



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade 'A' by NAAC

INTERNET OF THINGS

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE AWARD OF BACHELOR OF
ENGINEERING DEGREE IN

ELECTRICAL AND ELECTRONICS ENGINEERING



NAME: S PRAHADEESH

REG NO : 39140042



एन एस आई सी
N S I C

No. : 17767

राष्ट्रीय लघु उद्योग निगम लिमिटेड

NATIONAL SMALL INDUSTRIES CORPORATION LIMITED
(A Govt. of India Enterprise)

NSIC - TECHNICAL SERVICES CENTRE,

Sector B-24, Guindy Industrial Estate, Ekkaduthangal, Chennai - 600 032

CERTIFICATE

This is to certify that Mr/Ms. **PRAHADEESH S.** S/o /
D/o. Shri. **SRIDHAR R.** Student of III year B.E/B.Tech
(EEE), Sathyabama Institute of Science and
Technology, has undergone **Internship Training on**
IOT APPLICATIONS USING RASPBERRY PI
conducted by us for a period of two weeks from
23.09.2021 to 06.10.2021.



[Signature]
HEAD OF TRAINING

[Signature]
HEAD OF CENTRE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY for their kind encouragement in doing this internship and for completing it successfully.

I am grateful to them.

I convey my thanks to Dr .Nandhitha, Dean, School of Electronics and Communication Engineering and Dr .T.Ravi, Head of the Department, Dept. of Electronics and communication engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my guide Veda Narayanan for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my internship.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the inter

IOT APPLICATIONS USING RASBERRY PI

1. INTRODUCTION TO IOT



It is a network of physical objects or things sending, receiving, or communicating using the internet or other communication technologies.

The Internet of Things (IoT), as this intelligent interconnectivity between the real and the digital world is called, will rapidly transform every aspect of how we work and do business. By connecting apps with their integrated systems, businesses are able to transform their industry significantly. Today almost 90% of all data generated by tablets, smartphones or connected appliances is never acted upon. Imagine you could change that. It seems safe to say that we have never encountered a single technological platform that combines this much complexity, global reach and novelty. Since IoT allows devices to be controlled remotely across the internet, thus it created opportunities to directly connect & integrate the physical world to the computer-based systems using sensors and internet. The interconnection of these multiple embedded devices will be resulting in automation in nearly all fields and enabling advanced applications. This is resulting in improved accuracy, efficiency and economic benefit with reduced human intervention. It encompasses technologies such as smart grids, smart homes, intelligent transportation and smart cities.

IOT DEVELOPMENT WITH RASPBERRY Pi

The Raspberry Pi is a series of low-cost, programmable computers that include a set of GPIO, or 'General Purpose Input Output', pins that can be used to connect and control external electronic devices, and to create Internet of Things (IOT) solutions.

The exact number and role of the pins changes between individual models but is generally divided into power, ground, and general-purpose pins. The power and ground pins are not programmable. The power pin supplies a constant 3.3V/5V power to the circuit while the ground pin is used to connect to the cathode of the circuit.

The general-purpose pins are fully programmable and can be used in either an output or input mode. When set to output mode, the pins provide a constant 3.3V power that can be turned on or off. When set to input mode, the pin reads the current supplied by the circuit and returns a Boolean value indicating if it receives 3.3V power or not.

Of course, these capabilities aren't new and have been widely available to developers through microcontrollers such as Arduino or NodeMCU. However, these devices generally came with limited memory and computing power and required the use of particular programming languages.

The Raspberry Pi, on the other hand, supports a more robust CPU that is capable of running Linux and supports NodeJS, allowing JavaScript developers to use their existing skillset and build sophisticated devices with relative ease. To interact with the GPIO pins, we use a NodeJS module called onoff that provides simple access to the individual pins.

Rpi-ARCHITECTURE

The first generation (**Raspberry Pi Model B**) was released in February 2012, followed by the simpler and cheaper **Model A**.

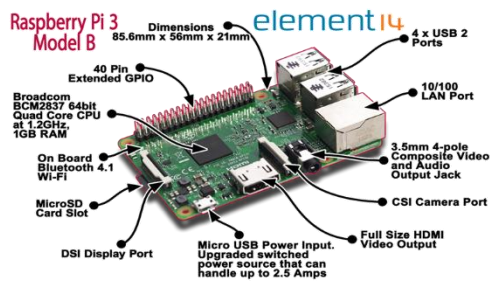
- In 2014, the Foundation released a board with an improved design, **Raspberry Pi Model B+**. These first generation boards feature ARM11 processors, are approximately credit-card sized and represent the standard *mainline* form-factor. Improved A+ and B+ models were released a year later. A "Compute Module" was released in April 2014 for embedded applications.



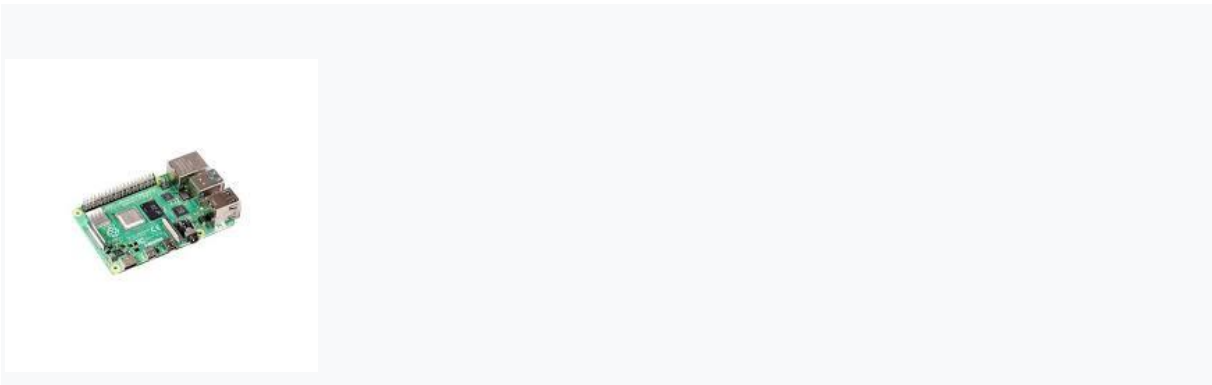
- The **Raspberry Pi 2** was released in February 2015 and initially featured a 900 MHz 32-bit quad-core ARM Cortex-A7 processor with 1 GB RAM. Revision 1.2 featured a 900 MHz 64-bit quad-core ARM Cortex-A53 processor (the same as that in the Raspberry Pi 3 Model B, but underclocked to 900 MHz).



- **Raspberry Pi 3 Model B** was released in February 2016 with a 1.2 GHz 64-bit quad core ARM Cortex-A53 processor, on-board 802.11n Wi-Fi, Bluetooth and USB boot capabilities.



- 2018, the **Raspberry Pi 3 Model B+** was launched with a faster 1.4 GHz processor, a three-times faster gigabit Ethernet (throughput limited to ca. 300 Mbit/s by the internal USB 2.0 connection), and 2.4 / 5 GHz dual-band 802.11ac Wi-Fi (100 Mbit/s). Other features are Power over Ethernet (POE) (with the add-on POE HAT), USB boot and network boot (an SD card is no longer required)).
- **Raspberry Pi 4 Model B** was released in June 2019^[2] with a 1.5 GHz 64-bit quad core ARM Cortex-A72 processor, on-board 802.11ac Wi-Fi, Bluetooth 5, full gigabit Ethernet (throughput not limited), two USB 2.0 ports, two USB 3.0 ports, 2-8 GB of RAM, and dual-monitor support via a pair of micro HDMI (HDMI Type D) ports for up to 4K resolution. The Pi 4 is also powered via a USB-C port, enabling additional power to be provided to downstream peripherals, when used with an appropriate PSU. The initial Raspberry Pi 4 board has a design flaw where third-party e-marked USB cables, such as those used on Apple MacBooks, incorrectly identify it and refuse to provide power. Tom's Hardware tested 14 different cables and found that 11 of them turned on and powered the Pi without issue.^[32] The design flaw was fixed in revision 1.2 of the board, released in late 2019.



- **Raspberry Pi 400** was released in November 2020. It features a custom board that is derived from the existing Raspberry Pi 4, specifically remodelled with a keyboard attached. The case was derived from that of the Raspberry Pi Keyboard. A robust cooling solution similar to the one found in Commodore 64 allows the Raspberry Pi 400's Broadcom BCM2711C0 processor to be clocked at 1.8 GHz, which is slightly higher than the Raspberry Pi 4 it's based on. The keyboard-computer features 4 GB of LPDDR4 RAM.



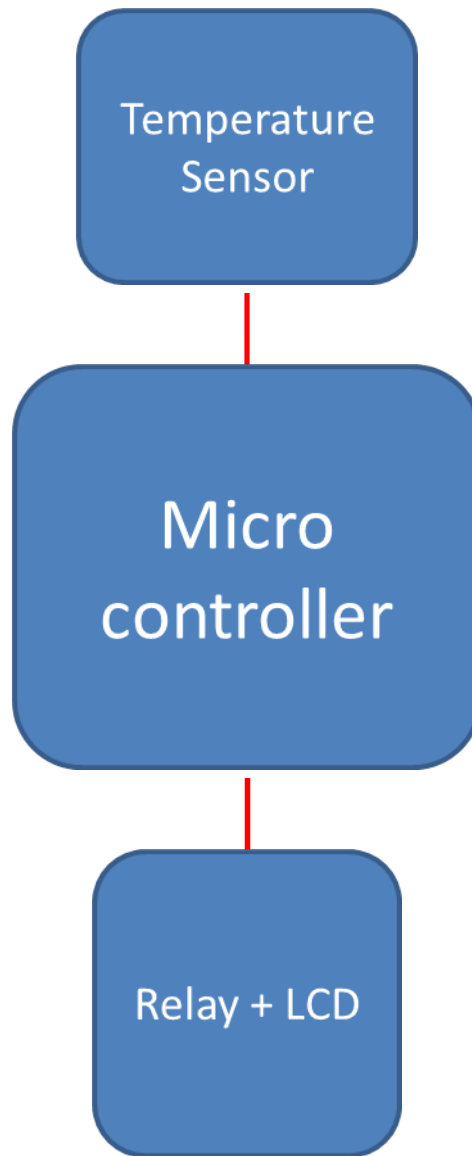
HOW TO DEVELOP EMBEDDED BASED PROJECT

EMBEDDED SYSTEMS- DEFINITION

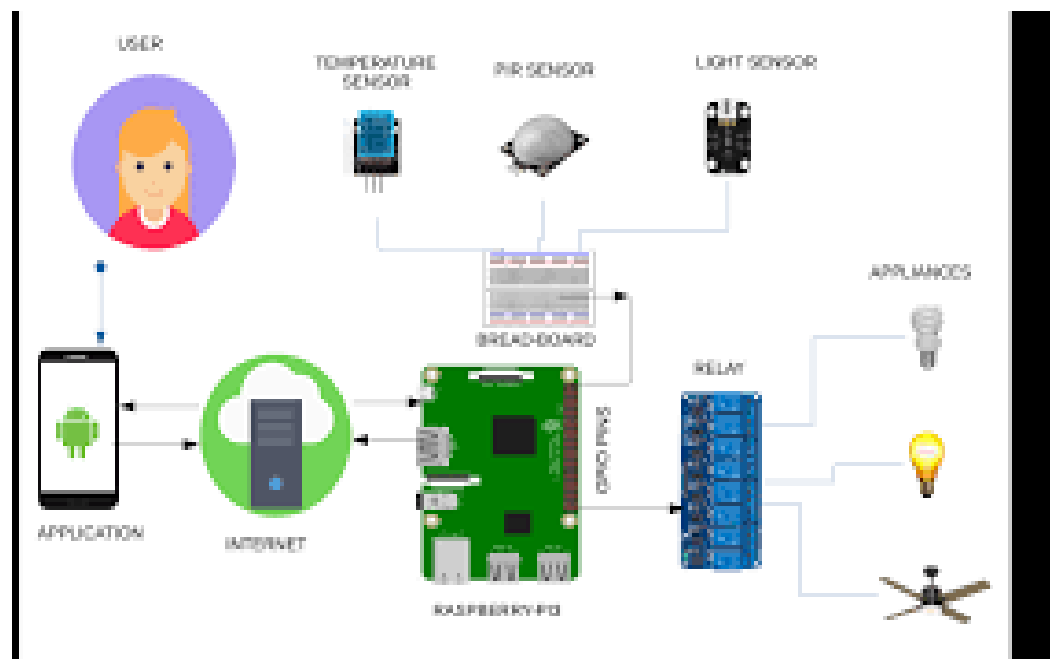
- **Micro controller based system used for specific purpose**
- **Will be used for real-time monitoring system**



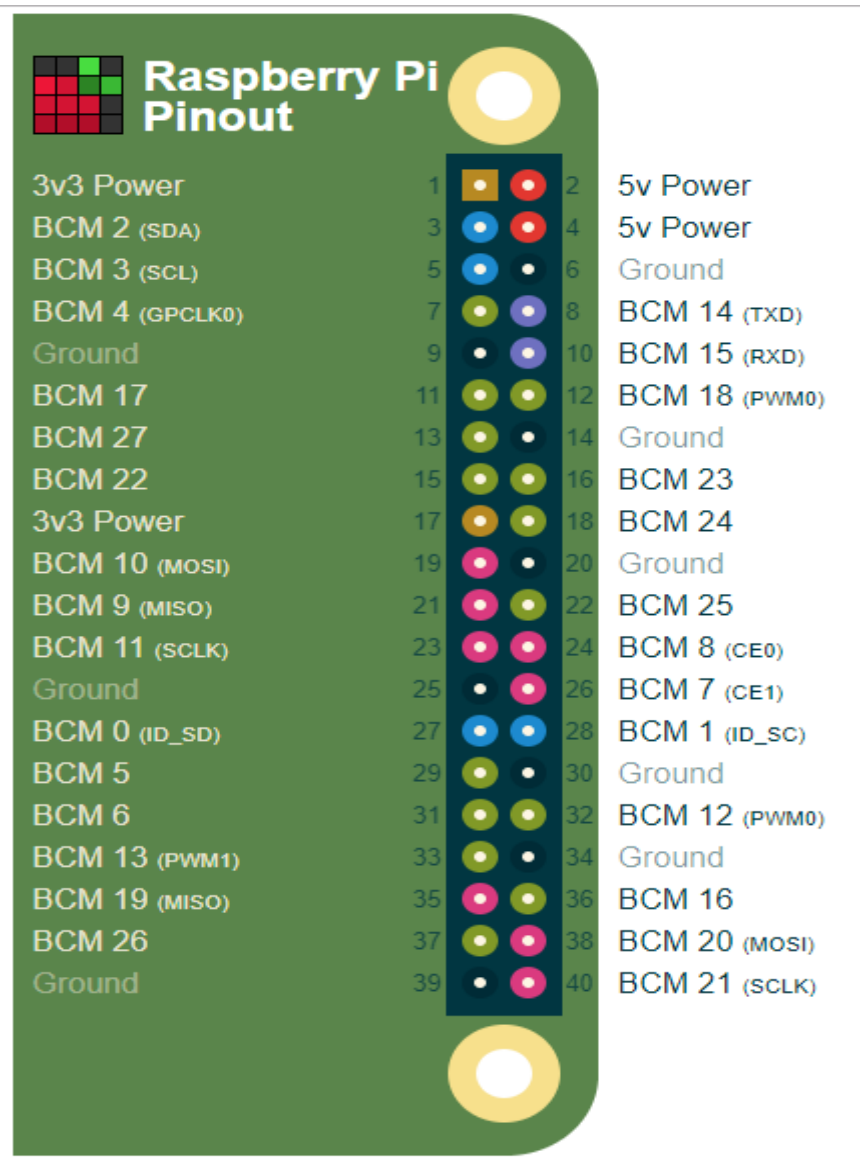
TEMPERATURE CONTROL SYSTEM



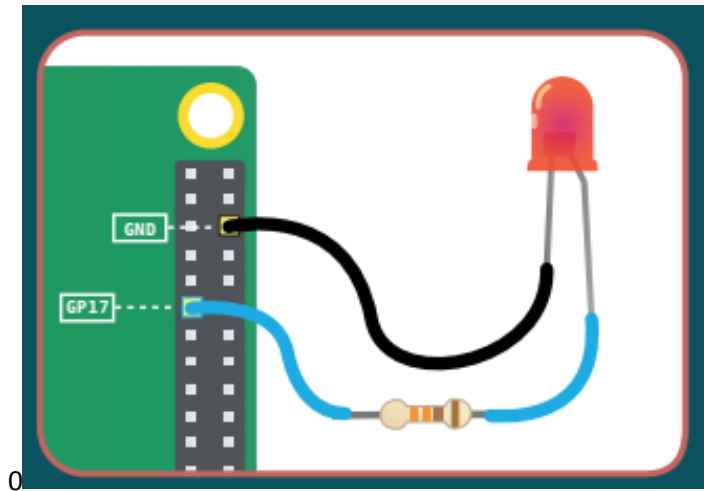
HOME AUTOMATION



GPIO PINOUT



LED



0

<code>led.on()</code>
<code>led.off()</code>
<code>led.toggle()</code>
<code>led.blink()</code>

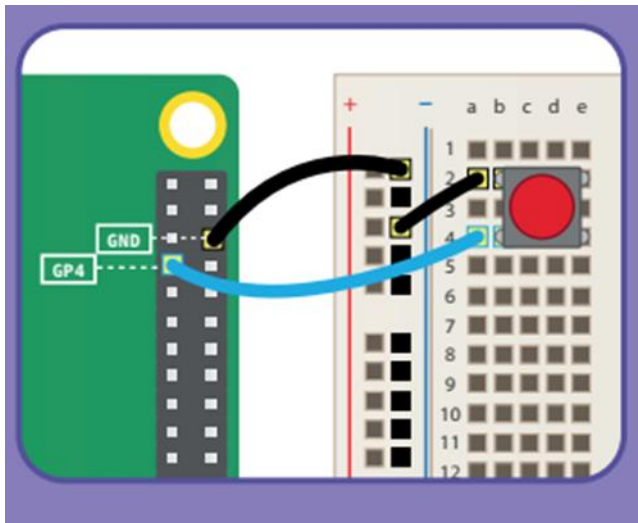
```
1 from gpiozero import PWMLED
  from time import sleep

  led = PWMLED(17)

  while True:
      led.value = 0 # off
      sleep(1)
      led.value = 0.5 # half brightness
      sleep(1)
      led.value = 1 # full brightness
      sleep(1)
```

```
from gpiozero import LED
led = LED(17)
led.on()
```

BUTTON



```
button.wait_for_press()
```

```
button.wait_for_release()
```

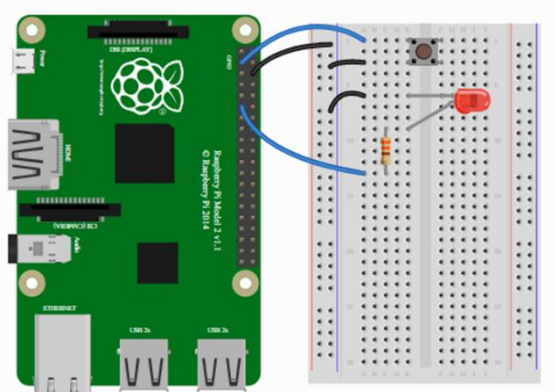
```
button.is_pressed
```

```
button.when_pressed = led.on
```

```
button.when_released = led.off
```

```
from gpiozero import Button
button = Button(4)
while True:
    if button.is_pressed:
        print("Button is pressed")
    else:
        print("Button is not pressed")
```

BUTTON CONTROLLED LED



```
from gpiozero import LED, Button
from signal import pause

led = LED(17)
button = Button(2)

button.when_pressed = led.on
button.when_released = led.off

pause()
```

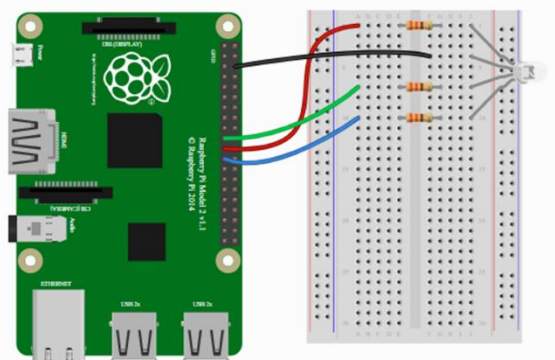
TRAFFIC LIGHT

```
from gpiozero import TrafficLights
from time import sleep

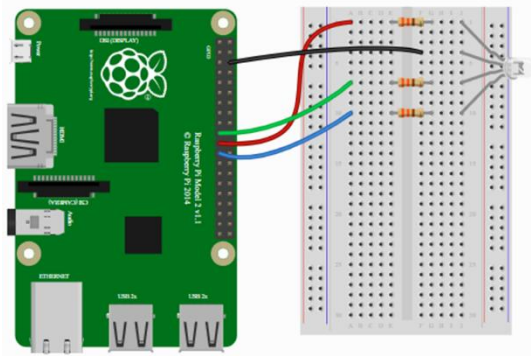
lights = TrafficLights(2, 3, 4)

lights.green.on()

while True:
    sleep(10)
    lights.green.off()
    lights.amber.on()
    sleep(1)
    lights.amber.off()
    lights.red.on()
    sleep(10)
    lights.amber.on()
    sleep(1)
    lights.green.on()
    lights.amber.off()
    lights.red.off()
```



RGB LED



```
from gpiozero import RGBLED
from time import sleep

led = RGBLED(red=9, green=10, blue=11)

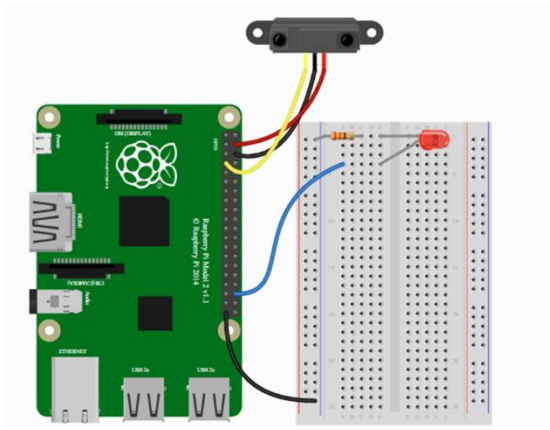
led.red = 1 # full red
sleep(1)
led.red = 0.5 # half red
sleep(1)

led.color = (0, 1, 0) # full green
sleep(1)
led.color = (1, 0, 1) # magenta
sleep(1)
led.color = (1, 1, 0) # yellow
sleep(1)
led.color = (0, 1, 1) # cyan
sleep(1)
led.color = (1, 1, 1) # white
sleep(1)

led.color = (0, 0, 0) # off
sleep(1)

# slowly increase intensity of blue
for n in range(100):
    led.blue = n/100
    sleep(0.1)
```

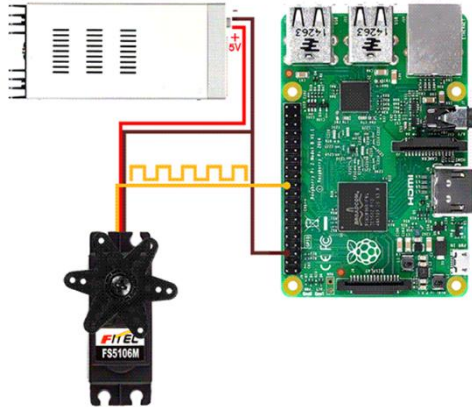
MOTION SENSOR



```
from gpiozero import MotionSensor

pir = MotionSensor(4)
pir.wait_for_motion()
print("Motion detected!")
```

SERVO



```
from gpiozero import Servo
from time import sleep

servo = Servo(17)
while True:
    servo.min()
    sleep(1)
    servo.mid()
    sleep(1)
    servo.max()
    sleep(1)
```

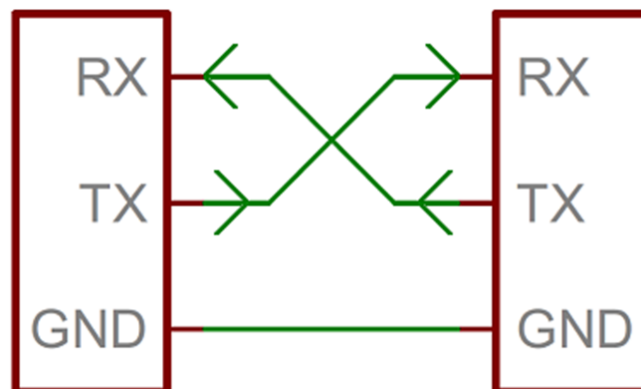
SERIAL COMMUNICATION

- Asynchronous communication
 - Data frame



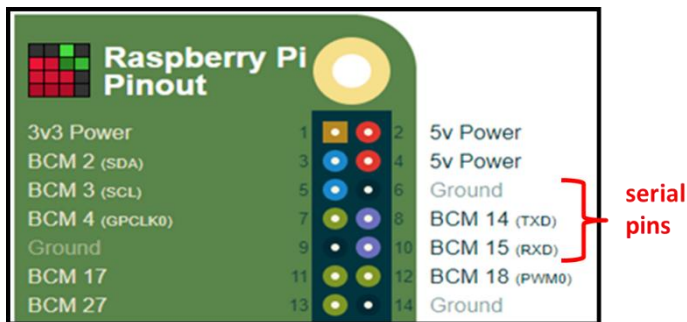
- Baud rate
 - 9600 bps
- Configuration example
 - 9600 8N1

WIRING



RASPBERRY PI - SERIAL

- to find serial port
dmesg | grep tty
- go to **raspi-config** and disable login shell to be accessible over serial in “Advanced options”.



MAXSONAR

- High Performance Sonar Range Finder
- Vcc is 2.5V - 5.5V



sudo leafpad /boot/config.txt

add the following lines

enable_uart=1

dtoverlay=pi3-disable-bt

dtoverlay=pi3-miniuart-bt

I2C PYTHON CODE

```
import smbus
import time
bus=smbus.SMBus(1)
address=0x49
def temp_read():
    temp_value=bus.read_byte_data(address,0)
    return temp_value
while True:
    temperature=temp_read()
    print (temperature)
    time.sleep(1)
```

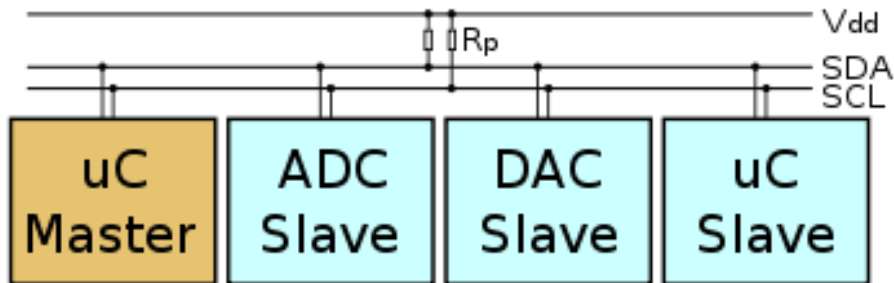
I2C

- I2C is a serial protocol for two-wire interface to connect low-speed devices like microcontrollers, EEPROMs, A/D and D/A converters.
- Invented by Philips
- I2C speed

Bus Speed	Data Transfer
Low Speed	10Kbit/Second
Standard Mode	100Kbit/Second
Fast Mode	400Kbit/Second
Fast Plus	1Mbit/Second
High Speed	3.4Mbits/Second

I2C INTERFACE

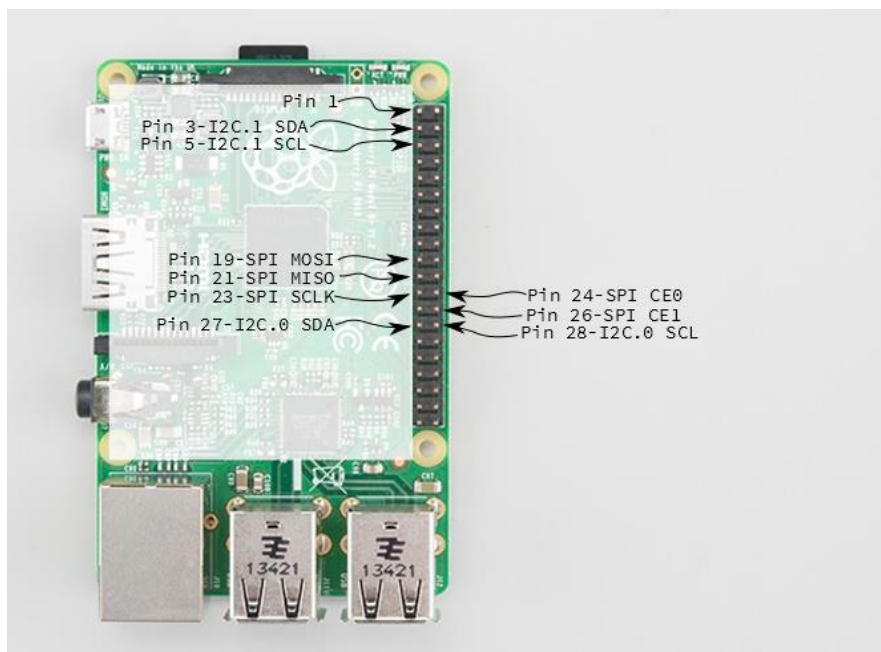
- SCL (serial clock) and SDA (serial data)
- Both need to be pulled up with a resistor to +Vdd



I2C COMMUNICATION

- 7 bit addresses
- Start Condition
 - the master device leaves SCL high and pulls SDA low

RASPBERRY PI –I2C



ENABLING I2C ON RPI

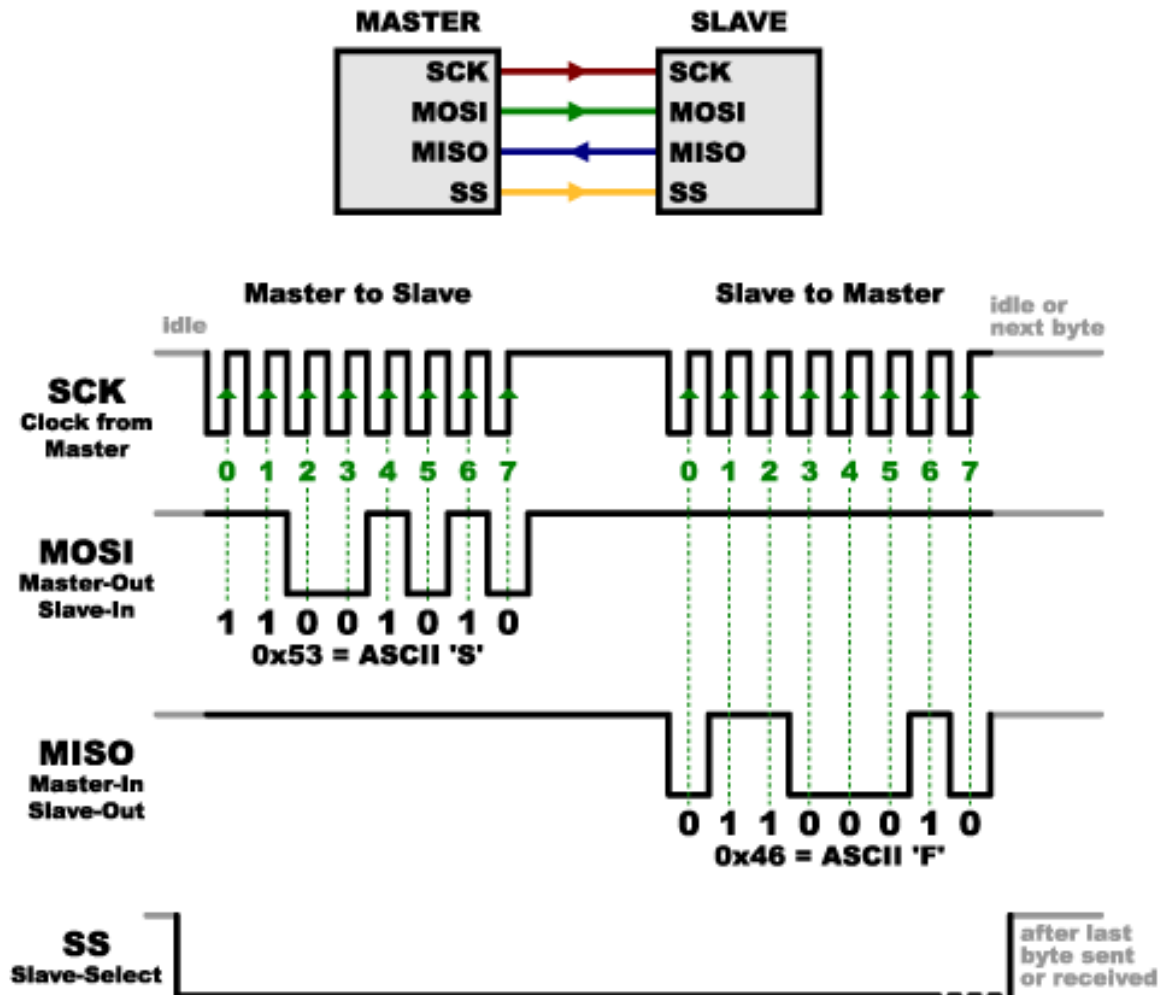
- go to raspi-config and enable i2c in “Interfacing options”.
- `ls /dev/*i2c*`
 `/dev/i2c-1`
- `i2cdetect -y 1`

```
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:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: 60 -- -- -- -- -- -- -- -- -- -- -- -- -- --
70:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

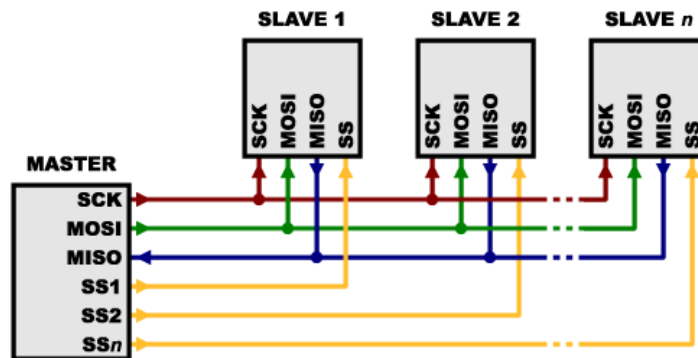
PMOD TMP3

- [Microchip TCN75A](#)
- 8 different I²C slave addresses ranging from 0x48 to 0x4F
- JP1, JP2, and JP3 for address configuration
- JP4 for SDA and JP5 for SCL – Pull up configuration
- 9-bits of temperature data are arranged in a [two's complement](#) format

SPI



SPI – MULTIPLE SLAVES



ENABLING SPI ON RPI

- go to *raspi-config* and enable SPI in “Interfacing options”.
- `ls /dev/*spi*`
`/dev/spidev0.0 /dev/spidev0.1`

PMODALS

- [Texas Instruments ADC081S021](#), an 8-bit analog-to-digital converter
- [Vishay Semiconductors TMT6000X01](#) ambient light sensor
- The data itself comes in a set of 15 bits
- The first 3 bits are leading zeros
- bits 4-11 are the information about the light level with the MSB first
- bits 12-15 are trailing zeros

SPI C CODE

```
#include <bcm2835.h>
#include <stdio.h>
int main(int argc, char **argv) {
    if (!bcm2835_init()) {
        printf("oops, could not init bcm2835\n");
        return 1;
    }
    bcm2835_spi_begin();
    bcm2835_spi_setBitOrder(BCM2835_SPI_BIT_ORDER_
MSBFIRST);    // The default
    bcm2835_spi_setDataMode(BCM2835_SPI_MODE0);
// The default
    bcm2835_spi_setClockDivider(BCM2835_SPI_CLOCK_
DIVIDER_64);    // ~ 4 MHz
    bcm2835_spi_chipSelect(BCM2835_SPI_CS0);
// The default
    bcm2835_spi_setChipSelectPolarity(BCM2835_SPI_CS0,
LOW);    // the default
    uint8_t mosi[10] = { 0x60, 0x00 };
    uint8_t miso[10] = { 0 };
    bcm2835_spi_transf(bcm2835_spi_write, mosi, miso, 2);
    printf("Analogue level from SPI: %04x\n", miso[1] +
((miso[0] & 3) << 8));
    bcm2835_spi_end();
    bcm2835_close();
}
```

WIRINGPI LIBRARY

```
# gpio -v
# gpio readall
# gpio -g mode 17 out // -g for interpreting BCM nos
# gpio -g write 17 1
# gpio -g write 17 0
```

INSTALLING APACHE 2

```
# sudo apt-get update
# sudo apt-get install apache2 -y
```

Browse to: <http://localhost/>

Default webpage:
</var/www/html/index.html>

UNINSTALLING APACHE2

```
# which apache2
# sudo service apache2 stop
# sudo apt-get purge apache2 apache2-utils apache2.2-bin
apache2-common
# sudo apt-get autoremove
# which apache2 //confirming the removal
# sudo apt-get purge php libapache2-mod-php
```

INSTALLING PHP

```
# sudo apt-get install php libapache2-mod-php -y
```

```
# sudo rm index.html
```

```
# sudo leafpad index.php
```

```
# sudo visudo
```

```
- www-data ALL=NOPASSWD: ALL in sudoers  
file
```

```
# sudo chown -R www-data:www-data /var/www
```

PHP EXAMPLES

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

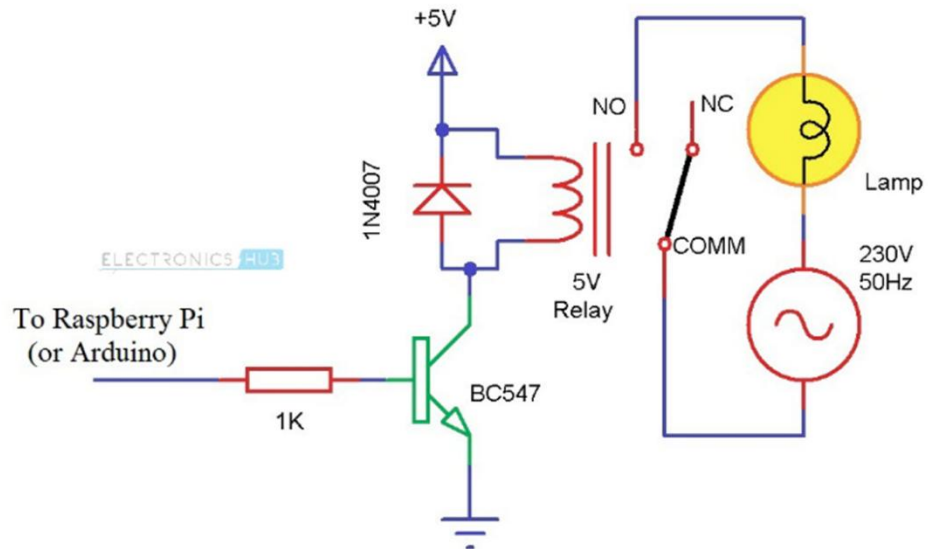
```
echo "My first PHP script!";
```

```
?>
```

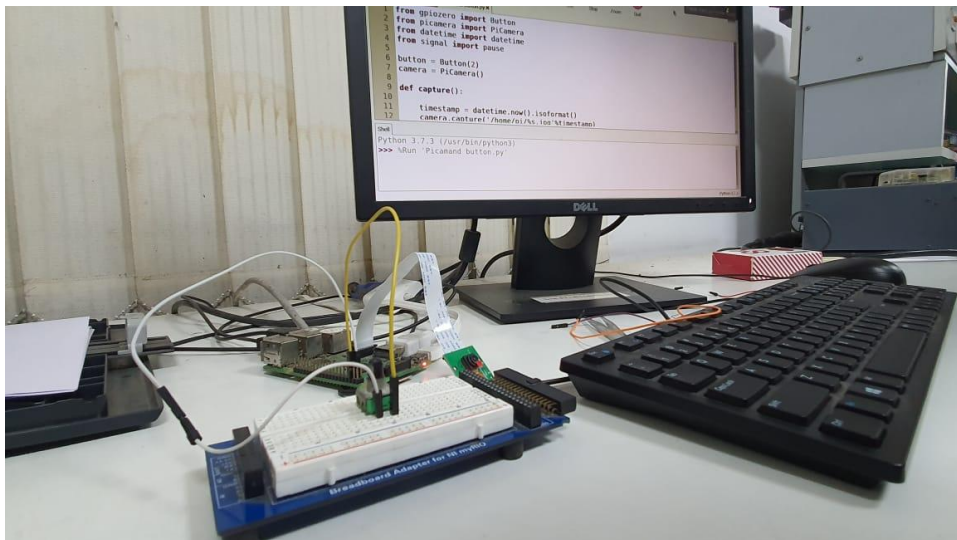
```
</body>
```

```
</html>
```

LAMP INTERFACING



PI CAM



TEMPERATURE USING THINGSPEAK

```
1 import http.client, urllib
2 import time
3
4 sleep = 5
5 key = '1ZD7E0NGC70UTL1W'
6
7 def thermometer():
8     while True:
9         temp = int(open('/sys/class/thermal/thermal_zone0/temp').read())/1e3
10        params = urllib.parse.urlencode({'field1':temp,'key':key})
11        headers = {'Content-type': 'application/x-www-form-urlencoded', 'Accept': 'text/plain'}
12        conn = http.client.HTTPConnection("api.thingspeak.com:80")
13        try:
14            conn.request("POST", "/update", params, headers)
15            response = conn.getresponse()
```

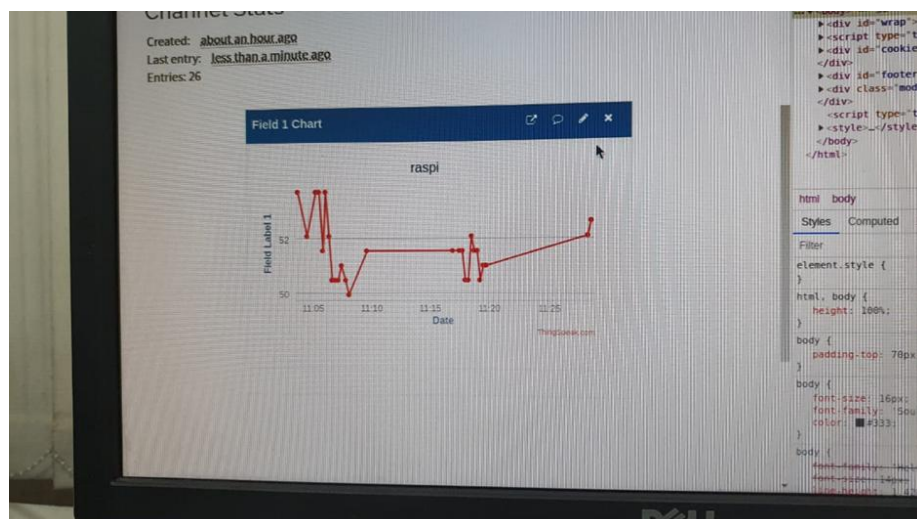
Shell

52.078
200 OK

```
13 try:
14     conn.request("POST", "/update", params, headers)
15     response = conn.getresponse()
16     print(temp)
17     print(response.status, response.reason)
18     data = response.read()
19     conn.close()
20 except:
21     print("Error")
22     break
23
24 if __name__ == "__main__":
25     while True:
26         thermometer()
27         time.sleep(sleep)
```

Shell

51.54
200 OK



THANK YOU