
Presented by
Lecture-1

Architecture of Simple As Possible computer (SAP-1)

Todays Topics

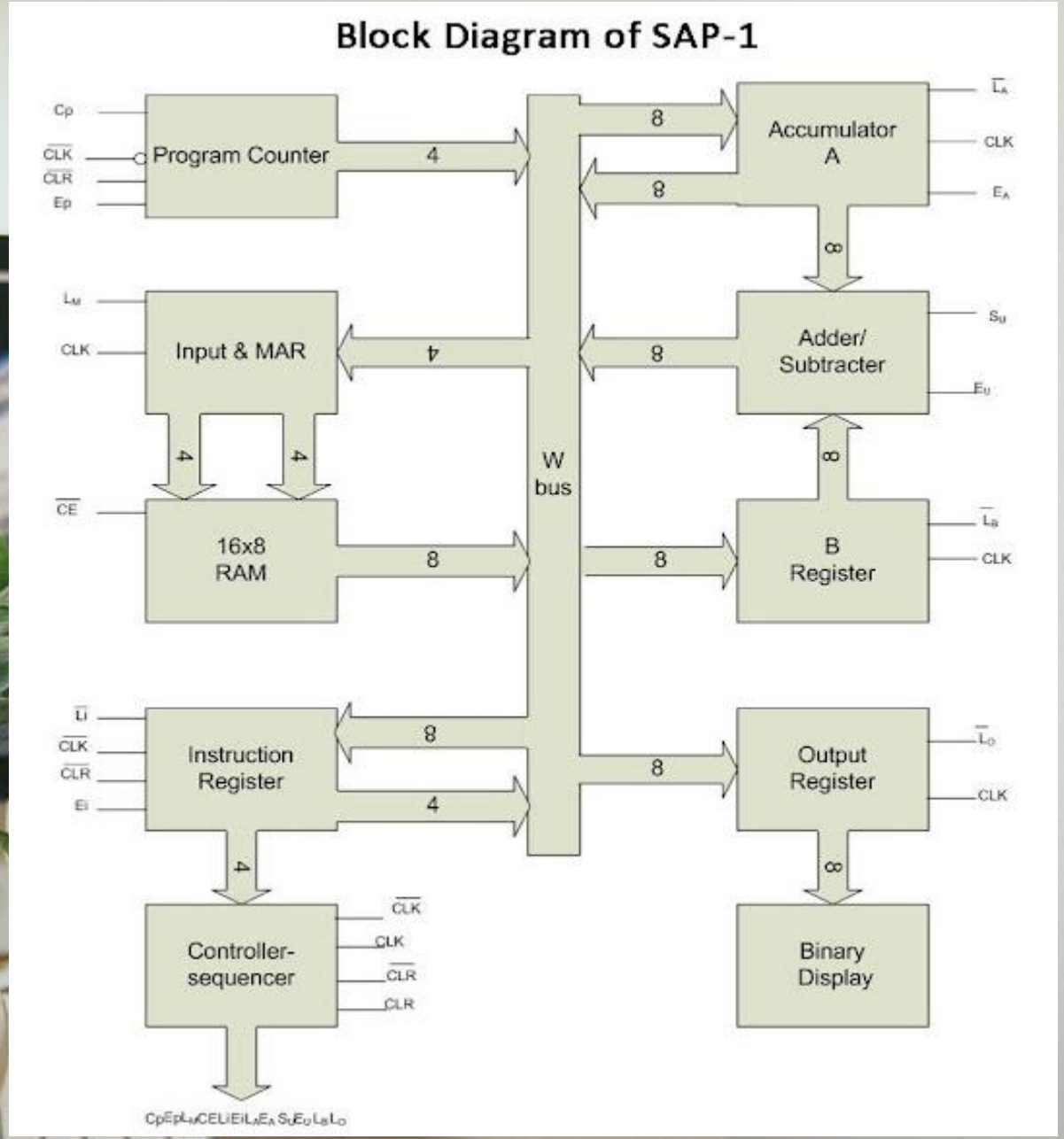
1.1 Define computer architecture
1.2 Describe architecture of Simple As Possible computer (SAP-1)
1.3 Describe function of control bits of SAP-1 Controller/Sequencer
1.4 Describe function of each instruction of SAP-1 computer
1.5 Write basic programs using SAP-1 instruction

Define computer architecture:

can be defined as a set of rules and methods that describe the functionality, management and implementation of computers. To be precise, it is nothing but rules by which a system performs and operates.

SAP-1 Computer Architecture

The SAP-1 computer is a bus-organized computer and makes use of Von-Neumann architecture. It makes use of an 8-bit central bus and has ten main components. A pictorial representation of its architecture is shown below. Each of the individual components that make up this computer are described right after.



SAP-1 Components

1.Program Counter: The program counter's job is to store and send out the memory address of the next instruction to be fetched and executed. The program counter, which is part of the control unit, counts from 0000 to 1111 as the program is stored at the beginning of the memory with the first instruction at binary address 0000, the second instruction at address 0001, the third at address 0010, and so on. At the start of each computer run, the program counter is reset to 0000. When the computer run starts, the program counter sends out the address 0000 to the memory and is then incremented by 1. After the first instruction is fetched and executed, the program counter sends the next address 0001 to the memory and again, after that, the program counter is incremented. In this way, the program counter keeps track of the next instruction to be fetched and executed.

2.Input and Memory Address Register (MAR): The MAR stores the 4-bit address of data or instruction which are placed in memory. When the SAP-1 is running, the 4-bit address is gotten from the Program Counter through the W-bus and then stored. This stored address is sent to the RAM where data or instructions are read from.

3.Random-Access Memory (RAM): The SAP-1 makes use of a 16 x 8 RAM (16 memory locations each storing 8 bits of data). The RAM can be programmed by means of the address and data switches allowing you to write to the memory before a computer run. During a computer run, the RAM receives its 4-bit address from the MAR and read operation is performed. In this way the instruction or data word stored in the RAM is placed on the W bus for use in some other part of the computer.

4.Instruction Register: The instruction receives and stores the instruction placed on the bus from the RAM. The content of the instruction register are then split into two nibbles. The upper nibble is a two-state output that goes into the Controller-sequencer while the lower nibble is a three-state output that is read from the bus when needed.

SAP-1 Components

5.Controller-Sequencer: The controller-sequencer sends out signals that control the computer and makes sure things happen only when they are supposed to. The 12 bit output signals from controller-sequencer is called the control word which determines how the registers will react to the next positive clock edge. It has the following format: ` CON = Cp Ep ~Lm ~CE ~Li ~Ei ~La Eu Su Eu ~Lb ~Lo`

6.Accumulator: The accumulator is an 8-bit buffer register that stores intermediate answers during a computer run. The accumulator has two outputs. The two-state output goes directly to the adder-subtractor and the three-state output goes to the bus. This implies that the 8-bit accumulator word continuously drives the adder-subtractor but only appears on the W bus when Ea is high.

7.Adder-Subtractor: The adder-subtractor asynchronously adds to or subtracts a value from the the accumulator depending on the value of Su. It makes use of 2's complement to achieve this When Su is low the output of the adder-subtractor is the sum of the values in the accumulator and in the B-register (O/P = A + B). When Su is high, the output is the difference between them (O/P = A + B').

8.B-Register: The B-register is a buffer register used in performing arithmetic operations. It supplies the number to be added or subtracted from the contents of the accumulator to the adder/subtractor. When data is available at the bus and Lb is low, at the positive clock edge, B register gets and stores the data.

9.Output Register: The output register gets and stores the value stored in the accumulator usually after the performance of an arithmetic operation. The answer that is stored in the accumulator is loaded into the output register through the W bus. This is done in the next positive clock edge when Ea is high and Lo is low. The processed data can now be displayed to the outside world.

10.Binary Display: The binary display is row of eight light emitting diodes(LEDs). The binary display shows us the contents of the output by connecting each LED to the output of the output register. This therefore enables viewing of the answer transferred from the accumulator to the output register in binary.

SAP-1 Instruction Set

The instruction set of a computer are the basic operations it can perform.
The instruction set of the SAP-1 is described in table below.

Operation	Description
Load	Load data from memory to the accumulator
Add	Add data from memory to value in the accumulator
Subtract	Subtract data from memory from value in the accumulator
Output	Load data from accumulator to the output register
Halt	Stop processing

Basic programs using SAP-1 instruction

Program to Add two numbers stored in memory :

```
LDA 00    ; Load accumulator with the contents at memory location 00
ADD 01    ; Add the contents at memory location 01 to the accumulator
STA 02    ; Store the result in memory location 02
HLT       ; Halt the program
```

Program to Subtract two numbers stored in memory::

```
LDA 00    ; Load accumulator with the contents at memory location 00
SUB 01    ; Subtract the contents at memory location 01 from the accumulator
STA 02    ; Store the result in memory location 02
HLT       ; Halt the program
```

Program to Multiply two numbers stored in memory:

```
LDA 00 ; Load accumulator with the contents at memory location 00
MUL 01 ; Multiply the contents at memory location 01 with the accumulator
STA 02 ; Store the result in memory location 02
HLT ; Halt the program
```


Presented by
Lecture-2

Basics of Computer Architecture

Todays Topics

2.1 Describe organization of Stored-program Computer system
2.2 Describe basic instruction types
2.3 Explain Expanding and Huffman op-code Encoding techniques
2.4 Compare between RISC and CISC
2.5 State different techniques of Parallel processing
2.6 Describe architecture of General register, accumulator-based and Stack based processor

What Is Computer Architecture?

Computer architecture refers to the end-to-end structure of a computer system that determines how its components interact with each other in helping to execute the machine's purpose (i.e., processing data), often avoiding any reference to the actual technical implementation.

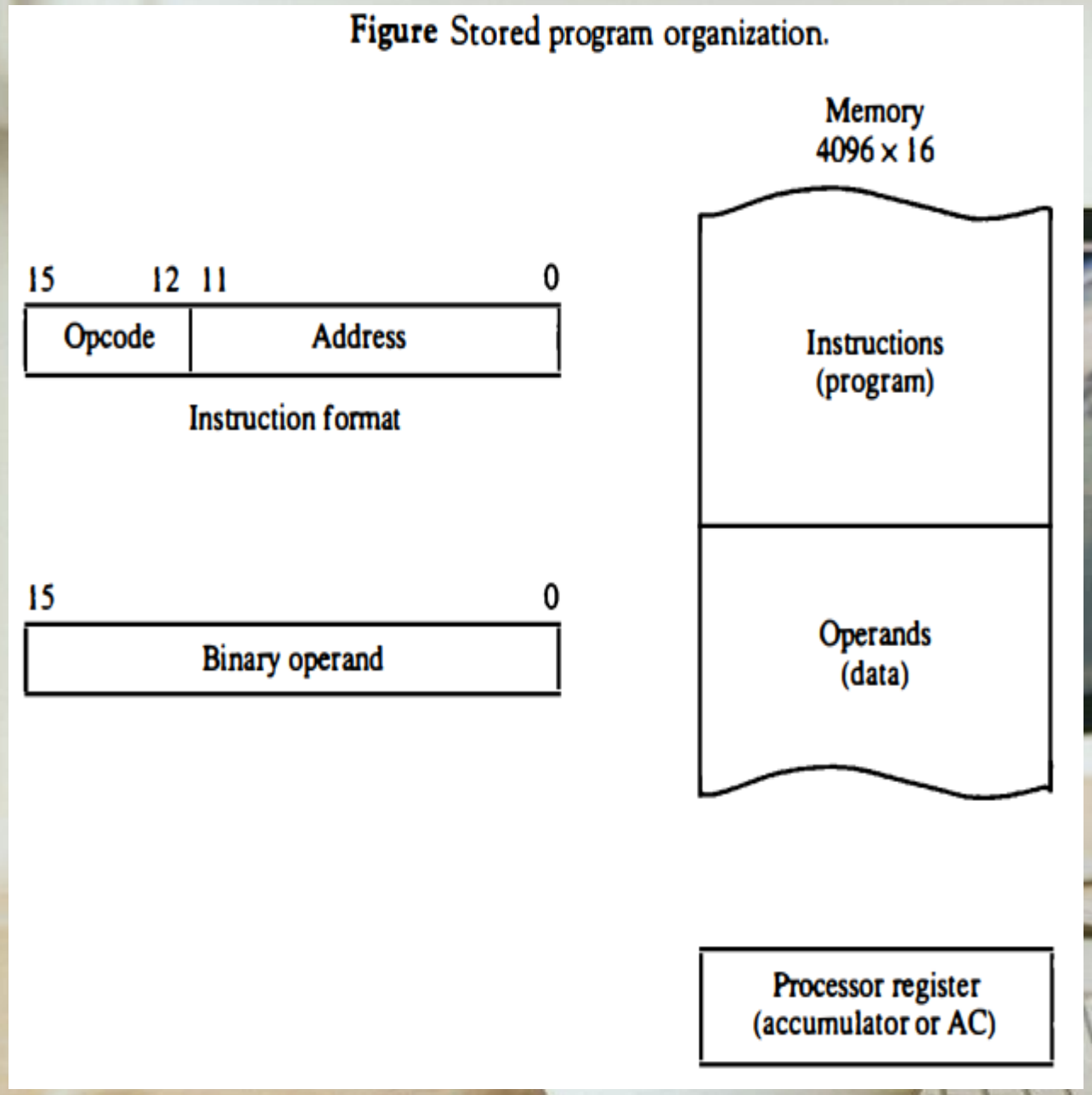
Basic Computer Organization and Design

- **The organization of the computer** is defined by its internal registers, timing and control structures, and the sets of instructions it uses.
- **The internal organization** of a digital system is defined by the sequence of micro-operations it performs on data stored in its registers.
- **The general purpose computer** is capable of executing various micro-operations and can be instructed as to what specific sequence of operations it must perform. The user of a computer can control the process by means of a program.
- **The ability to store and execute instructions**, the stored program concept, is the most important property of a general-purpose computer.
- **An instruction code is a group of bits** that instruct the computer to perform a specific operation. It is divided into parts, each having its own particular interpretation.
- **The operation part** of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory.
- **An instruction code** must therefore specify not only the operation but also the registers or the memory words where the operands are to be found, as well as the register or memory word where the result is to be stored.
- **Memory words** can be specified in instruction codes by their address. Processor registers can be specified by assigning to the instruction another binary code of k bits that specifies one of 2^k registers.
- **There are many variations** for arranging the binary code of instructions, and each computer has its own particular instruction code format. Instruction code formats are conceived by computer designers who specify the architecture of the computer.

Stored Program Organization

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts.

- **The first part** specifies the operation to be performed and the second specifies an address.
- **The memory address** tells the control where to find an operand in memory.
- **This operand** is read from memory and used as the data to be operated on together with the data stored in the processor register.
- **Figure below** depicts this type of organization. Instructions are stored in one section of memory and data in another. For a memory unit with 4096 words we need 12 bits to specify an address since $2^{12} = 4096$. If we store each instruction code in one 16-bit memory word, we have available four bits for the operation code (abbreviated opcode) to specify one out of 16 possible operations, and 12 bits to specify the address of an operand.
- **The control** reads a 16-bit instruction from the program portion of memory. It uses the 12-bit address part of the instruction to read a 16-bit operand from the data portion of memory.
- **It then executes** the operation specified by the operation code.



Computers that have a single-processor register usually assign to it the name accumulator and label it AC. The operation is performed with the memory operand and the content of AC.

•**If an operation** in an instruction code does not need an operand from memory, the rest of the bits in the instruction can be used for other purposes. For example, operations such as clear AC, complement AC, and increment AC operate on data stored in the AC register.

•**They do not need an operand** from memory. For these types of operations, the second part of the instruction code (bits 0 through 11) is not needed for specifying a memory address and can be used to specify other operations for the computer.

Huffman Coding

Huffman coding is a method of variable-length coding (VLC) in which shorter codewords are assigned to the more frequently occurring symbols to achieve an average symbol codeword length that is as close to the symbol source entropy as possible.

Encoding Given a code (corresponding to some alphabet Γ) and a message it is easy to encode the message. Just replace the characters by the codewords. Example: Γ = {a, b, c, d} If the code is C1{a = 00, b = 01, c = 10, d = 11}. then bad is encoded into 010011 If the code is C2 = {a = 0, b = 110, c = 10, d = 111} then bad is encoded into 1101111

Decoding C1 = {a = 00, b = 01, c = 10, d = 11}. C2 = {a = 0, b = 110, c = 10, d = 111}. C3 = {a = 1, b = 110, c = 10, d = 111} Given an encoded message, decoding is the process of turning it back into the original message. A message is uniquely decodable (vis-a-vis a particular code) if it can only be decoded in one way. For example relative to C1, 010011 is uniquely decodable to bad. Relative to C2 1101111 is uniquely decodable to bad. But, relative to C3, 1101111 is not uniquely decipherable since it could have encoded either bad or acad. In fact, one can show that every message encoded using C1 and C2 are uniquely decipherable. The unique decipherability property is needed in order for a code to be useful.

What Is CISC?

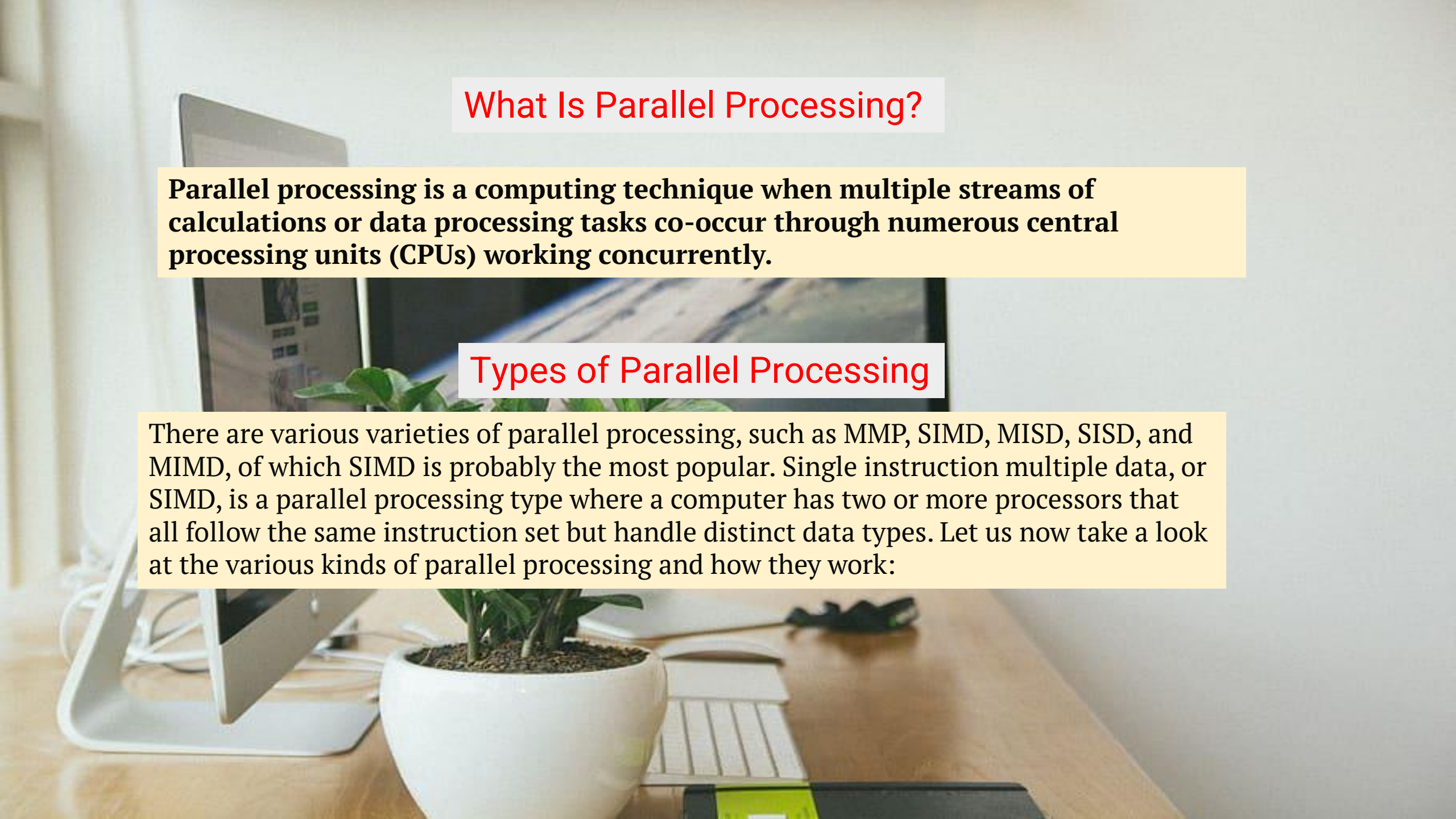
CISC stands for ‘Complex Instruction Set Computer.’ This architecture was introduced in the 1970s by Intel Corporation when the earliest computers focused on enhancing CPU speed by minimizing the number of instructions per program (as per equation 1). This objective was achieved by combining multiple simple commands into one complex instruction.

What Is RISC?

RISC stands for ‘Reduced Instruction Set Computer.’ These were introduced in the 1980s by David Patterson and John Hennessy to overcome the complexities of CISC processors. RISC processors work with more instructions; however, the number of cycles an instruction may take to execute is minimized. In general terms, a RISC machine takes one CPU cycle to complete one instruction. This may equate to the ‘sleep’ instruction you gave your trained pet.

Difference between RISC and CISC Processor

S.No.	RISC	CISC
1.	RISC is a reduced instruction set.	CISC is a complex instruction set.
2.	The number of instructions is less as compared to CISC.	The number of instructions is more as compared to RISC.
3.	The addressing modes are less.	The addressing modes are more.
4.	It works in a fixed instruction format.	It works in a variable instruction format.
5.	The RISC consumes low power.	The CISC consumes high power.
6.	The RISC processors are highly pipelined.	The CISC processors are less pipelined.
7.	It optimizes the performance by focusing on software.	It optimizes the performance by focusing on hardware.
8.	Requires more RAM.	Requires less RAM.



What Is Parallel Processing?

Parallel processing is a computing technique when multiple streams of calculations or data processing tasks co-occur through numerous central processing units (CPUs) working concurrently.

Types of Parallel Processing

There are various varieties of parallel processing, such as MMP, SIMD, MISD, SISD, and MIMD, of which SIMD is probably the most popular. Single instruction multiple data, or SIMD, is a parallel processing type where a computer has two or more processors that all follow the same instruction set but handle distinct data types. Let us now take a look at the various kinds of parallel processing and how they work:

Types of Parallel Processing

1. Single Instruction, Single Data (SISD)

In the type of computing called Single Instruction, Single Data (SISD), a single processor is responsible for simultaneously managing a single algorithm as a single data source. A computer organization having a control unit, a processing unit, and a memory unit is represented by SISD. It is similar to the current serial computer. Instructions are carried out sequentially by SISD, which may or may not be capable of parallel processing, depending on its configuration.

Sequentially carried-out instructions may cross over throughout their execution phases. There may be more than one functional unit inside a SISD computer. However, one control unit is in charge of all functional units. Such systems allow for pipeline processing or using numerous functional units to achieve parallel processing.

2. Multiple Instruction, Single Data (MISD)

Multiple processors are standard in computers that use the Multiple Instruction, Single Data (MISD) instruction set. While using several algorithms, all processors share the same input data. MISD computers can simultaneously perform many operations on the same batch of data. As expected, the number of operations is impacted by the number of processors available.

The MISD structure consists of many processing units, each operating under its instructions and over a comparable data flow. One processor's output becomes the input for the following processor. This organization's debut garnered little notice and wasn't used in architecture.

3. Single Instruction, Multiple Data (SIMD)

Computers that use the Single Instruction, Multiple Data (SIMD) architecture have multiple processors that carry out identical instructions. However, each processor supplies the instructions with its unique collection of data. SIMD computers apply the same algorithm to several data sets. The SIMD architecture has numerous processing components.

All of these components fall under the supervision of a single control unit. While processing numerous pieces of data, each processor receives the same instruction from the control unit. Multiple modules included in the shared subsystem aid in simultaneous communication with every CPU. This is further separated into organizations that use bit-slice and word-slice modes.

Types of Parallel Processing

4. Multiple Instruction, Multiple Data (MIMD)

Multiple Instruction, Multiple Data, or MIMD, computers are characterized by the presence of multiple processors, each capable of independently accepting its instruction stream. These kinds of computers have many processors. Additionally, each CPU draws data from a different data stream. A MIMD computer is capable of running many tasks simultaneously.

Although MIMD computers are more adaptable than SIMD or MIMD computers, developing the sophisticated algorithms that power these machines is more challenging. Since all memory flows are changed from the shared data area transmitted by all processors, a MIMD computer organization incorporates interactions between the multiprocessors.

The multiple SISD operation is equivalent to a collection of separate SISD systems if the many data streams come from various shared memories.

5. Single Program, Multiple Data (SPMD)

SPMD systems, which stand for Single Program, Multiple Data, are a subset of MIMD. Although an SPMD computer is constructed similarly to a MIMD, each of its processors is responsible for carrying out the same instructions. SPMD is a message passing programming used in distributed memory computer systems. A group of separate computers, collectively called nodes, make up a distributed memory computer.

Each node launches its application and uses send/receive routines to send and receive messages when interacting with other nodes. Systems can also use messages to provide barrier synchronization. It is possible to transfer the messages via a wide range of communication techniques, such as transmission control protocol (TCP/IP) over Ethernet and specialized high-speed interconnects, such as Supercomputer Interconnect and Myrinet.

6. Massively Parallel Processing (MPP)

A storage structure called Massively Parallel Processing (MPP) is made to manage the coordinated execution of program operations by numerous processors. With each CPU using its operating system and memory, this coordinated processing can be applied to different program sections.

As a result, MPP databases can handle enormous amounts of data and deliver analyses based on large datasets considerably faster. MPP processors typically communicate through a messaging interface and can have up to 200 or more processors working on an application. It functions by enabling the transmission of messages between processes via a set of corresponding data links.

The most common types of computers used in parallel processing systems are SIMD and MIMD. Although SISD computers can’t run in parallel on their own, a cluster can be created by connecting many of them. In a more extensive parallel system, the CPU of each computer can function as a processor. The computers work as a single supercomputer when used collectively. Grid computing is the name of this method.

Register-based and accumulator based architecture

A register-based CPU architecture has one or more general-purpose registers (where “general purpose register” excludes special-purpose registers, like stack pointer and instruction pointer).

Register Based Architecture

•All the arithmetic and logical operation consist of the operands of any registers (ABCD etc). The input/output operations here register A and register B are similar.

• The internal architecture of 8086 shows that the registers A, B, C, D and others directly with the ALU.

• Data can enter into the ALU from any registers.

• For I/O operation register A only can be used.

• The advantage of register-based architecture is extendibility and flexibility in programming.

· The processor will be enhanced in register-based architecture.

• The disadvantage is the requirement of complex circuitry.

An accumulator-based CPU architecture is a register-based CPU architecture that only has one general-purpose register (the accumulator).

Accumulator Based Architecture

• An accumulator is a most significant register then compared to other registers and most of the arithmetic and logic operations are performed using the accumulator performed via the accumulator.

•The internal architecture of 8085 shows that the registers B, C, D, E, H, and L are connected with the ALU through the accumulator and temporary register.

• Data can only enter into the ALU from the accumulator and the output of the ALU can be stared in accumulator through the data bus.


Presented by
Lecture-3

Basics of CPU design

Todays Topics

3.1 Interpret basic function of ALU and Control unit
3.2 Describe a typical CPU model
3.3 Explain the design of 4-bit General Register and 4-bit Parallel Adder
3.4 Discuss simple organization of a 4-bit Arithmetic unit
3.5 Discuss simple organization of a two function Logic unit
3.6 Explain the design structure of a 4-bit ALU
3.7 Describe the instruction interpretation and instruction sequencing of control unit
3.8 Illustrate Hardwired & Microprogramming approach for control unit design
3.9 Describe the techniques of coprocessor interfacing

Designing a central processing unit (CPU) is a complex and challenging task that requires a deep understanding of computer architecture, electrical engineering, and digital logic design. A CPU, also known as a microprocessor, is the brain of a computer and is responsible for executing instructions and performing various operations

The characteristics of the ALU are as follows:

- The ALU is responsible for performing all logical and arithmetic operations.

- Some of the arithmetic operations are as follows: addition, subtraction, multiplication and division.

- Some of the logical operations are as follows: comparison between numbers, letter and or special characters.

- The ALU is also responsible for the following conditions: Equal-to conditions, Less-than condition and greater than condition.

The characteristics of the CU or control unit are as follows:

- This part of the of the CPU is the one that is in charge of all the operations being carried out.

- It is responsible to direct the system to execute instructions.

- It helps in communication between the memory and the arithmetic logical unit.

- It also aids in the loading of data and instructions residing in the secondary memory to the main memory as required.

A typical CPU (Central Processing Unit) model is a crucial component of a computer or electronic device responsible for executing instructions from programs and performing arithmetic and logic operations. While there are various CPU models from different manufacturers, they generally share common characteristics. Here's a description of a typical CPU:

Architecture: CPUs are built based on specific architectures, such as x86, ARM, or RISC. The architecture dictates the design principles and instruction set that the CPU follows.

Clock Speed: This represents how quickly the CPU can execute instructions, measured in gigahertz (GHz). Higher clock speeds generally indicate faster processing capabilities.

Cores: Modern CPUs often have multiple processing cores, allowing them to handle multiple tasks simultaneously through parallel processing. Dual-core, quad-core, hexa-core, and octa-core configurations are common.

Threads: CPUs can support multiple threads per core, allowing for more efficient multitasking. Hyper-Threading (in Intel CPUs) and Simultaneous Multithreading (SMT, in AMD CPUs) are technologies that enable a single core to handle multiple threads.

Cache Memory: CPUs have multiple levels of cache memory (L1, L2, and L3) to store frequently accessed data and instructions, reducing the time needed to fetch information from the slower main memory.

Instruction Set: CPUs follow a specific instruction set architecture (ISA) that defines the set of instructions they can execute. Common ISAs include x86, x86-64, and ARM.

Manufacturing Process: CPUs are manufactured using a specific nanometer (nm) process technology, indicating the size of the transistors on the chip. Smaller nanometer processes generally lead to more power-efficient and faster CPUs.

Power Consumption: CPUs come with a Thermal Design Power (TDP) rating, indicating the maximum amount of heat the CPU is expected to generate under normal operation. Lower TDP values are generally associated with more power-efficient CPUs.

Socket Type: CPUs are designed to fit into specific sockets on the motherboard. Common socket types include Intel's LGA (Land Grid Array) and AMD's PGA (Pin Grid Array).

Compatibility: CPUs need to be compatible with the motherboard and other system components. This includes considerations for the chipset, memory type, and other features.

Integrated Graphics: Some CPUs come with integrated graphics processing units (GPUs), eliminating the need for a separate graphics card for basic display tasks.

Popular CPU manufacturers include Intel and AMD, each producing a wide range of models catering to different performance needs and budgets. The specific features and performance of a CPU can vary significantly based on the model and intended use.

4-bit register:

Various types of registers are available commercially. The simplest register is one that consists only of flip-flops, with no external gates. The 4-bit register circuit diagram shows that such a register is constructed with four D flip-flops.

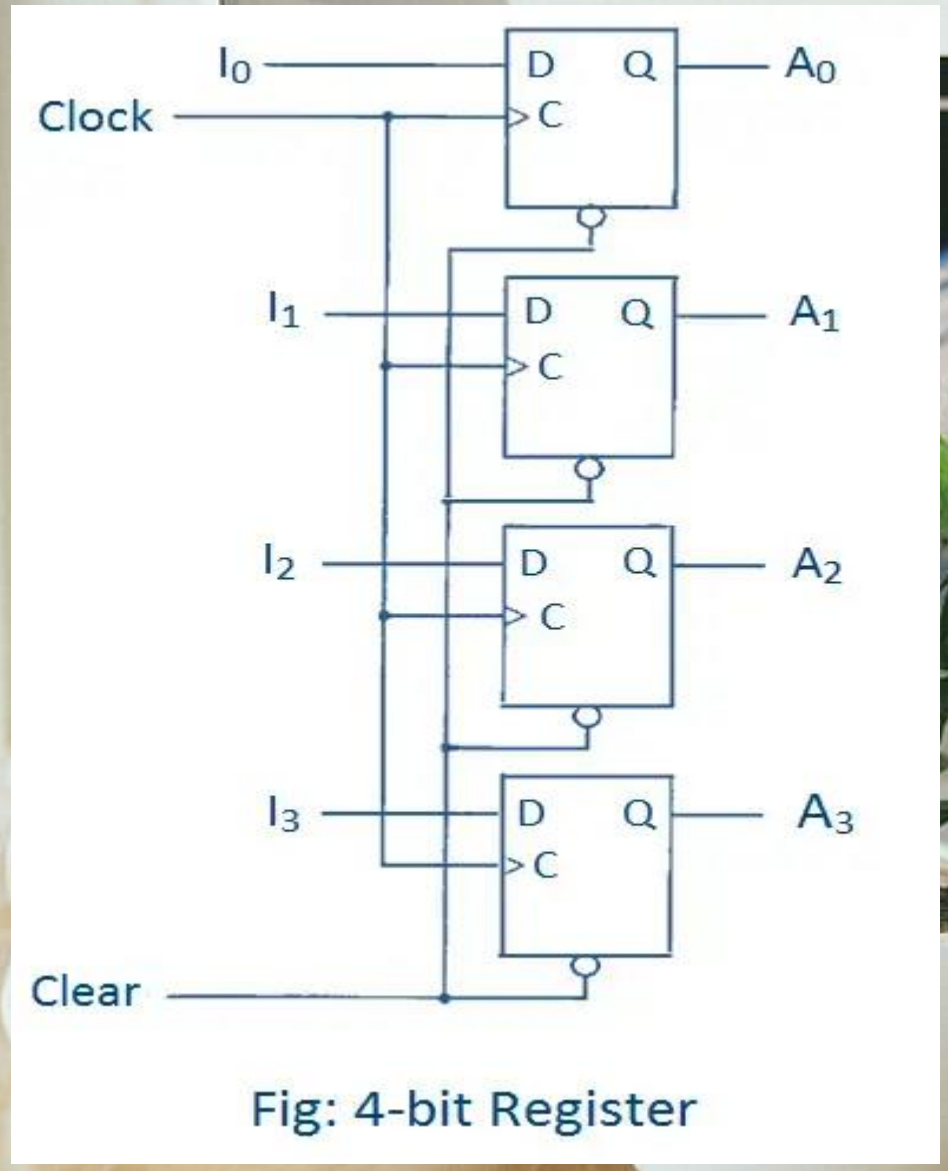


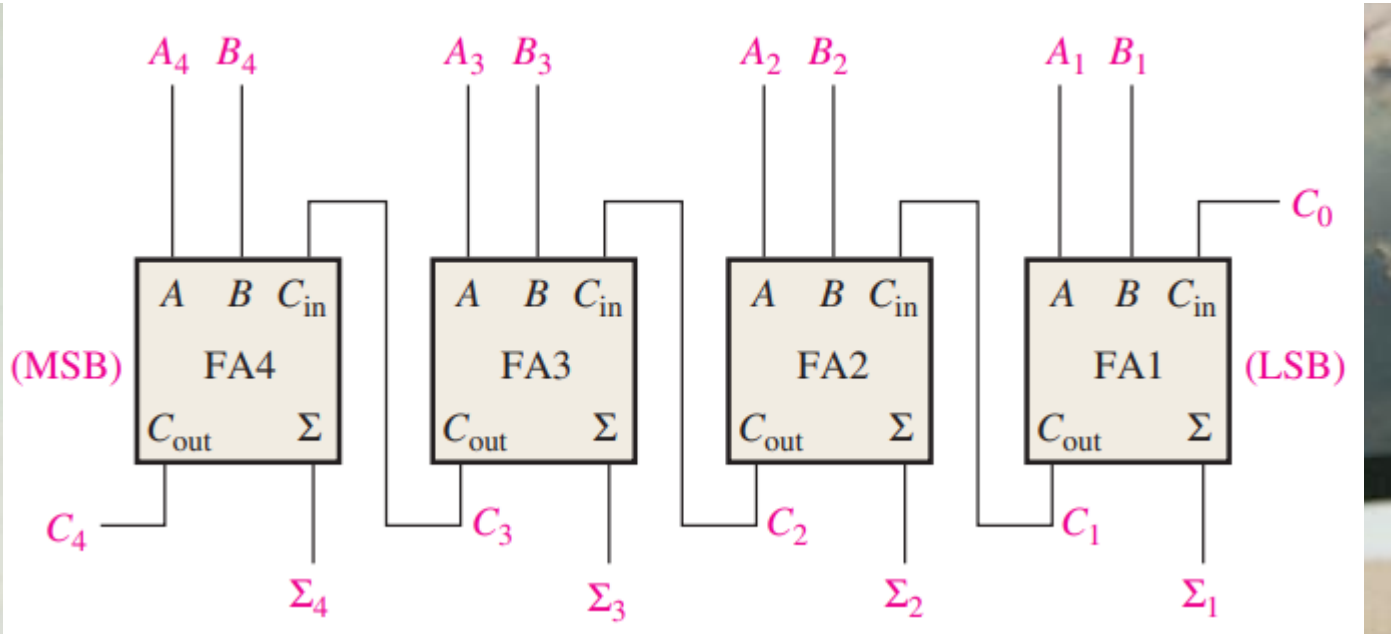
Fig: 4-bit Register

The common clock input triggers all flip-flops on the rising edge of each pulse, and the binary data available at the four inputs are transferred into the 4-bit register. The four outputs can be sampled at any time to obtain the binary information stored in the register. The clear input goes to a special terminal in each flip-flop. When this input goes to 0, all flip-flops are reset asynchronously. The clear input is useful for clearing the register to all O's before its clocked operation. The clear input must be maintained at logic 1 during normal clocked operation.

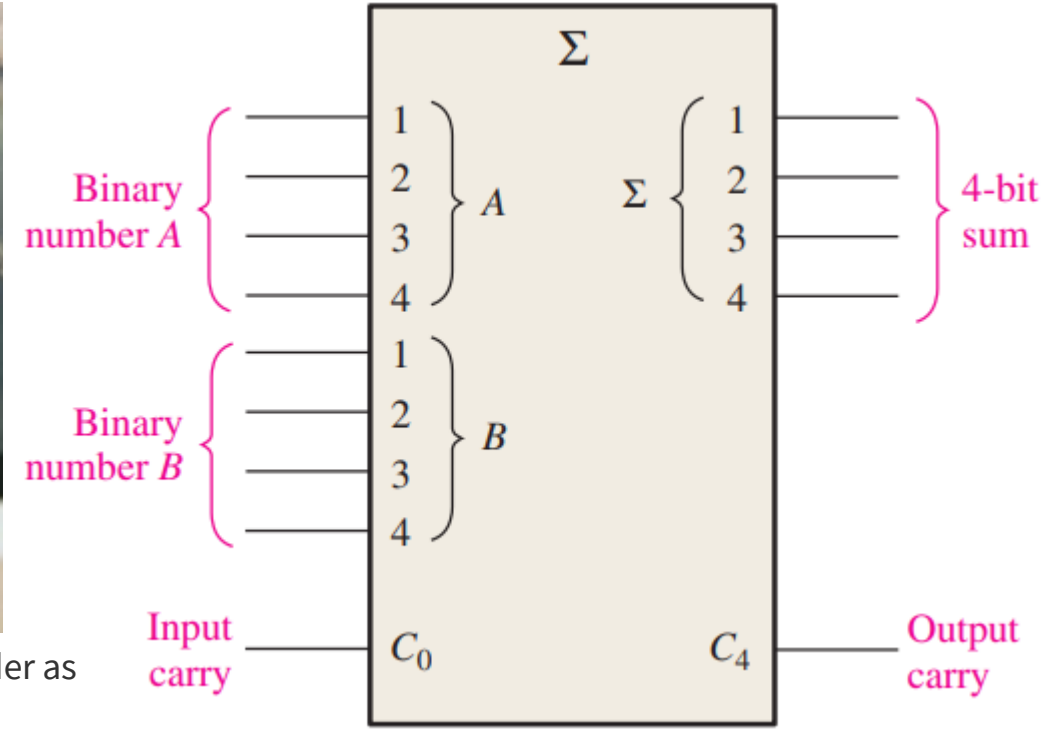
Four-Bit Parallel Adders:

•A group of four bits is called a nibble. A basic 4-bit parallel adder is implemented with four full-adder stages as shown in Figure (a).

•Again, the LSBs (A1 and B1) in each number being added go into the right-most full-adder; the higher-order bits are applied as shown to the successively higher-order adders, with the MSBs (A4 and B4) in each number being applied to the left-most full-adder.



The carry output of each adder is connected to the carry input of the next higher-order adder as indicated. These are called internal carries



•In terms of the method used to handle carries in a parallel adder, there are two types: the ripple carry adder and the carry look-ahead adder. These are discussed in Section (a).

•In keeping with most manufacturers' data sheets, the input labeled C0 is the input carry to the least significant bit adder; C4 , in the case of four bits, is the output carry of the most significant bit adder; and ©1 (LSB) through ©4 (MSB) are the sum outputs. The logic symbol is shown in Figure (b).

Truth Table for a 4-Bit Parallel Adder:

•Table (b) is the truth table for a 4-bit adder. On some data sheets, truth tables may be called function tables or functional truth tables.

•The subscript n represents the adder bits and can be 1, 2, 3, or 4 for the 4-bit adder.

C_{n-1}	A_n	B_n	Σ_n	C_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

•Cn-1 is the carry from the previous adder.

•Carries C1 , C2 , and C3 are generated internally.

•C0 is an external carry input and C4 is an output. Example illustrates how to use Table (b).

Thanks

Please