Chương 2: Biểu diễn thông tin trong máy tính

Nội dung

2.1 Hệ đếm (4t)

- **2.1.1 Hệ đếm cơ số 2**
- **2.1.2 hệ đếm cơ số 8**
- **2.1.3 Hệ đếm cơ số 16**
- **2.1.4 Hệ đếm BCD**
- **2.1.5 Chuyển đổi giữa các hệ đếm**

2.2 Biểu diễn thông tin trong máy tính (4t)

- **2.2.1 Biểu diễn số nguyên**
- **2.2.2 Biểu diễn số thực**
- **2.2.3 Biểu diễn ký tự**

Nội dung

2.1 Hệ đếm

- 2.1.1 Hệ đếm cơ số 2
- 2.1.2 hệ đếm cơ số 8
- 2.1.3 Hệ đếm cơ số 16
- 2.1.4 Hệ đếm BCD
- 2.1.5 Chuyển đổi giữa các hệ đếm

2.2 Biểu diễn thông tin trong máy tính

- 2.2.1 Biểu diễn số nguyên
- 2.2.2 Biểu diễn số thực
- 2.2.3 Biểu diễn ký tự

2.1 Hệ đếm

- Hệ đếm: là tập hợp các ký hiệu và quy tắc sử dụng tập ký hiệu đó để biểu diễn và xác định các giá trị của các số. Mỗi hệ đếm có một số ký số (digit) hữu hạn;
- Tổng số ký số của mỗi hệ đếm gọi là cơ số (base hay radix), ký hiệu là b;
- Hệ đếm cơ số b (b>=2, b là số nguyên dương) mang tính chất sau:
  - Có b ký số thể hiện giá trị số, ký số nhỏ nhất là 0 và lớn nhất là b-1
  - Giá trị vị trí thứ n trong một số của hệ đếm bằng cơ số b lũy thừa n: b^n
  - Số N(n) trong hệ đếm cơ số (b) được biểu diễn bởi:

$$N(b)=a_n a_{n-1} a_{n-2} \ldots a_1 a_0 a_{-1} a_{-2} \ldots a_m$$

  - Trong đó, số N(b) có n+1 ký số biểu diễn cho phần nguyên và m ký số là biểu diễn cho phần b_phân, và có giá trị là:

$$N(b)=a_n.b^n+a_{n-1}.b^{n-1}+a_{n-2}.b^{n-2}+\ldots+a_1.b^1+a_0.b^0+a_{-1}.b^{-1} +a_{-2}.b^{-2}+\ldots+a_{-m}.b^{-m}$$

2.1.1 Hệ cơ số 10

- Hệ đếm thập phân (Decimal system): Là một trong các phát minh của người Ả Rập Cổ, bao gồm 10 ký số theo ký hiệu sau: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Qui tắc tính giá trị của hệ đếm này là mỗi đơn vị ở một hàng bất kỳ có giá trị bằng 10 đơn vị của hàng kế cận bên phải.

$$D = \sum_{i=0}^{p-1} d_i 10^i$$

Ví dụ 1:

$123 = 1 * 10^2 + 2 * 10^1 + 3*10^0$

$5246 = 5 * 10^3 + 2 *10^2 + 4 * 10^1 + 6 * 10^0$

$= 5 * 1000 + 2 * 100 + 4 * 10 + 6 * 1$

$= 5000 + 200 + 40 + 6$

Ví dụ 2:

$254.68 = 2 * 10^2 + 5 * 10^1 + 4 * 10^0 + 6 * 10^{-1} + 8 * 10^{-2}$

2.1.2 Hệ cơ số 2

- Hệ đếm nhị phân (Binary system): Là hệ đếm đơn giản nhất với 2 chữ số là 0 và 1. Mỗi chữ số nhi phân gọi là BIT;

$$B = \sum_{i=-n}^{p-1} b_i 2^i$$

Ví dụ:

Số $11101.11_{(2)}$ sẽ tương đương với giá trị thập phân là :

Số nhị phân :	1	1	1	0	1.	1	1
Số vị trí :	4	3	2	1	0	-1	-2
Trị vị trí :	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}
Hệ 10 là :	16	8	4	2	1	0.5	0.25

Như vậy:

$11101.11_{(2)} = 1x16+1x8+1x4+0x2+1x1+1x0.5+1x0.25$

$= 29.75_{(10)}$

$10101_{(2)} = 1x2^4+ 0x2^3+ 1x2^2+ 0x2^1+ 1x2^0$

$= 16 + 0 + 4 + 0 + 1 = 21_{(10)}$

2.1.3 Hệ cơ số 8

- Hệ đếm cơ số 8 (Octal system): Là hệ đếm với b = 8 = 2^3
  Trong hệ bát phân, trị vị trí là lũy thừa của 8.

  Ví dụ:

  $235.64_{(8)} = 2x8^2 + 3x8^1 + 5x8^0 + 6x8^{-1} + 4x8^{-2}$

  $= 157.8125_{(10)}$

- Nếu dùng 1 tập hợp 3 bit thì có thể biểu diễn 8 trị khác nhau : 000, 001, 010, 011, 100, 101, 110, 111. Các trị này tương đương với 8 trị trong hệ thập phân là 0, 1, 2, 3, 4, 5, 6, 7

2.1.4 Hệ cơ số 16

- Hệ thập lục phân (Hexa-decimal system): Hệ đếm thập lục phân là hệ cơ sơ b=16 = 2^4, tương đương với tập hợp 4 chữ số nhị phân (4 bit);
- Khi thể hiện ở dạng hexa-decimal, ta có 16 ký tự gồm 10 chữ số từ 0 đến 9, và 6 chữ in A, B, C, D, E, F để biểu diễn các giá trị số tương ứng là 10, 11, 12, 13, 14, 15. Với hệ thập lục phân, trị vị trí là lũy thừa của 16;

Ví dụ:

$34F5C_{(16)} = 3x16^4 + 4x16^3 + 15x16^2 + 5x16^1 + 12x16^0$

$= 216294_{(10)}$

Ghi chú: một số ngôn ngữ lập trình qui định viết số hexa phải có chữ H ở cuối chữ số.

Ví dụ: Số 15 viết là FH.

2.1.4 Hệ BCD (số hệ mười mã hóa bằng hệ hai)

- BCD – Binary code decimal – Đây là hệ lai giữa hệ 10 và hệ 2
- Dùng 4 số hệ hai để mã hóa một số hệ mười có giá trị nằm trong khoảng 0 ..9
- Bảng mã BCD:

Thập phân	BCD	Thập phân	BCD
0	0000	8	1000
1	0001	9	1001
2	0010		1010
3	0011		1011
4	0100		1100
5	0101		1101
6	0110		1110
7	0111		1111

2.1.5 Chuyển đổi giữa các hệ đếm

Chuyển từ cơ số b (khác 10) sang hệ cơ số 10:

- **Bước 1**: Xác định giá trị vị trí của mỗi ký số
- **Bước 2**: Nhân giá trị vị trí với ký số của cột tương ứng
- **Bước 3**: Cộng kết quả của các phép tính nhân trong bước 2
- Tổng cuối cùng sẽ là giá trị của hệ thập phân

Ví dụ 1:

$$11001_{(2)} = ?_{(10)}$$
$$= 1x2^4 + 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0$$
$$= 16 + 8 + 0 + 0 + 1$$
$$= 25_{(10)}$$

Ví dụ 2:

$$4706_{(8)} = ?_{(10)}$$
$$= 4x8^3 + 7x8^2 + 0x8^1 + 6x8^0$$
$$= 2048 + 448 + 0 + 6$$
$$= 2502$$

Kết quả: $4706_{(8)} = 2502_{(10)}$

2.1.5 Chuyển đổi giữa các hệ đếm

Ví dụ 3:

$1AC_{(16)} = ?_{(10)}$

Giải

$1AC(16) = 1x16^2 + Ax16^1 + Cx16^0$

$= 1x256 + 10x16 + 12x1$

$= 256 + 160 + 12$

$= 428$

Kết quả: $1AC_{(16)} = 428_{(10)}$

Ví dụ 4:

$4052(7) = ?(10)$

Giải

$4052(7) = 4x7^3 + 0x7^2 + 5x7^1 + 2x7^0$

$= 1372 + 0 + 35 + 2$

$= 1409$

Kết quả: $4052_{(7)} = 1409_{(10)}$

2.1.5 Chuyển đổi giữa các hệ đếm

Bài tập ví dụ: chuyển đổi ra hệ 10

a) 110110_2

b) 2573_6

c) $2A3B_{16}$

d) 1234_9

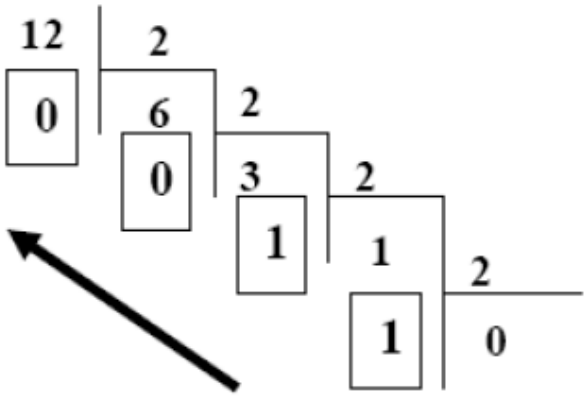
2.1.5 Chuyển đổi giữa các hệ đếm

Chuyển một số nguyên từ hệ cơ số 10 sang hệ cơ số b

- Tổng quát:
  - Lấy số **nguyên** thập phân N(10) lần lượt chia cho b cho đến khi thương số bằng 0;
  - Kết quả số chuyển đổi N(b) là các dư số trong phép chia viết ra theo thứ tự ngược lại.

Ví dụ:

Số 12(10) = ?(2). Dùng phép chia cho 2 liên tiếp, ta có một loạt các số dư như



Kết quả: $12_{10} = 1100_{(2)}$

2.1.5 Chuyển đổi giữa các hệ đếm

Bài tập ví dụ: chuyển các số sau:

a) 435_{10}

b) 1694_{10}

c) 32_{10}

d) 135_{10}

- Sang hệ nhị phân
- Hệ cơ số 8
- Hệ cơ số 16

2.1.5 Chuyển đổi giữa các hệ đếm

Chuyển phần thập phân từ hệ thập phân sang hệ cơ số b

Tổng quát:

- Lấy phần thập phân N(10) lần lượt nhân với b cho đến khi phần thập phân của tích số bằng 0;
- Kết quả số chuyển đổi N(b) là các số phần nguyên trong phép nhân viết ra theo thứ tự tính toán.

Ví dụ :

0. 6875(10) = ? (2)

0. 6875 * 2 = **1** . 375

0. 375 * 2 = **0** . 75

0. 75 * 2 = **1** . 5

0. 5 * 2 = **1** . 0

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

Bài tập: 456.375(10) = ?(2)

2.1.5 Chuyển đổi giữa các hệ đếm

- **Chuyển từ cơ số khác 10 sang cơ số khác 10:**

Bước 1: Chuyển số gốc sang hệ thập phân (hệ 10).

Bước 2: Chuyển số hệ thập phân thu được sang cơ số mới.

Ví dụ 1:

545(6) = ? (4)

Bước 1: Chuyển từ hệ 6 sang hệ 10

$545 = 5 \times 6^2 + 4 \times 6^1 + 5 \times 6^0$

$= 5 \times 36 + 4 \times 6 + 5 \times 1$

$= 180 + 24 + 5$

$= \mathbf{209_{(10)}}$

Bước 2: Chuyển $209_{(10)}$ sang hệ 4

209	4	Số dư
52		1
13		0
3		1
0		3

Kết quả: $545_{(6)} = 209_{(10)} = 3101_{(4)}$

101110(2) = ? (8)

11010011(2) = ? (16)

2.1.5 Chuyển đổi giữa các hệ đếm

Chuyển đổi giữa hệ nhị phân sang hệ cơ số 8, cơ số 16 và ngược lại:

- Hệ nhị phân sang hệ 8 (hệ 16): nhóm 3bit (4bit) rồi chuyển tương ứng
- Từ hệ 8 (hệ 16) sang hệ nhị phân: khai triển 1 số thành 3bit (4bit) tương ứng

Ví dụ 1:

Chuyển $562_{(8)} = ?_{(2)}$

Bước 1: Chuyển mỗi số bát phân thành 3 số nhị phân

$5_{(8)} = 101_{(2)}$

$6_{(8)} = 110_{(2)}$

$2_{(8)} = 010_{(2)}$

Bước 2: Kết nối các nhóm nhị phân.

$562_{(8)} = 101 \quad 110 \quad 010$

$5 \quad 6 \quad 2$

Kết quả $562_{(8)} = 101110010_{(2)}$

Chuyển $6751_{(8)} = ?_{(2)}$

2.1.5 Chuyển đổi giữa các hệ đếm

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

2.1.5 Chuyển đổi giữa các hệ đếm

Ví dụ:

Chuyển 11010011(2) = ? (16)

Bước 1: Chia số nhị phân thành các nhóm bốn chữ số.

1101 0011

Bước 2: Chuyển mỗi nhóm 4 số nhị phân thành một số thập lục phân

$1101_{(2)} = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$= 8 + 4 + 0 + 1$

$= 13_{(10)}$

$= D_{(16)}$

$0011_{(2)} = 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 0 + 0 + 2 + 1$

$= 3_{(16)}$

Kết quả: $\mathbf{11010011_{(2)} = D3_{(16)}}$

Chuyển $10110101100_{(2)} = ?_{(16)}$

2.1.5 Chuyển đổi giữa các hệ đếm

Ví dụ 1:

Chuyển 2AB(16) = ? (2)

Bước 1:

$$2_{(16)} = 2_{(10)} = 0010_{(2)}$$
$$A_{(16)} = 10_{(10)} = 1010_{(2)}$$
$$B_{(16)} = 11_{(10)} = 1011_{(2)}$$

Bước 2: Kết nối các nhóm nhị phân.

$$2AB_{(16)} = 0010\ \ 1010\ \ 1011$$
$$2\quad A\quad B$$

Kết quả $2AB_{(16)} = 001010101011_{(2)}$

Chuyển $ABC_{(16)} = ?_{(2)}$

2.1.5 Chuyển đổi giữa các hệ đếm

Ví dụ : Chuyển số 110.101(2) sang hệ 10

$$110.101_{(2)} = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$
$$= 4 + 2 + 0 + 0.5 + 0 + 0.125$$
$$= 6 + 0.5 + 0.125$$
$$= 6.625_{(10)}$$

Bài tập

Chuyển số 127.54(10) sang hệ 8
Chuyển số 2B.C4(16) sang hệ 10

Một số bài tập

1. Cơ số của một hệ thống số có nghĩa là gì? Cho ví dụ minh họa vai trò của cơ số trong hệ thống số.
2. Giá trị của các cơ số thập phân, thập lục phân, nhị phân và bát phân là gì?
3. Tìm các số thập phân tương đương với các số nhị phân sau:

1101011
11010
10110011
11011101
1110101
1000
10110001100
1010101100
110001
111

4. Tìm các số bát phân tương đương các số nhị phân của câu 3.
5. Tìm các số thập lục phân tương đương các số nhị phân của câu 3

Một số bài tập

6. Chuyển các số sau sang hệ thập phân
   110110(2)
   2573(6)
   2A3B(16)
   1234(9)
7. Chuyển các số hệ thập phân sang hệ nhị phân
   435(10)
   1694(10)
   32(10)
   135(10)
8. Chuyển các số thập phân trong câu 7 sang hệ bát phân.
9. Chuyển các số thập phân trong câu 7 sang hệ thập lục phân.
10. Tìm kết quả của các chuyển đổi sau:
    126(6) = ?(4)
    24(9) = ?(3)
    ABC(16) = ?(8)
    135(10) = ?(2)

Một số bài tập

11. Chuyển các số sau sang hệ nhị phân
    2AC(16)
    FAB(16)
    2614(8)
    562(8)
12. Tìm các số thập phân của các số sau:
    111.01(2)
    1001.011(2)
    247.65(8)
    A2B.D4(16)

Chương 2: Biểu diễn thông tin trong máy tính

2.1 Hệ đếm

- 2.1.1 Hệ đếm cơ số 2
- 2.1.2 hệ đếm cơ số 8
- 2.1.3 Hệ đếm cơ số 16
- 2.1.4 Hệ đếm BCD
- 2.1.5 Chuyển đổi giữa các hệ đếm

2.2 Biểu diễn thông tin trong máy tính

- 2.2.1 Biểu diễn số nguyên
- 2.2.2 Biểu diễn số thực
- 2.2.3 Biểu diễn ký tự

Chương 2: Biểu diễn thông tin trong máy tính

2.1 Hệ đếm

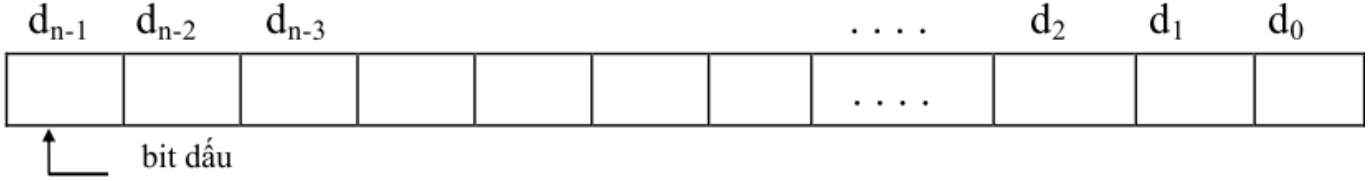
- 2.1.1 Hệ đếm cơ số 2
- 2.1.2 hệ đếm cơ số 8
- 2.1.3 Hệ đếm cơ số 16
- 2.1.4 Hệ đếm BCD
- 2.1.5 Chuyển đổi giữa các hệ đếm

2.2 Biểu diễn thông tin trong máy tính

- **2.2.1 Biểu diễn số nguyên**
- 2.2.2 Biểu diễn số thực
- 2.2.3 Biểu diễn ký tự

Biểu diễn số nguyên có dấu

- Có nhiều cách để biểu diễn một số n bit có dấu, trong tất cả mọi cách thì bit cao nhất luôn tượng trưng có dấu
- Khi đó: bit dấu có giá trị 0 thì số nguyên dương, bit dấu có giá trị là 1 thì là số nguyên âm (tuy nhiên cách biểu diễn này không đúng trong trường hợp biểu diễn bằng số thừa K – bit dấu có giá trị là 1 là số nguyên dương, bit dấu có giá trị 0 thì số là nguyên âm);



- Số nguyên có bit d_{n-1} là bit dấu và có trị số xác định bởi các bit từ d_0 đến d_{n-2}.

Biểu diễn số nguyên có dấu

- Biểu diễn bằng trị tuyệt đối và dấu
- Biểu diễn bằng số bù 1
- Biểu diễn bằng số bù 2
- Biểu diễn bằng số thừa K

Biểu diễn bằng trị tuyệt đối và dấu

- **<u>Phương pháp:</u>** bit d_{n-1} là bit dấu, các bit từ d_0 đến d_{n-2} cho giá trị tuyệt đối. Một từ n bit tương ứng với số nguyên thập phân có dấu:

$$N = (-1)^{d_{n-1}} \sum_{i=0}^{n-2} d_i 2^i$$

Ví dụ: $+25_{10}$ = 00011001B

-25_{10} = 10011001B

- **<u>Nhận xét:</u>**
  - Một byte (8bit) có thể biểu diễn các số có dấu từ -127 đến +127
  - Có 2 cách biểu diễn số 0 là 0000 0000 (+0) và 1000 0000 (-0)

Biểu diễn bằng số bù 1

- **Phương pháp:** số âm –N có được bằng cách thay các số nhị phân d_i của số dương N bằng số bù của nó (d_i=0 thì đổi thành 1 và ngược lại)

  Ví dụ: $+25_{10}$= 0001 1001B

  -25_{10} = 1110 0110B

- **Nhận xét:**
  - Một byte cho phép biểu diễn tất cả các số có dấu từ -127 đến +127
  - Có 2 cách biểu diễn số 0 là 0000 0000 (+0) và 1111 1111 (-0)

Biểu diễn bằng số bù 2

- **Phương pháp:** để có được số bù 2 của một số nào đó, người ta lấy số bù 1 rồi cộng thêm 1. như vậy một từ n bit ($d_{n-1}.....d_0$) có giá trị thập phân:

$$N = -d_{n-1}2^{n-1} + \sum_{i=0}^{n-2} d_i 2^i$$

- Một từ n bit có thể biểu diễn các số có dấu trong khoảng: -2^{n-1} đến $2^{n-1}-1$; chỉ có 1 cách duy nhất để biểu diễn số 0 (tất cả các bit =0)

  Ví dụ: $+25_{10}$= 0001 1001B

  -25_{10} = 1110 0111B

- **Nhận xét**: 1byte có thể biểu diễn số từ -128 đến +127

  Chỉ có 1 giá trị 0 (+ 0 = 0000 0000B; -0 = 0000 0000B)

  Số tận cùng bên trái thể hiện số đó là âm (giá trị 1) hoặc dương (giá trị 0).

Biểu diễn bằng số bù 2

- Ví dụ: số 4bit có dấu theo cách biểu diễn số âm bằng số bù 2

d_3	d_2	d_1	d_0	N
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7

d_3	d_2	d_1	d_0	N
1	0	0	0	-8
1	0	0	1	-7
1	0	1	0	-6
1	0	1	1	-5
1	1	0	0	-4
1	1	0	1	-3
1	1	1	0	-2
1	1	1	1	-1

- Số bù 2 được sử dụng rộng rãi trong các hệ thống máy tính, chúng ta sử dụng phương pháp này trong các phần còn lại của tài liệu này.

Biểu diễn bằng số thừa K

- **<u>Phương pháp:</u>**
  - Số dương của 1 số N có được bằng cách "***cộng thêm vào***" số thừa K được chọn sao cho tổng của K và một số âm bất kỳ luôn luôn dương.
  - Số âm –N của số N có được bằng cách lấy K-N (hay lấy bù hai của số vừa xác định)
- Phương pháp này đánh lừa máy tính, nó làm cho máy tính không phân biệt được là nó đang tương tác với số có dấu. Số thừa được tạo ra bằng cách xác định một giá trị thừa (hay còn gọi là giá trị dịch chuyển gốc 0) để làm mốc giá trị 0 giả sử là 128.
- **<u>Ví dụ 1:</u>** giá trị giả sử là +12 sẽ được tạo ra bằng cách tính toán $(128 + 12 = 140)_{10}$. Và giá trị 140 khi biểu diễn trong cơ số 2 sẽ đại diện cho số +12

Biểu diễn bằng số thừa K

$+12_{10}$ = 1000 1100B

-12_{10} = 0111 0100B

- **<u>Ví dụ 2:</u>** (+25 được biểu diễn: 128+25= 153, số 153 biểu biễn trong cơ số 2 đại diện cho +25, số -25 chính là bù 2 của +25)

  $+25_{10}$=1001 1001B

  -25_{10} = 0110 0111B

Nhận xét về biểu diễn số nguyên

- Biểu diễn số nguyên có dấu bằng số bù 2 được dùng rộng rãi cho các phép tính số nguyên, có lợi là không cần thuật toán đặc biệt nào cho các phép tính cộng, trừ; giúp dễ dàng phát hiện các trường hợp bị tràn;
- Cách biểu diễn dùng dấu, trị tuyệt đối hay số bù 1 dẫn đến thuật toán phức tạp và bất lợi vì luôn có 2 cách biểu diễn số 0; cách biểu diễn bằng dấu, trị tuyệt đối được dùng cho phép nhân của số có dấu chấm động
- Cách biểu diễn số thừa K dùng cho số mũ của các số có dấu chấm động. Cách này làm cho việc so sánh các số mũ có dấu khác nhau trở thành việc so sánh các số nguyên dương

Chương 2: Biểu diễn thông tin trong máy tính

2.1 Hệ đếm

- 2.1.1 Hệ đếm cơ số 2
- 2.1.2 hệ đếm cơ số 8
- 2.1.3 Hệ đếm cơ số 16
- 2.1.4 Hệ đếm BCD
- 2.1.5 Chuyển đổi giữa các hệ đếm

2.2 Biểu diễn thông tin trong máy tính

- 2.2.1 Biểu diễn số nguyên
- **2.2.2 Biểu diễn số thực**
- 2.2.3 Biểu diễn ký tự

Biểu diễn dạng dấu chấm tĩnh

- Số dấu chấm tĩnh có số lượng chính xác các bit để biểu diễn số.

  Ví dụ:

  Trong hệ thập phân, số 254_{10} có thể biểu diễn:

  $254*10^0$;$25.4*10^1$;$2.54*10^2$;$0.254*10^3$;$0.0254*10^4$,…

  Trong hệ nhị phân số $(0.00011)_2$ (tương đương với số 0.09375_{10}) có thể biểu diễn:

  $0.00011; 0.00011 * 2^0 ; 0.0011 * 2^{-1}; 0.011 * 2^{-2}; 0.11 * 2^{-3}; 1.1 * 2^{-4}$

- Các cách biểu diễn này gây khó khăn trong một số phép so sánh các số. Để dễ dàng trong các phép tính, các số được chuẩn hoá về một dạng biểu diễn:

  $\pm 1.\ fff...f \times 2^{\pm E}$ **Trong đó: f là phần lẻ; E là phần mũ**

Hạn chế của dấu chấm tĩnh

- Để biểu diễn số 1 tỉ, ta cần khoảng 40 bit ở phía bên trái của dãy số. Để biểu diễn độ chính xác 1 phần tỉ, ta cũng cần phải sử dụng khoảng 40 bit nữa ở phía bên phải "sau dấu chấm". Như vậy, ta cần khoảng 80 bit để biểu diễn số trên.

Ví dụ: khối lượng mặt trời: $M_\odot = 1,9891 \times 10^{30}\ \mathrm{kg}$

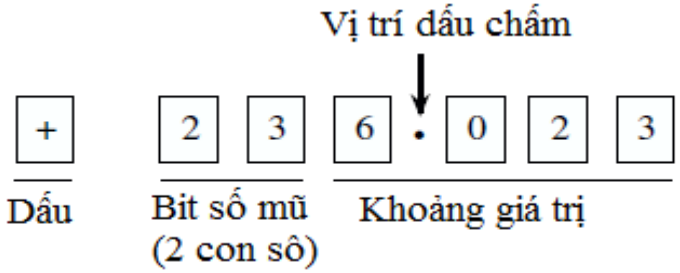
Khối lượng electron: 9.1094×10^{-31} kg

- Nếu dùng phương pháp dấu chấm tĩnh biểu diễn các số này có nhiều khó khăn và lãng phí

Biểu diễn số bằng dấu chấm động

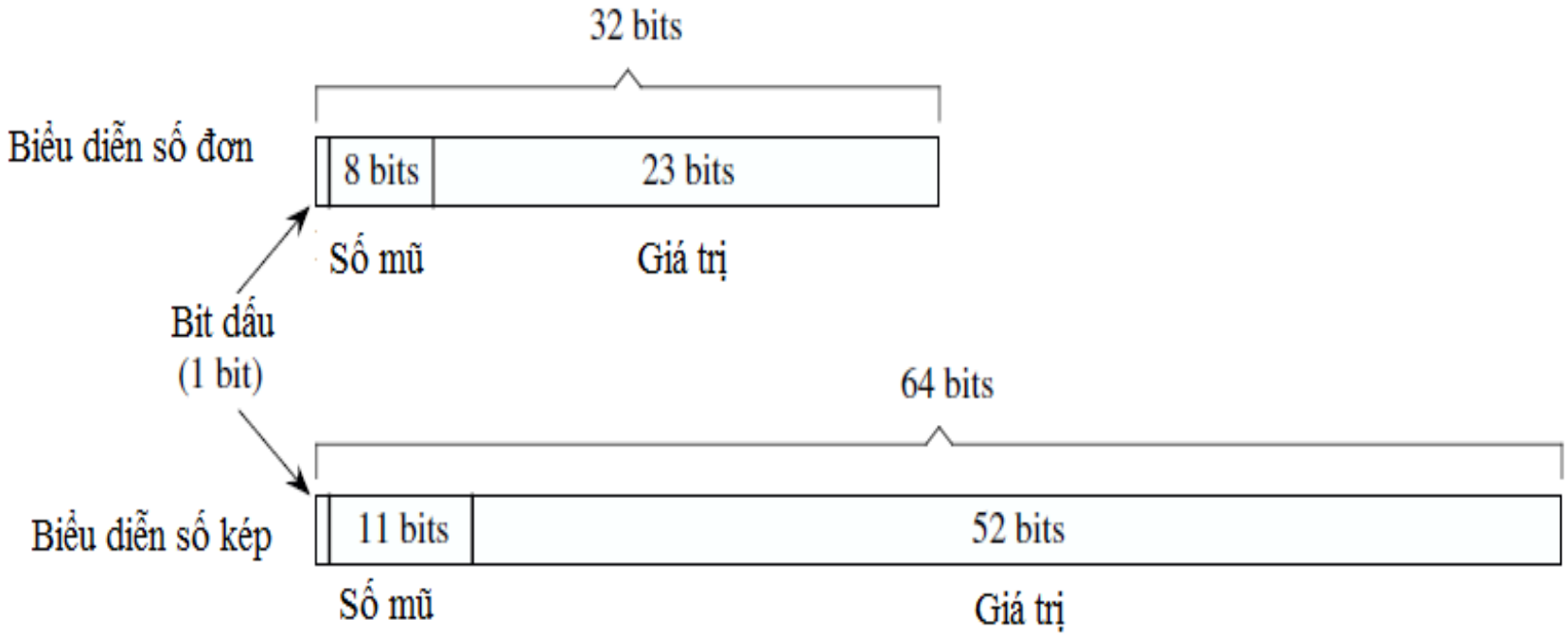
- Số chấm động được chuẩn hoá, cho phép biểu diễn gần đúng các số thập phân rất lớn hay rất nhỏ dưới dạng một số nhị phân theo một dạng qui ước.
- Thành phần của số chấm động bao gồm: **phần dấu, phần mũ và phần định trị**.
- Như vậy, cách này cho phép biểu diễn gần đúng các số thực, tất cả các số đều có cùng cách biểu diễn.
- **<u>Ví dụ:</u>**

$$+6.023 \times 10^{23}$$



Biểu diễn số bằng dấu chấm động

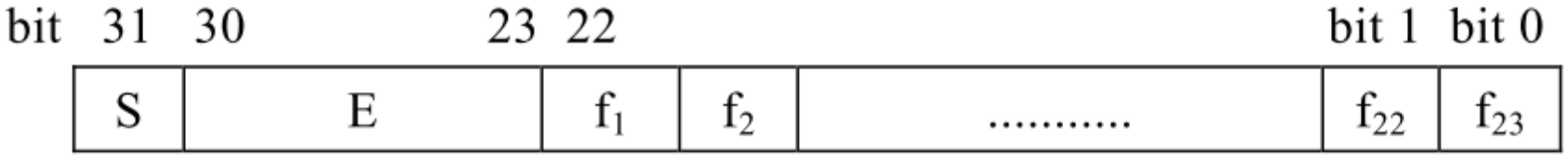
- Có nhiều cách biểu diễn dấu chấm động, trong đó cách biểu diễn theo chuẩn IEEE 754 được dùng rộng rãi trong khoa học máy tính hiện nay



Biểu diễn số bằng dấu chấm động

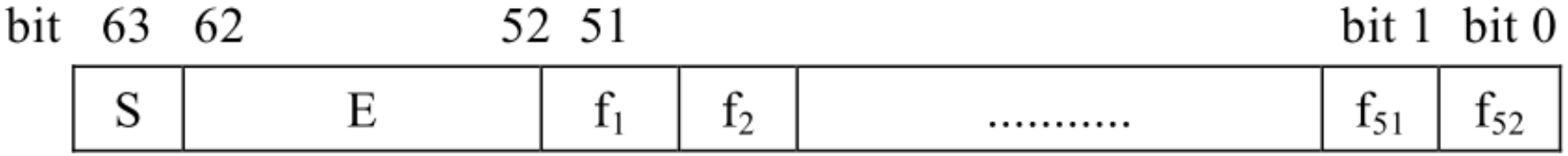
- **<u>Số thực có độ chính xác đơn:</u>**

Số này tương ứng với số thực $(-1)^S * (1,f_1\ f_2\ .....\ f_{23}) * 2^{(E\ -\ 127)}$



- **<u>Số thực có độ chính xác kép:</u>**

Số này tương ứng với số thực $(-1)^S * (1,f_1\ f_2\ .....\ f_{52}) * 2^{(E\ -\ 1023)}$



Biểu diễn số bằng dấu chấm động

- Để thuận lợi trong một số phép tính toán, IEEE định nghĩa một số dạng mở rộng của chuẩn IEEE 754

Tham số	Chính xác đơn	Mở rộng chính xác đơn	Chính xác kép	Mở rộng chính xác kép
Chiều dài (bit)	**32**	**≥ 43**	**64**	**≥ 79**
Chiều dài trường mũ (E)	**8**	**≥ 11**	**11**	**≥ 15**
Số thừa	**127**	**-**	**1023**	**-**
Giá trị mũ tối đa	**127**	**≥ 1023**	**1023**	**≥ 16383**
Giá trị mũ tối thiểu	**-126**	**≤ - 1022**	**-1022**	**≤ -16382**
Chiều dài trường lẻ F (bit)	**23**	**≥ 31**	**52**	**≥63**

Biểu diễn số bằng dấu chấm động

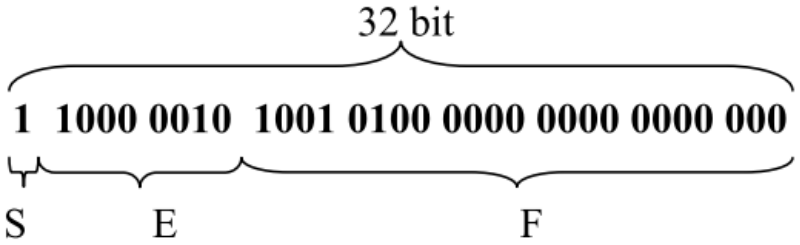
Ví dụ Biến đổi số thập phân -12.625_{10} sang số chấm động chuẩn IEEE 754 độ chính xác đơn (32 bit):

- Bước 1: Đổi số -12.625_{10} sang nhị phân:

  $-12.625_{10} = -1100.101_B$

- Bước 2: Chuẩn hoá: $-1100.101_B = -1.100101_B \times 2^3$ (Số 1.100101_2 dạng 1.f)
- Bước 3: Điền các bit vào các trường theo chuẩn:
  - Số âm: bit dấu S có giá trị 1
  - Phần mũ E với số thừa K=127, ta có: E-127=3

  Vậy: E = 3 + 127 = 130 (1000 0010B)
- Kết quả nhận được:



Biểu diễn số bằng dấu chấm động

- **<u>Một số ví dụ:</u>**

	Giá trị	Chuỗi bit thu được		
		Dấu	Số mũ	Giá trị
(a)	$+1.101 \times 2^{5}$	0	1000 0100	101 0000 0000 0000 0000 0000
(b)	-1.01011×2^{-126}	1	0000 0001	010 1100 0000 0000 0000 0000
(c)	$+1.0 \times 2^{127}$	0	1111 1110	000 0000 0000 0000 0000 0000
(d)	$+0$	0	0000 0000	000 0000 0000 0000 0000 0000
(e)	-0	1	0000 0000	000 0000 0000 0000 0000 0000
(f)	$+\infty$	0	1111 1111	000 0000 0000 0000 0000 0000
(g)	$+2^{-128}$	0	0000 0000	010 0000 0000 0000 0000 0000
(h)	+NaN	0	1111 1111	011 0111 0000 0000 0000 0000
(i)	$+2^{-128}$	0	011 0111 1111	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Biểu diễn số bằng dấu chấm động

Nhận xét:

- Trường bit dấu biểu diễn số âm (có giá trị 1) hoặc số dương (có giá trị 0).
- Trường số mũ được mã hóa theo phương pháp số thừa 127
- Trường giá trị biểu diễn số theo cơ số 2 có thực hiện dấu bit
- Có 1 quy ước biểu diễn giá trị vô cùng là số mũ là 1111 1111 và khoảng giá trị là các con số 0, bit dấu có thể là 0 hoặc 1
- Kết quả của phép toán $0/0$ là một số bất định. Khi đó, kết quả được trả về là NaN (Not a Number). Chuỗi quy ước có số mũ là 1111 1111, bit dấu là 0 hoặc 1, còn giá trị là một số khác 0

Biểu diễn các số thập phân

- Một vài ứng dụng, đặc biệt ứng dụng quản lý, bắt buộc các phép tính thập phân phải chính xác, không làm tròn số. Với một số bit cố định, ta không thể đổi một cách chính xác số nhị phân thành số thập phân và ngược lại.
- Vì vậy, khi cần phải dùng số thập phân, ta dùng cách biểu diễn số thập phân mã bằng nhị phân (BCD: Binary Coded Decimal) theo đó mỗi số thập phân được mã với 4 số nhị phân:

Số thập phân	d_3	d_2	d_1	d_0	Số thập phân	d_3	d_2	d_1	d_0
0	0	0	0	0	5	0	1	0	1
1	0	0	0	1	6	0	1	1	0
2	0	0	1	0	7	0	1	1	1
3	0	0	1	1	8	1	0	0	0
4	0	1	0	0	9	1	0	0	1

Biểu diễn các số thập phân

- Để biểu diễn số BCD có dấu, người ta thêm số 0 trước một số dương cần tính, ta có số âm của số BCD bằng cách lấy bù 10 số cần tính

  Ví dụ: biểu diễn số -079_{10} bằng số BCD

  - Trước hết ta lấy số bù 9 của số 079_{10} bằng cách: **999 - 079 = 920**
  - Cộng 1 vào số bù 9 ta được số bù 10: **920 + 1 = 921**
  - Biểu diễn số 921 dưới dạng số BCD, ta có: **$1001\ 0010\ 0001_{BCD}$**

vậy số -079_{10} trong biểu diễn số BCD là: **$1001\ 0010\ 0001_{BCD}$**

Chương 2: Biểu diễn thông tin trong máy tính

2.1 Hệ đếm

2.1.1 Hệ đếm cơ số 2

2.1.2 hệ đếm cơ số 8

2.1.3 Hệ đếm cơ số 16

2.1.4 Hệ đếm BCD

2.1.5 Chuyển đổi giữa các hệ đếm

2.2 Biểu diễn thông tin trong máy tính

2.2.1 Biểu diễn số nguyên

2.2.2 Biểu diễn số thực

2.2.3 Biểu diễn ký tự

Biểu diễn các ký tự

- Tuỳ theo các hệ thống khác nhau, có thể sử dụng các bảng mã khác nhau: ASCII, EBCDIC, UNICODE,....Các hệ thống trước đây thường dùng bảng mã ASCII (American Standard Codes for Information Interchange) để biểu diễn các chữ, số và một số dấu thường dùng mà ta gọi chung là ký tự
- Mỗi ký tự được biểu diễn bởi 7 bit trong một Byte. Hiện nay, một trong các bảng mã thông dụng được dùng là Unicode, trong bảng mã này, mỗi ký tự được mã hoá bởi 2 Byte

Bảng mã ASCII

00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	`	70	p
01	SOH	11	DC1	21	!	31	1	41	A	51	Q	61	a	71	q
02	STX	12	DC2	22	"	32	2	42	B	52	R	62	b	72	r
03	ETX	13	DC3	23	#	33	3	43	C	53	S	63	c	73	s
04	EOT	14	DC4	24	$	34	4	44	D	54	T	64	d	74	t
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e	75	u
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f	76	v
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g	77	w
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h	78	x
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i	79	y
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
0B	VT	1B	ESC	2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
0C	FF	1C	FS	2C	´	3C	<	4C	L	5C	\	6C	l	7C	|
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

NUL	Null	FF	Form feed	CAN	Cancel
SOH	Start of heading	CR	Carriage return	EM	End of medium
STX	Start of text	SO	Shift out	SUB	Substitute
ETX	End of text	SI	Shift in	ESC	Escape
EOT	End of transmission	DLE	Data link escape	FS	File separator
ENQ	Enquiry	DC1	Device control 1	GS	Group separator
ACK	Acknowledge	DC2	Device control 2	RS	Record separator
BEL	Bell	DC3	Device control 3	US	Unit separator
BS	Backspace	DC4	Device control 4	SP	Space
HT	Horizontal tab	NAK	Negative acknowledge	DEL	Delete
LF	Line feed	SYN	Synchronous idle		
VT	Vertical tab	ETB	End of transmission block		

EBCDIC

- Mã ASCII là nó chỉ có thể biểu diễn được 128 ký tự khác nhau bởi vì hạn chế các ký tự có trong bàn phím;
- Mã EBCDIC (Extended Binary Coded Decimal Interchange Code) được sử dụng để mở rộng thành mã 8 bit.
- Mã này được sử dụng rộng rãi trong các máy chủ mainframe của IBM.
- Từ các mã 7 bit của mã ASCII, ta thêm vào đó 1 bit 0 hoặc 1 để thu được mã EBCDIC

Bảng mã EBCDIC

- **EBCDIC is an 8-bit code.**

00	NUL	20	DS	40	SP	60	–	80		A0		C0	{	E0	\
01	SOH	21	SOS	41		61	/	81	a	A1	~	C1	A	E1	
02	STX	22	FS	42		62		82	b	A2	s	C2	B	E2	S
03	ETX	23		43		63		83	c	A3	t	C3	C	E3	T
04	PF	24	BYP	44		64		84	d	A4	u	C4	D	E4	U
05	HT	25	LF	45		65		85	e	A5	v	C5	E	E5	V
06	LC	26	ETB	46		66		86	f	A6	w	C6	F	E6	W
07	DEL	27	ESC	47		67		87	g	A7	x	C7	G	E7	X
08		28		48		68		88	h	A8	y	C8	H	E8	Y
09		29		49		69		89	i	A9	z	C9	I	E9	Z
0A	SMM	2A	SM	4A	¢	6A	‘	8A		AA		CA		EA	
0B	VT	2B	CU2	4B		6B	,	8B		AB		CB		EB	
0C	FF	2C		4C	<	6C	%	8C		AC		CC		EC	
0D	CR	2D	ENQ	4D	(	6D	_	8D		AD		CD		ED	
0E	SO	2E	ACK	4E	+	6E	>	8E		AE		CE		EE	
0F	SI	2F	BEL	4F	|	6F	?	8F		AF		CF		EF	
10	DLE	30		50	&	70		90		B0		D0	}	F0	0
11	DC1	31		51		71		91	j	B1		D1	J	F1	1
12	DC2	32	SYN	52		72		92	k	B2		D2	K	F2	2
13	TM	33		53		73		93	l	B3		D3	L	F3	3
14	RES	34	PN	54		74		94	m	B4		D4	M	F4	4
15	NL	35	RS	55		75		95	n	B5		D5	N	F5	5
16	BS	36	UC	56		76		96	o	B6		D6	O	F6	6
17	IL	37	EOT	57		77		97	p	B7		D7	P	F7	7
18	CAN	38		58		78		98	q	B8		D8	Q	F8	8
19	EM	39		59		79		99	r	B9		D9	R	F9	9
1A	CC	3A		5A	!	7A	:	9A		BA		DA		FA	|
1B	CU1	3B	CU3	5B	$	7B	#	9B		BB		DB		FB	
1C	IFS	3C	DC4	5C	·	7C	@	9C		BC		DC		FC	
1D	IGS	3D	NAK	5D	)	7D	'	9D		BD		DD		FD	
1E	IRS	3E		5E	;	7E	=	9E		BE		DE		FE	
1F	IUS	3F	SUB	5F	¬	7F	"	9F		BF		DF		FF	

STX	Start of text	RS	Reader Stop	DC
DLE	Data Link Escape	PF	Punch Off	DC
BS	Backspace	DS	Digit Select	DC
ACK	Acknowledge	PN	Punch On	CU
SOH	Start of Heading	SM	Set Mode	CU
ENQ	Enquiry	LC	Lower Case	CU
ESC	Escape	CC	Cursor Control	SY
BYP	Bypass	CR	Carriage Return	IFS
CAN	Cancel	EM	End of Medium	EC
RES	Restore	FF	Form Feed	ET
SI	Shift In	TM	Tape Mark	NA
SO	Shift Out	UC	Upper Case	SM
DEL	Delete	FS	Field Separator	SC
SUB	Substitute	HT	Horizontal Tab	IG
NL	New Line	VT	Vertical Tab	IR
LF	Line Feed	UC	Upper Case	IU

UNICODE

- Bảng mã ASCII và mã EBCDIC hỗ trợ các ký tự Latin được sử dụng trong máy tính, một tiêu chuẩn mới được xây dựng để hỗ trợ các ngôn ngữ trên toàn thế giới, đó là chuẩn Unicode
- Unicode là một chuẩn có tính chất mở. Trong phiên bản 2.0, đã có 38.885 ký tự được mã hóa bao gồm các ký tự của ngôn ngữ cơ bản của Châu Mỹ, Châu Âu, Trung đông, Châu Phi, Ấn độ, Châu Á Thái Bình Dương,...
- Chuẩn Unicode sử dụng mã 16 bit để biểu diễn ký tự, trong đó có một sự tương ứng 1-1 giữa mã 16 bit và ký tự biểu diễn
- Mặc dù Unicode hỗ trợ nhiều ký tự hơn ASCII hay EBCDIC nhưng đây cũng không chắc chắn là chuẩn cuối cùng mà chúng ta sử dụng. Thực tế, chuẩn Unicode 16 bit chỉ là một thành phần của chuẩn ký tự 32 bit ISO 10646 (UCS-4)

256 ký tự đầu tiên của Unicode 2.1

0000	NUL	0020	SP	0040	@	0060	`	0080	Ctrl	00A0	NBS	00C0	À	00E0	à
0001	SOH	0021	!	0041	A	0061	a	0081	Ctrl	00A1	¡	00C1	Á	00E1	á
0002	STX	0022	"	0042	B	0062	b	0082	Ctrl	00A2	¢	00C2	Â	00E2	â
0003	ETX	0023	#	0043	C	0063	c	0083	Ctrl	00A3	£	00C3	Ã	00E3	ã
0004	EOT	0024	$	0044	D	0064	d	0084	Ctrl	00A4	¤	00C4	Ä	00E4	ä
0005	ENQ	0025	%	0045	E	0065	e	0085	Ctrl	00A5	¥	00C5	Å	00E5	å
0006	ACK	0026	&	0046	F	0066	f	0086	Ctrl	00A6	¦	00C6	Æ	00E6	æ
0007	BEL	0027	'	0047	G	0067	g	0087	Ctrl	00A7	§	00C7	Ç	00E7	ç
0008	BS	0028	(	0048	H	0068	h	0088	Ctrl	00A8	¨	00C8	È	00E8	è
0009	HT	0029	)	0049	I	0069	i	0089	Ctrl	00A9	©	00C9	É	00E9	é
000A	LF	002A	*	004A	J	006A	j	008A	Ctrl	00AA	ª	00CA	Ê	00EA	ê
000B	VT	002B	+	004B	K	006B	k	008B	Ctrl	00AB	«	00CB	Ë	00EB	ë
000C	FF	002C	´	004C	L	006C	l	008C	Ctrl	00AC	¬	00CC	Ì	00EC	ì
000D	CR	002D	-	004D	M	006D	m	008D	Ctrl	00AD	–	00CD	Í	00ED	í
000E	SO	002E	.	004E	N	006E	n	008E	Ctrl	00AE	®	00CE	Î	00EE	î
000F	SI	002F	/	004F	O	006F	o	008F	Ctrl	00AF	¯	00CF	Ï	00EF	ï
0010	DLE	0030	0	0050	P	0070	p	0090	Ctrl	00B0	°	00D0	Đ	00F0	¶
0011	DC1	0031	1	0051	Q	0071	q	0091	Ctrl	00B1	±	00D1	Ñ	00F1	ñ
0012	DC2	0032	2	0052	R	0072	r	0092	Ctrl	00B2	²	00D2	Ò	00F2	ò
0013	DC3	0033	3	0053	S	0073	s	0093	Ctrl	00B3	³	00D3	Ó	00F3	ó
0014	DC4	0034	4	0054	T	0074	t	0094	Ctrl	00B4	´	00D4	Ô	00F4	ô
0015	NAK	0035	5	0055	U	0075	u	0095	Ctrl	00B5	µ	00D5	Õ	00F5	õ
0016	SYN	0036	6	0056	V	0076	v	0096	Ctrl	00B6	¶	00D6	Ö	00F6	ö
0017	ETB	0037	7	0057	W	0077	w	0097	Ctrl	00B7	·	00D7	×	00F7	÷
0018	CAN	0038	8	0058	X	0078	x	0098	Ctrl	00B8	¸	00D8	Ø	00F8	ø
0019	EM	0039	9	0059	Y	0079	y	0099	Ctrl	00B9	¹	00D9	Ù	00F9	ù
001A	SUB	003A	:	005A	Z	007A	z	009A	Ctrl	00BA	º	00DA	Ú	00FA	ú
001B	ESC	003B	;	005B	[	007B	{	009B	Ctrl	00BB	»	00DB	Û	00FB	û
001C	FS	003C	<	005C	\	007C	|	009C	Ctrl	00BC	1/4	00DC	Ü	00FC	ü
001D	GS	003D	=	005D	]	007D	}	009D	Ctrl	00BD	1/2	00DD	Ý	00FD	Þ
001E	RS	003E	>	005E	^	007E	~	009E	Ctrl	00BE	3/4	00DE	ý	00FE	þ
001F	US	003F	?	005F	_	007F	DEL	009F	Ctrl	00BF	¿	00DF	ß	00FF	ÿ

Các ví dụ áp dụng

Bài 1: Biểu diễn các số sau dạng dấu chấm động độ chính xác đơn và độ chính xác kép:

a) +138,125 -167,0625

b) +671,425 -234,250

c) +83,125 -125,425