EIE2050
Digital Logic and Systems

Professor Qijun Zhang

Office: Room 404, Research A Building

Functions of Combinational Logic

- Basic adders
- Parallel adders
- Comparators
- Decoders
- Encoders
- Code converters

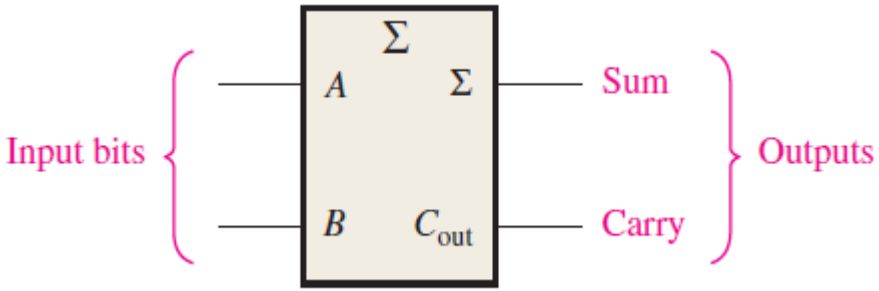
Reading material: Chapter 6 of Textbook:
Textbook: *Digital Fundamentals (global edition, 11th edition)*, by Thomas Floyd, Pearson 2015.
The examples used in the lecture are based on the textbook.

Functions of Combinational Logic

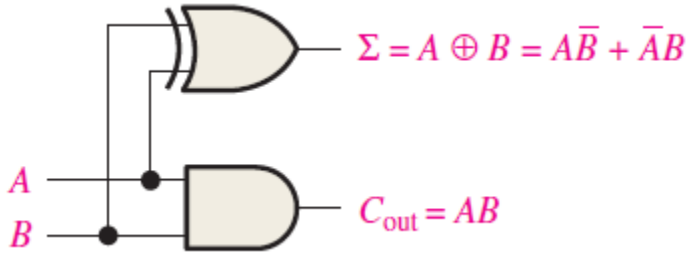
- Basic Adders
    - Half Adder
    - Full Adder

Functions of Combinational Logic

- Basic Adders
  - Half Adder



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

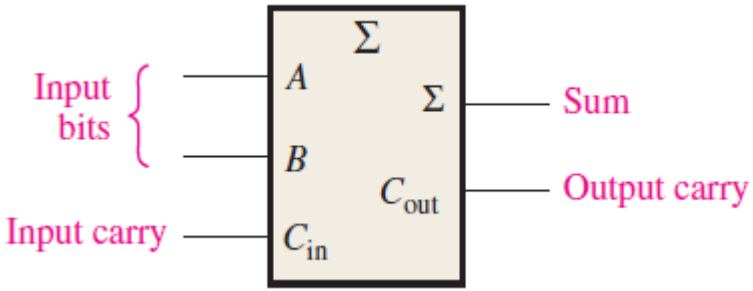


$$C_{out} = AB$$

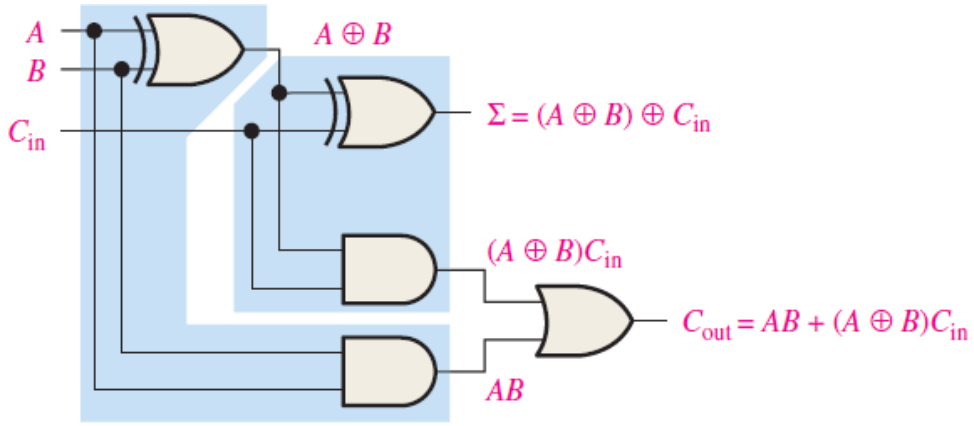
$$\Sigma = A \oplus B$$

Functions of Combinational Logic

- Basic Adders
  - 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

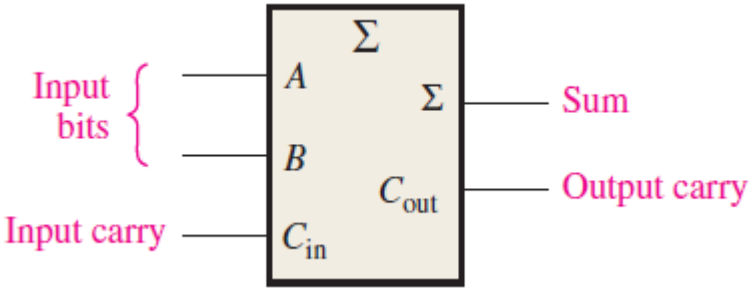


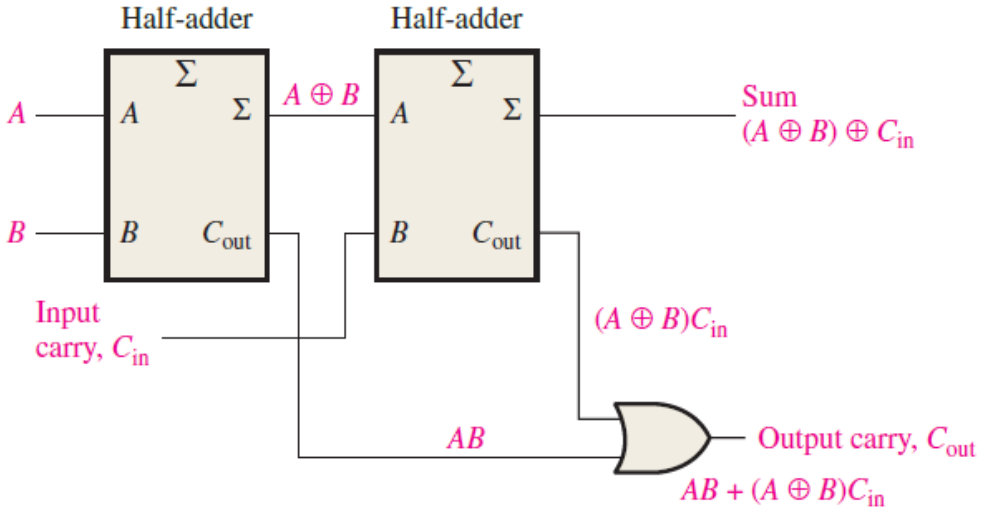
$$C_{out} = AB + (A \oplus B)C_{in}$$

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

Functions of Combinational Logic

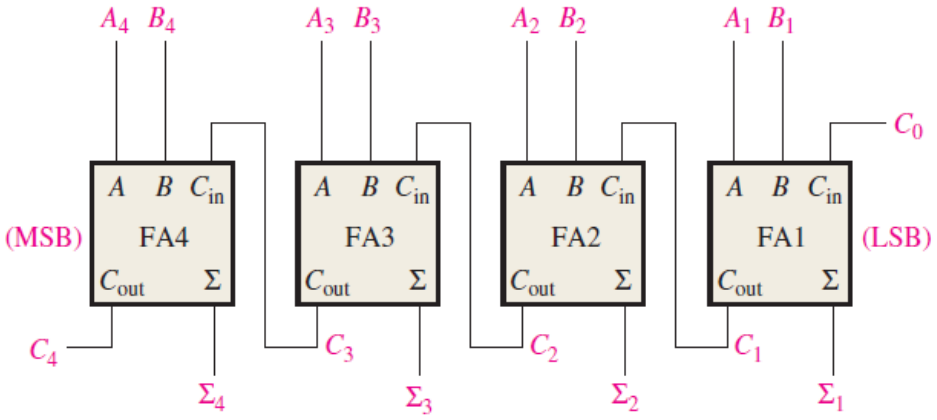
- Basic Adders
  - Full Adder: constructed using 2 half adders:



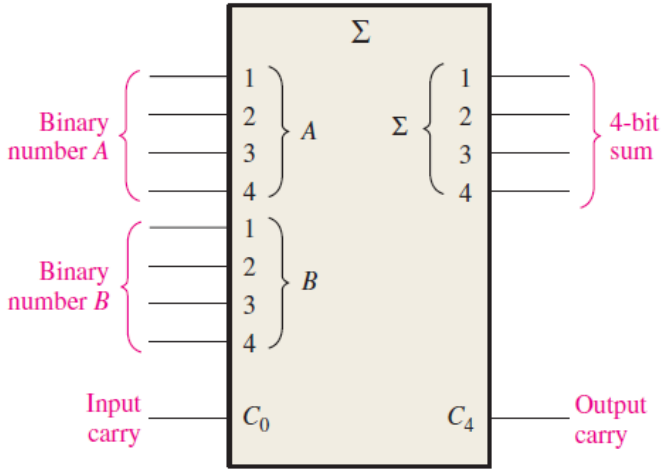


Functions of Combinational Logic

- Parallel Adders:



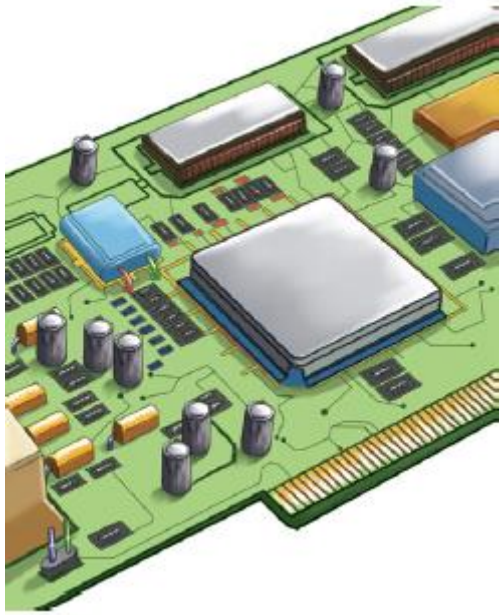
A 4-bit parallel adder

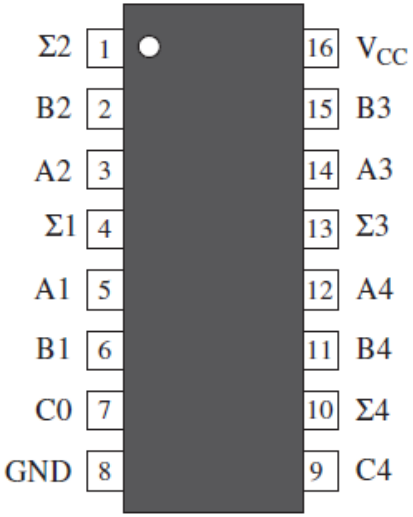


logic symbol

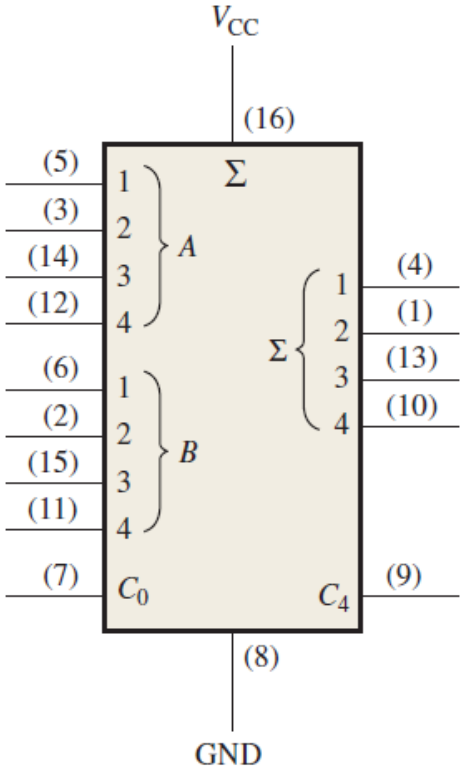
Functions of Combinational Logic

- 4-bit Parallel Adders: 74HC283, 74LS283





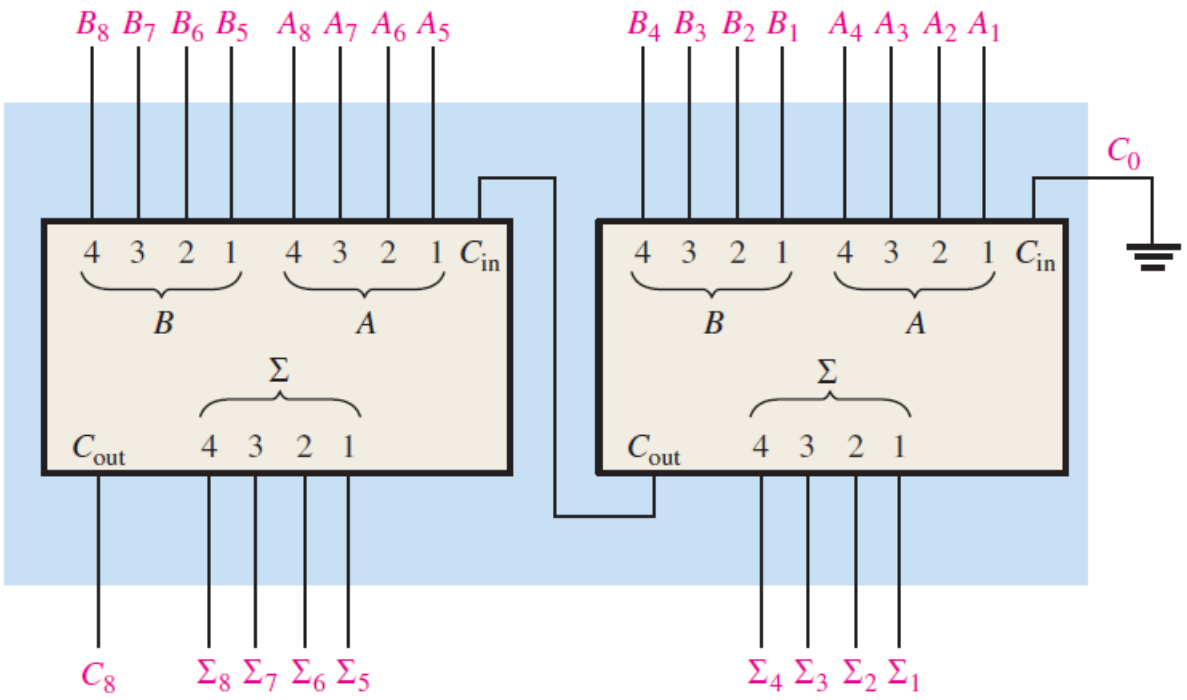
Pin diagram



logic symbol

Functions of Combinational Logic

- Adder expansion:

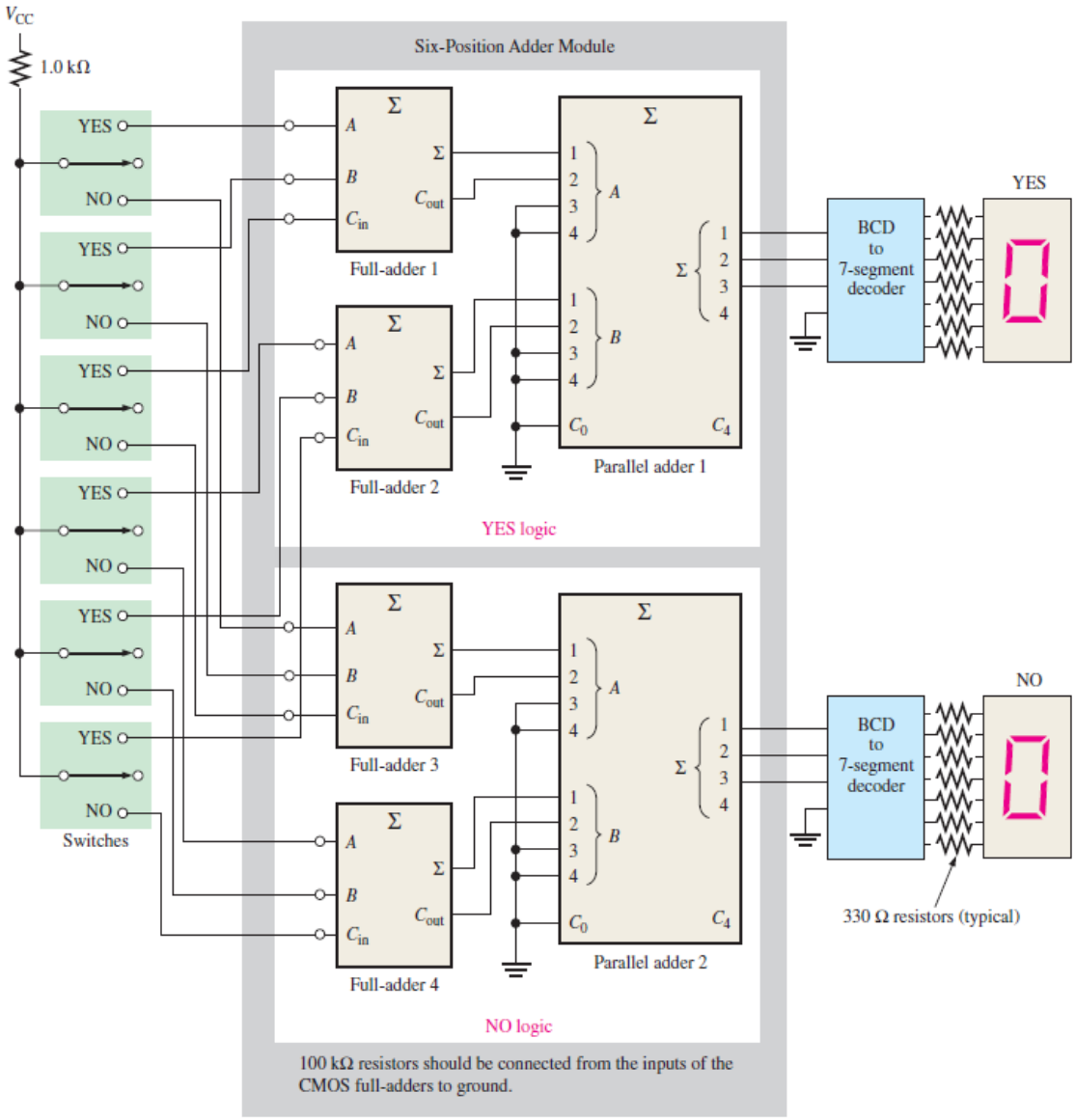


Cascading two 4-bit adders into a 8-bit adder

Functions of Combinational Logic

- Application Example:

  a voting system using adders

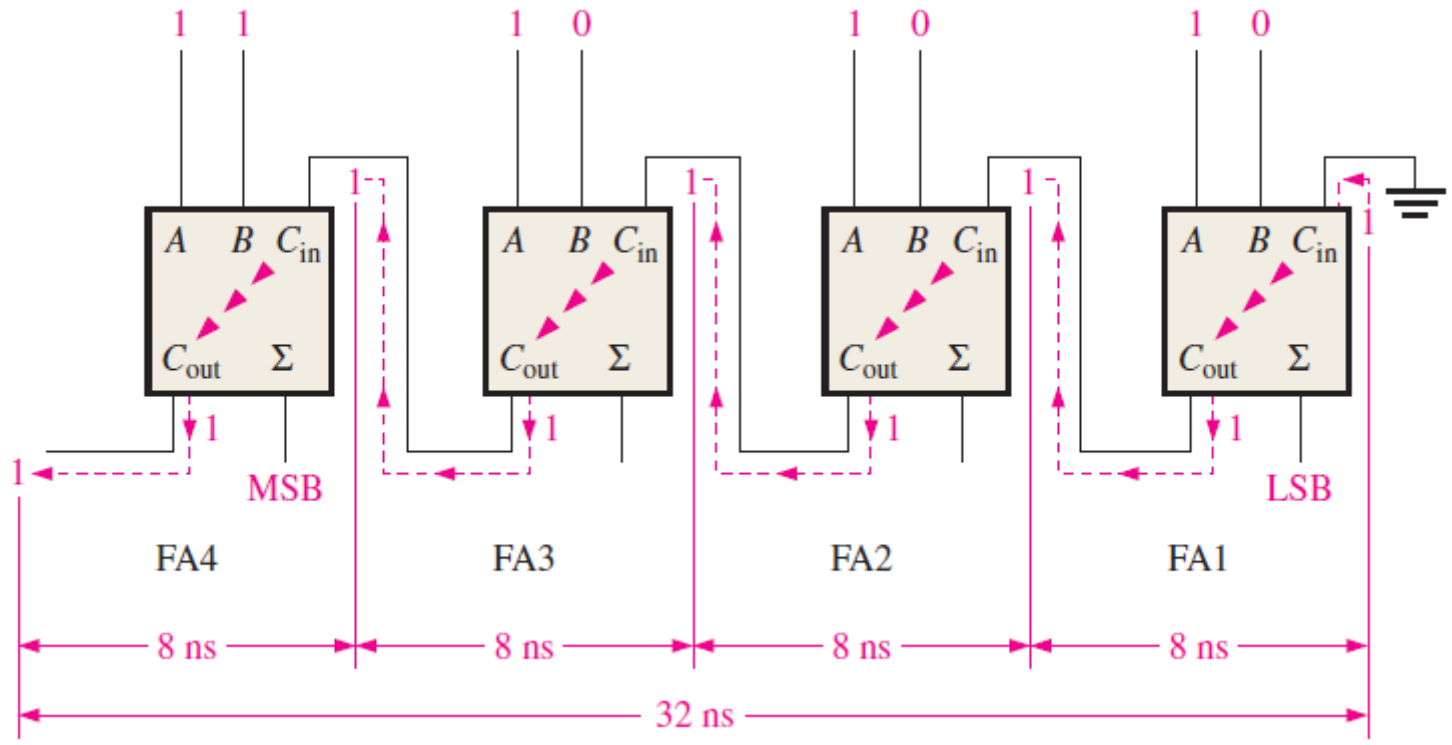


Functions of Combinational Logic

- Parallel Adders:
  - Ripple carry
  - Look-ahead carry
  - Combination of ripple carry and look-ahead carry adder

Functions of Combinational Logic

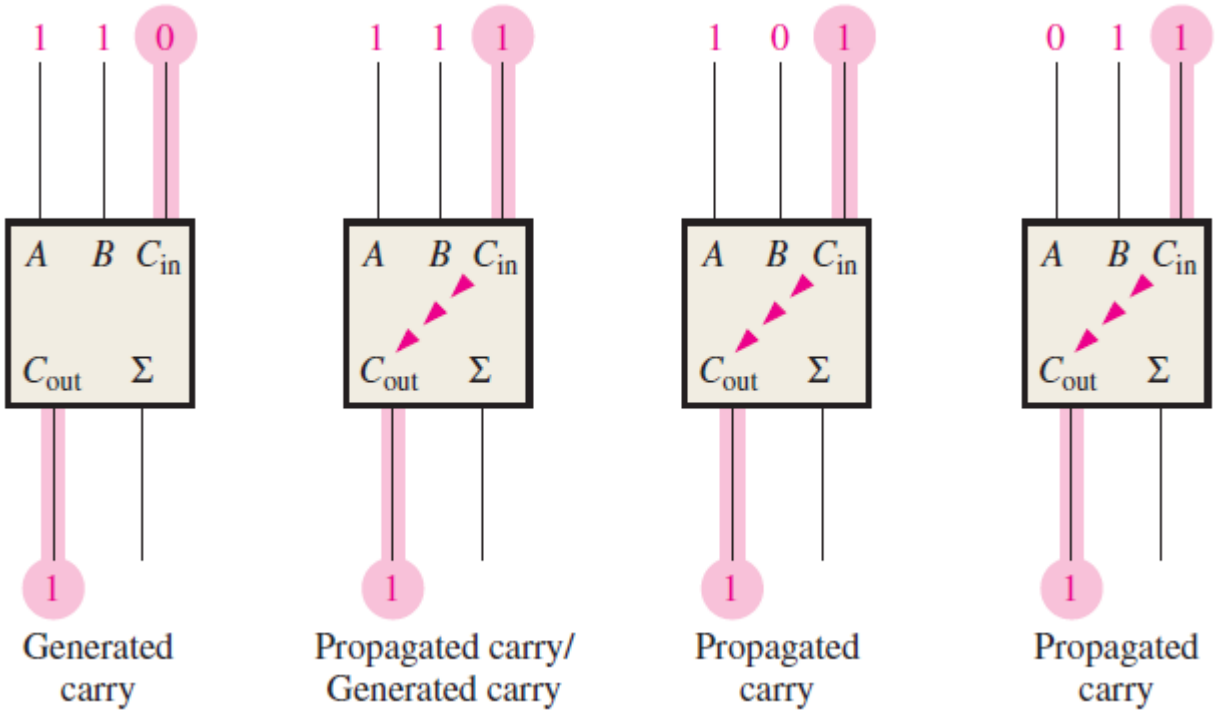
- Parallel Adders: Ripple Carry:



a 4-bit parallel ripple carry adder showing worst-case carry propagation

Functions of Combinational Logic

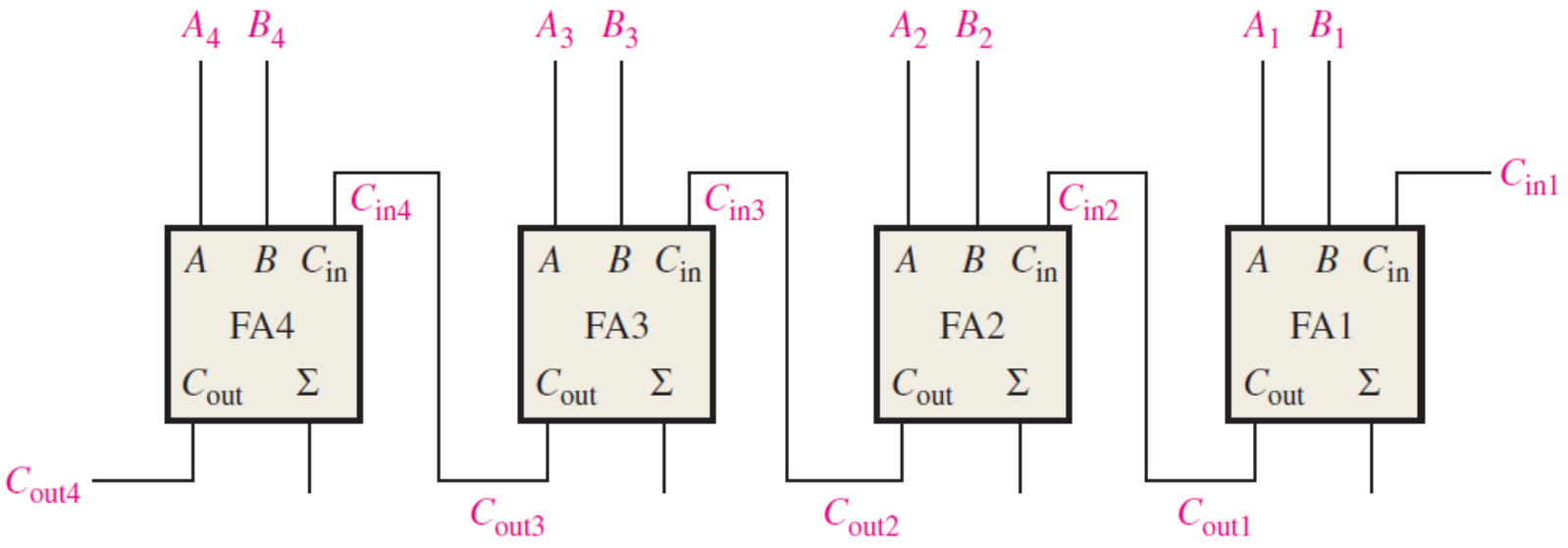
- Parallel Adders - Look-Ahead Carry:



C_{in} - input carry
C_p - propagated carry
C_g - generated carry

Functions of Combinational Logic

- Parallel Adders - Look-Ahead Carry:



Full-adder 4

$C_{g4} = A_4B_4$
$C_{p4} = A_4 + B_4$

Full-adder 3

$C_{g3} = A_3B_3$
$C_{p3} = A_3 + B_3$

Full-adder 2

$C_{g2} = A_2B_2$
$C_{p2} = A_2 + B_2$

Full-adder 1

$C_{g1} = A_1B_1$
$C_{p1} = A_1 + B_1$

Functions of Combinational Logic

- Parallel Adders - Look-Ahead Carry:

Full-adder 1:

$$C_{out1} = C_{g1} + C_{p1}C_{in1}$$

Full-adder 2:

$$C_{in2} = C_{out1}$$
$$C_{out2} = C_{g2} + C_{p2}C_{in2} = C_{g2} + C_{p2}C_{out1} = C_{g2} + C_{p2}(C_{g1} + C_{p1}C_{in1})$$
$$= C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{in1}$$

Full-adder 3:

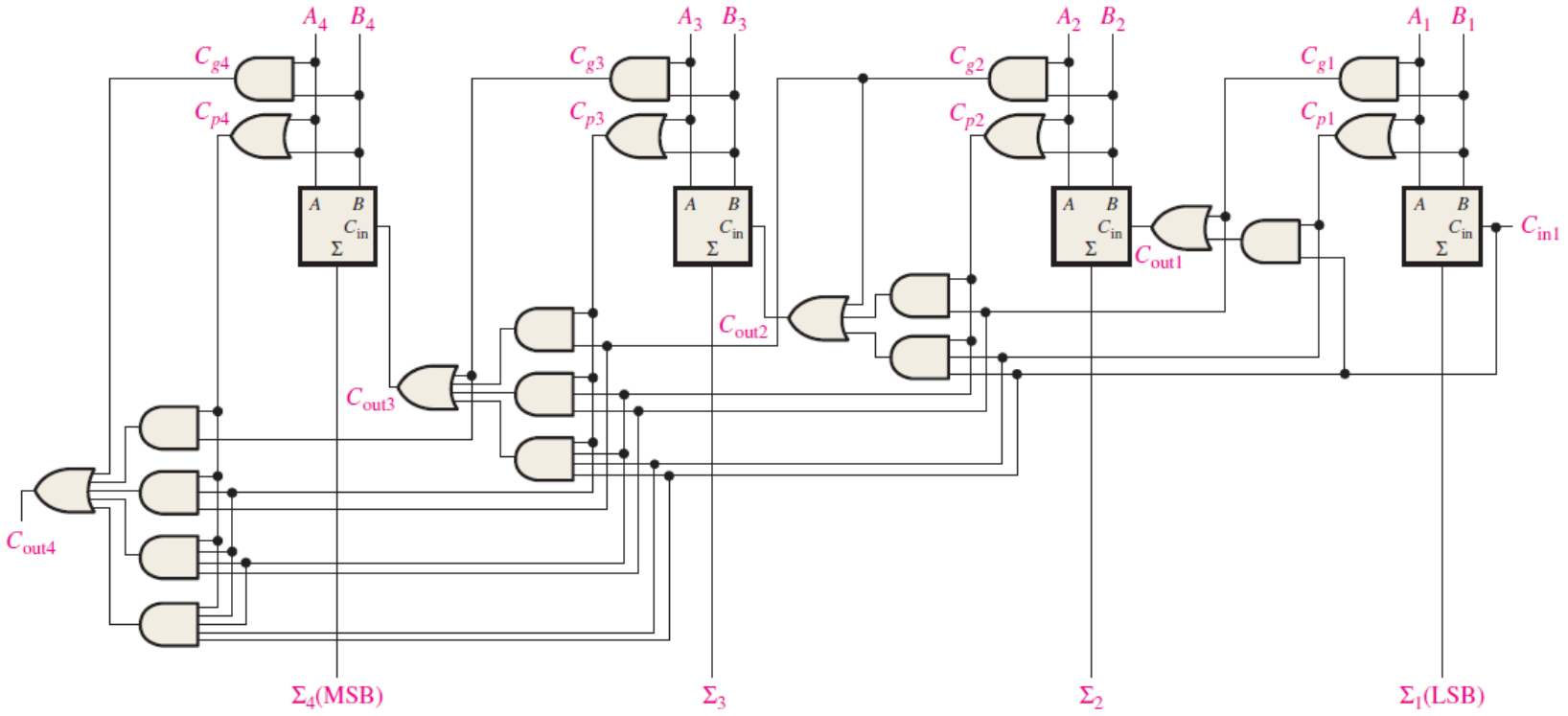
$$C_{in3} = C_{out2}$$
$$C_{out3} = C_{g3} + C_{p3}C_{in3} = C_{g3} + C_{p3}C_{out2} = C_{g3} + C_{p3}(C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{in1})$$
$$= C_{g3} + C_{p3}C_{g2} + C_{p3}C_{p2}C_{g1} + C_{p3}C_{p2}C_{p1}C_{in1}$$

Full-adder 4:

$$C_{in4} = C_{out3}$$
$$C_{out4} = C_{g4} + C_{p4}C_{in4} = C_{g4} + C_{p4}C_{out3}$$
$$= C_{g4} + C_{p4}(C_{g3} + C_{p3}C_{g2} + C_{p3}C_{p2}C_{g1} + C_{p3}C_{p2}C_{p1}C_{in1})$$
$$= C_{g4} + C_{p4}C_{g3} + C_{p4}C_{p3}C_{g2} + C_{p4}C_{p3}C_{p2}C_{g1} + C_{p4}C_{p3}C_{p2}C_{p1}C_{in1}$$

Functions of Combinational Logic

- Parallel Adders - Look-Ahead Carry:



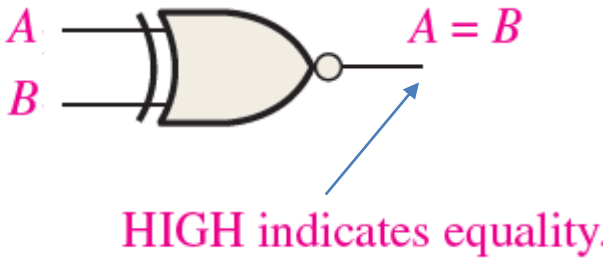
Logic diagram for a 4-stage look-ahead carry adder

Functions of Combinational Logic

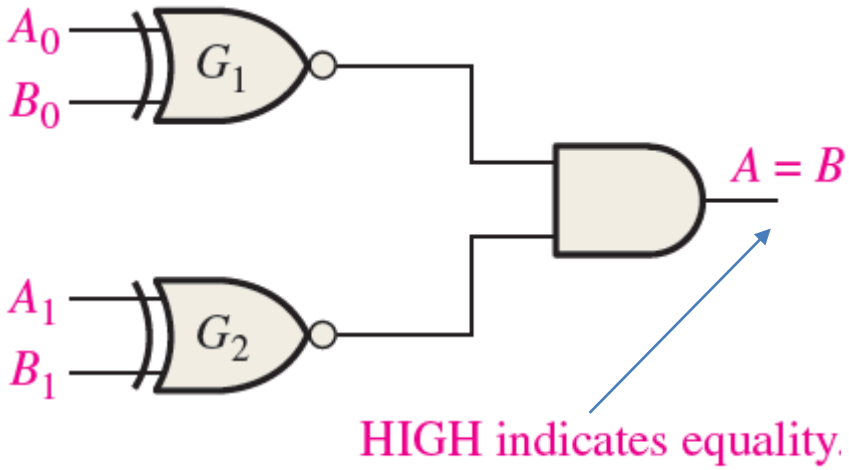
- Comparators:
  - Equality
  - Inequality

Functions of Combinational Logic

- Comparators - Equality:



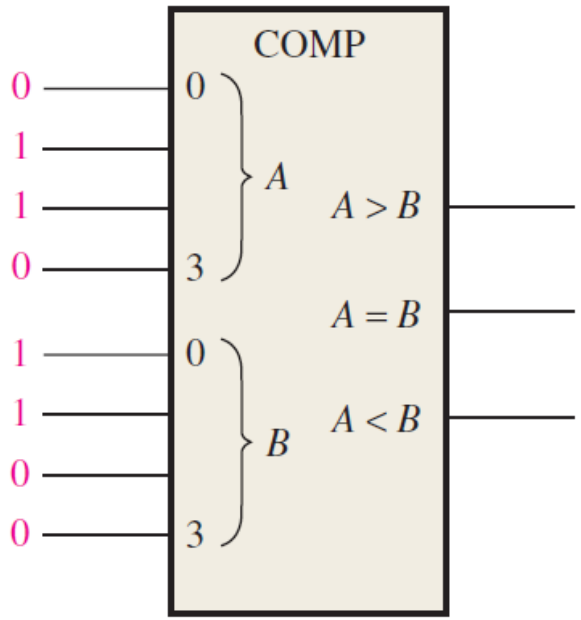
1-bit equality comparator



2-bit equality comparator

Functions of Combinational Logic

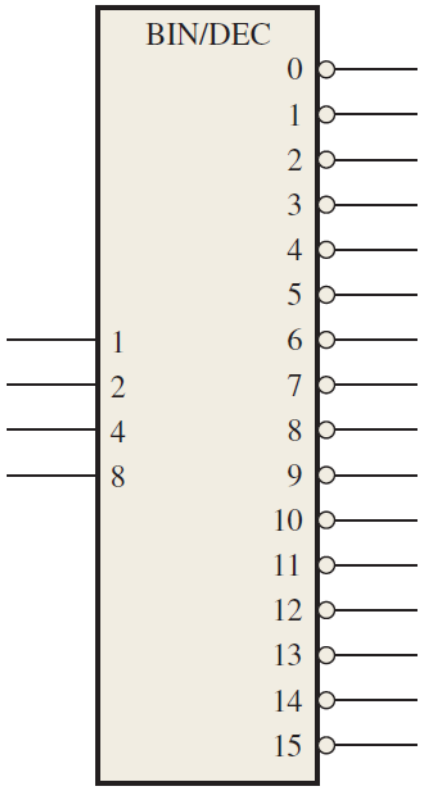
- Comparators - Inequality:



By comparing the highest (or the next highest..) bits of A and B, we can distinguish between A > B and A < B.

Functions of Combinational Logic

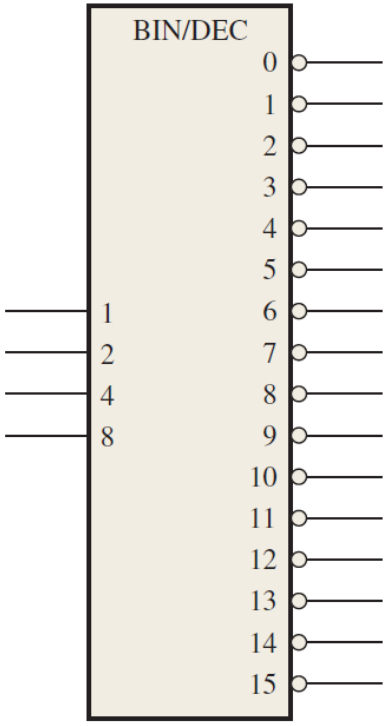
- Decoders



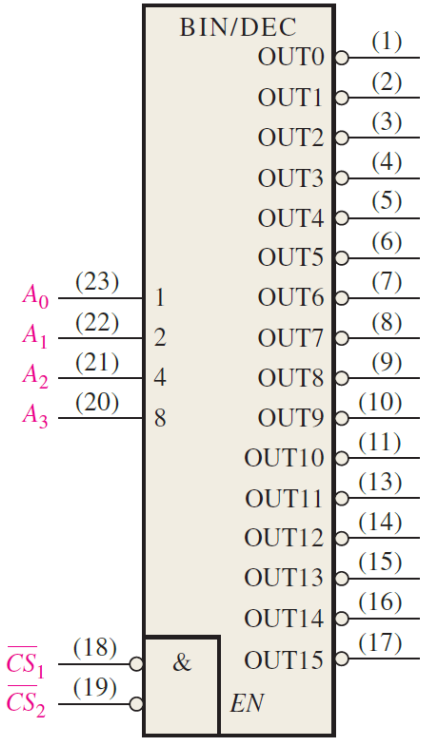
A 4-bit binary decoder

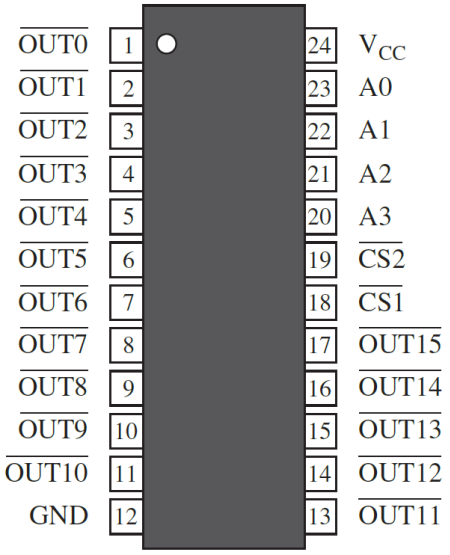
Functions of Combinational Logic

binary decoder



74HC154 binary decoder





CS_1: chip select
CS_2: chip select
EN: enable

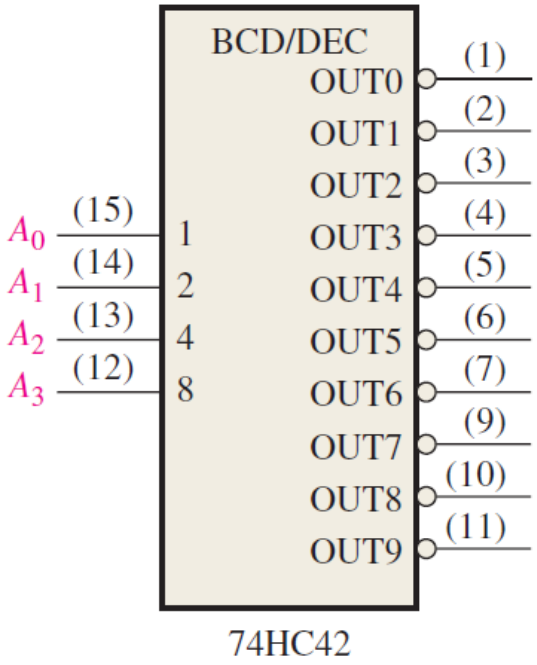
Example:

If $A_3A_2A_1A_0 = 0000$ and $\overline{CS}_1\overline{CS}_0 = 00$
then value at the output pins will be:

$\overline{OUT}_{15}\overline{OUT}_{14}..\overline{OUT}_1\overline{OUT}_0$ = 1111111111111110

Functions of Combinational Logic

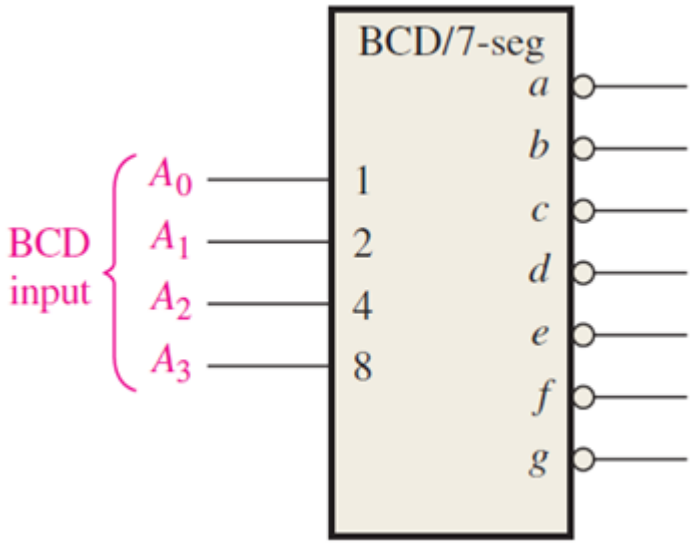
- Decoders

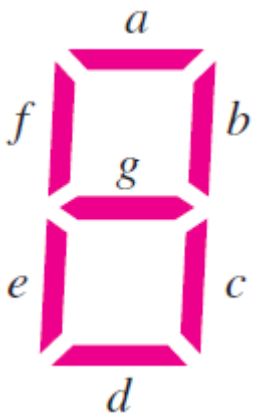


BCD to Decimal decoder
(shown here is a 74 series decoder)

Functions of Combinational Logic

- Decoders

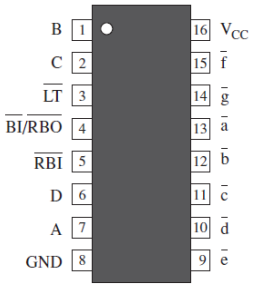


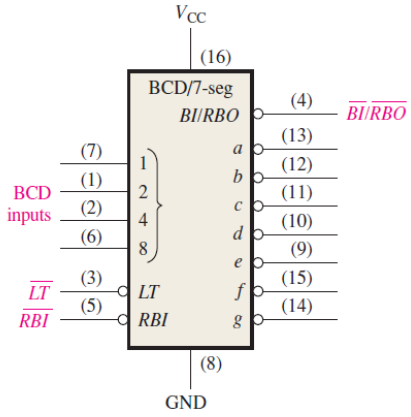


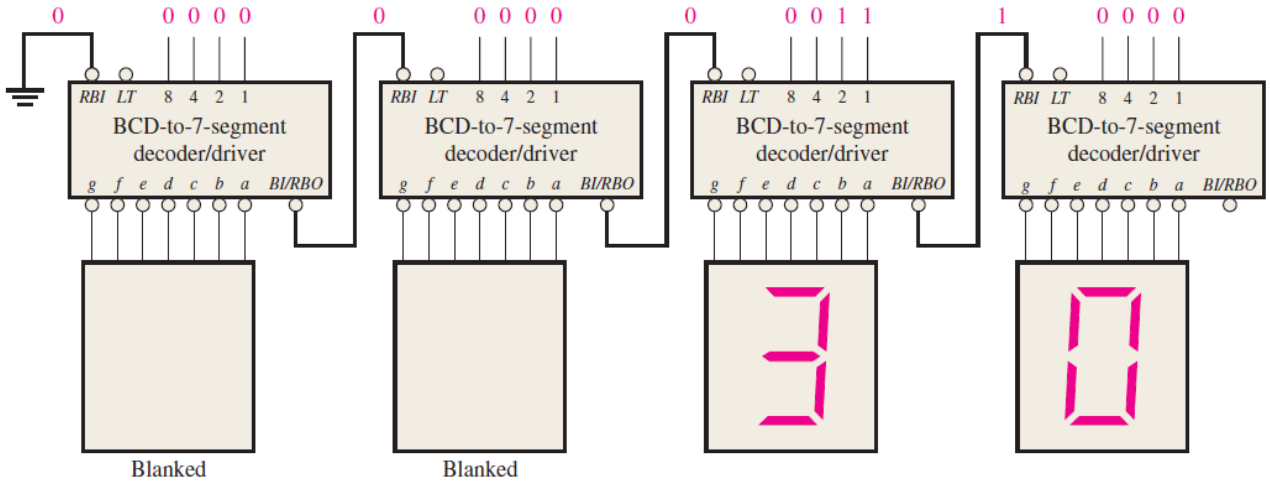
BCD to 7-segment decoder

Functions of Combinational Logic

- Decoders – BCD to 7 segment decoder (74HC47)



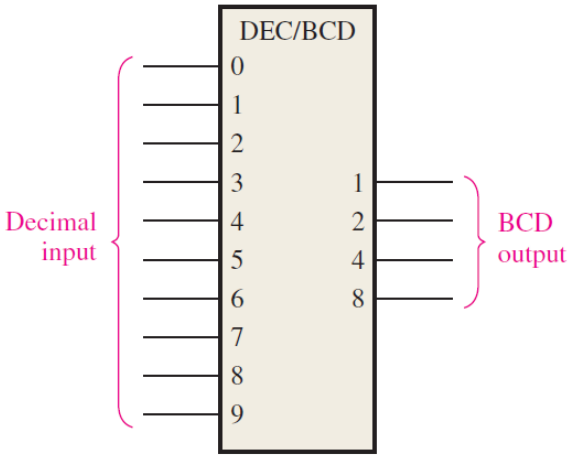




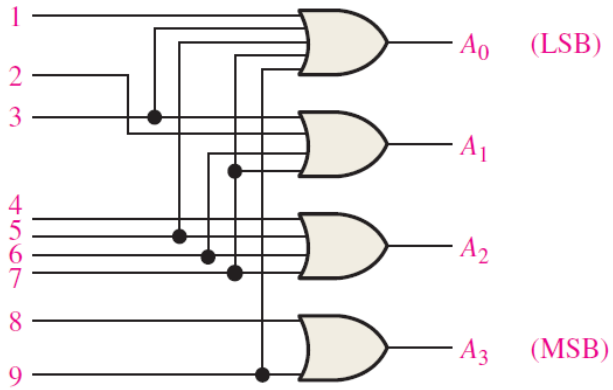
LT: lamp test
RBI: ripple blanking input (useful for zero suppression)
RBO: ripple blanking output (useful for zero suppression)

Functions of Combinational Logic

- Encoders: 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



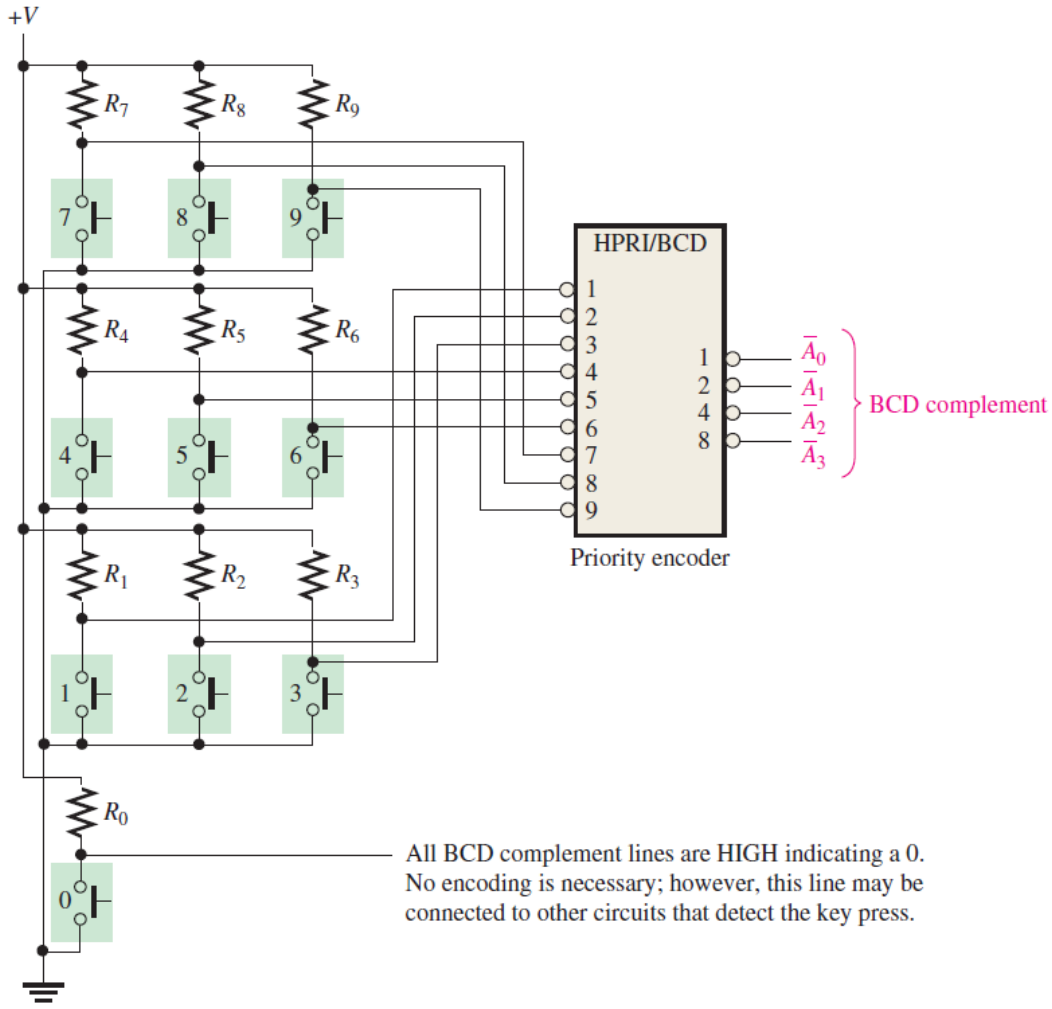
$$A_0 = 1 + 3 + 5 + 7 + 9$$

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

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

Functions of Combinational Logic

- Encoders: Decimal to BCD



Functions of Combinational Logic

- Code Converters: BCD to Binary

Example:

Decimal 87

BCD 1000 0111

Binary: 1010111

Binary representations of BCD bit weights.

BCD Bit	BCD Weight	(MSB) 64	32	16	8	4	2	(LSB) 1
		Binary Representation						
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

Convert the BCD numbers 00100111 (decimal 27) and 10011000 (decimal 98) to binary.

- **Solution**

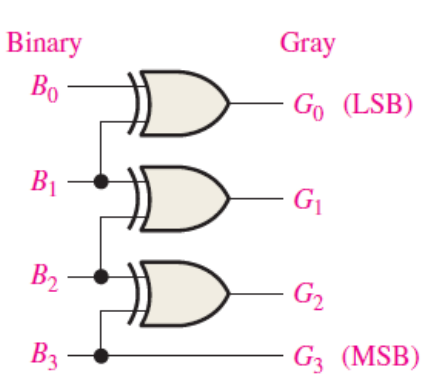
Write the binary representations of the weights of all 1s appearing in the numbers, and then add them together.

80	40	20	10	8	4	2	1		
0	0	1	0	0	1	1	1		
								0000001	1
								0000010	2
								0000100	4
								+ 0010100	20
								0011011	Binary number for decimal 27

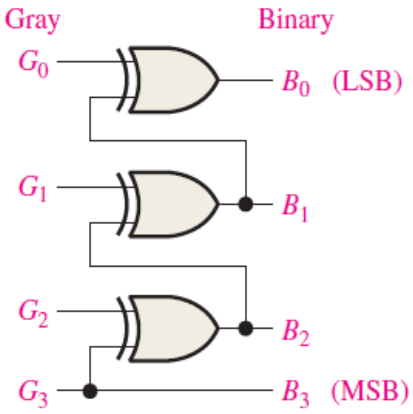
80	40	20	10	8	4	2	1		
1	0	0	1	1	0	0	0		
								0001000	8
								0001010	10
								+ 1010000	80
								1100010	Binary number for decimal 98

Functions of Combinational Logic

- Code Converters:



Binary to Gray



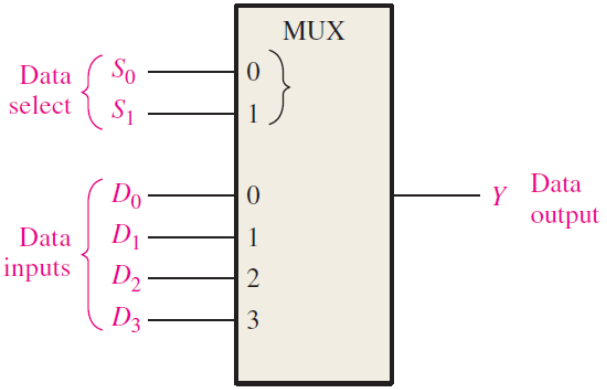
Gray to Binary

Four-bit Gray code.

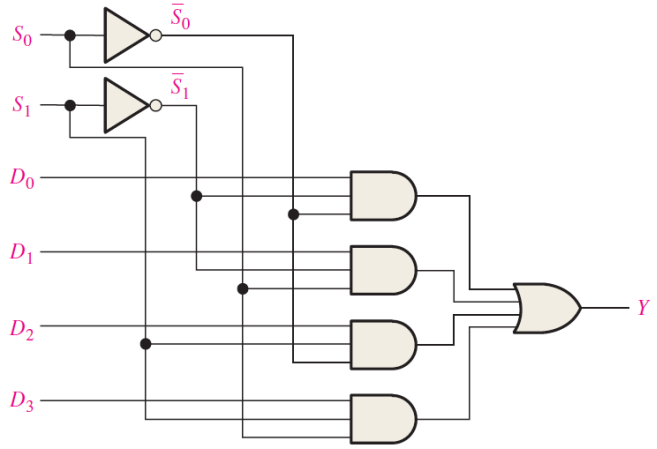
Decimal	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Functions of Combinational Logic

- Multiplexers (data selectors)



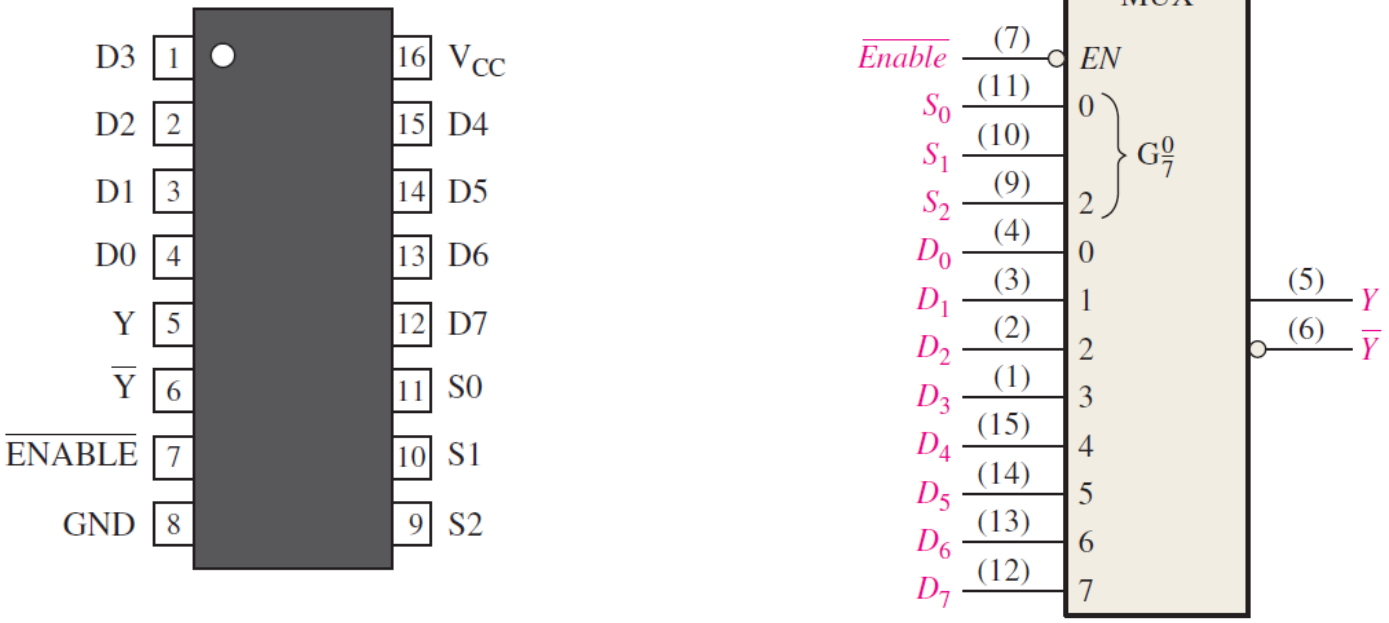
Data-Select Inputs		
S_1	S_0	Input Selected
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



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

Functions of Combinational Logic

- Multiplexers (data selectors)

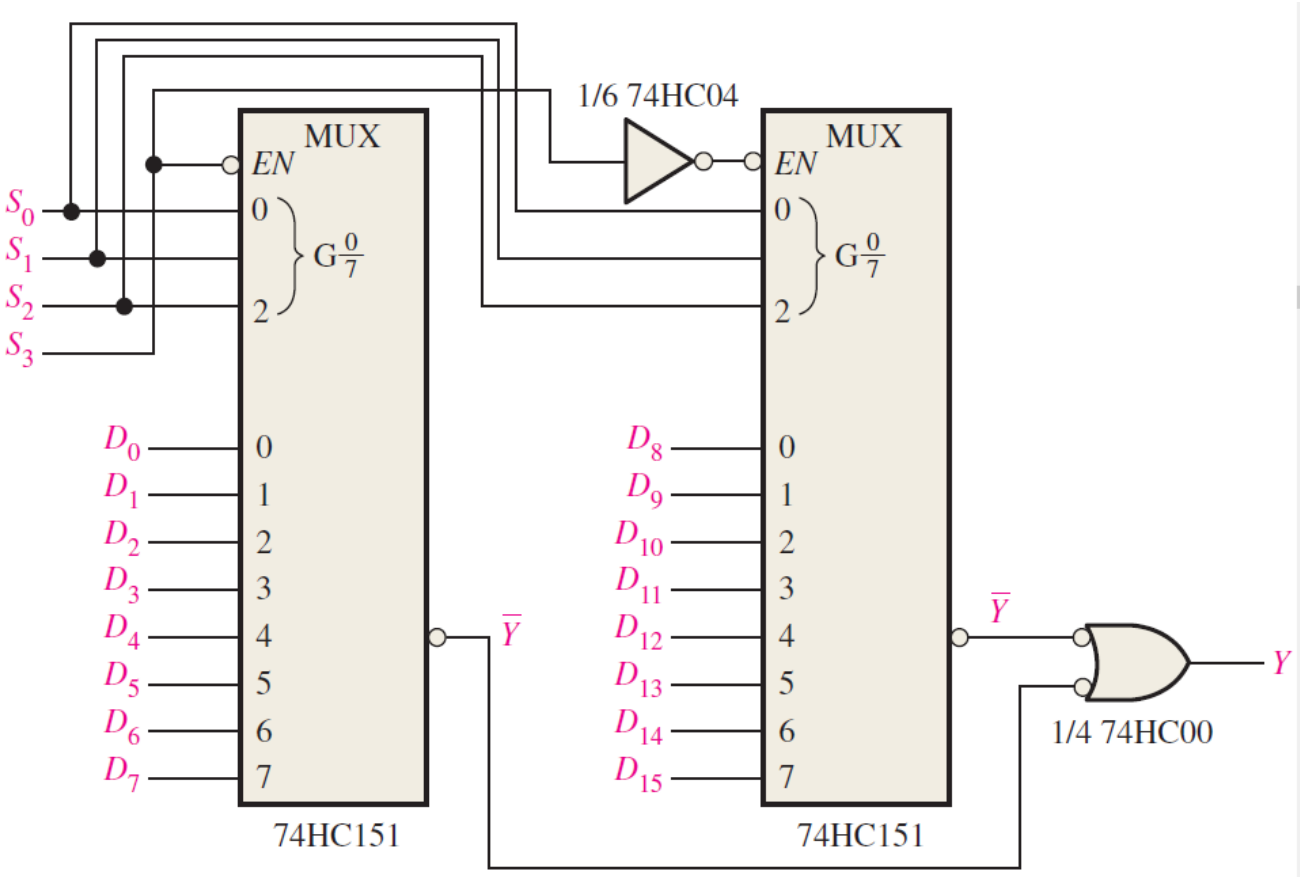


The 74HC151 eight-input data selector/multiplexer.

Functions of Combinational Logic

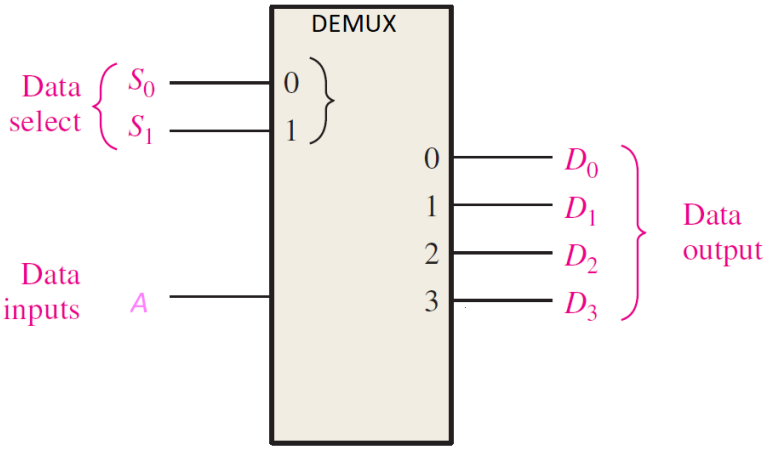
- Multiplexers (data selectors):

  a 16-input multiplexer

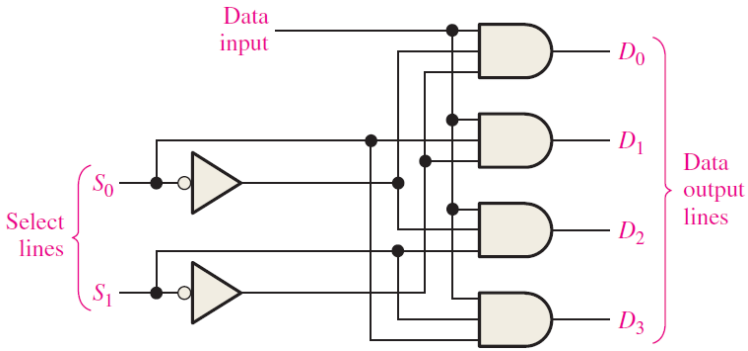


Functions of Combinational Logic

- Demultiplexers



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



$$D_0 = A\bar{S}_1\bar{S}_0$$

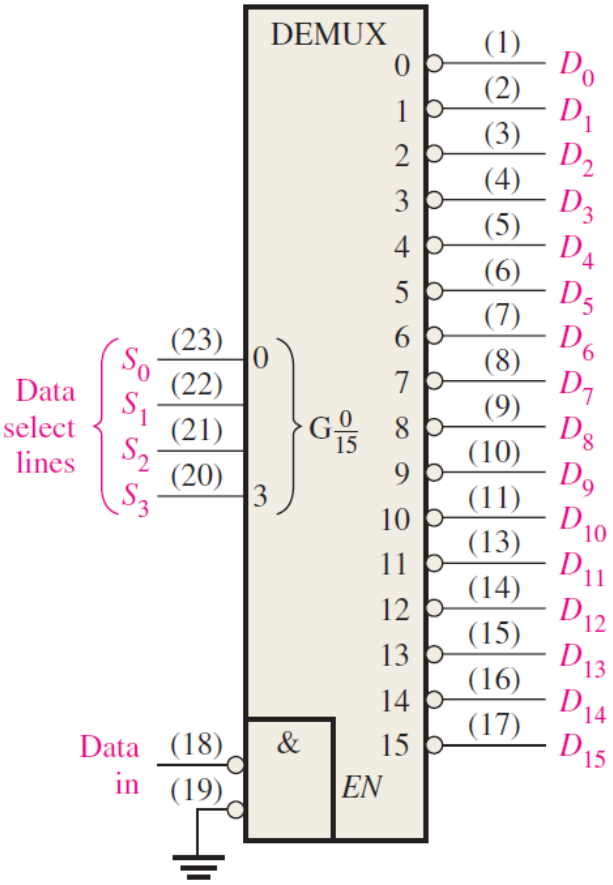
$$D_1 = A\bar{S}_1 S_0$$

$$D_2 = AS_1\bar{S}_0$$

$$D_3 = AS_1 S_0$$

Functions of Combinational Logic

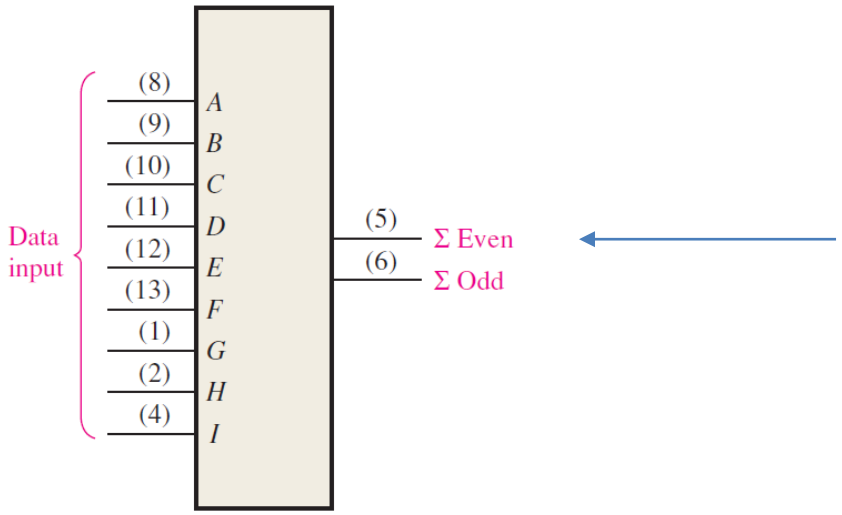
- Demultiplexers



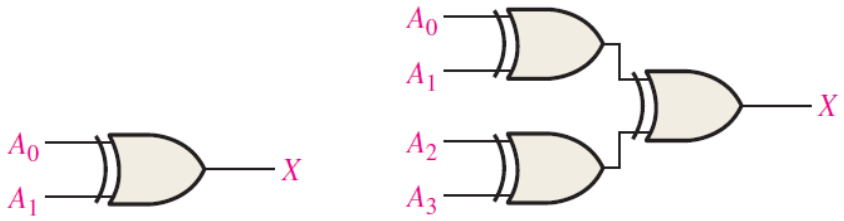
This decoder (74HC154) can be used as a demultiplexer

Functions of Combinational Logic

- Parity Generators/Checkers



This can be used as a parity checker, or as a parity generator. For example, when used as an even parity generator, the output Σ Odd is the parity bit to be appended to the input data



Summing of two bits

Summing of four bits

X will be high if the number of 1's is odd