

NHẬP MÔN MẠCH SỐ

Chương 5 – phần 1



**Mạch tổ hợp:
Mạch tính toán số học**

Tổng quan



Chương này sẽ học về:

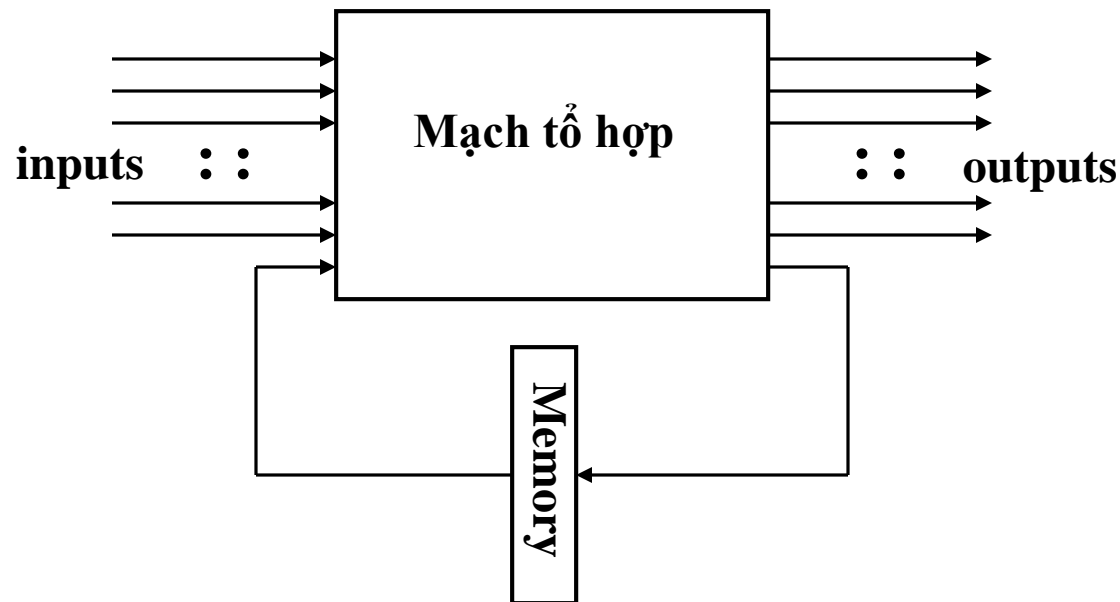
- Một số mạch logic tổ hợp thông dụng
- Thiết kế các mạch logic tổ hợp phức tạp sử dụng các mạch logic tổ hợp thông dụng

Phân biệt mạch tổ hợp và tuần tự



MẠCH TỔ HỢP


- Ngõ ra sẽ thay đổi lập tức khi ngõ vào thay đổi



MẠCH TUẦN TỰ

- Ngõ ra sẽ thay đổi phụ thuộc vào ngõ vào và trạng thái trước đó.
- Mạch có tính chất nhớ

Nội dung

- 
1. Mạch cộng (Carry Ripple (CR) Adder)
 2. Mạch cộng nhìn trước số nhớ - (Carry Look-Ahead (CLA) Adder)
 3. Mạch cộng/ mạch trừ
 4. Đơn vị tính toán luận lý (Arithmetic Logic Unit)
 5. Mạch giải mã (Decoder)/ Mạch mã hoá (Encoder)
 6. Mạch dồn kênh (Multiplexer)/ Mạch chia kênh (Demultiplexer)
 7. Mạch tạo Parity/ Mạch kiểm tra Parity
 8. Mạch so sánh (Comparator)

Nội dung

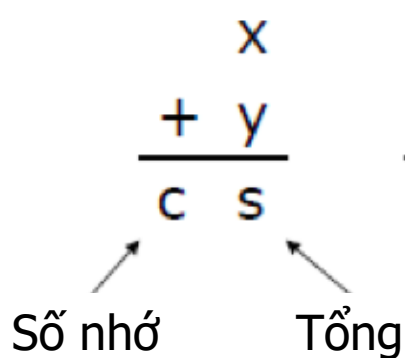
1. Mạch cộng (Carry Ripple (CR) Adder)
2. Mạch cộng nhìn trước số nhớ - (Carry Look-Ahead (CLA) Adder)
3. Mạch cộng/ mạch trừ
4. Đơn vị tính toán luận lý (Arithmetic Logic Unit)
5. Mạch giải mã (Decoder)/ Mạch mã hoá (Encoder)
6. Mạch dồn kênh (Multiplexer)/ Mạch chia kênh (Demultiplexer)
7. Mạch tạo Parity/ Mạch kiểm tra Parity
8. Mạch so sánh (Comparator)



1. Mạch cộng Carry Ripple (CR)

Mạch cộng bán phần (Half Adder)

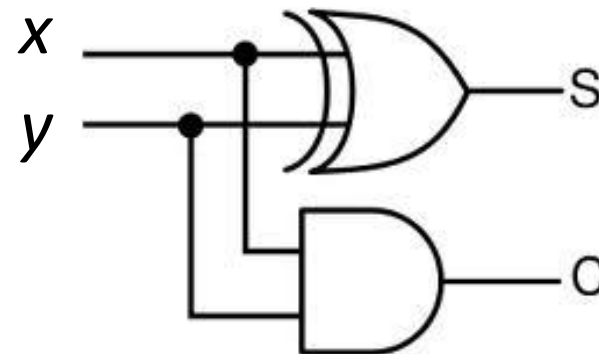
- Cộng 2 số 1 bit có 4 trường hợp



0	0	1	1
$+ 0$	$+ 1$	$+ 0$	$+ 1$
$0 \quad 0$	$0 \quad 1$	$0 \quad 1$	$1 \quad 0$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

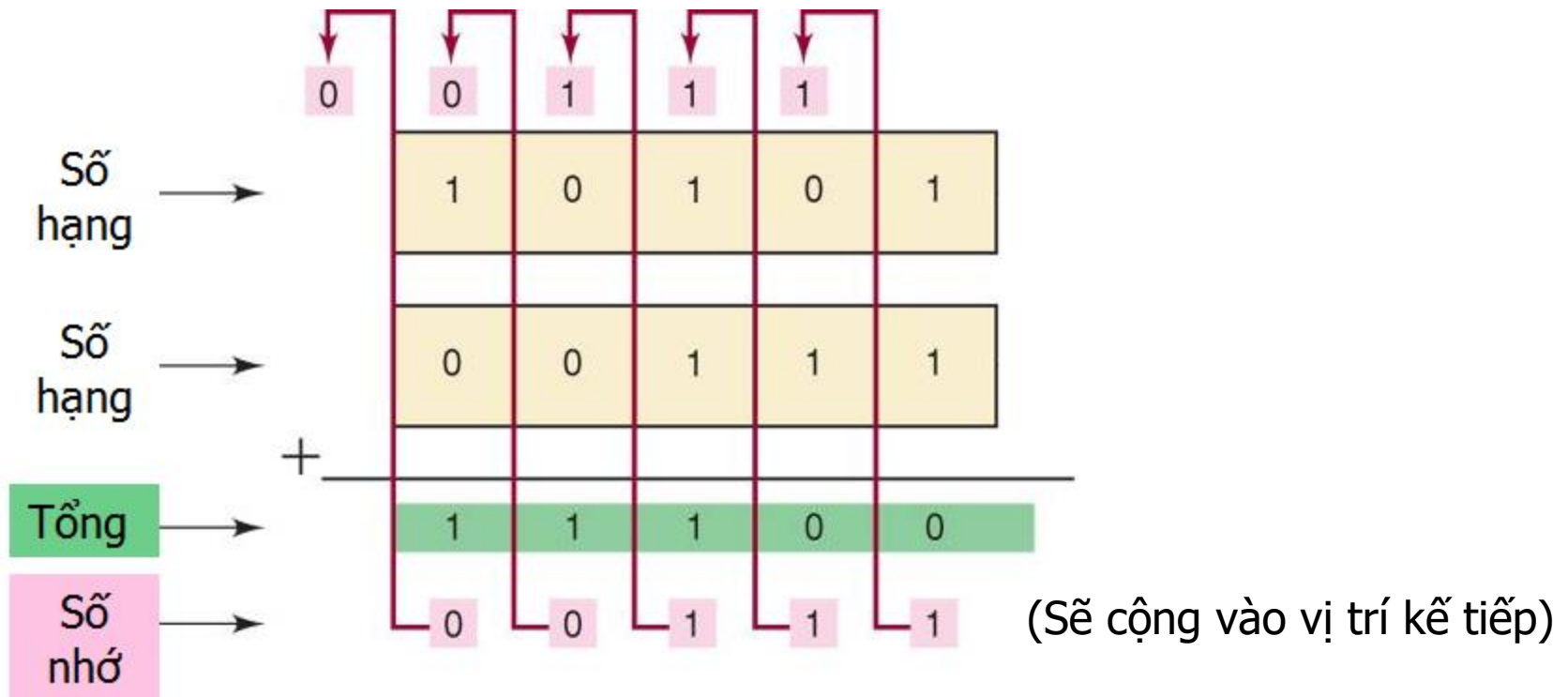
Mạch cộng 1 bit có tổng và số nhớ như thế này được gọi là mạch cộng bán phần (**HA**)



Sơ đồ mạch

Mạch cộng nhị phân song song

- Cộng những số có 2 hoặc nhiều bit
 - Cộng từng cặp bit bình thường
 - Nhưng ở vị trí cặp bit i , có thể có carry-in từ bit $i-1$



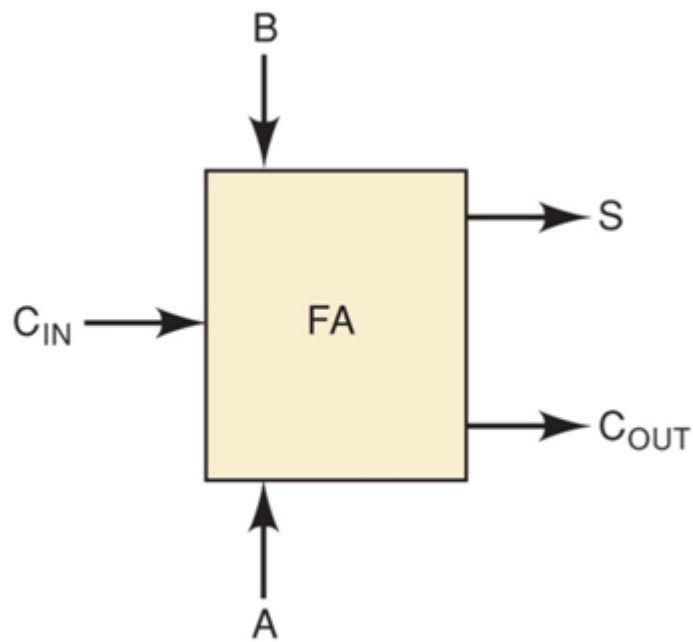
Thiết kế một bộ cộng toàn phần (Full Adder)



Bộ cộng toàn phần (**FA**)

- **3 ngõ vào** (2 ngõ vào cho 2 số 1-bit cần tính tổng, và 1 ngõ vào cho số nhớ đầu vào (**carry-in**))
- **2 ngõ ra** (1 ngõ ra cho tổng và 1 cho số nhớ đầu ra (**carry-out**))

Thiết kế một bộ cộng toàn phần (Full Adder)



Ký hiệu

Bảng sự thật

Augend bit input	Addend bit input	Carry bit input	Sum bit output	Carry bit output
A	B	C _{IN}	S	C _{OUT}
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

Thiết kế một bộ cộng toàn phần (Full Adder)

Bảng sự thật

Augend bit input	Addend bit input	Carry bit input	Sum bit output	Carry bit output
A	B	C_{IN}	S	C_{OUT}
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

$c \backslash xy$	00	01	11	10
0		1		1
1	1		1	

$$S_i = x_i \oplus y_i \oplus c_i$$

$c \backslash xy$	00	01	11	10
0			1	
1		1	1	1

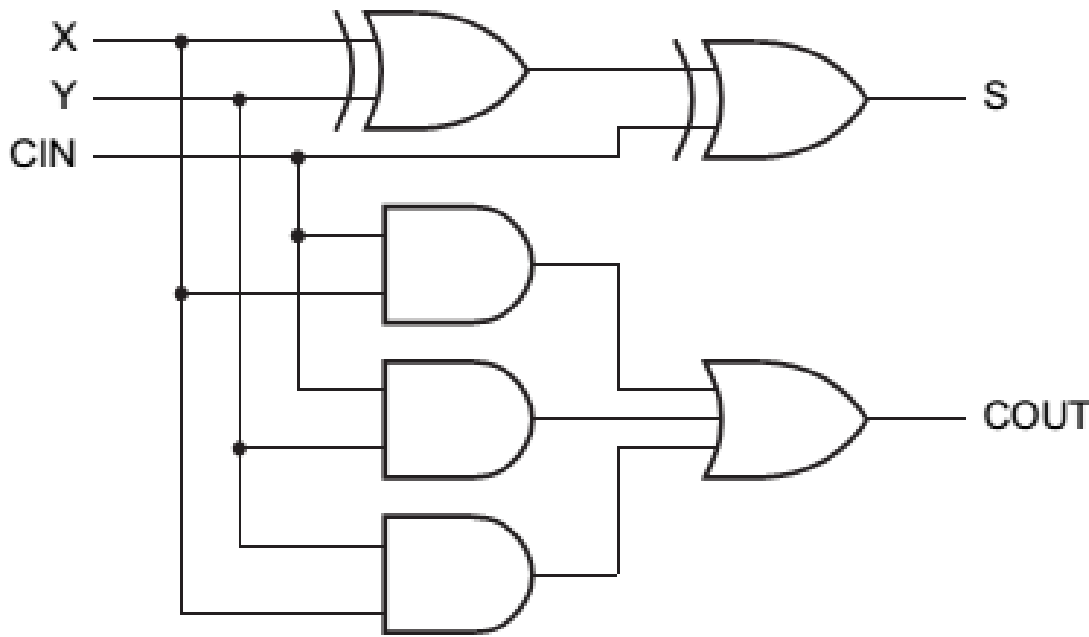
$$C_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

$$C_i = C_{IN} \quad C_{i+1} = C_{OUT}$$

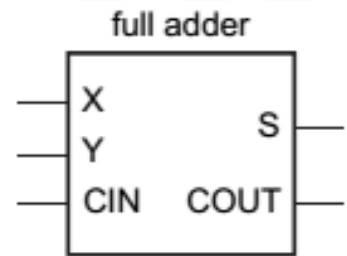
Thiết kế một bộ cộng toàn phần (Full Adder)

$$S_i = x_i \oplus y_i \oplus c_i$$

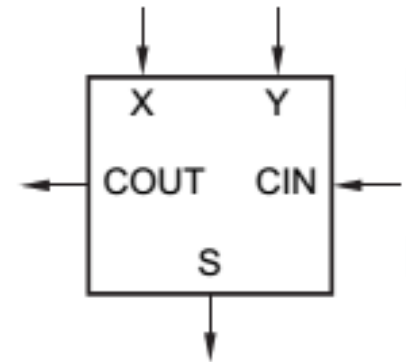
$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i \quad c_i = c_{IN}$$
$$c_{i+1} = c_{OUT}$$



Sơ đồ mạch



Ký hiệu



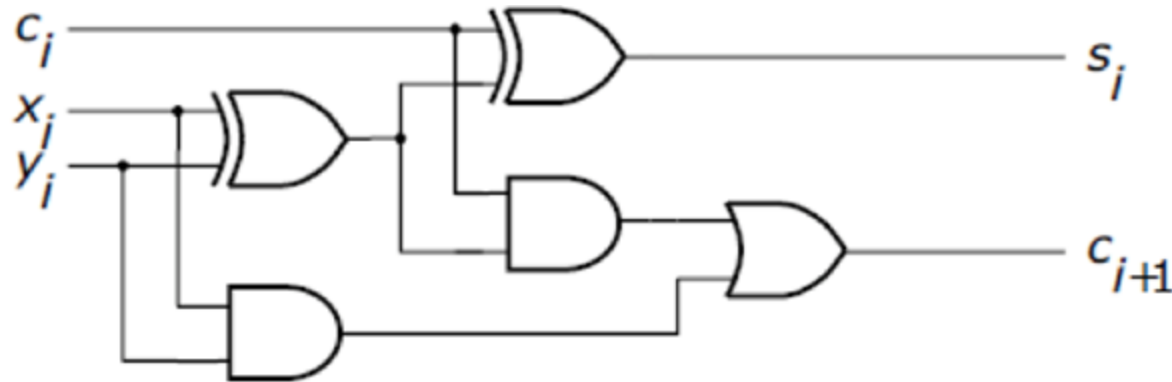
Ký hiệu khác

Thiết kế một bộ cộng toàn phần (Full Adder)

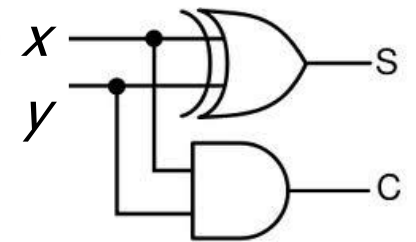
- Sử dụng lại HA

$$S_i = x_i \oplus y_i \oplus c_i \quad c_{i+1}$$

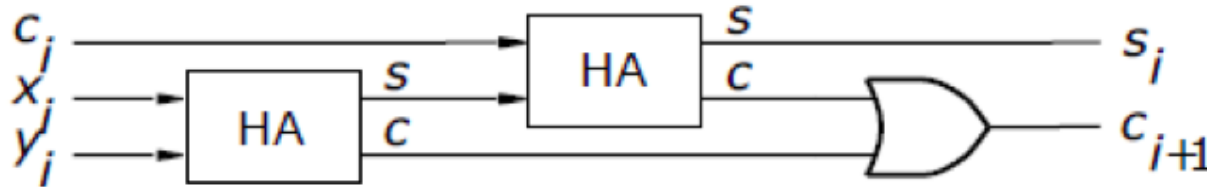
xy		00	01	11	10
c					
0				1	
1			1	1	1



Sơ đồ mạch



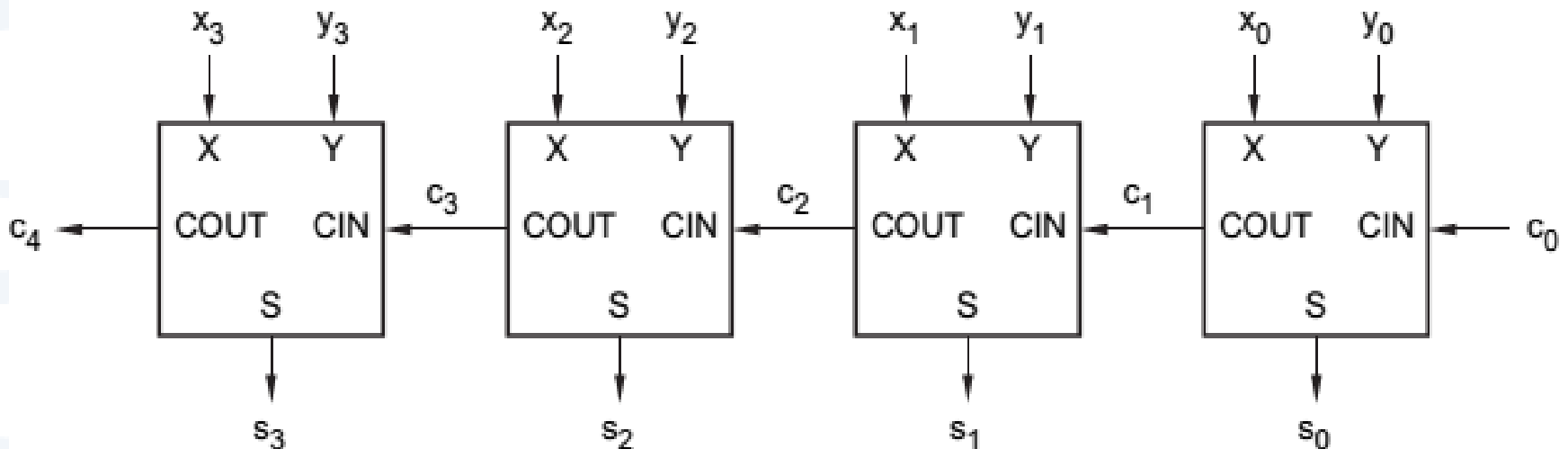
Sơ đồ mạch HA



Sơ đồ mạch **FA** sử dụng lại HA

Mạch cộng Carry Ripple (CR)

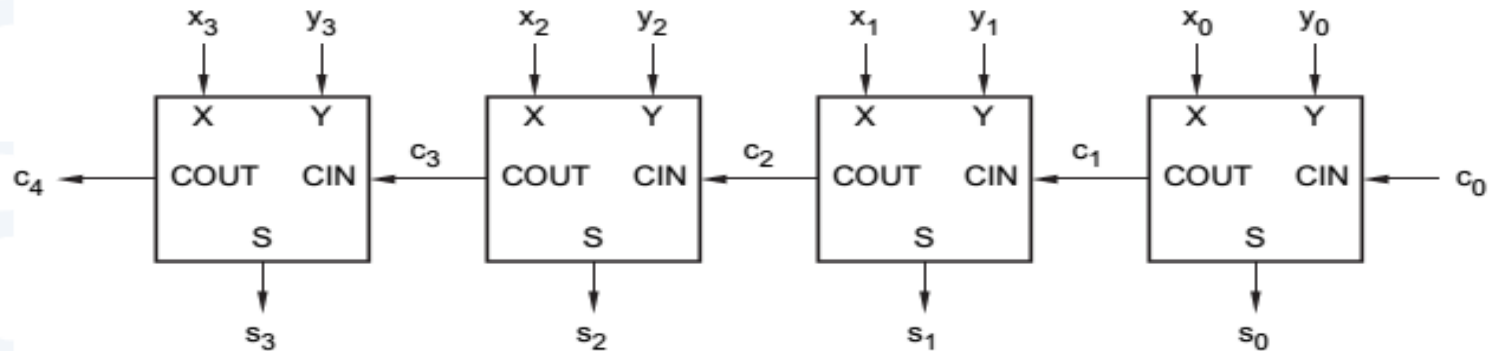
- Sơ đồ biểu diễn mạch cộng 4 bit song song sử dụng full adder



Mạch cộng Carry Ripple

- Mạch FA bắt đầu với việc cộng các cặp bit từ LSB đến MSB
 - Nếu carry xuất hiện ở vị trí bit i , nó được cộng thêm vào phép cộng ở vị trí bit thứ $i+1$
- Việc kết hợp như vậy thường được gọi là mạch cộng **Carry-Ripple**
 - vì carry được “ripple” từ FA này sang các FA kế tiếp
 - Tốc độ phép cộng bị giới hạn bởi quá trình truyền số nhớ

Mạch cộng Carry Ripple



- Mỗi FA có một khoảng trễ (delay), giả sử là Δt
 - Độ trễ phụ thuộc vào số lượng bit
 - Carry-out ở FA đầu tiên C_1 có được sau Δt
 - Carry-out ở FA đầu tiên C_2 có được sau $2\Delta t$
- $\Rightarrow C_n$ được tính toán sau $n\Delta t$

Mô hình carry look ahead (CLA) thường được sử dụng để cải thiện tốc độ



2. Mạch cộng nhìn trước số nhớ Carry Look-Ahead (CLA) Adder

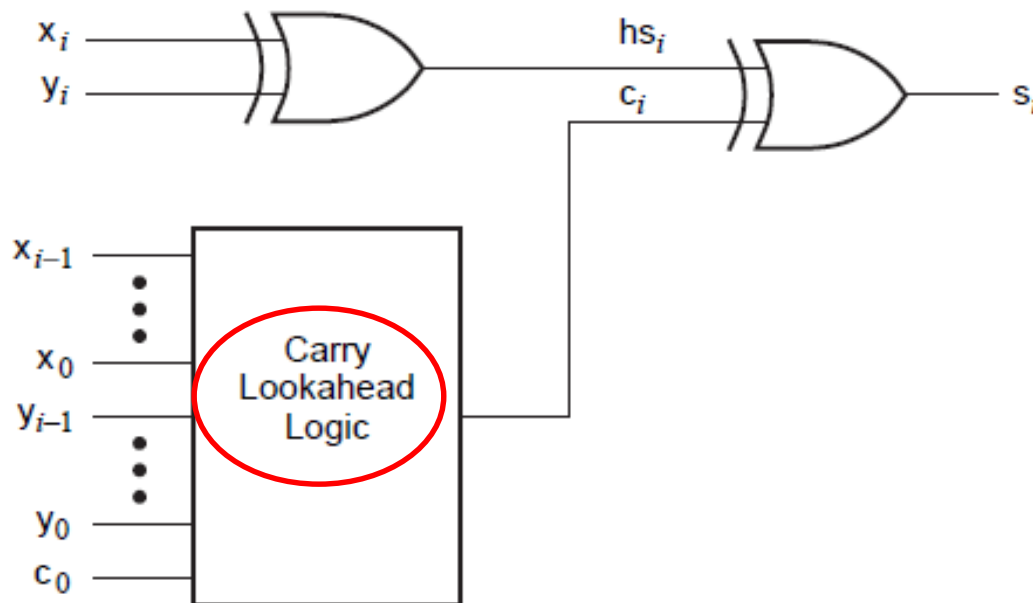
Hiệu năng



- Tốc độ của mạch bị giới hạn bởi độ trễ lớn nhất dọc theo đường nối trong mạch
 - Độ trễ lớn nhất được gọi là **critical-path-delay**
 - Đường nối gây ra độ trễ đó gọi là **critical path**

Carry Look-Ahead Adder (CLA)

- Cải thiện tốc độ mạch cộng bằng cách
 - Tại mỗi tầng (stage), ta sẽ xác định nhanh giá trị **carry-in** ở tầng cộng trước đó sẽ có giá trị 0 hay 1
- giảm critical-path-delay



Carry Look-Ahead Adder (CLA)

- Hàm xác định carry-out ở lần cộng thứ i

$$C_{i+1} = x_i y_i + x_i C_i + y_i C_i = x_i y_i + (x_i + y_i) C_i$$

- Đặt $g_i = x_i y_i$ và $p_i = x_i + y_i \Rightarrow C_{i+1} = g_i + p_i C_i$

➤ $g_i = 1$ khi cả x_i và y_i đều bằng 1, không quan tâm C_i

❖ g được gọi là **hàm generate**, vì carry-out luôn được generate ra khi $g=1$

➤ $p_i = 1$ khi $x_i = 1$ hoặc $y_i = 1$; carry-out = C_i

❖ p được gọi là **hàm propagate**, vì carry-in = 1 được propagate (truyền) ở tầng cộng thứ i

Carry Look-Ahead Adder (CLA)

- Xác định carry-out của mạch cộng n bit

$$C_n = g_{n-1} + p_{n-1}C_{n-1}$$

Mà $C_{n-1} = g_{n-2} + p_{n-2}C_{n-2}$

$$C_n = g_{n-1} + p_{n-1}(g_{n-2} + p_{n-2}C_{n-2})$$

$$C_n = g_{n-1} + p_{n-1}g_{n-2} + p_{n-1}p_{n-2}C_{n-2}$$

- Tiếp tục khai triển đến lần cộng đầu tiên

$$C_n = g_{n-1} + p_{n-1}g_{n-2} + p_{n-1}p_{n-2}g_{n-3} + \dots + p_{n-1}p_{n-2}\dots p_1g_0 + p_{n-1}p_{n-2}\dots p_1p_0C_0$$

Carry Look-Ahead Adder (CLA)

Số nhớ sinh ra ở lần cộng thứ **n-2** và được truyền qua các lần cộng còn lại

Số nhớ sinh ra ở lần cộng thứ 1 và được truyền qua các lần cộng còn lại

$$c_n = \underbrace{g_{n-1}}_{\substack{\uparrow \\ \text{Số nhớ sinh ra ở} \\ \text{lần cộng cuối cùng}}} + \underbrace{p_{n-1}g_{n-2}}_{\substack{\uparrow \\ \text{Số nhớ sinh ra ở lần cộng} \\ \text{thứ } n-3 \text{ và được truyền qua} \\ \text{các lần cộng còn lại}}} + \underbrace{p_{n-1}p_{n-2}g_{n-3}}_{\substack{\uparrow \\ \text{Số nhớ sinh ra ở lần cộng} \\ \text{thứ } n-3 \text{ và được truyền qua} \\ \text{các lần cộng còn lại}}} + \dots + \underbrace{p_{n-1}p_{n-2}\dots p_1g_0}_{\substack{\uparrow \\ \text{Số nhớ sinh ra ở lần cộng} \\ \text{thứ } n-3 \text{ và được truyền qua} \\ \text{các lần cộng còn lại}}} + \underbrace{p_{n-1}p_{n-2}\dots p_1p_0c_0}_{\substack{\uparrow \\ \text{Số nhớ đầu vào } c_0 \\ \text{được truyền qua} \\ \text{tất cả các lần cộng}}}$$

Số nhớ sinh ra ở lần cộng cuối cùng

Số nhớ sinh ra ở lần cộng thứ **n-3** và được truyền qua các lần cộng còn lại

Số nhớ đầu vào c_0 được truyền qua tất cả các lần cộng

Carry Look-Ahead Adder (CLA)

- Ví dụ: Trường hợp cộng 4 bit

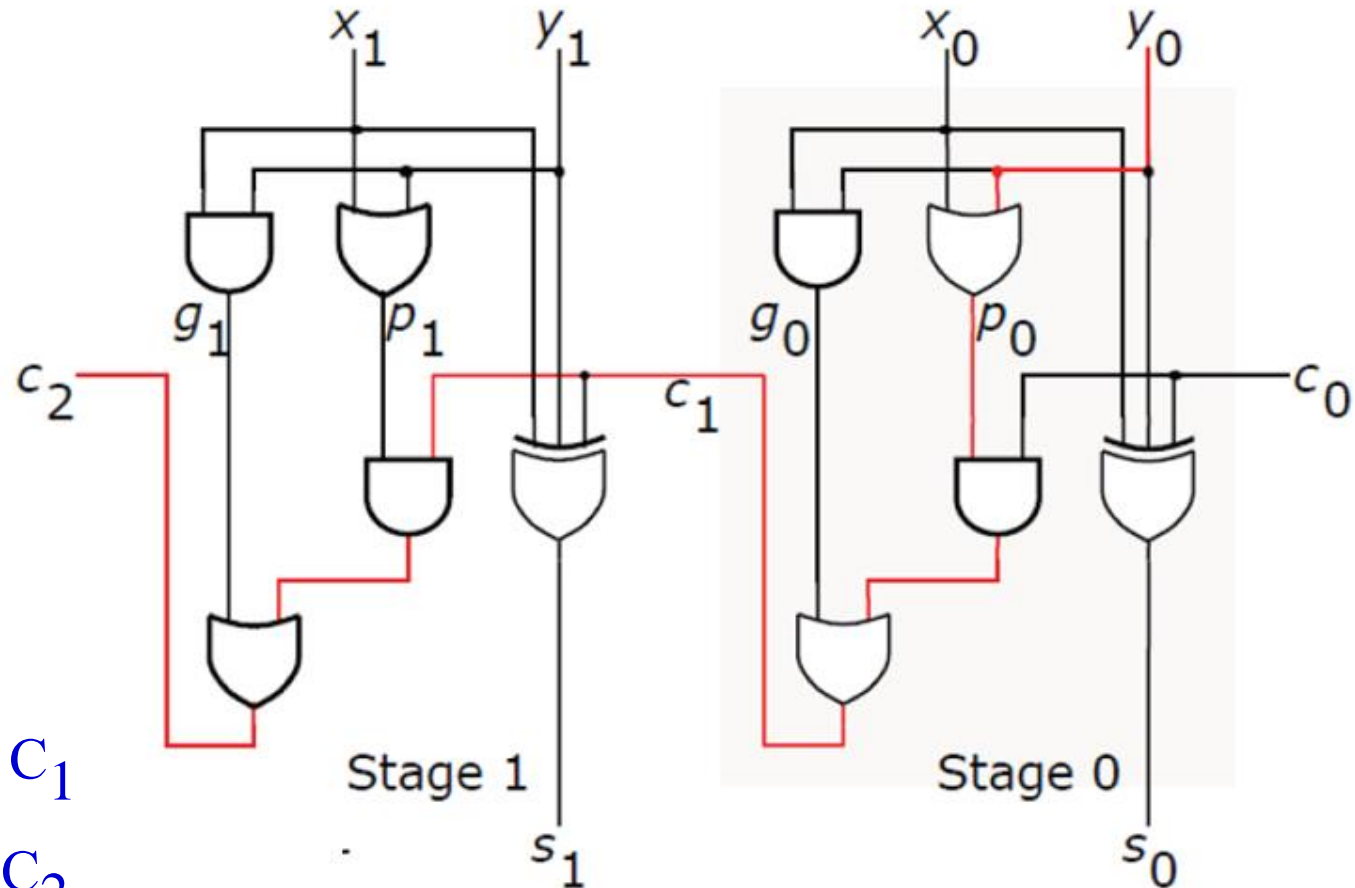
$$C_1 = G_0 + P_0.C_0$$

$$C_2 = G_1 + P_1.G_0 + P_1.P_0.C_0$$

$$C_3 = G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.C_0$$

$$C_4 = G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3.P_2.P_1.G_0 + P_3.P_2.P_1.P_0.C_0$$

Mạch cộng Carry Ripple - critical path



Độ trễ 3 cổng đối với C_1

Độ trễ 5 cổng đối với C_2

Tổng quát, độ trễ **$2n+1$** cổng đối
với mạch cộng Carry Ripple **n -bit**

Mạch cộng CLA - critical path

$$C_1 = G_0 + P_0.C_0$$

$$C_2 = G_1 + P_1.G_0 + P_1.P_0.C_0$$

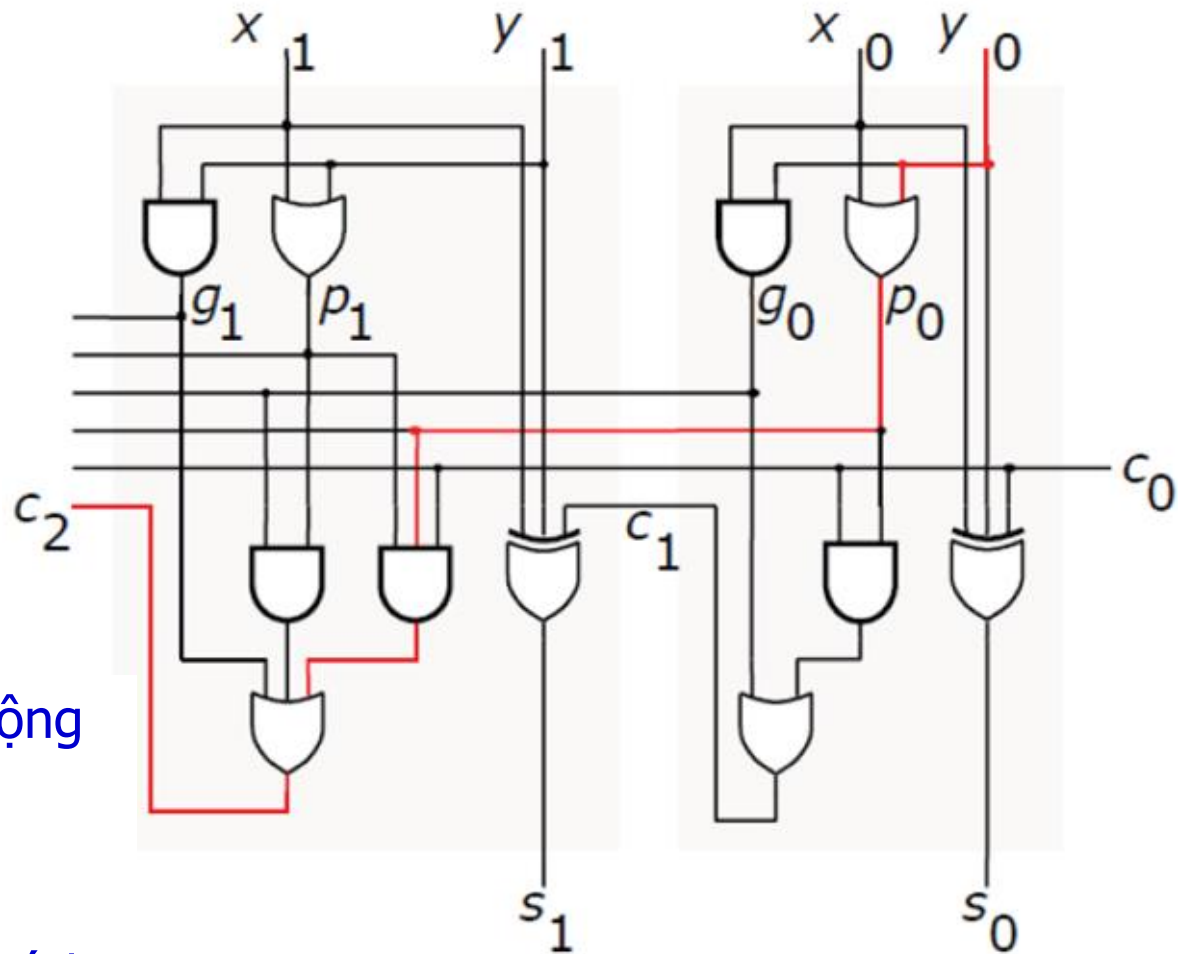
Độ trễ 3 cổng đối với C_1

Độ trễ 3 cổng đối với C_2

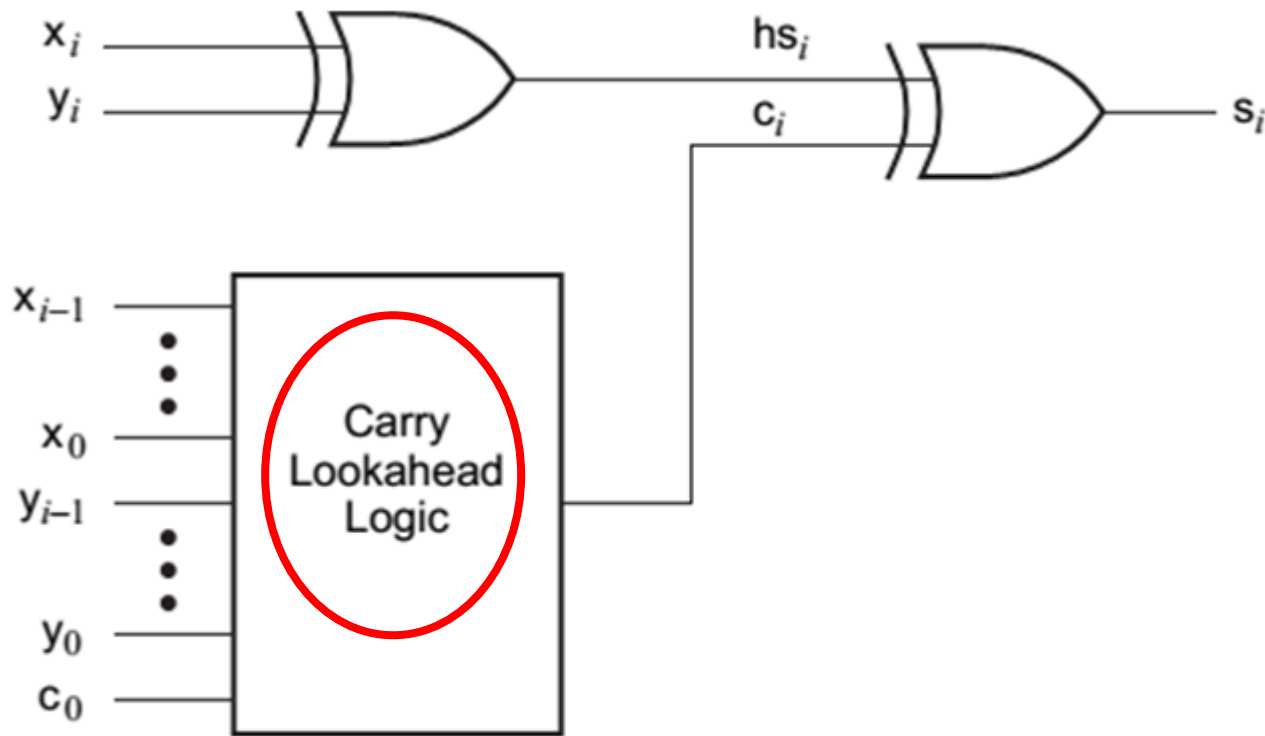
Độ trễ 3 cổng đối với C_n

Độ trễ tổng cộng cho mạch cộng
CLA n-bit là độ trễ **4 cổng**

- g_i, p_i : độ trễ 1 cổng
- C_i : độ trễ 2 cổng
- Độ trễ 1 cổng còn lại là do tính tổng s



Mạch cộng CLA



Cấu trúc của **một tầng** của mạch cộng CLA

Giới hạn của mạch cộng CLA

- Biểu thức tính carry trong mạch cộng CLA

$$c_n = g_{n-1} + p_{n-1}g_{n-2} + p_{n-1}p_{n-2}g_{n-3} + \dots + p_{n-1}p_{n-2}\dots p_1g_0 + p_{n-1}p_{n-2}\dots p_1p_0c_0$$

CLA là giải pháp tốc độ cao (2 level AND-OR)

- Độ phức tạp tăng lên nhanh chóng khi n lớn
- Vấn đề Fan-in có thể hạn chế tốc độ của mạch cộng CLA

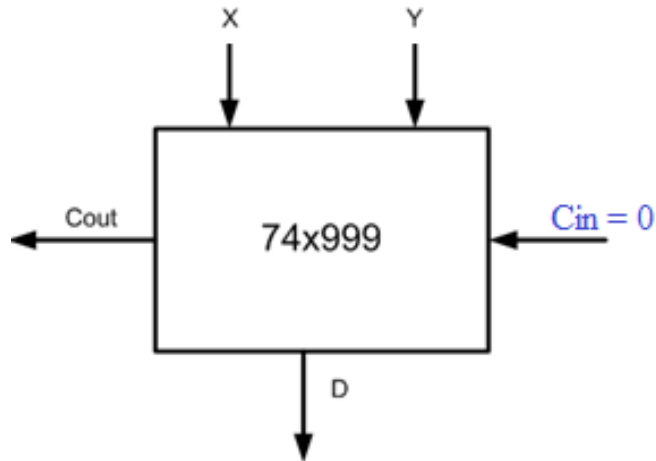


3 Adder/ Subtractor

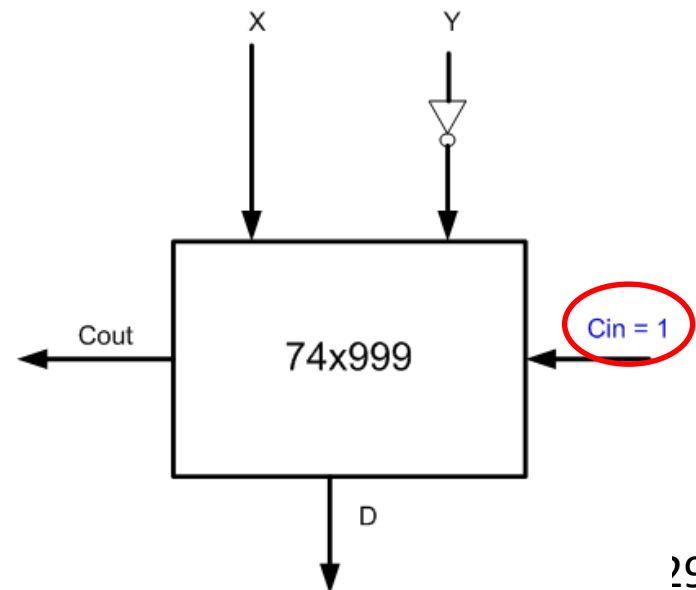
Mạch cộng/ trừ

- X, Y là 2 số không dấu n-bit

Phép cộng: $S = X + Y$

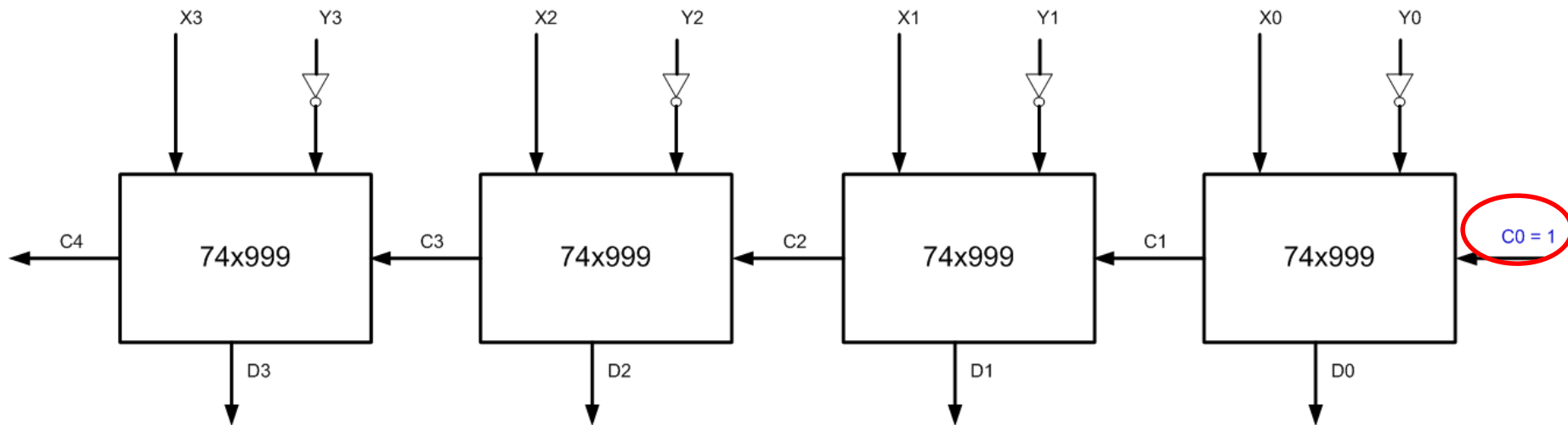


Phép trừ: $D = X - Y$
 $= X + (-Y)$
 $= X + (\text{Bù 2 của } Y)$
 $= X + (\text{Bù 1 của } Y) + 1$
 $= X + Y' + 1$



Mạch trừ

- Mạch cộng Carry Ripple có thể được dùng để xây dựng mạch trừ Carry Ripple bằng cách đảo Y và đặt số nhớ đầu tiên là 1



Tràn (Arithmetic Overflow)

- **Overflow** là khi kết quả của phép toán vượt quá số bit biểu diễn phần giá trị
 - n bit biểu diễn được số từ -2^{n-1} đến $+2^{n-1}-1$
 - Overflow luôn luôn cho ra 1 kết quả sai

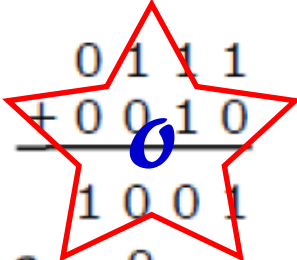
+9 →	0	1001
+8 →	0	1000
<hr/>		
	1	0001

incorrect sign ↑ incorrect magnitude

⇒ Mạch để xác định có overflow hay không

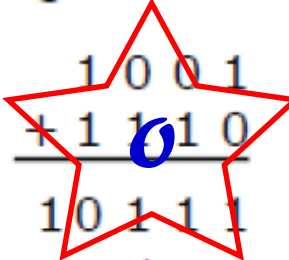
Ví dụ về arithmetic overflow

- Với số 4 bit, 3 bit giá trị và 1 bit dấu

$$\begin{array}{r}
 (+7) \quad 0 \ 1 \ 1 \ 1 \\
 + (+2) \quad + \ 0 \ 0 \ 1 \ 0 \\
 \hline
 (+9) \quad 1 \ 0 \ 0 \ 1 \\
 c_4 = 0 \\
 c_3 = 1
 \end{array}$$


$$\begin{array}{r}
 (+7) \quad 0 \ 1 \ 1 \ 1 \\
 + (-2) \quad + \ 1 \ 1 \ 1 \ 0 \\
 \hline
 (+5) \quad 1 \ 0 \ 1 \ 0 \ 1 \\
 c_4 = 1 \\
 c_3 = 1
 \end{array}$$

$$\begin{array}{r}
 (-7) \quad 1 \ 0 \ 0 \ 1 \\
 + (+2) \quad + \ 0 \ 0 \ 1 \ 0 \\
 \hline
 (-5) \quad 1 \ 0 \ 1 \ 1 \\
 c_4 = 0 \\
 c_3 = 0
 \end{array}$$

$$\begin{array}{r}
 (-7) \quad 1 \ 0 \ 0 \ 1 \\
 + (-2) \quad + \ 1 \ 1 \ 1 \ 0 \\
 \hline
 (-9) \quad 1 \ 0 \ 1 \ 1 \ 1 \\
 c_4 = 1 \\
 c_3 = 0
 \end{array}$$


- Overflow không xuất hiện khi cộng 2 số trái dấu

Arithmetic overflow

- Overflow có thể phát hiện được (từ ví dụ ở slide trước)

$$\text{Overflow} = c_3 \overline{c_4} + \overline{c_3} c_4$$

$$\text{Overflow} = c_3 \oplus c_4$$

- Với n bit

$$\text{Overflow} = c_{n-1} \oplus c_n$$

- Mạch cộng/ trừ có thể bổ sung mạch kiểm tra overflow với 1 cổng XOR

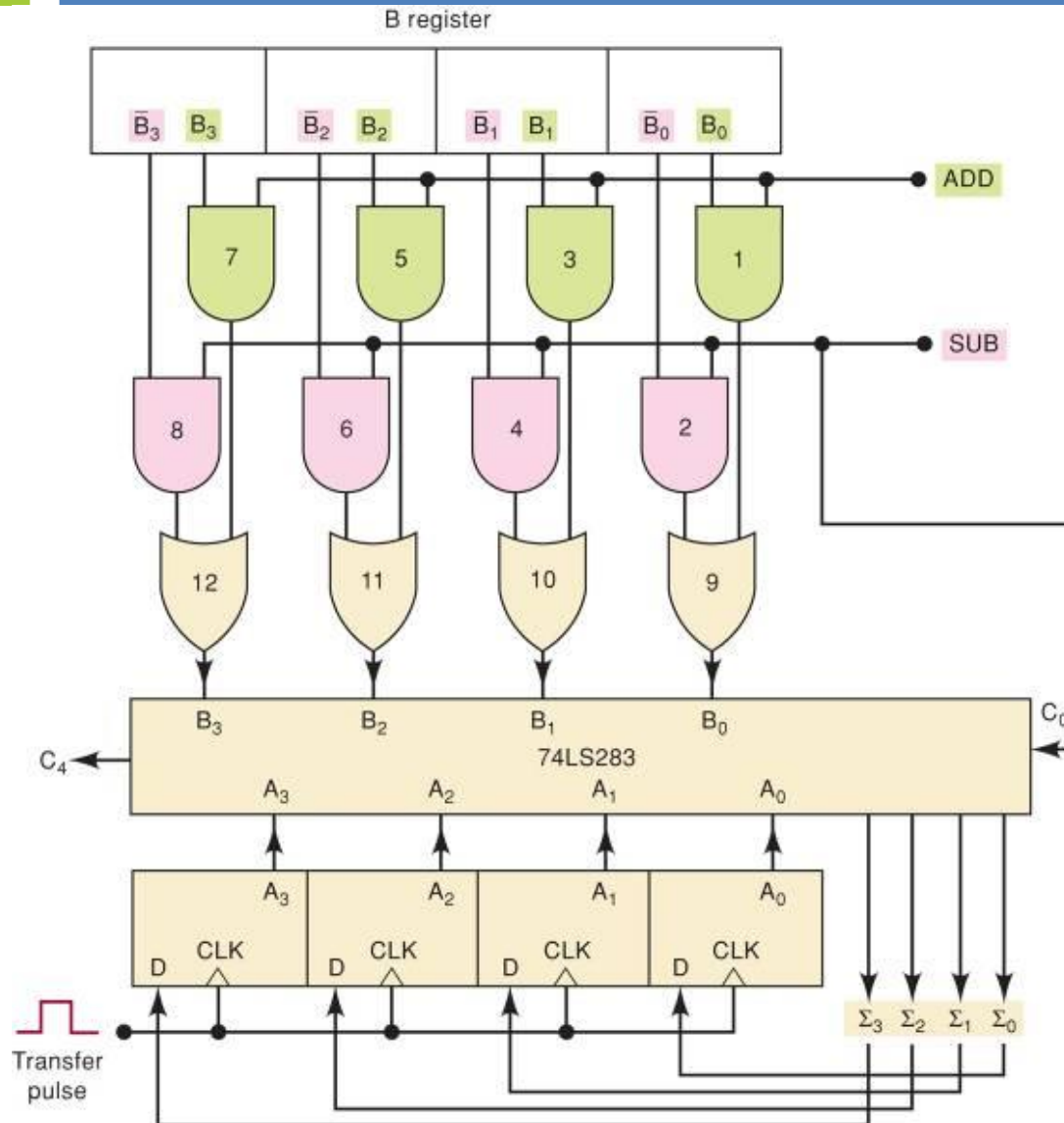
Ví dụ

- Thiết kế một mạch cộng/ trừ với 2 ngõ điều khiển ADD và SUB
 - $ADD = 1$: mạch cộng 2 số trong 2 thanh ghi A và B
 - $SUB = 1$: mạch thực hiện phép trừ số B-A

Chú ý:

Trong một lúc chỉ một trong hai ngõ ADD, SUB bằng 1

Ví dụ

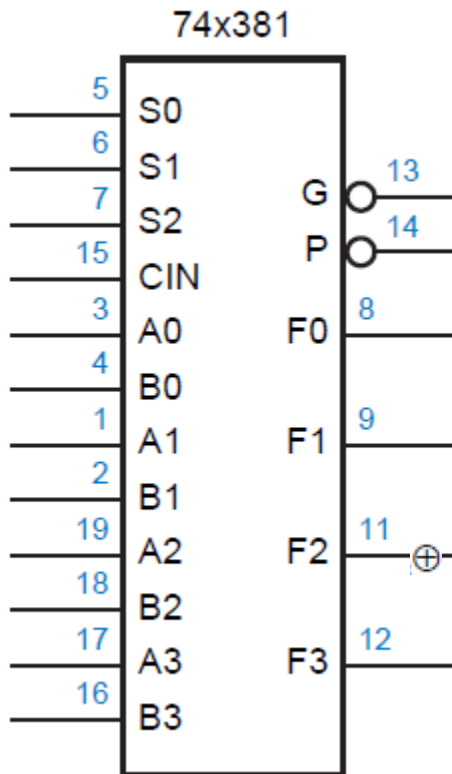




4 Arithmetic Logic Unit (ALU)

ALU

- ALUs có thể thực thi nhiều toán tử và hàm logic khác nhau
 - Các toán tử và hàm được xác định bởi một mã ngõ vào



Inputs			Function
S2	S1	S0	
0	0	0	$F = 0000$
0	0	1	$F = B - A - 1 + Cin$
0	1	0	$F = A - B - 1 + Cin$
0	1	1	$F = A + B + Cin$
1	0	0	$F = A \oplus B$
1	0	1	$F = A + B$
1	1	0	$F = A * B$
1	1	1	$F = 1111$



Any question?