Experiment No. 01

****

Title: Study of Raspberry-pi, Begal board , Audino and other microconroller.

Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

****

Raspberry Pi

Raspberry Pi is a small single board computer. By connecting peripherals like Keyboard, mouse, display to the Raspberry Pi, it will act as a mini personal computer.

Raspberry Pi is popularly used for real time Image/Video Processing, IoT based applications and Robotics applications.

Raspberry Pi is slower than laptop or desktop but is still a computer which can provide all the expected features or abilities, at a low power consumption.

Raspberry Pi Foundation officially provides Debian based Raspbian OS. Also, they provide NOOBS OS for Raspberry Pi. We can install several Third-Party versions of OS like Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, etc.

Raspbian OS is official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian have GUI which includes tools for Browsing, Python programming, office, games, etc.

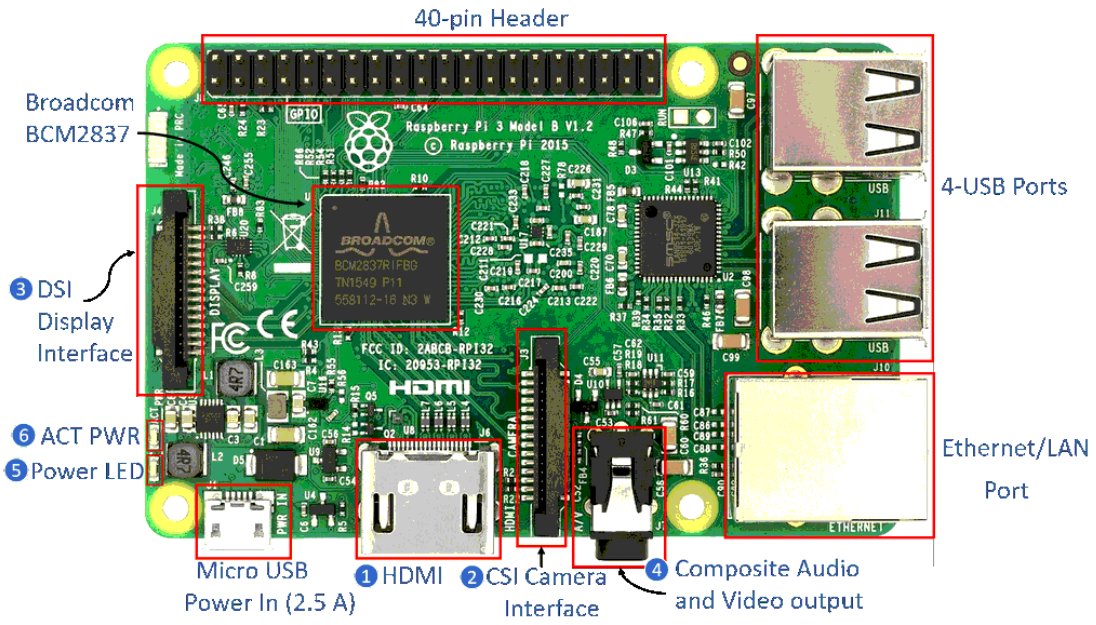
We should use SD card (minimum 8 GB recommended) to store the OS (operating System).

Raspberry Pi is more than computer as it provides access to the on-chip hardware i.e. GPIOs for developing an application. By accessing GPIO, we can connect devices like LED, motors, sensors, etc and can control them too.

It has ARM based Broadcom Processor SoC along with on-chip GPU (Graphics Processing Unit).

The CPU speed of Raspberry Pi varies from 700 MHz to 1.2 GHz. Also, it has on-board SDRAM that ranges from 256 MB to 1 GB.

Raspberry Pi also provides on-chip SPI, I2C, I2S and UART modules.



Some Hardware Components shown above are mention below:

HDMI (High-Definition Multimedia Interface): It is used for transmitting uncompressed video or digital audio data to the Computer Monitor, Digital TV, etc. Generally, this HDMI port helps to connect Raspberry Pi to the Digital television.

CSI Camera Interface: CSI (Camera Serial Interface) interface provides a connection in between Broadcom Processor and Pi camera. This interface provides electrical connections between two devices.

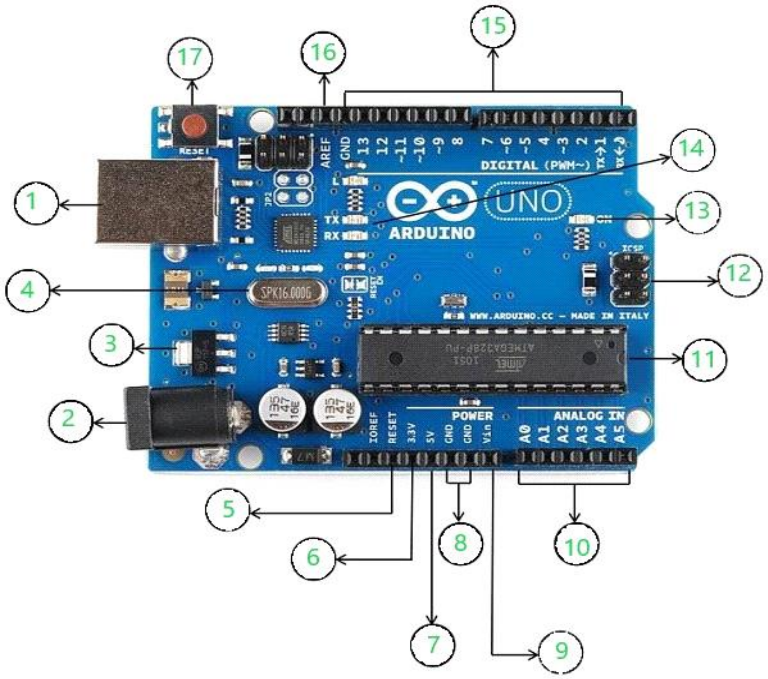
DSI Display Interface: DSI (Display Serial Interface) Display Interface is used for connecting LCD to the Raspberry Pi using 15-pin ribbon cable. DSI provides fast High-resolution display interface specifically used for sending video data directly from GPU to the LCD display.

Composite Video and Audio Output: The composite Video and Audio output port carries video along with audio signal to the Audio/Video systems.

Power LED: It is a RED colored LED which is used for Power indication. This LED will turn ON when Power is connected to the Raspberry Pi. It is connected to 5V directly and will start blinking whenever the supply voltage drops below 4.63V.

ACT PWR: ACT PWR is Green LED which shows the SD card activity.

AURDINO



Using the above image as a reference, the labeled components of the board respectively are-

USB: can be used for both power and communication with the IDE

Barrel Jack: used for power supply

Voltage Regulator: regulates and stabilizes the input and output voltages

Crystal Oscillator: keeps track of time and regulates processor frequency

Reset Pin: can be used to reset the Arduino Uno

3.3V pin: can be used as a 3.3V output

5V pin: can be used as a 5V output

GND pin: can be used to ground the circuit

Vin pin: can be used to supply power to the board

Analog pins(A0-A5): can be used to read analog signals to the board

Microcontroller(ATMega328): the processing and logical unit of the board

ICSP pin: a programming header on the board also called SPI

Power indicator LED: indicates the power status of the board

RX and TX LEDs: receive(RX) and transmit(TX) LEDs, blink when sending or receiving serial data respectively

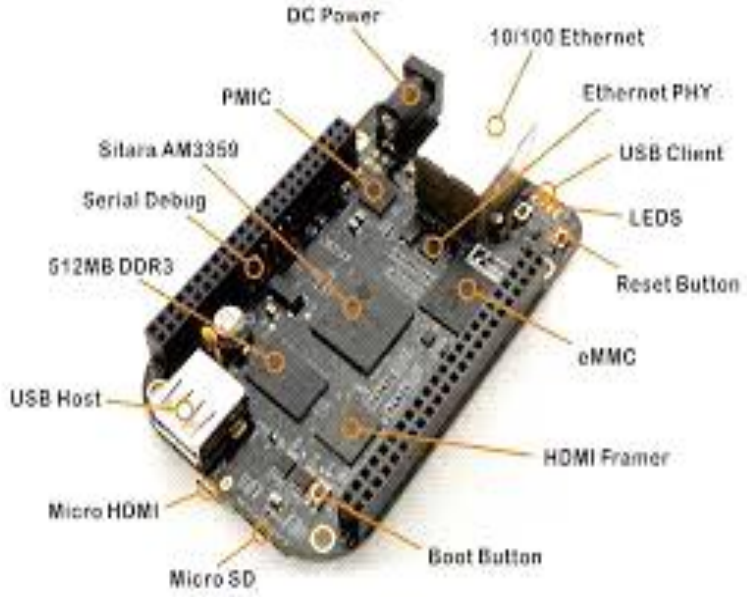
Digital I/O pins: 14 pins capable of reading and outputting digital signals; 6 of these pins are also capable of PWM

AREF pins: can be used to set an external reference voltage as the upper limit for the analog pins

Reset button: can be used to reset the board

BEGAL BOARD

The BeagleBoard is a single-board computer manufactured by Texas Instruments, introduced in 2008. It was designed for hobbyists and as an educational tool for the development of open source software. It uses an ARM Cortex-A8 CPU that runs at speeds up to 1 GHz, and can be configured with between 128 MB and 512 MB of RAM. It measures 7.5 mm on each side, and has all the functionality of a basic computer.



Conclusion:

Experiment No. 02

****

Title: Study of different operating systems fo raspberry-pi. Understanding the process of installation on raspberry-pi.

Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

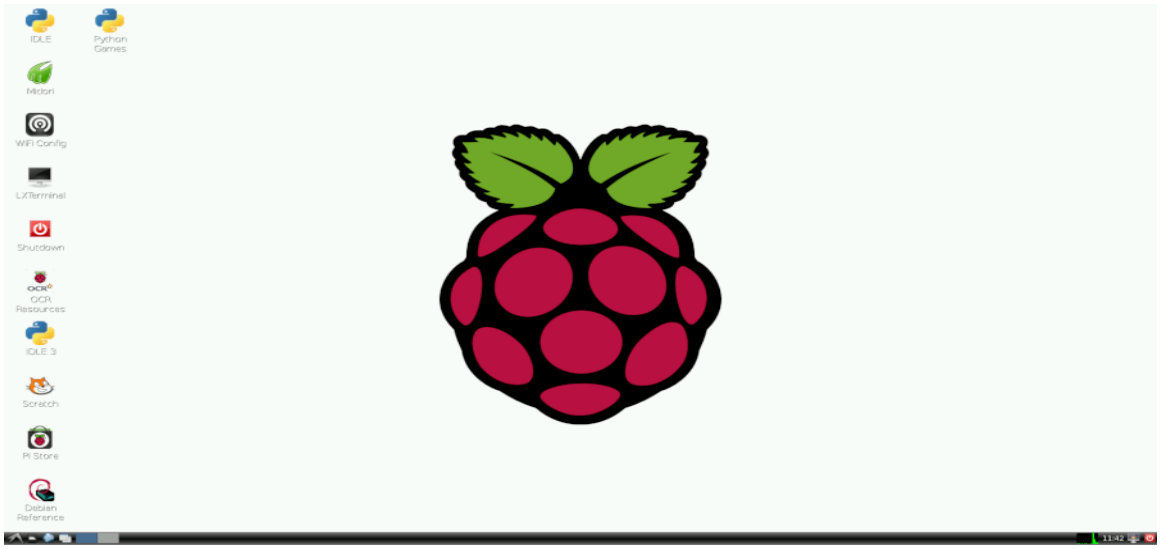
****

1. Raspbian

Raspbian is a Debian-based engineered especially for the Raspberry Pi and it is the perfect general-purpose OS for Raspberry users.

It employs the Openbox stacking window manager and the Pi Improved Xwindows Environment Lightweight coupled with a number of pre-installed software which includes Minecraft Pi, Java, Mathematica, and Chromium.

Raspbian is the Raspberry foundation's official supported OS and is capable of accomplishing any task you throw at it.

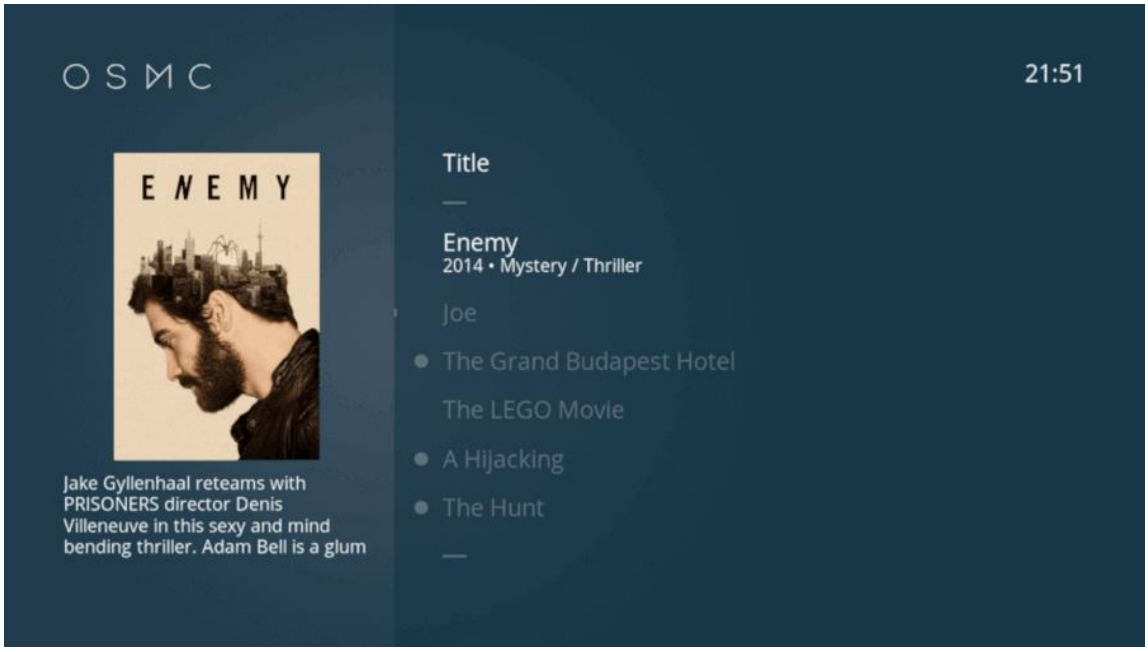


Raspbian is a Debian-based OS for Raspberry

2. OSMC

OSMC (Open Source Media Center) is a free, simple, open-source, and easy-to-use standalone Kodi OS capable of playing virtually any media format.

It features a modern beautiful minimalist User Interface and is completely customizable thanks to the several built-in images that it comes with. Choose OSMC if you run the Raspberry Pi for managing media content.

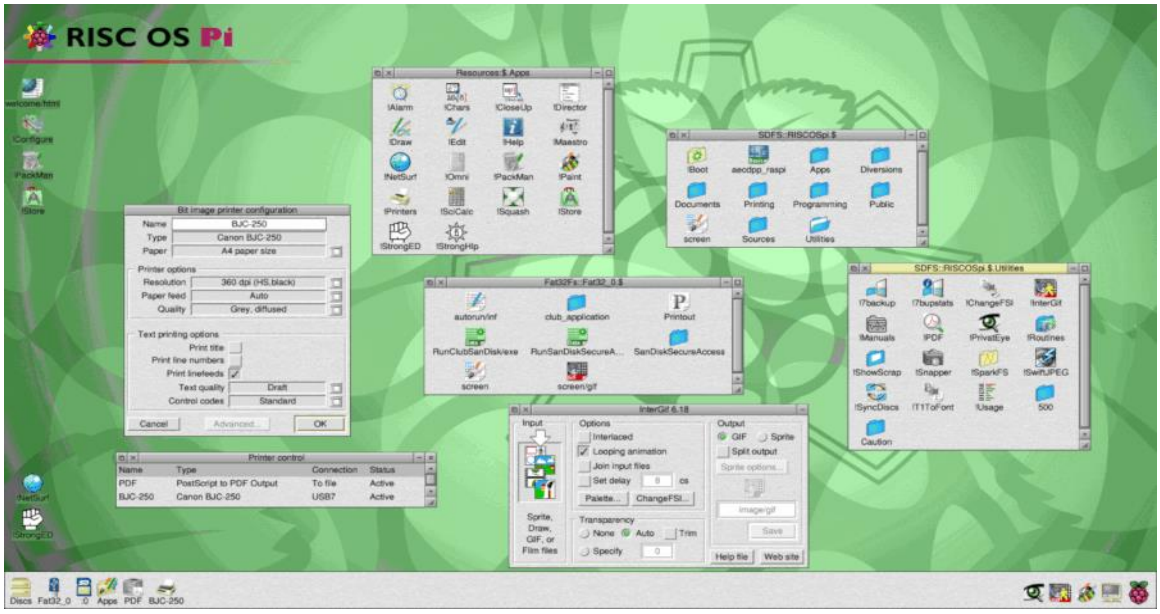


OSMC is a Kodi-centered Linux OS

3. RISC OS

RISC OS is a unique open-source OS designed specifically for ARM processors by the creators of the original ARM. It is neither related to Linux nor Windows and is being maintained by a dedicated community of volunteers.

If you want to choose RISC OS, you should know that it is very different from any Linux distro or Windows OS you have used so it will take some getting used to. A good place to start is here.

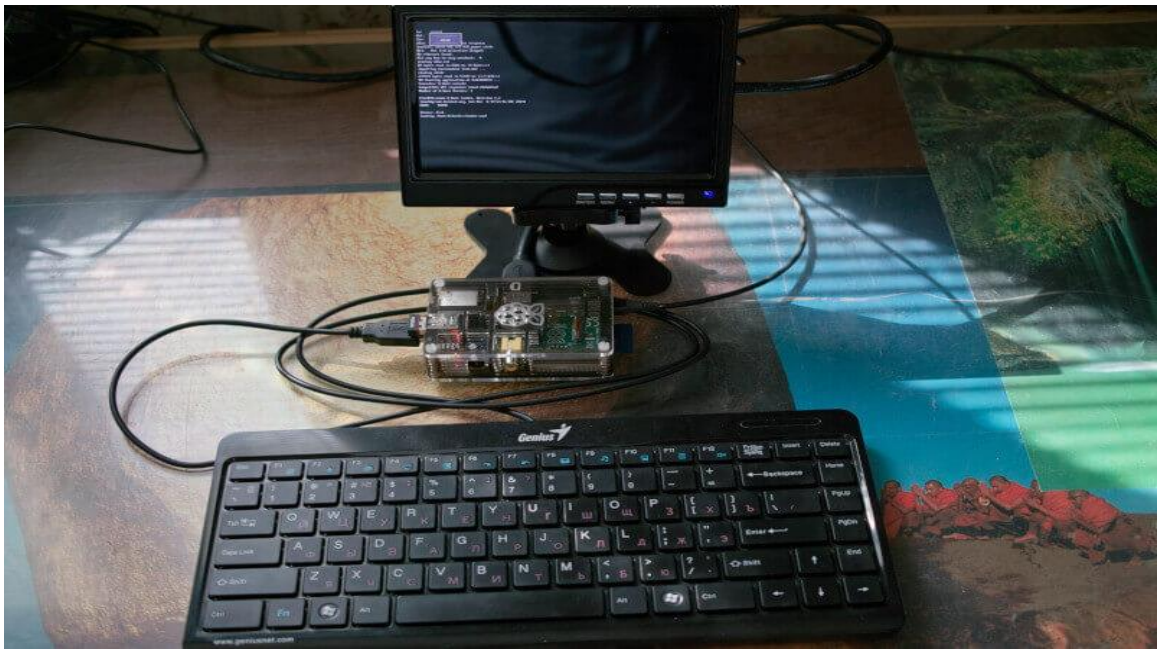


RISC OS for Raspberry Pi

4. RaspBSD

RaspBSD is a free and open-source image of FreeBSD 11 that has been preconfigured in 2 images for Raspberry Pi computers.

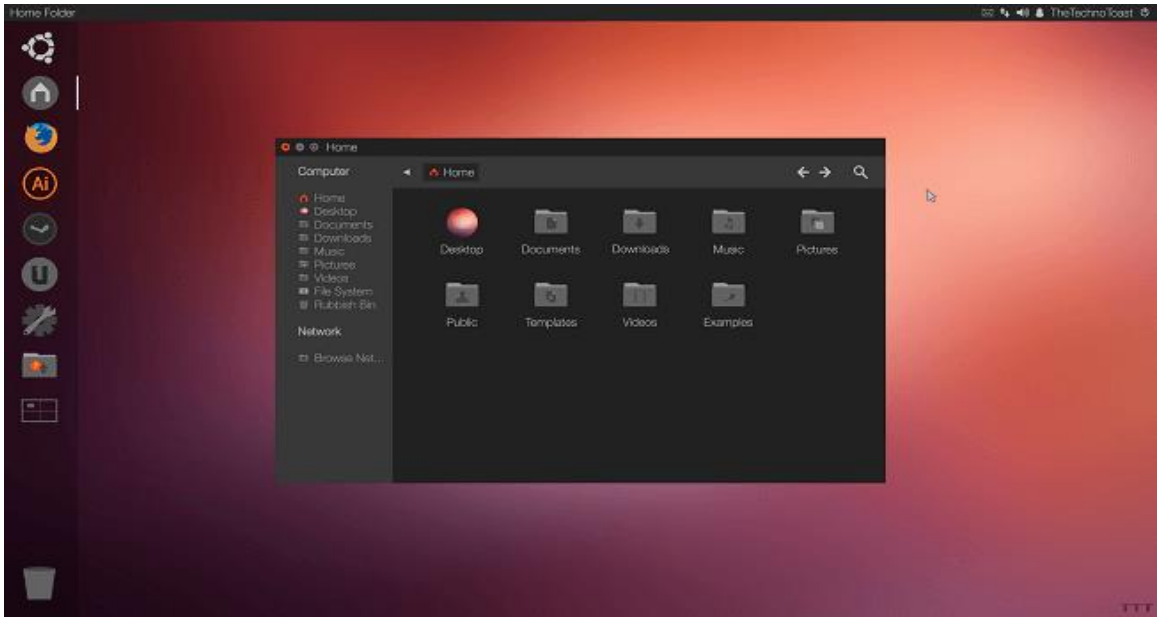
If you didn’t know, FreeBSD isn’t Linux, but it works in pretty much the same way as it is a descendant of the research by the Berkeley Software Distribution and it is among the world’s most broadly used Operating Systems today with its code existing in-game consoles e.g. PlayStation 4, macOS, etc.



Running RaspBSD on Raspberry Pi

5. Ubuntu Core

Ubuntu Core is the version of Ubuntu designed for Internet of Things applications. Ubuntu is the most popular Linux-based Operating System in the world with over 20+ derivatives and given that it has an active and welcoming forum, it will be easy to get up and running with Ubuntu Snappy Core on your Raspberry Pi.



Ubuntu Core for Raspberry P

Conclusion:

Experiment No. 03

****

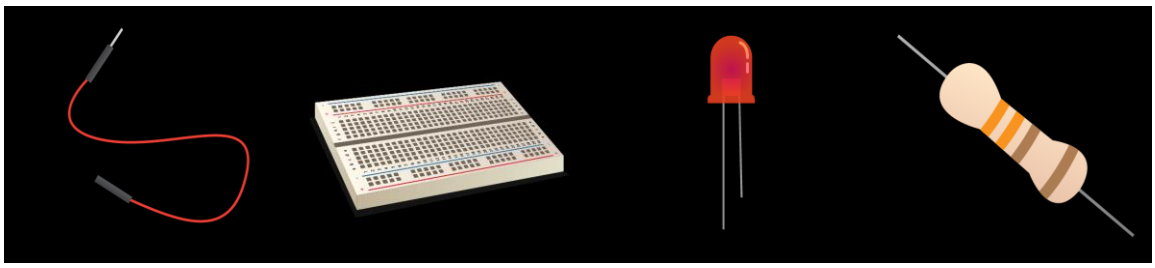
Title: Study of connectivity and configuratin of raspberry-pi circuit with basic peripherls, LED's. Understanding GPIO and its use in program.

Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

****

To light an LED, you need to build a circuit out of these components:



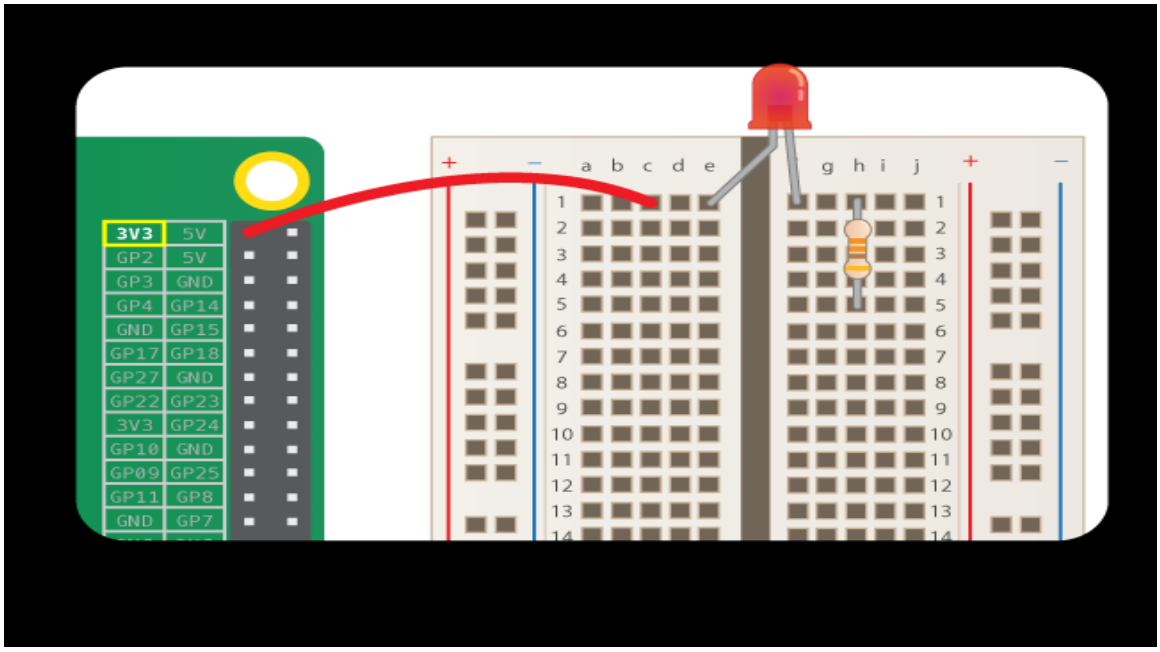
1. Female-to-Female Jumper
2. Breadboard
3. LED
4. Resister

Have a look at your LED. You should see that one leg is longer than the other. The long leg is the positive leg, and also called the anode. It should always be connected to the positive side of a circuit. The short leg is the negative leg, called the cathode. It needs to be connected to the negative side. One way to remember this is to imagine the long leg as having had something added and the short leg as having had something taken away.

You'll find that there are LEDs that have legs of the same length. In that case, the positive leg is the leg where the plastic edge of the LED is round. Where the negative leg is, the edge will be flattened, like in the image below.

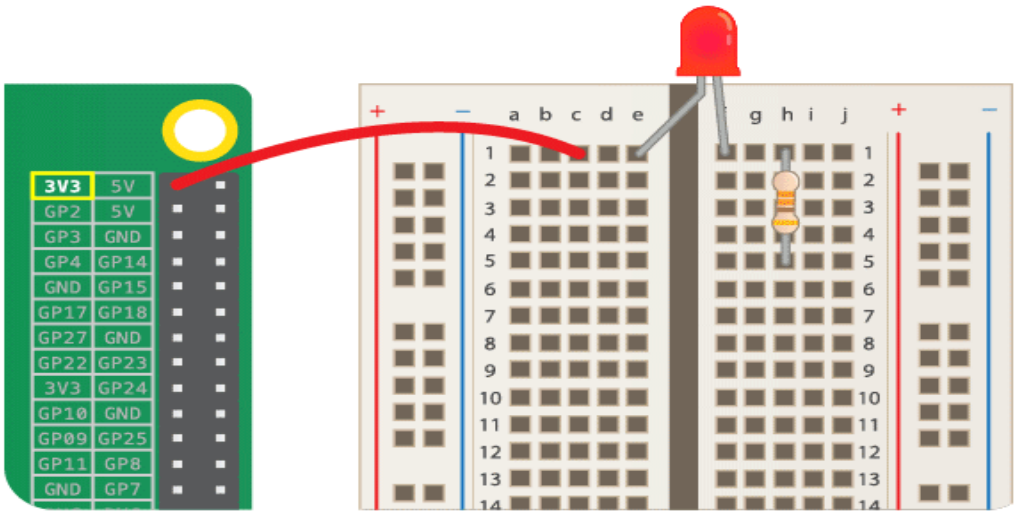
- Push the positive leg of the LED into row 1 of your breadboard, close to the left side of the ravine. Place the negative leg into row 1 on the other side of the ravine.
- Now find your resistor. A resistor is a non-polarised component, so it doesn't matter which way around you put it into the breadboard. Push one leg into the same row that the negative LED leg is in, so it connects to the LED. Push the other resistor leg into any other free row on the right side of the ravine.
- Now take a male-to-female jumper wire and push the male end into the same row as the LED, on the left side of the ravine near the LED's positive leg. Push the female end onto the 3V3 GPIO pin.

Your circuit should look a little like this:



Now, connect your components to the ground (GND) GPIO pin:

Make sure that your Raspberry Pi is powered on. Take another male-to-female jumper wire and push the male end into the same row as the resistor's second leg, on the same side of the ravine. Then push the female end onto your GND pin. Your LED should light up!



If

If your LED doesn’t light, try the following:

1) Check your Raspberry Pi is on

2) Check all your components are pushed firmly into the breadboard

3) Check your LED is the right way around

4) Make sure the legs of your components are on the right side of the ravine

5) Try another LED

Conclusion:

Experiment No. 04

****

Title: Understanding he connectivity of raspberry-pi ircuit with Temperature sensor. Write an application to read the environment temperature. If temperature crosses a threshold value, the application indicated user using LED's

.Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

****

Building the Raspberry Pi ds18b20 Circuit:

1. First, connect the 3v3 pin from the Pi to the positive rail and a ground pin to the ground rail on the breadboard.

2. Now place the DS18B20 sensor onto the breadboard.

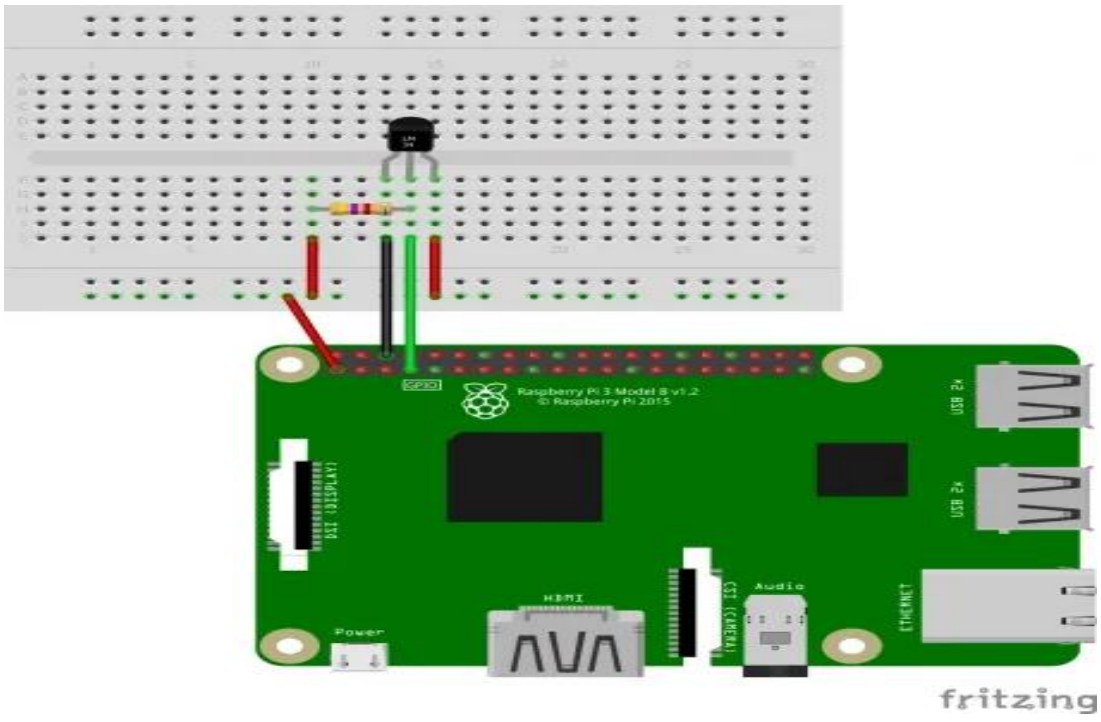
3. Place a 4.7k resistor between the positive lead and the output lead of the sensor.

4. Place a wire from the positive lead to the positive 3v3 rail.

5. Place a wire from the output lead back to pin #4 (Pin #7 if using physical numbering) of the Raspberry Pi.

6. Place a wire from the ground lead to the ground rail.

Once done, your circuit should look similar to the diagram below. Keep in mind some versions of the temperature sensor may have more wires than just three. Refer to the datasheet for more information for each of the wires.



The Raspberry Pi Temperature Sensor Code:

1. To add support for OneWire, we first need to open up the boot config file, and this can be done by running the following commandin nano text editor.

sudo nano /boot/config.txt

2. At the bottom of this file enter the following.

dtoverlay=w1-gpio

3. Once done save and exit by pressing CTRL + X and then Y . Now reboot the Pi by running the following command.

sudo reboot

4. You can skip to downloading the code onto the Pi or follow the next few steps to check that the sensor is actually working.

5. Once the Raspberry Pi has booted back up, we need to run modprobe so we can load the correct modules.

sudo modprobe w1-gpio

sudo modprobe w1-therm

6. Now change into the devices directory and use the ls command to see the folders and files in the directory.

cd /sys/bus/w1/devices

ls

7. Now run the following command, change the numbering after cd to what has appeared in your directory by using the ls command. (If you have multiple sensors there will be more than one directory)

cd 28-000007602ffa

8. Now run the following cat command.

cat w1_slave

9. This command should output data but as you will notice it is not very user-friendly.

The first line should have a YES or NO at the end of it. If it is YES, then a second line with the temperature should appear.This line will look similar to something like t=12323, and you will need to do a bit of math to make this a usable temperature that we can understand easily. For example, Celsius you simply divide by 1000.

Program:

```
import os                                   # import os module
import glob                                 # import glob module
import time                                 # import time module
os.system('modprobe w1-gpio')               # load one wire communication
device kernel modules
os.system('modprobe w1-therm')
base_dir = '/sys/bus/w1/devices/'           # point to the address
```

```
device_folder = glob.glob(base_dir + '28*')[0]          # find device with address
starting from 28*
device_file = device_folder + '/w1_slave'               # store the details
def read_temp_raw():
  f = open(device_file, 'r')
  lines = f.readlines()                          # read the device details
  f.close()
  return lines

def read_temp():
  lines = read_temp_raw()
  while lines[0].strip()[-3:] != 'YES':              # ignore first line
    time.sleep(0.2)
    lines = read_temp_raw()
  equals_pos = lines[1].find('t=')                   # find temperature in the details
  if equals_pos != -1:
    temp_string = lines[1][equals_pos+2:]
    temp_c = float(temp_string) / 1000.0             # convert to Celsius
    return temp_c

while True:
  print(read_temp())                             # Print temperature
  time.sleep(1)
```

Conclusion:

Experiment No. 05

****

Title: Understanding the connectivity of rasperry pi circuit with IR sensor. Write an application to detect obstacle and noify user using LED's.

.Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

****

IR sensor are most commonly use in small robots like line follower robot, Edge avoiding robot etc.. Simply putting, it can detect the presence of objects before it and also differentiate between white and black colour. Sounds cool right?

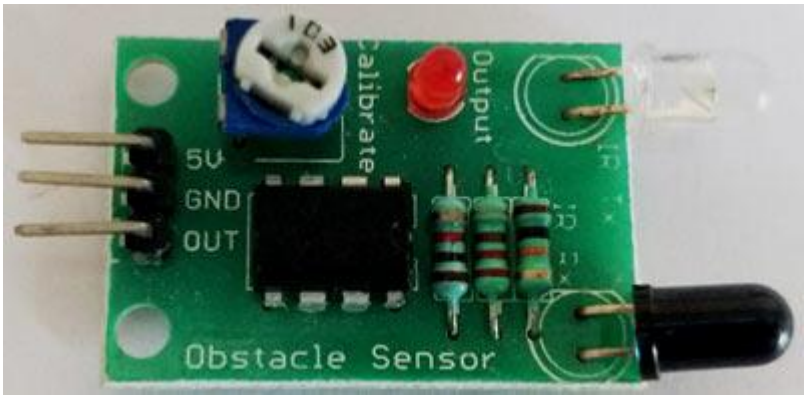
So lets learn how to interface this sensor with Raspberry Pi. In this project, when there is no object in front of IR sensor then the Red LED remains turned on and soon as we put something in front of IR sensor then red LED turns off and Green LED turn on. This circuit can also serve as Security Alarm Circuit.

Material Required:

1. Raspberry Pi 3 (any model)
2. IR sensor Module
3. Green and Red LED lights
4. Breadboard
5. Connecting wires

IR Sensor Module:

IR sensors (Infrared sensor) are modules which detect the presence of objects before them. If the object is present it give 3.3V as output and if it is not present it gives 0 volt. This is made possible by using a pair of IR pair (transmitter and receiver), the transmitter (IR LED) will emit an IR ray which will get reflected if there is a object present before it. This IR ray will be received back by the receiver (Photodiode) and the output will be made high after amplified using an op-amp link LM358. You can learn more about IR Sensor Module Circuit here.



The IR Sensor used in this project is shown above. Like all IR sensor it has three pins which are 5V, Gnd and Out respectively. The module is powered by the 5V pin from Raspberry Pi and the out pin is connected to GPIO14 of Raspberry Pi. The potentiometer on top of the module can be used to adjust the range of the IR sensor.

Programming your Raspberry Pi:

Here we are using Python Programming language for programming RPi. There are many ways to program your Raspberry Pi. In this tutorial we are using the Python 3 IDE, since it is the most used one. The complete Python program is given at the end of this tutorial. Learn more about Program and run code in Raspberry Pi here.

We are going to import GPIO file from library, below function enables us to program GPIO pins of PI. We are also renaming "GPIO" to "IO", so in the program whenever we want to refer to GPIO pins we will use the word 'IO'.

```
import RPi.GPIO as IO
```

Sometimes, when the GPIO pins, which we are trying to use, might be doing some other functions. In that case, we will receive warnings while executing the program. Below command tells the PI to ignore the warnings and proceed with the program.

```
IO.setwarnings(False)
```

We can refer the GPIO pins of PI, either by pin number on board or by their function number. Like 'PIN 29' on the board is 'GPIO5'. So we tell here either we are going to represent the pin here by '29' or '5'.

```
IO.setmode (IO.BCM)
```

We are setting 3 pins as input/output pins. The two output pins will control the LED and the input pin will read signal from the IR sensor.

```
IO.setup(2,IO.OUT) #GPIO 2 -> Red LED as output

IO.setup(3,IO.OUT) #GPIO 3 -> Green LED as output

IO.setup(14,IO.IN) #GPIO 14 -> IR sensor as input
```

Now we have to turn off the Green LED and turn on the Red LED when the object is far. This can be done by checking the GPIO14 pin.

```
if(IO.input(14)==True): #object is far away
    IO.output(2,True) #Red led ON
    IO.output(3,False) # Green led OFF
```

Similarly we have to turn on the Green LED and turn off the Red LED when the object is near.

```
If  (IO.input(14)==False): #object is near
    IO.output(3,True) #Green led ON
    IO.output(2,False) # Red led OFF
```

Program:

```
import time
import RPi.GPIO as GPIO
RUNNING = True
HIGH  = 1
LOW  = 0
DetectPin = 5

def InitSystem():
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DetectPin,GPIO.IN,pull_up_down=GPIO.PUD_UP)
        return
def DetectPerson():
        while True:
                input_state = GPIO.input(DetectPin)
                time.sleep(0.3)
```

```
                if input_state == 0:
                        return LOW
                else:
                        return HIGH
try:
        print "\nCounting using IR LED\n"
        print  "----------------------------------------------\n"
        InitSystem()
        count =0;
        while RUNNING:
                state = DetectPerson()
                if state == HIGH:
                        count+=1
                        print "person count =%d" %count
# If CTRL+C is pressed the main loop is broken
except KeyboardInterrupt:
    RUNNING = False
    print "\Stopping Elevator"
# Actions under 'finally' will always be called
finally:
    # Stop and finish cleanly so the pins
    # are available to be used again
    GPIO.cleanup()
```

Conclusion:

Experiment No. 06

****

Title: Understanding and connectivity of raspberry pi with camera.Write an application tocapture and store the image.

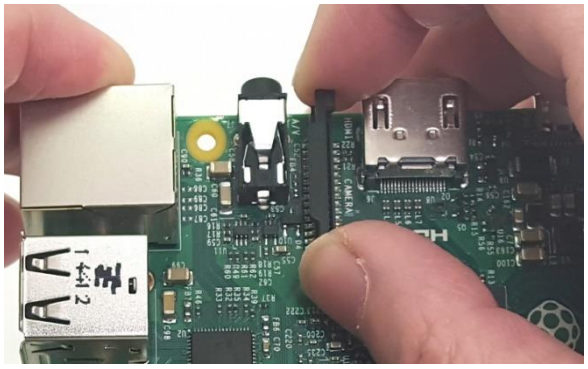
.Name: Rahulkumar Varma

Class: TE CSE **Batch: T1** **Roll No: 3020**

****

Open the Camera Port on the Raspberry Pi:

On the Raspberry Pi B+, 2 and 3, the camera port is between the audio port and the HDMI port. On the original Raspberry Pi B, it is between the Ethernet port and the HDMI port. To open the port, use two fingers and lift the end slightly. Please note that there is another port on the Pi board that looks just the same, but that other one is not me for the camera.

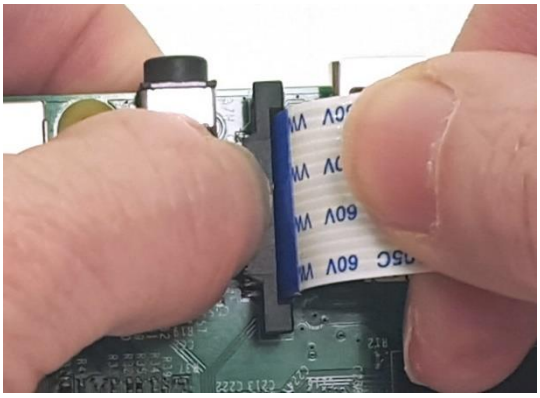


Insert the Camera Cable:

The cable has to be inserted with the right orientation: the blue has to face the Ethernet port, and the silver side is facing the HDMI port. Insert the cable so that almost no blue is showing. This photo shows the beginning of the insertion.

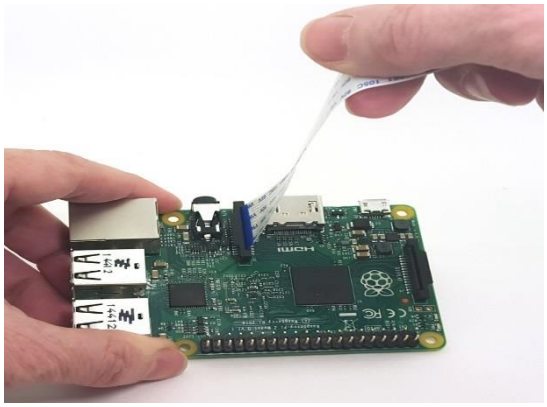
Close the Camera Port:

To close the port and keep the cable snugly in place, push the top of the port while holding the cable with the other hand. Because you're pushing it down, more blue will be exposed. That's okay.



Verify the Connection:

Hold the Pi in one hand, and the camera cable in the other. Give a gentle tug. No force is needed, really. If the cabl not properly held in place, it will slide out of the port. You will feel a resistance if it's inserted solidly. Just a little pull enough.



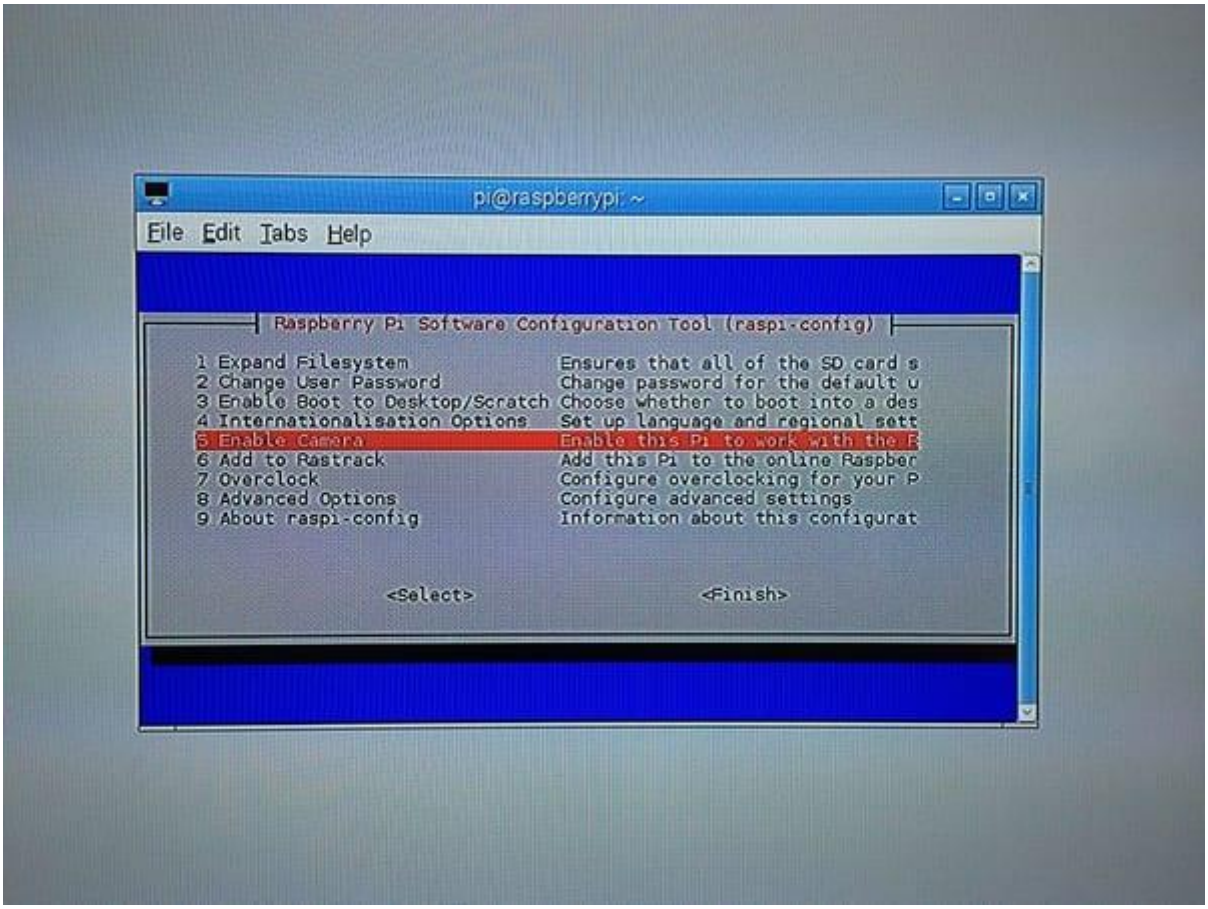
Enable Camera Module:

Now that you have your Raspberry Pi camera module installed, you need to enable it. Open up a terminal and execute the following command:

```
$ sudo raspi-config
```

Use your arrow keys to scroll down to **Option 5: Enable camera**, **hit your enter key** to enable the camera, and then arrow down to the **Finish** button and hit enter again. Lastly, **you'll need to reboot your Raspberry Pi** for the configuration to take affect.

This will bring up a screen that looks like this:



Enabling the Raspberry Pi camera module using the raspi-config command.

Test Out the camera module:

Before we dive into the code, let's run a quick sanity check to ensure that our Raspberry Pi camera is working properly.

```
$ raspistill -o output.jpg
```

Installing Picamera:

So at this point we know that our Raspberry Pi camera is working properly. But how do we interface with the Raspberry Pi camera module using Python?

The answer is the picamera module.

```
$ pip install "picamera[array]"
```

Program:

```
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
```

```
import time
import cv2

# initialize the camera and grab a reference to the raw camera capture
camera = PiCamera()
rawCapture = PiRGBArray(camera)

# allow the camera to warmup
time.sleep(0.1)

# grab an image from the camera
camera.capture(rawCapture, format="bgr")
image = rawCapture.array

# display the image on screen and wait for a keypress
cv2.imshow("Image", image)
cv2.waitKey(0)
```

Conclusion: