**04.12.2020 ИС 2к Основы архитектуры**

**Тема: 2.1.** **Арифметические основы цифровой техники**

***(Учебник «Основы архитектуры, устройство и функционирование вычислительных систем» Степина ВВ)***

***стр. 36 -40)*** [**https://yadi.sk/i/1PZf\_-ZO43S0fA**](https://yadi.sk/i/1PZf_-ZO43S0fA)

**Ответить на вопросы:**

1. Перевести число DCCXIX из римской системы счета в арабскую.
2. Составить алгоритм перевода из двоичной в восьмеричную и шестнадцатеричную системы.
3. Перевести числа из десятичной системы счисления в восьмеричную систему счисления:

23,1310 - Х8;

54,3910 - X8.

1. Перевести числа из двоичной системы счисления в шестнадцатеричную систему счисления:

11101101,10112 - X16;

1000100101,1010012 - X16.

1. Перевести числа из восьмеричной системы счисления в шестнадцатеричную систему счисления:

25,58 - X10;

36,318 - X10.

**Ответы присылать** [kuzn117@yandex.ru](mailto:kuzn117@yandex.ru)

**можно в ВК -**  **id480169637**

**Пишите тему письма и файл именуйте: Фамилия-arx-0512**

**Для тех у кого проблемы с Яндекс диском сегодняшнюю тему**

**смотрите ниже**

**Глава 2  
ЦИФРОВОЙ**

**ЛОГИЧЕСКИЙ УРОВЕНЬ**

**2.1. Арифметические основы цифровой техники**

В цифровых устройствах приходится иметь дело с различными  
видами информации. Это в чистом виде двоичная информация, та-  
кая как включен прибор или выключен, исправно устройство или  
нет. Информация может быть представлена в виде текстов, и тогда  
приходится буквы алфавита кодировать при помощи двоичных уров-  
ней сигнала. Достаточно часто информация может представлять со-  
бой числа. Числа могут быть представлены в различных системах  
счисления. Форма записи в них чисел существенно различается  
между собой, поэтому, прежде чем перейти к особенностям представ-  
ления чисел в цифровой технике, рассмотрим их запись в различных  
системах счисления.

***Системы счисления***

Система **счисления** — это совокупность приемов и правил для  
представления чисел с помощью цифровых знаков.

Существует множество способов записи чисел цифровыми знаками,  
но любая применяемая система счисления должна обеспечивать:

• диапазон представления любого числа;

• единственность представления (каждой комбинации символов  
соответствует только одна величина).

Все системы счисления делятся на позиционные и непозиционные.  
В ***непозиционной*** системе счисления значимость цифры в любом  
месте числа одинакова, т.е. не зависит от позиции расположения.  
Например, унарная система с одним символом, равным единице.  
Такая система счислений предназначена для суммарного счета  
(узелки на «память», зарубки, черточки, счет на пальцах и пр.). Для  
изображения какого-то числа в этой системе нужно записать число  
единиц (палочек), равное данному числу. Эта система неэффективна,  
так как запись числа получается слишком длинной.

Другой пример «почти непозиционной» системы счисления —  
римская система счета. В римской системе счета используются сле-  
дующие символы:

I— 1; V— 5;Х— 10; L - 50; С - 100; D - 500; М - 1000.

*43*

Правила перевода из римской системы счисления в арабскую сис-  
тему следующие.

Меньшая по величине цифра, стоящая справа от большей цифры,  
складывается с большей, а меньшая по величине цифра, стоящая  
слева от большей цифры, вычитается из большей.

Пример перевода из римской системы в арабскую систему счи-  
сления:

CCXWII = 100 + 100 + 10 + 5 + 5 + 1 + 1 = 222;

XIXIV= 10 + (10 - 1) = 19.

Как следует из правила перевода, римская система полностью не  
является непозиционной. Эта система применяется редко (цифер-  
блат, архитектура, история и т.д.).

***Позиционные*** системы счисления — это системы счисления, в ко-  
торых значение цифры в записи числа ***N*** зависит от ее позиции  
(места). Например, в десятичной системе счисления число 05 обозна-  
чает пять единиц, 50 обозначает пять десятков, 500 — пять сотен и т.д.

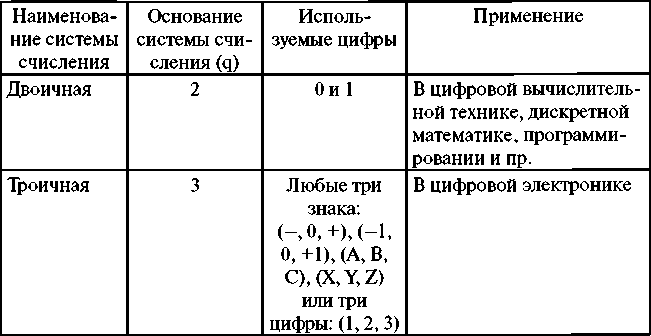
Основание (базис) системы счисления (***q) —*** это количество ис-  
пользуемых знаков или символов для изображения цифр в данной  
системе счисления.

Возможно бесчисленное множество позиционных систем счисле-  
ния, так как за основание можно принять любое число и образовать  
новую систему счисления.

Примеры некоторых позиционных систем счисления и их при-  
менение приведены в табл. 2.1.

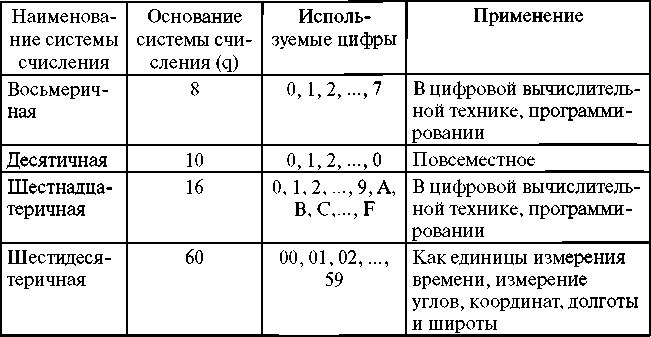
*Таблица 2.1*

**Примеры позиционных систем счисления**



*44*

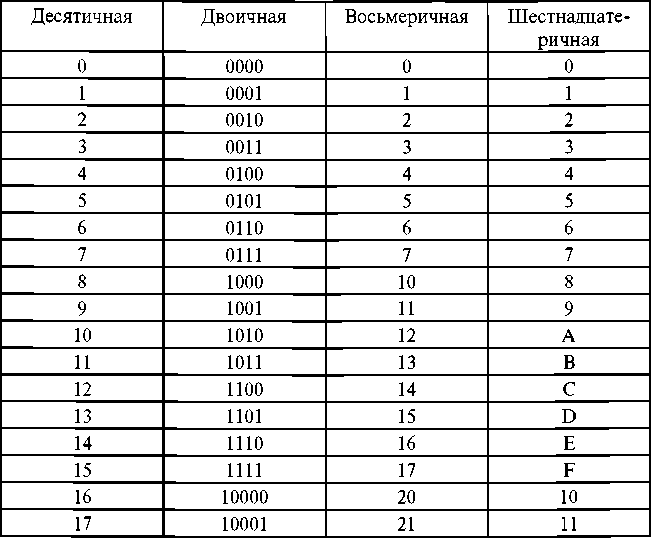
*Окончание табл. 2.1*



В табл. 2.2 для удобства сопоставления приведены первые  
23 числа натурального ряда чисел в различных системах счисления.

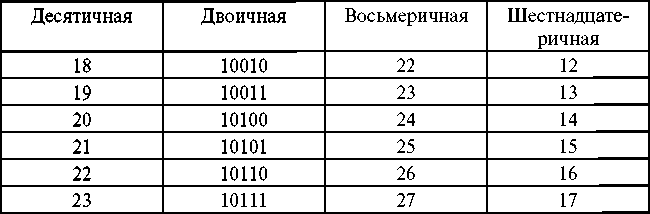
*Таблица 2.2*

**Натуральный ряд чисел в различных системах счисления**



*45*

*Окончание табл. 2.2*

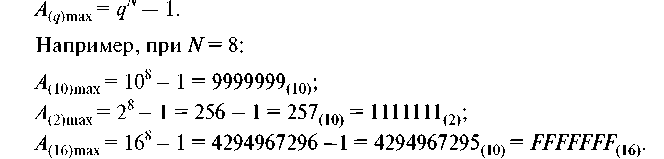


Как видно из табл. 2.2, для записи одного и того же числа в различ-  
ных системах счисления требуется разное число позиций или разря-  
дов. Например, 1410 = 11102 = 168 = ***Е16.*** То есть в десятичной системе  
счисления число 14 занимает две позиции (два разряда), в двоичной  
системе счисления — четыре позиции, в шестнадцатеричной системе  
счисления — одну позицию. Чем меньше основание системы счисле-  
ния ***q,*** тем больше длина числа (длина разрядной сетки).

При заданной длине разрядной сетки ограничивается максималь-  
ное по абсолютному значению число, которое можно записать.

Пусть длина разрядной сетки равна положительному числу ***N,***максимальное число равно:

*46*



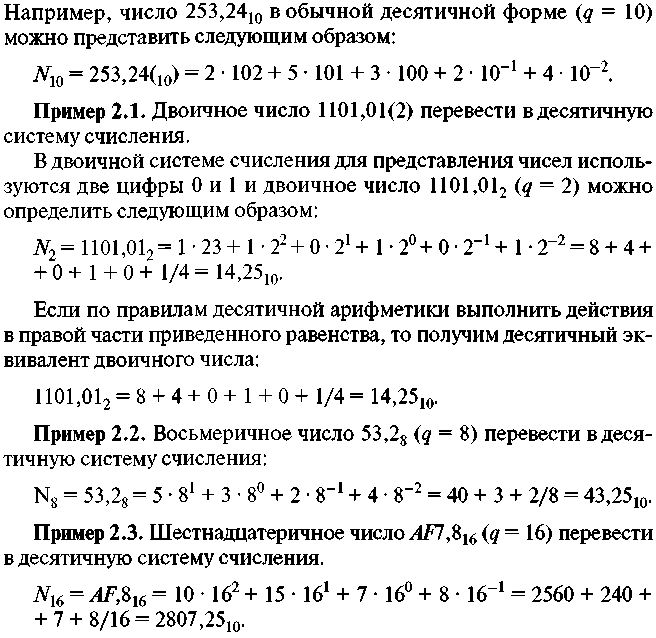
Таким образом, при одинаковой длине разрядной сетки ***N=*** 8  
максимальное по абсолютному значению Л(16)тах > Л(10)П1ах > Л(2)П1ах,  
т.е. чем больше ***q,*** тем большеyl(?)max.

***Перевод в позиционных системах счисления***

**1. Перевод в десятичную систему счисления.**

Любое число ***N*** в позиционной системе счисления можно пред-  
ставить в виде полинома

где ***q —*** основание системы счисления; ***а —*** элемент числа ***Nq,*** прини-  
мающий значения 0, 1, 2, ..., (#—1); *т* ***—*** номер разряда целой части,  
отсчитываемый от нулевого; ***к —*** число цифр в дробной части числа.



*47*

2. **Перевод чисел из десятичной системы счисления в произвольную  
систему счисления с основанием *q.***

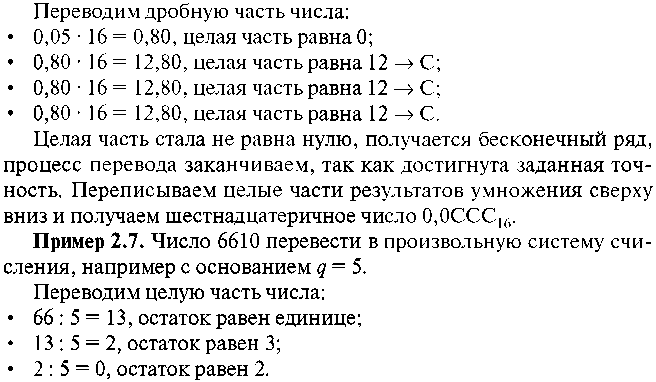
Правила перевода целой части десятичного числа следующие.  
Целую часть десятичного числа необходимо последовательно делить  
на ***q*** (основание произвольной системы счисления) до тех пор, пока  
десятичное число не станет равно нулю. Остатки, полученные при  
делении и записанные в последовательности, начиная с последнего  
остатка, являются цифрами числа ^-ричной системы счисления.

Правила перевода дробной части десятичного числа следующие.  
Дробную часть десятичного числа необходимо последовательно ум-  
ножать на ***q*** (основание произвольной системы) и отделять целую  
часть до тех пор, пока она не станет равной нулю или будет достиг-  
нута заданная точность перевода.

Целые части результатов умножения в порядке, соответствующем  
их получению, составляют число в новой системе.



*48*



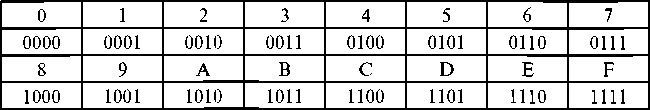
**Таблица триад и тетрад**

*Таблица 2.3*

Триады



Тетрады



*49*

Десятичное число стало равно нулю, деление закончено. Перепи-  
сываем все остатки снизу вверх и получаем пятеричное число 2315.

**3. Перевод из двоичной системы в восьмеричную и шестнадцатеричную.**

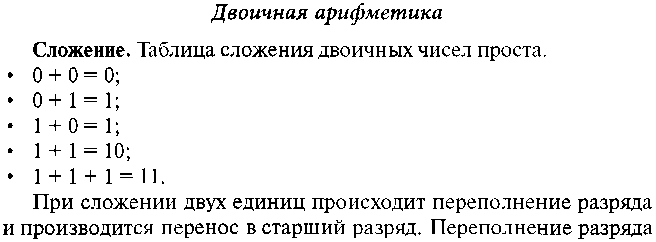
Для этого типа операций существует упрощенный алгоритм.

*Перевод целой части*

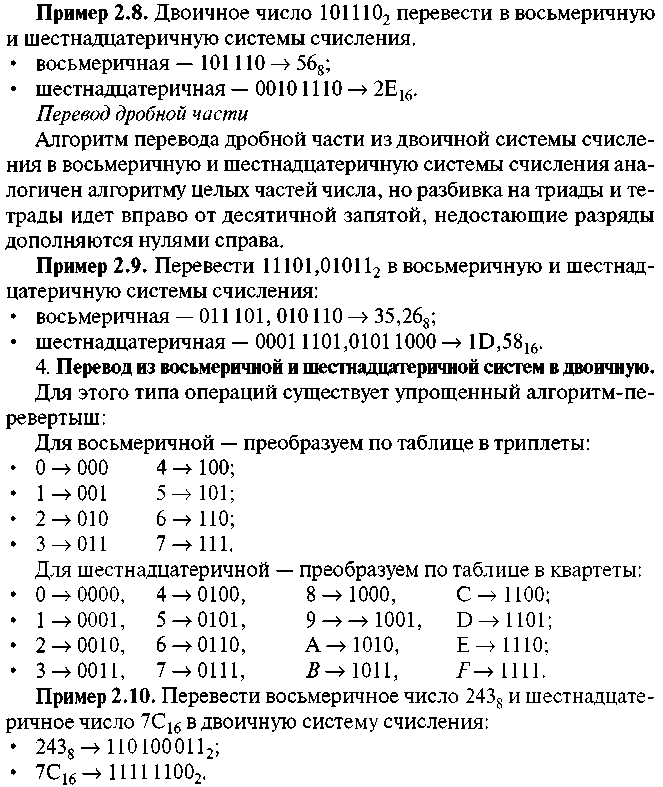
Число 2 возводится в ту степень, которая необходима для получе-  
ния основания системы, в которую требуется перевести. Для вось-  
меричной системы (8 = 23) получаем число 3 (триада), для шестнад-  
цатеричной системы (16 = 24) получаем число 4 (тетрада).

Разбиваем переводимое число на количество цифр, равное 3 для  
восьмеричной системы и равное 4 для шестнадцатеричной системы  
счисления.

Преобразуем триады по таблице триад восьмеричной системы и те-  
трады по таблице тетрад для шестнадцатеричной системы счисления.

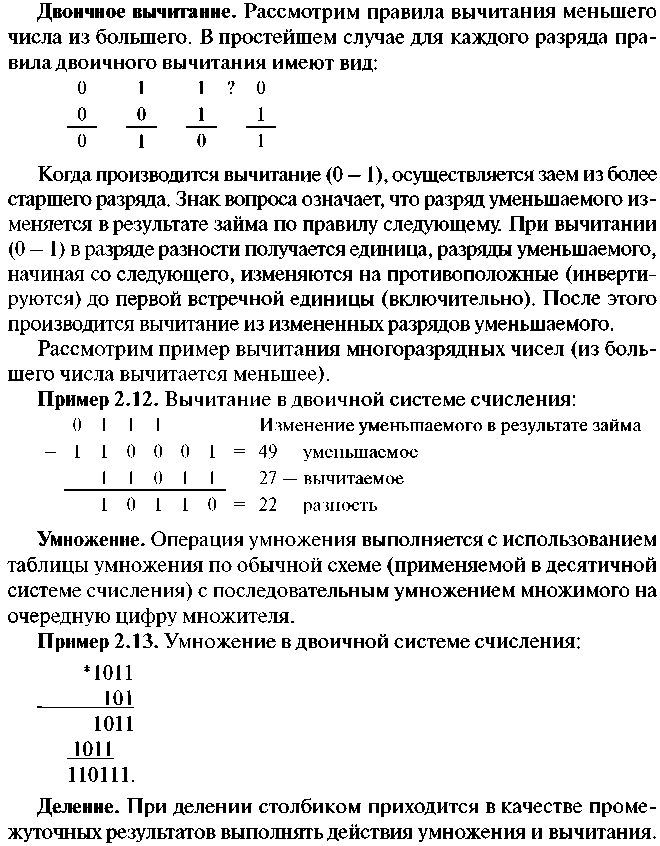
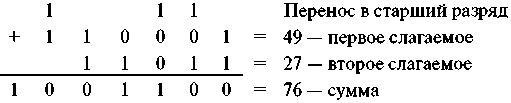


*50*



наступает тогда, когда величина числа в нем становится равной или  
большей основания.

Пример 2.11. Выполнить сложение в двоичной системе счисления:



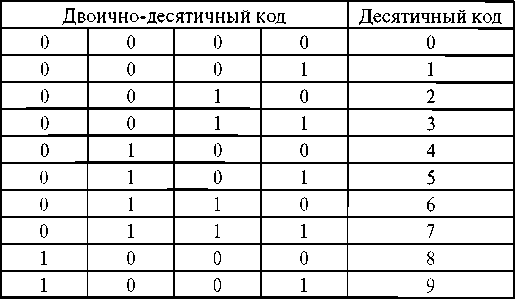
*51*

***Запись десятичных чисел (двоично-десятичный код)***

Иногда бывает удобно хранить числа в памяти процессора в де-  
сятичном виде (например, для вывода на экран дисплея). Для записи  
таких чисел используются двоично-десятичные коды. Для записи  
одного десятичного разряда используется четыре двоичных бита (те-  
трады). При помощи четырех битов можно закодировать шестнад-  
цать цифр (24 = 16). Лишние комбинации в двоично-десятичном  
коде являются запрещенными. Соответствие двоично-десятичного  
кода и десятичных цифр приведено в табл. 2.4.

*Таблица 2.4*

**Соответствие двоично-десятичного кода и десятичных цифр**



Остальные комбинации двоичного кода в тетраде являются за-  
прещенными.

Пример 2.14. Записать двоично-десятичный код числа 125810:

125810 = 000 1 00 1 00101 10002.

В первой тетраде записана цифра 1, во второй — 2, в третьей — 5,  
а в последней тетраде записана цифра 8. В данном примере для за-  
писи числа 1258 потребовалось четыре тетрады. Количество ячеек  
памяти микропроцессора зависит от его разрядности. При 16-раз-  
рядном процессоре все число уместится в одну ячейку памяти.

Пример 2.15. Записать двоично-десятичный код числа 58910:

58910 = 00000101 100010012.

В данном примере для записи числа достаточно трех тетрад, но  
ячейка памяти 16-разрядная. Поэтому старшая тетрада заполняется  
нулями. Они не изменяют значение цифры.

*52*

При записи десятичных чисел часто требуется записывать знак  
числа и десятичную запятую (в англоязычных странах — точку). Дво-  
ично-десятичный код часто применяется для набора телефонного  
номера или набора кодов телефонных служб. В этом случае кроме  
десятичных цифр часто применяются символы «\*» или «#». Для за-  
писи этих символов в двоично-десятичном коде применяются запре-  
щенные комбинации (табл. 2.5).

*Таблица 2.5*

**Соответствие двоично-десятичного кода и дополнительных символов**

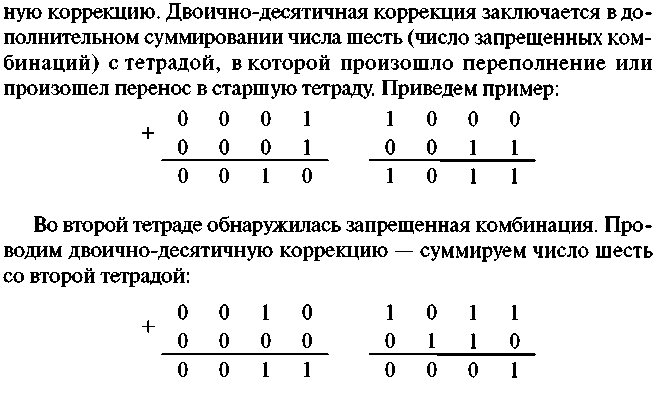
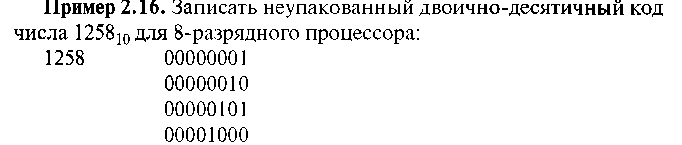


Достаточно часто в памяти процессора для хранения одной деся-  
тичной цифры выделяется одна ячейка памяти (8-, 16- или 32-раз-  
рядная). Это делается для повышения скорости работы программы.  
Для того чтобы отличить такой способ записи двоично-десятичного  
числа от стандартного, способ записи десятичного числа, как это  
показано в примере, называется упакованной формой двоично-де-  
сятичного числа.

*53*

В первой строке записана цифра 1, во второй — 2, в третьей — 5,  
а в последней строке записана цифра 8. В данном примере для за-  
писи числа 1258 потребовалось четыре строки (ячейки памяти).

Суммирование двоично-десятичных чисел. Суммирование двоично-  
десятичных чисел можно производить по правилам обычной двоич-  
ной арифметики, а затем производить двоично-десятичную коррек-  
цию. Двоично-десятичная коррекция заключается в проверке  
каждой тетрады на допустимые коды. Если в какой-либо тетраде  
обнаруживается запрещенная комбинация, то это говорит о пере-  
полнении. В этом случае необходимо произвести двоично-десятич-



*54*



***Формы представления в ЭВМ числовых данных***

В математике используются две формы записи чисел: есте-  
ственная (число записывается в естественном натуральном виде)  
и нормальная (запись числа может быть различной в зависимости от  
ограничений, накладываемых на форму).

Примеры ***естественной формы*** записи чисел:

15300 — целое число; 0,000564 — правильная дробь; 6,4540 — не-  
правильная дробь.

Пример нормальной формы записи одного и того же числа 25340  
в зависимости от ограничений, накладываемых на нормальную форму:

В вычислительной технике при естественном представлении чи-  
сел устанавливаются длина разрядной сетки, а также фиксированное  
распределение дробной и целой частей. Поэтому такой способ пред-  
ставления чисел называется с ***фиксированной запятой.***

Представление числа в нормальной форме называют представле-  
ние с ***плавающей запятой*** (положение запятой меняется).

С числами, представленными в форме с плавающей запятой, ра-  
ботают в основном универсальные ЭВМ, а специализированные  
ЭВМ — с фиксированной запятой, но целый ряд машин работает  
с числами в этих двух форматах.

Характер программирования зависит от способа представления  
чисел. Так, при написании программ для ЭВМ, работающих в системе

с фиксированной запятой, необходимо отслеживание положения за-  
пятой, а для выполнения операций с плавающей запятой требуется  
большее число микроопераций, что снижает быстродействие ЭВМ.

**Фиксированная запятая (точка).** В современных ЭВМ способ пред-  
ставления чисел с фиксированной запятой в вычислительной технике  
используется преимущественно для представления целых чисел.

Так как числа бывают положительными и отрицательными, то  
в разрядной сетке при их машинном представлен™ один или два раз-  
ряда (для модифицированных кодов) отводятся под знак числа,  
а остальные разряды образуют поле числа. В знаковые разряды, кото-  
рые могут располагаться как в начале, так и в конце числа, записыва-  
ется информация о знаке числа. Знак «+» кодируется нулем, знак «—»  
кодируется единицей. Для модифицированных кодов знак «+» коди-  
руется двумя нулями, знак «—» кодируется двумя единицами. Моди-  
фицированные коды введены для обнаружения неправильного резуль-  
тата вычислений, т.е. когда результат превышает максимальный раз-  
мер разрядной сетки и необходим перенос из значащего разряда.

Например, в результате выполнения операций в знаковом разряде  
число «01» свидетельствует о положительном переполнении разряд-  
ной сетки, а число «10» — об отрицательном переполнении разряд-  
ной сетки.

Поле числа имеет постоянное число разрядов — ***п.*** Диапазон  
представления целых чисел ограничивается значениями — ***(2п*** — 1)  
и + (2я — 1).

Например, в двоичном коде, используя 6-разрядную сетку,  
число 7 в форме с фиксированной запятой можно представить в виде

0**.**00111**2,**

где цифра левее точки — это знак числа, а пять цифр правее точки —  
мантисса числа в прямом коде.

Здесь подразумевается, что запятая фиксирована правее младшего  
разряда, а точка в изображении числа в данном случае просто разде-  
ляет знаковый бит от мантиссы числа. В дальнейшем часто будет  
использоваться в примерах такой вид представления числа в машин-  
ной форме. Можно использовать и другую форму представления  
числа в машинной форме:

**[**0**]**00111**2,**

где знаковый разряд выделяется квадратными скобками.

Количество разрядов в разрядной сетке, отведенное для изобра-  
жения мантиссы числа, определяет диапазон и точность представле-

55

Для случая, когда запятая фиксируется правее младшего разряда  
мантиссы, т.е. для целых чисел, числа, у которых модуль больше, чем  
(2(в 11 - 1) и меньше единицы, не представляются в форме с фикси-  
рованной запятой. Числа по абсолютной величине меньше единицы  
младшего разряда разрядной сетки называются в этом случае машин-  
ным нулем. Отрицательный ноль запрещен.

В некоторых случаях, когда можно оперировать только модулями  
чисел, вся разрядная сетка, включая самый старший разряд, отво-  
дится для представления числа, что позволяет расширить диапазон  
изображения чисел.

**Представление отрицательных чисел в формате с фиксированной  
запятой.** В компьютерах в целях упрощения выполнения арифмети-  
ческих операций применяются специальные двоичные коды для  
представления отрицательных чисел: ***обратный*** и ***дополнительный.***При помощи этих кодов упрощается определение знака результата  
операции при алгебраическом сложении. Операция вычитания (или  
алгебраического сложения) сводится к арифметическому сложению  
операндов, облегчается выработка признаков переполнения разряд-  
ной сетки. В результате упрощаются устройства компьютера, выпол-  
няющие арифметические операции.

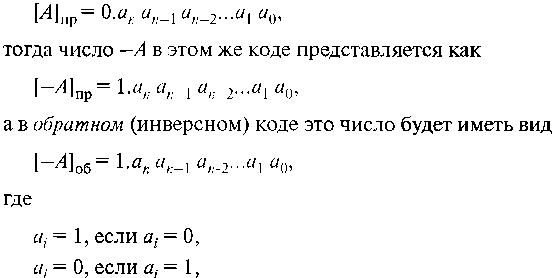
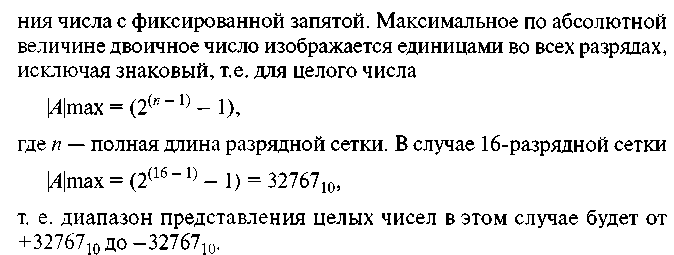
Известно, что одним из способов выполнения операции вычита-  
ния является замена знака вычитаемого на противоположный и при-  
бавление его к уменьшаемому:



Этим операцию арифметического вычитания заменяют опера-  
цией алгебраического сложения, которую можно выполнить при  
помощи двоичных сумматоров.

Для машинного представления отрицательных чисел используют  
коды прямой, дополнительный, обратный. Упрощенное определение  
этих кодов может быть дано следующим образом. Если число  
***А*** в обычном двоичном коде — ***прямом*** двоичном коде изобразить как

*56*



***at*** — цифра г-го разряда двоичного числа. Следовательно, при пере-  
ходе от прямого кода к обратному все цифры разрядов мантиссы  
числа инвертируются.

Тогда число ***—А*** в ***дополнительном*** коде изображается в виде

[—А]доп [—А]об + 1.

"

Таким образом, для получения дополнительного кода отрицатель-  
ных чисел нужно сначала инвертировать цифровую часть исходного  
числа, в результате чего получается его обратный код, а затем доба-  
вить единицу в младший разряд цифровой части числа.

Дополнительный код некоторого числа получается его заменой  
на новое число, дополняющее его до числа, равного весу разряда,  
следующего за самым старшим разрядом разрядной сетки, исполь-  
зуемой для представления мантиссы числа в формате с фиксирован-  
ной запятой. Поэтому такой код числа называется дополнительным.

Представим, что мы имеем только два разряда для представления  
чисел в десятичной системе счисления. Тогда максимальное число,  
которое можно изобразить, будет 99, а вес третьего несуществующего  
старшего разряда будет 102, т.е. 100. В таком случае для числа 20 до-  
полнительным будет число 80, которое дополняет 20 до 100 (100 —  
- 20 = 80). Следовательно, по определению вычитание

50 - 20 = 30

можно заменить на сложение:

50 + 80 = 1 30.

Здесь старшая единица выходит за пределы выделенной разряд-  
ной сетки, в которой остается только число 30, т.е. результат вычи-  
тания из 50 числа 20.

*57*

А теперь рассмотрим похожий пример для чисел, представленных  
4-разрядным двоичным кодом. Найдем дополнительное число для  
00 1 02 = 210. Надо из [1]0000 вычесть [0]0010, получим [0] 1110, которое  
и является дополнительным кодом 2. Разряд, изображенный в квад-  
ратных скобках, на самом деле не существует. Но так как у нас 4-  
разрядная сетка, то выполнить такое вычитание в принципе невоз-  
можно, а тем более, мы стараемся избавиться от вычитания. Поэтому  
дополнительный код числа получают способом, описанным ранее,  
т.е. сначала получают обратный код числа, а затем прибавляют  
к нему единицу. Проделав все это с нашим числом (2), нетрудно убе-  
диться, что получится аналогичный ответ.

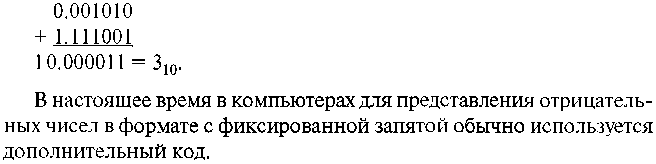
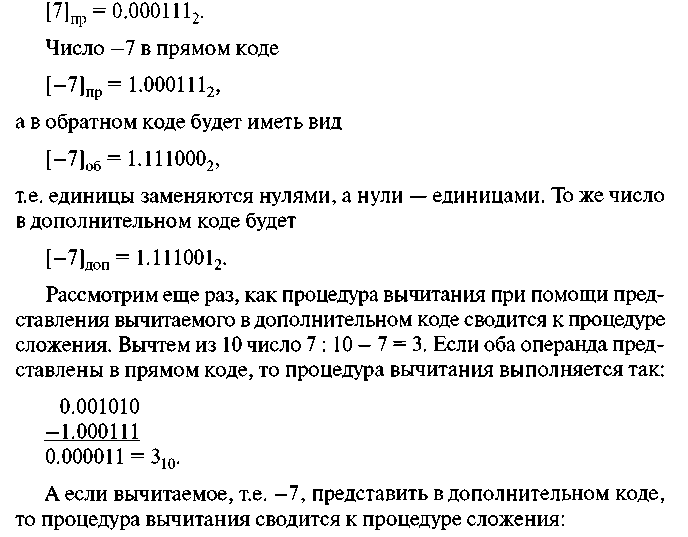
Подчеркнем, что дополнительный и обратный коды используются  
только для представления отрицательных двоичных чисел в форме  
с фиксированной запятой. Положительные числа в этих кодах не ме-  
няют своего изображения и представляются, как в прямом коде.

Таким образом, цифровые разряды отрицательного числа в пря-  
мом коде остаются неизменными, а в знаковой части записывается  
единица.

Рассмотрим простые примеры.

Семерка в прямом коде представляется так:

*58*



***Вещественные числа***

Числовые величины, которые могут принимать любые значения  
(целые и дробные), называются вещественными числами.

Вещественные числа в памяти компьютера представляются  
в форме с плавающей точкой. Форма с плавающей точкой использует  
представление вещественного числа ***R*** в виде произведения ман-  
тиссы ***т*** на основание системы счисления ***р*** в некоторой целой сте-  
пени и, которую называют порядком:

***R = т ■ рп.***

Например, число 25,324 можно записать в таком виде: 0.25324 • 102.

Здесь ***т*** = 0.25324 — мантисса; ***п = 2 —*** порядок. Порядок указы-  
вает, на какое количество позиций и в каком направлении должна  
«переплыть», т.е. сместиться, десятичная точка в мантиссе. Отсюда  
название «плавающая точка».

Однако справедливы и следующие равенства:

25,324 = 2,5324 • 101 = 0,0025324 • 104 = 2532,4 • 10 2 и т.п.

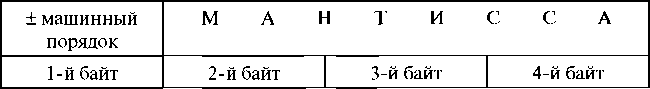
Получается, что представление числа в форме с плавающей точ-  
кой неоднозначно? Чтобы не было неоднозначности, в ЭВМ ***исполь-  
зуют нормализованное представление числа в форме с плавающей точ-  
кой.*** Мантисса в нормализованном представлении должна удовле-  
творять условию

0,1***„<=т< \р.***

Иначе говоря, мантисса меньше единицы и первая значащая  
цифра — не ноль. Значит, для рассмотренного числа нормализован-  
ным представлением будет 0.25324 • 102. В разных типах ЭВМ при-  
меняются различные варианты представления чисел в форме с пла-  
вающей точкой. Для примера рассмотрим один из возможных. Пусть  
в памяти компьютера вещественное число представляется в форме  
с плавающей точкой в двоичной системе счисления ***(р =*** 2) и зани-  
мает ячейку размером 4 байт. В ячейке должна содержаться следу-

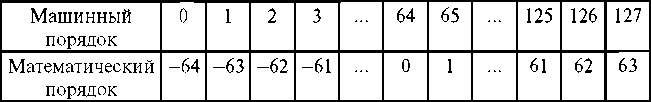
59

ющая информация о числе: знак числа, порядок и значащие цифры  
мантиссы. Вот как эта информация располагается в ячейке:



В старшем бите 1-го байта хранится знак числа. В этом разряде О  
обозначает плюс, 1 — минус. Оставшиеся 7 бит первого байта содер-  
жат машинный порядок. В следующих трех байтах хранятся знача-  
щие цифры мантиссы.

В семи двоичных разрядах помещаются двоичные числа в диапа-  
зоне от 0000000 до 1111111, В десятичной системе это соответствует  
диапазону от 0 до 127, всего 128 значений. Знак порядка в ячейке не  
хранится. Но порядок, очевидно, может быть как положительным,  
так и отрицательным. Разумно эти 128 значений разделить поровну  
между положительными и отрицательными значениями порядка.  
В таком случае между машинным порядком и истинным (назовем  
его математическим) устанавливается следующее соответствие:



Если обозначить машинный порядок ***Мр,*** а математический ***— р,***то связь между ними выразится такой формулой:

***Мр =р*** + 64.

Итак, машинный порядок смещен относительно математического  
на 64 единицы и имеет только положительные значения. При выпол-  
нении вычислений с плавающей точкой процессор это смещение  
учитывает.

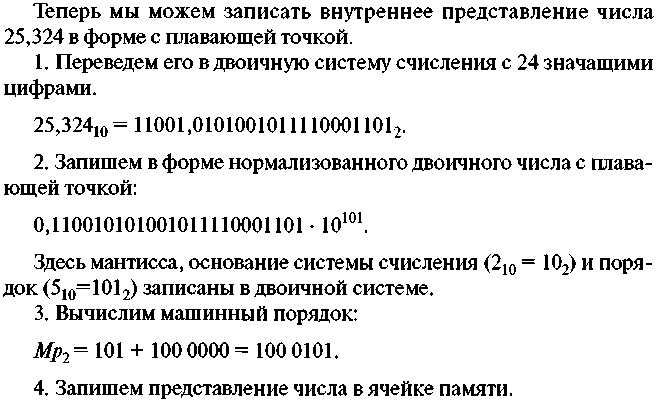
Полученная формула записана в десятичной системе. Поскольку  
6410=4016 (проверьте!), то в шестнадцатеричной системе формула  
примет вид

***МР\6 =Pi6*** + 401б-

И наконец, в двоичной системе:

*Мр2 =* /ъ + 10000002.

***60***



Это и есть искомый результат. Его можно переписать в более ком-  
пактной шестнадцатеричной форме:



Для того чтобы получить внутреннее представление отрицатель-  
ного числа —25,324, достаточно в полученном выше коде заменить  
в разряде знака числа 0 на 1.

Получим:



А в шестнадцатеричной форме:



Никакого инвертирования, как для отрицательных чисел с фик-  
сированной точкой, здесь не происходит.

Рассмотрим, наконец, вопрос о диапазоне чисел, представимых  
в форме с плавающей точкой. Очевидно, положительные и отрица-  
тельные числа расположены симметрично относительно нуля. Сле-  
довательно, максимальное и минимальное числа равны между собой

*61*

по модулю: Лтах = |Лт|п|. Наименьшее по абсолютной величине число  
равно нулю. Чему же равно Л1пах? Это число с самой большой ман-  
тиссой и самым большим порядком:

0,111111111111111111111111 • 1021ШШ

Если перевести в десятичную систему, то получится

Ятах=(1 - 2-24).264= 1019

Очевидно, что диапазон вещественных чисел значительно шире  
диапазона целых чисел. Если в результате вычислений получается  
число, по модулю большее, чем i?max, то происходит прерывание ра-  
боты процессора. Такая ситуация называется переполнением при  
вычислениях с плавающей точкой. Наименьшее по модулю ненуле-  
вое значение равно

(1/2) • 2~64= 2~66.

Любые значения, меньшие данного по абсолютной величине,  
воспринимаются процессором как нулевые.

Как известно из математики, множество действительных чисел  
бесконечно и непрерывно. Множество же вещественных чисел,  
представимых в памяти ЭВМ в форме с плавающей точкой, является  
ограниченным и дискретным. Каждое следующее значение получа-  
ется прибавлением к мантиссе предыдущего единицы в последнем  
(24-м) разряде. Количество вещественных чисел, точно представи-  
мых в памяти машины, вычисляется по формуле

***N=2'-(U-L+*** 1) + 1,

где ***t*** — количество двоичных разрядов мантиссы; ***U*** — максимальное  
значение математического порядка; ***L*** — минимальное значение по-  
рядка.

Для рассмотренного нами варианта ***(t =*** 24, ***U=*** 63, ***L =*** —64) по-  
лучается

***N=*** 2146683548.

Все же остальные числа, не попадающие в это множество, но на-  
ходящиеся в диапазоне допустимых значений, представляются в па-  
мяти приближенно (мантисса обрезается на 24-м разряде). А по-  
скольку числа имеют погрешности, то и результаты вычислений  
с этими числами также будут содержать погрешность. Из сказанного  
следует вывод: вычисления с вещественными числами в компьютере  
выполняются приближенно.

***62***