Biểu diễn thông tin trong máy tính

Bởi:

Ngô Như Khoa

Hệ nhị phân (Binary)

Khái niệm

Hệ nhị phân hay hệ đếm cơ số 2 chỉ có hai con số 0 và 1. Đó là hệ đếm dựa theo vị trí. Giá trị của một số bất kỳ nào đó tuỳ thuộc vào vị trí của nó. Các vị trí có trọng số bằng bậc luỹ thừa của cơ số 2. Chấm cơ số được gọi là chấm nhị phân trong hệ đếm cơ số 2. Mỗi một con số nhị phân được gọi là một bit (BInary digiT). Bit ngoài cùng bên trái là bit có trọng số lớn nhất (MSB, Most Significant Bit) và bit ngoài cùng bên phải là bit có trọng số nhỏ nhất (LSB, Least Significant Bit) như dưới đây:

	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	
MSB	1	0	1	0	.	1	1	LSB
					Chấm nhị phân			

Số nhị phân $(1010.11)_2$ có thể biểu diễn thành:

$(1010.11)_2 = 1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 + 1*2^{-1} + 1*2^{-2} = (10.75)_{10}$.

dùng dấu ngoặc đơn và chỉ số dưới để ký hiệu cơ số của hệ đếm.

Biến đổi từ nhị phân sang thập phân

Biến đổi số nhị phân $(11001)_2$ thành số thập phân:

Trọng số vị trí: 2^4 2^3 2^2 2^1 2^0

Giá trị vị trí: 16 8 4 2 1

Số nhị phân: 1 1 0 0 1

Số thập phân: $1*2^4 + 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0 = (25)_{10}$

Biến đổi thập phân thành nhị phân

Để thực hiện việc đổi từ thập phân sang nhị phân, ta áp dụng phương pháp chia lặp như sau: lấy số thập phân chia cho cơ số để thu được một thương số và số dư. Số dư được ghi lại để làm một thành tố của số nhị phân. Sau đó, số thương lại được chia cho cơ số một lần nữa để có thương số thứ 2 và số dư thứ 2. Số dư thứ hai là con số nhị phân thứ hai. Quá trình tiếp diễn cho đến khi số thương bằng 0.

Biến đổi số thập phân $(29)_{10}$ thành nhị phân:

29/2 = 14 + 1(LSB)

14/2 = 7 + 0

7/2 = 3 + 1

3/2 = 1 + 1

1/2 = 0 + 1(MSB)

Vậy $(29)_{10} = (1101)_2$.

Đối với phần lẻ của các số thập phân, số lẻ được nhân với cơ số và số nhớ được ghi lại làm một số nhị phân. Trong quá trình biến đổi, số nhớ đầu chính là **bit** MSB và số nhớ cuối là **bit** LSB.

Biến đổi số thập phân $(0.625)_{10}$ thành nhị phân:

0.625*2 = 1.250. Số nhớ là 1, là **bit** MSB.

0.250*2 = 0.500. Số nhớ là 0

0.500*2 = 1.000. Số nhớ là 1, là **bit** LSB.

Vậy : $(0.625)_{10} = (0.101)_2$.

Hệ thập lục phân (Hexadecima)

Khái niệm

Các hệ máy tính hiện đại thường dùng một hệ đếm khác là hệ thập lục phân.

Hệ thập lục phân là hệ đếm dựa vào vị trí với cơ số là 16. Hệ này dùng các con số từ 0 đến 9 và các ký tự từ A đến F như trong bảng sau:

Hệ thập lục phân:

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

Biến đổi thập lục phân thành thập phân.

Các số thập lục phân có thể được biến đổi thành thập phân bằng cách tính tổng của các con số nhân với giá trị vị trí của nó.

Biến đổi các số a.$(5B)_{16}$. b. $(2AF)_{16}$ thành thập phân.

a. Số thập lục phân: 5 B

Trọng số vị trí: 16^1 16^0

Giá trị vị trí : 16 1

Số thập phân: 5*16 + B*1 = $(91)_{10}$.

b. Số thập lục phân: 2 A F

Trọng số vị trí: 16^2 16^1 16^0

Giá trị vị trí : 256 16 1

Số thập phân: 2*256 + A*16 + F*1 = $(687)_{10}$.

Biến đổi thập phân thành thập lục phân.

Để biến đổi các số thập phân thành thập lục phân, ta sử dụng phương pháp chia lặp, với cơ số 16.

Biến đổi $(1776)_{10}$ thành thập lục phân.

1776/16 = 111 + 0 (LSB).

111/16 = 6 + 15 hoặc F.

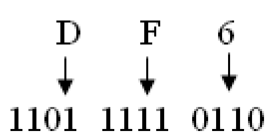
6/16 = 0 + 6 (MSB).

Số thập lục phân: $(6F0)_{16}$.

Biến đổi thập lục phân thành nhị phân

Các số thập lục phân rất dễ đổi thành nhị phân. Thực ra các số thập lục phân cũng chỉ là một cách biểu diễn các số nhị phân thuận lợi hơn mà thôi (bảng 2-1). Để đổi các số thập lục phân thành nhị phân, chỉ cần thay thế một cách đơn giản từng con số thập lục phân bằng bốn bit nhị phân tương đương của nó.

Đổi số thập lục $(DF6)_{16}$ thành nhị phân:



$(DF6)_{16} = (110111110110)_2$.

Biến đổi nhị phân thành thập lục phân

Để biến đổi một số nhị phân thành số thập lục phân tương đương thì chỉ cần gộp lại thành từng nhóm gồm 4 bit nhị phân, bắt đầu từ dấu chấm nhị phân.

Biến đổi số nhị phân $(1111101000010000)_2$ thành thập lục phân.

1111	1010	0001	0000
↓	↓	↓	↓
F	A	1	0

Số thập lục phân: $(FA10)_{16}$.

Hệ BCD (Binary Code decimal)

Giữa hệ thập phân và hệ nhị phân còn tồn tại một hệ lai: hệ BCD cho các số *hệ thập phân mã hoá bằng hệ nhị phân*, rất thích hợp cho các thiết bị đo có thêm phần hiển thị số ở đầu ra dùng các loại đèn hiện số khác nhau. Ở đây dùng bốn số hệ nhị phân (bốn bit) để mã hoá một số hệ thập phân có giá trị nằm trong khoảng từ 0..9. Như vậy ở đây ta không dùng hết các tổ hợp có thể có của 4 bit; vì tầm quan trọng của các số BCD nên các bộ vi xử lý thường có các lệnh thao tác với chúng.

$(35)_{10} = (00110101)_2$.

Bảng mã ASCII.(American Standard Code for Information Interchange)

Người ta đã xây dựng bộ mã để biểu diễn cho các ký tự cũng như các con số Và các ký hiệu đặc biệt khác. Các mã đó gọi là **bộ mã ký tự và số.** Bảng mã ASCII là mã 7 bit được dùng phổ biến trong các hệ máy tính hiện nay. Với mã 7 bit nên có $2^7 = 128$ tổ hợp mã. Mỗi ký tự (chữ hoa và chữ thường) cũng như các con số thập phân từ 0..9 và các ký hiệu đặc biệt khác đều được biểu diễn bằng một mã số như bảng 2-2.

Việc biến đổi thành ASCII và các mã ký tự số khác, tốt nhất là sử dụng mã tương đương trong bảng.

Đổi các ký tự BILL thành mã ASCII:

Ký tự	B	I	L	L
ASCI	1000010	1001001	1001100	1001100
HEXA	42	49	4C	4C

Mã ASCII.

Column bits($B_7B_6B_5$)

<table>
<tr><th colspan="5">Bits(row)</th><th></th><th></th><th>000</th><th>001</th><th>010</th><th>011</th><th>100</th><th>101</th><th>110</th><th>111</th></tr>
<tr><td>ROW</td><td>B4</td><td>B3</td><td>B2</td><td>B1</td><td></td><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>→</td><td></td><td>NUL</td><td>DLE</td><td>SP</td><td>0</td><td>@</td><td>P</td><td>\</td><td>p</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>→</td><td></td><td>SOH</td><td>DC1</td><td>!</td><td>1</td><td>A</td><td>Q</td><td>a</td><td>q</td></tr>
<tr><td>2</td><td>0</td><td>0</td><td>1</td><td>0</td><td>→</td><td></td><td>STX</td><td>DC2</td><td>“</td><td>2</td><td>B</td><td>R</td><td>b</td><td>r</td></tr>
<tr><td>3</td><td>0</td><td>0</td><td>1</td><td>1</td><td>→</td><td></td><td>ETX</td><td>DC3</td><td>#</td><td>3</td><td>C</td><td>S</td><td>c</td><td>s</td></tr>
<tr><td>4</td><td>0</td><td>1</td><td>0</td><td>0</td><td>→</td><td></td><td>EOT</td><td>DC4</td><td>$</td><td>4</td><td>D</td><td>T</td><td>d</td><td>t</td></tr>
<tr><td>5</td><td>0</td><td>1</td><td>0</td><td>1</td><td>→</td><td></td><td>ENQ</td><td>NAK</td><td>%</td><td>5</td><td>E</td><td>U</td><td>e</td><td>u</td></tr>
<tr><td>6</td><td>0</td><td>1</td><td>1</td><td>0</td><td>→</td><td></td><td>ACK</td><td>SYN</td><td>&</td><td>6</td><td>F</td><td>V</td><td>f</td><td>v</td></tr>
<tr><td>7</td><td>0</td><td>1</td><td>1</td><td>1</td><td>→</td><td></td><td>BEL</td><td>ETB</td><td>‘</td><td>7</td><td>G</td><td>W</td><td>g</td><td>w</td></tr>
<tr><td>8</td><td>1</td><td>0</td><td>0</td><td>0</td><td>→</td><td></td><td>BS</td><td>CAN</td><td>(</td><td>8</td><td>H</td><td>X</td><td>h</td><td>x</td></tr>
<tr><td>9</td><td>1</td><td>0</td><td>0</td><td>1</td><td>→</td><td></td><td>HT</td><td>EM</td><td>)</td><td>9</td><td>I</td><td>Y</td><td>i</td><td>y</td></tr>
<tr><td>A</td><td>1</td><td>0</td><td>1</td><td>0</td><td>→</td><td></td><td>LF</td><td>SUB</td><td>*</td><td>:</td><td>J</td><td>Z</td><td>j</td><td>z</td></tr>
<tr><td>B</td><td>1</td><td>0</td><td>1</td><td>1</td><td>→</td><td></td><td>VT</td><td>ESC</td><td>+</td><td>;</td><td>K</td><td>[</td><td>k</td><td>{</td></tr>
<tr><td>C</td><td>1</td><td>1</td><td>0</td><td>0</td><td>→</td><td></td><td>FF</td><td>FS</td><td>-</td><td><</td><td>L</td><td>\</td><td>l</td><td>|</td></tr>
<tr><td>D</td><td>1</td><td>1</td><td>0</td><td>1</td><td>→</td><td></td><td>CR</td><td>GS</td><td>,</td><td>=</td><td>M</td><td>]</td><td>m</td><td>}</td></tr>
<tr><td>E</td><td>1</td><td>1</td><td>1</td><td>0</td><td>→</td><td></td><td>SO</td><td>RS</td><td>.</td><td>></td><td>N</td><td>^</td><td>n</td><td>~</td></tr>
<tr><td>F</td><td>1</td><td>1</td><td>1</td><td>1</td><td>→</td><td></td><td>SI</td><td>US</td><td>/</td><td>?</td><td>O</td><td>_</td><td>o</td><td>DEL</td></tr>
</table>

Control characters:

NUL = Null; DLE = Data link escape; SOH = Start Of Heading;

DC1 = Device control 1; DC2 = Device control 2; DC3 = Device control 3.

DC4 = Device control 4; STX = Start of text; ETX = End of text;

EOT = End of transmission; ENQ = Enquiry; NAK = Negative acknowlege.

ACK = Acknowlege; SYN = Synidle; BEL = Bell.

ETB = End od transmission block; BS = Backspace; CAN = Cancel.

HT = Horizontal tab; EM = End of medium; LF = Line feed; SUB = Substitute.

VT = Vertical tab; ESC = Escape; FF = From feed; FS = File separator.

SO = Shift out; RS = Record separator; SI = Shift in; US = Unit separator.

Biểu diễn giá trị số trong máy tính

Biểu diễn số nguyên

a. Biểu diễn số nguyên không dấu:

Tất cả các số cũng như các mã ... trong máy vi tính đều được biểu diễn bằng các chữ số nhị phân. Để biểu diễn các số nguyên không dấu, người ta dùng n bit. Tương ứng với độ dài của số bit được sử dụng, ta có các khoảng giá trị xác định như sau:

Số bit	Khoảng giá trị
n bit:	0.. 2^n - 1
8 bit	0.. 255 Byte
16 bit	0.. 65535 Word

b. Biểu diễn số nguyên có dấu:

Người ta sử dụng bit cao nhất biểu diễn dấu; bit dấu có giá trị 0 tương ứng với số nguyên dương, bit dấu có giá trị 1 biểu diễn số âm. Như vậy khoảng giá trị số được biểu diễn sẽ được tính như sau:

Số bit	Khoảng giá trị:
n bit	2^{n-1}-1
8 bit	-128.. 127 Short integer
16 bit	-32768.. 32767 Integer
32 bit	-2^{31}.. 2^{31}-1 (-2147483648.. 2147483647) Long integer

Biểu diễn số thực(số có dấu chấm (phẩy) động)

Có hai cách biểu diễn số thực trong một hệ nhị phân: số có dấu chấm cố định (fīed point number) và số có dấu chấm động (floating point number). Cách thứ nhất được dùng trong những bộ VXL(micro processor) hay những bộ vi điều khiển (micro controller) cũ. Cách thứ 2 hay được dùng hiện nay có độ chính xác cao. Đối với cách biểu diễn số thực dấu chấm động có khả năng hiệu chỉnh theo giá trị của số thực. Cách biểu diễn chung cho mọi hệ đếm như sau:

R = m.B^e.

Trong đó m là phần định trị, trong hệ thập phân giá trị tuyệt đối của nó phải luôn nhỏ hơn 1. Số e là phần mũ và B là cơ số của hệ đếm.

Có hai chuẩn định dạng dấu chấm động quan trọng là: chuẩn MSBIN của Microsoft và chuẩn IEEE. Cả hai chuẩn này đều dùng hệ đếm nhị phân.

Thường dùng là theo tiêu chuẩn biểu diễn số thực của IEEE 754-1985(Institute of Electric & Electronic Engineers), là chuẩn được mọi hãng chấp nhận và được dùng trong bộ xử lý toán học của Intel. Bit dấu nằm tại vị trí cao nhất; kích thước phần mũ và khuôn dạng phần định trị thay đổi theo từng loại số thực.

Giá trị số thực IEEE được tính như sau:

$R = (-1)^{S} * (1+M_1*2^{-1} + ... +M_n*2^{-n})*2^{E\,7...E\,0\,-127}$.

Giá trị đầu tiên M0 luôn mặc định là 1

- Dùng 32 bit để biểu diễn số thực, được số thực ngắn: $-3,4.10^{38} < R < 3,4.10^{38}$

31	30 … 23	22 … 0
S	E7 - E0	Định trị (M1 - M23)

- Dùng 64 bit để biểu diễn số thực, được số thực dài: $-1,7.10^{308} < R < 1,7.10^{308}$

63	62 … 52	51 … 0
S	E10 - E0	Định trị (M1 - M52)

tính số thực:

0100 0010 1000 1100 1110 1001 1111 1100

Phần định trị: $2^{-4}+2^{-5}+2^{-8}+2^{-9}+2^{-10}+2^{-12}+2^{-15}+$
$+2^{-16}+2^{-17}+2^{-18}+2^{-19}+2^{-20}+2^{-21} = 0,1008906$.
Giá trị ngầm định là: 1,1008906.

Phần mũ: $2^8+2^2+2^0 = 133$
Giá trị thực (bit cao nhất là bit dấu): 133-128=6.

Dấu: 0 = số dương

Giá trị số thực là: $R = 1,1008906.2^6 = 70,457$.

Phương pháp đổi số thực sang số dấu phẩy động 32 bit:

- Đổi số thập phân thành số nhị phân.

- Biểu diễn số nhị phân dưới dạng ±1, xxxBy (B: cơ số 2).

- Bit cao nhất 31: lấy giá trị 0 với số dương, 1 với số âm.

- Phần mũ y đổi sang mã excess -127 của y, được xác định bằng cách: y + $(7F)_{16}$.

- Phần xxx là phần định trị, được đưa vào từ bit 22..0.

Biểu diễn số thực $(9,75)_{10}$ dưới dạng dấu phẩy động.

Ta đổi sang dạng nhị phân: $(9,75)_{10}$ = $(1001.11)_2$ = 1,00111B3.

Bit dấu: bit 31 = 0.

Mã excess - 127 của 3 là: 7F + 3 = $(82)_{16}$ = 82H = $(10000010)_2$. Được đưa vào các bit tiếp theo: từ bit 30 đến bit 23.

Bit 22 luôn mặc định là 0.

Cuối cùng số thực $(9,75)_{10}$ được biểu diễn dướiư dạng dấu phẩy động 32 bit như sau:

```
     0100 0001 0001 1100 0000 0000 0000 0000
bit |31|30        23|22                     0|
```