APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

LOGIC CIRCUIT DESIGN (ECT-203)

MODULE 1 NOTES

PREPARED BY : JIBIN EP, ASSISTANT PROFESSOR, EKC TC MANJERI

Index

Index

SL NO	TOPIC	SLIDE LINK	VIDEO LINK
7	Binary Subtraction Methods	BINARY SUBTRACTION METHODS	https://youtu.be/KXAQbY1jmQk
8	Octal Subtraction Methods	OCTAL SUBTRACTION	https://youtu.be/TKgNCOleZcU
9	Hexadecimal subtraction method	HEXADECIMAL SUBTRACTION	https://youtu.be/ifkq4MayHzo
10	Binary coded decimal	BINARY CODED DECIMAL	https://youtu.be/zZ77qwm2-3Y
11	Special Codes	SPECIAL CODES	https://youtu.be/KgjVmuwQjSw
12	Hamming Codes	HAMMING CODES	https://youtu.be/xfj2UMkfT_g
13	Introduction to Verilog Programming	INTRODUCTION TO VERILOG PROGRAMMING	https://youtu.be/TQNl363ZrAI

SYLLABUS

No	Topic	No. of Lectures
1	**Number Systems and Codes:**	
1.1	Binary, octal and hexadecimal number systems; Methods of base conversions;	2
1.2	Binary, octal and hexadecimal arithmetic;	1
1.3	Representation of signed numbers; Fixed and floating point numbers;	3
1.4	Binary coded decimal codes; Gray codes; Excess 3 code :	1
1.5	Error detection and correction codes - parity check codes and Hamming code-Alphanumeric codes:ASCII	3
1.6	Verilog basic language elements: identifiers, data objects, scalar data types, operators	2

MODULE 1
(TOPIC-1)

NUMBER SYSTEM (PART 1)

INDEX

https://youtu.be/7pDc0CjxsKY YouTube

Number System

- Number system is a basis for counting various items
- The **number system** or the **numeral system** is the **system** of naming or representing **numbers**
- It is a mathematical notation for representing numbers of a given set, using digits or other symbols in a consistent manner.

Number System Classification

BINARY NUMBER SYSTEM
- **BASE 2**
- 0,1

DECIMAL NUMBER SYSTEM
- **Base 10**
- 0,1,2,3,4,5,6,7,8,9

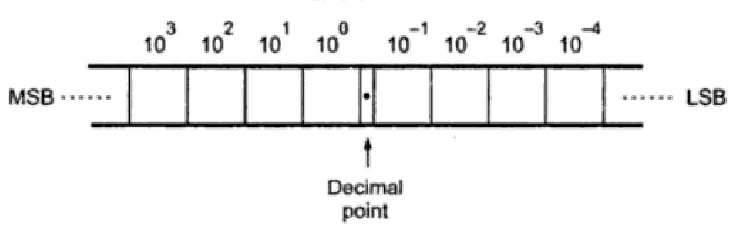
OCTAL NUMBER SYSTEM
- **Base 8**
- 0,1,2,3,4,5,6,7,

HEXA DECIMAL NUMBER SYSTEM
- **Base 16**
- 0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F

DECIMAL NUMBER SYSTEM

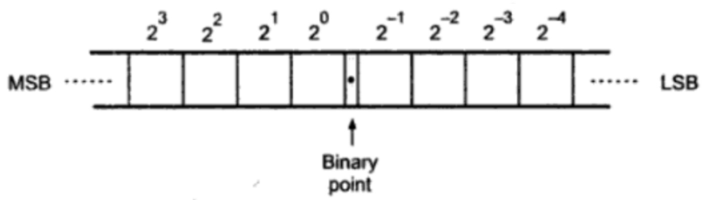
- Decimal number system is a **Radix-10** Number System
- There are 10 different digits/symbols
- These are **0,1,2,3,4,5,6,7,8,9**
- For Example,We can write a decimal number say

$$5678.9 = 5000 + 600 + 70 + 8 + 0.9$$



BINARY NUMBER SYSTEM

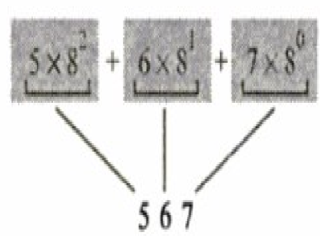
- Binary number system is a **Radix-2** Number System
- There are 2 different digits/symbols . These are **0 & 1**
- Binary Number system is expressed as a **Power of 2**



OCTAL NUMBER SYSTEM

- Octal number system is a **Radix-8** Number System
- Octal number system uses first eight digits of decimal number system **0,1,2,3,4,5,6,7**
- The Octal Number is expressed as a **Power of 8**

Example : The Given Octal number 567 is expressed as a power of 8



HEXADECIMAL NUMBER SYSTEM

- Hexadecimal number system is a **Radix-16** Number System
- Hexadecimal number system consist of 16 Different Symbols/Numbers **0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F**
- The Hexadecimal Number is expressed as a **Power of 16**

Example : The Given Octal number 3FD is expressed as a power of 16

$$3\times16^2 + F\times16^1 + D\times16^0$$

3 F D

RELATIOSHIP BETWEEN DECIMAL, BINARY, OCTAL AND HEXADECIMAL NUMBER

Decimal	Octal	Hexadecimal	Binary
0	0	0	0 0 0 0
1	1	1	0 0 0 1
2	2	2	0 0 1 0
3	3	3	0 0 1 1
4	4	4	0 1 0 0
5	5	5	0 1 0 1
6	6	6	0 1 1 0
7	7	7	0 1 1 1
8	10	8	1 0 0 0
9	11	9	1 0 0 1
10	12	A	1 0 1 0
11	13	B	1 0 1 1
12	14	C	1 1 0 0
13	15	D	1 1 0 1
14	16	E	1 1 1 0
15	17	F	1 1 1 1

NUMBER SYSTEM CONVERSIONS

- 2 BINARY
- 10 DECIMAL
- 8 OCTAL
- 16 HEXA DECIMAL

NUMBER BASE CONVERSIONS

Decimal Number to other Bases Conversion

- Decimal
  - Binary
  - Octal
  - Hexadecimal

1. Decimal to Binary Conversion

- The following two types of operations take place, while converting decimal number into its equivalent binary number.

Successive Division
- Division of integer part and successive quotients with base 2.

Successive Multiplication
- Multiplication of fractional part and successive fractions with base 2.

Shastra Technical Institute 7012716607 9633908979

2. Decimal to Octal Conversion

- The following two types of operations take place, while converting decimal number into its equivalent octal number.

Successive Division
- Division of integer part and successive quotients with base 8.

Successive Multiplication
- Multiplication of fractional part and successive fractions with base 8.

3. Decimal to Hexadecimal Conversión

- The following two types of operations take place, while converting decimal number into its equivalent octal number.

Successive Division
- Division of integer part and successive quotients with base 16.

Successive Multiplication
- Multiplication of fractional part and successive fractions with base 16.

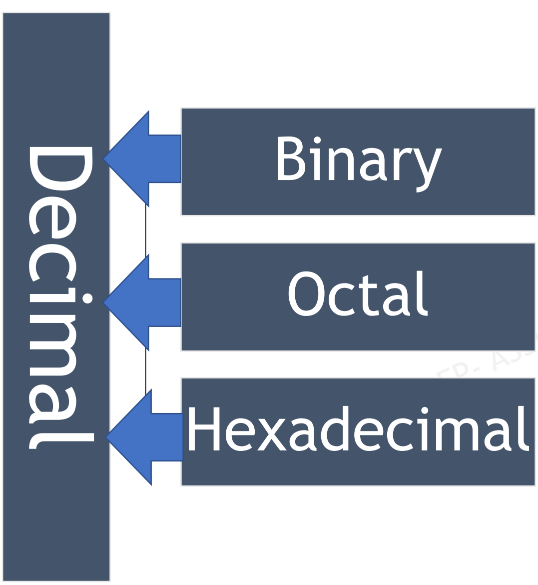
MODULE 1
(TOPIC-2)

NUMBER SYSTEM (PART 2)

INDEX

https://youtu.be/lZ4qOUcJeNA YouTube

Other Bases to Decimal Number Conversion



Positional Weight

$$a_3a_2a_1a_0.a_{-1}a_{-2}$$

$$(a_3 * r^3 + a_2 * r^2 + \cdots a_{-2} * r^{-2}$$

4. Binary to Decimal Conversion

- For converting a binary number into its equivalent decimal number, first multiply the bits of binary number with the respective positional weights and then add all those products.

Example

Consider the binary number 1101.11.

Mathematically, we can write it as

$(1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (1 \times 2^{-2})$

$\Rightarrow 8 + 4 + 0 + 1 + 0.5 + 0.25 = 13.75$

$\Rightarrow 1101.11_2 = 13.75_{10}$

Therefore, the **decimal equivalent** of binary number 1101.11 is 13.75.

5. Octal to Decimal Conversion

- For converting an octal number into its equivalent decimal number, first multiply the digits of octal number with the respective positional weights and then add all those products.

Example

Consider the **octal number 145.23**.

Mathematically, we can write it as

$145.23_8 = (1 \times 8^2) + (4 \times 8^1) + (5 \times 8^0) + (2 \times 8^{-1}) + (3 \times 8^{-2})$

$145.23_8 = 64 + 32 + 5 + 0.25 + 0.05 = 101.3$

$145.23_8 = 101.3_{10}$

- Therefore, the **decimal equivalent** of octal number 145.23 is 101.3.

6.Hexa-Decimal to Decimal Conversion

- For converting Hexa-decimal number into its equivalent decimal number, first multiply the digits of Hexa-decimal number with the respective positional weights and then add all those products.

Example

Consider the **Hexa-decimal number 1A5.2**

Mathematically, we can write it as

$1A5.2_{16} = (1 \times 16^2) + (10 \times 16^1) + (5 \times 16^0) + (2 \times 16^{-1})$

$1A5.2_{16} = 256 + 160 + 5 + 0.125 = 421.125$

$1A5.2_{16} = 421.125_{10}$

Therefore, the **decimal equivalent** of Hexa-decimal number 1A5.2 is 421.125.

MODULE 1
(TOPIC-3)

NUMBER SYSTEM (PART 3)

INDEX

https://youtu.be/KX0Veh8n0ds YouTube

OTHER CONVERSIONS

Binary to Octal	• Start from the binary point and make the groups of 3 bits on both sides of binary point
Binary to Hexadecimal	• Start from the binary point and make the groups of 4 bits on both sides of binary point
Octal to Binary	• By representing each octal digit with 3 bits
Hexadecimal to Binary	• By representing each octal digit with 4bits
Octal to Hexadecimal	• Convert octal number -**binary number.** • Convert the binary number -**Hexa-decimal number.**
Hexadecimal to Octal	• Convert HD -**binary number.** • Convert Binary number -**Octal Number**

7. Binary to Octal Conversion

- We know that the bases of binary and octal number systems are 2 and 8 respectively. Three bits of binary number is equivalent to one octal digit, since $2^3 = 8$.
- Follow these two steps for converting a binary number into its equivalent octal number.
- Start from the binary point and make the groups of 3 bits on both sides of binary point. If one or two bits are less while making the group of 3 bits, then include required number of zeros on extreme sides.
- Write the octal digits corresponding to each group of 3 bits.

8. Binary to Hexadecimal Conversion

- We know that the bases of binary and Hexa-decimal number systems are 2 and 16 respectively. Four bits of binary number is equivalent to one Hexa-decimal digit, since **$2^4 = 16$.**
- Follow these two steps for converting a binary number into its equivalent Hexa-decimal number.
- Start from the binary point and make the groups of 4 bits on both sides of binary point. If some bits are less while making the group of 4 bits, then include required number of zeros on extreme sides.
- Write the Hexa-decimal digits corresponding to each group of 4 bits.

9. Octal to Decimal Conversion

- For converting an octal number into its equivalent decimal number, first multiply the digits of octal number with the respective positional weights and then add all those products.

Example

Consider the **octal number 145.23.**

Mathematically, we can write it as

$145.23_8 = (1 \times 8^2) + (4 \times 8^1) + (5 \times 8^0) + (2 \times 8^{-1}) + (3 \times 8^{-2})$

$145.23_8 = 64 + 32 + 5 + 0.25 + 0.05 = 101.3$

$145.23_8 = 101.3_{10}$

- Therefore, the **decimal equivalent** of octal number 145.23 is 101.3.
-

10.Octal to Binary Conversion

- The process of converting an octal number to an equivalent binary number is just opposite to that of binary to octal conversion. By **representing each octal digit with 3 bits**, we will get the equivalent binary number.

Example

Consider the **octal number 145.23.**

Represent each octal digit with 3 bits.

$145.23_8 = 001100101.010011_2$

The value doesn't change by removing the zeros, which are on the extreme side.

$145.23_8 = 1100101.010011_2$

- Therefore, the **binary equivalent** of octal number 145.23 is 1100101.010011.

11.Octal to Hexa-Decimal Conversion

- Follow these two steps for converting an octal number into its equivalent Hexa-decimal number.
- Convert octal number into its equivalent **binary number.**
- Convert the above binary number into its equivalent **Hexa-decimal number.**

12.Hexa-Decimal to Decimal Conversion

- For converting Hexa-decimal number into its equivalent decimal number, first multiply the digits of Hexa-decimal number with the respective positional weights and then add all those products.

Example

Consider the **Hexa-decimal number 1A5.2**

Mathematically, we can write it as

$1A5.2_{16} = (1 \times 16^2) + (10 \times 16^1) + (5 \times 16^0) + (2 \times 16^{-1})$

$1A5.2_{16} = 256 + 160 + 5 + 0.125 = 421.125$

$1A5.2_{16} = 421.125_{10}$

Therefore, the **decimal equivalent** of Hexa-decimal number 1A5.2 is 421.125.

13.Hexa-Decimal to Octal Conversion

- Follow these two steps for converting Hexa-decimal number into its equivalent octal number.
- Convert Hexa-decimal number into its equivalent **binary number.**
- Convert the above binary number into its equivalent **octal number.**

PRACTICE PROBLMES

- $(250.55)_{10} - ()_{16}, ()_8$
- $(357)_8 - ()_{10}, ()_{16}$
- $(110101.1011)_2 - ()_8, ()_{16}$
- $(463.25)_{10} - ()_2, ()_8$
- $(36.25)_{10} - ()_8, ()_{16}$
- $(455)_{10} - ()_2, ()_4, ()_8$
- $(12.0625)_{10} - ()_{16}, ()_8$
- $(37.FC)_{16} - ()_2, ()_8, ()_{10}$

PRACTICE PROBLMES

$(250.55)_{10}$ to Hexadecimal

$(357)_8$ to Decimal

$(110101.1011)_2$ to Octal

MODULE 1
(TOPIC-4)

REPRESENTATION OF NEGATIVE NUMBER

INDEX

YouTube

Representation of Signed Numbers

- So far, we have considered only positive numbers. The representation of negative number is also equally important
- There are two ways of representing signed numbers
    - Sign Magnitude Form
    - Complement form
- There are two complement forms
    - 1’s Complement form
    - 2’s Complement form

1- Signed Magnitude Representation

- In Signed Magnitude form , an additional bit called **Sign Bit** is placed in front of the number
  - **If the signed bit is zero, Number is Positive**
  - **If the signed bit is one, Number is Negative**

Example :Signed Magnitude Representation

Sign bit	Magnitude						
0	1	0	1	0	0	1	= + 41

Sign bit	Magnitude						
1	1	0	1	0	0	1	= − 41

- Under the sign magnitude system , A great deal of manipulation is necessary to add a positive number to a negative number
- Thus, through a sign magnitude number system is possible, it is impractical

2- Representation of signed numbers using 2's or 1's complement

1. • If the number is positive, the magnitude is represented its true binary form and sign bit 0 is placed Infront of the MSB

2. • If the number is negative, the magnitude is represented in its 2's (or 1's) complement form and sign bit 1 placed in front of MSB

Example :
Representation of + 51 and -51 in 2's and 1's complement method

Sign bit	Magnitude							
0	1	1	0	0	1	1	= + 51	(In sign magnitude form) (In sign 2's complement form) (In sign 1's complement form)
1	1	1	0	0	1	1	= – 51	(In sign magnitude form)

Sign bit	Magnitude							
1	0	0	1	1	0	1	= – 51	(In sign 2's complement form)
1	0	0	1	1	0	0	= – 51	(In sign 1's complement form)

Example 2 : Each of the following number is signed binary number. Determine the decimal value in each case , if there are
(i) Signed Magnitude Form (ii) 2's Complement form (iii) 1's Complement form
(a) 01101 (b) 010111

Given Number	Signed Magnitude Form	2's Complement form	1's Complement form
01101	+13	+13	+13
010111	+23	+23	+23

Example 2 : Each of the following number is signed binary number. Determine the decimal value in each case , if there are
(i) Signed Magnitude Form (ii) 2's Complement form (iii) 1's Complement form
(c) 10111 (d) 1101010

Given Number	Signed Magnitude Form	2's Complement form	1's Complement form
10111	-7	-9	-8
1101010	-40	-22	-21

Example 3:
Express -45 in 8 bit 2's complement form

+45 = 00101101

-45=

1'S COMPLEMENT

11010010

2'S COMPLEMENT

11010011

Example 4:
Express -73.75 in 12 bit 2's complement form

+73.75 = 01001001.1100

-73.75

1'S COMPLEMENT

10110110.0011

2'S COMPLEMENT

10110110.0100

NEGATIVE NUMBER ADDITION (REFER VIDEO)

INDEX

YouTube

MODULE 1 (TOPIC-5)

FLOATING POINT NUMBERS

INDEX

YouTube

Floating Point Numbers

- In the decimal number system, Very large and very small numbers are expressed in scientific notations, By starting number (Mantissa) and an exponent of 10
- Example of Floating point numbers are $6.53\ X\ 10^{-27}$ & $1.58\ X\ 10^{21}$
- Binary numbers are also expressed in the same notations by starting number and exponent of 2
- So in general we can say that the floating point representation has 3 parts

  - Mantissa
  - Base
  - Exponent

Floating Point Numbers

- The scientific notation of floating point representation is

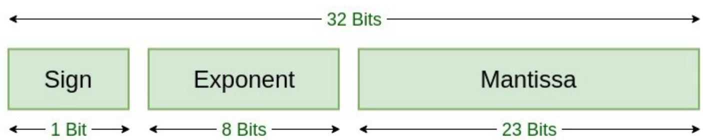
$$\pm M\ X\ B^{E}$$

Number	Mantissa	Base	Exponent
$9\ X\ 10^{8}$	9	10	8
$110\ X\ 2^{7}$	110	2	7
4364.784	4364784	10	-3

IEEE 754 Floating Point Representation

Single Precision Format

32 Bit number is Considered

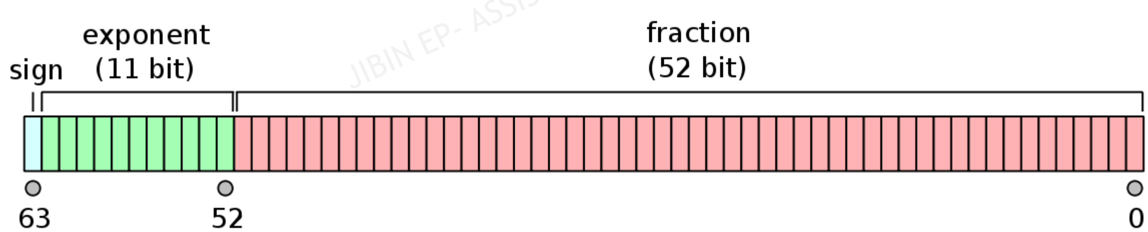


Single Precision
IEEE 754 Floating-Point Standard

IEEE 754 Floating Point Representation

Double Precision Format

64 Bit number is Considered



IEEE 754 Steps

1. Convert Decimal to Binary
2. Normalize the Number
3. Apply single precision and double precision equations

Equations for Single and Double Precisions

1. Convert Decimal to Binary
2. Normalize the Number
3. Apply single precision and double precision equations

Equations for Single and Double Precisions

Single Precision

$$(1.N)2^{E-127}$$

Double Precision

$$(1.N)2^{E-1023}$$

Equations for Single and Double Precisions

Numerical problems included in the video

MODULE 1
(TOPIC-6)

BINARY OCTAL HEXADECIMAL ADDITION (REFER VIDEO)

INDEX

https://youtu.be/Brcs1YOZvws YouTube

MODULE 1
(TOPIC-7)

BINARY SUBTRACTION METHODS (NUMERICALS WILL BE AVAILABLE IN THE VIDEO ONLY)

INDEX

https://youtu.be/KXAQbY1jmQk YouTube

BINARY SUBTRACTION

1. General Method
2. 1’s Complement Method
3. 2’s Complement Method

GENERAL METHOD OF BINARY SUBTRACTION

- 0 - 0 = 0
- 1 - 1 = 0
- 1 - 0 = 1
- 0 - 1 = 1
  With Borrow 1

1’s and 2’s complement of Binary Number

1’s complement Subtract each and every Binary Digit from **Number 1**

- Example : 1’s Compliment of 10101 is 01010 (Invert the numbers)
- Example : 1’s Compliment of 101.001 is 010.110 (Invert the numbers)

2’s Complement 1’s Complement +1

- Example : 2’s Compliment of 10101 is 01011 (01010+1)
- Example : 2’s Compliment of 101.001 is 010.111 (010.110+1)

BINARY SUBTRACTION USING 1’S COMPLEMENT

1
- Find the 1’s compliment of the number to be subtracted

2
- Perform Addition (Add first number and 1’s complement of Subtrahend)

3
- If Carry is generated, Add the carry with the result
- If there is no carry, Take the 1’s complement of the sum and assign negative sign

BINARY SUBTRACTION USING 2'S COMPLEMENT

1
- Find the 2's compliment of the number to be subtracted

2
- Perform Addition (Add first number and 2's complement of Subtrahend)

3
- If Carry is generated, Discard the carry
- If there is no carry, Take the 2's complement of the sum and assign negative sign

MODULE 1
(TOPIC-8)

OCTAL SUBTRACTION
(NUMERICALS WILL BE AVAILABLE IN THE VIDEO ONLY)

INDEX

https://youtu.be/TKgNCOleZcU YouTube

OCTAL SUBTRACTION

1 General Method

2 7's Complement Method

3 8's Complement Method

7's and 8's complement of Octal Number

7s complement Subtract each and every octal Digit from **Number 7**

- Example : 7's Compliment of 65 is 12 (77-65=12)
- Example : 7's Compliment of 423 is 354 (777-423=354)

8s Complement 7's Complement +1

- Example : 8's Compliment of 27 is 51 (77-27 + 1 =51)
- Example : 8's Compliment of 321 is 457 (777-321 + 1 =457)

OCTAL SUBTRACTION USING 7'S COMPLEMENT

1
- Find the 7's compliment of the number to be subtracted

2
- Perform Addition (Add first number and 7's complement of Subtrahend)

3
- If Carry is generated, Add the carry with the result
- If there is no carry, Take the 7's complement of the sum and assign negative sign

OCTAL SUBTRACTION USING 8'S COMPLEMENT

1
- Find the 8's compliment of the number to be subtracted

2
- Perform Addition (Add first number and 8's complement of Subtrahend)

3
- If Carry is generated, Discard the carry
- If there is no carry, Take the 8's complement of the sum and assign negative sign

MODULE 1
(TOPIC-9)

HEXADECIMAL SUBTRACTION
(NUMERICALS WILL BE AVAILABLE IN THE VIDEO ONLY)

INDEX

https://youtu.be/ifkq4MayHzo YouTube

HEXADECIMAL SUBTRACTION

1. General Method
2. 15's Complement Method
3. 16's Complement Method

15’s and 16’s complement of Hexadecimal Number

15’s complement Subtract each and every HD Digit from **Number F (15)**

- Example : 15’s Compliment of A86 is **579** (FFF-A86=579)
- Example : 15’s Compliment of C14 is 3EB (FFF=C14=3EB)

16’s Complement 15’s Complement +1

- Example : 15’s Compliment of A86 is **580** (FFF-A86+1=580)
- Example : 16’s Compliment of C14 is 3EC(FFF=C14+1=3EC)

HEXADECIMAL SUBTRACTION USING 15’S COMPLEMENT

1
- Find the 15’s compliment of the number to be subtracted

2
- Perform Addition (Add first number and 15’s complement of Subtrahend)

3
- If Carry is generated, Add the carry with the result
- If there is no carry, Take the 15’s complement of the sum and assign negative sign

HEXADECIMAL SUBTRACTION USING 16'S COMPLEMENT

1. • Find the 16's compliment of the number to be subtracted
2. • Perform Addition (Add first number and 16's complement of Subtrahend)
3. • If Carry is generated, Discard the carry
   • If there is no carry, Take the 16's complement of the sum and assign negative sign

MODULE 1
(TOPIC-10)

BINARY CODED DECIMAL

INDEX

YouTube

BINARY CODED DECIMAL

- In computing and electronic systems, binary-coded decimal (BCD) is a class of binary encodings of decimal numbers where each digit is represented by a fixed number of bits, usually four or eight. Sometimes, special bit patterns are used for a sign or other indications
- In this code each digit is represented by
- 4 bit Binary Number (8421)

(REFER VIDEO)

BINARY CODED DECIMAL

- In this code each decimal digit is represented by a 4-bit binary number. BCD is a way to express each of the decimal digits with a binary code. In the BCD, with four bits we can represent sixteen numbers (0000 to 1111). But in BCD code only first ten of these are used (0000 to 1001). The remaining six code combinations i.e. 1010 to 1111 are invalid in BCD

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

BCD ADDITION

1 $Sum \leq 9;$ $Carry\ 0$	• Answer is correct • Valid BCD
2 $Sum \leq 9;$ $Carry\ 1$	• Invalid BCD • Correct it by adding 6, 0110
3 $Sum > 9$	• Invalid BCD • Correct it by adding 6, 0110

BCD SUBTRACTION USING 9'S COMPLEMENT

1. Take 9s complement of the number to be subtracted
2. Add it to the result using BCD Addition
3. If the result is invalid, correct it by adding 6 (0110)
4. Shift the carry to next bit
5. If End around carry is generated , add it to the result

BCD SUBTRACTION USING 10'S COMPLEMENT

1. Take 10's complement of the number to be subtracted
2. Add it to the result using BCD Addition
3. If the result is invalid, correct it by adding 6 (0110)
4. Shift the carry to next bit
5. If End around carry is generated , discard the carry. (If number is negative, take 10's complement)

MODULE 1 (TOPIC-11)

SPECIAL CODES

INDEX

https://youtu.be/KgjVmuwQjSw

YouTube

TYPES OF CODES

- SPECIAL CODES
  - WEIGHTED CODE
  - NON WEIGHTED CODE
  - REFLECTIVE CODE
  - ERROR DETECTING CODE
  - ALPHA NUMERIC CODE

WEIGHTED CODE

- In Weighted code, Each position of the number represent a specific weight
- For Example, In Decimal code, if number is 567 then weight of 5 is 100, Weight of 6 Is 10 and Weight of 7 is 1
- **Types of Weighted Codes**
  - 8421 Code
  - 2421 Code

NON WEIGHTED CODE

- Non Weighted code are not assign with any weight to each digit positions, That is each digit position within the number is not assigned a fixed value
- **Types of Weighted Codes**

  - Excess 3 code
  - BCD code
  - Gray Codes

REFLECTIVE CODE (SELF COMPLEMENT CODE)

- A Code is said to be reflective when the code 9 is the complement of the code 0, for 8 is complement for 1, 7 for 2, 6 for 3 , 5 for 4….
- **Types of Reflected Codes**

  - 2421 Code
  - Excess 3 Code

BCD Codes

- In this code each decimal digit is represented by a 4-bit binary number. BCD is a way to express each of the decimal digits with a binary code. In the BCD, with four bits we can represent sixteen numbers (0000 to 1111). But in BCD code only first ten of these are used (0000 to 1001). The remaining six code combinations i.e. 1010 to 1111 are invalid in BCD

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

BCD Codes

In this code each digit is represented by
4 bit Binary Number (8421)

Excess 3 Codes

- Excess 3 code is a modified form or BCD number
- Excess 3 can be derived from natural BCD Code by adding 3 to each coded number
- For Example, Decimal 12 can be represented in BCD as 0001 0010 Now adding 3 to each digit we get excess 3 code as 0100 0101
- It is a non weighed code

Decimal digit	Excess-3 code			
0	0	0	1	1
1	0	1	0	0
2	0	1	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0
8	1	0	1	1
9	1	1	0	0

- In Excess 3 Code we get 9s complement of a number by just complementing each bit. Due to this excess 3 codes are called Self Complementing code / Reflective Code

2421 CODE

- The weights of this code are 2, 4, 2 and 1.
- This code has all positive weights. So, it is a **positively weighted code**.
- It is an **unnatural BCD** code. Sum of weights of unnatural BCD codes is equal to 9.
- It is a **self-complementing** code. Self-complementing codes provide the 9's complement of a decimal number, just by interchanging 1's and 0's in its equivalent 2421 representation.

2421 : SELF COMPLEMENTING CODE

Decimal	8421 BCD	2421 BCD
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	1011
6	0110	1100
7	0111	1101
8	1000	1110
9	1001	1111

OTHER CODES

INDEX

Gray Codes

- Gray code is a special case in **unit distance code.**
- In Unit distance code, bit patterns for two consecutive numbers differ only in **one bit position**
- These codes are also called cyclic codes

Gray Code

- As show in table two adjacent code groups differ only in one bit position
- The gray code is also called Reflected Code
- Two least significant bit for 4 through 7 are the mirror images of those for 0 through 3 (Similarly 8 -15, 0-7)

Decimal code	Gray code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

Binary to Gray Code Conversion

- The MSB (Most Significant Bit) of the gray code will be exactly equal to the first bit of the given binary number.
- The second bit of the code will be exclusive-or (XOR) of the first and second bit of the given binary number, i.e if both the bits are same the result will be 0 and if they are different the result will be 1.
- The third bit of gray code will be equal to the exclusive-or (XOR) of the second and third bit of the given binary number. Thus the binary to gray code conversion goes on. An example is given below to illustrate these steps.

Gray Code to Binary Conversion

- **Gray code to binary conversion** is again a very simple and easy process. Following steps can make your idea clear on this type of conversions.
- The MSB of the binary number will be equal to the MSB of the given gray code.
- Now if the second gray bit is 0, then the second binary bit will be the same as the previous or the first bit. If the gray bit is 1 the second binary bit will alter. If it was 1 it will be 0 and if it was 0 it will be 1.
- This step is continued for all the bits to do **Gray code to binary conversion**

ALPHA NUMERIC CODES(CHARACTER CODES)

- In order to communicate , we need not only numbers, but also letters and other symbols commonly used as Non Numeric data
- The codes which consist of Letters, digits and various special characters are called **Alpha numeric codes**
- **Types of Alpha Numeric Code**
  - ASCII CODE
  - EBCDIC CODE

ASCII CODES

- The **American Standard for Information Interchange** is a standard 7 bit code and it is a character encoding standard for electronic communication
- The standard ASCII Character set consist of 128 decimal numbers $(2^7 = 128)$
- ASCII Consist of Letters, Numbers and Special Characters
- Computers can only understand numbers, So an ASCII Code is an numerical representation of various characters
- Most modern character-encoding schemes are based on ASCII, although they support many additional characters.

LSBs	MSBs							
	000 (0)	001 (1)	010 (2)	011 (3)	100 (4)	101 (5)	110 (6)	111 (7)
0000 (0)	NUL	DLE	SP	0	@	P		p
0001 (1)	SOH	DC_1	!	1	A	Q	a	q
0010 (2)	STX	DC_2	"	2	B	R	b	r
0011 (3)	ETX	DC_3	#	3	C	S	c	s
0100 (4)	EOT	DC_4	$	4	D	T	d	t
0101 (5)	ENQ	NAK	%	5	E	U	e	u
0110 (6)	ACK	SYN	&	6	F	V	f	v
0111 (7)	BEL	ETB	'	7	G	W	g	w
1000 (8)	BS	CAN	(	8	H	X	h	x
1001 (9)	HT	EM	)	9	I	Y	i	y
1010 (A)	LF	SUB	*	:	J	Z	j	z
1011 (B)	VI	ESC	+	;	K	[	k	{
1100 (C)	FF	FS	,	<	L	\	l	|
1101 (D)	CR	GS	-	=	M	J	m	}
1110 (E)	SO	RS	.	>	N	↑	n	~
1111 (F)	SI	US	/	?	O	←	o	DEL

Definition of control abbreviations :

ACK	Acknowledge	FS	Form separator
BEL	Bell	GS	Group separator
BS	Backspace	HT	Horizontal tab
CAN	Cancel	LF	Line feed
CR	Carriage return	NAK	Negative acknowledge
DC_1-DC_4	Direct control	NUL	Null
DEL	Delete idle	RS	Record separator
DLE	Data link escape	SI	Shift in
EM	End of Medium	SO	Shift out
ENQ	Enquiry	SOH	Start of heading
EOT	End of transmission	STX	Start text
ESC	Escape	SUB	Substitute
ETB	End of transmission block	SYN	Synchronous idle
ETX	End text	US	Unit separator
FF	Form feed	VT	Vertical tab

Note : The hexadecimal digit representing each bit pattern is shown in parentheses.

ASCII CODE

A-Z	• 65-90
a-z	• 97-122
SPACE	• 32
.	• 46

EXAMPLE

Computer

A	B	C	D	E	F	G	H	I	J
1	2	3	4	5	6	7	8	9	10

K	L	M	N	O	P	Q	R	S	T
11	12	13	14	15	16	17	18	19	20

U	V	W	X	Y	Z
21	22	23	24	25	26

C- 3 + 64 = 67
o- 15 + 96 =111
m- 13 + 96 =109
p- 16 +96 =112
u- 21 + 96 = 117
t- 20 + 96 =116
e- 5 + 96 =101
r – 18 +96 =114

A-Z	65-90
a-z	97-122
SPACE	32
.	46

EXAMPLE : SHORTCUT

E	J	O	T	Y
5	10	15	20	25

How Old Are You

A
- 64+1

a
- 96+1

EBCDIC CODE

- The Extended Binary Coded Decimal Interchange Code is a standard 8 bit code Developed by IBM
- The standard EBCDIC Character set consist of 256 decimal numbers ($2^8 = 256$)
- This code is rarely used today. Used for IBM Mainframes only

MODULE 1
(TOPIC-12)

HAMMING CODES

INDEX

https://youtu.be/xfj2UMkfT_g YouTube

ERROR DETECTING AND CORRECTING CODE (HAMMING CODE/PARITY CODE)

- When the digital information in the binary form is transmitted from one system to another system an error may occur. This means signal is corresponding 0 may change to 1 or vice versa due to noise
- To maintain data integrity between transmitter and receiver, extra bit or more than one bit added in the data
- These extra bit allow the detection or correction of errors in the data
- Codes which allow only error detection are called error detecting codes
- Codes which allow error detection as well as correction are called error detecting and correcting codes

MODULE 1
(TOPIC-13)

INTRODUCTION TO VERILOG PROGRAMMING

INDEX

https://youtu.be/TQNl363ZrAI YouTube

Verilog Programming

- Verilog is a HARDWARE DESCRIPTION LANGUAGE (HDL).
- It is a language used for describing a digital system like a network switch or a microprocessor or a memory or a flip−flop.
- It means, by using a HDL we can describe any digital hardware at any level.
- Designs, which are described in HDL are independent of technology, very easy for designing and debugging, and are normally more useful than schematics, particularly for large circuits

Verilog Operators

- Arithmetic
- Logical
- Relational
- Equity
- Bit Wise
- Shift

1.Arithmetic Operations

Operator Symbol	Operations Performed
*	Multiply
/	Divide
+	Add
-	Subtract
%	Modulus
**	Power (Exponent)

2.Logic Operations

Operator Symbol	Operations Performed
!	Logical Negation
&&	Logical AND
||	Logical OR

3.Relational Operator

Operator Symbol	Operations Performed
>	Greater than
<	Less than
>=	Greater than or equal
>=	Less than or equal

4.EquityOperator

Operator Symbol	Operations Performed
==	Equality
!=	Inequality

5.Bitwise Operations

Operator Symbol	Operations Performed
~	Bitwise Negation
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
~^ / ^~	Bitwise XNOR

6.Shift Operations

Operator Symbol	Operations Performed
>>	Right Shift
<<	Left Shift
>>>	Arithmetic Right Shift
<<<	Arithmetic Left Shift

Gate Flow Modeling

- Designing a complex circuit using Basic Logic Gates is the goal of Gate Level Modeling
- We specify the gates of the circuit in our code. Verilog supports describing circuits using basic logic gates as predefined primitives.
- These primitives are instantiated like modules except that they are predefined in Verilog and do not need a module definition.

Data flow modeling

- Compared to gate-level modeling, dataflow modeling is a higher level of abstraction.
- Data Flow describes the circuits by their function rather than by their gate structure. That becomes handy because gate-level modeling becomes very complicated for large circuits because let's face it, a digital circuit with a bunch of gates can seem quite daunting.
- Hence, dataflow modeling is a very important way of implementing the design.
- We require the boolean logic equation and **Continous Assignment Statement** to build the designs. The continuous assignments are made using the keyword assign.

PREPARED BY

Jibin EP
Assistant Professor
Department of Electronics and Communication
Eranad Knowledge City Technical Campus Manjeri
Contact :9633908979, 70121716607
Email : epjibin@gmail.com, jibin@ekc.edu.in

https://www.youtube.com/ShastraTechnicalInstitute

115