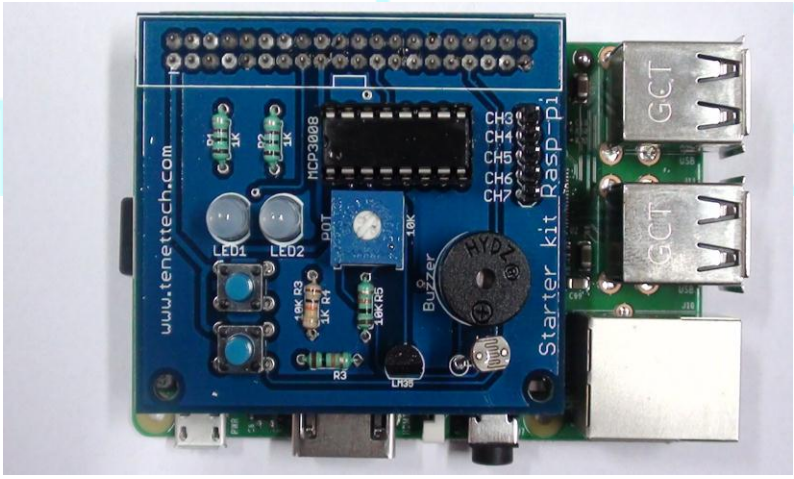
2016

TENET TECHNETRONICS

Interfacing starter kit with Raspberry pi



Author: Vivek g s

Introduction:

Raspberry Pi is a credit card sized computer that plugs into a computer monitor or TV, and uses standard keyboard and mouse. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. Here we are going to interface starter kit with Raspberry Pi.

Hardware Requirements:

1. Raspberry Pi board.
2. Starter kit

Starter kit:

A useful and fun project for hackers/developers wanting to start programming and access the Raspberry Pi's GPIO port.
The starter kit board is a simple self-assembly kit for Raspberry Pi, that adds 2 LEDs, a Buzzer, 2 Push-buttons, Temperature sensor, Light sensor and a Potentiometer with an analog to digital convertor chipset (MCP3008).

MCP3008 IC:

The MCP3008 10-bit Analog-to-Digital Converter (ADC) combines high performance and low power consumption in a small package, making it ideal for embedded control applications. The MCP3008 features a successive approximation register (SAR) architecture and an industry-standard SPI serial interface. The MCP3008 features 200k samples/second, 8 input channels, low power consumption (5nA typical standby, 425µA typical active), and is available in 16-pin PDIP and SOIC packages. Applications for the MCP3008 include data acquisition, instrumentation and measurement, multi-channel data loggers, industrial PCs, motor control, robotics, industrial automation, smart sensors, portable instrumentation and home medical appliances.

Pin diagram:

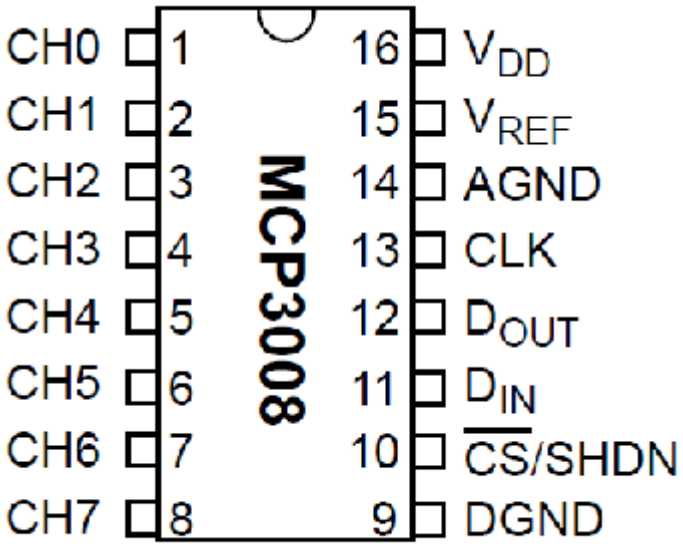
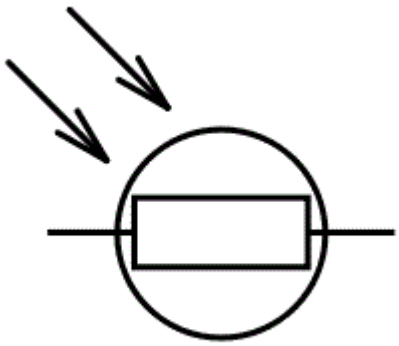


Figure 1

LDR:

A **Light Dependent Resistor** (LDR) or a photo resistor is a device whose resistivity is a function of the incident electromagnetic radiation. Hence, they are light sensitive devices. They are also called as photo conductors, photo conductive cells or simply photocells. They are made up of semiconductor materials having high resistance. There are many different symbols used to indicate a **LDR**, one of the most commonly used symbol is shown in the figure below. The arrow indicates light falling on it.



Potentiometer:

A **potentiometer**, informally a pot, is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor

Temperature sensor:

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly- proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling.

SPI Interface:

The Serial Peripheral Interface (SPI) bus was developed by Motorola to provide full-duplex synchronous serial communication between master and slave devices. The SPI bus is commonly used for communication with flash memory, sensors, real-time clocks (RTCs), analog-to-digital converters, and more.

As shown in Figure, standard SPI masters communicate with slaves using the serial clock (SCK), Master out Slave in (MOSI), Master in Slave out (MISO), and Slave Select (SS) lines. The SCK, MOSI, and MISO signals can be shared by slaves while each slave has a unique SS line.

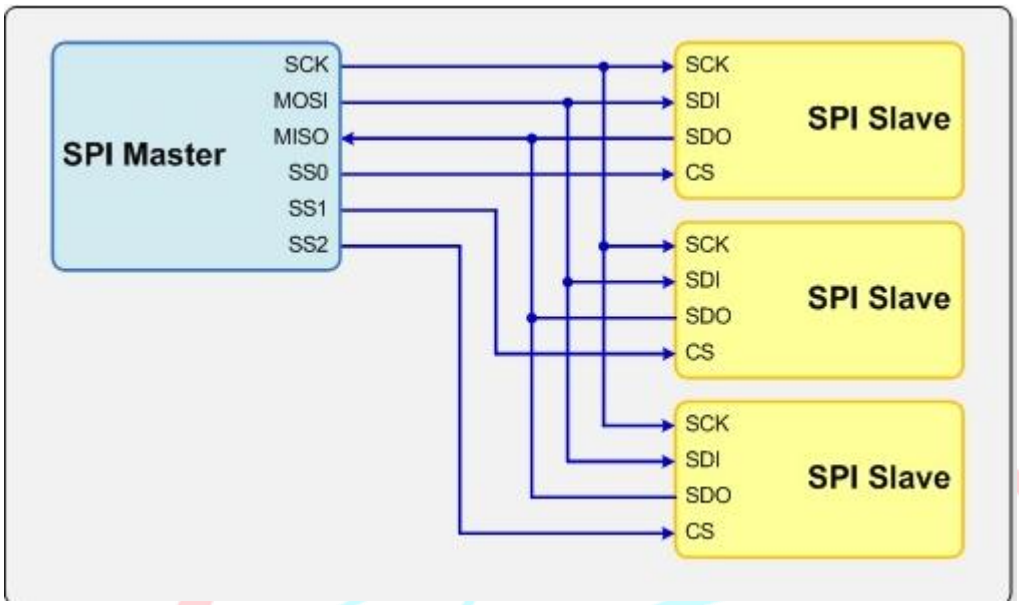


Figure 2

Polarity and Clock Phase

The SPI interface defines no protocol for data exchange, limiting overhead and allowing for high speed data streaming. Clock polarity (CPOL) and clock phase (CPHA) can be specified as '0' or '1' to form four unique modes to provide flexibility in communication between master and slave. If CPOL and CPHA are both '0' (defined as Mode 0) data is sampled at the leading rising edge of the clock. Mode 0 is by far the most common mode for SPI bus slave communication. If CPOL is '1' and CPHA is '0' (Mode 2), data is sampled at the leading falling edge of the clock. Likewise, CPOL = '0' and CPHA = '1' (Mode 1) results in data sampled at on the trailing falling edge and CPOL = '1' with CPHA = '1' (Mode 3) results in data sampled on the trailing rising edge. Table 1 below summarizes the available modes.

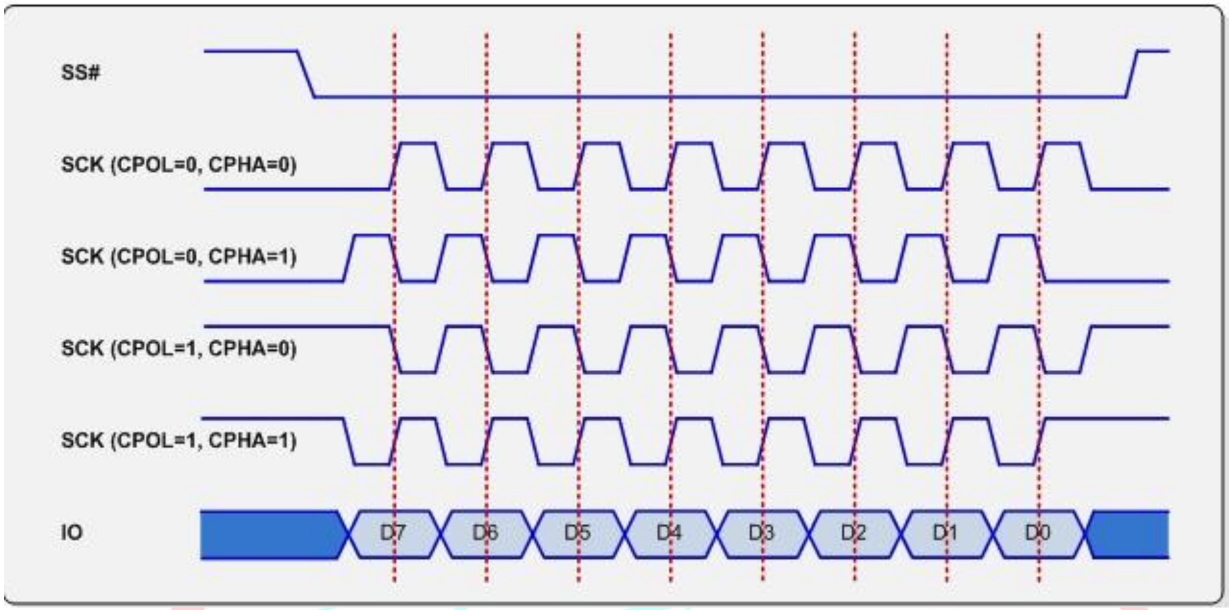


Figure 3

Enabling SPI on raspberry pi:

- **Start by running the following command :**

  Sudo raspi-config

- **This will launch the raspi-config utility. Select option 8 "Advanced Options".**

```
ââââââââââââââââââ¤ Raspberry Pi Software Configuration Tool (raspi-config) ââââââââââââââââââ
â Setup Options                                                                             â
â                                                                                           â
â    1 Expand Filesystem                     Ensures that all of the SD card storage        â
â    2 Change User Password                  Change password for the default user (p        â
â    3 Enable Boot to Desktop/Scratch        Choose whether to boot into a desktop e        â
â    4 Internationalisation Options          Set up language and regional settings t        â
â    5 Enable Camera                         Enable this Pi to work with the Raspber        â
â    6 Add to Rastrack                       Add this Pi to the online Raspberry Pi         â
â    7 Overclock                             Configure overclocking for your Pi             â
â    8 Advanced Options                      Configure advanced settings                    â
â    9 About raspi-config                    Information about this configuration to        â
â                                                                                           â
â                                                                                           â
â                     <Select>                          <Finish>                            â
â                                                                                           â
ââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââ
```

Figure 4

- **Select the “SPI” option.**

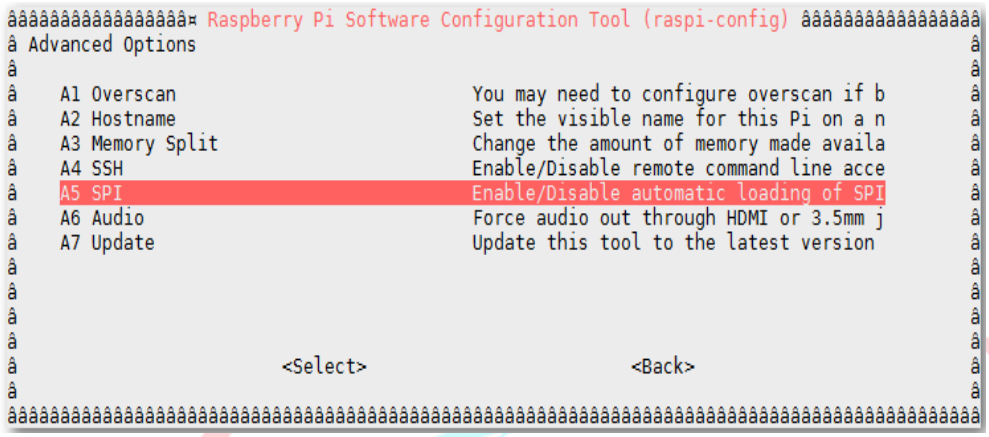


Figure 5

- **Set the option to “Yes”.**

Figure 6

- **Select “OK”.**

Figure 7

- **Select “Finish”**

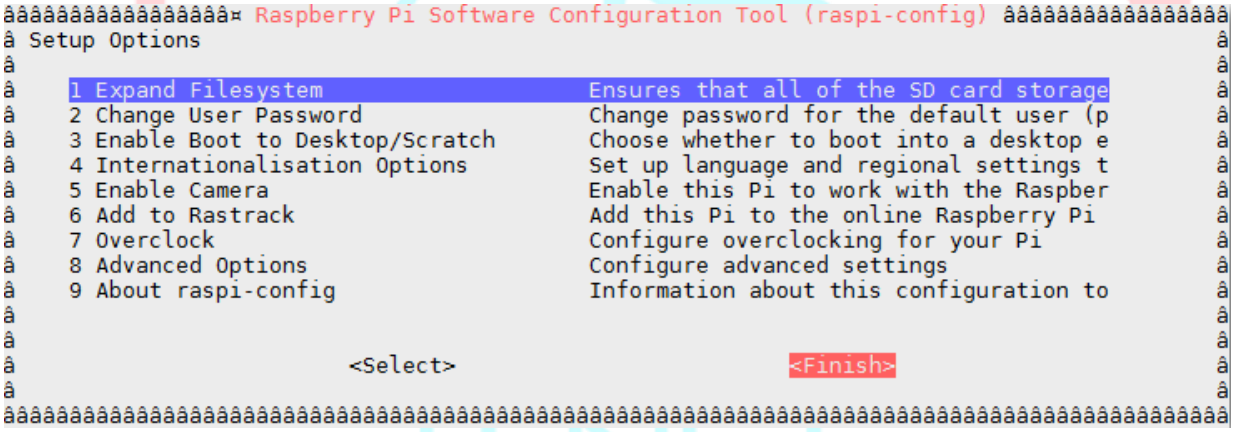


Figure 8

- **Reboot for the changes to take effect :**

  Sudo reboot

- **SPI is now enabled**.
- **In order to read data from the SPI bus in Python we can install a library called ‘py-spidev’. To install it we first need to install ‘python-dev’ :**

  Sudo apt-get install python2.7-dev

- **Then to finish we can download ‘py-spidev’ and compile it ready for use :**

  wget https://github.com/Gadgetoid/py-spidev/archive/master.zip

unzip master.zip

rm master.zip

cd py-spidev-master

sudo python setup.py install

cd ..

Raspberry Pin	Starter kit
7 & 11	LED's
15 & 33	Push Button
13	Buzzer
Channel 0 (MCP3008)	Potentiometer
Channel 1 (MCP3008)	LDR
Channel 2 (MCP3008)	LM35

Code:

LED Blinking

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode (GPIO.BOARD)
GPIO.setup (11,GPIO.OUT)
GPIO.setup (7,GPIO.OUT)

while 1:
    try:
        GPIO.output(11,GPIO.HIGH)
        sleep (1)
        GPIO.output (11,GPIO.LOW)
        sleep (1)
```

```
            GPIO.output(7,GPIO.HIGH)
            sleep (1)
            GPIO.output (7,GPIO.LOW)
            sleep (1)

    except KeyboardInterrupt:
            GPIO.cleanup()
```

Push Button

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode (GPIO.BOARD)
GPIO.setup(12,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
while 1:
    try:
            if GPIO.input(12)==True:
                    print "1"
            else:
                    print "0"
    except KeyboardInterrupt:
            GPIO.cleanup()
```

Buzzer

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode (GPIO.BOARD)
GPIO.setup (11,GPIO.OUT)
while 1:
```

```
try:
        GPIO.output(11,GPIO.HIGH)
        sleep (1)
        GPIO.output (11,GPIO.LOW)
        sleep (1)
except KeyboardInterrupt:
        GPIO.cleanup()
```

Measuring Temperature with an ADC

```
import spidev, time
spi = spidev. SpiDev()
spi. open(0, 0)
def analog_read(channel):
        r = spi. xfer2([1, (8 + channel) << 4, 0])
        adc_out = ((r[1] &3) << 8) + r[2]
        return adc_out
while True:
        reading = analog_read(0)
        voltage = reading * 3.3 / 1024
        temp_c = voltage /.01
        temp_f = temp_c * 9.0 / 5.0 + 32
        print("Temp C=%f\t\tTemp f=%f" % (temp_c, temp_f))
        time. sleep(1)
```

Measuring Potentiometer with an ADC

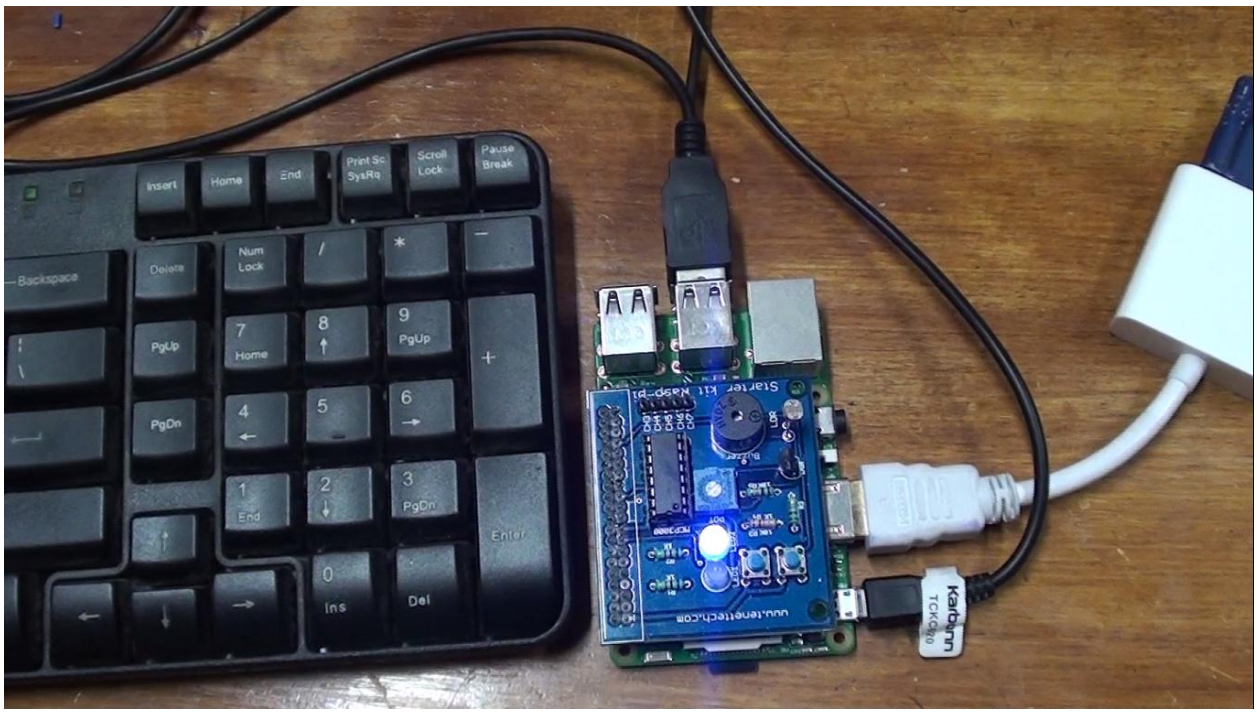
```
import spidev, time
```

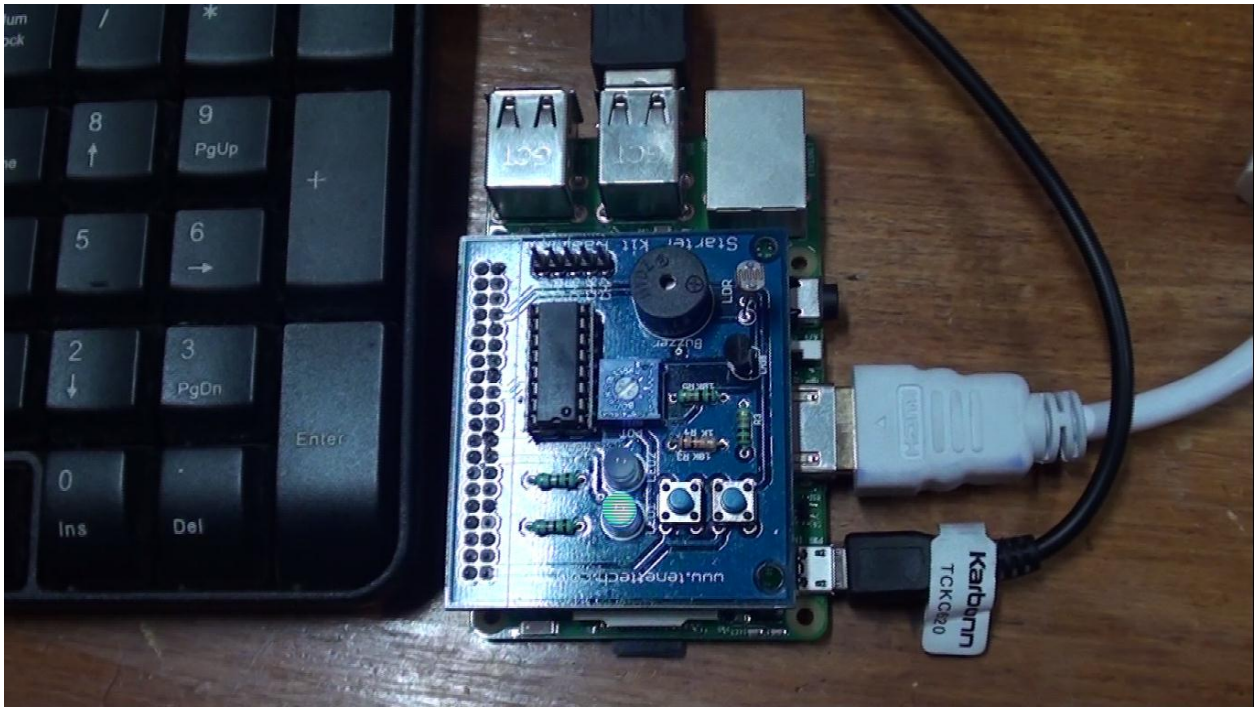
```
spi = spidev. SpiDev()
spi. open(0, 0)
def analog_read(channel):
        r = spi. xfer2([1, (8 + channel) << 4, 0])
        adc_out = ((r[1] &3) << 8) + r[2]
        return adc_out
while True:
        reading = analog_read(0)
        voltage = reading * 3.3 / 1024
        print("Potentiometer value =%d\t\tVoltage f=%f" % (reading, voltage))
        time. sleep(1)
```

Measuring LDR with an ADC

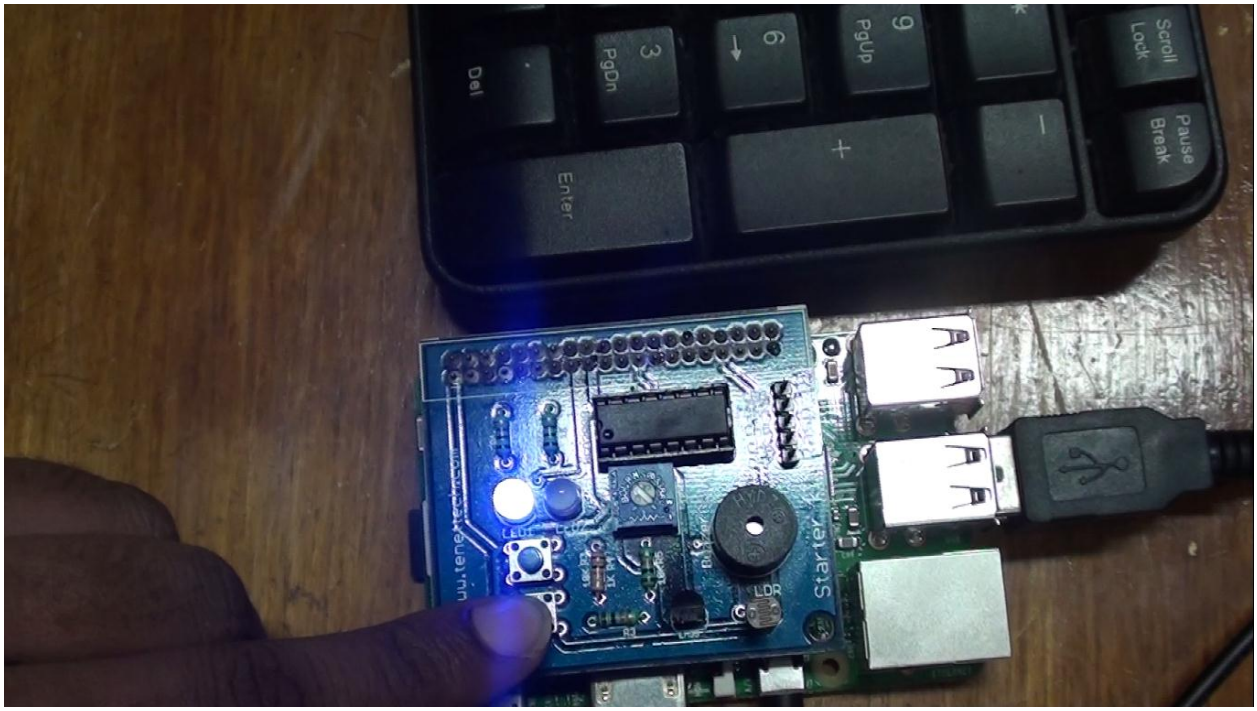
```
import spidev, time
spi = spidev. SpiDev()
spi. open(0, 0)
def analog_read(channel):
        r = spi. xfer2([1, (8 + channel) << 4, 0])
        adc_out = ((r[1] &3) << 8) + r[2]
        return adc_out
while True:
        reading = analog_read(0)
        voltage = reading * 3.3 / 1024
        print("LDR value =%d\t\tVoltage f=%f" % (reading, voltage))
        time. sleep(1)
```

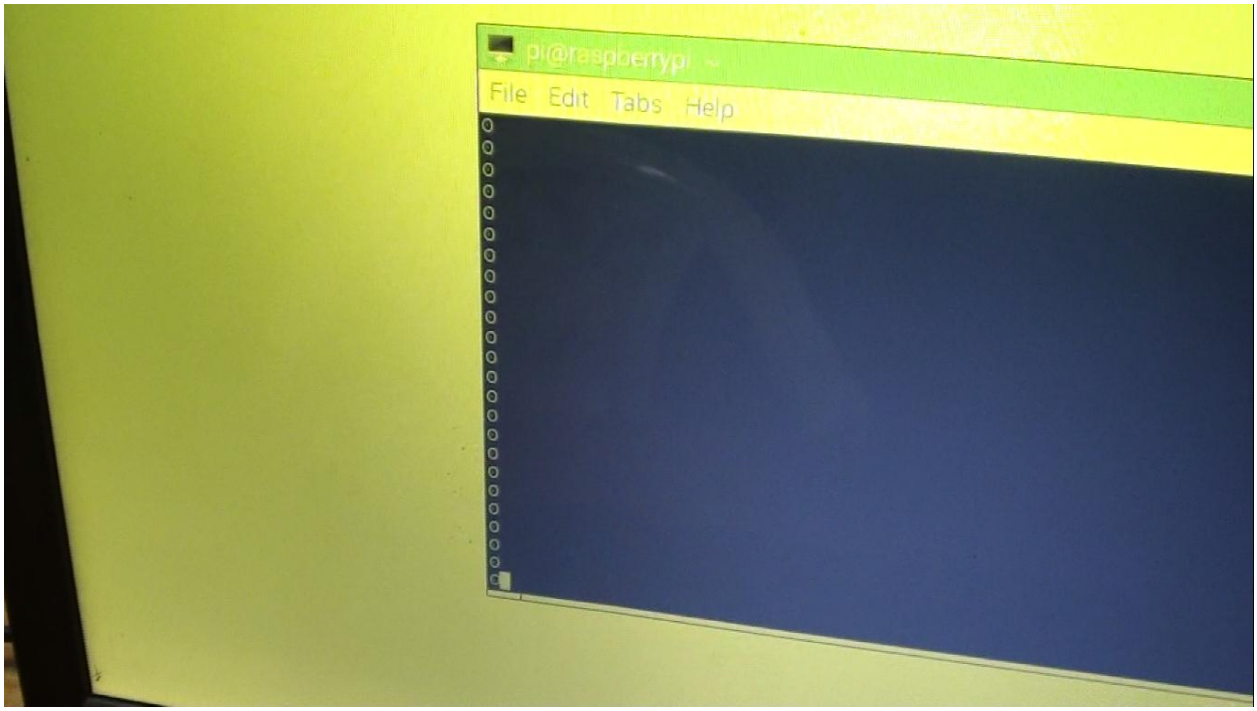
Experiment 01: LED interface

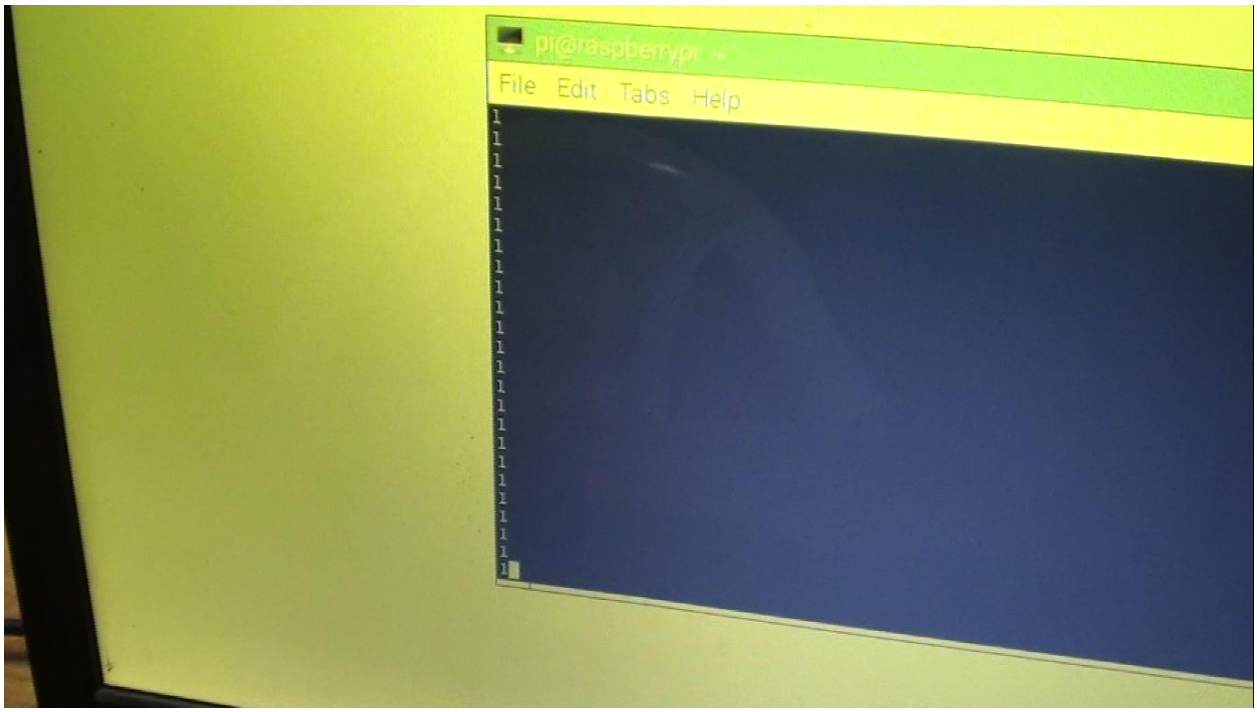




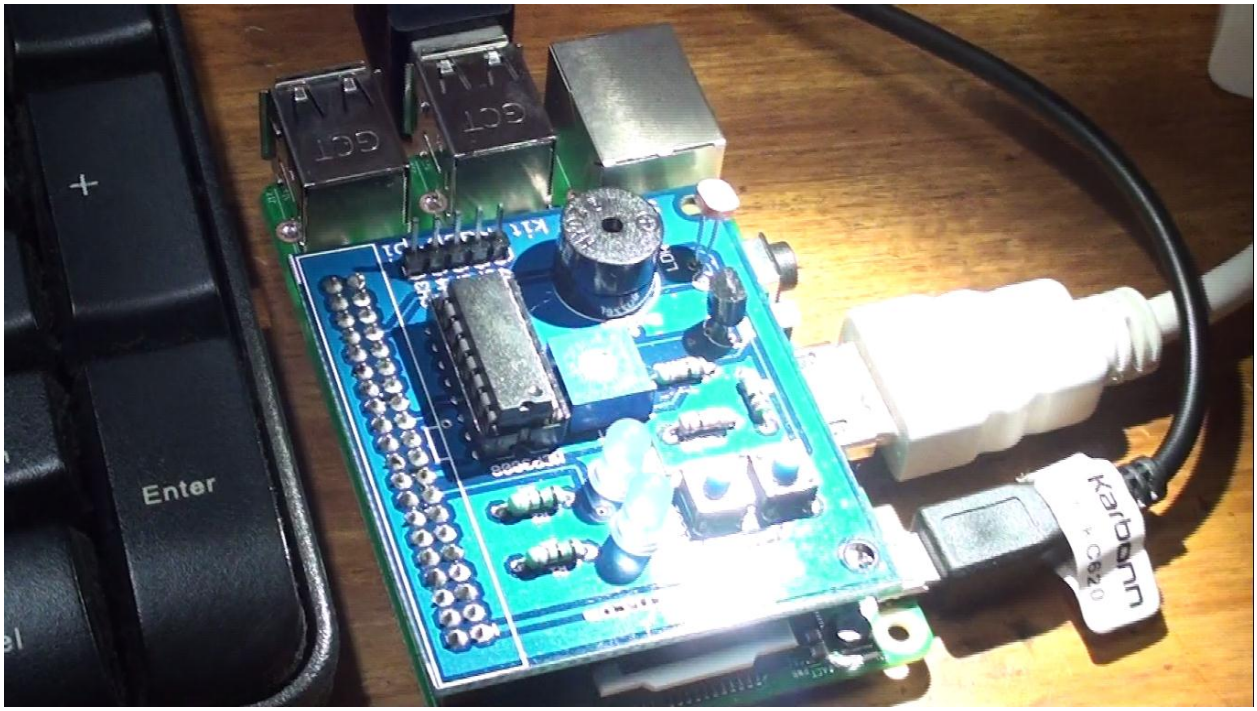
Experiment 02: Push Button

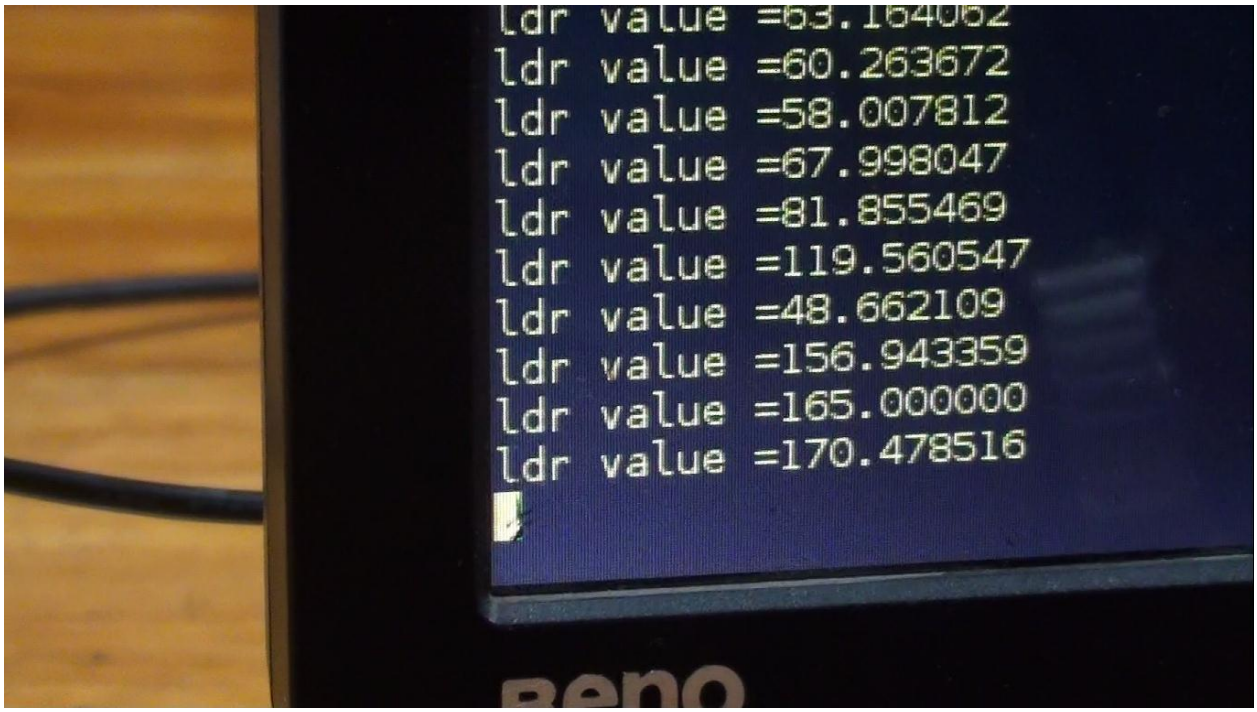




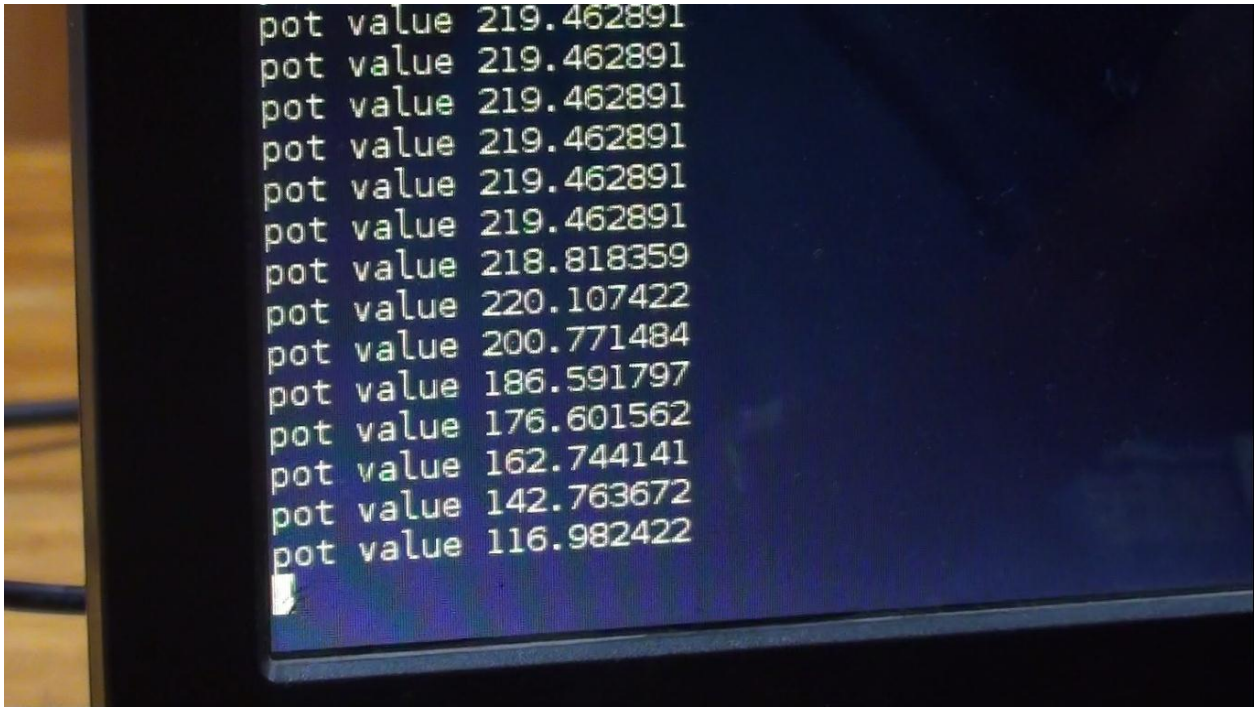


Experiment 03: Light sensor



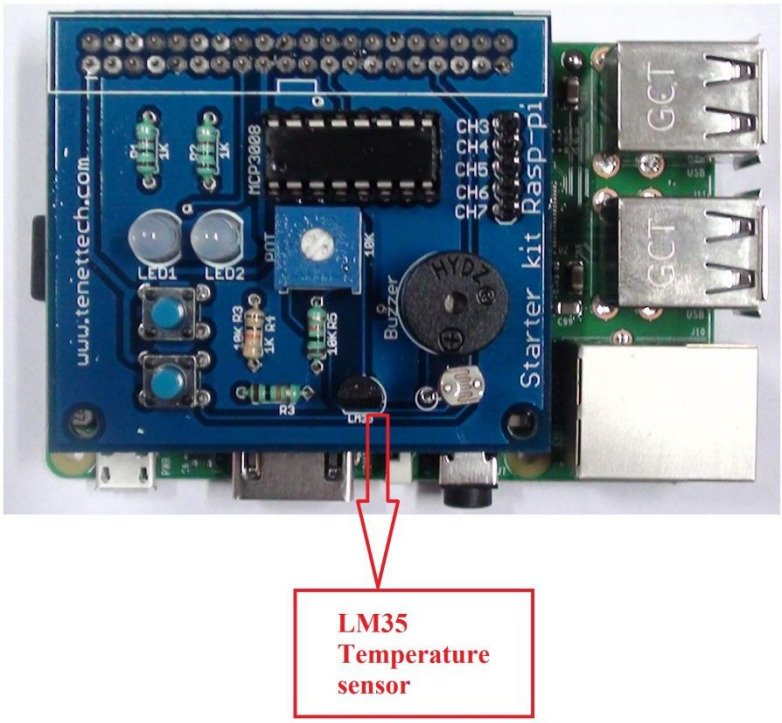


Experiment 04: Potentiometer

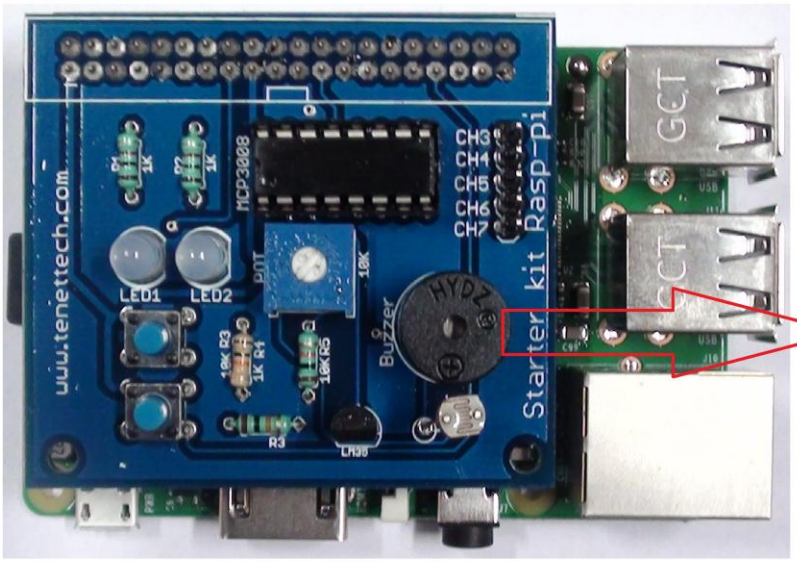


Experiment 05: Temperature sensor

```
pot value 329.677734
^CTraceback (most recent call last):
  File "spi_pot.py", line 14, in <module>
    time.sleep(1)
KeyboardInterrupt
pi@raspberrypi:~ $ sudo python spi.py
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
temp  C =30.615234 TEmp= 87.107422
```



Experiment 06: Buzzer



For more information please visit: www.tenettech.com

For technical query please send an e-mail: info@tenettech.com