4- Binary Codes

This section describes various popular Binary Code formats. In the coding, when numbers, letters or words are represented by a specific group of symbols, it is said that the number, letter or word is being encoded. The group of symbols is called as a code. The digital data is represented, stored and transmitted as group of binary bits. This group is also called as binary code. The binary code is represented by the number as well as alphanumeric letter .

+ Advantages of Binary Code

Following is the list of advantages that binary code offers.

- Binary codes are suitable for the computer applications.
- Binary codes are suitable for the digital communications.
- Binary codes make the analysis and designing of digital circuits if we use the binary codes.
- Since only 0 & 1 are being used, implementation becomes easy.

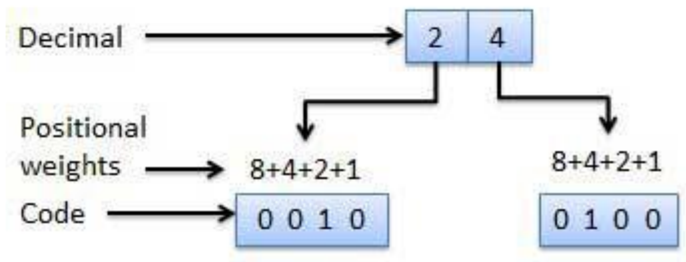
+ Classification of binary codes

The codes are broadly categorized into following six categories.

- Weighted Codes
- Non-Weighted Codes
- Binary Coded Decimal Code
- Alphanumeric Codes
- Error Detecting Codes
- Error Correcting Codes

+ Weighted Codes

Weighted binary codes are those binary codes which obey the positional weight principle. Each position of the number represents a specific weight. Several systems of the codes are used to express the decimal digits 0 through 9. In these codes each decimal digit is represented by a group of four bits.



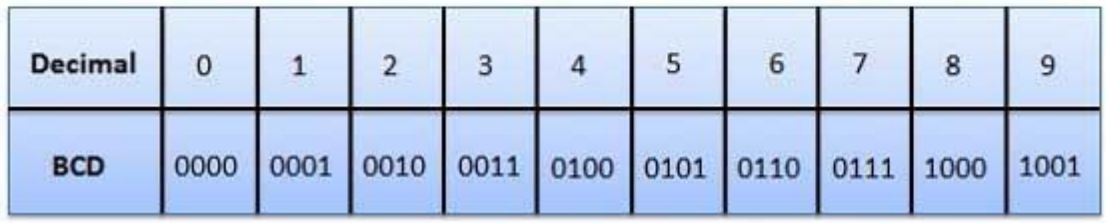
+Non-Weighted Codes

In this type of binary codes, the positional weights are not assigned. The examples of non weighted codes are Excess-3 code and Gray code.

- Binary Coded Decimal (BCD) code

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).

Note: *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

Advantages of BCD Code

- It is very similar to decimal system.
- We need to remember binary equivalent of decimal numbers 0 to 9 only.

Disadvantages of BCD Code

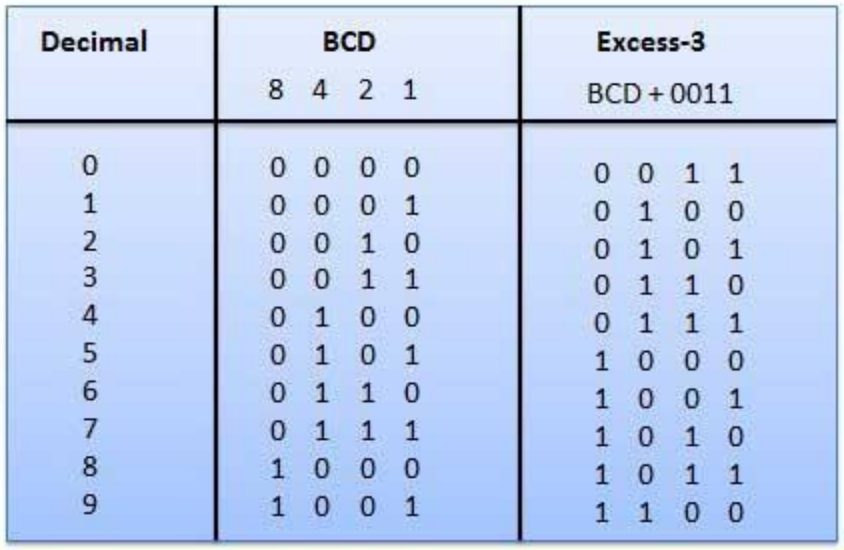
- The addition and subtraction of BCD have different rules.
- The BCD arithmetic is little more complicated.
- BCD needs more number of bits than binary to represent the decimal number. So BCD is less efficient than binary.

-Excess-3 Code

The Excess-3 code is also called as XS-3 code. It is non-weighted code used to express decimal numbers. The Excess-3 code words are derived from the 8421 BCD code words adding (0011)2 or (3)10 to each code word in 8421. The excess-3 codes are obtained as follows:

$$\begin{tikzcd}\text{Decimal Number} \arrow[r] & \text{8421 BCD} \arrow[r, "\text{Add}", "0011"'] & \text{Excess-3}\end{tikzcd}$$

Example :



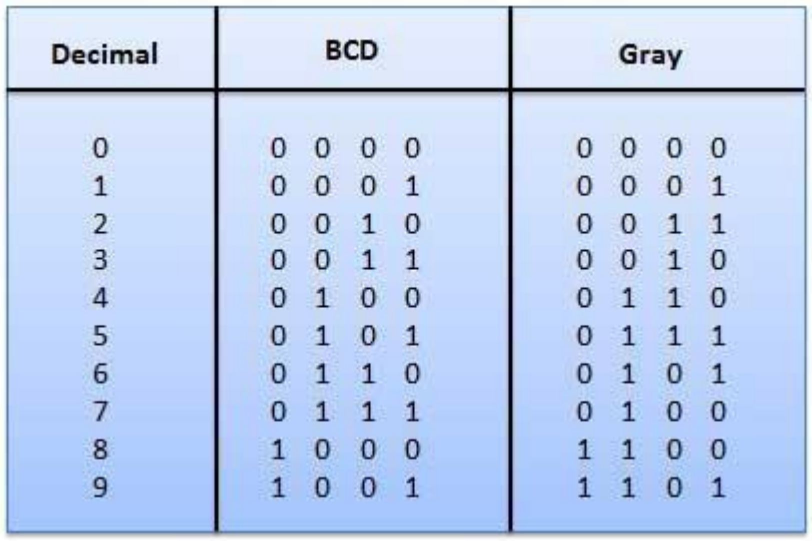
Decimal	BCD	Excess-3
	8 4 2 1	BCD + 0011
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

- Gray Code

It is the non-weighted code and it is not arithmetic codes. That means there are no specific weights assigned to the bit position. It has a very special feature that has only one bit will change, each time the decimal number is incremented as shown in fig. As only one bit changes at a time, the gray code is called as a unit distance code. The gray code is a cyclic code. Gray code cannot be used for arithmetic operation.

*Applications of Gray Code

- Gray code is popularly used in the shaft position encoders.
- A shaft position encoder produces a code word which represents the angular position of the shaft.



Decimal	BCD	Gray
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1

5- Codes Conversion

This section describes conversion of various digital code formats to one another.
there are many methods or techniques which can be used to convert code from one format to another. We'll demonstrate here the following:

- Binary to BCD Conversion
- BCD to Binary Conversion
- BCD to Excess-3
- Excess-3 to BCD
- Binary to Gray
- Gary to binary

+ Binary to BCD Conversion

Steps

Step 1 -- Convert the binary number to decimal.
Step 2 -- Convert decimal number to BCD.

Example : Convert (11101)2 to BCD.

Step 1 - Convert to Decimal

Binary Number: 11101_2
Calculating Decimal Equivalent:

Step 1: 111012= $((1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0))_{10}$
Step 2: 111012= $(16 + 8 + 4 + 0 + 1)_{10}$

Step 3: 111012 =29_{10}

Binary Number: 11101_2 = **Decimal Number**: 29_{10}

Step 2 – Convert to BCD

Decimal Number: 29_{10}

Calculating BCD Equivalent: Convert each digit into groups of four binary digits equivalent.

Step	Decimal	Number Conversion
Step 1	29_{10}	00102 1001_2
Step 2	29_{10}	00101001_{BCD}

Result = $(11101)_2$ = $(00101001)_{BCD}$

+BCD to Binary Conversion

Steps

- **Step 1** -- Convert the BCD number to decimal.
- **Step 2** -- Convert decimal to binary.

Example:

Convert (00101001)BCD to Binary.

Step 1 – Convert to Decimal

BCD Number: (00101001)BCD

Calculating Decimal Equivalent: Convert each four digit into a group and get decimal equivalent or each group.

Step	BCD Number	Conversion
Step 1	$(00101001)_{BCD}$	$0010_2 1001_2$
Step 2	$(00101001)_{BCD}$	$2_{10}9_{10}$
Step 3	$(00101001)_{BCD}$	29_{10}

BCD Number: $(00101001)_{BCD}$ = **Decimal Number**: 2

Step 2 – Convert to Binary

Used long division method for decimal to binary conversion.

Decimal Number: 2910

Calculating Binary Equivalent:

Step	Operation	Result	Remainder
Step 1	29 / 2	14	1
Step 2	14 / 2	7	0
Step 3	7 / 2	3	1
Step 4	3 / 2	1	1
Step 5	1 / 2	0	1

Note :As mentioned in Steps 2 and 4, the remainders have to be arranged in the reverse order so that the first remainder becomes the least significant digit (LSD) and the last remainder becomes the most significant digit (MSD).

Decimal Number: 2910 = **Binary Number:** 111012

Result : $(00101001)_{BCD} = (11101)_2$

+ BCD to Excess-3

Steps

Step 1 -- Convert BCD to decimal.
Step 2 -- Add (3)10 to this decimal number.
Step 3 -- Convert into binary to get excess-3 code

Example

Convert (1001)BCD to Excess-3.

Step 1 – Convert to Decimal

$(1001)_{BCD} = 93$

Step 2 – Add 3 to Decimal

(9)10 + (3)10 = (12)10

Step 3 – Convert to Excess-3

(12)10 = (1100)2

Result

$(1001)_{BCD} = (1100)_{XS\text{-}3}$

+Excess-3 to BCD Conversion

Steps

Step 1 -- Subtract (0011)2 from each 4 bit of excess-3 digit to obtain the corresponding BCD code.

Example

convert (10011010)XS-3 to BCD.

```
Given XS-3 number = 1 0 0 1 1 0 1 0
   Subtract (0011)2 = 0 0 1 1 0 0 1 1
                      --------------------
                BCD = 0 1 1 0 0 1 1 1
```

Result

$$(10011010)_{XS\text{-}3} = (01100111)_{BCD}$$

+ Gray To Binary Conversion

Steps

Step1: The **MSB** in the left is the **MSB** in binary number. In other word they are The same.

Step2: Add the first digit of the binary number to the second digit in Gary code, the carry is ignored , in other word , take XOR operation between them .

Step3: Generally working from the left to right digit, the n'th digit in the binary number is formed from summing the (n+1)'th digit in the binary number with n'th bit in the Gray code .

Example: Convert the Gary code 11011 to binary number?

$$\begin{tikzcd}
\textbf{Gray code} \arrow[r] & \textbf{1} \arrow[d] & \textbf{1} \arrow[d] & \textbf{0} \arrow[d] & \textbf{1} \arrow[d] & \textbf{1} \arrow[d] \\
\textbf{Binary Number} \arrow[r] & \textbf{1} \arrow[ur] & \textbf{0} \arrow[ur] & \textbf{0} \arrow[ur] & \textbf{1} \arrow[ur] & \textbf{0}
\end{tikzcd}$$

Then $(11011)_G= (10010)_2$

+ Binary To Gray Conversion

Steps:

Step1: The **MSB** digit in Gary code is the same as corresponding digit in the binary number.

Step2: going from left to right , add each adjacent pair of binary digit to get the next Gray digits , regardless carries .

Example: Convert the following binary number 100110 to Gray code?

Binary Number → 1 0 0 1 1 0

Gray Code → 1 1 0 1 0 1

Then $(100110)2 =(110101)_G$

Home work :

1- convert the BCD code $(1001010100000100)_{BCD}$ decimal number

2- $(12)_{10}$ +$(9)_{10}$ using BCD code

3- convert the gray code 11011 to binary number?

4- Covert (101110101)Gary = (?)2

5- Logic Gates

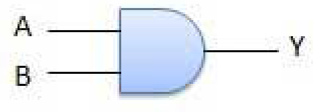
This section describes various types of logic gates. Logic gates are the basic building blocks of any digital system. It is an electronic circuit havingone or more than one input and only one output. The relationship between the input and the output is based on certain **logic**. Based on this logic gates are named as AND gate, OR gate, NOT gate etc.

1-AND Gate

A circuit which performs an AND operation is shown in figure. It has n input (n >= 2) and one output.

Y	=	A AND B AND C .... N
Y	=	A.B.C .... N
Y	=	ABC .... N

- **Logic Diagram**



- **Truth Table**

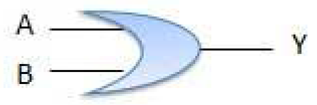
Inputs		Output
A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

2- OR Gate

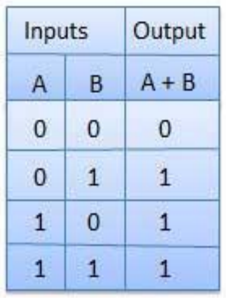
A circuit which performs an OR operation is shown in figure . It has n input (n >= 2) and one output.

Y	=	A OR B OR C .... N
Y	=	A + B + C .... N

- **Logic Diagram**



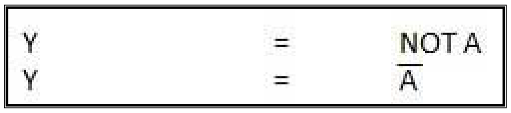
- **Truth Table**



Inputs		Output
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

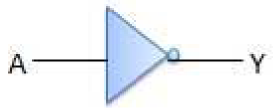
3-NOT Gate

NOT gate is also known as **Inverter**. It has one input A and one output Y.

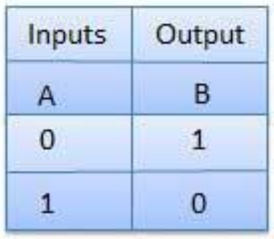


Y	=	NOT A
Y	=	$\overline{A}$

- **Logic Diagram**



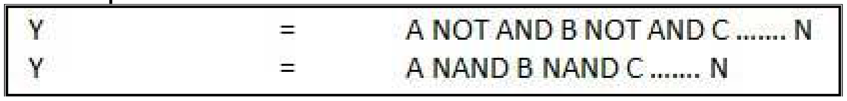
- **Truth Table**



Inputs	Output
A	B
0	1
1	0

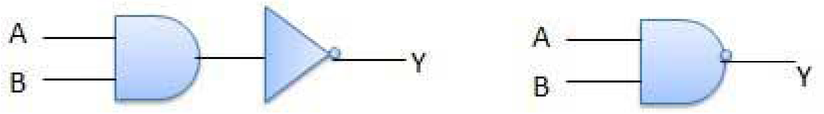
4-NAND Gate

A NOT-AND operation is known as NAND operation. It has n input (n >= 2) and one output.

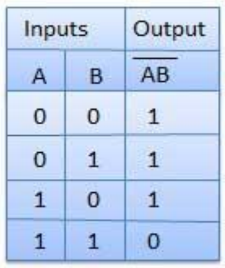


Y	=	A NOT AND B NOT AND C .... N
Y	=	A NAND B NAND C .... N

- **Logic Diagram**



- **Truth Table**



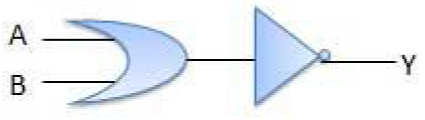
Inputs		Output
A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

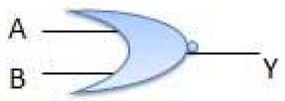
5-NOR Gate

A NOT-OR operation is known as NOR operation. It has n input (n >= 2) and one output.

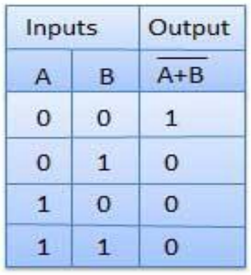
Y	=	A NOT OR B NOT OR C .... N
Y	=	A NOR B NOR C .... N

- **Logic Diagram**





- **Truth Table**



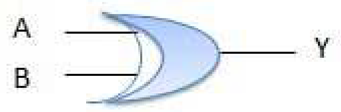
Inputs		Output
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

6- XOR Gate

XOR or Ex-OR gate is a special type of gate. It can be used in the half adder, full adder and subtractor. The exclusive-OR gate is abbreviated as EX-OR gate or sometime as X-OR gate. It has n input (n >= 2) and one output.

Y	=	A XOR B XOR C .... N
Y	=	$A \oplus B \oplus C$ .... N
Y	=	$A\overline{B} + \overline{A}B$

- **Logic Diagram**



- **Truth Table**

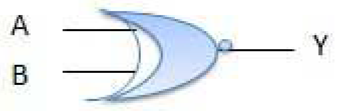
Inputs		Output
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

7- XNOR Gate

XNOR gate is a special type of gate. It can be used in the half adder, full adder and subtractor. The exclusive-NOR gate is abbreviated as EX-NOR gate or sometime as X-NOR gate. It has n input (n >= 2) and one output.

Y	=	A XOR B XOR C .... N
Y	=	$A \odot B \odot C$ .... N
Y	=	$\overline{A}\,\overline{B} + AB$

- **Logic Diagram**



- **Truth Table**

Inputs		Output
A	B	$A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1