Combinational Logic Circuits

- ✔ **Standard Boolean representation:** Sum of Product (SOP) & Product of Sum (POS), Maxterm and Minterm , Conversion between SOP and POS forms, realization using NAND/NOR gates.
- ✔ **K-map reduction technique for the Boolean expression:** Minimization of Boolean functions up to 4 variables (SOP & POS form)
- ✔ **Design of Airthmetic circuits and code converter using K-map:** Half and Full Adder, Half and Full Subtractor

Combinational Logic Circuits

Standard Representation

- Any logical expression can be expressed in the following two forms:

  - ✔ Sum of Product (SOP) Form
  - ✔ Product of Sum (POS) Form

SOP Form

For Example, logical expression given is;

Sum

$$Y = A.B + B.C + A.C$$

Produc
t

POS Form

For Example, logical expression given is;

Produc
t

$$Y = (A + B).(B + C).(A + C)$$

Sum

Standard or Canonical SOP & POS Forms

- ✔ We can say that a logic expression is said to be in the standard (or canonical) SOP or POS form if each product term (for SOP) and sum term (for POS) consists of all the literals in their complemented or uncomplemented form.

Standard SOP

$$Y = ABC + ABC\overline{\ } + \overline{\ }\ AB\overline{C}$$

Each product term consists all the literals

Standard POS

$$Y = (A + B + C).(A + \overline{B} + C).(A + \overline{B} + C)$$

Each sum term consists all the literals

Examples

Sr. No.	Expression	Type
1	$Y = AB + A\overline{B}C + \overline{A}BC$	Non Standard SOP
2	$Y = AB + A\overline{B} + \overline{A}\overline{B}$	Standard SOP
3	$Y = (\overline{A} + B).(A + \overline{B}).(\overline{A} + \overline{B})$	Standard POS
4	$Y = (\overline{A} + B).(A + \overline{B} + C)$	Non Standard POS

Conversion of SOP form to Standard SOP

Procedure:

1. Write down all the terms.
2. If one or more variables are missing in any product term, expand the term by multiplying it with the sum of each one of the missing variable and its complement .
3. Drop out the redundant terms

Example 1

Convert given expression into its standard SOP form

$$Y = AB + A\overline{C} + BC$$

$$Y = AB + AC + BC$$

Missing literal is

A Missing literal is B

Missing literal is C

$$Y = AB.(C + C) + AC.(B + B) + B\overline{C}.(A + A)$$

Term formed by ORing of missing
literal & its complement

Example 1

Continue….

$$Y = AB.(C + \overline{C}) + AC.(B + \overline{B}) + B\overline{C}.(A + \overline{A})$$

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

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

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

Standard SOP form
Each product term consists all the literals

Conversion of POS form to Standard POS

Procedure:

1. Write down all the terms.
2. If one or more variables are missing in any sum term, expand the term by adding the products of each one of the missing variable and its complement .
3. Drop out the redundant terms

Example 2

Convert given expression into its standard SOP form

$$Y = (A + B).(A + C).(\overline{B} + C)$$

$$Y = (A + B).(A + C).(\overline{B} + C)$$

Missing literal is

A Missing literal is B

Missing literal is C

$$Y = (A + B + CC).(A + C + BB).(B + C + AA)$$

Term formed by ANDing of missing
literal & its complement

Example 2

Continue….

$$Y = (A + B + CC).(A + C + BB).(B + C + AA)$$

$$Y = (A + B + C)(A + B + C).(A + B + C)(A + B + C).(A + B + C)(A + B + C)$$

$$Y = (A + B + C)(A + B + C)(A + B + C)(A + B + C)$$

$$Y = (A + B + C)(A + B + C)(A + B + C)(A + B + C)$$

Standard POS form
Each sum term consists all the literals

Concept of Minterm and Maxterm

- **Minterm:** Each individual term in the standard SOP form is called as “Minterm”.
- **Maxterm:** Each individual term in the standard POS form is called as “Maxterm”.

- ✔ The concept of minterm and max term allows us to introduce a very convenient shorthand notation to express logic functions

Minterms & Maxterms for 3 variable/literal logic function

Variables			Minterms	Maxterms
A	B	C	mi	Mi
0	0	0	$\overline{A}\overline{B}\overline{C} = m_0$	$A + B + C = M_0$
0	0	1	$\overline{A}\overline{B}C = m_1$	$A + B + \overline{C} = M_1$
0	1	0	$\overline{A}B\overline{C} = m_2$	$A + \overline{B} + C = M_2$
0	1	1	$\overline{A}BC = m_3$	$A + \overline{B} + \overline{C} = M_3$
1	0	0	$A\overline{B}\overline{C} = m_4$	$\overline{A} + B + C = M_4$
1	0	1	$A\overline{B}C = m_5$	$\overline{A} + B + \overline{C} = M_5$
1	1	0	$AB\overline{C} = m_6$	$\overline{A} + \overline{B} + C = M_6$
1	1	1	$ABC = m_7$	$\overline{A} + \overline{B} + \overline{C} = M_7$

Minterms and maxterms

- Each minterm is represented by m_i where $i=0,1,2,3,\ldots\ldots,2^{n-1}$
- Each maxterm is represented by M_i where $i=0,1,2,3,\ldots\ldots,2^{n-1}$
- If 'n' number of variables forms the function, then number of minterms or maxterms will be 2^n
  - i.e. for 3 variables function f(A,B,C), the number of minterms or maxterms are $2^3=8$

Minterms & Maxterms for 2 variable/literal logic function

Variables		Minterms	Maxterms
A	B	mi	Mi
0	0	$\overline{A}\overline{B} = m_0$	$A + B = M_0$
0	1	$\overline{A}B = m_1$	$A + \overline{B} = M_1$
1	0	$A\overline{B} = m_2$	$\overline{A} + B = M_2$
1	1	$AB = m_3$	$\overline{A} + \overline{B} = M_3$

Representation of Logical expression using minterm

$$Y = \underline{ABC} + \underline{\overline{A}BC} + \underline{A\overline{B}\overline{C}} + \underline{A\overline{B}C}$$

m_7 m_3 m_4 m_5

Logical Expression

Corresponding minterms

$$Y = m_7 + m_3 + m_4 + m_5$$

$$Y = \Sigma m(3, 4, 5, 7)$$

OR

$$Y = f(A, B, C) = \Sigma m(3, 4, 5, 7)$$

where ∑denotes sum of products

Representation of Logical expression using maxterm

$$Y = \underbrace{(A + \overline{B} + C)}_{M_2}.\underbrace{(A + B + C)}_{M_0}.\underbrace{(A + \overline{B} + \overline{C})}_{M_6}$$

Logical Expression

Corresponding maxterms

$$Y = M_2.M_0.M_6$$

$$Y = \Pi M(0, 2, 6)$$

OR

$$Y = f(A, B, C) = \Pi M(0, 2, 6)$$

where Π denotes product of sum

Conversion from SOP to POS & Vice versa

- ✔ The relationship between the expressions using minters and maxterms is complementary.

- ✔ We can exploit this complementary relationship to write the expressions in terms of maxterms if the expression in terms of minterms is known and vice versa

Conversion from SOP to POS & Vice versa

- For example, if a SOP expression for 4 variable is given by,

$$Y = \Sigma m(0,1, 3, 5, 6, 7,11,12,15)$$

- Then we can get the equivalent expression using POS the complementar relationship as follows, y

$$Y = \Pi M\,(2, 4,8, 9,10,13,14)$$

Examples

1. Convert the given expression into standard form

$$Y = A + BC + ABC$$

2. Convert the given expression into standard form

$$Y = (A + B).(A + C)$$

Karnaugh Map (K-map)

- ✔ In the algebraic method of simplification, we need to write lengthy equations, find the common terms, manipulate the expressions etc., so it is time consuming work.

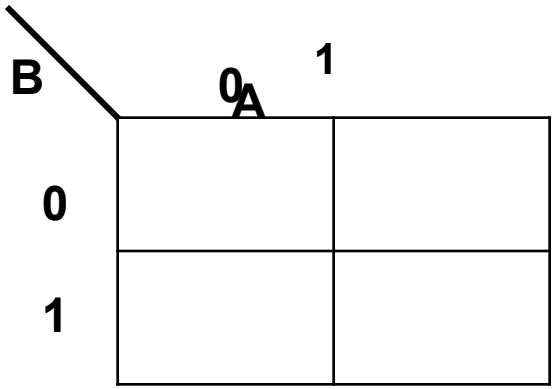
- ✔ Thus “K-map” is another simplification technique to reduce the Boolean equation.

Karnaugh Map (K-map)

- ✔ It overcomes all the disadvantages of algebraic simplification techniques.
- ✔ The information contained in a truth table or available in the SOP or POS form is represented on K-map.

Karnaugh Map (K-map)

- K-map Structure - 2 Variable
  - ✔ A & B are variables or inputs
  - ✔ 0 & 1 are values of A & B
  - ✔ 2 variable k-map consists of 4 boxes i.e. $2^2=4$



Karnaugh Map (K-map)

K-map Structure - 2 Variable

- ✔ Inside 4 boxes we have enter values of Y i.e. output

B \ A	$\overline{A}$ 0	A 1
$\overline{B}$ 0	$\overline{\overline{A}B}$	$\overline{A}B$
B 1	$\overline{A}B$	AB

B \ A	$\overline{A}$ 0	A 1
$\overline{B}$ 0	m_0	m_1
B 1	m_2	m_3

K-map & its associated minterms

Karnaugh Map (K-map)

✔ Relationship between Truth Table & K-map

A	B	Y
0	0	0
0	1	1
1	0	0
1	1	1

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

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

Karnaugh Map (K-map)

- K-map Structure - 3 Variable
  - ✔ A, B & C are variables or inputs
  - ✔ 3 variable k-map consists of 8 boxes i.e. $2^3=8$

C \ AB	00	01	11	10
0				
1				

A \ BC	00	01	11	10
0				
1				

BC \ A	0	1
00		
01		
11		
10		

Karnaugh Map (K-map)

✔ 3 Variable K-map & its associated product terms

C \ AB	00	01	11	10
0	$\overline{A}\overline{B}\overline{C}$	$\overline{A}B\overline{C}$	$AB\overline{C}$	$A\overline{B}\overline{C}$
1	$\overline{A}\overline{B}C$	$\overline{A}BC$	ABC	$A\overline{B}C$

A \ BC	00	01	11	10
0	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}C$	$\overline{A}BC$	$\overline{A}B\overline{C}$
1	$A\overline{B}\overline{C}$	$A\overline{B}C$	ABC	$AB\overline{C}$

BC \ A	0	1
00	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}\overline{C}$
01	$\overline{A}\overline{B}C$	$\overline{A}\overline{B}C$
11	$\overline{A}BC$	ABC
10	$\overline{A}B\overline{C}$	$AB\overline{C}$

Karnaugh Map (K-map)

✔ 3 Variable K-map & its associated minterms

C \ AB	00	01	11	10
0	m_0	m_2	m_6	m_4
1	m_1	m_3	m_7	m_5

A \ BC	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

BC \ A	0	1
00	m_0	m_4
01	m_1	m_5
11	m_3	m_7
10	m_2	m_6

Karnaugh Map (K-map)

- K-map Structure - 4 Variable
  - ✔ A, B, C & D are variables or inputs
  - ✔ 4 variable k-map consists of 16 boxes i.e. $2^4=16$

CD \ AB	00	01	11	10
00				
01				
11				
10				

AB	00	01	11	10
00				
01				
11				
10				

Karnaugh Map (K-map)

✔ 4 Variable K-map and its associated product terms

CD \ AB	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}B\bar{C}D$	$\bar{A}\bar{B}\bar{C}D$
01	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}B\bar{C}D$	$\bar{A}\bar{B}\bar{C}D$
11	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}CD$	$ABCD$	$\bar{A}\bar{B}CD$
10	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}\bar{D}$	$AB\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}\bar{D}$

AB \ CD	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}\bar{C}\bar{D}$
01	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}\bar{C}D$
11	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}CD$	$ABCD$	$AB\bar{C}\bar{D}$
10	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}\bar{C}\bar{D}$

Karnaugh Map (K-map)

✔ 4 Variable K-map and its associated minterms

CD \ AB	00	01	11	10
00	m_0	m_4	m_{12}	m_8
01	m_1	m_5	m_{13}	m_9
11	m_3	m_7	m_{15}	m_{11}
10	m_2	m_6	m_{11}	m_{10}

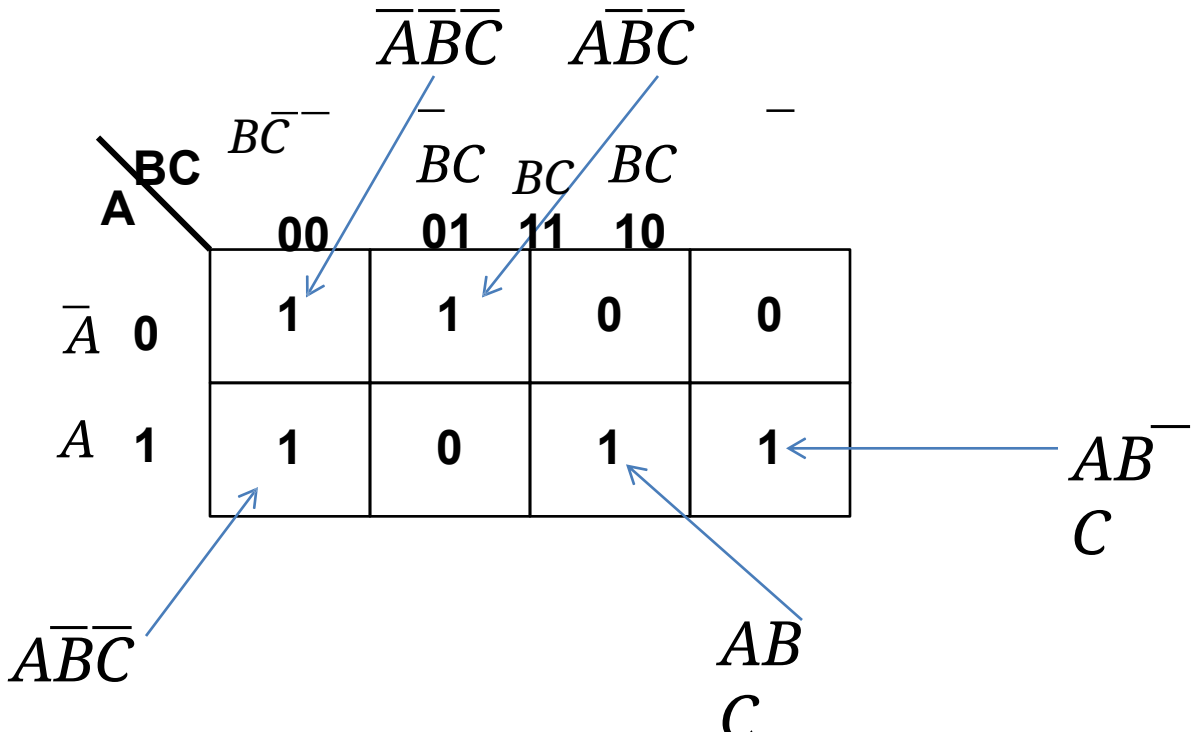
AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

Representation of Standard SOP form expression on K-map

For example, SOP equation is given as

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

- ✔ The given expression is in the standard SOP form.
- ✔ Each term represents a minterm.
- ✔ We have to enter '1' in the boxes corresponding to each minterm as below



Simplification of K-map

- ✔ Once we plot the logic function or truth table on K-map, we have to use the grouping technique for simplifying the logic function.
- ✔ Grouping means the combining the terms in adjacent cells.
- ✔ The grouping of either 1's or 0's results in the simplification of Boolean expression.

Simplification of K-map

- If we group the adjacent 1's then the result of

  simplification is SOP form
- If we group the adjacent 0's then

  the result of simplification is POS form

Grouping

- ✔ While grouping, we should group most number of 1's.
- ✔ The grouping follows the binary rule i.e we can group 1,2,4,8,16,32,……..number of 1's.
- ✔ We cannot group 3,5,7,……number of 1's
- ✔ ***Pair***: A group of two adjacent 1's is called as Pair
- ✔ ***Quad***: A group of four adjacent 1's is called as Quad
- ✔ ***Octet***: A group of eight adjacent 1's is called as Octet

Grouping of Two Adjacent 1's : Pair

✔ *A pair eliminates 1 variable*

$\bar{A}B\bar{C}$ $\bar{A}B\bar{C}$

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

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

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

$$Y = AB$$

$(\because C + \bar{C} = 1)$

Grouping of Two Adjacent 1's : Pair

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

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

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

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

Grouping of Two Adjacent 1's : Pair

AB \ CD	$\overline{C}\overline{D}$ (00)	$\overline{C}D$ (01)	CD (11)	$C\overline{D}$ (10)
$\overline{A}\overline{B}$ (00)	0	1	0	0
$\overline{A}B$ (01)	0	0	0	0
AB (11)	0	0	0	0
$A\overline{B}$ (10)	0	1	0	0

Possible Grouping of Four Adjacent 1's : Quad

✔ *A Quad eliminates 2 variable*

CD / AB	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\overline{B}$ 10	1	1	1	1

CD / AB	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	1	0	0
$\overline{A}B$ 01	0	1	0	0
AB 11	0	1	0	0
$A\overline{B}$ 10	0	1	0	0

Possible Grouping of Four Adjacent 1's : Quad

✔ *A Quad eliminates 2 variable*

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	1	1	0	0
AB 11	1	1	0	0
$A\overline{B}$ 10	0	0	0	0

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	1	1	0
$\overline{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\overline{B}$ 10	0	1	1	0

Possible Grouping of Four Adjacent 1's : Quad

✔ *A Quad eliminates 2 variable*

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	1	0	0	1
$\overline{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\overline{B}$ 10	1	0	0	1

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	1	0	0	1
AB 11	1	0	0	1
$A\overline{B}$ 10	0	0	0	0

Possible Grouping of Four Adjacent 1's : Quad

✔ *A Quad eliminates 2 variable*

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	0	1	1	1
AB 11	0	1	1	1
$A\overline{B}$ 10	0	0	0	0

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	0	1	1	0
AB 11	0	1	1	0
$A\overline{B}$ 10	0	1	1	0

Possible Grouping of Eight Adjacent 1's : Octet

✔ *A Octet eliminates 3 variable*

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	0	0	0
$\overline{A}B$ 01	0	0	0	0
AB 11	1	1	1	1
$A\overline{B}$ 10	1	1	1	1

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	0	1	1	0
$\overline{A}B$ 01	0	1	1	0
AB 11	0	1	1	0
$A\overline{B}$ 10	0	1	1	0

Possible Grouping of Eight Adjacent 1's : Octet

✔ *A Octet eliminates 3 variable*

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	1	1	1	1
$\overline{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\overline{B}$ 10	1	1	1	1

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	1	0	0	1
$\overline{A}B$ 01	1	0	0	1
AB 11	1	0	0	1
$A\overline{B}$ 10	1	0	0	1

Rules for K-map simplification

1. Groups may not include any cell containing a zero.

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

Not Accepted

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

Accepted

Rules for K-map simplification

2. Groups may be horizontal or vertical, but may not be diagonal

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

Not Accepted

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

Accepted

Rules for K-map simplification

3. Groups must contain 1,2,4,8 or in general 2^n cells

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

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

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

Not Accepted

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

Accepted

Rules for K-map simplification

4. Each group should be as large as possible

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

Not Accepted

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

Accepted

Rules for K-map simplification

5. Each cell containing a one must be in at least one group

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

Rules for K-map simplification

6. Groups may be overlap

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

Rules for K-map simplification

7. Groups may wrap around the table. The leftmost cell in a row may be grouped with rightmost cell and the top cell in a column may be grouped with bottom cell

AB \ CD	$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	CD 11	$C\overline{D}$ 10
$\overline{A}\overline{B}$ 00	1	1	1	1
$\overline{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\overline{B}$ 10	1	1	1	1

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

Rules for K-map simplification

8. There should be as few groups as possible, as long as this does not contradict any of the previous rules.

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

Not Accepted

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

Accepted

Rules for K-map simplification

9. *A pair eliminates one variable.*

10. *A Quad eliminates two variables.*

11. *A octet eliminates three variables*

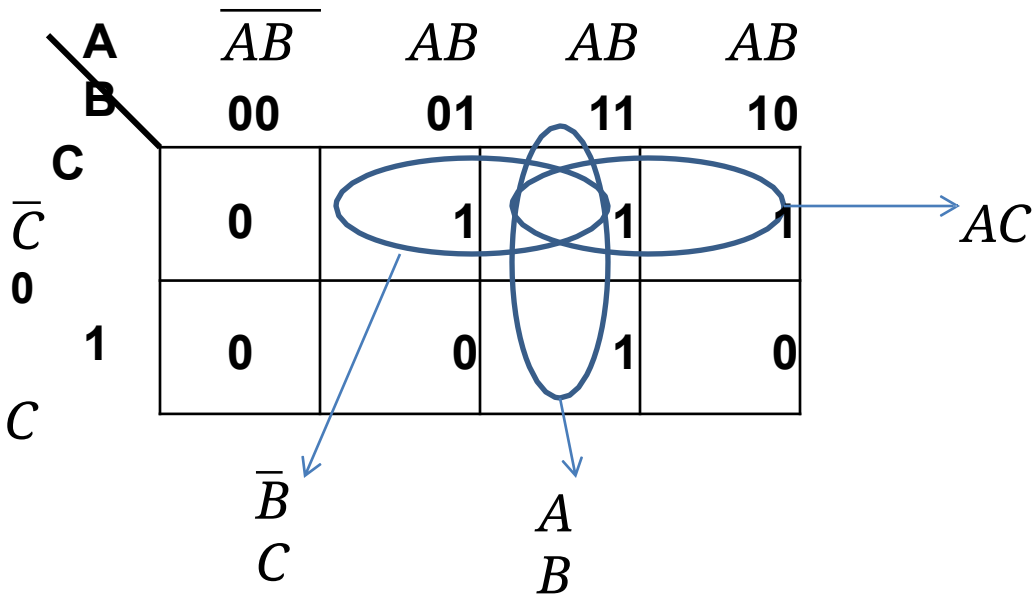
Example 1

For the given K-map write simplified Boolean expression

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

Example 1

continue…..



Simplified Boolean expression

$$Y = B\overline{C} + AB + A\overline{C}$$

Example 2

For the given K-map write simplified Boolean expression

C \ AB	$A\overline{B}$ 00	$\overline{A}B$ 01	AB 11	$A\overline{B}$ 10
$\overline{C}$ 0	1	1	0	1
C 1	1	0	0	1

Example 2

continue…..

C \ AB	$A\overline{B}$ 00	$\overline{A}B$ 01	AB 11	$A\overline{B}$ 10
$\overline{C}$ 0	1	1	0	1
C 1	1	0	0	1

$\overline{A}\overline{C}$

$\overline{B}$

Simplified Boolean expression

$$Y = \overline{B} + AC$$

Example 3

A logical expression in the standard SOP form is as follows;

$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\overline{C} + \overline{A}BC + \overline{A}BC$$

Minimize it with using the K-map technique

Example 3

continue……

$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + \overline{A}BC + \overline{A}BC$$

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

AB

$\overline{A}\,\overline{C}$

$A\overline{B}C$

Simplified Boolean expression

$$Y = \overline{A}\,\overline{C} + \overline{A}B + A\overline{B}\,\overline{C}$$

Example 4

A logical expression representing a logic circuit is;

$$Y = \Sigma m(0,1, 2, 5,13,15)$$

Draw the K-map and find the minimized logical expression

Example 4

continue…..

$$Y = \Sigma m(0,1,2,5,13,15)$$

C D / AB	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$ 00	00 0 1	01 1 1	11 3 0	10 2 1
$\overline{A}B$ 01	4 0	5 1	7 0	6 0
AB 11	12 0	13 1	15 1	14 0
$A\overline{B}$ 10	8 0	9 0	11 0	10 0

AB
D

$\overline{A}\,\overline{C}$
D

AB
D

Simplified Boolean expression

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

Example 5

Minimize the following Boolean expression using K-map ;

$$f(A, B, C, D) = \Sigma m(1, 3, 5, 9, 11, 13)$$

Example 5

continue…..

$$f(A, B, C, D) = \Sigma m(1, 3, 5, 9, 11, 13)$$

AB \ CD	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$ 00	00 0	01 1	11 3	10 2
$\overline{A}B$ 01	4 0	5 1	7 0	6 0
AB 11	12 0	13 1	15 0	14 0
$A\overline{B}$ 10	8 0	9 1	11 1	10 0

$\overline{B}D$

$\overline{C}D$

Simplified Boolean expression

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

$$f = D(\overline{B} + \overline{C})$$

Example 6

Minimize the following Boolean expression using K-map ;

$$f(A, B, C, D) = \Sigma m(4, 5, 8, 9, 11, 12, 13, 15)$$

Example 6

continue…..

$$f(A, B, C, D) = \Sigma m(4,5,8,9,11,12,13,15)$$

AB \ CD	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$ 00	00 0 0	01 1 0	11 3 0	10 2 0
$\overline{A}B$ 01	4 1	5 1	7 0	6 0
AB 11	12 1	13 1	15 1	14 0
$A\overline{B}$ 10	8 1	9 1	11 1	10 0

$\overline{B}C$

$\overline{A}C$

AD

Simplified Boolean expression

$$f = \overline{B}C + \overline{A}C + AD$$

Example 7

Minimize the following Boolean expression using K-map ;

$$f_2(A, B, C, D) = \Sigma m(0,1, 2, 3,11,12,14,15)$$

Example 7

continue…..

$$f_2(A, B, C, D) = \Sigma m(0,1, 2,3,11,12,14,15)$$

CD / AB	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	00 1 (0)	01 1 (1)	11 1 (3)	10 1 (2)
$\overline{A}B$	01 0 (4)	0 (5)	0 (7)	0 (6)
AB	1 1 1 (12)	0 (13)	1 (15)	1 1 (14)
$A\overline{B}$	1 0 0 (8)	0 (9)	1 (11)	1 0 (10)

A

B

$AB\overline{D}$

AC

D

Simplified Boolean expression

$$f_2 = \overline{A}\,\overline{B}\,\overline{D} + ABD + ACD$$

Example 8

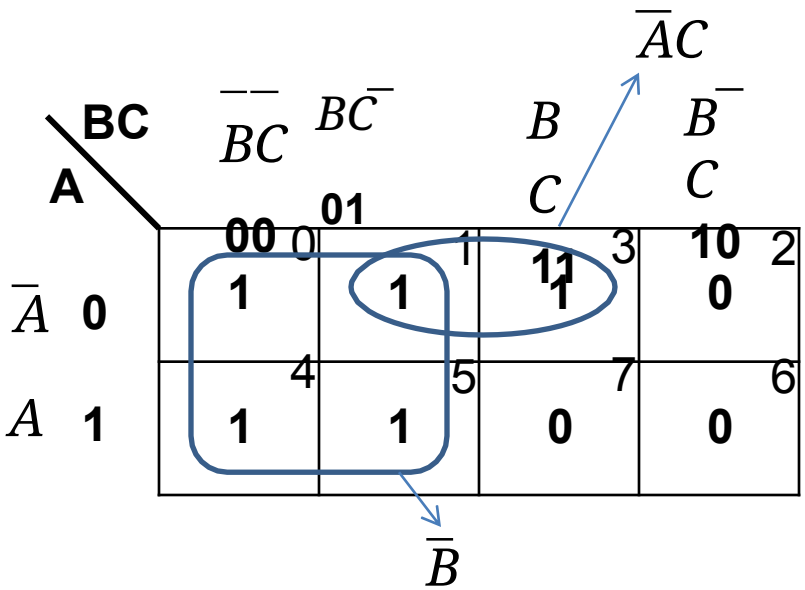
Solve the following expression with K-maps;

1. $f_1(A, B, C) = \Sigma m(0,1, 3, 4, 5)$
2. $f_2(A, B, C) = \Sigma m(0,1, 2, 3, 6, 7)$

Example 8

continue……

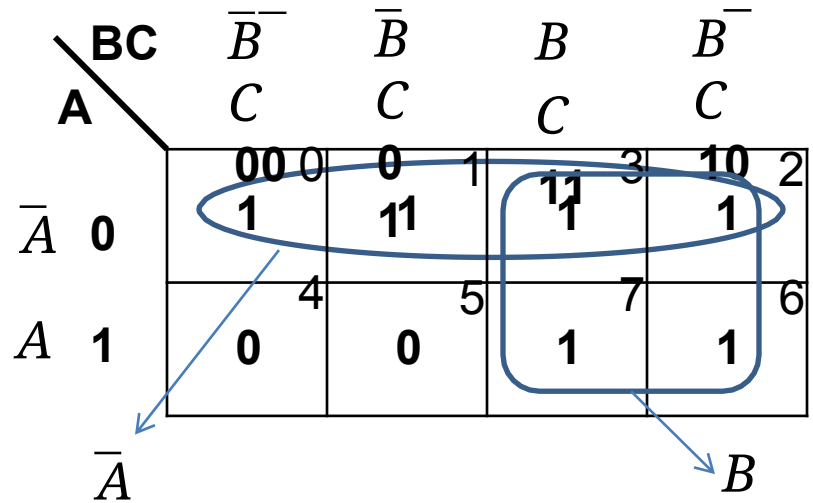
$f_1(A, B, C) = \Sigma m(0,1,3,4,5)$



Simplified Boolean expression

$f_1 = \overline{A}C + B\overline{C}$

$f_2(A, B, C) = \Sigma m(0,1,2,3,6,7)$



Simplified Boolean expression

$f_2 = \overline{A} + B$

Example 9

Simplify ;

$$f(A, B, C, D) = \Sigma m(0,1, 4, 5, 7,8, 9,12,13,15)$$

Example 9

continue…..

$$f(A, B, C, D) = \Sigma m(0,1, 4,5, 7,8,9,12,13,15)$$

AB \ CD	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$ 00	0: 1	1: 1	3: 0	2: 0
$\overline{A}B$ 01	4: 1	5: 1	7: 1	6: 0
AB 11	12: 1	13: 1	15: 1	14: 0
$A\overline{B}$ 10	8: 1	9: 1	11: 0	10: 0

$\overline{C}$

BD

Simplified Boolean expression

$$f = \overline{C} + BD$$

Example 10

Solve the following expression with K-maps;

1. $f_1(A, B, C, D) = \Sigma m(0,1, 3, 4, 5, 7)$
2. $f_2(A, B, C) = \Sigma m(0,1, 3, 4, 5, 7)$

Example 10

continue……

$f_1(A, B, C, D) = \Sigma m(0,1,3,4,5,7)$

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$ 00	00 0: 1	01 1: 1	11 3: 0	10 2: 0
$\bar{A}B$ 01	4: 1	5: 1	7: 1	6: 0
AB 11	12: 0	13: 0	15: 0	14: 0
$A\bar{B}$ 10	8: 0	9: 0	11: 0	10: 0

$\bar{A}\bar{C}$ $\bar{A}D$

Simplified Boolean expression

$f_1 = \bar{A}\bar{C} + \bar{A}D$

$f_2(A, B, C) = \Sigma m(0,1,3,4,5,7)$

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

$\bar{B}$ C

Simplified Boolean expression

$f_2 = \bar{B} + C$

K-map for Product of Sum Form (POS Expressions)

- Karnaugh map can also beused for Boolean expression in the Product of sum form (POS).
- The procedure for simplification of expression by grouping of cells is also similar

K-map for Product of Sum Form (POS Expressions)

- ✔ The letters with bars (NOT) represent 1 and unbarred

  letters represent 0 of Binary.

- ✔ A zero is put in the cell for which there is a term in

  the Boolean expression

- ✔ Grouping is done for adjacent cells containing zeros.

Example 11

Simplify ;

$$f(A, B, C, D) = \Pi M\,(0,1, 3, 5, 6, 7,10,14,15)$$

Example 11

continue…..

$f(A, B, C, D) = \Pi M(0,1,3,5,6,7,10,14,15)$

CD / AB	$\bar{C}\bar{D}$	$\bar{C}D$	$C\bar{D}$	CD
$\bar{A}\bar{B}$ 00	0 (0)	0 (1)	0 (3)	1 (2)
$\bar{A}B$ 01	1 (4)	0 (5)	0 (7)	0 (6)
$A\bar{B}$ 11	1 (12)	1 (13)	0 (15)	0 (14)
AB 10	1 (8)	1 (9)	1 (11)	0 (10)

$A + \bar{D}$

$\bar{B} + \bar{C}$

$A + B + C$

$\bar{A} + \bar{C} + D$

Simplified Boolean expression

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

Example 12

Simplify ;

$$f(A, B, C, D) = \Pi M\,(4, 6, 10, 12, 13, 15)$$

Example 12

continue…..

$f(A, B, C, D) = \Pi M(4, 6, 10, 12, 13, 15)$

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

$A + \bar{B} + D$

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

$\bar{A} + \bar{B} + D$

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

Simplified Boolean expression

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

K-map and don't care conditions

- ✔ For SOP form we enter 1's corresponding to the combinations of input variables which produce a high output and we enter 0's in the remaining cells of the K-map.
- ✔ For POS form we enter 0's corresponding to the combinations of input variables which produce a high output and we enter 1's in the remaining cells of the K-map.

K-map and don't care conditions

- ✔ But it is not always true that the cells not containing 1's (in SOP) will contain 0's, because some combinations of input variable do not occur.
- ✔ Also for some functions the outputs corresponding to certain combinations of input variables do not matter.

K-map and don't care conditions

- ✔ In such situations we have a freedom to assume a 0 or 1 as output for each of these combinations.
- ✔ These conditions are known as the "Don't Care Conditions" and in the K-map it is represented as 'X', in the corresponding cell.
- ✔ The don't care conditions may be assumed to be 0 or 1 as per the need for

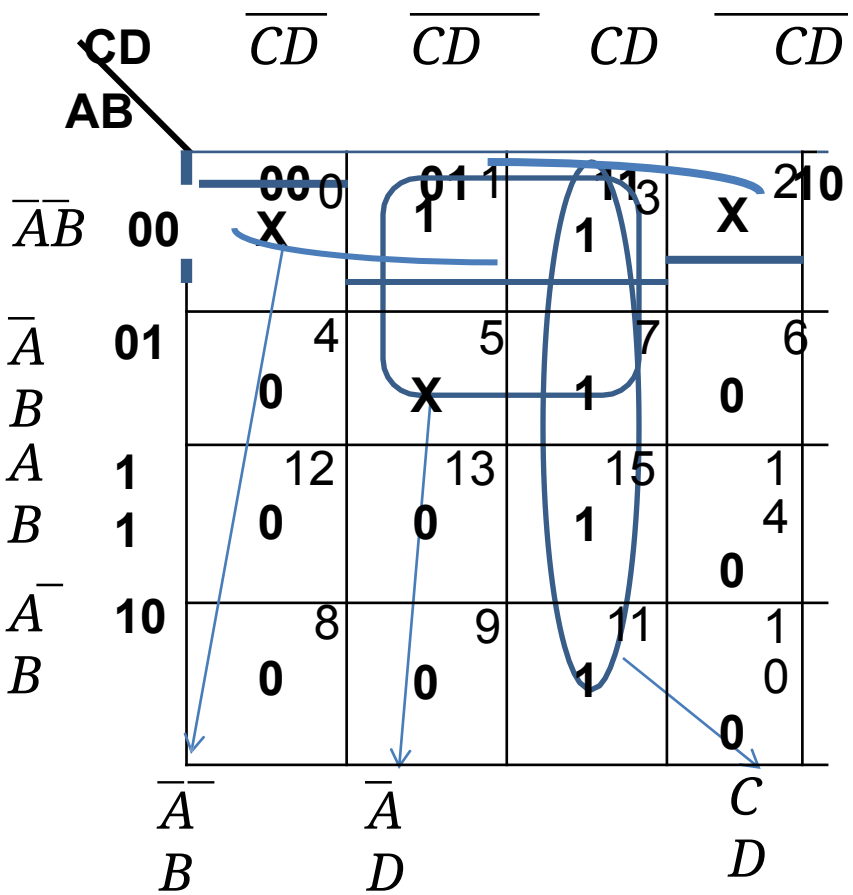
K-map and don’t care conditions - Example

Simplify ;

$$f(A, B, C, D) = \Sigma m(1, 3, 7, 11, 15) + d(0, 2, 5)$$

K-map and don't care conditions - Example

$$f(A, B, C, D) = \Sigma m(1,3, 7,11,15) + d(0, 2,5)$$



Simplified Boolean expression

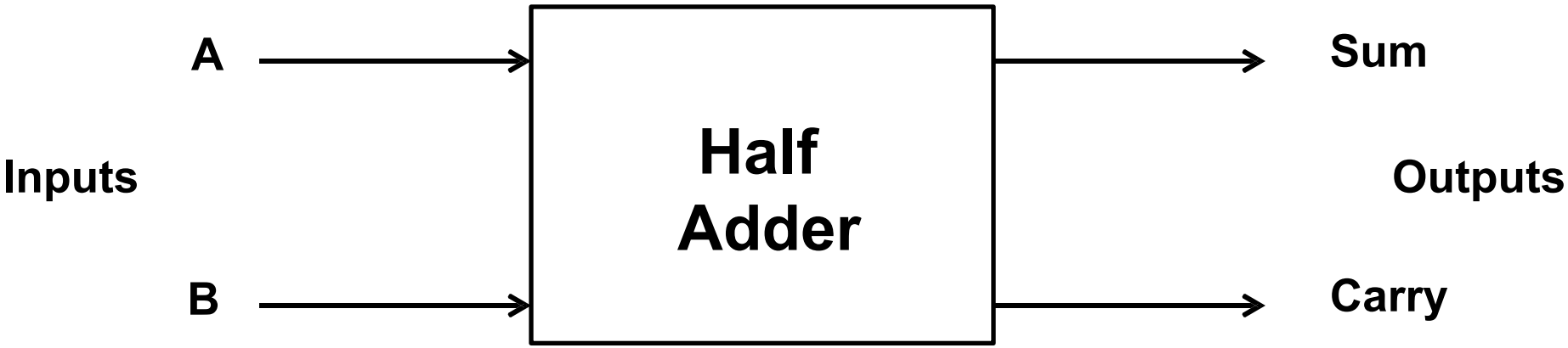
$$f = CD + \overline{A}\overline{B} + \overline{A}D$$

Combinational Logic Circuits

- ✔ **Standard Boolean representation:** Sum of Product (SOP) & Product of Sum (POS), Maxterm and Minterm , Conversion between SOP and POS forms, realization using NAND/NOR gates.
- ✔ **K-map reduction technique for the Boolean expression:** Minimization of Boolean functions up to 4 variables (SOP & POS form)
- ✔ **Design of Airthmetic circuits and code converter using K-map: Half Adder** and Full Adder, Half and Full Subtractor, Gray to Binary and Binary to Gray Code Converter (up to 4 bit).

Half Adder

- ✔ Half adder is a combinational logic circuit with two inputs and two outputs.
- ✔ It is a basic building block for addition of two single bit numbers.



Half Adder

Truth Table for Half Adder

Input		Output	
A	**B**	**Sum (S)**	**Carry (C)**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Half Adder

K-map for Sum Output:

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

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

$$S = A \oplus B$$

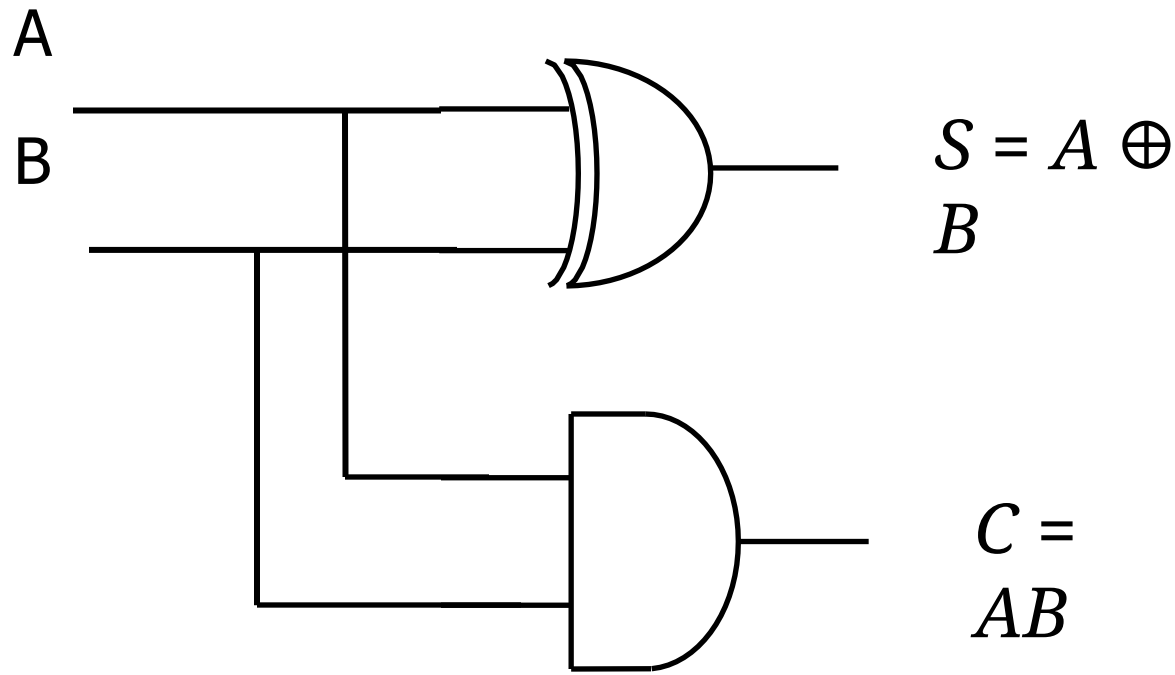
K-map for Carry Output:

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

$$C = AB$$

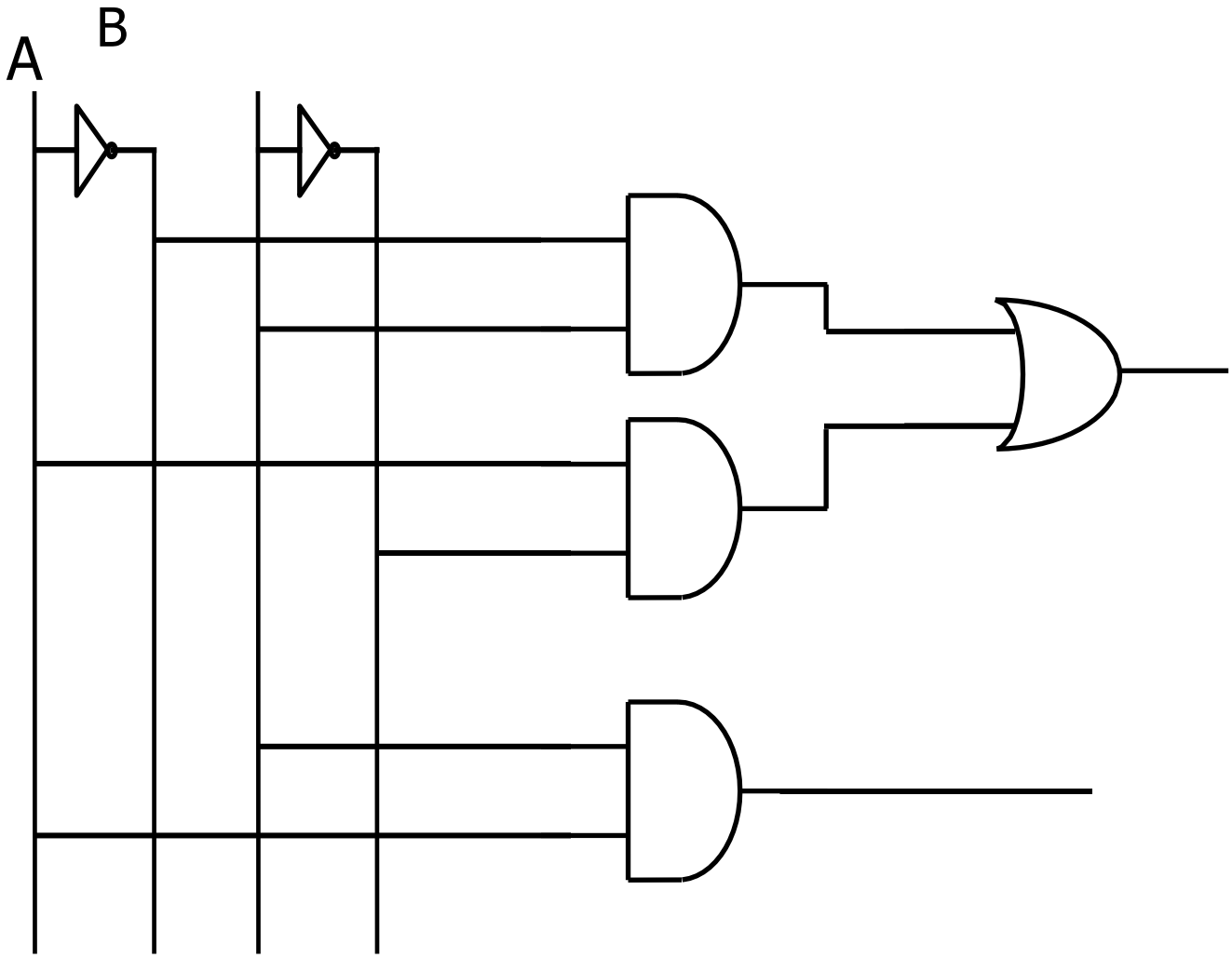
Half Adder

Logic Diagram:



Half Adder

Logic Diagram using Basic Gates:



$S = A \oplus B$

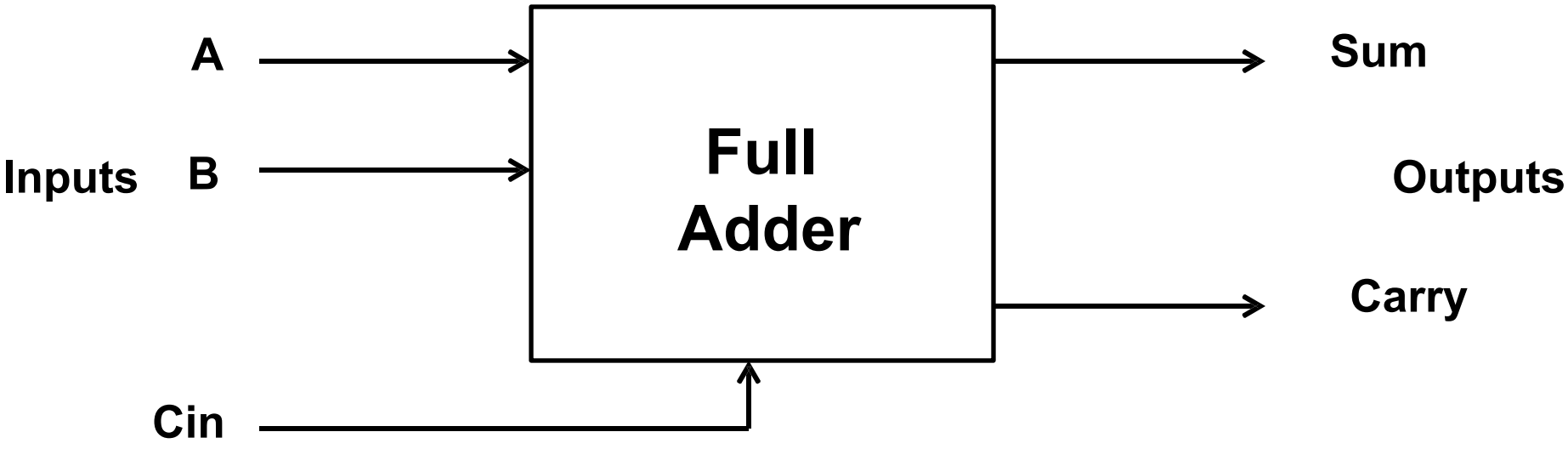
$C = AB$

Combinational Logic Circuits

- ✔ **Standard Boolean representation:** Sum of Product (SOP) & Product of Sum (POS), Maxterm and Minterm , Conversion between SOP and POS forms, realization using NAND/NOR gates.
- ✔ **K-map reduction technique for the Boolean expression:** Minimization of Boolean functions up to 4 variables (SOP & POS form)
- ✔ **Design of Airthmetic circuits and code converter using K-map:** Half and **Full Adder,** Half and Full Subtractor, Gray to Binary and Binary to Gray Code Converter (up to 4 bit).

Full Adder

✔ Full adder is a combinational logic circuit with three inputs and two outputs.



Full Adder

Truth Table

Inputs			Outputs	
A	B	Cin	Sum (S)	Carry (C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Full Adder

K-map for Sum Output:

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

$A\overline{B}\,\overline{C}$ $\quad$ $\overline{A}\overline{B}C$ $\quad$ $AB\overline{C}$ $\quad$ $\overline{A}\overline{B}\,\overline{C}$

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

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

$$S = C(\overline{A}\,\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B})$$

$$Let\, AB + AB = \overline{X}$$

$$\therefore S = C(\overline{X}) + \overline{C}(X)$$

$$S = C \oplus X$$

$$Let\, X = A \oplus B \quad S =$$

$$\therefore C \oplus A \oplus B$$

Full Adder

K-map for Carry Output:

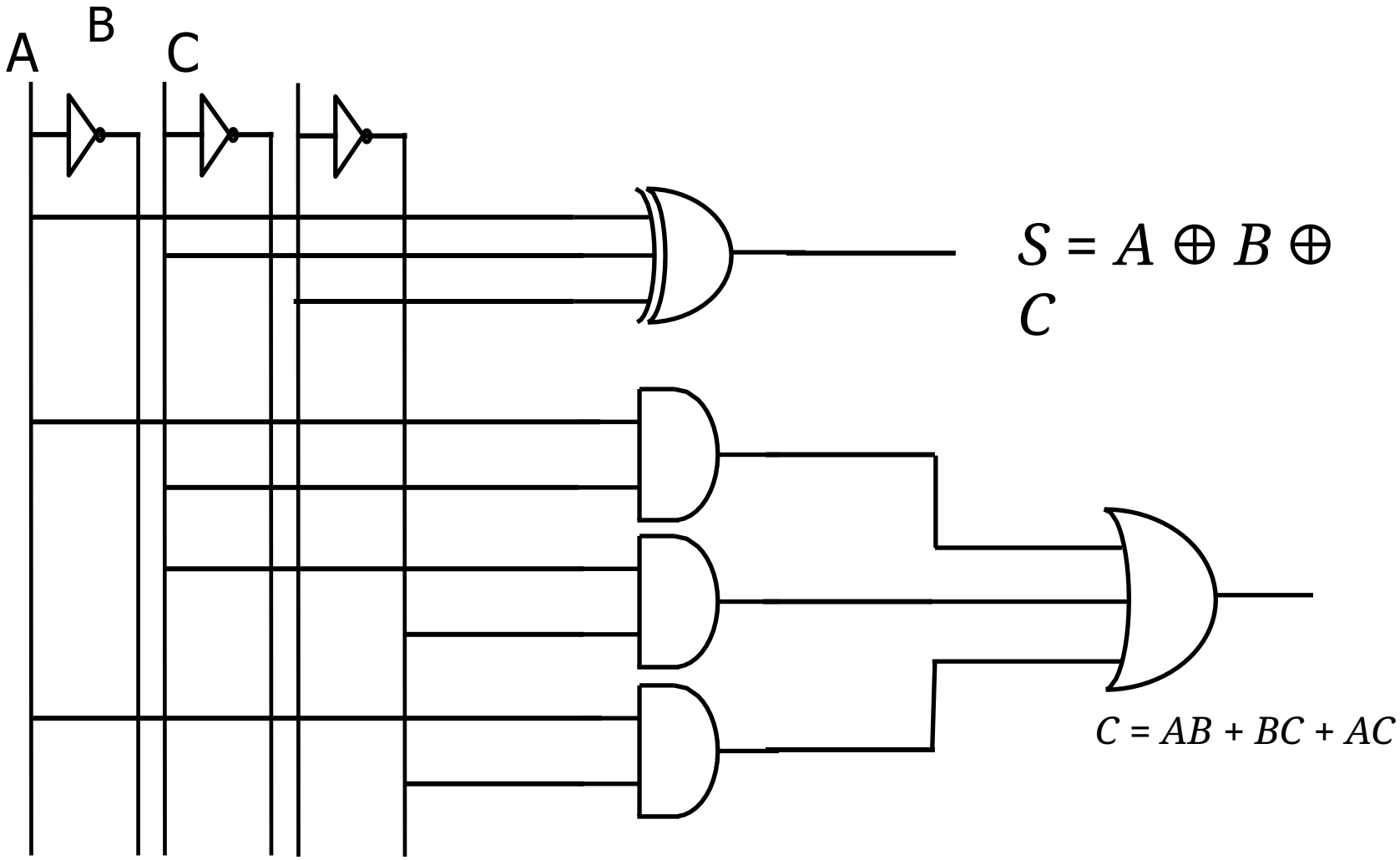
A \ BC	$\overline{B}\,\overline{C}$ 00	$\overline{B}C$ 01	$B\,C$ 11	$B\,\overline{C}$ 10
$\overline{A}$ 0	0	0	1	0
A 1	0	1	1	1

$A\,C$ $\quad$ $B\,C$ $\quad$ $A\,B$

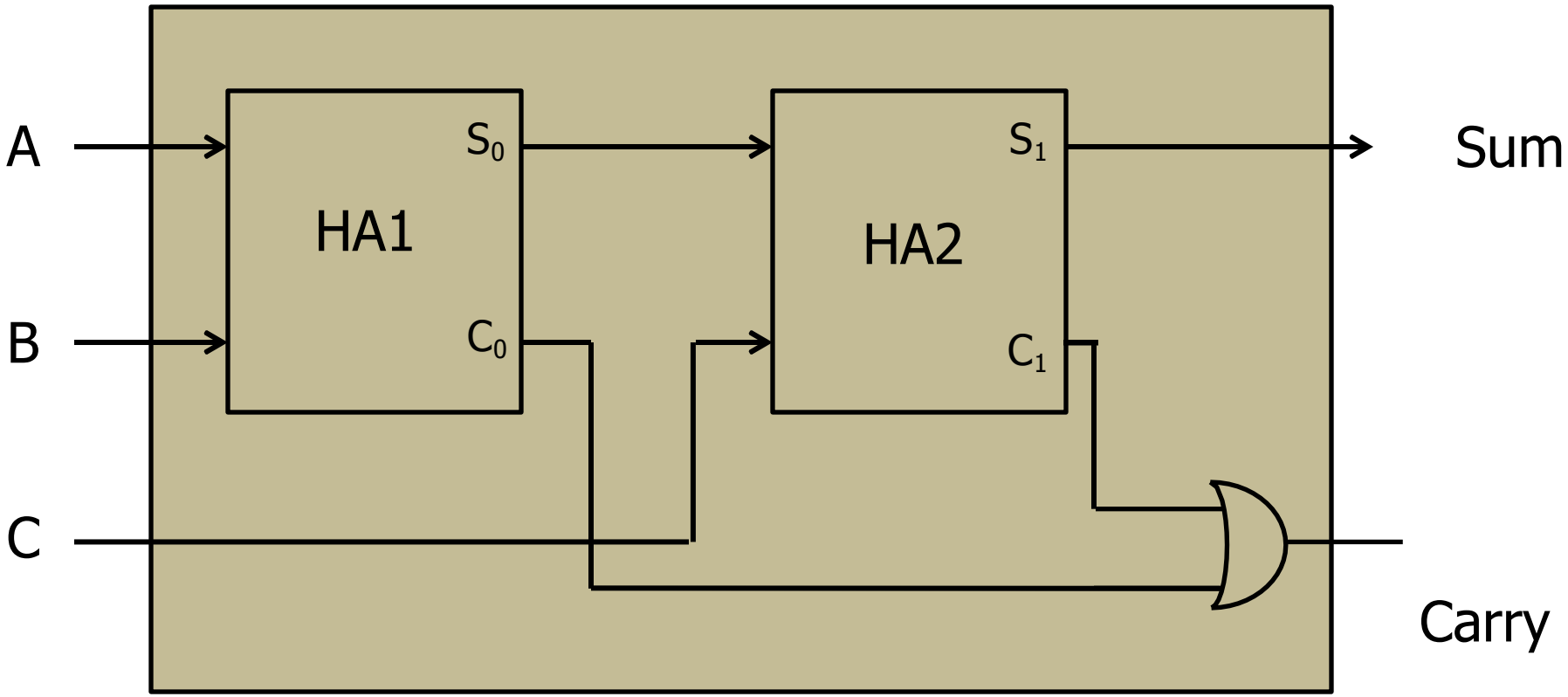
$$C = AB + BC + AC$$

Full Adder

Logic Diagram:



Full Adder using Half Adders

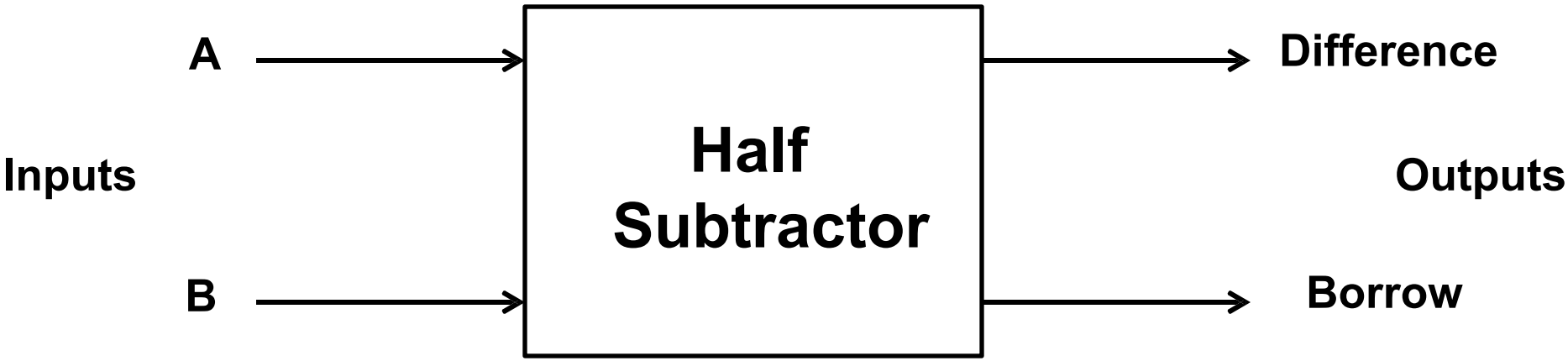


Combinational Logic Circuits

- ✔ **Standard Boolean representation:** Sum of Product (SOP) & Product of Sum (POS), Maxterm and Minterm , Conversion between SOP and POS forms, realization using NAND/NOR gates.
- ✔ **K-map reduction technique for the Boolean expression:** Minimization of Boolean functions up to 4 variables (SOP & POS form)
- ✔ **Design of Airthmetic circuits and code converter using K-map:** Half and Full Adder, **Half Subtractor** and Full Subtractor, Gray to Binary and Binary to Gray Code Converter (up to 4 bit).

Half Subtractor

- ✔ Half subtractoris a combinational logic circuit with two inputs and two outputs.
- ✔ It is a basic building block for subtraction of two single bit numbers.



Half Subtractor

Truth Table

Input		Output	
A	**B**	**Difference (D)**	**Borrow (B)**
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Half Subtractor

K-map for Difference Output:

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

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

$$D = A \oplus B$$

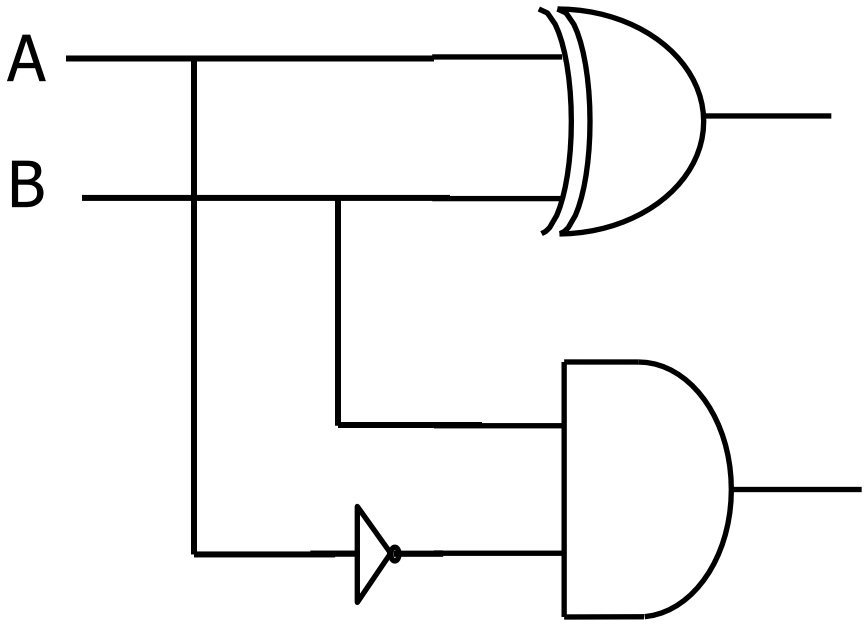
K-map for Borrow Output:

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

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

Half Subtractor

Logic Diagram:

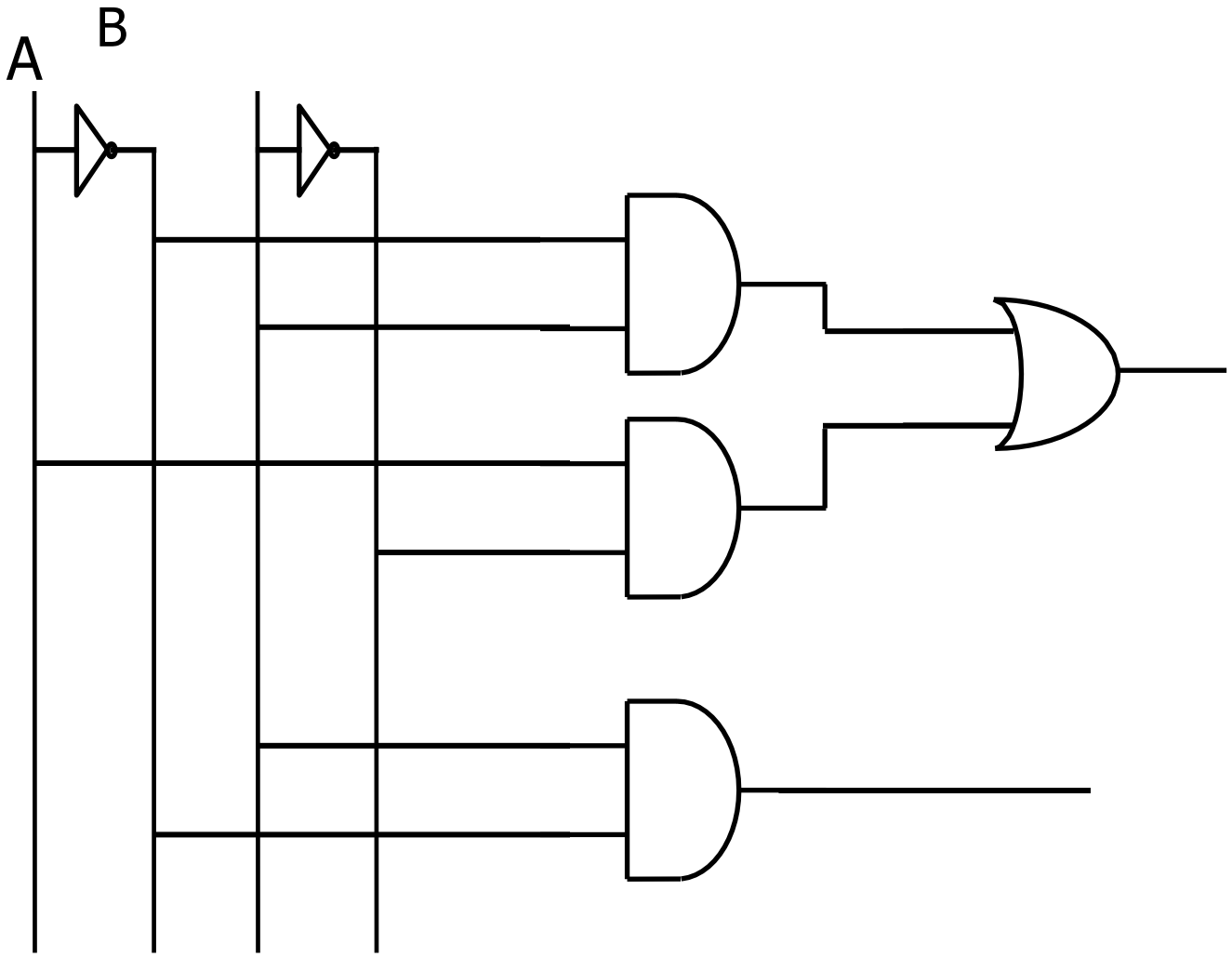


$D = A \oplus B$

$B = \overline{A}B$

Half Subtractor

Logic Diagram using Basic Gates:



$D = A \oplus B$

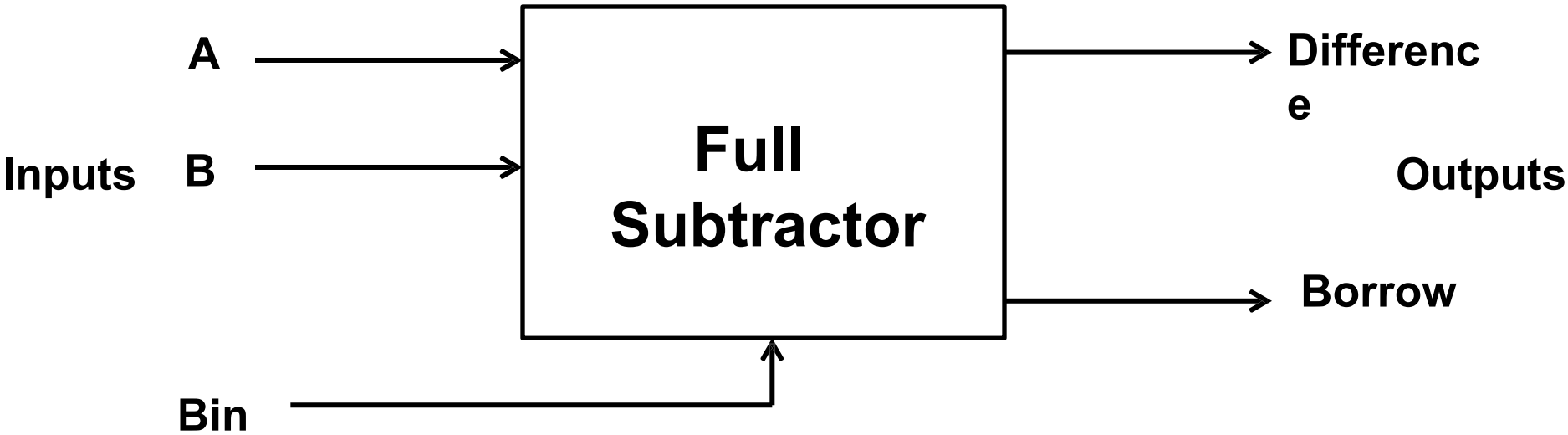
$B = \overline{A}B$

Combinational Logic Circuits

- ✔ **Standard Boolean representation:** Sum of Product (SOP) & Product of Sum (POS), Maxterm and Minterm , Conversion between SOP and POS forms, realization using NAND/NOR gates.
- ✔ **K-map reduction technique for the Boolean expression:** Minimization of Boolean functions up to 4 variables (SOP & POS form)
- ✔ **Design of Airthmetic circuits and code converter using K-map:** Half and Full Adder, Half and **Full Subtractor,** Gray to Binary and Binary to Gray Code Converter (up to 4 bit).

Full Subtractor

✔ Full subtractoris a combinational logic circuit with three inputs and two outputs.



Full Subtractor

Truth Table

Inputs			Outputs	
A	B	Bin (C)	Difference (D)	Borrow (B0)
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	0	0	0
1	1	1	1	1

Full Subtractor

K-map for Difference Output:

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

$A\overline{B}\,\overline{C}$ $\quad$ $\overline{A}\overline{B}C$ $\quad$ $AB\overline{C}$ $\quad$ $\overline{A}\overline{B}\,\overline{C}$

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

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

$$D = C(\overline{A}\,\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B})$$

$$Let\, AB + AB = \overline{X}$$

$$\therefore D = C(\overline{X}) + \overline{C}(X)$$

$$D = C \oplus X$$

$$Let\, X = A \oplus B \quad D =$$

$$\therefore C \oplus A \oplus B$$

Full Subtractor

K-map for Borrow Output:

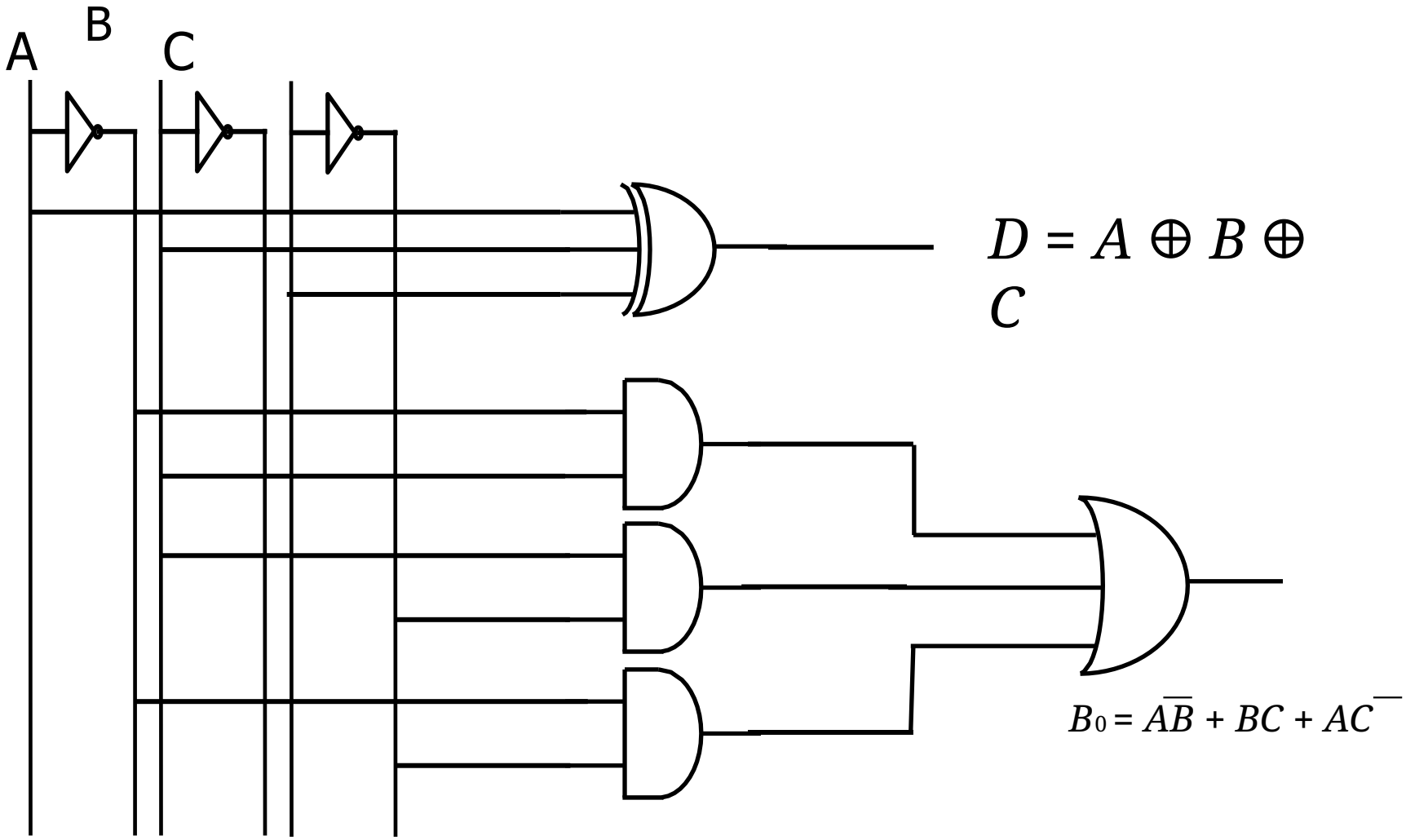
A \ BC	$\overline{B}\,\overline{C}$ 00	$\overline{B}C$ 01	BC 11	$B\overline{C}$ 10
$\overline{A}$ 0	0	1	1	1
A 1	0	0	1	0

$\overline{A}C$ $\quad$ BC $\quad$ $\overline{A}B$

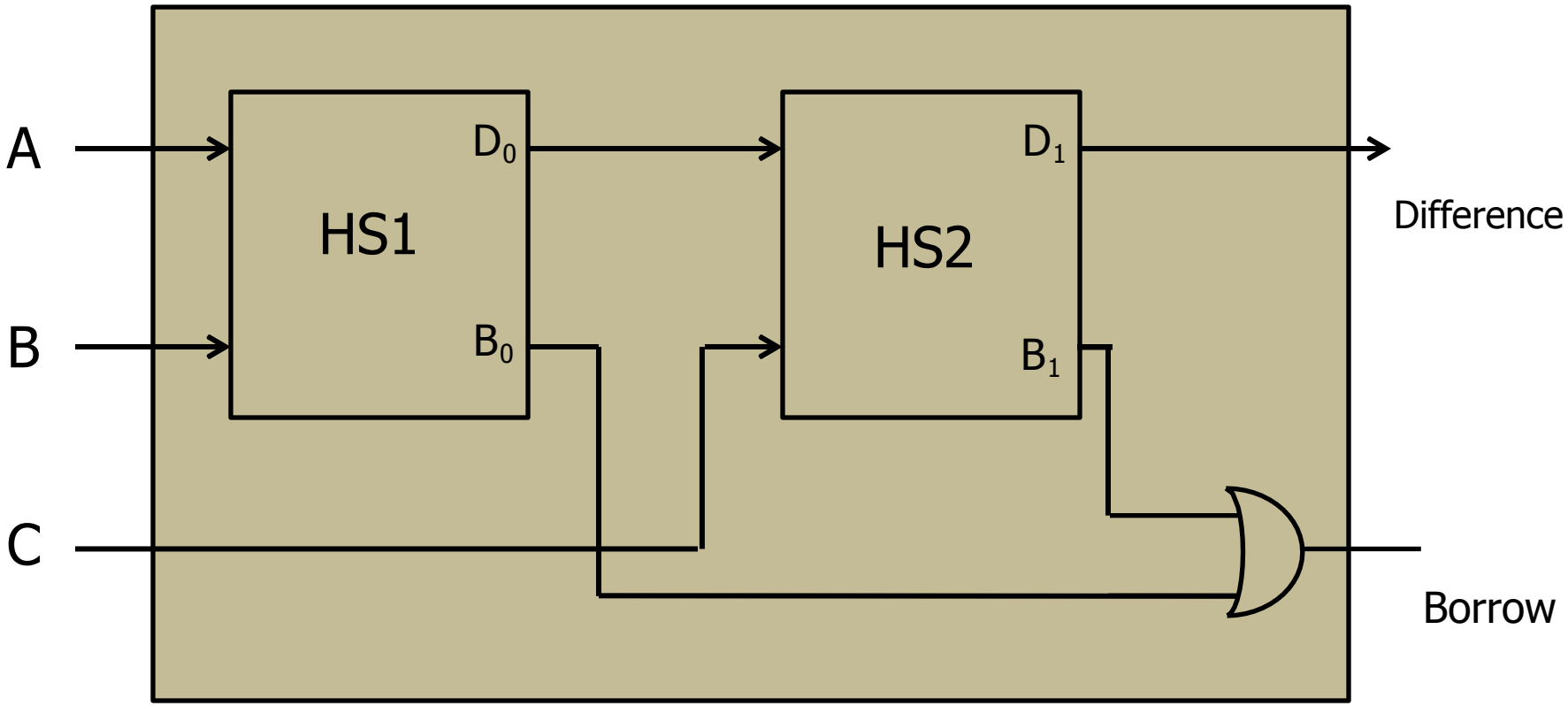
$$B_0 = \overline{A}B + BC + \overline{A}C$$

Full Subtractor

Logic Diagram:



Full Subtractor using Half Subtractor



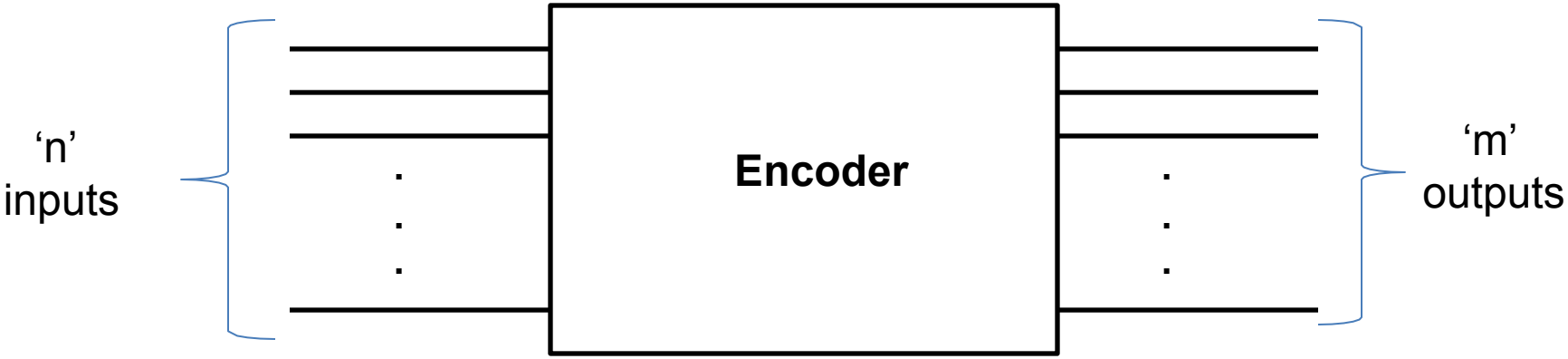
Combinational Logic Circuits

- ✔ **Airthmetic Circuits:** (IC 7483) Adder & Subtractor, BCD Adder
- ✔ **Encoder/Decoder: Basics of Encoder, decoder, comparison, (IC 7447) BCD to 7- Segment decoder/ driver.**
- ✔ **Multiplexer and Demultiplexer:** Working, truth table and applications of Multiplexers and Demultiplexers, MUX tree, IC 74151 as MUX, DEMUX tree, DEMUX as decoder, IC 74155 as DEMUX
- ✔ **Buffer:** Tristate logic, Unidirectional and Bidirectional buffer (IC 74LS244 and IC 74LS245)

Encoder

- Encoder is a combinational circuit which is designed to perform the inverse operation of decoder.
- An encoder has ‘n’ number of input lines and ‘m’ number of output lines.
- An encoder produces an m bit binary code corresponding to the digital input number.
- The encoder accepts an n input digital word and converts it into m bit another digital word

Encoder



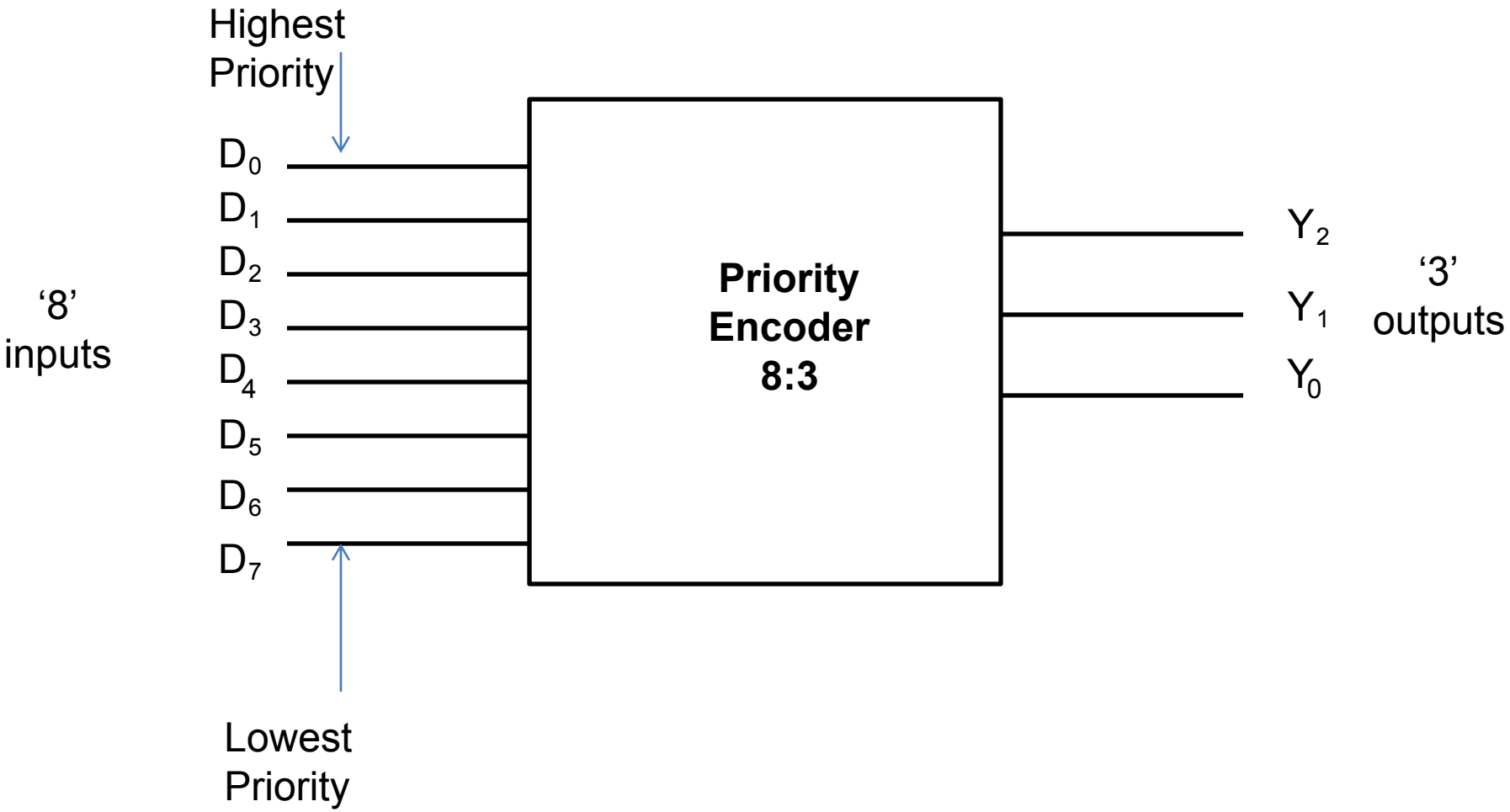
Types of Encoders

- ✔ Priority Encoder
- ✔ Decimal to BCD Encoder
- ✔ Octal to BCD Encoder
- ✔ Hexadecimal to Binary Encoder

Priority Encoder

- ✔ This is a special type of encoder.
- ✔ Priorities are given to the input lines.
- ✔ If two or more input lines are “1” at the same time, then the input line with highest priority will be considered.

Priority Encoder 8:3

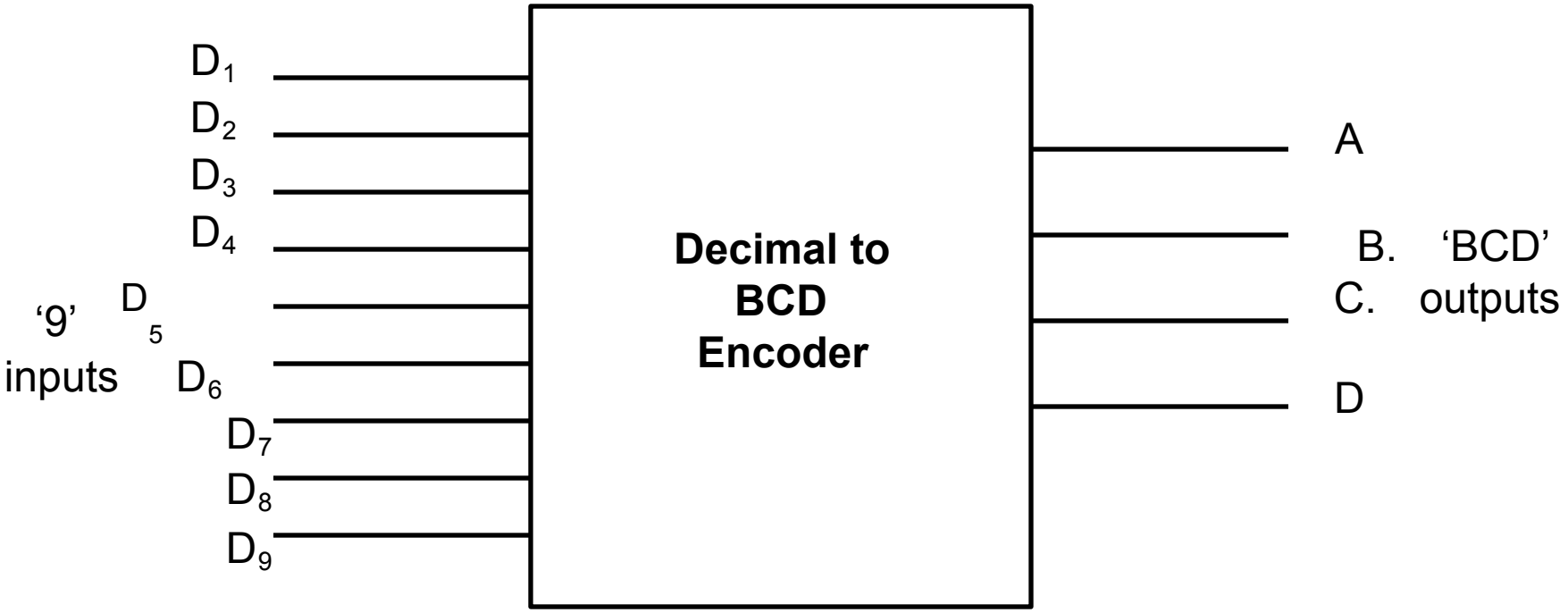


Priority Encoder 8:3

Truth Table:

Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	X	X	X
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	X	0	0	1
0	0	0	0	0	1	X	X	0	1	0
0	0	0	0	1	X	X	X	0	1	1
0	0	0	1	X	X	X	X	1	0	0
0	0	1	X	X	X	X	X	1	0	1
0	1	X	X	X	X	X	X	1	1	0
1	X	X	X	X	X	X	X	1	1	1

Decimal to BCD Encoder



Decimal to BCD Encoder

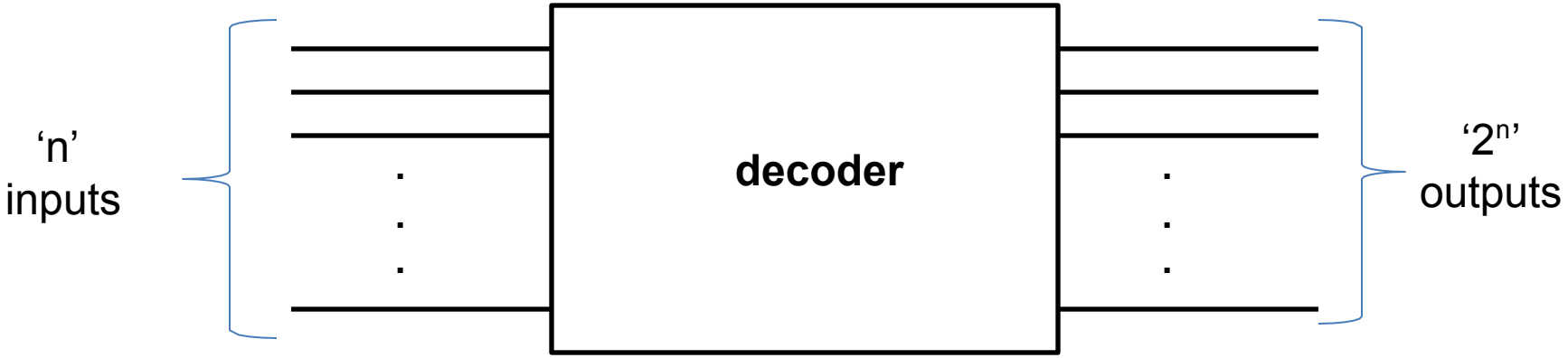
Truth Table:

Inputs									Outputs			
D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D	C	B	A
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	X	0	0	1	0
0	0	0	0	0	0	1	X	X	0	0	1	1
0	0	0	0	0	1	X	X	X	0	1	0	0
0	0	0	0	1	X	X	X	X	0	1	0	1
0	0	0	1	X	X	X	X	X	0	1	1	0
0	0	1	X	X	X	X	X	X	0	1	1	1
0	1	X	X	X	X	X	X	X	1	0	0	0
1	X	X	X	X	X	X	X	X	1	0	0	1

Decoder

- Decoder is a combinational circuit is designe to perform the inverse of d operation which encoder.
- An decoder has ‘n’ number of input lines and maximum ‘2^n’ number of output lines.
- Decoder is identical to a demultiplexer without data input.

Decoder



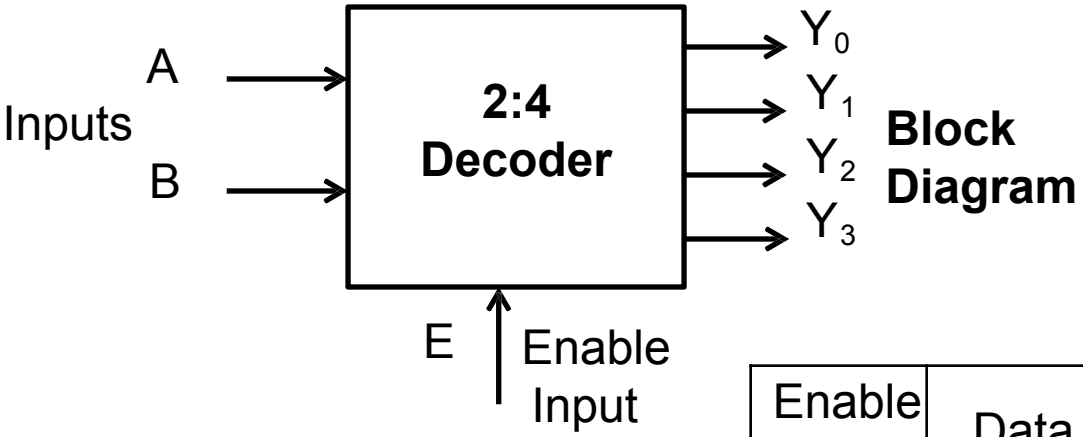
Typical applications of Decoders

- ✔ Code Converters
- ✔ BCD to 7 segment decoders
- ✔ Nixie tube decoders
- ✔ Relay actuators

Types of Decoders

- ✔ 2 to 4 line Decoder
- ✔ 3 to 8 line Decoder
- ✔ BCD to 7 Segment Decoder

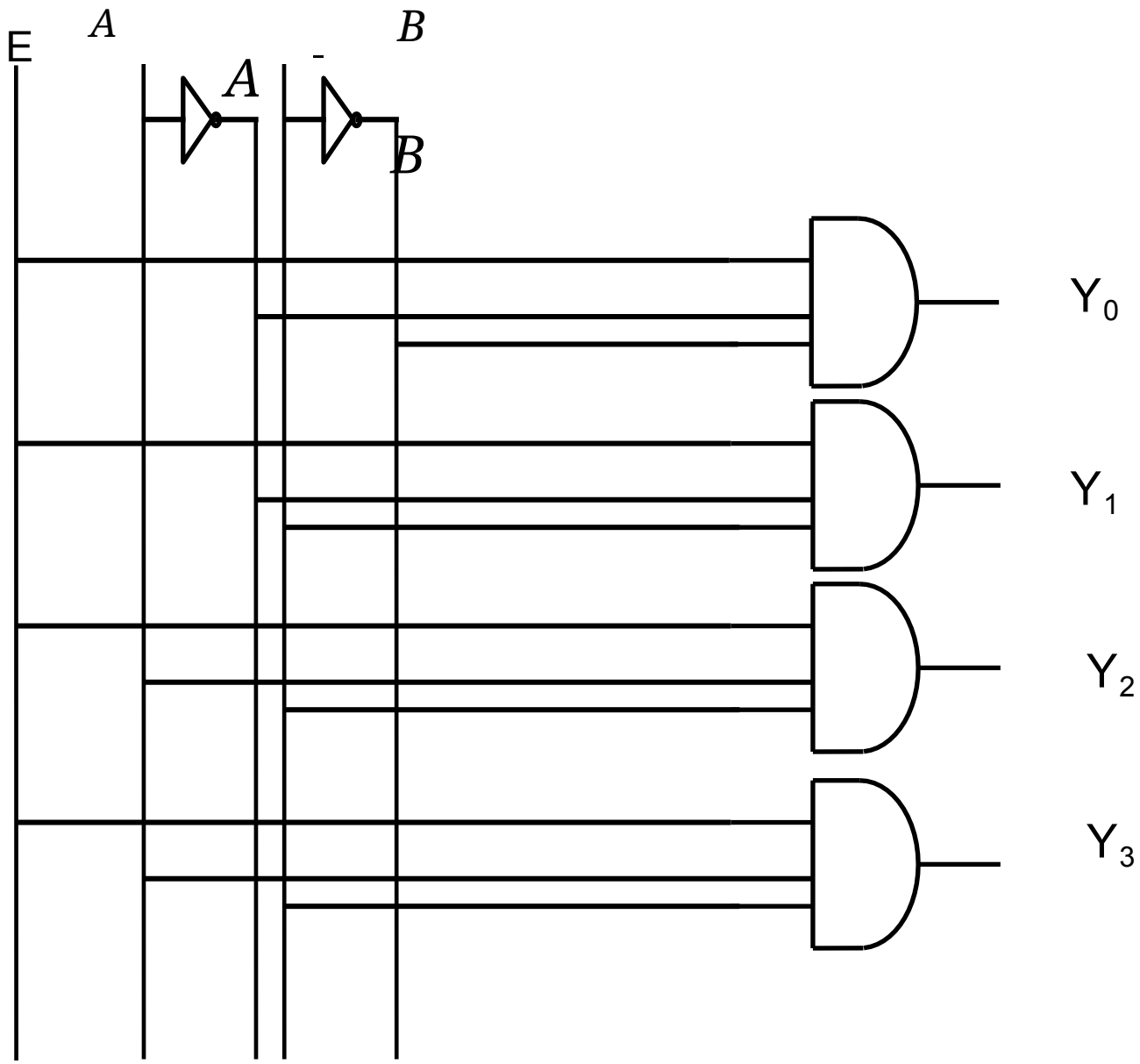
2 to 4 Line Decoder



Truth Table

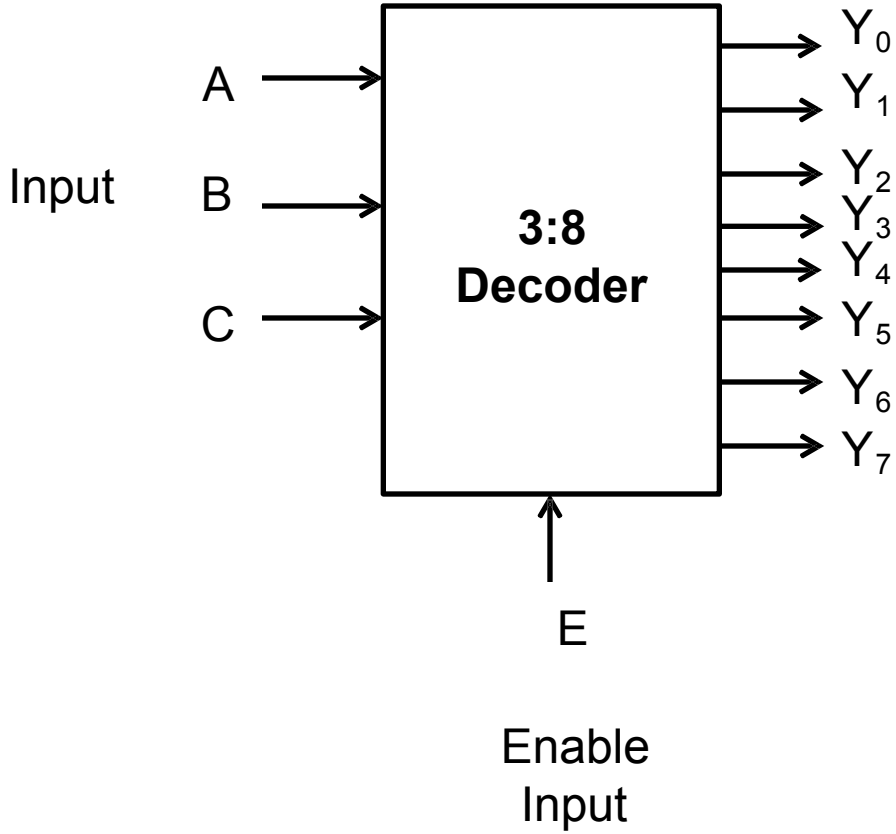
Enable i/p	Data Inputs		Outputs			
E	A	B	Y_0	Y_1	Y_2	Y_3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

2 to 4 Line Decoder



3 to 8 Line Decoder

Block Diagram



3 to 8 Line Decoder

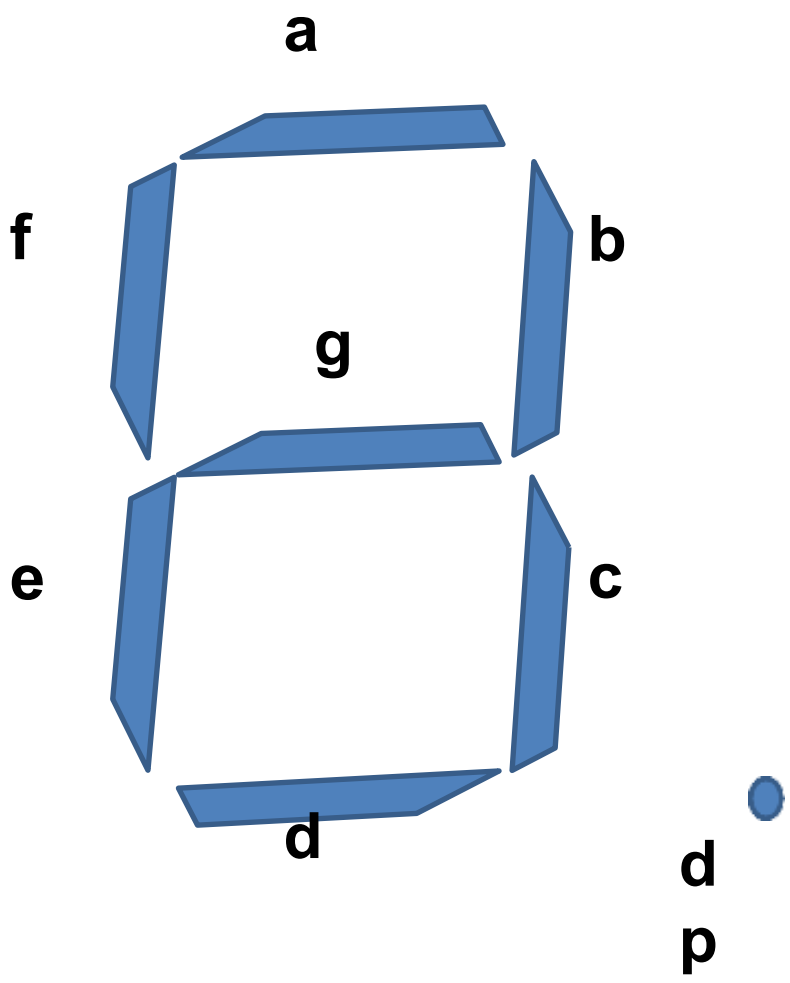
Truth Table

Enable i/p	Inputs			Outputs							
E	A	B	C	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	X	X	X	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	

Comparison between Encoder & Decoder

Sr. No.	Parameter	Encoder	Decoder
1	Input applied	Active input signal (original message signal)	Coded binary input
2	Output generated	Coded binary output	Active output signal (original message)
3	Input lines	2^n	n
4	Output lines	N	2^n
5	Operation	Simple	Complex
6	Applications	E-mail , video encoders etc.	Microprocessors, memory chips etc.

BCD to 7 Segment Decoder - Seven Segment Display



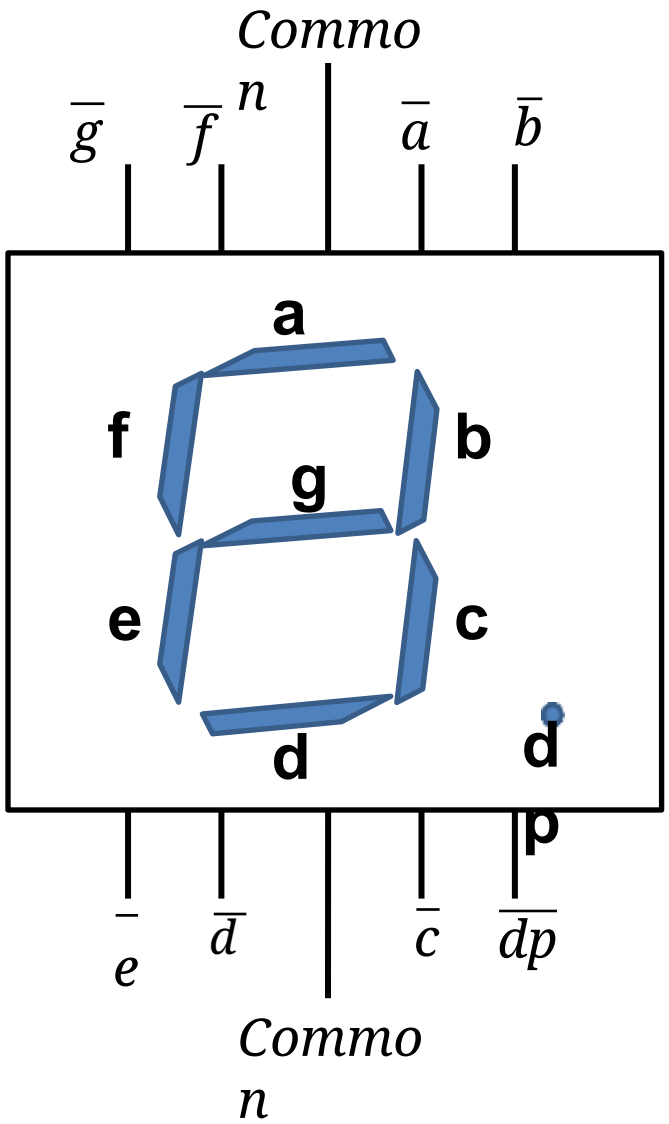
Seven Segment Display

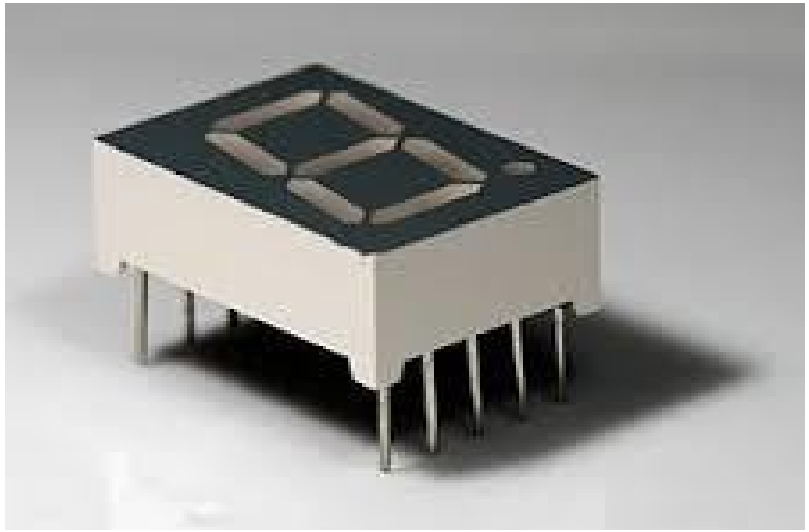
Segments							Display Number	Seven Segment Display
a	b	c	d	e	f	g		
ON	ON	ON	ON	ON	ON	OFF	0	
OFF	ON	ON	OFF	OFF	OFF	OFF	1	
ON	ON	OFF	ON	ON	OFF	ON	2	
ON	ON	ON	ON	OFF	OFF	ON	3	
OFF	ON	ON	OFF	OFF	ON	ON	4	
ON	OFF	ON	ON	OFF	ON	ON	5	
ON	OFF	ON	ON	ON	ON	ON	6	
ON	ON	ON	OFF	OFF	OFF	OFF	7	
ON	ON	ON	ON	ON	ON	ON	8	
ON	ON	ON	ON	OFF	ON	ON	9	

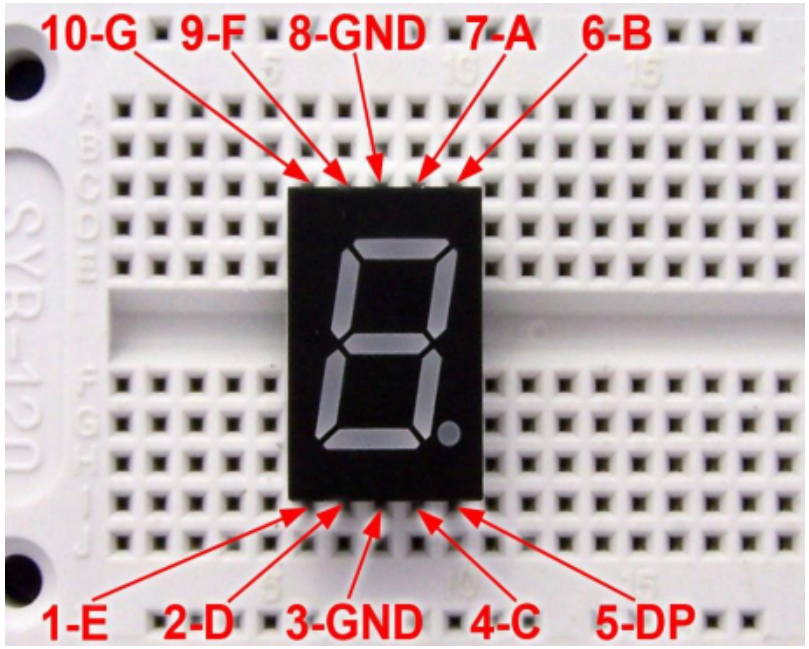
Types of Seven Segment Display

- ✔ Common Cathode Display
- ✔ Common Anode Display

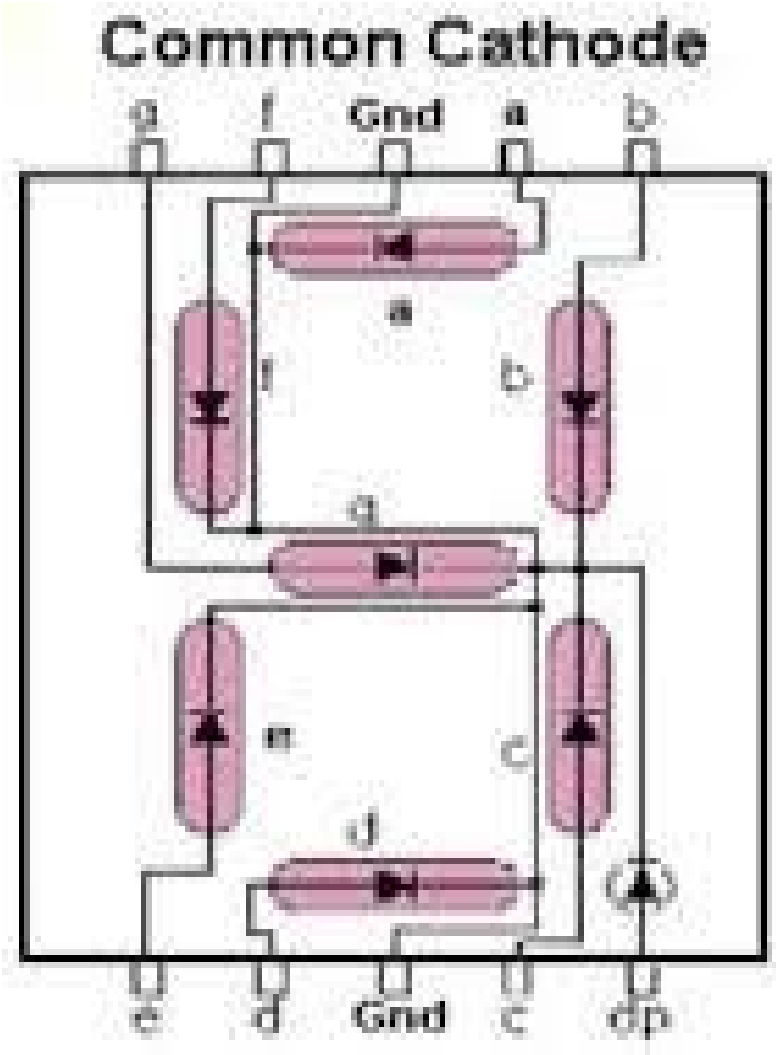
Display Configuration – LTS 542

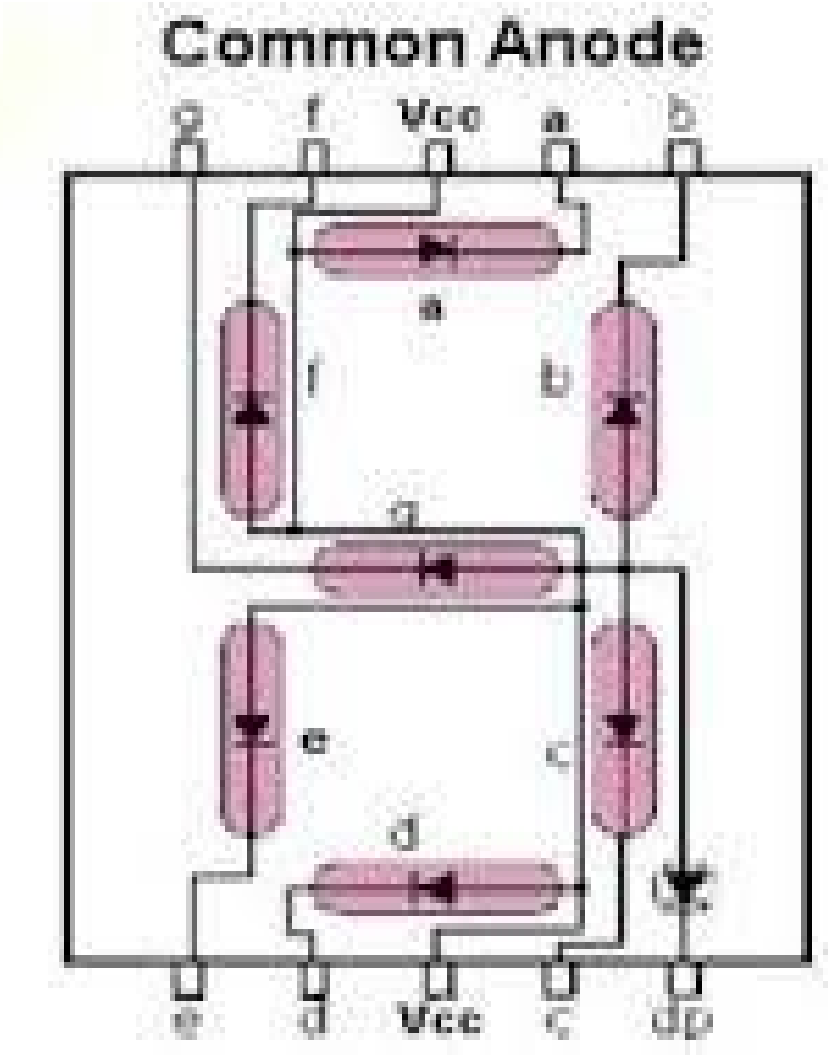






Display Configuration





Combinational Logic Circuits

- ✔ **Airthmetic Circuits:** (IC 7483) Adder & Subtractor, BCD Adder
- ✔ **Encoder/Decoder:** Basics of Encoder, decoder, comparison, (IC 7447) BCD to 7- Segment decoder/driver.
- ✔ **Multiplexer and Demultiplexer: Working, truth table and applications of Multiplexers** and Demultiplexers, **MUX tree, IC 74151 as MUX,** DEMUX tree, DEMUX as decoder, IC 74155 as DEMUX
- ✔ **Buffer:** Tristate logic, Unidirectional and Bidirectional buffer (IC 74LS244 and IC 74LS245)

Multiplexers

- ✔ Multiplexer is a circuit which has a number of inputs but only one output.
- ✔ Multiplexer is a circuit which transmits large number of information signals over a single line.
- ✔ Multiplexer is also known as “Data Selector” or MUX.

Necessity of Multiplexers

- ✔ In most of the electronic systems, the digital data is available on more than one lines. It is necessary to route this data over a single line.
- ✔ Under such circumstances we require a circuit which select one of the many inputs at a time.
- ✔ This circuit is nothing but a multiplexer. Which has many inputs, one output and some select lines.
- ✔ Multiplexer improves the reliability of the digital system because it reduces the number of external wired connections.

Advantages of Multiplexers

- ✔ It reduces the number of wires.
- ✔ So it reduces the circuit complexity and cost.
- ✔ We can implement many combinational circuits using Mux.
- ✔ It simplifies the logic design.
- ✔ It does not need the k-map and simplification.

Applications of Multiplexers

- ✔ It is used as a data selector to select one out of many data inputs.
- ✔ It is used for simplification of logic design.
- ✔ It is used in data acquisition system.
- ✔ In designing the combinational circuits.
- ✔ In D to A converters.
- ✔ To minimize the number of connections.

Block Diagram of Multiplexer

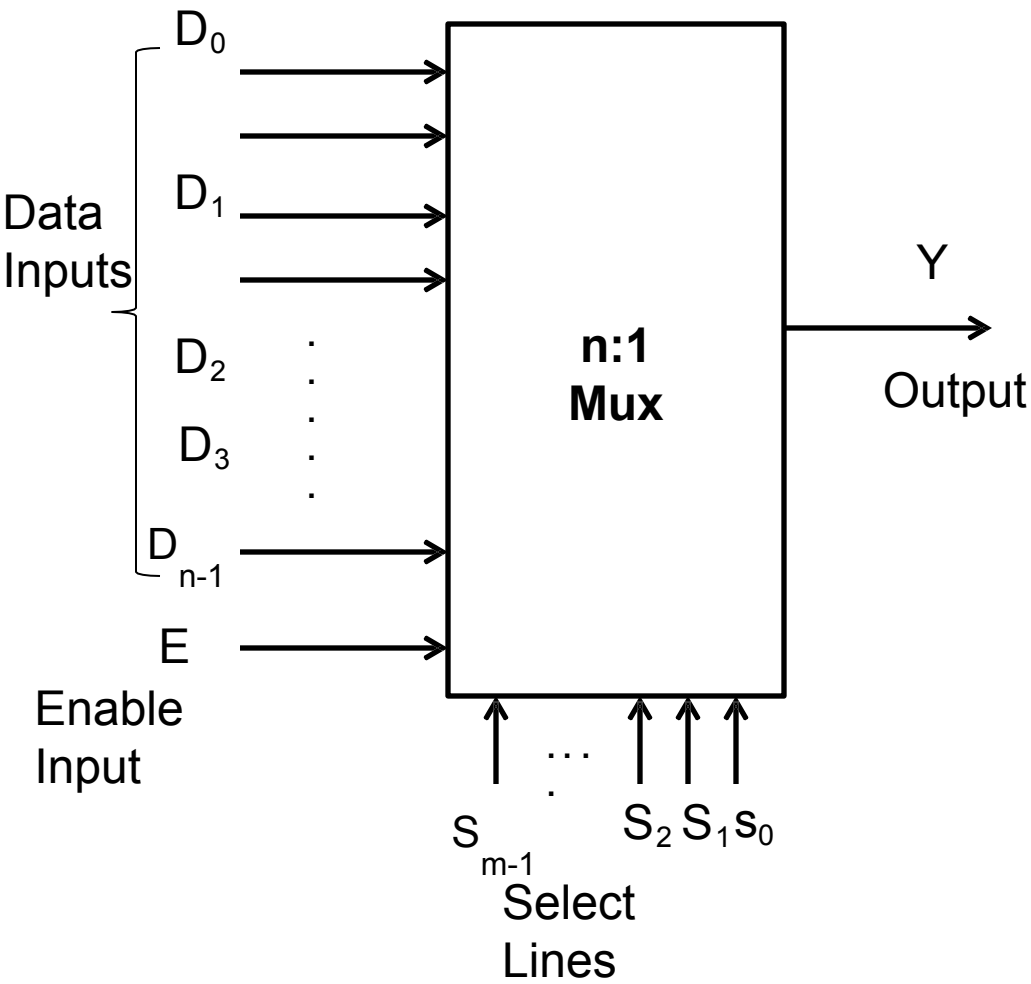


Fig. General Block Diagram

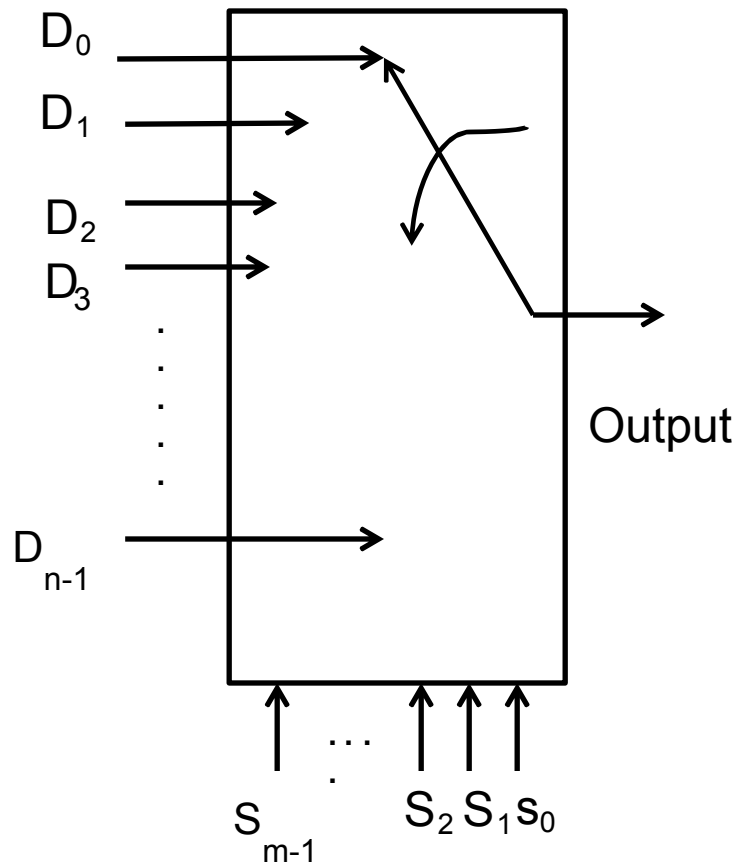


Fig. Equivalent Circuit

Relation between Data Input Lines & Select Lines

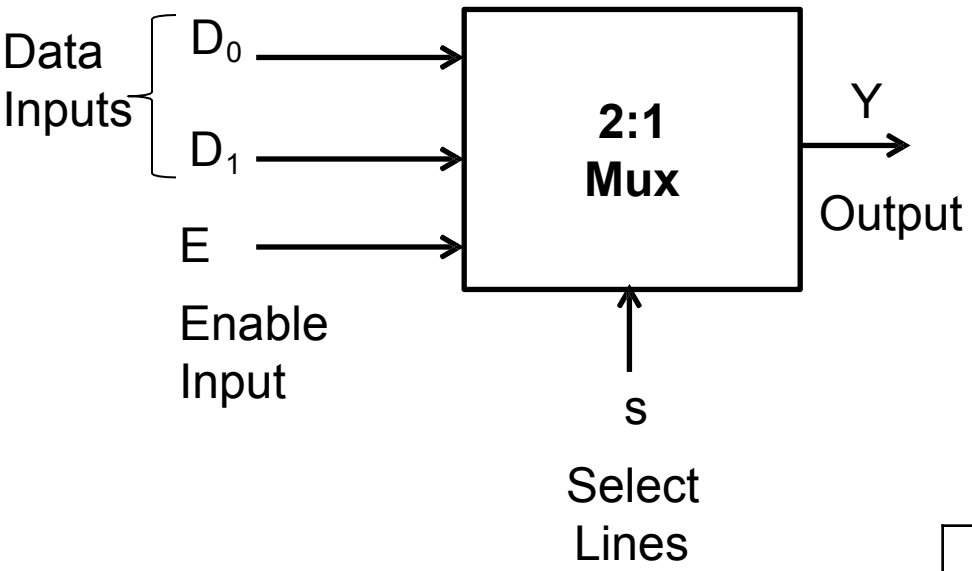
- In generalmultiplexer contains , n data lines, one output line and m select lines.
- Toselect n inputs we needm select lines such that $2^m=n$.

Types of Multiplexers

- ✔ 2:1 Multiplexer
- ✔ 4:1 Multiplexer
- ✔ 8:1 Multiplexer
- ✔ 16:1 Multiplexer
- ✔ 32:1 Multiplexer
- ✔ 64:1 Multiplexer

and so o
n……

2:1 Multiplexer

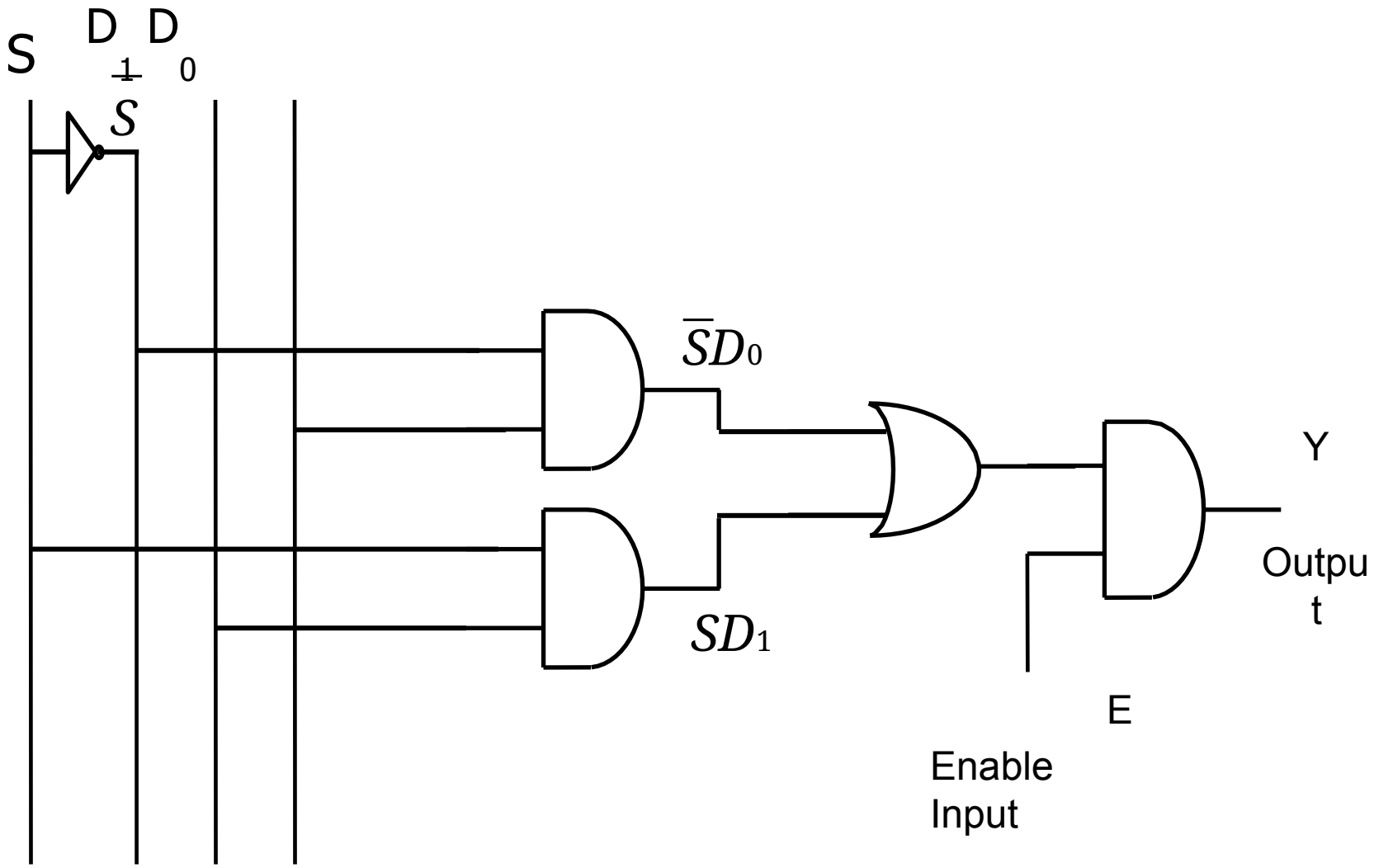


Block Diagram

Truth Table

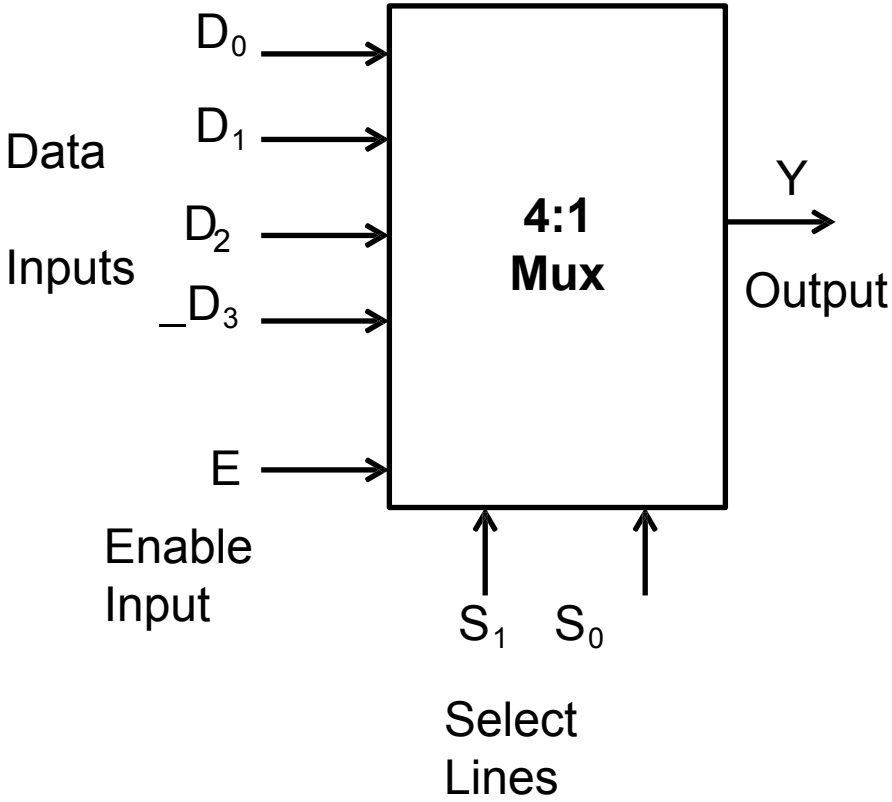
Enable i/p (E)	Select i/p (S)	Output (Y)
0	X	0
1	0	D_0
1	1	D_1

Realization of 2:1 Mux using gates



4:1 Multiplexer

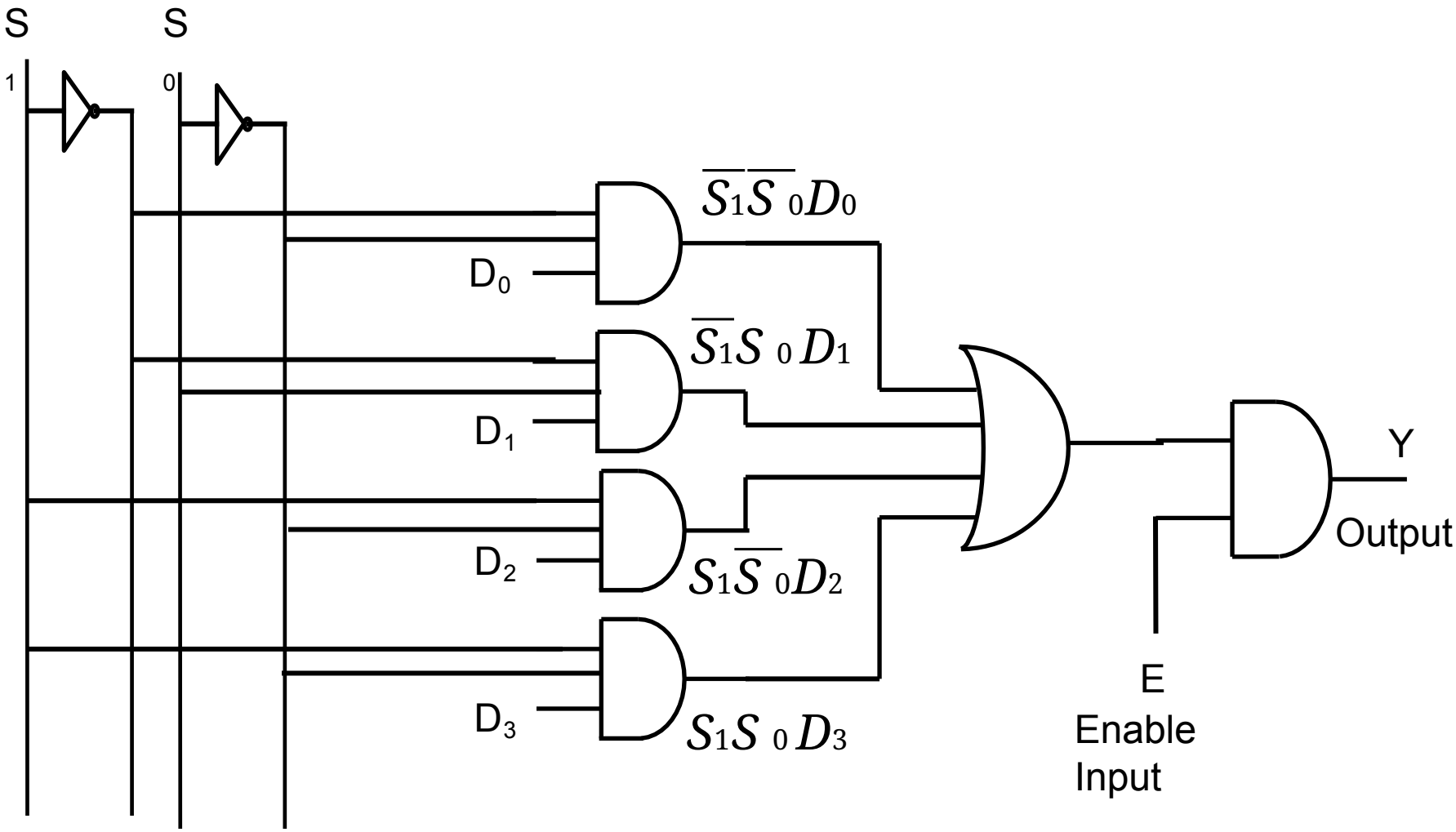
Block Diagram



Truth Table

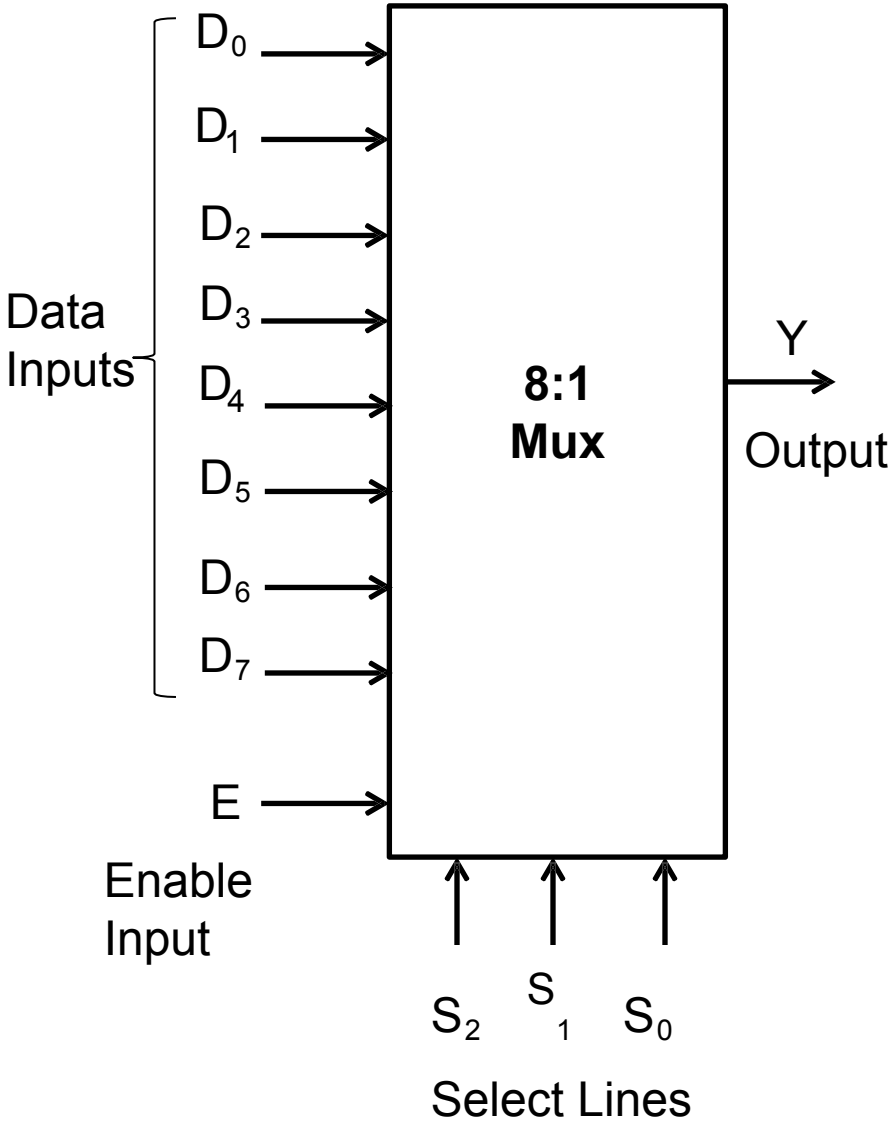
Enable i/p	Select i/p		Output
E	**S_1**	**S_0**	**Y**
0	X	X	0
1	0	0	D_0
1	0	1	D_1
1	1	0	D_2
1	1	1	D_3

Realization of 4:1 Mux using gates



8:1 Multiplexer

Block Diagram

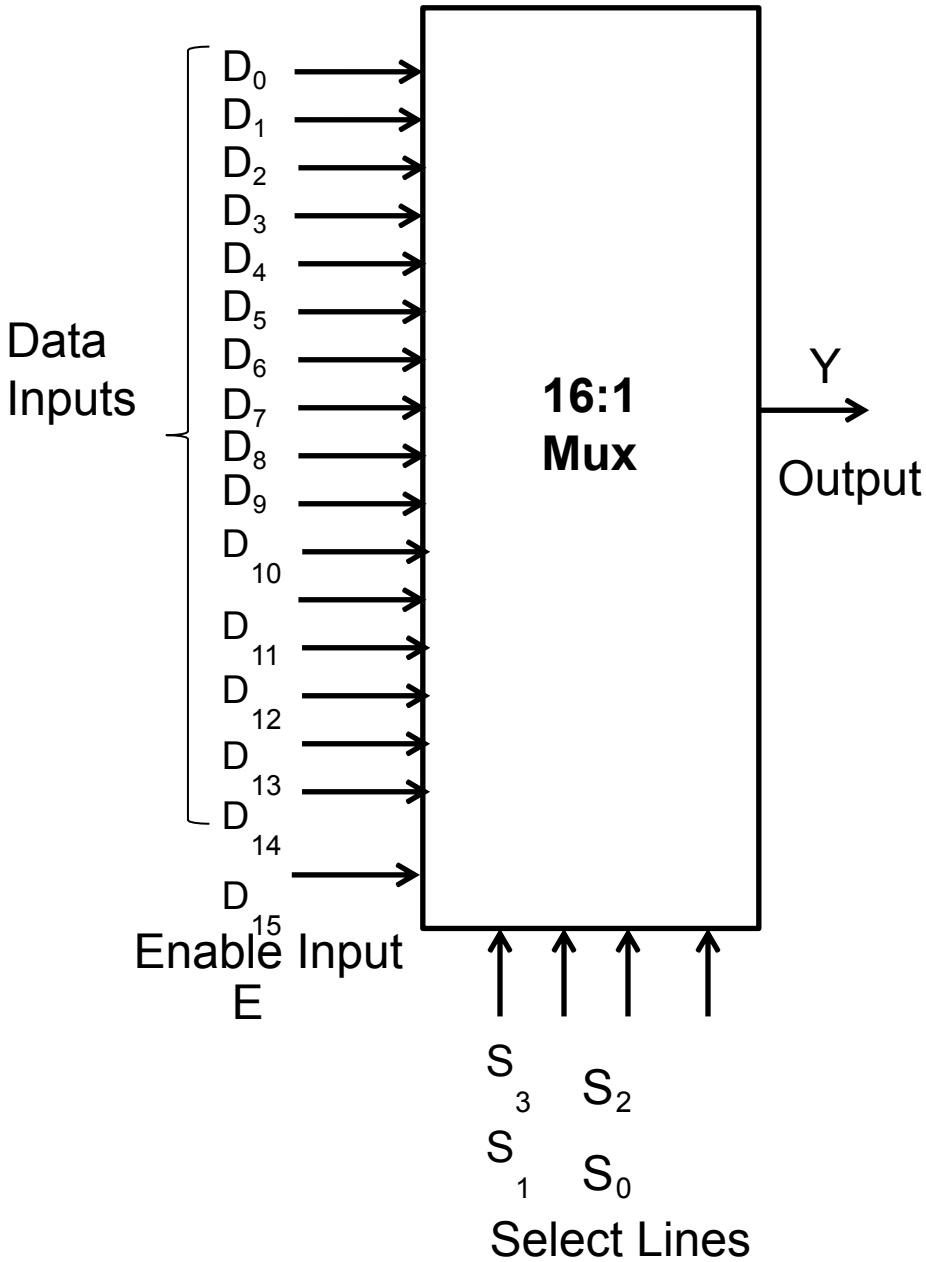


Truth Table

Enable i/p	Select i/p			Output
E	**S_2**	**S_1**	**S_0**	**Y**
0	X	X	X	0
1	0	0	0	D_0
1	0	0	1	D_1
1	0	1	0	D_2
1	0	1	1	D_3
1	1	0	0	D_4
1	1	0	1	D_5
1	1	1	0	D_6
1	1	1	1	D_7

16:1 Multiplexer

Block Diagram



16:1 Multiplexer

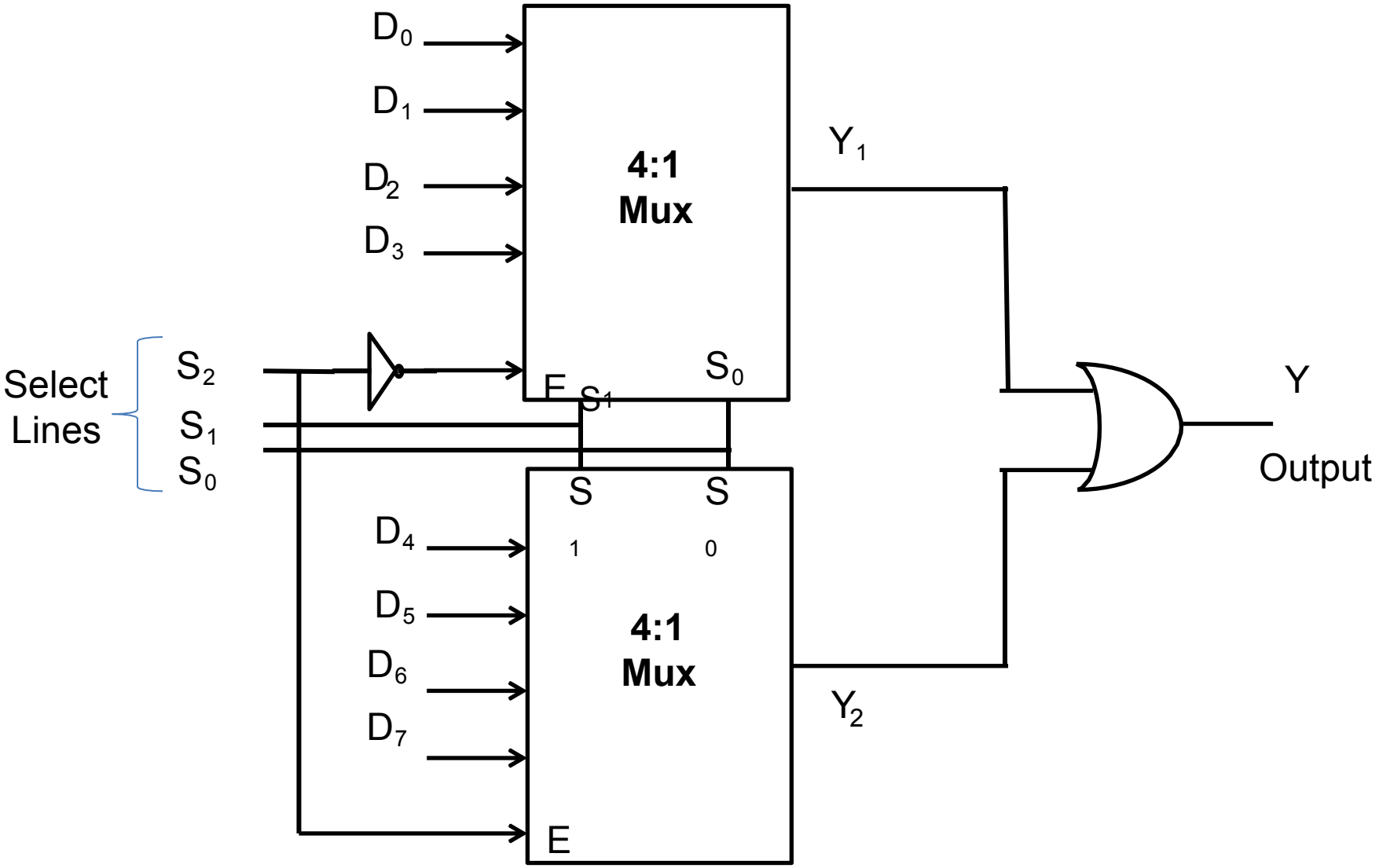
Truth Table

Enable	Select Lines				Output
E	S_3	S_2	S_1	S_0	Y
0	X	X	X	X	0
1	0	0	0	0	D_0
1	0	0	0	1	D_1
1	0	0	1	0	D_2
1	0	0	1	1	D_3
1	0	1	0	0	D_4
1	0	1	0	1	D_5
1	0	1	1	0	D_6
1	0	1	1	1	D_7
1	1	0	0	0	D_8
1	1	0	0	1	D_9
1	1	0	1	0	D_{10}
1	1	0	1	1	D_{11}
1	1	1	0	0	D_{12}
1	1	1	0	1	D_{13}
1	1	1	1	0	D_{14}

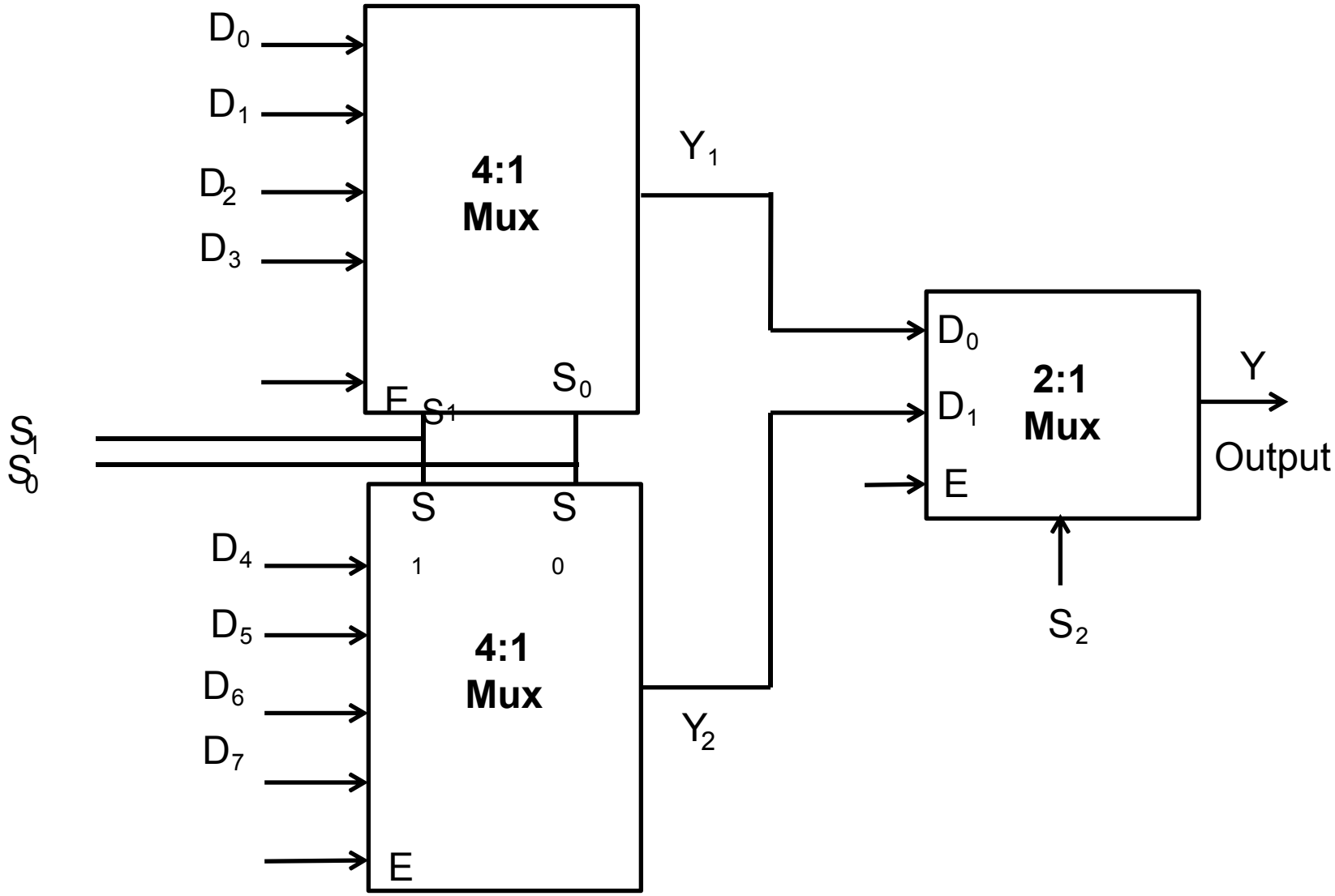
Mux Tree

- The multiplexers having more number of inputs can be obtained by cascading two or more multiplexers with less number of inputs. This is called as Multiplexer Tree.
- For example, 32:1 mux can be realized using two 16:1 mux and one 2:1 mux.

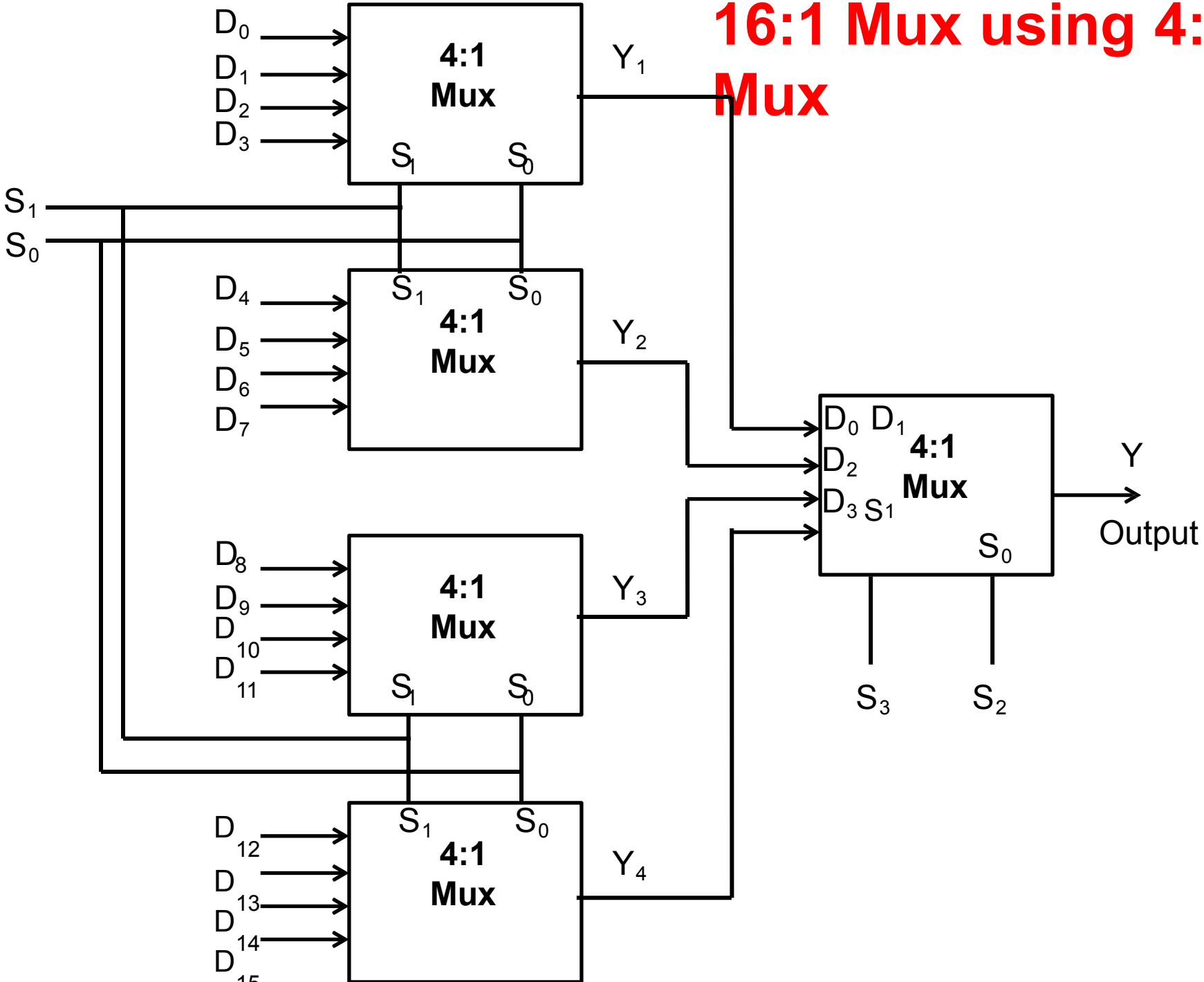
8:1 Multiplexer using 4:1 Multiplexer



8:1 Multiplexer using 4:1 Multiplexer



16:1 Mux using 4:1 Mux



Realization of Boolean expression using Mux

- ✔ We can implement any Boolean expression using Multiplexers.
- ✔ It reduces circuit complexity.
- ✔ It does not require any simplification

Example 1

Implement following Boolean expression using multiplexer

$$f(A, B, C) = \sum m(0, 3, 5, 6)$$

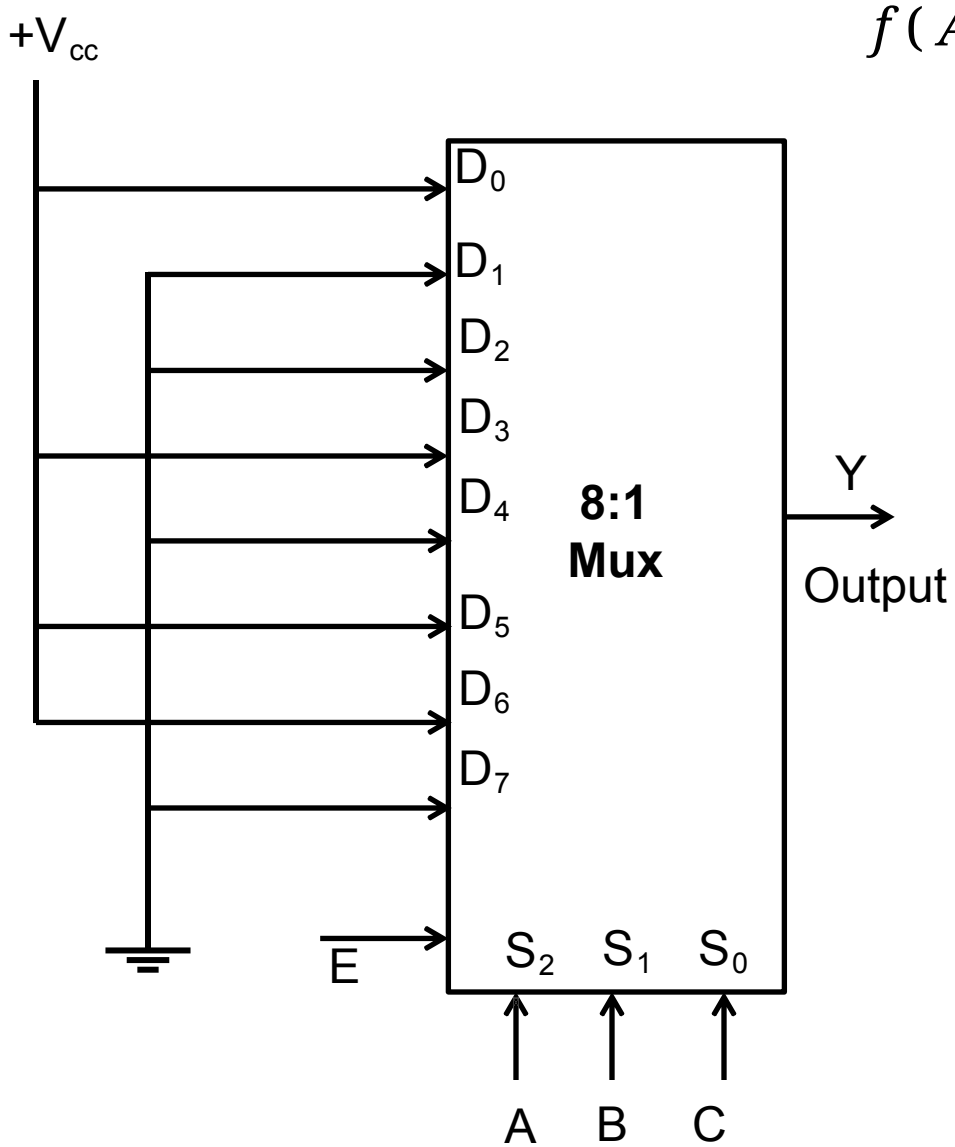
- ✔ Since there are three variables, therefore a multiplexer with three select input is required

  i.e. 8:1 multiplexer is required
- ✔ The 8:1 multiplexer is configured as below to implement given Boolean expression

Example 1

continue…..

$$f(A, B, C) = \sum m(0, 3, 5, 6)$$



Example 2

Implement following Boolean expression using multiplexer

$$f(A, B, C, D) = \sum m(0, 2, 3, 6, 8, 9, 12, 14)$$

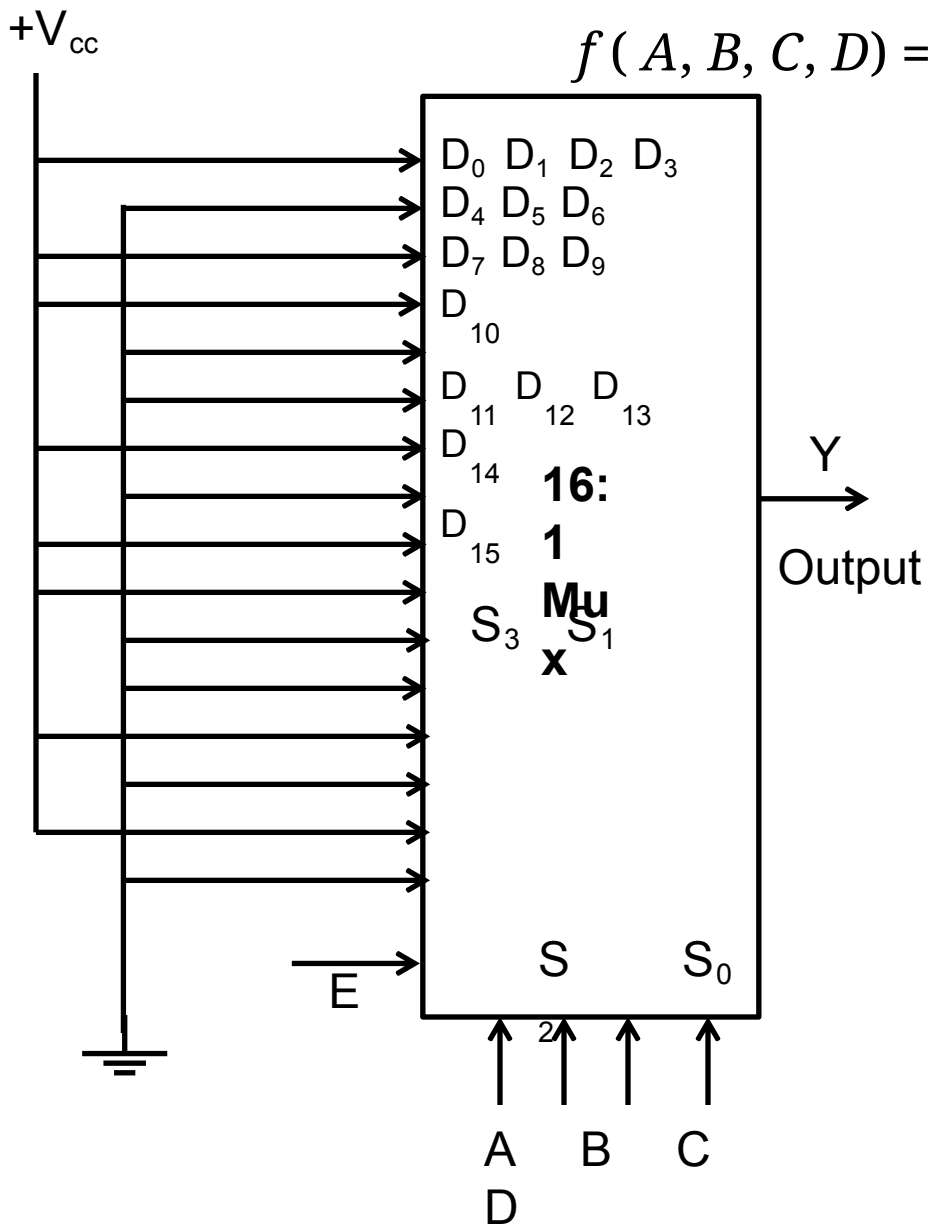
- ✔ Since there are four variables, therefore a multiplexer with four select input is required

  i.e. 16:1 multiplexer is required
- ✔ The 16:1 multiplexer is configured as below to implement given Boolean expression

Example 2

continue…..

$$f(A, B, C, D) = \sum m(0, 2, 3, 6, 8, 9, 12, 14)$$



Combinational Logic Circuits

- ✔ **Airthmetic Circuits:** (IC 7483) Adder & Subtractor, BCD Adder
- ✔ **Encoder/Decoder:** Basics of Encoder, decoder, comparison, (IC 7447) BCD to 7- Segment decoder/driver.
- ✔ **Multiplexer and Demultiplexer: Working, truth table and applications of** Multiplexers and **Demultiplexers,** MUX tree, IC 74151 as MUX, **DEMUX tree, DEMUX as decoder, IC 74155 as DEMUX**
- ✔ **Buffer:** Tristate logic, Unidirectional and Bidirectional buffer (IC 74LS244 and IC 74LS245)

De-multiplexer

- ✔ A de-multiplexer performs the reverse operation of a multiplexer i.e. it receives one input and distributes it over several outputs.
- ✔ At a time only one output line is selected by the select lines and the input is transmitted to the selected output line.
- ✔ It has only one input line, n number of output lines and m number of select lines.

Block Diagram of De-multiplexer

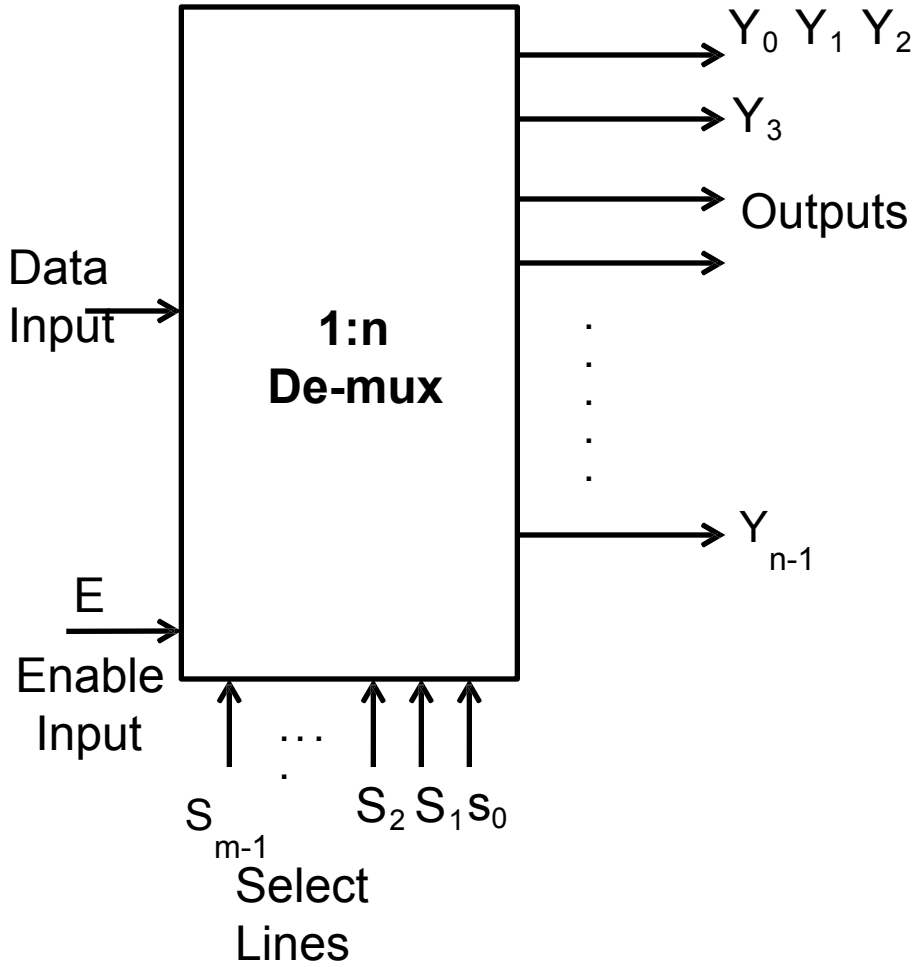


Fig. General Block Diagram

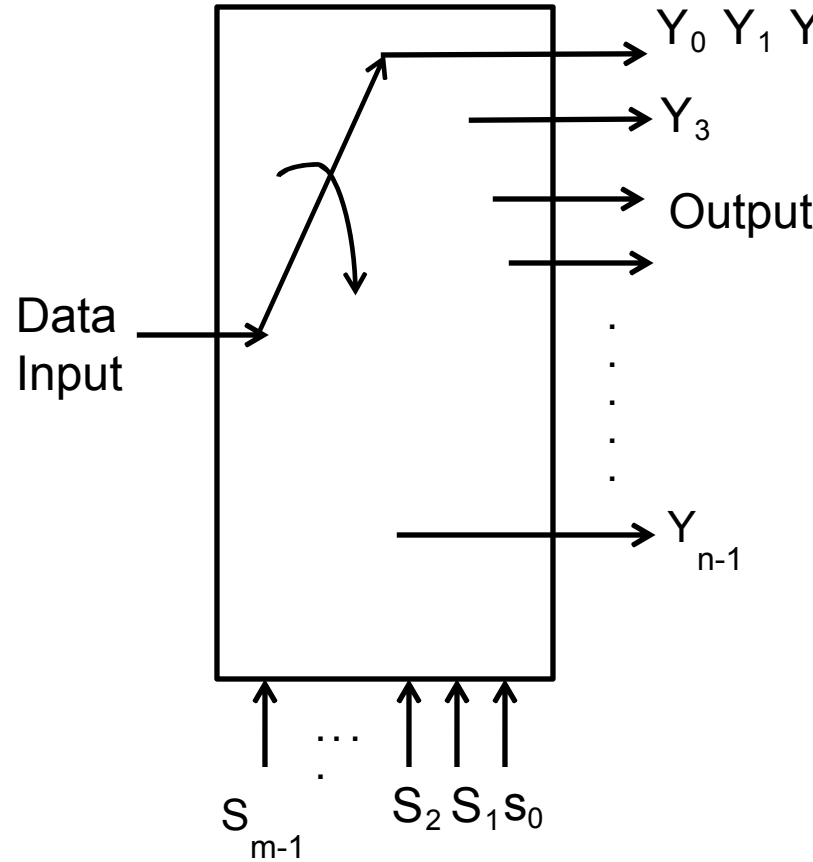


Fig. Equivalent Circuit

Relation between Data Output Lines & Select Lines

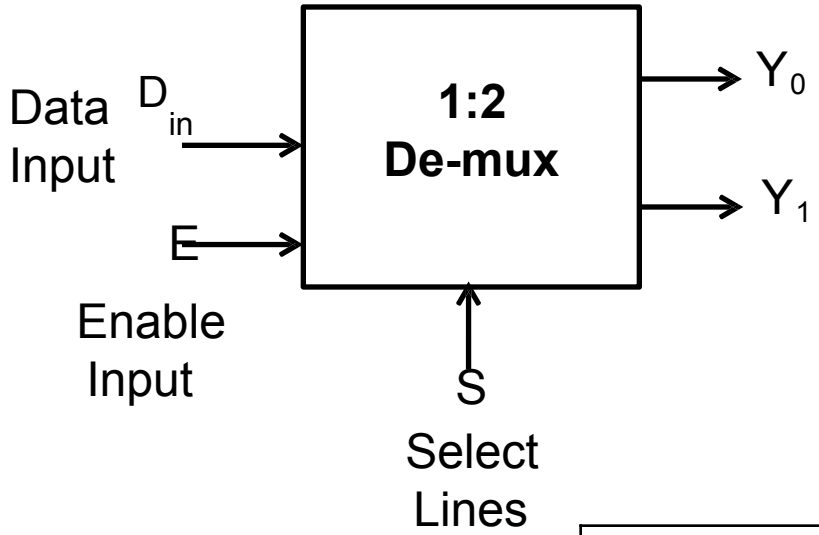
- ✔ In generalde-multiplexer contains , n output lines, one input line and m select lines.
- ✔ To select n outputs we need m select lines such that $n=2^m$.

Types of De-multiplexers

- ✔ 1:2 De-multiplexer
- ✔ 1:4 De-multiplexer
- ✔ 1:8 De-multiplexer
- ✔ 1:16 De-multiplexer
- ✔ 1:32 De-multiplexer
- ✔ 1:64 De-multiplexer

and so o n……

1: 2 De-multiplexer

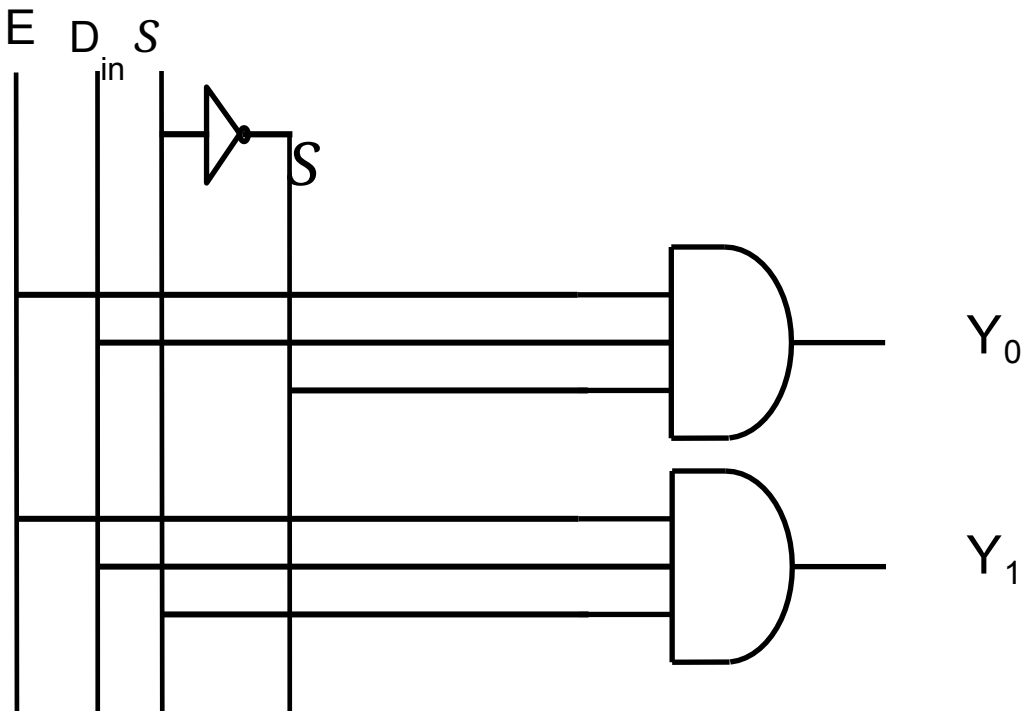


Block Diagram

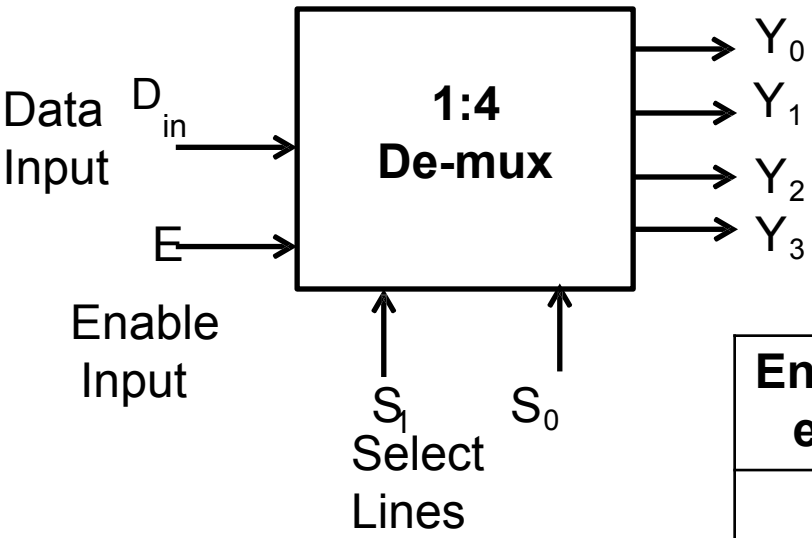
Truth Table

Enable i/p	**Select i/p**	**Outputs**	
E	**S**	**Y_0**	**Y_1**
0	X	0	0
1	0	D_{in}	0
1	1	0	D_{in}

1:2 De-mux using basic gates



1: 4 De-multiplexer

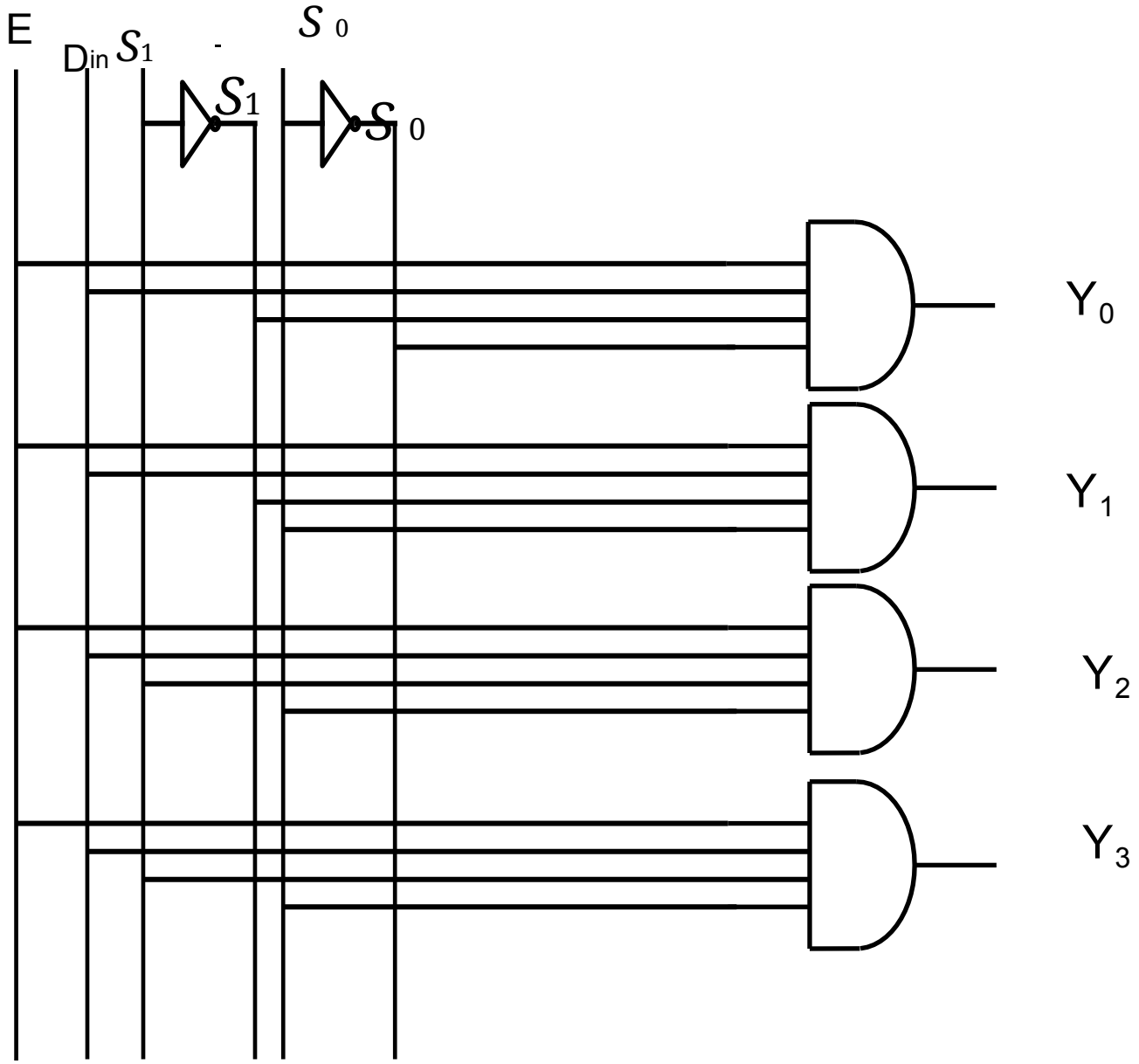


Block Diagram

Truth Table

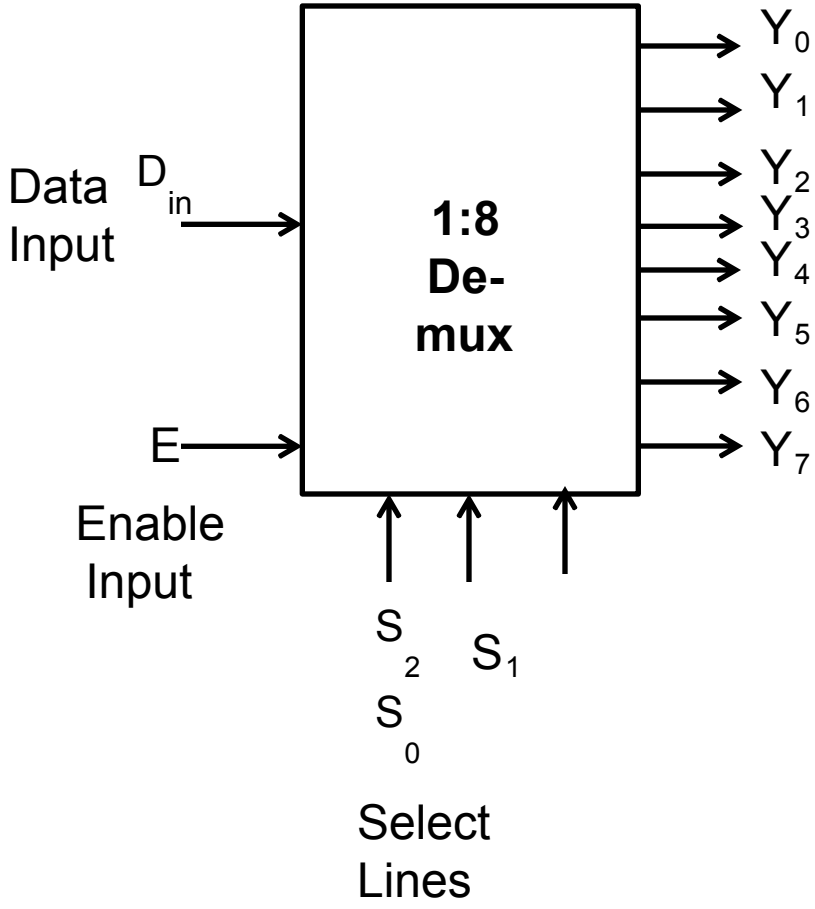
Enable i/p	Select i/p		Outputs			
E	**S_1**	**S_0**	**Y_0**	**Y_1**	**Y_2**	**Y_3**
0	X	X	0	0	0	0
1	0	0	D_{in}	0	0	0
1	0	1	0	D_{in}	0	0
1	1	0	0	0	D_{in}	0
1	1	1	0	0	0	D_{in}

1:4 De-mux using basic gates



1: 8 De-multiplexer

Block Diagram



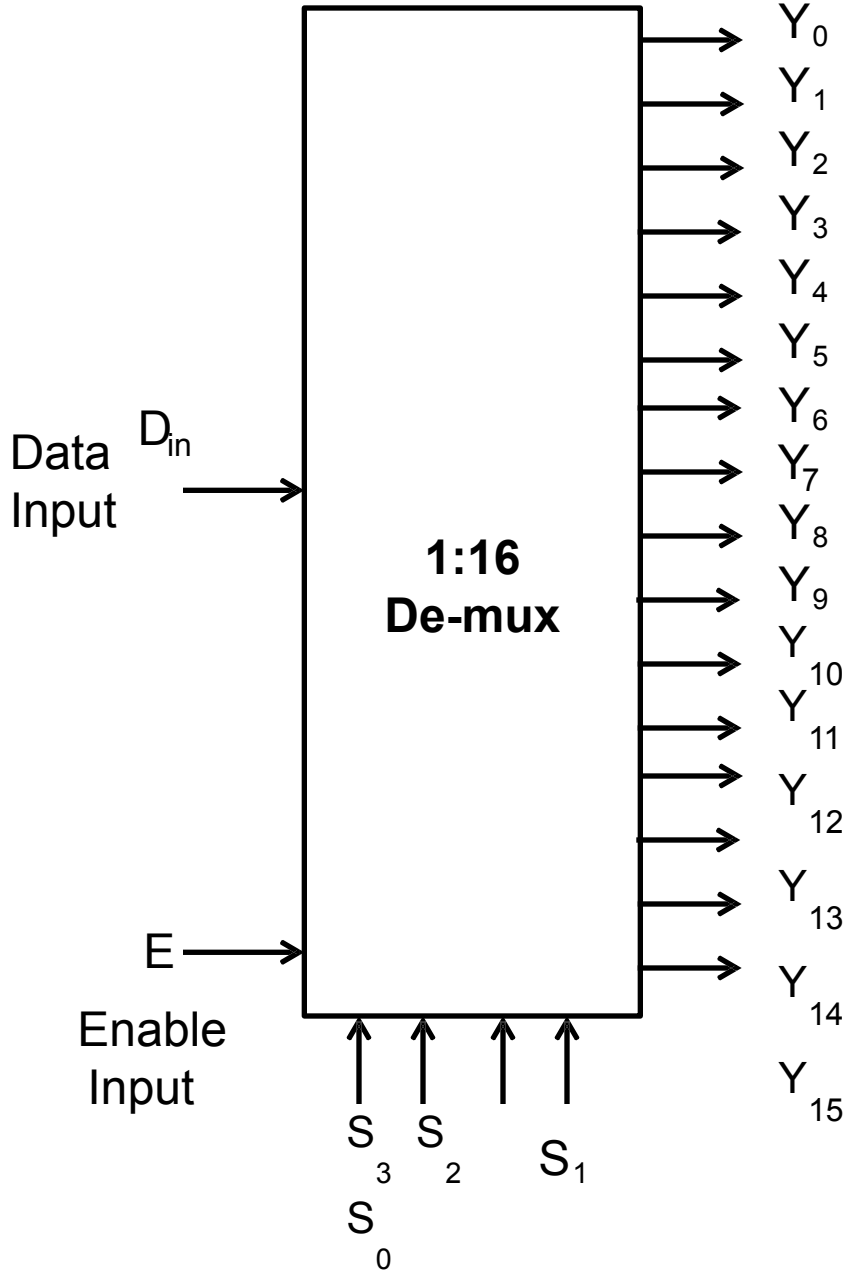
1: 8 De-multiplexer

Truth Table

Enable i/p	Select i/p			Outputs							
E	S_2	S_1	S_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	X	X	X	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	D_{in}
1	0	0	1	0	0	0	0	0	0	D_{in}	0
1	0	1	0	0	0	0	0	0	D_{in}	0	0
1	0	1	1	0	0	0	0	D_{in}	0	0	0
1	1	0	0	0	0	0	D_{in}	0	0	0	0
1	1	0	1	0	0	D_{in}	0	0	0	0	0
1	1	1	0	0	D_{in}	0	0	0	0	0	0

1: 16 De-multiplexer

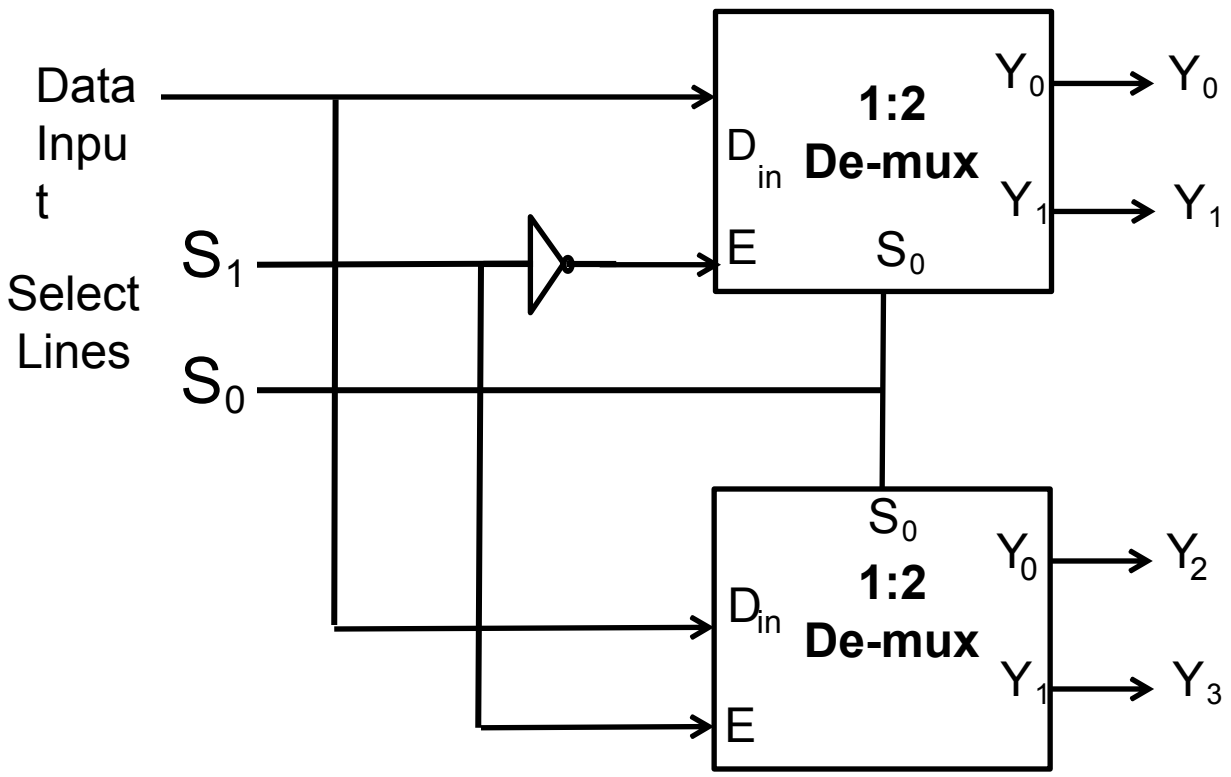
Block Diagram



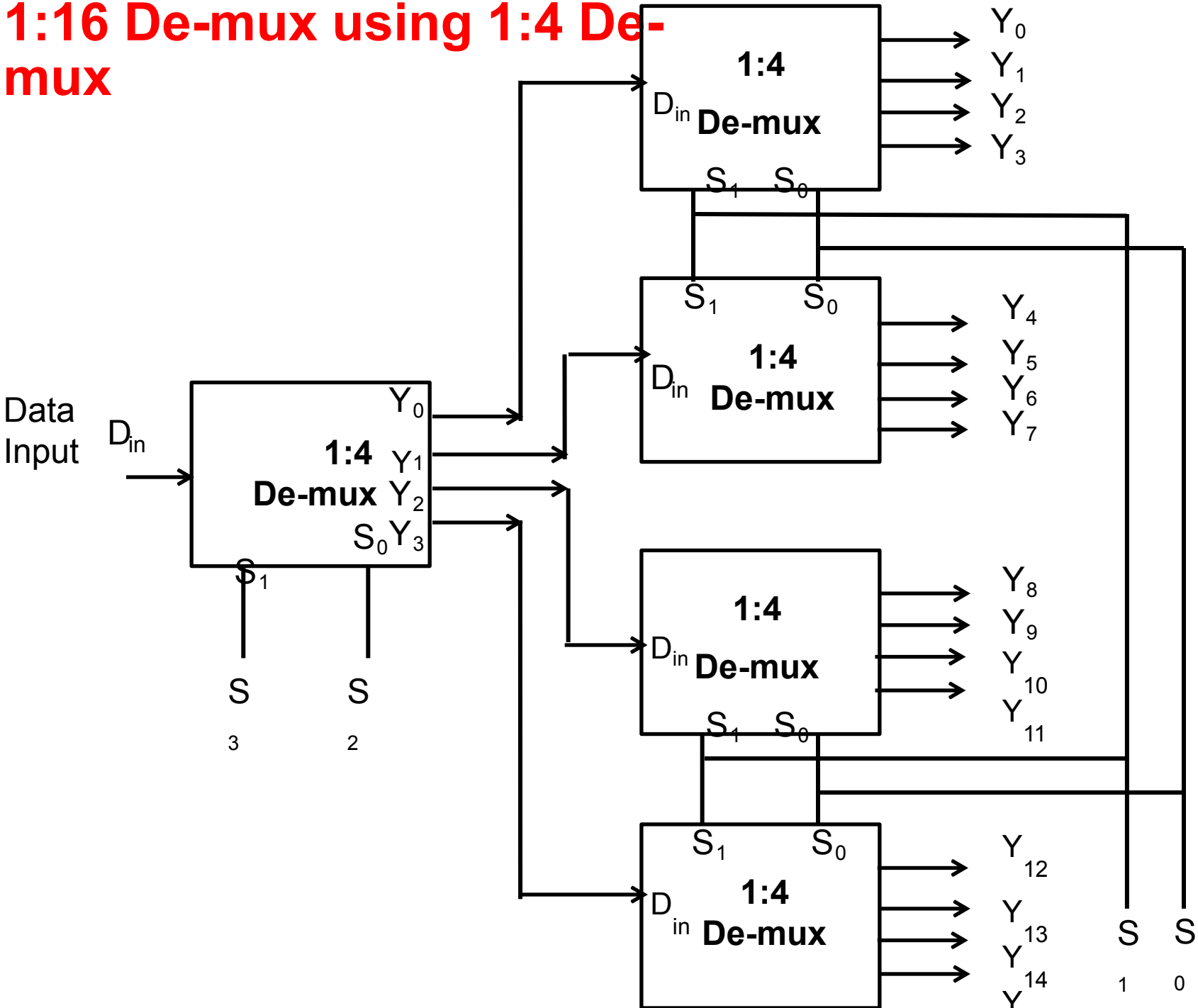
De-mux Tree

- ✔ Similar to multiplexer we can construct the de- multiplexer with more number of lines using de- multiplexer having less number of lines. This is call as “De-mux Tree”.

1:4 De-mux using 1:2 De-mux



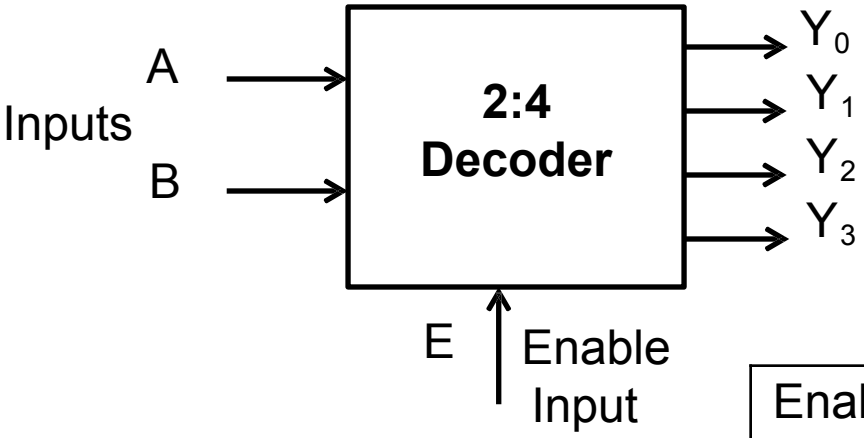
1:16 De-mux using 1:4 De-mux



Decoder

- ✔ Decoder is a combinational circuit.
- ✔ It converts n bit binary information at its input into a maximum of 2_n output lines.
- ✔ For example, if n=2 then we can design upto 2:4 decoder

2:4 Decoder



Block Diagram

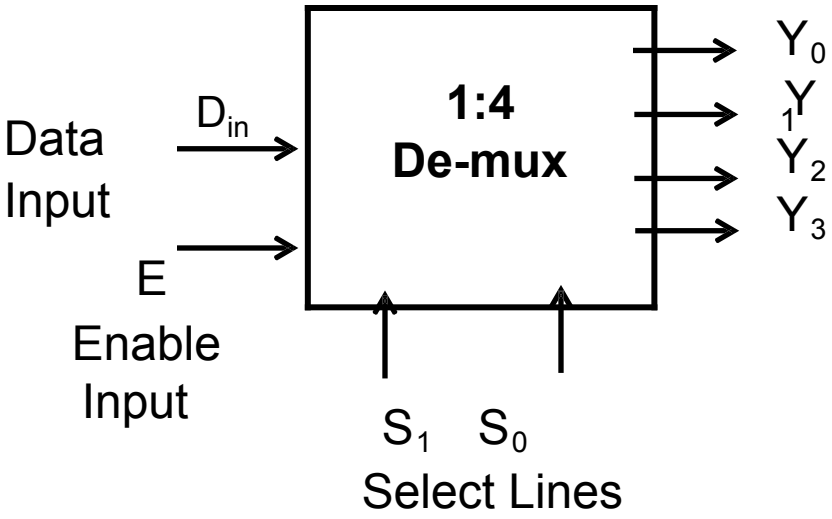
Truth Table

Enable i/p	Data Inputs		Outputs			
E	A	B	Y_0	Y_1	Y_2	Y_3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

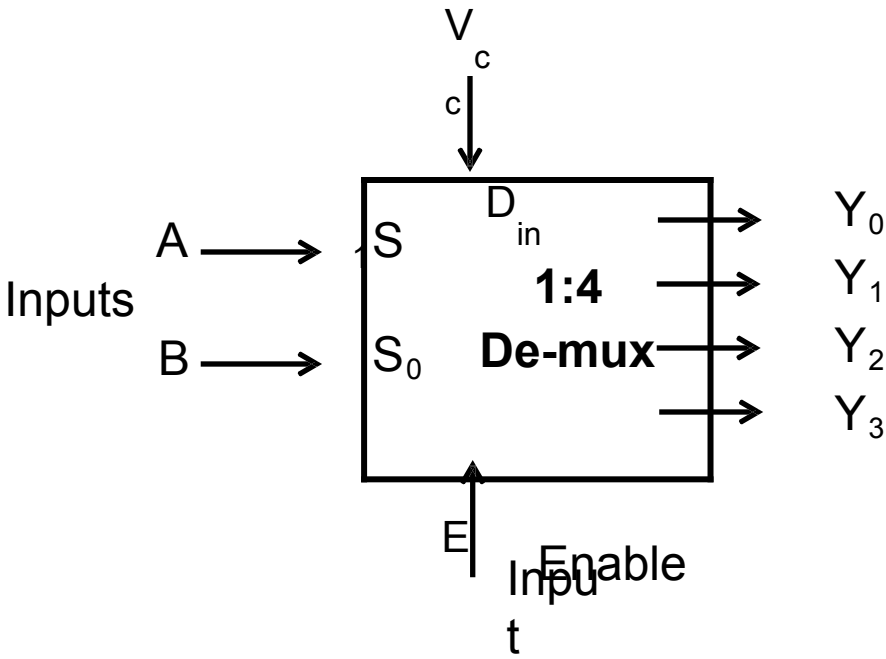
De-multiplexer as Decoder

- It is possible to operate a de-multiplexer as a decoder.
- Let us consider an example of 1:4 de-mux can be used as 2:4 decoder

1:4 De-multiplexer as 2:4 Decoder



1: 4 De-multiplexer



1: 4 De-multiplexer as 2:4 Decoder

Realization of Boolean expression using De-mux

- ✔ We can implement any Boolean expression using de-multiplexers.
- ✔ It reduces circuit complexity.
- ✔ It does not require any simplification

Example 1

Implement following Boolean expression using de-multiplexer

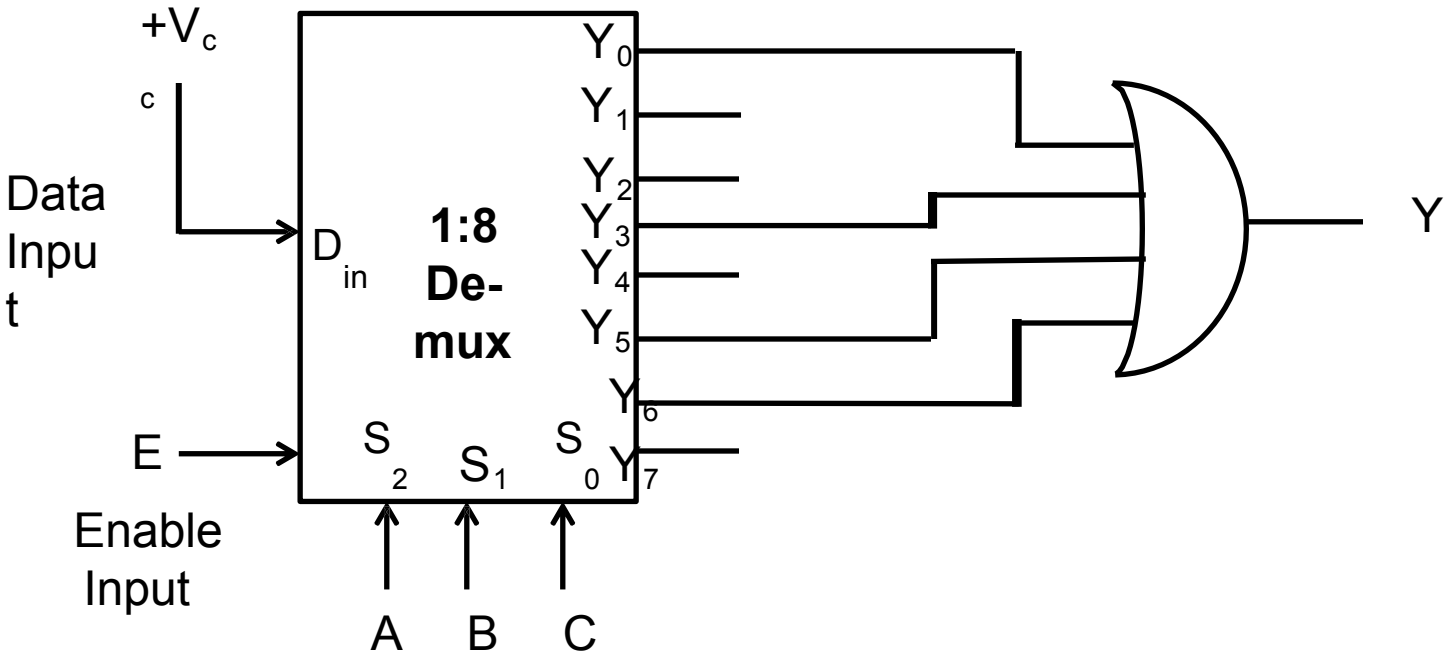
$$f(A, B, C) = \sum m(0, 3, 5, 6)$$

- ✔ Since there are three variables, therefore a de-multiplexer with three select input is required i. e. 1:8 de-multiplexer is required
- ✔ The 1:8 de-multiplexer is configured as below to implement given Boolean expression

Example 1

continue…..

$f(A, B, C) = \sum m(0, 3, 5, 6)$



Example 2

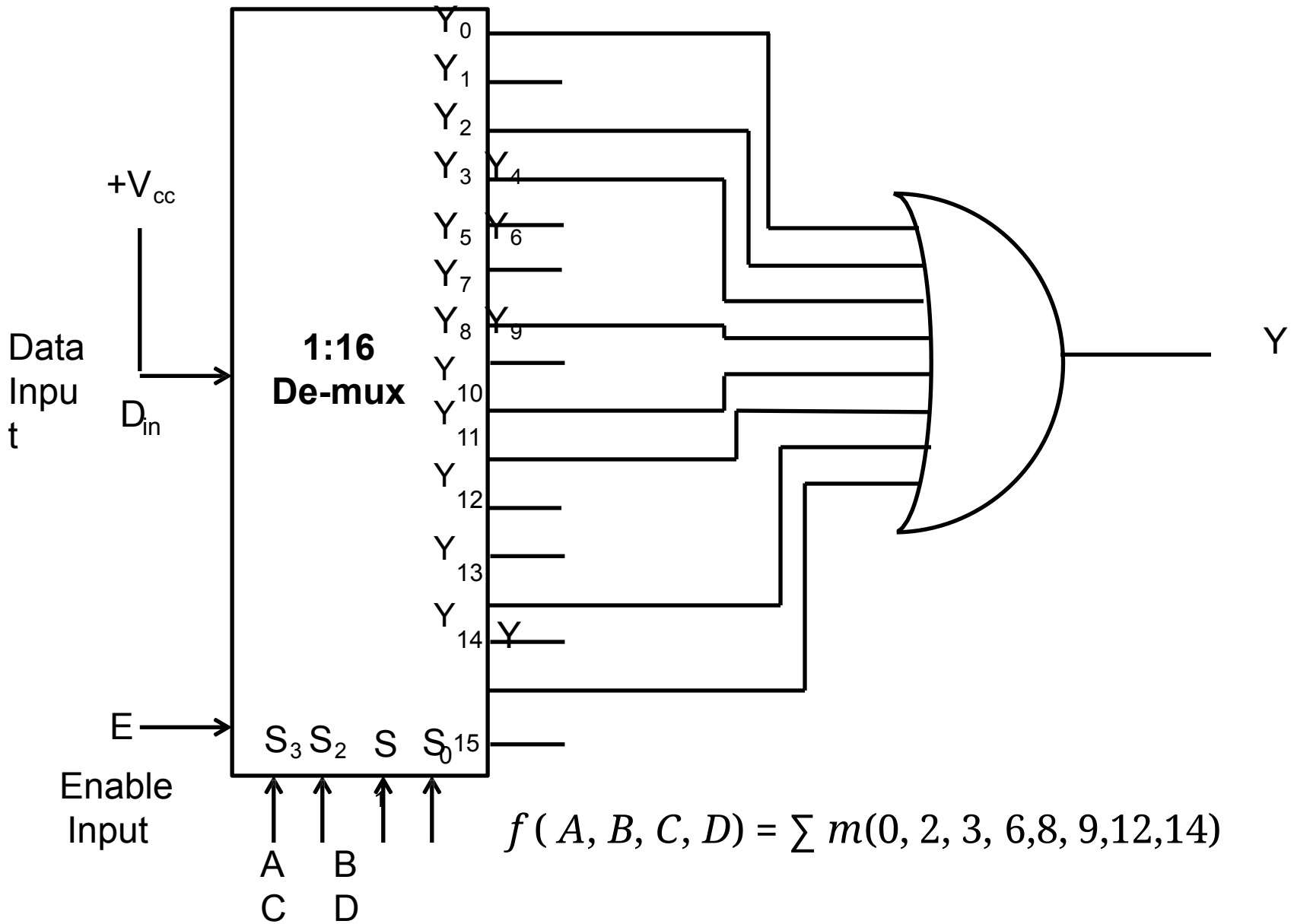
Implement following Boolean expression using de-multiplexer

$$f(A, B, C, D) = \sum m(0, 2, 3, 6, 8, 9, 12, 14)$$

- ✔ Since there are four variables, therefore a de-multiplexer with four select input is required i. e. 1:16 de-multiplexer is required
- ✔ The 1:16 de-multiplexer is configured as below to implement given Boolean expression

Example 2

continue…..



$$f(A, B, C, D) = \sum m(0, 2, 3, 6, 8, 9, 12, 14)$$

Multiplexer ICs

IC Number	Description	Output
IC 74157	Quad 2:1 Mux	Same as input
IC 74158	Quad 2:1 Mux	Inverted Output
IC 74153	Dual 4:1 Mux	Same as input
IC 74352	Dual 4:1 Mux	Inverted Output
IC 74151	8:1 Mux	Inverted Output
IC 74152	8:1 Mux	Inverted Output
IC 74150	16:1 Mux	Inverted Output

IC 74151 – General Description

- ✔ This Data Selector/Multiplexer contains full on-chip decoding to select one-of-eight data sources as a result of a unique three-bit binary code at the Select inputs.

- ✔ Two complementary outputs provide both inverting and non-inverting

  buffer operation.

- ✔ A Strobe input is provided which, when at the high level, disables all data inputs and forces the Y output to the low state and the

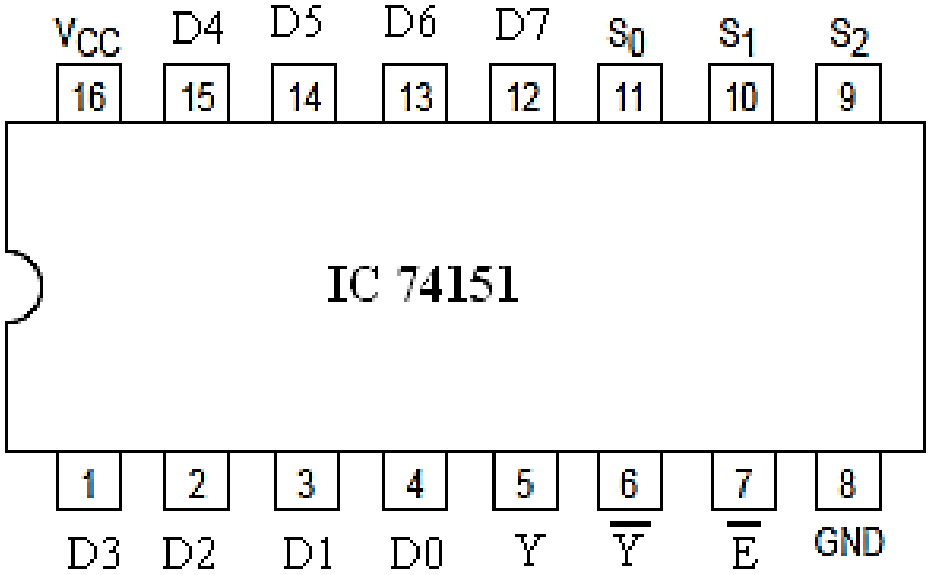
  $\overline{Y}$ output to the high state.

- ✔ The Select input buffers incorporate internal overlap features to ensure that select input changes do not cause invalid output transients.

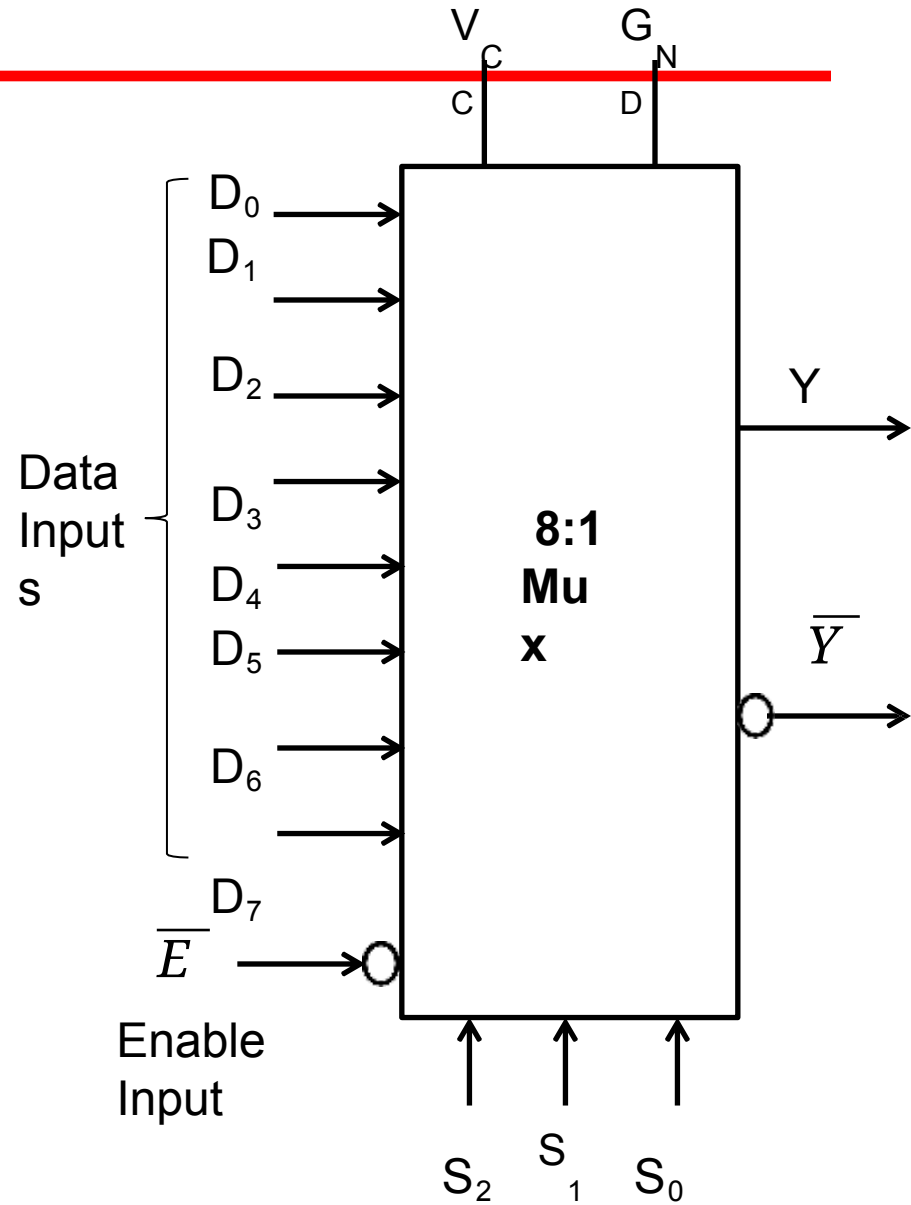
IC 74151 - Features

- ✔ Advanced oxide-isolated, ion-implanted Schottky TTL process
- ✔ Switching performance is guaranteed over full temperature and VCC supply range
- ✔ Pin and functional compatible with LS family counterpart
- ✔ Improved output transient handling capability

IC 74151 – Pin Diagram



Pin Diagram



Equivalent

De-multiplexer ICs

IC Number	Description
IC 74138	1:8 De-multiplexer
IC 74139	Dual 1:4 De-multiplexer
IC 74154	1:16 De-multiplexer
IC 74155	Dual 1:4 De-multiplexer

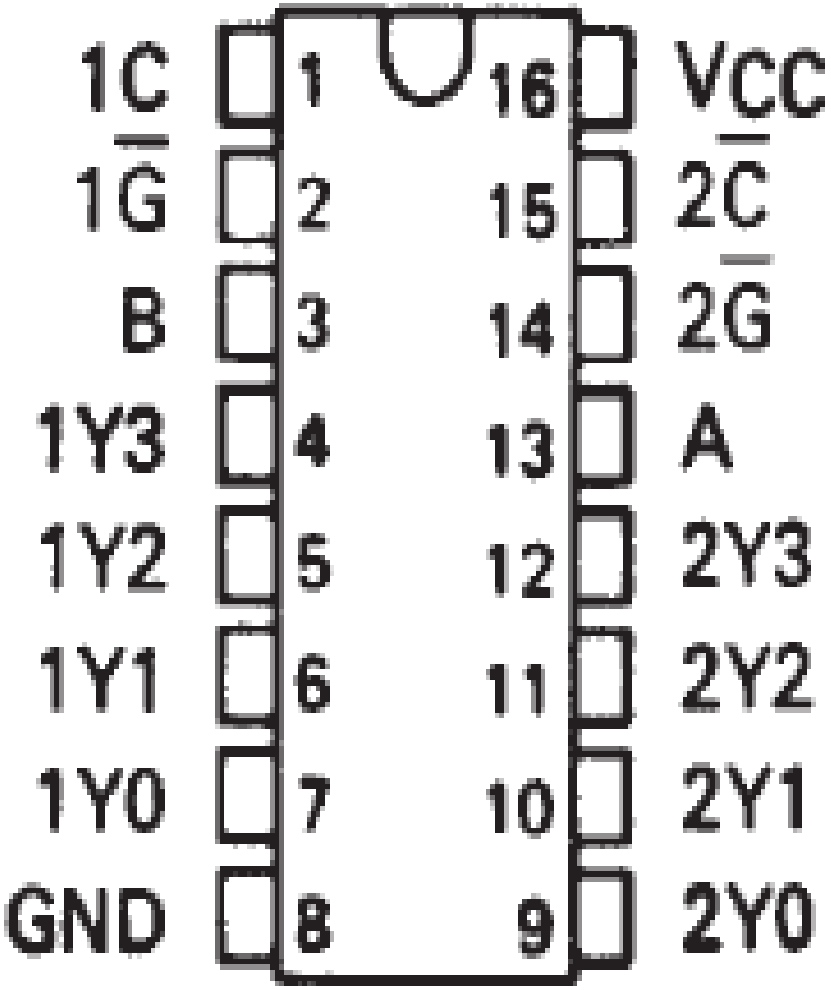
IC 74155 – General Description

- ✔ These monolithic TTL circuits feature dual 1 line to 4 line de-multiplexers with individual strobes and common binary address inputs in a single 16 pin package.
- ✔ The individual strobes permit activating or inhibiting each of the 4-bit sections as desired.

IC 74155 - Features

- ✔ Input clamping diodes simplify system design.
- ✔ Choice of outputs : Totem pole (‘LS155A) or open collector (‘LS156).
- ✔ Individual strobes simplify cascading for decoding or de- multiplexing larger words.
- ✔ Applications:
  - Dual 2 to 4 Line Decoder
  - Dual 1: 4 De-multiplexer
  - 3 to 8 line Decoder
  - 1 to 8 line de-multiplexer

IC 74155 – Pin Diagram



IC 74154 – General Description

- The M74HC154 is an high speed CMOS 4 TO 16 LINE DECODER/ DEMULTIPLEXER fabricated with silicon gate C2MOS technology.
- A binary code applied to the four inputs (A to D) provides a low level at the selected one of sixteen outputs excluding the other fifteen outputs, when both the strobe inputs, G1 and G2, are held low.
- When either strobe input is held high, the decoding function is inhibited to keep all outputs high.
- The strobe function makes it easy to expand the decoding lines through cascading, and simplifies the design of address decoding circuits in memory control systems.

IC 74154 - Features

- ✔ HIGH SPEED: t_{PD} = 16ns (TYP.) at VCC = 6V
- ✔ LOW POWER DISSIPATION: I_{CC} = 4mA(MAX.) at T_A=25°C
- ✔ HIGH NOISE IMMUNITY: $V_{NIH} = V_{NIL}$ = 28 % V_{CC} (MIN.)
- ✔ SYMMETRICAL OUTPUT IMPEDANCE: $|I_{OH}| = I_{OL}$ = 4mA (MIN)
- ✔ BALANCED PROPAGATION DELAYS: t_{PLH} @ t_{PHL}
- ✔ WIDE OPERATING VOLTAGE RANGE: V_{CC} (OPR) = 2V to 6V
- ✔ PIN AND FUNCTION COMPATIBLE WITH 74

IC 74154 – Pin Diagram