OTHER COMBINATIONAL LOGIC CIRCUITS

WEEK 7 AND WEEK 8 (LECTURE 2 OF 3)

- DECODERS
- ENCODERS

DECODER

- A decoder is a logic circuit that accepts a set of inputs that represents a binary number and activates only the output that corresponds to the input number.

- In other words, a decoder circuit looks at its inputs, determines which binary number is present there, and activates the one output that corresponds to that number ; all other outputs remain inactive

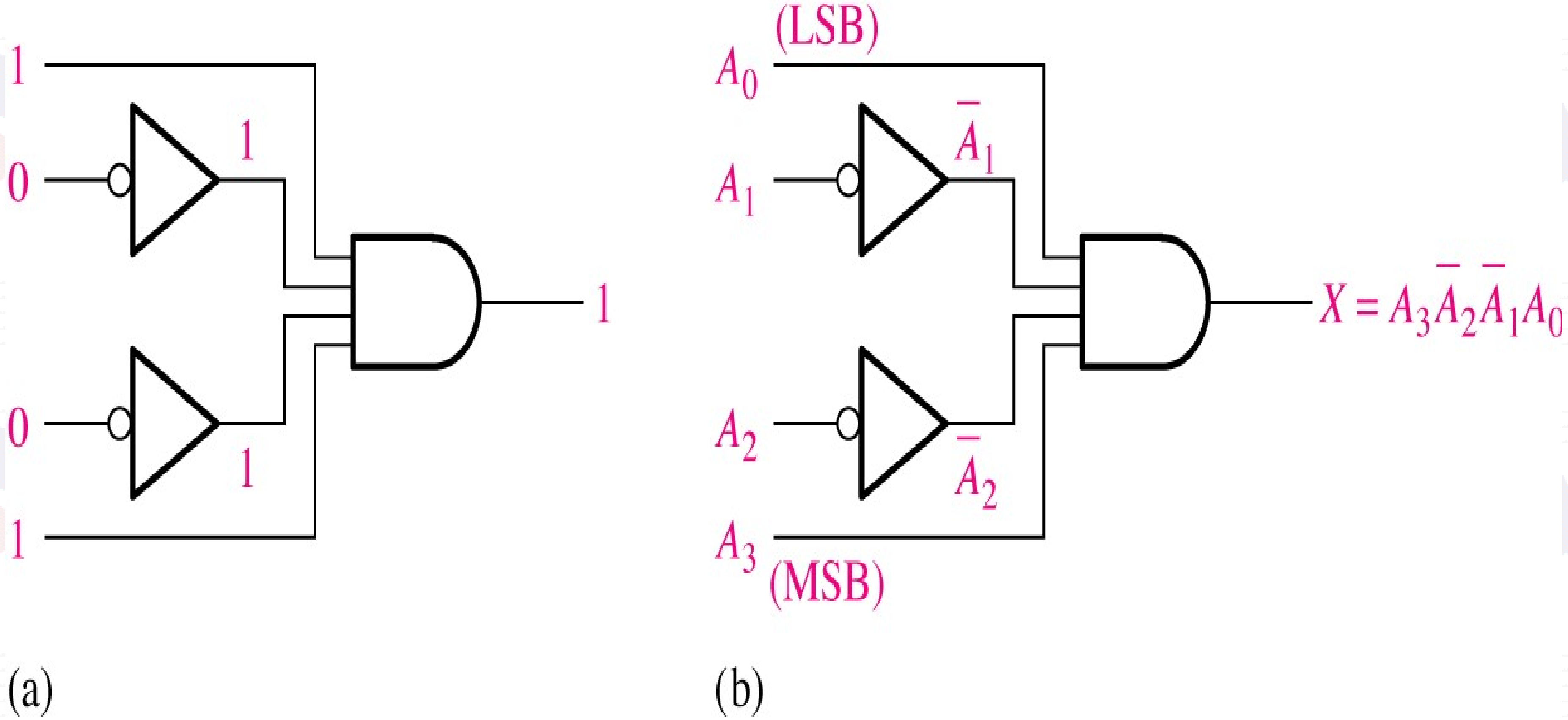
In its general form, a decoder has N input lines to handle N bits and form one to 2^N output lines to indicate the presence o one or more N-bit combinations.

The basic binary function

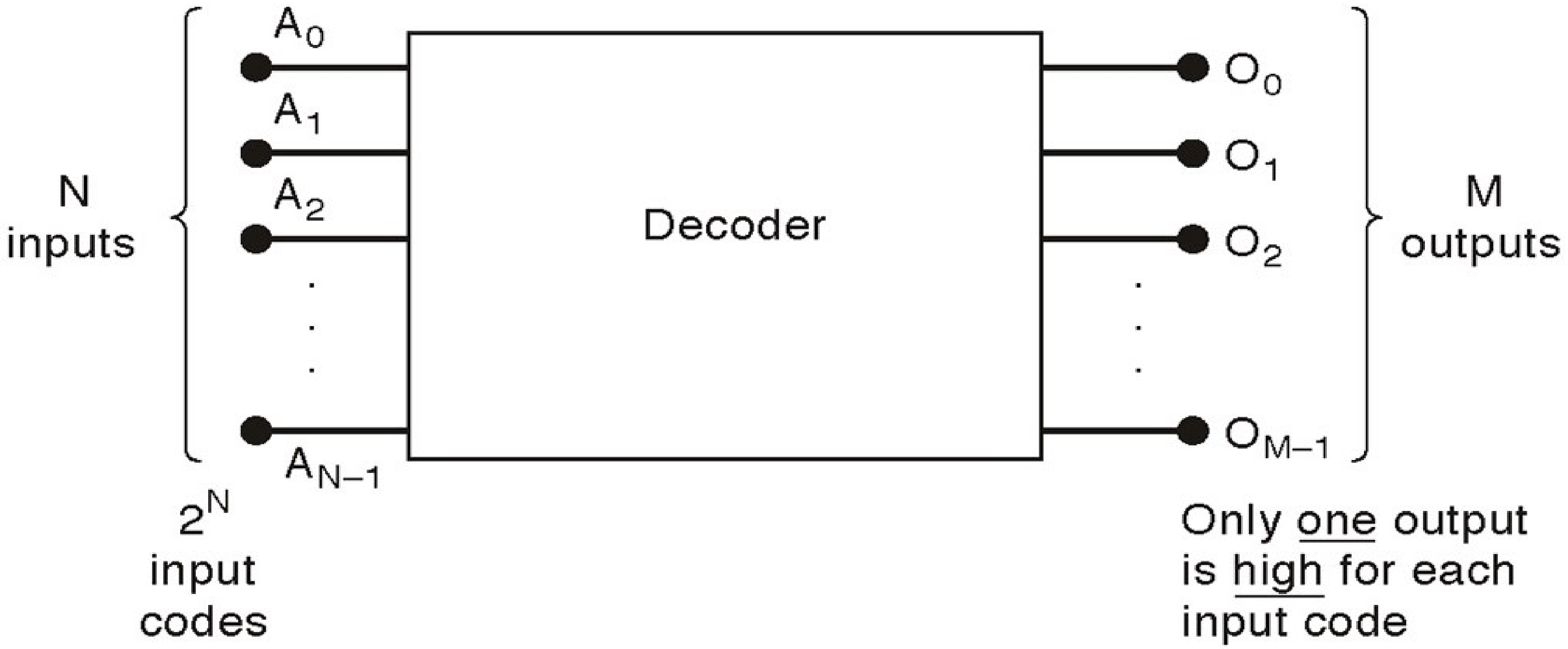
- An AND gate can be used as the basic decoding element because it produces a HIGH output only when all inputs are HIGH

Refer next slide for example

Decoding logic for the binary code 1001 with an active-HIGH output.



General decoder diagram



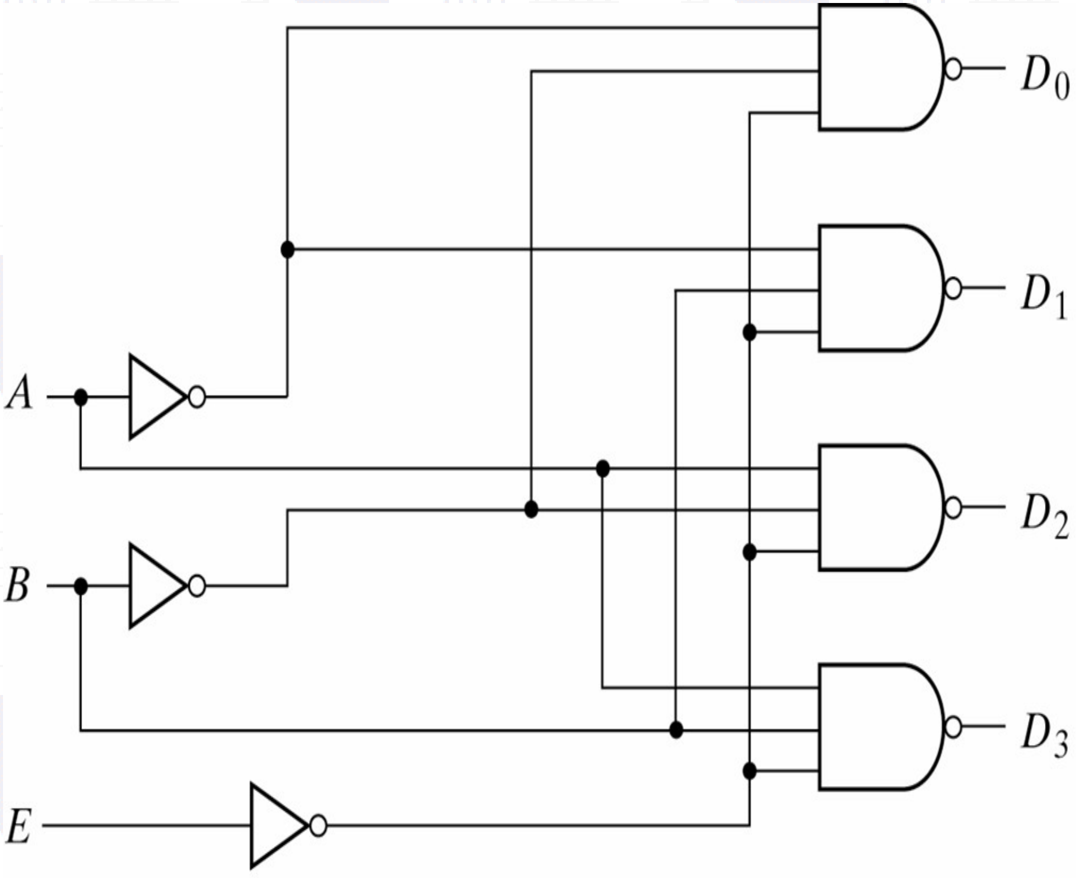
# There are 2^N possible input combinations, from A_0 to A_{N-1}.

For each of these input combinations only one of the *M* outputs will be active *HIGH* (1), all the other outputs are *LOW* (0).

- If an active-LOW output (74138, one of the output will low and the rest will be high) is required for each decoded number, the entire decoder can be implemented with
  1. NAND gates
  2. Inverters

- If an active-HIGH output (74139, one of the output will high and the rest will be low) is required for each decoded number, the entire decoder can be implemented with
  - AND gates
  - Inverters

2-to-4-Line Decoder
(with Enable input)-Active LOW output (1)...



(a) Logic diagram

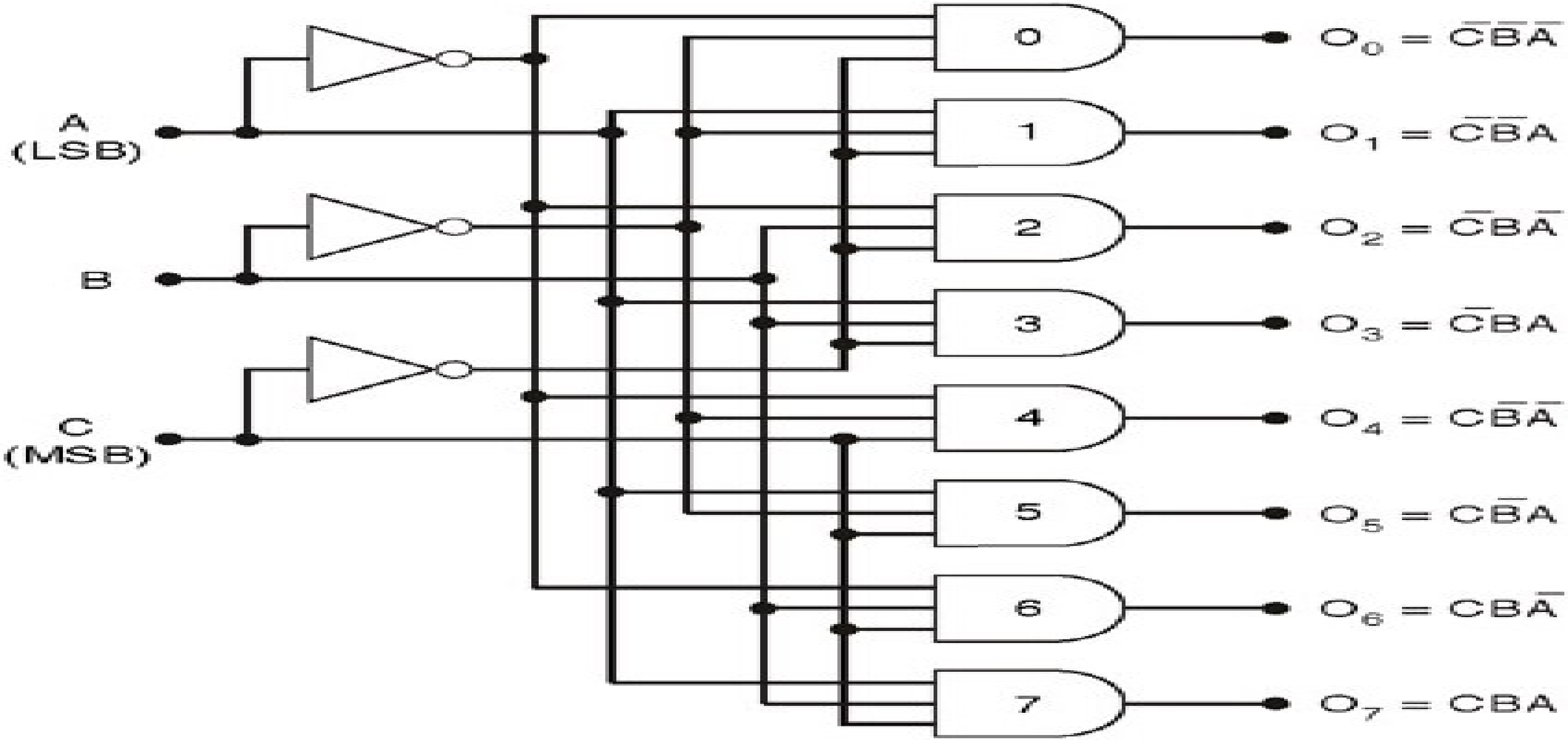
E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

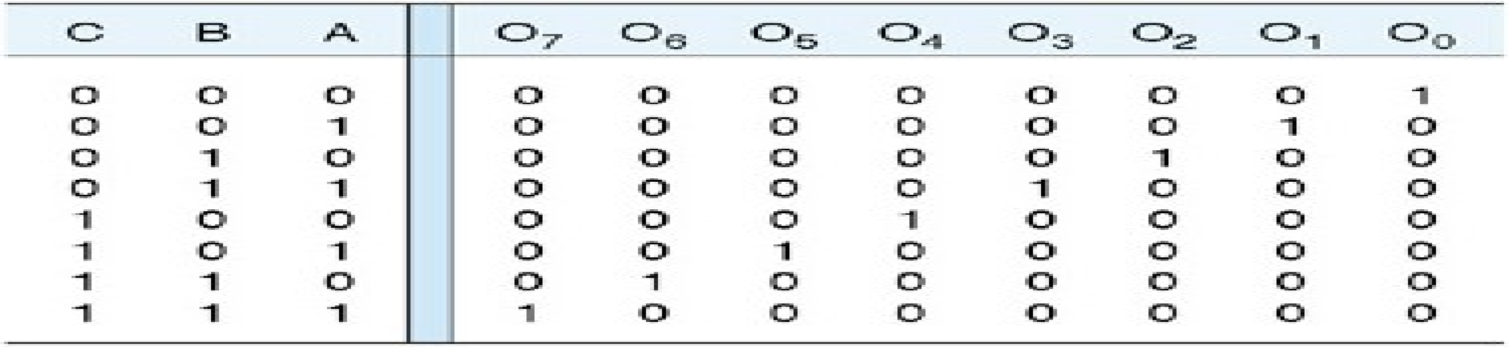
(b) Truth table

2-to-4-Line Decoder (with Enable input)-Active LOW output (2)

- The circuit operates with complemented outputs and a complement enable input. The decoder is enabled when E is equal to 0.
- Only one output can be equal to 0 at any given time, all other outputs are equal to 1.
- The output whose value is equal to 0 represents the minterm selected by inputs A and B
- The circuit is disabled when E is equal to 1.

3-8 line decoder (active-HIGH)



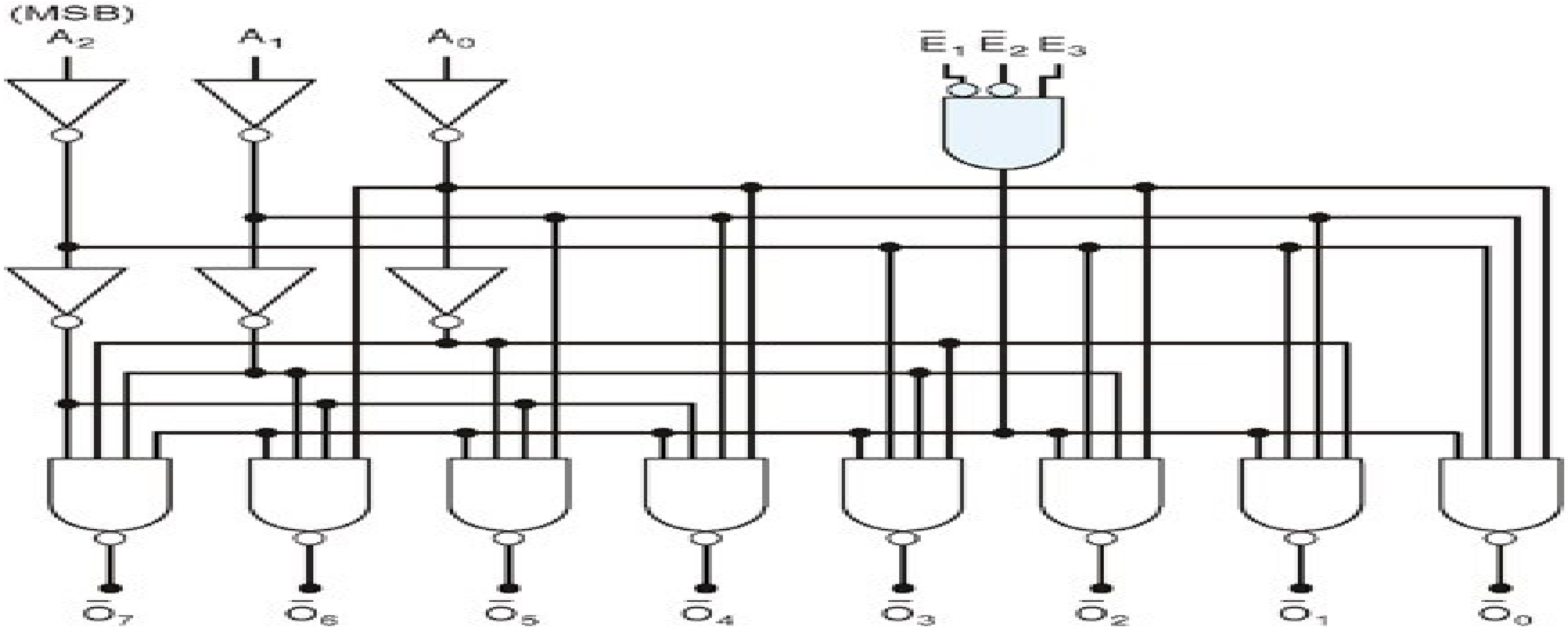


C	B	A	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

- This decoder can be referred to in several ways. It can be called a 3-line-to- 8-line decoder, because it has three input lines and eight output lines.

- It could also be called a binary-octal decoder or converters because it takes a three bit binary input code and activates the one of the eight outputs corresponding to that code. It is also referred to as a 1-of-8 decoder, because only 1 of the 8 outputs is activated at one time.

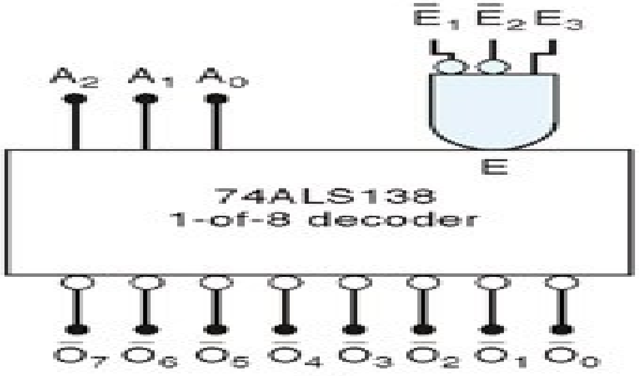
Logic diagram of 74138 (Example of a 3–Bit Decoder)



(a)

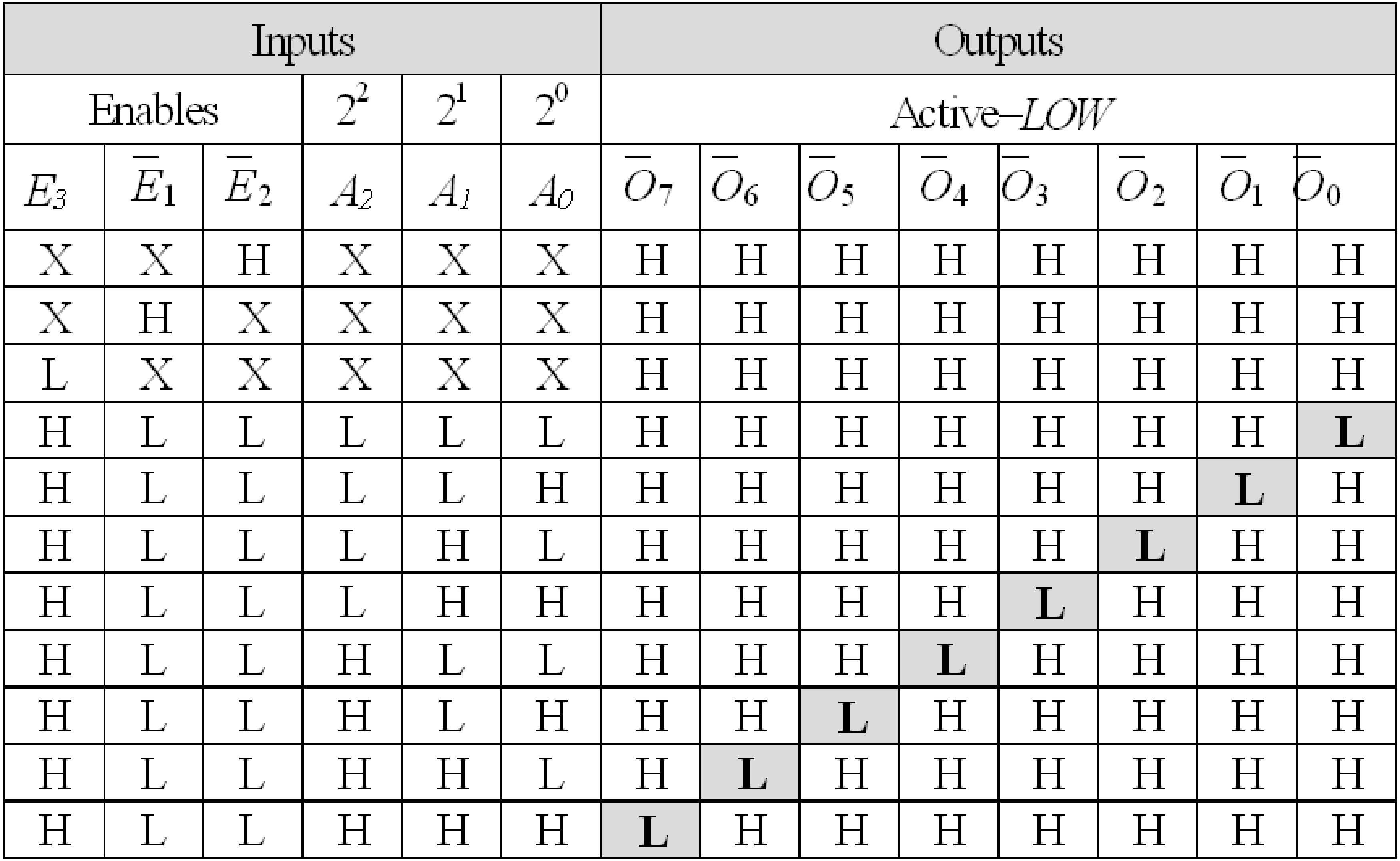
$\overline{E}_1$	$\overline{E}_2$	E_3	Outputs
0	0	1	Respond to input code $A_2A_1A_0$
1	X	X	Disabled — all HIGH
X	1	X	Disabled — all HIGH
X	X	0	Disabled — all HIGH

(b)


$\overline{E}_1$ $\overline{E}_2$ E_3
A_2 A_1 A_0
E
74ALS138
1-of-8 decoder
$\overline{O}_7$ $\overline{O}_6$ $\overline{O}_5$ $\overline{O}_4$ $\overline{O}_3$ $\overline{O}_2$ $\overline{O}_1$ $\overline{O}_0$

(c)

Truth table of 74138 (Example of a 3– 8 Bit Decoder) active-LOW



Inputs						Outputs							
Enables			2^2	2^1	2^0	Active–*LOW*							
E_3	$\overline{E}_1$	$\overline{E}_2$	A_2	A_1	A_0	$\overline{O}_7$	$\overline{O}_6$	$\overline{O}_5$	$\overline{O}_4$	$\overline{O}_3$	$\overline{O}_2$	$\overline{O}_1$	$\overline{O}_0$
X	X	H	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H	**L**
H	L	L	L	L	H	H	H	H	H	H	H	**L**	H
H	L	L	L	H	L	H	H	H	H	H	**L**	H	H
H	L	L	L	H	H	H	H	H	H	**L**	H	H	H
H	L	L	H	L	L	H	H	H	**L**	H	H	H	H
H	L	L	H	L	H	H	H	**L**	H	H	H	H	H
H	L	L	H	H	L	H	**L**	H	H	H	H	H	H
H	L	L	H	H	H	**L**	H	H	H	H	H	H	H

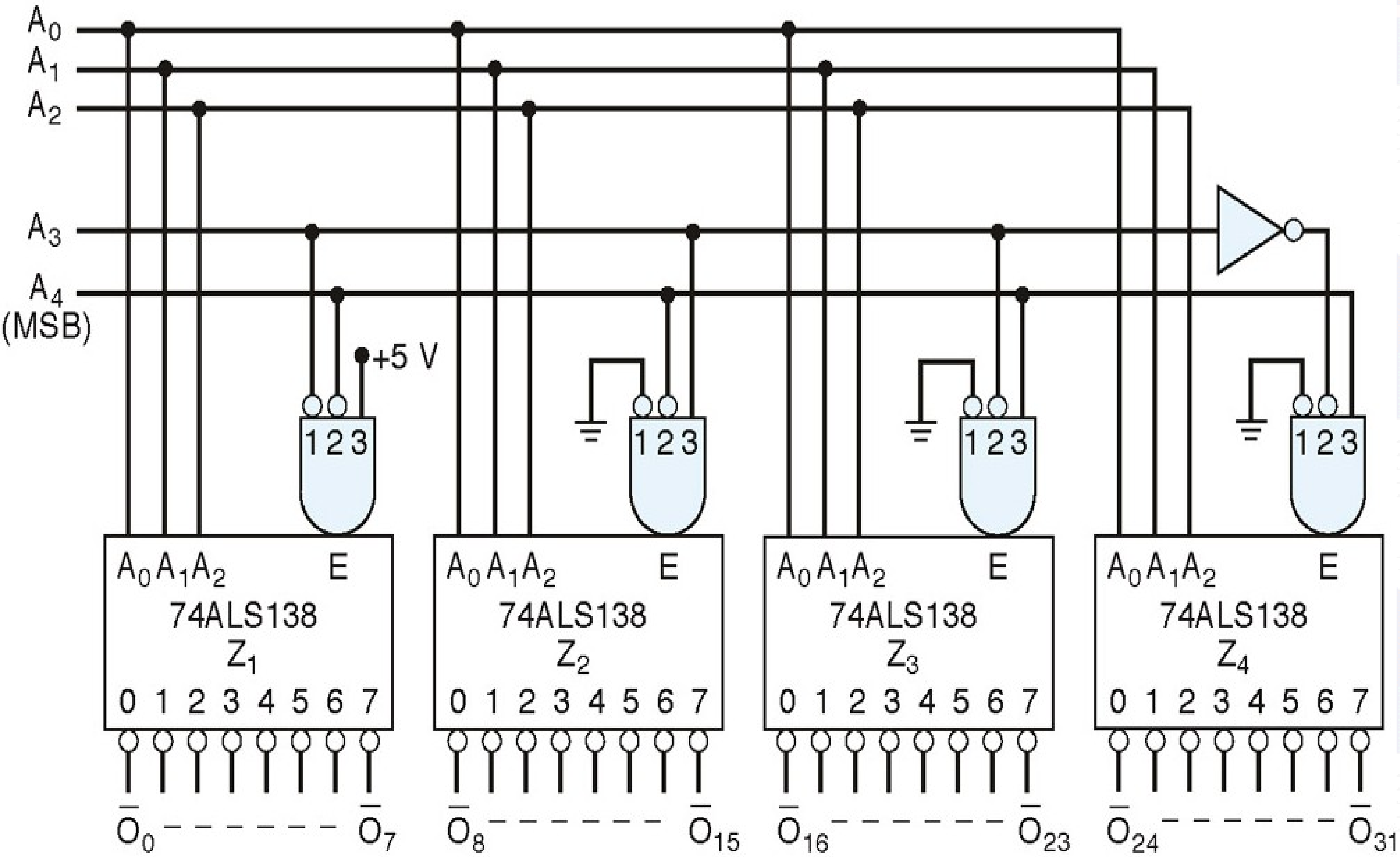
74138 (Example of a 3– 8 Bit Decoder)

- There is an enable function on this device, a *LOW* level on each input *E'*$_1$, and *E'*$_2$, and a *HIGH* level on input *E*$_3$, is required in order to make the enable gate output *HIGH*.

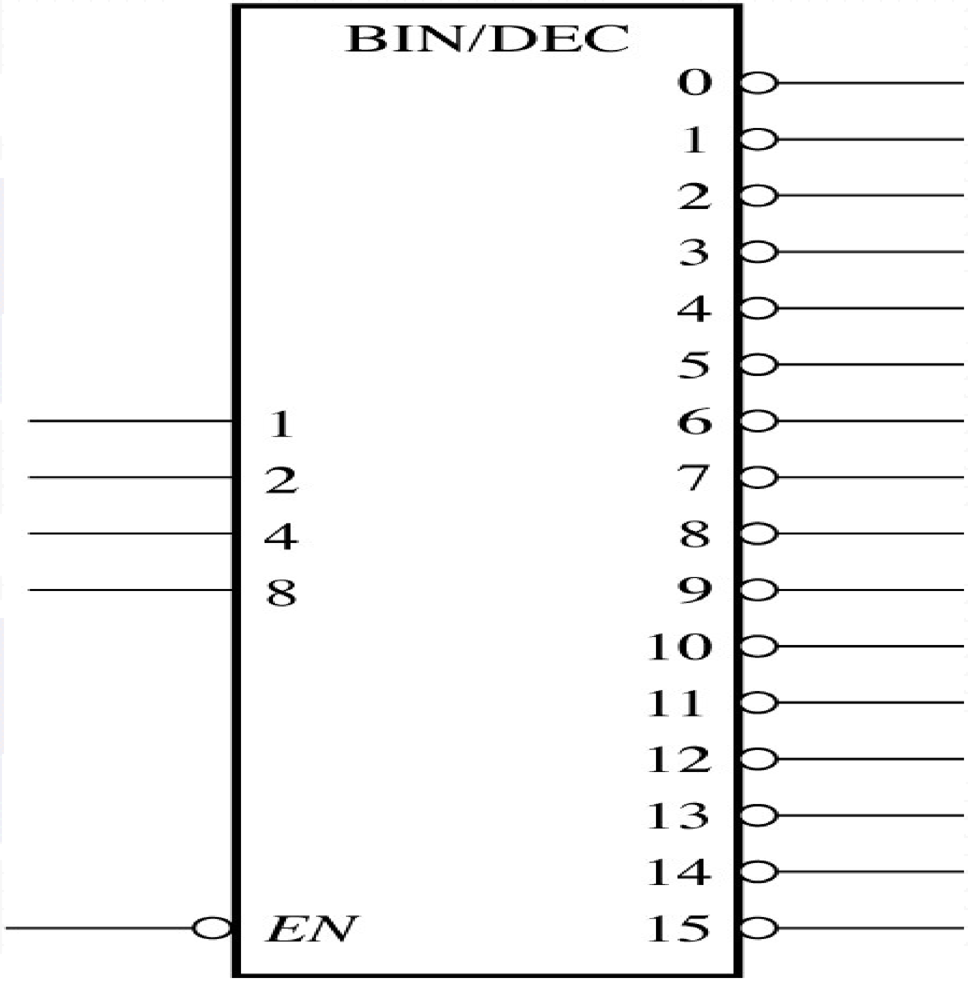
- The enable is connected to an input of each NAND gate in the decoder, so it must be *HIGH* for the NAND gate to be enabled.

- If the enable gate is not activated then all eight decoder outputs will be *HIGH* regardless of the states of the three input variables A_0, A_1, and A_2 .

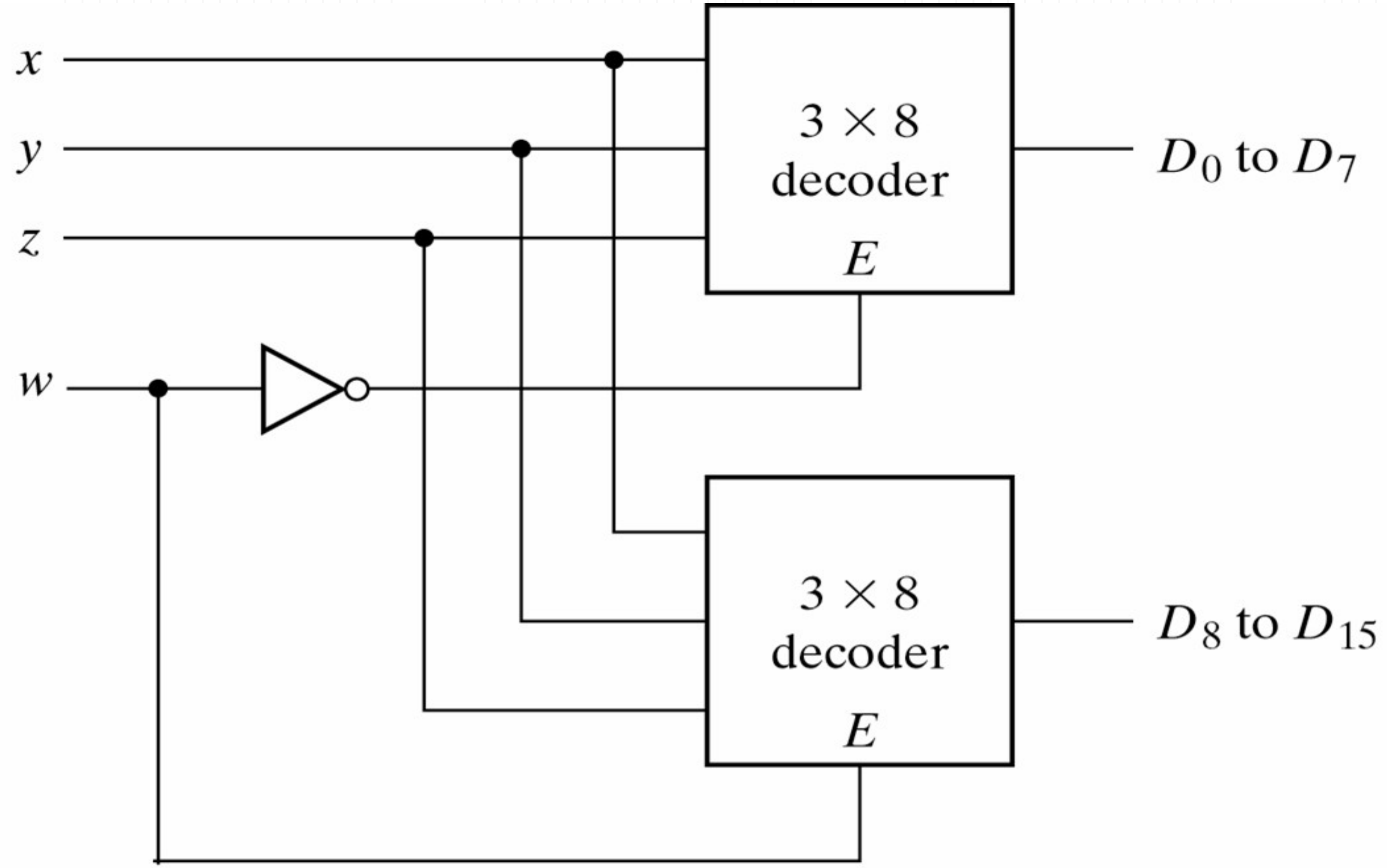
Example of a 5 to 32 Bit Decoder


A0
A1
A2
A3
A4
(MSB)
+5 V
1 2 3
1 2 3
1 2 3
1 2 3
A0 A1 A2 E
74ALS138
Z1
0 1 2 3 4 5 6 7
A0 A1 A2 E
74ALS138
Z2
0 1 2 3 4 5 6 7
A0 A1 A2 E
74ALS138
Z3
0 1 2 3 4 5 6 7
A0 A1 A2 E
74ALS138
Z4
0 1 2 3 4 5 6 7
$\overline{O}_0$ - - - - - - - $\overline{O}_7$
$\overline{O}_8$ - - - - - - - $\overline{O}_{15}$
$\overline{O}_{16}$ - - - - - - - $\overline{O}_{23}$
$\overline{O}_{24}$ - - - - - - - $\overline{O}_{31}$

Logic symbol for a 4-line-to-16-line (1-of-16) decoder. 74HC154



4-line-to-16 line Decoder constructed with two 3-line-to-8 line decoders (1)...

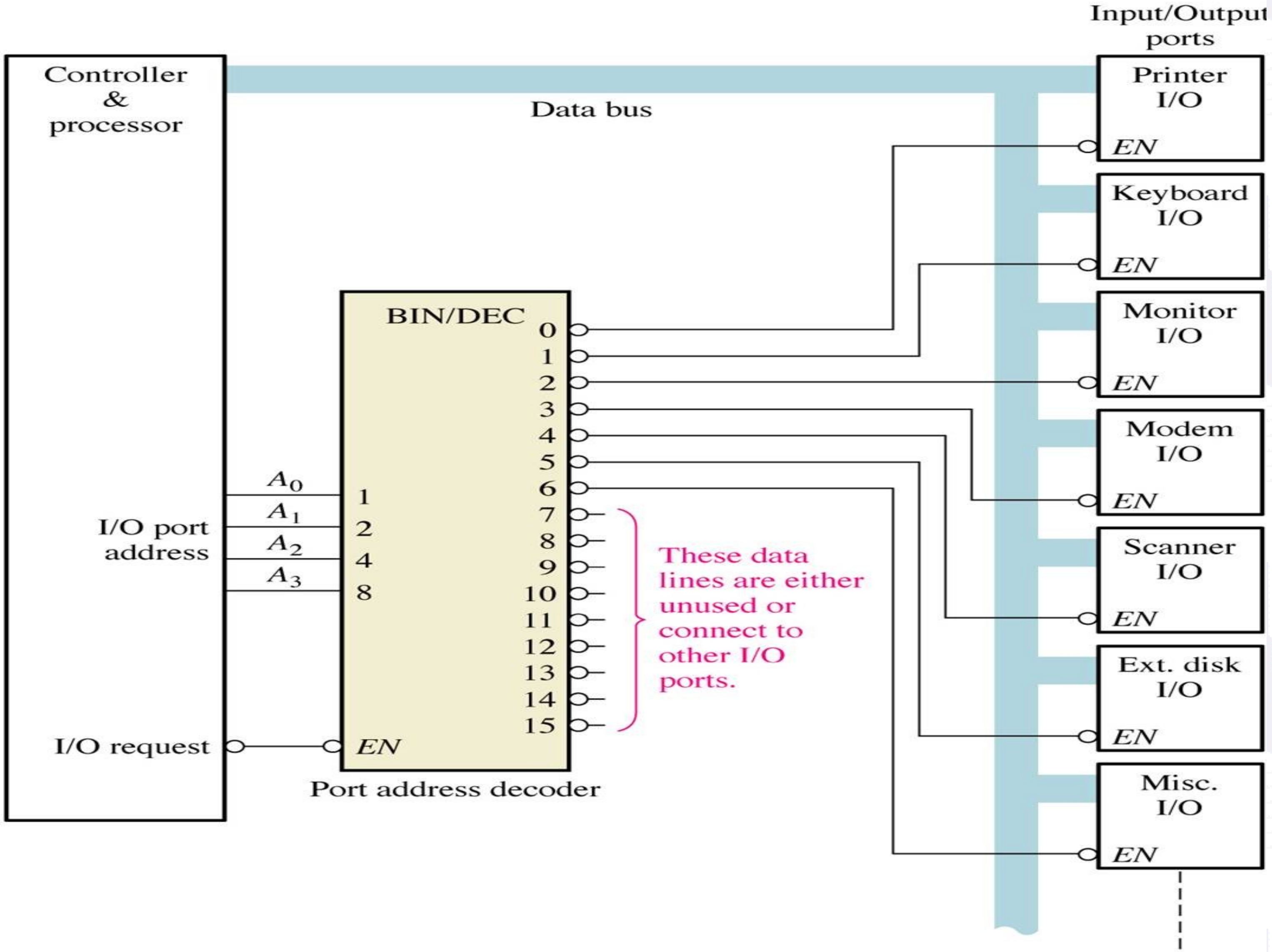


4-line-to-16 line Decoder constructed with two 3-line-to-8 line decoders (2)

- 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 min-terms 0000 to 0111.
- When w=1, the enable conditions are reversed. The bottom decoder outputs generate min-terms 1000 to 1111, while the outputs of the top decoder are all 0’s.

Application example

A simplified computer I/O port system with a port address decoder with only four address lines shown.



- Decoders are used in many types of applications. One example is in computers for I/O selection as in previous slide

- Computer must communicate with a variety of external devices called peripherals by sending and/or receiving data through what is known as input/output (I/O) ports

- Each I/O port has a number, called an address, which uniquely identifies it. When the computer wants to communicate with a particular device, it issues the appropriate address code for the I/O port to which that particular device is connected . The binary port address is decoded and appropriate decoder output is activated to enable the I/O port

- Binary data are transferred within the computer on a data bus, which is a set of parallel lines

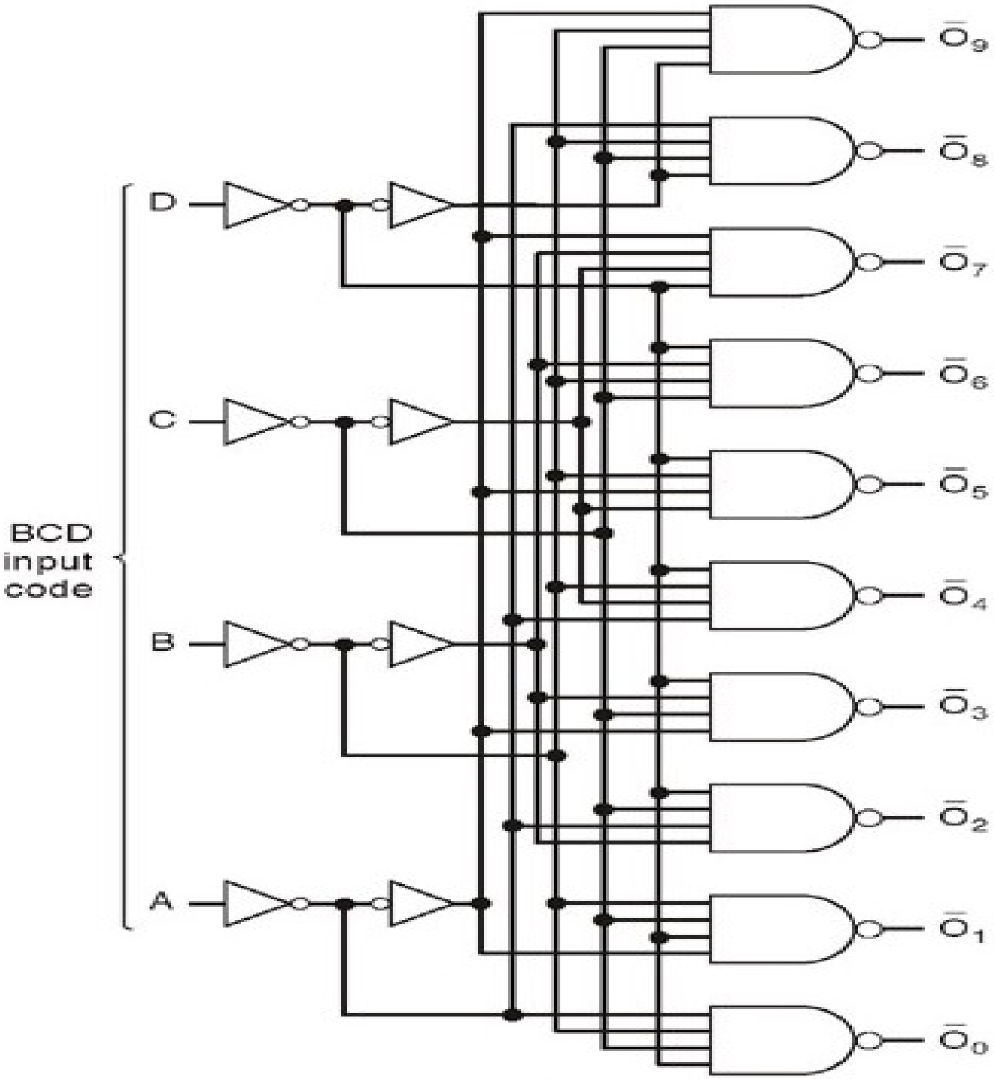
BCD -to- Decimal decoders

- The BCD- to-decimal decoder converts each BCD code into one of Ten Positionable decimal digit indications. It is frequently referred as a 4-line -to- 10 line decoder

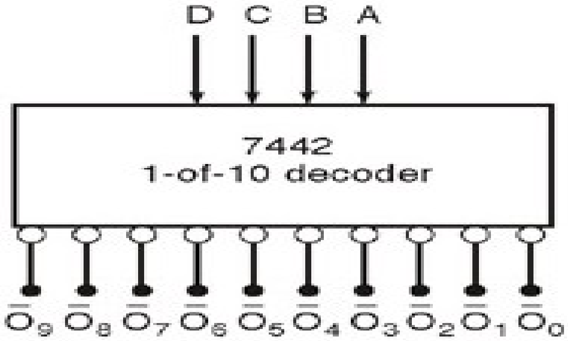
- The method of implementation is that only ten decoding gates are required because the BCD code represents only the ten decimal digits 0 through 9.

- Each of these decoding functions is implemented with NAND gates to provide active -LOW outputs. If an active HIGH output is required, AND gates are used for decoding

Logic diagram of BCD - decimal decoder (Active LOW output)



(a)

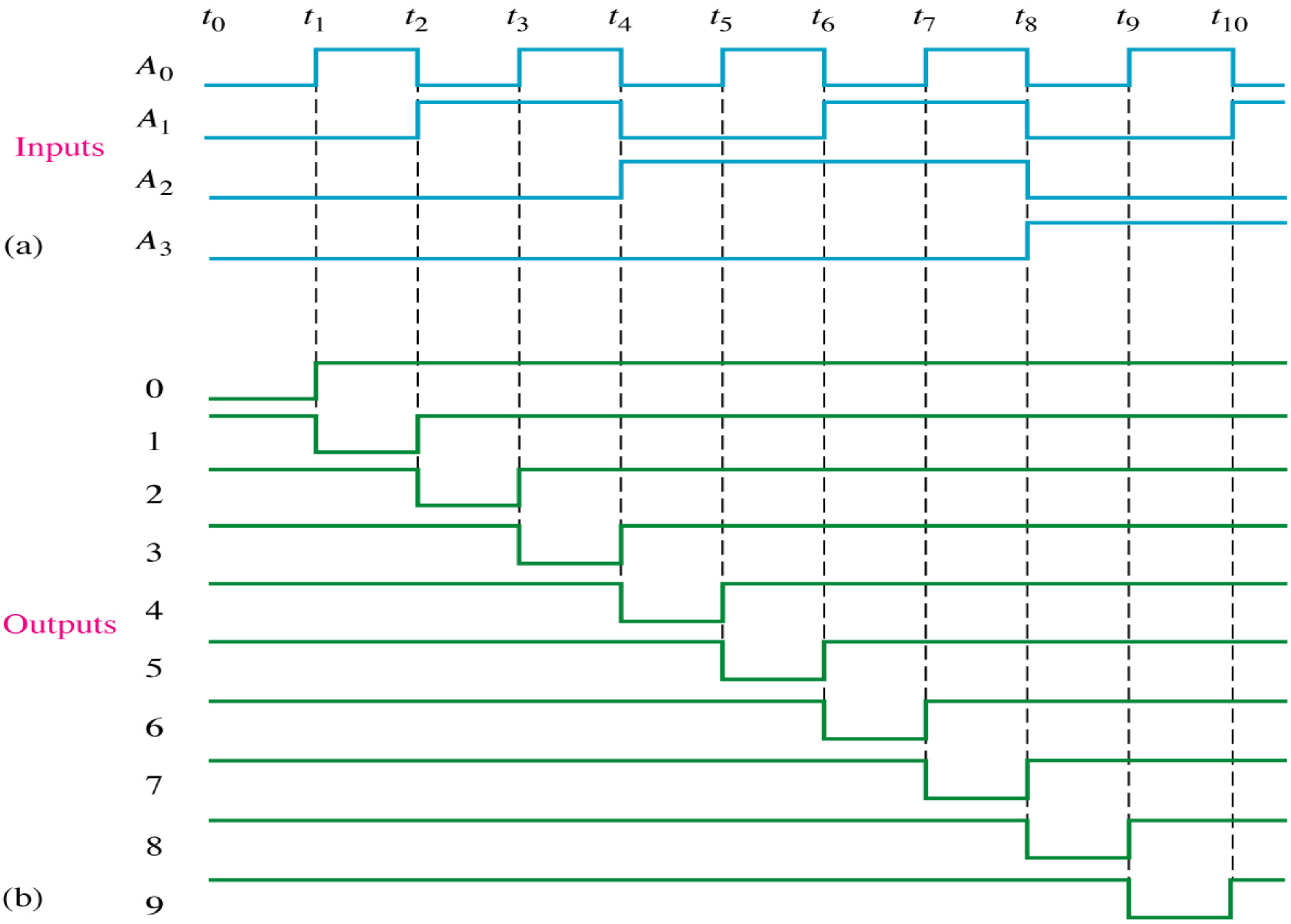

D C B A
7442
1-of-10 decoder
$\bar{O}_9$ $\bar{O}_8$ $\bar{O}_7$ $\bar{O}_6$ $\bar{O}_5$ $\bar{O}_4$ $\bar{O}_3$ $\bar{O}_2$ $\bar{O}_1$ $\bar{O}_0$

(b)

Inputs				
D	C	B	A	Active Output
L	L	L	L	$\bar{O}_0$
L	L	L	H	$\bar{O}_1$
L	L	H	L	$\bar{O}_2$
L	L	H	H	$\bar{O}_3$
L	H	L	L	$\bar{O}_4$
L	H	L	H	$\bar{O}_5$
L	H	H	L	$\bar{O}_6$
L	H	H	H	$\bar{O}_7$
H	L	L	L	$\bar{O}_8$
H	L	L	H	$\bar{O}_9$
H	L	H	L	None
H	L	H	H	None
H	H	L	L	None
H	H	L	H	None
H	H	H	L	None
H	H	H	H	None

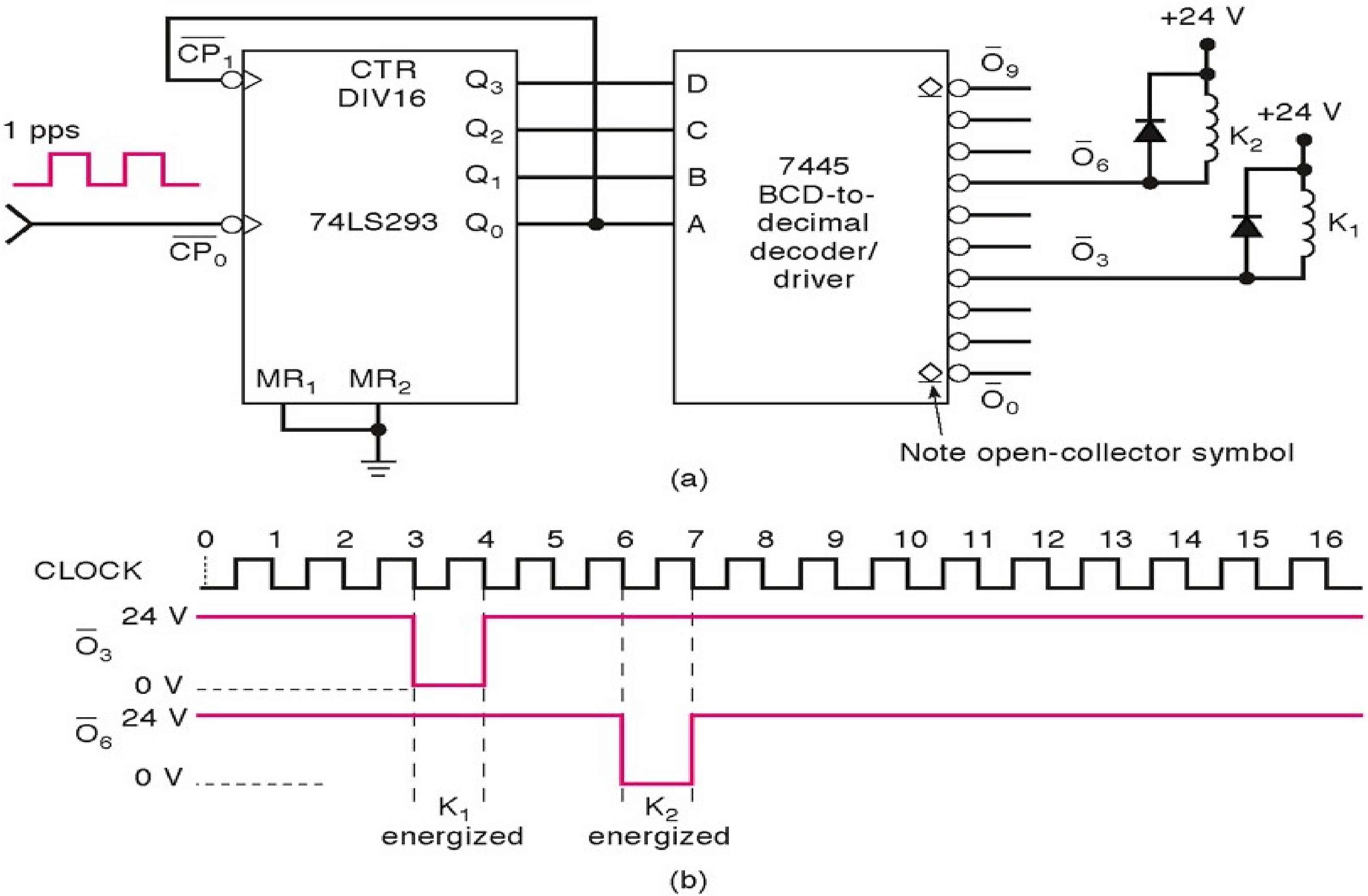
H = HIGH Voltage Level
L = LOW Voltage Level

(c)



Output Waveform for BCD Decoder

A Decoder Application - Counter -decoder combination used to provide timing and sequential operations (1)...

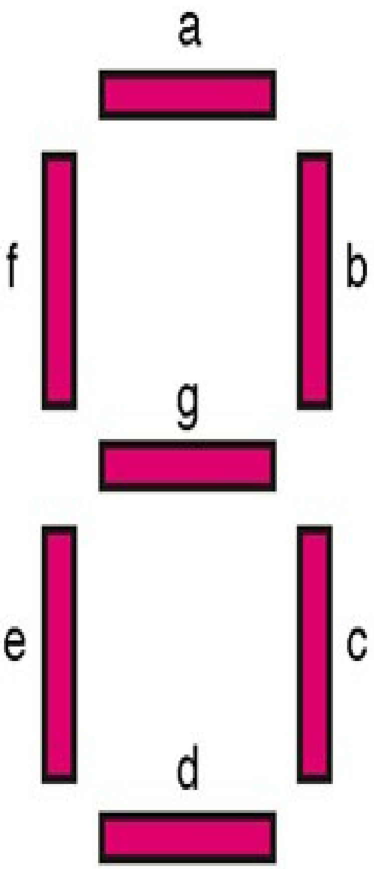


A Decoder Application - Counter -decoder combination used to provide timing and sequential operations (1)...

- Decoders are used whenever an output or a group of outputs is to be activated only on the occurrence of specific combination of input levels. These input levels are often provided by the outputs of a counter or register.

- When the decoder inputs come from a counter that is being continually pulsed, the decoder outputs will be activated sequentially, and there can be used as timing or sequencing signals to turn device on or off at specific times

BCD-7segment decoders/drivers

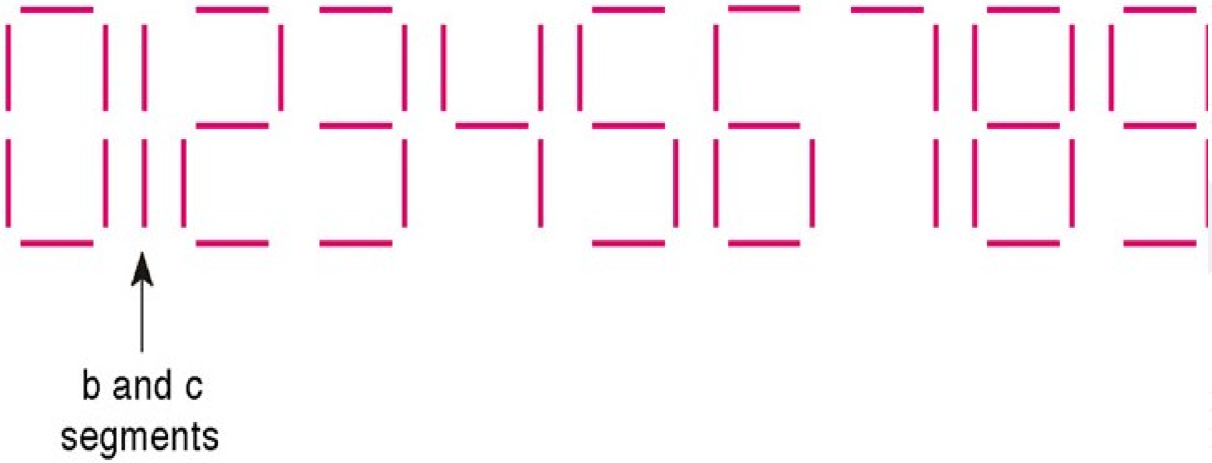


Most digital equipment has some means for displaying information in a form that can be understood by the user. This information is often numerical data but also be alphanumeric.

One of the simplest and most popular methods for displaying numerical digits uses a 7-segment configuration to form digital characters 0 to 9 and some times the hex characters A to F

One common arrangements uses light-emitting diodes (LED's) for each segment. By controlling the current thru each LED, some segments will be light and others will be dark so that desired character pattern will be generated

Figure shows the segment pattern that are used to display the various digits. For example, to display a "6" the segments a,c,d,e,f and g are made bright while segment b is dark

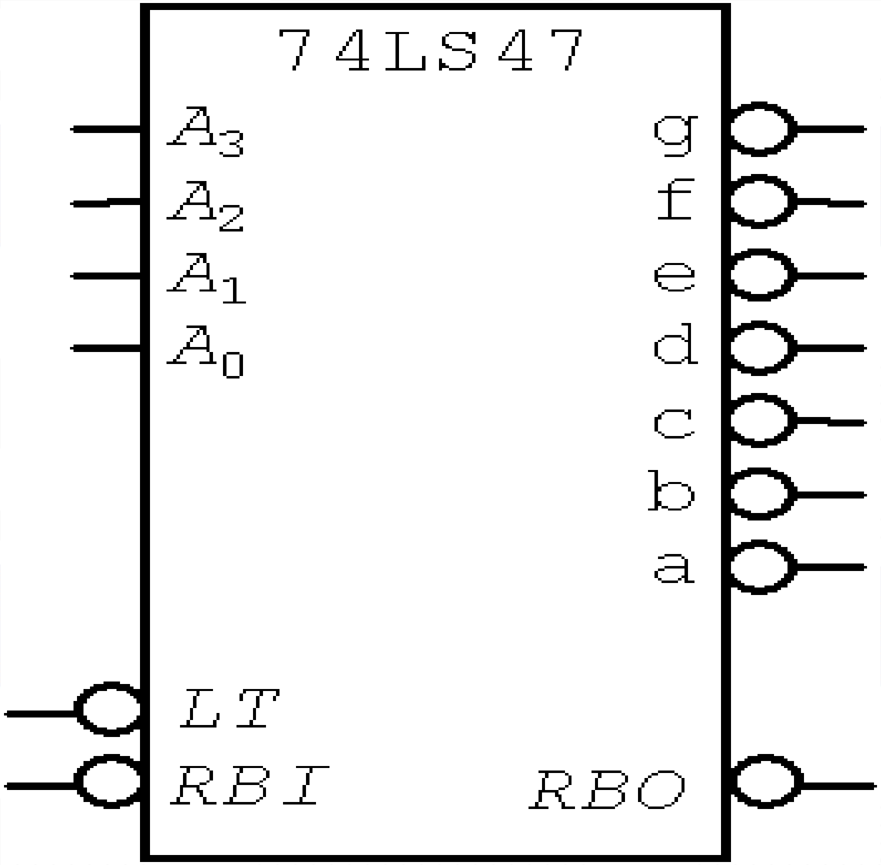


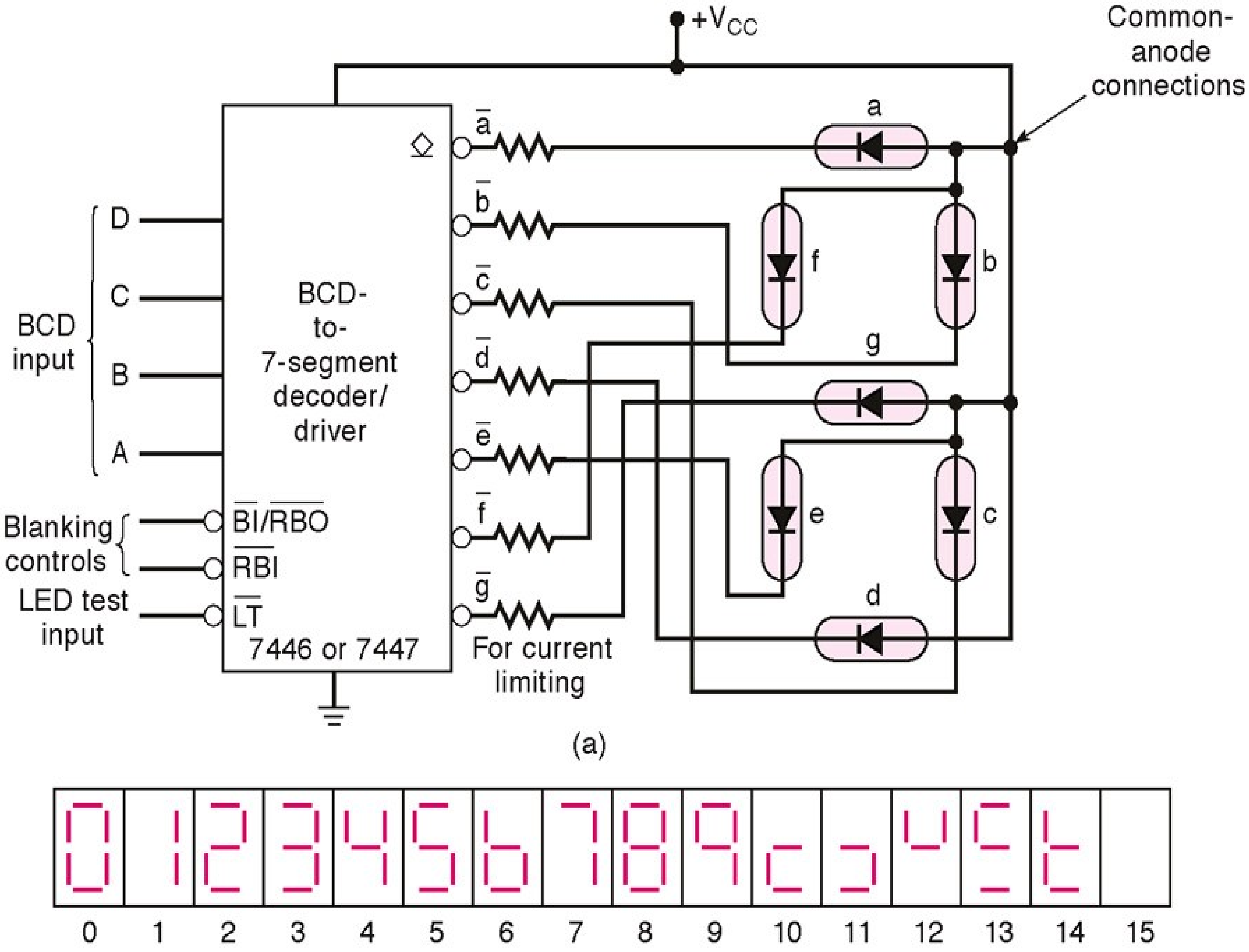
7-segment decoder

- A BCD-7 segment decoder/driver is used to take four-bit BCD input and provide the outputs that will pass current through the appropriate segments to display the decimal digit.

- The logic for this decoder is more complicated than the logic of decoders of earlier case, because each output is activated for more than one combination of inputs.

74LS47 (BCD–to–Seven–Segment Decoder)




+V_CC
Common-anode connections
a
b
c
d
e
f
g
ā
b̄
c̄
d̄
ē
f̄
ḡ
BCD input
D
C
B
A
BCD-to-7-segment decoder/driver
Blanking controls
BI/RBO
RBI
LED test input
LT
7446 or 7447
For current limiting
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

(a)

(b)

Lamp Test (LT)

- When $\overline{\text{LT}}$ = Low, $\overline{\text{BI/RBO}}$ = HIGH then all of the 7 segments in display are turned zero, LT is used to verify that no segments are burned out

Zero Suppression ($\overline{\text{BI}}$, $\overline{\text{RBI}}$, $\overline{\text{RBO}}$)

- Zero suppression is a feature used for multi digit displays to blank out unnecessary zeros.

Example:
In a 6-digit display the number 6.4 may be displayed as 006.400 if the zeros are not blanked out

- **Leading Zero Suppression**

Blanking the zeros at the front of a numbers

- **Trailing Zero Suppression**

Blanking the zeros at the back of the number

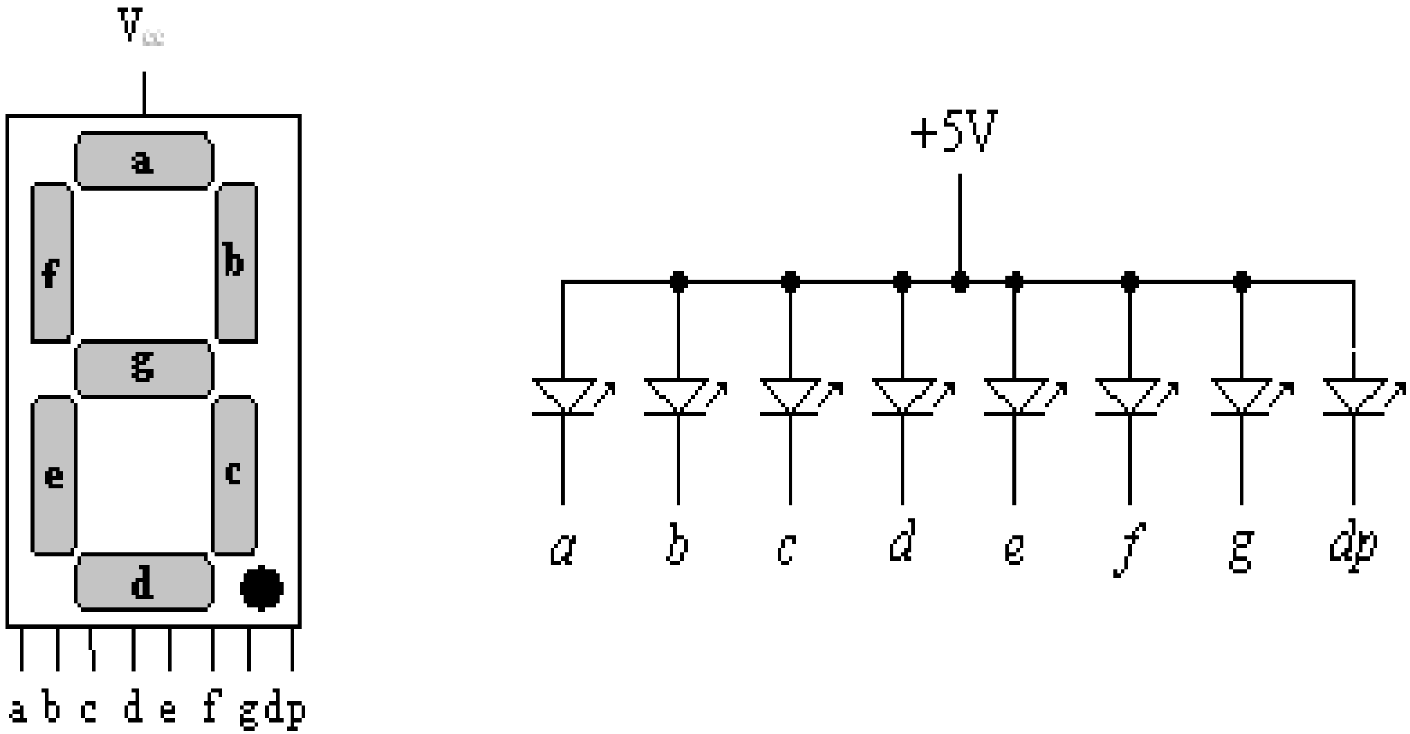
Only nonessential zeros are blanked, the number 030.080 will be displayed as 30.08 (the essential zeros remain)

7-segment display

- There are two types of 7–segment LED displays;
- A) common - anode
- B) common – cathode

Common Anode

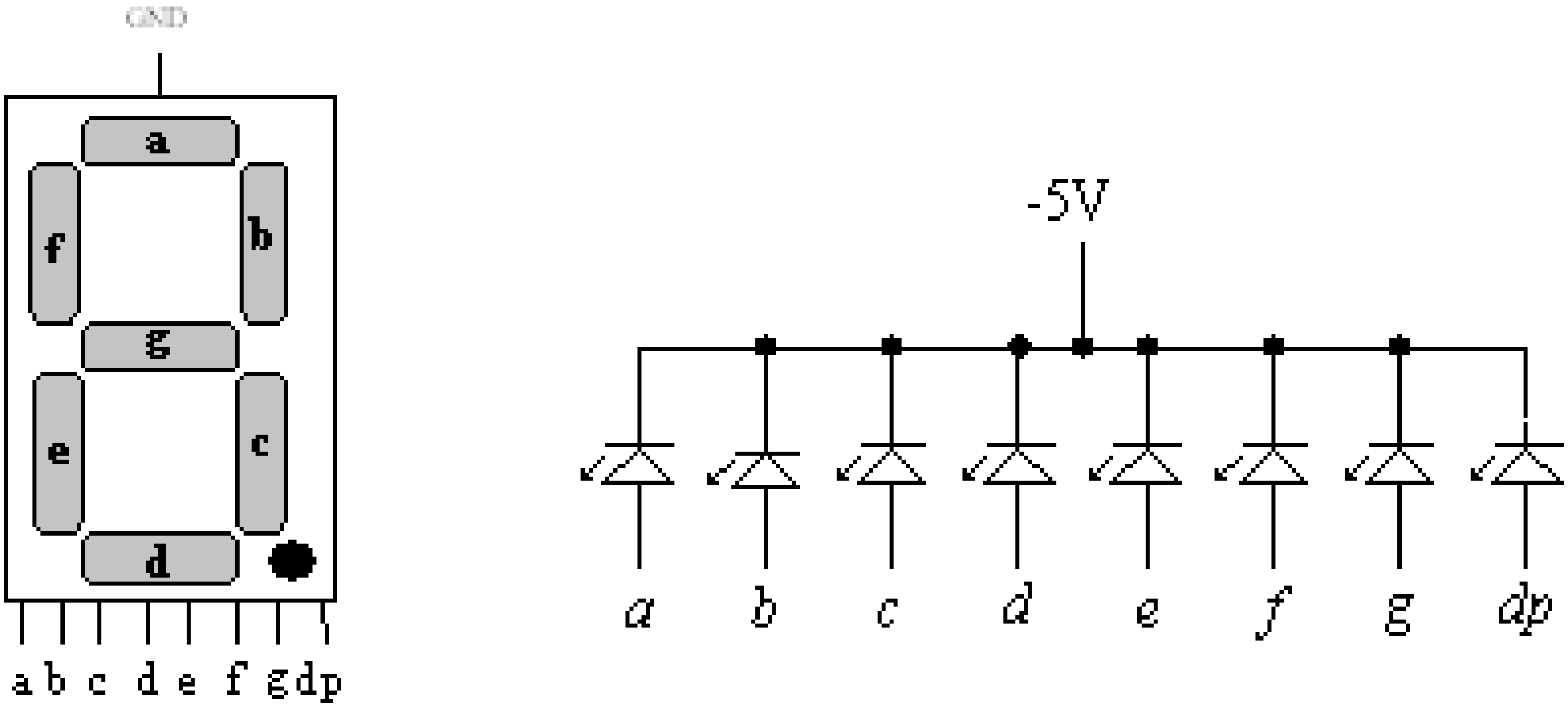
In common–anode, the anode of all of the LEDs are tied together to positive of the power supply (V_{cc}) as shown



Common–anode 7–segment LED display.

Common Cathode

- In common–cathode, the cathode of all of the LEDs are tied together to ground as shown.



Common–cathode 7–segment LED display.

Combinational Logic Circuit Implementation using a Decoder

- Any combinational logic circuit with n inputs and m outputs can be implemented with an n-to-2n-line decoder and m OR gates.
- Procedure:
  - Express the given Boolean function in sum of min-terms
  - Choose a decoder to generate all the min-terms of the input variables.
  - Select the inputs to each OR gate from the decoder outputs according to the list of min-term for each function.

Combinational Logic Circuit Implementation using a Decoder - An example (1)

- From the truth table of the full adder,

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

- the functions can be expressed in sum of min-terms.

$$S(x,y,z) = \Sigma m(1,2,4,7)$$

$$C(x,y,z) = \Sigma m(3,5,6,7)$$

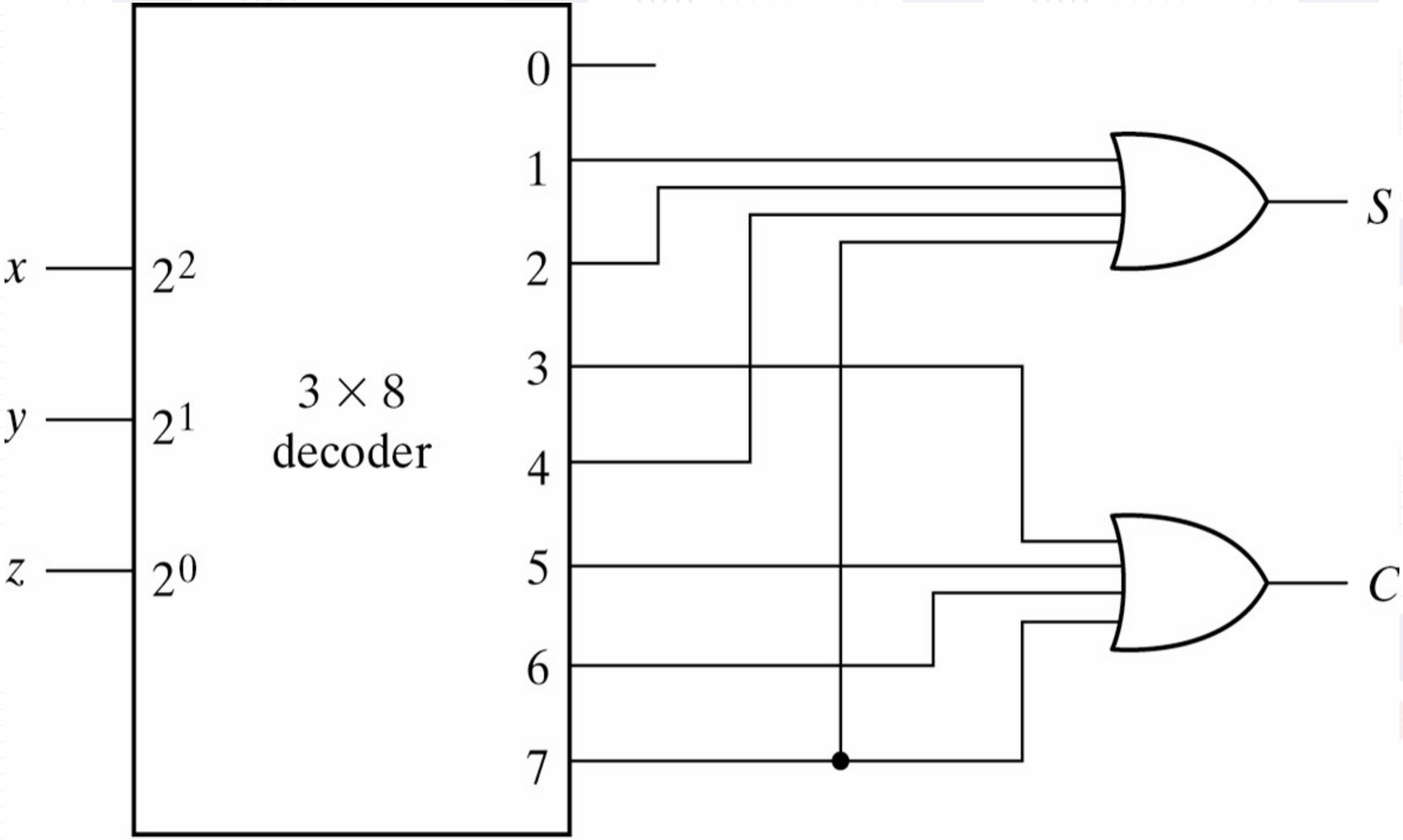
where Σ indicates sum, m indicates min-term and the number in brackets indicate the decimal equivalent

Combinational Logic Circuit Implementation using a Decoder - An example (2)

Since there are three inputs and a total of eight min-terms, we need a 3-to-8 line decoder.

- The decoder generates the eight min-terms for x,y,z
- The OR gate for output S forms the logical sum of min-terms 1,2,4, and 7.
- The OR gates for output C forms the logical sum of min-terms 3,5,6, and 7

Combinational Logic Circuit Implementation using a Decoder - example (3)



Implementation of a Full Adder with a Decoder

Encoder

- An encoder is a combinational logic circuit that essentially performs a “reverse” of decoder functions.

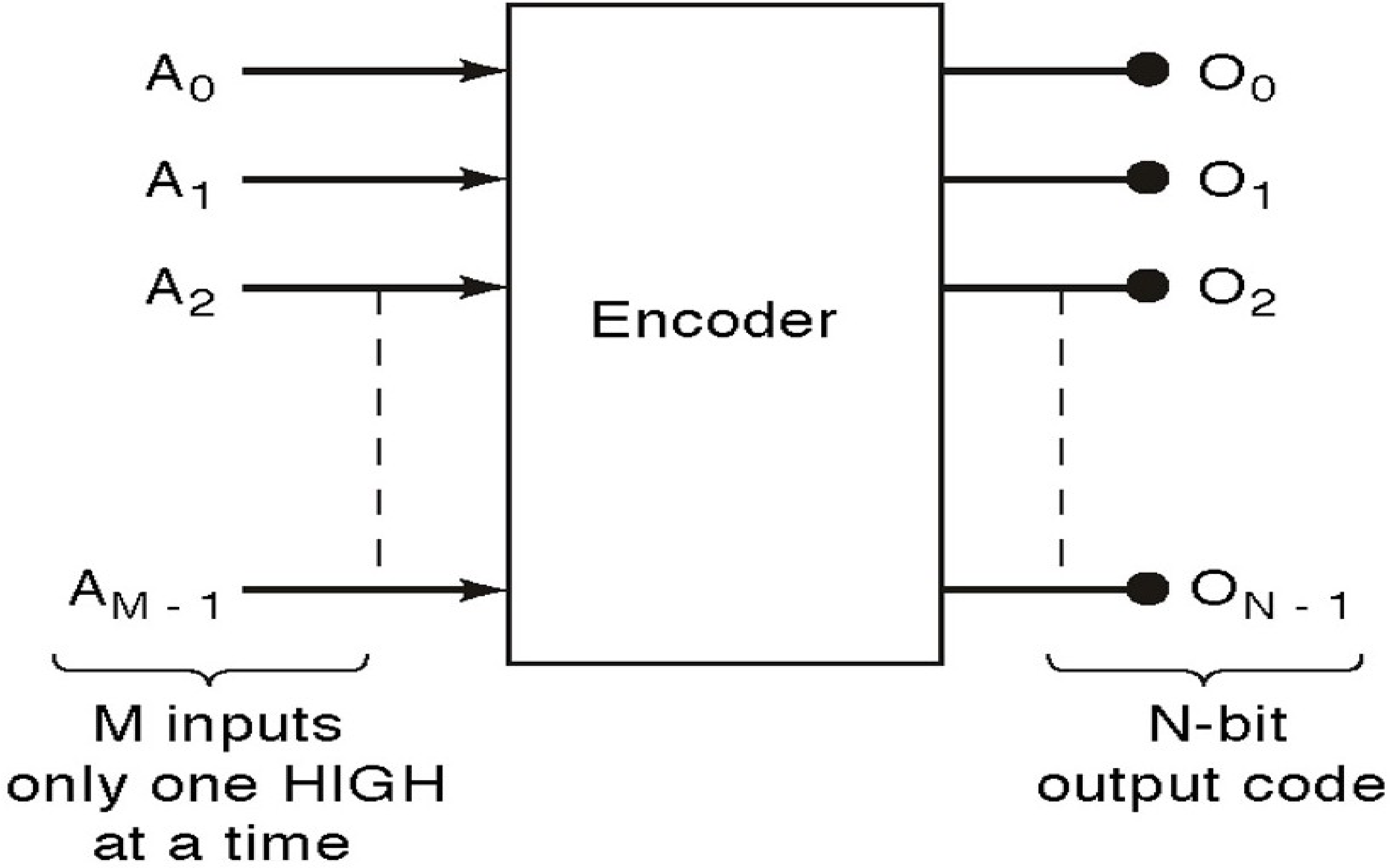
- An encoder accepts an active level on one of its inputs, representing digit, such as a decimal or octal digits, and converts it to a coded output such as BCD or binary.

- Encoders can also be devised to encode various symbols and alphabetic characters.

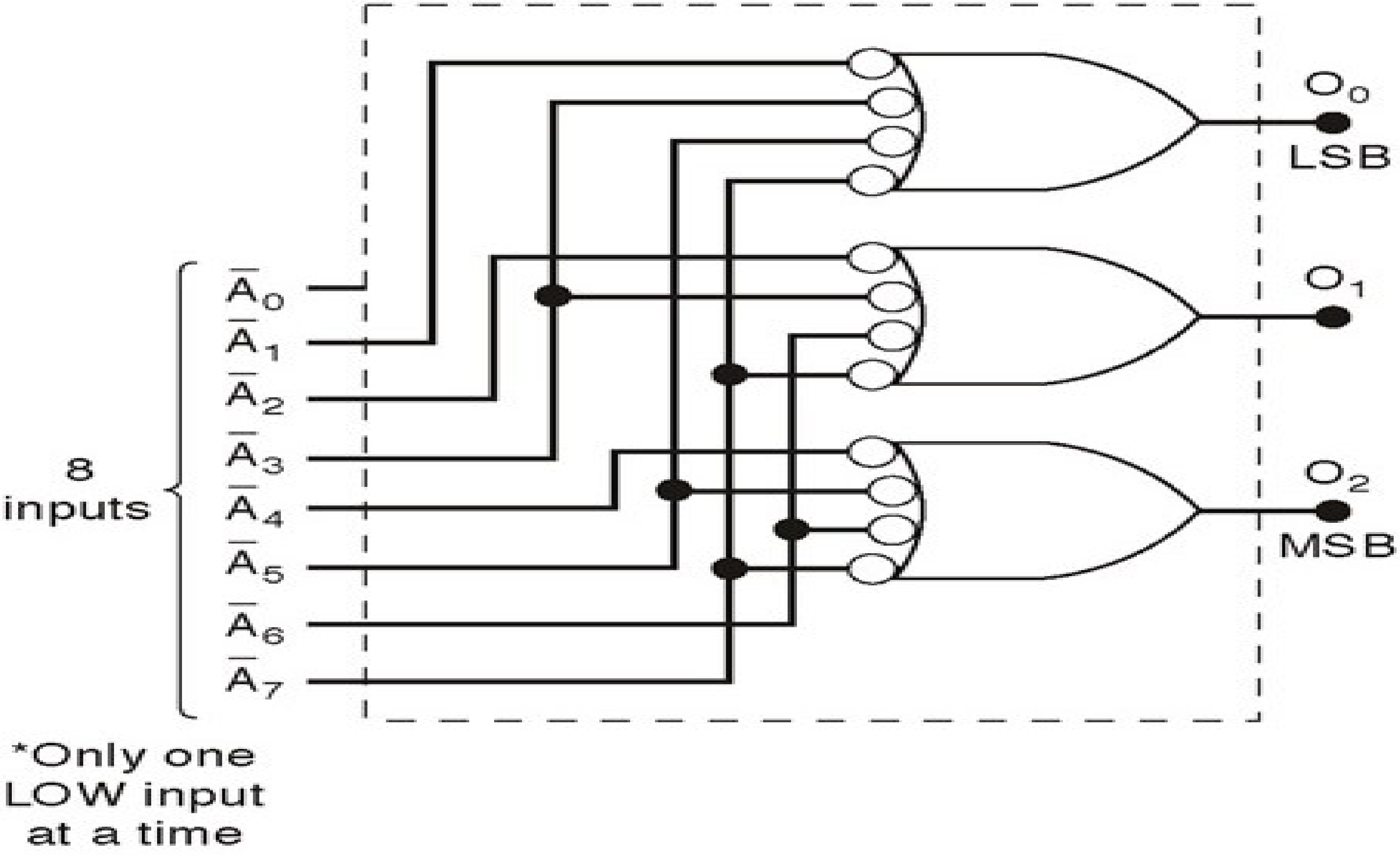
- The process of converting from familiar symbols or numbers to a coded format is called ***encoding***.

- Most decoders accept an input code and produce a HIGH
- (or a LOW) at one and only one output line. In otherworlds , a decoder identifies, recognizes, or detects a particular code. The opposite of this decoding process is called encoding and is performed by a logic circuit called an encoder.
- An encoder has a number of input lines, only one of which input is activated at a given time and produces an N-bit output code,depending on which input is activated.

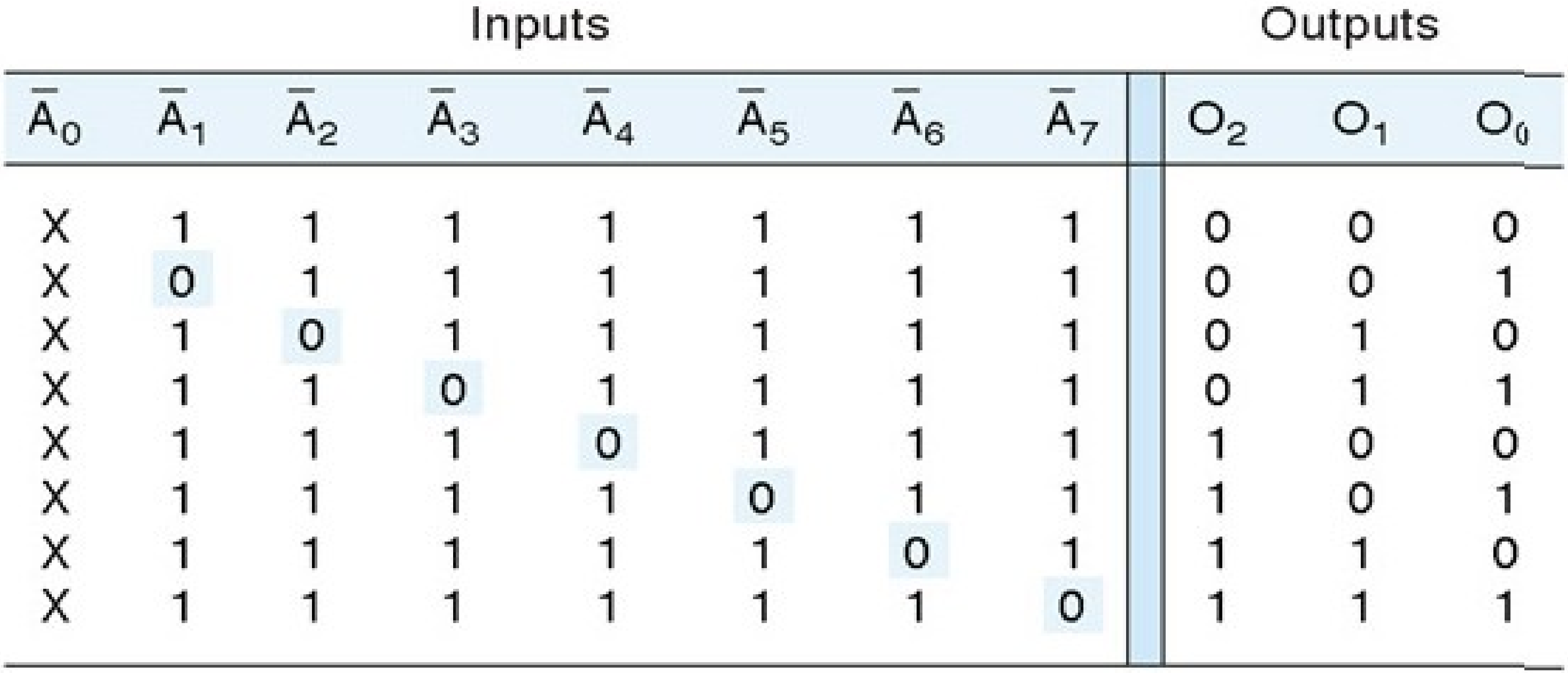
General encoder diagram



Logic circuit for octal-to binary encoder [8-line- 3-line]



Truth table for octal-to binary encoder [8-line- 3-line]



Inputs								Outputs		
$\overline{A_0}$	$\overline{A_1}$	$\overline{A_2}$	$\overline{A_3}$	$\overline{A_4}$	$\overline{A_5}$	$\overline{A_6}$	$\overline{A_7}$	O_2	O_1	O_0
X	1	1	1	1	1	1	1	0	0	0
X	0	1	1	1	1	1	1	0	0	1
X	1	0	1	1	1	1	1	0	1	0
X	1	1	0	1	1	1	1	0	1	1
X	1	1	1	0	1	1	1	1	0	0
X	1	1	1	1	0	1	1	1	0	1
X	1	1	1	1	1	0	1	1	1	0
X	1	1	1	1	1	1	0	1	1	1

A low at any single input will produce the output binary code corresponding to that input. For instance , a low at A_3’ will produce O_2 =0, O_1=1 and O_0 =1, which is binary code for 3. A_o’ is not connected to the logic gates because the encoder outputs always be normally at 0000 when none of the inputs is LOW

Design of 4-input Priority Encoder (4-line-to 2 line priority encoder) (1)...

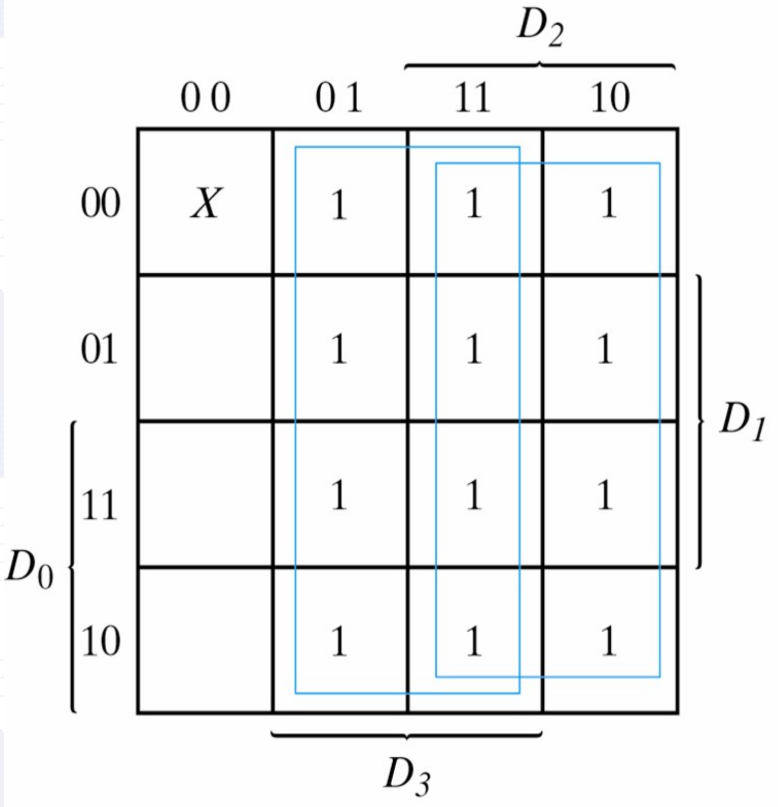
- A priority encoder is an encoder that includes the priority function
- If two or more inputs are equal to 1 at the same time, the input having the highest priority will take precedence.
- **Truth Table of a 4-input Priority Encoder:**

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**

Design of 4-input Priority Encoder (4-line-to 2 line priority encoder) (2)...

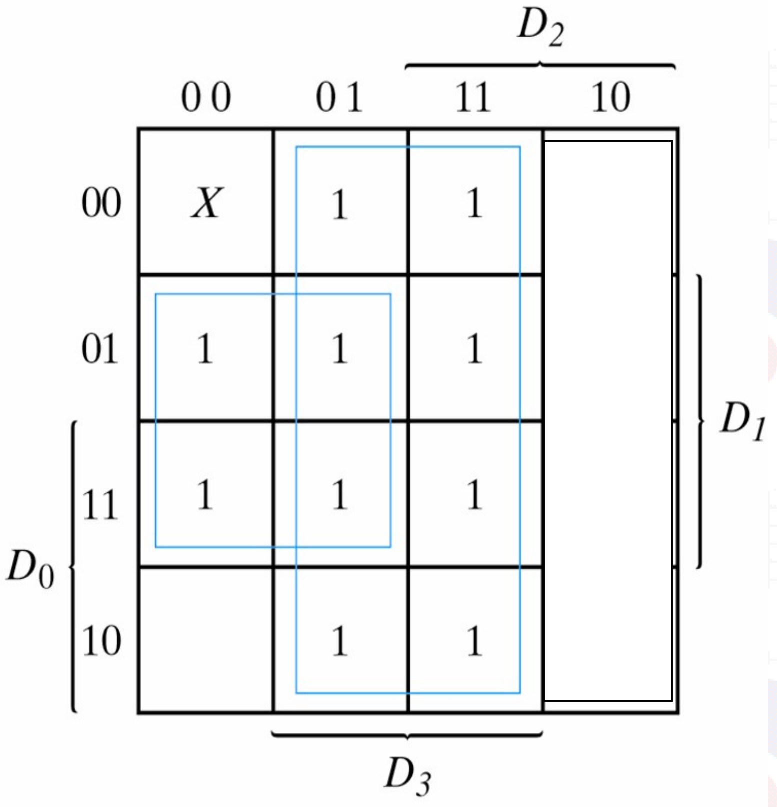
- In addition to two outputs x, and y, the truth table has a third output designated by V, which is a valid bit indicator that is set 1 when one or more inputs are equal to 1. If all inputs are 0, there is no valid input and V is equal to 0.
- X’s in the output column indicate don’t care conditions, the X’s in the input columns are useful for representing a truth table in condensed form.
- The higher the subscript number, the higher the priority of the input. Input D3 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)

Design of 4-input Priority Encoder (4-line-to 2 line priority encoder) (3)...



	00	01	11	10
00	X	1	1	1
01		1	1	1
11		1	1	1
10		1	1	1

$x = D_2 + D_3$



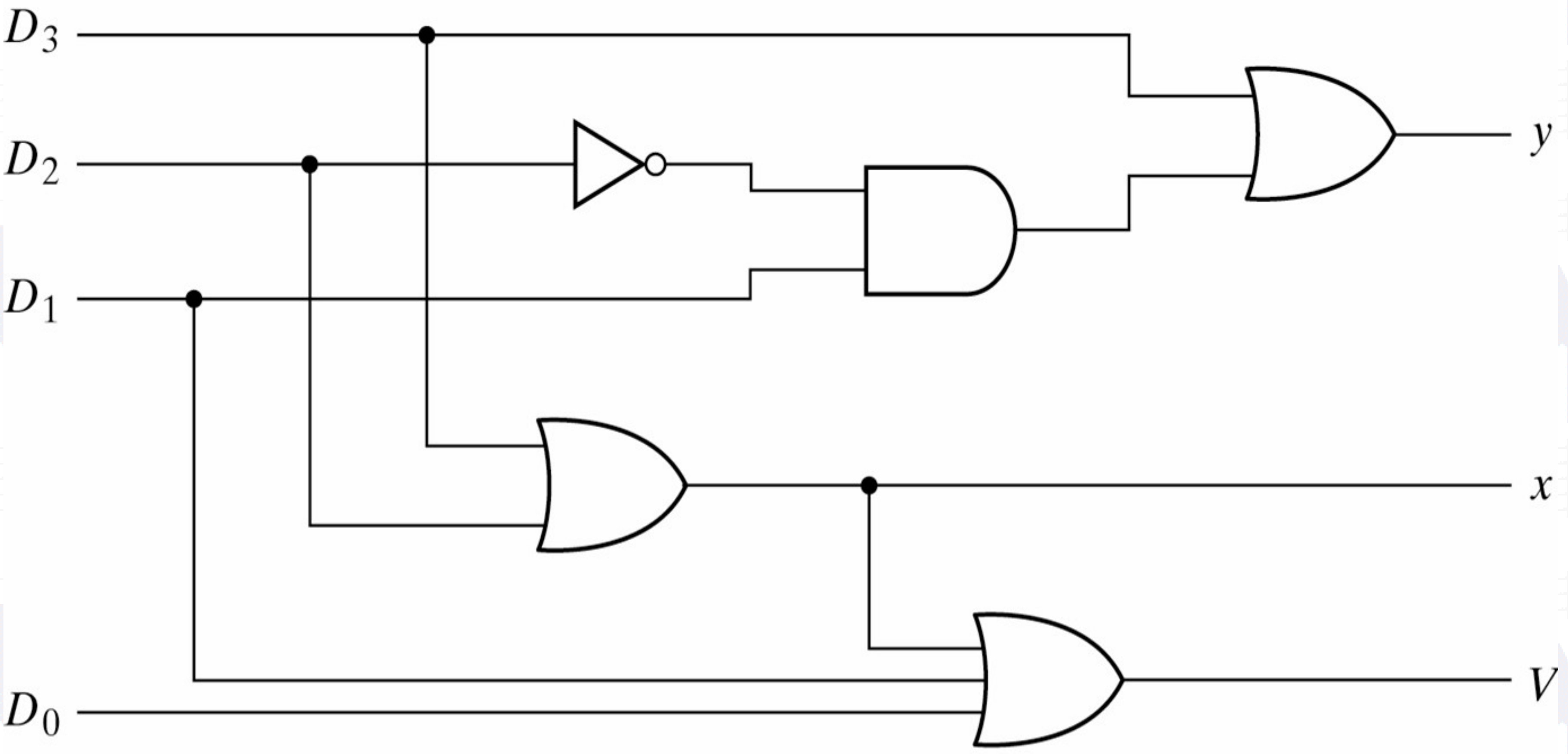
	00	01	11	10
00	X	1	1	
01	1	1	1	
11	1	1	1	
10		1	1	

$y = D_3 + D_1D'_2$

$V=D_0+D_1+D_2+D_3$

K-Maps for 4-input Priority Encoder

Design of 4-input Priority Encoder (4-line-to 2 line priority encoder) (4)



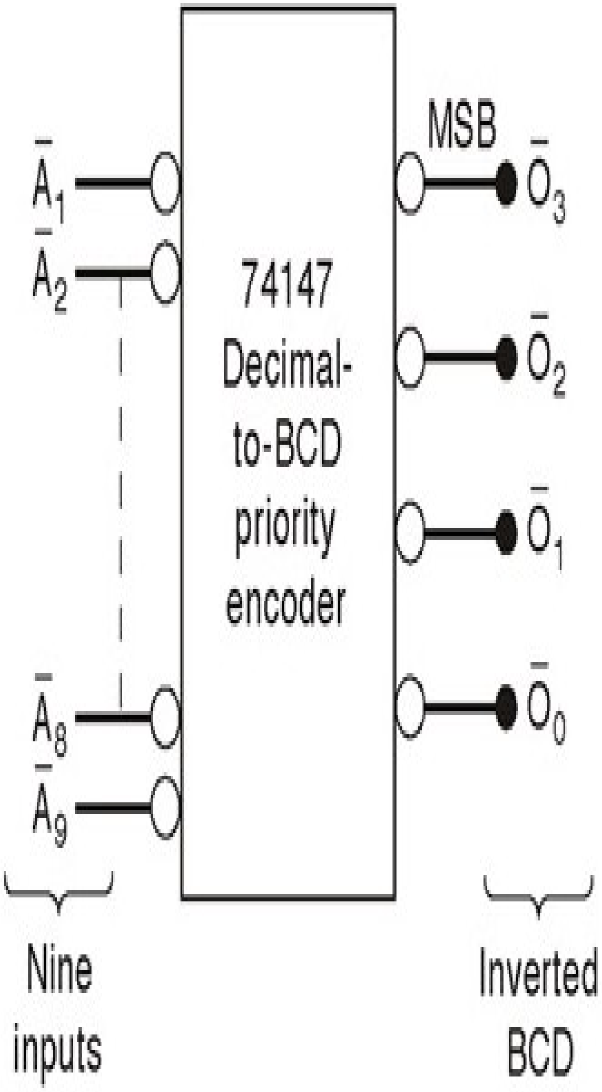
Logic Diagram for 4-input priority encoder

Decimal-BCD priority encoder

•Encoder will produce a BCD output corresponding to the highest-order decimal digit input that is active and will ignore any other lower order active inputs.

•For instance if the input 6 and the 3 are active, the output will be 1001, which is the inverse value of BCD output 0110 (which represents decimal 6)

74147 decimal-BCD priority encoder

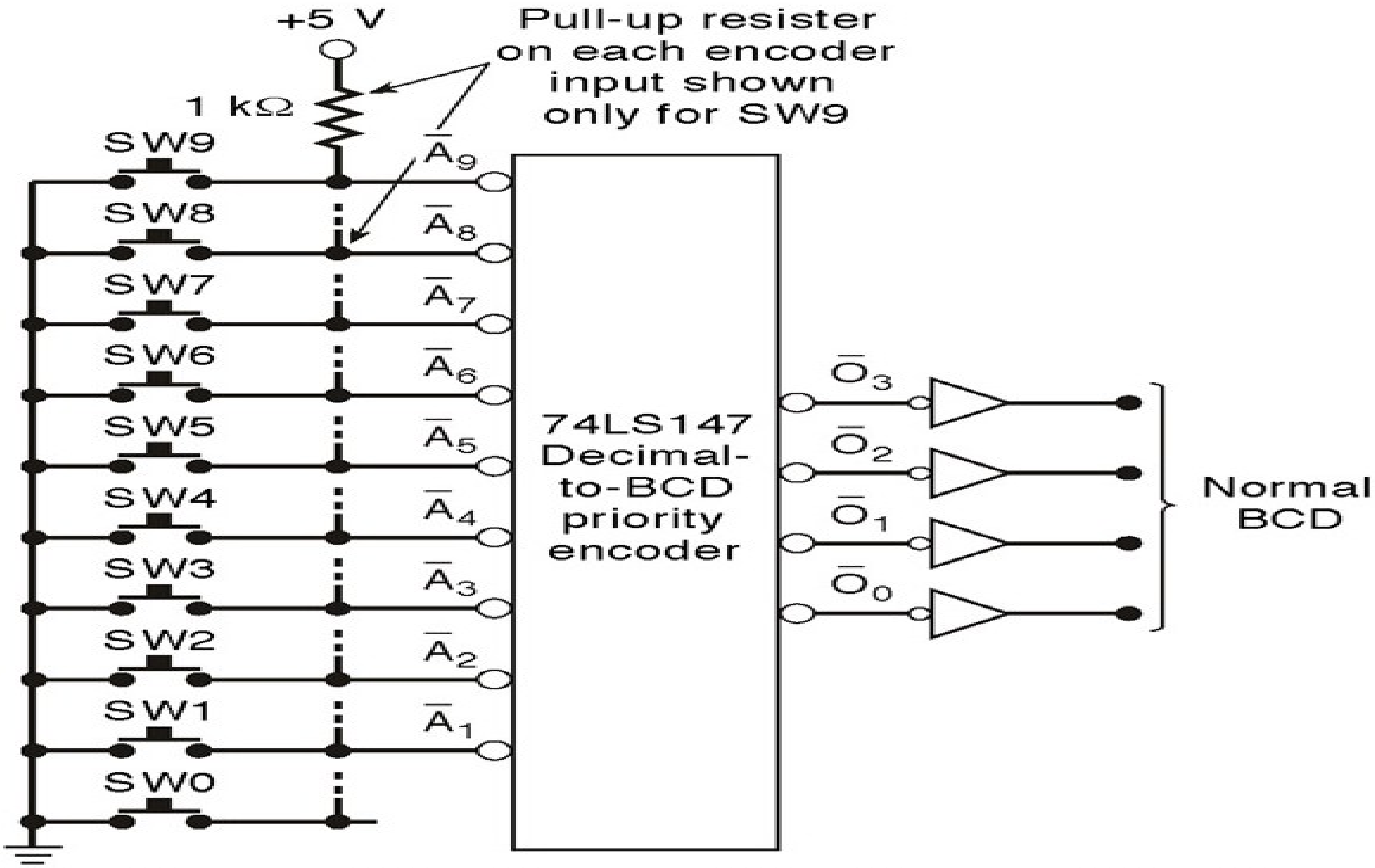


$\bar{A}_1$	$\bar{A}_2$	$\bar{A}_3$	$\bar{A}_4$	$\bar{A}_5$	$\bar{A}_6$	$\bar{A}_7$	$\bar{A}_8$	$\bar{A}_9$	$\bar{O}_3$	$\bar{O}_2$	$\bar{O}_1$	$\bar{O}_0$
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

X = either 0 or 1

When A9’ is low, the output is 0110, which is inverse of 1001 (eq to 9 in BCD)

Decimal- BCD switch decoder



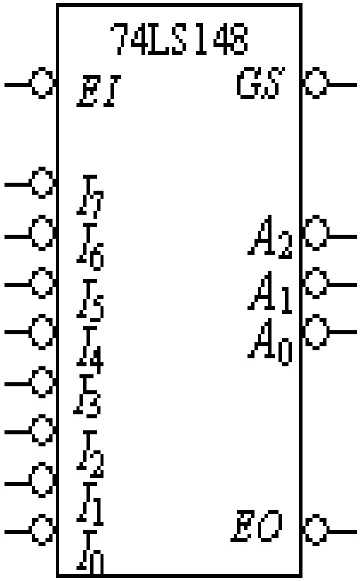
The output of the decoder are inversed to produce the normal BCD value

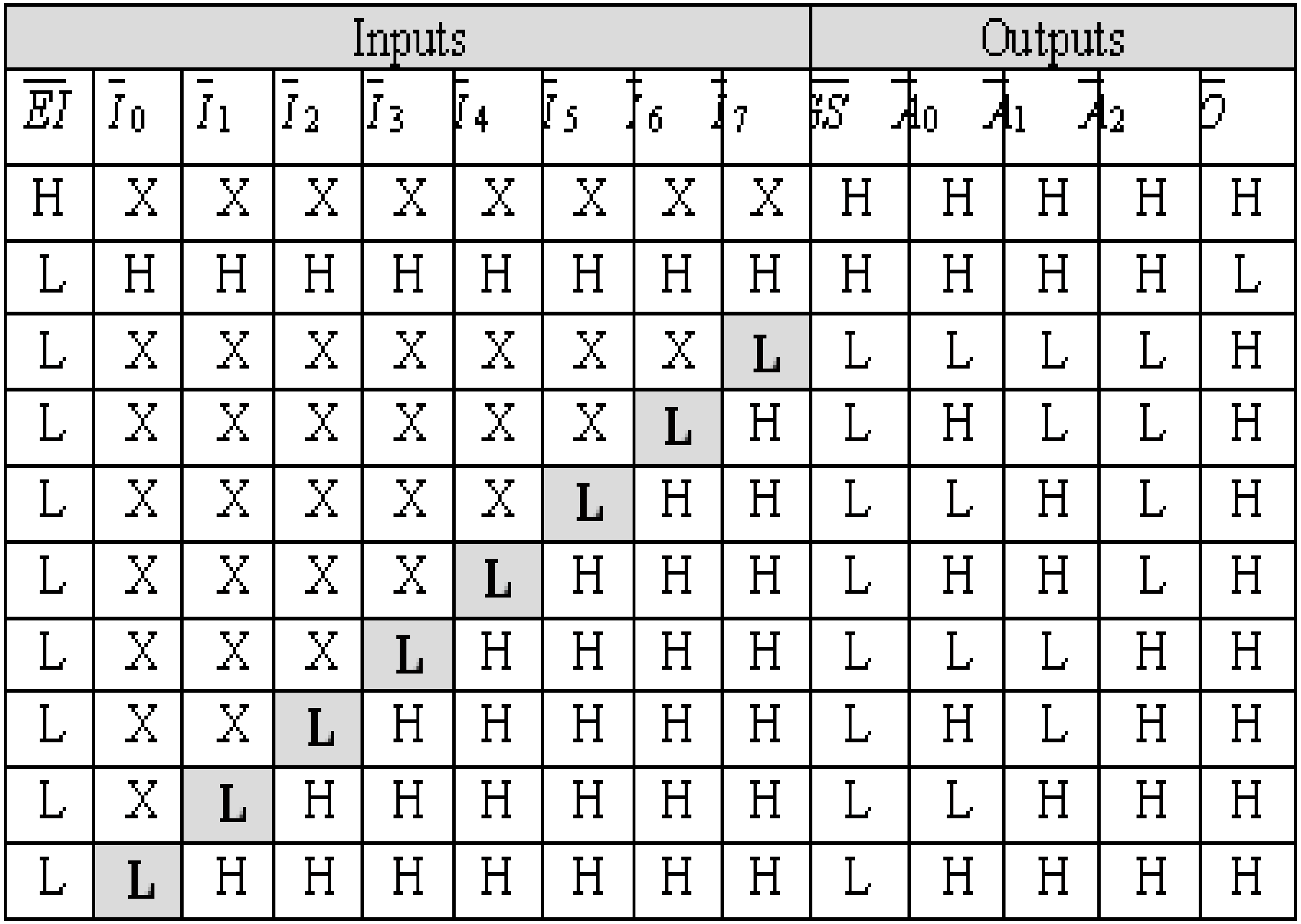
The Octal–to–Binary Priority Encoder- Example

- The 74LS148 is a priority encoder that has eight active *LOW* inputs and three active–*LOW* binary outputs

- To enable the device, the *EI* (enable input) must be *LOW*. It also has the *EO* (enable output) and *GS* (group signal output) for expansion purposes.

The Octal–to–Binary Encoder





Inputs									Outputs				
$\overline{EI}$	$\overline{I}_0$	$\overline{I}_1$	$\overline{I}_2$	$\overline{I}_3$	$\overline{I}_4$	$\overline{I}_5$	$\overline{I}_6$	$\overline{I}_7$	$\overline{GS}$	$\overline{A}_0$	$\overline{A}_1$	$\overline{A}_2$	$\overline{O}$
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	**L**	L	L	L	L	H
L	X	X	X	X	X	X	**L**	H	L	H	L	L	H
L	X	X	X	X	X	**L**	H	H	L	L	H	L	H
L	X	X	X	X	**L**	H	H	H	L	H	H	L	H
L	X	X	X	**L**	H	H	H	H	L	L	L	H	H
L	X	X	**L**	H	H	H	H	H	L	H	L	H	H
L	X	**L**	H	H	H	H	H	H	L	L	H	H	H
L	**L**	H	H	H	H	H	H	H	L	H	H	H	H

Logic symbol and truth table for 74LS148 8–line–to–3–line priority encoder.

The Octal–to–Binary Encoder

- $\overline{EI}$ Active–*LOW* enable input, a *HIGH* on the input forces all outputs to their inactive state (*HIGH*).

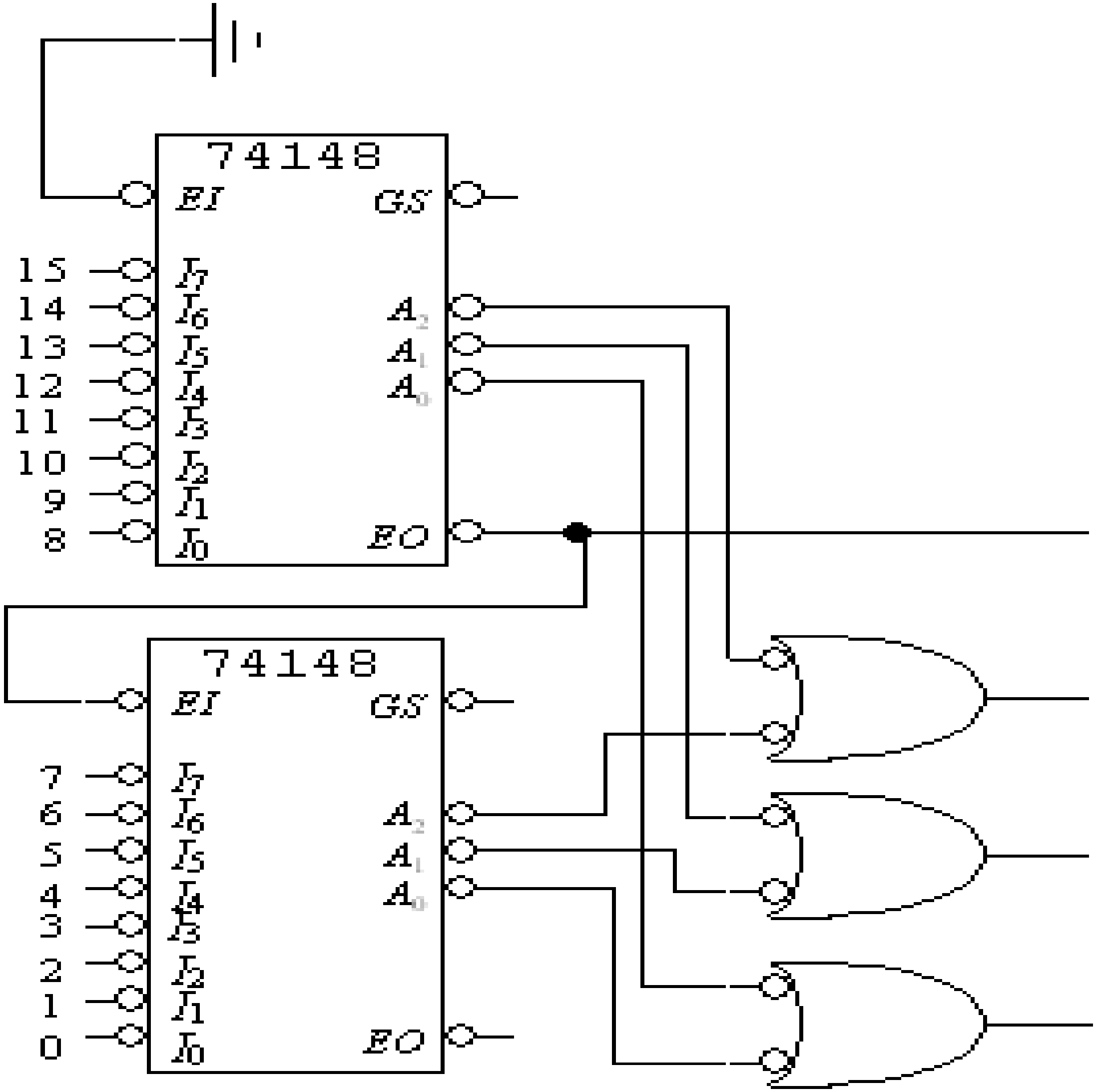
- $\overline{EO}$ Active–*LOW* enable output, the output pin goes *LOW* when all inputs are inactive (*HIGH*) and is *LOW*.

- $\overline{GS}$ Active–*LOW* group signal output, this output pin goes *LOW* whenever any of the inputs are active (*LOW*) and is *LOW*.

The 16 –to–4 Encoder

The 74LS148 can be expanded to a 16–line–to–4–line encoder by connecting the *EO* of the higher–order encoder to the *EI* of the lower–order encoder and negative–ORing the corresponding binary outputs as shown

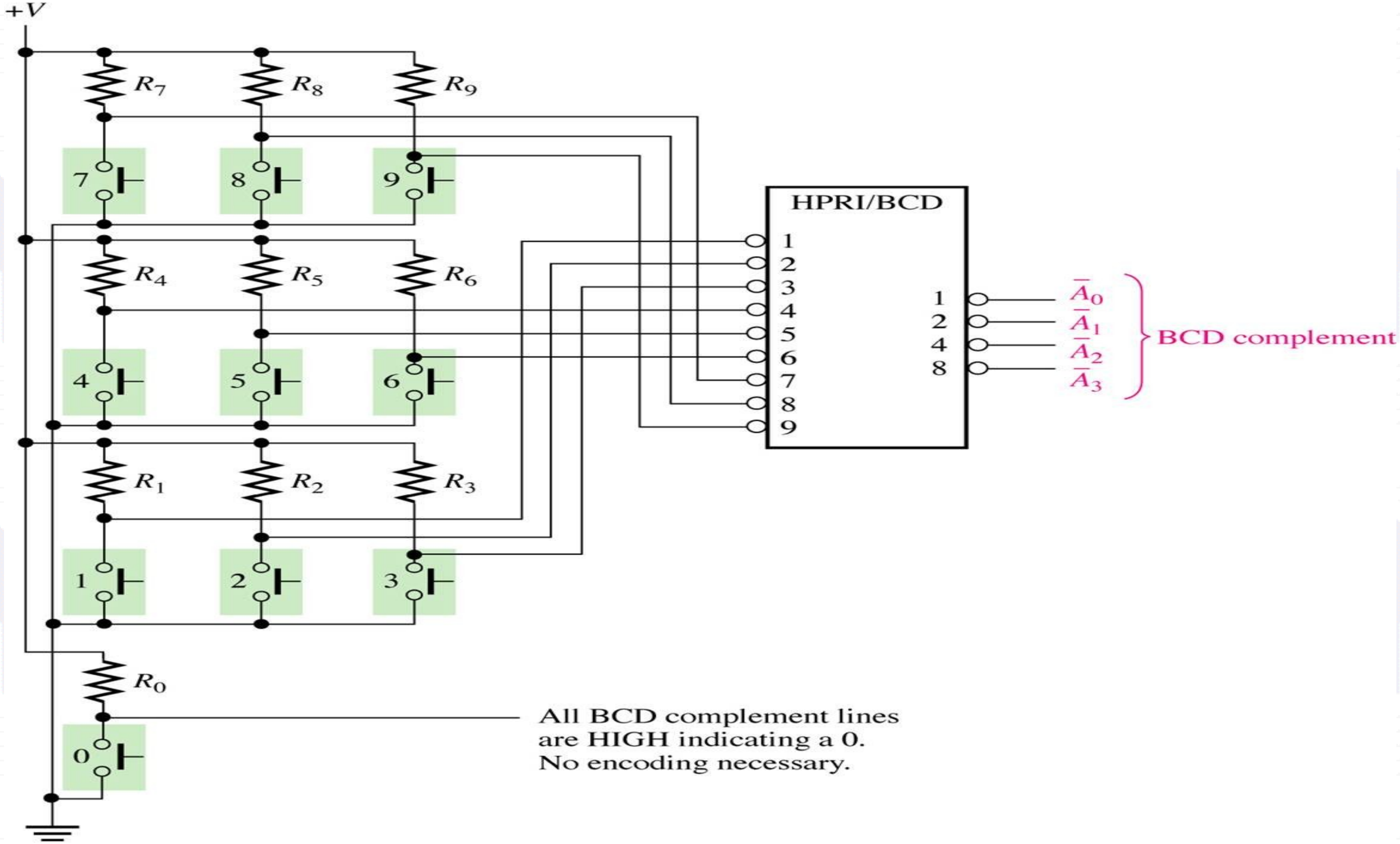
The 16 –to–4 Encoder



A 16–line–to–4–line encoder using 74LS148s and external logic.

Application example

A simplified keyboard encoder.



All BCD complement lines are HIGH indicating a 0. No encoding necessary.

•When one of the keys is pressed, the decimal digit is encoded to the corresponding BCD code

•The keys are represented by 10 push-button switches, each with a **pull-up resistor** to V+. The pull-up resistor ensures that the line is HIGH when a key is not depressed.

•When a key is depressed, the line is connected to ground, and a LOW is applied to the corresponding encoder input.

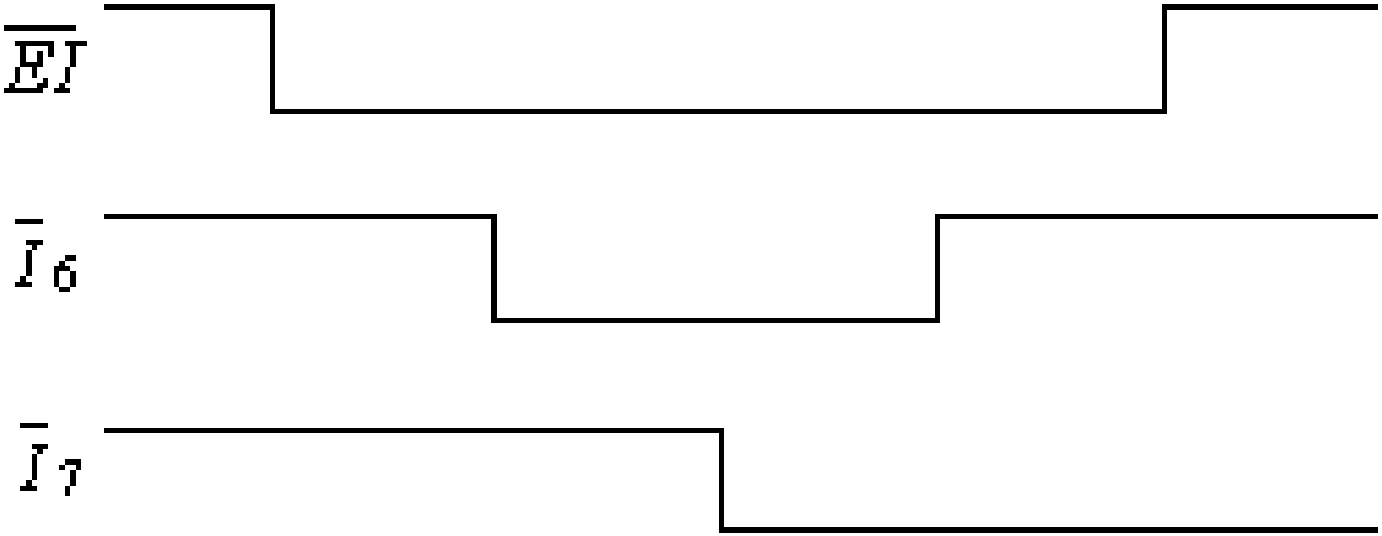
•The zero key is not connected because the BCD output represents zero when none of the other keys is depressed

•The BCD complement output of the encoder goes into a storage device, and each successive BCD code is stored until the entire number has been entered

Exercise

Sketch the output waveforms of the 74LS148 encoder based on the given waveforms.

Assume that $\bar{I}_0 = \bar{I}_1 = \bar{I}_2 = \bar{I}_3 = \bar{I}_4 = \bar{I}_5 = 1$


$\overline{EI}$
$\bar{I}_6$
$\bar{I}_7$