

Raspberry Pi

Introduction to Raspberry Pi

- Raspberry Pi, developed by Raspberry Pi Foundation in association with Broadcom, is a series of small single-board computers and perhaps the most inspiring computer available today.
- Earlier, the Raspberry Pi was used to teach basic computer science in schools but later, because of its low cost and open design, the model became far more popular than anticipated.
- It is widely used to make gaming devices, fitness gadgets, weather stations, and much more. But apart from that, it is used by thousands of people of all ages who want to take their first step in computer science.
- Raspberry Pi offers several advantages that contribute to its popularity and versatility in a wide range of applications

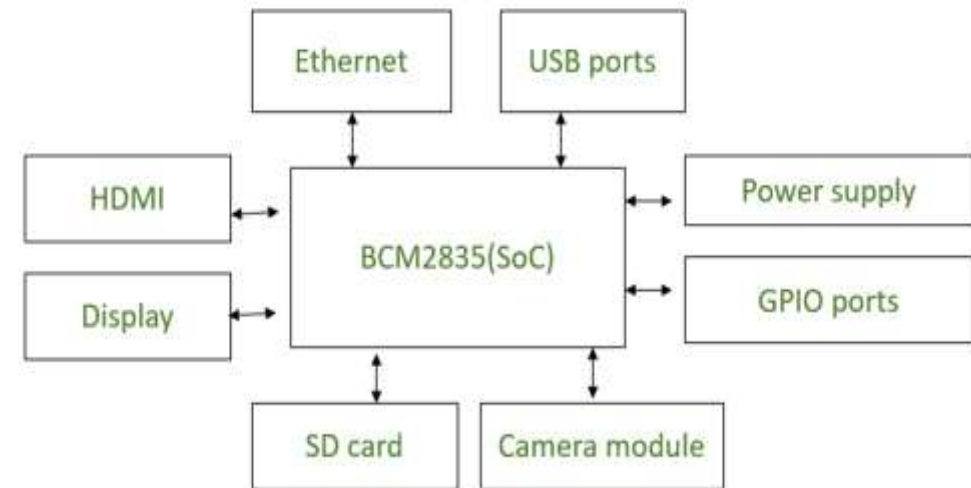
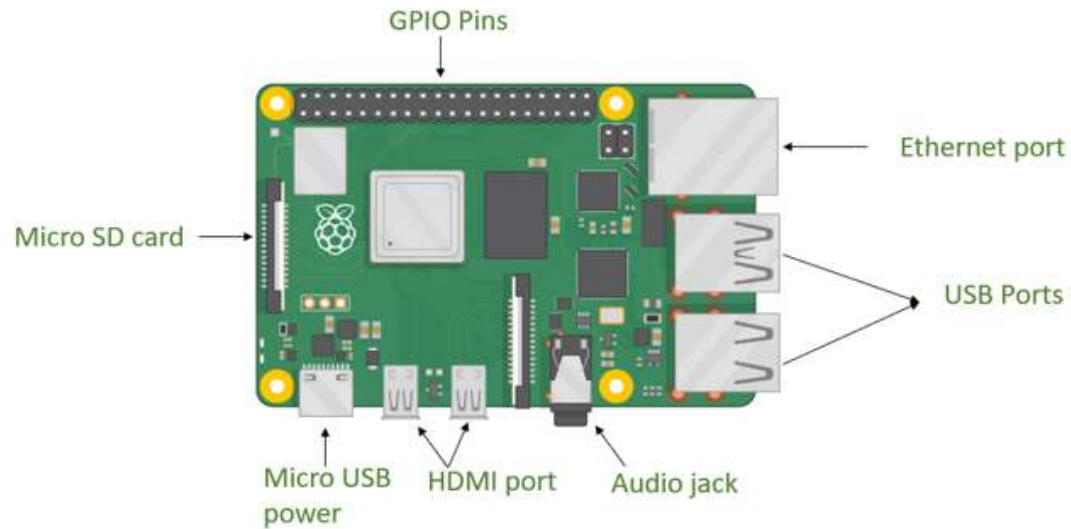
Key characteristics of Raspberry Pi



- **Single-Board Design:** Raspberry Pi is a credit card-sized computer with all major components, including the CPU, RAM, and I/O ports, integrated onto a single printed circuit board (PCB).
- **Broadcom SoC (System on a Chip):** Raspberry Pi boards are powered by Broadcom SoCs, which include a CPU, GPU, RAM, and other components in a compact package.
- **General-Purpose Input/Output (GPIO) Pins:** Raspberry Pi boards feature GPIO pins that allow users to interface with and control external devices such as sensors, LEDs, motors, and more.
- **Linux-Based Operating System Support:** Raspberry Pi commonly runs a variety of Linux-based operating systems, with Raspbian (now known as Raspberry Pi OS) being the official and recommended distribution. Other operating systems, including various Linux distributions and Windows IoT, are also compatible.

- **Affordability:** Raspberry Pi boards are designed to be affordable, making them accessible to a wide range of users, including students, hobbyists, and professionals.
- **Community and Ecosystem:** Raspberry Pi has a vibrant and active community of users and developers. The ecosystem includes a wide range of accessories, add-on boards (HATs), and software resources.
- **Educational and Hobbyist Use:** Raspberry Pi is widely used in educational settings to teach programming, electronics, and computer science. It is also a popular choice for hobbyist projects, DIY electronics, home automation, media centers, and more.
- **Various Models:** Over the years, several models of Raspberry Pi have been released, each with improvements in terms of hardware specifications and features. Some notable models include Raspberry Pi 4 Model B, Raspberry Pi 3 Model B+, Raspberry Pi Zero, and more.

Block Diagram of Raspberry Pi



Raspberry Pi mainly consists of the following blocks:

- **Processor:** Raspberry Pi uses Broadcom BCM2835 system on chip which is an ARM processor and Video core Graphics Processing Unit (GPU). It is the heart of the Raspberry Pi which controls the operations of all the connected devices and handles all the required computations.
- **HDMI:** High Definition Multimedia Interface is used for transmitting video or digital audio data to a computer monitor or to digital TV. This HDMI port helps Raspberry Pi to connect its signals to any digital device such as a monitor digital TV or display through an HDMI cable.
- **GPIO ports:** General Purpose Input Output ports are available on Raspberry Pi which allows the user to interface various I/P devices.
- **Audio output:** An audio connector is available for connecting audio output devices such as headphones and speakers.
- **USB ports:** This is a common port available for various peripherals such as a mouse, keyboard, or any other I/P device. With the help of a USB port, the system can be expanded by connecting more peripherals.
- **SD card:** The SD card slot is available on Raspberry Pi. An SD card with an operating system installed is required for booting the device.
- **Ethernet:** The ethernet connector allows access to the wired network, it is available only on the model B of Raspberry Pi.
- **Power supply:** A micro USB power connector is available onto which a 5V power supply can be connected.
- **Camera module:** Camera Serial Interface (CSI) connects the Broadcom processor to the Pi camera.
- **Display:** Display Serial Interface (DSI) is used for connecting LCD to Raspberry Pi using 15 15-pin ribbon cables. DSI provides a high-resolution display interface that is specifically used for sending video data.

Text Editors and Integrated Development Environments (IDEs) for Programming on Raspberry Pi



There are several Integrated Development Environments (IDEs) and text editors available for Raspberry Pi programming, catering to different preferences and programming languages.

1. Thonny:

- Thonny is a beginner-friendly IDE that comes pre-installed with many Raspberry Pi OS distributions. It is designed to make Python programming easy for beginners.
- Installation:
`sudo apt-get update`
`sudo apt-get install thonny`

2. IDLE:

- IDLE (Integrated Development and Learning Environment) is another Python-specific IDE that comes pre-installed with Python. It's lightweight and suitable for Python development.
- Installation:
`IDLE is included with the Python installation on Raspberry Pi.`

3. Geany:

- Geany is a lightweight text editor with IDE features. It supports multiple programming languages, including Python.
- Installation:
`sudo apt-get update`
`sudo apt-get install geany`

4. Visual Studio Code (VSCode):

- VSCode is a powerful and popular open-source code editor developed by Microsoft. It supports various programming languages, and there are extensions available for Raspberry Pi development.
- Installation: Install VSCode from the official website or use the following commands:
`sudo apt-get update`
`sudo apt-get install code`

5. Atom:

- Atom is a customizable and feature-rich text editor developed by GitHub. It supports various languages and has a large community that contributes packages and themes.
- Installation:
`You can download Atom from the official website.`

6. Emacs:

- Emacs is a highly customizable text editor that can function as an IDE. It has a steep learning curve but offers extensive features for those who become proficient.
- Installation:
`sudo apt-get update`
`sudo apt-get install emacs`

7. Nano:

- Nano is a simple and lightweight text editor for the command line. It's easy to use and suitable for quick edits or simple coding tasks.
- Installation:
Nano is often pre-installed on Raspberry Pi OS.

Supporting languages of raspberry pi



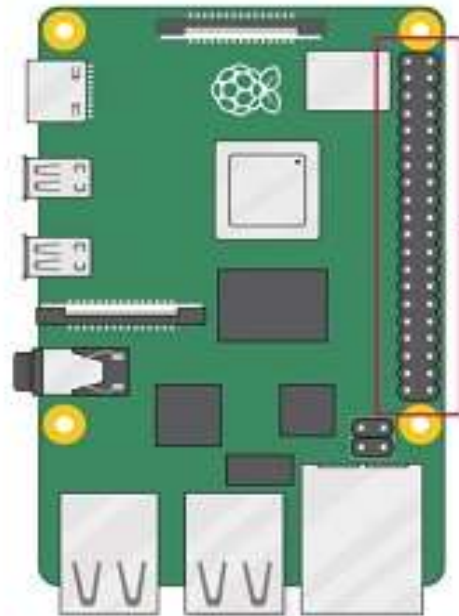
Raspberry Pi supports a variety of programming languages, offering flexibility for developers and users with different coding preferences. Here are some of the programming languages commonly used on Raspberry Pi:

- **Python:** Python is the most widely used and recommended programming language for Raspberry Pi. It comes pre-installed on Raspbian (now known as Raspberry Pi OS) and is well-supported with a large community. Python is suitable for a broad range of applications, from simple scripts to complex projects.
- **C:** C is a powerful and low-level programming language that can be used for Raspberry Pi development. It is commonly employed when performance optimization or hardware-level access is necessary. Developers can use tools like GCC for compiling C programs on Raspberry Pi.
- **C++:** C++ is an extension of the C programming language and can be used on Raspberry Pi for object-oriented programming. Like C, it is suitable for applications where performance is critical.

- **Java:** Java is a versatile, object-oriented language supported on Raspberry Pi. It is often used for applications where platform independence is essential. The Java Virtual Machine (JVM) is available for Raspberry Pi.
- **JavaScript:** JavaScript is commonly used for web development, and Node.js allows developers to run JavaScript on the Raspberry Pi. This is useful for building web-based applications and IoT projects.
- **Scratch:** Scratch is a visual programming language designed for beginners, and it comes pre-installed on Raspberry Pi OS. It allows users to create programs by stacking code blocks, making it an excellent tool for educational purposes.
- **Ruby:** Ruby is a dynamic, object-oriented programming language that is supported on Raspberry Pi. It is known for its simplicity and readability.
- **PHP:** PHP, a server-side scripting language commonly used in web development, can also be utilized on Raspberry Pi for server-related projects.
- **Shell Scripting (Bash):** Shell scripting, using languages like Bash, allows users to create scripts to automate tasks and manage the Raspberry Pi from the command line.

- **Go (Golang):** Go is a statically typed language developed by Google. It is suitable for building efficient and concurrent programs, and it is supported on Raspberry Pi.
- **Rust:** Rust is a systems programming language that emphasizes safety and performance. It is available on Raspberry Pi and can be used for applications where low-level control is necessary.
- **Perl:** Perl is a versatile scripting language that can be used for various tasks on Raspberry Pi.
- **Lisp, Haskell, and Other Languages:** Raspberry Pi's Linux-based environment allows for the installation and use of a wide range of programming languages, including Lisp, Haskell, and others.





















Raspberry Pi GPIO Pinout



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)

Raspberry Pi 3 Model B / Raspberry Pi 3 Model B+ / Raspberry Pi 4 Model B: These models typically follow the Broadcom numbering convention (BCM).

Raspberry Pi 3 GPIO Pinout

GPIO_GEN	Functions	GPIO	Pin	Pin	GPIO	Functions	GPIO_GEN
		3.3V	1		2	5V	
	SDA1 (I ² C)	GPIO2	3		4	5V	
	SCL1 (I ² C)	GPIO3	5		6	GND	
GCLK		GPIO4	7		8	GPIO14 TXD0 (UART)	
		GND	9		10	GPIO15 RXD0 (UART)	
GEN0		GPIO17	11		12	GPIO18 PWM0, CLK (PCM)	GEN1
GEN2		GPIO27	13		14	GND	
GEN3		GPIO22	15		16	GPIO23	GEN4
		3.3V	17		18	GPIO24	GEN5
	MOSI (SPI)	GPIO10	19		20	GND	
	MISO (SPI)	GPIO9	21		22	GPIO25	GEN6
	SCLK (SPI)	GPIO11	23		24	GPIO8 CE0 (SPI)	
		GND	25		26	GPIO7 CE1 (SPI)	
		ID_SD	27		28	ID_SC	
		GPIO5	29		30	GND	
		GPIO6	31		32	GPIO12 PWM0	
	PWM1	GPIO13	33		34	GND	
	FS (PCM), PWM1	GPIO19	35		36	GPIO16	
		GPIO26	37		38	GPIO20 DIN (PCM)	
		GND	39		40	GPIO21 DOUT (PCM)	

Raspberry Pi GPIO Setmodes

- The **GPIO.setmode()** function in the RPi.GPIO library for Raspberry Pi is used to set the numbering mode for the GPIO pins.
- It determines how you refer to the pins in your code.
- There are two available modes:
 1. GPIO.BCM
 2. GPIO.BOARD.
- These conventions refer to different ways of numbering the GPIO pins on the Raspberry Pi.

BCM (Broadcom) Mode:

- **Syntax:** `GPIO.setmode(GPIO.BCM)`
- In BCM mode, GPIO pins are identified by their Broadcom SOC channel numbers. These numbers correspond to the physical pin numbers on the Broadcom SOC (System on a Chip) used in Raspberry Pi. This mode provides a consistent way of identifying pins across different Raspberry Pi models.
- Example:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT) # BCM GPIO 18 (physical pin 12) configured as output
```
- BCM Mode is preferred for consistency across different Raspberry Pi models. Especially useful when migrating code between different Pi models, as the GPIO numbering remains the same.

BOARD Mode:

- **Syntax:** `GPIO.setmode(GPIO.BOARD)`
- In BOARD mode, GPIO pins are identified by their physical pin numbers on the Raspberry Pi header. This mode allows you to refer to the pins by their physical locations, making it easier for beginners to understand which pin is being used.
- Example:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(12, GPIO.OUT) # Physical pin 12 (BCM GPIO 18) configured as output
```
- BOARD Mode is Easier for beginners to understand, as it refers to the physical layout of the GPIO pins on the Raspberry Pi header. Board mode is suitable for projects where you are referencing pins based on their physical location on the Pi.

Operating Modes Of Raspberry Pi

In Raspberry Pi, the GPIO (General Purpose Input/Output) pins can be configured for different modes depending on how you want to use them. The main modes include Input, Output, and special modes like PWM (Pulse Width Modulation), I2C, SPI, etc.

1. Input Mode:

- **Mode Name:** **GPIO.IN** or **GPIO.INPUT**
- Configures the GPIO pin as an input, allowing you to read digital signals (HIGH or LOW) from external devices like buttons, sensors, etc.
- Example:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(GPIO_PIN, GPIO.IN)
input_state = GPIO.input(GPIO_PIN)
```

2. Output Mode:

- **Mode Name:** **GPIO.OUT** or **GPIO.OUTPUT**
- Configures the GPIO pin as an output, allowing you to send digital signals (HIGH or LOW) to control external devices like LEDs, relays, etc.
- Example:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(GPIO_PIN, GPIO.OUT)
GPIO.output(GPIO_PIN, GPIO.HIGH) # Turn ON
GPIO.output(GPIO_PIN, GPIO.LOW) # Turn OFF
```

3. PWM Mode:

- **Mode Name: GPIO.PWM**
- Configures the GPIO pin for Pulse Width Modulation, allowing you to control the brightness of LEDs, the speed of motors, etc.

- **Example:**

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(GPIO_PIN, GPIO.OUT)
pwm = GPIO.PWM(GPIO_PIN, 100) # 100 Hz frequency
pwm.start(50) # 50% duty cycle (0 to 100)
time.sleep(5) # Run for 5 seconds
pwm.stop()
```

4. I2C Mode:

- **Mode Name: GPIO.I2C**
- Configures the GPIO pins for I2C (Inter-Integrated Circuit) communication, allowing you to connect to I2C devices.

- **Example:**

```
import smbus
i2c = smbus.SMBus(1) # Use the appropriate bus number
```

5. SPI Mode:

- **Mode Name: GPIO.SPI**
- Configures the GPIO pins for SPI (Serial Peripheral Interface) communication, enabling communication with SPI devices.

- **Example:**

```
import spidev
spi = spidev.SpiDev()
spi.open(0, 0) # Bus 0, Device 0
# Perform SPI operations using the spidev library
```

6. Serial (UART) Mode:

- **Mode Name:** **GPIO.SERIAL** or **GPIO.UART**
- Configures the GPIO pins for serial communication (UART), allowing you to communicate with other serial devices.
- Example:

```
import serial
ser = serial.Serial("/dev/serial0", baudrate=9600, timeout=1)
# Perform serial communication using the serial library
```

7. Hardware PWM Mode (Raspberry Pi 2 and 3):

- **Mode Name:** **GPIO.HARD_PWM**
- Configures the GPIO pin for hardware PWM. It is similar to PWM but uses hardware support for better accuracy and performance.

8. Default Mode:

- **Mode Name:** **GPIO.IN** or **GPIO.INPUT**
- In the absence of explicit mode configuration, the GPIO pins are usually configured as inputs by default.
- Note:

Always import the RPi.GPIO module and set the GPIO mode using GPIO.setmode() before configuring pins.
Ensure proper cleanup using GPIO.cleanup() when your program exits to release the resources used by GPIO pins.

Libraries or modules of Raspberry Pi



The libraries or modules you need for Raspberry Pi programming depend on the specific tasks or projects you're working on. However, there are some common libraries that are frequently used in Raspberry Pi programming across various applications.

RPi.GPIO:

- Purpose: Provides access to the GPIO pins on the Raspberry Pi.
- Installation: Most Raspberry Pi OS distributions come with this library pre-installed. If not, you can install it using:

```
sudo apt-get update  
sudo apt-get install python3-rpi.gpio
```

smbus-cffi (for I2C):

- Purpose: Allows communication with I2C devices.
- Installation:

```
sudo apt-get update  
sudo apt-get install python3-smbus
```

spidev (for SPI):

- Purpose: Enables communication with SPI devices.
- Installation:
`sudo apt-get update`
`sudo apt-get install python3-spidev`

serial (for UART):

- Purpose: Provides serial communication capabilities.
- Installation:
`sudo apt-get update`
`sudo apt-get install python3-serial`

time:

- Purpose: Standard Python library for handling time-related functions.

datetime:

- Purpose: Standard Python library for working with dates and times.

subprocess:

- Purpose: Standard Python library for running shell commands.

os:

- Purpose: Standard Python library for interacting with the operating system.

requests:

- Purpose: Useful for making HTTP requests. Useful in projects involving internet connectivity.
- Installation:
`pip3 install requests`

picamera (for Camera Module):

- Purpose: Interface for the Raspberry Pi Camera Module.
- Installation:
`sudo apt-get update`
`sudo apt-get install python3-picamera`

math:

- Purpose: Standard Python library for mathematical operations.

pygame (for Multimedia Applications):

- Purpose: Useful for creating games or multimedia applications.
- Installation:
 - `sudo apt-get update`
 - `sudo apt-get install python3-pygame`

cv2 (OpenCV for Computer Vision):

- Purpose: Useful for computer vision tasks.
- Installation:
 - `pip3 install opencv-python`

Advantages of using Raspberry Pi

Raspberry Pi offers several advantages that contribute to its popularity and versatility in a wide range of applications:

- Affordability
- Compact Size
- Low Power Consumption
- Community Support
- Versatility
- Educational Tool
- GPIO (General Purpose Input/Output) Pins
- Expandability
- Open Source Software
- Rich Set of Accessories
- Robust Software Support
- Media Capabilities
- Internet of Things (IoT) Applications
- Rapid Prototyping

Disadvantages of Raspberry Pi

While Raspberry Pi is a versatile and popular platform, it does have some limitations and potential drawbacks. Here are some of the disadvantages of Raspberry Pi:

- Limited Processing Power
- Limited RAM
- Limited Graphics Performance
- Limited Storage Options
- Limited Connectivity
- No Real-Time Clock (RTC)
- Not Suitable for High-Performance Computing
- Limited Availability of GPU Documentation
- Power Requirements
- Limited Audio Output
- Learning Curve for Beginners
- Not Suitable for Real-Time Applications
- Single-Board Failure
- Not Designed for High-End Gaming

Real World Applications of Raspberry Pi



Raspberry Pi, a versatile and affordable single-board computer, has found applications across various domains due to its compact size, low cost, and low power consumption. Here are some real-world uses and applications of Raspberry Pi:

- Education
- Home Automation
- Media Center
- Network Monitoring
- Web Server
- Security Camera System
- Weather Station
- Gaming Console
- Robotics and IoT
- Desktop Computer
- Network Attached Storage (NAS)
- Educational Initiatives
- Telecommunication
- AI and Machine Learning