Roll No: **Experiment 1** **Date:**

---

1. Aim: Flashing the OS on to the device into a stable functional state by porting desktop environment with necessary packages.

Components Required : Raspberry Pi kit , SD card , card Reader , Ethernet cable , keyboard , mouse , Operating system , Power supply , AC to DC Adapter , Socket, HDMI 2 VGA cable.

Procedure :

Step 1 : We need to insert SD card into a card reader and format it because to avoid unpredictable issues that may occur during read and write operation in device.

Step 2: After formatting the SD card then download the Raspberry Pi OS and Raspberry Pi Imager and copy the Raspberry Pi OS into the SD card.

Step 3: For interfacing the screen and the Raspberry Pi we need use HDMI 2 VGA cable. Connect the VGI cable to the CPU and other end is connected to Raspberry Pi kit .

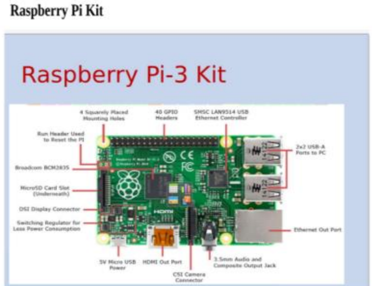
Step 4 : In order to give the input program we are connecting the Keyboard and mouse to Raspberry Kit USB ports. For internet access connect the Ethernet pin and on the power supply.

Step 5 : When the system starts opening ,for flashing the OS into Raspberry Pi some basic system settings must be done .

Step 6 : Go to the preferences and click on Raspberry Pi configuration and the whole description of Raspberry Pi Os is visible and go to interfaces and click Ok then we can observe the flashing of Raspberry Pi OS on Desktop.

Output :

Raspberry Pi Kit



Roll No : **Experiment-2** **Date :**

Aim: Accessing Graphical Desktop of Raspberry Pi Using SSH and VNC and displaying on to other systems.

Components Required:

- Raspberry Pi with an SD card running Raspbian operating system (OS) and connected with Ethernet cable or WiFi.
- Laptop running Linux OS—I prefer Ubuntu 16.04

Steps to be followed:

1. Connect all the required interfaces to the raspberry kit and start the system. After system has opened follow the following steps.
2. Accessing Graphical Desktop Of Raspberry Pi Using SSH And VNC Installing required packages on Raspberry Pi

Internet access is available on Raspberry Pi since Ubuntu connected to a Wi-Fi network has shared its connection over Ethernet. Run the following command to update the packages list from the repositories:

- *$ sudo apt-get update*

  Graphical desktop of Raspberry Pi can be shared using tightvncserver package. For that, run the following command:
- *$ sudo apt-get install tightvncserver*

  Run the following command to perform the initial set up of tightvncserver:
- *$ tightvncserver*

  Enter a suitable password for future connections.

  Run the following command to copy and paste from VNC server:
- *$ sudo apt-get install autocutsel*

  Add autocutsel -fork in /home/pi/.vnc/xstartup using the following command:
- *$ sudo nano /home/pi/.vnc/xstartup*

  Save it using Ctrl + x.

  Contents of the modified xstartup file can be viewed using the following command:
- *$ cat .vnc/xstartup*

  Restart VNC server for autocutsel to take effect, using the following command:
- *$ vncserver -kill :1 then command $ vncserver :1*

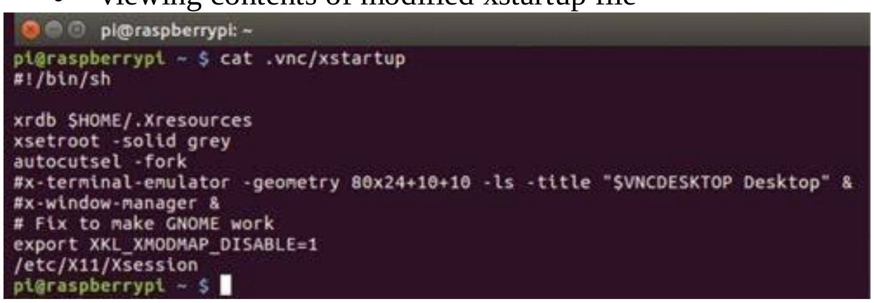
3. Accessing Raspberry Pi using VNC Viewer

   Although Raspberry Pi has the required packages installed to stream its graphical desktop, VNC Viewer is needed to access it. Since we are using an Ubuntu system to access Raspberry Pi, install VNC Viewer using the following command:

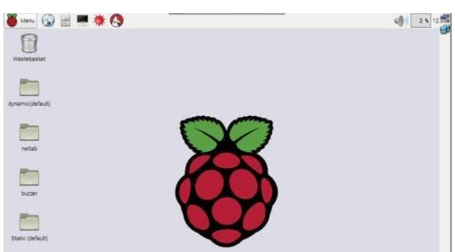
- *$ sudo apt-get install ssvnc*
  You may need to install xtightvncviewer using the following command:
- *$ sudo apt-get install xtightvncviewer*
  It is now possible to access the graphical desktop of Raspberry Pi using the following command:
- *$ xtightvncviewer 192.168.0.193:1*
  Replace 192.168.0.193 with the IP address of your Raspberry Pi. Enter the password to access it. Alternatively, the same results can be obtained using RealVNC Viewer package available on www.realvnc.com
  Raspberry Pi can be logged on over SSH using an Ethernet network with Ubuntu system. Graphical desktop of Raspberry Pi running VNC server can be accessed using VNC Viewer. Internet can be accessed on Raspberry Pi by sharing the Wi-Fi connection of Ubuntu system. This configuration lets you access Raspberry Pi from any remote location using a laptop and a regular Ethernet cable.

Output:

- viewing contents of modified xstartup file



- Raspbian OS home screen



Result:

In this way we access a graphical desktop of Raspberry Pi Using SSH And VNC Installing required packages on Raspberry Pi.

Roll No: **Experiment 3** **Date:**

Aim: Interface a push button to raspberry pi.

Components Required:

1) Raspberry pi

2) Push button

3) 5mm LED

4) 100-ohm resistor (1/4 watt)

5) Mini breadboard

6) Connecting wires

7) Power supply

Circuit Design:

1) First, instead of using a four terminal push button, use a two terminal push button. This won't make any difference.

2) One terminal of the push button is connected to GND, and the other terminal is connected to Physical Pin 16 (GPIO23) of Raspberry Pi.

3) A 5mm LED is used as an output device. The anode of the LED (long lead) is connected to Physical Pin 18 (GPIO24) of Raspberry Pi. The cathode of the LED (short lead) is connected to one terminal of a 100Ω Resistor.

Working:

1) Each GPIO pin in Raspberry Pi has software configurable pull-up and pull-down resistors. When using a GPIO pin as an input, you can configure these resistors so that one or either or neither of the resistors is enabled, using the optional pull_up_down parameter to GPIO.setup

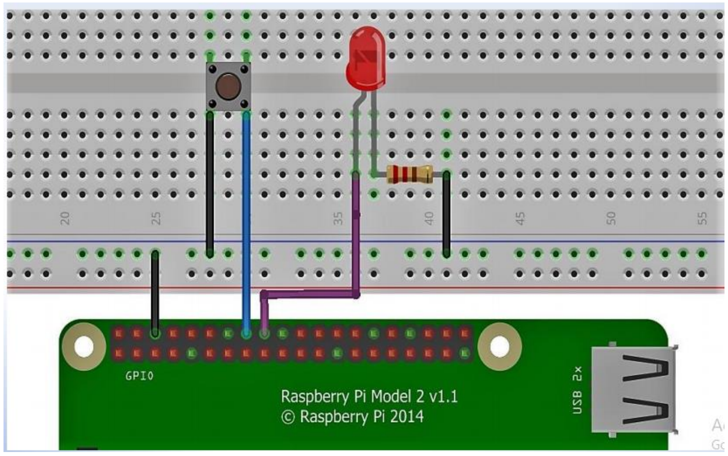
2) If it is set to GPIO.PUD_UP, the pull-up resistor is enabled; if it is set to GPIO.PUD_DOWN, the pull-down resistor is enabled.

3) 3) After running the code, when you push the Button LED should turn ON and in terminal window you will see the text "Button Pressed...". If not, then check your code and connections and try again.

Code:

```
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.SETUP(23, GPIO.IN, pull_up_down=GPIO.PUD_UP)

#button to GPIO 23

GPIO.SETU(23, GPIO.OUT)#led to GPIO 24

try:

	while True:

				button_state = GPIO.input(23)

				If button_state == false:

					GPIO.output(24, True)

					print('Button Pressed')

					Time.sleep(0.2)

				else:

GPIO.output(24, False)

Except:		GPIO.cleanup()
```

Ouptut:



Interface of a push button to raspberry pi.

Roll No.: **Experiment 4** **Date:**

4.<u>Aim</u> : To exploit the features of chronos ez430 by interfacing with raspberry pi

<u>Components Required</u>: Raspberry Pi kit, Chronos ez420 watch,SD card,Ethernet cable,Socket,Power supply,Operating system,HDMI 2 VGA cable.

<u>Procedure</u>:

1.Install few dependencies using the below command

$ sudo apt-get install python-serial tcl8.5 tk8.5 xdotool

2.To get access to an Intel/AMD Linux machine run the following commands

$ unzip slac388a.zip

$ ./Chronos-Setup

$ tar zcvf ccc.tgz ~/Texas Instruments/eZ430-Chronos

3.The ccc.tgz archive can then be copied to the Raspberry Pi and unpacked to a suitable location.

4.Set the watch to *ACC* mode and with RF enabled. Real-time data from the watch accelerometers is displayed, and by selecting *Mouse On* it's also possible to use the watch to control the Raspberry Pi mouse pointer through gesture.

5.The Control Center provides a number of other simple applications that can be selected via the tabs at the top.

<u>Result</u> :

Exploited the features of chronos ez430 by interfacing with raspberry pi

Roll No: **Experiment 5** **Date:**

Aim: Using the light sensors, monitor the surrounding light intensity & automatically turn ON/OFF the high intensity LED's by taking some pre-defined threshold light intensity value.

Components Required: raspberry pi, LED, HMDI, Keyboard, Mouse, LDR sensor, Night vision Camera, Resistor.

Procedure :

Step1: Connect the LED and the resistor to the breadboard.

Step 2: The LDR is connected with 3.3v and the other leg of the LDR is connected with the positive leg of the 10UF capacitor. while the other leg of the capacitor is connected with the ground.

Step 3: Now take a wire from the middle and connect it with pin number 7 of the raspberry pi. our circuit diagram is completed. So LED is connected with pin number 11 and the RC circuit is connected with pin number 7.

Step 4: Write the program that is below in the console and save it.

Step 5 :Now connect the Raspberry pi to the system where the program is written using a USB cable and run the program.

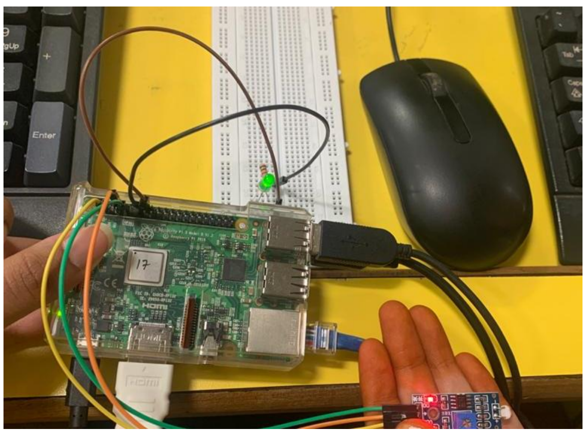
Step 5: Run the program and we see the change in intensity of the LED.

CODE:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
delayt = .1
value = 0 # this variable will be used to store the ldr value
ldr = 7 #ldr is connected with pin number 7
led = 11 #led is connected with pin number 11
GPIO.setup(led, GPIO.OUT) # as led is an output device so that’s why we set it to output.
GPIO.output(led, False) # keep led off by default
def rc_time (ldr):
    count = 0

    #Output on the pin for
    GPIO.setup(ldr, GPIO.OUT)
    GPIO.output(ldr, False)
    time.sleep(delayt)

    #Change the pin back to input
    GPIO.setup(ldr, GPIO.IN)

    #Count until the pin goes high
    while (GPIO.input(ldr) == 0):
        count += 1

    return count
```

```
#Catch when script is interrupted, cleanup correctly
try:
    while True:
        print("Ldr Value:")
        value = rc_time(ldr)
        print(value)
        if ( value <= 10000 ):
                print("Lights are ON")
                GPIO.output(led, True)
        if (value > 10000):
                print("Lights are OFF")
                GPIO.output(led, False)
except KeyboardInterrupt:
    pass
finally:
    GPIO.cleanup()
```

OUTPUT:

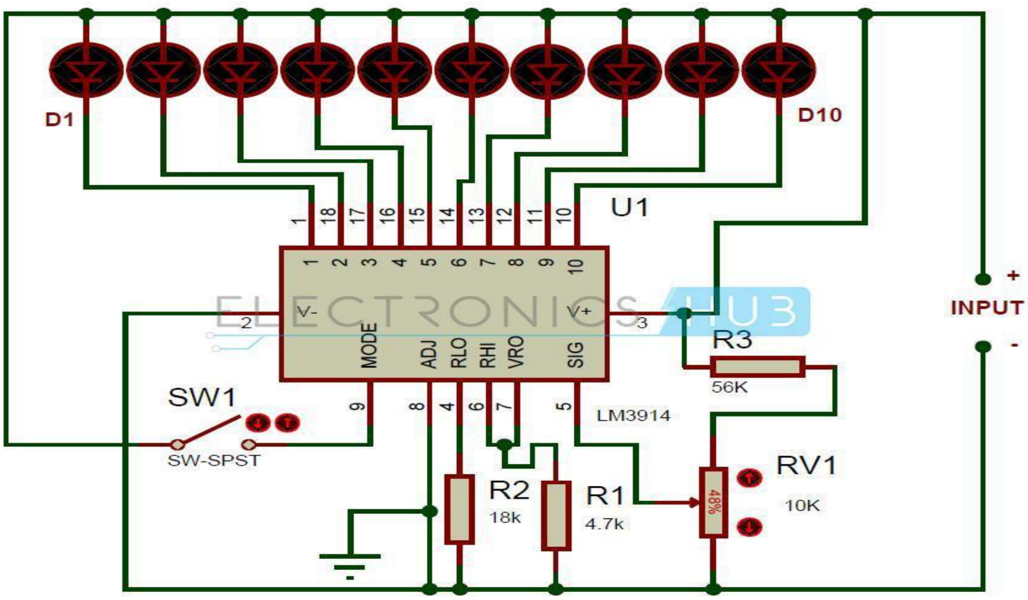


RESULT:

Monitored the surrounding light intensity using light sensors & automatically turn ON/OFF the high intensity LED.

Roll No. : 18WH1A12 **Experiment 6** **Date:**

6. Aim: Monitor the voltage level of the battery and indicating the same using multiple LED's



BATTERY LEVEL INDICATOR CIRCUIT DIAGRAM

Components Required:

- LM3914 IC
- LED's -10 (Red – 3, Yellow – 4, Green – 3)
- SPST Switch
- Resistors – 18KΩ, 4.7KΩ, 56KΩ
- Potentiometer – 10KΩ
- 12V Battery (to test)
- Connecting wires

Procedure:

- Connect battery to be tested to the input of the circuit.
- Now adjust the pot RV1 so that LED D1 just starts glowing.
- Now increase the input Dc voltage slowly and observe the LED's
- First led will glow for 1.2V and second LED is for 2.4 V and so on.

Below table shows the status of LED's with input voltage level

BATTERY LEVEL	PERCENT AGE	STATUS OF LEDS
1.2V	10	D1 - ON
2.4V	20	D1, D2 - ON
3.6V	30	D1, D2, D3 - ON
4.8V	40	D1, D2, D3, D4 - ON
6.0V	50	D1, D2, D3, D4, D5 - ON
7.2V	60	D1, D2, D3, D4, D5, D6 - ON
8.4V	70	D1, D2, D3, D4, D5, D6, D7 - ON
9.6V	80	D1, D2, D3, D4, D5, D6, D7, D8 - ON
10.8V	90	D1, D2, D3, D4, D5, D6, D7, D8, D9 - ON
12V	100	ALL LEDs - ON

Result: Monitored the voltage level of the battery and indicating the same multiple LED’s.

Roll No.: **Experiment 7** **Date:**

Aim: Instead of using the conventional dice, generate a random value similar to dice value and display thesame using a 16X2 LCD. A possible extension could be to provide the user with option of selectingsingle or double dice game.

Components Required: Arduino UNO, 16×2 LCD, Buzzer, BC547 Transistor, Push to ON switches, Resistances, USB Cable or 7805 IC, led

Procedure :

Step1: Connect the LED to the bread board.
Step 2: Now connect the LCD to the bread board using different wires. Also connect the buzzer and the arduino to the LED and LCD through the bread board.
Step 3:Write a program using the code written below for the random number to be generated whose functionality is similar to a dice.
Step 4 :Now connect the arduino to the system where the program is written using a USB cable and run the program.
Step 5: Run the program and we see a random number generated by the program on the LCD screen connected.

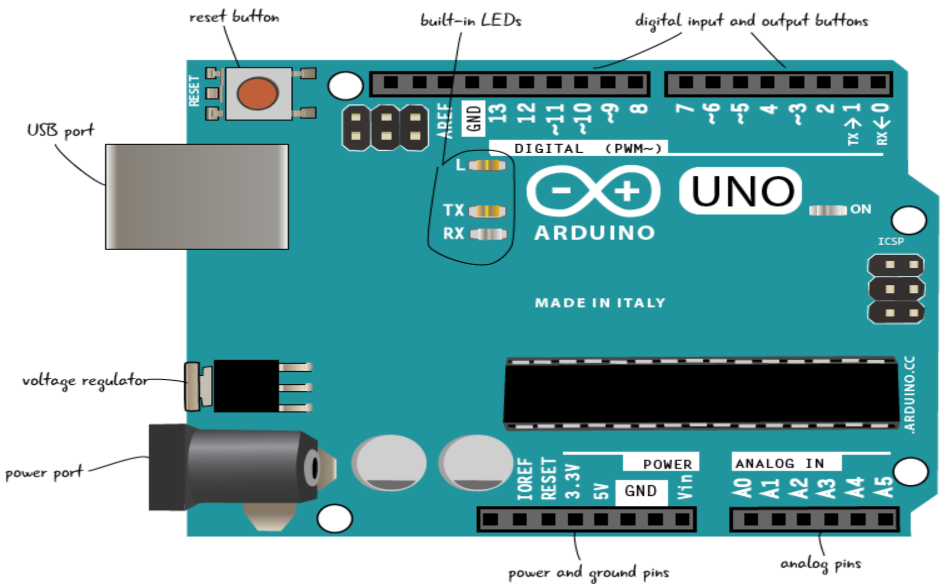
CODE:

```
#include <lt;LiquidCrystal.h>;
long randNumber;
int Led = 13; //define LED port
int Shock = 2; //define shock port
int val;//define digital variable val
// initialize the library with the numbers of the interface pins
LiquidCrystallcd(7, 8, 9, 10, 11, 12 );
byte customChar[] = {B00000,
B00000,
B11111,
B11001,



B10101,
B10011,
B11111,
B00000
```

```
};
void setup(){
        lcd.begin(16, 2);
        lcd.createChar(0, customChar);
        lcd.home();
        pinMode(Led, OUTPUT); //define LED as a output port
        randomSeed(analogRead(0));
        pinMode(Shock, INPUT); //define shock sensor as a output port
        lcd.write(byte( 0));
        lcd.print(“Digitaldice”);
        lcd.write(byte( 0));
        delay(1000);
}
void loop(){
        val = digitalRead(Shock); //read the value of the digital interface 3 assigned
        to val
        if (val == LOW){ //when the shock sensor have signal do the following
                lcd.clear();
                lcd.print(“Rolling dice...”);
                delay(4000);
                lcd.clear();
                lcd.setCursor(0, 0);


                randNumber = random(1,7);
                lcd.print(“Dice 1 = “);
                lcd.print(randNumber);
                lcd.setCursor(0, 1);
                randNumber = random(1,7);
                lcd.print(“Dice 2 = “);
                lcd.print(randNumber);
        }
        delay(150);
}
```

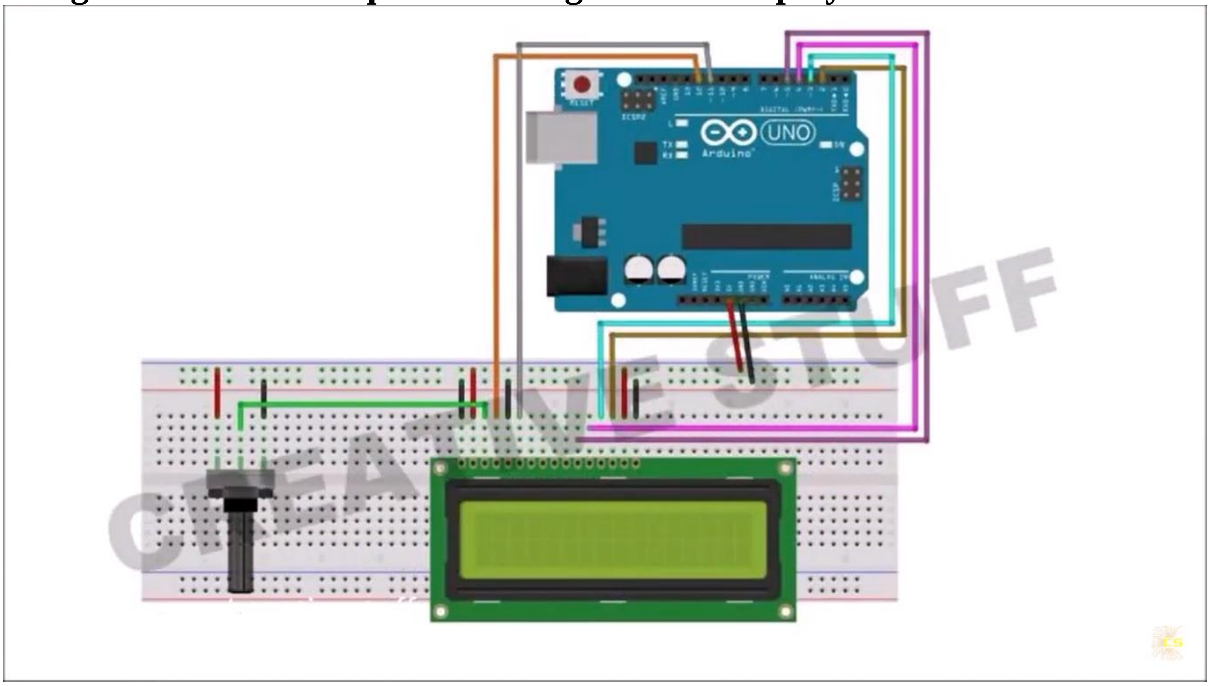
OUTPUT:

ARDUINO



16X2 LCD

Connecting the interfaces required for Digital Dice Display



RESULT:

In this way generate a random value similar to dice value and display the same using a 16X2 LCD.

Roll No. : **Experiment 8** **Date:**

8. Aim: Displaying the RSS news feed headlines on a LCD display connected to device. This can be adapted to other websites like twitter or other information websites. Python can be used to acquire data from the internet.

Components Required: Arduino Uno Board, Bread Board, LED, Jumper wires, Potentiometer, USB to Arduino Cable

Steps to be followed:

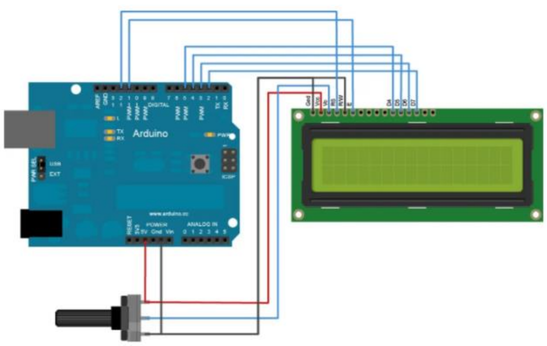
1. Connect the circuit as shown in the picture below. Potentiometer controls the contrast. It should also be noted that most LCD's use pins 15 and 16 on LCD as the +5v and GND for the backlight.
2. The LED just goes in digital pin 13 and the GND pin next to it.

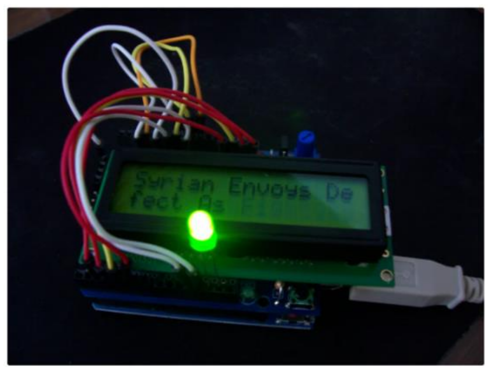
Python code:

```
#import library to do http requests:
import urllib2
#import pyserial Library
import serial
#import time library for delays
import time
#import xml parser called minidom:
from xml.dom.minidom import parseString
#Initialize the Serial connection in COM3 or whatever port your arduino uses at 9600 baud rate
ser = serial.Serial("\\.\COM3", 9600)
i = 1
#delay for stability while connection is achieved
time.sleep(5)
while i == 1:
    #download the rss file feel free to put your own rss url in here
    file = urllib2.urlopen('http://news.sky.com/feeds/rss/world.xml')
    #convert to string
    data = file.read()
    #close the file
    file.close()
    #parse the xml from the string
    dom = parseString(data)
    #retrieve the first xml tag (<tag>data</tag>) that the parser finds with name tagName change
tags to get different data
    xmlTag = dom.getElementsByTagName('title')[2].toxml()
 # the [2] indicates the 3rd title tag it finds will be parsed, counting starts at 0
  #strip off the tag (<tag>data</tag>  --->   data)
xmlData=xmlTag.replace('<title>','').replace('</title>','')
#write the marker ~ to serial
```

```
ser.write('~')
time.sleep(5)
#split the string into individual words
nums = xmlData.split(' ')
#loop until all words in string have been printed
for num in nums:
    #write 1 word
    ser.write(num)
    # write 1 space
    ser.write(' ')
    # THE DELAY IS NECESSARY. It prevents overflow of the arduino buffer.
    time.sleep(2)
# write ~ to close the string and tell arduino information sending is finished
ser.write('~')
# wait 5 minutes before rechecking RSS and resending data to Arduino
time.sleep(300)
```

Output:





Roll No: **Experiment 9** **Date:**

Aim : Attempt to use the device while connecting to a wifi network using a USB dongle and at the same time providing a wireless access point to the dongle.

Software Required:

- OpenWrt Image for Raspberry Pi
- 7z SD Card Formatter
- Etcher
- Putty
- Wireless Network Watcher
- WinSCP

Equipment Required:

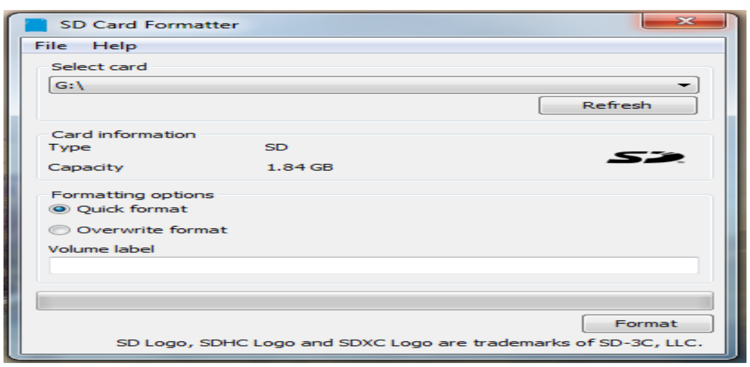
- Raspberry Pi Model B
- SD Card
- Ethernet Cable Power Adapter

OpenWrt Setup on Raspberry Pi Image Download

- First**,** to start this project we need to download the OpenWrt Raspberry Pi Image from the download link.
- After that, we need to flash the image to the SD Card**.**

Formatting the SD Card

- Now, We need the SD Card Formatter tool to format the SD card.



Flashing the SD Card

- After that, write the image to the SD Card using: Etcher

Setting up Raspberry Pi

- After that plugin, the SD Card, connect the Ethernet Cable and Power Supply to complete the setup.
- Followed by turn on the Raspberry Pi device it will take up to 2 minutes to boot the OpenWrt OS.
- Then run Wireless Network Watcher to check the IP Address of Raspberry Pi.
- Now, you can directly access your OpenWrt router page by typing.

OpenWrt Default IP Address : 192.168.1.1

Setup fixed IP Address using CLI [Command Line Interface]

- To set up a Static IP Address to our OpenWrt Router by using commands.
- We need to login to our router by using Putty and type the Router IP Address for router login

On the first login, we need to set a password for our OpenWrt Router because there is No default Password for this image.

Command to change the Password: psswd

- To set a new password to enter the new password 2 Time.

OpenWrt Default Username: root

OpenWrt Default Password: blank

- Then we need to type the following commands to setup a static IP address.

Uci set network.lan.ipaddr = 192.168.1.2
uci commit /etc/init.d/network restart

- After the commands successfully executed we need to reboot the OpenWrt Router.
- Reboot OpenWrt.

reboot -f

- Once the reboot finish, you can log in to your OpenWRT router at https://192.168.1.2

Update the Openwrt Router

- To update the OpenWrt router we need to change the static IP address to DHCP.

To change the IP address to DHCP we need to login to our OpenWrt router page i.e. available http://192.168

- Then we need to choose Network –> Interfaces and from here we need to edit the LAN connection. Because we are connected to our main router using the LAN connection.
- After we need to change the protocol to DHCP and click on switch protocol. Once it has done that we need to save the changes and we will get a new IP address from our main router.
- We can find the new IP Address by using Wireless Network Watcher and connect to our OpenWrt router easily.
- After that, we need to use this IP address to connect OpenWrt using Putty and then we can update our router using update command :

opkg update

Setup GUI for OpenWrt [In case of No GUI] Optional

- If you are not able to login to router GUI then you need to follow the below instructions.
- Use putty with default OpenWrt IP Address 192.168.1.1 and then type these commands.
- pkg update opkg install luci

If SSL

- pkg install luci-ssl
- After the installation has completed, the LuCI web GUI will be available at https://192.168.1.1

Result

You are all set to use OpenWrt on your Raspberry Pi.

Roll No. : 18WH1A12 **Experiment - 10** **Date:**

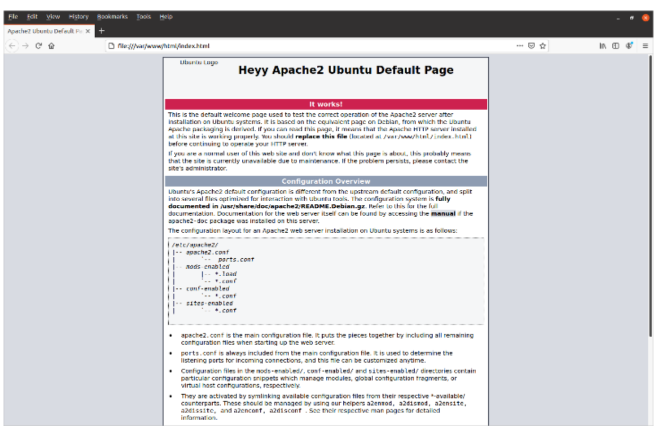
---

1. <u>Aim:</u>Building and hosting a simple website(static/dynamic) on the device and make it accessible online.There is a need to install server(e.g: Apache) and thereby host the website

<u>Procedure</u>:

Step 1: open terminal and type
sudo apt-get update
Step 2: next, install Apache
sudo apt-get install apache2 -y
Step 3: a default index.html page will be created in /var/www/html folder
if your able to see that page means then the Apache server installed
Successfully
Step 4: change the file permissions using -
sudochmod 777 html folder to create new files
Step 5: to edit index file
sudochmod 777 index.html in the specified folder
Step 6: create your own webpage and see the result in browser

Output :



➔ "Heyy" in the above file has been inserted/modified in the page.

Roll No. : **Experiment 11** **Date:**

Aim: Interfacing the regular usb webcam with the device and turn it into fully functional IP webcam & test the functionality.

Components Required:

1) Web-cam

2) 2) Raspberry-Pi kit

Procedure:

open terminal and type

1. sudo apt-get update

2. sudo apt-get upgrade

3. lsusb : to know whether device is detected or not

4. sudo apt-get install motion : which installs related webcam motion config file

5.sudo nano /etc/motion/motion.conf

opens conf file we have to set the following parameter in config file

Find the following lines and update them to the following.

daemon on

stream_localhost off

framerate 1000

Optional (Don’t include the text after the #)

stream_maxrate 100 # This change will allow for real-time streaming but requires more bandwidth & resources. Needs to be added to the config file, default is 1 framerate 100 # Changing this option will allow for 100 frames to be captured per second allowing for smoother video, default is 50.

width 640 #This line changes the width of the image displayed, default is 640

height 480 #This option changes the height of the image displayed, default is 480

6. sudo nano /etc/default/motion

start_motion_daemon = no

make it yes

start_motion_daemon=yes

and save and exit

7 sudo service motion start

8. If you need to stop the service, simply run the following command:

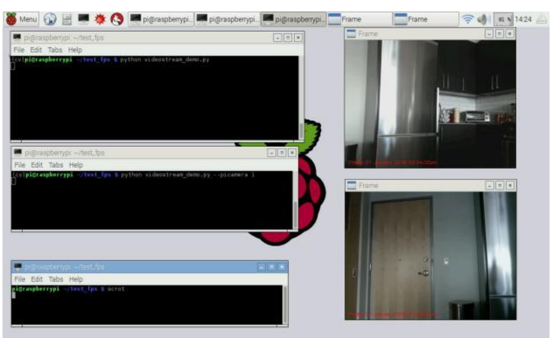
sudo service motion stop

8. Now you should be able to check out the Webcam Stream at the IP address of our Pi so in your browser go to the following address.

192.168.9.115:8081 or localhost:8081

Output:

Video Stream using webcam and Raspberry Pi



Roll No. : 18WH1A12 **Experiment – 12** **Date:**

Aim: Transforming the device into a regular fm transmitter capable of transmitting audio at desired frequency (generally 88-108 Mhz)

Apparatus Required:

1)Transistors - 2N3904 - 2

2)Resistors - 100k Ω, 100Ω, 1M Ω, 1k Ω, 10k Ω

3)Inductors **-** 0.1µH inductor (Air coil)

4)Capacitors – 0.1µF,40 pf trimmer, 4.7 pF – 1, 10pF – 1

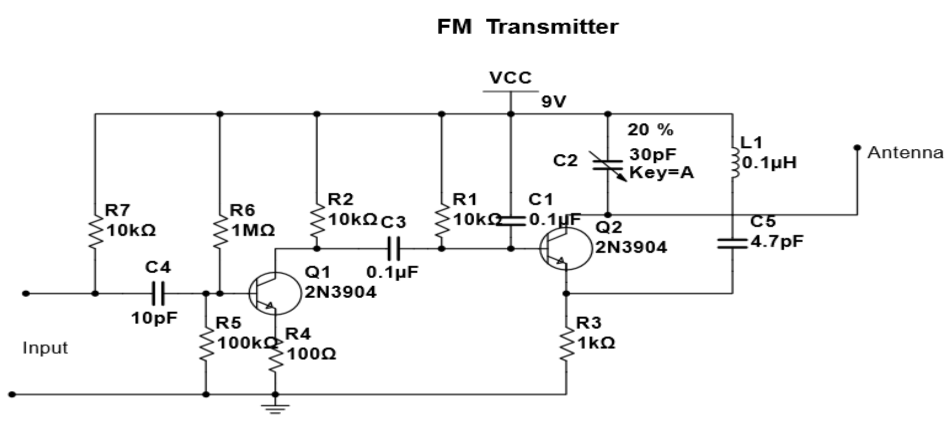
5) Antenna

6)9V battery and clip

7)PCB

Procedure

Circuit Diagram:



Step 1: Making PCB

1. Take the copper clad
2. Remove the dust by rubbing the copper clab with a scrubber
3. Now draw the layout of the circuit using the permanent marker or iron the prined glossy paper on the copper clad.
4. Add ferric chloride powder to water and mix it well.
5. Now place PCB in the solution until unwanted copper is dissolved
6. Clean PCB with dry cloth
7. Place PCB on support and drill the holes into it.

Step 2:Circuiting

1. Once the PCB is prepared insert the components in to pcb according to the circuit and solder it.
2. Now we need to make Inductor, Take a copper wire of 18 gauge or 22 gauge.
3. For 18 Gauge wire, form a inductor with 4-5 turns of 1/4 inch (or)for 22 Gauge wire, form a inductor with 8-10 turns of 1/4 inch.
4. Now solder the Inductor to the circuit,
5. If we have antenna, solder it or take hook up wire of 8-10cm as antenna.
6. We have to use 3.5mm female audio jack, because we can frequently plugin mic,audio devices easily.
7. If we use mic,it senses the audio and broadcast to nearbyFM radio. It can also be used as spy bug.

Step 3: Tuning Transmitter

1. Now the time to tune the transmitter, which is very hard and time taken process. Be patience while tuning.
2. By varying the trimmer capacitor ,you can vary the transmission frequency.

3. Slowly vary the trimmer capacitor, then at a point you can hear some distortion in radio.

4. Then slowly vary in that area, when transmitter and receiver frequency matches you can get the clear output from radio.

5. By tuning the frequency , making of FM transmitter is completed.

Output: