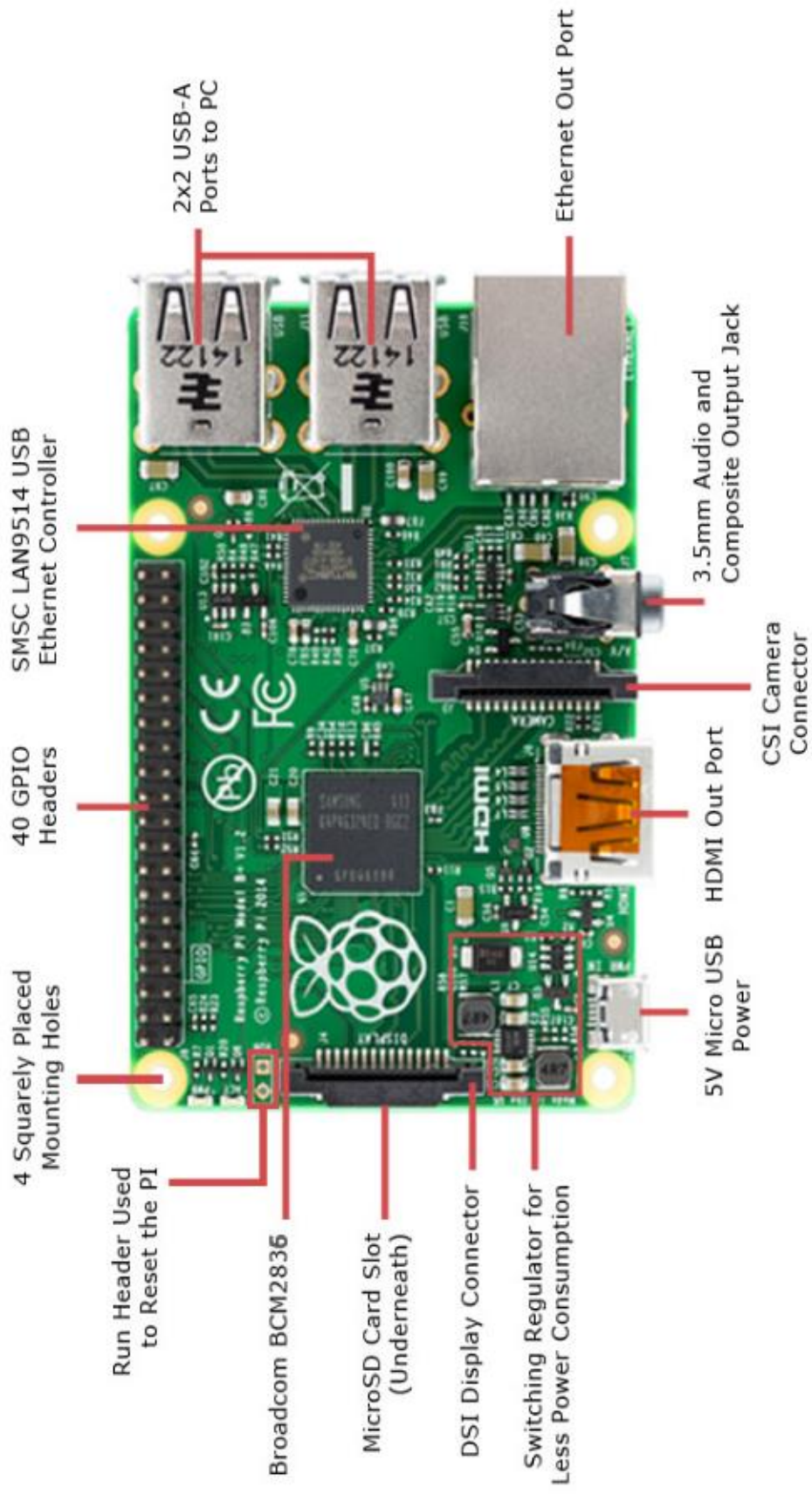
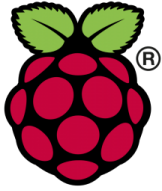



Raspberry Pi and Basic Linux

Author: Nick Lee



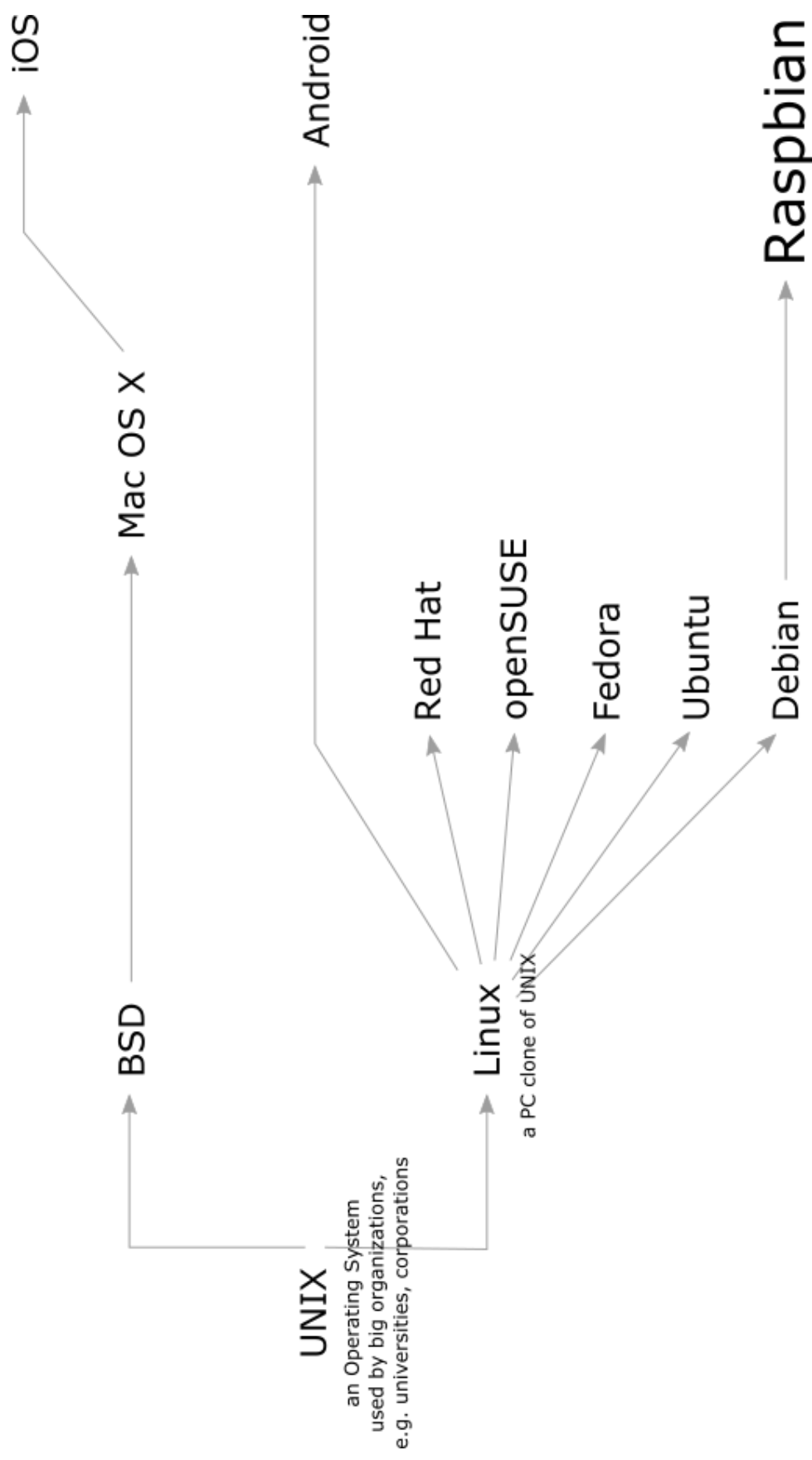
Raspberry Pi 3 Model B vs Arduino

		
	Micro-computer	Micro-controller
	Has OS Run your program along with many others	No OS Run your program ONLY
	Python, NodeJS, Ruby, Java, C, C++, etc.	C++
	Knowledge of Linux required	Program and debug on Windows
GPIO pins	Digital only	Analog and digital
Processor	BCM2837 (Broadcom)	ATmega (Atmel)
Speed	1.2 GHz, quad-core	8-16 MHz
RAM	1 GB	32-256 KB

From official documentations:

[Pi 3] can still be run from a 5V micro-USB power adapter ... we're recommending a 2.5A adapter if you want to connect power-hungry USB devices to the Raspberry Pi.

A Simplified UNIX Family Tree



Prepare an SD card

1. Download **Raspbian**, a Linux flavor commonly used for Raspberry Pi.
 - Go to the **Downloads** section on **raspberrypi.org**
 - Find **Raspbian**. Download its zip file.
 - Unzip it. The resulting file is called a **disk image**.
2. Download and install an image burner program.
 - On **Windows**, search for **Win32DiskImager**
 - On **Mac**, search for **Pi Filler**
 - Cross-platform: **Etcher**
3. With the program you have just installed, burn the Raspbian image onto an SD card.

To enable SSH on first startup, manually insert a file named "ssh" (without the quotes, contents don't matter) into the boot partition. For more details, refer to:

<https://www.raspberrypi.org/blog/a-security-update-for-raspbian-pixel/>

4. Insert the SD card to a Raspberry Pi.

Initial Configuration

```
$ sudo raspi-config
```

- **Change User Password:** Doesn't really matter, but good practice.

Notice that, as you type the password, the cursor NEVER moves. This is normal. Linux does not reveal the length of password as you type. If you type wrong, **[backspace]** all the way to start anew, or just **[enter]** to give yourself another chance.

If you do not change password, the default is:

User: **pi**

Password: **raspberry**

- **Network Options → Hostname:** Change the computer's name, so it is more easily recognized on the network.
- **Interfacing Options → SSH → Yes:** Enable SSH server, so you can login remotely.
- **Interfacing Options → VNC → Yes:** Enable VNC server, so you can use the graphical desktop remotely.
- **Finish and Reboot.**

After reboot, update the system:

```
$ sudo apt-get update
```

```
$ sudo apt-get upgrade
```

To calibrate touchscreen:

```
$ DISPLAY=:0.0 xinput_calibrator
```

VNC

On a PC within the same network:

1. Download RealVNC viewer
2. Connect to Pi's IP address

You may find the desktop resolution very small. To make it bigger, you may create a new desktop:

```
$ vncserver -geometry 1024x768
```

Take note of the **IP address:display number** printed out, e.g.
192.168.0.103:1

Connect to that <IP address:display number>.

For more tips on using Raspberry Pi:

<https://www.raspberrypi.org/documentation>

It is nearly impossible to use Raspberry Pi without knowing Linux. You have to be familiar with some basic commands.

ls

list

ls

What things are there?

ls -l

Give me more details.

ls -a

Show me the hidden stuff (those starting with a dot).

ls -l -a

Chain two options together to double the benefits.

ls -la

You may also chain them like this.

ls -tl

Sort by date.

ls *directory*

What is in that directory?

man ls

Explain this command.

ls --help

Another way to explain this command.

But the output overflows the page, what can I do?

cd

change directory

cd *directory*

Go to a directory.

cd ..

Go up a level.
Double-dot means the parent directory.

cd .

Stay in the same directory.
Single-dot means the current directory.

In practice, you never do this. It is here for educational purposes only.

cd ~

Go back to home directory.
The tilde (~) means home directory.

cd

Go back to home directory. Equivalent to **cd** ~.

pwd

print working directory

pwd

Where am I?

cp

copy

cp *oldfile newfile*

Make a copy of *oldfile*, and name it *newfile*.

cp -r *olddir newdir*

Copy an entire directory.

“-r” means “recursive”. That is, copy sub-directories, and their sub-directories, and their sub-directories, and so on.

Type **cp --help** or **man cp** to review the options.

mv

move or rename

mv *oldfile newfile*

oldfile now becomes *newfile*.

mv *file dir*

Move file to directory.

rm

remove

rm *filename*

Delete a file.

rm -r *directory*

Delete a directory.

“-r” means “recursive”. That is, delete sub-directories, and their sub-directories, and their sub-directories, and so on.

mkdir

make directory

mkdir *directory*

Create a new directory.

rmdir

remove directory

rmdir *directory*

You can only remove an empty directory. Delete everything inside first, or use **rm -r**.

nano

a text editor

nano

When you want to write something,
bring up a text editor.

nano *filename*

Edit an existing file.

Two shortcuts you need to know, for
now:

Ctrl-O to save

Ctrl-X to exit

more or less

show text content, or show output page-by-page

less *filename*

Display file content.

If content is longer than one page:

- Use ← ↑ ↓ → **[Page Up/Down]** to navigate
- Type **q** to quit.

ls --help | less

Display the preceding command's output in a page-by-page manner.

"|" is called a "pipe". We will see more of it later.

If you have trouble typing the "|":

1. Enter **sudo raspi-config**
2. Select **Internationalisation options**
3. Select **Change Keyboard Layout**, this may take a while.
4. Linux should have detected the keyboard for you. Press **[enter]**.
5. Then, it asks for Keyboard Layout. Default is English (UK). DO NOT trust that. You should select **Other**, then **English (US)**, then **English (US)** again.
6. Finally, it asks a few more questions. Press **[enter]** to skip all of them.
7. **Finish** and **reboot**.

sudo reboot, sudo halt -h

Reboot and shutdown

`sudo reboot`

“sudo” means “superuser do”. You cannot reboot the machine as an ordinary user. You must turn yourself into a *superuser* first.

`sudo halt -h`

To shutdown the machine, you also have to be a superuser.

Type **halt --help** or **man halt** to review the options.

DO NOT unplug the power until Raspberry Pi is completely shut down. The SD card may be corrupted if power is pulled prematurely.

Pay attention to the **blinking lights** on Raspberry Pi. Wait until only **one steady red light** is left and all other lights are dead. That’s when you can unplug the power safely.

How do you turn it on again? Just re-plug the power.

find

look for files with certain properties

```
find . -name *.wav
```

Any .wav files in current directory?

```
find python_games -name *.py
```

Any .py files in python_games directory?

```
find . -type d
```

Any directories in current directory?
“-type d” tells it to show directories only.

```
find directory -type f
```

Any files in that *directory*?
“-type f” tells it to show files only.

Type **man find** or **find --help** to learn more.

grep

look for lines with certain words

```
grep -lr import  
python_games
```

Which files in `python_games` contain the word “import”?

```
grep -Hr word  
directory_or_file
```

Try the command with the `-H` option. What are the differences?

You may look for any *word* in any *directory or file* with this command.

```
ps aux
```

It tells you what processes are running on the Raspberry Pi. The output is usually very long. What can you do?

```
ps aux | less
```

You may “pipe” it to a **less** command to read the information page-by-page.

```
ps aux | grep ssh
```

Sometimes, you want to look for a process to see if it is running, or how many of it are running.

For example, if you want to list all ssh-related processes, you may “pipe” the output to a **grep** command that looks for “ssh” and only shows those lines.

which

Which file is a command referring to?

which *command*

I have introduced many Linux commands so far. But what exactly is a command?

In Linux, a command is just a *file* that gets *executed*. For example, when you type **ls**, Linux *executes* a file called “ls”. When you type **find**, Linux *executes* a file called “find”.

But where are those files?

which ls

Use **which** to find out.

which find

Also try this.

ls and **find** live in different locations. How does Linux know where to look?

chmod

change mode

chmod +x *filename*

Turn *filename* into an executable.

After this, *filename* may be used as a command. Run it by typing:

./filename

The preceding “./” is to tell Linux to look for *filename* in current directory. If current directory is included in PATH, there is no need to type “./”.

chmod -x *filename*

Turn *filename* into an ordinary, non-executable file.

Other useful commands

`ps` - report a snapshot of current processes

`top` - report current processes in an interactive manner

`kill` - send a signal to a process. As the name implies, it is often used to kill a process.

`dd` - convert and copy a file, often used to create backups.

`df` - report disk space usage

`ifconfig` - network interface information

`wget` - download files from the internet

`tar` - archive and (de)compress files

`unzip` - decompress zip files

Run tasks in background

A lot of our programs have an infinite loop at the end. To kill them, we use **Ctrl-C**.

Another way is to put it into background using **Ctrl-Z** then **bg**. For example:

```
$ python3 run-forever.py  
[Ctrl-Z]  
$ bg
```

Ctrl-Z stops the process, but does not kill it. **bg** keeps the process running in the background.

This way, you get the prompt back and can keep working. The program's output will still be printed to screen.

You may run it in the background right from the beginning by appending a **&** to the command:

```
$ python3 run-forever.py &
```

How to find something to kill

Use **ps** or **pgrep** to find the PID (Process ID).

```
$ ps
$ ps aux
$ ps aux | grep ssh
$ ps aux | grep python
$ pgrep ssh
```

Use **kill** (with **-9** option if necessary) to kill.

```
$ kill 1234
$ kill -9 1234
```

Always remember that you are logged in as the user **pi**. To kill a process belonging to the user **root**, you have to **sudo**.

Think twice before you **sudo kill** something.

Discover IP addresses without having access to the router

Suppose you bring a Pi to a school, and have it connected to the school's network. Now, you want to SSH into the Pi, but you have a problem. Having no access to the school's router, how do you know the Pi's IP address?

One method is to use a **Port Scanner** software installed on a PC. Provided the PC is also connected to the school's network, the port scanner can inspect the network and report neighboring computers and their IP addresses. One of them should be your Pi.

For Windows, I recommend the **Advanced Port Scanner**.

On Mac or Linux, consider the **Nmap** utility.

SSH without knowing the IP address

Open **PuTTY**, try to connect to:

<hostname>.local

For example, if your Pi's hostname is **nickpi**:

nickpi.local

Can you connect? If yes, congratulations! Your PC already has **Bonjour** (or Zeroconf) set up.

Bonjour is a group of technologies to "automagically" discover systems and services on a local area network. Mac OS X already has Bonjour baked in. Windows does not have Bonjour support out of the box, but a few popular applications slip it in for their own needs, including **Skype**, Apple's **iTunes** and Adobe **Photoshop** CS3 or later. So you might not need to add anything at all!

Otherwise, it is most easily installed using **Bonjour Print Services for Windows**, available on Apple's website.

Nano is so ugly!

There is a lot of code to write. Instead of using such a primitive editor as nano, you would much prefer to work in a more user-friendly environment, probably on your PC. There is a few ways to do this.

1. **SFTP** - Do your work on PC, then transfer files to Pi.
2. **Samba** - enables a Linux machine to act as if it is a Windows file or print server. Once Samba is installed and configured on Pi, you can see it and access its files from another Windows machine.
Hint: Set masks in config file
create mask=0644
directory mask=0755
3. **Atom Editor + remote-sync** - The remote-sync plugin enables Atom editor to upload a file automatically every time you save a file locally, keeping files on PC and Pi in sync. **Sublime Text** (another editor) also has a similar plugin.
Hint: **Ctrl-Shift P**, type **remote sync**