A REAL TIME HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS USING RASPBERRY PI WITH IOT

An internship report submitted in partial fulfillment of the requirements for the Award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

BY

ADABALA SURESH 21JE1A0467

Under the Guidance of

Mr.F.Mohammad Khasim M.Tech

Assistant professor, ECE Department

ESWAR COLLEGE OF ENGINEERING

ELECTRONICS AND COMMUNICATION ENGINEERING

(Approved by AICTE, Affiliated to JNTU, Kakinada)

NARASARAOPET-522601, ANDHRA PRADESH

ESWAR COLLEGE OF ENGINEERING

2023-2024

ESWAR COLLEGE OF ENGINEERING

(Approved by AICTE &Affiliated to JNTU, Kakinada)

NARASARAOPET-522601, ANDHRA PRADESH

CERTIFICATE

This is to certify that the internship titled "A REAL TIME HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS USING RASPBERRY PI WITH IOT" is Submitted by ADABALA SURESH (21JE1A0467) Students of **B. Tech (ECE),** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **ELECTRONICS AND COMMUNICATION ENGINEERING**.

Internal Guide

Mr.F.Mohammed Khasim M.Tech

Assistant Professor

Head of the Department

Dr.SK.BASHEERA M.Tech, Ph.D.

Associate Professor & HOD

Submitted for Viva voice Examination held on _____

PROGRAM BOOK FOR
SHORT-TERM INTERNSHIP

(On site/Virtual)

Name of the Student: ADABALA SURESH

Name of the College: ESWAR COLLEGE OF ENGINEERING

Roll Number: 21JE1A0467

Period of Internship: 8 WEEKS

Name of the Internship: A REAL TIME HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS USING RASPBERRY PI WITH IOT

Name of the Intern Organization: SAK INFORMATICS

JNTUK, KAKINADA

2024

Student's Declaration

I Adabala Suresh a student of B Tech Program, Reg. No. 21JE1A0467 of the Department of ECE, Eswar College of Engineering, Narasaraopet do hereby declare that I have completed the mandatory internship from 03-JUNE-2024 to 27-JULY-2024 in Eswar College of Engineering organization under the Faculty Guide ship of Mr.F.Mohammed Khasim M.Tech Department of ECE, Eswar College of Engineering.

(Signature and Date)

Official Certification

This is to certify that Adabala Suresh Reg. No. 21JE1A0467 has completed his/her Internship in SAK INFORMATICS on A REAL TIME HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS USING RASPBERRY PI WITH IOT under my supervision as a part of partial fulfillment of the requirement for the Degree of B.Tech in ECE, Eswar College of Engineering.

This is accepted for evaluation.

Endorsements

Faculty Guide

Head of the Department

Principal

SAK INFORMATICS
Student@knowledge

CERTIFICATE OF INTERNSHIP

This is to certify that Mr./Miss. **SURESH ADABALA(21JE1A0467)** of B.Tech(ECE), IIIrd Year 2nd Semester student of **ESWAR COLLEGE OF ENGINEERING**, **NARASARAOPET**, Andhra Pradesh has completed his/her internship program on **"REAL TIME AUTOMATION USING RASPBERRY PI WITH IOT"** as a platform in **SAK INFORMATICS** for a period of **03-JUNE-2024 to 27- JULY- 2024.**

Our company internship program includes training and all round development of the candidate which he had the privilege to experience. We at **SAK INFORMATICS** wish him the best and success in his/her life and career.

Warm Regards,

Corporate Office: Plot No. 544, #102, Pragathi Nagar,Hyderabad-90
Contact:(0)-9603999243&(0)-9000188676,E-Mail:ceo@sakinformatics.com,Web:www.sakinformatics.com

ACKNOWLEDGEMENT

I indebted to the Management for providing me an opportunity with excellent academic, infrastructural Lab facilities to carry out my U.G program successfully.

I would like to express my deepest sense of gratitude towards my Internal Guide, **Mr.F.Mohammed Khasim** M.Tech Assistant Professor in Department of **ECE** for him encouragement and valuable guidance in bringing shape to this dissertation.

I grateful to **Dr.SHAIK BASHEERA,** M.Tech,Ph.D **HOD** Department of Electronics & Communication Engineering, ESWAR COLLEGE OF ENGINEERING, Narasaraopeta affiliated JNTUK Kakinada Who have given for his encouragement and motivation.

I express my deep gratitude and regards to our beloved principal **Dr.G.NAGA MALLESWARA RAO**, M.Tech,Ph.D who have given support for the Internship or in other aspects of my studies at ESWAR COLLEGE OF ENGINEERING, Kesanupalli, Narasaraopet, and affiliated JNTU Kakinada.

I thankful to all the Faculty Members in the department for their teachings and academic support and thanks to Technical Staff and Non-teaching staff in the department for their support.

At last, but not least I thank all my well-wishers for rendering necessary support during the execution of this work.

Adabala Suresh

21JE1A0467

DECLARATION

I hereby declare that the Project report titled "**A REAL TIME HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS USING RASPBERRY PI WITH IOT**" is bona fide work done by me, under the guidance, **Mr.F.Mohammed Khasim** M.Tech Assistant Professor, **ESWAR College of Engineering**. This report is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology in ECE.

Place: Kesanuaplli

Date:

Adabala Suresh

21JE1A0467

CONTENTS

CHAPTER 1

EMBEDDED SYSTEM

1.1 Embedded System

A schema is a strategy for operational, dealing with or doing one or various endeavours according to a settled orchestrate, package, or set of rules. A skeleton is moreover preparation its chip in as shown by the game plan or venture. For example, consider a watch. It is a period show schema. Its parts are its supplies, needles and battery with the radiant dial, body and strap. These parts make to show the nonstop reliably and continually overhaul the time reliably. The schema framework overhauls the presentation using three needles after consistently. It takes after a set of principles.

An inserted framework may be a framework that goes to attempt and do a unimportant assignment is that the connected outline and this level delineated as combo of every machine code and fittings. Generally handy importance of monitor or help the procedure of mechanical assembly, hardware or plant. "Installed" the very truth that partner degree inverse compelling a broadly functional smart phone could likewise be wont to administration the operation of an oversized muddled procedure plant, and its vicinity is self-evident.

All embedded systems square measure as well as computers or microprocessors. A number of these laptops square measure but terribly easy classifications related through private processer. The unpleasantly square measure expressions singularly one performs or set of capacities to fulfill one arranged reason. In extra confounded frameworks partner degree machine developer that permits the installed structure to be used for a picked reason in the midst of a specific application chooses the working of the embedded structure. This is versatility to claim projects imply a near embedded schema is used for an extent of diverse functions. Now and again a microchip could likewise be composed in such somehow that application machine code for a chose reason for existing is quality added to the fundamental machine code amid a second technique, when that its unendurable to structure extra changes.

The most run contraption contains one microchip (habitually intimated as a "chip"), which itself can be prepackaged with unique chips in the midst of a crossbreed structure or application particular microcircuit. Its data starts from a locator or contraption and its yield goes to a switch

or substance that could begin or stop the operation of a machine or, by operational a valve, could affiliation the stream of fuel to embellishment degree engine.

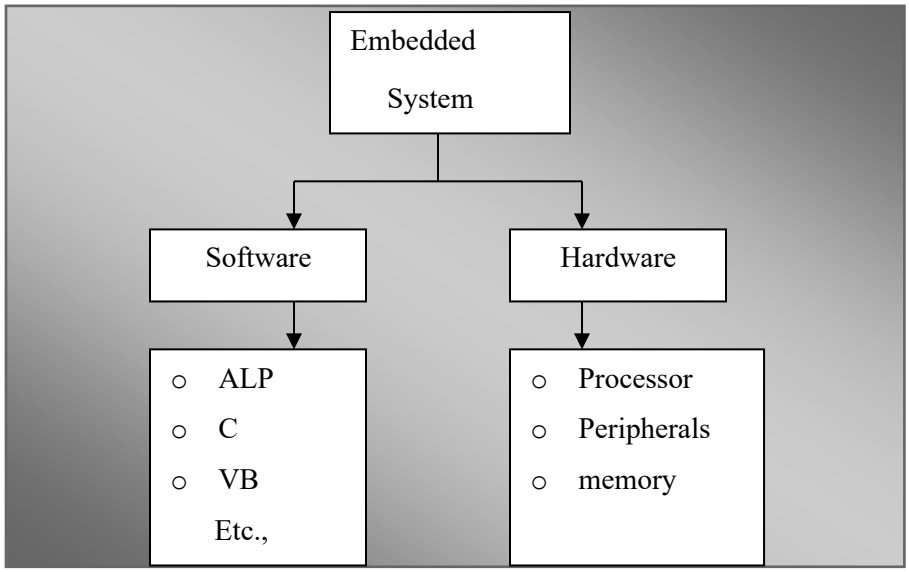


Fig.1.1.1: Block diagram of Embedded System

1.2 Classification of Embedded Systems

1.2.1 Small Scale Embedded Systems

The skeletons are delineated a single 8- or 16-bit microcontroller. The programming difficulties and incorporate sheet level design. Exactly when making embedded programming for these, an editor, developing executor and cross building operator, specific used, are the rule programming contraptions. By and large, "C" is used for making structures. "C" platform game plan is done into the social affair, also executable codes are then fittingly situated in the pattern memory. The thing needs to fit inside the memory open and keep in view the need to most inaccessible point power dispersal when framework is running on and on.

1.2.2 Medium Scale Embedded Systems

These skeletons are customarily made with a solitary or few 16- or 32-bit micro controllers or Dips or Reduced Instruction Set Computers (Risks). These have both apparatus and programming complexities. For complex programming chart, there are the running with programming mechanical gatherings Source code planner instrument, Simulator, debugger and incorporated improvement environment. Programming mechanical assemblies in like manner give

the responses for the gear complexities. A developing executor is of little use as a programming instrument. These skeletons may furthermore use the instantly available Asps and Is (elucidated later) for the diverse limits case in point, for the vehicle interfacing, scrambling, translating, discrete cosine change and regressively transformation, TCP/IP tradition stacking and framework joining limits.

1.2.3. Sophisticated Embedded System

Forefront presented skeletons have colossal fittings and programming complexities and may require adaptable processors or configurable processors and programmable defense bunches. They are utilized for front line applications that need fittings and programming co-design and trade off in continue going skeleton; on the other hand, they are obliged by the arranging paces open in their supplies units. Certain thing points of confinement, case in point, encryption and unraveling reckonings, discrete cosine change and inverse change estimations, TCP/IP custom stacking and system driver breaking points are executed in the fittings to get extra speeds by sparing time. A package of the points of confinement of the apparatus assets in the structure are besides acknowledged by the thing. Change mechanical assemblies for these schemas may not be instantly available at a sensible cost or may not be open at all.

1.3 Memory Architecture

There two diverse sort's memory architectures there are:

- Harvard Architecture
- Von-Neumann Architecture

1.3.1 Harvard Architecture

Workstations have separate memory regions for project guidelines and information. There are two or more inside information transports, which permit concurrent access to both directions and information. The CPU gets program directions on the system memory transpose.

The Harvard building design is workstation construction modeling through actually separate stockpiling indicator guidelines from the information. This started hand-off created workstation, put away directions information restricted information stockpiling, completely held inside the focal preparing unit, and gave no right to gain entrance to the direction stockpiling as information.

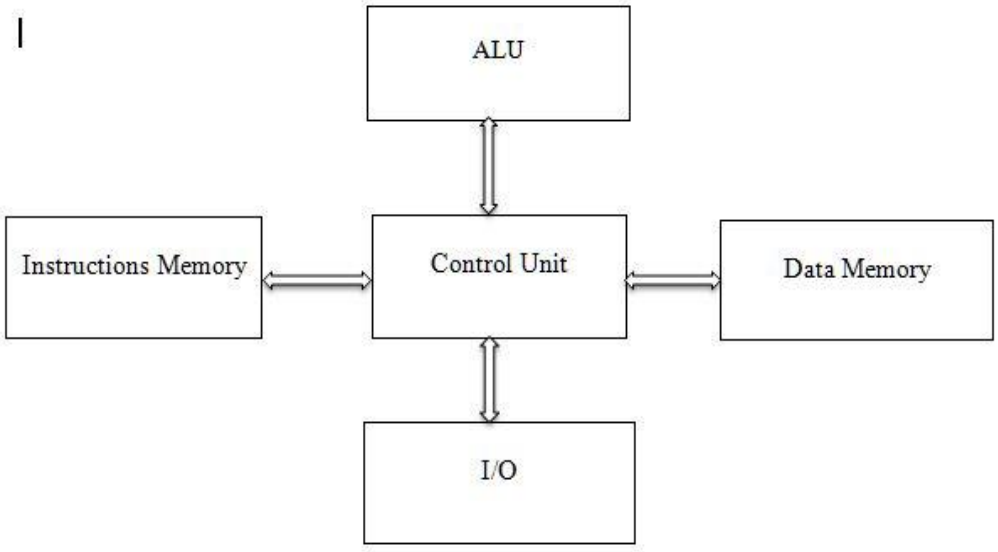


Fig.1.3.1: Harvard Architecture

Projects required to be stacked by an administrator, the processor couldn't boot it's the important playing point of the immaculate Harvard structural planning - synchronous framework decreased by adjusted utilizing current central processing unit store frameworks. Moderately immaculate Harvard building design machines are utilized generally within applications where tradeoffs, for example, the expense and force funds from excluding reserves, exceed punishments different code and information location places.

1.3.2 Von-Neumann Architecture

A machine has a solitary, basic memory space in which both project guidelines and information are put away. There is a solitary inside information transport that gets both directions and information. They can't be performed in the meantime

The Von Neumann construction modeling is an outline model for a put away program computerized workstation that is use a focal preparing unit control processing unit and a solitary separate stockpiling structure ("memory") to clasp both guidelines and information. This is machine researcher machines actualize an all-inclusive Turing machine and have a consecutive structural engineering. The expressions "von Neumann structural engineering" and "put away program workstation" are for the most part utilized conversely, and that utilization is followed in this article.

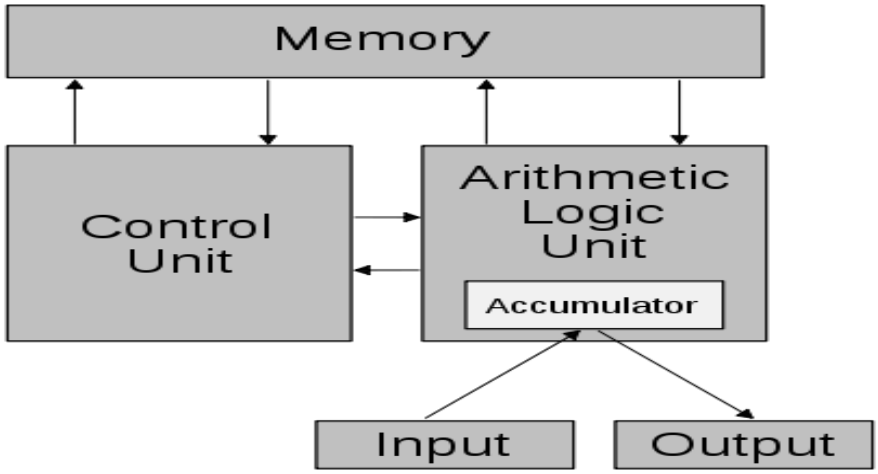


Fig.1.3.2: Von-Neumann Architecture

The systems for exchanging the information and directions between the CPU and memory are, be that as it may, extensively more unpredictable than the first von Neumann structural planning.

Essential Difference in the middle of Harvard and Von-Neumann Architecture:

- In a workstation utilizing the Von-Neumann structural planning without store; the focal handling unit (CPU) can either be perusing and direction or composing/perusing information to/from the memory. Both of these operations can't happen all the while as the information and guidelines utilize the same framework transport.
- In a workstation utilizing the Harvard structural planning the CPU can both read a direction and access information memory in the meantime without store. This implies that a workstation with Harvard structural engineering can possibly be quicker for a given circuit multifaceted nature on the grounds that information access and guideline brings don't fight for utilization of a solitary memory pathway.
- Harvard architectures are normally just utilized as a part of either particular frameworks or for certain employments. It is utilized within particular computerized indicator preparing (DSP), normally for feature and sound handling items. It is likewise utilized within a lot of people little microcontrollers utilized as a part of hardware applications, for example, Advanced RISK Machine.

CHAPTER 2

RASPBERRY PI 3

The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processer, 10x faster than the first generation Raspberry Pi. Additionally it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs.

FEATURES

1. Processor Broadcom BCM2387 chipset.
2. 1.2GHz Quad-Core ARM Cortex-A53
3. 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)
4. GPU Dual Core Video Core IV® Multimedia Co-Processor. Provides Open GL
5. ES 2.0, hardware-accelerated Open VG, and 1080p30 H.264 high-profiled code.
6. Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and
7. DMA infrastructure
8. Memory 1GB LPDDR2
9. Operating System Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IOT
10. Dimensions 85 x 56 x 17mm
11. Power Micro USB socket 5V1, 2.5A

CONNECTORS

1. Ethernet 10/100 Base T Ethernet socket
2. Video Output HDMI (rev 1.3 & 1.4Composite RCA (PAL and NTSC)
3. Audio Output Audio Output 3.5mm jack, HDMI
4. USB 4 x USB 2.0 Connector
5. GPIO Connector 40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines.

6. Camera Connector 15-pin MIPI Camera Serial Interface (CSI-2)
7. Display Connector Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
8. Memory Card Slot Push/pull Micro SDIO

KEY BENEFITS

1. Low cost
2. Consistent board format
3. 10x faster processing
4. Added connectivity

KEY APPLICATIONS

1. Low cost PC/tablet/laptop
2. IOT applications
3. Media centre
4. Robotics
5. Industrial/Home automation
6. Server/cloud server
7. Print server
8. Security monitoring
9. Web camera
10. Gaming
11. Wireless access point
12. Environmental sensing/monitoring (e.g. weather station)

PIN OUTS OF RASPBERRY PI 3

Raspberry Pi3 model B consists of 40 Pins as shown in figure 3.3. Out of which there are two 5v pins and one 3.3v supply pins which are shown with red colour in the figure. The black colored pins indicate ground pins.

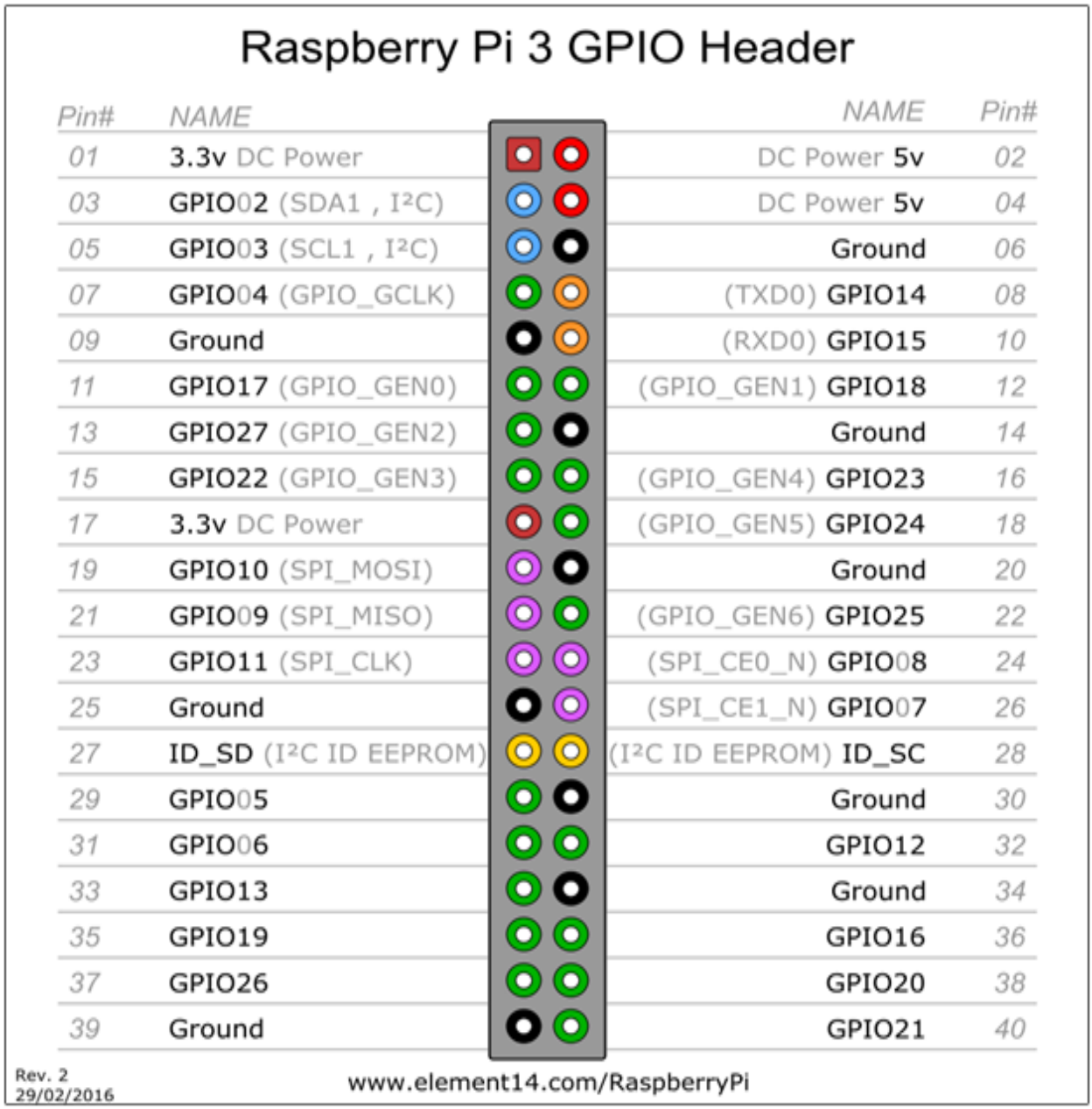
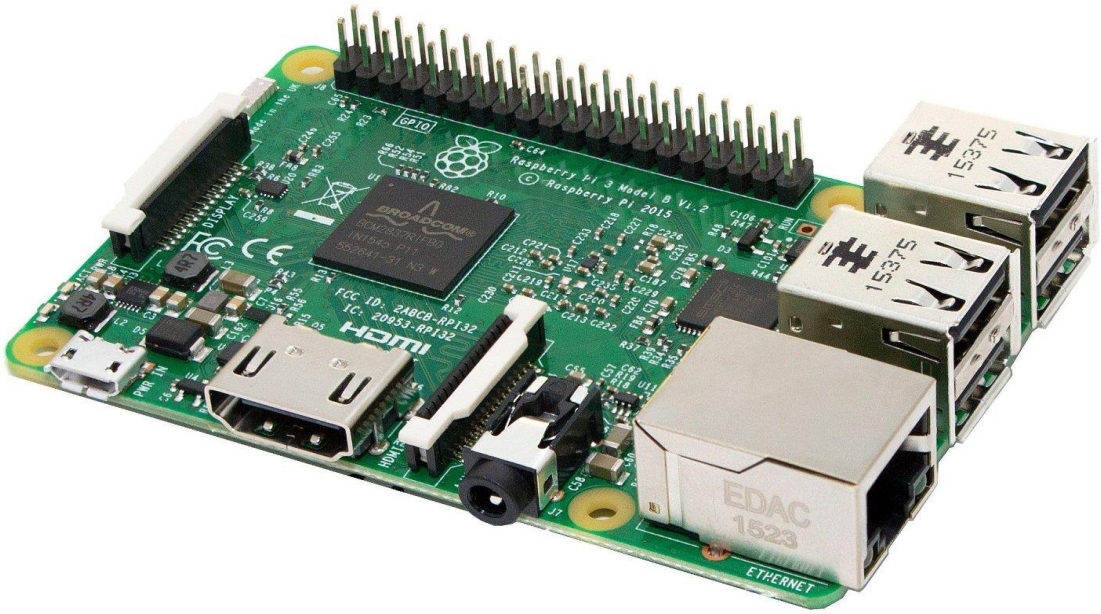


Fig 2.1: Pin diagram of Raspberry Pi 3

There are also some special purpose pins like SPI, I2c, TXD, and RXD apart from general purpose input output pins.



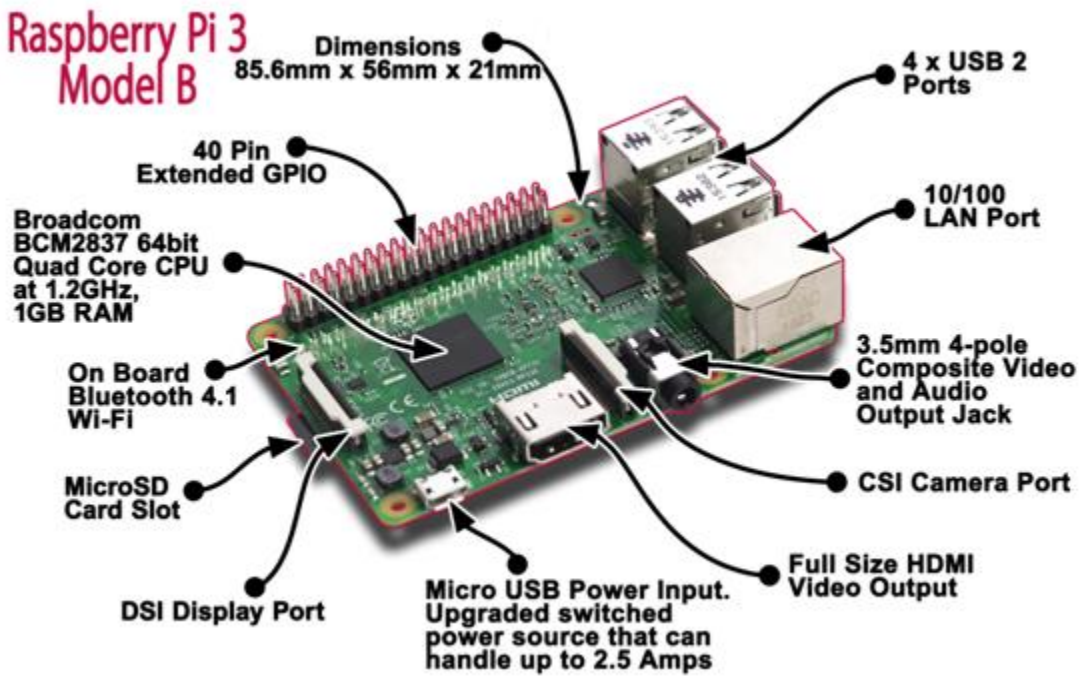


Fig 2.2: Raspberry Pi 3 Model B

RAM

On the older beta Model B boards, 128 MB was allocated by default to the GPU, leaving 128 MB for the CPU. On the first 256 MB release Model B (and Model A), three different splits were possible. The default split was 192 MB (RAM for CPU), which should be sufficient for standalone 1080p video decoding, or for simple 3D, but probably not for both together. 224 MB was for Linux only, with only a 1080p frame buffer, and was likely to fail for any video or 3D. 128 MB was for heavy 3D, possibly also with video decoding (e.g. XBMC). Comparatively the Nokia 701 uses 128 MB for the Broadcom Video Core IV. For the new Model B with 512 MB RAM initially there were new standard memory split files released(arm256_start.elf, arm384_start.elf, arm496_start.elf) for 256 MB, 384 MB and 496 MB CPU RAM (and 256 MB, 128 MB and 16 MB video RAM). But a week or so later the RPF released a new version of start.elf that could read a new entry in config.txt (GPU _ MEM=xx) and could dynamically assign an amount of RAM (from 16 to 256 MB in 8 MB steps) to the GPU, so the older method of memory splits became obsolete, and a single start.elf worked the same for 256 and 512 MB Raspberry Pies. The Raspberry Pi 2 and the Raspberry Pi 3 have 1 GB of RAM. The Raspberry Pi Zero and Zero W have 512 MB of RAM.

PERIPHERALS

The current Model B boards incorporate four USB ports for connecting peripherals. The Raspberry Pi may be operated with any generic USB computer keyboard and mouse. It may also be used with USB storage, USB to MIDI converters, and virtually any other device/component with USB capabilities. Other peripherals can be attached through the various pins and connectors on the surface of the Raspberry Pi.

Video

The video controller can emit standard modern TV resolutions, such as HD and Full HD, and higher or lower monitor resolutions and older standard CRT TV resolutions. As shipped (i.e., without custom over clocking) it can emit these: 640×350 EGA; 640×480 VGA; 800×600 SVGA; 1024×768 XGA; 1280×720 720p HDTV; 1280×768 WXGA variant; 1280×800 WXGA variant; 1280×1024 SXGA; 1366×768 WXGA variant; 1400×1050 SXGA+; 1600×1200 UXGA; 1680×1050 WXGA+; 1920×1080 1080p HDTV; 1920×1200 WUXGA. Higher resolutions, such as, up to 2048×1152, may work or even 3840×2160 at 15 Hz (too low a frame rate for convincing video). Note also that allowing the highest resolutions does not imply that the GPU can decode

video formats at those; in fact, the Pies are known to not work reliably for H.264 (at those high resolutions), commonly used for very high resolutions (most formats, commonly used, up to full HD, do work).

Real-time clock

None of the current Raspberry Pi models have a built-in real-time clock, and so they don't "know" the time of day. As a workaround, a program running on the Pi can get the time from a network time server or user input at boot time, thus knowing the time while powered on. To provide consistency of time for the file system, the PI does automatically save the time it has on shutdown, and re-installs that time at boot. A real-time hardware clock with battery backup, such as the DS1307, which is fully binary coded

The Raspberry Pi 3 is the third generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

The Raspberry Pi 3 has an identical form factor to the previous Pi 2 (and Pi 1 Model B+) and has complete compatibility with Raspberry Pi 1 and 2.

Monitor or TV

A monitor or TV with HDMI in can be used as a display with a Raspberry Pi.

HDMI to HDMI Cable

Connect Raspberry Pi to a Monitor or TV with a HDMI to HDMI cable.

USB Keyboard

A USB keyboard is used to input text into a computer, laptop or a Raspberry Pi.

USB Mouse

A USB mouse is used to control a cursor or mouse pointer when using a computer, laptop or a Raspberry Pi.

SD Card

SD Card and micro SD card are used to store files and software to use on your Raspberry Pi.

HOW TO INSTALL OS

Plug your power supply into the port marked "POWER IN", "PWR IN", or "PWR". Some Raspberry Pi models, such as the Zero series, have output USB ports with the same form factor as the power port. Be sure to use the correct port on your Raspberry Pi!

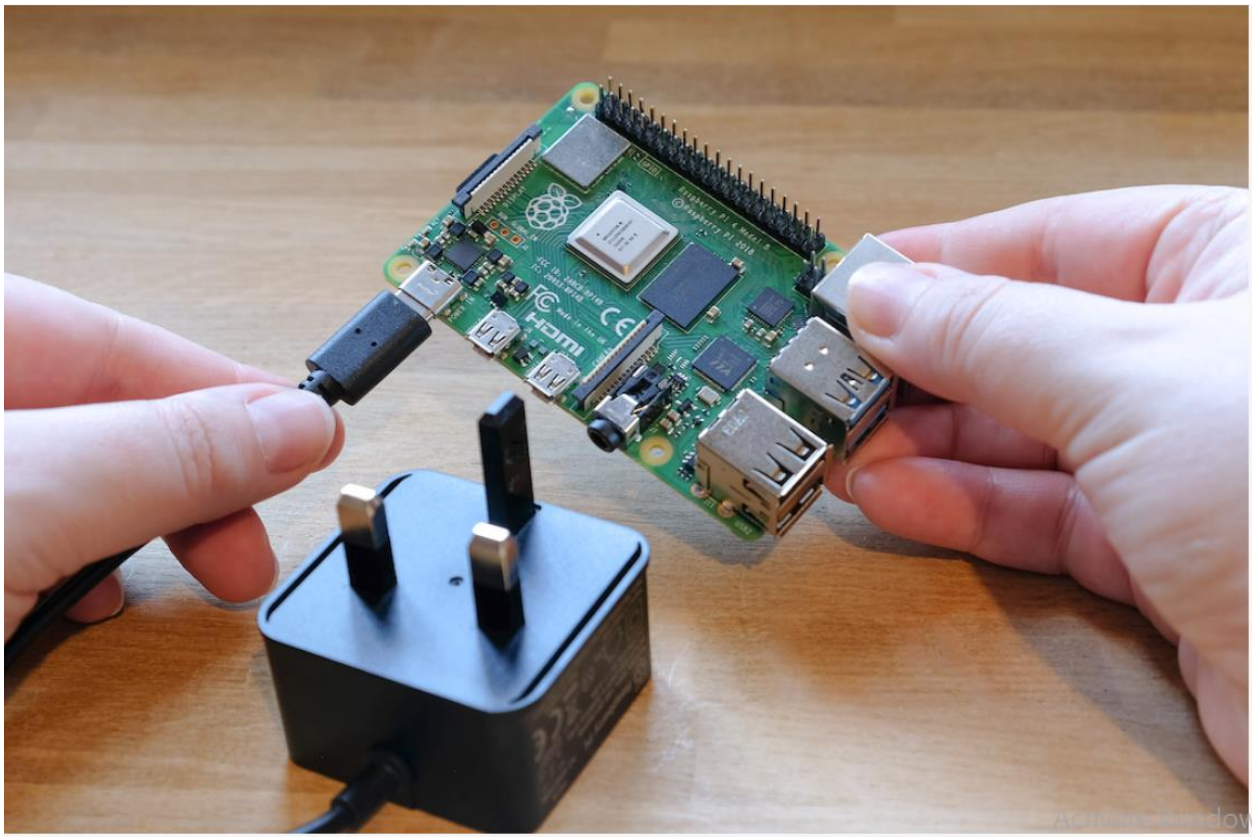


Fig 2.3: Connect to power supply pin to Raspberry pi 3 Model B

Boot Media

Raspberry Pi models lack onboard storage, so you have to supply it. You can boot your Raspberry Pi from an operating system image installed on any supported media: micro SD cards are used commonly, but USB storage, network storage, and storage connected via a PCIe HAT are also available. However, only recent Raspberry Pi models support all of these media types.

All Raspberry Pi consumer models since the Raspberry Pi 3 Model A+ feature a micro SD slot. Your Raspberry Pi automatically boots from the micro SD slot when the slot contains a card.

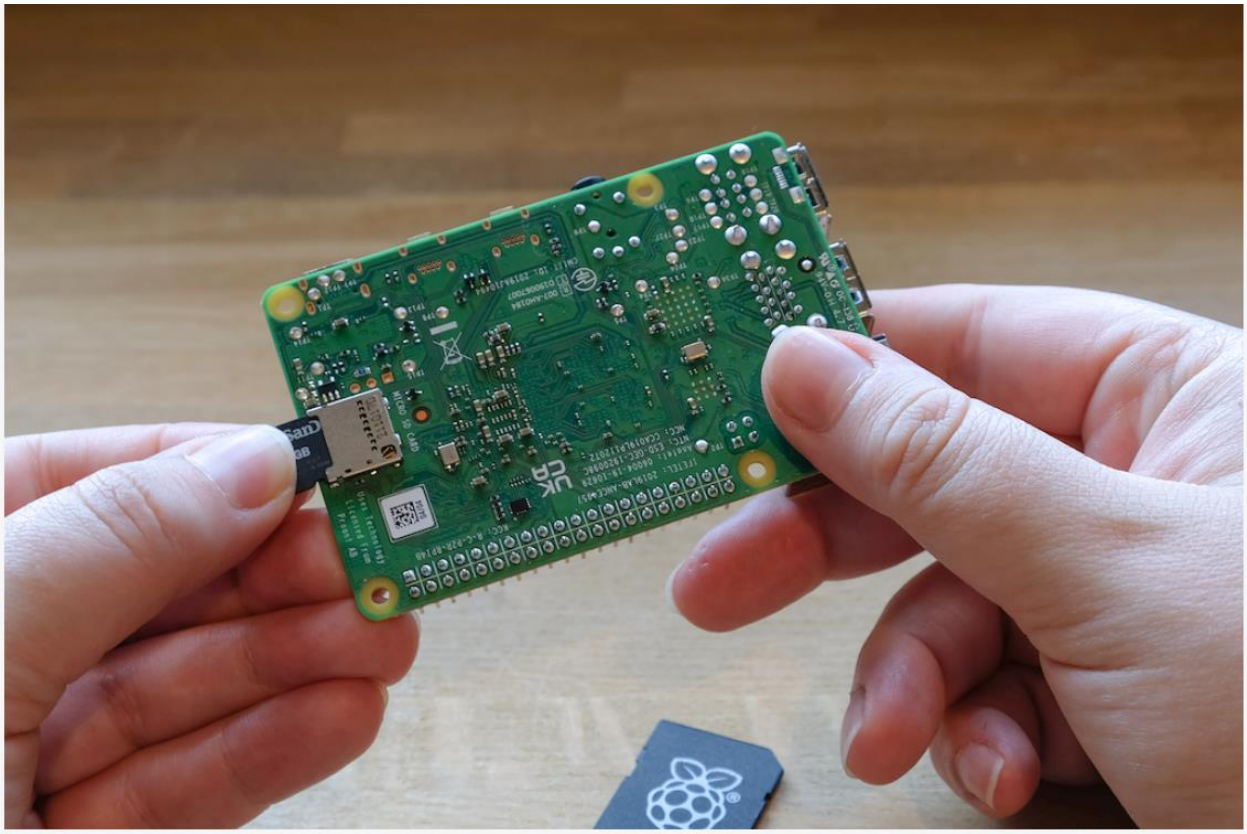


Fig 2.4: Insert the Memory Card

Keyboard

You can use any of the USB ports on your Raspberry Pi to connect a wired keyboard or USB Bluetooth receiver.

Fig 2.5: Connecting the Keyboard

Mouse

You can use any of the USB ports on your Raspberry Pi to connect a wired mouse or USB Bluetooth receiver.

Display

Raspberry Pi models have the following display connectivity:

Fig 2.6: Connecting the Mouse

Model	Display outputs
Raspberry Pi 5	2× micro HDMI
Raspberry Pi 4 (all models)	2× micro HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 3 (all models)	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 2 (all models)	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 1 Model B+	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi 1 Model A+	HDMI, audio and composite out via 3.5mm TRRS jack
Raspberry Pi Zero (all models)	mini HDMI

Most displays don't have micro or mini HDMI ports. However, you can use a micro-HDMI-to-HDMI cable or mini-HDMI-to-HDMI cable to connect those ports on your Raspberry Pi to any HDMI display. For displays that don't support HDMI, consider an adapter that translates display output from HDMI to a port supported by your display.

Install an operating system

To use your Raspberry Pi, you'll need an operating system. By default, Raspberry Pis check for an operating system on any SD card inserted in the SD card slot.

Depending on your Raspberry Pi model, you can also boot an operating system from other storage devices, including USB drives, storage connected via a HAT, and network storage.

To install an operating system on a storage device for your Raspberry Pi, you'll need:

- a computer you can use to image the storage device into a boot device
- a way to plug your storage device into that computer

Most Raspberry Pi users choose micro SD cards as their boot device. We recommend installing an operating system using Raspberry Pi Imager.

Raspberry Pi Imager is a tool that helps you download and write images on macOS, Windows, and Linux. Imager includes many popular operating system images for Raspberry Pi. Imager also supports loading images downloaded directly from Raspberry Pi or third-party vendors such as

Ubuntu. You can use Imager to preconfigure credentials and remote access settings for your Raspberry Pi.

Imager supports images packaged in the .img format as well as container formats like .zip.

If you have no other computer to write an image to a boot device, you may be able to install an operating system

OS customization

The OS customization menu lets you set up your Raspberry Pi before first boot. You can pre configure:

- a username and password
- Wi-Fi credentials
- the device hostname
- the time zone
- your keyboard layout
- remote connectivity

When you first open the OS customization menu, you might see a prompt asking for permission to load Wi-Fi credentials from your host computer. If you respond "yes", Imager will prefill Wi-Fi credentials from the network you're currently connected to. If you respond "no", you can enter Wi-Fi credentials manually.

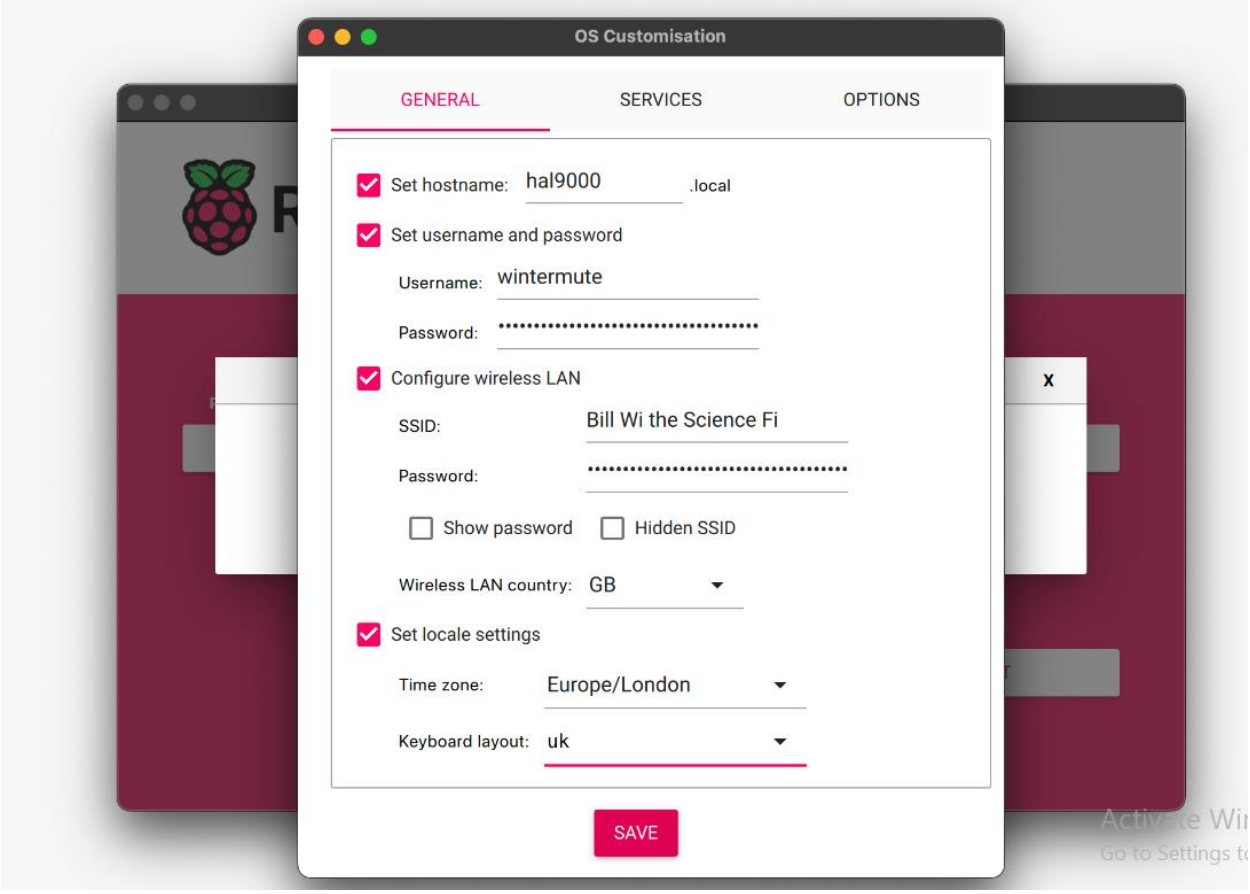
The hostname option defines the hostname your Raspberry Pi broadcasts to the network using mDNS. When you connect your Raspberry Pi to your network, other devices on the network can communicate with your computer using <hostname>.local or <hostname>.lan.

The username and password option defines the username and password of the admin user account on your Raspberry Pi.

The wireless LAN option allows you to enter an SSID (name) and password for your wireless network. If your network does not broadcast an SSID publicly, you should enable the "Hidden SSID" setting. By default, Imager uses the country you're currently in as the "Wireless LAN

country". This setting controls the Wi-Fi broadcast frequencies used by your Raspberry Pi. Enter credentials for the wireless LAN option if you plan to run a headless Raspberry Pi.

The locale settings option allows you to define the time zone and default keyboard layout for your Pi.



The Services tab includes settings to help you connect to your Raspberry Pi remotely.

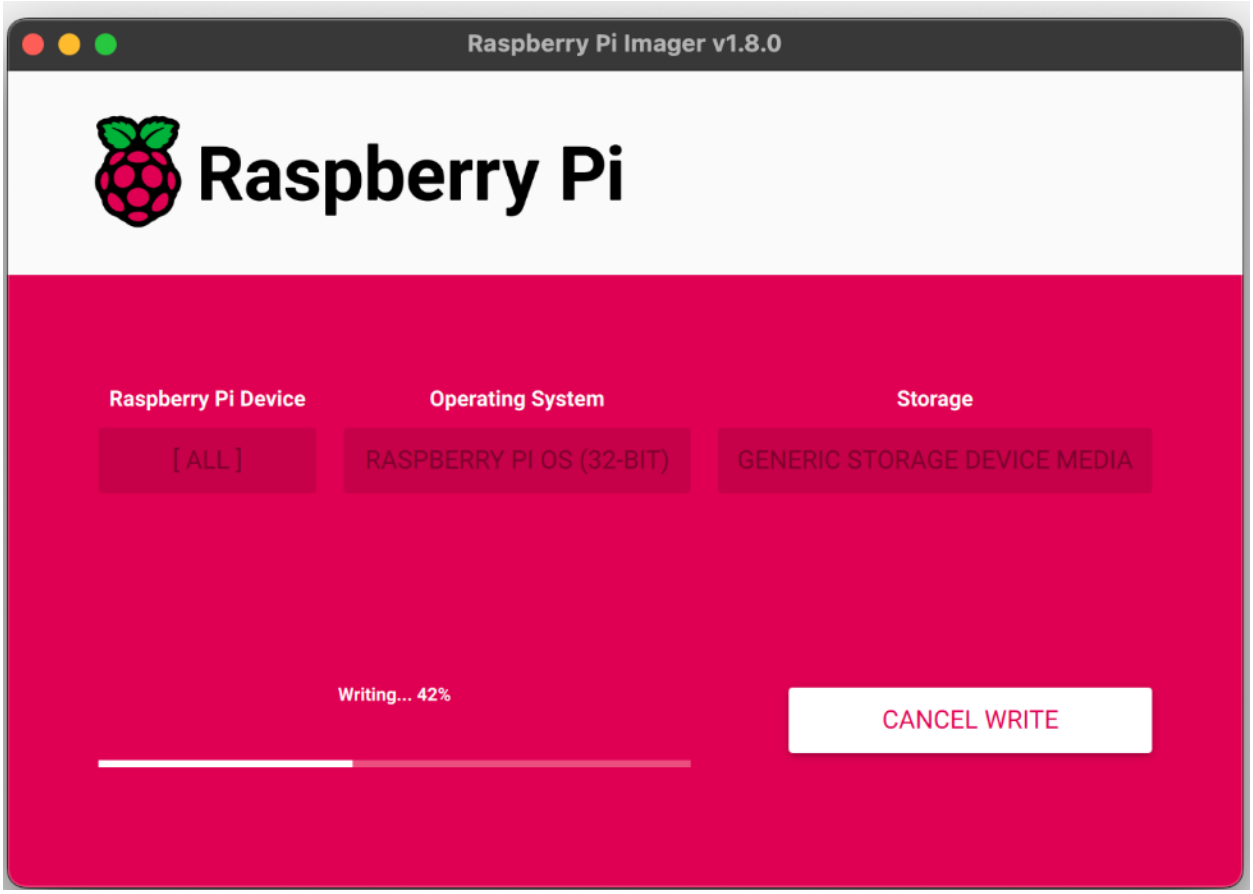
If you plan to use your Raspberry Pi remotely over your network, check the box next to Enable SSH. You should enable this option if you plan to run a headless Raspberry Pi.

•Choose the password authentication option to SSH into your Raspberry Pi over the network using the username and password you provided in the general tab of OS customization.

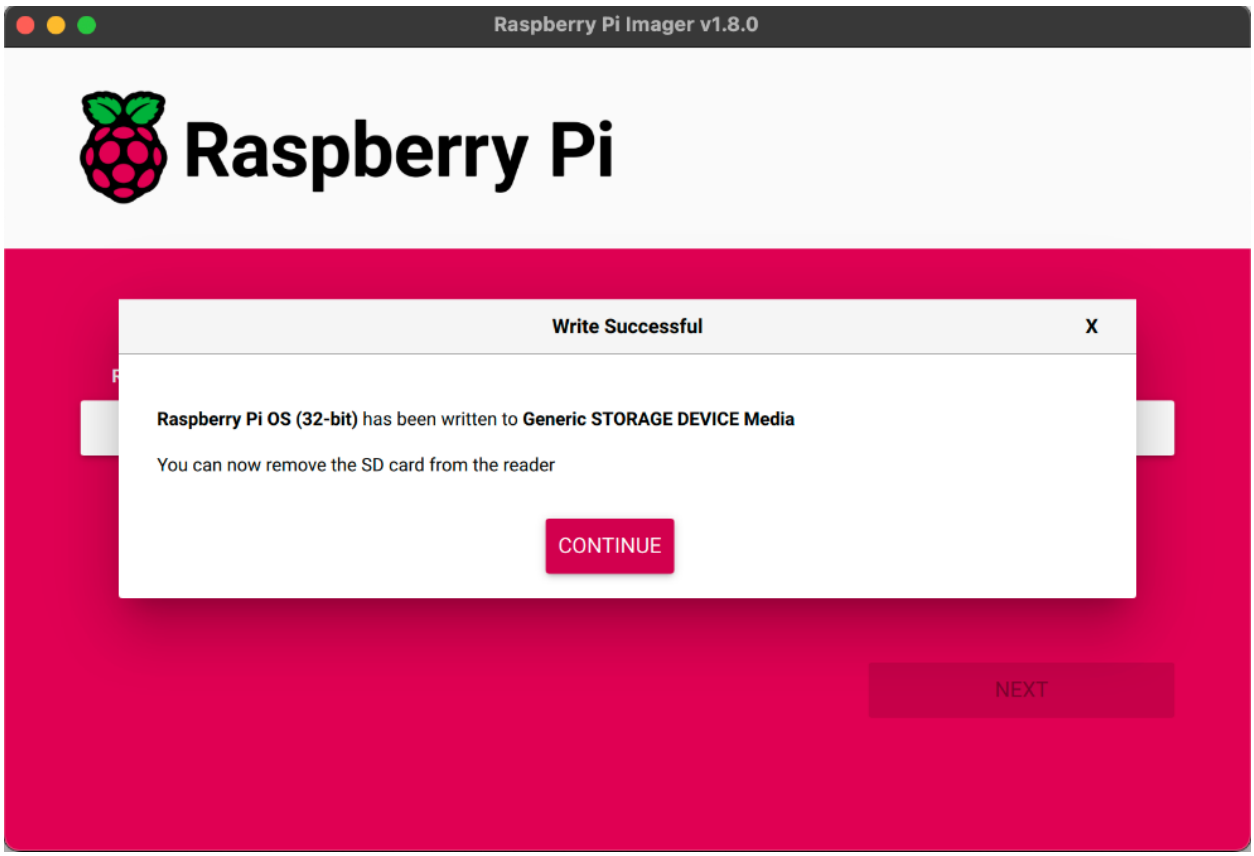
•Choose Allow public-key authentication only to preconfigured your Raspberry Pi for password less public-key SSH authentication using a private key from the computer you're currently using. If already have an RSA key in your SSH configuration, Imager uses that public key. If you don't,

you can click Run SSH-key gen to generate a public/private key pair. Imager will use the newly-generated public key.

If you see an admin prompt asking for permissions to read and write to your storage medium, grant Imager the permissions to proceed.



When you see the "Write Successful" popup, your image has been completely written and verified. You're now ready to boot a Raspberry Pi from the storage device!



Install over the network

Network Install enables a Raspberry Pi to install an operating system on a storage device using a version of Raspberry Pi Imager downloaded over the network. With Network Install, you can get an operating system installed on your Raspberry Pi with no separate SD card reader and no computer other than your Raspberry Pi. You can run Network Install on any compatible storage device, including SD cards and USB storage.

Network Install only runs on Raspberry Pi 4, 400, and 5. If your Raspberry Pi runs an older boot loader, you may need to update the bootloader to use Network Install.

Network Install requires the following:

- a compatible Raspberry Pi model running firmware that supports Network Install
- a monitor
- a keyboard

- a wired internet connection

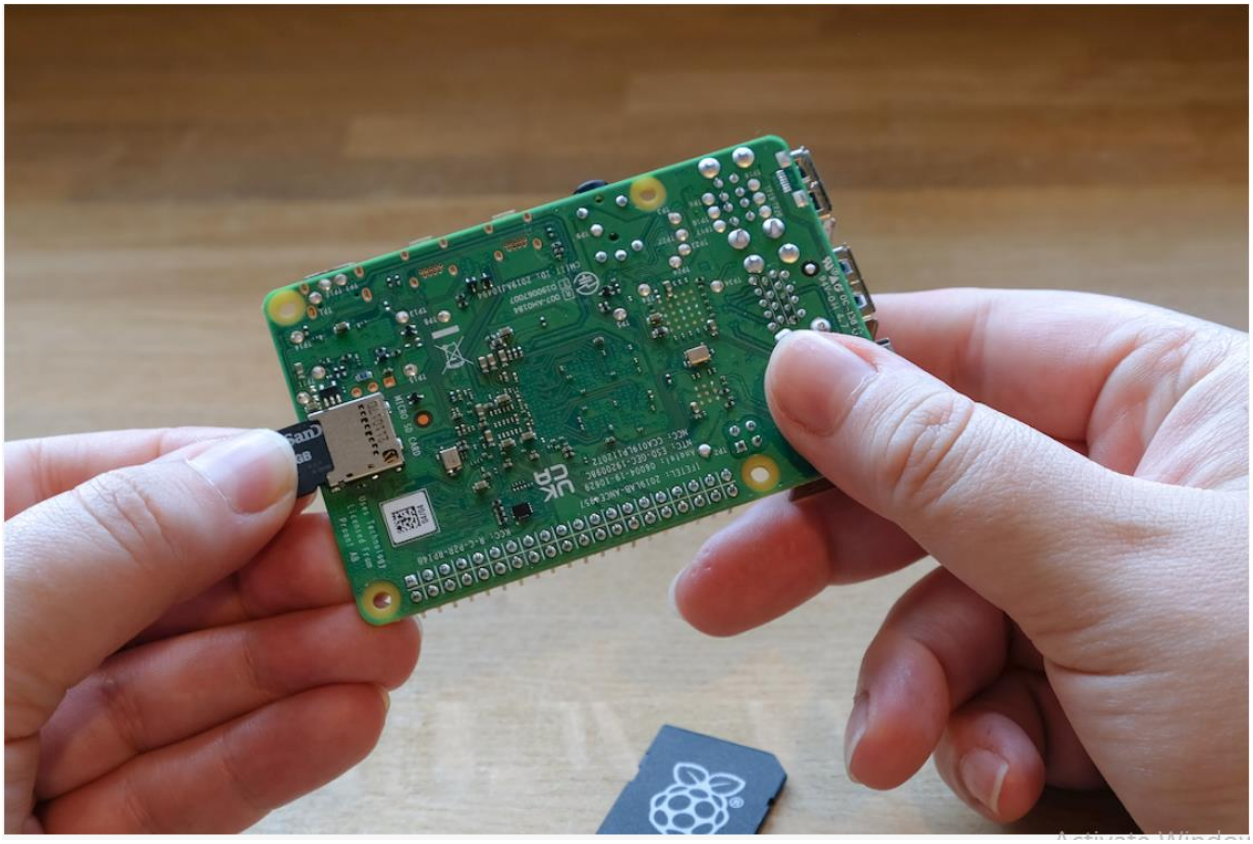
To launch Network Install, power on your Raspberry Pi while pressing and holding the SHIFT key in the following configuration:

- no bootable storage device
- attached keyboard
- attached compatible storage device, such as an SD card or USB storage

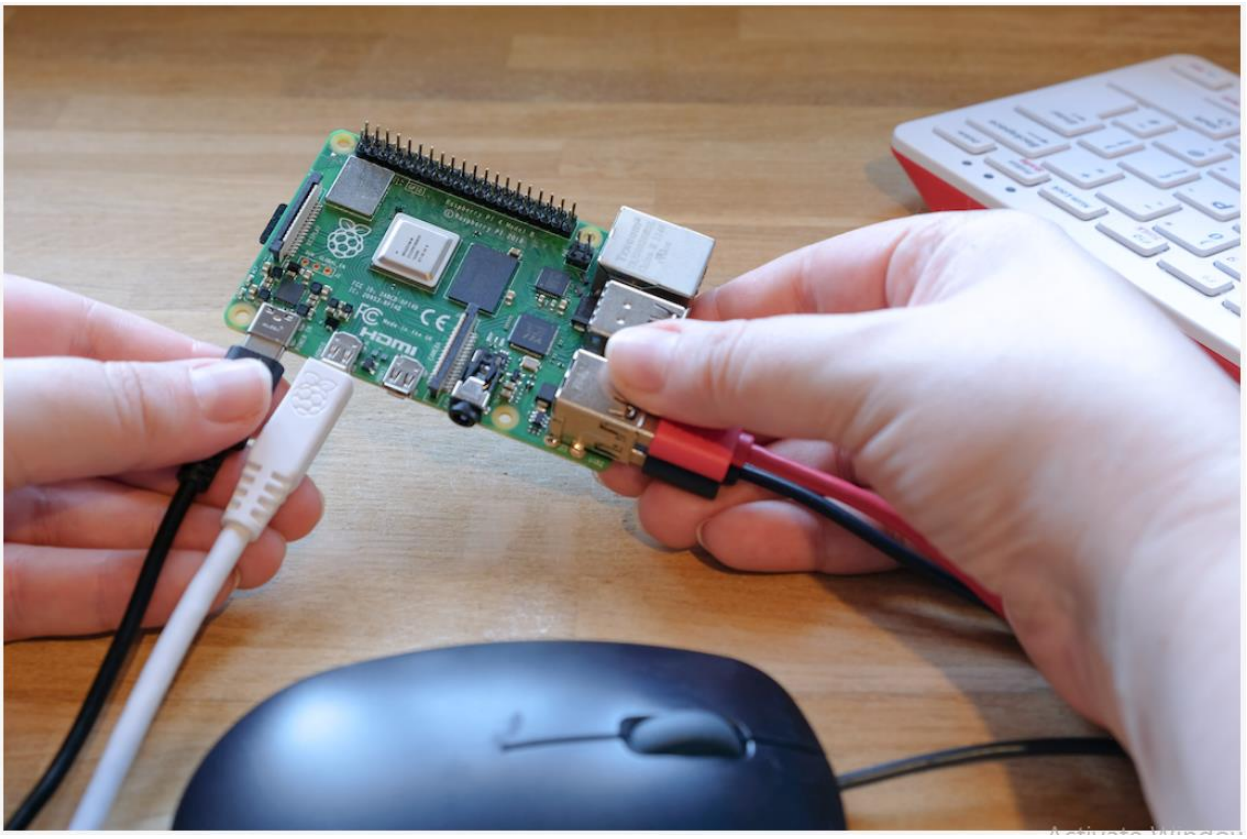
Set up your Raspberry Pi

After installing an operating system image, connect your storage device to your Raspberry Pi.

First, unplug your Raspberry Pi's power supply to ensure that the Raspberry Pi is powered down while you connect peripherals. If you installed the operating system on a microSD card, you can plug it into your Raspberry Pi's card slot now. If you installed the operating system on any other storage device, you can connect it to your Raspberry Pi now.

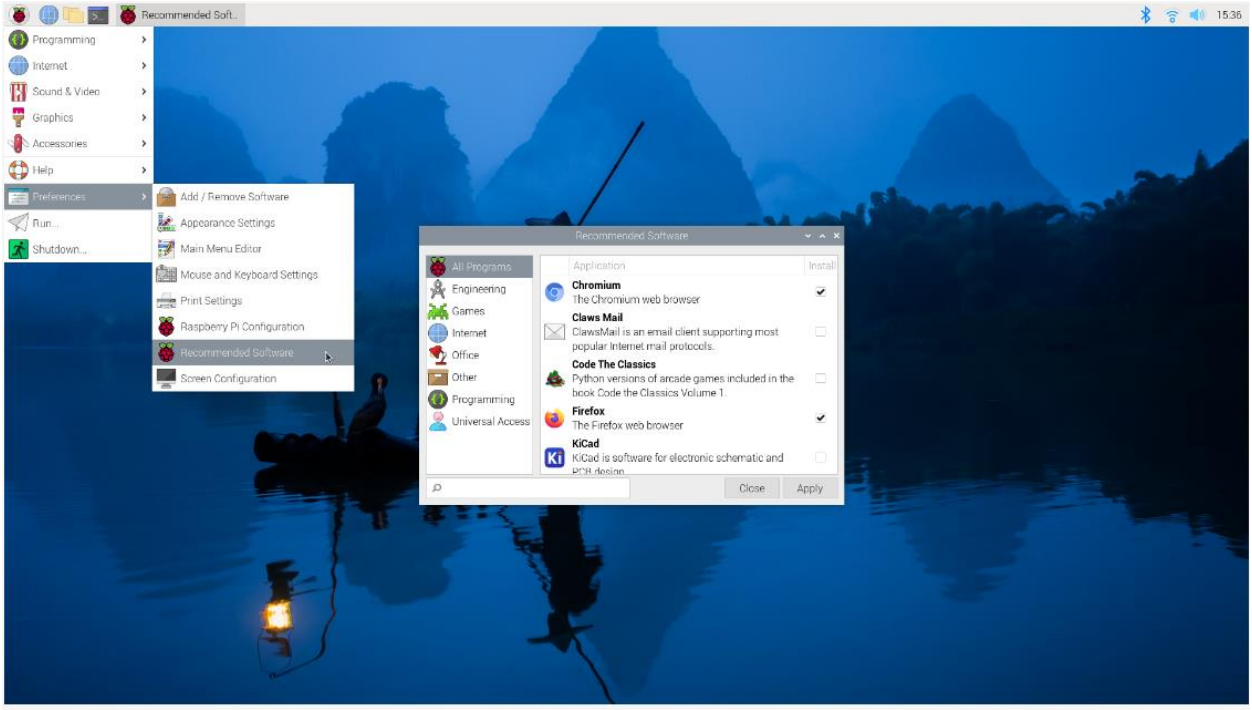


Then, plug in any other peripherals, such as your mouse, keyboard, and monitor.



Finally, connect the power supply to your Raspberry Pi. You should see the status LED light up when your Pi powers on. If your Pi is connected to a display, you should see the boot screen within minutes.

Raspberry Pi OS comes with many essential applications pre-installed so you can start using them straight away. If you'd like to take advantage of other applications we find useful, click the raspberry icon in the top left corner of the screen. Select Preferences > Recommended Software from the drop-down menu, and you'll find the package manager. You can install a wide variety of recommended software here for free.



if you plan to use your Raspberry Pi as a home computer, you might find Libre Office useful for writing and editing documents and spreadsheets. You can also make your Raspberry Pi more accessible with apps like a screen magnifier and Orca screen reader, found under Universal Access.

CHAPTER 3

SOFTWARE DESCRIPTION

This chapter gives the details about the software used in the project human detection in disaster zones along with some of the libraries and package installations.

7.1 ABOUT SOFTWARE

Raspberry pi need to be preloaded with any of the OS into the SD card which is inserted into it. In this project Raspberry Jessie is used to operate the robot whose detailed installation procedure is provide in this chapter.

7.1.1 RASPBERRY PI INSTALLATION

Preparing your SD card for the Raspberry Pi

The SD card contains the Raspberry Pi's operating system (the OS is the software that makes it work, like Windows on a PC or OSX on a Mac). This is very different from most computers and it is what many people find the most daunting part of setting up their Raspberry Pi. It is actually very straightforward just different!

The following instructions are for Windows users. Linux and Mac users can find instructions at www.raspberrypi.org/downloads

a. Download the Raspberry Pi operating system The recommended OS is called Raspberry an. Download it here: http://downloads.raspberrypi.org/images/raspbian/2012-12-16-wheezy-raspbian/2012-12-16-wheezy-raspbian.zip
b. Unzip the file that you just downloaded
    a. Right click on the file and choose "Extract all".
    b. Follow the instructions you will end up with a file ending in.img
    c. This.img file can only be written to your SD card by special disk imaging software, so…

c. Download the Win32DiskImager software
    a. Download win32diskimager-binary.zip (currently version 0.6) from: https://launchpad.net/win32-image-writer/+download
    b. Unzip it in the same way you did the Raspberrian .zip file
    c. You now have a new folder called win32diskimager-binary you are now ready to write the Raspbian image to your SD card.

d. Writing Raspbian to the SD card
    a. Plug your SD card into your PC
    b. In the folder you made in step 3(b), run the file named Win32DiskImager.exe (in Windows Vista, 7 and 8 we recommend that you right-click this file and choose "Run as administrator"). You will see something like this:

Fig 3.1 Disk Imager window

    c. If the SD card (Device) you are using isn't found automatically then click on the drop down box and select it
    d. In the Image File box, choose the Raspbian.img file that you downloaded

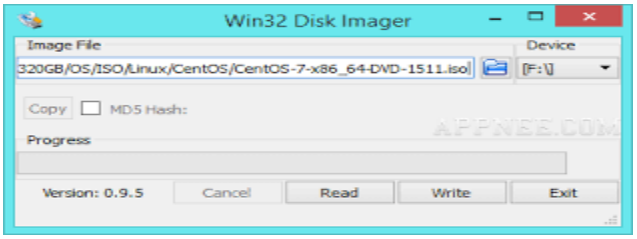


Fig 3.2 Image file window

    e. Click Write
    f. After a few minutes you will have an SD card that you can use in your Raspberry Pi

BOOTING YOUR RASPBERRY PI FOR THE FIRST TIME

a. Follow the Quick start guide on page 1
b. On first boot you will come to the Raspi-config window
c. Change settings such as time zone and locale if you want
d. Finally, select the second choice: expand roots and say 'yes' to a reboot
e. The Raspberry Pi will reboot and you will see raspberry Pi login: Type: **pi**
f. You will be asked for your Password Type: **raspberry**
g. You will then see the prompt: pi @raspberry ~ $
h. Start the desktop by typing: **start x**
i. You will find yourself in a familiar-but-different desktop environment.
j. Experiment, explore and have fun!

7.1.2 PYTHON

Python is a wonderful and powerful programming language that's easy to use (easy to read and write) and with Raspberry Pi lets you connect your project to the real world.Python is a powerful modern computer programming language. It bears some similarities to FORTRAN, one of the earliest programming languages, but it is much more powerful than FORTRAN. Python allows you to use variables without declaring them (i.e., it determines types implicitly), and it relies on indentation as a control structure. You are not forced to define classes in Python (unlike Java) but you are free to do so when convenient. This document focuses on learning Python for the purpose of doing mathematical calculations. We assume the reader has some knowledge of basic mathematics, but we try not to assume any previous exposure to computer programming, although some such exposure would certainly be helpful. Python is a good choice for mathematical calculations, since we can write code quickly, test it easily, and its syntax is similar to the way mathematical ideas are expressed in the mathematical literature. By learning Python you will also be learning a major tool used by many web developers.

Fig 3.3 Python Logo

Python syntax is very clean, with an emphasis on readability and uses Standard English keywords. Start by opening IDLE from the desktop.

IDLE

The easiest introduction to Python is through IDLE, a Python development environment. Open IDLE from the Desktop or applications menu:

Fig 3.4: Programming Process

IDLE gives you a REPL (Read-Evaluate-Print-Loop) which is a prompt you can enter Python commands in to. As it's a REPL you even get the output of commands printed to the screen without using

DEFINING FUNCTIONS

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have needed to perform a certain type of calculation many times, and then define a function. For a simple example, the compound command

```
>>>def f(x):

... return x*x

...
```

Defines the squaring function f(x) = x2, a popular example used in elementary math courses. Inthe definition, the first line is the function header where the name, f, of the function is specified.Subsequent lines give the body of the function, where the output value is calculated. Note thatthe final step is to return the answer; without it we would never see any results. Continuing theexample, we can use the function to calculate the square of any given input:

```
>>>f(2)

4

>>>f (2.5)

6.25
```

The name of a function is purely arbitrary. We could have defined the same function as above, but with the name square instead of f; then to use it we use the new function name instead of the old:

```
>>>def square (x):

... return x*x

...

>>> square (3)

9

>>> square (2.5)

6.25
```

Actually, a function name is not completely arbitrary, since we are not allowed to use a reserved word as a function name. Python's reserved words are: and, def, del, for, is, raise, assert, elf, from, lambda, return, break, else, global, not, try, class, except, if, or, while, continue, exec, import, pass, yield. By the way, Python also allows us to define functions using a format similar

to the Lambda Calculus in mathematical logic. For instance, the above function could alternatively be defined in the following way:

```
>>>square = lambda x: x*x
```

Here lambda x: x*x is known as a lambda expression. Lambda expressions are useful when you need to define a function in just one line; they are also useful in situations where you need a function but don't want to name it. Usually function definitions will be stored in a module (file) for later use. These are indistinguishable from Python's Library modules from the user's perspective.

INSTALLATION

If you use Mac OS X or Linux, then Python should already be installed on your computer by default. If not, you can download the latest version by visiting the Python home page, at

http://www.python.orgwhere you will also find loads of documentation and other useful information. Windows users can also download Python at this website. Don't forget this website it is your first point of reference for all things Python.

GETTING STARTED

The easiest way to get started is to run Python as an interpreter, which behaves similar to the way one would use a calculator. In the interpreter, you type a command, and Python produces the answer. Then you type another command, which again produces an answer, and so on. In OS X or Linux, to start the Python interpreter is as simple as typing the command python on the command line in a terminal shell. In Windows, assuming that Python has already been installed, you need to find Python in the appropriate menu. Windows users may choose to run Python in a command shell (i.e., a DOS window) where it will behave very similarly to Linux or OS X. For all three operating systems (Linux, OS X, Windows) there is also an integrated development environment for Python named IDLE. If interested, you may download and install this on your computer. Once Python starts running in interpreter mode, using IDLE or a command shell, it produces a prompt, which waits for your input. For example, this is what I get when I start Python in a command shell on my Linux box:

```
Doty @brauer:~% python
Python 2.5.2 ( r252 :60911 , Apr 21 2008 , 11:12:42)
[GCC 4.2.3 (Ubuntu 4.2.3 -2ubuntu7)] on linux2
Type “help", “copyright",” credits" or “license" for more
information.
```

Where the three symbols >>> indicates the prompt awaiting my input.

LOADING COMMANDS FROM THE LIBRARY

Python has a very extensive library of commands, documented in the Python Library Reference manual. These commands are organized into modules. One of the available modules is especially useful for us: the math module. Let’s see how it may be used.

```
>>>from math import sqrt , exp
>>>exp( -1)
0.36787944117144233
>>>sqrt (2)
1.4142135623730951
```

We first import the sqrt and exp functions from the math module, and then use them to compute e−1 = 1/e and p2. Once we have loaded a function from a module, it is available for the rest of that session. When we start a new session, we have to reload the function if we need it. Note that we could have loaded both functions sqrt and exp by using a wildcard

This tells Python to import all the functions in the math module. What would have happened if we forgot to import a needed function? After starting a new session, if we type

```
>>>sqrt (2)
Trace back (most recent call last):
File "<stdin>", line 1, in <module >
Name Error: name ’sqrt’ is not defined
```

We see an example of an error message, telling us that Python does not recognize sqrt.

OPENCV

Open CV-Python is a library of Python bindings designed to solve computer vision problems. ... Open CV-Python makes use of Numpy, which is a highly optimized library for numerical operations with MATLAB-style syntax. All the Open CV array structures are converted to and from Numpy arrays. Open CV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. In simple language it is library used for Image Processing. It is mainly used to do all the operation related to Images.

CHAPTER 4

GENERAL PROGRAMS

1.BUZZER RASPBERRY PI PROGRAM:-

```
import RPI.GPIO as GPIO
  import time
GPIO.setmode(GPIO.BCM)
GPIO.set warnings(False)
GPIO.setup(4,GPIO.OUT)
   while True:
    Print("buzzer On")
GPIO.output(4,GPIO,HIGH)
time.sleep(1)
Print("buzzer Off")
GPIO.output(4,GPIO,LOW)
  time.sleep(1)
```

2.SENSOR PROGRAMMING WITH BUZZER:-

```
import Rpi.GPIO as GPIO
  impot time
GPIO.setmode(GPIO.BCM)
```

```
GPIO.set warnings(False)
  Obs=2
  buz=3
GPIO.setup(2,GPIO.IN)
GPIO.setup(3,GPIO.OUT)
 while True:
   ob=GPIO.input(Obs)
   Print("OBJ:"+str(Ob))
   time.sleep(1)
   if(Ob==0);
GPIO.output(buz,1)(or)GPIO.output(3,GPIO,HIGH)
else;
GPIO.output(buz,0)(or)GPIO.output(3,GPIO.LOW)
```

3.STREET LIGHT PROGRAM:-

```
import Rpi.GPIO.as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.set warnigs(FALSE)
LDR=17
LED=22
GPIO.setup(LDR,GPIO.IN)
```

```
GPIO.setup(LED,GPIO.OUT)

while True;

LDR-sensor=GPIO.input(LDR)

Print("OBJ:")+str(LDR_sensor))

time.sleep(1)

if(LDR_sensor==0)

GPIO.Output(LED,GPIO,HIGH)

else

GPIO.output(LED,GPIO,LOW)
```

4.CAMERA USING FACE RECOGNIZATION CODE:-

```
import numpy as np

import Cv2

import time

Cap=Cv2_videocapture(0)

time.sleep(2)

while Cap.is Opened(1);

img=Cap.read()

Cv2.im show('img',img)

if Cv2.waitkey(1)==Ord('s')

break
```

```
Cap.release()

Cv2.destroy All Windows()
```

5.GAS LEAKAGE DETECTING USING RASPBERRY PI:-

```
import time

import Rpi.GPIO.as GPIO

import Urllib.request

GPIO.setmode(GPIO.BCM)

GPIO.set warning(False)

buz=3

gs=2

GPIO.setup(buz,GPIO.OUT)

GPIO.setup(gs,GPIO.IN)

while True:

gval=GPIO.input(gs)

Print("GAS:"+str(gval))

time.sleep(1)

if(gval==0)

GPIO.Output(buz,GPIO,HIGH)

else:

GPIO.Output(buz,GPIO,LOW)
```

6.ULTRA SONIC SENSOR WITH LCD:-

```
import Rpi,GPIO.as GPIO
from time import sleep
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
LCD_RS=21
LCD_E=20
LCD_D4=16
LCD_D5=26
LCD_D6=19
LCD_D7=13
TRIG=27
ECHO=17
GPIO.setup(TRIG.GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
GPIO.setup(LCD_E,GPIO.OUT)
GPIO.setup(LCD_D3,GPIO.OUT)
GPIO.setup(LCD_D4,GPIO.OUT)
GPIO.setup(LCD_D5,GPIO.OUT)
GPIO.setup(LCD_D6,GPIO.OUT)
```

```
GPIO.setup(LCD_D7,GPIO.OUT)
LCD_WIDTH=16
LCD_CHR=True
LCD_CMD=False
LCD_LINE-1=0*80
LCD_LINE-2=0*CO
E_PULSE=0.0005
E_DELAY=0.0005
def lcd_init();
Lcd_byte(0*33,LCD_CMD)
Lcd_byte(0*32,LCD_CMD)
Lcd_byte(0*06,LCD_CMD)
Lcd_byte(0*0C,LCD_CMD)
Lcd_byte(0*28,LCD_CMD)
Lcd_byte(0*01,LCD_CMD)
  time.sieep(E_DELAY)
  deflcd_byte(bits,mode);
GPIO.output(LCD_RS,mode)
GPIO.output(LCD_D4,False)
GPIO.output(LCD_D5,False)
GPIO.output(LCD_D6,False)
GPIO.output(LCD_D7,False)
  if bits&0*10= =0*10;
```

```
    GIPO.output(LCD_D4,True)
 if bits&0*20= =0*20;
    GIPO.output(LCD_D5,True)
 if bits&0*40= =0*40;
    GIPO.output(LCD_D6,True)
 if bits&0*50= =0*80;
    GIPO.output(LCD_D6,True)
  Lcd_taggle_enable()
  GIPO.output(LCD.D4,False)
  GIPO.output(LCD.D5,False)
  GIPO.output(LCD_D6,False)
  GIPO.output(LCD_D7,False)
  if bits&0*01= =0*01;
     GIPO.output(LCD_D4,True)
  if bits&0*02= =0*02;
     GIPO.output(LCD_D5,True)
  if bits&0*04= =0*04;
     GIPO.output(LCD_D6,True)
  if bits&0*08= =0*08;
     GIPO.output(LCD_D7,True)
    Lcd_taggle_enable()
   def lcd_taggle_enable()
       time.sleep(E_DELAY)
```

```
        GIPO.output(LCD_E,True)
       time.sleep(E_Pulse)
        GIPO.output(LCD_E,False)
       time.sleep(E_DELAY)
     def lcd_string(message,line);
message=message_just(LCD_WIDTH;" ")
    LCD_byte line,LCD_CMD)
  for i in range(LCD_WIDTH);
      LCD_byte(ord(message[1],LCD_CHR)
      LCD_init()
    lcd_byte(0*01,LCD_CMD)
    lcd_string("WELCOME",LCD_LINE-1)
          while(True);
       GPIO.Output(TRIG,True)
            time.sleep(0.00001)
         GPIO.Output(TRIG,False)
            while GPIO.input(ECHO)==0;
           pulse_start=time.time()
         while GPIO.input(ECHO)==1;
              pulse_end=time.time()
         pulse_duration=pulse_end_pulse_start
         ditance1=pulse_duration*17150
           d1=int(distance 1+1.15)
```

```
print("LEVEL:"+str(d1)+'cm')

print()

 lcd_string("L:"+str(d1),LCD_LINE_1)
```

CHAPTER 5

HOME AUTHORIZATION BASED AUTOMATIC SYSTEMS

I.INTRODUCTION:

Our houses, offices and other private properties are always at risk of being attacked by intruders that would be accessing them for many different reasons, such as theft, damaging property etc. Hence, restricting access to our properties is very crucial and would surely protect against intruders. One of the most efficient ways is the use facial recognition in smart home applications. Facial recognition can be implemented by using various algorithms that would be able to identify and highlight specific facial characteristics. Especially, during these ongoing circumstances involving Covid-19; it is a much better option than anything that would include touching such as using pins or fingerprints. Implementing such a method in a system that could be connected to the internet was made possible by using the Internet of Things technology. It is a technology of wirelessly interconnected devices over a network that can communicate with each other in real-time using the internet without any human intervention. In other words, it can be considered as machine-to-machine communication. This technology can collect data and information, analyse it and then perform an action based on the information gathered. This goal of this paper is to design a low-cost intrusion detection system using a Raspberry Pi 4 that is capable of the following:

- Executing detection/recognition algorithms upon motion detection and controlling the door look according to its results.
- Act as a smart intercom system and initiate an automatic video call with the owner of the property.
- Allow for wireless control over the door lock using an android mobile application developed specifically for this system
- Obtain the body temperature of the person standing by the door as a form of COVID-19 precautionary measure. This feature will be operating in two operational modes.

Many techniques have been innovated to create home automated systems. B. Shinde et. al. in has developed a home automation system using an Arduino microcontroller, along with an android mobile application that sends commands to the Arduino which will control the appliances accordingly. Both are connected to each other via Wi-Fi. In Krishna Rathi et. al. made use of hand gestures to control the appliances, by using an accelerometer, Raspberry pi, flex sensor

and an Arduino kit along with a hand the appliances at home. Whereas Saddam in, made use of the DTMF technology, this is where each number on the keypad has two frequencies, so by using a DTMF decoder, we can assign each number a command that must be performed when pressed. For merely security purposes, B Madhura Vani. et. al in, developed a system using Raspberry Pi, where When motion is detected, the face will be captured and processed by a face recognition algorithm, which will alert by a buzzer and sending an email/SMS, if the face is not recognized. Speech recognition technique was implemented by Nainsi Soni. et. al.in . This system allows total control on home appliances by speech commands only. The process of achieving this technique needs signal pre-processing, feature extraction from the signal, language and pronunciation modelling and finally decoding the message. Utilizing such system in home automation can be extremely helpful for people who might struggle with blindness or immobility. Sudhasmita Behera. et.al in designed a system that controls home appliances using SMS messages in which GSM wireless technology was used. First, the user sends SMS message to the system with the desired commands that needs to be done. Second, the system stores this message and generates control signal. Finally, the signal is sent to the hardware that was programmed to perform these commands. LabVIEW is a programming software that provides graphical approach to develop applications that requires control and measure. Due to its features and the ability to visualize every part of the program, S Jermilla. et. al in used it in a home automation project.

II.OBJECTIVE:

To design and implement a robust home authorization system utilizing Raspberry Pi 4 and camera technology, aimed at enhancing security and accessibility through automated entry management and monitoring.

The aim of this project is to develop a comprehensive home authorization system leveraging the capabilities of Raspberry Pi 4 and advanced camera technology. By integrating these technologies, the system seeks to significantly enhance both security and accessibility within residential environments. The primary goal is to create a reliable and efficient automated entry management system that can accurately authenticate and authorize individuals based on visual recognition.

Through the utilization of Raspberry Pi 4, the system will capitalize on its processing power and connectivity options to handle real-time data from multiple cameras effectively. This setup will

enable seamless monitoring of entry points and facilitate immediate responses to access requests or security breaches. The system's design will prioritize user-friendly interfaces and intuitive controls to ensure ease of operation for homeowners and authorized personnel.

Furthermore, the implementation will incorporate robust encryption protocols and secure authentication mechanisms to safeguard sensitive data and prevent unauthorized access attempts.

By deploying this technology, the project aims to establish a scalable solution adaptable to various home configurations and security requirements.

Ultimately, the envisioned system aims not only to streamline entry procedures but also to provide homeowners with enhanced peace of mind by offering reliable surveillance capabilities. By achieving these objectives, the project intends to set a new standard in residential security systems, combining innovation with practicality to meet the evolving needs of modern households.

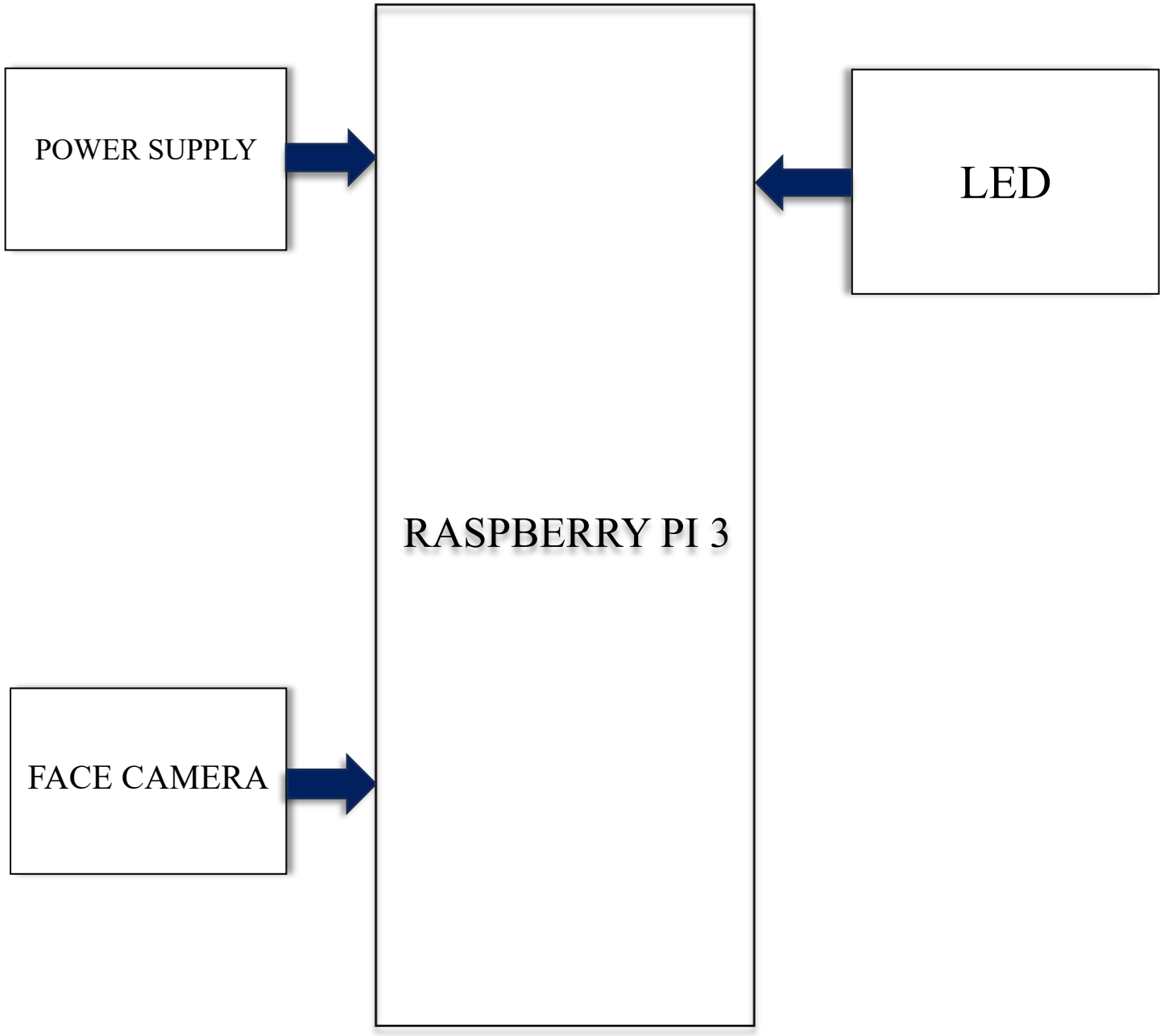
III.PROBLEM DEFINATION:

The challenge addressed in this project is the need for an efficient and secure home authorization system using Raspberry Pi 4 and camera technology. Current methods of managing home access often rely on traditional key-based systems or basic electronic locks, which can be cumbersome to manage and potentially insecure. There is a growing demand for a more sophisticated solution that integrates modern technologies to enhance security, streamline access management, and provide real-time monitoring capabilities.

The existing systems may lack flexibility in adapting to diverse household needs, such as granting temporary access to guests or service providers without compromising security. Moreover, manual monitoring of entry points can be time-consuming and prone to human error, necessitating a more automated approach that ensures accurate identification and authorization of individuals entering the premises.

Therefore, the project aims to develop a robust home authorization system that leverages the computational power of Raspberry Pi 4 and the visual recognition capabilities of cameras to address these challenges. The system will be designed to provide seamless and reliable authentication processes, allowing homeowners to remotely manage access permissions and monitor entry points from any location.

IV.BLOCK DIAGRAM:



V. WORKING METHADOLOGY:

Designing a home authorization-based automatic system using Raspberry Pi 3 involves several systematic steps. Initially, you define the system requirements, identifying what functionalities are needed, such as access control to specific areas or devices within the home. Once the requirements are clear, the Raspberry Pi 3 is set up with an appropriate operating system, typically Raspberry Pi OS, and necessary peripherals like sensors and actuators (e.g., electronic locks, lights) are connected to GPIO pins or USB ports.

The core of the system revolves around programming the Raspberry Pi 3, usually in Python due to its compatibility and ease of use with the Pi's GPIO libraries. This involves developing the logic for user authorization, which could be based on various inputs such as, biometric data (like facial recognition), or PIN entry. A database is often utilized to store user credentials and access permissions securely.

Once the authorization logic is implemented, integration testing ensures that all components work together seamlessly. This phase includes verifying sensor inputs, database interactions, and proper functioning of actuators based on authorization decisions. Throughout the process, debugging is crucial to identify and resolve any issues.

Finally, the deployed system is monitored for performance and security, with provisions made for future updates or enhancements. Documentation is essential to guide users on system operation and maintenance. By following this methodology, a robust home authorization system using Raspberry Pi 3 can be effectively developed and deployed to enhance security and convenience within a home environment.

VI.HARDWARE COMPONENTS REQUIRED:

1. **Raspberry Pi 3:** The Raspberry Pi 3 Model B is the latest version of the low-cost Raspberry Pi computer. The Pi isn't like your typical device; in its cheapest form it doesn't have a case, and is simply a credit-card sized electronic board -- of the type you might find inside a PC or laptop, but much smaller. It costs as little as $35, although you might want to choose the $55 version with its 4GB of RAM for its better all-round performance.

Fig 5.1 Raspberry Pi 3 Model B Board

2. **Led :** LED stands for Light Emitting Diode. It is a semiconductor device that emits light when an electric current passes through it. LEDs are used in various applications such as indicator lights, display screens, automotive lighting, household lighting, and many more. They are known for their energy efficiency, durability, and versatility compared to traditional lighting technologies like incandescent bulbs and fluorescent lights. LED technology has advanced significantly in recent years, making them a popular choice for

both commercial and residential lighting due to their long lifespan and lower energy consumption.

General Purpose Input/Output pins on the Raspberry Pi 3 used for interfacing with external components like LED's. A current-limiting resistor (typically 330 ohms) connected in series with the LED to prevent it from burning out due to excessive current

from the GPIO pin .Light Emitting Diode that emits light when current flows through it in the correct direction (from anode to cathode).

Fig 5.2: LED

3. **Face Camera :** It can be used with software to recognize faces for security purposes, access control systems, or even for fun projects like identifying friends and family.It can detect the presence of a face within its field of view and track its movements. This is useful in applications where you need to follow a person's movement or keep them in focus for a video call or similar usess part of a setup for video conferencing or video calls, a face camera can provide a clear image of the user's face for communication purposes.When choosing a face camera for a Raspberry Pi 3, consider factors such as resolution, frame

rate, ease of integration with the Raspberry Pi (typically through the CSI or USB interface), and compatibility with any specific software or libraries you plan to use for facial recognition or other applications. Popular choices include the Raspberry Pi Camera Module (officially supported by Raspberry Pi Foundation) and various USB webcams that are compatible with the Raspberry Pi ecosystem.

Fig 5.3: Camera

4. **Power Supply :** Raspberry Pi 3 requires a 5V DC input. The official recommendation is to use a power supply that can provide 5V with a minimum current capacity of 3A for stable operation, especially if you plan to use peripherals that draw additional power (like external hard drives or high-power USB devices).The Raspberry Pi 3 uses a USB-C connector for power input. It's essential to use a good-quality USB-C cable that can handle the power requirements without voltage drops or interference. Ensure the power adapter you use is capable of supplying sufficient current and voltage stability. Using underpowered adapters (e.g., smartphone chargers rated below 3A) can lead to instability, especially under load. The Raspberry Pi Foundation offers an official USB-C power supply designed specifically for the Raspberry Pi.

   This power supply provides 5.1V at 3A, which meets the recommended specifications and ensures reliable performance. The power consumption of the Raspberry Pi 3 can vary depending on the workload and peripherals connected. For typical use cases without additional high-power peripherals, a 3A power supply should suffice. If you plan to connect power-hungry USB devices or peripherals, consider a higher-rated power supply (up to 3.5A or more). The Raspberry Pi 4 has improved power management compared to earlier models, but it's still important to ensure a stable power supply to avoid issues like undervoltage warnings or system crashes.

Fig 5.4: Power Supply Adapter

VII.PROGRAM:

```
import cv2

import face_recognition

import RPi.GPIO as GPIO

import os

import time


# Setup GPIO for relay control

RELAY_PIN = 17  # Adjust GPIO pin as necessary

GPIO.setmode(GPIO.BCM)

GPIO.setup(RELAY_PIN, GPIO.OUT)
```

```
def load_known_faces():
    known_face_encodings = []
    known_face_names = []

    # Load images from the dataset folder
    for filename in os.listdir('dataset'):
        if filename.endswith('.jpg'):
            image = face_recognition.load_image_file(f"dataset/{filename}")
            encoding = face_recognition.face_encodings(image)[0]
            known_face_encodings.append(encoding)
            known_face_names.append(filename.split('_')[0])  # Extract name from filename

    return known_face_encodings, known_face_names

def control_device(action):
    if action == 'on':
        GPIO.output(RELAY_PIN, GPIO.HIGH)  # Turn on the device
        print("Device turned ON")
    elif action == 'off':
        GPIO.output(RELAY_PIN, GPIO.LOW)   # Turn off the device
        print("Device turned OFF")
```

```
def main():
    known_face_encodings, known_face_names = load_known_faces()
    cam = cv2.VideoCapture(0)
    while True:
        ret, frame = cam.read()
        if not ret:
            print("Failed to grab frame")
            break
        # Find all the faces and face encodings in the current frame
        face_locations = face_recognition.face_locations(frame)
        face_encodings = face_recognition.face_encodings(frame, face_locations)

        for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
            matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
            name = "Unknown"

            # Use the known face with the smallest distance to the new face
            face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
            best_match_index = np.argmin(face_distances)
            if matches[best_match_index]:
                name = known_face_names[best_match_index]

            # Draw a rectangle around the face and label it
```

```
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 2)
            cv2.putText(frame, name, (left, top - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2)

            # If recognized, control the device
            if name != "Unknown":
                control_device('on')  # Turn on the device
                time.sleep(5)  # Keep it on for 5 seconds
                control_device('off')  # Turn off the device

        # Display the resulting frame
        cv2.imshow('Video', frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cam.release()
    cv2.destroyAllWindows()
    GPIO.cleanup()

if __name__ == "__main__":
    main()
```

VIII.RESULT:

Enhanced home security through biometric and RFID systems reduces the risk of unauthorized access. Users can easily enter their homes without traditional keys. Comprehensive logs help track who enters and exits the home, providing insights into activity. Users receive real-time notifications, allowing them to respond quickly to unauthorized access attempts.

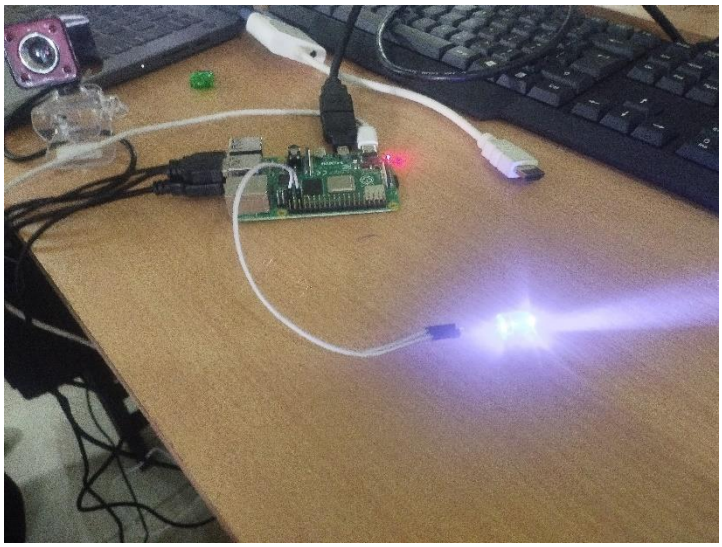


Fig 7.1: Light is ON for Detected

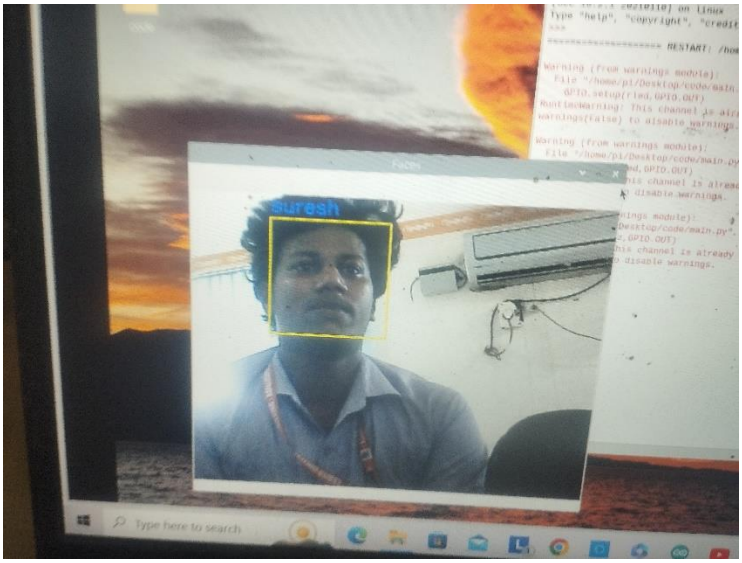


Fig 7.2: Detecting the Person

IX.CONCLUSION:

Implementing a home authorization-based automatic system with a Raspberry Pi 3 can provide robust security, ease of use, and a modern approach to home management. The versatility of the Raspberry Pi allows for expansion and integration with other smart home technologies.

X.FUTURE SCOPE:

The future of home authorization-based automatic systems using Raspberry Pi 3 is bright, with vast potential for innovation and improvement. As technology advances, these systems will likely become more sophisticated, user-friendly, and integral to modern living. Homeowners can look forward to enhanced security, convenience, and connectivity in their homes.

XI.REFERENCE:

- Online Resources.
- A community-driven platform with step-by-step guides on various home automation projects using Raspberry Pi.
- "Raspberry Pi Home Automation with Arduino" by Andrew K. Dennis Book.
- "Getting Started with Raspberry Pi" by Matt Richardson and Shawn Wallace Book.
- Research Papers and Articles.

These resources should provide you with a solid foundation to explore home authorization-based automatic systems using Raspberry Pi 3.

TOTAL REPORT ACTIVITY LOG FOR 8 WEEKS

Week & Date (03-06-2024 to 27-07-2024)	Brief description of the Weekly Activity	Learning outcome	Person in charge signature
Week 1 (03-06-2024 to 08-06-2024)	Introduction to Block Diagram of ES & Microcontrollers	Introduction to Microcontrollers and B.D. of ES	
Week 2 (10-06-2024 to 15-06-2024)	Introduction to Hardware Modules like Power supply, Sensors, LED's etc.,	Introduction to Hardware Modules	
Week 3 (17-06-2024 to 22-06-2024)	Raspberry pi Installation Procedure	I done the Raspberry pi Installation	
Week 4 (24-06-2024 to 29-06-2024)	Explain the Buzzer program using Raspberry pi	I learned the Buzzer program	
Week 5 (01-07-2024 to 06-07-2024)	Explain the Sensor Programming with Buzzer	I learned the Sensor Programming with Buzzer	
Week 6 (08-07-2024 to 13-07-2024)	Explain the Street Light & Camera using Face Recognition programs	I learned the Street Light & Camera using Face Recognition sensor programs	

Week & Date (22-05-2023 to 17-06-2023)	**Brief description of the Weekly Activity**	**Learning outcome**	**Person in charge signature**
Week 7 (15-07-2024 to 20-07-2024)	Explain the Gas Leakage detecting using Raspberry pi program	I learned the Gas Leakage detecting using Raspberry pi program	
Week 8 (22-07-2024 to 27-07-2024)	Explain the ultrasonic sensor with LCD Program	I learned the ultrasonic sensor with LCD Program	