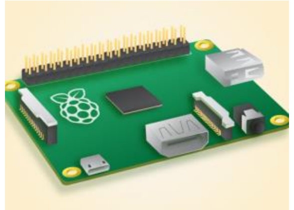
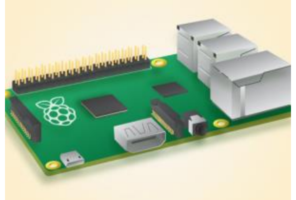
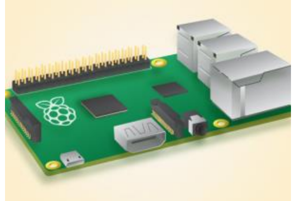


www.sarahwithee.com/raspberrypi

Raspberry Pi Information

Raspberry Pi Models



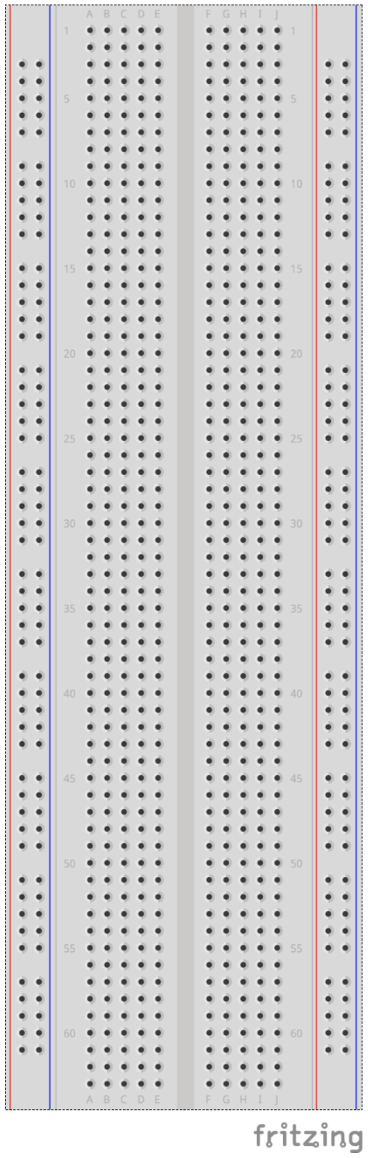
	Model A	**Model B**	**Model A+**	**Model B+**	**2 Model B**
Processor	700 MHz	700 MHz	700 MHz	700 MHz	4x 900MHz
Memory	256 MB	512 MB	256 MB	512 MB	1 GB
Expansion	Full SD	Micro SD	Full SD	Micro SD	Micro SD
USB Ports	1	2	1	4	4
Ethernet	No	10/100	No	10/100	10/100
A/V	HDMI, composite	HDMI, composite	HDMI, 3.5mm jack	HDMI, 3.5mm jack	HDMI, 3.5mm jack
GPIO Pins	26	26	40	40	40
Power	300mA*, 1.5W	700mA*, 3.5W	200mA*, 1W	600mA*, 3W	800mA*, 4W
Cost	$25	$35	$20	$30	$35

* Barebones Pi with no peripherals attached. 1.2A minimum power supply recommended.

Intro to Hardware

What Is a Breadboard?

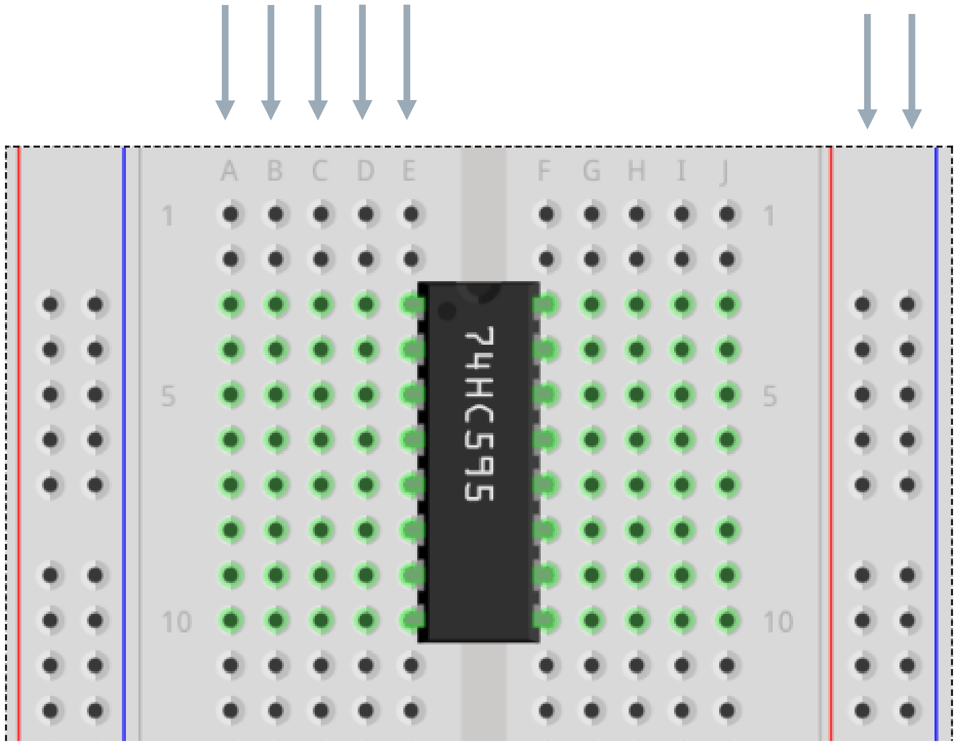
- A type of board used for prototyping
- Each row of 5 is connected
- Each column of 50 is connected
- Columns on side are power rails:
  - Red provides power (3.3 or 5 volts from source)
  - Blue provides ground (power back to source)
- Any wire plugged into a row is connected to any other wire in the same row



What Is a Breadboard?

Each row of 5 holes are connected to each other

Each column of holes are connected together



Why?

Each row is designed to allow multiple wires to plug into an integrated circuit chip.

Why?

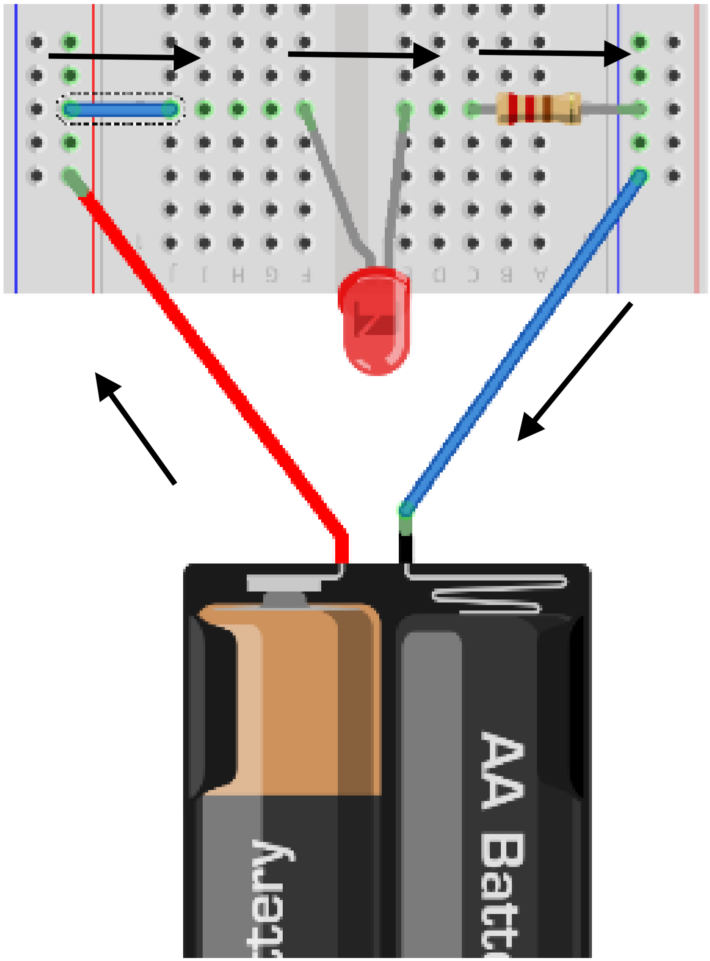
Columns are designed to allow easy way to provide power and ground for all components

Making a Circuit

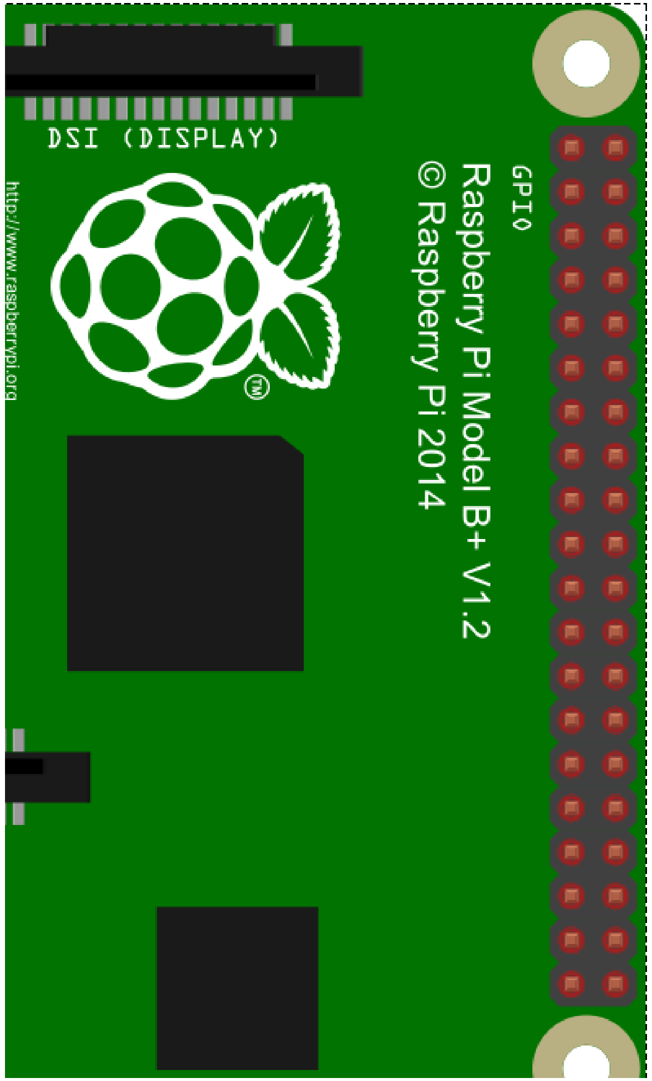
- Electricity must flow in a full circle from source, through elements, back to the source
- Source of power is called positive terminal
- Must return back to the source at negative terminal

- In this example, power flows:

Battery +	→	Red rail
Red rail	→	Blue wire
Blue wire	→	LED
LED	→	Resistor
Resistor	→	Blue rail
Blue rail	→	Battery -



Powering Circuits

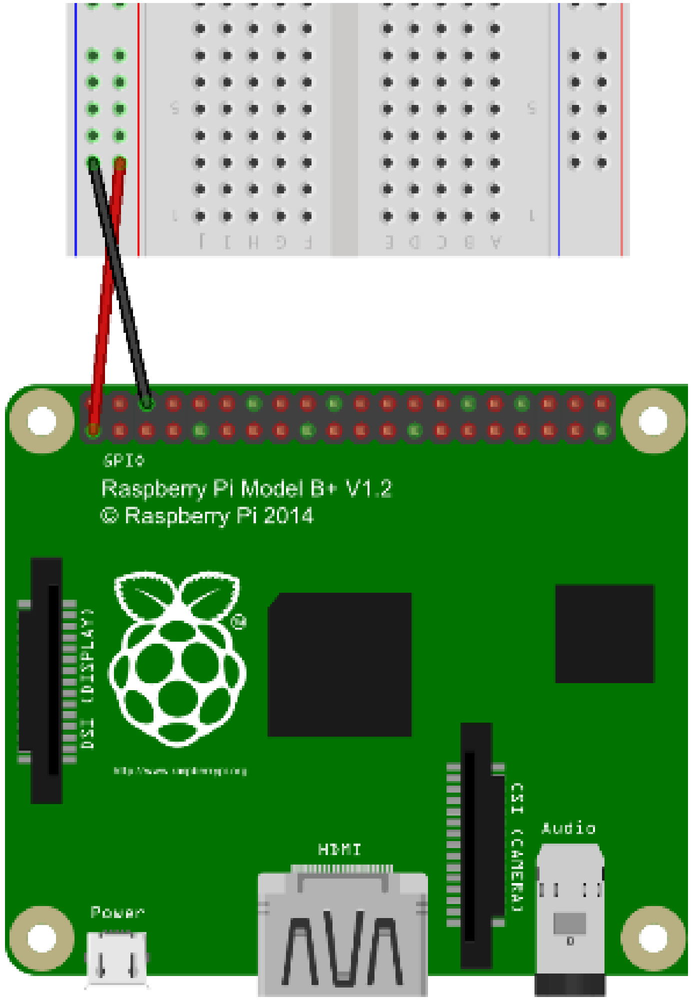


Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I2C)	DC Power 5v	04
05	GPIO03 (SCL1 , I2C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)	(I2C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Powering Circuits

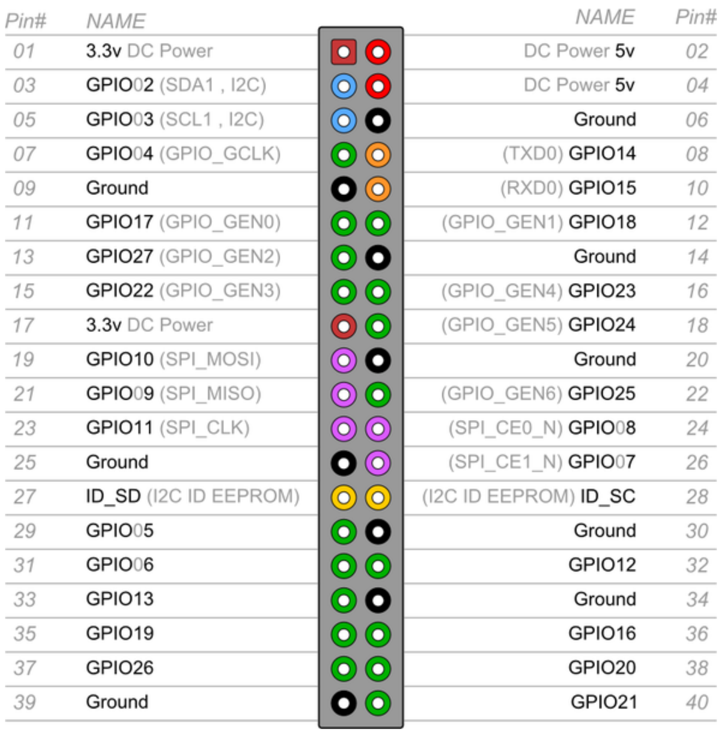
- Outgoing Power:
  - Pi provides 3.3 volts (V) and 5 volts, but we will use 3.3
  - Often labelled as +3.3V, +5V, or Vcc
  - This is the “positive terminal”
- Return Power:
  - Often labelled as Ground or GND
  - Most devices have several of them, you can use any of them

- Typically we wire +3.3/+5 to red rail and GND to blue rail on breadboard



General Purpose Input/Output (GPIO)

- Raspberry Pi provides 26 or 40 pins, depending on mode.
- 40 pins (A+, B+, 2 B) are backward compatible with 26 pins (A, B)
- Two +5V, Two +3.3V, Eight Ground
- Others are “general purpose”, meaning can be input or output based on programming
- Output pins send +3.3V
- Some pins have special purposes and can’t be swapped



Programming GPIO Pins

- Python (and other languages) for Pi include GPIO library
- Python's requires:
- At start, must set output mode:
  - Board – physical pin #
  - BCM – GPIO pin #
  - My examples use Board
- At end, must exit cleanly:

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setmode(GPIO.BCM)
```

```
GPIO.cleanup()
```

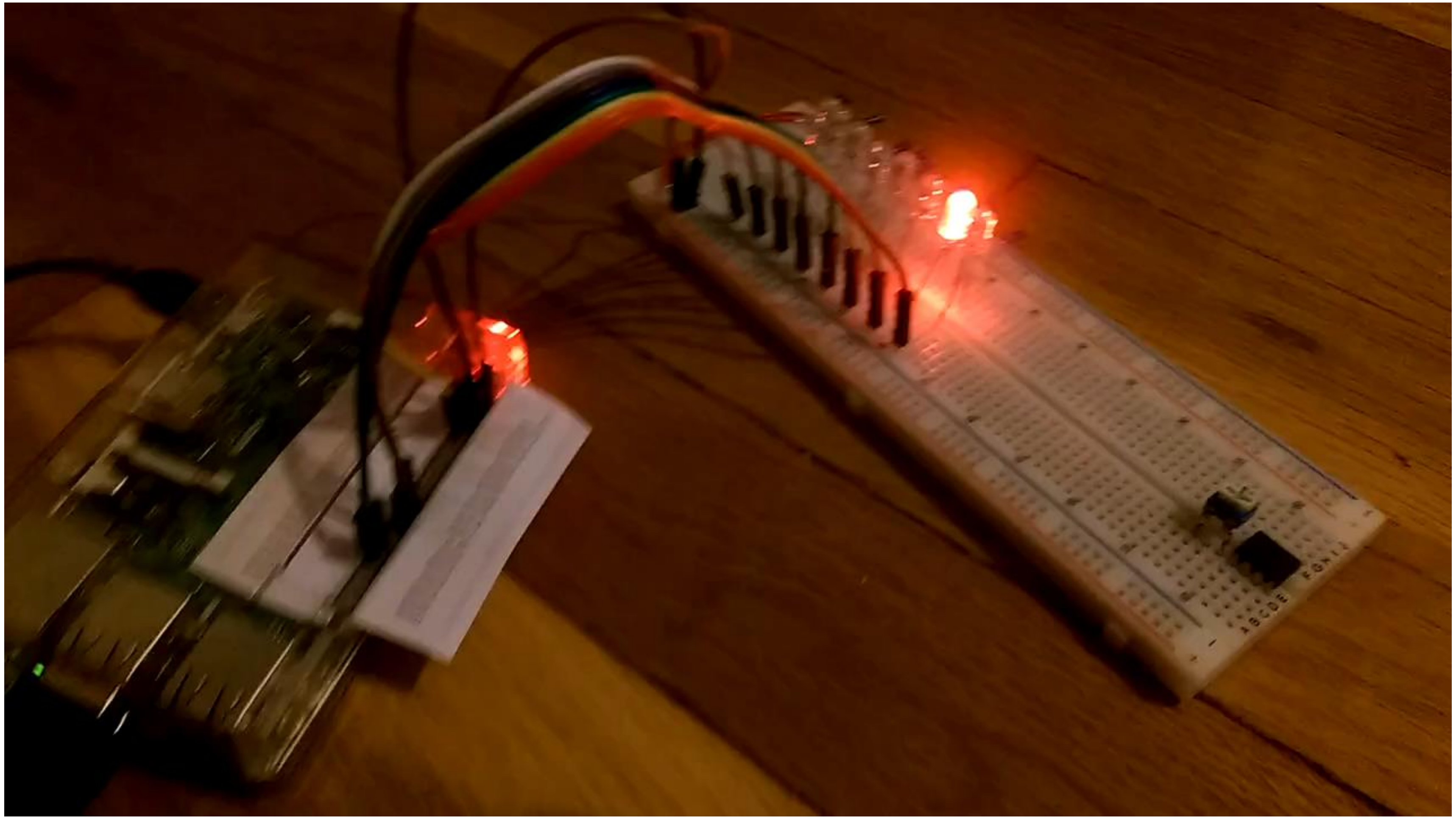
Programming GPIO Pins

- For each GPIO pin used, you must set it up as input or output first:
- Can read/write to it with functions:

```
GPIO.setup(3, GPIO.OUT)
GPIO.setup(4, GPIO.IN)


GPIO.output(3, True)   # +3.3V
GPIO.output(3, GPIO.HIGH)

GPIO.output(3, False)     # 0V
GPIO.output(3, GPIO.LOW)

x=GPIO.input(4)     # Bool or
                    # 0/1
```

Project 1 (LEDs) – Cylon/Knight Rider Lights

Demo



Materials List

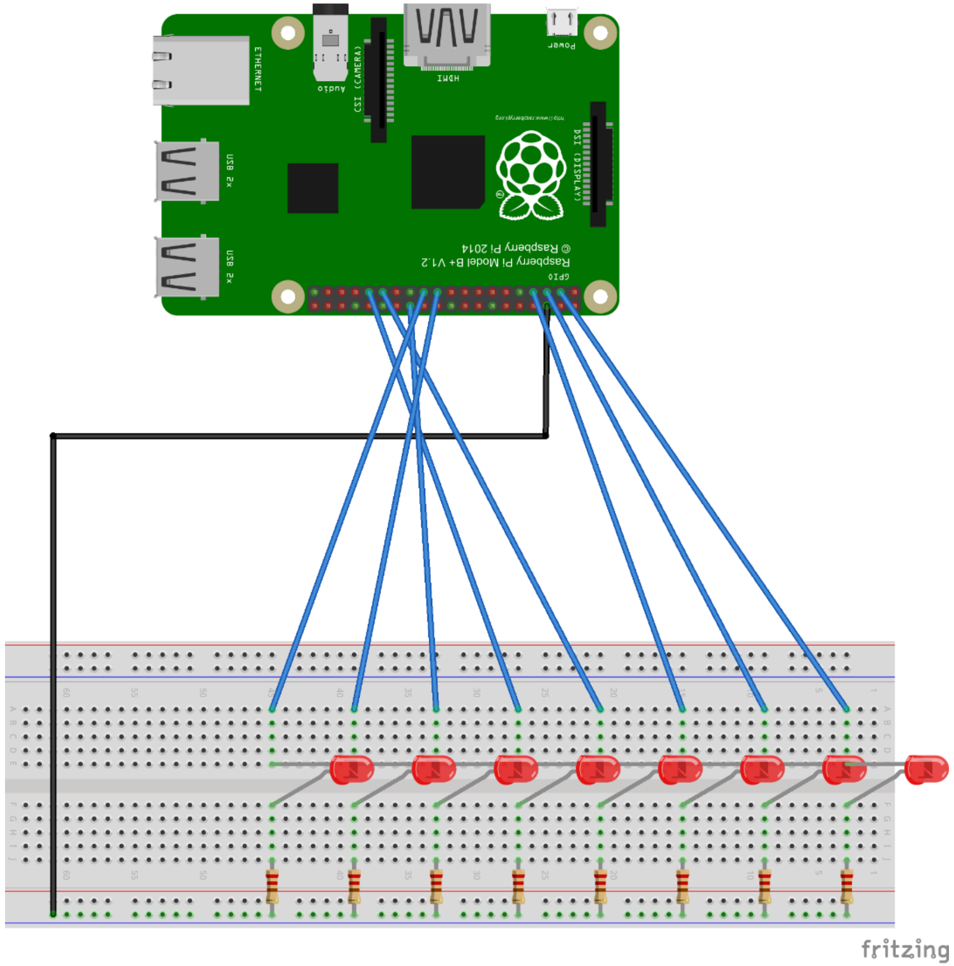
› Raspberry Pi (any model)
› Power supply (I recommend at least 1.5A for this project)
› 1 Breadboard
› 8 light emitting diodes*
› 8 resistors*
› 9 jumper wires

* I’ll explain these in a minute

Light Emitting Diodes (LED)

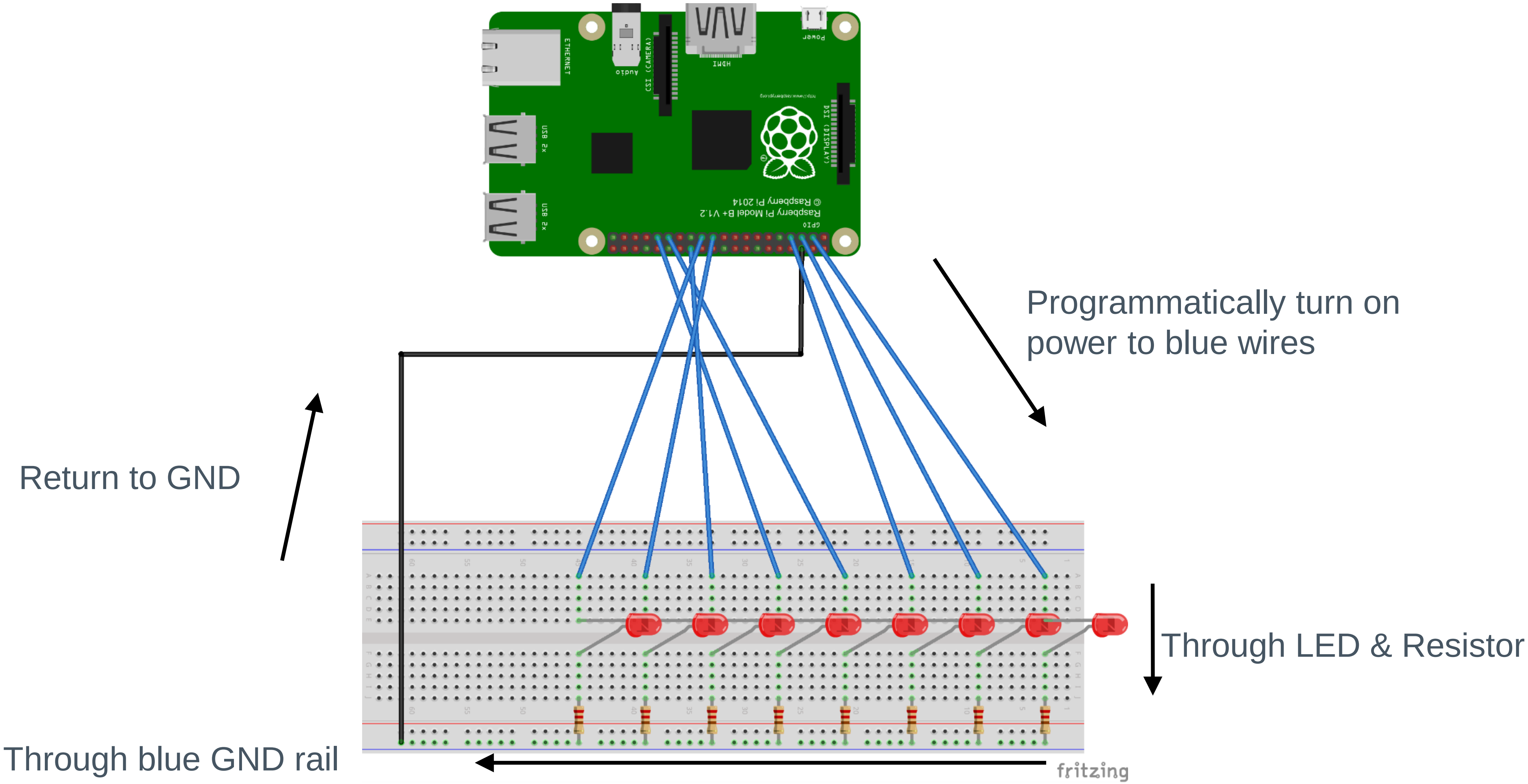
- Diode – Electronic device that allows electricity to flow in one direction
- Anode – longer wire (+)
- Cathode – shorter wire (-)
- When power is applied (correctly), it glows. If incorrect, nothing happens
- Resistor needed otherwise LED will glow bright white then fry
- I recommend just buying LED packs that include resistors with them

Hardware Layout



- Wire up GND wire to blue rail
- +3.3V rail not needed
- GPIO 2-10 should wire up to each LED's anode (longer wire)
- LED's cathode (shorter wire) should go to resistor
- Resistors should go to blue rail

Hardware Layout



Code

```
# Import libraries
import time
import RPi.GPIO as GPIO

# Create array of GPIO pins
pins = [3, 5, 7, 29, 31, 26, 24, 21]
# Speed lights will blink (secs)
speed = .2

```

Code

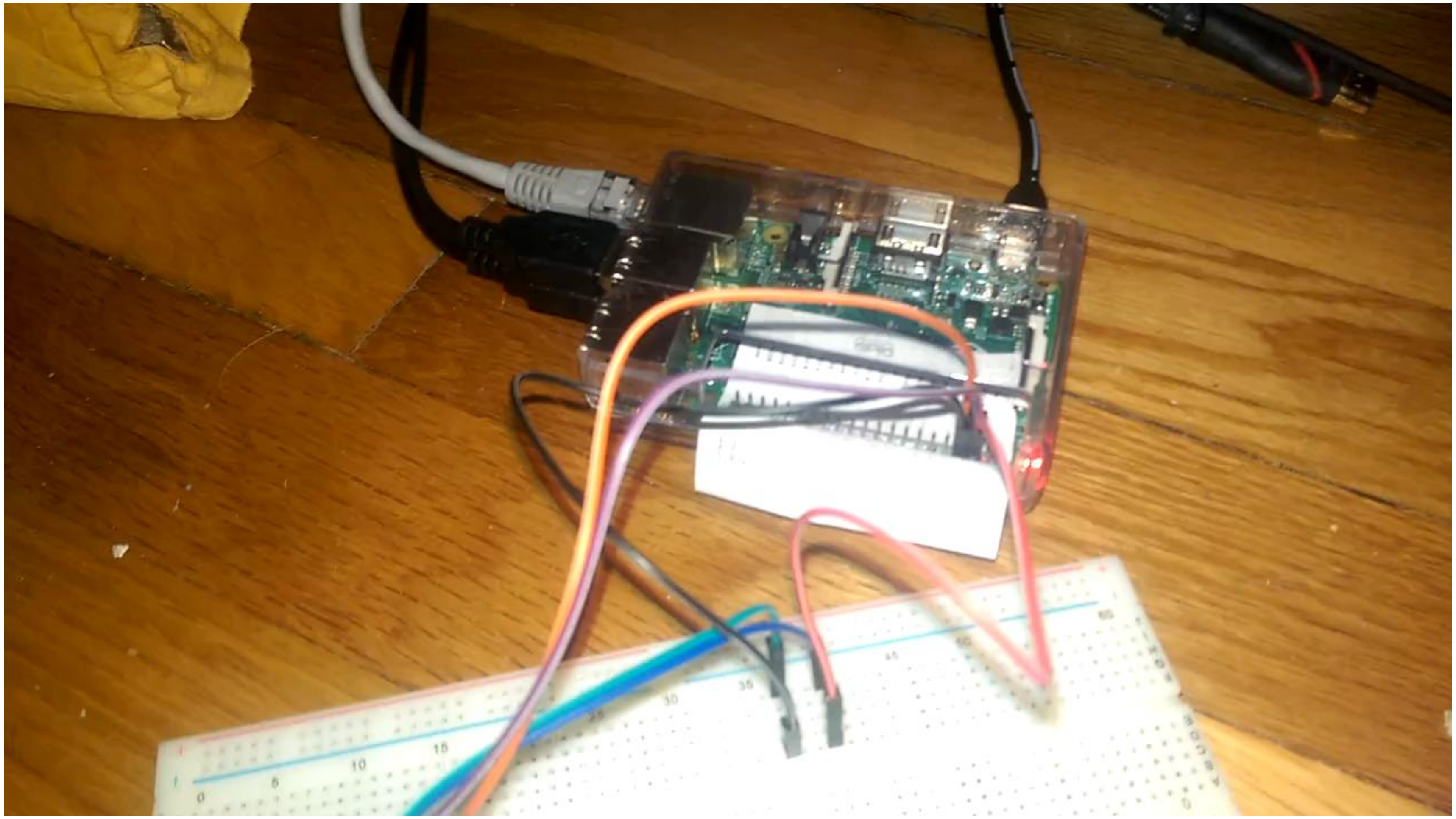
```
# Set GPIO pins to board (physical pins) mode
GPIO.setmode(GPIO.BOARD)

# Set up all pins as output pins
for i in pins:
  GPIO.setup(i, GPIO.OUT)
```

Code

```
while True:
  # Loop forward
  for i in range(len(pins)):
    GPIO.output(pins[i], True)
    time.sleep(speed)
    GPIO.output(pins[i], False)
  # Loop backward
  for i in range(len(pins)-1, -1, -1):
    GPIO.output(pins[i], True)
    time.sleep(speed)
    GPIO.output(pins[i], False)
```

Project 2 (LCD Screen) – Bouncy Ball

Demo



Materials List

- Raspberry Pi (any model)
- Power supply (I recommend at least 1.5A for this project)
- 4 jumper wires (female to female)
- 1 Breadboard (needed if you have other wires)
- 1 LCD screen
  - They vary in size/shape/color/etc. Use one with I2C or Serial capabilities

LCD Screen

- Uses 4x20 characters per screen
- Each character is 8x5 pixels
- Can add custom characters
- Really easy to use with serial or I2C chip on the back
- Does have small delay when writing/erasing screen

I^2C

- “Inter-Integrated Circuit”, pronounced “I squared C”
- Communications protocol for circuitry
- Uses master/slave bus
- Each device has an address and can be contacted by it
- Communication shares the same 2 wires for all devices connected: SDA, SCL
- A lot of support is built into various hardware (Raspberry Pi, Arudinos, Beaglebones, etc.)

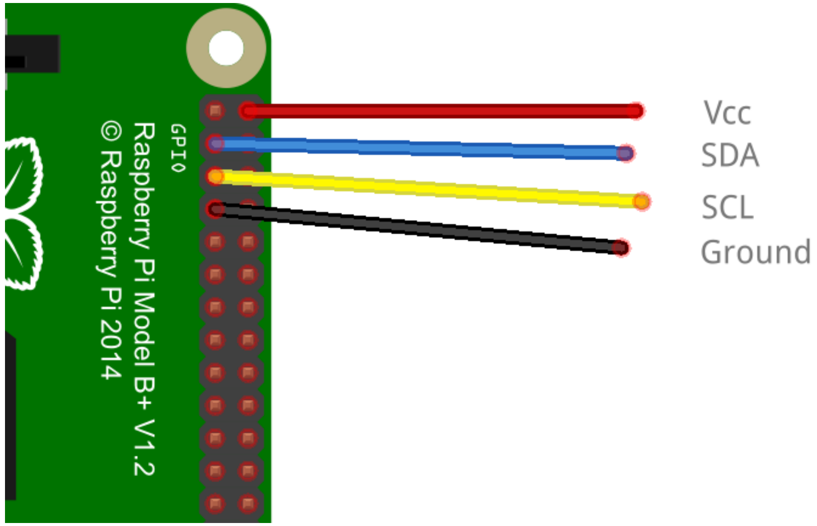
I^2C

- Must be enabled in Raspberry Pi first
- Run: sudo raspi-config
  - Choose Advanced options
  - Choose I2C
  - Choose to enable it
  - Choose to enable it on startup

Wiring

- No breadboard needed if using female-to-female wires
- Use breadboard as jumping point otherwise

Raspberry Pi	LCD Screen
+5V	Vcc
SDA (GPIO3)	SDA
SCL (GPIO5)	SCL
Ground	Gnd



Code

```
# Requires RPi_I2C_driver.py for I2C and LCD
# This is on my website
import RPi_I2C_driver
from time import *

# Create the object
mylcd = RPi_I2C_driver.lcd()
```

Code

```
# Print happy welcome message on lines 1 and 3
mylcd.lcd_display_string("Welcome to", 1)
mylcd.lcd_display_string(" Nebraska.Code() !!", 3)
sleep(2) # 2 sec delay

# Erase screen
mylcd.lcd_clear()
```

Code

```
# These are the char codes to start writing on each of the 4 rows
rows= [0x80, 0xC0, 0x94, 0xD4]

x = 0
y = -1
changex = 1
changey = 1

```

Code

```
while True:
    x = x + changex
    y = y + changey
    if x < 2:
        changex = 1
        x = 1
    if x >= 4:
        changex = -1
    if y < 1:
        y = 0
        changey = 1
    if y >= 19:
        changey = -1
```

Code

```
mylcd.lcd_clear()
mylcd.lcd_display_string_pos("o",x,y)
sleep(.5)
```

Project 3 (Sensors) – Distance Tracker

Demo

Well, actually… it’s not much of an exciting video…

Materials List

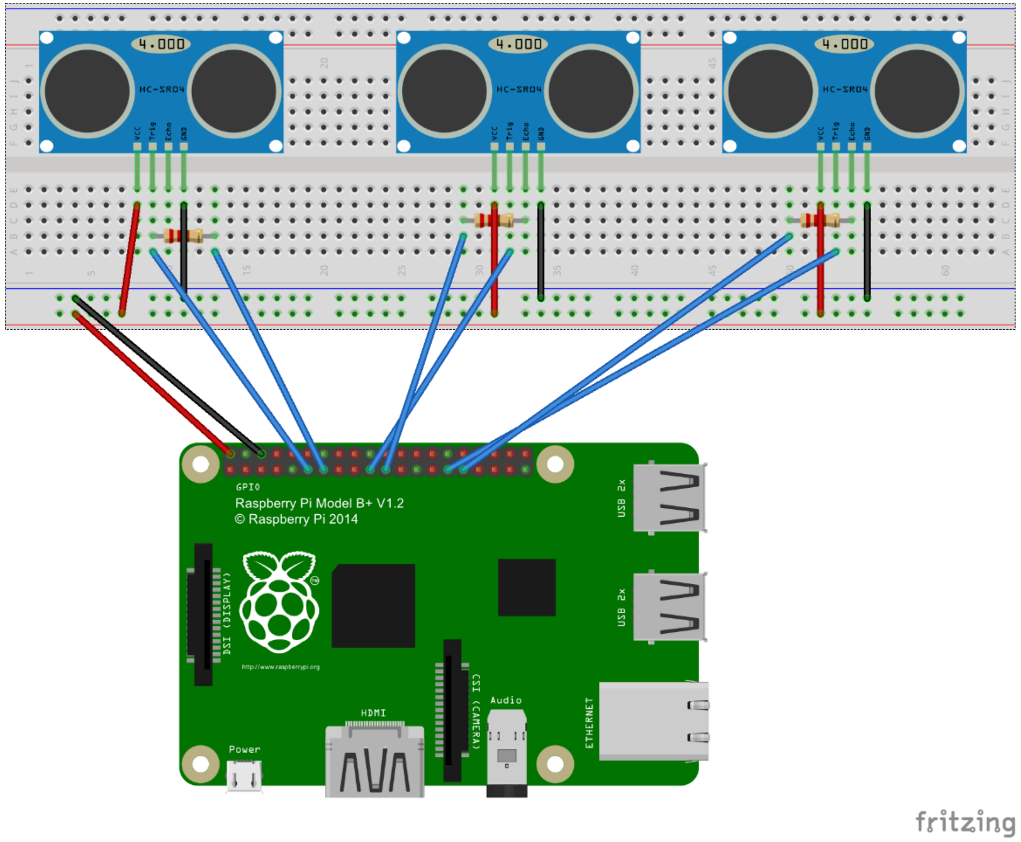
- Raspberry Pi (any model)
- Power supply (I recommend at least 1.5A for this project)
- 1 Breadboard
- 3 ultrasonic sensors (HC-SR04)
- Lots of jumper wires
- 3 1kΩ resistors

Ultrasonic Sensor

- Ultrasonic sensors are one of a variety of sensors you can use
- They detect distance with sound
- Ultrasonic = outside of human hearing range
- Pulses 40KHz signal from “T” side
- “R” side listens for it
- Time taken in between is time for sound to travel

Wiring

- Wire Ground from Pi to breadboard rail
- Wire +5V from Pi to breadboard rail
- Wire ALL ground on sensors to breadboard ground
- Wire ALL +5V on sensors to breadboard's +5V



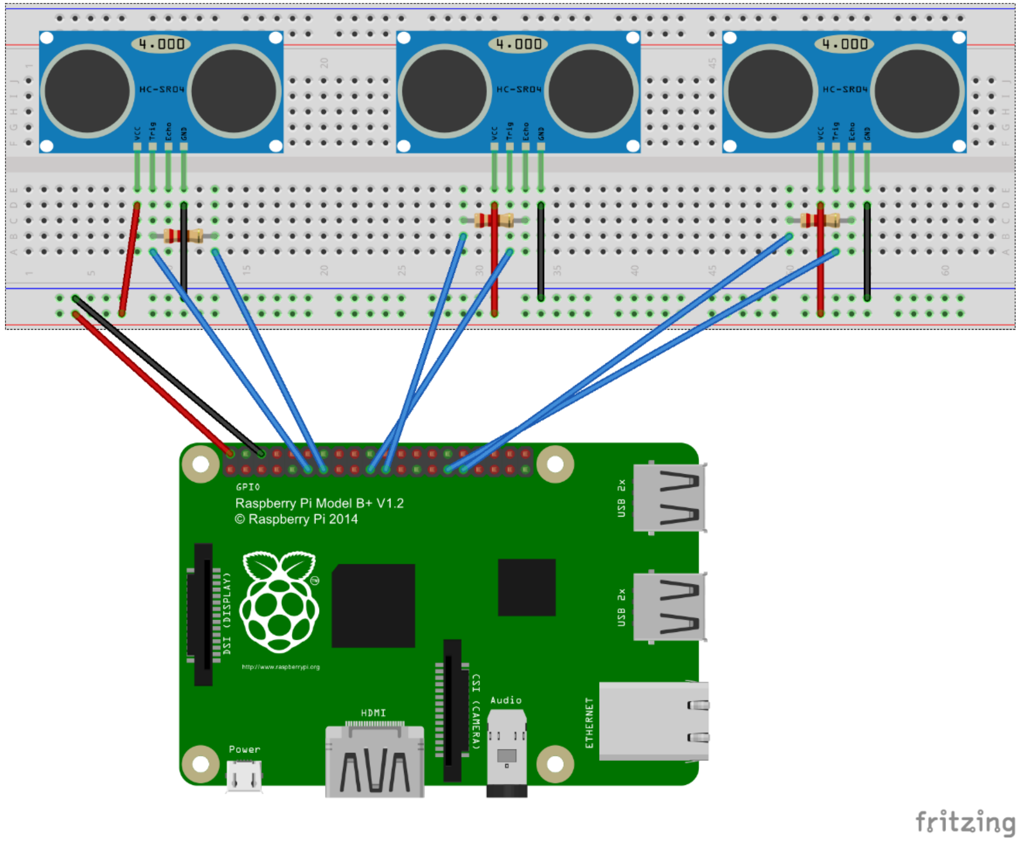
Wiring

› Trigger pins on sensors should go to GPIO pins

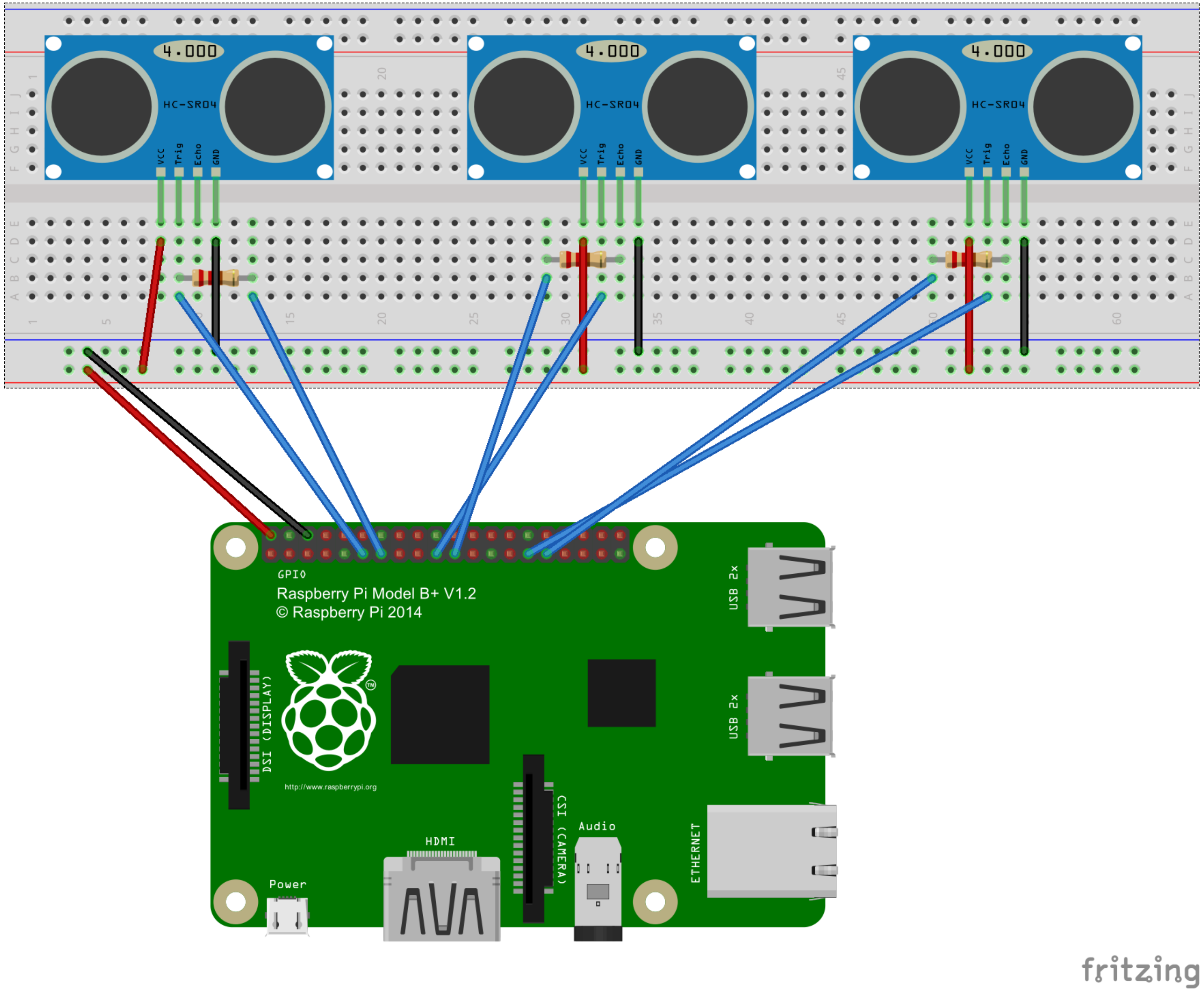
› Echo pins should go to 1kΩ resistors

› Resistors should go to GPIO pins

› Write down what is where, you’ll need it in a minute



Wiring


4.000
HC-SR04
VCC
Trig
Echo
GND
GPIO
Raspberry Pi Model B+ V1.2
© Raspberry Pi 2014
DSI (DISPLAY)
http://www.raspberrypi.org
CSI (CAMERA)
Audio
HDMI
Power
USB 2x
USB 2x
ETHERNET
fritzing

Code

```
import time
import RPi.GPIO as GPIO


# Set up the ultrasonic sensor pins
u1Trig = 11
u1Echo = 13
u2Trig = 19
u2Echo = 21
u3Trig = 29
u3Echo = 31

```

Code

```
def reading(trigger, echo):
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(trigger, GPIO.OUT)
    GPIO.setup(echo ,GPIO.IN)
    GPIO.output(trigger, GPIO.LOW)
    time.sleep(0.3)
```

Code

```
GPIO.output(trigger, True)
time.sleep(0.00001)
GPIO.output(trigger, False)

signaloff = 0
while GPIO.input(echo) == 0:
    signaloff = time.time()
```

Code

```
signalon = 0
while GPIO.input(echo) == 1:
    signalon = time.time()

timepassed = signalon - signaloff

distance = timepassed * 17000

return distance

```

Code

```
# Main program

while True:
    reading1 = reading(u1Trig, u1Echo)
    reading2 = reading(u2Trig, u2Echo)
    reading3 = reading(u3Trig, u3Echo)

    readingAvg = (reading1 + reading2 + reading3) / 3
    print(readingAvg)
```

More Ideas

Ideas

- Customized cookie maker machine
- Tracking cat/dog door
- Show tweets/emails on LCD screen
- AI-programmed remote control cars
- Robots… of any variety…
- Turn on/off heater/AC when a room is occupied/unoccupied

More Information and Conclusion

Lessons Learned

- Don’t plan a conference talk while still a very active student
- Hardware sometimes fails randomly. If software quits working, try testing the hardware
- Development always takes longer than you expect it to. Hacking projects are the same.

Conclusion

- Raspberry Pi is more than just a PC, it can interface with other electronics
- You've seen some sample projects
- You've seen how to code items like sensors, LCD screens, and LEDs
- Hopefully you're inspired to use these ideas to do your own projects

Contact Me

Twitter: @geekygirlsarah

Email: sarah@sarahwithee.com

Slides/Resources: sarahwithee.com/raspberrypi

- Feel free to contact with questions or to show off projects
- Let me know how this talk was and how to improve: http://nebraskacode.com/Evals/Submit?id=34
- Check out the projects before you leave!