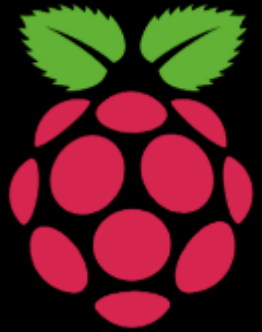
MAKER WORKSHOPS

INTRO to RASPBERRY PI



Instructors: Raphael Kopala, Shawn Nook

Unlondon Digital Media Assoc.

UnLondon
Digital Media Association

Enabling Exploration, Creativity, and Excellence In Art+Make+Tech

Challenging and embracing ideas related to new technologies and social platforms through the education, entertainment and engagement of our membership and the community-at-large.

- 121Studios: Coworking for Creatives
- Unlab: Hackerspace
- Events: STEAM Outreach & Edu., ExplodeConf, Nuit Blanche

Raphael

Day Job

Mechanical Engineer

- Working in the medical device industry
- Experience in medical device R&D and Manufacturing
- Teaching SolidWorks CAD at Fanshawe in evenings

Raphael

The Fun Stuff

Thinkier, Jack of all Trades – Master of None

- Arduino & RPi for Fun, and Odd Jobs
- 3D Printer Hobbyist
- PC Builder & Gamer
- Fish keeper

Shawn

Day Job

Freelance Embedded Systems Engineer

- Indoor location tracking w/ Bluetooth
- Keychain / Fitness Band Widgets
- Joystick for VR
- Remote Controls
- Internet of S*#t

Shawn

The Fun Stuff

Hacker, Church of the Weird Machine, Odd Duck

- Arduino compatible implant
- EEG Games / Toy Hacking
- Brain Stimulation
- Be Weird, Make Weird, Have Fun!

RASPBERRY PI
Essentials

RASPBERRY PI
ESSENTIALS

What is it?

The Raspberry Pi is a low cost credit-card sized computer.

Wide range of possibilities
- Normal computing capabilities (web browsing, spreadsheets)
- Interact with external world through sensors and actuators

Created by Raspberry Pi Foundation 

RASPBERRY PI
ESSENTIALS

Accessories

Display

Camera

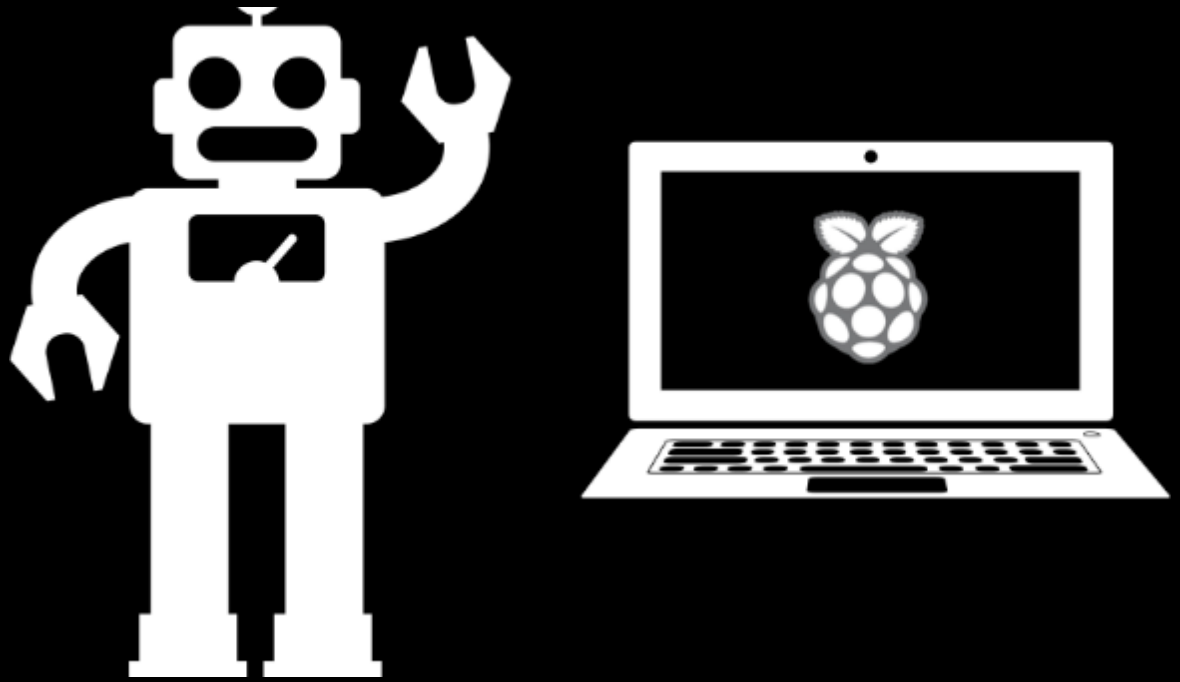
Motor Controller

Remote Control

WiFi

Case

RASPBERRY PI
WHAT CAN YOU DO



RASPBERRY PI

What Can You Do?

Projects

Mini-Laptop

Robot

Mini-Arcade

Home Automation

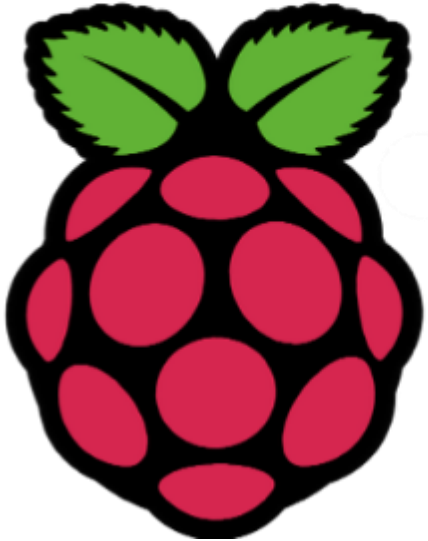
Radio

LED

RASPBERRY PI

What Can You Do?

Brainstorming time!

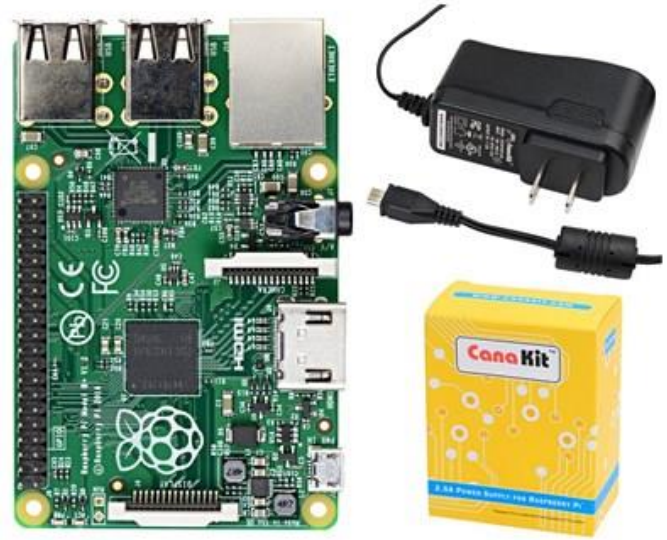


+ ____ = ?

RASPBERRY PI
SETUP

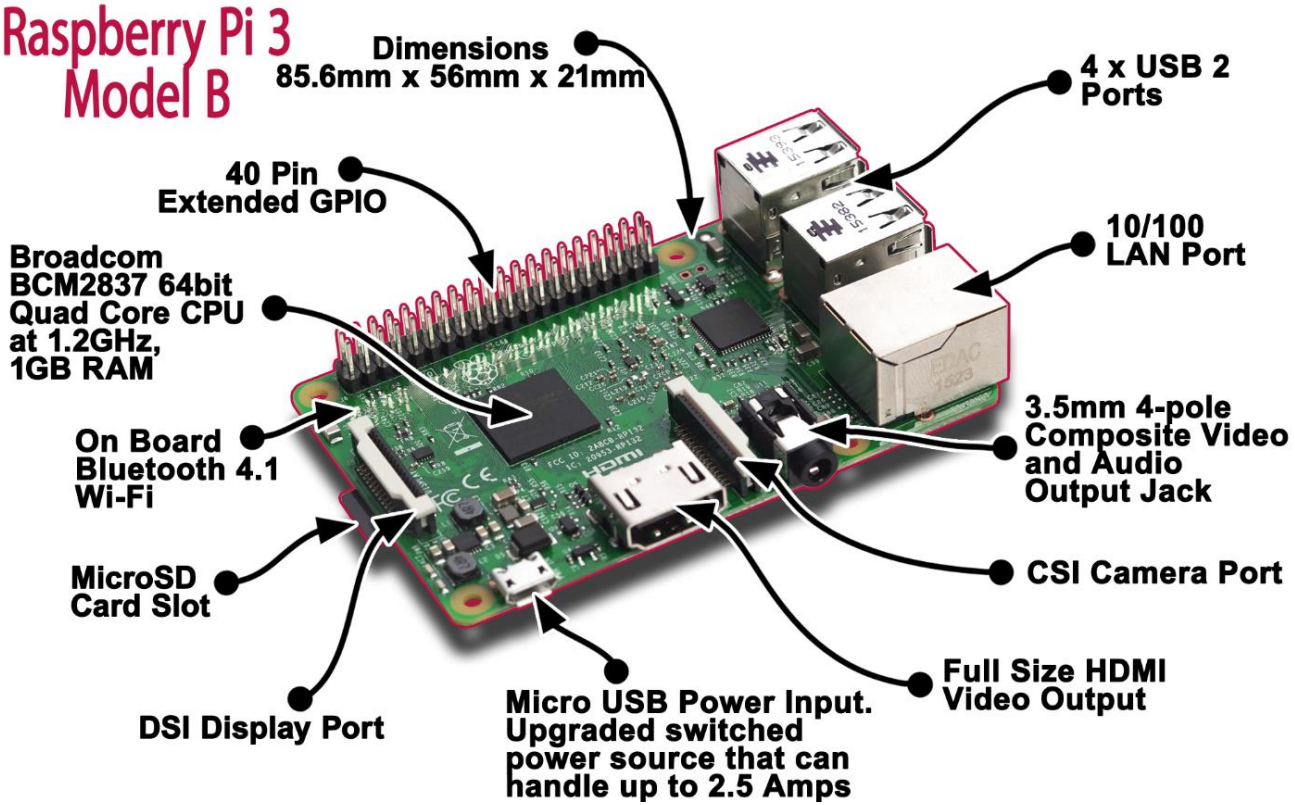
RASPBERRY PI
SETUP

Your Kit



RASPBERRY PI
ESSENTIALS

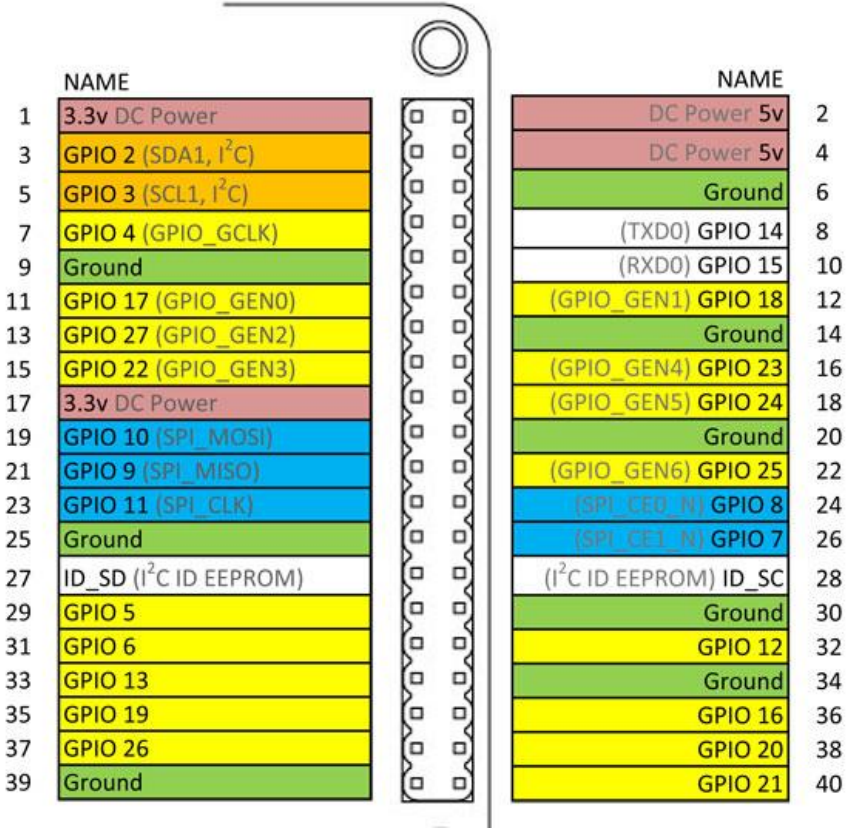
Components



http://www.memoryexpress.com/Products/MX61461

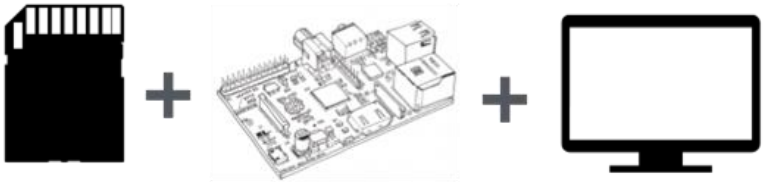
RASPBERRY PI
GPIO

http://blog.mcmelectronics.com/image.axd?picture=%2F2016%2F03%2FGPIO-Chart2.jpg



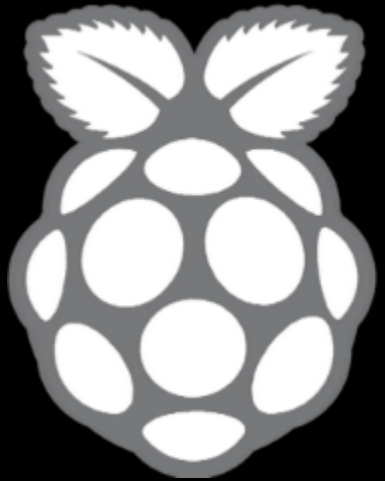
RASPBERRY PI
SETUP

1. GET CONNECTED



2. PLUG IT IN

RASPBERRY PI
LINUX BASICS

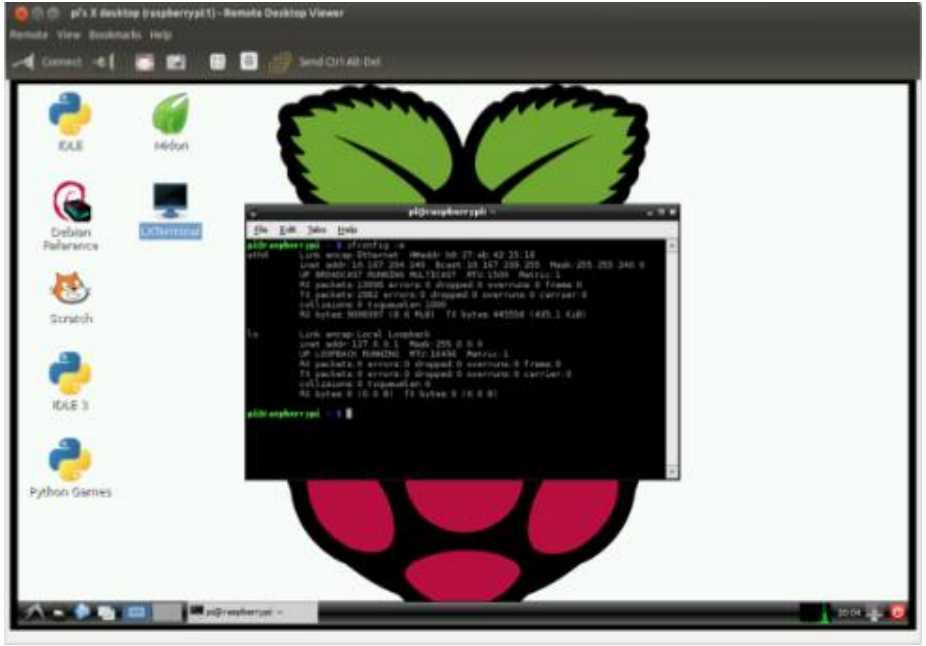


RASPBERRY PI

Linux Basics

- Similar to Windows or Mac
- GUI or console
- Good linux command intro:

www.ee.surrey.ac.uk/Teaching/Unix/



RASPBERRY PI
Linux Basics

Linux Command	Description
`cd 'directory'`	Change to '*directory*'
`cd ..`	Change to parent directory
`ls`	List current directory contents
`pwd`	Print name of current directory
`mkdir 'directory'`	Create '*directory*'
`rmdir 'directory'`	Remove '*directory*'
`rm 'file'`	Remove '*file*'

RASPBERRY PI

Linux Basics

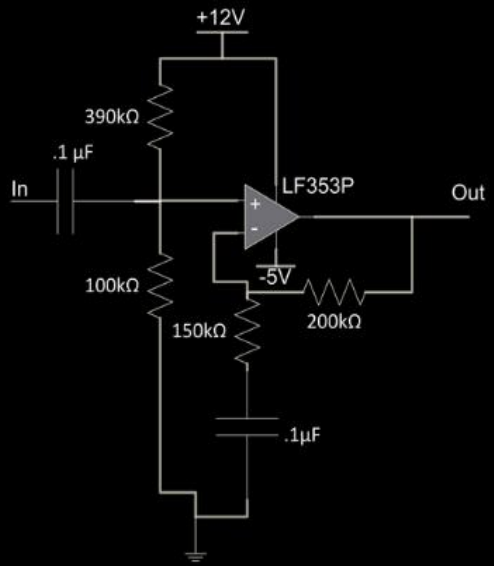
Linux Command	Description
`sudo 'command'`	Run a '*command*' as super user
`nano 'file'`	Use text editor to create/edit '*file*'
`cp 'file_1' 'file_2'`	Copies file_1 as file_2
`startx`	Start Rasbian Desktop
`sudo shutdown`	Shutdown
`sudo reboot`	Reboot
`ifconfig`	View Networking Information
`sudo apt-get update`	Update repositories
`sudo apt-get upgrade`	Upgrade repositories

Login + Launch GUI

Username: pi
Password: raspberry (*Note: keystrokes will not show up for the password)

Type startx to start the rasbian gui

RASPBERRY PI
Electronics Basics



RASPBERRY PI
Electronics Basics

Current
- Measured in Amps (A)
- Represents the speed, or flow of charge in a circuit

Voltage
- Measured in Volts (V)
- Represents potential, or pressure applied to a charge in a circuit

Resistance
- Measured in Ohms (Ω)
- Represents the resistance in a circuit that impedes the flow of charge

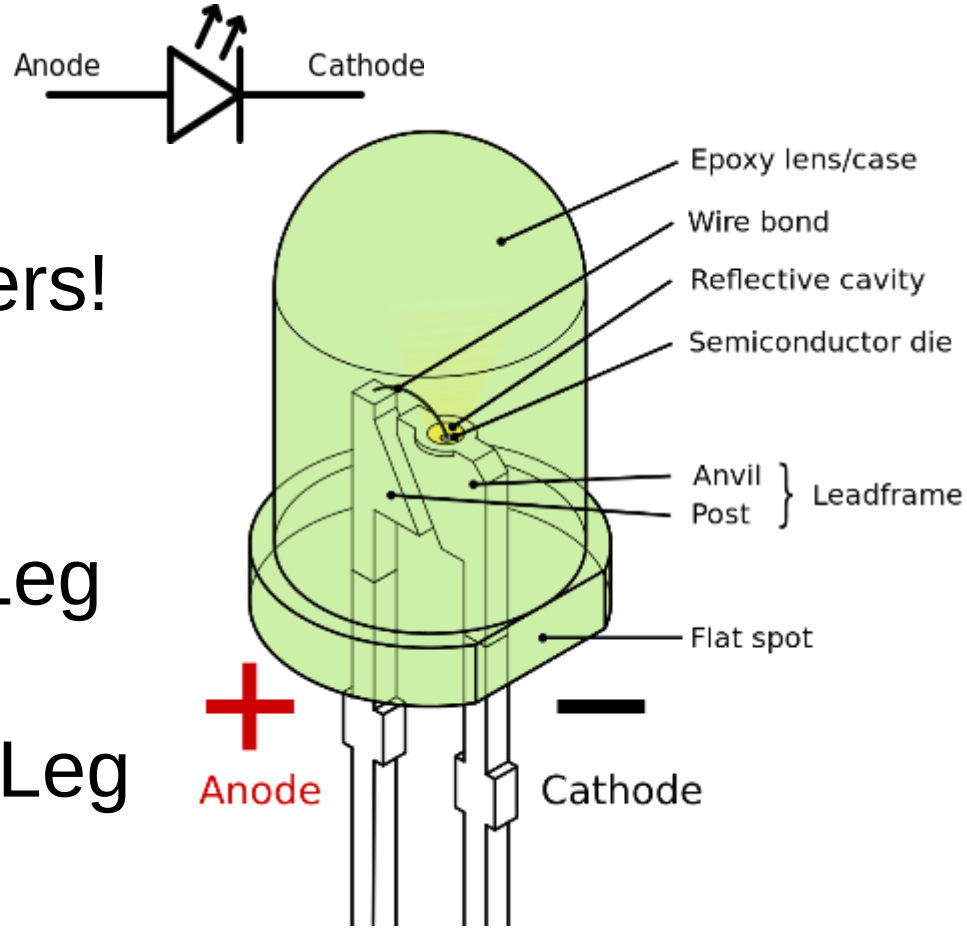
LED / Diode

Direction/polarity matters!

- Like a one way valve

Positive (+) = Longer Leg

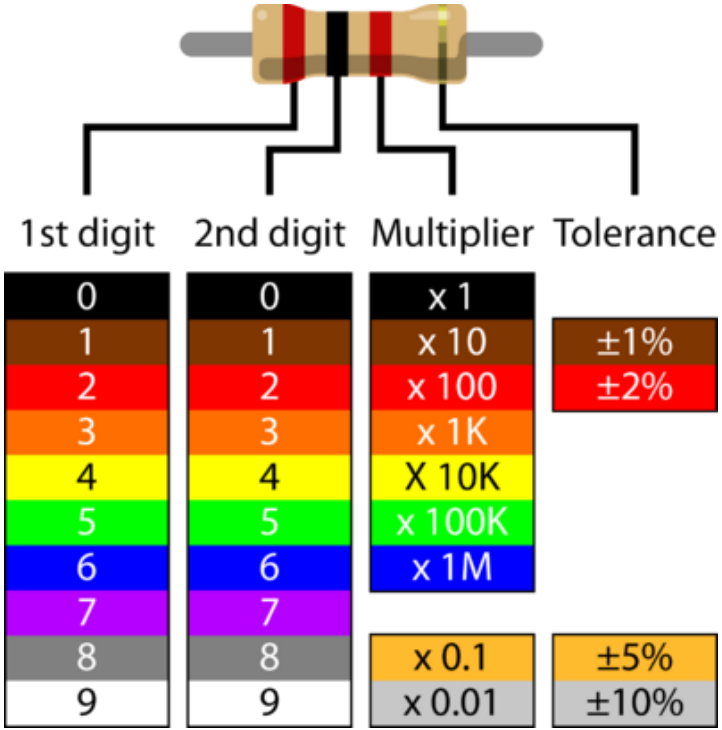
Negative (-) = Shorter Leg



Resistors

- Direction/polarity doesn’t matter
- Each has a fixed value
- Value indicated by coloured bands

Resistor Colour Bands



Example; what is the resistance of the resistor below?

Using the chart:

Brown, Black, Brown, Gold

1 + 0 x 10 ± 5%

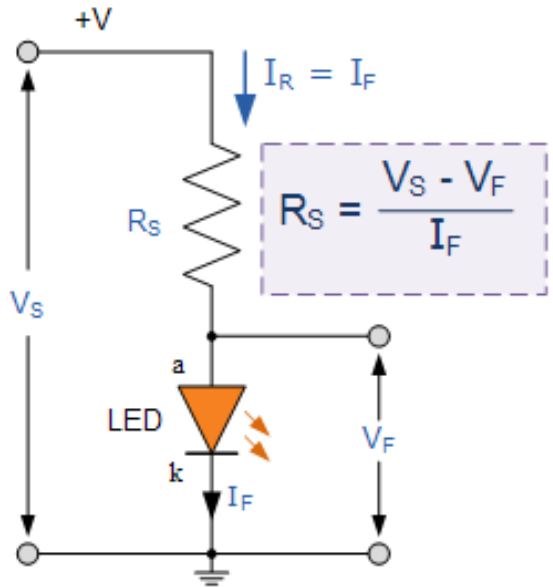
Results in:

100 +/- 5% ohms

Between (95-105 ohms)

Source:
http://www.education.rec.ri.cmu.edu/content/electronics/common/resistors/images/400px-Resistor_color_code_chart.png

Calculating Resistors for LED's



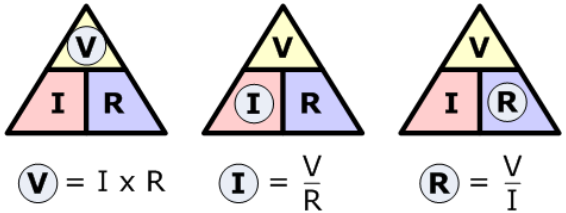
Source:
http://www.electronics-tutorials.ws/diode/diode_8.html
http://www.electronics-tutorials.ws/dccircuits/dcp23.gif

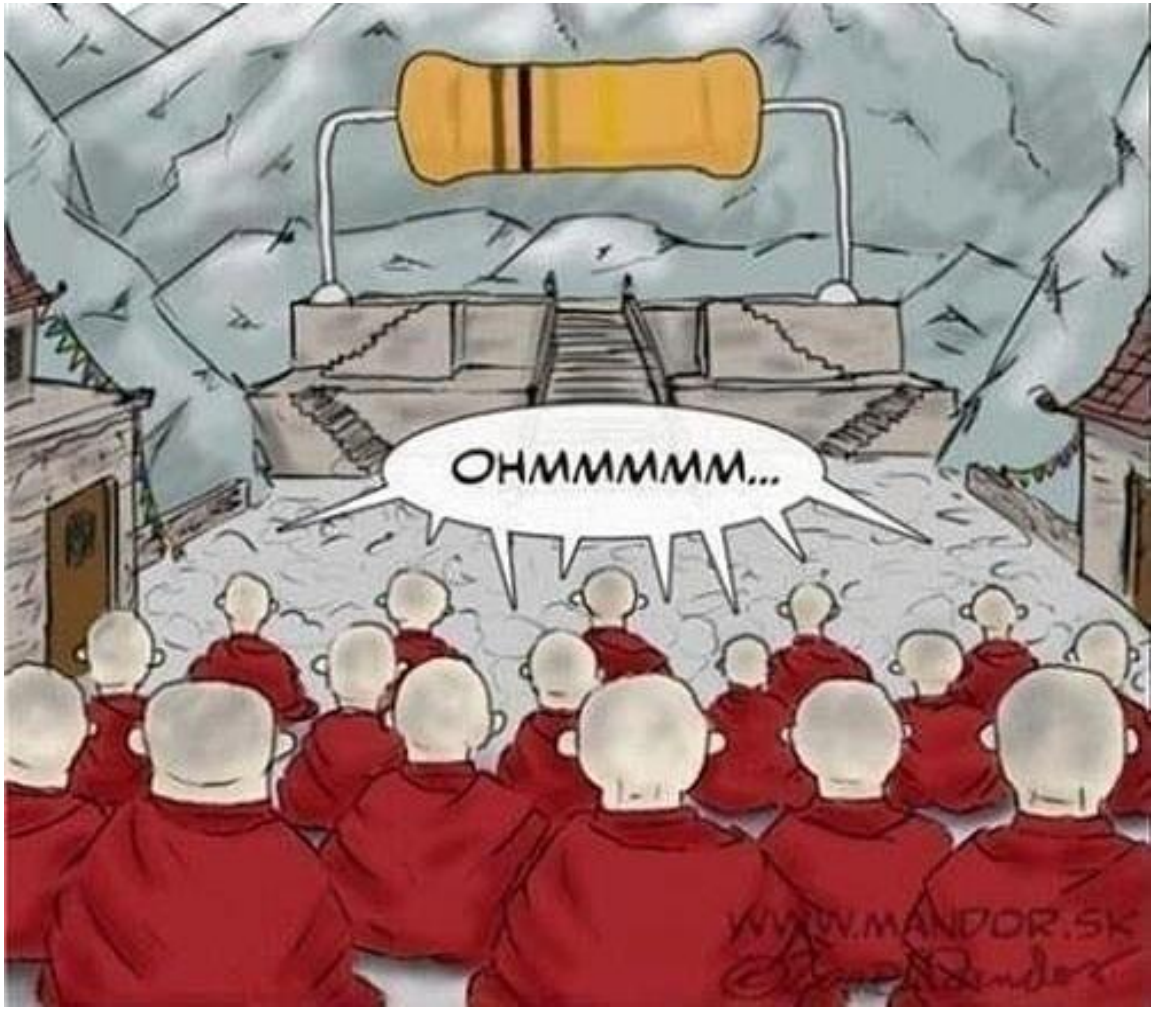
Example: 3.3V Supply with a 1.8V LED @ 0.01A

$$R_S = \frac{V_S - V_F}{I_F} = \frac{3.3V - 1.8V}{10mA} = \frac{3.3V - 1.8V}{0.01A} = 150\Omega$$

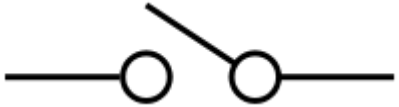
Most LEDs are specified with a forward voltage (Vs) to turn on the LED, and a maximum current draw (If)

Remember; (ohms law)

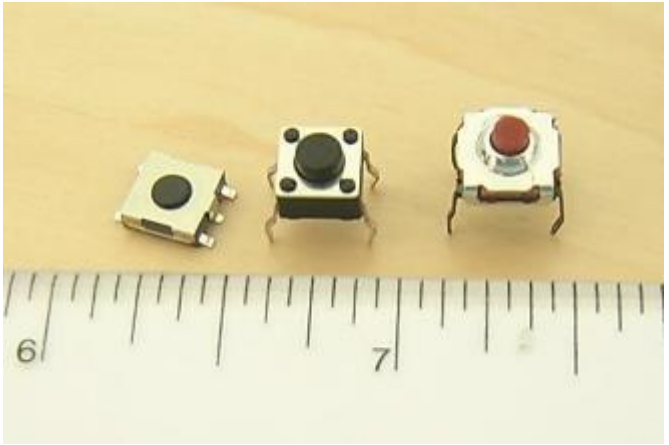


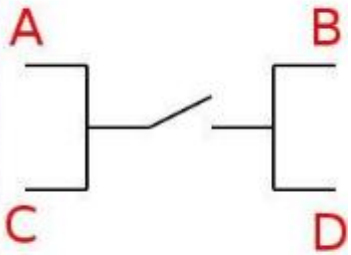

OHMMMMM...
WWW.MANDOR.SK

Switch



- On/Off
- Momentary vs. Toggles





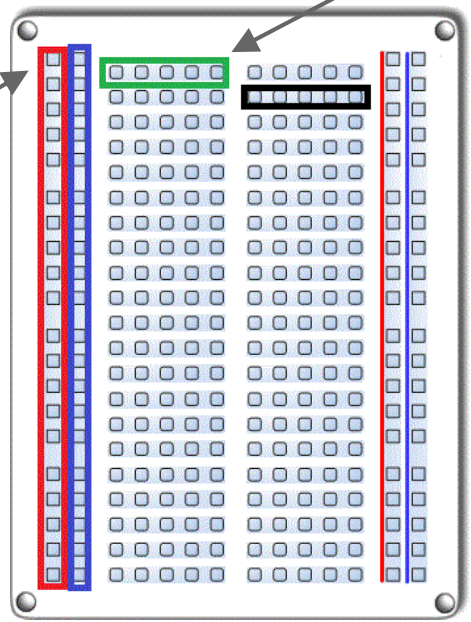
Pins A & C, and B & D are connected always.

When pressed. All pins are connected.

Source:
http://www.varesano.net/files/imagecache/500width/pushbutton_legs_final.jpg

Breadboards

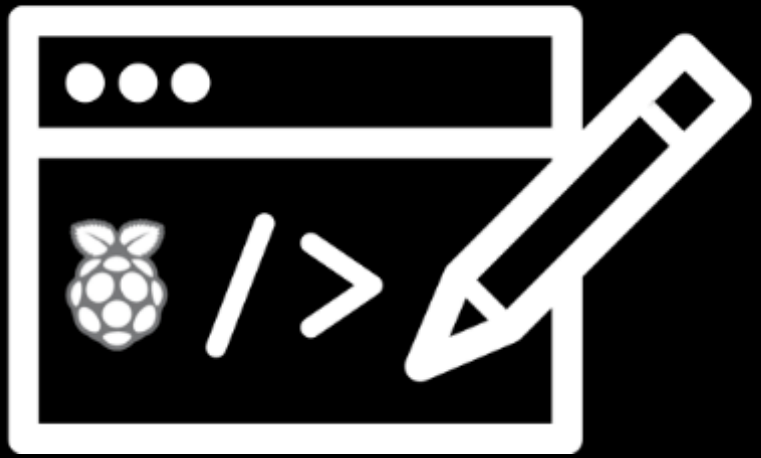
Short horizontal lines are connected



Long vertical lines are connected. Usually for power(+) and ground (-)

Source:
http://cdn.instructables.com/F1G/DVCP/HPZLM93S/F1GDVCPHPZLM93S.MEDIUM.gif

RASPBERRY PI
DEMO APP



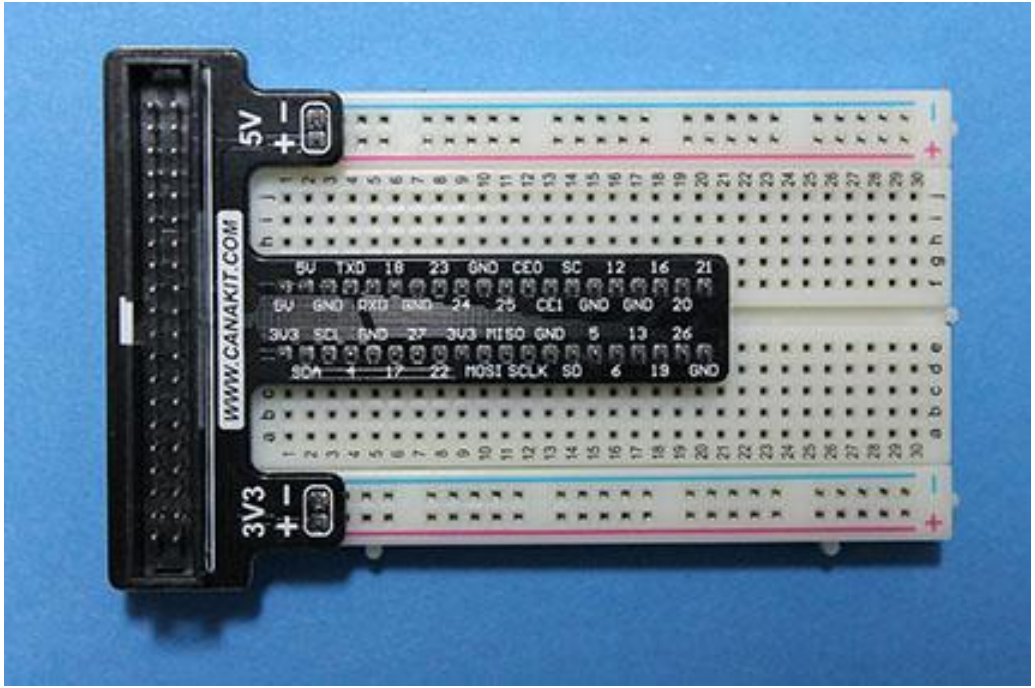
Programming

- We'll try a few basic programs to see how we can interact with lights and switches
- Programs are created using the python programming language

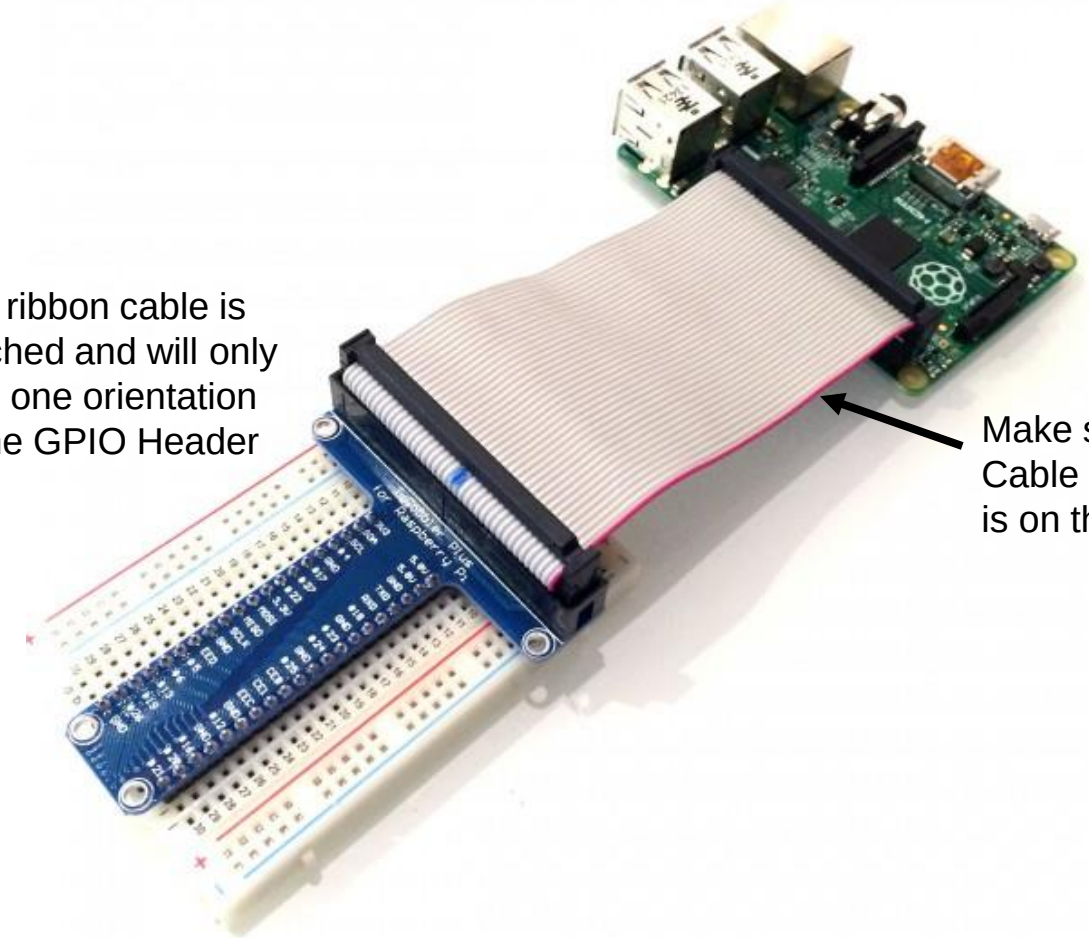
Caution!

- GPIO pins can supply up to 16 mA of current safely
- Do not exceed more than 50mA across all GPIO pins
- GPIO pins operate on 3.3V, never input more than 3.3V
- Keep this in mind to avoid burning your pi

Connect your GPIO Header Like This



The ribbon cable is notched and will only fit in one orientation in the GPIO Header

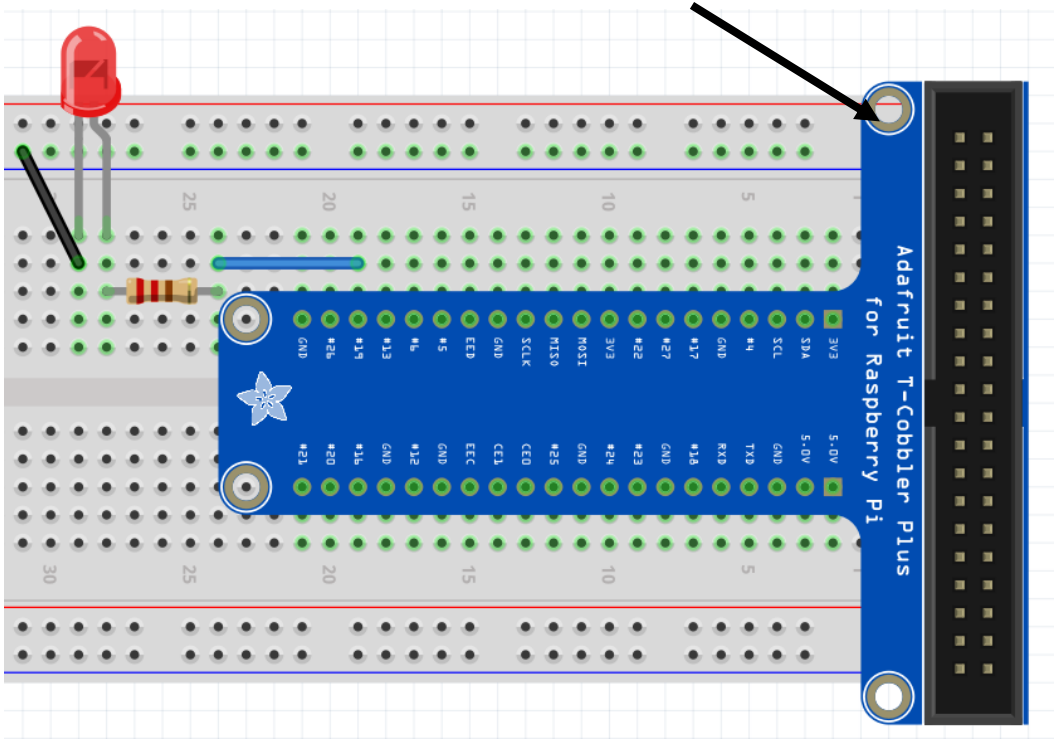


Make sure the Red Cable on the Ribbon is on this side

Project 1 Blinking LED

- Use a 220 Ohm resistor (Red – Red – Brown)

Your breakout board looks different. The 3.3V will be on this side.



Project 1 Blinking LED

In a terminal, type

sudo nano blinker.py

Type the code →

```
import RPi.GPIO as GPIO
import time
import atexit

ledPin = 19

def cleanup():
        GPIO.cleanup()

atexit.register(cleanup)

GPIO.setmode(GPIO.BCM)
GPIO.setup(ledPin, GPIO.OUT)

while (True):
        GPIO.output(ledPin,True)
        time.sleep(0.5)
        GPIO.output(ledPin,False)
        time.sleep(0.5)
```

Ctrl-O to save, Ctrl-X to exit

Type '*sudo python blinker.py*' to run . >**Press Ctrl-C to cancel**

Programming Note: Variables

Python variables do not need to be explicitly defined as a particular data type. When assigning variables the python interpreter allocates the appropriate memory for each variable (integers, decimals, chars).

```
<variableName> = <initial value>
```

Programming Note: Functions

A function is grouping reusable code to preform an action and allow the re-use of existing code. Functions may be input a value(s) upon which an operation is made, and a value may be returned.

```
def functionName(parameters):
        code
        return [expression]    (optional)
```

Programming Note: Python Modules

A module can be a grouping of functions, variables and classes.

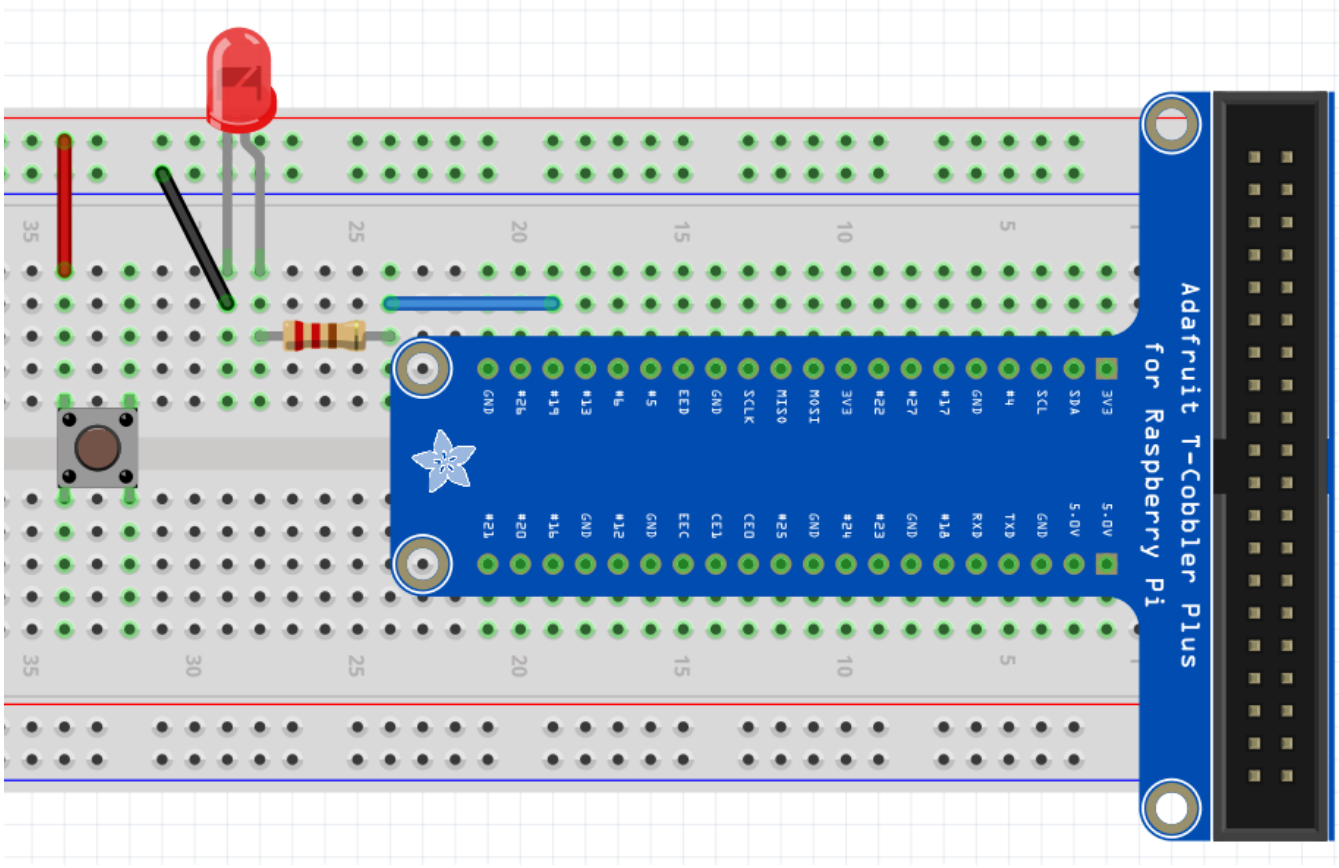
Modules may be imported and loaded into memory at the start of your code:

```
import module
```

Variables and functions within the module can then be accessed using the . (dot) operator.

```
module.function1(paramter)
```

Project 2 Button



Project 2 Button

In a terminal, type

sudo nano button.py

Type the code →

```
import RPi.GPIO as GPIO
import atexit

ledPin = 19
buttonPin = 23

def cleanup():
        GPIO.cleanup()

atexit.register(cleanup)

GPIO.setmode(GPIO.BCM)

GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(ledPin, GPIO.OUT)

lastState = False

while (True):
        currentState=GPIO.input(buttonPin)
        if lastState != currentState:
                GPIO.output(ledPin,currentState)
                print(currentState)
                lastState = currentState
```

- Ctrl-O to save, Ctrl-X to exit
- Type '*sudo python button.py*' to run. > Press Ctrl-C to cancel

Project 2 Button

GPIO.PUD_DOWN vs GPIO.PUD_UP

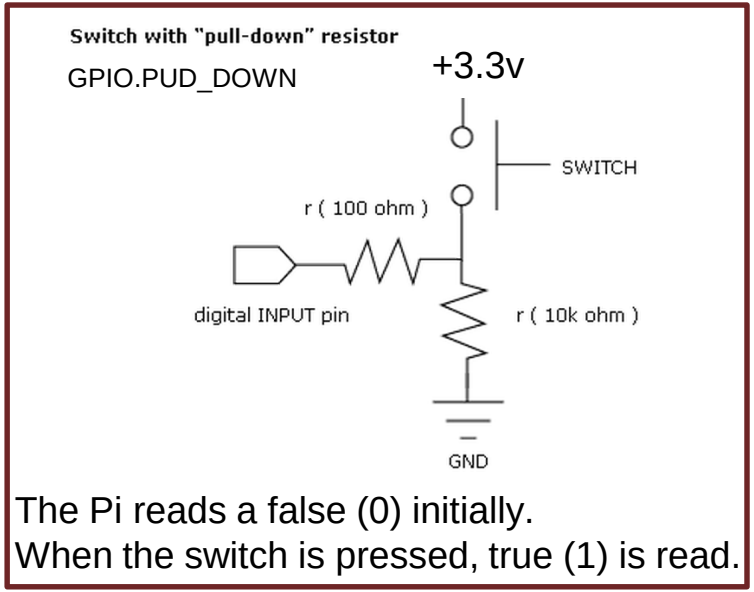
- What's this?

```
GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

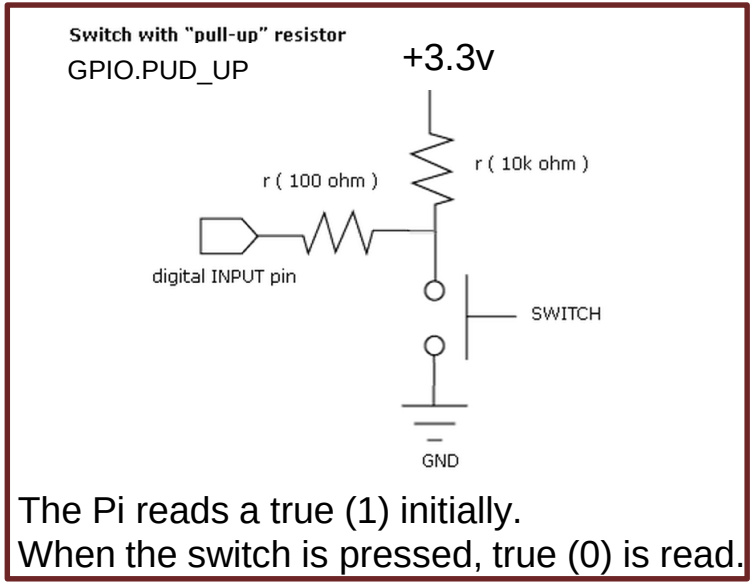
- Raspberry Pi has internal pull up, or pull down resistors configurable withing the programming
- This ensures:
  - the Pi is always reads a 1 or a 0 (floating pin)
  - you are not shorting 3.3v to ground (path of least resistance)
- Configured as either:
  - GPIO.PUD_DOWN or GPIO.PUD_UP
- This can be instead done by physical resistors (next slide)

Project 2 Button

Pull Up vs Pull Down Resistors



The Pi reads a false (0) initially.
When the switch is pressed, true (1) is read.



The Pi reads a true (1) initially.
When the switch is pressed, true (0) is read.

Programming Note: If Statements

Decision making can occur within `if` and `if-else` statements. When an `if` statement is determined to be true, it's contents are executed. If the statement is determined to be false, the inner code is skipped and an optional `else` statement is executed if the `if` statement was false.

```
if statement:
        code executed if statement is true
else:
        code executed if statement is false
```

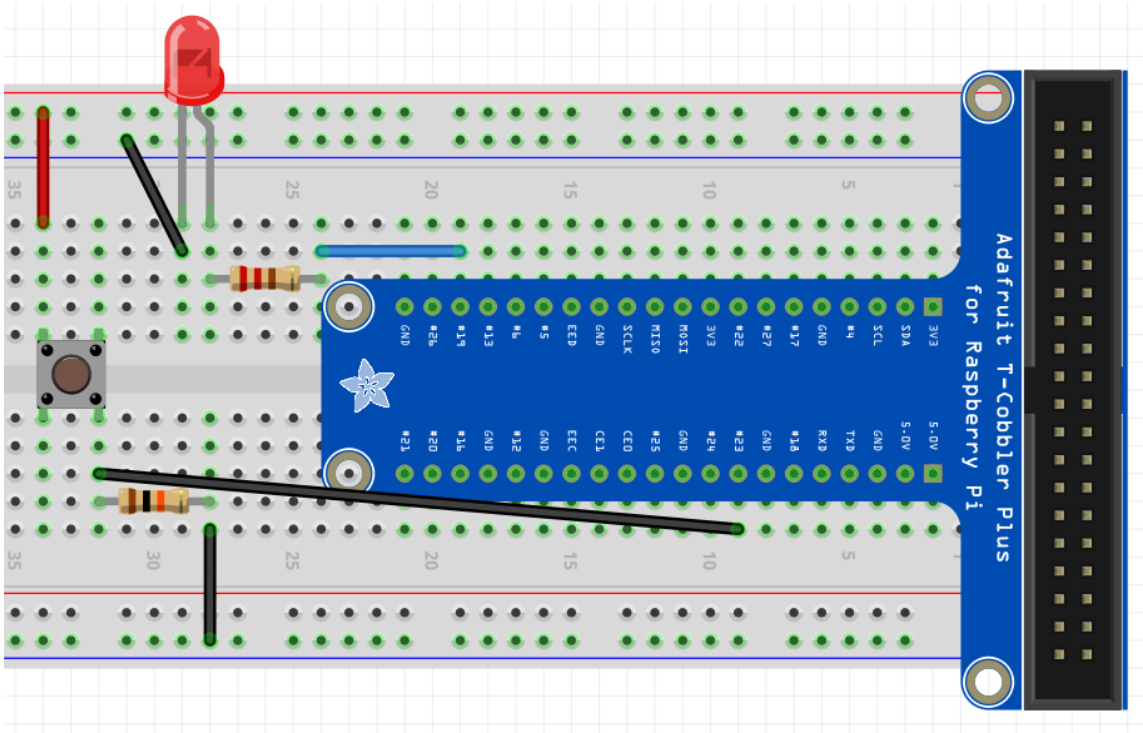
Programming Note: Comparison Operators

Comparison operators are used within statements to test conditions. The result is either a true, or a false result.

Operators	
==	Equal to
!=	Does not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Project 2 Button

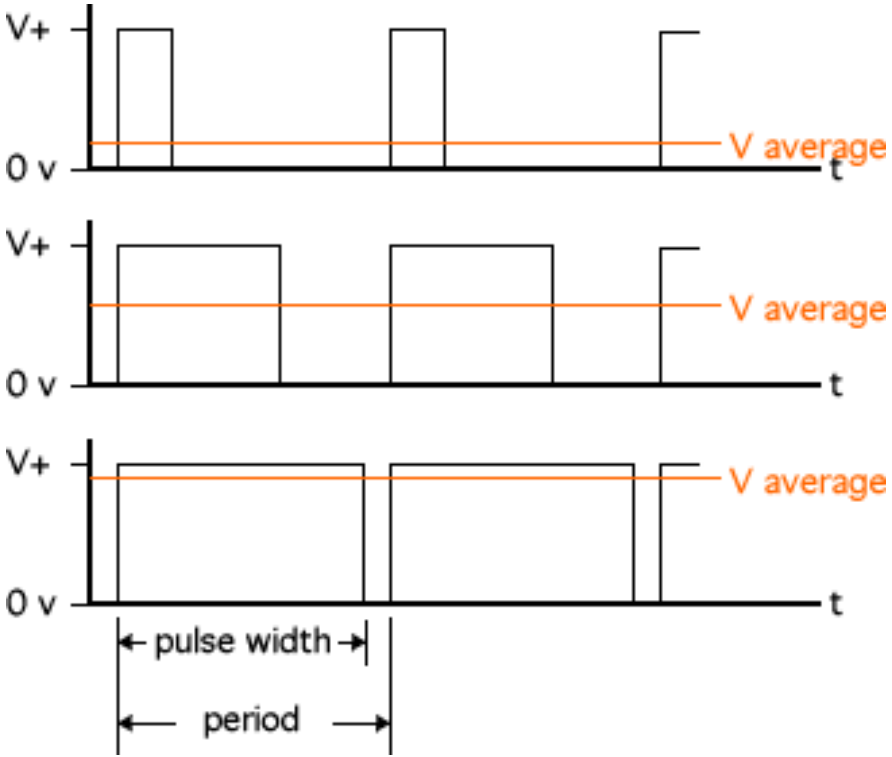
Physical Wiring Pull Down (FIY)



Project 3 PWM

Pulse Width Modulation

- We can use our last circuit
- Pulse width modulation turns the pin on and off very fast:
  - The period defines the frequency of the pulse
  - The pulse width is the percent of time the signal is on (also called duty cycle).
  - This can make an LED appear dimmer, or motor spin slower



Project 3 PWM

In a terminal, type

sudo nano pwm.py

Type the code →

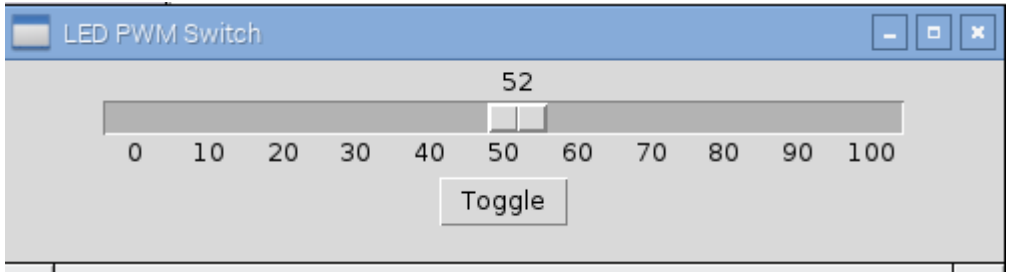
```
import RPi.GPIO as GPIO
import atexit

ledPin = 19

def cleanup():
        GPIO.cleanup()

atexit.register(cleanup)

GPIO.setmode(GPIO.BCM)
GPIO.setup(ledPin, GPIO.OUT)

pwmLed = GPIO.PWM(ledPin, 500)
pwmLed.start(100)


while (True):
        input = raw_input("Enter brightness (0-100):")
        duty = int(input)
        if (duty>=0 and duty<=100):
                pwmLed.ChangeDutyCycle(duty)
        else:
                print "Try Again"
```

- Ctrl-O to save, Ctrl-X to exit
- Type '*sudo python pwm.py*' to run. > Press Ctrl-C to cancel

Project 4 GUI Interface

- We will be using the Tkinter (“Tk interface”) module to create a GUI interface from within Rasbian

- The interface will change the PWM of the LED, toggle the LED state from on to off, and notify us if the button has been pressed.



Project 4 GUI Interface

- For this project, lets use the Python IDLE (Integrated Development Environment)
- Accessed from Menu > Programming > Python 2
- Create File, File > New
- Save File
- To Run, Run > Run Module

Project 4 GUI Interface (Code 1 of 4)

```
from Tkinter import *
import RPi.GPIO as GPIO
import atexit
import tkMessageBox

ledPin = 19
buttonPin = 23
storedPwm = 0

def cleanup():
        GPIO.cleanup()
        win.destroy()

def update(duty):
        pwm_led.ChangeDutyCycle(int(duty))
```

Project 4 GUI Interface (Code 2 of 4)

```
def toggle():
        if scale.get() > 0:
                global storedPwm
                storedPwm= scale.get()
                scale.set(0)
                pwm_led.ChangeDutyCycle(0)
        else:
                pwm_led.ChangeDutyCycle(int(storedPwm))
                scale.set(storedPwm)

def task():
        win.update()
        if GPIO.input(buttonPin):
                tkMessageBox.showwarning("Warning!","Button Hit!")
```

Project 4 GUI Interface (Code 3 of 4)

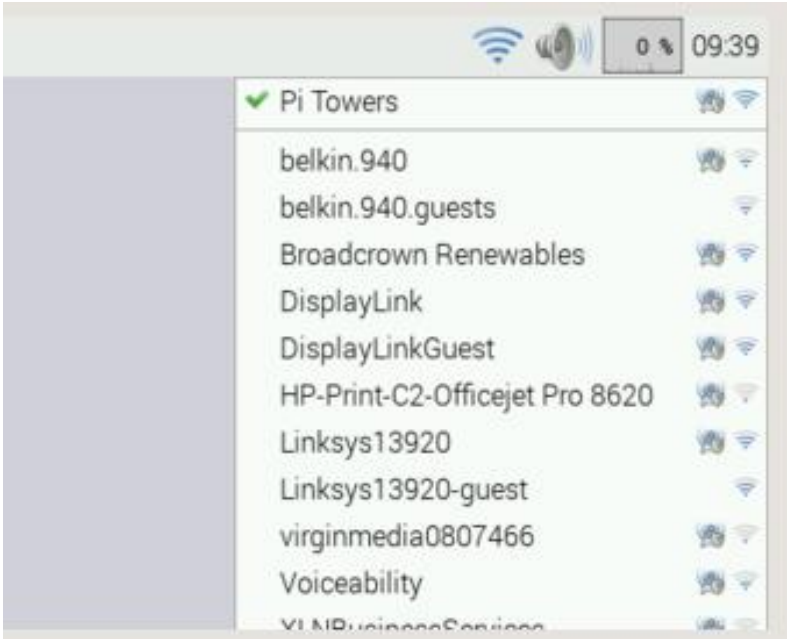
```
atexit.register(cleanup)

GPIO.setmode(GPIO.BCM)
GPIO.setup(ledPin, GPIO.OUT)
GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

pwm_led = GPIO.PWM(ledPin, 500)
pwm_led.start(100)

win = Tk()

win.title("LED PWM Switch")
win.geometry("500x100")
win.protocol("WM_DELETE_WINDOW", cleanup)
```

Project 4 GUI Interface (Code 4 of 4)

```
scale = Scale(win, from_=0, to= 100, orient=HORIZONTAL,
              length = 400, tickinterval=10, command=update)

scale.pack()
scale.set(100)

toggleButton = Button(win, text="Toggle", command=toggle)
toggleButton.pack()

while 1:
        task()
```

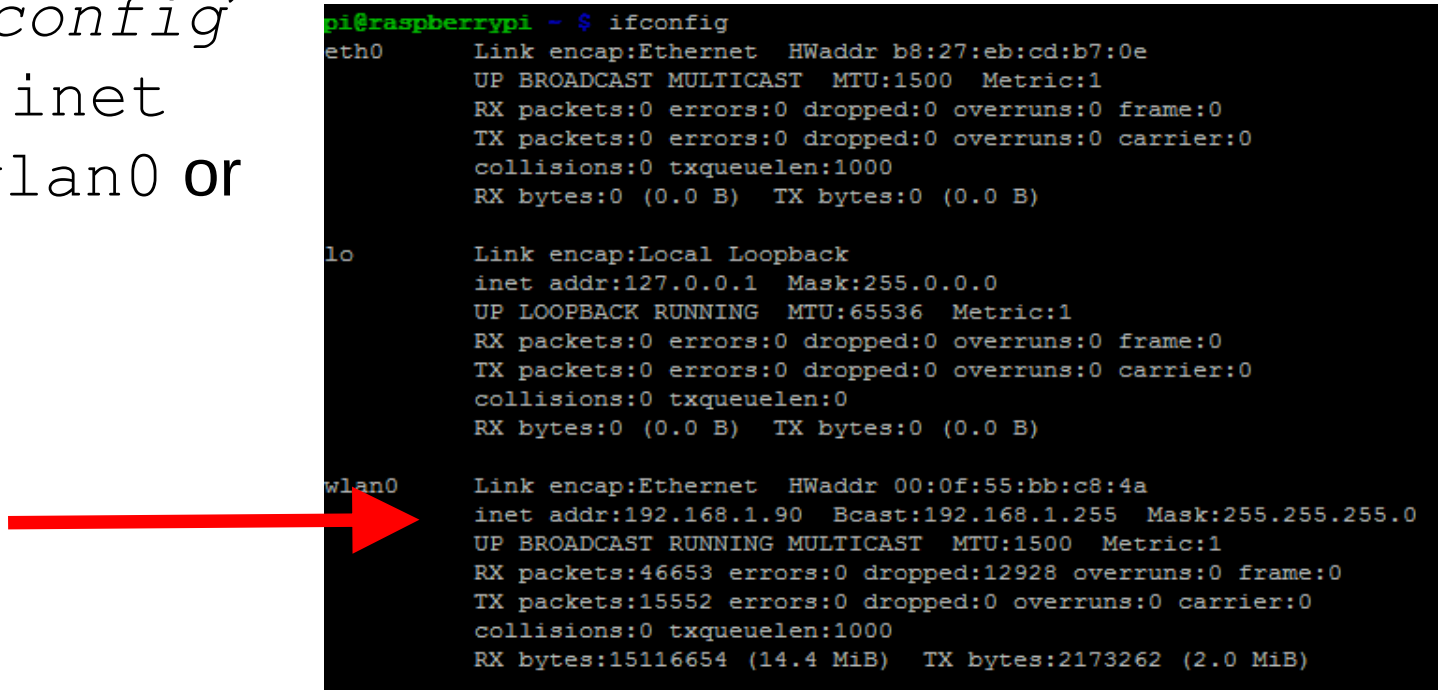
Setting Up Wifi

- Select Wifi dropdown from top bar

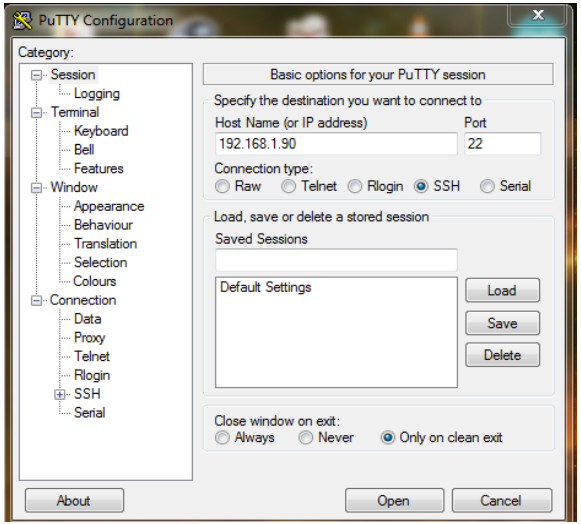


Find IP Address

- Run '*ifconfig*'
- Look for `inet` beside `wlan0` or `eth0`



Remote Login – SSH, Port 22



Windows - Install and Run ‘Putty’
Use port 22 with SSH
http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

Mac/Linux. Type:

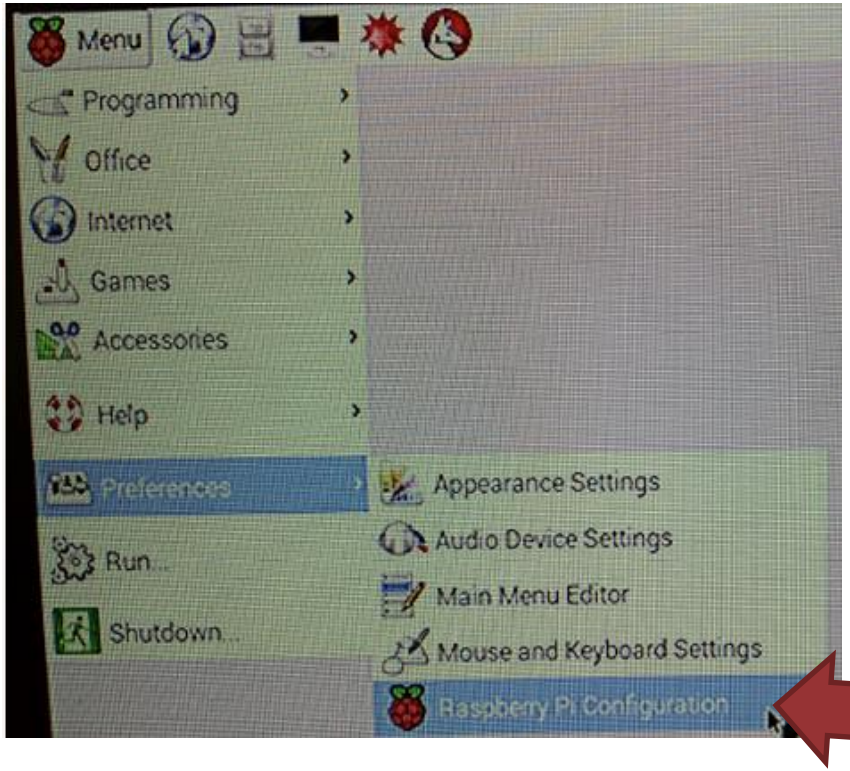
```
ssh <ip address of your Rpi> -l <username of RPi user you set up previously>
```

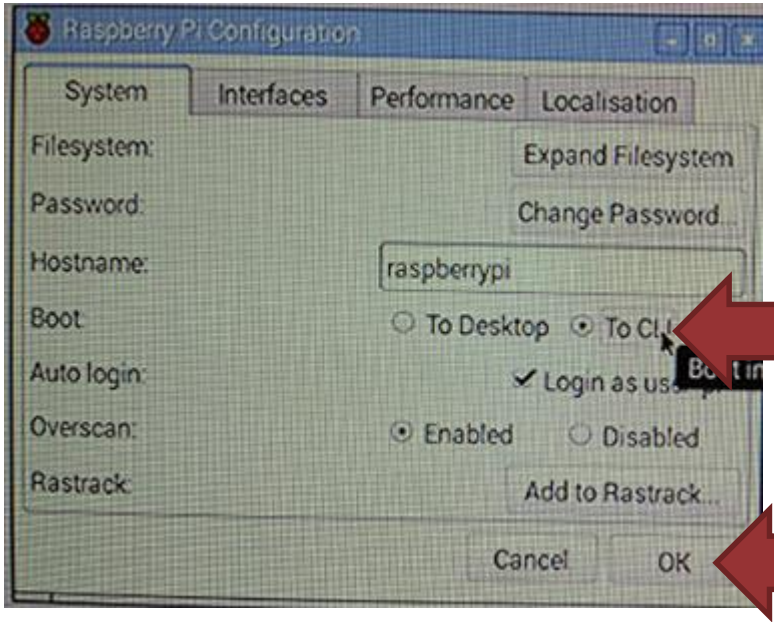
i.e., `ssh 192.168.1.75 – l pi`

Once logged in, try running the `blinker`, `button`, or `pwm` programs remotely.

username: pi, password: raspberry

Booting to Desktop / Console Configuration





FORMATTING SD CARD

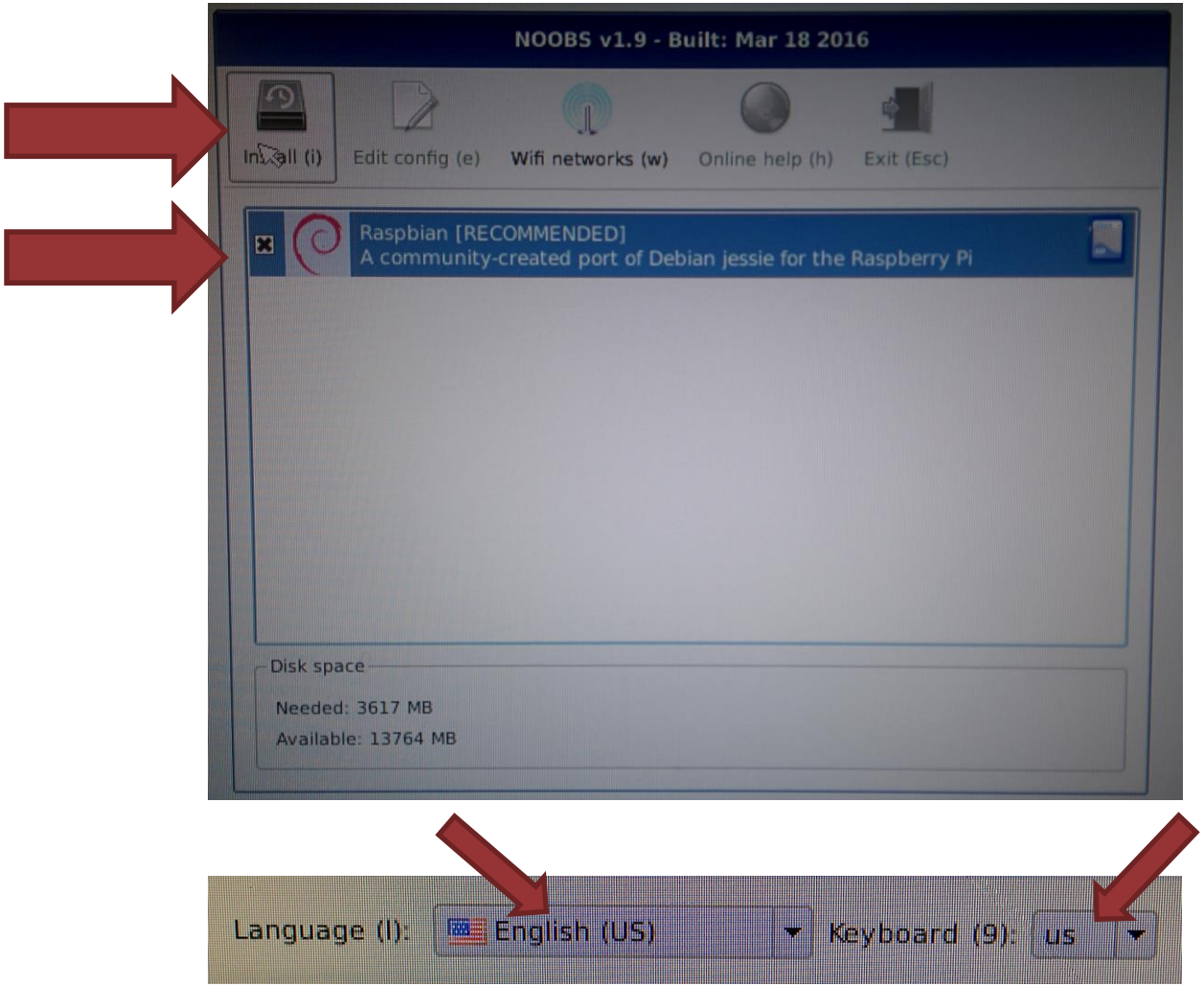
In the event you want to start fresh, following these steps.

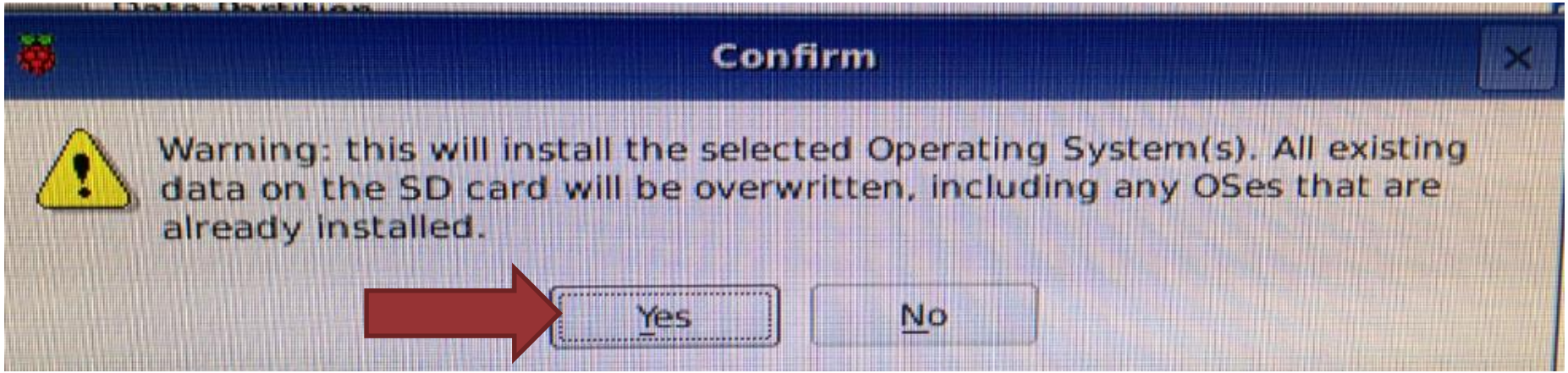
1. Download & Install SD Formatter from
   https://www.sdcard.org/downloads/formatter_4/

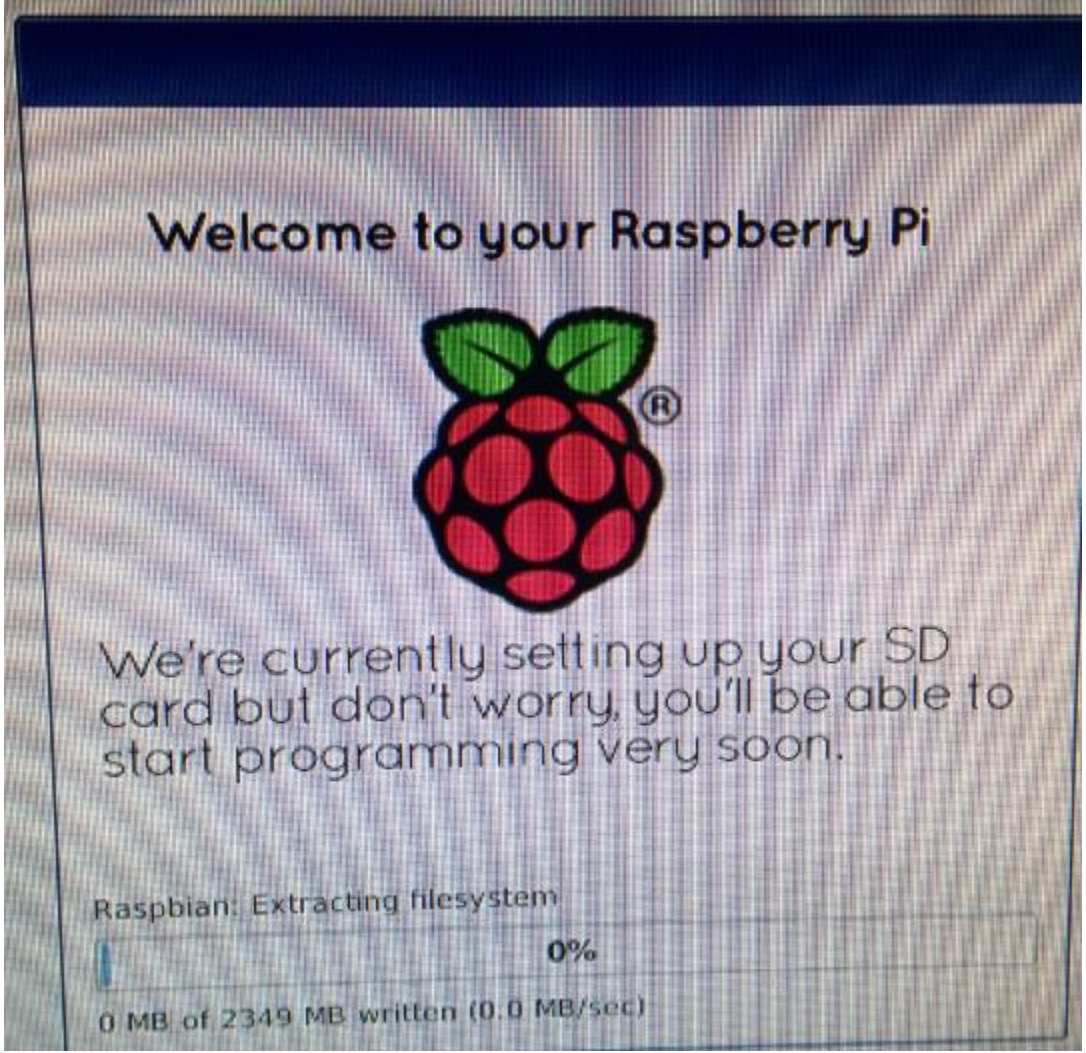
2. Format SD card with SD Formatter

3. Download NOOBS or your choice of distribution from
   https://www.raspberrypi.org/downloads

4. Extract contents of distribution zip file to SD card

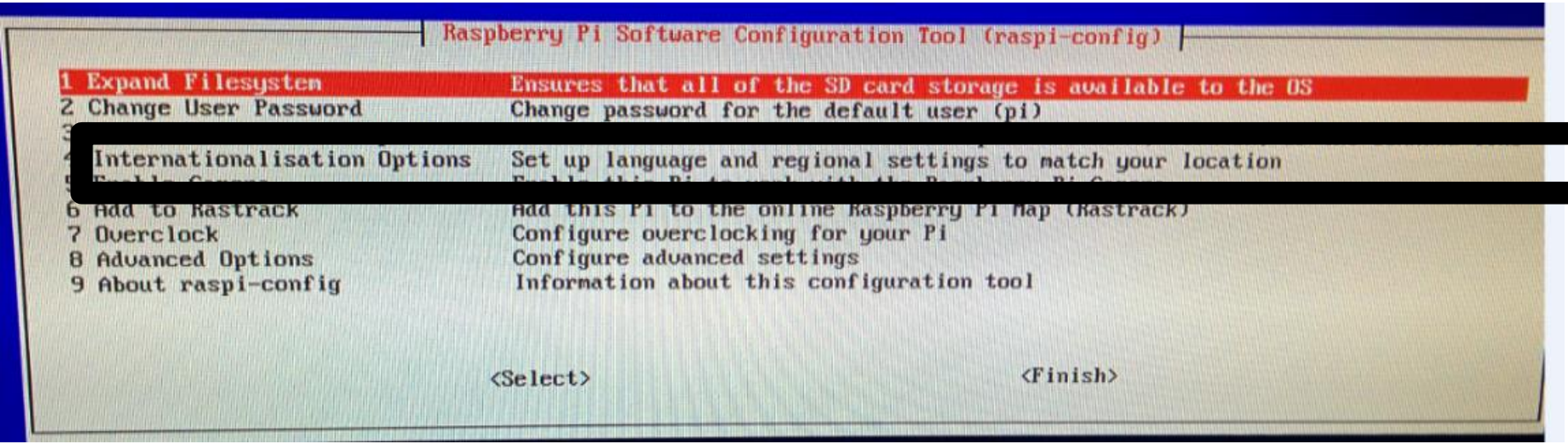

NOOBS v1.9 - Built: Mar 18 2016
Install (i)
Edit config (e)
Wifi networks (w)
Online help (h)
Exit (Esc)
Raspbian [RECOMMENDED]
A community-created port of Debian jessie for the Raspberry Pi
Disk space
Needed: 3617 MB
Available: 13764 MB
Language (l):
English (US)
Keyboard (9):
us

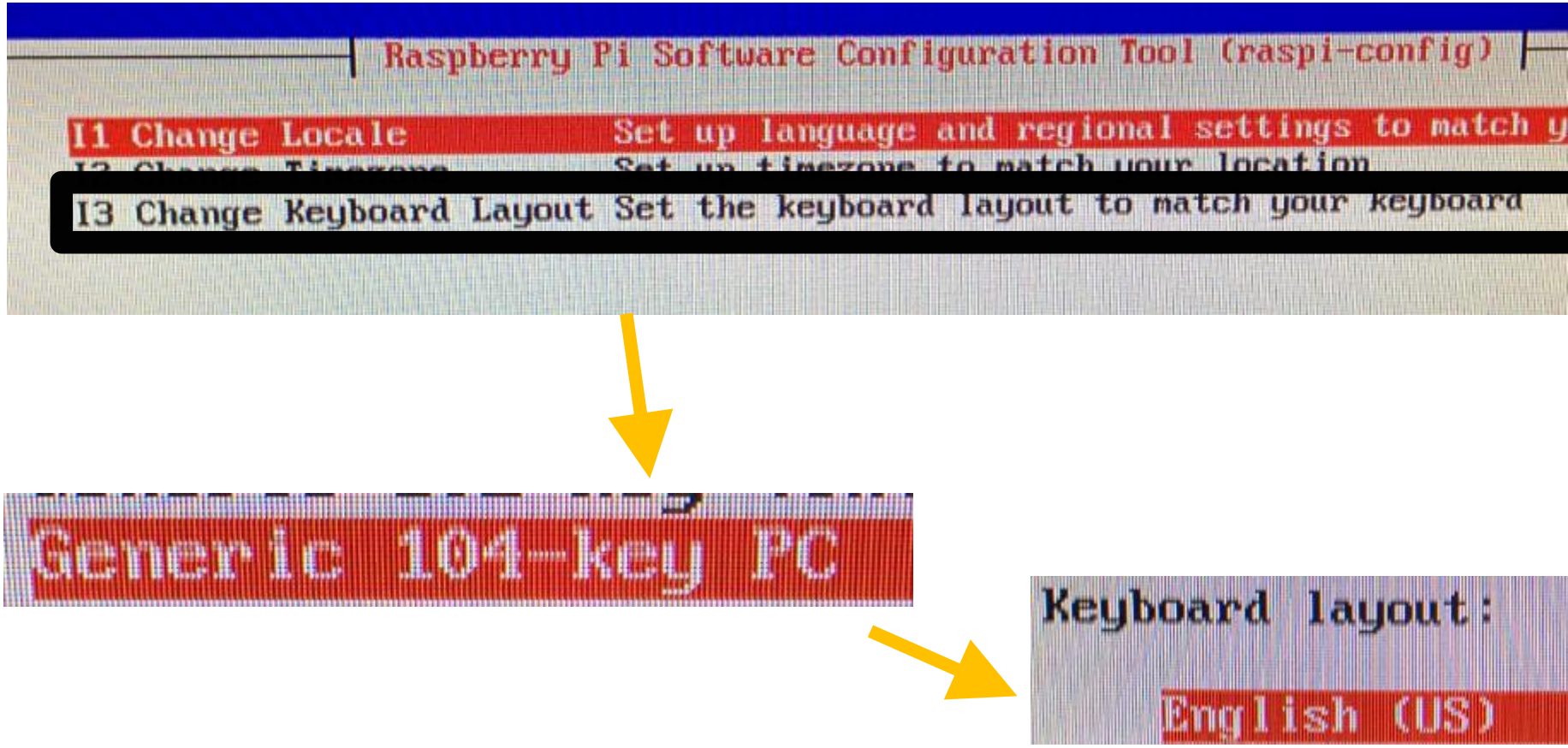

Confirm
Warning: this will install the selected Operating System(s). All existing data on the SD card will be overwritten, including any OSes that are already installed.
Yes
No

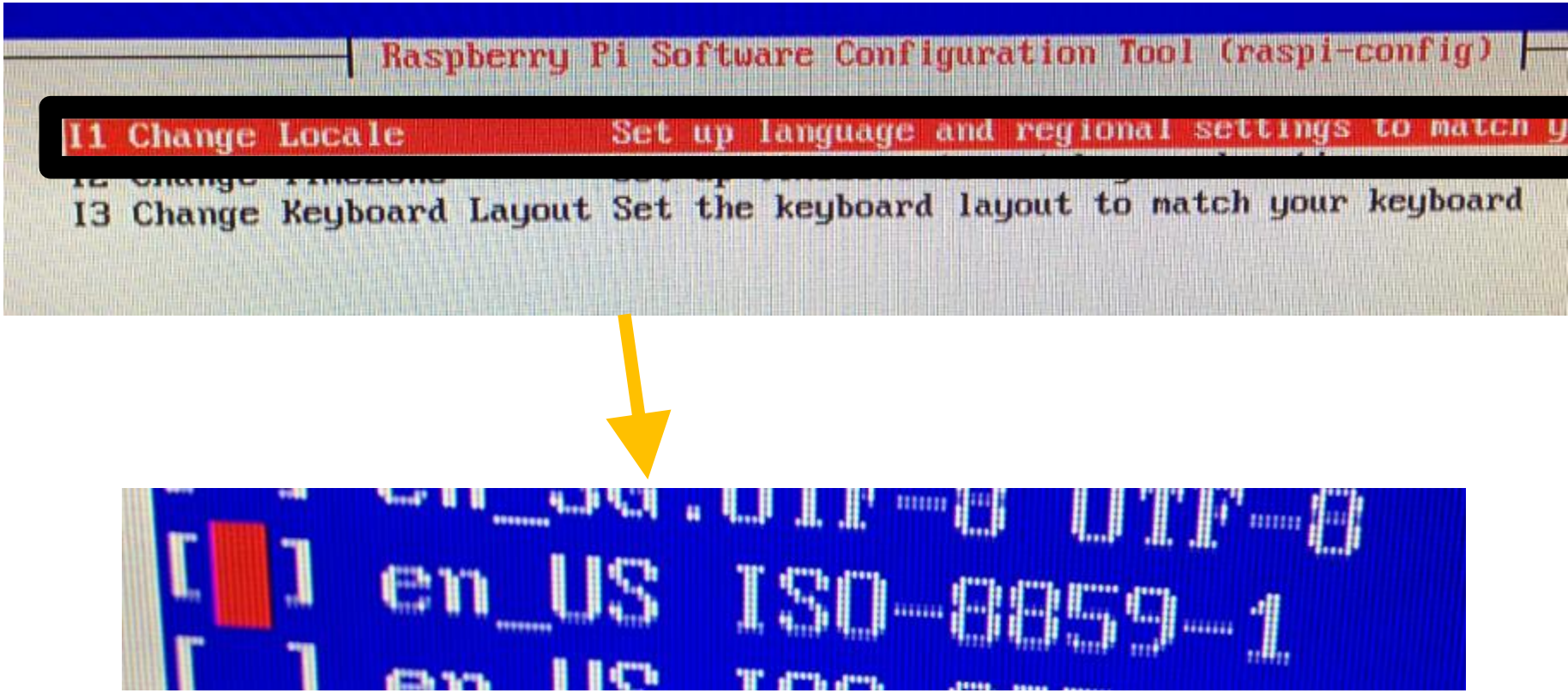

Welcome to your Raspberry Pi
We're currently setting up your SD card but don't worry, you'll be able to start programming very soon.
Raspbian: Extracting filesystem
0%
0 MB of 2349 MB written (0.0 MB/sec)

If you forgot to select English language & US Keyboard. Follow these steps. Otherwise proceed to press Finish.

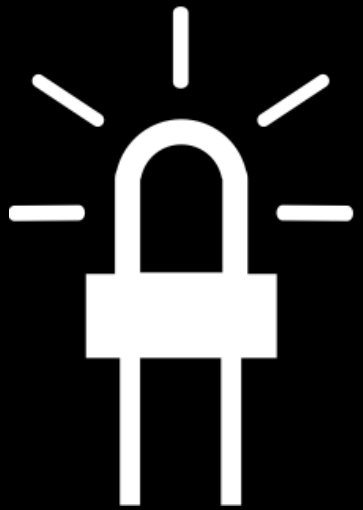
This can be accessed by typing *raspi-config* at the terminal




Raspberry Pi Software Configuration Tool (raspi-config)
I1 Change Locale Set up language and regional settings to match y
I3 Change Keyboard Layout Set the keyboard layout to match your keyboard
Generic 104-key PC
Keyboard layout:
English (US)


Raspberry Pi Software Configuration Tool (raspi-config)
I1 Change Locale
Set up language and regional settings to match y
I3 Change Keyboard Layout
Set the keyboard layout to match your keyboard
en_US ISO-8859-1

RASPBERRY PI QUESTIONS?



RASPBERRY PI

THE END

Thanks for Joining Us!