Multiplexers and Demultiplexers

In this lesson, you will learn about:

1. Multiplexers
2. Combinational circuit implementation with multiplexers
3. Demultiplexers
4. Some examples

Multiplexer

A Multiplexer (see Figure 1) is a combinational circuit that selects one of the 2^n input signals (D_0, D_1, D_2, ……, D_{2^n-1}) to be passed to the single output line Y.

Q. How to select the input line (out of the possible 2^n input signals) to be passed to the output line?

A. Selection of the particular input to be passed to the output is controlled by a set of n input signals called "*Select Inputs*" (S_0, S_1, S_2, ……., S_{n-1}).

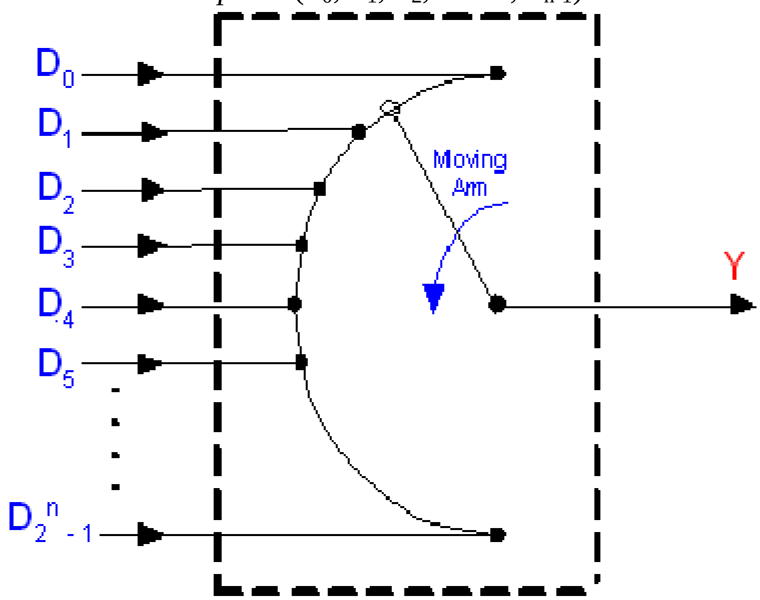


Figure 1: Multiplexer

Example 1: 2x1 Mux

A 2x1 Mux has 2 input lines (D_0 & D_1) , one select input (S), and one output line (Y). (see Figure 2)

IF S=0, then $Y= D_0$
Else (S=1) $Y= D_1$

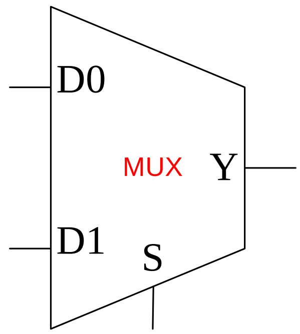


Figure 2: A 2 X 1 Multiplexer

Thus, the output signal Y can be expressed as:

$$Y = \overline{S}\,D_0 + S\,D_1$$

Example 2: 4x1 Mux

A 4x1 Mux has 4 input lines (D_0, D_1, D_2, D_3), two select inputs (S_0 & S_1), and one output line Y. (see Figure 3)

IF S_1S_0=00, **then** Y= D_0
IF S_1S_0=01, **then** Y= D_1
IF S_1S_0=10, **then** Y= D_2
IF S_1S_0=11, **then** Y= D_3

Thus, the output signal Y can be expressed as:

$$Y = \underbrace{\overline{S}_1\overline{S}_0}_{\text{minterm m0}} D_0 + \underbrace{\overline{S}_1 S_0}_{\text{minterm m1}} D_1 + \underbrace{S_1\overline{S}_0}_{\text{minterm m2}} D_2 + \underbrace{S_1 S_0}_{\text{minterm m3}} D_3$$

Obviously, the input selected to be passed to the output depends on the minterm expressions of the select inputs.

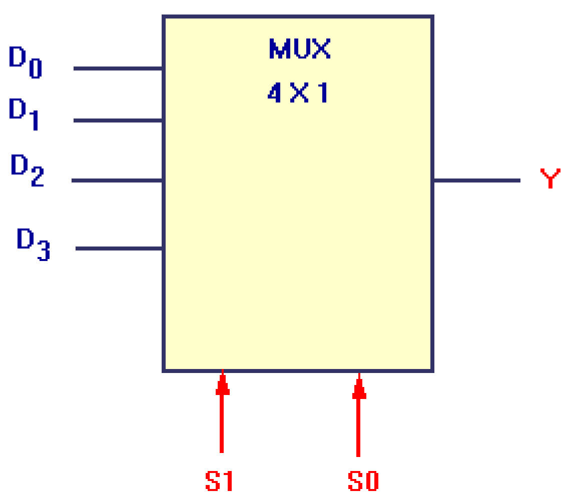


Figure 3: A 4 X 1 Multiplexer

In General,

For MUXes with n select inputs, the output Y is given by

$$\mathbf{Y = m_0D_0 + m_1D_1 + m_2D_2 + \ldots. + m_{2^n-1}D_{2^n-1}}$$

Where m_i = $i^{\underline{th}}$ *minterm* of the *Select Inputs*

Thus

$$Y = \sum_{i=0}^{2^n-1} m_i \, D_i$$

Example 3: Quad 2X1 Mux

Given two 4-bit numbers A and B, design a multiplexer that selects one of these 2 numbers based on some select signal S. Obviously, the output (Y) is a 4-bit number.

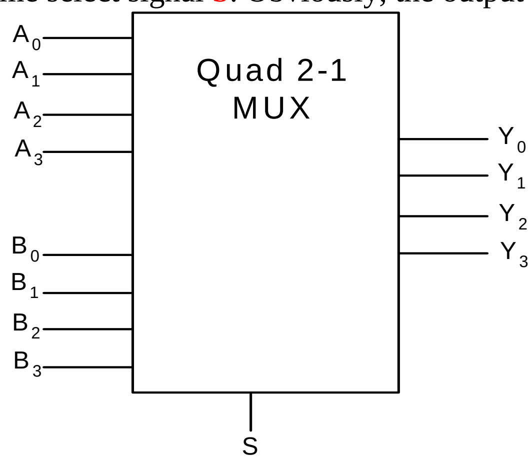


Figure 4: Quad 2 X 1 Multiplexer

The 4-bit output number Y is defined as follows:

Y = A **IF** S=0, **otherwise** Y = B

The circuit is implemented using four 2x1 Muxes, where the output of each of the Muxes gives one of the outputs (Y_i).

Combinational Circuit Implementation using Muxes

Problem Statement:

Given a function of n-variables, show how to use a MUX to implement this function. This can be accomplished in one of 2 ways:

- Using a Mux with n-select inputs
- Using a Mux with n-1 select inputs

Method 1: Using a Mux with n-select inputs

n variables need to be connected to *n* select inputs. For a MUX with *n* select inputs, the output Y is given by:

$Y = m_0 D_0 + m_1 D_1 + m_2 D_2 + \ldots + m_{2^n-1} D_{2^n-1}$
Alternatively,

$$Y = \sum_{i=0}^{2^n-1} m_i D_i$$

Where m_i = $i^{\underline{th}}$ *minterm* of the *Select Inputs*

The MUX output expression is a *SUM of minterms* expression for all minterms (m_i) which have their corresponding inputs (D_i) equal to 1.

Thus, it is possible to implement any function of *n-variables* using a MUX with *n*-select inputs by proper assignment of the input values ($D_i \in \{0, 1\}$).
$Y(S_{n-1} \ldots S_1 S_0) = \sum(minterms)$

Example 4: Implement the function F (A, B, C) $= \sum(1, 3, 5, 6)$ (see Figure 5)
Since number of variables n = 3, this requires a Mux with 3 select inputs, i.e. an 8x1 Mux

The most significant variable A is connected to the most significant select input S_2 while the least significant variable C is connected to the least significant select input S_0 , thus:

$$S_2 = A,\ S_1 = B, \text{ and } S_0 = C$$

For the MUX output expression (sum of minterms) to include minterm 1 we assign $D_1 = 1$

Likewise, to include minterms 3, 5, and 6 in the sum of minterms expression while excluding minterms 0, 2, 4, and 7, the following input (D_i) assignments are made

$$D_1 = D_3 = D_5 = D_6 = 1$$
$$D_0 = D_2 = D_4 = D_7 = 0$$

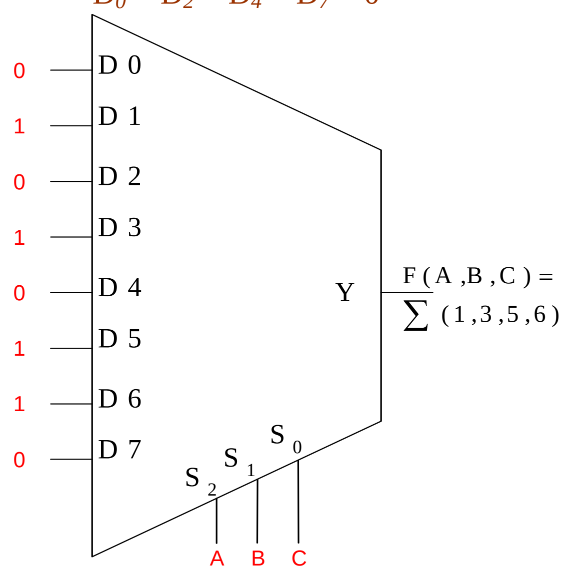


Figure 5: Implementing function with Mux with n select inputs

Method 2: Using a Mux with (n-1) select inputs

Any n-variable logic function can be implemented using a Mux with only (n-1) select inputs (e.g 4-to-1 mux to implement any 3 variable function)

This can be accomplished as follows:

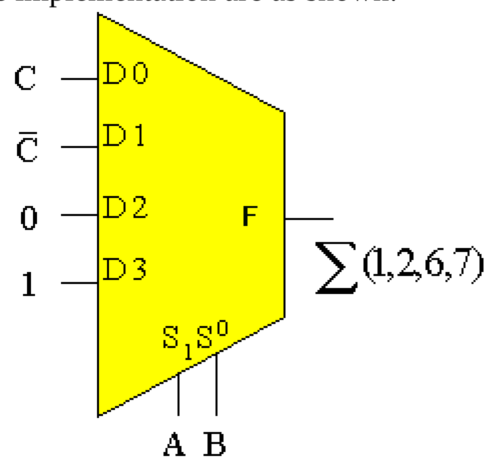
- Express function in canonical sum-of-minterms form.
- Choose n-1 variables to be connected to the mux select lines.
- Construct the truth table of the function, but grouping the n-1 select input variables together (e.g. by making the n-1 select variables as most significant inputs).

The values of D_i (mux input line) will be 0, or 1, or n^{th} variable or complement of n^{th} variable of value of function F, as will be clarified by the following example.

Example 5: Implement the function $F(A, B, C) = \sum(1, 2, 6, 7)$ (see figure 6)
This function can be implemented with a 4-to-1 line MUX.
A and B are applied to the select line, that is

$$A \Rightarrow S_1,\ B \Rightarrow S_0$$

The truth table of the function and the implementation are as shown:



	A	B	C	F	
1	0	0	0	0	F = C
	0	0	1	1	
2	0	1	0	1	F = C′
	0	1	1	0	
3	1	0	0	0	F = 0
	0	1	1	0	
4	1	1	0	1	F = 1
	1	1	1	1	

Figure 6: Implementing function with Mux with n-1 select inputs

Example 6: Consider the function F(A,B,C,D)=∑(1,3,4,11,12,13,14,15)

This function can be implemented with an 8-to-1 line MUX (see Figure 7)
A, B, and C are applied to the select inputs as follows:

$$A \Rightarrow S_2,\ B \Rightarrow S_1,\ C \Rightarrow S_0$$

The truth table and implementation are shown.

A	B	C	D	F	
0	0	0	0	0	F = D
0	0	0	1	1	
0	0	1	0	0	F = D
0	0	1	1	1	
0	1	0	0	1	F = $\overline{D}$
0	1	0	1	0	
0	1	1	0	0	F = 0
0	1	1	1	0	
1	0	0	0	0	F = 0
1	0	0	1	0	
1	0	1	0	0	F = D
1	0	1	1	1	
1	1	0	0	1	F = 1
1	1	0	1	1	
1	1	1	0	1	F = 1
1	1	1	1	1	

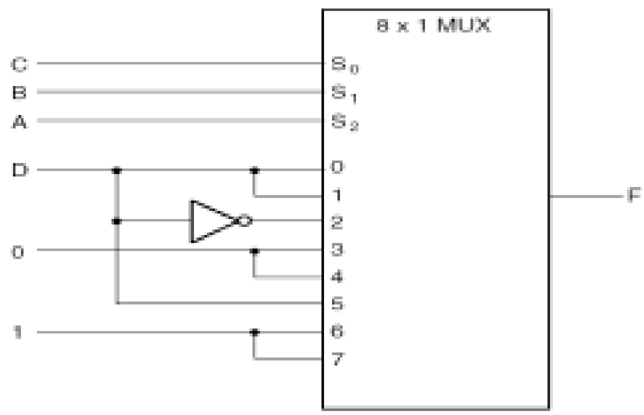


Figure 7: Implementing function of Example 6

Demultiplexer

It is a digital function that performs inverse of the multiplexing operation.
It has one input line (E) and transmits it to one of 2^n possible output lines (D_0, D_1, D_2, …, D_{2^n-1}). The selection of the specific output is controlled by the bit combination of *n* select inputs.

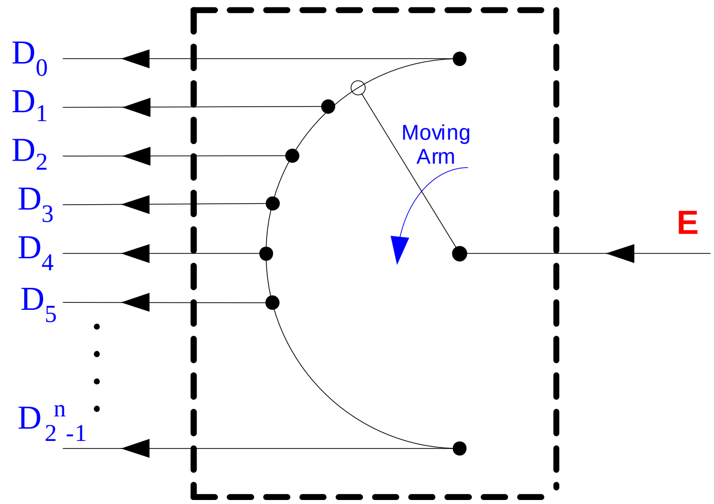


Figure 8: A demultiplexer

Example 7: A 1-to-4 line Demux

The input E is directed to one of the outputs, as specified by the two select lines S_1 and S_0.

D_0 = E if $S_1S_0 = 00 \Rightarrow D_0 = S_1' S_0' E$

D_1 = E if $S_1S_0 = 01 \Rightarrow D_1 = S_1' S_0 E$

D_2 = E if $S_1S_0 = 10 \Rightarrow D_2 = S_1 S_0' E$

D_3 = E if $S_1S_0 = 11 \Rightarrow D_3 = S_1 S_0 E$

A careful inspection of the Demux circuit shows that it is identical to a 2 to 4 decoder with enable input.

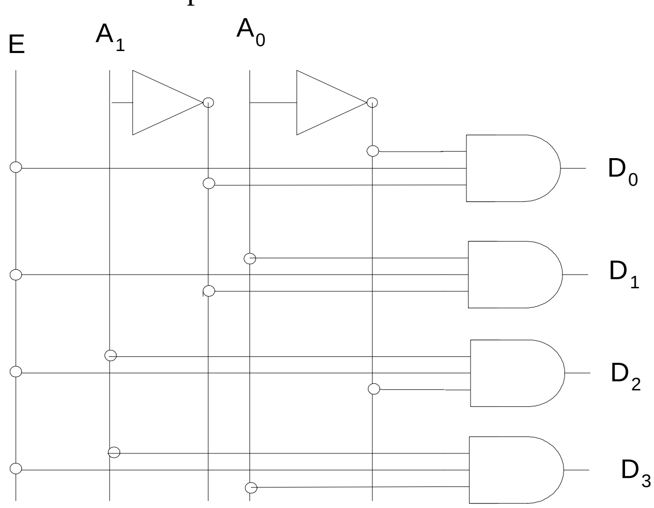


Figure 8: A 1-to-4 line demultiplexer

- For the decoder, the inputs are A_1 and A_0, and the enable is input E. (see figure 9)
- For demux, input E provides the data, while other inputs accept the selection variables.
- Although the two circuits have different applications, their logic diagrams are exactly the same.

Decimal value	Enable	Inputs		Outputs			
	E	A_1	A_0	D_0	D_1	D_2	D_3
	0	X	X	0	0	0	0
0	1	0	0	1	0	0	0
1	1	0	1	0	1	0	0
2	1	1	0	0	0	1	0
3	1	1	1	0	0	0	1

Figure 9: Table for 1-to-4 line demultiplexer