3

COMBINATIONAL LOGIC CIRCUITS AND ITS DESIGN

The digital system consist of two types of circuits, they are :

(*i*) Combinational circuits
(*ii*) Sequential circuits.

3.1 COMBINATIONAL CIRCUITS

A combinational circuit consists of logic gates, where outputs at any instant are determined only by the present combination of inputs without regard of previous inputs or previous state of outputs e.g. Multiplexers, Adders, Subtractors.

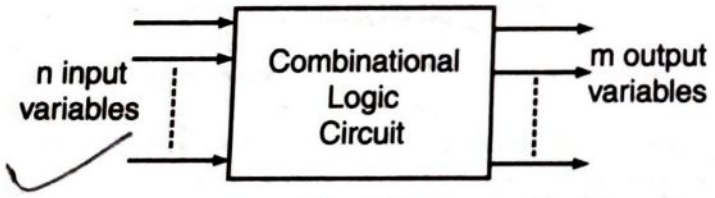

Fig. 3.1.

A combinational circuit consists of input variables, logic gates, and output variables as shown figure 3.1. The logic gates accepts signals from inputs and output signals are generated according to the logic circuits employed in it. For a number of input variables to a combinational circuit, 2^n possible combinations of binary input states are possible. For each possible combination, there is one and only one possible output combination. A combinational logic circuit can be described by m Boolean functions and each output can be expressed in terms of n input variables.

3.2 SEQUENTIAL CIRCUITS

The logic circuits where outputs at any instant of time depend on the present inputs as well as on the past outputs are called sequential circuits.

In This Chapter

- Combinational Circuits
- Sequential Circuits
- Analysis Procedure
- Multiplexers or Data Selectors
- Applications of Multiplexers
- Cascading of Multiplexers
- Demultiplexer/Data Distributors
- Block Representation of Demultiplexer
- Cascading of Demultiplexers
- Adders
- Subtractors
- Half Subtractor
- Full Subtractor
- Realization of Logic Expression of D and B by Gates
- Parallel Binary Adders
- Encoders
- Code Converters
- Priority Encoder
- Decoder
- Magnitude Comparator
- BCD to Seven Segment Decoder
- Comparison Between Demultiplexer & Decoder
- BCD Adder/Subtracts

They contain logic gates as well as memory elements. In sequential circuits, the output signals are feed back to the inputs side. Block diagram of a sequential circuit is as given :

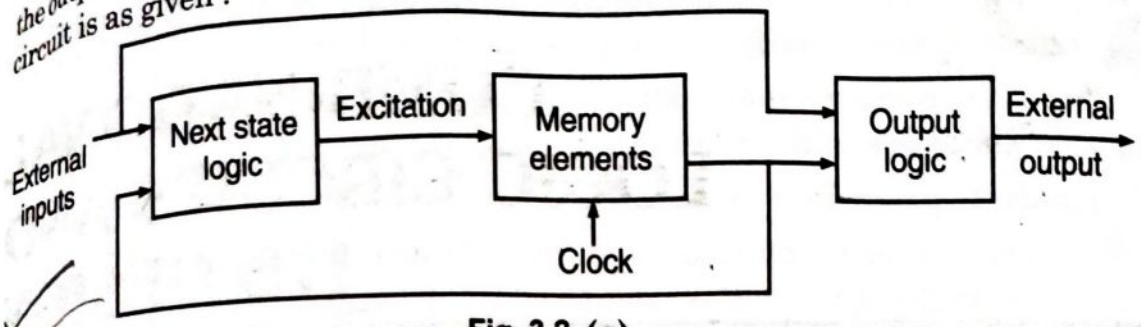

Fig. 3.2 (*a*)

From the figure 3.2 (*a*) we find that it consists of a combinational circuits which accept digital signals from external inputs and from outputs of memory elements and generates signals for external outputs and for inputs to memory element known as excitation.

A memory element is a medium in which one bit of information can be stored and thereafter its contents can be replaced by new value. The contents of memory elements can be changed by the outputs of combinational circuits which is connected to its input.

In other way the model of sequential circuit is represented as follows in figure 3.2 (*b*).

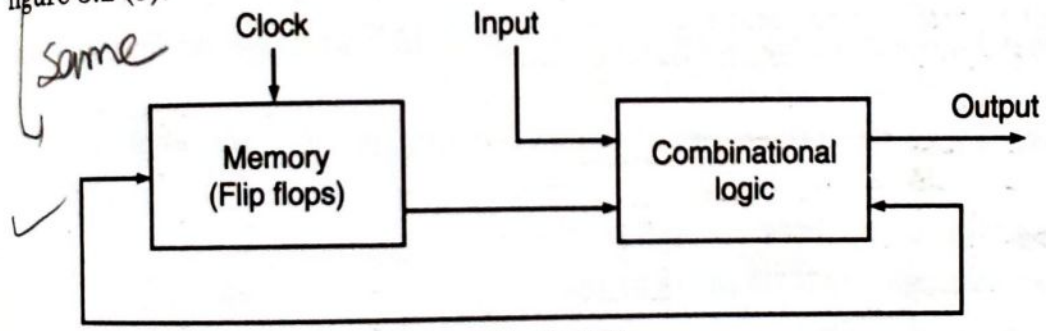

Fig. 3.2 (*b*)

During one clock cycle following steps are performed :

(*i*) Flip flops (may) change state.
(*ii*) Changes propagate through combinational logic.
(*iii*) Combinational logic gates and flip flops inputs became stable.

This cycle repeats indefinitely as long as the clock is running.

3.3 ANALYSIS PROCEDURE

Combinational circuits are made up of logic gates and output depends only present combination of inputs. Operation of combinational circuit can be specified by using boolean functions. These circuits don't have any feedback path or storage elements.

Any combinational circuit can be analyzed by knowing about boolean function it is going to perform, its truth table and its functional operation.

Analysis steps :

1. **Verification of logic diagram :** Whether it is combinational or sequential logic circuit.

2. Find the output boolean function from the verified logic diagram.
3. Obtain the truth table for the provided logic diagram.

By the manual analysis, we can come to know that whether functionality of the verified circuit actually meet the desired specifications or not.

Instead of doing manual analysis of a combinational circuit, one can also go for logic simulation and verification by using verilog HDL.

Analysis procedure in detail:

- Verification of combinational circuit is done by checking that, there is no feedback path present in the given logic diagram as well as circuit comprises of only logic gates. No storage element is present.
- Procedure for finding the output boolean function :

Step 1 : Take arbitrary symbols for marking of output of only those gates which are function of input variables.

Step 2 : Then find out the boolean functions corresponding to those arbitrary symbols.

Step 3 : Now take another set of arbitrary symbols for marking of output of only those gates which are function of previously defined arbitrary symbols. Now find out the boolean expressions for them also.

Step 4 : Keep on repeating the above procedure until we get final outputs of provided logic diagram.

Step 5 : Find out the overall boolean function based on the whole analysis done.

Let's take an example to explain this –

Take the logic diagram given below

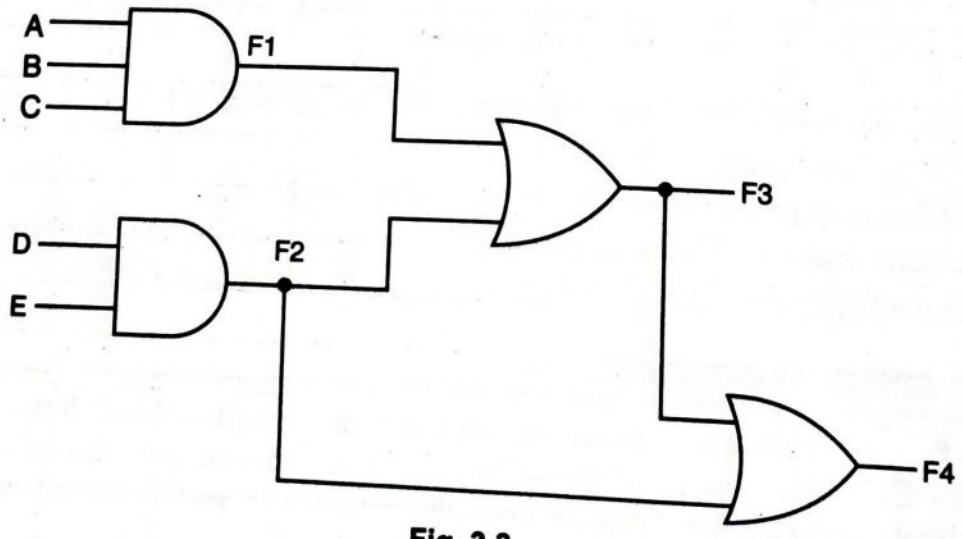

Fig. 3.3.

Step 1 : Taking arbitrary symbols F1 and F2. F1 is function of input variably A, B, C and F2 is function of input variables D, E.

Step 2 : $F1 = A \,.\, B \,.\, C$

$F_2 = D \,.\, E$

We have determined the boolean functions corresponding to defined arbitrary symbols F1 and F2.

Step 3 : Taking arbitrary symbols F3 which is function of already defined arbitrary symbols F1 and F2. Now, we determine the boolean function corresponding to $F3$. $F3 = F1 + F2$

Step 4 : Taking arbitrary symbol F4 which is function of already defined arbitrary symbols $F2$ and $F3$. Now we define boolean function corresponding to $F4$.

$$\begin{aligned} F4 &= F3 + F2 \\ &= F1 + F2 + F2 \\ &= F1 + F2 \\ &= ABC + DE \end{aligned}$$

Analysis can be done by determination of truth table directly from the logic diagram as–

Step 1 : Find out the number of input variables in the provided logic diagram. For n inputs, there are 2^n possible input combinations. We write those combination in n columns from (0) – $(2^n - 1)$.

Step 2 : Obtain the truth table for outputs of those gates which are function of only input variables and those outputs are assigned with some arbitrary labels.

Step 3 : Now obtain the truth table for the output of those gates which are function of already assigned arbitrary labels.

Step 4 : Above procedure goes on repeating until we come to final output.

Lets draw the truth table of provided logic diagram which was analyzed for determination of boolean function earlier.

A	B	C	D	E	F1	F2	F3	F4
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	0	1	1	0	1	1	1
0	0	1	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0	0
0	0	1	1	1	0	1	1	1
0	1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	0
0	1	0	1	1	0	1	1	1
0	1	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0	0
0	1	1	1	1	0	1	1	1

A	B	C	D	E	F1	F2	F3	F4
1	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0	0
1	0	0	1	1	0	1	1	1
1	0	1	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0
1	0	1	1	0	0	0	0	0
1	0	1	1	1	0	1	1	1
1	1	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0
1	1	0	1	0	0	0	0	0
1	1	0	1	1	0	1	1	1
1	1	1	0	0	1	0	1	1
1	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	1

3.4 MULTIPLEXERS OR DATA SELECTORS

Multiplexer is a combinational circuit that selects binary information from one of the many input channels and transmits to a single output line. That is why the multiplexer are also called as *Data Selectors*. The selection of particular input is controlled by select lines.

It is also considered as a multiple Input, Single Output Switch. In other words Multiplexer means "many to one". "Many" stands for the input combinations and "one" shows the output state. In telecommunication, a multiplexer is a device that combines several input information signals into one output signal, which carries several communication channels, by means of some multiplexing techniques.

A digital multiplexer of 2^n input channels can be controlled by a numbers of select lines and input line is selected according to the bit combinations of select lines.

Let us suppose, there are "*M*" inputs then relation of multiplexer with respect to select lines is given as :

$$M = 2^N$$

where N is the number of address lines or select lines and *M* is the number of inputs

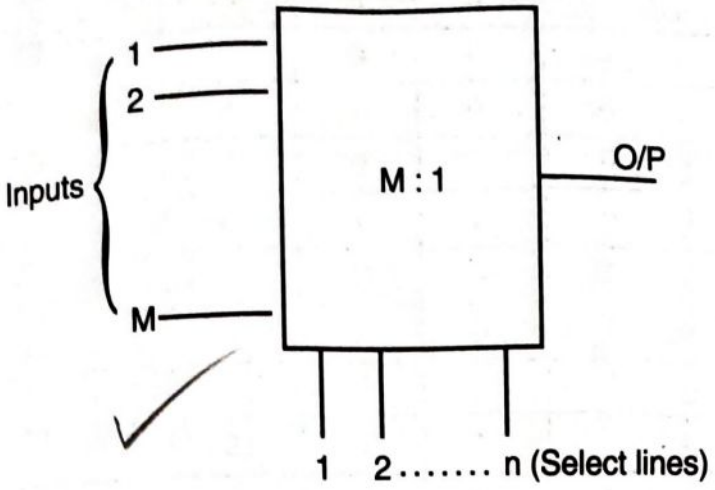

Fig. 3.4. A M : 1 Multiplexer where $M = 2^n$

A 4 to 1 line multiplexer is defined as the multiplexer consisting of four input channels and information of one of the channels can be selected and transmitted to output line according to the select inputs combinations. Selection of 4 inputs is possible with the help of two select lines. Input channels I_0, I_1, I_2 and I_3 are selected by the combinations of select inputs S_1 and S_0.

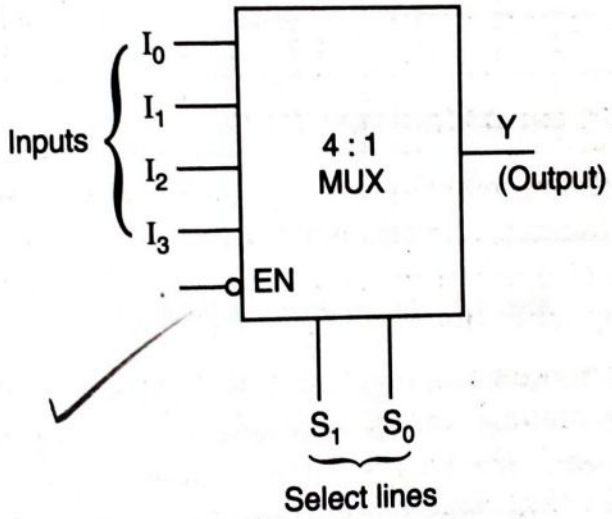

Fig. 3.5.

Therefore we can say that in multiplexer several inputs are combined and produces single output. In multiplexer it has the following parts :

(*i*) Input lines denoted by I_0, I_1, I_2, I_3 .....

(*ii*) Select lines denoted by S_1, S_0 .....

(*iii*) '*EN*' known as Enable or strobe pin is used for proper Control Operation of Multiplexer. It is "Active Low" Pin. It is also useful to expand two or more multiplexer ICs to a digital multiplexer with a larger number of inputs. In another way it used for cascading of multiplexers.

(*iv*) The output which is denoted by '*Y*'. The circuit diagram and truth table for 4 : 1 Multiplexer is given as :

Truth Table

Select Lines		Data Input	Output Y
S_0	S_1		
0	0	I_0	$I_0 \bar{S}_0 \bar{S}_1$
0	1	I_1	$I_1 \bar{S}_0 S_1$
1	0	I_2	$I_2 S_0 \bar{S}_1$
1	1	I_3	$I_3 S_0 S_1$

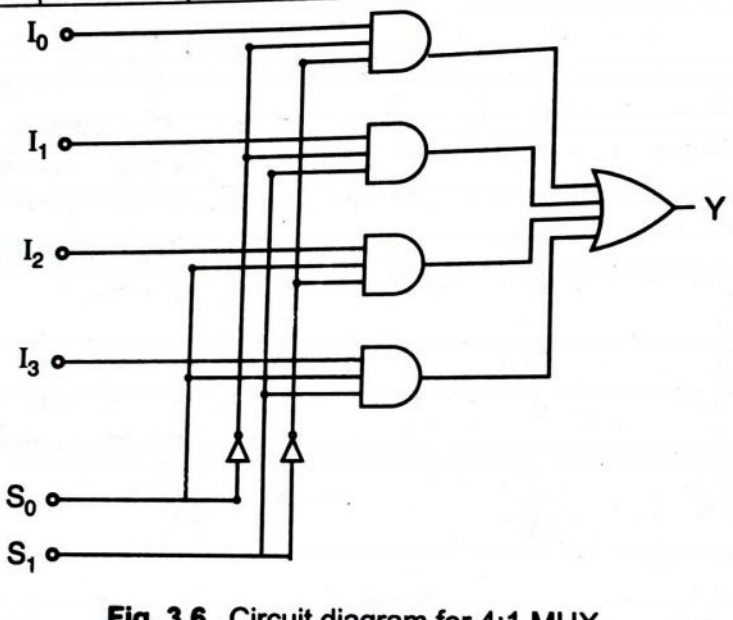

Fig. 3.6. Circuit diagram for 4:1 MUX.

Advantages of Multiplexer

A multiplexer is defined as a process which combines the small capacity or low speed sub-channels into a high capacity or high speed channel and transmits the combined channel over the communication link/transmission medium. On the receiving side, each of the sub-channels is separated from the high speed channel and sent to respective receiving devices operating at different speeds. This technique of combining low speed sub-channels into one high speed channel and transmitting over in communication link/transmission medium improves the link/medium utilization and also the throughput of the system. The use of multiplexer offers the following advantages :

(*i*) They are relatively cheap and reliable.

(*ii*) The response time of various hosts connected via the multiplexer is not affected.

(*iii*) The high volume of the data stream can be easily handled by them without any interferences from other hosts.

Multiplexer accept multiple low-speed data (voice, data, signal) and combine them into one high-speed channel for transmission onto a telephone line. On the receiving side, a de-multiplexer separates the incoming high-speed channel into appropriate low-speed individual signals. Thus, we are not only utilizing the high speed lines but also using the same link for multiple channels containing a variety of application (voice, data, video).

3.5 APPLICATIONS OF MULTIPLEXERS

Multiplexers find numerous and varied applications in digital system of all types. These applications include data selection, data routing, parallel-to-serial conversion, waveform generation and logic function generation.

3.6 CASCADING OF MULTIPLEXERS

Cascading means combining two or more multiplexer together in order to have large number of inputs *e.g.* combining of two 16 : 1 multiplexer together in order to perform 32 : 1 multiplexer operation.

In order to perform cascading, we have to increase the number of inputs which will depends on the selection lines *i.e.* ($M = 2^n$). So in order to provide additional select line, "Enable" or "Strobe" pin is used.

For example: ***A* 8 : 1 Multiplexer is realized by two 4 to 1 line multiplexer.**

Operation. In 4 : 1 Multiplexer two select lines are present *i.e.* S_1 S_0 but in cascading 8 : 1, three select lines are present. In order to provide extra select line, Enable is used. Enable pin is connected to upper multiplexer and acts as select line A and Enable Pin is connected to lower multiplexer with NOT gate and acts as select line $\bar{A}$.

Case I

When Enable is "active low", then the Multiplexer (upper) '1' will work and selects the inputs from I_0 to I_3 and at that time Multiplexer '2' will be deactivated because Enable is high at that time.

Case II

When Enable is "active high", then Multiplexer 2 will work and select inputs from I_4 to I_7 and Multiplexer 1 will be deactivated.

The Enable Pin *i.e.* A and $\bar{A}$ is acting as Most Significant bit (MSB) and its value either '1' or '0' decide the selection of Multiplexer '2' or multiplexer '1' as shown in figure 3.7.

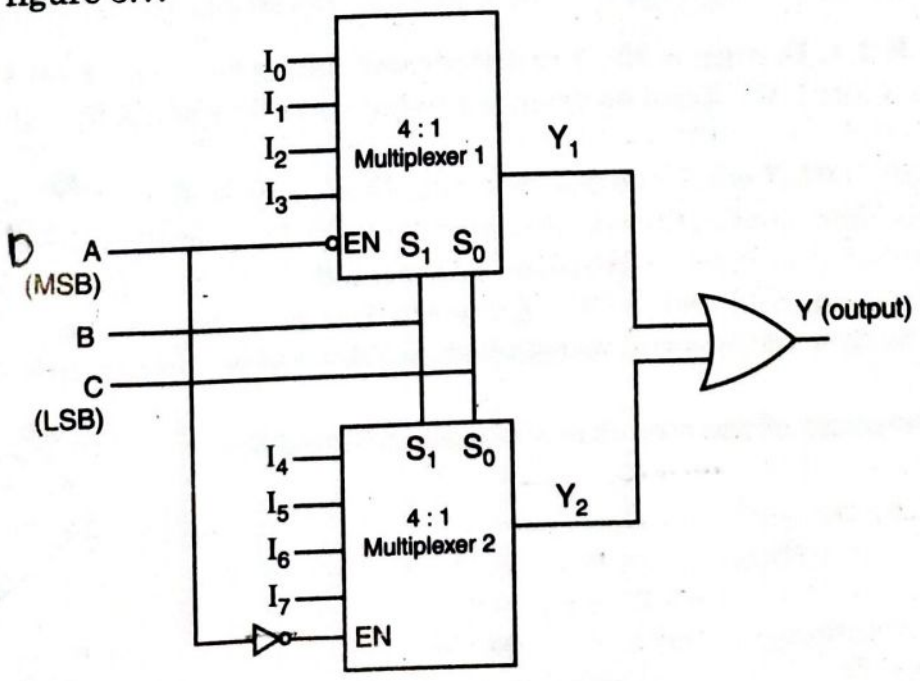

Fig. 3.7.

Similarly 16 to 1 multiplexer may be developed by two 8 to 1 multiplexer as shown in figure 3.8.

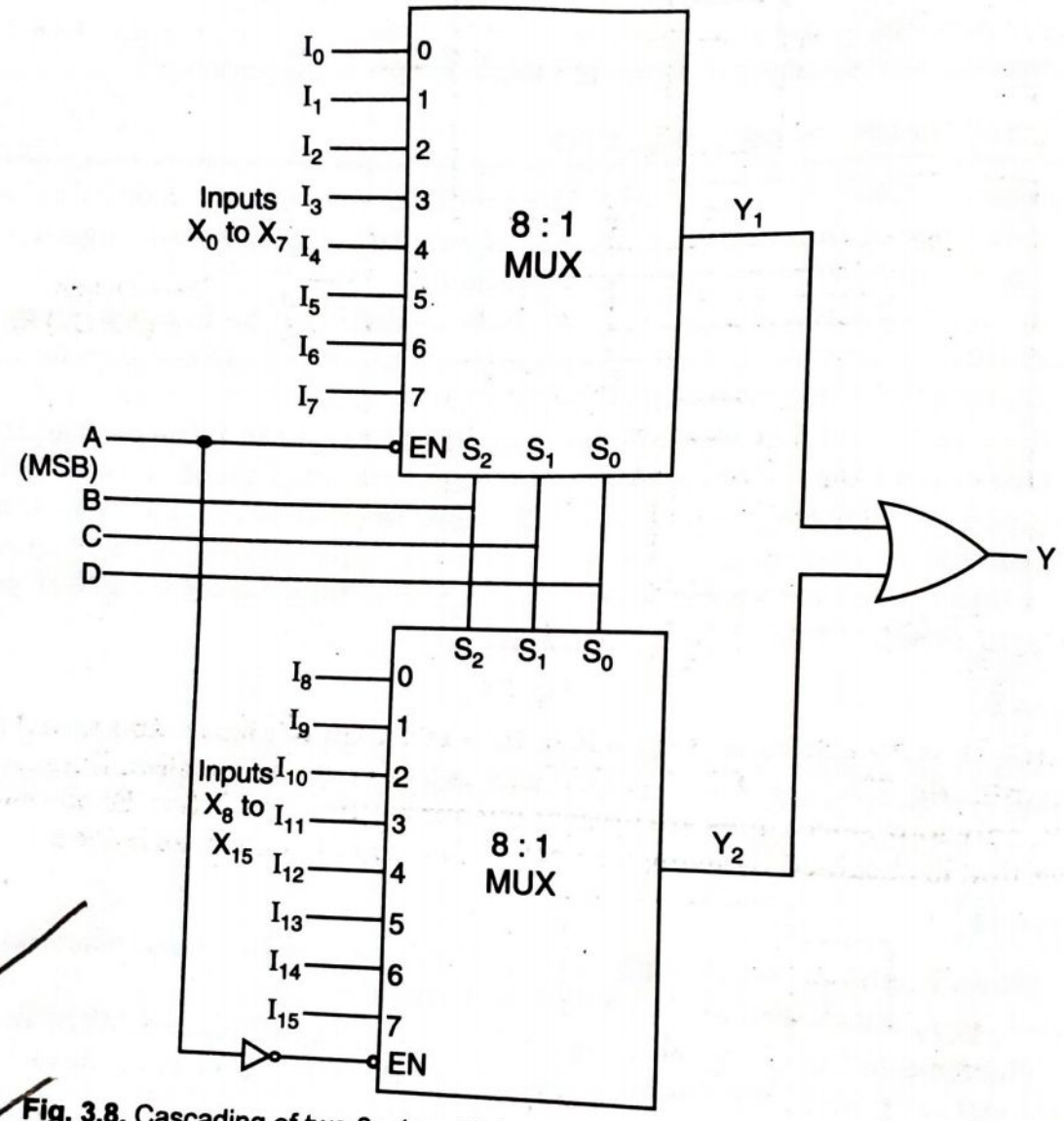

Fig. 3.8. Cascading of two 8 : 1 multiplexers to convert into 16 : 1 multiplexer

EXAMPLE 3.1. Design a 32 : 1 multiplexer using two 16 : 1 multiplexer.

Solution: A 32 : 1 MUX can be designed using two 16 : 1 MUX by following two ways :

(*i*) A 32 : 1 MUX will have five selection lines, say A, B, C, D, E, where A is MSB. If A is connected to the enable input of one of 16 : 1 multiplexers while the enable input of the other multiplexer is connected to $\bar{A}$, then for $A = 0$, the first multiplexer is enabled and for $A = 1$, the another multiplexer is enabled. Thus for $A = 0$, the first 16 lines will be selected and $A = 1$, the another 16 lines will be selected.

Designing of the circuit is given as in figure 3.9

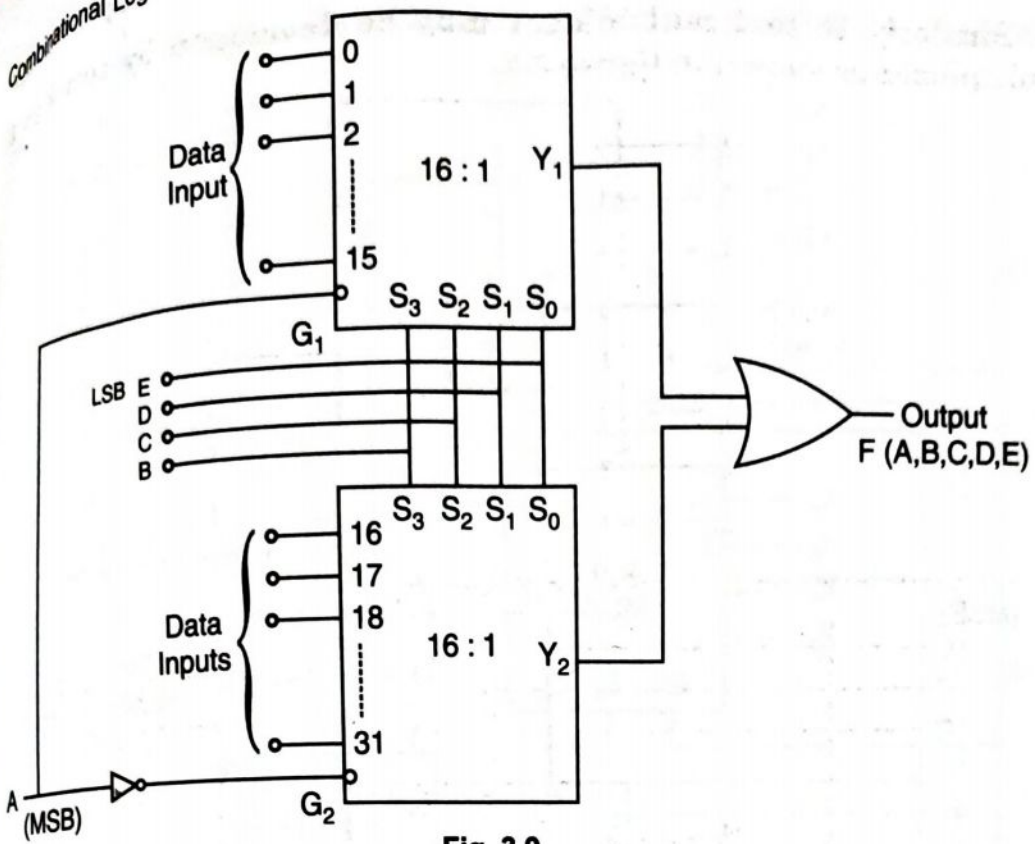

Fig. 3.9

***(ii)* Second method is by using two 16 : 1 multiplexer and one 2 : 1 Multiplexer.** The use of 2 : 1 multiplexer is to give one more select lines as in 16 : 1, only four select lines are present but our requirement is five. So one more select line is obtained by using 2 : 1 MUX. The circuit is given as follows :

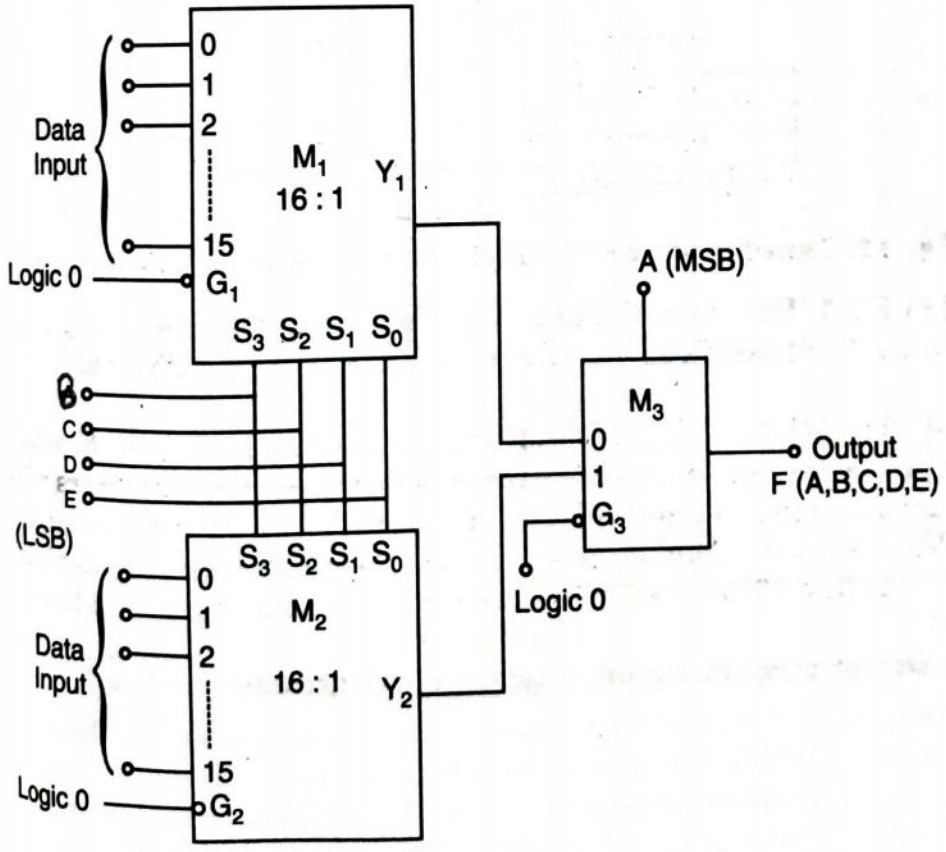

Fig. 3.10

EXAMPLE 3.2. Designing of 16 : 1 Multiplexer using 4 : 1 multiplexers.

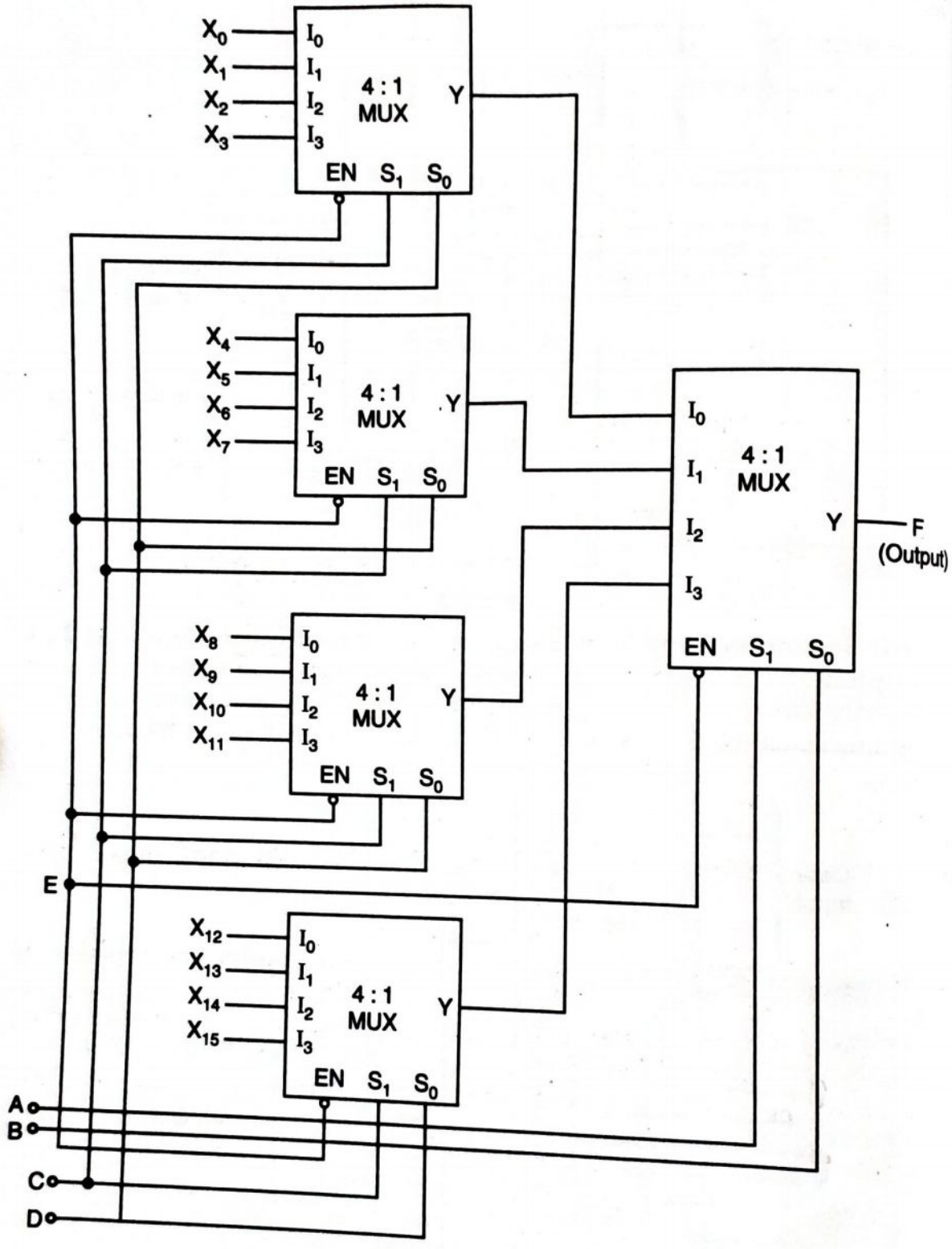

Fig. 3.11

EXAMPLE 3.3. Design a 40 : 1 multiplexer using 8 : 1 MUX.

Solution: In 40 : 1 multiplexer, there are 40 data input lines (I_0 through I_{39}), 6 select lines FEDCBA. The lower order three select bits C, B and A are used as S_2, S_1, S_0 select inputs respectively for 8 : 1 multiplexers M_1 through M_5. The higher order three select bits F, E and D are used as select inputs S_2, S_1 and S_0 for the multiplexer M_6 which selects output of one of the multiplexers M_1 through M_5.

For example if the select inputs are 010111, data input 7 of M_3 (I_{23}) will appear at the output Y.

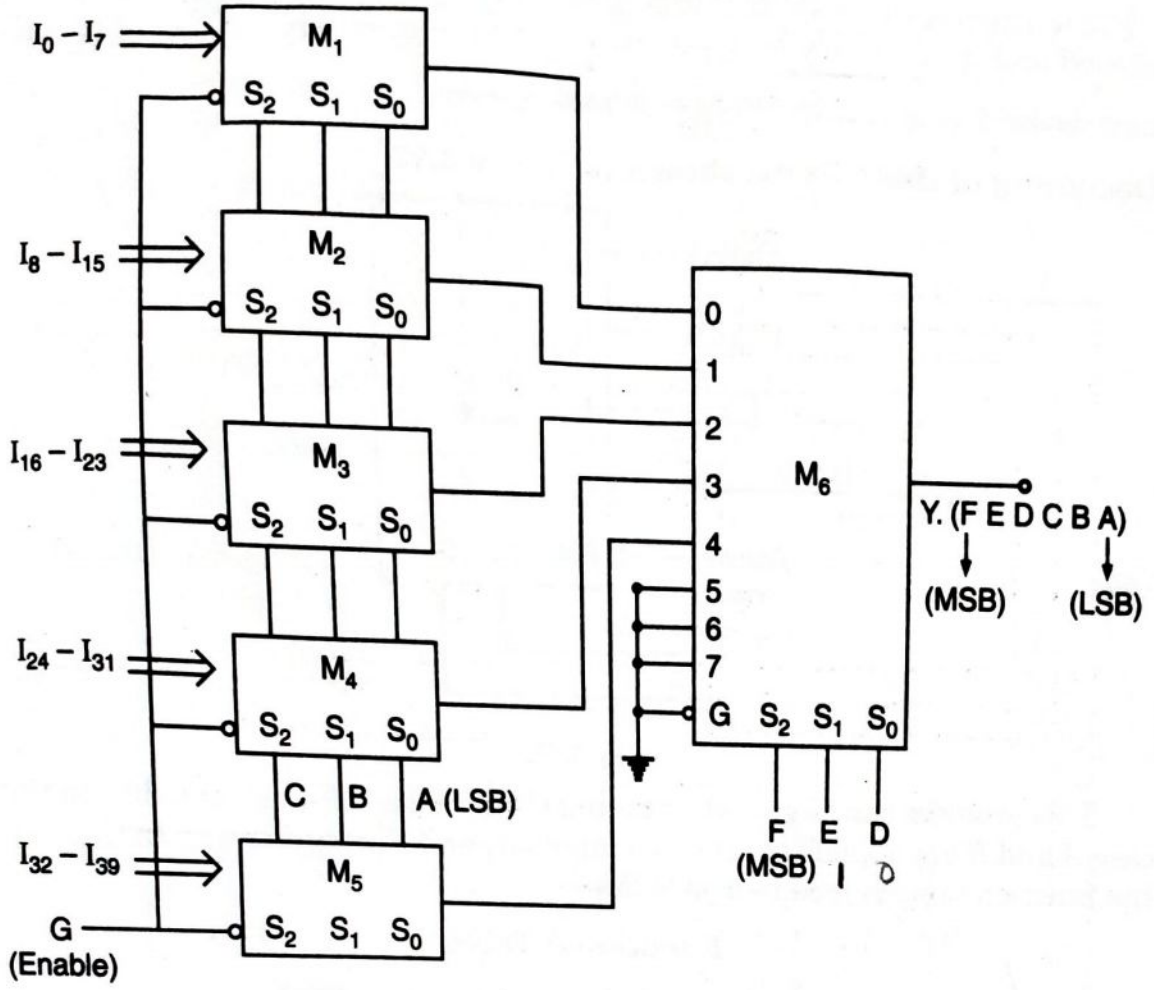

Fig. 3.12

EXAMPLE 3.4. Implement the following function using multiplexer, *i.e.*, 4 : 1 MUX.

$$F(A, B, C) = \Sigma (0, 2, 4, 7)$$

Solution: In order to implement this function using Multiplexer, the lower significant variables B and C are applied to S_1 and S_0 respectively. The inputs of multiplexer I_0 to I_3 are listed at the upper most row. $\bar{A}$ and its corresponding minterms 0 to 3 are placed at next row. Variable A and rest of the minterms 4 to 7 are placed next as shown below. Now circle the minterms 0, 2, 4 and 7 as these minterms produce logic 1 output.

Table

	I_0	I_1	I_2	I_3
$\bar{A}$	(0)	1	(2)	3
A	(4)	5	6	(7)
	1	0	$\bar{A}$	A

From this table, it can be seen that both the elements of the first column 0 and 4 are circled. Therefore '1' is placed at the bottom of that column. At the second column no elements are circled and so '0' is placed at the bottom of the

column. At third column only '2' is circled. Its corresponding variable is $\bar{A}$ and $\bar{A}$ is written at the bottom of this column. And finally, at fourth column '7' is circled and A is marked at the bottom of the column. Multiplexer inputs are now decided as $I_0 = 1$, $I_1 = 0$, $I_2 = \bar{A}$ and $I_3 = A$.

Designing of this Circuit shown in figure 3.13.

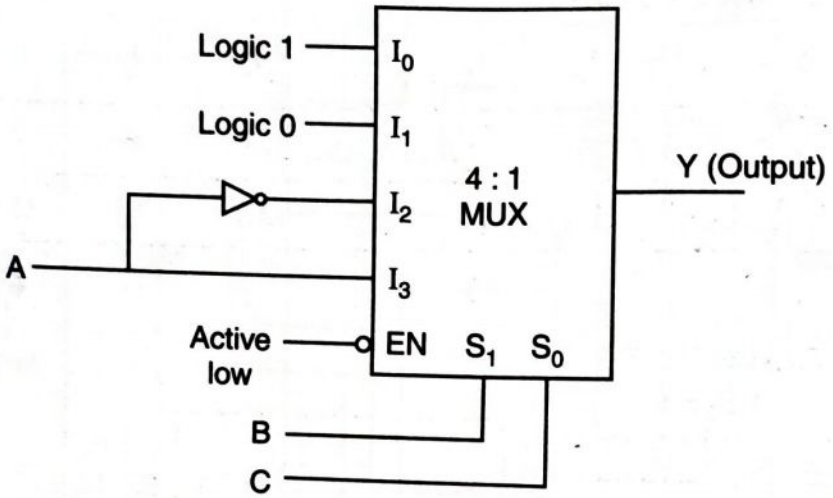

Fig. 3.13.

This problem can be solved by taking C bit instead of A variable/bit. In this case A and B are applied to selection inputs S_1 and S_0 respectively. In this case the function table is modified as follows :

Functional Table

	I_0	I_1	I_2	I_3
$\bar{C}$	(0)	(2)	(4)	6
C	1	3	5	(7)
	$\bar{C}$	$\bar{C}$	$\bar{C}$	C

Designing shown in figure 3.14.

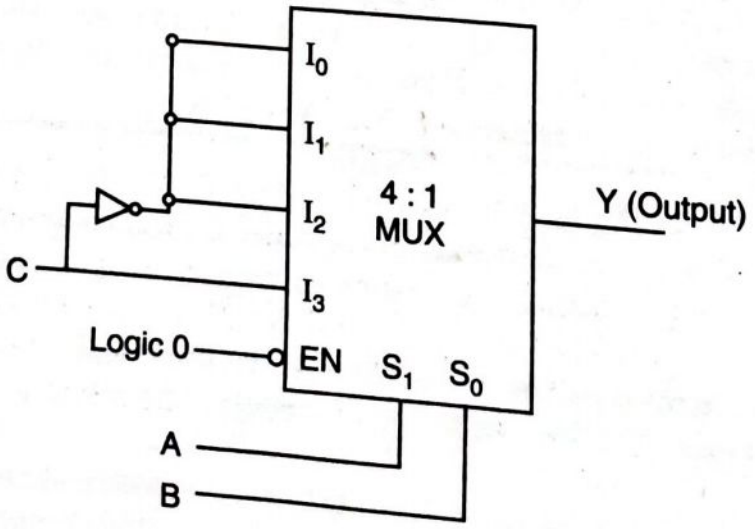

Fig. 3.14.

EXAMPLE 3.5. Implement the following function with a multiplexer (8 : 1).

$$F(A, B, C, D) = \Sigma m\ (0, 1, 3, 4, 8, 9, 15)$$

Solution: The given function contains four variables but in case of 8 : 1 Multiplexer three select lines are present. The implementation table and circuit diagram is given as follows :

(a) Implementation Table

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\bar{A}$	(0)	(1)	2	(3)	(4)	5	6	7
A	(8)	(9)	10	11	12	13	14	(15)
	1	1	0	$\bar{A}$	$\bar{A}$	0	0	A

(b) Circuit Diagram

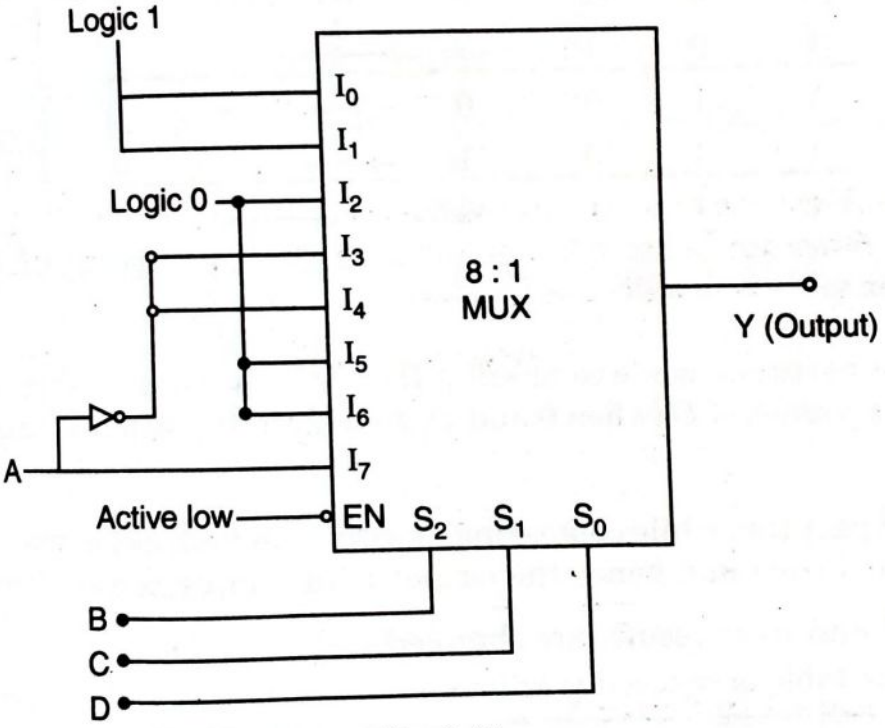

Fig. 3.15.

SOLVED EXAMPLES

EXAMPLE 3.6. Realize this truth table using 8 : 1 Multiplexer.

Solution: To realize a four variable truth table or logic expression using an 8 : 1 multiplexer the truth table is partitioned as shown by dotted lines. In this A, B, C inputs are connected to S_3, S_2 and S_1. Now we compare input D and output Y for each group of two rows. There are four possible values of Y and there are 0, 1, D and $\bar{D}$. The designing of circuit is as follows:

Inputs				*Output*	
A	*B*	*C*	*D*	*Y*	
0	0	0	0	0	0
0	0	0	1	0	
0	0	1	0	1	$\bar{D}$
0	0	1	1	0	
0	1	0	0	1	$\bar{D}$
0	1	0	1	0	
0	1	1	0	1	1
0	1	1	1	1	
1	0	0	0	0	*D*
1	0	0	1	1	
1	0	1	0	1	1
1	0	1	1	1	
1	1	0	0	1	$\bar{D}$
1	1	0	1	0	
1	1	1	0	0	*D*
1	1	1	1	1	

Case I

In first partition, while comparing D with Y, the output (Y) is zero in both condition *i.e.* values of D (when 0 and 1), therefore the output is taken as zero.

Case II

In 2nd partition while comparing D with Y, in both cases the output is in complemented from and hence the output is in complemented form and the output is $\bar{D}$ and other results are obtained.

Now the table is reduced is follows :

A	*B*	*C*	*Y*
0	0	0	0
0	0	1	$\bar{D}$
0	1	0	$\bar{D}$
0	1	1	1
1	0	0	*D*
1	0	1	1
1	1	0	$\bar{D}$
1	1	1	*D*

The designing of this truth table is as given below :

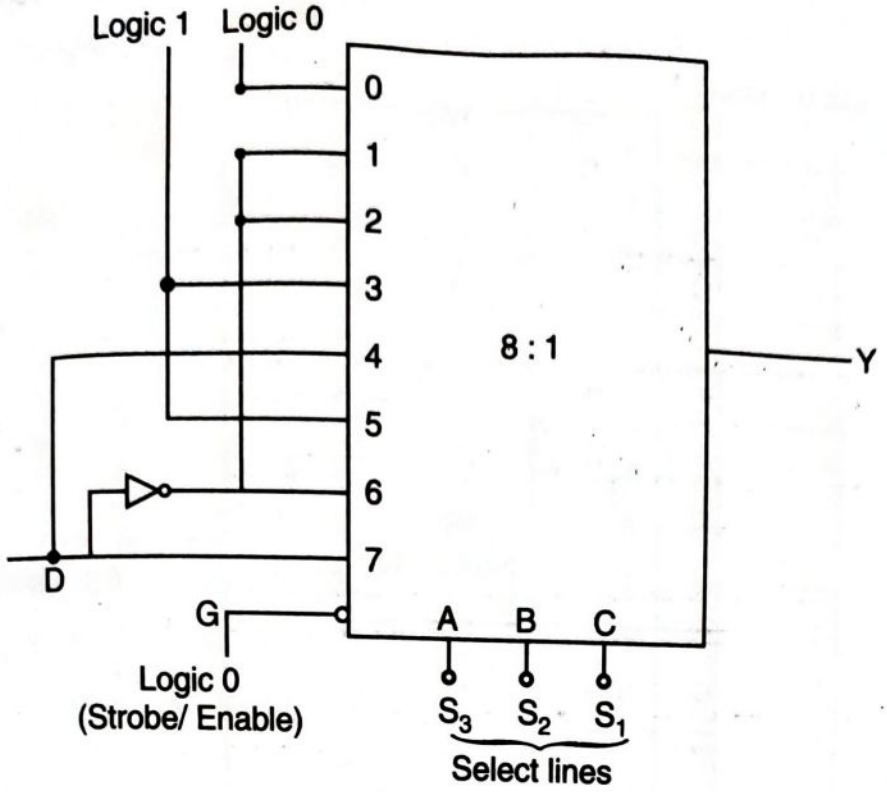

Fig. 3.16.

EXAMPLE 3.7. Realize the logic function as in table using a 16 : 1 multiplexer.

Table

Inputs				*Output*
A	*B*	*C*	*D*	*Y*
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Solution: The data output is 1, when the input 2, 4, 6, 7, 9, 10, 11, 12, 15. Therefore, the data input lines corresponding to these input are connected to logic 1 and data input lines 0, 1, 3, 5, 8, 13 and 14 are connected to logic 0. The signals corresponding to input variables A, B, C and D are applied at the selection lines S_3, S_2, S_1 and S_0 respectively and enable input (G) is connected to logic 0.

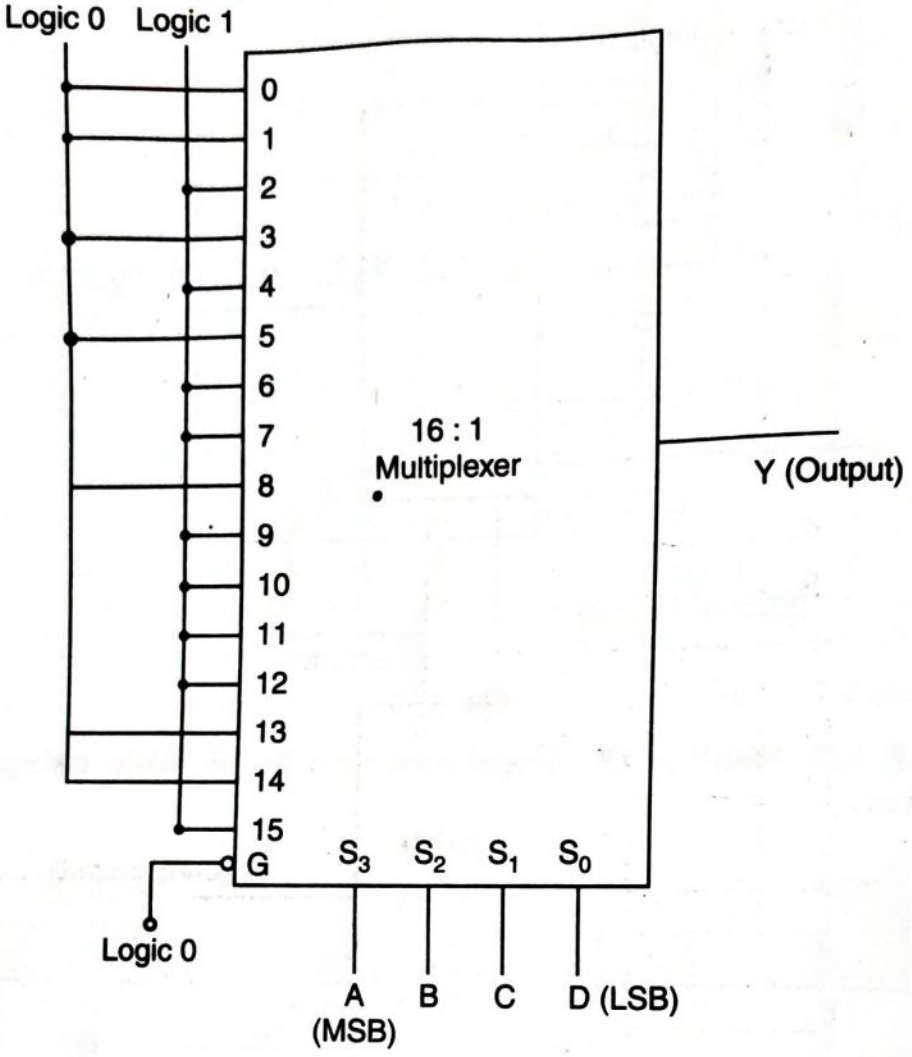

Fig. 3.17.

EXAMPLE 3.8. Design the circuit of full-adder using 8 : 1 multiplexer.

Solution: The truth-table for full-adder is given as follows :

Inputs			*Outputs*	
A_n	B_n	C_{n-1}	S_n	C_n
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

Here S_n and C_n are sum and carry output.

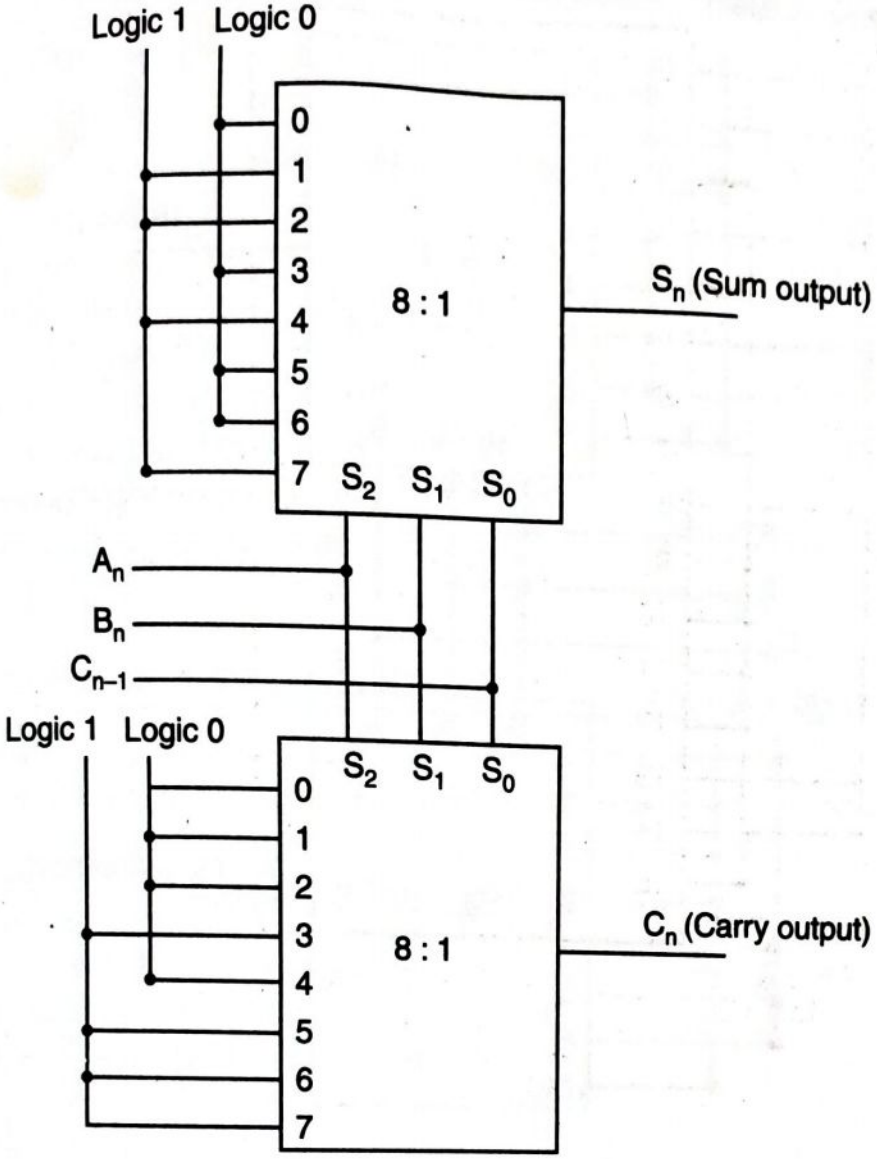

Fig. 3.18.

EXAMPLE 3.9. Design the circuit of full subtractor using 8 : 1 multiplexer.

Solution: In full subtractor, the three inputs are present and two outputs *i.e.* Borrow and difference. In order to design its circuit by multiplexer, we first make the truth table of full subtractor.

A_n	B_n	C_{n-1}	*Borrow* (B_0)	*Difference* (D_0)
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

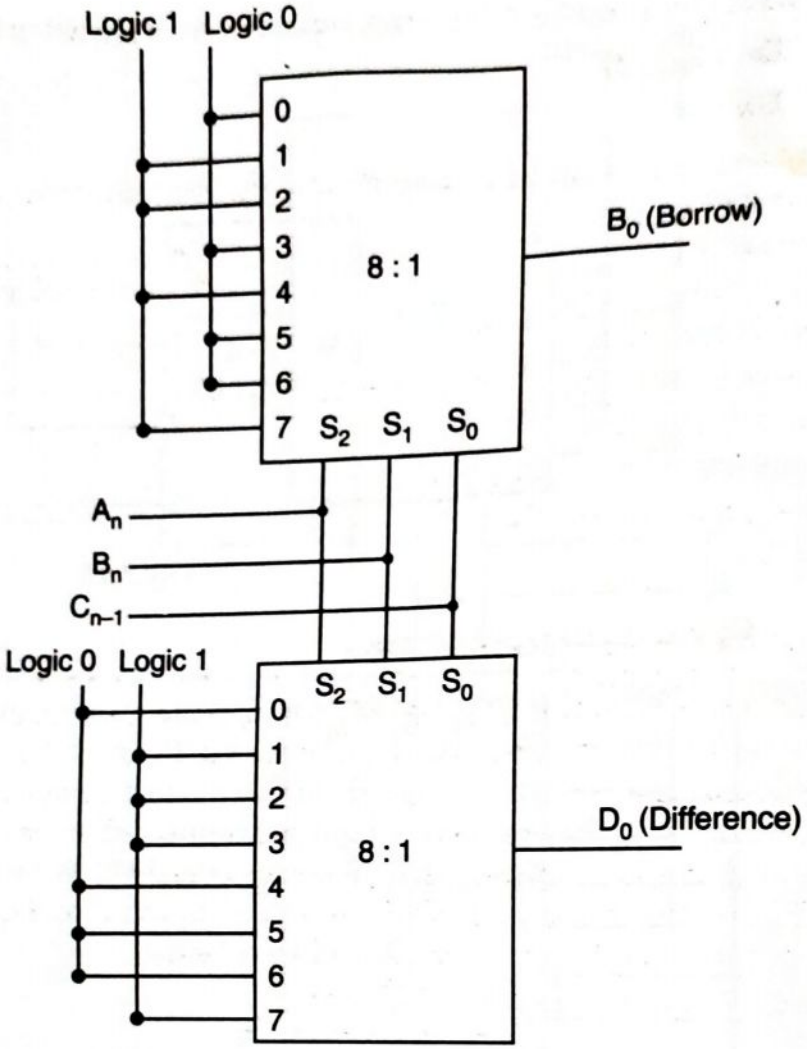

Fig. 3.19. Circuits diagram

3.7 DEMULTIPLEXER/DATA DISTRIBUTORS

The demultiplexer is a combinational circuit which means "one to many". It states that it is having one input line and many outputs are present. A demultiplexer is the logic circuit which takes information through a single input line and transmits the same information over one of the possible 2^m output lines. The selection of specific output line is controlled by the bit combinations of selection lines.

The select input code determines to which output the data input will be transmitted.

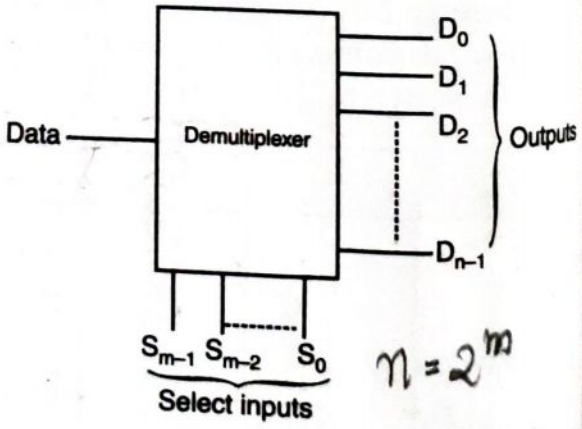

Fig. 3.20.

3.8 BLOCK REPRESENTATION OF DEMULTIPLEXER

The number of output lines is n and the number of select lines is m, where $n = 2^m$ as shown in figure 3.20. The output of most of these devices are active-low, also there is an active-low enable/data input terminal available.

EXAMPLE 3.10. Design the following outputs by demultiplexer

$F_1 = \Sigma m\ (1, 2, 3)$

$F_2 = \Sigma m\ (0)$

Solution: In the given output function F_1 and F_2, the maximum minterm is 3 and it can be realize by using 1 : 4 demultiplexer. The selection lines are given to A and B variables. The minterms are passed to 'NAND' gate because the output of demultiplexer is low. The designing is as follows in fig. 3.21.

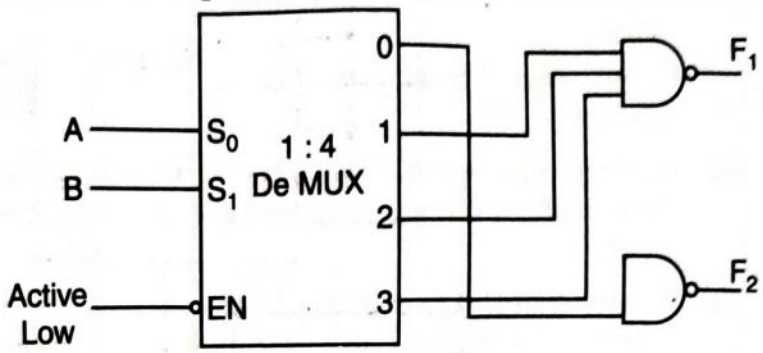

Fig. 3.21.

3.9 CASCADING OF DEMULTIPLEXERS

Like multiplexers cascading (joining) of two or more demultiplexers is also possible in order to attain large number of outputs. Figure 3.22 demonstrates how a 1 to 8 demultiplexer can be formed with two 1 to 4 demultiplexers. Here highest significant bit A of the selection inputs is connected to the enable inputs, one directly and one is complemented. When A is logic '0' ; one of the output lines D_0 to D_3 will be selected according to select inputs B and C, and for A is logic '1' one of the output lines D_4 to D_7 will be selected.

The circuit diagram is given as follows :

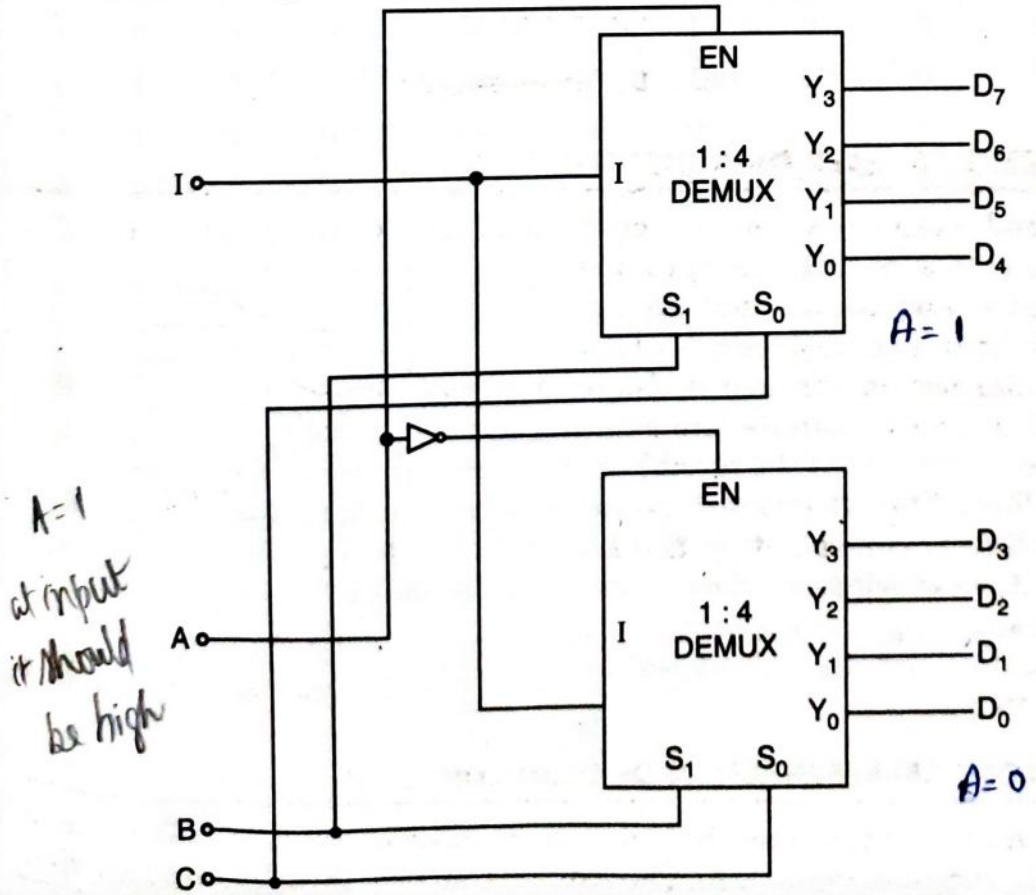

Fig. 3.22. Cascading of two 1 : 4 Demultiplexer to Convert into 1 : 8 Demultiplexer.

Comparison between Multiplexer and Demultiplexer

Multiplexer	*Demultiplexer*
(*i*) It means "Many into One" *i.e.* it has multiple input and single output.	(*i*) It stands for "One into Many" *i.e.* it has one input and many outputs.
(*ii*) It is also known as "Data Selector".	(*ii*) It is known as "Data Distributor".
(*iii*) Realization using multiplexer does not require additional gates.	(*iii*) Realization using demultiplexer requires additional gates *i.e.* NAND gates.
(*iv*) It is used in Data selection, Data, routing, parallel to serial conversion.	(*iv*) It is used in serial to parallel conversion, binary to decimal conversion.
(*v*) The *example* are 4 : 1, 8 : 1, 16 : 1 etc.	(*v*) The example are 1 : 4, 1 : 8, 1 : 16, etc.

EXAMPLE 3.11. Design a Gray-to-BCD Code converter using one 1 : 16 demultiplexer and NAND gates.

Solution: The truth table for Gray-to-BCD code converter is given as follows:

Gray Code					*BCD Code*			
G_3	G_2	G_1	G_0		D	C	B	A
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	1	2	0	0	1	0
0	0	1	0	3	0	0	1	1
0	1	1	0	4	0	1	0	0
0	1	1	1	5	0	1	0	1
0	1	0	1	6	0	1	1	0
0	1	0	0	7	0	1	1	1
1	1	0	0	8	1	0	0	0
1	1	0	1	9	1	0	0	1

In order to design this we need to use outputs A, B, C, D which is obtained by passing each output '1' to bubbled OR gate which is similar to NAND gate *e.g.* for A the outputs are '1' corresponding to decimal 1, 2, 7, 4, 13.

The designing is as follows :

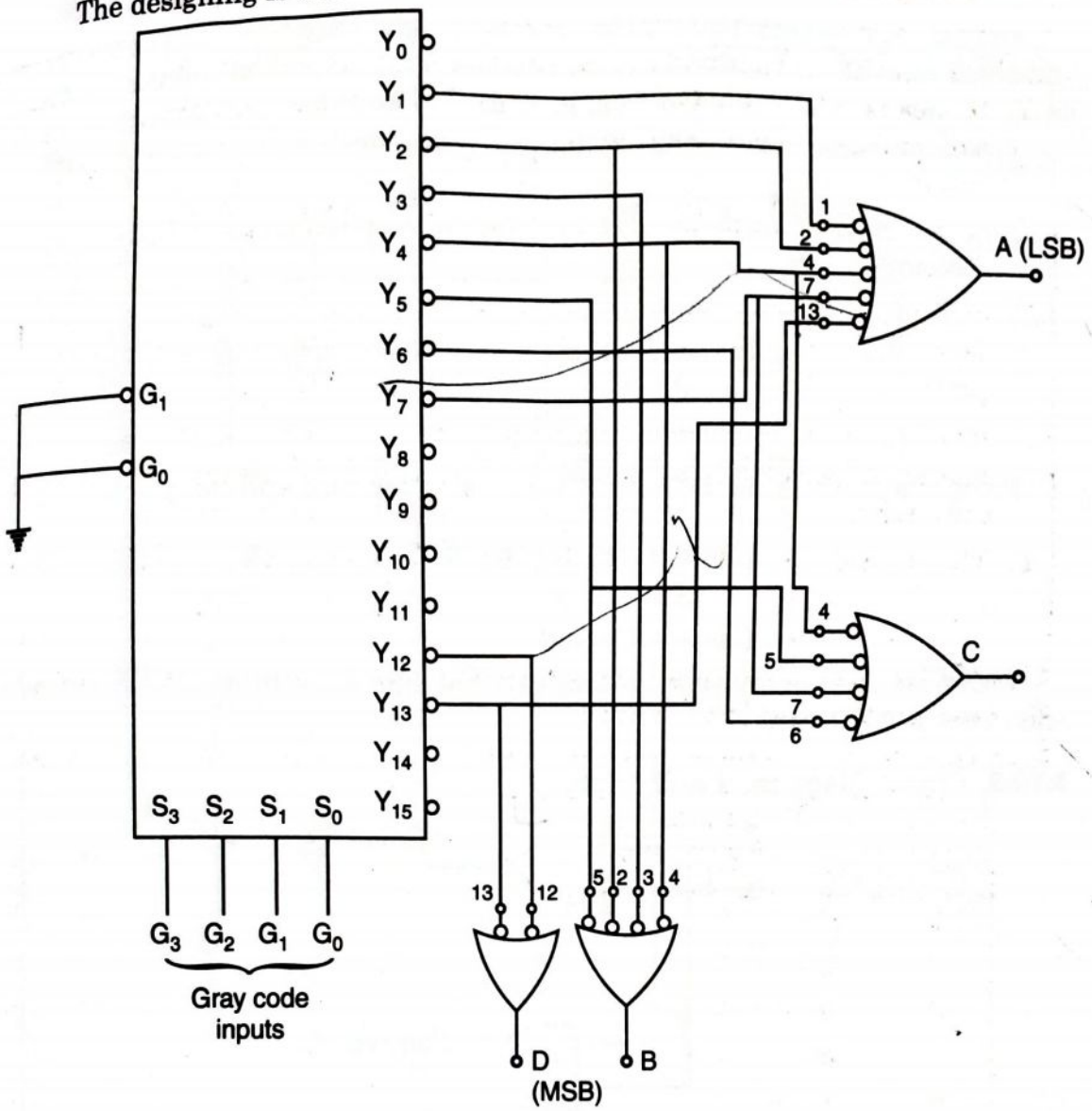

Fig. 3.23.

3.10 ADDERS

Adders are the combinational circuit that perform addition of bits. Addition of two binary digits is most basic arithmetic operation. Table format the simple addition consists of four possible operations, which are

$$0 + 0 = 0,$$
$$0 + 1 = 1,$$
$$1 + 0 = 1$$

and $$1 + 1 = 10.$$

The first three operations produce a sum of one digit and last operation produces sum consists of two digits. The higher significant bit of this result is called carry. Adders is classified into two types :

(*i*) Half Adder

(*ii*) Full Adder

3.10.1. Half Adder

A combinational circuit that performs the addition of two bits is known as Half Adder.