

Kiến trúc máy tính

Số học máy tính

NGUYỄN Ngọc Hoá

Bộ môn Hệ thống thông tin, Khoa CNTT

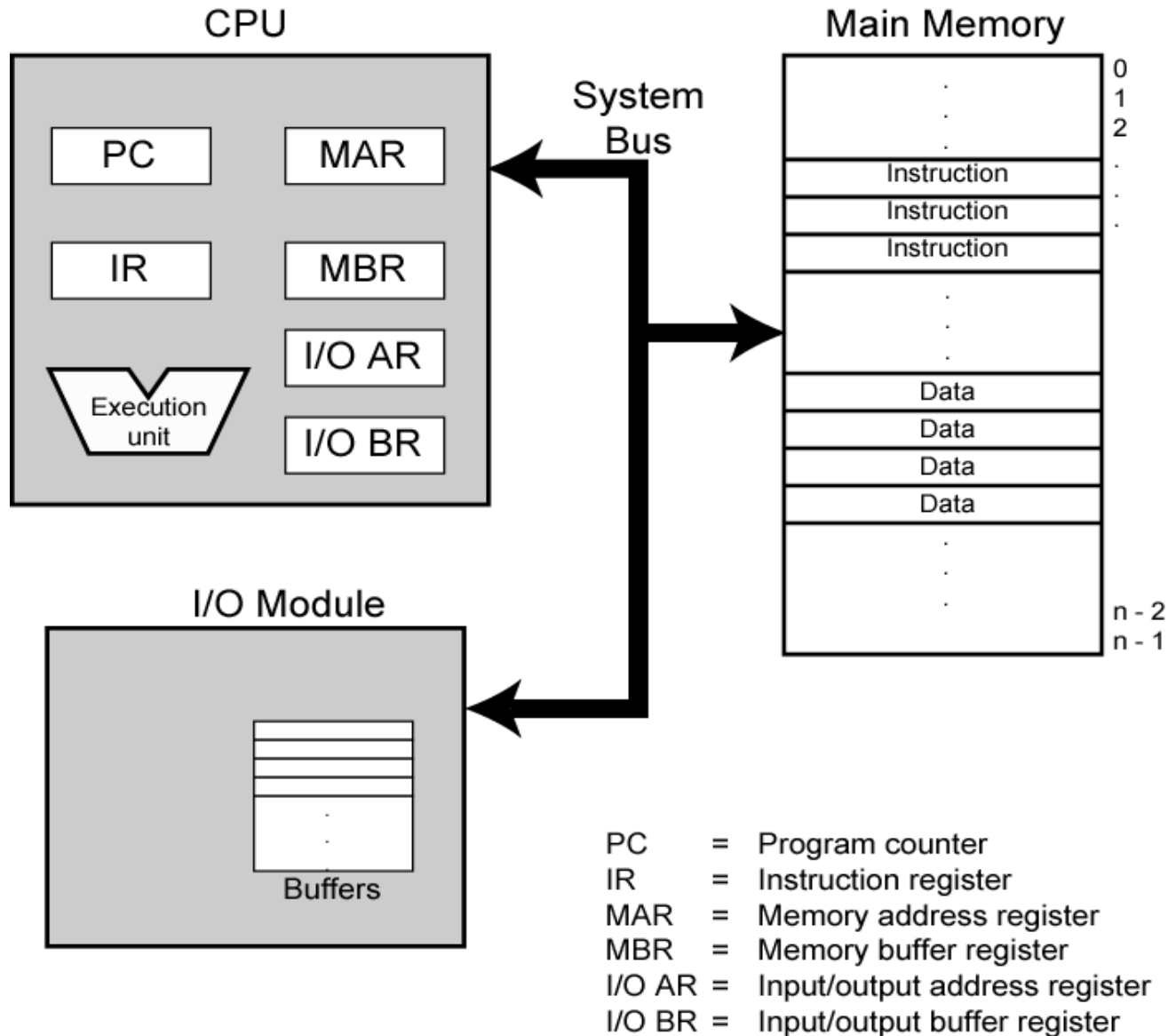
Trường Đại học Công nghệ,

Đại học Quốc gia Hà Nội

Nội dung

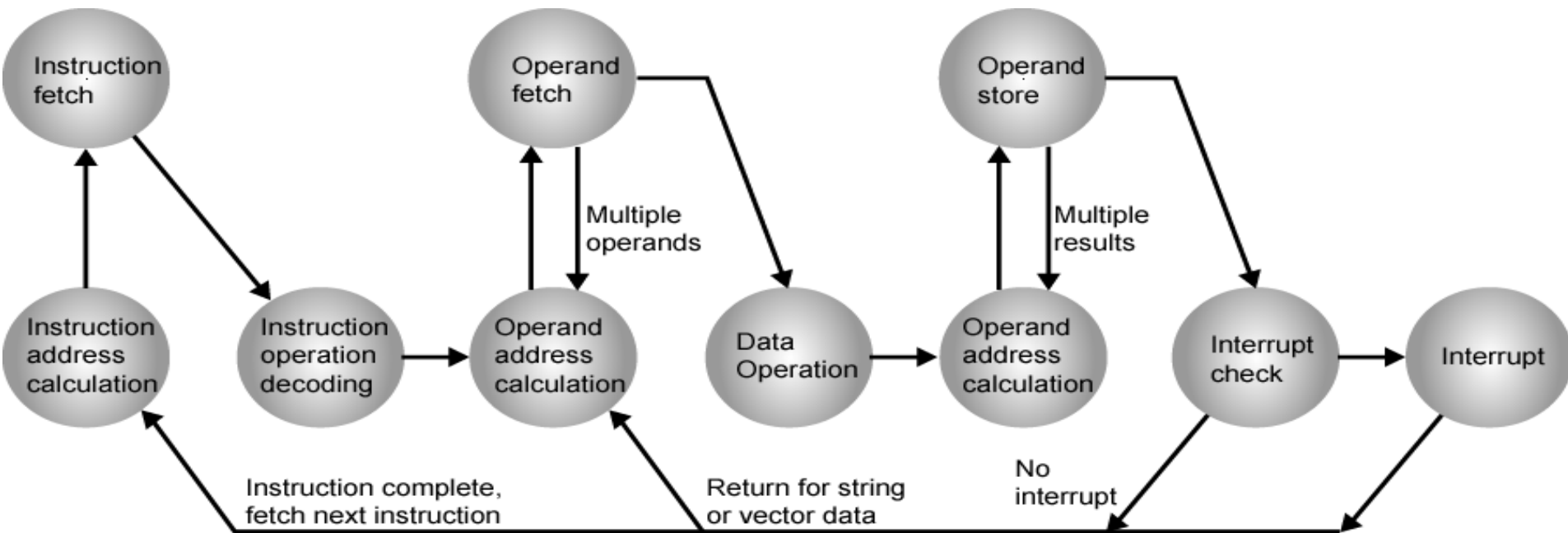
- Tổng quan về CPU
- Biểu diễn thông tin số
 - Khái niệm thông tin số
 - Biểu diễn ký tự
 - Biểu diễn số nguyên
 - Biểu diễn số thực
- Logic số
 - Mạch kết hợp
 - Bộ số học và logic
 - Mạch tuần tự

Kiến trúc tổng quan



Chức năng máy tính

- Thực thi chương trình, đã được xây dựng thông qua tập các lệnh của CPU, lưu trong bộ nhớ
- Các bước chính khi thực thi chương trình trong CPU
 - Tải lệnh từ bộ nhớ (fetch)
 - Thực thi lệnh (execute)
 - Lưu kết quả (store)



Khái niệm thông tin

- Thông tin số: tri thức về một trạng thái trong số một số hữu hạn các trạng thái có thể có
- Lượng tử thông tin:
 - 1 bit là đại lượng thông tin gắn với tri thức của một trạng thái trong số hai.
 - 1 bit thông tin : được biểu diễn bởi số nhị phân 0,1
 - N bits $\rightarrow 2^n$ trạng thái khác nhau
 - Lượng thông tin chứa trong tri thức của một trạng thái trong số N là $I = \lceil \log_2 N \rceil$
 - Độ lớn thông tin mà máy tính có thể thao tác: 8, 16, 32, 64 bits

Mã hoá

$$I = \{i_1, \dots, i_m\}$$

Tập các thông tin

$$A = \{a_1, \dots, a_n\}$$

Bộ ký tự

- a_i : ký tự của A
- $a_1 a_3 a_4 a_8$: từ của A
- $|A|$: cơ số mã hoá

→ Mã hoá I : gán mỗi phần tử của I với một từ của A

Đặc điểm

- Dư thừa: 1 phần tử được gán với nhiều từ (mã)
 - Dư thừa: Số điện thoại cố định
 - Không dư thừa: Số chứng minh thư
- Độ dài:
 - Thay đổi: tín hiệu morse
 - Cố định: số điện thoại di động
- Với bộ mã độ dài cố định n , cơ số mã hoá b :
 - Có thể biểu diễn được b^n phần tử và
 - Có $b^n!$ cách mã hoá khác nhau

Một vài bộ mã

■ Biểu diễn số:

- Cần phân biệt số và cách thể hiện số.
- Thể hiện một số là một cách mã hoá
- Với cơ số b , ta có

$$a_n a_{n-1} \dots a_1 a_0 = \sum_0^n a_i \times b^i$$

- Mã nhị phân: $A = \{0, 1\}$

- VD: $7 = (111)_2$

- Mã hexa: $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$

- Mã DCB (Decimal Coded Binary): Mỗi chữ số được mã hoá nhị phân bằng 4 bits:

0 : 0000

1 : 0001

2 : 0010

...

10 : 0001 0000

25 : 0010 0101

...

Chuyển cơ số

■ Từ cơ số b về 10

- $a_n a_{n-1} \dots a_1 a_0$ với cơ số b (ký hiệu $a_n a_{n-1} \dots a_1 a_0 b$) :

$$a_n \times b^n + a_{n-1} \times b^{n-1} + \dots + a_1 \times b + a_0$$

- Phần phân:

$$a_1 \times b^{-1} + a_2 \times b^{-2} + \dots + a_n \times b^{-n}$$

■ Từ cơ số 10 về cơ số b

- A là số nguyên:

$$A_{10} = a_n \times b^n + a_{n-1} \times b^{n-1} + \dots + a_1 \times b + a_0$$

$$= ((\dots (a_n \times b + a_{n-1}) \times b + \dots) \times b + a_1) \times b + a_0$$

với a_0 là phần dư của phép chia của A với cơ số b

- A là phần phân

$$A_{10} = a_1 \times b^{-1} + a_2 \times b^{-2} + \dots + a_n \times b^{-n}$$

$$= (a_1 + (a_2 + (\dots + (a_{n-1} + a_n \times b^{-1})b^{-1} \dots)b^{-1})b^{-1})b^{-1}$$

với a_1 là phần nguyên của phép nhân A với b

Nguyên lý chuyển

■ Phần nguyên:

- Chia liên tiếp với cơ số
- Sử dụng phần dư

$$25_{10} / 2 = 12_{10} \text{ dư } 1$$

$$12_{10} / 2 = 6_{10} \text{ dư } 0$$

$$6_{10} / 2 = 3_{10} \text{ dư } 0$$

$$3_{10} / 2 = 1_{10} \text{ dư } 1$$

$$1_{10} / 2 = 0_{10} \text{ dư } 1$$

Vậy

$$25_{10} = 11001_2$$

■ Phần phân:

- Nhân liên tiếp với cơ số
- Sử dụng phần nguyên

$$0,78125_{10} \times 2 = 1,5625_{10} \text{ phần nguyên } 1$$

$$0,5625_{10} \times 2 = 1,125_{10} \text{ phần nguyên } 1$$

$$0,125_{10} \times 2 = 0,25_{10} \text{ phần nguyên } 0$$

$$0,25_{10} \times 2 = 0,5_{10} \text{ phần nguyên } 0$$

$$0,5_{10} \times 2 = 1_{10} \text{ phần nguyên } 1$$

Vậy

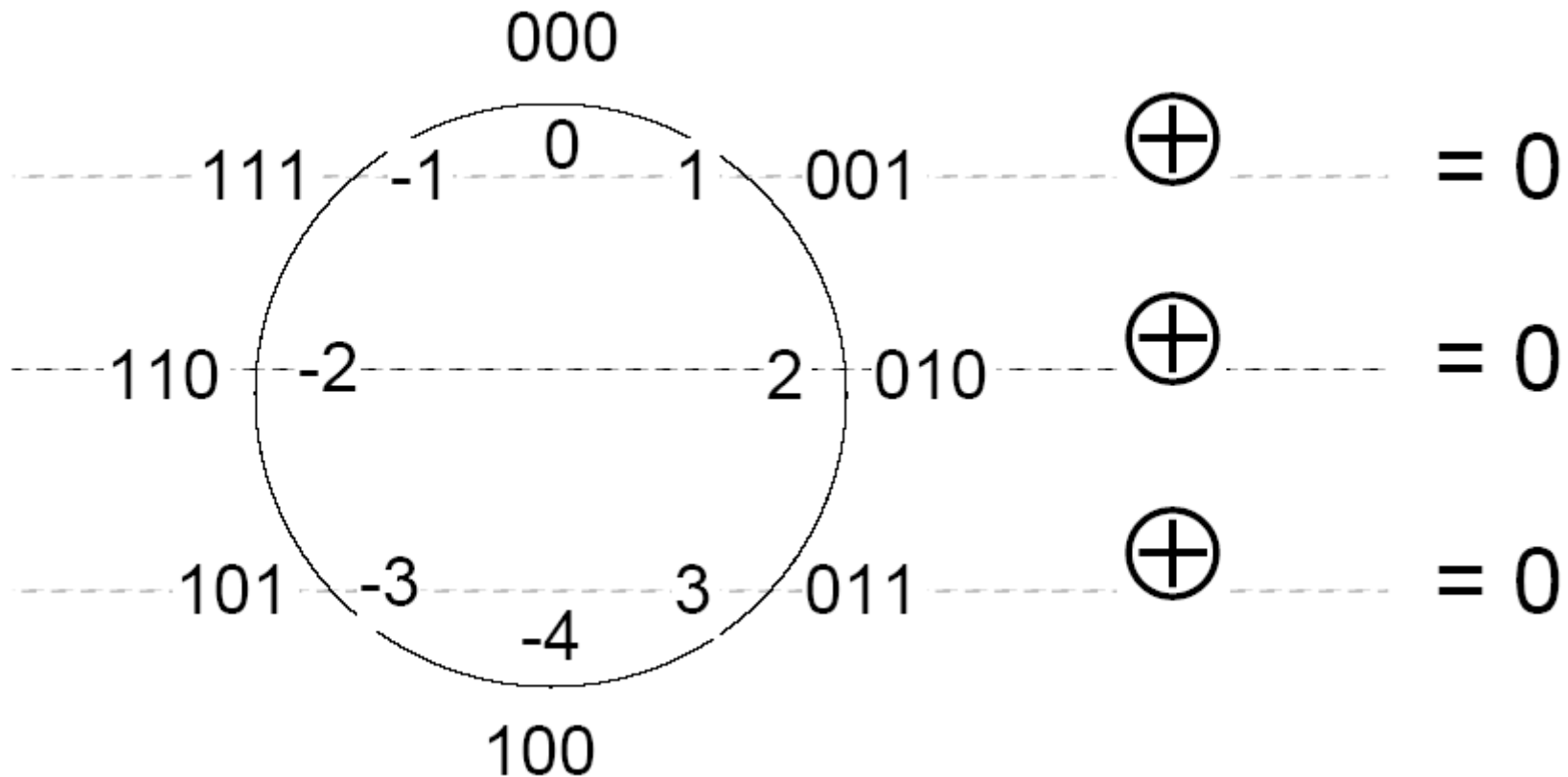
$$0,78125_{10} = 0,11001_2$$

Biểu diễn ký tự

- ASCII (American Standard Code for Information Interchange) : sử dụng 1 byte để mã hoá ký tự
 - ANSI: 7 bits
 - A: 41_H
 - 9: 39_H
 - ISO-8859: 8 bits để biểu diễn những ký tự có dấu (Ê : CA_H)
- Unicode:
 - Biểu diễn 1 ký tự thông qua 2 bytes
 - Được sử dụng để biểu diễn những ký tự không phải latin
 - ~ UCS (ISO 10646)

Biểu diễn số nguyên

- Số tự nhiên: sử dụng cơ số 2 để biểu diễn
 - Với n bits, ta có thể biểu diễn được những số tự nhiên N trong khoảng $[0, 2^n - 1]$
- Số nguyên:



Biểu diễn số nguyên

- Dấu và giá trị tuyệt đối : với n bits
 - Dấu: bit phải nhất (0 : dương, 1 : âm)
 - Giá trị tuyệt đối: n – 1 bits
 - Khoảng giá trị biểu diễn: $[-2^{n-1} + 1, 2^{n-1} - 1]$
- Bù 1: với n bits
 - Đảo bit của giá trị tuyệt đối
 - $|x| + (-|x|) = 2^n - 1$
 - Khoảng giá trị biểu diễn: $[-2^{n-1} + 1, 2^{n-1} - 1]$
- Bù 2: với n bits
 - Bù 1 + 1
 - $|x| + (-|x|) = 2^n$
 - Khoảng giá trị biểu diễn: $[-2^{n-1}, 2^{n-1} - 1]$

	Với 3 bits: [-3,3]
000	0
001	1
010	2
011	3
100	0
101	1
110	2
111	3
	Với 3 bits: [-4,3]
000	0
001	1
010	2
011	3
100	-3
101	-2
110	-1
111	-0

Biểu diễn số nguyên

■ Dư: với n bits

- Thêm giá trị dư
- Thường dư được lấy $= 2^{n-1}$, và $-|x| = 2^{n-1} - |x|$
- Khoảng giá trị biểu diễn: $[-2^{n-1}, 2^{n-1} - 1]$
- $x < 0$ có thể biểu diễn được nếu $x \geq$ giá trị dư

Với 3 bits, dư $2^2=4$: $[-4,3]$

000	-4
001	-3
010	-2
011	-1
100	0
101	1
110	2
111	3

Biểu diễn số thực

- Một số thực $\pm m \times b^e$ được biểu diễn bởi:
 - Dấu \pm
 - Phần định trị m
 - Phần mũ e
 - Cơ sở b
- ➔ có vô số cách biểu diễn có thể có với một số thực
- Chuẩn hoá: chỉ dùng một chữ số khác 0 trước dấu phẩy
- Khó khăn:
 - Giới hạn số chữ số mà máy tính có thể xử lý được ➔ làm tròn
 - Tiêu chuẩn chính xác (cách làm tròn), xử lý số quá lớn/quá nhỏ
 - VD: IEEE 754 xuất hiện 1977 nhưng đến 1985 mới được công nhận

Chuẩn IEEE754

Chuẩn đơn:

- $e = E_{10} - 127$
- $e \in [-127, 128]$

1	8	23
s	E (mũ)	f (định trị)

Chuẩn kép:

- $e = E_{10} - 1023$
- $e \in [-1023, 1024]$

1	11	52
s	E (mũ)	f (định trị)

Giá trị biểu diễn

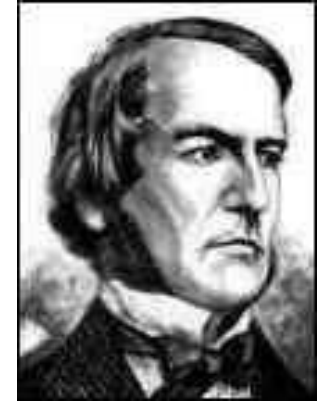
	e	f	Giá trị
Chuẩn	$e_{\min} < e < e_{\max}$	f	$(-1)^s \times 1, f \times 2^e$
Không chuẩn	$e = e_{\min}$	≠0	$(-1)^s \times 0, f \times 2^e$
Zero	$e = e_{\min}$	0	$(-1)^s \times 0$
Vô cùng	$e = e_{\max}$	0	$(-1)^s \times \infty$
NaN	$e = e_{\max}$	≠0	NaN

http://en.wikipedia.org/wiki/IEEE_floating-point_standard

Đại số Boole

- Đề xuất bởi Georges Boole (1815-1864) với :

- Một tập E
- Hai phần tử đặc biệt của E : 0 và 1
- Hai phép toán nhị nguyên trên E : + và .
- Một phép toán đơn nguyên trên E : -



- Tiên đề : cho $a, b \in E$

- Giao hoán: $a+b = b+a$ $ab = ba$
- Kết hợp: $(a+b)+c = a+(b+c)$ $(ab)c = a(bc)$
- Phân phối: $a(b+c) = ab+ac$ $a+(bc) = (a+b)(a+c)$
- Phần tử trung hoà: $a+0 = a$ $a1 = a$
- Bù: $a + \bar{a} = 1$ $a\bar{a} = 0$

Đại số Boole

■ Định lý:

- Dư thừa: $a+a = a$ $aa = a$
- Phần tử hấp thụ: $a+1 = 1$ $a0 = 0$
- Hấp thụ: $a+ab = a$ $a(a+b) = a$
- De Morgan: $\overline{a + b} = \overline{a} \cdot \overline{b}$ $\overline{ab} = \overline{a} + \overline{b}$

■ Chứng minh:

- $aa = aa + 0 = aa + a\bar{a} = a(a + \bar{a}) = a1 = a$
- $a+a = (a+a)(a+a) = (aa+aa)+(aa+aa) = aa + aa \rightarrow a+a = a$
- $a + 1 = a + a + \bar{a} = a + \bar{a} = 1$
- $a0 = aa\bar{a} = a\bar{a} = 0$
- $a+ab = a(1+b) = a1 = 1$
- $a(a+b) = aa + ab = a + ab = a$

Đại số Boole tối thiểu

- $E = \{0,1\}$ và ta có:
 - 1 : “đúng”
 - 0 : “sai”
 - + : “hoặc” (hợp)
 - . : “và” (giao)
 - - : “Not” (phủ định)
- Bảng chân lý: miêu tả một phép toán logic

a	\bar{a}	a	b	a+b	a	b	ab	a	b	$a \oplus b$
0	1	0	0	0	0	0	0	0	0	0
1	0	0	1	1	0	1	0	0	1	1
		1	0	1	1	0	0	1	0	1
		1	1	1	1	1	1	1	1	0

Hàm boolean

- Hàm nhị phân của các biến nhị phân : $\{0,1\}^n \rightarrow \{0,1\}$
- Thể hiện:
 - Bảng chân lý
 - Biểu thức boolean
- Chuyển từ bảng chân lý sang biểu thức logic
 - Tổng nhân:

$$F(a,b,c) = \bar{a}bc + a\bar{b}c + abc\bar{c}$$

- Nhân tổng:

$$F(a,b,c) = (a+b+c)(a+b+\bar{c})$$

$$(a+\bar{b}+c)(\bar{a}+b+c)(\bar{a}+\bar{b}+\bar{c})$$

a	b	c	F(a,b,c)
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	0

Đơn giản hóa biểu thức boolean

- Sử dụng các tiên đề và định lý trong đại số Bool

- Ví dụ:
$$Z = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$
$$= (\bar{A}BC + ABC) + (A\bar{B}C + ABC) + (AB\bar{C} + ABC)$$
$$= BC + AC + AB$$

- Yếu điểm: Khó có thể khẳng định biểu thức cuối cùng là tối ưu nhất hay chưa

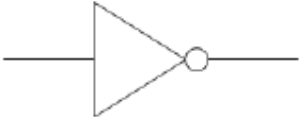
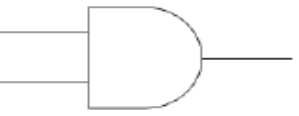
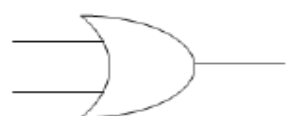
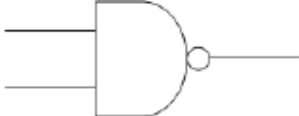
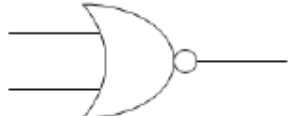

- Sử dụng bảng Karnaugh:

B\A	0	1
0	1	1
1	1	0

BC\A	0	1
00	0	0
01	0	1
11	1	1
10	0	1

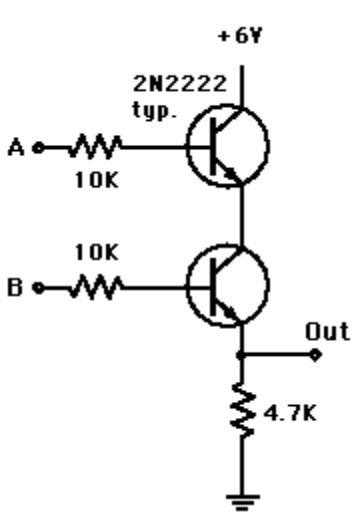
Mạch logic

- Những phép toán trong đại số Boole được thực hiện thông qua các mạch logics cơ bản, được gọi là các **cổng logics**
- Cổng logics cơ bản

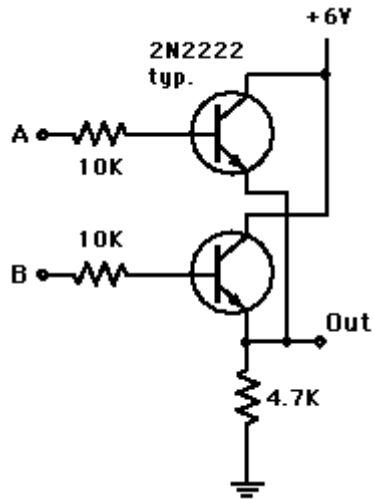
	NOT
	AND
	OR
	NAND
	NOR
	XOR

Transistor Gates

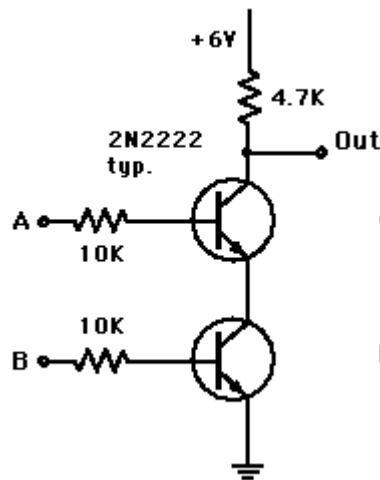
AND



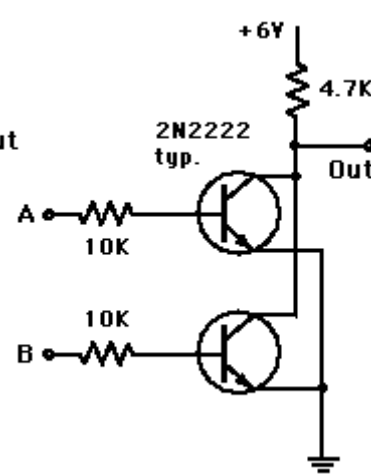
OR



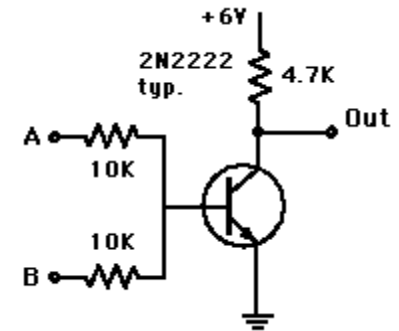
NAND



NOR



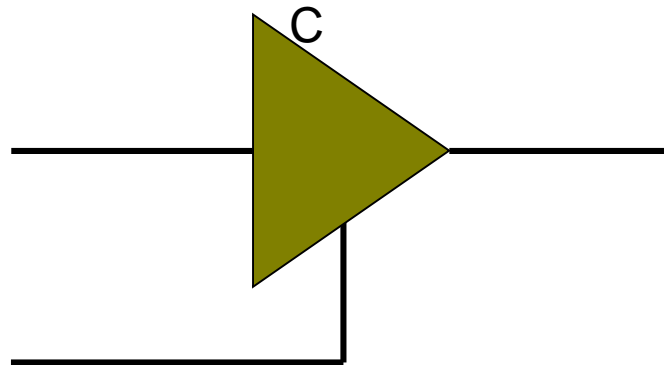
NOR



Mạch logic

- Cổng 3 trạng thái

C	A	Y
1	0	0
1	1	1
0	X	Treo



Mạch logic

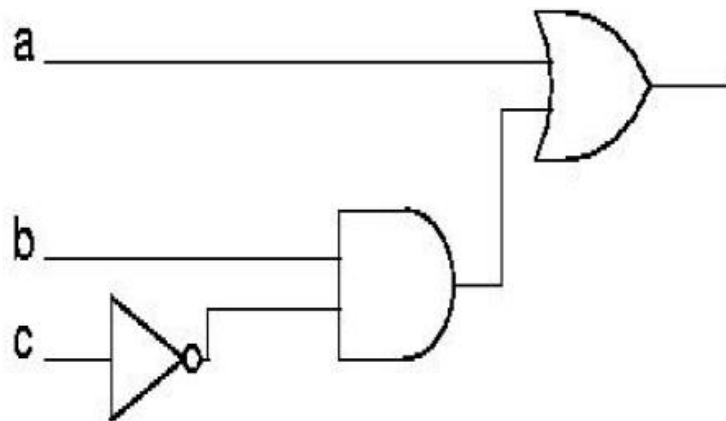
■ NAND và NOR

- Đầy đủ: cho phép xây dựng được bất kỳ hàm boolean
- Dễ sản xuất
- Là thành phần cơ bản của hầu hết các mạch in trong các máy tính hiện nay

■ Biểu thức boolean:

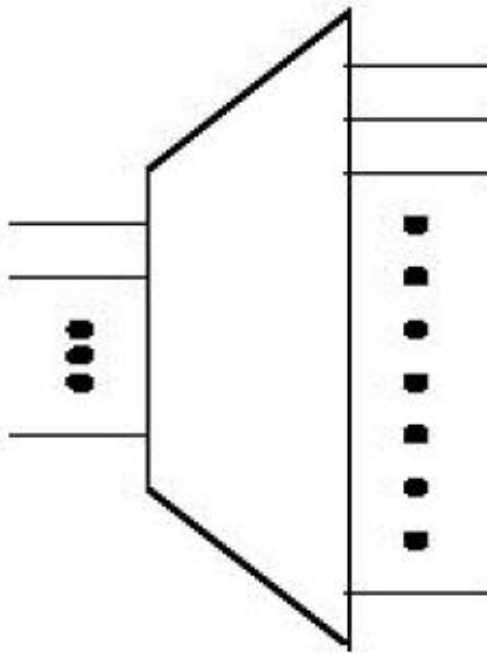
- Có thể được thực hiện thông qua các cổng logic cơ bản
- Ví dụ:

$$a + b\bar{c}$$



Mạch logic tổ hợp

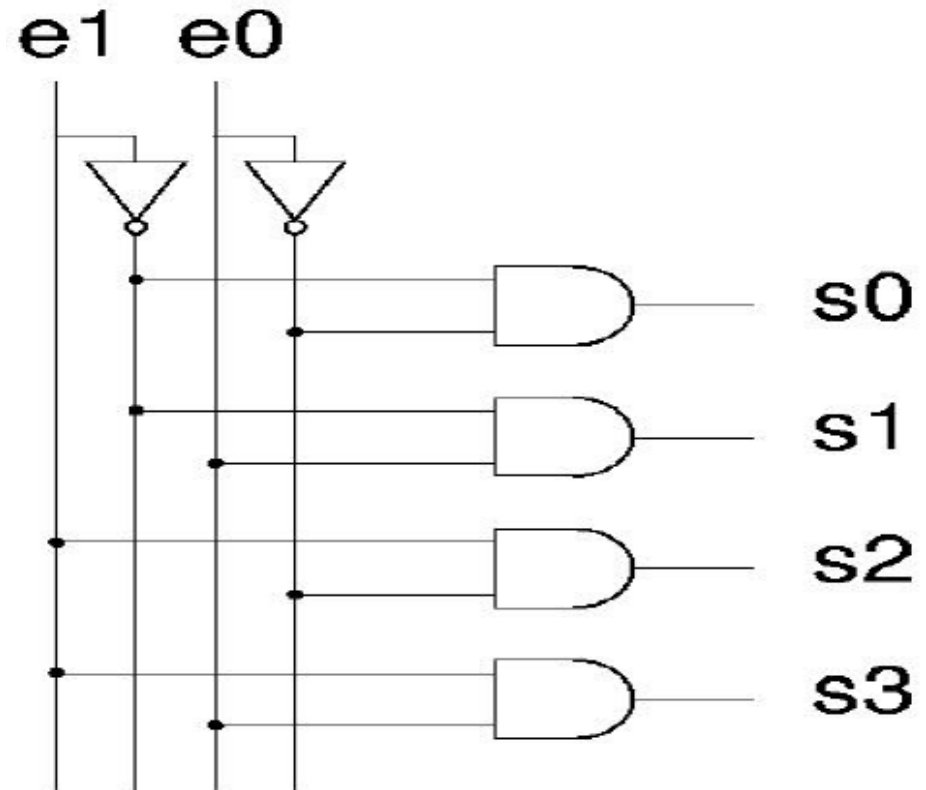
- Mạch có đầu ra biểu diễn biểu thức logic của các biến đầu vào
- Bộ giải mã :
 - cho phép gửi tín hiệu đến một đường ra chọn trước
 - n đường vào, 2^n đường ra



Mạch logic tổ hợp...

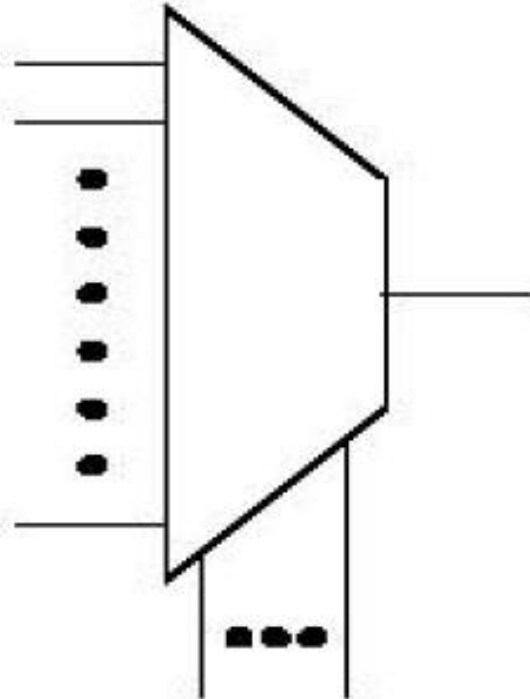
- Ví dụ: bộ giải mã $n=2$

e_1	e_0	s_0	s_1	s_2	s_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



Mạch logic tổ hợp...

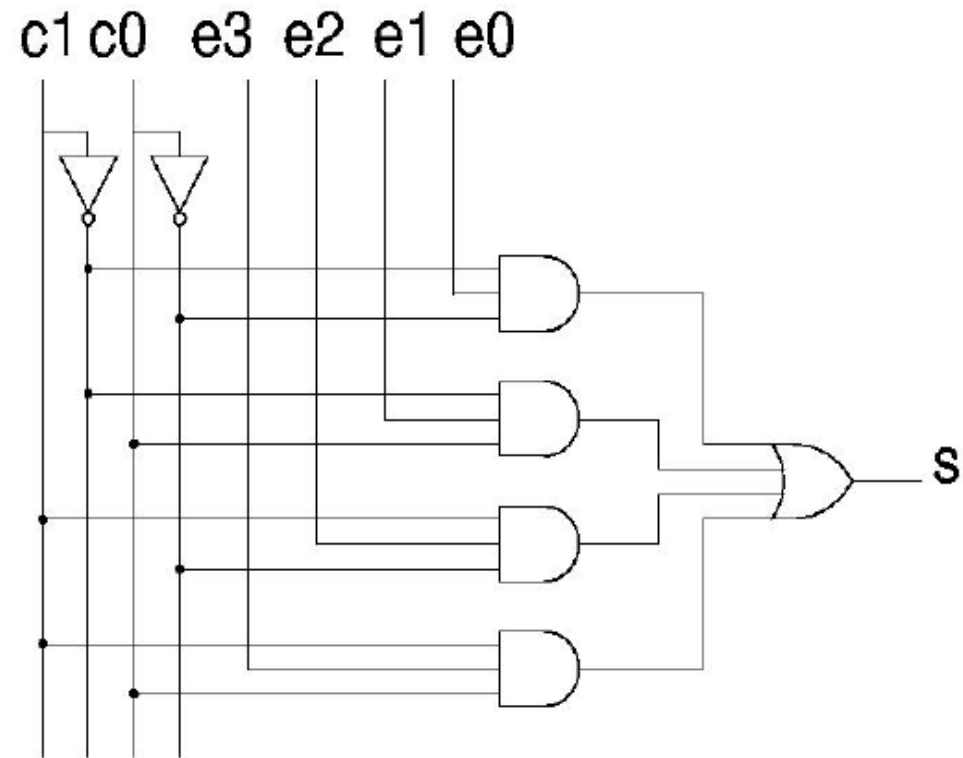
- Bộ dồn kênh: chọn một từ nhiều đầu vào
 - 2^n đầu vào
 - n đường chọn
 - 1 đầu ra



Mạch logic tổ hợp...

- VD: $n=2$

e_3	e_2	e_1	e_0	c_1	c_0	s
-	-	-	x	0	0	x
-	-	x	-	0	1	x
-	x	-	-	1	0	x
x	-	-	-	1	1	x



ALU (Arithmetic & Logic Unit)

- Bộ bán cộng 1-bit:

- $S = x \oplus y$
- $R = xy$

x	y	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

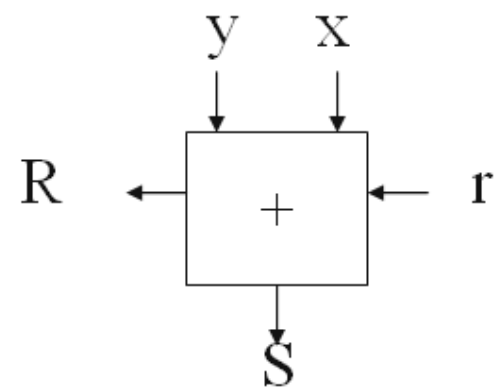
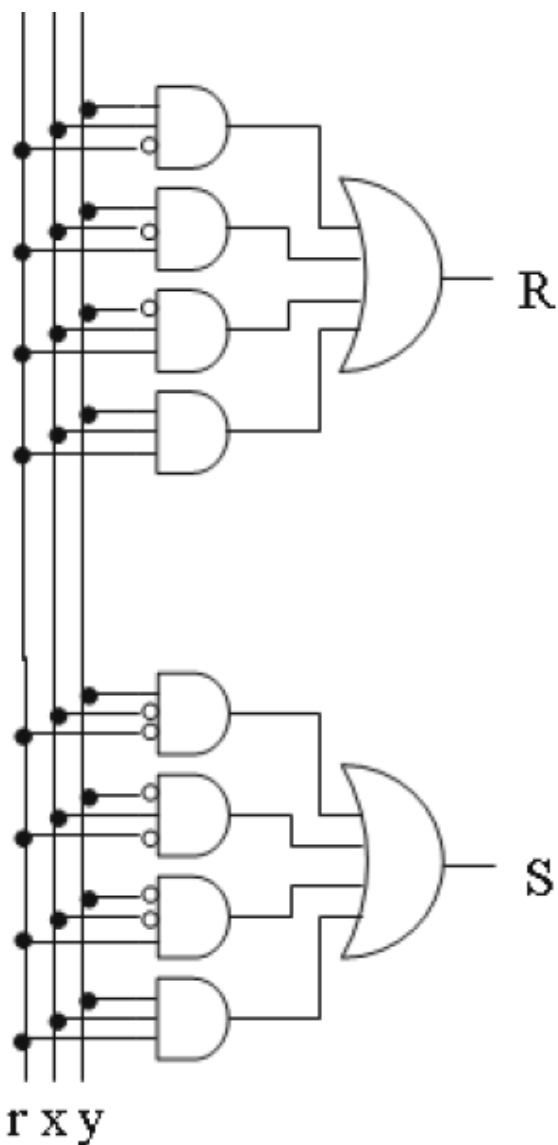
- Bộ cộng 1-bit đầy đủ:

- $S = x \oplus y \oplus R_{in}$
- $R_{out} = xy + R_{in}(x + y)$

R_{in}	x	y	S	R_{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

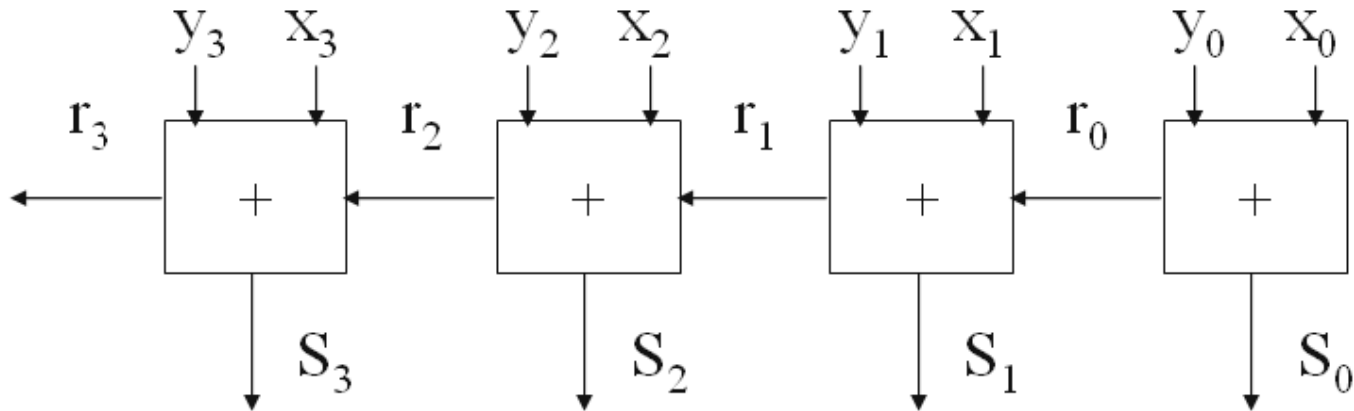
ALU...

Bộ cộng 1-bit đầy đủ



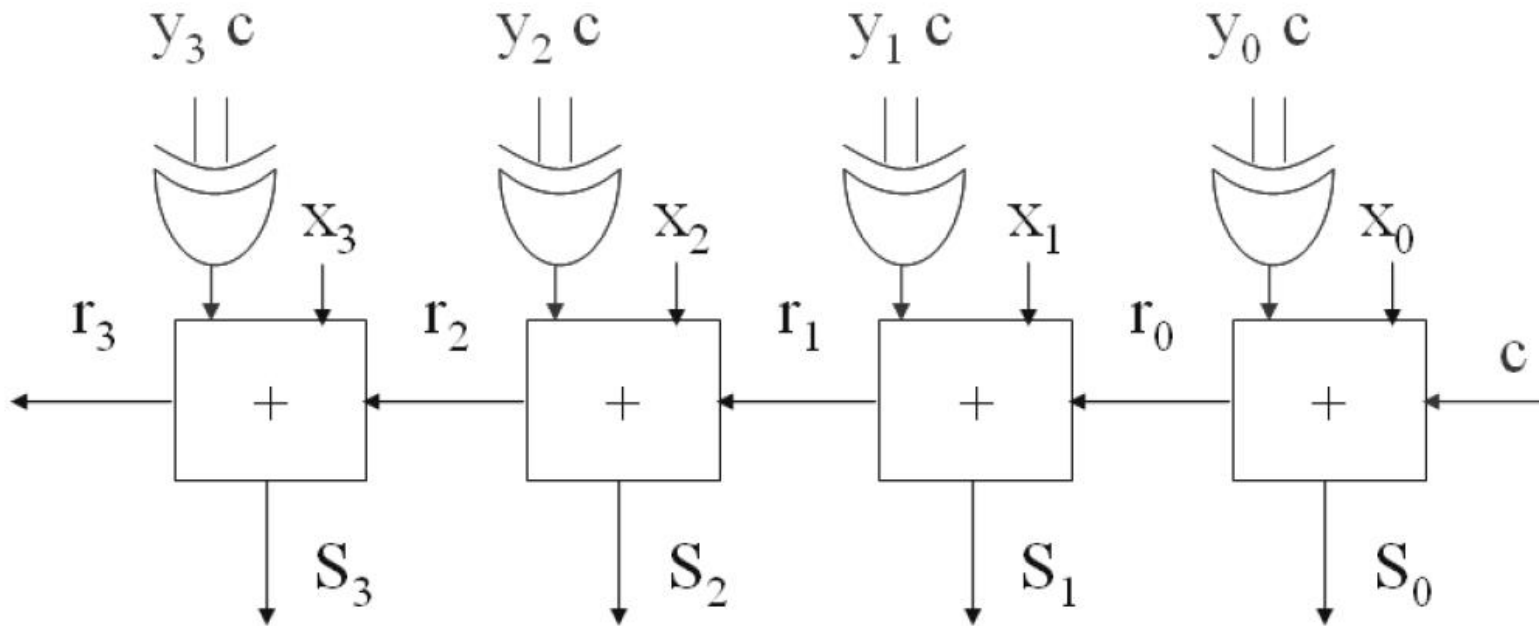
ALU...

- Bộ cộng n-bits: ghép nối n bộ cộng đầy đủ 1-bit



ALU...

- Bộ trừ n-bits: sử dụng bộ cộng n-bits
 - $x - y = x + \tilde{y} + 1$

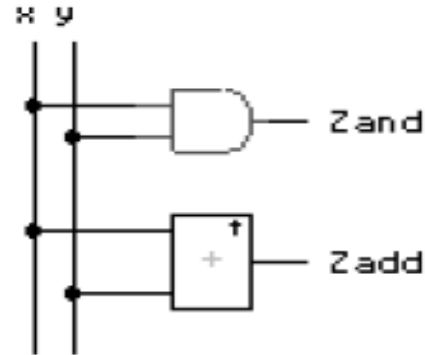


C=0: Cộng

C=1: Trừ

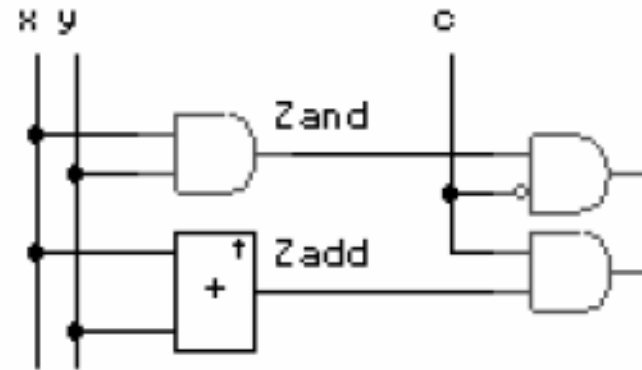
ALU...

- ALU: 3 phần tử cơ bản: ADD, AND và NOT
- ALU 1-bit:



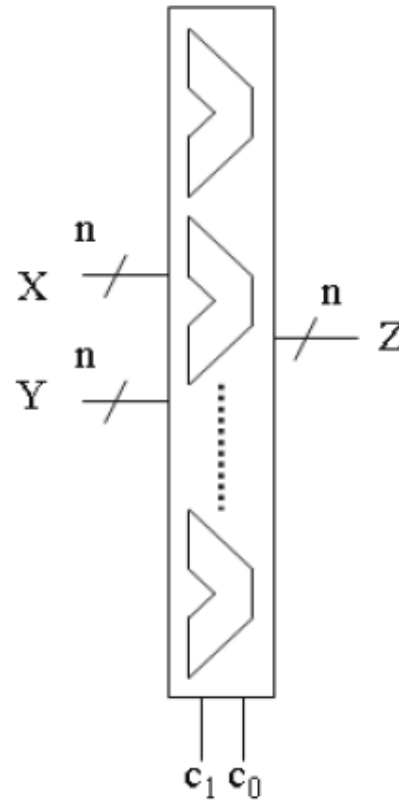
- Lựa chọn 1 đầu ra cho ALU 1-bit

c	Z_{AND}	Z_{ADD}	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



ALU...

- ALU n-bits: kết hợp n ALU 1-bit



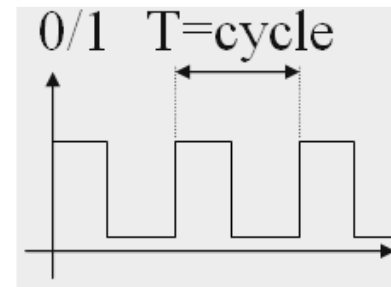
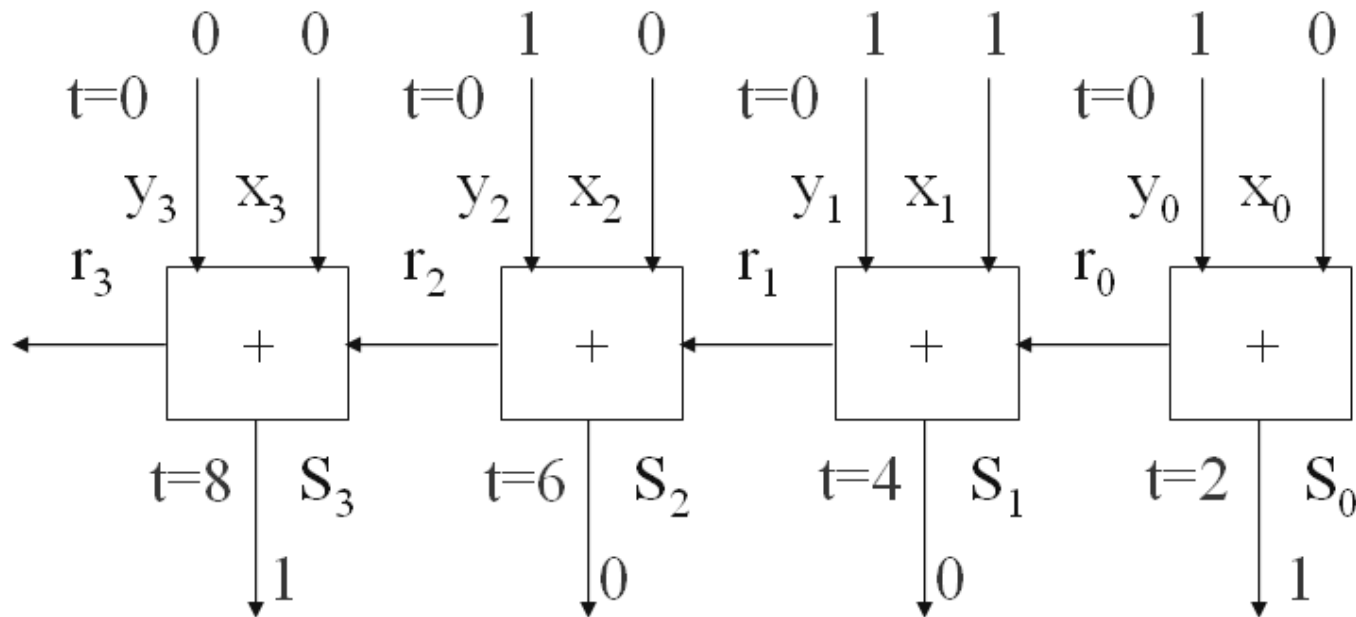
Mạch tuần tự

- Mạch kết hợp:
 - Không thể hiện được khái niệm thời gian
 - Không thể hiện được khái niệm nhớ
- Mạch tuần tự: đầu ra phụ thuộc
 - Trạng thái của các biến vào
 - Trạng thái trước đó của một vài đầu ra
- Mạch tuần tự bao gồm:
 - Đầu vào I
 - Đầu ra O
 - Trạng thái trong S

và được định nghĩa bởi hàm $O = f(I, S)$ xác định đầu ra mới
 $S' = g(I, S)$ chỉ trạng thái mới

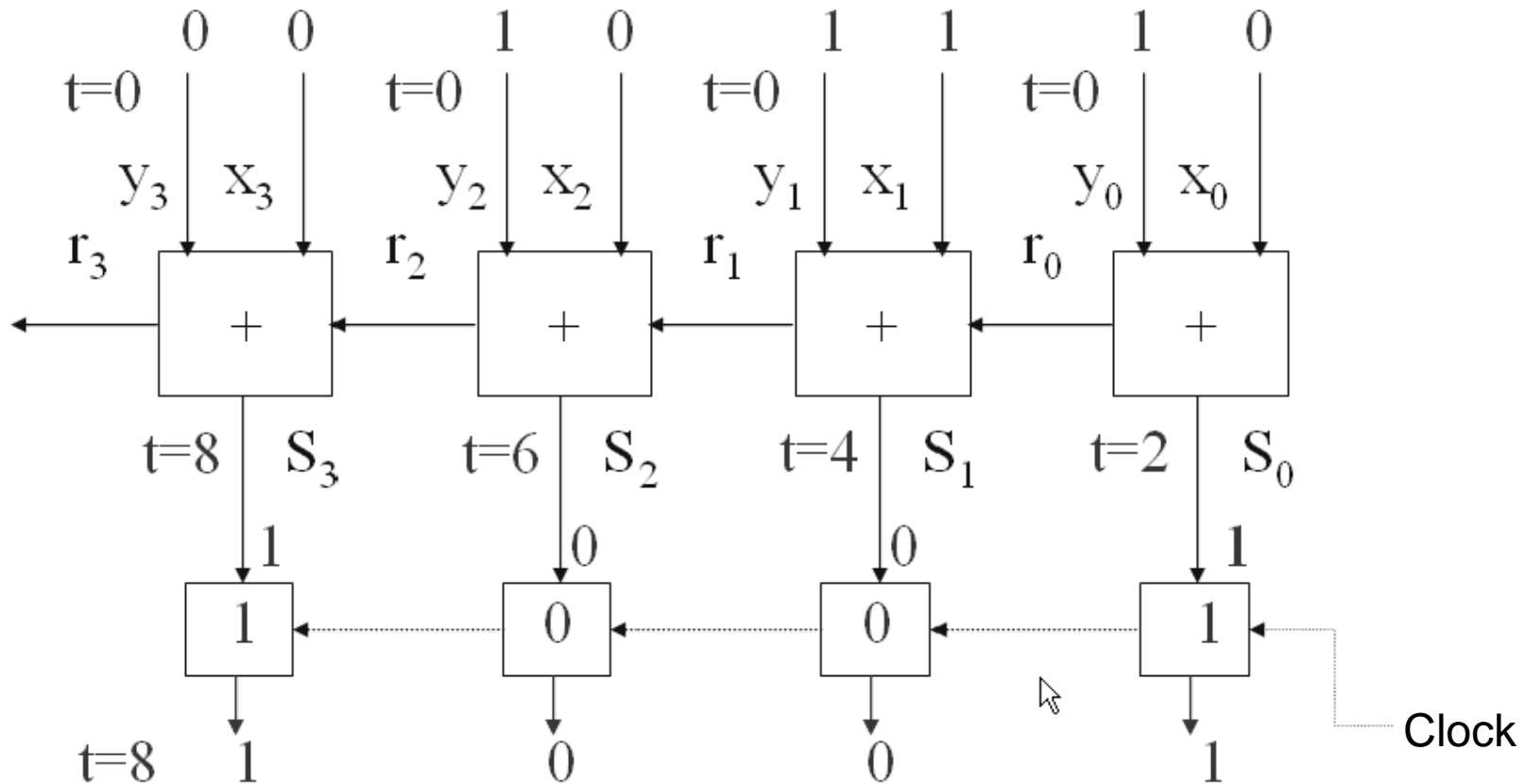
Ràng buộc về thời gian

- Cần phải ước lượng thời gian chuyển đổi qua mỗi thành phần và cắm truyền kết quả cho thành phần kế tiếp khi tính toán chưa xong → ràng buộc = xung đồng hồ



Ràng buộc về thời gian...

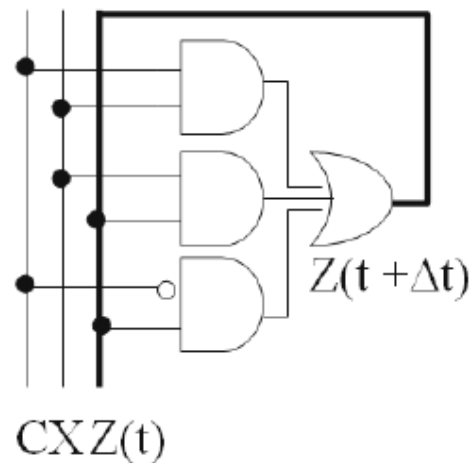
- Tác vụ một thành phần phải được hoàn thành trong một chu kỳ



Khái niệm nhớ

- Tác vụ một thành phần có thể kéo dài tối đa 1 cycle => phải lưu lại giá trị đầu vào trong 1 cycle
- Đầu ra của 1 thành phần là đầu vào của thành phần kế tiếp => cần phải lưu lại giá trị đầu ra
- Khi xung clock $c=1$: mở rào chắn(barrier), cho qua đầu ra Z thông tin hiện có ở đầu vào X
- Khi $c=0$: đóng rào chắn, cung cấp đầu ra thông tin trước đó

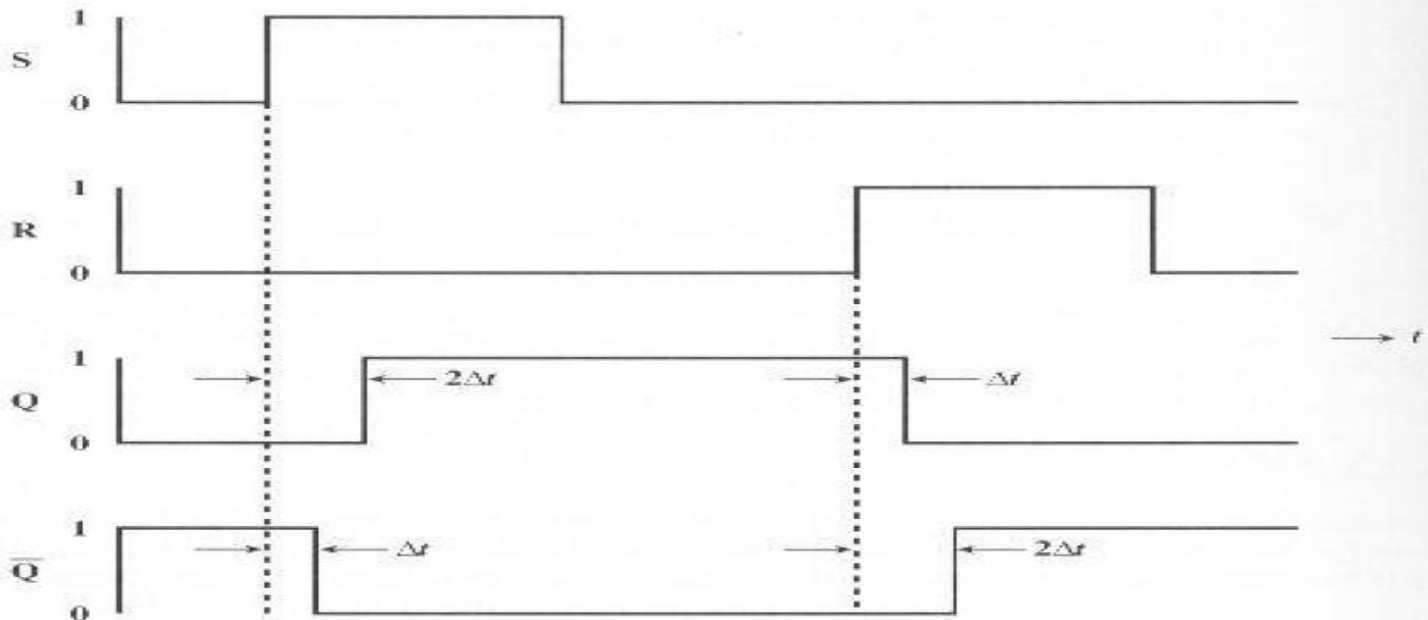
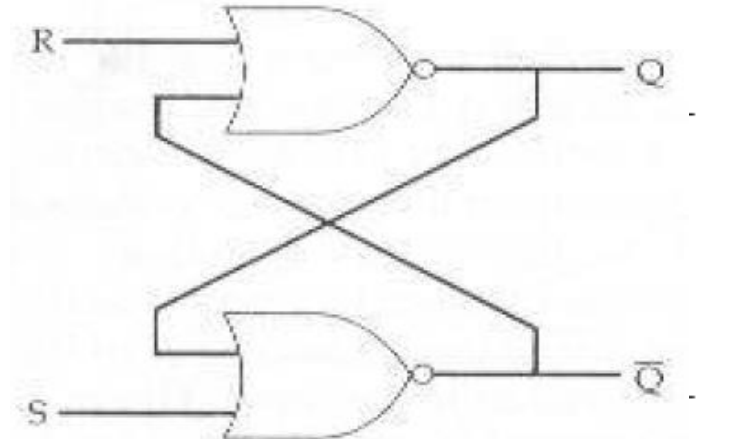
$C(t)$	$X(t)$	$Z(t)$	$Z(t+\delta t)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



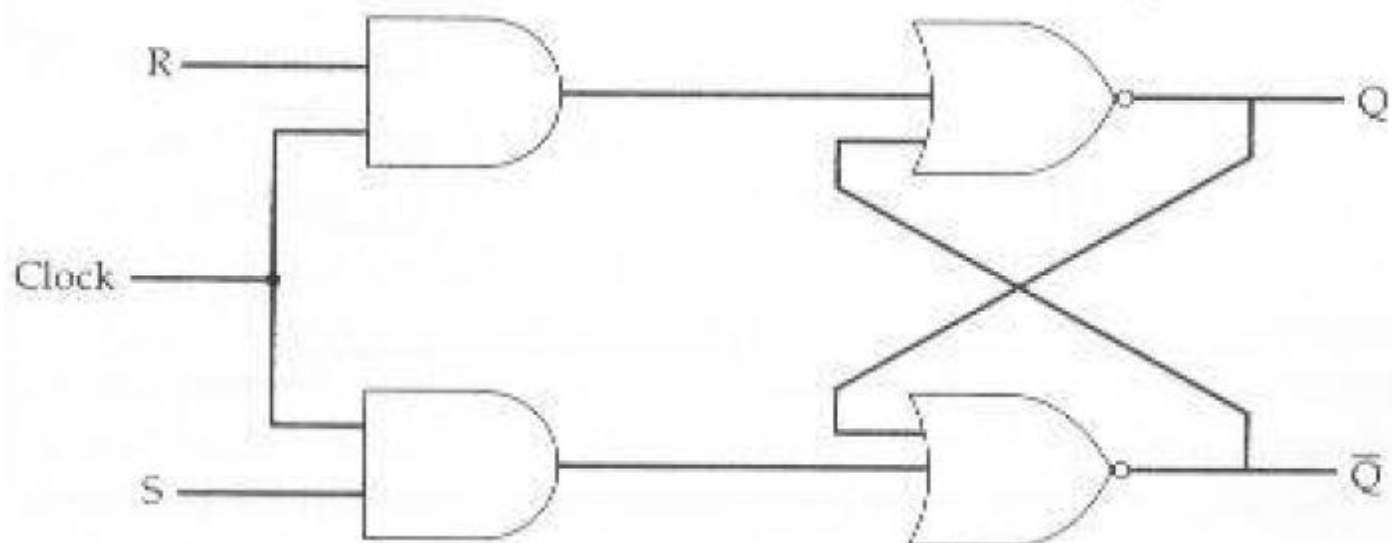
Mạch tuần tự : Mạch lật

- Latch SR: 2 tín hiệu điều khiển S (Set) và R (Reset)

R	S	Q_i	Q_{i+1}
0	0	x	x
0	1	x	1
1	0	x	0
1	1	x	Cấm



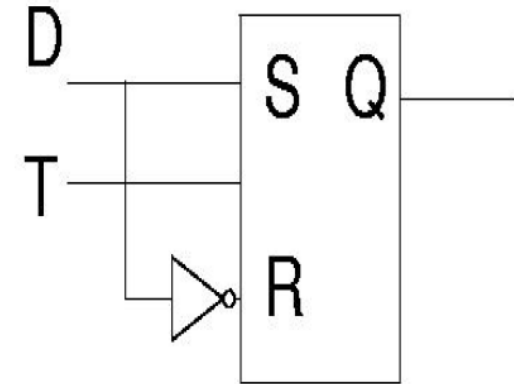
Mạch lật theo xung đồng hồ



Mạch lật...

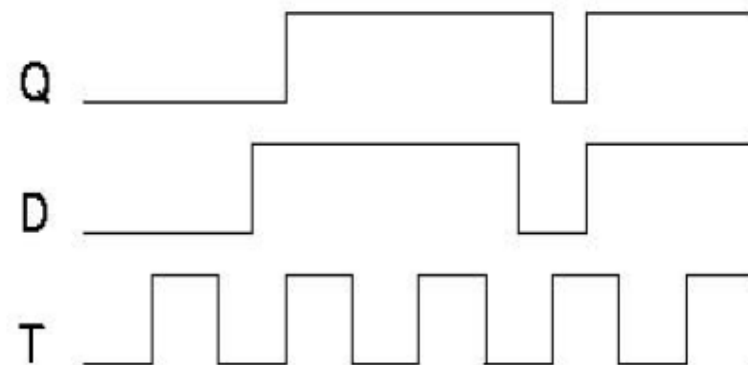
- Latch D: Sử dụng 1 tín hiệu điều khiển D (delay)

D	Q
0	=D=0
1	=D=1



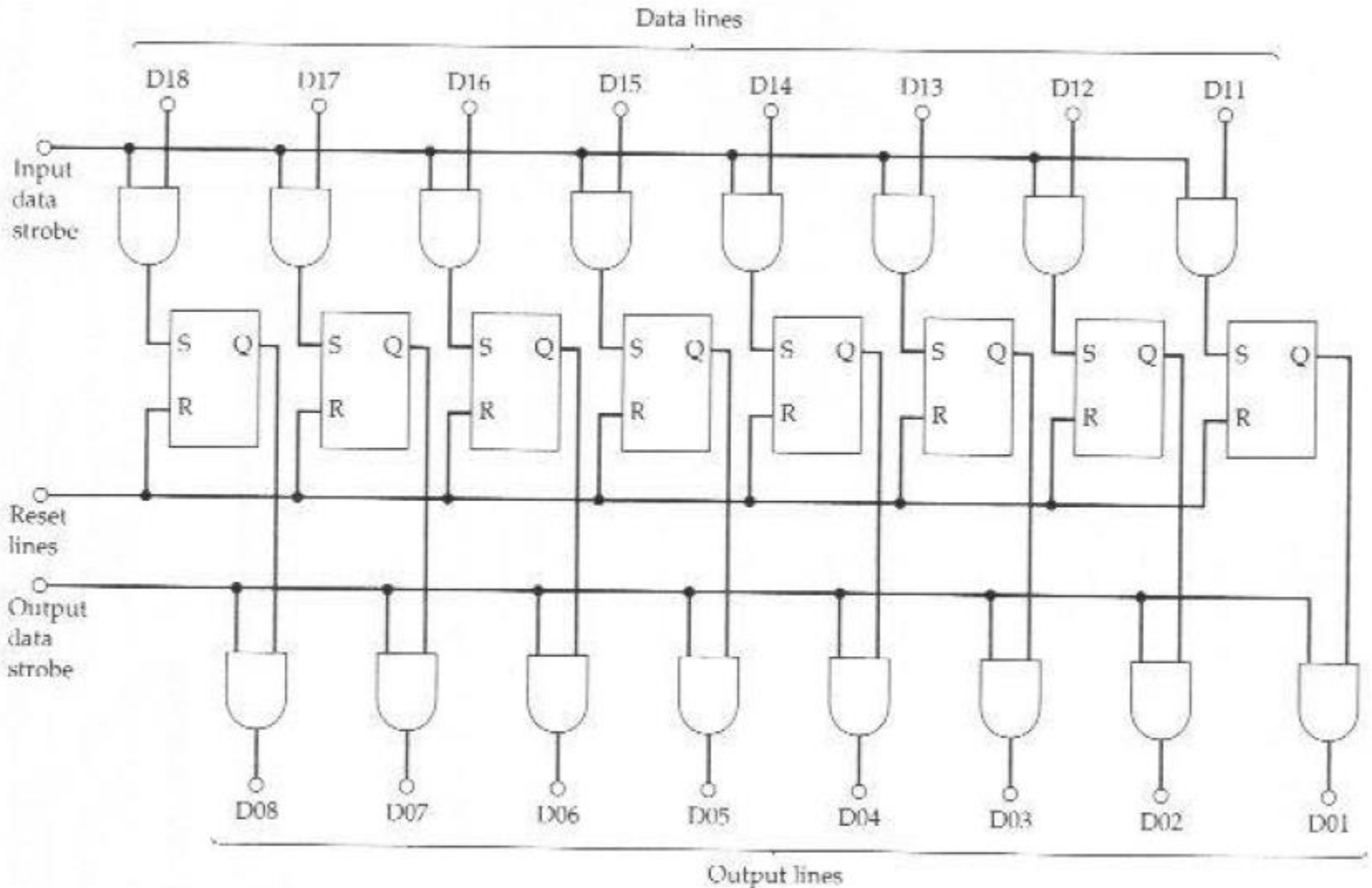
- Latch D hoạt động theo xung nhịp đồng hồ

C	D	SR	Q_{i+1}
0	0	0 1	Q_i
0	1	1 0	Q_i
1	0	0 1	0
1	1	1 0	1



Mạch tuần tự

- Thanh ghi: lưu một từ nhớ



Tham khảo thêm

- Tràn – overflow
- Làm tròn – roundness
- Parity bit
- Mạch nhân
- Mạch chia

Tổng kết

- Biểu diễn thông tin số: ký tự, số nguyên (dấu, bù-1, bù-2, dư), số thực (IEEE-754 đơn, kép)
- Đại số Bool và phổ ứng dụng trong việc thiết kế các mạch logic số tổ hợp và tuần tự
 - Tối ưu hoá biểu thức logic (sử dụng tiên đề/định lý, sử dụng bảng karnaugh)
 - Mạch logic tổ hợp điển hình: bộ giải mã, bộ dồn kênh, bộ cộng 1-bit/n-bit, ALU 1-bit/n-bit.
 - Mạch tuần tự: mạch lật RS, latch D, register, ...