Module I

Ancient Indian Contributions to Computer Architecture

The roots of modern computing—specifically binary logic and formal language theory—can be traced back to ancient Indian scholars.

1. Acharya Pingala and the Binary System (3rd Century BCE)

Long before Gottfried Leibniz formalized the binary system in the 17th century, Acharya Pingala described it in his treatise **Chandaśśāstra**, which focused on the mathematics of poetic meters (prosody).

A. The Logic of Laghu and Guru

Pingala categorized syllables into two states:

- **Laghu (L):** Short/Light syllable (Numerical value **0**).
- **Guru (G):** Long/Heavy syllable (Numerical value **1**).

By using these two states, Pingala could represent any poetic meter as a sequence of 0s and 1s. This is the fundamental basis of **Digital Logic** and **Computer Architecture**, where transistors represent information through "on" (1) and "off" (0) states.

B. The Prastara (Permutation Table)

Pingala developed an algorithm called **Prastara** to list all possible combinations of L and G for a given length n. For a 3-syllable meter, he would generate 2^3 = 8 combinations. This is exactly how we construct **Truth Tables** in digital electronics today.

C. Conversion Algorithms (Nashtam and Uddishtam)

Pingala provided mathematical rules to convert between the rank of a meter and its binary representation:

- **Uddishtam:** Equivalent to **Binary-to-Decimal** conversion.
- **Nashtam:** Equivalent to **Decimal-to-Binary** conversion.

D. Meru Prastara (Pascal’s Triangle)

Pingala described a diagrammatic representation of these combinations, later known as **Meru Prastara**. It provides the coefficients for binomial expansions and helps determine how many combinations have a specific number of Gurus or Laghus. In modern computing, this is used in **Probability Theory** and **Error-Detection Algorithms**.

2. Panini and Formal Language Theory (4th Century BCE)

If Pingala laid the hardware foundation (Binary), Panini laid the software foundation (Grammar and Logic). His work, the **Astadhyayi**, is a complete, descriptive, and generative grammar of the Sanskrit language.

A. The Astadhyayi as a Compiler

Panini's grammar consists of approximately 4,000 rules (**Sutras**) that function like a modern computer program. Given a root word and a suffix, the Sutras act as a **Compiler** to process the input and produce a grammatically correct word.

B. Panini-Backus Form (PBF)

The **Backus-Naur Form (BNF)** is a notation technique used to describe the syntax of programming languages (like C, Java, or Python). Computer scientists have noted that Panini's notation was so mathematically precise that it is virtually identical to BNF. This is why some scholars refer to it as "Panini-Backus Form."

C. Use of Meta-language and Symbols

Panini used a "Meta-language" to describe Sanskrit. He employed:

- **Auxiliary markers (Anubandhas):** Similar to "Tags" or "Flags" in programming.
- **Recursion:** Rules that refer back to themselves to generate complex structures.
- **Operator Precedence:** Rules on which Sutra takes priority when two rules apply simultaneously.

D. The Shiva Sutras and Data Compression

The **Shiva Sutras** organized Sanskrit phonemes in a way that allowed Panini to create **Pratyaharas** (short-hand codes). For instance, the code "HAL" represents all consonants. This is a primitive yet sophisticated form of **Symbolic Logic** and **Data Compression**, where a small string represents a large set of data points.

3. Summary of Relevance to Computer Science

1. **Binary Foundation:** Pingala proved that complex patterns can be mapped to a two-state (binary) system.
2. **Algorithmic Thinking:** The Prastyayas (algorithms) for conversion show that ancient Indians were thinking in terms of step-by-step computational logic.
3. **Syntactic Precision:** Panini showed that natural language could be treated as a formal system, which is the cornerstone of **Natural Language Processing (NLP)** and **Artificial Intelligence** today.

Feature	**Indian Knowledge System**	**Computer Architecture Equivalent**
Data Unit	Laghu / Guru	Bit (0 / 1)
Logic Gate	Combinatorics of Meters	Logic Gates (AND, OR, NOT)
Syntax	Paninian Sutras	Programming Language Syntax (BNF)
Storage	Pratyahara (Short-codes)	Pointers / Hashing

Modern Contribution in Computer System

1. **Dr Vinod Dham**

- Dr. Vinod Dham is a legendary figure in the semiconductor industry, widely celebrated as the "Father of the Pentium Chip." His work at Intel during the late 1980s and early 1990s revolutionized personal computing by moving processors from simple calculators to high-performance multimedia engines.

1. Family and Early Life: The 8 Dream

Dr. Vinod Dham's journey is a classic example of the "Immigrant Success Story" that defined Silicon Valley.

- **Background:** Born in 1950 in Pune, India. His family had moved from Rawalpindi (now in Pakistan) to India during the 1947 Partition.
- **Education:** He completed his B.E. in Electrical Engineering from **Delhi College of Engineering** (now Delhi Technological University) in 1971.
- **The Struggle:** At age 25, he left for the US to pursue an MS in Physics (Solid State) at the University of Cincinnati. He famously arrived in America with only **$8 in his pocket**, relying on a research assistantship to fund his studies.
- **Personal Life:** He is married to **Sadhana Dham**, and they have two sons. Despite his high-tech career, his favorite hobby is surprisingly grounded: **carpentry**.

2. Recognition and Major Awards

Dr. Dham's contributions have been recognized by both the Indian government and the global tech industry.

A. Padma Bhushan (2025)

In a major recent recognition of his lifelong service to Science and Engineering, the Government of India conferred the **Padma Bhushan**—the third-highest civilian award—upon Dr. Dham in **January 2025**. This award highlights his role in the "Pentium Revolution" and his current efforts to build India's semiconductor ecosystem.

B. "Father of the Pentium"

This is not just a nickname but a formal recognition by the industry. He led the "P5" project at Intel, managing a team of hundreds of engineers to create the world's most successful microprocessor line.

C. Smithsonian Institution Honor

Dham's work was featured in an exhibition at the **National Museum of Natural History (Smithsonian)** titled *"Beyond Bollywood."* He was celebrated as one of the key Indian-Americans who "shaped America" by driving the computing revolution that fueled the US economy in the 90s.

D. Industry Rankings

- **Top 25 Executives:** He was named one of the top 25 executives in the entire computer industry during the height of the Pentium's success.
- **Top 100 Most Influential Asian Americans:** Recognized for his impact over a decade of technological leadership.

3. Professional Achievements & Milestones

Milestone	Significance
Intel Career (16 years)	Co-inventor of Flash Memory (ETOX) and leader of 386, 486, and Pentium teams.
NexGen/AMD Merger	Orchestrated the $857 million merger that created the AMD K6, bringing competition to the CPU market.
Silicon Spice	Developed the architecture for VOIP (Voice over IP) chips; sold to Broadcom for $1.2 billion.
Venture Capital (IUVP)	Founded **Indo-US Venture Partners**, focusing on early-stage startups to boost the Indian tech ecosystem.
India Semiconductor	Served as an advisor to the Government of India, helping

Milestone	Significance
Mission	draft policies for domestic chip manufacturing.

4. Academic and Social Impact

- **Honorary Distinguished Professor:** Recently appointed at his alma mater, **DTU**, where he is helping establish a "Center of Excellence in Semiconductors."
- **William Tuft Medal:** Awarded by the University of Cincinnati for exceptional professional achievement.
- **Presidential Advisor:** He served as an advisor to **President Bill Clinton** on the "President's Advisory Commission on Asian Americans and Pacific Islanders."

Dr. Dham’s achievements represent a **bridge between India and Silicon Valley**. His life shows that Indian engineers were not just "users" of modern computer architecture, but the primary **designers** who built the hardware foundations of the 21st century. His recent **Padma Bhushan** (2025) serves as a perfect concluding point to show his continued relevance in the era of AI and domestic chip manufacturing.

2. Dr Ajay Bhat

Dr. Ajay Bhatt is another legendary Indian-American computer architect whose contribution is likely used by you every single day: he is the **Father of the USB (Universal Serial Bus)**.

Ajay Bhatt represents the "Input/Output (I/O) Architecture" side of computing, complementing the processor architecture of Vinod Dham.

1. Early Life and Roots in Gujarat

Ajay Bhatt was born in **1957** in India. His family background is rooted in the academic and professional middle class of Gujarat, a state known for its entrepreneurial and mathematical heritage.

- **Educational Environment:** He grew up in an environment that prioritized technical education. He attended the **Maharaja Sayajirao University of Baroda (MSU)**, one of India's oldest and most prestigious institutions for engineering.
- **The "MSU" Foundation:** At MSU, he completed his Bachelor's degree in Electrical Engineering. It was here that he developed the fundamental understanding of **Circuit Theory** and **Logic Design** that would later allow him to envision a "Universal" bus for data transfer.

2. Migration and Higher Studies in the US

Like many pioneers of his generation (including Vinod Dham), Bhatt moved to the United States in the late 1970s to pursue the "Silicon Valley Dream."

- **The CUNY Years:** He enrolled at the **City University of New York (CUNY)** for his Master's degree.
- **Financial Struggle:** Coming from a modest Indian background, he faced the typical challenges of an international student, balancing rigorous academic research with the need to establish a career in a foreign land.
- **Marriage and Family:** Ajay Bhatt is a family-oriented man. He is married to **Sangeeta Bhatt**, and they have two daughters. Interestingly, it was a **family problem** that led to the invention of the USB.

3. The Visionary Behind USB (Universal Serial Bus)

Before the mid-1990s, connecting a device to a computer was a nightmare. You had separate, bulky ports for everything: a serial port for the mouse, a parallel port for the printer, a DIN port for the keyboard, and often unique cards for joysticks or scanners.

A. The "Problem" He Solved

Most ports back then were not **"Plug and Play."** You had to turn off the computer, plug in the device, and often manually configure "jumpers" or "IRQ settings" (Interrupt Requests).

B. The USB Architecture

Working at **Intel** in the early 90s, Bhatt led the team that created the USB standard. His architectural goals were:

- **Universal:** One port for all devices.
- **Hot-Swappable:** The ability to plug and unplug devices while the computer is running.
- **Self-Powering:** The cable should carry both data and electricity (eliminating many power bricks).

4. Major Technical Contributions

Beyond the USB, Ajay Bhatt holds over **130 U.S. and international patents**. His work defines how data moves *inside* and *outside* the computer.

- **PCI Express (PCIe):** He was a key architect in developing the PCIe standard. This is the high-speed interface used to connect Graphics Cards (GPUs) and NVMe SSDs to the motherboard.
- **AGP (Accelerated Graphics Port):** Before PCIe, he helped create AGP, which revolutionized 3D gaming by giving the graphics card a direct "high-speed lane" to the CPU.
- **Platform Power Management:** He developed architectures that allow laptops to "sleep" and "wake up" efficiently, significantly extending battery life.

5. Awards and Global Recognition

Ajay Bhatt’s impact is so massive that he even became a "pop culture" icon for engineers.

- **The "Intel Rockstar" Commercial:** In 2009, Intel ran a famous TV ad where Ajay Bhatt was treated like a rockstar, with fans screaming as he walked through the office. This was a cultural moment that celebrated engineers as the true heroes of the modern world.
- **European Inventor Award (2013):** He won this prestigious award in the "Non-European Countries" category for his work on the USB.
- **Light of India Award (2012):** Recognized for his excellence in Science and Technology.
- **Outstanding Achievement in Science & Technology:** Awarded at the Asian Awards in London (2013).

6. Summary Table: Ajay Bhatt vs. Modern Computing

Feature	**Pre-Bhatt Era**	**Post-Bhatt (USB/PCIe) Era**
Connectivity	Multiple incompatible ports	Single Universal Port (USB)
Device Setup	Restart required (Cold-plug)	Plug & Play (Hot-swappable)
Data Speed	Slow Serial/Parallel speeds	High-speed PCIe (GB/s)
Power	Separate power adapters	Power over USB (Charging)

3. **Dr Vinod Khosla**

Dr. Vinod Khosla (born January 28, 1955) is an Indian-American billionaire businessman, venture capitalist, and engineer. He is one of the most influential figures in Silicon Valley history, recognized primarily as a co-founder of Sun Microsystems and the founder of Khosla Ventures.

1. Introduction: The Architect of Networking

Dr. Vinod Khosla (born 1955) is an Indian-American engineer and entrepreneur best known as the **co-founder of Sun Microsystems**. His most significant contribution to computer science is the philosophy: **"The Network is the Computer."** This idea moved computing away from isolated desktop boxes toward the interconnected, cloud-based world we live in today.

2. Family Background and Early Life

- **Roots:** Born into a Punjabi family in Delhi. His father was an officer in the **Indian Army**, which instilled in him a sense of discipline and rigorous logic.
- **Education (The Triple Threat):** * **IIT Delhi:** He earned his B.Tech in Electrical Engineering.
    - **Carnegie Mellon University:** He obtained an MS in Biomedical Engineering.
    - **Stanford University:** He earned an MBA, which allowed him to bridge the gap between complex engineering and market-ready products.
- **Personal Life:** He is married to **Neeru Khosla**. Together, they are major philanthropists. Neeru co-founded **CK-12**, a non-profit that creates open-source high-school textbooks to make education affordable globally.

3. Major Professional Achievements

A. Founding Sun Microsystems (1982)

Khosla co-founded Sun (Stanford University Network) with the vision of building high-performance workstations.

- **The Workstation Architecture:** Before Sun, computers were either "Mainframes" (huge and expensive) or "PCs" (weak). Khosla's team built the **Sun Workstation**, which used **Unix** and **RISC processors**, giving engineers massive power on their desks.
- **NFS (Network File System):** Under his leadership, Sun developed NFS, which allowed computers to share files over a network as if they were stored locally—the fundamental precursor to **Cloud Storage**.

B. SPARC Architecture (Reduced Instruction Set Computing)

Khosla pushed for the development of **SPARC** (Scalable Processor Architecture).

- **Technical Impact:** Unlike Intel's Pentium (CISC), SPARC was a **RISC** architecture. It simplified the instruction set to make the processor run much faster for scientific and server-based tasks.

C. The Birth of Java

Though developed by James Gosling at Sun, Khosla provided the strategic push for **Java**. Java's "Write Once, Run Anywhere" (WORA) architecture was designed to allow code to run on any networked device, from a toaster to a supercomputer.

4. Recognition and Awards

Dr. Khosla is recognized as one of the most influential "Tech Visionaries" in history.

- **Padma Bhushan (2025):** In January 2025, the Government of India conferred the **Padma Bhushan** on Dr. Khosla for his monumental contribution to Trade and Industry and for putting Indian engineering on the global map.
- **Silicon Valley Hall of Fame:** Inducted for his role in creating the workstation and server industry.
- **Forbes 400:** Consistently ranked as one of the wealthiest and most influential people in technology.

- **Lifetime Achievement Award:** Given by the **TiE (The Indus Entrepreneurs)**, an organization he helped build to mentor young Indian founders.

5. Current Impact: AI and Future Architecture

Through his firm, **Khosla Ventures**, he is now the architect of the "AI Era."

- **OpenAI:** He was the first venture capitalist to invest in **OpenAI** (the creators of ChatGPT). He predicted that AI would become the new "operating system" for the world.
- **Clean Tech:** He has spent billions of dollars on "Green Architecture," trying to use technology to solve climate change.

6. Summary

Aspect	**Contribution**
Foundational Concept	"The Network is the Computer"
System Contribution	Developed **Workstations** and **Unix-based servers**.
Hardware Contribution	Championed **RISC / SPARC** architecture.
Software Contribution	Led the company that created **Java**.
Recent Recognition	**Padma Bhushan (2025)** by the Govt. of India.

4. Dr Vijay P Bhatkar

Dr. Vijay Pandurang Bhatkar (born October 11, 1946) is a preeminent Indian computer scientist, IT leader, and educationalist. He is globally celebrated as the **"Father of Indian Supercomputing"** for his pioneering role in developing the **PARAM** series of supercomputers. His work represents a landmark in Indian "Atmanirbharta" (self-reliance), proving that India could build world-class hardware despite international technology embargoes.

1. Introduction: The Architect of PARAM

In the late 1980s, the United States denied India the purchase of a Cray supercomputer (needed for critical weather forecasting) due to "dual-use" technology concerns. In response, the Government of India established **C-DAC** (Centre for Development of Advanced Computing) and appointed Dr. Bhatkar as its founding Executive Director.

- **The Mission:** He was given a budget of **₹30 crores** and a timeline of **3 years** to build a supercomputer that would be faster than the Cray.
- **The Achievement:** In 1991, his team launched the **PARAM 8000**. It was not only India's first supercomputer but was benchmarked as the second fastest in the world at the time, costing only a fraction of the American machine's price.
- **Architectural Shift:** He moved India from "Serial Computing" to **Parallel Processing Architecture**, where thousands of off-the-shelf processors work together to solve massive mathematical problems.

2. Family Background and Early Life

- **Roots:** Born in the village of **Muramba**, Akola district, Maharashtra.
- **Parental Influence:** His parents were dedicated **Gandhian freedom fighters**. His father was a principal and his mother was a headmistress. They famously chose to live in a small, remote village to educate and serve the rural population, a choice that instilled a deep sense of national service in Dr. Bhatkar.
- **Academic Journey:** * **B.E. (Electrical Engineering):** From Nagpur University (VNIT).
    - **M.E.:** From M.S. University of Baroda (the same institution that produced Ajay Bhatt).

- **Ph.D.:** From **IIT Delhi** (1972).

3. Major Achievements and Contributions

Dr. Bhatkar's impact goes far beyond hardware; he is a key architect of the modern Indian IT ecosystem.

A. Parallel Processing & High-Performance Computing (HPC)

By developing the PARAM series, he ensured that India had the computing power required for space research, nuclear modeling, and molecular biology.

B. Multilingual Computing (GIST)

He led the development of **GIST (Graphics and Intelligence based Script Technology)**. Before this, computers in India could only be used in English. GIST allowed computers to process all Indian languages, which was crucial for digital literacy and government administration in rural India.

C. Institution Building

He played a pivotal role in establishing several national-level institutions:

- **C-DAC:** Founding Director.
- **IIITM-K:** Indian Institute of Information Technology and Management, Kerala.
- **Nalanda University:** He served as the **Chancellor** of this historic international university.
- **ETH Research Lab:** Founded the "Education to Home" mission to provide low-cost computers for children.

4. Awards and Recognition

Dr. Bhatkar has received over 25 prestigious national and international awards for his service to science and society.

Award	Year	Significance
Padma Bhushan	2015	India's third-highest civilian award for Science and Engineering.
Padma Shri	2000	For his pioneering role in the Indian IT revolution.
Maharashtra Bhushan	1999	The highest civilian honor of the State of Maharashtra.
Saint Dnyaneshwar World Peace Prize	2010	For his efforts in synthesizing Science and Spirituality.
Dataquest Lifetime Achievement Award	2003	For his long-standing leadership in the IT industry.
Krishi Ratna Award	2014	For using technology to help farmers with weather forecasting.

Dr. Vijay Bhatkar’s transition from a village boy in Akola to a global leader in supercomputing is the ultimate story of Indian excellence. His legacy continues through the **PARAM Siddhi-AI**, which is currently one of the world's most powerful AI supercomputers, continuing the architecture he started over 30 years ago.

Module VI

Peripheral Devices:

A peripheral device is any external device connected to a computer that adds functionality but is not part of the core computer system. These devices typically handle input, output, or both, and are also known as input-output (I/O) devices. While not essential for a computer to perform its basic operations, peripherals enhance the user's experience by expanding the system's capabilities. Common examples include keyboards, mice, printers, and external drives. Although the term is sometimes used loosely, it generally refers to devices located outside the computer case.

Classification of Peripheral Devices

It is generally classified into four basic categories which are given below:

1. Input Devices:

An input device is defined as a device that converts incoming data and instructions into a pattern of electrical signals in binary code that are comprehensible to a digital computer.

- **Keyboard:** A keyboard is an input device that allows users to enter text and commands into a computer system.
- **Mouse:** A mouse is an input device that allows users to control the cursor on a computer screen.
- **Scanner:** A scanner is an input device that allows users to convert physical documents and images into digital files.
- **Microphone**: A microphone is an input device that allows users to record audio.

2. Output Devices:

An output device is generally the reverse of the input process and generally translates the digitized signals into a form intelligible to the user. The output device is also performed for sending data from one computer system to another. For some time punched card and paper tape readers were extensively used for input, but these have now been replaced by more efficient devices. Example:

- **Monitor:** It is an output device that displays visual information from a computer system.
- **Printer:** It is an output device that produces physical copies of documents or images.
- **Speaker:** It is an output device that produces audio.

3. Storage Devices:

Storage devices are used to store data in the system which is required for performing any operation in the system. The storage device is one of the most required devices and also provides better compatibility. Example:

- **Hard Drive:** A hard drive is a storage device that stores data and files on a computer system.
- **USB Drive:** A USB drive is a small, portable storage device that connects to a computer system to provide additional storage space.
- **Memory Card:** A memory card is a small, portable storage device that is commonly used in digital cameras and smartphones.
- **External Hard Drive:** An external hard drive is a storage device that connects to a computer system to provide additional storage space.

4. Communication Devices:

Communication devices are hardware devices that enables data exchange between computer systems or networks. These include:

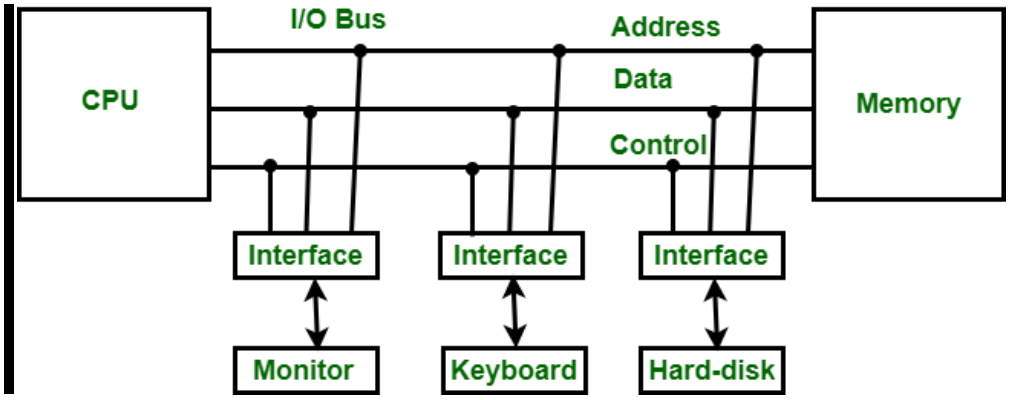
- **Modem:** A modem is a communication device that allows a computer system to connect to the internet.
- **Network Card:** A network card is a communication device that allows a computer system to connect to a network.
- **Router:** A router is a communication device that allows multiple devices to connect to a network.

- **Increased Efficiency**: Peripherals improve the overall efficiency and usability of the computer by adding specialized functions.

I/O Interface

Input Output Interface is used as a method which helps in transferring information between the internal storage devices, i.e., memory, and the external peripheral device. A peripheral device provides input and output for the computer; it is also called an Input-Output device.

For Example, A keyboard and mouse, which provide input to the computer, are called input devices while a monitor and printer provide output to the computer, are called output devices. Just like the external hard drives, there is also availability of some peripheral devices which are able to provide both input and output.



In micro-computer base system, the only purpose of peripheral devices is just to provide **special communication links** for the interfacing them with the CPU. To resolve the differences between peripheral devices and CPU, there is a special need for communication links.

The major differences are as follows:

1. The nature of peripheral devices is electromagnetic and electro-mechanical. The nature of the CPU is electronic. There is a lot of difference in the mode of operation of both peripheral devices and CPU.
2. There is also a synchronization mechanism because the data transfer rate of peripheral devices are slow than CPU.
3. In peripheral devices, data code and formats are differ from the format in the CPU and memory.
4. The operating mode of peripheral devices is different and each may be controlled so as not to disturb the operation of other peripheral devices connected to CPU.

There is a special need of the additional hardware to resolve the differences between CPU and peripheral devices to supervise and synchronize all input and output devices.

Functions of Input-Output Interface:

1. It is used to synchronize the operating speed of CPU with respect to input-output devices.
2. It selects the input-output device which is appropriate for the interpretation of the input-output signal.
3. It is capable of providing signals like control and timing signals.
4. In this data buffering can be possible through data bus.
5. There are various error detectors.
6. It converts serial data into parallel data and vice-versa.
7. It also convert digital data into analog signal and vice-versa.

Data Transfer Scheme/ Modes of Transfer

We store the binary information received through an external device in the memory unit. The information transferred from the CPU to external devices originates from the memory unit. Although the CPU processes the data, the target and source are always the memory unit. We can transfer this information using three different modes of transfer.

1. **Programmed I/O**
2. **Interrupt- initiated I/O**
3. **Direct memory access(DMA)**

1. Programmed I/O

Programmed I/O uses the I/O instructions written in the computer program. The instructions in the program initiate every data item transfer. Usually, the data transfer is from a memory and CPU register. This case requires constant monitoring by the peripheral device's CPU.

Advantages:

- Programmed I/O is simple to implement.
- It requires very little hardware support.
- CPU checks status bits periodically.

Disadvantages:

- The processor has to wait for a long time for the I/O module to be ready for either transmission or reception of data.
- The performance of the entire system is severely degraded.

2. Interrupt-initiated I/O

In the above section, we saw that the CPU is kept busy unnecessarily. We can avoid this situation by using an interrupt-driven method for data transfer. The interrupt facilities and special commands inform the interface for issuing an interrupt request signal as soon as the data is available from any device. In the meantime, the CPU can execute other programs, and the interface will keep monitoring the i/O device. Whenever it determines that the device is ready for transferring data interface initiates an interrupt request signal to the CPU. As soon as the CPU detects an external interrupt signal, it stops the program it was already

executing, branches to the service program to process the I/O transfer, and returns to the program it was initially running.

Working of CPU in terms of interrupts:

- CPU issues read command.
- It starts executing other programs.
- Check for interruptions at the end of each instruction cycle.
- On interruptions:-
  - Process interrupt by fetching data and storing it.
  - See operation system notes.
- Starts working on the program it was executing.

Advantages:

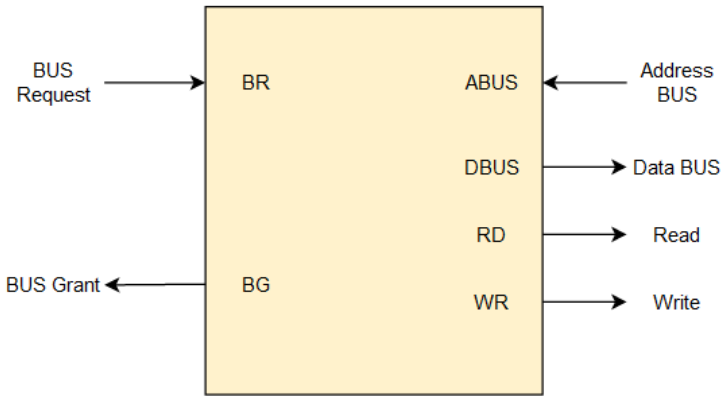
- It is faster and more efficient than Programmed I/O.
- It requires very little hardware support.
- CPU does not check status bits periodically.

Disadvantages:

- It can be tricky to implement if using a low-level language.
- It can be tough to get various pieces of work well together.
- The hardware manufacturer / OS maker usually implements it, e.g., Microsoft.

3. Direct Memory Access (DMA)

The data transfer between any fast storage media like a memory unit and a magnetic disk gets limited with the speed of the CPU. Thus it will be best to allow the peripherals to directly communicate with the storage using the memory buses by removing the intervention of the CPU. This mode of transfer of data technique is known as Direct Memory Access (DMA). During Direct Memory Access, the CPU is idle and has no control over the memory buses. The DMA controller takes over the buses and directly manages data transfer between the memory unit and I/O devices.



CPU Bus Signal for DMA transfer

Bus Request - We use bus requests in the DMA controller to ask the CPU to relinquish the control buses.
Bus Grant - CPU activates bus grant to inform the DMA controller that DMA can take control of the control buses. Once the control is taken, it can transfer data in many ways.

Types of DMA transfer using DMA controller:

- **Burst Transfer:** In this transfer, DMA will return the bus control after the complete data transfer. A register is used as a byte count, which decrements for every byte transfer, and once it becomes zero, the DMA Controller will release the control bus. When the DMA Controller operates in burst mode, the CPU is halted for the duration of the data transfer.
- **Cyclic Stealing:** It is an alternative method for data transfer in which the DMA controller will transfer one word at a time. After that, it will return the control of the buses to the CPU. The CPU operation is only delayed for one memory cycle to allow the data transfer to "steal" one memory cycle.

Advantages

- It is faster in data transfer without the involvement of the CPU.
- It improves overall system performance and reduces CPU workload.
- It deals with large data transfers, such as multimedia and files.

Disadvantages

- It is costly and complex hardware.
- It has limited control over the data transfer process.
- Risk of data conflicts between CPU and DMA.

Input/Output Processor: (I/O Processor)

An input-output processor (IOP) is a processor with direct memory access capability. In this, the computer system is divided into a memory unit and number of processors.

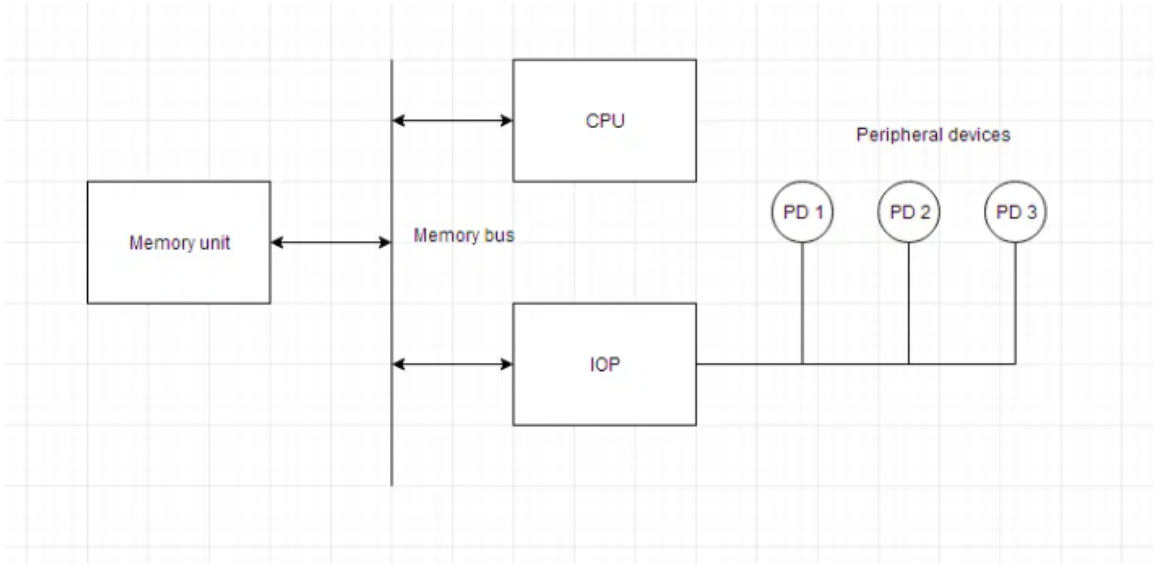
Each IOP controls and manage the input-output tasks. The IOP is similar to CPU except that it handles only the details of I/O processing. The IOP can fetch and execute its own instructions. These IOP instructions are designed to manage I/O transfers only.

Block Diagram Of I/O Processor

Below is a block diagram of a computer along with various I/O Processors. The memory unit occupies the central position and can communicate with each processor.

The CPU processes the data required for solving the computational tasks. The IOP provides a path for transfer of data between peripherals and memory. The CPU assigns the task of initiating the I/O program.

The IOP operates independent from CPU and transfer data between peripherals and memory.



The communication between the IOP and the devices is similar to the program control method of transfer. And the communication with the memory is similar to the direct memory access method.

In large scale computers, each processor is independent of other processors and any processor can initiate the operation.

The CPU can act as master and the IOP act as slave processor. The CPU assigns the task of initiating operations but it is the IOP, who executes the instructions, and not the CPU. CPU instructions provide operations to start an I/O transfer. The IOP asks for CPU through interrupt.

Instructions that are read from memory by an IOP are also called *commands* to distinguish them from instructions that are read by CPU. Commands are prepared by programmers and are stored in memory. Command words make the program for IOP. CPU informs the IOP where to find the commands in memory.

Processor vs Memory Speed

1. Fundamental Definitions

- **Processor Speed (CPU Speed):** This refers to how many calculation cycles a CPU can perform per second, measured in **Gigahertz (GHz)**. It represents the "thinking" speed of the computer.
- **Memory Speed (RAM Speed):** This refers to how quickly data can be transferred between the RAM and the CPU, measured in **Megatransfers per second (MT/s)** or **Megahertz (MHz)**. It represents the "delivery" speed of information.

2. The "Memory Wall" (The Speed Gap)

The core problem in modern computing is that processors have evolved much faster than memory.

- **CPU Cycles:** A modern 2026 processor might run at **5.5 GHz**, meaning it ticks once every **0.18 nanoseconds**.
- **RAM Latency:** Even high-end **DDR5-8000** memory has a latency (wait time) of roughly **10 to 15 nanoseconds**.

The Result: For every single request the CPU makes to the RAM, it might have to wait for hundreds of its own clock cycles. This is called a **Memory Stall**. The CPU is like a world-class chef who can chop vegetables in seconds but has to wait minutes for the delivery truck to bring the ingredients.

3. Key Differences at a Glance

Feature	**Processor (CPU)**	**Memory (RAM)**
Primary Unit	Gigahertz (GHz)	MT/s or MHz
Core Function	Executing instructions (Logic)	Temporary data storage (Short-term memory)

Feature	Processor (CPU)	Memory (RAM)
Technology	SRAM (Static RAM - for Caches)	DRAM (Dynamic RAM - requires refresh)
Typical Speed	3.5 – 6.0 GHz	4800 – 8400+ MT/s
Wait Time	Near-instant (<1 ns for internal cache)	10 – 100 ns (Accessing main RAM)

4. How Modern Systems Bridge the Gap

Since RAM cannot keep up with the CPU, architects use several "tricks" to keep the processor busy:

- **CPU Caches (L1, L2, L3):** These are tiny, ultra-fast memory pools built *inside* the processor. They use **SRAM**, which is much faster than the **DRAM** used in your RAM sticks.
- **HBM (High Bandwidth Memory):** In 2026, high-end AI chips (like NVIDIA's B300 or R100) stack memory directly on top of the processor to reduce the physical distance data has to travel.
- **Speculative Execution:** The CPU "guesses" what data it will need next and asks the RAM for it *before* it actually needs it, helping to hide the latency.

High Speed memories:

High-speed memory in computer architecture, **primarily CPU registers and cache memory (L1, L2, L3), acts as a high-speed buffer between the CPU and main memory (RAM)**. It stores frequently accessed data and instructions to bridge the speed gap between the fast processor and slower main memory, significantly reducing access time and improving system performance.

Key Types of High-Speed Memory

- **Registers:** Located directly inside the CPU, they are the fastest possible storage in a computer, holding the data the processor is currently working on.
- **Cache Memory (SRAM):** Extremely fast, small-capacity memory located close to or on the CPU chip. It holds data and instructions that are frequently accessed.
    - **L1 Cache:** Fastest and smallest, built directly into the processor.
    - **L2 Cache:** Larger than L1, slightly slower.
    - **L3 Cache:** Largest and slowest of the cache levels, often shared between cores.
- **Static RAM (SRAM):** A type of volatile memory that is used for cache because it is much faster than Dynamic RAM (DRAM), which is typically used for main memory

- **Key Characteristics and Functions**

- **Locality of Reference:** Cache works based on the principle that programs tend to reuse data and instructions they have used recently.
- **Speed Optimization:** The cache stores segments of programs currently being executed.
- **Cost and Density:** High-speed memory is expensive and has lower storage density compared to DRAM or hard drives.
- **Cache Hit/Miss:** A "hit" occurs when the CPU finds data in the cache, allowing for instant access. A "miss" requires fetching data from the slower main memory.
- **Organization:** It acts as a buffer, ensuring the CPU does not have to wait for slower main memory to provide data.

Comparison Summary (2026 Standards)

Memory Type	Location	Technology	Speed/Latency	Typical Size
Registers	Inside CPU Core	Flip-Flops	< 0.1 ns	Bytes
L1/L2 Cache	Inside CPU Chip	SRAM	0.5 - 2.0 ns	KB to MB
HBM4	On-Package (Stacked)	SDRAM + TSV	5 - 10 ns (Ultra High Bandwidth)	24GB - 64GB
GDDR7	GPU Board	SGRAM	High Bandwidth / Med Latency	8GB - 24GB
Main RAM	Motherboard Slots	DDR5/DDR6	15 - 80 ns	16GB - 128GB

Computer Memory

- Memory is the electronic storage space where a computer keeps the instructions and data it needs to access quickly. It's the place where information is stored for immediate use. Memory is an important component of a computer, as without it, the system wouldn't operate correctly. The computer's operating system (OS), hardware, and software all rely on memory to function properly.

  Computer memory functions similarly to the human brain, storing data, information, and instructions. It acts as a storage unit or device where data to be processed and the instructions necessary for processing are kept. Both input and output data can be stored in memory.

How Computer Memory Communicates With the CPU?

Computer memory communicates with the CPU through a structured system of electronic pathways and controllers, enabling the CPU to fetch and store data rapidly and efficiently. Here's a detailed breakdown:

- **System Bus Structure:**
  The main channel for communication between the CPU and memory is the *system bus*, which is a collection of three types of buses:
  - **Data Bus:** Transfers the actual data between CPU and memory.
  - **Address Bus:** Carries the memory address that specifies where data should be read from or written to.
  - **Control Bus:** Sends signals that coordinate and control the activity, such as indicating read or write operations.
- **Memory Controller:**
  Communication is orchestrated by a *memory controller*, which manages the flow of data and ensures that signals between the CPU and memory are synchronized. In older systems, this controller was located on the motherboard; in modern computers, it's typically integrated into the CPU for greater speed and efficiency

Communication Process:

1. When the CPU needs to access data or instructions in memory, it places the address of the required memory location on the address bus.
2. The CPU sends a control signal (read or write command) on the control bus.
3. If reading, the memory controller retrieves the data from the specified address and sends it back to the CPU via the data bus. If writing, the CPU sends data over the data bus to be stored at the designated memory location.

4. This process is repeated billions of times per second during computing operations, forming the backbone of the *fetch-decode-execute* cycle used to run programs

Types of Computer Memory

In general, computer memory is divided into three types:

- Primary memory
- Secondary memory
- Cache memory

Now we discuss each type of memory one by one in detail:

1. Primary Memory

It is also known as the main memory of the computer system. It is used to store data and programs, or instructions during computer operations. It uses semiconductor technology and hence is commonly called semiconductor memory. Primary memory is of two types:

RAM (Random Access Memory):

It is a volatile memory. Volatile memory stores information based on the power supply. If the power supply fails/ interrupted/stopped, all the data and information on this memory will be lost. RAM is used for booting up or starting the computer. It temporarily stores programs/data which has to be executed by the processor. RAM is of two types:

- **S RAM (Static RAM):**S RAM uses transistors and the circuits of this memory are capable of retaining their state as long as the power is applied. This memory consists of the number of flip flops with each flip flop storing 1 bit. It has less access time and hence, it is faster.
- **D RAM (Dynamic RAM):**D RAM uses capacitors and transistors and stores the data as a charge on the capacitors. They contain thousands of memory cells. It needs refreshing of charge on capacitor after a few milliseconds. This memory is slower than S RAM.

ROM (Read Only Memory):

It is a non-volatile memory. Non-volatile memory stores information even when there is a power supply failed/ interrupted/stopped. ROM is used to store information that is used to operate the system. As its name refers to read-only memory, we can only read the programs and data that are stored on it. It contains some electronic fuses that can be programmed for a piece of specific information. The information is stored in the ROM in binary format. It is also known as permanent memory. ROM is of four types:

- **MROM(Masked ROM):** Hard-wired devices with a pre-programmed collection of data or instructions were the first ROMs. Masked ROMs are a type of low-cost ROM that works in this way.
- **PROM (Programmable Read Only Memory):** This read-only memory is modifiable once by the user. The user purchases a blank PROM and uses a PROM program to put the required contents into the PROM. Its content can't be erased once written.
- **EPROM (Erasable Programmable Read Only Memory):**EPROM is an extension to PROM where you can erase the content of ROM by exposing it to Ultraviolet rays for nearly 40 minutes.
- **EEPROM (Electrically Erasable Programmable Read Only Memory):** Here the written contents can be erased electrically. You can delete and reprogram EEPROM up to 10,000 times. Erasing and programming take very little time, i.e., nearly 4 -10 ms(milliseconds). Any area in an EEPROM can be wiped and programmed selectively.

2. Secondary Memory

It is also known as auxiliary memory and backup memory. It is a non-volatile memory and used to store a large amount of data or information. The data or information stored in secondary memory is permanent, and it is slower than primary memory. A CPU cannot access secondary memory directly. The data/information from the auxiliary memory is first transferred to the main memory, and then the CPU can access it.

Characteristics of Secondary Memory

- It is a slow memory but reusable.
- It is a reliable and non-volatile memory.
- It is cheaper than primary memory.
- The storage capacity of secondary memory is large.
- A computer system can run without secondary memory.
- In secondary memory, data is stored permanently even when the power is off.

Types of Secondary Memory

1. Magnetic Tapes**:** Magnetic tape is a long, narrow strip of plastic film with a thin, magnetic coating on it that is used for magnetic recording. Bits are recorded on tape as magnetic patches called RECORDS that run along many tracks. Typically, 7 or 9 bits are recorded concurrently. Each track has one read/write head, which allows data to be recorded and read as a sequence of characters. It can be stopped, started moving forward or backwards or rewound.

2. Magnetic Disks: A magnetic disk is a circular metal or a plastic plate and these plates are coated with magnetic material. The disc is used on both sides.

Bits are stored in magnetized surfaces in locations called tracks that run in concentric rings. Sectors are typically used to break tracks into pieces.

Hard discs are discs that are permanently attached and cannot be removed by a single user.

3. Optical Disks: It's a laser-based storage medium that can be written to and read. It is reasonably priced and has a long lifespan. The optical disc can be taken out of the computer by occasional users.

Types of Optical Disks

CD - ROM

- It's called a compact disk. Only read from memory.
- Information is written to the disc by using a controlled laser beam to burn pits on the disc surface.
- It has a highly reflecting surface, which is usually aluminium.
- The diameter of the disc is 5.25 inches.
- 16000 tracks per inch is the track density.
- The capacity of a CD-ROM is 600 MB, with each sector storing 2048 bytes of data.
- The data transfer rate is about 4800KB/sec. & the new access time is around 80 milliseconds.

WORM-(WRITE ONCE READ MANY)

- A user can only write data once.
- The information is written on the disc using a laser beam.
- It is possible to read the written data as many times as desired.
- They keep lasting records of information but access time is high.
- It is possible to rewrite updated or new data to another part of the disc.
- Data that has already been written cannot be changed.
- Usual size - 5.25 inch or 3.5 inch diameter.
- The usual capacity of a 5.25-inch disk is 650 MB,5.2GB etc.

DVDs

The term "DVD" stands for "Digital Versatile/Video Disc," and there are two sorts of DVDs:

- DVDR (writable)
- DVDRW (Re-Writable)
- **DVD-ROMS (Digital Versatile Discs)**: These are read-only memory (ROM) discs that can be used in a variety of ways. When compared to CD-ROMs, they can store a lot more data. It has a thick polycarbonate plastic layer that

serves as a foundation for the other layers. It's an optical memory that can read and write data.

- **DVD-R**: DVD-R is a writable optical disc that can be used just once. It's a DVD that can be recorded. It's a lot like WORM. DVD-ROMs have capacities ranging from 4.7 to 17 GB. The capacity of 3.5 inch disk is 1.3 GB.

3. Cache Memory

Cache Memory is a type of high-speed semiconductor memory that can help the CPU run faster. Between the CPU and the main memory, it serves as a buffer. It is used to store the data and programs that the CPU uses the most frequently.

Advantages of Cache Memory

- It is faster than the main memory.
- When compared to the main memory, it takes less time to access it.
- It keeps the programs that can be run in a short amount of time.
- It stores data for temporary use.

Disadvantages of Cache Memory

- Because of the semiconductors used, it is very expensive.
- The size of the cache (amount of data it can store) is usually small.

Computer memory is important for storing and processing data in a computer. Data is stored in the form of bits and bytes. As the amount of data grows, it's important to understand how larger units like kilobytes, megabytes, and gigabytes are used to store and manage bigger files. Understanding file sizes makes it easier to organize and manage data.

Ancient Manuscript Storage(Nalanda,Takshashila Libraries):

The ancient universities of **Nalanda** and **Takshashila** were not just centers of learning; they were the world's first "Data Centers." The way they managed millions of manuscripts (data units) mirrors the **Hierarchical Memory** and **Indexing** structures used in modern computer architecture.

1. Hierarchical Storage Structure (Memory Hierarchy)

Modern memory is organized by speed and capacity (Registers $\rightarrow$ Cache $\rightarrow$ RAM $\rightarrow$ Disk). The libraries at Nalanda (specifically the three massive buildings: *Ratnasagara*, *Ratnodadhi*, and *Ratnaranjaka*) followed a similar physical hierarchy.

A. The "Registers" (Active Use)

- **Ancient:** Manuscripts currently being studied by a scholar or monk were kept in small, accessible niches near their living quarters.
- **Computing:** These are like **Registers** or **L1 Cache**—extremely small capacity but immediately available for the CPU (the scholar).

B. The "Main Memory" (The Library Floor)

- **Ancient:** The *Dharmaganja* (the library complex) housed the primary collection. Manuscripts were categorized by subject (Logic, Medicine, Astronomy) on specific floors.
- **Computing:** This mirrors **RAM**. It is the primary workspace where data is organized and retrieved for active "processing" (reading/copying).

C. The "Secondary Storage" (Archives/Backups)

- **Ancient:** Rare or fragile original texts were kept in the deepest, most protected parts of the library, often in underground or specialized stone chambers to regulate temperature.
- **Computing:** This is the **Hard Drive** or **Cloud Archive**. It is meant for long-term "Persistence" rather than high-speed access.

2. Indexing and Metadata (The Grantha-Suchi)

With over 9 million manuscripts (at Nalanda's peak), retrieving a specific text without an **Index** would be impossible.

A. Tags and Labels (Metadata)

- **Ancient:** Every manuscript was wrapped in cloth (often red for insect protection) and had a **Tala-patra tag** hanging from the cord. This tag contained the *Title*, *Author*, and *Subject*.
- **Computing:** This is exactly like a **File Header** or **Metadata** in a file system. It allows the system to identify the content without "opening" (reading) the entire file.

B. Classification Systems (Indexing)

- **Ancient:** They used a "Suchi" (List) categorized by *Vidya* (Branch of Knowledge). A scholar looking for "Ayurveda" would go to a specific wing, then a specific shelf.
- **Computing:** This mirrors **B-Trees** or **Hash Tables** used in databases. By categorizing data into "buckets," the search time is reduced from O(n) to O(\log n).

3. Data Integrity and Redundancy

Ancient libraries treated knowledge like "Mission-Critical Data."

- **The Backup System (Lipi-daan):** Since palm leaves decay (Data Rot), Nalanda maintained a scriptorium where monks continuously copied old texts onto new ones. This is **Data Mirroring** or **RAID 1** logic—ensuring that if one "disk" (manuscript) fails, the data exists on another.
- **The "Firewall" (The Gatekeepers):** To enter Nalanda, a student had to pass a verbal exam by the *Dwarapandita* (Gate Scholar).
    - **Computing:** This is the **Authentication and Authorization** layer. Only "verified users" could access the "Data Server" (the library).

5. Architectural Similarity Table

Feature	Ancient Library (Nalanda/Takshashila)	Modern Computer Architecture
Storage Unit	Manuscript (Palm Leaf/Birch Bark)	Data Block / File

Categorization	Vidya / Shastra (Subject-based)	Directory / Folder Structure
Search Method	Suchi (Catalogue)	Indexing / Hashing
Memory Level	Study Niche → Shelf → Archive	Cache → RAM → Storage
Persistence	Lipi-daan (Re-copying)	Data Backup / Redundancy
Access Control	Dwarapandita (Gate Scholar)	Firewall / Login Authentication

Module V

Processor and Control Unit:

Difference Between Hardwired and Micro programmed Control Unit

- An integral part of a computer's **Central Processing Unit (CPU)** that controls processor function is called a **control unit (CU)**. To regulate how instructions are carried out by the CPU, the Control Unit produces control signals. Hardwired control units and microprogrammed control units are the two primary categories of control units. Although they are both employed to create control signals, their functionality, design, and methods of implementation are different. This article examines the distinctions between microprogrammed and hardwired control units and explains their benefits, drawbacks, and uses.

What is a Hardwired Control Unit?

Hardwired control units are a particular kind of control unit where the hardware implementation is done using fixed logic circuits like flip-flops, gates, decoders, and other combinational circuits. Based on the logic created with these components, it produces control signals. Although this kind of control unit is renowned for its quickness, bigger systems with several control signals may find it to be very complex.

Advantages of Hardwired Control Unit

- **High Speed:** Without the need to read instructions, the Hardwired Control Unit creates control signals directly via fixed logic circuits, making it quicker.
- **Requires Less Chip Area:** It takes up less space on the chip since it requires fewer components.
- **Appropriate for Basic Instructions:** RISC (Reduced Instruction Set Computing) microprocessors and other systems with a limited number of basic instructions are good candidates for this technology.

Disadvantages of Hardwired Control Unit

- **Lack of Flexibility:** It cannot be made to adapt to new guidelines or modifications to system requirements. Wiring adjustments are necessary when making modifications to the design, and a complete circuit redesign may be necessary.

- **Complexity:** Handling complicated instruction sets gets more difficult as the control unit's size increases due to its very complex architecture.
- **Increased Error Occurrence:** Because wiring and circuit design are complicated, there is a greater chance of mistakes.

What is a Microprogrammed Control Unit?

One kind of control unit that generates control signals via programming is called a microprogrammed control unit. It uses a series of micro-operations performed by running a program made up of micro-instructions rather than fixed hardware circuits. Each of these micro-instructions, which are kept in a control memory, is in charge of producing a particular set of control signals.

Advantages of Microprogrammed Control Unit

- Control logic design and implementation are made simpler by microprogrammed control units. Designers may utilize microinstructions to direct the execution of an operation without of creating intricate combinational circuits for every action. This lowers the control unit's complexity, which facilitates design, modification, and debugging.
- Instead of changing hardware, changes to the instruction set or architecture may be made by modifying the microprogram, as the control signals are produced by microinstructions that are stored in memory. Without requiring significant modifications to the hardware, this flexibility makes it simple to adjust to new features, issue patches, or optimizations.
- A single hardware platform may support many instruction sets or several iterations of the same instruction set thanks to the usage of microprogramming. To provide compatibility across different software ecosystems or processor generations, for instance, distinct microprograms may be placed into the control store to handle different instruction sets.

Disadvantages of Microprogrammed Control Unit

- **Slower pace:** Because a microprogrammed control unit must understand and carry out micro-instructions, its pace is slower than that of a hardwired control unit.
- **Greater Chip Area Needed:** Because control memory is used to store microprograms, a larger chip area is needed.
- **Increased Cost for Large Control Memory:** If a large control memory is required to hold a large number of microprograms, the cost may go up.

Difference Between Hardwired and Microprogrammed Control Units

ATTRIBUTES	HARDWIRED CONTROL UNIT	MICROPROGRAMMED CONTROL UNIT
Speed	Speed is fast	Speed is slow
Cost of Implementation	More costlier.	Cheaper.
Flexibility	Not flexible to accommodate new system specification or new instruction redesign is required.	More flexible to accommodate new system specification or new instruction sets.
Ability to Handle Complex Instructions	Difficult to handle complex instruction sets.	Easier to handle complex instruction sets.
Decoding	Complex decoding and sequencing logic.	Easier decoding and sequencing logic.
Applications	RISC Microprocessor	CISC Microprocessor
Instruction set of Size	Small	Large

ATTRIBUTES	HARDWIRED CONTROL UNIT	MICROPROGRAMMED CONTROL UNIT
Control Memory	Absent	Present
Chip Area Required	Less	More
Occurrence	Occurrence of error is more	Occurrence of error is less

Difference between RISC and CISC Processor

- The **microprocessor** is a processing unit on the single chip. It is the integrated circuit that performs the core functions of the computer CPU. It is the multipurpose programmable silicon chip constructed using a Metal Oxide Semiconductor (MOS) technology which is clock driven and register based. It accepts a binary data as a input and provides output after processing it as per a specification of instructions stored in a memory. These microprocessors are capable of the processing the 128 bits at the time at the speed of a one billion instructions per second.

Characteristics of a Microprocessor

1. **Instruction Set** : The set of complete instructions that the microprocessor executes is termed the instruction set.
2. **Word Length** : The number of bits processed in a single instruction is called word length or word size. The Greater the word size is the larger the processing power of the CPU.
3. **System Clock Speed** : A Clock speed determines how fast the single instruction can be executed in the processor. The microprocessor is controlled by the System Clock. A Clock speeds are generally measured in the millions of a cycles per second (MHz) and thousand million cycles per second GHz. A Clock speed is considered to be the very important aspect of predicting a performance of the processor.

What is Reduced Instruction Set Computer (RISC)

It stands for Reduced Instruction Set Computer. It is a type of microprocessor architecture that uses a small set of instructions of uniform length. These are simple instructions that are generally executed in one clock cycle. RISC chips are relatively simple to design and inexpensive. The setback of this design is that the computer has to repeatedly perform simple operations to execute a larger program having a large number of processing operations.
Examples: SPARC, POWER PC, etc.

Advantages of Reduced Instruction Set Computer (RISC)

- Faster execution speed RISC processors use the simpler instructions that can be executed more quickly leading to improved overall performance in the many tasks.
- Lower power consumption The simpler design of a RISC processors often results in the lower power usage making them the ideal for mobile devices and energy efficient computing.
- Easier to design and manufacture RISC processors have a simpler architecture which can make them the easier and potentially cheaper to design and produce.

Disadvantages of Reduced Instruction Set Computer (RISC)

- Larger code size RISC processors often require more lines of code to perform complex tasks, which can lead to larger program sizes and increased memory usage.
- More work for compilers The simpler instruction set means compilers for RISC processors need to do more work to translate high-level programming languages into machine code.
- Limited built-in functionality RISC processors have fewer complex instructions built into hardware, which can make certain specialized tasks less efficient without additional software support.

What is Complex Instruction Set Computer (CISC)?

It stands for Complex Instruction Set Computer. These processors offer the users, hundreds of instructions of variable sizes. CISC architecture includes a complete set of special-purpose circuits that carry out these instructions at a very high speed. These instructions interact with memory by using complex addressing modes. CISC processors reduce the program size and hence lesser number of memory cycles are required to execute the programs. This increases the overall speed of execution.
Examples: Intel architecture, AMD

Advantages of Complex Instruction Set Computer (CISC)

- Smaller code size CISC processors can perform complex operations with single instructions, often resulting in more compact code and reduced memory usage.
- Rich instruction set The diverse set of complex instructions can make programming easier and more intuitive for certain tasks, especially in assembly language.
- Backwards compatibility CISC architectures, like x86, often maintain compatibility with older software, making system upgrades easier for users and businesses.

Disadvantages of Complex Instruction Set Computer (CISC)

- Slower execution speed Complex instructions typically take longer to execute, potentially resulting in slower overall performance compared to RISC processors.
- Higher power consumption The more complex hardware required for the CISC processors often leads to a increased power usage making them less suitable for the mobile devices.

- More complex hardware design a CISC processors require more complex circuitry to handle their varied the instructions which can make them a more challenging and expensive to design and manufacture.

Difference between RISC and CISC processor

CISC	RISC
A large number of a instructions are present in the architecture.	Very few instructions are present. The number of instructions is generally less than 100.
Some instructions with long execution times. These include instructions that copy an entire block from one part of memory to another and others that copy multiple registers to and from memory.	No instruction with a long execution time due to a very simple instruction set. Some early RISC machines did not even have an integer multiply instruction, requiring compilers to implement multiplication as a sequence of additions.
Variable-length encodings of the instructions. **Example:** IA32 instruction size can range from 1 to 15 bytes.	Fixed-length encodings of the instructions are used. **Example:** In IA32, generally all instructions are encoded as 4 bytes.
Multiple formats are supported for specifying operands. A memory operand specifier can have many different combinations of displacement, base, and index register.	Simple addressing formats are supported. Only base and displacement addressing is allowed.
CISC supports array.	RISC does not support an array.
Arithmetic and logical operations can be applied to both memory and register	Arithmetic and logical operations only use register operands. Memory referencing is only

CISC	RISC
operands.	allowed by loading and storing instructions, i.e. reading from memory into a register and writing from a register to memory respectively.
Implementation programs are hidden from machine-level programs. The ISA provides a clean abstraction between programs and how they get executed.	Implementation programs exposed to machine-level programs. Few RISC machines do not allow specific instruction sequences.
Condition codes are used.	No condition codes are used.
The stack is being used for procedure arguments and returns addresses.	Registers are being used for procedure arguments and return addresses. Memory references can be avoided by some procedures.
Successful pipeline with one instruction per cycle	Unsuccessful pipeline
Heavy use of RAM	More efficient use of RAM

The Reduced Instruction Set Computing and a Complex Instruction Set Computing are the method of a processor design. The RISC processors use the fewer and simpler instructions that execute quickly while the CISC processors have the complex instructions that can perform the multiple operations. The RISC focuses on a efficiency and speed using a simpler hardware and relying on the software. CISC aims for a versatility with more built in functionality in the hardware. RISC is a generally faster and more energy efficient while a CISC can be more compact in a terms of code size.

Basics of Pipelining

- Pipelining is a mechanism used to improve system performance in which tasks are executed in an overlapping manner. In this technique, the problem is divided into subproblems & assigned to the pipes, then the pipes operate under the same clock.
  - Pipelining is the process of arranging hardware elements of a CPU such that its overall performance is increased.
  - Simultaneous execution of more than one instruction takes place in a pipelined processor.
  - In pipelining, multiple instructions are overlapped in execution.
  - Accepting new input before the previously accepted input appears as an output at the other end.

Design of Pipeline

The diagram shows the concept of pipelining with stages:

- Pipelining has two ends: the input end & output end. Between the input & output ends, Multiple pipes are interconnected to satisfy the functionality.
- These Pipes are called stage or segments. Between the stages buffer are used to store to intermediate result.
- This Buffer is also called as pipeline Register/Interface.

Pipeline Stages
A typical instruction pipeline is comprised of several discrete stages. Each stage completes a part of the instruction cycle, such as:

- **Instruction Fetch (IF)**: Retrieves the next instruction from memory.
- **Instruction Decode (ID)**: Decodes the instruction to determine required operations.
- **Execute (EX):** Performs arithmetic or logical operations.
- **Memory Access (MEM):** Reads or writes data from/to memory.
- **Writeback (WB):** Writes the result back to the register file.

The output from one stage becomes the input for the next, passed via pipeline registers. All stages progress in parallel, synchronised by a common clock.

Benefits of Pipelining

- **Increased Throughput:** Multiple instructions are executed simultaneously, improving CPU performance.

- **Efficient Resource Usage:** Stages keep hardware busy nearly all the time, akin to an assembly line.
- **Higher Clock Rates:** Pipelined CPUs can run at higher frequencies since stages work on simpler operations within each clock cycle.

Challenges

Pipelining can encounter several issues called hazards, which disrupt smooth execution:

- **Data Hazards:** When instructions depend on results of previous instructions still in the pipeline.
- **Control Hazards**: Caused by branch instructions altering the instruction flow.
- **Structural Hazards:** Occur when hardware resources are insufficient for concurrent stages.

Introduction to Pipelined data Path and Control:

A pipelined datapath and control enhances CPU performance by overlapping instruction execution, effectively creating an assembly line where multiple instructions are processed simultaneously in stages (IF, ID, EX, MEM, WB). It increases throughput using pipeline registers to separate stages, requiring careful control signal management and hazard handling to manage data dependencies and branching

Key Concepts of Pipelined Datapath & Control

- Five-Stage Pipeline Stages:
    1. Instruction Fetch (IF): Retrieves the next instruction from memory.
    2. Instruction Decode (ID): Decodes the instruction and reads registers.
    3. Execute (EX): Performs arithmetic or logical operations.
    4. Memory Access (MEM): Reads or writes to data memory.
    5. Writeback (WB): Writes results back to the register file.
- Pipeline Registers: Inserted between stages (IF/ID, ID/EX, EX/MEM, MEM/WB) to pass data and control signals to the next stage, synchronizing operations.
- Performance Impact: While a single instruction might take the same time to complete, the *throughput* (instructions completed per second) increases significantly.
- Pipelined Control: Control signals are generated during the Instruction Decode stage and passed down the pipeline along with the instruction to be used in the appropriate stage. Pipeline HazardsHazards prevent the next instruction from executing in the next cycle, causing stalls:

- Structural Hazards: Hardware conflicts where two instructions need the same resource (e.g., using one memory for both instructions and data).
- Data Hazards: Occur when an instruction depends on the result of a previous instruction still in the pipeline.
- Control Hazards: Result from branches and jumps, where the next instruction cannot be determined immediately.

  Mitigation Techniques

- Forwarding/Bypassing: Passing calculated data directly to a dependent instruction before it is written back to the register file.
- Stalling (Bubbles): Delaying instructions to resolve hazards.
- Branch Prediction: Guessing the branch outcome to reduce control hazards.

Handling Data Hazard and Control Hazard

Data hazards (RAW, WAR, WAW) are handled via forwarding (bypassing) to feed data directly to dependent instructions, or by stalling (inserting bubbles/no-ops) when data isn't ready. Control hazards (branch/jump) are managed by branch prediction, delayed branching (compiler reordering), or flushing the pipeline if a branch is mispredicted

Handling Data Hazards

Data hazards arise when instructions dependent on previous results are executed simultaneously in a pipeline.

- Data Forwarding (Bypassing): Hardware directly forwards the result from one functional unit (like the ALU) to another's input (like the ALU's next input) instead of waiting for the data to be written to a register file.
- Stalling (Pipeline Interlock): If data cannot be forwarded in time (e.g., in a load-use hazard), the pipeline inserts "bubbles" or "nop" (no-operation) instructions to freeze the dependent instruction until the data is ready.
- Code Reordering (Compiler Scheduling): The compiler rearranges instructions to increase the distance between dependent instructions, reducing the need for stalls.

Handling Control Hazards

Control hazards occur because the pipeline fetches the next instruction before knowing the outcome of a branch or jump.

- Branch Prediction: The processor guesses the direction of a branch to continue fetching instructions, reducing stall time. If the guess is correct, there is no performance penalty.
- Delayed Branching: The processor always executes one or more instructions following a branch, allowing useful work to be done, which reduces the penalty.
- Flushing the Pipeline (Branch Penalty): If a branch is mispredicted, the instructions already fetched into the pipeline must be discarded (flushed), creating a penalty

Summary of Hazard Mitigation

Hazard Type	Main Causes	Typical Solutions
Data (RAW)	Instruction depends on previous result	Forwarding, Stalling (Bubbles)
Data (WAR/WAW)	Out-of-order execution	Pipeline scheduling, Register renaming
Control	Conditional branches, Jumps	Branch Prediction, Delayed Branching

These techniques ensure that the pipeline runs as efficiently as possible without violating the logical flow of the program.

What is Stack Organization?

Stack organization is a fundamental concept in computer architecture and programming that involves the management of memory using a Last-In-First-Out (LIFO) data structure. A stack is a specialized form of data storage that operates on the principle of pushing and popping elements. It is widely used in various computing contexts, including function calls, memory allocation, and expression evaluation.

Stack Operations:

The primary operations associated with a stack are push and pop. When an element is pushed onto the stack, it is added to the top of the stack, becoming the most recent element. When an element is popped from the stack, the most recent element is removed, and the stack's top pointer is adjusted accordingly.

The push operation involves the following steps:

- Check if the stack is full.
- If not full, increment the top pointer and place the new element at the updated top position.
- Store the element's value and update the top pointer.

The pop operation involves the following steps:

- Check if the stack is empty.
- If not empty, retrieve the element at the top position.
- Decrement the top pointer.
- Return the retrieved element.
- In addition to push and pop, other stack-related operations include peek (to view the top element without removing it) and isEmpty (to check if the stack is empty).

Stack Implementation:

Implementing a stack can be achieved using various programming constructs and data structures. Two common methods are array-based implementation and linked list-based implementation.

1. Array-Based Implementation:

In this approach, an array of fixed size is used to store stack elements. The top pointer indicates the index of the last inserted element.

Advantages:

- Simple implementation.
- Efficient memory usage when the maximum size is known.

Disadvantages:

- Fixed-size, which can lead to overflow or underflow.
- Inefficient resizing if the array becomes full.

2. Linked List-Based Implementation:

In this approach, a linked list is used to represent the stack. Each node in the linked list holds an element and a reference to the next node.

Advantages:

- Dynamic size: memory usage grows as needed.
- No fixed capacity limitations.

Disadvantages:

- Slightly more complex implementation compared to array-based.
- Slightly more memory overhead due to node references.

Register Stack

A register stack, also known as a register file stack, is a specialized type of hardware structure within a computer's central processing unit (CPU) that facilitates efficient register management and context switching. Registers are small, fast memory locations that hold data that the CPU uses for immediate calculations and operations.

Key Features and Functions:

- Context Switching:
  Register stacks allow for efficient context switching between different tasks or processes. When the CPU switches from one task to another, it can simply swap the entire register stack context, reducing the overhead associated with saving and restoring individual register values.

- Function Calls:
  During function calls, a register stack can be utilized to save the caller's context, including return addresses and function arguments. This enables the CPU to seamlessly switch between different functions without losing track of execution.
- Performance Enhancement:
  Register stacks can improve performance by reducing memory access latency. Registers are much faster to access compared to main memory. By utilizing multiple registers in a stack, the CPU can keep frequently accessed data closer, minimizing the need to fetch data from slower memory locations.
- Parallel Execution:
  Modern CPUs often have multiple execution units that can perform different tasks simultaneously. Register stacks can help manage the data flow between these units efficiently, allowing for better utilization of the CPU's parallel processing capabilities.

Memory Stack

A memory stack, also known simply as a stack, is a crucial data structure used in computer programming and computer architecture for managing memory in a Last-In-First-Out (LIFO) manner. It operates as a dynamic storage area where data is organized in a way that the most recently added item is the first to be removed. The stack's primary purpose is to keep track of program execution flow, local variables, and function calls.

Key Characteristics and Functions:

- LIFO Structure:
  Data added and removed in last-in, first-out order, similar to a stack of plates.
- Function Calls:
  Stacks manage function calls by storing return addresses and local variables, enabling program flow after function completion.
- Local Variables:
  Stack holds function local variables, managing distinct data for various function instances.
- Expression Evaluation:
  Stacks assist in evaluating mathematical expressions with parentheses and operators, preserving operation order for precise calculation.

Implementation and Usage:

Memory stacks are commonly implemented using arrays or linked lists. In array-based implementation, a fixed-size array is used to hold the stack's elements. In linked list-based implementation, nodes with data and pointers to the next node create the stack's structure.

Instruction Formats

- Instruction format defines how instructions are represented in a computer's memory. There are different types of instruction formats, including zero, one, two, and three-address instructions.

  Defines how the CPU decodes and executes instructions.

  - **Opcode:** This field specifies the operation to be performed by the CPU, such as addition, subtraction, or data transfer.
  - **Operands:** These fields contain the data or references (addresses) to data on which the operation acts.
  - **Addressing Mode:** This specifies how to interpret or locate the operand, such as direct, indirect, or immediate addressing.

Types of Instruction Formats
Instruction formats are classified into zero, one, two, and three-address types, depending on how many address fields they have. Each type works differently and is used in various ways in computer architecture.

NOTE: We will use the X = (A+B)*(C+D) expression to showcase the procedure.

1) **Zero Address Instructions**

These instructions do not specify any operands or addresses. Instead, they operate on data stored in registers or memory locations implicitly defined by the instruction. For example, a zero-address instruction might simply add the contents of two registers together without specifying the register names.

A stack-based computer does not use the address field in the instruction. To evaluate an expression, it is first converted to reverse Polish Notation i.e. Postfix Notation.

Expression: X = (A+B)*(C+D)
Postfixed : X = AB+CD+*
TOP means top of stack
M[X] is any memory location

Instruction	Stack (TOP Value After Execution)
PUSH A	TOP = A
PUSH B	TOP = B
ADD	TOP = A + B
PUSH C	TOP = C
PUSH D	TOP = D
ADD	TOP = C + D
MUL	TOP = (C + D) * (A + B)
POP X	M[X] = TOP

2) **One Address Instructions**

These instructions specify one operand or address, which typically refers to a memory location or register. The instruction operates on the contents of that operand, and the result may be stored in the same or a different location. For example, a one-address instruction might load the contents of a memory location into a register.

This uses an implied ACCUMULATOR register for data manipulation. One operand is in the accumulator and the other is in the register or memory location. Implied means that the CPU already knows that one operand is in the accumulator so there is no need to specify it.

Expression: X = (A+B)*(C+D)
AC is accumulator
M[] is any memory location
M[T] is temporary location

Instruction	Stack / Register (AC / M[])
AC = A	AC = A
AC = AC + B	AC = A + B
M[T] = AC	M[T] = A + B
AC = C	AC = C
AC = AC + D	AC = C + D
M[] = AC	M[] = C + D
AC = AC * M[T]	AC = (A + B) * (C + D)
M[X] = AC	M[X] = (A + B) * (C + D)

3)Two Address Instructions

These instructions specify two operands or addresses, which may be memory locations or registers. The instruction operates on the contents of both operands, and the result may be stored in the same or a different location. For example, a two-address instruction might add the contents of two registers together and store the result in one of the registers.

This is common in commercial computers. Here two addresses can be specified in the instruction. Unlike earlier in one address instruction, the result was stored in the

accumulator, here the result can be stored at different locations rather than just accumulators, but require more number of bit to represent the address.

Here destination address can also contain an operand.

Expression: X = (A+B)*(C+D)
R1, R2 are registers
M[] is any memory location

Instruction	Registers / Memory (R1, R2, M[])
R1 = A	R1 = A
R1 = R1 + B	R1 = A + B
R2 = C	R2 = C
R2 = R2 + D	R2 = C + D
R1 = R1 * R2	R1 = (A + B) * (C + D)
M[X] = R1	M[X] = (A + B) * (C + D)

3) **Three Address Instructions**

These instructions specify three operands or addresses, which may be memory locations or registers. The instruction operates on the contents of all three operands, and the result may be stored in the same or a different location. For example, a three-address instruction might multiply the contents of two registers together and add the contents of a third register, storing the result in a fourth register.

This has three address fields to specify a register or a memory location. Programs created are much short in size but number of bits per instruction increases. These instructions make the creation of the program much easier but it does not mean that program will run much faster because now instructions only contain more information but each micro-operation (changing the content of the register, loading address in the address bus etc.) will be performed in one cycle only.

Expression: X = (A+B)*(C+D)
R1, R2 are registers
M[] is any memory location

Instruction	Description
R1 = A	Load value A into R1
R1 = R1 + B	Add B to R1
M[T1] = R1	Store R1 into memory at M[T1]
R2 = C	Load value C into R2
R2 = R2 + D	Add D to R2
M[T2] = R2	Store R2 into memory at M[T2]
R1 = M[T1]	Load M[T1] into R1
R1 = R1 * M[T2]	Multiply R1 by M[T2] and store in R1
M[X] = R1	Store R1 into memory at M[X]

What is data transfer and manipulation in computer system architecture.

Data transfer involves the movement of data between different components of a computer system, such as from the processor to memory, between registers, or across network interfaces. This movement is essential for fetching instructions, storing results, and communicating between various parts of a system.

Data manipulation, on the other hand, refers to the processes that transform and operate on data to produce desired outcomes. This includes arithmetic operations, logical operations, bit-level manipulations, and data shifting. Each of these operations plays a vital role in how computers process information and execute tasks.

Different types of manipulation in computer
Data manipulation instructions can be categorized into three parts:

1) Arithmetic instruction
2) Logical and bit manipulation instructions
3) Shift instructions

Arithmetic Instruction
Arithmetic instructions include increment, decrement, add, subtract, multiply, divide, add with Carry, subtract with Borrow, negate that is (2's) two's complement. If there's a negative number, it is considered as negate (so two's complement).

Name	Mnemonic	Example	Explanation
Increment	INC	INC R1	Increases the value stored in register R1 by 1.
Decrement	DEC	DEC R2	Decreases the value stored in register R2 by 1.
Add	ADD	ADD R3, R4	Adds the value in register R4 to the value in register R3.
Subtract	SUB	SUB R5, R6	Subtracts the value in register R6 from the value in R5.
Multiply	MUL	MUL R7, R8	Multiplies the value in register R7 by the value in R8.

Divide	DIV	DIV R9, R10	Divides the value in register R9 by the value in R10.
Add with carry	ADDC	ADDC R11, R12	Adds the value in register R12 and the carry flag to R11.
Subtract with borrow	SUBB	SUBB R13, R14	Subtracts the value in R14 and the borrow flag from R13.

Generally, most computers carry instructions for all four of these operations. If computers have only addition(ADD) and possibly subtraction(SUB) instructions, the other two operations, i.e. multiplication(MUL) and division(DIV) must be generated using software subroutines. These four basic arithmetic operations are sufficient for solving scientific problems when expressed in numerical analysis methods.
The table given below shows the Arithmetic Instructions:

Logical and Bit Manipulation Instruction
We have another list of instructions that is logical and bit manipulation instructions starting with clear (that means clear the content of accumulator), complement the accumulator, AND, OR, Exclusive-OR, Clear carry, Set carry, Complement carry, Enable interrupts, Disable interrupts, all these are logical and bit manipulation instructions.
These logical instructions consider each operand bit individually and treat it as a Boolean variable. Basically, logical instructions help perform binary operations on strings of bits stored in registers.

- Clear instruction means making all the bits of a register '0'.
- AND instruction is sometimes referred to as bit clear instruction or mask.
- OR instruction is sometimes referred to as bit set instruction.
- Set instruction means making all the bits of a register '1'.
- XOR instruction is referred to as bit complement instruction.

Name	Mnemonic	Example	Explanation
Clear	CLR	CLR R1	Sets the value in register R1 to 0.
Complement	COM	COM R2	Inverts all the bits in the value stored in register R2.
AND	AND	AND R3, R4	Performs a bitwise AND between values in R3 and R4, stores the result in R3.

Name	Mnemonic	Example	Explanation
OR	OR	OR R5, R6	Performs a bitwise OR between values in R5 and R6, stores the result in R5.
Exclusive-OR	XOR	XOR R7, R8	Performs a bitwise XOR between values in R7 and R8, stores the result in R7.
Clear carry	CLRC	CLRC	Clears the carry flag (sets it to 0).
Set Carry	SETC	SETC	Sets the carry flag to 1.
Complement Carry	COMC	COMC	Inverts the carry flag (if it was 1, it becomes 0, and vice versa).
Enable Interrupt	EI	EI	Enables the interrupt system, allowing interrupts to occur.
Disable Interrupt	DI	DI	Disables the interrupt system, preventing interrupts from occurring.

Shift Instructions

Shift instructions allow the bits of a memory byte or register to be shifted one-bit place to the right or the Left.

There are basically two types of shift instructions — arithmetic and logical. Arithmetic shifts consider the contents of the memory byte or register to be a signed number. So, when the shift is made, the number is arithmetically divided by two (right shift) or multiplied by two (left shift). Logical shifts consider the contents of the register or memory byte to be just a bit pattern when the shift is made.

- OP is the opcode field
- RL (It tells whether to shift it right or left).
- REG (It determines which register is to be shifted).
- COUNT (It tells the number of places to be shifted).
- TYPE(It tells the type of shifting from the list given below).

In right-shift operations, zeros are shifted into high-order vacated positions. And in the case of the left-shift operation, shifts the zero into low-order vacated positions.

Name	Mnemonic
Logical Shift Right	SHR

Name	Mnemonic
Logical Shift Left	SHL
Arithmetic Shift Right	SHRA
Arithmetic Shift Left	SHLA
Rotate Right	ROR
Rotate Left	ROL
Rotate Right through carry	RORC
Rotate Left through carry	ROLC

Program Control Instructions

- Program Control Instructions are machine code instructions that manage the flow of execution in a microprocessor or microcontroller. They allow the processor to make decisions, repeat operations, or stop execution as needed. These instructions are commonly written in assembly language or generated from high-level languages during compilation.
  - Direct the processor to execute specific tasks and access different program segments.
  - Enable decision-making and looping within a program.
  - Control how and when instructions are executed.

Types of Program Control Instructions

Following are some control instructions with their examples:

1. Compare Instruction

Compare instruction is specifically provided, which is similar to a subtract instruction except the result is not stored anywhere, but flags are set according to the result.

Example:

CMP R1, R2 ;

2. Unconditional Branch Instruction

It causes an **u**nconditional change in the execution sequence, meaning the processor directly jumps to a new memory location and continues execution from there without checking any conditions.

Example:
JUMP L2
Mov R3, R1 goto L2

3. Conditional Branch Instruction

A conditional branch instruction is used to examine the values stored in the condition code register to determine whether the specific condition exists and to branch if it does.

Example:
Assembly Code : BE R1, R2, L1
Compiler allocates R1 for x and R2 for y
High Level Code: if (x==y) goto L1;

4. Subroutines

A subroutine is a program fragment that lives in user space, performs a well-defined task. It is invoked by another user program and returns control to the calling program when finished.

Example:
CALL and RET

5. Halting Instructions

Halting Instructions are special instructions used to either pause or stop the execution of a program without performing any data processing operations. They help in managing processor timing, synchronization, or bringing the system to a controlled stop.

- **NOP (No Operation):** Causes no change in the processor state other than advancing the program counter. It is often used to synchronize timing.
- **HALT:** Brings the processor to an orderly halt and keeps it in an idle state until restarted by an interrupt, trace, reset, or external action.

6. Interrupt Instructions

Interrupt is a mechanism by which an I/O or an instruction can suspend the normal execution of processor and get itself serviced.

- **RESET** - It reset the processor. This may include any or all setting registers to an initial value or setting program counter to standard starting location.
- **TRAP** - It is non-maskable edge and level triggered interrupt. TRAP has the highest priority and vectored interrupt.
- **INTR** - It is level triggered and maskable interrupt. It has the lowest priority. It can be disabled by resetting the processor.

What is General Register Organization?

A set of flip-flops forms a register. A register is a unique high-speed storage area in the CPU. They include combinational circuits that implement data processing. The information is always defined in a register before processing. The registers speed up the implementation of programs.

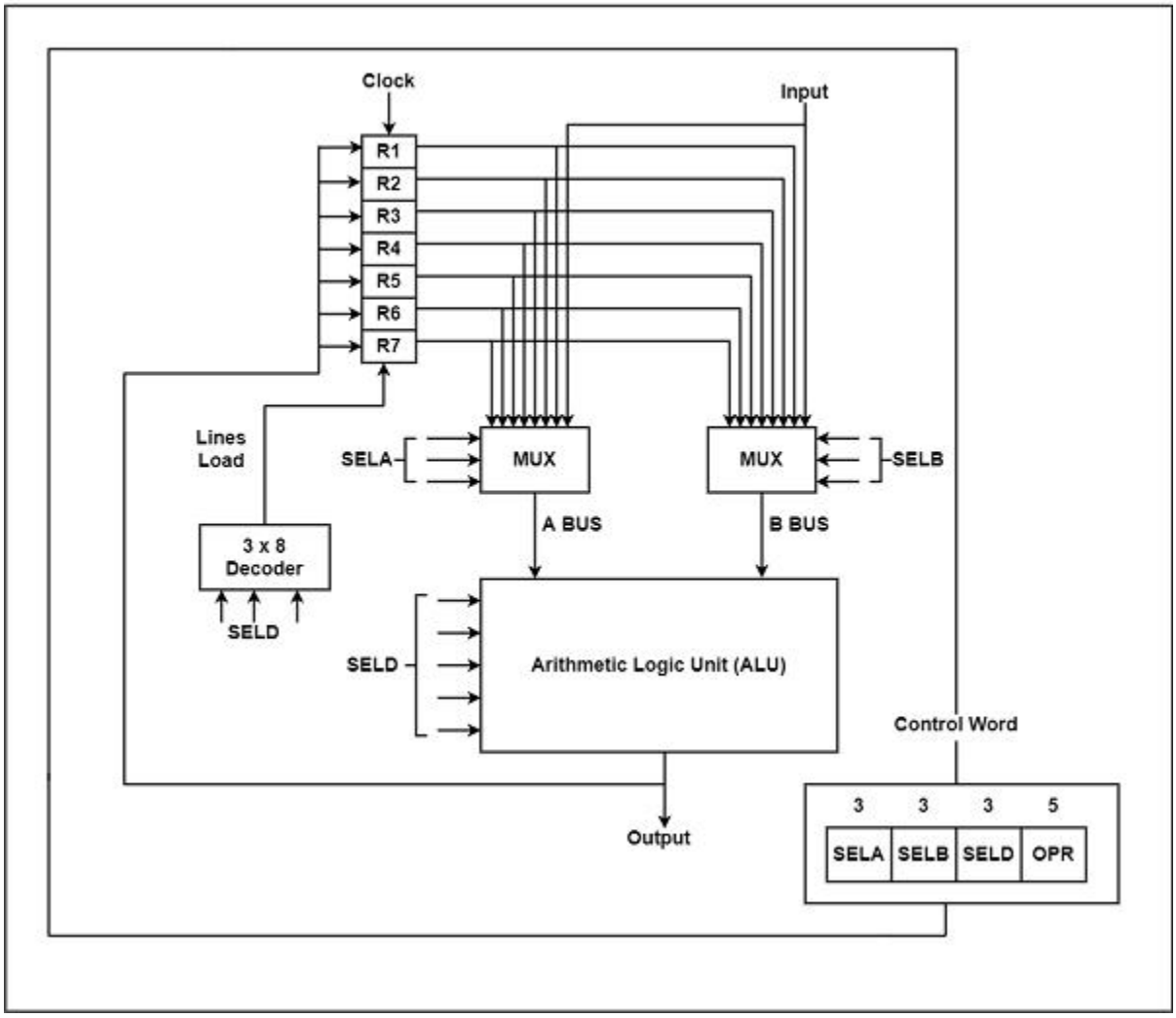
Registers implement two important functions in the CPU operation are as follows −

- It can support a temporary storage location for data. This supports the directly implementing programs to have fast access to the data if required.
- It can save the status of the CPU and data about the directly implementing program.

Example − Address of the next program instruction, signals get from the external devices and error messages, and including different data is saved in the registers.

If a CPU includes some registers, therefore a common bus can link these registers. A general organization of seven CPU registers is displayed in the figure.

General Organization of Registers



The CPU bus system is managed by the control unit. The control unit explicit the data flow through the ALU by choosing the function of the ALU and components of the system.

Consider R1 ← R2 + R3, the following are the functions implemented within the CPU −

MUX A Selector (SELA) − It can place R2 into bus A.

MUX B Selector (SELB) − It can place R3 into bus B.

ALU Operation Selector (OPR) − It can select the arithmetic addition (ADD).

Decoder Destination Selector (SELD) − It can transfers the result into R1.

The multiplexers of 3-state gates are performed with the buses. The state of 14 binary selection inputs determines the control word. The 14-bit control word defines a micro-operation.

The encoding of register selection fields is specified in the table.

Encoding of Register Selection Field

Binary Code	SELA	SELB	SELD
000	Input	Input	None
001	R1	R1	R1
010	R2	R2	R2
011	R3	R3	R3
100	R4	R4	R4
101	R5	R5	R5
110	R6	R6	R6
111	R7	R7	R7

There are several micro-operations are implemented by the ALU. Few of the operations implemented by the ALU are displayed in the table.

Encoding of ALU Operations

OPR Select	Operation	Symbol

OPR Select	Operation	Symbol
00000	Transfer A	TSFA
00001	Increment A	INCA
00010	Add A + B	ADD
00101	Subtract A - B	SUB
00110	Decrement A	DECA
01000	ADD A and B	AND
01010	OR A and B	OR
01100	XOR A and B	XOR
01110	Complement A	COMA
10000	Shift right A	SHRA
11000	Shift left A	SHLA

There are some ALU micro-operations are shown in the table.

ALU Micro-Operations

Micro-operation	SELA	SELB	SELD	OPR	Control Word

R1 ← R2 – R3	R2	R3	R1	SUB	010	011	001	00101
R4 ← R4 ∨ R5	R4	R5	R4	OR	100	101	100	01010
R6 ← R6 + R1	-	R6	R1	INCA	110	000	110	00001
R7 ← R1	R1	-	R7	TSFA	001	000	111	00000
Output ← R2	R2	–	None	TSFA	010	000	000	00000
Output ← Input	Input	-	None	TSFA	000	000	000	00000
R4 ← shl R4	R4	-	R4	SHLA	100	000	100	11000
R5 ← 0	R5	R5	R5	XOR	101	101	101	01100