Cơ sở lập trình
Bài 1. Kiến thức cơ bản về máy tính

Biên soạn TS. Trần Minh Thái
Giảng viên: Lê Thị Minh Nguyện
Email: nguyenltm@huflit.edu.vn

Mục tiêu

1. Hiểu một số thuật ngữ và các khái niệm liên quan đến máy tính
2. Biết được lịch sử máy tính
3. Biết được hệ thống số
4. Hiểu và vận dụng các biểu diễn thông tin trên máy tính
5. Hiểu về các cơ chế hoạt động tính toán trong hệ thống máy tính

Các khái niệm

- Máy tính?
- Lệnh?
- Chương trình máy tính?
- Ngôn ngữ lập trình?

Các khái niệm

- Máy tính (computer) là 1 thiết bị đặc biệt
  - *+ có thể thực hiện 1 số hữu hạn các chức năng cơ bản (tập lệnh)*
  - *+ cơ chế thực hiện các lệnh là tự động và tuần tự*
  - *+ danh sách các lệnh được thực hiện được gọi là chương trình*

Các khái niệm (tt)

- Các lệnh mà máy hiểu và thực hiện được được gọi là *lệnh máy*.
- *Ngôn ngữ lập trình* dùng để miêu tả các lệnh, gồm 2 yếu tố chính:
  - + *cú pháp* qui định trật tự kết hợp các phần tử để cấu thành 1 lệnh (câu)
  - + *ngữ nghĩa* cho biết ý nghĩa của lệnh đó
- Quá trình máy tính giải quyết công việc ngoài thực tế được gọi là *lập trình* (qui trình xác định trình tự đúng các lệnh)

Các khái niệm (tt)

- Nhu cầu về máy luận lý với tập lệnh (được đặc tả bởi ngôn ngữ lập trình) cao cấp và gần gủi hơn với con người. Ta thường gồm: 1 *máy vật lý* + 1 *chương trình dịch*

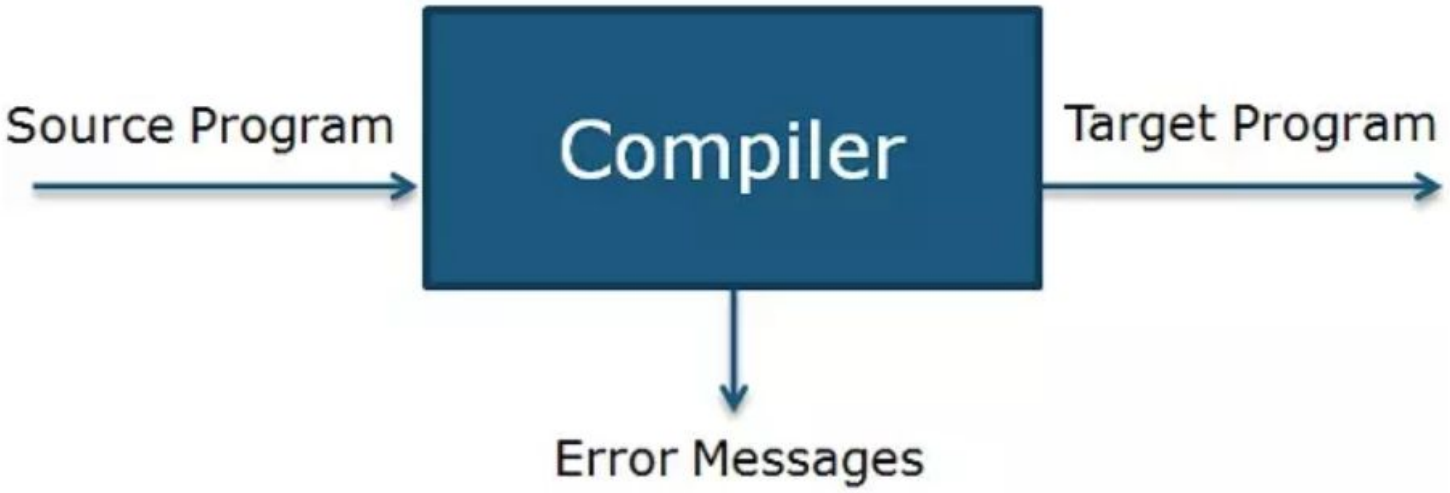
❖ Có 2 loại *chương trình dịch* :

- ❖ *trình biên dịch* (*compiler*)
- ❖ *trình thông dịch* (*interpreter*)

Các khái niệm (tt)

Trình biên dịch (*compiler*)

☐ là một trình dịch đọc một chương trình được viết bằng ngôn ngữ cấp cao và chuyển đổi nó thành ngôn ngữ máy hoặc ngôn ngữ cấp thấp và báo cáo các lỗi có trong chương trình.



Các khái niệm (tt)

- Ngôn ngữ cấp cao theo trường phái lập trình cấu trúc (Pascal, C,...)

  tập lệnh của ngôn ngữ khá mạnh và gần với tư duy con người

- Ngôn ngữ hướng đối tượng (C++, Visual Basic, Java, C#,...)

  Cải tiến PPLT cấu trúc sao cho trong sáng, ổn định, dễ phát triển và thay thế từng thành phần

- Ngôn ngữ máy vật lý ngôn ngữ cấp thấp nhất mà có thể lập trình

  là loại ngôn ngữ mà máy vật lý có thể hiểu trực tiếp, nhưng con người thì gặp khó khăn trong việc viết và bảo trì chương trình

- Ngôn ngữ assembly gần với ngôn ngữ máy + "lệnh macro" để nâng sức mạnh miêu tả giải thuật

  những lệnh cơ bản nhất tương ứng với lệnh máy dưới dạng ký hiệu gợi nhớ

Các khái niệm (tt)

- *Trình thông dịch* (*interpreter*): thực hiện kiểm tra từ vựng, phân tích cú pháp và kiểm tra các kiểu tương tự như trình biên dịch. Nhưng trình thông dịch xử lý cây cú pháp trực tiếp để truy cập các biểu thức và thực thi câu lệnh thay vì tạo mã trung gian

Hệ thống số (number system)

- Hệ thống số là công cụ để biểu thị đại lượng
- Một hệ thống số gồm 3 thành phần chính:
  1) cơ số số lượng ký số (ký hiệu để nhận dạng các số cơ bản)
  2) qui luật kết hợp các ký số để miêu tả 1 đại lượng nào đó
  3) các phép tính cơ bản trên các số

chỉ có thành phần 1 là khác nhau giữa các hệ thống số, còn 2 thành phần 2 và 3 thì giống nhau giữa các hệ thống số

Hệ thống số (tt)

- **hệ thập phân** (decimal, denary) dùng 10 ký số

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9

- **hệ nhị phân** (binary) dùng 2 ký số

  0, 1

- hệ bát phân (octal) dùng 8 ký số

  0, 1, 2, 3, 4, 5, 6, 7

- **hệ thập lục phân** (hexadecimal) dùng 16 ký số

  0 ... 9, A, B, C, D, E, F

Hệ thống số - Cơ số

Trước khi có máy tính, con người dùng hệ số đếm thập phân (10)

Ký số

0 1 2 3 4
5 6 7 8 9

Quy tắc đếm

0 → 1 → 2 → . . . → 9 →

10 → 11 → 12 → . . . → 19 →

20 → 21 → 22 → . . . → 29 → . . . → 90 → 91 → 92 → . . . → 99 →

100 → 101 → . . . → 109 → . . . → 990 → 991 → . . . → 999 →

1000 → 1001 → 1002 → . . . → 1009 →

→ . . .

Hệ thống số – Cơ số (tt)

Sau khi máy tính ra đời, các hệ số mới hình thành

Hệ nhị phân (Binary)

Ký số

0 1

Quy tắc đếm

0 → 1 →
10 → 11 →
100 → 101 → 110 → 111 →
1000 → 1001 → . . . → 1110 → 1111 →
10000 → 10001 →
→ . . .

Hệ thống số – Cơ số (tt)

- Số hệ nhị phân dài ☐ khó nhớ chỉ dùng cho máy
- Con người dùng số hệ bát phân (8) và thập lục phân (16) thay cho hệ nhị phân

Hệ bát phân (Octal)

Ký số

0 1 2 3
4 5 6 7

Quy tắc đếm

0 → 1 → 2 → . . . → 7 →
10 → 11 → 12 → . . . → 17 →
20 → 21 → 22 → . . . → 77 →
100 → 101 → 102 → . . . → 107 → . . . → 777 →
1000 → 1001 → 1002 → . . . → 1007 →
→ . . .

Hệ thống số – Cơ số (tt)

- Một ký số hệ 8 bằng 3 ký số hệ 2
- Một ký số hệ 16 bằng 4 ký số hệ 2

Hệ thập lục phân (hexadecimal)

Ký số

0 1 2 3 4 5 6 7
8 9 A B C D E F

Quy tắc đếm

0 → 1 → 2 → . . . → 9 → A → B → . . . → F →
10 → 11 → 12 → . . . → 19 → 1A → . . . → 1F → 20 → . . . → 9F →
A0 → A1 → A2 → . . . → AF → . . . → F0 → F1 → F2 → . . . → FF →
100 → 101 → 102 → . . . → 10F → . . . → FFF →
1000 → 1001 → 1002 → . . . → 100F →
→ . . .

Hệ thống số – Công thức tính trị số

Nếu B là cơ số, v_i là ký số ở hàng i (*0 là hàng đơn vị, 1 là hàng "chục", 2 là hàng "trăm", ...*) thì giá trị Q của số tính trong hệ 10 theo công thức:

$$\overline{v_n v_{n-1} \ldots v_0 . v_{-1} \ldots v_{-m}}$$

Chấm B phân

$$Q = v_n \times B^n + v_{n-1} \times B^{n-1} + \ldots + v_0 \times B^0 + v_{-1} \times B^{-1} + \ldots + v_{-m} \times B^{-m}$$

hay $$\sum_{i=-m}^{n} v_i \times B^i$$

Hệ thống số – Công thức tính trị số (tt)
VD số nguyên

1011_B

$1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 = 8+0+2+1 = 11_D$

173_O

$1\times8^2 + 7\times8^1 + 3\times8^0 = 64+56+3 = 123_D$

$A4B5_H$

$A\times16^3 + 4\times16^2 + B\times16^1 + 5\times16^0$

$10\times4096 + 4\times256 + 11\times16 + 5\times1 = 40960+1024+176+5$

$= 42165_D$

Hệ thống số – Công thức tính trị số (tt)
VD số lẻ

1011.01_2

$1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2}$

$1\times8 + 0\times4 + 1\times2 + 1\times1 + 0\times0.5 + 1\times0.25 = 11.25_D$

10.4_8

$1\times8^1 + 0\times8^0 + 4\times8^{-1}$

$1\times8 + 0\times1 + 4\times0.125 = 8.5_D$

Hệ thống số - Bảng chuyển đổi

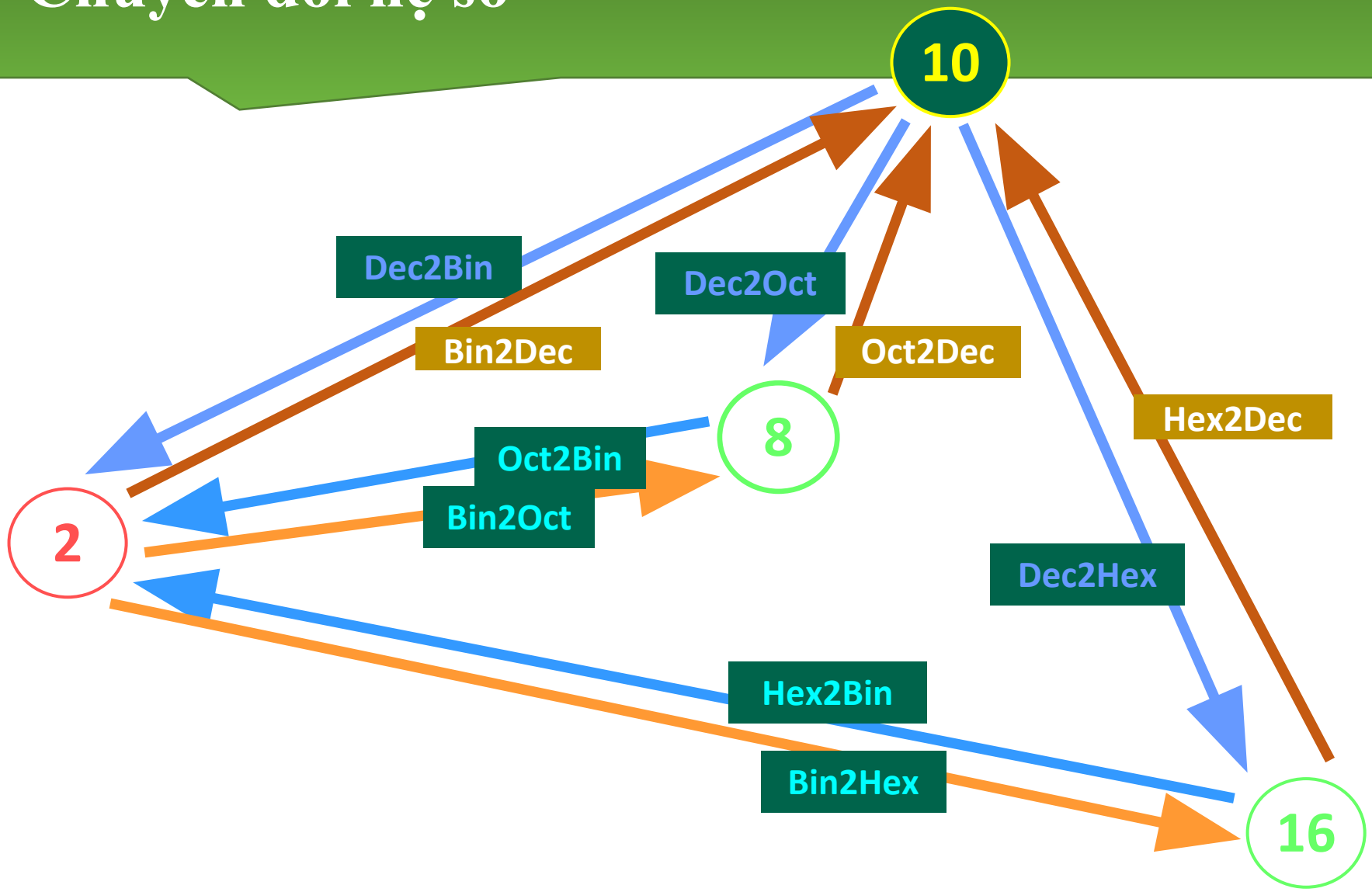
Số hệ 10	Số hệ 16	Số hệ 2
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Hệ thống số - Các phương pháp chuyển đổi

Để chuyển 1 miêu tả số từ hệ thống số này sang hệ thống số khác, ta cần dùng 1 phương pháp chuyển thích hợp. Có 4 phương pháp sau tương ứng với từng yêu cầu chuyển tương ứng:

- chuyển từ hệ thống số khác về thập phân.
- chuyển từ hệ thống số thập phân về hệ thống số khác
- chuyển từ nhị phân về thập lục phân (hay bát phân)
- chuyển từ thập lục phân (hay bát phân) về nhị phân

Chuyển đổi hệ số



Từ hệ thống số khác về thập phân

Xxx2Dec

- Để chuyển 1 miêu tả số từ hệ thống số khác (nhị phân, thập lục phân hay bát phân) sang hệ thập phân, ta dùng công thức tính Q.

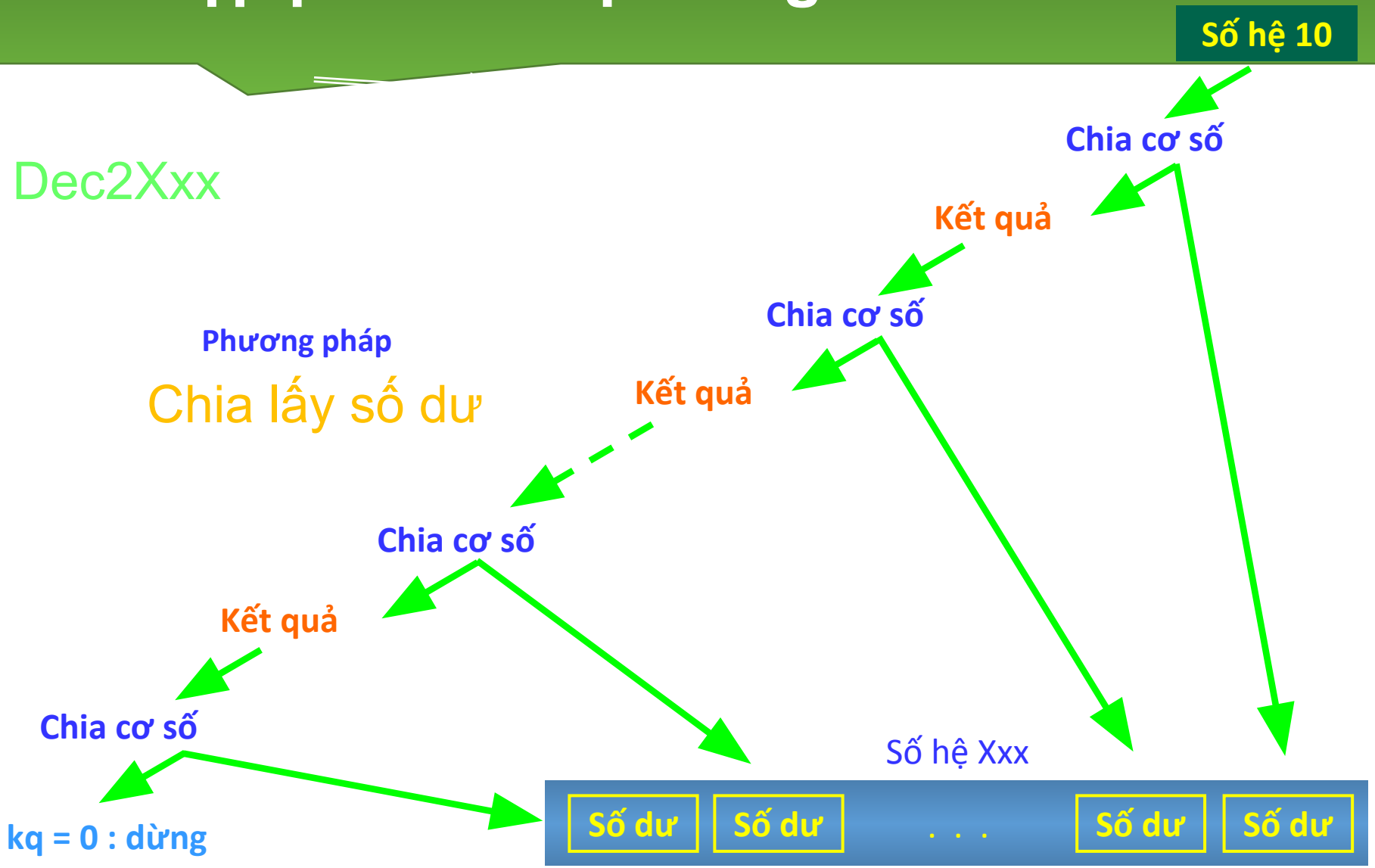
Ví dụ

$1A2_H = 1*16^2+10*16^1+2*16^0 = 256+160+2 = 418_D$

$642_O = 6*8^2+4*8^1+2*8^0 = 384+32+2 = 418_D$

$110100010_B = 2^8 + 2^7+2^5+2^1 = 256+128+32+2 = 418_D$

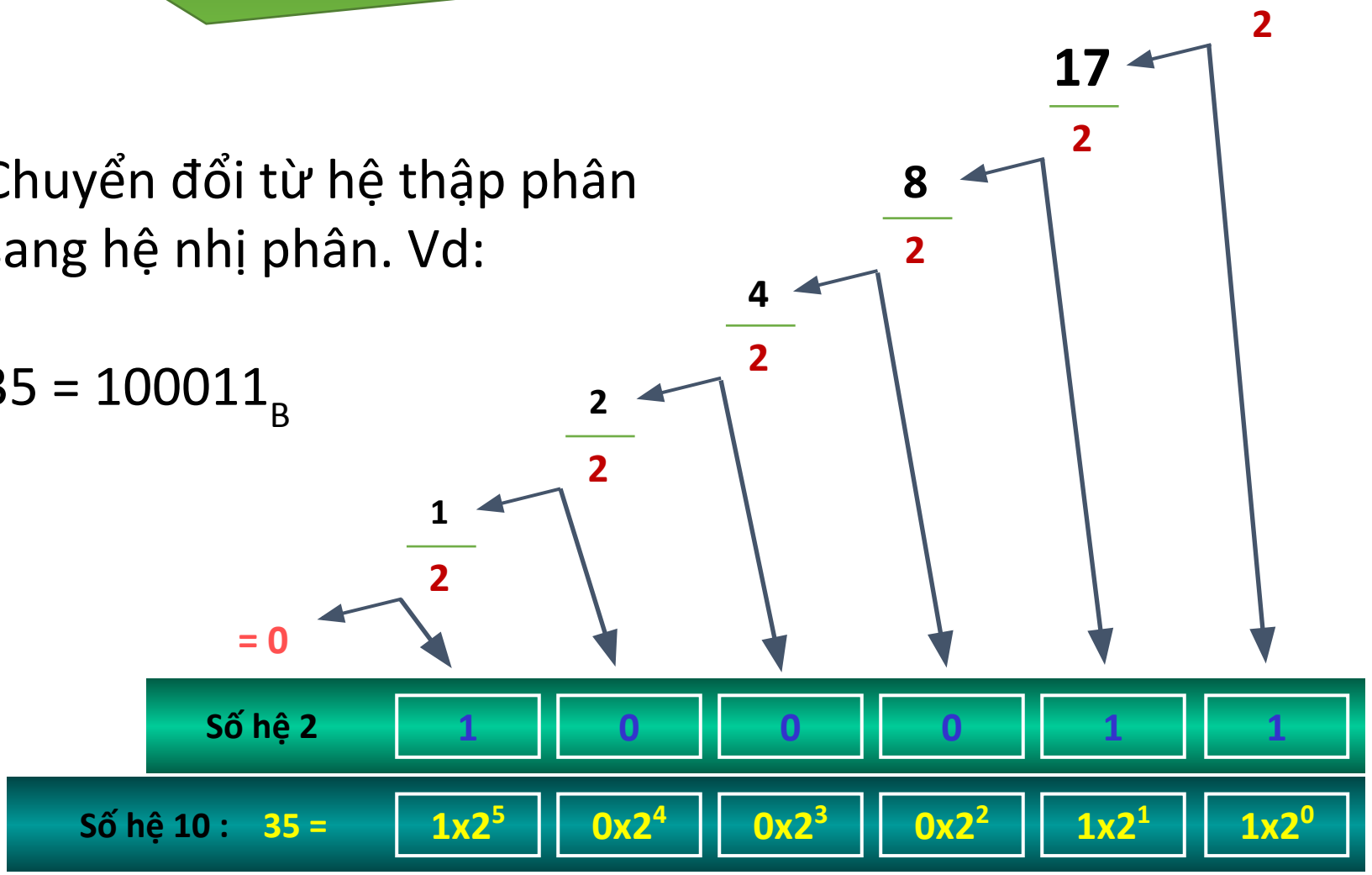
Từ thập phân về hệ thống số khác



Ví dụ Dec2Bin

Chuyển đổi từ hệ thập phân sang hệ nhị phân. Vd:

$35 = 100011_B$

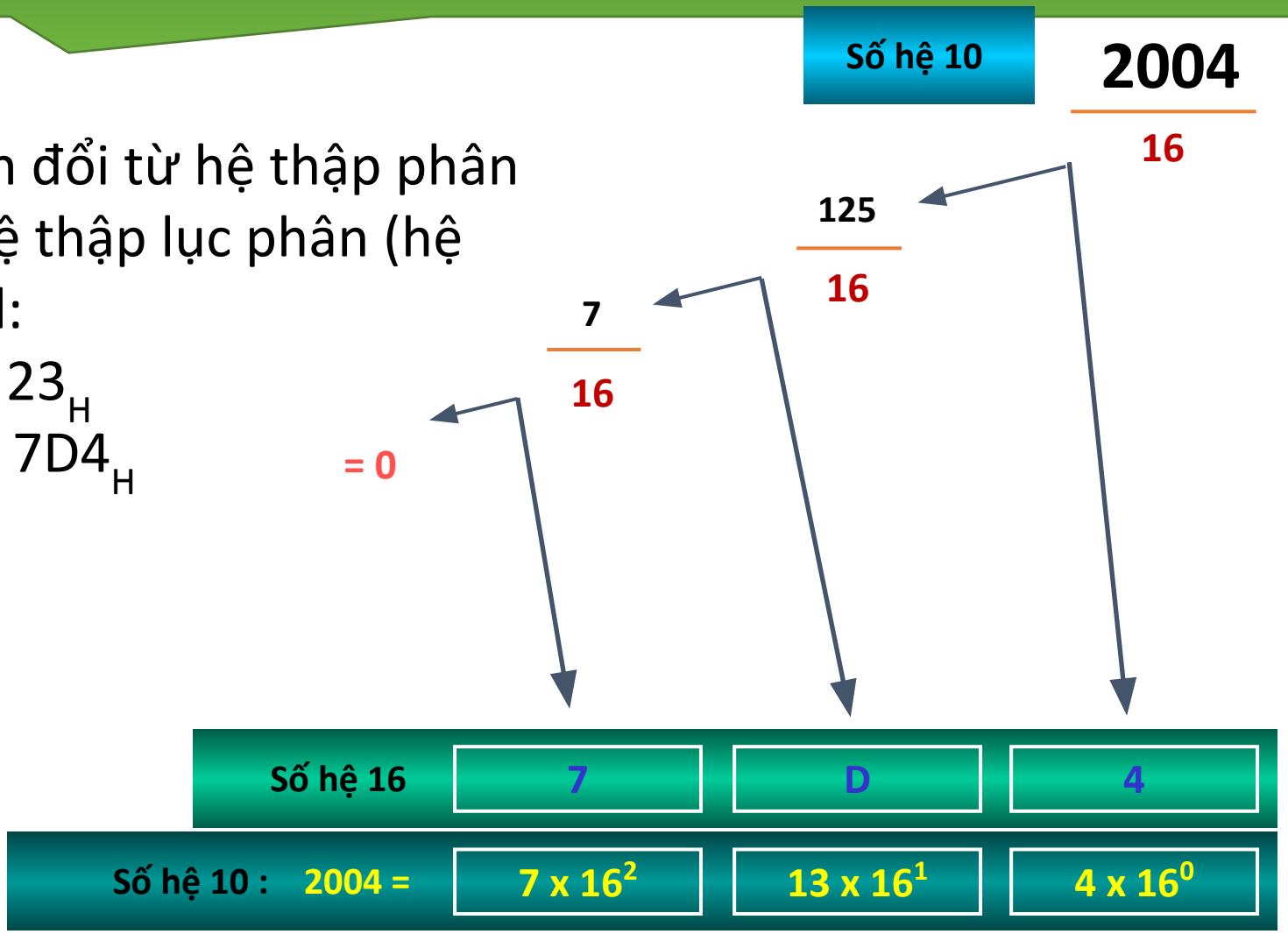


Ví dụ Dec2Hex

Chuyển đổi từ hệ thập phân sang hệ thập lục phân (hệ 16). Vd:

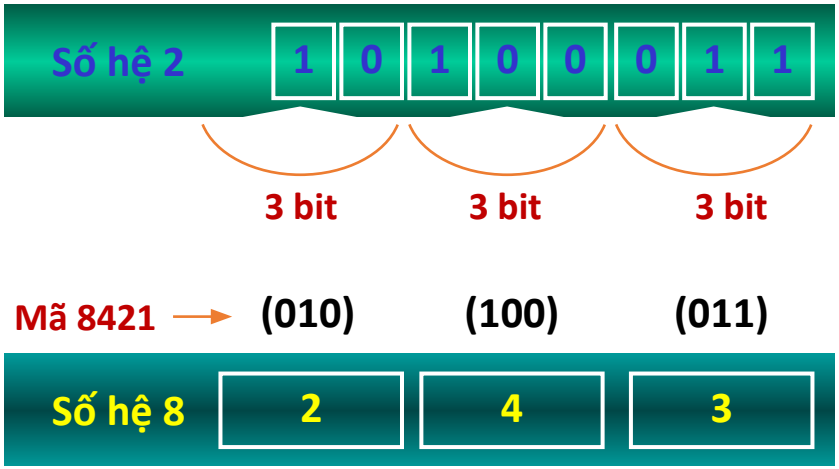
$35 = 23_H$

$2004 = 7D4_H$

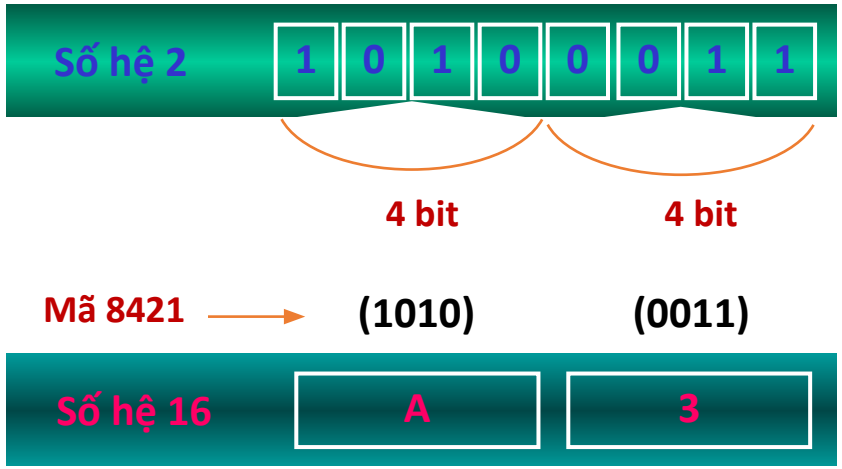


Đổi hệ 2 ra hệ 8, 16

Bin2Oct → Ghép nhóm ++ bảng chuyển miêu tả số



Bin2Hex



$10100011_B = 163$

$10100011_B = 243_O$

$10100011_B = A3_H$

Bảng chuyển miêu tả số

Số hệ 10	Số hệ 16	Số hệ 8	Số hệ 2
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
10	A	12	1010
11	B	13	1011
12	C	14	1100
13	D	15	1101
14	E	16	1110
15	F	17	1111

Chuyển đổi số

Ví dụ

$$1A2_H = 1*16^2+10*16^1+2*16^0 = 256+160+2 = 418_D$$

$$642_O = 6*82+4*81+2*80 = 384+32+2 = 418_D$$

$$110100010_B = 2^8+2^7+2^5+2^1 = 256+128+32+2 = 418_D$$

Hệ thống số đếm và các phép tính

Các phép tính cơ bản trong 1 hệ thống số:

1. phép cộng (+)
2. phép trừ (-)
3. phép chia (/)
4. phép nhân (*)
5. phép dịch trái n ký số (<<n)
6. phép dịch phải n ký số (>>n)

Hệ thống số đếm và các phép tính

Ngoài ra do đặc điểm của hệ nhị phân, hệ này còn cung cấp 1 số phép tính sau (các phép tính luận lý):

1. phép OR bit (**|**)
2. phép AND bit (**&**)
3. phép XOR bit (**^**)
4. ....

Hệ thống số đếm và các phép tính

Vd: các số sau đều ở hệ nhị phân

$$\begin{array}{r} 0\,1\,1\,0 \\ +\;0\,0\,1\,1 \\ \hline 1\,0\,0\,1 \end{array}$$

$$\begin{array}{r} 1\,0\,0\,1 \\ -\;0\,0\,1\,1 \\ \hline 0\,1\,1\,0 \end{array}$$

$$\begin{array}{r} 1\,0\,0\,1 \\ \times\;\;0\,1\,0\,1 \\ \hline 1\,0\,0\,1 \\ 0\,0\,0\,0 \\ 1\,0\,0\,1 \\ 0\,0\,0\,0 \\ \hline 0\,1\,0\,1\,1\,0\,1 \end{array}$$

$$\begin{array}{r|l} 1\,0\,1\,1 & 10 \\ \cline{2-2} -\;1\,0 & 101 \\ \cline{1-1} 0\,1 & \\ -\;0\,0 & \\ \cline{1-1} 1\,1 & \\ -\;1\,0 & \\ \cline{1-1} 0\,1 & \end{array}$$

- Cộng

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10

- Trừ

0 – 0 = 0

0 – 1 = -1 (mượn)

1 – 0 = 1

1 – 1 =0

-1 – 1 =-10

Các phép tính của đại số Boole (1)

Biểu thức Boole là 1 biểu thức toán học cấu thành từ các phép toán Boole trên các toán hạng là các biến chỉ chứa 2 trị 0 và 1.

X : biến mang giá trị {0, 1}
NOT: toán tử

X	NOT X
0	1
1	0

X	NOT X
True	False
False	True

Các phép tính của đại số Boole (2)

Biểu thức Boole là 1 biểu thức toán học cấu thành từ các phép toán Boole trên các toán hạng là các biến chỉ chứa 2 trị 0 và 1.

X, Y : hai biến
AND: toán tử

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X AND Y
False	False	False
False	True	False
True	False	False
True	True	True

Các phép tính của đại số Boole (3)

Biểu thức Boole là 1 biểu thức toán học cấu thành từ các phép toán Boole trên các toán hạng là các biến chỉ chứa 2 trị 0 và 1.

X	Y	X AND Y	X NAND Y
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Các phép tính của đại số Boole (4)

Biểu thức Boole là 1 biểu thức toán học cấu thành từ các phép toán Boole trên các toán hạng là các biến chỉ chứa 2 trị 0 và 1.

X	Y	X OR Y	X NOR Y	X XOR Y
0	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	0	0

Các phép tính của đại số Boole (5)

Biểu thức Boole là 1 biểu thức toán học cấu thành từ các phép toán Boole trên các toán hạng là các biến chỉ chứa 2 trị 0 và 1.

X	Y	NOT X	X AND Y	X NAND Y	X OR Y	X NOR Y	X XOR Y
0	0	1	0	1	0	1	0
0	1	1	0	1	1	0	1
1	0	0	0	1	1	0	1
1	1	0	1	0	1	0	0

Biểu diễn thông tin bằng hệ nhị phân

BIT (BInary digiT) : 0,1

BYTE = tổ hợp 8 bit : 01001101, 11111111

(BYTE được chọn làm đơn vị tổ chức thông tin trong máy tính)

WORD = tổ hợp 2 byte : 01011010 11100101

DWORD = tổ hợp 4 byte

1 KiloByte (KB) = 1024 bytes = 2^{10} bytes

1 MegaByte (MB) = 1024 KB = 2^{20} bytes

1 GigaByte (GB) = 1024 MB = 2^{30} bytes

1 TetraByte (TB) = 1024 GB = 2^{40} bytes

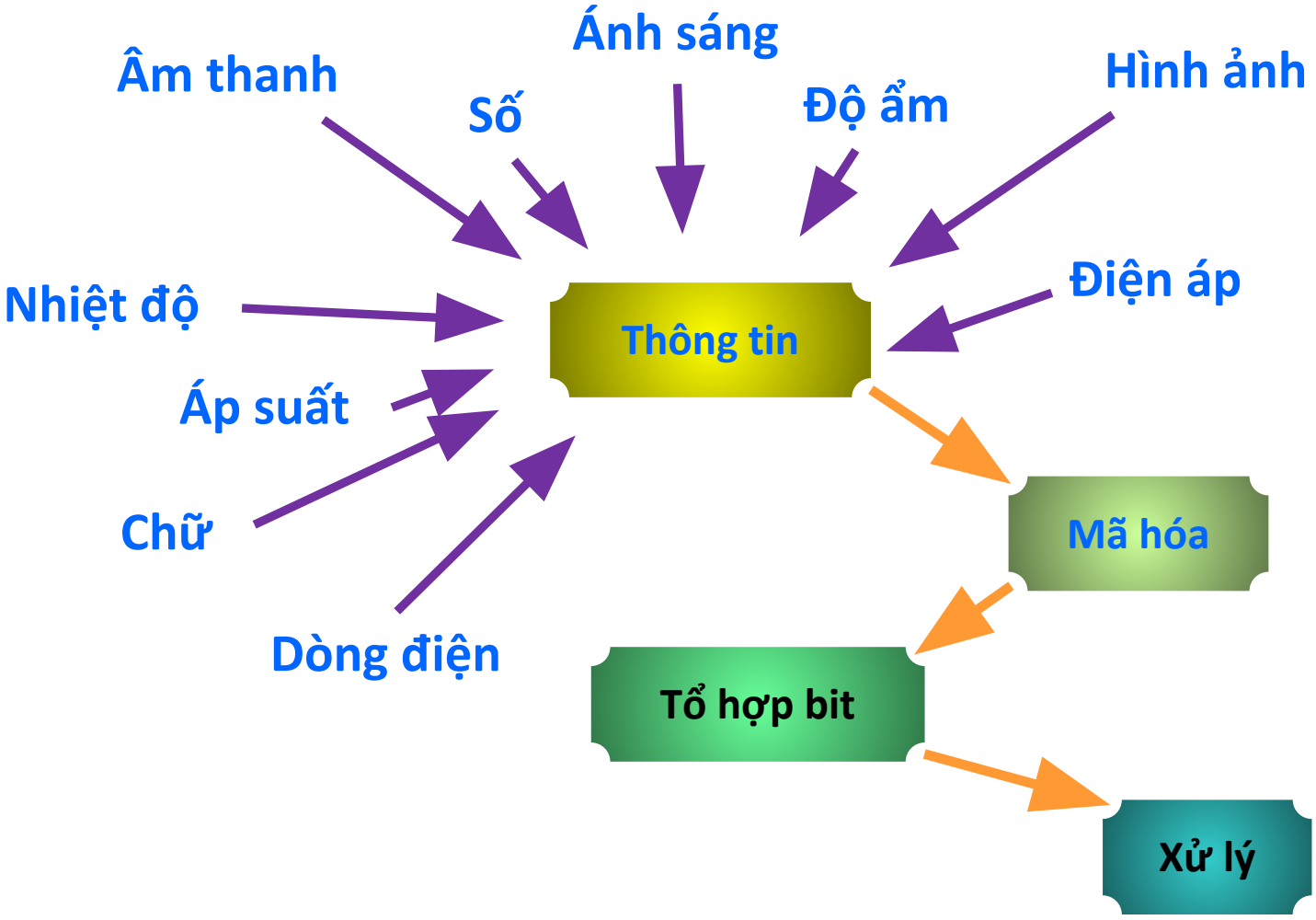
1 PetaByte (PB)= 1024 TB = 2^{50} bytes

1 số dài n bit thì biểu diễn được 2^n giá trị

Biểu diễn dữ liệu

- Máy tính làm việc trên số nhị phân.
- Con người không thể làm việc với số nhị phân vì dài, khó nhớ.
- Dữ liệu cần biểu diễn, xử lý, lưu trữ bằng máy tính gồm có đại lượng số và phi số.
- Dữ liệu đưa vào máy tính phải được mã hóa thành số nhị phân (code) rồi mới xử lý.

Mã hóa thông tin đầu vào



Biểu diễn số (nguyên) n-bit

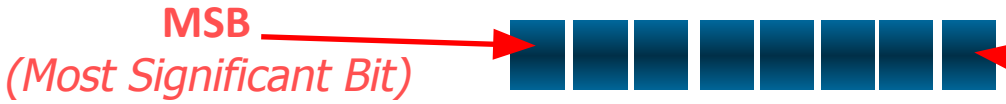
Số không dấu

Số n bit có giá trị : $0 \div (2^n - 1)$

Số 8 bit có giá trị : 0 ÷ 255
Số 16 bit có giá trị : 0 ÷ 65 535
Số 32 bit có giá trị : 0 ÷ 4 294 967 295

Số có dấu

Qui ước: chọn bit có trọng số cao nhất (MSB) làm bit dấu



bit dấu = 0 là số dương - bit dấu = 1 là số âm
sử dụng số bù 2 : -1 = 1111 1111, -2 = 1111 1110, . . .
-127 = 1000 0001, -128 = 1000 0000
Số 8 bit có dấu có giá trị : -128 ÷ +127
Số 16 bit có dấu có giá trị : -32768 ÷ +32767

Biểu diễn số nguyên có dấu (1)

Số nguyên 16-bit có dấu:

- Phần dương: [0, 1, ..., 32767], được miêu tả theo công thức Q.
- Phần âm: [-1, -2,..., -32768], được miêu tả ở dạng số bù 2 như sau :
- ✔ Số bù 1 của 1 số n bit là n bit mà mỗi bit là ngược với bit gốc (0 → 1 và 1 → 0)
- ✔ Số bù 2 của 1 số n bit là số bù 1 của số đó rồi tăng lên 1 đơn vị.

Sự biểu diễn	Giá trị
0000 0000 0000 0000	0
0000 0000 0000 0001	1
...	...
0111 1111 1111 1111	32767
1000 0000 0000 0000	-32768
1000 0000 0000 0001	-32767
...	...
1111 1111 1111 1111	-1

Biểu diễn số nguyên có dấu (2)

- Vì mỗi ô nhớ máy tính chỉ chứa được 1 byte, do đó ta phải dùng nhiều ô liên tiếp (2 hay 4) để chứa số nguyên. Có 2 cách chứa các byte của số nguyên (hay dữ liệu khác) vào các ô nhớ : BE & LE.
- Cách BE (Big Endian) chứa byte trọng số cao nhất vào ô nhớ địa chỉ thấp trước, sau đó lần lượt đến các byte còn lại.
- Cách LE (Little Endian) chứa byte trong số nhỏ nhất vào ô nhớ địa chỉ thấp trước, sau đó lần lượt đến các byte còn lại.
- ✔ CPU Intel & HĐH Windows sử dụng cách LE để chứa số nguyên vào bộ nhớ (Integer và Long).

Biểu diễn số nguyên có dấu (3)

❖ Số 15 được miêu tả dưới dạng nhị phân 16 bit như sau :

0000 0000 0000 1111

- Nếu dùng 2 byte để lưu trữ, có thể dùng dạng 16 bit viết ngắn gọn là $000F_H$. Nếu lưu vào bộ nhớ dưới dạng LE (Little Endian) thì ô nhớ có địa chỉ thấp (i) chứa byte $0F_H$, và ô nhớ kế (i+1) chứa byte 00_H.
- Trường hợp dùng 4 byte: $0000000F_H$ và lưu vào bộ nhớ dạng LE tốn 4 ô nhớ với giá trị lần lượt từ địa chỉ thấp đến cao là $0F_H$, 00_H, 00_H, 00_H.

✔ Số bù 1 của 15 là 1111 1111 1111 0000,

✔ Số bù 2 của 15 là 1111 1111 1111 0001

 Như vậy -15 được lưu vào máy dạng Integer là 2 byte có giá trị $FFF1_H$. Nếu lưu vào ô nhớ dạng LE thì ô nhớ có địa chỉ thấp (i) chứa byte $F1_H$, và ô nhớ kế (i+1) chứa byte FF_H.

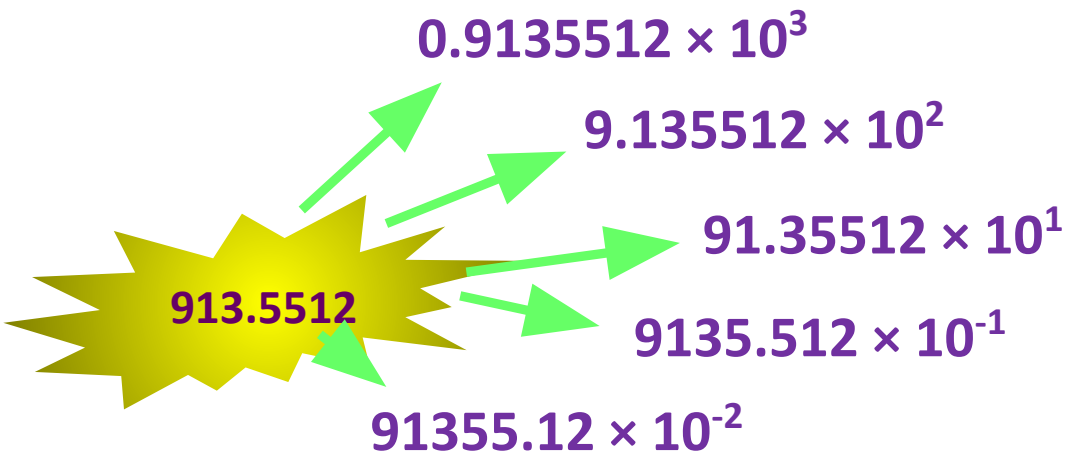
Số thực - số chấm động

Số chấm động (floating point) dùng để tính toán trên số thực.

$\pm m \times B^{\pm e}$

- m (mantissa) quyết định độ chính xác
- B (base)
- e (exponent) quyết định độ lớn/nhỏ

Một giá trị có thể biểu diễn dưới nhiều dạng



- Khó xử lý
- Cần chuẩn hóa

Q&A