Глава 4. Представление и обработка чисел в компьютере

4.1. Системы счисления

Система счисления *- это правило записи чисел с помощью заданного набора специальных знаков - цифр.*

Позиционными *- значение каждой цифры в изображении числа определяется ее положением (позицией) в ряду других цифр.*

$$272{,}12 = 2 \cdot 10^{2} + 7 \cdot 10^{1} + 2 \cdot 10^{0} + 1 \cdot 10^{-1} + 2 \cdot 10^{-2}.$$

Пусть *p* - основание системы счисления.

Тогда любое число Z, удовлетворяющее условию Z < p^k ($k \geq 0$, целое), может быть представлено в виде многочлена со степенями *p*:

$$Z_p = a_{k-1} \cdot p^{k-1} + a_{k-2} \cdot p^{k-2} + \ldots + a_1 \cdot p^1 + a_0 \cdot p^0 = \sum_{j=0}^{k-1} a_j \cdot p^i.$$

$$Z_p = (a_{k-1}\ a_{k-1} \ldots\ a_1\ a_0).$$

a_j - целые числа, удовлетворяющие условию:

$$0 \le a_j \le p - 1.$$

значение p = 2 - является минимальным для позиционных систем. Система счисления с основанием 2 называется *двоичной.*

4.2. Представление чисел в различных системах счисления

4.2.1. Перевод целых чисел из одной системы счисления в другую

$$Zp \rightarrow Zr \rightarrow Zq$$

r – основание, **для которого арифметические операции выполнить легко.**

$$r = 1 \text{ и } r = 10$$

т.е. перевод осуществляется через унарную или десятичную систему счисления.

Преобразование $Zp \to Z_1 \to Zq$

Пусть $Zq = 0$; из числа Zp вычтем 1 по правилам вычитания системы *p*,

$$Z_P := Z_P - 1$$

Добавим ее к *Z*q по правилам сложения системы *q*,

$$Zq := Zq + 1$$

Повторять пока не достигнем $Zp = 0$.

Пример 4.1 Выполнить $22_3 \to Z_6$.

Шаг	0	1	2	3	4	5	6	7	8
$Z_3 - 1$	**22**	21	20	12	11	10	2	1	0
$Z_6 + 1$	0	1	2	3	4	5	10	11	**12**

$$22_3 = 12_6.$$

Преобразование $Zp \rightarrow Z_{10} \rightarrow Zq$

$Z_{10} \rightarrow Zq$

- целочисленно разделить исходное число (Z_{10}) на основание (q) и найти остаток от деления;
- частное от деления снова целочисленно разделить на q; процедуру продолжать до тех пор, пока частное от деления не окажется меньше q;
- образовавшиеся остатки, записать в обратном порядке;

$Z_p \rightarrow Z_{10}$

Необходимо Zp представить в форме многочлена и выполнить все операции по правилам десятичной арифметики.

Пример 4.3

- Выполнить преобразование $443_5 \rightarrow Z_{10}$

$$443_5 = 4 \cdot 5^2 + 4 \cdot 5^1 + 3 \cdot 5^0 = 4 \cdot 25 + 4 \cdot 5 + 3 \cdot 1 = 123_{10}$$

4.2.2. Перевод дробных чисел из одной системы счисления в другую

Правильную дробь в исходной системе счисления *p* будем записывать в виде

$$0,Y\mathrm{p}$$

Дробь в системе *q*

$$0,Yq$$

Преобразование: $0,Yp \rightarrow 0,Yq.$

$$0,Yp \rightarrow 0,Y_{10} \rightarrow 0,Yq$$

Перевода $0,Y_{10} \rightarrow 0,Yq$

- умножить исходную дробь в 10-ной системе счисления на *q,* выделить целую часть
- для оставшейся дробной части операцию умножения повторять, пока в дробной части не окажется 0 или не будет достигнута желаемая точность конечного числа *(exact);*
- записать дробь в виде последовательности цифр после ноля с разделителем в порядке их появления.

Пример 4.4.

Выполнить преобразование

$$0{,}375_{10} \rightarrow 0{,}Y_2$$

$$\begin{array}{l}
0{,}375 \cdot 2 = 0{,}750 \\
0{,}75 \cdot 2 = 1{,}50 \\
0{,}5 \cdot 2 = 1{,}0
\end{array}$$

Таким образом, $0{,}375_{10} = 0{,}011_2$.

Перевод $0,Yp \to 0,Y_{10}$,

Сводится к вычислению значения многочлена в десятичной системе счисления.

$$0,011_2 = 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0,25 + 0,125 = 0,375_{10}.$$

Пример 4.5

Выполнить преобразование

$$5{,}3(3)_{10} \to X_3.$$

Ответ: $5{,}3(3)_{10} = 12{,}1_3$.

4.2.3. Понятие экономичности системы счисления

Число в системе счисления *p* с *k* разрядами, будет иметь наибольшее значение, если все цифры числа окажутся максимальными, т.е. равными *p* - 1.

$$(Z_p)^{\max} = \underbrace{<p-1> \ldots <p-1>}_{k \text{ цифр}} = p^k - 1.$$

Количество разрядов числа при переходе от одной системы счисления к другой в общем случае меняется.

Если $p = q^{\sigma}$ (σ - не обязательно целое), то $(Zp)max = p^k - 1 = q^{\sigma k} - 1$.

Количество разрядов числа в системах счисления p и q будут различаться в σ раз.

$$\sigma = \frac{\log p}{\log q}.$$

Сравним количество цифр в числе 99_{10} и его представлении в двоичной системе счисления: $99_{10} = 1100011_2$; т.е. двоичная запись требует 7 цифр вместо 2 в десятичной

$$\sigma = \ln(10)/\ln(2) = 3{,}322;$$

$$2*3{,}322 = 6{,}644 = 7.$$

Экономичность системы счисления - *количество чисел, которое можно записать в данной системе с помощью определенного количества цифр.*

Пусть имеется 12 цифр. Можно разбить их на 6 групп по 2 цифры («0» и «1») и получить шестиразрядное двоичное число.

Общее количество таких чисел, равно 26.

Основание системы счисления (p)	**2**	**3**	**4**	**6**	**12**
Разрядность числа (k)	6	4	3	2	1
Общее количество различных чисел (N)	$2^6 = 64$	$3^4 = 81$	$4^3 = 64$	$6^2 = 36$	$12^1 = 12$

наиболее экономичной оказывается троичная система счисления

Пусть имеется n знаков для записи чисел, p - основание системы счисления.

Количество разрядов числа $k = n/p$,

N - общее количество чисел которые могут быть составлены.

$$N = p^{\frac{n}{p}}.$$

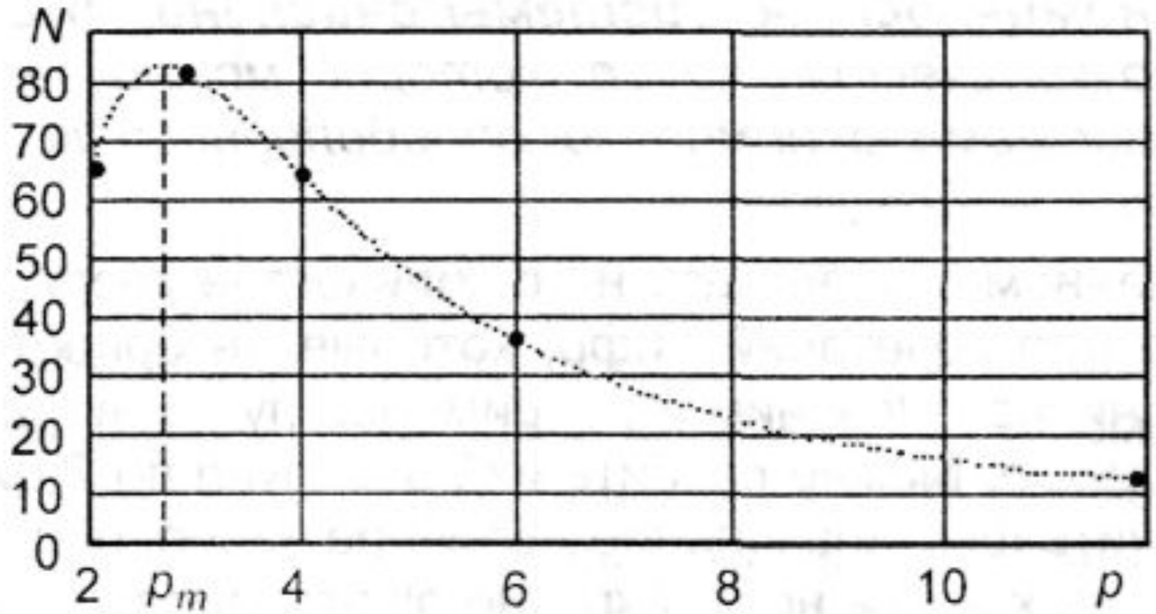


Рис. 4.1. Зависимость количества чисел от основания системы счисления при использовании 12-ти возможных цифр для записи чисел

$$\frac{dN}{dp} = -\frac{n}{p^2} p^{\frac{n}{p}} \ln p + \frac{n}{p} p^{\frac{n}{p}-1} = np^{\frac{n}{p}-2}(1-\ln p).$$

получаем $\ln p = 1$, или $p = e$,

$e = 2{,}71828..$

Ближайшее к e целое число 3

Троичная система счисления оказывается самой экономичной для представления чисел.

4.2.4. Преобразование нормализованных чисел

Вещественное число X может быть представлено в двух формах - естественной и нормализованной.

Число X_{10} называется нормализованным, если оно представлено в виде

$$X_{10} = \pm M_{10} \cdot 10^{\pm k}.$$

M_{10} - *мантисса* нормализованного числа.

Значения мантиссы лежат в интервале

$$0{,}1 \le M_{10} \le 1;$$

k - *порядком* нормализованного числа, это целое положительное десятичное число.

Примеры:

$-1234_{10} = -0{,}1234 \cdot 10^{4}$

$0{,}03456_{10} = 0{,}3456 \cdot 10^{-1}$.

!!!! Нормальное представления числа, мантисса лежит в интервале

$$1 \le M_{10} < 10,$$

например, $k_N = 2{,}38 \cdot 10^{-2}$.

Нормализованная форма числа в произвольной системе счисления *p:*

$$X_p = \pm M_p \cdot p^{\pm k_p}.$$

$$p^{-1} \le Mp < 1$$

Для p = 2:

$$X_2 = -101{,}01_2 = -0{,}10101_2 \cdot 2^{11_2}.$$

В компьютере все вещественные числа хранятся и обрабатываются в нормализованном двоичном представлении

при их вводе осуществляется перевод

$$X_{10} \to X_2,$$

при выводе обратный перевод

$$X_2 \to X_{10}.$$

Пример 4.8

$$16{,}5_{10} \to X_2.$$

Отдельно целую и дробную части

$$16_{10} = 10000_2,$$

$$0{,}5_{10} = 0{,}1_2$$

$$16{,}5_{10} = 10000{,}1_2$$

Нормализуем

$$10000{,}1_2 = (0{,}100001 \cdot 2^{101})_2.$$

Пример 4.9

$$(0{,}11 \cdot 2^{110})_2 \rightarrow X_{10}$$

$$0{,}11_2 = 0{,}75_{10}$$

$$(2^{110})_2 = (2^6)_{10} = 64$$

$$(0{,}11 \cdot 2^{110})_2 = 0{,}75 \cdot 64 = 48_{10}.$$

4.3. Кодирование чисел в компьютере и действия над ними

4.3.1. Кодирование и обработка в компьютере целых чисел без знака

Для записи числа выделяется фиксированное количество двоичных разрядов.

Память компьютера имеет байтовую структуру.

Размер одной адресуемой ячейки обычно составляет несколько байт.

Например, в компьютерах IBM ячейка памяти объединяет 2 байта (16 двоичных разрядов) - такая *комбинация связанных соседних ячеек, обрабатываемая совместно, называется машинным словом.*

Пусть количество разрядов *k* и *p* = 2 тогда,

$$(Z_2)max = 2^k - 1.$$

при *k* = 16

$$(Z_2)max = 2^{16} - 1 =$$
$$=1111111111111111_2 = 65535_{10}.$$

Число 65636 и более в компьютере не может существовать!!!

Минимальным целым числом в беззнаковом представлении, является

$$(Z_2)min = 0000000000000000_2 = 0_{10}.$$

В языке программирования PASCAL целые числа без знака, для записи которых отводится 2 байта, определены как тип **Word**.

Тип устанавливает способ кодирования числа, количество отводимых для записи ячеек памяти (т.е. разрядность числа), а также перечень допустимых операций при обработке.

Выход за границу **65535** возможен только путем увеличения количества разрядов для записи числа.

Необходим новый тип со своим Zmax;

Тип Longint с максимальным значением 2147483647_{10}, числа которого занимают 4 байта.

4.3.2. Кодирование и обработка в компьютере целых чисел со знаком

1. Прямой код.

Один (старший) разряд машинного слова отводится для записи знака числа;

знак «+» *- -1*

знак «-» - - 0

Остается 15 двоичных разрядов, что обеспечивает наибольшее значение числа

$$Zmax = 2^{15} - 1 = 32767_{10}.$$

Однако его применение усложняет порядок обработки чисел !!!

Пример 4.10

$1594_{10} + 17563_{10}$ при беззнаковой двоичной кодировке и 16-битном машинном слове.

Переносы:	1 11		1111
	0010 0110 1001 0100		9876
Регистр	+ 0011 0000 0011 1001	+	12345
переполнения	$\boxed{0}$0101 0110 1100 1101		22221.

Пример 4.11 – переполнение типа!

Найти сумму $65534_{10} + 3_{10}$

Переносы:	1111 1111 1111 11	1 1 1
	1111 1111 1111 1110	65534
	+ 0000 0000 0000 0011	+ 3
	$\boxed{1}$ 0000 0000 0000 0001	1.

2. Дополнительный код

Ось целых положительных чисел,

(0 ÷ 65535),

сместим положение «О» на середину,
числа, попадающие в первую половину

(0÷32767) -- положительные

(32768 ÷ 65535) -- отрицательные.

Пример

$1000000000000001_2 = 32769_{10}$ – код отрицательного числа.

$0000000000000001_2 = 1_{10}$ - код положительного числа.

Старший бит - у кодов положительных чисел «0», отрицательных - «1».

Способ построения дополнительного кода целых чисел

Дополнением (D) k-разрядного *целого числа Z в системе счисления* p *называется величина*

$$D(Z_P, k) = p^k - Z.$$

Дополнительный код (DK) двоичных целых чисел строится по следующим правилам:

- *для положительных $Z_2 \geq 0$ дополнительный код совпадает с самим числом*

$$DK = Z_2$$

- *для отрицательных $Z_2 < 0$ дополнительный код совпадает с дополнением модуля числа*

$$DK = D(|Z_2|, k).$$

Пример 4.14

Построить дополнительные двоичные коды чисел (a) 3_{10} и (b) -3_{10}.

(a) так как $Z > 0$,	*DK*	0000 0000 0000 0011
(b) так как $Z < 0$	(1) модуль числа	0000 0000 0000 0011
	(2) инверсия числа	1111 1111 1111 1100
	(3) *DK*	1111 1111 1111 1101

Проверка:

$$
\begin{array}{rl}
 & 1111\ 1111\ 1111\ 111 \quad \textit{переносы} \\
 & 0000\ 0000\ 0000\ 0011 \\
+ & \underline{1111\ 1111\ 1111\ 1101} \\
\boxed{1} & 0000\ 0000\ 0000\ 0000
\end{array}
$$

Прямой десятичный код	Прямой двоичный код с 16-ю разрядами	Дополнительный двоичный код с 16-ю разрядами
– 32 769	–	–
– 32 768	–	1000 0000 0000 0000
– 32 767	1111 1111 1111 1111	1000 0000 0000 0001
– 32 766	1111 1111 1111 1110	1000 0000 0000 0010
...	...	...
– 3	1000 0000 0000 0011	1111 1111 1111 1101
– 2	1000 0000 0000 0010	1111 1111 1111 1110
– 1	1000 0000 0000 0001	1111 1111 1111 1111
0	0000 0000 0000 0000	0000 0000 0000 0000
+ 1	0000 0000 0000 0001	0000 0000 0000 0001
+ 2	0000 0000 0000 0010	0000 0000 0000 0010
...	...	...
+ 32766	0111 1111 1111 1110	0111 1111 1111 1110
+ 32767	0111 1111 1111 1111	0111 1111 1111 1111
+ 32768	–	–

в 2-байтном машинном слове интервал чисел

[-32768; 32767] - типа Integer в языке PASCAL.

Перевод в дополнительный код происходит автоматически при вводе чисел; в таком виде числа хранятся в ОЗУ и затем участвуют в арифметических операциях.

Операция вычитания двух чисел как самостоятельная отсутствует - она заменяется сложением первого числа с дополнительным кодом второго.

Пример. Найти значение $(27\text{-}3)_{10}$ в двоичной кодировке.

$$
\begin{array}{lcl}
\textit{Переносы:} & & 1111\ 1111\ 1111\ 111 \\
DK(27) = & & 0000\ 0000\ 0001\ 1011 \\
DK(-3) = & + & \underline{1111\ 1111\ 1111\ 1101} \\
 & \boxed{1} & 0000\ 0000\ 0001\ 1000_2 = 24_{10}
\end{array}
$$

При выполнении вычитания отрицательного числа оно из дополнительного кода переводится в прямой, и вновь вместо вычитания производится сложение.

При выполнении операции умножения – перевод в прямой код.

4.3.3. Кодирование и обработка в компьютере вещественных чисел

Основная форма представления кодов вещественных чисел - двоичная нормализованная.

Записываются и хранятся в памяти все составляющие нормализованной формы (знак числа, мантисса, знак порядка и порядок).

Например, типа Real («вещественный») из языка PASCAL размещаются в 6 байтах, т. е. 48 двоичных разрядах.

Поскольку значение мантиссы лежит в интервале **0,1 < М < 1**, ноль в разряде целых и разделитель десятичных разрядов в представление не включаются, т. е. мантисса содержит только цифры дробной части.

Первая цифра мантиссы также не сохраняется (она для всех равна единице)

Количество разрядов числа при переходе от одной системы счисления к другой в общем случае меняется. Количество разрядов числа в системах счисления p и q будут различаться в σ раз.

$$\sigma = \frac{\log p}{\log q}.$$

Пример. Количество цифр в числе 99 и его представлении в двоичной системе счисления 99 = 1100011 ; т. е. двоичная запись требует 7 цифр вместо 2 в десятичной.

$$\sigma = \ln(10)/\ln(2) = 3{,}322$$

следовательно, количество цифр в десятичном представлении нужно умножить на 3,322 и округлить в большую сторону: 2*3,322 = 6,644 ≈7.

Пример 4.17.

Установить распределение разрядов двоичного представления числа типа Real, если для его записи отводится 48 бит, а максимальное значение десятичного порядка 38. Какова точность обработки таких чисел?

1) 2 бита расходуется на запись знака числа и порядка;

2) согласно (4.9) $k_2 = 3{,}322 k_{10}$; поскольку $k_{10} = 38$, очевидно, максимальный показатель порядка двоичного числа $k_2 = 3{,}322 \cdot 38 \approx 126_{10}$, что требует в двоичном представлении согласно формуле Хартли 7 бит;

3) под запись мантиссы отводится $48 - 2 - 7 = 39$ бит;

4) с учетом скрытого разряда точность обработки составит $\dfrac{39+1}{3{,}322} \approx$ ≈ 12 десятичных разрядов.

В процессе выполнения арифметических действий с нормализованными числами отдельно обрабатываются мантиссы и порядки.

Сложение нормализованных чисел

$$X_2 = M_2 p^{k_2} \text{ и } X_1 = M_1 p^{k_1}$$

1. находиться модуль разности Δk =abs(k 1- k 2)
2. сдвиг вправо на Δk разрядов мантиссы того числа, у которого k меньше.
3. сложение мантисс, порядок результата присваивается значение большего из имеющихся
4. результат нормализуют.

Пример 4.18.

Найти сумму $X_1 = 0{,}87654 \cdot 10^1$ и $X_2 = 0,94567 \cdot 10^2$, если для записи мантиссы отводится 5 разрядов.

Согласно алгоритму $\Delta k = 1$ и $k_1 < k_2$. Следовательно, $k = k_2 = 2$, а мантисса числа X_1 должна быть сдвинута на 1 разряд вправо (при этом из-за ограниченности разрядно сетки пропадет цифра 4). Новая мантисса получается суммированием $M = 0{,}94567 + 0{,}08765 = 1{,}03332$; поскольку она выходит за допустимый интервал представления мантисс, необходимо его нормализовать: $M' = 0{,}10333$ (при этом теряется цифра 2 в младшем разряде); $k' = k + 1 = 3$. Окончательно получаем: $X = 0{,}10333 \cdot 10^3$. Точный результат суммирования оказался бы 103,3324, т. е. последние два десятичных разряда оказываются потерянными.

Вычитание нормализованных чисел, как и чисел целых, не является самостоятельной операцией и сводится к сложению с дополнительным кодом числа.

Умножение нормализованных чисел $X_1 \otimes X_2$ выполняется в соответствии с правилами: если по-прежнему $X_2 = M_2 p^{k_2}$ и $X_1 = M_1 p^{k_1}$, то, очевидно, мантисса произведения $M = M_1 M_2$, а порядок $k = k_1 + k_2$; при необходимости полученное число нормализуется.

1. В компьютерах арифметические устройства выполняют действия не с самими двоичными числами по правилам двоичной арифметики, а с их двоичными кодами (представлениями) по правилам арифметики двоичных кодов.

2. Причиной отличий правил арифметики двоичных кодов от правил обычной арифметики является ограниченность разрядной сетки, применяемой для записи чисел в компьютере. По этой же причине отличаются понятия «*ноль*» и «*машинный ноль*», «*бесконечность*» — «*максимальное число*», а также становится возможной ситуация переполнения, что требует ее постоянного отслеживания.

3. Применение при вычислениях формы представления чисел с плавающей запятой обеспечивает единообразие при их записи и обработке, и, что важно, в результате автоматического масштабирования числа на каждом этапе его обработки сокращается погрешность вычислений.

4. Различие правил обработки целых и нормализованных чисел приводит к необходимости точного описания типов переменных перед их использованием в программах. Вторая причина описания типов состоит в оптимизации расходования памяти компьютера, поскольку числа разных типов требуют для хранения различных ресурсов памяти.

Выполнить вычитание двоичных нормализованных чисел $0.10101 \cdot 2^{10}$ и $0.11101 \cdot 2^{1}$.

Разность порядков уменьшаемого и вычитаемого здесь равна 1, поэтому перед вычитанием мантисса второго числа сдвигается на один разряд вправо:

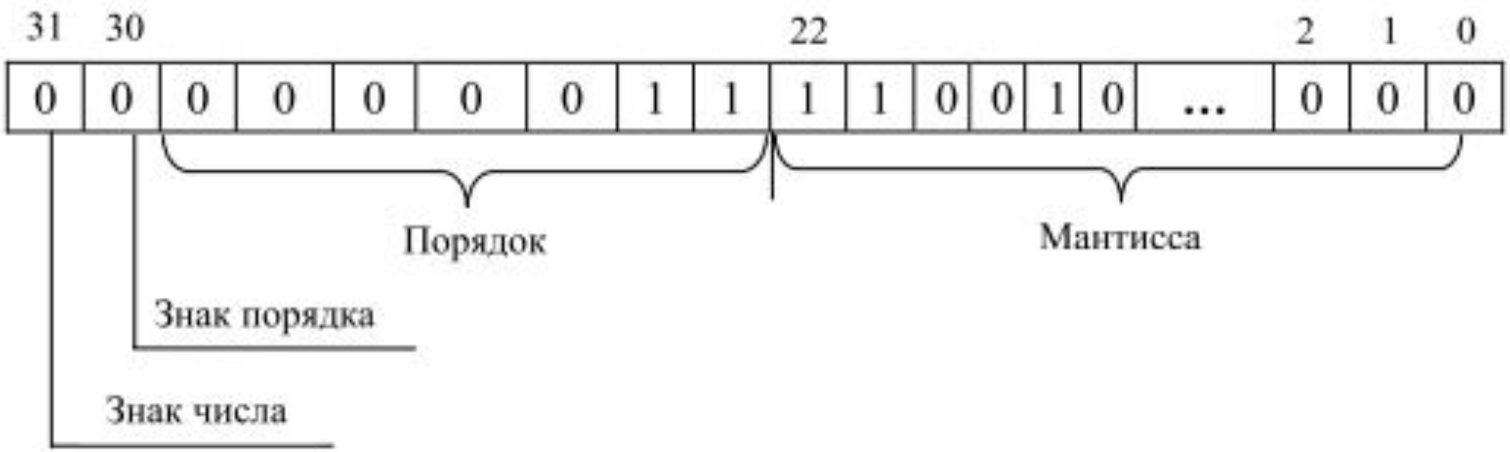
$$
\begin{array}{r}
- \begin{array}{l} 0.10101 \; * 2^{10} \\ 0.011101 * 2^{10} \end{array} \\ \hline
0.001101 * 2^{10}
\end{array}
$$

Выполнить умножение двоичных нормализованных чисел:

$$(0.11101 \cdot 2^{101}) \cdot (0.1001 \cdot 2^{11})$$

$$(0.11101 \cdot 0.1001) \cdot 2^{(101+11)} = 0.100000101 \cdot 2^{1000}.$$

Число $6.25_{10} = 110.01_2 = 0.11001 \cdot 2^{11}$:



Число $-0.125_{10} = -0.001_2 = -0.1 \cdot 2^{-10}$ (отрицательный порядок записан в дополнительном коде):