



Khoa  
**CÔNG NGHỆ THÔNG TIN**  
ĐH Khoa học Tự nhiên TP HCM

# Bài 02: Số nguyên

**Phạm Tuấn Sơn**

**[ptson@fit.hcmus.edu.vn](mailto:ptson@fit.hcmus.edu.vn)**

# Hệ cơ số 10

- $A = 123 = 100 + 20 + 3 = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$

- Tổng quát số hệ cơ số  $q$

$$X_{n-1} \dots X_1 X_0 = X_{n-1} \times q^{n-1} + \dots + X_1 \times q^1 + X_0 \times q^0$$

Mỗi chữ số  $X_i$  lấy từ tập  $X$  có  $q$  phần tử

- $q=2, X=\{0,1\}$  : hệ nhị phân (binary)
- $q=8, X=\{0,1,2,\dots,7\}$  : hệ bát phân (octal)
- $q=10, X=\{0,1,2,\dots,9\}$  : hệ thập phân (decimal)
- $q=16, X=\{0,1,2,\dots,9,A,B,\dots,F\}$  : hệ thập lục phân (hexadecimal)

$$A = 123d = 01111011b = 173o = 7Bh$$

# Hệ nhị phân

$$X_{n-1} \dots X_1 X_0, X = \{0, 1\}$$

- Được dùng nhiều trong máy tính. Tại sao ?
- n gọi là chiều dài bit của số đó
- Bit trái nhất  $X_{n-1}$  là bit có giá trị nhất (MSB)
- Bit phải nhất  $X_0$  là bit ít có giá trị nhất (LSB)
- Giá trị thập phân:

$$X_{n-1} \times 2^{n-1} + \dots + X_1 \times 2^1 + X_0 \times 2^0$$

Phạm vi biểu diễn: từ 0 đến  $2^n - 1$

- Để chuyển đổi sang hệ 16, chỉ cần gom từng nhóm 4 bit từ phải sang trái

• Ví dụ:  $A = 01111011b$   
 $= \underbrace{0111}_7 \underbrace{1011}_B h$

0000 – 0	1000 – 8
0001 – 1	1001 – 9
0010 – 2	1010 – A
0011 – 3	1011 – B
0100 – 4	1100 – C
0101 – 5	1101 – D
0110 – 6	1110 – E
0111 – 7	1111 – F

# Bits có thể biểu diễn mọi thứ !

- Ký tự?

- 26 ký tự  $\Rightarrow$  5 bits ( $2^5 = 32$ )
- Ký tự hoa/ thường + dấu  
 $\Rightarrow$  7 bits (in 8) (“ASCII”)
- Bảng mã chuẩn cho tất cả ngôn ngữ trên thế giới  
 $\Rightarrow$  8,16,32 bits (“Unicode”) [www.unicode.com](http://www.unicode.com)

- Giá trị luận lý (logic)?

- 0  $\Rightarrow$  False, 1  $\Rightarrow$  True

- Màu sắc ? Ví dụ: **Red (00)** **Green (01)** **Blue (11)**

- Địa chỉ ? Lệnh ?

- Bộ nhớ: N bits  $\Leftrightarrow 2^N$  ô nhớ



# Biểu diễn số âm

- Số không dấu (unsigned number)



- Lượng dấu (sign and magnitude)

– Qui định MSB là dấu



11111 ... 10001 10000

0x00000000 và 0x80000000 ???

- Bù 1 (One's Complement)

– Lấy bit bù



10000 ... 11110 11111

0x00000000 và 0xFFFFFFF<sub>5</sub> ???

Phạm vi biểu diễn

# Số bù 2

- Khắc phục vấn đề có 2 biểu diễn số 0 khác nhau?
  - 0000 và 1111 ?
  - Lấy bù rồi cộng thêm 1
- Như số lượng dấu và số bù 1, số bắt đầu bằng 0 là số dương, số bắt đầu bằng 1 là số âm
  - 000000...xxx :  $\geq 0$ , 111111...xxx :  $< 0$
  - 1...1111 là -1, không phải -0 (như số bù 1)
- Giá trị thập phân của biểu diễn dạng bù 2
$$X_{n-1} \times (-2^{n-1}) + X_{n-2} \times (2^{n-2}) + \dots + X_1 \times 2^1 + X_0 \times 2^0$$

Phạm vi biểu diễn: từ  $-2^{n-1}$  tới  $2^{n-1} - 1$

Ví dụ:  $11010110 = -2^7 + 2^6 + 2^4 + 2^2 + 2^1 = -42$

## Ví dụ số bù 2

$$+123 = 01111011b$$

$$-123 = 10000101b$$

$$0 = 00000000b$$

$$-1 = 11111111b$$

$$-2 = 11111110b$$

$$-3 = 11111101b$$

$$-127 = 10000001b$$

$$-128 = 10000000b$$

Đổi dấu:

$$-3 \rightarrow +3 \rightarrow -3$$

$$x : 1101 \text{ b}$$

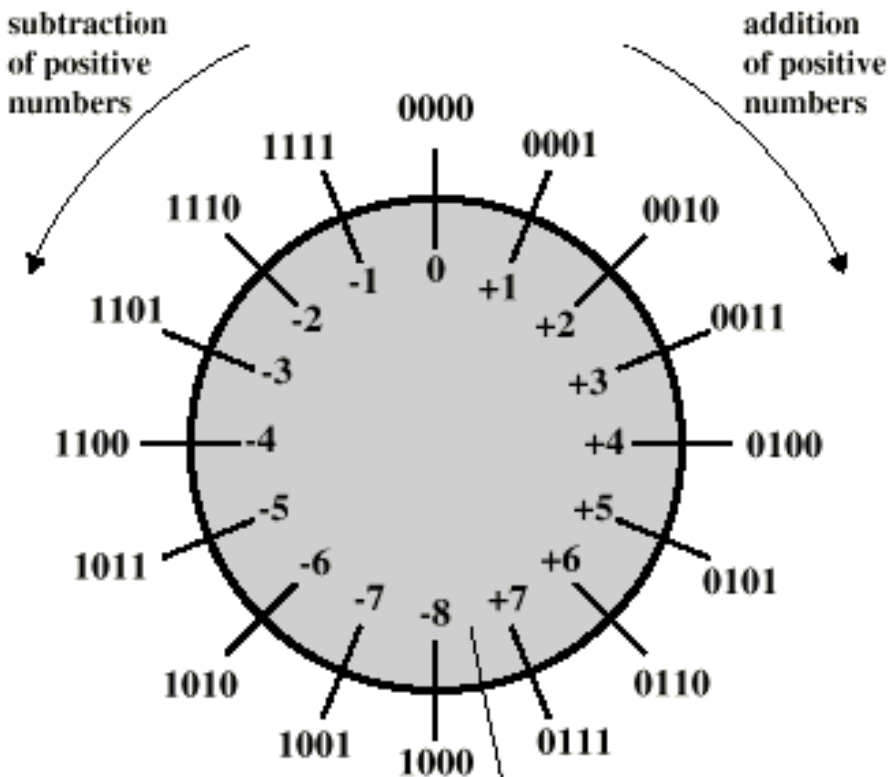
$$x' : 0010 \text{ b}$$

$$+1 : 0011 \text{ b}$$

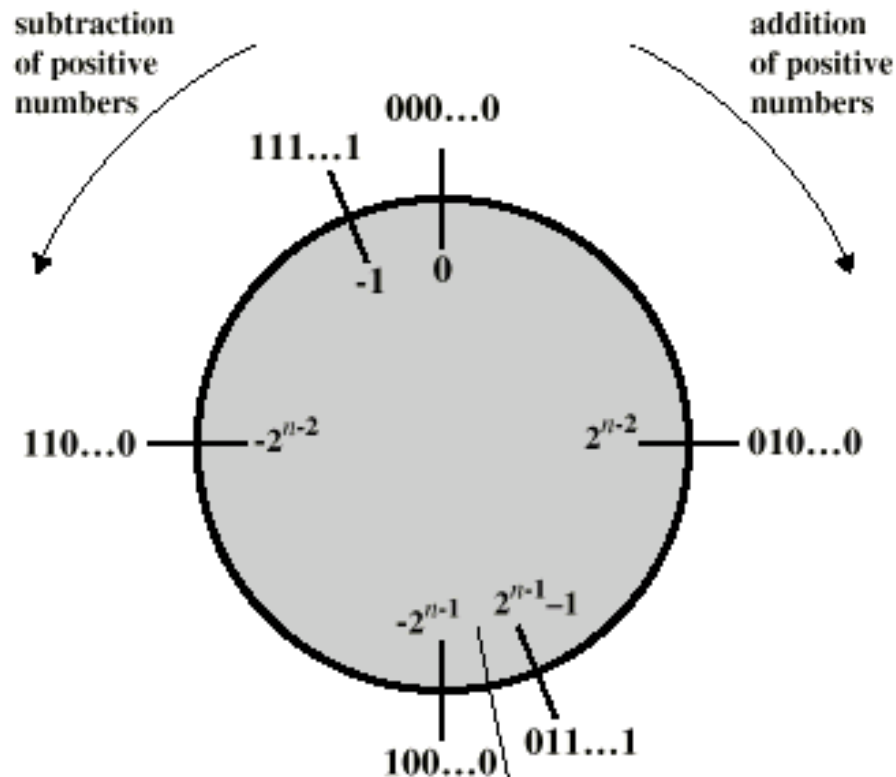
$$()': 1100 \text{ b}$$

$$+1 : 1101 \text{ b}$$

# Phạm vi biểu diễn số bù 2



(a) 4-bit numbers



(b) n-bit numbers



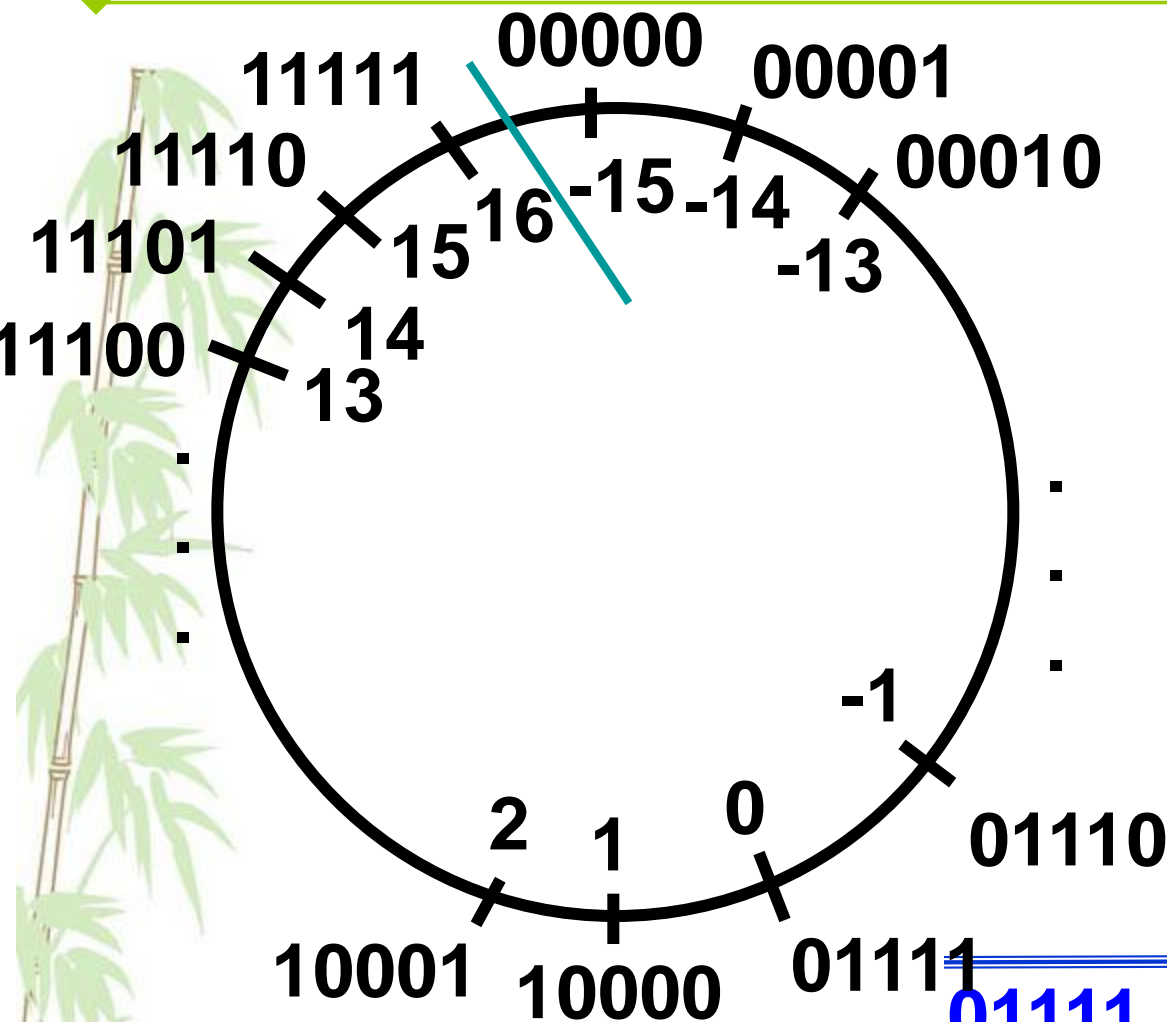
# Sign extension

- Chuyển số bù 2 từ biểu diễn  $n$  bit thành biểu diễn  $m$  bit (với  $m > n$ )
- Giá trị của các bit từ  $n+1$  tới  $m$  là giá trị của MSB
  - Chuyển giá trị -4 từ biểu diễn 16-bit thành biểu diễn 32-bit:

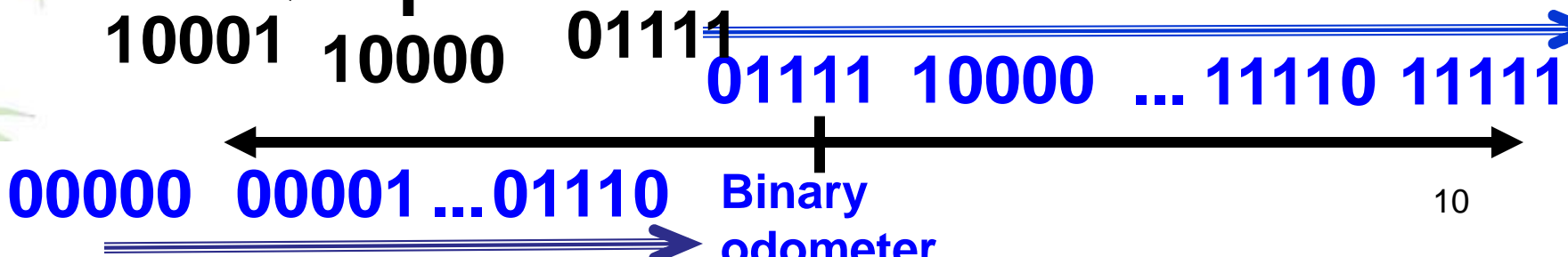
1111 1111 1111 1100<sub>two</sub>

1111 1111 1111 1111 1111 1111 1111 1100<sub>two</sub>

# Biểu diễn Bias số N=5 bit



- Bias cho số N bits là  $(2^{N-1}-1)$
- Giá trị = unsigned - bias
- 1 số zero
- Bao nhiêu số dương?



# Biểu diễn số BCD

- Quy tắc:

- Biểu diễn thành từng nibble (4bit) tương ứng với số Decimal

Decimal	BCD		Decimal	BCD
0	0000		5	0101
1	0001		6	0110
2	0010		7	0111
3	0011		8	1000
4	0100		9	1001

- Số dương +BCD → thêm một số 0 vào vào số BCD
- Số nguyên âm -BCD → số bù 10 của số +BCD.
  - Số bù 10: số bù 9 cộng thêm 1
  - Số bù 9: lấy 9 trừ cho từng số hạng trong số BCD
  - Ví dụ:  $+25_{\text{BCD}} = 0000\ 0010\ 0101$   
 $-25_{\text{BCD}} = 1001\ 0111\ 0101$

# AND, OR, NOT, XOR

AND	0	1	OR	0	1	XOR	0	1		0	1
0	0	0	0	0	1	0	0	1	NOT	1	0
1	0	1	1	1	1	1	1	0			

$$\begin{array}{r}
 \text{AND} \quad 11010011 \\
 \quad \quad 00001111 \\
 \hline
 \quad \quad 00000011
 \end{array}$$

$$\begin{array}{r}
 \text{OR} \quad 00000011 \\
 \quad \quad 01100000 \\
 \hline
 \quad \quad 01100011
 \end{array}$$

$$\begin{array}{r}
 \text{XOR} \quad 01100011 \\
 \quad \quad 01100011 \\
 \hline
 \quad \quad 00000000
 \end{array}$$

$$\begin{array}{r}
 \text{NOT} \quad 11010011 \\
 \hline
 = \quad 00101100
 \end{array}$$

# Sử dụng phép AND

- Nhận xét: bit nào `and` với 0 sẽ ra 0, `and` với 1 sẽ ra chính nó.
- Phép `and` được sử dụng để giữ lại giá trị 1 số bit, trong khi xóa tất cả các bit còn lại. Bit nào cần giữ giá trị thì `and` với 1, bit nào không quan tâm thì `and` với 0. Dãy bit có vai trò này gọi là mặt nạ (**mask**).

– Ví dụ:

'a' (61h)

Mask (DFh)

0110 0001

1101 1111

– Kết quả sau khi thực hiện `and`:

'A' (41h)

0100 0001

– Ý nghĩa: chuyển từ ký tự thường sang ký tự hoa

# Sử dụng các phép OR

- Nhận xét: bit nào `or` với 1 sẽ ra 1, `or` với 0 sẽ ra chính nó.
- Phép `or` được sử dụng để bật lên 1 số bit, trong khi giữ nguyên giá trị tất cả các bit còn lại. Bit nào cần bật lên thì `or` với 1, bit nào không quan tâm thì `or` với 0.

– Ví dụ:

1 (01h)

Mask (30h)

– Kết quả sau khi thực hiện `or`:

'1' (31h)

– Ý nghĩa: chuyển từ số sang ký tự số

0000 0001

0011 0000

0100 0001

# Phép dịch bit và phép quay



(a) Logical right shift



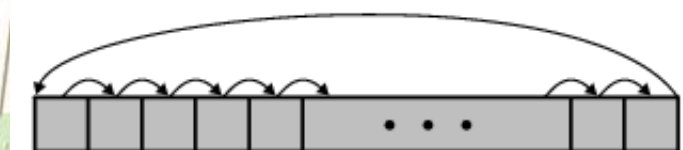
(b) Logical left shift



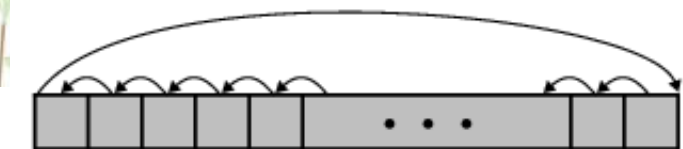
(c) Arithmetic right shift



(d) Arithmetic left shift



(e) Right rotate



(f) Left rotate

Input	Operation	Result
01010101 (85)	Logical right shift (2 bits)	00010101 (21 = $85/2^2$ )
01010101 (85)	Logical left shift (1 bit)	10101010 (170 = $85*2^1$ )
11101010 (-22)	Arithmetic right shift (2 bits)	11111010 $-6 = (-22)/2^2$
11101010 (-22)	Arithmetic left shift (1 bit)	11010100 $-44 = (-22)*2^1$
10100110	Right rotate (3 bits)	11010100
10100110	Left rotate (3 bits)	00110101

# Ví dụ

$$X = 00001000 \text{ b} = 8 \text{ d}$$

$$X \text{ shl } 2 = 00100000 \text{ b} = 32 \text{ d}$$

$$(X \text{ shl } 2) \text{ or } X = 00101000 \text{ b} = 40 \text{ d}$$

$$Y = 01001010 \text{ b} = 74 \text{ d}$$

$$((Y \text{ and } 0Fh) \text{ shl } 4) = 10100000$$

or

$$((Y \text{ and } F0h) \text{ shr } 4) = 00000100$$

=

$$10100100 \text{ b} = 164 \text{ d} = -92 \text{ d}$$



# Phép cộng

- $n=4$

$$\begin{array}{r} 1001 = -7 \\ + 0101 = 5 \\ \hline 1110 = -2 \end{array}$$

(a)  $(-7) + (+5)$

$$\begin{array}{r} 1100 = -4 \\ + 0100 = 4 \\ \hline 10000 = 0 \end{array}$$

(b)  $(-4) + (+4)$

$$\begin{array}{r} 0011 = 3 \\ + 0100 = 4 \\ \hline 0111 = 7 \end{array}$$

(c)  $(+3) + (+4)$

$$\begin{array}{r} 1100 = -4 \\ + 1111 = -1 \\ \hline 11011 = -5 \end{array}$$

(d)  $(-4) + (-1)$

$$\begin{array}{r} 0101 = 5 \\ + 0100 = 4 \\ \hline 1001 = \text{Overflow} \end{array}$$

(e)  $(+5) + (+4)$

$$\begin{array}{r} 1001 = -7 \\ + 1010 = -6 \\ \hline 10011 = \text{Overflow} \end{array}$$

(f)  $(-7) + (-6)$

# Phép trừ

•  $n=4$

$$\begin{array}{r} 0010 = 2 \\ + 1001 = -7 \\ \hline 1011 = -5 \end{array}$$

(a)  $M = 2 = 0010$   
 $S = 7 = 0111$   
 $-S = 1001$

$$\begin{array}{r} 0101 = 5 \\ + 1110 = -2 \\ \hline 10011 = 3 \end{array}$$

(b)  $M = 5 = 0101$   
 $S = 2 = 0010$   
 $-S = 1110$

$$\begin{array}{r} 1011 = -5 \\ + 1110 = -2 \\ \hline 11001 = -7 \end{array}$$

(c)  $M = -5 = 1011$   
 $S = 2 = 0010$   
 $-S = 1110$

$$\begin{array}{r} 0101 = 5 \\ + 0010 = 2 \\ \hline 0111 = 7 \end{array}$$

(d)  $M = 5 = 0101$   
 $S = -2 = 1110$   
 $-S = 0010$

$$\begin{array}{r} 0111 = 7 \\ + 0111 = 7 \\ \hline 1110 = \text{Overflow} \end{array}$$

(e)  $M = 7 = 0111$   
 $S = -7 = 1001$   
 $-S = 0111$

$$\begin{array}{r} 1010 = -6 \\ + 1100 = -4 \\ \hline 10110 = \text{Overflow} \end{array}$$

(f)  $M = -6 = 1010$   
 $S = 4 = 0100$   
 $-S = 1100$

# Tràn số


- Tràn số xảy ra khi kết quả phép tính vượt quá độ chính xác giới hạn cho phép (của máy tính).
- Dấu hiệu nhận biết tràn số đối với số không dấu:

- Nhớ ra 1 bit
- Ví dụ (số nguyên không dấu 4-bit):

+15	1111
<u>+3</u>	<u>0011</u>
+18	10010

- Nhưng không có chỗ để chứa cả 5 bit nên chỉ chứa kết quả 4 bit 0010, là +2 → sai.
- Dấu hiệu nhận biết tràn số đối với số có dấu:
  - Dương cộng dương ra kết quả âm và âm cộng âm ra kết quả dương
  - Dương cộng âm và âm cộng dương không bao giờ cho kết quả tràn số
- Một số ngôn ngữ có khả năng phát hiện tràn số (Ada), một số không (C)

# Phép nhân – Số không dấu

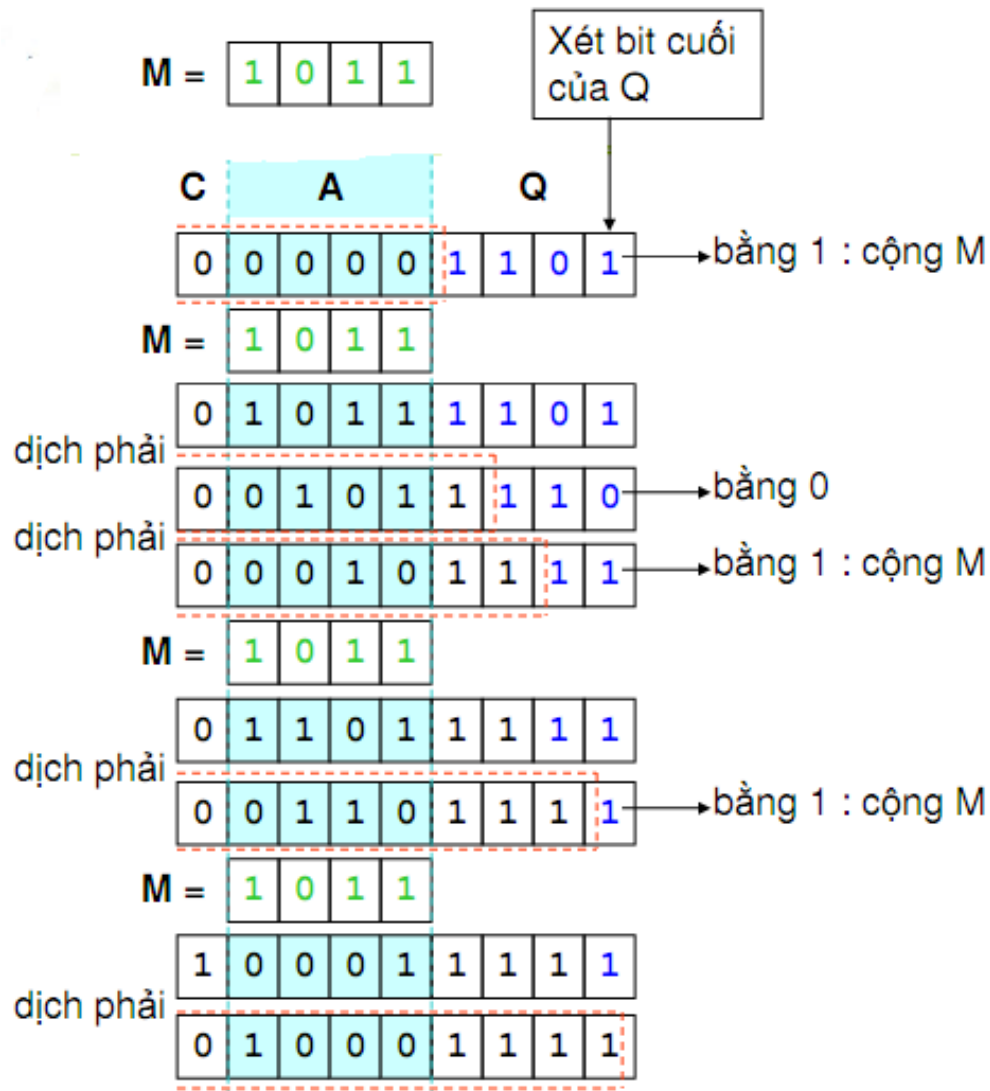
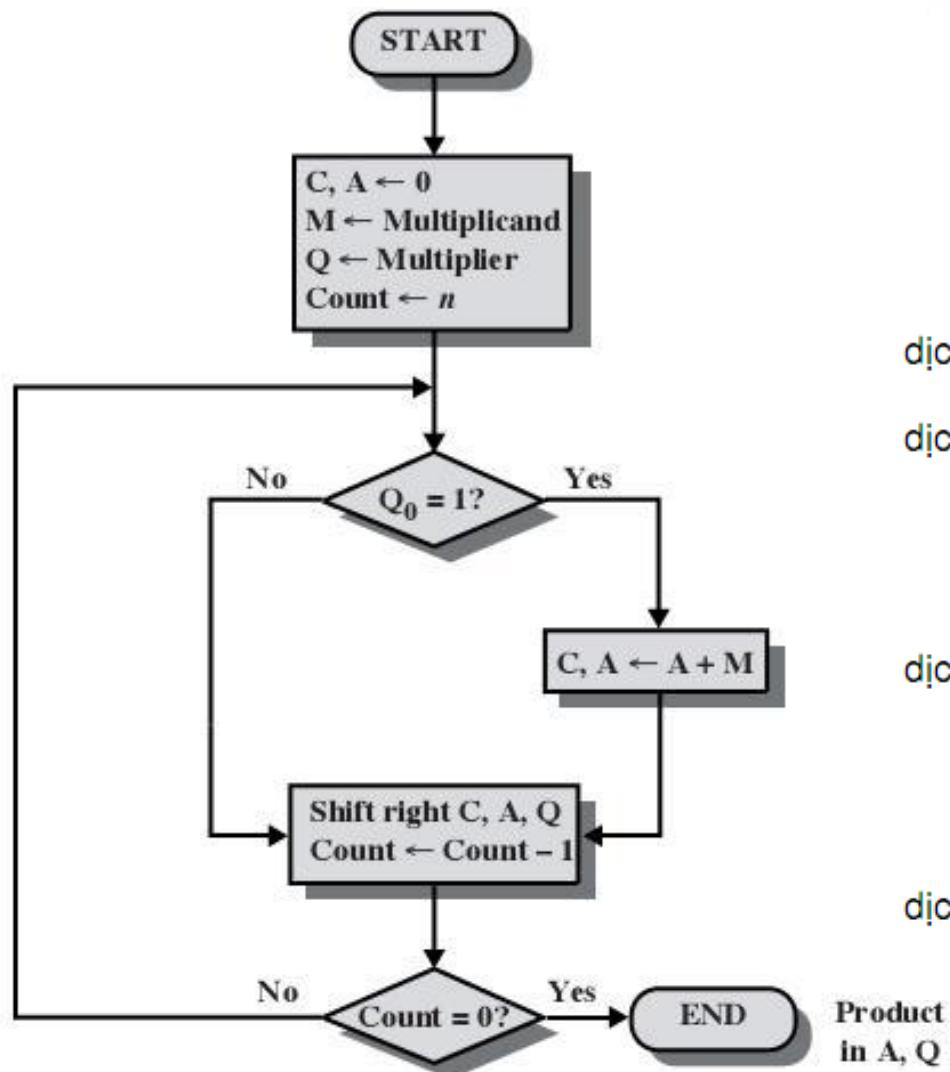


$$\begin{array}{r}
 1011 \\
 \times 1101 \\
 \hline
 1011 \\
 0000 \\
 1011 \\
 1011 \\
 \hline
 10001111
 \end{array}$$

$1011 = 11$   
 $1101 = 13$   
 $10001111 = 143$

$$\begin{array}{r}
 1011 \\
 \times 1101 \\
 \hline
 00000000 \\
 + 1011 \\
 \hline
 00001011 \\
 + 0000 \\
 \hline
 00001011 \\
 + 1011 \\
 \hline
 00110111 \\
 + 1011 \\
 \hline
 10001111
 \end{array}$$

# Thuật toán nhân không dấu



# Phép nhân – Số bù 2

$\begin{array}{r} 1001 \quad (-7) \\ \times 0011 \quad (3) \\ \hline 11111001 \quad (-7) \times 2^0 = (-7) \\ 11110010 \quad (-7) \times 2^1 = (-14) \\ \hline 11101011 \quad (-21) \end{array}$	$\begin{array}{r} 1001 \quad (-7) \\ \times 1100 \quad (-4) \\ \hline 11100100 \quad (-28) \\ 11001000 \quad (-56) \\ \hline 10101100 \quad (-84) ??? \end{array}$
--	--

- Tại sao ?
  - Thừa số 2:  $1100 \neq -(2^3 + 2^2)$  ( $1100 = -2^2$ )
- Giải pháp 1
  - Chuyển thừa số 2 thành số dương
  - Nhân theo thuật toán nhân không dấu
  - Nếu khác dấu, đổi dấu
- Giải pháp 2
  - Thuật toán Booth

# Thuật toán Booth – Ý tưởng

## Positive

$$2^n + 2^{n-1} + \dots + 2^{n-K} = 2^{n+1} - 2^{n-K}$$

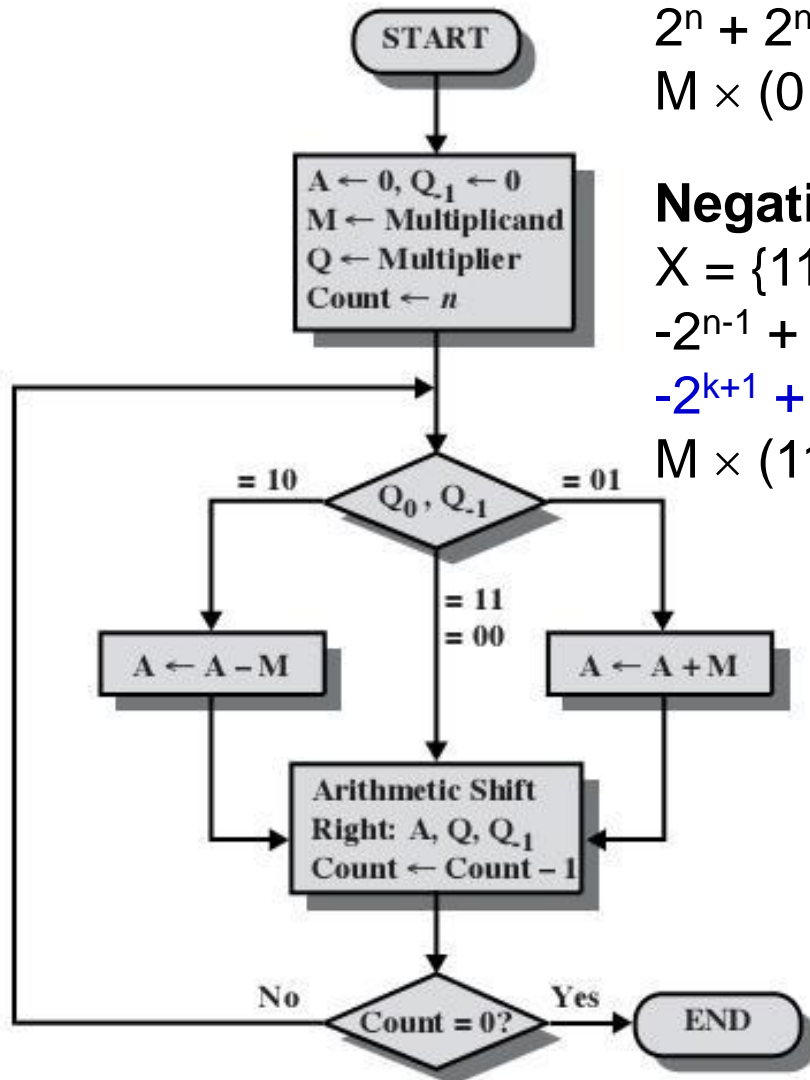
$$\begin{aligned} M \times (01111010) &= M \times (2^6 + 2^5 + 2^4 + 2^3 + 2^1) \\ &= M \times (2^7 - 2^3 + 2^2 - 2^1) \end{aligned}$$

## Negative

$$X = \{111..10x_{k-1}x_{k-2}\dots x_1x_0\}$$

$$\begin{aligned} -2^{n-1} + 2^{n-2} + \dots + 2^{k+1} + (x_{k-1} \times 2^{k-1}) + \dots + (x_0 \times 2^0) &= \\ -2^{k+1} + (x_{k-1} \times 2^{k-1}) + \dots + (x_0 \times 2^0) &= \end{aligned}$$

$$\begin{aligned} M \times (11111010) &= M \times (-2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^1) \\ &= M \times (-2^3 + 2^1) \\ &= M \times (-2^3 + 2^2 - 2^1) \end{aligned}$$



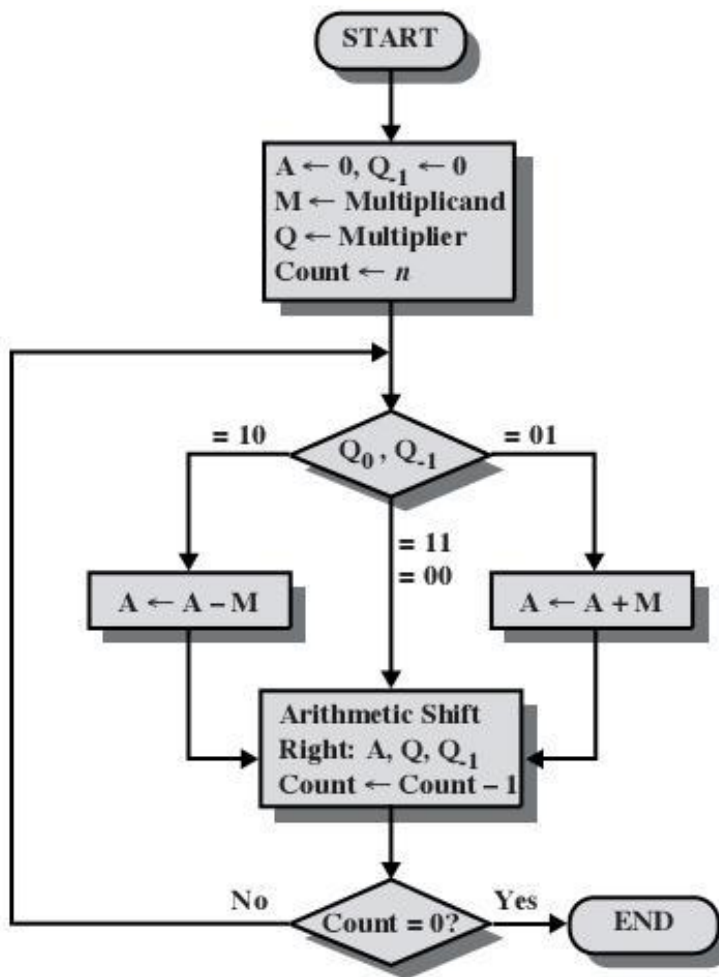


# Thuật toán Booth – Cơ sở thuật toán

- Bước 0:  $A = (0 + (Q_{-1} - Q_0) \cdot M)$
- Bước 1:  $A = (0 + (Q_{-1} - Q_0) \cdot M + (Q_0 - Q_1) \cdot M \cdot 2)$   
 $= M \cdot (Q_{-1} - Q_0 + Q_0 \cdot 2 - Q_1 \cdot 2)$
- Bước 2:  $A = (M \cdot (Q_{-1} - Q_0 + Q_0 \cdot 2 - Q_1 \cdot 2) + (Q_1 - Q_2) \cdot M \cdot 2^2)$   
 $= M \cdot (Q_{-1} - Q_0 + Q_0 \cdot 2 - Q_1 \cdot 2 + Q_1 \cdot 2^2 - Q_2 \cdot 2^2)$
- Bước 3:  
 $A = M \cdot (Q_{-1} - Q_0 + Q_0 \cdot 2 - Q_1 \cdot 2 + Q_1 \cdot 2^2 - Q_2 \cdot 2^2 + Q_2 \cdot 2^3 - Q_3 \cdot 2^3)$   
 $= M \cdot (Q_{-1} + Q_0 + Q_1 \cdot 2 + Q_2 \cdot 2^2 - Q_3 \cdot 2^3)$
- Bước n-1:  
 $A = M \cdot (Q_{-1} + Q_0 + Q_1 \cdot 2 + Q_2 \cdot 2^2 + Q_3 \cdot 2^3 + \dots + Q_{n-2} \cdot 2^{n-2} - Q_{n-1} \cdot 2^{n-1})$   
Vì  $Q_{-1} = 0$  và  $Q_{n-1}$  chính là bit xác định dấu nên phần trong dấu ngoặc chính là Q. Vậy  $A = M \cdot Q$



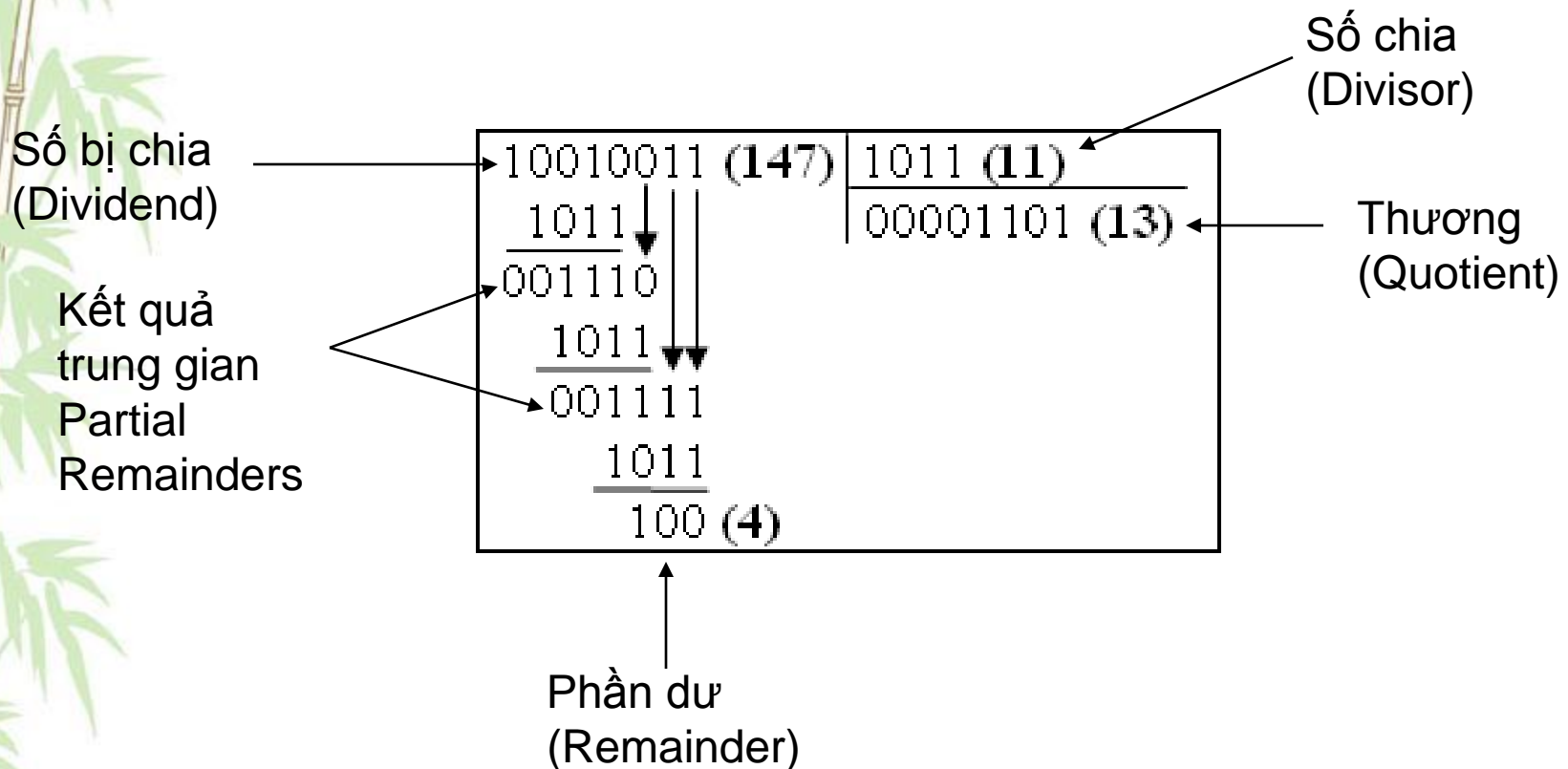
# Thuật toán Booth – Ví dụ



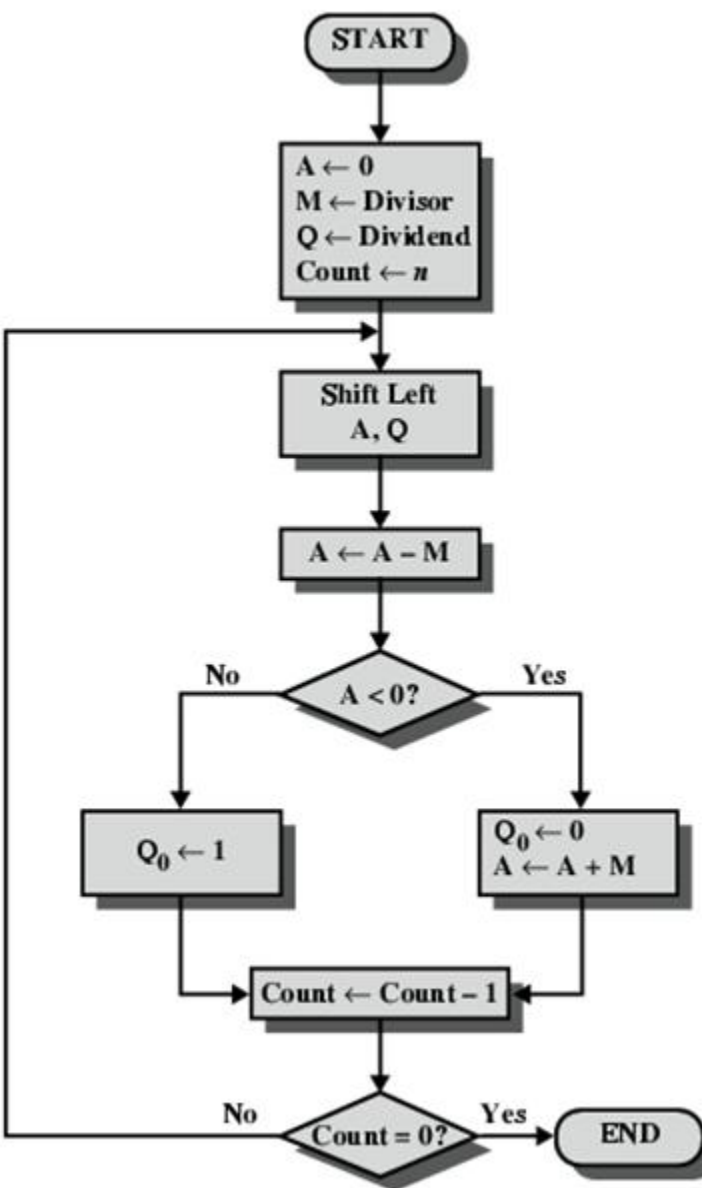
	A	Q	Q <sub>-1</sub>	M
Khởi đầu	0000	1101	0	0111
Bước 0: A=A-M	1001	1101	0	0111
shift	1100	1110	1	0111
Bước 1: A=A+M	0011	1110	1	0111
shift	0001	1111	0	0111
Bước 2: A=A-M	1010	1111	0	0111
shift	1101	0111	1	0111
Bước 3: shift	1110	1011	1	0111

Kết quả 11101011 = -21

# Phép chia – Số không dấu



# Thuật toán chia không dấu



A	Q	M = 0011
0000	0111	Initial value
0000	1110	Shift
1101		Subtract
0000	1110	Restore
0001	1100	Shift
1110		Subtract
0001	1100	Restore
0011	1000	Shift
0000		Subtract
0000	1001	Set $Q_0 = 1$
0001	0010	Shift
1110		Subtract
0001	0010	Restore

(7)/(3)

Quotient in Q  
Remainder in A

# Phép chia – Số bù 2

- Thực hiện như phép chia không dấu
- Nếu số chia và số bị chia khác dấu  $\rightarrow$  đổi dấu thương

A	Q	M = 1101
0000	0111	Initial value
0000	1110	Shift
1101		Add
0000	1110	Restore
0001	1100	Shift
1110		Add
0001	1100	Restore
0011	1000	Shift
0000		Add
0000	1001	Set $Q_0 = 1$
0001	0010	Shift
1110		Add
0001	0010	Restore

(7)/(-3)

# Bài tập

- Hãy trình bày phép nhân 2 số nguyên  
 $(-127) \times (-5)$
- Hãy trình bày phép chia 2 số nguyên  
 $(-7) / (-3)$
- Phép toán trên các loại số khác: số bù 1, số bias, số BCD,...



# Tham khảo

- Chương 3, P&H

