Chương 2

SỐ HỌC VÀ LOGIC

Nội dung chương 2

- Các hệ thống số đếm
- Các phép toán số học
- Các phép toán logic

Các hệ thống số

Hệ thống số đếm	Cơ số	Các ký tự
Nhị phân	2	0,1
Bát phân	8	0, 1, 2, 3, 4, 5, 6, 7
Thập phân	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Thập lục phân	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- Số N trong hệ cơ số X:

$$N_X = a_n a_{n-1} \dots a_1 a_0 . b_1 b_2 \dots b_{m-1} b_m$$

- Phân bố trọng số:

…	X^3	X^2	X^1	X^0	.	X^{-1}	X^{-2}	X^{-3}	…

Chuyển đổi giữa các hệ cơ số

- **Cơ số X sang cơ số 10:**
  - Số N trong hệ cơ số X:

$$N_X = a_n a_{n-1} \dots a_1 a_0 . b_1 b_2 \dots b_{m-1} b_m$$

  - N_X có giá trị là:

$$N_X = a_n X^n + a_{n-1} X^{n-1} + \dots + a_1 X^1 + a_0 X^0 + b_1 X^{-1} + \dots b_m X^{-m}$$

  - Ví dụ:

$$1001_2 = 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 9$$
$$35_8 = 3 * 8^1 + 5 * 8^0 = 29$$
$$A2F_{16} = 10 * 16^2 + 2 * 16^1 + 15 * 16^0 = 2607$$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 10 sang cơ số X:**
  - Phần nguyên:
    - Chia phần nguyên của N cho X được thương và số dư a_o
    - Tiếp tục chia phần thương cho X được thương mới và số dư a_1
    - Tiếp tục cho đến khi thương bằng 0 và số dư a_n
    - Phần nguyên biểu diễn trong hệ cơ số X là $a_n \ldots a_1 a_o$
  - Phần thập phân
    - Nhân phần thập phân của N với X được tích có phần nguyên là b_1
    - Tiếp tục nhân phần thập phân của tích với X được tích mới có phần nguyên là b_2
    - Tiếp tục cho đến khi phần thập phân của tích nhận được bằng 0 hoặc sau một số bước nhất định tùy theo độ chính xác yêu cầu
    - Phần thập phân biểu diễn trong hệ cơ số X là $b_1 b_2 \ldots b_m$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 10 sang cơ số X:**
  - Ví dụ 1: biến đổi 8.625_{10} sang nhị phân

8 : 2 = 4 *dư* 0 *(LSB)*
4 : 2 = 2 *dư* 0
2 : 2 = 1 *dư* 0
1 : 2 = 0 *dư* 1

1 0 0 0 . 1 0 1

0.625 x 2 = 1.25 *phần nguyên* 1 *(MSB)*
0.25 x 2 = 0.5 *phần nguyên* 0
0.5 x 2 = 1.0 *phần nguyên* 1

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 10 sang cơ số X:**
  - Ví dụ 2: biến đổi 1480.4296875_{10} sang thập lục phân

1480 : 16 = 92 *dư* **8** *(LSD)*
92 : 16 = 5 *dư* **12**
5 : 16 = 0 *dư* **5**

5 C 8 . 6 E

0.4296875 x 16 = 6.875 *phần nguyên* **6** *(MSD)*
0.875 x 16 = 14.0 *phần nguyên* **14**

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 8 sang cơ số 2:**
  - Biến mỗi ký tự trong hệ bát phân thành 3 bit nhị phân tương ứng

Octal	0	1	2	3	4	5	6	7
Binary	000	001	010	011	100	101	110	111

  - Ví dụ: biến đổi 472_8 sang nhị phân

$$\begin{array}{ccc} 4 & 7 & 2 \\ \downarrow & \downarrow & \downarrow \\ 100 & 111 & 010 \end{array} \qquad \boxed{100111010_2}$$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 16 sang cơ số 2:**
  - Biến mỗi ký tự trong hệ thập lục phân thành 4 bit nhị phân tương ứng
  - Ví dụ: biến đổi $10AF_{16}$ sang nhị phân

$$\begin{array}{cccc} 1 & 0 & A & F \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 0001 & 0000 & 1010 & 1111 \end{array} \quad \boxed{1000010101111_2}$$

Hexa	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Chuyển đổi giữa các hệ cơ số (tt)

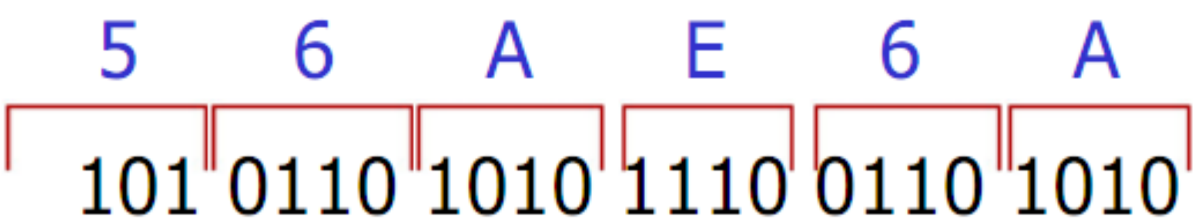
- **Cơ số 2 sang cơ số 8:**
  - Bắt đầu từ phải sang trái, nhóm các bit nhị phân thành các nhóm 3 bit
  - Biến đổi mỗi nhóm 3 bit thành một Octal
  - Ví dụ: biến đổi 1011010111_2 sang bát phân

$$\begin{array}{cccc} 1 & 3 & 2 & 7 \\ 1 & 011 & 010 & 111 \end{array}$$

$$1011010111_2 = 1327_8$$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 2 sang cơ số 16:**
  - Bắt đầu từ phải sang trái, nhóm các bit nhị phân thành các nhóm 4 bit
  - Biến đổi mỗi nhóm 4 bit thành một Hexa
  - Ví dụ: biến đổi $10101101010111001101010_2$ sang thập lục phân



$$10101101010111001101010_2 = 56AE6A_{16}$$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 8 sang cơ số 16:**
  - Biến đổi số bát phân thành số nhị phân
  - Biến đổi số nhị phân thành số thập lục phân
  - Ví dụ: biến đổi 1076_8 sang thập lục phân

$$\begin{array}{cccc} 1 & 0 & 7 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 001 & 000 & 111 & 110 \end{array}$$

$$\underbrace{001\,0}_{2}\underbrace{00\,11}_{3}\underbrace{1\,110}_{E}$$

$$1076_8 = 23E_{16}$$

Chuyển đổi giữa các hệ cơ số (tt)

- **Cơ số 16 sang cơ số 8:**
  - Biến đổi số thập lục phân thành số nhị phân
  - Biến đổi số nhị phân thành số bát phân
  - Ví dụ: biến đổi $1F0C_{16}$ sang bát phân

$$
\begin{array}{cccc}
1 & F & 0 & C \\
\downarrow & \downarrow & \downarrow & \downarrow \\
0001 & 1111 & 0000 & 1100
\end{array}
$$

$$1 \quad 7 \quad 4 \quad 1 \quad 4$$

$$1F0C_{16} = 17414_8$$

Nội dung chương 2

- Các hệ thống số đếm
- Các phép toán số học
- Các phép toán logic

Phép cộng nhị phân

- Cộng hai bit nhị phân

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	0 (nhớ 1)

Phép cộng nhị phân (tt)

- Cộng hai số nhị phân không dấu

$$
\begin{array}{cccccc}
 & & \mathit{1} & \mathit{1} & \mathit{1} & \\
 & 1 & 0 & 1 & 1 & 1 \\
+ & & & 1 & 0 & 1 \\
\hline
 & 1 & 1 & 1 & 0 & 0
\end{array}
$$

Phép trừ nhị phân

- Trừ hai bit nhị phân

A	B	A - B
0	0	0
0	1	1 (mượn 1)
1	0	1
1	1	0

Phép trừ nhị phân (tt)

- Trừ hai số nhị phân không dấu

$$
\begin{array}{cccccc}
 & & -1 & -1 & -1 & \\
 & 1 & 1 & 0 & 1 & 0 \\
- & & & 1 & 1 & 1 \\
\hline
 & 1 & 0 & 0 & 1 & 1
\end{array}
$$

Phép nhân nhị phân

- Nhân hai bit nhị phân

A	B	A x B
0	0	0
0	1	0
1	0	0
1	1	1

Phép nhân nhị phân (tt)

- Nhân hai số nhị phân

$$
\begin{array}{cccccccc}
 & & & & 1 & 0 & 1 & 1 \\
 & & & \times & 1 & 0 & 0 & 1 \\
\cline{4-8}
 & & & & 1 & 0 & 1 & 1 \\
+ & & & 0 & 0 & 0 & 0 & \\
 & & 0 & 0 & 0 & 0 & & \\
 & 1 & 0 & 1 & 1 & & & \\
\hline
 & 1 & 1 & 0 & 0 & 0 & 1 & 1
\end{array}
$$

Phép chia nhị phân

- Chia hai số nhị phân

```
  1 0 0 1 0 0 0 1 | 1 0 1 1
- 1 0 1 1         |---------
  -------         | 1 1 0 1
    1 1 1 0
  - 1 0 1 1
    -------
        1 1 0 1
      - 1 0 1 1
        -------
            1 0
```

Số nhị phân có dấu – Bù hai

- **Số bù hai**: đảo bit cộng 1
  - Ví dụ: số bù hai (8 bit) của 5

5	$= 00000101_2$
Đảo bit của 5	$= 11111010_2$
Cộng 1	$= 11111011_2$
-5	$= 11111011_2$

  - Bit có trọng số lớn nhất quy ước dấu
    - Bit dấu bằng 0 xác định số dương
    - Bit dấu bằng 1 xác định số âm
  - Tìm giá trị của số âm:
    - Cách 1: Khai triển như số dương nhưng bit có trọng số lớn nhất được nhân thêm với (-1)
    - Cách 2: Lấy bù hai của nó được số dương có cùng biên độ
  - Phạm vi biểu diễn của số nhị phân có dấu n bit: -2^{n-1} đến $2^{n-1} - 1$

Cộng trừ số bù hai

- Thực hiện như số không dấu
- Thực hiện trên toán hạng có cùng chiều dài và kết quả cũng có cùng số bit
- Kết quả đúng nếu nằm trong phạm vi biểu diễn số dấu, nếu kết quả sai thì cần mở rộng chiều dài bit

Cộng trừ số nhị phân có dấu – Bù hai (tt)

- Ví dụ:

$$\begin{array}{rrcr} & -6 & : & 1010 \\ + & +3 & : & 0011 \\ \hline & -3 & : & 1101 \end{array} \qquad \begin{array}{rrcr} & -2 & : & 1110 \\ + & -5 & : & 1011 \\ \hline & -7 & : & 1001 \end{array}$$

$$\begin{array}{rrcr} & -6 & : & 1010 \\ - & -2 & : & 1110 \\ \hline & -4 & : & 1100 \end{array} \qquad \begin{array}{rrcr} & +2 & : & 0010 \\ - & -5 & : & 1011 \\ \hline & +7 & : & 0111 \end{array}$$

Cộng trừ số bù hai (tt)

- Ví dụ:

	+4	:	0100	→	00100
+					
	+5	:	0101	→	00101
	-7	:	1001 *(Kq sai)*		01001 : +9 *(Kq đúng)*

	-7	:	1001	→	11001
-					
	+5	:	0101	→	00101
	+4	:	0100 *(Kq sai)*		10100 : -12 *(Kq đúng)*

Số nhị phân có dấu – Dấu lượng

- **Dấu lượng** (sign-and-magnitude)
  - Bit có trọng số lớn nhất quy ước dấu
    - Bit dấu bằng 0 xác định số dương
    - Bit dấu bằng 1 xác định số âm
  - Các bit còn lại biểu diễn giá trị độ lớn của số
  - Ví dụ: biểu diễn 8 bit của 5 và -5

    5 $= 00000101_2$

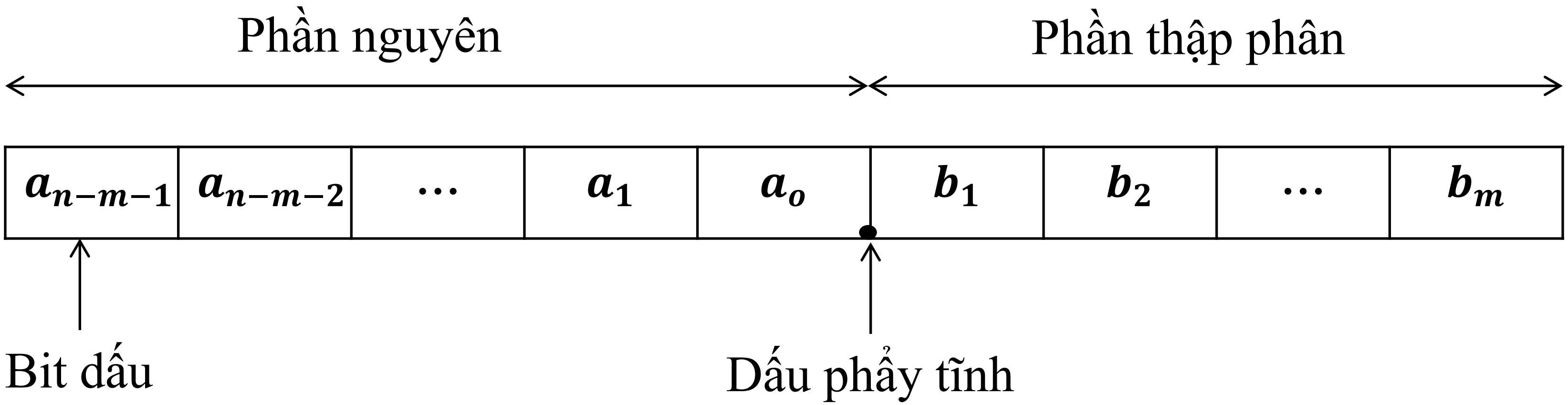
    -5 $= 10000101_2$
  - Phạm vi biểu diễn của số nhị phân có dấu n bit: $-2^{n-1}+1$ đến $2^{n-1} - 1$

Cộng trừ số dấu lượng

- Cộng hai số nhị phân có dấu:
  - Nếu hai số cùng dấu thì thực hiện phép cộng phần biểu diễn giá trị và sử dụng bit dấu cùng dấu với hai số đó
  - Nếu hai số khác dấu thì kết quả sẽ nhận dấu của toán tử lớn hơn, và thực hiện phép trừ giữa toán tử có giá trị lớn hơn với toán tử bé hơn
- Trừ hai số nhị phân có dấu:
  - Đổi dấu số trừ và thực hiện phép cộng hai số nhị phân có dấu giữa số bị trừ và số trừ sau khi đổi dấu

Số dấu phẩy tĩnh (Fixed-point number)

- Sử dụng dấu chấm ảo để biểu diễn một số thực, phân biệt giữa phần biểu diễn giá trị nguyên và phần lẻ thập phân
- Định dạng số dấu phẩy tĩnh *n* bit với *m* bit dùng cho phần lẻ thập phân:



Số dấu phẩy động (Floating-point number)

- Số thực N thường được biểu diễn dưới dạng số dấu phẩy động :

$$N = \pm M * X^{E} \quad (1 <= M < X)$$

Trong đó, M là phần định trị (Mantissa)

X là cơ số (Radix)

E là phần lũy thừa (Exponent)

Số dấu phẩy động (tt)

- Chuẩn IEEE 754:

Kiểu	Dấu	Phần mũ	Phần định trị	Tổng số bit	Phân cực mũ
Nửa	1	5	10	16	15
Đơn	1	8	23	32	127
Kép	1	11	52	64	1023
Bậc bốn	1	15	112	128	16383

- Ví dụ số dấu phẩy động kiểu đơn (32 bit):

1 bit	8 bit	23 bit
S	e	m

Trong đó, S là bít dấu

$e = E + 127$

m là phần lẽ của phần định trị M ($M = 1.m$)

Công thức xác định giá trị số thực như sau:

$$X = (-1)^S * 1.m * 2^{e-127}$$

Nội dung chương 2

- Các hệ thống số đếm
- Các phép toán số học
- Các phép toán logic

Đại số Boole

- Các hằng và biến trong đại số Boole chỉ có hai giá trị 0 và 1
- Trong đại số Boole không có phân số, số âm, lũy thừa, căn số, …
- Đại số Boole chỉ có 3 toán tử: nhân logic (AND), cộng logic (OR) và bù logic (NOT)

x	y	$x.y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

x	$\overline{x}$
0	1
1	0

Các tiên đề

- **Tính giao hoán:**

$$x + y = y + x$$
$$x . y = y . x$$

- **Tính kết hợp:**

$$(x + y) + z = x + (y + z)$$
$$(x . y) . z = x . (y . z)$$

- **Tính phân phối:**

$$x . (y + z) = x . y + x . z$$
$$x + (y . z) = (x + y) . (x + z)$$

- **Phần tử đồng nhất:**

$$x + 1 = 1$$
$$x + 0 = x$$
$$x . 1 = x$$
$$x . 0 = 0$$

- **Phần tử bù:**

$$x + \bar{x} = 1$$
$$x . \bar{x} = 0$$

Các định lý cơ bản

- $\bar{\bar{\mathbf{x}}} = \mathbf{x}$
- $\mathbf{x + x = x}$
- $\mathbf{x \,.\, x = x}$
- **Định lý hấp thu:**

(1) $\mathbf{x + x \,.\, y = x}$

<u>Chứng minh:</u> $x + x \,.\, y = x\,(1 + y) = x\,.1 = x$

(2) $\mathbf{x \,.\, (x + \mathit{y}) = x}$

<u>Chứng minh:</u> $x \,.\, (x + y) = x \,.\, x + x \,.\, y = x + x \,.\, y = x$

(3) $\mathbf{x + (\bar{x} \,.\, \mathit{y}) = x + y}$

<u>Chứng minh:</u> $x + (\bar{x} \,.\, y) = (x + \bar{x}).\,(x + y) = 1.\,(x + y) = x + y$

(4) $\mathbf{x \,.\, (\bar{x} + \mathit{y}) = x \,.\, y}$

<u>Chứng minh:</u> $x \,.\, (\bar{x} + y) = x \,.\, \bar{x} + x \,.\, y = 0 + x \,.\, y = x \,.\, y$

(5) $\mathbf{x \,.\, y + \bar{x} \,.\, z + y \,.\, z = x \,.\, y + \bar{x} \,.\, z}$

<u>Chứng minh:</u>
$$\begin{aligned} x \,.\, y + \bar{x} \,.\, z + y \,.\, z &= x \,.\, y + \bar{x} \,.\, z + y \,.\, z(x + \bar{x}) \\ &= x \,.\, y + \bar{x} \,.\, z + x \,.\, y \,.\, z + \bar{x} \,.\, y \,.\, z \\ &= (x \,.\, y + x \,.\, y \,.\, z) + (\bar{x} \,.\, z + \bar{x} \,.\, y \,.\, z) \\ &= x \,.\, y + \bar{x} \,.\, z \end{aligned}$$

Các định lý cơ bản (tt)

- Định lý De Morgan:

$$\overline{\boldsymbol{x}+\boldsymbol{y}} = \overline{\boldsymbol{x}}\,.\,\overline{\boldsymbol{y}} \quad (\mathbf{1})$$

$$\overline{\boldsymbol{x}\,.\,\boldsymbol{y}} = \overline{\boldsymbol{x}} + \overline{\boldsymbol{y}} \quad (\mathbf{2})$$

<u>Chứng minh (1):</u>

$\boldsymbol{x}$	$\boldsymbol{y}$	$\boldsymbol{x}+\boldsymbol{y}$	$\overline{\boldsymbol{x}+\boldsymbol{y}}$	$\overline{\boldsymbol{x}}$	$\overline{\boldsymbol{y}}$	$\overline{\boldsymbol{x}}\,.\,\overline{\boldsymbol{y}}$
0	0	0	**1**	1	1	**1**
0	1	1	**0**	1	0	**0**
1	0	1	**0**	0	1	**0**
1	1	1	**0**	0	0	**0**

Mở rộng cho n biến:

$$\overline{x_1 + x_2 + \cdots + x_n} = \overline{x_1}.\overline{x_2} \ldots \overline{x_n}$$

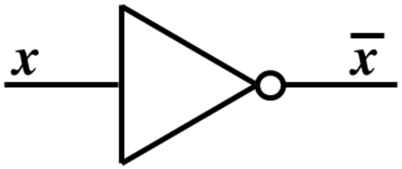
$$\overline{x_1 x_2 \ldots x_n} = \overline{x_1} + \overline{x_2} + \cdots + \overline{x_n}$$

Các định lý cơ bản (tt)

- Áp dụng các tiên đề, định lý của đại số Boole rút gọn các biểu thức sau:
  1. $AB\bar{C} + \overline{AB\bar{C}} = 1$
  2. $A + \bar{B}C + \bar{D}(A + \bar{B}C) = A + \bar{B}C$
  3. $(A + \bar{B})(\bar{A}B + BCD) = (A + \bar{B})BCD = ABCD$
  4. $\bar{A}(B + \bar{C})(A + \bar{B}C) = 0$

Cổng logic (tt)

- Cổng đảo (NOT)
  - Ký hiệu (ANSI):

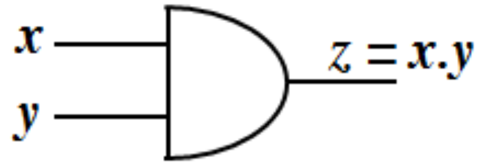


  - Phương trình logic: $y = \overline{x}$
  - Bảng trạng thái:

x	$y = \overline{x}$
0	1
1	0

Cổng logic

- Cổng AND
  - Ký hiệu (ANSI):



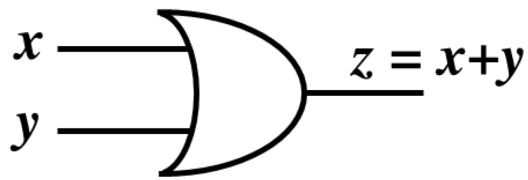
  - Phương trình logic: $z = x.y$
  - Bảng trạng thái:

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

$$\boxed{y = x_1x_2 \dots x_n = \begin{cases} 1 \text{ nếu } x_1 = x_2 = \cdots = x_n = 1 \\ 0 \text{ nếu } \exists\, x_i = 0 \ (i = 1, \dots, n) \end{cases}}$$

Cổng logic (tt)

- Cổng OR
  - Ký hiệu (ANSI):



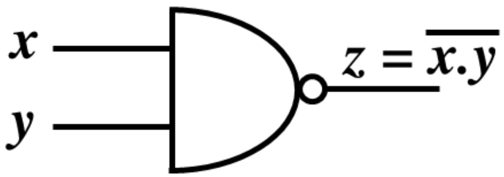
  - Phương trình logic: z = x + y
  - Bảng trạng thái:

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

$$\boxed{y = x_1 + x_2 + \cdots + x_n = \begin{cases} 0 \ \ nếu\ x_1 = x_2 = \cdots = x_n = 0 \\ 1 \ \ nếu\ \exists\ x_i = 1\ (i = 1, \dots, n) \end{cases}}$$

Cổng logic (tt)

- Cổng NAND
  - Ký hiệu (ANSI):



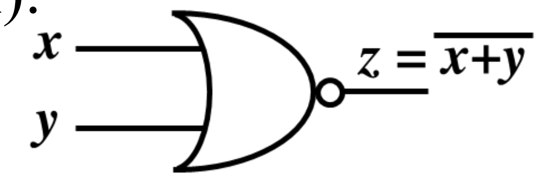
  - Phương trình logic: $z = \overline{x\,.\,y}$
  - Bảng trạng thái:

x	*y*	*z*
0	0	1
0	1	1
1	0	1
1	1	0

$$\boxed{y = \overline{x_1 x_2 \ldots x_n} = \begin{cases} 0 \text{ nếu } x_1 = x_2 = \cdots = x_n = 1 \\ 1 \text{ nếu } \exists\, x_i = 0 \ (i = 1, \ldots, n) \end{cases}}$$

Cổng logic (tt)

- Cổng NOR
  - Ký hiệu (ANSI):



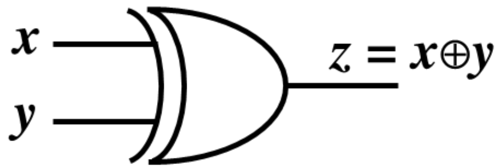
  - Phương trình logic: $z = \overline{x + y}$
  - Bảng trạng thái:

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

$$y = \overline{x_1 + x_2 + \cdots + x_n} = \begin{cases} 1 \;\; n\acute{\hat{e}}u\; x_1 = x_2 = \cdots = x_n = 0 \\ 0 \;\; n\acute{\hat{e}}u\; \exists\, x_i = 1\; (i = 1, \dots, n) \end{cases}$$

Cổng logic (tt)

- Cổng XOR (Exclusive_OR)
  - Ký hiệu:



  - Phương trình logic: $z = x \oplus y = \bar{x}y + x\bar{y}$
  - Bảng trạng thái:

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

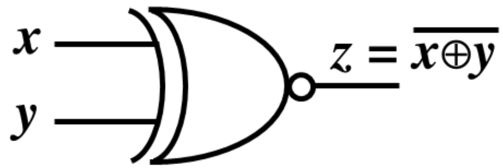
$$y = x_1 \oplus x_2 \oplus \ldots \oplus x_n = \begin{cases} 1 \ \textit{nếu số đầu vào bằng } 1 \textit{ là số lẻ} \\ 0 \ \textit{nếu số đầu vào bằng } 1 \textit{ là số chẵn} \end{cases}$$

Cổng logic (tt)

- Các tính chất của phép XOR
  - $x \oplus y = y \oplus x$
  - $x . (y \oplus z) = (x. y) \oplus (x. z)$
  - $x \oplus 0 = x$
  - $x \oplus 1 = \overline{x}$
  - $x \oplus x = 0$
  - $x \oplus \overline{x} = 1$

Cổng logic (tt)

- Cổng XNOR (Exclusive_NOR)
  - Ký hiệu:

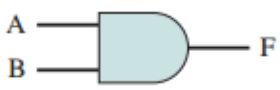
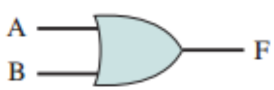
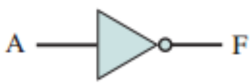
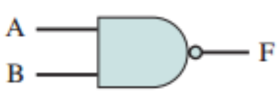
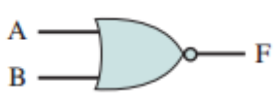
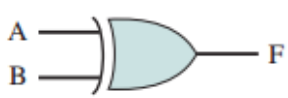


  - Phương trình logic: $z = \overline{x \oplus y} = (\bar{x} + y)(x + \bar{y}) = \bar{x}\bar{y} + xy$
  - Bảng trạng thái:

x	*y*	*z*
0	0	1
0	1	0
1	0	0
1	1	1

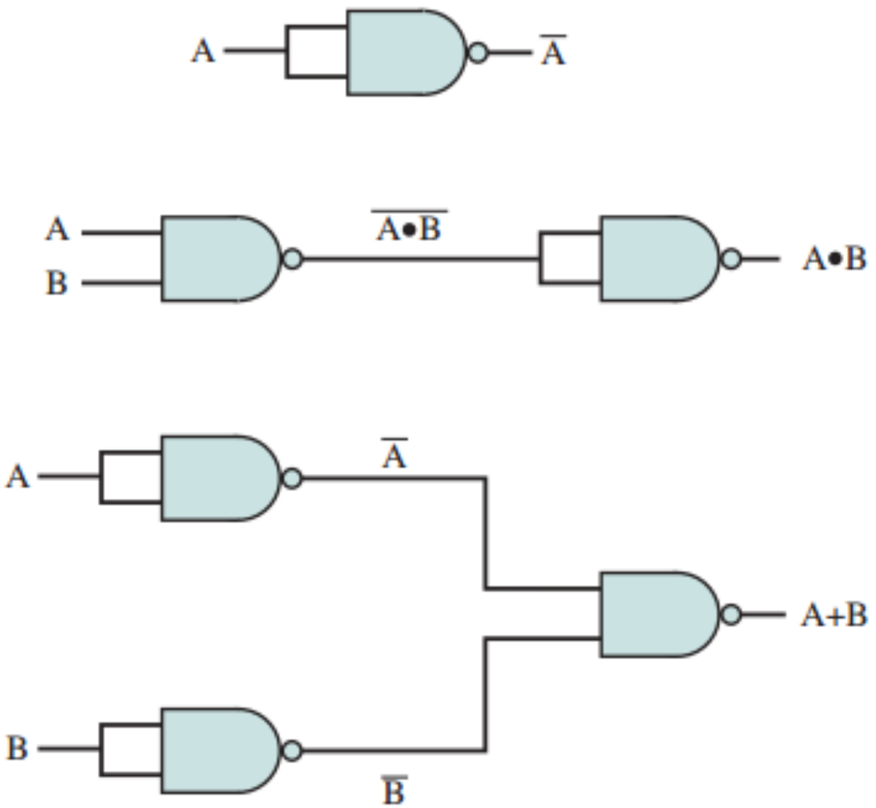
$$y = \overline{x_1 \oplus x_2 \oplus \ldots \oplus x_n} = \begin{cases} 0 \ \textit{nếu số đầu vào bằng } 1 \textit{ là số lẻ} \\ 1 \ \textit{nếu số đầu vào bằng } 1 \textit{ là số chẳn} \end{cases}$$

Cổng logic (tt)

Name	Graphical Symbol	Algebraic Function	Truth Table
AND		$F = A \bullet B$ or $F = AB$	A B | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$F = A + B$	A B | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
NOT		$F = \overline{A}$ or $F = A'$	A | F 0 | 1 1 | 0
NAND		$F = \overline{AB}$	A B | F 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		$F = \overline{A + B}$	A B | F 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
XOR		$F = A \oplus B$	A B | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0

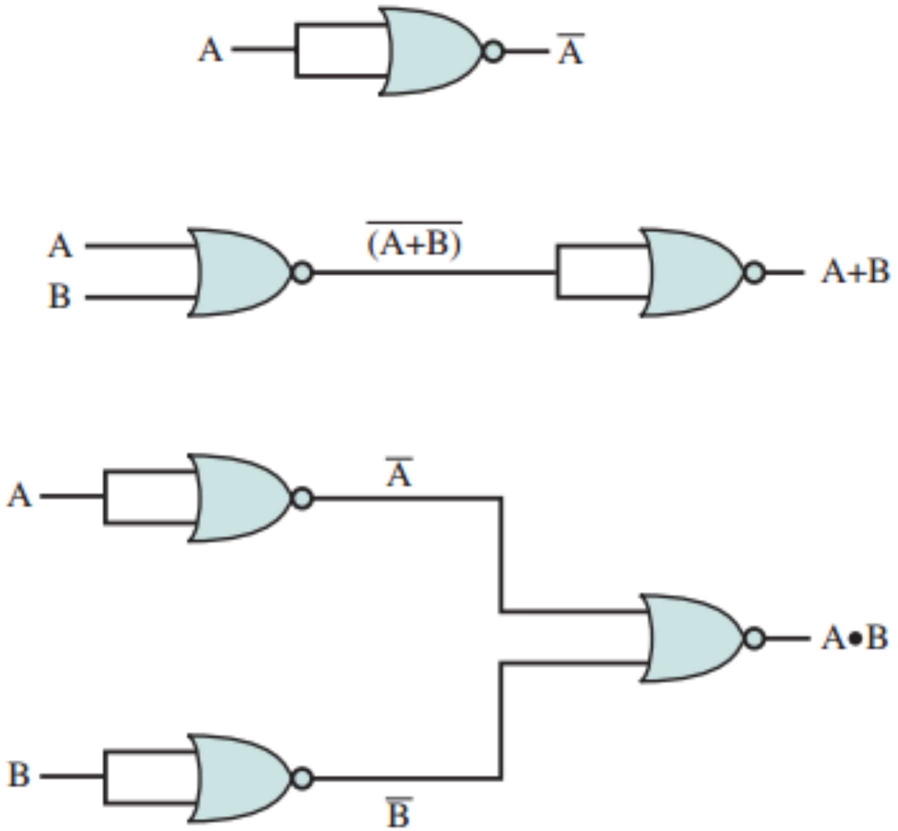
Cổng logic (tt)

- Dùng cổng NAND thực hiện các cổng khác:



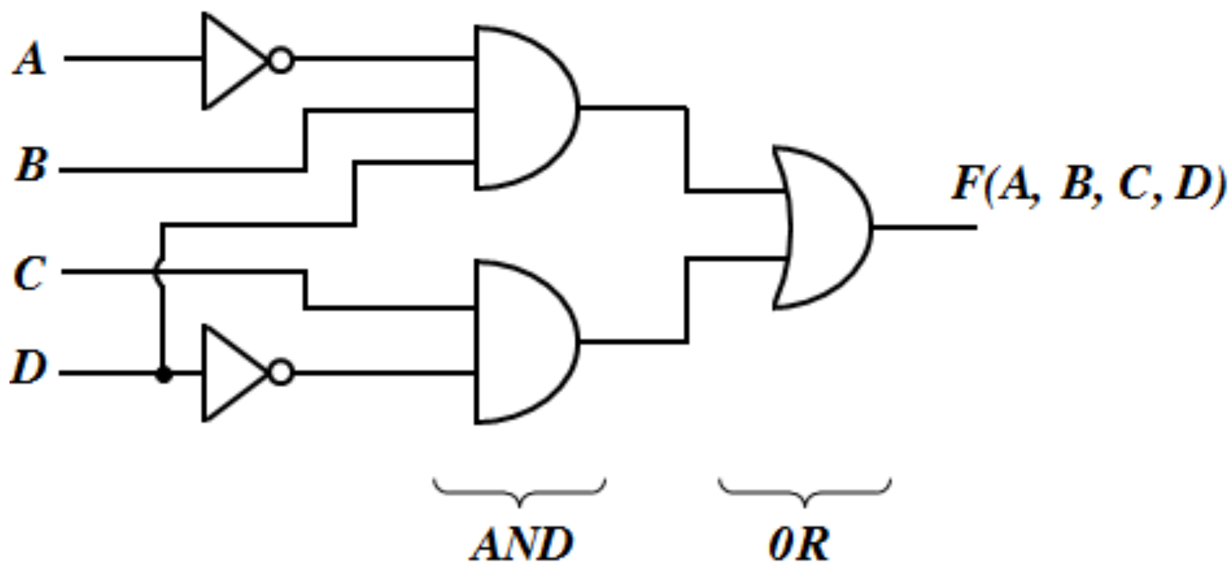
Cổng logic (tt)

- Dùng cổng NOR thực hiện các cổng khác:



Thực hiện hàm Boole bằng cổng logic

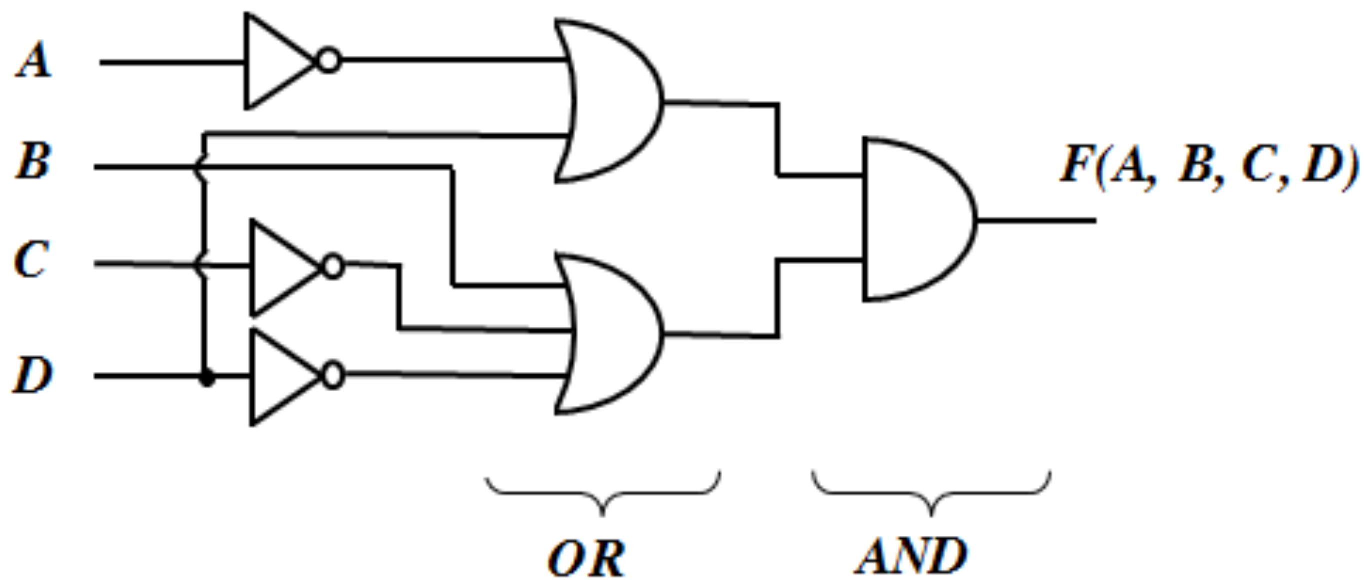
- Cấu trúc cổng AND – OR:
  - Thực hiện hàm Boole biểu diễn theo dạng chính tắc 1 (tổng các tích)
  - Ví dụ:



$$F(A, B, C, D) = \overline{A}BD + C\overline{D}$$

Thực hiện hàm Boole bằng cổng logic (tt)

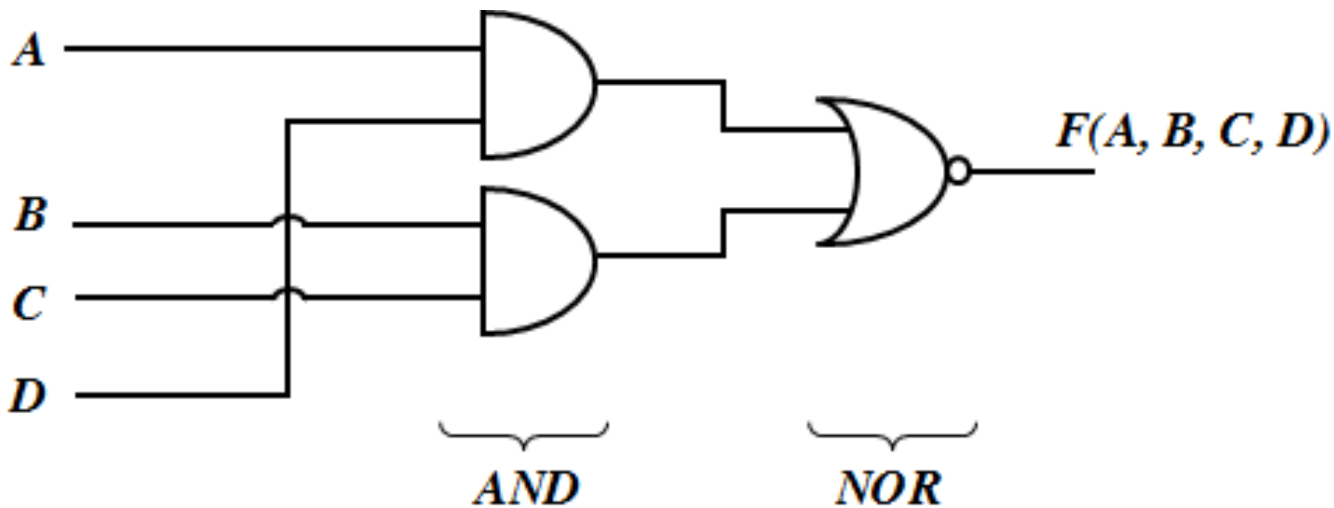
- Cấu trúc cổng OR – AND:
  - Thực hiện hàm Boole biểu diễn theo dạng chính tắc 2 (tích các tổng)
  - Ví dụ:



$$F(A,B,C,D) = (\bar{A} + D)(B + \bar{C} + \bar{D})$$

Thực hiện hàm Boole bằng cổng logic (tt)

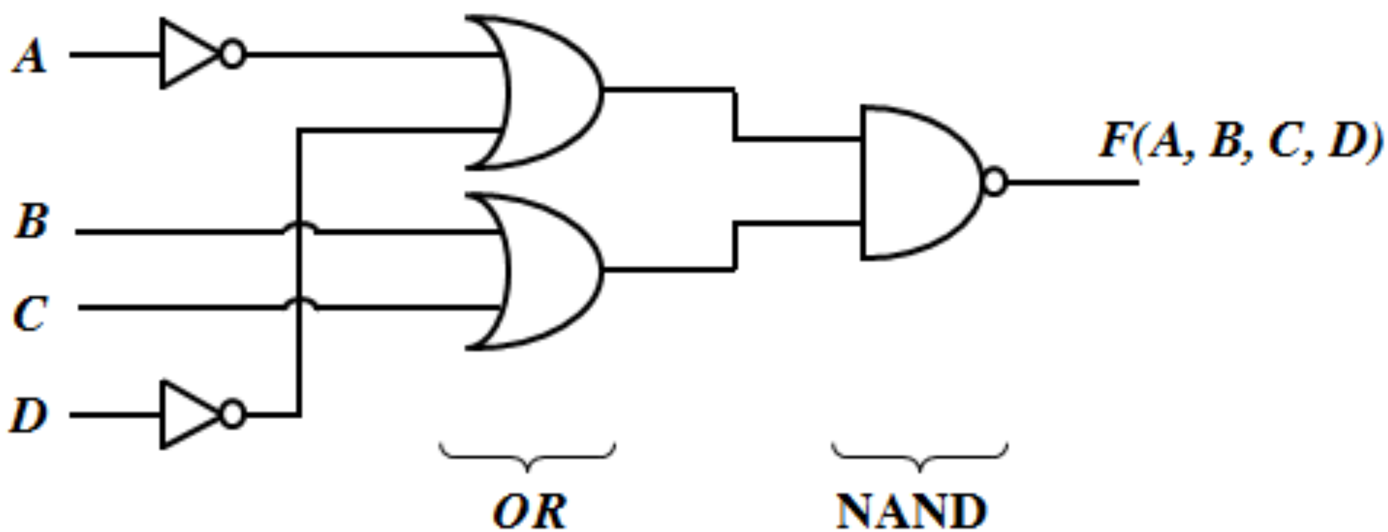
- Cấu trúc cổng AND – OR – INVERTER (AOI):
  - Thực hiện hàm Boole biểu diễu diễn theo dạng bù của tổng các tích
  - Ví dụ:



$$F(A, B, C, D) = \overline{AD + BC}$$

Thực hiện hàm Boole bằng cổng logic (tt)

- Cấu trúc cổng OR – AND – INVERTER (OAI):
  - Thực hiện hàm Boole biểu diễu diễn theo dạng bù của tích các tổng
  - Ví dụ:



$$F(A, B, C, D) = \overline{(\bar{A} + \bar{D})(B + C)}$$

Bài tập

- 9.5-9.8, 9.10-9.16, 10.1, 10.2, 10.10, 10.11, 10.23, 10.24, 10.27, 11.1-11.6