**BASIC CPU ARCHITECTURE**

**CPU:**

CPU – Central Processing Unit is a component of a computer responsible for executing instructions from a computer program.

**How CPU works:**

**1. Fetch –** The CPU fetches instructions from the computer’s memory. The Program counter (PC) keeps track of the memory address of the next instruction to be fetched.

**2.Decode –** The fetched instruction is decoded to determine the operation to be performed. This involves breaking down the instruction into its various components, such as the opcode (operation code) and operands.

**3.Execute –** The CPU executes the decoded instruction by performing the specified operation. This may involve arithmetic calculations, data manipulation or control flow operations

**4.Memory Access –** Some instructions require access to data stored in memory. The CPU may need to read or write data to/ from RAM (Random Access Memory) during this stage.

**5. Write Back –** If the executed instruction produces a result that needs to be stored, the CPU writes the result back to the appropriate location, which could be a register or memory.

**6.Repeat –** The cycle repeats, with the program counter being updated to point the instruction in memory. The CPU continues to fetch, decode, execute and write back instructions until the program completes or an external event occurs

**7.Registers –** The CPU has a set of registers, which are small, fast storage location directly accessible by the CPU. Registers store data that is frequently used during program execution.

**8.Control Unit –** The control Unit manages the operation of the CPU. It oversees the fetch decode execute cycle, controls of the flow of data between the CPU and other components and coordinates the operation of the entire system.

**9.ALU (Arithmetic Logic Unit) –** The ALU performs arithmetic and logic operations. It is a crucial component for executing instructions that involve calculations or decision making.

**10.Clock –** The CPU relies on a clock signal to synchronize its internal operations. The clock signal determines the rate at which the CPU processes instructions and it defines the basic unit of time for the fetch -decode- execute cycle.

**CPU Architecture:**

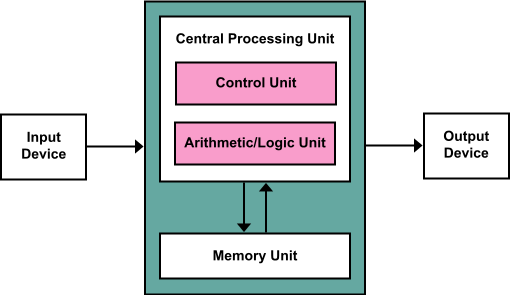


Figure: 1

1. Memory holds both data and instructions

2. The arithmetic/logic gate unit is capable of performing arithmetic and logic operations on data

3. A processor register is a quickly accessible location available to a digital processor’s processing unit. Registers usually consist of a small amount of fast storage, although some registers have specific hardware functions and may be read-only or write-only

4. The Control Unit controls the flow of data within the CPU – (which is the Fetch-Execute cycle)

5. Input arrives into a CPU via a bus

6. Output exits the CPU via a bus

**SMPS:**

SMPS stands for Switched-Mode Power Supply. It is a type of power supply that uses modern power electronics to efficiently convert electrical power. SMPS is commonly used in electronic devices, computers and various other applications.

Components of SMPS:

**Switching components** – SMPS uses switching devices such as transistors to rapidly switch the input voltage on and off. This switching action allows for efficient power conversion.

**Transformer** – Unlike traditional linear power supplies, SMPS employs a smaller and lighter transformer. The transformer operates at a high frequency, allowing for reduced size while maintaining efficiency.

**Rectification** – The input AC voltage is rectified to produce a DC voltage. This is typically done using diodes or a bridge rectifier.

**Filtering** – The rectified DC voltage is filtered to smooth out fluctuations and reduce ripple. Capacitors and inductors are used in the filtering stage.

**Switching Regulator** – The heart of the SMPS is the switching regulator , which efficiently regulates the output voltage. It controls the on-off timing of the switching components to achieve the desired output voltage.

**Output filter** – An output filter is used to further smooth the pulsed DC output from the switching regulator. This filter includes capacitors and inductors to ripple and provide a stable DC output.

**Feedback Circuit** – SMPS systems often incorporate a feedback loop to continuously monitor the output voltage and adjust the switching components accordingly. This helps maintain a stable and regulated output.

**Control Circuit** – The Control Circuit manages the operation of the switching components, ensuring that the output voltage remains within the desired range.

**Protection Features --** SMPS may include protection features, such as overcurrent protection , overvoltage protection and short circuit protection , to prevent damage to the power supply and connected devices.

**BIOS:**

BIOS – Basic Input/Output System.

It is a firmware, or low-level software, that provides essential functions for a computer’s hardware and operating system. BIOS is stored on a non-volatile memory chip on the computer’s motherboard, and it is typically the first software that runs when the computer is powered on or restarted.

**POST – Power-On-Self-Test**

When you turn a computer , the BIOS begins the boot process by performing a Power on Self Test . During POST, the BIOS checks the integrity of essential hardware components such as the CPU , memory (RAM). Graphics card , storage devices and other peripherals.

MBR is commonly used on systems with BIOS firmware.

**MBR:**

MBR stands for Master Boot Record and it refers to a type of partitioning scheme used on storage devices such as hard drives and solid-state drives. The MBR contains information about how the partitions on a storage device are organized and where the bootloader, a small program that initiates the operation system, can be found.

MBR uses a sector-based addressing system. Each sector is typically 512 bytes in size. The MBR itself is located in the first sector (the boot sector) of the storage device.

Bootloader (446 bytes) – The first 446 bytes of the MBR are usually allocated for the bootloader code

Partition Table (64 bytes) – Following the bootloader code, the next 64 bytes are allocated for the partition table. The partition table contains information about the disk’s partitions, including their starting and ending sectors, partition type and other details. Each entry in the partition table is 16 bytes, and there can be up to four entries in a standard MBR partition table.

MBR signature (2 bytes) – The last 2 bytes of the MBR are reserved for the MBR signature, also known as the “magic number”. These bytes have fixed values(0\*55AA) and serve as a marker to indicate that the MBR is valid.

The Master Boot Record(MBR) partitioning scheme has limitations that affect its ability to support storage devices larger than 2 terabytes.

**UEFI:**

**UEFI –** Unified Extensible Firmware Interface. It is a modern firmware interface that has largely replaced the traditional BIOS (Basic Input/ Output System).

Key Features:

Boot Process: UEFI provides a more flexible and feature-rich boot process compared to the traditional BIOS.

GUI: UEFI firmware often includes a graphical user interface. This allows users to interact with system settings using a mouse and provides a more user-friendly experience.

Security Features: UEFI includes security features such as Secure Boot. Which helps prevent the loading of unauthorized or malicious code during the boot process. Secure Boot verifies the digital signatures of the boot loaders and operating system components, enhancing system security.

Partitioning Schemes: UEFI supports both MBR and GPT partitioning schemes. GPT (GUID Partition Table) is the preferred choice for UEFI systems due to its ability to support larger disk sizes and a greater number of partitions.

Networking and Internet Support: UEFI firmware can include networking capabilities, allowing for features such as remote diagnostics, firmware updates over the network, and other network-based services.

**GUID Partition Table (GPT):**

The GUID Partition Table is a partitioning scheme commonly associated with the unified Extensible Firmware Interface.

The GPT uses globally unique identifiers to define partitions on a storage device.

GPT allows for a larger number of partitions compared to MBR. Each partition entry include information about the partition’s size, start and end sectors and attributes.

GPT uses a 64-bit addressing scheme for sector addresses. This allows GPT to support large number of sectors – 18.4 million TB (terabytes).

GPT can support a larger number of partitions compared to MBR. It also supports different file systems , such as FAT32 , NTFS , exFAT and various Linux file systems.

GPT is often associated with UEFI(Unified Extensible Firmware Interface) firmware. While GPT can be used on systems with both UEFI and

**GRUB:**

**GRUB –** Grand Unified Bootloader, is a widely used boot loader in the Linux and Unix-like operating systems. Its primary function is to manage the boot process of the computer by loading the operating system kernel into memory.

Key Features of GRUB:

Multiboot Support:

GRUB supports multiboot configurations , allowing users to install and boot multiple operating systems on the same computer.

Boot Menu:

GRUB presents a menu during the boot process that allows users to choose which operating system or kernel configuration they want to boot.

Configuration Files:

GRUB uses configuration files to define how the boot process should be handled. The main configuration file is typically located at ‘/boot/grub/grub.cfg’ in linux systems.

**Kernel:**

Kernel refers to the core component of an operating system. It is the central part that manages system resources, facilitates communication between software and hardware and provides essential services to other parts of the operating system and user applications. The kernel acts as an intermediary between application programs and the computer hardware.

Key aspects:

I)Resource Management

II)Process Management

III)Memory Management

IV)Device Drivers

V)File System Management

**LIB (Libraries):**

Libraries are compiled collections of code that can be linked with a program to provide additional functionality. These libraries often have file extensions like ‘.lib’ on windows or ‘.a’ on Unix-Like systems.

**SHELL:**

A shell is a user interface that provides a way for users to interact with an operating system and execute commands. It acts as a bridge between the user and the underlying system, facilitating communication and control. Shells can be either command-line interfaces (CLI) or graphical user interfaces, depending on the user’s preference and the requirements of the system.

**Key aspects;**

CLI – Command Line Interface

GUI – Graphical User Interface

Examples: Bash (Bourne Again Shell), PowerShell, Zsh (Z Shell) and the command prompt on windows. Each shell may have its own features, syntax and scripting language

**SHELL architecture:**

**Input UI ----- user interface Output**

**| |**

**Shell ---- Bash (stored in bin)**

**| |**

**Kernel --- OS**

**| |**

**Hardware --- Binaries (0s and 1s)**

Shell architecture refers to the design and structure of a shell, which is a user interface that allows users to interact with an operating system or software application. Shells are typically command-line interfaces or graphical user interfaces that enable users to communicate with the underlying system, execute commands and manage files and processes.

Command Interpreter – The core of a shell is the command interpreter, responsible for interpreting and executing user commands. It takes user input, interprets the commands and communicates with the operating system to perform the requested actions.

Parser – The parser is a component of the shell that analyzes the structure of user input. It breaks down commands into individual components, such as command names, options and arguments, to prepare them for execution.

Command execution – The shell architecture includes mechanisms for executing commands. This may involve launching new processes, managing their execution and handling input/output redirection.

Scripting and Programming: Shells often support scripting and programming capabilities. Users can write scripts or programs using shell scripting languages to automate tasks, create functions and enhance the functionality of the shell.

Built in commands: Shells typically include a set of built in commands that are executed directly by the shell, without launching external processes. These built in commands often include basic file and process management operations.

Command History: Many shells maintain a command history allowing users to recall and re-execute previous commands.

**Run Levels:**

**1)Run Level 0**: Target – power-off or shutdown or halt. The system is brought to a state where it is safe to turn off the power.

**2)Run Level 1:** Target – rescue or emergency or trouble shoot or single user mode. Often used for system maintenance or recovery tasks, providing a minimal environment with only essential services.

**3)Run Level 2:** Target – multi-user. Multi users with networking. Typically used for normal system operation with networking capabilities.

**4)Run Level 3:** Target – multi user. Multi users without networking. Similar to Run level 2 but without network services.

**5)Run Level 4:** Target – unused. Historically, Run level 4 was left unused and available for local administrators to define their own custom run levels.

**6)Run Level 5:** Target – GUI mode. The system starts in a mode with a graphical login screen and users can login using graphical credentials

**7)Run Level 6:** reboot or restart, The system is restarted and the operating system goes through the boot process again.

**init –** the init system is responsible for initializing the system and managing its state during boot and shutdown**.**

**init.d –** The init.d directory often located at ‘/etc/init.d/’, is a directory that contains system initialization scripts. These scripts are responsible for starting, stopping and restarting system services during the boot process or when the system changes run levels.

**systemd --** Systemd is a system and service manager for linux operating systems. It is designed to be a replacement for the traditional System V init system and provides a range of features for managing system services, processes and other aspects of the system.

**systemctl –** systemctl is a command utility in Linux systems that is used for controlling the system system and service manager. Systemd is the init system and service manager for many modern Linux distributions. ‘systemctl’ allows users to interact with and control various aspects of the system system, including services, targets, sockets and more.

It is used to start a service, stop a service, restart a service, enable a service, Disable a service, check service status, List All services, List all targets, switch to a target, show service configuration, check journal logs for a service.

**Cores:**

A core is an independent processing unit within a CPU. It can execute its own set of instructions and operate on data independently of other cores.

**Multicore Processors:** Modern CPUs often have multiple cores on a single chip. For example, a dual-core processor has two cores, a quad-core processor has four cores, and so on.

**Parallel Processing:** Cores allow for parallel processing, meaning multiple tasks can be executed simultaneously. Each core can work on a separate task, improving overall system performance.

**Threads:**

A thread is a sequence of instructions that can be scheduled and executed by a core. Each core can handle multiple threads, allowing for concurrent execution of tasks.

**Simultaneous Multithreading (SMT):** SMT is a technology that enables a single core to execute multiple threads simultaneously. Each thread is scheduled and processed independently, sharing the resources of the core.

**Hyper-Threading (HT):** Hyper-Threading is Intel's implementation of SMT. CPUs with Hyper-Threading appear to the operating system as having double the number of logical cores. For example, a quad-core processor with Hyper-Threading may present itself as having eight logical cores.

**Benefits of Threads:** Threads can improve overall system responsiveness and efficiency by allowing a CPU to switch between tasks quickly. Multithreading is particularly useful in applications with parallelizable workloads.

**Relationship Between Cores and Threads:**

Cores vs. Threads: Cores and threads are related but distinct concepts. A multicore processor may have multiple cores, and each core may support multiple threads through SMT or HT.

Multithreaded Applications: Applications can be designed to take advantage of multiple threads, distributing tasks across available cores. This is beneficial for performance in scenarios where tasks can be parallelized.

Resource Sharing: Cores within a CPU share certain resources, such as cache memory and the memory bus. Threads running on the same core share these resources, and the efficient use of these resources is crucial for optimal performance.

**Examples of cores and threads:**

**Intel Processors:**

**Intel Core i3:**

I) Example: Intel Core i3-13100 (4 cores, 8 threads)

II) Latest features: Improved IPC (Instructions per Clock), support for DDR5 memory, integrated Intel UHD Graphics 770.

**Intel Core i5:**

I)Example: Intel Core i5-14600K (6 cores, 12 threads)

II)Latest features: Higher clock speeds, Hyper-Threading support, better overclocking capabilities.

**Intel Core i7:**

I)Example: Intel Core i7-13700K (8 cores, 16 threads)

II)Latest features: More L3 cache, enhanced integrated graphics (Intel UHD Graphics 770), AI acceleration features.

**Intel Core i9:**

I)Example: Intel Core i9-13900K (24 cores, 32 threads)

II)Latest features: Flagship performance, maximum cores and threads, DDR5/DDR4 memory support, robust overclocking potential.

**Intel Xeon:**

Example: Intel Xeon W-3375 (16 cores, 32 threads)

Latest features: Designed for workstations, higher memory capacity, ECC memory support, security and reliability enhancements.

**AMD Processors:**

**AMD Ryzen 3:**

I)Example: AMD Ryzen 3 7600 (6 cores, 12 threads)

II)Latest features: Increased core count, higher clock speeds, improved integrated graphics (AMD Radeon Graphics).

**AMD Ryzen 5:**

I)Example: AMD Ryzen 5 7600X (6 cores, 12 threads)

II)Latest features: Mid-range performance sweet spot, better overclocking support, DDR5/DDR4 memory compatibility.

**AMD Ryzen 7:**

I)Example: AMD Ryzen 7 7800X (8 cores, 16 threads)

II)Latest features: Excellent multi-threading performance, ideal for gaming and content creation, higher cache capacity.

**AMD Ryzen 9:**

I)Example: AMD Ryzen 9 7950X (16 cores, 32 threads)

II)Latest features: Top-of-the-line performance for demanding workloads, 3D V-Cache technology for enhanced gaming performance, DDR5/DDR4 memory support.

**AMD EPYC:**

I)Example: AMD EPYC 7773X (64 cores, 128 threads)

II)Latest features: Designed for server environments, high core count and thread count, robust memory capacity, security and virtualization features.

**RUFUS:**

RUFUS is a popular and widely used open-source software tool designed to create bootable USB drives. It is commonly used for creating bootable USB flash drivers to install or run operating systems, such as windows, Linux or other tools that require a bootable medium.

Day – 1

27/12/2023

Wednesday