IOT with Raspberry PI

Chapter 4

Building IOT with Raspberry PI

- Single-board mini computer called Raspberry Pi
- It is widely accessible, inexpensive, and available from multiple vendors
- Extensive information is available on their programming and use both on the Internet and other mediums
- Building blocks of a generic single-board (SBC) based computer IoT device
- It includes CPU, GPU, RAM, storage and various types of interfaces and peripherals
- Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do
- Raspberry Pi allows interfacing sensors and actuators through the general purpose I/O pins
- Since Raspberry Pi runs Linux operating system, it supports Python “out of the box”

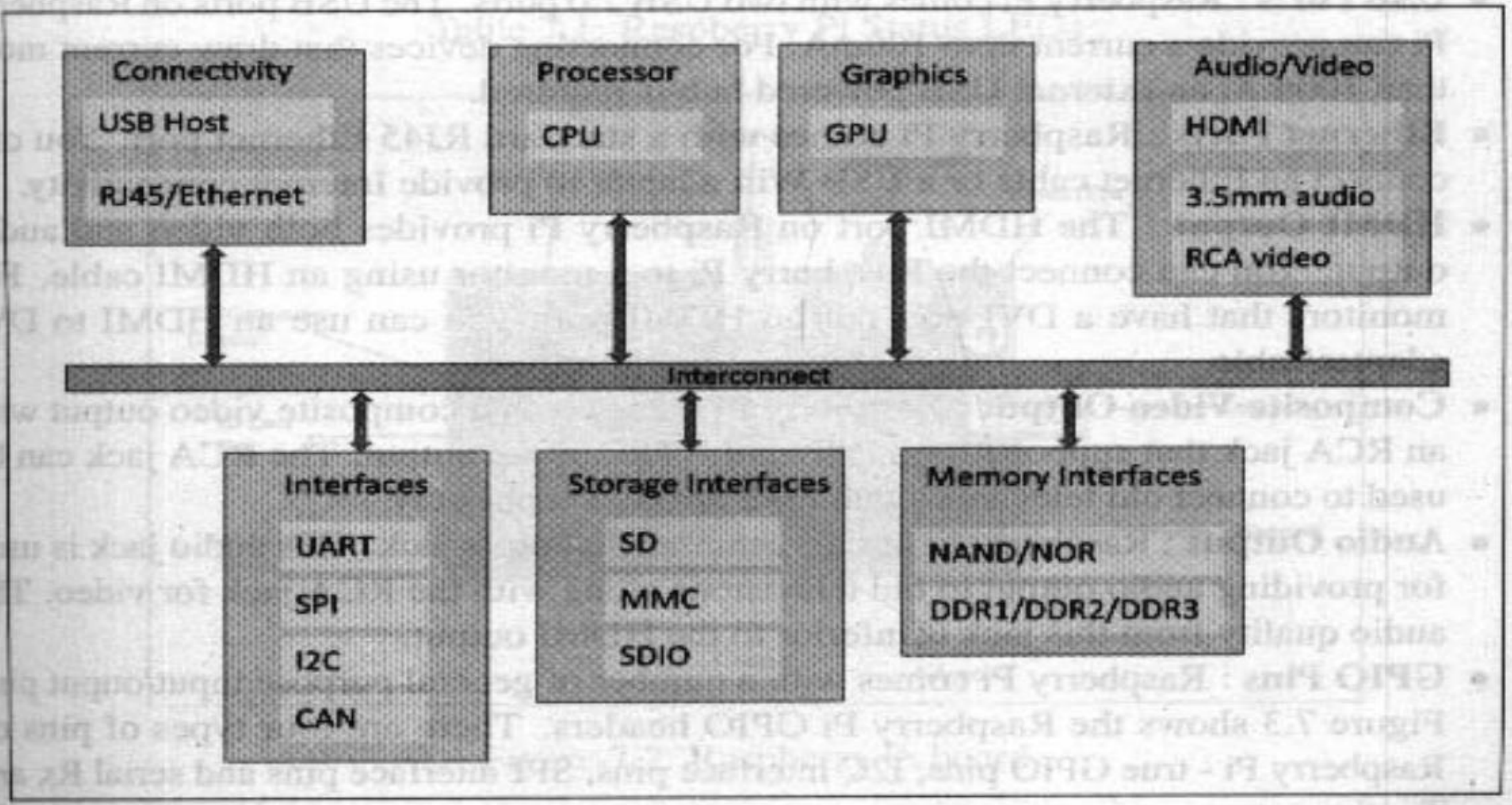
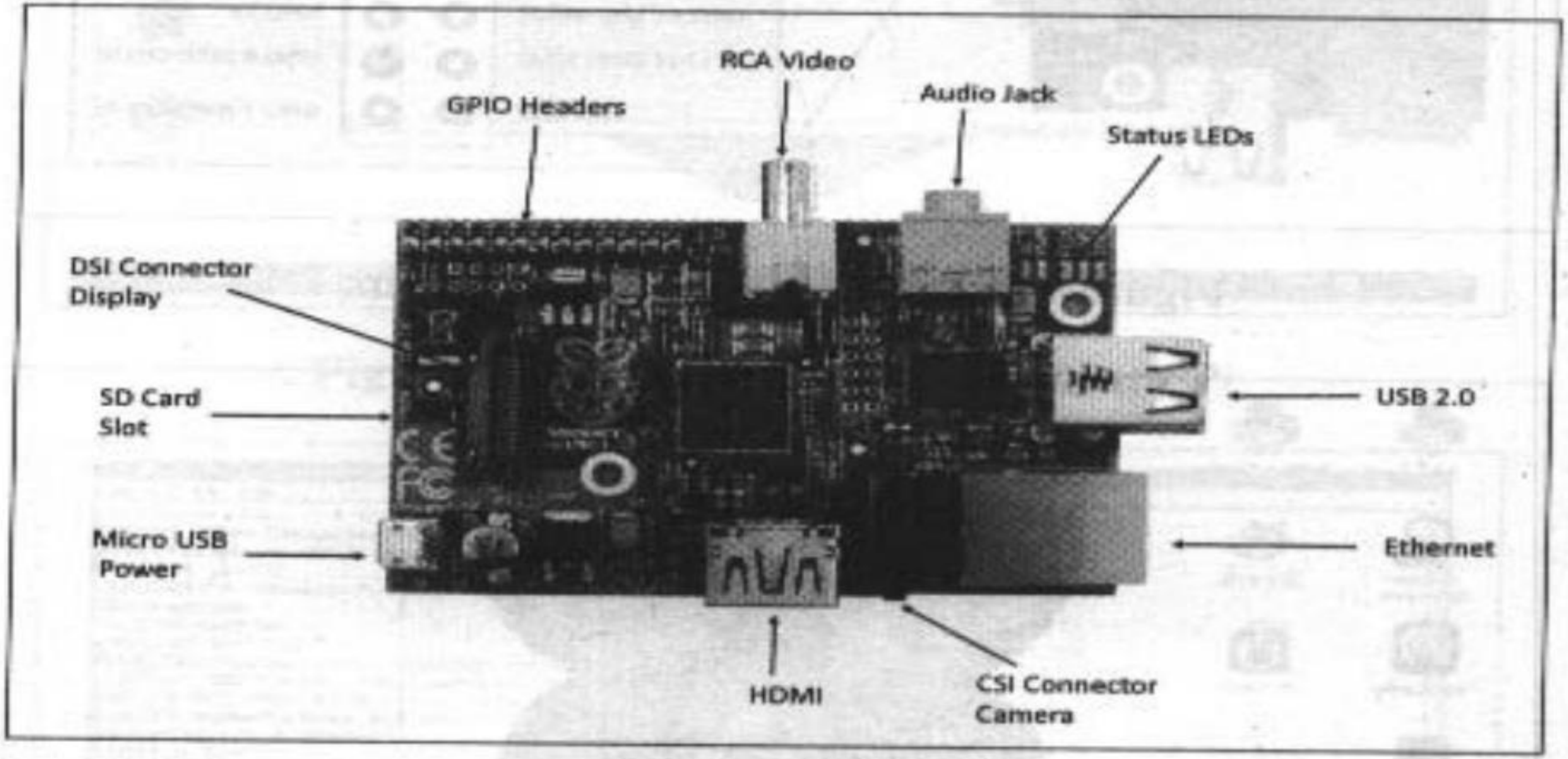


Figure 7.1: Block diagram of an IoT Device



Raspberry Pi board

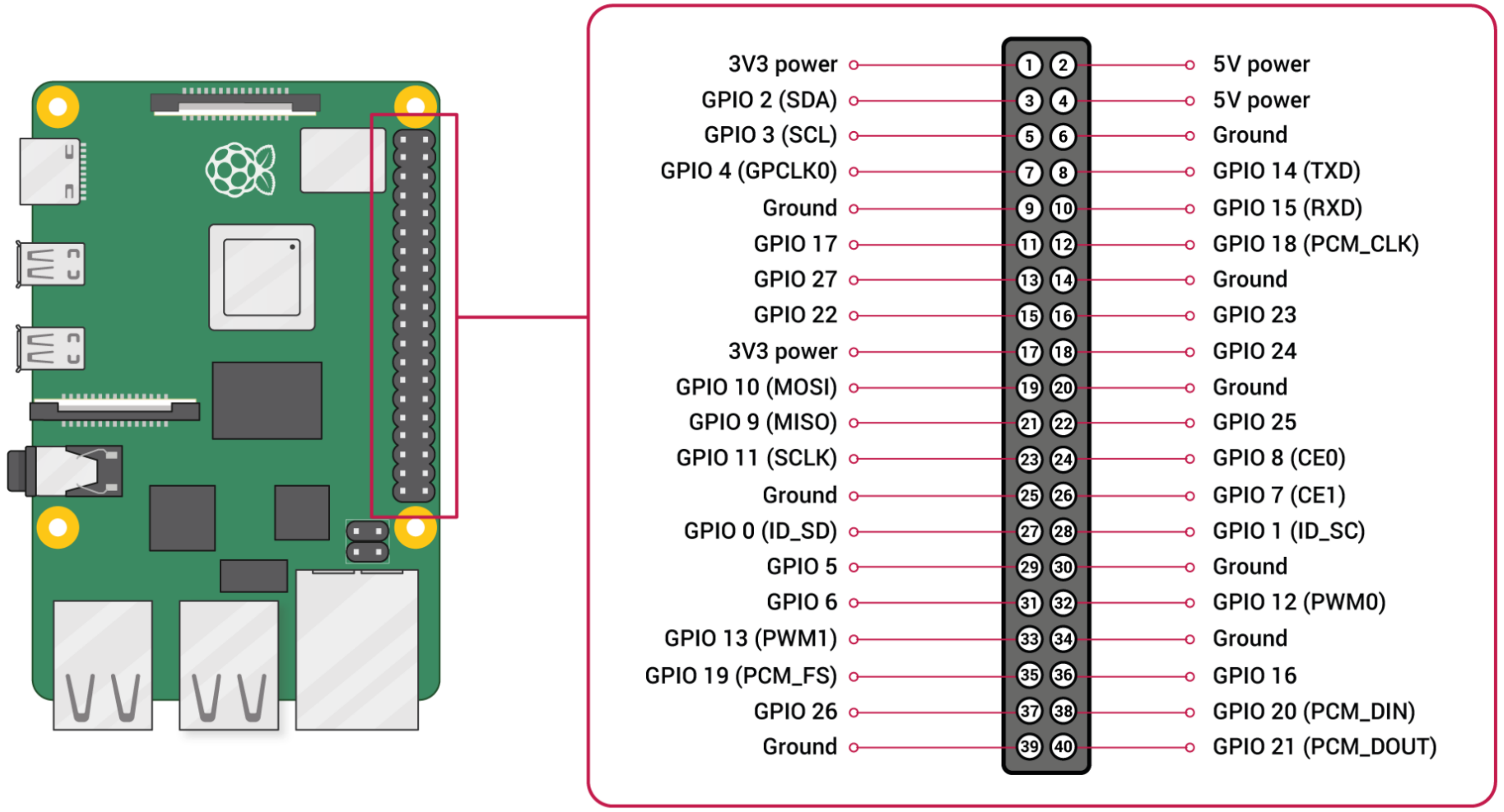
Common Raspberry Pi Models

1. Raspberry Pi 4 Model B: The latest and most powerful model, available with 1GB, 2GB, 4GB, or 8GB of RAM.
2. Raspberry Pi 3 Model B+: Popular for its balance of performance and cost, includes 1GB RAM, and both Wi-Fi and Bluetooth connectivity.
3. Raspberry Pi Zero and Zero W: Smaller, more affordable versions with lower performance, suitable for simpler projects. The Zero W includes Wi-Fi and Bluetooth.

Main components and features typically found on a Raspberry Pi board

1. Processor (CPU): The heart of the Raspberry Pi, usually an ARM-based CPU. For instance, the Raspberry Pi 4 uses a quad-core ARM Cortex-A72 CPU.
2. Memory (RAM): Varies between models, with options typically ranging from 1GB to 8GB of LPDDR4 SDRAM on the Raspberry Pi 4.
3. Storage: No built-in storage, but it uses a microSD card slot for the operating system and file storage. Some models also support booting from USB drives.
4. USB Ports: Several USB ports for connecting peripherals such as keyboards, mice, and storage devices. The Raspberry Pi 4 includes USB 3.0 and USB 2.0 ports.
5. HDMI Ports: One or two micro HDMI ports for video output, supporting up to 4K resolution on newer models like the Raspberry Pi 4.

6. GPIO Pins: General Purpose Input/Output pins for connecting various sensors, LEDs, and other hardware projects. Typically, there are 40 GPIO pins.
7. Networking: Options for both wired (Ethernet) and wireless (Wi-Fi and Bluetooth) connectivity. The Raspberry Pi 4 has Gigabit Ethernet and dual-band 802.11ac Wi-Fi.
8. Power Supply: A micro USB or USB-C power port (depending on the model) for supplying power to the board. The Raspberry Pi 4 uses a USB-C power supply.
9. Audio Output: 3.5mm audio jack and HDMI audio output for connecting speakers or headphones.
10. Camera and Display Interfaces: Dedicated ports for connecting the Raspberry Pi Camera Module and the official Raspberry Pi display.
11. Operating System: Typically runs a version of Linux, with Raspberry Pi OS (formerly Raspbian) being the most common. It can also run other operating systems like Ubuntu, and even specialized distributions like RetroPie for gaming.


3V3 power 1 2 5V power
GPIO 2 (SDA) 3 4 5V power
GPIO 3 (SCL) 5 6 Ground
GPIO 4 (GPCLK0) 7 8 GPIO 14 (TXD)
Ground 9 10 GPIO 15 (RXD)
GPIO 17 11 12 GPIO 18 (PCM_CLK)
GPIO 27 13 14 Ground
GPIO 22 15 16 GPIO 23
3V3 power 17 18 GPIO 24
GPIO 10 (MOSI) 19 20 Ground
GPIO 9 (MISO) 21 22 GPIO 25
GPIO 11 (SCLK) 23 24 GPIO 8 (CE0)
Ground 25 26 GPIO 7 (CE1)
GPIO 0 (ID_SD) 27 28 GPIO 1 (ID_SC)
GPIO 5 29 30 Ground
GPIO 6 31 32 GPIO 12 (PWM0)
GPIO 13 (PWM1) 33 34 Ground
GPIO 19 (PCM_FS) 35 36 GPIO 16
GPIO 26 37 38 GPIO 20 (PCM_DIN)
Ground 39 40 GPIO 21 (PCM_DOUT)

Linux on Raspberry Pi

Raspberry Pi supports various flavors of Linux including:

- Raspbian: Raspbian Linux is a Debian Wheezy port optimized for Raspberry Pi. This is the recommended Linux for Raspberry Pi.
- Arch
- Pidora
- RaspBMC
- OpenELEC
- RISC OS

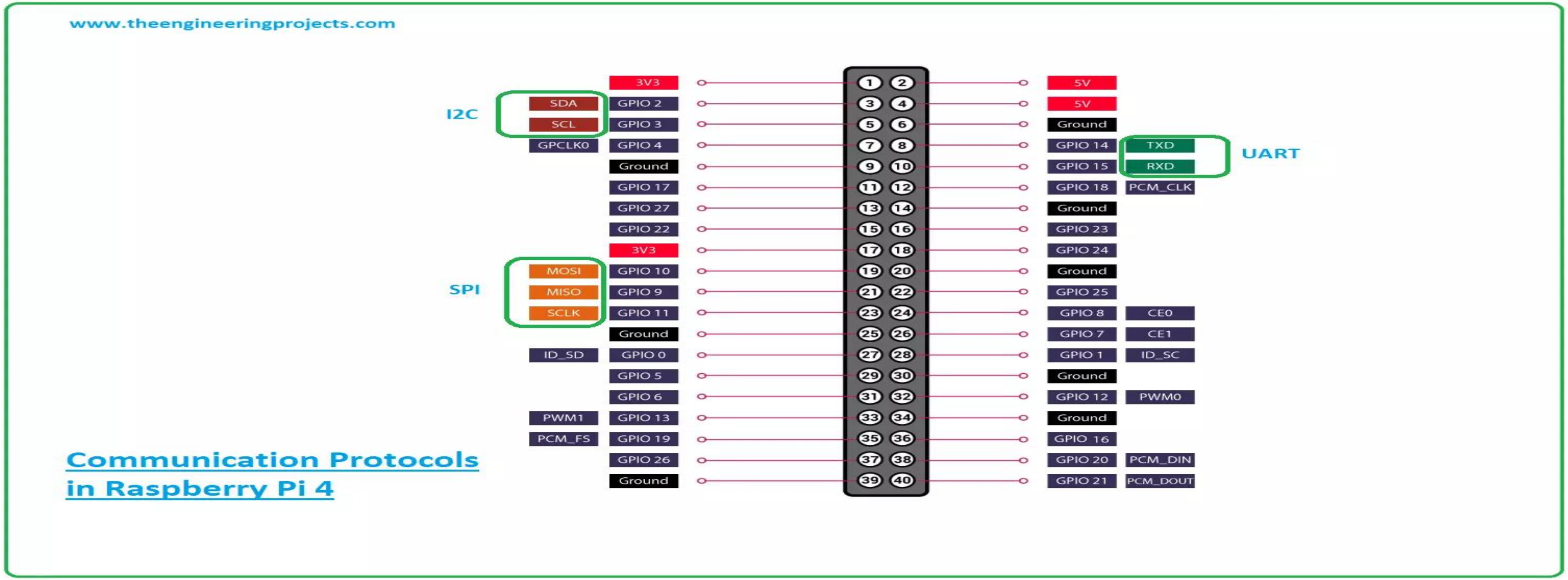
- To configure Raspberry Pi, the raspi-config tool is used which can be launched from command line as ($raspi-config)
- Using the configuration tool you can expand root partition to fill SD card, set keyboard layout, change password, set locale and timezone, change memory split, enable or disable SSH server and change boot behavior
- It is recommended to expand the root file system so that you can use the entire space on the SD card
- Though Raspberry Pi comes with an HDMI output, it is more convenient to access the device with a VNC connection or SSH
- This does away with the need for a separate display for Raspberry Pi and you can use Raspberry Pi from desktop or laptop computer

Command	Function	Example
cd	Change directory	cd /home/pi
cat	Show file contents	cat file.txt
ls	List files and folders	ls /home/pi
locate	Search for a file	locate file.txt
lsusb	List USB devices	lsusb
pwd	Print name of present working directory	pwd
mkdir	Make directory	mkdir /home/pi/new
mv	Move (rename) file	mv sourceFile.txt destinationFile.txt
rm	Remove file	rm file.txt
reboot	Reboot device	sudo reboot
shutdown	Shutdown device	sudo shutdown -h now
grep	Print lines matching a pattern	grep -r "pi" /home/
df	Report file system disk space usage	df -Th
ifconfig	Configure a network interface	ifconfig
netstat	Print network connections, routing tables, interface statistics	netstat -lntp
tar	Extract/create archive	tar -xzf foo.tar.gz
wget	Non-interactive network downloader	wget http://example.com/file.tar.gz

Table [illegible]. Raspberry Pi frequently used commands

Raspberry Pi Interfaces

- Raspberry Pi has serial, SPI and I2C interfaces for data transfer



Serial

- The serial interface on Raspberry Pi has receiver (Rx) and transmit (Tx) pins for communication with serial peripherals

Serial Peripheral Interface (SPI)

- Serial Peripheral Interface (SPI) is a synchronous serial data protocol used for communicating with one or more peripheral devices
- In an SPI connection, there is one master device and one or more peripheral devices
- There are five pins on Raspberry Pi for SPI interface:
    - MISO (Master In Slave Out): Master line for sending data to the peripherals
    - MOSI (Master Out Slave In): Slave line for sending data to the master
    - SCK (Serial Clock): Clock generated by master to synchronize data transmission
    - CE0 (Chip Enable 0): To enable or disable devices
    - CE1 (Chip Enable 1): To enable or disable devices

I2C

- The I2C interface pins on Raspberry Pi allow you to connect hardware modules, I2C interface allows synchronous data transfer with just two pins - SDA (data line) and SLC (clock line)

Programming Raspberry Pi with Python

- We can run any Python program that runs on a normal computer
- We can interface a wide variety of sensor and actuators with Raspberry Pi using the GPIO pins and the SPI, I2C and serial interfaces
- Input from the sensors connected to Raspberry Pi can be processed and various actions can be taken, like sending data to a server, sending an email, triggering a relay switch

Controlling LED with Raspberry Pi

- How to turn the LED ON/OFF from command line:
    - Lets connect the LED to GPIO PIN 18. Or, LED can also be connected to any other GPIO PIN
- Python program for blinking an LED connected to Raspberry every second
- The program uses the Raspberry Pi (RPi)
- GPIO Module to control the GPIO on Raspberry Pi
- Set Pin 18 direction to output and then write True/False alternatively after a delay of 1 second

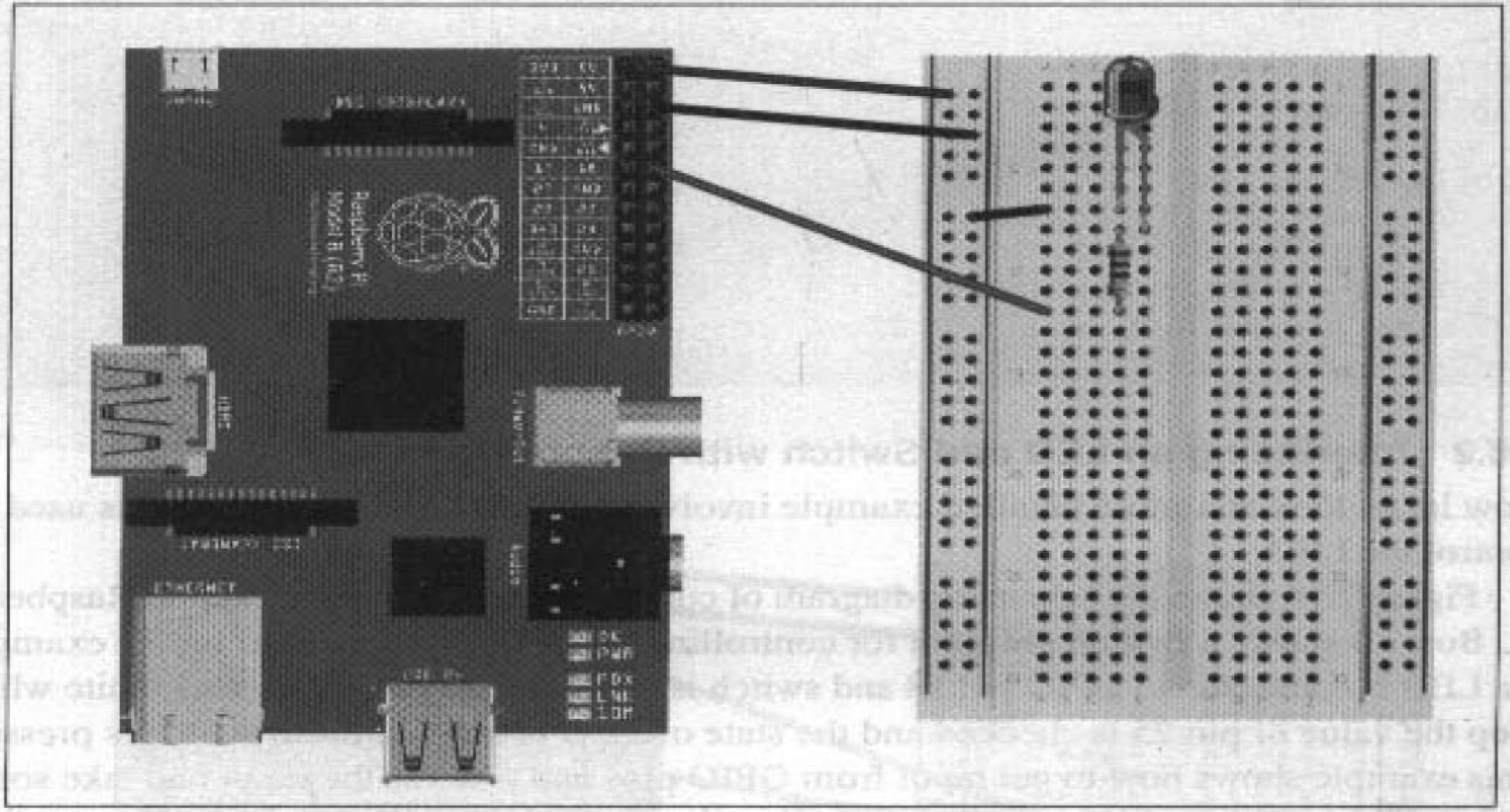


Figure 7.9: Controlling LED with Raspberry Pi

Switching LED ON/OFF from Raspberry Pi Console

```
$echo 18 > /sys/class/gpio/export

$cd /sys/class/gpio/gpio18

#Set pin 18 direction to out

$echo out > direction

#Turn LED on

$echo 1 > value

Turn LED off

$echo 0 > value
```

Python program for blinking LED

```
Import RPi.GPIO as GPIO

Import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(18,GPIO.OUT)

While True:

GPIO.output(18,True)

time.sleep(1)

GPIO.output(18,False)

time.sleep(1)
```

Interfacing a LED and SWITCH with Raspberry Pi

- The LED is connected to GPIO PIN 18 and SWITCH is connected to GPIO PIN 25
- In the infinite WHILE Loop the value of PIN 25 is checked and the state of LED is toggled if the switch is pressed

Python program for controlling a LED with a SWITCH

```
from time import sleep

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

#Switch Pin

GPIO.setup(25,GPIO.IN)

#LED Pin

GPIO.setup(18,GPIO.OUT)

state=false
```

```
def toggleLED(pin):
state=not state
GPIO.output(pin,state)
while True:
try:
if (GPIO.input(25)==True):
                toggleLED(pin)
sleep(.01)
except keyboardInterrupt:
exit()
```