Assignment 5

Name : Shardul Vikram Dharmadhikari

Class : TE Comp

Roll : 8016

Title:

Understanding the connectivity of Raspberry Pi board with camera. Writing an application to capture and store the image.

Aim/Objectives:

- To understand the working of Raspberry Pi Camera
- To interface Raspberry Pi Camera with Raspberry Pi model
- To program the Raspberry Pi model to control the Raspberry Pi Camera Preview
- To program the Raspberry Pi model to capture still images from the Raspberry Pi Camera

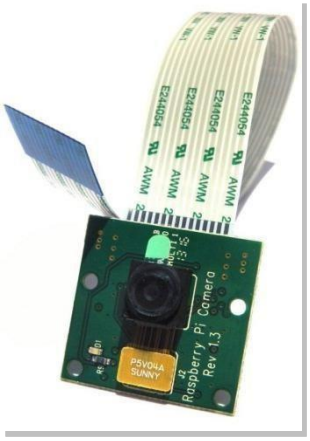
Software:

- Raspbian OS
- IDLE IDE

Hardware Modules:

- Raspberry Pi Board module
- Pi-Camera module
- Monitor

Theory:



- The Raspberry Pi Camera Board plugs directly into the CSI connector of the Raspberry Pi.
- It is able to deliver a crystal clear 5MP resolution image or 1080p HD video at the recording speed of 30 fps.
- This camera is designed and manufactured by the Raspberry Pi Foundation in the UK.
- The Raspberry Pi Camera Board features a 5MP (2592×1944 pixels) Omni vision 5647 sensor in a fixed focus module.

- This module is attached to Raspberry Pi, by way of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI), which was designed especially for interfacing to cameras.
- The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data to the BCM2835 processor.
- The board itself is tiny, at around 25mm x 20mm x 9mm, and weighs just over 3g, making it perfect for mobile or other applications where size and weight are important.
- The sensor itself has a native resolution of 5 megapixels, and has a fixed focus lens onboard.
- In terms of still images, the camera is capable of 2592 x 1944 pixel static images, and also supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 video recording.
- The camera is supported in the latest version of Raspbian, the Raspberry Pi's preferred operating system.

Safety precautions:

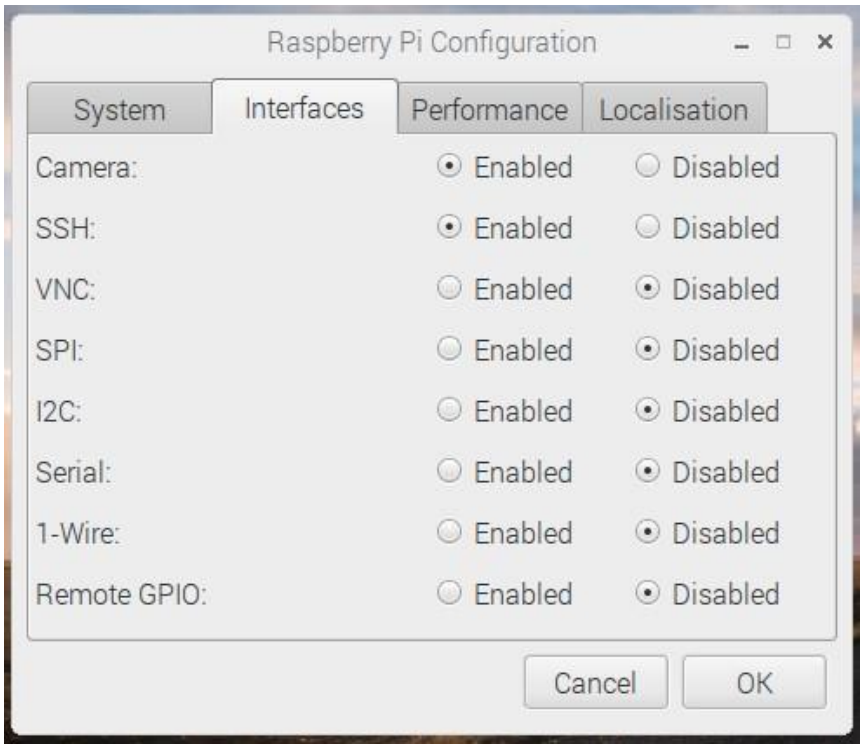
- Raspberry-Pi provides 3.3V and 5V VCC pins Raspberry-Pi operates on 3.3V.
- Various sensors and actuators operate on different voltages.
- Read datasheet of a given sensor or an actuator and then use appropriate VCC pin to connect a sensor or an actuator.
- Ensure that signal voltage coming to the Raspberry-Pi from any sensor or actuator does not exceed 3.3V.
- If signal/data coming to Raspberry-Pi is greater than 3.3V then use voltage level shifter module to decrease the incoming voltage.
- The Raspberry-Pi is a costly device, hence you should show the circuit connections to your instructor before starting your experiment.

Interface diagram:

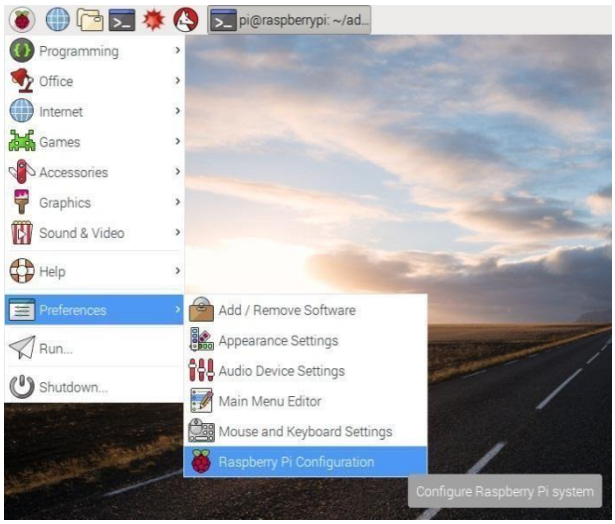
Procedure:

- First of all, switch off the Raspberry Pi board and connect the Camera Module to the Raspberry Pi's camera port.

- Then start the Raspberry Pi
- Now we have to ensure that the Camera software is enabled.
- For this, open the **Raspberry Pi Configuration Tool** from the main menu as shown in the fig. below.



- Now click on Interfaces
- Here the first option is Camera. Ensure that the 'Enabled' button is clicked.
- If not then click the 'Enabled' and again **Reboot the Raspberry Pi module**.
- Now write the program as per the algorithm given. Run code using Run module.



Algorithm:

- To program the Raspberry Pi model to control the Raspberry Pi Camera Preview:

  - Import Picamera library o Import time library o Create a variable(instance) of PiCamera

class o Display the camera preview on screen using the command "start_preview()". ○ We can define 10 second delay to see the camera preview. o To stop camera preview after 10 second, use the command "stop_preview()".

- To program the Raspberry Pi model to capture still images from the Raspberry Pi Camera ○ Import picamera library o Import time library o Create a variable(instance) of PiCamera class o Display the camera preview on screen using start_preview(). o We can define 5 second delay to see the camera preview. o Capture the image using camera.capture('path of the image. extension') o Then stop the camera preview using the command "stop_preview()".

Observation:

- Observe the output on Display screen of raspberry pi desktop for camera preview, captured images and recorded videos.
- To play the video type following command in terminal window: omxplayer video.h264
- Then press ENTER.

Code:

```
[#1]
from picamera import PiCamera
import time

camera = PiCamera()
camera.resolution = (1280, 720)

camera.start_preview() camera.annotate_text
= "This is Samyak Shah"
camera.start_recording('video.h264') time.sleep(10)
camera.stop_recording()
camera.stop_preview()


[#2] import picamera, time
import smtplib,
os
from email.MIMEMultipart import MIMEMultipart
from email.MIMEBase import MIMEBase from email.MIMEText
import MIMEText
from email import Encoders

camera = picamera.PiCamera() camera.start_preview() time.sleep(5)
# hang for preview for 5 seconds
os.remove('snapshot.jpg') camera.capture('snapshot.jpg')
camera.stop_preview()
```

```
server = smtplib.SMTP('smtp.gmail.com', 587)
server.ehlo() server.starttls()  #Next, log in to the server
server.login("sam@gmail.com", "passwrd")
#Send the mail msg = MIMEMultipart()
msg['Subject'] = "New person is coming..."
msg['From'] = "sam@gmail.com"
msg['To'] = ', '.join("sam2@gmail.com")

part = MIMEBase('application', "octet-stream") part.set_payload(open("snapshot.jpg", "rb").read())
Encoders.encode_base64(part)

part.add_header('Content-Disposition', 'attachment; filename="snapshot.jpg"')

body = "Open lock from here: http://192.168.43.164/servo.php" msg.attach(MIMEText(body, 'plain'))  msg.attach(part)

server.sendmail("sam@gmail.com", "sam2@gmail.com",msg.as_string())


[#3]  import
picamera, time

camera = picamera.PiCamera()

camera.start_preview()

time.sleep(5) # hang for preview for 5 seconds

camera.capture('snapshot.jpg')

camera.stop_preview()
```