Digital Circuit
&
Logic Design
(CSDC-0101)

Decoders

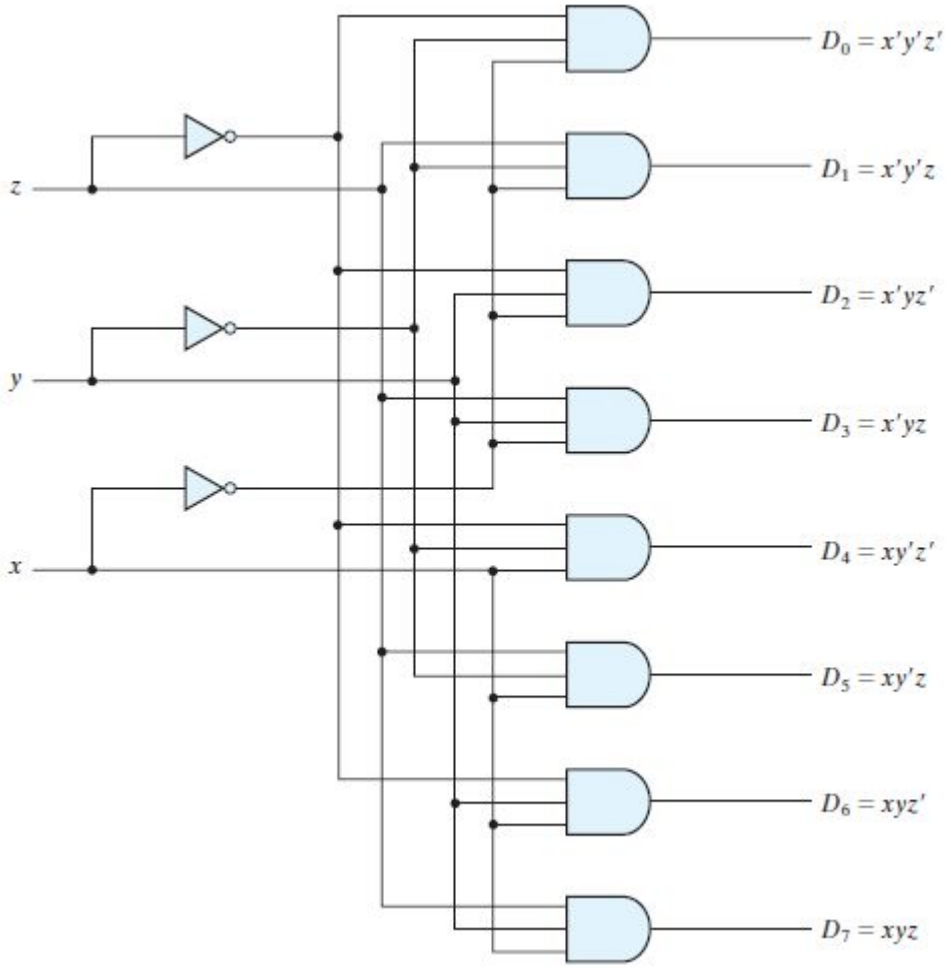
A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines.

Each combination of inputs in a decoder will assert a unique output, each representing one of the minterms of the number of input variables.

The name decoder is also used in conjunction with other code converters, such as a BCD-to-seven-segment decoder.

Example of a decoder is three-to-eight-line decoder where three inputs are decoded into eight outputs, each representing one of the minterms of the three input variables. The three inverters provide the complement of the inputs, and each one of the eight *AND gates* generates one of the minterms. A particular application of this decoder is binary-to-octal conversion.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1



Truth Table of a Three-to-Eight-Line Decoder

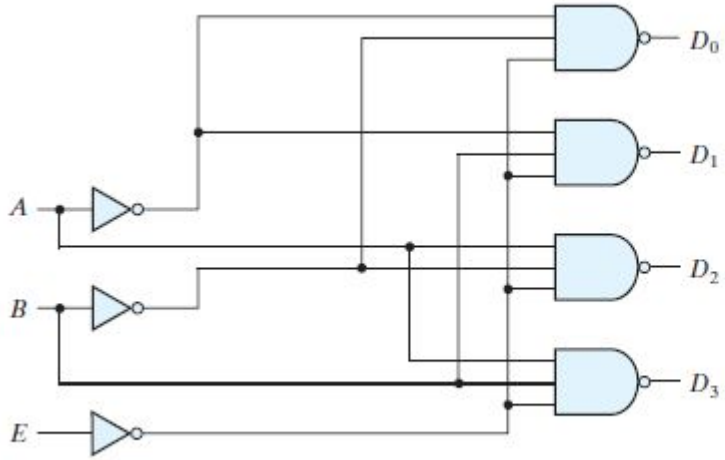
Inputs			Outputs							
x	**y**	**z**	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6**	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Three-to-eight-line decoder

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Some decoders are constructed with *NAND gates*. Since a NAND gate produces the AND operation with an inverted output, it becomes more economical to generate the decoder minterms in their complemented form. Furthermore, decoders include one or more enable inputs to control the circuit operation.

The decoder is enabled when E is equal to 0 (i.e., active-low enable).



(a) Logic diagram

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

(b) Truth table

Two-to-four-line decoder with enable input

A decoder with enable input can function as a **demultiplexer**— a circuit that receives information from a single line and directs it to one of 2^n possible output lines.

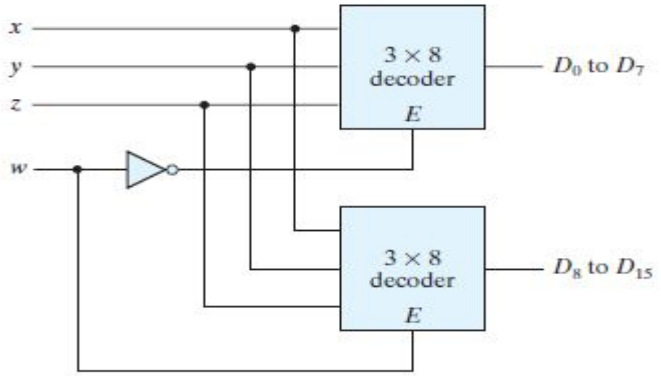
The selection of a specific output is controlled by the bit combination of n selection lines.

The two-to-four-line decoder can function as a one-to-four-line demultiplexer when E is taken as a data input line and A and B are taken as the selection inputs. The single input variable E has a path to all four outputs, but the input information is directed to only one of the output lines, as specified by the binary combination of the two selection lines A and B. This feature can be verified from the truth table of the circuit. For example, if the selection lines AB = 10, output D_2 will be the same as the input value E, while all other outputs are maintained at 1. Because decoder and demultiplexer operations are obtained from the same circuit, a decoder with an enable input is referred to as a *decoder – demultiplexer*.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Decoders with enable inputs can be connected together to form a larger decoder circuit.

Two 3-to-8-line decoders with enable inputs connected to form a 4-to-16-line decoder. When w = 0, the top decoder is enabled and the other is disabled. The bottom decoder outputs are all 0's, and the top eight outputs generate minterms 0000 to 0111. When w = 1, the enable conditions are reversed: The bottom decoder outputs generate minterms 1000 to 1111, while the outputs of the top decoder are all 0's.



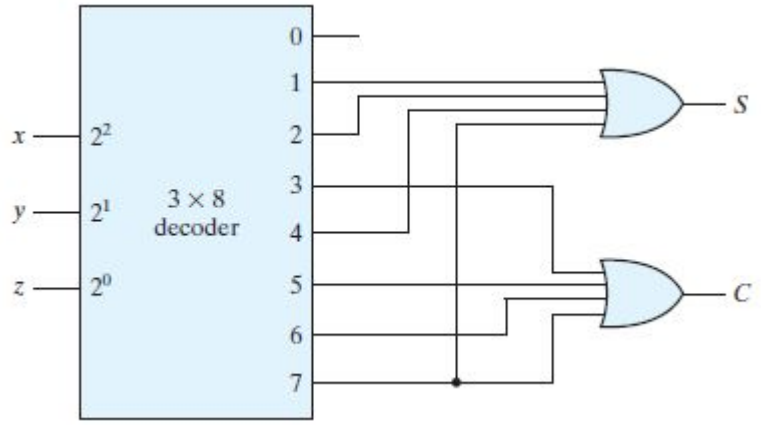
In general, enable inputs are a convenient feature for interconnecting two or more standard components for the purpose of combining them into a similar function with more inputs and outputs.

Combinational Logic Implementation

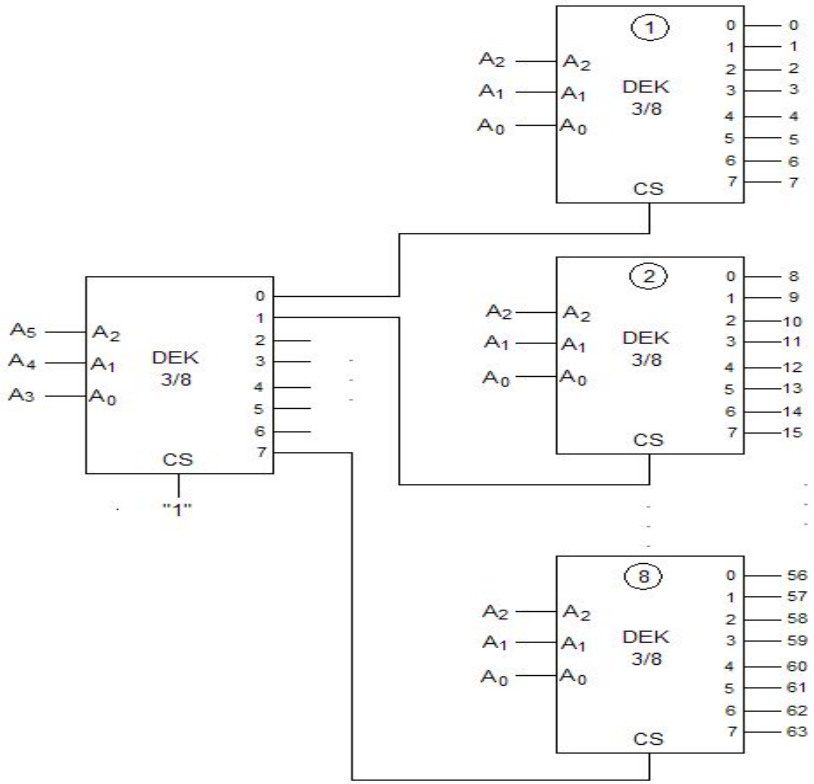
A decoder provides the 2^n minterms of n input variables. Each asserted output of the decoder is associated with a unique pattern of input bits. Since any Boolean function can be expressed in sum-of-minterms form, a decoder that generates the minterms of the function, together with an external OR gate that forms their logical sum, provides a hardware implementation of the function. In this way, any combinational circuit with n inputs and m outputs can be implemented with an n -to-2^n-line decoder and m OR gates.

$S(x, y, z) = \sum(1, 2, 4, 7)$
$C(x, y, z) = \sum(3, 5, 6, 7)$

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



How many 3-to- 8 line decoders with an enable input are needed to construct a 6-to-64 line decoder.



Number of M bit MUX/DeMUX/Decoder/Encoder to construct N bit MUX/DeMUX/Decoder/Encoder is:

ceil (N-1)/(M-1)

Encoders

An encoder is a digital circuit that performs the inverse operation of a decoder. An encoder has 2^n (or fewer) input lines and n output lines. The output lines, as an aggregate, generate the binary code corresponding to the input value.

The encoder can be implemented with OR gates whose inputs are determined directly from the truth table.

Truth Table of an Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

$$z = D_1 + D_3 + D_5 + D_7$$
$$y = D_2 + D_3 + D_6 + D_7$$
$$x = D_4 + D_5 + D_6 + D_7$$

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Encoders

The encoder defined here has the limitation that only one input can be active at any given time. If two inputs are active simultaneously, the output produces an undefined combination.

To resolve this ambiguity, encoder circuits must establish an input priority to ensure that only one input is encoded.

If D_3 and D_6 are 1 simultaneously, the output of the encoder will be 111 because all three outputs are equal to 1 which does not represent either binary 3 or binary 6.

If both D_3 and D_6 are 1 at the same time, the output will be 110 because D_6 has higher priority than D3.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

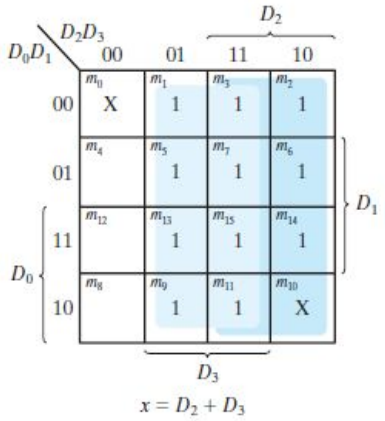
Priority Encoder

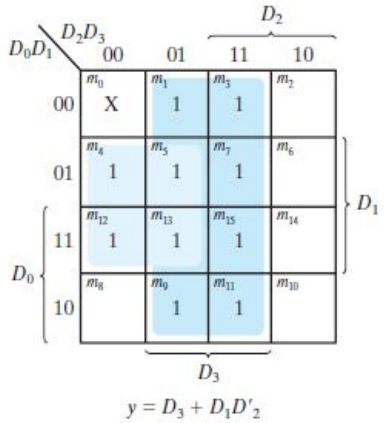
A priority encoder is an encoder circuit that includes the priority function.

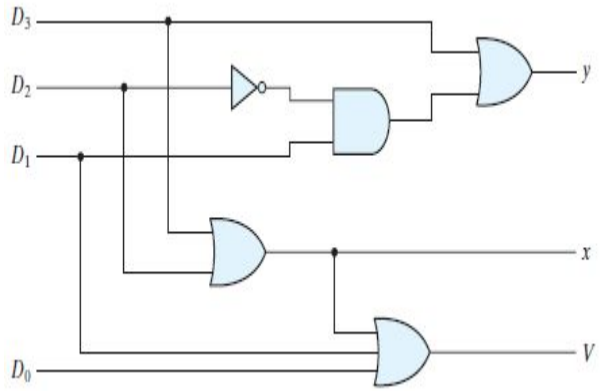
The operation of the priority encoder is such that if two or more inputs are equal to 1 at the same time, the input having the highest priority will take precedence.

Truth Table of a Priority Encoder

Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1







Priority Encoder

V is a valid bit indicator that is set to 1 when one or more inputs are equal to 1. If all inputs are 0, there is no valid input and V is equal to 0.
Input D_3 has the highest priority, so, regardless of the values of the other inputs, when this input is 1, the output for xy is 11 (binary 3). D_2 has the next priority level. The output is 10 if $D_2 = 1$, provided that $D_3 = 0$, regardless of the values of the other two lower priority inputs. The output for D_1 is generated only if higher priority inputs are 0, and so on down the priority levels.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Multiplexers

A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. A multiplexer is also called a *data selector*, since it selects one of many inputs and steers the binary information to the output line.

The selection of a particular input line is controlled by a set of selection lines.

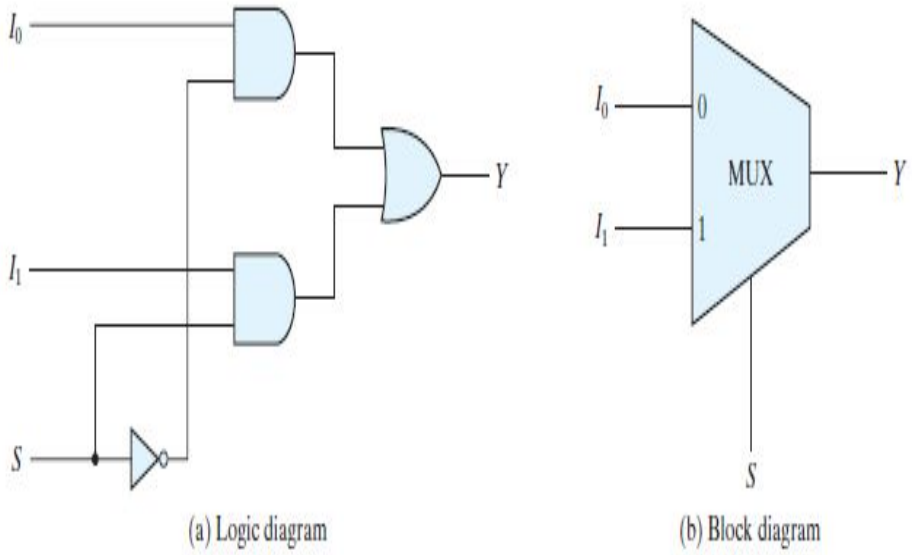
Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.

The multiplexer acts like an electronic switch that selects one of two sources.

The AND gates and inverters in the multiplexer resemble a decoder circuit, and indeed, they decode the selection input lines.

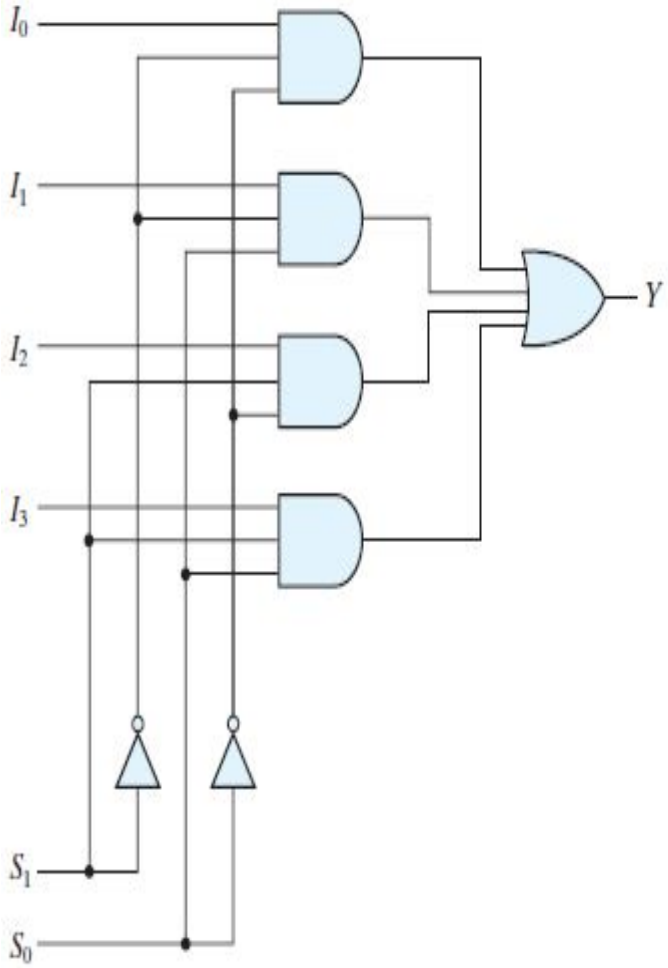
In general, a 2^n-to-1-line multiplexer is constructed from an n-to-2^n decoder by adding 2^n input lines to it, one to each AND gate. The outputs of the AND gates are applied to a single OR gate.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1



(a) Logic diagram (b) Block diagram

Two-to-one-line multiplexer

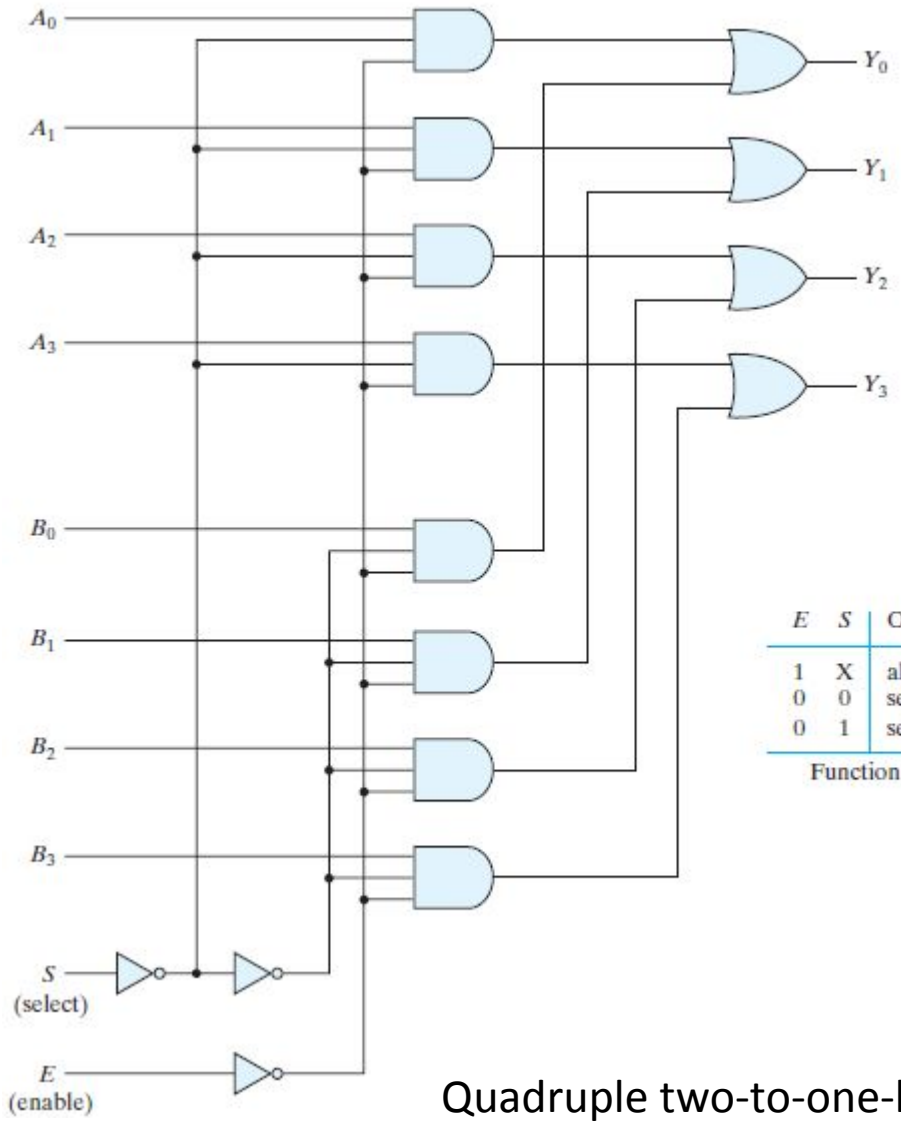


(a) Logic diagram

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(b) Function table

Four-to-one-line multiplexer



E	S	Output Y
1	X	all 0's
0	0	select A
0	1	select B

Function table

Quadruple two-to-one-line multiplexer

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Boolean Function Implementation

A decoder can be used to implement Boolean functions by employing external OR gates.

An examination of the logic diagram of a multiplexer reveals that it is essentially a decoder that includes the OR gate within the unit.

The minterms of a function are generated in a multiplexer by the circuit associated with the selection inputs. The individual minterms can be selected by the data inputs, thereby providing a method of implementing a Boolean function of n variables with a multiplexer that has n selection inputs and 2^n data inputs, one for each minterm.

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

The two variables x and y are applied to the selection lines in that order; x is connected to the S1 input and y to the S0 input.

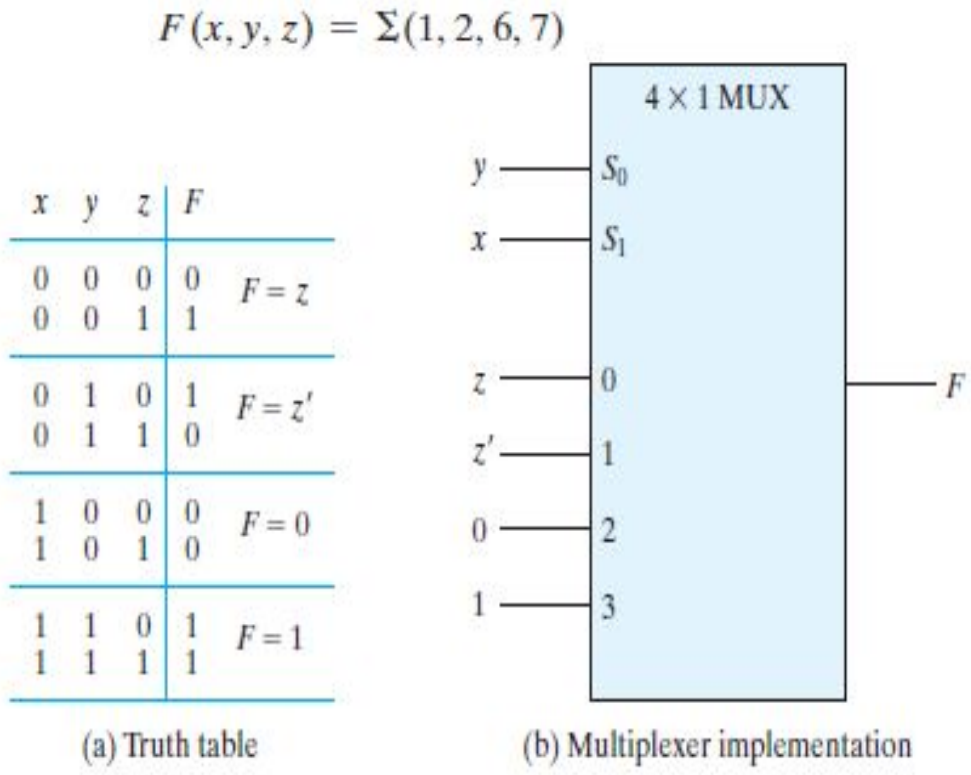
The values for the data input lines are determined from the truth table of the function.

When xy = 00, output F is equal to z because F = 0 when z = 0 and F = 1 when z = 1. This requires that variable z be applied to data input 0.

The operation of the multiplexer is such that when xy = 00, data input 0 has a path to the output, and that makes F equal to z .

In a similar fashion, we can determine the required input to data lines 1, 2, and 3 from the value of F when xy = 01, 10, and 11, respectively.

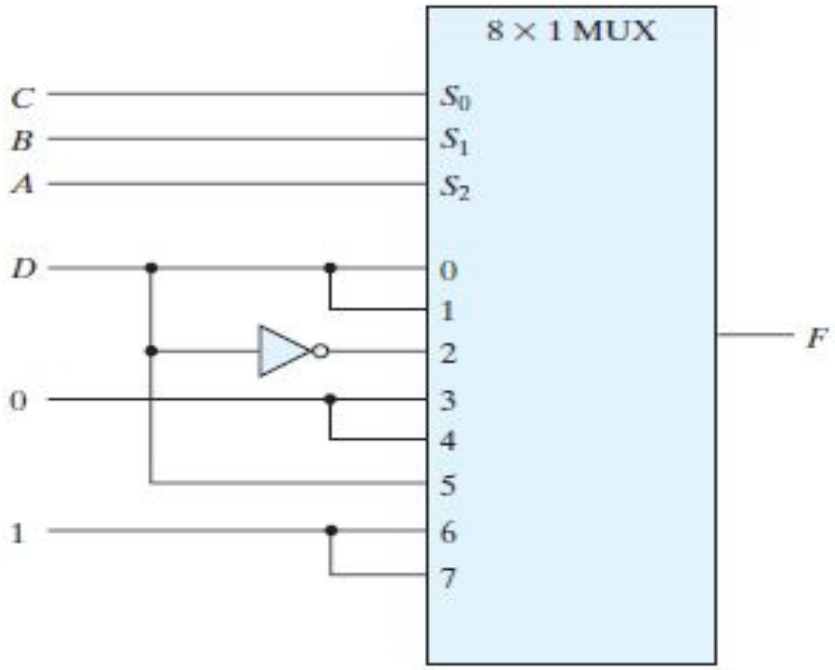
This particular example shows all four possibilities that can be obtained for the data inputs.



Implementing a Boolean function with a multiplexer

$$F(A, B, C, D) = \Sigma(1, 3, 4, 11, 12, 13, 14, 15)$$

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	$F = D'$
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	



SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1

Three-State Gates

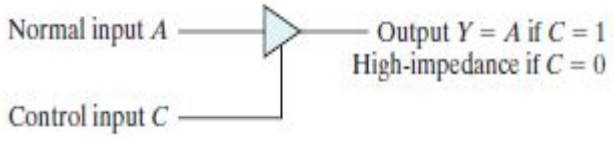
A multiplexer can be constructed with three-state gates—digital circuits that exhibit three states.

Two of the states are signals equivalent to logic 1 and logic 0 as in a conventional gate.

The third state is a high-impedance state in which

- (1) the logic behaves like an open circuit, which means that the output appears to be disconnected,
- (2) the circuit has no logic significance, and
- (3) the circuit connected to the output of the three-state gate is not affected by the inputs to the gate.

Three-state gates may perform any conventional logic, such as AND or NAND. However, the one most commonly used is the buffer gate.



Graphic symbol for a three-state buffer

SOURCE: HTTPS://DOI.ORG/10.1007/978-3-319-66775-1_1