Parul® University

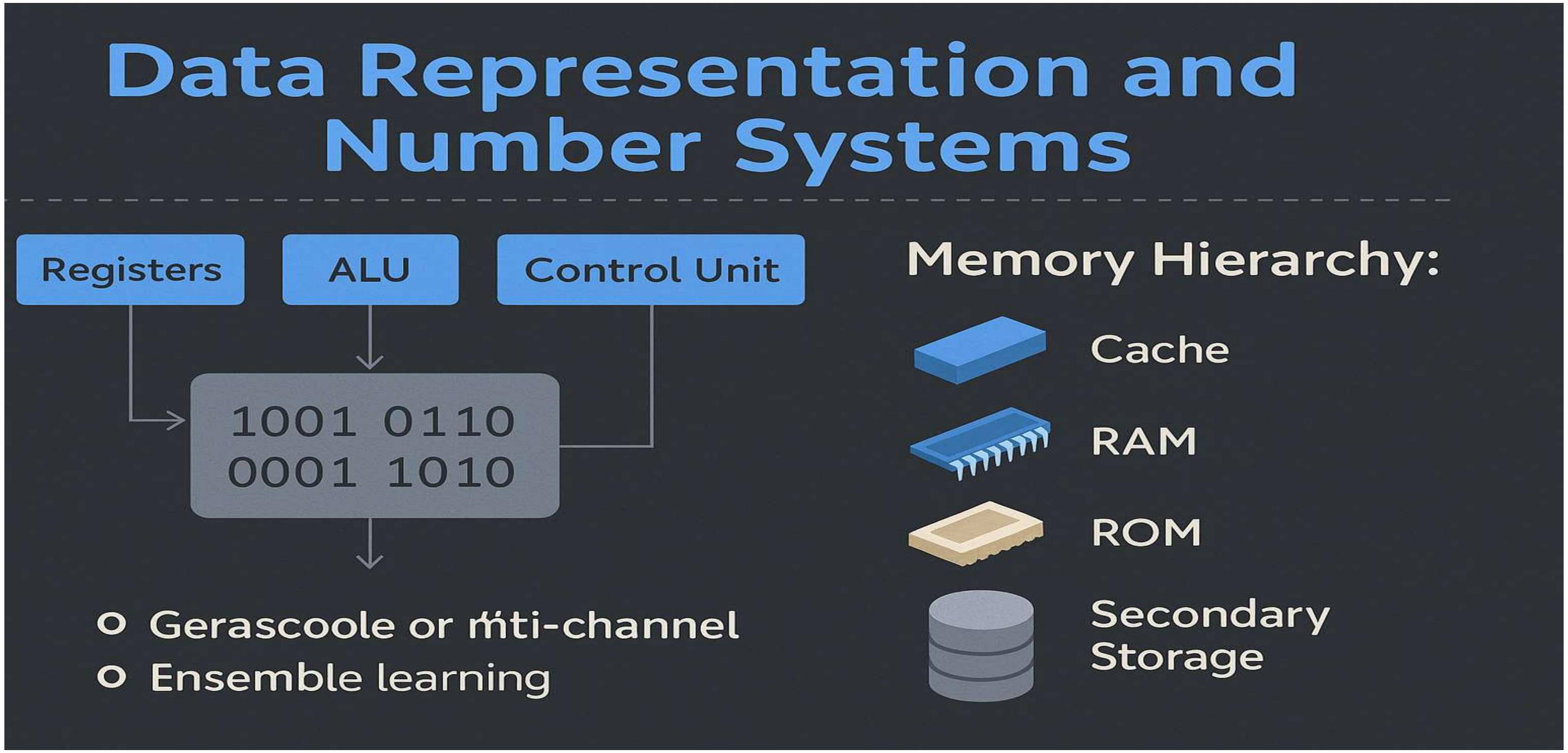
# Subject: Fundamental of Computer Science

## Unit-2 NUMBER SYSTEM

Parul® University

DIGITAL LEARNING CONTENT

# Numbers System




Understanding How Computers Store and Process Data

# 1. Introduction To Numbers System

- Data representation refers to the way information—such as numbers, text, images, and audio—is encoded and stored in a computer system. All forms of data inside a computer are ultimately represented as sequences of binary digits (bits): 0s and 1s. This binary format is fundamental because digital circuits can easily distinguish between two states:

  on (1) and off (0).

- **Bit**: The smallest unit of data, representing a binary value (0 or 1).
- **Byte:** A group of 8 bits, commonly used as the basic addressable element in memory.

# 1. Numbers Representation

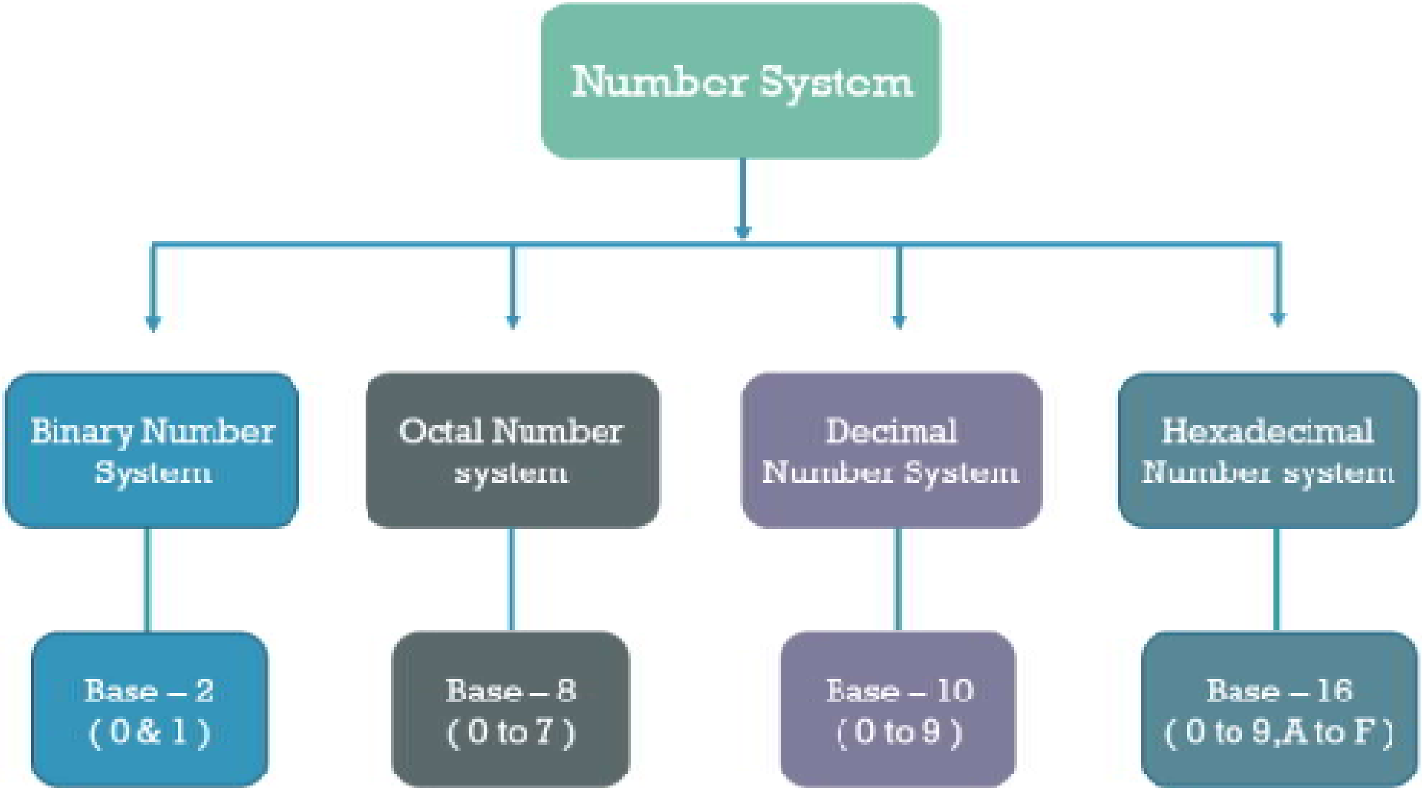
| Decimal System | Binary System | Octal System | Hexadecimal System |
|---|---|---|---|
| 0 | 00000000 | 000 | 0 |
| 1 | 00000001 | 001 | 1 |
| 2 | 00000010 | 002 | 2 |
| 3 | 00000011 | 003 | 3 |
| 4 | 00000100 | 004 | 4 |
| 5 | 00000101 | 005 | 5 |
| 6 | 00000110 | 006 | 6 |
| 7 | 00000111 | 007 | 7 |
| 8 | 00001000 | 010 | 8 |
| 9 | 00001001 | 011 | 9 |
| 10 | 00001010 | 012 | A |
| 11 | 00001011 | 013 | B |
| 12 | 00001100 | 014 | C |
| 13 | 00001101 | 015 | D |
| 14 | 00001110 | 016 | E |
| 15 | 00001111 | 017 | F |

# 1. Why Are Number Systems Important?

- Number systems provide the foundation for representing and processing all kinds of data in computers. Different systems are used for various purposes, such as simplifying binary data, performing calculations, or displaying information to users.

- Provide structure to how data is stored and processed
- Used for calculations, debugging, memory addressing, etc.

# 1. Numbers with Their Base



# 1. Types of Number Systems

| Number System | Base | Digits Used | Typical Use Cases |
|---|---|---|---|
| Binary | 2 | 0, 1 | Core data processing, memory, logic circuits |
| Decimal | 10 | 0–9 | Human interfaces, input/output |
| Octal | 8 | 0–7 | Shorthand for binary, digital electronics |
| Hexadecimal | 16 | 0–9, A–F | Memory addressing, color codes, debugging |

# 1. Binary Number System (Base-2)

## Binary Number System (Base-2)

- Uses only two digits: 0 and 1.
- Each digit's place value is a power of 2 (e.g., $2^0$, $2^1$, $2^2$).
- Every piece of data in a computer—numbers, letters, images—is ultimately stored as binary.
- The Binary system is the foundation for data representation in computers and digital electronics.
- Example: Decimal Number '5' is represented as 101 in binary.

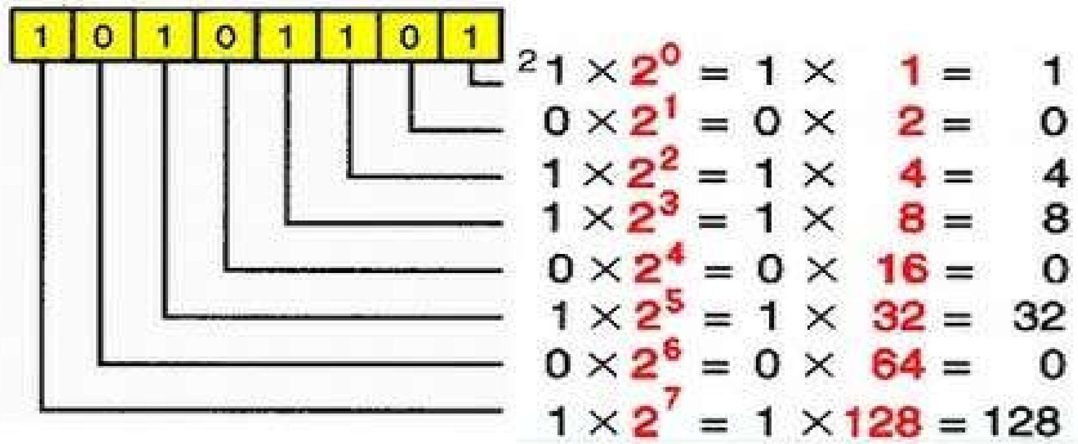

# 1. Decimal Number System (Base-10)

## Decimal Number System (Base-10)

- Uses ten digits: 0,1,2,3,4,5,6,7,8 and 9.
- The standard system for human calculations.
- Each digit’s place value is a power of 10 (e.g., $10^0$, $10^1$, $10^2$).
- Example: The number 254 in decimal is what we use daily.

# 1. Decimal Number System (Base-10)

## Binary System

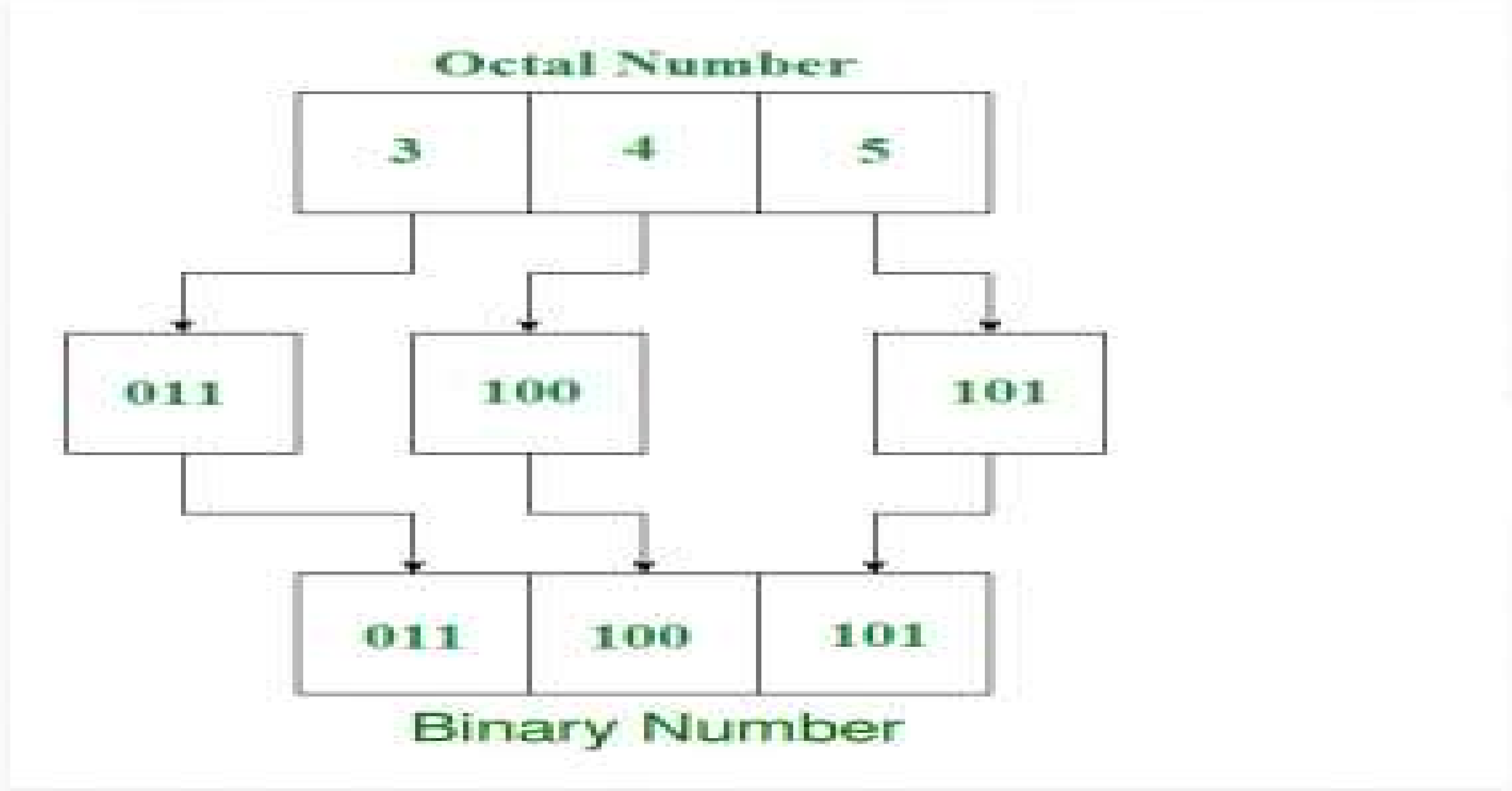


# 2. Octal Number System (Base-8)

## Octal Number System (Base-8)

- Uses eight digits: 0,1,2,3,4,5,6 and 7.
- Used as a compact representation of binary numbers in some programming and digital systems.
- Example: Binary 011100101 is 345 in octal.

# 2. Octal Number System (Base-8)



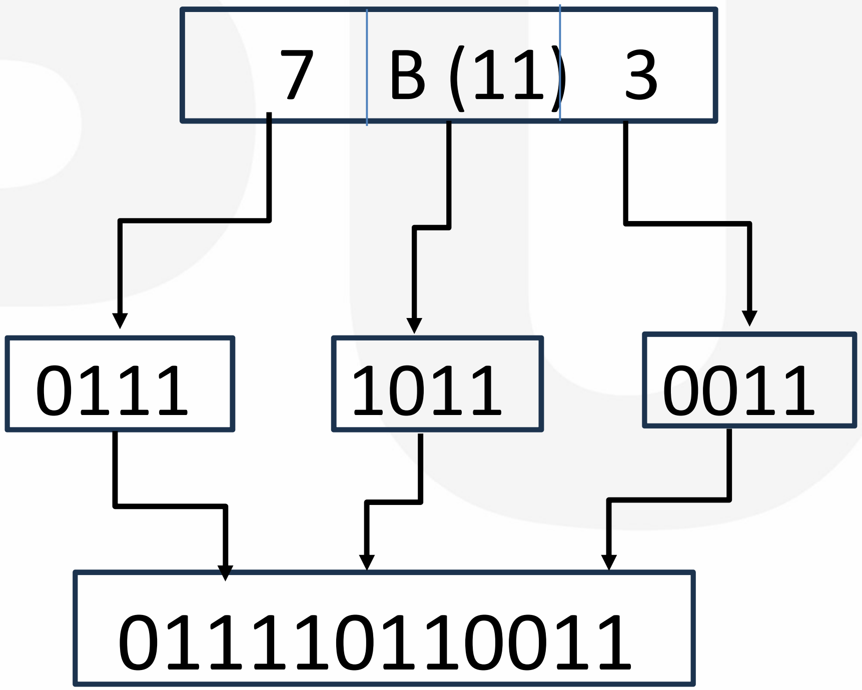
# 2. Hexadecimal Number System(Base-16)

## Hexadecimal Number System (Base-16)

- Uses sixteen symbols: 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E and F where (A=10, B=11, ..., F=15).
- Makes binary data more readable for humans, commonly used in programming, memory addresses, and color codes.
- Each digit's place value is a power of 16 (e.g., $16^0$, $16^1$, $16^2$).
- Hexadecimal simplifies binary by representing every 4 bits as one digit (0-F).
- Example: Decimal 255 is FF in hexadecimal.

# 2. Hexadecimal Number System(Base-16)

**Hexadecimal Number System (Base-16)**




- Binary Equivalent to 7B3 is 011110110011

# 2. Conversion from Decimal to Binary Number

**For Integer Part:**

- Divide the decimal number by 2.
- Record the remainder (0 or 1).
- Continue dividing the quotient by 2 until the quotient is 0.
- The binary equivalent is the remainders read from bottom to top.

**For Fractional Part:**

- Multiply the fractional part by 2.
- Record the integer part (0 or 1).
- Take the fractional part of the result and repeat the multiplication.
- Continue until the fractional part becomes 0 or reaches the desired precision.
- The binary equivalent is the integer parts recorded in sequence.

# 2. Conversion from Decimal to Binary Number

**Example:** $(10.25)_{10} = ( \ ? \ )_2$

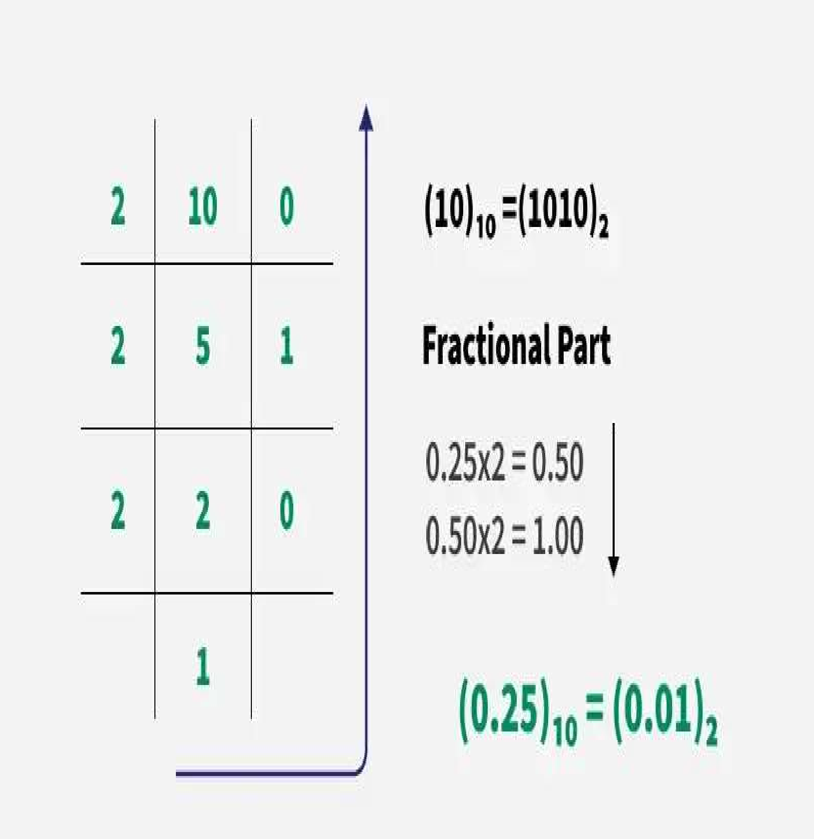
**For Integer Part (10):**

- Divide 10 by 2 → Quotient = 5, Remainder = 0
- Divide 5 by 2 → Quotient = 2, Remainder = 1
- Divide 2 by 2 → Quotient = 1, Remainder = 0
- Divide 1 by 2 → Quotient = 0, Remainder = 1
- Reading the remainders from bottom to top gives 1010.

**For Fractional Part (0.25):**

- Multiply 0.25 by 2 → Result = 0.5, Integer part = 0
- Multiply 0.5 by 2 → Result = 1.0, Integer part = 1
- The fractional part ends here as the result is now 0. Reading from top to bottom gives 01.

Thus, the binary equivalent of $(10.25)_{10}$ is $(1010.01)_2$.



# 2. Binary to Decimal Number System Conversion

**For Integer Part:**

- Write down the binary number.
- Multiply each digit by 2 raised to the power of its position, starting from 0 (rightmost digit).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the binary integer.

**For Fractional Part:**

- Write down the binary fraction.
- Multiply each digit by 2 raised to the negative power of its position, starting from -1 (first digit after the decimal point).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the binary fraction.

## 2. Binary to Decimal Number System Conversion

**Example:** $(1010.01)_2 = (\ ?\ )_{10}$

$1x2^3 + 0x2^2 + 1x2^1 + 0x2^0 + 0x2^{-1} + 1x2^{-2}$

$= 1x8 + 0x4 + 1x2 + 0x1 + 0x.50 + 1x.25$

$= 8+0+2+0+0+0.25 = 10.25$

Thus, $(1010.01)_2 = (10.25)_{10}$

Parul® University

DIGITAL LEARNING CONTENT

# 2. Decimal to Octal Conversion

**For Integer Part:**

- Divide the decimal number by 8.
- Record the remainder (0 to 7).
- Continue dividing the quotient by 8 until the quotient is 0.
- The octal equivalent is the remainders read from bottom to top.

**For Fractional Part:**

- Multiply the fractional part by 8.
- Record the integer part (0 to 7).
- Take the fractional part of the result and repeat the multiplication.
- Continue until the fractional part becomes 0 or reaches the desired precision.
- The octal equivalent is the integer parts recorded in sequence.

## 2. Decimal to Octal Conversion

**Example:** $(20.25)_{10}$ = ( ? $)_8$

**For Integer Part (10):**

Divide 20 by 8 → Quotient = 2, Remainder = 4

Divide 2 by 8 → Quotient = 0, Remainder = 2

Octal equivalent = 24 (write the remainder, read from bottom to top).

So, the octal equivalent of the integer part 20 is 24.

**For Fractional Part (0.25):**

Multiply 0.25 by 8 → Result = 2.0, Integer part = 2

The fractional part ends here as the result is now 0. So, the octal equivalent of the fractional part 0.25 is 0.2.

The octal equivalent of $(20.25)_{10} = (24.2)_8$

# 3. Octal to Decimal Number Conversion

**For Integer Part:**

- Write down the octal number.
- Multiply each digit by 8 raised to the power of its position, starting from 0 (rightmost digit).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the octal integer.

**For Fractional Part:**

- Write down the octal fraction.
- Multiply each digit by 8 raised to the negative power of its position, starting from -1 (first digit after the decimal point).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the octal fraction.

# 3. Octal to Decimal Number Conversion

- **Example:** $(24.2)_8 = (\ ?\ )_{10}$

$2 \times 8^1 + 4 \times 8^0 + 2 \times 8^{-1}$

$= 2x8 + 4x1 + 2/8$

$= 16 + 4 + 0.25$

$= 20.25$

Thus, $(24.2)_8 = (20.25)_{10}$

# 3. Decimal to Hexadecimal Conversion

**For Integer Part:**

- Divide the decimal number by 16.
- Record the remainder (0-9 or A-F).
- Continue dividing the quotient by 16 until the quotient is 0.
- The hexadecimal equivalent is the remainders read from bottom to top.

**For Fractional Part:**

- Multiply the fractional part by 16.
- Record the integer part (0-9 or A-F).
- Take the fractional part of the result and repeat the multiplication.
- Continue until the fractional part becomes 0 or reaches the desired precision.
- The hexadecimal equivalent is the integer parts recorded in sequence.

# 3. Decimal to Hexadecimal Conversion

**Example:** $(14.25)_{10}$

**Integer part:**

14 ÷ 16 = 0, Remainder = E (14 in decimal is E in hexadecimal)

Hexadecimal equivalent = E

**Fractional part:**

0.25 × 16 = 4, Integer part = 4

Hexadecimal equivalent = 0.4

Thus, $(14.25)_{10} = (E.4)_{16}$

# 3. Hexadecimal to Decimal Conversion

**For Integer Part:**

- Write down the hexadecimal number.
- Multiply each digit by 16 raised to the power of its position, starting from 0 (rightmost digit).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the hexadecimal integer.

**For Fractional Part:**

- Write down the hexadecimal fraction.
- Multiply each digit by 16 raised to the negative power of its position, starting from -1 (first digit after the decimal point).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the hexadecimal fraction.

# 3. Hexadecimal to Decimal Conversion

**For Integer Part:**

- Write down the hexadecimal number.
- Multiply each digit by 16 raised to the power of its position, starting from 0 (rightmost digit).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the hexadecimal integer.

**For Fractional Part:**

- Write down the hexadecimal fraction.
- Multiply each digit by 16 raised to the negative power of its position, starting from -1 (first digit after the decimal point).
- Add up the results of these multiplications.
- The sum is the decimal equivalent of the hexadecimal fraction.

# 3. Hexadecimal to Decimal Conversion

Example: $(E.4)_{16} = (\quad ? \quad)_{10}$

$(E \times 16^{0}) + (4 \times 16^{-1})$

$= (14 \times 1) + (4 \times 0.0625)$

Thus, $(E.4)_{16} = (14.25)_{10}$

# 3. Hexadecimal to Binary Conversion

To convert from Hexadecimal to Binary:

Each hexadecimal digit (0-9 and A-F) is represented by a 4-bit binary number.

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

# 3. Hexadecimal to Binary Conversion

**Example:** $(5B)_{16} = (\ \ ?\ \ )_2$

$(5)_{16} = (0101)_2$

$(B)_{16} = (1011)_2$

Thus, $(5B)_{16} = (01011011)_2$

# 3. Binary to Hexadecimal Conversion

- Start from the rightmost bit and divide the binary number into group of 4 bits each.
- If the number of bits isn't a multiple of 4, pad the leftmost group with leading zeros.
- Each 4-bit binary group corresponds to a single hexadecimal digit.
- Replace each 4-bit binary group with the
- corresponding hexadecimal digit.

**Example:** $(1111011011)_2$

<u>0011</u> <u>1101</u> <u>1011</u>

| | |

3 D B

Thus, $(001111011011)_2 = (3DB)_{16}$

# 3. Octal to Binary Number Conversion

To convert from octal to binary:

- Each octal digit (0-7) corresponds to a 3-bit binary number.
- For each octal digit, replace it with its corresponding 3-bit binary equivalent.

**Example:** $(153)_8$

Break the octal number into digits: 1, 5, 3

Convert each digit to binary:

1 in octal = 001 in binary

5 in octal = 101 in binary

3 in octal = 011 in binary

Thus, $(153)_8 = (001101011)_2$

# 3. Binary to Octal Number Conversion

To convert from binary to octal:

- Starting from the rightmost bit, divide the binary number into groups of 3 bits.
- If the number of bits is not a multiple of 3, add leading zeros to the leftmost group.
- Each 3-bit binary group corresponds to a single octal digit.

Parul® University

DIGITAL LEARNING CONTENT

# 3. Binary to Octal Number Conversion

- The binary-to-octal conversion
for each 3-bit group is as follows:

| Octal Number | Binary Number |
|---|---|
| 0 | 000 |
| 1 | 0001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

**Example:** $(111101101)_2$

111 101 101

| | |

7 5 5

Thus, $(111101101)_2 = (755)_8$

# 4. Arithmetic Operation on Binary Number

**Binary addition**

An important rule to be kept in mind is that just like real numbers, binary arithmetic calculations begin from the right side. For addition, we have four simple rules to remember:

0 + 0 = 0 ,

0 + 1 = 1 ,

1 + 0 = 1 , and

1 + 1 = 0 (with a carry to the adjacent left bit)

# 4. Arithmetic Operation on Binary Number

## Binary addition

| A | B | A+B | Carry |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |

# 4. Arithmetic Operation on Binary Number

## Binary addition

**Example 24+15**

```
         1
24 =    1 1 0 0 0
15 =      1 1 1 1
-----------------
39 = 1 0 0 1 1 1
```

| A | B | A+B | Carry |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |

# 4. Arithmetic Operation on Binary Number

## Binary Subtraction

**Example 29 - 11**

```
            0 10
29 =   1 1 1 0 1
11 =     1 0 1 1
18 =   1 0 0 1 0
```

| A | B | A - B | Borrow |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 from previous an become 10 | 1 | 10-1= 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

# 4. Arithmetic Operation on Binary Number

## Binary Multiplication

Example 18 x 5

18 = 1 0 0 1 0

5 = 1 0 1

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

90 1 0 1 1 0 1 0

| A | B | A x B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

# 4. Arithmetic Operation on Binary Number

## Binary Division

**Example 17 / 2**

| | | | |
|---|---|---|---|
| | | 1 0 0 0 | 8 |
| 2 | 10 | 1 0 0 0 1 | 17 |
| | | 1 0 | |
| | | 0 0 0 1 | |

# 5. Arithmetic Operation on Octal Number

**Octal addition**

**Steps:**

- **Convert the Octal number to Decimal number and add them.**
- **Convert the resultant decimal number to octal number**

**Example $(24)_8 + (15)_8$**

Full Calculation:

$(24)_8 = (20)_{10}$

$(15)_8 = (13)_{10}$

$(20)_{10} + (13)_{10} = 33_{10} = \mathbf{41_8}$

**Octal Subtraction**

**Steps:**

- **Convert the Octal number to Decimal number and subtract them.**
- **Convert the resultant decimal number to octal number**

**Example $(37)_8 + (15)_8$**

Full Calculation:

$(37)_8 = (31)_{10}$

$(15)_8 = (13)_{10}$

$(31)_{10} - (13)_{10} = 18_{10} = \mathbf{22_8}$

# 5. Arithmetic Operation on Octal Number

**Octal Multiplication**

**Steps:**

- **Convert the Octal number to Decimal number and multiply them.**
- **Convert the resultant decimal number to octal number**

**Example $(62)_8 + (5)_8$**

Full Calculation:

$\mathbf{(62)_8} = (50)_{10}$

$\mathbf{(5)_8} = (5)_{10}$

$(50)_{10} \times (5)_{10} = 250_{10} = \mathbf{372_8}$

**Octal Division**

**Steps:**

- **Convert the Octal number to Decimal number and divide them.**
- **Convert the resultant decimal number to octal number**

**Example $(47)_8 + (15)_8$**

Full Calculation:

$\mathbf{(47)_8} = (39)_{10}$

$\mathbf{(15)_8} = (13)_{10}$

$(39)_{10} / (13)_{10} = 3_{10} = \mathbf{3_8}$

# 5. Arithmetic Operation on Hexadecimal Number

**Hexadecimal addition**

**Steps:**

- A carry is generated to the next higher column if the sum is greater than or equal to 16.

**Example $(5A)_{16}$ + $(BF)_{16}$**

$$\begin{array}{r} 5\ A \\ +B\ F \\ \hline 1\ 1\ 9 \end{array}$$

- A + F = 10 + 15 = 25 = 16 + 9. Here, 16 forms a carry to the next column. Thus, the sum is 9 with a 1 as carry to the next column.
- A + F = 10 + 15 = 25 = 16 + 9. Here, 16 forms a carry to the next column. Thus, the sum is 9 with a 1 as carry to the next column.

# 5. Arithmetic Operation on Hexadecimal Number

## Hexadecimal Subtraction

### Steps:

- Hexadecimal subtraction is similar to decimal subtraction. The only difference is that in hexadecimal subtraction, when the minuend digit is smaller than the subtrahend digit, a borrow 1, which is equivalent to 16, is taken from the higher column digit

### Example $(BC5)_{16}$ – $(1DA)_{16}$

$$\begin{array}{rrr} & 27 & \\ A & \cancel{B} & 21 \\ \cancel{B} & \cancel{C} & \cancel{5} \\ -1 & D & A \\ \hline 9 & E & B \\ \hline \end{array}$$

- Start from the digits in rightmost column: 5 A. Since 5 is less than A (10), so we have to borrow from the next higher-order digit. After borrowing from the next column (C), the digit 5 will become 5 + 16 (as 16 is equivalent to borrow 1) = 21. Thus, 21 – A = 11 (B). Write down B as the difference.
- Move to the next column and subtract the digits: B D. Again, B is smaller than D, so we take a borrow from the higher order digit B. After getting a borrow, B will become B + 16 = 27. Thus, 27 D = 14 (E). Write down the digit E as difference.
- Move to the leftmost column and subtract the digits: A 1 = 9. Write down the result.

# 5. Arithmetic Operation on Hexadecimal Number

## Hexadecimal Multiplication

**Steps:**

- Hexadecimal multiplication is similar to the decimal multiplication. In hexadecimal multiplication, a carry is generated to the next column when the product is greater than or equal to 16

.**Example** Multiply $(A19)_{16}$ by $(B)_{16}$

$$\begin{array}{r} 6\ 1\ 6\phantom{\ 9} \\ A\ 1\ 9 \\ \times\quad B \\ \hline 6\ F\ 1\ 3 \\ \hline \end{array}$$

Multiply the digit $(B)_{16}$ with each digit of the number $(A19)_{16}$ and write down the result.
Firstly, we multiply B by 9, it gives 99 = 96 + 3. Hence, 3 is written as product and 96 as carry 6 (16 6 = 96) to the next column.
Then, we multiply B by 1 and add the carry 6 to the product. It gives 17 = 16 + 1. Here, the result is 1 and carry is 1.
Finally, we multiply B by A and add the carry 1 overed from previous step to product. It gives 96 + 15 (F in hexadecimal). The result is F with a carry 6.
Thus, the final hexadecimal product of A19 and B is 6F13.

# 5. Arithmetic Operation on Hexadecimal Number

**Hexadecimal Division**

**Steps:**

**Step 1** – Start diving from the leftmost digit of the dividend.

**Step 2** – Multiply the obtained quotient by the divisor and subtract from the dividend.

**Step 3** – Bring down the next significant digit or digits of the dividend.

**Step 4** – Repeat the process explained in the above three steps until all the digits in the dividend are used.

.**Example** Divide $(A29)_{16}$ by $(5)_{16}$

$$
\begin{array}{r}
208 \\
5\overline{)A29} \\
\underline{A\phantom{29}} \\
029 \\
\underline{28} \\
1
\end{array}
$$

In this hexadecimal division, we have obtained the quotient $(208)_{16}$ and remainder $(1)_{16}$.

# 6. Signed and Unsigned Numbers system

*Unsigned binary numbers* are, by definition, positive numbers and thus do not require an arithmetic sign. An *m*-bit unsigned number represents all numbers in the range 0 to $2^m - 1$. For example, the range of 8-bit unsigned binary numbers is from 0 to $255_{10}$ in decimal and from 00 to $FF_{16}$ in hexadecimal.

***Signed numbers***, on the other hand, require an arithmetic sign. The most significant bit of a binary number is used to represent the sign bit. If the sign bit is equal to zero, the signed binary number is positive; otherwise, it is negative. The remaining bits represent the actual number. There are three ways to represent negative numbers.

# 6. Signed and Unsigned Numbers system

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. The following examples illustrate the sign-magnitude representation:

$$(+3) = 0011 \quad \Rightarrow \quad (-3) = 1011$$

$$(+7) = 0111 \quad \Rightarrow \quad (-7) = 1111$$

$$(+0) = 0000 \quad \Rightarrow \quad (-0) = 1000$$

# 6. 1's Complement

The 1s complement of a number is obtained by **complementing all the bits** of signed binary number. So, 1s complement of positive number gives a negative number. Similarly, 1s complement of negative number gives a positive number.
That means, if you perform two times 1s complement of a binary number including sign bit, then you will get the original signed binary number.

## Example

Consider the **negative decimal number -108**. The magnitude of this number is 108. We know the signed binary representation of 108 is 01101100.
It is having 8 bits. The MSB of this number is zero, which indicates positive number. Complement of zero is one and vice-versa. So, replace zeros by ones and ones by zeros in order to get the negative number.
$(108)_{10} = (10010011)_2$
Therefore, the **1s complement of $(108)_{10}$** is **$(10010011)_2$**.

# 6. 2's Complement

The 2s complement of a binary number is obtained by **adding one to the 1's complement** of signed binary number. So, 2's complement of positive number gives a negative number. Similarly, 2's complement of negative number gives a positive number.

That means, if you perform two times 2's complement of a binary number including sign bit, then you will get the original signed binary number.

**Example**

Consider the **negative decimal number -108**.

We know the 1s complement of $\mathbf{(108)_{10}}$ is $\mathbf{(10010011)_2}$

*2s compliment of* $(108)_{10}$ = *1s compliment of* $(108)_{10}$ + *1.*

= 10010011 + 1

= 10010100

Therefore, the **2s complement of** $\mathbf{(108)_{10}}$ is $\mathbf{(10010100)_2}$.