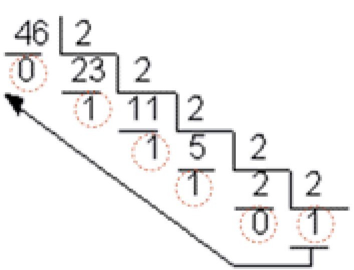
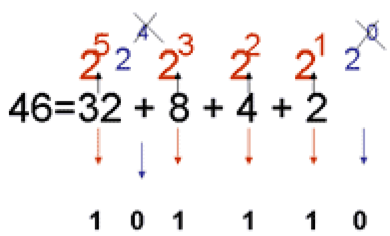
Представление информации в памяти ЭВМ: целые числа, вещественные числа.

Целые числа: $46_{10} \rightarrow 101110_2$





Действительные числа:

$25{,}324 = 0{,}25324 * 10^2$

0,25324 – мантисса, 2 – порядок.

$X = m * p^n$, где m – мантисса,

n – порядок.

В повседневной практике наиболее распространенной является форма представления чисел в виде последовательности цифр, разделенной запятой на целую и дробную части.

Числа, представленные в такой форме, называются числами *с естественной запятой* или *числами в естественной форме*.

Например:

- 12560 - целое число,
- 0,003572 - правильная дробь,
- 4,89760 - неправильная дробь.

Формой представления чисел в цифровых автоматах называется совокупность правил, позволяющих установить взаимное соответствие между записью числа и его количественным эквивалентом.

Для представления чисел в ЭВМ используют битовые наборы — последовательности нулей и единиц фиксированной длины.

Организовать обработку наборов фиксированной длины технически легче, чем наборов переменной длины.

Позиция в битовом наборе называется разрядом.

В памяти ЭВМ целые и вещественные числа представлены в разных форматах : **с фиксированной и плавающей запятой (точкой)**.

В обычной арифметике для записи числа можно использовать теоретически бесконечное число разрядов, но в компьютере для представления каждого числа используется лишь фиксированное число разрядов (бит).

Целые числа

Хранятся в формате с фиксированной запятой. Каждому разряду ячейки памяти соответствует всегда один и тот же разряд числа, а «запятая» находится справа после младшего разряда, вне разрядной сетки.

Для беззнаковых чисел — все разряды предназначены для записи значения числа.

Для чисел со знаком — старший (самый левый) бит предназначен для знака: если старший бит равен нулю, то это — положительное число, если равен единице — отрицательное.

Максимальное по абсолютной величине двоичное число изображается единицами во всех разрядах, исключая знаковый,

т.е. для целого числа $A_{max} = (2^{(n-1)} - 1)$,

где n - полная длина разрядной сетки.

В случае 16-разрядной сетки $A_{max} = (2^{(16-1)} - 1) = 32767_{10}$,

т.е. диапазон представления целых чисел в этом случае будет

от -32768_{10} до $+32767_{10}$.

Целые числа, у которых модуль больше, чем $(2^{(n-1)} - 1)$ и меньше единицы не представляются в форме с фиксированной запятой.

Числа, по абсолютной величине меньше единицы младшего разряда разрядной сетки, называются в этом случае <u>машинным нулем.</u>

Максимальные значения целых чисел при их 8 и 32 разрядном представлении:

	8 разрядов	32 разряда	числа в 16-разрядной сетке в формате с фиксированной точкой: 25 , -610:
со знаком	2^7-1=127	2^{31}-1=2,15x10^9	[0]000 0000 0001 1001
без знака	2^8-1=255	2^{32}-1=4,29x10^9	[1]000 0010 0110 0010

Диапазон допустимых значений целых чисел зависит от размера ячеек памяти, используемых для их хранения.

для 16-разрядной ячейки — от 0 до 65535,
для 8-разрядной — от 0 до 255.

Представление в ЭВМ целых чисел

Целые числа без знака

00000000_2	11111111_2
0	2^8-1
0	255
$00000000\ 00000000_2$	$11111111\ 11111111_2$
0	$2^{16}-1$
0	65535

Пример:

$X = 72_{10}$

$72_{10} = 1001000_2$

7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0

$65535_{10} = 11111111\ 11111111_2$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Целые числа со знаком

1 байт	$-2^7 \ldots 2^7-1$	$-128 \ldots 127$
2 байта	$-2^{15} \ldots 2^{15}-1$	$-32768 \ldots 32767$
4 байта	$-2^{31} \ldots 2^{31}-1$	
	$-2147483648 \ldots 2147483647$	

Для представления знаковых целых чисел используются три способа:

1) прямой код;
2) обратный код;
3) дополнительный код.

Все три способа используют самый левый (старший) разряд битового набора длины k для кодирования знака числа:

знак “плюс” кодируется нулем, а “минус” — единицей.

Остальные k-1 разрядов (называемые мантиссой или цифровой частью) используются для представления абсолютной величины числа.

При помощи этих кодов:

- упрощается определение знака результата операции,
- операция вычитания чисел сводится к арифметическому сложению их кодов,
- облегчается выработка признака переполнения разрядной сетки.

Положительные целые числа (и число 0)

Положительные числа в прямом, обратном и дополнительном кодах изображаются одинаково — цифровая часть содержит двоичную запись числа, в знаковом разряде содержится 0. Например, для k = 8:

Число $1_{10} = 1_2$

Число $127_{10} = 1111111_2$

Отрицательные целые числа

Отрицательные числа в прямом, обратном и дополнительном кодах имеют разное изображение.

Прямой код отрицательных чисел

В знаковый разряд помещается цифра 1, а в разряды цифровой части числа — двоичный код его абсолютной величины.

Прямой код числа –1

Прямой код числа –127

Обратный код отрицательных чисел

Получается инвертированием всех цифр двоичного кода абсолютной величины числа, включая разряд знака: нули заменяются единицами, а единицы — нулями.

Пример (k = 8):

Число: –1

Код модуля числа:

0	0	0	0	0	0	0	1

Обратный код числа:

1	1	1	1	1	1	1	0

Знак числа "–"

Число: –127

Код модуля числа:

0	1	1	1	1	1	1	1

Обратный код числа:

1	0	0	0	0	0	0	0

Знак числа "–"

Дополнительный код отрицательных чисел

Получается образованием обратного кода с последующим прибавлением единицы к его младшему разряду.

Дополнительный код числа –1:

1	1	1	1	1	1	1	1

Знак числа "–"

Дополнительный код числа –127:

1	0	0	0	0	0	0	1

Знак числа "–"

Один и тот же битовый набор длины k можно интерпретировать по-разному:

1) как представление некоторого числа без знака;

2) как представление некоторого числа со знаком (в прямом, обратном или дополнительном коде).

Компьютер не знает, что именно представляет тот или иной битовый набор —

для него это просто слово в алфавите {0,1},

а смысл этого слова известен программисту.

Числовым значением такого слова будем называть неотрицательные целое, двоичная (k-разрядная) запись которого совпадает с данным словом.

Пусть x — число со знаком. Тогда числовое значение его обратного и дополнительного кодов можно определить с помощью функций:

$$обр(x) = \begin{cases} x, \text{ если } x \geq 0 \\ 2^k - 1 - |x|, \text{ если } x < 0 \end{cases} \qquad доп(x) = \begin{cases} x, \text{ если } x \geq 0 \\ 2^k - |x|, \text{ если } x < 0 \end{cases}$$

Например, при $k = 8$ $обр(+1)=доп(+1)=1=00000001_2$;

$обр(-127)= 255 - 127 = 128 = 10000000_2$; $доп(-127) = 256 - 127 = 129 = 10000001_2$;

$обр(-1) = 255 - 1 = 254 = 11111110_2$; $доп(-1) = 256 - 1 = 255 = 11111111_2$;

Обратный код

-1_{10}
1 0000001
1 1111110

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	0

Знак числа

-127_{10}
1 1111111
1 0000000

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0

Знак числа

-83_{10}
1 1010011
1 0101100

7	6	5	4	3	2	1	0
1	0	1	0	1	1	0	0

Знак числа

Дополнительный код

-1_{10}
1 0000001
1 1111110
1 1111111

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1

Знак числа

-127_{10}
1 1111111
1 0000000
1 0000001

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	1

Знак числа

-83_{10}
1 1010011
1 0101100
1 0101101

7	6	5	4	3	2	1	0
1	0	1	0	1	1	0	1

Знак числа

Действия с использованием обратного кода

При использовании алгебраического суммирования отрицательные числа представляются в **обратном коде**, положительные – в прямом и происходит поразрядное суммирование, включая знаковые разряды, которые при этом рассматриваются как старшие разряды.

Если возникает единица переноса из знакового разряда, то она прибавляется к младшему разряду суммы кодов. Такой перенос называется круговым или циклическим.

Если знаковый разряд суммы равен "*0*", то это означает, что результат положительный и представлен в прямом коде.

Если в знаковом разряде суммы единица, то это означает, что результат отрицательный и представлен в обратном коде.

Пример: A=5, B=4

00000101	+5 прямой код	00000100	+4 прямой код
11111010	-5 обратный код	11111011	-4 обратный код

A+B

```
          00000101        +5 прямой код
+         00000100        +4 прямой код
          00001001        +9 прямой код
```

A-B

```
          A+[B]об
          00000101        +5 прямой код
+         11111011        -4 обратный код
        1 00000000        перенос из знакового разряда суммируется
                +1
          00000001        прямой код числа 1, знаковый разряд результата равен 0.
          Из этого следует, что результат положительный и представлен в прямом коде.
```

имя	A	B	-A	-B
значение	5	4	-5	-4
прямой код	00000101	00000100	10000101	10000100
обратный код	00000101	00000100	11111010	11111011

Теперь из меньшего числа вычтем большее.

B-A= -1

$B+[A]_{об}$

	00000100	+4 прямой код
+	11111010	-5 обратный код
=	11111110	Переноса из знакового разряда не было.
		Знаковый разряд результата равен 1.

Следовательно, результат получился отрицательным и представлен в обратном коде.

Прямой код результата равен:

10000001 - прямой код числа -1

Действия с использованием дополнительного кода

- При алгебраическом суммировании с использованием дополнительного кода, отрицательные числа представляются в дополнительном коде, а положительные – в прямом коде и производится суммирование кодов чисел, включая знаковый разряд.
- При возникновении единицы переноса из знакового разряда, эта единица отбрасывается в отличие от обратного кода. В результате получается алгебраическая сумма в прямом коде, если сумма положительна, и в дополнительном, если она отрицательна.

Пример: A=5,		B=4	
00000101	+5 прямой код	00000100	+4 прямой код
11111010	-5 обратный код	11111011	-4 обратный код
11111011	-5 дополнит. код	11111100	-4 дополнит. код

Пример: A-B=1

	00000101	+5 прямой код
+	11111100	-4 дополнительный код
=	1 00000001	Получили +1, прямой код.
		Единица переноса в знаковый разряд отбрасывается.

Если образовался “0” в знаковом разряде, то число получилось положительным и представлено в прямом коде, а если “1”, то это число отрицательное и представлено в дополнительном коде.

Пример: <u>B-A</u>

	0000 0100	+4 прямой код
+	1111 1011	-5 дополнительный код
=	1111 1111	

Знаковый разряд результата равен 1.
Из этого следует, что результат получился отрицательным и представлен в дополнительном коде.
Прямой код результата равен:

	1000 0000	
+	1	
	1000 0001	Результат : -1 прямой код

Два варианта преобразования после получения отрицательного результата:

- меняем все разряды кроме знакового (0 на 1, 1 на 0) и прибавляем единицу
- сначала вычитаем единицу, а потом инвертируем разряды кроме знакового.

Вещественные числа

Хранятся в формате с плавающей запятой.

Неудобство представления чисел в форме с фиксированной точкой проявляется при решении задач, в которых фигурируют как очень малые, так и очень большие числа.

Точность числа определяется не его длиной, а количеством верных значащих цифр.

Число с плавающей запятой состоит из набора отдельных разрядов, условно разделенных на знак, порядок и мантиссу.

Порядок и мантисса — целые числа, которые вместе со знаком дают представление числа с плавающей запятой в следующем виде (например, в 2 системе счисления) :

$$(-1)^{S} \times M \times 2^{k}$$

где s — знак, k — порядок, M — мантисса.

Примеры:

десятичное число

$$155,625 = 1,55625 \cdot 10^{+2}$$

при переводе в 2 СС

$$2_{10} = 10_{2} = 1.000e{+}1 = 0.100e{+}2 = 0.010e{+}3$$

IEEE 754 — стандарт формата представления вещественных чисел (чисел с плавающей точкой) в компьютере.

(**IEEE -** *Institute of Electrical and Electronics Engineers)*

Этот стандарт нужен, чтобы разные компьютерные архитектуры могли одинаково удобно и эффективно работать с вещественными числами.

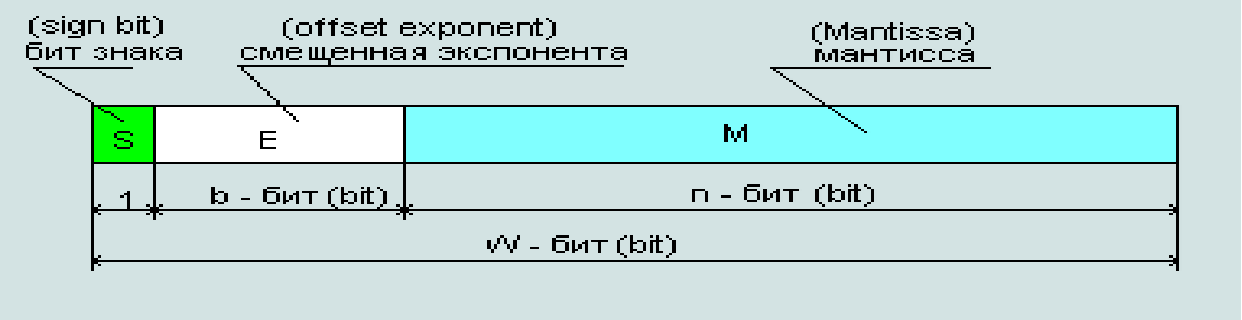
Стандарт IEEE 754 определяет:

- как представлять нормализованные положительные и отрицательные числа с плавающей точкой
- как представлять денормализованные положительные и отрицательные числа с плавающей точкой
- как представлять нулевые числа
- как представлять специальную величину бесконечность (Infinity)
- как представлять специальную величину "Не число" (NaN или NaNs)
- четыре режима округления

Как представлять нормализованные положительные и отрицательные числа с плавающей точкой?

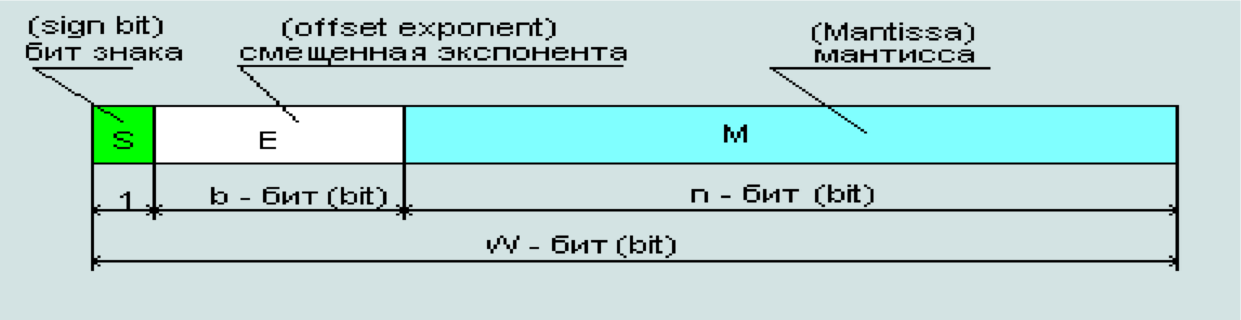
Некоторые особенности стандарта:

- Число с плавающей запятой состоит из набора отдельных двоичных разрядов, условно разделённых на знак, порядок и мантиссу.
- Один бит используется для указания знака числа (0 — если число положительное, 1 — если число отрицательное).
- Порядок записывается как целое число в коде со сдвигом, а мантисса — в нормализованном виде, своей дробной частью в двоичной системе счисления.



Формальное представление чисел в стандарте IEEE 754 для любого формата точности:

- S - бит знака, если S=0 - положительное число; S=1 - отрицательное число
- E - смещенная экспонента двоичного числа; Для определения знака экспоненты, чтобы не вводить еще один бит знака, добавляют смещение к экспоненте $exp_2 = E - (2^{(b-1)} - 1)$ - экспонента двоичного нормализованного числа с плавающей точкой, где $(2^{(b-1)} - 1)$ - заданное смещение экспоненты (в 32-битном ieee754 оно равно +127, в половину байта)
- M - остаток мантиссы двоичного нормализованного числа с плавающей точкой

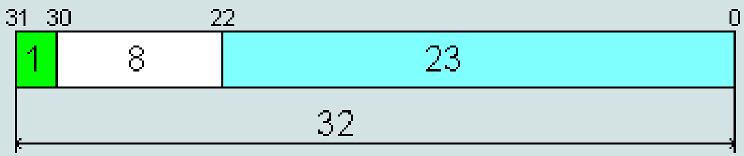


Смещенный порядок *n*-разрядного нормализованного числа вычисляется следующим образом:
если для задания порядка выделено *k* разрядов, то к истинному значению порядка, представленного в дополнительном коде, прибавляют смещение, равное $(2^{k-1} - 1)$.

Таким образом, порядок, принимающий значения в диапазоне от -128 до +127, преобразуется в смещенный порядок в диапазоне от 0 до 255.

Смещенный порядок хранится в виде беззнакового числа, что упрощает операции сравнения, сложения и вычитания порядков, а также упрощает операцию сравнения самих нормализованных чисел.

Формат числа одинарной точности (single-precision) 32 бита



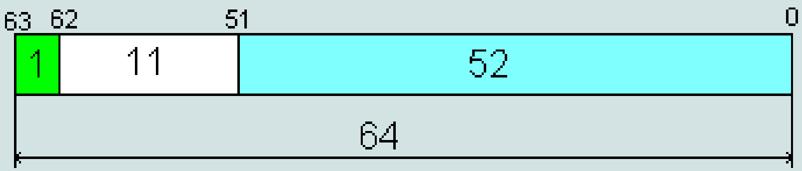
$$F = (-1)^S 2^{(E-127)}(1+M/2^{23})$$

	E – смещенная экспонента	E-127	Реальный порядок
0	0000 0000	0-127	-127
1	0000 0001	1-127	-126
2	0000 0010	2-127	-125
…	…	…	…
…	…	…	…
126	0111 1110	126-127	-1
127	0111 1111	127-127	0
128	1000 0000	128-127	1
129	1000 0001	129-127	2
130	1000 0010	130-127	3
…	…	…	…
…	…	…	…
253	1111 1101	253-127	126
254	1111 1110	254-127	127
255	1111 1111	255-127	128

Количество разрядов, отводимых под порядок, влияет на диапазон от наименьшего отличного от нуля числа до наибольшего числа, представимого в машине при заданном формате.

В числах одинарной точности порядок состоит из 8 бит, а мантисса – из 23. Эффективный(реальный) порядок определяется как E-127.

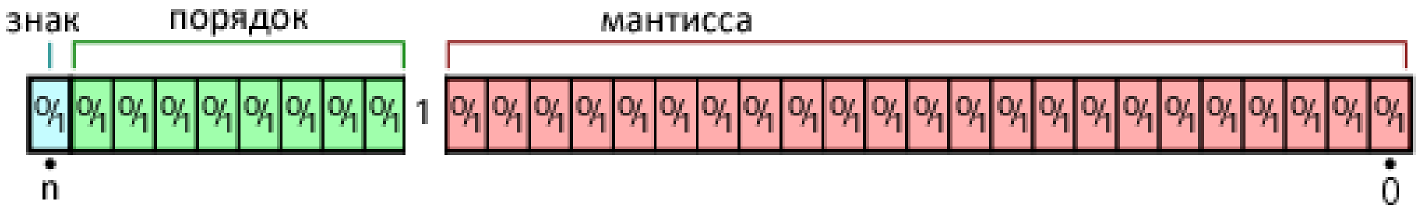
Формат числа двойной точности (double-precision) 64 бита



$$F = (-1)^S \; 2^{(E-1023)} \, (1+M/2^{52})$$

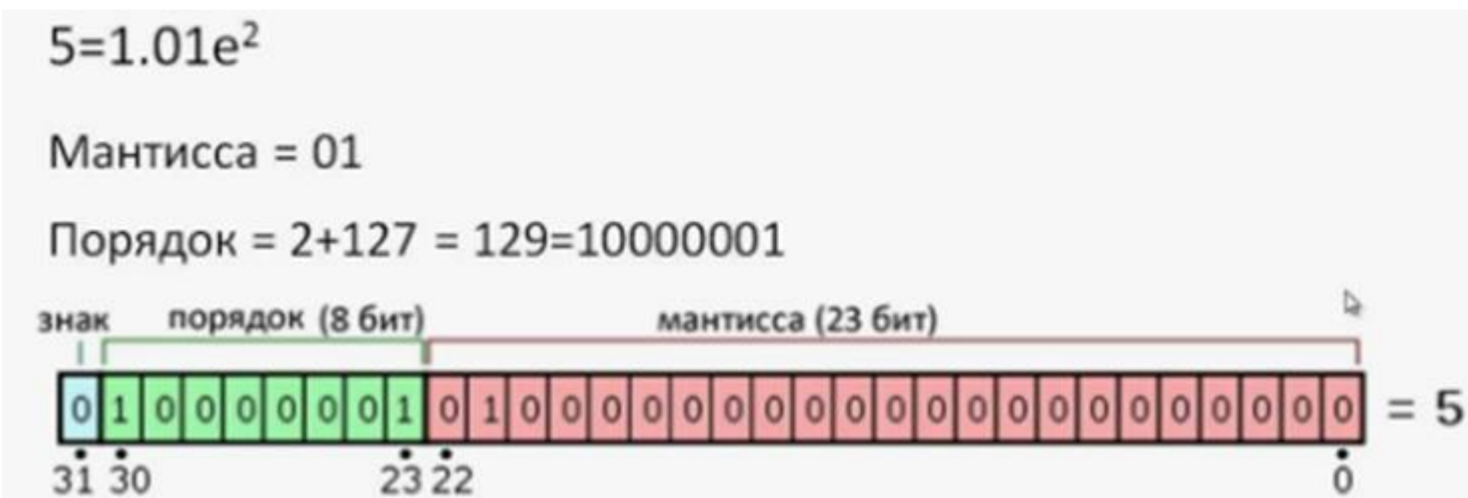
Стандартные форматы представления вещественных чисел

Формат	Что хранится	Кол-во битов, отводимых под смещенный порядок	Кол-во битов, отводимых под мантиссу
Одинарный	**32-разрядное нормализованное число со знаком**	**8**	**23**
Двойной	**64-разрядное нормализованное число со знаком**	**11**	**52**
Расширенный	**80-разрядное число со знаком (возможно ненормализованные)**	**15**	**64**



- Чем больше разрядов отводится под запись мантиссы, тем выше точность представления числа.
- В связи с тем, что у нормализованных вещественных чисел старший бит мантиссы всегда равен 1, этот <u>старший бит не хранится в памяти</u>. Это экономит один бит, так как неявную единицу не нужно хранить в памяти.
- Число 2 имеет единственное нормализованное представление («1.000e+1»), поэтому мантисса хранится в памяти как «000», т.к. старшая единица подразумевается неявно.

- Число 5 в двоичной системе счисления равно 101, его запись в памяти в виде нормализованного числа:



Нормализованное вещественное число - 118.625

-118.625

118d = 1110110b

$0.625d = 1\cdot2^{0}+1\cdot2^{-1}+0\cdot2^{-2}+1\cdot2^{-3}=0.101b$

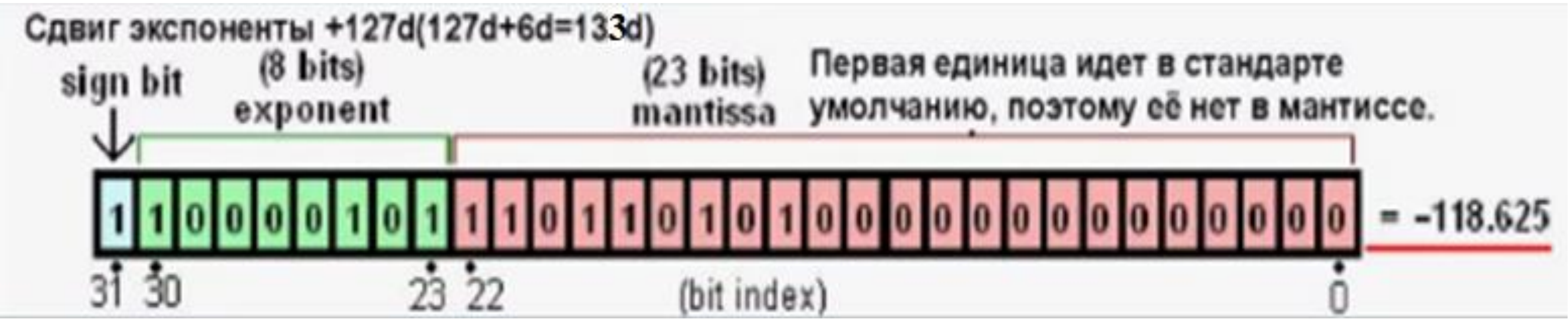
-1110110.101b

$$-\,1.110110101 \ * \ 2^{+6}$$

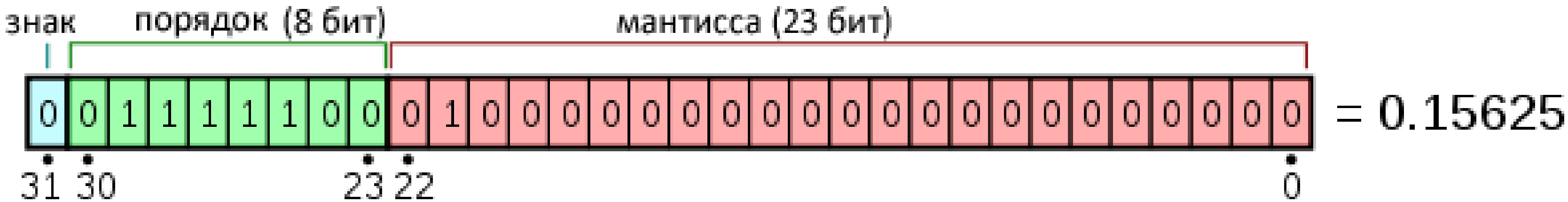
Знак мантиссы

Мантисса

Порядок (экспонента)



Например, число 0,15625 будет записано в памяти как число нормализованное:



В этом примере: Знак $s=0$ (положительное число)

Порядок $01111100_2-127_{10} = 124_{10}-127_{10} = -3$

Мантисса M 1.01_2 (первая единица неявная)

В результате получаем число

$$F = 1.01e\text{-}3 = 0.00101 = 2^{-3}+2^{-5} = 0{,}125 + 0{,}03125 = 0{,}15625$$

Пример:

Представить двоичное число 101.10_2 в нормализованном виде, записать в 32-битом стандарте IEEE754.

Мантисса M=1.011 Экспонента $exp_2=2$

Смещенная экспонента 2+127=129 1000 0001

Результат:

0 1000 0001 01100000000000000000000

Пример:

409,625 = 110011001,101 = 1,10011001101 e +8

E-127=8

E =8+127 = 135 1000 0111 - смещенная экспонента

Результат:

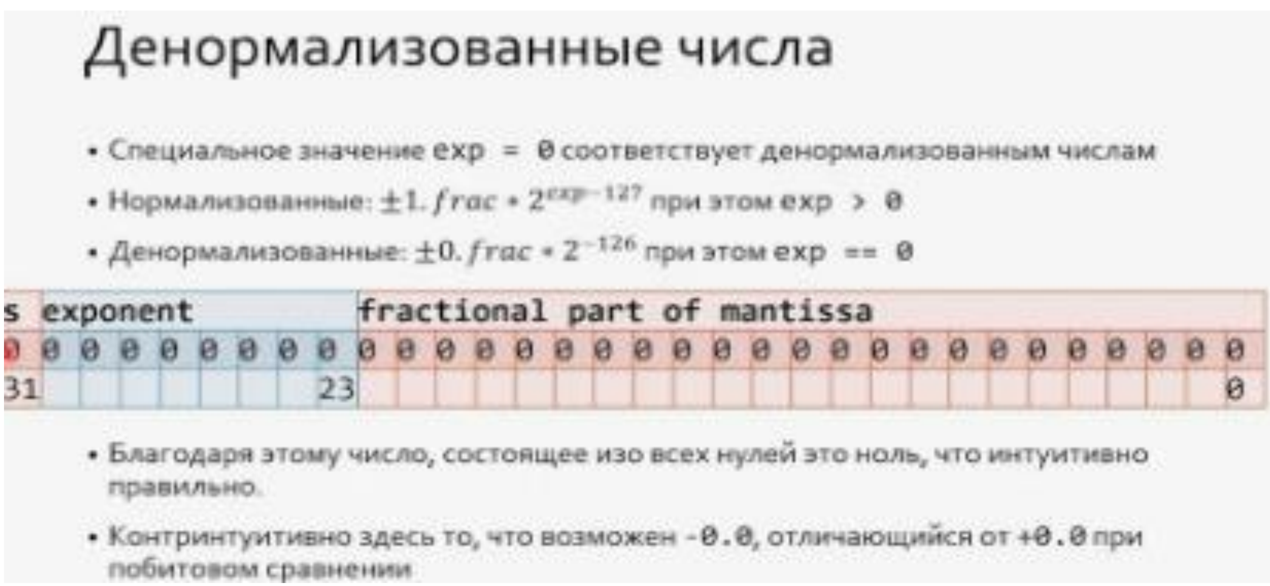
0 1000 0111 10011001101000000000000

Денормализованные числа

Для повышения точности вычислений:

каждое значение денормализованного числа меньше самого маленького нормализованного значения числа с плавающей запятой.

Мантисса: 0.1 <= M < 1



Денормализованное число

- Условие: exp = 000…0
- Значение порядка: `E = –Смещение + 1`
  (вместо `E = 0 – Смещение`)
- Мантисса кодируется с ведущим `0`: `M = 0.xxx…x`$_2$
  - `xxx…x`: биты поля `frac`
- Примеры
  - exp = 000…0, frac = 000…0
    - Представляет число ноль
    - Различные кодировки для +0 и –0
  - exp = 000…0, frac ≠ 000…0
    - Кодируются числа близкие к 0.0