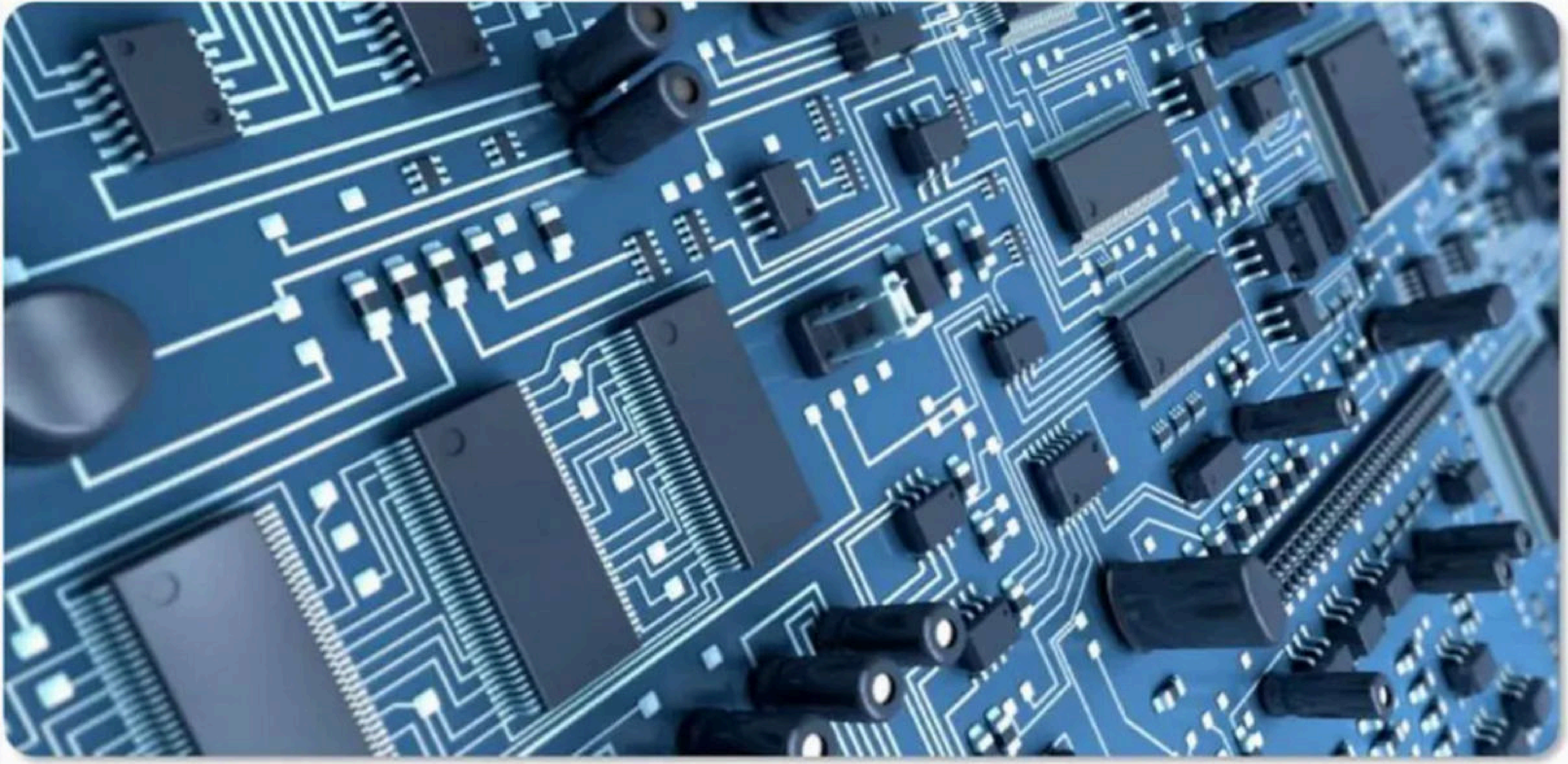
Boolean Algebra & Expression - Part I

Complete Course on Digital Electronics - GATE, 2023

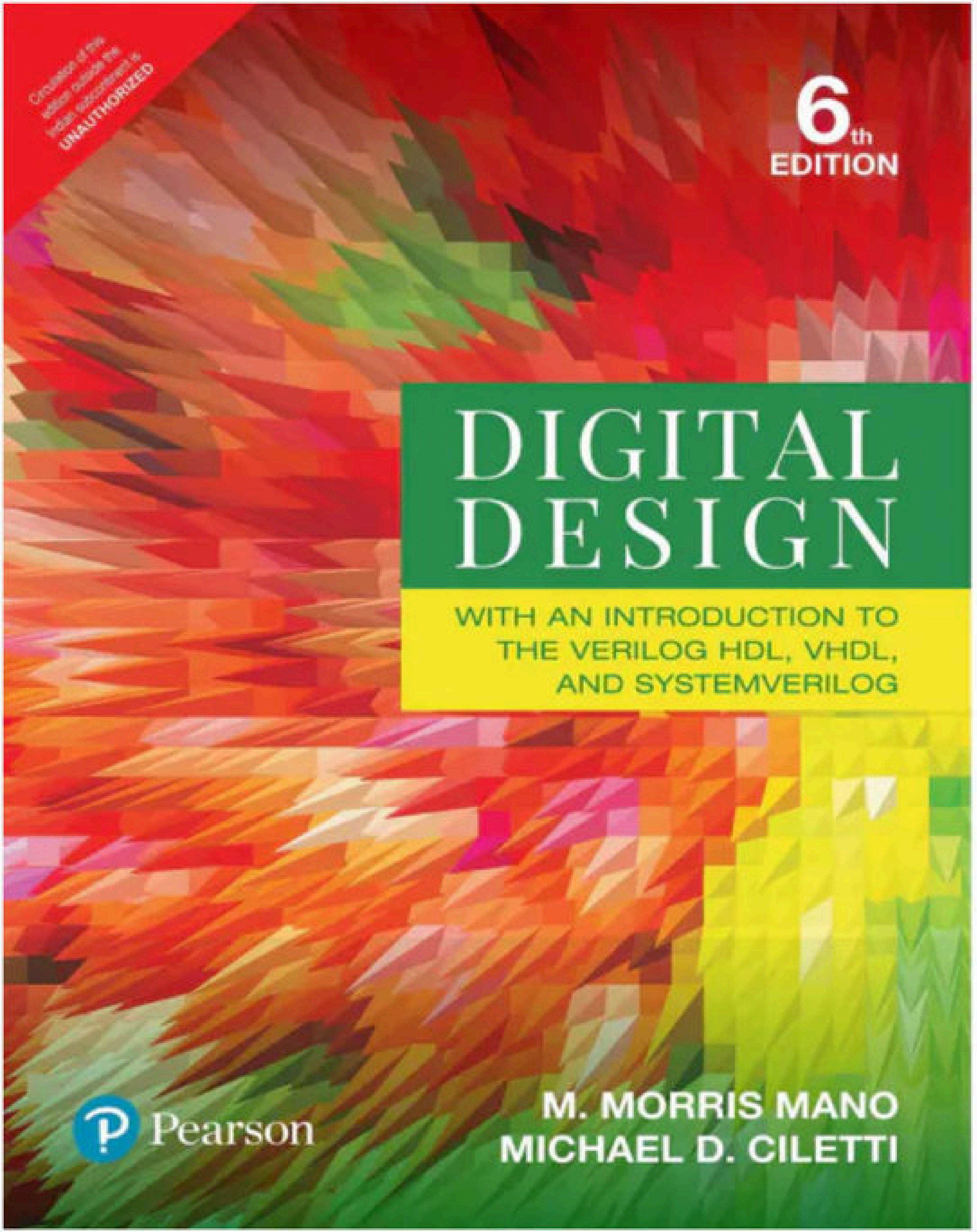
Sanchit Jain • Lesson 1 • Nov 29, 2022

Digital Electronics



Digital Electronics + COA ⇛ 15

- Core subjects for CS/IT Students
- In GATE 7-8 Marks out of 100 Marks, and 5-6 questions on an average
- In NET 20-22 Marks out of 200 marks and 10-12 questions
- Most questions are Numerical
- Needs less time, good scoring
- Not asked in Industry

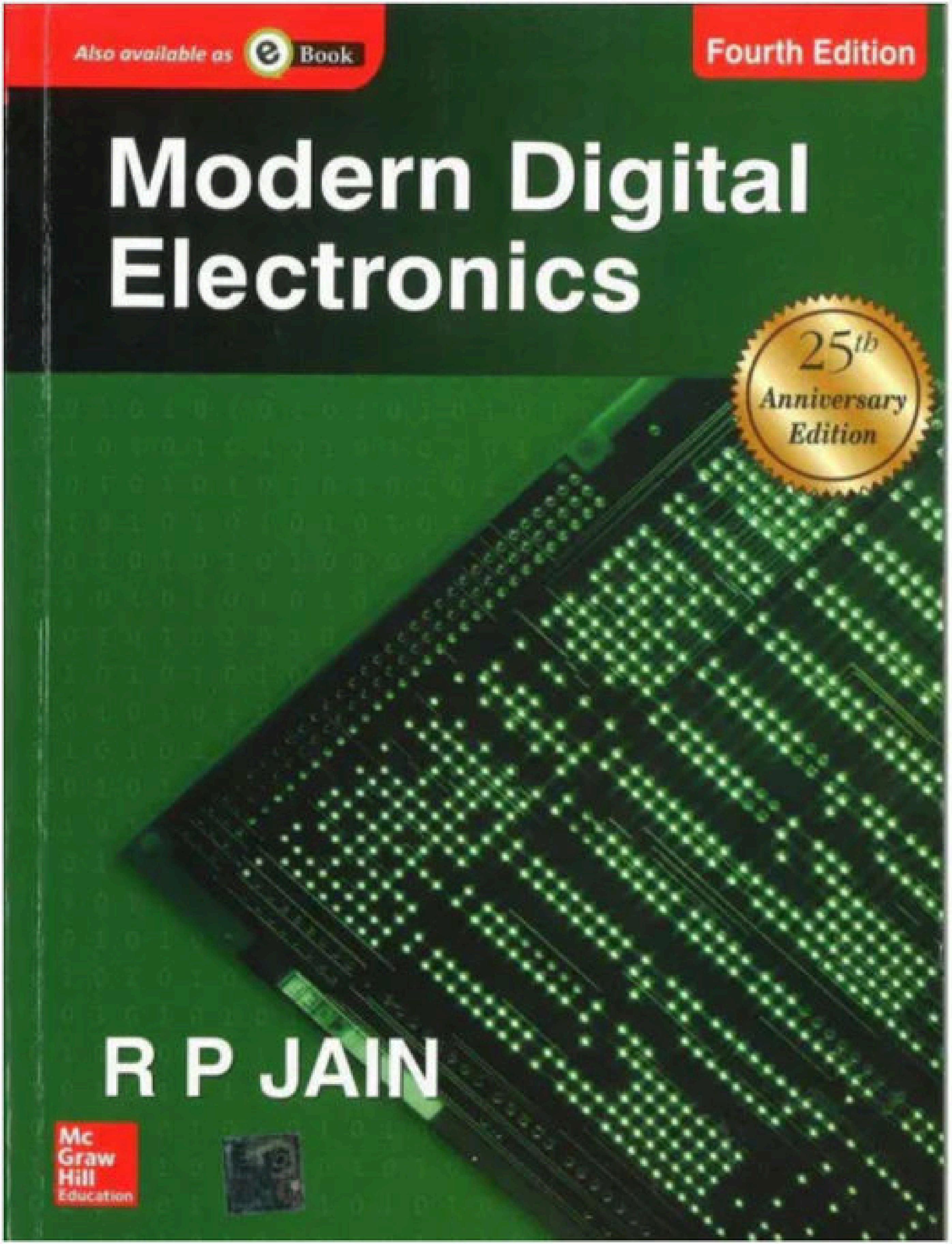


- **Book Name** - Digital Design

- **Writers** -
  - M. Morris Mano
  - Michel D. Ciletti

- **Publisher** – Pearson

- **Edition** – 6th



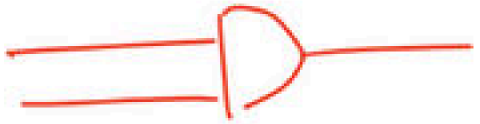
- Book Name – Modern Digital Electronics
- Writers – R P Jain
- Publisher – McGraw Hill
- Edition – 4th

COA

Section 2: Digital Logic

+ COA

Boolean algebra. Combinational and sequential circuits. Minimization. Number representations and computer arithmetic (fixed and floating point).



Syllabus

- Understanding Digital Electronics (History and Motive) 0
- Boolean Algebra Laws and Logic Gates (1-2)
- Boolean Expression (SOP and POS) (Minimization) 2
- Combinational Circuit (1-2)
- Sequential Circuit 2
- Number System and Number Representation (1-2)

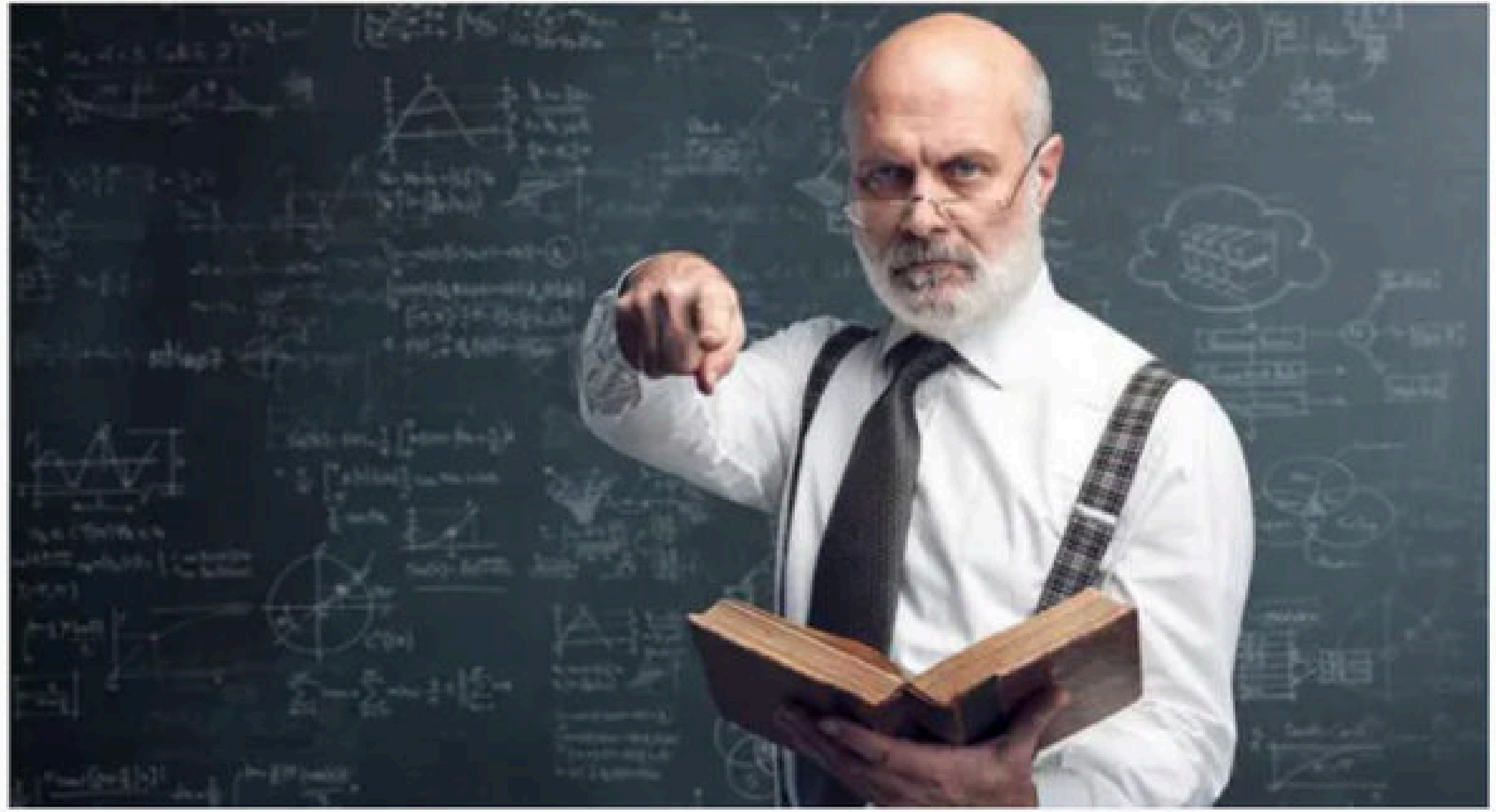
1' 2' Sm

What you can expect from me

- Will take care of theory and numerical both
- Will give more weightage to the topics that are asked more frequently in GATE
- Will not emphasize more than required, on a topic
- Will provide PDF of related books

What i expect from you

- There is no hurry, feel free to ask questions any time through out the class, but first listen
- Please revise the entire lecture before and after the class
- Be regular, Consistency is most important
- More you practice, more clarity you will get
- If we follow all the above specified points Success in guaranteed

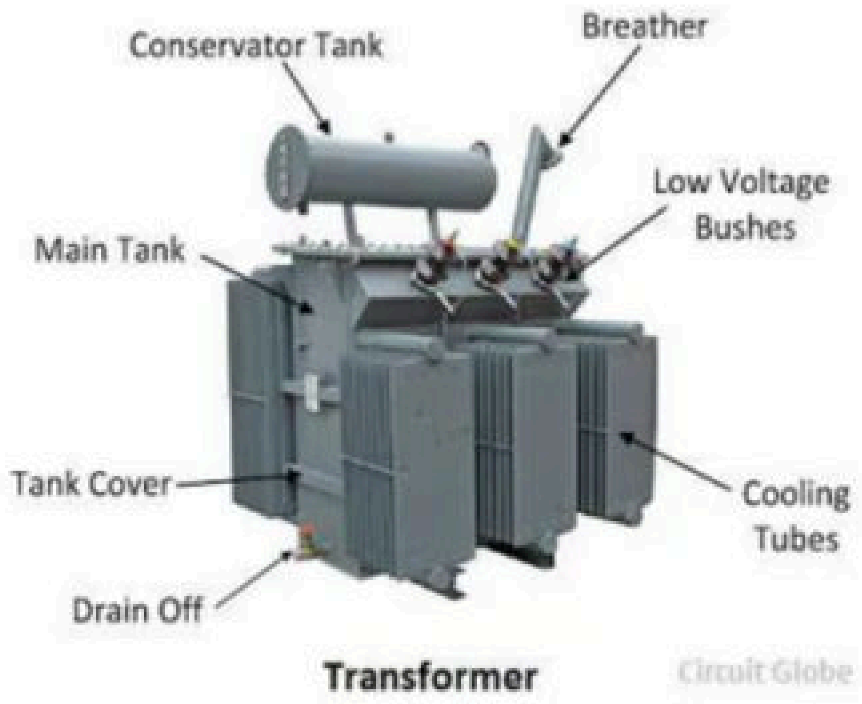


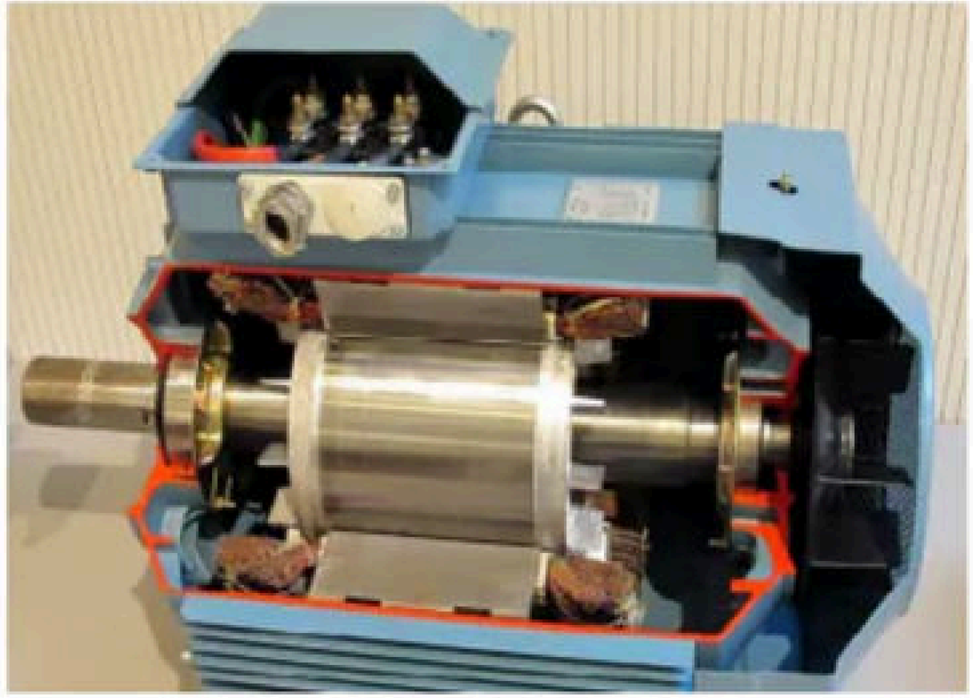
Break

Basics

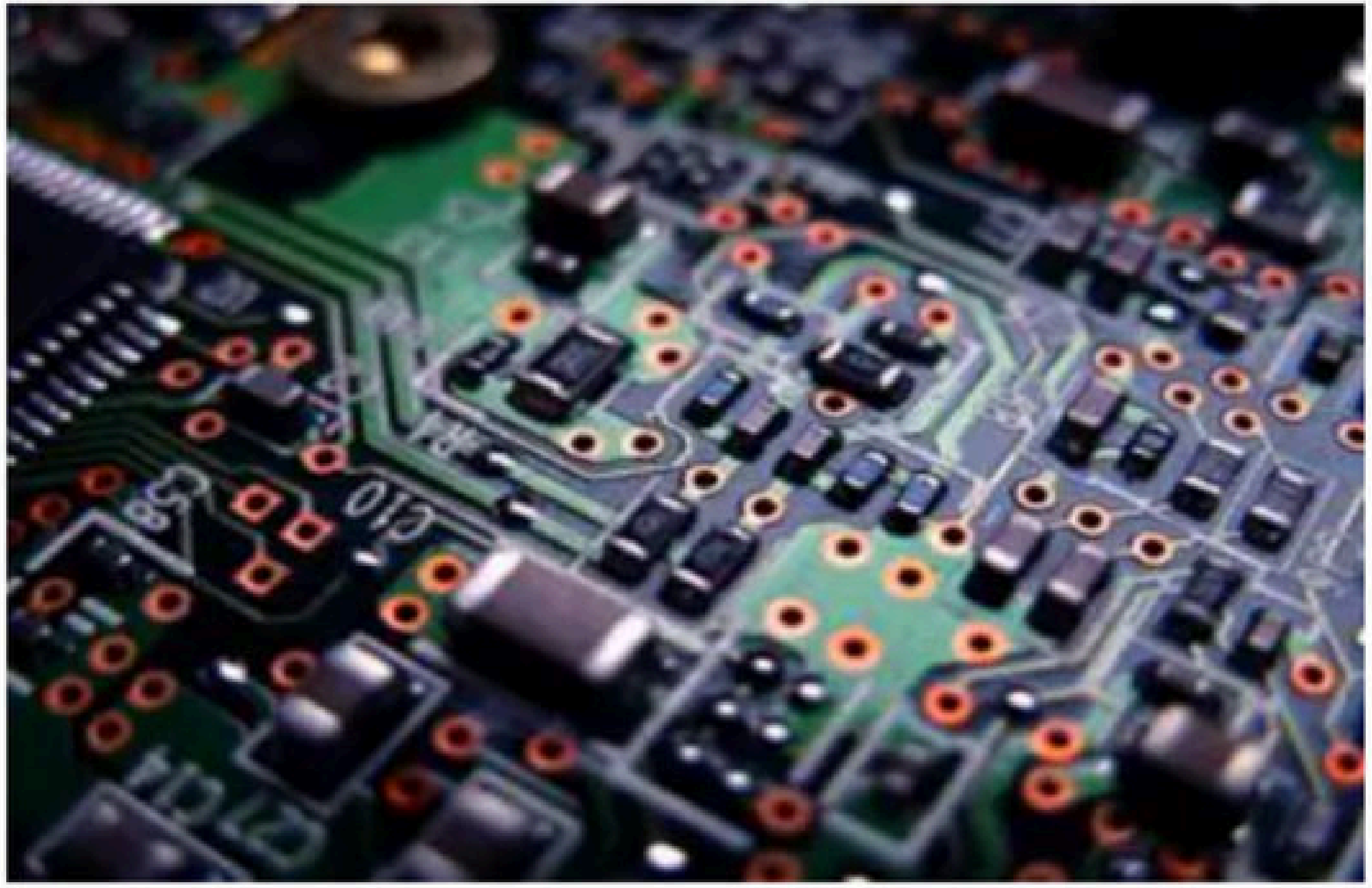
- **Electrical engineering** is a professional engineering discipline that generally deals with the study and application of electricity, electronics and electromagnetism and heavy voltage devices like transformers, motors etc.

1985 →

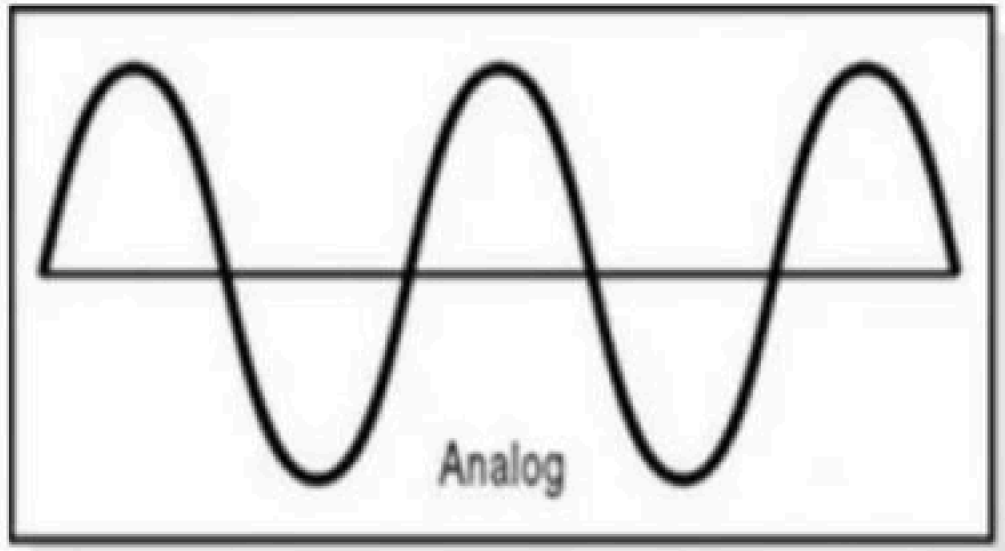


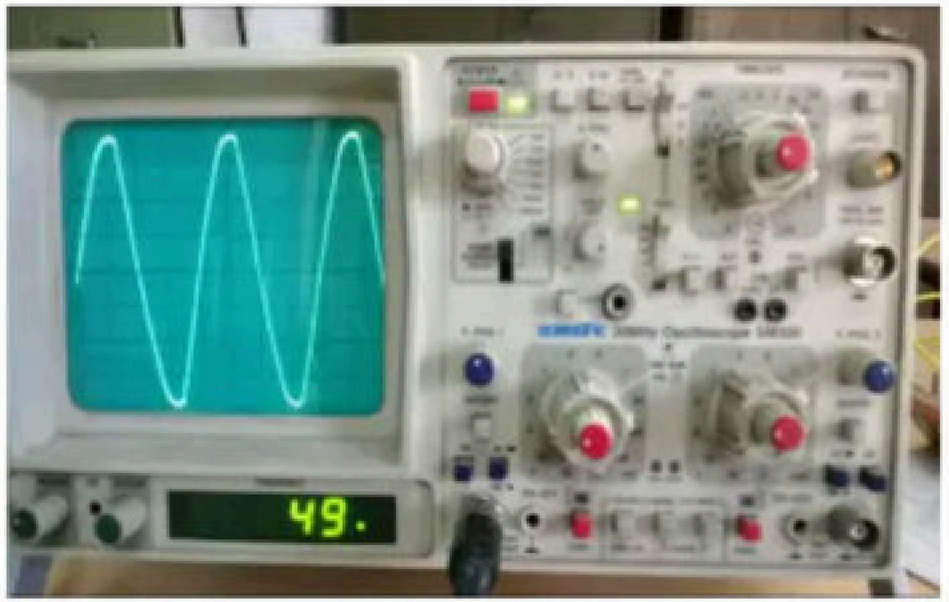


- **<u>Electronic engineering</u>** is an electrical engineering discipline where we work on low voltage devices (such as semiconductor devices, especially transistor, diodes and integrated circuits) to design electronic circuits, VLSI devices and systems.

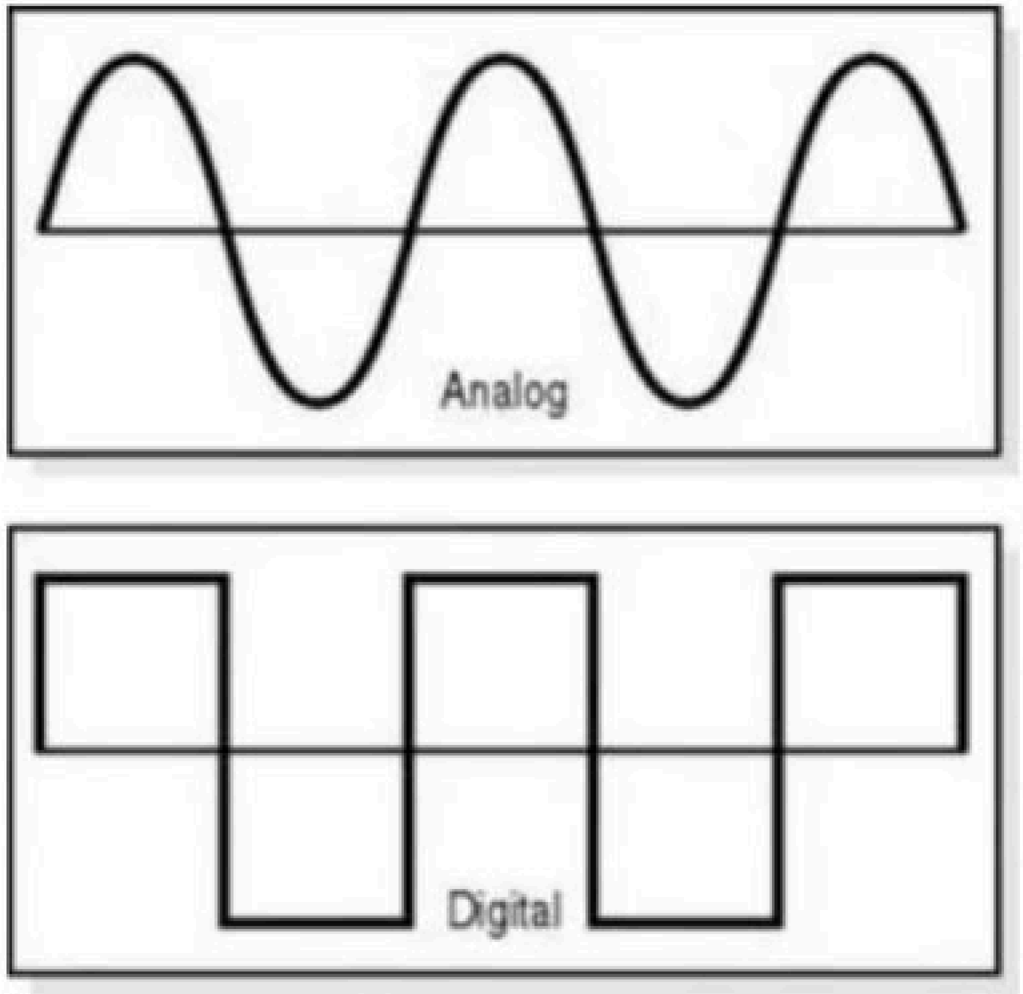


- Electronic systems are generally of two types –
  - **Analog system** in an analog representation, a continuous value is used to denote the information. Analog signal is defined as any physical quantity which varies continuously with respect to time. e.g. amplifier, cro, ecg. Watch, radio.





- **Digital system** – the information is denoted by a finite sequence of discrete value or digits. Digital signal is defined as any physical quantity having discrete values. E.g. digital watch, calculator, bp machine, thermometer etc.



Advantage of Digital system

1) Digital system are easy to design because they do not require sound engineering and mathematical knowledge, uses only switching circuit.

2) Storage for long time and processing of information is easy.

VCD → CD → PD →

3) They provide better accuracy and precision compared to analog devices, as digital devices are less affected by noise, sound, electric or magnetic field etc.

4) Better modularity and easy fabrication. It means the of digital devices are very small compared to analog devices. E.g. integrated circuits.

10 → 2

5) Less cost.

W →

K →

C →

→ 25

6,00,000 ⟶ 4,50,000

40,00,000 ⟶ 35,00,000

20 ——

Disadvantage of Digital system

- Only analog signal is available in the real world, so an extra hardware is required to convert this analog signal to digital signal by using analog to digital converter.

Conclusion

- Digital systems have such a prominent role in everyday life that we refer to the present technological period as digital age.

- Digital system are used in communication, business, transactions, traffic control, space guidance, medical treatment, weather forecasting, the internet, and many other commercial, industrial and scientific enterprises.

Digital India
Power To Empower

Note

- Early digital computers were used for numeric computation. The discrete elements were the digits, from this application, the term digital computer emerged.

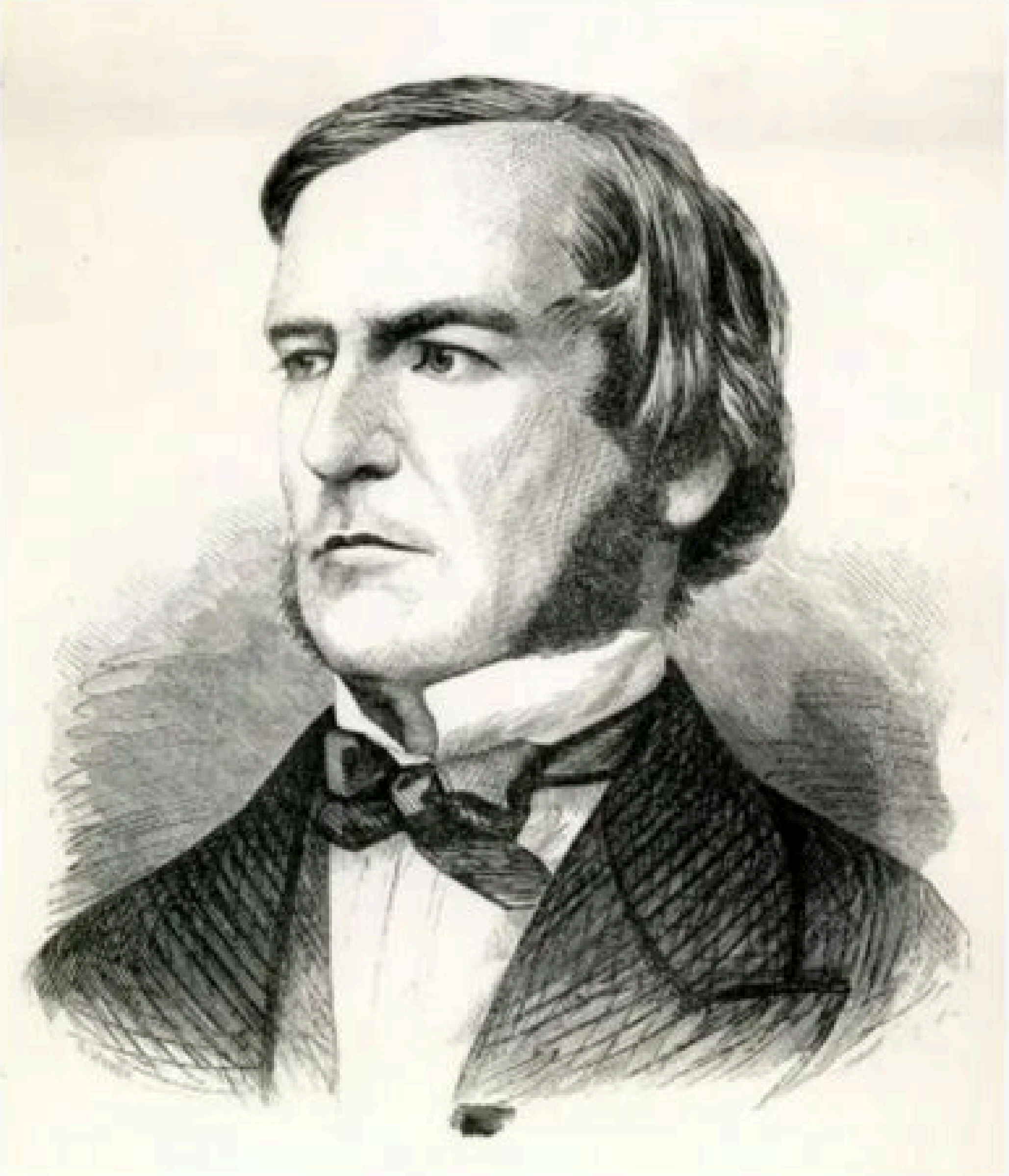
Break

Boolean algebra

- The signal in most present day electronic digital system uses just two discrete values and are therefore said to be binary.

- Boolean algebra was introduced by George Boole in his first book The Mathematical Analysis of Logic (1847), and set forth more fully in his An Investigation of the Laws of Thought (1854).

- George boole introduced the concept of binary number system in the studies of the mathematical theory of logic and developed its algebra known as Boolean algebra.



Boolean algebra

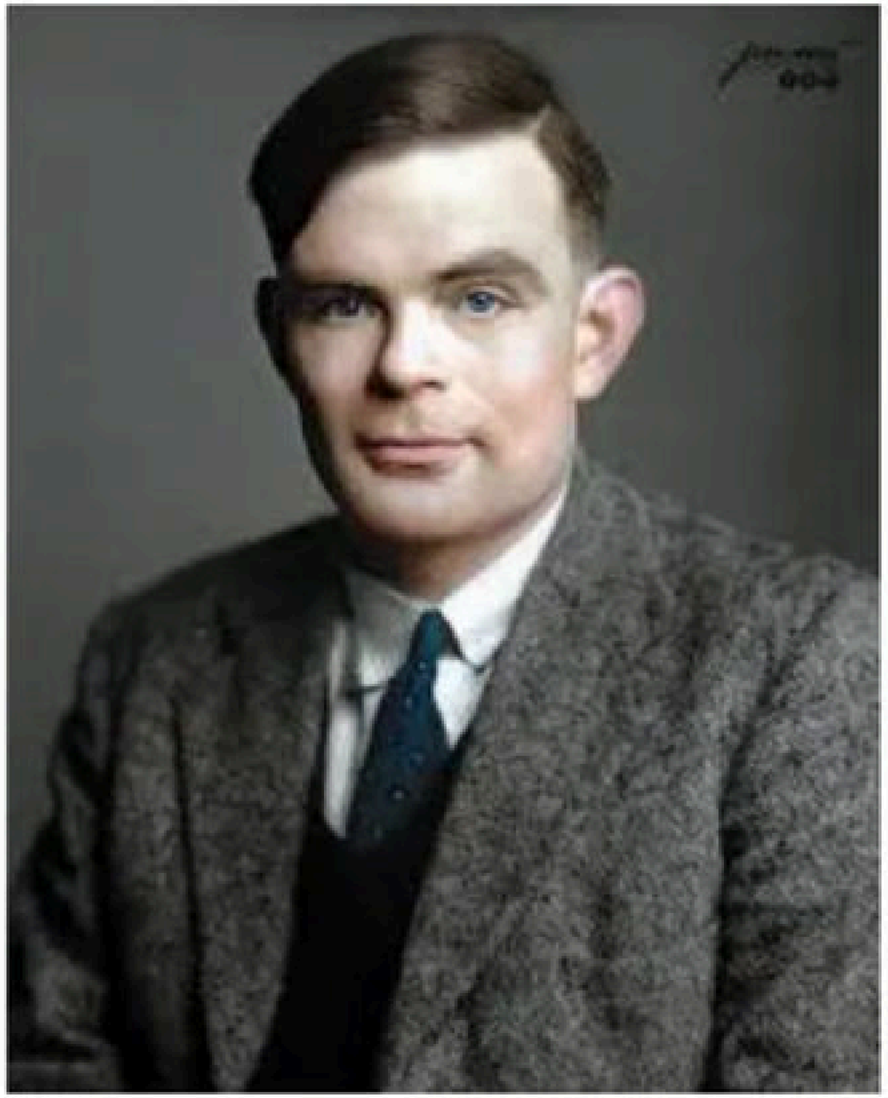
1. In mathematics and mathematical logic, Boolean algebra is the branch of algebra in which the values of the variables are the truth values true and false, usually denoted 1 and 0 respectively.

2. Instead of elementary algebra where the values of the variables are numbers, and the prime operations are addition and multiplication. The main operations of Boolean algebra are
   - The conjunction and denoted as $\wedge$
   - The disjunction or denoted as $\vee$
   - The negation not denoted as $\neg$

Turing Machine

- *The Church-Turing thesis states that any algorithmic procedure that can be carried out by human beings/computer can be carried out by a Turing machine.(1936)*

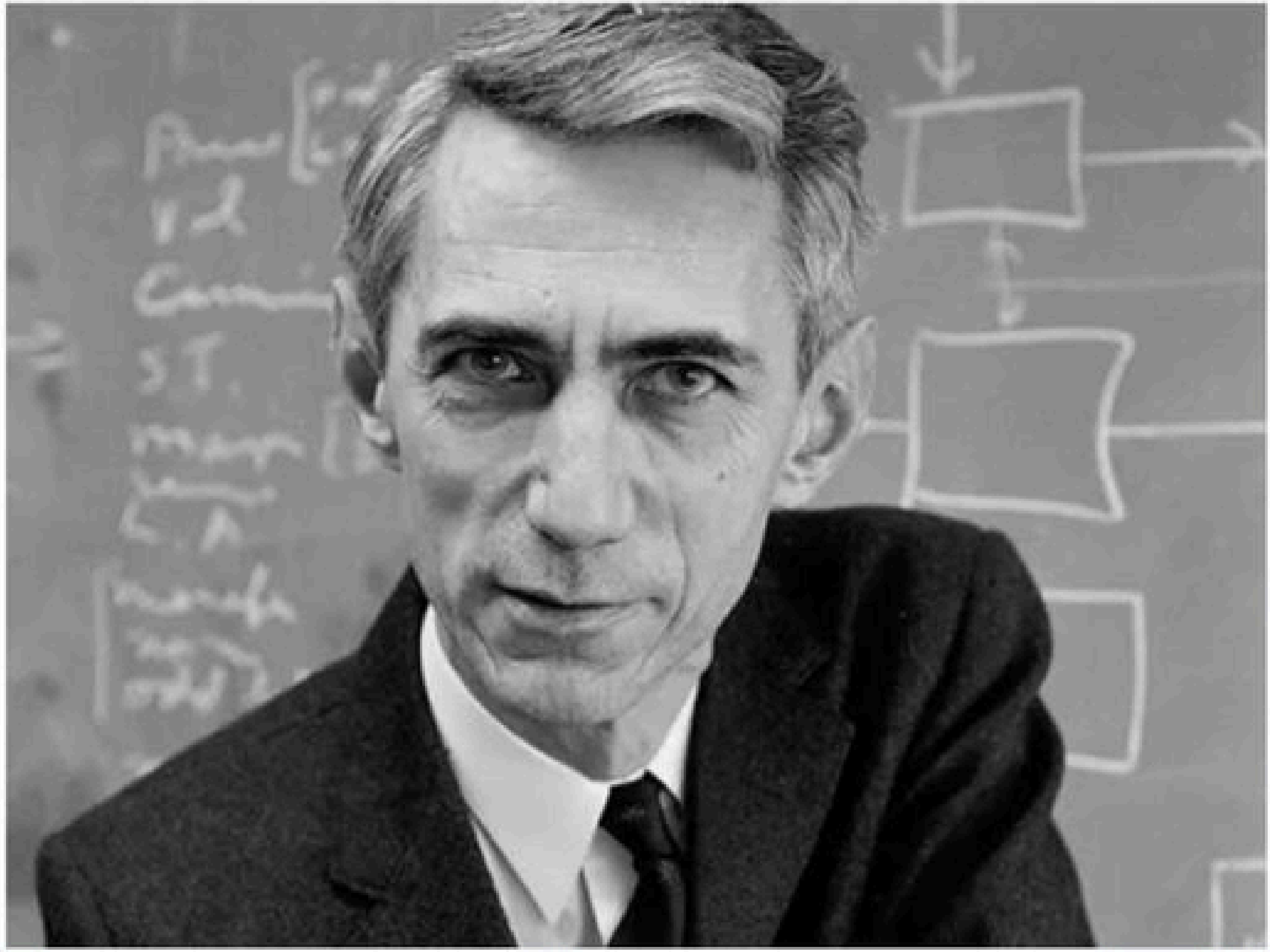
- It has been universally accepted by computer scientists that the Turing machine provides an ideal theoretical model of a computer.



Digital System

- In the 1930s, while studying switching circuits, Claude Shannon observed that one could also apply the rules of Boole's algebra in this setting, and he introduced switching algebra as a way to analyze and design circuits by algebraic means in terms of logic gates.

- These logic concepts have been adapted for the design of digital hardware since 1938 Claude Shannon (father of information theory), organized and systematized Boole’s work.



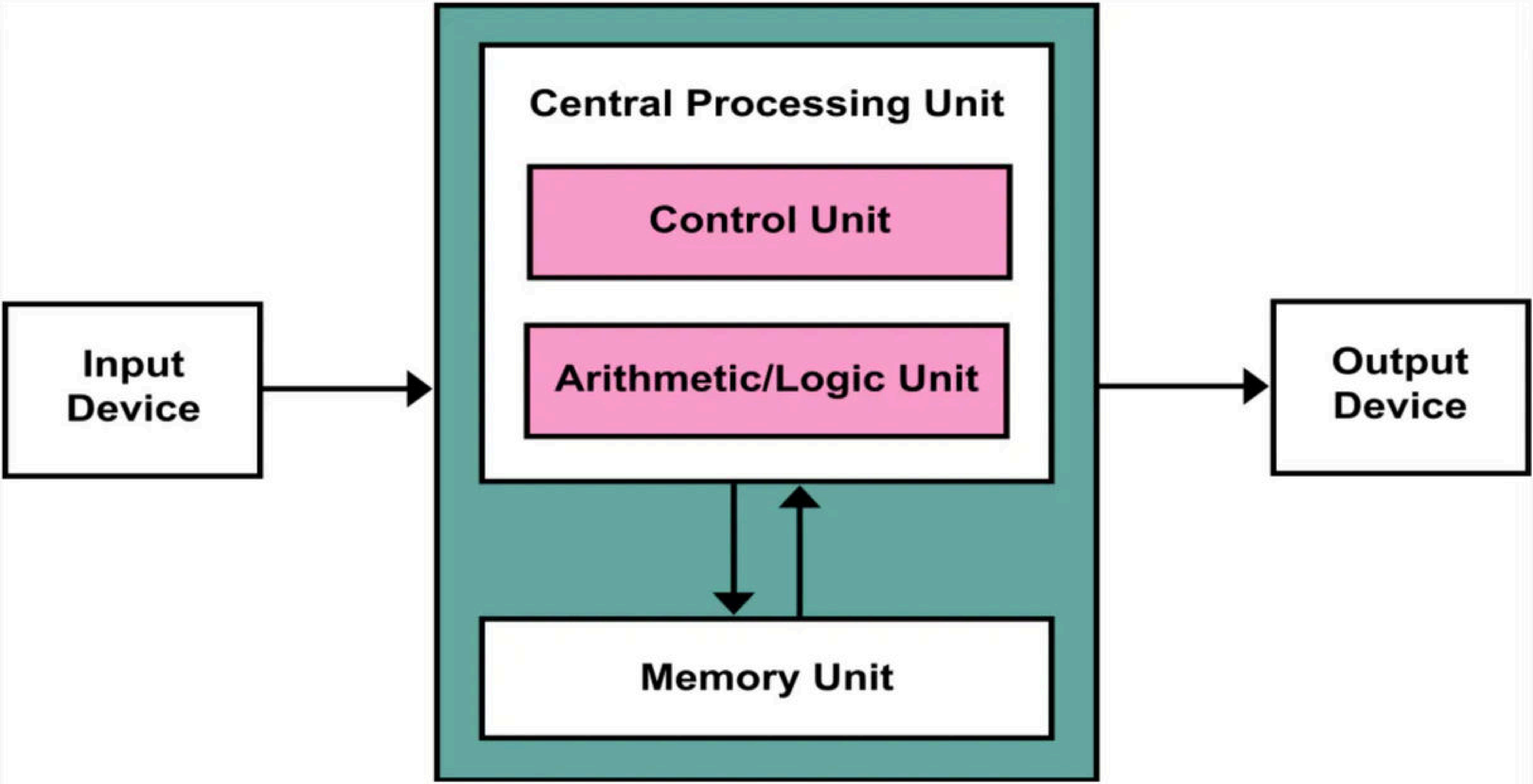
1) Shannon already had at his disposal the Boolean algebra, thus he cast his switching algebra as the two-element Boolean algebra. Efficient implementation of Boolean functions is a fundamental problem in the design of combinational logic circuits.

2) Boolean constants are denoted by 0 or 1. Boolean variables are quantities that can take different values at different times. They may represent input, output or intermediate signals.

3) Here we can have n number of variables usually written as a, b, c…. (lower case) and it satisfy all Boolean laws, which will be discussed later.

von Neumann architecture

- The **von Neumann architecture** is a computer architecture based on a 1945 description by John von Neumann. That document describes a design architecture for an electronic digital computer with these components:
  - A processing unit that contains an arithmetic logic unit and processor registers
  - A control unit that contains an instruction register and program counter
  - Memory that stores data and instructions
  - External mass storage
  - Input and output mechanisms



Central Processing Unit
Control Unit
Arithmetic/Logic Unit
Input Device
Output Device
Memory Unit

Break

- Digital systems have two logic levels
    - The lower voltage level is called a logic low or logic 0 (0-1 v), which represent digit 0.
    - The higher voltage level is called a logic high or logic 1 (3.5-5 v), which represent digit 1.

Digital system designing

- Let's try an example of designing a digital device, using this example, we will understand step by step process to design a digital system starting from problem statement.

Q Design a digital system for a car manufacturing company, where we want to design a warning signal for a car, there are three inputs, lights of the car(L), day or night(D), ignition (on/off).?

Solution

1) Understand the problem- here we will Understand the definition of the problem

Design the truth table –

Light	Day	Engine	Warning
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

2) Write the Boolean expression

- $W(L, D, E) = \sum_m (1, 4, 6, 7)$

SOP (min) → 1

- $W(L, D, E) = \prod_M (0, 2, 3, 5)$

POS (max) → 0

Light	Day	Engine	Warning
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

3) Minimize Boolean expression

W = a'b'c + ab'c' + abc' + abc

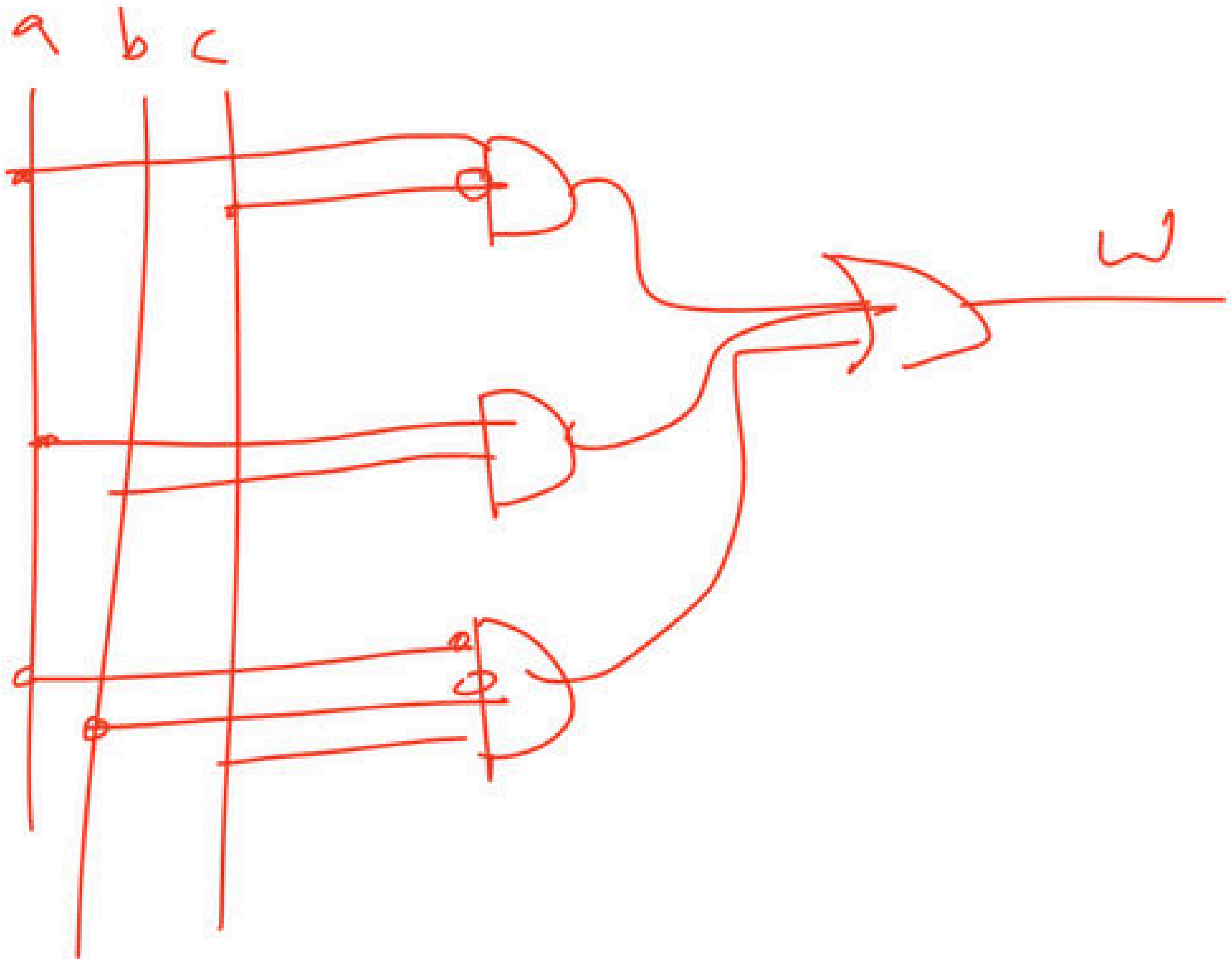
	ab	a'b'	a'b	ab	ab'
c		00	01	11	10
c'	0	0	2	1 6	1 4
c	1	1 1	3	1 7	5

W = $\bar{a}\bar{b}c + ab + a\bar{c}$

Light	Day	Engine	Warning
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

4) Implement the expression using logic gates, draw the implementation using logic gates

$W = ac' + ab + a'b'c$



Light	Day	Engine	Warning
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Break

Boolean Algebra Laws

Idempotent Law

Associative law

Commutative law

Distributive law

Boolean Algebra Laws

Idempotent Law

$a . a = a$

$a + a = a$

Associative law

$a. (b. c) = (a. b). c$

$a + (b + c) = (a + b) + c$

Commutative law

$a. b = b. a$

$a + b = b + a$

Distributive law

$a. (b + c) = a. b + a. c$

$a + (b. c) = (a + b). (a + c)$

De-Morgan law

Identity law

Complementation law

Involution law

De-Morgan law

$(a + b)' = a' . b'$

$(a. b)' = a' + b'$

$$\overline{a+b} = \bar{a} \cdot \bar{b}$$

$$\overline{a.b} = \bar{a} + \bar{b}$$

Identity law

$a + 0 = a$

$a. 0 = 0$

$a + 1 = 1$

$a. 1 = a$

$+ \longrightarrow \cup$

$. \longrightarrow \cap$

0 — ϕ

1 — U

$a \cup \phi = a$

$a \cap \phi = \phi$

$a \cup U = U$

$a \cap U = a$

Complementation law

$0' = 1$

$1' = 0$

$a. a' = 0$

$a + a' = 1$

Involution law

$(a')' = a$

$\bar{\bar{a}} = a$

Break

Q The idempotent law in Boolean algebra says that: **(NET-JUNE-2008)**

(A) ~(~x) = x

(B) x + x = x

(C) x + xy = x

(D) x(x + y) = x

Q A V A = A is called: **(NET-DEC-2004)**

(A) Identity law — 12

(B) De Morgan's law 4

(C) Idempotent law 84

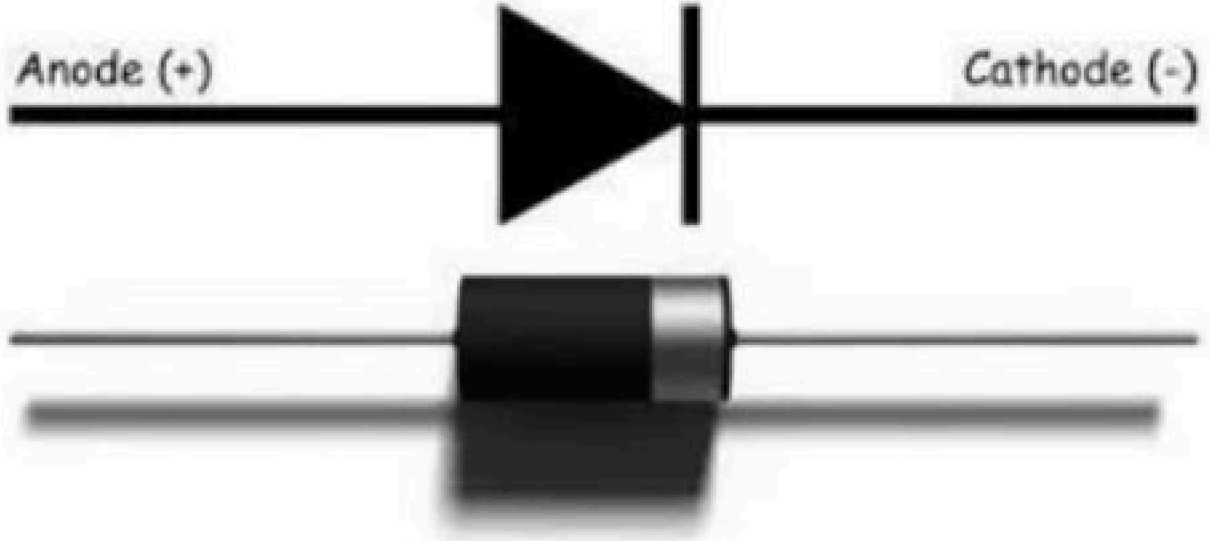
(D) Complement law 0

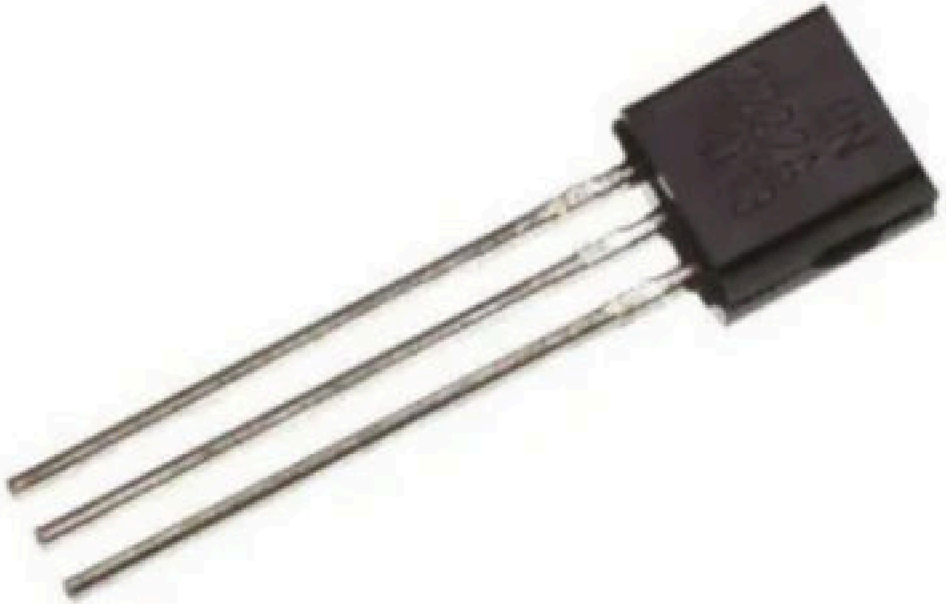
Logic gate

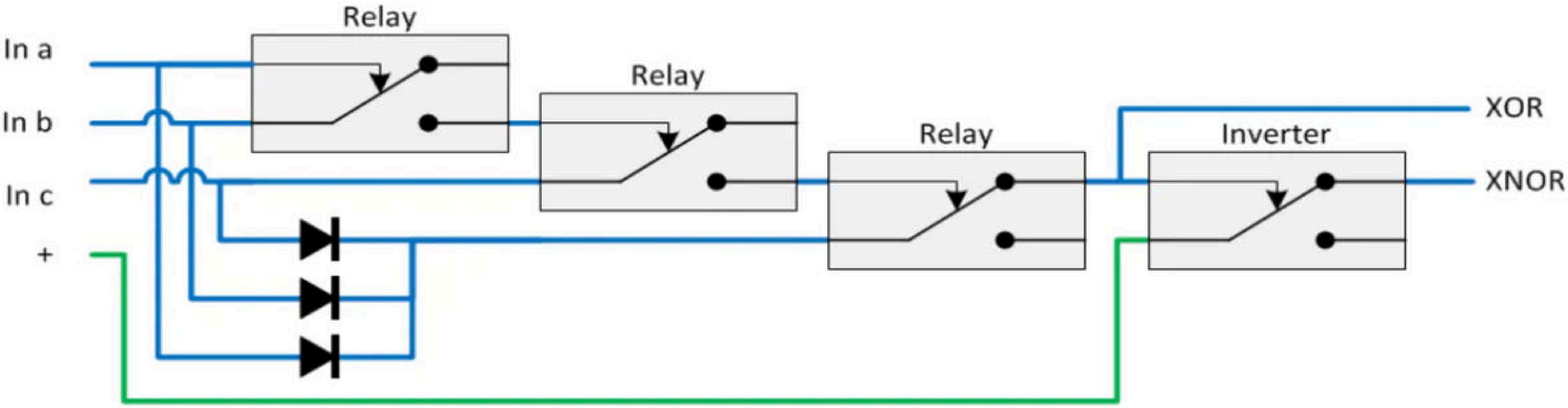
- In electronics, a logic gate is a physical device implementing a Boolean function
- Logic gate performs a logical operation on one or more binary inputs signals and produces a single binary output signal.
- Logic gate is the basic building block from which many kinds of logic circuits can be constructed.

Logic gate

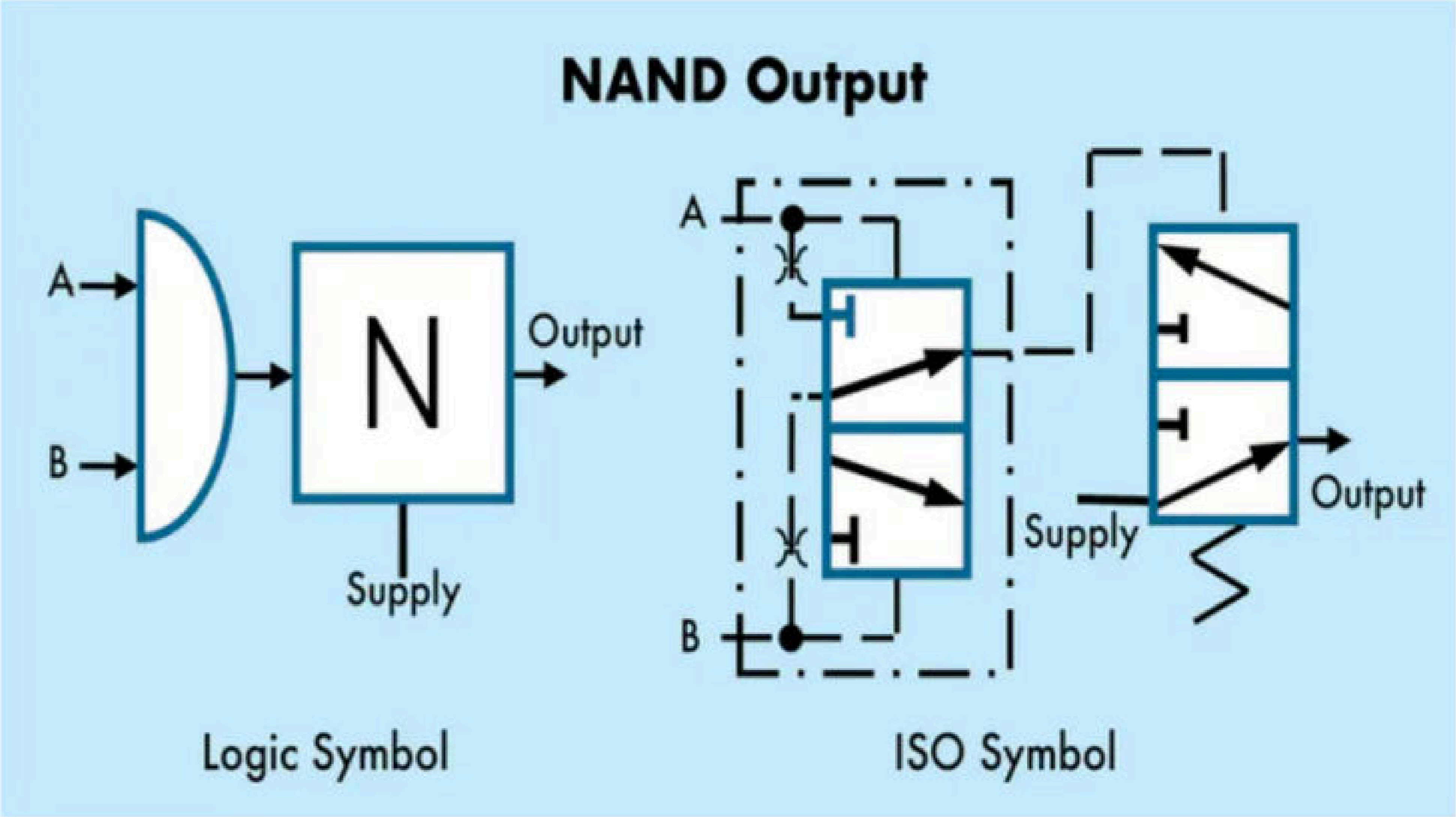
- Logic gates are primarily implemented using diodes or transistors acting as electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements.







electromagnetic relays

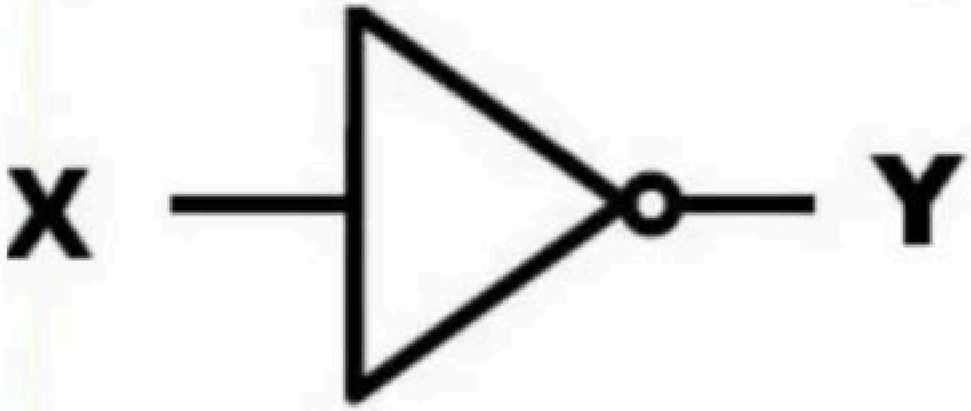


pneumatic logic

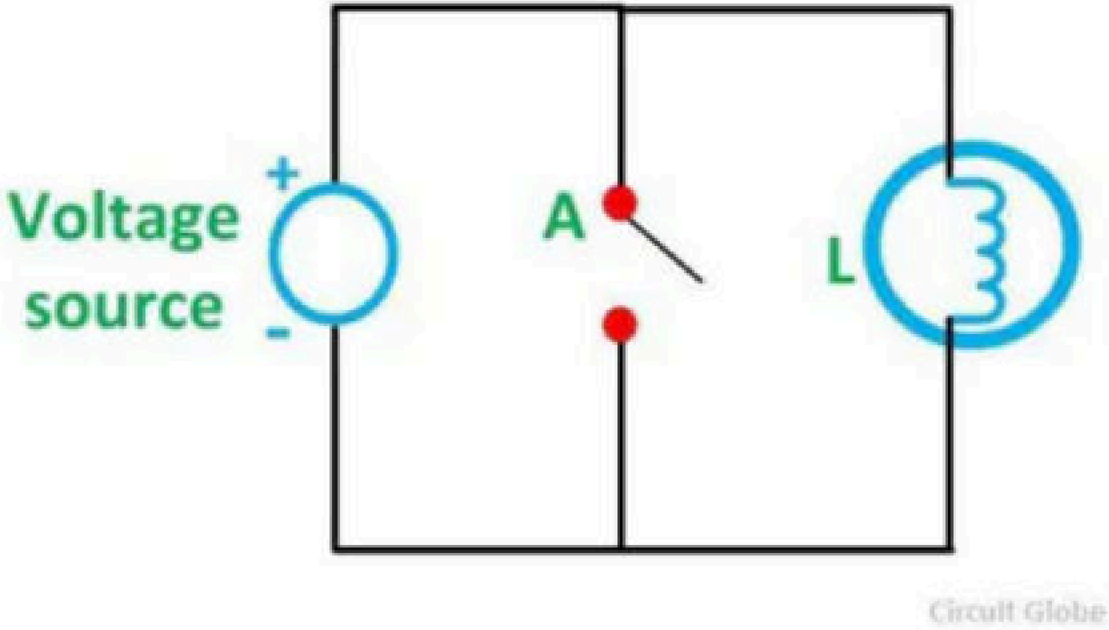
Break

Not Gate(Inverter)

- It represents not logical operator is also known as inverter, it is a unary operator, which simply complement the input.



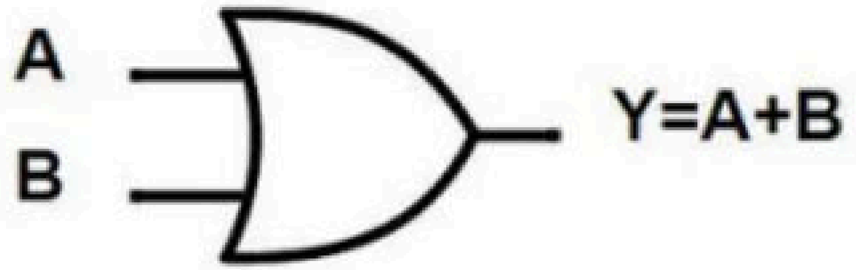
Truth Table	
Input	Output
X	Y = X’
0	
1	



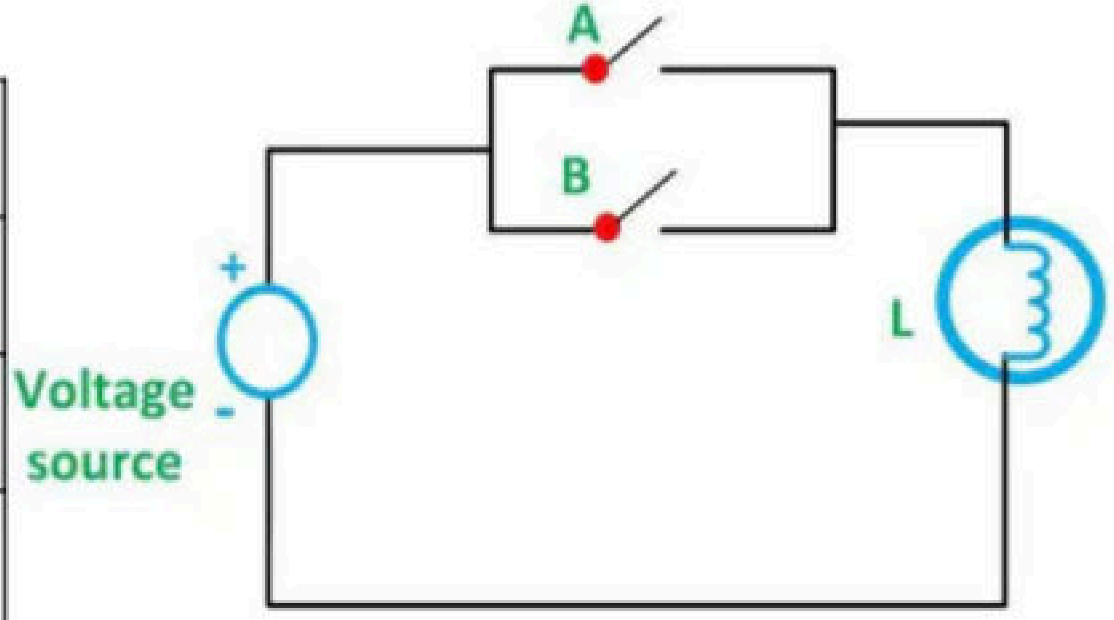
Break

OR Gate

- It is a digital logic gate, that implements logical disjunction. The output will be high if at least one of the input lines is high.



Truth Table		
Input		Output
A	B	Y = A + B
0	0	
0	1	
1	0	
1	1	

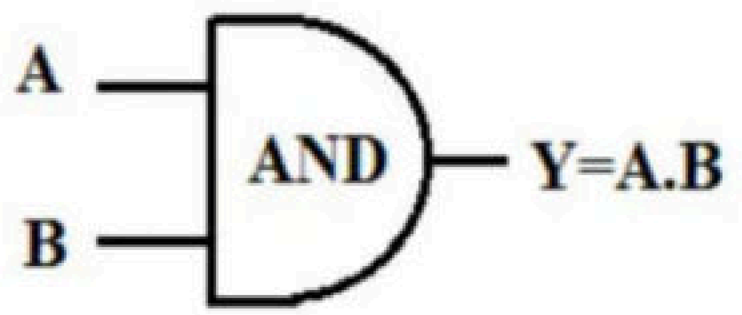


- OR gate satisfy all the 3 rules idempotent, associative, and commutative.
  1. **Idempotent Law:** $a + a = a$
  2. **Associative law:** $a + (b + c) = (a + b) + c$
  3. **Commutative law:** $a + b = b + a$

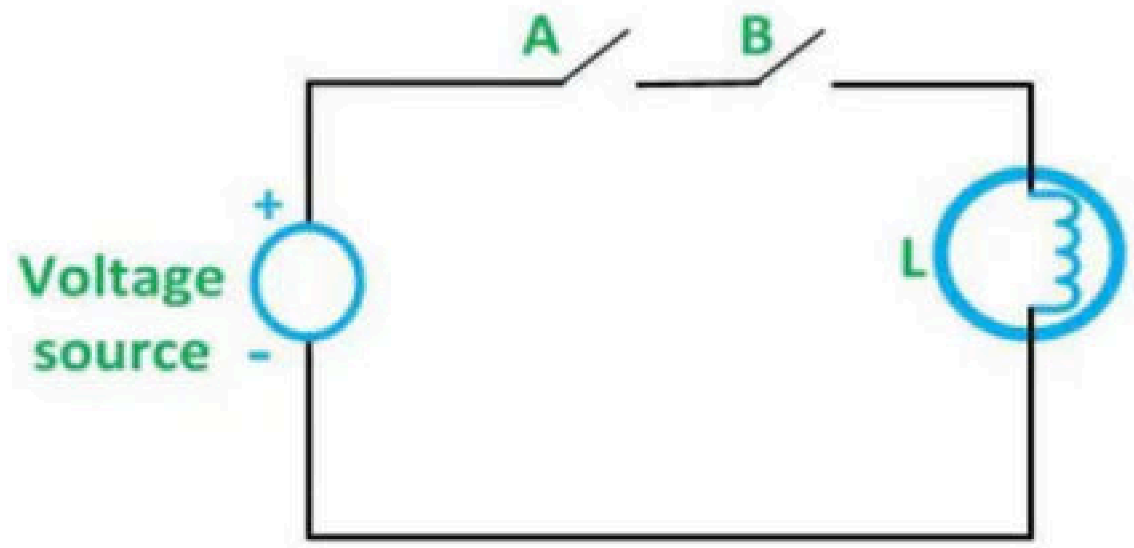
Break

And Gate

- It is a digital logic gate, that implements logical conjunction. Output will be high if and only if all input are high otherwise low.



Truth Table		
Input		Output
A	B	Y = A . B
0	0	
0	1	
1	0	
1	1	



- And gate satisfy all the 3 rules idempotent, associative, and commutative.
  1. **Idempotent Law:** a. a = a

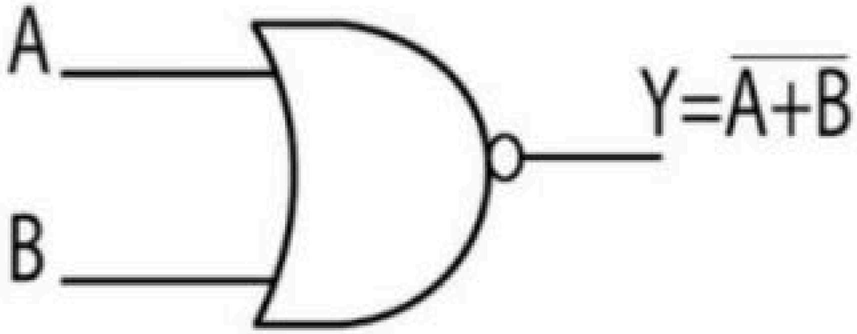
  2. **Associative law:** a.(b. c) = (a. b). C

  3. **Commutative law:** a. b = b. a

Break

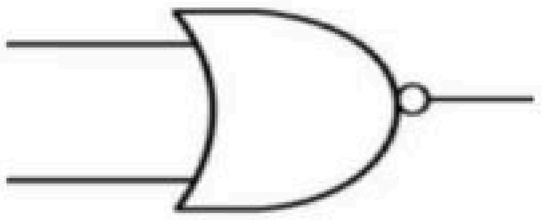
Nor gate

- The output will be high if and only if all inputs are low. Or simply a OR gate followed by an invertor.
- NOR gate is also called universal gate because it can be used to implement any other logic gate. We will cover this property extensively in the next chapter.

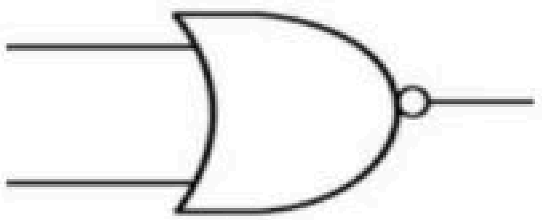


Truth Table		
Input		Output
A	B	Y = (A + B)'
0	0	
0	1	
1	0	
1	1	

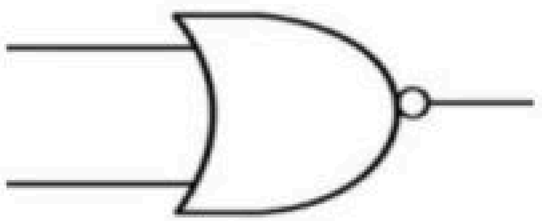
1. NOR with same gives __________
    - $(a + a)' =$



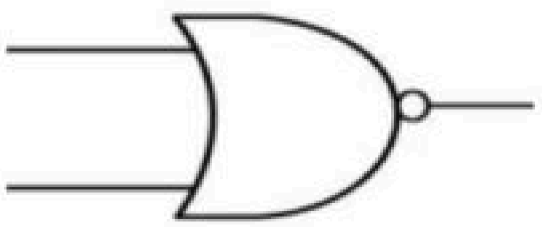
2. NOR with zero gives __________
    - $(a + 0)' =$



3. NOR with complement gives__________
    - $(a + a')' =$



4. NOR with one gives_________
    - $(a + 1)' =$



- Idempotent and associative law
  - $(a + a)' \boxed{}\ a$
  - $((a + b)' + c)' \boxed{}\ (a + (b + c)')'$

- Commutative law.
  - $(a + b)' \boxed{}\ (b + a)'$

Conclusion

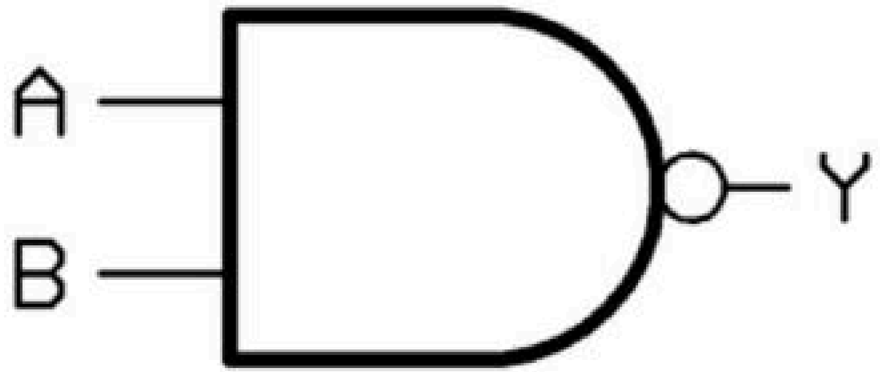
- NOR with same gives complement → $(a + a)' = a'$
- NOR with zero gives complement → $(a + 0)' = a'$
- NOR with complement gives zero → $(a + a')' = 0$
- NOR with one gives zero → $(a + 1)' = 0$

- NOR does not satisfy idempotent and associative law
  - $(a + a)' \mathrel{!}= a$
  - $((a + b)' + c)' \mathrel{!}= (a + (b + c)')'$
- It satisfies commutative law.
  - $(a + b)' = (b + a)'$

Break

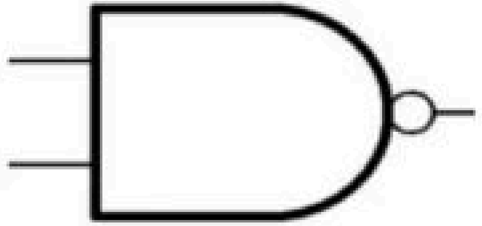
NAND Gate

- The output will be low if and only if all inputs are high. Or simply an and gate followed by an invertor
- NAND gate is also called universal gate because it can be used to implement any other logic gate. We will cover this property extensively in the next chapter.

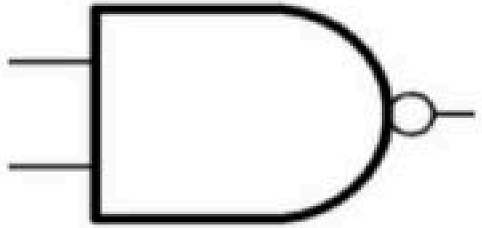


Truth Table		
Input		Output
A	B	Y = (A . B)'
0	0	
0	1	
1	0	
1	1	

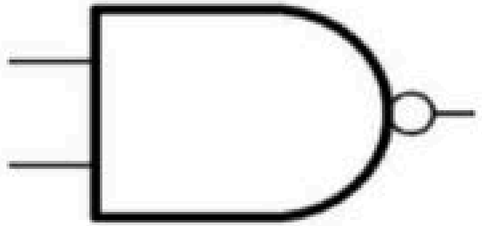
1. NAND with ZERO give_________
    - (a. 0)’ =



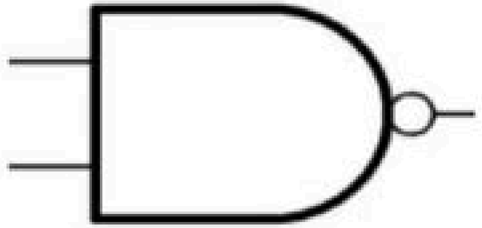
2. NAND with COMPLEMENT give_________
    - (a. a’)’ =



3. NAND with SAME give_________
    - (a. a)’ =



4. NAND with ONE give_________
    - (a. 1)’ =



- NAND with idempotent and associative law
  - (a. a)’ $\boxed{\qquad}$ a
  - ((a. b)’. c)’ $\boxed{\qquad}$ (a. (b. c)’)’

- NAND with commutative law
  - (a. b)’ $\boxed{\qquad}$ (b. a)’

Conclusion

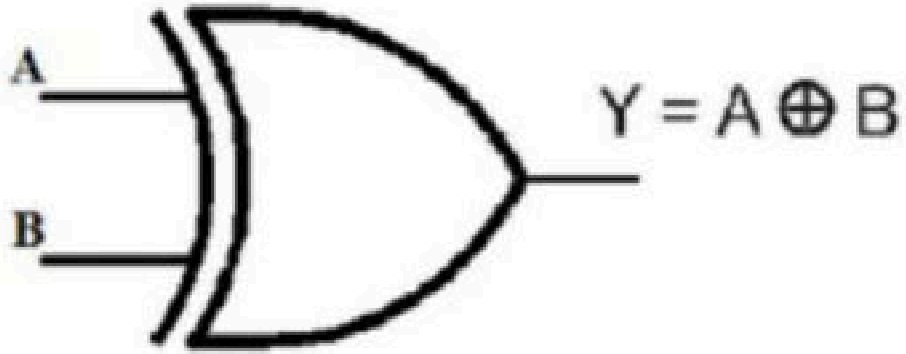
- NAND with ZERO give ONE → $(a. 0)' = 1$
- NAND with COMPLEMENT give ONE → $(a. a')' = 1$
- NAND with SAME give COMPLEMENT → $(a. 1)' = a'$
- NAND with ONE give COMPLEMENT → $(a. a)' = a'$

- NAND does not satisfy idempotent and associative law
  - $(a. a)' \; != a$
  - $((a. b)'. c)' \; != (a. (b. c)')'$
- It satisfies commutative law
  - $(a. b)' = (b. a)'$

Break

EX-OR

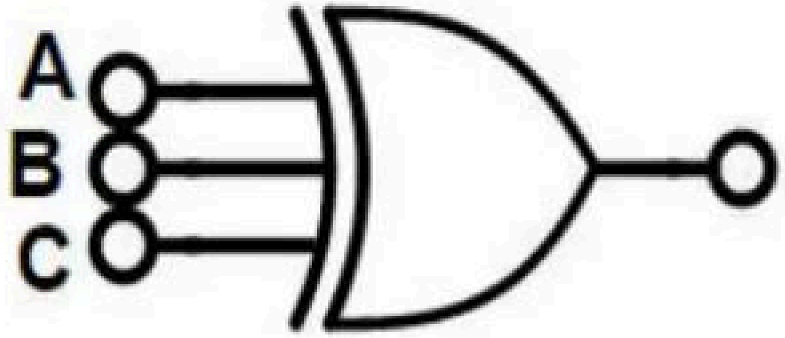
- For two inputs, output will be high if and only if both the input values are different.
- $a \oplus b = a'.\,b + a.\,b'$



Truth Table		
Input		Output
A	B	$Y = A \oplus B$
0	0	
0	1	
1	0	
1	1	

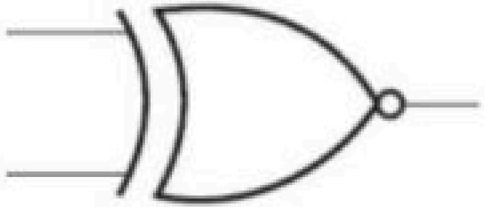
EX-OR

- The XOR gate is a digital logic gate that gives High as output when the number of inputs High are odd.

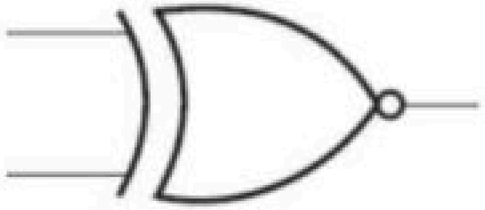


A	B	C	$a \oplus b \oplus c$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

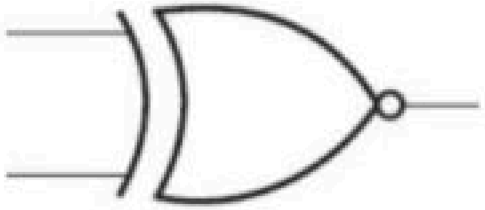
1. EX-OR with ZERO give__________
   - $(a \oplus 0) =$



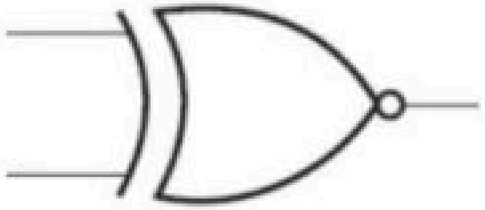
2. EX-OR with ONE give__________
   - $(a \oplus 1) =$



3. EX-OR with SAME give__________
   - $(a \oplus a) =$



4. EX-OR with COMPLEMENT give___________
   - $(a \oplus a') =$



- EX-OR with idempotent law
  - $(a \oplus a) \; \boxed{\qquad} \; a$

- EX-OR with associative and commutative law.
  - $((a \oplus b) \oplus c) \; \boxed{\qquad} \; (a \oplus (b \oplus c))$
  - $(a \oplus b) \; \boxed{\qquad} \; (b \oplus a)$

Conclusion

- EX-OR with ZERO give SAME → $(a \oplus 0) = a$
- EX-OR with ONE give COMPLEMENT → $(a \oplus 1) = a'$
- EX-OR with SAME give ZERO → $(a \oplus a) = 0$
- EX-OR with COMPLEMENT give ONE → $(a \oplus a') = 1$

- EX-OR does not satisfy idempotent
  - $(a \oplus a) != a$
- It satisfies associative and commutative law.
  - $((a \oplus b) \oplus c) = (a \oplus (b \oplus c))$
  - $(a \oplus b) = (b \oplus a)$

Break

Q Which of the following logic expressions is incorrect? **(NET-JULY-2016)**

a) $1 \oplus 0 = 1$

b) $1 \oplus 1 \oplus 1 = 1$

c) $1 \oplus 1 \oplus 0 = 1$

d) $1 \oplus 1 = 0$

Q Consider the Boolean operator with the following properties: **(GATE-2016) (1 Marks)**

$x \# 0 = x$,

$x \# 1 = x'$,

$x \# x = 0$

and $x \# x' = 1$

Then $x \# y$ is equivalent to

a) $xy' + x'y$ **b)** $xy' + x'y'$ **c)** $x'y + xy$ **d)** $xy + x'y'$

Q The binary operator # is defined by the following truth table. **(GATE-2015) (1 Marks)**

Which of the following is true about the binary operator # ?

a) Both commutative and associative

b) Commutative but not associative

c) Not commutative but associative

d) Neither commutative nor associative

p	q	P # q
0	0	0
0	1	1
1	0	1
1	1	0

Q Let $\oplus$ denote the Exclusive OR (XOR) operation. Let '1' and '0' denote the binary constants. Consider the following Boolean expression for F over two variables P and Q.

$F(P, Q) = ((1 \oplus P) \oplus (P \oplus Q)) \oplus ((P \oplus Q) \oplus (Q \oplus 0))$

The equivalent expression for F is

(GATE-2014) (2 Marks)

(A) $P + Q$

(B) $(P + Q)'$

(C) $P \oplus Q$

(D) $(P \oplus Q)'$

Q If $A \oplus B = C$, then which of the following are invalid: **(NET-JUNE-2007)**

(A) $A \oplus C = B$ **(B)** $B \oplus C = A$ **(C)** $A \oplus B \oplus C = 1$ **(D)** $A \oplus B \oplus C = 0$

A	**B**	**$A \oplus B = C$**

Q Consider the following sequence of instructions: **(NET-DEC-2005)**

$a = a \oplus b$, $b = a \oplus b$, $a = b \oplus a$. This sequence

(A) retains the value of the a and b

(B) complements the value of a and b

(C) swap a and b

(D) negates values of a and b

Q An example of a connective which is not associative is: **(NET-DEC-2004)**

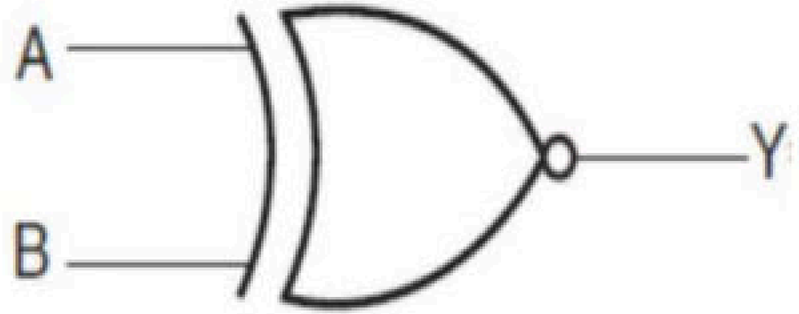
(A) AND **(B)** OR **(C)** EX-OR **(D)** NAND

Q $A \oplus P \oplus A \oplus P \oplus B \oplus P \oplus B \oplus P \oplus B$?

Break

EX-NOR

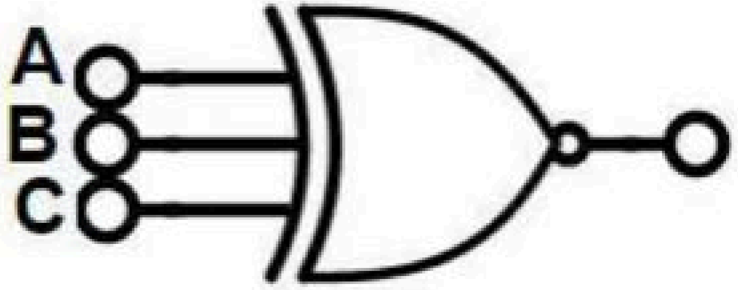
- For two input, output will be high if and only if both the input values are same
- $a \odot b = a'.\, b' + a.\, b$



Truth Table		
Input		Output
A	B	$Y = A \odot B$
0	0	
0	1	
1	0	
1	1	

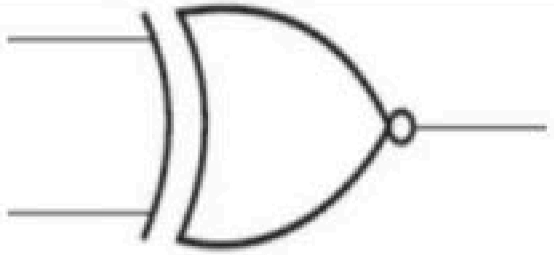
EX-NOR

- The EX-NOR gate is a digital logic gate that gives output High when the number inputs low are even.

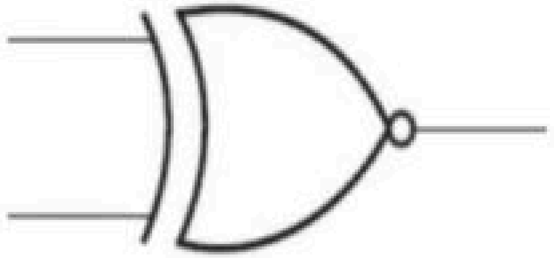


A	B	C	a ⊙ b ⊙ c
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

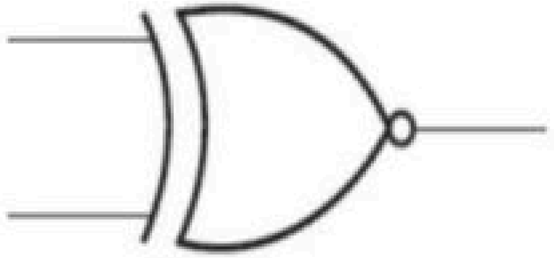
1. EX-NOR with ZERO give__________
   - $(a \odot 0) =$



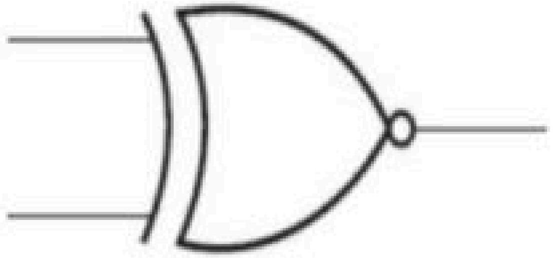
2. EX-NOR with ONE give__________
   - $(a \odot 1) =$



3. EX-NOR with SAME give__________
   - $(a \odot a) =$



4. EX-NOR with COMPLEMENT give__________
   - $(a \odot a') =$



- EX-NOR with idempotent law
  - $(a \odot a) \; \boxed{} \; a$

- EX-NOR with associative and commutative law
  - $((a \odot b) \odot c) \; \boxed{} \; (a \odot (b \odot c))$
  - $(a \odot b) \; \boxed{} \; (b \odot a)$

Conclusion

- EX-NOR with ZERO give COMPLEMENT →$(a \odot 0) = a'$
- EX-NOR with ONE give SAME →$(a \odot 1) = a$
- EX-NOR with SAME give ONE →$(a \odot a) = 1$
- EX-NOR with COMPLEMENT give ZERO →$(a \odot a') = 0$

- EX-NOR does not satisfy idempotent
  - $(a \odot a) \mathrel{!}= a$
- it satisfies associative and commutative law.
  - $((a \odot b) \odot c) = (a \odot (b \odot c))$
  - $(a \odot b) = (b \odot a)$

Break

Q A Boolean operator s is defined as follows:

1 s 1 = 1, 1 s 0 = 0, 0 s 1 = 0 and 0 s 0 = 1

What will be the truth value of the expression (x s y) s z = x s (y s z)? **(NET-DEC-2013)**

(A) Always false

(B) Always true

(C) Sometimes true

(D) True when x, y, z are all true

Q Define the connective * for the Boolean variables X and Y as: X * Y = XY + X’ Y’. Let Z = X * Y.

(GATE-2007) (2 Marks)

Consider the following expressions P, Q and R.

P: X = Y⋆Z

Q: Y = X⋆Z

R: X⋆Y⋆Z=1

X	**Y**	**X ⊙ Y = Z**

Which of the following is TRUE?

(A) Only P and Q are valid

(B) Only Q and R are valid.

(C) Only P and R are valid.

(D) All P, Q, R are valid.

Break

A	B	C	$a \oplus b \oplus c$	$a \odot b \odot c$
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Conclusion

A	B	C	$a \oplus b \oplus c$	$a \odot b \odot c$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

- Ex-or and ex-nor gate behaves as a complement of each other if the number of input variable is even.
- Ex-or and ex-nor gate behave same if no of input variables are odd.

Conclusion

1. Ex-or and ex-nor gate behaves as a complement of each other if the number of input variable is even.

2. Ex-or and ex-nor gate behave same if no of input variables are odd.

<u>Relation of EX-OR and EX-NOR</u>

$$a \oplus b = a' \square b = a \square b' = (a \square b)' = (a' \square b')' = a' \square b' = (a' \square b)' = (a \square b')'$$

$$a \odot b = a' \square b = a \square b' = (a \square b)' = (a' \square b')' = a' \square b' = (a' \square b)' = (a \square b')'$$

Conclusion

$$a \oplus b = a' \odot b = a \odot b' = (a \odot b)' = (a' \odot b')' = a' \oplus b' = (a' \oplus b)' = (a \oplus b')'$$

$$a \odot b = a' \oplus b = a \oplus b' = (a \oplus b)' = (a' \oplus b')' = a' \odot b' = (a' \odot b)' = (a \odot b')'$$

Q Which one of the following is NOT a valid identity? **(GATE-2019) (2 Marks)**

(A) $(x \oplus y) \oplus z = x \oplus (y \oplus z)$

(B) $(x + y) \oplus z = x \oplus (y + z)$

(C) $x \oplus y = x + y$, if $xy = 0$

(D) $x \oplus y = (xy + x'y')'$

Q Let $\oplus$ and $\odot$ denote the Exclusive OR and Exclusive NOR operations, respectively.

Which one of the following is NOT CORRECT? **(GATE-2018) (2 Marks)**

(A) $(P \oplus Q)' = P \odot Q$

(B) $\overline{P} \oplus Q = P \odot Q$

(C) $\overline{P} \oplus \overline{Q} = P \oplus Q$

(D) $(P \oplus \overline{P}) \oplus Q = (P \odot \overline{P}) \odot \overline{Q}$

Q Let, $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$ where X_1, X_2, X_3, X_4 are Boolean variables, and $\oplus$ is the XOR operator. Which one of the following must always be TRUE? **(GATE-2016) (2 Marks)**

a) $X_1X_2X_3X_4 = 0$

b) $X_1X_3+X_2 = 0$

c) $X'_1 \oplus X'_3 = X'_2 \oplus X'_4$

d) $X_1 + X_2 + X_3 + X_4 = 0$

	X_1	X_2	X_3	X_4	$X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$	$X_1X_2X_3X_4 = 0$	$X_1X_3+X_2 = 0$	$X'_1 \oplus X'_3 = X'_2 \oplus X'_4$	$X_1 + X_2 + X_3 + X_4 = 0$
0	0	0	0	0					
1	0	0	0	1					
2	0	0	1	0					
3	0	0	1	1					
4	0	1	0	0					
5	0	1	0	1					
6	0	1	1	0					
7	0	1	1	1					
8	1	0	0	0					
9	1	0	0	1					
10	1	0	1	0					
11	1	0	1	1					
12	1	1	0	0					
13	1	1	0	1					
14	1	1	1	0					
15	1	1	1	1					

Q Which one of the following expressions does NOT represent exclusive NOR of x and y? **(GATE-2013) (1 Marks)**

(A) $xy + x' y'$

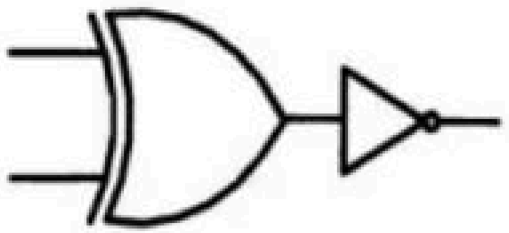
(B) $x \wedge y'$ where ^ is XOR

(C) $x' \wedge y$ where ^ is XOR

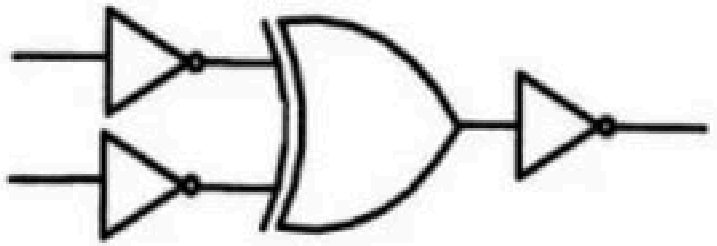
(D) $x' \wedge y'$ where ^ is XOR

Q Which one of the following circuits is NOT equivalent to a 2-input XNOR (exclusive NOR) gate? **(GATE-2011) (1 Marks)**

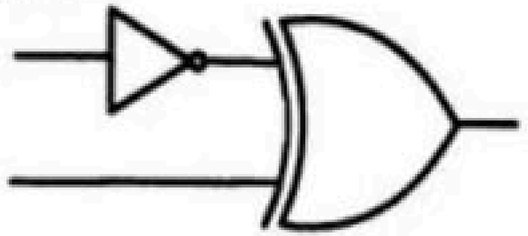
(A)



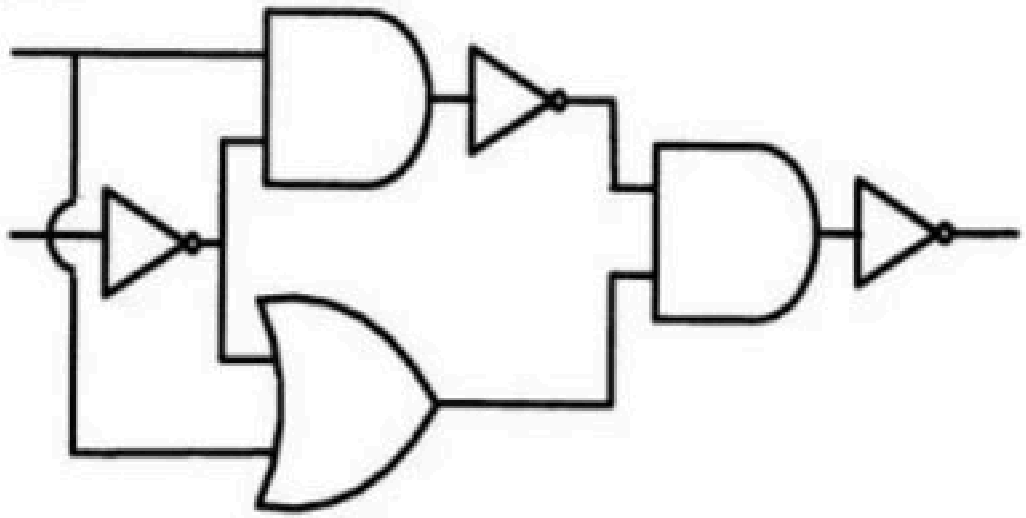
(B)



(C)



(D)



Q The simultaneous equations on the Boolean variables x, y, z and w,

$x + y + z = 1$

$xy = 0$

$xz + w = 1$

$xy + z'w' = 0$

have the following solution for x, y, z and w, respectively. **(GATE-2000) (2 Marks)**

(A) 0 1 0 0

(B) 1 1 0 1

(C) 1 0 1 1

(D) 1 0 0 0