①. PROGRAMMABLE LOGIC ARRAY :- (PLA)

× → Programmable
• → Fixed

- AND & OR gates are Programmable.

PROBLEM 1 :

① Implement the following functions using PLA

$F_1 = \Sigma m(1, 2, 4, 6)$, $F_2 = \Sigma m(0, 1, 6, 7)$, $F_3 = \Sigma m(2, 6)$.

Step 1 : Truth table.

Inputs			Outputs		
A	B	C	F_1	F_2	F_3
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	1	0

Step 2: K- Map

F_1

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0		1		1
A 1	1			1

$$\boxed{F_1 = \bar{A}\bar{B}C + A\bar{C} + B\bar{C}}$$

F_2:

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	1	1		
A 1			1	1

$$\boxed{F_2 = \bar{A}\bar{B} + AB}$$

F_3:

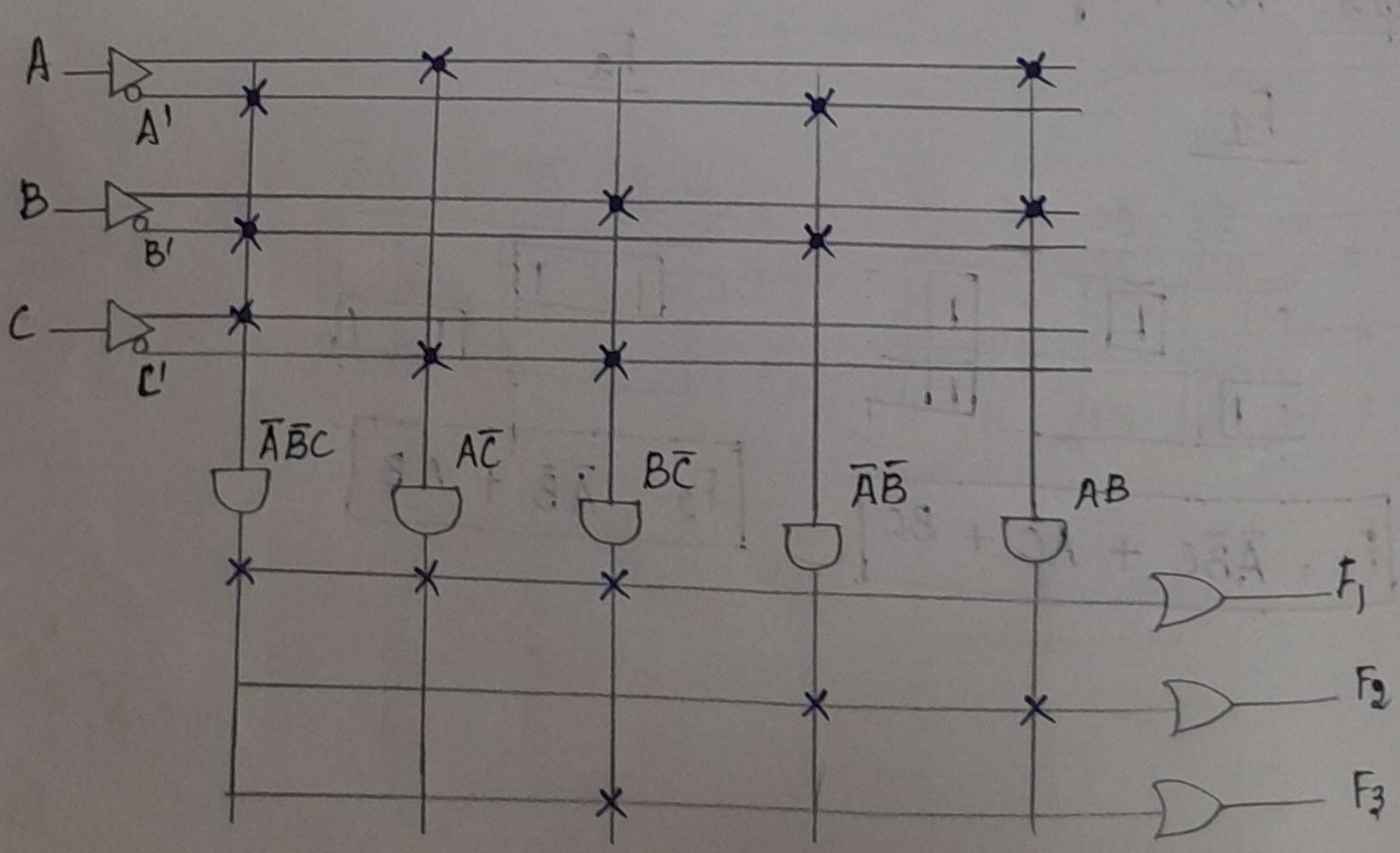
A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0				1
A 1				1

$$\boxed{F_3 = B\bar{C}}$$

Step 3: PLA Program table:

Product terms	Inputs			Outputs		
	A	B	C	F_1	F_2	F_3
$\bar{A}\bar{B}C$	0	0	1	1	–	–
$A\bar{C}$	1	–	0	1	–	–
$B\bar{C}$	–	1	0	1	–	1
$\bar{A}\bar{B}$	0	0	–	–	1	–
AB	1	1	–	–	1	–

Step 4: Logic diagram:



② PROGRAMMABLE ARRAY LOGIC: (PAL)

- AND gate → Programmable
- OR gate → Fixed.

PROBLEM:

Implement the following function using PAL

$F_1 = \Sigma m(3, 5, 6, 7)$, $F_2 = \Sigma m(0, 1, 6, 7)$

Step 1: Truth table

Inputs			Outputs	
A	B	C	F_1	F_2
0	0	0	0	1
0	0	1	0	1
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

Step 2: K-map

F_1

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1	3	2
A 1	4	5	7	6

$\boxed{F_1 = AC + BC + AB}$

F_2

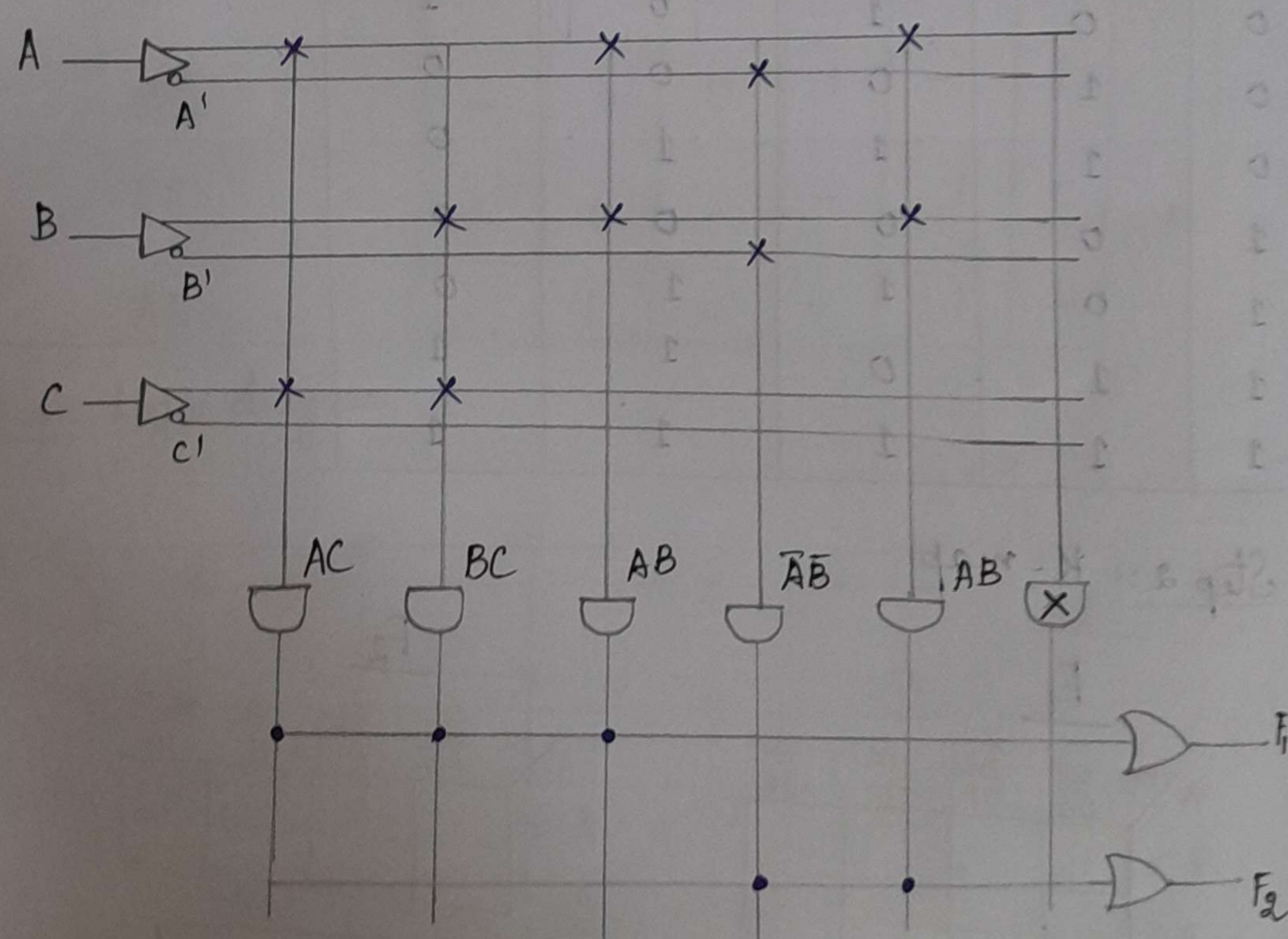
A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1	3	2
A 1	4	5	7	6

$\boxed{F_2 = \bar{A}\bar{B} + AB}$

Step 3: PAL Program table:

Product terms	Inputs			Outputs	
	A	B	C	F_1	F_2
AC	1	-	1	1	-
BC	-	1	1	1	-
AB	1	1	-	1	1
$\bar{A}\bar{B}$	0	0	-	-	1

Step 4: Logic diagram:



③. PROGRAMMABLE READ ONLY MEMORY: (PROM)

- AND gate → Fixed
- OR gate → Programmable

PROBLEM:

Design a Combinational circuit using a PROM, the circuit accepts 3 bit binary number and generates its equivalent Excess-3-code.

Step 1: Truth table.

Inputs			Outputs			
			8	4	2	1
B_2	B_1	B_0	E_3	E_2	E_1	E_0
0	0	0	0	0	1	1
0	0	1	0	1	0	0
0	1	0	0	1	0	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	0	0
1	1	0	1	0	0	1
1	1	1	1	0	1	0

Min terms:

$E_0 = \Sigma m(0, 2, 4, 6)$

$E_1 = \Sigma m(0, 3, 4, 7)$

$E_2 = \Sigma m(1, 2, 3, 4)$

$E_3 = \Sigma m(5, 6, 7)$

Step 3: Logic diagram: