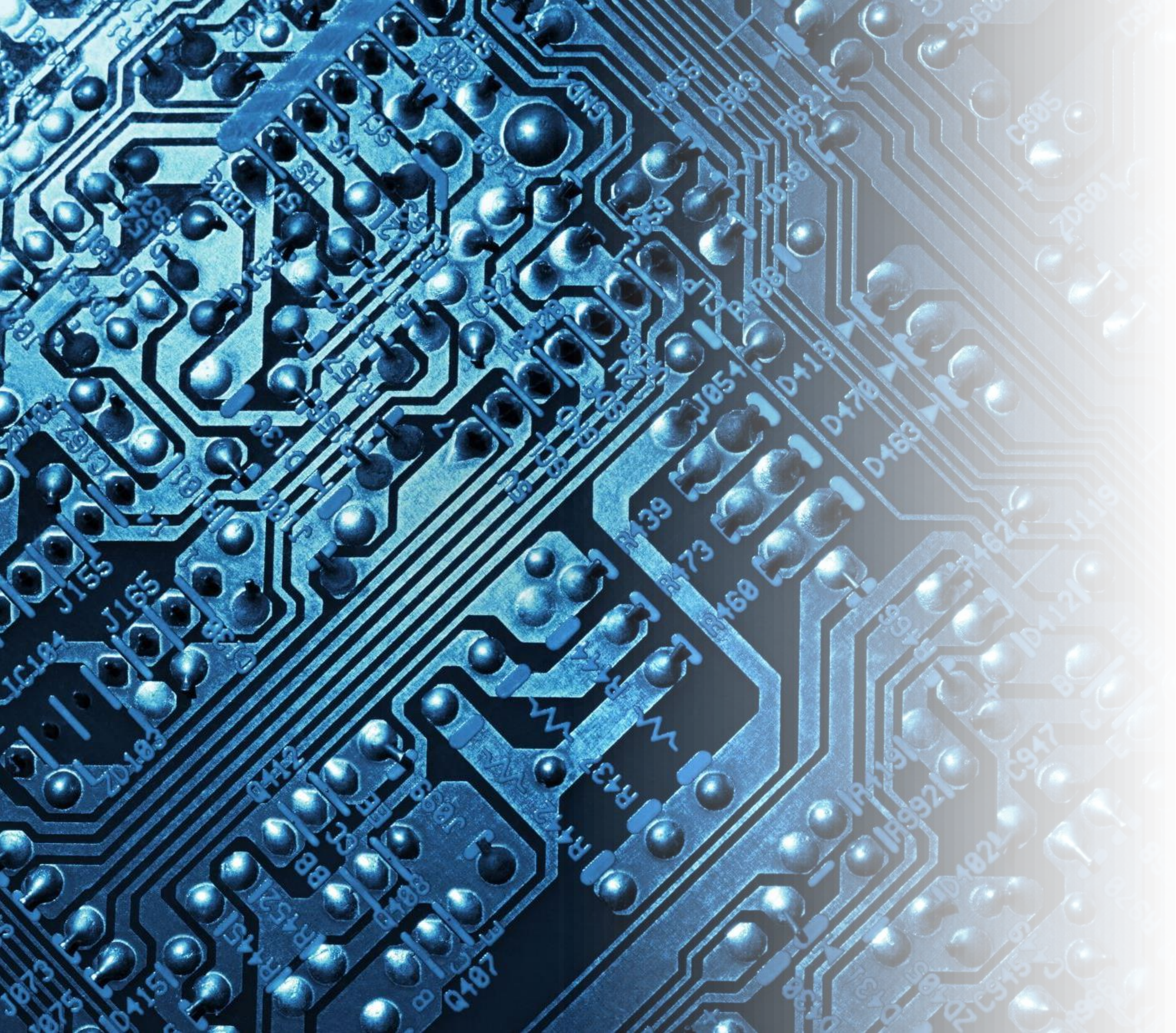


104010 : BASIC ELECTRONICS ENGINEERING

UNIT III
DIGITAL ELECTRONICS

Number Systems

Digital Devices

- A digital device contains an electrical circuit that uses discrete (exact) values in its design and function.
- These discrete values are usually zero's (0) and one's (1).
- You have probably heard of this system of notation based on only two possible values.
- What do we call it?

Binary…!!

Analog and Digital Signals

Analog Signals

- Continuous
- Infinite range of values
- More exact values, but more difficult to work with

Digital Signals

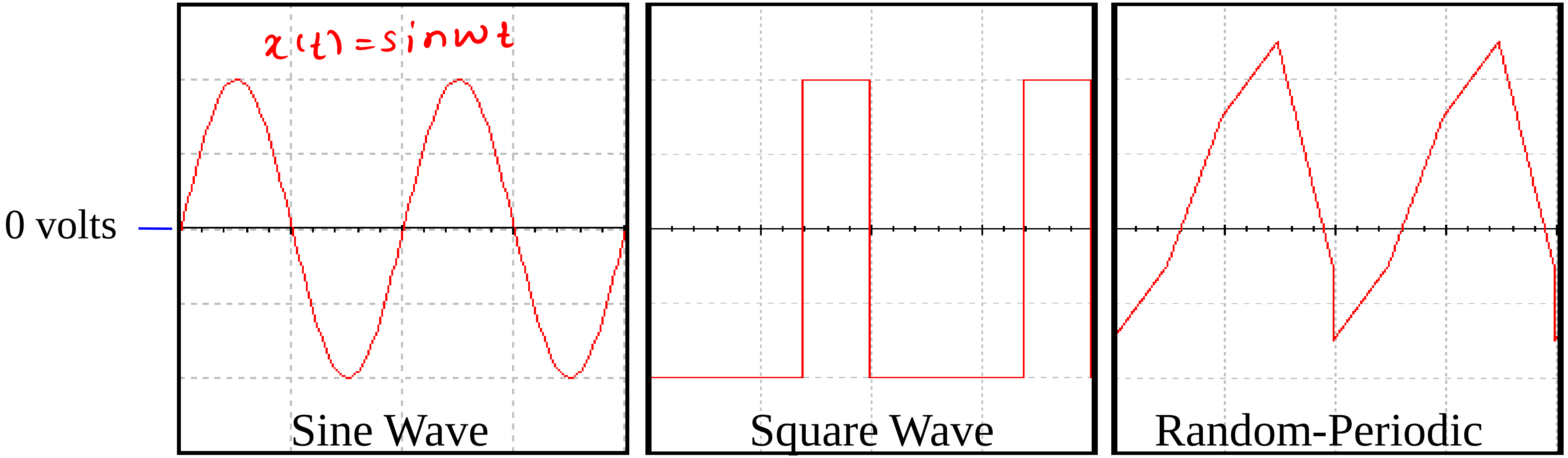
- Discrete
- Finite range of values
- Not as exact as analog, but easier to work with

Example:

A digital thermostat in a room displays a temperature of 72°. An analog thermometer measures the room temperature at 72.482°. The analog value is continuous and more accurate, but the digital value is more than adequate for the application and significantly easier to process electronically.

Example of Analog Signals

- An analog signal can be any time-varying signal.
- Minimum and maximum values can be either positive or negative.
- They can be periodic (repeating) or non-periodic. $x(t) = x(t+T)$
- Sine waves and square waves are two common analog signals. T = period of signal



Sine Wave

Square Wave
(not digital)

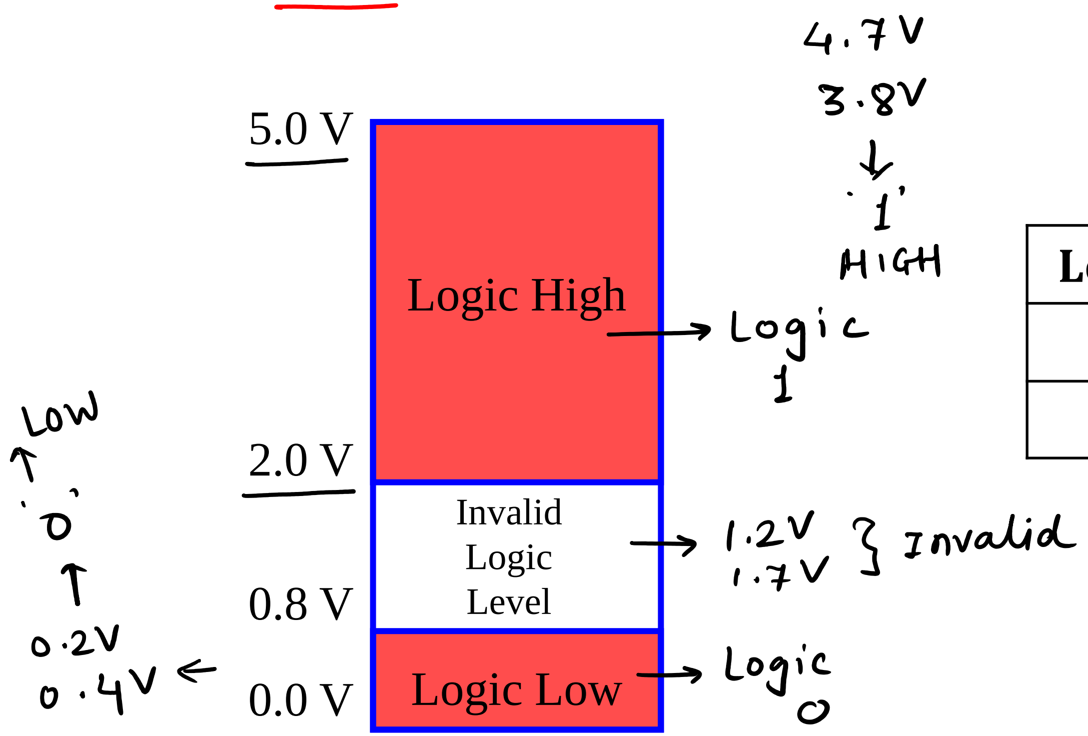
Random-Periodic

Logic Levels

Before examining digital signals, Lets define logic levels. A logic level is a voltage level that represents a defined digital state.

Logic HIGH: The higher of two voltages, typically 5 volts (5V)

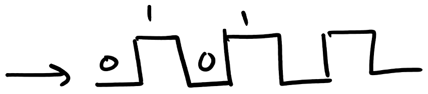
Logic LOW: The lower of two voltages, typically 0 volts (0V)



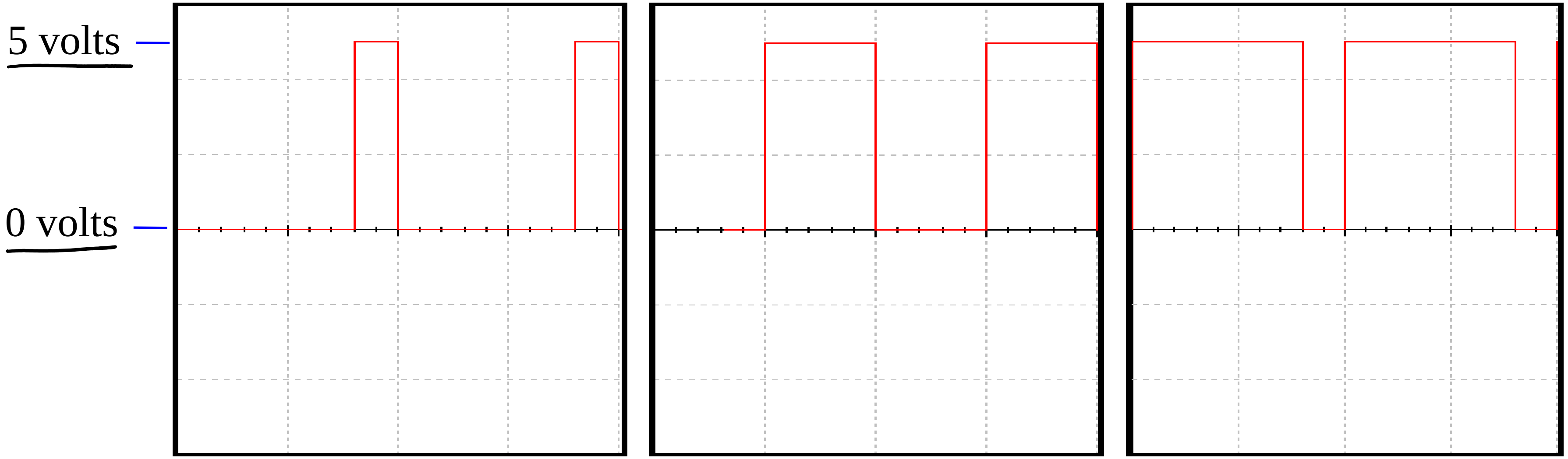
Logic Level	Voltage	True/False	On/Off	0/1
HIGH →	5 volts →	True →	On →	1
LOW →	0 volts →	False →	Off →	0

Example of Digital Signals

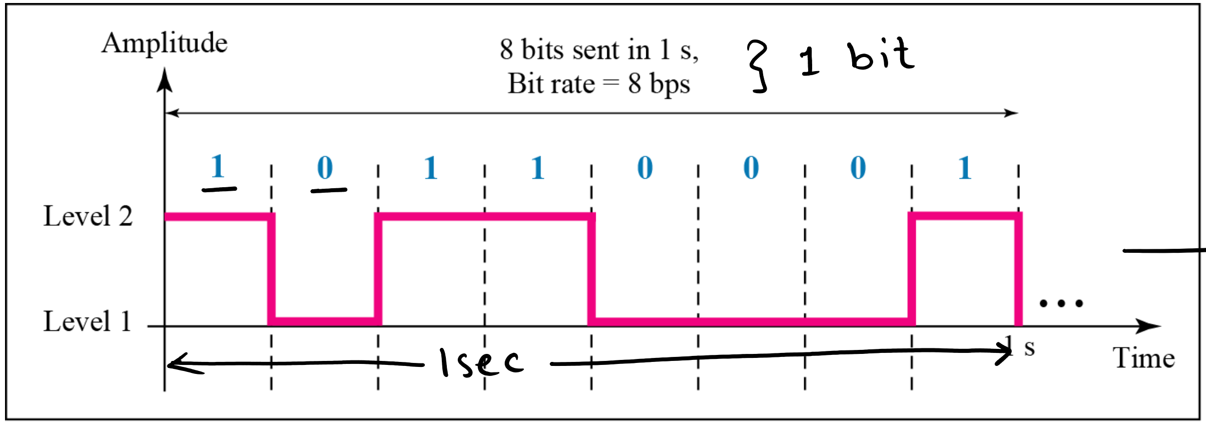
- Digital signal are commonly referred to as square waves or clock signals.



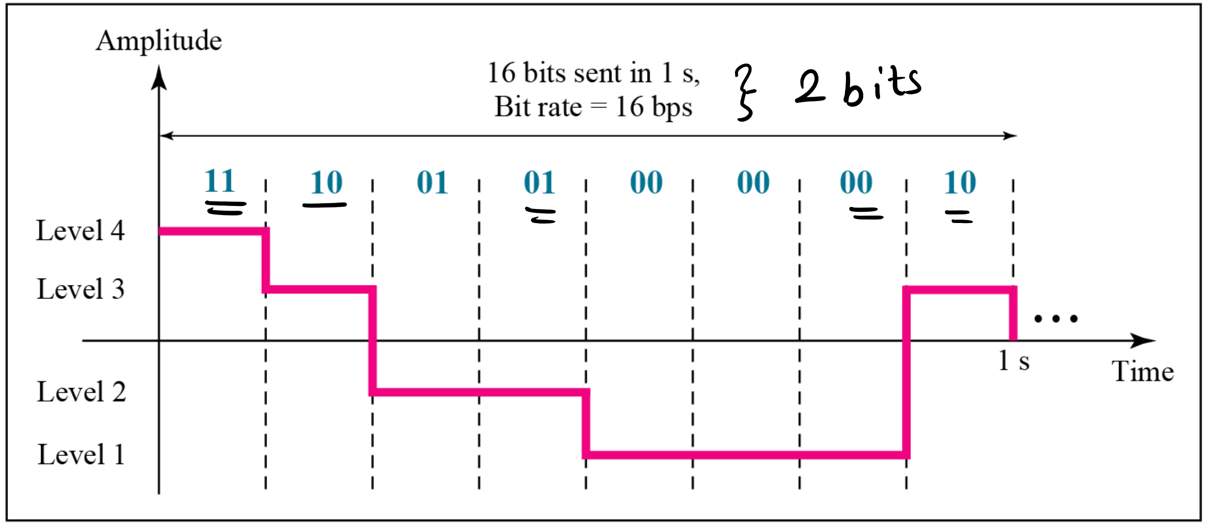
- Their minimum value must be 0 volts, and their maximum value must be 5 volts.
- They can be periodic (repeating) or non-periodic.



Examples of Digital Signals



a. A digital signal with two levels



b. A digital signal with four levels

$n = 16$

$2^n = 65536$

$n = 8$

$2^n = 256$

Two digital signals: one with two signal levels and the other with four signal levels

→ 8 Bits in 1 sec → 8 bps

n bits are used for representation.

$\therefore$ No. of levels = 2^n → 0 to $2^n - 1$

i) $n = 1$

$2^n = 2^1 = 2$ → 0, 1

ii) $n = 2$

$2^n = 2^2 = 4$

- 00 → 0
- 01 → 1
- 10 → 2
- 11 → 3

4 → (0 to 3)

iii) $n = 3$

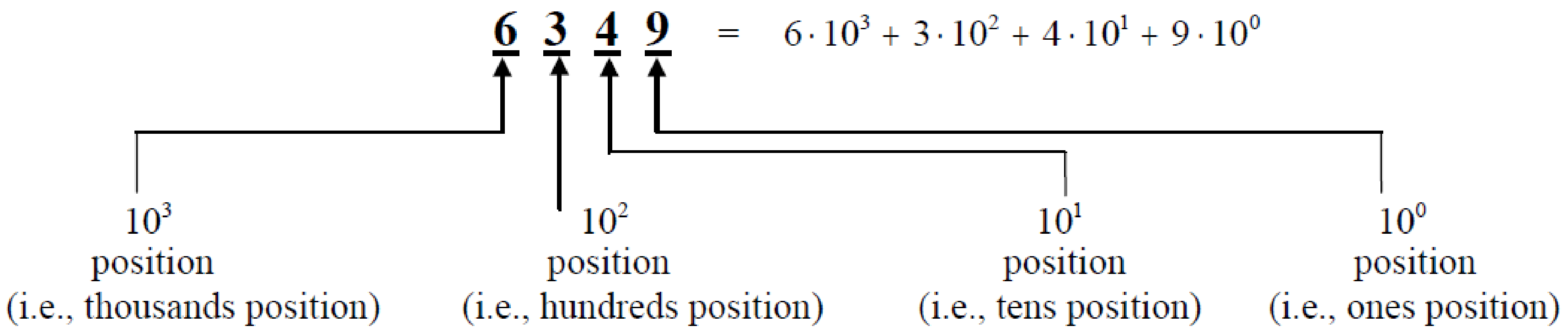
$2^n = 2^3 = 8$

- 000 → 0
- 001 → 1
- 010 → 2
- 011 → 3
- 100 → 4
- 101 → 5
- 110 → 6
- 111 → 7

Number System

Decimal

- The number system we are familiar with, used every day, is the *decimal number system.*
- Referred to as the **base-10** system.
- The base-10 number system has 10 distinct symbols, or digits (**0, 1, 2, 3,…8, 9)**.



Binary (Base - 2)

4 bits : Nibble
8 bits : Byte

- All data in a computer is represented in binary. A **base-2** system, or *binary number system.*
- The term 'bit' ($\rightarrow 0$, $\rightarrow 1$) is a contraction of the words 'binary' and 'digit'.
- The base-2 system has exactly two symbols**: 0 and 1.**
- 0 and 1 are *logical values, not the values of a physical quantity.*
- A string of eight bits (such as 11000110) is termed a *byte. A collection of four bits (such as 1011) is* smaller than a byte and is hence termed a *nibble.*

4 bit binary no.

$$\mathbf{1}\ \mathbf{1}\ \mathbf{0}\ \mathbf{1} = 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 13_{10}$$

$$= 8 + 4 + 0 + 1 = 13 \Rightarrow (13)_{10} = (1101)_2$$

2^3 2^2 2^1 2^0

2^3 position (i.e., eights position)

2^2 position (i.e., fours position)

2^1 position (i.e., twos position)

2^0 position (i.e., ones position)

Examples:

i) $(1110010) = 64 + 32 + 16 + 2 = (114)_{10}$

$2^6\,2^5\,2^4\,2^3\,2^2\,2^1\,2^0$

ii) $(011011) = 16 + 8 + 2 + 1 = (27)_{10}$

$2^4\,2^3\ \ 2^1\,2^0$

BCD Binary-Coded Decimal

- In some practical applications, a digital logic circuit is used to drive a numeric display, where each individual display unit displays a single digit.

100 111 010

- So, for example, we might have the number 472 in our logic circuit, and we would like to display this on three separate display units (one for the 4, one for the 7 and one for the 2).

1 9 2
0001 1001 0010

- Working with this sort of display hardware is facilitated using binary-coded decimal (BCD), where each individual digit is represented by a 4-bit number.

6 → 9
↳ 4 bits

- For example, to represent the decimal number 472 in binary-coded decimal (BCD), we convert each digit to a four bit binary number, independent of the other decimal digits. Thus, 472 equal 0100 0111 0010 in BCD.

$\underset{4}{0100}$ $\underset{7}{0111}$ $\underset{2}{0010}$

Example : $(13)_{10} \rightarrow (1101)_2$

BCD $\rightarrow (\underbrace{0001}_{1}\ \underbrace{0011}_{3})$

$(27)_{10} = (011011)_2$

$\rightarrow (\underbrace{0010}_{2}\ \underbrace{0111}_{7})$

Octal → Base-8

- Octal Numbering System Eight symbols:
  - 0, 1, 2, 3, 4, 5, 6, 7
- Notice that we no longer use 8 or 9
- Base Comparison: Base 10: $\underbrace{0, 1, 2, 3, 4, 5, 6, 7,}$ 8 , 9, 10, 11, 12…
- Base 8: 0, 1, 2, 3, 4, 5, 6, 7
- Example: What is $(15)_8$ in base 10?
  - 8^1 8^0
  - $(15)_8 = (1 \times 8^1) + (5 \times 8^0) = (13)_{10}$ ⇒ $(15)_8 = (13)_{10}$
- Example: What is $(7061)_8$ in base 10?
  - $(7061)_8 = (7 \times 8^3) + (0 \times 8^2) + (6 \times 8^1) + (1 \times 8^0)$ = 36331

Example:

i) $(123)_8 = (1 \times 8^2) + (2 \times 8^1) + (3 \times 8^0)$

$= 83$

$(123)_8 = (83)_{10}$

ii) $(17)_8 = (1 \times 8^1) + (7 \times 8^0)$

$= 8 + 7$

$= 15$

$\therefore (17)_8 = (15)_{10}$

Hexadecimal (Base-16)

- The base-16 hexadecimal number system has 16 digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E and F).

Valid symbols -

0
1
2
3
4
5
6
7
8
9

$10 \rightarrow A$
$11 \rightarrow B$
$12 \rightarrow C$
$13 \rightarrow D$
$14 \rightarrow E$
$15 \rightarrow F$

$(0 \rightarrow F)$

$0 \rightarrow F$
$10 \rightarrow 1F$
$20 \rightarrow 2F$

Example:

1) $(1A)_{16} = (1 \times 16^1) + (10 \times 16^0)$

$16^1 \quad 16^0$

$= 16 + 10$

$= 26.$

$(1A)_{16} = (26)_{10}$

1> Decimal → Binary

Convert $(26)_{10}$ into a Binary number.

↓ Base - 2

Solution:

Division	Quotient	Generated remainder
$\frac{26}{2}$	13	0
$\frac{13}{2}$	6	1
$\frac{6}{2}$	3	0
$\frac{3}{2}$	1	1
$\frac{1}{2}$	0	1

2	26	
2	13	0 → LSB
2	6	1
2	3	0
2	1	1
	0	1 → MSB

$(1\,0\,0\,1)_2$

MSB Most Significant Bit

LSB Least Significant Bit

$(26)_{10} = (11010)_2$

Hence the converted binary number is $(11010)_2$.

Decimal → Binary

Example:

$(47)_{10} = (101111)_2$

2	47	
2	23	1
2	11	1
2	5	1
2	2	1
2	1	0
	0	1

2	13	1	LSB
2	6	0	
2	3	1	
2	1	1	MSB
	0		

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

Binary → Decimal

Example :

1) $(1110111)_2$

$= (119)_{10}$

$2^3\, 2^2\, 2^1\, 2^0$

$$1101 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$= 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$$

$$= 8 + 4 + 0 + 1$$

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

1, 2, 4, 8, 16, 32, 64, 128, 256, 512, ….

2^0 2^1 2^2 2^3 2^4 2^5 2^6 2^7 2^8 2^9

Octal → Decimal

Example :

1) $(125)_8$

$= 1 \times 8^2 + 2 \times 8^1 + 5 \times 8^0$

$= (85)_{10}$

ii) $(28)_8$

$\downarrow$ (0 – 7)

Invalid octal number

$$137 = 1 \times 8^2 + 3 \times 8^1 + 7 \times 8^0$$
$$= 1 \times 64 + 3 \times 8 + 7 \times 1$$
$$= 64 + 24 + 7$$

$$\boxed{(137)_8 = (95)_{10}}$$

- Digits used in Octal number system – 0 to 7

Decimal $\rightarrow$ Octal

↳ Base - 8

i) $(85)_{10}$

8	85	
8	10	5
8	1	2
	0	1

$(85)_{10} = (125)_8$

8	95	7
8	11	3
8	1	1
	0	

LSP

MSP

$$\boxed{(95)_{10} = (137)_8}$$

ii) $(888)_{10}$

8	888	
8	111	0
8	13	7
8	1	5
	0	1

$(888)_{10} = (1570)_8$

Converting a Hexadecimal Number to a Decimal Number

- Write the hexadecimal number as a sum of powers of 16.

- For example, considering the hexadecimal number 1A9B above, we convert this to decimal as:

$16^3 \quad 16^2 \quad 16^1 \quad 16^0$

$$\underline{1A9B} = 1 \text{ x}16^3 + \underline{A}x16^2 + 9\text{x}16^1 \; + \underline{B}x16^0$$

$$= 4096 + 10(256) + 9(16) + 11(1) = 6811$$

1) $(2C3D)_{16} = (2\times16^3) + (12\times16^2) + (3\times16^1) + (13\times16^0)$

$= (11325)_{10}$

$\therefore (2C3D)_{16} = (11325)_{10}$

Hex → Decimal

$$\text{BAD} = 11 \times 16^2 + 10 \times 16^1 + 13 \times 16^0$$
$$= 11 \times 256 + 10 \times 16 + 13 \times 1$$
$$= 2816 + 160 + 13$$

$$\boxed{(\text{BAD})_{16} = (2989)_{10}}$$

A = 10, B = 11, C = 12, D = 13, E = 14, F = 15

Converting a Decimal Number to a Hexadecimal Number

↳ Base - 16

- To convert from decimal to hexadecimal is to use the same division algorithm that you used to convert from decimal to binary, but repeatedly dividing by 16 instead of by 2.
- As before, we keep track of the remainders, and the sequence of remainders forms the hexadecimal representation.

- Convert (348)10 into a hexadecimal number.

Example:

1> $(2989)_{10} = (BAD)_{16}$

16	2989		
16	186	13	→ D
16	11	10	→ A
	0	11	→ B

Division	Quotient	Generated remainder
$\frac{348}{16}$	21	12
$\frac{21}{16}$	1	5
$\frac{1}{16}$	0	1

Hence the converted hexadecimal number is $(15C)_{16}$.

Decimal → Hex

16	2989	13	LSP
16	186	10	
16	11	11	
	0		MSP

$$\boxed{(2989)_{10} = (BAD)_{16}}$$

Hexadecimal → Binary

i) $(15C)_{16}$

$\downarrow$

$(348)_{10}$ ×

$\downarrow$

$(101011100)_2$

$\downarrow$

$(\underbrace{0001}_{1}\ \underbrace{0101}_{5}\ \underbrace{1100}_{C})$

ii) $(27A)_{16}$

$= (\underbrace{0010}_{2}\ \underbrace{0111}_{7}\ \underbrace{1010}_{A})_2$

Converting a Hexadecimal Number to a Binary Number

- We can convert directly from hexadecimal notation to the equivalent binary representation by using the following procedure:
  - Convert each hexadecimal digit to a four digit binary number, independent of the other hexadecimal digits.
  - Concatenate the resulting four-bit binary numbers together.
- For example, to convert the hexadecimal number 4DA9 to binary, we first convert each hexadecimal digit to a four-bit string:

  4 = 0100 D = 1101 A = 1010 9 = 1001

  and then concatenate the results:

  The resulting binary number is: $\underbrace{0100}_{4}\ \underbrace{1101}_{D}\ \underbrace{1010}_{A}\ \underbrace{1001}_{9}$.
- We can drop leading zeros (from the leftmost quartet only!), giving us:

  4DA9 = 100110110101001

Converting a Binary Number to a Hexadecimal Number

- Convert directly from binary notation to the equivalent hexadecimal representation by using the following procedure:
  - Starting at the right, collect the bits in groups of 4
  - Convert each group of 4 bits into the equivalent hexadecimal digit
  - Concatenate the resulting hexadecimal digits
- For example, to convert $\underbrace{1101}_{D}\underbrace{1010}_{A}\underbrace{1001}_{9}$ to hexadecimal, we collect the bits into groups of 4 starting at the right: 1101 1010 1001, and then we convert each collection of bits into a hexadecimal digit:

  1101 1010 1001

  D A 9

  Thus, 110110101001 = DA9

Example →

i) $(\underbrace{0111}_{7}\,\underbrace{0111}_{7})_2 = (77)_{16}$

ii) $(\underbrace{11}_{\underbrace{0011}_{3}}\,\underbrace{1011}_{B})_2 = (3B)_{16}$

Conversion of Floating Numbers

Binary to Decimal

(0.5) (0.25)

$\frac{1}{2}$ $1/4$ $1/8$ $1/16$

2^{-1} 2^{-2} 2^{-3} 2^{-4}

$\underbrace{10111}.\underbrace{0111}$

$$(1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 1\times 2^1 + 1\times 2^0).(0\times 2^{-1} + 1\times 2^{-2} + 1\times 2^{-3} + 1\times 2^{-4})$$

$$(16+0+4+2+1).(0+0.25+0.125+0.0625)$$

23.4375

$2^{-1} = \frac{1}{2} = 0.5$

$2^{-2} = \frac{1}{4} = 0.25$

$2^{-3} = \frac{1}{8} = 0.125$

$2^{-4} = \frac{1}{16} = 0.0625$

1) 1011.0110

$= (1\times 2^3 + 1\times 2^1 + 1\times 2^0).(0\times 2^{-1} + 1\times 2^{-2} + 1\times 2^{-3} + 0\times 2^{-4})$

$= 11.375$

Decimal to Binary

23.4375

$23 \Rightarrow 2\overline{)23}$ Remainder 1 LSB

$2\overline{)11}$ 1

$2\overline{)5}$ 1

$2\overline{)2}$ 0

1 1 MSB $(23)_{10} = (10111)_2$

$0.4375 \Rightarrow 0.4375 \times 2 = 0.875 \times 2 = 1.75 \Rightarrow 0.75 \times 2 = 1.5 \Rightarrow 0.5 \times 2 = 1$

0 1 1 1

$(23.4375)_{10} = (10111.0111)_2$

$(11.375) = (1011.011)_2$

i) $0.4375 \times 2 = 0.875 \rightarrow 0$

ii) $0.875 \times 2 = 1.75 \rightarrow 1$

iii) $0.75 \times 2 = 1.5 \rightarrow 1$

iv) $0.5 \times 2 = 1 \rightarrow 1$

i) $0.375 \times 2 = 0.75 \rightarrow 0$

ii) $0.75 \times 2 = 1.5 \rightarrow 1$

iii) $0.5 \times 2 = 1 \rightarrow 1$

Number Representation

Decimal	Octal	Binary	Hex
0	0	0000	0
1	1	0001	1
2	2	0010	2
3	3	0011	3
4	4	0100	4
5	5	0101	5
6	6	0110	6
7	7	0111	7
8	10	1000	8
9	11	1001	9
10	12	1010	A
11	13	1011	B
12	14	1100	C
13	15	1101	D
14	16	1110	E
15	17	1111	F
16	20	0001 0000	10
17	21	0001 0001	11
18	22	0001 0010	12
19	23	0001 0011	13

Example:

$(777)_{10}$

i) Binary

$= (1100001001)_2$

ii) Octal

$= (1411)_8$

iii) Hexadecimal

$= (309)_{16}$

iv) BCD

$= (0111\,0111\,0111)$

Unsigned Binary Numbers

For n bit number,
2^n combinations
↳ 0 to $2^n - 1$

- An n-bit binary number

  B = $b_{n-1}b_{n-2}.... b_2b_1b_0$

  2^n distinct combinations are possible, 0 to 2^n-1.

$2^3 = 8$

- For example, for n = 3, there are 8 distinct combinations.
  - 000, 001, 010, 011, 100, 101, 110, 111

    000 → 0 ↳ 0

    111 → 7 ↳ $2^3 - 1$

- Range of numbers that can be represented

  n=8 → 2^8 → 0 to 2^8-1 (255)

  n=16 → 2^{16} → 0 to 2^{16}-1 (65535)

  n=32 → 2^{32} → 0 to 2^{32}-1 (4294967295)

Signed Integer Representation

- Many of the numerical data items that are used in a program are signed (positive or negative).
  - Question:: How to represent sign?

- Three possible approaches:
  - Sign-magnitude representation
  - One’s complement representation
  - Two’s complement representation

Sign-magnitude Representation

Example:

1) 1101

Unsigned

13

Signed

1 101

sign magnitude

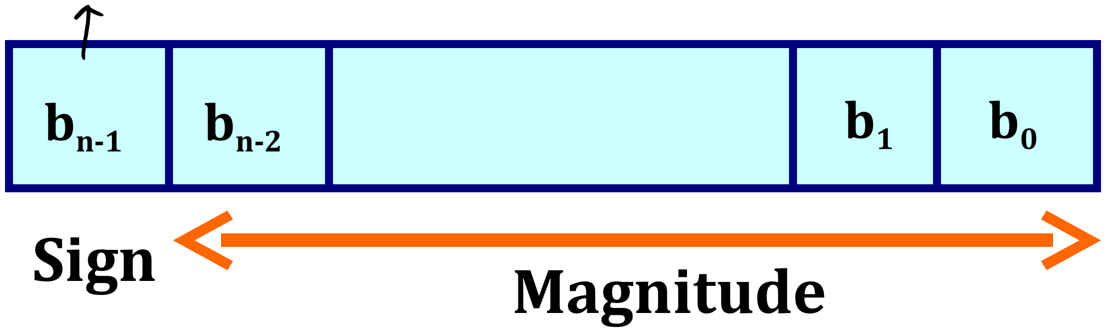
-5

Signed

0101

= (+5)

- **For an n-bit number representation**
  - The most significant bit (MSB) indicates sign
    - 0 positive
    - 1 negative
  - The remaining n-1 bits represent magnitude.



Contd.

- **Range of numbers that can be represented:**

  Maximum :: + $(2^{n-1} - 1)$

  Minimum :: - $(2^{n-1} - 1)$

Unsigned Numbers (n bits) → 0 to $2^n - 1$

Example → $n = 4$ ⇒ -7 to $+7$

1111 → -7

0111 → $+7$

$-7 \rightarrow -(2^{n-1} - 1)$

$(2^{4-1} - 1)$

$(2^3 - 1)$

(7)

$+7 \rightarrow +(2^{n-1} - 1)$

- **A problem:**

  Two different representations of zero.

+0	0 000....0
-0	1 000....0

4 bits → $\left.\begin{array}{l} +0 : 0000 \\ -0 : 1000 \end{array}\right\}$ Not valid

One's Complement Representation

- **Basic idea:**
  - Positive numbers are represented exactly as in sign-magnitude form.
  - Negative numbers are represented in 1's complement form.

- **How to compute the 1's complement of a number?**
  - Complement every bit of the number ($1 \longrightarrow 0$ and $0 \longrightarrow 1$) .
  - MSB will indicate the sign of the number.

    $0 \longrightarrow$ positive

    $1 \longrightarrow$ negative

Example :: n=4 → −7 to +7 → $-(2^{n-1}-1)$ to $+(2^{n-1}-1)$

signed mag.

$-7 \Rightarrow 1111$

1000

sign

16 →

0000	→	+0	1000	→	-7
0001	→	+1	1001	→	-6
0010	→	+2	1010	→	-5
0011	→	+3	1011	→	-4
0100	→	+4	1100	→	-3
0101	→	+5	1101	→	-2
0110	→	+6	1110	→	-1
0111	→	+7	1111	→	-0

MSB ↑ ↳mag

Example:

i) −16 (6 bits)

010000 ←

101111 ← 1's.

ii) −13 (5 bits)

01101

→ 10010

iii) +9 (5 bits)

01001

To find the representation of, say, -4, first note that

+4 = 0100 +16 = 010000

−16 = 101111

-4 = 1's complement of 0100 = 1011

Contd.

- **Range of numbers that can be represented:**

  Maximum :: $+ (2^{n-1} - 1)$

  Minimum :: $-(2^{n-1} - 1)$

- **A problem:**

  Two different representations of zero.

+0	$\underline{0}\ \underbrace{000....0}$
-0	$\underline{1}\ \underbrace{111....1}$

- **Advantage of 1's complement representation**
  - * **Subtraction can be done using addition.**
  - **Leads to substantial saving in circuitry.**

Two's Complement Representation

- **Basic idea:**
  - **Positive numbers are represented exactly as in sign-magnitude form.**
  - **Negative numbers are represented in 2's complement form.**

- **How to compute the 2's complement of a number?**
  - **Complement every bit of the number (1 $\rightarrow$ 0 and 0 $\rightarrow$ 1), and then *add one* to the resulting number.**
  - **MSB will indicate the sign of the number.**

    0 $\rightarrow$ positive

    1 $\rightarrow$ negative

Example :: n=4

$\hookrightarrow$ (-8 to +7)

$(-2^{n-1}$ to $(2^{n-1}-1))$

$\boxed{0000 \rightarrow +0}$	$1000 \rightarrow -8$
$0001 \rightarrow +1$	$1001 \rightarrow -7$
$0010 \rightarrow +2$	$1010 \rightarrow -6$
$0011 \rightarrow +3$	$1011 \rightarrow -5$
$0100 \rightarrow +4$	$1100 \rightarrow -4$
$0101 \rightarrow +5$	$1101 \rightarrow -3$
$0110 \rightarrow +6$	$1110 \rightarrow -2$
$0111 \rightarrow +7$	$1111 \rightarrow -1$

To find the representation of, say, -4, first note that

+4 = 0100

-4 = 2's complement of 0100 = 1011+1 = 1100

Addition:

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1 *

e.g. →

1 + 1 = 2

```
  1
  0 1
+ 0 1
------
 1 0 → 2
```

Example : i) +15

01111

ii) -17

```
+17 : 010001
1's : 101110
     +     1
    -------
      101111
```

iii) -9

```
+9 : 01001
1's : 10110
     +    1
    -------
      10111
```

e.g.

```
+4  →  0 1 0 0
1's →  1 0 1 1
     +       1
     ---------
2's →  1 1 0 0
```

```
+0  →  0 0 0 0
1's →  1 1 1 1
     +       1
     ---------
     1 0 0 0 0
```

Neglect

Example : -16

```
+16 : 010000
1's : 101111
     +     1
    --------
-16 ← 110000
```

Contd.

- **Range of numbers that can be represented:**

  Maximum :: $+ (2^{n-1} – 1)$

  Minimum :: $– 2^{n-1}$

- **Advantage:**
  - ***Unique representation of zero*.**
  - **Subtraction can be done using addition.**
  - **Leads to substantial saving in circuitry.**
- **Almost all computers today use the 2’s complement representation for storing negative numbers.**

Decimal	2's Complement	1's complement	Signed Magnitude
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	-----	1111	1000
-1	1111	1110	1001
-2	1110	1101	1010
-3	1101	1100	1011
-4	1100	1011	1100
-5	1011	1010	1101
-6	1010	1001	1110
-7	1001	1000	1111
-8	1000	------	-------

Binary Arithmetic

- Addition / subtraction
- Unsigned
- Signed
  - Using negative numbers

Unsigned: Addition

Bit - A	Bit - B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Like normal decimal addition

$$\begin{array}{r} 0101\ (5) \\ +\ 1001\ (9) \\ \hline 1110\ (14) \end{array}$$

The carry out of the MSB is neglected

Unsigned: Subtraction

Like normal decimal subtraction

Bit - A	Bit - B	Diff	Barrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$\begin{array}{r} 1001\ (9) \\ -\ 0101\ (5) \\ \hline 0100\ (4) \end{array}$$

Unsigned

Binary Addition

	Decimal		Binary						
Carry	1		1	1	1	1	1	1	
Num - A	2	9 (5)		0	1	1	1 o	0	1
Num - B	3	9 (6)		1	0	0	1	1	1
Sum	6	8	1	0	0	0	1	0	0

carry out of MSB

Signed Arithmetic

- Use a negative number representation scheme

- Reduces subtraction to addition

Overflow / Underflow

- Maximum value N bits can hold : $2^n - 1$
- When addition result is bigger than the biggest number of bits can hold.
  - Overflow
- When addition result is smaller than the smallest number the bits can hold.
  - Underflow

Overflow Example

$$\begin{array}{l} 011\ (+3)_{10} \\ 011\ (+3)_{10} \\ \hline 110\ \ (+6)_{10} \quad ???? \end{array}$$

1's complement computer interprets it as – 1 !!

$(+6)_{10} = (0110)_2$ requires four bits !

Underflow Examples

Two's complement addition

$$
\begin{array}{lrrl}
 & & 101 & (-3)_{10} \\
 & & 101 & (-3)_{10} \\
\hline
\text{Carry} & 1 & 010 & (-6)_{10} \quad ????
\end{array}
$$

The computer sees it as +2.

$(-6)_{10} = (1010)_2$ again requires four bits !

One's(1's) Complement Arithmetic

A = 8
B = 9

A + B = 17
signed 17
6.

- Given two numbers in Binary : A and B → Max (A, B, A + B)
- Find the number of bits that are required to represent the signed number.
- The number of bits required to represent the sign number is n, such that $2^{n-1}-1$ is greater than or equal to maximum of A or B or A+B.
- Find 1's complement of a number by replacing '0' by '1' and '1' by '0'.
- Find the binary addition.
- Check the carry, If the carry is generated, add carry to LSB and take 1's Complement.
- If MSB is '1' then the answer is negative, and it is in 1's complement form.
- If MSB is '0' then the answer is positive and in true form.

Addition of Two Positive Numbers (8+9)

$A = 8$; $B = 9$

$A + B = 17$

1's comp. repre.

+8 : 001000

+9 : 001001

$$\begin{array}{r} 001000 \\ +\ 001001 \\ \hline 010001 \end{array}$$

$2^4 = 16$

MSB = 0

↳ Result is +ve

↳ True form

- Step 1 : 8+9=17, Max (8,9,17)=17, number of binary digits to represent 17 is 6 *

Step 2 : c → 1

- 1's Complement Rep'n of (A=+8) = 001000
- 1's Complement Rep'n of (B=+9) = 001001

□ Sum 010001

□ MSB is '0' ⇒ Sign of the sum is positive

Sum is + 10001 = 17

+1 : 00001

−1 : 11110

+9 : 01001

−9 : 10110

Example;

$8 + (-9) = (-1)$

i) 5 bits

ii) 1's comp.

+8 : 01000

−9 : 10110

$$\begin{array}{r} 01000 \\ +\ 10110 \\ \hline 11110 \end{array}$$

MSB = 1

↳ Result is −ve

↳ 1's comp form

↳ (−1)

Addition of Positive and Negative Numbers: 8+(-9)

- Step 1 : 8-9=-1, Max(8,9,1)=9, number of binary digits to represent 9 is 5
- Step 2 :
  - 1's Complement Rep'n of (A=+8) = 01000
  - 1's Complement Rep'n of (B= -9) = 10110
  - 01001(+9) $\Rightarrow$ 10110(-9)
  - Sum 11110
  - No carry, but MSB is '1' $\Rightarrow$ Sign of the sum is Negative, result is in 1's complement form
  - Sum is = 11110 $\Rightarrow$ 10001=(-1)

$A = -8$

$B = -9$

$A + B = -17$

1) No. of bits = 6.

ii) 1's comp. repre.

		1	1	0	1	1	1
−8		1	1	0	1	1	1
−9	+	1	1	0	1	1	0
	1	1	0	1	1	0	1
		+					1
		1	0	1	1	1	0

carry out of MSB

Add to LSB

+8 : 001000

−8 : 110111

+9 : 001001

−9 : 110110

HW = −8+9

8+9

8−9

−8−9

+17 : 010001

−17 : 101110.

Result : 101110

MSB = 1

Result is −ve → −17

Addition of Two and Negative Numbers: -8+(-9)

- Step 1 :-8-9=-17, Max(8,9,17)=17, number of binary digits to represent 17 is 5
- Step 2 : carry → 110110
  - 1's Complement Rep'n of (A=-8) = 110111
    - 001000(+8) → 110111(-8)
  - 1's Complement Rep'n of (B= -9) = 110110
    - 001001(+9) → 110110(-9)
  - Sum 1101101
  - MSB has carry, add this carry to LSB leading to =101110
  - MSB is '1, Sign of the sum is Negative, result is in 1's complement form → 101110
  - Result is = 110001 110001=(-17)

Two’s(2’s) Complement Arithmetic

- Given two numbers in Binary : A and B.
- Find the number of bits that are required to represent the sign number.
- Find 1’s complement of a number by replacing ‘0’ by ‘1’ and ‘1’ by ‘0’.
- Find two’s complement by adding ‘1’ to the 1’s complement.
- Carry out the binary addition.
- Check the carry, If the carry is generated, discard the carry Complement.
- If No carry ----If MSB is ‘1’ then the answer is negative, and it is in 2’s complement form.
- If MSB is ‘0’ then the answer is positive and in true form

Addition of Two Positive Numbers (8+9)

- Step 1 : 8+9=17, Max(8,9,17)=17, number of binary digits to represent 17 is 6
- Step 2 :
  - 1's Complement Rep'n of (A=+8) = 001000
  - 1's Complement Rep'n of (B=+9) = 001001
- Step 3:
  - 2's Complement Rep'n of (A=+8)=001000
  - 2's Complement Rep'n of (B=+9)=001001

Carry		1				
A(8)	0	0	1	0	0	0
B(9)	0	0	1	0	0	1
Result	0	1	0	0	0	1

  - Result = 010001
  - MSB is '0' Sign of the sum is positive
  - Sum is + 10001 = 17

Addition of Positive and Negative Numbers: 8+(-9)

- Step 1 : 8+-9=-1, Max(8,9,1)=9, number of binary digits to represent 9 is 5
- Step 2 :
  - 1's Complement Rep'n of (A=+8) = 01000
  - 1's Complement Rep'n of (B=-9) = 10110
  - +9(01001) ⟹ -9(10110)
- Step 3:
  - 2's Complement Rep'n of (A=+8)=001000
  - 2's Complement Rep'n of (B=+9)=001001
  - +9(001001) ⟹ -9(110110+01)
  - ⟹ 110111
  - No carry, but MSB is '1', result is negative and the result is in 2's complement
  - Result = 11111(2's)-01 = 1 1 1 1 0 (1' s) = 1 0 0 0 1 = -1

Carry					
A(+8)(2's)	0	1	0	0	0
B(-9)(2's)	1	0	1	1	1
Result(2's)	1	1	1	1	1
-				0	1
Result(1's)	1	1	1	1	0

Addition of Two Negative Numbers: -8+(-9)

- Step 1 : -8+(-9)=-17, Max(8,9,17)=17, number of binary digits to represent 17 is 6
- Step 2 :
  - 1's Complement Rep'n of (A=-8) = 001000
  - +8(001000) ⟹ -8(110111)
  - 1's Complement Rep'n of (B=-9) = 110110
  - +9(001001) ⟹ -9(110110)
- Step 3:
  - 2's Complement Rep'n of (A=-8)=111000
  - +8(001000) ⟹ -8(110111)+(01)
  - 2's Complement Rep'n of (B=-9)=110111
  - +9(001001) ⟹ -9(110110+01)
  - 110111
  - Carry is generated, carry discarded
  - MSB is '1', result is negative and the result is in 2's complement
  - Result = 101111(2's)+01 = 1 01110 (1' s) = = 1 10001 = -17

Carry 1	1					
A(+8)(2's)	1	1	1	0	0	0
B(-9)(2's)	1	1	0	1	1	1
Result(2's)	1	0	1	1	1	1
+					0	1
Result(1's)	1	0	1	1	1	0

Additional Examples(42-22)

42-22=20, Max(42,22,20)=42,   n=7

1's Complement Method								
	1	1		1				
+42(1's)=0101010		0	1	0	1	0	1	0
+22(1's)=0010110 -22(1's) =1101001		1	1	0	1	0	0	1
Carry Generated, add it to LSB	1	0	0	1	0	0	1	1
								1
MSB is zero, therefore, the result is in positive and in true form		0	0	1	0	1	0	0
Result(20)		0	0	1	0	1	0	0

Additional Examples(42-22)

42-22=20, Max(42,22,20)=42,   n=7

2’s Complement Method								
	1	1		1		1		
+42(1’s)=0101010		0	1	0	1	0	1	0
+22(1’s)=0010110 -22(1’s) =1101001 -22(2’s)=1101010		1	1	0	1	0	1	0
Carry Generated, ignore it	1	0	0	1	0	1	0	0
								1
MSB is zero, therefore, the result is in positive and in true form		0	0	1	0	1	0	0
Result(20)		0	0	1	0	1	0	0

Additional Examples(-42+22)

42-22=20, Max(42,22,20)=42, n=7

1's Complement Method								
			1		1			
+42(1's)=0101010 -42(1's)= 1010101		1	0	1	0	1	0	1
+22(1's)=0010110		0	0	1	0	1	1	0
No Carry		1	1	0	1	0	1	1
MSB is '1', therefore, the result is negative and in 1's complement form								
		1	0	1	0	1	0	0
Result(-20)		1	0	1	0	1	0	0

Additional Examples(-42+22)

42-22=20, Max(42,22,20)=42,   n=7

2’s Complement Method								
			1		1			
+42(1’s)=0101010 -42(1’s)= 1010101 -42(2’s)=1010110		1	0	1	0	1	1	0
+22(1’s)=0010110		0	0	1	0	1	0	0
No Carry		1	1	0	1	0	1	0
MSB is ‘1’, therefore, the result is negative and in 2’s complement form		1	1	0	1	0	1	0
								1
Result(1’s)		1	1	0	1	0	1	1
Result(-20)		1	0	1	0	1	0	0

Text and Reference Books

- “Modern Digital Electronics” by R.P. Jain, 4th Edition, Tata McGraw Hill
- “Digital Fundamentals” by Thomas. L. Floyd, 11th Edition, Pearson

Acknowledgements

Modern Digital Electronics: R P Jain

Web sources

Thank you!