Lecture 1

Introduction to Raspberry Pi

Learning outcomes

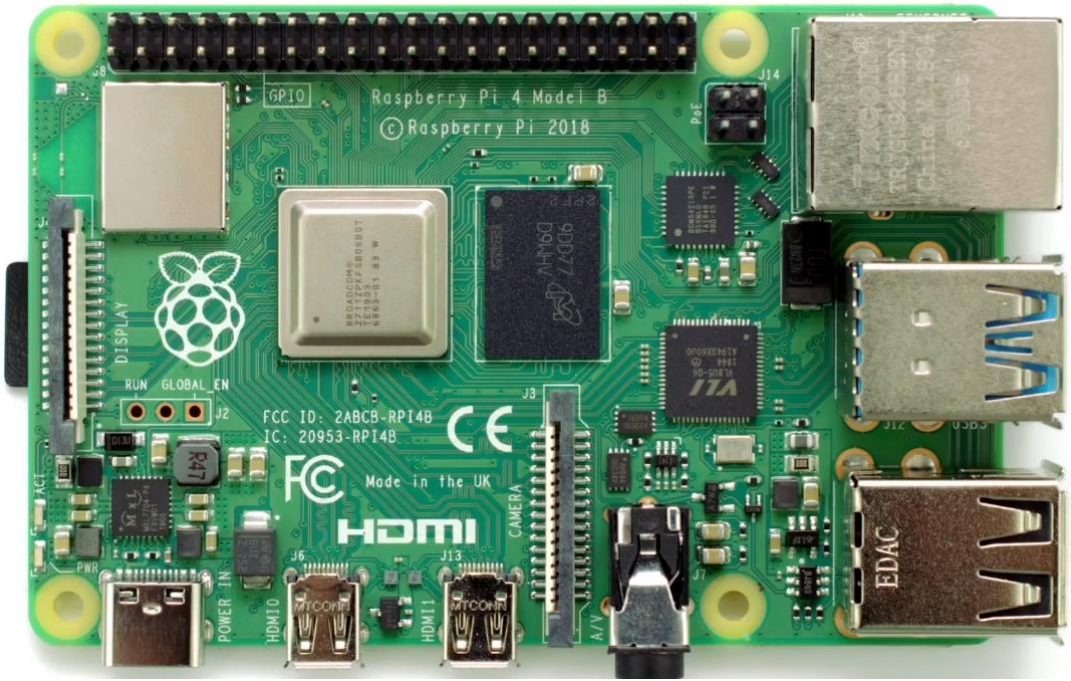
- Introduce Raspberry Pi
- How to Install Raspberry Pi OS
- Start programming with Python on Raspberry Pi

Introduce Raspberry Pi

What is Raspberry Pi?

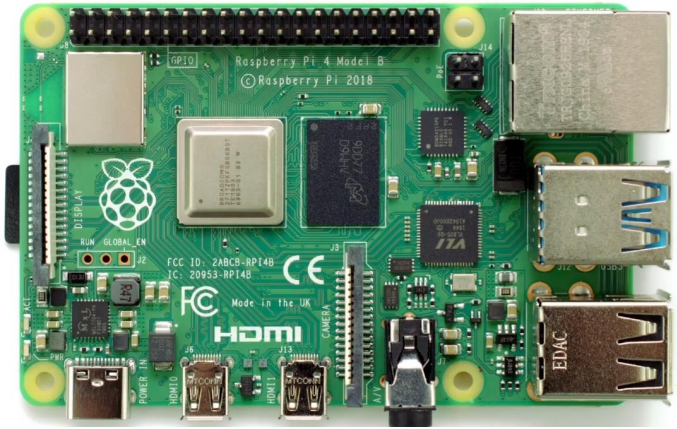
- The Raspberry Pi is a very small computer with the size even smaller than a smart phone.

- Don't underestimate it by its size – it is quite powerful and can run a lot of applications.



What can you do with Raspberry Pi?

- Run different Operating Systems
- Build Security and Alarm System
- Build Retro gaming console
- Home automation
- Server Hosting for Mobile Application
- Build robots
  - Drones
  - Arms
  - Cars
- Build Smart Mirror or Wall and much more….



Why Raspberry Pi

- Powerful and solid quality hardware
- Great official support
- Huge Community
  - Learn and get inspiration from other people's work
  - Ask questions and get help from veteran
  - Collaborate with other users on same projects and progress together.

Raspberry Pi background

- Created by the Raspberry Pi Foundation (2009)
  - The goal is to make computers and digital making available for everyone across the world.
- Very reasonable Price
  - Raspberry Pi Board starting from around $40
  - The whole set from around $100

Raspberry Pi background

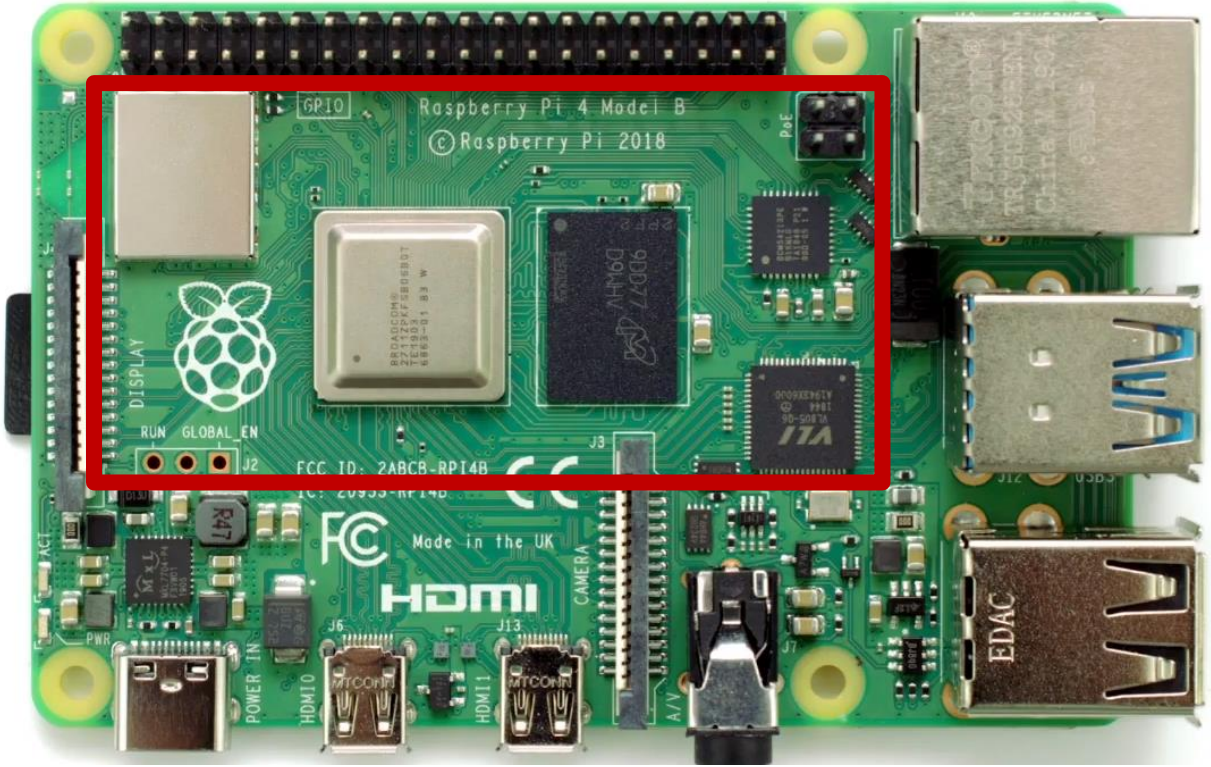
- Raspberry Pi 1 was released in 2012
  - The foundation has continued to develop new iteration to make it more powerful, more functionalities, easier to use and develop the community
- Raspberry Pi 2 was released in 2015
  - Much more computational power than version 1
- Raspberry Pi 3 was released in 2016
  - more powerful
  - WIFI feature directly implemented in the board

Raspberry Pi background

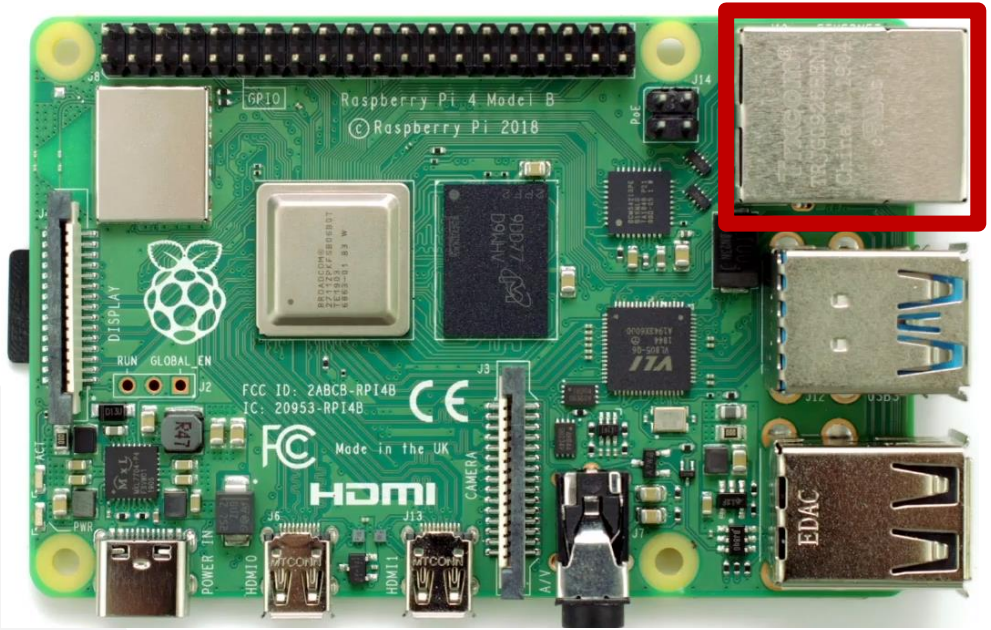
- Raspberry Pi 4 was released in 2019 (The one we are using)
  - A few changes in the external connectors to make it compatible with modern applications
  - amount of RAM is upgraded a lot (different options)
  - Pi 2 and Pi 3 had 1 GB RAM
  - Pi 4 has options of 2, 4 and 8 GB RAM
- Raspberry Pi 5 was released in 2023 which is the newest version.
  - It comes with a power button!

What are the components?

- All components to make the Raspberry Pi work
  - CPU
  - RAM
  - WIFI & Bluetooth chip
  - Ethernet and USB controller

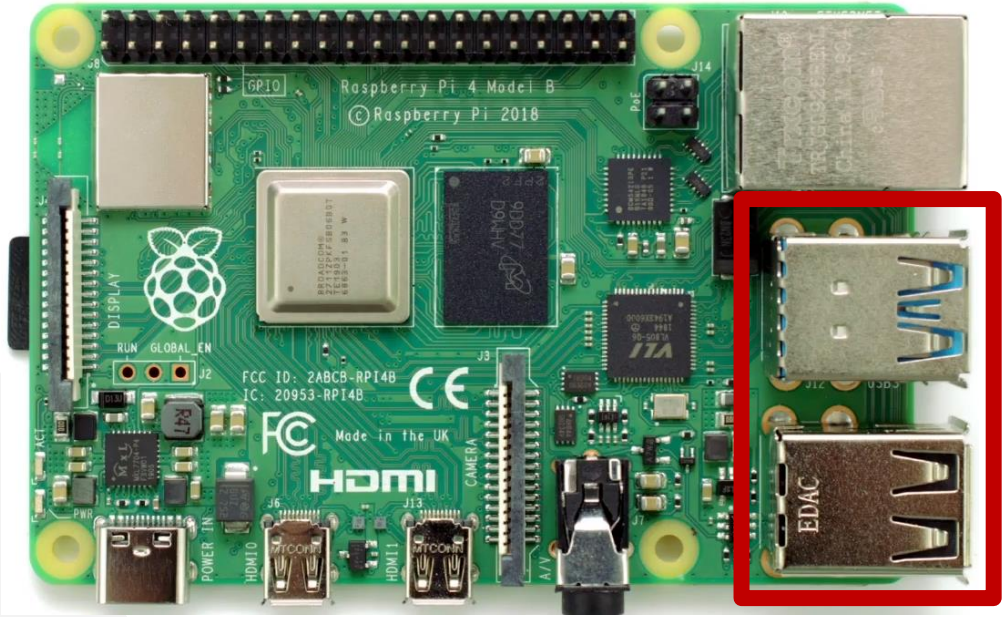


What are the components?



Gigabit
Ethernet Port

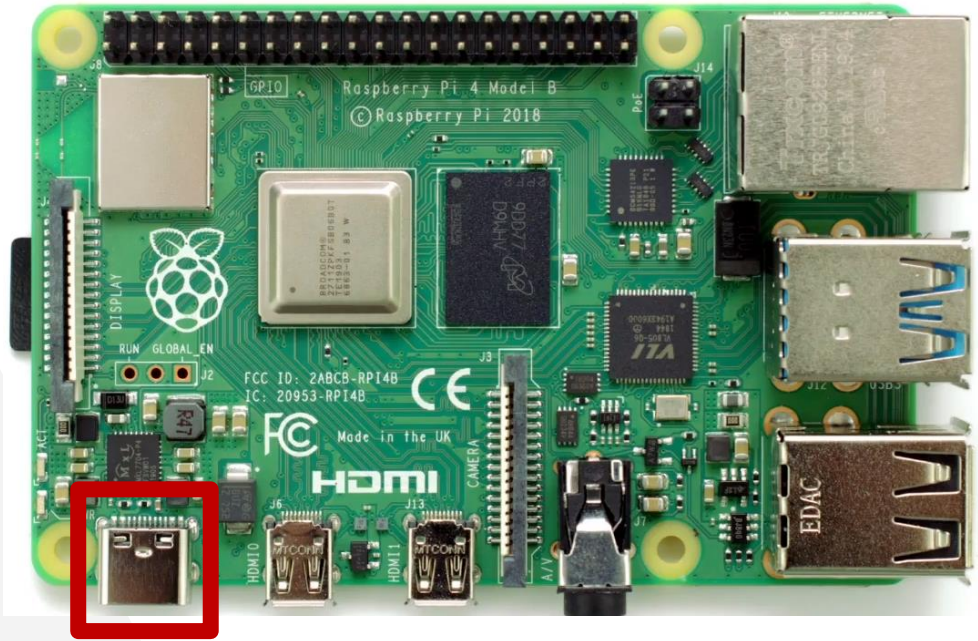
What are the components?



2 USB 3

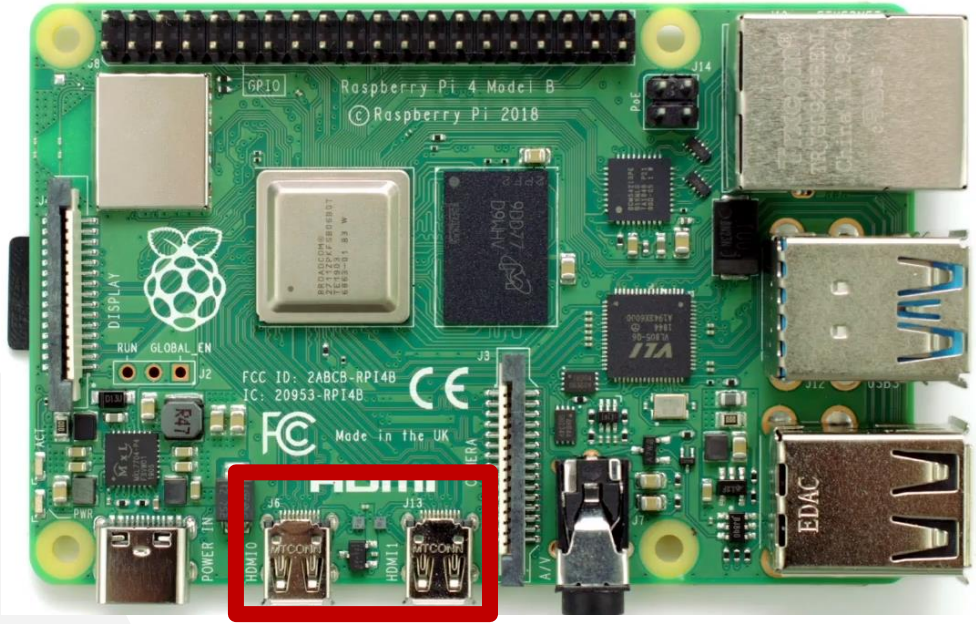
2 USB

What are the components?



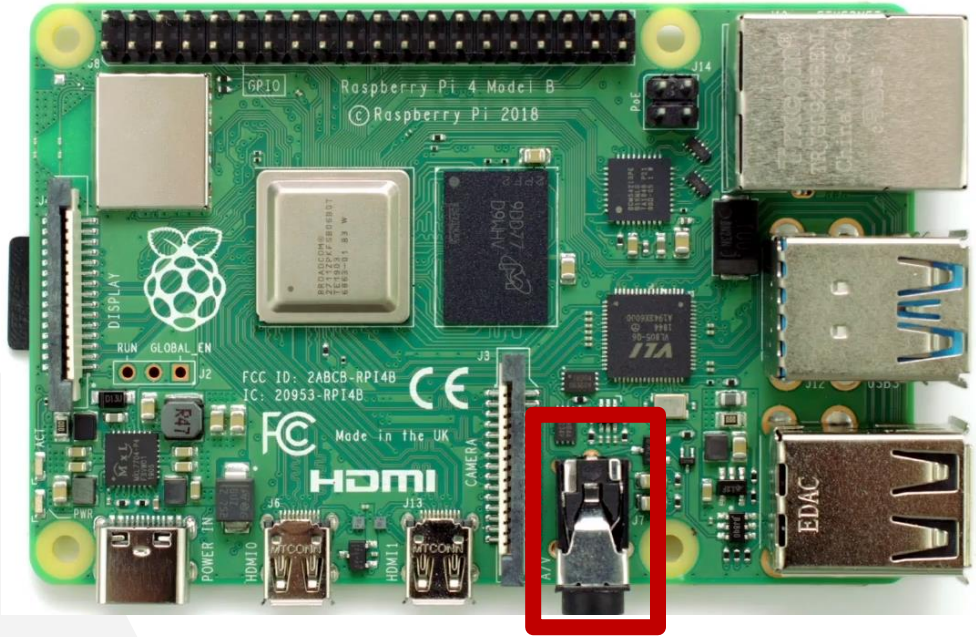
USB C port – Power supply

What are the components?



2 Micro HDMI Ports with 4K video support

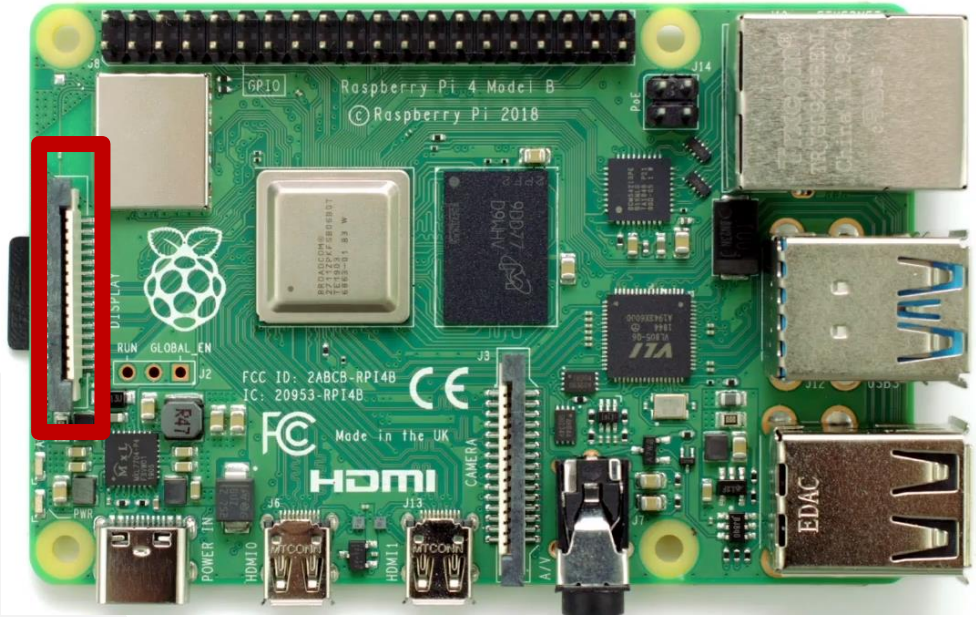
What are the components?



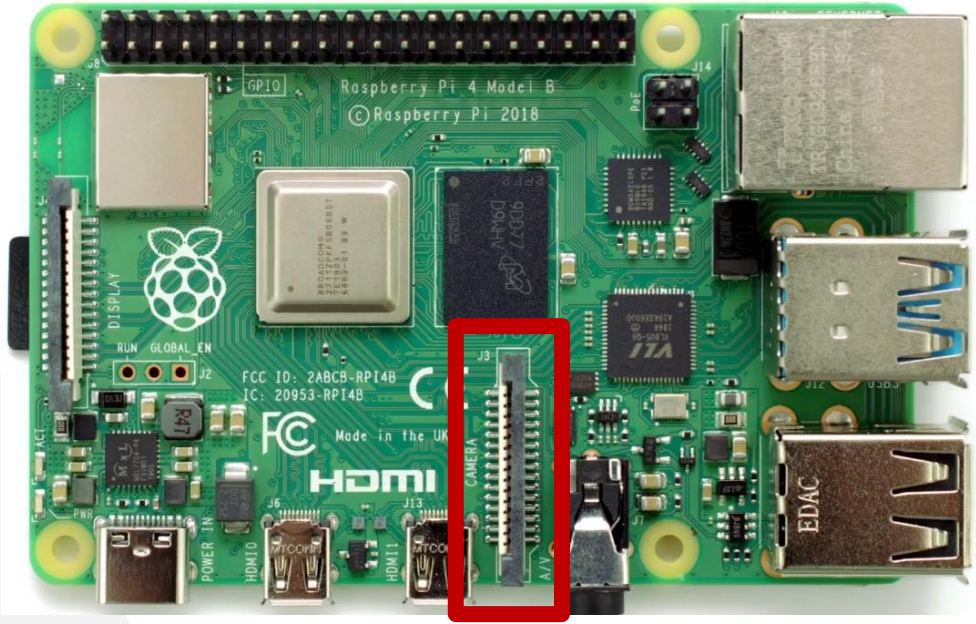
Audio Output Port

What are the components?

Display port for Raspberry Pi touch screen



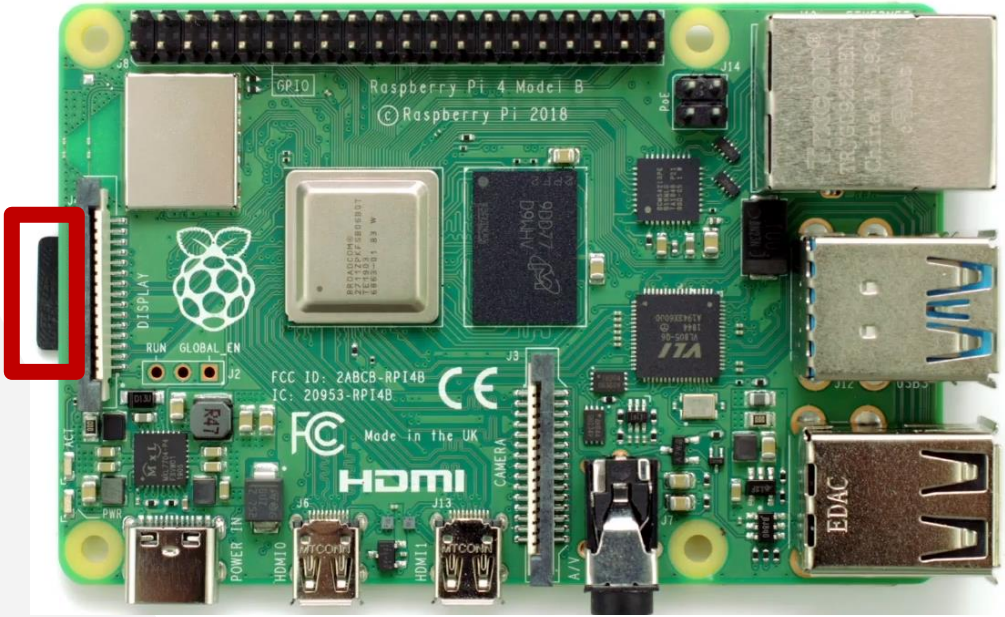
What are the components?



Port for Raspberry Pi camera module

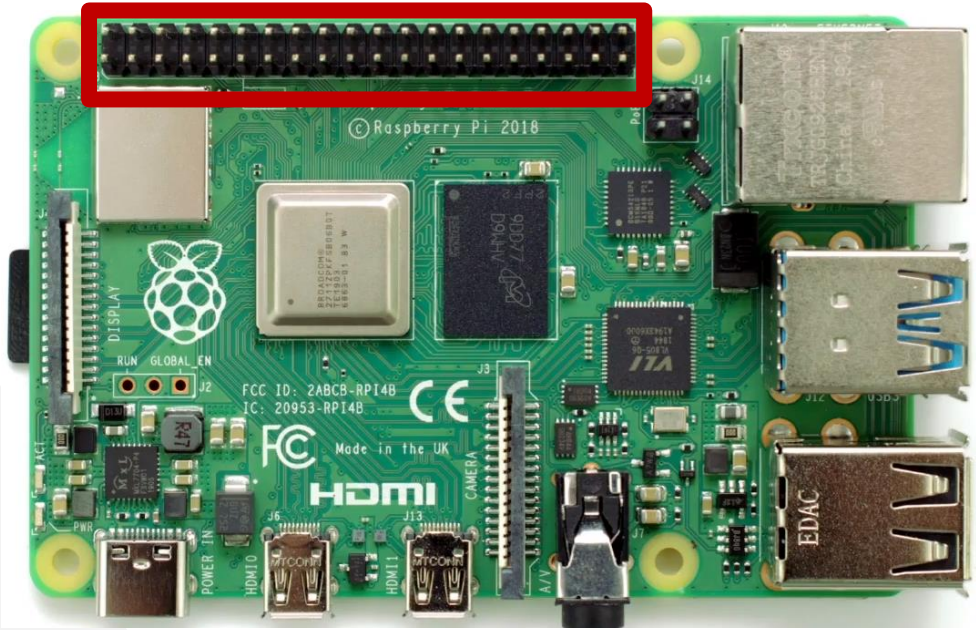
What are the components?

Micro SD card slot



What are the components?

40 GPIOs (General Purpose Input/Output) – We will use it a lot!



List of items

- Power supply
  - Even a smart phone charger (min 5V – 3A)
  - But not directly connect to computer's USB port
- Micro SD card
  - class 10 and at least 8 GB
- Breadboard
- Ribbon Cable
- Wires (female-female, female-male, male-male)
- Resistors (330Ohm, 10kOhm)

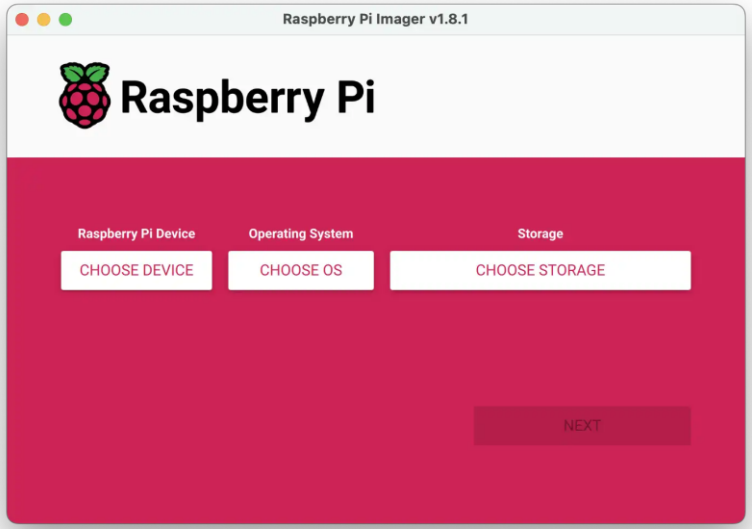
List of items (cont.)

- different color LEDs
- RGB LEDs
- Push buttons
- PIR sensor (HC-SR501)
- Raspberry PI camera module v2

How to Install Raspberry Pi OS

Install Raspberry Pi OS

- There are several ways of doing it
- The easiest way would be using the Raspberry Pi Imager
- Another option would be using NOOBS
  - Go to page 14 of the start guide document and follow the instructions



Install Raspberry Pi OS

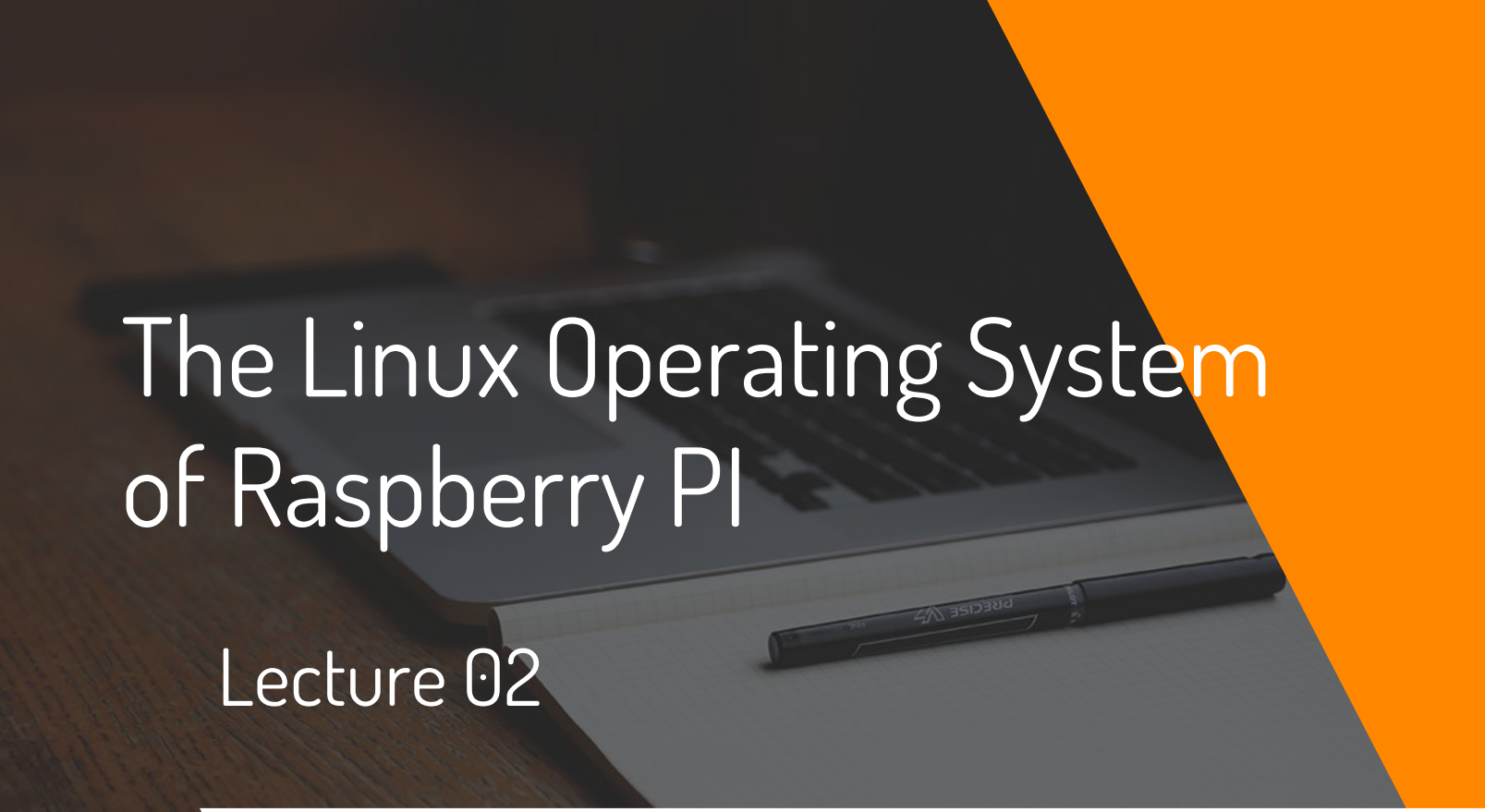
- After the installation, there are several startup configuration recommended
  - Connect to wifi
  - Update the OS
  - Enable auto-login and boot to desktop
  - SSH - allows you to remotely access the **command line** of the Raspberry Pi from another computer
  - VNC – allows RealVCN to get remote access to **raspberry pi desktop** from another computer
  - Adjust Appearance Settings to make it comfortable for you (optional)
  - Change username and password as you prefer

Power off Raspberry Pi

- Now you could play around the Raspberry Pi OS. Explorer it.
- One important thing to know before that is to correctly power off Raspberry Pi
  - Same as your computer, we do not just switch off the power
  - Make sure you using the menu or terminal to shut down the system and then remove the power

The Linux Operating System of Raspberry PI

Lecture 02



Instructor: Link

Course: CMPT 2200

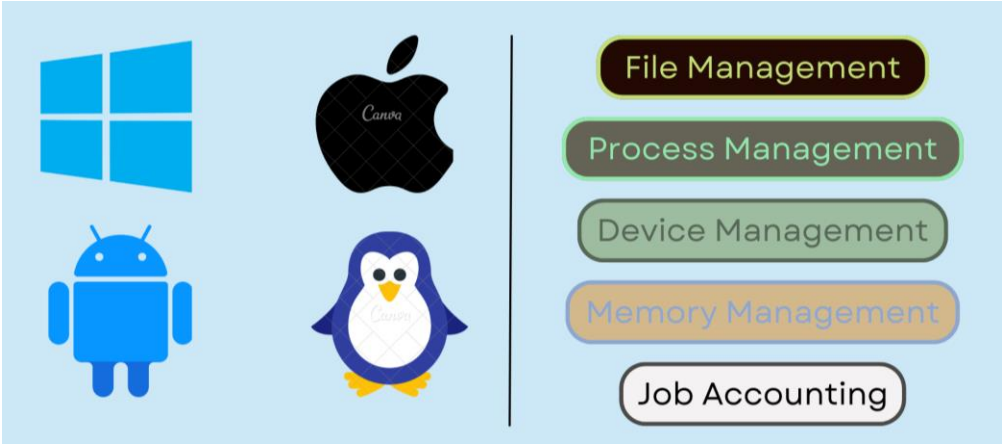
Learning outcomes

- Introduce Linux Operating System
- How to work with terminal of Linux
- Introduce Linux File System

Introduce Linux Operating System

What is an operating System?

- Resource manager
- Need management because of sharing
  - Multiple users
  - Multiple devices connected to the computer
  - Multiple programs running at the same time
  - Multiple files sitting on a disc

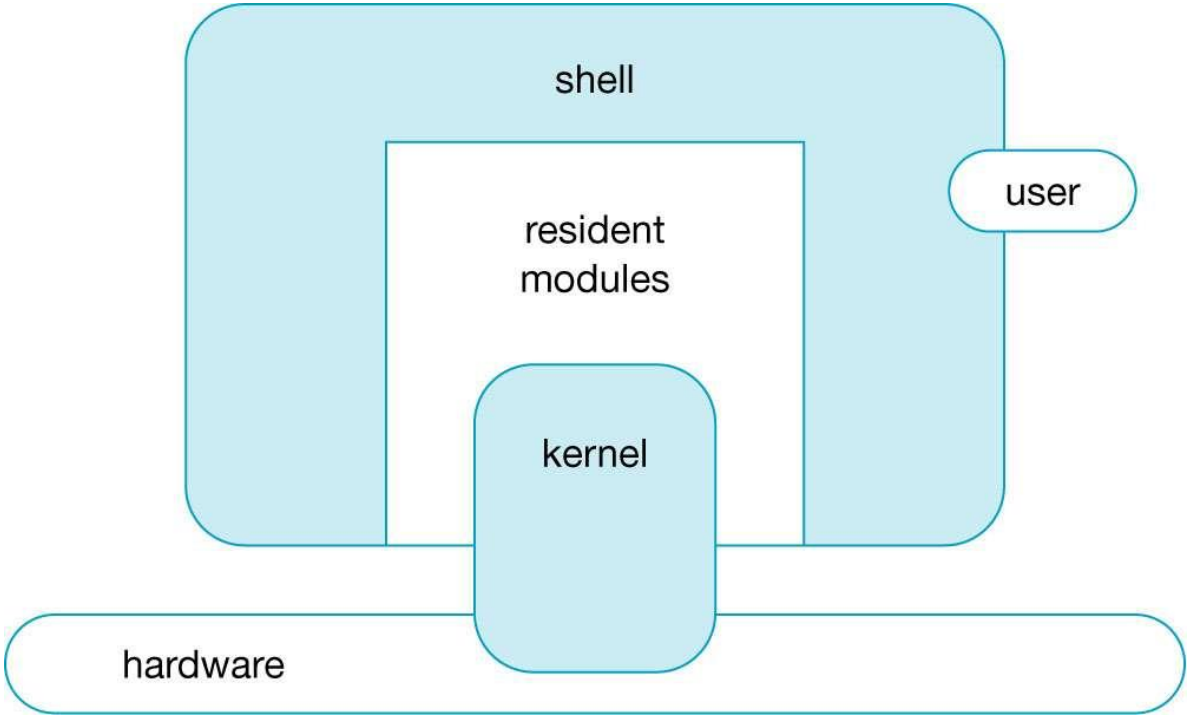


Unix Operating System Architecture

- Created from 1969 as a research project; Written in C language; Cross Platform
- Highly influence the modern operating system design and architecture building
- A layered-style model
  - **Kernal**, also called the base operating system, is the layer that manages all the hardware-dependent functions
  - **Resident Modules Layer**, provides service routines that perform user-requested service
  - **Utility Layer**, the user interface, commonly referred to as the shell

Unix Operating System Architecture

- **Kernal**
- **Resident Modules Layer**
- **Utility Layer**

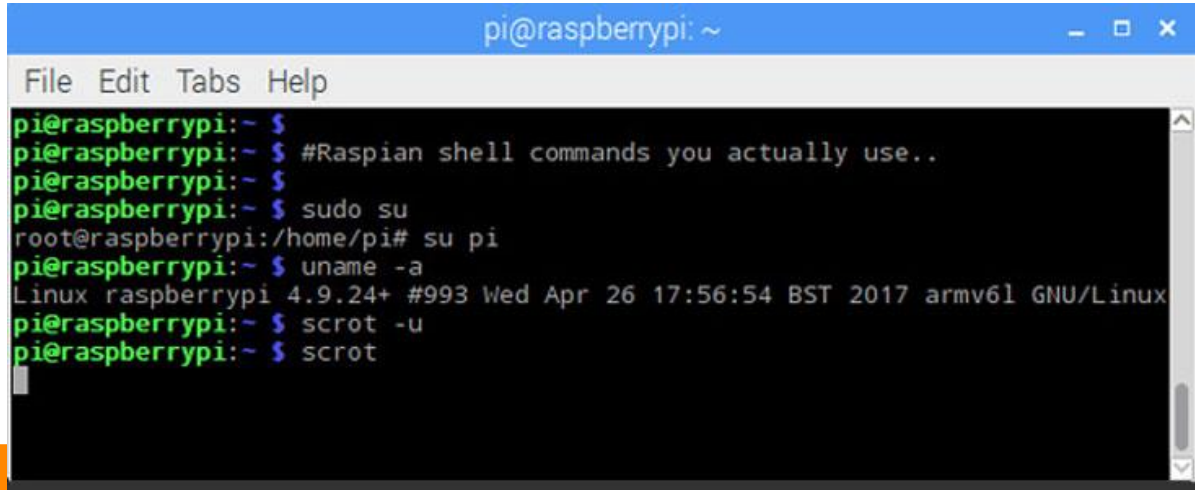


The Linux Operating System

- The Linux Kernal - Created in 1991 by Linus Torvalds
  - Purpose is creating a free, open source operating system
  - Highly influenced by the Unix Operating System
    - **Command line/ Terminal interface**
    - You could find a lot of utility legacies from Unix including
      - System administration
      - File Manipulation
      - Text editing

The Command Line: It's just another program

- GUI, or graphical User Interfaces - likes the ones you used in windows
- **Command line / Terminal interface**
  - Alternate Interface to the operating system
  - Very resource efficient
    - Effective for fast or slow connections

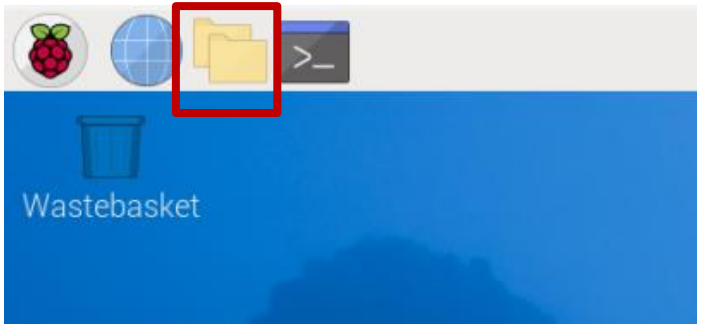


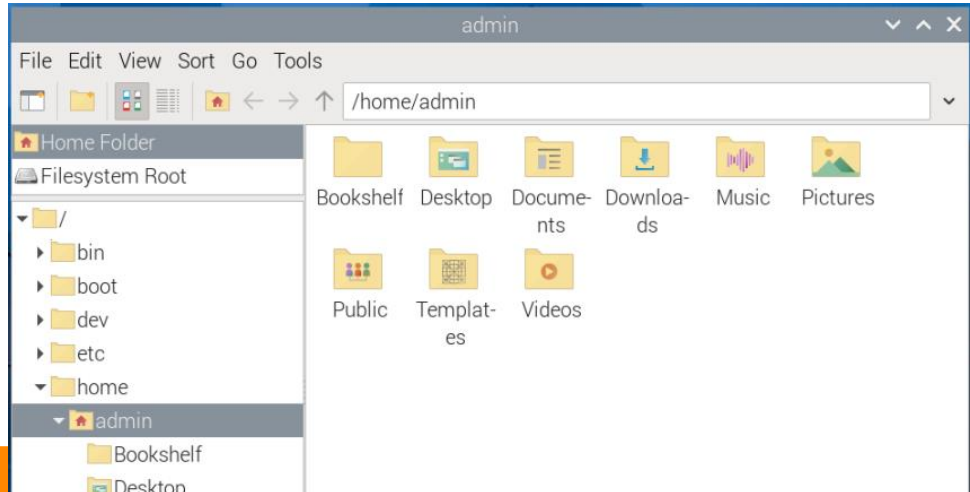
Raspbian, one Linux distribution for Raspberry Pi

- **Linux distributions** are publicly available version of Linux that share the Linux Kernel, but packaged with different modules and shell - a set of software programs, tools and configurations.
- There are many different Linux distributions, which are typically focused on different applications. For example,
  - High-end server systems: Red Hat Enterprise or OpenSUSE;
  - Personal Desktop systems: Ubuntu, Debian, Fedora, or Linux Mint.
- Our Raspberry Pi is using the distribution called **Raspbian**, which is a version of Debian.
  - Debian is also the foundation of Ubuntu, the most popular Linux Desktop Distribution, so you might find it familiar

GUI of Raspbian – File Manager

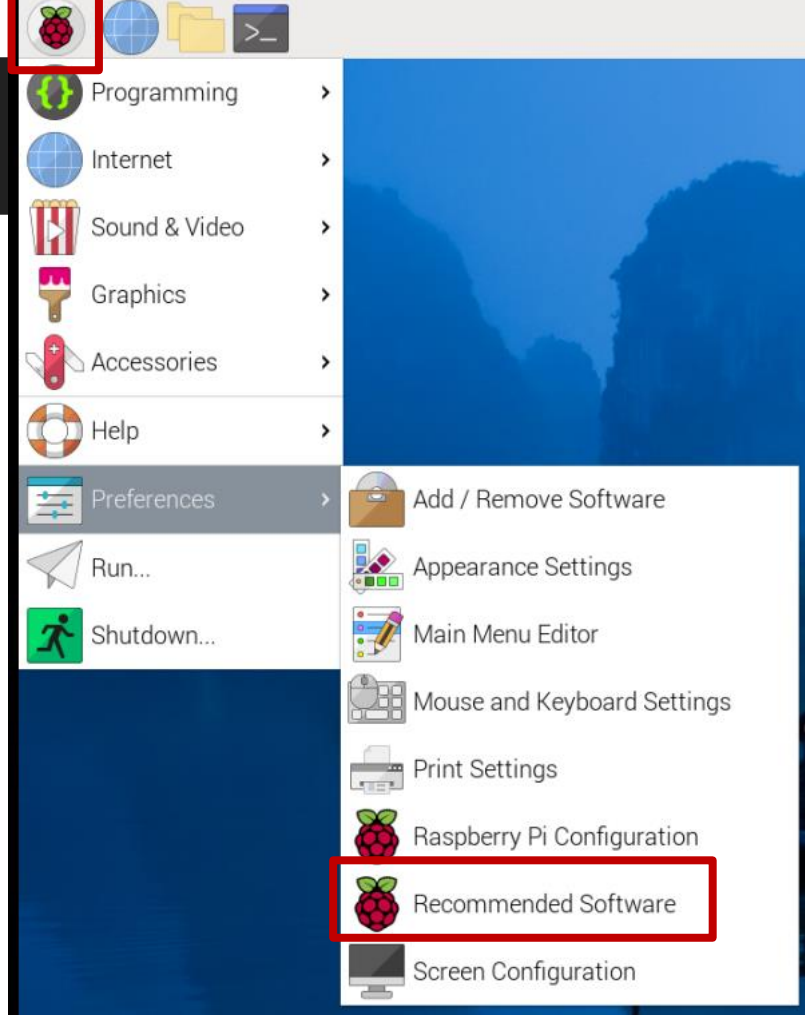
- The File Manager – Open by clicking the folder icon
  - It works very similar to Windows
  - You could create/move/delete/copy/paste file or folder, move file/folder
  - Left side of the File Manager shows the folder structure
  - Right side shows the files/subfolders in the current folder
  - Above the files shows the path of the current folder





GUI of Raspbian – Start Menu

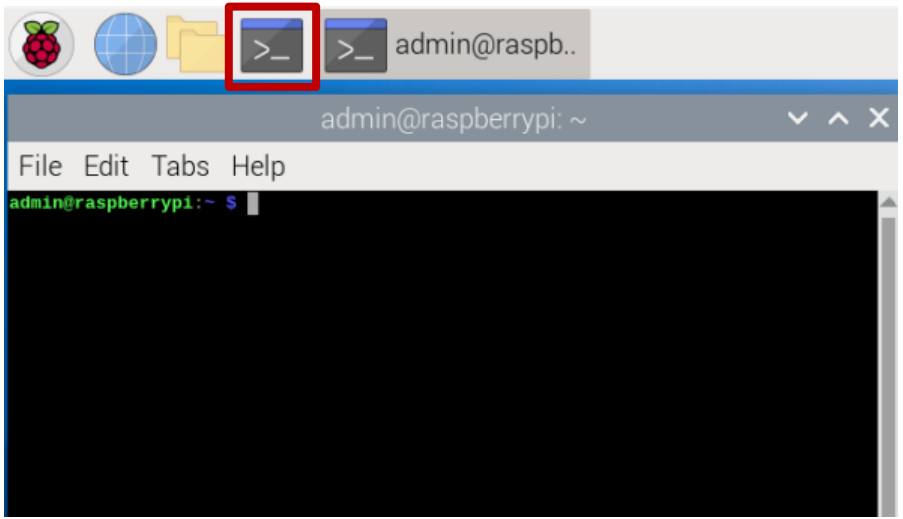
- The Start Menu – Open by clicking the Raspi icon
  - A portal to a lot of settings and applications
  - You could install extra software (recommended for Raspberry Pi) by choosing the **Recommended Software**
  - Coding by Choosing **Programming**
  - Open browser by choosing **Internet**



How to work with terminal of Linux

Starting a Terminal Session

- As we mentioned, Command Line Interface (Terminal) is a very important part of Linux distribution system. We could open it by either:
  - At the top of the Raspberry Pi desktop, select the Terminal icon.
  - Select from Start Menu -> Accessories -> Terminal
  - Press `Ctrl+Alt+T`



What you need to know before using Terminal

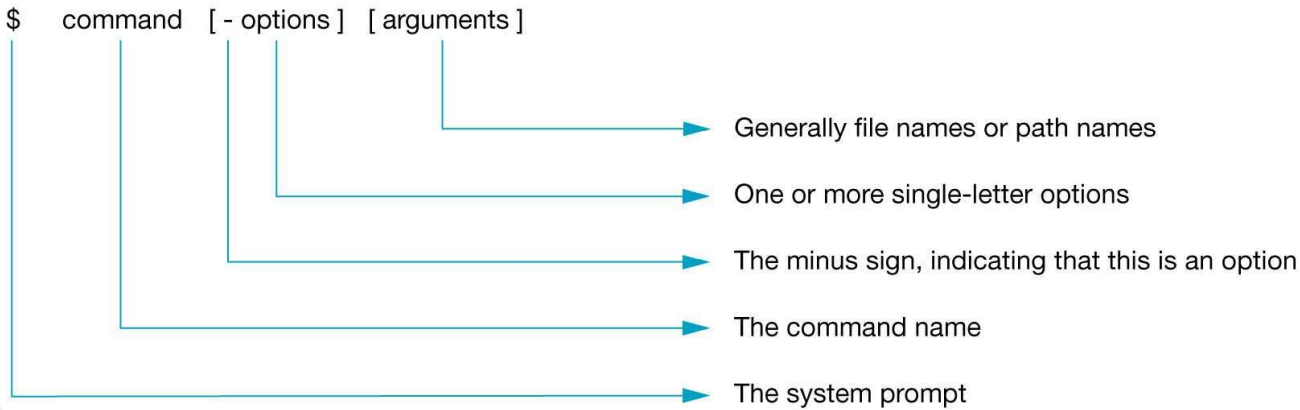
- We can almost do anything in Linux Distribution System with Terminal
- When the Terminal starts, it is set to your home directory (/home/admin). – The same directory when you open the File Manager
- You can open as many Terminal sessions as you want. It is often useful to have a couple of sessions open in different directories so that you don't need to constantly switch directories using `cd` command
- When using the Terminal, everything is case sensitive. That is, if you are using a command, you must use the correct case when you're typing.

Actions in this session

1. Command Line Format
2. Manage the user and password
3. A few simple commands
4. Correct typing mistakes

1 Basic Command Line Structure

- Each command line consists of three fields:
  - Command name
  - Options
  - Arguments



2.0.1 Run as administrator: The `sudo` Command

- By prefixing any command with `sudo`, it will run the command with **root privileges**
- You could consider it equivalent to the "**run as administrator**" in windows
- If you encounter a situation when Linux refuse to run your command, add `sudo` prefix would solve the issue most of the time

```
admin@raspberrypi:~ $ apt-get install
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?
admin@raspberrypi:~ $ sudo apt-get install
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
admin@raspberrypi:~ $
```

2.0.2 Escalate privileges to the root level: `sudo su`

- If you don't want to type sudo prefix every time, another option would be logging yourself into root environment
- In such case, running **any** command would be assumed with root privileges

```
admin@raspberrypi:~ $ sudo su
root@raspberrypi:/home/admin# apt-get install
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@raspberrypi:/home/admin#
```

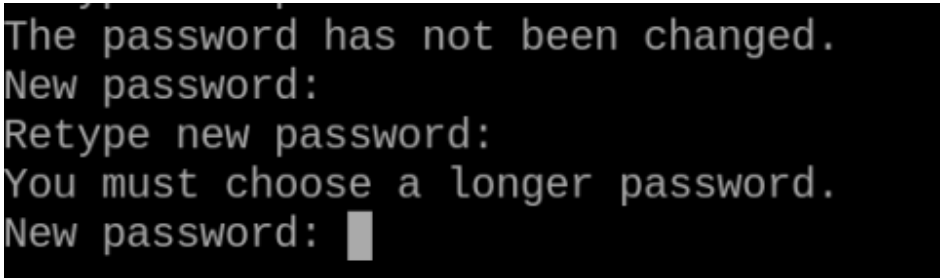
2.1 Changing Your Password: The `passwd` Command

- The passwd command changes your current password. If you do not have a password, it creates one.
- Type `passwd` and press [`Return`]
- Enter your current password and press [`Return`]
- Enter your new password
- Retype the new password

```
admin@raspberrypi:~ $ passwd
Changing password for admin.
Current password:
New password:
Retype new password:
```

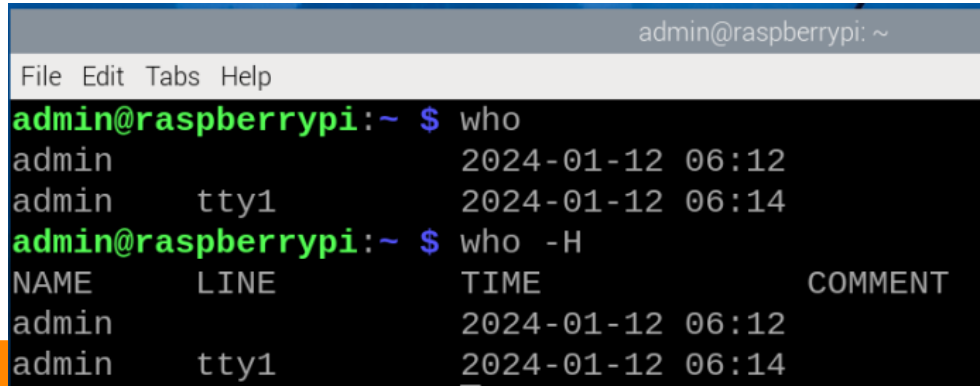
2.2 Password Format

- Your password must comply with the following criteria:
  - The new password must differ from the old one by at least three characters
  - The password must be at least six characters long and must contain at least two characters and one number
  - The password must differ from your User ID
- If Linux detects anything wrong with your password, it displays an error message and shows the New password: prompt again.



2.3.1 Check the current user: The `who` Command

- The `who` command lists the login names, terminal lines, and login times of the users who are currently logged on the system
- The `who` command is used to check the level of activity in the system or to find out whether a particular person is on the system
  - The first column shows the login name of the user
  - The second column identifies the terminal being used
  - The tty number gives you some indication about the location of the terminal
  - The third and fourth columns show the date and time that each user logged in



2.3.2 who Options

- Table lists some of the who command options
  - Linux also provides some alternative and new options
  - Under different Linux distributions, some of these options are not available, or some of the options outputs are slightly different
  - Linux new and alternative options are preceded by two minus signs (--)

Option	New & Alternative	Operation
`-q`	`--count`	The quick who; just displays the name and number of users
`-H`	`--heading`	Displays heading above each column
`-b`		Displays the time and date of the last reboot
	`--help`	Displays a usage message

2.3.3 `whoami` Command

- Type `whoami`
- Linux displays who the system thinks you are

```
admin@raspberrypi:~ $ whoami
admin
```

3.1 Date and Time Display: The `date` Command

- The `date` command displays the current date and time on the screen
- The date and time are set by the system administrator and users cannot change them

```
admin@raspberrypi:~ $ date
Fri 12 Jan 08:50:26 GMT 2024
```

3.2.1 GETTING HELP

- The `help` command brings up a computer-aided instruction program that is arranged in a series of courses and lessons.

```
admin@raspberrypi:~ $ help
GNU bash, version 5.2.15(1)-release (aarch64-unknown-linux-gnu)
These shell commands are defined internally.  Type `help' to see this li
st.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                            history [-c] [-d offset] [n] or >
 (( expression ))                        if COMMANDS; then COMMANDS; [ el>
 . filename [arguments]                  jobs [-lnprs] [jobspec ...] or j>
```

3.2.2 Using the help Command ...

- Use `help` command together with other commands will provide us details of how the command works. For example `help echo` or `help history`

```
admin@raspberrypi:~ $ help echo
echo: echo [-neE] [arg ...]
    Write arguments to the standard output.

    Display the ARGs, separated by a single space character and followed
 by a
    newline, on the standard output.

    Options:
      -n        do not append a newline
      -e        enable interpretation of the following backslash escapes
      -E        explicitly suppress interpretation of backslash escapes

    `echo' interprets the following backslash-escaped characters:
      \a        alert (bell)
      \b        backspace
      \c        suppress further output
```

3.2.2 Using the help Command ...

- Use `help` command together with other commands will provide us details of how the command works. For example `help echo` or `help history`

```
admin@raspberrypi:~ $ help history
history: history [-c] [-d offset] [n] or history -anrw [filename] or his
tory -ps arg [arg...]
    Display or manipulate the history list.

    Display the history list with line numbers, prefixing each modified
    entry with a `*'.  An argument of N lists only the last N entries.

    Options:
      -c        clear the history list by deleting all of the entries
      -d offset delete the history entry at position OFFSET. Negative
                offsets count back from the end of the history list

      -a        append history lines from this session to the history fi
le
```

3.2.3 Getting More Info: Manual

- You can find a detailed description of the Raspbian OS system online called Raspberry Pi Documentation.
  - It is more like a user guide than a reference manual
- We could also use the Electronic Manual: The `man` Command
  - The `man` (manual) command shows pages from the online system documentation.
  - Type `man` followed by the name of the command
  - You can use `man` to get the details about the commands
  - For example, type: `man echo [Return]`

3.4 Correct typing mistakes

The shell program interprets the command line after you press the `[Return]` key.

- **Erasing Characters**
  - Use the `[Backspace]` key for erasing characters or use `[Ctrl-h]` once for each character you intend to erase
- **Erasing an Entire Line**
  - You can erase an entire line any time before pressing `[Return]`
  - `[Ctrl-u]` removes the entire command line and the cursor moves to a blank line
  - The character that erases the entire line is called the kill character
- Terminating Program Execution

3.4 Correct typing mistakes

The shell program interprets the command line after you press the `[Return]` key.

- Erasing Characters
- Erasing an Entire Line
- **Terminating Program Execution**
  - The character that terminates your running program is called the interrupt character
  - On most systems, `[Del]` or `[Ctrl-c]` is assigned as the interrupt character
  - The interrupt character stops the running program and causes the shell prompt `$` to be displayed

3.4 Correct typing mistakes

- If you mistype a command name and end the command line by pressing `[Return]`, Linux displays a generic error message.
- It responds with the same error message if you type a command that is not installed on your system.
- For example, the `date` command is mistyped

```
admin@raspberrypi:~ $ daye
bash: daye: command not found
admin@raspberrypi:~ $
```

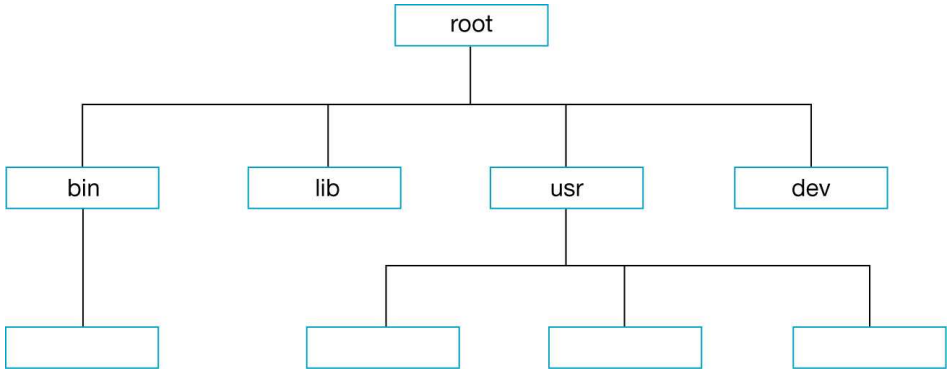
Introduce Linux File System

DISK ORGANIZATION

- Linux allows you to divide your hard disk into many units (called directories), and subunits (called subdirectories), thereby nesting directories within directories.
- Linux provides commands to create, organize, and keep track of directories and files on the disk.
- It has three categories of files:
  - **Regular Files**: contain sequences of bytes that could be programming code, data, text, and so on.
  - **Directory Files**: the directory file is a file that contains information (like the file name) about other files. It consists of a number of such records in a special format defined by your operating system.
  - **Special Files(device files)**: contain specific information corresponding to peripheral devices such as printers, disks, and so on.

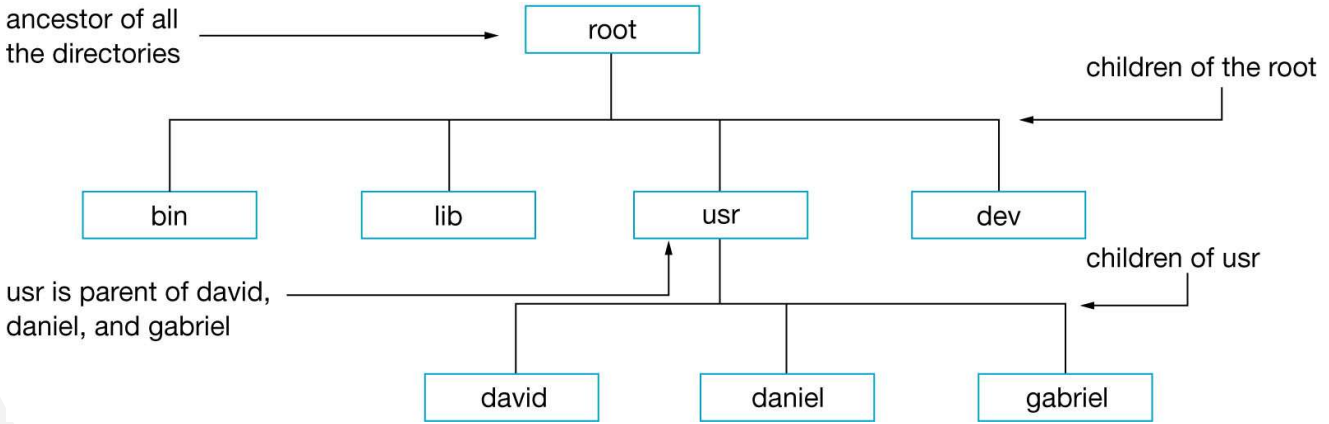
ALL ABOUT DIRECTORIES

- Directories are an essential feature of the Linux file system
- The directory system provides the structure for organizing files on a disk
- In UNIX, the directory structure is organized in levels and is known as a hierarchical structure
- The highest-level directory is called the root and all other directories branch directly or indirectly from it



ALL ABOUT DIRECTORIES

- The terms parent and child describe the relationship between levels of the hierarchy.
- Below shows this relationship. Only the root directory has no parents. It is the ancestor of all the other directories.



Important Directories

Following are summaries of some of the more important directories on your UNIX System:

- **/**
  This is the root directory. It is the highest-level directory and all other directories branch from it
- **/home**
  Also called the login directory, is the directory that serves as the repository for a user's personal files, directories and programs.

Important Directories

- **/usr**
  This directory holds unix system resources directories. In other words, /usr is where user- land programs and data (as opposed to 'system land' programs and data) are.
  - **/usr/games**
    This directory holds game programs
  - **/usr/bin**
    This directory holds user-oriented UNIX programs
  - **/usr/sbin**
    This directory holds system administration files

Important Directories

Following are summaries of some of the more important directories on your UNIX System:

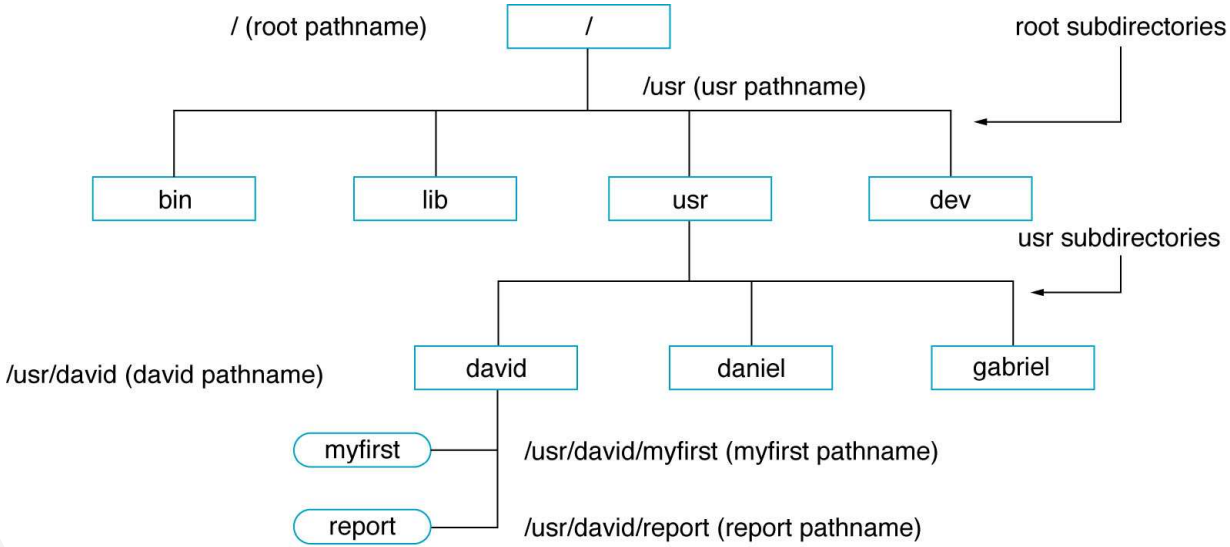
- **/bin**
  This directory holds many of the basic UNIX program files
- **/dev**
  This directory holds device files. These are special files that represent the physical computer components such as printer or disk
- **/sbin**
  This directory holds system files that usually are run automatically by the UNIX system.
- **/etc**
  This directory and its subdirectories hold many of the UNIX configuration files

The Home Directory

- The system administrator creates all user accounts on the system and associates each user account with a particular directory
- This directory is the home directory
- The log on process places you into your home directory
- From your home directory you can expand your directory structure according to your needs
- You can add as many subdirectories as you like and dividing subdirectories into additional subdirectories

Understanding Paths and Pathnames

- Every file has a pathname. The pathname locates the file in the file system
- You determine a file's pathname by tracing a path from the root directory to the file, going through all intermediate directories



Absolute Pathname

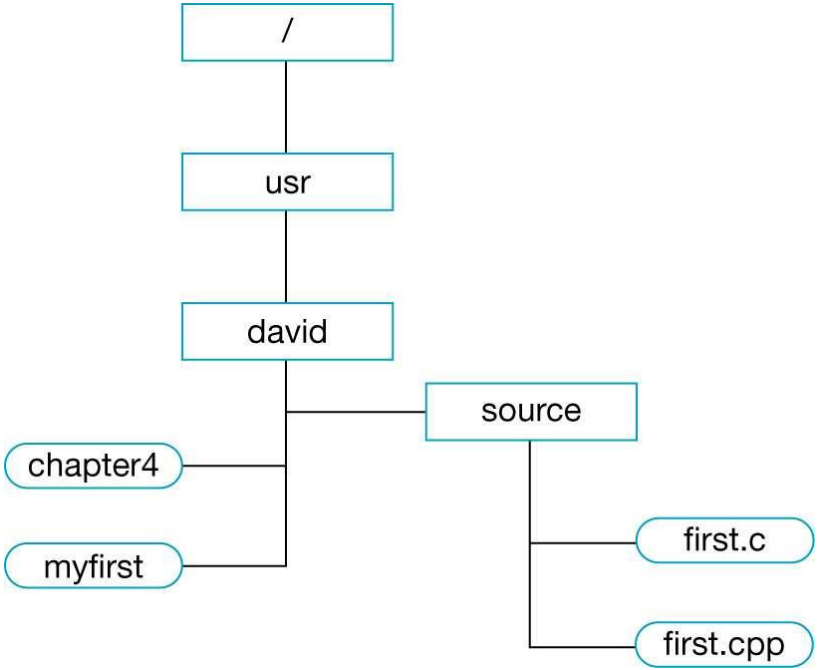
- An absolute pathname (full pathname) traces a path from the root to the file.
- An absolute pathname always begins with the name of the root directory, forward slash (`/`). For example, if your working directory is home, the absolute pathname of the file called `myfirst` under the directory `david` is `/home/david/myfirst`.
- The image in the previous slide are showing different absolute pathnames

Relative Pathname

- A relative pathname is a shorter form of the pathname. It traces path from the working directory to a file
- Like the absolute pathname, the relative pathname can describe a path through many directories.
- For example, if `david's` working directory is `home (~)`, the relative pathname to the file called report under the `david` directory is simple `report`
- **Note**: There is no initial forward slash (`/`) for a relative pathname. It always starts from your current directory.

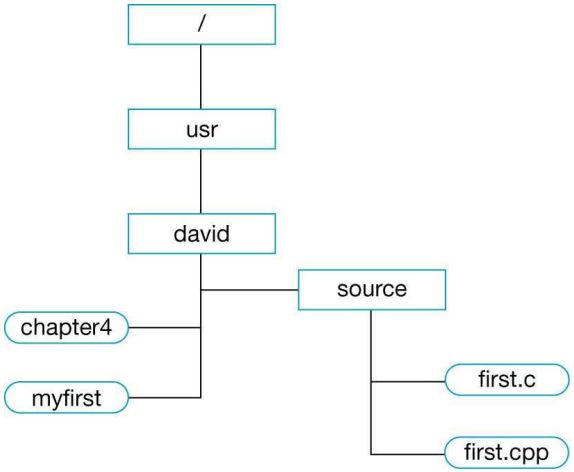
DIRECTORY COMMANDS

- Assuming this is our directory structure, and our home directory is `david`.



DIRECTORY COMMANDS

- `/home/david` is your home directory pathname.
- `/home/david` is also your current or working directory pathname.
- `/home/david` is an absolute pathname because it begins with /, tracing the path of your home directory from the root.
- `david` is your login name and your home directory name.

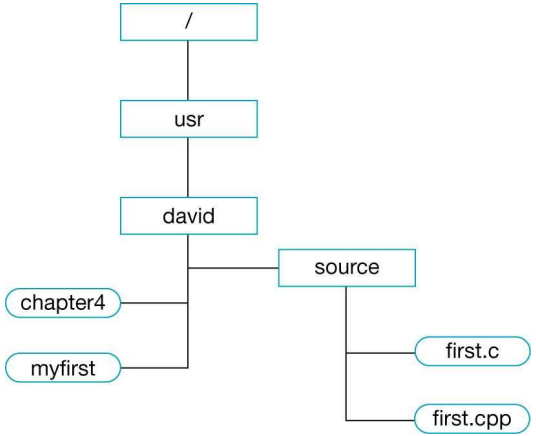


Displaying a Directory Pathname: The `pwd` Command

- The `pwd` (print working directory) command displays the absolute pathname of your working (current) directory.

```
$ pwd [Return]    #Display your HOME directory path.

/home/david
```



Changing Your Working Directory: The `cd` Command

- To change your working directory to the source directory, use the following command sequence:

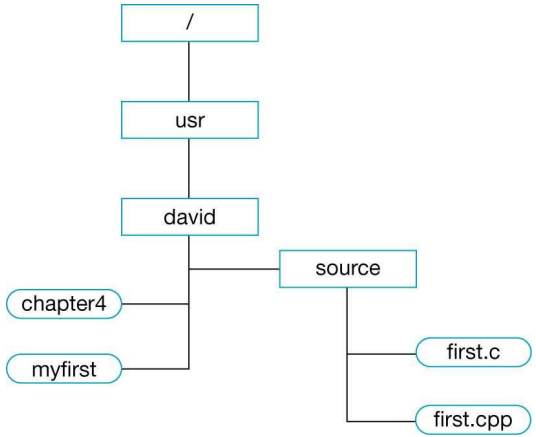
```
$ pwd [Return]          #Check your current directory.

/home/david

$ cd source [Return]    #Change to source directory.

$ pwd [Return]          #Display your working directory.

/home/david/source
```



Creating Directories

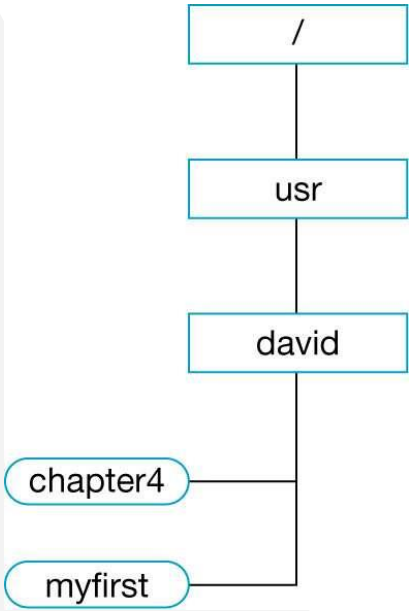
- The very first time you log on to the UNIX system, you begin work from your home directory, which is also your working directory.
- Advantages of Creating Directories
  - Grouping related files in one directory makes it easier to remember and access them.
  - Displaying a shorter list of your files on the screen enables you to find a file more quickly.
  - You can use identical filenames for files that are stored in different directories.

Directory Creation: The `mkdir` Command

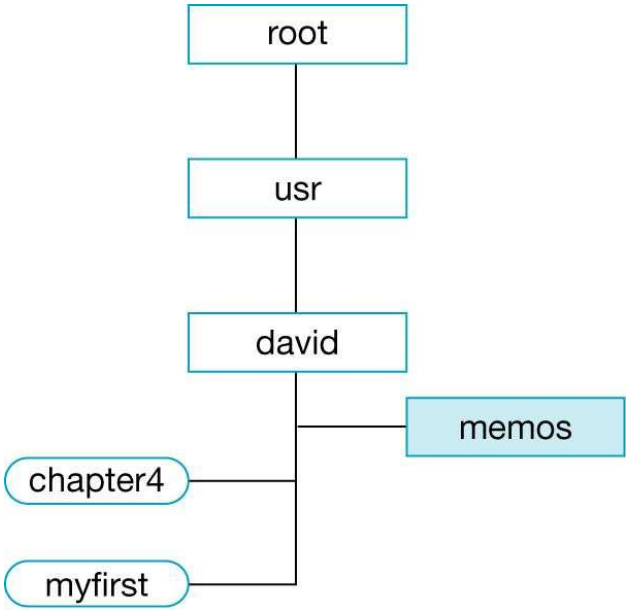
- The `mkdir` (make directory) command creates a new subdirectory under your working directory or any other directory you specify as part of the command.
- Create a directory called memos under your HOME directory:

```
$ cd [Return]           #Make sure you are in your HOME directory.
$ mkdir memos [Return] #Create a directory called memos.
$ pwd [Return]          #Check your working directory.
/home/david
$ cd memos [Return]     #Change to memos directory.
$ pwd [Return]          #Check your working directory.
/home/david/memos       #Your current directory is memos.
```

Directory Creation: The `mkdir` Command



Before



After

Removing Directories: The `rmdir` Command

- The `rmdir` (remove directory) command removes (deletes) the specified directory. However, it removes only empty directories - directories that contain no subdirectories
- Remove the important directory from your memos directory:

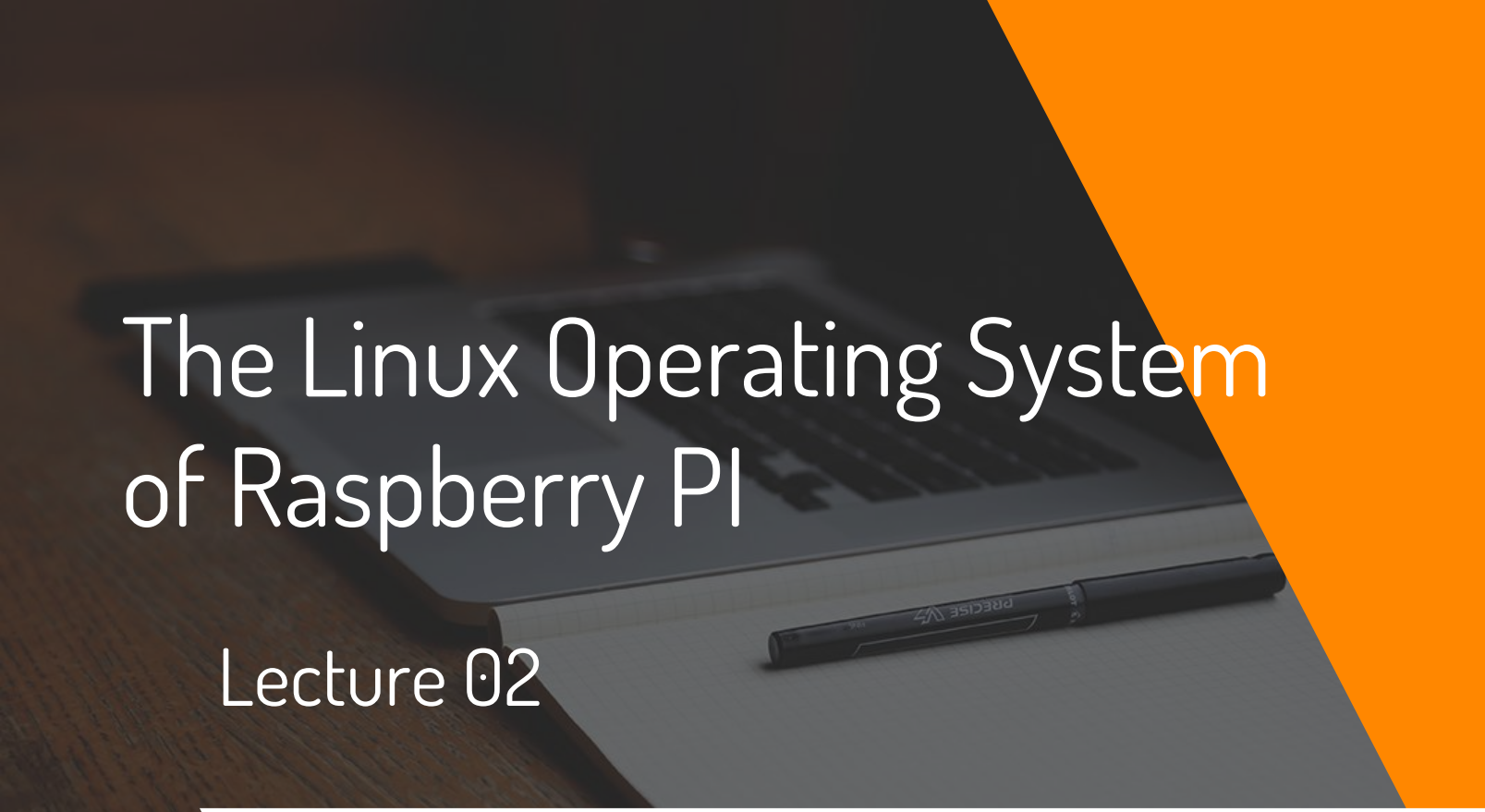
```
$ cd [Return]                  #Make sure you are in your HOME directory.
$ cd memos [Return]            #Change your working directory to memos.
$ pwd [Return]                 #Make sure you are in memos.
/home/david/memos
$_                             #Yes, you are in memos.
$ rmdir important [Return]     #Remove the important directory.
```

Other useful commands

- It is impossible to introduce all the commands for linux
- We will just mention several useful ones
- Feel free to explorer more!
  - `ls` `#list all the directories and files in the working directory`
  - `rm filename` `#delete a file with filename`
  - `rm –r directorypath` `#delete the directory with all its contents`
  - `cat filename` `#peek the content of a file`
  - `cat > filename` `#create a file called filename`
  - `mv d1 d2` `#move d1 to d2`
  - `nano filename` `# open file with the basic nano text editor`

The Linux Operating System of Raspberry PI

Lecture 02



Instructor: Link

Course: CMPT 2200

Learning outcomes

- Learn how to analyze our circuits
- Learn the basic circuit principles
- Learn some discrete components

Learn how to analyze our circuits

Analyzing Your Circuits

- When developing electronics circuits for Raspberry Pi, it is useful to have tools to analyze a circuit
- It should be done before we connect it to the RPi inputs/outputs or power supply
- We do this in order to reduce the chance of damaging our board.
- We will mainly use **Digital multimeter** to do so.

Digital Multimeter (DMM)

- A digital multimeter (DMM) is an invaluable tool for measuring the voltage, current, and resistance of our circuits.
- It usually has the following features:
  - Auto power off
  - Multi range – to select different measurement range
  - Continuity testing – The multimeter will beep only when circuit is complete; good to test if conductors are open or shorted, switches are operating properly, circuit paths are clear

Basic Circuit Principles

Electronic Components

- Electronic circuits contain arrangements of components that can be described as being either passive or active.
  - Active components, such as transistors, are those that can adaptively control the flow of current,
  - Passive components cannot (e.g., resistors, capacitors, diodes).
- The challenge in building circuits is designing a suitable arrangement of appropriate components. Fortunately, there are circuit analysis equations to help you

Voltage, Current, Resistance, and Ohm's Law

The most important equation that you need to understand is Ohm's law. It is simply stated as follows:

$$V = I \times R$$

where:

- **Voltage (V)**, measured in volts (V), is the difference in potential energy that forces electrical current to flow in the circuit.
- **Current (I)**, measured in amps (A), is the flow of electrical charge.
- **Resistance (R)**, measured in ohms (Ω), discourages the flow of charge.

Voltage

- **Voltage (V)**, measured in volts (V), is the difference in potential energy that forces electrical current to flow in the circuit.
- A water analogy is very useful when thinking of voltage; many houses have a buffer tank of water in the attic that is connected to the taps in the house.
- Water flows when a tap is turned on, due to the height of the tank and the force of gravity.
- If the tap were at the same height as the top of the tank of water, no water would flow, because there would be no potential energy.
- Voltage behaves in much the same way; when a voltage on one side of a component, such as a resistor, is greater than on the other side, electrical current can flow across the component.

Current

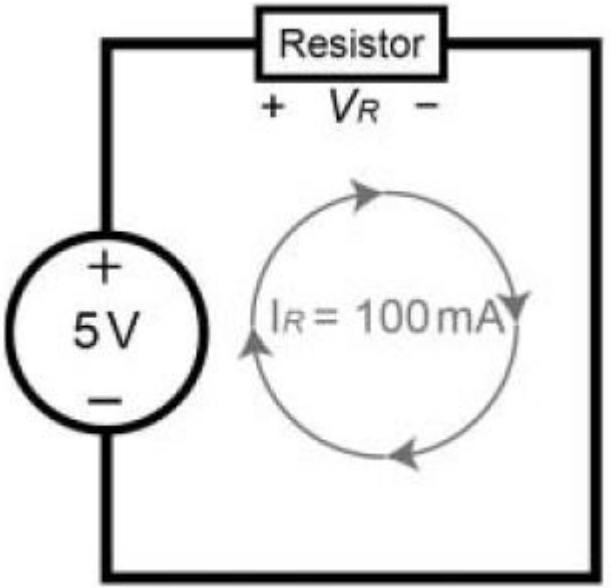
- **Current (I)**, measured in amps (A), is the flow of electrical charge.
- To continue the water analogy, current would be the flow of water from the tank (with a high potential) to the tap (with a lower potential).
- Remember that the tap still has potential and water will flow out of the drain of the sink, unless it is at ground level (GND).
- To put the level of current in context, when we build circuits to interface with the RPi's GPIOs, they usually source or sink only about 3 mA, where a milliamp is one thousandth of an amp.

Resistance

- **Resistance (R)**, measured in ohms (Ω), discourages the flow of charge.
- A resistor is a component that reduces the flow of current through the dissipation of power. It does this in a linear fashion, where the power dissipated in watts (W), is given by $P = V \times I$ or, alternatively by integrating Ohm's law: $P = I^2 R = V^2 / R$.
- The power is dissipated in the form of heat, and all resistors have a maximum dissipated power rating. Common metal film or carbon resistors typically dissipate 0.125 W to 1 W
- To finish with the water analogy, resistance is the friction between the water and the pipe, which results in a heating effect and a reduction in the flow of water. This resistance can be increased by increasing the surface area over which the water has to pass, while maintaining the pipe's cross-sectional area (e.g., placing small pipes within the main pipe).

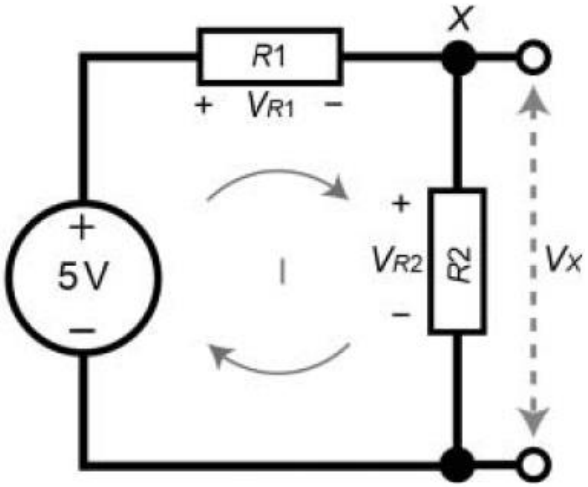
Checkpoint

- If we had to use a resistor that limits the flow of current to 100 mA when using a 5V supply, as illustrated on the right, which resistor should we use?
- The voltage dropped across the resistor, VR, must be 5 V, as it is the only component in the circuit.
- Because V = I × R, it follows that the resistor should have the value R = V⁄ I = (5 V) ⁄ (100 mA) = 50 Ω.
- And the power dissipated by this resistor can be calculated using any of the general equations P = VI = $I^2 R = V^2 / R$ as 0.5 W.



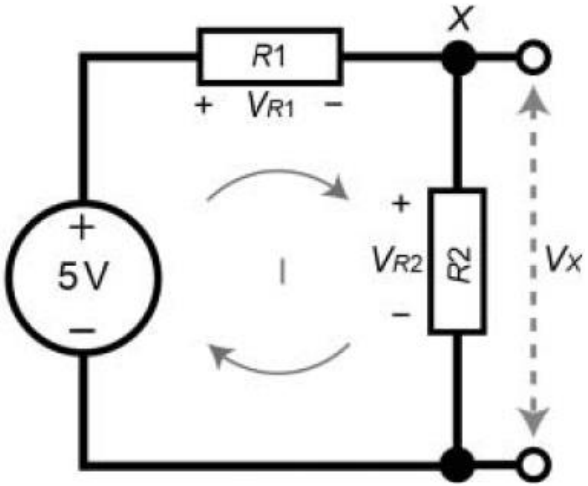
Voltage Division

- If the circuit in previous slide is modified to add another resistor in series as illustrated in Figure on the right, what will be the impact on the circuit?
- Because one resistor is after the other (they're in series), the total resistance that the current must pass through to circulate in the circuit is the sum of the two values: R_T = R1 + R2 .
- The supply voltage must drop across the two resistors, so you can say that $V_{supply} = V_{R1} + V_{R2}$. The voltage that drops across each resistor is inversely proportional to the resistor's value. This circuit is called a voltage divider.



Voltage Division

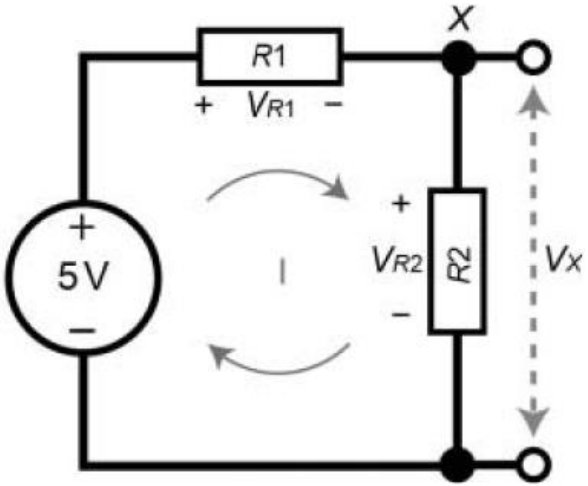
- Suppose we want to calculate the voltage value at point X if R1 = 25 Ω and R2 = 75 Ω. The total resistance in the circuit is R_T = 25 + 75 = 100 Ω, and the total voltage drop across the resistors must be 5 V.
- Therefore, by using Ohm's law, the current flowing in the circuit is I = V/R = 5 V/100 Ω = 50 mA. If the resistance of R1 is 25 Ω, then the voltage drop across V_{R1} = I × R = 0.05 A × 25 Ω = 1.25 V and the voltage drop across V_{R2} = I × R = 0.05 A × 75 Ω = 3.75 V. The sum of these voltages is 5 V, thus obeying Kirchoff's voltage law, which states that the sum of the voltage drops in a series circuit equals the total voltage applied.



Voltage Division

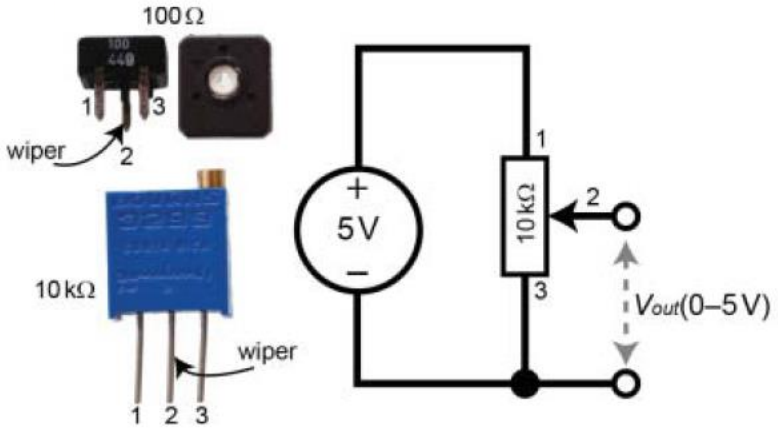
- To calculate the value of V_X, the voltage divider rule can be generalized to the following:

$$V_x = V \times \frac{R2}{R1 + R2}$$



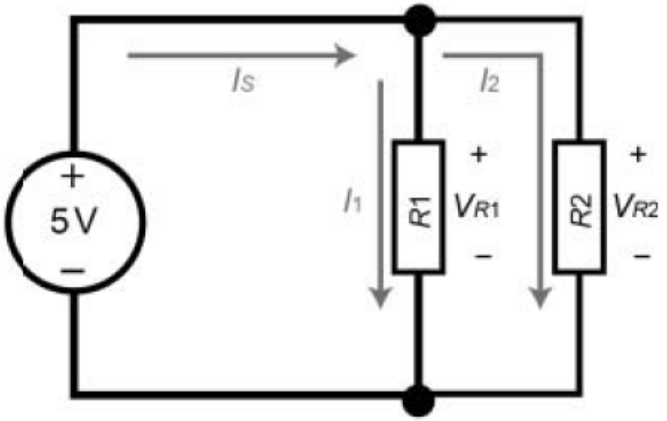
Voltage Division

- Figure on the right captures a variable resistor, and an associated circuit where it is used as a standalone voltage divider. The resistance between pins 1 and 3 is a fixed value, 10 kΩ in the case.
- However, the resistance between pins 3 and the wiper pin (pin 2) varies between 0 Ω and 10 kΩ.
- Therefore, if the resistance between pins 2 and 3 is 2 kΩ, then the resistance between pins 1 and 2 will be 10 kΩ − 2 kΩ = 8 kΩ.



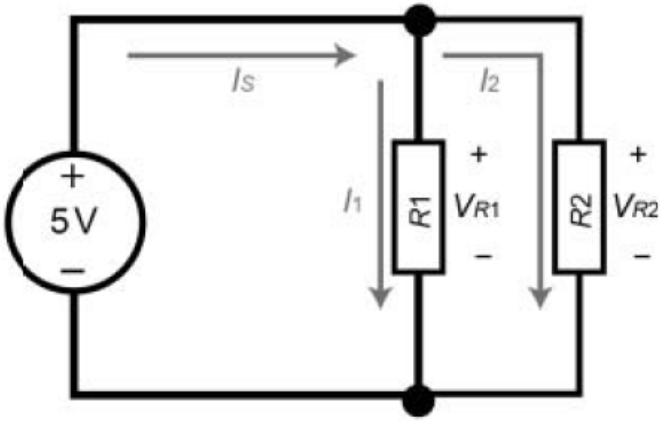
Current Division

- If the circuit is modified as in Figure on the right, to place the two resistors in parallel, we now have a current divider circuit.
- Current will follow the path of least resistance, so if R1 = 100 Ω and R2 = 200 Ω, then a greater proportion of the current will travel through R1.
- So, what is this proportion?



Current Division

- In this case the voltage drop across R1 and R2 is 5 V in both cases. Therefore, the current I1 will be I = V/R = 5 V/100 Ω = 50 mA and the current I2 will be I = 5 V/200 Ω = 25 mA.
- Therefore, twice as much current travels through the 100 Ω resistor as the 200 Ω resistor. Clearly, current favors the path of least resistance.

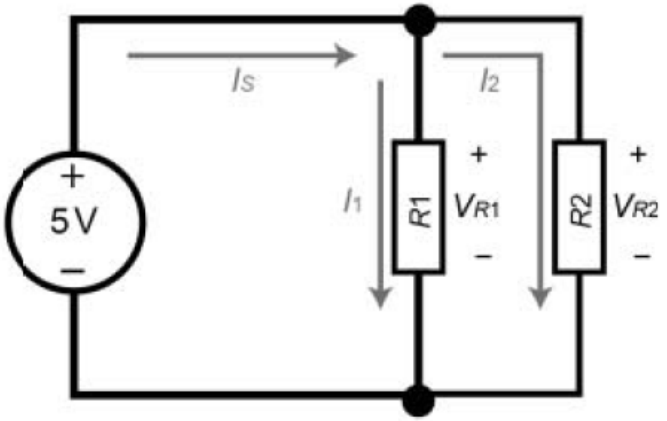


Current Division

- Kirchoff's current law states that the sum of currents entering a junction equals the sum of currents exiting that junction. This means that IS = I1 + I2 = 25 mA + 50 mA = 75 mA. The current divider rule can be stated generally as follows:

$$I1 = I \times (\frac{R2}{R1 + R2})$$

$$I2 = I \times (\frac{R1}{R1 + R2})$$



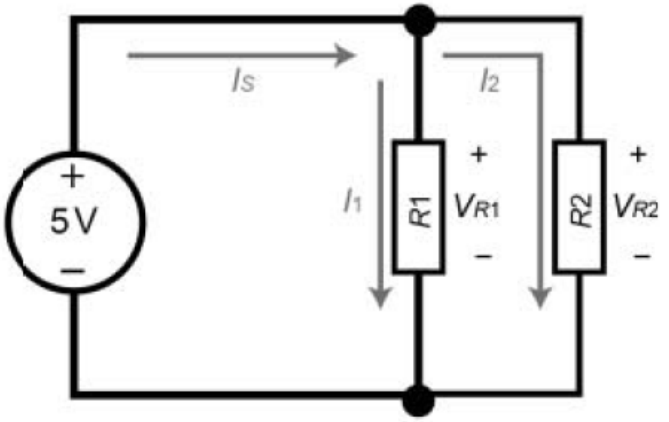
Current Division

- However, this requires that we know the value of the current I (IS in this case) that is entering the junction. To calculate IS directly, we need to calculate the equivalent resistance (RT) of the two parallel resistors, which is given as follows:

$$\frac{1}{R_T} = \frac{1}{R1} + \frac{1}{R2}$$

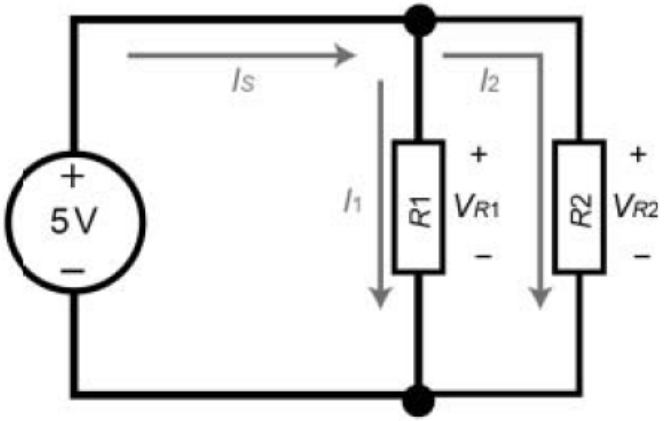
or

$$R_T = \frac{R1 \times R2}{R1 + R2}$$



Current Division

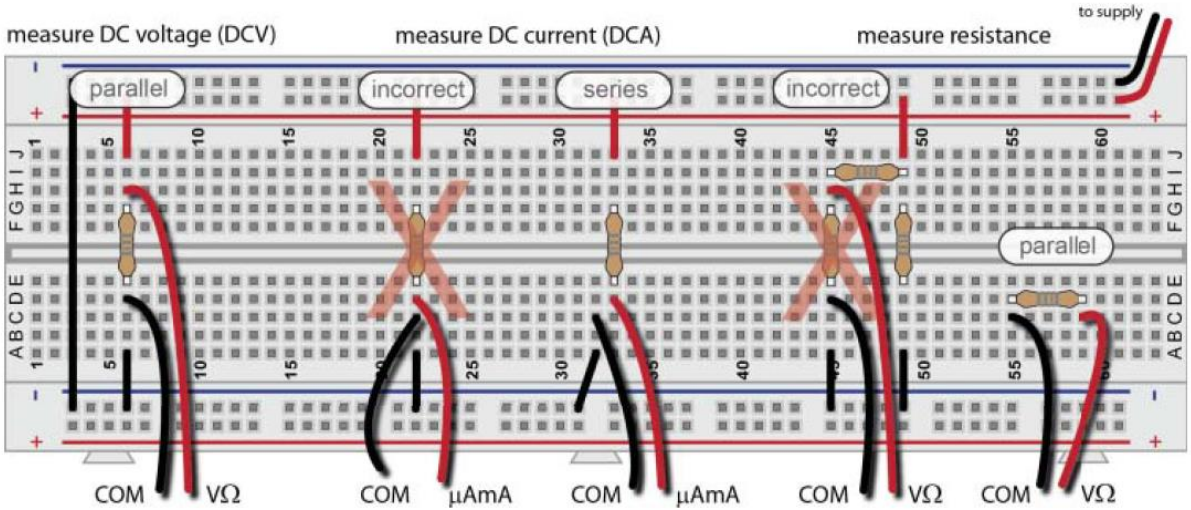
- This is 66.66 Ω in Figure; therefore IS = V/R = 5 V/66.66 Ω = 75 mA, which is consistent with the initial calculations.
- The power delivered by the supply: P = VI = 5 V × 0.075 A = 0.375 W. This should be equal to the sum of the power dissipated by R1 = V2/R = 52/100 = 0.25 W and, R2 = V2/R = 52/200 = 0.125 W giving 0.375 W total, confirming that the law of conservation of energy applies!



Discrete Components

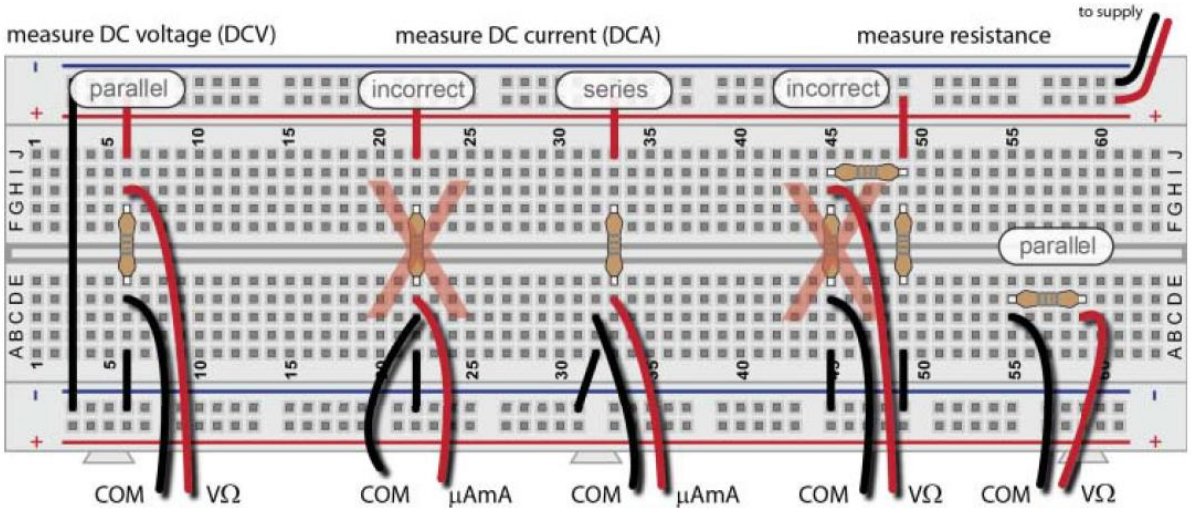
Breadboard

- The breadboard is a great platform for prototyping circuits and it works perfectly with the RPi.
- We could measure voltage, current and resistance on the breadboard following several rules



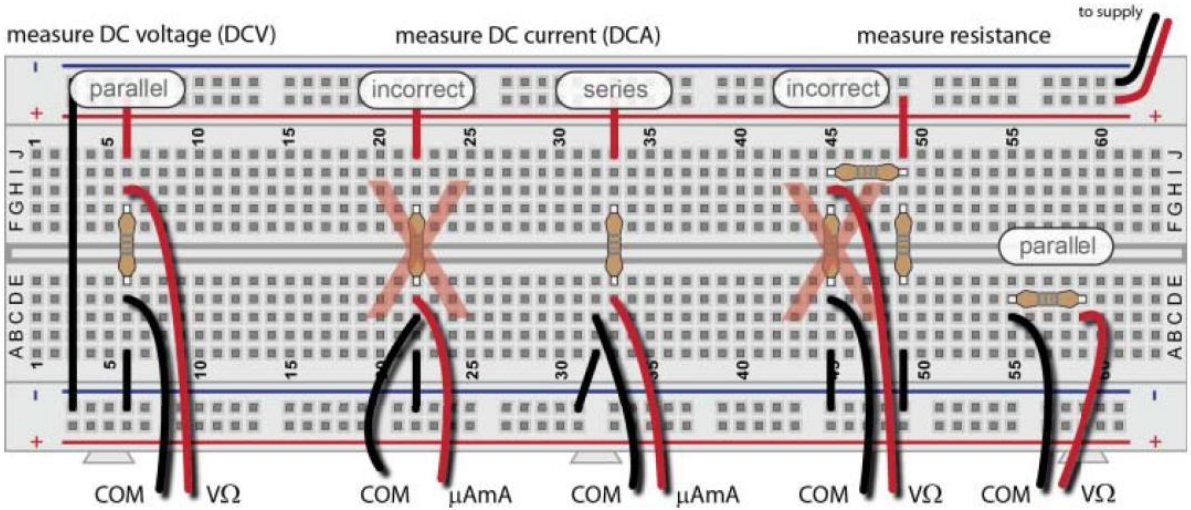
Measure with rules

- DC voltage (DCV) is measured in parallel with (i.e., across) the component that experiences the voltage drop. The meter should have the black probe in the COM (common) DMM input.



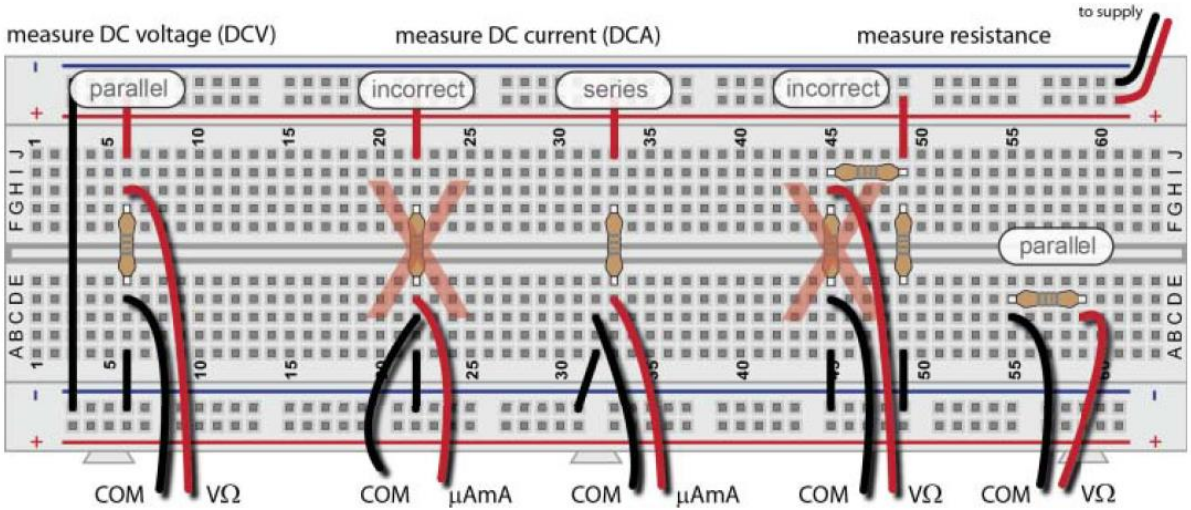
Measure with rules

- DC current (DCA) is measured in series, so we will have to “break” the connection in our circuit and wire the DMM as if it were a component in series with the conductor in the circuit in which you are measuring current.



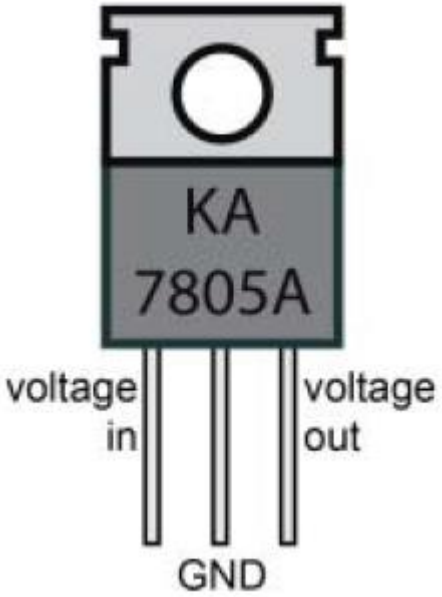
Measure with rules

- Resistance cannot usually be measured in-circuit, because other resistors or components will act as parallel/series loads in your measurement. Isolate the component and place your DMM red probe in the VΩ input and set the meter to measure Ω.



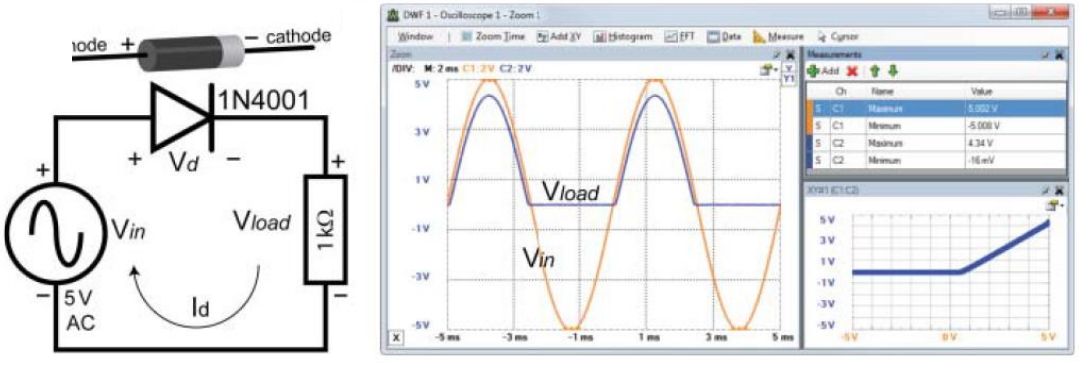
Voltage Regulation

- A voltage regulator is a complex but easy-to-use device that accepts a varied input voltage and outputs a constant voltage almost regardless of the attached load, at a lower level than the input voltage.
- The voltage regulator maintains the output voltage within a certain tolerance, preventing voltage variations from damaging downstream electronics devices.



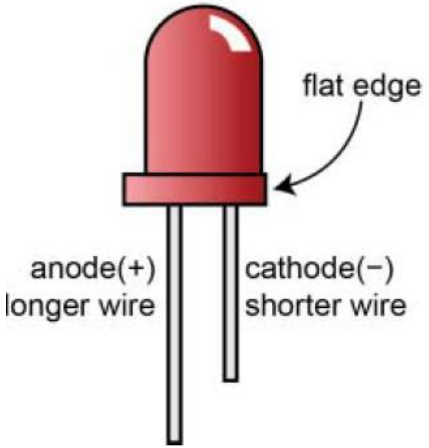
Diodes

- Simply put, a diode is a discrete semiconductor component that allows current to pass in one direction but not the other.
- When a diode is forward biased it allows current to flow through it; when it is reverse-biased, no current can flow.
- The diode is used in the circuit as a reverse polarity protector. It should be clear from the plot in Figure why it is effective, as when Vin is negative, the Vload is zero.



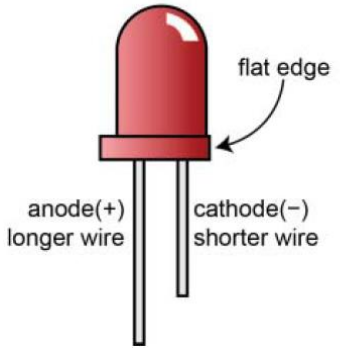
Light-Emitting Diodes (LEDs)

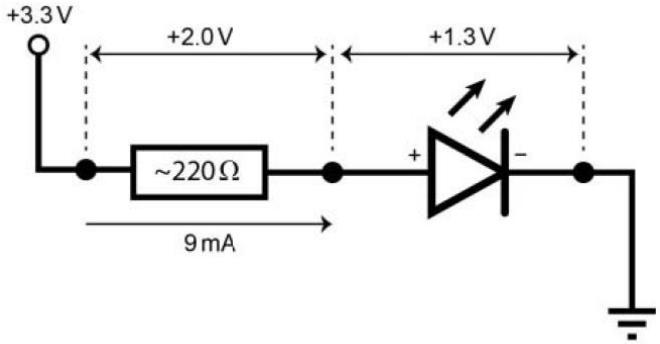
- A light-emitting diode (LED) is a semiconductor-based light source that is often used as a state indication light in all types of devices.
- Today, high-powered LEDs are being used in car lights, in back lights for televisions and for general-purpose lighting (e.g., home lighting, traffic lights, etc.) mainly due to their longevity and extremely high efficiency in converting electrical power to light output.
- LEDs provide very useful status and debug information about your circuit, often used to indicate whether a state is true or false.



Light-Emitting Diodes (LEDs)

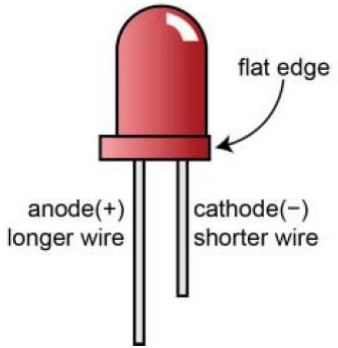
- Like diodes, LEDs are polarized. The symbol for an LED is illustrated To cause an LED to light, the diode needs to be forward biased by connecting the anode (+) to a more positive source than the cathode (−).
- For example, the anode could be connected to +3.3 V and the cathode to GND.
- Figure also illustrates an LED that has one leg longer than the other. The longer leg is the anode (+) and the shorter leg is the cathode (−).

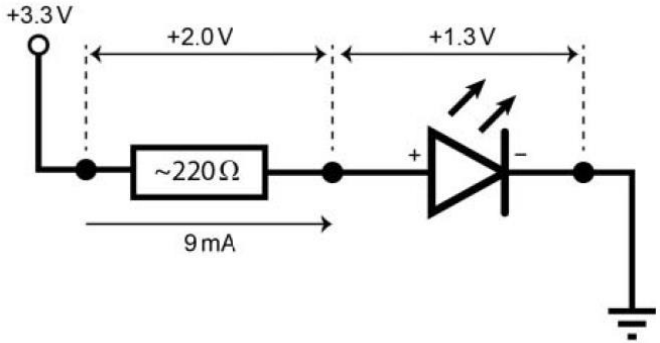




Light-Emitting Diodes (LEDs)

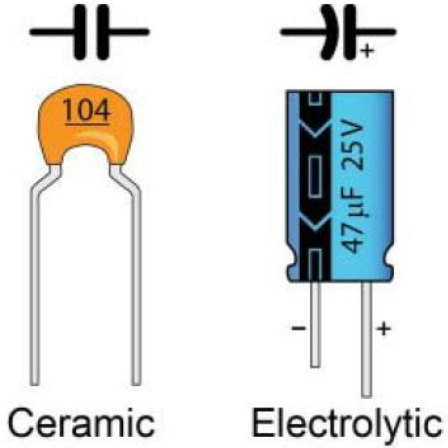
- An LED does not have a significant resistance, so if you were to connect the LED directly across our RPi's 3.3 V supply, the LED would act like a short circuit, and you would drive a very large current through the LED, damaging it—but more important, damaging our RPi!
- Therefore, to operate an LED within its limits we need a resistor, called a current-limiting resistor. Choose this value carefully to maximize the light output of the LED and to protect the circuit.





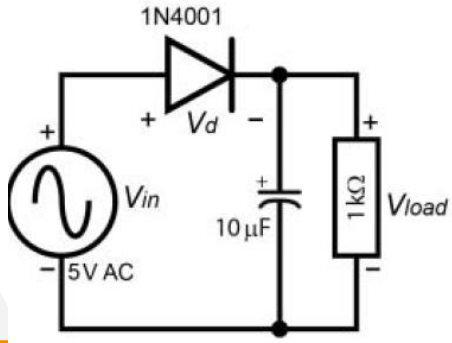
Capacitors

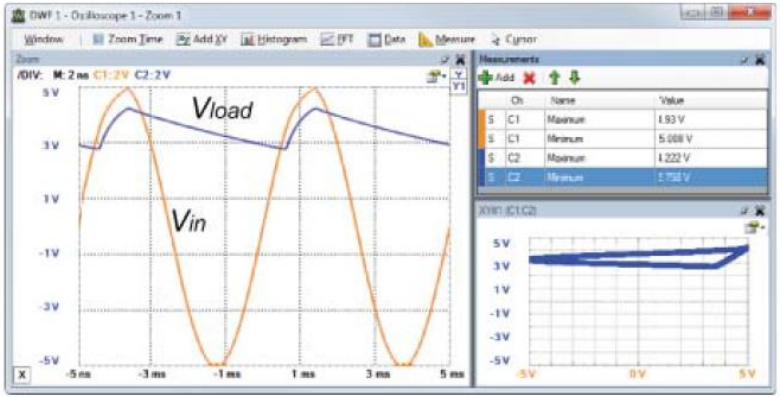
- A capacitor is a passive electrical component that can be used to store electrical energy between two insulated plates when there is a voltage difference between them.
- The energy is stored in an electric field between the two plates, with positive charge building on one plate and negative charge building on the other plate.
- When the voltage difference is removed or reduced, then the capacitor discharges its energy to a connected electrical circuit.



Capacitors

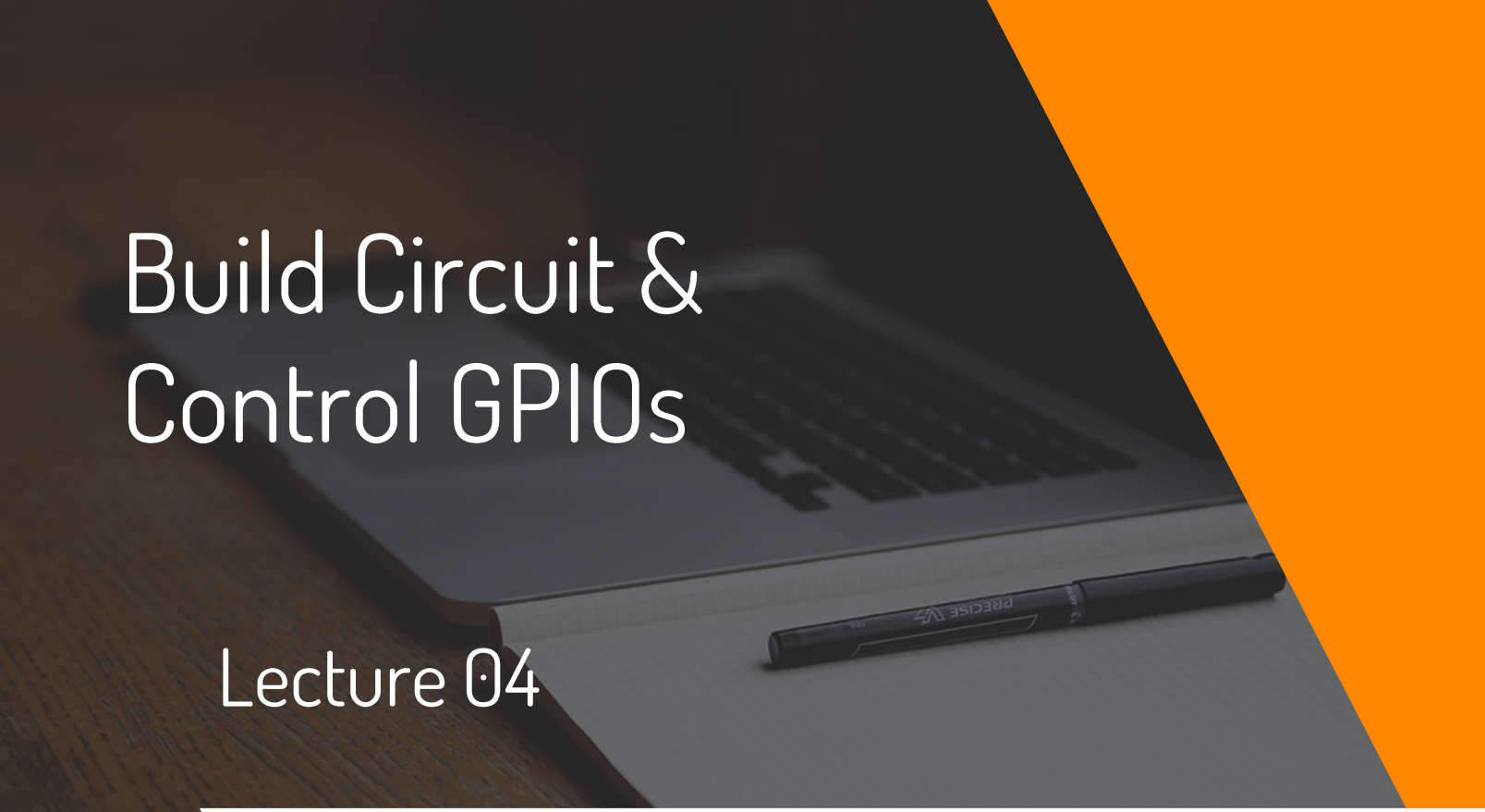
- For example, if we modified the diode circuit to add a 10 μF smoothing capacitor in parallel with the load resistor, the output voltage would appear as shown in Figure.
- When the diode is forward biased there is a potential across the terminals of the capacitor and it quickly charges.
- When the diode is reverse biased, there is no external supply, so the potential across the terminals of the capacitor (because of its charge) causes a current to flow through the load resistor, and the capacitor starts to discharge.





Build Circuit & Control GPIOs

Lecture 04



Instructor: Link

Course: CMPT 2200

Learning outcomes

- Learn how to safely use our board
- Understand how a breadboard works (again!)
- Learn how to read resistor color code
- Learn how to build our first LED Circuit

Safe Manipulation of Board

Risk Warning ahead!

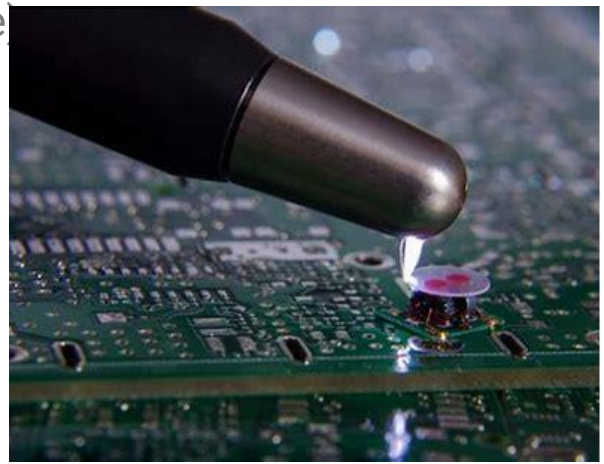
- Working with Rapsberry Pi without following the instructions could be quite dangerous
  - Like plug a wire in the wrong place, or mess up the electric current direction
- Not dangerous to ourselves
  - The current is too low to hurt anyone
- But might fry our board, or even break the Raspberry Pi

Safety Check List

1. **Wish to make a change in the circuit? - Always power OFF the Raspberry Pi first**
   - Even if it is just a simple connection or connection of single component or wire
   - You could turn off it through Menu or through command line
     - `$ sudo shutdown now`
   - Or you could customize a power off button yourself. (After we learn

Safety Check List

1. Wish to make a change in the circuit? - Always power OFF the Raspberry Pi first
2. **Raspberry Pi is powered On? - Always avoid touching the circuit or components**
   - Neither with your finger nor any other tool like a screwdriver
   - May damage the Pi with ESD (electrostatic discharge)



Safety Check List

1. Wish to make a change in the circuit? - Always power OFF the Raspberry Pi first
2. Raspberry Pi is powered On? - Always avoid touching the circuit or components
3. **Wish to plug in power cable? - Double/Triple check everything every time**
   - Even if you are pretty sure about it

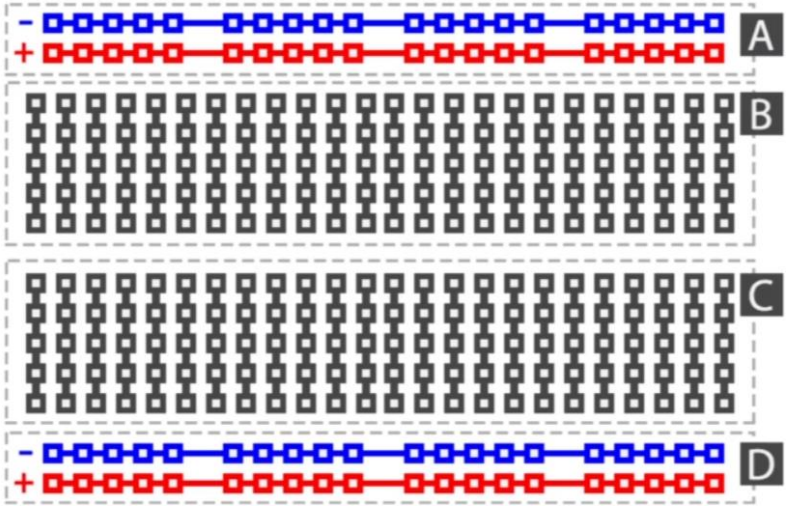
Safety Check List

1. Wish to make a change in the circuit? - Always power OFF the Raspberry Pi first
2. Raspberry Pi is powered On? - Always avoid touching the circuit or components
3. Wish to plug in power cable? - Double/Triple check everything every time
4. Start working on circuit? - Always start by connecting GND(grounding) for all components
5. Don't put a 5V signal on a GPIO pin – pin only supports 3.3V max

The Breadboard

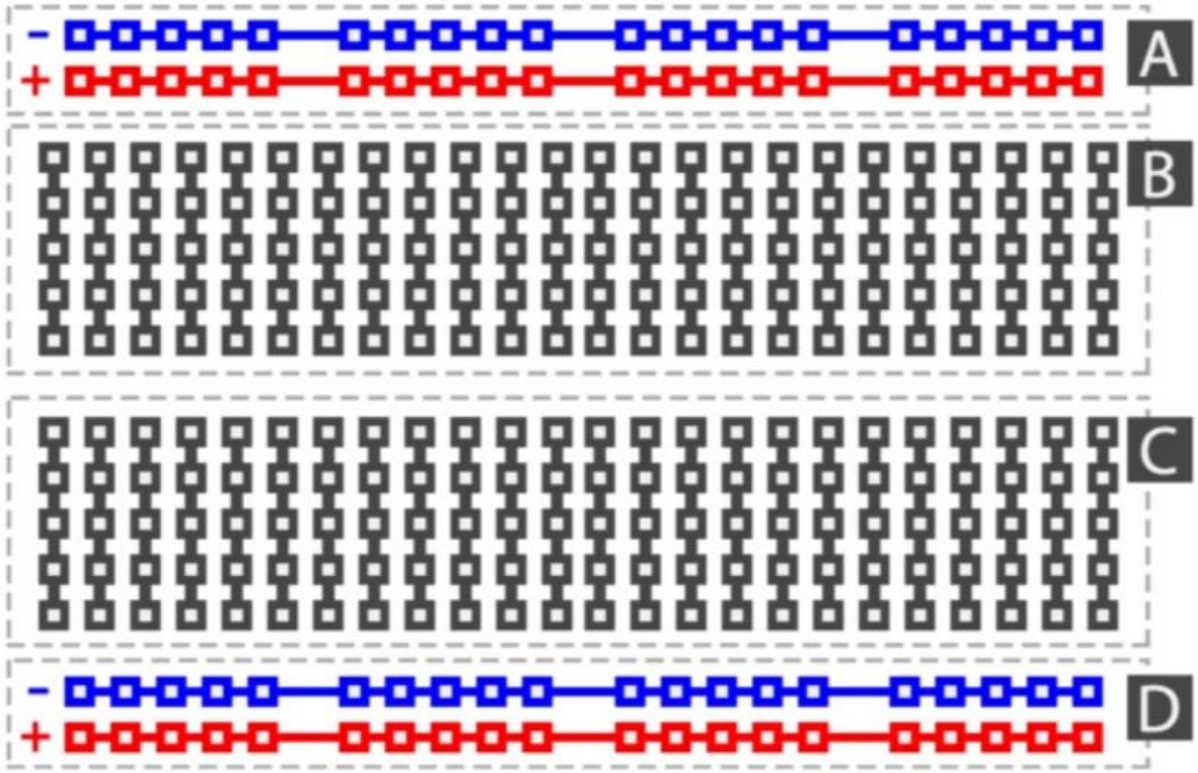
The Breadboard

- **Breadboard** is used to connect multiple components together and also GPIOs
  - Wire, resister, LED, Button
- Underneath the surface there are some metal lines that make connections between components
  - A component that we plug on a line is electrically connected to all the other components on that line

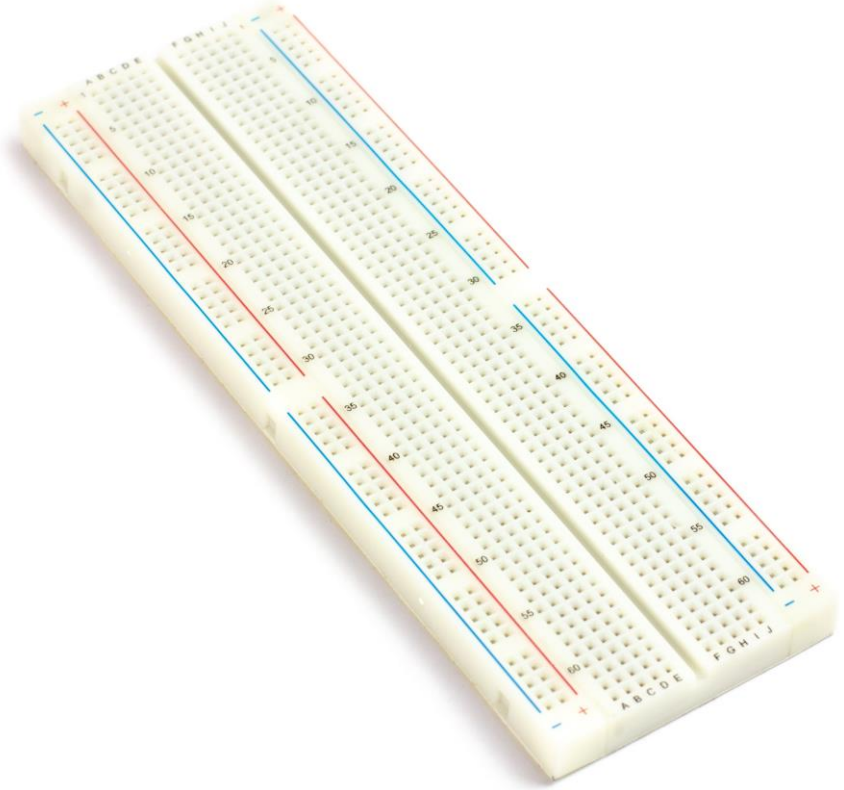


The Breadboard

- A & D: Horizontal main lines
  - Blue line: for GND
  - Red line: for Power Supply
  - All points in each row is connected
- B & C: Vertical column lines
  - Each column is independent
  - All points in each column is connected



Check the actual board!



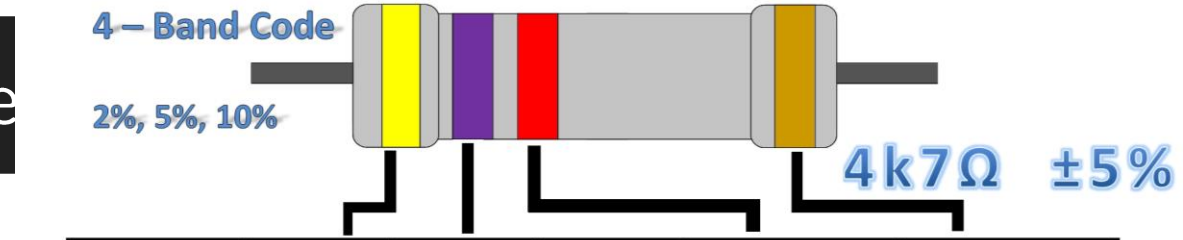
The Resistor Color Code

Pick a Resistor

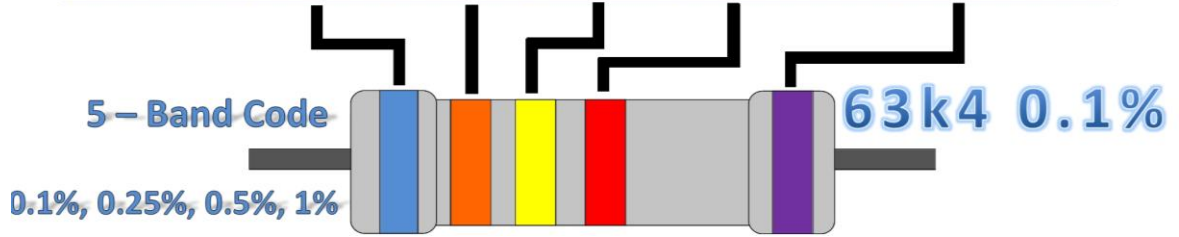
- As mentioned, resistor is usually in circuit to lower the amount of current that goes through other components, for example LEDs.
- It is also helping to protect the GPIOs, so they won't get burned with too much current.
- In our course, we will mainly use 220Ohm, 10kOhm resistors.
- How to recognize a resistor?

Color Code Table

- We will have to read the value from the color bands of the resistors.
- Right is a table to help you find the value of a resistor with the color on it.
- You will also find the Resistor Colors Quick Reference Cards in our Raspberry Pi Box.

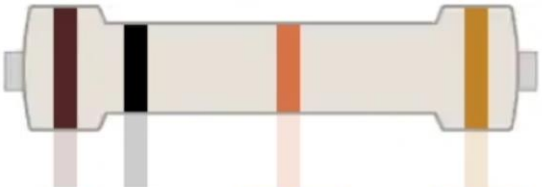


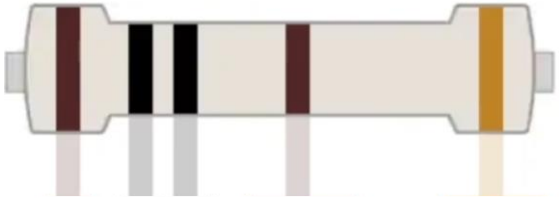
Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1%
Red	2	2	2	100Ω	± 2%
Orange	3	3	3	1kΩ	
Yellow	4	4	4	10kΩ	
Green	5	5	5	100kΩ	± 0.5%
Blue	6	6	6	1MΩ	± 0.25%
Violet	7	7	7	10 MΩ	± 0.1%
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1Ω	± 5%
Silver				0.01Ω	± 10%

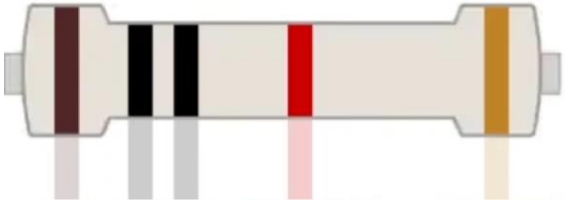


Checkpoint

- What is the resistance value for each resistor?







Pick a Resistor for LED

How to choose resistor for LED?

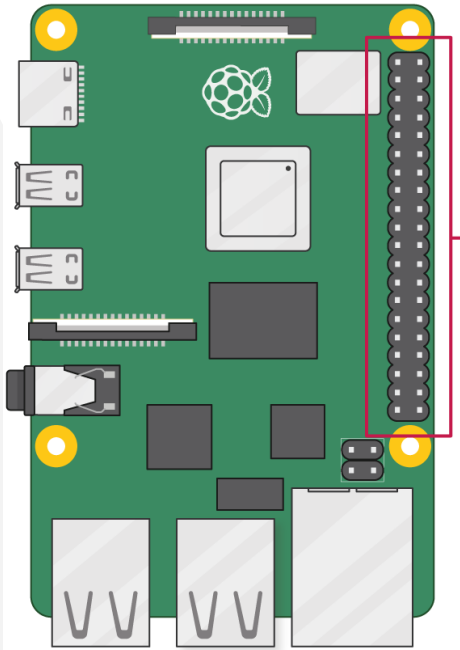
1. Understand three concept:
   - Supply Voltage – The power we provide to the circuit. (5V or 3.3V from Raspberry Pi)
   - LED Forward Voltage – The Voltage drop when electric goes through the LED
   - LED Forward Current – The proper current that goes through LED (could vary within a range)
2. Check the LED Voltage and Current
   - Usually LED Voltage is between 2V – 3V, LED Current is between 15 mA – 25 mA
3. Use the formular we learned last week to calculate necessary resistance
4. Specifically for LED, we could also use Led Calculator online

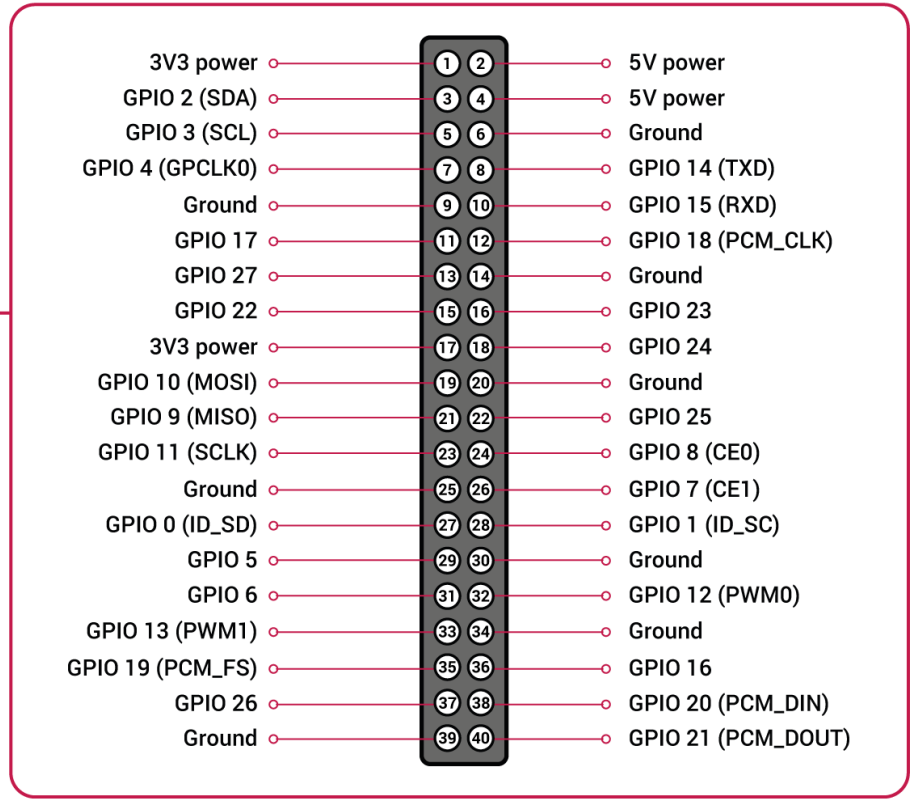
Our First Circuit

What we need

- We need at least 2 wires
  - female-male if you wish to directly connect GPIOs to breadboard
  - male-male if you have T-Shaped Breakout Board and Ribbon Cable connected with GPIOs
- A 220 Ω resistor
- An LED with the color of your choice

GPIO pinout Schematics





Steps of connection

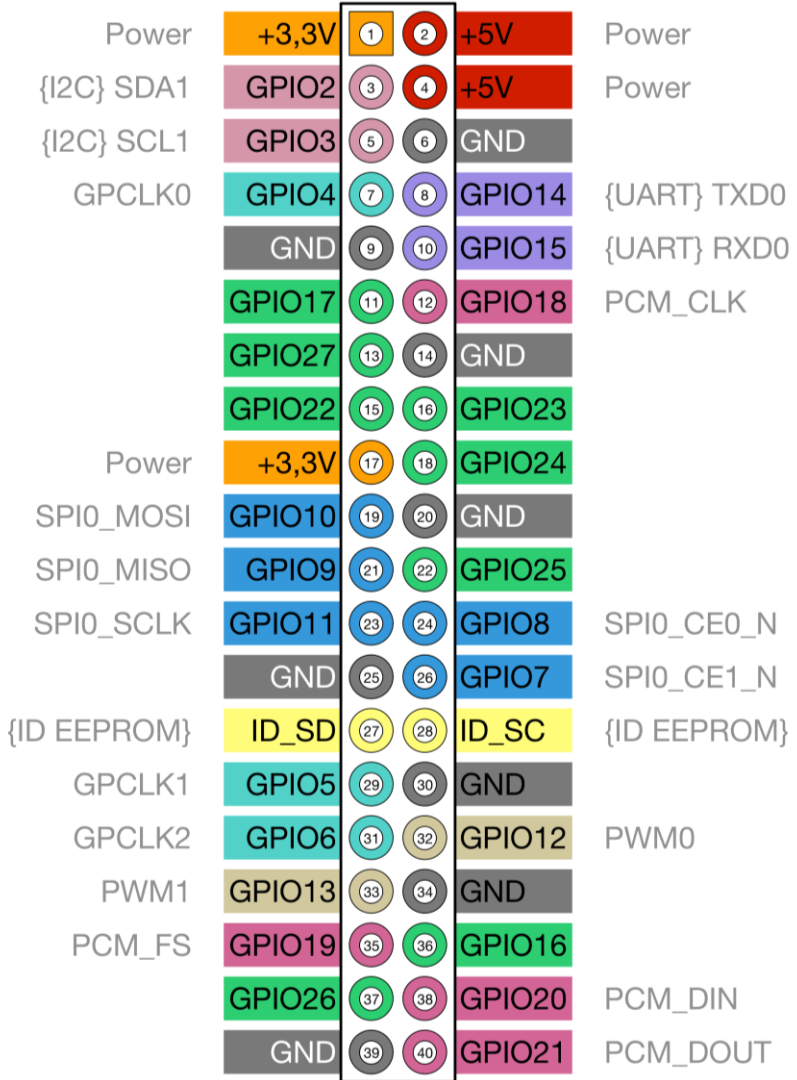
Warning: Do not connect the two legs of any individual component

1. Connect the LED short leg to the ground line
2. Connect the LED long leg to one leg of resistor
3. Connect another leg of resistor to the power line
4. Connect ground line to one of GND pin on GPIOs (let's choose pin 6)
5. Connect power line to 5V power pin on GPIOs (let's choose pin 2)
6. Now power on the Raspberry Pi

Control Raspberry Pi's GPIOs with Python

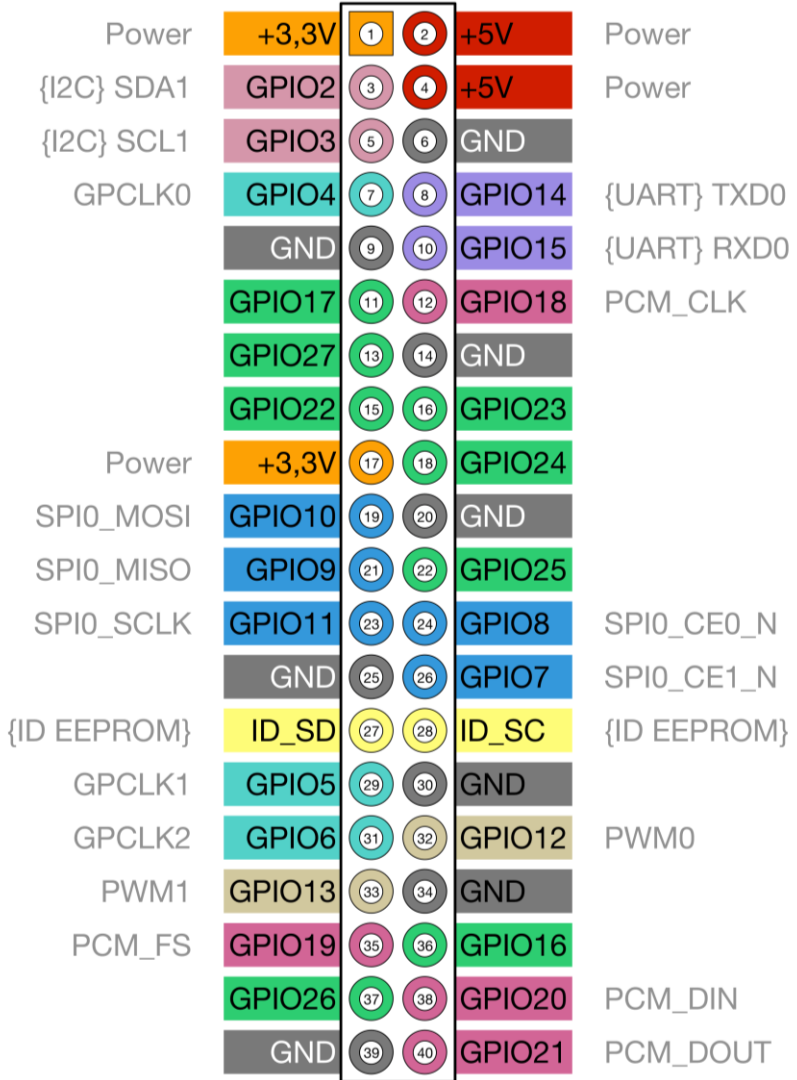
Explorer more of GPIOs (with colors!)

- Right is the complete pinout for Raspberry Pi 4 (also 2, 3 & 5)
- Pins in grey represent the ground. – They are connected together!
- Pins in red(5V) and orange(3.3V) represent power.
- 2 Reserved Pin (27 & 28)
  - Used for HAT's ROM communication
  - Not covered in our course



Explorer more of GPIOs (with colors!)

- Most of the rest are GPIOs
  - Each GPIO has a number that we could use in our python program to interact with the specific GPIO
  - We could choose to select pin based on its Pin Number or GPIO number
  - GPIOs are 3.3V pins.
  - Some pins also have alternate function – More advanced usage



GPIOs – Main Function

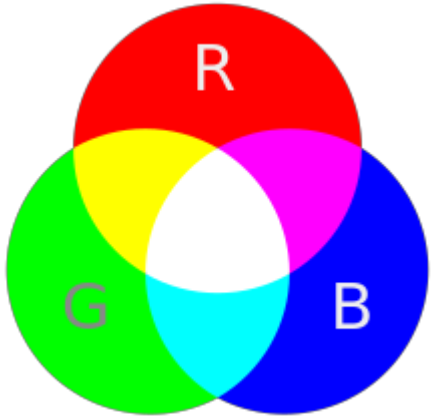
- We first set up a GPIO as an input pin or output pin
  - Use input when reading data (e.g. from a push button or a sensor)
    - Input pin will read a value, which has only 2 states – Voltage High or Low, 1 or 0.
  - Use output when writing data (e.g. send a signal to LED)
    - We can set output pin to be High or Low, 1 or 0 – to power on or off component

LED Blink Python Program

- Now, let's power off the Raspberry Pi and modify the circuit a bit
  - We change the wire that connects from resistor to power line, now from resistor to GPIO 23.
- Power on the Raspberry Pi again, and let's write some code

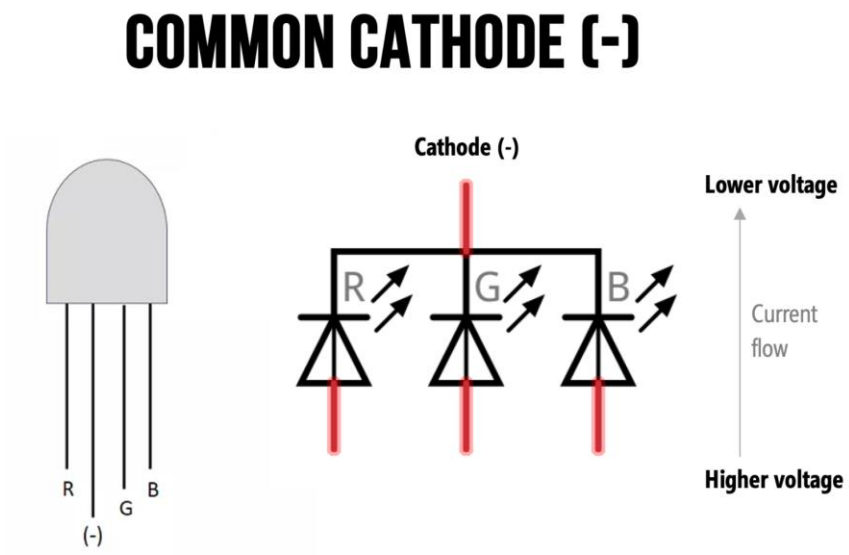
Intro to RGB LED

- An RGB LED is a type of LED that can produce multiple colors by combining the three primary colors of light: red, green, and blue. As shown on the right.
- We can turn Red, Green and Blue on or off to get different colors.
  - For example, if we turn red and green on and blue off, we will get yellow!
  - If we turn only blue on, we will get, of course, blue!
  - If we turn red, green and blue all on, then we will get white!



Intro to RGB LED

- There are two types of RGB LED, what we are using is common cathode type.
  - The longer leg should be connected to the ground.
- The 3 shorter legs should be connected to possibly voltage high. As shown on the right.
  - Always connect power to resistors and use resistors to connect to shorter legs.
  - Remember which leg is corresponding to which color.



RGB LED Python Program

- Now, let's power off the Raspberry Pi and modify the circuit again
  - We connect 3 GPIOs to 3 resistors, and each resistor is connected to one of the shorter legs of RGB LED.
  - Then we use wire to connect longer leg of RGB LED to the ground.
- Power on the Raspberry Pi again, and let's test the RGB LED by shining some colors!

GPIOs input

Lecture 05

Instructor: Link

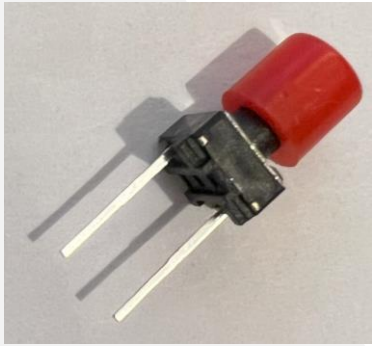
Course: CMPT 2200

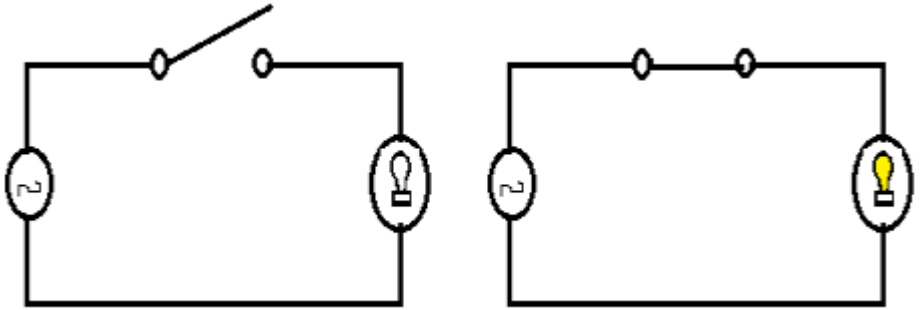
Learning outcomes

- Learn how to connect switch button to the circuit
- Learn how to use GPIOs input to get the signal when we click button
- Learn how to use switch button to control the circuit.

2 legs and 4 legs Switch Button

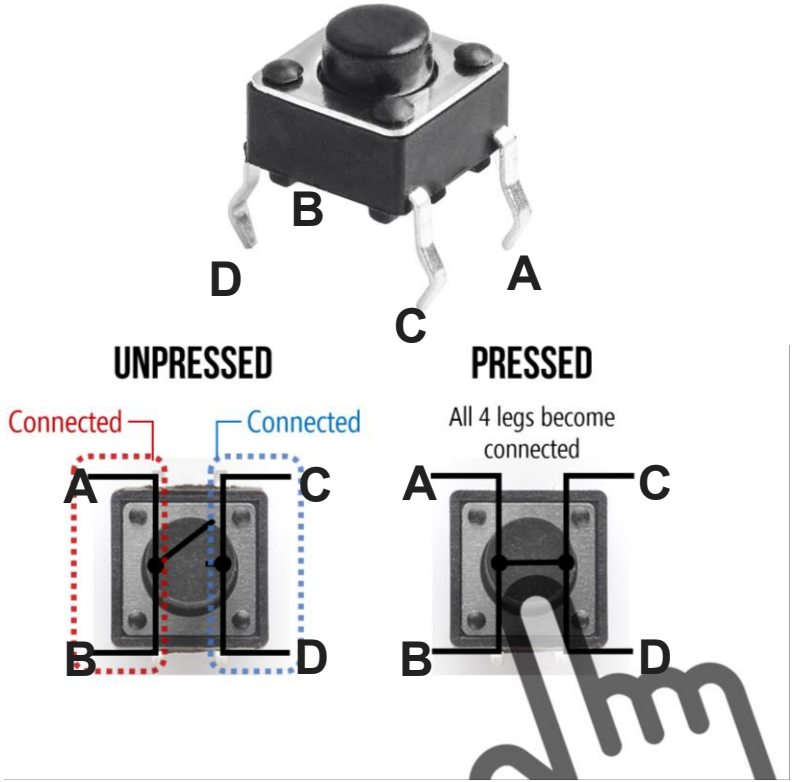
- A 2 legs or 4 legs switch button is a button that we used to control the circuit connections.
- Normally, when we add a switch button to the circuit, it makes the circuit disconnected.
- Only when we press the button, the circuit is connected. Below is diagram and sample photo of a 2 legs switch button.





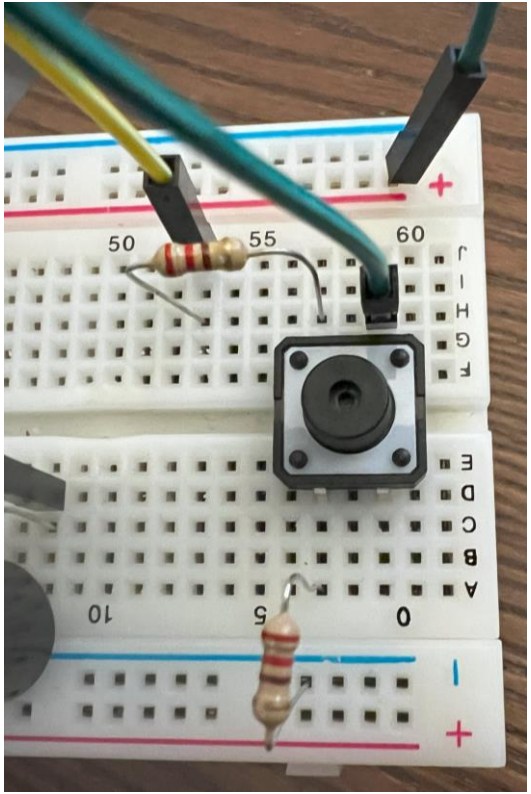
2 legs and 4 legs Switch Button

- 4 legs Switch Button is a bit different.
- Initially, Leg A and B are connected, C and D are connected internally
- When we press the button down, all 4 legs will be connected.
- Now let's connect a 4 legs switch button to the circuit.



Steps of connection

1. Connect the switch button to the middle of the breadboard. Two legs above the gap, and two legs below the gap.
2. Connect the 5V or 3.3V power to the right top leg of our button.
3. Use a resistor (220 ohm) to connect the left bottom leg to the ground.
4. Use a resistor (220 ohm) to connect the left top leg to a wire.
5. Connect that wire to one of the GPIO of our raspberry pi.



GPIOs – Input (switch.py)

- We could set a GPIO pin as input pin by using following code:

```
GPIO.setup(NUMBER, GPIO.IN)
```

- After that, we could visit the current reading value of GPIO pin by using the code below:

```
GPIO.input(NUMBER)
```

- The value of the code above will be either `GPIO.HIGH(1)` or `GPIO.LOW(0)`
- A GPIO input pin will have floating value between 0 and 1 if it is not connected to a circuit.

GPIOs – Input behavior

- A GPIO input pin will have **fixed value of 1** if **power (high voltage) is connected** to the pin.
- A GPIO input pin will have **fixed value of 0** if **GND (ground) is connected** to the pin.
- A GPIO input pin will have **floating value between 0 and 1** if it is **not connected**.
  - We could set input pin to be constant 1 or 0 with the code below:

```
GPIO.setup(NUMBER, GPIO.IN, GPIO.PUD_UP) # constant 1
GPIO.setup(NUMBER, GPIO.IN, GPIO.PUD_DOWN) # constant 0
```

  - The value of such input pin will be overridden once a stronger force (high voltage for `GPIO.PUD_DOWN` or ground for `GPIO.PUD_UP`)

GPIOs – Input control (switch2.py)

- Now turn off the raspberry pi and add 1 LED to the bread board, connect long leg to a GPIO pin and short leg to the ground. Don’t forget the resistor!
- We will use the switch to send signal to our program, and once program receives the signal, we will turn on or off the LED by setting the GPIO output pin as 1 or 0.
- Once we press down the button, LED should be powered on; once we release the button, LED should be turned off

GPIOs – Input control meticulous (switch3.py)

- No need to change the Circuit, let's control the LED in a more elegant way.
- We now wish to press button once (has both the press down and release actions), the LED will be on.
- And another button press will turn the LED off.
- The idea is to use a state variable to monitor if the button is being pressed or not. And we enter different condition depending on the state variable's value.

Explorer more

- Now we have learned about GPIO input and output. The door has opened to us to build different circuits and different programs to control them.
- For example, you could try a 2 legs button!
- You could try a Buzzer. Use GPIO output to give it a high voltage so it will beep!
  - Buzzer usually has two legs (or wires), another leg should be connected to the ground.
  - Think how we can use Buzzer together with time.sleep(*) to play rhythm.
- But no matter which component you are using, always make sure you have resistor connected in the circuit so current running through the circuit would not fly too high.

Sensors

Lecture 06

Instructor: Link

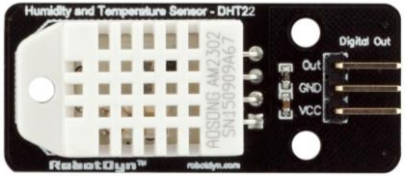
Course: CMPT 2200

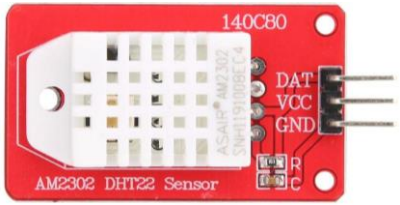
Learning outcomes

- Introduce DHT humidity and temperature sensor
- Introduce PIR movement sensor
- Introduce IR object sensor
- Introduce ultrasonic distance sensor

DHT humidity and temperature sensor

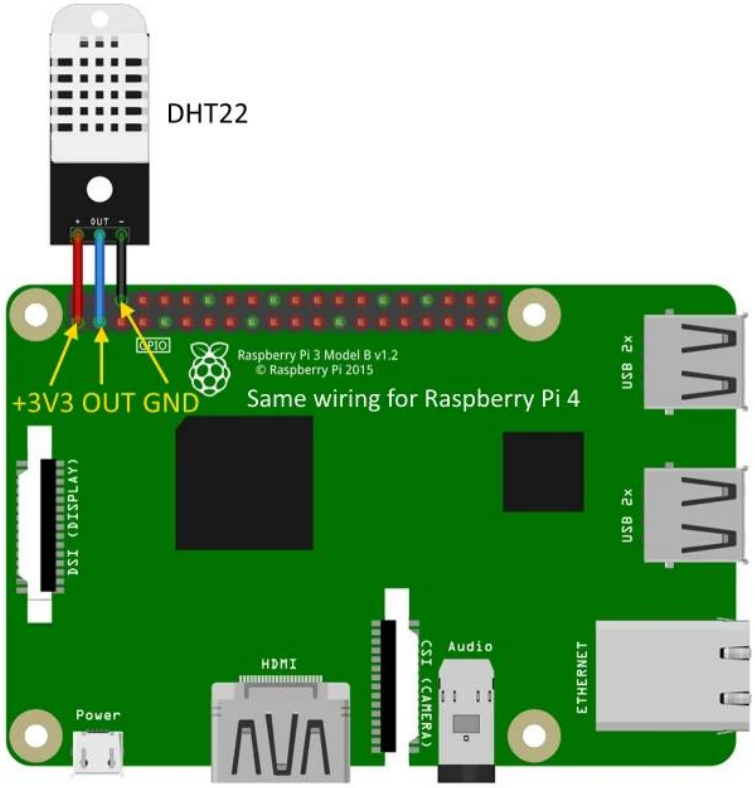
- A digital-output relative humidity and temperature sensor uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin.
- Typical models are DHT11 and DHT22.





DHT Circuit

- Connection to DHT sensor is simple.
- Connect VCC to the 3.3v power of Raspberry Pi
- Connect GND to the ground
- Connect DAT to one of the GPIO pin.



DHT program (DHT.py)

- It is complicated to use GPIO raw input to interpret the data from sensor.
- So DHT library is recommended to get the temperature
- It could directly get the number result

PIR movement sensor

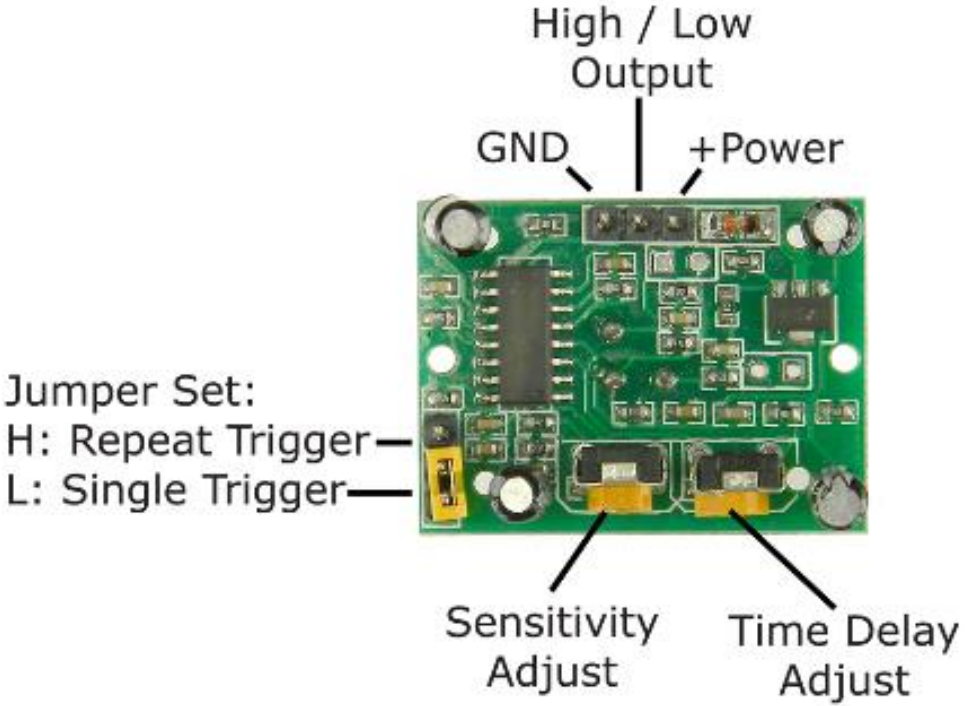
- We as humans are emitting some infrared light, which depends on the temperature of our body.
- The **passive infrared (PIR)** sensor measures infrared (IR) light radiating from objects in its field of view
- We can't see the infrared light with our eyes, but the sensor can detect it.
- The sensor will detect variations of that infrared light, which means, in other words, movement.

PIR movement sensor

- Real life applications
  - Automatic lighting
  - Security alarms
- Output is simple: it is binary output - Either there is a movement which has been detected or there is not.

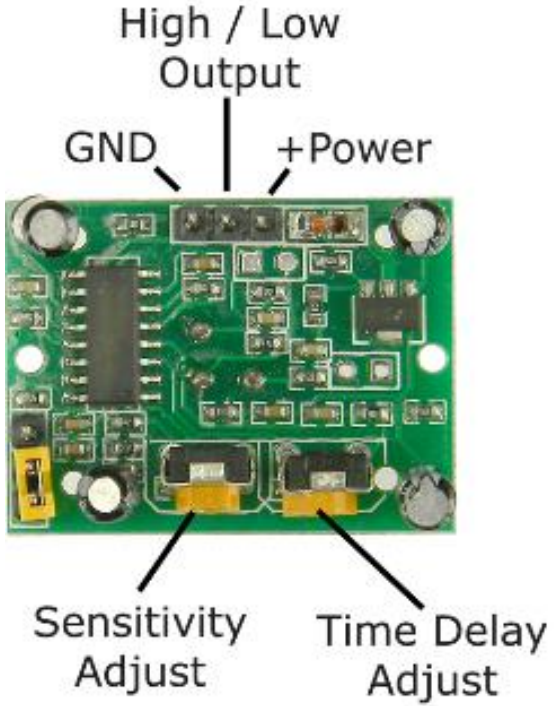
PIR movement sensor (Jumper Set)

- Jumper Set is used to set the mode of the sensor
  - If we set it to High (connect High pin and the middle pin with jumper), it is **Retrigger mode**. The output of sensor will remain HIGH as long as movement is being detected
  - If set to Low (**Normal mode**), every movement detected will result in a HIGH then LOW pulse output.
  - **Retrigger mode** is more useful most of the time.



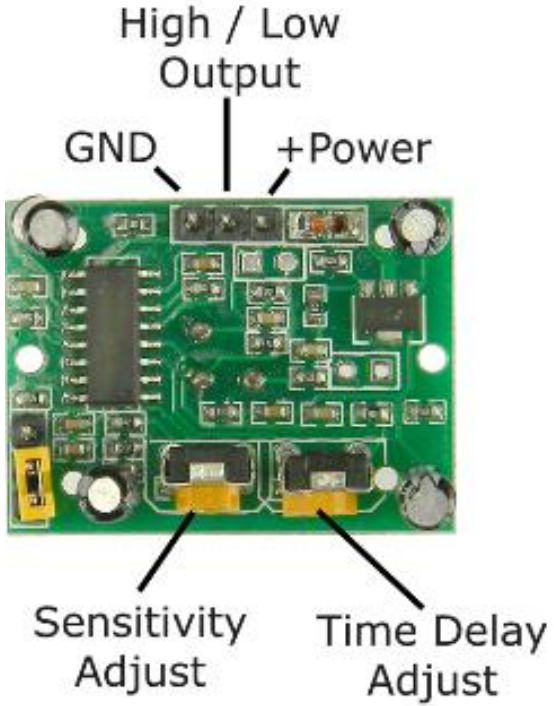
PIR movement sensor (Adjust)

- There are two adjustment on the sensor, we could use screwdriver to adjust.
- **Sensitivity Adjust** defines range of the sensor
  - if we turn it counterclockwise, range will drop to minimum of 3 meters
  - If we turn it clockwise, range will increase to maximum of 7 meters
- **Time Delay Adjust** defines how long output will remain HIGH when movement is detected
  - counterclockwise will make the time shorter down to 1 sec.
  - clockwise will make the time longer up to 5 min.



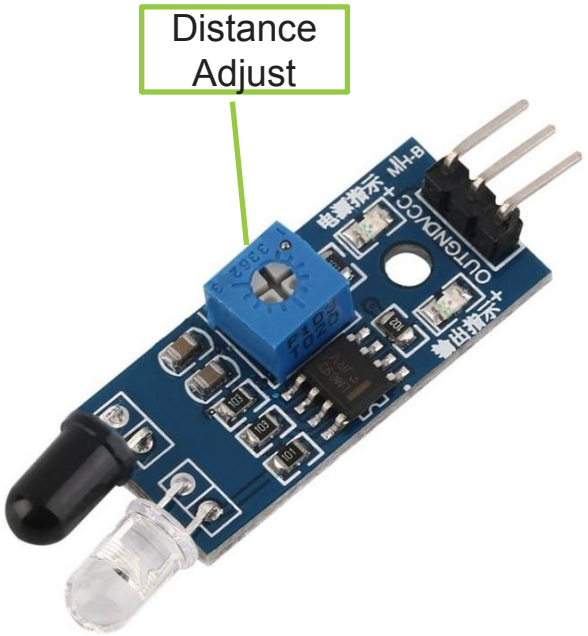
PIR movement sensor (sensor_move.py)

- We just connect the Power to 5V power, GND to ground, and Output to one GPIO pin (use a resistor between to protect GPIO).
- Now we are able to read PIR's data with Python.



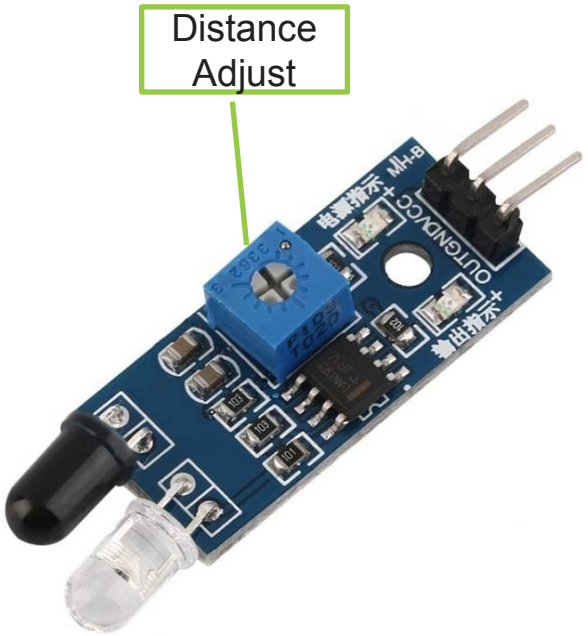
IR object sensor

- **Infrared (IR) sensor** has one IR LED which will emit Infrared light.
- It also has an IR photodiode which act as light detector. Photodiode is sensitive to IR light of the same wavelength which is emitted by the IR LED.
- So we could use IR sensor to detect object in front of it as the IR emitted will be reflected back by object and received by photodiode.
- It also has a potentiometer which is used to adjust the detection distance.
  - Turning it counterclockwise will make it only detect very close object (or can not detect at all)



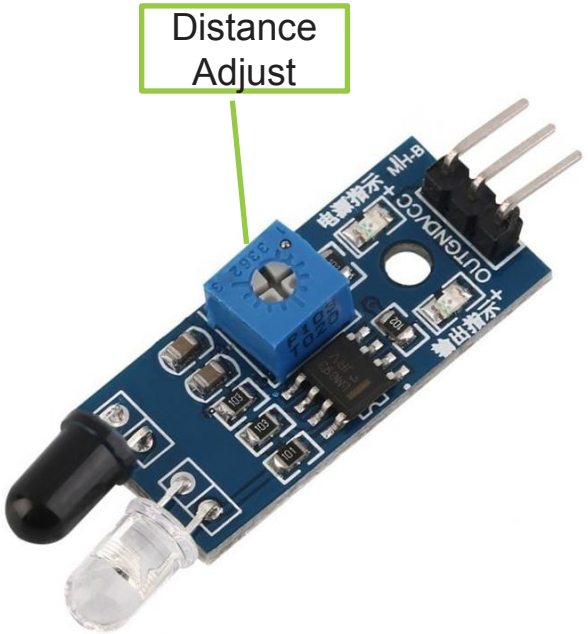
IR object sensor

- You will find two small LEDs on the sensor, one will turn on once connected to a circuit with power. Another will turn on once it detect something. We could use such feature for easy testing.
- As IR sensor is detecting emitted infrared light, if a surface could absorb most of the light (black color surface basically), then it could not detect anything (output will always be LOW).
- Following same logic, if we cast light upon the photodiode, then it considered item detected all the time (output will always be HIGH).



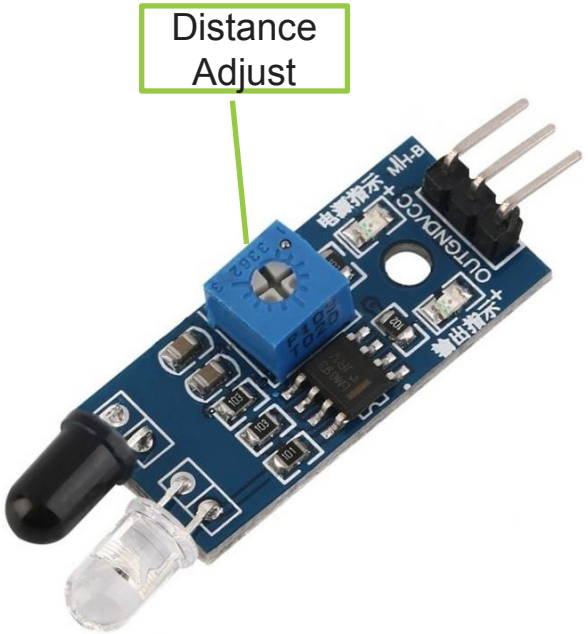
IR object sensor

- As a result, we could also use IR sensor to distinguish between a black and a white color object. (Useful in case like line tracing robot).
  - Mount the IR sensor about 1 to 2 inches above a black surface.
  - Turn the potentiometer fully clockwise.
  - Slowly turn the potentiometer counterclockwise until the signal LED just turns off.
  - Do not change the distance between the IR sensor and the black surface now.
  - The signal LED will turn on when the sensor is above a white surface and off when it is above a black surface.



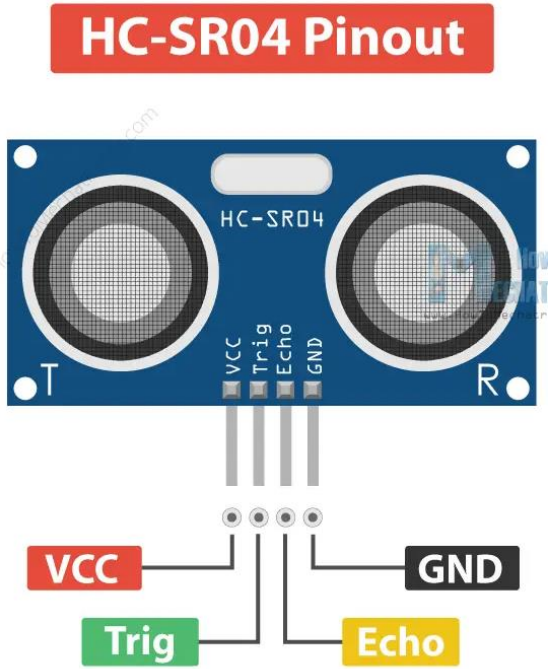
IR object sensor (sensor_object.py)

- Connect IR sensor to circuit is also very simple.
- We connect the Power to 5V power, GND to ground, and Output to one GPIO pin (use a resistor between to protect GPIO).
- Now we are able to read IR sensor's data with Python.



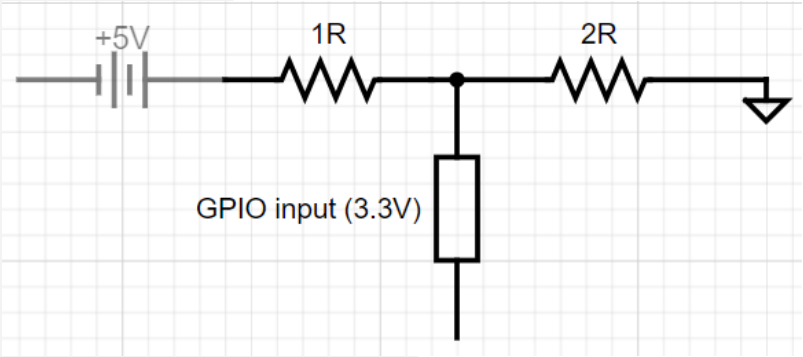
ultrasonic distance sensor (HC-SR04)

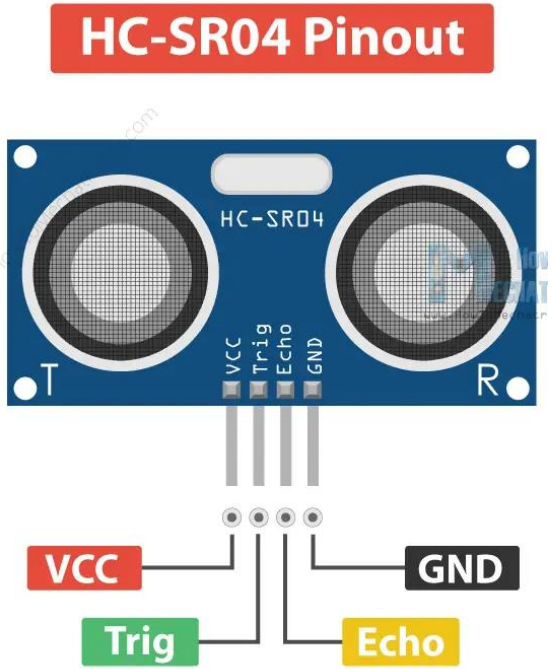
- An **Ultrasonic Sensor** is a sensor that send ultrasonic sound and hear it back. By calculating the sending time and hearing time, we could calculate the distance from the sensor to object in front up to 4 meters away.
- It could be used in robot car to avoid obstacles.
- It could also be used to simulate the parking alarm feature of car which beep more frequently or louder when car is closer to another object.



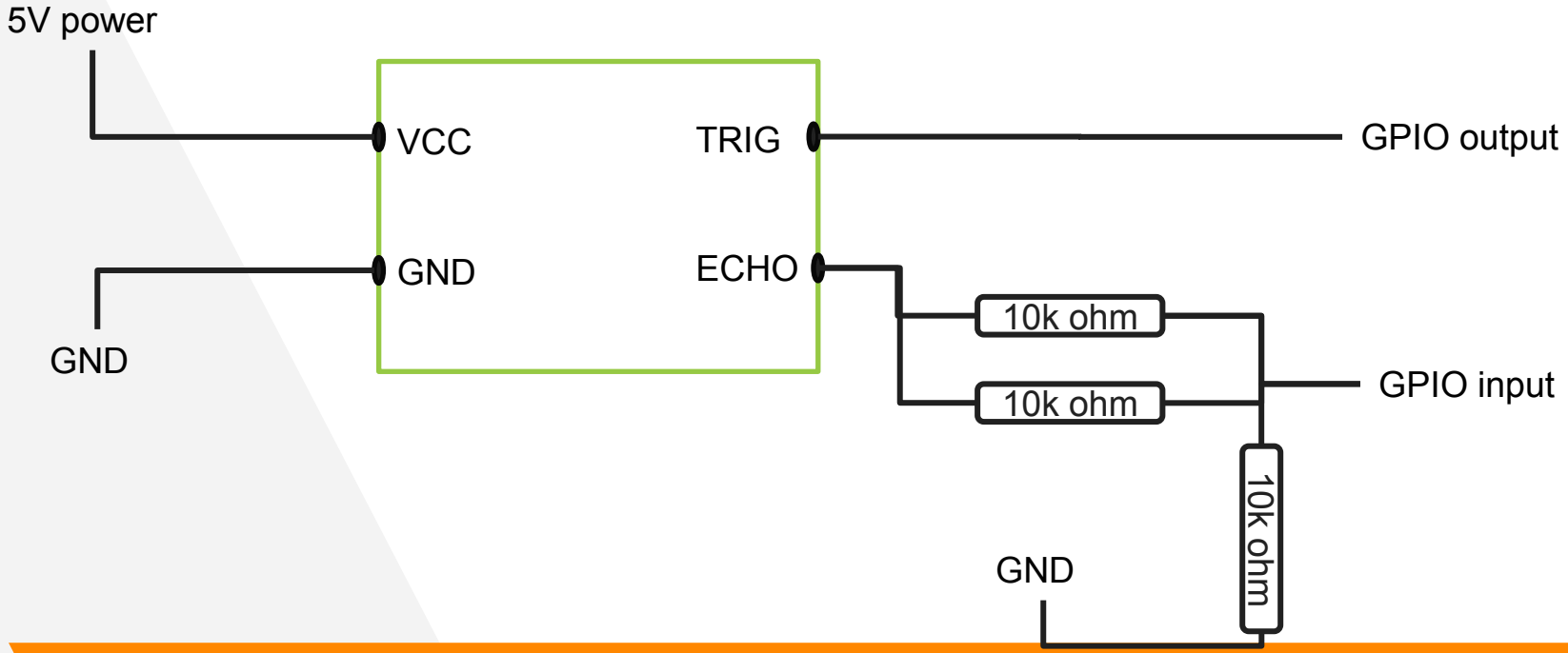
ultrasonic distance sensor (HC-SR04)

- VCC of ultrasonic sensor is 5V
- Output from Echo is also 5V.
  - So we can not directly connect it to GPIO pin (it could only accept no more than 3.3V).
  - But we could divide the voltage to make it work.



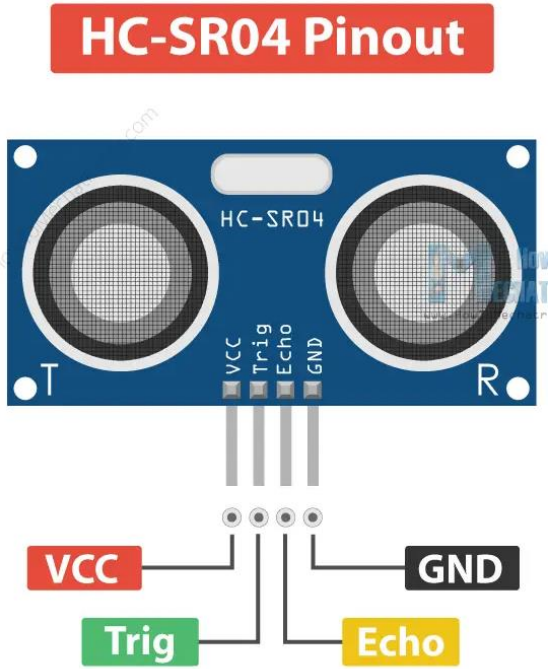


Circuit Diagram



ultrasonic distance sensor (sensor_distance.py)

- Now you are able to run the Python program.
- Read the comments carefully.



Pulse-Width Modulation

Lecture 07

Instructor: Link

Course: CMPT 2200

Learning outcomes

- Introduce Pulse-Width Modulation (PWM)
- Introduce how we use Raspberry Pi to send PWM signal
- Introduce DC motor and motor driver

Control the brightness of an LED

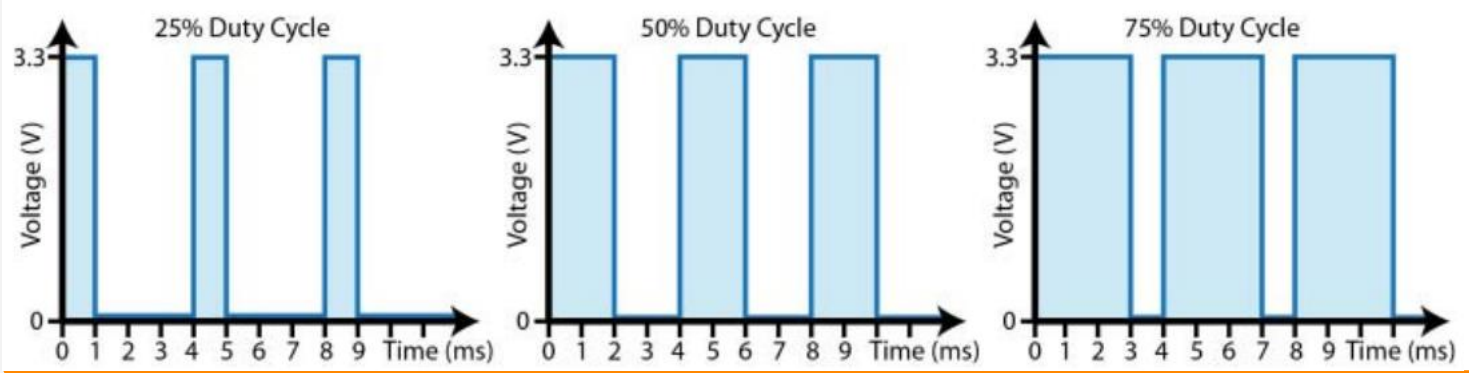
- When we were controlling LED with raspberry Pi, we never mentioned how to control its brightness.
- The first idea that we might come up with would be controlling it by changing the voltage going across the LED.
- However, as we mentioned, the voltage going cross LED would be considered as consistent. An LED is considered as a current-controlled device, where driving a current through the LED causes the forward voltage drop with a constant value. Therefore, trying to control an LED with a variable voltage will not work as you might expect.
- We could control the brightness by changing current, that is true. But the risk of damaging it is high, so we need to limit the current to a relative low value.
- Another idea is using pulsed current.

Control the brightness of an LED **using pulse signal**

- A pulse-width modulated (PWM) signal, is a current signal that rapidly switch between the state of High and Low.
- This could essentially rapidly switch the LED on and off.
- For example, if a rapid PWM signal is applied to the LED that is off for half of the time and on for half of the time, then the LED will appear to be only emitting about half of its regular operating condition light level.
- Our eyes don't see the individual changes if they are fast enough; they average over the light and dark interval to see a constant, but dimmer illumination.
- Also, our eyes would consider pulse-controlled LED to be brighter, even though the current stays the same.

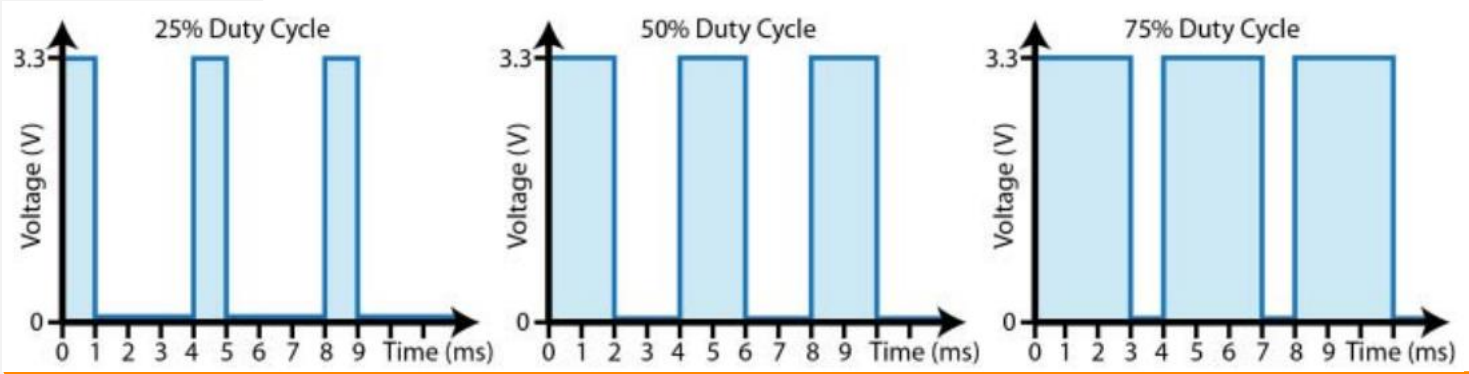
Pulse-Width Modulation (PWM)

- Below illustrates a PWM square wave signal at different duty cycles.
- The **duty cycle** is the percentage of time that the signal is high versus the time that the signal is low.
- In this example, a high is represented by a voltage of 3.3 V and a low by a voltage of 0 V. A duty cycle of 0% means that the signal is constantly low, and a duty cycle of 100% means that the signal is constantly high.



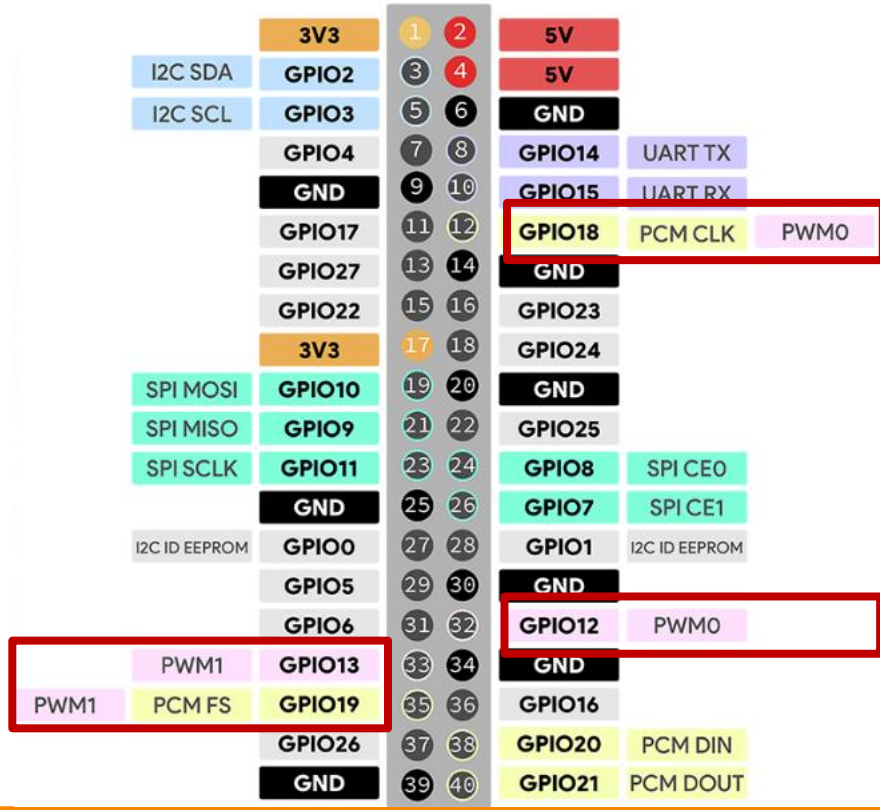
Pulse-Width Modulation (PWM)

- The **period (T)** of a repeating signal is the time it takes to complete a full cycle. In the example, the period of the signal in all three cases is 4 ms.
- The **frequency (f)** of a periodic signal describes how often a signal goes through a full cycle in a given time period.
- Therefore, for a signal with a period of 4 ms, it will cycle 250 times per second (1/0.004), which is 250 hertz (Hz).



software PWM and hardware PWM

- There are two ways of creating PWM signal in Raspberry Pi GPIO, Hardware and Software.
- **Hardware PWM** is produced by specific GPIOs as highlighted on the right.
  - The signal will be more stable, but it require other library or language to drive it (language C or PiGPIO library)
- **Software PWM** could be produced by any GPIOs, simulating pulse signal by turning it on or off. It will not be very stable.
  - This is what we will use and could be achieved by RPi.GPIO.



PWM in RPi.GPIO

- We first create a PWM instance using selected GPIO pin:
```
pwm = GPIO.PWM(PIN_NUM, frequency) # set initial frequency in Hz
```
- We could then start PWM by calling:
```
pwm.start(dc) # dc is the duty cycle with range [0,100]
```
- To change the frequency, we could call:
```
pwm.ChangeFrequency(f)   # f is the new frequency
```
- To change the frequency, we could call:
```
pwm.ChangeDutyCycle(d) # d is the new duty cycle
```
- To stop PWM, we just call:
```
pwm.stop()
```

Experience PWM with sound (buzzer_tune.py)

- Let's set up a very simple circuit.
- We are going to use sound to understand frequency and duty cycle.
- We connect 1 GPIO to 1 resistors, and another leg of resistor is connected to one Buzzer (+ side).
- Then we use wire to connect another side of Buzzer to the ground.
- Power on the Raspberry Pi, and run the program. You should enter the frequency and duty cycle in the command line.

use PWM to control LED (pwm_led.py)

- We have connected to LED many times, it is just the same this time.
- This program will continually change the duty cycle of signal, which simulates a breathing effect of LED light.
- Try modify the frequency and observe the differences.

DC motor

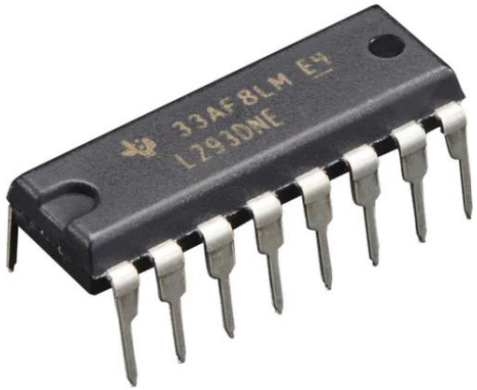
- DC motors are used in many applications, from toys to advanced robotics.
- They are ideal motors to use when continuous rotation is required, such as in the wheels of an electric vehicle. Typically, they have only two electrical terminals to which a voltage is applied. The speed of rotation and the direction of rota-tion can be controlled by varying this voltage.
- Most DC motors require more current than the RPi can supply, so we could use help of a motor driver and external power.

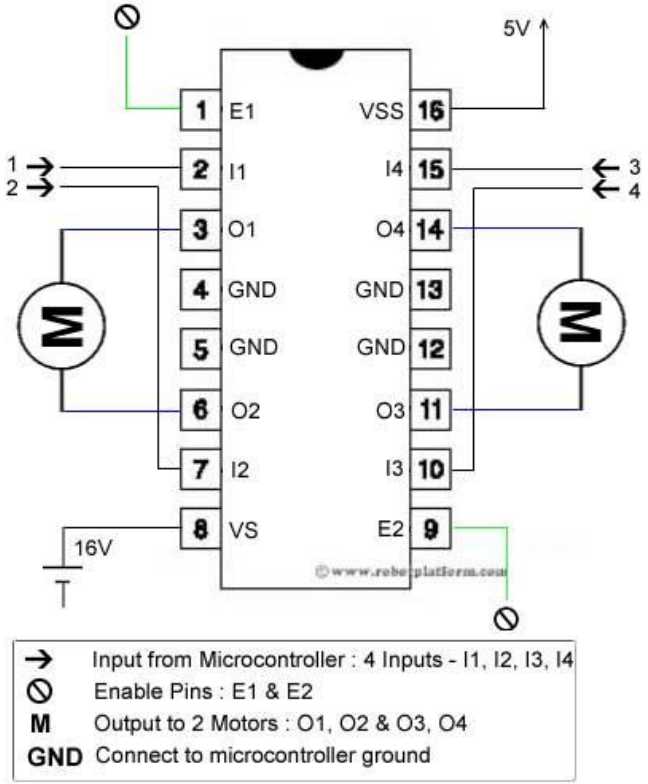
DC motor driver

- A **motor driver** transfer electrical energy from a source that could be of a given voltage, current at a certain frequency to an electrical output of desired voltage, current, frequency to the motor.
- It protects GPIO, it provides voltage and current higher than GPIO could produce, and it provides more control possibilities (like direction of the motor spinning).
- Usually, **we would need external power connected to driver.**

DC motor driver and diagram

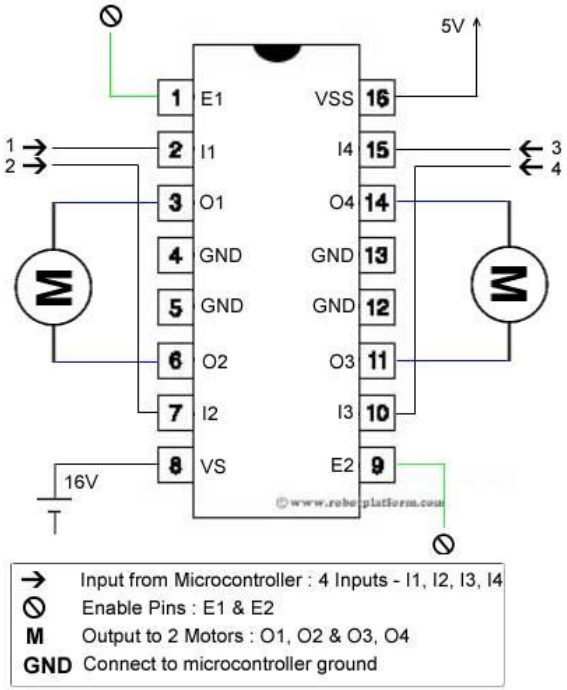
- A **motor driver or motor controller** is a microcontroller or processor we used to control a motor.
- We introduce **L293D** today (also because we have many of these).
- Right is the circuit diagram for L293D motor driver.





DC motor driver

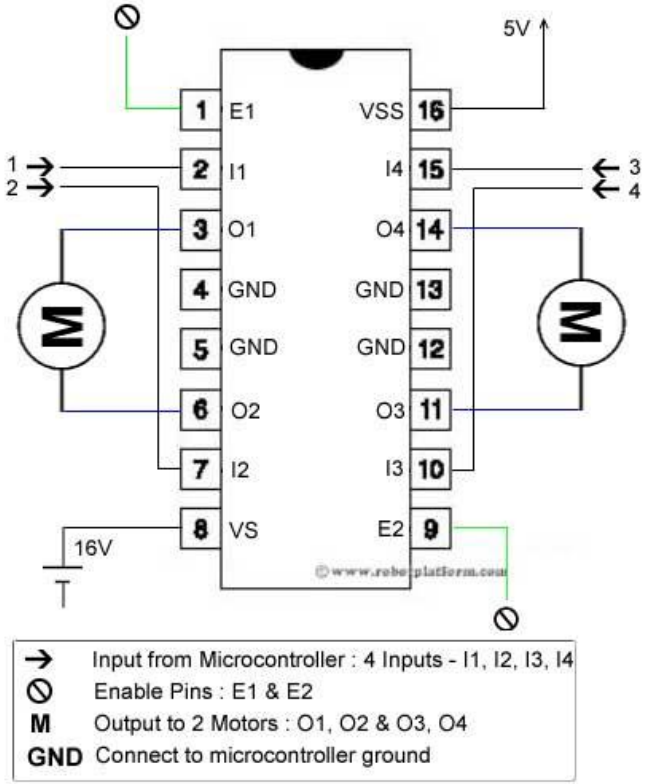
- We could control 2 motors with one driver.
- Below is the table of PIN signals and corresponding behavior.
- Use GPIO to control these pins.



Pin 1	Pin 2	Pin 7	Function
High	High	Low	Turn counter-clockwise (Reverse)
High	Low	High	Turn clockwise (Forward)
High	High	High	Stop
High	Low	Low	Stop
Low	X	X	Stop

Motor and PWM

- PWM can be used to control the light level of LEDs, but it can also be used to control the speed of DC motors
- Typically, the frequency is in the kHz range for motor control. (I would say 5kHz to 20kHz is a safe range).
- Then we could just change the duty cycle to control its speed.



Motor

Lecture 08

Instructor: Link

Course: CMPT 2200

Learning outcomes

- Review of DC motor and motor driver
- Learn how to control Motor using motor driver in our lab

DC motor

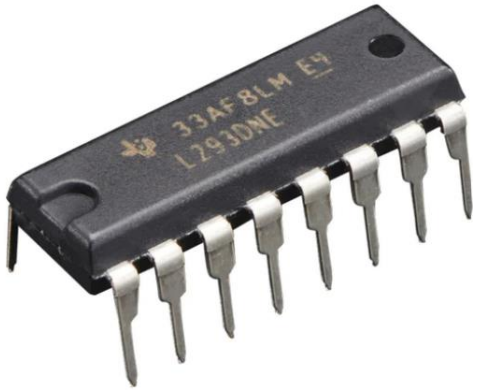
- DC motors are used in many applications, from toys to advanced robotics.
- They are ideal motors to use when continuous rotation is required, such as in the wheels of an electric vehicle. Typically, they have only two electrical terminals to which a voltage is applied. The speed of rotation and the direction of rotation can be controlled by varying this voltage.
- Most DC motors require more current than the RPi can supply, so we could use help of a motor driver and external power.

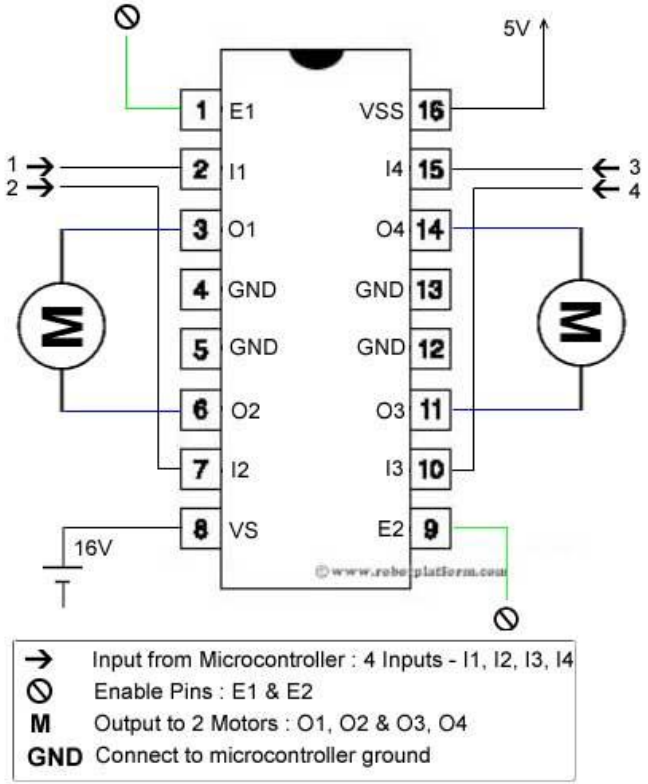
DC motor driver

- A **motor driver** transfer electrical energy from a source that could be of a given voltage, current at a certain frequency to an electrical output of desired voltage, current, frequency to the motor.
- It protects GPIO, it provides voltage and current higher than GPIO could produce, and it provides more control possibilities (like direction of the motor spinning).
- Usually, **we would need external power connected to driver.**

DC motor driver and diagram

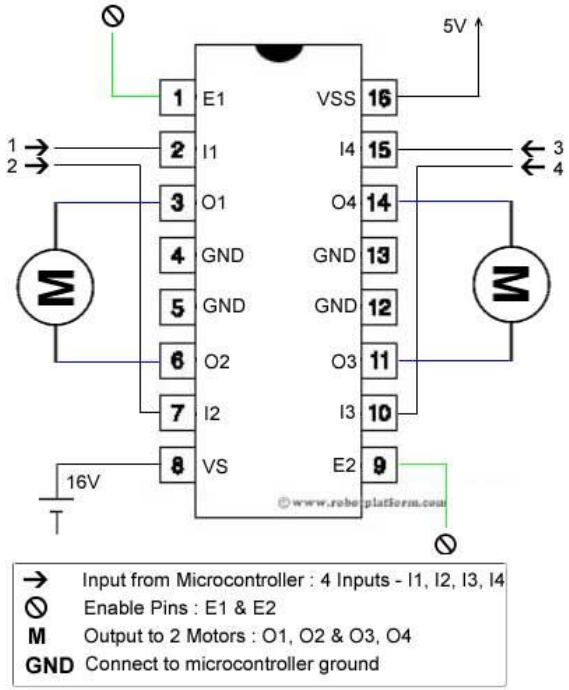
- A **motor driver or motor controller** is a microcontroller or processor we used to control a motor.
- We introduce **L293D** today (also because we have many of these).
- Right is the circuit diagram for L293D motor driver.





DC motor driver

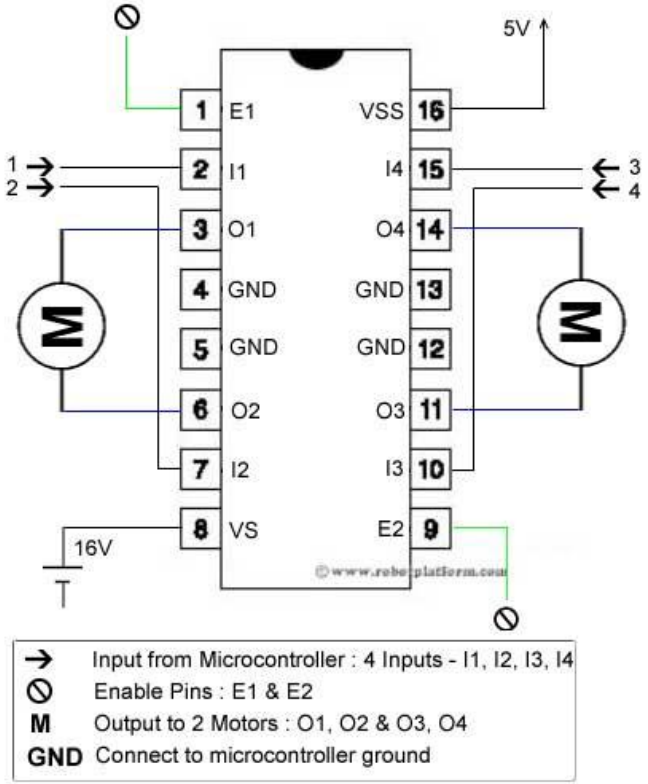
- We could control 2 motors with one driver.
- Below is the table of PIN signals and corresponding behavior.
- Use GPIO to control these pins.



Pin 1	Pin 2	Pin 7	Function
High	High	Low	Turn counter-clockwise (Reverse)
High	Low	High	Turn clockwise (Forward)
High	High	High	Stop
High	Low	Low	Stop
Low	X	X	Stop

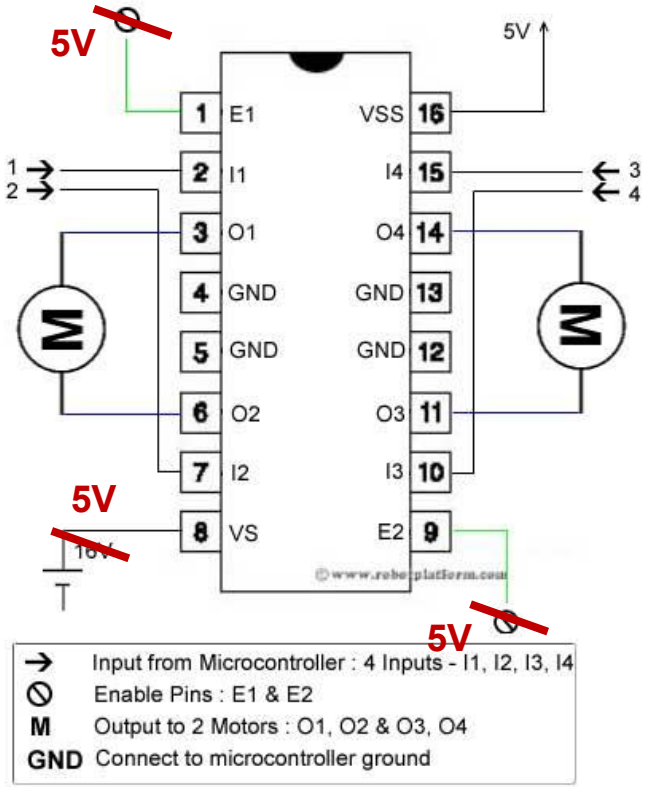
Motor and PWM

- PWM can be used to control the light level of LEDs, but it can also be used to control the speed of DC motors
- Typically, the frequency is in the kHz range for motor control. (I would say 5kHz to 20kHz is a safe range).
- Then we could just change the duty cycle to control its speed.



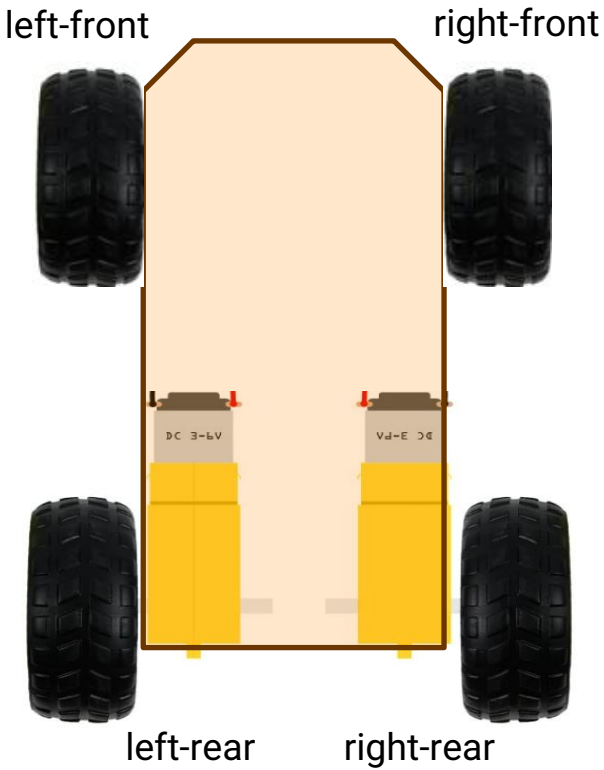
Motor in our lab

- In our lab, we don't have enough power source to drive the motor driver in its full functionality.
- In such case, we could just use 5V connected to pin 8.
- But because power voltage is not high enough, we need consistent power connecting to our Enable Pins.
  - Directly connect E1 and E2 to the 5V power supply (or 3.3V if you don't need large torque)
  - We can not use E1 and E2 to control motor driver anymore. Instead, use I1/I2 and I3/I4 to turn the motor on or off.



Motor to drive car

- A very popular way (but not the only way) to control a motor driving car is as shown on the right.
  - You could choose front-wheel drive or rear-wheel drive. Our example is choosing rear-wheel drive (which means we connect motor to the rear tires).
  - To move forward on straight line, both motors need to run with same speed clockwise (counter-clockwise if we wish to move backward).
  - To turn left when moving, left motor should spin slower than the right motor. Same logic for turning right.
  - To make a left U-turn , left motor should stop and right motor should spin. Same logic for right U-turn.



Email, Camera, File

Lecture 09

Instructor: Link

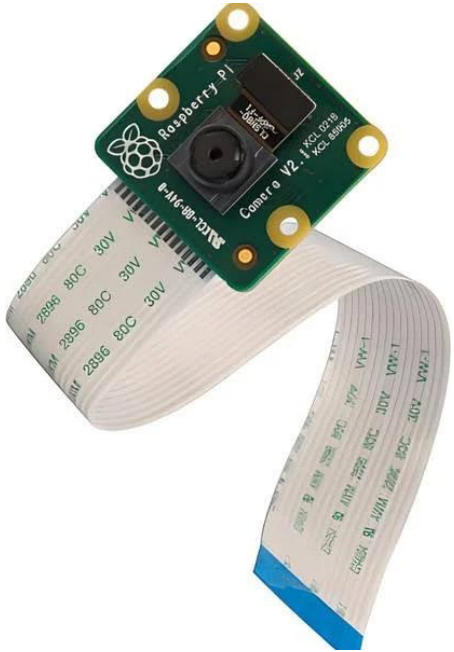
Course: CMPT 2200

Learning outcomes

- Using Raspberry Pi Camera V2 Module to take photos and videos
- Send an email from Raspberry Pi using python program
- Create and manipulate file from Terminal or using Python

Raspberry Pi Camera V2 Module

- Raspberry Pi provides a camera module which could easily extract photos and videos.
- With camera, you are able to create much more smart and impressive applications (like home security system).

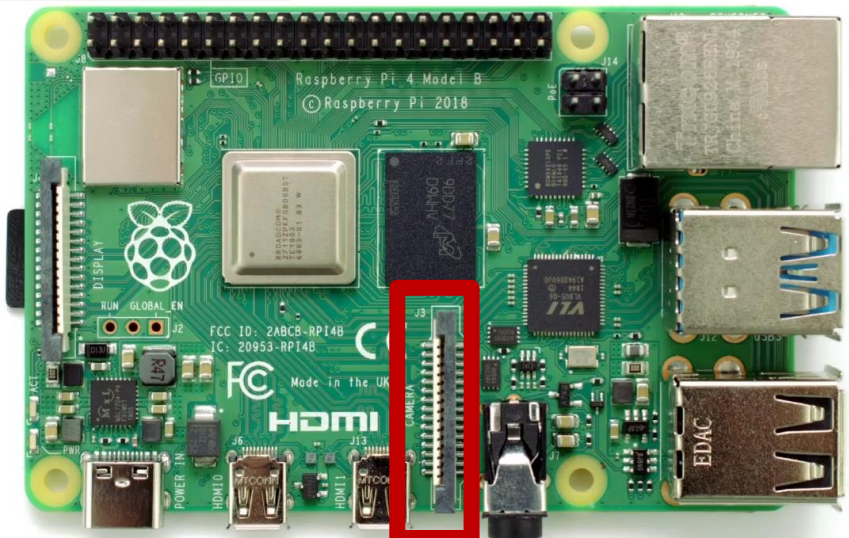


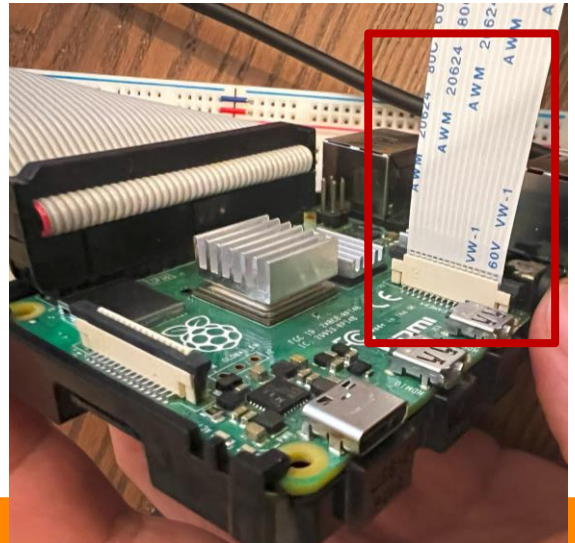
Work with the Pi Camera

- There are 2 available types of Pi camera:
  - The standard camera, which is mounted on a green PC board is for general use – It gives nice quality images/videos when the light of the room is good enough.
  - The NoIR camera (black) which means no infrared is mounted on the black PC board – it works great in low light environments so you can use it in a completely dark room or at night.
    - However, the images will not look as good as the standard one in daylight.

Plug Camera to Raspberry Pi

- To recap, we will put the wire to the Raspberry Pi's camera port.
- Pull out the plastic gear a bit from the port (not completely), then plug wire to the port.
- The blue side of wire should be facing the USB ports.
- When it is fully plugged in, you could push back the black stuff to stuck the wire.





Take a Photo From the Terminal

- We are going to use **rpicam** applications from Terminal to start working on camera.
- In the recent version of Raspberry Pi OS, the **rpicam** apps are already installed.
- We could start the preview window of our camera for 5 seconds with command:
  rpicam-hello
- We can change how long it will last by change –t option values:
  rpicam-hello -t 1000 will open the window for 1 second (1000 milliseconds)
- Now let's take a photo after 5 seconds using rpicam-still command, we also need option –o with file name to take image as file: **rpicam-still -o test1.jpg**
- There are many options we could use, below command:
  rpicam-still -o test2.jpg --width 1920 -height 1080 -shutter 20000 -t 2000
  waits for 2 seconds, then takes photo with size of 1920x1080, exposure time of 20ms

Take Video From the Terminal

- We use **rpicam-vid** command to take video:

```
rpicam-vid -t 5000 -o video1.h264
```

- take videos of 5 seconds and stores it in the file **video1.h264**.
- **--width**, **--height** options also work for **rpicam-vid**.

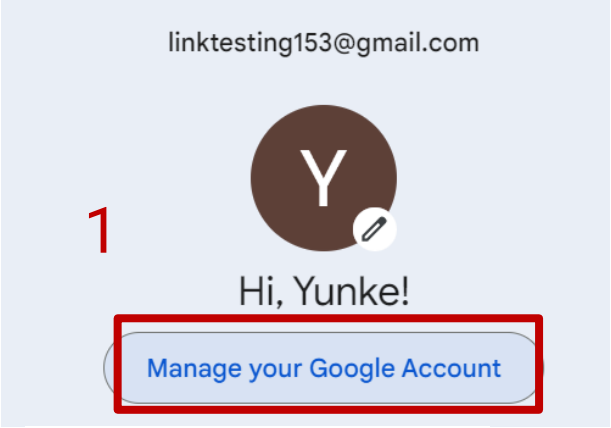
Take Photo or Video with Python

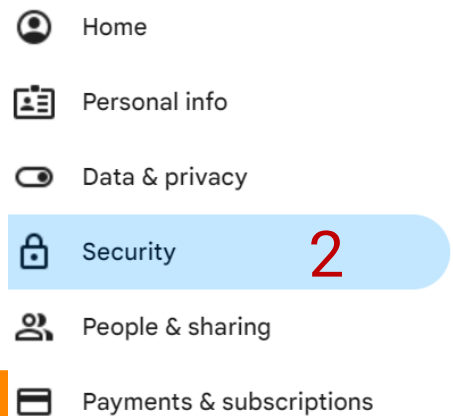
- Picamera2 is a Python library that gives convenient access to the camera system of Raspberry Pi.
- Check the camera1.py (take photo) and camera2.py(take video) and test the functionality.
- Picamera2 is much more powerful than what we show in the sample code. Visit the official documentation: https://datasheets.raspberrypi.com/camera/picamera2-manual.pdf for more information.

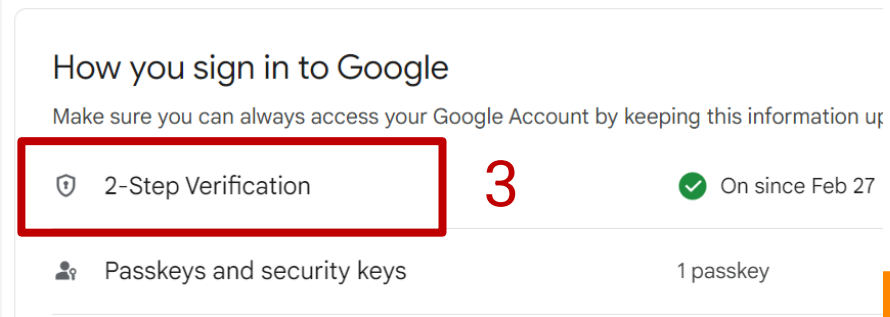
Create Gmail account for Raspberry Pi usage

Creating a new Gmail account won't be an issue, but we need to change some settings to bypass the security checking.

1. First, click Manage your Google Account button for the account we just created.
2. Go to the security tab on the left.
3. Turn on the 2-Step Verification

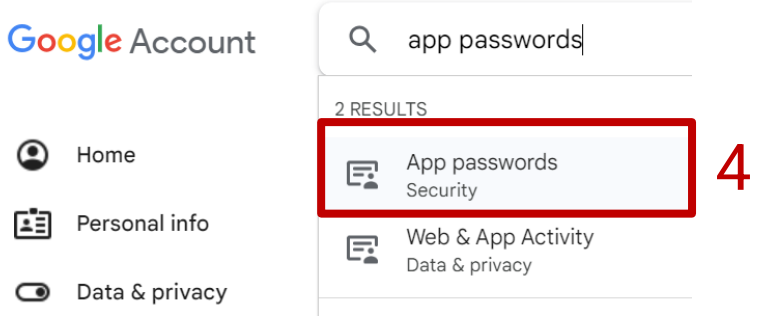


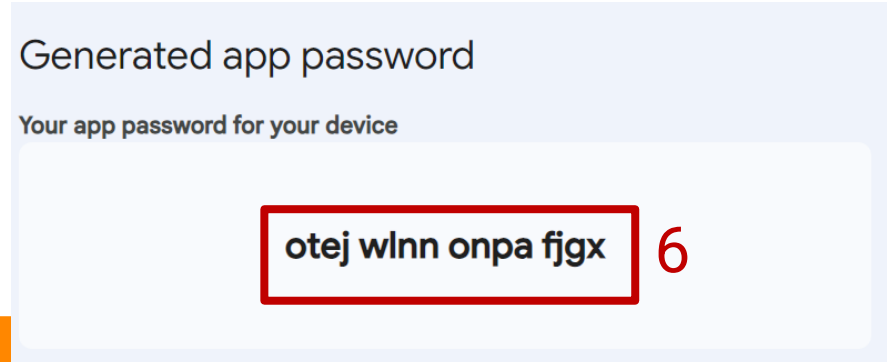




Create Gmail account for Raspberry Pi usage

4. Search for App passwords from the search bar.
5. Create a new app password, choose any name you like.
6. In the end, record down the passcode, that would be the password you should use in Python program of Raspberry Pi.





Install yagmail with Terminal

- Yagmail module is used to sending email from our Python program.
- Open the terminal on Raspberry Pi, and install the yagmail package with the command below:

```
sudo pip3 install yagmail --break-system-package
```

- Now after installation, in Terminal, type **`python`** to enter the python script mode, and type **`import yagmail`**.
  - If installation succussed, you won't receive any error message.

Create a hidden password file

- It might not be a wise idea to directly put password in your python program.
- A better idea would be putting the password in a separate file (better if it is hidden).
- We have learned how to create file in Linux. Here we will create a file starts with **dot**(**.**)and save the password in it. File starts with **dot**(**.**) would be considered as hidden file -- you would not be able to see it by normal ways. But python program could visit it.
  - **touch** command would be able to create a file: **touch .email_pwd**
  - The normal **ls** command can not see the file, but we could add **-la** option to see files including hidden ones: **ls -la**
  - **nano** command could edit the file using the terminal nano editor even file is hidden: **nano .email_pwd**

Sending email with Raspberry Pi

- Now testing if you could send email to from Raspberry Pi to your daily use email address.
  - mail1.py sends a simply email from email account you newly created
  - mail2.py creates a new file and send it as attachment in the email

Gates and Circuits

Lecture 10

Instructor: Link

Course: CMPT 2200

Learning outcomes

- Identify the basic gates and describe the behavior of each
- Combine basic gates into circuits.
- Describe the behavior of a gate or circuit using truth tables, and logic diagrams.

Computers and Electricity

- Computers are electronic devices; the most fundamental hardware elements of a computer control the flow of electricity. In a very primitive sense, we use technology to control such circuit, so that we can perform calculations and make decisions.
- Any given electronic signal has a level of voltage. As we mentioned many times, we distinguish between the two values of interest (binary 0 and 1) by the voltage level of a signal.
- In general, a voltage level around 0 (or ground) is considered “**low**” and is interpreted as a binary **0**.
- A signal significantly higher than 0 is considered "**high**" and interpreted as a binary **1**.

Logic Gate

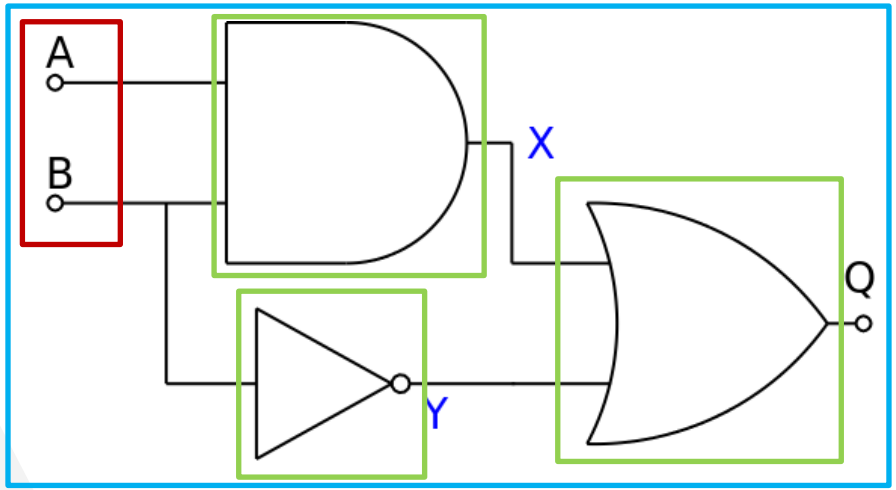
- A **logic gate** is a device that performs a basic operation on electrical signals.
  - It accepts one or more input signals and produces a single output signal.
  - Several types of gates exist; we examine the 3 most fundamental types. Each type of gate performs a particular logical function.
- Gates are combined into **circuits** to perform more complicated tasks.
  - For example, circuits can be designed to perform arithmetic and to store values.
  - In a circuit, the output value of one gate often serves as the input value for one or more other gates.
  - The flow of electricity through a circuit is controlled by the carefully designed logic of the interacting gates.

Methodology to describe logic circuit

- Three different, but equally powerful, notational methods are used to describe the behavior of gates and circuits:
  - Boolean expressions
  - Logic diagrams
  - Truth tables
- We will focus on the latter two types of representation during our discussion of gates and circuits.

Definition of Logic Diagram

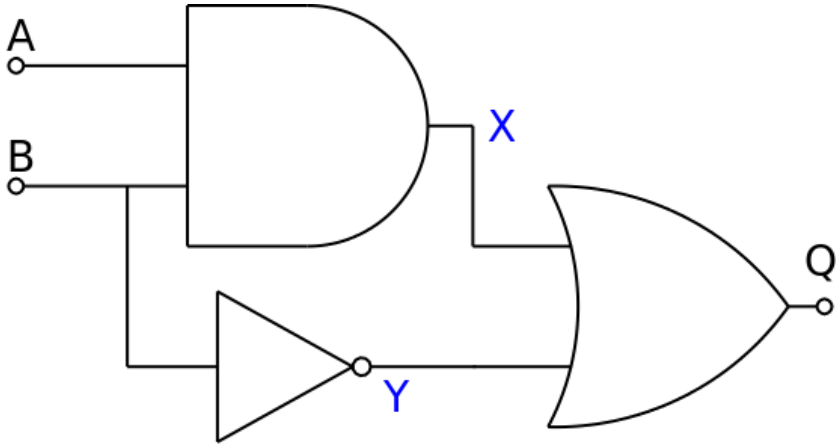
- A **logic diagram** is a graphical representation of a circuit.
- Each type of gate is represented by a specific graphical symbol.
- By connecting those symbols in various ways, we can visually represent the logic of an entire circuit.



A	B	X	Y	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

Definition of Truth Table

- A truth table defines the function of a gate by listing all possible input combinations that the gate could encounter, along with corresponding output.
- We can design more complex truth tables with sufficient rows and columns to show how entire circuits perform for any set of input values.



A	B	X	Y	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

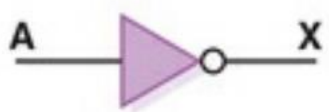
Major Three Logic Gates

- Each logic gate in a computer perform just one logical function. That is, each gate accepts **one or more input** values and produces a **single output** value.
- Because we are dealing with binary information, each input and output value is either 0, corresponding to a low-voltage signal, or 1, corresponding to a high voltage signal. The type of gate and the input values determine the output value.
- Let's examine the processing of the following three types of gates. We then show how they can be combined into circuits to perform arithmetic operations:
  - NOT
  - AND
  - OR

NOT Gate

- A **NOT gate** accepts one input value and produces one output value as shown in the figure below represented in two ways: as its logical diagram symbol, and through a truth table.
- In each representation, the variable A represents the input signal, which is either 0 or 1. The variable X represents the output signal, whose value (0 or 1) is determined by the value of A.
- By definition, if the input value for a NOT gate is 0, the output value is 1; if the input value is 1, the output is 0. A NOT gate is sometimes referred to as an inverter because it inverts the input value.

Logic Diagram Symbol

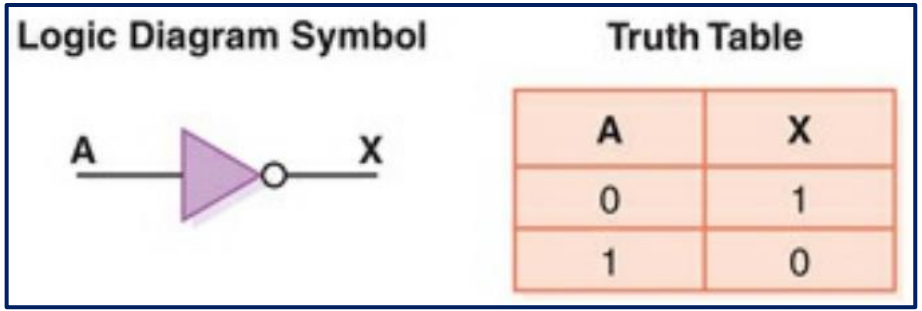


Truth Table

A	X
0	1
1	0

NOT Gate

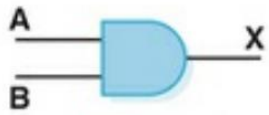
- The logic diagram symbol for a NOT gate is a triangle with a small circle (called an inversion bubble) on the end. The input and output are shown as lines flowing into and out of the gate. Sometimes these lines are labeled, though not always.
- The truth table shows all possible input values for a NOT gate as well as the corresponding output values. Because there is only one input signal to a NOT gate, and that signal can be only a 0 or a 1, there are only two possibilities for the column labeled A in the truth table. The column labeled X shows the output of the gate, which is the inverse of the input.



AND Gate

- Figure below defines an **AND gate**.
- Unlike a NOT gate, which accepts one input signal, an AND gate **accepts two input signals**. The values of both input signals determine what the output signal will be. **If the two input values for an AND gate are both 1, the output is 1; otherwise, the output is 0.**

Logic Diagram Symbol



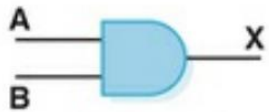
Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Gate

- The truth table showing the behavior of the AND gate has four rows, showing all four possible input combinations.
- Because there are two inputs and two possible values for each input, four possible combinations of 1 and 0 can be provided as input to an AND gate. Therefore, four situations can occur:
  - 0 · 0 equals 0
  - 0 · 1 equals 0
  - 1 · 0 equals 0
  - 1 · 1 equals 1

Logic Diagram Symbol



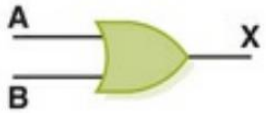
Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

- Figure below defines an OR gate.
- Like the AND gate, the OR gate has two inputs. If both input values are 0, the output value is 0; otherwise, the output is 1.
- The OR gate has two inputs, each of which can be one of two values. Thus, as with an AND gate, there are four input combinations and therefore four rows in the truth table:
  - 0 + 0 equals 0
  - 0 + 1 equals 1
  - 1 + 0 equals 1
  - 1 + 1 equals 1

Logic Diagram Symbol



Truth Table

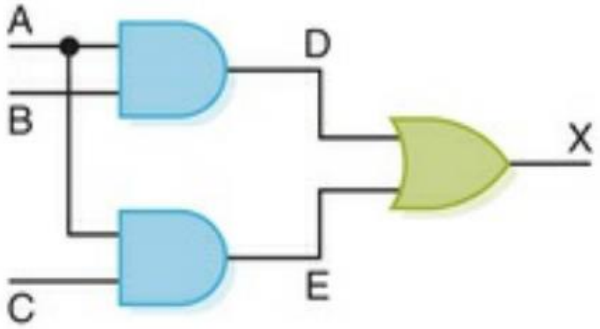
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Combinational Circuits

- Now that we know how individual gates work and how they are constructed, let's examine how we combine gates to form circuits.
- Circuits can be classified into two general categories. In a **combinational circuit**, the input values explicitly determine the output. In a sequential circuit, the output is a function of the input values as well as the existing state of the circuit. The circuits we examine in our lecture are combinational circuits.
- We mentioned before, as with gates, we can describe the operations of entire circuits using logic diagrams, and truth tables.

Combinational Circuits

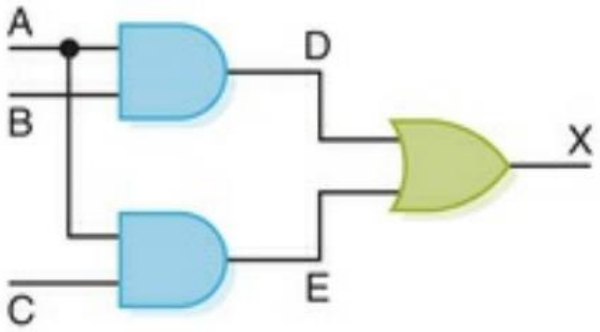
- Gates are combined into circuits by using the output of one gate as the input for another gate. For example, consider the following logic diagram of a circuit:



- The output of the two AND gates is used as the input to the OR gate. The input value A is used as input to both AND gates. The dot indicates that two lines are connected. If the intersection of two crossing lines does not have a dot, you should think of one as “jumping over” the other without affecting each other.

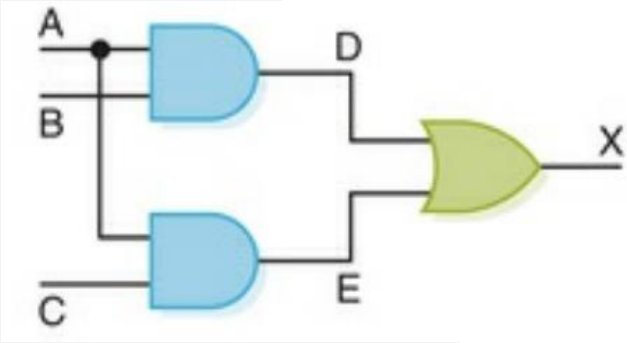
Combinational Circuits

- What does this logic diagram mean? Well, let's work backward to see what it takes to get a particular result.
- For the final output X to be 1, either D must be 1 or E must be 1. For D to be 1, A and B must both be 1. For E to be 1, both A and C must be 1. Both E and D may be 1, but that isn't necessary.
- Examine this circuit diagram carefully; make sure that this reasoning is consistent with your understanding of the types of gates used.



Combinational Circuits

- Now let's represent the processing of this entire circuit using a truth table.
- Because there are three inputs to this circuit, eight rows are required to describe all possible input combinations. Intermediate columns show the intermediate values (D and E) in the circuit.



A	B	C	D	E	X
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

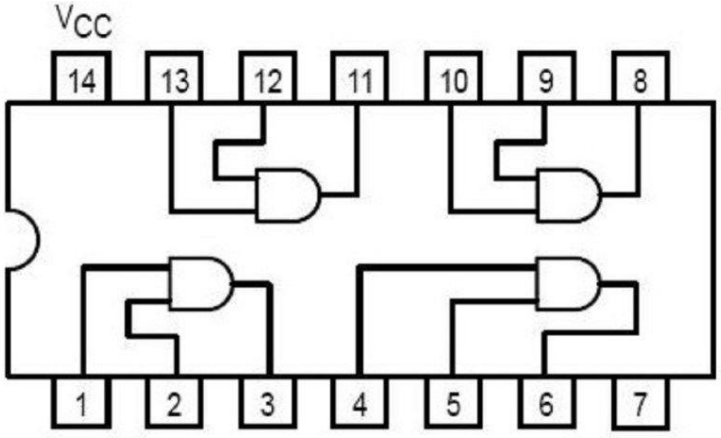
Integrated Circuits

- An integrated circuit (also called a chip) is a piece of silicon on which multiple gates have been embedded. These silicon pieces are mounted on a plastic or ceramic package with pins along the edges that can be soldered onto circuit boards or inserted into appropriate sockets. Each pin connects to the input or output of a gate, or to power or ground.
- Integrated circuits (IC) are classified by the number of gates contained in them. These classifications also reflect the historical development of IC technology:

Abbreviation	Name	Number of Gates
SSI	Small-scale integration	1 to 10
MSI	Medium-scale integration	10 to 100
LSI	Large-scale integration	100 to 100,000
VLSI	Very-large-scale integration	more than 100,000

Integrated Circuits

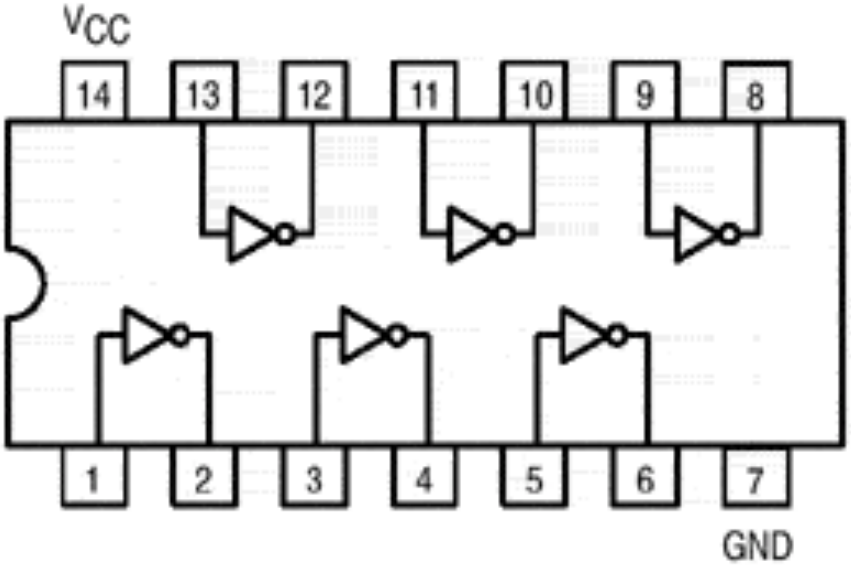
- An SSI chip has a few independent gates, such as the one shown in figure below.
- This chip has 14 pins: eight for inputs to gates, four for output of the gates, one for ground, and one for power. Similar chips can be made with different gates. Here is IC 7408 with 4 AND gates.



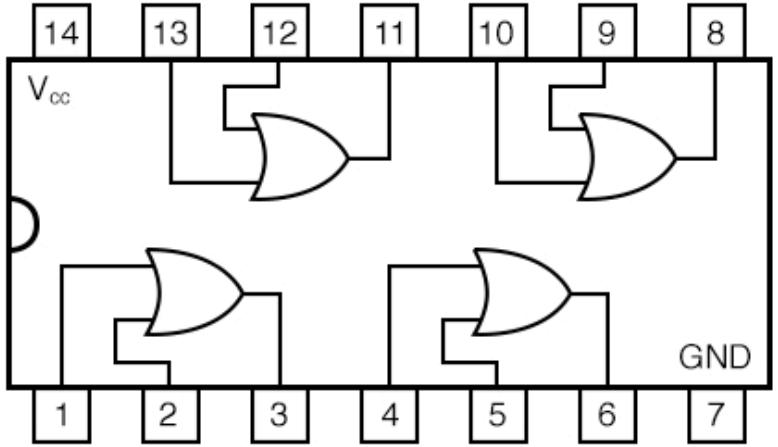
IC 7408 Pinout

Integrated Circuits

- Below are IC 7404 and IC 7432



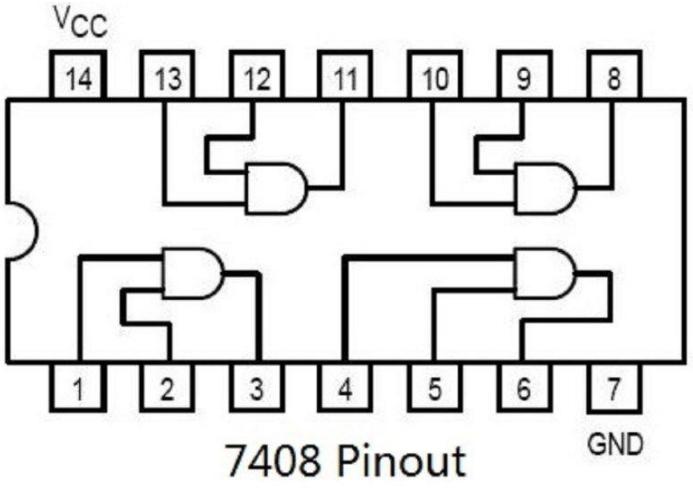
IC 7404 Pinout



IC 7432 Pinout

Integrated Circuits

- An SSI chip has a few independent gates, such as the one shown in figure below.
- This chip has 14 pins: eight for inputs to gates, four for output of the gates, one for ground, and one for power. Similar chips can be made with different gates.



7408 Pinout

Good to know: Integrated Circuits

- How can a chip have more than 100,000 gates on it? - That would imply the need for 300,000 pins!
- The key is that the gates on a VLSI chip are not independent, as they are in small-scale integration. VLSI chips embed circuits with a high gate-to-pin ratio. That is, many gates are combined to create complex circuits that require only a few input and output values.
- Each CPU chip contains a large number of pins through which essentially all communication in a computer system occurs. This communication connects the CPU to memory and I/O devices, which are themselves, at fundamental levels, advanced circuits.