

# Chương 2: Biểu diễn thông tin trong máy tính

- 2.1. Các hệ đếm cơ bản
- 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính
- 2.3. Biểu diễn số nguyên
- 2.4. Thực hiện các phép toán số học với số nguyên
- 2.5. Số dấu phẩy động
- 2.6. Biểu diễn ký tự

## 2.1. Các hệ đếm cơ bản

- Hệ thập phân (Decimal System)
  - con người sử dụng
- Hệ nhị phân (Binary System)
  - máy tính sử dụng
- Hệ mười sáu (Hexadecimal System)
  - dùng để viết gọn cho số nhị phân
- Hệ bát phân (Octal System)

# 1. Hệ thập phân

## Các hệ thống số

Thập phân

Ký số

0	1	2	3	4
5	6	7	8	9

10

Quy tắc đếm

0 → 1 → 2 → ... → 9 →  
10 → 11 → 12 → ... → 19 →  
20 → 21 → 22 → ... → 29 →  
→ ...  
90 → 91 → 92 → ... → 99 →  
100 → 101 → 102 → ... → 109 →  
→ ...  
990 → 991 → 992 → ... → 999 →  
1000 → 1001 → 1002 → ... → 1009 →  
→ ...

# 1. Hệ thập phân

- Cơ số 10
- 10 chữ số: 0,1,2,3,4,5,6,7,8,9
- Dùng n chữ số thập phân có thể biểu diễn được  $10^n$  giá trị khác nhau:
- $00\dots000 = 0$
- $99\dots999 = 10^n - 1$

# Dạng tổng quát của số thập phân

- Giá trị của A được hiểu như sau:

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} \dots a_{-m}$$

$$A = a_n 10^n + a_{n-1} 10^{n-1} + \dots + a_1 10^1 + a_0 10^0 + a_{-1} 10^{-1} + \dots + a_{-m} 10^{-m}$$


$$A = \sum_{i=-m}^n a_i 10^i$$

- Ví dụ:


# Ví dụ số thập phân

$$472.38 = 4 \times 10^2 + 7 \times 10^1 + 2 \times 10^0 + 3 \times 10^{-1} + 8 \times 10^{-2}$$

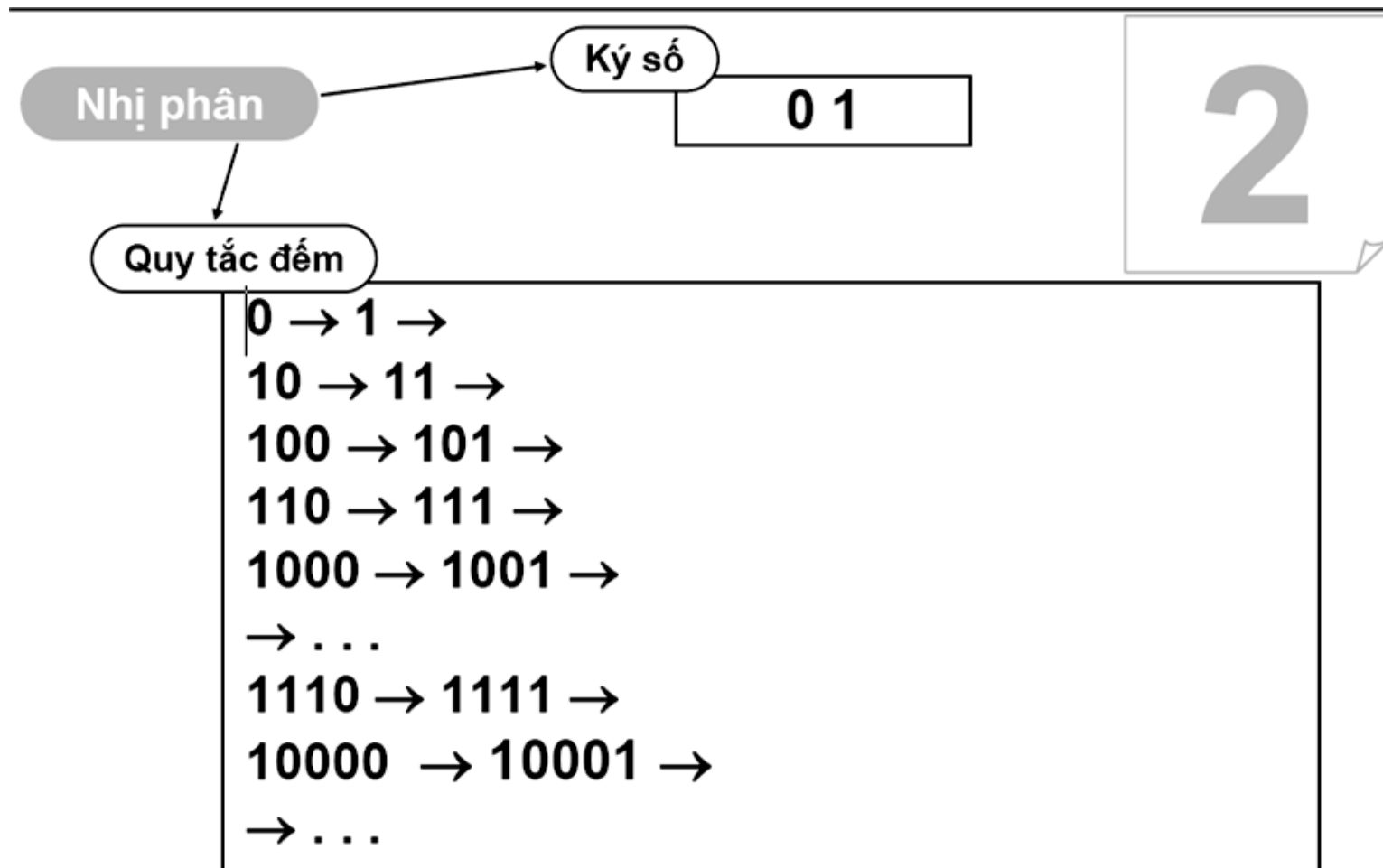
## ■ Các chữ số của phần nguyên:

- $472 : 10 = 47$  dư 2
  - $47 : 10 = 4$  dư 7
  - $4 : 10 = 0$  dư 4
- 

## ■ Các chữ số của phần thập phân:

- $0.38 \times 10 = 3.8$  phần nguyên = 3
  - $0.8 \times 10 = 8.0$  phần nguyên = 8
- 

## 2. Hệ nhị phân



## 2. Hệ nhị phân

- Cơ số 2
- 2 chữ số nhị phân: 0 và 1
- chữ số nhị phân gọi là ***bit (binary digit)***
- Bit là đơn vị thông tin nhỏ nhất
- Dùng n bit có thể biểu diễn được  $2^n$  giá trị khác nhau:
  - $00\dots000 = 0$
  - $11\dots111 = 2^n - 1$



## 2. Hệ nhị phân

- Có một số nhị phân  $A$  như sau:

$$A = a_n a_{n-1} \dots a_1 a_0 . a_{-1} \dots a_{-m}$$

- Giá trị của  $A$  được tính như sau:

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \dots + a_0 2^0 + a_{-1} 2^{-1} + \dots + a_{-m} 2^{-m}$$

- Ví dụ:

$$1101001.1 \ 0 \ 1 \ 1_{(2)}$$

$$\mathbf{6543210-1-2-3-4}$$

$$= 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$$

$$= 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$$

$$= 105.6875_{(10)}$$

# Dạng tổng quát của số nhị phân

- Có một số nhị phân A như sau:

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} \dots a_{-m}$$

- Giá trị của A được tính như sau:

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \dots + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + \dots + a_{-m} 2^{-m}$$

$$A = \sum_{i=-m}^n a_i 2^i$$

# Ví dụ số nhị phân

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 0 & 0 & 1 & . & 1 & 0 & 1 & 1 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & & -1 & -2 & -3 & -4 \end{array} \quad {}_{(2)} =$$

$$= 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$$

$$= 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$$

$$= 105.6875_{(10)}$$

## 2. Hệ nhị phân


- Chuyển đổi số nguyên thập phân sang nhị phân
  - Phương pháp 1: chia dần cho 2 rồi lấy phần dư
  - Phương pháp 2: Phân tích thành tổng của các số  $2^i \Rightarrow$  nhanh hơn

## 2. Hệ nhị phân

### ■ Phương pháp chia dần cho 2

#### ■ Ví dụ: chuyển đổi $105_{(10)}$

■	$105 : 2 =$	52	dư	1
■	$52 : 2 =$	26	dư	0
■	$26 : 2 =$	13	dư	0
■	$13 : 2 =$	6	dư	1
■	$6 : 2 =$	3	dư	0
■	$3 : 2 =$	1	dư	1
■	$1 : 2 =$	0	dư	1



#### ■ Kết quả: $105_{(10)} = 1101001_{(2)}$

# Phương pháp phân tích thành tổng của các $2^i$

■ Ví dụ 1: chuyển đổi  $105_{(10)}$

■  $105 = 64 + 32 + 8 + 1 = 2^6 + 2^5 + 2^3 + 2^0$

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
0	1	1	0	1	0	0	1

■ Kết quả:  $105_{(10)} = 0110\ 1001_{(2)}$

■ Ví dụ 2:  $17000_{(10)} = 16384 + 512 + 64 + 32 + 8$   
 $= 2^{14} + 2^9 + 2^6 + 2^5 + 2^3$

$$17000_{(10)} = 0100\ 0010\ 0110\ 1000_{(2)}$$

# Chuyển đổi số lẻ thập phân sang nhị phân

■ Ví dụ 1: chuyển đổi  $0.6875_{(10)}$


■	$0.6875 \times 2 = 1.375$	phần nguyên = 1	↓
■	$0.375 \times 2 = 0.75$	phần nguyên = 0	
■	$0.75 \times 2 = 1.5$	phần nguyên = 1	
■	$0.5 \times 2 = 1.0$	phần nguyên = 1	

■ Kết quả :  $0.6875_{(10)} = 0.1011_{(2)}$

# Chuyển đổi số lẻ thập phân sang nhị phân (tiếp)

## ■ Ví dụ 2: chuyển đổi $0.81_{(10)}$

■	0.81	x 2 =	1.62	phần nguyên	=	1
■	0.62	x 2 =	1.24	phần nguyên	=	1
■	0.24	x 2 =	0.48	phần nguyên	=	0
■	0.48	x 2 =	0.96	phần nguyên	=	0
■	0.96	x 2 =	1.92	phần nguyên	=	1
■	0.92	x 2 =	1.84	phần nguyên	=	1
■	0.84	x 2 =	1.68	phần nguyên	=	1




■  $0.81_{(10)} \approx 0.1100111_{(2)}$



# Chuyển đổi số lẻ thập phân sang nhị phân (tiếp)

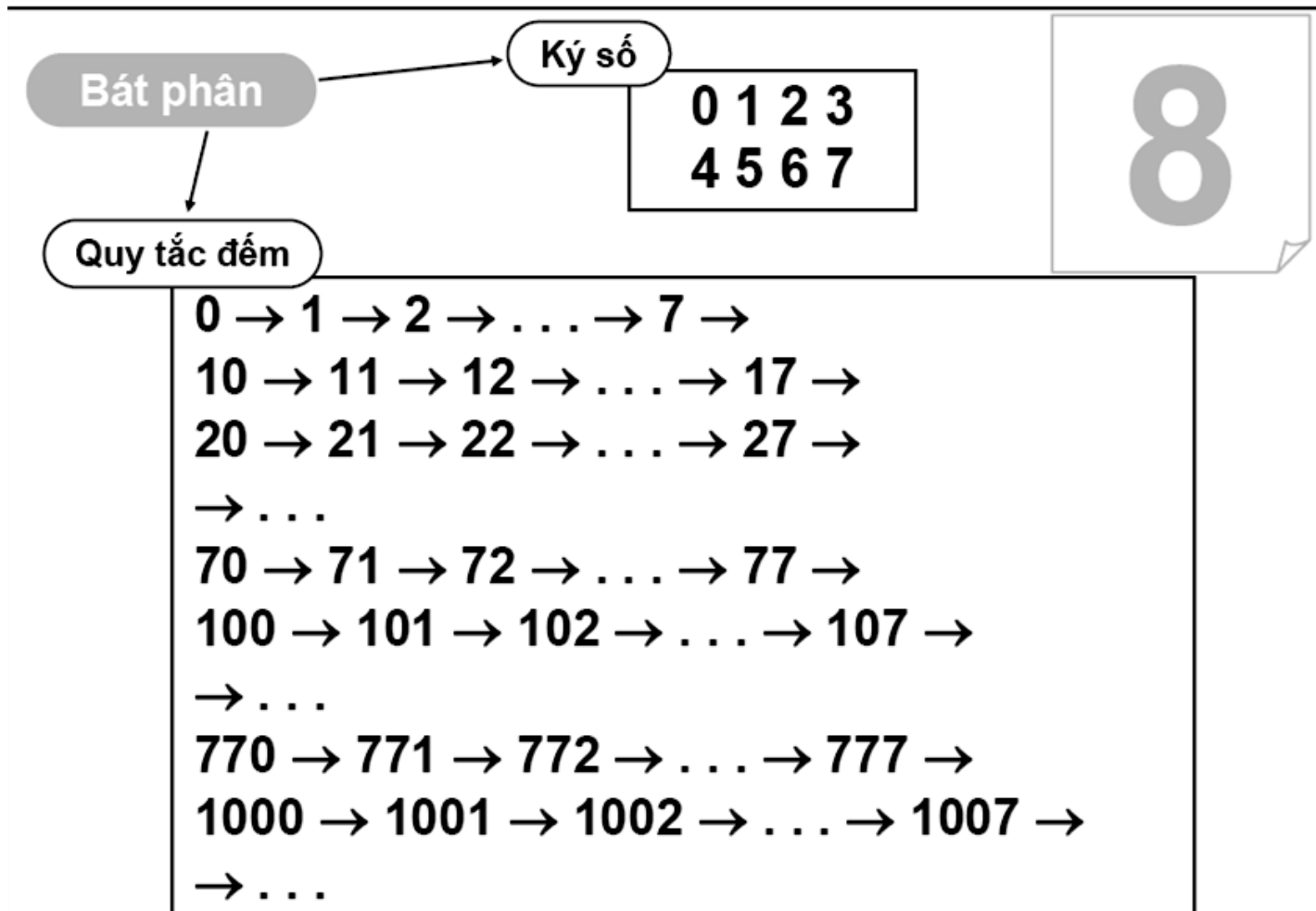
## ■ Ví dụ 3: chuyển đổi $0.2_{(10)}$

■	0.2	x 2 =	0.4	phần nguyên	=	0
■	0.4	x 2 =	0.8	phần nguyên	=	0
■	0.8	x 2 =	1.6	phần nguyên	=	1
■	0.6	x 2 =	1.2	phần nguyên	=	1
■	0.2	x 2 =	0.4	phần nguyên	=	0
■	0.4	x 2 =	0.8	phần nguyên	=	0
■	0.8	x 2 =	1.6	phần nguyên	=	1
■	0.6	x 2 =	1.2	phần nguyên	=	1

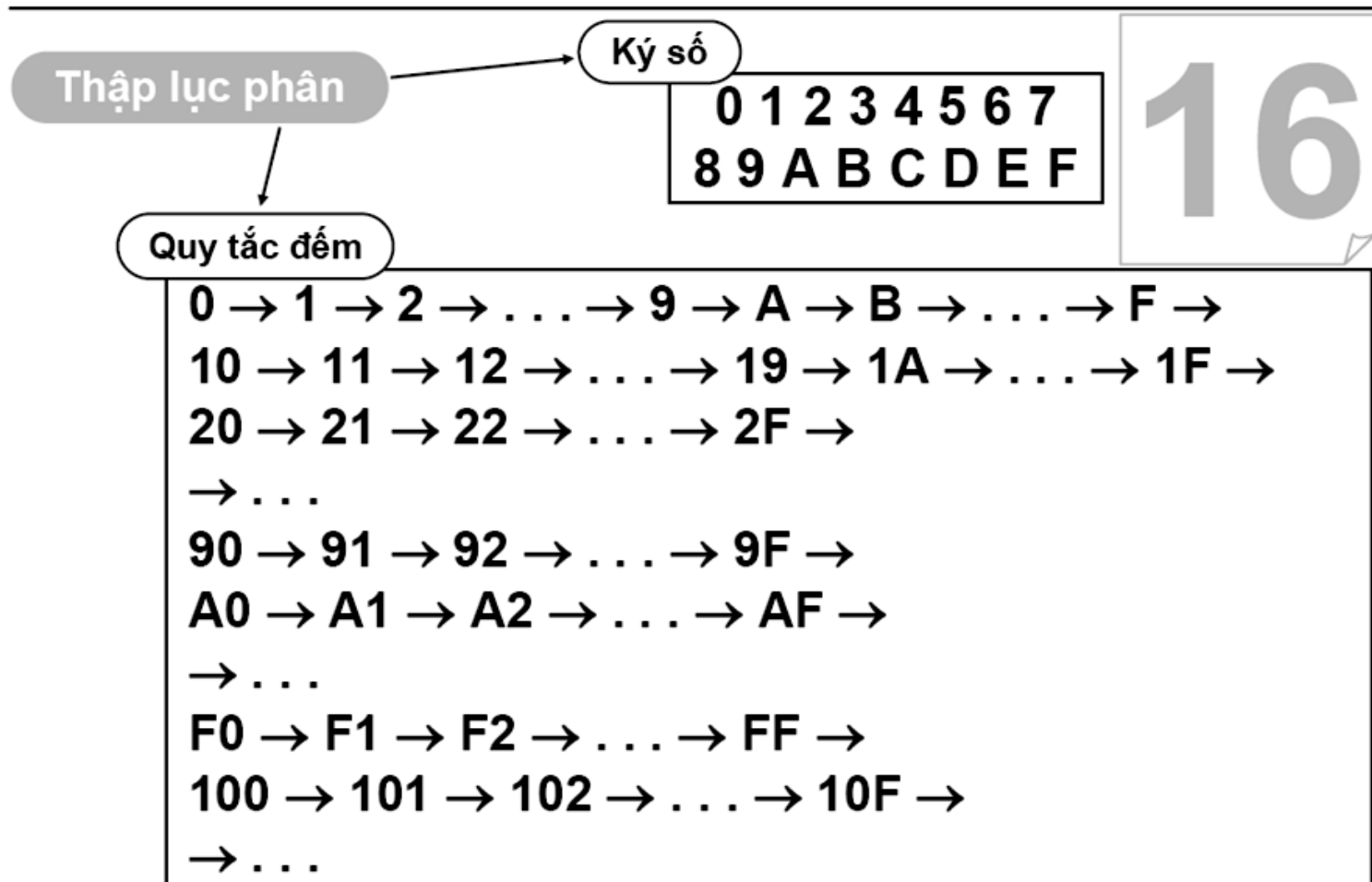


■  $0.2_{(10)} \approx 0.00110011_{(2)}$

# Hệ bát phân



### 3. Hệ mười sáu (Hexa)



### 3. Hệ mười sáu (Hexa)

- Cơ số 16
- 16 chữ số: 0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F
- Dùng để viết gọn cho số nhị phân: cứ một nhóm 4-bit sẽ được thay bằng một chữ số Hexa

### 3. Hệ mười sáu (Hexa)

- Ví dụ chuyển đổi số nhị phân → số Hexa:
- $1011\ 0011_2 = B3_{16}$
- $0000\ 0000_2 = 00_{16}$
- $0010\ 1101\ 1001\ 1010_2 = 2D9A_{16}$
- $1111\ 1111\ 1111\ 1111_2 = FFFF_{16}$

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

## 4. Chuyển đổi giữa các hệ đếm

- Chuyển đổi hệ 10  $\leftrightarrow$  16
- Chuyển đổi hệ 2  $\leftrightarrow$  8
- Chuyển đổi hệ 2  $\leftrightarrow$  16
- Chuyển đổi hệ 8  $\leftrightarrow$  16

## 4. Chuyển đổi giữa các hệ đếm

Số hệ 10	Số hệ 16	Số hệ 2
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
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

(0+4+2+0)

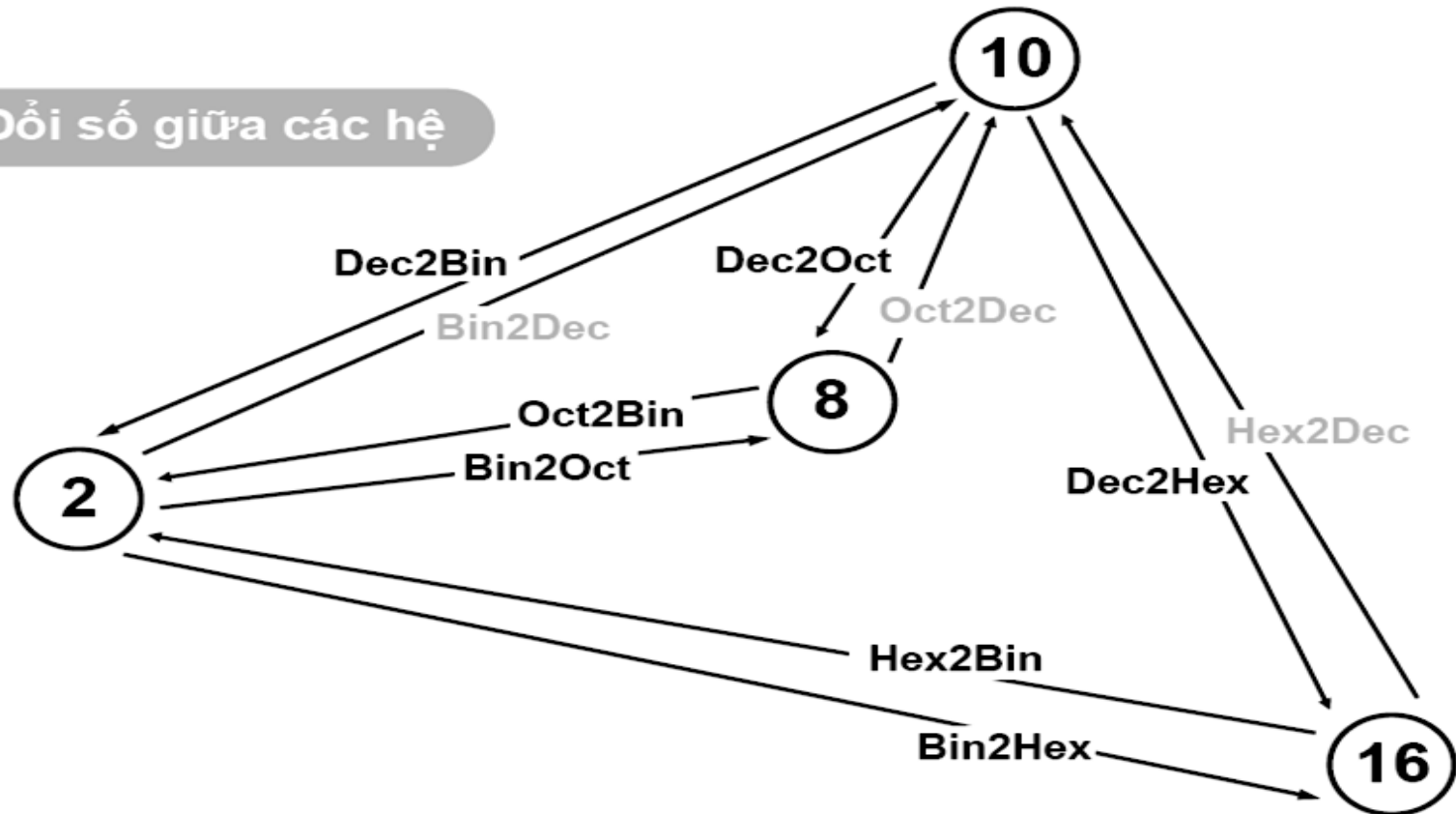
(8+0+2+1)

(8+4+2+0)

Mã 8421

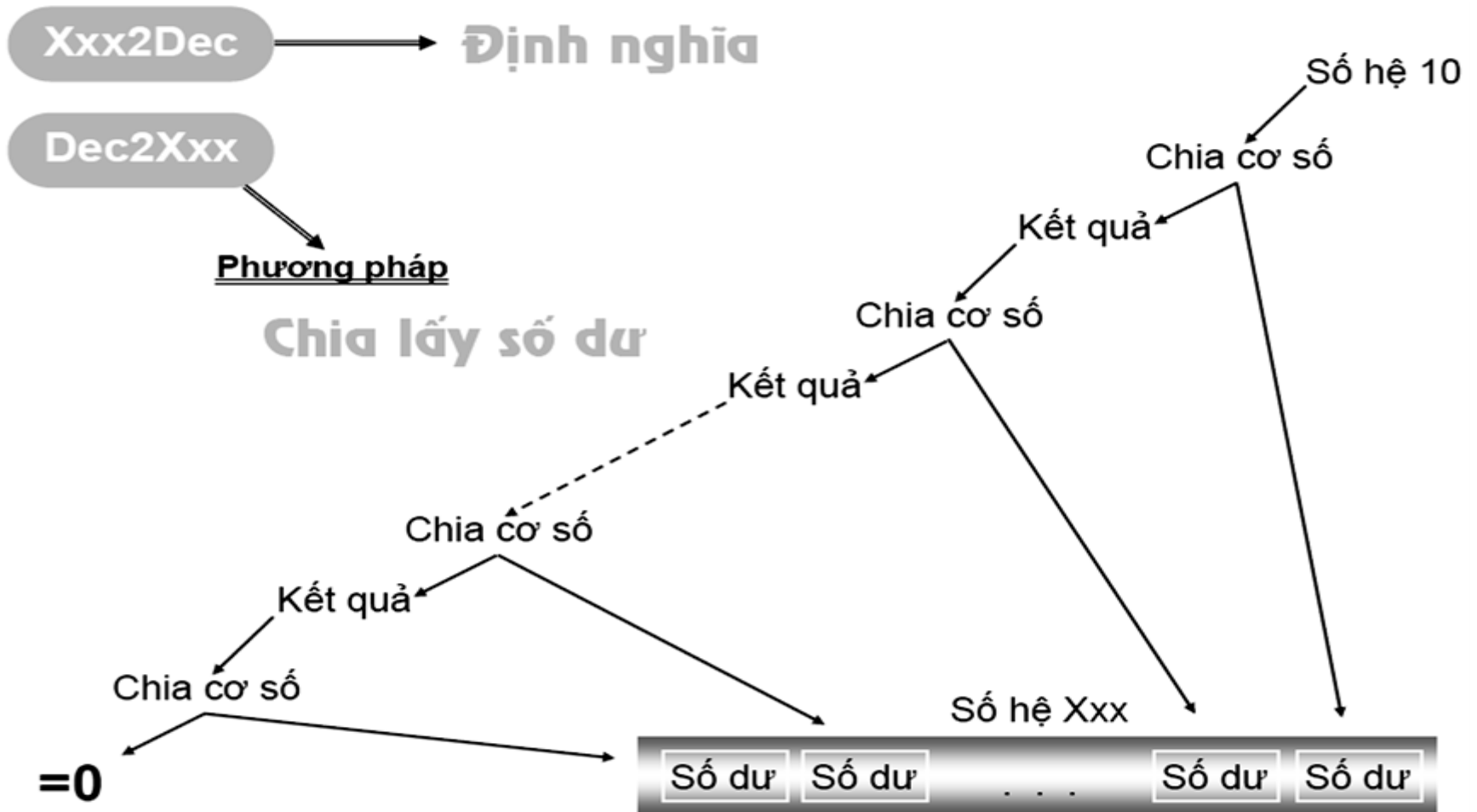
# 4. Chuyển đổi giữa các hệ đếm

Đổi số giữa các hệ





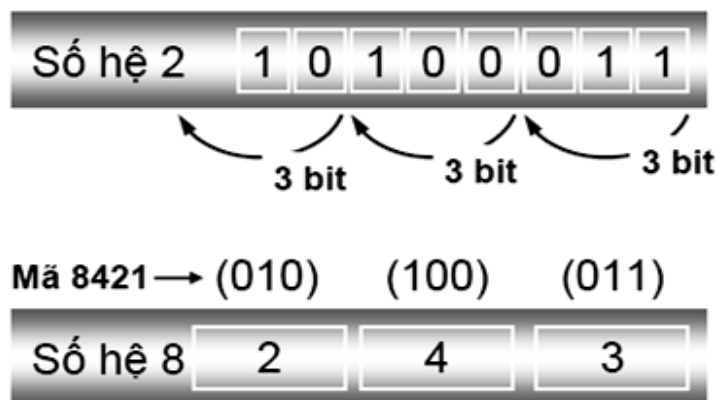
# Phương pháp chuyển đổi



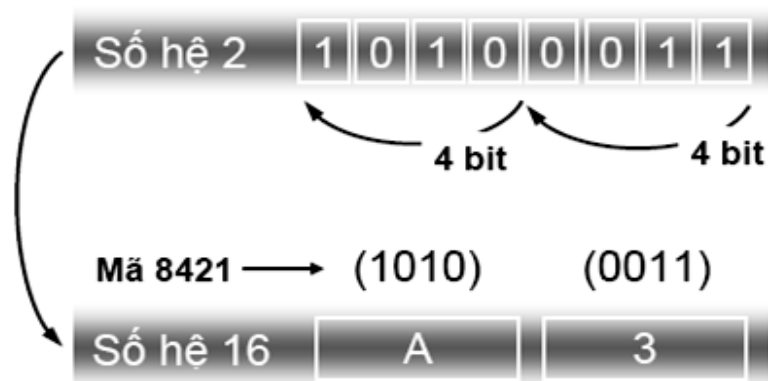
# 4. Chuyển đổi giữa các hệ đếm

Bin2Oct

Ghép nhóm + bảng thuộc lòng



Bin2Hex



## ■ Ví dụ:

- $1029 \rightarrow X_2, Y_8, Z_{16};$
- $14AC_{16} \rightarrow X_2, Y_8, Z_{10}$

## ■ Giải

Chuyển từ 10  $\rightarrow X_2$ ;

$$1029 = 1024 + 4 + 1 = 2^{10} + 2^2 + 2^0$$

$$= 0100000101_2; \rightarrow Y_8 = 2005_8$$

$$= 0100000101 = Z_{16} = 405_{16};$$

$$14AC_{16} \rightarrow X_2 = 0001010010101100_2$$

$$\begin{aligned} 2^{12} + 2^{10} + 2^7 + 2^5 + 2^3 + 2^2 &= \\ 4096 + 1024 + 128 + 32 + 8 + 4 &= 5120 + 172 = 5292 \end{aligned}$$

# Biểu diễn thông tin trong hệ nhị phân

- BIT (Binary digit) :

0 1

- BYTE = tổ hợp 8 bit :

01001101 11111111

- WORD = tổ hợp nhiều bit :

10110 1011100101

- 1 KiloByte (KB) = 1024 byte =  $2^{10}$  byte
- 1 MegaByte (MB) = 1024 KB =  $2^{20}$  byte
- 1 GigaByte (GB) = 1024 MB =  $2^{30}$  byte

## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính

### 1. Nguyên tắc chung về mã hóa dữ liệu

- Mọi dữ liệu đưa vào máy tính đều phải được mã hóa thành số nhị phân
- Các loại dữ liệu
  - Dữ liệu nhân tạo: do con người qui ước
  - Dữ liệu tự nhiên: tồn tại khách quan với con người

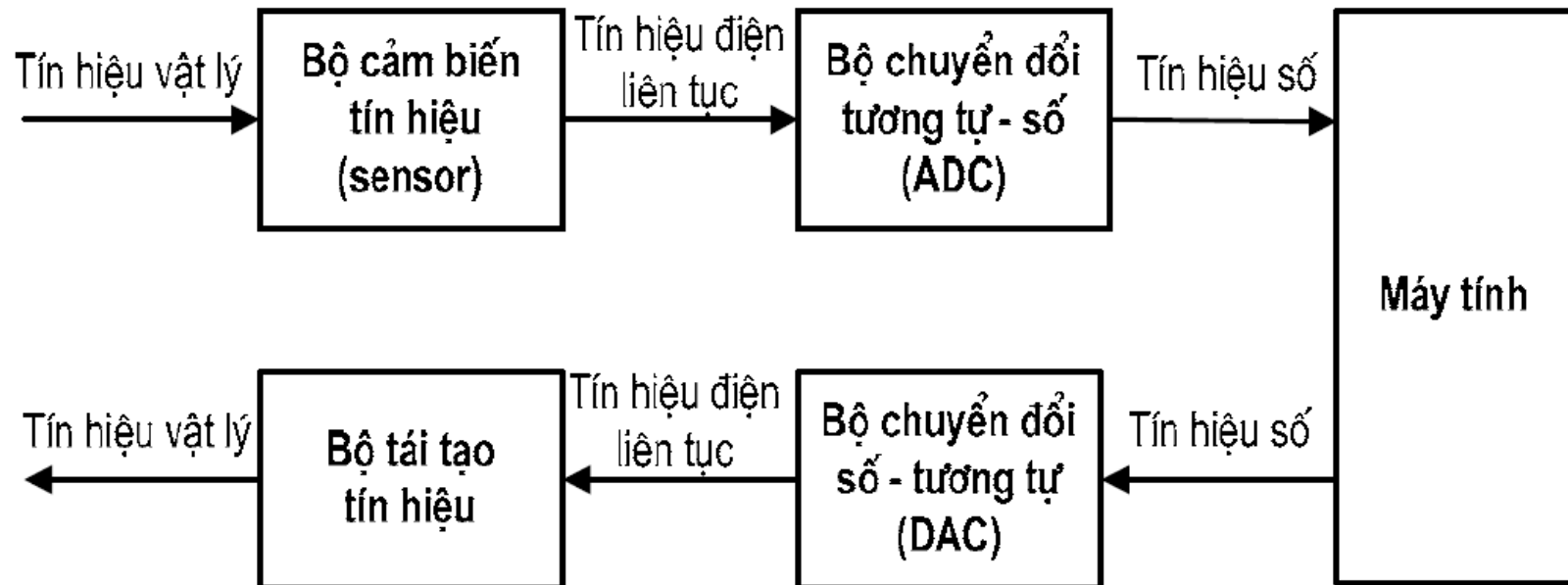
## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

### ■ Mã hoá dữ liệu nhân tạo

- Mã hóa theo các chuẩn quy ước
- Dữ liệu số:
  - Số nguyên: mã hóa theo một số chuẩn
  - Số thực: mã hóa bằng số dấu phẩy động
- Dữ liệu ký tự: mã hóa theo bộ mã ký tự

## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

### ■ Mã hóa và tái tạo tín hiệu vật lý



Các dữ liệu vật lý thông dụng

- Âm thanh
- Hình ảnh

## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

### ■ Độ dài từ dữ liệu

- Độ dài từ dữ liệu là số bit được sử dụng để mã hóa loại dữ liệu tương ứng
- Thường là bội của 8-bit
- VD: 8, 16, 32, 64 bit



## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

### 2. Thứ tự lưu trữ các byte trong bộ nhớ chính

- Bộ nhớ chính thường tổ chức theo byte
- Hai cách lưu trữ dữ liệu nhiều byte:
  - Đầu nhỏ (*Little-endian*): Byte có ý nghĩa thấp được lưu trữ ở ngăn nhớ có địa chỉ nhỏ, byte có ý nghĩa cao được lưu trữ ở ngăn nhớ có địa chỉ lớn.
  - Đầu to (*Big-endian*): Byte có ý nghĩa cao được lưu trữ ở ngăn nhớ có địa chỉ nhỏ, byte có ý nghĩa thấp được lưu trữ ở ngăn nhớ có địa chỉ lớn.

## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

### ■ Ví dụ lưu trữ dữ liệu 32-bit

0001 1010 0010 1011 0011 1100 0100 1101

1A	2B	3C	4D
----	----	----	----

4D	300
3C	301
2B	302
1A	303


little-endian

1A	300
2B	301
3C	302
4D	304

big-endian

## 2.2. Mã hóa và lưu trữ dữ liệu trong máy tính (tiếp)

- Lưu trữ của các bộ xử lý điển hình
  - Intel 80x86 và các Pentium: little-endian
  - Motorola 680x0, SunSPARC: big-endian
  - Power PC, Itanium: big-endian



## 2.3. Biểu diễn số nguyên

- Số nguyên không dấu (Unsigned Integer)
- Số nguyên có dấu (Signed Integer)

# 1. Biểu diễn số nguyên không dấu

- Nguyên tắc tổng quát: Dùng  $n$  bit biểu diễn số nguyên không dấu  $A$ :

$$a_{n-1}a_{n-2} \cdots a_2a_1a_0$$

- Giá trị của  $A$  được tính như sau:

$$A = \sum_{i=0}^{n-1} a_i 2^i$$

- Dải biểu diễn của  $A$ : từ 0 đến  $2^n - 1$

# Các ví dụ

- Ví dụ 1. Biểu diễn các số nguyên không dấu sau đây bằng 8-bit:

$$A = 41 ; B = 150$$

Giải:

- $A = 41 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$
- $41 = 0010\ 1001$
- $B = 150 = 128 + 16 + 4 + 2 = 2^7 + 2^4 + 2^2 + 2^1$   
 $150 = 1001\ 0110$

# Các ví dụ (tiếp)

- Ví dụ 2. Cho các số nguyên không dấu M, N được biểu diễn bằng 8-bit như sau:

- $M = 0001\ 0010$

- $N = 1011\ 1001$

Xác định giá trị của chúng ?

Giải:

- $M = 0001\ 0010 = 2^4 + 2^1 = 16 + 2 = 18$

- $N = 1011\ 1001 = 2^7 + 2^5 + 2^4 + 2^3 + 2^0$   
 $= 128 + 32 + 16 + 8 + 1 = 185$

# Với $n = 8$ bit

- Biểu diễn được các giá trị từ 0 đến 255

$$0000\ 0000 = 0$$

$$0000\ 0001 = 1$$

$$0000\ 0010 = 2$$

$$0000\ 0011 = 3$$

...

$$1111\ 1111 = 255$$

Chú ý:

$$1111\ 1111$$

$$+ \underline{0000\ 0001}$$

$$1\ 0000\ 0000$$

Vậy:  $255 + 1 = 0$  ?

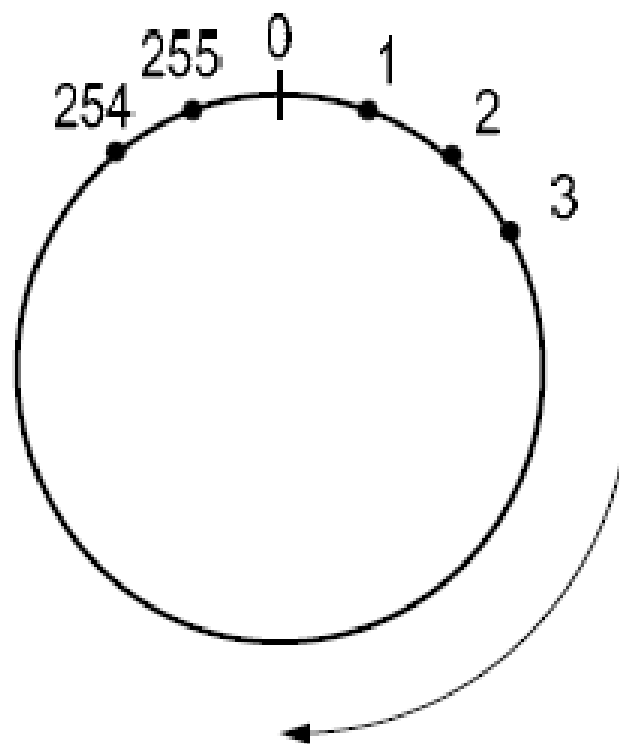
→ do tràn nhớ ra ngoài



# Trục số học với $n = 8$ bit



Trục số học máy tính:



# Với $n = 16$ bit, 32 bit, 64 bit

- $n = 16$  bit: dải biểu diễn từ 0 đến 65535 ( $2^{16} - 1$ )
  - 0000 0000 0000 0000 = 0
  - ...
  - 0000 0000 1111 1111 = 255
  - 0000 0001 0000 0000 = 256
  - ...
  - 1111 1111 1111 1111 = 65535
- $n = 32$  bit: dải biểu diễn từ 0 đến  $2^{32} - 1$
- $n = 64$  bit: dải biểu diễn từ 0 đến  $2^{64} - 1$

## 2. Biểu diễn số nguyên có dấu

a. Số bù chín và Số bù mười

- Cho một số thập phân  $A$  được biểu diễn bằng  $n$  chữ số thập phân, ta có:
  - Số bù chín của  $A = (10^n - 1) - A$
  - Số bù mười của  $A = 10^n - A$
- Số bù mười của  $A = (\text{Số bù chín của } A) + 1$

# Số bù chín và Số bù mười (tiếp)

■ Ví dụ: với  $n=4$ , cho  $A = 3265$

■ Số bù chín của  $A$ :

$$\begin{array}{r} 9999 \quad (10^4-1) \\ - \underline{3265} \quad (A) \\ 6734 \end{array}$$

■ Số bù mười của  $A$ :

$$\begin{array}{r} 10000 \quad (10^4) \\ - \underline{3265} \quad (A) \\ 6735 \end{array}$$

## b. Số bù một và Số bù hai

- Định nghĩa: Cho một số nhị phân  $A$  được biểu diễn bằng  $n$  bit, ta có:
- Số bù một của  $A = (2^n - 1) - A$
- Số bù hai của  $A = 2^n - A$
- Số bù hai của  $A = (\text{Số bù một của } A) + 1$

## b. Số bù một và Số bù hai (tiếp)

- Ví dụ: với  $n = 8$  bit, cho  $A = 0010\ 0101$

- Số bù một của  $A$  được tính như sau:

$$\begin{array}{r} 1111\ 1111 \quad (2^8-1) \\ - 0010\ 0101 \quad (A) \\ \hline 1101\ 1010 \end{array}$$

→ đảo các bit của  $A$

- Số bù hai của  $A$  được tính như sau:

$$\begin{array}{r} 1\ 0000\ 0000 \quad (2^8) \\ - 0010\ 0101 \quad (A) \\ \hline 1101\ 1011 \end{array}$$

→ thực hiện khó khăn

# Quy tắc tìm Số bù một và Số bù hai

- Số bù một của A = đảo giá trị các bit của A
- (Số bù hai của A) = (Số bù một của A) + 1
- Ví dụ:

- Cho A = 0010 0101
- Số bù một = 1101 1010
- Số bù hai = 
$$\begin{array}{r} 1101 \ 1010 \\ + \quad \quad 1 \\ \hline 1101 \ 1011 \end{array}$$

- Nhận xét:

$$\begin{array}{rcl} A & = & 0010 \ 0101 \\ \text{Số bù hai} & = & + \ 1101 \ 1011 \\ & & \underline{\hspace{1cm}} \\ & & 1 \ 0000 \ 0000 = 0 \end{array}$$

(bỏ qua bit nhớ ra ngoài)

→ Số bù hai của A = -A

## c. Biểu diễn số nguyên có dấu bằng mã bù hai

- Nguyên tắc tổng quát: Dùng  $n$  bit biểu diễn số nguyên có dấu  $A$ :

$$a_{n-1}a_{n-2} \dots a_2a_1a_0$$

- Với  $A$  là số dương: bit  $a_{n-1} = 0$ , các bit còn lại biểu diễn độ lớn như số không dấu
- Với  $A$  là số âm: được biểu diễn bằng số bù hai của số dương tương ứng, vì vậy bit  $a_{n-1} = 1$



# Biểu diễn số dương

- Dạng tổng quát của số dương A:

$$0a_{n-2} \dots a_2 a_1 a_0$$

- Giá trị của số dương A:

$$A = \sum_{i=0}^{n-2} a_i 2^i$$

- Dải biểu diễn cho số dương: 0 đến  $2^{n-1}-1$

# Biểu diễn số âm

- Dạng tổng quát của số âm A:

$$1a_{n-2} \dots a_2 a_1 a_0$$

- Giá trị của số âm A:

$$A = -2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

- Dải biểu diễn cho số âm: -1 đến  $-2^{n-1}$

# Biểu diễn tổng quát cho số nguyên có dấu

- Dạng tổng quát của số nguyên A:

$$a_{n-1}a_{n-2} \dots a_2a_1a_0$$

- Giá trị của A được xác định như sau:

$$A = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

- Dải biểu diễn: từ  $-(2^{n-1})$  đến  $+(2^{n-1}-1)$

# Các ví dụ

- Ví dụ 1. Biểu diễn các số nguyên có dấu sau đây bằng 8-bit:

$$A = +58 ; B = -80$$

Giải:

$$A = +58 = 0011\ 1010$$

$$B = -80$$

$$\text{Ta có: } +80 = 0101\ 0000$$

$$\text{Số bù một} = 1010\ 1111$$

$$+ \underline{\quad\quad\quad 1}$$

$$\text{Số bù hai} = 1011\ 0000$$

$$\text{Vậy: } B = -80 = 1011\ 0000$$

# Các ví dụ (tiếp)

- Ví dụ 2. Hãy xác định giá trị của các số nguyên có dấu được biểu diễn dưới đây:

- $P = 0110\ 0010$

- $Q = 1101\ 1011$

Giải:

- $P = 0110\ 0010 = 64 + 32 + 2 = +98$

- $Q = 1101\ 1011 = -128 + 64 + 16 + 8 + 2 + 1 = -37$

# Với $n = 8$ bit

- Biểu diễn được các giá trị từ -128 đến +127

$$0000\ 0000 = 0$$

$$0000\ 0001 = +1$$

$$0000\ 0010 = +2$$

$$0000\ 0011 = +3$$

...

$$0111\ 1111 = +127$$

$$1000\ 0000 = -128$$

$$1000\ 0001 = -127$$

...

$$1111\ 1110 = -2$$

$$1111\ 1111 = -1$$

Chú ý:

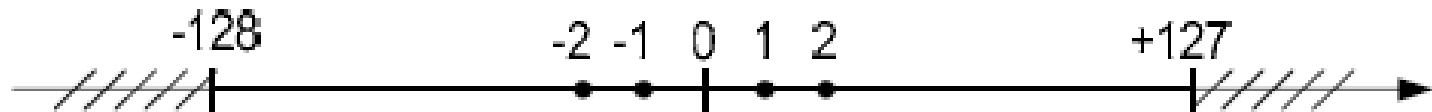
$$+127 + 1 = -128$$

$$-128 - 1 = +127$$

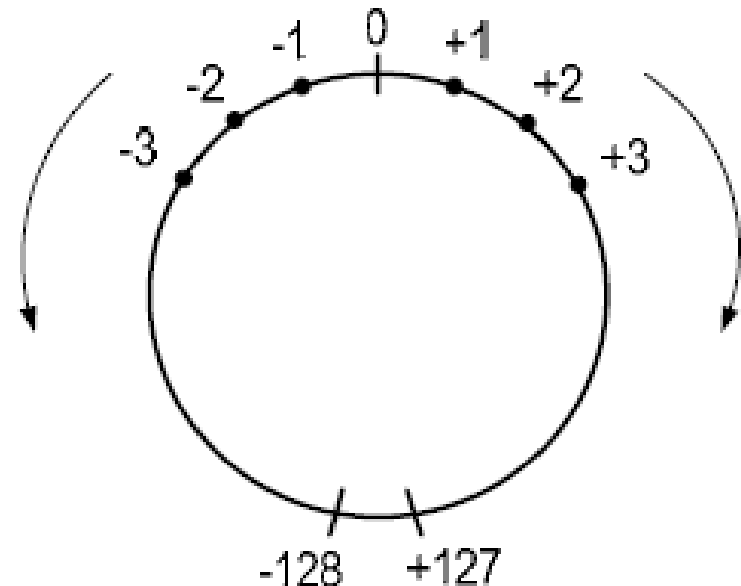
→ do tràn xảy ra

# Trục số học số nguyên có dấu với $n = 8$ bit

- Trục số học:



- Trục số học máy tính:



# Với $n = 16 \text{ bit}, 32 \text{ bit}, 64 \text{ bit}$

- Với  $n=16\text{bit}$ : biểu diễn từ  $-32768$  đến  $+32767$

- 0000 0000 0000 0000 = 0
- 0000 0000 0000 0001 = +1
- ...
- 0111 1111 1111 1111 = +32767
- 1000 0000 0000 0000 = -32768
- ...
- 1111 1111 1111 1111 = -1

- Với  $n=32\text{bit}$ : biểu diễn từ  $-2^{31}$  đến  $2^{31}-1$

- Với  $n=64\text{bit}$ : biểu diễn từ  $-2^{63}$  đến  $2^{63}-1$



# Chuyển đổi từ byte thành word

- Đối với số dương:

+19 = 0001 0011 (8bit)

+19 = 0000 0000 0001 0011 (16bit)

→ thêm 8 bit 0 bên trái

- Đối với số âm:

- 19 = 1110 1101 (8bit)

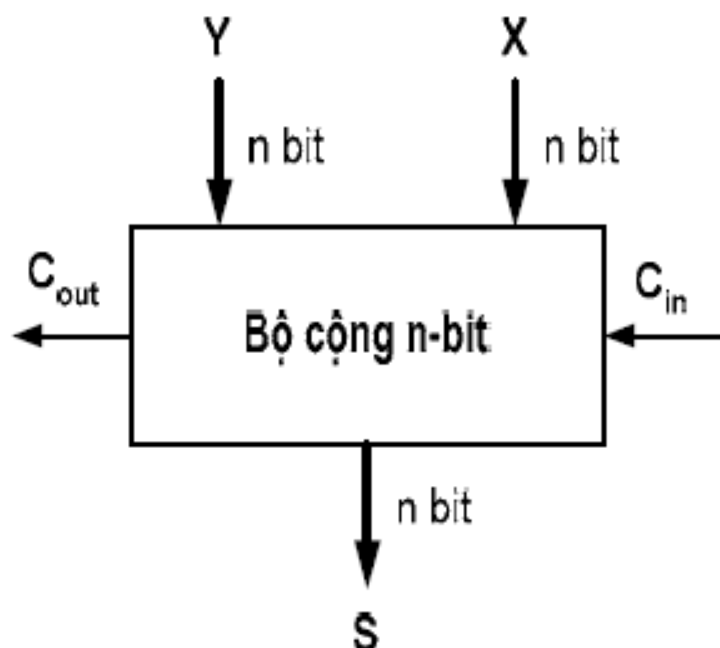
- 19 = 1111 1111 1110 1101 (16bit)

→ thêm 8 bit 1 bên trái

## 2.4. Thực hiện các phép toán số học với số nguyên

### 1. Phép cộng số nguyên không dấu

- Bộ cộng n-bit



# Nguyên tắc cộng số nguyên không dấu

- Khi cộng hai số nguyên không dấu n-bit, kết quả nhận được là n-bit:
  - Nếu  $C_{out}=0 \rightarrow$  nhận được kết quả đúng.
  - Nếu  $C_{out}=1 \rightarrow$  nhận được kết quả sai, do tràn nhớ ra ngoài (*Carry Out*).
  - Tràn nhớ ra ngoài khi:  $tổng > (2^n - 1)$

# Ví dụ cộng số nguyên không dấu

$$\begin{array}{rcl} \blacksquare & 57 & = 0011\ 1001 \\ & +\ 34 & = +\ 0010\ 0010 \\ & \hline & 91 & 0101\ 1011 = 64+16+8+2+1=91 \rightarrow \text{đúng} \end{array}$$

$$\begin{array}{rcl} \blacksquare & 209 & = 1101\ 0001 \\ & +\ 73 & = +\ 0100\ 1001 \\ & \hline & 282 & \textcolor{red}{1}\ 0001\ 1010 \\ & & 0001\ 1010 = 16+8+2=26 \rightarrow \text{sai} \\ & & \rightarrow \text{có tràn nhớ ra ngoài (C}_{\text{out}}\text{=1)} \end{array}$$

Để có kết quả đúng ta thực hiện cộng theo 16-bit:

$$\begin{array}{rcl} 209 & = & 0000\ 0000\ 1101\ 0001 \\ +\ 73 & = + & 0000\ 0000\ 0100\ 1001 \\ & \hline & 0000\ 0001\ 0001\ 1010 & = 256+16+8+2 = 282 \end{array}$$

## 2. Phép đảo dấu

- Ta có:

$$\begin{array}{rcll} + 37 & = & 0010\ 0101 & \\ \text{bù một} & = & 1101\ 1010 & \\ & + & \underline{\phantom{1101\ 1010}1} & \\ \text{bù hai} & = & 1101\ 1011 & = -37 \end{array}$$

- Lấy bù hai của số âm:

$$\begin{array}{rcll} - 37 & = & 1101\ 1011 & \\ \text{bù một} & = & 0010\ 0100 & \\ & + & \underline{\phantom{0010\ 0100}1} & \\ \text{bù hai} & = & 0010\ 0101 & = +37 \end{array}$$

- Kết luận: Phép đảo dấu trong máy tính thực chất là lấy bù hai

### 3. Cộng số nguyên có dấu

Khi cộng hai số nguyên có dấu n-bit, kết quả nhận được là n-bit và **không cần quan tâm đến bit  $C_{out}$**

- Cộng hai số khác dấu: **kết quả** luôn luôn **đúng**.
- Cộng hai số cùng dấu:
  - nếu dấu kết quả cùng dấu với các số hạng thì **kết quả là đúng**.
  - nếu kết quả có dấu ngược lại, khi đó có **tràn** xảy ra (**Overflow**) và **kết quả bị sai**.
- Tràn xảy ra khi tổng nằm ngoài dải biểu diễn:  
$$[-(2^{n-1}), +(2^{n-1}-1)]$$

# Ví dụ cộng số nguyên có dấu không tràn

$$\begin{array}{rcl} \blacksquare & (+70) & = 0100\ 0110 \\ & + \underline{(+42)} & = \underline{0010\ 1010} \\ & + 112 & 0111\ 0000 = +112 \end{array}$$

$$\begin{array}{rcl} \blacksquare & (+97) & = 0110\ 0001 \\ & + \underline{(-52)} & = \underline{1100\ 1100} \quad (+52=0011\ 0100) \\ & + 45 & 1\ 0010\ 1101 = +45 \end{array}$$

$$\begin{array}{rcl} \blacksquare & (-90) & = 1010\ 0110 \quad (+90=0101\ 1010) \\ & + \underline{(+36)} & = \underline{0010\ 0100} \\ & - 54 & 1100\ 1010 = -54 \end{array}$$

$$\begin{array}{rcl} \blacksquare & (-74) & = 1011\ 0110 \quad (+74=0100\ 1010) \\ & + \underline{(-30)} & = \underline{1110\ 0010} \quad (+30=0001\ 1110) \\ & -104 & 1\ 1001\ 1000 = -104 \end{array}$$

# Ví dụ cộng số nguyên có dấu bị tràn

$$\begin{array}{rcl} \blacksquare \quad (+75) & = & 0100\ 1011 \\ +(+82) & = & \underline{0101\ 0010} \\ +157 & & 1001\ 1101 \\ & & = -128+16+8+4+1 = -99 \rightarrow \text{sai} \end{array}$$

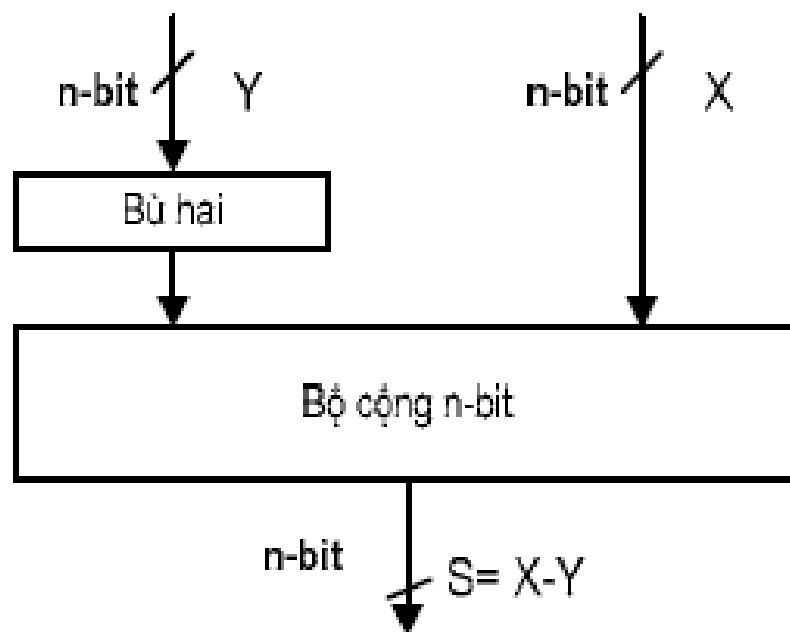
$$\begin{array}{rcl} \blacksquare \quad (-104) & = & 1001\ 1000 \quad (+104=0110\ 1000) \\ +(-43) & = & \underline{1101\ 0101} \quad (+43=0010\ 1011) \\ -147 & & 1\ 0110\ 1101 \\ & & = 64+32+8+4+1 = +109 \rightarrow \text{sai} \end{array}$$

- Cả hai ví dụ đều tràn vì tổng nằm ngoài dải biểu diễn  $[-128, +127]$



## 4. Nguyên tắc thực hiện phép trừ

- Phép trừ hai số nguyên:  $X - Y = X + (-Y)$
- Nguyên tắc: Lấy bù hai của  $Y$  để được  $-Y$ , rồi cộng với  $X$



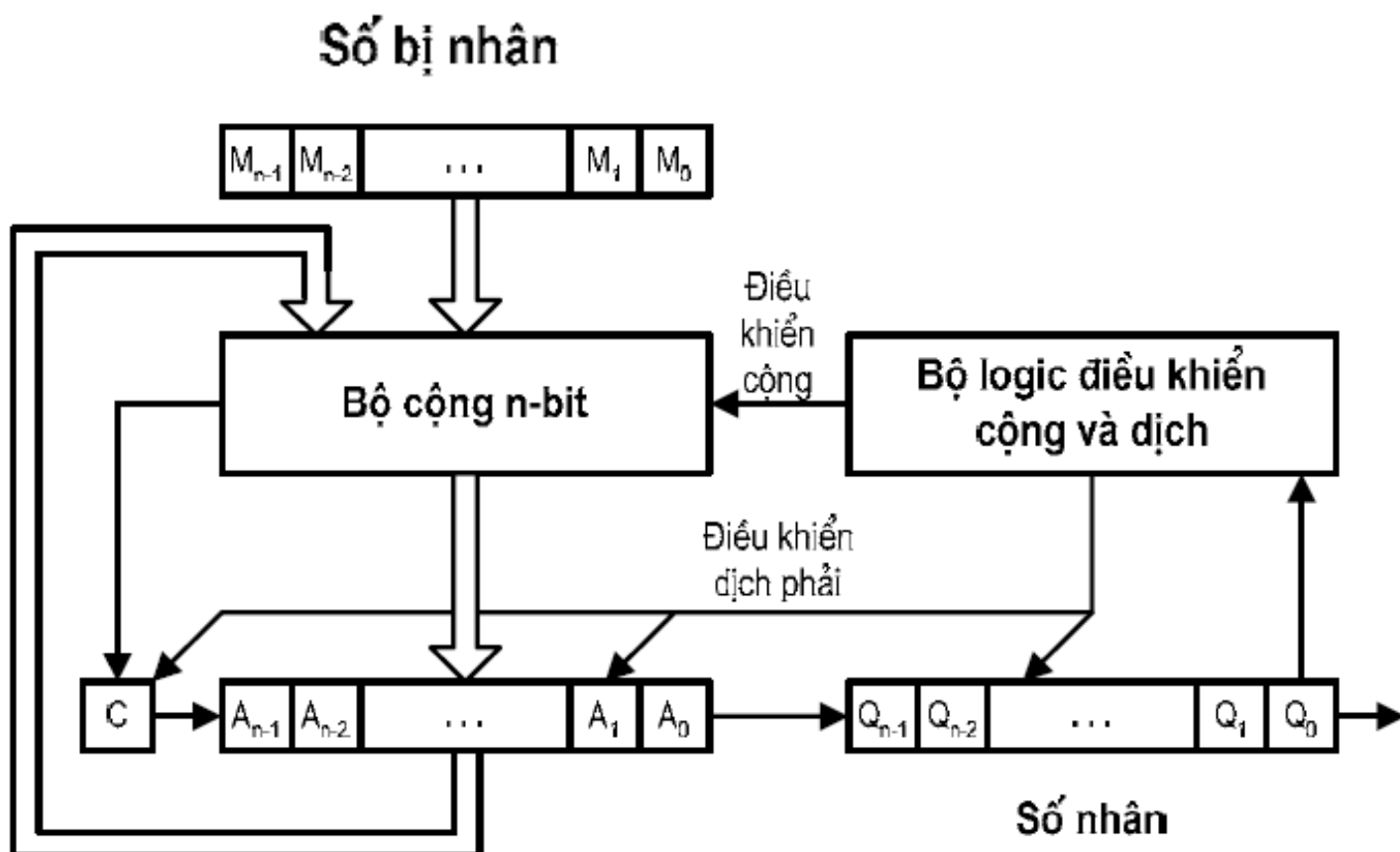
## 5. Nhân số nguyên không dấu

1011	Số bị nhân (11)
x <u>1101</u>	Số nhân (13)
1011	} Các tích riêng phần
0000	
1011	
1011	
<u>1011</u>	
10001111	Tích (143)

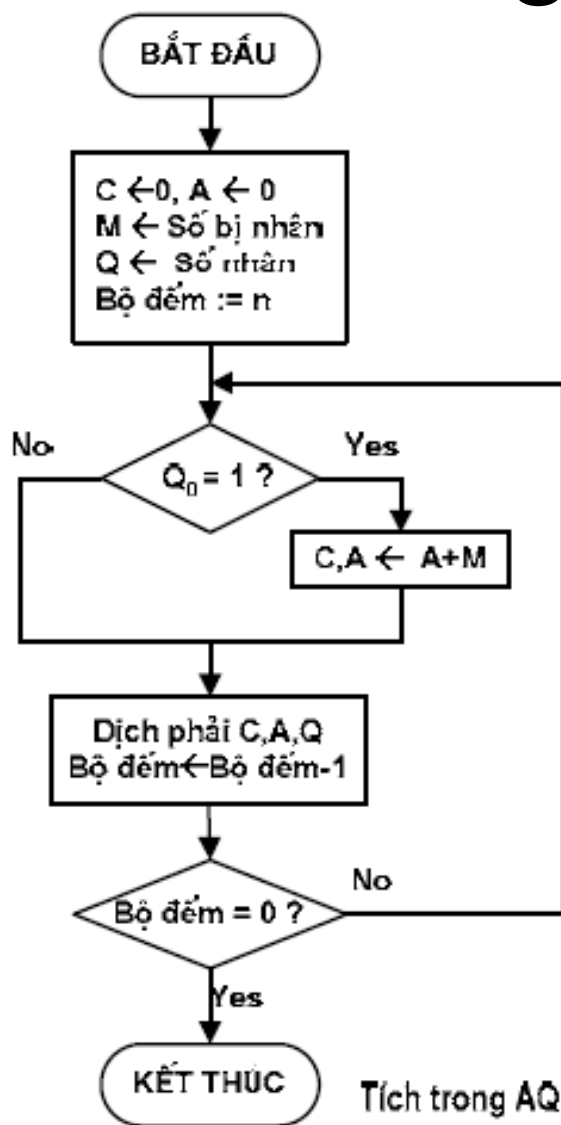
## 5. Nhân số nguyên không dấu (tiếp)

- Các tích riêng phần được xác định như sau:
  - Nếu bit của số nhân bằng 0 → tích riêng phần bằng 0.
  - Nếu bit của số nhân bằng 1 → tích riêng phần bằng số bị nhân.
  - Tích riêng phần tiếp theo được dịch trái một bit so với tích riêng phần trước đó.
- Tích bằng tổng các tích riêng phần
- Nhân hai số nguyên  $n$ -bit, tích có độ dài  $2n$  bit (không bao giờ tràn).

# Bộ nhân số nguyên không dấu



# Lưu đồ nhân số nguyên không dấu



# Ví dụ nhân số nguyên không dấu

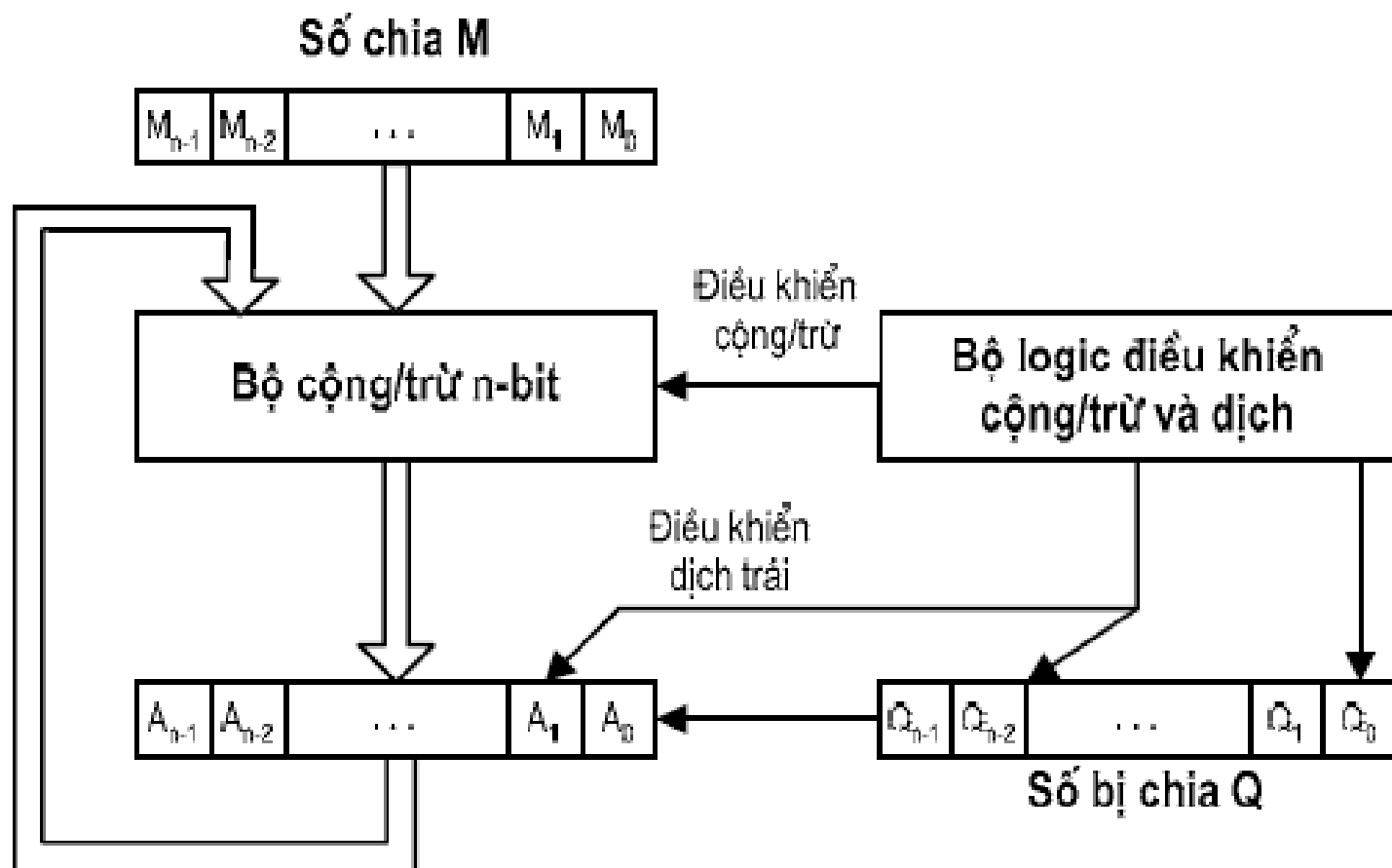
- Số bị nhân M = 1011 (11)
- Số nhân Q = 1101 (13)
- Tích = 1000 1111 (143)

- |   |        |      |                      |
|---|--------|------|----------------------|
| C | A      | Q    |                      |
| 0 | 0000   | 1101 | Các giá trị khởi đầu |
|   | + 1011 |      |                      |
- |   |      |      |           |
|---|------|------|-----------|
| 0 | 1011 | 1101 | A ← A + M |
| 0 | 0101 | 1110 | Dịch phải |
- |   |        |      |           |
|---|--------|------|-----------|
| 0 | 0010   | 1111 | Dịch phải |
|   | + 1011 |      |           |
- |   |        |      |           |
|---|--------|------|-----------|
| 0 | 1101   | 1111 | A ← A + M |
| 0 | 0110   | 1111 | Dịch phải |
|   | + 1011 |      |           |
- |   |      |      |           |
|---|------|------|-----------|
| 1 | 0001 | 1111 | A ← A + M |
| 0 | 1000 | 1111 | Dịch phải |

## 7. Chia số nguyên không dấu

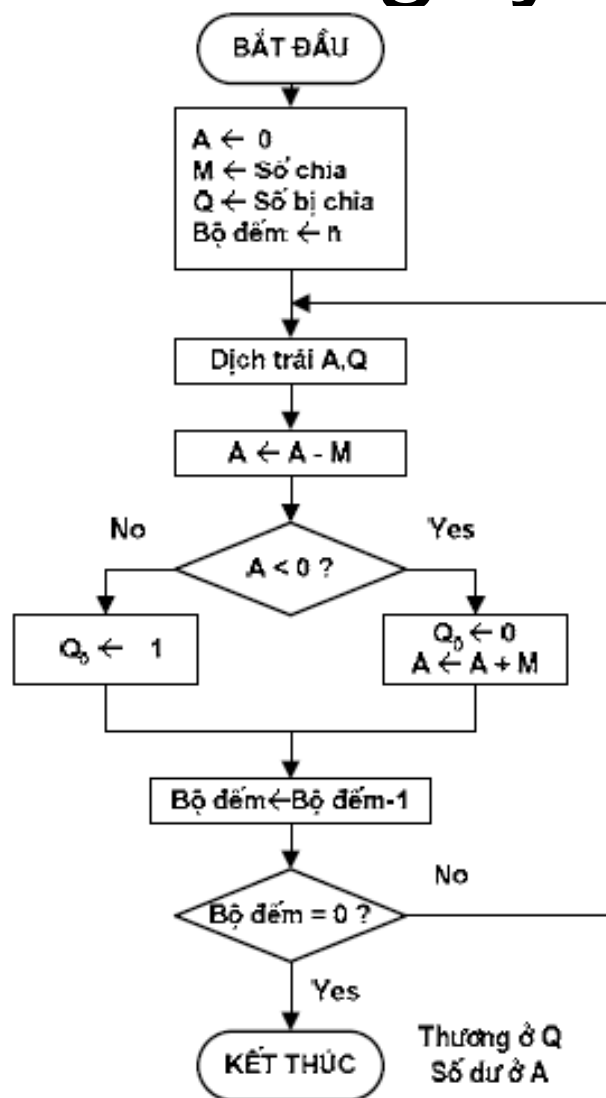
Số bị chia	$\begin{array}{r} 10010011 \\ \underline{1011} \\ 001110 \\ \underline{1011} \\ 001111 \\ \underline{1011} \\ 100 \end{array}$	$\begin{array}{r} 1011 \\ \hline 00001101 \end{array}$	Số chia Thương
			Phần dư

# Bộ chia số nguyên không dấu





# Lưu đồ chia số nguyên không dấu



## 8. Chia số nguyên có dấu

- Bước 1. Chuyển đổi số bị chia và số chia về thành số dương tương ứng.
- Bước 2. Sử dụng thuật giải chia số nguyên không dấu để chia hai số dương, kết quả nhận được là thương Q và phần dư R đều là dương
- Bước 3. Hiệu chỉnh dấu của kết quả như sau:  
(Lưu ý: phép đảo dấu thực chất là thực hiện phép lấy bù hai)

Số bị chia	Số chia	Thương	Số dư
dương	dương	giữ nguyên	giữ nguyên
dương	âm	đảo dấu	giữ nguyên
âm	dương	đảo dấu	đảo dấu
âm	âm	giữ nguyên	đảo dấu

## 2.5. Số thực dấu phẩy động

### 1. Nguyên tắc chung

- Floating Point Number → biểu diễn cho số thực
- Tổng quát: một số thực  $X$  được biểu diễn theo kiểu số dấu phẩy động như sau:

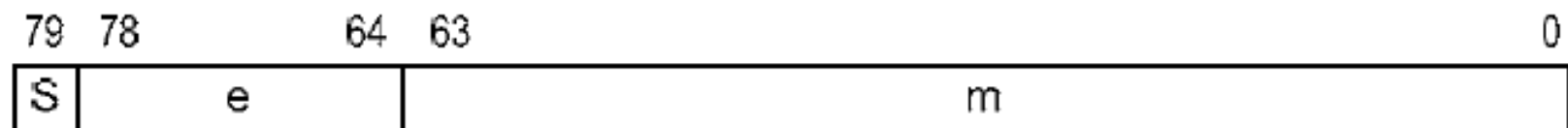
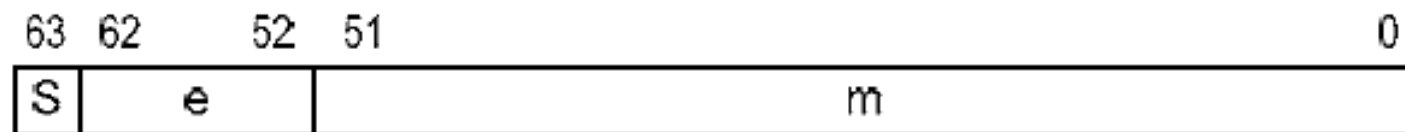
$$X = M * R^E$$

- $M$  là phần định trị (Mantissa),
- $R$  là cơ số (Radix),
- $E$  là phần mũ (Exponent).

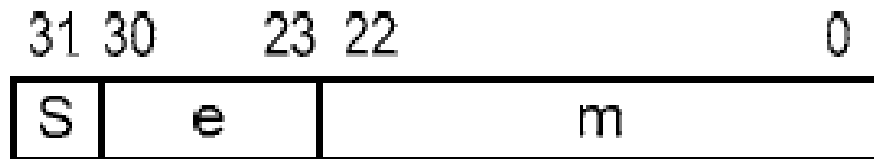
## 2. Chuẩn IEEE754/85

- Cơ số  $R = 2$
- Các dạng:
  - Dạng 32-bit
  - Dạng 64-bit
  - Dạng 80-bit

31	30	23	22	0
S	e	m		



# Dạng 32 bit



- **S** là bit dấu:
  - $S = 0 \rightarrow$  số dương
  - $S = 1 \rightarrow$  số âm
- **e** (8 bit) là mã *excess-127* của phần mũ E:
  - $e = E + 127 \rightarrow E = e - 127$
  - giá trị 127 gọi là độ lệch (bias)
- **m** (23 bit) là phần lẻ của phần định trị M:
  - $M = 1.m$
- Công thức xác định giá trị của số thực:

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

# Ví dụ 1

Xác định giá trị của số thực được biểu diễn bằng 32-bit như sau:

■ 1100 0001 0101 0110 0000 0000 0000 0000

■  $S = 1 \rightarrow$  số âm

■  $e = 1000\ 0010_2 = 130 \rightarrow E = 130 - 127 = 3$

Vậy

$$X = -1.10101100 \cdot 2^3 = -1101.011 = -13.375$$

■ 0011 1111 1000 0000 0000 0000 0000 0000 = ?  
= +1.0

## Ví dụ 2

Biểu diễn số thực  $X = 83.75$  về dạng số dấu phẩy động IEEE754 32-bit

Giải:

- $X = 83.75_{(10)} = 1010011.11_{(2)} = 1.01001111 \times 2^6$

- Ta có:

- $S = 0$  vì đây là số dương

- $E = e - 127 = 6 \rightarrow e = 127 + 6 = 133_{(10)} = 1000\ 0101_{(2)}$

- Vậy:

$$X = 0100\ 0010\ 1010\ 0111\ 1000\ 0000\ 0000\ 0000$$



# Ví dụ 3

Biểu diễn số thực  $X = -0,2$  về dạng số dấu phẩy động IEEE754 32-bit

Giải:

- $X = -0,2_{(10)} = -0.00110011...0011..._{(2)} =$   
 $= -1.100110011...0011... \times 2^{-3}$
- Ta có:
  - $S = 1$  vì đây là số âm
  - $E = e - 127 = -3 \rightarrow e = 127 - 3 = 124_{(10)} = 0111\ 1100_{(2)}$
- Vậy:  
 $X = 1011\ 1110\ 0100\ 1100\ 1100\ 1100\ 1100\ 1100$

# Các quy ước đặc biệt

- Các bit của e bằng 0, các bit của m bằng 0, thì  $X = \pm 0$   
 $x\text{000 0000 0000 0000 0000 0000 0000 0000} \rightarrow X = \pm 0$
- Các bit của e bằng 1, các bit của m bằng 0, thì  $X = \pm \infty$   
 $x\text{111 1111 1000 0000 0000 0000 0000 0000} \rightarrow X = \pm \infty$
- Các bit của e bằng 1, còn m có ít nhất một bit bằng 1, thì nó không biểu diễn cho số nào cả (NaN - not a number)

# Dải giá trị biểu diễn

- $2^{-127}$  đến  $2^{+127}$
- $10^{-38}$  đến  $10^{+38}$



# Dạng 64-bit

- $S$  là bit dấu
- $e$  (11 bit): mã *excess-1023* của phần mũ  $E \rightarrow E = e - 1023$
- $m$  (52 bit): phần lẻ của phần định trị  $M$
- Giá trị số thực:

$$X = (-1)^S \cdot 1.m \cdot 2^{e-1023}$$

- Dải giá trị biểu diễn:  $10^{-308}$  đến  $10^{+308}$

# Dạng 80-bit

- $S$  là bit dấu
- $e$  (15 bit): mã *excess-16383* của phần mũ  $E \rightarrow E = e - 16383$
- $m$  (64 bit): phần lẻ của phần định trị  $M$
- Giá trị số thực:

$$X = (-1)^S * 1.m * 2^{e-16383}$$

- Dải giá trị biểu diễn:  $10^{-4932}$  đến  $10^{+4932}$

### 3. Thực hiện phép toán số dấu phẩy động

- $X1 = M1 * R^{E1}$
- $X2 = M2 * R^{E2}$
- Ta có
  - $X1 * X2 = (M1 * M2) * R^{E1+E2}$
  - $X1 / X2 = (M1 / M2) * R^{E1-E2}$
  - $X1 \pm X2 = (M1 * R^{E1-E2} \pm M2) * R^{E2}$  , với  $E2 \geq E1$

# Các khả năng tràn số

- Tràn trên số mũ (Exponent Overflow): mũ dương vượt ra khỏi giá trị cực đại của số mũ dương có thể. ( $\rightarrow \infty$ )
- Tràn dưới số mũ (Exponent Underflow): mũ âm vượt ra khỏi giá trị cực đại của số mũ âm có thể ( $\rightarrow 0$ ).
- Tràn trên phần định trị (Mantissa Overflow): cộng hai phần định trị có cùng dấu, kết quả bị nhớ ra ngoài bit cao nhất.
- Tràn dưới phần định trị (Mantissa Underflow): Khi hiệu chỉnh phần định trị, các số bị mất ở bên phải phần định trị.

# Phép cộng và phép trừ

- Kiểm tra các số hạng có bằng 0 hay không
- Hiệu chỉnh phần định trị
- Cộng hoặc trừ phần định trị
- Chuẩn hoá kết quả



## 2.6. Biểu diễn ký tự

- Bộ mã ASCII (American Standard Code for Information Interchange)
- Bộ mã Unicode

# 1. Bộ mã ASCII

- Do ANSI (American National Standard Institute) thiết kế
- Bộ mã 8-bit  $\rightarrow$  có thể mã hóa được  $2^8$  ký tự, có mã từ:  $00_{16} \div FF_{16}$ , trong đó:
  - 128 ký tự chuẩn có mã từ  $00_{16} \div 7F_{16}$
  - 128 ký tự mở rộng có mã từ  $80_{16} \div FF_{16}$

# Các ký tự chuẩn

- Các ký tự hiển thị chuẩn:
  - Các chữ cái Latin
  - Các chữ số thập phân
  - các dấu câu: . , : ; ...
  - các dấu phép toán: + - \* / % ...
  - một số ký hiệu thông dụng: &, \$, @, #
  - dấu cách
- Các mã điều khiển
  - Các mã điều khiển định dạng văn bản
  - Các mã điều khiển truyền số liệu
  - Các mã điều khiển phân tách thông tin
  - Các mã điều khiển khác

HEXA	0	1	2	3	4	5	6	7
0	<NUL> 0	<DLE> 16	<space> 32	0 48	@ 64	P 80	` 96	p 112
1	<SOH> 1	<DC1> 17	! 33	1 49	A 65	Q 81	a 97	q 113
2	<STX> 2	<DC2> 18	" 34	2 50	B 66	R 82	b 98	r 114
3	<ETX> 3	<DC3> 19	# 35	3 51	C 67	S 83	c 99	s 115
4	<EOT> 4	<DC4> 20	\$ 36	4 52	D 68	T 84	d 100	t 116
5	<ENQ> 5	<NAK> 21	% 37	5 53	E 69	U 85	e 101	u 117
6	<ACK> 6	<SYN> 22	& 38	6 54	F 70	V 86	f 102	v 118
7	<BEL> 7	<ETB> 23	' 39	7 55	G 71	W 87	g 103	w 119
8	<BS> 8	<CAN> 24	( 40	8 56	H 72	X 88	h 104	x 120
9	<HT> 9	<EM> 25	) 41	9 57	I 73	Y 89	i 105	y 121
A	<LF> 10	<SUB> 26	* 42	: 58	J 74	Z 90	j 106	z 122
B	<VT> 11	<ESC> 27	+ 43	: 59	K 75	[ 91	k 107	{ 123
C	<FF> 12	<FS> 28	, 44	< 60	L 76	\ 92	l 108	 124
D	<CR> 13	<GS> 29	- 45	= 61	M 77	] 93	m 109	} 125
E	<SO> 14	<RS> 30	. 46	> 62	N 78	^ 94	n 110	~ 126
F	<SI> 15	<US> 31	/ 47	? 63	O 79	- 95	o 111	<DEL> 127

# Các ký tự hiển thị chuẩn

- 26 chữ cái hoa 'A' đến 'Z' có mã từ  $41_{(16)}$  đến  $5A_{(16)}$  (65 đến 90):

■ 'A'	→	0100	0001	=	$41_{(16)}$
■ 'B'	→	0100	0010	=	$42_{(16)}$
■ 'C'	→	0100	0011	=	$43_{(16)}$
■ ...					
■ 'Z'	→	0101	1010	=	$5A_{(16)}$

- 26 chữ cái thường 'a' đến 'z' có mã từ  $61_{(16)}$  đến  $7A_{(16)}$  (97 đến 122):

■ 'a'	→	0110	0001	=	$61_{(16)}$
■ 'b'	→	0110	0010	=	$62_{(16)}$
■ 'c'	→	0110	0011	=	$63_{(16)}$
■ ...					
■ 'z'	→	0111	1010	=	$7A_{(16)}$

# Các ký tự hiển thị chuẩn (tiếp)

- 10 chữ số thập phân từ '0' đến '9' có mã từ  $30_{(16)}$  đến  $39_{(16)}$  (48 đến 57):

- '0'  $\rightarrow$  0011 0000 =  $30_{(16)}$

- '1'  $\rightarrow$  0011 0001 =  $31_{(16)}$

- '2'  $\rightarrow$  0011 0010 =  $32_{(16)}$

- . . .

- '9'  $\rightarrow$  0011 1001 =  $39_{(16)}$

# Các ký tự hiển thị chuẩn (tiếp)

- Các ký hiệu khác:
  - các dấu câu: . , : ; ...
  - các dấu phép toán: + - \* / % ...
  - một số ký hiệu thông dụng: &, \$, @, #
  - dấu cách

# Các mã điều khiển: có mã

$00_{16} \div 1F_{16}$  và  $7F_{16}$

Các mã ký tự điều khiển định dạng (điều khiển màn hình, máy in ...)	
BS	Backspace – Lùi lại một vị trí: Ký tự điều khiển con trỏ lùi lại một vị trí.
HT	Horizontal Tab - Tab ngang: Ký tự điều khiển con trỏ dịch tiếp một khoảng đã định trước.
LF	Line Feed – Xuống một dòng: Ký tự điều khiển con trỏ chuyển xuống dòng dưới.
VT	Vertical Tab – Tab đứng: Ký tự điều khiển con trỏ chuyển qua một số dòng đã định trước.
FF	Form Feed - Đẩy sang đầu trang: Ký tự điều khiển con trỏ di chuyển xuống đầu trang tiếp theo.
CR	Carriage Return – Về đầu dòng: Ký tự điều khiển con trỏ di chuyển về đầu dòng hiện hành.



# Các mã điều khiển (tiếp)

Các mã ký tự điều khiển truyền tin	
SOH	Start of Heading - Bắt đầu tiêu đề: Ký tự đánh dấu bắt đầu phần thông tin tiêu đề.
STX	Start of Text - Bắt đầu văn bản: Ký tự đánh dấu bắt đầu khối dữ liệu văn bản và cũng chính là để kết thúc phần thông tin tiêu đề.
ETX	End of Text – Kết thúc văn bản: Ký tự đánh dấu kết thúc khối dữ liệu văn bản đã được bắt đầu bằng STX.
EOT	End of Transmission - Kết thúc truyền: Chi ra cho bên thu biết kết thúc truyền.
ENQ	Enquiry – Hỏi: Tín hiệu yêu cầu đáp ứng từ một máy ở xa.
ACK	Acknowledge - Báo nhận: Ký tự được phát ra từ phía thu báo cho phía phát biết rằng dữ liệu đã được nhận thành công.
NAK	Negative Acknowledge - Báo phủ nhận: Ký tự được phát ra từ phía thu báo cho phía phát biết rằng việc nhận dữ liệu không thành công.
SYN	Synchronous / Idle - Đồng bộ hoá: Được sử dụng bởi hệ thống truyền đồng bộ để đồng bộ hoá quá trình truyền dữ liệu.
ETB	End of Transmission Block – Kết thúc khối truyền: Chi ra kết thúc khối dữ liệu được truyền.

# Các mã điều khiển (tiếp)

<b>Các mã ký tự điều khiển phân cách thông tin</b>	
FS	File Separator - Ký hiệu phân cách tập tin: Đánh dấu ranh giới giữa các tập tin.
GS	Group Separator - Ký hiệu phân cách nhóm: Đánh dấu ranh giới giữa các nhóm tin (tập hợp các bản ghi).
RS	Record Separator - Ký hiệu phân cách bản ghi: Đánh dấu ranh giới giữa các bản ghi.
US	Unit Separator - Ký hiệu phân cách đơn vị: Đánh dấu ranh giới giữa các phần của bản ghi.

# Các mã điều khiển (tiếp)

Các mã ký tự điều khiển khác	
NUL	Null - Ký tự rỗng: Được sử dụng để điền khoảng trống khi không có dữ liệu.
BEL	Bell - Chuông: Được sử dụng phát ra tiếng bíp khi cần gọi sự chú ý của con người.
SO	Shift Out – Dịch ra: Chỉ ra rằng các mã tiếp theo sẽ nằm ngoài tập ký tự chuẩn cho đến khi gặp ký tự SI.
SI	Shift In – Dịch vào: Chỉ ra rằng các mã tiếp theo sẽ nằm trong tập ký tự chuẩn.
DLE	Data Link Escape - Thoát liên kết dữ liệu: Ký tự sẽ thay đổi ý nghĩa của một hoặc nhiều ký tự liên tiếp sau đó.
DC1÷DC4	Device Control - Điều khiển thiết bị : Các ký tự dùng để điều khiển các thiết bị phụ trợ.
CAN	Cancel – Huỷ bỏ: Chỉ ra rằng một số ký tự nằm trước nó cần phải bỏ qua.
EM	End of Medium – Kết thúc phương tiện: Chỉ ra ký tự ngay trước nó là ký tự cuối cùng có tác dụng với phương tiện vật lý.
SUB	Substitute – Thay thế: Được thay thế cho ký tự nào được xác định là bị lỗi.
ESC	Escape – Thoát: Ký tự được dùng để cung cấp các mã mở rộng bằng cách kết hợp với ký tự sau đó.
DEL	Delete – Xóa: Dùng để xóa các ký tự không mong muốn.

# Các ký tự mở rộng

- Các ký tự mở rộng được định nghĩa bởi:
  - nhà chế tạo máy tính
  - người phát triển phần mềm.
- Ví dụ:
  - Bộ mã ký tự mở rộng của IBM → IBM-PC.
  - Bộ mã ký tự mở rộng của Apple → Macintosh
  - Bộ mã tiếng Việt TCVN3.



## 2. Bộ mã hợp nhất: Unicode

- Do các hãng máy tính hàng đầu thiết kế
- Bộ mã 16, 32-bit
- Bộ mã đa ngôn ngữ
- Có hỗ trợ các ký tự tiếng Việt



HẾT CHƯƠNG 2