

217529- Internet of Things

Unit Number: 1

Unit Name: **Fundamentals of Computer Organization & Digital Electronics**

Unit Outcomes: CO1

Have a thorough understanding of the structure, function and characteristics of computer systems and Understand the structure of various number systems and its application in digital design.

Syllabus

- Basic Organization of Computers, Classification Micro, Mini, Mainframe and Super Computer. System Bus and Interconnection, PCI, Computer Function, I-Cycle, Interrupt and Class of Interrupts.
- Number systems, Decimal Number system, Binary number system, Octal & Hexadecimal number system, 1's & 2's complement, Binary Fixed Point Representation.

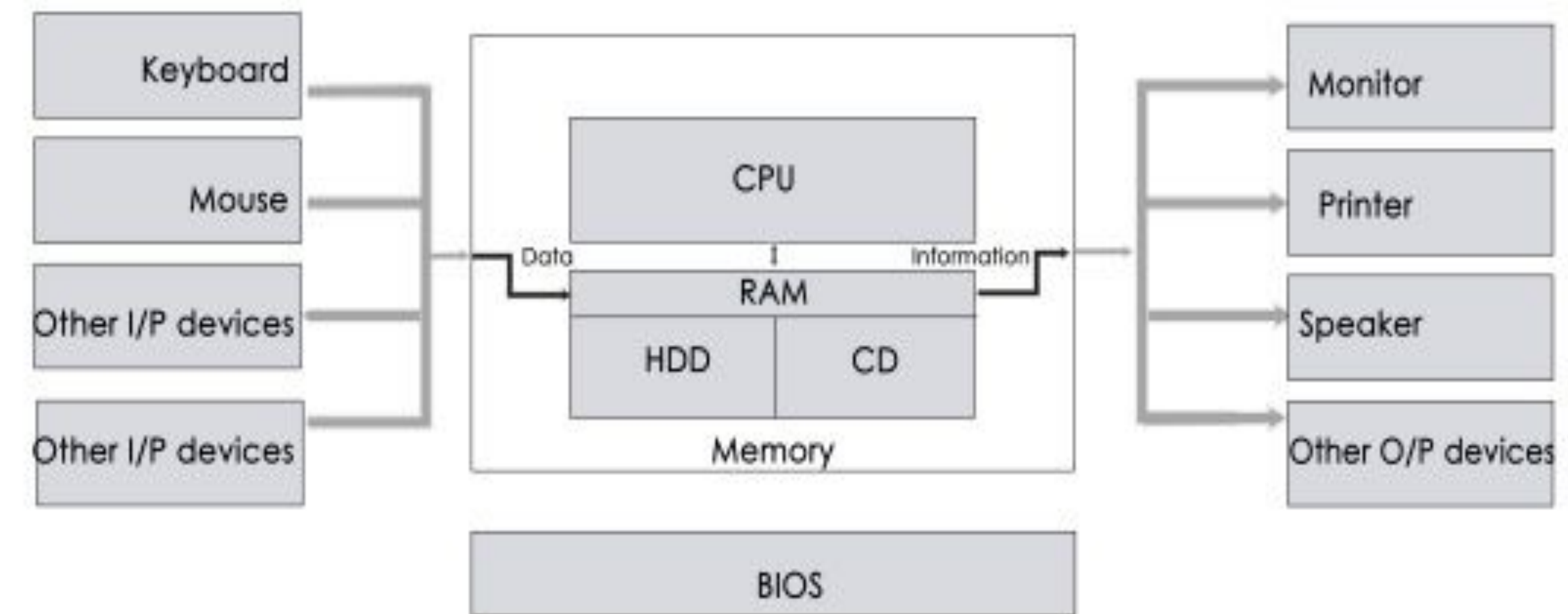
What is Computer?

❑ A computer is an electronic device that manipulates information, or data. It has the ability to store, retrieve, and process data. You may already know that you can use a computer to type documents, send email, play games, and browse the Web. You can also use it to edit or create spread sheets, presentations, and even videos.

❑ What is Hardware and software's in computer?

❑ Hardware refers to the physical and visible components of the system such as a monitor, CPU, keyboard and mouse.

❑ Software, on the other hand, refers to a set of instructions which enable the hardware to perform a specific set of tasks.



Generations of computers



GENERATION OF COMPUTERS

1
GEN

FIRST-GENERATION COMPUTERS (1940-1956)
based on vacuum tubes.

2
GEN

SECOND-GENERATION COMPUTERS (1956-1963)
transistors was used

3
GEN

THIRD-GENERATION COMPUTERS (1964-1971)
Integrated Circuits (IC) was used

4
GEN

FOURTH GENERATION COMPUTERS (1971-1980)
Very Large Scale Integrated (VLSI) circuits was used

5
GEN

FIFTH-GENERATION COMPUTERS (1980-PRESENT)
ULSI (Ultra Large Scale Integration) technology was used

| Generation (Period) | Key hardware technologies | Key software technologies | Key characteristics | Some representative systems |
|----------------------|---|---|---|---|
| Fifth (1989-Present) | <ul style="list-style-type: none">▪ ICs with ULSI technology▪ Larger capacity main memory, hard disks with RAID support▪ Optical disks as portable read-only storage media▪ Notebooks, powerful desktop PCs and workstations▪ Powerful servers, supercomputers▪ Internet▪ Cluster computing | <ul style="list-style-type: none">▪ Micro-kernel based, multithreading, distributed OS▪ Parallel programming libraries like MPI & PVM▪ JAVA▪ World Wide Web▪ Multimedia, Internet applications▪ More complex supercomputing applications | <ul style="list-style-type: none">▪ Portable computers▪ Powerful, cheaper, reliable, and easier to use desktop machines▪ Powerful supercomputers▪ High uptime due to hot-pluggable components▪ Totally general purpose machines▪ Easier to produce commercially, easier to upgrade▪ Rapid software development possible | <ul style="list-style-type: none">▪ IBM notebooks▪ Pentium PCs▪ SUN Workstations▪ IBM SP/2▪ SGI Origin 2000▪ PARAM 10000 |

Basic Organization of Computers

❑ Any computer can perform the four basic operations of Input, Processing, Output, and Storage (IPOS).

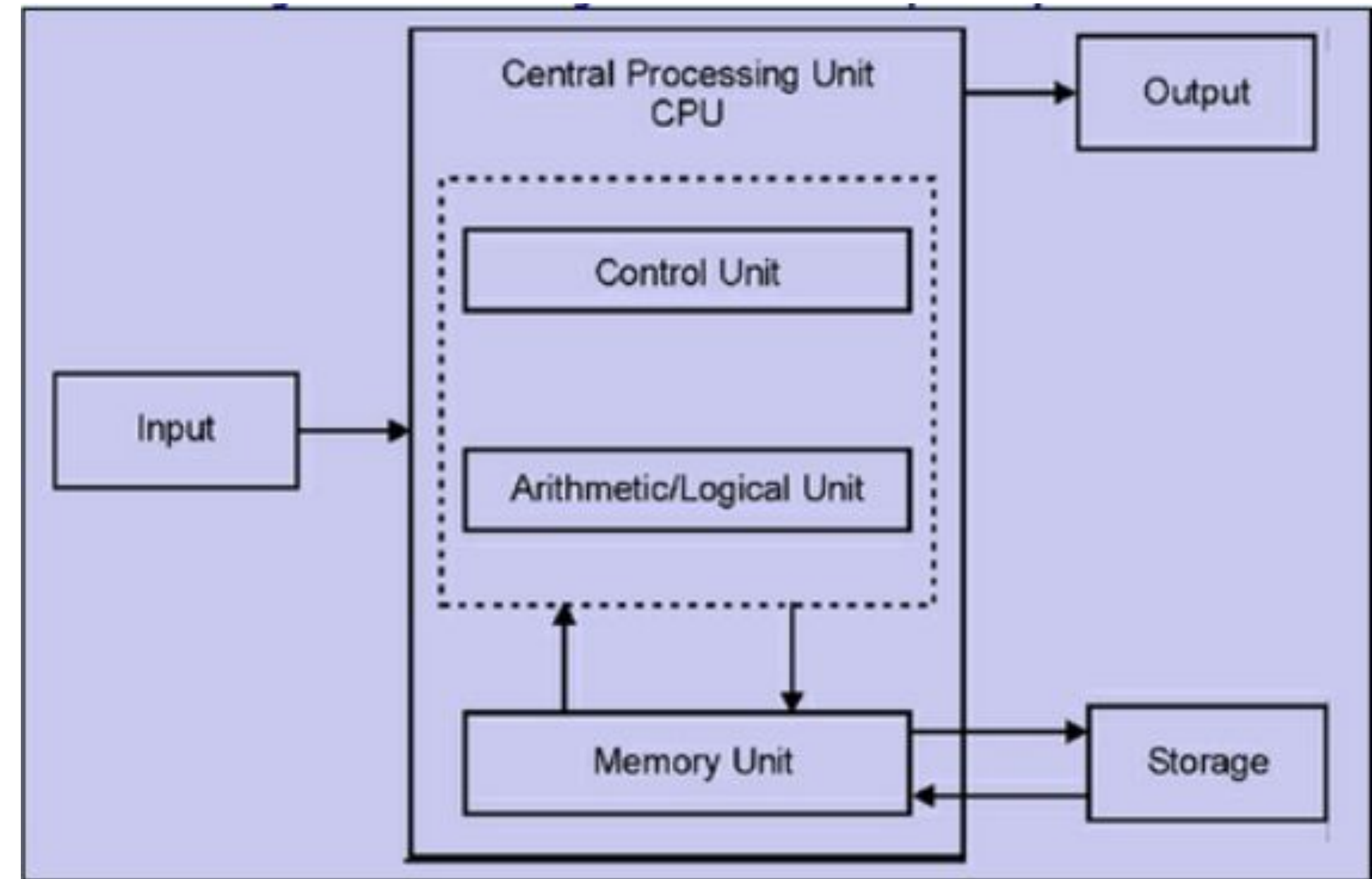
❑ The basic structure of the computer depends hugely on some of the aspects which are mentioned below

- Input Devices
- Output Devices
- Central Processing Unit
- ALU [Arithmetic Logic Unit]
- CU [Control Unit]

❑ Modern-day computers are used to store huge amounts of data and information almost permanently.

❑ This data or information can be retrieved and process on request and demand for further modification and implementations.

❑ The processing speed of the computer is incredible these days thanks a lot to the CPU which in common terms is called Central Processing Unit.

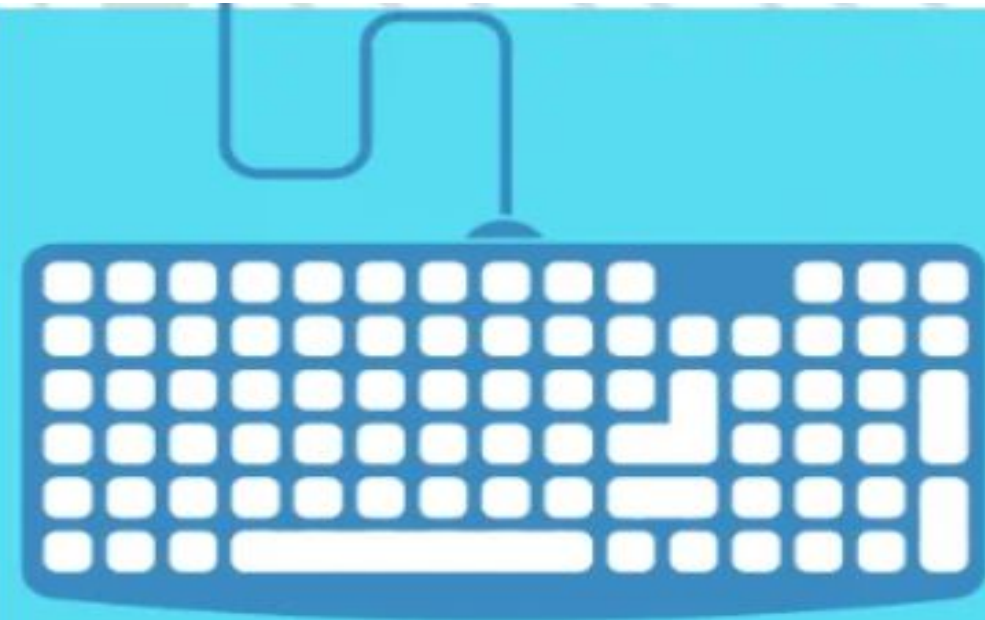


Cont...

1.) INPUT DEVICES

The input devices are nothing but small electronic devices that convert the digital signal or logical signals into binary form which are in human readable form.

Example - Keyboard, Mouse, Joystick, etc.



2.) OUTPUT DEVICES

The output devices are electronic devices or gadgets that accept data, instructions and information's from outside world and translates these data in such a form which can be easily read.

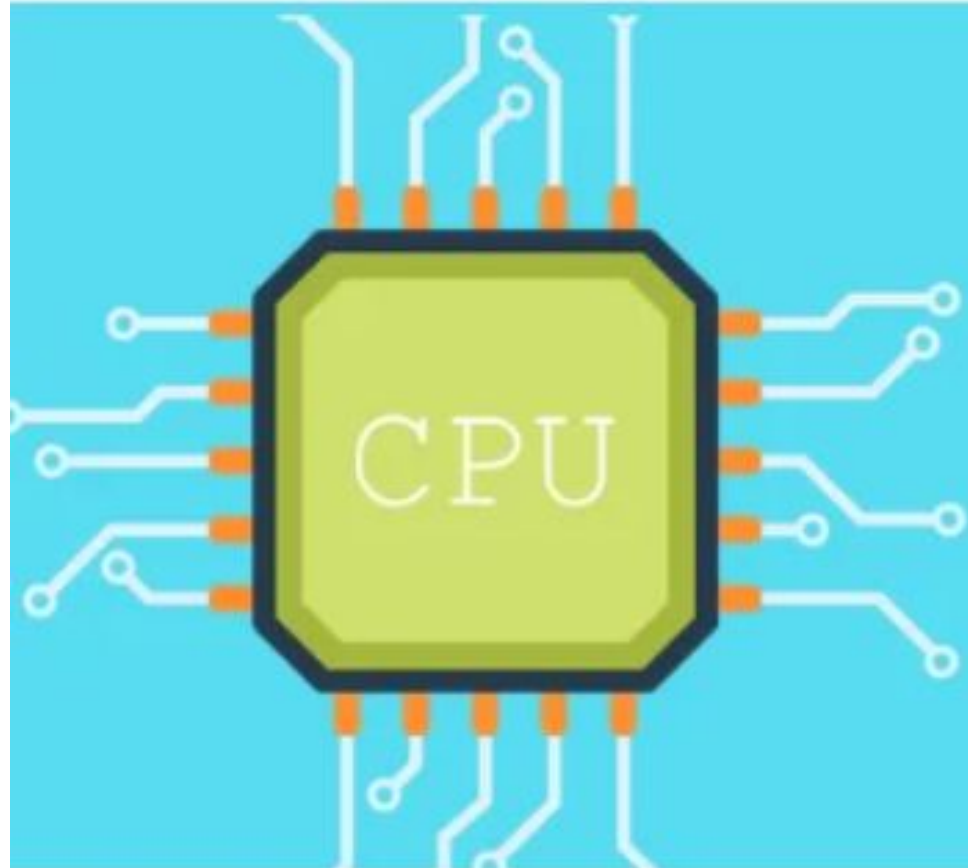
Example - Printer, VDU , Speakers, etc.



3.) CENTRAL PROCESSING UNIT

CPU stands for **Central Processing Unit** or **Processor** which is known as "**Heart of Computer**" or "**Brain of computer**". The CPU is nothing but a piece of electronic & hardware device which carries out all arithmetic and logical operations with high speed and unbelievable accuracy.

The famous CPU manufacturing Companies are INTEL & AMD.



4.) ARITHMETIC LOGIC UNIT [ALU]

ALU stands for Arithmetic Logic Unit

Arithmetic Operations Performs (Additions, Subtraction, Division, Multiplication) and Logical Operations executes operations like (And, OR , = , Comparison, Less, Than , Greater than).



5.) STORAGE UNIT

There are two types of storage or memory unit in computer.

- **Primary Storage - RAM**
- **Secondary Storage -HARDDISK**



- The CU represents Control Unit. The Control Unit controls all the activities and undertakings acted in the PC system.
- The control unit is mindful to execute all tasks specifically and consecutive requests.
- The CU figures out which activity is to perform before another undertaking or activity.
- The Control Unit gets guidelines from the memory and it changes over them into control flags later these signs are moved to the CPU for additional handling.

Classification of Computers

□ All computers classified in



Cont..

❑ Analog Computer

- ❑ They perform the task of measuring physical quantities such as pressure, temperature, length, and height, etc. and express their results in figures.
- ❑ They are used in the fields of science and engineering because the dimensions are more used in these areas.
- ❑ Speedometers, odometers, and mercury thermometers are examples of analog computers.

❑ Digital Computer

- ❑ A digit means a number, that is, a digital computer is a computer that counts digits. A computer using the principle of digital calculations can be called a digital computer.
- ❑ Digital computers have two digits 0 and 1, which is called binary number system, it works on the basis of these numbers. These two digits are called bits.
- ❑ We use digital computers for various applications like education, banking, business entertainment, etc. and nowadays it is also very popular

Cont..

❑ Hybrid Computer

- ❑ Computers that have characteristics of both analog and digital computers are called hybrid computers.
- ❑ Hybrid computers are mostly used in medicines such as the patient's blood pressure, To measure beats, etc. we first convert analog data into digital data using a hybrid computer and then the result is displayed on the screen in digital form.

❑ Classification of computers on basis of purpose

- ❑ Based on purpose, computers can be classified into two types. They are as follows:
- ❑ General-purpose computer
- ❑ Special-purpose computer

Cont..

❑ General-purpose computers

- ❑ The computers, which can be theoretically used for any type of application, are called general-purpose computers.

❑ Special-purpose computers

- ❑ The computers are made and used for specific jobs like air traffic control systems. Controlling fuel in automobiles is called special-purpose computers

❑ Classification of computers on basis of size and capability

The digital computers that are available nowadays vary in their sizes and types. Based on size and capability, the digital computer can be classified as:

- ❑ Super Computer
- ❑ Mainframe computer
- ❑ Mini Computer
- ❑ Micro Computer
- ❑ Workstation

Cont..

| r.No. | Type | Specifications |
|-------|----------------|--|
| 1 | Micro computer | <p>It is a single user computer system having moderately powerful microprocessor</p> <p>Eg desktops,laptops,PDA(tablets and Mobiles).</p> |
| 2 | Mini Computer | <p>It is a multi-user computer system which is capable of supporting hundreds of users simultaneously.Also known as host computers.</p> <p>Eg IBM AS/400,IBM SYSTEM 360</p> |
| 3 | Main Frame | <p>It is a multi-user computer system which is capable of supporting thousands and billion of users simultaneously. Famous with the name servers.</p> <p>Eg IBM S/390,amdalh 580</p> |
| 4 | Supercomputer | <p>It is an extremely fast computer which can execute hundreds of millions of instructions per second.</p> <p>Eg cray,param,control data cyber 205</p> |

Question Bank

❏ Question on this topic:

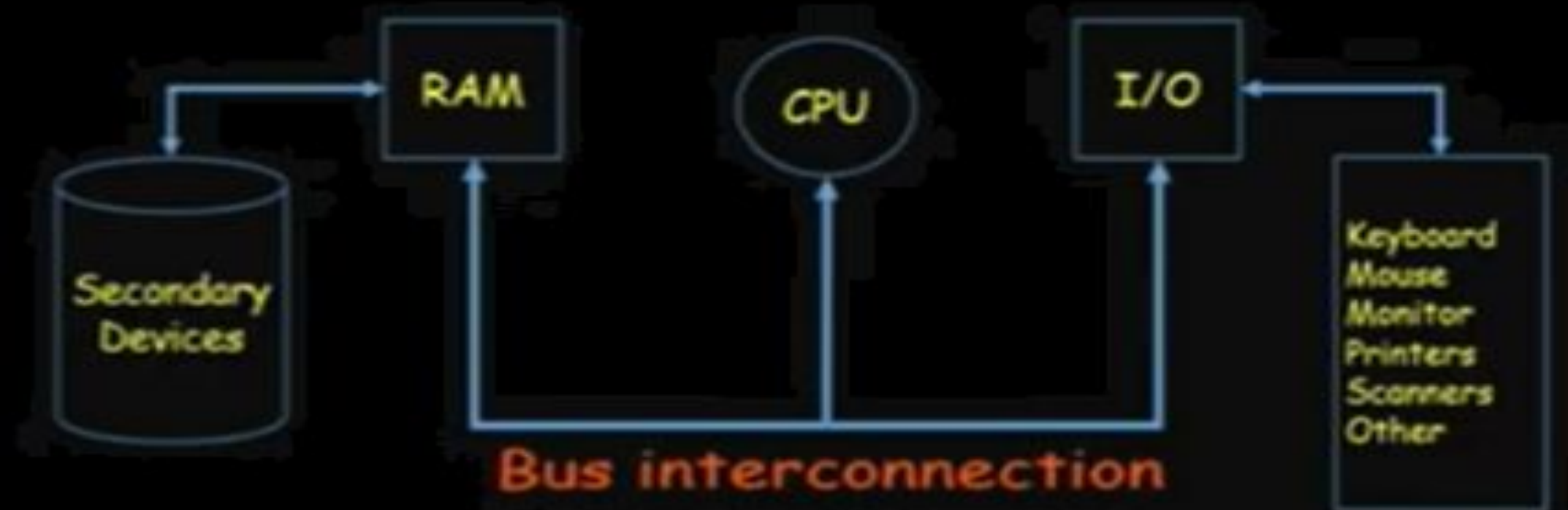
1. Differentiate between workstation and server.
2. List few characteristics of mini computers.
3. Write a short on hybrid computers.
4. Differentiate between mainframe and minicomputer.
5. Differentiate between supercomputer and minicomputer.
6. Explain Basic organization of computer with suitable diagram?

System Bus and Interconnection

Bus interconnection

A computer system consists of different devices such as CPU, Main memory and I/O devices. These devices are connected to an internal communication channel of the computer system to transfer data between these devices.

The internal communication channel of the computer system is called bus interconnection.



Computer BUS:

A bus consists of a set of parallel lines. It is used to transfer data between different components of the computer.



1. One line of bus can transfer one bit at a time.
2. The capacity of computer bus depends on the number of data lines in it.
3. A bus with 16 lines can carry 16 bits and with 32 lines can carry 32 bits.
4. The amount of data that a bus can carry at one time is called bus width.

Types of Buses

Types of Buses:

1. System Buses

2. Expansion Buses

System Buses:

System bus is used to connect the main components of a computer such as CPU and main memory. System buses are part of motherboard. Computers normally have system bus of 70-100 lines.

Three types of system buses are as follows:

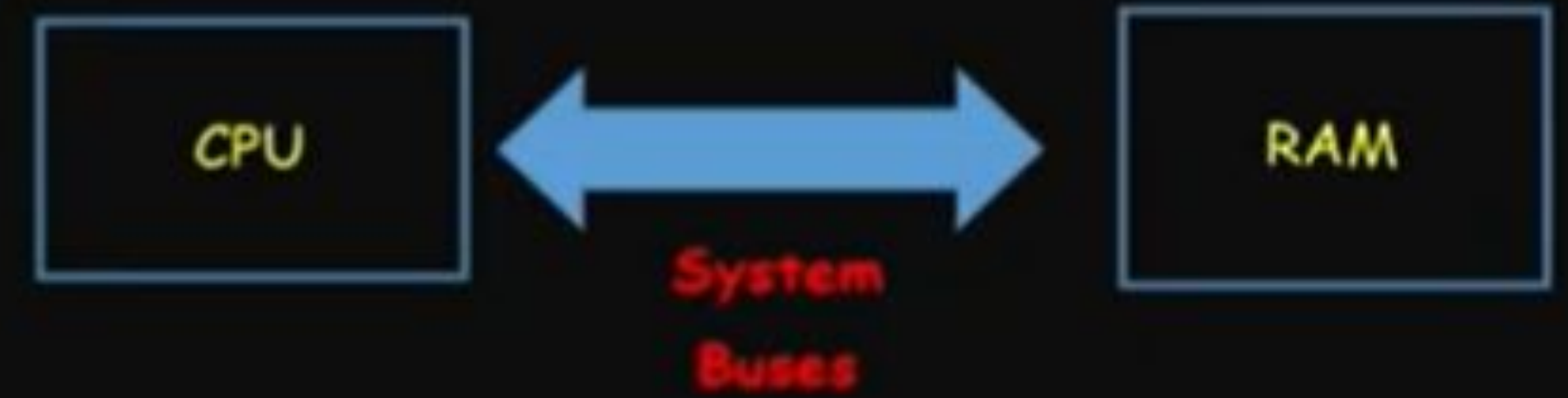
1. Data Buses

2. Address Buses

3. Control Bus

Expansion Buses:

Expansion bus is used to connect CPU with peripheral devices such as mouse, keyboard, modem and scanner etc.



System Buses

System Buses And It's Types

System Bus:

System bus is used to connect the main components of a computer such as CPU and main memory. System buses are part of motherboard.



Three types of system buses are as follows:

1. Data Bus

2. Address Bus

3. Control Bus

Data Bus

Data bus is the most common type of bus. It is used to transfer data between different components of computer.

1. The number of lines in data bus effects the speed of data transfer between different components.
2. A 64 bit line data bus can transfer 64 bits of data at one time.
3. CPU can read data from memory using these line.
4. CPU can write data to memory locations using these lines.



Cont..

System Buses And It's Types

Address Bus:

Many components are connected to one another through buses. Each component is assigned a Unique ID. This ID is called the address of that component.

1. It is uni-directional.
2. It can carry information only in one direction.
3. It carries address of memory location from microprocessor to the main memory.



Control Bus:

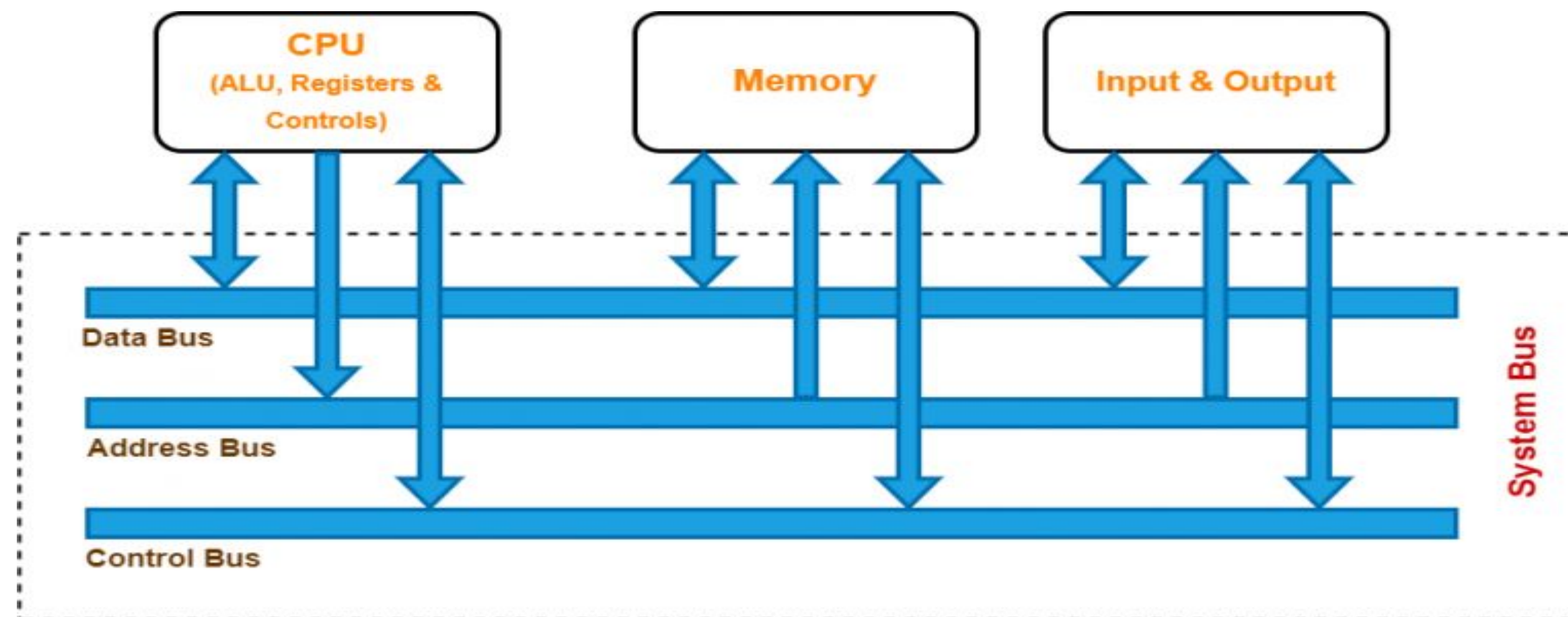
Control Bus is used to transmit different commands or control signals from one component to another component.

1. It specifies the time for which a device can use data and address bus.
2. It specifies the type of operation to be performed.



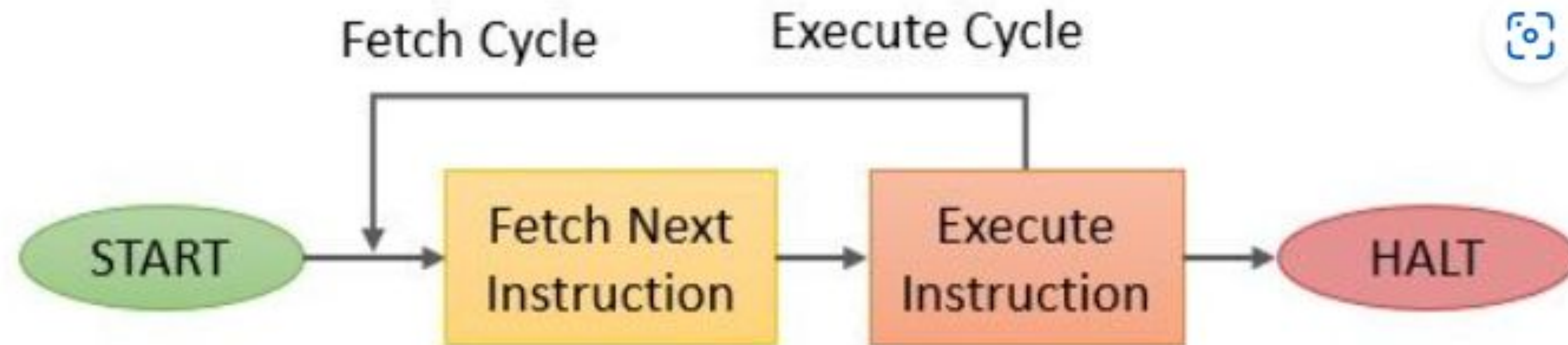
Cont..

- ❑ The operation of the bus is as follows. If one module wishes to send data to another, it must do two things: (1) obtain the use of the bus, and (2) transfer data via the bus. If one module wishes to request data from another module, it must (1) obtain the use of the bus, and (2) transfer a request to the other module over the appropriate control and address lines. It must then wait for that second module to send the data.
- ❑ System bus consists, typically, of from about fifty to hundreds of separate lines. Each line is assigned a particular meaning or function. Although there are many different bus designs, on any bus the lines can be classified into three functional groups in below figure : data, address, and control lines. In addition, there may be power distribution lines that supply power to the attached modules



Instruction Cycle

- ❑ It is defined as the basic cycle carried out in a computer system in which the computer system fetched the instruction from memory, decodes the instruction and then executes the instruction. It is also known as **Fetch-Execute-Cycle**.
- ❑ The figure below shows you the processing of the basic instruction cycle.



- ❑ CPU perform fetch, decode and execute cycle to execute one program instruction. The Machine cycle is the part of instruction cycle.

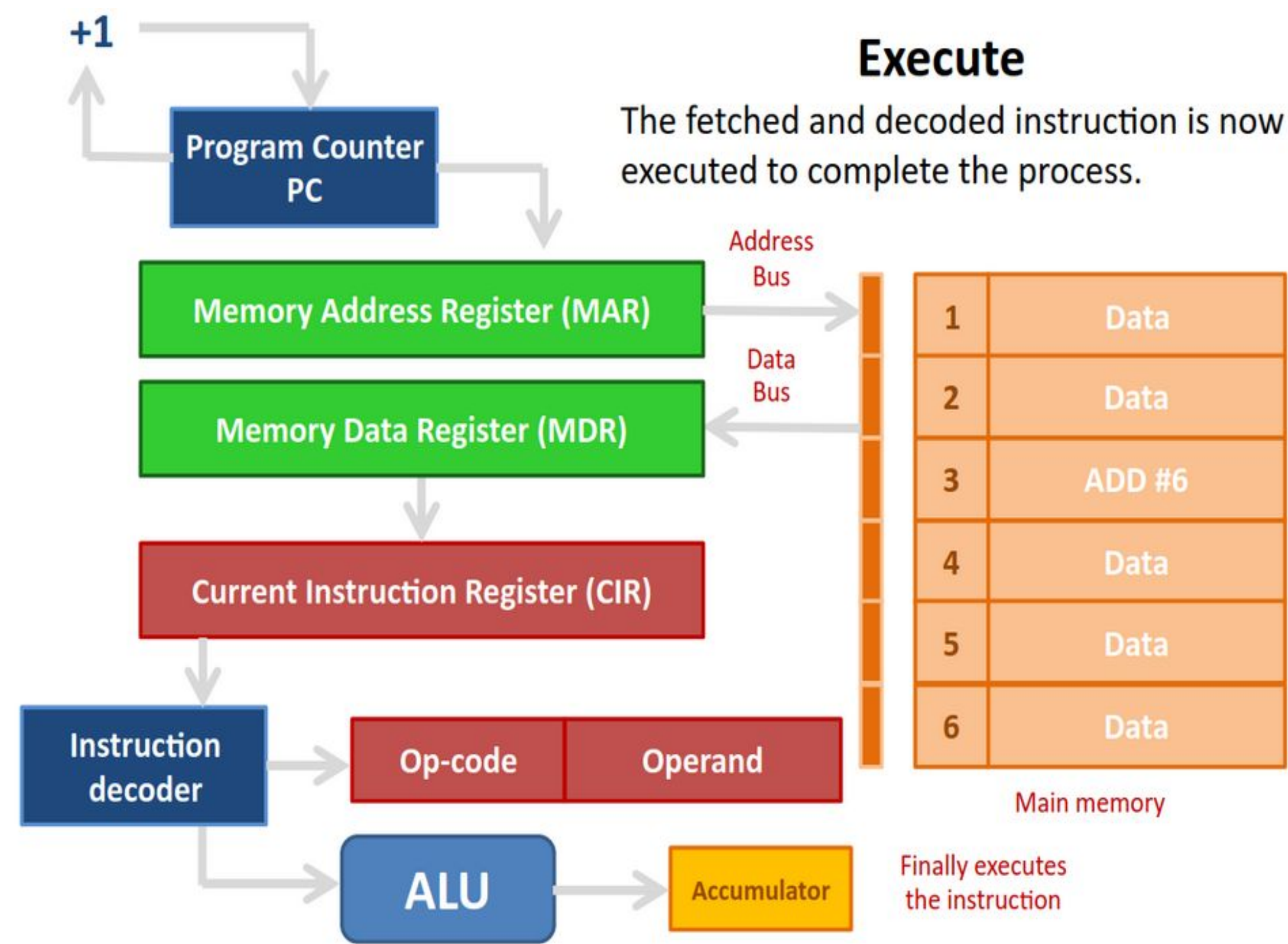
Cont..

- ❑ In the computer system, all the instructions are executed in the RAM of the computer system. The CPU is responsible for executing the instruction.
- ❑ The CPU system first fetches the data and instruction from the main memory and store in the temporary memory, which is known as registers. This phase is known as the fetch cycle.
- ❑ After fetching an instruction from memory, the next step taken by the CPU is decoding of fetched instruction. This phase is known as the decode phase.
- ❑ The CPU contains an instructions set which contain all the predefined list of instructions. In the last phase of the instruction cycle, data processing takes place. The instruction executes in this phase, and the result is stored in another register.
- ❑ After the fetch-decode-execute cycle, the CPU reset itself for another instruction cycle. The CPU system is considered the basic operation cycle, which is carried out in RAM of the central processing unit and executes the instruction.

Fetch Cycle

During the fetch execute cycle, the computer retrieves a program instruction from its memory. It then establishes and carries out the actions that are required for that instruction.

Step of Fetch Cycle



| Step | Fetch execute cycle steps | Simplified description |
|------|---|---|
| 1 | The PC contains the address of the memory location that has the next instruction which has to be fetched | PC has address of next instruction |
| 2 | This address is then copied from the PC to the MAR via the address bus | PC copied to the MAR |
| 3 | The contents (instruction) at the memory location (address) contained in MAR are then copied into the MDR | Lookup MAR and get contents. Copy contents into the MDR |
| 4 | The contents (instruction) in the MDR is then copied and placed into the CIR | Copy MDR contents into the CIR |
| 5 | The value in the PC is then incremented by 1 so that it now points to the next instruction which has to be fetched | PC is then incremented by 1 |
| 6 | The instruction is finally decoded and then executed by sending out signals (via control bus) to the various components of the computer | The instruction is decoded and then executed |
| 7 | Repeat | |

Decode and Execute Cycle

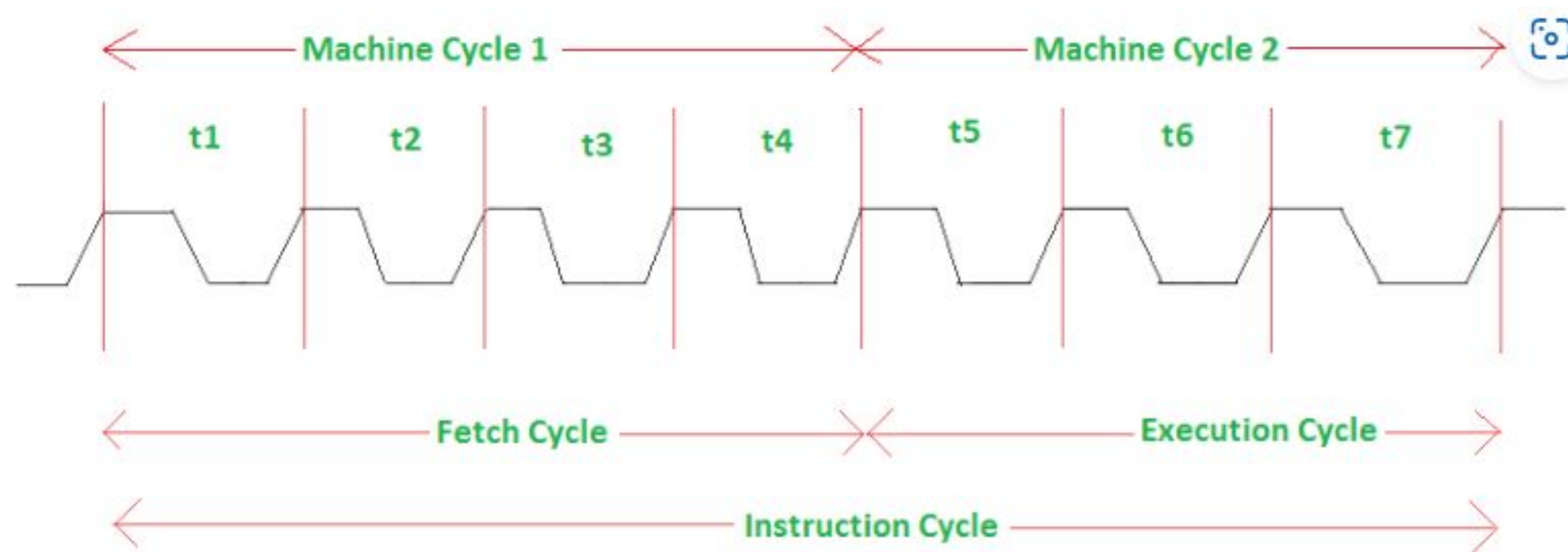
- ❑ During the decode cycle The instruction in MDR is decoded by the CU. CU prepared for next step by loading values into MAR or MDR.
- ❑ The contents of the CIR are divided. Part of the instruction might be an operation (like ADD) and part of the instruction might be data, or in our case, an address where data can be found, like 75567. The ADD part is known as the OPERATOR and the data part is known as the OPERAND.

❑ EXECUTE

The instruction can now be executed. Arithmetic and logical instructions are carried out using the Accumulator(s) in a CPU.

Cont..

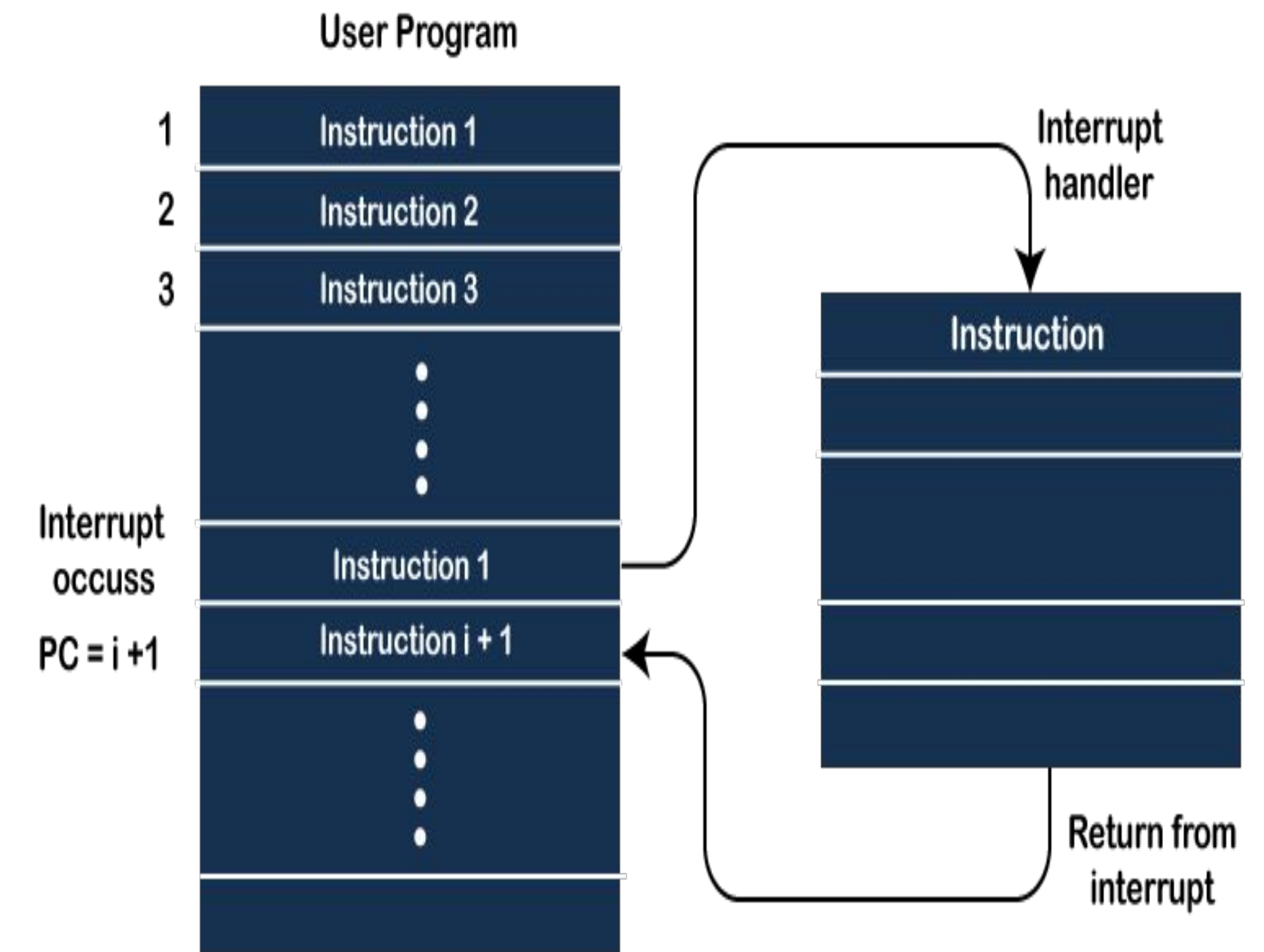
- ❑ Time required to execute and fetch an entire instruction is called ***instruction cycle***.
- ❑ The time required by the microprocessor to complete an operation of accessing memory or input/output devices is called ***machine cycle***.
- ❑ One time period of frequency of microprocessor is called *t-state*. A t-state is measured from the falling edge of one clock pulse to the falling edge of the next clock pulse.
- ❑ Fetch cycle takes four t-states and execution cycle takes three t-states



What is Interrupt

❑ **Definition:** Transfer of program control from a currently running program to hardware and software-generated signal request for the processor, which creates a disturbance to a running program. This disturbance is called an Interrupt.

- Interrupts is an signal. In this scheme the processor issue a command to an I/O devices for input and output operation. The device generate an interrupt signal to the processor when it ready.
- The CPU handles all the processing when it received an interrupts. And perform I/O operation. CPU handle interrupt very carefully, and priority is given to all the interrupts to execute every instruction.
- This situation arises due to some error conditions, user requirements, software and hardware requirement.

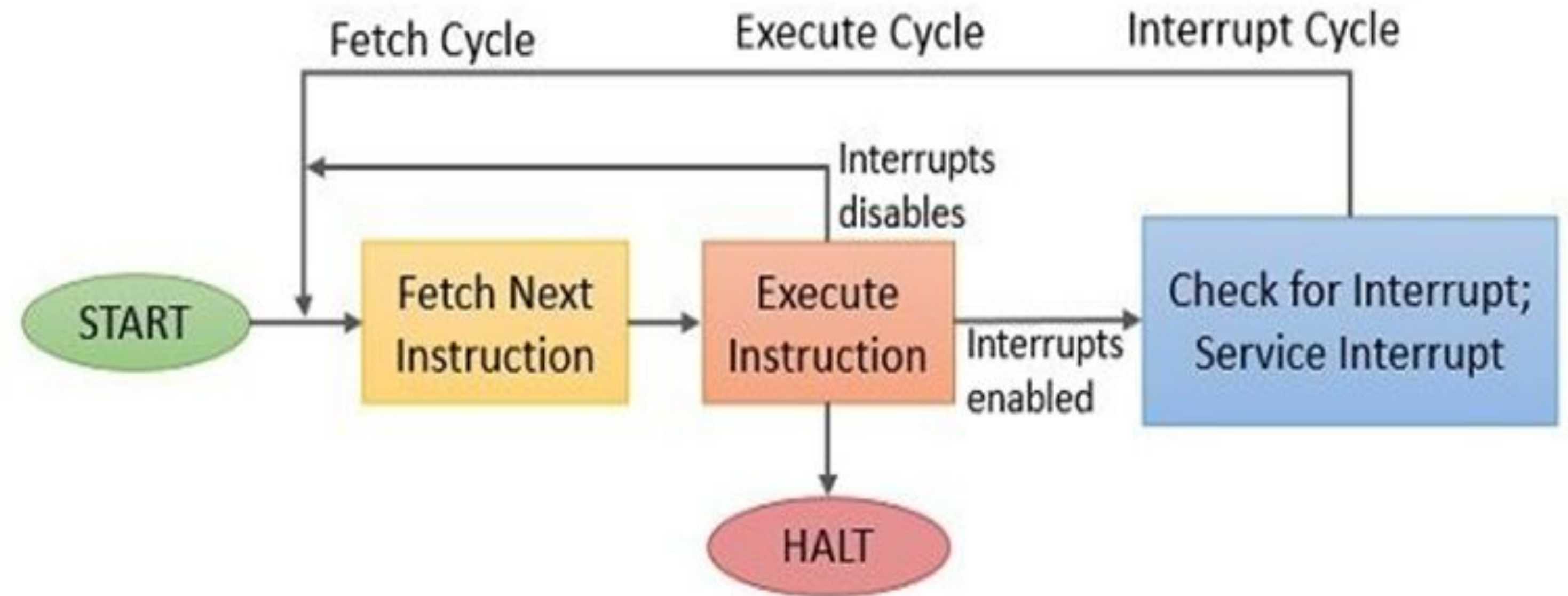


Types of Interrupts

- ❑ The interrupts can be various type but they are basically classified into hardware interrupts and software interrupts.
- ❑ **Hardware Interrupts:** If a processor receives the interrupt request from an external I/O device it is termed as a hardware interrupt.
- ❑ **Software Interrupts:** The software interrupts are the interrupts that occur when a condition is met or a system call occurs.
- ❑ Common classes of interrupts are

| | |
|-------------------------|---|
| Program | Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space. |
| Timer | Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis. |
| I/O | Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions. |
| Hardware failure | Generated by a failure, such as power failure or memory parity error. |

Interrupt Cycle

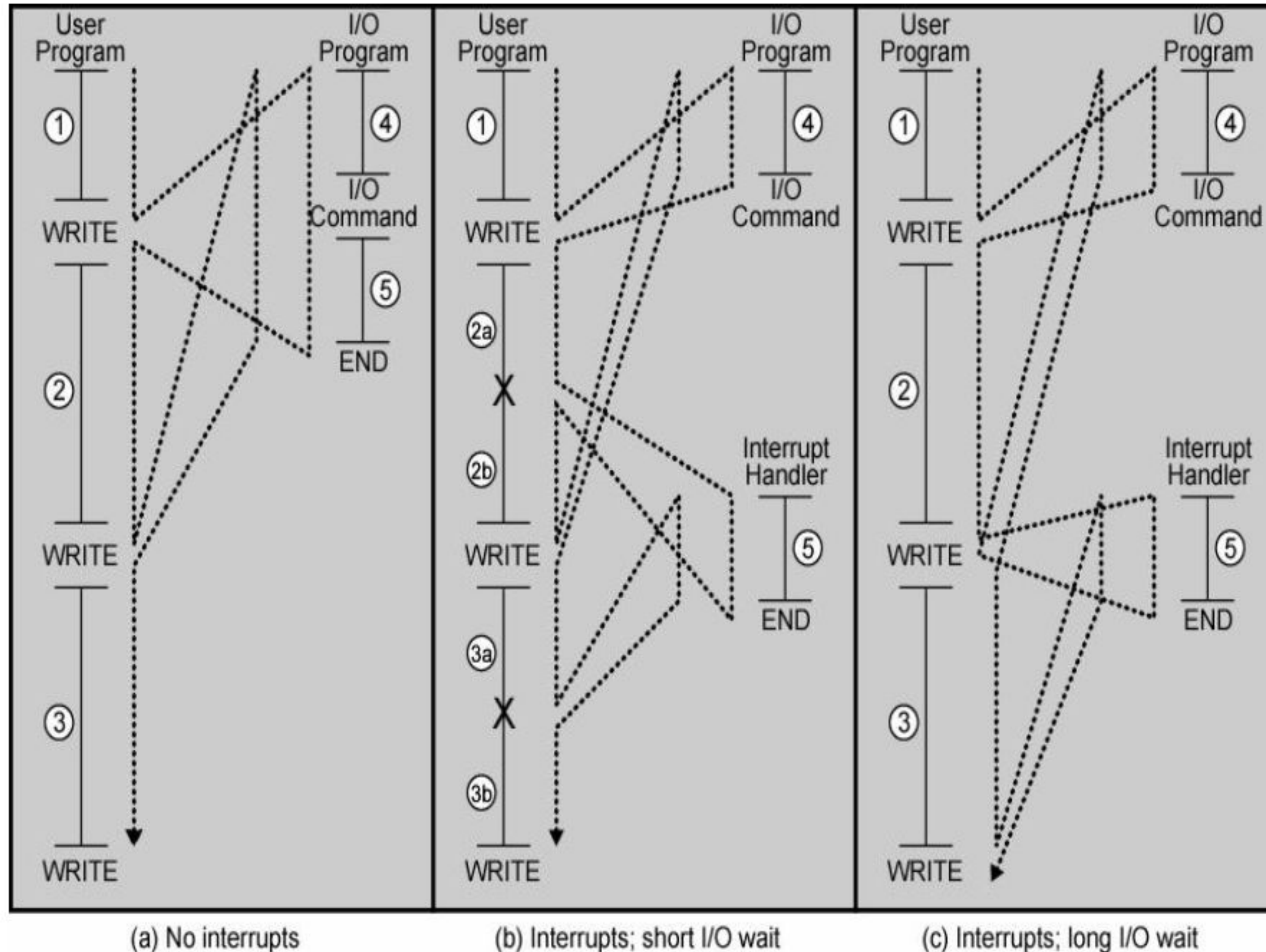


- ❑ In the interrupt cycle, the processor checks to see if any interrupts have occurred, indicated by the presence of an interrupt signal. If no interrupts are pending, the processor proceeds to the fetch cycle and fetches the next instruction of the current program. If an interrupt is pending, the processor does the following:
- ❑ It suspends execution of the current program being executed and saves its context. This means saving the address of the next instruction to be executed (current contents of the program counter) and any other data relevant to the processor's current activity.
- ❑ It sets the program counter to the starting address of an interrupt handler routine.

Cont..

- ❑ Consider the condition that the interrupts are **enabled**. In this case, **if an interrupt occurs then the processor halt the execution of the current program**. Thereby it saves the address of the instruction that has to be executed next and service the occurred interrupt.
- ❑ To process the interrupts the processor set the program counter with starting address of the interrupt service routine. This would let the processor fetch the first instruction of interrupt service routine and service the occurred interrupt. Once the interrupt is serviced the processor resumes back to the execution of the program it has halted to service the interrupt. It set the program counter with the address of the next instruction to be executed.
- ❑ If the interrupts are **disabled** then the processor will simply ignore the occurrence of interrupts. **The processor will smoothly execute the currently running program and will check the pending interrupts once the interrupts are enabled.**

Program Flow control with and without Interrupt



- The user program (depicted in figure 5a) performs a series of WRITE calls interleaved with processing.
- Code segments 1, 2, and 3 refer to sequences of instructions that do not involve I/O. The WRITE calls arc to an I/O program that is a System utility and that will perform the actual I/O operation.
- The I/O program consists of three sections:
 - A sequence of instructions, labeled 4 in the figure, to prepare for the actual I/O operation. This may include copying the data to be output into a special buffer and preparing the parameters for a device command.
 - The actual I/O command. Without the use of interrupts, once this command is issued, the program must wait for the I/O device to perform the requested function (or periodically poll the device). The program might wait by simply repeatedly performing a test operation to determine if the I/O operation is done.
 - A sequence of instructions, labeled 5 in the figure, to complete the operation. This may include setting a flag indicating the success or failure of the operation.

Multiple Interrupt

- ❑ An interrupt may occur during the processing of another interrupt. This may lead to nested interrupt handling. This is permitted in many situations.
- ❑ But it is also possible to control the response to interrupts by masking certain interrupts and by disabling interrupts.
- ❑ Interrupts are useful not only in IO processing but also in multiprogramming.
- ❑ **Two approaches can be taken to dealing with multiple interrupts.**
- ❑ The first **is to disable interrupts while an interrupt is being processed**. A disabled interrupt simply means that the processor can and will ignore that interrupt request signal.
- ❑ A second approach **is to define priorities for interrupts and to allow an interrupt of higher priority to cause a lower-priority interrupt handler to be itself interrupted**

Sequential interrupt processing

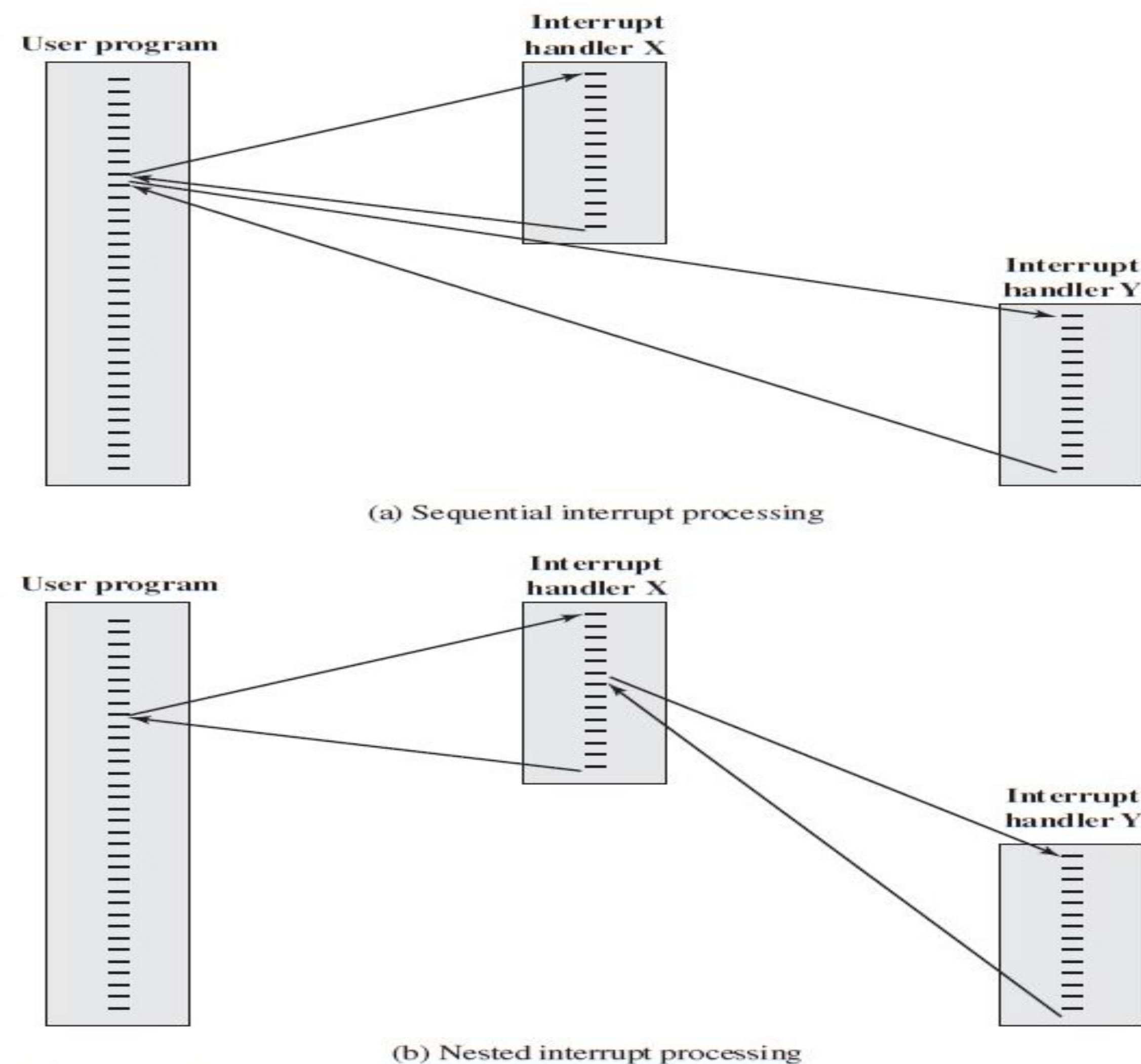


Figure 3.13 Transfer of Control with Multiple Interrupts

Time Sequence of Nested Interrupts

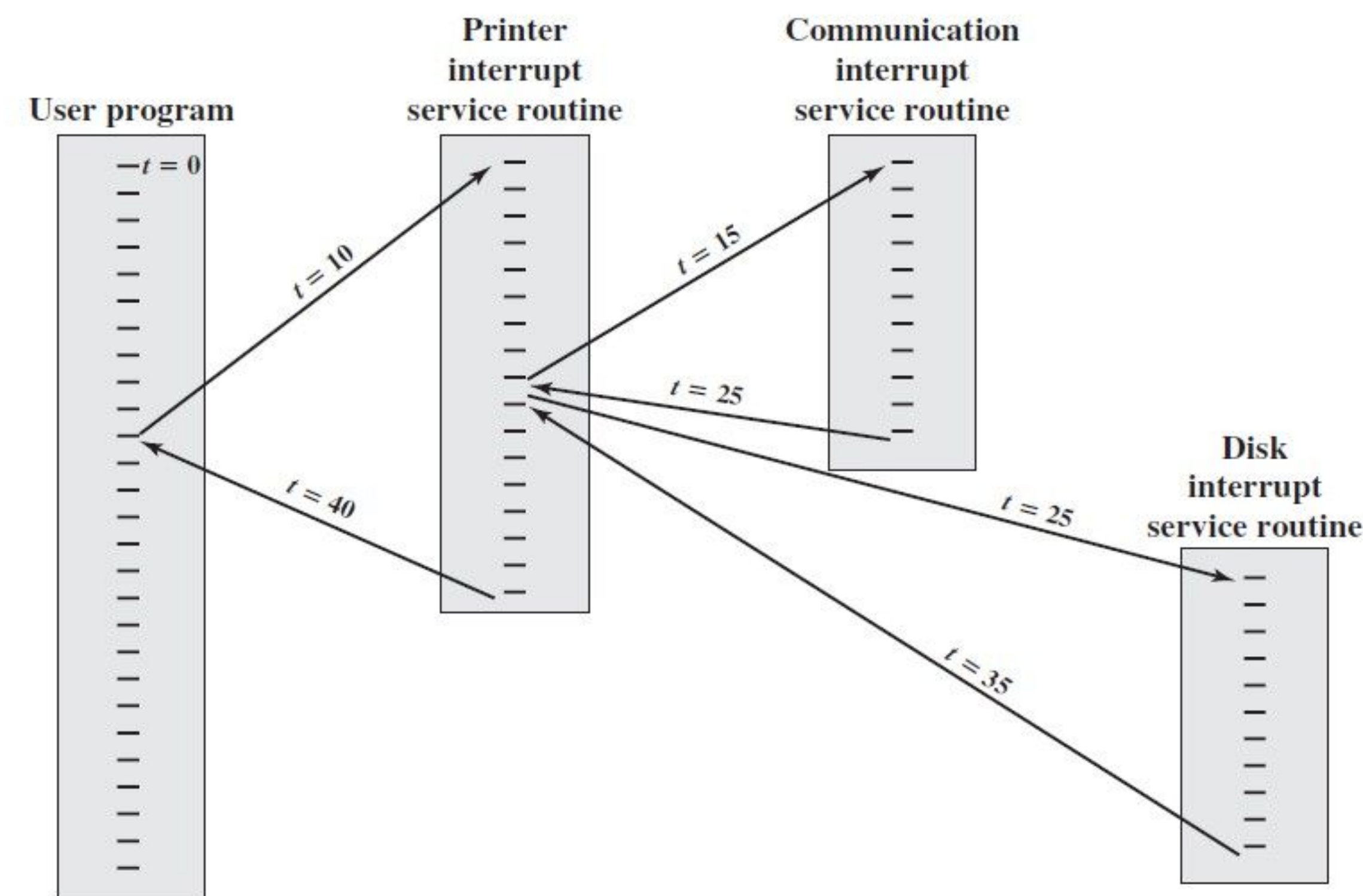


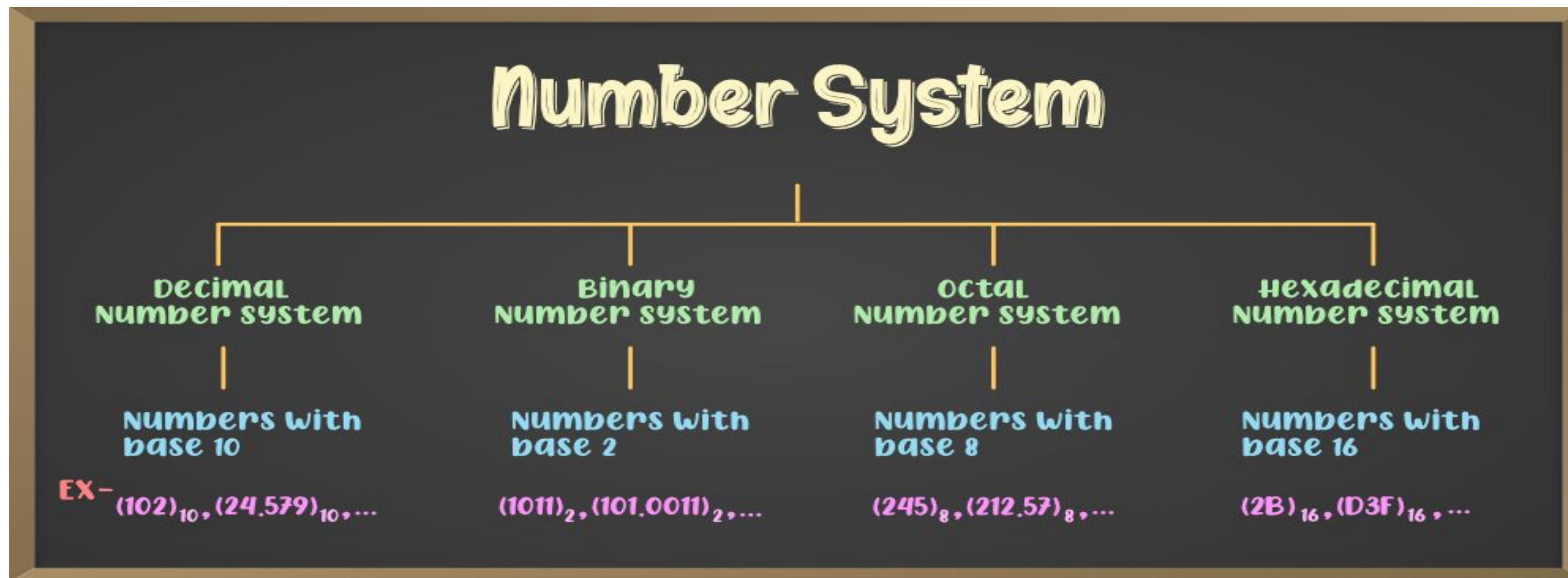
Figure 3.14 Example Time Sequence of Multiple Interrupts

Questions

1. Define instruction cycle.
2. With a neat schematic, explain the steps involved in fetch and decode phases using register transfer instructions.
3. Write about Interrupt and its types?
4. Illustrate the phases of Interrupt Cycle with a neat flowchart?
5. Define Interrupt and classify types of Interrupts?
6. Explain instruction cycle with example?
7. Explain Fetch-Execute cycle with diagram?
8. Explain system bus with suitable example?

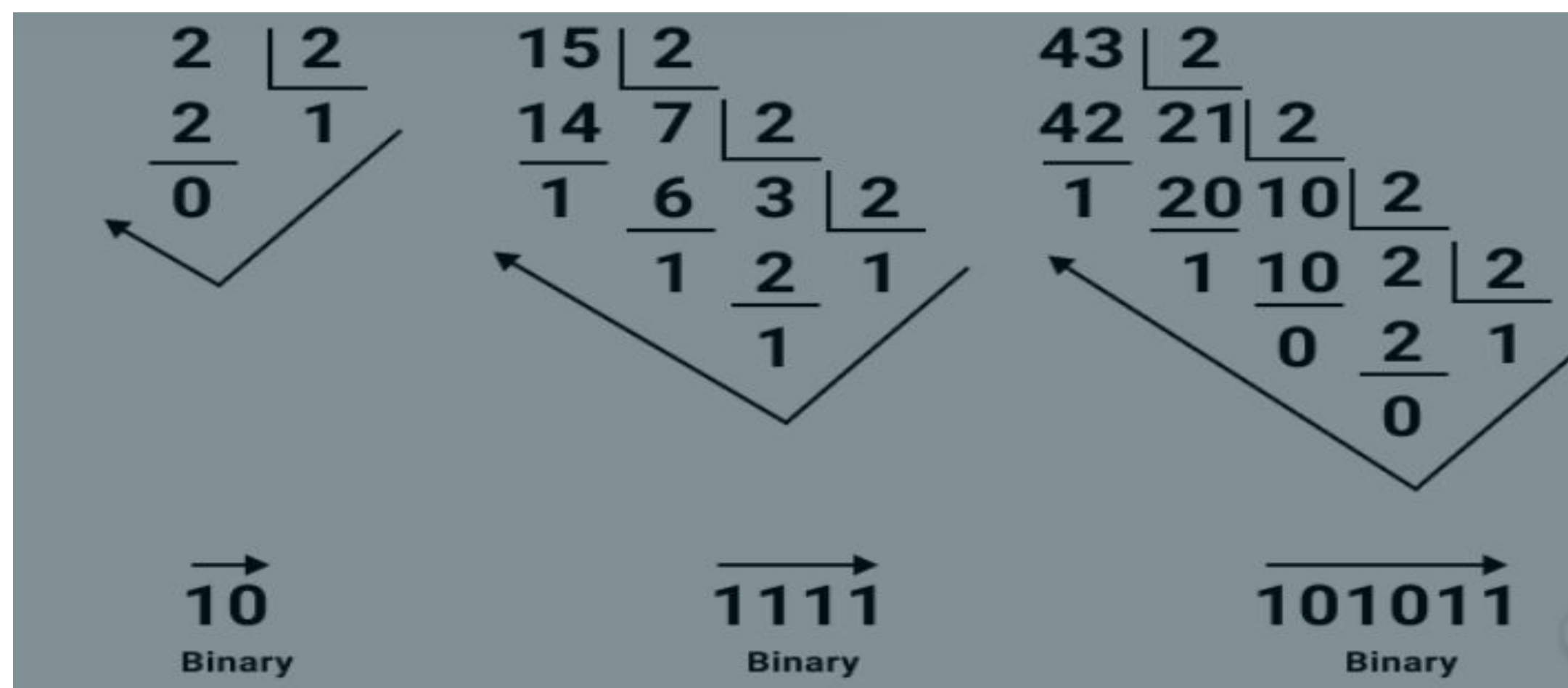
Number System

- ❑ Number system defines a set of values used to represent quantity.
 - Quantity represent using numbers like 1,2,3
- ❑ It is also called the system of numeration and it defines a **set** of values to represent a quantity.
- ❑ There are four number systems that a computer supports. They are



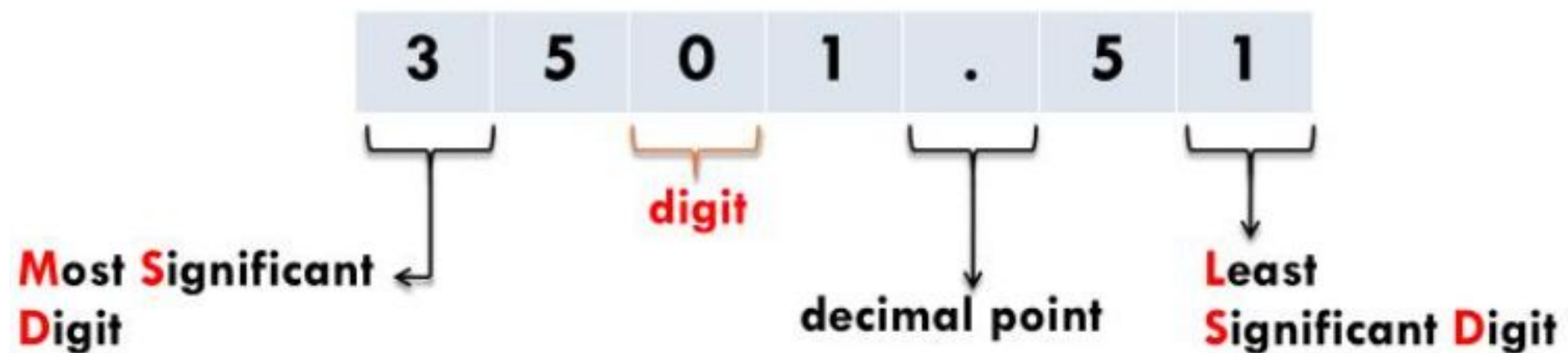
Radix of the Number System

- In number system modern method of representing numbers symbolically is based on positional notations. The total number of different symbols which are used in a particular number system is called the base or radix of the system and the weight of each position of a particular number is expressed as a power of the base. For example $(135)_{10}$, $(1001010)_2$, $(207)_8$ and $(9FF)_{16}$
- For example, in the base 10 number system, there are a total of 10 digits used (zero through nine). Therefore, its radix is 10. Another way of putting this could be: the single number obtained from the date of birth is called a radix. For example, if your birthday is on the 14th, then $1+4 = 5$.



Decimal number system

- ❑ The decimal number system is composed using ten digits: 0,1,2,3,4,5,6,7,8 and 9 with the base number as 10.
- ❑ The decimal number system is the system that we generally use to represent numbers in real life. If any number is represented without a base, it means that its base is 10.
- ❑ **Example: 3501.51**



- ❑ With each movement from right to left, the number value increases by 10. The position on the left of the decimal is for tens, hundreds, thousands, and so on units.

Cont..

- ❑ **Example** – The decimal number 4567 consists of 7 at the unit position, 6 in the tens position, 2 in the hundreds position, and 1 in the thousands position.
- ❑ $4567 = (4 \times 1000) + (5 \times 100) + (6 \times 10) + (7 \times 1)$
 $= 4000 + 500 + 60 + 7$
 $= 4567$
- ❑ The same principle holds for decimal fractions, but negative powers of 10 are used. Thus, the decimal fraction 0.256 stands for 2 tenths plus 5 hundredths plus 6 thousandths:

$$\underline{0.256} = (2 \times 10^{-1}) + (5 \times 10^{-2}) + (6 \times 10^{-3})$$
$$= 0.256$$

In general, for the decimal representation of $X = \{ \dots d_2 d_1 d_0 . d_{-1} d_{-2} d_{-3} \dots \}$, the value of X is

$$X = \sum_i (d_i \times 10^i)$$

- ❑ **Examples to Solve or Verify** 3501, 10285, 83, $(15.25)_{10}$

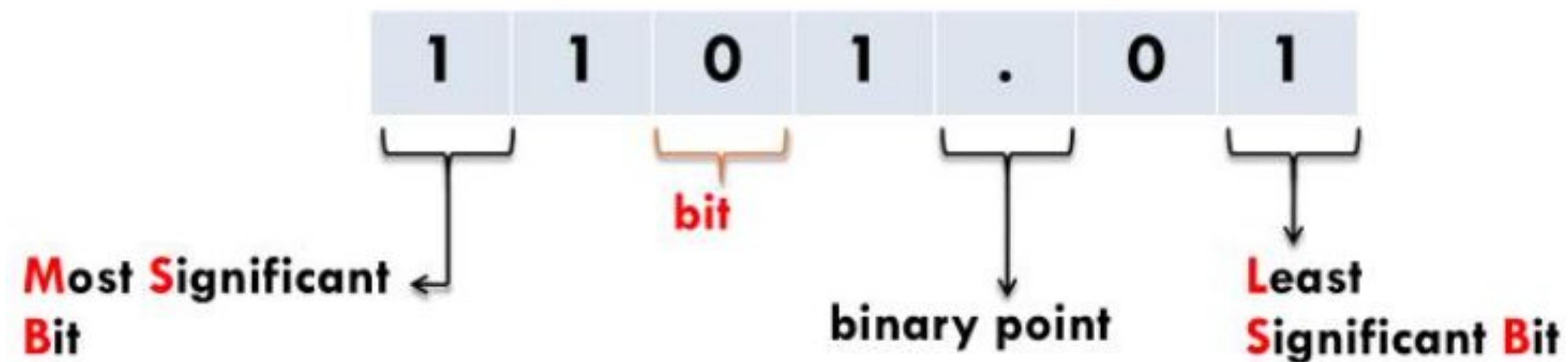
❑ 3 5 0 1 (base-10)

| | | |
|---|-----------------|------|
| 1 | $\times 10^0 =$ | 1 |
| 0 | $\times 10^1 =$ | 0 |
| 5 | $\times 10^2 =$ | 500 |
| 3 | $\times 10^3 =$ | 3000 |

$3000 + 500 + 0 + 1 = 3501$

Binary Number System

- ❑ The binary number system uses only two digits: 0 and 1. The numbers in this system have a base of 2. Digits 0 and 1 are called bits and 8 bits together make a byte.
- ❑ The data in computers is stored in terms of bits and bytes. The binary number system does not deal with other numbers such as 2,3,4,5 and so on.
- ❑ **Example: 1101.01**



- ❑ **Number representation in binary format**

1. 14 can be written as 1110
2. 19 can be written as 10011
3. 50 can be written as 110010

Cont

- The digits 1 and 0 in binary notation have the same meaning as in decimal notation

$$0_2 = 0_{10}$$

$$1_2 = 1_{10}$$

- To represent larger numbers, as with decimal notation, each digit in a binary number has a value depending on its position

$$10_2 = (1 \times 2^1) + (0 \times 2^0) = 2_{10}$$

$$11_2 = (1 \times 2^1) + (1 \times 2^0) = 3_{10}$$

$$100_2 = (1 \times 2^2) + (0 \times 2^1) + (0 \times 2^0) = 4_{10}$$

- and so on. Again, fractional values are represented with negative powers of the radix:

$$1001.101 = 2^3 + 2^0 + 2^{-1} + 2^{-3} = 9.625_{10}$$

In general, for the binary representation of $Y = \{\dots b_2 b_1 b_0 . b_{-1} b_{-2} b_{-3} \dots\}$, the value of Y is

$$Y = \sum_i (b_i \times 2^i) \quad (9.3)$$

Octal Number System

- ❑ The octal number system uses eight digits: 0,1,2,3,4,5,6 and 7 with the base of 8.
- ❑ **Octal Number System** has a base of eight and uses the numbers from 0 to 7. The octal numbers, in the number system, are usually represented by binary numbers when they are grouped in pairs of three.
- ❑ Example:

| | |
|------------------------|--|
| Binary Number: | 101010011.110100 |
| Group of three digits: | <u>101</u> <u>010</u> <u>011</u> . <u>110</u> <u>100</u> |
| | ↓ ↓ ↓ ↓ ↓ |
| Octal Equivalent: | 5 2 3 6 4 |
| | = (523.64) ₈ |

- ❑ octal number 128 is expressed as 0010102 in the binary system, where 1 is equivalent to 001 and 2 is equivalent to 010. And Decimal (135)₁₀ can be written as (207)₈

Hexadecimal Number System

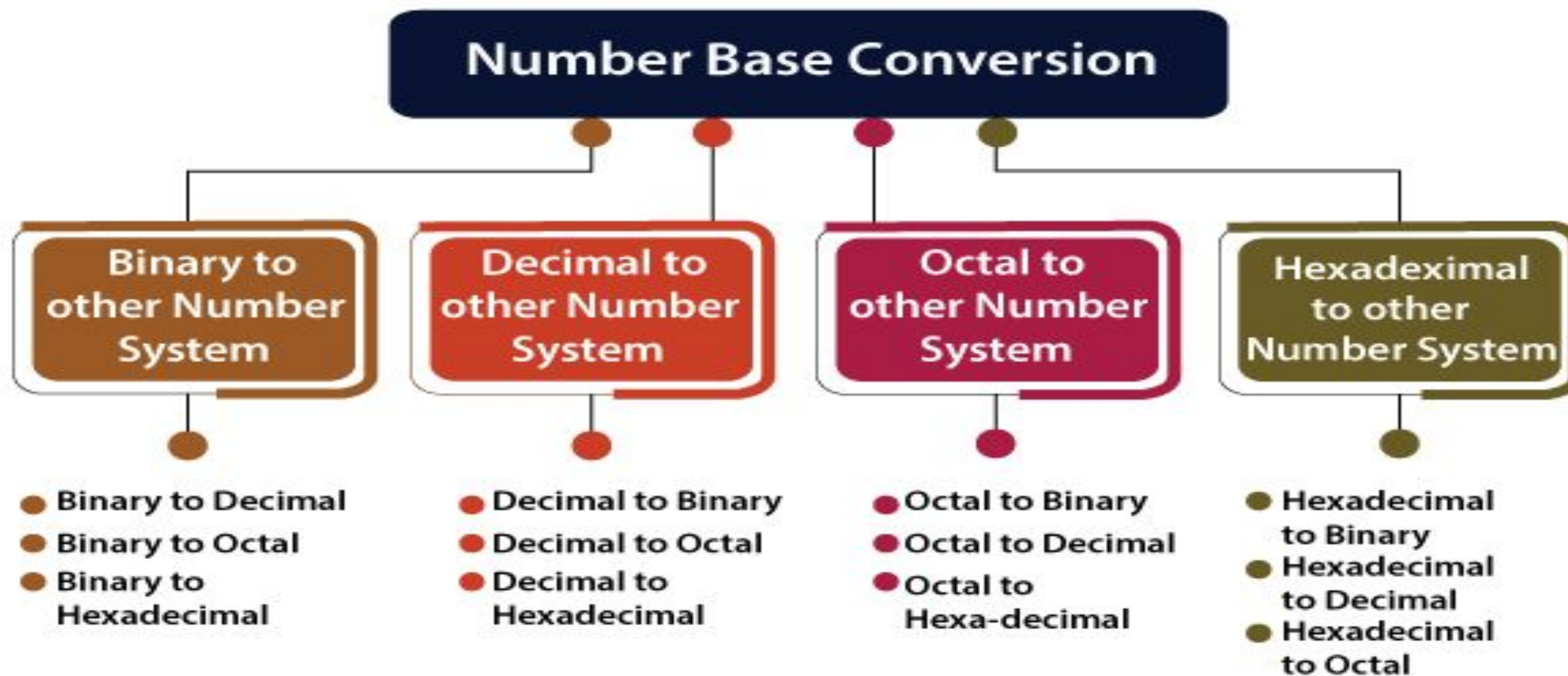
- ❑ The hexadecimal number system uses sixteen digits/alphabets: 0,1,2,3,4,5,6,7,8,9 and A,B,C,D,E,F with the base number as 16.
- ❑ Here, A-F of the hexadecimal system means the numbers 10-15 of the decimal number system respectively. This system is used in computers to reduce the large-sized strings of the binary system.
- ❑ Binary digits are grouped into sets of four bits, called a **nibble**. Each possible combination of four binary digits is given a symbol, as follows:

| | | | |
|----------|----------|----------|----------|
| 0000 = 0 | 0100 = 4 | 1000 = 8 | 1100 = C |
| 0001 = 1 | 0101 = 5 | 1001 = 9 | 1101 = D |
| 0010 = 2 | 0110 = 6 | 1010 = A | 1110 = E |
| 0011 = 3 | 0111 = 7 | 1011 = B | 1111 = F |

- ❑ Binary number $(11111111)_2$ can be written as $(FF)_{16}$ Decimal number $(255)_{10}$ can be written as $(FF)_{16}$

Number Base Conversion

- We have four types of number systems so each one can be converted into the remaining three systems. There are the following conversions possible in Number System



Decimal to binary conversion

- Decimal numbers are represented in base 10, but the binary numbers are of base 2. Hence, to convert a decimal number to binary number, the base of that number is to be changed. Follow the steps given below:

Step 1: Divide the Decimal Number with the base of the number system to be converted to. Here the conversion is to binary, hence the divisor will be 2. **Note :** Check if the given decimal number is less than 2. If it is less than 2 then the given Decimal No is the same when converted to its binary equivalent. **If the given decimal number is greater than 2, then divide the given number by 2.**

Step 2: The remainder obtained from the division will become the least significant digit of the new number.

Step 3: The quotient obtained from the division will become the next dividend and will be divided by base 2.

Step 4: The remainder obtained will become the second least significant digit i.e. it will be added in the left of the previously obtained digit.

- Now, the steps 3 and 4 are repeated until the quotient obtained becomes 0, and the remainders obtained after each iteration are added to the left of the existing digits.

Example

- Example 1 convert 13 to binary

$(13)_{10} \rightarrow (\quad)_2$

| Divisor | Dividend | rem |
|---------|----------|-----|
| 2 | 13 | 1 |
| 2 | 6 | 0 |
| 2 | 3 | 1 |
| 2 | 1 | 1 |
| 2 | 0 | 0 |
| 2 | 0 | 0 |

1 1 0 1

- Example 2 convert 27 to binary

$(27)_{10} = (11011)_2$

| 2 | 27 | Remainder |
|---|----|-----------|
| 2 | 13 | 1 |
| 2 | 6 | 1 |
| 2 | 3 | 0 |
| 2 | 1 | 1 |
| | 0 | 1 |

Fraction Cont..

❑ Convert Fraction From Decimal to Binary

- ❑ There is a simple method to find the equivalent binary representation of a decimal fractional number. The steps to convert Decimal Fraction to Binary numbers are given number:

1st Step: Multiply the given decimal fraction number by 2.

2nd Step: Note down the Integer Part and multiply the fraction part.

3rd Step: Repeat Step 2 until the fractional part is 0 and note down all the integer parts.

4th Step: Now write the numbers from bottom to top

❑ Question: Convert $(0.59375)_{10}$ to Binary

Answer: The Binary Representation of $(0.59375)_{10}$ is $(0.10011)_2$

Example

- Example: Convert decimal number $(96.46)_{10}$ to Binary number
 - First calculate binary format of 98 and second calculate binary format of .46
 - Then merge both answers to get binary equivalent of 96.46

Binary equivalent of 98 is 1100010_2

Fractional part:

| |
|----------------------------|
| $0.46 \times 2 = 0.92 = 0$ |
| $0.92 \times 2 = 1.84 = 1$ |
| $0.84 \times 2 = 1.68 = 1$ |
| $0.68 \times 2 = 1.36 = 1$ |
| $0.36 \times 2 = 0.72 = 0$ |



| | | |
|---|----|-----|
| 2 | 98 | |
| 2 | 49 | - 0 |
| 2 | 24 | - 1 |
| 2 | 12 | - 0 |
| 2 | 6 | - 0 |
| 2 | 3 | - 0 |
| 1 | | - 1 |

$1100010.01110 \dots_2$

- Convert the following examples into binary format

1. 28.125 2. 156 3. 128 4. 32 5. 48 6. 27.156 7. 29.30

Decimal to Octal Conversion

- ❑ Octal Numbers are represented in base 8. Hence, to convert a decimal number to octal number, the base of that number is to be changed. Follow the steps given below:

Step 1: Divide the Decimal Number with the base of the number system to be converted to. Here the conversion is to octal, hence the divisor will be 8.

Step 2: The remainder obtained from the division will become the least significant digit of the new number.

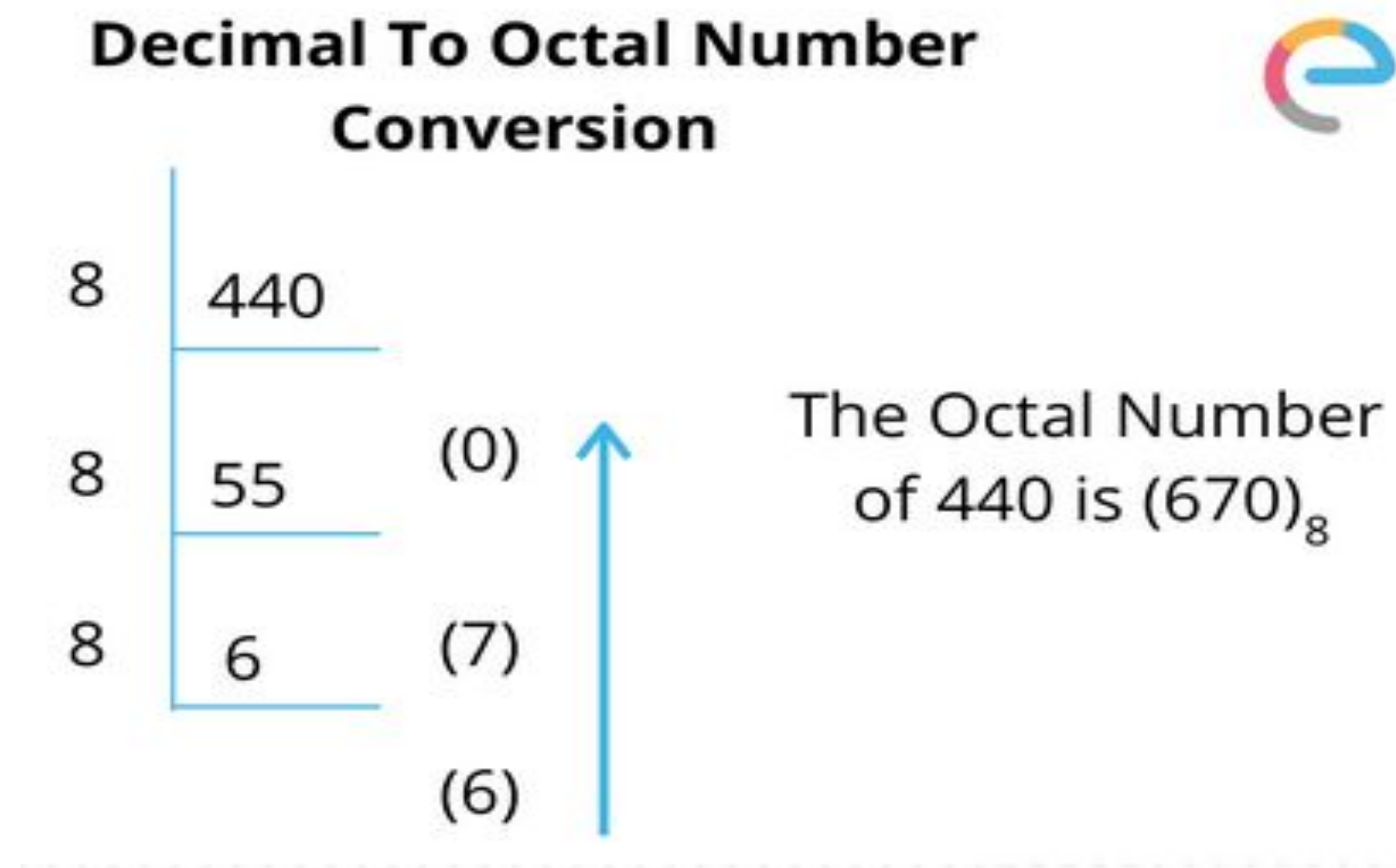
Step 3: The quotient obtained from the division will become the next dividend and will be divided by base i.e. 8.

Step 4: The remainder obtained will become the second least significant digit i.e. it will be added in the left of the previously obtained digit.

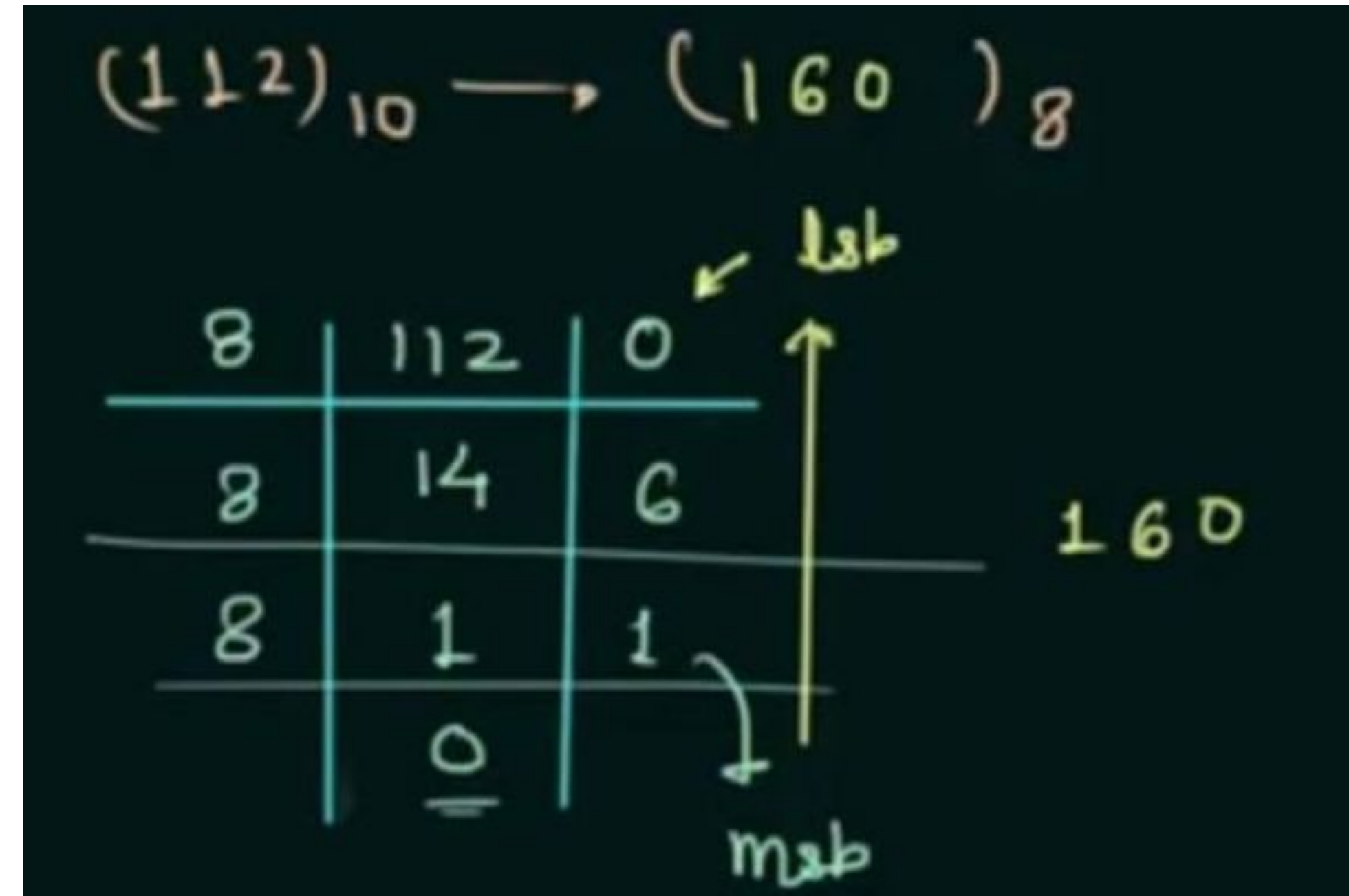
- ❑ Now, the steps 3 and 4 are repeated until the quotient obtained becomes 0, and the remainders obtained after each iteration are added to the left of the existing digits.

Example

□ Example convert $(440)_{10}$ to Octal number



Example convert $(112)_{10}$ to Octal number




□ Convert 256, 100, 2980, 158, 112, 334, 394 to Octal Number

Fraction Cont..

□ **Example** : Conversion of $(.45)_{10}$ to equivalent octal number.

□ We will multiply the given number by the base of octal number and then write the integer part as the given direction by arrow.

| Multiplication | Result | Integer Portion | Fraction Portion |
|-----------------|--------|-----------------|------------------|
| 0.45×8 | 3.60 | 3 | .60 |
| 0.60×8 | 4.80 | 4 | .80 |
| 0.80×8 | 6.40 | 6 | .40 |
| 0.40×8 | 3.20 | 3 | .20 |
| 0.20×8 | 1.60 | 1 | .60 |




$$(.45)_{10} = (.34631..)_{8}$$

Examples

❑ Example : Conversion of $(123.45)_{10}$ to octal number.

❑ conversion of its integer part first.


| Division | Result | Remainder |
|----------|--------|-----------|
| 123/8 | 15 | 3 |
| 15/8 | 1 | 7 |
| 1/8 | 0 | 1 |



$$(123)_{10} = (173)_8$$

❑ Conversion of fraction portion of this given decimal number

| Multiplication | Result | Integer Portion | Fraction Portion |
|-----------------|--------|-----------------|------------------|
| 0.45×8 | 3.60 | 3 | .60 |
| 0.60×8 | 4.80 | 4 | .80 |
| 0.80×8 | 6.40 | 6 | .40 |
| 0.40×8 | 3.20 | 3 | .20 |
| 0.20×8 | 1.60 | 1 | .60 |



$$(.45)_{10} = (.34631..)_{8}$$

Now, we have got the total octal equivalent of this decimal number is $(173.34631..)_{8}$.

Decimal to Hexadecimal Conversion

❑ Hexadecimal Numbers are represented in base 16. Hence, to convert a decimal number to hexadecimal number, the base of that number is to be changed. Follow the steps given below:

Step 1: Divide the Decimal Number with the base of the number system to be converted to. Here the conversion is to Hex hence the divisor will be 16.

Step 2: The remainder obtained from the division will become the least significant digit of the new number.

Step 3: The quotient obtained from the division will become the next dividend and will be divided by base i.e. 16.

Step 4: The remainder obtained will become the second least significant digit i.e. it will be added in the left of the previously obtained digit.

❑ Now, the steps 3 and 4 are repeated until the quotient obtained becomes 0, and the remainders obtained after each iteration are added to the left of the existing digits

Example

□ Example Conversion of $(1973)_{10}$ to hexadecimal number

| | | | |
|----|--|------|----|
| 16 | | 1973 | |
| 16 | | 123 | 5 |
| | | 7 | 11 |

Here, 11 = B
 $(1973)_{10} = (7B5)_{16}$

□ Example: Conversion of $(450)_{10}$ to hexadecimal number

| Division | Result | Remainder |
|----------|--------|-----------|
| 450/16 | 28 | 2 |
| 28/16 | 1 | 12(C) |
| 1/16 | 0 | 1 |

$(450)_{10} = (1C2)_{16}$

Fraction Cont..

□ Example :Rules for fraction part

Step 1: Multiply the fraction part by 16 and write the integer portion.

Step 2: Again, multiply the resultant fraction part which you have got from the first process and write down the integer part and continue this process until you get zero at the fraction part (minimum 4 to 5 times if it repeats).

Step 3: Now write down the integers you have got at the same order you have got them.

□ Now, let's see an example conversion of fraction number from decimal to hex. We will just implement the given rules in this example.

Cont..

❑ **Example** : Convert $(0.78125)_{10}$ to equivalent hexadecimal number

| Multiplication | Result | Integer Portion | Fraction Portion |
|---------------------|--------|-----------------|------------------|
| 0.78125×16 | 12.50 | 12(C) | .50 |
| 0.50×16 | 8.00 | 8 | .00 |

- ❑ Convert $(125.85)_{10}$ to hexadecimal num
- conversion of **integer** portion first

$(.78125)_{10} = (.C8)_{16}$

| Division | Result | Remainder |
|----------|--------|-----------|
| $125/16$ | 7 | 13(D) |
| $7/16$ | 0 | 7 |

- We have used the rules for converting which we have mentioned above. Now, we will convert its fraction part to equivalent hexadecimal number.

| Multiplication | Result | Integer Portion | Fraction Portion |
|------------------|--------|-----------------|------------------|
| 0.85×16 | 13.60 | 13(D) | .60 |
| 0.60×16 | 9.60 | 9 | .60 |
| 0.60×16 | 9.60 | 9 | .60 |

- Final Solution : hexadecimal equivalence is $(7D.D99...)_{16}$

$(.85)_{10} = (.D99...)_{16}$

Conversion from Binary Number System to Other Number Systems

❑ Binary Numbers are represented with digits 0 and 1 and with base 2. Conversion of a number system means conversion from one base to another.

❑ Binary to Decimal Conversion

- Binary numbers are represented in base 2 but the decimal numbers are of base 10. Hence, to convert the binary number into a decimal number, the base of that number is to be changed. Follow the steps given below:
- **Step 1:** Multiply each digit of the Binary number with the place value of that digit, starting from right to left i.e. from LSB to MSB.
- **Step 2:** Add the result of this multiplication and the decimal number will be formed.

Example

- **Example:** To convert $(1101)_2$ into a decimal number

\square 1 1 0 1 (base-2)

| | |
|------------------|---|
| $1 \times 2^0 =$ | 1 |
| $0 \times 2^1 =$ | 0 |
| $1 \times 2^2 =$ | 4 |
| $1 \times 2^3 =$ | 8 |

$8 + 4 + 0 + 1 = 13$

$1101_2 = 13_{10}$

- **Example:** To convert $(11101011)_2$ into a decimal number

$$(11101011)_2 \longrightarrow (?)_{10}$$
$$1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$
$$128 + 64 + 32 + 0 + 8 + 0 + 2 + 1$$
$$(235)_{10}$$

Fraction Cont..

□ Example :Rules for fraction part

- Determine the positional value of every digit of binary fraction number multiplying every digit by 2^{-1} 2^{-2} 2^{-3} 2^{-4} and so on from left to right.
- Add all these values which will give the equivalent value of fractional binary number to decimal number.

□ For example,

$$(1110.011)_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\ = (15.375)_{10}$$

□ Convert the following number into decimal equivalent number

a. 1111 b.101.101 c. 00011 d.101010 e. 0.00001 f.1010.1111

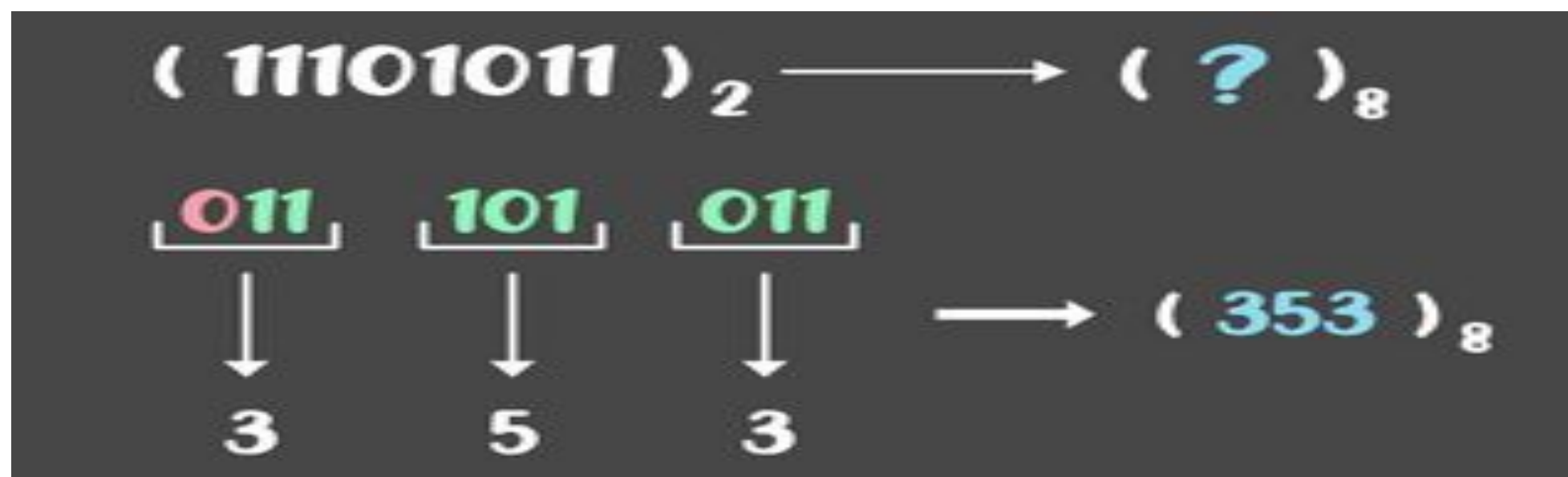
Binary to Octal Conversion

□ Binary numbers are represented in base 2 but the octal numbers are of base 8. Hence, to convert the binary number into octal number, the base of that number is to be changed. Follow the steps given below:

Step 1: Divide the binary number into groups of three digits starting from right to left i.e. from LSB to MSB. and fill up by taking zeros if there is not available digit at the left to consist 3 bits.

Step 2: Convert these groups into equivalent octal digits. Replace every 3 bit by their equivalent octal value

- **Example:** To convert $(11101011)_2$ into an octal number



Example

Example : Conversion of $(1110101011)_2$ to equivalent octal number

$$\begin{array}{cccc} \underbrace{001} & \underbrace{110} & \underbrace{101} & \underbrace{011} \\ 1 & 6 & 5 & 3 \end{array}$$

$$\text{So, } (1110101011)_2 = (1653)_8$$

Rules for fraction number :

Take every 3 bit from left to right and fill up by taking zeros if there is not available digit at the right to consist 3 bits.

Replace every 3 bit by their equivalent octal value.

- Although the concept is same as the conversion of integer part we will also see an example of converting a fraction number from binary to octal below

Example : Conversion of $(0.10001000111)_2$ to equivalent octal number.

- Here also we will take each 3 digits of binary number from left to right. We can take necessary zeros at the rightmost side to fill up the last pair

$$\begin{array}{cccc} \underbrace{100} & \underbrace{010} & \underbrace{001} & \underbrace{110} \\ 4 & 2 & 1 & 6 \end{array}$$

$$\text{So, } (0.10001000111)_2 = (4216)_8$$

Example

❑ Example : Conversion of $(10010111.11001)_2$ to equivalent octal number.

- Now, let's go to the conversion method bellow.

$$\begin{array}{ccccc} \underbrace{010}_2 & \underbrace{010}_2 & \underbrace{111}_7 & \underbrace{.110}_{.6} & \underbrace{010}_2 \\ \text{So, } (10010111.11001)_2 & = & (227.62)_8 \end{array}$$

- Example : Conversion of 101010011.110100 to equivalent octal number

| | |
|------------------------|--|
| Binary Number: | 101010011.110100 |
| Group of three digits: | $\begin{array}{ccccc} \overline{101} & \overline{010} & \overline{011} & \overline{.110} & \overline{100} \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array}$ |
| Octal Equivalent: | 5 2 3 6 4 |
| | $= (523.64)_8$ |

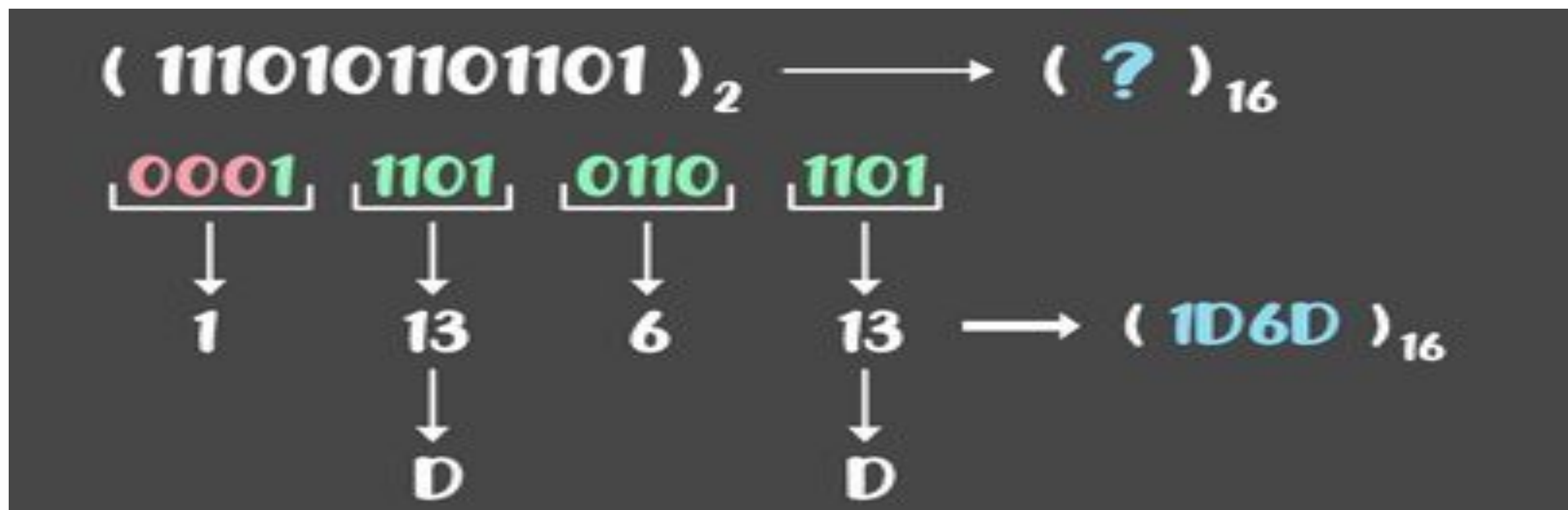
Binary to Hex

❑ Binary numbers are represented in base 2 but the Hexadecimal numbers are of base 16. Hence, to convert the binary number into Hex number, the base of that number is to be changed. Follow the steps given below:

Step 1: Divide the binary number into groups of four digits starting from right to left i.e. from LSB to MSB. and fill up by taking zeros if there is not available digit at the left to consist 4 bits.

Step 2: Replace every 4 bit by their equivalent hexadecimal value.

Example: To convert $(1110101101101)_2$ into a hex number



Examples

❑ Example : Convert 10111100111_2 to equivalent hexadecimal number

$$\begin{array}{ccc} \underbrace{0101} & \underbrace{1110} & \underbrace{0111} \\ 5 & E & 7 \end{array}$$

$$\text{So, } (10111100111)_2 = (5E7)_{16}$$

❑ Rules for fraction number

Take every 4 bit from left to right and fill up by taking zeros if there is not available digit at the right to consist 4 bits.

Replace every 4 bit by their equivalent hexadecimal value.

❑ Example : Conversion of 0.11001101001_2 to equivalent hexadecimal number

$$\begin{array}{ccc} \underbrace{.1100} & \underbrace{1101} & \underbrace{0010} \\ C & D & 2 \end{array}$$

$$\text{So, } (0.11001101001)_2 = (0.CD2)_{16}$$

Examples

- Example : Conversion of 10100110.10101_2 to equivalent hexadecimal number

$$\begin{array}{cccc} \underbrace{1010} & \underbrace{0110} & \underbrace{.1010} & \underbrace{1000} \\ A & 6 & A & 9 \end{array}$$

So, $(10100110.10101)_2 = (A6.A9)_{16}$

- Example 2 Conversion of $(1011010)_2$ to equivalent hexadecimal number

101 **1010**
group 2 group 1

*Group 2 containing only 3 bits,
so add 0 to the left*

0101 **1010**
↓ ↓
5 **A**

Binary 01010010 is equal to 5A

$$01010010_2 = 5A_{16}$$

Conversion from Octal Number System to Other Number Systems

❑ Octal Numbers are represented with digits 0-7 and with base 8. Conversion of a number system means conversion from one base to another.

❑ Following are the conversions of the Octal Number System to other Number Systems:

Octal to decimal

Octal to binary

Octal to Hex

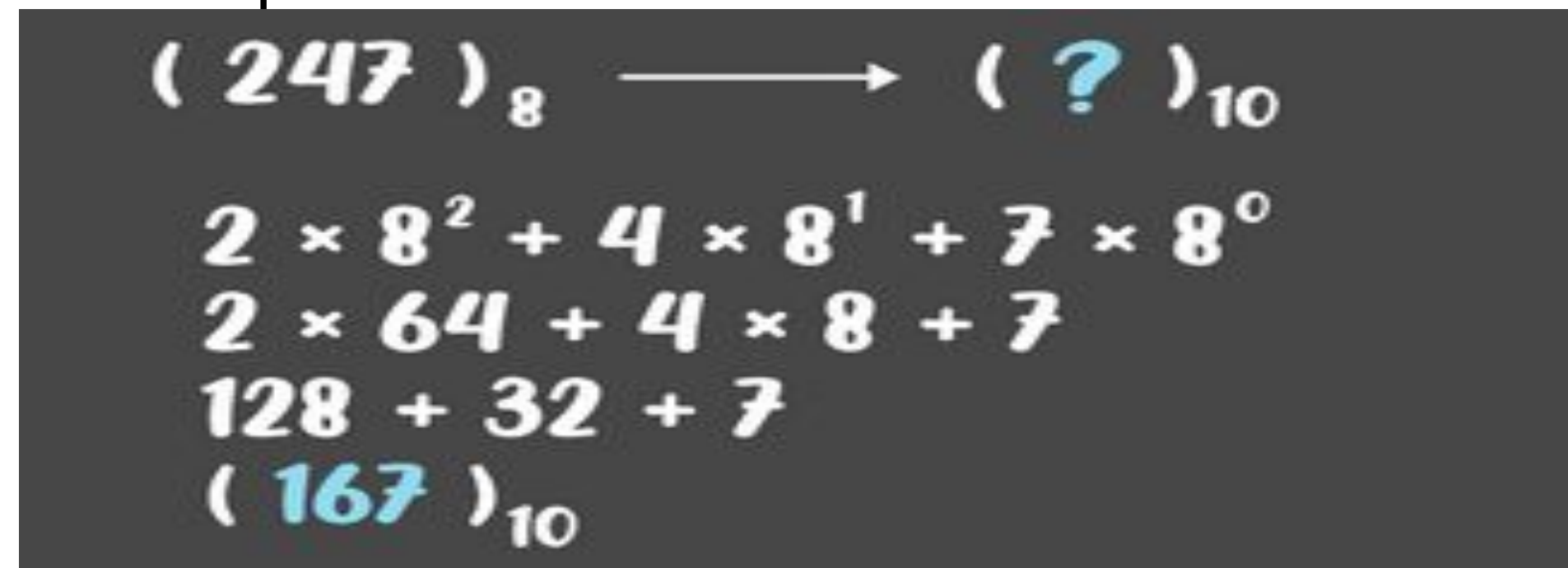
❑ Octal to Decimal Conversion:

❑ Octal numbers are represented in base 8, but the decimal numbers are of base 10. Hence, to convert an octal number to a decimal number, the base of that number is to be changed

Octal to decimal

□ Follow the steps given below:

- **Step 1:** Multiply each digit of the Octal number with the place value of that digit, starting from right to left i.e. from LSB to MSB. Multiply every digit by 8⁰, 8¹, 8², 8³, 8⁴ and so on from right to left.
- **Step 2:** Add the result of this multiplication and the decimal number will be formed. Example


$$\begin{aligned} (247)_8 &\longrightarrow (?)_{10} \\ 2 \times 8^2 + 4 \times 8^1 + 7 \times 8^0 \\ 2 \times 64 + 4 \times 8 + 7 \\ 128 + 32 + 7 \\ (167)_{10} \end{aligned}$$

- **Example :** Conversion of (16512)₈ to equivalent decimal number.
- **Solution :** $(16512)_8$
 $= 1 \times 8^4 + 6 \times 8^3 + 5 \times 8^2 + 1 \times 8^1 + 2 \times 8^0$
 $= 4096 + 3072 + 320 + 8 + 2$
 $= 7498$
- So, we can tell that the decimal equivalent of octal number 16512 is 7498.

Example

❑ Rules for fraction part of the number:

- ❑ Determine the positional value of every digit of octal fraction number multiplying every digit by 8^{-1} , 8^{-2} , 8^{-3} , 8^{-4} and so on from left to right.
- ❑ Add all these values which will give the equivalent value of fractional octal number to decimal number.

❑ $(0.145)_8$

$$\begin{aligned} &= 1 \times 8^{-1} + 4 \times 8^{-2} + 5 \times 8^{-3} \\ &= 0.125 + 0.0588 + 0.0098 \\ &= 0.1936 \end{aligned}$$

Note: Here, we have taken everything from 3 digits after **radix point**.

Example

□ Convert 5217 into decimal

□ 5 2 1 7 (base-8)

| | | |
|---|-------------------------------|------|
| 7 | $\times 8^0 = 7 \times 1 =$ | 7 |
| 1 | $\times 8^1 = 1 \times 8 =$ | 8 |
| 2 | $\times 8^2 = 2 \times 64 =$ | 128 |
| 5 | $\times 8^3 = 5 \times 512 =$ | 2560 |

$$2560 + 128 + 8 + 7 = 2703$$
$$5217_8 = 2703_{10}$$

□ Convert 1725.43 into decimal

$(1725.43)_8 = (?)_{10}$

| | | | | | | | | | | |
|----------------|------------------|------------------|------------------|----|---------------------|---------------------|---|-----|---|----------|
| 1 | 7 | 2 | 5 | . | 4 | 3 | | | | |
| 1×8^3 | $+ 7 \times 8^2$ | $+ 2 \times 8^1$ | $+ 5 \times 8^0$ | | $+ 4 \times 8^{-1}$ | $+ 3 \times 8^{-2}$ | | | | |
| 512 | + | 448 | + | 16 | + | 5 | + | 0.5 | + | 0.046875 |

$$= 981.546875$$

$\therefore (1725.43)_8 = (981.546875)_{10}$

Octal to Binary Conversion

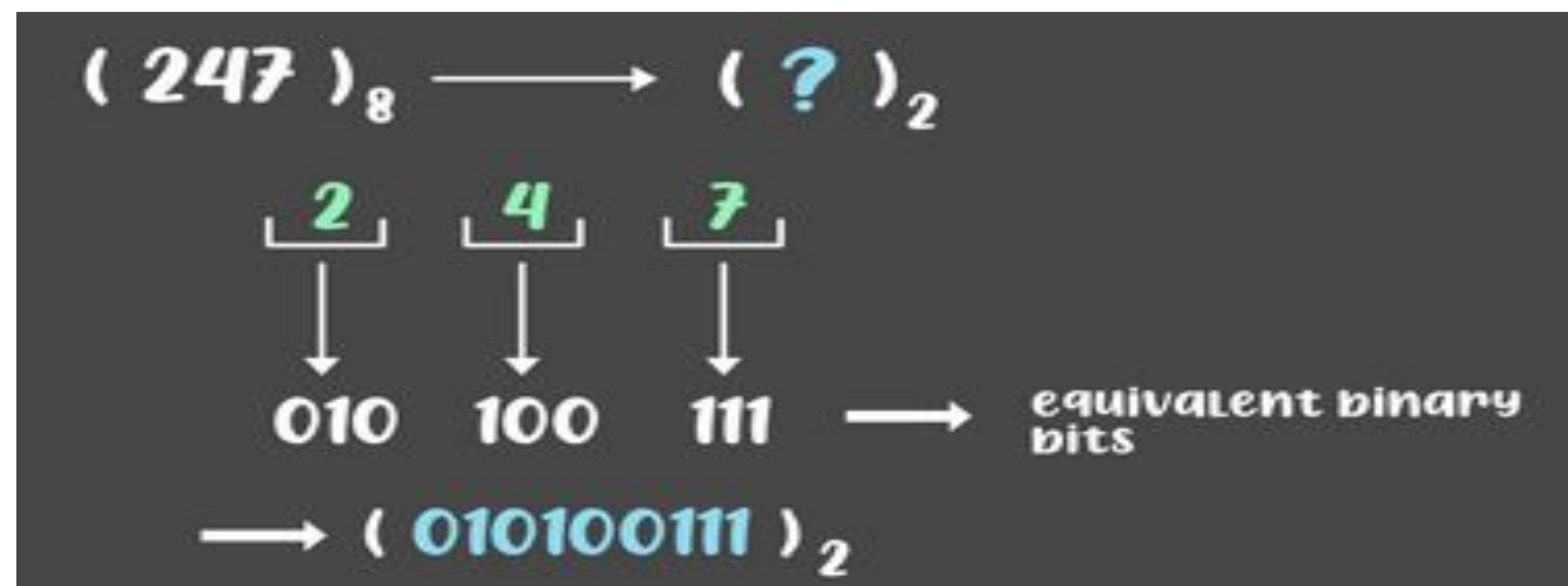
❑ Octal numbers are represented in base 8, but the binary numbers are of base 2. Hence, to convert an octal number to a binary number, the base of that number is to be changed. Follow the steps given below:

Step 1: Write each digit of the octal number separately.

Step 2: Convert each digit into an equivalent group of three binary digits.

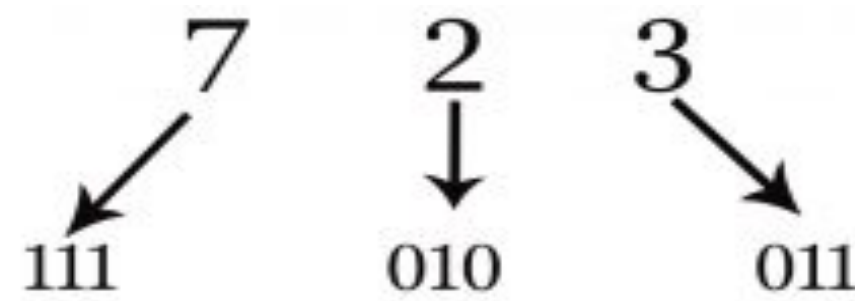
Step 3: Combine these groups to form the whole binary number.

- **Example:** $(247)_8$ is to be converted to binary



Example

- **Example : Conversion of 7238 to equivalent binary number**

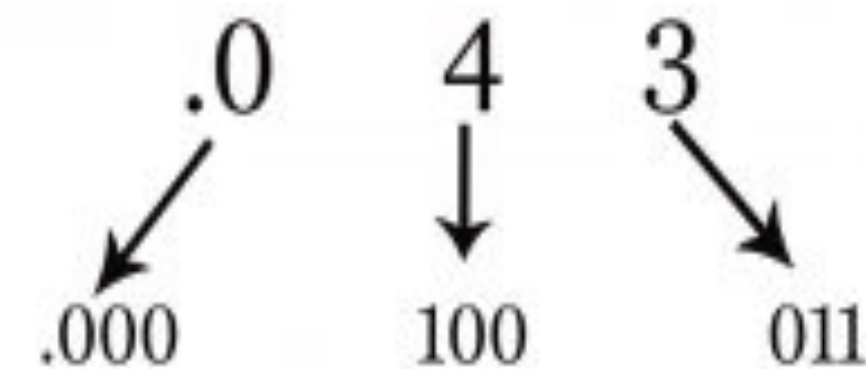


$$\text{So, } (723)_8 = (111010011)_2$$

- **Rules for fraction number**

- Replace every octal digit by their equivalent binary value of 3 bits.
- Write them one after another with the same order as octal number.

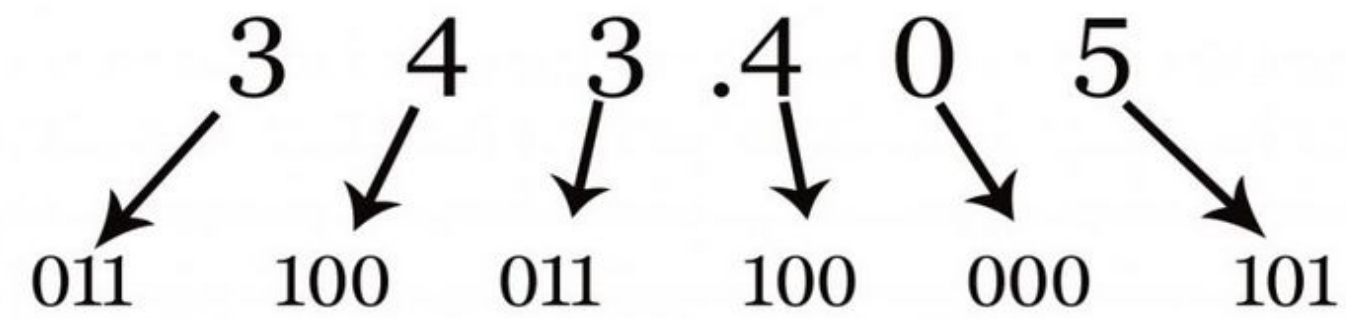
- **Example: Convert 0.0438 to equivalent binary number.**



$$\text{So, } (0.043)_8 = (0.000100011)_2$$

Example

- Example: Conversion of 343.4058 to equivalent binary number



So, $(343.405)_8 = (11100011.100000101)_2$

- Convert 12.5 and 721 to to equivalent binary number

$$\begin{array}{ccc} \mathbf{1} & \mathbf{2} & \mathbf{.5} \\ / & / & / \\ 001 & 010 & 101 \end{array}$$
$$(12.5)_8 = (001010.101)_2$$

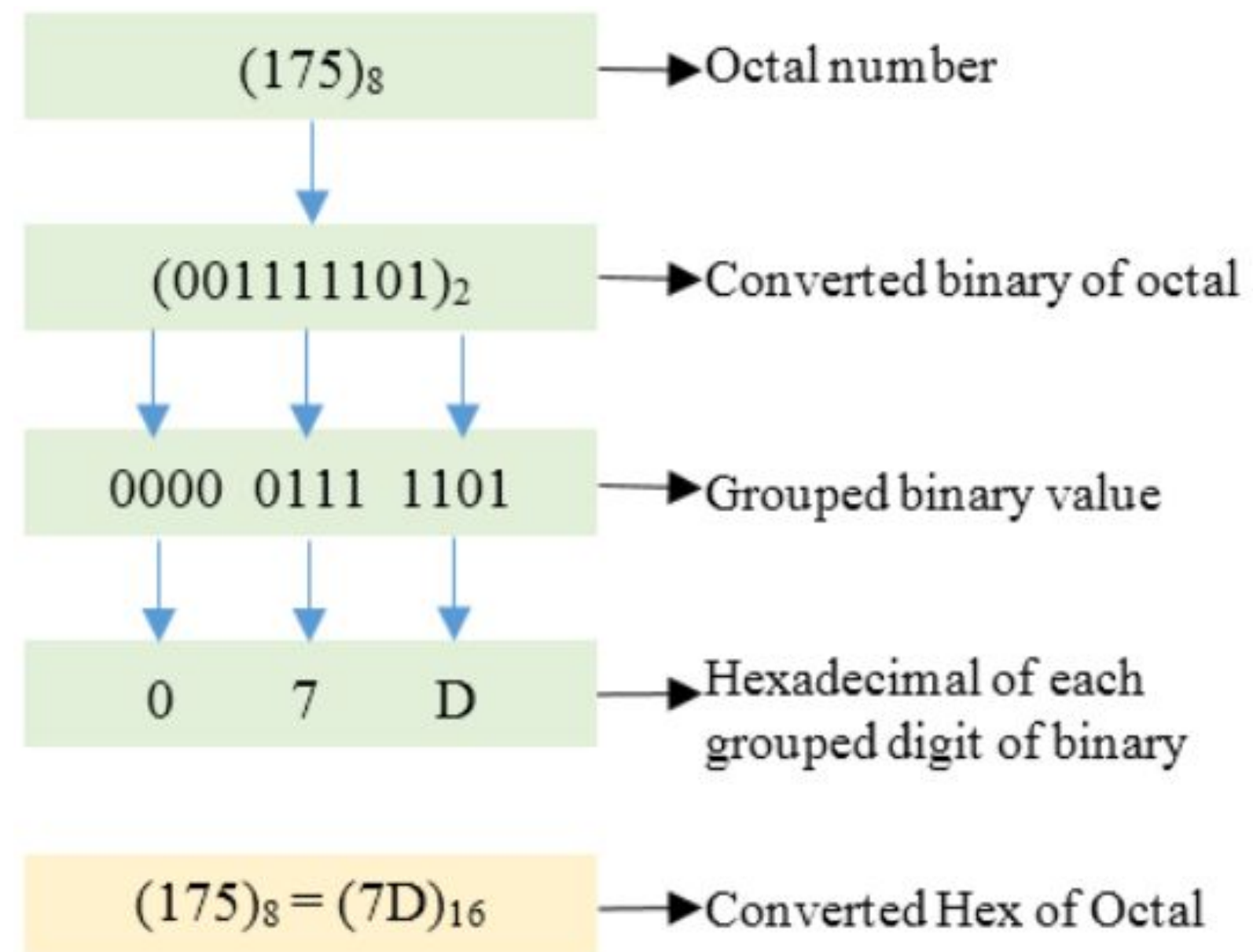
$$\begin{array}{ccc} \mathbf{7} & \mathbf{2} & \mathbf{1} \\ / & / & / \\ 111 & 010 & 001 \end{array}$$
$$(721)_8 = (111010001)_2$$

Octal to Hexadecimal Conversion:

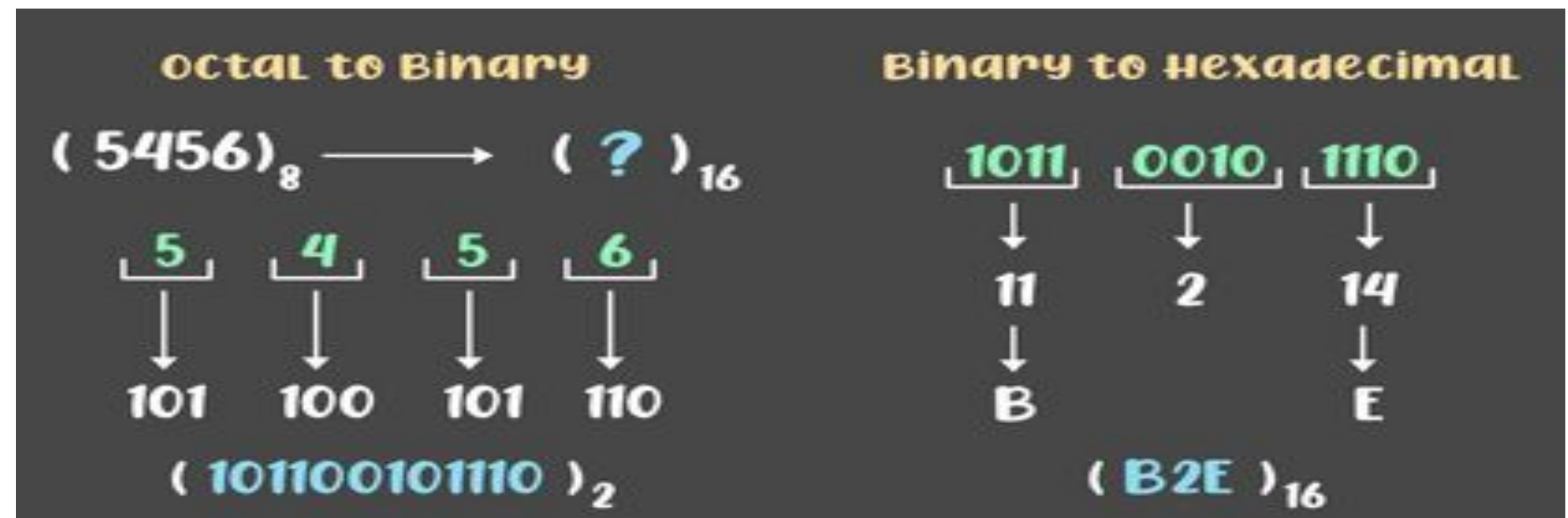
- Octal numbers are represented in base 8, but the hexadecimal numbers are of base 16. Hence, to convert an octal number to a hex number, the base of that number is to be changed. Follow the steps given below:
- **Step 1:** We need to convert the Octal number to Binary first. For that, follow the steps given in the above conversion.
- **Step 2:** Now to convert the binary number to Hex number, divide the binary digits into groups of four digits starting from right to left i.e. from LSB to MSB.
- **Step 3:** Add zeros prior to MSB to make it a proper group of four digits(if required)
- **Step 4:** Now convert these groups into their relevant decimal values.
- **Step 5:** For values from 10-15, convert it into Hex symbols i.e from A-F

Example

- Convert 175 into hexadecimal format

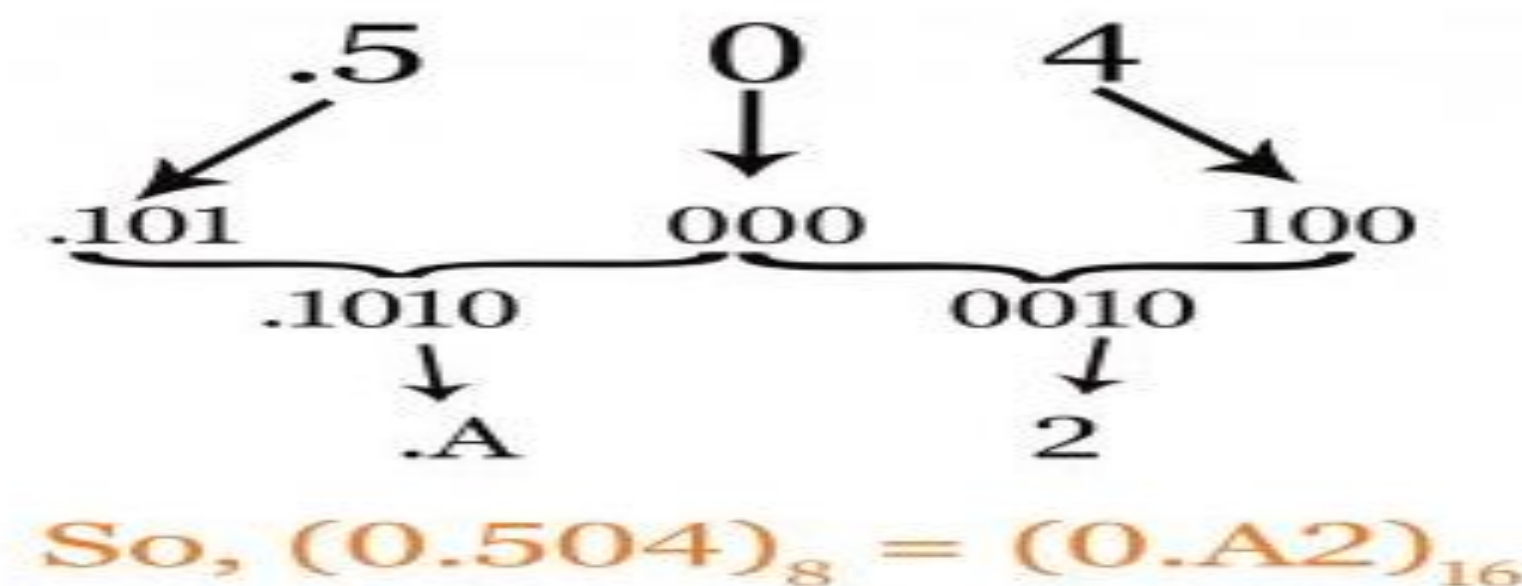


- Example:** $(5456)_8$ is to be converted to hex



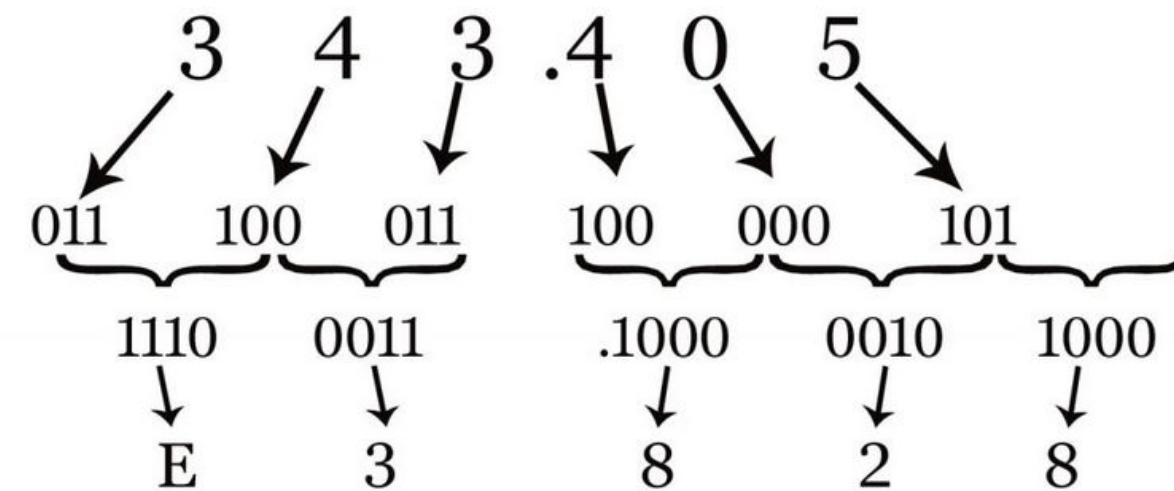
Cont..

- Rules for fraction number
- Replace each octal digit by their equivalent binary number and write them side by side as their octal order.
- Take every 4 bit from left to right and fill up by taking zeros if there is not available digit at the right to consist 4 bits.
- Replace every 4 bit by their equivalent hexadecimal value.
- **Example : Conversion of 0.5048 to equivalent hexadecimal number**



Example

- Example: Conversion of 343.4058 to equivalent hexadecimal number



So, $(343.405)_8 = (E3.828)_{16}$

- Example 1 : Convert $(456)_8$ into Hexadecimal Number system.

$$\begin{aligned}
 456 \text{ (octal)} &= (100) \ (101) \ (110) \\
 &\quad \quad \quad 4 \quad \quad 5 \quad \quad 6 \\
 &= 100101110 \text{ (binary)} \\
 100101110 \text{ (binary)} &= (1) \quad (0010) \ (1110) \\
 &= (0001) \ (0010) \ (1110) \\
 &\quad \quad \quad 1 \quad \quad 2 \quad \quad E \\
 &= 12E \text{ (hexadecimal)}.
 \end{aligned}$$

Conversion from Hexadecimal Number System to Other Number Systems

❑ Hex Numbers are represented with digits 0-9 and with letters A-F and with base 16. Conversion of a number system means conversion from one base to another. Following are the conversions of the Hexadecimal Number System to other Number Systems:

- ❑ Hex to decimal
- ❑ Hex to binary and
- ❑ Hex to octal

Hexadecimal to Decimal Conversion

❑ Hexadecimal numbers are represented in base 16 but the decimal numbers are of base 10. Hence, to convert a hexadecimal number to a decimal number, the base of that number is to be changed. Follow the steps given below:

Step 1: Write the decimal values of the symbols used in the Hex number i.e. from A-F

Step 2: Multiply each digit of the Hex number with its place value. starting from right to left i.e. LSB to MSB. Determine the positional value of every digit of hexadecimal number. Multiply every digit by 16^0 16^1 16^2 16^3 16^4 and so on from right to left.

Step 3: Add the result of multiplications and the final sum will be the decimal number.

Example: To convert $(8EB4)_{16}$ into a decimal value

$$\begin{aligned} & (8EB4)_{16} \longrightarrow (?)_{10} \\ & \begin{array}{cccc} 8 & 14 & 11 & 4 \\ 8 \times 16^3 & + & 14 \times 16^2 & + & 11 \times 16^1 & + & 4 \times 16^0 \\ 32768 & + & 3584 & + & 176 & + & 4 \\ & (36532)_{10} \end{array} \end{aligned}$$

fraction part Example

- ❑ Rules for fraction hex number
- ❑ Determine the positional value of every digit of hexadecimal fraction number multiplying every digit by 16^{-1} 16^{-2} 16^{-3} 16^{-4} and so on from left to right.
- ❑ Add all these values which will give the equivalent value of fractional hexadecimal number to decimal number.
- ❑ Example: Convert 0.2A5₁₆ to equivalent decimal number.

$$\begin{aligned}(0.2A5)_{16} &= 2 \times 16^{-1} + 10 \times 16^{-2} + 5 \times 16^{-3} \\ &= 0.125 + 0.0391 + 0.0012 \\ &= 0.1653\end{aligned}$$

Note: We have taken 4 number after radix point here.

Example

- Example 1 : Convert $(ABC)_{16}$ into decimal number.
- Since, $A = 10$, $B = 11$ & $C = 12$. The equivalent decimal number is calculated below:

$$\begin{aligned} &= (ABC)_{16} \\ &= (10 \times 16^2 + 11 \times 16^1 + 12 \times 16^0)_{10} \\ &= (256 + 176 + 12)_{10} \\ &= (444)_{10} \end{aligned}$$

- Example 2 : Convert $(1F.01B)_{16}$ into decimal number.
- Since, $B = 11$ & $F = 15$. Hence, equivalent decimal number is calculated below:

$$\begin{aligned} &= (1F.01B)_{16} \\ &= (1 \times 16^1 + 15 \times 16^0 + 0 \times 16^{-1} + 1 \times 16^{-2} + 11 \times 16^{-3})_{10} \\ &= (31.0065918)_{10} \text{ which is answer.} \end{aligned}$$

Hex to Binary conversion

Hex numbers are represented in base 16, but the binary numbers are of base 2. Hence, to convert a hexadecimal number to a binary number, the base of that number is to be changed. Follow the steps given below:

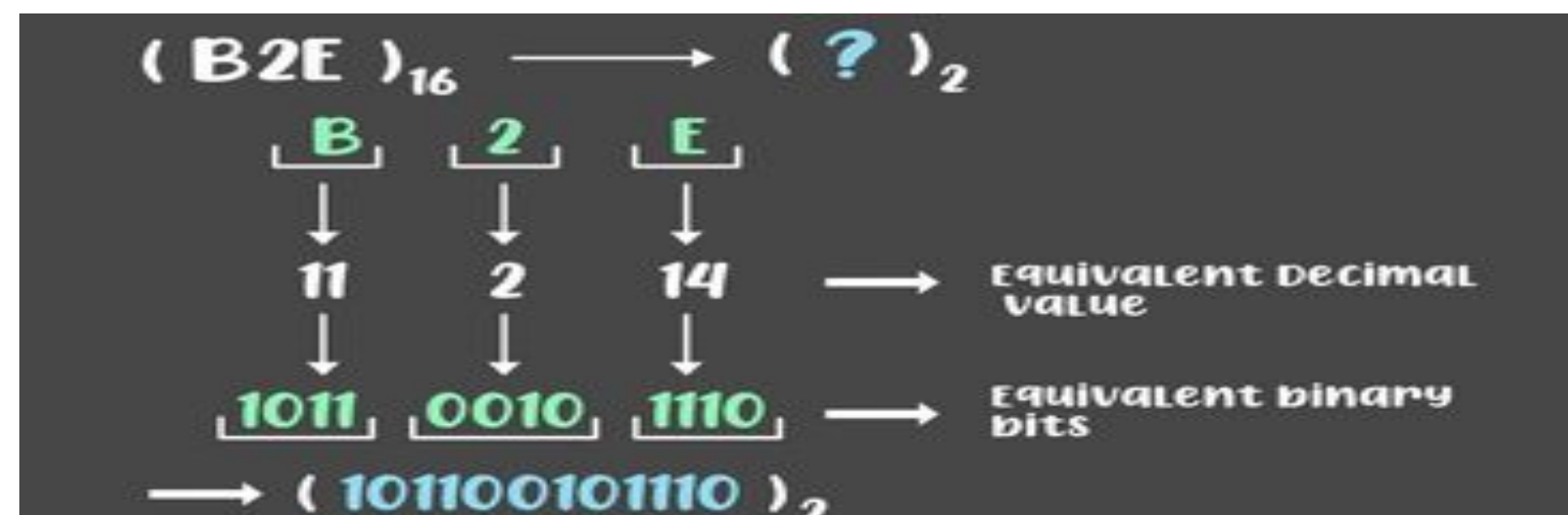
Step 1: Convert the Hex symbols into its equivalent decimal values.

Step 2: Write each digit of the Hexadecimal number separately.

Step 3: Convert each digit into an equivalent group of four binary digits.

Step 4: Combine these groups to form the whole binary number.

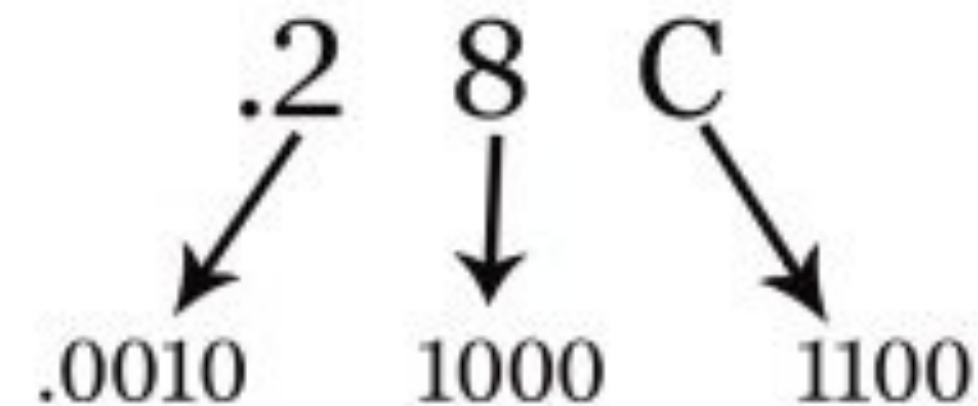
Example: (B2E)₁₆ is to be converted to binary



fraction part Example

□ Rules for fraction number

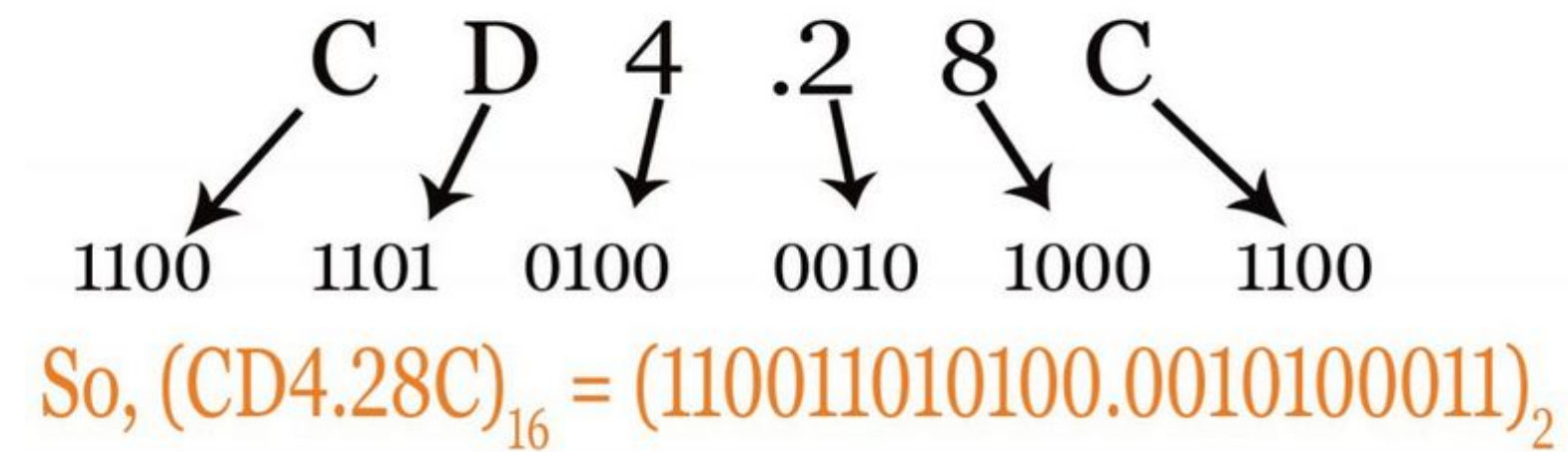
- Replace every hexadecimal digit by their equivalent binary value of 4 bits.
- Write them one after another with the same order as hexadecimal number.
- All these rules are the same as integer conversion. Here we will use the same method for converting the given number from hexadecimal to binary.
- **Example: Conversion of (0.28C)₁₆ to equivalent binary number**



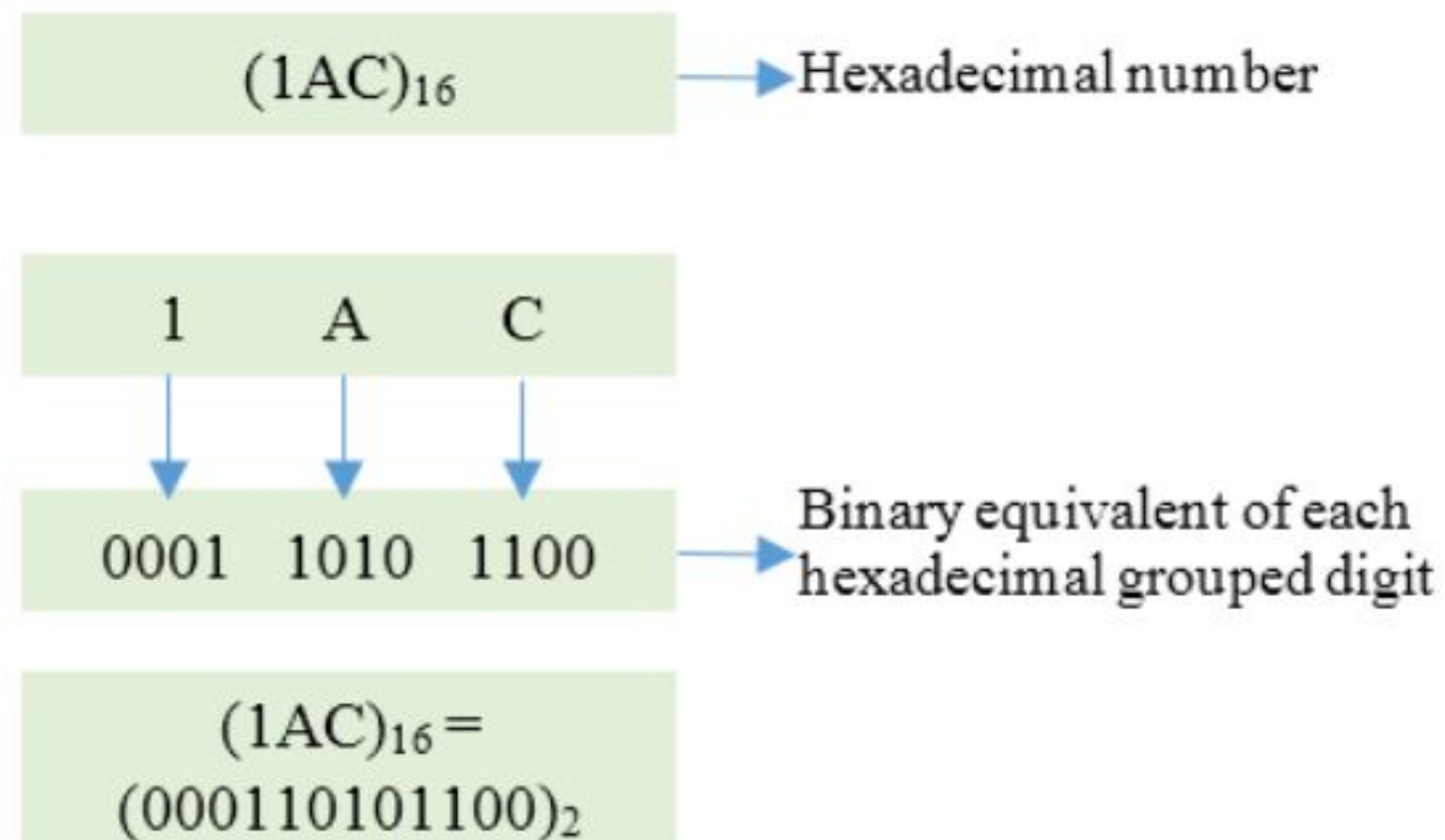
$$\text{So, } (0.28C)_{16} = (0.0010100011)_2$$

Example

- Example: Convert $(CD4.28C)_{16}$ to equivalent binary number



- Example: Convert $(1AC)_{16}$ to equivalent binary number



Hexadecimal to Octal Conversion

❑ Hexadecimal numbers are represented in base 16, but the octal numbers are of base 8. Hence, to convert a hex number to an octal number, the base of that number is to be changed. Follow the steps given below:

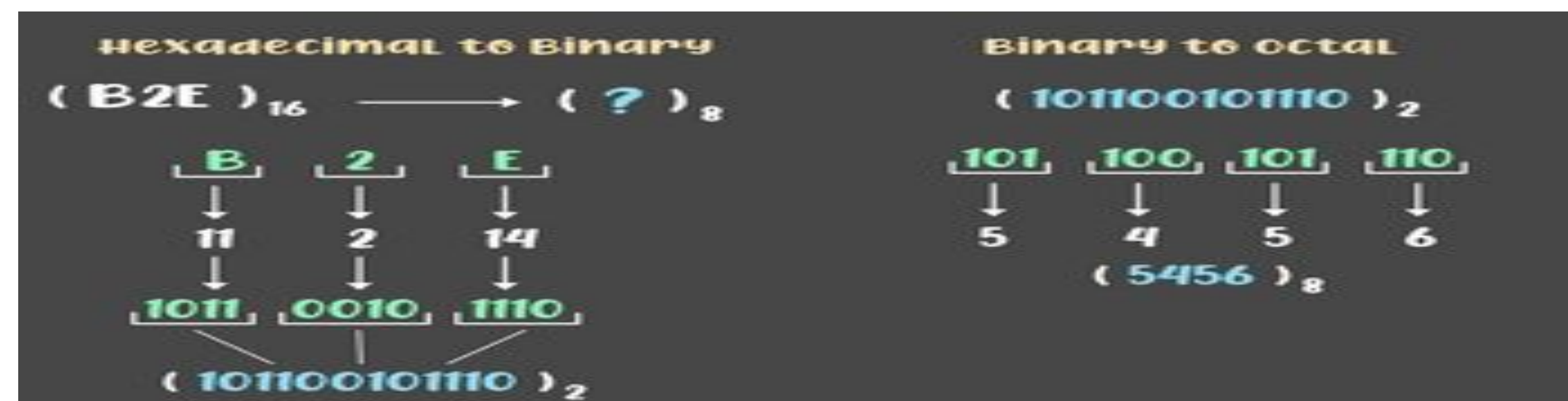
Step 1: We need to convert the Hexadecimal number to Binary first. For that, follow the steps given in the above conversion.

Step 2: Now to convert the binary number to Octal number, divide the binary digits into groups of three digits starting from right to left i.e. from LSB to MSB.

Step 3: Add zeros prior to MSB to make it a proper group of three digits(if required)

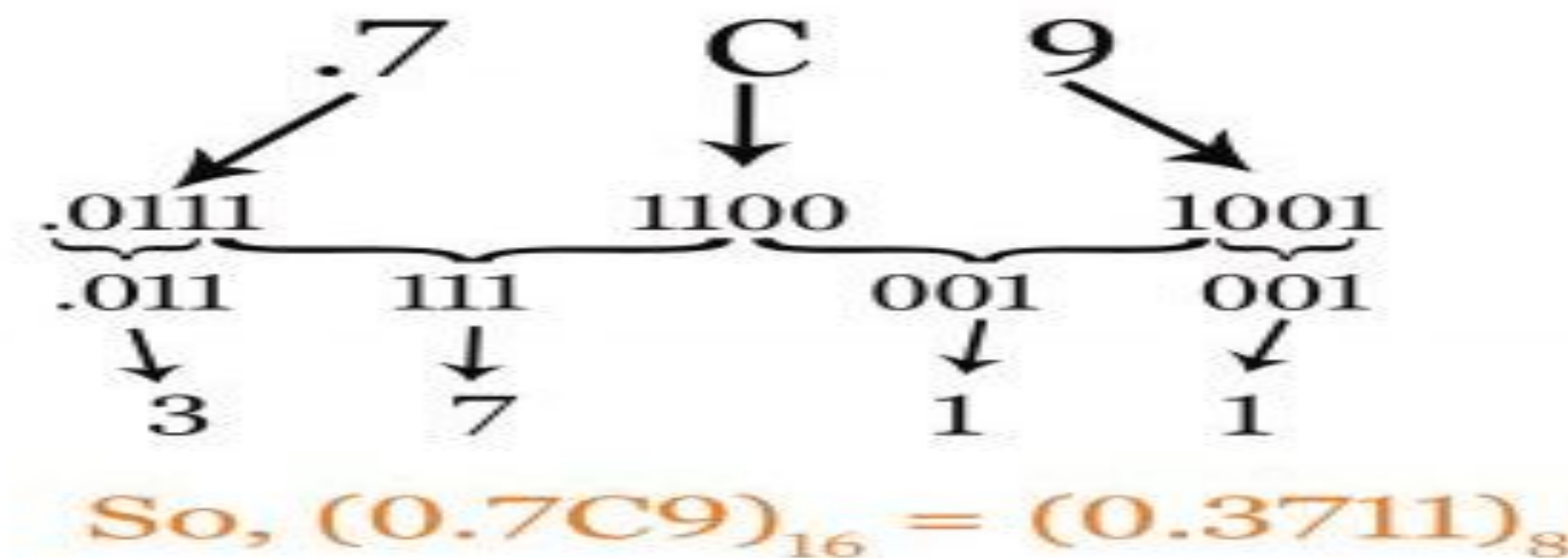
Step 4: Now convert these groups into their relevant decimal values

Example: (B2E)₁₆ is to be converted to hex



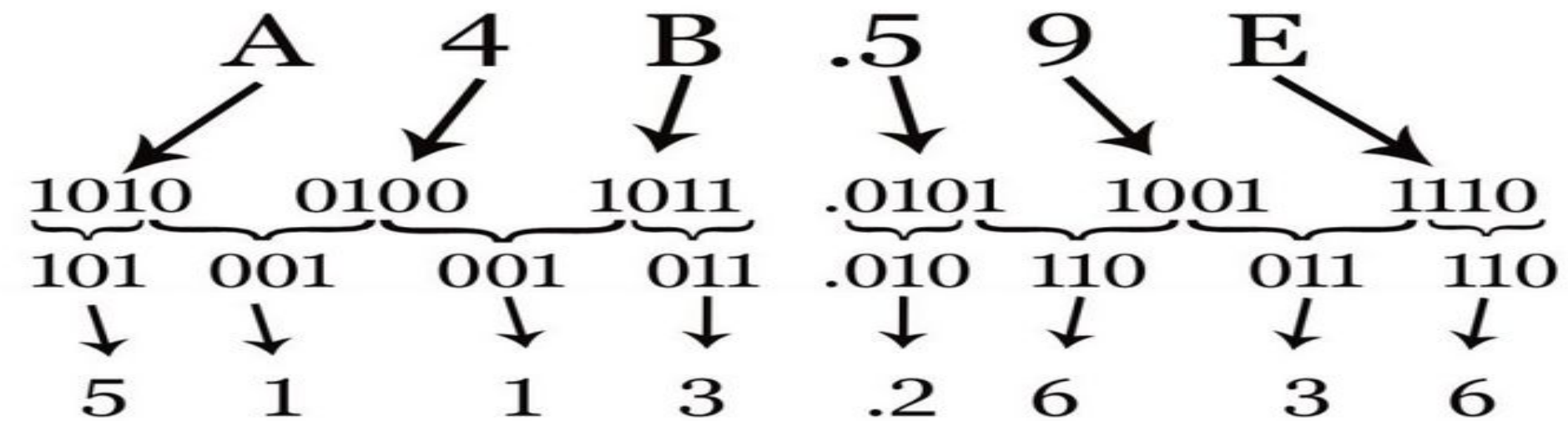
fraction part Example

- Rules for fraction number
- Replace each hexadecimal digit by their equivalent binary number and write them side by side as their hexadecimal order.
- Take every 3 bit from left to right and fill up by taking zeros if there is not available digit at the right to consist 3 bits.
- Replace every 3 bit by their equivalent hexadecimal value.
- **Example: Conversion of $(0.7C9)_{16}$ to equivalent octal number.**



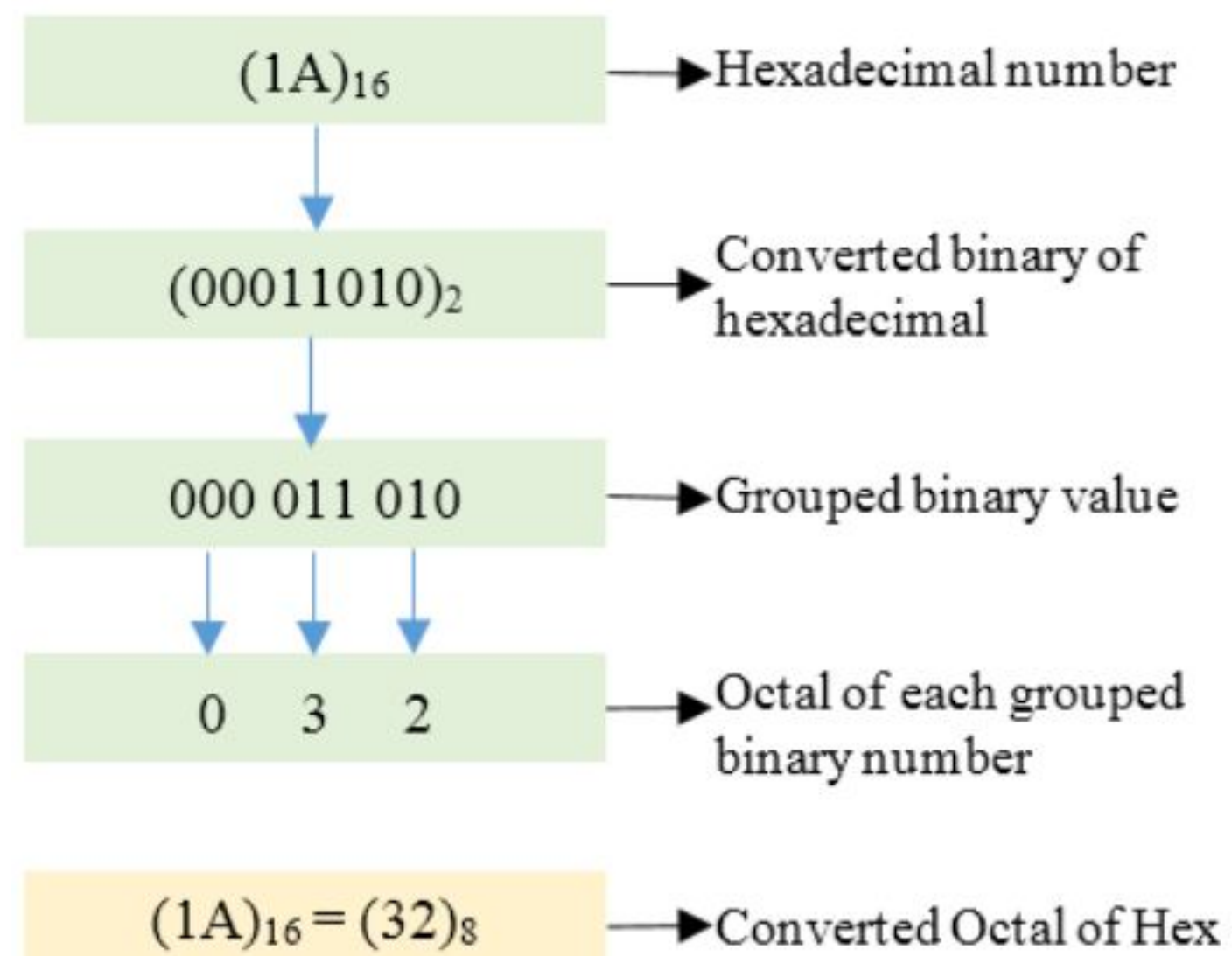
Example

- Conversion of $(A4B.59E)_{16}$ to equivalent octal number



So, $(A4B.59E)_{16} = (5113.2636)_8$

- Conversion of $(1A)_{16}$ to equivalent octal number

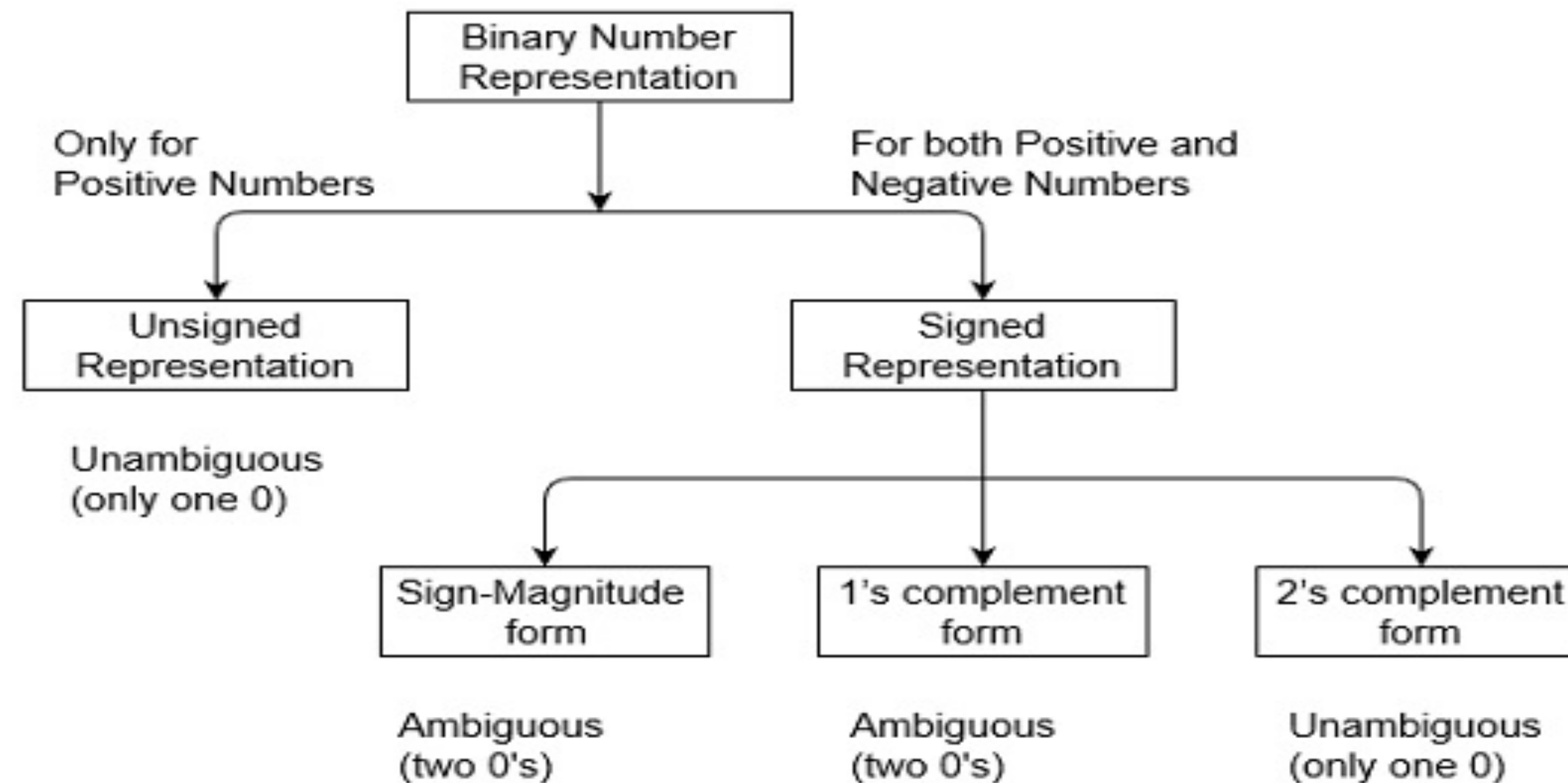


Practice Problems on Number System Conversion

- Convert 146_{10} into a binary number system
 - Convert $1A7_{16}$ into the decimal number system
 - Convert $(110010)_2$ into octal number system
 - Convert $DA2_{16}$ into the binary number system
 - Convert 4652_8 into the binary number system
- Convert these binary system numbers to decimal system numbers
 - 100101101
 - 11100.1001
 - 111111
 - 100000.0111
 - Convert these decimal system numbers to binary system numbers
 - 127
 - 38
 - 22.5
 - 764.375

Ones and Twos Complement

- 1s complement and 2s complement are way of representing the signed binary numbers.
- In general, the binary number can be represented in two ways.
- Unsigned Binary Numbers
- Signed Binary Numbers

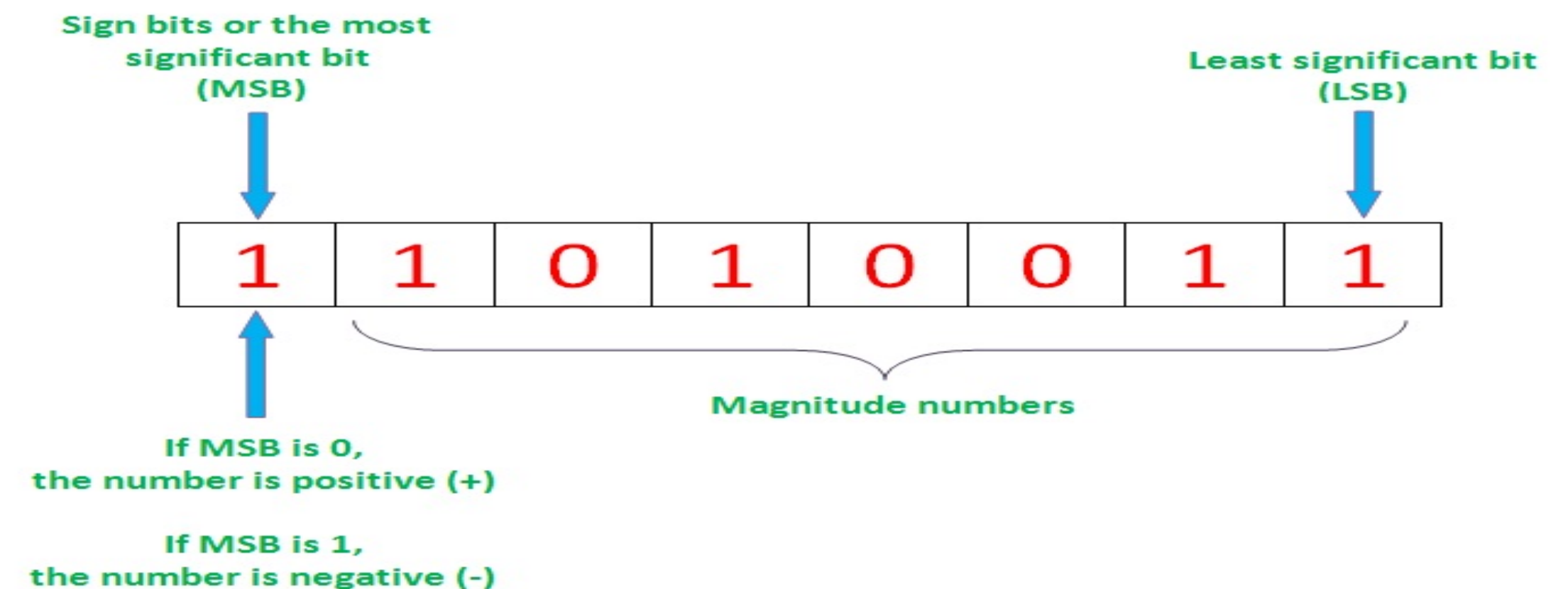
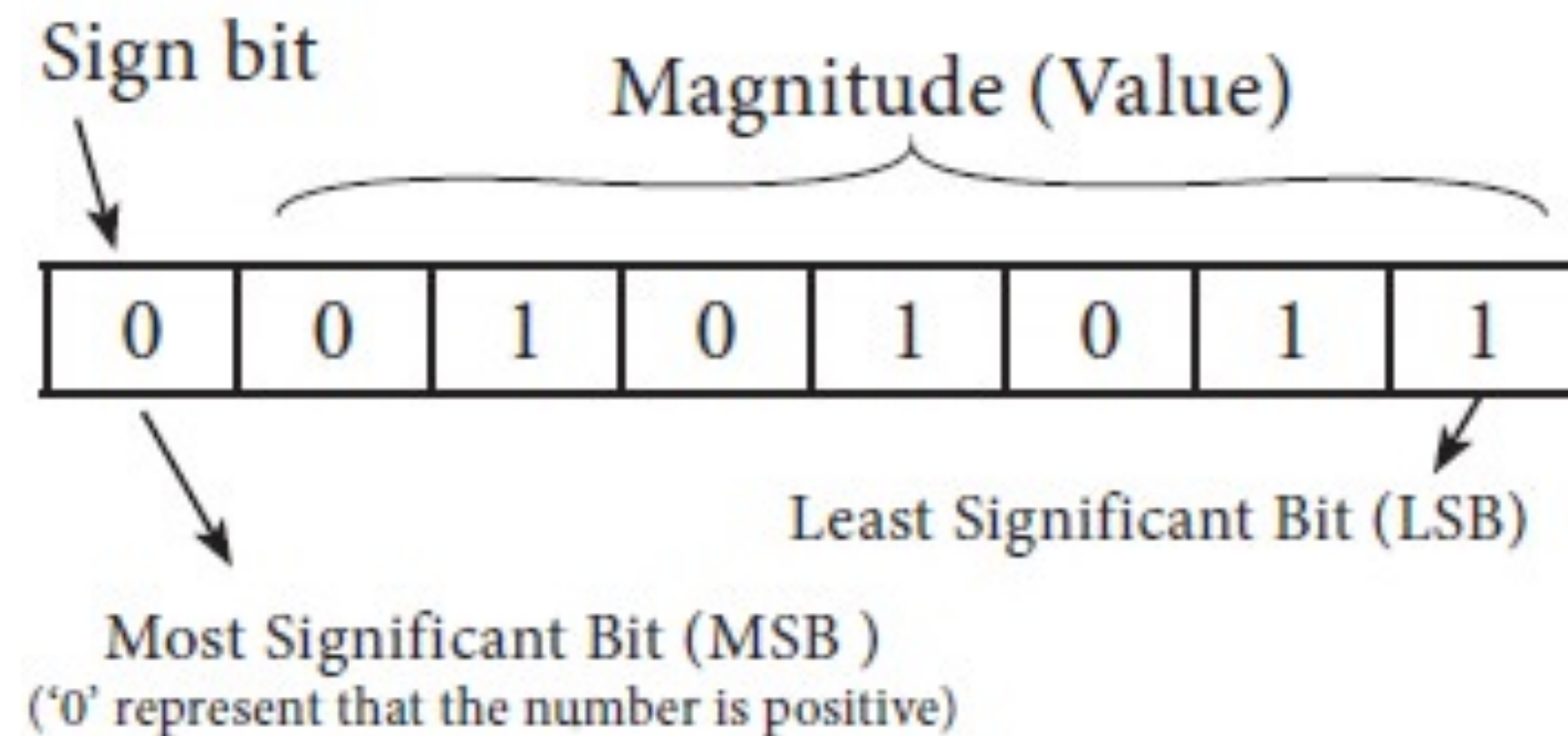


Unsigned binary numbers

- *Unsigned binary numbers* are, by definition, positive numbers and thus do not require an arithmetic sign. For n-bit unsigned binary numbers, all n-bits are used to represent the magnitude of the number.
- For example, if we represent decimal 12 in 5- bit unsigned number form then $(12)_{10} = (01100)_2$. Here all 5 bit are used to represent the magnitude of the number.
- In unsigned binary number representation, using n-bits, we can represent the numbers from 0 to $2^n - 1$. For example, using 4 -bits we can represent the number from 0 to 15 in unsigned binary number representation.

Signed Binary Numbers

- Using signed binary number representation both positive and negative numbers can be represented. Signed numbers contain sign flag, this representation distinguish positive and negative numbers.
- In signed binary number representation the most significant bit (MSB) of the number is a sign bit. For positive numbers, the sign bit is 0 and for negative number, the sign bit is 1.



signed binary numbers representation

- There are three different ways the signed binary numbers can be represented.
- Signed Magnitude Form
- 1's Complement Form
- 2's Complement Form

Signed Magnitude Form

- ❑ In the sign-magnitude representation method, a number is represented in its binary form. The most significant bit (MSB) represents the sign. A 1 in the MSB bit position denotes a negative number; a 0 denotes a positive number. The remaining $n - 1$ bits are preserved and represent the magnitude of the number.
- ❑ Here is the representation of + 34 and -34 in a 8-bit sign-magnitude form.

+ 34 = 0 0 1 0 0 0 1 0

- 34 = 1 0 1 0 0 0 1 0

- ❑ Since the magnitude of both numbers is the same, the first 7 bits in the representation are the same for both numbers. For +34, the MSB is 0, and for -34, the MSB or sign bit is 1.
- ❑ In sign magnitude representations, there are two different representations for 0.

+ 0 = 0 0 0 0 0 0 0 0

- 0 = 1 0 0 0 0 0 0 0

Example

- Convert the given number into binary format and represent using signed magnitude format

(a).

| | |
|------|----------------|
| 41/2 | |
| 20/2 | 1 → LSB |
| 10/2 | 0 |
| 5/2 | 0 |
| 2/2 | 1 |
| 1/2 | 0 |
| 0 | 1 → MSB |

$$(+ 41)_{10} = (\mathbf{0} \ \underline{0101001})_2$$

(+) Sign ⌞ ⌞ Magnitude

(b).

| | |
|------|----------------|
| 47/2 | |
| 23/2 | 1 → LSB |
| 11/2 | 1 |
| 5/2 | 1 |
| 2/2 | 1 |
| 1/2 | 0 |
| 0 | 1 → MSB |

$$(- 47)_{10} = (\mathbf{1} \ \underline{0101111})_2$$

(-) Sign ⌞ ⌞ Magnitude

1's Complement Representation

- 1's complement of a number is obtained by inverting each bit of given number. So, we represent positive numbers in binary form and negative numbers in 1's complement form

- To get 1's complement of a number just inverse the digit. Change all 0's to 1 and all 1's to 0

- For Example

Binary representation of 5 is: 0 1 0 1

1's Complement of 5 is: 1 0 1 0

- Using n-bits, the range of numbers that can be represented in 1's complement is from $(2^n - 1) - B$

- Where n= total number of bit and B= number

- For **Example :** Convert -5_{10} to 4-bit 1's complement

$$\begin{aligned} \text{1's complement} &= (2^4 - 1) - 5 \\ &= (16 - 1) - 5 \\ &= 10_{10} \rightarrow 1010_2 \end{aligned}$$

- . $-5_{10} = 1010_2$

Example

- Example : Convert -120 to a 8-bit 1's complement representation

$$\begin{aligned} \text{1's complement} &= (2^8 - 1) - 120 \\ &= 256 - 1 - 120 \\ &= 135_{10} \rightarrow 1000\ 0111_2 \end{aligned}$$

- Let's look again to simplify 1's complement representation.

For 4-bits

5 \rightarrow 0101

-5 \rightarrow 1010

For 8-bits

120 \rightarrow 01111000

-120 \rightarrow 10000111

1's Complement Table

| Binary Number | 1's Complement |
|---------------|----------------|
| 0000 | 1111 |
| 0001 | 1110 |
| 0010 | 1101 |
| 0011 | 1100 |
| 0100 | 1011 |
| 0101 | 1010 |
| 0110 | 1001 |
| 0111 | 1000 |
| 1000 | 0111 |
| 1001 | 0110 |
| 1010 | 0101 |
| 1011 | 0100 |
| 1100 | 0011 |
| 1101 | 0010 |
| 1110 | 0001 |
| 1111 | 0000 |

2's Complement Representation

- ❑ Two's **complement** is a mathematical operation to reversibly convert a positive binary number into a negative binary number with equivalent (but negative) value, using the binary digit with the greatest place value to indicate whether the binary number is positive or negative (the sign).
- ❑ There are 3 methods to calculate Two's **complement** of the number.
- ❑ **First Method** : For n-bit number N, its 2's complement is $(2^n - N)$.

Example 1: Convert -5_{10} to 4-bit 2's complement

$$\begin{aligned} \text{2's complement} &= 2^4 - 5 \\ &= 16 - 5 \\ &= 11_{10} \rightarrow 1011_2 \end{aligned}$$

$$-5_{10} = 1011_2$$

Example 2: Convert -120_{10} to 8-bit 2's complement representation

$$\begin{aligned} \text{2's complement} &= 2^8 - 120 \\ &= 256 - 120 \\ &= 136 \rightarrow 1000\ 1000_2 \end{aligned}$$

$$-120_{10} = 10001000_2$$

Cont..

• Second Method

- Another method to calculate 2's complement
 - ▣ Convert number to 1's complement
 - ▣ Then, add 1 to that number

- **Example :** Convert -120_{10} to 8-bit 2's complement representation

$$\begin{aligned}120_{10} &= 01111000 \\ \text{1's complement} &\rightarrow 10000111 \text{ (invert bits)} \\ \text{2's complement} &\rightarrow 10000111 + 1 = 10001000_2 \\ -120_{10} &= 10001000_2\end{aligned}$$

Third Method

- Another method to calculate 2's complement
 - ▣ Keep same bit from LSB \rightarrow MSB until found "1"
 - ▣ Do 1's complement on the rest bits.

- **Example :** Convert -120_{10} to 8-bit 2's complement representation

$$\begin{aligned}120_{10} &= \overleftarrow{0111}1000 \\ &= 10001000\end{aligned}$$

Example

- Examples: Convert the following binary numbers to their 2'S Complements

a. 101

b. 10011

Solution

a. 010 → 1'S Complement of the binary number 101

+1 → adding 1

011 → 2'S Complement of the number 101

b. 01100 → 1'S Complement of the binary number 10011

+1 → adding 1

01101 → 2'S Complement of the number 10011

Homework

1. Determine the decimal equivalent of the following signed numbers: -

a. $(10010001)_2$

b. $(00110111)_2$

Answer

a. $(-17)_{10}$

b. $(+55)_{10}$

2. Express, the following signed decimal numbers in 1's complement form using byte representation: -

a. $(-22)_{10}$

b. $(-11)_{10}$

Answer

a. $(11101001)_2$

b. $(11110100)_2$

3. Express, the signed numbers shown in question 2 above in 2's complement form.

Answer

a. $(11101010)_2$

b. $(11110101)_2$

Binary Subtraction Using 2's Complement

- subtraction of a smaller number from a larger number using 2's complement subtraction, the following steps are to be followed:
 - Step 1: Determine the 2's complement of the smaller number
 - Step 2: Add this to the larger number.
 - Step 3: Omit the carry. Note that there is always a carry in this case.
- Example: Subtract $(1010)_2$ from $(1111)_2$ using 2's complement method.

- **Step 1:** 2's complement of $(1010)_2$ is $(0110)_2$.

- **Step 2:** Add $(0110)_2$ to $(1111)_2$.

$$\begin{array}{r} 1111 \\ + 0110 \\ \hline 10101 \end{array}$$

Omit this carry → 1 0 1 0 1

↓

0 1 0 1

Binary Subtraction Using 2's Complement

- ❑ To subtract a larger number from a smaller number using 2's complement subtraction, the following steps are to be followed:
 - ❑ Step 1: Determine the 2's complement of the smaller number.
 - ❑ Step 2: Add this to the larger number.
 - ❑ Step 3: There is no carry in this case. The result is in 2's complement form and is negative.
 - ❑ Step 4: To get an answer in true form, take 2's complement and change its sign.

Example

□ $10110 - 11010$

□ Solution:

□ 2's complement of 11010 is $(00101 + 1)$ i.e. 00110. Hence

□ Minued - $1\ 0\ 1\ 1\ 0$

2's complement of subtrahend - $0\ 0\ 1\ 1\ 0$

Result of addition - $1\ 1\ 1\ 0\ 0$

□ As there is no carry over, the result of subtraction is negative and is obtained by writing the 2's complement of 11100 i.e. $(00011 + 1)$ or 00100.

□ Hence the difference is -100 .

Example

110110 - 10110

Solution:

The numbers of bits in the subtrahend is 5 while that of minuend is 6. We make the number of bits in the subtrahend equal to that of minuend by taking a '0' in the sixth place of the subtrahend.

Now, 2's complement of 010110 is $(101101 + 1)$ i.e. 101010. Adding this with the minuend.

1 1 0 1 1 0 Minuend

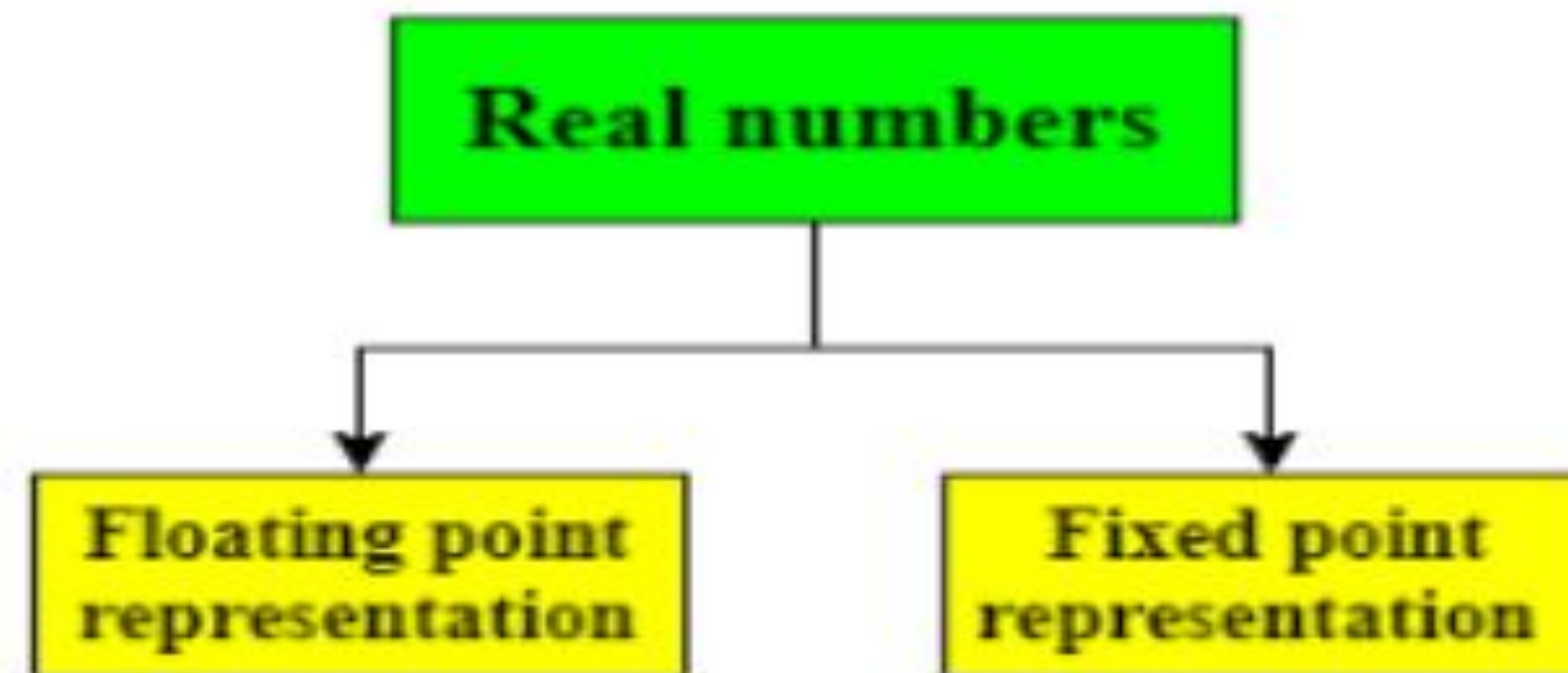
1 0 1 0 1 0 2's complement of subtrahend

Carry over 1 1 0 0 0 0 0 Result of addition

After dropping the carry over we get the result of subtraction to be 100000.

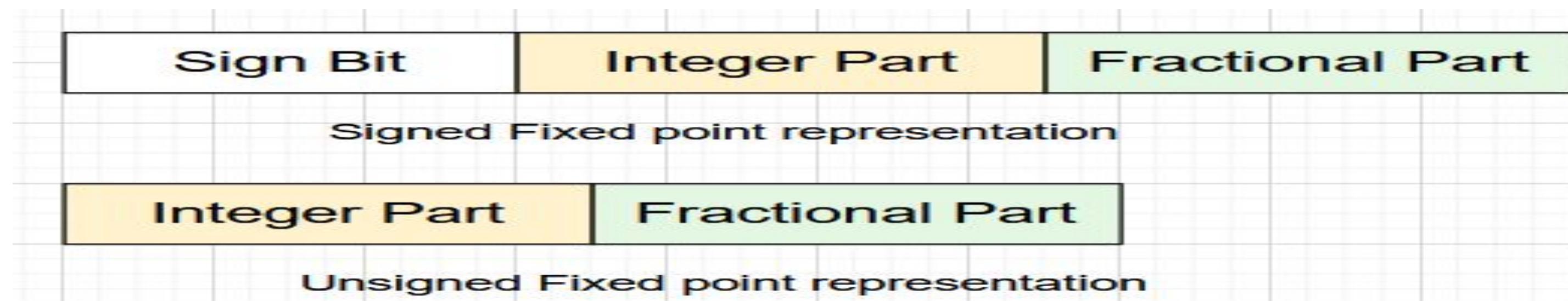
Fixed and Floating-Point Representation

- ❑ In digital technology, data is stored in memory registers with binary bits 0's and 1's because the computer only understands binary language. When we enter data in the system, it is converted into binary bits, and it is processed and used in the CPU in different ways.
- There two types of approaches that are developed to store real numbers with the proper method.
- Fixed point number
- Floating point number



Fixed Point Representation

- ❑ In computers, fixed-point representation is a real data type for numbers. Fixed point representation can convert data into binary form, and then the data is processed, stored, and used by the computer. It has a fixed number of bits for the integral and fractional parts.
- ❑ There are three parts of the fixed-point number representation: **Sign bit, Integer part, and Fractional part**. The below figure depicts it.



- **Sign bit:-** The fixed-point number representation in binary uses a sign bit. The negative number has a sign bit 1, while a positive number has a bit 0.
- **Integral Part:-** The integral part in fixed-point numbers is of different lengths at different places. It depends on the register's size; for an 8-bit register, the integral part is 4 bits.
- **Fractional part:-** The Fractional part is of different lengths at different places. It depends on the registers; for an 8-bit register, the fractional part is 3 bits.

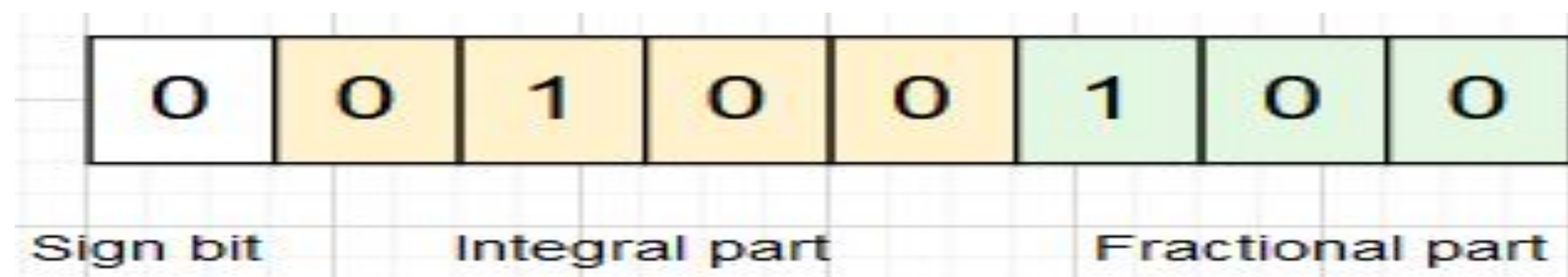
Cont..

- Size of Sign Bit, Integer Part, and Fractional Part for different registers are displayed below

| Register | Sign Bit | Integer Part | Fraction Part |
|-----------------|----------|--------------|---------------|
| 8-bit register | 1 bit | 4 bits | 3 bits |
| 16-bit register | 1 bit | 9 bits | 6 bits |
| 32-bit register | 1 bit | 15 bits | 9 bits |

- Example The number **considered is 4.5**

- Step 1:** We will convert the number 4.5 to binary form. **$4.5 = 100.1$**
- Step 2:** Represent the binary number in fixed-point notation with the following format.



Floating-Point Representation

- ❑ The computer uses floating-point number representation to convert the input data into binary form. This binary form number is converted into scientific notation, which is converted into floating-point representation.
- ❑ The floating-point representation has two types of notation:
 - ❑ **Scientific notation**
 - ❑ **Normalization notation**

Scientific notation

- ❑ Scientific notation is further converted into floating-point notation because floating-point notation only accepts scientific notation.
- ❑ Scientific notation is the method of representing binary numbers into **$M \times B^e$** form.
- ❑ Where M is Mantissa, B is Base and e is Exponent
 - For example:-
 - Number = 376.423(its not scientific notation)Number in scientific = 36.4423×10^1 or 3.64423×10^2

| Number | Mantissa | Base | Exponent |
|------------------|----------|------|----------|
| 9×10^8 | 9 | 10 | 8 |
| 110×2^7 | 110 | 2 | 7 |
| 4367.784 | 4367784 | 10 | -3 |

It is further converted into floating-point representation. For example,

Number = 32625

Number in Scientific Notation = 32.625×10^3

Number in binary form = 1101.101×2^{101}

Here, Mantissa is **1101.101** and Base part is 2^{101} .

Normalization notation

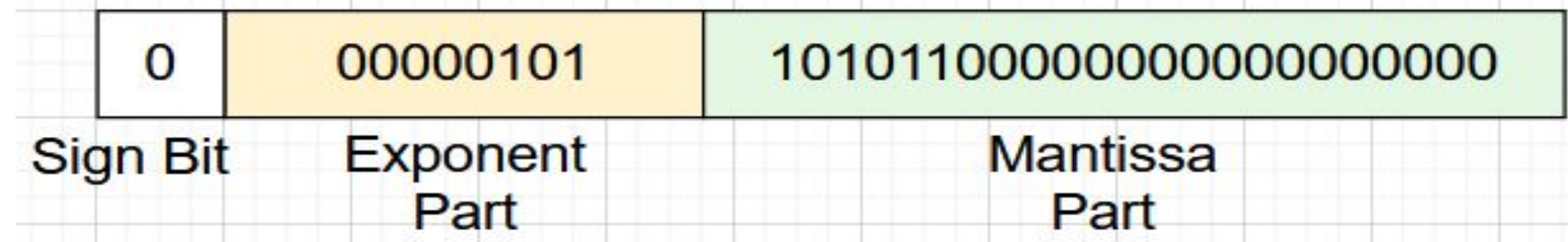
- **Normalization notation:** It is a special case of scientific notation. Normalized means that we have at least one non-zero digit after the decimal point.
- A floating-point representation has three parts: **Sign bit**, **Exponent Part**, and **Mantissa**. We can see the below diagram to understand these parts



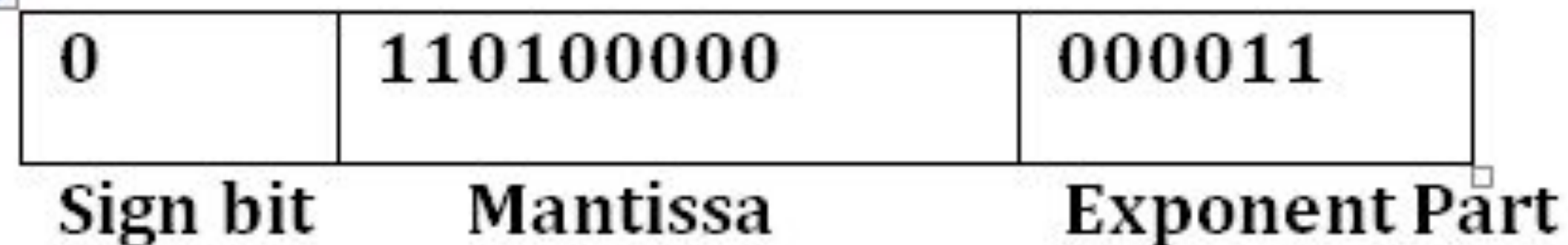
- **Sign bit:-** The floating-point numbers in binary uses a sign bit. A negative number has a sign bit 1, while a positive number has a sign bit 0. The sign of any number depends on mantissa, not on exponent.
- **Mantissa Part:-** The mantissa part is of different lengths at different places. It depends on registers like for a 16-bit register, and mantissa part is of 8 bits.
- **Exponent Part:-** It is the power of the number. It depends on the size of the register. For example, in the 16-bit register, the exponent part is of 7 bits.

Example

- The number considered is **53.5**
- **Step 1:** We will convert the number 53.5 to binary form. **$53.5 = 110101.1$**
- **Step 2:** Normalize the number (base is 2) = $(1.101011) * 2^5$.
- **Step 2:** Represent the binary number in floating-point notation with the following format.



- **Example 2 :** Convert the floating point binary number into decimal number 01101000000000011 Assume 9 bit mantissa and 6 bit exponent



Example

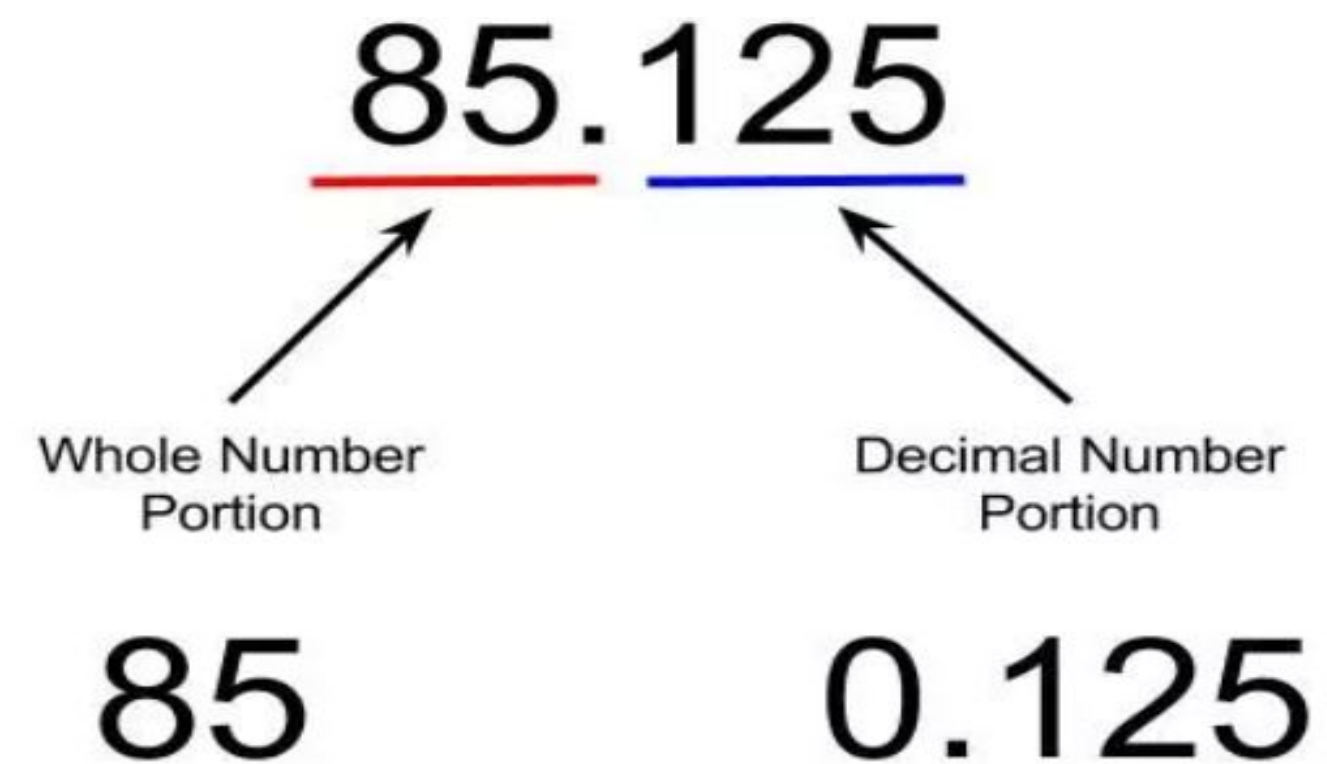
- **Example how to convert number from decimal to floating point representation Number is 85.125**
- **Follow the following steps:**
- **Step 1 Choose single or double precision.**
- **Step 2: Separate the whole and the decimal part of the number.**
- **Step 3: Convert the whole number into binary**
- **Step 4: Convert the binary number into base 2 scientific notation**
- **Step 5: Determine the sign of the number and display in binary format**
- **Step 6: Get the exponent based on precision**
- **Step 7: Turn the exponent into binary. After you determine your final exponent**
- **Step 8: Determine the mantissa**
- **Step 9 Compile 3 parts into one final number**

Step 1 Choose single or double precision.

- When writing a number in single or double precision, the steps to a successful conversion will be the same for both, the only change occurs when converting the exponent and mantissa.
- First we must understand what single precision means. In floating point representation, each number (0 or 1) is considered a “bit”. Therefore single precision has 32 bits total that are divided into 3 different subjects. These subjects consist of a sign (1 bit), an exponent (8 bits), and a mantissa or fraction (23 bits).
- Double precision, on the other hand, has the same setup and same 3 parts as single precision; the only difference is that it will be larger and more precise number. In this case, the sign will have 1 bit, the exponent will have 11 bits and the mantissa will have 52 bits.
- In this example will convert the number 85.125 into IEEE 754 single precision.

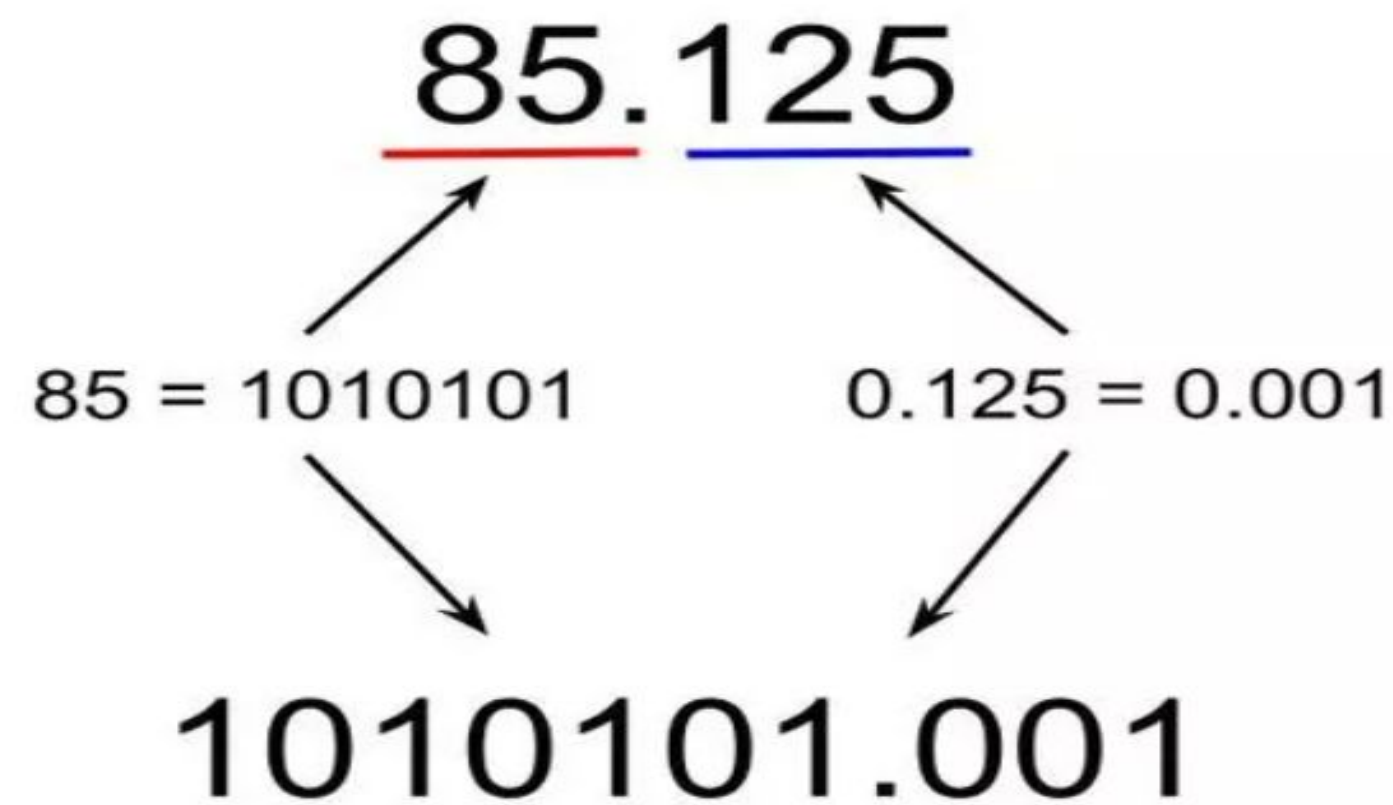
Step 2: Separate the whole and the decimal part of the number

- **Step 2: Separate the whole and the decimal part of the number.** Take the number that you would like to convert, and take apart the number so you have a whole number portion and a decimal number portion. This example will use the number 85.125. You can separate that into whole number 85, and the decimal 0.125.



Step 3: Convert the whole number into binary.

- **Step 3: Convert the whole number into binary.** This would be the 85 from 85.125, which will be 1010101 when converted into binary.
- Convert the decimal portion into binary. This would be the 0.125 from 85.125, which will be 0.001 when converted into binary



Step 4: Convert the binary number into base 2 scientific notation

- **Step 4: Convert the binary number into base 2 scientific notation.** You can convert the number into base 2 scientific notation by moving the decimal point over to the left until it is to the right of the first bit. These numbers are normalized which means the leading bit will always be 1. As for the exponent, the number of times that you moved the decimal will be your exponent in base 2 scientific notation.
- Remember that moving the decimal to the left will result in a positive exponent while moving the decimal to the right will result in a negative exponent.
- For our example, you will need to move the decimal 6 times in order to get it to the right of the first bit. The resulting notation will be $01.010101001 * 2^6$ this number will be used in future steps.

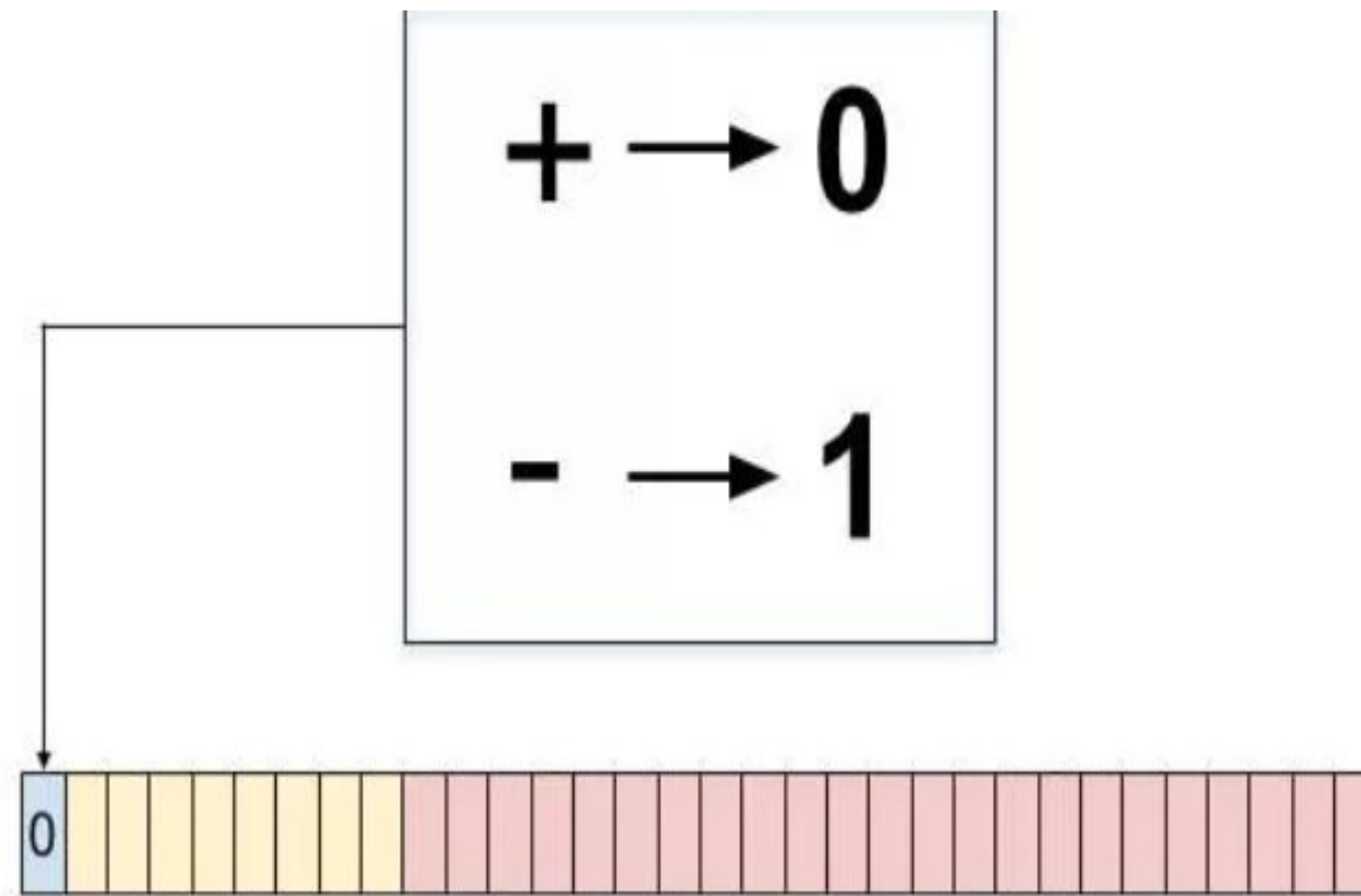
1010101.001

Move decimal 6 places

1.010101001 x 2⁶

Step 5: Determine the sign of the number and display in binary format

- **Step 5: Determine the sign of the number and display in binary format.** You will now determine if your original number is positive or negative. If the number is positive, you will record that bit as 0, and if it is negative, you will record that bit as 1. Since your original number, 85.125, is positive, you will record that bit as 0. This will be the first bit out of the 32 total bits in your IEEE 754 single precision representation.



Step 6: Get the exponent based on precision

- Step 6: **Get the exponent based on precision.** There are set biases for both single and double precision. The exponent bias for single precision is 127, which means we must add the base 2 exponent found previously to it. Thus, the exponent you will use is $127+6$ which is 133.
- Double precision as perceived from the name is more precise and can hold larger numbers. Therefore its exponent bias is 1023. The same steps used for single precision apply here, so the exponent you can use to find double precision is 1029.

$$1.010101001 \times 2^6$$


$$127 + 6 = 133$$

Step 7: Turn the exponent into binary. After you determine your final exponent

- Step 7: Turn the exponent into binary. After you determine your final exponent, you will need to convert it into binary so that it could be used in the IEEE 754 conversion. For the example, you can convert the 133 that you found in the last step into 10000101.

| Whole Number Division | Result | Remainder |
|-----------------------|--------|-----------|
| 133/2 | 66 | 1 |
| 66/2 | 33 | 0 |
| 33/2 | 16 | 1 |
| 16/2 | 8 | 0 |
| 8/2 | 4 | 0 |
| 4/2 | 2 | 0 |
| 2/2 | 1 | 0 |
| 1/2 | 0 | 1 |

133 = 10000101
Exponent



Step 8: Determine the mantissa.

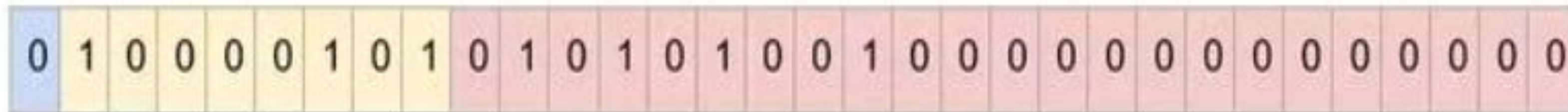
- Step 8: **Determine the mantissa.** The mantissa aspect, or the third part of the IEEE 754 conversion, is the rest of the number after the decimal of the base 2 scientific notation. You will just drop the 1 in the front and copy the decimal portion of the number that is being multiplied by 2. No binary conversion needed! For the example, the mantissa would be

010101001 from $01.010101001 * 2^6$.

- **Step 9 Compile 3 parts into one final number.**
- Finally, you will compile all that we have calculated thus far into your conversion. It will first begin with a 0 or 1 bit that you determined in step 7 based on sign. For the example, you will have a 0 to start it.
- Next up, you will have the exponent section that you determined in step 9. For the example, your exponent will be 10000101.
- Now, you have the mantissa, which is the third and last part of the conversion. You derived this earlier when you took the decimal portion of the base 2 conversion. For the example, the mantissa would be 010101001.

- Finally, you combine these all together. The order should go sign-exponent-mantissa. After you connect those three binary numbers, you then fill out the rest of the mantissa with 0s.
- For the example the solution is **0 10000101 0101010010000000000000** as 85.125 converted into IEEE 754 format

85.125



Example 2

- Example 2 : how to represent given decimal number 263.3 into 32 bit floating point notation

263.3

2 | 263
2 | 131
2 | 65
2 | 32
2 | 16
2 | 8
2 | 4
2 | 2
2 | 1
2 | 0

1
1
1
1
0
0
0
0
0
1

263 : 100000111
0.3 : 01001100110011...

I] 263.3
→ 1.000001110100110011...

Scientific notation:
1.000001110100110011... × 2⁸

II] mantissa

III] $\underbrace{x}_{1 \text{ sign bit}}$, $\underbrace{xxxxxxx}_{8 \text{ exp bits}}$, $\underbrace{xxxxx...xx}_{23 \text{ fraction bits}}$

0 127 + 8 = 135
10000111

exp bias: 127

| | | |
|---------|-----|---|
| 0.3 × 2 | 0.6 | 0 |
| 0.6 × 2 | 1.2 | 1 |
| 0.2 × 2 | 0.4 | 0 |
| 0.4 × 2 | 0.8 | 0 |
| 0.8 × 2 | 1.6 | 1 |
| 0.6 × 2 | 1.2 | 1 |

0
0
1
1
0
0
1
1
0
0
1
1
0
0
1
1
0

0 1 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1 0 1 0 0 1 1 0 0 1 1 0 0 1 1 0

SPPU QP April 2023

- Q1)** a) Draw a block diagram of basic components of a computer system. Explain each component in detail. [10]
- b) With an example, write the steps to subtract a large number from a smaller number using 2's complement method. [5]

OR

- Q2)** a) Explain the instruction cycle in detail. [7]
- b) Difference between microprocessor and microcontroller. [8]