



Computer Organization & Architecture

Chapter 9 – Addition & Subtraction of Signed Numbers

Zhang Yang 张杨

cszyang@scut.edu.cn

Autumn 2025



Content of this lecture

- 9.1 Addition and Subtraction of Signed Numbers
 - Addition of Signed Numbers
 - Subtraction of Signed Numbers
 - Ripple Carry Adders

Addition of 1-bit Positive Numbers

■ Addition of 1-bit Positive Numbers

| | | | |
|-------|-------|-------|------------|
| 0 | 1 | 0 | 1 |
| + 0 | + 0 | + 1 | + 1 |
| <hr/> | <hr/> | <hr/> | <hr/> |
| 0 | 1 | 1 | 1 0 |

Carry-out

Addition of n-bit Signed Numbers (1)

- Addition Rule of n-bit Signed Numbers
 - In Signed Two's Complement Form
 - To add two numbers, add their n-bit representations, treating the sign bit as the most significant bit (MSB), ignoring the carry-out signal from the MSB position.
 - The sum will be algebraically correct value in the two's-complement representation as long as the answer is in the range -2^{n-1} through $+2^{n-1} - 1$.

Addition of n-bit Signed Numbers (2)

■ Examples of 4-bit Signed Numbers Addition

□ Textbook P14(a)-(d)

□ Overflow Examples

■ (+5) + (+4)

$$\begin{array}{r} 0\ 1\ 0\ 1 \\ +0\ 1\ 0\ 0 \\ \hline 1\ 0\ 0\ 1 \end{array} \leftarrow \text{overflow}$$

■ (-6) + (-7)

$$\begin{array}{r} 1\ 0\ 0\ 1 \\ +1\ 0\ 1\ 0 \\ \hline \underline{1}\ 0\ 0\ 1\ 1 \end{array} \leftarrow \text{overflow}$$

Addition of n-bit Signed Numbers (3)

■ Arithmetic Overflow

- The result of an arithmetic operation is outside the representable range.
- If two numbers are added, and they have the same sign, then overflow occurs if and only if the result has the opposite sign to both summands.
- The carry-out signal from the sign-bit position is not a sufficient indicator of overflow when adding signed numbers. Overflow can occur whether or not there is a carry-out.



Content of this lecture

- 9.1 Addition and Subtraction of Signed Numbers
 - Addition of Signed Numbers
 - Subtraction of Signed Numbers
 - Ripple Carry Adders

Subtraction of n-bit Signed Numbers (1)

- Subtraction Rule of n-bit Signed Numbers
 - In Signed Two's Complement Form
 - To subtract two numbers X and Y , that is, to perform $X - Y$, form the 2's-complement of Y and then add it to X , as in rule1.
 - The result will be the algebraically correct value in the two's-complement representation system if the answer is in the range -2^{n-1} through $+2^{n-1} - 1$.
 - Essence of Rule $X - Y = X + (-Y)$

Subtraction of n-bit Signed Numbers (2)

- Twos Complement Operation (Negation)
 - Take the boolean complement of each bit of the integer (including the sign bit). That is ,set each 1 to 0 and each 0 to 1.
 - Treating the result as an unsigned binary integer, add 1.

Subtraction of n-bit Signed Numbers (3)

■ Twos Complement Operation (ctd.)

□ Example

- $Y=6$ and it can be represented by 0110 using signed 2's complement form. What is the signed 2's complement form of $-Y$?
- Solution:
 - Because $Y=0110$, after negation 1001
 - So $(-Y) = 1001 + 0001 = 1010$

Subtraction of n-bit Signed Numbers (4)

■ Examples of 4-bit Signed Numbers Subtraction

□ Textbook P14 Figure 1.6 (e)-(f)

□ Overflow Example: $X = 6$, $Y = -7$, $X - Y = ?$

■ $X = 6 = 0110$

■ $Y = -7 = 1001$

■ $-Y = 0111$

$$\begin{array}{r} 0110 \\ +0111 \\ \hline 1101 = (-3) \quad \longleftarrow \text{overflow} \end{array}$$

Subtraction of n-bit Signed Numbers (5)

■ Conclusions

- The examples in Figure 1.6 (P14) show that two, n-bit, signed numbers can be added using n-bit binary addition, treating the sign bit the same as the other bit.
- In other words, a logic circuit that is designed to add unsigned binary numbers can also be used to add signed numbers in 2's-complement.

Addition and Subtraction Summary

■ 知识点: Addition and Subtraction of Signed Numbers

- Addition Rule
- Subtraction Rule
- Arithmetic Overflow
- Twos Complement Operation

■ 掌握程度

- 给定两个带符号整数，正确计算出两个数的和，并判断是否溢出。
- 给定两个带符号整数，正确计算出两个数的差，并判断是否溢出。

Exercise (1)

- 1. Assume that X and Y are two 8-bit signed two's complement numbers 01011110 and 11001010. What is the result of $X+Y$?
 - ☐ A. 00101000
 - ☐ B. 10010100
 - ☐ C. 01000100
 - ☐ D. 00011100

Exercise (2)

- 2. Assume that X and Y are two 8-bit signed two's complement numbers 01011110 and 11001010. What is the result of X-Y?
 - ☐ A. 00101000
 - ☐ B. 10010100
 - ☐ C. 01000100
 - ☐ D. overflow

Exercise (3)

- 3. Subtracting a negative integer from another negative integer in 2's-complement binary arithmetic can cause an overflow.
 - ☐ A. true
 - ☐ B. false



Content of this lecture

- 9.1 Addition and Subtraction of Signed Numbers
 - Addition of Signed Numbers
 - Subtraction of Signed Numbers
 - Ripple Carry Adders

1-bit Full Adder (1)

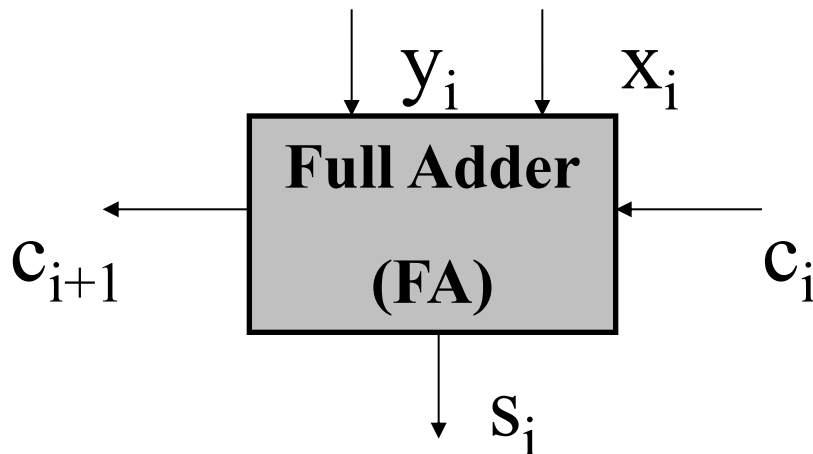
- Example: $X = 7$, $Y = 6$, $X + Y = ?$

$$\begin{array}{r} X \qquad \qquad 7 \\ + Y \\ \hline Z \end{array} = \begin{array}{r} \qquad \qquad + 6 \\ \hline 13 \end{array} = \begin{array}{r} \qquad \qquad 0 \ 1 \ 1 \ 1 \\ + \textcolor{red}{0} \ 0 \ \textcolor{red}{1} \ 1 \ \textcolor{red}{1} \ 1 \ \textcolor{red}{0} \ 0 \ 0 \\ \hline \qquad \qquad 1 \ 1 \ 0 \ 1 \end{array}$$

1-bit Full Adder (2)

■ Full Adder

- A **full adder** circuit takes three bits of input, and produces a two-bit output consisting of a sum and a carry out.



$$\begin{array}{r} x_i \\ y_i \\ c_i \\ \hline s_i \end{array}$$

1-bit Full Adder (3)

■ Logic Truth Table

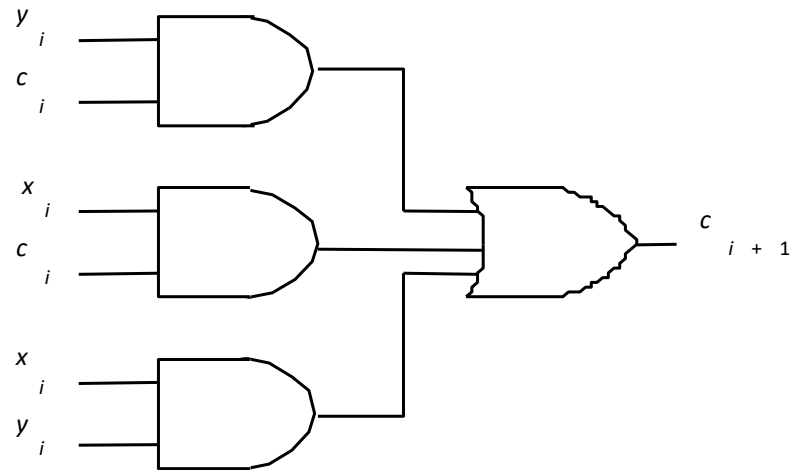
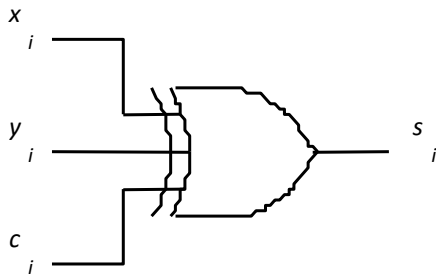
| x_i | y_i | c_i | s_i | c_{i+1} |
|-------|-------|-------|-------|-----------|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

1-bit Full Adder (4)

■ Logic Expressions

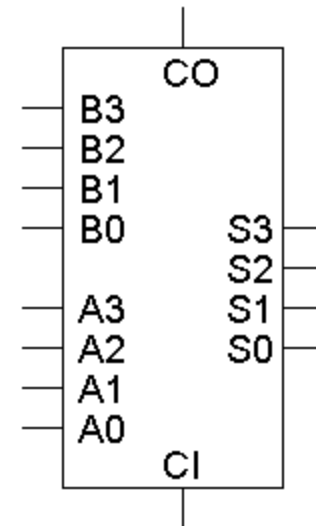
$$\begin{aligned} S_i &= \bar{x}_i \bar{y}_i c_i + \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + x_i y_i c_i \\ &= x_i \oplus y_i \oplus c_i \end{aligned}$$

$$C_{i+1} = y_i c_i + x_i c_i + x_i y_i$$



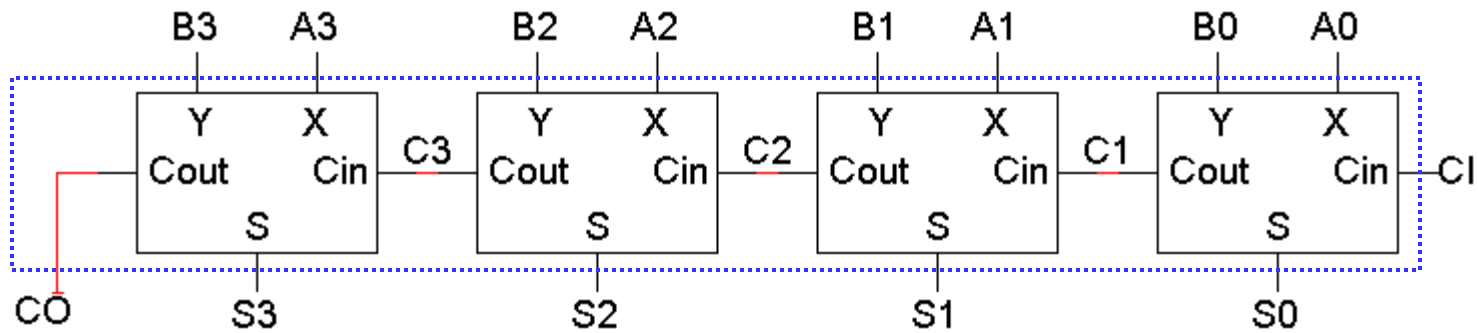
A 4-bit Adder (1)

- Four full adders together make a 4-bit adder.
- There are nine total inputs:
 - Two 4-bit numbers, A3 A2 A1 A0 and B3 B2 B1 B0
 - An initial carry in, CI
- The five outputs are:
 - A 4-bit sum, S3 S2 S1 S0
 - A carry out, CO



A 4-bit Adder (2)

■ Internal Figure



n-bit Ripple-Carry Adder (1)

Input: $\begin{cases} X = x_{n-1} \dots x_1 x_0 \\ Y = y_{n-1} \dots y_1 y_0 \\ c_0 \end{cases}$

■ n-bit Ripple-Carry Adder

Output: $\begin{cases} S = s_{n-1} \dots s_1 s_0 \\ c_n \end{cases}$

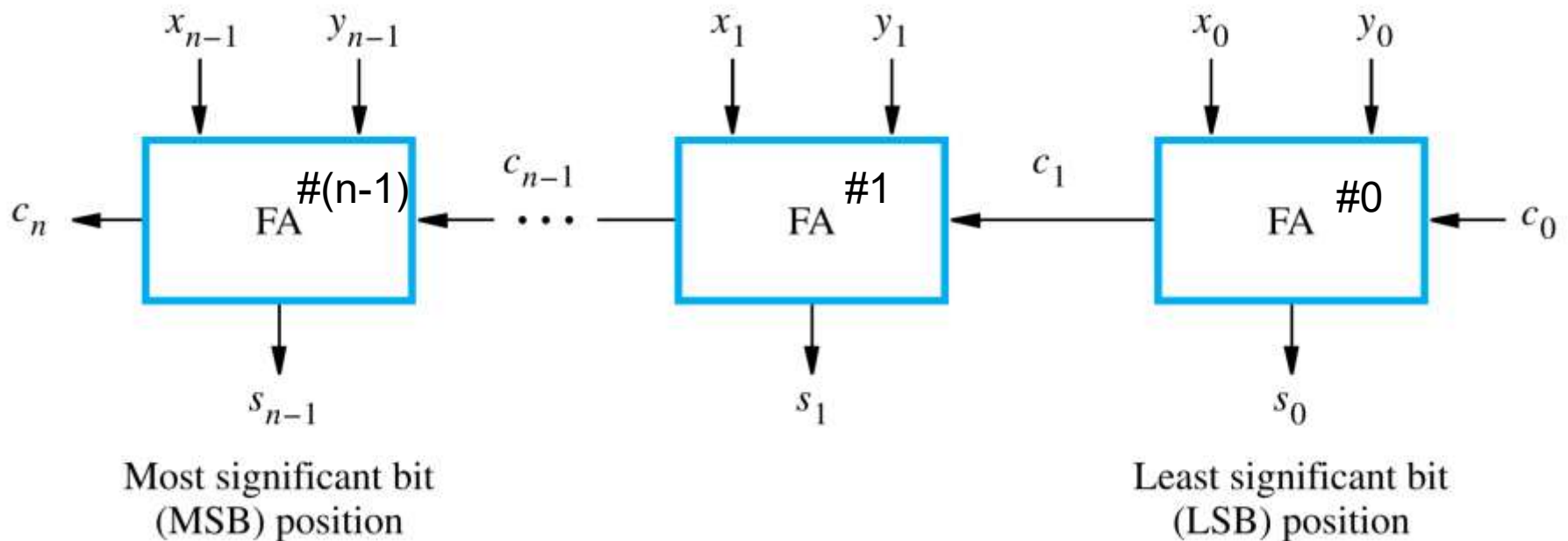


Figure 9.2 (b) An n-bit ripple-carry adder

n-bit Ripple-Carry Adder (2)

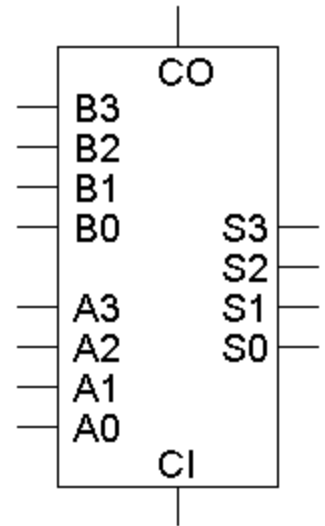
■ Overflow Detection

□ Overflow = $x_{n-1}y_{n-1}\bar{s}_{n-1} + \bar{x}_{n-1}\bar{y}_{n-1}s_{n-1}$

| x_{n-1} | y_{n-1} | s_{n-1} | Overflow |
|-----------|-----------|-----------|----------|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

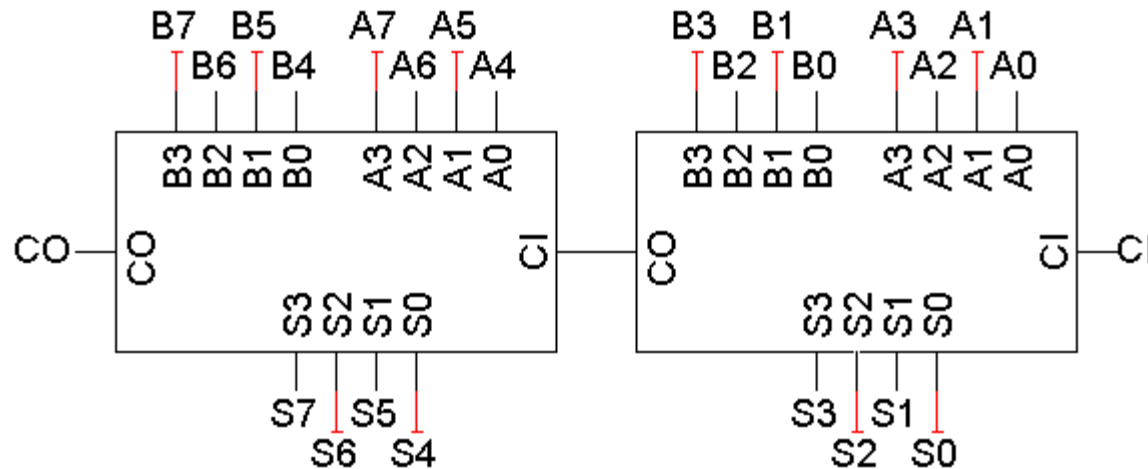
Hierarchical Adder Design (1)

- Question: When you add two 4-bit numbers the carry in is always 0, so why does the 4-bit adder have a CI input?
- Solution:
 - One reason is so we can put 4-bit adders together to make even larger adders!
This is just like how we put four full adders together to make the 4-bit adder in the first place.



Hierarchical Adder Design (2)

- Using 4-bit Adders to Design A 8-bit Adder



Hierarchical Adder Design (3)

- Using n -bit Adders to Design A kn -bit Adder

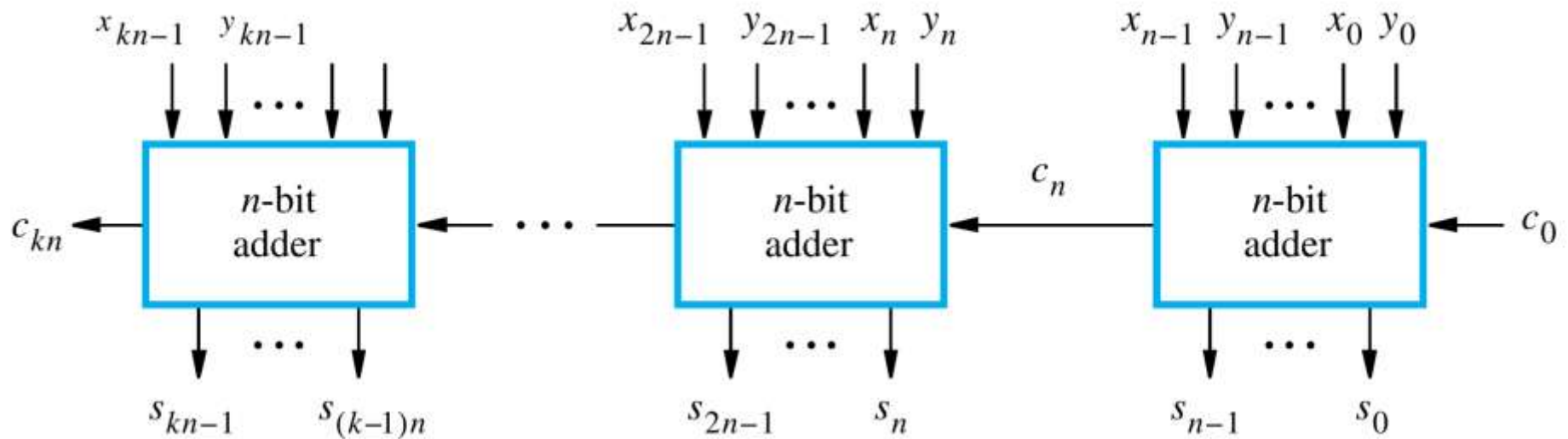
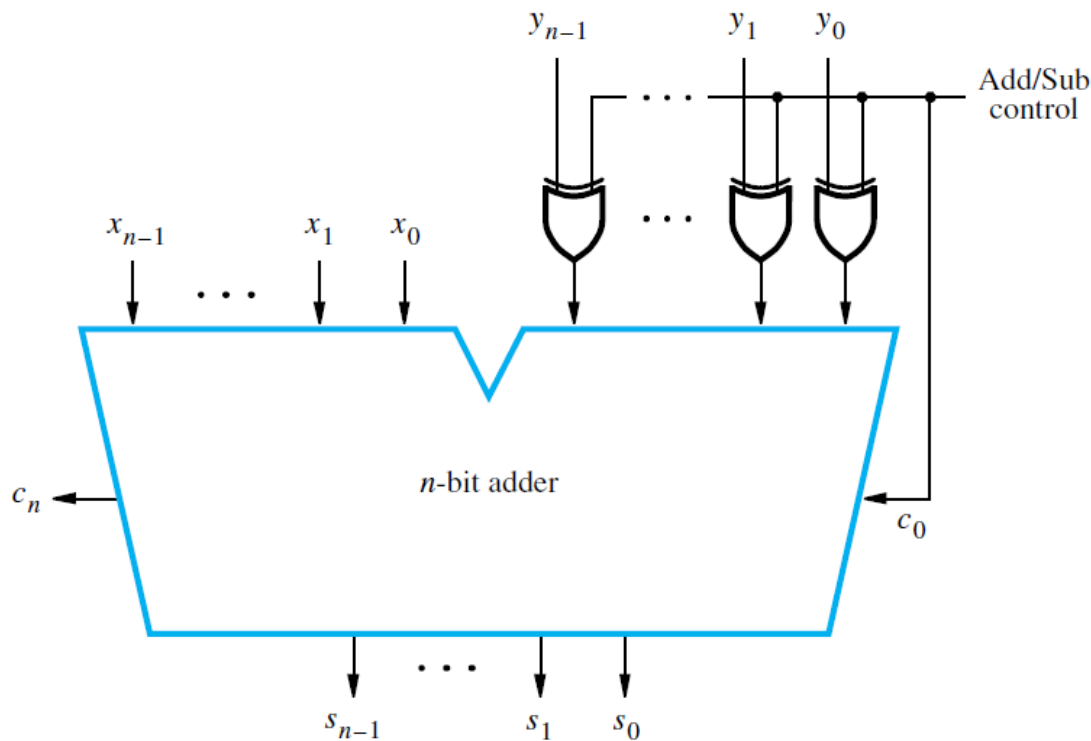


Figure 9.2 (c) Cascade of k n -bit adders

Addition/Subtraction Logic Unit

■ Addition/Subtraction Logic Unit



$$0 \oplus A = A$$

$$1 \oplus A = \overline{A}$$

Control=0 $c_0=0$

Perform addition

Control=1 $c_0=1$

Perform subtraction

Figure 9.3 Binary addition/subtraction logic circuit.

Ripple Carry Adder Summary

- 知识点: Ripple Carry Adder

- ☐ 1-bit full adder
- ☐ n-bit ripple-carry adder
- ☐ Hierarchical adder

- 掌握程度

- ☐ 能够写出1位全加器中和与进位输出的逻辑表达式，并画出逻辑图。
- ☐ 掌握n位行波进位加法器的原理。
- ☐ 给定一个位数较少的加法器，掌握用来构造较多位数加法器的方法。

Exercise (1)

- 1. In a 1-bit full adder, which expression is s_i ?
 - ☐ A. $x_i + y_i$
 - ☐ B. $x_i \oplus y_i$
 - ☐ C. $x_i y_i$
 - ☐ D. $x_i \oplus y_i \oplus c_i$

Exercise (2)

- 2. In a 1-bit full adder, which expression is c_{i+1} ?
 - ☐ A. $x_i + y_i + c_i$
 - ☐ B. $x_i y_i c_i$
 - ☐ C. $x_i y_i + x_i c_i + y_i c_i$
 - ☐ D. $x_i \oplus y_i \oplus c_i$

Exercise (3)

- 3. If we want to construct a 64-bit adder, how many adders do we need if we have some 4-bit ripple carry adders?
 - ☐ A. 32
 - ☐ B. 16
 - ☐ C. 8
 - ☐ D. 4



Class Discussion

- How to improve the speed of the n -bit ripple carry adder?