Реферат на тему:

**Внутрішнє подання даних стандартних типів**

**1. Біт, байт та інші**

*У комп'ютері числа зберiгаються та обробляються в двiйковiй системі числення*. Двійкова цифра 0 або 1 відображається станом елемента пам'яті, який вважається неподільним і називається ***бiтом***. Послідовність із 8 бітів називається ***байтом***. Байт своїми станами відображає 28=256 комбінацій із 0 та 1, а саме:

00000000

00000001



11111110

11111111

Множині цих комбінацій можна взаємно однозначно поставити у відповідність деякі множини значень: цілі числа від -128 до 127, або числа від 0 до 255, або пари 16-кових цифр, або символи від chr(0) до chr(255) чи якісь інші множини з 256 елементів.

У двох сусідніх байтах подаються 28 28=65536 комбінацій із 0 та 1. Їм взаємно однозначно ставляться у відповідність цілі числа від 0 до 65535, або числа від -32768 до 32767 чи інші множини з 65536 елементів.

Аналогічно чотири сусідні байти відображають (28)4=4294967296 комбінацій із 0 та 1, яким зiставляються числа від 0 до 4294967295, або числа від -2147483648 до 2147483647 чи інші множини з 4294967296 елементів.

Два байти утворюють одиницю пам'яті, яка називається ***словом***. Іноді таке слово називається ***напівсловом***, а словом – послідовність із чотирьох байтів.

Послідовність із 1024 байтів утворює одиницю виміру розмірів пам'яті комп'ютера. Цю одиницю позначають Kбайт, проте це "K" – латинська літера, що читається "кей" і позначає не тисячу, а 1024.

Послідовність із 1K Kбайтів, тобто 1048576 байтів, називається Mбайтом. Ці дві одиниці у світі програмістів і користувачів часто не зовсім точно називають відповідно "кілобайт" і "мегабайт", хоча це зовсім не тисяча і не мільйон байтів. До речі, 1Гбайт, хоча й читається "гігабайт", позначає не мільярд, а 1073741824 байти.

**2. Подання цілих чисел, символів та бульових значень**

*Бульовi значення* **false** та **true** подаються, як правило, в одному байтi комбінаціями відповідно 00000000 та 00000001.

*Символи* від chr(0) до chr(255) зображаються в одному байтi комбінаціями з нулів та одиниць відповідно від 00000000 до 11111111. Наприклад, символ chr(32), або ' ' (пропуск), зображається як 00100000, символ chr(48), або '0', – як 00110000 тощо.

Цілі числа подаються в комп'ютері, головним чином, у двох формах – ***беззнаковій*** та ***знаковій***. Далі ми *будемо ототожнювати числа з їх поданням*, усвідомлюючи, що з точки зору математики це не може бути правильним.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 7 … 0 | … | 7 … 0 | 7 … 0 |
| 8N-1 … |  | 15 … 8 | 7 … 0 |

*Беззнаковi* *числа* займають певну кількість *N* байтiв, яка задає дiапазон (множину) цих чисел від 0 до 28*N*-1. Найчастiше *N*=1, 2 або 4, і діапазони чисел – від 0 до відповідно 255, 65535 та 4294967295. Байти записуються від молодших до старших справа наліво та нумеруються від 0 до *N*-1. Біти всередині байтiв так само записуються від молодших до старших справа наліво й нумеруються від 0 до 7 (рис. 11.1). Усього в *N* байтах є 8*N* бітів, які нумеруються справа наліво від 0 до 8*N*-1. Біти з номерами 8*N*-1,  , 8*N*-8 утворюють старший байт (він ліворуч), а з номерами 7,  , 0 – молодший (праворуч). Комбінація бітів *x*8*N*-1,  , *x*0 зображає в двійковій системі число

*x*8*N*-1 28*N*-1+ *x*1 2+*x*0.

Наприклад, комбінація 00 00 задає число 0, комбінація 00 01 – "один", 00 10 – "два", 11 11 – число 28*N*-1.

|  |  |
| --- | --- |
| *Таблиця 11.1* | |
| **число** | **код** |
| 28*N*-1 - 1 | 01 11 |
| 28*N*-1 - 2 | 01 10 |
|  |  |
| 1 | 00 01 |
| 0 | 00 00 |
| -1 | 11 11 |
| -2 | 11 10 |
|  |  |
| -28*N*-1 + 1 | 10 01 |
| -28*N*-1 | 10 00 |

*Знаковi* *числа* займають ті самі *N* , тобто 1, 2 або 4 байти. Найстарший біт зображає знак числа: 0 – знак '+', 1 – знак '-'. *Додатні числа* подаються так само, як i беззнакові, лише за рахунок знакового біта дiапазон їх менший – від 0 до 28*N*-1-1. За *N*=1, 2 або 4 це відповідно 127, 32767 та 2147483647. Таке подання називається ***прямим кодом***. Наприклад, прямим кодом максимального цілого є 011 1.

*Від'ємні числа* подаються в коді, названому ***додатковим***. Для від'ємного числа *A* він позначається *D* (*A*) й утворюється так:

1) *за прямим кодом числа |A| заміною всіх 0 на 1 та всіх 1 на 0 будується* ***обернений код*** *R(A);*

2) *за R(A) як беззнаковим цілим числом обчислюється D(A)=R(A)+1.*

Очевидно, що *D*(*A*)=*R*(|A|-1). Наприклад, побудуємо двобайтовий додатковий код числа –144. Прямим двобайтовим кодом числа 144 буде

0000'0000'1001'0000

(апострофи записано для наочності), оберненим –

1111'1111'0110'1111.

До нього додається 1:

1111'1111'0110'1111

1

1111'1111'0111'0000,

і ми одержуємо додатковий код числа -144. Він є також оберненим кодом числа -143.

За додатковим кодом від'ємне число "відновлюється" у зворотному порядку:

1) *D(A) вважається беззнаковим цілим; обчислюється R(A)=D(A)-1;*

2) *код, обернений до R(A), є прямим кодом числа | A |.*

Той самий результат можна дістати, якщо

1) *побудувати код R(D(A)), обернений до D(A);*

2) *до R(D(A)) як до беззнакового додати 1.*

Відповідність знакових цілих чисел та їх кодів наведено в табл. 11.1. Як бачимо, від'ємних чисел на одне більше, ніж додатних.

Елемент *X* *довільного типу-переліку* подається як беззнакове цiле число ord(*X*).

**3. Принципи подання дійсних чисел**

*Дiйснi числа* в більшості комп'ютерів подаються в *N*=4, 6, 8 або 10 байтах, поділених на ***поля*** (послідовності бітів):

<знак><порядок><мантиса>.

Поле <знак> має довжину 1, а довжини двох інших позначимо *d* і *r* відповідно. Зрозуміло, що 1+*d*+*r*=8*N*. Нехай *s*, *e*, *m* – значення цих полів як беззнакових цілих. Вони подають:

*s* = 0 – знак '+', *s* = 1 – знак '-';

*e* – його ***порядок*** *t* = *e* - (2*d*-1-1);

*m* – ***мантису*** (дробову частину) *m*1 = *m* 2–*r*.

За значень *e*, відмінних від крайніх значень 0 та 2*d*-1, поля <знак><порядок><мантиса> задають число, що є значенням виразу

(-1)*s* (1+*m*1) 2*t* (11.2)

Оскільки 1 1+*m*1<2, то кажуть, що число подається в ***нормалiзованому виглядi***. Показник *t* називається ***справжнім порядком*** числа, а *e* – "***зсуненим***" (він на 2*d*-1-1 більше від справжнього). Отже, значення *e* від 1 до 2*d*-2 задають справжні порядки *t* від 1-(2*d*-1-1)=2-2*d*-1 до 2*d*-2-(2*d*-1-1)=2*d*-1-1.

Наприклад, нехай *d*=5, *r*=10, що задає двобайтове подання. Зсув порядку 25-1-1=24-1. Розглянемо зображення числа -12.375:

-12.375 = (-1100.011)2 = (-1.100011)2 23 ,

тобто *t*=3, *m*1=0.100011.Звідси *s*=1, *e*=3+(24-1)=18=(10010)2, *m*=1000110000, і число подається послідовністю бітів 1'10010'1000110000. Тут для наочності поля відокремлено апострофами.

Послідовність бітів 0'00001'0000000000 подає мінімальне додатне число, зображуване за *d*=5, *r*=10:

(1 + 0) 21-24+1 = 2-14.

Наступним числом, що подається як 0'00001'0000000001, буде

(1+2-10)  21-24+1=2-14+2-24.

Послідовність бітів 0'11110'11111111111 подає максимальне число

(1+(210-1) 2-10) 225-2-24+1 = (2-2-10) 215 =216 - 25 = 65504.

Попереднє перед ним число має подання 0'11110'11111111110 і є

(1+(210-2) 2-10) 225-2-24+1 = (2-2-9) 215 =216 - 26 = 65472.

Як бачимо, різниця між двома сусідніми числами міняється від 2-24 до 25=32.

За *e*=0 незалежно від *s* і *m* подається число 0. За *e*=2*d*-1 подання числа використовуєтьсся спеціальним чином, про що ми говорити не будемо (докладніше про це див., наприклад, [Григ]).

Зазначимо, що розташування й довжини полів у поданні дійсних чисел залежать від конкретного типу комп’ютера і можуть відрізнятися від указаних тут. Можливі й інші особливості.