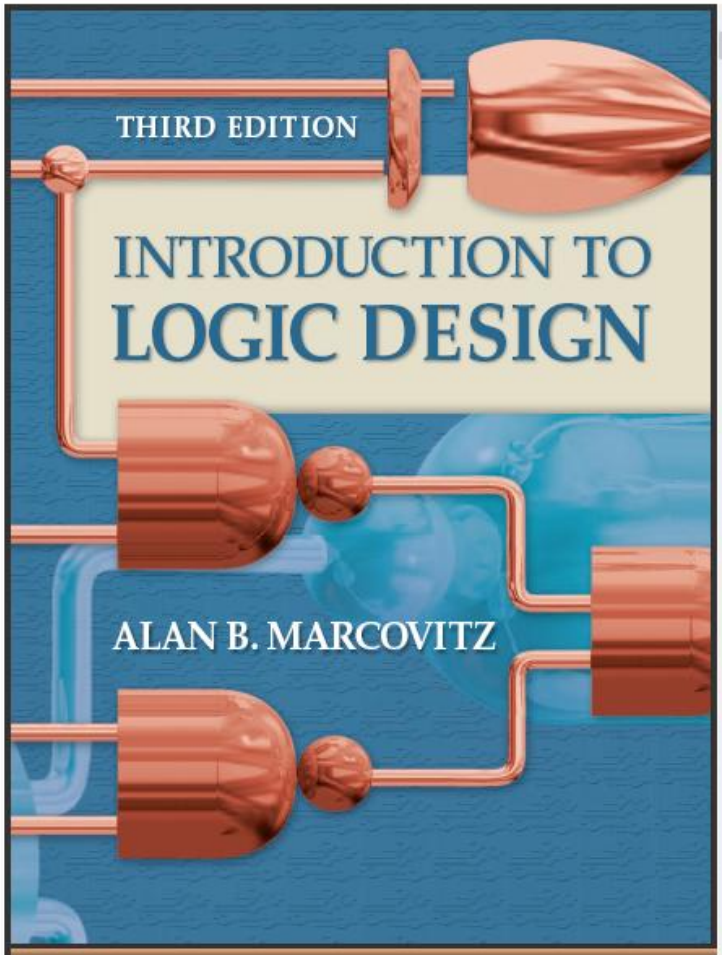
# Digital Logic Design
# Lecture 2

Instructors:
Prof . Hala Zayed
Ass. Prof Mazen M. Selim
Dr Mona A. S. Ali

# Text Book

- Introduction to Logic Design 3rd Edition



# Course Outline & Schedual

| W # | Topics | Textbook Sections |
|---|---|---|
| 1 | Number systems | Ch 1 sec1-5 (Dr Mona) |
| 2 | Digital codes | Ch 1 reminder (Dr Mona) |
| 3 | Logic Gates | Ch 2 all (Dr Mona) |
| 4 | Boolen Algebra | Ch 3 (Dr Mona) |
| 5 | Switching functions and canonical forms & **Quiz 1** | Ch 3 (Dr Mona) |
| 6 | Simplification using Karnaugh maps | Ch 4 (Dr Mona) |
| 7 | Digital combinational logic (decoders, encoders, multiplexers, demultiplexers) | Ch5 (Ass. Prof Mazen) |
| 8 | Digital combinational logic (adders and subtractors) | Ch5 (Ass. Prof Mazen) |
| 9 | Digital combinational logic (comparators, multipliers, dividers) | Ch 5(Ass. Prof Mazen) |
| 10 | Sequential logic and flip flop (part I) **Quiz 2** | Ch 6 (Prof Hala) |
| 11 | Sequential logic and flip flop (part 2) | Ch6 (Prof Hala) |
| 12 | Analysis of sequential circuits | Ch7 (Prof Hala) |
| 13 | Design of sequential circuits | Ch8 (Prof Hala) |
| 14 | Counter circuits | Ch9 (Prof Hala) |

# CHAPTER 1

## NUMBER SYSTEMS AND CODES

# Contents

- ***BINARY CODED DECIMAL (BCD)***
- ***THE ASCII CODE***
- ***The Excess-3 Code***
- ***ERROR-DETECTION CODE***

# Binary Coded decimal

## BCD

- Binary coded decimal (BCD) is a way to express each of the decimal digits with a binary code.
- There are only ten code groups in the BCD system.
- In BCD, 4 bits represent each decimal digit.
- The 8421 code is a type of BCD.
- BCD means that each decimal digit, 0 through 9, is represented by a binary code of four bits.
- The designation 8421 indicates the binary weights uf the four bits $(2^3, 2^2, 2^1, 2^0)$.

# Complements

***Decimal/binary conversion***

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

- **Decimal/binary conversion**
  - With four bits 16 numbers (0000 through 1111) can be represented.
  - In BCD only ten of these are used.
  - The six code combinations that are not used.
  - 1010, 1011, 1100, 1101, 111O, and 1111-are invalid in the 8421 BCD code.

- **To express any decimal number in BCD, simply replace each decimal digit with 4-bit code,**
- Convert each of the following decimal numbers to BCD:
- (a) 35 (b) 98 (c) 170 (d) 2469

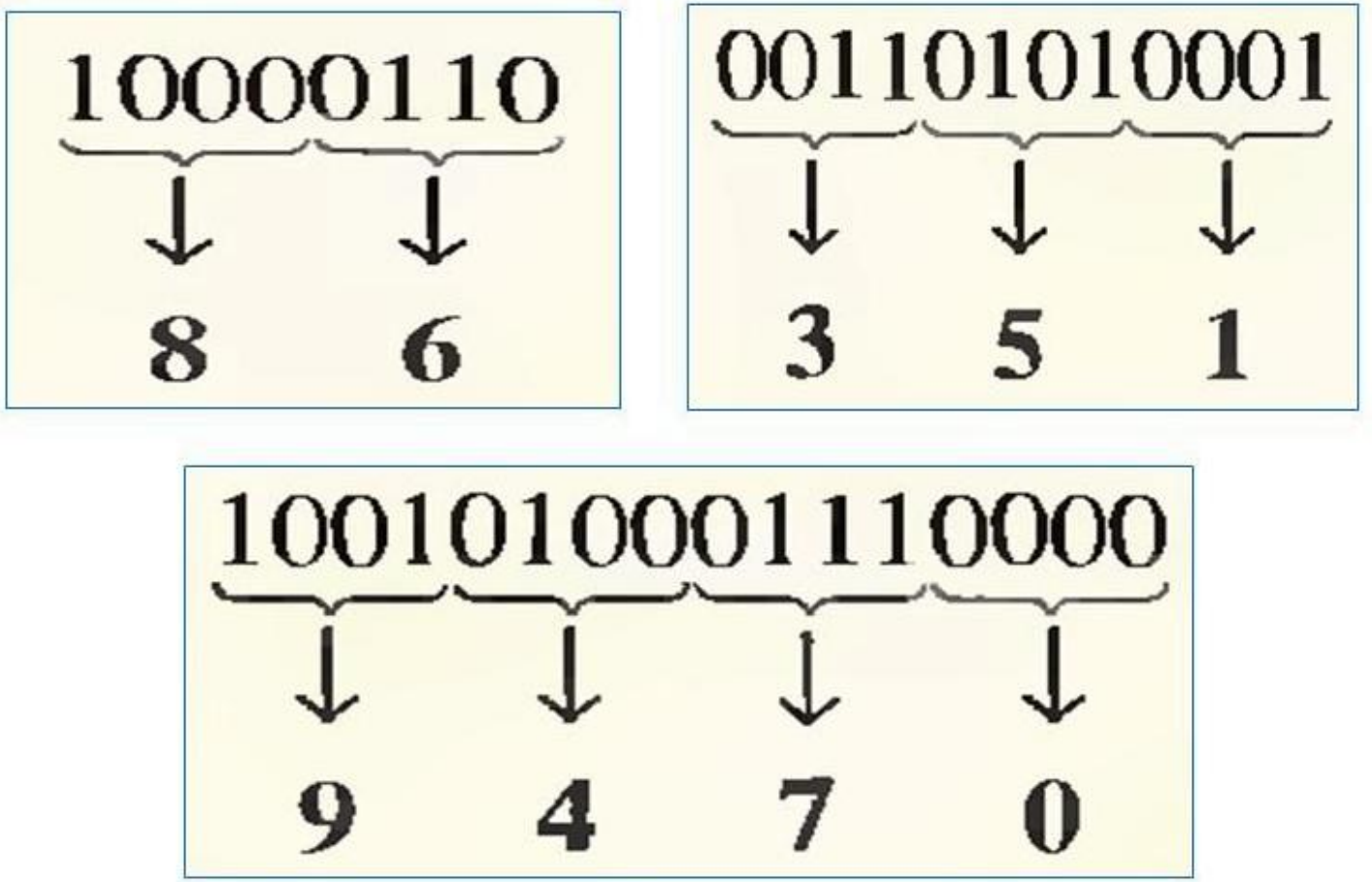
*Solution*

(a) 3 5 → 0011 0101

(b) 9 8 → 1001 1000

(c) 1 7 0 → 0001 0111 0000

(d) 2 4 6 9 → 0010 0100 0110 1001


10000110
8 6
00110101010001
3 5 1
1001010001110000
9 4 7 0

# BCD Addition

## Here is how to add two BCD numbers

- Add the two BCD numbers, using the rules for binary addition.
- If a 4-bit sum is equal to or less than 9, it is a valid BCD number.
- If a 4-bit sum is greater than 9, or if a carry out of the 4-bit group is generated, it is an invalid result. Add 6 (0110) to the 4-bit sum in order to skip the six invalid states and return the code to 8421. If a carry results when 6 is added, simply add the carry to the next 4-bit group.

- The decimal number additions are shown for comparison.

| | | |
|---|---|---|
| (a) | 0011 | 3 |
| | + 0100 | + 4 |
| | **0111** | 7 |

| | | | |
|---|---|---|---|
| (c) | 1000 | 0110 | 86 |
| | + 0001 | 0011 | + 13 |
| | **1001** | **1001** | 99 |

| | | | |
|---|---|---|---|
| (b) | 0010 | 0011 | 23 |
| | + 0001 | 0101 | + 15 |
| | **0011** | **1000** | 38 |

| | | | | |
|---|---|---|---|---|
| (d) | 0100 | 0101 | 0000 | 450 |
| | + 0100 | 0001 | 0111 | + 417 |
| | **1000** | **0110** | **0111** | 867 |

# Example (invalid cases)

**Example 1**

The decimal number additions are shown for comparison.

| | | | |
|---|---|---|---|
| (a) | 1001 | | 9 |
| | + 0100 | | +4 |
| | 1101 | Invalid BCD number (>9) | 13 |
| | + 0110 | Add 6 | |
| 0001 | 0011 | Valid BCD number | |
| ↓ | ↓ | | |
| 1 | 3 | | |

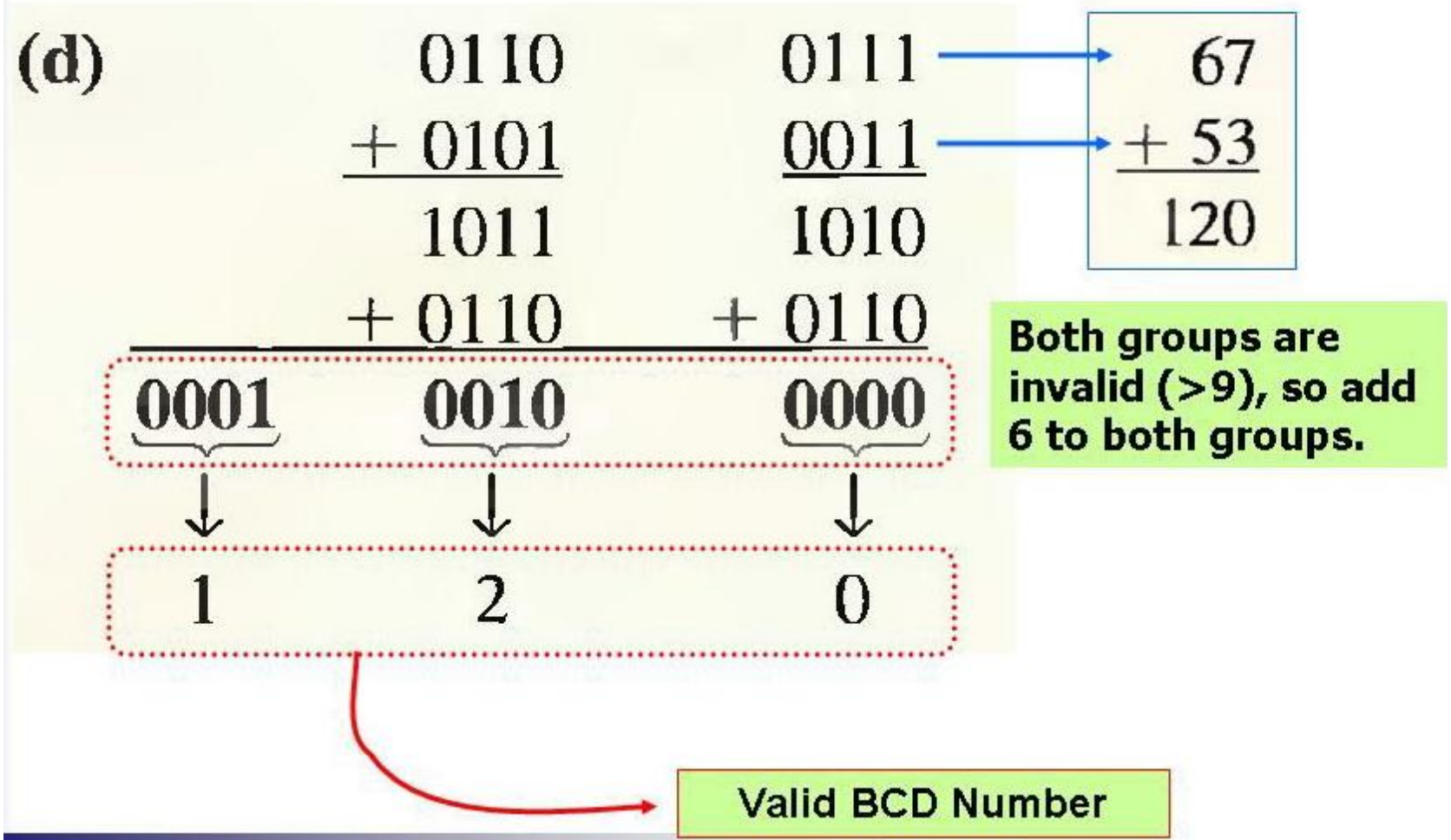
## Example 2

| | | | |
|---|---|---|---|
| (b) | 1001 | | 9 |
| | + 1001 | | + 9 |
| | 1 0010 | Invalid because of carry | 18 |
| | + 0110 | Add 6 | |
| | 0001 1000 | Valid BCD number | |
| | ↓ ↓ | | |
| | 1 8 | | |

## Example 3

| | | | | |
|---|---|---|---|---|
| (c) | 0001 | 0110 | | 16 |
| | + 0001 | 0101 | | + 15 |
| | 0010 | 1011 | Right group is invalid (>9), left group is valid. | 31 |
| | | + 0110 | Add 6 to invalid code. Add carry, 0001, to next group. | |
| | **0011** | **0001** | Valid BCD number | |
| | ↓ | ↓ | | |
| | 3 | 1 | | |

## Example 4


(d)
0110
+ 0101
1011
+ 0110
0001
0010
1
2
0111
0011
1010
+ 0110
0000
0
67
+ 53
120
Both groups are invalid (>9), so add 6 to both groups.
Valid BCD Number

# Digital codes

BCD code is a numeric code.

Some other codes are alphanumeric.

- they are used to represent numbers, letters, symbols, and instructions.
- That is they are used to represent letters, numbers, symbols,
  - For example Gray code, ASCII Code.
    - Reading Assignment (Alphanumeric, EBCDIC, Error Detection and Correction code etc)

# ASCII

- ASCII is the abbreviation for American Standard Code for Information Interchange.
- ASCII is a universally accepted alphanumeric code used in most computers and other electronic equipment.
- When you enter a letter. a number, or control command, the corresponding ASCII code goes into the computer.
- ASCII has 128 characters and symbols represented by a 7-bit binary code.
- Actually, ASCII can be considered an 8-bit code with the MSB always O.

# ASCII

- The first thirty-two ASCII characters are nongraphic commands that are never printed or displayed.
- These are used for control purposes.
  - E.g. line of feed, null, start of text, escape etc.
- The other characters are graphic symbols that can be printed includes,
  - Letter of alphabet, 10 decimal digits, punctuation symbols and other commonly used symbols.

| CONTROL CHARACTERS | | | | GRAPHIC SYMBOLS | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NAME | DEC | BINARY | HEX | SYMBOL | DEC | BINARY | HEX | SYMBOL | DEC | BINARY | HEX | SYMBOL | DEC | BINARY | HEX |
| NUL | 0 | 0000000 | 00 | space | 32 | 0100000 | 20 | @ | 64 | 1000000 | 40 | ` | 96 | 1100000 | 60 |
| SOH | 1 | 0000001 | 01 | ! | 33 | 0100001 | 21 | A | 65 | 1000001 | 41 | a | 97 | 1100001 | 61 |
| STX | 2 | 0000010 | 02 | " | 34 | 0100010 | 22 | B | 66 | 1000010 | 42 | b | 98 | 1100010 | 62 |
| ETX | 3 | 0000011 | 03 | # | 35 | 0100011 | 23 | C | 67 | 1000011 | 43 | c | 99 | 1100011 | 63 |
| EOT | 4 | 0000100 | 04 | $ | 36 | 0100100 | 24 | D | 68 | 1000100 | 44 | d | 100 | 1100100 | 64 |
| ENQ | 5 | 0000101 | 05 | % | 37 | 0100101 | 25 | E | 69 | 1000101 | 45 | e | 101 | 1100101 | 65 |
| ACK | 6 | 0000110 | 06 | & | 38 | 0100110 | 26 | F | 70 | 1000110 | 46 | f | 102 | 1100110 | 66 |
| BEL | 7 | 0000111 | 07 | ' | 39 | 0100111 | 27 | G | 71 | 1000111 | 47 | g | 103 | 1100111 | 67 |
| BS | 8 | 0001000 | 08 | ( | 40 | 0101000 | 28 | H | 72 | 1001000 | 48 | h | 104 | 1101000 | 68 |
| HT | 9 | 0001001 | 09 | ) | 41 | 0101001 | 29 | I | 73 | 1001001 | 49 | i | 105 | 1101001 | 69 |
| LF | 10 | 0001010 | 0A | * | 42 | 0101010 | 2A | J | 74 | 1001010 | 4A | j | 106 | 1101010 | 6A |
| VT | 11 | 0001011 | 0B | + | 43 | 0101011 | 2B | K | 75 | 1001011 | 4B | k | 107 | 1101011 | 6B |
| FF | 12 | 0001100 | 0C | , | 44 | 0101100 | 2C | L | 76 | 1001100 | 4C | l | 108 | 1101100 | 6C |
| CR | 13 | 0001101 | 0D | – | 45 | 0101101 | 2D | M | 77 | 1001101 | 4D | m | 109 | 1101101 | 6D |
| SO | 14 | 0001110 | 0E | . | 46 | 0101110 | 2E | N | 78 | 1001110 | 4E | n | 110 | 1101110 | 6E |
| SI | 15 | 0001111 | 0F | / | 47 | 0101111 | 2F | O | 79 | 1001111 | 4F | o | 111 | 1101111 | 6F |
| DLE | 16 | 0010000 | 10 | 0 | 48 | 0110000 | 30 | P | 80 | 1010000 | 50 | p | 112 | 1110000 | 70 |
| DC1 | 17 | 0010001 | 11 | 1 | 49 | 0110001 | 31 | Q | 81 | 1010001 | 51 | q | 113 | 1110001 | 71 |
| DC2 | 18 | 0010010 | 12 | 2 | 50 | 0110010 | 32 | R | 82 | 1010010 | 52 | r | 114 | 1110010 | 72 |
| DC3 | 19 | 0010011 | 13 | 3 | 51 | 0110011 | 33 | S | 83 | 1010011 | 53 | s | 115 | 1110011 | 73 |
| DC4 | 20 | 0010100 | 14 | 4 | 52 | 0110100 | 34 | T | 84 | 1010100 | 54 | t | 116 | 1110100 | 74 |
| NAK | 21 | 0010101 | 15 | 5 | 53 | 0110101 | 35 | U | 85 | 1010101 | 55 | u | 117 | 1110101 | 75 |
| SYN | 22 | 0010110 | 16 | 6 | 54 | 0110110 | 36 | V | 86 | 1010110 | 56 | v | 118 | 1110110 | 76 |
| ETB | 23 | 0010111 | 17 | 7 | 55 | 0110111 | 37 | W | 87 | 1010111 | 57 | w | 119 | 1110111 | 77 |
| CAN | 24 | 0011000 | 18 | 8 | 56 | 0111000 | 38 | X | 88 | 1011000 | 58 | x | 120 | 1111000 | 78 |
| EM | 25 | 0011001 | 19 | 9 | 57 | 0111001 | 39 | Y | 89 | 1011001 | 59 | y | 121 | 1111001 | 79 |
| SUB | 26 | 0011010 | 1A | : | 58 | 0111010 | 3A | Z | 90 | 1011010 | 5A | z | 122 | 1111010 | 7A |
| ESC | 27 | 0011011 | 1B | ; | 59 | 0111011 | 3B | [ | 91 | 1011011 | 5B | { | 123 | 1111011 | 7B |
| FS | 28 | 0011100 | 1C | < | 60 | 0111100 | 3C | \ | 92 | 1011100 | 5C | \| | 124 | 1111100 | 7C |
| GS | 29 | 0011101 | 1D | = | 61 | 0111101 | 3D | ] | 93 | 1011101 | 5D | } | 125 | 1111101 | 7D |
| RS | 30 | 0011110 | 1E | > | 62 | 0111110 | 3E | ^ | 94 | 1011110 | 5E | ~ | 126 | 1111110 | 7E |
| US | 31 | 0011111 | 1F | ? | 63 | 0111111 | 3F | _ | 95 | 1011111 | 5F | Del | 127 | 1111111 | 7F |

# Examples

- The first thirty-two ASCII characters are called the control characters.
- These are used to allow devices such as a computer and printer to communicate with each other when passing information and data.
- The extended ASCII characters are represented by an 8-bit code.
- Extended ASCII has additional 128 characters to present other symbols.
- The extended ASCII character set is presented in the table.

# The Excess-3 Code

- **Excess-3** is a digital code related to BCD that is derived by adding 3 to each decimal digit and then converting the result of that addition to 4-bit binary.
  - 25 = 01011000
    - ADD THREE TO BOTH DIGITS TO BE 5 (0101) AND 8 (1000) THEN PUT THE REPRESENTATION OF ALL DIGITS TOGETHER.
  - 630 = 100101100011

# Self-Complementing Property

- This means that the 1's complement of an excess-3 number is the excess-3 code for the 9's complement of the corresponding decimal number
- How?

# ERROR-DETECTION CODE

- One of the most common ways to achieve error detection is by means of a *parity bit.*
- A parity bit is an extra bit included with a message to make the total number of 1's transmitted either odd or even.

# Parity bit

| Odd parity | | Even parity | |
|---|---|---|---|
| Message | *P* | Message | *P* |
| 0000 | 1 | 0000 | *0* |
| 0001 | 0 | 0001 | 1 |
| 0010 | 0 | 0010 | 1 |
| 0011 | 1 | 0011 | 0 |
| 0100 | 0 | 0100 | 1 |
| 0101 | 1 | 0101 | 0 |
| 0110 | 1 | 0110 | 0 |
| 0111 | 0 | 0111 | 1 |
| 1000 | 0 | 1000 | 1 |
| 1001 | 1 | 1001 | 0 |
| 1010 | 1 | 1010 | 0 |
| 1011 | 0 | 1011 | 1 |
| 1100 | 1 | 1100 | 0 |
| 1101 | 0 | 1101 | 1 |
| 1110 | 0 | 1110 | 1 |
| 1111 | 1 | 1111 | 0 |