1

SOC AND RASPBERRY PI

Unit Structure

1.0 OBJECTIVES

After going through this unit, you will be able to:

- Understand the concept and structure of System on Chip
- Describe and illustrate several SoC products
- Explain and describe about the ARM 8 Architecture
- Introduce and prepare Raspberry Pi with hardware and installation

1.1 INTRODUCTION

System on Chip

A system on chip also known as SoC is an integrated circuit (IC) that integrates all the components into a single chip that is the complete system is present on the same single chip and hence its name. It has analog, digital,

mixed signal and other radio frequency function all present on a single chip substrate. Now-a-days, SoCs applications are more commonly found in electronics industry due to its low power consumption. Also, the greater use of SoCs is found in embedded system applications. SoCs generally consists of control unit (comprises of microprocessor, microcontroller, digital signal processor etc.); memory blocks (i.e ROM, RAM, Flash memory, EEPROM); timing units (oscillators, PLLs); other peripherals (it consists counter timers, real-time timers and power on reset generators); basic SoC interfaces (analog interfaces, external interfaces, voltage regulators and power management units).

Raspberry Pi

Raspberry Pi is a series of compact single-board computers developed by the Raspberry Pi Foundation in collaboration with Broadcom in the United Kingdom. These projects are generally inclined towards teaching and promoting basic computer science in schools and in developing countries. Due to its low cost, modularity and open design it finds wide application ranging from weather monitoring, robotics and many more.

Several generations have been released of Raspberry Pi's such as Raspberry Pi Model B (February 2012), followed by Model A, Model B+ (in 2014), Raspberry Pi 2(February 2015), Raspberry Pi Zero (November 2015), Raspberry Pi Zero W (On 28 February 2017), Raspberry Pi Zero WH (On 12 January 2018), Raspberry Pi 3 Model B (February 2016), Raspberry Pi 3 Model B+ (2018), Raspberry Pi 4 Model B (released in June 2019), Raspberry Pi 400 (November 2020) and Raspberry Pi Pico (in January 2021).

1.2 SYSTEM ON CHIP

1.2.1 What is System on Chip (SoC)?

SoC is an integrated circuit embedded on a small single platform coin-sized chip with a microprocessor / microcontroller along with all other electronic components integrated onto it. As the name suggests it is the entire system (complete circuit) on a single chip. SoC design usually incorporates central processing unit, input and output ports, memory, secondary storage devices, as well as analog input and output blocks, digital or analog signal processing system or a floating-point unit and peripheral interfaces such as I2C, SPI, UART, CAN, Timers, etc. It is capable of performing several tasks including signal processing, wireless communication, artificial intelligence and more.

1.2.1.2 Why SoC?

As technology is becoming more and more advanced, the main motivator and primary goal is to reduce energy waste, save up on spending costs, as well as reduce the space occupied by large systems. This essential requirement is possible by SoC as it size-down multichip design onto a single processor comparatively consuming less power than before. These chips helped us to developed portable devices that can be carried anywhere

and everywhere easily without compromising on the capability and functionality of the gadgets. SoCs have a plethora of practical uses that are both unlimited and priceless. These are associated with systems pertaining to the Internet of Things, embedded systems, smartphones, cameras, tablets, cars, wireless technologies etc.

The working of a SoC can be best described with an example of smartphones. When you use your cell phones you not only make and receive the calls; you LSO use it for browsing the internet, listening audio, watching videos, taking photos, playing games etc. Multiple components, such as a graphics card, internet support, wireless connections, GPS, and several more aspects, make all of these features feasible. All of these components can be merged into a single chip, which can then be shrunk down to fit in the palm of your hand and carried about. In recent years, SoC technology are used in small sized personal computers, laptops for reducing power consumption thus further improving the performance by using a single chip managing all the functionalities of the system.

1.2.1.3 SoC Building Blocks

SoC comprises of several building blocks as shown in the figure 1.1.

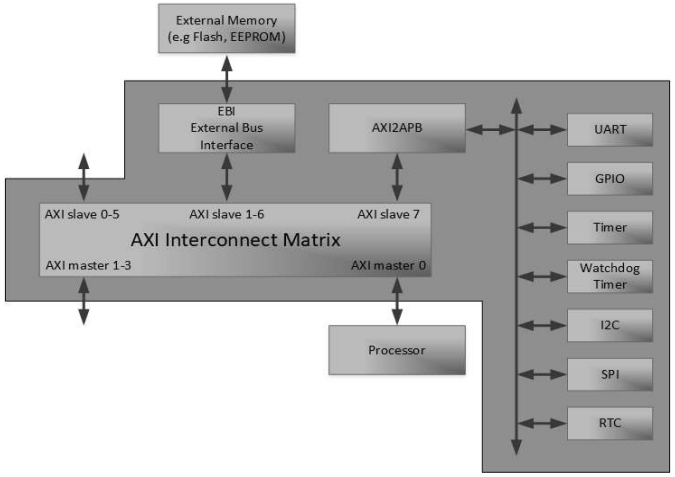


Figure 1.1 Building blocks of SoC

- Firstly, at its core, a system on chip must consist of a processor that will define all its functions. Generally, an SoC has multiple processor cores. A microcontroller, microprocessor, digital signal processor, or application specific instruction set processor can all be used.
- Secondly, for performing the computation the chip must have its memory. It could have memory in the form of RAM, ROM, EEPROM, or even flash memory.

- External interfaces are the next requirement, as they will allow it to conform to industry standard communication protocols such as USB, Ethernet, and HDMI. It can also make use of wireless technology and protocols such as WiFi and Bluetooth.
- For visualizing the interfaces, it must have a Graphical Processing Unit (GPU).
- Voltage regulators, phase lock loop control systems and oscillators, clocks and timers, analog to digital and digital to analog converters must also be included in SoC.
- For connecting all the individual blocks it must have an internal interface bus or a network.

1.2.1.4 Advantages of SoC

- Low power
- Low cost
- High reliability
- Small form factor
- High integration levels
- Fast operation
- Greater design
- Small size
- Low latency
- Better efficiency and performance
- Less time to market

1.2.1.5 Disadvantages of SoC

- More verification.
- Fabrication cost.
- Increased complexity.
- Time to market demands

1.2.1.6 SoC varieties

- **NVIDIA Tegra 3 is a graphics processor from NVIDIA**

  The NVIDIA Tegra 3 is a Tegra family SoC that is found in a variety of Android handsets. The Tegra 3 is used in some devices such as the Asus Eee Pad, HTC One X, and Google Nexus Tablet. This model includes a CPU with five cores. Each core is an ARM Cortex A9 chip, with the fifth core running at 500MHz and using a low-power silicon process.

- **Qualcomm's Snapdragon S4 processor**

  When it comes to Android smartphones and tablets, Qualcomm is crucial. It is powered by a processor that is similar to the ARM Cortex A15.

- **Samsung Exynos 4 Quad**

  The ARM architecture is used in this SoC. It has a quad-core ARM Cortex-A9 CPU and a 1.4GHz ARM Mali-400 MP4 quad-core GPU. This processor can handle a wide range of tasks, including 3D gaming, multitasking, and video recording and playback.

- **Intel Medfield**

  The Medfield SoCs from Intel are not based on the ARM architecture. These SoCs are built using x86 technology. Medfield SoCs can provide OEMs with a single-core processor running at 1.6-2GHz with a PowerVR SGX540 GPU.

- **OMAP 4 from Texas Instruments**

  The ARM Cortex A9 45nm architecture is used in the fourth generation of OMAPs. Motorola Atrix 2, Motorola Droid RAZR, LG Optimus 3D, and LG Optimus Max are some Android devices that employ this SoC.

1.2.1.7 SoC design challenges

The different SoC design challenges are given below:

1. Strategy for architecture
2. Strategy for test design
3. Strategy for validation
4. Backend Strategy and Synthesis
5. Integration Strategy
6. On chip Isolation

- **Strategy for Architecture**

  The type of processor we employ to create the SoC is an extremely significant issue to consider. Furthermore, the type of bus that must be used is a question of decision.

- **Strategy for Test Design**

  The majority of frequent physical problems are represented as faults in this approach. The essential circuitry incorporated in the SoC architecture assist in defect detection.

- **Strategy for validation**

  There are two primary concerns to consider while validating SoC designs. The first issue is that we need to double-check

the IP cores. The second issue is that we need to double-check the system's integration.

- **Backend Strategy and Synthesis**

  Many physical effects must be taken into account while planning the SoC synthesis and strategy. IR drop, cross talk, 3D noise, antenna effects, and EMI effects are all examples of effects. Chip planning, power planning, DFT planning, clock planning, timing, and area budgeting are all required early in the design process to address these difficulties.

- **Strategy for Integration**

  To create a smooth integration strategy, all of the above-mentioned data must be examined and combined.

- **On chip Isolation**

  Many factors must be considered in on-chip isolation, including the impact of process technology, grounding effects, guard rings, shielding, and on-chip decoupling.

1.2.1.8 SoC applications

Following are few applications of SoC

- **Market for mobile phones**

  The mobile industry, particularly in smartphones, is the most popular and basic application of SoC. Because smartphones are becoming slimmer and lighter as technology advances, the use of a SoC (whose size is shrinking at an alarming rate with each new generation) is the greatest fit for this change. Furthermore, high performance and low power consumption are two significant elements that influence smartphone performance, and SoC excels at both. As illustrated in this image, the A6 CPU was the first system on chip used in the deconstruction of the iPhone 5.

- **Embedded systems**

  In the modern world, almost every microcontroller and CPU has a SoC operating on top of it. Component coupling is tighter, resulting in greater reliability and performance.

  Embedded systems can be seen in Apple's smart watch. The apple S1 SoC is used in this smart watch.

  The SoC in the Samsung Galaxy Gear is based on the ARM Cortex M4 microcontroller. For example STM32F401B.

- **Personal computers**

  Another important application of the SoC is personal computers; many modern personal computers do not have a motherboard, instead relying on the SoC to provide great performance and minimal size.

1.2.1.9 Examples of SoCs

The majority of SoCs on the market today are ARM-based. Qualcomm's Snapdragon SoCs, Apple's A4, and Nvidia's Tegra series are some examples of smartphone SoCs. The Raspberry Pi 2 uses the Broadcom BCM2836 SoC. The Open Cores community has created a number of SoCs.

1.2.2 Structure of System on Chip: Design Flow

SoC design flow structure is as shown in the following figure 1.2

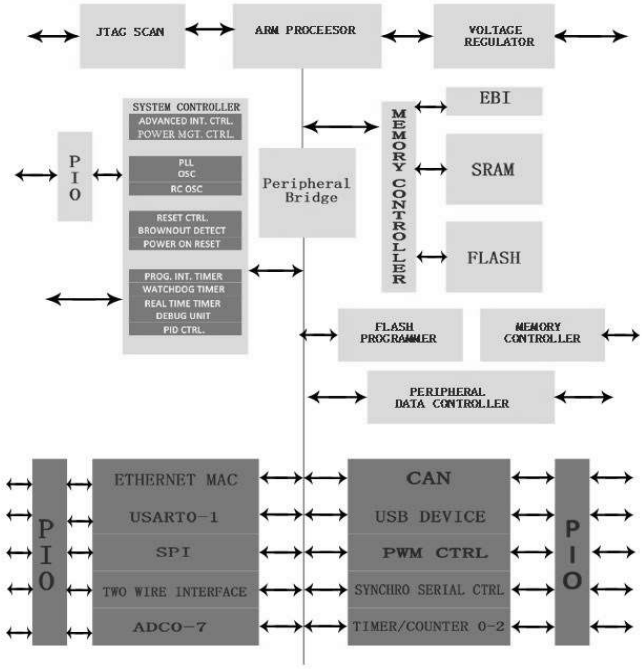


Figure 1.2 SoC design flow structure

The goal of the SoC design flow is to build the hardware and software of SoC designs. In general, the design pipeline of SoCs includes the following steps:

- Hardware and Software Modules: SoC hardware blocks are made up of pre-qualified hardware components and software modules that are combined in a software development environment. For the development of the modules, hardware description languages like as Verilog, VHDL, and SystemC are utilised.

- Functional Verification: Before being sent to the foundry, the SoCs are tested for logic accuracy.

- Verify hardware and software designs: Engineers have used FPGA, simulation acceleration, emulation, and other technologies to verify and debug the hardware and software of SoC designs.

- Place and Route: After the SoC has been debugged, the next step is to place and route the whole design onto the integrated circuit before it is sent to manufacture. Full custom, standard cell, and FPGA technologies are widely used in the fabrication process.

1.3 SOC PRODUCTS

This section describes various SoC products such as FPGA, GPU, APU and compute unit in detail.

1.3.1 FPGA

Field Programmable Gate Array is the abbreviation for FPGA. It's an integrated circuit that may be configured by a user after it's been created for a specific purpose. Adaptive logic modules (ALMs) and logic elements (LEs) are coupled via programmable interconnects in modern FPGAs. These blocks combine to form a physical array of logic gates that can be configured to execute a variety of tasks. This distinguishes them from other types of microcontrollers or Central Processing Units (CPUs), whose configuration is fixed and cannot be changed by the manufacturer. FPGA overview is shown in the figure 1.3 below

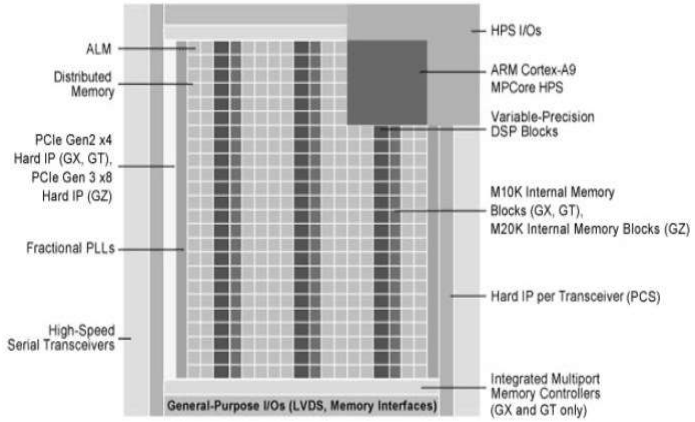


Figure 1.3 An overview of FPGA

The early programmable circuits were quite basic, consisting solely of logic gates. This was sufficient to execute a variety of logical functions with zeros and ones as inputs and outputs. Programmable circuits became increasingly and more powerful over time. You programme logic cells that can act as registers, adders, multiplexers, or lookup tables in programmable circuits.

While the circuit is running, the way the cells work and the structure of the cells can both be altered. A circuit can be reprogrammed to fulfil several roles, including those of an ARM processor, a network interface card, or a video encoder, to mention a few. Figure 1.4 shows the adaptive logic module of FPGA.

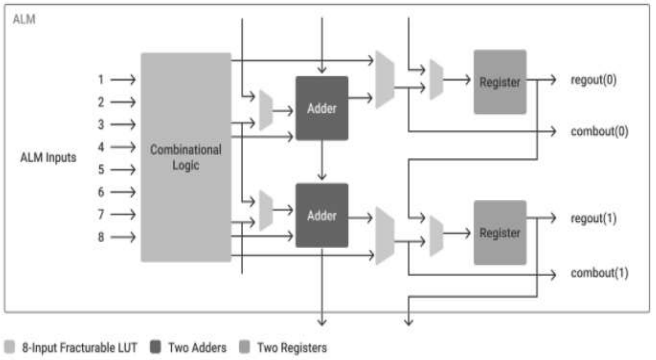


Figure 1.4 **Adaptive Logic Module of an Altera/Intel FPGA**

1.3.1.1 Working of FPGA

The FPGAs are made up of logical modules that are linked together through routing channels. Each module is made up of a programmable lookup table that is used to manipulate the elements that make up each cell and perform logical functions on those elements. Each cell, in addition to the lookup table, contains cascaded adders that allow addition to be performed. Subtraction can also be accomplished by altering the input's logical states. There are additionally registers (logical elements used to conduct the most basic memory functions) and multiplexers in addition to these (switching elements).

Depending on the manufacturer model, FPGAs can also incorporate static and dynamic on-chip memory. CPU cores, memory controllers, USB controllers, and network cards are among the ready-to-use components found in FPGAs. There is no need to include these components in the FPGA framework because they are so widely used. Instead, you can use a component that has already been made.

1.3.1.2 What can you do using FPGA programming?

FPGAs are primarily employed in the development of **application-specific integrated circuits** (ASICs). To begin, you must create the circuit's architecture. The prototype is then built and tested using an FPGA. Errors are reversible. Once the prototype has proven to be functional, an ASIC project based on the FPGA design is produced and fabricated. Because manufacturing an integrated circuit is a complex and time-consuming procedure, this helps you to save time. It also saves money because a single FPGA can handle multiple versions of the same project. In this regard, it's

worth noting that modern **Tensor Processing Units** (TPUs), often known as crypto currency miners, were first designed as FPGAs and only then built.

In real-time systems, when response time is critical, FPGAs are also used. Response time is not fixed in ordinary CPUs, thus you never know when you'll get a response once the initial signal comes. Real-time operating systems are used to reduce it or keep it within a certain range. Even yet, in cases requiring a quick response time (sub milliseconds), this falls short. To solve this problem, the requested algorithm must be implemented in FPGA using combinational or sequential logic to guarantee a consistent response time of less than milliseconds. Once ready, a real-time system designed in FPGA can be changed and pushed into production. This method will result in a considerably speedier and more energy-efficient integrated circuit.

FPGAs are also employed in applications where the hardware configuration is subject to change and a circuit that can adapt to these changes is required. FPGA becomes an obvious alternative if your hardware supplier's move and the new hardware does not have the needed interface.

1.3.1.3 How to program FPGAs?

It's possible that the term "FPGA programming" is a misleading. After all, unlike CPUs and GPUs, there is no true program to run sequentially. FPGA programming is designing a hardware architecture that can run a given algorithm and describing it using a hardware description language (HDL). As a result, unlike normal programs executed by CPUs or GPUs, the building blocks of this algorithm will not be a memory register and a set of operations to be done. Low-level elements such as logic gates, adders, registers, and multiplexers will make up a "FPGA program."

This provides you with a lot of versatility. If your data type is 20 bits, for example, you can only use 20-bit instructions to conduct operations. In the realm of CPUs, there are only manufacturer-set registers and instructions that cannot be modified. You can change to the data type in FPGAs, on the other hand, because you design the hardware architecture yourself.

You can also use general-purpose CPUs to perform processes that are either complex or time-consuming. CPUs, for example, conduct block cyphers and cryptographic tasks in many cycles, taking substantially longer than FPGAs.

1.3.1.4 Languages used in FPGA programming

Specific languages, like as VHDL or Verilog, are used to programme FPGAs. The syntax of VHDL is more close to Pascal than C, allowing for programming that is distinct from that of traditional high-level languages. Verilog, on the other hand, is similar to C, making it more intuitive and user-friendly for those with no prior familiarity with low-level programming.

VHDL is an obsolete language with a number of drawbacks, one of which being the difficulty in determining whether the architecture works as planned. Python is used to generate chunks of the code in various applications to make our lives easier. Of course, everything could be written in VHDL, but Python is more user-friendly.

The HDL simulator is the most important tool for designing hardware. It enables you to simulate how the architecture functions using sample input data. As a result, you can see how the data flows. The HDL simulator is particularly important since compiling a given hardware description into an FPGA board and programming the board itself, even for a basic programme, might take a long time. You can use the simulator to extensively test the algorithm you wish to put on an FPGA board.

1.3.1.5 FPGA Architecture

A typical FPGA design (Figure 1.5) is made up of thousands of basic elements called configurable logic blocks (CLBs), which are connected by a system of programmable interconnects called a fabric that routes signals between CLBs. The FPGA and external devices are connected using input/output (I/O) blocks.

The CLB is also known as a logic block (LB), a logic element (LE), or a logic cell (LC), depending on the manufacturer.

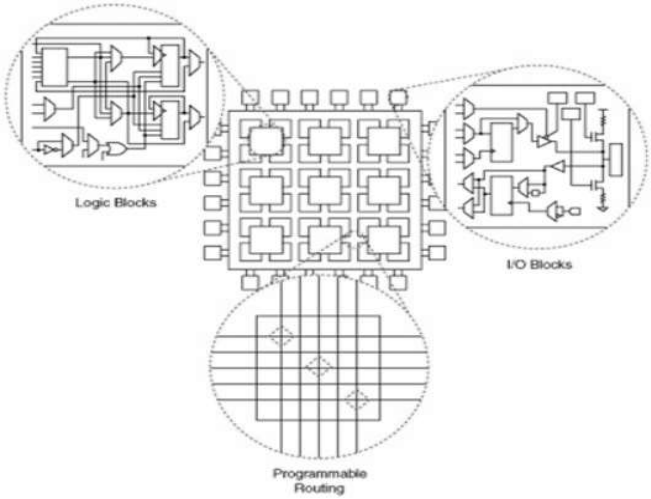


Figure 1.5: The fundamental FPGA architecture

A CLB is made up of numerous logic blocks (see Figure 1.6). An FPGA's lookup table (LUT) is a distinguishing feature. LUTs with four to six input bits are commonly utilised, and they hold a predefined list of logic outputs for any combination of inputs. Multiplexers (mux), full adders (FAs), and flip-flops are all commonly used logic functions.

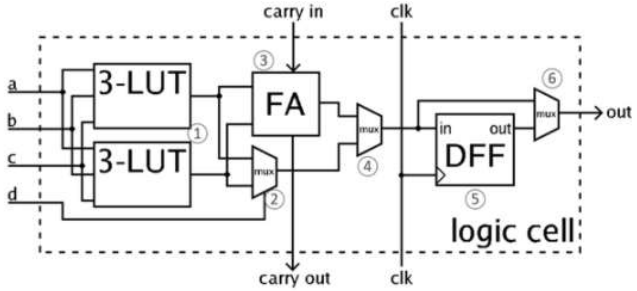


Figure 1.6: *A simplified CLB: The four-input LUT is formed from two three-input units*

The number and location of components in the CLB vary depending on the device; in the simplified example in Figure 1.6, during FPGA programming, two three-input LUTs (1), an FA (3), and a D-type flip-flop (5), as well as a standard mux (2) and two muxes, (4) and (6), are configured.

There are two modes of operation for this reduced CLB. The LUTs are merged with Mux 2 to produce a four-input LUT in normal mode, and the LUT outputs are provided as inputs to the FA along with a carry input from another CLB in arithmetic mode. Mux 4 switches between the FA and LUT outputs. The D flip-flop in Mux 6 decides whether the operation is asynchronous or synchronised to the FPGA clock.

CLBs in current-generation FPGAs can combine for more complicated operations such as multipliers, registers, counters, and even digital signal processing (DSP) capabilities; CLBs can combine for more sophisticated operations such as multipliers, registers, counters, and even DSP functions.

1.3.1.6 FPGA Applications

FPGAs are well-suited to a variety of markets because to their programmability. Xilinx, as the industry leader, provides wide range of solutions that include FPGA devices, powerful software, and configurable, ready-to-use IP cores for markets and applications including:

- Aerospace and Defense - FPGAs for image processing, waveform synthesis, and partial reconfiguration for SDRs that are radiation-tolerant.
- ASIC Prototyping - ASIC prototyping with FPGAs allows for quick and accurate SoC system modelling and embedded software verification.

- Automobiles - Silicon and IP solutions for gateway and driving assistance systems, as well as comfort, convenience, and in-vehicle infotainment.
- Broadcast & Pro AV - With Broadcast Targeted Design Platforms and solutions for high-end professional broadcast systems, you can adapt to changing requirements faster and extend product life cycles.
- Consumer Electronics - Affordably priced technologies that enable next-generation, full-featured consumer applications such convergent handsets, digital flat panel display, information appliances, home networking, and residential set-top boxes.
- Data Center - Designed for high-bandwidth, low-latency servers, networking, and storage applications to boost cloud deployment value.
- Network Attached Storage (NAS), Storage Area Network (SAN), servers, and storage appliances solutions for high-performance computing and data storage.
- Industrial - Xilinx FPGAs and targeted design platforms for Industrial, Scientific, and Medical (ISM) enable higher degrees of flexibility, faster time-to-market, and lower overall non-recurring engineering costs for a wide range of applications including industrial imaging and surveillance, industrial automation, and medical imaging equipment (NRE).
- Wired Communications - Wired Communications' end-to-end solutions cover reprogrammable networking linecard packet processing, framer/MAC, serial backplanes, and more.
- Wireless Communications - RF, baseband, connection, transport, and networking solutions for wireless equipment, including WCDMA, HSDPA, WiMAX, and other standards.

1.3.1.7 Artificial Intelligence: The next frontier for FPGAs

FPGAs are now gaining traction in another field: artificial intelligence (AI) using deep neural networks (DNNs) (AI). It needs a lot of processing resources to run DNN inference models. GPUs are frequently used to speed up inference processing, but in some circumstances, high-performance FPGAs may surpass GPUs when it comes to evaluating massive amounts of data for machine learning.

Microsoft is already utilizing Intel FPGA flexibility for AI acceleration. Customers can use Microsoft Azure cloud services to access Intel Stratix FPGAs as part of Project Brainwave. These FPGAs have been configured specifically for running deep learning models on cloud servers with these FPGAs. Developers can use the Microsoft service to tap into the capabilities of FPGA chips without having to buy or configure additional gear or

software. Developers can instead use open-source tools like the Microsoft Cognitive Toolkit or the Tensor Flow AI programming framework.

1.3.1.8 Benefits by using FPGAs

i] Flexibility

- Every time the device is powered up, the FPGA functionality can vary. So, if a design engineer wants to make a modification, all they have to do is download a new configuration file into the device and try it out.
- Frequently, updates to the FPGA can be made without the need for costly PC board replacements.
- ASSPs and ASICs have fixed hardware functionality that cannot be modified without a significant financial and time investment.

ii] Acceleration

- Improve your system's performance and/or get items to market faster.
- In comparison to ASICs, FPGAs are available “off the shelf” (which require manufacturing cycles taking many months).
- OEMs can deploy systems as soon as the design is operational and proven because to FPGA flexibility.
- FPGAs provide CPUs with off-load and acceleration features, allowing the entire system to run faster.

iii] Integration

- On-die CPUs, transceiver I/Os at 28 Gbps (or faster), RAM blocks, DSP engines, and other features are available in today's FPGAs. More functionality in the FPGA mean fewer devices on the circuit board, which improves reliability by decreasing device failures.

iv] Total Cost of Ownership (TCO)

- While ASICs are less expensive per unit than FPGAs, they require a non-recurring expense (NRE), expensive software tools, specialist design teams, and extended production cycles to produce.
- Long lifecycles (15 years or more) are supported by Intel FPGAs, eliminating the cost of rebuilding and requalifying OEM production equipment whenever one of the electronic components on-board becomes obsolete (EOL).

- FPGAs lower risk by allowing prototype systems to be shipped to clients for field testing while still allowing for quick changes before ramping up to volume production.

1.3.2 GPU

The CPU (central processing unit) is referred to as a computer's brain and GPU its soul. GPUs, on the other hand, have broken out from the limits of the PC over the last decade.

GPUs have sparked a global AI craze. They've evolved into an important component of current supercomputing. They've been incorporated into new hyper scale data centers that are expansive. They've evolved into accelerators, speeding up everything from cryptography to networking to artificial intelligence.

They also continue to drive gaming and professional graphics advancements in workstations, desktop PCs, and a new generation of laptops.

1.3.2.1 What is a GPU?

GPUs (graphics processing units) are now much more than the PCs in which they first appeared, but they are still based on a much older concept known as parallel computing. GPUs are extremely powerful because of this.

CPUs, to be sure, are still necessary. CPUs are quick and versatile, and they race through a series of tasks that require a lot of interaction. For example, retrieving data from a hard drive in response to a user's keystrokes.

GPUs, on the other hand, divide complex problems into thousands or millions of smaller tasks and solve them all at once. This makes them ideal for graphics, where textures, lighting, and shape rendering must all be done at the same time to keep images moving across the screen.

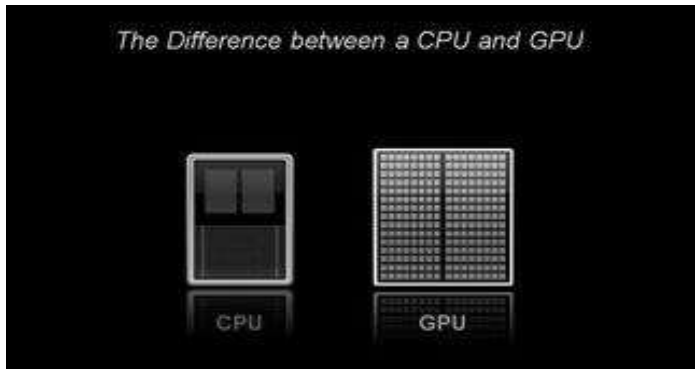


Figure 1.7: Difference between CPU and GPU

1.3.2.2 Difference between CPU and GPU

CPU	GPU
Full Form: Central Processing Unit	Full Form: Graphics Processing Unit
Few cores	Many cores
Low latency	High throughput
Good for serial processing	Good for parallel processing
Can do a limited operations at once	Can do thousands of operations at once

Because the CPU only has a few cores and a lot of cache memory, it can only process a few software threads at a time, whereas the GPU consisting of hundreds of cores thus handling thousands threads at once.

Parallel computing, which was once an esoteric technology, is now available thanks to GPUs. It's a technology with an illustrious pedigree that includes names like Seymor Cray, the father of supercomputing. GPUs put this idea to work in the desktops and gaming consoles of over a billion gamers, rather than taking the form of hulking supercomputers.

1.3.2.3 What does a GPU do?

The graphics processing unit (GPU) has emerged as one of the most important types of computing technology for both personal and business computing. The GPU, which was created for parallel processing, is used in a variety of applications, including graphics and video rendering. GPUs are becoming more popular for use in creative production and AI, despite being best known for their gaming capabilities.

GPUs were created with the intention of speeding up the rendering of 3D graphics. They improved their capabilities by becoming more flexible and programmable over time. With advanced lighting and shadowing techniques, graphics programmers were able to create more interesting visual effects and realistic scenes. Others began to use GPUs to dramatically speed up additional workloads in high-performance computing (HPC), deep learning, and other areas.

1.3.2.4 Unified GPU Architecture

A parallel array of multiple programmable processors encourages unified GPU architectures. Unlike earlier GPUs, which had distinct processors specialized to each processing type, they combine vertex, geometry, and pixel shader processing and parallel computing on the same processors. For texture filtering, rasterization, raster operations, anti-aliasing, compression, decompression, display, video decoding, and high-definition video

processing, the programmable processor array is intimately linked with fixed function processors.

Many core GPUs have a distinct architectural design point than multicore CPUs, focusing on efficiently running many parallel threads on many processor cores. More of the per-chip transistor budget is given to computation, and less to on-chip caches and overhead, by using many simpler cores and optimizing for data-parallel behavior among groups of threads.

The logical pipeline, which consists of discrete independent programmable stages, is mapped onto a physical dispersed array of processors in shown in the figure 1.8.

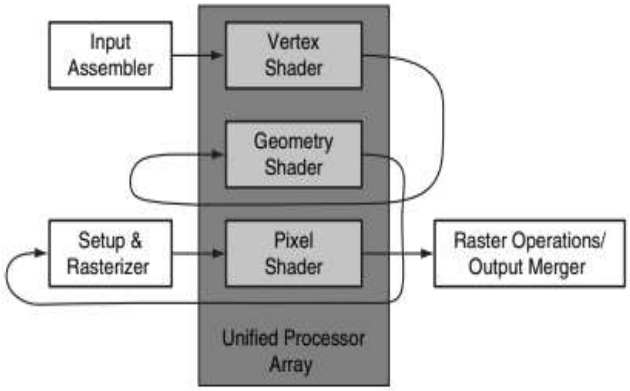


Figure 1.8 Logical pipeline mapped to physical processors. On the array of unified processors, the programmable shader stages run, and the logical graphics pipeline dataflow recirculates via the processors.

Processor Array

Many processor cores are grouped into multithreaded multiprocessors in a unified GPU processor array. A GPU with 112 streaming processor (SP) cores structured as 14 multithreaded streaming multiprocessors (SM) is shown in Figure 1.9. Each SP core is extremely multithreaded, with 96 concurrent threads and their hardware states to manage. An interconnection network connects the CPUs to four 64-bit-wide DRAM partitions. Each SM is equipped with eight SP cores, two SFUs, instruction and constant caches, a multithreaded instruction unit, and shared memory. This is the NVIDIA GeForce 8800's implementation of the Tesla architecture. Traditional graphics applications such as vertex, geometry, and pixel shading run on the unified SMs and SP cores, while computation programs operate on the same processors.

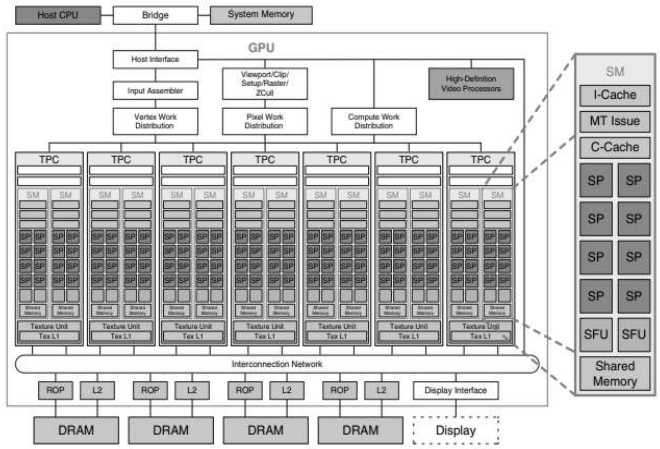


Figure 1.9 Basic unified GPU architecture

By adjusting the number of multiprocessors and memory partitions, the processor array design can be scaled to smaller and bigger GPU systems. Seven clusters of two SMs share a texture unit and a texture L1 cache in Figure 1.9. Given a set of coordinates into a texture map, the texture unit returns filtered results to the SM. Because the filter regions of support for subsequent texture requests frequently overlap, a tiny streaming L1 texture cache can help minimize the number of queries to the memory system. A GPU-wide interconnection network connects the processor array to raster operation (ROP) processors, L2 texture caches, external DRAM storage, and system memory. The number of processors and memories in a GPU system can be scaled to meet the needs of various performance and market sectors.

1.3.2.5 Applications of GPU

GPUs were largely employed to accelerate real-time 3D graphics applications, such as gaming, two decades ago. Computer experts believed that GPUs had the potential to address some of the world's most complex computing challenges as the twenty-first century began.

The general-purpose GPU era began as a result of this insight. Graphics technology is now being used to solve an ever-widening range of challenges. GPUs are more programmable than they've ever been, allowing them to speed up a wide range of applications beyond graphics rendering.

- GPUs for gaming: With hyper realistic graphics and enormous, sophisticated in-game worlds, video games have become more computationally intensive. With the rise of virtual reality games and improved display technologies such as 4K panels and high refresh rates, graphics processing demands are rapidly increasing. GPUs can render visuals in both 2D and 3D modes. Games can be played at

greater resolutions, quicker frame rates, or both with superior visual performance.

- GPUs for Video Editing and Content Creation: Long rendering times have plagued video editors, graphic designers, and other creative professions for years, sapping computing resources and stifling creative flow. GPUs' parallel processing now makes rendering video and graphics in higher-definition formats faster and easier.

- GPUs for Machine Learning: AI and machine learning are two of the most interesting applications for GPU technology. GPUs can give amazing acceleration in workloads that take use of GPUs' highly parallel nature, such as image recognition, because they have such a large amount of processing capability. Many of today's deep learning solutions rely on GPUs and CPUs working together.

1.3.3 APU

An APU is a 64-bit microprocessor that combines the processing capabilities of a CPU (Central Processing Unit) and a GPU (Graphics Processing Unit) onto a single chip. While APU may sound like any other computer processor, it is only used by AMD as the brand name for the CPU/GPU combo chips they produce. To understand what an APU is, it's helpful to know a little about the two CPUs it combines.

The CPU, also known as the "brain" of the computer, is the main processing unit that receives and executes instructions from software or applications. It also transmits instructions to other elements of the system, instructing them on what they should do. It is the most important component of a computer system; without it, the computer would be rendered useless.

The GPU performs comparable tasks to the CPU, but it only handles graphics-related data and generates graphical output. A computer without a GPU is blind, with no video output, just as a computer without a CPU is.

The CPU and GPU are two independent components in most systems. Except that the data transfer rate will improve if the two processors are closer to each other, there isn't much of a problem with this. Furthermore, having these two units running at the same time results in higher power usage, which AMD is well aware of. They released their first high-performance and energy-efficient processor, the APU, in 2011, which merged the benefits of the CPU and GPU into a single chip.

1.3.3.1 Evolution of APU

AMD has been developing structured and efficient architecture for their CPUs and GPUs as a major manufacturer of computer hardware. Their APUs is usually a combination of their existing CPU and GPU designs. The resulting processor outperforms the typical CPU and GPU when used together. It was formerly known as the "Fusion" before being renamed the "APU." The term was eventually changed to APU because to a trademark infringement concern.

AMD makes two kinds of APUs: one for high-performance devices and the other for low-power devices. Llano was the codename for the first generation APU for high-performance devices, which contained K10 CPU cores and a Radeon HD 6000-series GPU. Similarly, the first APU for low-power devices, codenamed Brazos, had the Bobcat microarchitecture and a Radeon HD 6000-series GPU. Trinity, AMD's second generation of high-performance APU, and Brazos 2.0, AMD's second generation of low-power APU, were released in 2012. As AMD's CPU and GPU architecture improved, the APU improved as well, with performance at the forefront of each improvement. Following versions used the most up-to-date architecture available at the time, and each iteration saw significant advancements over the previous one. Apart from performance, AMD has increased the upgradability of its products. Previously, future CPU upgrades were not conceivable, but starting with the APU Ryzen series, this was no longer the case. Renoir, the 2020 release, is built on the Zen 2 core architecture and features Vega 8 graphics.

APUs are still evolving today, and with AMD's newest and more powerful architectures, the next generation of APUs is on the way.

1.3.3.2 Benefits over CPU and GPU

The game-changing technology of the APU is a key advancement in the computing industry, with various advantages over the CPU + GPU configuration.

Improved performance: The data transfer rate was greatly enhanced by combining the CPU and GPU in the same chip because they now share the same bus and resources. OpenCL (Open Computer Language), a standard interface for parallel computing that makes use of the computing power supplied by GPUs, is also supported by APUs. Tasks that demand the high processing power of a CPU and the fast image processing of a GPU can benefit from the performance of an APU's multi-core CPU and GPU.

Power-efficient: Not only do combining two chips reduce space, but it also saves electricity. Apart from enhancing the APU's performance, AMD is constantly working to reduce the chip's power consumption, despite the fact that it is already low. Low Thermal Design Power is a feature of newer models (TDP). The Ryzen Embedded 1102G, for example, has the lowest TDP of only 6W.

Cost-effective: The cost benefit of AMD's APU over a CPU and GPU combination is arguably the most significant. Buying an APU is often less expensive than buying a CPU and GPU individually, with prices ranging from $100 to $400 depending on the specifications. Though the higher-end components are more expensive, they are still less expensive than the cost of a CPU and GPU with the same level of performance. This applies to future upgrades as well. Due to AMD's permissive attitude toward APU upgradability and compatibility, consumers can save a lot of money by replacing just one processor rather than both.

1.3.3.3 What sectors can benefit from APUs?

Accelerated Processing Units have been used in a variety of areas, including:

- **Software Development**

  Software developers can employ APUs to create heterogeneous computing architectures that blend CPU and GPU technology. This combination allows them to work on projects that require a high level of speed and processing power. Today's APUs also supports Open Computing Language (OpenCL) pictures, which helps. OpenCL is a standard interface for task- and data-based parallelism in parallel computing. The majority of activities necessitate a lot of computer power (from CPUs) and quick picture processing (a GPU feature). However, CPUs and GPUs rarely process data at the same time. The process is sped up by APUs, which combine both capabilities and allow parallel processing.

  APUs are also less expensive than buying a CPU and GPU, making them perfect for software developers who don't need a lot of processing power.

- **Visual content creation**

  The majority of today's digital material is mainly visual. Digital content creators may quickly create high-quality videos that elevate the user experience with an APU-powered computer.

  Advanced Micro Devices (AMD), the company that invented APUs, allows content makers to employ built-in universal video decoders (UVDs) to enhance video content so that it may be displayed on a big screen without losing quality.

  APUs allow content creators to clean up photos and movies in addition to offering high-quality displays, simplifying and streamlining the content creation process.

- **Gaming**

  APUs for gaming are also handy for gamers who want to build their own computers. These enable them to take advantage of improved and quicker graphics processing, enhancing their gaming experience without breaking the bank.

1.3.3.4 Is it a Better Processor?

APUs have been found in a variety of devices, including desktops, laptops, servers, mobile phones, and gaming consoles. For a decade, businesses and consumers have supported this heterogeneous chip. Can it, however, truly replace the CPU and GPU? In the end, it would be determined by the wants and demands of the user.

Consumers, PC builders, and budget gamers can take use of APU's advantages. The majority of APUs are capable of delivering adequate performance. In fact, it has the ability to exceed mid-range CPUs and GPUs. It's an excellent alternative for customers who don't want intense graphics or the maximum available CPU speed. It will also work well with ordinary PCs at home and in the business. AMD continues to create sophisticated APUs, with latest models capable of handling graphics-intensive tasks.

When it comes to intensive gaming, though, an APU will not suffice. It's still unable to match the graphical experience provided by high-end discrete graphics cards. An APU, on the other hand, would be an excellent choice for low-budget, entry-level PC building and gaming.

Although an APU cannot totally replace the CPU and GPU, it is a suitable high-performance, low-power option in many circumstances. As AMD's designs improve and new technologies emerge, it wouldn't be surprising if future generations of the APU can completely replace both the CPU and the GPU.

1.3.4 Compute Units

Compute units are comparable to host groups, but they have the added feature of granularity, allowing cluster-wide structures that mimic network architecture to be constructed. Task scheduling that considers processing unit resource needs optimizes job placement based on the underlying system architecture, eliminating communications bottlenecks. When conducting communication-intensive parallel operations across multiple hosts, compute units are extremely handy. Compute units represent the topology of a cluster network for workloads that require a lot of communication between processes. Computing units, for example, can help reduce network latency and take use of fast interconnects by putting all job operations in the same rack, rather than making several network hops.

Availability of resources Strings can be used to indicate compute unit requirements such as performing a job solely (excl), evenly distributing a job across many compute units (balancing), or selecting compute units depending on other criteria.

A computation unit is made up of 64 shader processors and four TMUs. The compute unit is independent from the render output units, yet it feeds into them (ROPs). A CU Scheduler, a Branch & Message Unit, four SIMD Vector Units (each 16-lane wide), four 64KiB VGPR files, one scalar unit, a 4 KiB GPR file, a 64 KiB local data share, four Texture Filter Units, sixteen Texture Fetch Load/Store Units, and a 16 KiB L1 Cache are all contained in each Compute Unit. A 16KB L1 instruction cache and a 32KB L1 data cache are shared by four computing units and are both read-only. A SIMD-VU can process 16 items at a time (per cycle), but an SU can only process one element at a time (per cycle). In addition, the SU performs other tasks like as branching.

Every SIMD-VU has its own private memory where its registers are stored. There are two sorts of registers: scalar registers (s0, s1, etc.), which contain

four bytes of data, and vector registers (v0, v1, etc.), which hold 64 bytes of data. Every operation on the 64 numbers in the vector registers is performed in simultaneously. When you work with them, you're truly working with 64 inputs. For instance, suppose you're working on 64 separate pixels at the same time (for each of them the inputs is slightly different, and thus you get slightly different color at the end). There are 512 scalar registers and 256 vector registers in each SIMD-VU.

1.3.4.1 Compute unit configuration

Compute unit configuration must meet the following requirements to ensure consistency:

- Hosts and host groups are only found in the highest granularity compute unit type.
- At most one compute unit of the finest granularity's membership list contains hosts.
- The same type of compute units (or hosts) is members of all compute units of the same type.

1.3.4.2 Where to use compute units?

The following parameters in LSF configuration files can be defined using LSF compute units:

- The compute unit type allowed for the queue is EXCLUSIVE in lsb.queues.
- The hosts on which jobs from this queue can be run are listed in lsb.queues as HOSTS.
- RES REQ in lsb.queues is used to track resource requirements for queue compute units.
- For application profile compute unit resource needs, see RES REQ in lsb.applications.

1.3.4.3 Different configurations of compute unit

Customers can select from the following compute unit configurations based on their requirements as shown in the following Table I:

Table I: Compute unit configurations

Compute Unit	Configuration	Size Parameter Value
Lite edition	1 CPU Core; 3072 MB Main memory	lite
Professional edition	2 CPU Cores; 4096 MB Main memory	pro

Compute Unit	Configuration	Size Parameter Value
Premium edition	4 CPU Cores; 8192 MB Main memory	prem
Premium Plus edition	8 CPU Cores; 16384 MB Main memory	prem-plus

1.4 ARM 8 ARCHITECTURE

1.4.1 SoC on ARM 8

A RISC processor is what the ARM processor is. Around 1980, the RISC was born out of processor development programs at Stanford and Berkeley universities. Between 1983 and 1985, Acorn Computers Limited in Cambridge, England, developed the ARM processor. It was the first commercially available RISC CPU, and it differs significantly from subsequent RISC architectures.

ARM Limited was founded in 1990 as a distinct company with the sole purpose of expanding the use of ARM technology. Since then, the ARM has been licensed to a number of semiconductor manufacturers throughout the world. It has established itself as an industry leader in low-power, low-cost embedded applications. Without the support of hardware and software development tools, no processor is very valuable. An instruction set emulator for hardware modeling and software testing and benchmarking, an assembler, C and C++ compilers, a linker, and a symbolic debugger are all part of the ARM toolset.

The Acorn RISC Machine

Between October 1983 and April 1985, Acorn Computers Limited in Cambridge, England, created the first ARM processor. ARM stood for Acorn RISC Machine at the time and until the foundation of Advanced RISC Machines Limited (later called simply ARM Limited) in 1990. Because of the popularity of the BBC (British Broadcasting Corporation) microcomputer, Acorn had established a strong position in the UK personal computer market. The BBC micro was an 8-bit microprocessor-based machine that quickly established itself as the dominant machine in UK schools following its launch in January 1982 in support of a series of BBC television programs. It also received enthusiastic support from the hobbyist community and was adopted by a number of research labs and higher education institutions.

Following the success of the BBC micro, Acorn's developers looked at different microprocessors to use in a successor computer, but all of the commercial options were missing. In 1983, 16-bit CISC microprocessors were available, although they were slower than ordinary memory

components. They also featured instructions that required many clock cycles (in some cases hundreds of clock cycles) to complete, resulting in extremely lengthy interrupt latency. The BBC micro profited immensely from the 6502's fast interrupt response, thus Acorn's designers were adamant that this feature of the processor's performance not be compromised.

The design of a proprietary microprocessor was contemplated as a result of these problems with commercial microprocessor products. The main stumbling issue was the fact that the Acorn team was well aware that commercial microprocessor programmes had consumed hundreds of man-years of design time. Because Acorn was a small company with only about 400 people, it couldn't consider such a large investment. It had to come up with a superior design in a fraction of the time, with no prior experience in bespoke chip design other than a few modest gate arrays for the BBC micro.

The papers on the Berkeley RISC I sprang out of nowhere in this seemingly improbable scenario. This was a processor that had been built in less than a year by a few postgraduate students and was competitive with the leading commercial products. There were no complex instructions to compromise the interrupt latency because it was fundamentally simple. It also came with supporting arguments that suggested it could be a harbinger of things to come, yet technical merit, no matter how strongly backed by academic reasoning, is no guarantee of commercial success.

The ARM was born as a result of a fortunate confluence of events, and it went on to become the main component of Acorn's product line. It later contributed its name to the firm founded to expand its market beyond Acorn's product line following a careful revision of the acronym expansion to Advanced RISC Machine. Despite the name change, the architecture is still quite similar to the Acorn design.

1.4.2 ARM 8 Architecture Introduction

1.4.2.1 Architectural inheritance

The Berkeley RISC I and II and the Stanford MIPS (which stands for Microprocessor without Interlocking Pipeline Stages) were the only examples of RISC architectures at the time the first ARM chip was designed, though some earlier machines, such as the Digital PDP-8, the Cray-1, and the IBM 801, which predated the RISC concept, shared many of the characteristics that later came to be associated with the RISC concept.

A number of Berkeley RISC design concepts were included into the ARM architecture, although others were discarded. A load-store architecture, fixed-length 32-bit instructions, and 3-address instruction formats were all used.

The following features were used on Berkeley RISC concepts that were rejected by ARM designers:

- **Register windows**

The Berkeley RISC processors' register banks contained a vast number of registers, with 32 of them visible at any given moment. The visible 'window' was shifted to provide each operation access to fresh registers, minimizing the data traffic between the CPU and memory caused by register saving and restoring.

The main issue with register windows is the high amount of chip space used up by the large number of registers. Although the shadow registers used to manage exceptions on the ARM are not too dissimilar in concept, this functionality was rejected on cost concerns.

Because it was included in the Berkeley prototypes in the early days of RISC, the register window technique was firmly connected with the RISC concept, although only the Sun SPARC architecture has embraced it in its original form since then.

- **Delayed branches**

Branches in pipelines obstruct the smooth flow of instructions, producing problems. Most RISC processors address the issue by employing delayed branches, which take effect after the next instruction has completed. Delay branches have the drawback of removing the atomicity of individual instructions. They function well on single-issue pipelined processors, but they don't scale well to super-scalar implementations and can cause problems when combined with branch prediction methods.

Delay branches were not utilized on the original ARM because they made exception handling more complicated; however, this has proved out to be a smart decision in the long run because it simplifies re-implementing the architecture with a new pipeline.

- **Single-cycle execution of all instructions**

Although the ARM can process most data in a single clock cycle, many other instructions require numerous clock cycles. The reasoning behind this was based on the fact that even a basic load or store instruction requires at least two memory accesses when using a single memory for both data and instructions (one for the instruction and one for the data). As a result, single-cycle operation of all instructions is only achievable with separate data and instruction memory, which were deemed too costly for the ARM application areas.

Instead of executing all instructions in a single cycle, the ARM was designed to use the fewest amount of cycles possible for memory accesses. Where this was more than one, the extra cycles were employed to do something beneficial, such as enable auto-indexing addressing modes, whenever possible. This minimizes the overall amount of ARM instructions needed to complete any given series of operations, resulting in improved performance and code density.

The need to keep the design basic was a major priority for the original ARM design team. Acorn designers had only worked with gate arrays with complexities of up to 2,000 gates prior to the first ARM processors; therefore the full-custom CMOS design medium was treated with caution. When travelling into unfamiliar area, it's best to limit the hazards that you can control, because there are still major risks from things that aren't well understood or fundamentally uncontrollable.

The ARM's simplicity is more visible in the hardware structure and implementation than in the instruction set architecture. From the perspective of the programmer, it manifests itself as conservatism in the ARM instruction set design, which, although adhering to the essential principles of the RISC approach, is less radical than many subsequent RISC designs.

The ARM's power-efficiency and tiny core size are due to the combination of basic hardware with an instruction set that is based on RISC ideas but preserves a few essential CISC elements, resulting in a substantially higher code density than a pure RISC.

1.4.2.2 About ARM Architecture

The ARM architecture, defines the behaviour of an abstract machine known as a Processing Element, or PE for short. Implementations that follow the ARM architecture must follow the Processing Element's defined behaviour. It is not intended to specify how to construct a PE implementation or to limit the scope of such implementations to the behaviours stated.

The programmer-visible behaviour of an implementation that is consistent with the ARM architecture must be the same as a simple sequential execution of the program on the processor element, unless the architecture specifies otherwise. The execution time of the program is not included in this programmer-visible behaviour.

- **An associated debug architecture**

- Corresponding trace architectures, which describe trace macrocells that implementers can implement with the associated processor hardware, are all defined in the ARM architecture.

The ARM architecture is RISC architecture with the following RISC architecture characteristics:

- A big file of uniform registers.

- A load/store architecture, in which data-processing operations are performed on register contents rather than memory contents directly.

- Modes with simple addressing, where all load/store addresses are determined only by register contents and instruction fields

The architecture specifies how the Processing Element interacts with memory, which includes caches, as well as a memory translation

mechanism. It also explains how several Processing Elements in a system interact with one another and with other observers.

The ARM architecture allows for implementations at a variety of performance levels. The ARM architecture is known for its small implementation size, high performance, and low power consumption.

Backwards compatibility, paired with the freedom to implement in a wide range of conventional and specific use cases, is a key characteristic of the ARMv8 architecture.

- AArch64, a 64-bit execution state compatible with prior versions of the ARM architecture
- AArch32, a 32-bit execution state compatible with previous generations of the ARM architecture

1.4.2.3 Architecture Profiles

Since its introduction, the ARM architecture has changed tremendously, and ARM continues to improve it. To date, eight major versions of the architecture have been defined, with version numbers ranging from 1 to 8. The first three versions are no longer in use.

The 64-bit and 32-bit execution states are referred to as AArch64 and AArch32, respectively.

- **AArch64:** This is the 64-bit execution state, which means that addresses are stored in 64-bit registers and that instructions in the base instruction set can operate 64-bit registers. The A64 instruction set is supported by the AArch64 state.
- **AArch32:** This is the 32-bit execution state, which means that addresses are stored in 32-bit registers and instructions in the base instruction sets are processed using 32-bit registers. The T32 and A32 instruction sets are supported by the AArch32 state.

Three architecture profiles are defined by ARM:

A: Application profile

- Supports a Memory Management Unit-based Virtual Memory System Architecture (VMSA) (MMU)
- AArchv8-A is the name given to an ARMv8-A implementation.
- The A64, A32, and T32 instruction sets are supported.

R: Real time profile

- Based on a Memory Protection Unit in real-time (MPU) it provides support for Protected Memory System Architecture (PMSA).
- Both A32 and T32 instruction sets are supported.

M: Microcontroller profile

- Provides a programmers' model for low-latency interrupt processing, including hardware register stacking and support for interrupt handlers written in high-level languages.
- Implements an R-profile PMSA variation.
- Supports a T32 instruction set variant.

1.4.2.4 ARMv8 architectural concepts

ARMv8 makes significant improvements to the ARM architecture while keeping a high level of compatibility with earlier versions.

The subsections that follow explain fundamental ARMv8 architectural concepts. Each section begins with an explanation of the concepts that are used to describe the architecture:

i] Execution state

- The PE execution environment is defined by the Execution state, which includes:
- The supported register widths.
- The instruction sets that are supported.
- Important features of:
  - The model of the outlier.
  - The Architecture of the Virtual Memory System (VMSA).
  - The model of the programmers.

The following are the execution states:

- **AArch64**
  - The state of 64-bit execution
  - Provides a 64-bit program counter (PC), stack pointer (SPs), and exception link registers
  - Provides 31 64-bit general-purpose registers, of which X30 is utilized as the procedure link register (ELRs)
  - Supports SIMD vector and scalar floating-point calculations with 32 128-bit registers
  - Has a single instruction set, A64
  - Support for 64-bit virtual addressing
  - Defines a number of PSTATE components that retain PE state
  - Defines the ARMv8 Exception model, with up to four Exception levels, EL0 - EL3, that give an execution privilege hierarchy Instructions that operate directly on certain PSTATE elements are included in the A64 instruction set

- Each system register is given a suffix that corresponds to the lowest Exception level at which it can be accessed.

- **AArch32**
  - The execution state is 32 bits.
  - Provides 13 32-bit general-purpose registers, as well as a 32-bit PC, SP, and link register in this execution state (LR). Both an ELR and a procedure link register, the LR is utilized.
  - For use in different PE modes, some of these registers contain numerous banked instances.
  - Provides a single ELR for Hyp mode exception returns.
  - Supports Advanced SIMD vector and scalar floating-point with 32 64-bit registers.
  - Both A32 and T32 instruction sets are included.
  - Supports the ARMv7-A exception model, which is based on PE modes, and translates it to the ARMv8 Exception model, which is based on Exception levels.
  - 32-bit virtual addresses are used.
  - The PE state is stored in a single Current Program State Register (CPSR).
  - Interprocessing is the process of switching between the AArch64 and AArch32 execution states.

Only by changing the Exception level can the PE switch between execution states. This means that software layers executing at distinct Exception levels, such as an application, an operating system kernel, and a hypervisor, might execute in separate execution states.

ii] ARM instruction sets

The potential instruction sets in ARMv8 are determined by the execution state:

- **AArch64:** Only one instruction set, A64, is supported by the AArch64 state. This is a 32-bit instruction set with a fixed length instruction set.
- **AArch32:** The following instruction sets are supported by the AArch32 state:
- **A32:** This is a 32-bit instruction set with a fixed length instruction set. It can be used with the ARMv7 instruction set
- **T32:** This is a variable-length instruction set with both 16-bit and 32-bit encodings. The ARMv7 Thumb® instruction set is supported
- Each of these instruction sets is expanded by ARMv8.

The instruction set that the PE executes is determined by the PE Instruction set state. SIMD and scalar floating-point instructions are supported by the ARMv8 instruction sets.

iii] System registers

Control and status information for architected features are provided through system registers. The naming format for System registers is <register_name>.<bit_field_name> to identify specific registers, as well as control and status bits within a register, use bit field name.

Bits can also be expressed numerically in the form <register name>[x:y] or in the generic form bits[x:y].

In addition, most register names in the AArch64 state include the lowest Exception level that can access the register as a suffix: <register_name>_ELx, where x is 0, 1, 2, or 3

The System registers consists of:

- General system control registers
- Registers for debugging.
- Timer registers that are generic.
- Performance Monitor can optionally register.
- Trace registers are optional.
- Generic Interrupt Controller (GIC) CPU interface registers are optional.

iv] ARMv8 Debug

The following are supported by ARMv8:

- **Debugging on your own server**

  The PE generates debug exceptions in this model. The ARMv8 Exception model includes debug exceptions.

- **Debugging from the outside**

  Debug events cause the PE to enter the Debug state in this model. The PE is managed by an external debugger in the Debug stage.

  Both models are supported by all ARMv8 implementations. The model chosen by a given user is determined by the debug requirements at various phases of the product's design and development life cycle. External debug, for example, may be utilized during hardware implementation and OS bring-up, while self-hosted debug could be used during program development.

1.4.2.5 Supported data types

The following integer data types are supported by the ARMv8 architecture:

- **Byte:** 8 bits
- **Halfword:** 16 bits
- **Word:** 32 bits
- **Doubleword:** 64 bits
- **Quadword:** 128 bits

Floating-point data types such as half precision, single precision, double precision, are also supported by the architecture.

It also supports:

- Fixed-point word and doubleword interpretation.
- Vectors, which consist of numerous elements of the same data type held in a single register.

There are two register files in the ARMv8 architecture:

- A registration file that can be used for general purpose.
- A file with SIMD and floating-point registers.

The available register sizes in each of them are determined by the Execution state.

In AArch64 state:

- A general-purpose register file comprises 64-bit registers in the AArch64 state
  - These registers can be accessed as 64-bit registers or as 32-bit registers by using only the bottom 32 bits in many operations.
- There are 128-bit registers in a SIMD and floating-point register file
  - The quadword integer data types are applicable only to the SIMD and floating-point register files.
  - The floating-point data types are applicable only to the SIMD and floating-point register files. Despite the fact that the AArch64 vector registers provide 128-bit vectors, the effective vector length depends on the A64 instruction encoding utilized.

In AArch32 state:

- A general-purpose register file comprises 32-bit registers in the AArch32 state:
  - A doubleword can be supported by two 32-bit registers.
  - The use of vector formatting is possible.
- 64-bit registers are contained in a SIMD and floating-point register file:
  - The quadword integer and floating-point data types are not supported in the AArch32 state.
  - A 128-bit register is made up of two successive 64-bit registers.

1.4.2.6 ARM memory model

The ARM memory model supports the following:

- Exception generation on an unaligned memory access is supported by the ARM memory model.
- Restricting application access to specific memory locations.
- Converting virtual addresses from executable instructions to physical addresses.
- Switching between big-endian and little-endian interpretation of multi-byte data.
- Managing the order in which memory accesses are made.
- Caches and address translation structures are under control.
- Multiple PEs accessing shared memory at the same time.

Support for virtual addresses (VA) is conditional on the Execution state, as follows:

AArch64 state

The Translation Control Register determines the VA range supported by the AArch64 state, which supports 64-bit virtual addressing. Two distinct VA ranges with their own translation controls are supported by execution at EL1 and EL0.

AArch32 state

The Translation Control Register determines the VA range supported by the AArch32 state, which supports 32-bit virtual addressing. The VA range can be split into two subranges, each with its own translation controls, for execution at EL1 and EL0.

System software can discover the supported physical address space, which is IMPLEMENTATION DEFINED. The Virtual Memory System Architecture (VMSA) can translate VAs to blocks or pages of memory anywhere within the supporting physical address space, regardless of the Execution state.

1.5 INTRODUCTION TO RASPBERRY PI

Raspberry Pi is a series of compact single-board computers developed by the Raspberry Pi Foundation in collaboration with Broadcom in the United Kingdom. These projects are generally inclined towards teaching and promoting basic computer science in schools and in developing countries. Due to its low cost, modularity and open design it finds wide application ranging from weather monitoring, robotics and many more.

Several generations have been released of Raspberry Pi's such as Raspberry Pi Model B (February 2012), followed by Model A, Model B+ (in 2014), Raspberry Pi 2(February 2015), Raspberry Pi Zero (November 2015), Raspberry Pi Zero W (On 28 February 2017), Raspberry Pi Zero WH (On 12 January 2018), Raspberry Pi 3 Model B (February 2016), Raspberry Pi 3 Model B+ (2018), Raspberry Pi 4 Model B (released in June 2019), Raspberry Pi 400 (November 2020) and Raspberry Pi Pico (in January 2021).

1.5.1 Introduction to Raspberry Pi

The Raspberry Pi is a fascinating device: it's a fully functional computer packed into a small and inexpensive compact. Whether you want to use the Raspberry Pi to surf the web or play games, learn how to write your own programs, or build your own circuits and physical devices, the Raspberry Pi – and its incredible community – will be there to help you every step of the way.

The Raspberry Pi is a single-board computer, which means it's a computer that's similar to a desktop, laptop, or smartphone but is built on a single printed circuit board. The Raspberry Pi, like most single-board computers, is little – it has about the same footprint as a credit card – but that doesn't mean it's not powerful: it can accomplish everything a larger, more power-hungry computer can do, just not as rapidly.

The Raspberry Pi family was created out of a desire to promote more hands-on computer education throughout the world. The Raspberry Pi Foundation, which was founded by its designers, had no clue it would become so popular: the first few thousand units manufactured in 2012 to test the waters were quickly sold out, and millions have been distributed all over the world in the years afterwards. These circuit boards have been found in homes, classrooms, businesses, data centers, factories, and even self-driving boats and space balloons.

Since the initial Model B, other Raspberry Pi variants have been released, each with enhanced specifications or functionality tailored to a certain use-case. The Raspberry Pi Zero line, for example, is a miniature version of the full-size Raspberry Pi that foregoes a few capabilities – notably multiple USB ports and a wired network interface – in favor of a much smaller footprint and lower power consumption.

Raspberry Pi is a single-board computer with a compact footprint. The Raspberry Pi may be used as a little computer by adding peripherals such as a keyboard, mouse, and display. Raspberry Pi is a popular platform for real-time image/video processing, IoT applications, and robotics. The Raspberry Pi is slower than a laptop or desktop computer, but it is still a computer that can give all of the expected features and abilities while using very little power.

Raspbian OS is based on Debian and is officially provided by the Raspberry Pi Foundation. They also offer NOOBS OS for Raspberry Pi. Several Third-Party OS versions, such as Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, and others, can be installed.

Raspbian OS is an approved operating system that may be used for free. This operating system is well-suited to the Raspberry Pi. Raspbian has a graphical user interface (GUI) that provides tools for browsing, Python programming, office, gaming, and more. To save the OS, we should use an SD card (minimum 8 GB is advised) (operating System).

Raspberry Pi is more than a computer because it allows developers to access on-chip hardware, such as GPIOs, to create applications. By using GPIO, we may connect and control devices such as LEDs, motors, and sensors. It includes an ARM-based Broadcom Processor SoC as well as an on-chip GPU (Graphics Processing Unit).

Raspberry Pi's CPU speed ranges from 700 MHz to 1.2 GHz. It also includes SDRAM on board, which varies from 256 MB to 1 GB. On-chip SPI, I2C, I2S, and UART modules are also available for the Raspberry Pi.

The Raspberry Pi is available in a variety of versions, which are listed below:

1. Raspberry Pi 1 Model A
2. Raspberry Pi 1 Model A+
3. Raspberry Pi 1 Model B
4. Raspberry Pi 1 Model B+
5. Raspberry Pi 2 Model B
6. Raspberry Pi 3 Model B
7. Raspberry Pi Zero

The following are the features of the aforementioned versions of Raspberry Pi that are most commonly used as described in the Table II:

Table II: Features of various versions of Raspberry Pi

Features	Raspberry Pi Model B+	Raspberry Pi 2 Model B	Raspberry Pi 3 Model B	Raspberry Pi zero
SoC	BCM2835	BCM2836	BCM2837	BCM2835
CPU	ARM11	Quad Cortex A7	Quad Cortex A53	ARM11
Operating Freq.	700 MHz	900 MHz	1.2 GHz	1 GHz
RAM	512 MB SDRAM	1 GB SDRAM	1 GB SDRAM	512 MB SDRAM
GPU	250 MHz Videocore IV	250MHz Videocore IV	400 MHz Videocore IV	250MHz Videocore IV
Storage	micro-SD	Micro-SD	micro-SD	micro-SD
Ethernet	Yes	Yes	Yes	No
Wireless	WiFi and Bluetooth	No	No	No

1.5.1.1 What's the Raspberry Pi foundation?

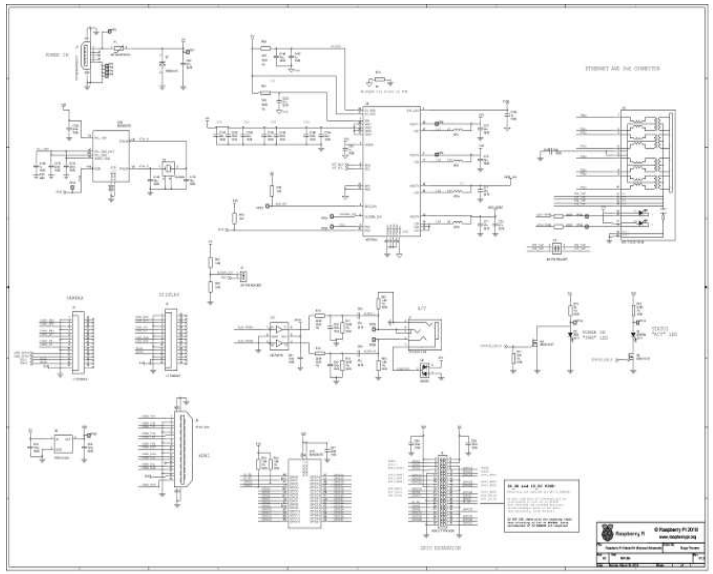
The Raspberry Pi Foundation is dedicated to putting the power of computers and digital fabrication into the hands of people all over the world. It accomplishes this by making low-cost, high-performance computers available for people to learn, solve issues, and have fun with. It conducts outreach and education to assist more people gain access to computing and digital making—it creates free materials to help people learn about computers and how to make things with them, and it also trains educators to help others learn.

The Raspberry Pi Foundation sponsors Code Club and CoderDojo, however both programs are platform-agnostic (they aren't bound to Raspberry Pi hardware). The Raspberry Pi Foundation promotes these clubs and assists in the expansion of the network around the world, ensuring that every child has the opportunity to learn about computers. Raspberry Jams, on the other hand, are Raspberry Pi-focused gatherings where people of all ages can learn about the Raspberry Pi and exchange ideas and projects.

1.5.1.2 Is Raspberry Pi open source?

The Raspberry Pi runs Linux (a number of versions), and its main supported operating system, Pi OS, is open source and runs a suite of open source software. The Raspberry Pi Foundation contributes to the Linux kernel and other open source projects, as well as publishing open source versions of many of its own software.

The schematics for the Raspberry Pi are frequently given as documentation, but the board is not open hardware.



1.5.1.3 Uses of Raspberry Pi

- **Community**

  One of the most intriguing aspects of the project, according to Jamie Ayre of FLOSS software business AdaCore, is the Raspberry Pi community. According to community blogger Russell Davis, the Foundation's strength allows it to focus on documentation and education. The community created The MagPi, a fanzine based on the platform that was handed over to the Raspberry Pi Foundation by its volunteers in 2015 to be maintained in-house. Across the UK and around the world, a series of community Raspberry Jam events have taken place.

- **Education**

  As of January 2012, inquiries about the board had been received from schools in both the public and private sectors in the United Kingdom, with the latter receiving around five times as much interest. Businesses are hoped to finance purchases for less fortunate schools. Premier Farnell's CEO stated that the government of a Middle Eastern country has expressed interest in distributing a board to every schoolgirl in order to improve her employment possibilities.

  The Raspberry Pi Foundation engaged a number of members of its community, including former teachers and software developers, in 2014 to create a set of free instructional tools on its website. The Foundation also launched Picademy, a teacher training program aimed at assisting teachers in preparing to teach the new computing curriculum using the Raspberry Pi in the classroom.

  NASA launched the JPL Open Source Rover Project in 2018 to encourage students and hobbyists to get involved in mechanical, software, electronics, and robotics engineering. The JPL Open Source Rover Project is a scaled-down version of the Curiosity rover that uses a Raspberry Pi as the control module.

- **Home automation**

  The Raspberry Pi is being used by a variety of developers and applications for home automation. These programmers are working to turn the Raspberry Pi into a low-cost energy monitoring and power usage solution. Because of the Raspberry Pi's low price, it has become a popular and cost-effective alternative to more expensive commercial solutions.

- **Industrial automation**

  TECHBASE, a Polish industrial automation company, released ModBerry, an industrial computer based on the Raspberry Pi Compute Module, in June 2014. The device includes a variety of interfaces, including RS-485/232 serial ports, digital and analogue inputs/outputs, CAN, and low-cost 1-Wire buses, all of which are

common in the automation sector. Because of the design, the Compute Module can be utilized in tough industrial conditions, implying that the Raspberry Pi is no longer confined to home and science projects, but can be extensively used as an Industrial IoT solution to fulfill Industry 4.0 goals.

SUSE announced commercial support for SUSE Linux Enterprise on the Raspberry Pi 3 Model B in March 2018, with a handful of unknown customers using the Raspberry Pi to provide industrial monitoring. TECHBASE introduced a Raspberry Pi Compute Module 4 cluster in January 2021 for usage as an AI accelerator, routing, and file server. One or more regular Raspberry Pi Compute Module 4s are housed in an industrial DIN rail enclosure, with some variants including one or more Coral Edge tensor processing units.

- **Commercial products**

  Critter & Guitari designed and manufactured the Organelle, a portable synthesiser, sampler, sequencer, and effects processor. It has a Raspberry Pi computer module that runs Linux on it. Next Thing Co. invented the OTTO digital camera. It has a Raspberry Pi Compute Module built in. It was successfully crowdfunded through a Kickstarter effort in May 2014. Slice is a digital media player that is powered by a Compute Module. It was funded through a Kickstarter effort in August of 2014. Slice's operating system is based on Kodi. The Raspberry Pi is used in a number of commercial thin client computer terminals.

- **Covid-19 pandemic**

  During the coronavirus pandemic in Q1 2020, Raspberry Pi computers saw a significant increase in demand, owing to an increase in working from home, as well as the use of many Raspberry Pi Zeros in ventilators for COVID-19 patients in countries like Colombia, which helped to relieve strain on the healthcare system. Raspberry Pi sales surpassed 640,000 units in March 2020, the second highest month in the company's history.

1.5.1.4 Raspberry Pi foundation hall of fame

Members of the Raspberry Pi Hall of Fame include:

A] Eben Upton

Eben Christopher Upton is presently employed by Broadcom as a Technical Director and ASIC Architect. He is the man who is known for being the founder and former trustee of the Raspberry Pi Foundation, as well as the current CEO of the Raspberry Pi trading firm. Eben Upton's primary responsibility is the creation of the Raspberry Pi device's general software and hardware architecture.

B] Paul Beech

Paul Beech designed the current Raspberry Pi Foundation logo and is now working on producing diagrams, posters, and developing the Official Raspberry Pi website. Pimoroni, which makes Pibow, PiHub, Pibrella, and other useful doo-hickeys to make raspberry pi more fun to study and engage with, counts him as a founding member.

C] Alex Bradbury

Alex Bradbury, a Ph.D. student at the University of Cambridge, has been a volunteer for the Raspberry Pi Foundation since its inception. Alex is in charge of maintaining repositories that contain custom versions of the Raspbian operating system, and he has even co-authored a popular book titled "Learning Python with Raspberry Pi."

D] Dom Cobley

Dom Cobley (Engineer at Broadcom) has made a number of successful contributions to turning the Raspberry Pi into a Media Streaming device. Dom Cobley has provided VideoCore firmware for the Raspberry Pi, Kernel maintenance, and even developed XBMC (Xtreme Box Media Center) as a developer to enable media streaming (Video and Audio) over the Raspberry Pi.

E] Peter Green

Peter Green created the Raspbian Debian derivative and manages the Raspbian repository. Peter is now working on making a stable Raspbian version based on Debian Jessie (Debian 8) available.

F] James Hughes

Since 2011, James Hughes has been one of the original volunteers for the Raspberry Pi Foundation. He is now the chief developer of Camera Board Software, the Moderator of the Pi Forum, and the diligent maintainer of the Raspberry Pi website and Twitter page.

G] Mike Thompson

Mike Thompson collaborated on the Raspbian operating system for the Raspberry Pi alongside Peter Green.

H] Gert Van Loo

Gert Van Loo is a Broadcom engineer who was responsible for the development of the first hardware design of Alpha boards in 2011, which later became known as the "Raspberry Pi." In addition, he created the Gertboard and Gertduino expansion boards for the Raspberry Pi.

I] Rob Mullins

Along with Eben Upton, Rob Mullins was a co-founder of the Raspberry Pi Foundation and served as a trustee until 2014. He is currently employed as a Senior Lecturer in the University of Cambridge's Computer Laboratory. Computer Architecture and VLSI- On-chip Interconnection networks, chip-multi-processors, and innovative parallel processing fabrics are among his areas of specialization.

1.5.1.5 Advantages of Raspberry Pi

The following are the benefits of using a Raspberry Pi:

1. The Raspberry Pi is a small, powerful, and efficient cum compact form factor computer that is also quite inexpensive to purchase. Raspberry Pi can be used by a variety of small and medium-sized businesses to perform functions such as web server, database server, and media server. As a result, a significant amount of money can be saved on the purchase of numerous servers.
2. Raspberry Pi can be used as a single platform for a wide range of programming tasks. Pi supports a variety of programming languages, and users can install the appropriate compiler to ensure proper code execution. Python, the main programming language used by Pi, is a simple and easy-to-learn language. It allows for more efficient code creation, fewer lines of code, and automatic memory management.
3. The product is open source and supports open source operating systems and apps. As a result, Raspberry Pi has access to a large number of operating systems in various variants of Linux, as well as millions of apps for that operating system.
4. The Raspberry Pi includes add-on hardware such as the Camera, Component Moduler Kit, Gertboard, and HAT board, allowing users to connect thousands of third-party devices like as buttons and LEDs to perform various tasks on the Pi.
5. The product is energy efficient and offers small businesses a greener, more ethical option. This credit card-sized product is simple to recycle and saves money on cooling solutions.

1.5.1.6 Disadvantages/Limitations of Raspberry Pi

The following are the Raspberry Pi's limitations/drawbacks:

1. Because the Ethernet Port and Processing CPU are not fast enough to process multitasking computing cycles, it cannot function as a full-fledged computer.
2. Does not work with a fully functional Windows operating system.
3. The product is limited to SMEs and is not particularly beneficial, whereas larger organizations/enterprises have access to a wide range of facilities.

4. Doesn't have a battery-backed Real Time Clock (RTC). NTP Server is the sole way to work with time, and most operating systems do this automatically.
5. There is no built-in ADC converter. For ADC, an external charger is used.
6. Bluetooth and Wi-Fi are not supported out of the box, and numerous USB-based dongles are likewise not supported for wireless connectivity.

1.5.1.7 Generations of Raspberry Pi

Various generations of Raspberry Pi, ranging from the Raspberry Pi Model A through the Raspberry Pi Model B+, as well as the recently introduced Raspberry Pi Zero, will be addressed in this part, along with their extensive technical specifications.

Raspberry Pi Model A

The Raspberry Pi Model A is the first generation of Raspberry Pi models to be launched. Two models of the Raspberry Pi Model A were released: the Raspberry Pi 1 Model A and the Raspberry Pi 1 Model A+.

A] Raspberry Pi: Model A

The Raspberry Pi Model A as shown in the figure 1.10 is a lower-spec version of the Raspberry Pi. Because this model of Pi lacks crucial hardware interfaces, it was designed specifically for embedded projects. In comparison to Model B, Model A is lighter and uses less energy. Model A has become obsolete and is no longer easily accessible on the market.

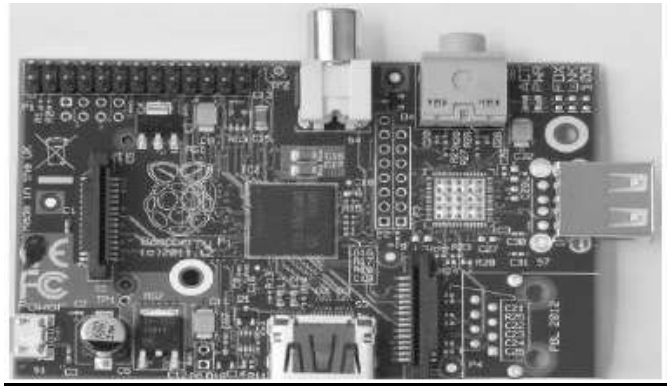


Figure 1.10 Raspberry Pi: Model A

The Technical Specification of the Raspberry Pi 1 Model A is listed in the Table III below:

Table III: Specifications of Raspberry Pi: Model A

Hardware parameters	Description
SoC	Broadcom BCM2835
CPU	700 MHz Single Core ARM 1176JZF-S
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	256 MB
Onboard Ports	1 USB; 1 HDMI (Ver 1.4); 3.5mm Sound Jack
Video Input	15-pin MIPI camera interface (CSI) Connector
Audio Input	2 Boards via I2S
Onboard Storage	SD/MMC/SDIO Card slot
Ethernet	No
GPIO	8 GPIO including UART, I2C, SPI Bus with two chip selects, I2S audio, +3.3 V, +5V, GND
Adapter Rating	5V; 300 mA
Launch Date	February 2013
Price	$25

B] Raspberry Pi: Model A+

In terms of size and power consumption, the Raspberry Pi 1 Model A+ as illustrated in the figure 1.11 was the successor and well-updated model of the Raspberry Pi 1 Model A. Additional GPIO pins, MicroSD card capability, and better audio reproduction are among the enhancements made with the Model A+. The Raspberry Pi Model A+ could also run a variety of operating systems and serve as a solid backbone for a variety of space-related applications and media center operations. Because more advanced variants are now available, the Model A+ is likewise being phased out of the market.

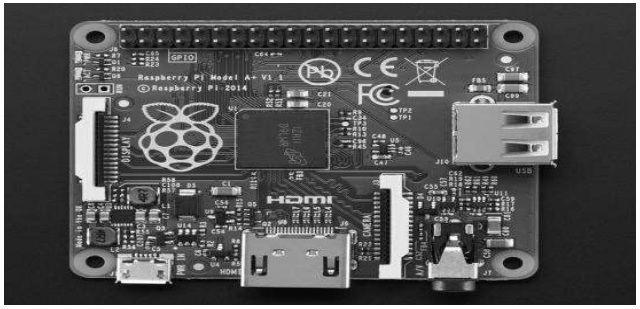


Figure 1.11 Raspberry Pi: Model A+

The Technical Specification of the Raspberry Pi 1 Model A+ is listed in the table IV below:

Table IV: Specifications of Raspberry Pi: Model A+

Hardware parameters	Description
SoC	Broadcom BCM2835
CPU	700 MHz Single Core ARM 1176JZF-S
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	256 MB
Onboard Ports	1 USB; 1 HDMI (Ver 1.4); 3.5mm Sound Jack
Video Input	15-pin MIPI camera interface (CSI) Connector
Audio Input	2 Boards via I2S
Onboard Storage	MicroSD Card slot
Ethernet	No
GPIO	17 GPIO including UART, I2C, SPI Bus with two chip selects, I2S audio, +3.3 V, +5V, GND, HAT ID Bus
Adapter Rating	5V; 200 mA
Launch Date	February 2014
Price	$20

C] Raspberry Pi: Model B

Because of the large RAM, additional USB port slots, and Ethernet port, the Raspberry Pi 1 Model B (Figure 1.12) was viewed as a higher specification model of the Pi 1 Model A with good working performance. Raspberry Pi 1 Model B paved the way for children to pursue computing as a hobby, leading to education, programming, and home projects.

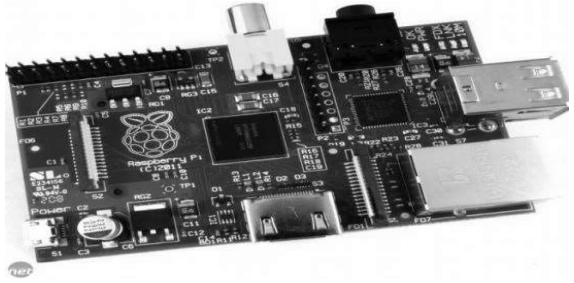


Figure 1.12 Raspberry Pi: Model B

The Technical Specification of the Raspberry Pi 1 Model B is listed in the table V below:

Table V: Specifications of Raspberry Pi: Model B

Hardware parameters	Description
SoC	Broadcom BCM2835
CPU	700 MHz Single Core ARM 1176JZF-S
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	512 MB
Onboard Ports	2 USB; 1 HDMI (Ver 1.4); 3.5mm Sound Jack
Video Input	15-pin MIPI camera interface (CSI) Connector
Audio Input	2 Boards via I2S
Onboard Storage	SD/MMC/SDIO Card
Ethernet	10/100 Mbps
GPIO	8 GPIO including UART, I2C, SPI Bus with two chip selects, I2S audio, +3.3 V, +5V, GND, Additional 4 GPIO on P5 pad
Adapter Rating	5V; 700 mA
Launch Date	February 2012
Price	$35

C] Raspberry Pi: Model B+

Under the Raspberry Pi 1 models category, Model B+ (Figure 1.13) was considered the last cum final version. Model B+ superseded Model B and had more enhanced hardware features like as more GPIO, more USB ports, a better MicroSD card, lower power consumption, and better audio output when compared to all Raspberry 1 generations products.

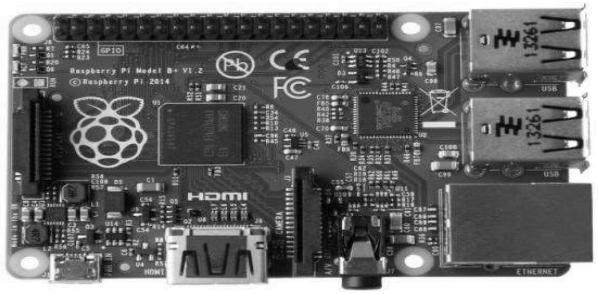


Figure 1.13 Raspberry Pi: Model B+

The Technical Specification of the Raspberry Pi 1 Model B+ is listed in the table VI below:

Table VI: Specifications of Raspberry Pi: Model B+

Hardware parameters	Description
SoC	Broadcom BCM2835
CPU	700 MHz Single Core ARM 1176JZF-S
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	512 MB
Onboard Ports	4 USB; 1 HDMI (Ver 1.4); 3.5mm Sound Jack
Video Input	15-pin MIPI camera interface (CSI) Connector
Audio Input	2 Boards via I2S
Onboard Storage	MicroSD Card
Ethernet	10/100 Mbps
GPIO	17 GPIO including UART, I2C, SPI Bus with two chip selects, I2S audio, +3.3 V, +5V, GND, HAT ID bus
Adapter Rating	5V; 600 mA
Launch Date	July 2014
Price	$25

Raspberry Pi 2 Model B

After the Model A generations, the Raspberry Pi Model B generations were released, with improved functionality, more powerful hardware, and better operating system support.

A] Raspberry Pi 2: Model B

The Raspberry Pi 2 Model B (Figure 1.14) is the Raspberry Pi's second iteration. In terms of a powerful CPU, RAM, GPIO, and other hardware connector features, it superseded the Raspberry Pi 1 Model B+ variants.

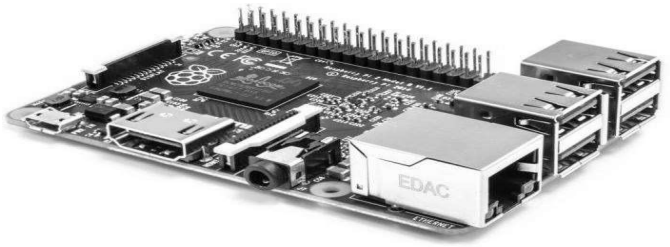


Figure 1.14 Raspberry Pi2: Model B

The Technical Specification of the Raspberry Pi 2 Model B is listed in the table VII below:

Table VII: Specifications of Raspberry Pi2: Model B

Hardware parameters	Description
SoC	Broadcom BCM2836
CPU	900 MHz Quad-Core ARM Cortex-A7
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	1 GB
Onboard Ports	4 USB; 1 HDMI (Ver 1.4); 3.5mm Sound Jack
Video Input	15-pin MIPI camera interface (CSI) Connector
Audio Input	2 Boards via I2S
Onboard Storage	MicroSD Card
Ethernet	10/100 Mbps
GPIO	17 GPIO including UART, I2C, SPI Bus with two chip selects, I2S audio, +3.3 V, +5V, GND, HAT ID bus
Adapter Rating	5V; 800 mA
Launch Date	February 2015
Price	$35

Raspberry Pi - Zero

The Raspberry Pi ZERO (Figure 1.15) is a new member of the Raspberry Pi family. It is the cheapest and most affordable board, costing around $5. Raspberry Pi Zero is capable of running Raspbian and all other programs that other Pi's can. The size is approximately half that of the A+ model, and the quantity of utilities is doubled.

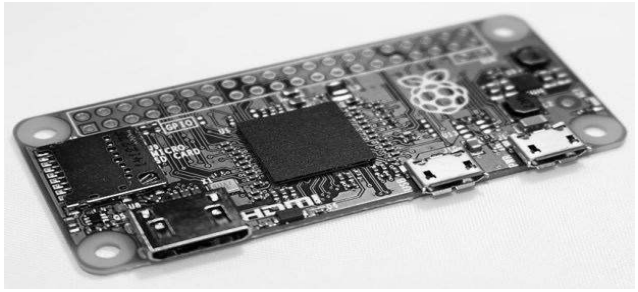


Figure 1.15 Raspberry Pi - Zero

The Technical Specification of the Raspberry Pi Zero is listed in the table VIII below:

Table VIII: Specifications of Raspberry Pi Zero

Hardware parameters	Description
SoC	Broadcom BCM2835
CPU	1GHz ARM 1176JZF-S Single Core
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	512 MB
Onboard Ports	Micro-USB; Mini-HDMI (Ver 1.4); Audio via PWN on GPIO
Video Input	N/A
Audio Input	2 Boards via I2S
Onboard Storage	MicroSD Card
Ethernet	N/A
GPIO	40 GPIO Pins
Adapter Rating	5V; 160 mA
Launch Date	November 2015
Price	$5

1.5.1.8 Raspberry Pi operating systems

The operating system is considered to be the most important software for computer hardware to function and to provide an interface between the computer hardware and the programs that are running. There are numerous operating systems for the Raspberry Pi that are based on Linux and are free and open source.

Raspberry Pi and Linux

Linus Torvalds, the creator of the Linux operating system, made Linux available as a platform for community development. The Raspberry Pi Foundation opted to include Raspbian Pi, an official Linux distribution that is tailored for Raspberry Pi.

Firmware and Kernel

Kernel is regarded as the "Brain" of the whole system, with the operating system serving as the "Outer Body." Kernel is an operating system

component that interacts with installed hardware devices. Because it is software that is semi-permanently written on Partition 1 of the SD card, kernel is also known as "Firmware." This section will provide an overview of the numerous operating systems that can be installed and supported by the Raspberry Pi.

There are two types of operating systems available for the Raspberry Pi:

A] Officially available operating systems:

1] Raspbian Operating System

Based on Debian, the Raspbian operating system is tailored for Raspberry Pi devices. Raspbian is a collection of programmes and utilities that run on the Raspberry Pi. It comes with over 35000 packages and is straightforward to install on the Raspberry Pi.

Raspbian Pi, as the Pi's primary operating system, has been designed for speed and stability, and is also being actively developed by the open source community.

Download: https://www.raspbian.org/RaspbianImages

Latest Version: Raspbian Jessie; Kernel Version: 4.1

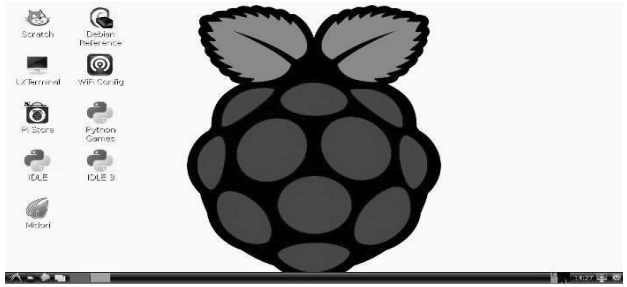


Figure 1.16 GUI interface of Raspbian operating system

2] Arch Linux ARM

Arch Linux ARM is a Linux distribution that is specifically designed for ARM processors. Arch Linux ARM is known for being easy to use and giving end users complete control. It provides a lightweight foundation framework that allows users to configure the system according to their needs, and it is only because of this that Arch Linux ARM lacks a GUI interface.

Arch, like Raspbian, is under constant development and is updated on a regular basis.

Download: https://www.archlinux.org/download/

Latest Version: 2015.12.1

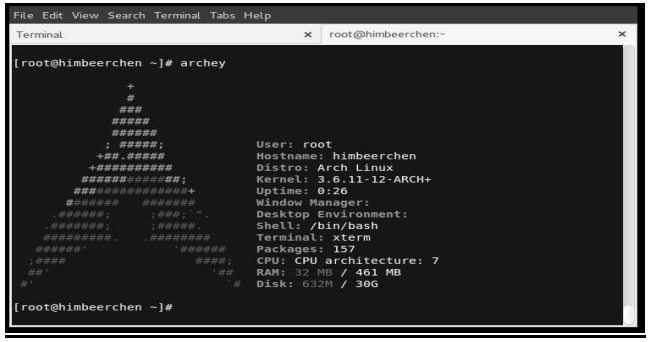


Figure 1.17 CUI Interface of ARCH Linux for Raspberry Pi

3] OpenELEC (Open Embedded Linux Entertainment Center)

OpenELEC is a Linux-based distribution that is specifically built for managing HTPCs and is based on KODI (XBMC-Media Player).

XBMC Frodo 12.1 is supported by OpenELEC. OpenELEC is primarily intended for speedier system booting, and it can transform any blank PC into a full-fledged media streaming computer in about 15 minutes. The OpenELEC operating system is optimised for a variety of architectures, including Atom, ION, Intel, Fusion, Raspberry Pi, and others.

Download: http://openelec.tv/

Latest Version: 3.0.0

Figure 1.18 GUI Interface of OpenELEC Raspberry Pi

4] Pidora

The “Raspberry Pi Fedora Remix” operating system is also known as Pidora. Pidora is a Linux distro created specifically for the Raspberry Pi. It consists of Fedora Project software packages that have been specifically tailored/modified to work on the Raspberry Pi. Pidora, like a Fedora-based operating system, also provides a platform for the open source community to submit apps to the operating system.

Download: http://pidora.ca/

Latest Version: Pidora 2014

Figure 1.19 GUI Interface of Pidora Operating System

5] Puppy Linux

Puppy Linux is a lightweight distribution that focuses on ease of use and minimal memory use. Puppy Linux has been modified for the Raspberry Pi and includes a large number of application suites. Like other distributions, open source community developers and even penetration testers are working throughout the world to improve the system's reliability, performance, and efficiency, and the community provides regular software and updates for the operating system, as well as bug fixes.

Download:
http://puppylinux.org/main/Download%20Latest%20Release.htm

Latest Version: Slacko Puppy 6.3

Figure 1.20 GUI Interface for Puppy Linux for Raspberry Pi

6] RISC OS

The ARM Team created RISC OS specifically for ARM processors. Because it is not tied to Windows or Linux, RISC OS is an extremely fast, compact, and efficient operating system. It includes a full desktop environment as well as a library of applications for Raspberry Pi.

Download:
https://www.riscosopen.org/content/downloads/raspberry-pi

Latest Version: RISC OS 14

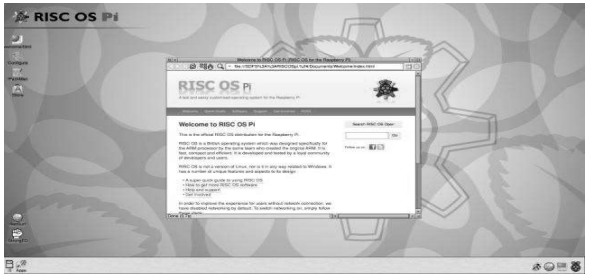


Figure 1.21 GUI Interface of RISC OS for Raspberry Pi

7] OSMC (Open Source Media Center)

Open Source Media Center is a Linux distribution centred on a free and open source media player with over 30000 packages. OSMC is a free and open source operating system that just takes a few minutes to set up. OSMC has a thriving community that releases updates and new packages on a monthly basis. “As OSMC says, ‘Play Anything from Anywhere.”

Download: https://osmc.tv/download/

Version: 2015.11.1

Figure 1.22 GUI Interface of OSMC for Raspberry Pi

8] Ubuntu Mate

Ubuntu Mate is a Raspberry Pi-specific version of Ubuntu 15.10 that was launched on October 22, 2015. With Ubuntu Mate, the Pi now has access to the same huge software repository as Ubuntu. Ubuntu Mate is a full-featured desktop environment that can run a variety of graphical applications as well as other standard Ubuntu tasks. Ubuntu Mate is the result of Raspberry Jams' efforts to improve “out of the box” GPIO functionality.

Download: https://ubuntu-mate.org/wily/

Latest Version: Ubuntu Mate 15.10

Figure 1.23 GUI Interface for Ubuntu Mate 15.10

9] Window 10 IoT core

Microsoft's Windows 10 IoT Core is a platform for creating IoT-based applications for the Raspberry Pi. The Windows 10 IoT core delivers the power of Windows to Raspberry Pi, making it simple to integrate rich experiences such as natural user interfaces, searching, online storage, and even cloud computing with gadgets.

Download: http://ms-iot.github.io/content/en-US/Downloads.htm

Latest Version: Windows 10

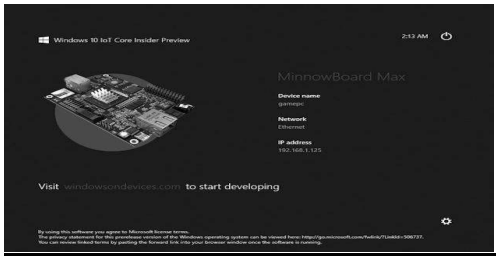


Figure 1.24 Windows 10 IoT Core

B] Miscellaneous Operating system

Other operating systems that can be downloaded and installed on the Raspberry Pi include as follows:

1] Q4OS: Raspberry Pi operating systems that are fast and powerful, with a focus on security, dependability, long-term stability, and cautious incorporation of validated new features.

2] Xbian: Xbian is a media centre distribution for the Raspberry Pi that is tiny, fast, and lightweight. Based on Debian minor, this is the fastest Kodi solution for a variety of small form factor computers.

3] openSUSE: SuSE Linux Professional, previously known as SUSE Linux, is a good platform for open source tools for software developers and administrators, as well as a user-friendly desktop and feature-rich server GUI interface.

4] FressBSD: Since November 2012, FreeBSD has supported the Raspberry Pi and is identical to Linux. FreeBSD is a full-featured operating system that includes kernel, device drivers, userland utilities, and documentation.

5] Kali Linux: Kali Linux, a Debian-based forensics and penetration testing operating system, now includes support for the Raspberry Pi. It comes with over 600 testing programs, a graphical user interface, and other ethical hacking tools.

6] SailPi: The SailPi operating system is based on the Sailfish OS 2.0.0.10 version. This operating system has a more powerful OS core, supports a variety of architectures, including Intel Atom and the Raspberry Pi, and offers good security, multitasking, and a better user interface.

1.5.2 Raspberry Pi Hardware

Unlike a standard computer, which has all of its components, ports, and features hidden behind a cover, a Raspberry Pi has all of its components, ports, and functions on show — though you may purchase a case for added protection if you like. This makes it an excellent tool for learning about the functions of various computer components, as well as for figuring out where to plug in the numerous extras (known as peripherals) you'll need to get started. Figure 1.25 (below) depicts the Raspberry Pi from above.

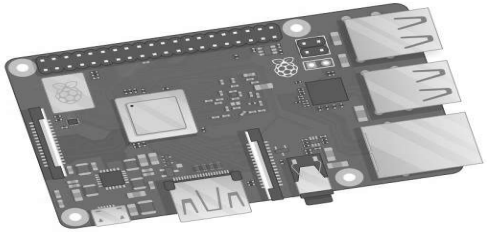


Figure 1.25 Raspberry Pi Model B+

While the Raspberry Pi appears to have a lot crammed (crowd) into its little board, it's actually quite simple to understand, starting with its components and the inner workings that keep the gadget running.

The Pi, like any computer, is made up of a variety of components, each of which plays an important function in its operation. The first, and possibly most essential, of these is the system-on-chip, which can be found right above the center point on the top side of the board (Figure 1.26), covered in a metal cap (SoC).

Figure 1.26 The Raspberry Pi's system-on-chip

The name system-on-chip gives you a good idea of what you'll find if you pry the metal cover off the Raspberry Pi: a silicon chip, also known as an integrated circuit that houses the majority of the Raspberry Pi's system. This includes the central processing unit (CPU), which is known as a computer's "brain," and the graphics processing unit (GPU), which is in charge of the visual side of things.

However, a brain is useless without memory, and on the Raspberry Pi's underbelly, you'll find just that: another chip, which looks like a small black

plastic square (Figure 1.27). These components work together to create the Pi's volatile and non-volatile memories: the volatile RAM loses its contents when the Pi is turned off, whilst the non-volatile microSD card preserves its contents.

Figure 1.27 Raspberry Pi's random access memory (RAM)

When you flip the board over, you'll notice another metal lid in the upper-right corner, this one with an etched Raspberry Pi logo (Figure 1.28). This section discusses the radio, which allows the Raspberry Pi to communicate wirelessly with other devices. In actuality, the radio has two main functions: a WiFi radio for connecting to computer networks, and a Bluetooth radio for connecting to peripherals such as mice and sending and receiving data from nearby smart devices such as sensors and smartphones.

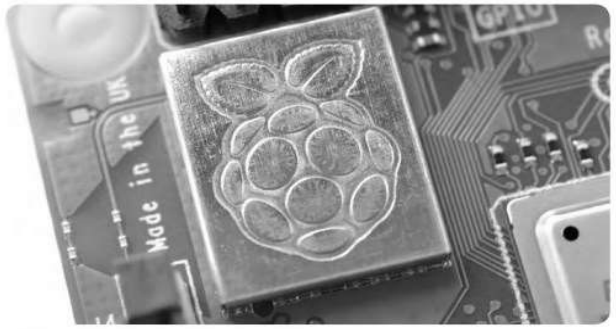


Figure 1.28 The Raspberry Pi's radio module

Just behind the middle row of USB ports, another black, plastic-covered chip can be seen near the bottom border of the board. This is the network and USB controller, which is in charge of the Ethernet port as well as the four USB ports. A final black chip, much smaller than the others, can be found just above the micro USB power connector on the upper-left side of the board (Figure 1.29); this is known as a power management integrated circuit (PMIC), and it handles converting the power from the micro USB port into the power the Pi requires to run.

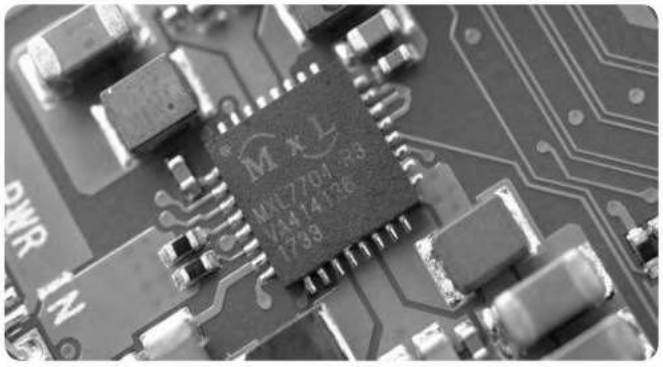


Figure 1.29 Raspberry Pi's power management integrated circuit (PMIC)

The Raspberry Pi's port

The Raspberry Pi features a variety of ports, starting with four USB ports on the center and right-hand sides of the bottom edge (Figure 1.30). These ports allow you to attach any USB-compatible peripheral to the Pi, including keyboards, mouse, digital cameras, and flash drives. These are known as USB 2.0 ports in technical terms, which indicates they are based on the Universal Serial Bus standard version two.

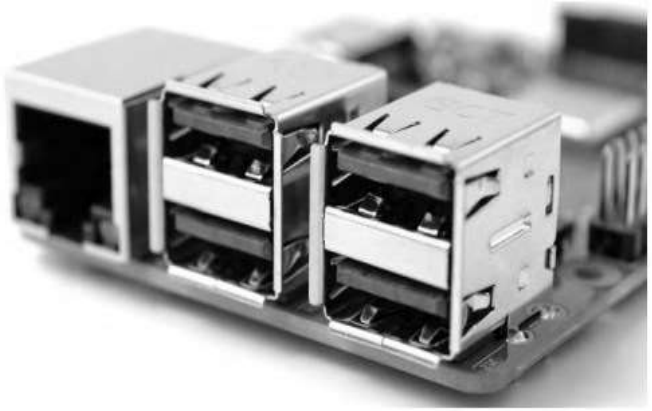


Figure 1.30 The Raspberry Pi's USB ports

An Ethernet port, often known as a network port, is located to the left of the USB ports (Figure 1.31). This port can be used to connect the Raspberry Pi to a wired computer network through a cable with an RJ45 connector on the other end. If you look closely at the Ethernet port, you'll notice two light-emitting diodes (LEDs) on the bottom; these are status LEDs that indicate whether or not the connection is active.

Figure 1.31 The Raspberry Pi's Ethernet ports

A 3.5 mm audio-visual (AV) jack is located just above the Ethernet port on the Raspberry Pi's left-hand edge (Figure 1.32). This is also known as the headphone jack, and it can be used for that purpose — albeit connecting it to amplified speakers rather than headphones will provide superior sound. The 3.5 mm AV jack, however, has a secret feature: in addition to audio, it transmits a video signal that can be linked to TVs, projectors, and other displays that support a composite video signal with a special connection known as a tip-ring-ring-sleeve (TRRS) adapter.

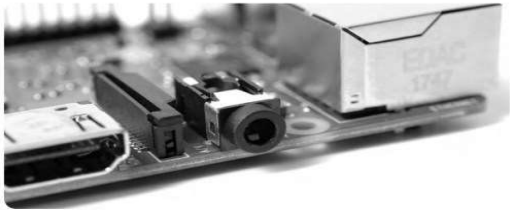


Figure 1.32 The Raspberry Pi's 3.5mm AV jack

A strange-looking connector with a plastic flap that can be pushed up sits directly above the 3.5 mm AV jack; this is the camera connector, also known as the Camera Serial Interface (CSI) (Figure 1.33). This enables you to use the Raspberry Pi Camera Module, which was created specifically for the Raspberry Pi.

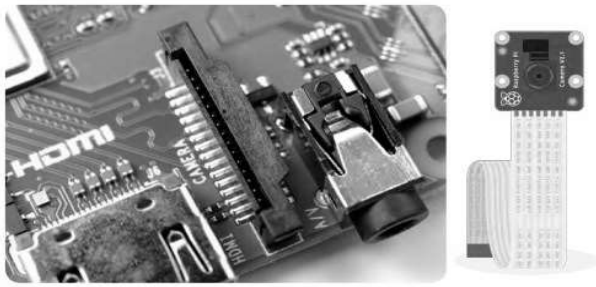


Figure 1.33 Raspberry Pi's camera connector

The High-Definition Multimedia Interface (HDMI) connection (Figure 1.34), which is the same sort of connector seen on a games console, set-top box, and TV, is located above that, still on the left-hand edge of the board. The multimedia component of its name indicates that it can carry both audio and video information, while high-definition indicates that the quality will be superb. This will be used to link the Raspberry Pi to your display device, which could be a computer monitor, television, or projector.

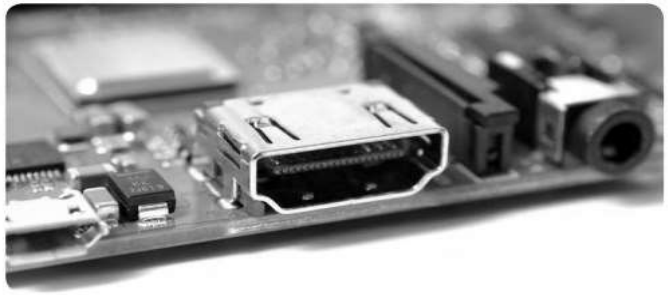


Figure 1.34 Raspberry Pi's HDMI port

A micro USB power port (Figure 1.35), located above the HDMI port, is used to connect the Raspberry Pi to a power source. On smartphones, tablets, and other portable gadgets, the micro USB port is a regular appearance. So you could use a regular phone charger to power the Pi, but the official Raspberry Pi USB Power Supply is recommended for optimum performance.

Figure 1.35 The Raspberry Pi's micro USB power port

Another strange-looking connector (Figure 1.35) can be seen towards the top edge of the board, which at first glance appears to be identical to the camera connector. This, on the other hand, is a display connector, or Display Serial Interface (DSI), made specifically for the Raspberry Pi Touch Display (Figure 1.36).

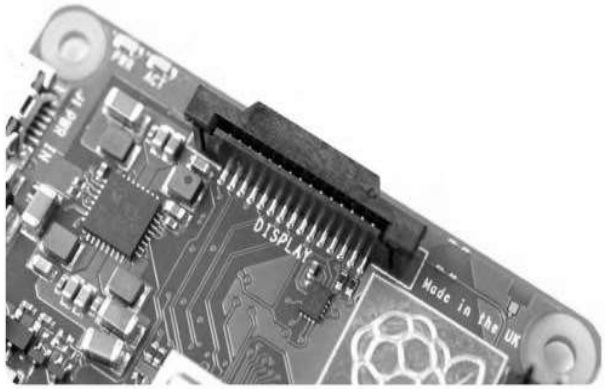


Figure 1.35 The Raspberry Pi's display connector (DSI)

Figure 1.36 The Raspberry Pi's touch display

There are 40 metal pins on the right-hand edge of the board, divided into two rows of 20 pins each (Figure 1.37). The GPIO (general-purpose input/output) header is a feature of the Raspberry Pi that allows it to communicate with external hardware such as LEDs, buttons, temperature sensors, joysticks, and pulse-rate monitors. Another, smaller header with four pins is just below and to the left of this header: This is used to attach the Power over Ethernet (PoE) HAT, an optional add-on that allows the Raspberry Pi to get power through a network connection instead of the micro USB socket.

Figure 1.37 The Raspberry Pi's GPIO header

The Raspberry Pi has one more port, but it's not visible from the top. Turn the board over, and on the opposite side of the board from the display connector is a microSD card connector (Figure 1.38). The Raspberry Pi's storage is as follows: All of the files you save, all of the software you install, and the operating system that makes the Raspberry Pi function are stored on the microSD card put in this slot.

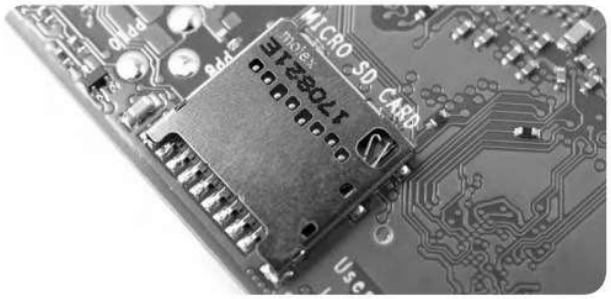


Figure 1.38 The Raspberry Pi's microSD card connector

The Raspberry Pi's peripherals

A Raspberry Pi by itself can't do much, like a desktop computer by itself isn't much more than a door-stop. A microSD card for storage, a monitor or TV to view what you're doing, a keyboard and mouse to tell the Pi what to do, and a 5 volt (5 V) micro USB power source rated at 2.5 amps (2.5 A) or better are all required for the Raspberry Pi to work. You've got yourself a completely functional computer with those.

The Raspberry Pi Case helps protect the Pi while you're using it without blocking access to its various ports; the Camera Module, the Raspberry Pi Camera Module; the Raspberry Pi Touch Display, which connects to the display port and provides both a video display and a tablet-style touchscreen interface; and the SPI Display, which connects to the display port and provides both a video display and a tablet-style touchscreen interface; the SPI Display and the sense HAT (figure 1.39)

A wide range of third-party accessories are also available, ranging from kits to convert a Raspberry Pi into a laptop or tablet to add-ons that allow it to comprehend and respond to your voice.

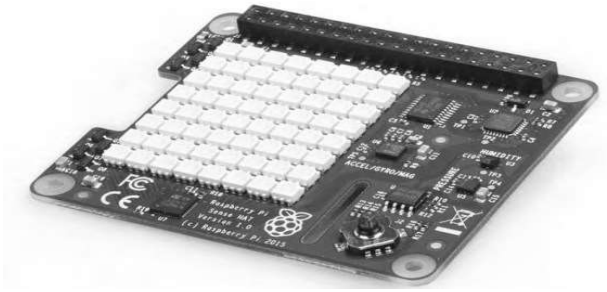


Figure 1.39 The sense HAT

1.5.3 Preparing your raspberry Pi

The Raspberry Pi was created to be as simple to set up and operate as possible, but it, like any computer, is dependent on a variety of external components known as peripherals. While it's fine to look at the Raspberry Pi's bare circuit board, which differs drastically from the encased, closed-off computers you're used to, and worry that things are about to get complicated, this isn't the case. Simply follow the procedures outlined in the next section to have the Raspberry Pi up and running in under ten minutes.

If you have the Raspberry Pi Starter Kit, you'll have almost everything you need to get started: all you need is a computer monitor or TV with an HDMI connection (the same type of connector used by set-top boxes, Blu-ray players, and games consoles) so you can see what the Raspberry Pi is up to. If you don't have the Raspberry Pi Starter Kit, you'll also need the following items in addition to the Raspberry Pi 3 Model B+:

USB power supply: A power supply having a micro USB connector and a rating of 2.5 amps (2.5A) or 12.5 watts (12.5W). The Official Raspberry Pi Power Supply is the best option because it can handle the Raspberry Pi's fast switching power demands.

NOOBS on a microSD card: The microSD card serves as the Raspberry Pi's permanent storage, storing all of the data you generate and software you install, as well as the operating system itself. A 8GB card will get you started, but a 16GB card will give you more room to expand. Using a card with pre-installed NOOBS (New Out-Of-Box Software) will save your time.

USB keyboard and mouse: The Raspberry Pi can be controlled with the help of a USB keyboard and mouse. Almost any USB-connected wired or wireless keyboard and mouse will function with the Raspberry Pi, though

some ‘gaming' keyboards with colourful LEDs may drain too much power to be used reliably.

HDMI Cable: The HDMI cable connects your Raspberry Pi to your TV or monitor and transmits sound and video. There's no need to splurge on a high-end HDMI cable. If you want to connect your Raspberry Pi to an older TV that uses composite video or has a SCART socket, use a 3.5 mm tip-ring-ringsleeve (TRRS) audio/video cable; if you want to connect your Raspberry Pi to an older TV that uses composite video or has a SCART socket, use a 3.5 mm tip-ring-ringsleeve (TRRS) audio/video cable.

Without a case, the Raspberry Pi is safe to use as long as it is not placed on a metal surface that could conduct electricity and cause a short-circuit. However, an optional case can give further protection; the Starter Kit contains the Official Raspberry Pi Case, while third-party cases can be found at any respectable retailer.

You'll also need a network cable if you wish to use the Raspberry Pi on a wired network rather than a wireless (WiFi) network. This should be connected to your network's switch or router on one end. You won't need a cable if you wish to utilize the Raspberry Pi's built-in wireless radio; you will, however, need to know the name and key or pass for your wireless network.

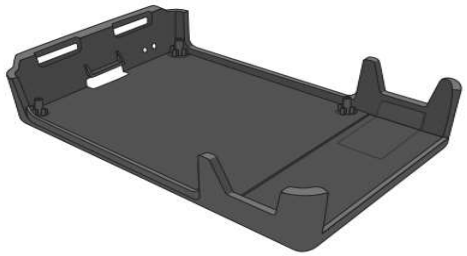
1.5.3.1 Setting up the hardware

You'll also need a network cable if you wish to use the Raspberry Pi on a wired network rather than a wireless (WiFi) network. This should be connected to your network's switch or router on one end. You won't need a cable if you wish to utilize the Raspberry Pi's built-in wireless radio; you will, however, need to know the name and key or pass for your wireless network.

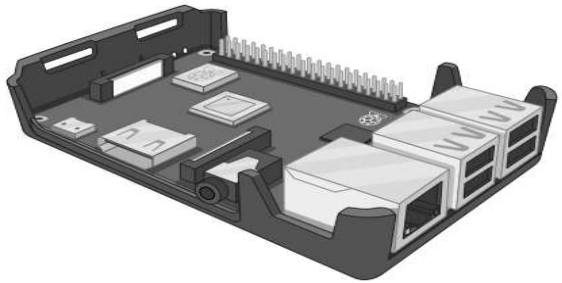
Assembling the case

It should be your first step if you're placing your Pi in a case. If you're using the Official Raspberry Pi Case, start by separating the five pieces: the red base, two white sides, red upper and white lid.

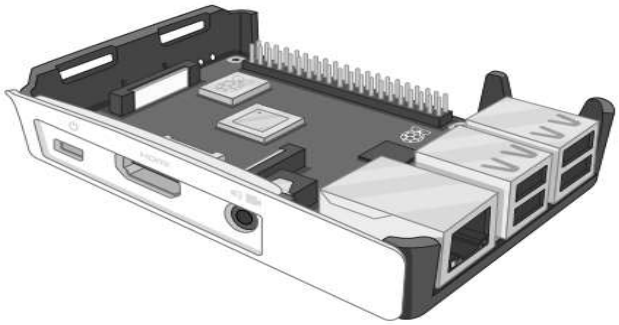
1] Take the base and place it on your left side with the elevated end facing you and the lower end facing you.



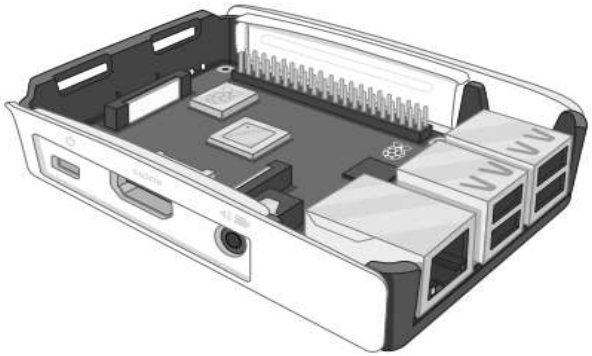
2] Holding the Pi by its USB and Ethernet ports and the GPIO header at the top, insert the left-hand side into the case at an angle, then slowly drop the right-hand side down until it lies flat.



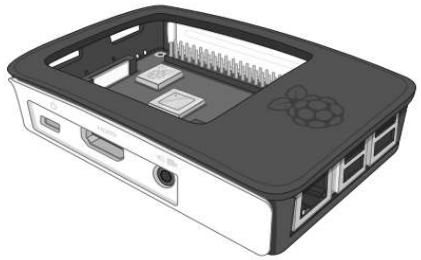
3] Find the one with the cutouts for the power connector, HDMI port, and 3.5 mm AV jack among the two white side parts. Line it up with the Raspberry Pi's ports and carefully press it in until you hear a click.



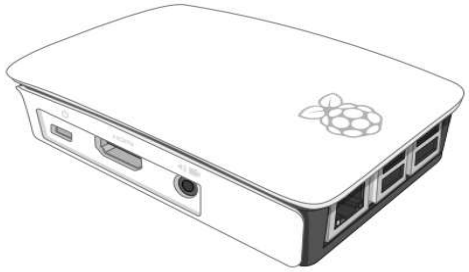
4] Place the solid white side piece on the GPIO header side of the casing and click it in place.



5] Place the two clips on the left of the red plastic upper piece into the matching holes on the left of the base, above the microSD card slot. Push the right-hand side (above the USB ports) down until you hear a click once they're in place.

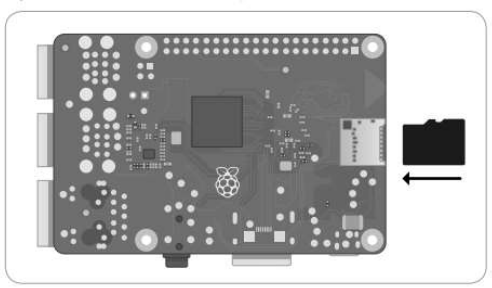


6] Finally, carefully press the white lid down until you hear a click, making that the Raspberry Pi logo is to your right and the small raised clips on its underside are lined up with the hole on the top of the case. Your case is now complete.

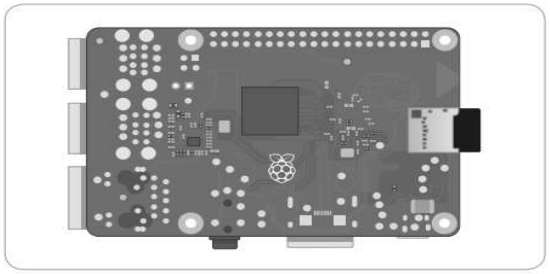


Connecting the microSD card

Turn the Raspberry Pi over and insert the microSD card into the microSD slot with the label facing away from the Pi to install the microSD card, which is the Raspberry Pi's storage. It can only go one way and should go into place without too much difficulty.



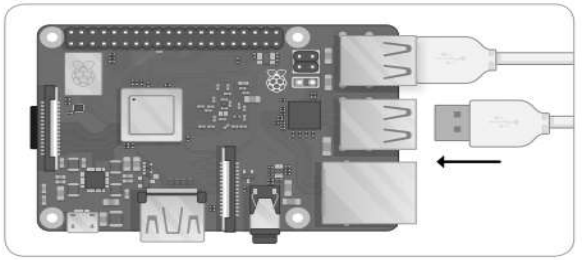
The microSD card will go into the connector and then come to a halt without making a click.



If you want to remove it in the future, simply grab the card's end and carefully pull it out. If you're using an earlier Raspberry Pi, you'll need to gently push the card to unlock it; this isn't necessary if you're using a Raspberry Pi 3 or newer.

Connecting a keyboard and a mouse

Connect the USB connection from the keyboard to one of the Raspberry Pi's four USB ports. Once the keyboard is connected, attach the mouse in the same way.

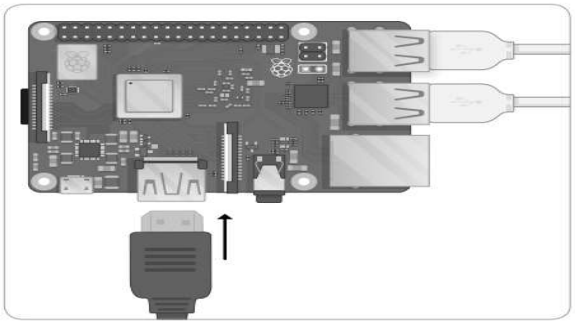


The USB connectors for the keyboard and mouse should slip into place without too much force; if you have to force them in, something is amiss. Make sure the USB connector is pointing in the appropriate direction!

- MOUSE & KEYBOARD: The keyboard and mouse are your primary way of instructing the Raspberry Pi; these are known as input devices in computing, as opposed to the display, which is an output device.

Connecting a display

Connect one end of the HDMI cable to your Raspberry Pi and the other end to your monitor (it doesn't matter which). Look for a port number next to the connector itself if your display has more than one HDMI port; you'll need to switch the TV to this input to see the Pi's display. Don't worry if you can't see a port number: simply switch through each input until you find the Pi.

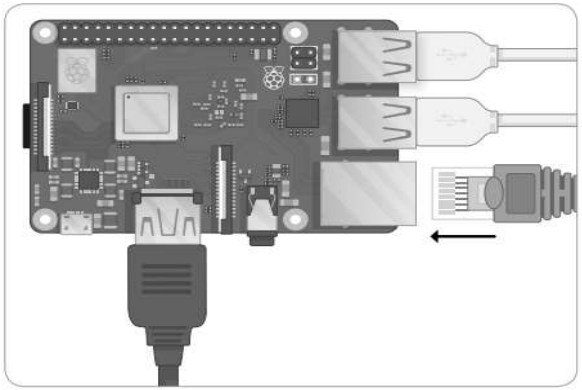


- CONNECTION TO THE TV: It doesn't imply you can't use the Raspberry Pi if your TV or monitor doesn't have an HDMI port. Adapter cables, which can be found at any electronics store, can convert the Raspberry Pi's HDMI port to DVI-D, Display Port, or VGA for use with older computer monitors; they are simply attached to the Pi's HDMI port, and then an appropriate cable is used to connect the adapter cable to the monitor. If your TV only has a composite video or SCART input, you can buy 3.5 mm tip-ring-ring-sleeve (TRRS) adapter cables and composite-to-SCART adapters to plug into the 3.5 mm AV port.

Connecting a network cable (optional)

To connect your Raspberry Pi to a wired network, insert a network cable – also known as an Ethernet cable – into the Ethernet port on the Pi, with the plastic clip facing down, until you hear a click. If the cable needs to be removed, simply squeeze the plastic clip inwards towards the plug and gently slip the cable free.

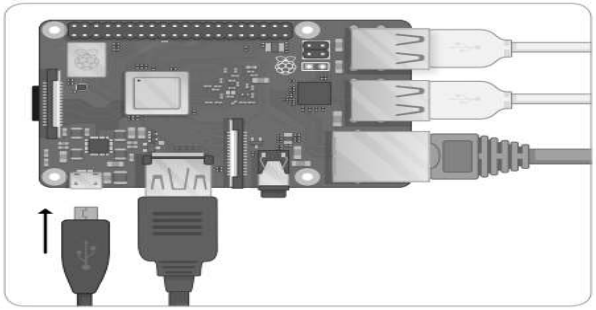
In the same way, attach the opposite end of your network cable to any free port on your network hub, switch, or router.



Connecting a power supply

The last stage in the hardware setup procedure is to connect the Raspberry Pi to a power source, which you should do only when you're ready to set up its software: the Raspberry Pi lacks a power switch and will turn on as soon as it's attached to a live power supply.

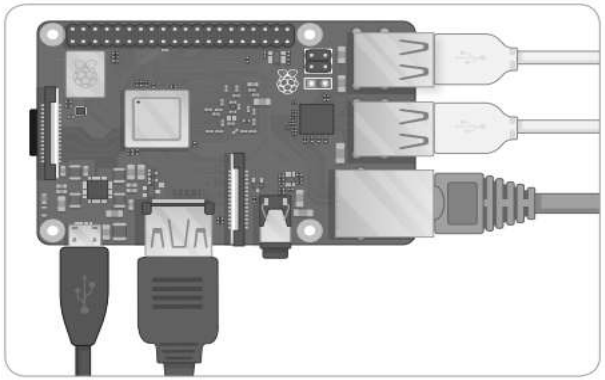
Connect the micro USB end of the power supply cable to the Raspberry Pi's micro USB power connection. It can only travel one way, with the thin part of the connector pointing down, and should softly slide home.



- POWER SUPPLY: If you're using the Official Raspberry Pi Power Supply, you'll see that it comes with multiple mains connectors that are compatible with different nations' sockets. Choose the one that corresponds to the socket type in your nation, and then slide it onto the power supply body until you hear a click.

Finally, connect the power supply to a mains socket and turn it on; the Raspberry Pi will start running instantly.

You've completed the assembly of your Raspberry Pi!



Setting up the software

You'll need to set up the Raspberry Pi's software, particularly its operating system, which regulates what the Pi can do, before you can start using it in earnest. NOOBS, or New Out-Of-Box Software, is designed to make this process as simple as possible by allowing you to choose from a variety of operating systems and have them installed automatically. Even better, you can accomplish all of this with only a few mouse clicks.

You'll see a screen with the Raspberry Pi logo on it and a small progress window at the upper-left when the Pi is initially switched on, or booted, with a fresh installation of NOOBS on its microSD card. You'll see the screen shown in Figure 1.40 after a brief wait, which might take up to a minute the first time you use the NOOBS microSD card.

Figure 1.40 The NOOBS menu without any operating system installed

- ARE THERE NO PICTURES?

  Check that you're using the correct input if you can't see the Raspberry Pi on your screen. If your TV or monitor has multiple HDMI inputs, use the ‘Source' or ‘Input' buttons to cycle through each one until you get the NOOBS menu.

  This is the NOOBS menu, which allows you to select an operating system for your Raspberry Pi. Raspbian, a version of the Debian Linux operating system customised exclusively for the Raspberry Pi, and LibreELEC, a version of the Kodi Entertainment Centre software, are supplied as standard with NOOBS. You can also download and

install different operating systems if the Pi is connected to the network - either through a wired connection or using the ‘Wifi networks (w)' option from the top bar of icons.

Use the mouse to draw a cross in the box to the left of Raspbian Full: position the pointer at the white box and click once with the left mouse button to begin installing an operating system. When you've done so, the ‘Install I menu icon will no longer be greyed-out, indicating that your operating system is ready to install (Figure 1.41)

Figure 1.41 Choosing an operating system to install through NOOBS

When you press the left mouse button on the ‘Install (i)' icon, a warning notice appears, informing you that installing the operating system would overwrite any data currently saved on the microSD card, except for NOOBS, which will remain intact. The installation process will begin once you click ‘Yes' (Figure 1.42)

Figure 1.42 Installing the Raspbian operating system

Depending on the speed of your microSD card, the installation process can take anywhere from 10 to 30 minutes. Progress is presented in a bar down the bottom of the window as the operating system is installed, and you'll watch a slide show outlining some of its important features.

- **WARNING!**

It's critical that the installation isn't interrupted because doing so risks destroying the software through a process known as data corruption. While the operating system is being installed, do not remove the microSD card or unplug the power cable; if something happens to interrupt the installation, unplug the Raspberry Pi from its power supply, then press the SHIFT key on the keyboard while reconnecting the Raspberry Pi to its power supply to bring up the NOOBS menu. This is known as recovery mode, and it's a terrific way to get a Pi back into working condition when its software has been corrupted. After a successful installation, it also allows you to access the NOOBS menu, where you can reinstall the operating system or install one of the other operating systems.

When the installation is complete, a popup with a ‘OK' button will appear; click this to restart the Pi in its newly installed operating system. The boot messages will scroll up the screen (Figure 1.43), and the first time you boot into Raspbian, it may take a minute or two as it adapts to make the greatest use of the free space on your microSD card. Things will move more rapidly the next time you boot.

Figure 1.43 The Raspbian boot messages

Finally, before the Raspbian desktop and setup wizard display, you'll see a window with the Raspberry Pi logo on it, as shown in Figure 1.44. Your operating system has now completed installation and is ready for configuration.

Figure 1.44 The Raspbian desktop

1.6 RASPBERRY PI BOOT

There is no BIOS on the Raspberry Pi. Your SDCard contains GPU firmware that allows you to use the GPU. The GPU kicks off the ARM processor and loads the Linux kernel. Hundreds of documents detailing how that procedure works can be found on the internet.

On the Raspberry Pi, Android is a NON-STARTER.

Windows 10 IoT isn't the same as the Windows you're used to (and despise) on your laptop. It's a specialized Windows internet of things system that only operates on an RPI2.

1.6.1 Learn how this small SoC boots without BIOS

Instead of BIOS, the Raspberry Pi uses "firmware." To add to the confusion, all B models require this firmware to be installed on the SD card. You won't even get error messages if your SD card isn't working or if you neglect to put the firmware on it. The Raspberry Pi will do nothing. The simplest method for dual booting a Pi is to use different SD cards. The SD card functions similarly to a hard disc on a desktop or laptop computer; swapping it, however, allows you to use a different operating system. It's similar to a Gameboy cartridge. The GPU handles everything, after which the kernel is loaded and the CPU is turned on.

1.6.2 Configuring boot sequences and hardware

To begin, you must understand that the Raspberry Pi does not operate in the same way as a traditional desktop computer. The graphics processor starts up before the ARM processor!

Here's a diagram of the Raspberry Pi with the Broadcom BCM2835 SoC highlighted before we dive into the details.

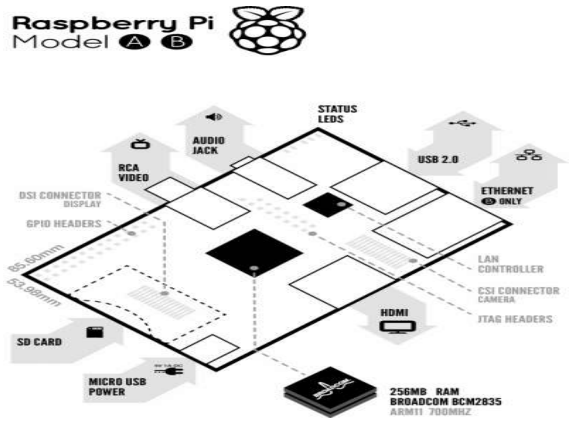


Figure 1.45 Raspberry Pi with the Broadcom BCM2835 SoC

The ARM CPU, VideoCore Graphics Processor, ROM (Read-Only Memory) chips, SDRAM, and other components are all found on the SoC (or System-on-Chip). Consider a SoC to be a combination of your motherboard and CPU crammed into a single chip.

The initial bits of code that run when you turn on your Raspberry Pi are saved in a ROM chip in the SoC that was integrated into the Pi during manufacturing! The first-stage bootloader is what it's called. On startup, the SoC is hardwired to run this code on a tiny RISC Core (Reduced Instruction

Set Computer). It's utilized to access the second-stage bootloader by mounting the FAT32 boot partition on your SDCard. So, what exactly is this SD Card's "second-stage bootloader"? It's called 'bootcode.bin'. If you had mounted the SD Card in Windows, you might have seen this file. Here's when it gets tricky. Your ARM CPU (which is in reset) and RAM have not yet been initialized by the first-stage bootloader. As a result, the second-stage bootloader must operate on the GPU as well. The bootloader.bin file is loaded and executed from the GPU's 128K 4 way set associative L2 cache. This activates the RAM and loads start.elf from your SD Card. This is the most significant of the three-stage bootloaders. It's the GPU's firmware, which means it has the settings or, in our case, instructions for loading the settings from config.txt on the SD Card. The config.txt file can be thought of as the 'BIOS settings' (as is mentioned in the forum). The following are some of the options available to you:

- **arm_freq**: It is the ARM frequency in MHz. Default frequency is 700MHz.
- **gpu_freq**: This sets core_freq, h264_freq, isp_freq, v3d_freq together.
- **core_freq**: It is the GPU processor core frequency and is expressed in MHz. Default frequency is 250MHz.
- **h264_freq**: This is the hardware video block frequency in MHz. Default is 250MHz.
- **isp_freq:** It illustrates the image sensor pipeline block frequency in MHz. Default is 250MHz.
- **v3d_freq:** It describes the frequency of 3D block in MHz. Default is 250MHz.
- **sdram_freq:** This is the SDRAM frequency expressed in MHz. Default is 400MHz.

In addition, the start.elf divides the RAM between your GPU and the ARM CPU. Only the address space left over from the GPU address space is accessible to the ARM. The MMU (Memory Management Unit) of the VideoCore maps the physical addresses detected by the ARM core to another address in the VideoCore (0xC0000000 and beyond). Because the config.txt is loaded after the split, you can't specify the dividing amounts there. However, the SD Card has a variety of .elf files with various splits. You can rename those files to start.elf and boot the Pi, according on your needs. The GPU wins every time on the Raspberry Pi!

The start.elf additionally loads cmdline.txt if it exists, in addition to loading config.txt and splitting RAM. It specifies the command line parameters for the kernel to be loaded. The boot process has now reached its end. The commencement .elf loads kernel.img, the binary file containing the OS kernel (DUH!?) and relieves the CPU reset. The ARM CPU then executes the kernel.img instructions, which loads the operating system.

Here's how it goes:

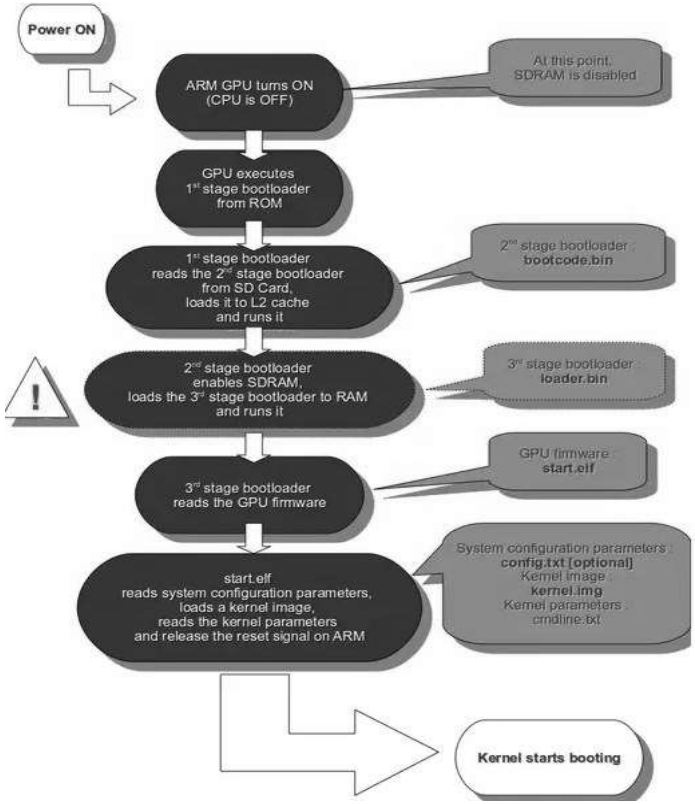


Figure 1.46 Flow process of boot sequence configuration

So the Raspberry Pi, unlike a PC, does not require a BIOS sequence because the required startup functions are incorporated into the GPU.

1.7 SUMMARY

In this chapter we presented the concept and approaches used in creating a system-on-chip (SoC) based on a microprocessor core, as well as the microprocessor core itself, were introduced in this course. The reader gained a better grasp of how SoCs are built and used, as well as why current processors are designed the way they are.

The ARM images bring reality to topics that can otherwise appear ethereal to the reader who only wants to know the basic principles; the general principles reveal the logic for the ARM being as it is to the reader who wants to understand the design of the ARM.

The technical insights about the Raspberry Pi are discussed that covered up the basic introduction of Pi, various generations of Raspberry Pi models. The content acquainted the reader to set up their own operating system and can also connect the wiring and circuits directly with the Raspberry Pi board along with the discussion of onboard hardware components. Lastly, the unit concluded with the discussion on configuring the boot sequence and hardware.

1.8 LIST OF REFERENCES

1) Learning Internet of Things, Peter Waher, Packt Publishing(2015)

2) Mastering the Raspberry Pi, Warren Gay, Apress(2014)

3) Abusing the Internet of Things, Nitesh Dhanjani, O'Reilly

4) Michael J. Flynn, Wayne Luk, Computer System Design: System on Chip, John Wiley and Sons Inc. 2011, ISBN 978-0-470-64336-5

5) SystemC: From the Ground Up, 2nd Edition, D.C. Black, J Donovan, B. Bunton, A. Keist, Springer 2010, ISBN 978-0-387-69958-5.

6) On-Chip Communication Architectures, System on Chip Interconnect, S. Pascricha and N. Dutt, Morgan Kaufmann-Elsevier Publishers 2008, ISBN 978-0-12-373892-9.

7) https://www.cs.cmu.edu/afs/cs/academic/class/15462-f11/www/lec_slides/lec19.pdf

8) https://www.researchgate.net/publication/260687001_GPU_computing

9) https://cdn.iiit.ac.in/cdn/cstar.iiit.ac.in/~kkishore/GPUArchitecture.pdf

10) http://web.eecs.umich.edu/~prabal/teaching/eecs373_f12/readings/ARM_Architecture_Overview.pdf

11) https://web.sonoma.edu/users/f/farahman/sonoma/courses/es310/310_arm/lectures/Chapter_3-and-1_ARM.pdf

12) https://www.cs.unca.edu/~bruce/Fall14/360/RPi UsersGuide.pdf

13) http://meseec.ce.rit.edu/551-projects/spring2017/2-3.pdf

1.9 UNIT END EXERCISES

1) Explain the concept of SoC
2) Write a note on significance and design challenges of SoC.
3) Describe the advantages, disadvantages and applications of system on chip.
4) Write a short note on system on chip.