Introduction

Monday, 20 September 2021 1:32 PM

introduction

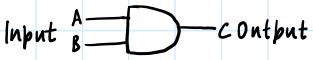
Sujith Alwyn Castelino

- Analog information can take any values b/w the range.
- Digital information takes discrete values. (either 0 or 1 – binary)

→ TRUTH TABLE:

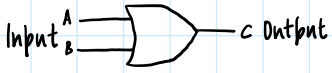
- depicts outputs for all possible input combinations.

AND C = A.B



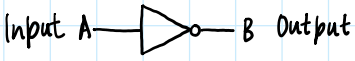
INPUTS		OUTPUT
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

OR C = A + B



INPUTS		OUTPUT
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

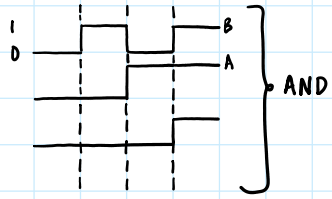
NOT B = A'

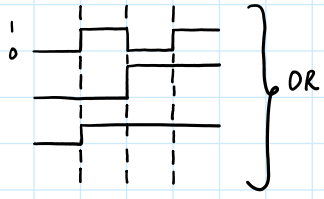


INPUT	OUTPUT
A	C
0	1
1	0

→ TIMING DIAGRAM:

eg:



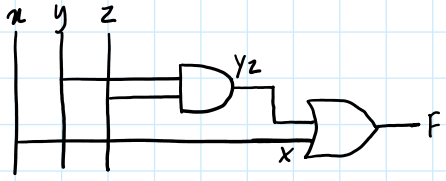


→ BOOLEAN FUNCTION:

eg: $F(x, y, z) = x + yz$

2^3 inputs = 8

	2^2	2^1	2^0 → least significant	
	x	y	z	Output
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1

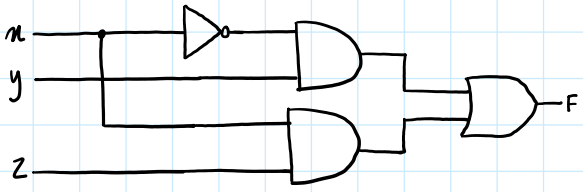


2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

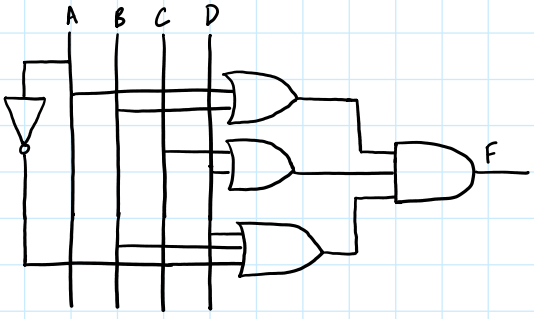
eg: find Boolean fnctn. and truth table.

	x	y	z	Output
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

$F(x,y,z) = \bar{x}y + xz$



eg: $F = (A+B)(C+D)(A'+B+D)$



→ BOOLEAN THEOREM:

Table 2.1
Postulates and Theorems of Boolean Algebra

Postulate 2	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
Theorem 1	(a)	$x + x = x$	(b)	$x \cdot x = x$
Theorem 2	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$
Theorem 3, involution		$(x')' = x$		
Postulate 3, commutative	(a)	$x + y = y + x$	(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
Theorem 6, absorption	(a)	$x + xy = x$	(b)	$x(x + y) = x$

eg: (i) $X(X'+Y) = XY$

(ii) $X + X'Y = (X + X')(X+Y) = X+Y$

(iii) $X' + XY = X' + Y$

(iv) $(X+Y)(X+Y') = XX + XY' + YX + YY'$

$= XX + X = X(1+X) = X$

(iv) $(X+Y)(X+Y') = XX + XY' + YX + YY'$

$= XX + X = X(1+X) = X$

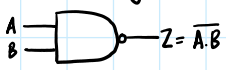
eg: $A'B(D'+C'D) + B(A+A'CD)$

$= \bar{A}B\bar{D} + \bar{A}B\bar{C}\underline{D} + AB + \bar{A}BC\underline{D}$

$= \bar{A}B\bar{D} + \bar{A}BD + AB = \bar{A}B + AB = B$

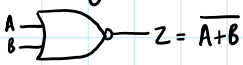
→ UNIVERSAL GATES:

NAND gates:



A	B	Z= $\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR gates:

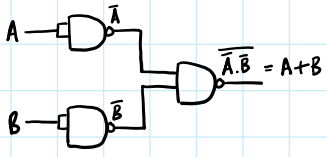


A	B	Z= $\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

eg: OR using NAND gates:

$Y = \overline{A.B} = \bar{A} + \bar{B}$

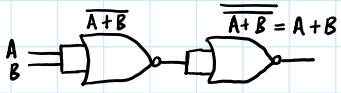
$\overline{\bar{A}.\bar{B}} = \bar{\bar{A}} + \bar{\bar{B}} = A + B$



eg: OR using NOR gates:

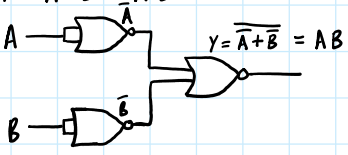
$Y = \overline{A+B}$

$A+B = \overline{\overline{A+B}}$



eg: AND using NOR gates:

$Y = \overline{\bar{A}+\bar{B}} = \bar{A}.\bar{B}$



eg:

(i) Truth table

$\Sigma F(2,3,5,7) = \bar{A}B + AC$

	A	B	C	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

(ii) Drawing the circuit

eg:

(i) Truth table

A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	1	0	0	1

$Y_3 = A_1 . A_0 = \Sigma m(3)$

$Y_2 = A_1 . \bar{A}_0 = \Sigma m(2)$

$Y_1 = 0$

$Y_0 = \bar{A}_1 A_0 + A_1 A_0 = \Sigma m\ (1,3)$

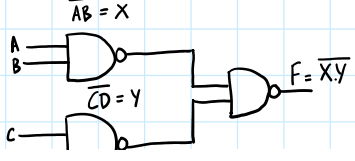
$= A_0$

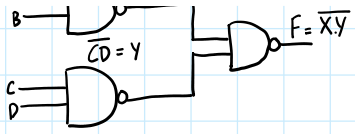
(ii) Draw the circuit.

eg: $F(A,B,C,D) = AB + CD$ using only NAND gates

$\bar{F} = \overline{AB + CD} = \overline{AB} . \overline{CD}$

$\bar{\bar{F}} = \overline{\overline{AB} . \overline{CD}} = F$





eg: $F(x,y,z) = xz + y'z + x'yz$ using only NAND gates.

minterm & maxterm

→ MINTERMS AND MAXTERMS:

eg: Consider a 3-variable Boolean fnctn. F(a, b, c).

	Inputs A B C	Minterms (product term)	Decimal notations (minterms)	Maxterms	Decimal notations (maxterms)
0	0 0 0	$\bar{A}\bar{B}\bar{C}$	m_0	$A+B+C$	M_0
1	0 0 1	$\bar{A}\bar{B}C$	m_1	$A+B+\bar{C}$	M_1
2	0 1 0	$\bar{A}B\bar{C}$	m_2	$A+\bar{B}+C$	M_2
3	0 1 1	$\bar{A}BC$	m_3	$A+\bar{B}+\bar{C}$	M_3
4	1 0 0	$A\bar{B}\bar{C}$	m_4	$\bar{A}+B+C$	M_4
5	1 0 1	$A\bar{B}C$	m_5	$A+\bar{B}+C$	M_5
6	1 1 0	$AB\bar{C}$	m_6	$\bar{A}+\bar{B}+C$	M_6
7	1 1 1	ABC	m_7	$\bar{A}+\bar{B}+\bar{C}$	M_7

eg: $f(x,y,z)$ → truth table given

x y z	f
0 0 0	1 ✓
0 0 1	1 ✓
0 1 0	0 ✗
0 1 1	0 ✗
1 0 0	1 ✓
1 0 1	1 ✓
1 1 0	1 ✓
1 1 1	0 ✗

(i) Sum of minterms.

$f(x,y,z) = \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + \bar{x}\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z}$

$= m_0 + m_1 + m_4 + m_5 + m_6$

$= \sum_m (0,1,4,5,6)$ → canonical form

$= \bar{y} + xy\bar{z} = \bar{y} + x\bar{z}$ → standard form.

(ii) Product of maxterms

$= (x+\bar{y}+z).(x+\bar{y}+\bar{z}).(\bar{x}+\bar{y}+\bar{z})$

$= M_2 . M_3 . M_7 = \Pi M(2,3,7)$

$= (x+\bar{y})(\bar{y}+\bar{z})$

→ RELATIONSHIP B/W SOM & POM:

→ RELATIONSHIP B/W SOM & POM:

$F'(x,y,z) = \sum_m (2,3,7) = m2 + m3 + m7$

Taking complement on both sides and applying De Morgan's theorem,

$F(x,y,z) = (m2 + m3 + m7)' = m2' . m3' . m7'$

$= (x'yz')'(x'yz)' . (xyz)' = (x+y'+z) . (x+y'+z') . (x'+y'+z')$

$= M2 . M3 . M7$

$= \Pi (2,3,7)$

$mj' = Mj$

→ TWO METHODS:

① Identify the missing term and include them in the expression using the postulates:
$x + 0$, $x . 1 = 1$, $x + x' = 1$, $x . x' = 0$.

eg: $F(a,b,c) = ab' + c$ using method 1.

$= a\bar{b} . 1 + c' . 1$

$= a\bar{b}(c+c') + \bar{c}(a+\bar{a})$

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

$= a\bar{b}c + a\bar{b}c + a\bar{c}(b+\bar{b}) + \bar{a}\bar{c}(b+\bar{b})$

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

$= m0, m5, m4, m6, m2)$

② Write truth table from given expression and then write sum of mid terms and product of maxterms

eg: $F(a,b,c) = ab' + c'$ using method 2.

a b c	f
0 0 0	1
0 0 1	0
0 1 0	1
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1

$f(a,b,c) = \sum m(0,2,4,5,6)$

$= \Pi M(1,3,7)$

1 0 1	1
1 1 0	1
1 1 1	0

- Steps to solve:
  (i) Truth table
  (ii) $F = \Sigma m(\,)$ or $\Pi M(\,)$
  (iii) Simplified expression for F_2
  (iv) Draw the circuit

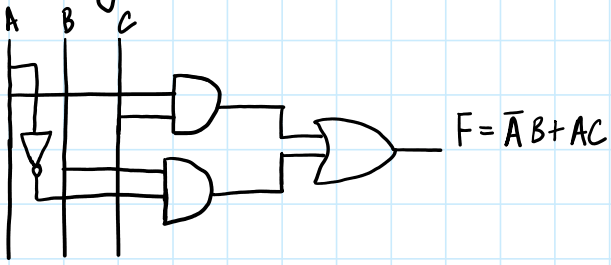
eg: Design a combinational circuit that takes 3-bit input & generates an output high whenever the input is a prime no.

(i) Truth table

	A B C	F
0	0 0 0	0
1	0 0 1	0
2	0 1 0	1
3	0 1 1	1
4	1 0 0	0
5	1 0 1	1
6	1 1 0	0
7	1 1 1	1

$\Sigma F(2,3,5,7) = \bar{A}B + AC$

(ii) Drawing the circuit



eg: Design a combinational circuit that takes 2-bit inputs and outputs the square of the input.

(i) Truth table

A_1 A_0	Y_3	Y_2	Y_1	Y_0
0 0	0	0	0	0

$Y_3 = A_1 . A_0 = \Sigma m(3)$
$Y_2 = A_1 . \bar{A}_0 = \Sigma m(2)$

A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	1	0	0	1

$Y_3 = A_1 . A_0 = \Sigma_i m(3)$

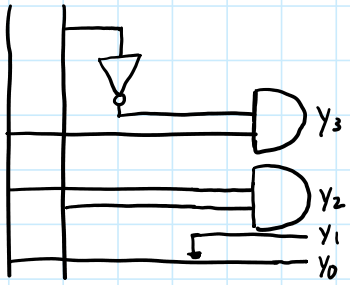
$Y_2 = A_1 . \overline{A_0} = \Sigma_i m(2)$

$Y_1 = 0$

$Y_0 = \overline{A_1} A_0 + A_1 A_0 = \Sigma_i m\ (1,3)$

$= A_0$

(ii) Draw the circuit.

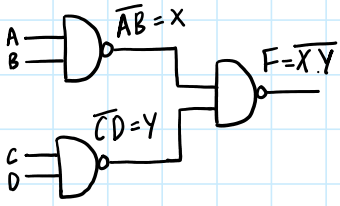


→ DRAWING CKT. USING ONLY UNIVERSAL GATES:

eg: $F(A,B,C,D) = AB + CD$ using only NAND gates.

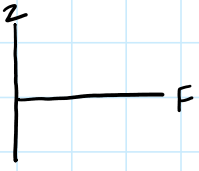
$\overline{F} = \overline{AB+CD} = \overline{AB} . \overline{CD}$

$\overline{\overline{F}} = \overline{\overline{AB} . \overline{CD}} = F$



eg: $F(x,y,z) = xz + y'z + x'yz$ using only NAND gates.

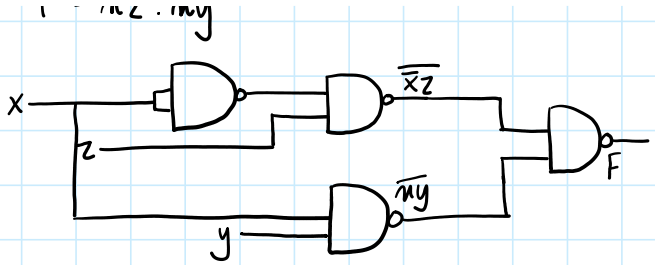
$= xz + z(y' + x'y) = xz + \bar{y}z + \bar{x}z = z$



eg: $F(x,y,z) = \bar{x}z + xy$ using only NAND gates

$\overline{F} = \bar{x}z . \overline{xy}$

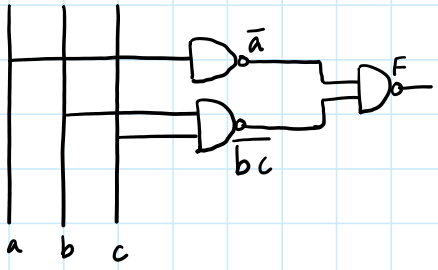
$\overline{\overline{F}} = \overline{\bar{x}z . \overline{xy}}$



eg: $F(a,b,c) = a + bc$ using only NAND gates.

$\overline{F} = \overline{a}.\overline{bc}$

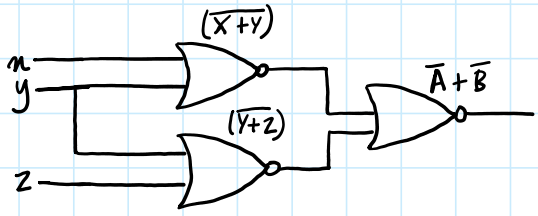
$\overline{\overline{F}} = \overline{\overline{a}.\overline{bc}}$



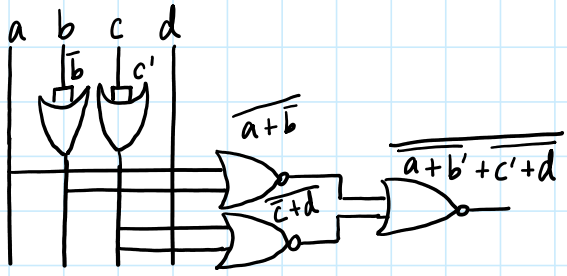
eg: $F(x,y,z) = (x+y)(y+z)$ using only NOR gates.

$\overline{F} = \overline{x+y} + \overline{y+z}$

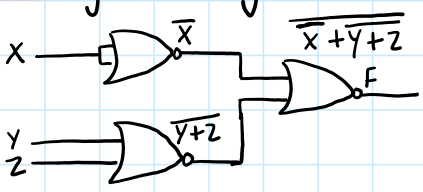
$= \overline{\overline{(x+y)} + \overline{(y+z)}}$

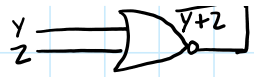


eg: $F(a,b,c,d) = (a+b')(c'+d') = \overline{\overline{a+b'} + \overline{c'+d'}}$

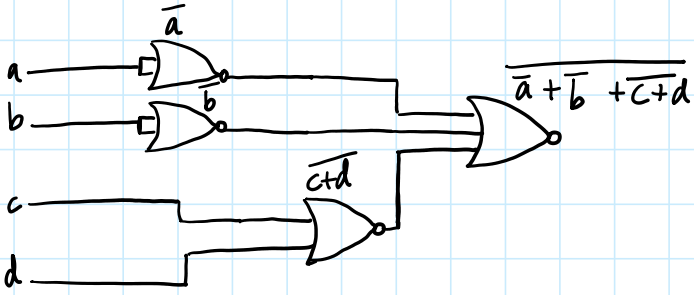


eg: $F(x,y,z) = x.(y+z) = \overline{\overline{x} + \overline{y+z}}$





eg: $f(a,b,c,d) = a.b(c+d) = \overline{\bar{a}+\bar{b}+(\overline{c+d})}$



Karnaugh-map

→ K-MAP:

- Pictorial form of truth table.
- Graphical tool to simplify a logical eqn. by forming groups of cells.
- Why simplificatn? To reduce cost.

2 variable K-Map

A	B	SOP	POS
0	0	$\bar{A}.\bar{B}$	$A+B$
0	1	$\bar{A}.B$	$A+\bar{B}$
1	0	$A.\bar{B}$	$\bar{A}+B$
1	1	$A.B$	$\bar{A}+\bar{B}$

SOP:

A \ B	$\bar{B}$ 0	B 1
$\bar{A}$ 0	$\bar{A}.\bar{B}$ (m_0)	$\bar{A}.B$ (m_1)
A 1	$A.\bar{B}$ (m_2)	$A.B$ (m_3)

POS:

A \ B	B 0	$\bar{B}$ 1
A 0	$A+B$ (M_0)	$A+\bar{B}$ (M_1)
$\bar{A}$ 1	$\bar{A}+B$ (M_2)	$\bar{A}+\bar{B}$ (M_3)

eg: Solve using K-map.

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

$F(A,B)$. $SOP = \sum_m(0,1) = \bar{A}\bar{B} + \bar{A}B$

$POS = \prod_M(2,3) = (\bar{A}+B)(\bar{A}+\bar{B})$

A \ B	$\bar{B}$ 0	B 1
$\bar{A}$ 0	1 (m_0)	1 (m_1)
A 1	0 (m_2)	0 (m_3)

A \ B	B 0	$\bar{B}$ 1
A 0	1 (m_0)	1 (m_1)
$\bar{A}$ 1	0 (m_2)	0 (m_3)

eg: Solve using K-map.

A	B	F
0	0	1
0	1	1
1	0	0

A \ B	$\bar{B}$ 0	B 1
$\bar{A}$ 0	1	1
A 1	0	1

$F = \bar{A}\bar{B} + \bar{A}B + AB$

Explanatn:

$F = \bar{A}\bar{B} + \bar{A}B + AB$

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

$= \bar{A}(\bar{B}+B) + (\bar{A}+A)B$

0	1	1
1	0	0
1	1	0

A \	U	$\bar{U}$
1		

$F = \bar{A}\bar{B} + \bar{A}B + AB$

$= \bar{A} + B$

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

$= \bar{A}(\bar{B} + B) + (\bar{A} + A)B$

$= \bar{A} + B$

eg: Solve using K-map.

A	B	F
0	0	1
0	1	1
1	0	1
1	1	1

A \ B	$\bar{B}$ 0	B 1
$\bar{A}$ 0	1	1
A 1	1	1

$F = 1$

A \ B	B 0	$\bar{B}$ 1
A 0	1	1
$\bar{A}$ 1	1	1

$F = 1$

3 variable K-map.

A	B	C	SOP	POS
0	0	0	$\bar{A}\bar{B}\bar{C}$	$A+B+C$
0	0	1	$\bar{A}\bar{B}C$	$A+B+\bar{C}$
0	1	0	$\bar{A}B\bar{C}$	$A+\bar{B}+C$
0	1	1	$\bar{A}BC$	$A+\bar{B}+C$
1	0	0	$A\bar{B}\bar{C}$	$\bar{A}+B+C$
1	0	1	$A\bar{B}C$	$\bar{A}+B+\bar{C}$
1	1	0	$AB\bar{C}$	$\bar{A}+\bar{B}+C$
1	1	1	ABC	$\bar{A}+\bar{B}+\bar{C}$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	m_0 $\bar{A}\bar{B}\bar{C}$	m_1 $\bar{A}\bar{B}C$	m_3 $\bar{A}BC$	m_2 $\bar{A}B\bar{C}$
A 1	m_4 $A\bar{B}\bar{C}$	m_5 $A\bar{B}C$	m_7 ABC	m_6 $AB\bar{C}$

A \ BC	B+C 00	$B+\bar{C}$ 01	$\bar{B}+\bar{C}$ 11	$\bar{B}+C$ 10
A 0	M_0 $A+B+C$	M_1 $A+B+\bar{C}$	M_3 $A+\bar{B}+\bar{C}$	M_2 $A+\bar{B}+C$
$\bar{A}$ 1	M_4 $\bar{A}+B+C$	M_5 $\bar{A}+B+\bar{C}$	M_7 $\bar{A}+\bar{B}+C$	M_6 $\bar{A}+\bar{B}+C$

eg: Solve using K-map.

$F = \bar{A}C + \bar{A}B + A\bar{B}C + BC$

$F(A,B,C) = \bar{A}(\bar{B}+B)C + \bar{A}B(\bar{C}+C) + A\bar{B}C + (A+\bar{A})BC$

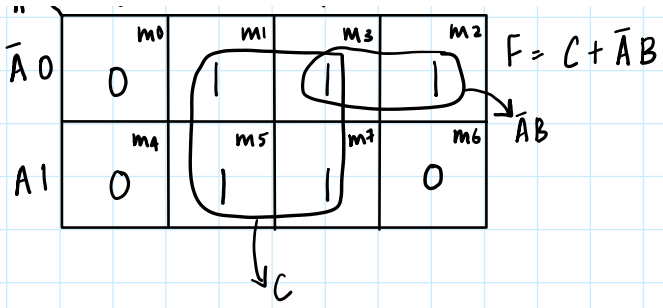
$= \bar{A}\bar{B}C + \bar{A}BC + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + ABC + \bar{A}BC$

$F = \sum_m (1,2,3,5,7)$ SOP $\quad \bar{F} \Rightarrow$ SOP $= \sum_m (0,4,6)$

$F = \Pi(0,4,6)$

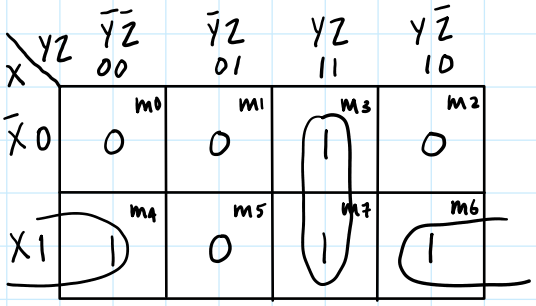
A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	m_0 0	m_1 1	m_3 1	m_2 1

$F = C + \bar{A}B$



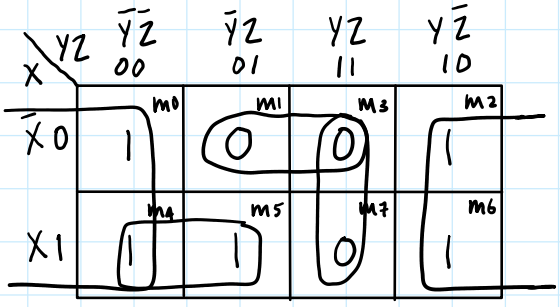
$F = C + \bar{A}B$

eg: $F(x,y,z) = \sum(3,4,6,7)$



$F = YZ + X\bar{Z} + (\quad)$ redundant

eg: $F(x,y,z) = \sum(0,2,4,5,6)$



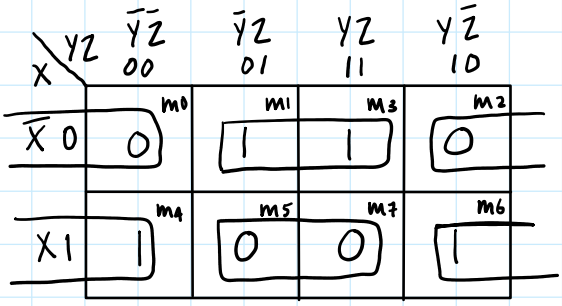
$F = \bar{Z} + X\bar{Y}$

$= SOP$

$POS = (x + y + \bar{z})(x + \bar{y} + \bar{z})$ (M_1, M_3)

$= (x + \bar{z})(\bar{y} + \bar{z})$

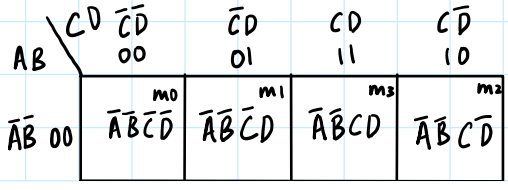
eg: $F(x,y,z) = \Pi(0,2,5,7)$



$F = (X + Z)(\bar{X} + \bar{Z})$

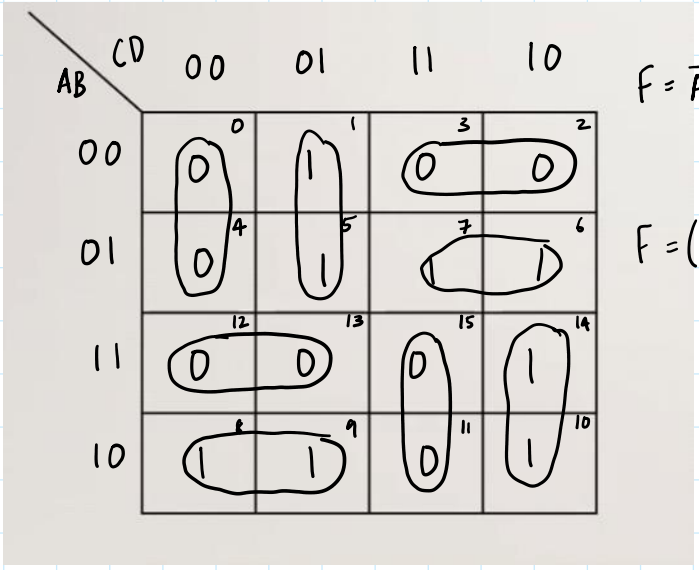
$F = \bar{X}Z + X\bar{Z}$

4 variable K-map $2^4 = 16$ combinations



$\bar{A}\bar{B}$ 00	m_0 $\bar{A}\bar{B}\bar{C}\bar{D}$	m_1 $\bar{A}\bar{B}\bar{C}D$	m_3 $\bar{A}\bar{B}CD$	m_2 $\bar{A}\bar{B}C\bar{D}$
$\bar{A}B$ 01	m_4 $\bar{A}B\bar{C}\bar{D}$	m_5 $\bar{A}B\bar{C}D$	m_7 $\bar{A}BCD$	m_6 $\bar{A}BC\bar{D}$
AB 11	m_{12} $AB\bar{C}\bar{D}$	m_{13} $AB\bar{C}D$	m_{15} $ABCD$	m_{14} $ABC\bar{D}$
$A\bar{B}$ 10	m_8 $A\bar{B}\bar{C}\bar{D}$	m_9 $A\bar{B}\bar{C}D$	m_{11} $A\bar{B}CD$	m_{10} $A\bar{B}C\bar{D}$

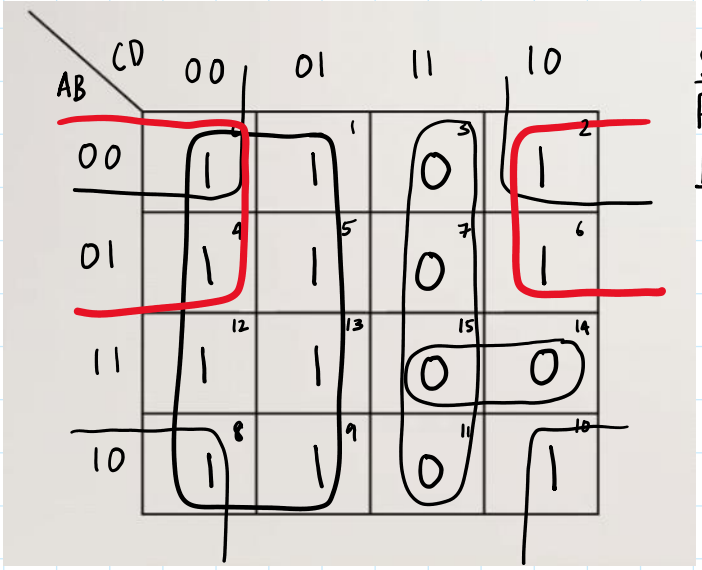
eg: $F(A,B,C,D) = \sum_m(1,5,6,7,8,9,10,14) = \prod_m(0,2,3,4,11,12,13,15)$



$F = \bar{A}\bar{C}D + \bar{A}BC + AC\bar{D} + A\bar{B}\bar{C}$

$F = (A+C+D)(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+C)(\bar{A}+\bar{C}+\bar{D})$

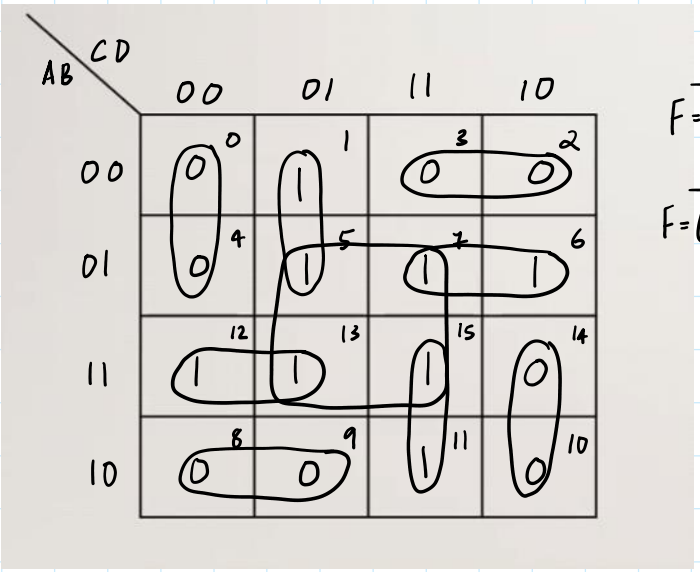
eg: $F(A,B,C,D) = \sum(0,1,2,4,5,6,8,9,10,12,13)$



SOP:
$F = \bar{C} + \bar{B}\bar{D} + \bar{A}\bar{D}$

POS:
$F = (\bar{C}+\bar{D})(\bar{A}+\bar{B}+\bar{C})$

eg: $F(A,B,C,D) = \prod_M (0,2,3,4,8,9,10,14) = \Sigma_1 (1,5,6,7,11,12,13,15)$



SOP:

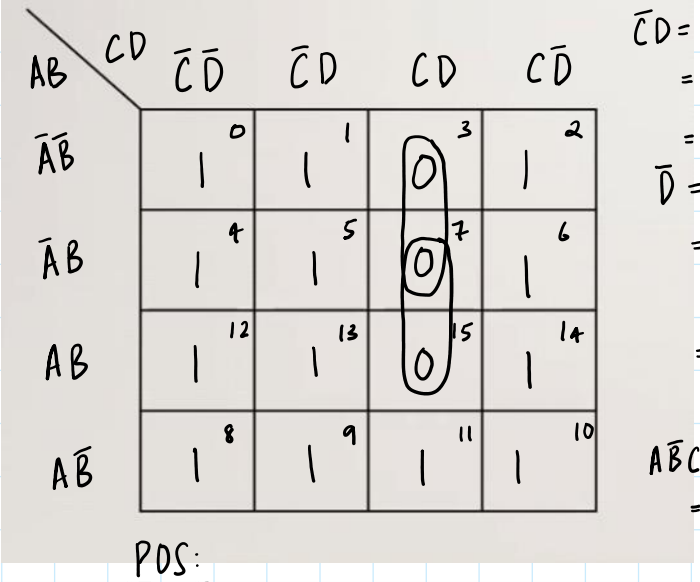
$F = \bar{A}\bar{C}D + \bar{A}BC + ACD + AB\bar{C}$

POS:

$F = (A+C+D)(A+B+\bar{C})(\bar{A}+B+C)(\bar{A}+\bar{C}+D) + (\quad)$ redundant

$(5,7,13,15)$

eg: $F(A,B,C,D) = \bar{C}(\bar{A}\bar{B}\bar{D} + D) + A\bar{B}C + \bar{D} = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{C}D + A\bar{B}C + \bar{D}$



$\bar{C}D = (\bar{A}+A)(\bar{B}+B)\bar{C}D$

$= \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}D + AB\bar{C}D + A\bar{B}\bar{C}D$

$= m_1 + m_5 + m_4 + m_{13}$

$\bar{D} = (A+\bar{A})(B+\bar{B})(C+\bar{C})\bar{D}$

$= ABC\bar{D} + AB\bar{C}D + A\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D}$
$+ \bar{A}BC\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}\bar{D}$

$= m_{14} + m_{13} + m_{10} + m_8 + m_4 + m_6 + m_2 + m_0$

$A\bar{B}C = A\bar{B}C(D+\bar{D}) = A\bar{B}CD + A\bar{B}C\bar{D}$

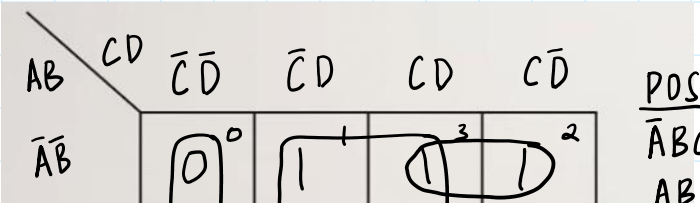
$= m_{11} + m_{10}$

POS:

$F = (A + \bar{C} + \bar{D})(\bar{B} + \bar{C} + \bar{D})$

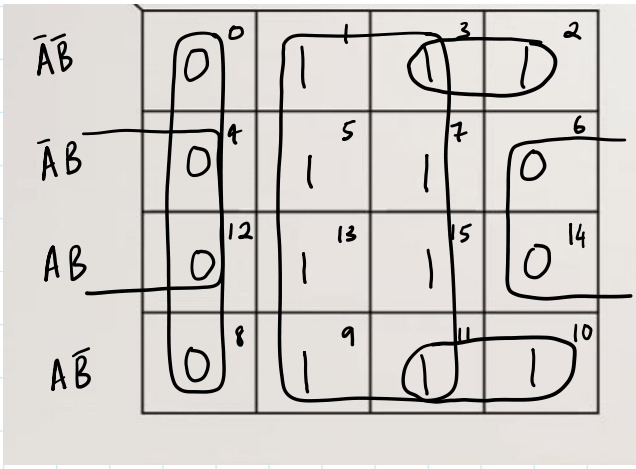
eg: $F(A,B,C,D) = D(\bar{A}+B) + \bar{B}(C+AD)$

$= \bar{A}D + BD + \bar{B}C + AD\bar{B}$



POS:

$\bar{A}BCD + \bar{A}B\bar{C}D + \bar{A}\bar{B}CD + A\bar{B}\bar{C}D +$
$ABCD + AB\bar{C}D + \bar{A}B\bar{C}D + \bar{A}BCD$

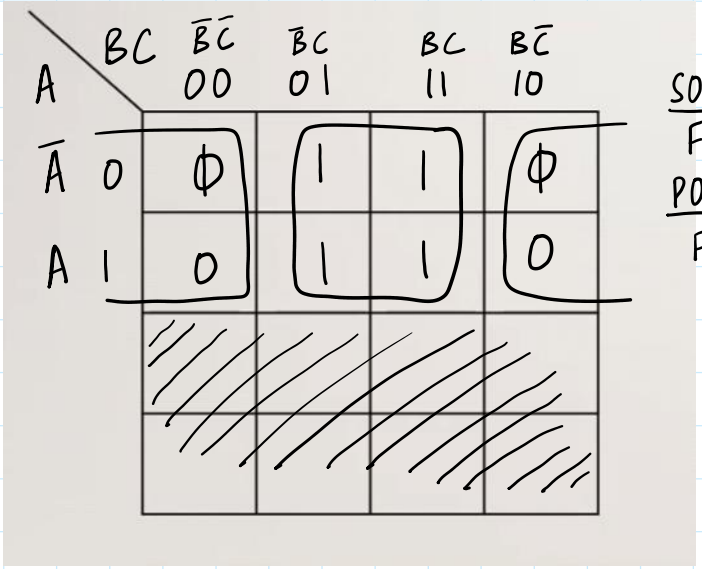


$\bar{A}BCD + \bar{A}B\bar{C}D + \bar{A}\bar{B}CD + A\bar{B}\bar{C}D +$
$ABCD + AB\bar{C}D + \bar{A}B\bar{C}D + \bar{A}BCD$
$A\bar{B}CD + \bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D} + A\bar{B}C\bar{D}$
$A\bar{B}CD + A\bar{B}\bar{C}D$
$= \sum_m (1,2,3,5,7,9,10,11,13,15)$
$= \prod_M (0,4,6,8,12,14)$

POS:
$F = (C+D)(\bar{B}+D)$

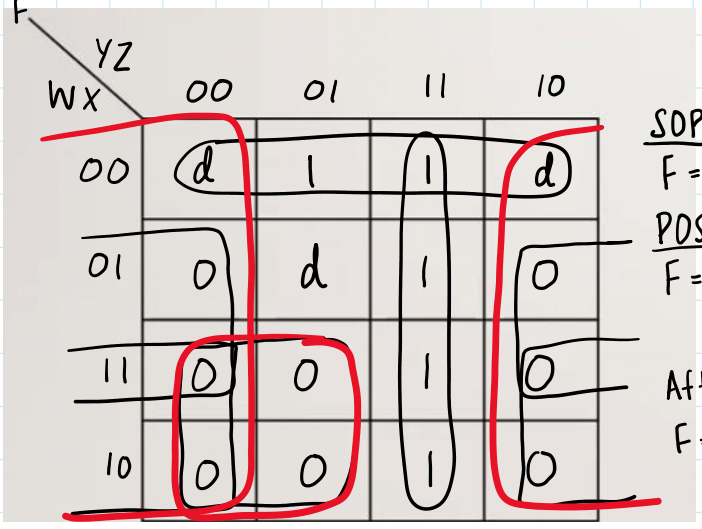
→ DON'T CARE CONDTN:

eg: $F(A,B,C) = \sum_m (1,3,5,7) + \sum_d (0,2)$



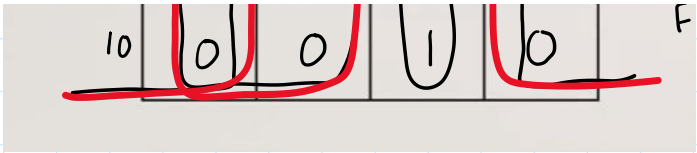
SOP:
$F = C$
POS:
$F = C$

eg: $F(W,X,Y,Z) = \sum_m (1,3,7,11,15) + \sum_d (0,2,5)$



SOP:
$F = \bar{W}\bar{X} + YZ$
POS:
$F = (\bar{W}+\bar{Y})(\bar{W}+Z)(\bar{W}+Z)(\bar{X}+Z)$

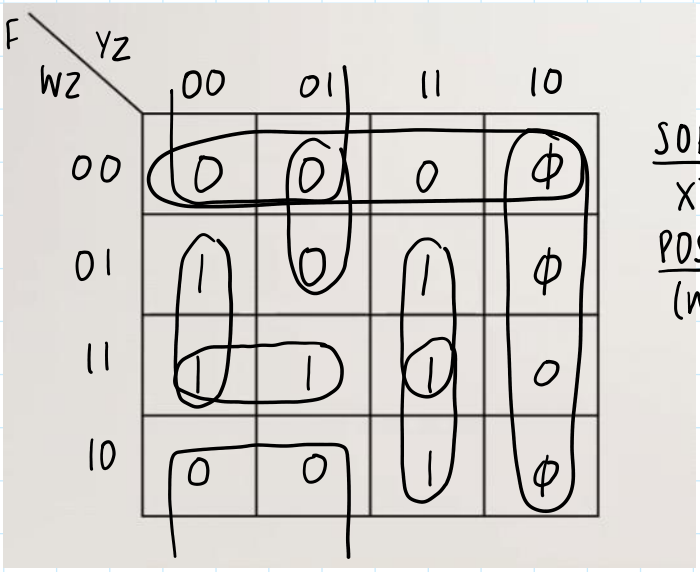
After clubbing don't cares:
$F = Z\ (\bar{W}+Y)$



$F = Z\,(\bar{W} + Y)$

eg: $F(W, X, Y, Z) = \prod_{M}(0, 1, 3, 5, 8, 9, 14) \cdot \prod_{D}(2, 6, 10)$

$= \sum'_{m}(4, 7, 11, 12, 13, 15) + \sum'_{d}(2, 6, 10)$



SOP:

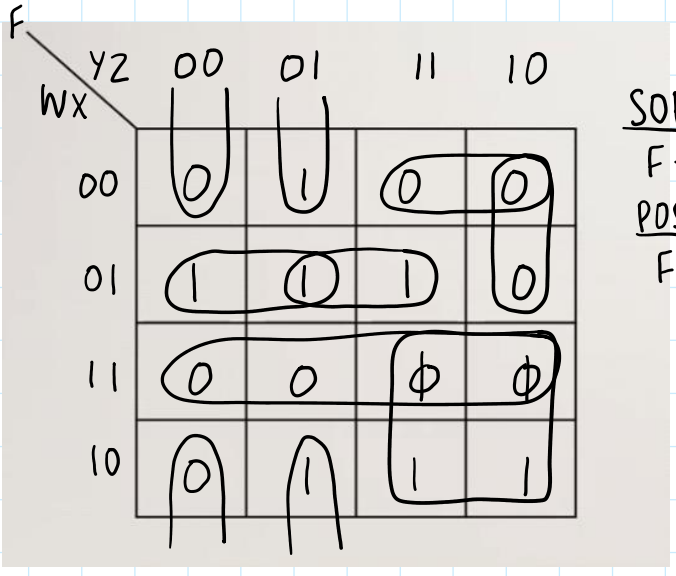
$X\bar{Y}\bar{Z} + WX\bar{Y} + XYZ + WYZ$

POS:

$(W + X)(\bar{Y} + Z)(X + Y)(W + Y + \bar{Z})$

eg: $F(W, X, Y, Z) = \sum'_{1}(1, 4, 5, 7, 9, 10, 11) + D(14, 15)$

$= \prod^{m}_{M}(0, 2, 3, 6, 8, 12, 13) \cdot D(14, 15)$



SOP:

$F = \bar{X}\bar{Y}Z + WY + \bar{W}X\bar{Y} + \bar{W}XZ$

POS:

$F = WXY + XYZ + W\bar{Y}Z + \bar{W}\bar{X}$

eg: Design a combinational circuit to check for even parity of 4 bits. A logic 'I' output is reqd. when the 4 bits constitute an even parity.

Q. Design a combinational circuit to check for even parity of 4 bits.
A logic 'I' output is reqd. when the 4 bits constitute an even parity.

F	YZ = 00	01	11	10
WX = 00	ϕ	0	1	0
01	0	1	0	1
11	1	0	1	0
10	1	0	0	1

For even no. of 1's → Yes → 1, No → 0

W	X	Y	Z	F
0	0	0	0	ϕ
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

$$4\text{ bit} = \bar{W}\bar{X}\bar{Y}\bar{Z} + WX\bar{Y}Z + \bar{W}XY\bar{Z} + WXYZ + WX\bar{Y}\bar{Z} + \bar{W}X\bar{Y}Z + W\bar{X}\bar{Y}Z + W\bar{X}YZ$$

$$= \bar{W}\bar{X}(\bar{Y}Z + Y\bar{Z}) + \bar{W}X(\bar{Y}Z + Y\bar{Z}) + WX(\bar{Y}\bar{Z} + YZ) + W\bar{X}(\bar{Y}Z + Y\bar{Z})$$

$$= \bar{W}\bar{X}(Y \oplus Z) + WX(Y \oplus Z) + \bar{W}X(Y + Z) + W\bar{X}(Y \oplus Z)$$

$$= (W \oplus X)(Y \oplus Z) + (W \oplus X)(Y \oplus Z) = W \oplus X \oplus Y \oplus Z$$

code converters

eg: Design a combinational circuit with 4-input lines that represent a decimal digit in BCD and 4-output lines that generates 2's complement of input digit.

Decimal digit	8 4 2 1 A B C D	2's complement Y_3 Y_2 Y_1 Y_0
0	0 0 0 0	0 0 0 0
1	0 0 0 1	1 1 1 1
2	0 0 1 0	1 1 1 0
3	0 0 1 1	1 1 0 1
4	0 1 0 0	1 1 0 0
5	0 1 0 1	1 0 1 1
6	0 1 1 0	1 0 1 0
7	0 1 1 1	1 0 0 1
8	1 0 0 0	1 0 0 0
9	1 0 0 1	0 1 1 1

Sign bit = 0 = (+) ve
1 = (−) ve

4 bit i/p = 2^4 = 16 input combinations

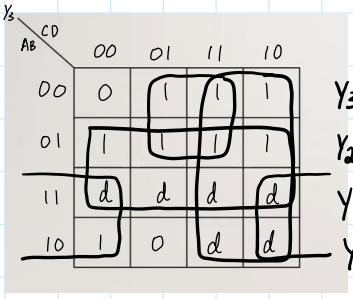
$Y_3 = \sum_m (1,2,3,4,5,6,7,8) + d(10,11,12,13,14,15)$

$Y_2 = \sum_m (1,2,3,4,9) + d(10,11,12,13,14,15)$

$Y_1 = \sum_m (1,2,5,6,9) + d(10,11,12,13,14,15)$

$Y_0 = \sum_m (1,3,5,7,9) + d(10,11,12,13,14,15)$

$Y_3 =$

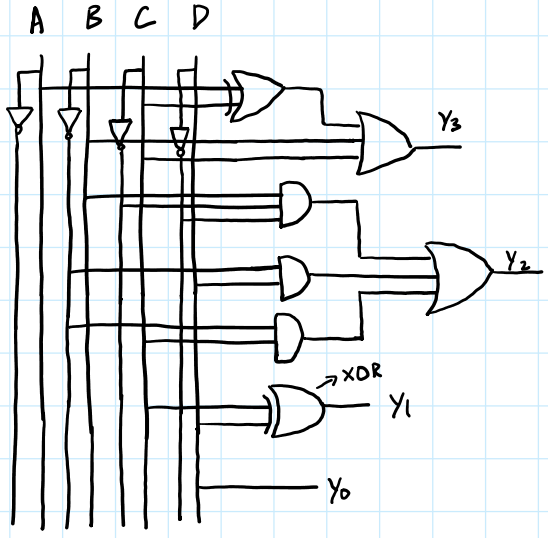


$Y_3 = B + C + \bar{A}D + A\bar{D}$

$Y_2 = B\bar{C}\bar{D} + \bar{B}D + \bar{B}C$

$Y_1 = \bar{C}D + C\bar{D} = C \oplus D$

$Y_0 = D$



→ CODE CONVERTERS:

- Converts coded information in one form to a different coding form.
- Coded representatn for 10 decimal symbols → binary coded decimal
- Min. 4-bits reqd. to represent decimal symbol.
- Only 10 combinations are used to represent 10 decimal symbols and remaining 6 will be don't cares.

Decimal	Binary	BCD (binary representatn of each digit)
0	0000	0000
...	...	...
9	1001	1001
11	1101	0001 0000
12	1100	0001 0001
13	1101	0001 0010

Decimal digit	8 4 2 1 (BCD)	Excess 3	8 4 -2 -1	2 4 2 1	Gray code (by changing only 1 digit starting from LSB.)
0	0000	0011	0000	0000	0000
1	0001	0100	0111	0001	0001
2	0010	0101	0110	0010	0011
3	0011	0110	0101	0011	0010
4	0100	0111	0100	0100	0110
5	0101	1000	1011	1011	0111
6	0110	1001	1010	1100	0101
7	0110	1010	1001	0111	0100
8	1000	1011	1000	1110	1100
9	1100	1100	1111	1111	1101
don't cares	1010, 1011, 1100, 1101, 1110, 1111	0000, 0001, 0010, 1101, 1110, 1111	0001, 0010, 0011, 1100, 1101, 1110	1000, 1001 1010, 0101, 0110, 0111	1110, 1111 1010, 1000, 1011, 1001

→ COMPLEMENTS:

→ COMPLEMENTS:

- **(R-1)'s complement :** $(R^n - 1) - N$

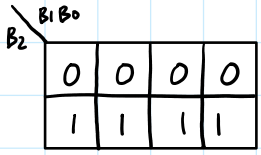
  R → base

  N → no. whos complement is to be taken.

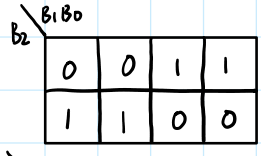
  n → no. of digits /bits in the number N.
- **R's complement :** $R^n - N$

eg: 3-bit binary to gray code converter.

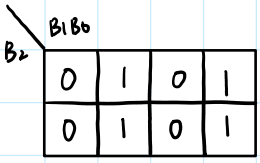
3-bit binary			Gray code		
B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0



$G_2 = B_2$



$G_1 = \overline{B_2}B_1 + \overline{B_1}B_2$
$= B_2 \oplus B_1$

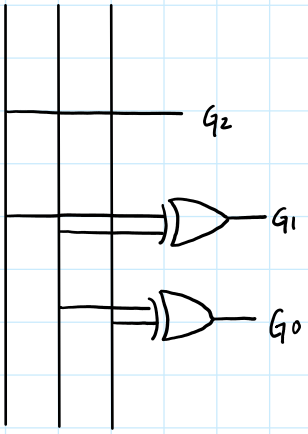


$G_0 = \overline{B_1}B_0 + B_1\overline{B_0}$
$= B_1 \oplus B_0$

(b) $G_2 = \sum m(4,5,6,7)$

$G_1 = \sum m(2,3,4,5)$

$G_0 = \sum m(1,2,5,6)$



eg: Design a code converter to convert a decimal digit represented in 8421 code to a decimal digit represented in Excess code 3.

Decimal digit	8 4 2 1 A B C D	Excess 3 code E3 E2 E1 E0
0	0000	0011
1	0001	0100

$E_3 = \sum m(5,6,7,8,9) + d(10,11,12,13,14,15)$

$E_2 = \sum m(1,2,3,4,9) + d(10,11,12,13,14,15)$

digit	A B C D	E3 E2 E1 E0
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100
Don't cares	1010,1011, 1100,1101,1110, 1111	----------

10/8/2021

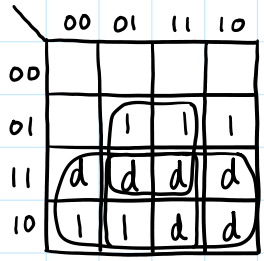
$E3 = \Sigma m(5,6,7,8,9) + d(10,11,12,13,14,15)$

$E2 = \Sigma m(1,2,3,4,9) + d(10,11,12,13,14,15)$

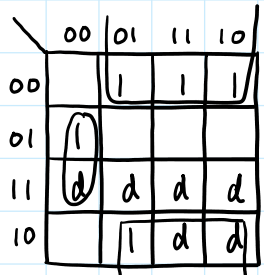
$E1 = \Sigma m(0,3,4,7,8) + d(10,11,12,13,14,15)$

$E0 = \bar{D}$ (from truth table)

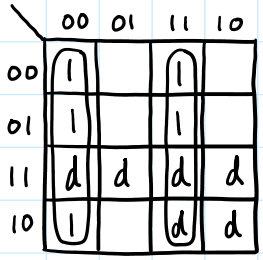
$E3$:



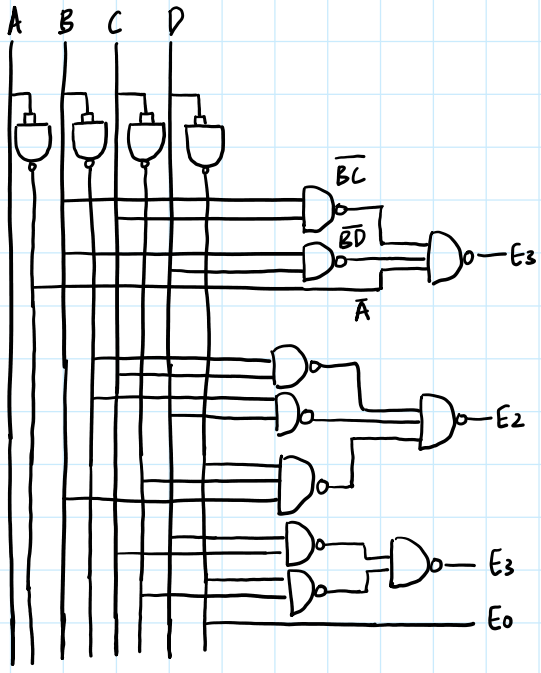
$E3 = A + BC + B\bar{D}$



$E2 = \bar{B}\bar{C} + \bar{B}\bar{D} + B\bar{C}\bar{D}$



$E1 = \bar{C}\bar{D} + C\bar{D}$



$\overline{E_3} = \overline{A + BC + BD} = \bar{A} \cdot \overline{BC} \cdot \overline{BD}$

$\overline{\overline{E_3}} = \overline{\bar{A} \cdot \overline{BC} \cdot \overline{BD}}$

eg: 8421 → 8-4-21

Decimal digit	8 4 2 1 A B C D	8-4-2-1
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 1 1 1
2	0 0 1 0	0 1 1 0
3	0 0 1 1	0 1 0 1
4	0 1 0 0	0 1 0 0
5	0 1 0 1	1 0 1 1
6	0 1 1 0	1 0 1 0
7	0 1 1 1	1 0 0 1
8	1 0 0 0	1 0 0 0
9	1 0 0 1	1 1 1 1

$Y_3 = \Pi M(0,1,2,3,4) \cdot d(10,11,12,13,14,15)$

$Y_2 = \Pi M(0,5,6,7,8) \cdot d(10,11,12,13,14,15)$

$Y_1 = \Pi M(0,3,4,7,8) \cdot d(10,11,12,13,14,15)$

8	1 0 0 0	1 0 0 0
9	1 0 0 1	1 1 1 1
Don't cares:	10, 11, 12 13, 14, 15	

Q: Excess-3 code ⟶ 8 4-2 -1

Decimal digit	EXCESS -3 CODE A B C D	8 4 -2 -1 Y3 Y2 Y1 Y0
0	0011	0000
1	0100	0111
2	0101	0110
3	0110	0101
4	0111	0100
5	1000	1011
6	1001	1010
7	1010	1001
8	1011	1000
9	1100	1111
Don't cares	0000,0001,0010, 1101,1110,1111	----------

Q: 8 4-2-1 ⟶ 2421

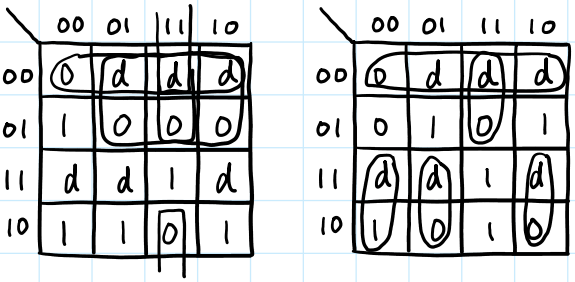
Decimal digit	8 4 -2 -1 A B C D	2 4 2 1 Y3 Y2 Y1 Y0
0	0000	0000
1	0111	0001
2	0110	0010
3	0101	0011
4	0100	0100
5	1011	1011
6	1010	1100
7	1001	1101
8	1000	1110
9	1111	1111
Don't cares	0001,0010,0011, 1100,1101,1110	_____

$Y_3 = \sum m\,(8,9,10,11,15) + d\,(1,2,3,12,13,14) = A$

$Y_2 = \sum m\,(4,8,9,10,15) + d\,(1,2,3,12,13,14)$

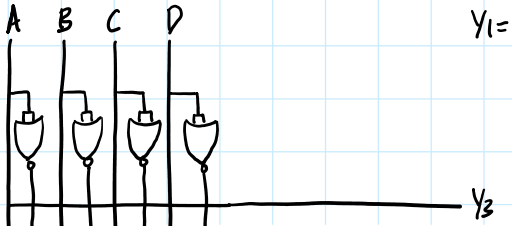
$Y_1 = \sum m\,(5,6,8,11,15) + d\,(1,2,3,12,13,14)$

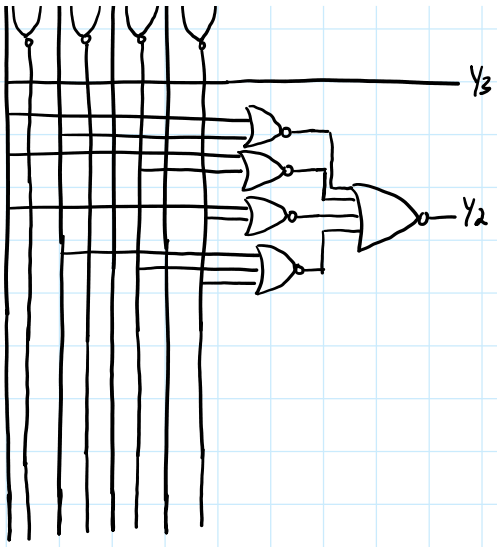
$Y_0 = D$



$Y_2 = (A+B)(A+\bar{C})(A+\bar{D})(B+\bar{C}+\bar{D})$

$Y_1 = (A+B)(\bar{A}+C+D)(\bar{A}+C+\bar{D})(A+\bar{C}+\bar{D})$





Q: 2421 → gray code

Decimal digit	2 4 2 1 A B C D	Gray codes $G_3 G_2 G_1 G_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	1 0 1 1	0 1 1 1
6	1 1 0 0	0 1 0 1
7	1 1 1 0	0 1 0 0
8	1 1 1 0	1 1 0 0
9	1 1 1 1	1 1 0 1
Don't cares	0101, 0110, 0111	
	1000, 1001, 1010	

$G_3 = \sum m(14,15) + d(5,6,7,8,9,10)$

$G_2 = \sum m(4,11,12,13,14,15) + d(5,6,7,8,9,10)$

$G_1 = \sum m(2,3,4,11) + d(5,6,7,8,9,10)$

$G_0 = \sum m(1,2,11,12,15) + d(5,6,7,8,9,10)$

Q: Design a combinational circuit that multiplies by '5'

adders and subtracters

→ ADDER:

```
  1  1  0
  1  0  1  0  +
  0  1  1  1
-----------------
 10  0  0  1
 ↓
Final carry
```

→ HALF ADDER: (HA)

- Adds 2, 1-bit numbers A and B, generated two outputs sum (S) and carry (C)

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$Sum = \sum_m 1,2 = \prod_M 0,3$

$Cout = \sum_m 3 = \prod_M 0,1,2$

Sum

A \ B	0	1
0	0	1
1	1	0

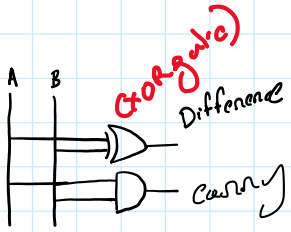
Cout

A \ B	0	1
0	0	0
1	0	1

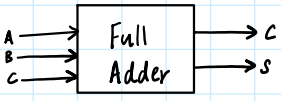
$S = A\bar{B} + \bar{A}B$

$S = A \oplus B$

$Cout = AB$



→ FULL ADDER:



A	B	C	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$Carry = \sum_m 3,5,6,7 = \prod_M 0,1,2,4$

$Sum = \sum 1,2,4,7 = \prod_M 0,3,5,6$

Sum

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

$Sum = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + ABC$
$= \bar{A}(B \oplus C) + A(\bar{B}C + BC)$
$= A \oplus B \oplus C$

Carry

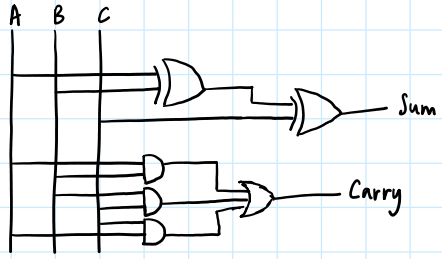
A \ BC	00	01	11	10
0	0	1	0	0
1	0	1	1	1

$Carry = AB + BC + AC$

- Full adder circuit XOR operations:

$Sum = A \oplus B \oplus C$

$Carry = AB + AC + BC$

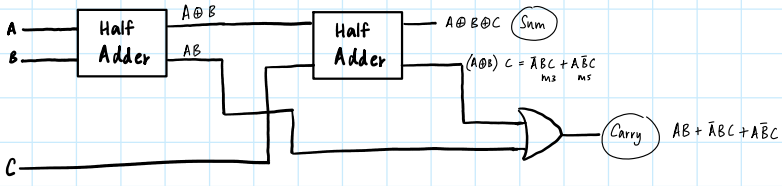


- Full adder circuit using 2 HAs & one external gate:

Half adder:

$S = A \oplus B$

$C = A.B$



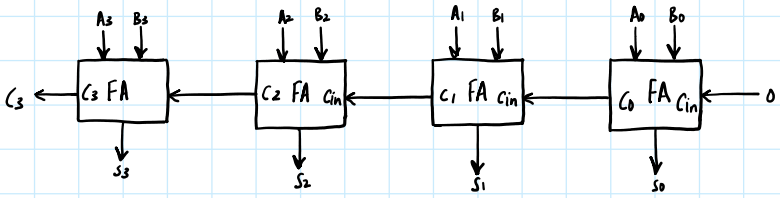
→ 4-BIT PARALLEL ADDER:

- Consider addition of 2, 4-bit numbers: $(A_3\ A_2\ A_1\ A_0)$ and $(B_3\ B_2\ B_1\ B_0)$

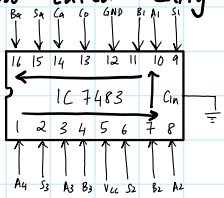
FA FA FA ← carry propagated

C_3	C_2	C_1	C_0	
↓	A_3	A_2	A_1	A_0
	B_3	B_2	B_1	B_0

OR A (4-bit number)
B (4-bit number)



- Also called Carry Propagat^n Adder (CPA)



- 16 pin dual line-in package
- Top side → notch
- Groove → Pin Numbering starts.
- Vcc → pin 5 , Ground → Pin 12.

→ HALF SUBTRACTOR:

A	B	D (Difference)	B (Borrow)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Diff. = $\sum_m 1,2 = \prod_M 0,3$

Borrow = $\sum_m 1 = \prod_M 0,2,3$

Diff.

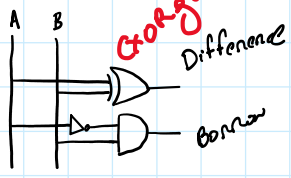
A \ B	0	1
0	0	1
1	1	0

Diff. = $A \oplus B$

Borrow

A \ B	0	1
0	0	1
1	0	0

Borrow = $\bar{A}B$



Assume you need to Subtract

A − B (if it is binary subtraction)

A − B

where A → 1 bit number 0, 1

B → 1 bit number 0, 1

0 − 0 = 0

0 − 1 = 1, get a borrow 1

1 − 0 = 1

1 − 1 = 0

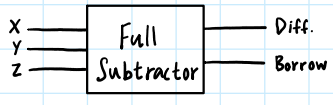
D − 0 | 1 | 1 | 0

B − 0 (no borrow) | 1 | 0 | 0

→ FULL SUBTRACTOR:

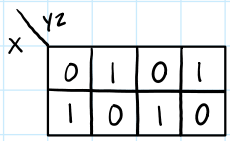
- Difference (D) = X-Y-Z , Borrow (B)

X	Y	Z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	1	1	1

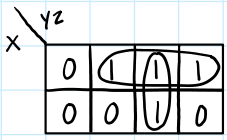


D = $\sum_m 1,2,4,7 = \prod_M 0,3,5,6$

B = $\sum_m 1,2,3,7 = \prod_M 0,4,5,6$



$D = X \oplus Y \oplus Z$

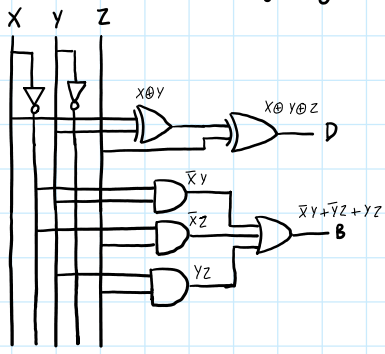


$B = \bar{X}Z + \bar{X}Y + YZ$

- **FS circuit:**

(i) basic logic gates only

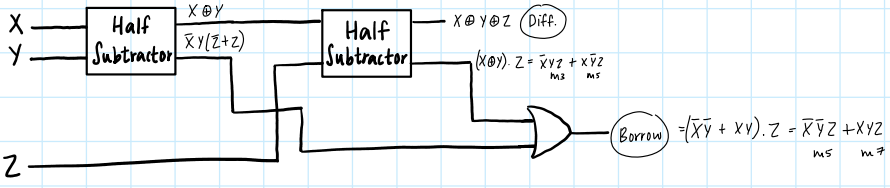
(ii) XOR and basic logic gates



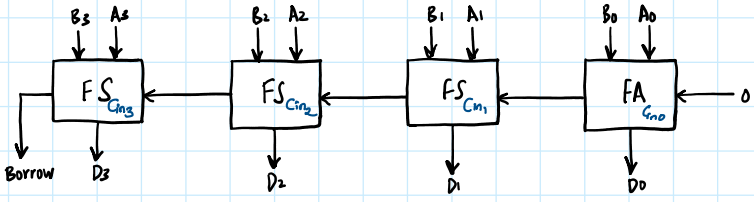
- **Full subtractor using 2 HS and one external gate:**

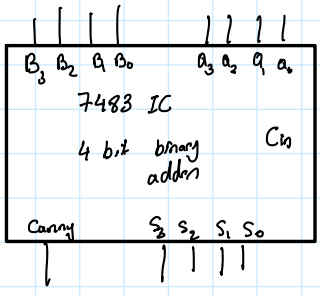
HS:	FS:
Diff: $X \oplus Y$	$D = X \oplus Y \oplus Z$
Borrow: $\bar{X}Y$	$B = \bar{X}Y + \bar{X}Z + YZ$

$\bar{X}Y\bar{Z} + \bar{X}YZ + \bar{X}\bar{Y}Z + X\bar{Y}Z + XYZ + X\bar{Y}\bar{Z}$



→ 4-BIT PARALLEL SUBTRACTOR:





→ COMPLEMENTS: (subtraction)

(i) Using 2's complement:

eg: 8 → 1000 → 1000

−2 → −0010 → 1101 (2's complement) } 1's complement + 1

+1

6 — 1010

6

2's complement: +1] 1's complement +1

1 0 1 1 0

Retain sum terms: diff → 6

1 means (+)ve

0 means (−)ve

eg: 2 → 0 0 1 0 → 0 0 1 0

−8 → −1 0 0 0 → 0 1 1 1 } 1's complement +1

6

2's complement: +1

0 1 0 1 0

Retain sum terms: diff → 6

0 means (−)ve

(ii) Using 1's complement:

eg: 8 → 1 0 0 0 → 1 0 0 0 → 0 1 0 1

−2 → −0 0 1 0 → +1 1 0 1 +1

6

1's complement: 1 0 1 0 1 0 1 1 0

(+)ve

eg: 2 → 0 0 1 0 → 0 0 1 0

−8 → −1 0 0 0 → +0 1 1 1

−6

1's complement: 0 1 0 0 1 → answer : complement = 0110 = −6

(−)ve

Q: Design a 4-bit adder/subtractor using FA blocks or 7843 IC and minimum external gates, i.e. if the control input bit K=0, the circuit should add the input numbers if K=1, the circuit should subtract the 2 numbers using 2's complement method.

Sol: K=0 A+B

$A_3 A_2 A_1 A_0$

$+ B_3 B_2 B_1 B_0$

K=1 A−B $A_3 A_2 A_1 A_0$ → $A_3 A_2 A_1 A_0$

$B_3 B_2 B_1 B_0$ → $+ \overline{B_3}\ \overline{B_2}\ \overline{B_1}\ \overline{B_0}$ (1's comp.)

K	Input B_3	Output
0	0	0
0	1	1
1	0	1
1	1	0

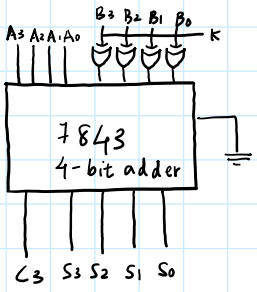
Output = $\overline{K} B_3 + K \overline{B_3}$

Ex-Or gate

$A_3 A_2 A_1 A_0$ $B_3 B_2 B_1 B_0$ K

| 1 | 1 | 0 |

Ex-Or gate



Control input

if K=0, B

K=1, 1's comp. B.

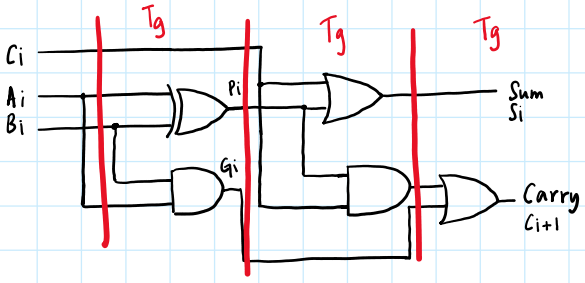
If K=0, $C_3=0 \rightarrow C_3S_2S_1S_0$

If K=1, $C_3=0 \rightarrow$ Ans(-)ve 1's complement of $S_3S_2S_1S_0$.

$C_3=1 \rightarrow$ Ans(+)ve $S_3S_2S_1S_0$

→ CARRY LOOK AHEAD (CLA) ADDER:

- Propagatⁿ delay in CPA is $3T_g$ with respect to foll. circuit, where $T_g \rightarrow$ propagatⁿ delay of a gate.
- All gates are assumed to have propagatⁿ delay of T_g.
- $P_i \rightarrow$ carry propagate term and $G_i \rightarrow$ carry generate term.



$P_i = A_i \oplus B_i$ (input with EX-OR)

$G_i = A_i . B_i$ (responsible for generating carry term using AND gate)

$S_i = A_i \oplus B_i \oplus C_i$

$C_{i+1} = A_iB_i + (A_i \oplus B_i).C_i$

- Carry generatⁿ in CLA from A_i, B_i & C_0.

$C_0 \rightarrow$ input carry

$C_1 = G_0 + P_0C_0 = A_0B_0 + (A_0 \oplus B_0).C_0$

$C_2 = G_1 + P_1C_1 = G_1 + P_1[G_0 + P_0C_0] = G_1 + P_1G_0 + P_1P_0C_0$

$= A_1B_1 + (A_1 \oplus B_1)A_0B_0 + (A_1 \oplus B_1)(A_0 \oplus B_0)C_0$

$C_3 = G_2 + P_2C_2 = G_2 + P_2[G_1 + P_1(G_0 + P_0C_0)]$

$= G_2 + P_2G_1 + P_2P_1G_0 + P_0C_0P_2$

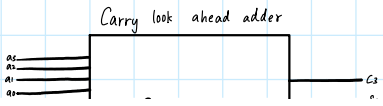
- Expression for sum:

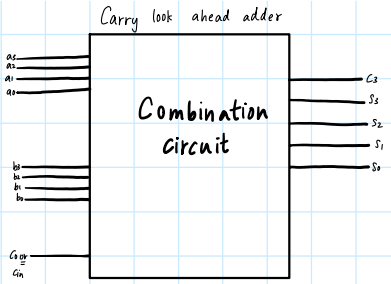
$S_0 = A_0 \oplus B_0 \oplus C_0$

$S_1 = A_1 \oplus B_1 \oplus C_1 = A_1 \oplus B_1 \oplus (G_0 + P_0C_0)$

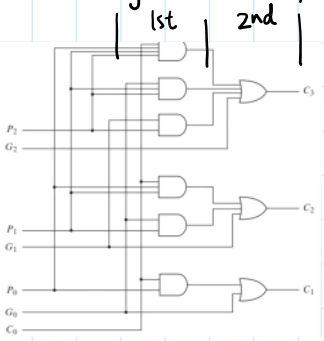
$S_2 = A_2 \oplus B_2 \oplus C_2 = A_2 \oplus B_2 \oplus (G_1 + P_1C_1)$

Q: Draw the combinational ckt. to generate C_1, C_2 and C_3 from P_i, G_i and C_0 terms.



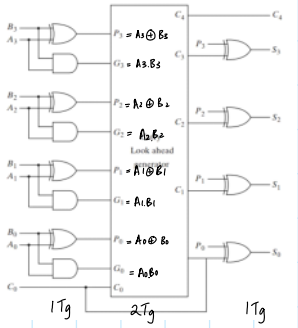


Q: Time to generate C_1, C_2, C_3 in terms of T_g?



→ $2T_g$

Q: Time to generate S_1, S_2, S_3, C_4 in terms of T_g?



$4T_g$

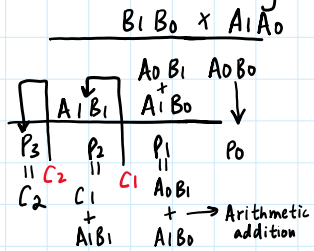
→ DECIMAL ADDER:

- Used to add decimal numbers represented in binary coded form.

multipliers and magnitude comparators

→ MULTIPLIERS:

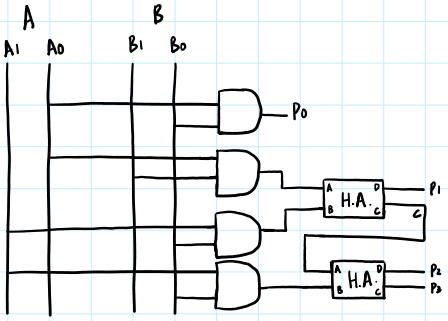
- 2 bit x 2 bit binary multiplier using adders and ext. gates.



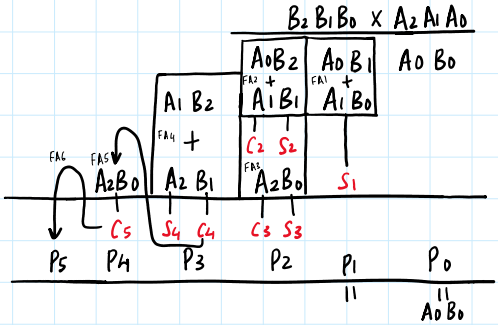
eg: 10×11

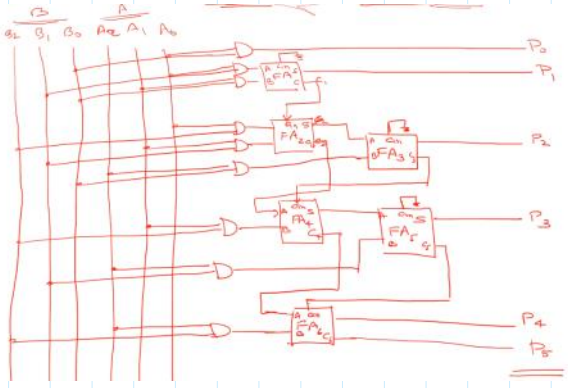
1×1 0×1 0×1 1×1

a0	b0	Product
0	0	0
0	1	0
1	0	0
1	1	1



- Design a 3bit x 3 bit binary multipliers using Full adders and ext. AND gates.

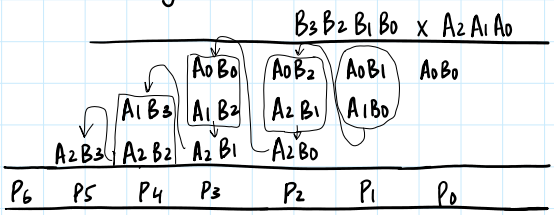


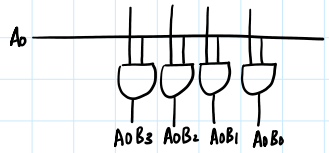


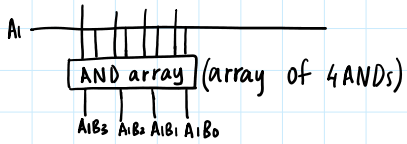
Use 7483 4-bit binary adder IC & external AND gates.
to realize 3X3 bit multiplication.

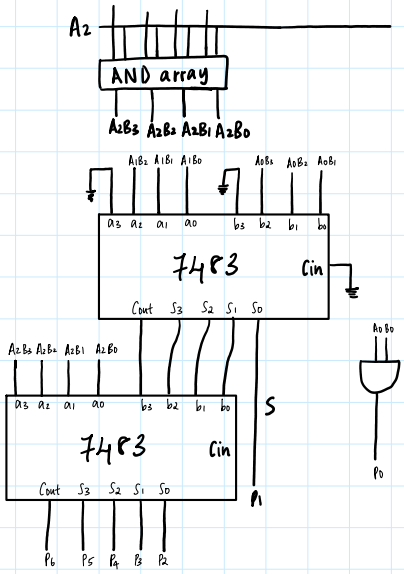
- Design 4 bit x 3 bit binary multipliers using 7483 ICs (4 bit binary adders) and ext. gates.

- Design 4 bit x 3 bit binary multipliers using 7483 ICs (4 bit binary adders) and ext. gates.



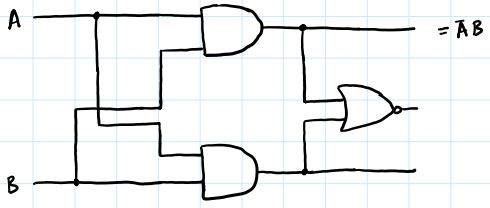






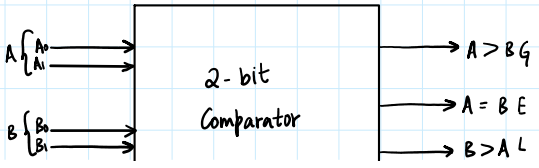
→ MAGNITUDE COMPARATOR:

Input		Output		
A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

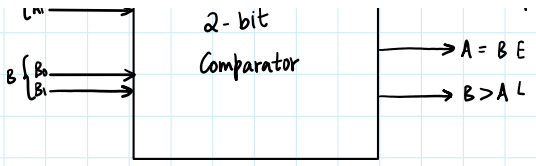


- 2 bit magnitude comparator:

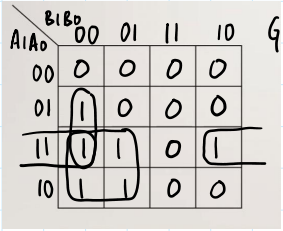
	A0	A1	B0	B1	A>B	A=B	A<B
m0	0	0	0	0	0	1	0
m1	0	0	0	1	0	0	1
m2	0	0	1	0	0	0	1
m3	0	0	1	1	0	0	1
m4	0	1	0	0	1	0	0



m_2	0	0	1	0	0	0	1
m_3	0	0	1	1	0	0	1
m_4	0	1	0	0	1	0	0
m_5	0	1	0	1	0	1	0
m_6	0	1	1	0	0	0	1
m_7	0	1	1	1	0	0	1
m_8	1	0	0	0	1	0	0
m_9	1	0	0	1	1	0	0
m_{10}	1	0	1	0	0	1	0
m_{11}	1	0	1	1	0	0	1
m_{12}	1	1	0	0	1	0	0
m_{13}	1	1	0	1	1	0	0
m_{14}	1	1	1	0	1	0	0
m_{15}	1	1	1	1	0	1	0



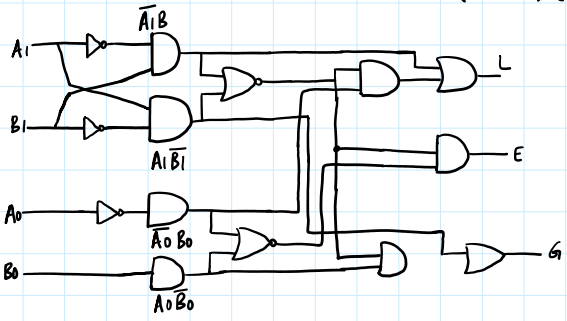
$G = A>B = \sum_m 4,8,9,12,13,14 = (A_1 > B_1) + (A_1 = B_1)(A_0 > B_0)$



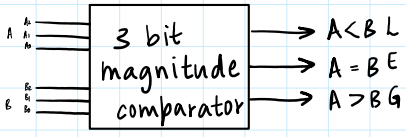
$G = A_1\overline{B_1} + A_0\overline{B_1}\,\overline{B_0} + A_1A_0\overline{B_0}$
$= A_1\overline{B_1} + A_0\overline{B_0}\,[A_1 + \overline{B_1}]$
$= A_1\overline{B_1} + A_0\overline{B_0}\,(\overline{A_1}\overline{B_1} + A_1B_1)$
$= A_1\overline{B_1} + A_0\overline{B_0}\,(A_1 \oplus B_1)$

$E = A=B = \sum_m 0,5,10,15 = (A_1 = B_1).(A_0 = B_0)$
$= (\overline{A_1 \oplus B_1}).(\overline{A_0 \oplus B_0})$

$L = A<B = \sum_m 1,2,3,6,7,11 = (A_1 < B_1) + (A_1 = B_1)(A_0 < B_0)$
$= A_1\overline{B_1} + (A_1 \odot B_1)(A_0\overline{B_0})$



- **3 bit** magnitude using 1-bit magnitude comparator:

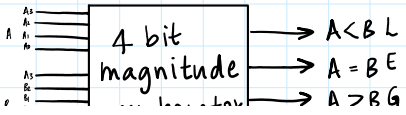


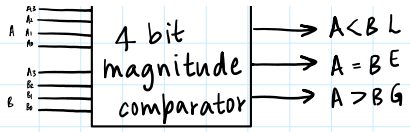
G or $A>B = G_1 + E_2G_1 + E_2E_1G_0 = A_2\overline{B_2} + (A_2 \odot B_2)A_1\overline{B_1} + (A_2 \odot B_2)(A_1 \odot B_1)\overline{A_0}.B_0$
$= (A_2 > B_2) + (A_2 = B_2)(A_1 > B_1) + (A_2 = B_2)(A_1 = B_1)(A_0 > B_0)$

E or $A=B = E_2.E_1.E_0 = (A_2 \odot B_2).(A_1 \odot B_1).(A_0 \odot B_0)$
$= (A_2 = B_2)(A_1 = B_1)(A_0 = B_0)$

L or $A<B = L_2 + E_2L_1 + E_2E_1L_0 = \overline{A_2}B_1 + (A_2 \odot B_2).\overline{A_1}B_1 + (\overline{A_2 \oplus B_2})(\overline{A_1 \oplus B_1})\overline{A_0}B_0$
$= (A_2 < B_2) + (A_2 = B_2)(A_1 < B_1) + (A_2 = B_2)(A_1 = B_1)(A_0 < B_0)$

- **4- bit magnitude comparator:**



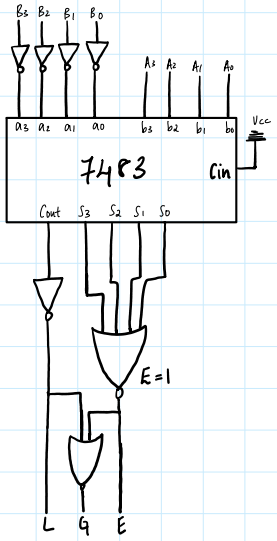


$G = A > B = G_3 + E_3G_2 + E_3E_2G_1 + E_3E_2E_1G_0$

$E = A = B \Rightarrow E_3 . E_2 . E_1 . E_0$

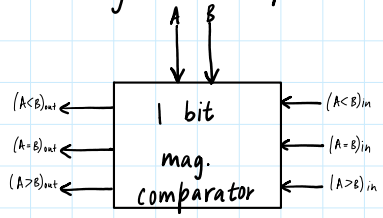
$L = A < B \Rightarrow L_3 + E_3L_2 + E_2E_1L_1 + E_3E_2E_1L_0$

- 4 bit binary adder using 7483 IC & external gates:

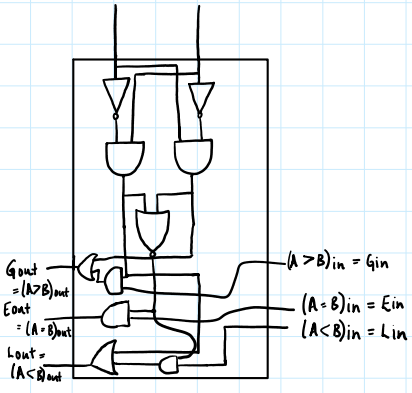


A → A
−B +2's comp B

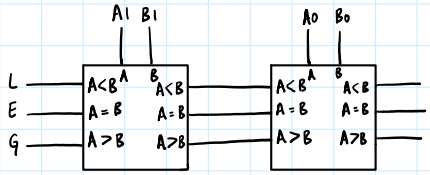
- 1bit magnitude comparator with cascading input:



$(A>B)_{out} = (A>B) + (A=B)(A>B)_{in}$
$(A<B)_{out} = (A<B) + (A=B)(A<B)_{in}$
$(A=B)_{out} = (A=B)(A=B)_{in}$



- 2 bit:

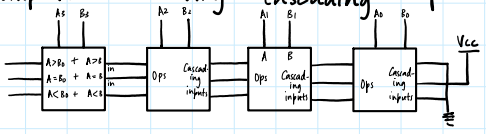


$A>B = (A_1>B_1) + (A_1=B_1).(A_0>B_0) \Rightarrow G = G_1 + E_1G_0$

$A<B = (A_1<B_1) + (A_1=B_1).(A_0<B_0) \Rightarrow L = L_1 + E_1L_0$

$(A=B) = (A_1=B_1).(A_2=B_2) \Rightarrow E = E_1.E_0$

- Design a 4 bit magnitude comparators using 1 bit magnitude comparator having cascading inputs.

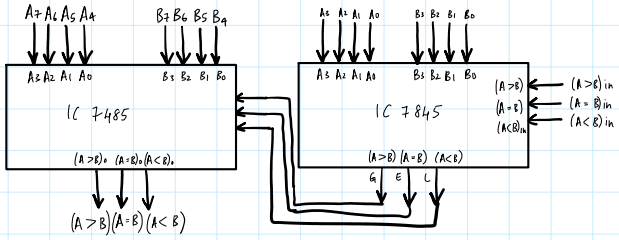


$A>B = G$

$A=B = E$

$A<B = L$

Q: Design 8-bit mag. comparators using 7485 ICs.



Q: Design a combinational circuit using 7483 IC and ext. gates to perform the arithmetic operations as shown below:

$F = 2X+Y$ when $m=0$ & $F = X+2Y$ when $m=1$

When X, Y are 2 bit numbers & output F is a 4 bit number.

Control pin: m if $m=0$ $F = 0.X + 1.Y$

$m=1$ $F = 1.X + 0.Y$

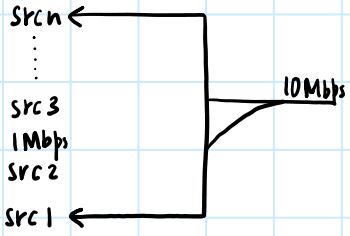
$10 \times X + 01 \times Y$

$01 \times X + 10 \times Y$

Multiplexers

Monday, November 1, 2021 11:22 PM

multiplexers



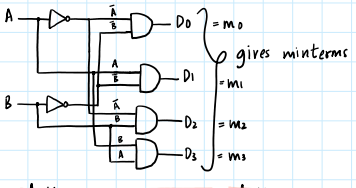
- Multiplexer is an nseful MSI device and are also called data selectors.

decoders and encoders

→ DECODER : (only one output is active at a time)

- Combinational ckt:
- Converts binary info. from n-input lines to a maximum of 2^n unique output and one or more enable inputs. eg: 2-to-4 line, 3-to-8 line etc.
- In standard decoders, only one output line will be active at a time corresponding to the input binary combinat^n.

→ 2-4-LINE DECODER:



Active HIGH output :

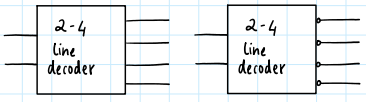
$B^{i/p}$	A	D_3	$D_2^{o/p}$	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0
		m_3	m_2	m_1	m_0

AND gates

Active LOW output :

$B^{i/p}$	A	D_3	$D_2^{o/p}$	D_1	D_0
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1
		m_3	m_2	m_1	m_0

NAND gates



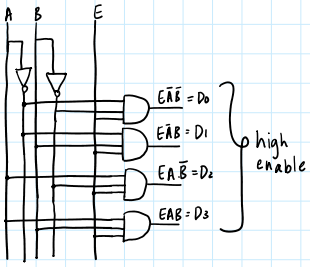
Active high

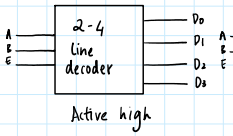
Active low

- 2-4-decoder- with enable control:

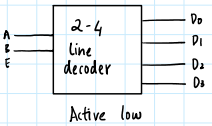
Active HIGH enable :

E	A	B	D_3	D_2	D_1	D_0
0	0	0	X	X	X	X
0	0	1	X	X	X	X
0	1	0	X	X	X	X
0	1	1	X	X	X	X
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0





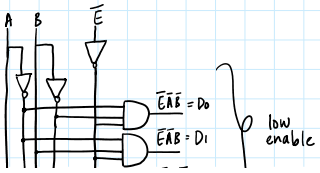
Active high



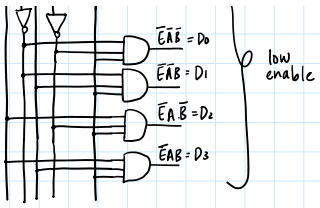
Active low

Active LOW enable:

E	A	B	D_3	D_2	D_1	D_0
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	1	0	0
0	1	1	1	0	0	0



0	0	1	0	0	1	0
0	1	0	0	1	0	0
0	1	1	1	0	0	0
1	0	0	X	X	X	X
1	0	1	X	X	X	X
1	1	0	X	X	X	X
1	1	1	X	X	X	X

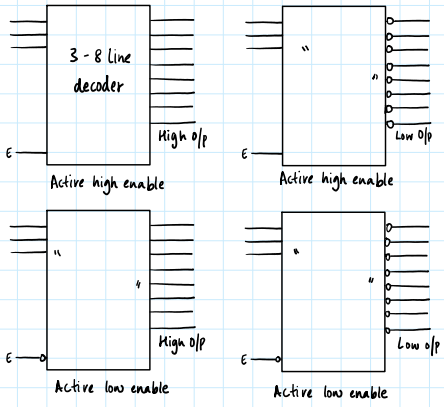


- Write truth table, logic diagram and block diagram of 3-to-8 line decoder.

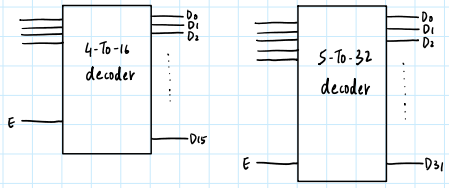
Inputs				Outputs:							
E	A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
1	X	X	X	*	*	*	*	*	*	*	*
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0

$D_0 = \overline{\bar{E}\bar{A}\bar{B}\bar{C}}$
$D_1 = \overline{\bar{E}\bar{A}\bar{B}C}$
$D_2 = \overline{\bar{E}\bar{A}B\bar{C}}$
$D_3 = \overline{\bar{E}AB\bar{C}}$
$D_4 = \overline{\bar{E}A\bar{B}\bar{C}}$

- NOTE: Assume o/p and enable i/p to be active low.

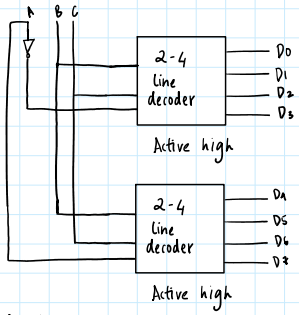


4-TO-16 AND 5-TO-32:

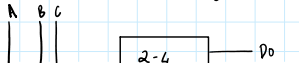


Q. Design 3-to-8 decoder using min. no. of:
(i) 2-to-4 decoders with enable i/p and one ext. gate.
(ii) 2-to-4 decoders with enable i/ps only.

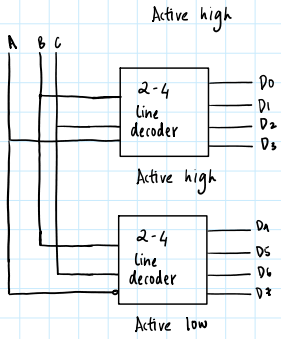
(i)



(ii)



(ii)

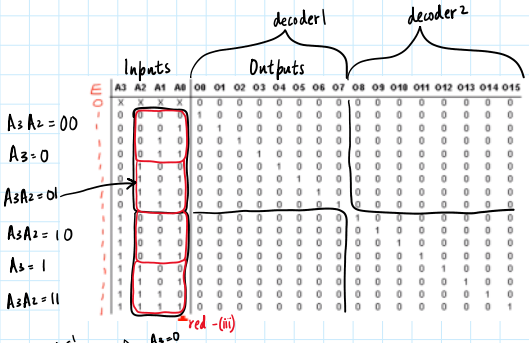


Q: 4 to 16 decoder with :

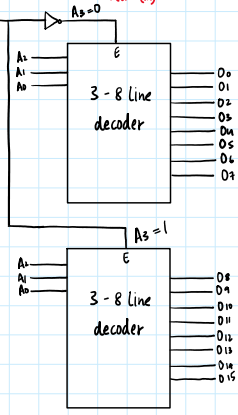
(i) 3-to 8 decoder

(ii) only 3-to-8 and 2-to 4

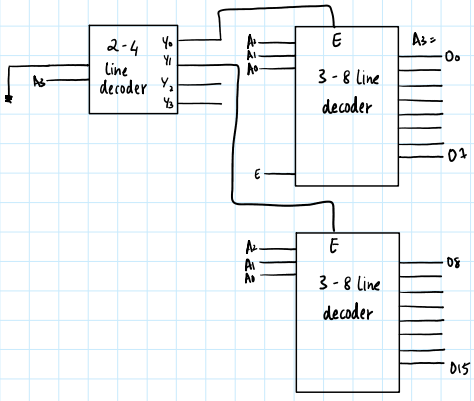
(iii) only 2-to-4 decoders



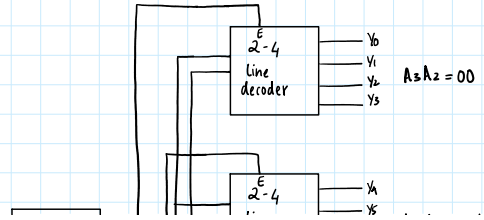
(i)

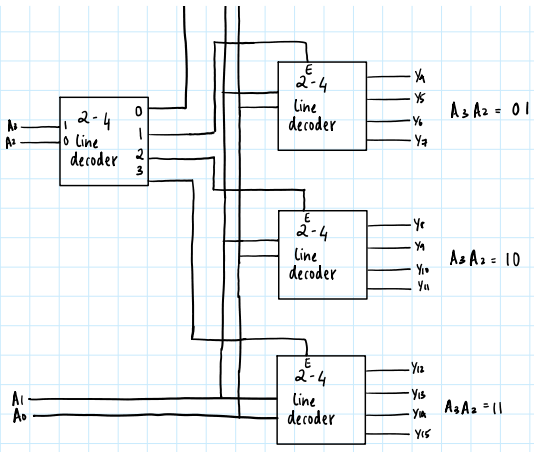


(ii)

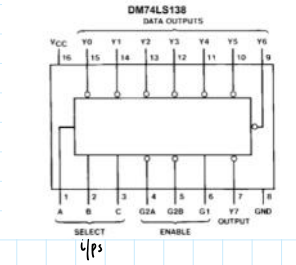


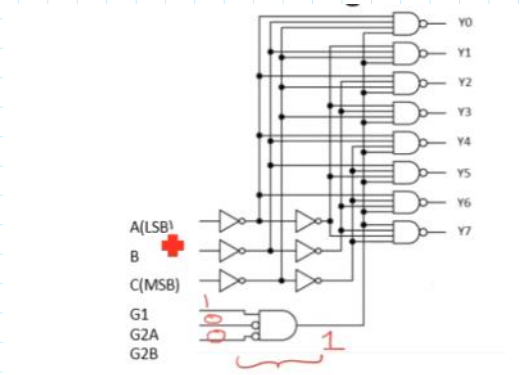
(iii)





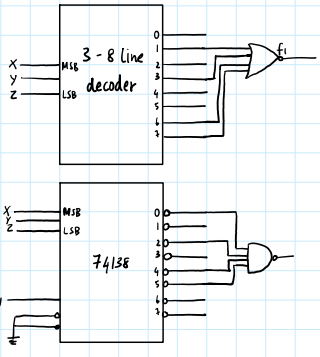
* 74183 IC : 3-to-8 line decoder with active low output:





*

eg: $f1(x,y,z) = \Pi M(1,3,6,7)$ using
(a) 3-to-8 line decoder with active high outputs and suitable gates.
(b) 74138 decoder and suitable gates



eg: Design a code converter to convert a decimal digit represented in 84-2-1 code to a decimal digit represented in excess-3 code using 74183 decoder and ext. gates

8-4-2-1				excess-3		
A	B	C	D	e3	e2	e1	e0
0	0	0	0	0	0	0	0

$e3 = A$
$e3 = \Sigma m(0,1,2,3,4)$
$e2 = \Sigma m(0,5,6,7,8) = B$

eg: Implement $f(A,B,C) = \Sigma m(0,2,4,6,9,10)$ using suitable decoder and external gates

A B C D	e_3	e_2	e_1	e_0
0 0 0 0	0	0	0	0
0 1 1 1	0	1	0	0
0 1 1 0	1	0	0	1
0 1 0 1	1	0	0	0

$e_3 = A$

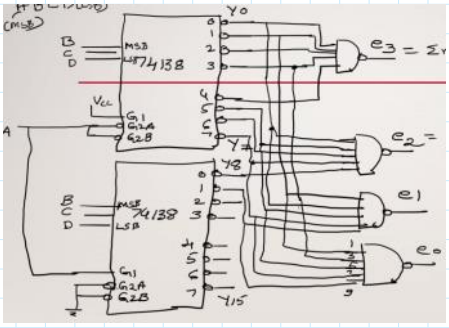
$e_3 = \Sigma m\,(0,1,2,3,4)$

$e_2 = \Sigma m\,(0,5,6,7,8) = B$

$e_1 = \Sigma m\,(1,2,5,6,9) = \bar{C}$

$e_0 = \Sigma m\,(1,3,5,7,9) = \bar{D}$

$e_3 = (A,B,C,D)$



eg: $f(x,y,z) = x' + y'z$ using 3-to-8 line decoder and ext. gates.

Minterms for $x' = (x = 0 \text{ or } 1,\ y = z = \text{don't care})$

$\Rightarrow \Sigma m(0,1,2,3)$

X Y Z	
0 0 0	$\bar{Y}\bar{Z} = (x = 0 \text{ or } 1,\ Y = 0 \text{ & } z = 1) = 1,5$
0 0 1	$f(x,y,z) = \bar{x} + \bar{y}z = \Sigma m(0,1,2,3,5) = \Pi M(4,6,7)$
0 1 0	
0 1 1	

→ ENCODER:

- Combinat^{nl} circuit that performs inverse operations of a decoder.
- Encoder has 2^n (or fewer) input lines and n output lines. (eg: 4-to-2 and 8-to-3)

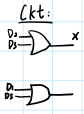
eg: 4-to-2

Inputs				Outputs	
D_0	D_1	D_2	D_3	X	Y
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

$X = D_2 + D_3$

$Y = D_1 + D_3$

Ckt:

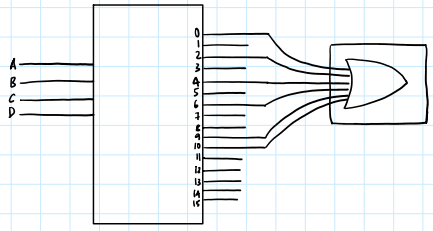


eg: 3-to-8

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	X	Y	Z
1	0	0	0	0	0	0	0			
	1				0	0	0			
		1			0	0	0			
			1		0	0	0			
				1	0	0	0			
					1	0	0			
					0	1	0			
						0	1			

Kisna) yon such

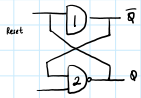
eg: Implement $f(A,B,C) = \Sigma m\,(0,2,4,6,9,10)$ using suitable decoder and external gates



eg:

sequential circuits

→ NAND LATCH:



Set	Reset	Q(t)	Q(t+1)	Q'(t+1)	
0	0	0	1	1	invalid
0	0	1	1	1	
0	1	0	1	0	Q=1 (set)
0	1	1	1	0	
1	0	0	0	1	Q=0 (reset)
1	0	1	0	1	
1	1	0	0	1	No change
1	1	1	1	0	

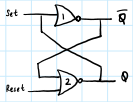
Function Table:

Set	Reset	Output
0	0	Invalid
0	1	Set Q=1
1	0	Reset Q=1
1	1	No change

Functioning:

Present o/p Q(t)	Next o/p Q(t+1)	SET	RESET
0 →	0	1	0 X
0 →	1	0	1
1 →	0	1	0
1 →	1	1 X	1 1

→ NOR LATCH:



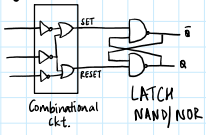
Set	Reset	Q(t)	Q(t+1)	Q'(t+1)	
0	0	0	0	1	No change
0	0	1	1	0	
0	1	0	0	1	Q=0 (reset)
0	1	1	0	1	
1	0	0	1	0	Q=1 (set)
1	0	1	1	0	
1	1	0	0	0	Invalid
1	1	1	0	0	

Function Table:

Set	Reset	Output
0	0	No change
0	1	Reset Q=0
1	0	Set Q=1
1	1	Invalid

→ SR FLIP FLOP:

(i) Using NAND latch: (Active low latch)



Clk	S	R	Q(t)	Q(t+1)	Set	Reset
0	0	0	0	0	1	X
0	0	0	1	1	X	1
0	0	1	0	0	1	X
0	0	1	1	1	X	1
0	1	0	0	0	1	X
0	1	0	1	1	X	1
0	1	1	0	0	1	X
0	1	1	1	1	X	1
1	0	0	0	0	1	X
1	0	0	1	1	X	1
1	0	1	0	0	1	X
1	0	1	1	1	1	0
1	1	0	0	1	0	1
1	1	0	1	1	X	1
1	1	1	0	X	X	X
1	1	1	1	X	X	X

Requirements for SR flip flop:

Clk	S	R	Output
0	X	X	No change
1	0	0	No change
1	0	1	Reset
1	1	0	Set
1	1	1	Invalid

Set

Clk \ R(t)	00	01	11	10
00	1	X	X	1
01	1	X	X	1
11	0	X	X	X
10	1	X	1	1

$\overline{CLK} + \overline{S}$ = SET

$\overline{CLK.S}$

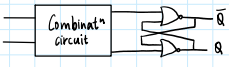
Reset

CLK \ R(t)	00	01	11	10
00	X	0	0	X
01	X	0	0	X
11	0	0	X	X
10	X	0	1	X

RESET = $\overline{\overline{CLK} + \overline{R}}$

= $\overline{CLK.R}$

(ii) Using NOR Latch:



Clk	S	R	Q(t)	Q(t+1)	Set	Reset
0	0	0	0	0	0	X
0	0	0	1	1	X	0
0	0	1	0	0	0	X
0	0	1	1	1	X	0
0	1	0	0	0	0	X
0	1	0	1	1	X	0
0	1	1	0	0	0	X
0	1	1	1	1	X	0
1	0	0	0	0	0	X
1	0	0	1	1	X	0
1	0	1	0	0	0	X
1	0	1	1	1	X	0
1	1	0	0	1	1	0
1	1	0	1	1	X	0
1	1	1	0	X	X	X
1	1	1	1	X	X	X

Q(t)	Q(t+1)	SET	RESET
0 →	0	0	X
0 →	1	1	0
1 →	0	0	1
1 →	1	X	0

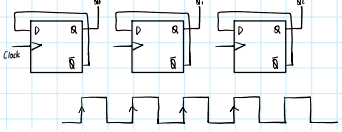
Set

Clk \ R(t)	00	01	11	10

Reset

CLK \ R(t)	00	01	11	10

Q: Design a 3-bit asynchronous UP counter using (+)ve edge triggered D flip flops.



Q_2	Q_1	Q_0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Set

Clk \ R(t)	00	01	11	10
00	0	X	X	0
01	0	X	X	0
11	1	X	X	X
10	0	X	0	0

SET = CLK.S

Reset

CLK \ R(t)	00	01	11	10
00	X	0	0	X
01	X	0	0	X
11	0	0	X	X
10	X	0	1	X

RESET = CLK.R

→ D FLIP FLOP:

- Function Table:

Clk	D	Output
0	X	No change
1	0	0
1	1	1

→ works only when clk is high.

Clk	D	Q(t)	Q(t+1)	Set	Reset
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	1	X	0
1	0	0	0	0	X
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	X	0

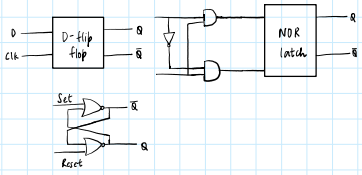
- Equation for SET and RESET:

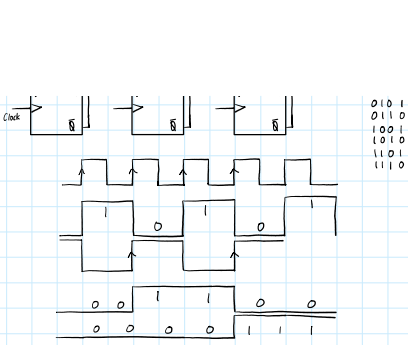
SET, CLK \ D,Q(t)	00	01	11	10
0	1	X	X	1
1	1	1	X	0

SET = $\overline{CLK} + \overline{D} = \overline{CLK.D}$

RESET, CLK \ D,Q(t)	00	01	11	10
0	X	1	1	X
1	X	0	1	1

RESET = $\overline{CLK} + D$
$= \overline{CLK.\overline{D}}$





→ DOWN COUNTER:

Q Design a 3-bit asynchronous DOWN counter using +ve edge triggered D flip flops

Sol: MOD 8 ⇒ MOD 4 x MOD 2 = MOD 2 x MOD 4

MOD MN ⇒ MOD M x MOD N