Function of Combinational Logic

ENT263

Chapter Objectives

- Distinguish between half-adder and full-adder
- Use BCD-to-7-segment decoders in display systems
- Apply multiplexer in data selection
- Use decoders as multiplexer
- ..... and more...

Half-Adder

Simple Binary Addition

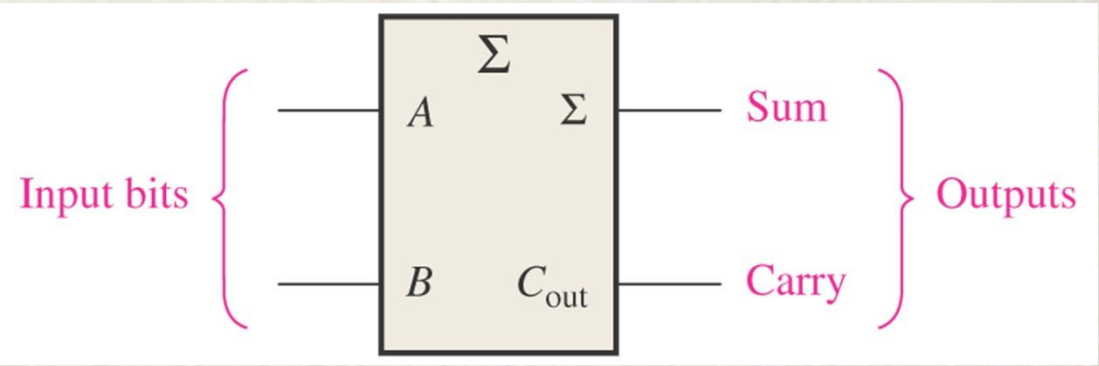
0 + 0 = 0	Zero plus zero equals zero
0 + 1 = 1	Zero plus one equals one
1 + 0 = 1	One plus zero equals one
1 + 1 = 10	One plus one equals zero with a carry of one

Basic Adder

Adder are important in computers and also in other types of digital systems in which numerical data are processed

Definition:

The half-adder accepts two binary digits on its inputs and produces two binary digits on its outputs, a sum bit and a carry bit



Half-Adder Logic

$C_{out} = AB$

$\Sigma = A \oplus B$

Combinational Logic

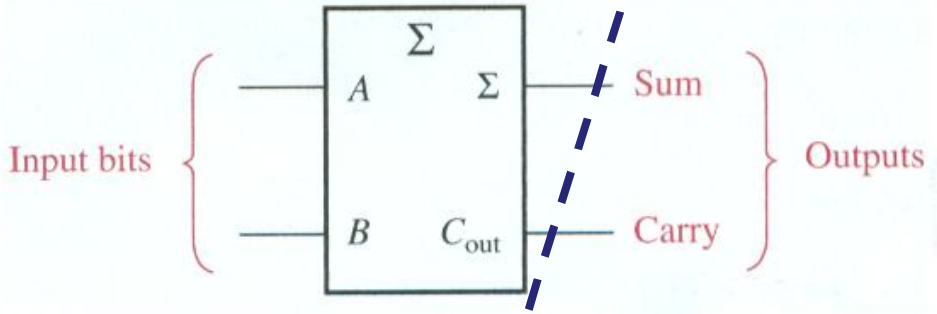
A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

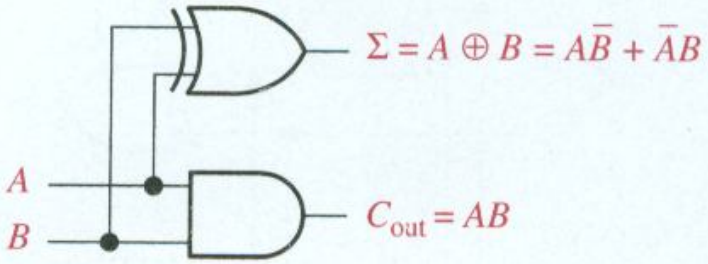
Σ = sum

C_{out} = output carry

A and B = input variables (operands)

Half-Adder Truth Table

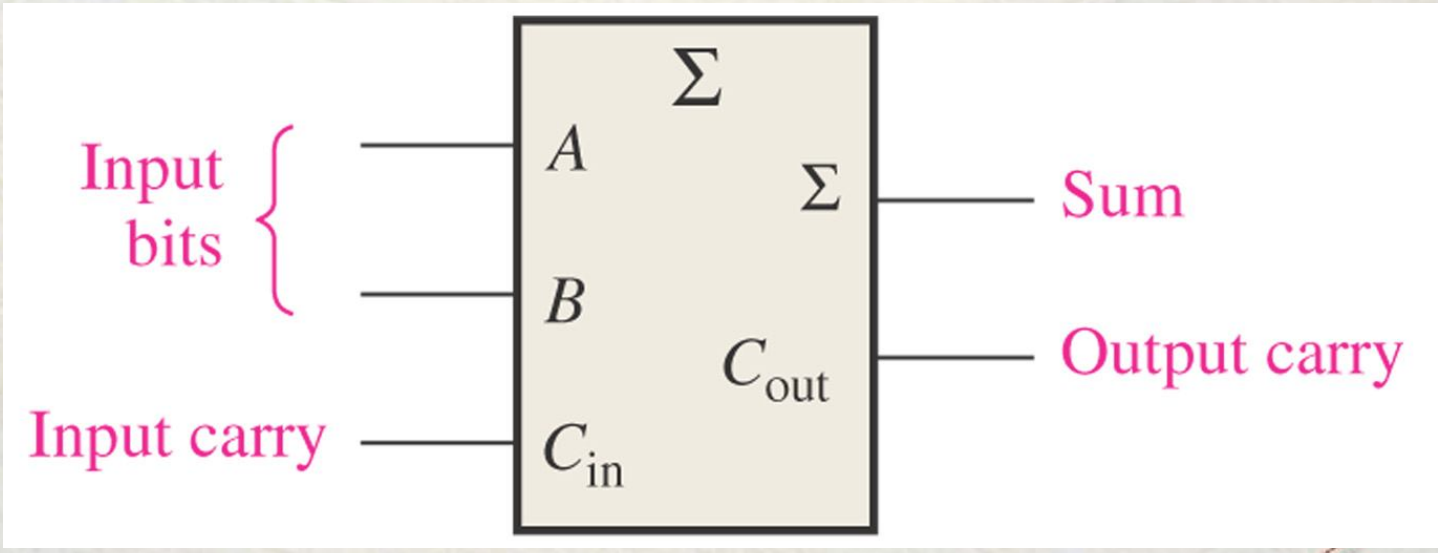




The Full-Adder

Definition: The Full-Adder accepts two input bits and an input carry and generates a sum output and an output carry

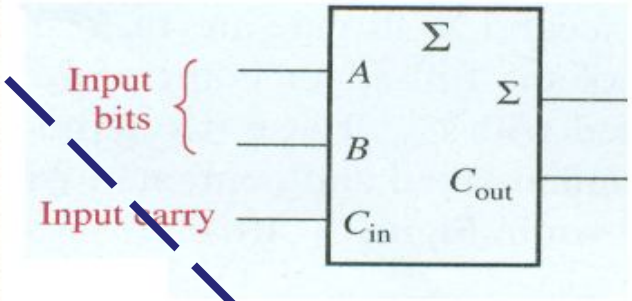
The basic different between a full-adder and a half-adder is that the full-adder accepts an input carry.



The full-adder must add the two input bits and the input carry. From the half-adder, the sum of the input bits A and B is the exclusive-OR of those two variables. For the input carry (C_{in}) to be added to the input bits, it must be exclusive-ORed, and last yield the equation for the sum output of the full-adder

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)

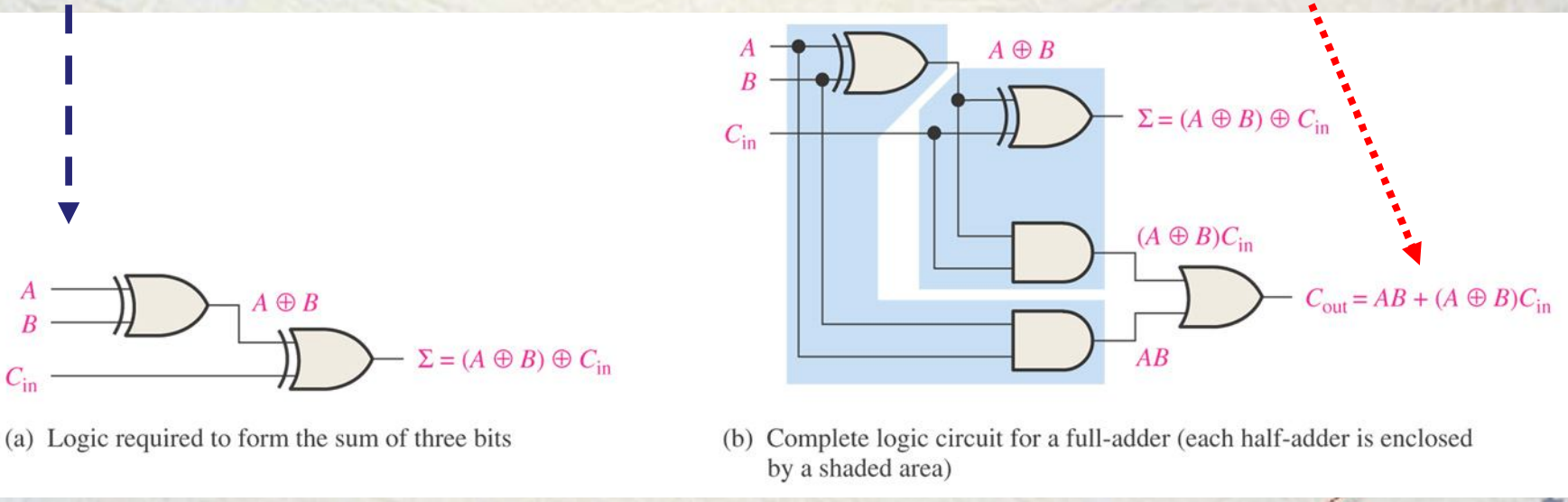


$$\Sigma = (A \oplus B) \oplus C_{in}$$

$$C_{out} = AB + (A \oplus B)C_{in}$$

This is mean that to implement the full-adder sum function, two 2-input exclusive-OR gates can be used . The first must generate the term A⊕B and the second has as its inputs the output of the first XOR gate and the input carry.

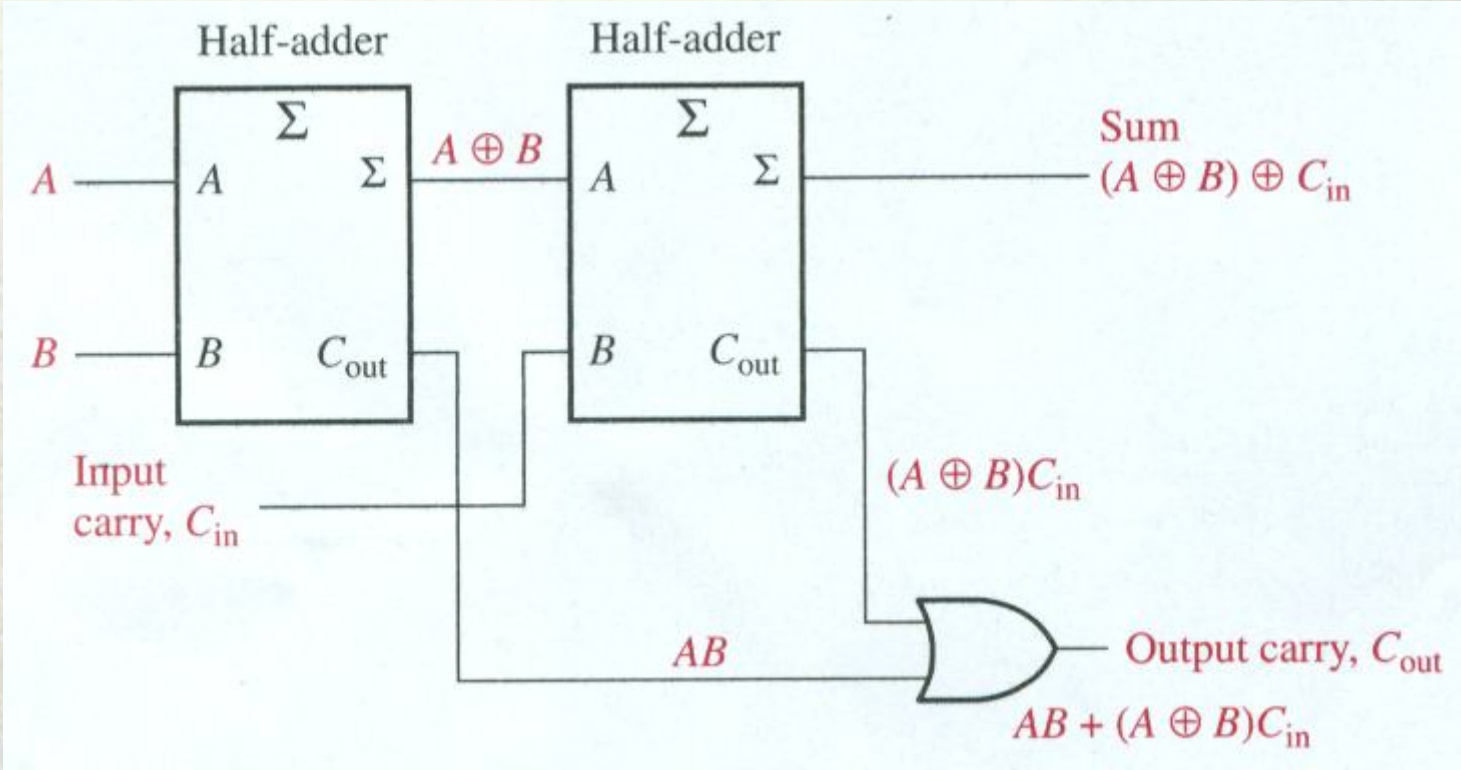
The output carry is a 1 when both inputs to the first XOR gate are 1s or when both inputs to the second XOR gate are 1s. The output carry of full-adder is therefore produced by the inputs A ANDed with B and A⊕B ANDed with C_{in}.



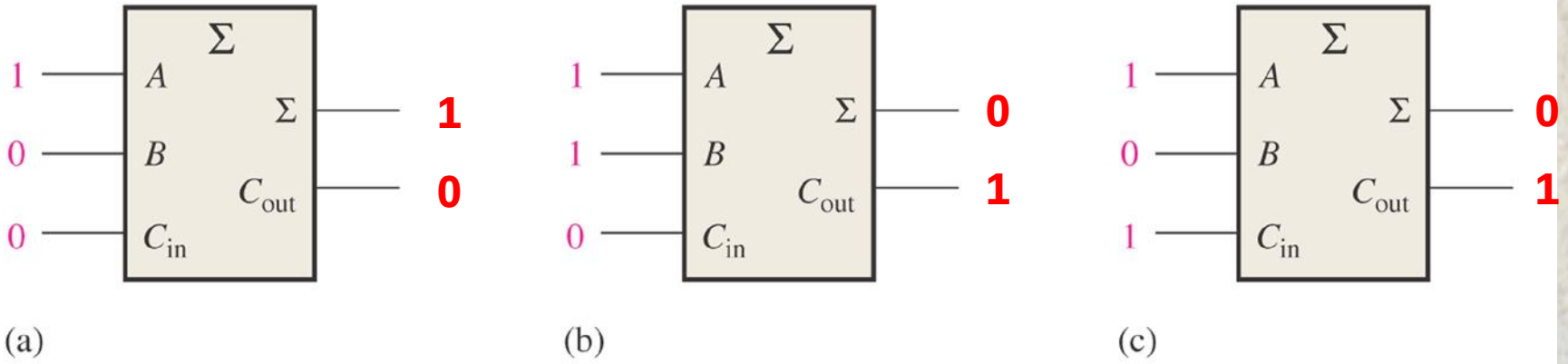
(a) Logic required to form the sum of three bits

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

Full Adder from Two Half-Adder Circuits



Example: Determine the outputs for the inputs shown



If A = 1, B = 1 and C_{in} = 1 ????

Σ = 1

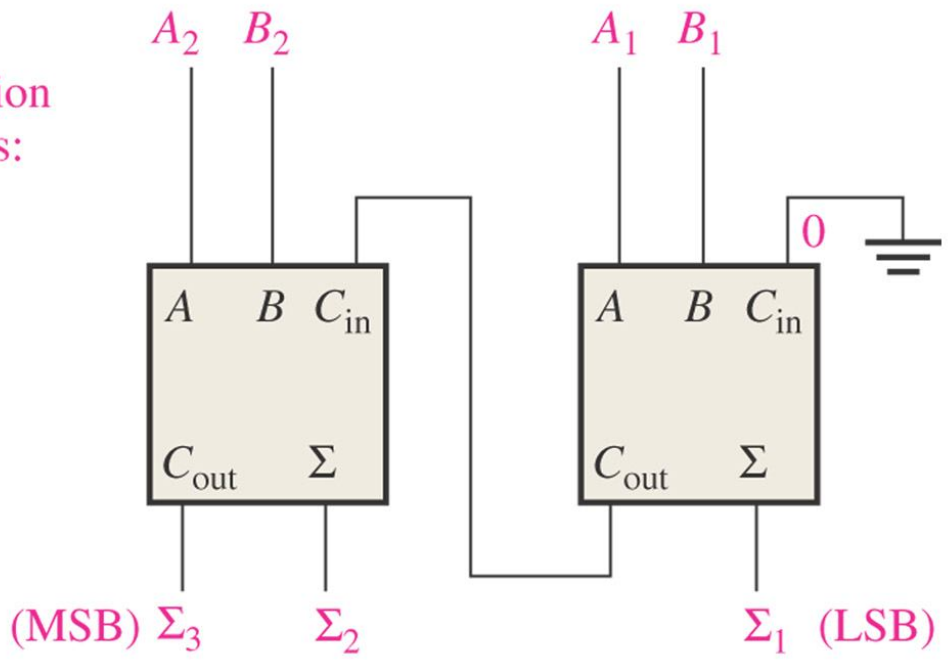
C_{out} = 1

Parallel Binary Adder

A single full-adder is capable of adding two 1-bit numbers and an input carry. To add binary numbers with more than one bit, we must use additional full-adders.

General format, addition of two 2-bit numbers:

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



1 bit – 1FA

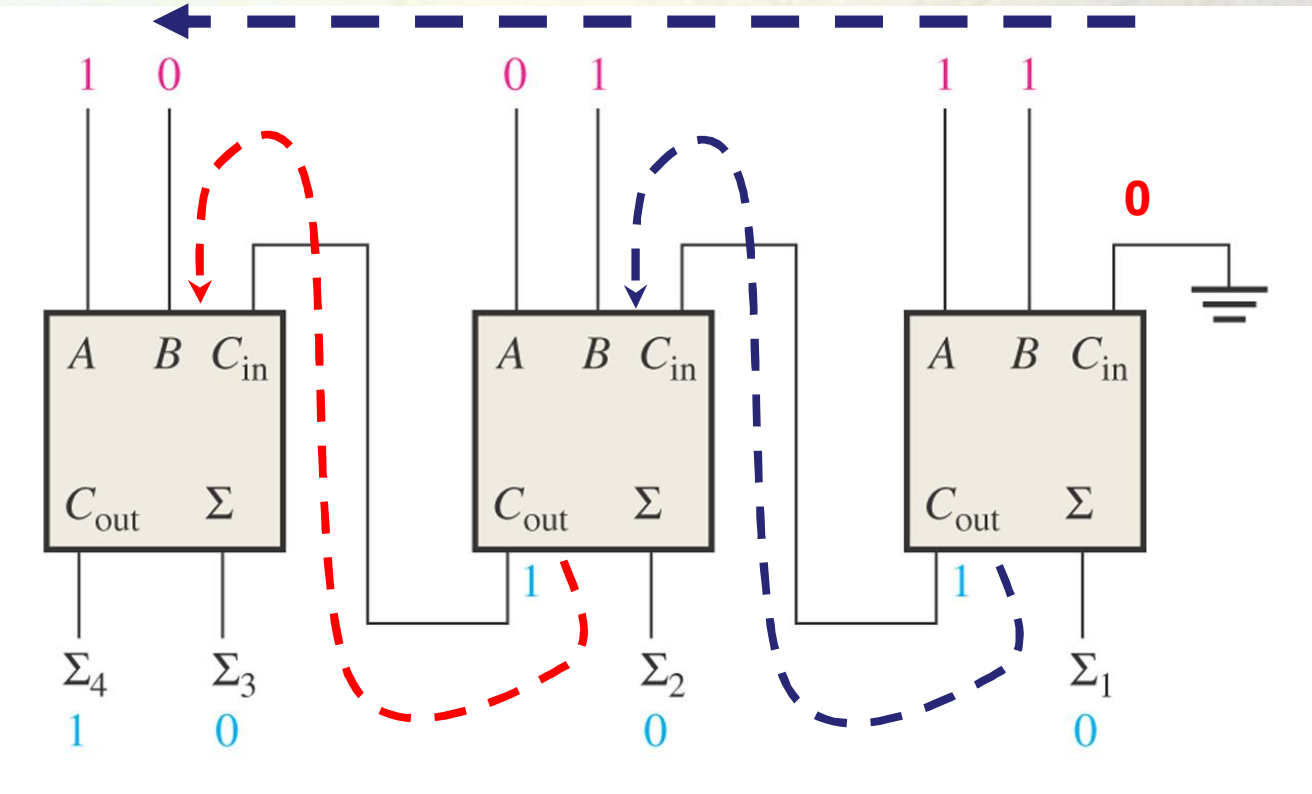
2 bit – 2FA

3 bit – 3FA

4 bit – 4FA

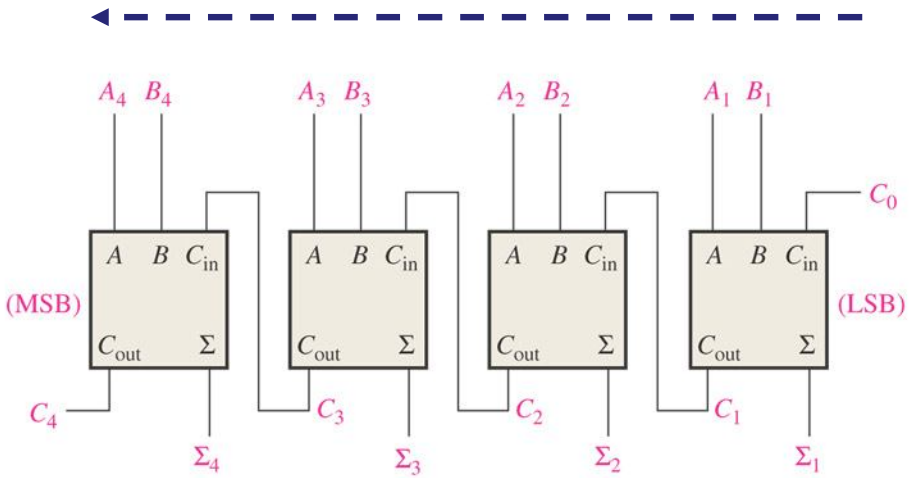
..and so on..

Example: Determine the sum generated by the 3-bit parallel adder

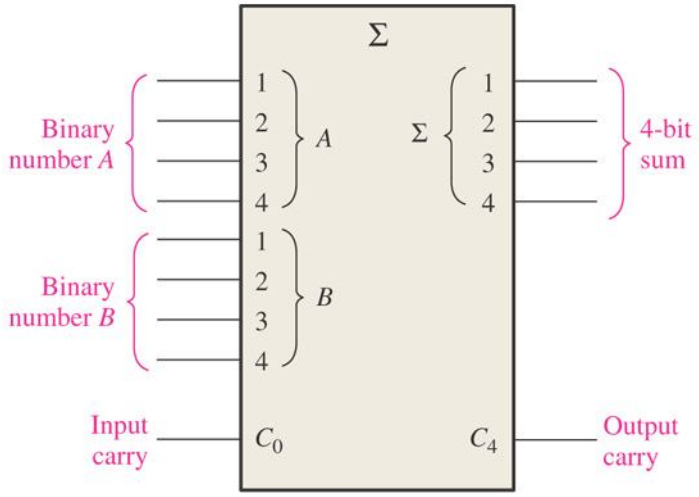


Four-Bit Parallel Adders

4-Bits – Nibble



(a) Block diagram



(b) Logic symbol

Truth Table for a 4-Bit Parallel Adder

C_{n-1}	A_n	B_n	Σ_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

Example:

Use the 4-bit parallel adder truth table to find the sum and output carry for the following two 4-bit numbers if the input carry (C_{n-1}) is 0.

$A_4A_3A_2A_1 = 1100$, $B_4B_3B_2B_1 = 1100$

Solution:

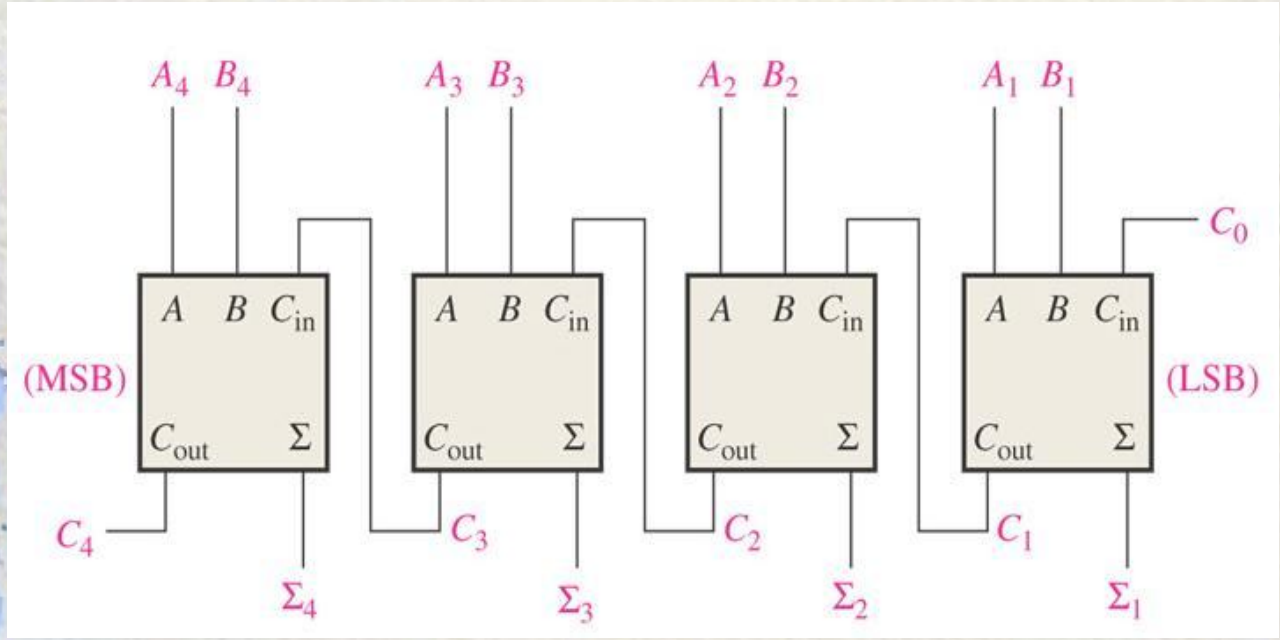
11000

n=1, $A_1 = 0$, $B_1 = 0$ and $C_{n-1} = 0$ → $\Sigma_1 = 0$ and $C_1 = 0$

n=2, $A_2 = 0$, $B_2 = 0$ and $C_{n-1} = 0$ → $\Sigma_2 = 0$ and $C_2 = 0$

n=3, $A_3 = 1$, $B_3 = 1$ and $C_{n-1} = 0$ → $\Sigma_3 = 0$ and $C_3 = 1$

n=4, $A_4 = 1$, $B_4 = 1$ and $C_{n-1} = 1$ → $\Sigma_4 = 1$ and $C_4 = 1$

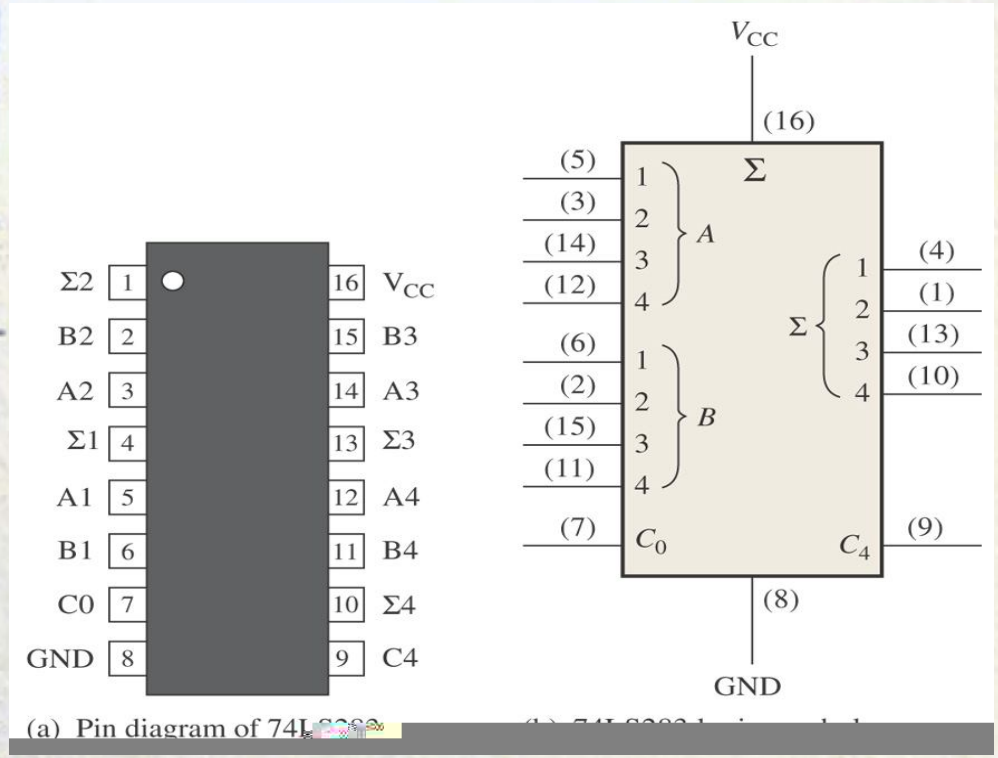


Try This!

1011 add with 1010 and Assume $C_{n-1} = 0$

10101

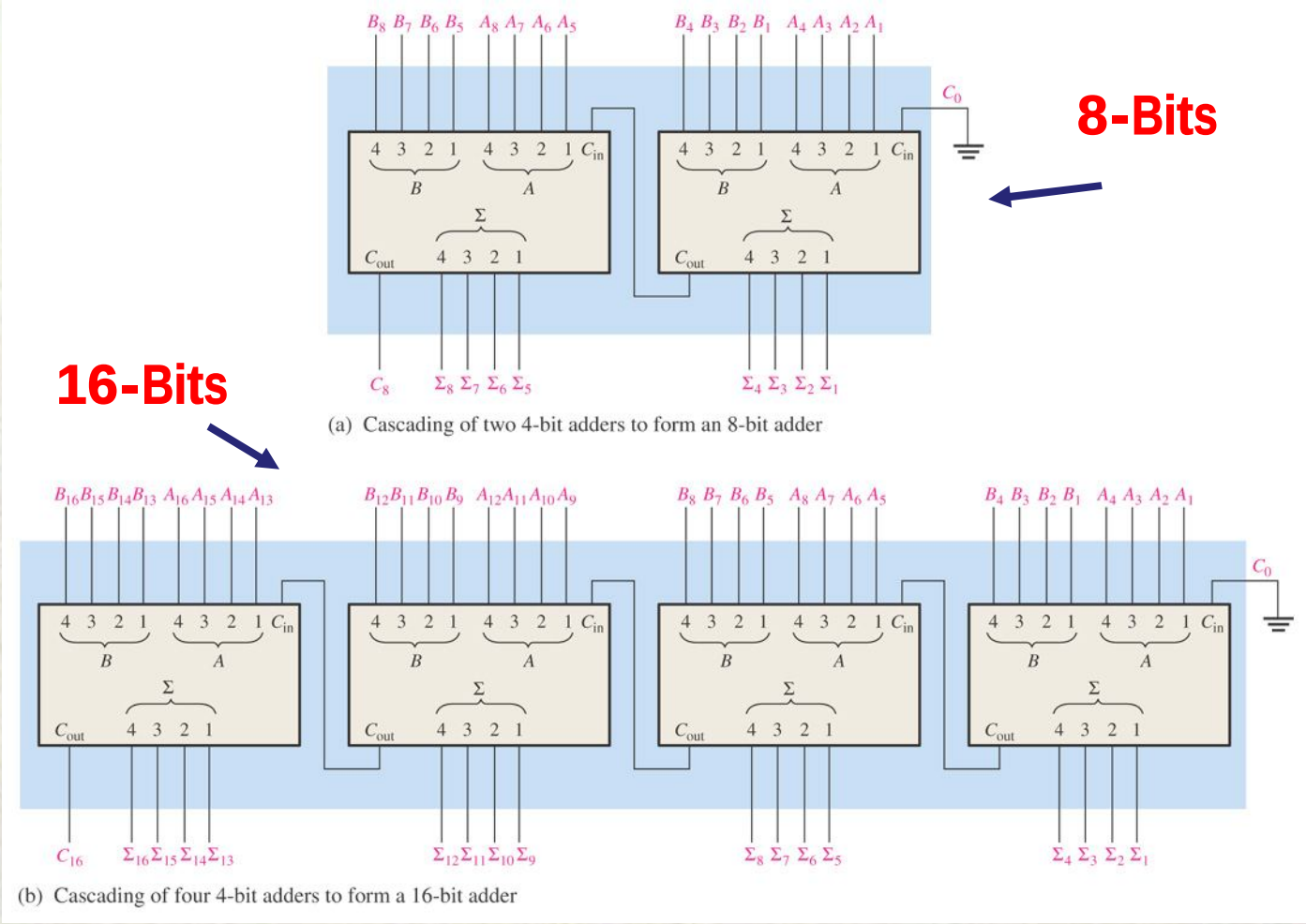
The 74LS283 4-Bit Parallel Adder



(a) Pin diagram of 74L

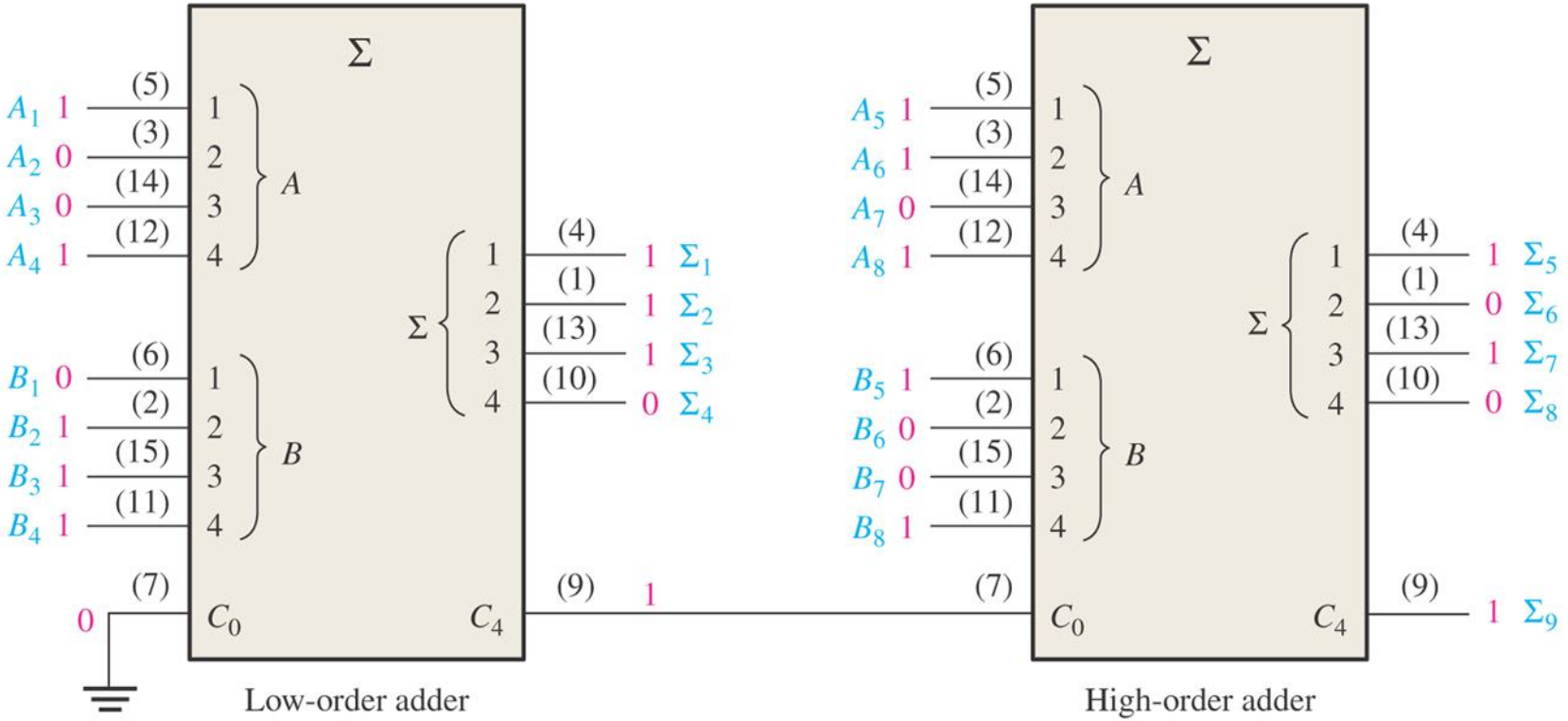
Symbol	Parameter	Limits			Unit
		Min	Typ	Max	
t_{PLH} t_{PHL}	Propagation delay, C_0 input to any Σ output		16 15	24 24	ns
t_{PLH} t_{PHL}	Propagation delay, any A or B input to Σ outputs		15 15	24 24	ns
t_{PLH} t_{PHL}	Propagation delay, C_0 input to C_4 output		11 11	17 22	ns
t_{PLH} t_{PHL}	Propagation delay, any A or B input to C_4 output		11 12	17 17	ns

Adder Expansion



Example:

Show how two 74LS283 adders can be connected to form an 8-bit parallel adder. Show output bits for the following 8-bit input numbers: $A_8A_7A_6A_5A_4A_3A_2A_1 = 10111001$ and $B_8B_7B_6B_5B_4B_3B_2B_1 = 10011110$



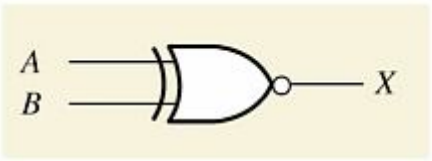
Answer: $\Sigma_8\Sigma_7\Sigma_6\Sigma_5\Sigma_4\Sigma_3\Sigma_2\Sigma_1 = 101010111$

Comparators

The basic function of a comparator is to compare the magnitude of two binary quantities to determine the relationship of those quantities

- 1-Bit Comparator
- 2-Bit Comparator
- 4-Bit Comparator

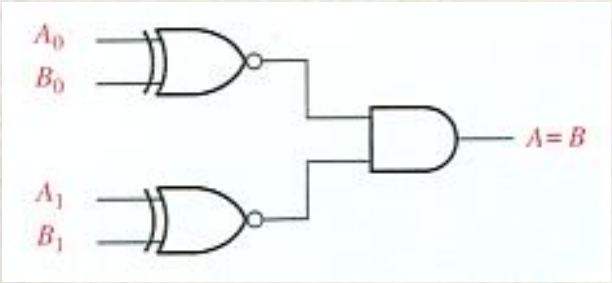
- 1-Bit Comparator



A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

The output is 1 when the inputs are equal

- 2-Bit Comparator

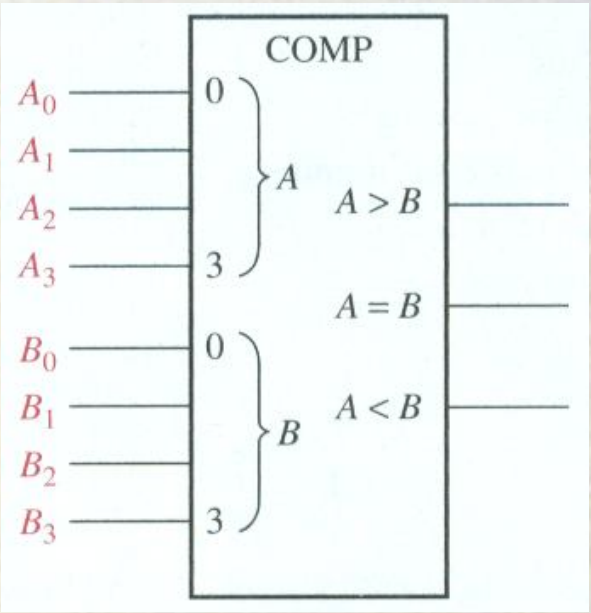


The output is 1 when $A_0 = B_0$ AND $A_1 = B_1$

- 4-Bit Comparator

  One of three outputs will be HIGH:

  - A greater than B (A > B)
  - A equal to B (A = B)
  - A less than B (A < B)

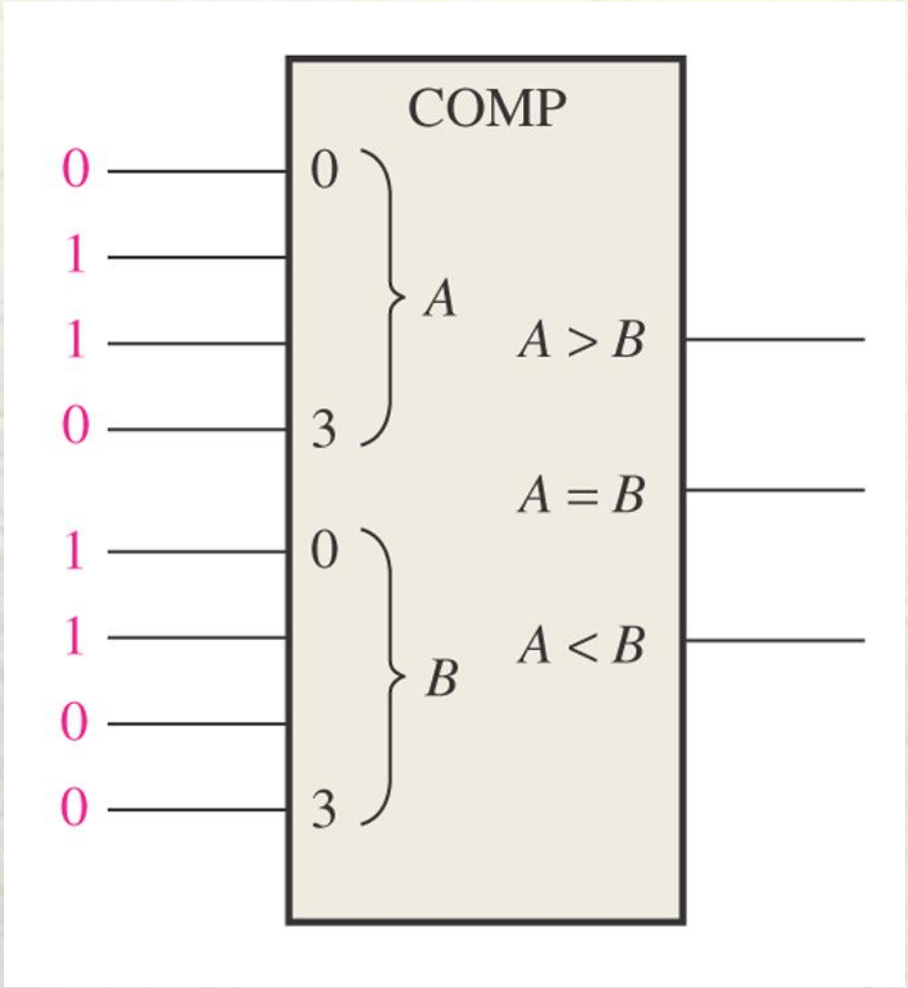


To determine an inequality of binary numbers A and B,first examine the highest order bit in each number. The following conditions are possible:

1. If $A_3 = 1$ and $B_3 = 0$, number A is greater than number B
2. If $A_3 = 0$ and $B_3 = 1$, number A is less than number B
3. If $A_3 = B_3$ then you must examine the next lower bit position for an inequality

Example:

Determine the A = B, A > B and A < B outputs for the input numbers shown on Figure below:



A > B is HIGH and the other outputs are LOW

Try This:

$A_3A_2A_1A_0$	$B_3B_2B_1B_0$
1 0 0 1	1 0 1 0

A < B is HIGH and the other outputs are LOW

Try This:

$A_3A_2A_1A_0$	$B_3B_2B_1B_0$
1 0 1 1	1 0 1 0

Decoders

- Binary decoder
- 4-bit decoder
- BCD-to-decimal decoder
- BCD-to-7-segement decoder

- Binary decoder

  The output is 1 only when:

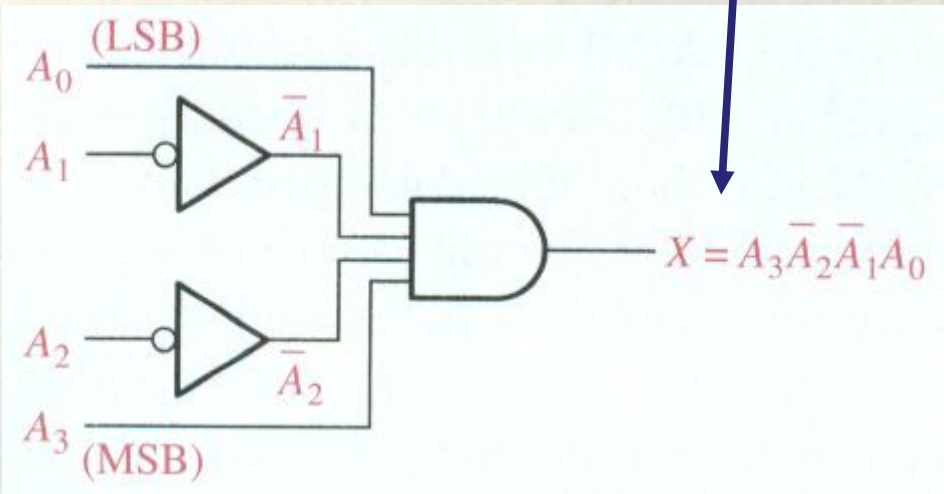
  $A_0 = 1$

  $A_2 = 0$

  $A_3 = 0$

  $A_4 = 1$

The output is only what we want!



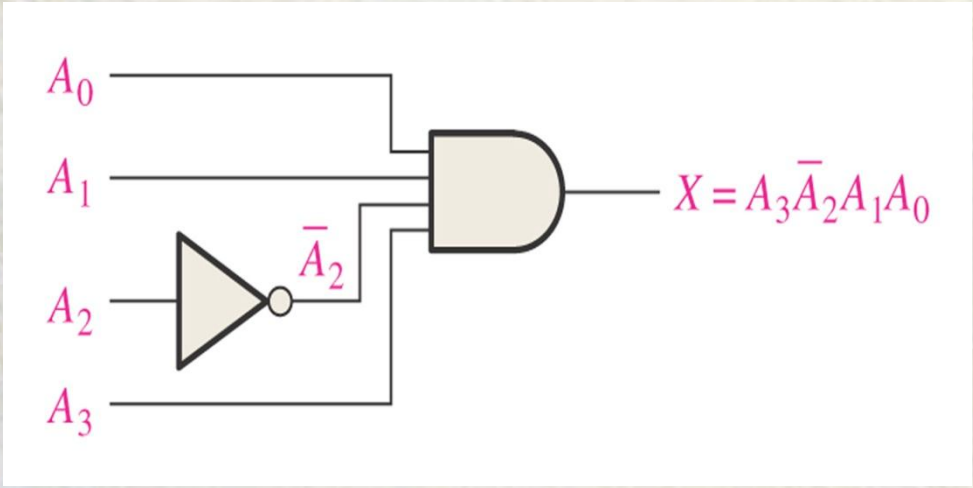
This is only one of an infinite number of examples

Example:

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

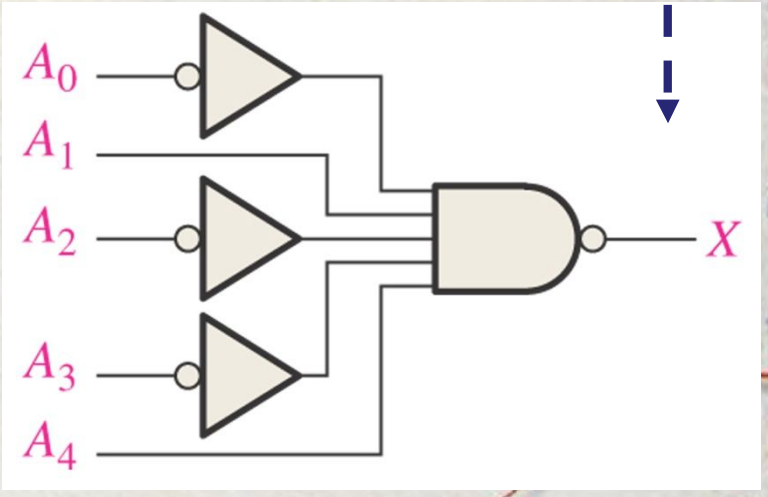
Solution:

$$X = A_3 \overline{A_2} A_1 A_0 = 1011$$



Try This:

Develop the logic required for 10010 and produce an active LOW output



The 4-Bit Decoder

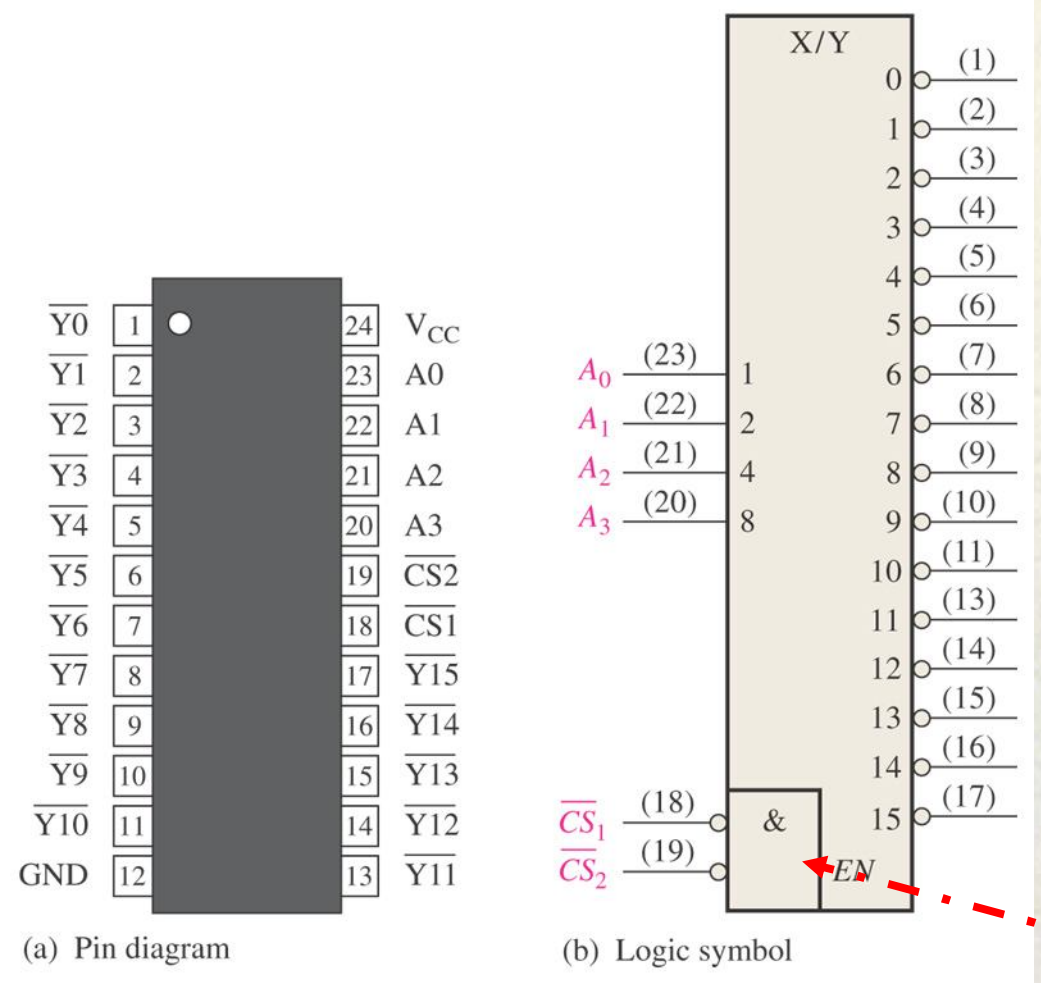
In order to decode all possible combinations of 4-bits, sixteen gates are required ($2^4 = 16$). This type of decoder is commonly called either *4-line-to-16-line* decoder or *1-of-16 decoder*.

BINARY INPUTS				DECODING	OUTPUTS															
A_3	A_2	A_1	A_0	FUNCTION	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
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
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
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
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
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
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
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
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
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
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
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
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
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
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
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
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

Truth Table

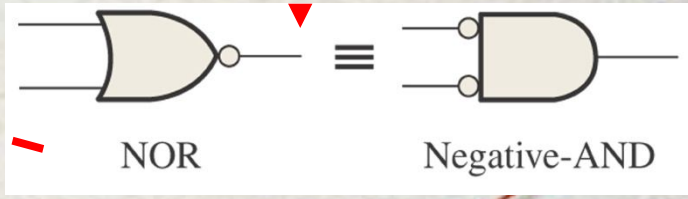
Output is Active Low

The 74HC154 1-of-16 Decoder



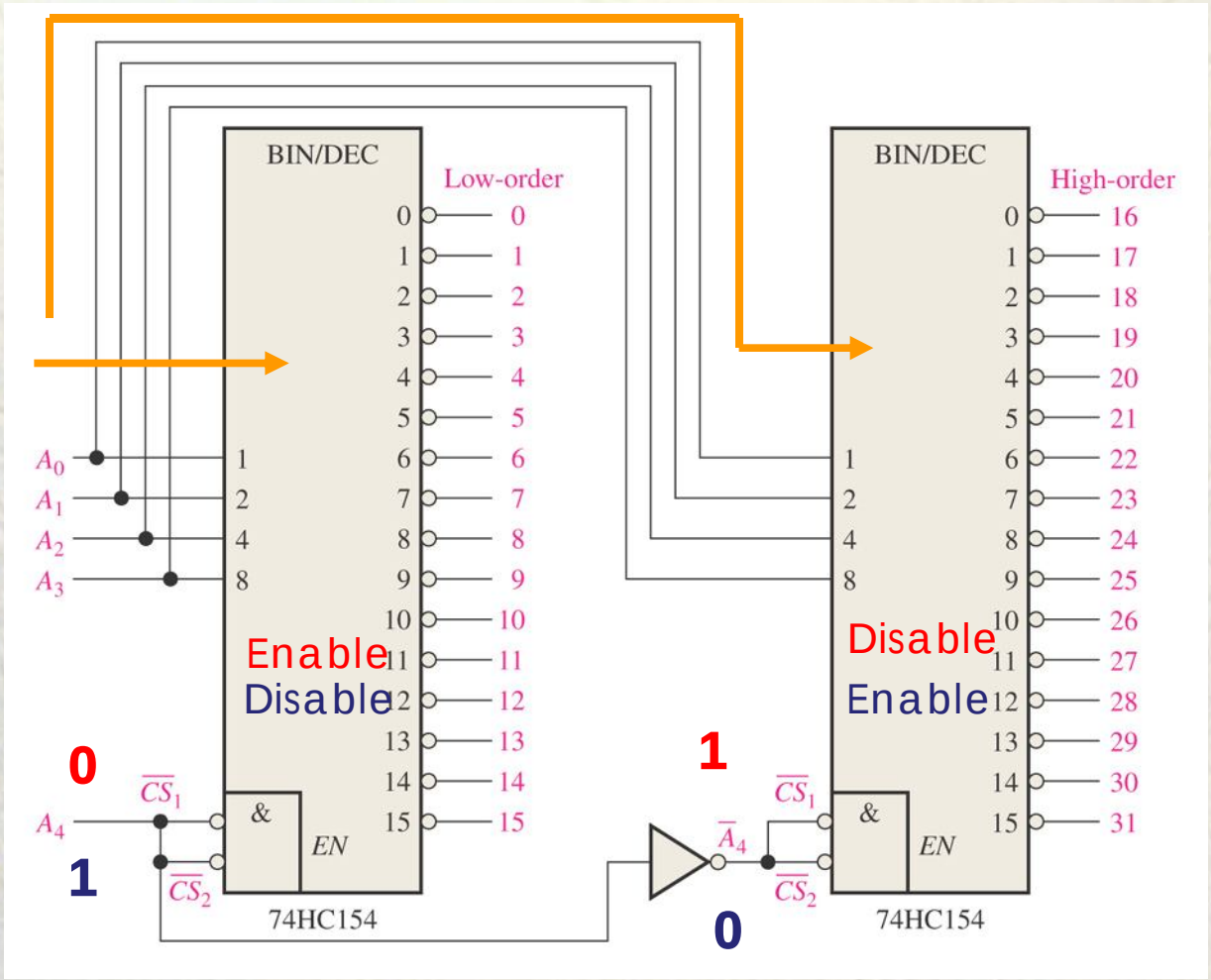
The IC will active if gate output (EN) is HIGH

If /CS_1 and /CS_2 are LOW, so EN will HIGH and IC is active!



Example:

A certain application requires that a 5-bit number be decoded. Use 74HC154 decoders to implement the logic. The binary number is represented by the format $A_4A_3A_2A_1A_0$.



Determine the output in Figure that is activated for the binary input **1 0 1 1 0 ?**

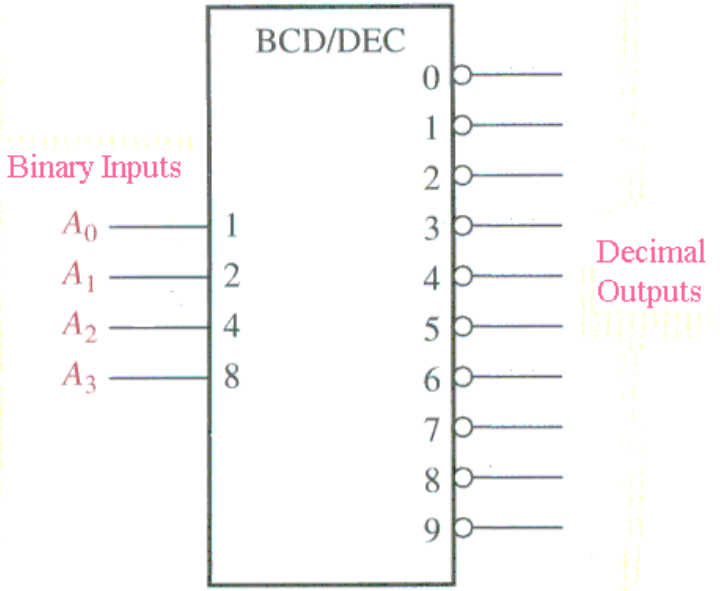
Answer:

22

The BCD-to-Decimal Decoder

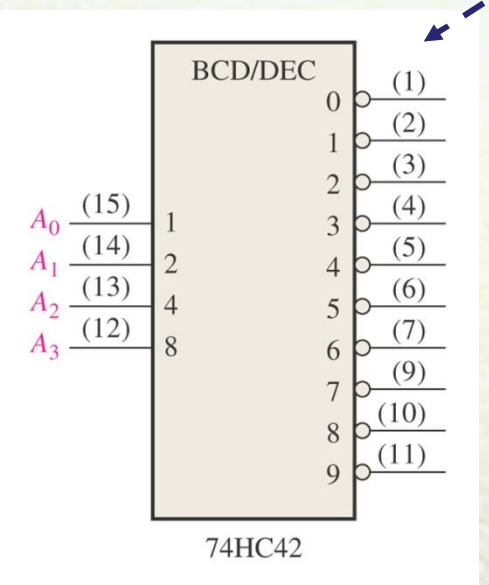
The BCD-to-decimal converts each BCD code into one of ten possible decimal digit indications. It is frequently referred as *4-line-to-10- line decoder* or a *1-of-10 decoder*. The method of implementation is the same as for the 1-of-16 decoder.

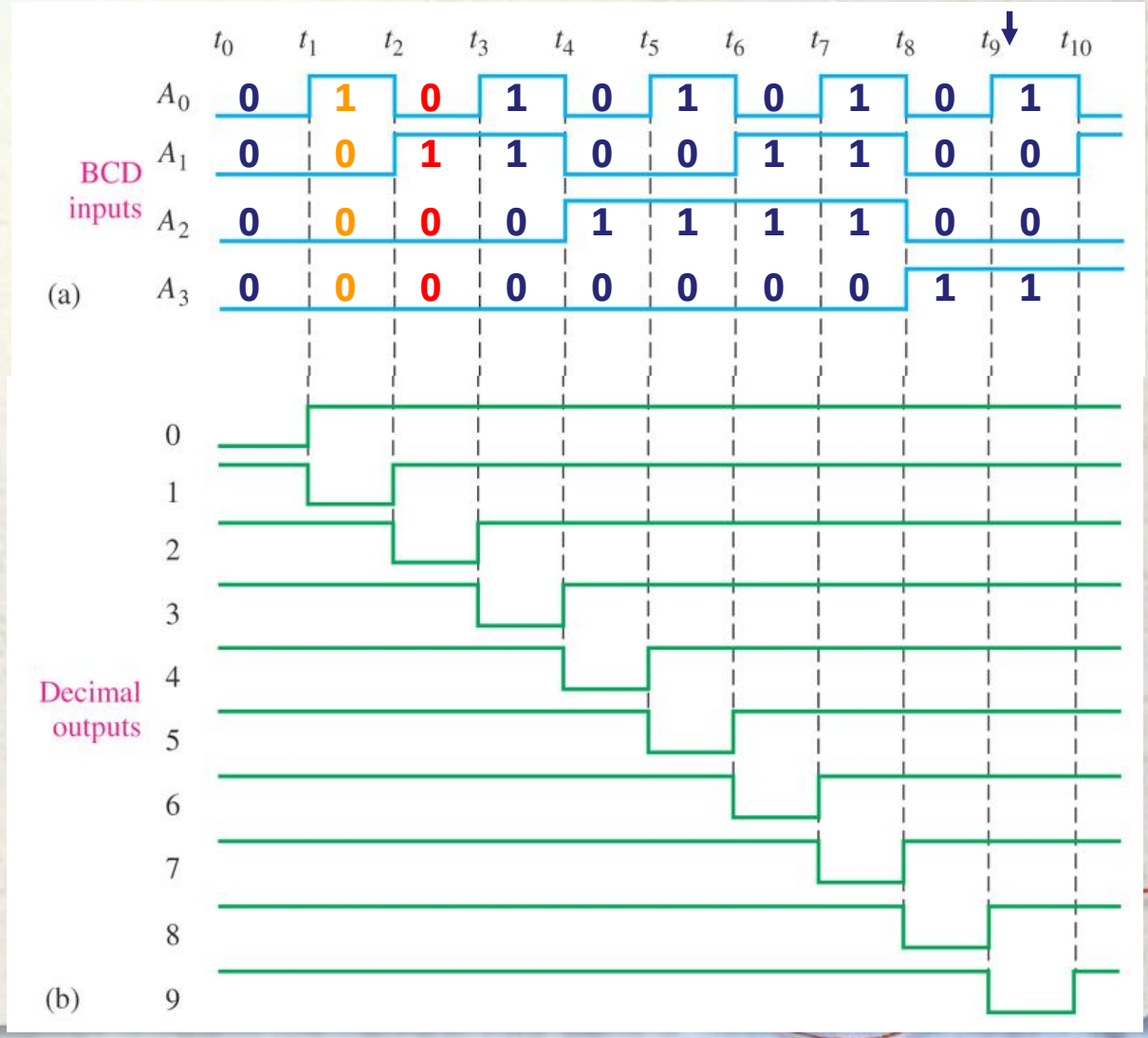
DECIMAL DIGIT	BCD CODE A_3	A_2	A_1	A_0	DECODING FUNCTION
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$
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	$A_3\overline{A}_2\overline{A}_1A_0$



Example:

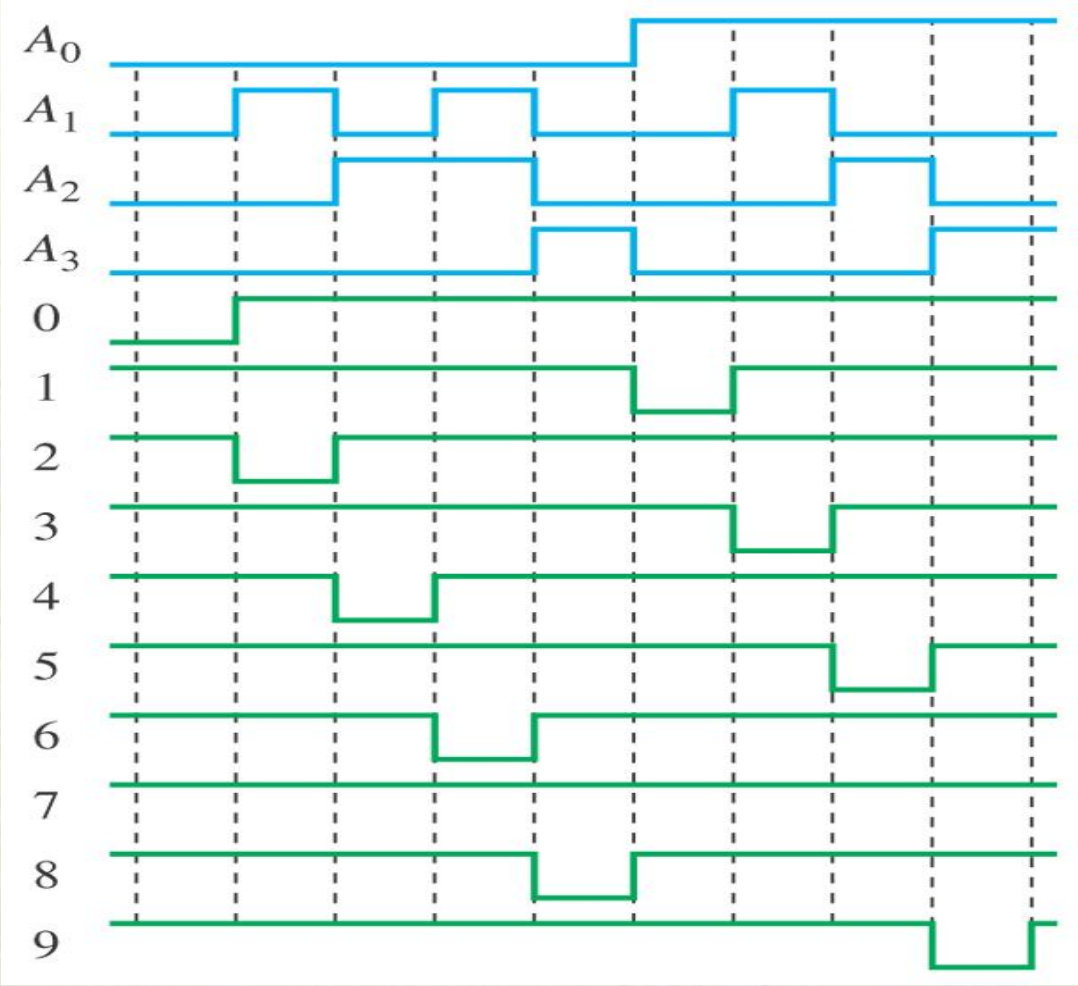
The 74HC42 is an integrated circuit BCD-to-decimal decoder. The logic symbol is shown in Figure 1 below. If the input waveforms in Figure 2 are applied to the inputs of the 74HC42, show the output waveforms.





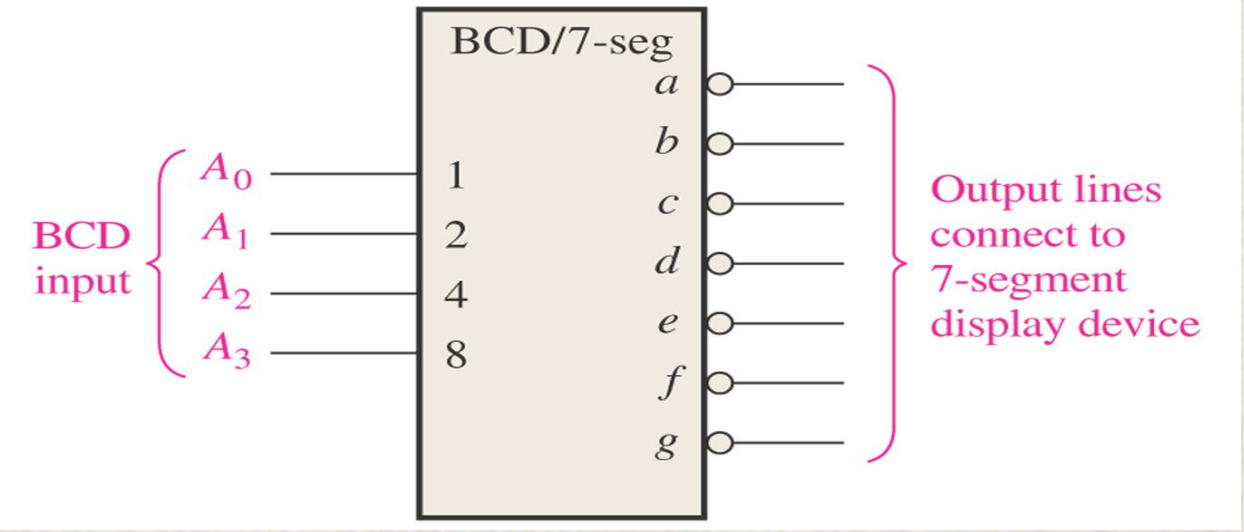
Exercise:

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



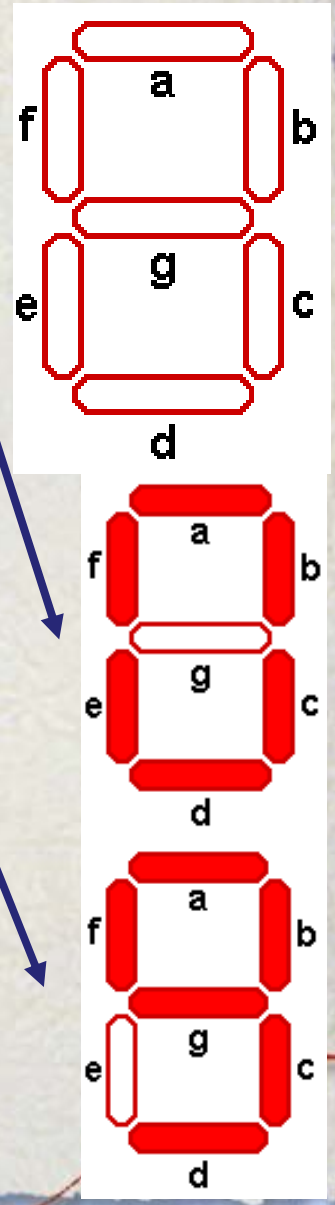
The BCD-to-7 Segment Decoder

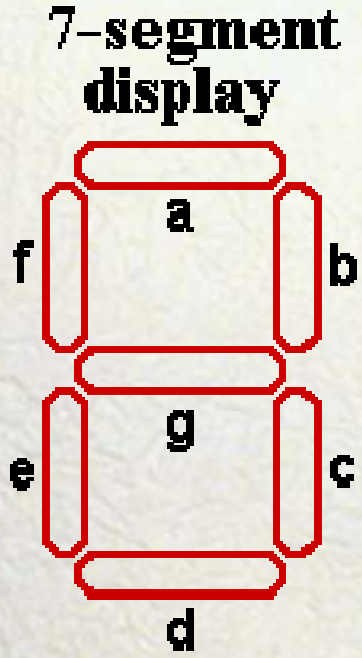
The BCD-to-7-segment decoder accepts the BCD code on its inputs and drive 7-segment display devices to produce a decimal readout.



Truth Table for BCD-to-7 Segment Decoder

DECIMAL DIGIT	INPUTS				SEGMENT OUTPUTS						
	D	*C*	*B*	*A*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
10	1	0	1	0	X	X	X	X	X	X	X
11	1	0	1	1	X	X	X	X	X	X	X
12	1	1	0	0	X	X	X	X	X	X	X
13	1	1	0	1	X	X	X	X	X	X	X
14	1	1	1	0	X	X	X	X	X	X	X
15	1	1	1	1	X	X	X	X	X	X	X




7-segment
display
a
b
c
d
e
f
g

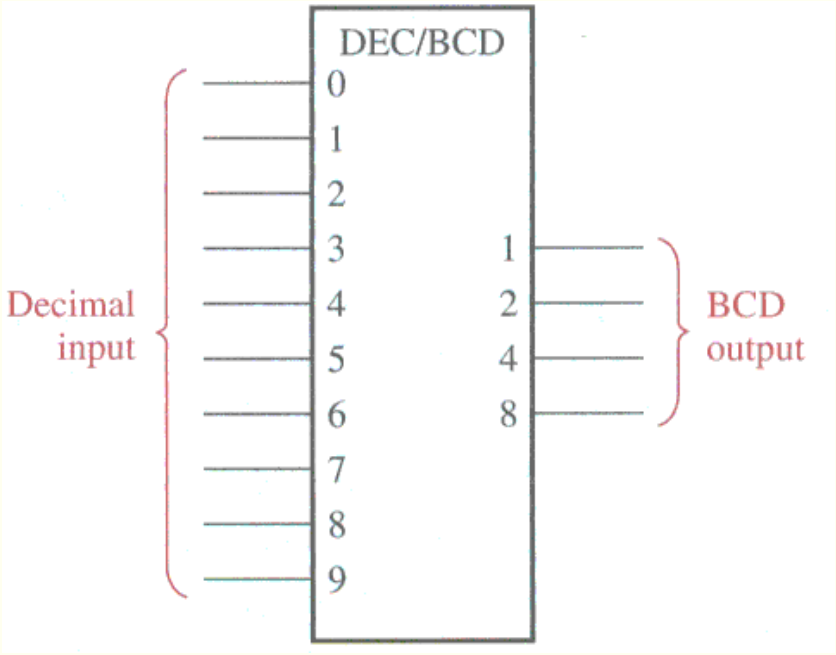
Encoders

- Decimal-to-BCD encoder
- 8-line-to-3-line encoder

An encoder is a combinational logic circuit that essentially performs a “reverse” decoder function.

The Decimal-to-BCD Encoder

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



The Decimal-to-BCD Encoder

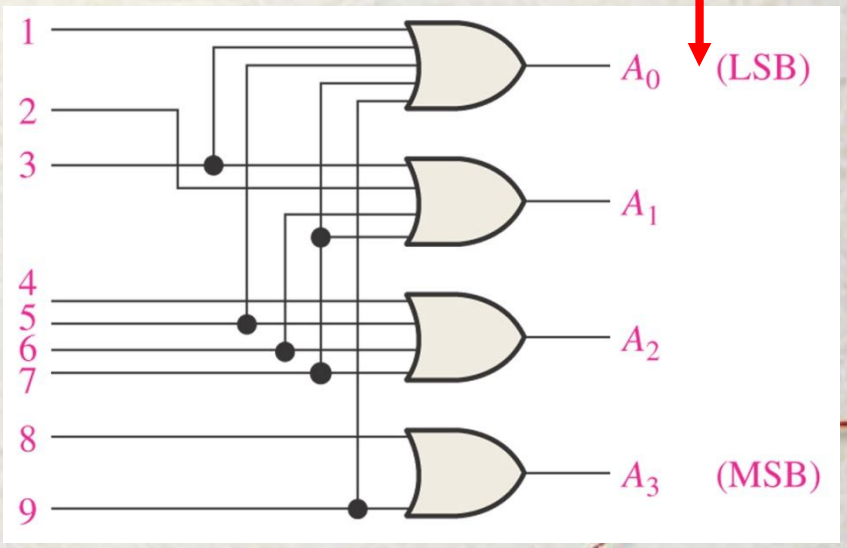
DECIMAL DIGIT	BCD CODE A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

$$A_3 = 8 + 9$$

$$A_2 = 4 + 5 + 6 + 7$$

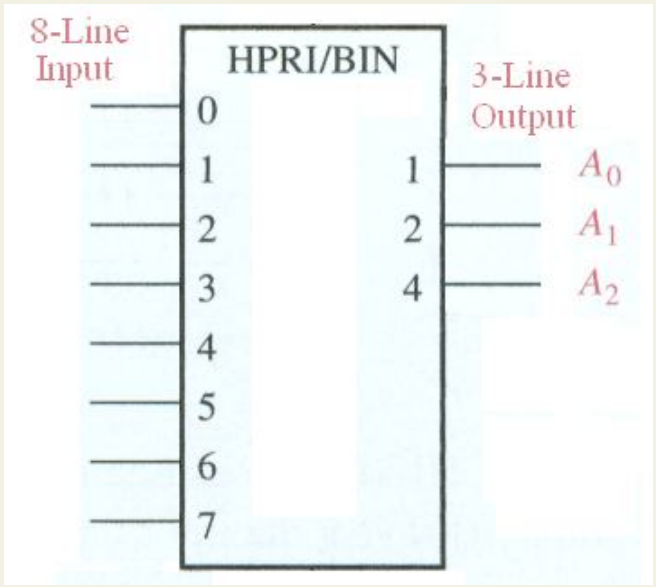
$$A_1 = 2 + 3 + 6 + 7$$

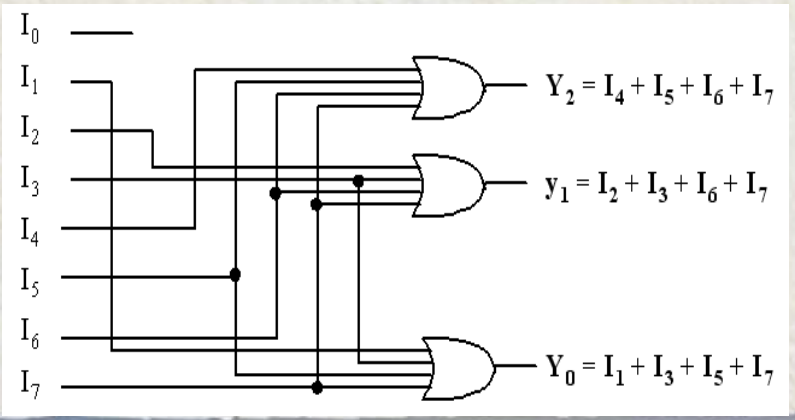
$$A_0 = 1 + 3 + 5 + 7 + 9$$



8-Line-to-3-Line Encoder

8-LINE INPUT	3-LINE OUTPUT		
	A_2	A_1	A_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

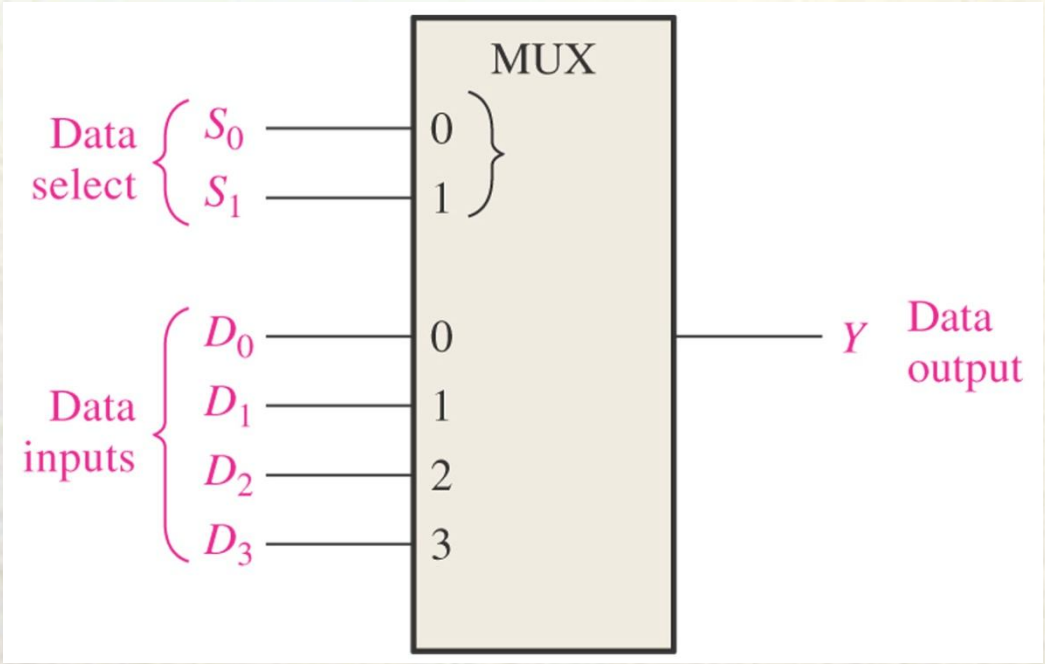


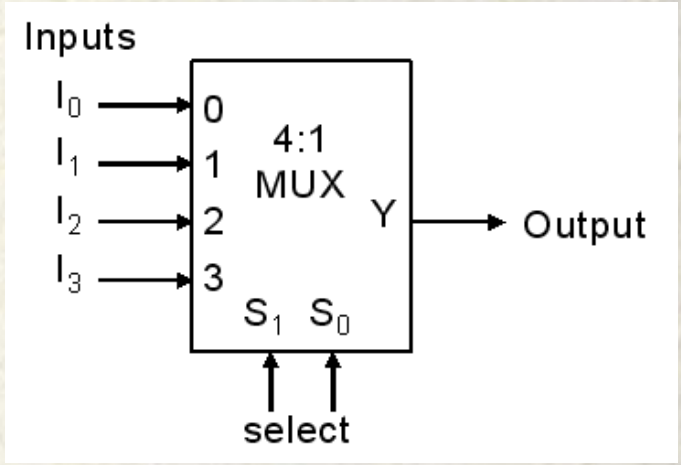


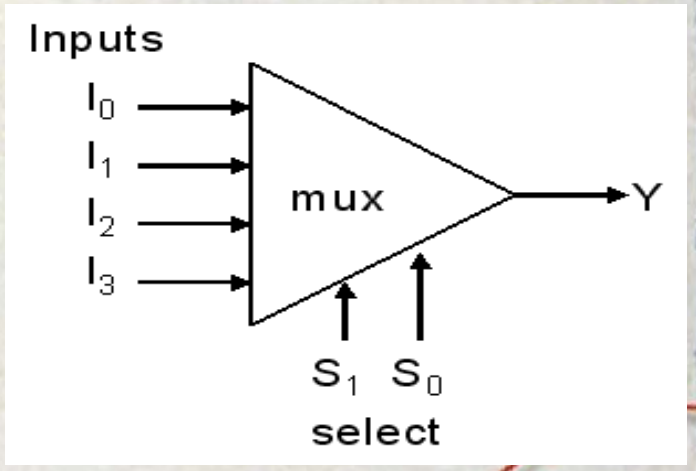
Multiplexer (Data Selectors)

- A multiplexer (MUX) is a device that allows digital information from several sources to be routed onto a single line for transmission over that line to a common destination.
- The basic multiplexer 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.

1-of-4 data MUX





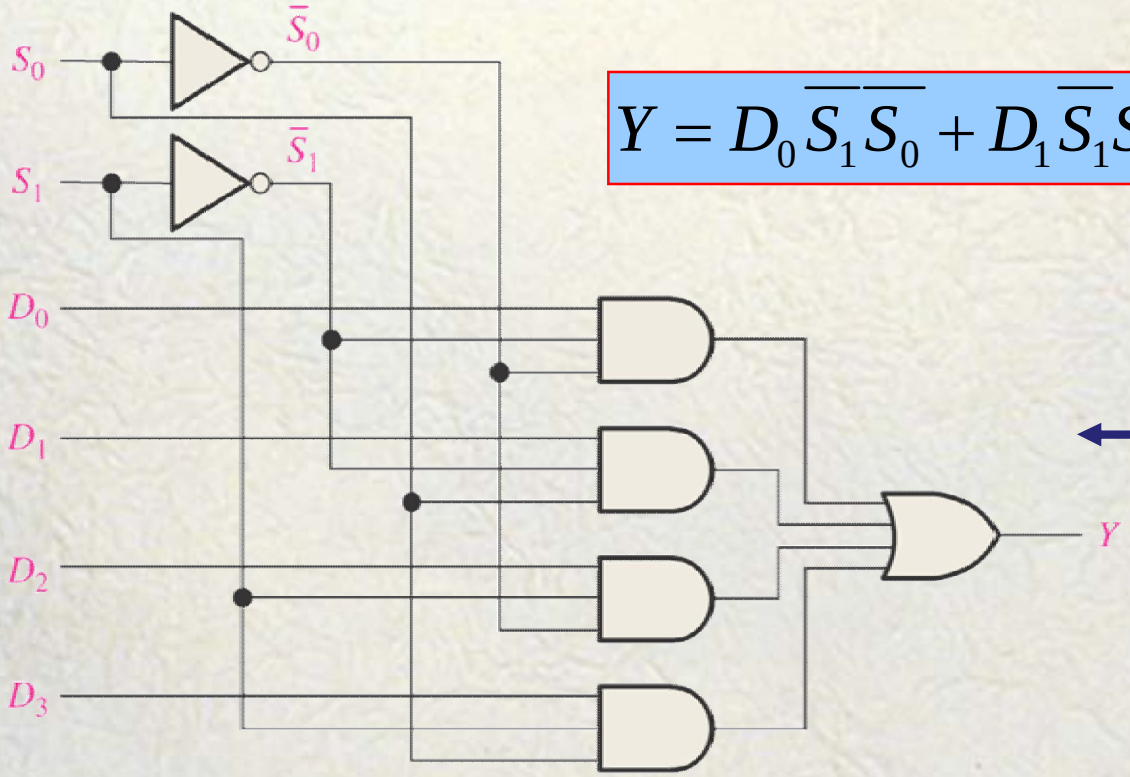


DATA-SELECT INPUTS S_1	S_0	INPUT SELECTED
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

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

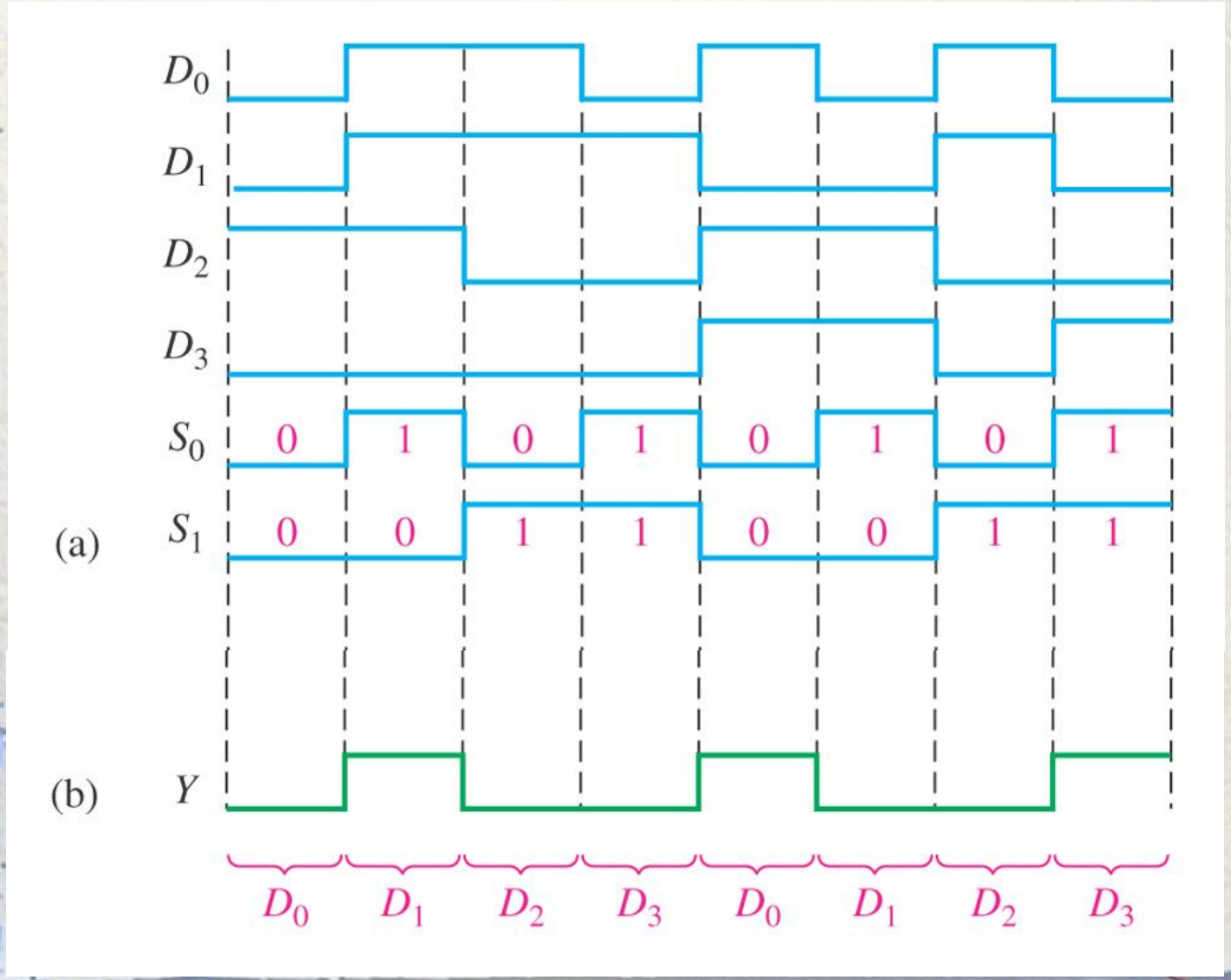
$$Y = D_0 \overline{S_1}\,\overline{S_0}$$
$$Y = D_1 \overline{S_1} S_0$$
$$Y = D_2 S_1 \overline{S_0}$$
$$Y = D_3 S_1 S_0$$

$$Y = D_0 \overline{S_1}\,\overline{S_0} + D_1 \overline{S_1} S_0 + D_2 S_1 \overline{S_0} + D_3 S_1 S_0$$

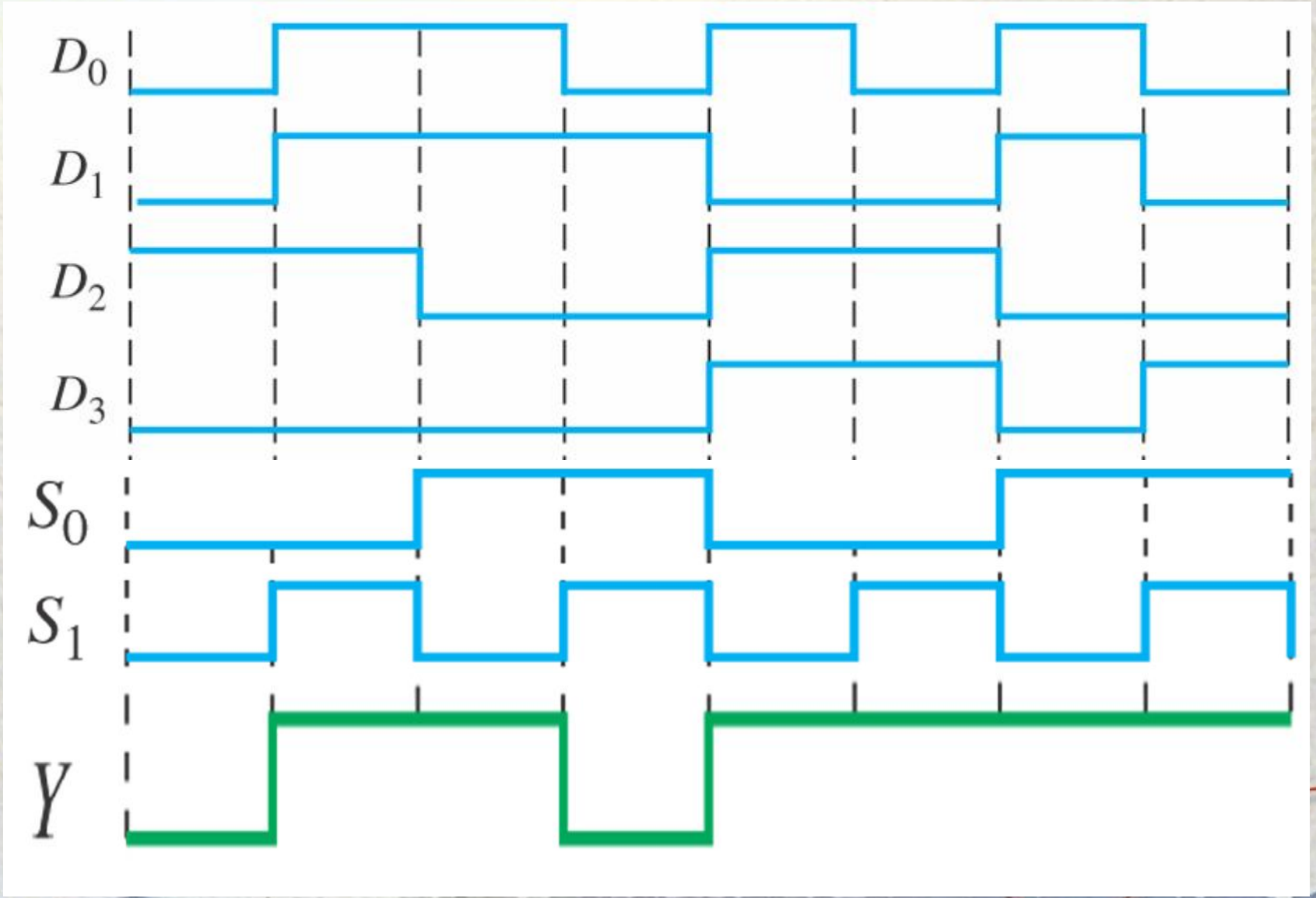


Example:

Determine the output waveform in relation to the inputs

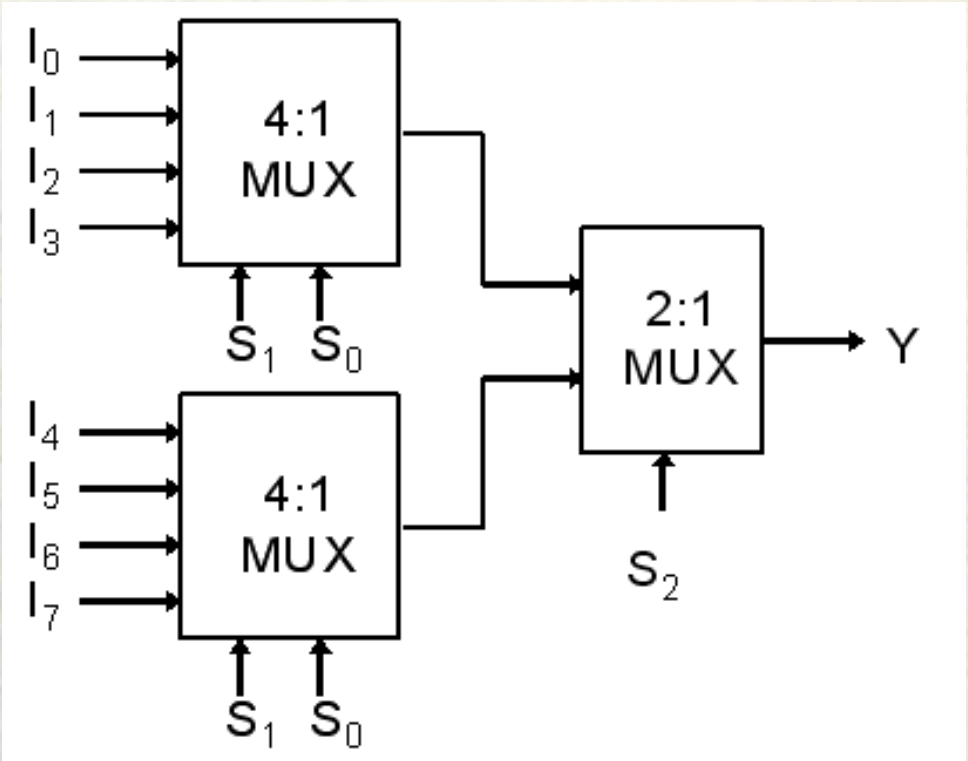


Exercise: Determine the output waveform in relation to the inputs



Multiplexer

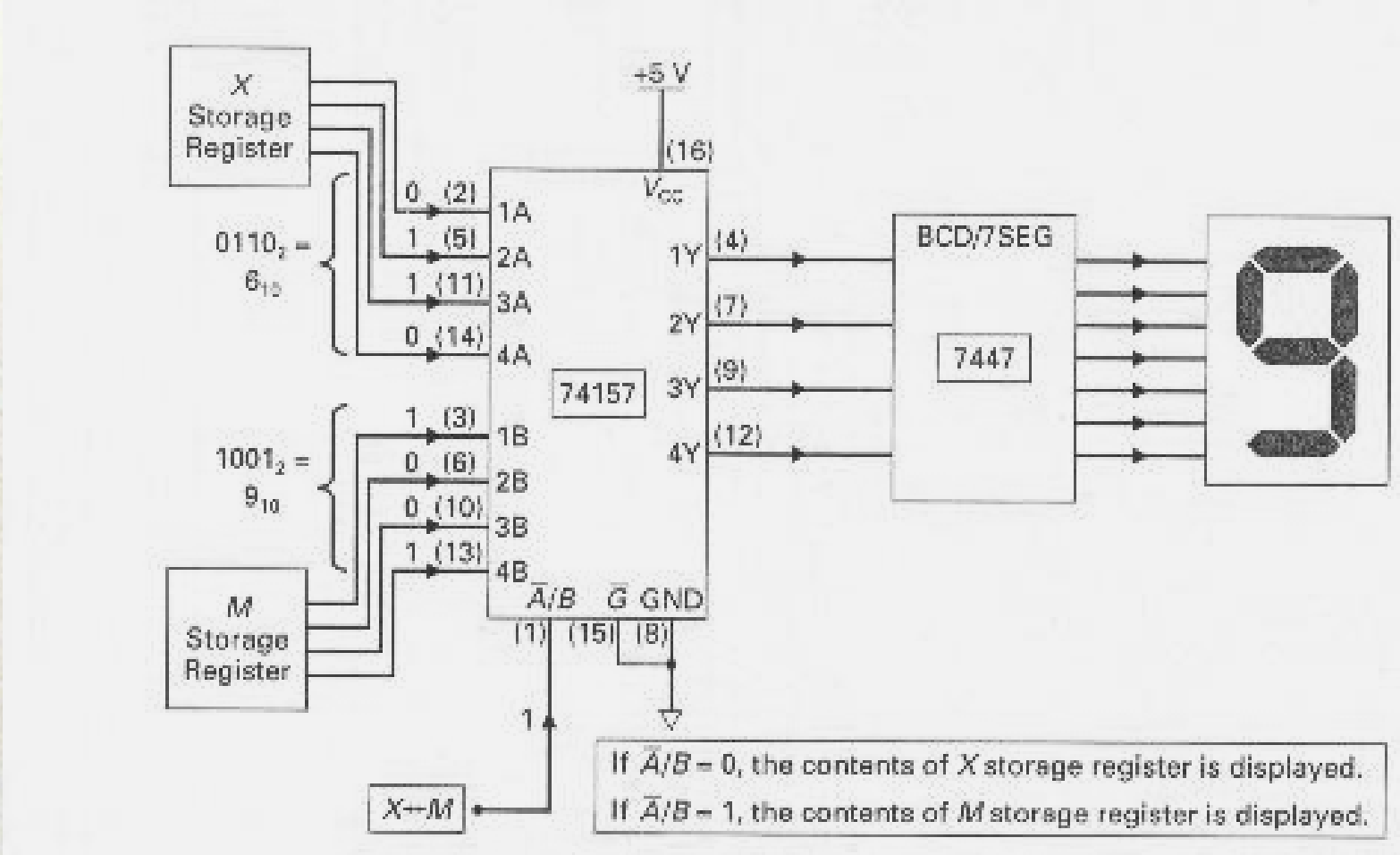
- Larger multiplexers can be constructed from smaller ones.
- An 8-to-1 multiplexer can be constructed from smaller multiplexers as shown:



S_2	S_1	S_0	Y
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

Multiplexer

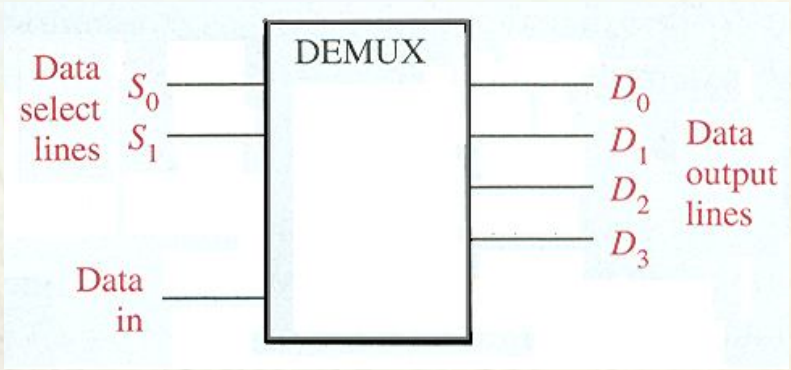
- Application Example:



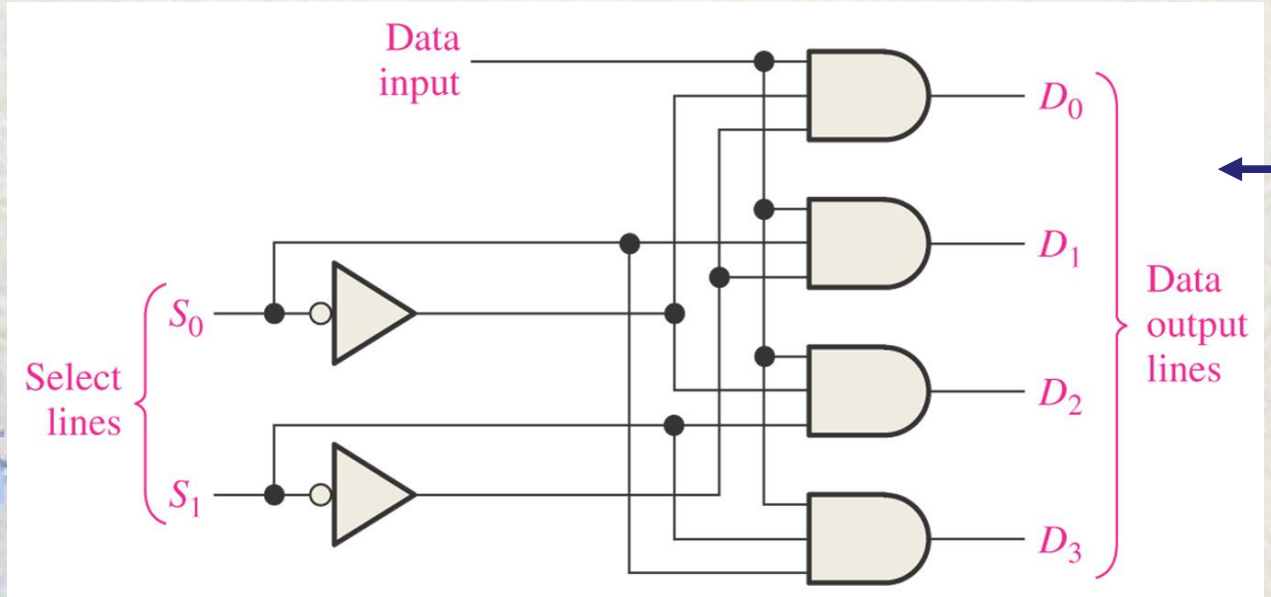
Demultiplexers

- A demultiplexer (DEMUX) basically reverses the multiplexing function.
- It takes digital information from one line and distributes it to a given number of output lines.

1-to-4 DEMUX



DATA-SELECT INPUTS S_1	S_0	OUTPUT SELECTED
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



$$D_0 = I \bullet \overline{S_1}\,\overline{S_0}$$

$$D_1 = I \bullet \overline{S_1} S_0$$

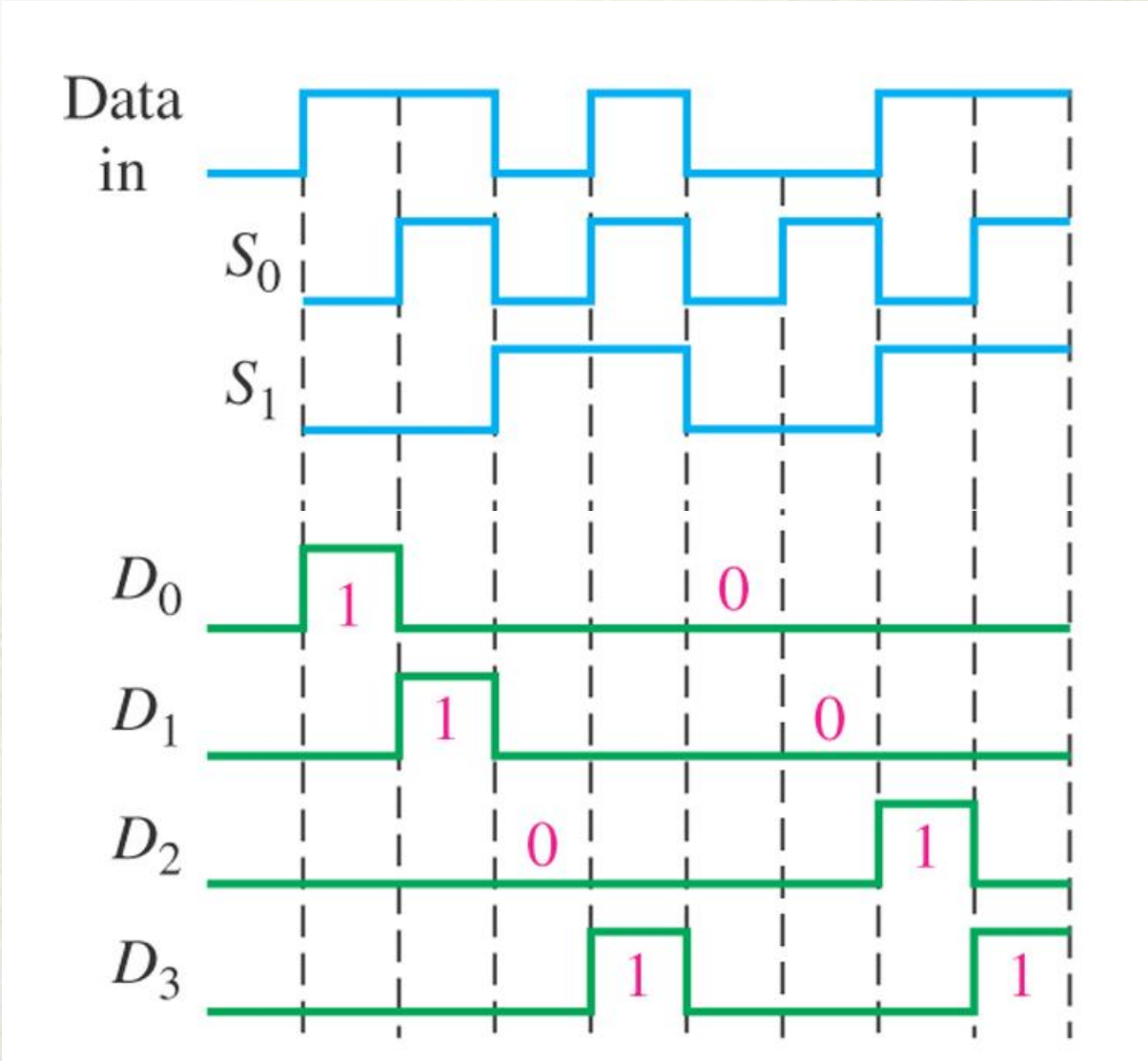
$$D_2 = I \bullet S_1 \overline{S_0}$$

$$D_3 = I \bullet S_1 S_0$$

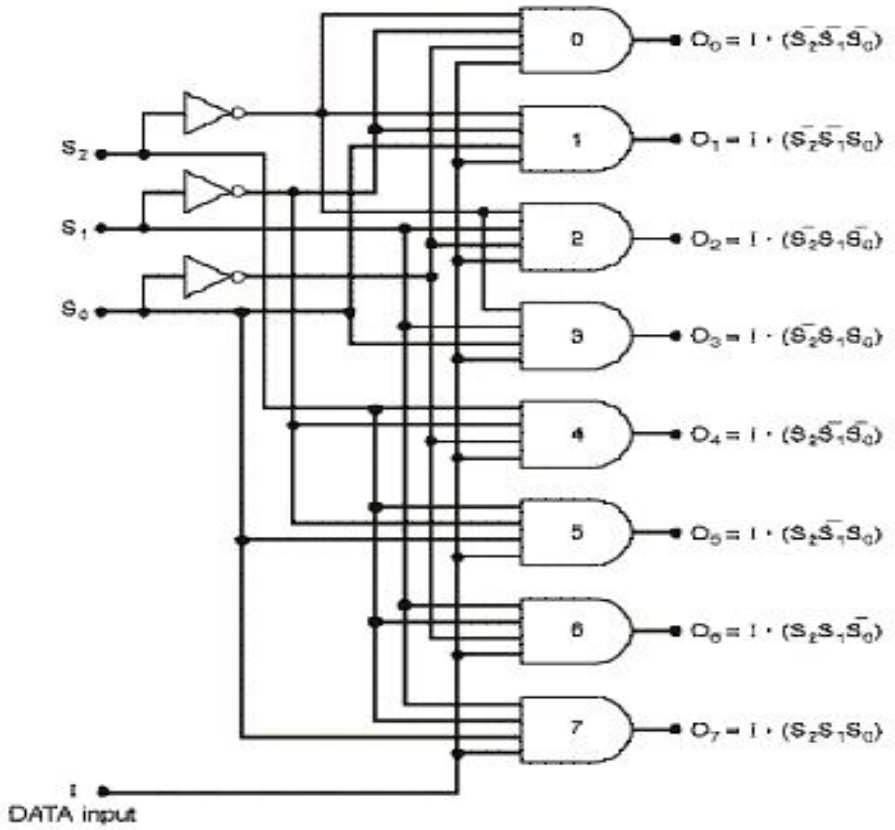
Example: Data Input = 1 and S_1 and S_0 = 1 **D_3**

Exercise:

Determine the data-output waveforms



1-to-8 DEMUX

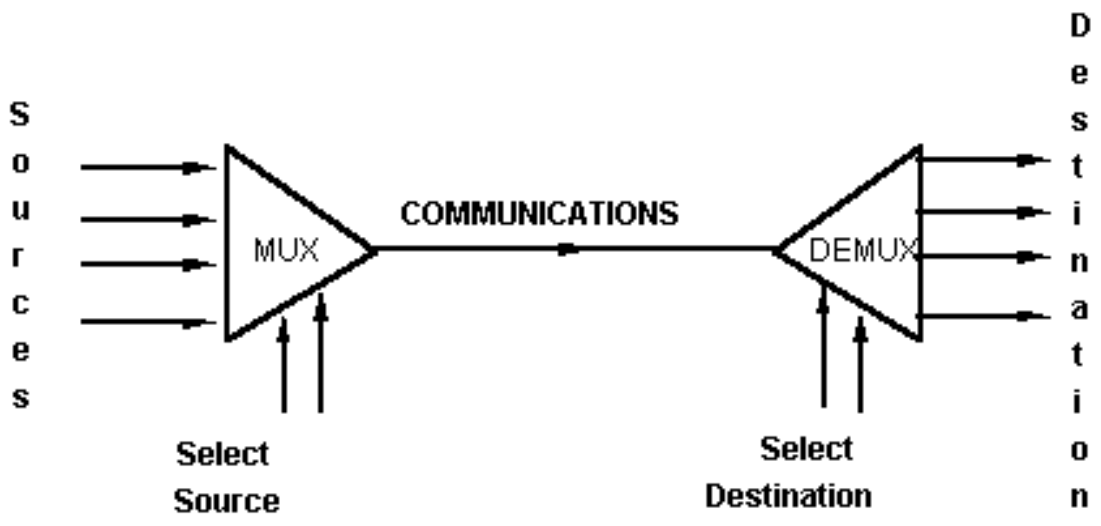


1-line-to-8-line multiplexer

SELECT code			OUTPUTS							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

Note: I is the data input

Mux-Demux Application Example



This enables sharing a single communication line among a number of devices.

At any time, only one source and one destination can use the communication line.

Thank You

“Hati seorang yang bodoh terletak di mulutnya, tetapi mulut seorang yang bijak terletak di hatinya”