Raspberry pi

- FLASH RASPBERRY PI OS ON YOUR MICROSD CARD
  - GO TO RASPBERRYPI.ORG
  - DOWNLOAD IMAGER
  - CHOOSE DEVICE , SD CARD AND CLICK WRITE
    (EVERYTHING IN THE SD CARD WILL BE FORMATTED)
- SETUP WIFI AND SSH DIRECTLY ON THE MICRO SD CARD
  - CLICK SD CARD
  - CREATE NEW TEXT FILE
  - NAME IT wpa_supplicant.conf
  - PASTE THIS IN THAT TEXT FILE

    country=US
    ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
    update_config=1

    network={
    scan_ssid=1
    ssid="your_wifi_ssid"
    psk="your_wifi_password"
    }

  - Create new text file
  - Name it ssh
- BOOT YOUR RASPBERRY PI FOR THE FIRST TIME AND FIND ITS IP ADDRESS
  - MAKE SURE PI IS TURNED OFF
  - PUT SD CARD IN SD CARD SLOT OF RASPBERRY PI
  - USE CHARGER FOR BETTER

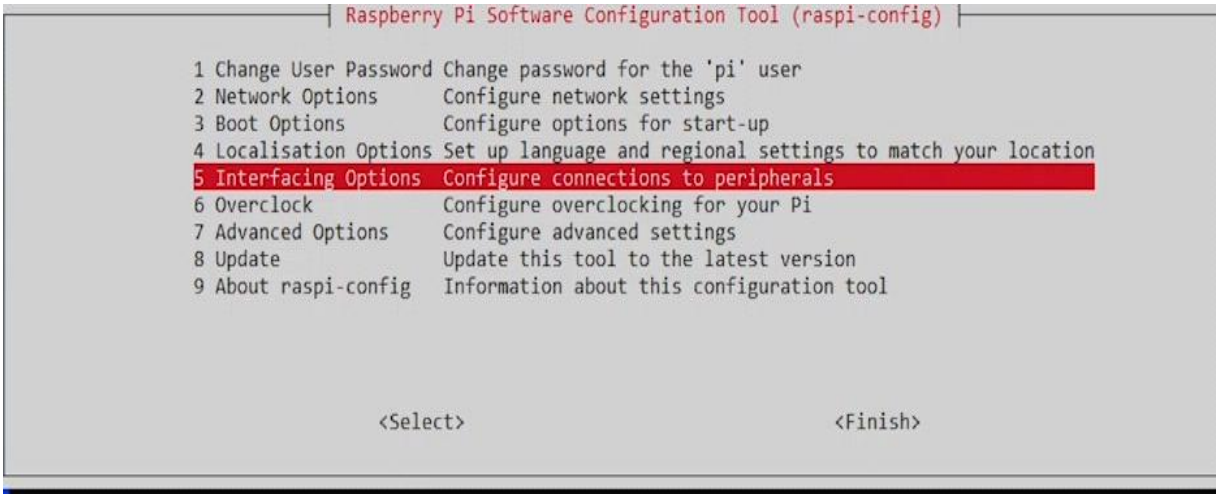
- DOWNLOAD ANGRYIP SCANNER
- MAKE SURE RASPBERRY PI AND PC IS CONNECTED TO SAME WIFI
- CLICK IP AND CHOOSE THE ONE WITH WIFI
- CHANGE IP RANGE TO .1 AND .255
- SELECT MAC Vendor fetcher
- Press start
- See mac vendor column : the one with raspberry pi trading is the ip address of raspberry pi

- CONNECT TO YOUR PI USING SSH

  - SEE IF ANY .ssh FILES ARE AVILABLE
  - IF ANY REMOVE THEM
  - OPEN COMMAND PROMPT
  - TYPE ssh pi@ip_address_of_raspberry
  - PRESS ENTER AND TYPE YES
  - IT ASKS FOR PASSWORD
  - TYPE your_password and press enter

- SETUP VNC TO GET REMOTE ACCESS TO RASPBERRY PI OS

  - TYPE shh pi@raspberry_pi_ip_address
  - TYPE password_of_raspberry_pi
  - TYPE sudo raspi-config



- CLICK ENTER ON INTERFACING OPTIONS AND THEN VNC
- PRESS YES
- PRESS ENTER IN FINISH

- TYPE sudo reboot
- TYPE shh pi@raspberry_pi_ip_address
- TYPE password_of_raspberry_pi
- TYPE sudo raspi-config

- GO TO BOOT OPTIONS
- SELECT DESKTOP/CLI
- SELECT DESKTOP OR DESKTOP AUTOLOGIN

- ASKS TO REBOOT CLICK YES

- TYPE sudo reboot
- TYPE shh pi@raspberry_pi_ip_address
- TYPE password_of_raspberry_pi
- TYPE sudo raspi-config

- GO TO ADVANCED OPTIONS
- CHOOSE SCREEN RESOLUTION
- CHOOSE RESOLUTION OF LAPTOP

- ASKS TO REBOOT CLICK YES

- GO TO realvnc.com
- DOWNLOAD FOR WINDOWS
- LAUNCH THE APPLICATION

- CLICK ON NEW CONNECTION
- ON VNC SERVER PUT ip_address_of_raspberry_pi
- GIVE USERNAME AND PASSWORD OF RASPBERRY PI

- FINISHING THE STARTUP CONFIGURATION

  - ENTER CONTRY , LANGUAGE AND TIME ZONE
  - ENTER NEW PASSWORD AND CONFIRM IT
  - GO TO PREFERNECES AND THEN INTERFACES
  - ENABLE DISABLE SSH CAMERA ETC

  - TYPE sudo nano/boot/config.txt
  - ADD # IN dtoverlay
  - PRESS CTRL S

  - TYPE sudo reboot

- YOUR FIRST PYTHON PROGRAM

  - CLICK ON MENU THEN PROGRAMMING THEN THONNYPYTHON IDE
  - CLICK ON SWITCH TO REGULAR MODE
  - CLOSE AND REOPEN IT AGAIN

  - print("Hello Rpi") and press run

- VARIABLES
  - CONTAINER WHERE YOUU STORE INFORMATION TO USE LATER
  - EG : a =1
    print(a)

- VARIABLE DATA TYPES
  - print(type(a))
    gives the data type of variable

- FUNCTIONS
  - REUSABLE BLOCK OF CODE THAT YOU CAN CALL FROM ANOTHER PART OF PROGRAM
  - EG : def triple_number(number):
  - return number*3
  - a = 2
  - b = triple_number(a)

- FUNTION TO CONCATENATE TWO UPPERCASE STRINGS

```
def conc_upp_str(str_a , str_b):
    new_str = str_a.upper() + " " + str_b.upper()
    return new_str
```

```
result = conc_upp_str("Hello" , "World")
print(result)
```

- CONDITIONS
  - IF AND ELSE

- OPERATORS
  - < , <= , > , >= , == , != , or , and

- VALIDATE USER INPUT

```
number = int(input("Enter a number between 1 and 100: "))
if number <=0 or number > 100:
    print("Error: Number out of range." + str(number))
else:
    print("Number accepted: " + str(number))
```

- LOOPS
  - for , while

- LISTS
  - COLLECTION OF VARIABLES

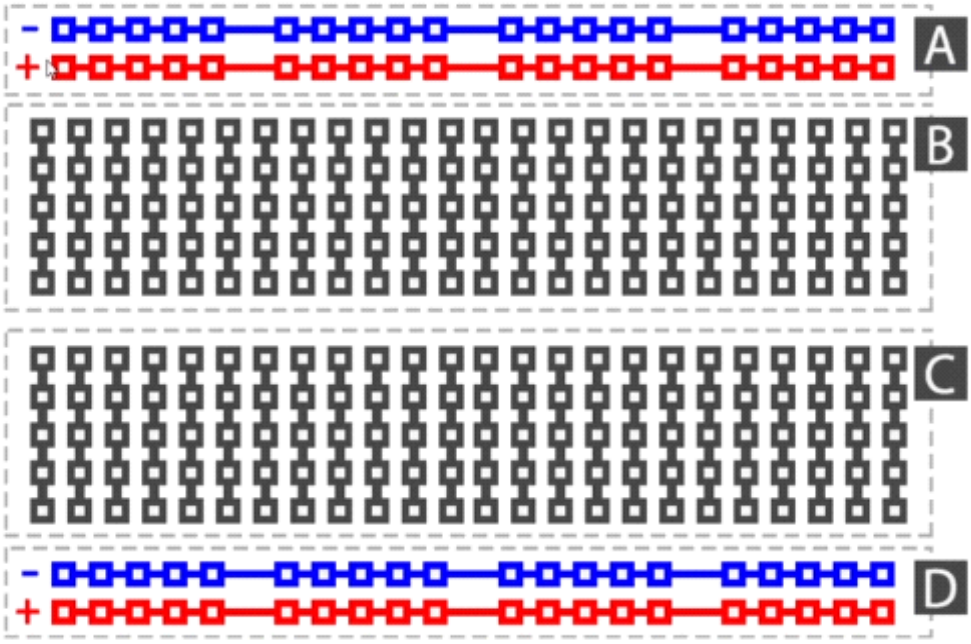
- COMPUTE MAX VALUE INSIDE A LIST

```
def find_maximum(number_list):
    max_value = 0
    for number in number_list:
        if number > max_value:
            max_value = number
    return max_value

number_list = [3, 5, 78, -7, 45]
max_value = find_maximum(number_list)
print("The maximum value is:"+ str(max_value))
```

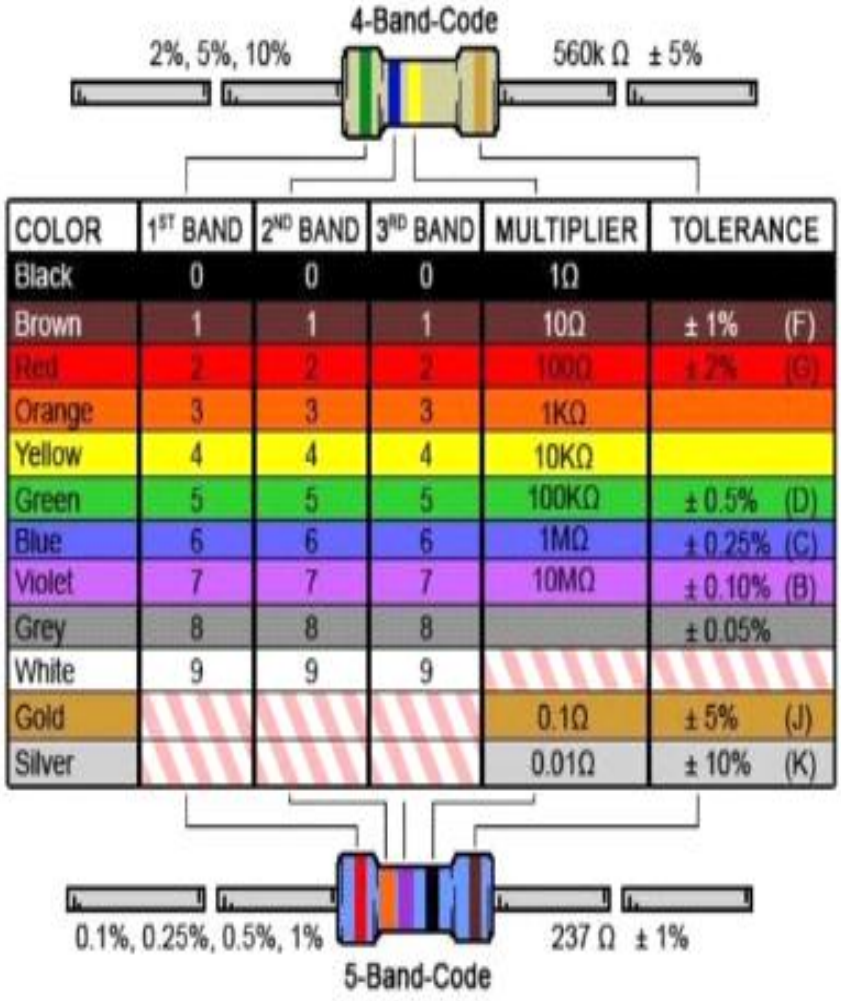
- RASPBERRY PI AND GPIOS - WARNING
  - ALWAYS POWER OFF THE PI FIRST
  - WHEN POWERED ON , AVOID TOUCHING THE CIRCUIT
  - DOUBLE CHECK EVERYTHING BEFORE PUTTING THE POWER CABLE
  - START THE CIRCUIT BY CONNECTING GROUND FOR ALL COMPONENTS
  - DON’T PUT 5V SIGNAL ON GPIO PIN (3.3V MAX)

- UNDERSTAND HOW A BREADBOARD WORKS

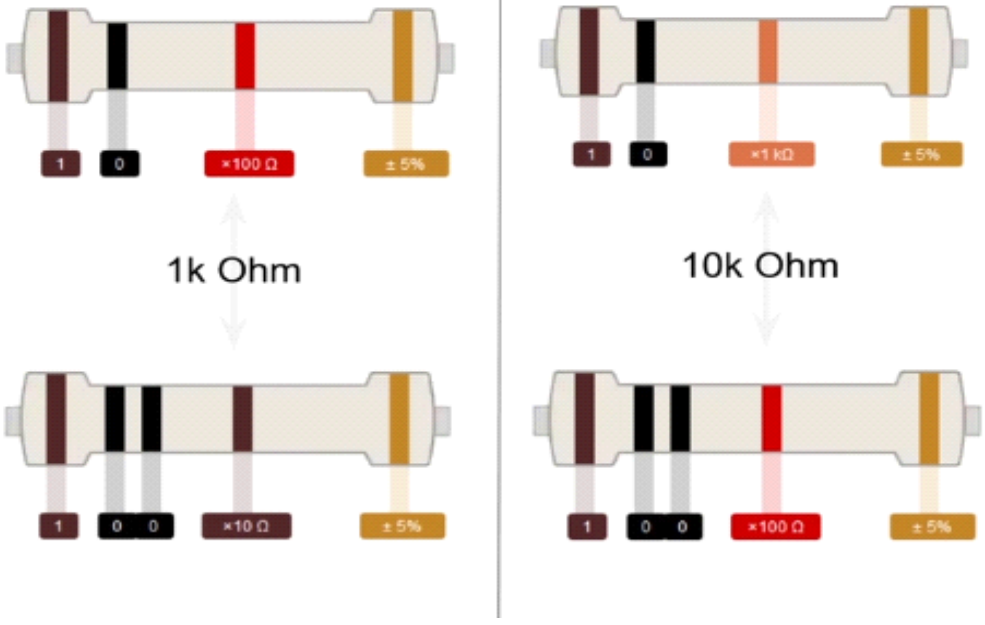
- 

  - CONNECTED IN A WAY THE LINE IS SHOWN / SAME COLUMN
  - \+ IS POWER SUPPLY
  - \- IS GROUND

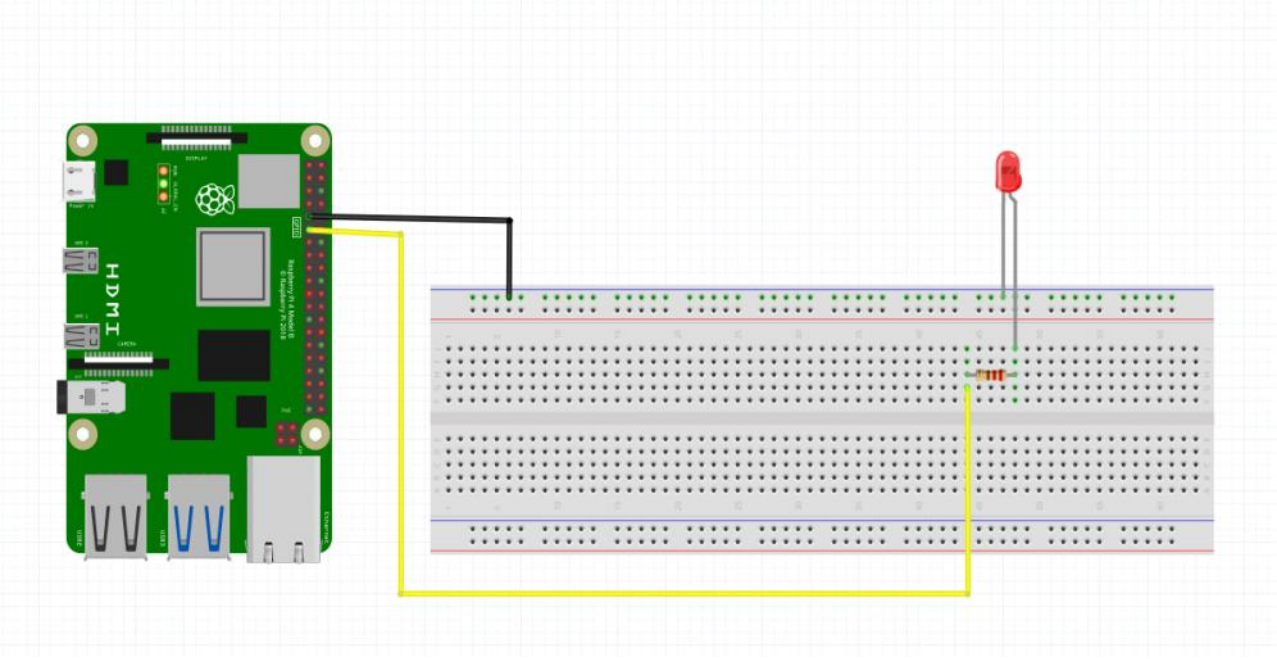
- RESISTOR COLOR CODE



COLOR	1ST BAND	2ND BAND	3RD BAND	MULTIPLIER	TOLERANCE
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1% (F)
Red	2	2	2	100Ω	± 2% (G)
Orange	3	3	3	1KΩ	
Yellow	4	4	4	10KΩ	
Green	5	5	5	100KΩ	± 0.5% (D)
Blue	6	6	6	1MΩ	± 0.25% (C)
Violet	7	7	7	10MΩ	± 0.10% (B)
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1Ω	± 5% (J)
Silver				0.01Ω	± 10% (K)

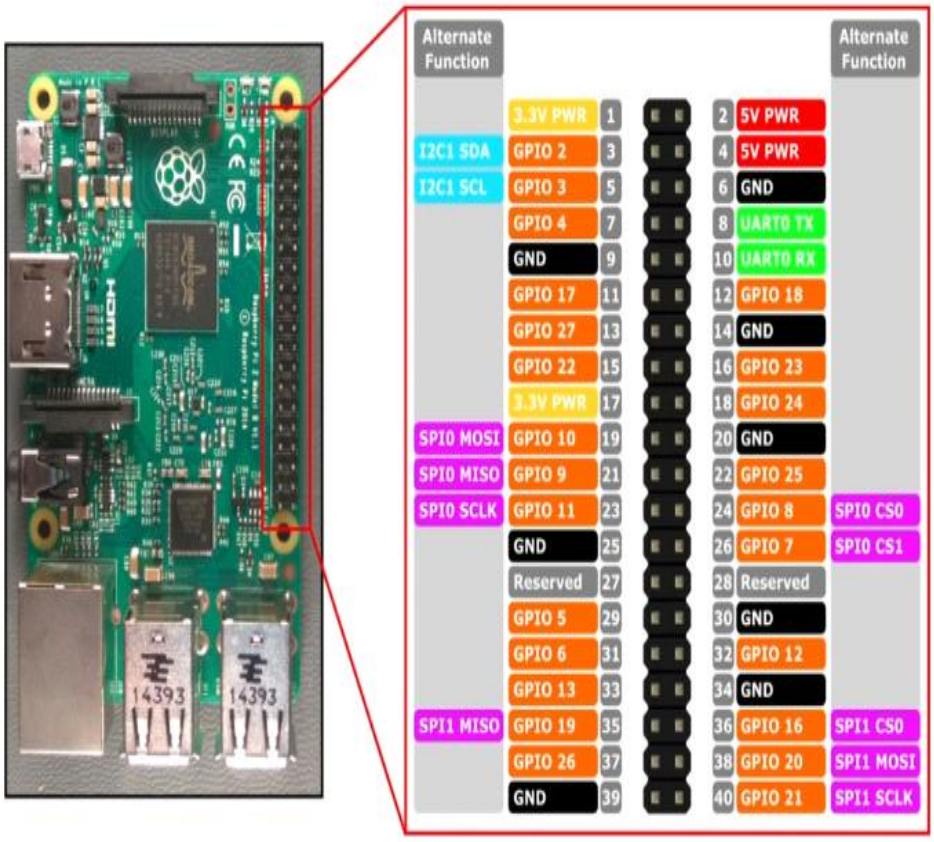


- CURRENT CONSUMPTION LIMIT OF RASPBERRY PI : 50 mA
- HIGHER THE RESISTANCE , LESSER THE CURRENT YOU'LL TAKE

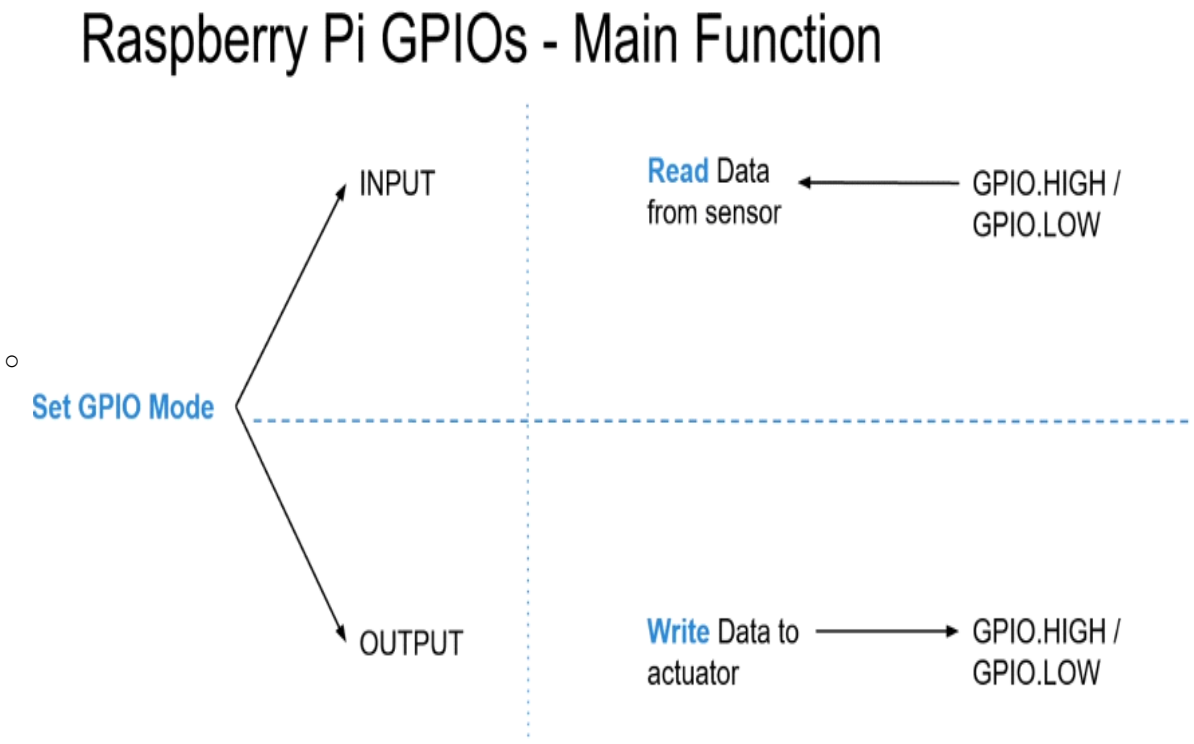
- FIRST CIRCUIT - 1 LED AND 1 RESISTOR
  - 

- HOW GPIOS WORK

Raspberry Pi GPIOs - Pinout

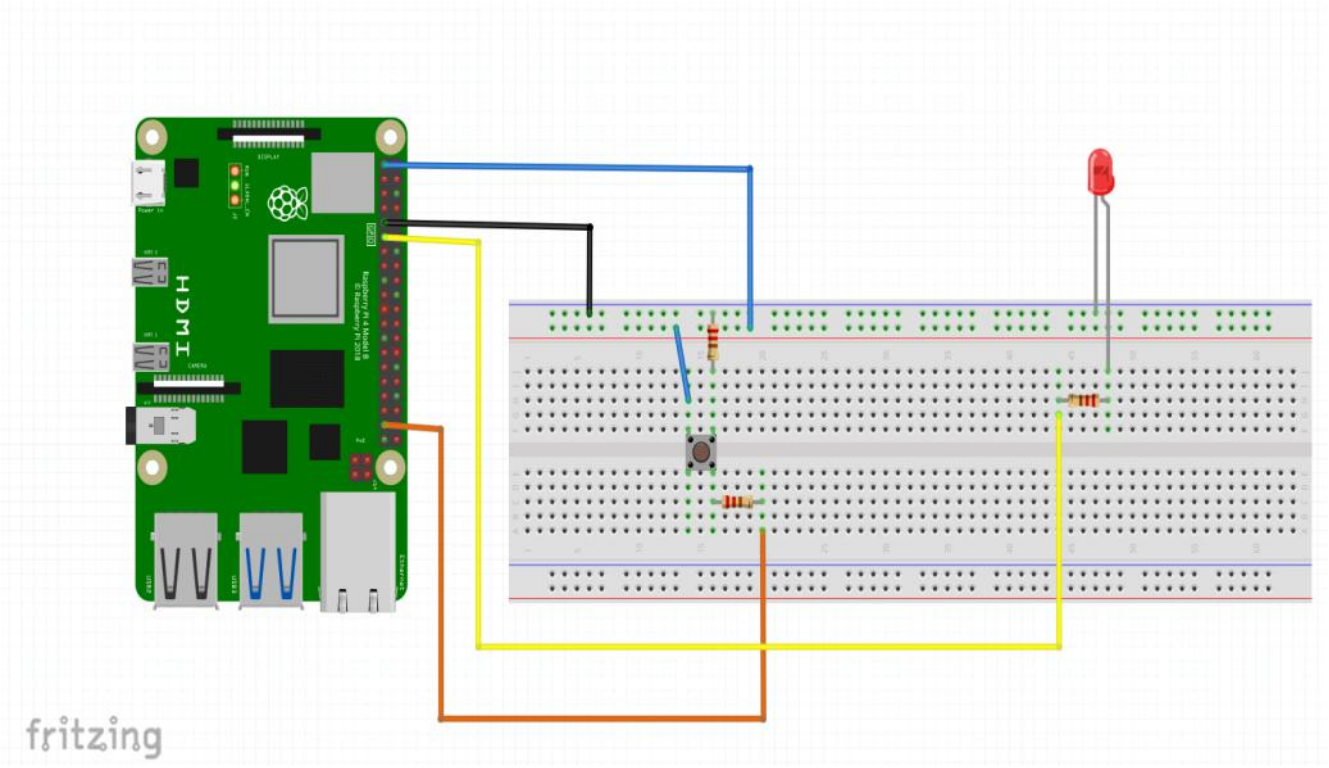


- GPIO NUMBER IS THE NUMBER YOU WILL USE IN YOUR PYTHON CODE TO INTERACT WITH THE GPIO
- GPIO ARE 3.3 V PIN

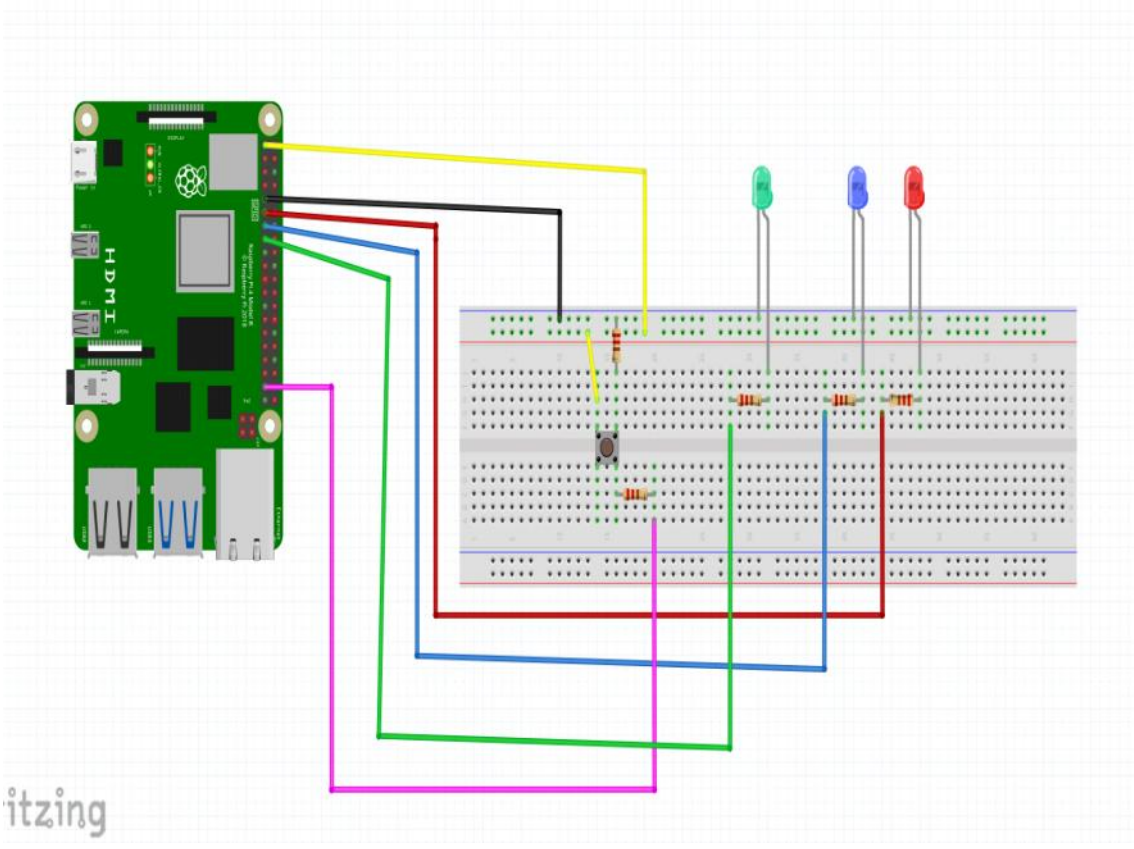


- RESERVED PINS SHOULD NOT BE USED
- TWO PINS IN GREEN CAN BE USED FOR UART COMMUNICATION
- GPIO 2 AND 3 CAN BE USED FOR I2C COMMUNICATION
- ONE WITH PINK LABEL CAN BE USED FOR SPI COMMUNCIATION

- ADD A PUSH BUTTON TO YOUR CIRCUIT

- 

- ADD TWO LEDS TO YOUR CIRCUIT

- 

- PIR SENSOR

- PIR - PASSIVE INFRARED SENSOR
- MOTION DETECTOR
- USED FOR AUTOMATIC LIGHTING , SECURITY ALARMS , ETC

- TUNE THE PIR SENSOR
  - 1ST POTENTIOMETER INCREASE RANGE BY GOING CLOCKWISE
  - 2ND POTENTIOMETER MAKE MINIMUM BY GOING ANTI CLOCKWISE

- USE TERMINAL ON YOUR RASPBERRY PI

- NAVIGATION AND FILE SYSTEM
  - ls - LIST ALL FOLDERS FROM WHERE YOU ARE
  - pwd - TELLS WHERE YOU ARE RIGHT NOW
  - cd folder_name - GO TO THAT FOLDER
  - cd .. - GO TO PREVIOUS FOLDER
  - cd folder_name/folder_name - GO INSIDE ANOTHER FOLDER OF PREVIOUS FOLDER
  - ls -a - SHOWS THE HIDDEN FILES
  - cd / - TAKES TO ROOT FOLDER

- EDIT FILES ON TERMILA WITH NANO
  - nano file_name - YOU ENTER NANO EDITOR AND CREATE NEW FILE

- cat file_name - ALLOWS YOU TO PRINT THE FILE CONTENTS IN TERMINAL

- nano ~/.nanorc - PROVIDE CONFIGUATATIONS FOR NANO EDITOR

- CONFIGURATIONS FOR EASY PYTHON PROGRAMMING

  - set tabsize 4
  - set tabstospaces

- CREATE , REMOVE AND MANIPULATE FILES

  - touch file_name - CREATE FILE
  - rm file_name - REMOVE FILE
  - mkdir folder_name - CREATE FOLDER
  - rm -rf folder_name - DELETE FOLDER
  - mv previous_name new_name - RENAME FILE
  - mv file_name ~/folder_name/new_name - MOVE FILE INSIDE FOLDER AND RENAME
  - cp file_name new_file_name - COPY THE FILE AND CREATE NEW

- INSTALL AND UPDATE SOFTWARE

  - sudo apt update - CHECKS PACKAGES WHICH NEED TO BE UPDATED
  - sudo apt install software_name - TO DOWNLOAD NEW SOFTWARE
  - sudo apt remove software_name - TO REMOVE THE SOFTWARE
  - sudo apt upgrade - UPGRADES THE PACKAGES WHICH

NEED TO BE UPGRADED

- MORE COMMANDS
  - hostname -I - TO KNOW THE IP ADDRESS
  - df -h - INFORMATION ABOUT AVAILABLE SPACE IN SD CARD
  - sudo raspi-config - SET UP DIFFERENT OPTIONS
  - sudo shutdown now - SHUTDOWN RASPBERRY PI
  - sudo reboot - REBOOT RASPBERRY PI

- INSTALL PYTHON MODULES
  - pip3 install python_module_name - INSTALL PYTHON MODULE
  - pip3 list - LIST ALL PYTHON MODULES INSTALLED
  - pip3 list | grep words_to_find_in_module - FIND MODULE FROM KEYWORDS
  - pip3 help - COMMANDS AND OPTIONS YOU CAN USE
  - pip3 uninstall python_module_name - UNINSTALL PYTHON MODULE

- WORK WITH PYTHON FROM THE TERMINAL
  - python3 - GET PYTHON SHELL
  - CTRL + C AND exit() - GET OUT OF SHELL
  - python3 file_name - EXECUTE THE PYTHON FILE ON TERMINAL DIRECTLY
  - touch test.py - CREATE NEW PYTHON FILE

- READ , WRITE AND MANIPULATE FILES WITH PYTHON
  - with open ("path_of_file" , "r") as f:
  - print(f.read) - READ THE FILE

  - with open ("path_of_file" , "w") as f:
  - f.write("new text") - WRITE IN THE FILE

  - with open ("path_of_file" , "w+r") as f:
  - f.write("new text") - READ AND WRITE IN THE FILE

  - with open ("path_of_file" , "a") as f:
  - f.write("new text") - APPENDS IN THE FILE

  - import os
  - if os.path.exists("location_of_file"):
  - print("File exists") - CHECKS IF FILE IS THERE OR NOTT

  - import os
  - if os.path.exists("location_of_file"):
  - print("File exists")
  - os.remove("location_of_file") - REMOVE THE FILE

- SEND EMAIL FROM YOUR RASPBERRY PI
  - CREATE NEW GMAIL
  - GO TO SECURITY AND ENABLE ALLOW LESS SECURE APPS

- INSTALL PYTHON MODULE YAGMAIL
  - GO TO TOOLS AND THEN MANAGE PACKAGES
  - TYPE YAGMAIL AND ENTER
  - CLICK ON INSTALL AND WAIT IT IS INSTALLED

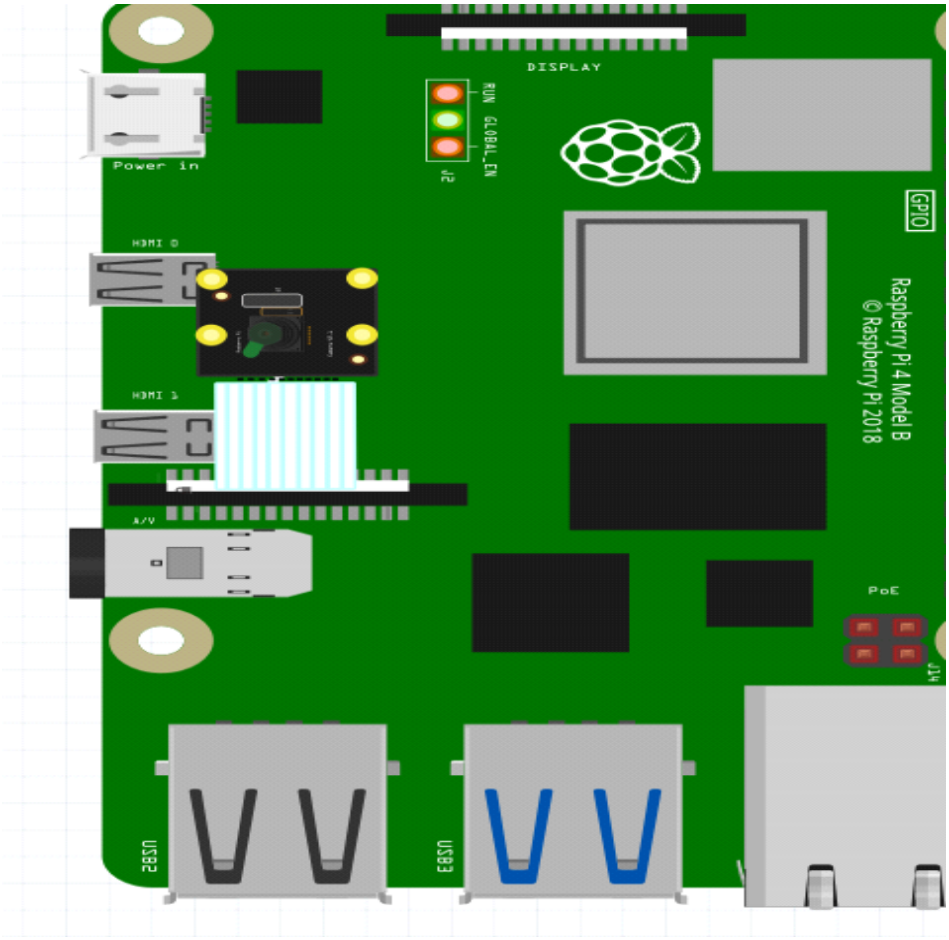
  - OR OPEN TERMINAL AND TYPE
  - pip3 install yagmail

- GET PASSWORD IN YOUR PYTHON PROGRAM
  - OPEN TERMINAL AND TYPE ls -la
  - TYPE cd.local/share/
  - TYPE touch .email_password
  - TYPE ls -la
  - TYPE nano .email_password
  - COPY AND PASTE PASSWORD
  - SAVE AND CLOSE THE FILE

- WORK WIH PI CAMERA
  - STANDARD (GREEN) - STANDARD CAMERA BETTER FOR DAYLIGHT
  - NoIR (BLACK) - BETTER FOR LOW LIGHT AND TO SEE IN THE DARK

- PLUG THE CAMERA TO RASPBERRY PI
  - CLICK ON PLASTIC PART OF HOLDER OF CAMERA
  - BLUE INDICATION OF CAMERA WILL FACE PORTS

- AGAIN CLICK ON PLASTIC PART OF CAMERA HOLDER



- ENABLE THE CAMERA
  - USE THE OPTIONS OF RASPBERRY AND ENABLE CAMERA
  - IN TERMINAL
    - sudo raspi-config
    - go to interface options
    - go to camera and enable
    - click finish and then reboot

- TAKE A PHOTO FROM TERMINAL

- mkdir camera - CREATE NEW FOLDER NAMED CAMERA
- raspistill -o file_name - CLICK THE PHOTO AND SAVE AS file_name IN HOME DIRECTORY
- raspistill -o camera/file_name.jpg - CLICK THE PHOTO AND SAVE IT IN CAMERA DIRECTORY
- raspistill -o camera/file_name.jpg -rot 180 - ROTATE IF NOT IN GOOD ORIENTATION
- raspistill -o camera/file_name.jpg -w width_size -h height_size - TO CHANGE THE

  RESOLUTION

- RECORD A VIDEO FROM TERMINAL

  - raspivid -o camera/file_name.h264 - RECORD VIDEO OF 5 SEC
  - raspivid -o camera/file_name.h264 -t time_in_millisec - RECORD VIDEO OF CUSTOM TIME

  - OTHER COMMANDS SAME AS IMAGE

- The first Python program will:

  - Monitor the PIR sensor. When some movement is detected, make sure the movement is detected for 3 consecutive seconds. To do that you can create an infinite while loop.You'll have to keep the last PIR sensor state into a variable and compare it with the current state. As an additional visual indicator you can also power on an LED when the PIR sensor detects some movement.

  - At this point, take a photo and save it to the /home/pi/camera folder. Create a function "take_photo()" to take a photo and save it. The file name starts with "img_" and then has a timestamp to make it unique. To get a timestamp you can call the function "time.time()". Then call this function in the appropriate place in

your main while loop.

- Then, add the name of the photo (complete path + file name) into a log file named photo_logs.txt (use the append mode with “a” when opening the file). During the setup of the program, check if this file exists and remove it so it doesn’t contain data from previous program runs.

- Then, send an email to yourself with the photo as an attachment. Setup email (with Yagmail) in the setup section of the program. Create a “send_email_with_photo” function to send the photo which was just taken by email.

- Also, as an additional feature, make sure you don’t take more than one photo (+ write to log file and send email) every 60 seconds. For that you can also keep a “last_time_photo_taken” variable with the time.

- SECOND PYTHON PROGRAM WILL BE

  - Run a Flask application with a default route “/” just returning “Hello”

  - Add another route “/check-movement”

  - When the function for this URL is called, read the log file (photo_logs.txt created by the other Python program, using the read mode “r”). Count the number of lines in the file,and get the last photo path (last line). You can use a “for line in f” loop to go through all the lines in the file.

  - Then, compare the line count with the previous line count (= the last time you called this URL). For that you’ll need to use a global variable to keep track of the count.

- Return a string telling how many new photos have been taken since the last time the URL was called, + give the path to the photo file.

- And finally, use an <img src=”photo_path”> tag inside the string you return to print the image in the browser. To make things work you’ll also have to change the line “app = Flask(__name__)” to become “app = Flask(__name__, static_url_path=CAMERA_FOLDER_PATH, static_folder=CAMERA_FOLDER_PATH)”, where CAMERA_FOLDER_PATH is "/home/pi/camera".