

CHAPTER

8

# LOGICAL CIRCUIT DESIGN



KHOA CÔNG NGHỆ THÔNG TIN  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

[fit@hcmus](mailto:fit@hcmus)

# REMIND

☐ Boolean algebra

Read chapter 11 Computer Organization and Architecture  
10<sup>th</sup> edition, William Stallings



# PREREQUITES

☐ Install Logisims tool already

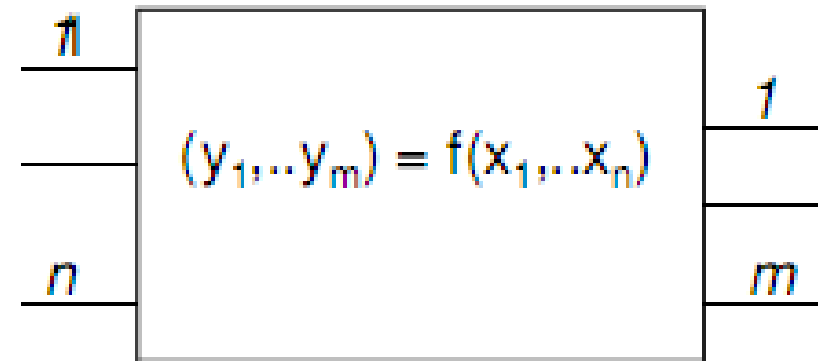


# What will you learn?

- ☐ Combinational circuit
  - ☐ How to design a combinational circuit?
  - ☐ Application of combinational circuit
- ☐ Basic ALU design
  - ☐ Sequential circuit
  - ☐ Application of sequential circuit

# Combinational circuit

- A combinational circuit is an interconnected set of gates whose *output at any time is a function only of the input at that time*
- Consists of  $n$  binary input,  $m$  binary output
- Can be defined in three ways:
  - ▣ True table
  - ▣ Graphic symbol
  - ▣ Boolean equation



# Implement of Boolean functions

- Sum of Product: The **SOP** form expresses that the output is 1 if **any of the input combinations that produce 1 is true**

$$f = u_1 + u_2 + \dots + u_n \quad \text{With } u_j = x_1 \cdot x_2 \dots x_i$$

- Product of Sum: The **POS** form expresses that the output is 1 if **all the input combinations that produce 1 is true**

$$f = u_1 \cdot u_2 \cdot \dots \cdot u_n \quad \text{With } u_j = x_1 + x_2 + \dots + x_i$$



# Sum of Product (SOP)

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

$$f = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}z$$

$\bar{x}\bar{y}z$

$\bar{x}y\bar{z}$

$x\bar{y}z$

# Product of Sum (POS)

x	y	z	$f = \bar{g}$	g
0	0	0	1	0
0	0	1	1	0
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	0

$$g = \bar{x} \cdot y \cdot \bar{z} + x\bar{y}\bar{z}$$

$$f = \bar{g} = (x + \bar{y} + z)(\bar{x} + y + z)$$



# Simplify the Boolean function

- Using simplification methods:
  - Algebraic Simplification
  - Karnaugh map



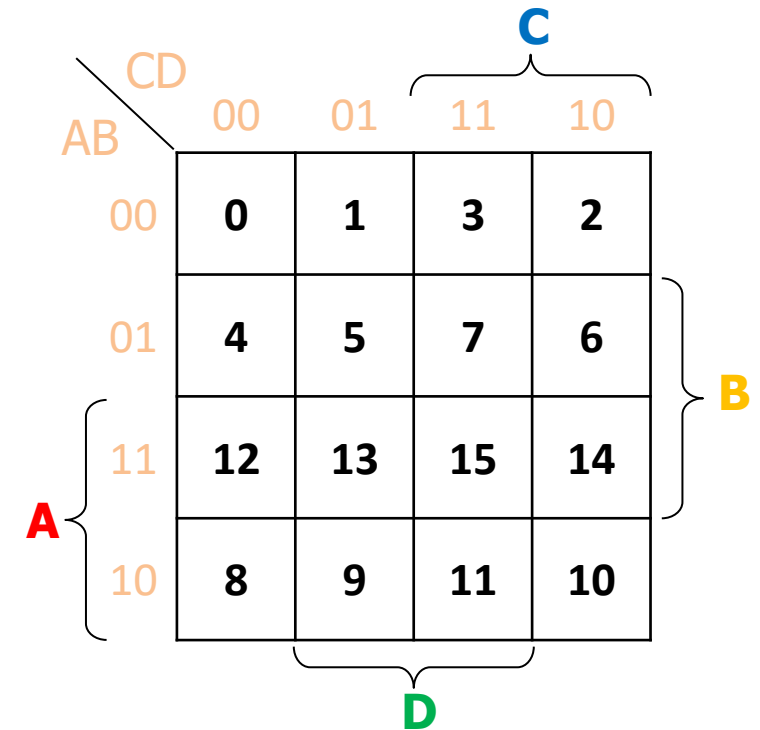
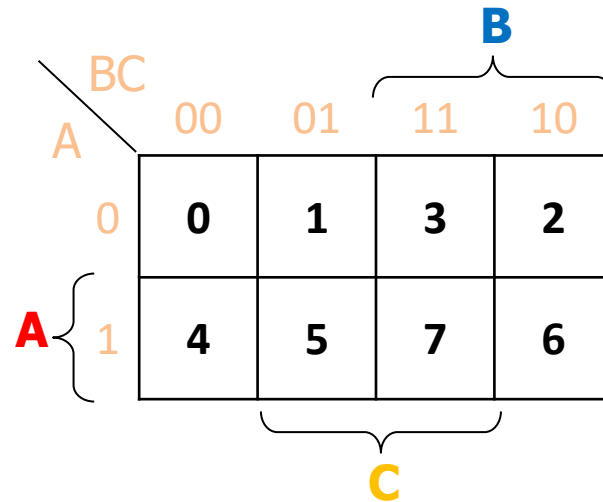
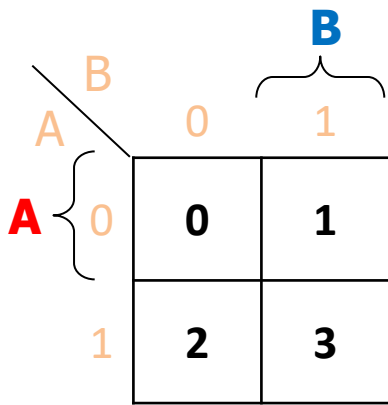
# Simplify the Boolean function

Refer to Basic Identities of Boolean Algebra

Basic Postulates		
$A \cdot B = B \cdot A$	$A + B = B + A$	Commutative Laws
$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A + B) \cdot (A + C)$	Distributive Laws
$1 \cdot A = A$	$0 + A = A$	Identity Elements
$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$	Inverse Elements
Other Identities		
$0 \cdot A = 0$	$1 + A = 1$	Associative Laws
$A \cdot A = A$	$A + A = A$	
$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$	DeMorgan's Theorem
$\overline{A \cdot B} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A} \cdot \bar{B}$	

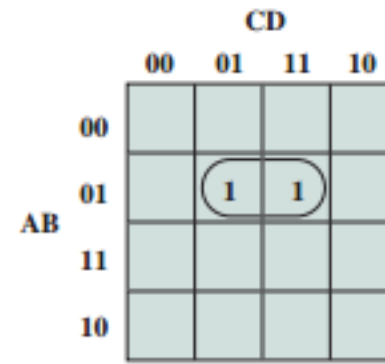
# Simplify the Boolean function

## □ Basic types of Karnaugh map

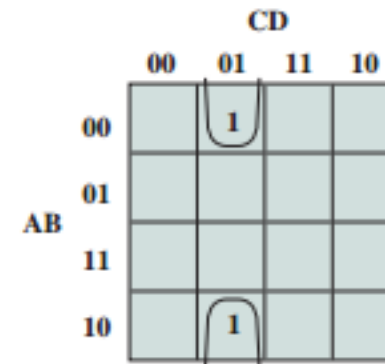


## Simplify the Boolean function

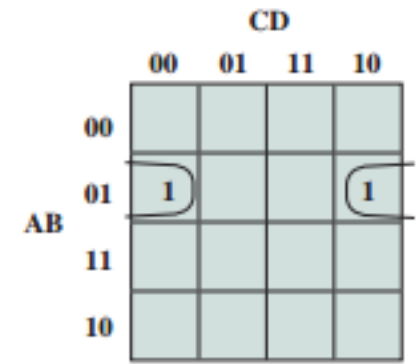
- Any two squares that are adjacent differ in only one of the variables
- The top square of a column is adjacent to the bottom square, and the leftmost square of a row is adjacent to the rightmost square
- 4 cells located in 4 corners of the map are also considered adjacent cells



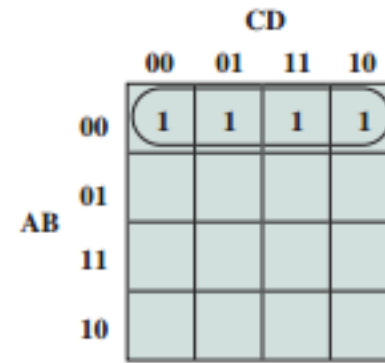
(a)  $\bar{A}BD$



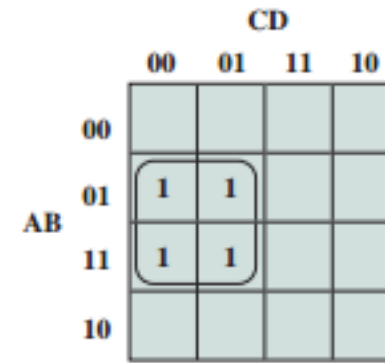
(b)  $\bar{B}CD$



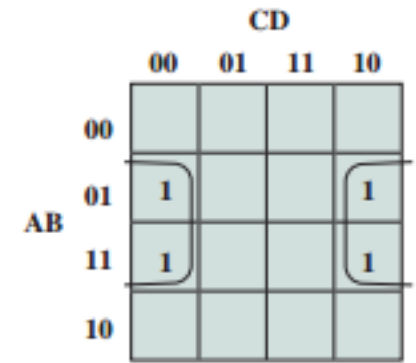
(b)  $\bar{A}\bar{B}D$



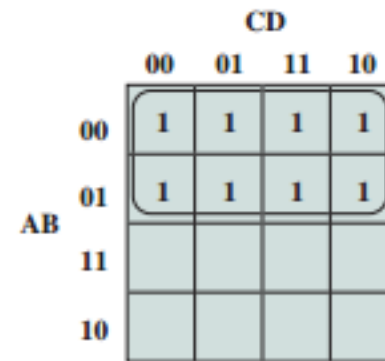
(d)  $\bar{A}\bar{B}$



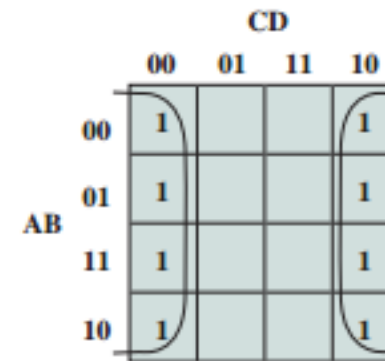
(e)  $\bar{B}\bar{C}$



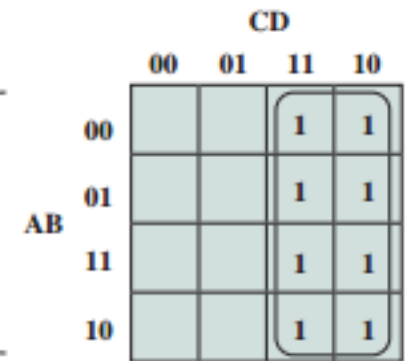
(f)  $\bar{B}\bar{D}$



(g)  $\bar{A}$



(h)  $\bar{D}$



(i)  $C$

# Design a combinational circuit

3 steps:

- ☐ Step 1: Construct the true table
- ☐ Step 2: Identify the Boolean function
- ☐ Step 3: Draw the logical circuit and test



# Design a combinational circuit

Example:

Design a 3-input, 1-output combination circuit, so that the logic value of the output is the majority of the inputs.



# Design a combinational circuit

Step 1: The true table

$$f(x, y, z) = \sum (3, 5, 6, 7)$$

x	y	z	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

# Design a combinational circuit

$$f(x, y, z) = \sum (3, 5, 6, 7)$$

Step 2: Boolean Algebra function

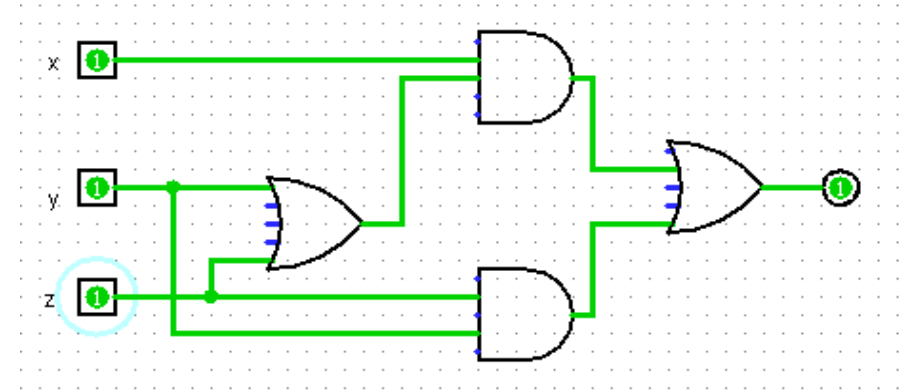
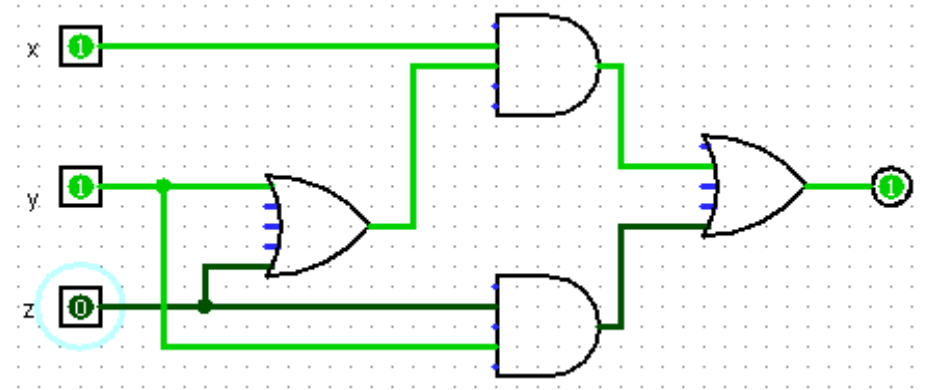
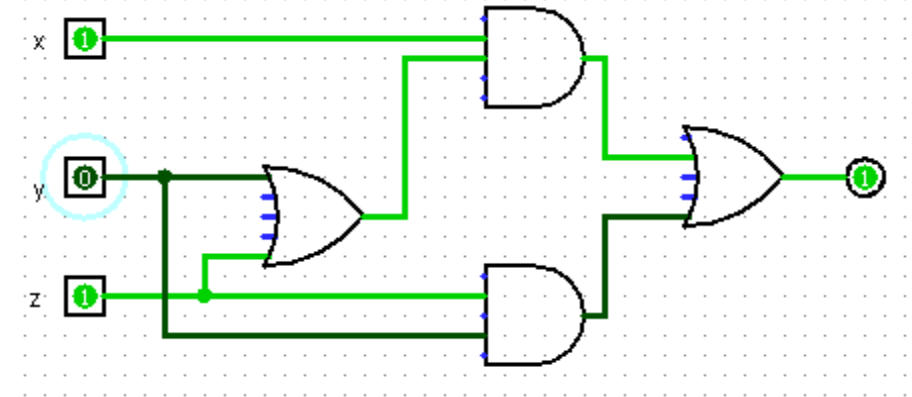
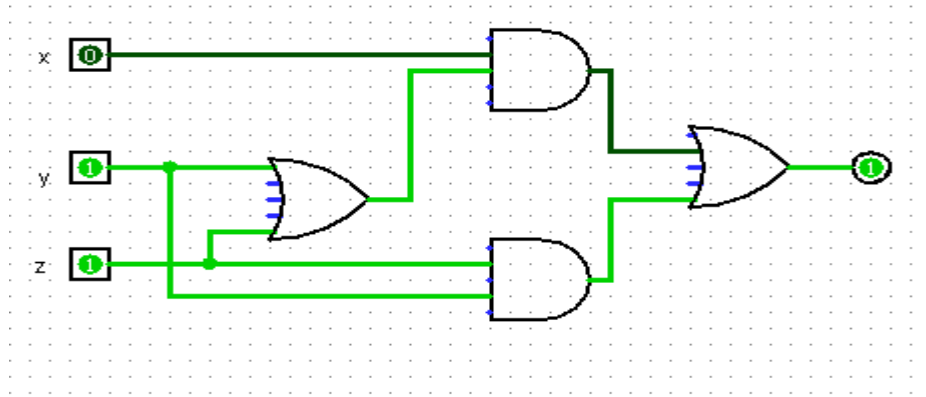
$$f = xz + xy + yz = x.(y + z) + yz$$

	00	01	11	10
0			1	
1		1	1	1



# Design a combinational circuit

## □ Step 3: Drawing the logic circuit and testing

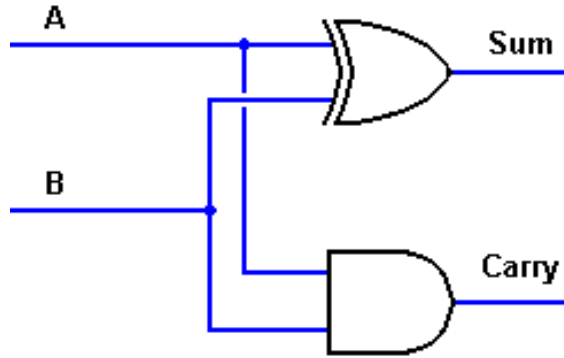


# Application of combinational circuit

- ☐ Adder/ Subtractor
- ☐ Encode/ Decode
- ☐ Multiplexer/ Demultiplexer
- ☐ ALU
- ☐ ...

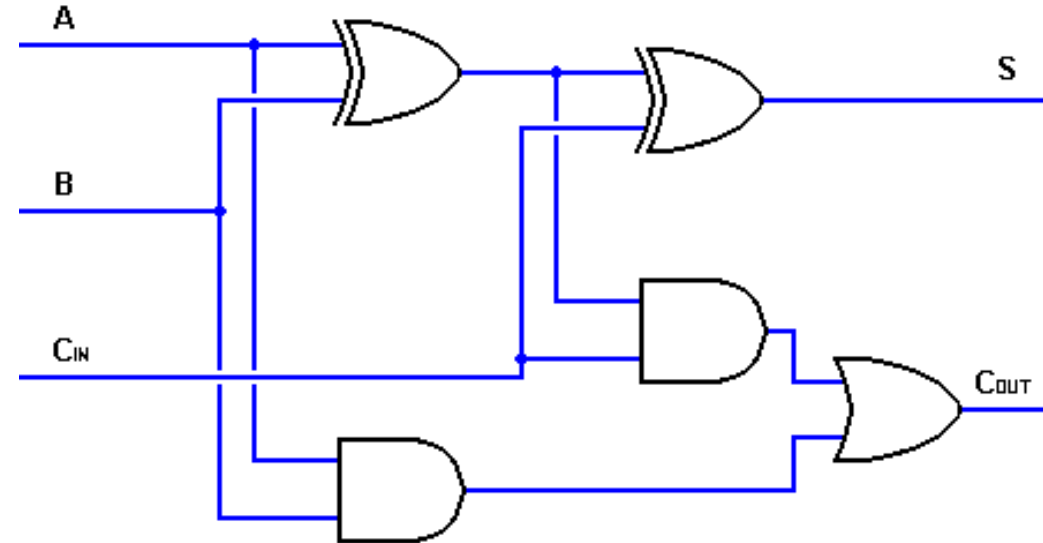


# 1-bit Adder



$$S = F(A, B) = \Sigma(1, 2)$$

$$C_{\text{carr}} = F(A, B) = \Sigma(0, 3)$$

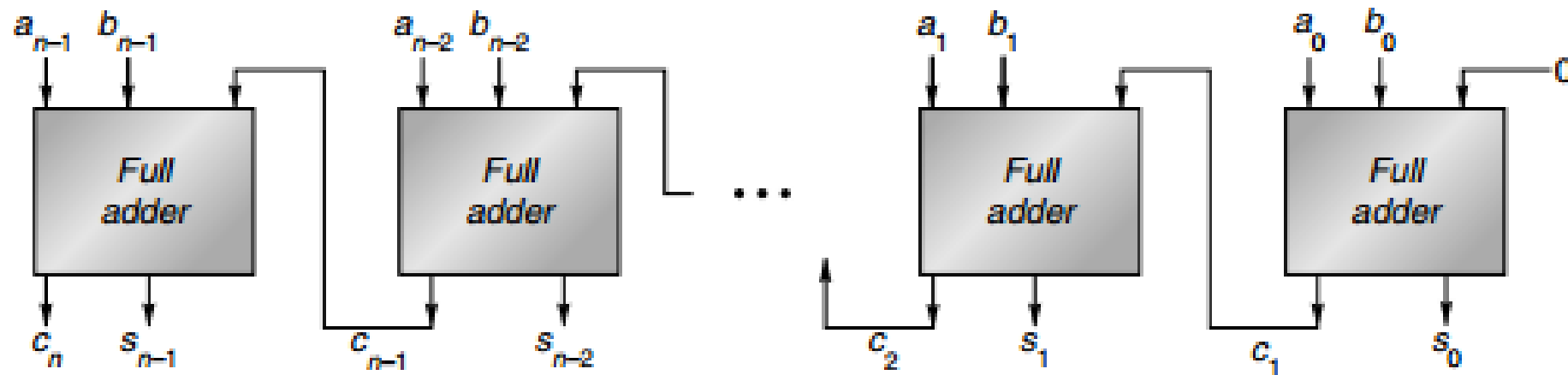


$$S = F(A, B, C_{\text{in}}) = \Sigma(1, 2, 4, 7)$$

$$C_{\text{carr}} = F(A, B, C_{\text{in}}) = \Sigma(3, 5, 6, 7)$$

# n-bit Full Adder

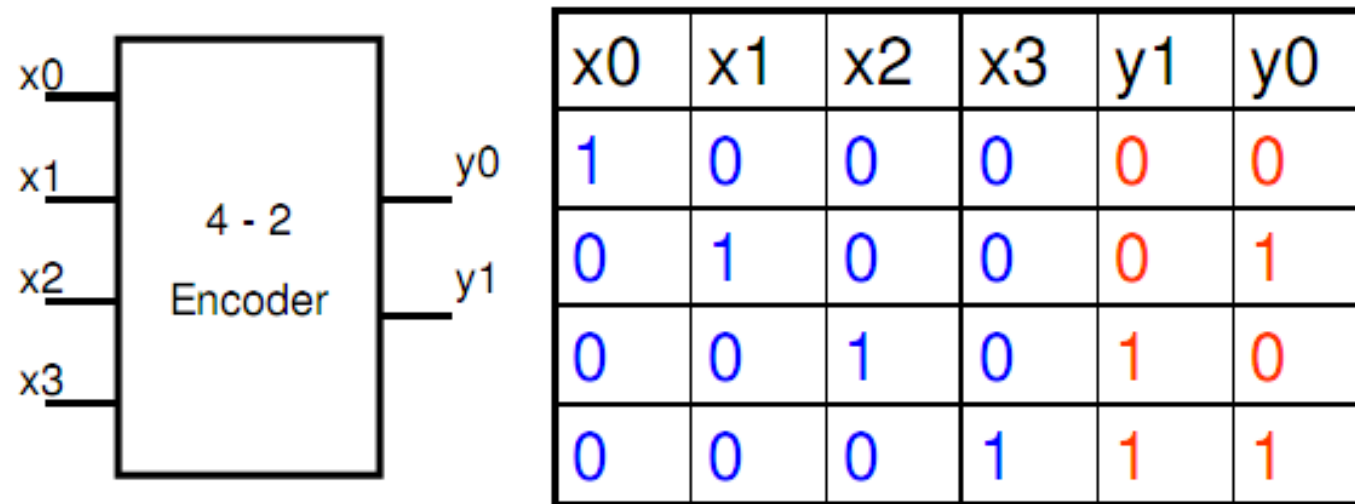
- The carry-out of one full adder is connected to the carry-in of the adder for the next most-significant bit
- The carries ripple from the least-significant bit (on the right) to the most-significant bit (on the left)



The ripple-carry adder, consists of n-bit full adders

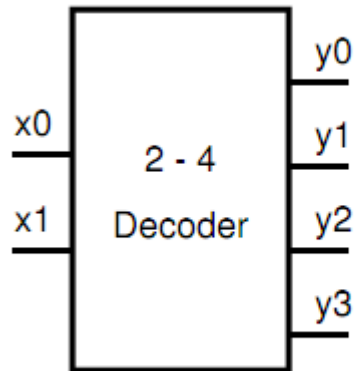
# Encoder

- Consists of  $2^n$  input,  $n$  output
- Only one of input has the 1's value at the time
- If the  $k^{\text{th}}$  input has 1's value, the output will perform a value equal to  $k$



# Decoder

- Consists of n input,  $2^n$  output
- Only one of which is asserted at any time
- If the inputs form a binary pattern with the value k, then the output = 1 is the  $k^{\text{th}}$  output



$x_1$	$x_0$	$y_0$	$y_1$	$y_2$	$y_3$
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$$y_0 = \overline{x_1}.\overline{x_0}$$

$$y_1 = \overline{x_1}.x_0$$

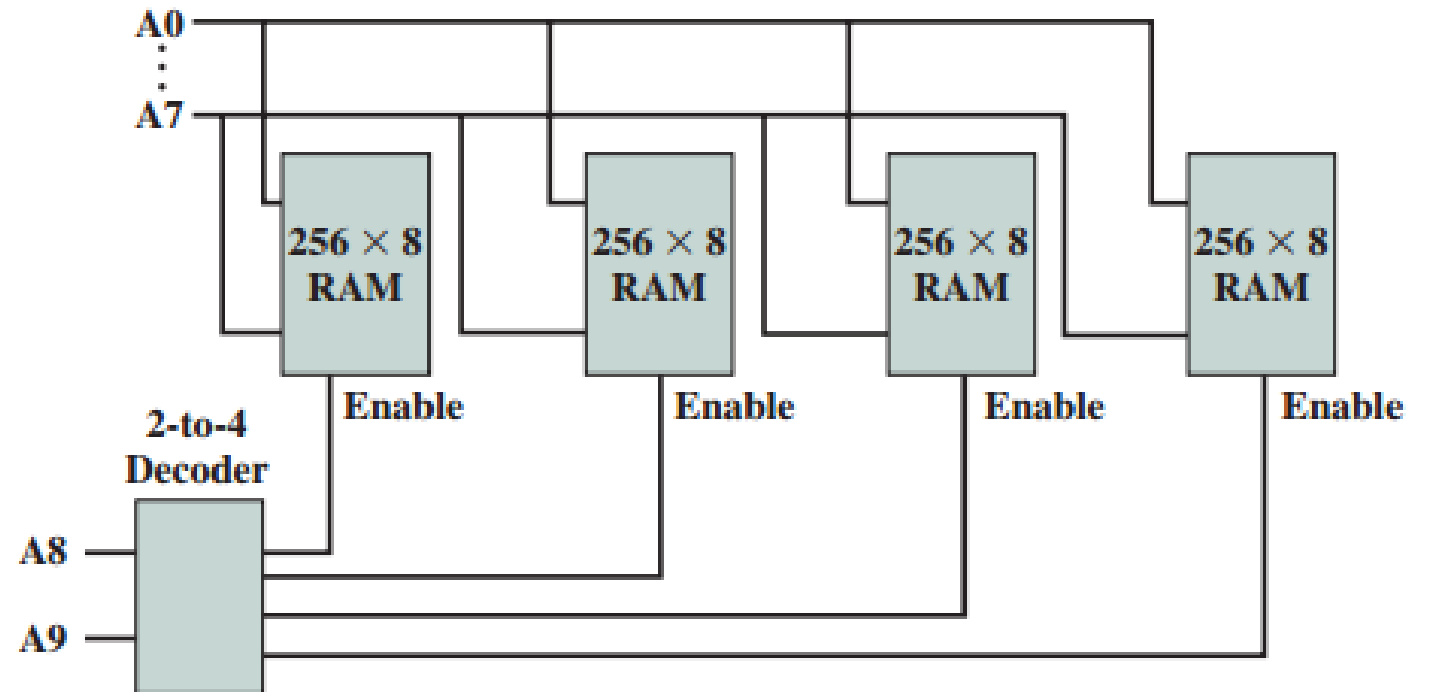
$$y_2 = x_1.\overline{x_0}$$

$$y_3 = x_1.x_0$$

# Decoder

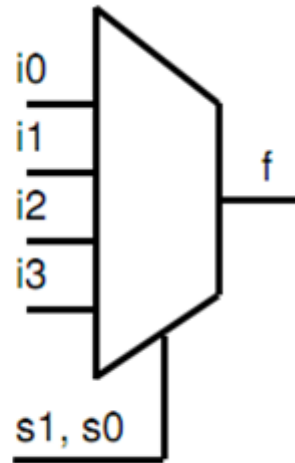
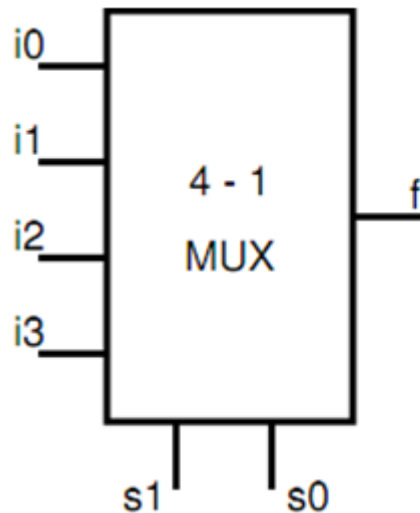
Ex: Construct a 1K-byte memory using four  $256 \times 8$ -bit RAM chips. A single unified address space, which can be broken down as follows:

Address	Chip
0000–00FF	0
0100–01FF	1
0200–02FF	2
0300–03FF	3



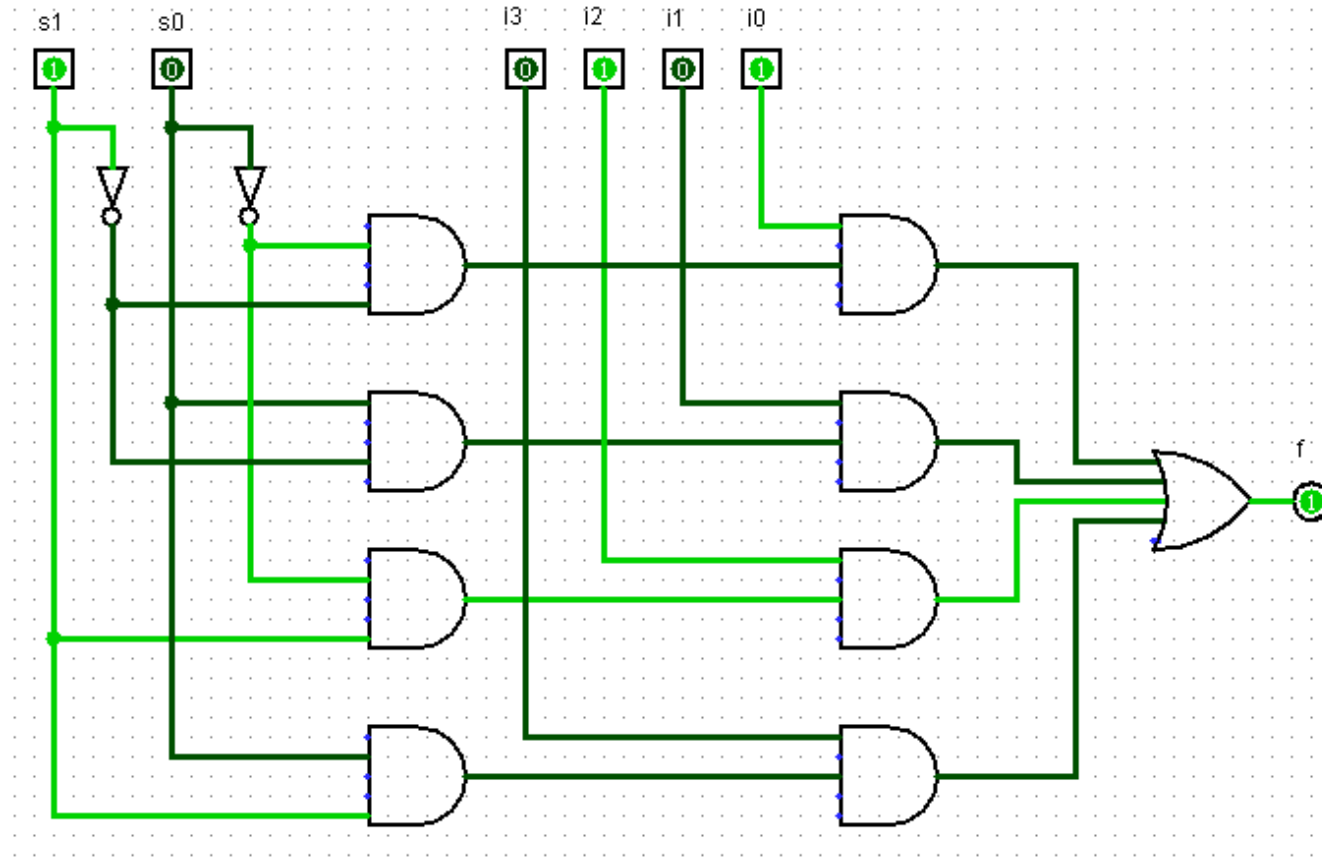
# Multiplexer

- Selects one of the  $2^n$  inputs to be only one output based on the  $n$  control input
- Ex: MUX 4-1 has 4 inputs, 1 output, and 2 control input



$s_1$	$s_0$	$f$
0	0	$i_0$
0	1	$i_1$
1	0	$i_2$
1	1	$i_3$



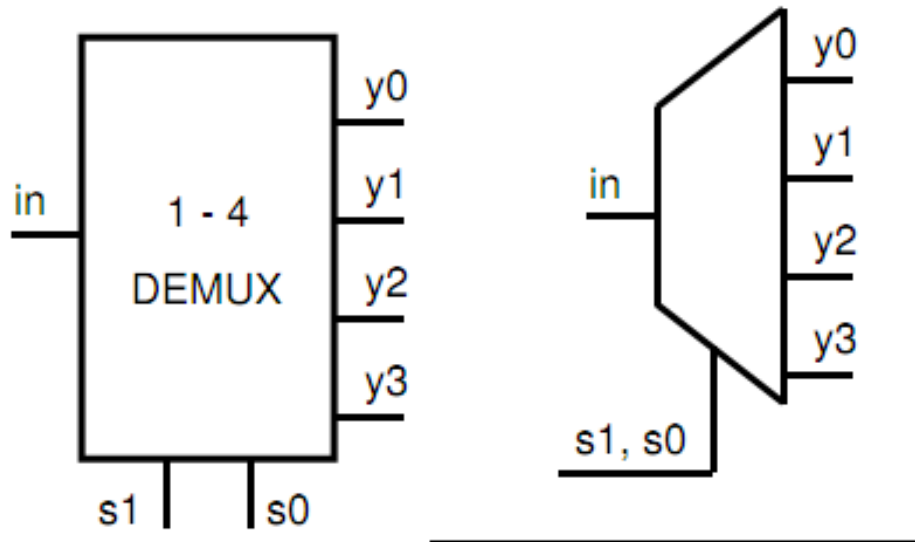


## Multiplexer

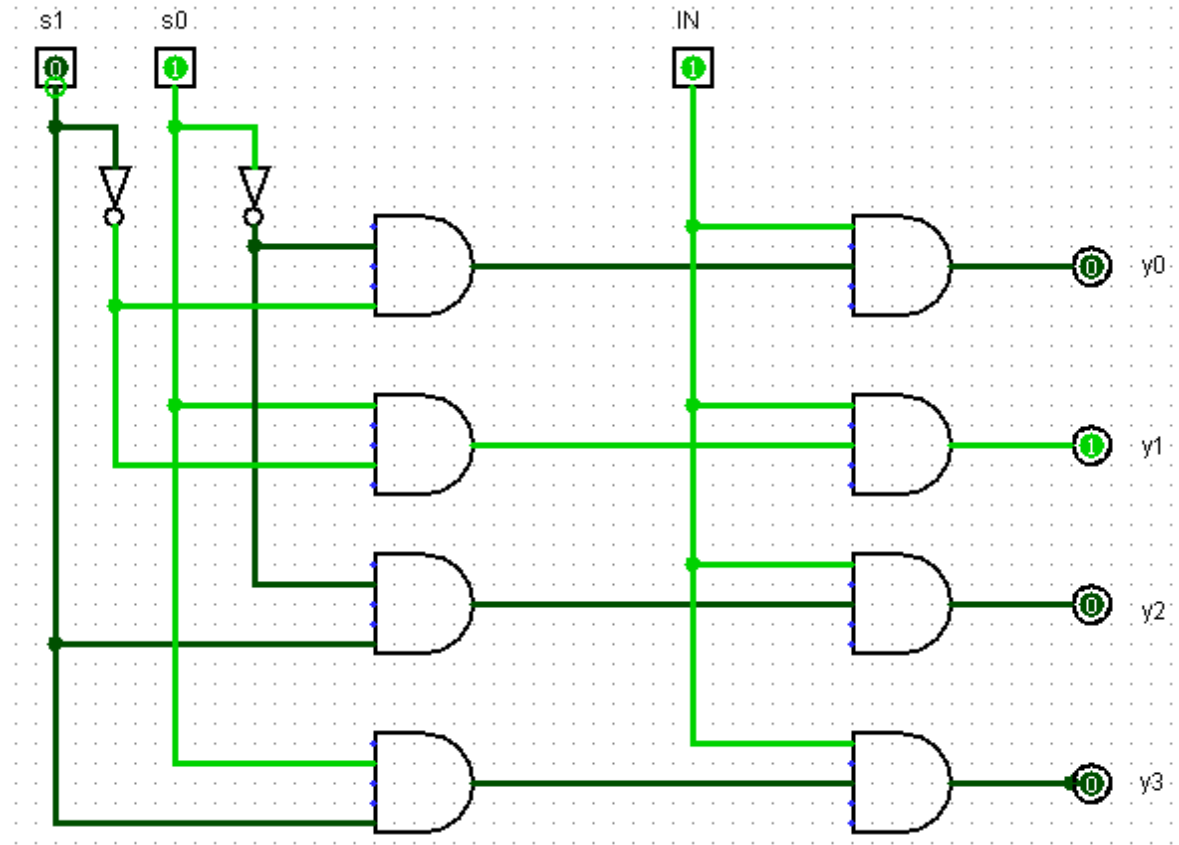
The logic circuit of MUX 4-1

# Demultiplexer

- Selects one of the  $2^n - 1$  output from only one input based on the  $2^n$  control input
- Ex: DEMUX 1-4 has 1 input, 4 output and 2 control input



s1	s0	y0	y1	y2	y3
0	0	in	0	0	0
0	1	0	in	0	0
1	0	0	0	in	0
1	1	0	0	0	in

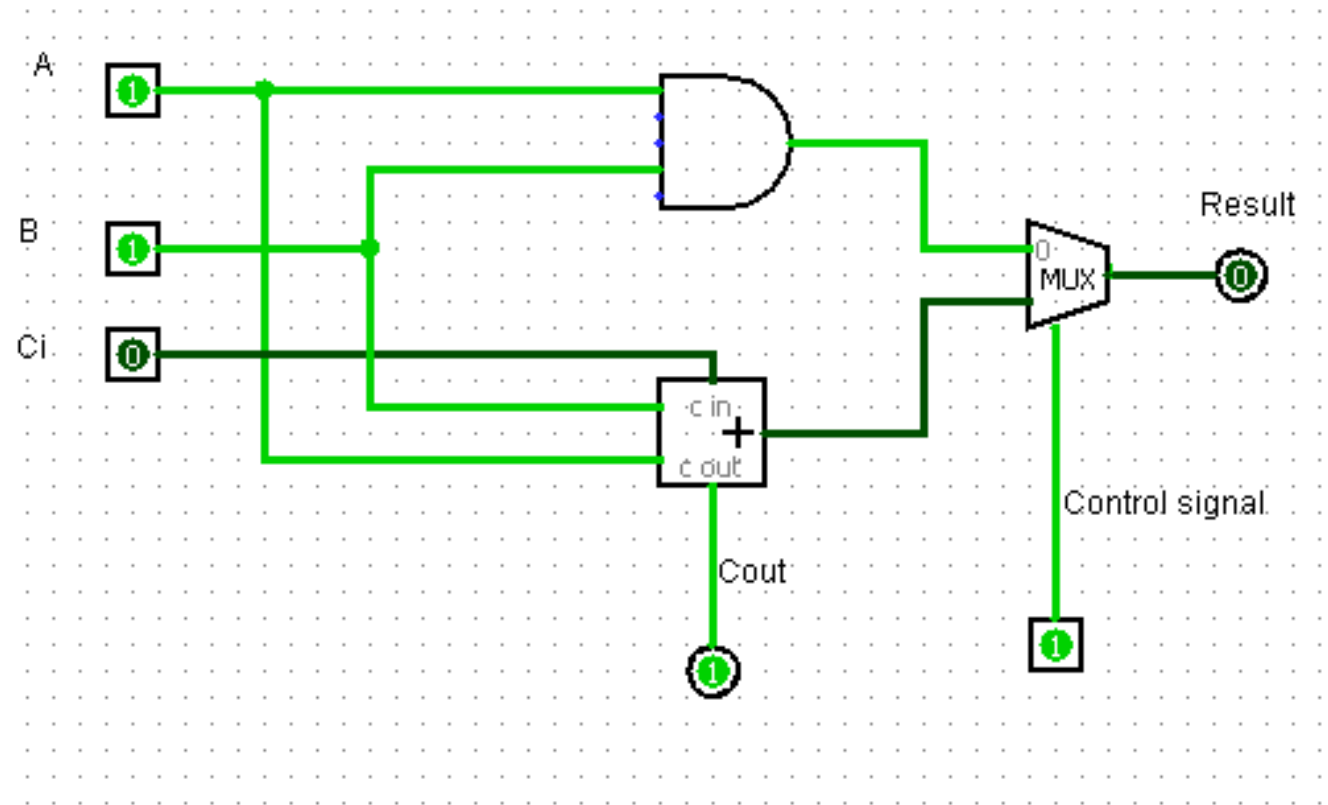


## Demultiplexer

The logic circuit of DEMUX 1-4

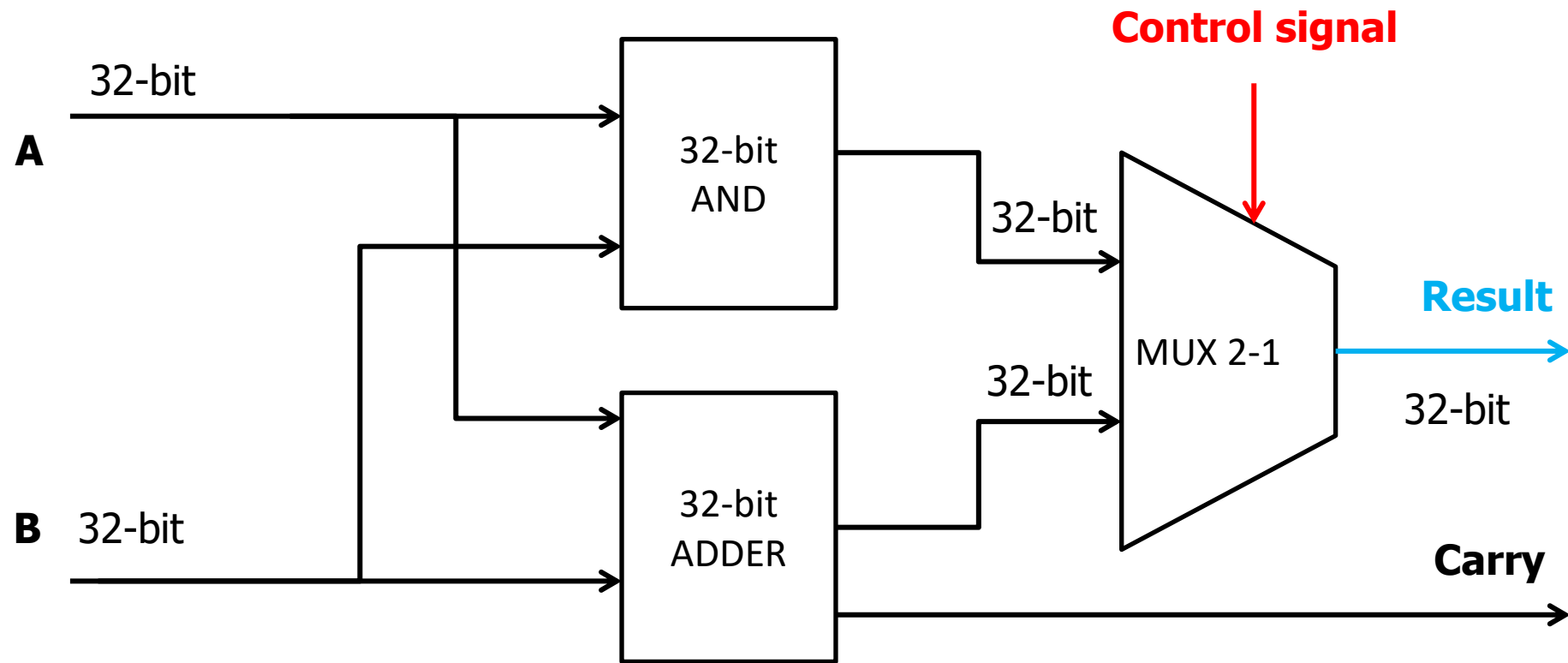
# ALU

□ The 1-bit ALU with 2 functions: and, add



# ALU

□ The 32-bits ALU with 2 functions: and, add

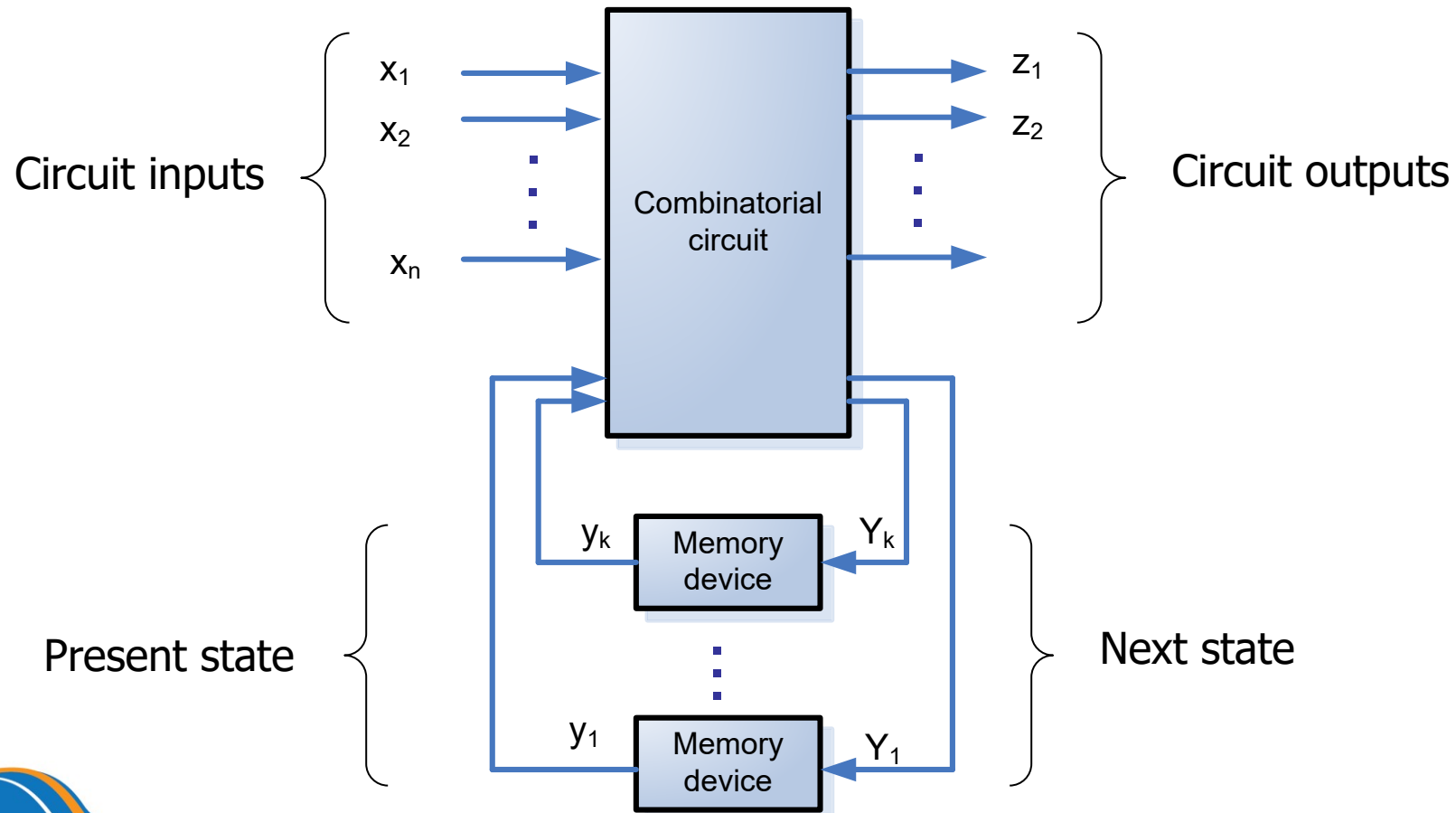


# Sequential circuit

- A sequential circuit is an interconnected set of gates and the clock pulse
- The current output of a sequential circuit depends not only on the current input, but also on the history of inputs
- Sequential circuit is capable of "remembering past states"

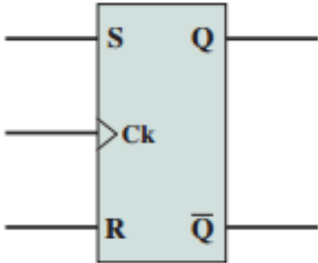
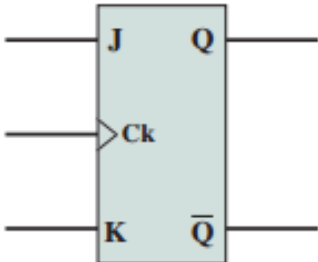
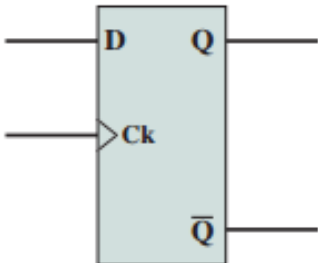


# Sequential circuit



# Sequential circuit

- Basic sequential circuit is flip-flops
- Some types of flip-flops circuit is shown in the next table

Name	Graphical Symbol	Truth Table															
S-R		<table><tr><th>S</th><th>R</th><th><math>Q_{n+1}</math></th></tr><tr><td>0</td><td>0</td><td><math>Q_n</math></td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>-</td></tr></table>	S	R	$Q_{n+1}$	0	0	$Q_n$	0	1	0	1	0	1	1	1	-
S	R	$Q_{n+1}$															
0	0	$Q_n$															
0	1	0															
1	0	1															
1	1	-															
J-K		<table><tr><th>J</th><th>K</th><th><math>Q_{n+1}</math></th></tr><tr><td>0</td><td>0</td><td><math>Q_n</math></td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td><math>\overline{Q_n}</math></td></tr></table>	J	K	$Q_{n+1}$	0	0	$Q_n$	0	1	0	1	0	1	1	1	$\overline{Q_n}$
J	K	$Q_{n+1}$															
0	0	$Q_n$															
0	1	0															
1	0	1															
1	1	$\overline{Q_n}$															
D		<table><tr><th>D</th><th><math>Q_{n+1}</math></th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	D	$Q_{n+1}$	0	0	1	1									
D	$Q_{n+1}$																
0	0																
1	1																



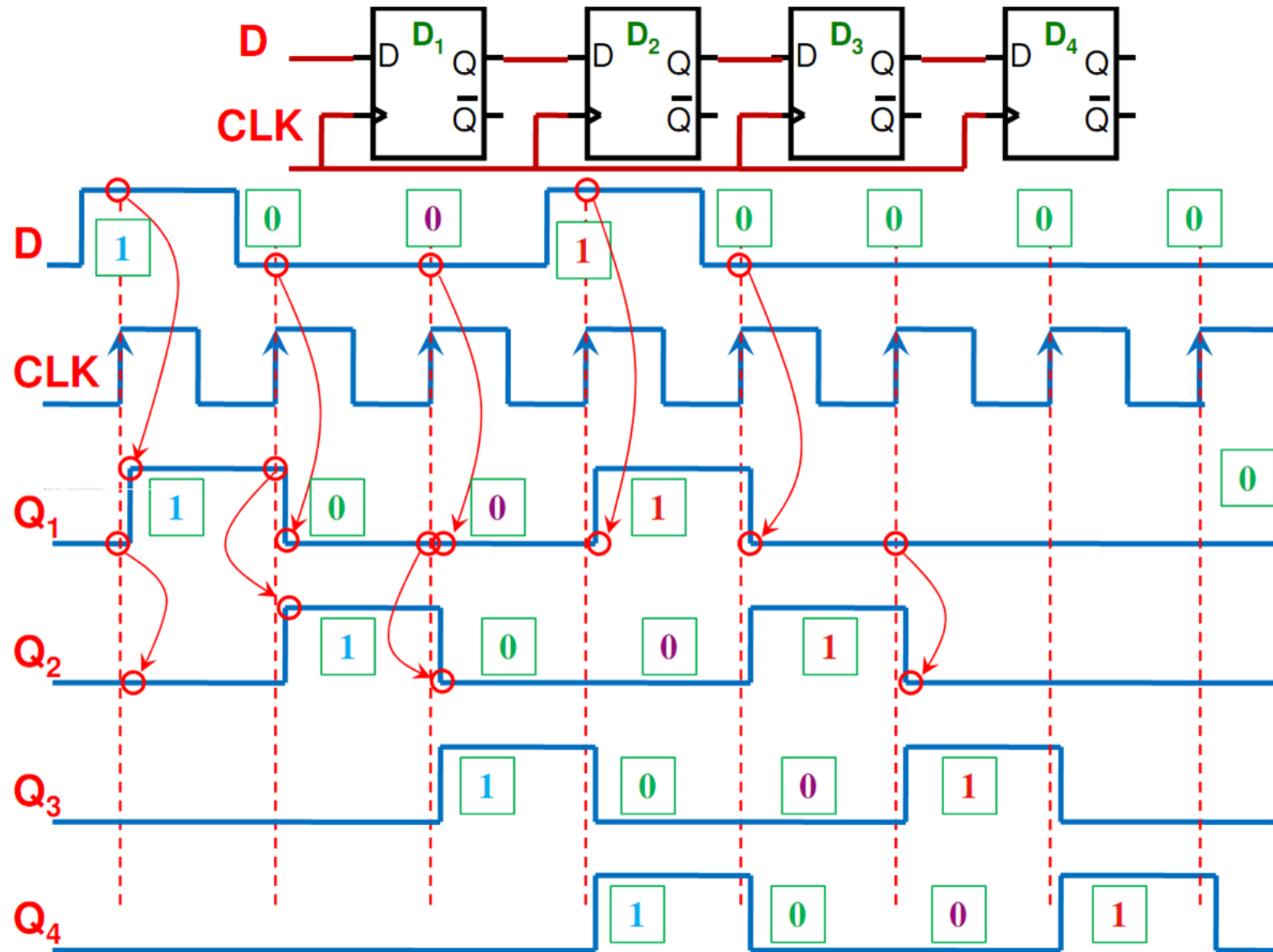
# Application of sequential circuit

- ☐ Register
- ☐ Counter
- ☐ ...



# Register

4-bit shift register



# Counter

□ A synchronous counter

