BCA0303 – Computer Organization

UNIT-I : Data Representation

Topic 2: Number System

What is a Number System?

Definition

A **Number System** is a method of writing and representing numbers using a set of digits or symbols. Every number system has a fixed **base (radix)** that determines the number of digits used.

A number system helps us perform calculations, store data, and represent information. In computers, different number systems are used because the computer understands only binary (0 and 1).

Example

We use the **Decimal Number System** in our daily life.

Examples:

- 25
- 150
- 9876
- 2026

A computer stores the same numbers in **Binary**.

Example:

Decimal	Binary
5	101
10	1010
25	11001

Why Do We Need Different Number Systems?

Different number systems make computer operations easier.

Uses

- Store data
- Perform calculations
- Memory addressing
- Programming
- Digital electronics
- Computer architecture

Types of Number Systems

There are **two main types** of number systems.

```
                Number System
                     │
          ┌──────────┴────────────┐
          │                       │
Non-Positional              Positional
Number System             Number System
          │                       │
   Roman Numbers        Binary
                         Decimal
                         Octal
                         Hexadecimal
```

---

1. Non-Positional Number System

Definition

In a **Non-Positional Number System**, the value of a symbol does **not** depend on its position.

Every symbol always has the same value.

Example

Roman Numbers

Symbol	**Value**
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Example:

```
VIII = 8

V = 5

III = 3

Total = 8
```

Another Example

```
XX = 20

XXX = 30

XL = 40
```

The value of X remains **10** everywhere.

Therefore, Roman Numbers are **Non-Positional Numbers**.

Advantages

- Easy to understand
- Used in clocks
- Used in book chapters
- Used in outlines

Disadvantages

- Difficult calculations
- No zero
- Not suitable for computers
- Large numbers are difficult to write

---

2. Positional Number System

Definition

In a **Positional Number System**, the value of a digit depends on **its position** in the number.

The same digit may have different values depending on where it is placed.

Example

Number

`356`

Digit	Position Value
3	300
5	50
6	6

Here,

```
356

=

3 × 100

+

5 × 10

+

6 × 1

=

356
```

Notice that the digit **3** has a value of **300**, not **3**, because of its position.

Another Example

```
222
```

Digit	Position	Value
2	Hundreds	200
2	Tens	20
2	Ones	2

Although all digits are **2**, their values are different because of their positions.

Characteristics of Positional Number System

- Each system has a base (radix).
- Position determines the value.
- Easy arithmetic operations.
- Used in digital computers.

Comparison Between Positional and Non-Positional Number System

Positional Number System	Non-Positional Number System
Value depends on position	Value does not depend on position
Uses base (radix)	No base
Used by computers	Not used by computers
Easy calculations	Difficult calculations
Example: Binary, Decimal	Example: Roman Numbers

Radix (Base)

Definition

The **Radix** (or **Base**) is the total number of unique digits or symbols used in a number system.

Each number system has a different radix.

Examples

Number System	Base	Digits Used
Binary	2	0, 1
Octal	8	0–7
Decimal	10	0–9
Hexadecimal	16	0–9, A–F

Radix Point

Definition

The **Radix Point** separates the **integer part** from the **fractional part** of a number.

It is similar to the decimal point in the decimal number system.

Example in Decimal

```
125.75
125 | 75
```

- Left side = Integer Part
- Right side = Fractional Part

Example in Binary

```
1011.101₂
1011 | 101
```

Here,

- 1011 = Integer Part
- 101 = Fractional Part

Importance of Radix Point

- Separates whole and fractional values.
- Used in floating-point calculations.
- Helps represent decimal values accurately.
- Important in scientific and engineering computations.

Key Points

- A number system represents numbers using symbols.
- There are two types of number systems.
- Computers use positional number systems.
- Binary has base 2.
- Decimal has base 10.
- Octal has base 8.
- Hexadecimal has base 16.
- Radix means base.
- Radix point separates the integer and fractional parts.

Practice Questions

Short Questions

1. Define Number System.
2. What is a Positional Number System?
3. What is a Non-Positional Number System?
4. Define Radix.
5. What is a Radix Point?

Long Questions

1. Explain the Positional Number System with suitable examples.
2. Explain the Non-Positional Number System.
3. Differentiate between Positional and Non-Positional Number Systems.
4. Explain Radix with examples.

5. Explain the role of the Radix Point.

Multiple Choice Questions

1. Which number system is used by computers?

A. Roman

B. Binary

C. Greek

D. Decimal

Answer: B. Binary

---

2. The base of the Decimal Number System is:

A. 2

B. 8

C. 10

D. 16

Answer: C. 10

---

3. Which is a Non-Positional Number System?

A. Binary

B. Decimal

C. Roman

D. Octal

Answer: C. Roman

---

4. The base of the Hexadecimal Number System is:

A. 8

B. 10

C. 12

D. 16

Answer: D. 16

---

5. The Radix Point separates:

A. Digits and symbols

B. Integer and fractional parts

C. Binary and decimal numbers

D. None of these

Answer: B. Integer and fractional parts

Topic 3: Decimal, Binary, Octal and Hexadecimal Number Systems

---

Introduction

A **Number System** is a method of representing numbers using a set of digits. Different number systems are used in computers because they make calculations, programming, and data storage easier.

The four most commonly used number systems are:

1. Decimal Number System
2. Binary Number System

3. Octal Number System
4. Hexadecimal Number System

---

1. Decimal Number System (Base 10)

Definition

The **Decimal Number System** is the most commonly used number system in our daily life. It has a **base (radix) of 10** because it uses **10 digits**.

Digits Used

```
0, 1, 2, 3, 4, 5, 6, 7, 8, 9
```

Positional Values

Position	Power of 10
Ones	$10^0 = 1$
Tens	$10^1 = 10$
Hundreds	$10^2 = 100$
Thousands	$10^3 = 1000$

Example

Number:

5732_{10}

Expanded form:

5×10^3

+

7×10^2

+

3×10^1

+

```
2 × 10⁰
5000 + 700 + 30 + 2

= 5732
```

Advantages

- Easy to understand
- Used in daily calculations
- Simple arithmetic operations

Disadvantages

- Computers cannot process decimal numbers directly.
- Must be converted into binary for processing.

---

2. Binary Number System (Base 2)

Definition

The **Binary Number System** is the language of computers. It has a **base of 2** because it uses only **two digits**.

Digits Used

```
0 and 1
```

These digits are called **Binary Digits (Bits).**

Why Do Computers Use Binary?

Electronic circuits inside a computer have only two stable states:

```
OFF = 0

ON = 1
```

Therefore, computers use binary numbers.

Positional Values

Position	**Power of 2**
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32

Example

Binary Number

110101_2

Expanded Form

1×32

$+$

1×16

$+$

0×8

$+$

1×4

$+$

0×2

$+$

1×1
$32+16+4+1$

$=53_{10}$

Advantages

- Used by all digital computers.
- Easy hardware implementation.
- Reliable and accurate.
- Fast processing.

Disadvantages

- Long representation of numbers.
- Difficult for humans to read.

---

3. Octal Number System (Base 8)

Definition

The **Octal Number System** has a **base of 8** and uses **eight digits**.

Digits Used

```
0
1
2
3
4
5
6
7
```

Digit **8** is **not allowed**.

Positional Values

Position	Power of 8
8^0	1
8^1	8
8^2	64
8^3	512

Example

Number

725_8

Expanded Form

7×64

+

2×8

+

5×1
448

+

16

+

5

=

469_{10}

Advantages

- Shorter than binary.
- Easy conversion to binary.
- Used in digital systems.

Disadvantages

- Less commonly used today.
- Limited digits.

---

4. Hexadecimal Number System (Base 16)

Definition

The **Hexadecimal Number System** has a **base of 16**.

It uses **16 symbols**.

Digits Used

```
0

1

2

3

4

5

6

7

8

9

A

B

C

D

E

F
```

Where

```
A = 10

B = 11

C = 12

D = 13

E = 14

F = 15
```

Positional Values

Position	Power of 16
16^0	1
16^1	16
16^2	256
16^3	4096

Example

$3A5_{16}$

Expanded Form

```
3×256

+

10×16

+

5×1
768

+

160

+

5

=
```

933_{10}

Advantages

- Very compact representation.
- Easy conversion to binary.
- Widely used in programming.
- Used for memory addresses and color codes.

Disadvantages

- Requires remembering A–F.
- Not commonly used in daily life.

Comparison of Number Systems

Number System	Base	Digits Used	Common Use
Decimal	10	0–9	Daily calculations
Binary	2	0,1	Computer processing
Octal	8	0–7	Digital electronics
Hexadecimal	16	0–9, A–F	Programming and memory

Memory Tip

```
Decimal     → Base 10

Binary      → Base 2

Octal       → Base 8

Hexadecimal → Base 16
```

Real-Life Applications

Decimal

- Banking
- Shopping
- Education

Binary

- Computer memory
- Digital circuits
- Data storage

Octal

- UNIX file permissions
- Digital systems

Hexadecimal

- HTML color codes (#FF0000)
- Memory addresses
- Programming and debugging

Summary

- Decimal uses **10 digits (0–9)**.
- Binary uses **2 digits (0 and 1)**.
- Octal uses **8 digits (0–7)**.
- Hexadecimal uses **16 symbols (0–9 and A–F)**.
- Computers process all data in binary.
- Octal and hexadecimal are compact ways of representing binary data.

Practice Questions

Short Answer Questions

1. Define the Decimal Number System.
2. Why is Binary called the language of computers?
3. List the digits used in the Octal Number System.
4. What symbols are used in the Hexadecimal Number System?
5. State one application of the Hexadecimal Number System.

Long Answer Questions

1. Explain the Decimal Number System with an example.
2. Explain the Binary Number System and its advantages.
3. Describe the Octal Number System.
4. Explain the Hexadecimal Number System with examples.
5. Compare the four number systems.

Multiple Choice Questions

1. Which number system is used internally by a computer?

A. Decimal
B. Binary
C. Octal
D. Roman

Answer: B. Binary

---

2. The base of the Octal Number System is:

A. 2
B. 8
C. 10
D. 16

Answer: B. 8

---

3. Which symbol represents the decimal value 15 in hexadecimal?

A. D
B. E
C. F
D. G

Answer: C. F

---

4. Which digit is not valid in the Octal Number System?

A. 6
B. 7
C. 8
D. 5

Answer: C. 8

---

5. The base of the Binary Number System is:

A. 2
B. 8
C. 10
D. 16

Answer: A. 2

Number System Conversion

Learning Objectives

After studying this topic, students will be able to:

- Convert Decimal to Binary.
- Convert Binary to Decimal.
- Convert Decimal to Octal.
- Convert Octal to Decimal.
- Convert Decimal to Hexadecimal.
- Convert Hexadecimal to Decimal.
- Convert Binary to Octal.
- Convert Octal to Binary.
- Convert Binary to Hexadecimal.
- Convert Hexadecimal to Binary.
- Convert Octal to Hexadecimal.
- Convert Hexadecimal to Octal.

Introduction

Different number systems are used in computers. Sometimes, it is necessary to convert a number from one number system to another.

For example,

- Decimal → Binary
- Binary → Decimal
- Binary → Hexadecimal

This process is called **Number System Conversion**.

---

1. Decimal to Binary Conversion

Definition

To convert a decimal number into binary, repeatedly divide the number by **2** and write the remainders. Read the remainders from **bottom to top**.

Steps

Step 1: Divide the decimal number by 2.

Step 2: Write the remainder.

Step 3: Divide the quotient again by 2.

Step 4: Continue until the quotient becomes 0.

Step 5: Read the remainders from bottom to top.

Example 1

Convert $\mathbf{25_{10}}$ into Binary.

Division	Quotient	Remainder
25 ÷ 2	12	1
12 ÷ 2	6	0
6 ÷ 2	3	0
3 ÷ 2	1	1
1 ÷ 2	0	1

Read from bottom to top

11001_2

Answer

$25_{10} = 11001_2$

Example 2

Convert **45_{10}** into Binary.

Division	Quotient	Remainder
45 ÷ 2	22	1
22 ÷ 2	11	0
11 ÷ 2	5	1
5 ÷ 2	2	1
2 ÷ 2	1	0
1 ÷ 2	0	1

Read from bottom to top

101101_2

Shortcut

Always remember

```
Repeated Division by 2

↓

Write Remainders
```

↓

Read Bottom to Top

---

2. Binary to Decimal Conversion

Definition

Multiply every binary digit by the corresponding power of 2 and add all the values.

Steps

Step 1: Write powers of 2 from right to left.

Step 2: Multiply each bit by its positional value.

Step 3: Add all the results.

Example

Convert

101101_2

into Decimal.

Bit	Position	Value
1	2^5	32
0	2^4	0
1	2^3	8
1	2^2	4
0	2^1	0
1	2^0	1

Total

32 + 8 + 4 + 1

=45

Answer

$101101_2 = 45_{10}$

---

3. Decimal to Octal Conversion

Definition

Divide the decimal number repeatedly by **8**.

Read the remainders from bottom to top.

Example

Convert

125_{10}

into Octal.

Division	Quotient	Remainder
125 ÷ 8	15	5
15 ÷ 8	1	7
1 ÷ 8	0	1

Read from bottom to top

175_8

Answer

$125_{10} = 175_8$

---

4. Octal to Decimal Conversion

Multiply every digit by powers of 8.

Example

Convert

175_8

to Decimal.

1×8^2

$+$

7×8^1

$+$

5×8^0
64

$+$

56

$+$

5

$=$

125

Answer

$175_8 = 125_{10}$

---

5. Decimal to Hexadecimal Conversion

Divide repeatedly by **16**.

Example

Convert

250_{10}

to Hexadecimal.

Division	Quotient	Remainder
250 ÷16	15	10(A)
15 ÷16	0	15(F)

Read from bottom to top

FA_{16}

Answer

$250_{10} = FA_{16}$

---

6. Hexadecimal to Decimal Conversion

Multiply digits by powers of 16.

Remember

A=10

B=11

C=12

D=13

E=14

F=15

Example

Convert

$2AF_{16}$

to Decimal.

2×16^2

+

10×16^1

+

15×16^0
512

+

160

+

15

=

687

Answer

$2AF_{16} = 687_{10}$

Summary of Conversion Methods

Conversion	Method
Decimal → Binary	Repeated division by 2
Binary → Decimal	Multiply by powers of 2
Decimal → Octal	Repeated division by 8
Octal → Decimal	Multiply by powers of 8
Decimal → Hexadecimal	Repeated division by 16
Hexadecimal → Decimal	Multiply by powers of 16

Practice Questions

Decimal → Binary

1. 35
2. 67
3. 145
4. 255
5. 512

Binary → Decimal

1. 10110
2. 111001
3. 1101101
4. 10010011
5. 11111111

Decimal → Octal

1. 56
2. 90
3. 250
4. 400
5. 1024

Octal → Decimal

1. 127_8
2. 345_8
3. 567_8
4. 701_8
5. 777_8

Decimal → Hexadecimal

1. 255
2. 500
3. 1024
4. 2048
5. 4095

Hexadecimal → Decimal

1. $A5_{16}$
2. $2B_{16}$
3. FF_{16}
4. $3C8_{16}$
5. $10A_{16}$

Multiple Choice Questions

1. Which method is used for Decimal to Binary conversion?

A. Divide by 2 repeatedly
B. Multiply by 2
C. Divide by 10
D. Multiply by 8

Answer: A

---

2. Which method is used for Binary to Decimal conversion?

A. Divide by 2
B. Multiply by powers of 2
C. Divide by 16
D. Multiply by 8

Answer: B

---

3. The hexadecimal digit A represents:

A. 8
B. 9
C. 10
D. 15

Answer: C

---

4. Which base is used in the Octal Number System?

A. 2
B. 8
C. 10
D. 16

Answer: B

---

5. Which base is used in the Hexadecimal Number System?

A. 2
B. 8
C. 10
D. 16

Answer: D

UNIT-I

Binary ↔ Octal ↔ Hexadecimal Conversion

Learning Objectives

After studying this topic, students will be able to:

- Convert Binary to Octal.
- Convert Octal to Binary.
- Convert Binary to Hexadecimal.
- Convert Hexadecimal to Binary.
- Convert Octal to Hexadecimal.
- Convert Hexadecimal to Octal.
- Use shortcut methods for quick conversions in examinations.

Introduction

Earlier, we learned conversions using repeated division and positional values. However, there are faster methods for converting between **Binary, Octal, and Hexadecimal**.

These shortcut methods are widely used because:

- They save time.
- They reduce calculation errors.
- They are frequently asked in university examinations.

---

1. Binary to Octal Conversion

Rule

- Start grouping binary digits from the **right side**.
- Make groups of **3 bits**.
- If a group has fewer than 3 bits, add leading zeros.
- Convert each group into its octal equivalent.

Binary to Octal Conversion Table

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Example 1

Convert

101101_2

Step 1

Group into 3 bits.

```
101   101
```

Step 2

Convert each group.

```
101 = 5
```

```
101 = 5
```

Answer

$101101_2 = 55_8$

Example 2

Convert

1110101_2

Step 1

Add one leading zero.

```
011   101   01
```

Correct grouping:

```
001   110   101
```

Step 2

```
001 = 1
```

```
110 = 6
```

```
101 = 5
```

Answer

$1110101_2 = 165_8$

---

2. Octal to Binary Conversion

Rule

Replace every octal digit with its **3-bit binary equivalent**.

Example

Convert

527_8

Step 1

Replace each digit.

5 = 101

2 = 010

7 = 111

Answer

$527_8 = 101010111_2$

---

3. Binary to Hexadecimal Conversion

Rule

- Group binary digits into **4 bits** from the right side.
- Add leading zeros if necessary.
- Replace each group with its hexadecimal value.

Binary to Hexadecimal Table

Binary	Hex
0000	0
0001	1
0010	2

Binary	**Hex**
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

Example

Convert

101011111_2

Step 1

Group into 4 bits.

```
0001 0101 1111
```

Step 2

```
0001 = 1
```

```
0101 = 5
```

```
1111 = F
```

Answer

$101011111_2 = 15F_{16}$

---

4. Hexadecimal to Binary Conversion

Rule

Replace every hexadecimal digit with its **4-bit binary equivalent**.

Example

Convert

$2AF_{16}$

Step 1

2 = 0010

A = 1010

F = 1111

Step 2

Join them.

001010101111_2

Answer

$2AF_{16} = 001010101111_2$

---

5. Octal to Hexadecimal Conversion

Method

Convert in two steps.

Octal

↓

Binary

↓

Hexadecimal

Example

Convert

725_8

Step 1

Convert to Binary.

```
7 =111
```

```
2 =010
```

```
5 =101
```

Binary

```
111010101
```

Step 2

Group into four bits.

```
0001
```

```
1101
```

```
0101
```

Step 3

```
0001 =1
```

```
1101 =D
```

```
0101 =5
```

Answer

$725_8 = 1D5_{16}$

---

6. Hexadecimal to Octal Conversion

Method

Convert in two steps.

```
Hexadecimal
↓
Binary
↓
Octal
```

Example

Convert

$3A_{16}$

Step 1

```
3 =0011
A =1010
```

Binary

```
00111010
```

Step 2

Group into three bits.

```
000
111
010
```

Step 3

```
000 =0
111 =7
010 =2
```

Answer

$3A_{16} = 72_8$

Complete Shortcut Table

Binary	Octal	Hexadecimal
000	0	-
001	1	-
010	2	-
011	3	-
100	4	-
101	5	-
110	6	-
111	7	-

Binary	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

Memory Tips

Binary → Octal

```
3 Bits = 1 Octal Digit
```

Binary → Hexadecimal

```
4 Bits = 1 Hexadecimal Digit
```

These are the most important shortcut rules for exams.

Practice Questions

Binary → Octal

1. 110101_2
2. 111001_2
3. 10011101_2
4. 101010101_2
5. 11111111_2

Octal → Binary

1. 237_8
2. 651_8
3. 777_8
4. 145_8
5. 530_8

Binary → Hexadecimal

1. 11001111_2
2. 10101010_2
3. 11110000_2
4. 100111101_2
5. 111111111_2

Hexadecimal → Binary

1. $A5_{16}$
2. FF_{16}
3. $2BC_{16}$
4. $1F_{16}$
5. $3D7_{16}$

Multiple Choice Questions

1. In Binary to Octal conversion, binary digits are grouped into:

A. 2 bits

B. 3 bits

C. 4 bits

D. 8 bits

Answer: B. 3 bits

---

2. In Binary to Hexadecimal conversion, binary digits are grouped into:

A. 2 bits

B. 3 bits

C. 4 bits

D. 5 bits

Answer: C. 4 bits

---

3. The hexadecimal equivalent of binary 1111_2 is:

A. E

B. D

C. F

D. A

Answer: C. F

---

4. The binary equivalent of hexadecimal A_{16} is:

A. 1000

B. 1001

C. 1010

D. 1011

Answer: C. 1010

---

5. Binary 101_2 is equal to which octal digit?

A. 4

B. 5

C. 6

D. 7

Answer: B. 5

UNIT-I

Binary Coded Decimal (BCD), Alphanumeric Codes and Hamming Code

---

Part 1: Binary Coded Decimal (BCD)

Learning Objectives

After studying this topic, students will be able to:

- Define Binary Coded Decimal (BCD).
- Convert Decimal numbers to BCD.
- Convert BCD to Decimal.
- Understand BCD arithmetic.
- Explain the advantages and disadvantages of BCD.

---

1. What is Binary Coded Decimal (BCD)?

Definition

Binary Coded Decimal (BCD) is a coding system in which **each decimal digit (0–9) is represented separately using a 4-bit binary number**.

Unlike pure binary, the entire number is **not** converted at once. Each decimal digit is converted individually.

Example 1

Convert **25** into BCD.

Step 1

Separate the digits.

```
25

↓

2         5
```

Step 2

Convert each digit into 4-bit binary.

Decimal Digit	BCD Code
2	0010
5	0101

Answer

```
25 = 0010 0101 (BCD)
```

Example 2

Convert **489** into BCD.

Step 1

Separate digits.

```
4

8

9
```

Step 2

Convert individually.

Digit	BCD
4	0100
8	1000
9	1001

Answer

```
489 =

0100 1000 1001
```

BCD Table

Decimal	BCD
0	0000

Decimal	BCD
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Note: BCD uses only the binary combinations **0000 to 1001**. The combinations **1010 to 1111** are **invalid** in BCD because there are no decimal digits greater than 9.

Decimal to BCD Conversion

Example

Convert **356** into BCD.

Step 1

Separate digits.

```
3
5
6
```

Step 2

Convert each digit.

```
3 =0011
5 =0101
6 =0110
```

Answer

`356`

`=`

`0011 0101 0110`

BCD to Decimal Conversion

Example

Convert

`0011 0101 0110`

to Decimal.

Separate into groups of four bits.

`0011`

`0101`

`0110`

Convert each group.

`0011 =3`

`0101 =5`

`0110 =6`

Answer

`356`

Advantages of BCD

- Easy conversion between decimal and binary.
- Used in calculators and digital clocks.
- High accuracy in decimal calculations.
- Easy for humans to understand.

Disadvantages of BCD

- Requires more storage than binary.
- Arithmetic operations are slower.
- Less efficient for computer processing.

Applications of BCD

- Digital clocks
- Calculators
- Banking systems
- Electronic meters
- Financial applications

Part 2: BCD Arithmetic

BCD arithmetic follows decimal arithmetic but with an additional correction step.

BCD Addition Rule

1. Add the BCD numbers as binary.
2. If the result of a 4-bit group is greater than **1001 (9)** or produces a carry, add **0110 (6)** to that group.
3. The corrected result is the valid BCD answer.

Example

Add **5 + 4** in BCD.

BCD values:

```
5 =0101
```

```
4 =0100
```

Binary addition:

```
 0101
+0100
------
 1001
```

The result **1001** equals decimal **9**, which is a valid BCD value. Therefore, **no correction is needed**.

Answer: `1001` (BCD for 9)

Example (Correction Required)

Add **8 + 7** in BCD.

BCD values:

`8 =1000`

`7 =0111`

Binary addition:

```
 1000
+0111
------
1111
```

`1111` is greater than `1001`, so add `0110`.

```
1111
+0110
------
1 0101
```

The result is:

`0001 0101`

This represents **15** in BCD.

Part 3: Alphanumeric Codes

Definition

An **Alphanumeric Code** is a coding system used to represent **letters, numbers, symbols, and special characters** in binary form.

Computers store text using these codes.

Types of Alphanumeric Codes

1. ASCII
2. EBCDIC
3. Unicode

---

1. ASCII

ASCII stands for:

American Standard Code for Information Interchange

ASCII uses **7 bits**, allowing **128 characters**.

Examples:

Character	ASCII Decimal	Binary
A	65	1000001
B	66	1000010
a	97	1100001
0	48	0110000

Advantages of ASCII

- Simple
- Widely used
- Easy data exchange

Limitations

- Supports only 128 characters.
- Cannot represent many international languages.

---

2. EBCDIC

EBCDIC stands for:

Extended Binary Coded Decimal Interchange Code

- Uses **8 bits**
- Developed by IBM
- Mainly used on IBM mainframe computers

---

3. Unicode

Unicode is a modern coding standard that supports characters from almost every language in the world.

Examples:

- English
- Hindi
- Punjabi
- Chinese
- Arabic

Common Unicode formats:

- UTF-8
- UTF-16
- UTF-32

Comparison of Character Codes

Feature	ASCII	EBCDIC	Unicode
Bits	7	8	Variable
Characters	128	256	More than one million

Feature	ASCII	EBCDIC	Unicode
Used In	General computers	IBM systems	Modern computers and the Internet

---

Part 4: Hamming Code

Definition

Hamming Code is an error-detection and error-correction technique developed by **Richard Hamming**.

It helps detect and correct errors that occur during data transmission.

Why is Hamming Code Needed?

Sometimes, data changes while travelling through a communication channel due to noise or interference.

For example:

Original data:

```
101101
```

Received data:

```
100101
```

One bit has changed.

Hamming Code helps find and correct this error automatically.

Hamming Code Process

```
Original Data
```

↓

```
Add Parity Bits
```

```
↓
Transmit Data
↓
Receive Data
↓
Check for Errors
↓
Correct Error (if one bit is incorrect)
↓
Output Correct Data
```

Advantages

- Detects single-bit errors.
- Corrects single-bit errors.
- Improves data reliability.
- Used in computer memory (ECC RAM) and communication systems.

Limitations

- More parity bits increase storage overhead.
- Efficient only for limited error patterns.

Summary

- **BCD** stores each decimal digit separately using **4 bits**.
- **BCD Arithmetic** may require adding **0110** when a digit exceeds 9.
- **ASCII** uses **7 bits** and represents 128 characters.
- **EBCDIC** is an **8-bit** code used mainly by IBM systems.
- **Unicode** supports characters from almost every language.
- **Hamming Code** detects and corrects single-bit errors during data transmission.

Practice Questions

Short Answer Questions

1. What is Binary Coded Decimal (BCD)?
2. Write the BCD code for the decimal number **57**.
3. What is ASCII?
4. Expand EBCDIC.
5. Why is Unicode used?
6. What is Hamming Code?
7. What is the purpose of parity bits?

Long Answer Questions

1. Explain Binary Coded Decimal (BCD) with suitable examples.
2. Explain BCD Arithmetic with examples.
3. Describe ASCII, EBCDIC, and Unicode. Compare them.
4. Explain the working of Hamming Code with a neat diagram.
5. Write the advantages and applications of BCD and Hamming Code.

Multiple Choice Questions

1. **BCD represents each decimal digit using:**
   - A. 2 bits
   - B. 3 bits
   - C. 4 bits
   - D. 8 bits

   Answer: C
2. **ASCII uses:**
   - A. 6 bits
   - B. 7 bits
   - C. 8 bits
   - D. 16 bits

   Answer: B
3. **Unicode is mainly used to:**
   - A. Compress files
   - B. Store graphics
   - C. Represent characters of many languages
   - D. Increase processor speed

   Answer: C

4. **Hamming Code is used for:**
    - A. Data compression
    - B. Error detection and correction
    - C. Image processing
    - D. File encryption

    Answer: B
5. **Which organization developed EBCDIC?**
    - A. Intel
    - B. Microsoft
    - C. IBM
    - D. Apple

    Answer: C