LECTURE 5

Binary Codes

Binary Codes

- Digital systems use signals that have two distinct values and circuit elements that have two stable states.

- There is a direct analogy among binary signals, binary circuit elements, and binary digits.

- A binary number of *n digits, for example, may be represented* by *n binary circuit elements, each having an output signal equivalent to 0 or 1.*

Binary Codes

- Digital systems represent and manipulate not only binary numbers, but also many other discrete elements of information.

- Any discrete element of information that is distinct among a group of quantities can be represented with a binary code (i.e., a pattern of 0's and 1's).

- The codes must be in binary because, in today's technology, only circuits that represent and manipulate patterns of 0's and 1's can be manufactured economically for use in computers.

Binary Codes

- However, it must be realized that binary codes merely change the symbols, not the meaning of the elements of information that they represent.

- If we inspect the bits of a computer at random, we will find that most of the time they represent some type of coded information rather than binary numbers.

- An *n-bit binary code is a group of n bits that assumes up to* r^n *distinct combinations* of 1's and 0's, with each combination representing one element of the set that is being coded.

Binary Codes

- A set of four elements can be coded with two bits, with each element assigned one of the following bit combinations: 00, 01, 10, 11.

- A set of eight elements requires a three-bit code and a set of 16 elements requires a four-bit code.

- The bit combination of an *n-bit code is determined from the count in binary from 0 to* $2^n - 1$. *Each element* must be assigned a unique binary bit combination, and no two elements can have the same value; otherwise, the code assignment will be ambiguous.

Binary Coded Decimal (BCD)

Binary-Coded Decimal (BCD)

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

A decimal number in BCD is the same as its equivalent binary number only when the number is between 0 and 9.

The binary combinations 1010 through 1111 are not used and have no meaning in BCD.

Binary Coded Decimal (BCD)

- A number with *k decimal* digits will require 4*k bits in BCD. Decimal 396 is represented in BCD with 12 bits as* 0011 1001 0110, with **each group of 4 bits representing one decimal digit.**

- **A decimal** number in BCD is the same as its equivalent binary number only when the number is between 0 and 9. A BCD number greater than 10 looks different from its equivalent binary number, even though both contain 1's and 0's.

- Moreover, **the binary combinations 1010 through 1111 are not used and have no meaning in BCD. Consider decimal** 185 and its corresponding value in BCD and binary:

$$(185)_{10} = (0001\ 1000\ 0101)_{\text{BCD}} = (10111001)_2$$

BCD Addition

4	0100	4	0100	8	1000
+5	+0101	+8	+1000	+9	1001
9	1001	12	1100	17	10001
			+0110		+0110
			10010		10111

BCD Addition

BCD	1	1		
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary sum	0111	10000	1010	
Add 6		0110	0110	
BCD sum	0111	0110	0000	760

Other Binary Codes

Four Different Binary Codes for the Decimal Digits

Decimal Digit	BCD 8421	2421	Excess-3	8, 4, −2, −1
0	0000	0000	0011	0000
1	0001	0001	0100	0111
2	0010	0010	0101	0110
3	0011	0011	0110	0101
4	0100	0100	0111	0100
5	0101	1011	1000	1011
6	0110	1100	1001	1010
7	0111	1101	1010	1001
8	1000	1110	1011	1000
9	1001	1111	1100	1111
	1010	0101	0000	0001
Unused	1011	0110	0001	0010
bit	1100	0111	0010	0011
combi-	1101	1000	1101	1100
nations	1110	1001	1110	1101
	1111	1010	1111	1110

Binary Codes

- The 2421 and the excess-3 codes are examples of self-complementing codes. Such codes have the property that the 9's complement of a decimal number is obtained directly by changing 1's to 0's and 0's to 1's (i.e., by complementing each bit in the pattern).

- For example, decimal 395 is represented in the excess-3 code as **0110 1100 1000**. The 9's complement of 604 is represented as **1001 0011 0111**, which is obtained simply by complementing each bit of the code (as with the 1's complement of binary numbers).

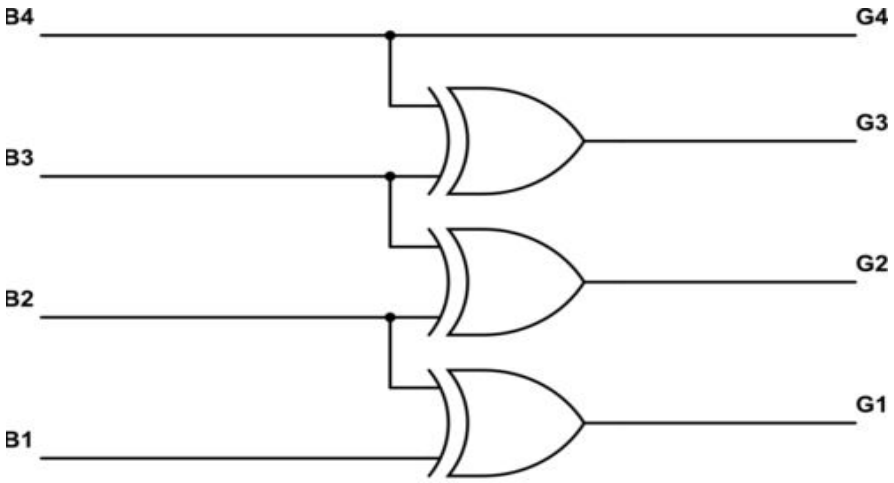
- The excess-3 code has been used in some older computers because of its self complementing property.

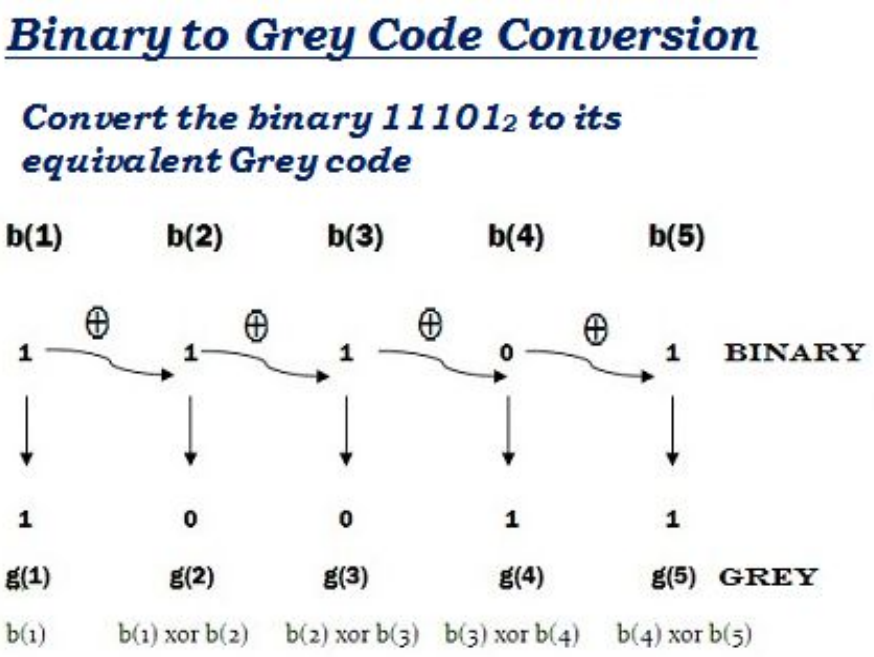
- **Excess-3 is an unweighted code in which each coded combination is obtained from the corresponding binary value plus 3.**

- **Note that the BCD** code is not **self-complementing**.

Gray code

Gray code – also known as **Cyclic Code**, **Reflected Binary Code** (RBC), **Reflected Binary** (RB) or **Grey code** – is defined as an ordering of the binary number system such that each incremental value can only differ by one bit. In gray code, while traversing from one step to another step only one bit in the code group changes. That is to say that two adjacent code numbers differ from each other by only one bit.





Gray code

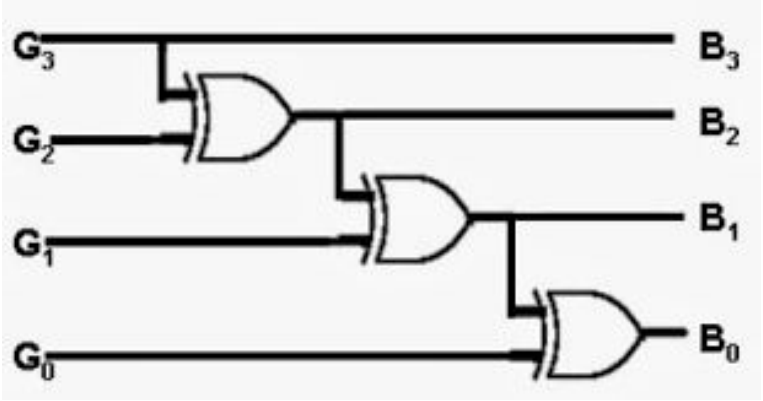
- The Gray code is used in applications in which the normal sequence of binary numbers generated by the hardware may produce an error or ambiguity during the transition from one number to the next.

- If binary numbers are used, a change, for example, from 0111 to 1000 may produce an intermediate erroneous number 1001 if the value of the rightmost bit takes longer to change than do the values of the other three bits.

- This could have serious consequences for the machine using the information. The Gray code eliminates this problem, since only one bit changes its value during any transition between two numbers.

BINARY INPUT				GRAY CODE OUTPUT			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Gray to Binary Conversion



Grey Code to Binary Conversion

Convert the Grey code 1010 to its equivalent Binary

g(3)	g(2)	g(1)	g(0)	
1	0	1	0	GREY
b(3)	b(2)	b(1)	b(0)	
1	1	0	0	BINARY

i.e $b(3) = g(3)$

$b(2) = b(3) \oplus g(2)$

$b(1) = b(2) \oplus g(1)$

$b(0) = b(1) \oplus g(0)$

GRAY CODE INPUT				BINARY OUTPUT			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Parity Bit

The parity generating technique is one of the most widely used error detection techniques for the data transmission. In digital systems, when binary data is transmitted and processed , data may be subjected to noise so that such noise can alter 0s (of data bits) to 1s and 1s to 0s. **parity bit** is added to the word containing data in order to make number of 1s either even or odd. Thus it is used to detect errors , during the transmission of binary data .

Parity Generator

It is combinational circuit that accepts an n-1 bit stream data and generates the additional bit that is to be transmitted with the bit stream. This additional or extra bit is termed as a parity bit.

In **even parity** bit scheme, the parity bit is '**0**' if there are **even number of 1s** in the data stream and the parity bit is '**1**' if there are **odd number of 1s** in the data stream.

In **odd parity** bit scheme, the parity bit is '**1**' if there are **even number of 1**s in the data stream and the parity bit is '**0**' if there are **odd number of 1s** in the data stream. Let us discuss both even and odd parity generators.

Even Parity Generator

Let us assume that a 3-bit message is to be transmitted with an even parity bit.

Let the three inputs A, B and C are applied to the circuits and output bit is the parity bit P.

The total number of 1s must be even, to generate the even parity bit P. The figure shows the truth table of even parity generator in which 1 is placed as parity bit in order to make all 1s as even when the number of 1s in the truth table is odd.

3-bit message			Even parity bit generator (P)
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Odd Parity Generator

Let us consider that the 3-bit data is to be transmitted with an odd parity bit. The three inputs are A, B and C and P is the output parity bit. The total number of bits must be odd in order to generate the odd parity bit.

In the given truth table below, 1 is placed in the parity bit in order to make the total number of bits odd when the total number of 1s in the truth table is even.

3-bit message			Odd parity bit generator (P)
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Some Additional Problems on Binary Codes:

- Represent the unsigned decimal numbers 791 and 658 in BCD, and then show the steps necessary to form their sum.
- Represent the decimal number 6,248 in (a) BCD, (b) excess-3 code, (c) 2421 code, and (d) a 6311 code.
- The state of a 12-bit register is 100010010111. What is its content if it represents

  (a) Three decimal digits in BCD?

  (b) Three decimal digits in the excess-3 code?

  (c) Three decimal digits in the 84-2-1 code?

  (d) A binary number?

Contd.

- Express the decimal 5280 in Excess-3 code.
- What is the decimal equivalent of this excess-3 code:

  0110 1001 1100 0111
- Convert the Gray Code 1110 to its BCD equivalent?
- Generate the even parity bit table for a 4-bit message.
- Generate an odd parity bit table for a 2-bit message.

Contd.

- Draw truth table for 3-bit Binary to Gray code and Gray Code to Binary code.
- Do the BCD addition of the followings BCD nos:

  a. 86 and 39

  b. 99 and 0

  c. 123 and 37