Арифметические основы компьютерной схемотехники

Принципы построения системы счисления

Числовая информация в компьютерах характеризуется:

- системой счисления (двоичная, десятичная и др.);
- видом числа (числа: вещественные, комплексные и др.);
- типом числа (целое, дробное, смешанное);
- формой представления числа (место запятой):
  - с фиксированной,
  - с плавающей запятой;
- форматом числа и его разрядностью;
- диапазоном и точностью представления чисел;
- способом кодирования отрицательных чисел:
  - прямой код;
  - обратный код;
  - дополнительный кода;
- алгоритмами выполнения арифметических операций.

Системой счисления называется совокупность цифр и правил для записи чисел.

Запись числа в некоторой системе счисления называется его **кодом.**

Все системы счисления делятся на **позиционные** и **непозиционные**.

В непозиционных системах счисления значение каждой цифры не зависит от ее позиции (положения) в общем ряду чисел.
Самой известной непозиционной системой счисления является **римская**, в которой используются семь знаков – **I**, **V**, **X**, **L**, **C**, **D**, **M**, соответствующих таким значениям:

I	**V**	**X**	**L**	**C**	**D**	**M**
1	5	10	50	100	500	1000

Например: **III** – 3; **LIX** – 59; **DLV** – 555; **MMXVIII** – 2018

Недостатком непозиционной системы является отсутствие нуля и формальных правил записи чисел и соответственно арифметических действий с ними.

Для записи чисел в позиционной системе счисления используют определенное количество графических знаков (цифр и букв), которые отличаются один от другого. Число таких знаков *q* называют **основанием позиционной системы счисления.**
В компьютерах используют позиционные системы с разным основанием.

Система счисления с основанием **два** (цифры 0 и 1) называется **двоичной,** система счисления с основанием **три** (цифры 0, 1, 2) – **троичной** и т.д.

В системах счисления с основанием

меньше десяти (<10) используют **десятичные цифры**,

больше десяти добавляют буквы латинского алфавита – ***A, B, C, D, E, F****.*

Далее в обозначениях в случае необходимости пишут десятичный индекс, равный применяемому основанию системы счисления

Основание q	Система счисления	Знаки
2	Двоичная	0, 1
3	Троичная	0, 1, 2
5	Пятеричная	0, 1, 2, 3, 4
8	Восьмеричная	0, 1, 2, 3, 4, 5, 6, 7
10	Десятичная	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
16	Шестнадцатеричная	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *A*, *B*, *C*, *D*, *E*, *F*

Таблица представления некоторого числового диапазона в различных системах счисления:

Системы счисления				
Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная	Двоично-десятичная
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101
16	10000	20	10	0001 0110

Система счисления должна обеспечивать:

- возможность представления любого числа в заданном диапазоне;
- однозначность, сжатость записи числа и простоту выполнения арифметических операций;
- достижение высокого быстродействия машины в процессе обработки информации.

Число в позиционной системе можно представить полиномом:

$$A_q = a_k \cdot q^k + a_{k-1} \cdot q^{k-1} + \ldots + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \ldots + a_{-m} \cdot q^{-m} = \sum_{i=-m}^{k} a_i \cdot q^i ,$$

где q – основание системы счисления; q^i – вес позиции; a Є {0, 1, …, (q – 1)} – цифры в позициях числа; *0, 1, k* – номера разрядов целой части числа; *-1, -2, …, -m* – номера разрядов дробной части числа.

Позиционные системы с одинаковым основанием в каждом разряде называют **однородными.**
Поскольку на значение q нет никаких ограничений, то теоретически возможно бесконечное множество позиционных систем счисления.

Достоинством двоичной системы является:

- простота выполнения арифметических операций;
- наличие надежных микроэлектронных схем с двумя устойчивыми состояниями (триггеров), предназначенных для хранения значений двоичного разряда – цифр 0 или 1.

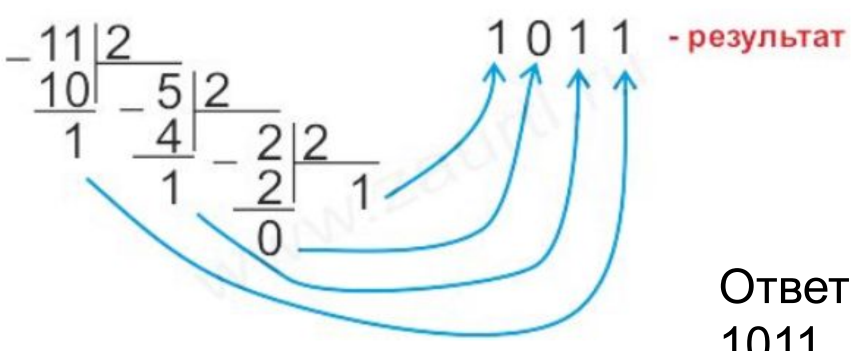
Способы перевода чисел из одной системы счисления в другую

Перевод чисел из одной позиционной системы счисления в другую:

1. Перевод целых чисел

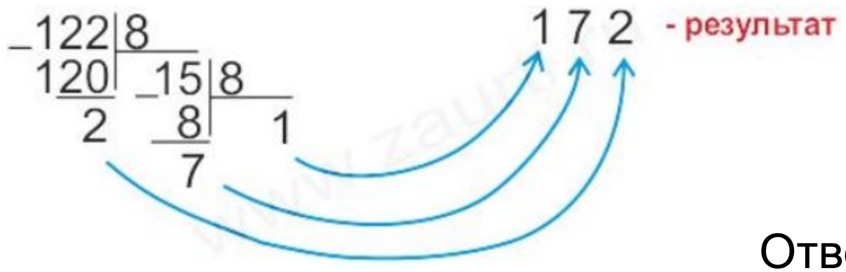
Чтобы перевести целое число из одной системы счисления с основанием q_1 в другую с основанием q_2 необходимо **последовательно делить** это число и получаемые частные на основание q_2 новой системы до тех пор, пока не получится частное меньше основания q_2. Последнее частное – старшая цифра числа в новой системе счисления с основанием q_2, а следующие за ней цифры – это остатки от деления, записываемые в последовательности, обратной их получению. Арифметические действия надо выполнять в той системе счисления, в которой записано переводимое число.

Пример 1. Перевести число 11_{10} в двоичную систему счисления.



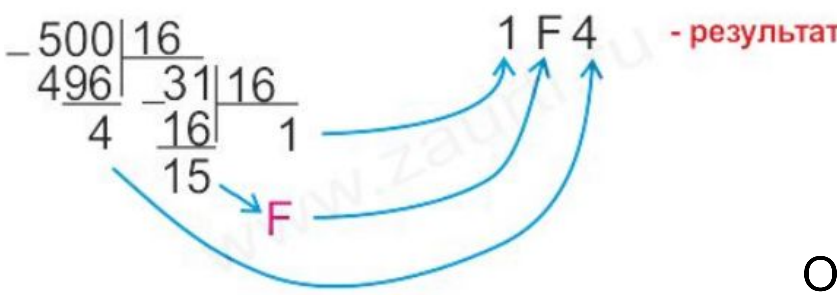
Ответ: $11_{10} = 1011_2$

Пример 2. Перевести число 122_{10} в восьмеричную систему счисления.



Ответ: $122_{10} = 172_8$.

Пример 3. Перевести число 500_{10} в шестнадцатеричную систему счисления.



Ответ: $500_{10} = 1F4_{16}$.

2. Перевод правильных дробей.

Чтобы перевести правильную дробь из системы счисления с основанием q_1 в систему с основанием q_2, необходимо **последовательно умножать** исходную дробь и дробные части получающихся произведений на основание новой системы счисления q_2. Правильная дробь числа в новой системе счисления с основанием q_2 формируется в виде целых частей получающихся произведений, начиная с первого.

Если при переводе получается дробь в виде бесконечного или расходящегося ряда, процесс можно закончить при достижении необходимой точности.

При переводе смешанных чисел, необходимо в новую систему перевести отдельно целую и отдельно дробную части по правилам перевода целых чисел и правильных дробей, а затем оба результата объединить в одно смешанное число в новой системе счисления.

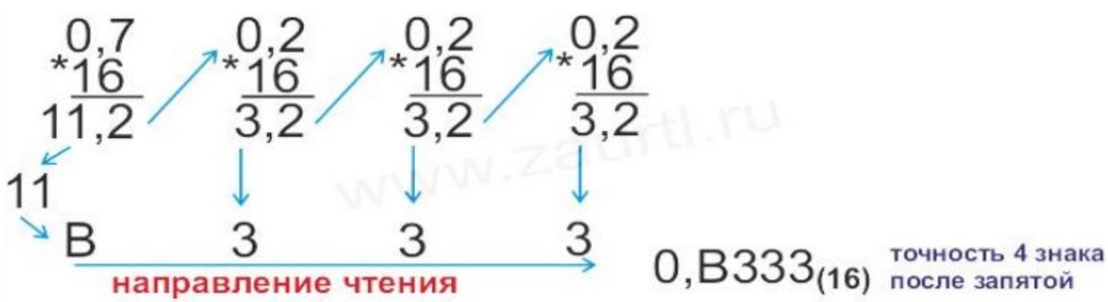
Пример 1. Перевести число $0{,}625_{10}$ в двоичную систему счисления.

Ответ: $0{,}625_{10} = 0{,}101_2$.

Пример 2. Перевести число $0{,}6_{10}$ в восьмеричную систему счисления.

Ответ: $0{,}6_{10} = 0{,}463_{8}$.

Пример 3. Перевести число $0{,}7_{10}$ в шестнадцатеричную систему счисления.



Ответ: $0{,}7_{10} = 0{,}B333_{16}$.

Для перевода числа X-ичной системы в десятичную необходимо использовать следующую формулу разложения:

$$a_n a_{n-1} \ldots a_1 a_0 = a_n X^n + a_{n-1} X^{n-1} + \ldots + a_1 X^1 + a_0 X^0$$

Пример 1. Перевести число $101{,}11_2$ в десятичную систему счисления.

$$\overset{2}{1}\overset{1}{0}\overset{0}{1},\overset{-1}{1}\,\overset{-2}{1}_{(2)} \to (10) = 1*2^2+1*2^1+0*2^0+1*2^{-1}+1*2^{-2}=5{,}75_{(10)}$$

Ответ: $101{,}11_2 = 5{,}75_{10}$.

Пример 2. Перевести число $57{,}24_8$ в десятичную систему счисления.

$$\overset{1}{5}\overset{0}{7},\overset{-1}{2}\,\overset{-2}{4}_{(8)} \to (10) = 5*8^1+7*8^0+2*8^{-1}+4*8^{-2}=47{,}3125_{(10)}$$

Ответ: $57{,}24_8 = 47{,}3125_{10}$

Пример 3. Перевести число $7A{,}84_{16}$ в десятичную систему счисления.

$$\overset{1}{7}\overset{0}{A},\overset{-1}{8}\,\overset{-2}{4}_{(16)} \to (10) = 7*16^1+10*16^0+8*16^{-1}+4*16^{-2}=122{,}515625_{(10)}$$

Ответ: $7A{,}84_{16} = 122{,}515625_{10}$

Перевод восьмеричных и шестнадцатеричных чисел в двоичную систему счисления и обратно.

Для перевода числа из восьмеричной системы счисления в двоичную необходимо каждую цифру этого числа записать трехразрядным двоичным числом (триадой).

Пример: записать число $16,24_8$ в двоичной системе счисления.

Ответ: $16,24_8 = 1110,0101_2$.

<u>Примечание</u>: *незначащие нули слева для целых чисел и справа для дробей не записываются.*

Для обратного перевода двоичного числа в восьмеричную систему счисления, необходимо исходное число разбить на триады влево и вправо от запятой и представить каждую группу цифрой в восьмеричной системе счисления. Крайние неполные триады дополняют нулями.

Пример: записать число $1110,0101_2$ в восьмеричной системе счисления.

Ответ: $1110,0101_2 = 16,24_8$.

Для перевода числа из шестнадцатеричной системы счисления в двоичную необходимо каждую цифру этого числа записать четырехразрядным двоичным числом (тетрадой).

Пример: записать число $7A,5E_{16}$ в двоичной системе счисления.

Ответ: $7A,5E_{16}$ = $1111010,0111111_2$.

Примечание: незначащие нули слева для целых чисел и справа для дробей не записываются.

Для обратного перевода двоичного числа в шестнадцатеричную систему счисления, необходимо исходное число разбить на тетрады влево и вправо от запятой и представить каждую группу цифрой в шестнадцатеричной системе счисления. Крайние неполные триады дополняют нулями.

Пример: записать число $1111010,0111111_2$ в шестнадцатеричной системе счисления.

Ответ: $1111010,0111111_2$ = $7A,5E_{16}$.

Понятия двоичной логики

- **Код** – двоичное число,
- **Кодирование** – метод представления двоичных чисел;
- **Разрядность кода** – количество двоичных разрядов кода
    - кило – 2^{10} = 1 024,
    - Мега – 2^{20} = 1 048 576;
    - Гига – 2^{30} = 1 073 741 824;
- **Бит** – один разряд двоичного числа (от англ. *binary digit*);
- **Байт** – восемь двоичных разрядов (битов) – принимает 2^8 значений: от 0 до 255;
- **Тетрада** (полубайт, ниббл) – четыре двоичных разряда, половина байта – принимает 2^4 значений: от 0 до 15;
- **Слово** – код, состоящий из нескольких байтов – чаще всего:
    - 2 байта – 16 разрядов,
    - 4 байта – 32 разряда,
    - 8 байт – 64 разряда;

Арифметические действия над двоичными числами

Беззнаковые двоичные коды

В этих кодах **каждый двоичный разряд** представляет собой **степень цифры 2**

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
1	1	1	1	1	1	1	1	Максимально возможное число
			…					
0	0	0	0	0	0	0	0	Минимально возможное число

Рис. 1. Диапазон представлений беззнаковых двоичных кодов

Минимально возможное число, которое можно записать таким двоичным кодом, равно **0**.

Максимально возможное число, которое можно записать таким двоичным кодом, можно определить как $\boldsymbol{M = 2^{n-1}}$.

Прямые целые знаковые коды

Старший разряд в слове используется для представления знака числа.

В прямом знаковом коде **нулем** обозначается знак '**+**', а **единицей** - знак '**–**'.

В результате введения знакового разряда диапазон чисел смещается в сторону отрицательных чисел:

Знак числа	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	1	1	1	1	1	1	1	Максимально возможное число
			…					
0	0	0	0	1	0	1	0	+10
			…					
0	0	0	0	0	0	0	0	+0
1	0	0	0	0	0	0	0	-0
			…					
1	0	0	0	1	0	1	0	-10
			…					
1	1	1	1	1	1	1	1	Минимально возможное число

Рис. 2. Диапазон представлений прямых целых знаковых двоичных кодов

Недостатками такого кода является то, что знаковый разряд и цифровые разряды приходится обрабатывать раздельно и имеет место положительный и отрицательный **0**, хотя 0 всегда считается строго положительным числом.

Дополняющие формы представление двоичных чисел

Для того, **чтобы иметь малые затраты на аппаратурную часть** (*hardware*) компьютера, были предприняты усилия по **сведению к одному алгоритму** процедуры **вычитания и сложения**. Этого можно добиться, если применять **двоичные цифры в** их **дополняющей форме.**

Различают:

а) **единичное дополнение** (*one's complement*) и

(обратный код, поразрядное дополнение)

$$1001100111 \rightarrow 0110011000$$

б) **двойное дополнение** (*two's complement*) (точное дополнение).

$$11011101011 \rightarrow 00100010100 + 1 \rightarrow 00100010101$$

Знаковые обратные двоичные коды

Обратные двоичные коды отличаются от прямых только тем, что отрицательные числа в них получаются путём инвертированием всех разрядов числа. При этом знаковый и цифровые разряды не различаются.

Знак числа	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	1	1	1	1	1	1	1	Максимально возможное число
			…					
0	0	0	0	1	0	1	0	+10
			…					
0	0	0	0	0	0	0	0	+0
1	1	1	1	1	1	1	1	–0
			…					
1	1	1	1	0	1	0	1	–10
			…					
1	0	0	0	0	0	0	0	Минимально возможное число

Рис. 3. Диапазон представлений знаковых обратных двоичных кодов

При работе с обратными кодами требуется специальный алгоритм распознавания знака, вычисления абсолютного значения числа, восстановления знака результата числа.

Кроме того, **в прямом и обратном коде для запоминания числа 0 используется два кода**, тогда как известно, что **число 0 положительное** и **отрицательным не может быть никогда**!

Знаковые дополнительные двоичные коды

Дополнительные двоичные коды позволяют непосредственно суммировать положительные и отрицательные числа не анализируя знаковый разряд и при этом получать правильный результат.

Это становится возможным благодаря тому, что дополнительные числа являются естественным кольцом чисел

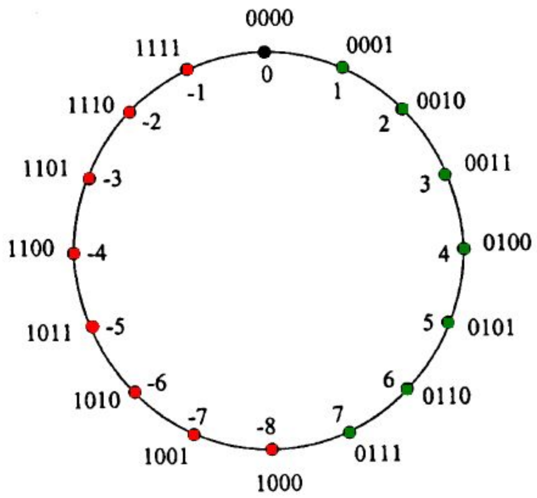


Рис. 4. Представление 4-х битовых двоичных чисел в виде круговой диаграммы

Наибольшее и наименьшее представляемые числа можно выразить как:

$$A_{max} = 2^{n-1} = 7_{10}$$
$$A_{min} = -\,2^{n-1} = -\,8_{10}$$

Важно! При точном дополнении имеется только один ноль: $\pm\, 0_{10} = 0000_2$.

При поразрядном дополнении – два нуля: $+0_{10} = 0000_2$; $-\,0_{10} = 1111_2$

Переполнение (*overflow*) **при сложении чисел в дополнительном коде**

При **сложении двух положительных двоичных *n*-разрядных чисел** может появиться число из ***n* + 1** разрядов. **Признак**ом появления такого числа служит **перенос из старшего разряда**. В дополнительном коде этот разряд резервируется для знака.

Сложение двух отрицательных чисел вызывает систематическое переполнение данного разряда и соответственно перенос, поскольку у каждого из подобных чисел указанный разряд равен 1.

При обработке чисел с произвольным знаком появление переноса в разряде переполнения вовсе не означает, что переполнение действительно имело место.

Переполнение числового диапазона может происходить только в двух случаях:

а) когда **суммируются два положительных числа**

Если складываются два положительных числа, результат также должен быть положительным.

Если сумма вышла за пределы разрядной сетки, в знаковом разряде появляется перенос, то есть результат оказывается отрицательным.

$$\begin{array}{rr} & 0101 \quad 5_{10} \\ + & 0101 \quad 5_{10} \\ \hline = & 1010 \quad -6_{10} \end{array}$$

результат является неправильным

б) когда **суммируются два отрицательных числа**.

$$
\begin{array}{rrr}
+ & 1011 & -5_{10} \\
 & 1011 & -5_{10} \\
\hline
= & (1)\,0110 & 6_{10}
\end{array}
$$

результат является неправильным

Отрицательное переполнение при сложении двух отрицательных чисел обнаруживается по положительному результату.

При сложении положительного и отрицательного чисел переполнение невозможно, так как модуль результата будет заведомо меньше модулей слагаемых.

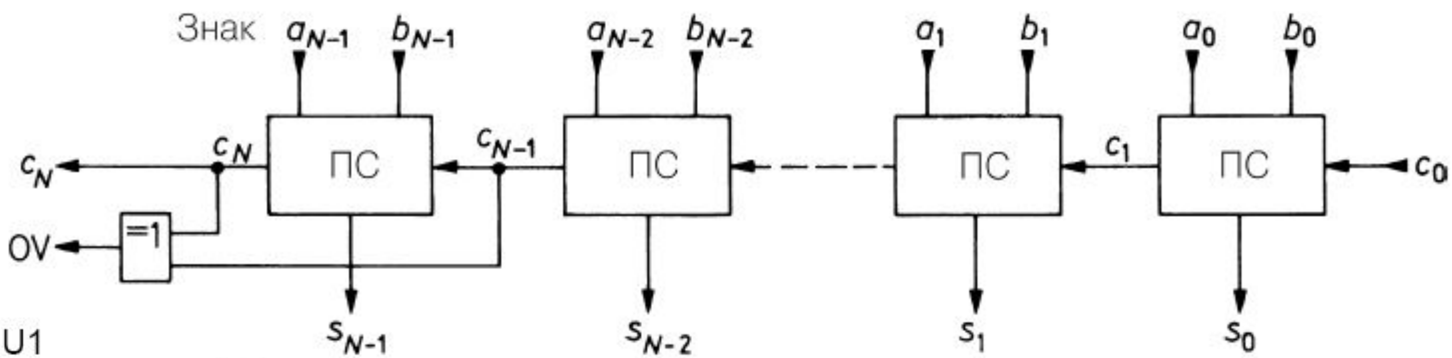
Таблица 1. Перенос переполнения ***c*** при сложении в случае *n*-разрядного представления на основе точного дополнения

	Правильный результат	Перенос переполнения
$A+B$	$c_n=0,\ c_{n-1}=0$	$c_n=0,\ c_{n-1}=1$
$A-B$	$c_n=c_{n-1}$	невозможен
$-A-B$	$c_n=1,\ c_{n-1}=1$	$c_n=1,\ c_{n-1}=0$

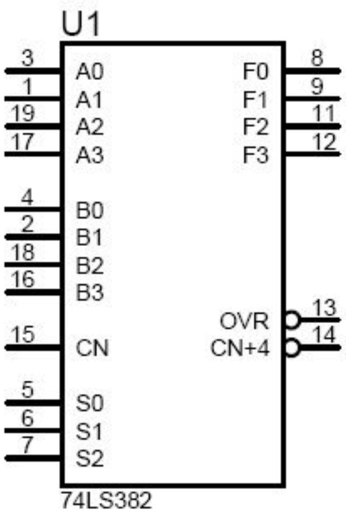
Правильный результат имеет место тогда, когда $c_n = c_{n-1}$, неправильный результат – когда $c_n \neq c_{n-1}$

Переполнение легко найти, сравнивая перенос c_{n-1} в знаковый разряд с переносом c_n из знакового разряда.

Переполнение имеет место именно тогда, когда эти **переносы неодинаковы**. Выявить данный случай возможно воспользовавшись функцией **Исключающее ИЛИ**.
Например, такой выход имеется у 4-х разрядного арифметическо-логического устройства (АЛУ) SN74LS382.



а)



б)

Рис. 5. а) структурная схема АЛУ SN74LS382 и б) условно графическое обозначение

Представление дробных чисел в двоичном коде с фиксированной запятой

Кроме целых чисел часто требуется работать с дробными числами.

Дробные коды как и в случае целых чисел могут быть **беззнаковые** и **знаковые**. Для записи знаковых чисел могут быть использованы прямые, обратные и дополнительные коды. Принцип их построения точно такой же, как и в случае целых чисел.

Для целых чисел предполагается, что **двоичная запятая находится правее самого младшего разряда.**

Но она может находится слева от самого старшего разряда – это только предмет договора. Тогда в такой переменной можно будет записывать только дробные числа:

2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-9}	2^{-10}	
1	1	1	1	1	1	1	1	Максимально возможное число (0,999)
			…					
0	0	0	0	0	0	0	0	Минимально возможное число (0)

Можно договориться, что она находится точно посередине переменной, и тогда можно записывать смешанные числа

2^{3}	2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	
1	1	1	1	1	1	1	1	Максимально возможное число (15,9375)
			…					
0	0	0	0	0	0	0	0	Минимально возможное число (0)

Представление чисел в двоичном коде с плавающей запятой

Существует стандарт **IEEE 754** для представления чисел с одинарной точностью (*float*) и с двойной точностью (*double*).

Для записи **числа** в формате с плавающей запятой **одинарной точности** требуется **32-х (тридцатидвухбитовое слово)**. Для записи чисел с **двойной точностью** требуется **64-х (шестидесятичетырёхбитовое слово)**.

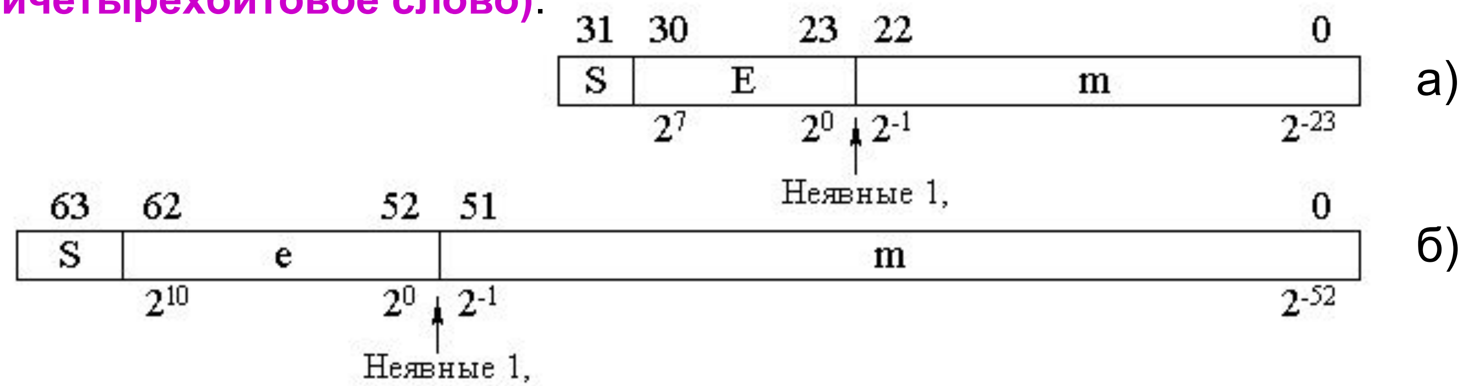


Рис. 6. а) Числа в формате с плавающей запятой одинарной точности и
б) числа в формате с плавающей запятой удвоенной точности.

S обозначен знак числа, **0** – это положительное число, **1** – отрицательное число.

E (e) – смещённый порядок числа. Смещение требуется, чтобы не вводить в число еще один знак.

Смещённый порядок всегда положительное число.

Для смещённого порядка одинарной точности выделено 8 бит.

Для смещённого порядка двойной точности отводится 11 бит.

Для одинарной точности смещение принято 127, а для двойной точности – 1023.

В двоичной мантиссе после запятой может (должна) присутствовать 1. Но *физически её нет* так как она только подразумевается, также как и двоичная запятая. Кроме того, в формате чисел с плавающей запятой принято, что мантисса всегда больше 1. Диапазон значений мантиссы располагается от 1 до 2.

Пример:

1. Определить число с плавающей запятой, лежащее в четырёх соседних байтах:

11000001 01001000 00000000 00000000

– **Знаковый бит**, равный **1** показывает, что число отрицательное.

– **Порядок 10000010** соответствует числу $\mathbf{130_{10}}$. 130 – 127 = 3.

– Теперь запишем **мантиссу**: **1,100 1000 0000 0000 0000 0000**

– И, наконец, определим десятичное число: $\mathbf{1100{,}1_2}$ = – $12{,}5_{10}$

2. Определить число с плавающей запятой, лежащее в четырёх соседних байтах:

11000011 00110100 00000000 00000000

– **Знаковый бит**, равный **1** показывает, что число отрицательное.

– **Порядок 10000110** соответствует числу $\mathbf{134_{10}}$. 134 – 127 = 7.

– Теперь запишем **мантиссу**: **1,011 0100 0000 0000 0000 0000**

– И, наконец, определим десятичное число: $\mathbf{10110100_2}$ = – 180_{10}

Для того чтобы записать **ноль**, достаточно записать в смещенный порядок число $\mathbf{00000000_2}$. Значение мантиссы при этом не имеет значения.

Число, в котором все байты равны 0, тоже попадает в этот диапазон значений.

Бесконечность соответствует смещенному порядку $\mathbf{11111111_2}$ и мантиссе, равной **1,0**. При этом существует **минус бесконечность** и **плюс бесконечность** (переполнение и антипереполнение).