UNIT 4/5

COMBINATIONAL LOGIC

2ND SESSION

Implementation of Boolean functions using Decoder and Multiplexers

1. Using Decoder

- Any combinational circuit with ‘n’ inputs and ‘m’ outputs can be implemented with ‘n to 2^n’ decoder with ‘m’ OR gate.
- **Ex:** *Implement the following Boolean expression using decoder*:

F1=∑m(0,4,6), F2=∑m(0,5) and **F3=∑m(1,2,3,7)** where F is the function of A,B,C.

No. of Inputs=3

Decoder size=3:2^3=3:8

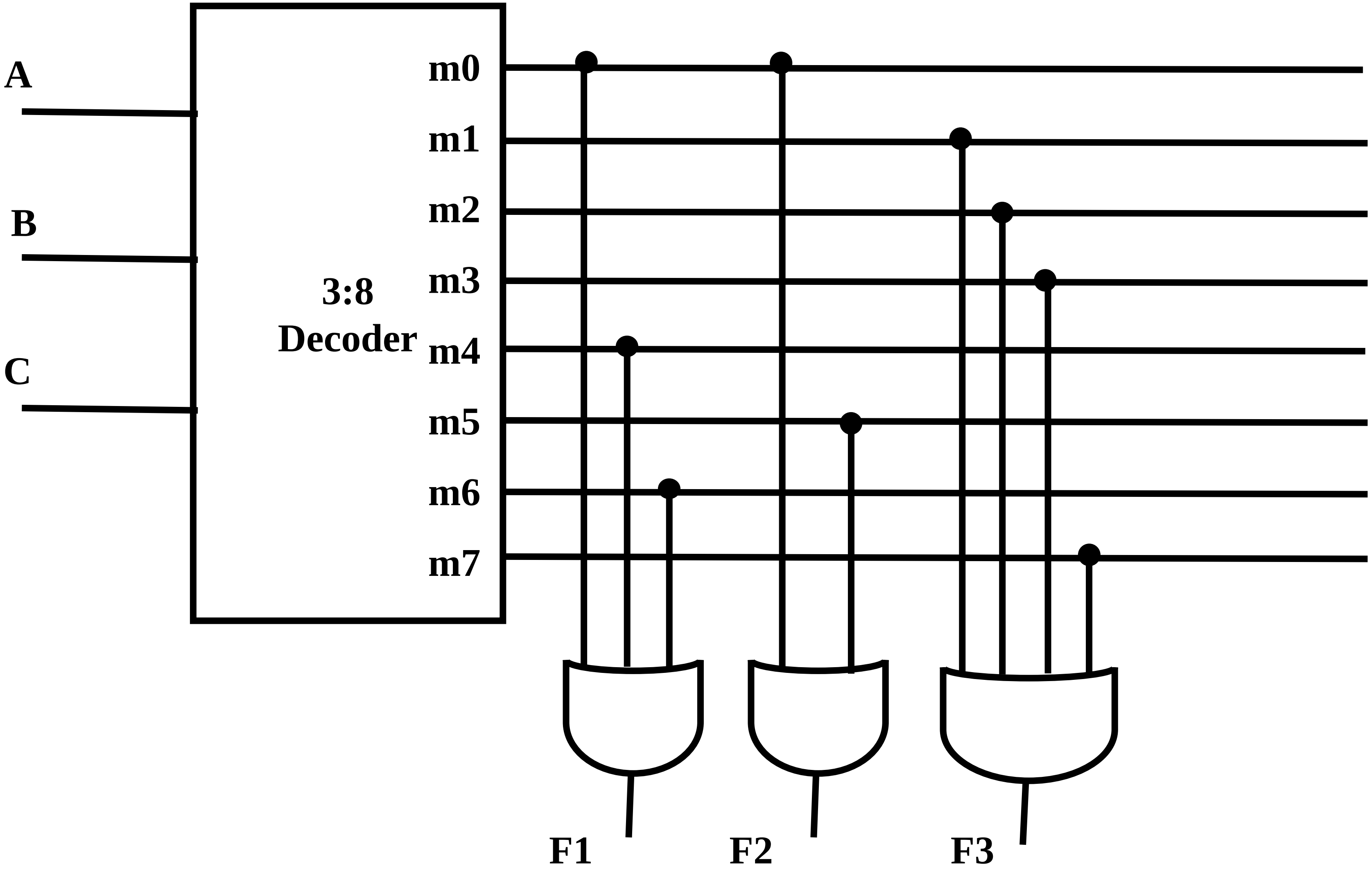
No of outputs=3 (F1,F2,F3)

No, of OR gate=3

F1=∑m(0,4,6) = m0+m4+m6 = $\overline{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}BC$

F2=∑m(0,5) = m0+m5 = $\overline{A}\bar{B}\bar{C} + A\bar{B}C$

F3=∑m(1,2,3,7) = m1+m2+m3+m7 = $\overline{A}\bar{B}C + \bar{A}B\bar{C} + \overline{A}BC + ABC$


A
B
C
3:8
Decoder
m0
m1
m2
m3
m4
m5
m6
m7
F1
F2
F3

Ex: *Implement of the Full Adder using decoder*:

No. of Inputs=3

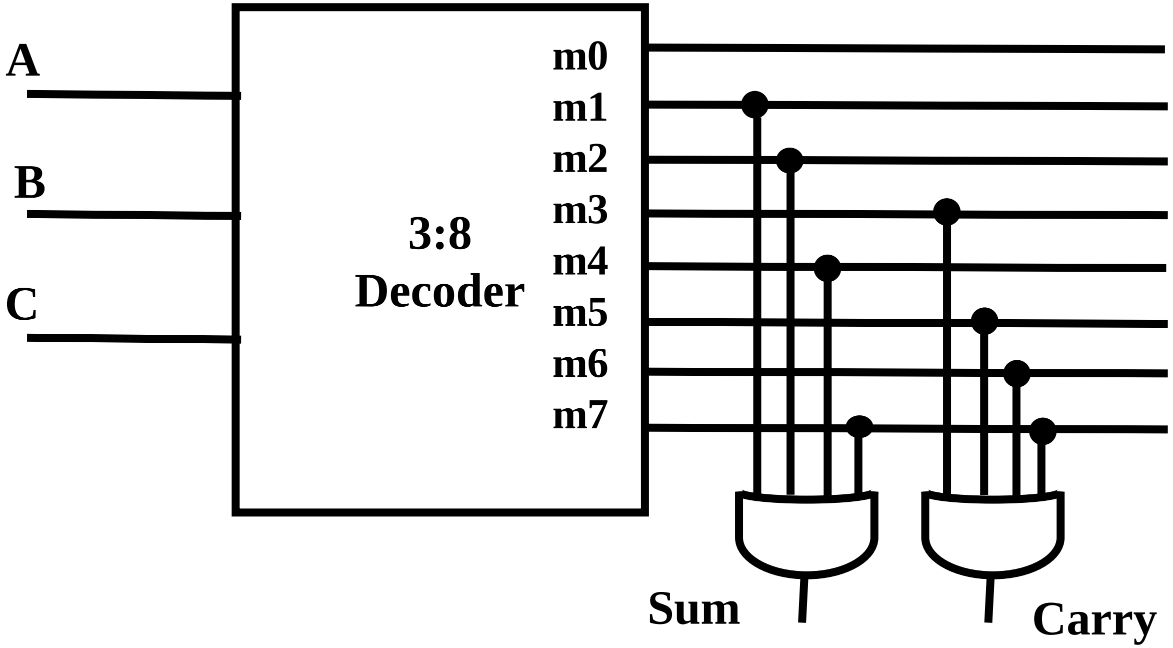
Decoder size=3:2^3=3:8

No of outputs=2 (Sum and Carry)

No, of OR gate=2

Sum (S) =∑m(1,2,4,7) = m1+m2+m4+m7 = $\bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$

Carry (C) =∑m(3,5,6,7) = m3+m5+m6+m7 = $\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

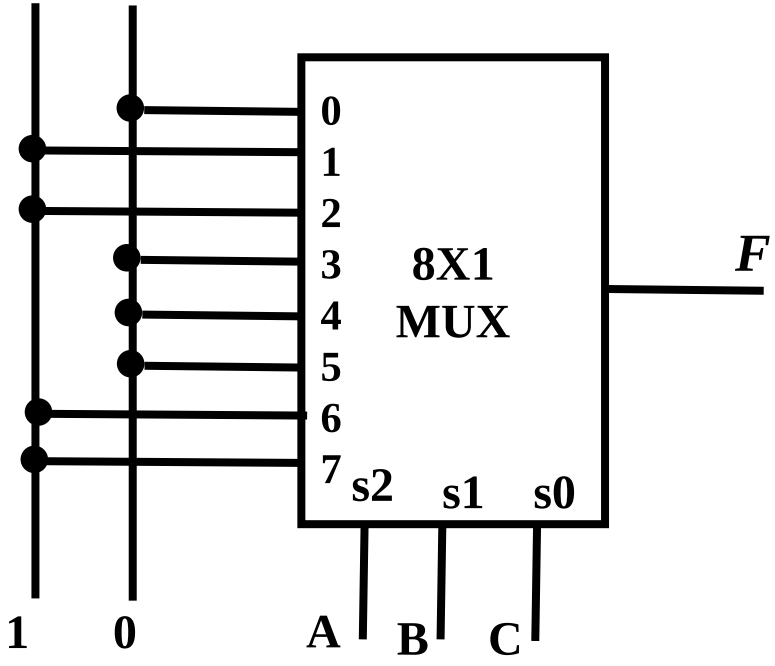


2. Using Multiplexer

- The inputs of the Boolean function are the selection lines of the MUX.
- **Ex:** *Implement the function $F(A,B,C)=\sum m(1,2,6,7)$*

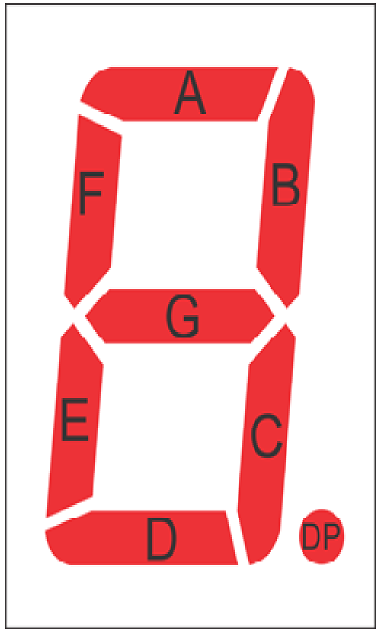
  no. of variables =3=no. of selection lines

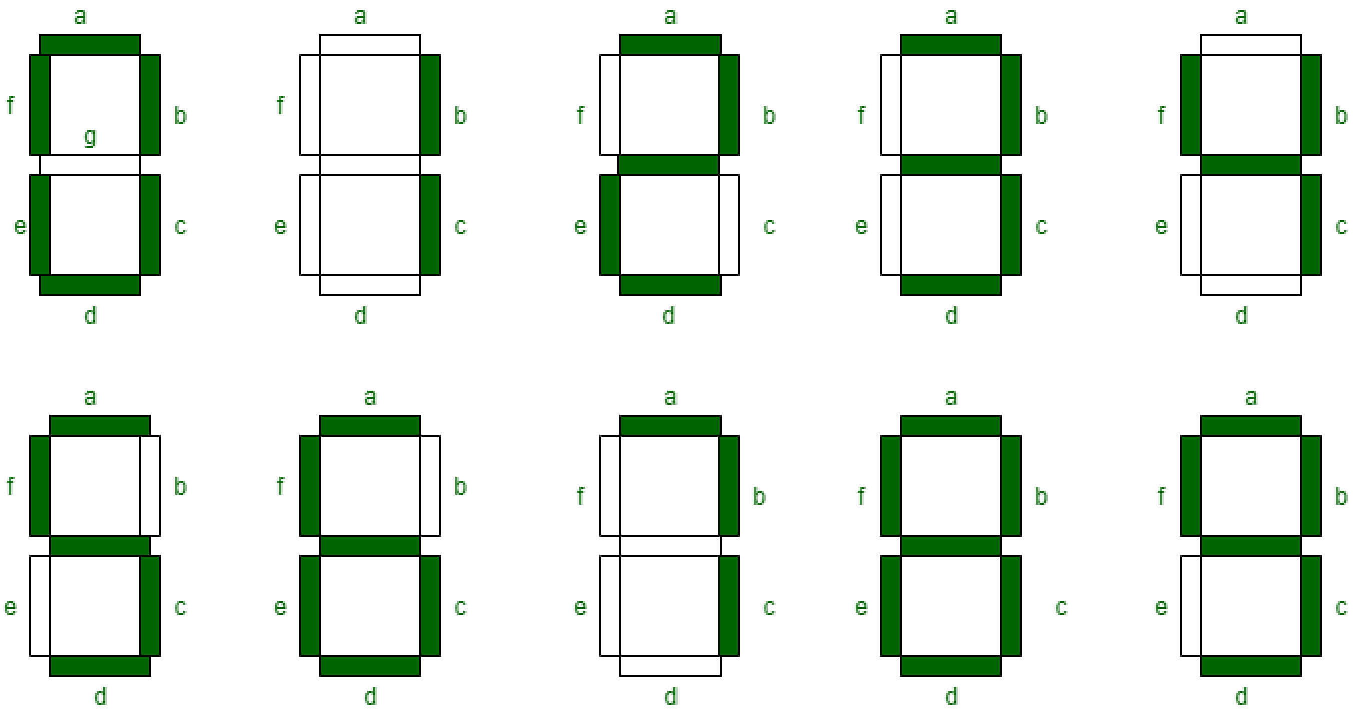
  size of MUX=$2^3 \times 1 = 8 \times 1$

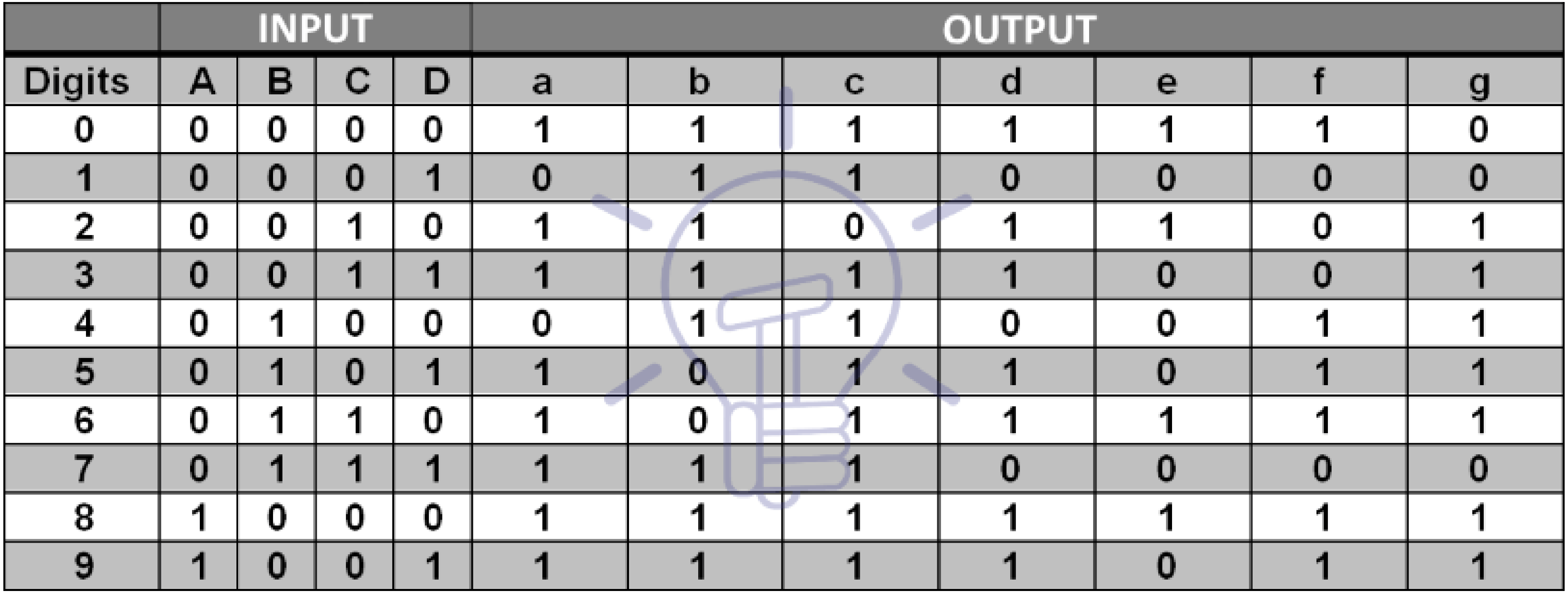


Seven Segment Decoder

- An electronic device which consists of seven light emitting diodes (LEDs) arranged in a some definite pattern (common cathode or common anode type),
- Display hexadecimal numerals(input is BCD i.e., 0-9).





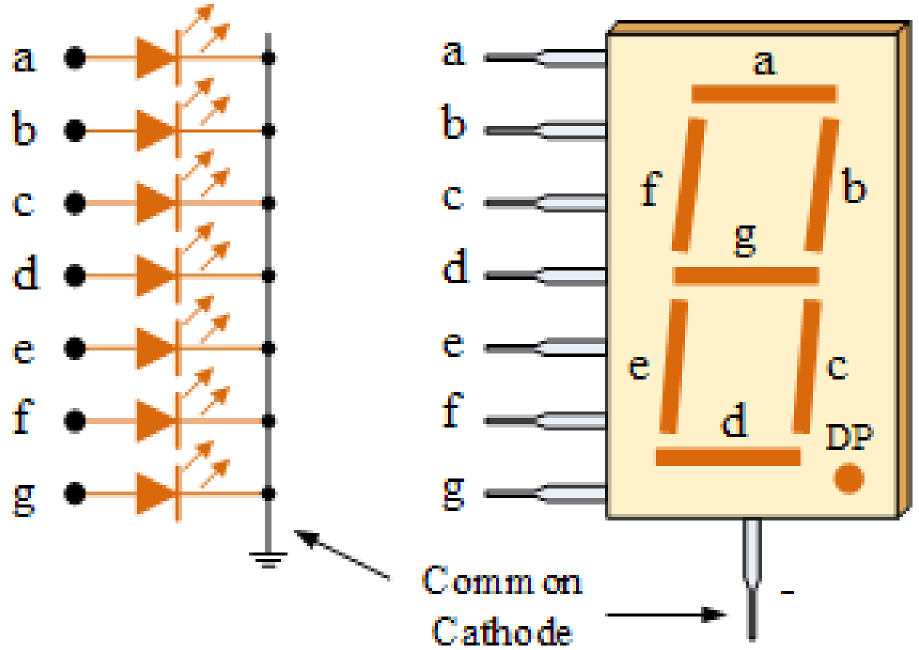


	INPUT				OUTPUT						
Digits	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

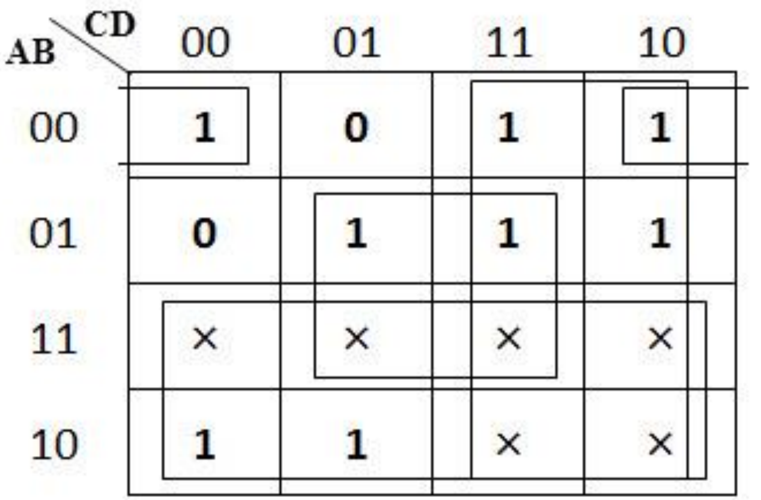
Truth Table for common cathode

From the above truth table, the Boolean expressions of each output functions can be written as;

a = F1 (A, B, C, D) = ∑m (0, 2, 3, 5, 7, 8, 9)
b = F2 (A, B, C, D) = ∑m (0, 1, 2, 3, 4, 7, 8, 9)
c = F3 (A, B, C, D) = ∑m (0, 1, 3, 4, 5, 6, 7, 8, 9)
d = F4 (A, B, C, D) = ∑m (0, 2, 3, 5, 6, 8)
e = F5 (A, B, C, D) = ∑m (0, 2, 6, 8)
f = F6 (A, B, C, D) = ∑m (0, 4, 5, 6, 8, 9)
g = F7 (A, B, C, D) = ∑m (2, 3, 4, 5, 6, 8, 9)

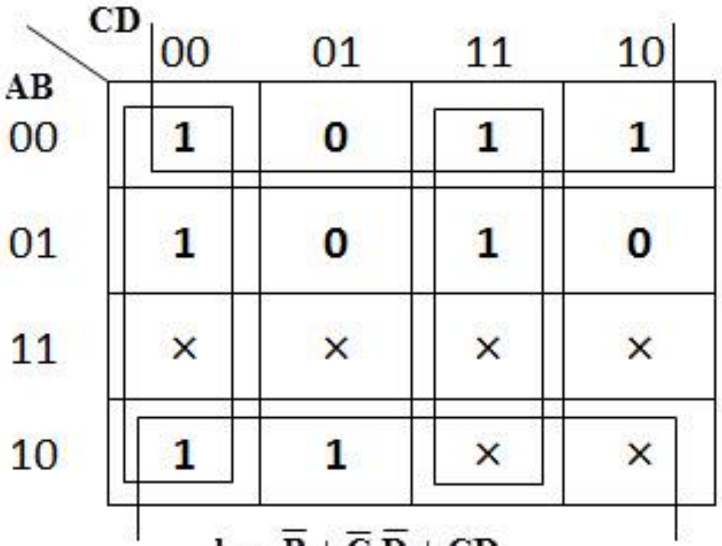


K-Map Simplification



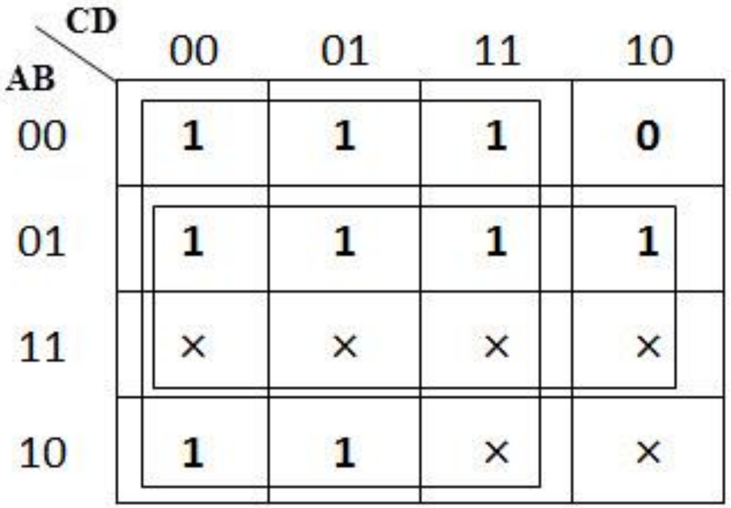
AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	×	×	×	×
10	1	1	×	×

$a = A + C + BD + \overline{B}\,\overline{D}$



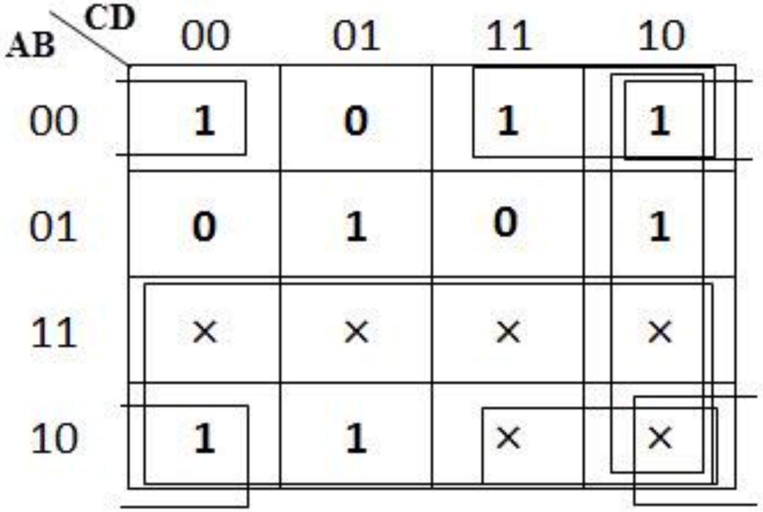
AB \ CD	00	01	11	10
00	1	0	1	1
01	1	0	1	0
11	×	×	×	×
10	1	1	×	×

$b = \overline{B} + \overline{C}\,\overline{D} + CD$



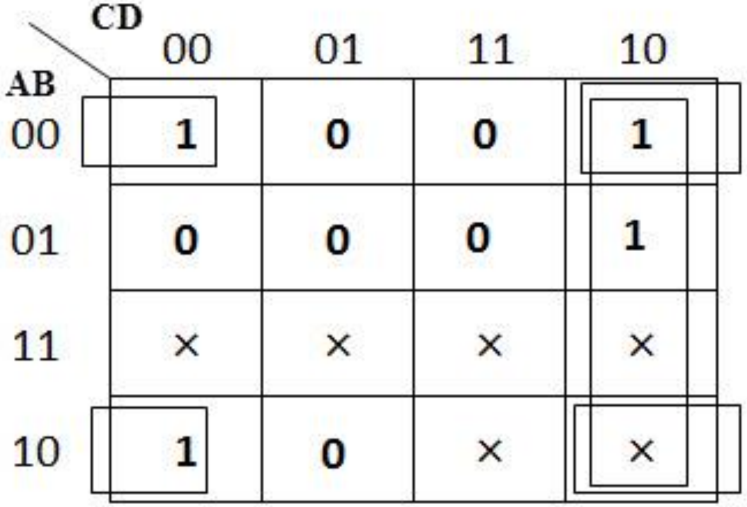
AB \ CD	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	×	×	×	×
10	1	1	×	×

$c = B + \overline{C} + D$



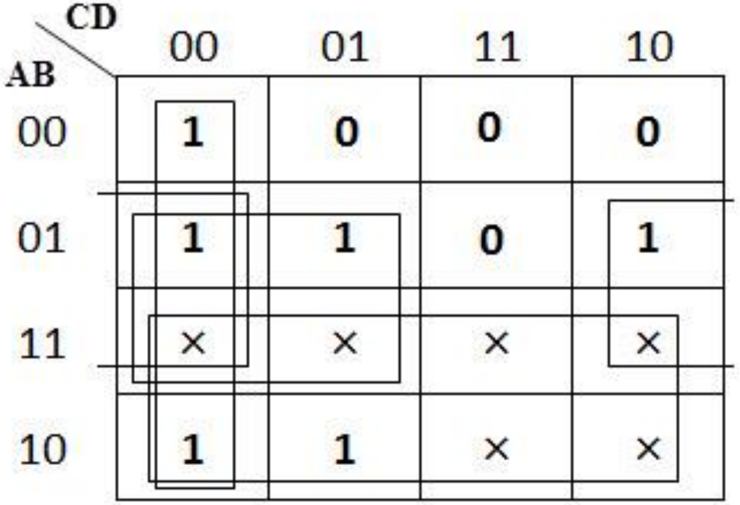
AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	×	×	×	×
10	1	1	×	×

$d = \overline{B}\,\overline{D} + C\,\overline{D} + B\,\overline{C}\,D + \overline{B}\,C + A$



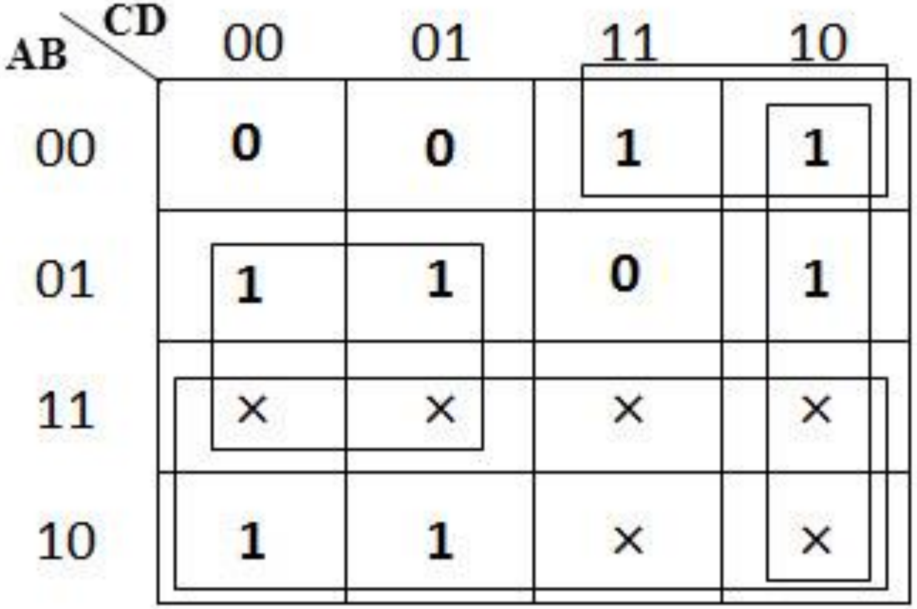
AB \ CD	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	×	×	×	×
10	1	0	×	×

$e = \overline{B}\,\overline{D} + C\,\overline{D}$



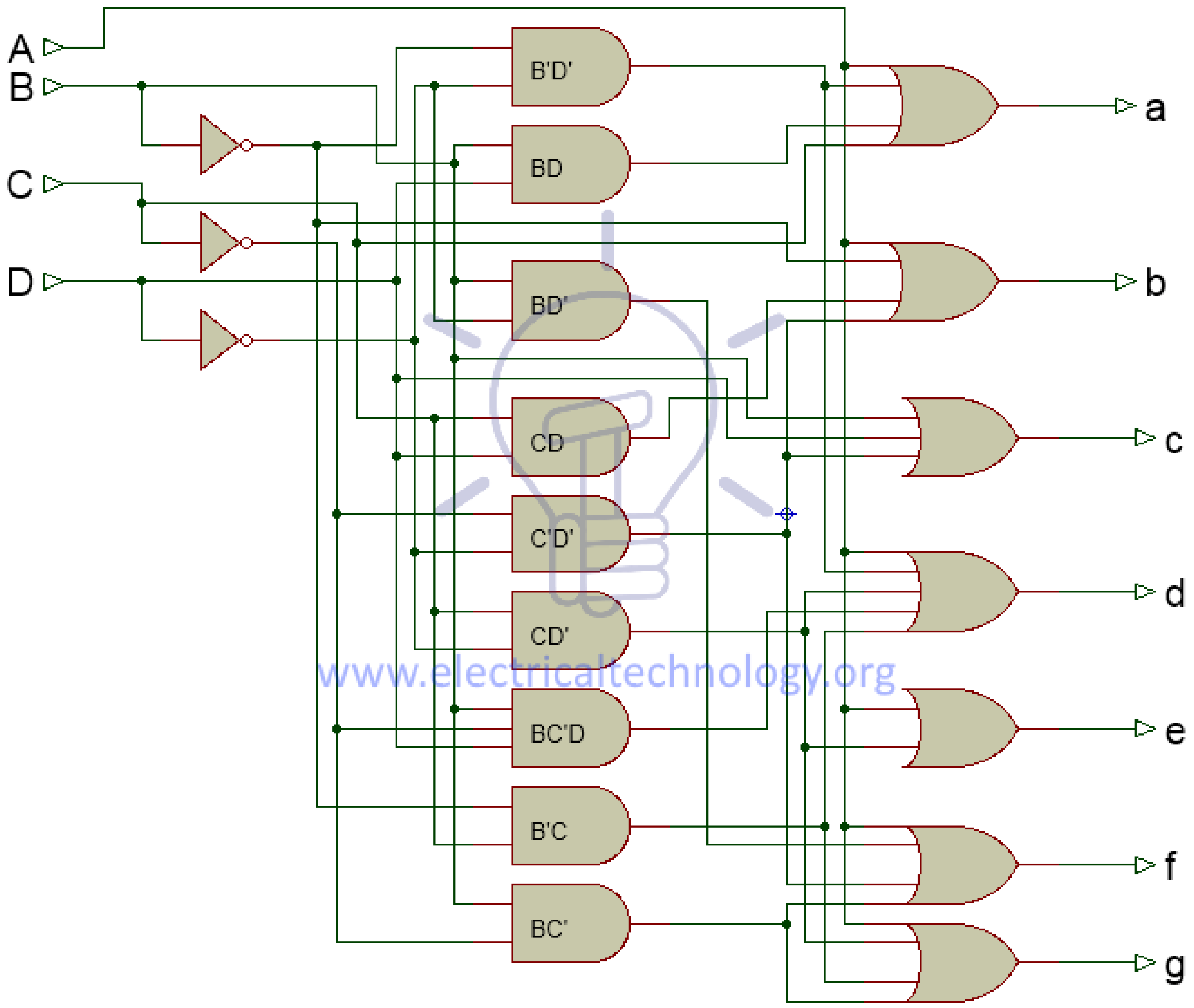
AB \ CD	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	×	×	×	×
10	1	1	×	×

$f = A + \overline{C}\,\overline{D} + B\,\overline{C} + B\,\overline{D}$



AB \ CD	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	×	×	×	×
10	1	1	×	×

$$g = \overline{B}\,C + C\,\overline{D} + B\,\overline{C} + B\,\overline{C} + A$$



Schematic of BCD to 7-Segment Decoder

Magnitude Comparator

- A combinational circuit that **compares two digits or binary numbers** in order to find out whether one binary number is equal, less than or greater than the other.
- Three output terminals, one for **A > B** condition, one for **A = B** condition and one for **A < B** condition.

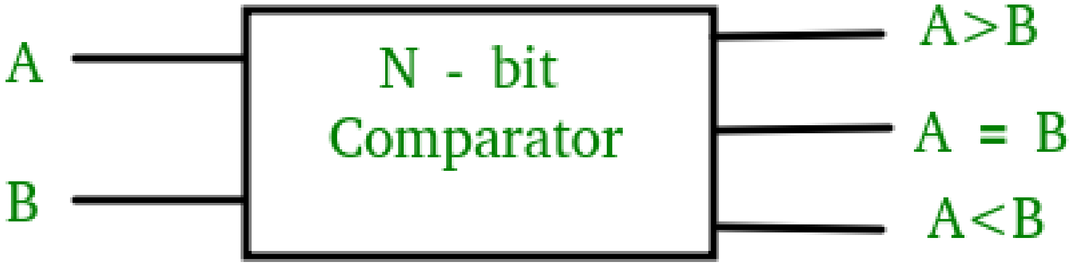


Fig: Block diagram for n-bit comparator

1-Bit Magnitude Comparator

- Compares two bits.
- Two inputs each for two single bit numbers and three outputs.

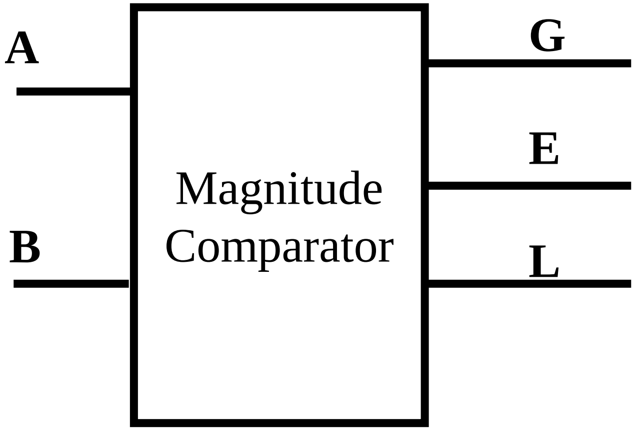


Fig: Block diagram for 1-bit comparator

A	B	G (A>B)	L (A<B)	E (A=B)
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

Truth Table

From the above truth table, the Boolean expressions of each output functions can be written as;

G=A$\bar{B}$

L=$\bar{A}B$

E=$\bar{A}\bar{B} + AB = A \odot B$

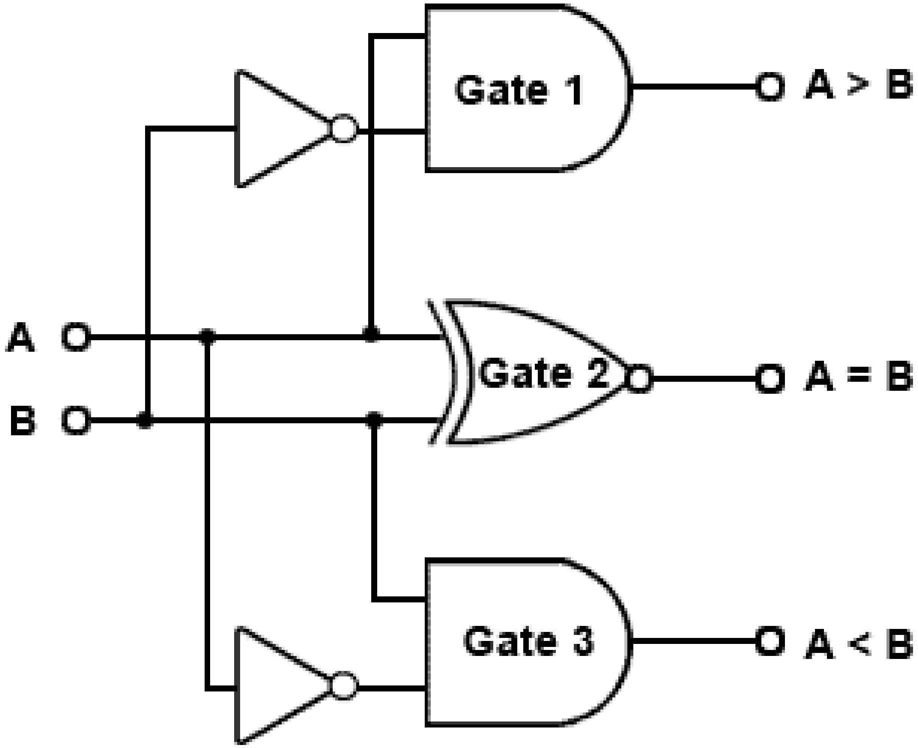


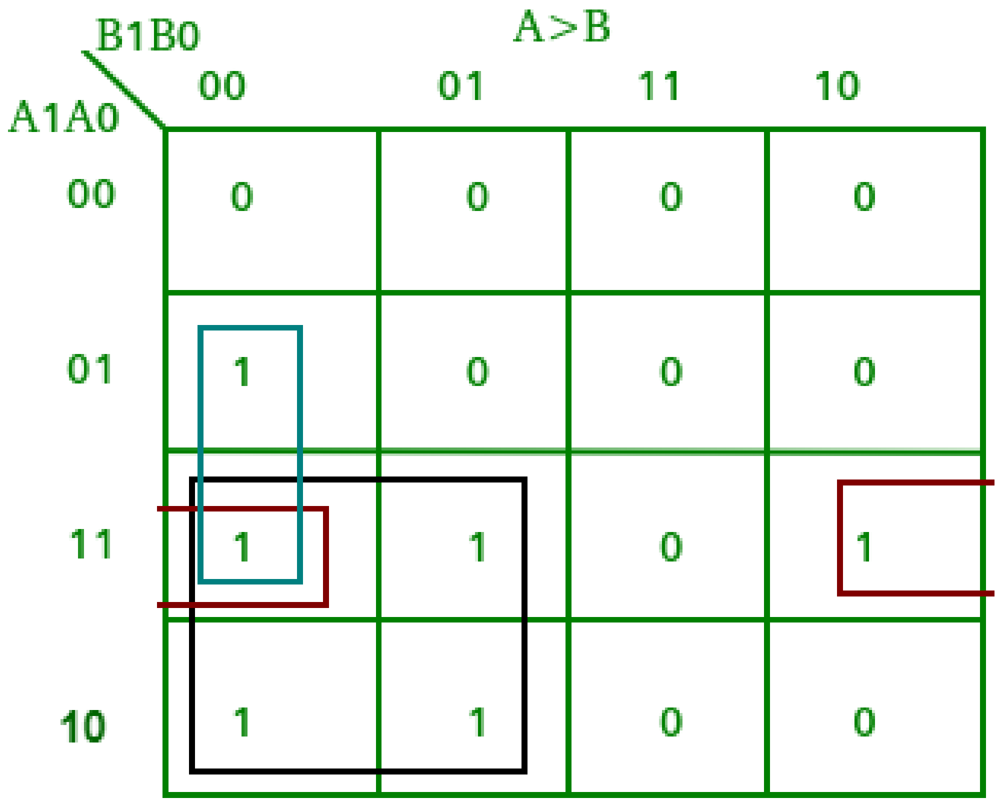
Fig: Circuit diagram for 1-bit comparator

2-bit comparator

A1	A0	B1	B0	A>B	A<B	A=B
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	0	1

Truth Table

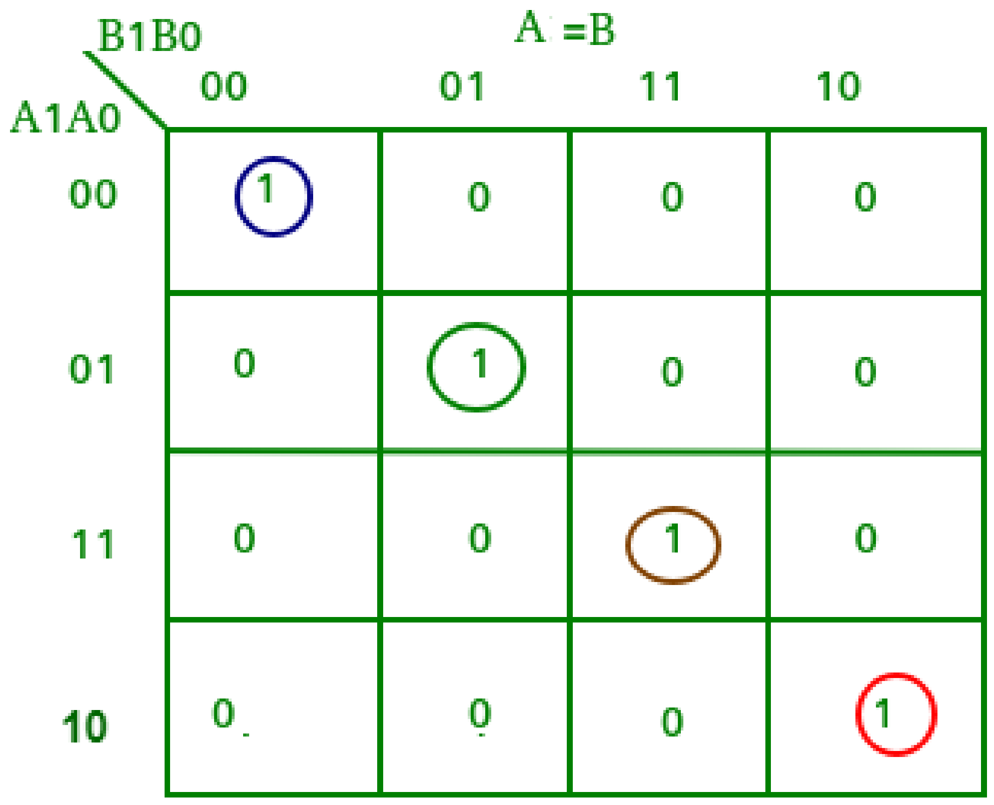
For G (A>B)



A>B

B1B0 A1A0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

For G (A=B)



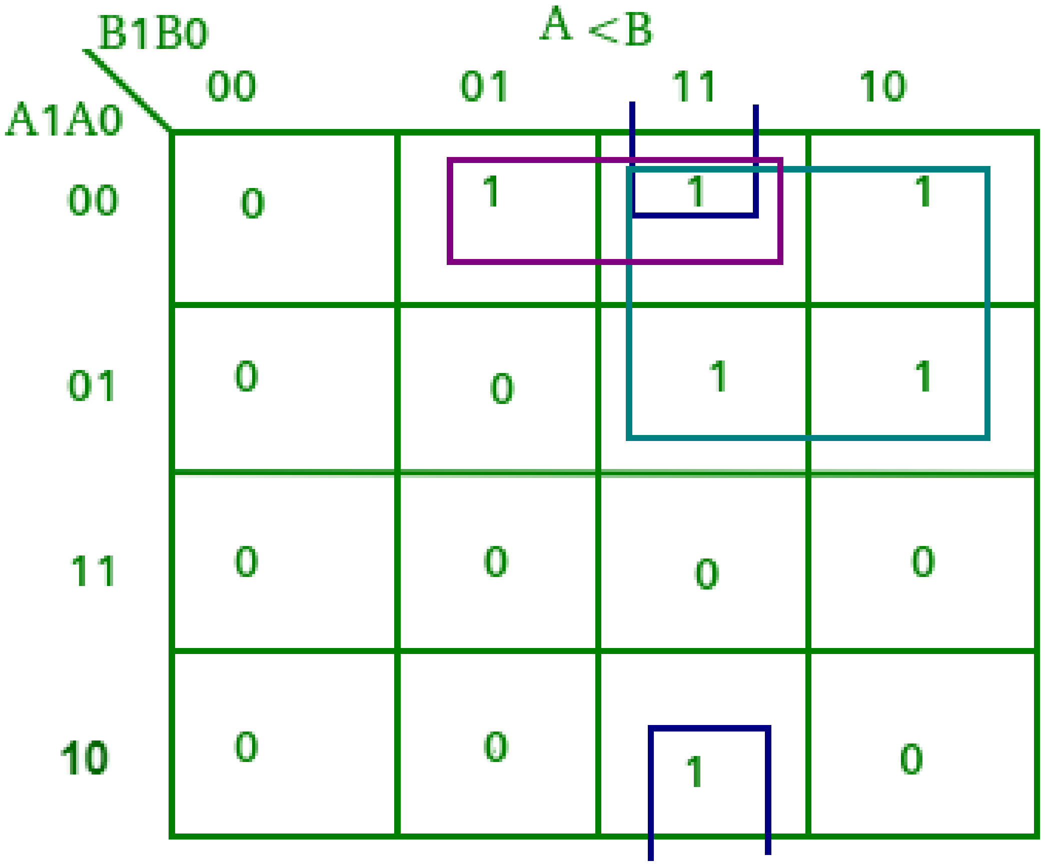
A =B

B1B0 A1A0	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

G (A>B) = A1B1’ + A0B1’B0’ + A1A0B0’

E (A+B) = (A0 ⊙ B0) (A1 ⊙ B1)

For L (A<B)



A <B

A1A0 \ B1B0	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

L(A<B) = A1’B1 + A0’B1B0 + A1’A0’B0

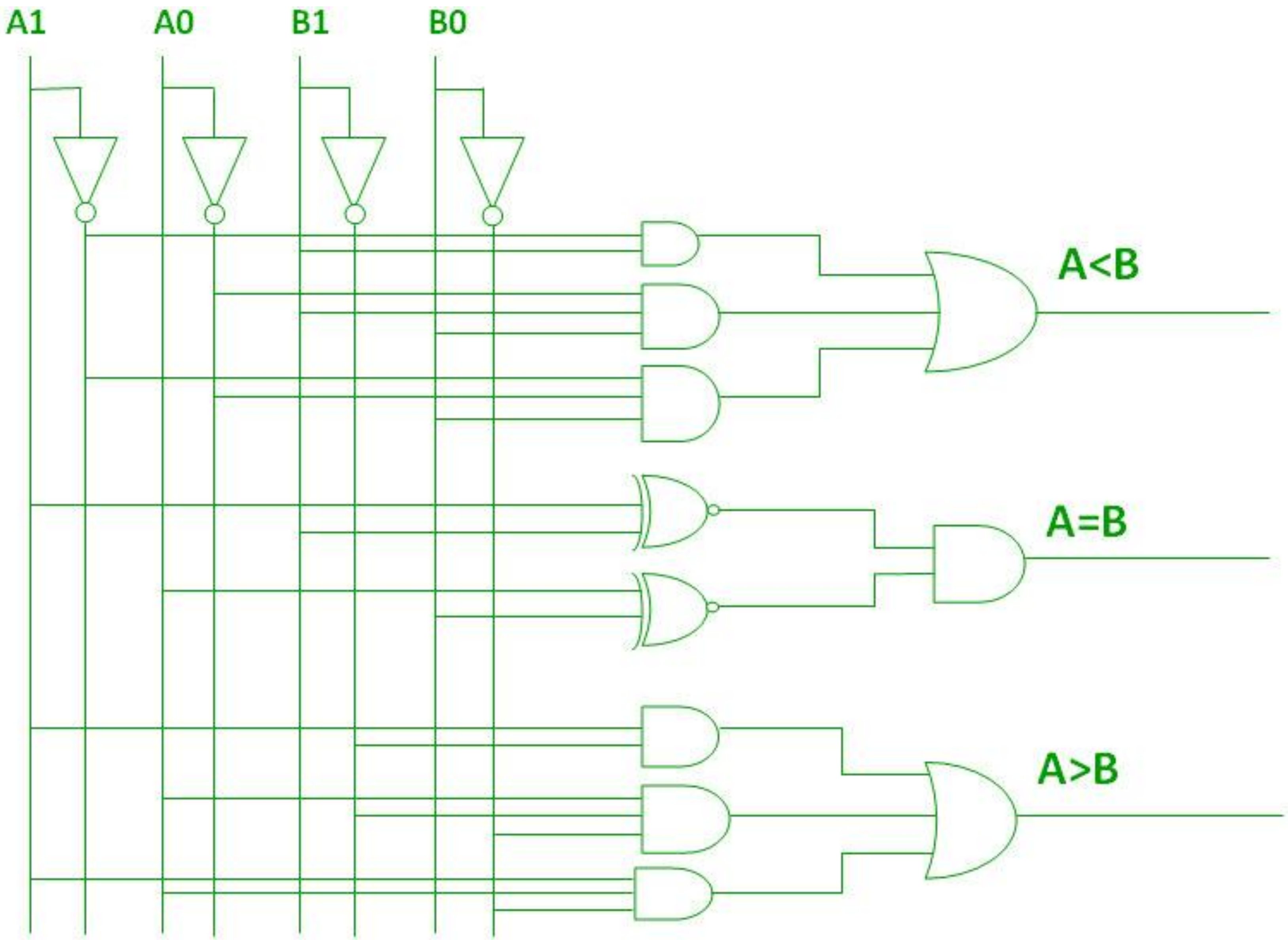


Fig: Circuit diagram for 2-bit comparator

Parity Generator and Checker

- **3-bit odd parity generator:**

$$P = \overline{A}\bar{B}\bar{C} + A\bar{B}C + \bar{A}BC + AB\bar{C}$$

$$= A\odot(B\oplus C)$$

K-map

A \ BC	00	01	11	10
0	1	0	*1*	0
1	0	1	*0*	*1*

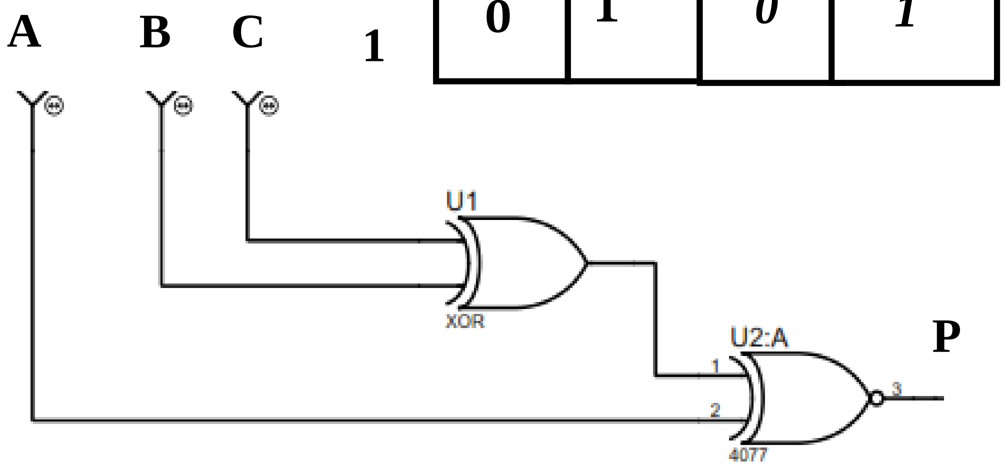


Fig: Circuit diagram for 3-bit odd parity generator

Information	Odd Parity bit
ABC	**P**
000	1
001	0
010	0
011	1
100	0
101	1
110	1
111	0

4-bit odd parity checker:

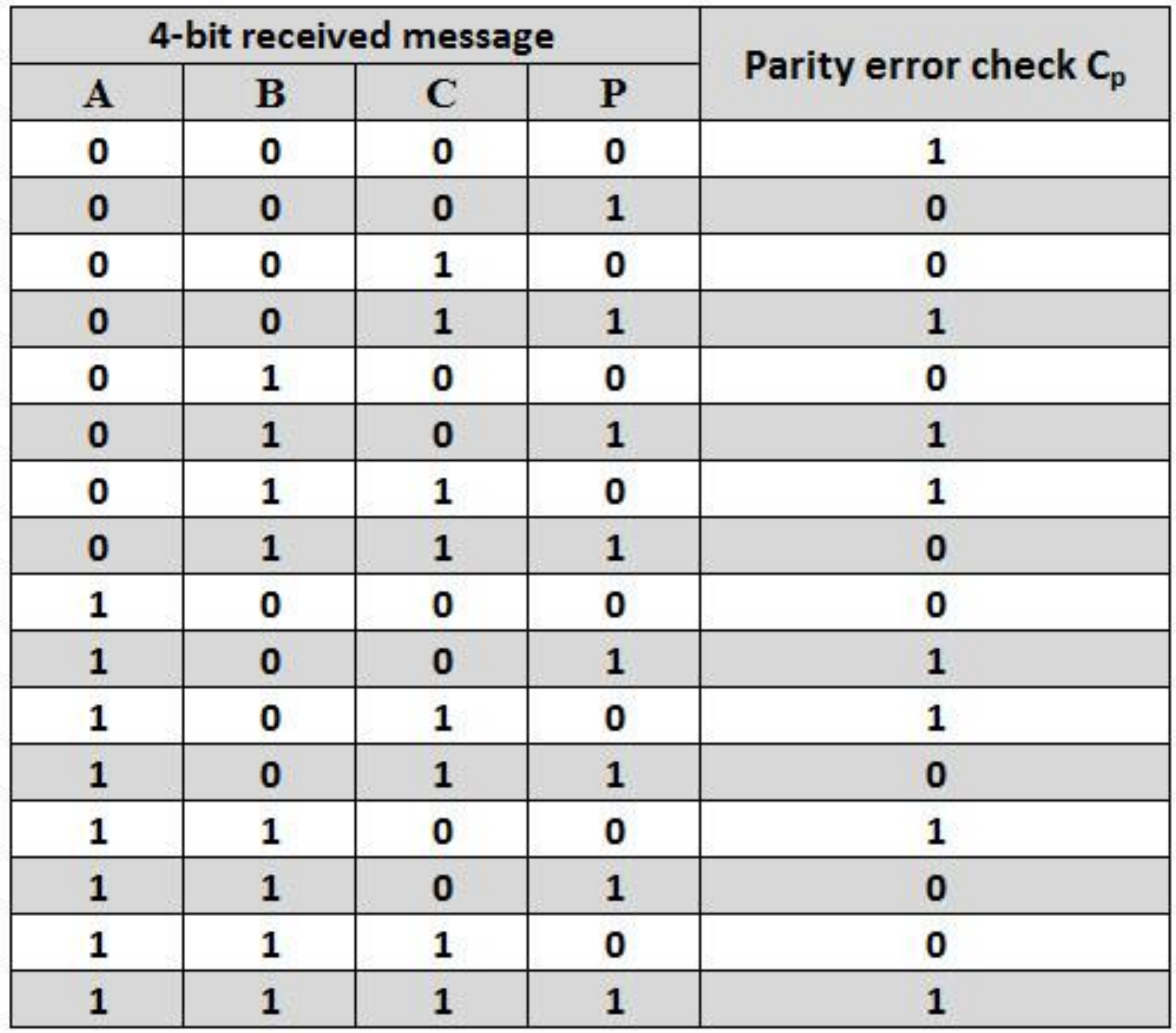
K-map

AB \ CP	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	*1*	*0*	*1*	*0*
10	*0*	*1*	*0*	*1*

$C_p = (A\odot B)(C\odot P) + (A\oplus B)(C\oplus P)$

Let $X = A\oplus B$ AND $Y = C\oplus P$

$\therefore C_P = X\odot Y$



4-bit received message				Parity error check C_p
A	B	C	P	
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

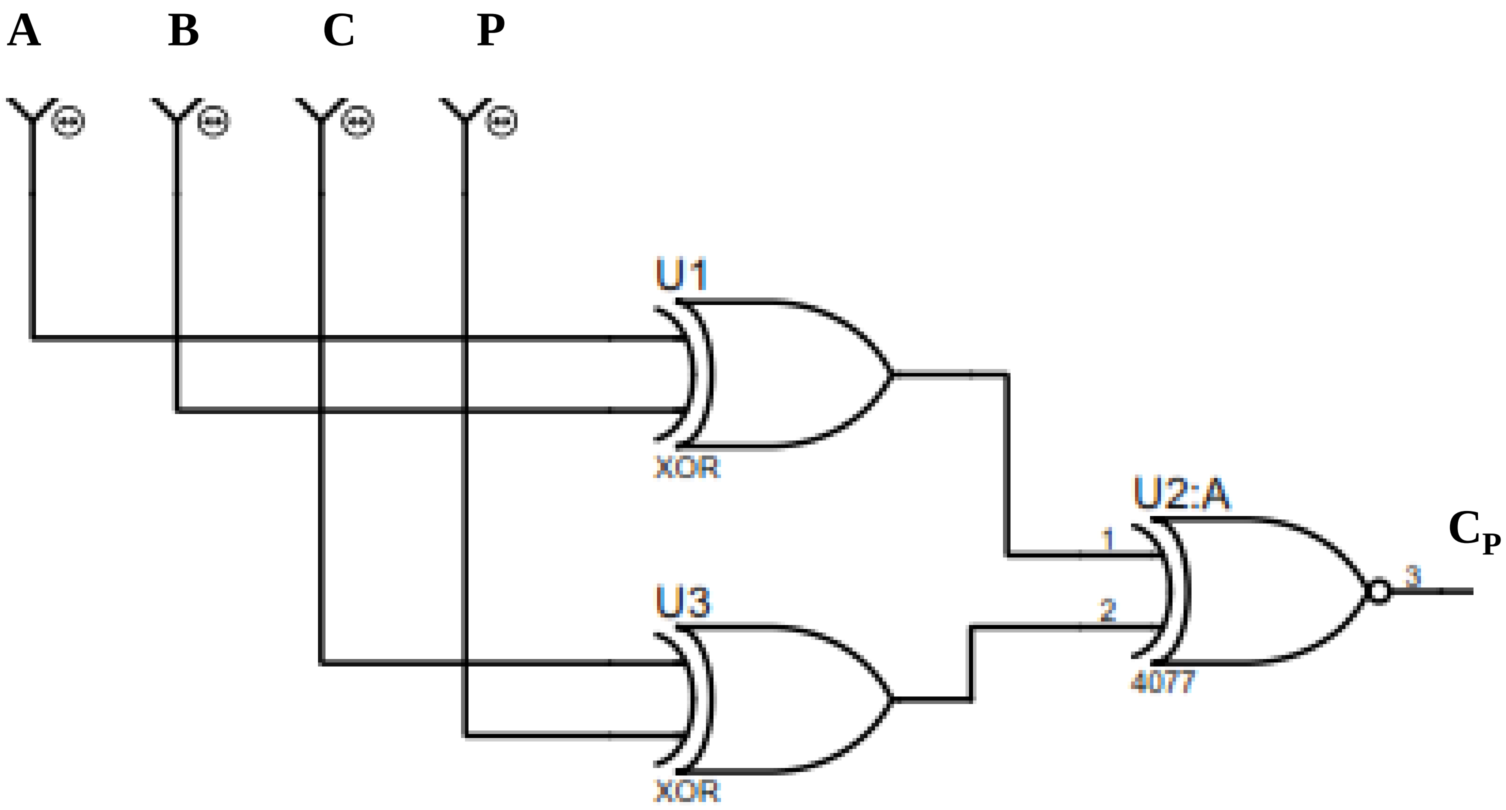


Fig: Circuit diagram for 4-bit odd parity checker

Read only Memory

- **Read Only Memory (ROM)-**Memory (Storage) device
- Special links that can be fused or broken to store data.
- Non-volatile

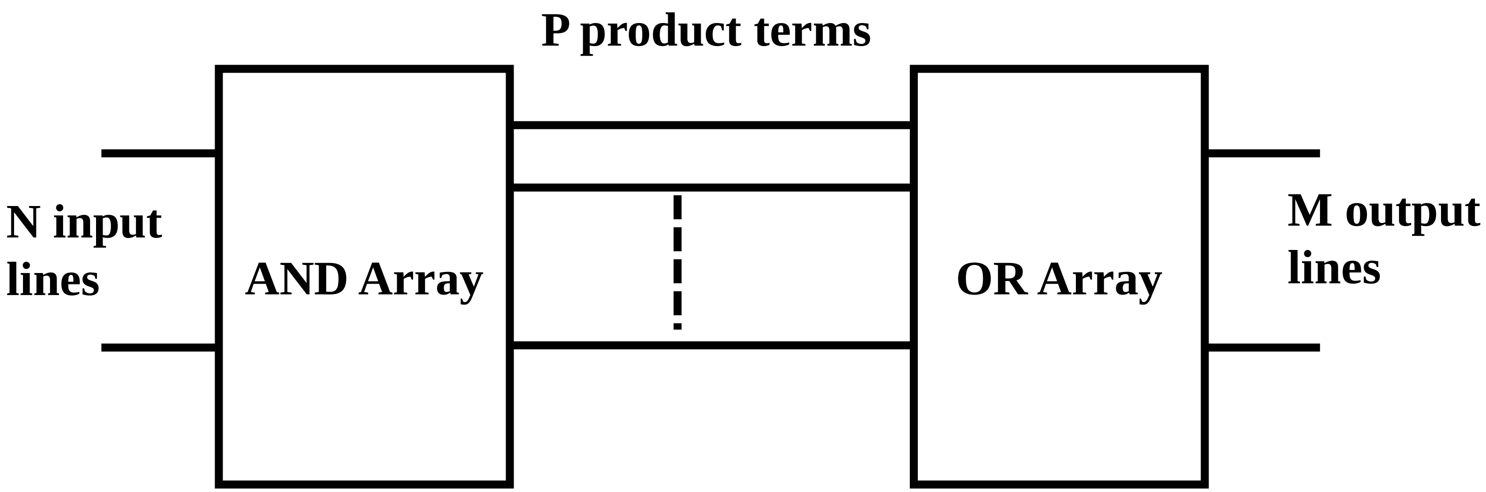
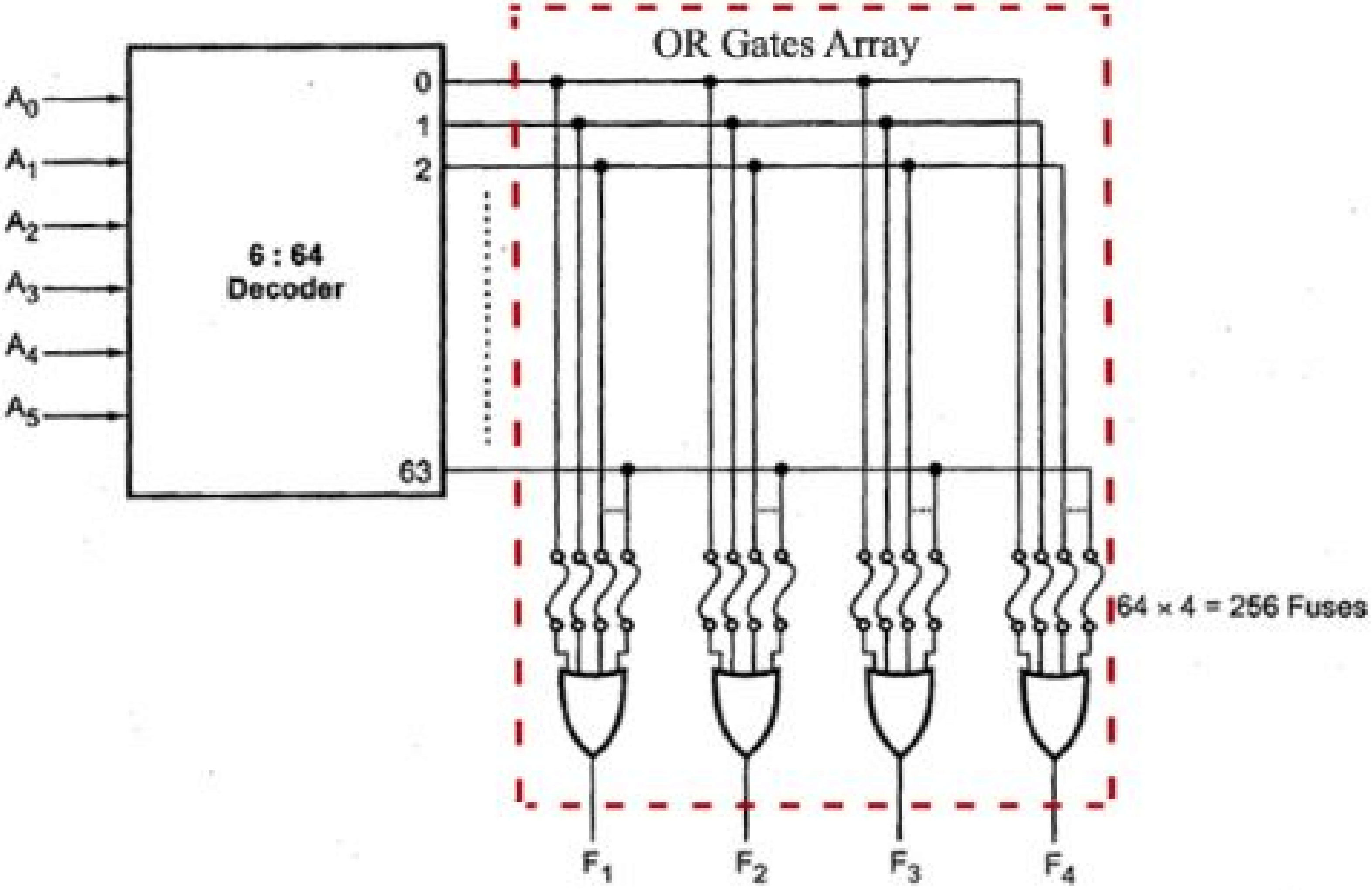


Fig: Block diagram of ROM



Internal construction of 64x4 ROM

Types:

1. Mask ROM (MROM)
2. Programmable ROM (PROM)
3. Erasable and Programmable ROM (EPROM)
    a. **Electrically Erasable and Programmable ROM (EEPROM)**
    b. **Ultraviolet EPROM (UV EPROM)**
4. FLASH ROM

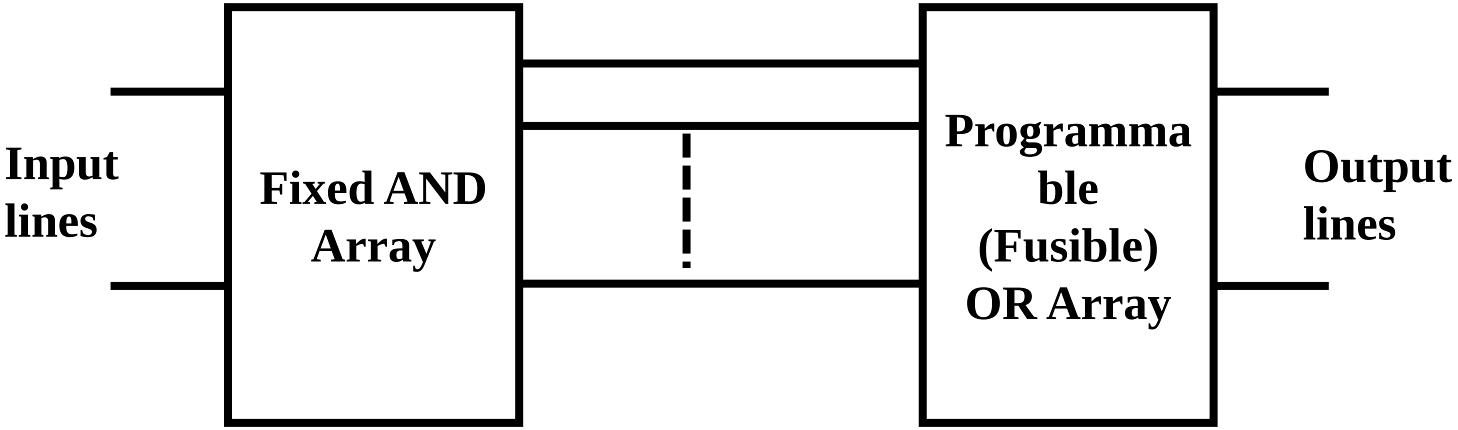


Fig: Block diagram of PROM

Programmable Array Logic (PAL)

- Most common one time programmable (OTP) logic device.
- Implemented with bipolar technology (TTL or ECL).

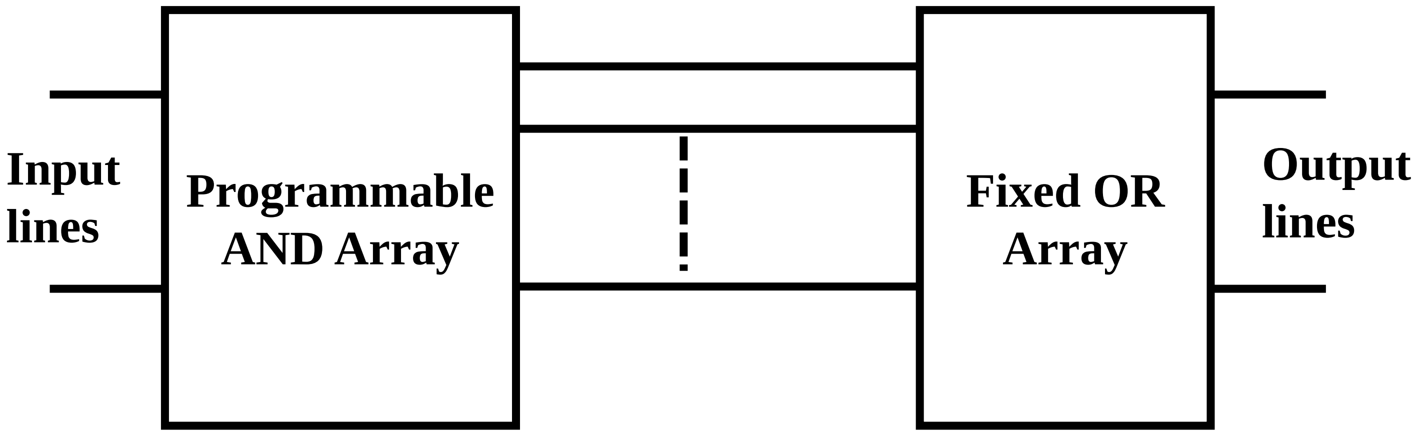


Fig: Block diagram of PAL

Programmable Logic Array (PLA)

- Also called Field Programmable Logic array (FPLA).
- User programs as per his/her requirement.

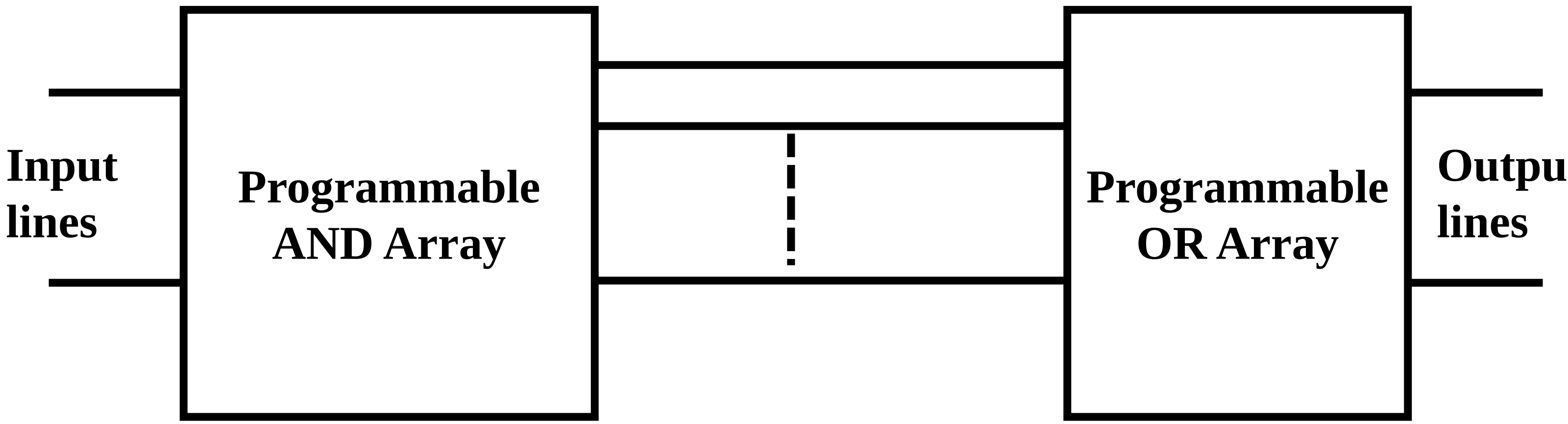
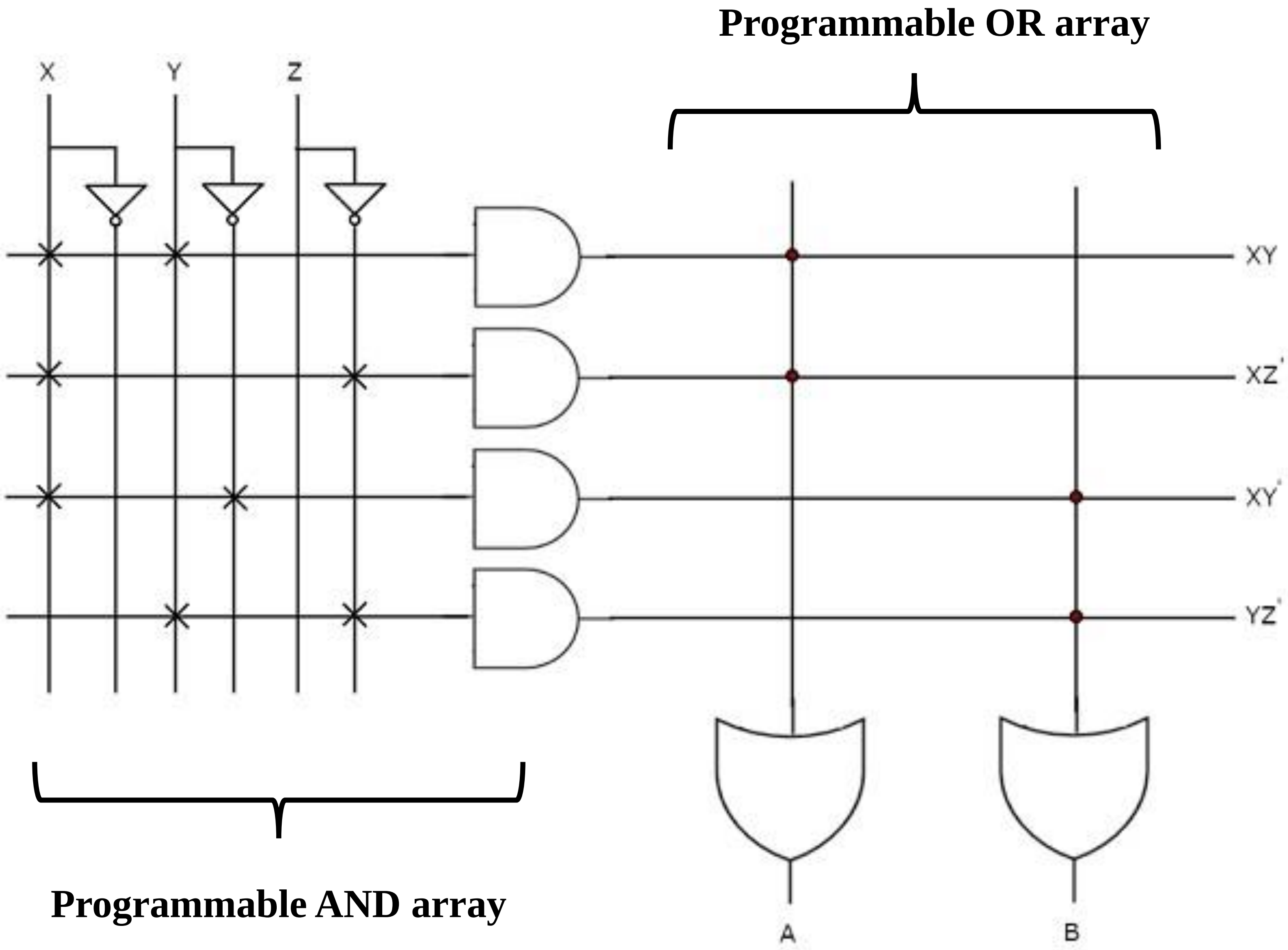


Fig: Block diagram of PLA



Programmable OR array

Programmable AND array

Fig: Circuit diagram of PLA

EX: Implement function using PLA**:**

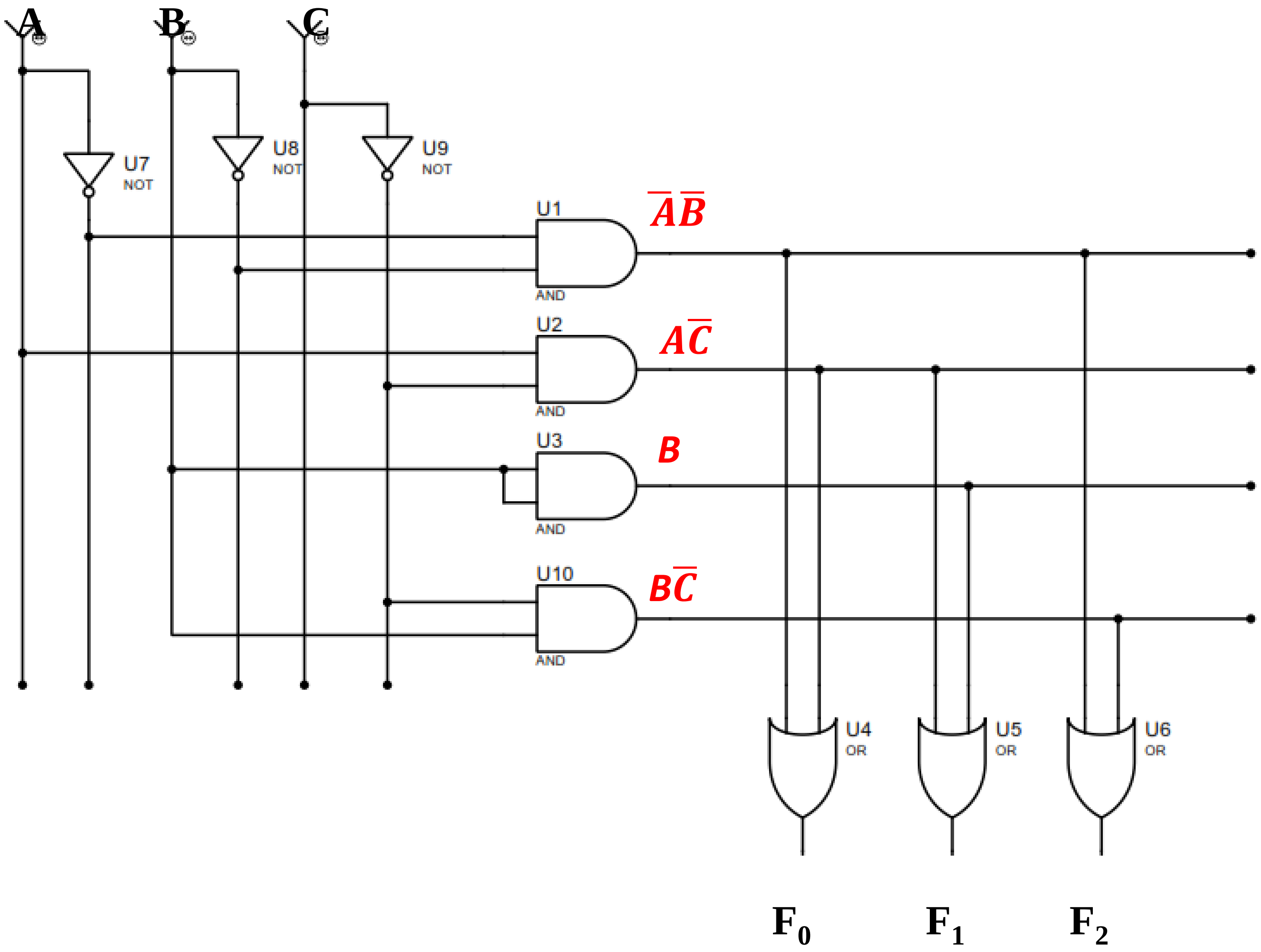
F_0=∑m(0,1,4,6), F_1=∑m(2,3,4,6,7), F_2=∑m(0,1,2,6)

$$F_0 = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + A\overline{B}\,\overline{C} + AB\overline{C} = \overline{A}\,\overline{B} + A\overline{C}$$

$$F_1 = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C} + ABC = B + A\overline{C}$$

$$F_2 = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + AB\overline{C} = \overline{A}\,\overline{B} + B\overline{C}$$

Product	**Inputs**			**Outputs**		
	A	***B***	***C***	F_0	F_1	F_2
$\overline{A}\,\overline{B}$	0	0	-	1	0	1
$A\overline{C}$	1	-	0	1	1	0
B	-	1	-	0	1	0
$B\overline{C}$	-	1	0	0	0	1


A
B
C
U7
NOT
U8
NOT
U9
NOT
U1
$\overline{A}\overline{B}$
AND
U2
$A\overline{C}$
AND
U3
B
AND
U10
$B\overline{C}$
AND
U4
OR
U5
OR
U6
OR
F_0
F_1
F_2

References

- **M. Morris Mano**, “Digital Logic & Computer Design”
- **Brain Holdsworth**, “Digital Logic Design”, Elsevier Science.
- **John Patrick Hayes**, “Introduction to Digital Logic Design”, **Addison-Wesley.**
- **M. Morris Mano and Charles Kime**, “Logic and Computer Design Fundamentals”, Pearson New International.
- Wikipedia/Geeks For Geeks/Electronics Hub