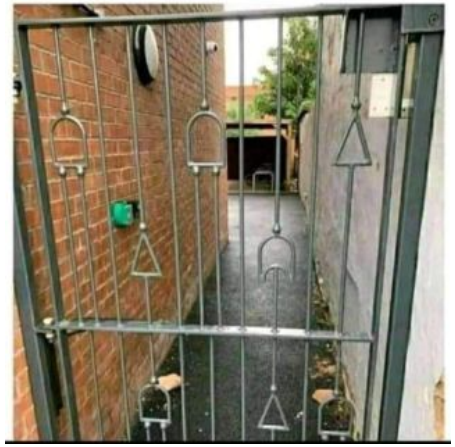
Logic Gate:



Functions of Combinational Logic

Chapter 6

Sumaiyah Zahid

Half Adder

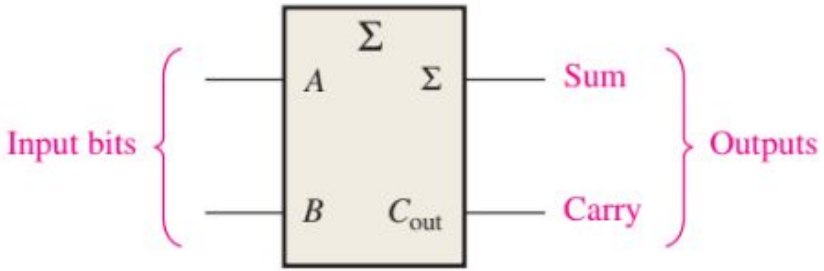


TABLE 6–1

Half-adder truth table.

A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Σ = sum
C_{out} = output carry
A and B = input variables (operands)

Half Adder

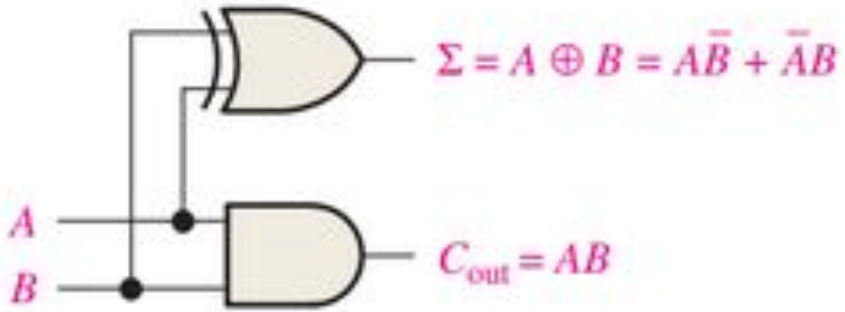


FIGURE 6–2 Half-adder logic diagram.

Full Adder

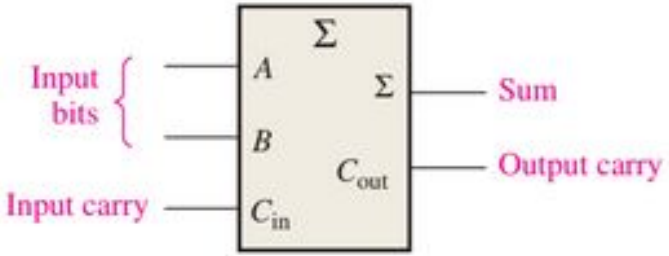


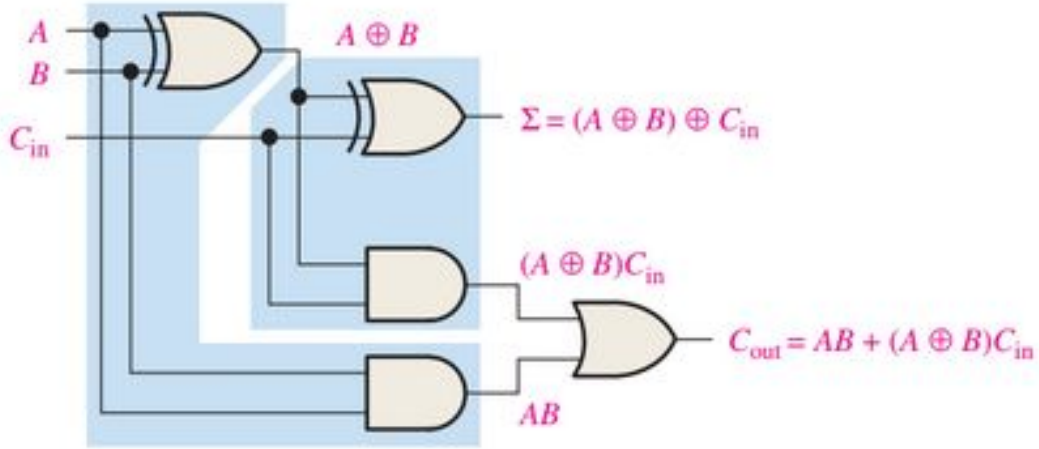
TABLE 6–2

Full-adder truth table.

A	B	C_{in}	C_{out}	Σ
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

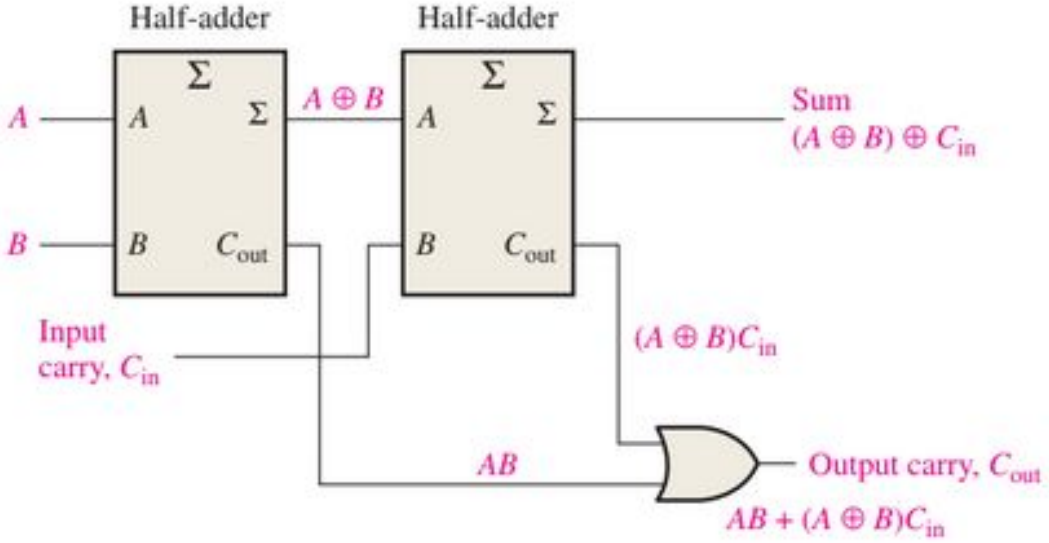
C_{in} = input carry, sometimes designated as CI
C_{out} = output carry, sometimes designated as CO
Σ = sum
A and B = input variables (operands)

Full Adder



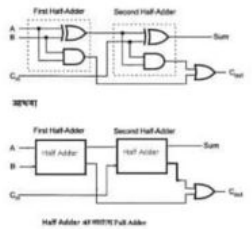
(b) Complete logic circuit for a full-adder (each half-adder is enclosed by a shaded area)

Full Adder from 2 Half Adder

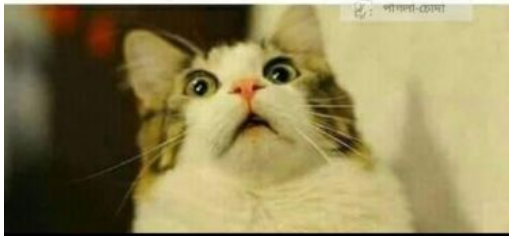


(a) Arrangement of two half-adders to form a full-adder

Sir-



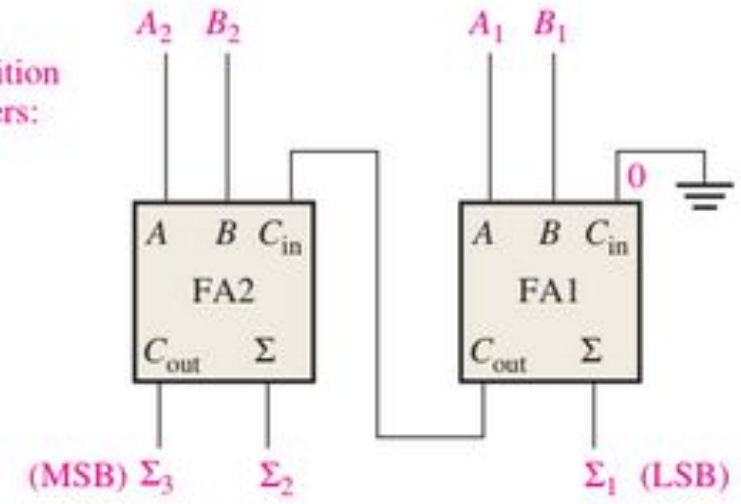
My Reaction-



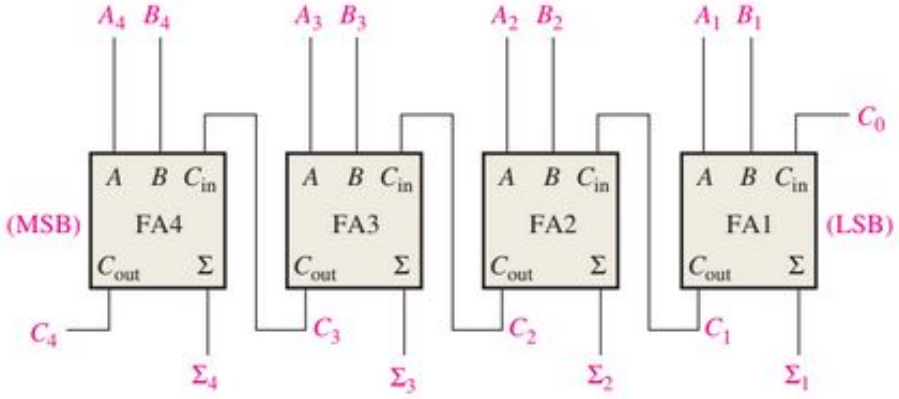
2 Bit Addition

General format, addition of two 2-bit numbers:

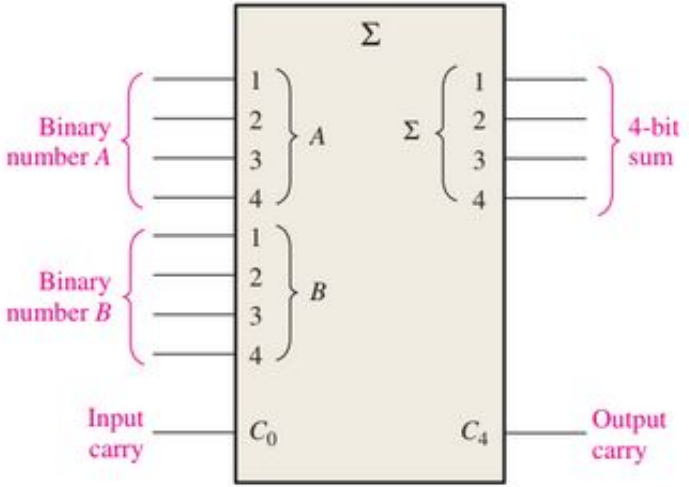
$$\begin{array}{r} A_2A_1 \\ +\ B_2B_1 \\ \hline \Sigma_3\Sigma_2\Sigma_1 \end{array}$$



4 Bit Addition



(a) Block diagram



(b) Logic symbol

FIGURE 6–9 A 4-bit parallel adder.

8 Bit Addition

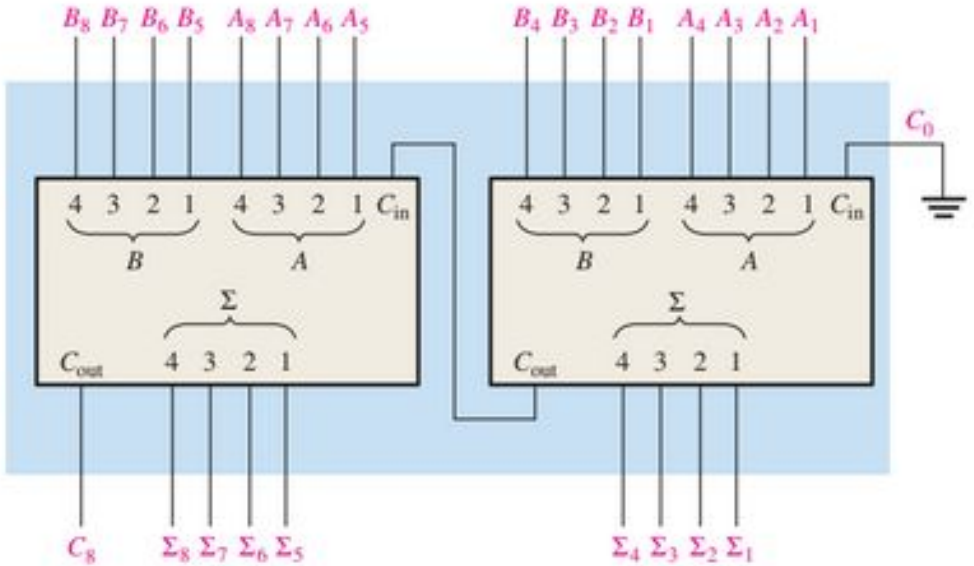
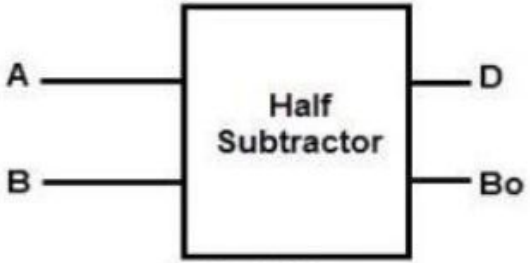


FIGURE 6–11 Cascading of two 4-bit adders to form an 8-bit adder.

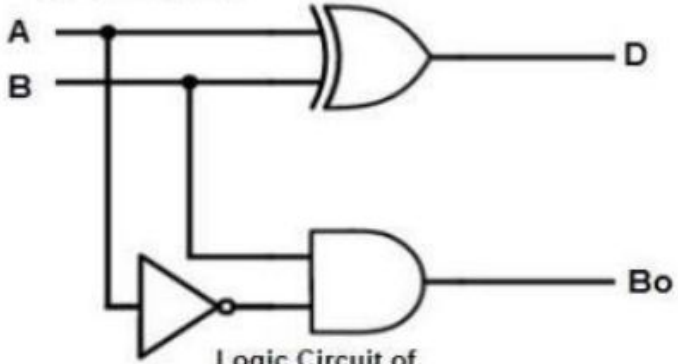
Half Subtractor



Block Diagram

A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Truth Table

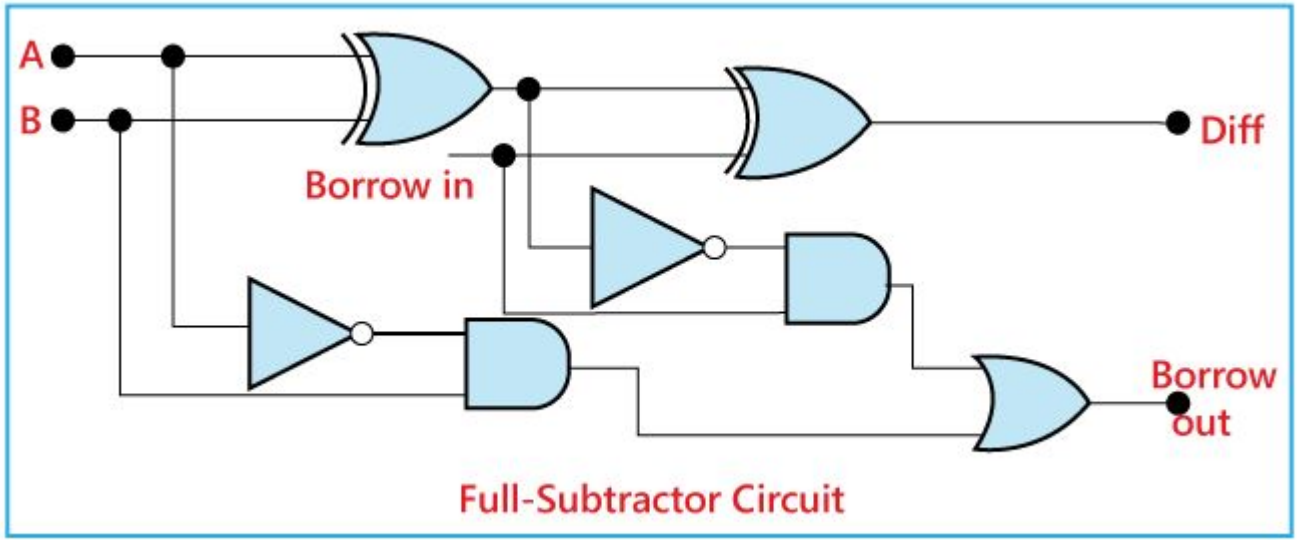


Logic Circuit of
Half Subtractor

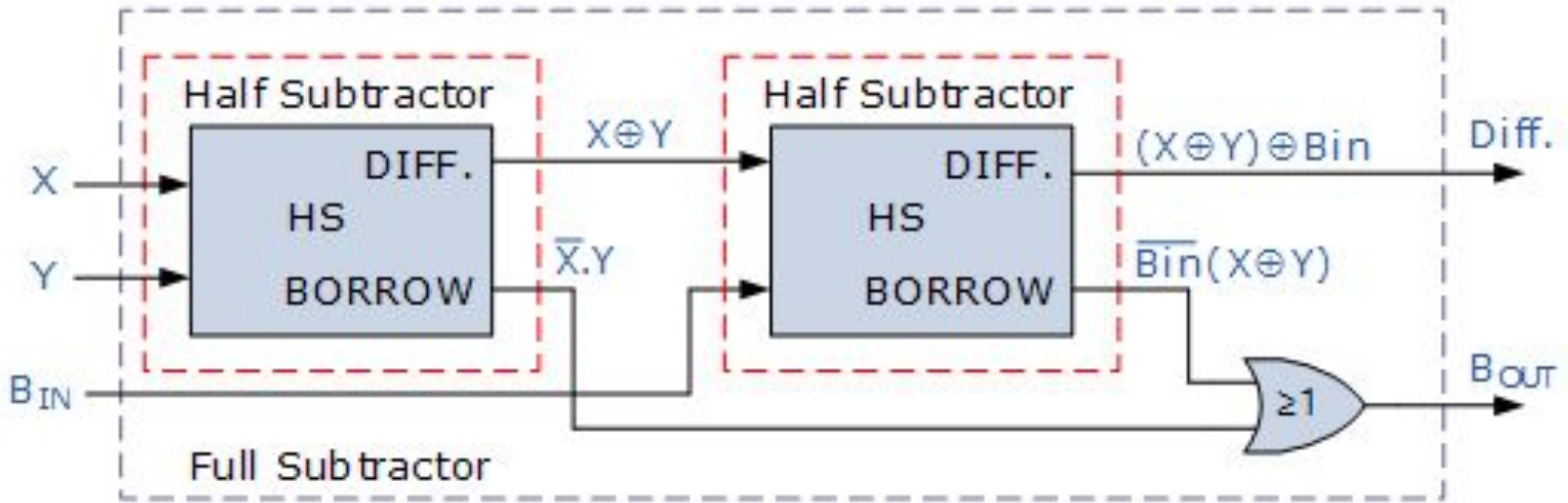
Full Subtractor

Inputs			Outputs	
A	B	$Borrow_{in}$	Diff	Borrow
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

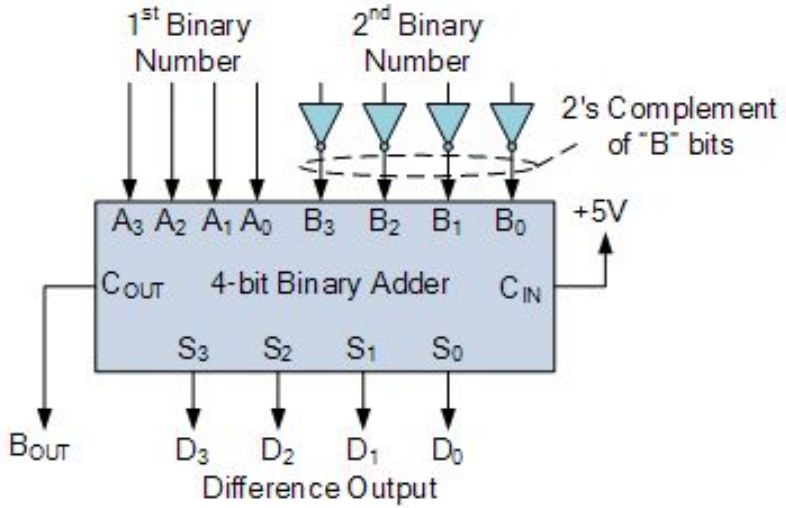
Full Subtractor



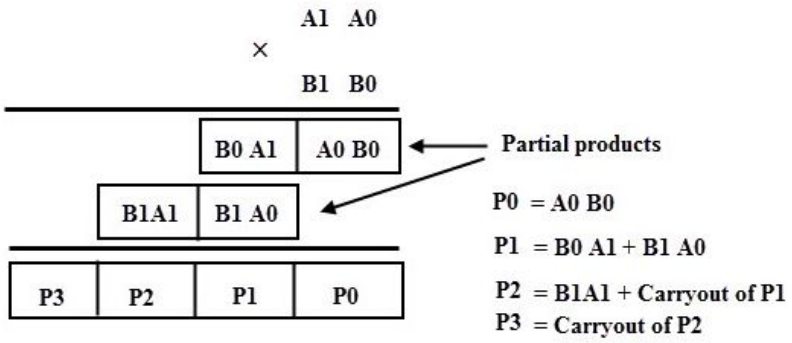
Full Subtractor From Half SUbtractor

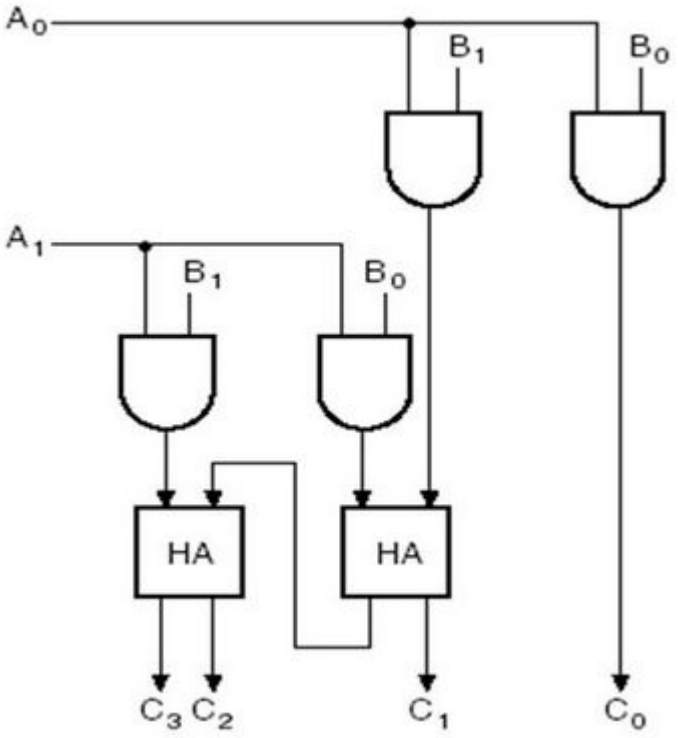


Binary Subtractor Using 2's Complement

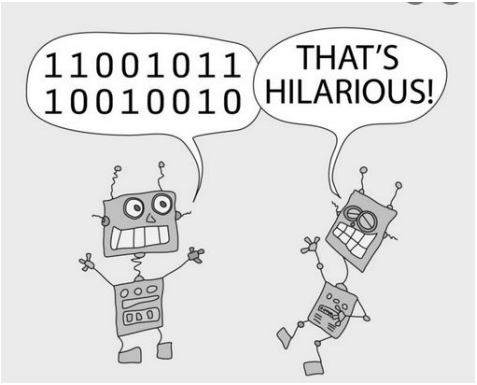


Binary Multiplication



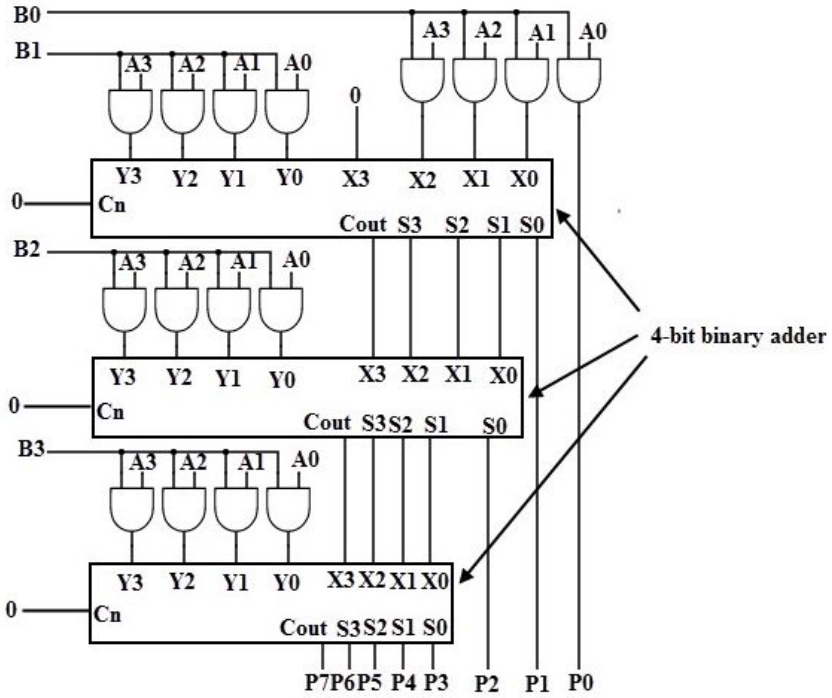


Binary Multiplication



		A3	A2	A1	A0	
	×	B3	B2	B1	B0	
		B0 A3	B0 A2	B0 A1	B0 A0	
	B1A3	B1 A2	B1A1	B1 A0		Partial products
B2A3	B2 A2	B2A1	B2 A0			
+ B3A3	B3 A2	B3A1	B3 A0			
P7	P6	P5	P4	P3	P2	P1	P0

Binary Multiplication



Ripple carry Adders

Solution is Look Ahead Carry Adder

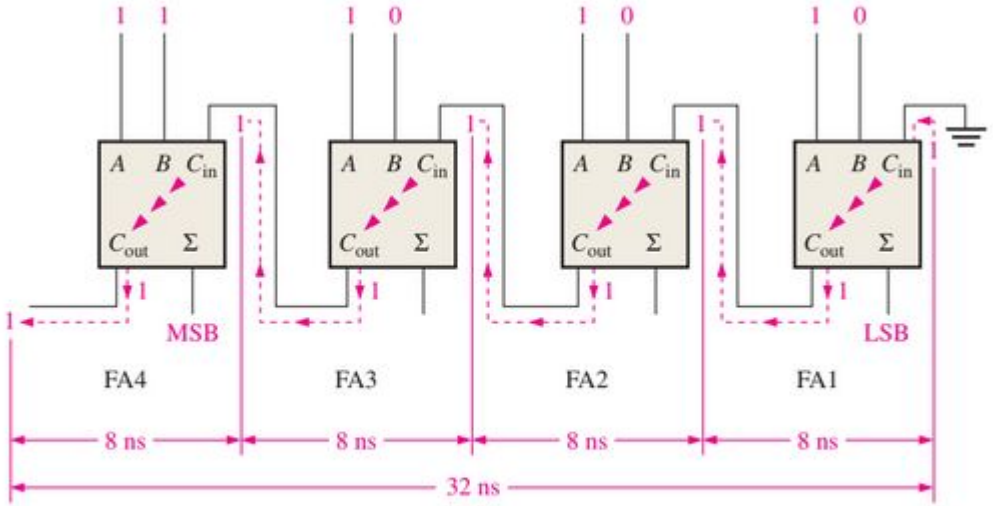


FIGURE 6–14 A 4-bit parallel ripple carry adder showing "worst-case" carry propagation delays.

Comparators

FIGURE 6–18 Basic comparator operation.

Comparators

EXAMPLE 6–5

Apply each of the following sets of binary numbers to the comparator inputs in Figure 6–20, and determine the output by following the logic levels through the circuit.

(a) 10 and 10 **(b)** 11 and 10

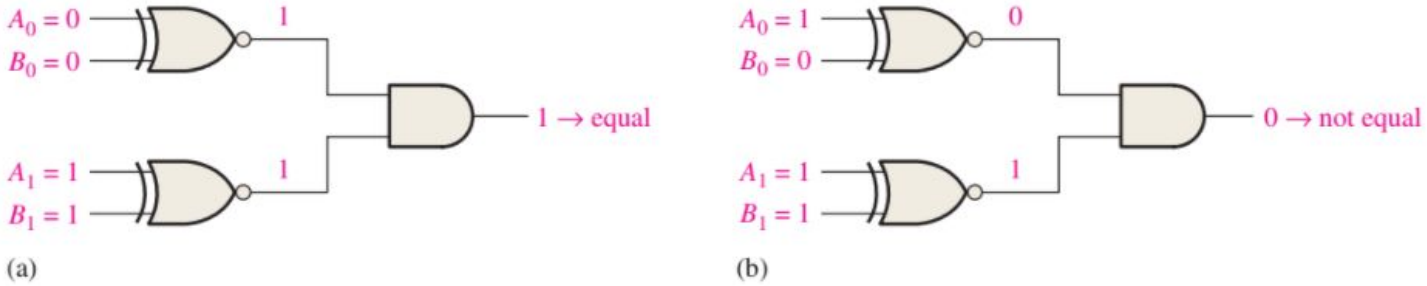
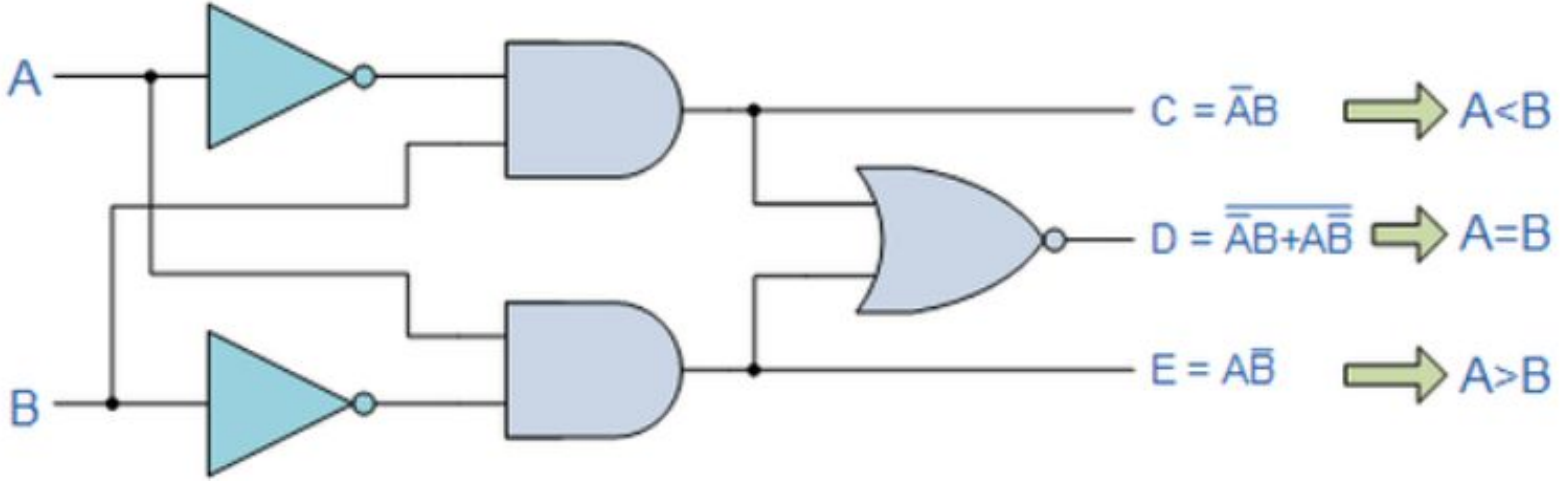


FIGURE 6–20

1 Bit Comparators



4 Bit Comparators

1. If A3 = 1 and B3 = 0, number A is greater than number B.

2. If A3 = 0 and B3 = 1, number A is less than number B.

3. If A3 = B3, then you must examine the next lower bit position for an inequality.

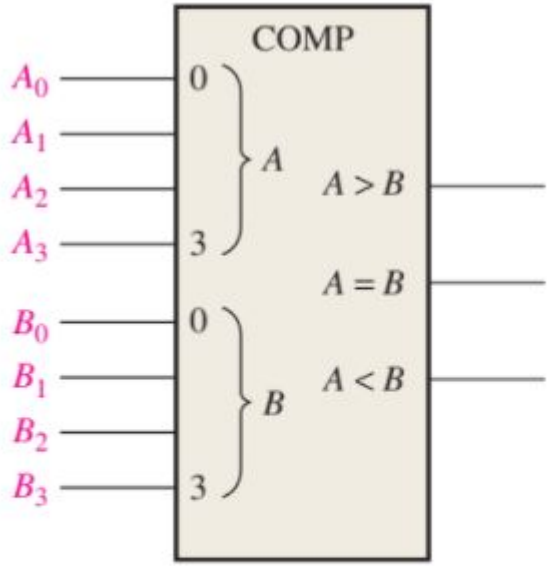


FIGURE 6–21 Logic symbol for a 4-bit comparator with inequality indication.

4 Bit Comparators

EXAMPLE 6–6

Determine the $A = B$, $A > B$, and $A < B$ outputs for the input numbers shown on the comparator in Figure 6–22.

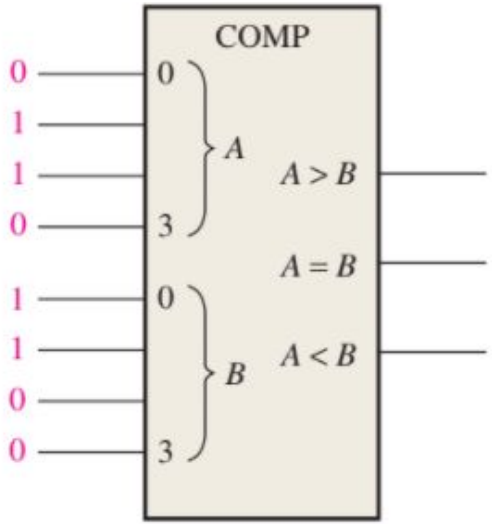


FIGURE 6–22

Decoders

Me : Ammi aj doston k sath sehri bahar karunga

Ammi:

Decoders

A decoder is a digital circuit that detects the presence of a specified combination of bits (code) on its inputs and indicates the presence of that code by a specified output level.

a decoder has n input lines to handle n bits and from one to 2^n output lines

Basic Binary Decoder

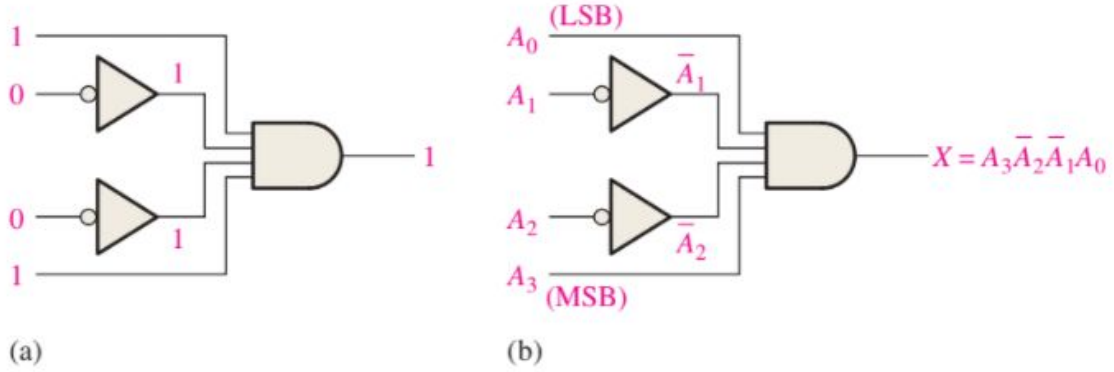


FIGURE 6–26 Decoding logic for the binary code 1001 with an active-HIGH output.

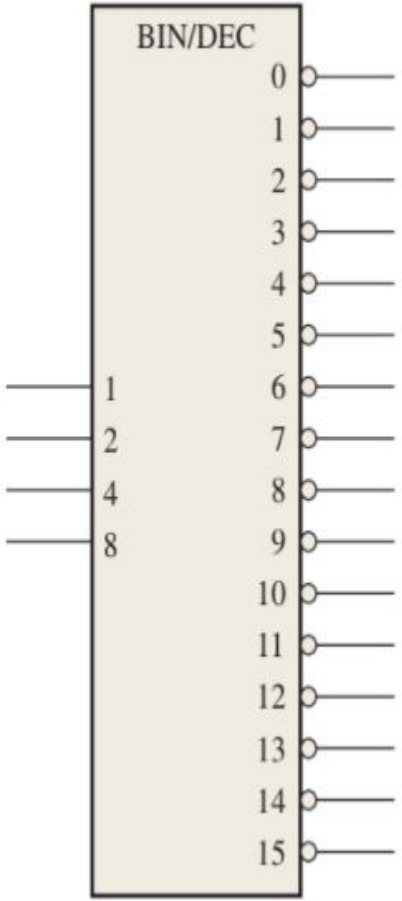
Determine the logic required to decode the binary number 1011 by producing a HIGH level on the output.

4 Bit Decoder

To decode all possible combinations of four bits, sixteen decoding gates are required (2^4 = 16).

4-line-to-16-line decoder because there are four inputs and sixteen outputs

1-of-16 decoder because for any given code on the inputs, one of the sixteen outputs is activated.



4 Bit Decoder

TABLE 6–4

Decoding functions and truth table for a 4-line-to-16-line (1-of-16) decoder with active-LOW outputs.

Decimal Digit	Binary Inputs A_3	A_2	A_1	A_0	Decoding Function	Outputs 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
10	1	0	1	0	$A_3\overline{A}_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
11	1	0	1	1	$A_3\overline{A}_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
12	1	1	0	0	$A_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
13	1	1	0	1	$A_3A_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
14	1	1	1	0	$A_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
15	1	1	1	1	$A_3A_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Home Task

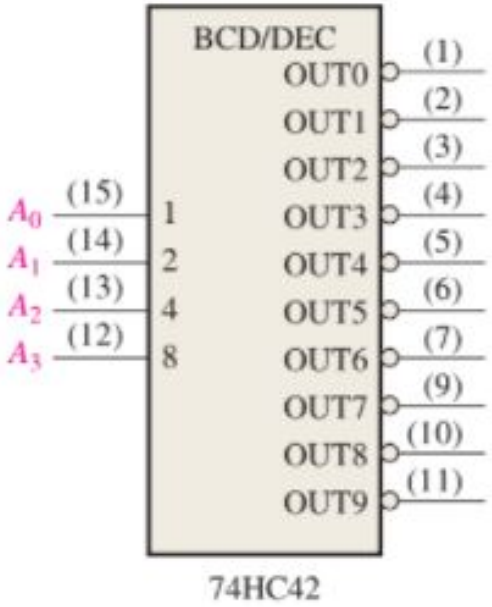
Design a 3-line-to-8-line decoder can be used for binary-to-octal decoding.

BCD To DECIMAL DECODER

TABLE 6–5

BCD decoding functions.

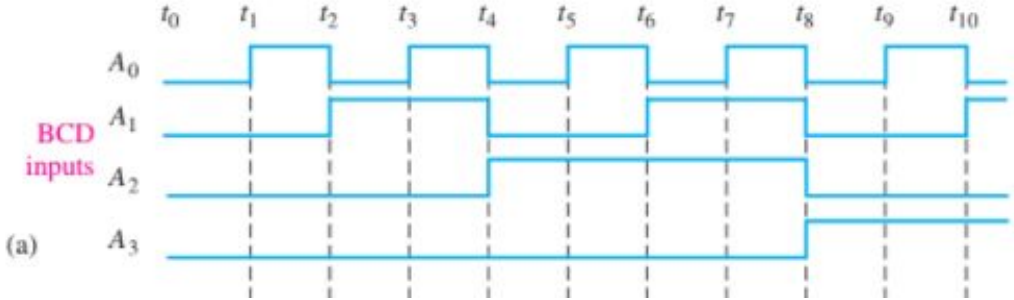
Decimal Digit	BCD Code A_3	A_2	A_1	A_0	Decoding Function
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$



BCD To DECIMAL DECODER

EXAMPLE 6–10

If the input waveforms in Figure 6–32(a) are applied to the inputs of the 74HC42, show the output waveforms.



BCD To DECIMAL DECODER

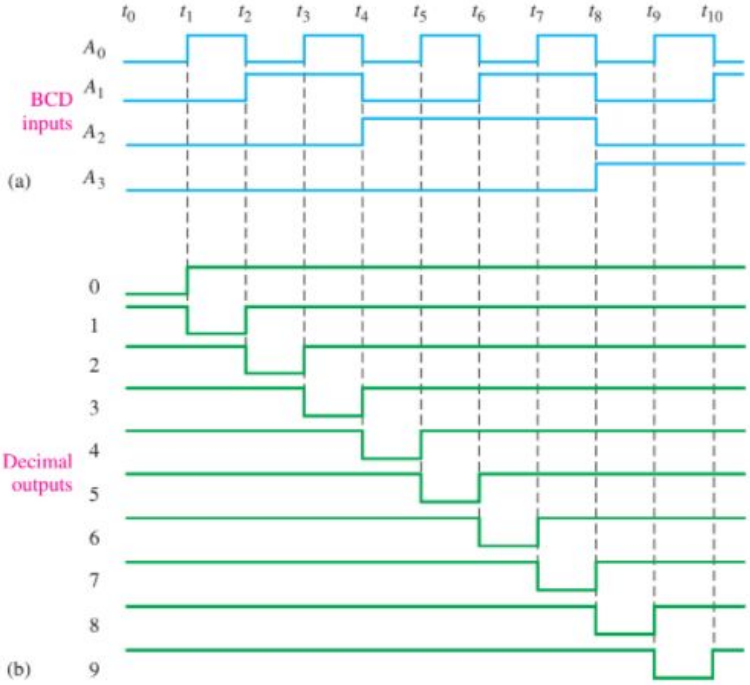


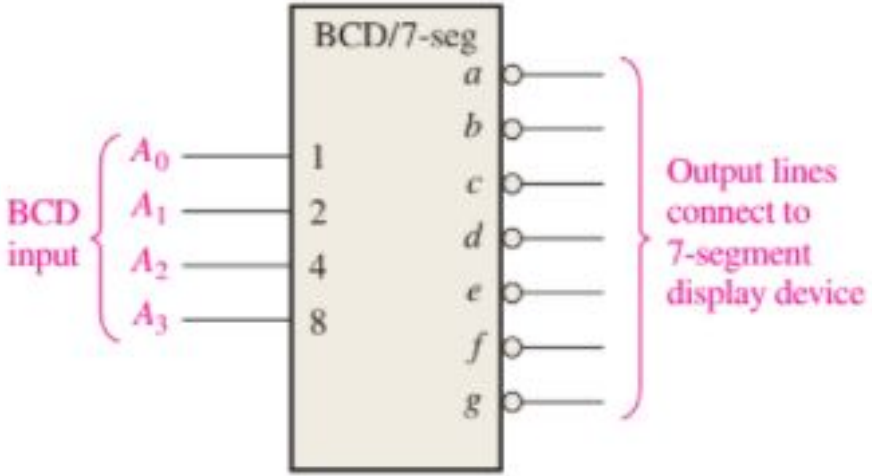
FIGURE 6–32

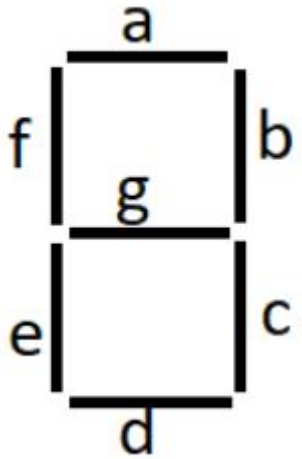
BCD To DECIMAL DECODER

Related Problem

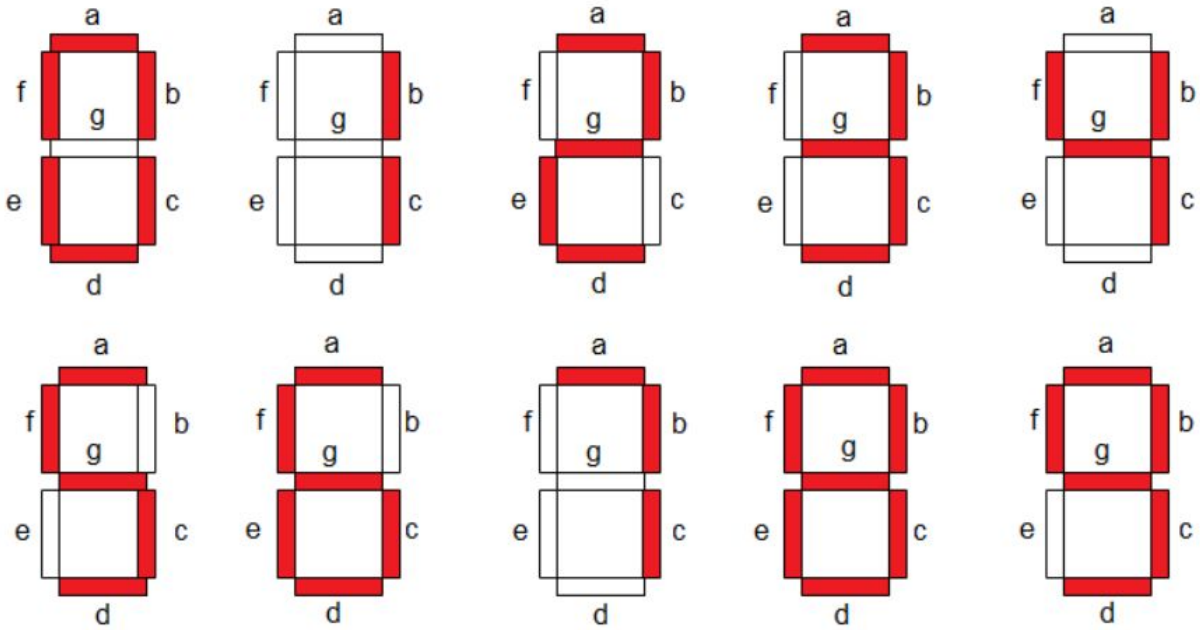
Construct a timing diagram showing input and output waveforms for the case where the BCD inputs sequence through the decimal numbers as follows: 0, 2, 4, 6, 8, 1, 3, 5, and 9.

BCD To 7-Segment Decoder





BCD To 7-Segment Decoder

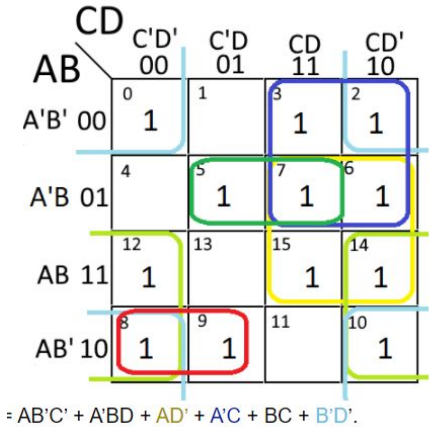


BCD To 7-Segment Decoder

Inputs				Segments							
A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	1	1	1	1	1	1	0	For display 0
0	0	0	1	0	1	1	0	0	0	0	For display 1
0	0	1	0	1	1	0	1	1	0	1	For display 2
0	0	1	1	1	1	1	1	0	0	1	For display 3
0	1	0	0	0	1	1	0	0	1	1	For display 4
0	1	0	1	1	0	1	1	0	1	1	For display 5
0	1	1	0	1	0	1	1	1	1	1	For display 6
0	1	1	1	1	1	1	0	0	0	0	For display 7
1	0	0	0	1	1	1	1	1	1	1	For display 8
1	0	0	1	1	1	1	1	0	1	1	For display 9
1	0	1	0	1	1	1	0	1	1	1	For display A
1	0	1	1	0	0	1	1	1	1	1	For display b
1	1	0	0	1	0	0	1	1	1	0	For display C
1	1	0	1	0	1	1	1	1	0	1	For display d
1	1	1	0	1	0	0	1	1	1	1	For display E
1	1	1	1	1	0	0	0	1	1	1	For display F

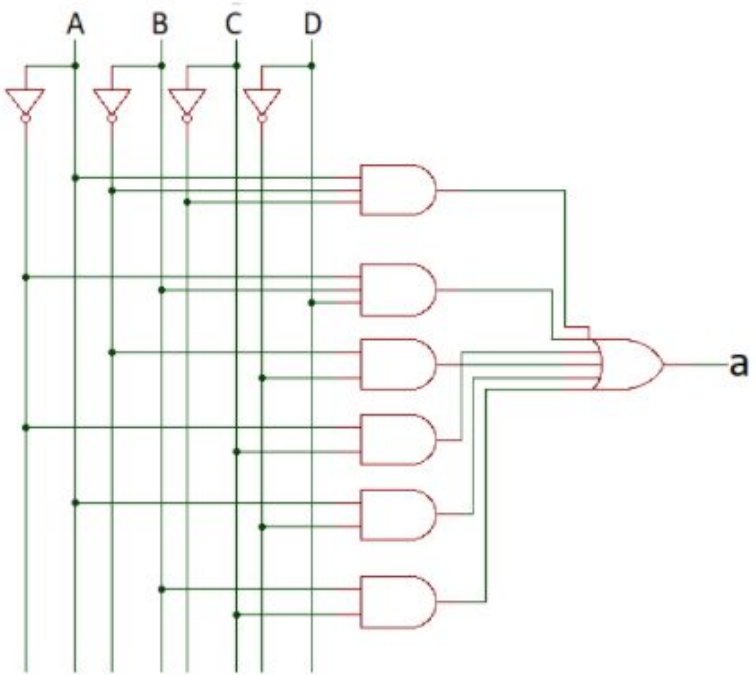
BCD To 7-Segment Decoder

For segment ‘a’

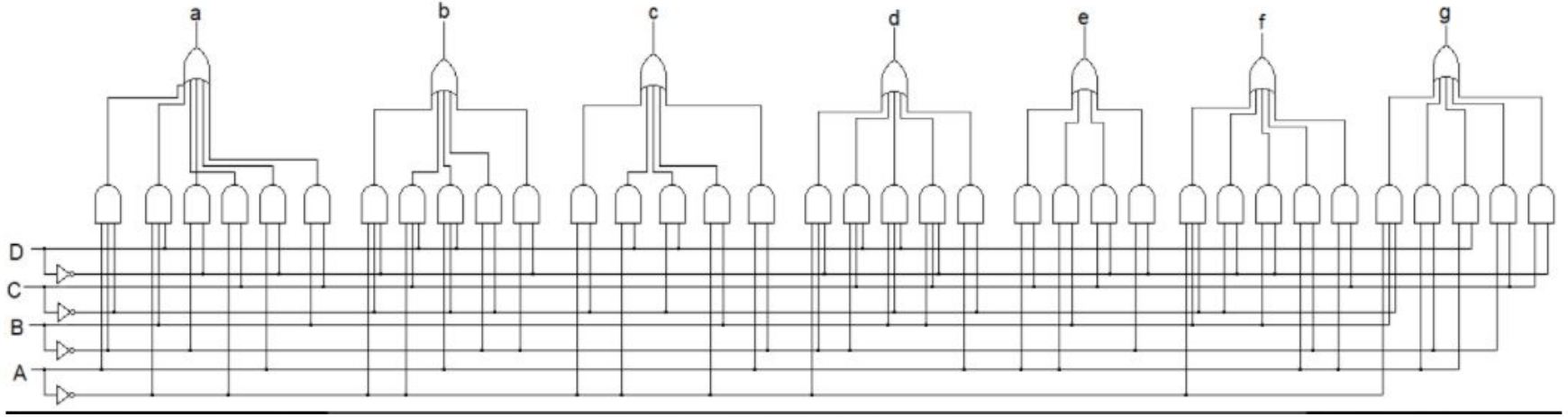


= AB'C' + A'BD + AD' + A'C + BC + B'D'.

Inputs				Segment
A	B	C	D	a
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1



BCD To 7-Segment Decoder



https://electronics-fun.com/7-segment-hex-decoder/

Encoders

Reverse Decoder

The process of converting from familiar symbols or numbers to a coded format is called encoding.

Applications: Keypad-binary numbers

Decimal to BCD Encoders

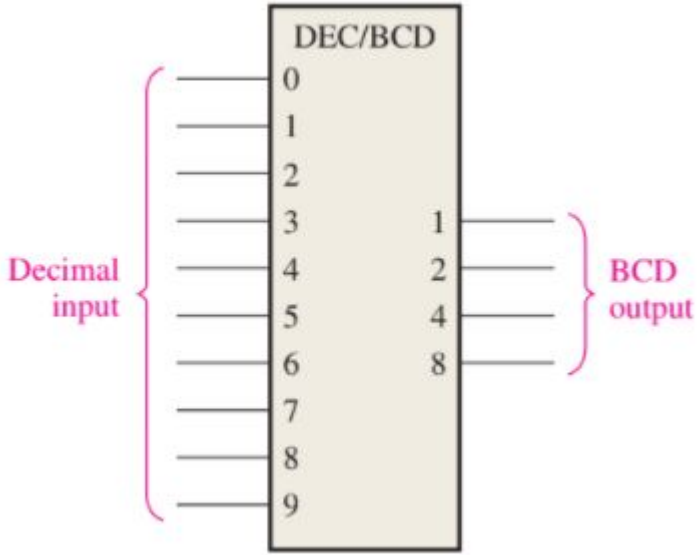


TABLE 6–6

Decimal Digit	BCD Code A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Decimal to BCD Encoders

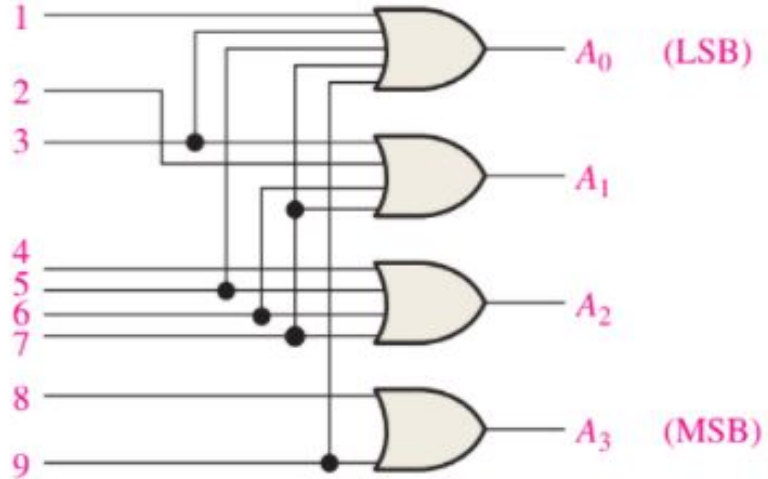


TABLE 6–6

Decimal Digit	BCD Code A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Decimal to BCD Priority Encoders

The priority function means that the encoder will produce a BCD output corresponding to the highest-order decimal digit input that is active.

For instance, if the 6 and the 3 inputs are both active, the BCD output is 0110 (which represents decimal 6).

Code Converter

BCD-to-Binary Conversion:

1. The value, or weight, of each bit in the BCD number is represented by a binary number.

2. All of the binary representations of the weights of bits that are 1s in the BCD number are added.

3. The result of this addition is the binary equivalent of the BCD number.

The binary numbers representing the weights of the BCD bits are summed to produce the total binary number.

BCD-to-Binary Conversion

$$\underbrace{1000}_{8} \qquad \underbrace{0111}_{7}$$

	Tens Digit				Units Digit			
Weight:	80	40	20	10	8	4	2	1
Bit designation:	B_3	B_2	B_1	B_0	A_3	A_2	A_1	A_0

TABLE 6-7

Binary representations of BCD bit weights.

BCD Bit	BCD Weight	(MSB) Binary Representation						(LSB)
		64	32	16	8	4	2	1
A_0	1	0	0	0	0	0	0	1
A_1	2	0	0	0	0	0	1	0
A_2	4	0	0	0	0	1	0	0
A_3	8	0	0	0	1	0	0	0
B_0	10	0	0	0	1	0	1	0
B_1	20	0	0	1	0	1	0	0
B_2	40	0	1	0	1	0	0	0
B_3	80	1	0	1	0	0	0	0

Binary-to-Gray and Gray-to-Binary Conversion

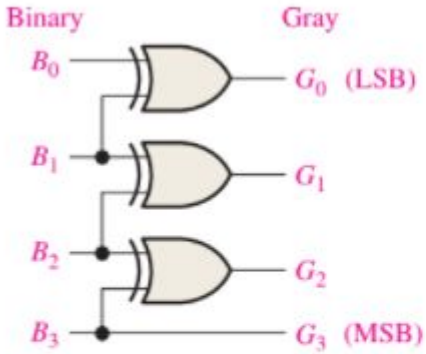
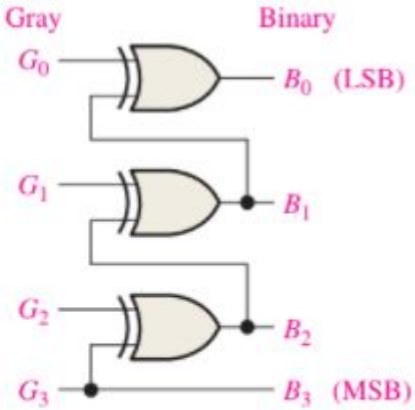


FIGURE 6–40 Four-bit binary-to-Gray conversion logic. Open file F06-40 to verify operation.



MultiSim

FIGURE 6–41 Four-bit Gray-to-binary conversion logic. Open file F06-41 to verify operation.

How many exclusive-OR gates are required to convert 8-bit binary to Gray?

Multiplexers (MUX)

Allows digital information from several sources to be routed onto a single line for transmission over that line to a common destination.

It has several data-input lines and a single output line.

It also has data-select inputs, which permit digital data on any one of the inputs to be switched to the output line.

Also known as data selectors.

Applications: Router, Telephone Network, Computer memory

MultiPlexers (MUX)

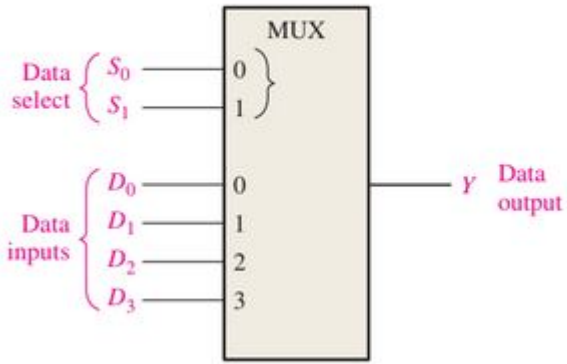


TABLE 6–8

Data selection for a 1-of-4-multiplexer.

Data-Select Inputs		
S_1	S_0	Input Selected
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

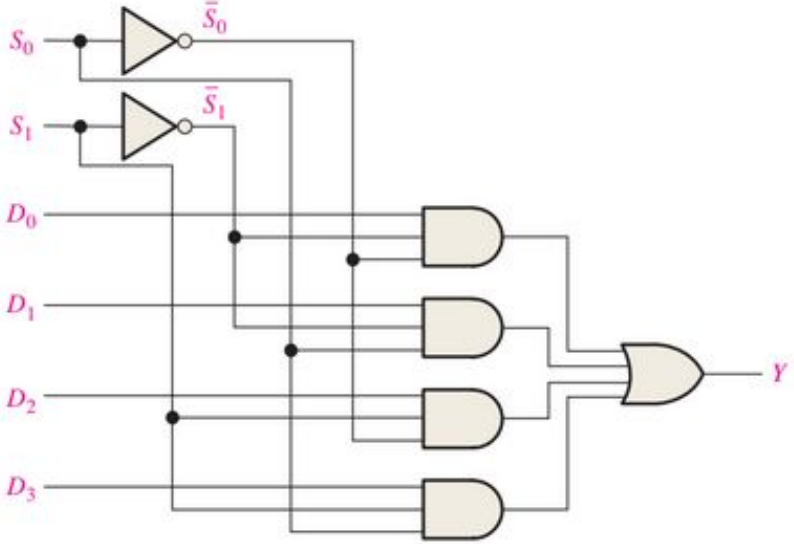
The data output is equal to D_0 only if $S_1 = 0$ and $S_0 = 0$: $Y = D_0\overline{S}_1\overline{S}_0$.

The data output is equal to D_1 only if $S_1 = 0$ and $S_0 = 1$: $Y = D_1\overline{S}_1S_0$.

The data output is equal to D_2 only if $S_1 = 1$ and $S_0 = 0$: $Y = D_2S_1\overline{S}_0$.

The data output is equal to D_3 only if $S_1 = 1$ and $S_0 = 1$: $Y = D_3S_1S_0$.

MultiPlexers (MUX)



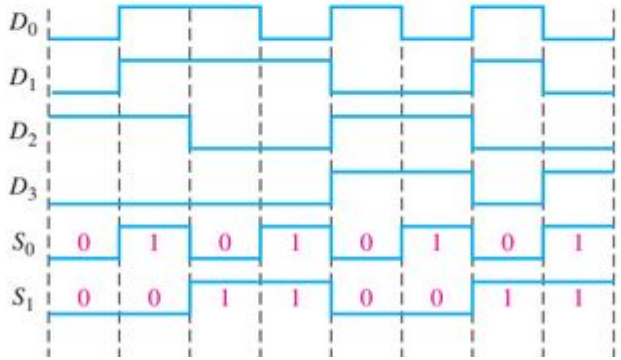
$$Y = D_0\bar{S}_1\bar{S}_0 + D_1\bar{S}_1S_0 + D_2S_1\bar{S}_0 + D_3S_1S_0$$

MultiPlexers (MUX)

EXAMPLE 6–14

The data-input and data-select waveforms in Figure 6–45(a) are applied to the multiplexer in Figure 6–44. Determine the output waveform in relation to the inputs.

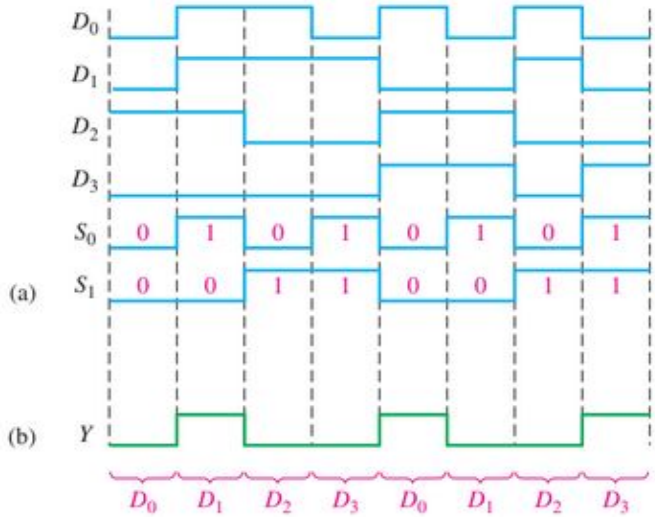
(a)



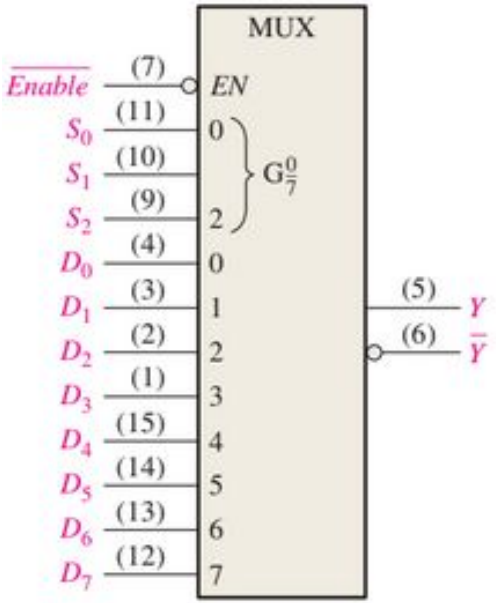
MultiPlexers (MUX)

EXAMPLE 6–14

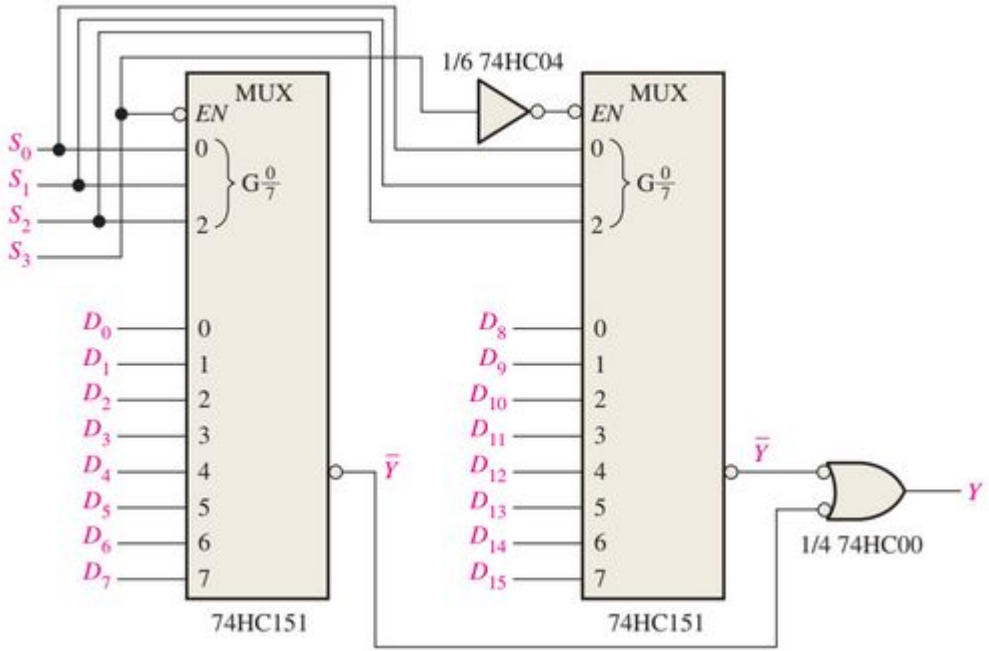
The data-input and data-select waveforms in Figure 6–45(a) are applied to the mul
plexer in Figure 6–44. Determine the output waveform in relation to the inputs.



8 Bit MUX IC


MUX
$\overline{Enable}$
(7)
EN
S_0
(11)
0
S_1
(10)
S_2
(9)
2
$G_{\overline{7}}^{0}$
D_0
(4)
0
D_1
(3)
1
D_2
(2)
2
D_3
(1)
3
D_4
(15)
4
D_5
(14)
5
D_6
(13)
6
D_7
(12)
7
(5)
Y
(6)
$\overline{Y}$

16 Bit Mux using 8 Bit Mux IC



Home Task

Make a block diagram to implement 16 bit MUX using 4 bit MUX IC.

MultiPlexers (MUX)

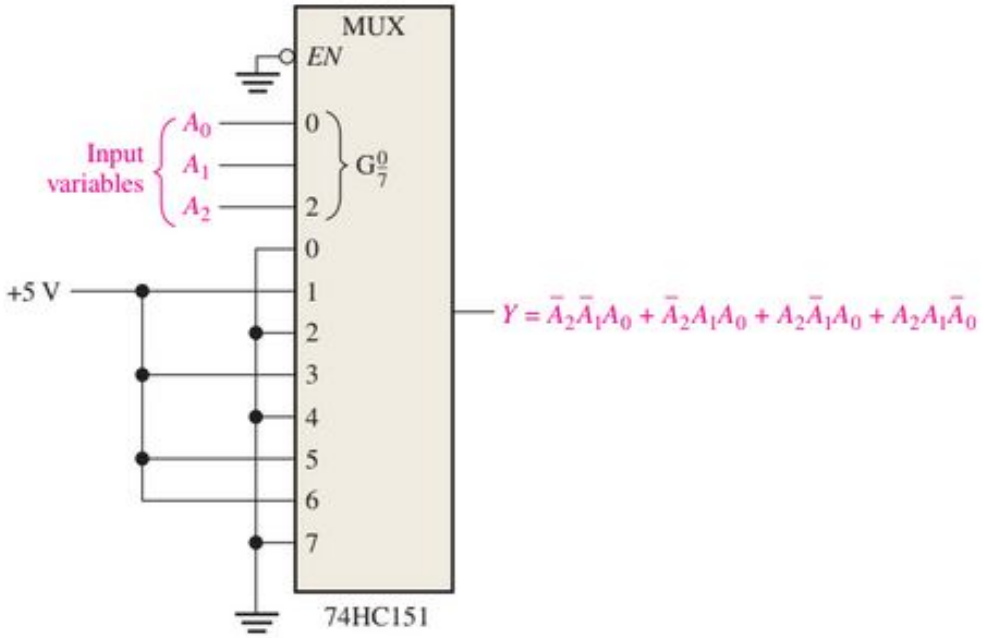
EXAMPLE 6–16

Implement the logic function specified in Table 6–9 by using a 74HC151 8-input data selector/multiplexer. Compare this method with a discrete logic gate implementation.

TABLE 6–9

Inputs			Output
A_2	A_1	A_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

MultiPlexers (MUX)



MultiPlexers (MUX)

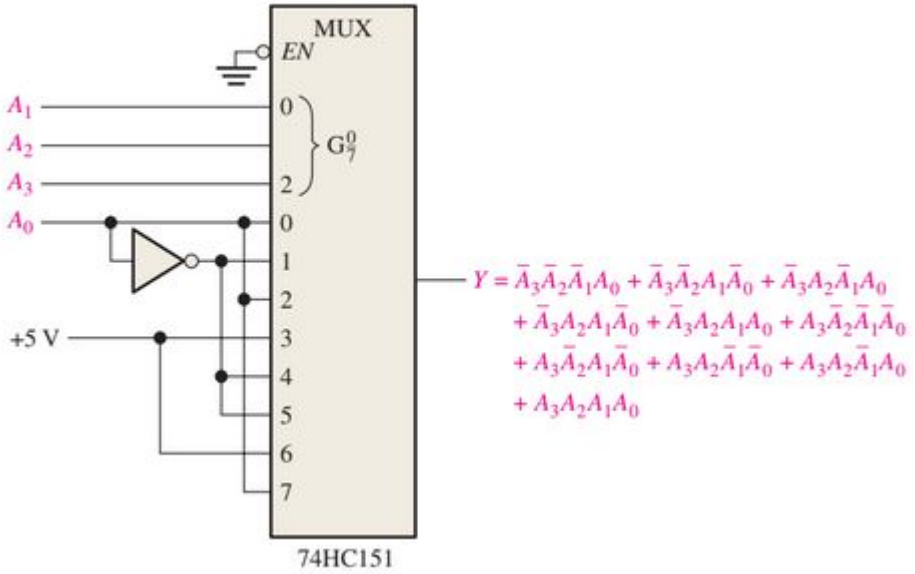
EXAMPLE 6–17

Implement the logic function in Table 6–10 by using a 74HC151 8-input data selector/multiplexer. Compare this method with a discrete logic gate implementation.

TABLE 6–10

Decimal Digit	Inputs				Output
	A_3	A_2	A_1	A_0	Y
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

MultiPlexers (MUX)

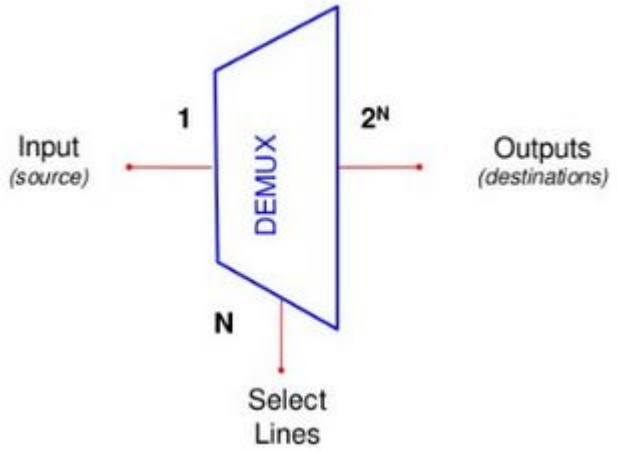


De-Multiplexers (DEMUX)

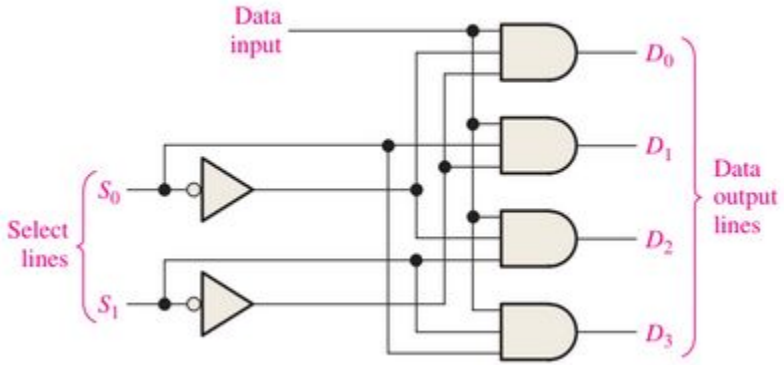
A (DEMUX) basically is reverse multiplexer.

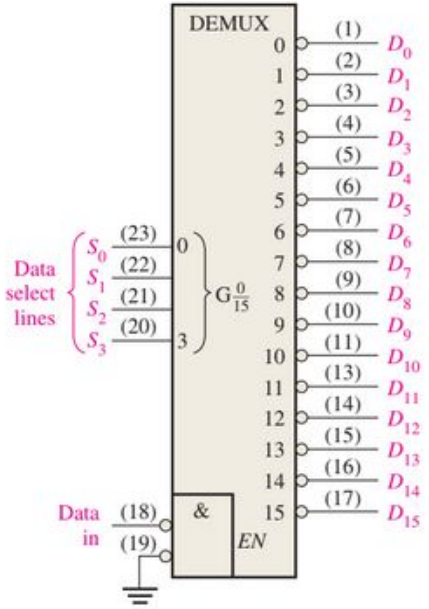
It takes digital information from one line and distributes it to a given number of output lines.

Also known as a data distributor.



De-Multiplexers (DEMUX)





The decoder used as a demultiplexer.

Applications: Router, FM radio, Serial-to-Parallel Converter