NEXT IAS

BIG LEARNINGS MADE EASY

Digital Electronics

Electrical Engineering

Comprehensive Theory *with* Solved Examples

Civil Services Examination

MADE EASY Publications Pvt. Ltd.

Corporate Office: 44-A/4, Kalu Sarai (Near Hauz Khas Metro Station), New Delhi-110016
E-mail: infomep@madeeasy.in
Contact: 9021300500

Visit us at: www.madeeasypublications.org

Digital Electronics

First Edition : 2018
Second Edition : 2019
Third Edition : 2021
Reprint : 2022
Reprint : 2023

Reprint : 2024

Contents

Digital Electronics

Chapter 1

Number Systems..... 1-31

1.1 Positional Number Systems/Weighted Number Systems ...2
1.2 Codes 8
1.3 Arithmetic Operations..... 13
1.4 Signed Number Representation..... 18
1.5 Over Flow Concept..... 24
Practice Questions..... 26

Chapter 2

Boolean Algebra and Logic Gates 32-86

2.1 Logic Operations..... 32
2.2 Laws of Boolean Algebra..... 33
2.3 Boolean Algebraic Theorems..... 34
2.4 Minimization of Boolean Functions 38
2.5 Representation of Boolean Functions 38
2.6 Karnaugh Map..... 45
2.7 Logic Gates..... 56
Practice Questions..... 78

Chapter 3

Combinational Circuits..... 87-147

3.1 Design Procedure for Combinational Circuit 87
3.2 Adders and Subtractors..... 95
3.3 Binary Adders 100
3.4 Code Converters..... 107
3.5 Parity Bit Generators and Checkers 110
3.6 Magnitude Comparators 112
3.7 Encoders..... 114
3.8 Decoders 118
3.9 Multiplexers..... 126
3.10 Demultiplexers..... 136
Practice Questions..... 139

Chapter 4

Flip-Flops and Registers..... 148-178

4.1 Latches and Flip-Flops..... 149
4.2 Conversion from One Type of Flip-Flop to Another Type 160
4.3 Operating Characteristics of a Flip-Flop 163
4.4 Application of Latches and Flip-Flops 166
4.5 Registers 167
Practice Questions..... 174

Chapter 5

Counters 179-225

5.1 Asynchronous Counters (or Ripple Counters) 181
5.2 Decoding Errors (Glitches or Spikes) in Asynchronous Counters..... 185
5.3 Synchronous Counters..... 188
5.4 Synchronous Series Carry Counter 198
5.5 Synchronous Parallel Carry Counter..... 199
5.6 Design of Synchronous Counters..... 200
5.7 Pulse Train Generation (Sequence Generators) 211
Practice Questions..... 219

Chapter 6

Synthesis of Synchronous Sequential Circuits 226-264

6.1 Finite State Machine (FSM) 226
6.2 State Reduction 230
6.3 State Assignment 235
6.4 Design of a Sequential Circuit or Finite State Machine..... 237
6.5 The Sequence Detector 246
Practice Questions..... 256

Chapter 7

Programmable Logic Devices, Multivibrators, Timers and Memories..... 265-291

7.1 Programmable Logic Devices 266
7.2 Semiconductor Memories 278
7.3 MUltivibrators 283
7.4 555 Timer Circuit 284
Practice Questions 288

Chapter 8

Logic Families 292-343

8.1 Switching Circuits 293
8.2 Classification of Digital Logic Family 297
8.3 Characteristics of Digital Logic Family 298
8.4 Logic Families 304
8.5 Interfacing of Logic Families 335
Practice Questions 338

Chapter 9

A/D and D/A Converters 344-372

9.1 Digital to Analog Converter 344
9.2 Analog to Digital Converters 358
Practice Questions 369

■■■■

Digital Electronics

01

CHAPTER

Number Systems

INTRODUCTION

Electronic systems are of two types:

(*i*) Analog systems (*ii*) Digital systems

Analog systems are those systems in which voltage and current variations are continuous through the given range and they can take any value within the given specified range, whereas a digital system is one in which the voltage level assumes finite number of distinct values. In all modern digital circuits, there are just two discrete voltage level.

Digital circuits are often called switching circuits, because the voltage levels in a digital circuit are assumed to be switched from one value to another instantaneously. Digital circuits are also called logic circuits, because every digital circuit obeys a certain set of logical rules.

Digital systems are extensively used in control systems, communication and measurement, computation and data processing, digital audio and video equipments, etc.

Advantages of Digital Systems

Digital systems have number of advantages over analog systems which are summarized below:

1. **Ease of Design :** The digital circuits having two voltage levels, OFF and ON or LOW and HIGH, are easier to design in comparison with analog circuits in which signals have numerical significance ; so their design is more complicated.
2. **Greater Accuracy and Precision :**Digital systems are more accurate and precise than analog systems because they can be easily expanded to handle more digits by adding more switching circuits.
3. **Information Storage is Easy :** There are different types of semiconductor memories having large capacity, which can store digital data.
4. **Digital Systems are More Versatile :** It is easy to design digital systems whose operation is controlled by a set of stored instructions called program. However in analog systems, the available options for programming is limited.
5. **Digital Systems are Less Affected by Noise :** The effect of noise in analog system is more. Since in analog systems the exact values of voltages are important. In digital system noise is not critical because only the range of values is important.
6. **Digital Systems are More Reliable :** As compared to analog systems, digital systems are more reliable.

Limitations of Digital System

(*i*) The real world is mainly analog.

(*ii*) Human does not understand the digital data.

A number system is a way to represent numbers. Number systems are basically of two types:

1. **Non-positional number system:** In non-positional number system, each symbol represents the same value regardless of its position.
   Eg.:- Roman Number System. Here, the symbol V always means "Five" whether it occurs last in a numeral string (eg. XXV) or fourth from last (XXVIII).
2. **Positional number system:** In a positional number system, each symbol represents different value depending on the position they occupy in a number.
   Eg.:- Decimal Number System.

REMEMBER

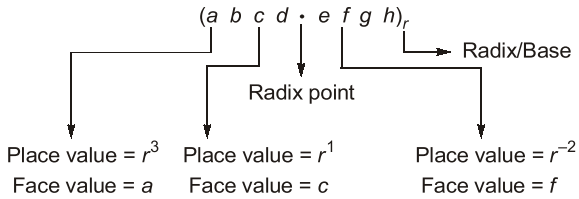
- Positional Number System is also known as weighted Number System.
- Non-positional Number System is also known as Unweighted Number System.

1.1 POSITIONAL NUMBER SYSTEMS/WEIGHTED NUMBER SYSTEMS

Many number systems are used in digital technology. The most commonly used number systems are:

- Decimal number system
- Octal number system
- Binary number system
- Hexadecimal number system

Number representation in weighted number system :



r = Base or Radix ; Place value = positional weight

The digit present in greatest positional weight = Most Significant Digit (MSD)

The digit present in lowest positional weight = Least Significant Digit (LSD)

a, b, c, d, e, f, g, h = Different symbols used to represent a number in a number system. A number system with base 'b' will have 'b' different digits ranging from 0 to (b – 1).

(i) Decimal ⇒ 10 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

(ii) Binary ⇒ 2 (0, 1)

(iii) Octal ⇒ 8 (0, 1, 2, 3, 4, 5, 6, 7, 8, 7)

(iv) Hexadecimal ⇒ 16 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

In the positional number system represented by sequence $(abcd.efgh)_r$, it's value can be computed in the decimal number system using the following formula:

$$(N)_{10} = a \times r^3 + b \times r^2 + c \times r^1 + d \times r^0 + e \times r^{-1} + f \times r^{-2} + g \times r^{-3} + h \times r^{-4}$$

EXAMPLE - 1.1

In a particular number system, 24 + 17 = 40. Find the base of the system.

Solution:

$$[(2\times r)+(4\times 1)]+[(1\times r)+(7\times 1)] = (4\times r)+(0\times 1)$$
$$3r+11 = 4r$$
$$r = 11$$

EXAMPLE - 1.2

In a particular number system, $\sqrt{41}=5$. Find the base of the system.

Solution:

$$\sqrt{(4\times r)+(1\times 1)} = 5\times 1$$
$$\sqrt{(4r+1)} = 5$$
$$4r+1 = 25$$
$$r = 6$$

EXAMPLE - 1.3

In a particular number system, roots of $x^2 - 11x + 22 = 0$ are 3, 6. Find the base of the system.

Solution:

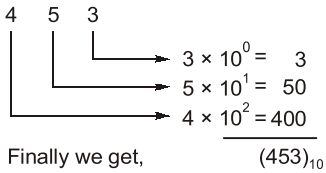
For $ax^2 + bx + c = 0$, product of roots $= \frac{c}{a}$; Sum of roots $= -\frac{b}{a}$

$$(3)_r(6)_r = \frac{(22)_r}{(1)_r}$$
$$(3\times r^0)(6\times r^0) = \frac{(2\times r^1)+(2\times r^0)}{(1\times r^0)}$$
$$18 = \frac{2r+2}{1}$$
$$r = 8$$

1.1.1 Decimal Number System

- This system has **'base 10'**.
- It has 10 distinct symbols (0, 1, 2, 3, 4, 5, 6, 7, 8 and 9).
- This is a positional value system in which the value of a digit depends on its position.

  $\Rightarrow$ Let we have $(453)_{10}$ as a decimal number. Then,



$\therefore$ We can say "3" is the least significant digit(LSD) and "4" is the most significant digit(MSD).

1.1.2 Binary Number System

- It has base '2' i.e. it has two base numbers 0 and 1 and these base numbers are called "Bits".
- Each position in a binary number system represents a power of the base (2).

- In this number system, group of "Four bits" is known as "Nibble" and group of "Eight bits" is known as "Byte".

  i.e. 4 bits = 1 Nibble; 8 bits = 1 Byte

Binary to Decimal Conversion

A binary number is converted to decimal equivalent simply by summing together the weights of various positions in the binary number which contains **'1'**.

EXAMPLE - 1.4

Find the decimal number representation of $(101101.10101)_2$.

Solution:

$$(101101.10101)_2 = 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3} + 0\times2^{-4} + 1\times2^{-5}$$

$$= 32 + 0 + 8 + 4 + 0 + 1 + \frac{1}{2} + 0 + \frac{1}{8} + 0 + \frac{1}{32} = (45.65625)_{10}$$

Decimal to Binary Conversion

The integral decimal number is repeatedly divided by '2' and writing the remainders after each division until a quotient '0' is obtained.

EXAMPLE - 1.5

Convert $(13)_{10}$ into its equivalent binary number.

Solution:

	Quotient	Remainder	
13 ÷ 2	6	1	LSB
6 ÷ 2	3	0	
3 ÷ 2	1	1	
1 ÷ 2	0	1	MSB

$\therefore$ $(13)_{10} = (1101)_2$

REMEMBER To convert Fractional decimal into binary, Multiply the number by '2'. After first multiplication integer digit of the product is the first digit after binary point. Later only fraction part of the first product is multiplied by 2. The integer digit of second multiplication is second digit after binary point, and so on. The multiplication by 2 only on the fraction will continue like this based on conversion accuracy or until fractional part becomes zero.

EXAMPLE - 1.6

Convert $(0.65625)_{10}$ into its equivalent binary number.

Solution:

0.65625 × 2	0.31250 × 2	0.62500 × 2	0.25000 × 2	0.50000 × 2
1.31250	0.62500	1.25000	0.50000	1.00000
↓ 1	↓ 0	↓ 1	↓ 0	↓ 1

Thus, $(0.65625)_{10} = (0.10101)_2$

1.1.3 Octal Number System

- It is very important in digital computer because by using the octal number system, the user can simplify the task of entering or reading computer instructions and thus save time.
- It has a base of '8' and it posses 8 distinct symbols (0,1...7).
- It is a method of grouping binary numbers in group of three bits.

Octal to Decimal Conversion

An octal number can be converted to decimal equivalent by multiplying each octal digit by its positional weight.

EXAMPLE - 1.7

Convert $(6327.4051)_8$ into its equivalent decimal number.

Solution:

$$(6327.4051)_8 = 6\times8^3+3\times8^2+2\times8^1+7\times8^0+4\times8^{-1}+0\times8^{-2}+5\times8^{-3}+1\times8^{-4}$$

$$= 3072+192+16+7+\frac{4}{8}+0+\frac{5}{512}+\frac{1}{4096}$$

$$= (3287.5100098)_{10}$$

Thus, $(6327.4051)_8 = (3287.5100098)_{10}$

Decimal to Octal Conversion

- It is similar to decimal to binary conversion.
- For integral part, number is repeatedly divided by '8' and for fraction part, it is multiplied by '8' based on conversion accuracy or till fractional part becomes zero.

EXAMPLE - 1.8

Convert $(3287.5100098)_{10}$ into its equivalent octal number.

Solution:

For integral part:

	Quotient	Remainder
3287 ÷ 8	410	7
410 ÷ 8	51	2
51 ÷ 8	6	3
6 ÷ 8	0	6

$\therefore$ $(3287)_{10} = (6327)_8$

Now for fractional part:

0.5100098 × 8 = 4.0800784 → 4

0.0800784 × 8 = 0.6406272 → 0

0.6406272 × 8 = 5.1250176 → 5

0.1250176 × 8 = 1.0001408 → 1

$\therefore$ $(0.5100098)_{10} = (0.4051)_8$

Finally, $(3287.5100098)_{10} = (6327.4051)_8$

Octal-to-Binary Conversion

This conversion can be done by converting each octal digit into binary individually.

EXAMPLE - 1.9

Convert $(472)_8$ into its equivalent binary number.

Solution:

$$\therefore \quad (472)_8 = (\underset{\Downarrow}{4}\ \underset{\Downarrow}{7}\ \underset{\Downarrow}{2})_2 = (100\ 111\ 010)_2$$

Binary-to-Octal Conversion

In this conversion the binary bit stream are grouped into groups of three bits starting at the LSB of integer part and then each group is converted into its octal equivalent. If the leftmost group has fewer bits, attach the required number of leading 0's to complete group of 3-bits. After decimal point, grouping start from left and add required number of 0's in the last to complete group of 3-bits.

EXAMPLE - 1.10

Convert $(1011011110.11001010011)_2$ into its equivalent octal number.

Solution:

For left-side of the radix point, we grouped the bits from LSB: $\underset{1}{001}\ \underset{3}{011}\ \underset{3}{011}\ \underset{6}{110}$

Here two 0's at MSB are added to make a complete group of 3 bits.

For right-side of the radix point, we grouped the bits from MSB: $\underset{\text{radix point}}{\bullet}\ \underset{6}{110}\ \underset{2}{010}\ \underset{4}{100}\ \underset{6}{110}$

Here a '0' at LSB is added to make a complete group of 3 bits.

Finally, $(1011011110.11001010011)_2 = (1336.6246)_8$

1.1.4 Hexadecimal Number System

- The base for this system is "16", which requires 16 distinct symbols to represent the numbers.
- It is a method of grouping 4 bits.
- This number system contains numeric digits (0, 1, 2,....9) and alphabets (*A*, *B*, *C*, *D*, *E* and *F*) both, so this is an "ALPHANUMERIC NUMBER SYSTEM".
- The letters *A* through *F* are number digits representing value 10 to 15.
- Microprocessor deals with instructions and data that use hexadecimal number system for programming purposes.

- To signify a hexadecimal number, a subscript 16 or letter '*H*' is used i.e. $(A7)_{16}$ or $(A7)_H$.

Hexadecimal	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Hexadecimal-to-Decimal Conversion

EXAMPLE - 1.11

Convert $(3A.2F)_{16}$ into its equivalent decimal number.

Solution:

$$(3A.2F)_{16} = 3 \times 16^1 + 10 \times 16^0 + 2 \times 16^{-1} + 15 \times 16^{-2}$$

$$= 48 + 10 + \frac{2}{16} + \frac{15}{16^2} = (58.1836)_{10}$$

Decimal-to-Hexadecimal Conversion

EXAMPLE - 1.12

Convert $(675.625)_{10}$ into its equivalent Hexadecimal number.

Solution:

For Integral Part:

	Quotient	Remainder
$675 \div 16$	42	3
$42 \div 16$	2	$10 = A$
$2 \div 16$	0	2

$\therefore$ $(675)_{10} = (2A3)_{16}$

For Fractional Part:

$0.625 \times 16 = 10 = A$

$\therefore$ $(0.625)_{10} = (0.A)_{16}$

Finally, $(675.625)_{10} = (2A3.A)_{16}$

Hexadecimal-to-Binary Conversion

For this conversion replace each hexadecimal digit by its 4 bit binary equivalent.

EXAMPLE - 1.13

Convert $(2F9A)_{16}$ into its equivalent binary number.

Solution:

2	F	9	A
↓	↓	↓	↓
0010	1111	1001	1010

$\therefore \quad (2F9A)_{16} = (0010\ 1111\ 1001\ 1010)_2$

Binary-to-Hexadecimal Conversion

For this conversion the binary bit stream is grouped into pairs of four (starting from LSB) and hex number is written for its equivalent binary group. 0's at MSB are added to complete a group of 4-bits.

EXAMPLE - 1.14

Convert $(10100110101111)_2$ into its equivalent hexadecimal number.

Solution:

00 10	10 01	10 10	1111
↓	↓	↓	↓
2	9	A	F

Here two 0's at MSB are added to make a complete group of 4 bits.

$\therefore \quad (10100110101111)_2 = (29AF)_{16}$

NOTE

The number systems can also be classified as weighted binary number and unweighted binary number. Where weighted number system is a positional weighted system for example, Binary, Octal, Hexadecimal *BCD*, 2421 etc. The unweighted number systems are non-positional number system, for example Gray code, Excess-3 code etc.

EXAMPLE - 1.15

Evaluate $(1.2)_4 + (2.3)_4 = (\_\_\_\_\_)_4$.

Solution:

$$\begin{array}{r} 1.2 \\ 2.3 \\ \hline 1 \\ \hline 10.1 \end{array}$$

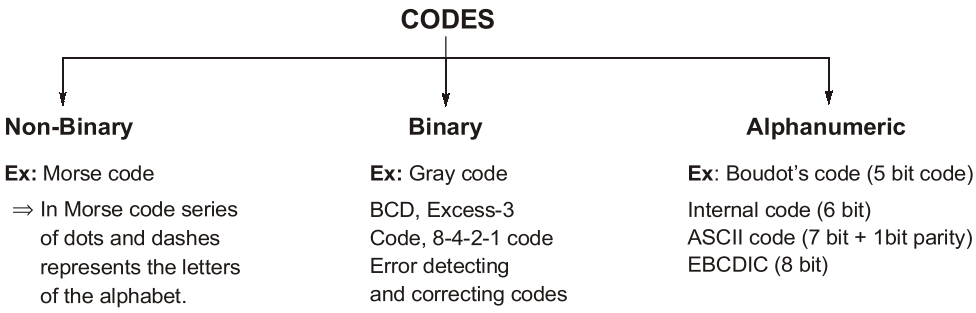
$4\,|\,5$, $1-1$ ⇓ $(5)_{10} = (11)_4$

$4\,|\,4$, $1-0$ ⇓ $(4)_{10} = (10)_4$

$(1.2)_4 + (2.3)_4 = (10.1)_4$

1.2 CODES

When numbers, letters or words are represented by a special group of symbols, we say that they are being encoded, and the group of symbols is called "CODE".



1.2.1 Binary Coded Decimal Code (BCD)

- In this code, each digit of a decimal number (0 to 9) is represented by binary equivalent.
- It is a 4-bit binary code.
- It is also known as "8-4-2-1 code" or simply "BCD Code".
- It is very useful and convenient code for input and output operations in digital circuits.
- Also, it is a "weighted code system".
- It is not a self-complementing code.

- A code is said to be self-complementing, if the codeword of the 9's complement of *N*, i.e., of (9 – *N*) can be obtained from the code word of N by interchanging all the 0s and 1s.
- In a self-complementing code, the code for 9 is the complement for the code of 0, the code for 8 is the complement for the code of 1 and so on.

For example:

(i) $(943)_{decimal} \longrightarrow (.....)_{BCD}$

$\Rightarrow$ 9 4 3

↓ ↓ ↓

1001 0100 0011

$\therefore$ $(943)_{10} = (100101000011)_{BCD}$

(ii) $(23.15)_{10} \longrightarrow (.....)_{BCD}$

2 3 · 1 5

↓ ↓ ↓ ↓

0010 0011 0001 0101

$(23.15)_{10} = (00100011 \cdot 00010101)_{BCD}$

Advantages of BCD Code

- The main advantage of the BCD code is relative ease of converting to and from decimal.
- Only 4-bit code groups for the decimal digits "0 through 9" need to be remembered.
- This case of conversion is especially important from the hardware standpoint.

$\Rightarrow$ In 4-bit binary formats, total number of possible representation = 2^4 = 16 i.e. 0 to 15

Then, Valid BCD codes = 10

Invalid BCD codes = 6

$\Rightarrow$ In 8-bit binary formats, total number of possible representation = 2^8 = 256 i.e. 0 to 255

Valid BCD codes = 100 (00 to 99)

Invalid BCD codes = 256 – 100 = 156

BCD Weighted Codes

- The BCD code described above is also known as 8421 BCD code, with 8, 4, 2 and 1 representing the weights of different bits in the four-bit groups, starting from MSB and proceeding towards LSB.
- Other weighted BCD codes include the 4221 BCD and 5421 BCD codes. Again, 4, 2, 2 and 1 in the 4221 BCD code and 5, 4, 2 and 1 in the 5421 BCD code represent weight of the relevant bits. Table below shows a comparison of 8421, 4221 and 5421 BCD codes:

Decimal	8421 BCD code	4221 BCD code	5421 BCD code
0	0000	0000	0000
1	0001	0001	0001
2	0010	0010 (or 0100)	0010
3	0011	0011 (or 0101)	0011
4	0100	1000 (or 0110)	0100
5	0101	0111 (or 1001)	1000 (or 0101)
6	0110	1100 (or 1010)	1001 (or 0110)
7	0111	1101 (or 1011)	1010
8	1000	1110	1011
9	1001	1111	1100

REMEMBER Since 8421 code is the most popular of all the BCD codes, it is simply referred to as the BCD code.

BCD to Binary Conversion

A given BCD number can be converted into an equivalent binary number by first writing its decimal equivalent and then converting it into its binary equivalent.

BCD Number: $0010\ 1001.\ 01110101$

$$\downarrow$$

$$(29.75)_{10}$$

$$\downarrow$$

$$(11101.11)_2$$

Binary to BCD Conversion

The process of binary to BCD conversion is same as the process of BCD to binary conversion executed in reverse order.

Example: $(10101011.101)_2 \rightarrow (171.625)_{10}$

$$\downarrow$$

$$(0001\ 0111\ 0001.0110\ 0010\ 0101)_{BCD}$$

1.2.2 Excess-3 Code

- It is a 4-bit code.
- The excess-3 code for a given decimal number is determined by adding '3' to each decimal digit (0 to 9) in the given number and then replacing each decimal digit by its four bit binary equivalent.
- It can be derived from BCD code by adding 3(i.e. 0011) to each coded number.
- It is an "unweighted code".
- It is a "self-complementing code" i.e. the 1's complement of an excess-3 number is the excess-3 code for the 9's complement of corresponding decimal number.
- This code is used in arithmetic circuits because of its property of self complementing.

PRACTICE QUESTIONS

Question: 1

Consider the following arithmetic operations. Each of them are correct in atleast one number system. Calculate the minimum non-zero base for which the following operations are true:

(i) $3 \times 10 = 30$

(ii) $\frac{54}{4} = 13$

(iii) $\frac{302}{20} = 12.1$

Question: 2

Convert

(i) 110101_2 into octal number system

(ii) $DADA.C_{16}$ into decimal

(iii) 1000111001.01_2 into hexadecimal

(iv) 243.3125_{10} into Binary

Question: 3

(a) Determine the binary representation of the decimal number 10.05 with an absolute error equal to 0.005.

(b) Represent the decimal number 0.452 in binary number system with relative error of 0.1%.

Question: 4

Formulate a weighted binary code for the decimal digits using weights 6, 3, 1, 1.

Question: 5

Assign a binary code in some orderly manner to the 52 playing cards. Use the minimum number of bits.

Question: 6

Using signed 2's complement method, perform the following operations:

(i) $(-29)_{10} + (-48)_{10}$

(ii) $(29)_{10} - (48)_{10}$

(iii) $(29)_{10} + (48)_{10}$

Question: 7

Represent the decimal number 4, 246 in

(i) Binary Decimal code

(ii) Excess-3 code

(iii) 2421 code

(iv) 6311 code

Question: 8

Perform the following operations:

(i) $4A6_{16} - 1B3_{16}$ using Hexadecimal subtraction

(ii) $983_{10} + 187_{10}$ using Excess-3 addition

(iii) $984_{10} + 599_{10}$ using BCD addition

(iv) $55644_8 + 02621_8$ using Octal addition

Practice Questions Explanations

1. ***Solution*:**

Let the base of the expression be 'x'. Hence,

(i) $(3)_x \times (10)_x = (30)_x$

$\Rightarrow \quad 3(x + 0) = (3x + 0)$

$\Rightarrow \quad 3x = 3x$

Thus, the equation is valid for any value of x. So, the minimum base value will be 4 as it requires four different digits 0, 1, 2 and 3.

(ii) $$\frac{(54)_x}{(4)_x} = (13)_x$$

$$\Rightarrow \quad \frac{5x + 4}{4} = x + 3$$

$\Rightarrow \quad 5x + 4 = 4x + 12$

$\Rightarrow \quad x = 8$

Hence, the minimum non-zero base is 8.

(iii) $$\frac{(302)_x}{(20)_x} = (12.1)_x$$

$$\frac{3x^2 + 0 + 2}{2x + 0} = x + 2 + x^{-1}$$

$$\Rightarrow \quad \frac{3x^2 + 2}{2x} = \frac{x^2 + 2x + 1}{x}$$

$\Rightarrow \quad 3x^2 + 2 = 2x^2 + 4x + 2$

$\Rightarrow \quad x^2 - 4x = 0$

$\Rightarrow \quad x = 4 \quad (x = 0$ is not possible)

Thus, the minimum non-zero base is equal to 4.

2. ***Solution*:**

(i) $(110101)_2$ **into octal :**

To convert into octal, the binary bit stream is grouped into pairs of three, starting from LSB and octal number is written for the binary equivalent.

$$\underset{\downarrow}{110}\ \underset{\downarrow}{101}$$
$$6 \quad 5$$

Hence, $(110101)_2 = (65)_8$

(ii) $(DADA.C)_{16}$ **into decimal :**

$$(DADA.C)_{16} = 13 \times (16)^3 + 10 \times (16)^2 + 13 \times (16)^1 + 10 \times (16)^0 + 12 \times (16)^{-1}$$
$$= 53248 + 2560 + 208 + 10 + 0.75 = (56026.75)_{10}$$

(iii) 1000111001.01_2 into hexadecimal :

For left-side of the radix point, we group bits from LSB in group of 4 and add 0's at the MSB to make complete group of 4-bits.

For right-side of the radix point, we group bits from MSB in group of 4 and add 0's at LSB to make complete group of 4-bits.

$$\begin{array}{cccc} 0010 & 0011 & 1001\cdot & 0100 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 2 & 3 & 9 & 4 \end{array}$$

$$\therefore \quad (1000111001.01)_2 = (239.4)_{16}$$

(iv) 243.3125_{10} into binary :

2	243	
2	121	1
2	60	1
2	30	0
2	15	0
2	7	1
2	3	1
2	1	1
	0	1

LSB ↑ MSB

0.3125 × 2	0.625 × 2	0.25 × 2	0.50 × 2
0.625	1.25	0.50	1.00
↓ 0	↓ 1	↓ 0	↓ 1

$$\therefore \quad (243.2125)_{10} = (11110011.0101)_2$$

3. *Solution*:

(a) To ensure that the difference between the binary represented value and the decimal number 10.05 remains less than or equal to 0.005, the required number of bits 'n' must follow the relationship:

$$2^{-n} \le 0.005$$

$$\Rightarrow \quad n \ge -\log_2(0.005)$$

$$\Rightarrow \quad n \ge 7.64$$

$$\Rightarrow \quad n = 8$$

0.05 × 2	0.10 × 2	0.20 × 2	0.40 × 2	0.80 × 2	0.60 × 2	0.20 × 2	0.40 × 2
0.10	0.20	0.40	0.80	1.60	1.20	0.40	0.80
↓ 0	↓ 0	↓ 0	↓ 0	↓ 1	↓ 1	↓ 0	↓ 0

$$\Rightarrow \quad (0.05)_{10} = (00001100)_2$$

$$(10.05)_{10} = (1010.00001100)_2$$

(b) The desired value of the absolute error is $0.001 \times 0.452 = 0.000452$. Hence,

$$2^{-n} \le 0.000452$$

$$\Rightarrow \quad n \ge -\log_2(0.000452)$$

$$\Rightarrow \quad n \ge 11.11$$

$$\Rightarrow \quad n = 12$$

0.452 × 2	0.904 × 2	0.808 × 2	0.616 × 2	0.232 × 2	0.464 × 2	0.928 × 2	0.856 × 2
0.904	1.808	1.616	1.232	0.464	0.928	1.856	1.712

0.696 × 2	0.848 × 2	0.424 × 2	0.712 × 2
1.392	1.696	0.848	1.424

Hence, $(0.452)_{10} = (0.011100111011)_2$ with error of 0.1%