SE-IOT: Internet of Things

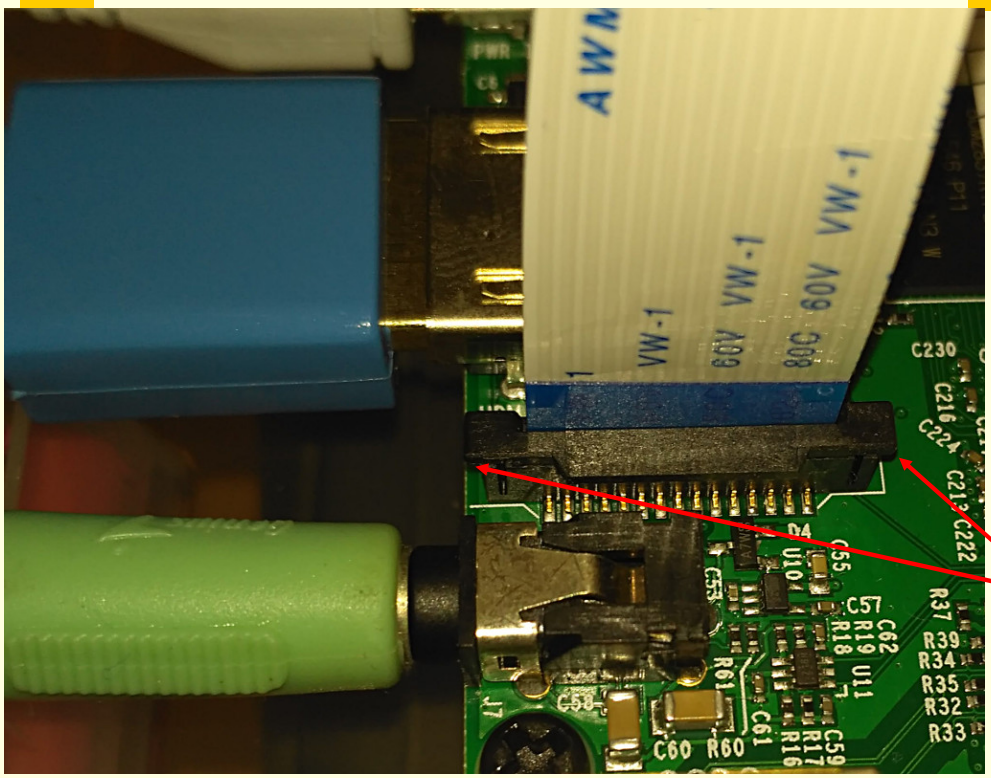
Media Devices

Derek Kiong
dkiong@nus.edu.sg

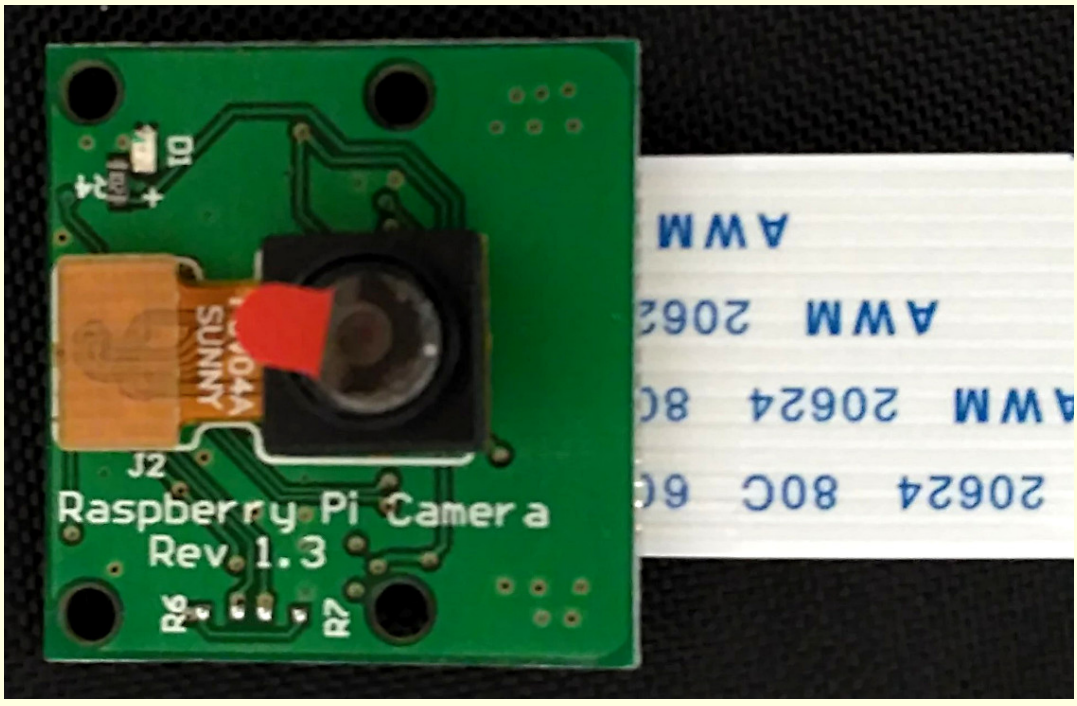
Incorporating additional devices

- Device drivers
  - GPIO
  - I²C
  - 1Wire
- Optional camera extension
  - connected via ribbon cable
  - must be enabled in RPi configuration
- Built-in audio chipset (only audio out)
  - Choose between 3.5□jack or HDMI output
- USB Audio/Microphone/Camera
  - Additional sound card or microphone
- USB RFID reader/writer
  - character device abstraction

Camera connection



Camera Module



Camera connection via ribbon strip

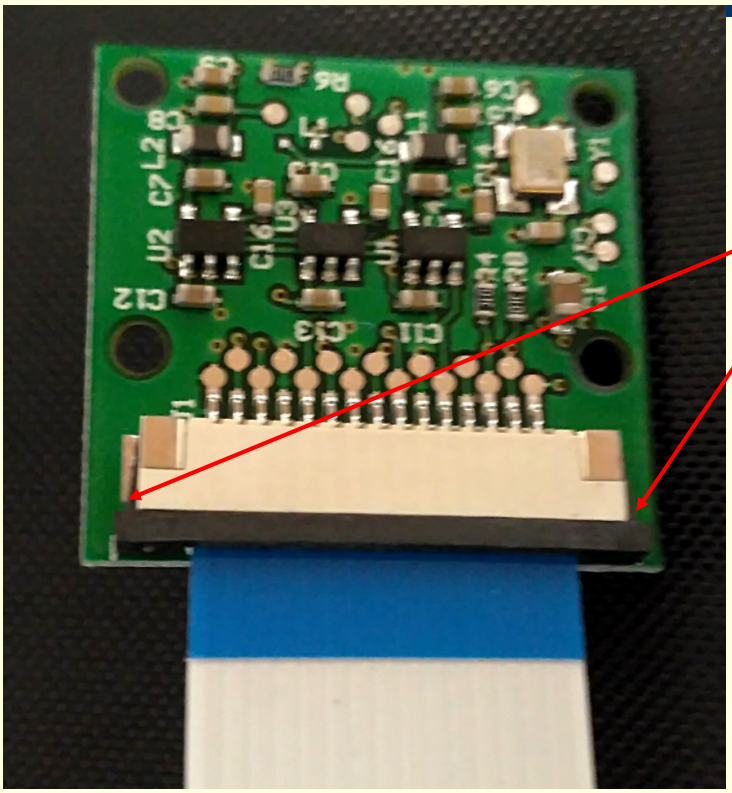


Image Capture

- Still image achieved via **`rapistill`** command
  - **`raspistill`** without options gives help summary
- **`picamera`** is the Python library for customised actions

```
import time, picamera

camera = picamera.PiCamera()
camera.start_preview()
time.sleep(5000)
camera.capture("image1.jpg")
```


Audio

- **aplay** command plays an audio clip
- **arecord** command records audio
  - (see command options in **man arecord**)
- **alsamixer** allows configuration of audio device

Audio Streaming

- Install **mpc** and **mpd** packages (for music daemon and controller)
- Add Internet radio URLs
- Initiate via **mpc** controller

```
sudo apt-get install mpd mpc
mpc clear
mpc add http://mediacorp.rastream.com/905fm
mpc add http://mediacorp.rastream.com/950fm
mpc play
mpc next
mpc stop
```


Device abstraction

- Character device as stream (**`/dev/ttyUSB0`**)

Local Integration via Pipes

- Python library **`os.system()`** and **`subprocess.Popen()`** provides adequate integration mechanisms

```
import subprocess

p = subprocess.Popen(['tr','[a-z]','[A-Z]'],
                     stdin=subprocess.PIPE)
f = p.stdin
f.write("This is a line\n")
f.write("Second line\n")
f.close()
```


Remote Integration

- **`ssh`** forms basis for **`scp`** and encrypted tunnels (**`-L`** for forward tunnel; **`-R`** for a backward tunnel)
- **`ssh`** key-based authentication extends usage beyond a remote shell

```
import os

# capture image1.jpg and send over to NSCC
os.system("scp image1.jpg userid@nus.nuscc.sg:p1.jpg")
```


Summary

- Components accessed via interface abstraction (GPIO, I2C etc)
- Device drivers
- Character devices accessed as files (**`/dev/ttyUSB0`**)
- Local pipes
- Network streams

