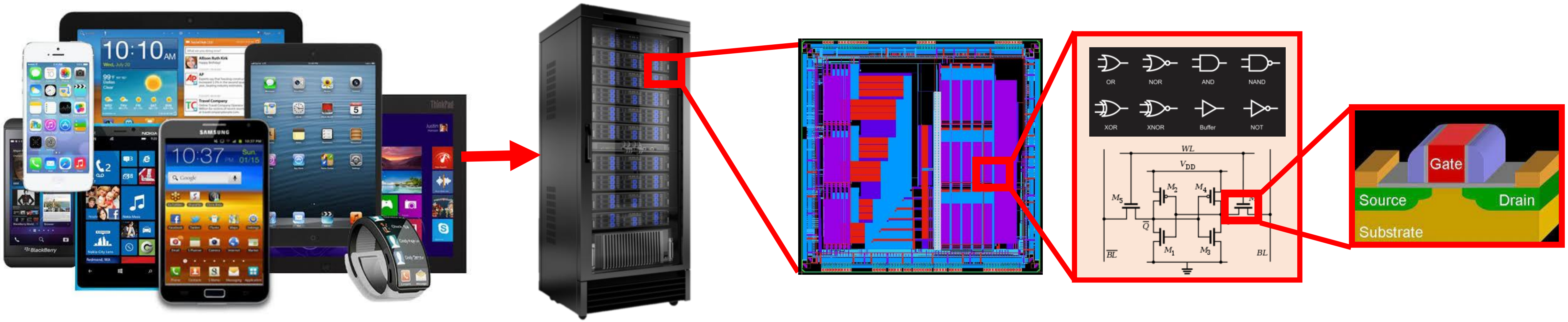


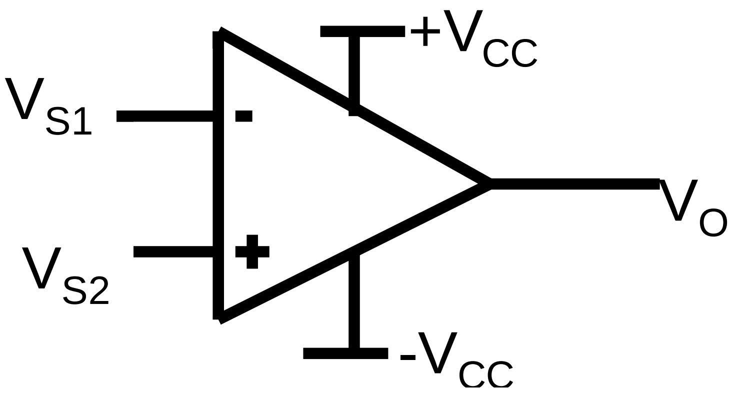
ESC201: INTRODUCTION TO ELECTRONICS

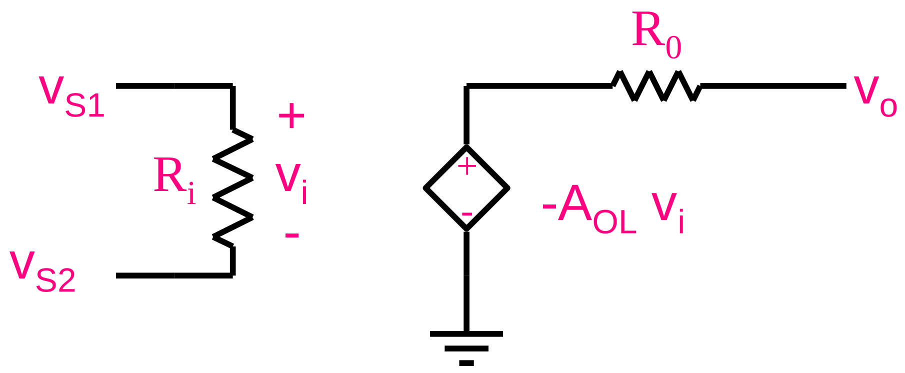
MODULE 6: DIGITAL CIRCUITS

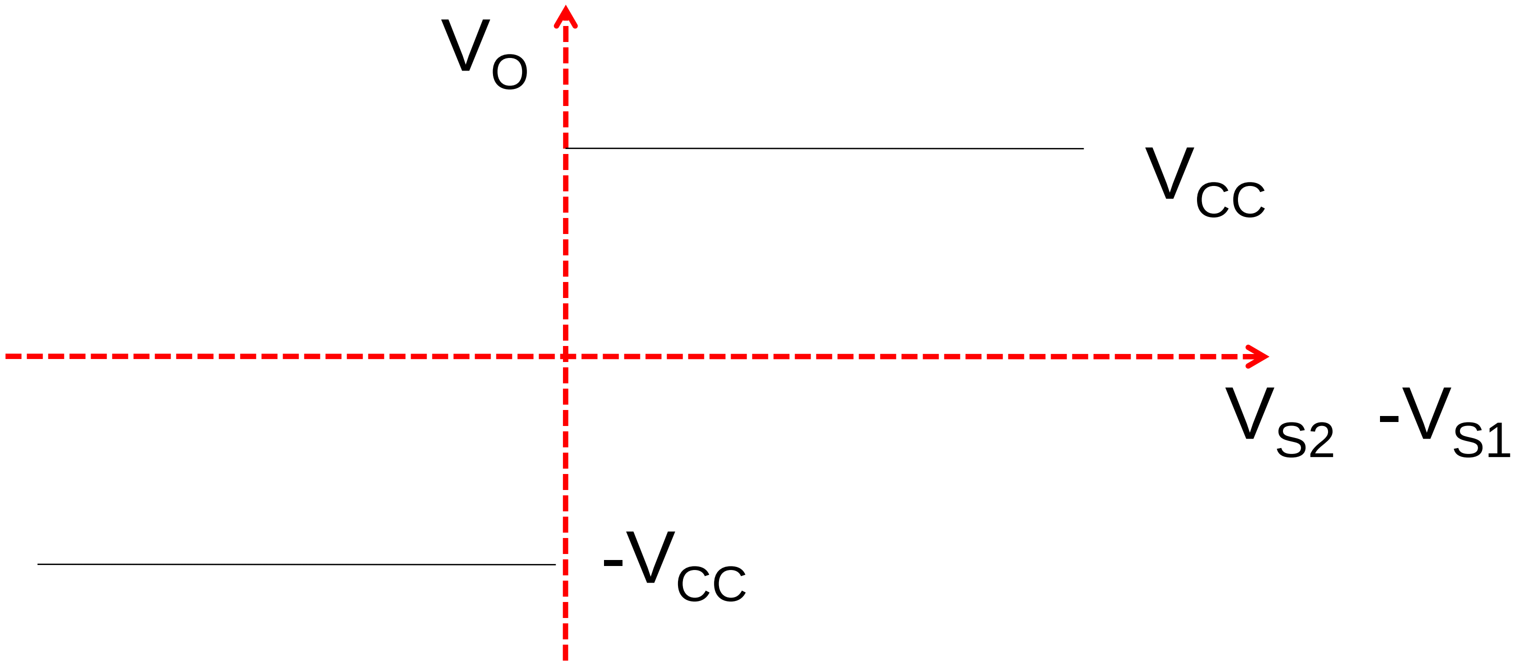
Dr. Shubham Sahay,
Associate Professor,
Department of Electrical Engineering,
IIT Kanpur

Comparator

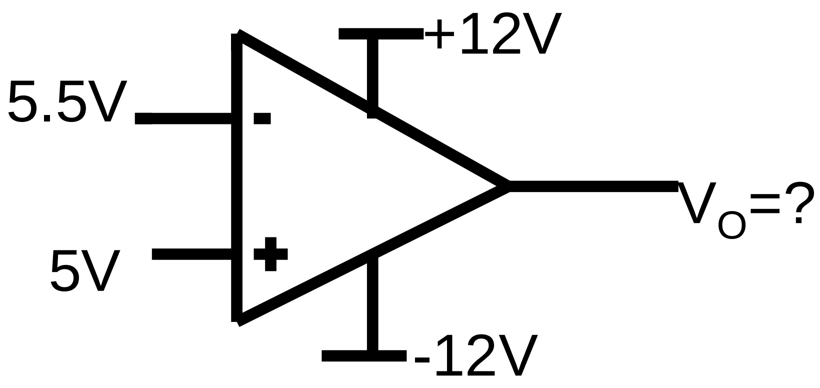
Open Loop condition



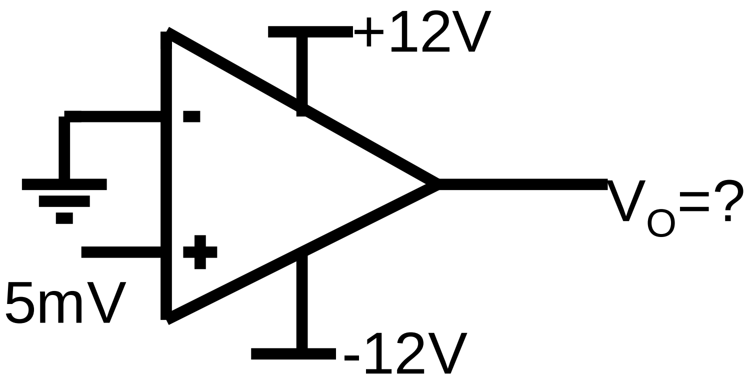




Example 1



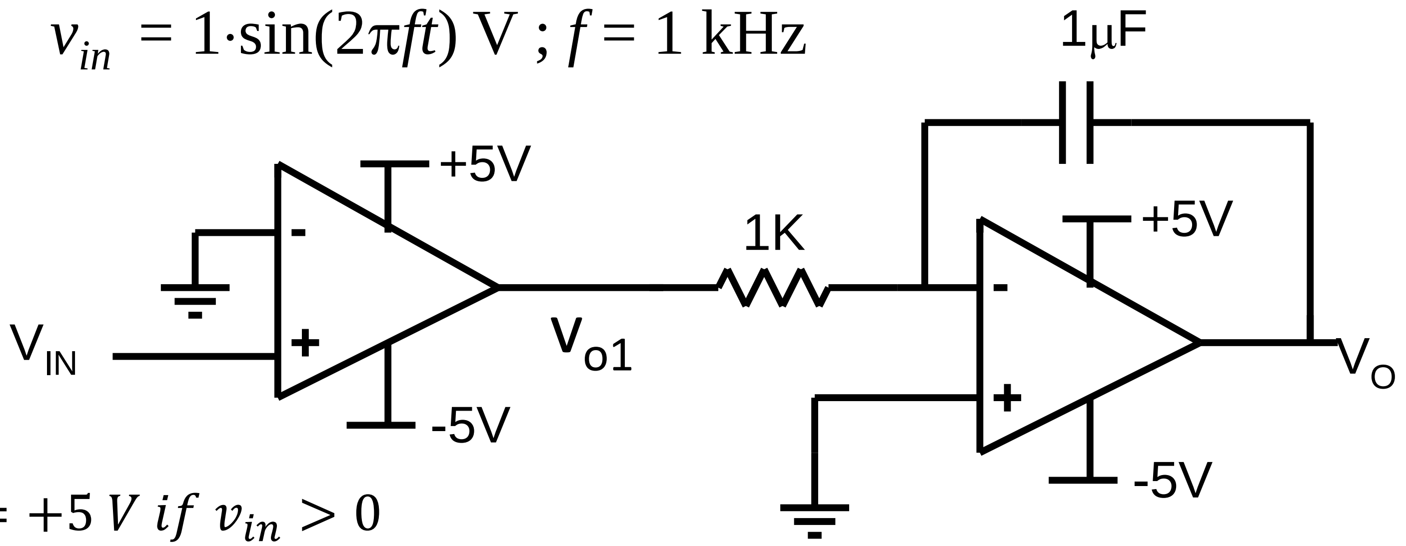
~ **-12V**



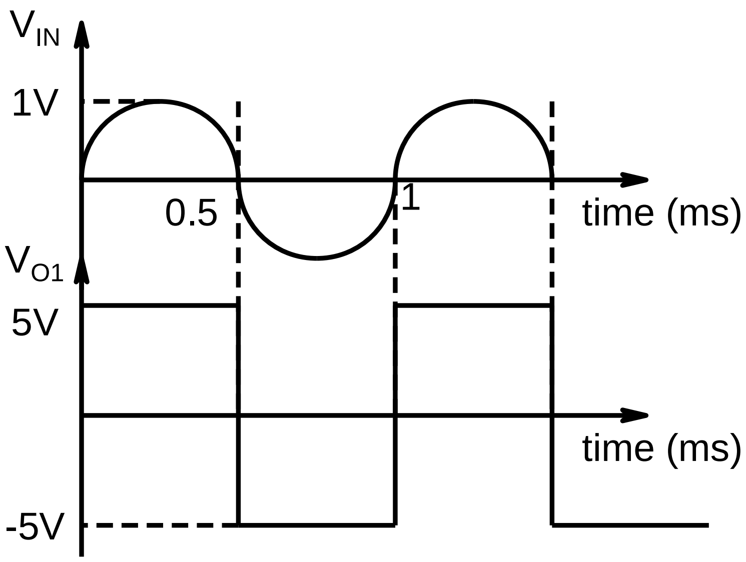
~ **+12V**

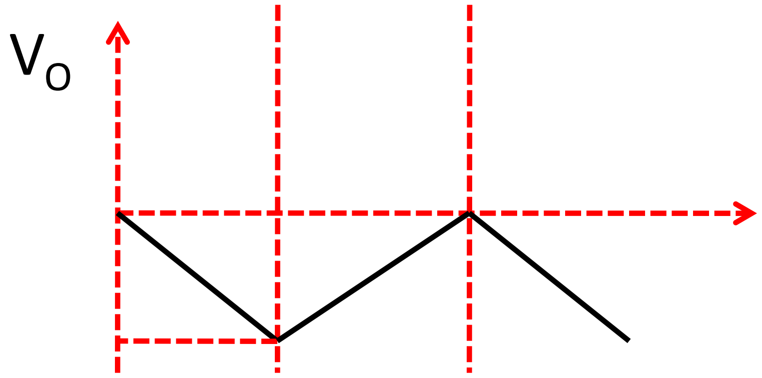
Example 2

$$v_{in} = 1 \cdot \sin(2\pi f t)\ \mathrm{V}\ ;\ f = 1\ \mathrm{kHz}$$



$$V_{O1} = +5\ V\ if\ v_{in} > 0$$
$$= -5\ V\ if\ v_{in} < 0$$

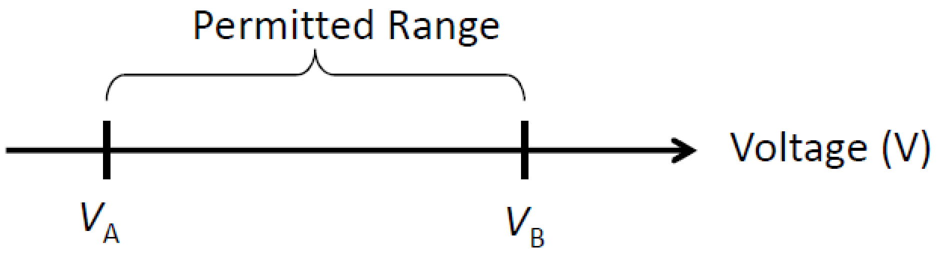




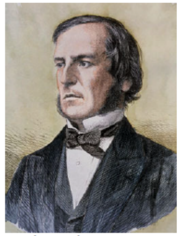
Analog and Digital

- Analog: Continuous amplitude signals.
  - If the voltage at a node (w.r.t. ground) is between V_A and V_B

Infinite values of voltage between V_A and V_B possible



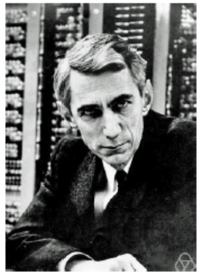
George Boole 1815-1864



Wikipedia

Served as first professor of mathematics at Queen's College, Cork, Ireland

Claude Shannon 1916 -2001

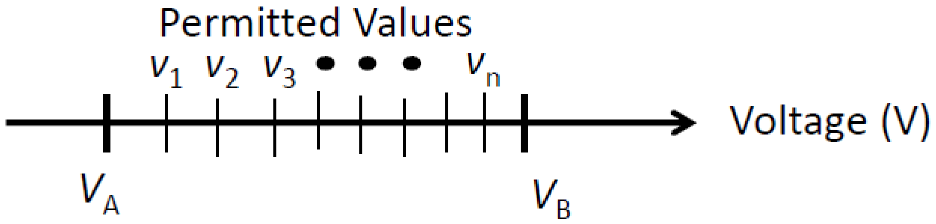


Wikipedia

Primarily Bell Labs and later MIT

- Digital: Discrete amplitude signals.
- Obtained via sampling of Analog signals or quantization.
  - Only discrete values of current and voltage in a range allowed

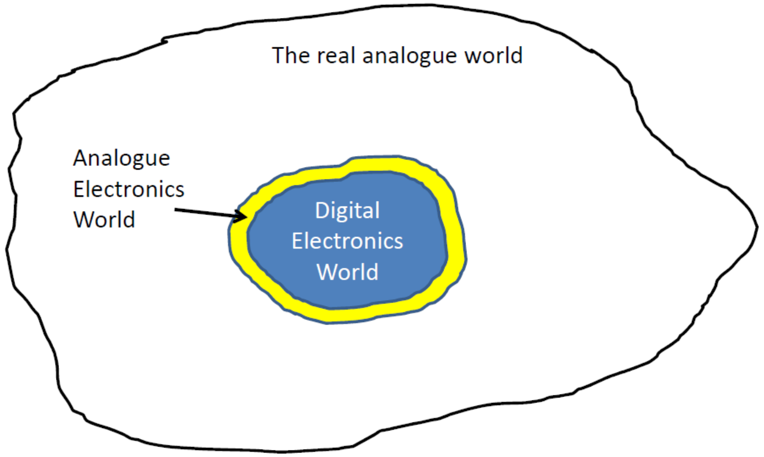
Only <u>finite</u> number of values of voltage between V_A and V_B allowed



- Digital: digit means finger, 10 fingers: base 10 numbers represented with 10 symbols (digits) 0,1….9.
- Even two digits (symbols) sufficient for arithmetic operations: George Boole: Boolean Algebra (1847).
- Foundation for Digital Electronics: MS thesis of Claude Shannon (1937).
- Father of information theory, worked with relay switches.

WHY DIGITAL?

- **Noise Margin**
- Levels separated by large distance: less impact of noise.
- **Robustness**
- **Scalability**
- Analog circuit technology stuck at 28 nm, Digital already at 3 nm.
- **Data Storage**
- Analog data is continuous: huge amount of data.
- **Ease of data handling**
- Analog signal processing is too resource intensive.
- **Fidelity/Regeneration**



MacRumors

Apple Supplier TSMC Readies 2nm Chips for 2024

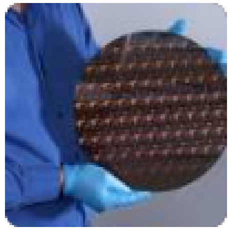
TSMC's manufacturing capabilities are also considerably more advanced than rival companies like Intel, which have been mired by delays and ...

3 days ago

Indiatimes.com

How IBM Built World's First 2-Nanometer Chip During Covid-19 Pandemic

A couple of months ago, IBM unveiled a world's first, a semiconductor design breakthrough -- something that even Intel, Apple or Samsung hadn't ...



NUMBERS

They can be represented in many ways!

		Base
Egypt		10
Babylon		10 + 60
Indian	০ ১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯ and many other varieties	10
Western Arabic Numerals	0 1 2 3 4 5 6 7 8 9	10

Some Non-10 based bases

Languages in the Nigerian Middle Belt Janji, Gbiri-Niragu, Piti, and the Nimbia dialect of Gwandara; Chepang language of Nepal, and the Mahl dialect of Maldivian; dozen-gross-great gross counting; 12-hour clock and months timekeeping; years of Chinese zodiac; foot and inch; Roman fractions; penny and shilling	12
Basque, Celtic, Maya, Muisca, Inuit, Yoruba, Tlingit, and Dzongkha numerals; Santali, and Ainu languages; shilling and pound	20
Undecimal – Māori (NZ), Pangwa (Tanzania)	11

Roman I V X L C D M

https://en.wikipedia.org/wiki/List_of_numeral_systems

Positional Notation for Number

A positional notation is commonly used to express numbers

$$(\ldots a_n a_{n-1} \ldots a_2 a_1 a_0 . a_{-1} a_{-2} \ \ldots a_{-m+1} a_{-m} \ldots)_r$$

radix point

$$= \ \ldots a_n r^n + a_{n-1} r^{n-1} \ldots + a_2 r^2 + a_1 r^1 + a_0 r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + \ldots + a_{-m+1} r^{-m+1} + a_{-m} r^{-m} + \ldots$$

$$= \sum_{i:\,-\infty}^{\infty} a_i r^i$$

- Here r is called the base or radix, a_i are coefficients and index $i \,\varepsilon\, \mathbf{I}$ here
- To represent numbers, the radix is chosen as a positive integer.
- $a_i \,\varepsilon\, \mathbf{I}$ and $0 \leq a_i < r$
- When representing integers, the $(a_i = 0) \ \forall \ i < 0$

Decimal Numbers

For decimal number representation, base or radix r = 10

'International Form of Indian Numerals' or 'Western Arabic Numerals' or 'European Numerals' symbols used to represent decimal numbers are (0,1,2,3,4,5,6,7,8,9)

$$(2009)_{10} = 2 \times 10^3 + 0 \times 10^2 + 0 \times 10^1 + 9 \times 10^0$$

$$(123.24)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 2 \times 10^{-1} + 4 \times 10^{-2}$$

decimal point

This form of representation is the most popular in the world today

The symbols used might vary, but the mathematical concept is the same

Binary Numbers

- A binary system of representation uses a base of 2
- It is the smallest base we can use to represent all numbers
- It requires only two symbols 0, 1 to represent all the numbers
- Each symbol is called a bit (symbol b)

$$(1101)_2 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0$$

Which decimal number does this correspond?

$$(1101)_2 = 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 = 13$$

binary point

1 1 0 1 . 1 0 0 1

2^3 2^2 2^1 2^0 2^{-1} 2^{-2} 2^{-3} 2^{-4}

2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
0.5	0.25	0.125	0.0625	0.03125	0.015625

2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512
2^{10}	1024 (K)*
2^{20}	1048576 (M)

* $K \equiv 2^{10}$ but $k \equiv 10^3$
(both are referred as kilo)

Examples

Developing Fluency with Binary Numbers

Binary	Decimal
1 1 0 0 1 = ?	25
1100001 = ?	64+32+1=97
0.101 = ?	0.5+0.125=0.625
11.001 = ?	3+0.125=3.125

An Octal System

An octal system of representation uses a base of 8

It needs 8 symbols which may be (0,1,2,3,4,5,6,7) – a subset of decimal symbols

$$(2007)_8 = 2 \times 8^3 + 0 \times 8^2 + 0 \times 8^1 + 7 \times 8^0$$

What decimal number does it represent?

$$(2007)_8 = 2 \times 512 + 0 \times 64 + 0 \times 8 + 7 \times 1 = (1031)_{10}$$

What binary number does it represent?

$$(2007)_8 = (010\ 000\ 000\ 111)_2$$

Notice the ease of conversion from binary to octal (and vice versa)

A Hexadecimal System

Number	Symbol
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

A hexadecimal system uses a base of 16

Colloquially shortened to 'hex' system

$$(2BC9)_{16} = 2 \times 16^3 + B \times 16^2 + C \times 16^1 + 9 \times 16^0$$

How do we convert it into decimal number representation?

$$(2BC9)_{16} = 2 \times 4096 + 11 \times 256 + 12 \times 16 + 9 \times 1$$
$$= (11209)_{10}$$

How do we represent the number in binary?

$(2BC9)_{16}$ = $(0010\ 1011\ 1100\ 1001)_2$

- Notice ease of conversion from hex to binary (and vice versa)
- Eight bits (or two hex symbol equivalent in binary) is called: a byte (symbol B)
- Four bits (or one hex symbol equivalent in binary) is called: a nibble
- Hex representation is compact to depict large numbers

Example 1

Converting decimal integer to binary represenation

Convert 45 to binary number

$$(45)_{10} = b_n b_{n-1} \ldots\ldots b_0$$

$$45 = b_n 2^n + b_{n-1} 2^{n-1} \ldots\ldots b_1 2^1 + b_0$$

Divide both sides by 2

$$\frac{45}{2} = 22.5 = b_n 2^{n-1} + b_{n-1} 2^{n-2} \ldots\ldots b_1 2^0 + b_0 \times 0.5$$

$$22 + 0.5 = b_n 2^{n-1} + b_{n-1} 2^{n-2} \ldots\ldots + b_1 2^0 + b_0 \times 0.5$$

$$\Rightarrow \; b_0 = 1$$

Example 1 (continued)

$$22+0.5= b_n 2^{n-1} + b_{n-1} 2^{n-2} .... + b_1 2^0 + b_0 \times 0.5 \qquad \Rightarrow \; b_0 = 1$$

$$22= b_n 2^{n-1} + b_{n-1} 2^{n-2} .... b_2 2^1 + b_1 2^0$$

Divide both sides by 2

$$\frac{22}{2} = 11 = b_n 2^{n-2} + b_{n-1} 2^{n-3} .... b_2 2^0 + b_1 \times 0.5 \qquad \Rightarrow \; b_1 = 0$$

$$11 = b_n 2^{n-2} + b_{n-1} 2^{n-3} ...... + b_3 2^1 + b_2 2^0$$

$$5.5 = b_n 2^{n-3} + b_{n-1} 2^{n-4} ...... + b_3 2^0 + 0.5 b_2 \qquad \Rightarrow \; b_2 = 1$$

$$5 = b_n 2^{n-3} + b_{n-1} 2^{n-4} ...... b_4 2^1 + b_3 2^0$$

Example 1 (continued)

$$5 = b_n 2^{n-3} + b_{n-1} 2^{n-4} \ldots\ldots b_4 2^1 + b_3 2^0$$

$$2.5 = b_n 2^{n-4} + b_{n-1} 2^{n-5} \ldots\ldots b_4 2^0 + 0.5 b_3 \quad \Rightarrow \; b_3 = 1$$

$$2 = b_n 2^{n-4} + b_{n-1} 2^{n-5} \ldots\ldots b_5 2^1 + b_4 2^0$$

$$1 = b_n 2^{n-5} + b_{n-1} 2^{n-6} \ldots\ldots b_5 2^0 + 0.5 b_4 \quad \Rightarrow \; b_4 = 0$$

$$\Rightarrow \; b_5 = 1$$

$$(45)_{10} = b_5 b_4 b_3 b_2 b_1 b_0 = 101101$$

Example 1 (continued)

Converting decimal integer to binary representation

Method of successive division by 2

45	remainder
22	1
11	0
5	1
2	1
1	0
0	1

45 = 1 0 1 1 0 1

Example 2

Convert $(153)_{10}$ to octal number system

$$(153)_{10} = (b_n b_{n-1}\ldots\ldots b_0)_8$$

$$(153)_{10} = b_n 8^n + b_{n-1} 8^{n-1} \ldots\ldots b_1 8^1 + b_0$$

Divide both sides by 8

$$\frac{153}{8} = 19.125 = b_n 8^{n-1} + b_{n-1} 8^{n-2} \ldots\ldots b_1 8^0 + \frac{b_0}{8} \quad \Rightarrow \quad \frac{b_0}{8} = 0.125 \quad \Rightarrow \quad b_0 = 1$$

153	remainder
19	1
2	3
0	2

153 = $(231)_8$

Example 3

Converting a fraction in decimal to binary representation

Convert $(0.35)_{10}$ to binary number

$$(0.35)_{10} = 0.b_{-1}b_{-2}b_{-3}.......b_{-n}$$

$$0.35 = 0 + b_{-1}2^{-1} + b_{-2}2^{-2} + ...b_{-n}2^{-n}$$

Note that ½+1/4+1/8+……≤1

If it is 0.1111111… or 1 recurring, the number is equal to 1.0

How do we find the b_{-1} b_{-2} …coefficients?

Multiply both sides by 2

$$0.7 = b_{-1} + b_{-2}2^{-1} + ...b_{-n}2^{-n+1} \qquad \Rightarrow \; b_{-1} = 0$$

$$0.7 = b_{-2}2^{-1} + b_{-3}2^{-2} + ...b_{-n}2^{-n+1}$$

Example 3 (continued)

$$0.7 = b_{-2}2^{-1} + b_{-3}2^{-2} + ...b_{-n}2^{-n+1}$$

Multiply both sides by 2

$$1.4 = b_{-2} + b_{-3}2^{-1} + ...b_{-n}2^{-n+2} \quad \Rightarrow \; b_{-2} = 1$$

$$0.4 = b_{-3}2^{-1} + b_{-4}2^{-2}.......b_{-n}2^{-n+2}$$

$$0.8 = b_{-3} + b_{-4}2^{-1}.......b_{-n}2^{-n+3} \quad \Rightarrow b_{-3} = 0$$

$1.6 \Rightarrow b_{-4} = 1;$ $\quad 1.2 \Rightarrow b_{-5} = 1;$ $\quad 0.4 \Rightarrow b_{-6} = 0;$ $\quad 0.8 \Rightarrow b_{-6} = 0; \ldots$

It keeps recurring after this.

So $(0.35)_{10} = (0.010110011001\ldots)_2 = (0.010110011)_2$

We represent bits till we get sufficient accuracy as required or are allowed

Example 4

Converting decimal fraction to binary number

0.125 = ?

0 .	125
	x2
0 .	25
	x2
0 .	5
	x2
1.	0

0.125 = $(.001)_2$

0.8125 = ?

0 .	8125
	x2
1 .	625
	x2
1 .	25
	x2
0.	5
	x2
1.	0

0.8125 = $(.1101)_2$

More on Binary Representation

decimal	2bit	3bit	4bit	5bit
0	00	000	0000	00000
1	01	001	0001	00001
2	10	010	0010	00010
3	11	011	0011	00011
4		100	0100	00100
5		101	0101	00101
6		110	0110	00110
7		111	0111	00111
8			1000	01000
9			1001	01001
10			1010	01010
11			1011	01011
12			1100	01100
13			1101	01101
14			1110	01110
15			1111	01111

Least significant bit or LSB

1011000111

Binary digit = bit

Most significant bit or MSB

The above is a 10 bit binary number

A 10 bit number can represent:

- 1024 numbers
- These could be 0 to 1023

N-bit binary number can represent

- 2^N numbers
- numbers from 0 to 2^N -1

Hex to Binary and Binary to Hex

Number	Symbol
0(0000)	0
1(0001)	1
2(0010)	2
3(0011)	3
4(0100)	4
5(0101)	5
6(0110)	6
7(0111)	7
8(1000)	8
9(1001)	9
10(1010)	A
11(1011)	B
12(1100)	C
13(1101)	D
14(1110)	E
15(1111)	F

$$(b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0)_b = (h_1, h_0)_{Hex}$$

$$b_7 2^7 + b_6 2^6 + b_5 2^5 + b_4 2^4 + b_3 2^3 + b_2 2^2 b_1 2^1 + b_0 = h_1 16^1 + h_0$$

$$(\underbrace{b_7 2^3 + b_6 2^2 + b_5 2^1 + b_4}_{h_1})2^4 + (\underbrace{b_3 2^3 + b_2 2^2 b_1 2^1 + b_0}_{h_0}) = h_1 16^1 + h_0$$

$$(10110011)_b = (1011)(0011) = (B3)_{Hex}$$

$$(110011)_b = (11)(0011) = (33)_{Hex}$$

$$(EC)_{Hex} = (1110)(1100) = (11101100)_b$$

Binary Addition

$$\begin{array}{r} 0 \\ \underline{0} \\ \underline{0} \end{array} \qquad \begin{array}{r} 1 \\ \underline{0} \\ \underline{1} \end{array} \quad \begin{array}{r} 0 \\ \underline{1} \\ \underline{1} \end{array} \qquad \begin{array}{r} 1 \\ \underline{1} \\ \underline{1\ 0} \end{array} \qquad \begin{array}{r} 1 \\ 1 \\ \underline{1} \\ \underline{1\ 1} \end{array}$$

$$\begin{array}{r} 1\,0\,1 \\ \underline{1\,1\,0} \\ 1\,0\,1\,1 \end{array} \qquad \begin{array}{r} 1\,1\,0\,1 \\ +\ \underline{1\,1\,1\,0} \\ 1\,1\,0\,1\,1 \end{array}$$

Complements of Numbers

- Pairing up of numbers in given number-space
- Complement of complement is original number

- For n digits representation of number X with base r :
  - (r-1)'s complement → $(r^n - 1) - X$
    - Each digit of X is subtracted from r
  - r's complement → $r^n - X$

- This can be useful to pair up numbers
- For example, defining positive and negative numbers

Complements of Decimal Numbers

Decimal system: → 9's complement

→ 10's complement

9's complement of n-digit number X: $(10^n - 1) - X = \underbrace{99....9999}_{\text{n 9s}} - X$

Largest number representable with n digits

10's complement of n-digit number X is $10^n - X = \underbrace{100...0000}_{\text{1 followed by n 0s}} - X$

Example

9's complement of 85 ? $(10^2 - 1) - 85$ = $99 - 85 = 14$

10's complement of 85 ? $10^2 - 85$ = 15 which is (9's complement) + 1

Example

$9's \text{ complement of } 123 \;= 999 - 123 = 876$

$10's \text{ complement of } 123 = 1000 - 123 \;= 877 = (9's \text{ complement of } 123){+}1$

1's Complement of a Binary Number

Binary system: 1's complement / 2's complement

1's complement of n-bit number X is $(2^n - 1) - X$ $= \overbrace{11\ldots1111}^{\text{n 1s}} - X$

Example

1's complement of 1011 ? $(2^4 - 1) - 1011 = 1111 - 1011 = 0100$

1's complement is simply obtained by flipping a bit (changing 1 to 0 and 0 to 1)

Example

$1\text{'s complement of } 1001101 = ?$ 0110010

Finding 2's Complement of a Binary Number

Basic Definition

2's complement of n-bit number X is 2^n - X $= \overbrace{100\ldots0000}^{\text{1 followed by n 0s}} - X$

i.e., 2's complement is 1's complement + 1

Example

$2's\text{ complement of }1010 = 10000 - 1010 = 0110$

$2's\text{ complement of }1010 = (1's\text{ complement of }1010) + 1 = 0101 + 1 = 0110$

Another algorithm to find 2's complement of a binary number:

Leave all least significant 0's as they are,

leave first 1 unchanged

and then flip all subsequent bits

Examples

$2's\text{ complement of}$

$110010 \leftrightarrow 001110$

$101101100 \leftrightarrow 010010100$

$1011 \leftrightarrow 0101$

Arithmetic Including Negative Numbers

- A digital system has finite number of bits
- For n bits available, 2^n unique numbers can be represented

- There is need to be able to represent negative number
- We would like a link between negative and positive number
    - Likely to make the math easy
- We would like to have a unique representation of zero

- We would like to do arithmetic (addition and subtraction)
- Positive and negative numbers are generated during arithmetic operations

- Finite size available to represent numbers will bring in constraints
- But we want to optimise as much as possible within the constraints

Representing Positive and Negative Numbers

Extra bit needed to carry sign information
"MSB" is often the sing bit

One option
Sign bit = 0 represents non-negative nos.
Sign bit = 1 represents negative numbers

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

magnitude
magnitude

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

magnitude
magnitude

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

magnitude
magnitude

Unique zero representation

$2^{n-1} - 1$ positive nos.

2's comp. of 0 and -8 are themselves

2^{n-1} negative nos.