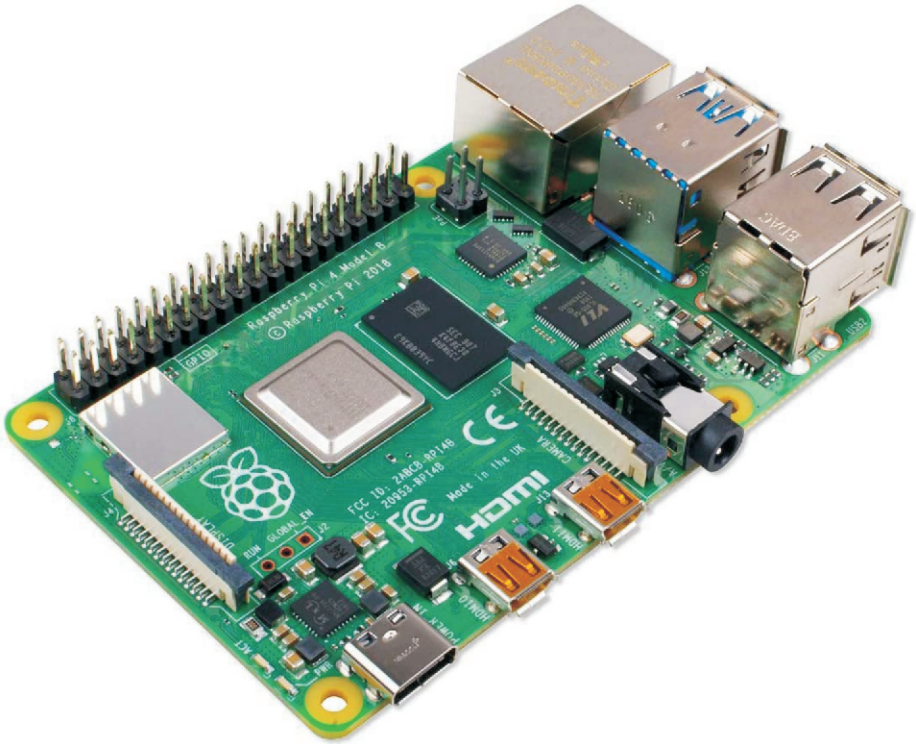
EDKITS ELECTRONICS

Raspberry Pi Guide for CS



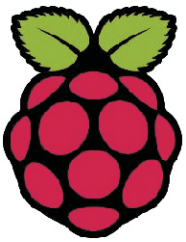


Table of Contents

Raspberry Pi Hardware Preparation and Installation

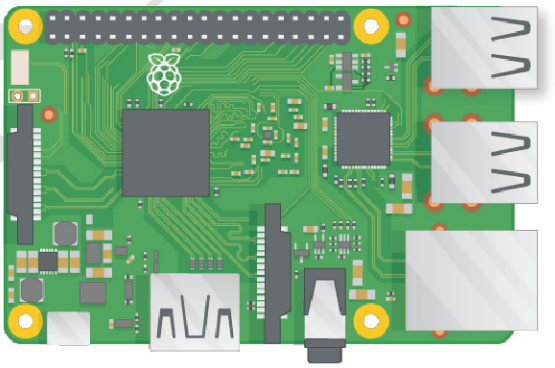
Hardware Guide:

For getting started with raspberry pi for the first time you will require the following hardware

1. Raspberry Pi (latest Model)
2. Monitor or TV
3. HDMI cable
4. Ethernet cable
5. USB keyboard
6. USB mouse
7. Micro USB power supply
8. 8GB or larger microSD card
9. SD Card Reader

Raspberry Pi 3 Model B:

The Raspberry Pi 3 is the third generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016. Compared to the Raspberry Pi 2 it has:



A 1.2GHz 64-bit quad-core ARMv8 CPU
802.11n Wireless LAN
Bluetooth 4.1
Bluetooth Low Energy (BLE)

Like the Pi 2, it also has:

4 USB ports
40 GPIO pins
Full HDMI port
Ethernet port
Combined 3.5mm audio jack and composite video
Camera interface (CSI)
Display interface (DSI)
Micro SD card slot (now push-pull rather than push-push)
Video Core IV 3D graphics core
The Raspberry Pi 3 has an identical form factor to the previous Pi 2 (and Pi 1 Model B+) and has complete compatibility with Raspberry Pi 1 and 2.

Monitor or TV:

A monitor or TV with HDMI in can be used as a display with a Raspberry Pi. Most modern television sets and monitors have an HDMI port, and are the easiest to get working with the

Raspberry Pi. You can use an HDMI cable to connect the Raspberry Pi directly to the television or monitor.

Some older monitors have a DVI port. These work well with the Raspberry Pi, although you'll need an HDMI-to-DVI adapter to attach to an HDMI cable, or a one-piece HDMI-to-DVI cable. Some old monitors have a VGA port. These can be trickier to use as you'll need an HDMI-to-VGA converter, which can change digital video to analogue video. A simple port adapter won't work.

HDMI to HDMI Cable:

Connect Raspberry Pi to a Monitor or TV with a HDMI to HDMI cable.

Ethernet cable:

Ethernet cable will allow your Pi to connect with the internet. It is also useful for headless setup of Raspberry Pi

USB Keyboard and Mouse:

Any standard USB keyboard and mouse can be used with the Raspberry Pi. This plug and play devices will work without any additional driver. Simply plug them into the Raspberry Pi and they should be recognised when it starts up.

Power Supply:

It is recommended that you use a 5V, 2A USB power supply for all models of Raspberry Pi.

SD Card:

The latest version of Raspbian, the default operating system recommended for the Raspberry Pi, requires an 8GB (or larger) micro SD card. SD card will store the operating systems as well as all the file and applications created by you.

Installation Guide:

Now since you have all the required hardware, we will now learn how to get the operating system onto your microSD card so that you can start using software on your Raspberry Pi

Get Raspbian OS on your microSD card:

Raspbian comes pre-installed with plenty of software for education, programming and general use. It has Python, Scratch, Sonic Pi, Java, Mathematica and more.

1. To download Raspbian log on to raspberrpi.org and click on the download, then click on Raspbian and lastly download the RASPBIAN BUSTER WITH DESKTOP file. You can choose either the Torrent file or ZIP file.
2. The downloaded file will be in zip format. To unzip the file, you will require an unzip tool. You can use any unzipping tool viz. WINRAR, 7ZIP etc. After unzipping the file, you will find a disc image file in the unzipped folder.
3. Now format the SD Card before writing the disc image file on the SD card. You can use SD Formatter tool or any other tool of your wish.
4. To write the image file of the operating system on the SD card you will require a Disk Imager tool. For this you can use Win32 Disk Imager tool.
5. Once the image is written on the SD Card, your untitled SD card will now have the name boot. Your SD Card will now hold the Raspbian Operating system required for the first-time setup.

Plugging in your Raspberry Pi:

1. Begin by placing your SD card into the SD card slot on the Raspberry Pi. It will only fit one way.
2. Next, plug your keyboard and mouse into the USB ports on the Raspberry Pi.
3. Make sure that your monitor or TV is turned on, and that you have selected the right in put (e.g. HDMI 1, DVI, etc).
4. Connect your HDMI cable from your Raspberry Pi to your monitor or TV.
5. If you intend to connect your Raspberry Pi to the internet, plug an Ethernet cable into the Ethernet port, or connect a WiFi dongle to one of the USB ports (unless you have a Raspberry Pi 3).
6. When you're happy that you have plugged all the cables and SD card in correctly, connect the micro USB power supply. This action will turn on and boot your Raspberry Pi.

NOTE: for Raspberry Pi 4 please follow the instruction given below.

1. To download Raspberry pi OS (buster)log on to **raspberrypi.com** and click on the download and then click of rasberry pi os and click on **Raspberry pi OS buster.**

2. Follow the above mention step from 2 to 5 of **INSTALLATION GUIDE** section.

3. After writing the image on SD Card plug in the SD Card in to the Raspberry pi 4. If boot normally and you see the raspberry pi screen than ok. But if not than make the following changes in the SD card ‘config' file with notepad++.

1. Uncomment the line 'hdmi_force_hotplug=1’
2. Uncomment the line 'hdmi_group=1’
3. Uncomment the line 'hdmi_mode=1'
4. Save and exit the 'config' file
5. Now insert again the memory card into the raspberry pi it will work.

Raspberry Pi Headless Setup (SSH)

A headless setup is one where you do not need an external monitor, mouse and keyboard. You can use the same monitor, mouse and keyboard as that of your personal PC or laptop. So here it reduces cost of getting started with Raspberry Pi. A headless setup really unleashes the potential of the Raspberry Pi, allowing you to setup a remote system to perform functions that can be updated from the comfort of your home computer.

Hardware Guide:

For the headless setup, you will require the following hardware

1. Raspberry Pi 3 (latest model)
2. 5VDC Micro USB power supply
3. 8GB or larger microSD card
4. SD Card Reader
5. Ethernet cable
6. Home Router

Installation Guide:

Software requirement is as follows

1. Raspberry Operating System
2. Putty

Get Raspbian OS on your microSD card:

The installation process for the Raspbian operating system is same as we learned earlier but the only difference is here you need to enable SSH connection to communicate with Raspberry Pi from your PC or Laptop. Follow all the 4 steps as we discussed in the previous lesson.

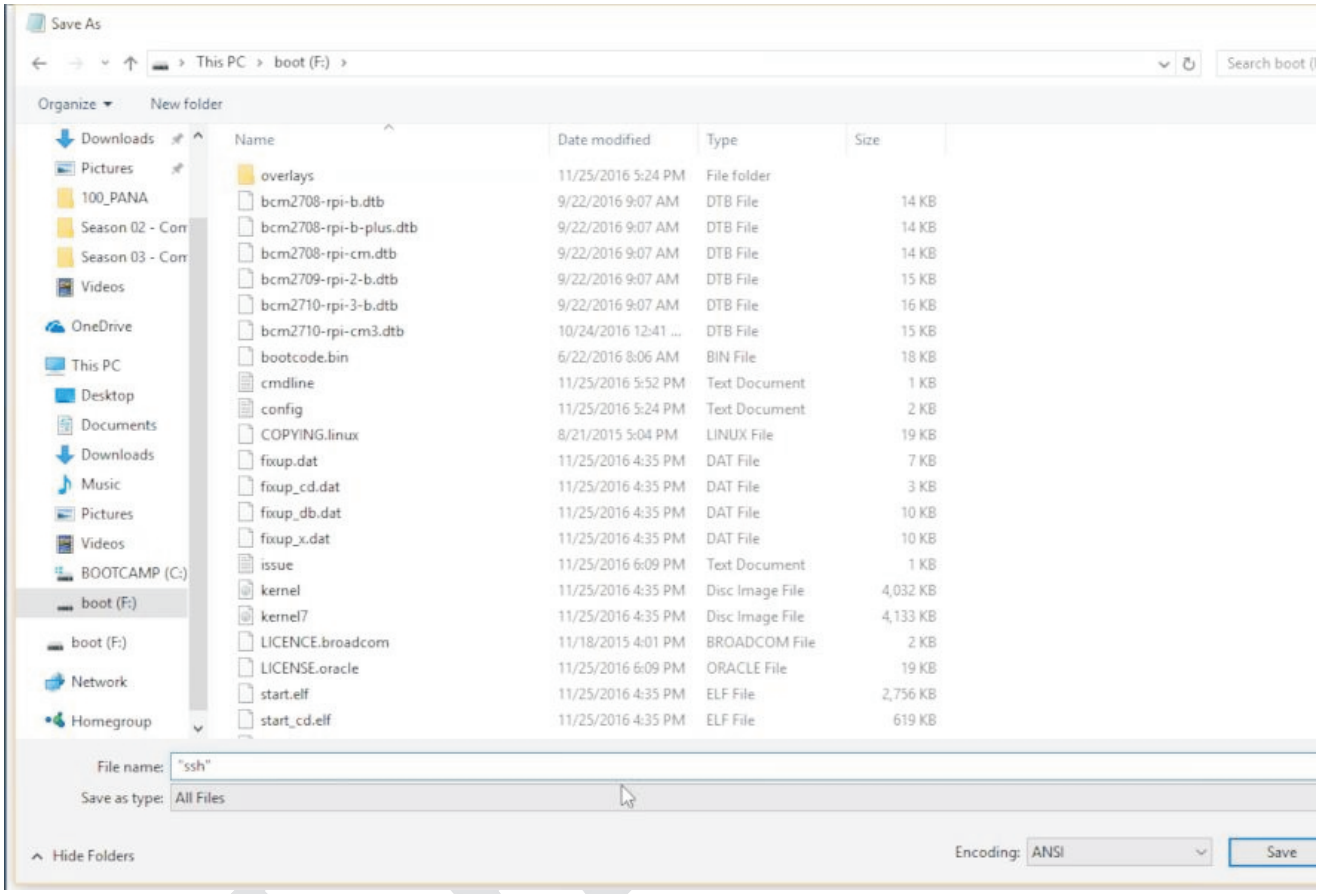
Enabling SSH and installing Putty:

1. Once the image is written on the SD Card, your untitled SD card will now have the name boot. Now navigate to the boot (i.e. SD Card), then open an instance of Notepad and save it with the file name “ssh” and file type as ‘!ll files’ in the boot (i.e. SD Card). (Note that you have the save the file with the name “ssh” and not ssh).

SSH?

Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network. The best-known example application is for remote login to computer systems by users.

SSH provides a secure channel over an unsecured network in a client-server architecture, connecting an SSH client application with an SSH server. Common applications include remote command-line login and remote command execution, but any network service can be secured with SSH.

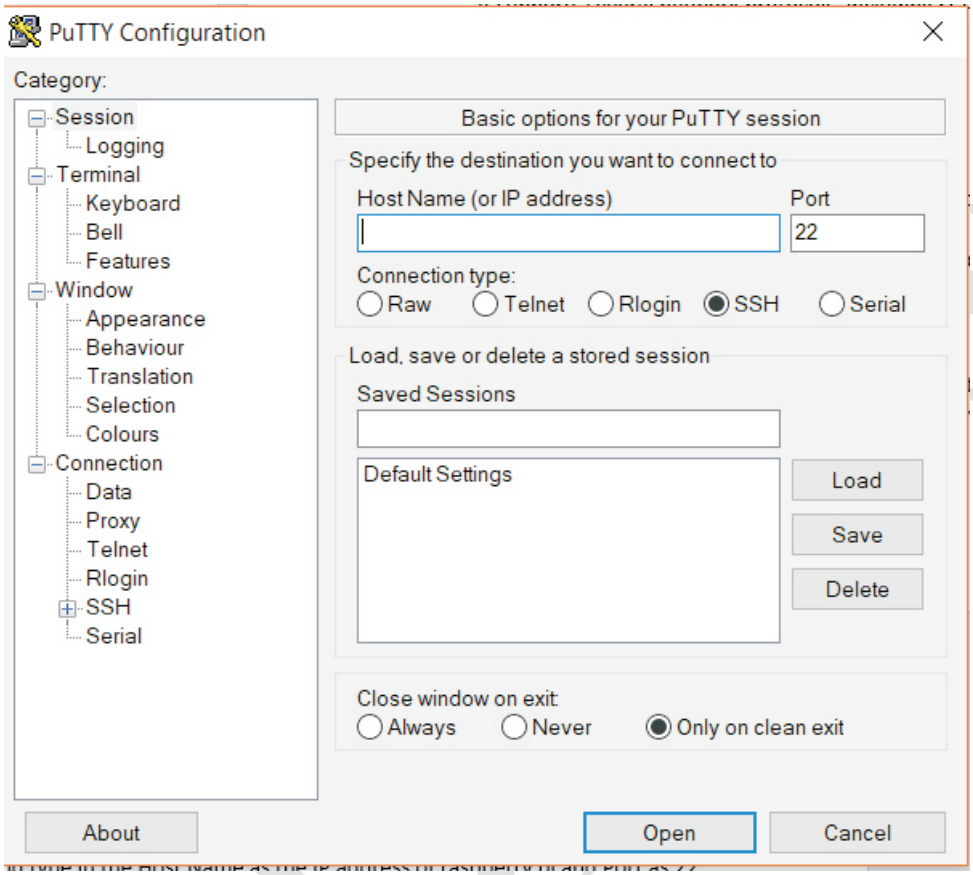


2. Saving this file will enable SSH access for the raspberry pi Image so that we can SSH into it. Now you can eject the SD card and plug it into your raspberry pi.
3. Log on to www.putty.org , download and Install putty on your personal PC or laptop which you will be using for SSH access. Putty is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port

Plugging in your Raspberry Pi:

1. Begin by placing your SD card into the SD card slot on the Raspberry Pi. It will only fit one way.
2. Plug in one end of the ethernet cable into the raspberry pi and the other end into your home router.
3. Once you are happy, connect the micro USB power supply and turn it on. This action will turn on your raspberry pi.
4. Now connect your personal PC or laptop to the router.

5. Find the IP address assigned to your raspberry pi by typing the following command in command prompt: - ping -4 raspberrypi.local
6. Or else Log in to your router and navigate to DHCP Client List and find out the IP address assigned to your raspberry pi.
7. Now open putty and type in the Host Name as the IP address of raspberry pi and Port as 22. Select the connection type as SSH and click on Open.



8. Now it will ask you to login. This is the user name and your default user name is pi and password is raspberry. (Note that here the password typed will not be seen. This is a security feature I raspberry pi). Now press Enter you will be logged into your raspberry pi.

Tips:

In Putty, you can write the Hostname as raspberrypi.local instead of finding the IP address, every time you login.

But sometimes writing raspberrypi.local will give you an error in windows pc, so for that you will have to download and install !pple's Bonjour Print Services for Windows v2.0.2 on your windows pc.

Remote access to the Pi's graphical interface (VNC)

Sometimes it is not convenient to work directly on the Raspberry Pi. Maybe you would like to work on it from another device by remote control.

VNC is a graphical desktop sharing system that allows yo u to remotely control the desktop interface of one computer (running VNC Server) from another computer or mobile device (running VNC Viewer). VNC Viewer transmits the keyboard and either mouse or touch events to VNC Server, and receives updates to the scre en in return.

You will see the desktop of the Raspberry Pi inside a window on your computer or mobile device. You'll be able to control it as though you were working on the Raspberry Pi itself.

Hardware Guide:

For the headless setup, you will require the following hardware

1. Raspberry Pi 3 (latest model)
2. 5VDC Micro USB power supply
3. 8GB or larger microSD card
4. SD Card Reader
5. Ethernet cable
6. Home Router

Installation Guide:

Software requirement is as follows

1. Raspbian Operating System
2. Putty
3. VNC Viewer

Follow all the instruction as discussed in earlier lessons for getting Raspbian OS on your SD card, enabling SSH and installing putty.

Enabling VNC and Installing VNC Viewer:

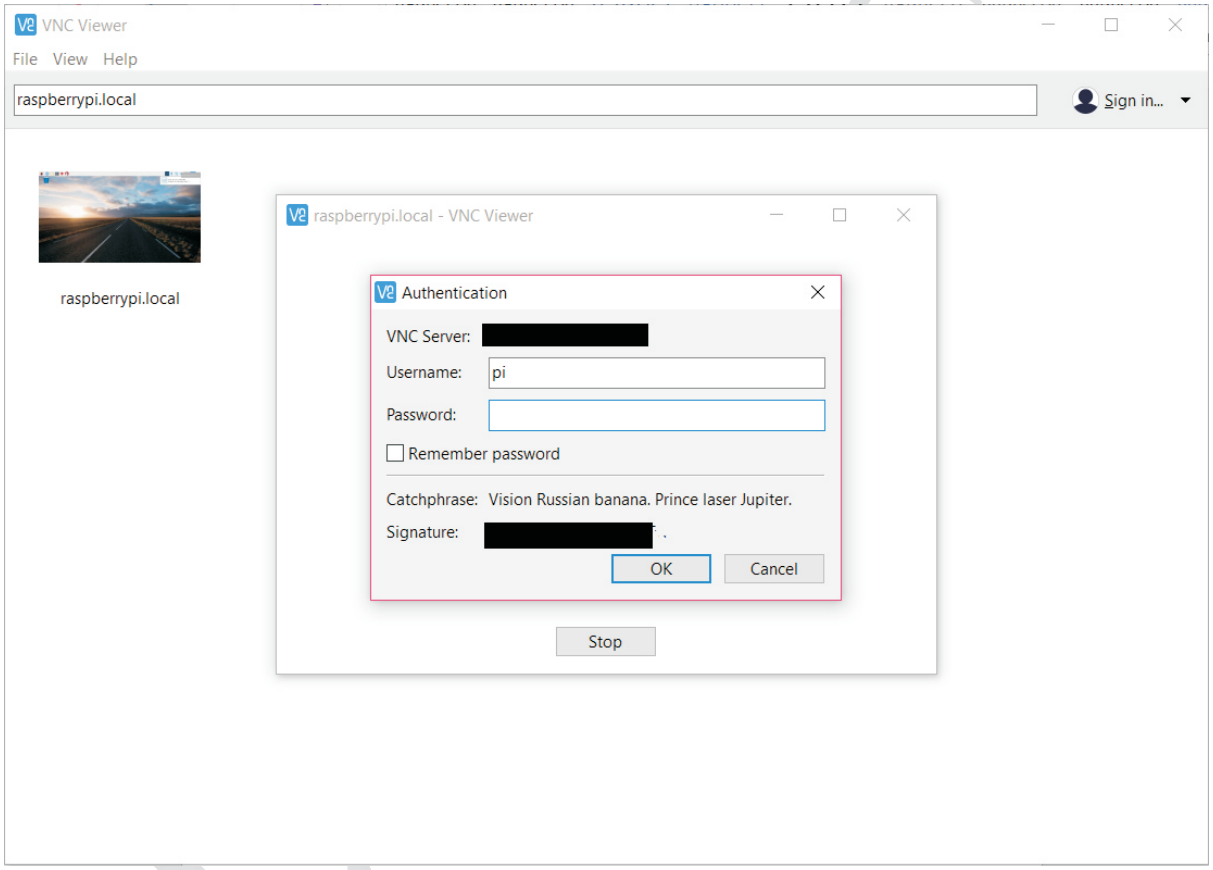
Now open putty, start the session using the host name raspberrypi.local. Then type in your default user pi and password as raspberry.

1. Once logged in, type the commands as follows: - sudo raspi-config
2. Now navigate to Interfacing options
3. Scroll down and select VNC > Yes >OK > FINISH
4. Now reboot your raspberry pi by typing sudo reboot.

Now download and install VNC Viewer on your PC or Laptop which you will be using for remote access. RealVNC is one of the recommended tool.

Plugging in your Raspberry Pi:

1. Begin by placing your SD card into the SD card slot on the Raspberry Pi. It will only fit one way.
2. Plug in one end of the ethernet cable into the raspberry pi and the other end into your home router.
3. Once you are happy, connect the micro USB power supply and turn it on. This action will turn on your raspberry pi.
4. Now connect your personal PC or laptop to the router.
5. Now open VNC viewer on your PC or laptop and type in raspberrypi.local and hit enter.
6. Now enter the username as pi and password as raspberry to login into your raspberry pi.



Linux Commands: Exploring the Raspbian

Here are some fundamental and common Linux commands with example usage:

FILESYSTEM

LS

The ls command lists the content of the current directory (or one that is specified). It can be used with the -l flag to display additional information (permissions, owner, group, si ze, date and timestamp of last edit) about each file and directory in a list format. The -a flag allows you to view files beginning with . (i.e. dotfiles).

CD

Using cd changes the current directory to the one specified. You can use relative (i.e. cd directoryA) or absolute (i.e. cd /home/pi/directoryA) paths.

PWD

The pwd command displays the name of the present working directory: on a Raspberry Pi, entering pwd will output something like /home/pi.

MKDIR

You can use mkdir to create a new directory, e.g. mkdir newDir would create the directory newDir in the present working directory.

RMDIR

To remove empty directories, use rmdir. So, for example, rmdir oldDir will remove the directory oldDir only if it is empty.

RM

The command rm removes the specified file (or recursively from a directory when used with -r). Be careful with this command: files deleted in this way are mostly gone for good!

CP

Using cp makes a copy of a file and places it at the specified location (this is similar to copying and pasting). For example, cp ~/fileA /home/otherUser/ would copy the file fileA from your home directory to that of the user otherUser (assuming you have permission to copy it there). This command can either take FILE FILE (cp fileA fileB), FILE DIR (cp fileA /directoryB/) or -r DIR DIR (which recursively copies the contents of directories) as arguments.

MV

The mv command moves a file and places it at the specified location (so where cp performs a 'copy-paste', mv performs a 'cut-paste'). The usage is similar to cp. So mv ~/fileA /home/otherUser/ would move the file fileA from your home directory to that of the user otherUser. This command can either

take FILE FILE (mv fileA fileB), FILE DIR (mv fileA /directoryB/) or DIR DIR (mv /directoryB /directoryC) as arguments. This command is also useful as a method to rename files and directories after they've been created.

TOUCH

The command touch sets the last modified time-stamp of the specified file(s) or creates it if it does not already exist.

CAT

You can use cat to list the contents of file(s), e.g. cat thisFile will display the contents of thisFile. Can be used to list the contents of multiple files, i.e. cat *.txt will list the contents of all .txt files in the current directory.

HEAD

The head command displays the beginning of a file. Can be used with -n to specify the number of lines to show (by default ten), or with -c to specify the number of bytes.

TAIL

The opposite of head, tail displays the end of a file. The starting point in the file can be specified either through -b for 512 byte blocks, -c for bytes, or -n for number of lines.

CHMOD

You would normally use chmod to change the permissions for a file. The chmod command can use symbols u (user that owns the file), g (the files group), and o (other users) and the permissions r (read), w (write), and x (execute). Using chmod u+x *filename* will add execute permission for the owner of the file.

CHOWN

The chown command changes the user and/or group that owns a file. It normally needs to be run as root using sudo e.g. sudo chown pi:root *filename* will change the owner to pi and the group to root.

SSH

ssh denotes the secure shell. Connect to another computer using an encrypted network connection.

SCP

The scp command copies a file from one computer to another using ssh.

SUDO

The sudo command enables you to run a command as a superuser, or another user. Use sudo -s for a superuser shell.

DD

The dd command copies a file converting the file as specified. It is often used to copy an entire disk to a single file or back again. So, for example, dd if=/dev/sdd of=backup.img will create a backup image from an SD card or USB disk drive at /dev/sdd. Make sure to use the correct drive when copying an image to the SD card as it can overwrite the entire disk.

DF

Use df to display the disk space available and used on the mounted filesystems. Use df -h to see the output in a human-readable format using M for MBs rather than showing number of bytes.

UNZIP

The unzip command extracts the files from a compressed zip file.

TAR

Use tar to store or extract files from a tape archive file. It can also reduce the space required by compressing the file similar to a zip file.
To create a compressed file, use tar -cvzf *filename.tar.gz* *directory/* To extract the contents of a file, use tar -xvzf *filename.tar.gz*

PIPES

A pipe allows the output from one command to be used as the input for another command. The pipe symbol is a vertical line |. For example, to only show the first ten entries of the ls command it can be piped through the head command ls | head

TREE

Use the tree command to show a directory and all subdirectories and files indented as a tree structure.

&

Run a command in the background with &, freeing up the shell for future commands.

WGET

Download a file from the web directly to the computer with wget. So wget https://www.raspberrypi.org/documentation/linux/usage/commands.md will download this file to your computer as commands.md

CURL

Use curl to download or upload a file to/from a server. By default, it will output the f ile contents of the file to the screen.

MAN

Show the manual page for a file with man. To find out more, run man man to view the manual page of the man command.

SEARCH

GREP

Use grep to search inside files for certain search patterns. For example, grep "search" *.txt will look in all the files in the current directory ending with .txt for the string search.
The grep command supports regular expressions which allows special letter combinations to be included in the search.

AWK

awk is a programming language useful for searching and manipulating text files.

FIND

The find command searches a directory and subdirectories for files matching certain patterns.

WHEREIS

Use whereis to find the location of a command. It looks through standard program locations until it finds the requested command.

NETWORKING

PING

The ping utility is usually used to check if communication can be made with another host. It can be used with default settings by just specifying a hostname (e.g. ping raspberrypi.org) or an IP address (e.g. ping 8.8.8.8). It can specify the number of packets to send with the -c flag.

NMAP

nmap is a network exploration and scanning tool. It can return port and OS information about a host or a range of hosts. Running just nmap will display the options available as well as example usage.

HOSTNAME

The hostname command displays the current hostname of the system. A privileged (super) user can set the hostname to a new one by supplying it as an argument (e.g. hostname new-host).

IFCONFIG

Use ifconfig to display the network configuration details for the interfaces on the current system when run without any arguments (i.e. ifconfig). By supplying the command with the name of an interface (e.g. eth0 or lo) you can then alter the configuration: check the manual page for more details.

GPIO: Light the LED with Python

After setting up the raspberry pi and having hands on practice with the Linux commands, you are now familiar with raspberry pi. Now it's time to work with the GPIO pins of the raspberry pi to have an external interface with the raspberry pi.

Hardware Guide:

Along with the basic setup you will require the following components to get started with the GPIO pins as follows:

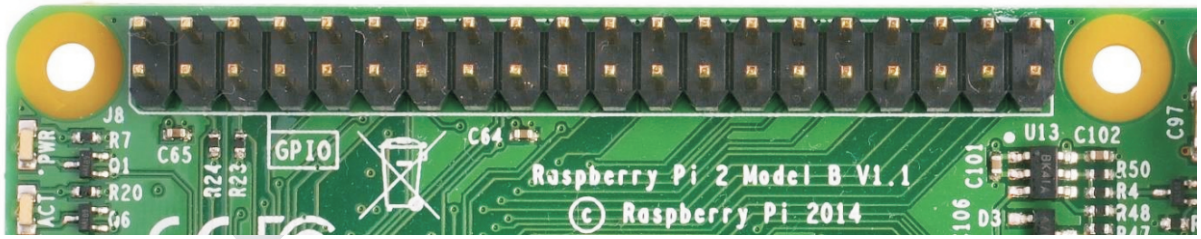
1. LED
2. Resistor
3. Connecting wires
4. Breadboard

Before learning this lesson, you must understand the pin numbering system of the GPIO pins.

GPIO?

One powerful feature of the Raspberry Pi is the row of GPIO (general purpose input/output) pins along the top edge of the board.

These pins are a physical interface between the Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input) or that the Pi can turn on or off (output). Of the 40 pins, 26 are GPIO pins and the others are power or ground pins (plus two ID EEPROM pins which you should not play with unless you know your stuff!)



What are they for? What can we do with them?

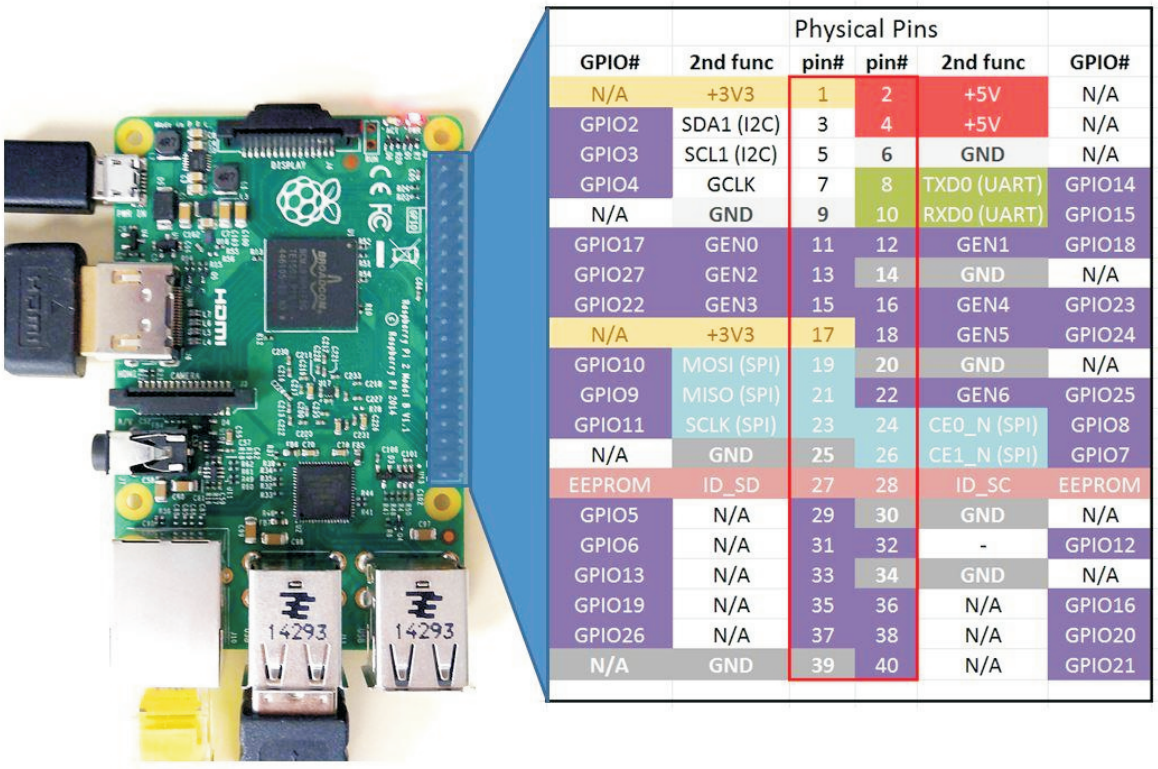
You can program the pins to interact in amazing ways with the real world. Inputs don't have to come from a physical switch; it could be input from a sensor or a signal from another computer or device, for example. The output can also do anything, from turning on an LED to sending a signal or data to another device. If the Raspberry Pi is on a network, you can control devices that are attached to it from anywhere** and those devices can send data back. Connectivity and control of physical devices over the internet is a powerful and exciting thing, and the Raspberry Pi is ideal for this.

How the GPIO pins work?

Output
When we use a GPIO pin as an output. Each pin can turn on or off, or go HIGH or LOW in computing terms. When the pin is HIGH it outputs 3.3 volts (3v3); when the pin is LOW it is off.

Input
GPIO outputs are easy; they are on or off, HIGH or LOW, 3v3 or 0v. Inputs are a bit trickier because of the way that digital devices work. Although it might seem reasonable just to connect a button across an input pin and a ground pin, the Pi can get confused as to whether the button is on or off. It might work properly, it might not. It's a bit like floating about in deep space; without a reference, it would be hard to tell if you were going up or down, or even what up or down meant!
Therefore, you will see phrases like "pull up" and "pull down" in Raspberry Pi GPIO tutorials. It's a way of giving the input pin a reference so it knows for certain when an input is received.

Warning: Randomly plugging wires and power sources into your Pi, however, may kill it. Bad things can also happen if you try to connect things to your Pi that use a lot of power.



		Physical Pins			
GPIO#	2nd func	pin#	pin#	2nd func	GPIO#
N/A	+3V3	1	2	+5V	N/A
GPIO2	SDA1 (I2C)	3	4	+5V	N/A
GPIO3	SCL1 (I2C)	5	6	GND	N/A
GPIO4	GCLK	7	8	TXD0 (UART)	GPIO14
N/A	GND	9	10	RXD0 (UART)	GPIO15
GPIO17	GEN0	11	12	GEN1	GPIO18
GPIO27	GEN2	13	14	GND	N/A
GPIO22	GEN3	15	16	GEN4	GPIO23
N/A	+3V3	17	18	GEN5	GPIO24
GPIO10	MOSI (SPI)	19	20	GND	N/A
GPIO9	MISO (SPI)	21	22	GEN6	GPIO25
GPIO11	SCLK (SPI)	23	24	CE0_N (SPI)	GPIO8
N/A	GND	25	26	CE1_N (SPI)	GPIO7
EEPROM	ID_SD	27	28	ID_SC	EEPROM
GPIO5	N/A	29	30	GND	N/A
GPIO6	N/A	31	32	-	GPIO12
GPIO13	N/A	33	34	GND	N/A
GPIO19	N/A	35	36	N/A	GPIO16
GPIO26	N/A	37	38	N/A	GPIO20
N/A	GND	39	40	N/A	GPIO21

A note on pin numbering:

When programming the GPIO pins there are two different ways to refer to them: GPIO numbering and physical numbering.

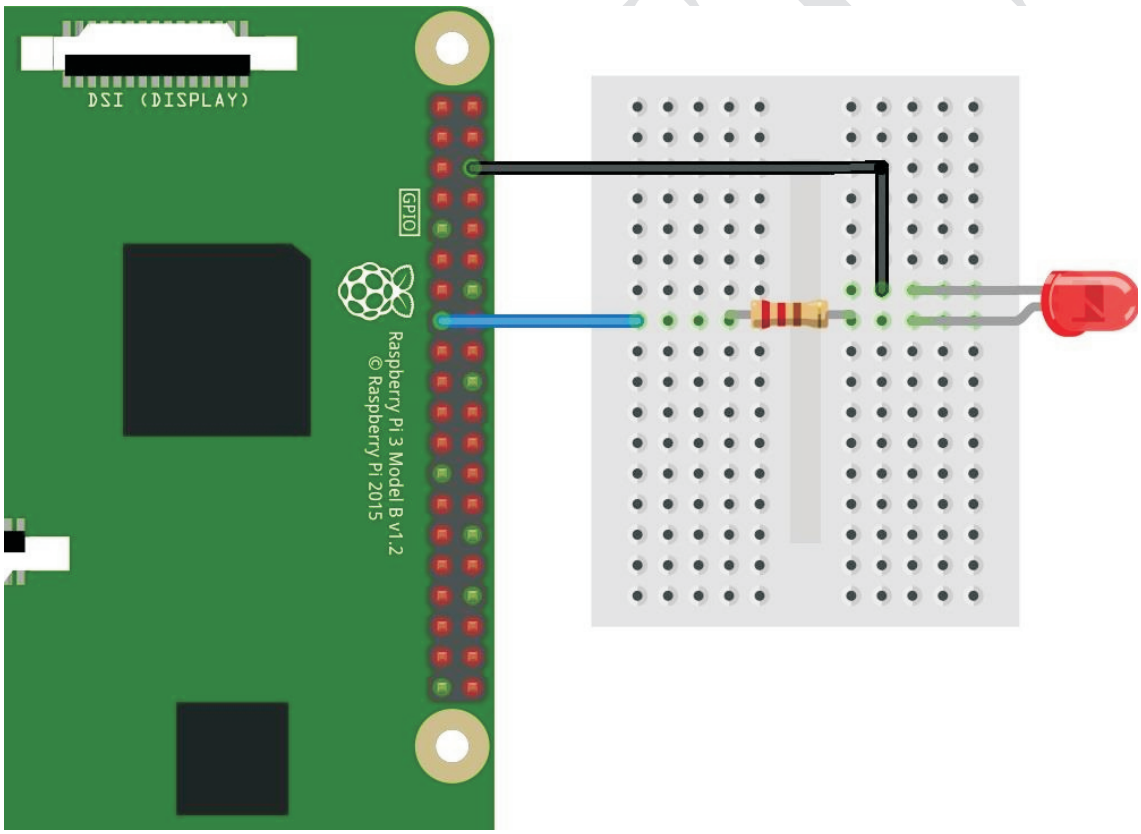
GPIO numbering:
These are the GPIO pins as the computer sees them. The numbers don't make any sense to humans, they jump about all over the place, so there is no easy way to remember them. You will need a printed reference or a reference board that fits over the pins.

Physical numbering:
The other way to refer to the pins is by simply counting across and down from pin 1 at the top le ft (nearest to the SD card).

Wiring up your Circuit:

1. Connect the GPIO22 (i.e. Physical Pin 15) Pin od raspberry pi to one end of the resistor.
2. Connect another end of resistor to the positive end (anode) of LED
3. Connect the negative end (cathode) of LED to Ground of raspberry pi.
4. Then Power on your raspberry pi

Circuit Diagram:



Software Guide:

Raspbian OS comes with many preinstalled programming environments. Here we will be using Python for coding.

To open Python, click on the application Menu, navigate t o Programming, then click on Python 3 (IDLE) an Integrated Development Environment for Python 3. After opening the IDE, go to files and open new file to start your code.

Code:

```
#Blink LED Program
#Connect the LED to GPIO 22 Pin
#LED Blink Progarm
#Connect the LED to GPIO22 (i.e. Physical Pin15)


#import GPIO and time library
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)        #set the Pin mode you will be working with

ledPin = 22                   #this is GPIO22 pin i.e Physical Pin15

#setup the ledPin(i.e. GPIO22) as output
GPIO.setup(ledPin, GPIO.OUT)
GPIO.output(ledPin, False)

try:
  while True:
    GPIO.output(ledPin, True)  #Set the LED Pin to HIGH
    print("LED ON")
    sleep(1)                   #Wait for 1 sec
    GPIO.output(ledPin, False) #Set the LED Pin to LOW
    print("LED OFF")
    sleep(1)                   #wait for 1 sec
finally:
  #reset the GPIO Pins
  GPIO.output(ledPin, False)
  GPIO.cleanup()

#end of code
```

After writing your code save is on a desired location and run it to enjoy the fun.
Great! You have completed your first GPIO program. Now you are confident enough to go forward to explore and interface new things with raspberry Pi.
If you were unable to blink the LED, don't worry, recheck your connection and debug the error so that you don't repeat it again.
Now in the next lessons we will explore and interface new things

GPIO: Program the 8x8 LED Grid Module

In this lesson, we will be interfacing 8x8 LED matrix Module with raspberry pi. Since you are now familiar with the GPIO pins, it will be a fun to know more about the python coding and installing libraries to do more advance things.

Hardware Guide:

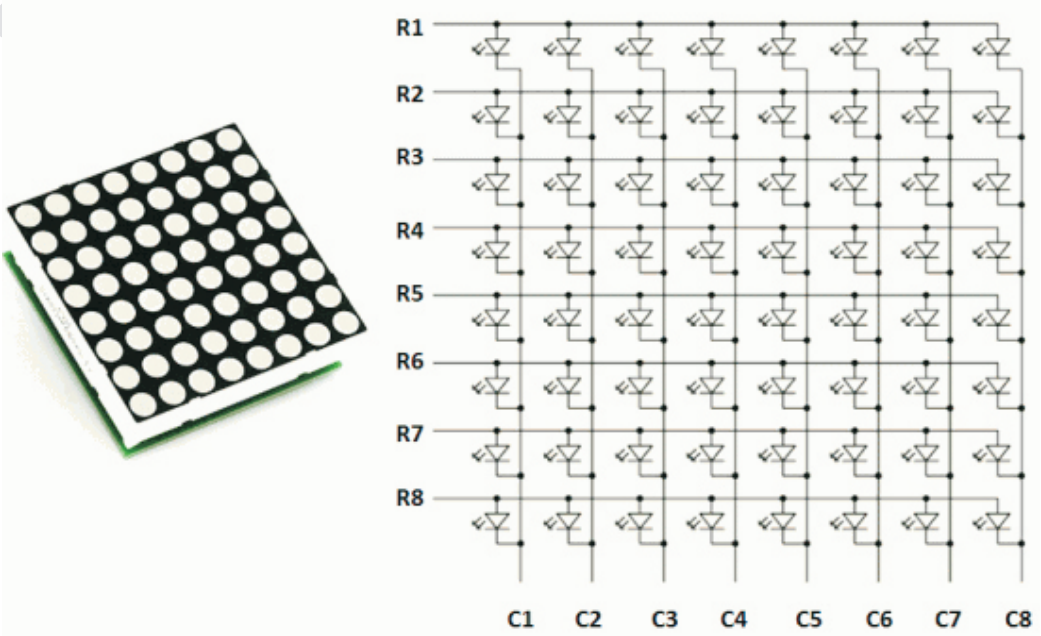
For completing this lesson, you will require the following things along with your initial raspberry pi setup

1. 8x8 LED matrix module
2. 7219 driver board
3. Connecting wires

8x8 LED matrix Module:

A LED-Matrix Display is a display device which contains light emitting diodes aligned in the form of matrix. This LED matrix displays are used in applications where Symbol, Graphic, Characters, Alphabets, Numerals are needed to be displayed together in static as well as Scrolling motion. LED Matrix Display is manufactured in various dimensions like 5x7,8x8,16x8,128x16, 128x32 and 128x64 where the numbers represent LED's in rows and columns, respectively. Also, these displays come in different colours such as Red, Green, Yellow, Blue, Orange, White.

In LED matrix display, multiple LED's are wired together in rows and columns, to minimize the number of pins required to drive them. The matrix pattern is made either in row anode-column cathode or row cathode-column anode pattern. In row anode-column cathode pattern, the entire row is anode while all columns serve as cathode which is shown below and it is vice-versa in row cathode-column anode pattern.



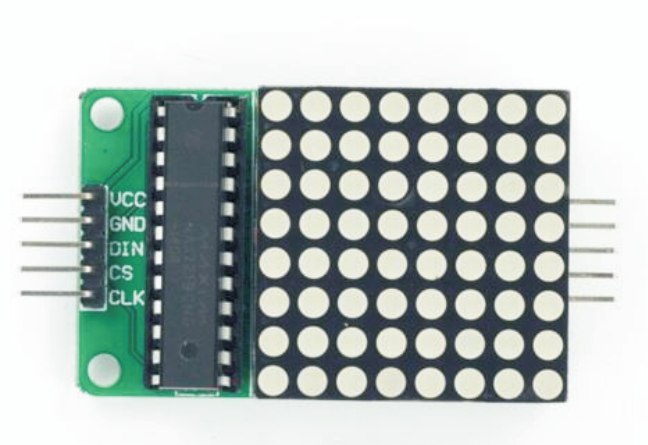
7219 Driver board:

Before interfacing LED matrix with raspberry pi, we need to connect the Max7219 IC which is an Led driver to the LED matrix display. The reason behind using this led driver is that it drives the 64 Led's simultaneously which in turn reduces the number of wires so that the user will find it easy to connect the display to the raspberry pi.

The MAX7219 has four wire SPI interface (we need only these four wires to interface it to the raspberry pi):

1. Din - MOSI - Master Output Serial Input.
2. Chip select - Load (CS) - active low Chip select.
3. Clock – SCK
4. Ground.

And off course VCC (5V) is required.



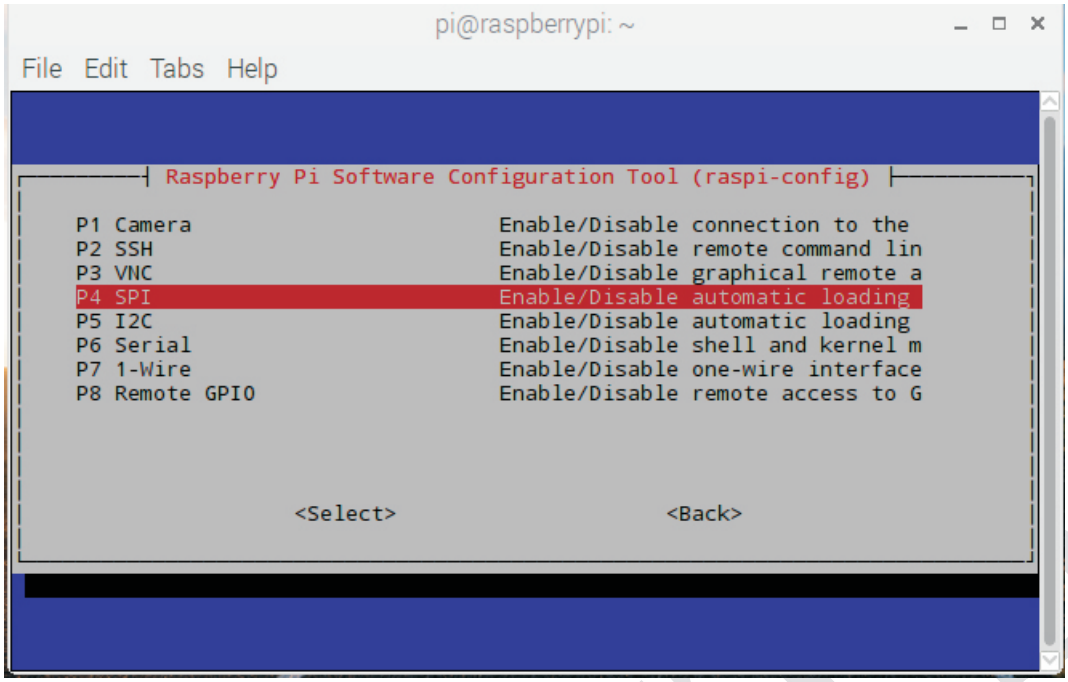
Software Guide:

Here we will be writing our code I python. But before writing our code we need to enable SPI and we need to install the library for driving the LED Matrix Module using our raspberry pi.

Pre-requisites:

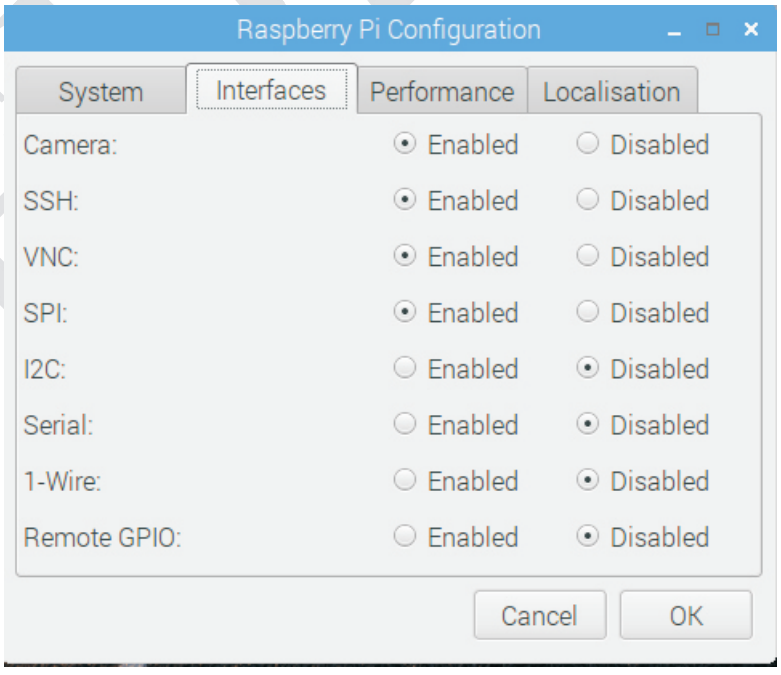
By default, the SPI kernel driver is NOT enabled on the Raspberry Pi Raspian image . Enable the SPI as follows:

1. Open terminal and type sudo raspi-config and press Enter.
2. Use the down arrow to select 5 Interfacing options
3. Arrow down to P4 SPI.
4. Select yes when it asks you to enable SPI,
5. Also select yes when it asks about automatically loading the kernel module.
6. Use the right arrow to select the <Finish> button.
7. Select yes when it asks to reboot.



Alternatively using GUI, you can also follow the following steps to enable SPI :

1. Select Preference from the raspberry pi application menu.
2. From Preference select Raspberry Pi Configuration.
3. Now form the Raspberry Pi Configuration window, navigate to Interfaces option.
4. Select the enabled radio button in front of SPI to enable it and click on OK.
5. Finally, do not forget to reboot your raspberry pi after changing this setting.



Installing the Library:

Before installing the library make sure Raspberry pi is connected to Internet.

Clone the code from github:
$ **git clone https://github.com/freedomwebtech/max7219voicecontrol**

Now change directory to max7219voicecontrol-main
$ pi@raspberrypi:~ $ **cd max7219voicecontrol-main**

Run ls command
$ pi@raspberrypi:~/max7219voicecontrol-main $ **ls**
buttons.py install.sh ledmatrix.py __pycache__

Run following command
pi@raspberrypi:~/max7219voicecontrol-main $ **sudo chmod 775 install.sh**

For installation run the following command
$ pi@raspberrypi:~/max7219voicecontrol-main $ sudo ./install.sh

Now it is the time to write our code. Open Python3, navigate to files and open a new file and write the code given below

Code:

```
from luma.led_matrix.device import max7219
from luma.core.interface.serial import spi, noop
from luma.core.render import canvas
from luma.core.virtual import viewport
from luma.core.legacy import text, show_message
from luma.core.legacy.font import proportional, CP437_FONT, TINY_FONT, SINCLAIR_FONT, LCD_FONT
from datetime import datetime
import time


serial = spi(port=0, device=0, gpio=noop())
device = max7219(serial, cascaded=1, block_orientation=-90, blocks_arranged_in_reverse_order=True)
device.contrast(16)


def test():
  now = datetime.now()
#   dt1_string = now.strftime("%H:%M:%S")
  dt1_string = now.strftime("%I:%M:%S")

  with canvas(device) as draw:
       text(draw, (3, 1), dt1_string, fill="white", font=proportional(TINY_FONT))
#       show_message(device, "Hello EDKITS", fill="red",font=(CP437_FONT),scroll_delay=0.08)


while True:
    test()
```

IMPORTANT:

There is some change is main code please change the code in ledmatrix.py as shown below

Comment this line in ledmatrix.py file

text(draw, (3, 1), dt1_string, fill="white", font=proportional(TINY_FONT))

and uncomment this line in ledmatrix.py file

show_message(device, "Hello EDKITS", fill="red",font=(CP437_FONT),scroll_delay=0.08)

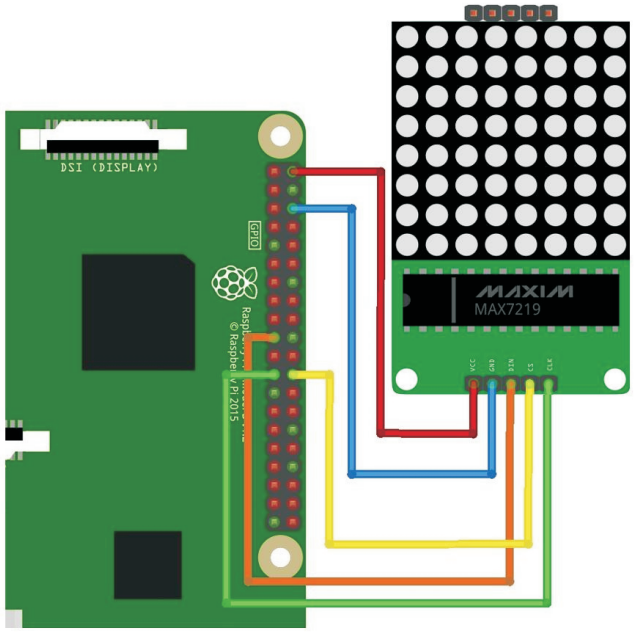
save the file. then run the command after conneting LEDmatrix module to Raspberry pi.

Wiring up your Circuit:

We are using SPI protocol for wiring LED matrix module to raspberry pi, since it reduces the number pins required for wiring the circuit. You can follow the diagram given below while wiring your circuit.

1. Connect the VCC pin of 7219 driver board to Pin2 of raspberry pi.
2. Connect the Gnd pin of 7219 driver board to Pin6 of raspberry pi.
3. Connect the DIN pin of 7219 driver board to Pin19 of raspberry pi.
4. Connect the CS pin of 7219 driver board to Pin24 of raspberry pi.
5. Lastly, connect the CLK Pin of 7219 driver board to Pin23 of raspberry pi.

Circuit Diagram:



After ensuring that the connections are done properly, power on your raspberry pi. Now open Python3 and run the code that you have written for this lesson.
So now you have learned how to interface 8x8 LED matrix module with your raspberry pi, how to install libraries and how to enable SPI.
Next lesson will bring new things and new fun, so stay tuned!

Camera Connection and capturing Images

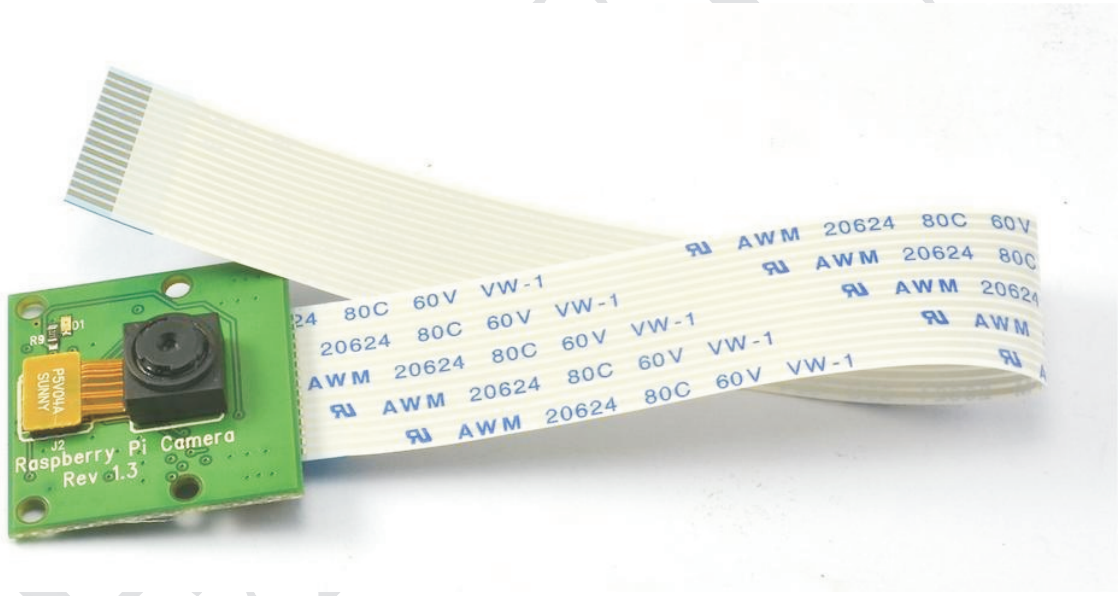
The Camera Module is a great accessory for the Raspberry Pi, allowing users to take still pictures and record video in full HD.

Hardware Guide:

For completing this lesson, you will require the Camera Module along with your initial raspberry pi setup.

Camera Module:

The Raspberry Pi Camera Board plugs directly into the CSI connector on the Raspberry Pi. The camera is supported in the latest version of Raspbian, the Raspberry Pi's preferred operating system.



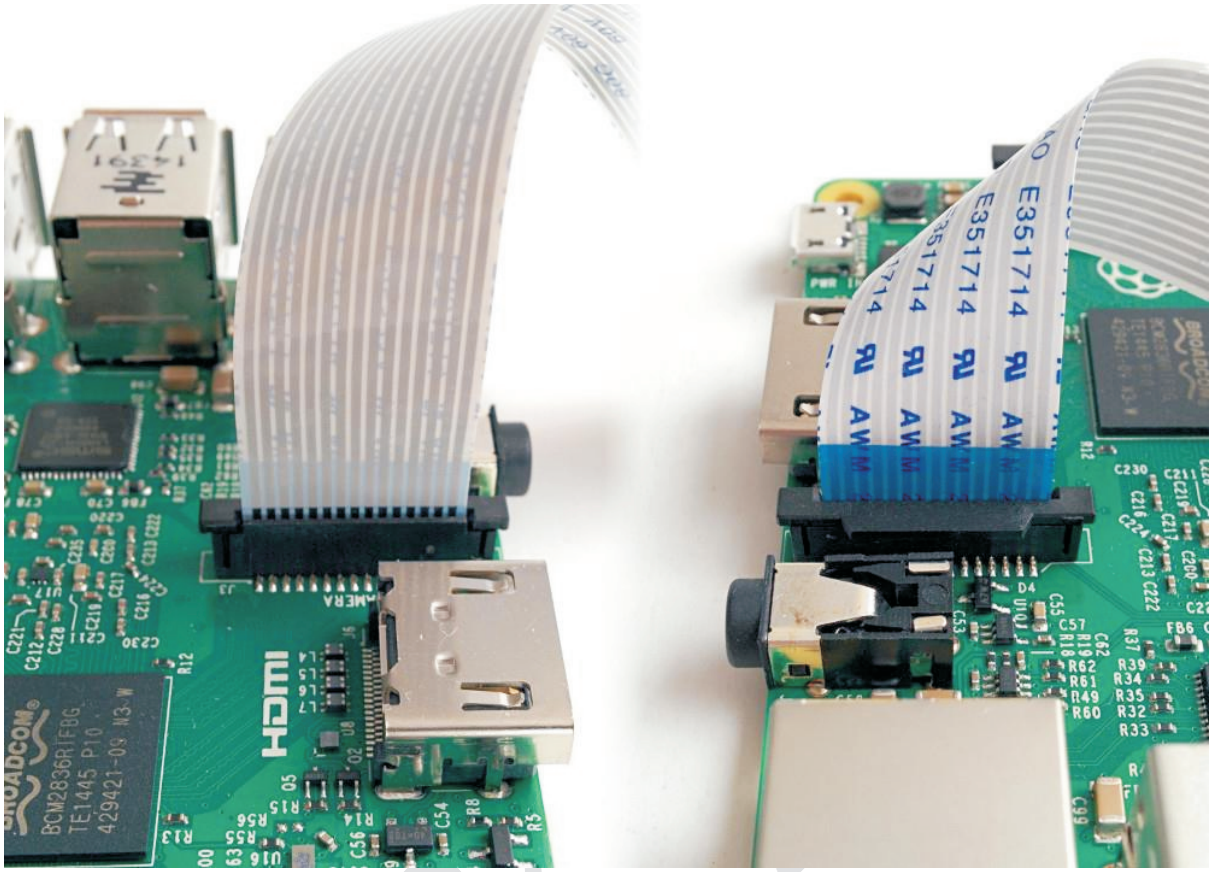
The Raspberry Pi Camera Board Features:

1. Fully Compatible with Both the Model A and Model B Raspberry Pi
2. 5MP Omnivision 5647 Camera Module
3. Still Picture Resolution: 2592 x 1944
4. Video: Supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 Recording
5. 15-pin MIPI Camera Serial Interface – Plugs Directly into the Raspberry Pi Board
6. Size: 20 x 25 x 9mm
7. Weight 3g
8. Fully Compatible with many Raspberry Pi cases

Connect the Camera Module:

First of all, with the Pi switched off, you'll need to connect the Camera Module to the Raspberry Pi's camera port, then start up the Pi and ensure the software is enabled.

1. Locate the camera port and connect the camera:



2. Start up the Pi.
3. Open the Raspberry Pi Configuration Tool from the main menu.
4. Ensure the camera software is enabled. If it's not enabled, enable it and reboot your Pi to begin.

Software Guide:

Now your camera is connected and the software is enabled, you can get started by capturing an image.

You can capture an image by just typing a single line command. Open terminal window and type the command as follows:

$ sudo raspistill -o /home/pi/Desktop/image.jpg

This command will capture an image and store it at the specified location (here the location specified is /home/pi/Desktop) with the specified name (here the name is 'image.jpg').

You can even write a code in Python to capture an image using raspberry pi camera.Open Python3, create a new file and type the code as follows:

Code:

```
#Camera Program

# import time and picamera library
from time import sleep
from picamera import PiCamera

camera = PiCamera()
camera.resolution = (1280, 720)      # selecting resolution 1280x720 px
camera.start_preview()
# Camera warm-up time
sleep(2)
camera.capture('/home/pi/Pictures/newImage.jpg') #capture and save image at specified location
camera.stop_preview()


#end of code
```

Hurray! We have learned how to interface camera wi th raspberry pi and how to capture image. You can also take videos and do much more things.

Controlling Stepper Motor with Raspberry Pi

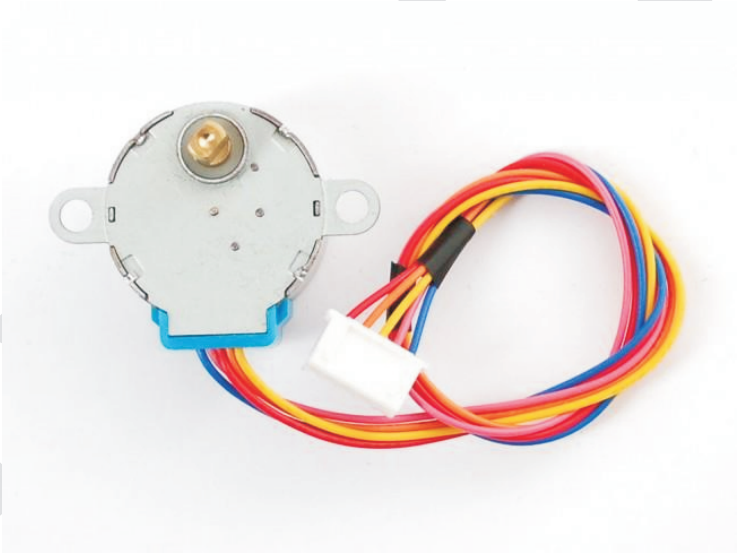
Hardware Guide:

For completing this lesson, you will require the following things along with your initial raspberry pi setup

1. Stepper Motor
2. Stepper motor driver
3. Connecting wires

Stepper Motor:

This is a great first stepper motor, good for small projects and experimenting with steppers. This uni-polar motor has a built in mounting plate with two mounting holes. There are only 32 step (11.25 degree) per revolution, and inside is a 1/16 reduction gear set. (Actually its 1/16.032 but for most purposes 1/16 is a good enough approximation) What this means is that there are really 32*16.032 steps per revolution = 513 steps! The shaft is flattened so it's easy to attach stuff to it with a set-screw.

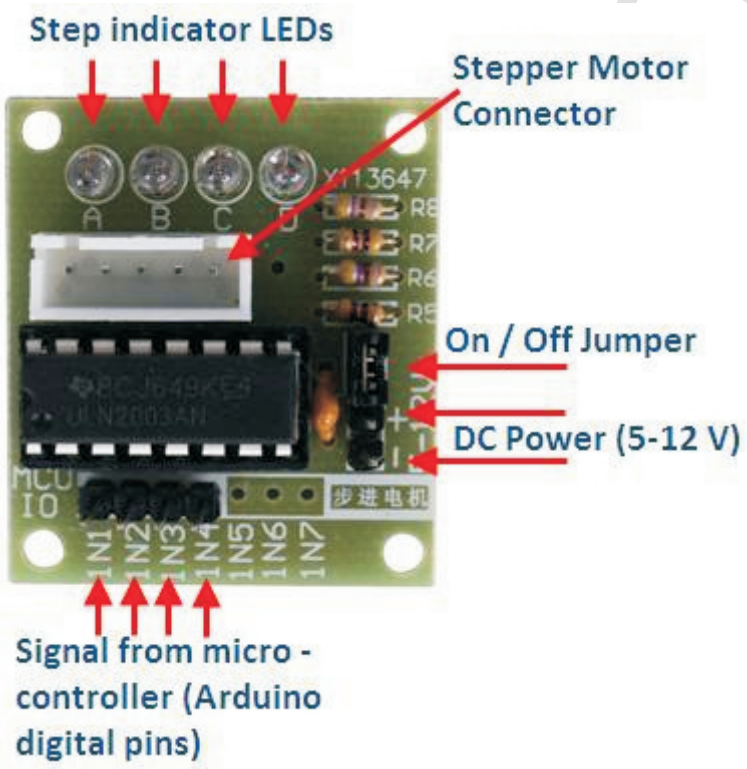


Technical details:

1. Unipolar stepper with 0.1" spaced 5-pin cable connector
2. 513 steps per revolution
3. 1/16.032 geared down reduction
4. 5V DC suggested operation
5. Weight: 37 g.
6. Dimensions: 28mm diameter, 20mm tall not including 9mm shaft with 5mm diameter
7. 9" / 23 cm long cable
8. Holding Torque: 150 gram-force*cm, 15 N*mm/ 2 oz-force*in
9. Shaft: 5mm diameter flattened
10. Approx. 42-ohm DC impedance per winding

Stepper motor driver board:

The ULN2003 stepper motor driver board allows you to easily control t he 28BYJ-48 stepper motor. One side of the board side has a 5-wire socket where the cable from the stepper motor hooks up and 4 LEDs to indicate which coil is currently powered. Th e motor cable only goes in one way, wh ich always helps. On the side, you have a motor on / off jumper (keep it on to enable power to the stepper). The two pins below the 4 resistors, is where you provide power to the stepper. Note that powering the stepper from the 5 V rail of the Raspberry Pi is not recommended. A separate 5-12 V 1 Amp power supply or battery pack should be used, as the motor may drain more current than the microcontroller can handle and could potentially damage it. In the middle of the bo ard we have the ULN2003 chip. At the bottom are the 4 control inputs that should be connected to four GPIO pins of raspberry pi.



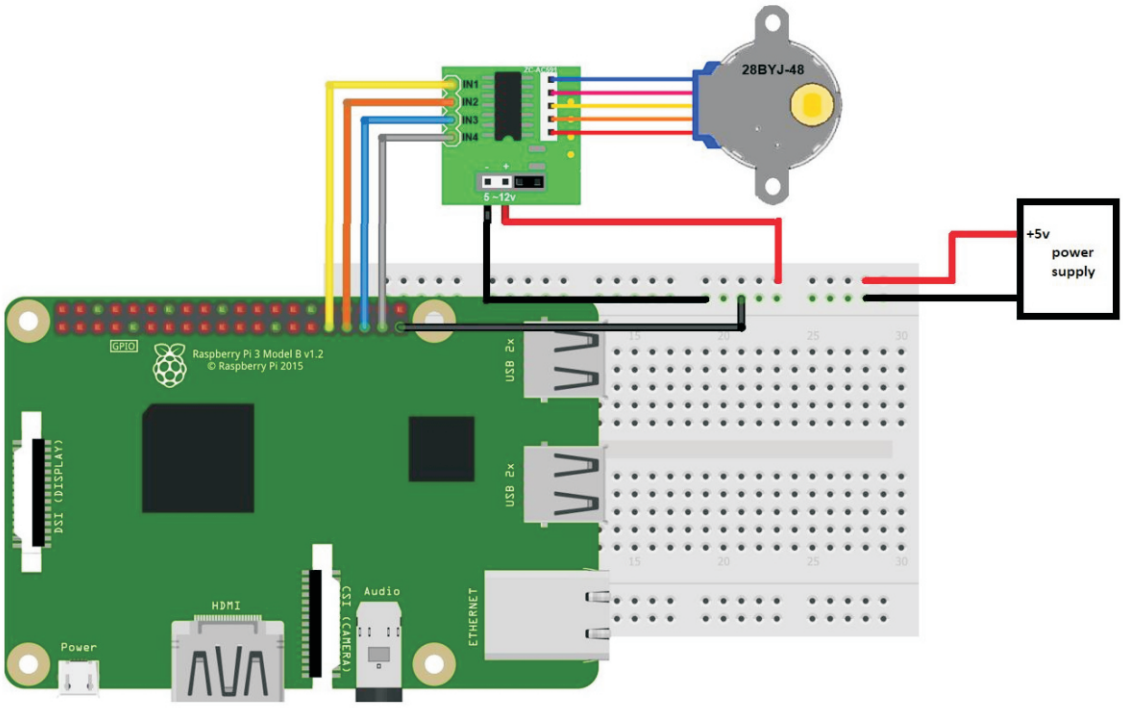
Wiring up Your circuit:

Wire your circuit as follows

1. Connect the IN1 pin of driver board to Physical Pin31 of raspberry pi
2. Connect the IN2 pin of driver board to Physical Pin33 of raspberry pi
3. Connect the IN3 pin of driver board to Physical Pin35 of raspberry pi
4. Connect the IN4 pin of driver board to Physical Pin37 of raspberry pi
5. Connect the + pin of driver board to and external 5v power supply
6. Connect the – (ground) pin of driver board, the ground pin of power supply and the ground pin of raspberry pi (Physical Pin6)to each other using a bread board.

Now connect the Stepper motor to the stepper motor connector on the driver boa rd. It will fit only in one way. After ensuring that the connections are proper, power on your raspberry pi

Circuit diagram:



Software Guide:

Now open Python3, navigate to files and create a new file write the code as follows:

Code:

```
#import GPIO and time library
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)          #set the mode to be used (BCM/BOARD)

stepPin1 = 31                     #define the pins used for stepper motor
stepPin2 = 33
stepPin3 = 35
stepPin4 = 37

GPIO.setup(stepPin1, GPIO.OUT)    #set the GPIO pins as output
GPIO.setup(stepPin2, GPIO.OUT)
GPIO.setup(stepPin3, GPIO.OUT)
GPIO.setup(stepPin4, GPIO.OUT)

GPIO.output(stepPin1, False)      #initialise GPIO pins to LOW
GPIO.output(stepPin2, False)
```

```
GPIO.output(stepPin3, False)
GPIO.output(stepPin4, False)

def singleStep(stepVal1, stepVal2, stepVal3, stepVal4):    #defining function
   GPIO.output(stepPin1, stepVal1)
   GPIO.output(stepPin2, stepVal2)
   GPIO.output(stepPin3, stepVal3)
   GPIO.output(stepPin4, stepVal4)

def clockWiseRotate(delay, steps 1):                       #defining function
   for i in range (0, steps1):
      singleStep(1, 0, 0, 0)
      time.sleep(delay)
      singleStep(1, 1, 0, 0)
      time.sleep(delay)
      singleStep(0, 1, 0, 0)
      time.sleep(delay)
      singleStep(0, 1, 1, 0)
      time.sleep(delay)
      singleStep(0, 0, 1, 0)
      time.sleep(delay)
      singleStep(0, 0, 1, 1)
      time.sleep(delay)
      singleStep(0, 0, 0, 1)
      time.sleep(delay)
      singleStep(1, 0, 0, 1)
      time.sleep(delay)

def anticlockWiseRotate(delay, steps2):                    #defining function
   for i in range (0, steps2):
      singleStep(1, 0, 0, 1)
      time.sleep(delay)
      singleStep(0, 0, 0, 1)
      time.sleep(delay)
      singleStep(0, 0, 1, 1)
      time.sleep(delay)
      singleStep(0, 0, 1, 0)
      time.sleep(delay)
      singleStep(0, 1, 1, 0)
      time.sleep(delay)
      singleStep(0, 1, 0, 0)
      time.sleep(delay)
      singleStep(1, 1, 0, 0)
      time.sleep(delay)
      singleStep(1, 0, 0, 0)
      time.sleep(delay)

try:
```

```
    while 1:                                                    #infinite loop
        delay = input("Enter delay betwwen steps (in miliseconds): ")
        steps1 = input("How many steps clockwise?: ")
        steps2 = input("How many steps Anticlcokwise?: ")
        clockWiseRotate(int(delay)/1000.0, int(steps2))
        anticlockWiseRotate(int(delay)/1000.0, int(steps2))


finally:
    GPIO.cleanup()


#end of file
```

Setting up a Web Server using Raspberry Pi

You can use a web server on a Raspberry Pi to host a full website (locally on your network or globally on the internet), or just use it to display some information you wish to share to other machines on your network.

Various web servers are available, with different advantages for usage. Here we will be using Apache web server.

Hardware Guide:

For completing this lesson, you will require the same hardware that was required for the initial raspberry pi setup except the card reader.

Here you need to connect your raspberry pi with internet. You can do it by connecting your ethernet cable to the ethernet port or by connecting the raspberry pi to internet wirelessly to router using the on board WIFI module.

Installation Guide:

Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages.

On its own, Apache can serve HTML files over HTTP, and with additional modules can serve dynamic web pages using scripting languages such as PHP.

Install Apache:

First install the apache2 package by typing the following command in to the Terminal:

sudo apt-get install apache2 -y

Test the Web Server:

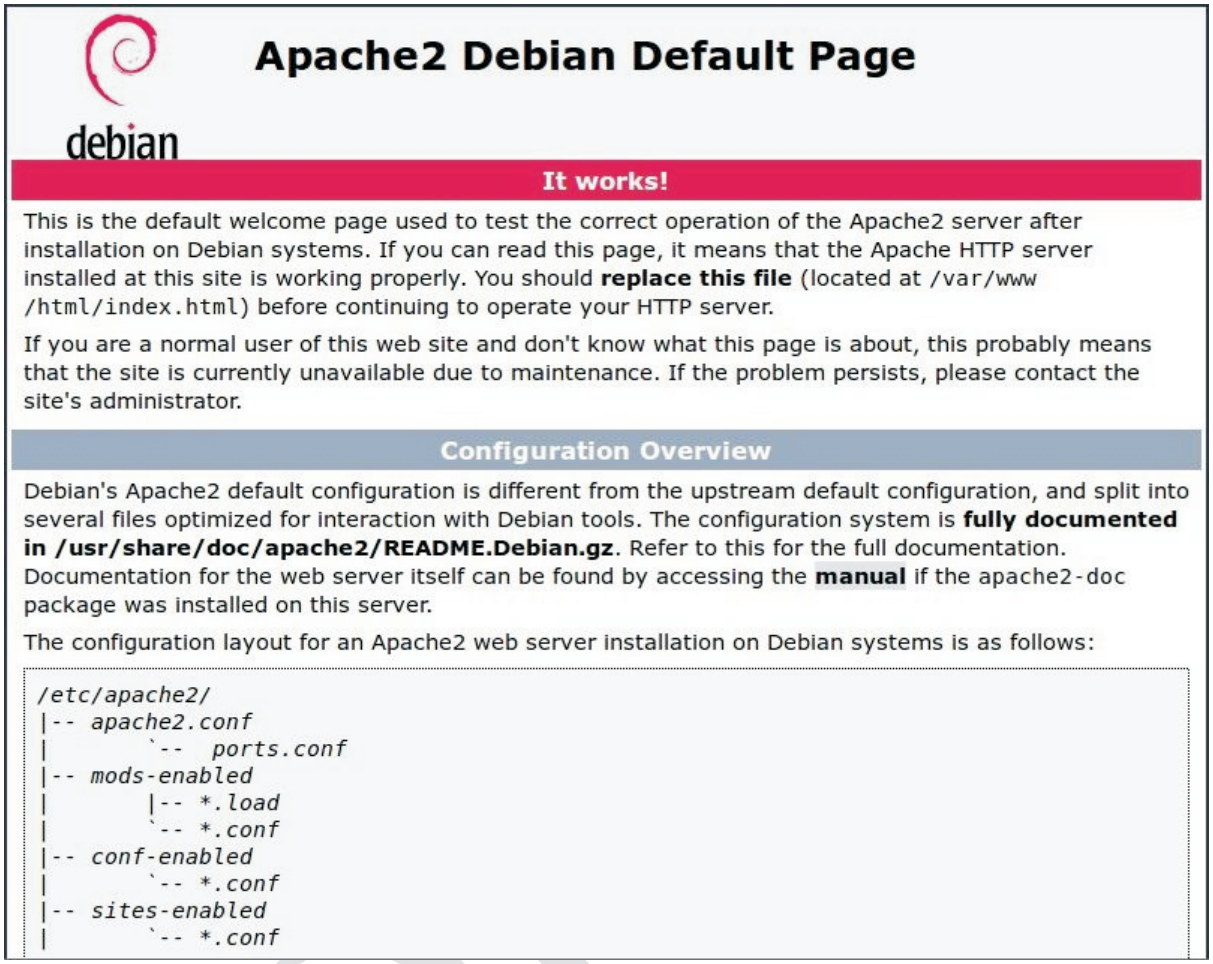
By default, Apache puts a test HTML file in the web folder.

This default web page is served when you browse to http://localhost/ on the Pi itself, or http://XX.XX.XX.XX (whatever the Pi's IP address is) from another computer on the network.

To find the Pi's IP address, type hostname -I at the command line.

Browse to the default web page either on the Pi or from another computer on the network and you should see the following:

This means you have Apache working!



Changing the default web page:

This default web page is just a HTML file on the filesystem. It is located at

/var/www/html/index.html

Navigate to this directory in a terminal window and have a look at what's inside using the following command:

cd /var/www/html

By default, there is one file in /var/www/html/ called index.html and it is owned by the root user (as is the enclosing folder). To edit the file, you need to change its ownership to your own username. Change the owner of the file (the default pi user is assumed here) using:

sudo chown pi: index.html

You can now try editing this file and then refreshing the browser to see the web page change.

If you know HTML you can put your own HTML files and other assets in this directory and serve them as a website on your local network.

Node RED: Connect LED to Internet of Things

Node-RED is a drag-and-drop visual tool which comes pre-installed on Raspbian. In this lesson, we will use Node-RED to control LEDs via the Raspberry Pi's GPIO pins.

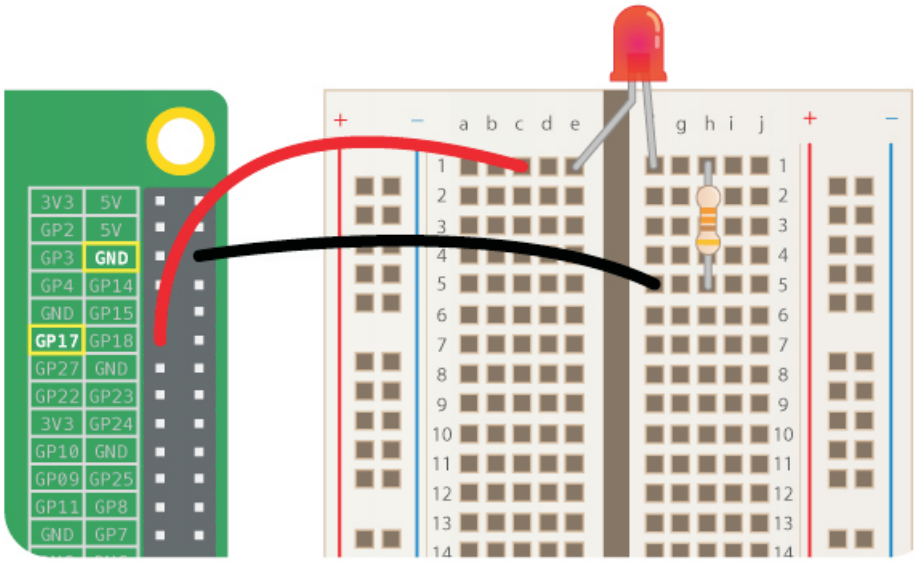
Hardware Guide:

Along with the basic setup you will require the following components to get started with the Node RED as follows:

1. LED
2. Resistor
3. Connecting wires
4. Breadboard

Wiring up the circuit:

1. Connect the raspberry pi to internet by connecting ethernet cable to the ethernet port or by connecting the on board WIFI module to the router.
2. Wire up an LED to GPIO17 (i.e. Physical pin11) on your Raspberry Pi



Software Guide:

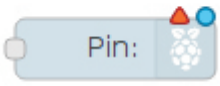
1. Start up your Raspberry Pi. Click on the Raspberry icon, then the Programming menu to open Node-RED.
2. You should see a window displaying information about Node-RED starting up.
3. Now go to the Internet menu and open Chromium Web Browser.
4. In Chromium, locate the address bar at the top and type in localhost:1880, then press Enter. This will display the Node-RED interface. (Your Raspberry Pi does not need to be connected to the internet to use Node-RED: localhost is the address the Raspberry Pi uses to refer to itself and :1880 means that it is looking at port 1880.)

Programming in Node RED:

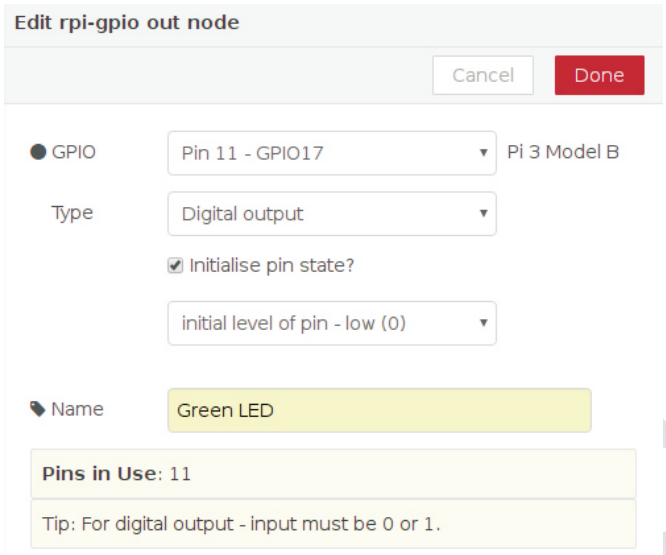
Programs in Node-RED are called flows. You can see that your blank page is labelled as Flow 1 in the tab at the top. You can create as many flows as you want and they can all run at the same time. For this guide, we will only need one flow.

1. The coloured blocks on the left side of the interface are the nodes. Scroll right down to the bottom of the list and you will see some nodes labelled Raspberry Pi.

2. You will see two nodes with the label rpi gpio: these are the ones we will use to talk to the GPIO pins on the Raspberry Pi. The first one in the list, with the raspberry icon on the left, is for inputs. Using a button push to control something would be an example of an input. The second node, with the raspberry icon on the right, is for outputs. Switching on an LED would be an example of an output. Drag an output node onto the blank page in the middle.

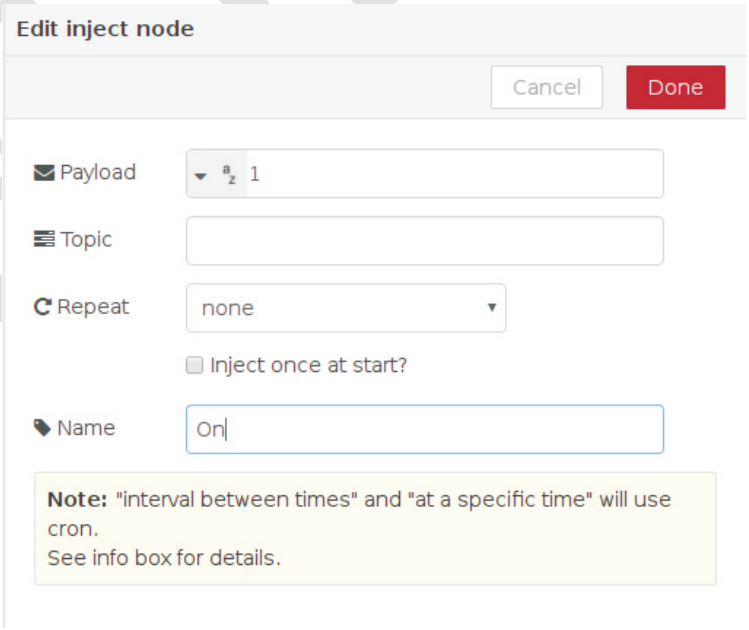


3. Double-click on the node and a box will appear to let you configure the node. Change the GPIO pin to be GPIO17 and tick Initialise pin state? Leave the setting for Initial level of pin on low. Give the node a name - we called it Green LED because the LED we used was green, but if yours is a different colour feel free to change the name. When you are finished, click Done.

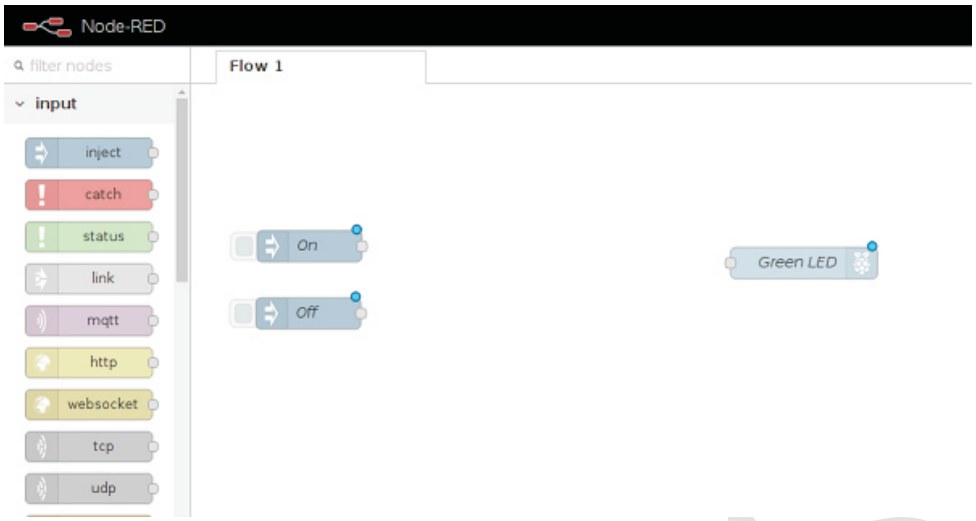


4. Now scroll back up to the list of nodes. To turn the LED on and off, we need an input. In Node-RED we can inject messages into the flow and cause things to happen as a result. Drag an inject node onto the flow.

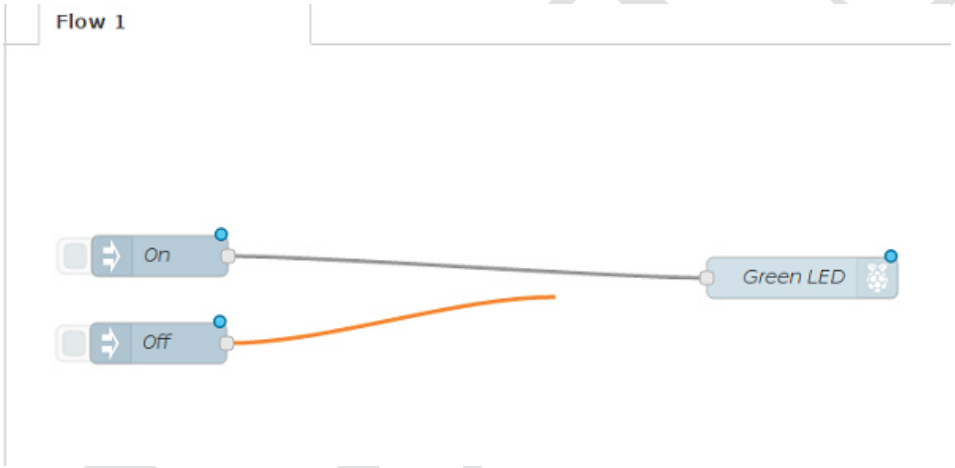
5. Double-click on the inject node. Use the drop down next to Payload to change the data type to string and type 1 in the Payload box - this will be the message. Type On in the Name box. Press Done.



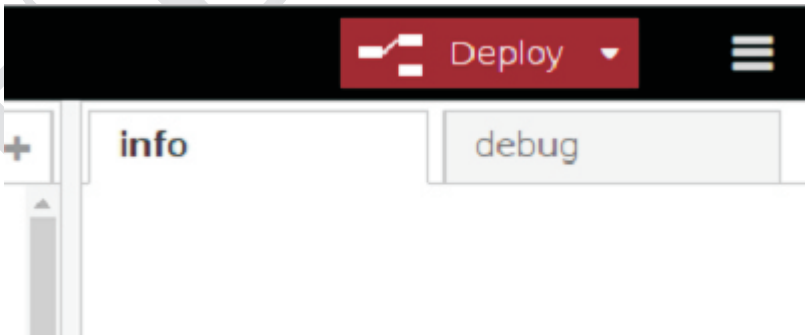
6. Repeat the previous steps to create an other inject node, except this time add 0 as the payload message, and call this node Off.



7. Now look for the grey dot on the right side of the inject nodes. Click and drag from the grey dot on the On node to the grey dot on your LED node to join them up. Repeat for the Off node, also joining it to the LED node.



8. Our flow is finished, so we can deploy it. Click on the big red Deploy button on the top right of the screen. A message should pop up at the top saying "Successfully deployed". This is like pressing the green flag in Scratch or F5 to run your code in Python.



9. Now click on the blue square on the left of the On node to inject the message 1. The Green LED node receives the message and your LED should light up. You should be able to turn the LED off by clicking the blue square on the Off node, which injects the message 0.


Node-RED
Successfully injected: On
filter nodes
Flow 1
input
inject
catch
status
link
mqtt
http
On
Off
Green LED
1

Controlling the Raspberry Pi RTC Module - I2C Real Time Clock

The Raspberry Pi does not save the date permanently. With the help of the Raspberry Pi RTC (Real Time Clock) module DS1307 you have a real-time clock – regardless of an existing internet connection. This is an advantage in many applications that require a timestamp but cannot be connected to the Internet. Especially for outdoor Pi's and logging systems, it is important to know the exact date and time. This tutorial is about installing a real-time clock (RTC) and synchronizing the system time of Linux/Raspbian.

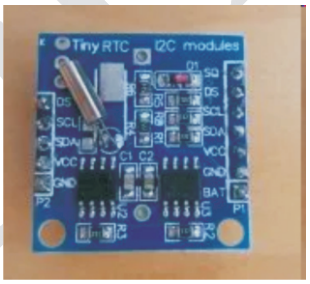
Hardware Guide:

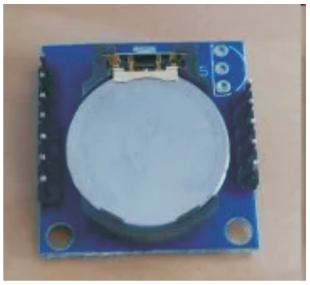
For completing this lesson, you will require the following things, along with your initial raspberry pi setup.

1. Tiny RTC DS1307 I2C Module
2. Connecting Wires

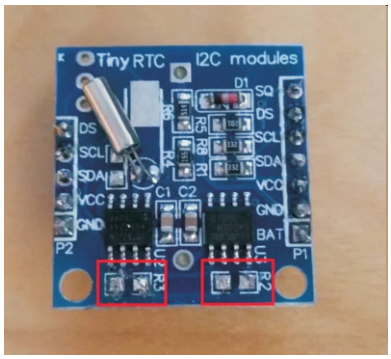
Preparation

The Tiny RTC modules are actually made for the Arduino, whose IO pins work at 5V. Since the GPIOs of the Raspberry Pi only work at 3.3V and a higher voltage can damage them, we have to remove two resistors from the board.





On the back are the two resistors (R2 and R3) which are removed. To do this, we heat the fastenings with the soldering tip. In addition, pin strips can be soldered, which are usually included. We only need the side with 7 pins (right):



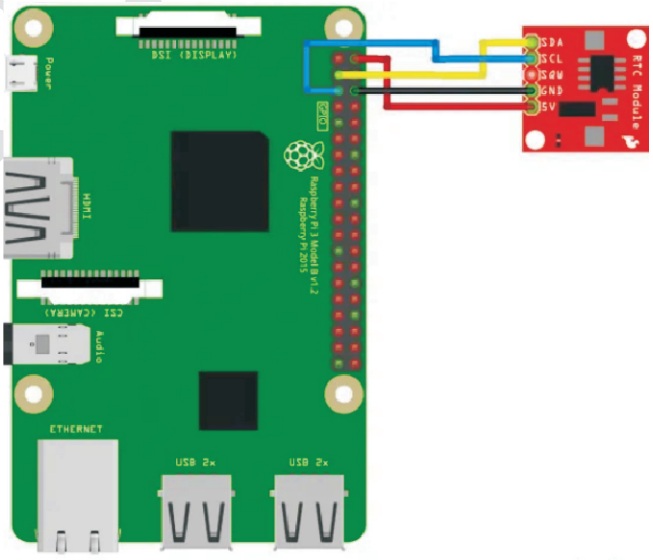
The resistors R2 and R3 must be removed not to damage the Raspberry Pi with a too high voltage.

Connection of the Raspberry Pi RTC I2C Module

The module is connected via I²C interface. For this, we use the right pin side (which has 7 pins), because in the offline mode of the Raspberry the current is to be drawn from the battery so that the clock does not stop. Other modules may only have a pin strip. The assignment is as follows:

RTC Module	
SCL	GPIO 3 / SCL (Pin 5)
SDA	GPIO 2 / SDA (Pin 3)
VCC / 5V	5V (Pin 2)
GND	GND (Pin 6)

The RTC Modules differ slightly from the structure, so the following picture should only be taken as a guide. It is important that the connected pins are correct:



Raspberry Pi RTC Software

Before we can really get started, it must be ensured that all packages and package sources are up to date, so we update them first and then install the I2C software:

sudo apt-get update && sudo apt-get upgrade --yes

sudo apt-get install i2c-tools

Afterwards, the I2C bus has to be activated, if that's not already done:

sudo raspi-config

Activate everything under “Advanced Options”> “I2C” (simply confirm with Yes). A restart may be necessary.

Now we edit the modules file

sudo nano /etc/modules

and add the missing entries at the end:

i2c-bcm2708

i2c-dev

rtc-ds1307

You save and exit with CTRL + O, CTRL + X.

To activate the modules, they must be loaded:

sudo modprobe i2c_bcm2708

sudo modprobe i2c_dev

sudo modprobe rtc-ds1307

We can now see whether the RTC module was recognized by I2C(the parameter –y 1 indicates that it is Rev.2. Only that the first Raspberry Pi corresponds to Revision 1):

i2cdetect -y 1

You should see the following output:

```
pi@raspberrypi:~ $ i2cdetect -y 1
 0 1 2 3 4 5 6 7 8 9 a b c d e f
00: -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

The module is therefore recognized and can be queried using i2cget -y 1 0x68. Since a hex code is difficult to read, we enter the module as a new I2C device:

sudo bash

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device

exit

Then we can simply read the time using **sudo hwclock -r.** You can get the local time of the system with date.

For me, the real-time clock was not set correctly (January 1, 2000), which is why I had to set it first. Since the local system time is correct (automatically accessed via an NTP server), I have synchronized it as follows (by the way, all commands can be found in the hwclock documentation):

sudo hwclock --set --date="$(date "+%m/%d/%y %H:%M:%S")"

If you want to change the system time (and time zone!), You should do this using **sudo raspi-config**, before you synchronize the times.

In order to set the system time automatically with every restart, we have to write a set command in the autostart. To do this, we edit the file:

sudo nano /etc/rc.local

The following two lines are added before exit 0:

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device

hwclock --hctosys

After a restart, the correct time should now be read and set from the Raspberry Pi RTC module – without any internet connection.

Interfacing 16x2 LCD Display with Raspberry Pi

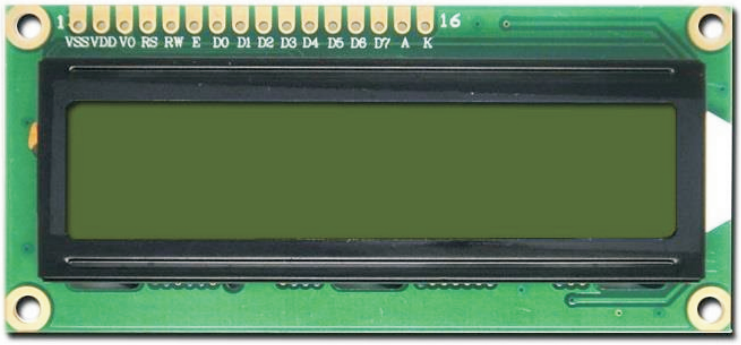
Building the 16×2 LCD Display Circuit:

1. To start our tutorial, we will first begin setting up the 16×2 LCD. We will be quickly running through the process of setting this all up.

For this section of the tutorial make sure that you have the following ready to go.

- 8 pieces of Male to Male Breadboard Wire
- 8 pieces of Male to Female Breadboard Wire
- 16×2 LCD Display
- 10k Ohm Potentiometer
- 330 Ohms Resistor
- Breadboard

2. Once you have all the parts required, you can start assembling the circuit by observing the table and steps given below.



Connecting the LCD to your Raspberry Pi is a pretty simple process if you follow our guide. We have included the physical pin number for each connection that you need to make. To begin with, let's connect up our various components with the breadboard.

- Raspberry Pi 5V (Physical Pin 4) to LCD pin 2 and one side of the potentiometer.
- LCD pin 15 to Raspberry Pi 5V (Physical Pin 4) through 330 Ohms Resistor.
- Raspberry Pi GND to LCD pin 1, 5, 16, and the opposite side of the potentiometer.
- Potentiometer output (middle pin) to LCD pin 3.
- Raspberry Pi GPIO26 (Physical Pin 37) to LCD pin 4.
- Raspberry Pi GPIO19 (Physical Pin 35) to LCD pin 6.
- Raspberry Pi GPIO25 (Physical Pin 22)t o LCD pin 11.
- Raspberry Pi GPIO24 (Physical Pin 18) to LCD pin 12.
- Raspberry Pi GPIO22 (Physical Pin 15) to LCD pin 13.
- Raspberry Pi GPIO27 (Physical Pin 13) to LCD pin 14.

Testing the 16×2 LCD Display:

1. Now that the circuit has been set up let's go ahead and test it to ensure that everything was wired correctly.

To start, You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Once done from your command line run the following command:

sudo pip3 install adafruit-circuitpython-charlcd

2. To demonstrate the usage of the character LCD we'll initialize it and display text using Python code. Create a new python file "lcd.py" at /home/pi and copy the following code in it.

PYTHON CODE:

```
#!/usr/bin/python
import time
import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd

# Define LCD column and row size for 16x2 LCD.
lcd_columns = 16
lcd_lines   = 2

# Raspberry Pi pin configuration:
lcd_rs       = digitalio.DigitalInOut(board.D26)
lcd_en       = digitalio.DigitalInOut(board.D19)
lcd_d4       = digitalio.DigitalInOut(board.D23)
lcd_d5       = digitalio.DigitalInOut(board.D24)
lcd_d6       = digitalio.DigitalInOut(board.D22)
lcd_d7       = digitalio.DigitalInOut(board.D27)
lcd_backlight = digitalio.DigitalInOut(board.D4)

# Initialize the LCD Class
lcd = characterlcd.Character_LCD_Mono(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_lines, lcd_backlight)

# Print a two line message
lcd.message = "Edkits\nElectronics"
time.sleep(5)
lcd.clear()
```

3. Now before we go ahead and run our new lcd.py example we will need to install the Raspberry Pi's GPIO Python library.

To install the required library run the following command.

sudo pip3 install RPi.GPIO

4. To test that everything is working lets now run that python script by running the command below. If everything is working as it should, you should now see text displayed across your LCD.

pi@raspberrypi:~ $ **sudo python3 lcd.py**