

Тема 2. АРИФМЕТИЧЕСКИЕ ОСНОВЫ ЭВМ

Содержание

1. Системы счисления.....	1
1.1. Непозиционные и позиционные системы счисления	1
1.2. Представления чисел в позиционных системах счисления.....	3
1.3. Выбор оптимальных систем счисления.....	6
2. Выполнение арифметических операций в различных системах счисления	7
2.1. Выполнение арифметических операций в двоичной системе счисления.....	7
2.2. Выполнение арифметических операций в восьмеричной и шестнадцатеричной системах счисления	9
3. Перевод чисел из одной позиционной системы счисления в другую	12
3.1. Основные способы перевода чисел.....	12
3.2. Перевод целых чисел.....	16
3.3. Перевод правильной дроби.....	17
3.4 Перевод произвольных чисел.....	18
3.5 Смешанная система счисления	19
4. Формы представления чисел в ЭВМ.....	20
4.1 Представление чисел в форме с фиксированной точкой.....	21
4.2. Представление чисел в форме с плавающей запятой.....	23
5. Арифметические операции в ЭВМ	28
5.1. Машинные коды	29
5.2. Арифметические операции над числами с фиксированной точкой	30
5.3. Арифметические операции над двоичными числами с плавающей точкой.....	35
5.4. Арифметические операции над двоично-десятичными кодами чисел	37

Рассмотрим арифметические основы ЭВМ, опирающиеся на такие понятия, как **системы счисления и способы представления цифровой информации**.

1. Системы счисления

1.1. Непозиционные и позиционные системы счисления

Система счисления – совокупность приемов и правил изображения чисел цифровыми знаками. Системы счисления делятся на непозиционные и позиционные.

Непозиционная система счисления – система, в которой значение символа не зависит от его положения в числе. Они использовались в древности римлянами, египтянами, славянами и другими народами. Примером непозиционной системы счисления, дошедшей до наших дней, служит римская система счисления.

Цифры в римской системе обозначаются различными знаками: 1-I; 2-II; 3-III; 5-V; 10-X; 50-L; 100-C; 500-D; 1000-M. Для записи промежуточных чисел существует правило: каждый меньший знак, поставленный справа от большего, прибавляется к его значению, а слева – вычитается из него. Так, IV обозначает 4, VI

– 6, XL – 60, XC – 90 и т.д. Основной недостаток непозиционных систем – большое число различных знаков и сложность выполнения арифметических операций.

Непозиционная система счисления – система, в которой значение символа зависит от его места в ряду цифр, изображающих число.

Для позиционных систем счисления характерно представление числа в виде последовательности цифр $X = x_1x_2x_3...x_k...x_n$, в которой значение каждой цифры x_k зависит от места ее расположения в последовательности.

Например, в числе 7382 первая цифра слева означает количество тысяч, вторая – количество сотен, третья – количество десятков и четвертая – количество единиц. Позиционные системы счисления более удобны для вычислительных операций, поэтому они и получили наибольшее распространение. Позиционная система счисления характеризуется основанием.

Основание (базис) системы счисления – с одной стороны, определяет количество знаков или символов, используемых в разрядах для изображения числа в данной системе счисления, а с другой – число, показывающее, во сколько раз вес цифры данного разряда меньше веса цифры соседнего старшего разряда.

Для позиционной системы счисления с одним основанием справедливо равенство

$$X(q) = a_n q^n + a_{n-1} q^{n-1} + \dots + a_1 q^1 + a_0 q^0 + a_{-1} q^{-1} + \dots + a_{-m} q^{-m} = \sum_{i=-m}^{i=n} a_i q^i \quad (1)$$

где q – основание позиционной системы счисления – целое положительное число; $X(q)$ – произвольное число, записанное в системе счисления с основанием q ; a_i – коэффициент ряда (цифры системы счисления); n, m – количество целых и дробных разрядов.

На практике используют сокращенную запись чисел, т.е.

$$X(q) = a_n a_{n-1} \dots a_1 a_0 a_{-1} \dots a_{-m}$$

Возможно множество позиционных систем, так как за основание можно принять любое целое число.

Так, базис десятичной системы счисления составляют цифры 0, 1, 2, 3, 4, 5, 6, 7, 8 и 9. Всякое число, заданное в позиционной системе счисления, в соответствии с (1), может быть представлено в виде полинома, например

$$7942 = 7 \cdot 10^3 + 9 \cdot 10^2 + 4 \cdot 10^1 + 2 \cdot 10^0$$

$$7942,561 = 7 \cdot 10^3 + 9 \cdot 10^2 + 4 \cdot 10^1 + 2 \cdot 10^0 + 5 \cdot 10^{-1} + 6 \cdot 10^{-2} + 1 \cdot 10^{-3}$$

Основанием системы счисления может быть любое натуральное число, большее 1. Возьмем в качестве основания системы счисления число 2. Тогда базис будет включать цифры 0 и 1, и все числа могут состоять только из нулей и единиц. Числа 1, 2, 3, 4 и 5, заданные в десятичной системе счисления будут выглядеть так: 1, 10, 11, 100 и 101. В троичной системе счисления (базис включает цифры 0, 1 и 2) числа могут содержать только цифры 0, 1 и 2 и эти же числа запишутся так 1, 2, 10, 11, 12. Нетрудно заметить, что роль “десятки” играет основание системы

счисления.

Для представления в ЭВМ любого n -разрядного числа (без учета знака) необходимо либо n физических элементов с q устойчивыми состояниями, либо $n \cdot k_x$ элементов с двумя устойчивыми состояниями, где k_x – минимально необходимое число двоичных разрядов, требующихся для кодирования любой цифры числа с основанием q , выбирается из условия $q \leq 2^{k_x}$.

Для первого случая, если рассматриваемые физические элементы для любого числа устойчивых состояний одинаковы по своим основным параметрам (надежность, быстродействие, габариты, стоимость), то наиболее рациональной системой счисления с точки зрения минимума оборудования для представления числа в ЭВМ будет система с большим основанием q . Однако в настоящее время элементы с более чем двумя устойчивыми состояниями (декатроны, трохотроны и др.) имеют существенные недостатки по указанным выше основным параметрам, поэтому в ЭВМ в основном используются **двоичные** и так называемые **двоично-кодированные системы счисления**, а не привычная десятичная система счисления.

В вычислительной технике наибольшее применение получили **двоичная, четверичная, восьмеричная и шестнадцатеричная системы счисления**. Основания различных систем счисления обычно называют в десятичной системе счисления. *В теоретическом отношении все системы счисления равноправны. Лишь на практике отдельным системам счисления отдаётся предпочтение для тех или иных применений. Во всех системах счисления по одним и тем же правилам выполняются арифметические операции, справедливы одни и те же законы: ассоциативный, дистрибутивный и др.*

Кратко рассмотрим представления чисел в указанных системах счисления.

1.2. Представления чисел в позиционных системах счисления

В **десятичной системе счисления** основание $q=10$; любое целое число записывается как сумма величин $10^0, 10^1, 10^2$ и т. д., каждая из которых может быть взята 1–9 раз. Число 10 изображается цифрами 1 и 0.

Например, последовательность цифр 4627,31, изображающая число в десятичной системе счисления, представляет собой сокращенную запись выражения $4 \cdot 10^3 + 6 \cdot 10^2 + 2 \cdot 10^1 + 7 \cdot 10^0 + 3 \cdot 10^{-1} + 1 \cdot 10^{-2}$.

В **двоичной системе счисления** для записи чисел используются две цифры: 0 и 1. Основание системы $q=2$ записывается как $10_2 = [1 \cdot 2^1 + 0 \cdot 2^0]_{10}$. В данной системе любое число может быть представлено последовательностью двоичных цифр. Эта запись соответствует сумме степеней цифры 2, взятых с указанными в ней коэффициентами:

$$X = a_m \cdot 2^m + a_{m-1} \cdot 2^{m-1} + \dots + a_1 \cdot 2^1 + a_0 \cdot 2^0 + a_{-1} \cdot 2^{-1} + a_{-2} \cdot 2^{-2} + \dots$$

Например, двоичное число

$$(10101101,101)_2 = (1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 +$$

$$+1*2^2+0*2^1+1*2^0+1*2^{-1}+0*2^{-2}+1*2^{-3})_{10}=173,625_{10}.$$

В **троичной системе счисления** для записи чисел используют цифры 0, 1, 2.

Например, $2122_{(3)}=(2*3^3+1*3^2+2*3^1+2*3^0)$.

Значения шестнадцати целых чисел в системах соответственно с основаниями $q=2, 4, 8$ и 16 приведены в таблице 1.

В системе счисления с основанием $q=4$ используют цифры 0–3; с $q=8$ – цифры 0–7; с $q=16$ – цифры 0–9 и буквы А, В, С, Д, Е, F. Здесь А, В, С, D, Е, F обозначают соответственно цифры 10, 11, 12, 13, 14, 15. Например, число 159_{10} в системах счисления с $q=4, 8, 16$ будет иметь вид $159_{10}=2133_4=[2*4^3+1*4^2+3*4^1+3*4^0]_{10}$;

$$159_{10}=[237_8=2*8^2+3*8^1+7*8^0]_{10}; 159_{10}=9F_{16}=[9*16^1+F*16^0]_{10}.$$

Вес разряда p_i числа в позиционной системе счисления есть отношение вида $p_i=q_i/q_0=q_i$, где i -номер разряда справа налево.

Если разряд имеет вес $p_i=q_i$, то следующий старший разряд будет иметь вес $p_{i+1}=q_{i-1}$. Таким образом, в позиционной системе счисления вес разряда определяется его положением (позицией) в числе.

Таблица 1.

Система счисления				
Десятичная	Двоичная	Четверичная	Восьмеричная	Шестнадцатеричная
0	0	0	0	0
1	1	1	1	1
2	10	2	2	2
3	11	3	3	3
4	100	10	4	4
5	101	11	5	5
6	110	12	6	6
7	111	13	7	7
8	1000	20	10	8
9	1001	21	11	9
10	1010	22	12	A
11	1011	23	13	B
12	1100	30	14	C
13	1101	31	15	D
14	1110	32	16	E
15	1111	33	17	F

Ниже приведены веса разрядных позиций в десятичной системе

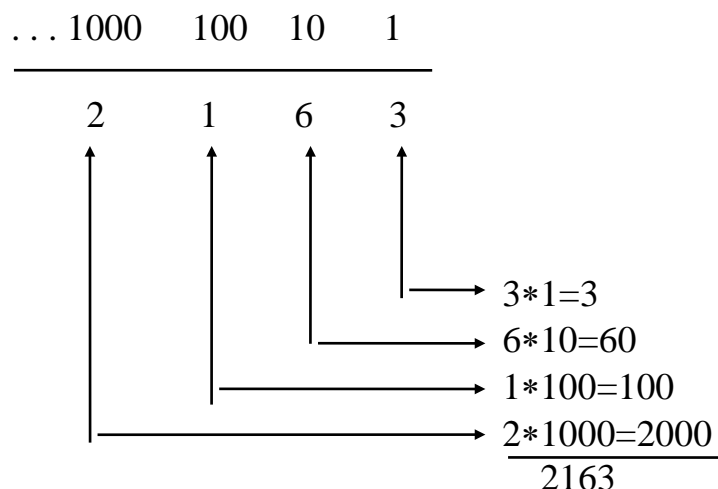
счисления:

Веса цифровой

позиции

Пример десятичного

числа



Здесь вес последующего разряда в 10 раз больше веса предыдущего разряда. Например, цифра 6 в приведенном примере имеет значение 60, так как она расположена во втором справа разряде (позиции) числа. Такая взаимосвязь разрядов приводит к необходимости передачи информации между ними. Если при сложении в данном разряде накопилось значение, равное или большее q , то должна происходить передача единицы в соседний старший разряд с уменьшением его на q , т.е. передача единицы переноса. Если при вычитании в данном разряде число единиц оказалось меньше нуля, то должна происходить передача единицы займа соседнего старшего разряда с увеличением его на q для данного разряда. Передача переносов или займов происходит последовательно от разряда к разряду.

Длина числа – количество разрядов (позиций) в записи числа.

Длина разрядной сетки – термин, используемый для определения длины числа. В разных системах счисления длина разрядной сетки при записи одного и того же числа неодинаковая. Например, $96_{10} = 140_8 = 10120_3 = 1100000_2$. Из примера видно, что одно и то же число, записанное в разных системах счисления, имеет разную длину разрядной сетки. Чем меньше основание системы, тем больше длина разрядной сетки. Предположим, что длина разрядной сетки равна какому-то положительному числу n , тогда:

$$X_{\max} = q^n - 1.$$

Диапазон представления (ДП) чисел в заданной системе счисления – интервал числовой оси, заключенный между максимальными и минимальными числами, представленными длиной разрядной сетки, т. е.

$$X_{\max} \geq \text{ДП} \geq X_{\min}. \text{ Обычно } X_{\min} = 0.$$

1.3. Выбор оптимальных систем счисления

От того, какая система счисления будет использована в ЭВМ, зависят скорость вычислений, емкость памяти, сложность алгоритмов выполнения арифметических операций. При выборе системы счисления учитывается зависимость длины числа и количества устойчивых состояний функциональных элементов (для изображения цифр) от основания системы счисления. Например, при десятичной системе счисления функциональный элемент должен иметь десять устойчивых состояний, а при двоичной системе счисления – два. Кроме того, система счисления должна обладать простотой выполнения арифметических и логических операций.

Десятичная система счисления, привычная для нас в повседневной жизни, не является наилучшей для использования в ЭВМ. Это объясняется тем, что известные в настоящее время функциональные элементы с десятью устойчивыми состояниями (элементы на основе сегнетокерамики, декастроны и др.) имеют низкую скорость переключения и, таким образом, не могут удовлетворять требованиям, предъявляемым к ЭВМ по быстродействию. Поэтому в большинстве случаев в ЭВМ используют двоичные или двоично-кодированные системы счисления. Широкое распространение этих систем обусловлено тем, что элементы ЭВМ способны находиться лишь в одном из двух устойчивых состояний. Например, полупроводниковый транзистор в режиме переключения может быть в открытом или закрытом состоянии, а, следовательно, иметь на выходе высокое или низкое напряжение. Ферритовый сердечник в устойчивом состоянии может иметь положительную или отрицательную остаточную магнитную индукцию. Такие элементы принято называть двухпозиционными. Если одно из устойчивых положений элемента принять за 0, а другое – за 1, то достаточно просто изображаются разряды двоичного числа.

Арифметические операции над двоичными числами отличаются простотой и легкостью технического выполнения.

Правила двоичной арифметики:

Сложение:

$$\begin{array}{ll} 0+0=0 & 1+0=1 \\ 0+1=1 & 1+1=\boxed{1}0 \end{array}$$

└───────────> перенос единицы в старший разряд

Вычитание:

$$\begin{array}{ll} 0-0=0 & 1-1=0 \\ 1-0=1 & \boxed{1}0-1=1 \end{array}$$

└───────────> заем единицы в старшем разряде

Умножение:

$$\begin{array}{ll} 0\times 0=0 & 1\times 0=0 \\ 0\times 1=0 & 1\times 1=1 \end{array}$$

Двоичная система счисления является основной для использования в ЭВМ, удобной из-за простоты выполнения арифметических операций над двоичными числами. С точки зрения затрат оборудования на создание ЭВМ эта система уступает только троичной системе счисления.

В двоично-кодированных системах счисления, имеющих основание q , отличное от 2 ($q > 2$), каждая цифра числа представляется в двоичной системе счисления. Наибольшее применение в ЭВМ получили **шестнадцатеричная и десятичная двоично-кодированные системы счисления.**

2. Выполнение арифметических операций в различных системах счисления

Рассмотрим выполнение арифметических операций в различных системах счисления. Для этого следует составить таблицы сложения и умножения в соответствующей системе счисления.

2.1. Выполнение арифметических операций в двоичной системе счисления

Для двоичной системы счисления правила выполнения арифметических операций над двоичными числами остаются такими же, как и в привычной для всех десятичной системе счисления.

Основой выполнения арифметических операций являются следующие таблицы сложения, вычитания и умножения одnorазрядных чисел:

Таблица сложения

	0	1
0	0	1
1	1	10 ← единица переноса в старший разряд

Таблица вычитания

0	0	1
0	0	1 → $10 - 1 = 1$
1	1	0

↑ с учетом заема единицы в старшем разряде

Таблица умножения

	0	1
0	0	0
1	0	1

Сложение двух чисел в двоичной системе можно выполнить столбиком, начиная с младших разрядов. При этом в каждом разряде складываются две цифры одноименных разрядов (в соответствии с таблицей сложения) и единицы переноса из соседнего младшего разряда, если он имел место. В результате сложения получим цифру соответствующего разряда суммы и возможную единицу переноса в старший соседний разряд.

Вычитание чисел, как и сложение, также выполняется столбиком (в соответствии с таблицей вычитания). Особым случаем является тот, когда необходимо занимать единицу из соседнего старшего разряда, которая равна двум единицам данного разряда.

Умножение двоичных многоразрядных чисел осуществляется последовательным сложением частичных произведений, каждое из которых (в соответствии с таблицей умножения) равно множимому, сдвинутому на соответствующее число разрядов, если в разряде множителя стоит единица, или нулю, если в разряде множителя стоит 0.

Деление двоичных чисел производится аналогично делению десятичных чисел, но с учетом специфики операции вычитания двоичных чисел. Положение запятой результата умножения и деления определяется так же, как и для десятичных чисел.

Приведем примеры выполнения действий в двоичной системе счисления:

1. Сложение:

$$\begin{array}{r} 101100 \\ + \quad 1011 \\ \hline 110111 \end{array} \qquad \begin{array}{r} 101,1001 \\ + \quad 1,1111 \\ \hline 111,1000 \end{array}$$

2. Вычитание:

$$\begin{array}{r} 110111 \\ - \quad 1011 \\ \hline 101100 \end{array} \qquad \begin{array}{r} 111,1000 \\ - \quad 1,1111 \\ \hline 101,1001 \end{array}$$

3. Умножение:

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ + 101 \\ \hline 1111 \end{array}$$

4. Деление:

$$\begin{array}{r|l} 1111 & 11 \\ - 11 & 101 \\ \hline 11 & \\ - 11 & \\ \hline 0 & \end{array}$$

2.2. Выполнение арифметических операций в восьмеричной и шестнадцатеричной системах счисления

Восьмеричная и шестнадцатеричная системы счисления относятся к классу двоично-кодированных систем, так как основание этих систем представляют целые степени двойки: 2^3 – для восьмеричной и 2^4 – для шестнадцатеричной системы счисления.

Изображение целых чисел в восьмеричной и шестнадцатеричной системах счисления вместе с их двоичным и десятичным эквивалентами представлены в таблице 1 раздела 1.2.

Для систем счисления с основанием $q \leq 10$ для изображения цифровых символов, используются цифры от 0 до $(q - 1)$, а для $q > 10$ помимо цифр используются первые шесть букв латинского алфавита.

Большим достоинством восьмеричной и шестнадцатеричной систем счисления является, во-первых, возможность более компактно представить запись двоичного числа, а именно, запись одного и того же двоичного числа в восьмеричной и шестнадцатеричной системах будет соответственно в 3 и 4 раза короче двоичной. Во-вторых, сравнительно просто осуществляется преобразование чисел из двоичной в восьмеричную и шестнадцатеричную системы и наоборот. Действительно, так как для восьмеричного числа каждый разряд представляется группой из трех двоичных разрядов (триад), а для шестнадцатеричного – группой из четырех двоичных разрядов (тетрад), то для такого преобразования достаточно объединить двоичные цифры в группы по 3 и 4 бита соответственно, продвигаясь от отдельной запятой вправо и влево. При этом в случае необходимости добавлять нули в начале и

в конце числа и каждую такую группу – триаду или тетраду – заменяют эквивалентной восьмеричной или шестнадцатеричной цифрой.

Указанные достоинства восьмеричных и шестнадцатеричных систем счисления определили использование их при составлении программ для более короткой и удобной записи двоичных чисел, команд и специальных двоичных слов, с которыми оперирует ЭВМ. Особенно оказалось удобным использование шестнадцатеричной системы, когда разрядность чисел и команд выбрана кратной байту, при этом каждый двоичный код байта запишется в виде 2-разрядного шестнадцатеричного числа.

Использование шестнадцатеричной системы счисления в ЭВМ общего назначения, как будет видно из дальнейшего изложения, позволяет расширить допустимый диапазон представления нормализованных чисел.

Представим таблицы сложения и умножения для шестнадцатеричной системы счисления.

Таблица сложения в шестнадцатеричной системе счисления.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Таблица умножения в шестнадцатеричной системе счисления.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	–	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	–	–	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3	–	–	–	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	–	–	–	–	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	–	–	–	–	–	19	1E	23	28	2D	32	37	3C	41	46	4B
6	–	–	–	–	–	–	24	2A	30	36	3C	42	48	4E	54	5A
7	–	–	–	–	–	–	–	31	38	3F	46	4D	54	5B	62	69
8	–	–	–	–	–	–	–	–	40	48	50	58	60	68	70	78
9	–	–	–	–	–	–	–	–	–	51	5A	63	6C	75	7E	87
A	–	–	–	–	–	–	–	–	–	–	64	6E	78	82	8C	96
B	–	–	–	–	–	–	–	–	–	–	–	79	84	8F	9A	A5
C	–	–	–	–	–	–	–	–	–	–	–	–	90	9C	A8	B4
D	–	–	–	–	–	–	–	–	–	–	–	–	–	A9	B6	C3
E	–	–	–	–	–	–	–	–	–	–	–	–	–	–	C4	D2
F	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	E1

Очевидно, что таблица сложения симметрична относительно главной диагонали. Это утверждение справедливо и для таблицы умножения, поэтому в нее занесем только те элементы, которые расположены на главной диагонали и выше нее.

Приведем примеры действий в шестнадцатеричной системе счисления:

1. Сложение:

$$\begin{array}{r} A1F \\ + \quad 1 \\ \hline A20 \end{array}$$

$$\begin{array}{r} A19 \\ + \quad BC \\ \hline AD5 \end{array}$$

$$\begin{array}{r} A20 \\ + \quad F5 \\ \hline B15 \end{array}$$

2. Вычитание:

$$\begin{array}{r} AD5 \\ - A19 \\ \hline BC \end{array}$$

$$\begin{array}{r} B15 \\ - \quad F5 \\ \hline A20 \end{array}$$

3. Умножение:

$$\begin{array}{r} A0F \\ \times 1A \\ \hline 6496 \\ \hline A0F \\ \hline 10586 \end{array}$$

$$\begin{array}{r} FFA3 \\ \times DE \\ \hline DFAEA \\ \hline CFB47 \\ \hline DDAF5A \end{array}$$

4. Деление:

$$\begin{array}{r|l} DDAF5A & FFA3 \\ - CFB47 & DE \\ \hline DFAEA & \\ - DFAEA & \\ \hline 0 & \end{array}$$

Аналогично строятся таблицы и осуществляются арифметические операции и для других позиционных систем счисления.

3. Перевод чисел из одной позиционной системы счисления в другую

В дальнейшем изложении при записи чисел основание системы счисления будем записывать в виде нижнего и индекса справа от числа, например,

$$101_2 \qquad 83_{10} \qquad 2AF,5E_{16}$$

Существует несколько способов перевода чисел из одной системы счисления в другую. Наибольшее распространение получили два из них. Первый способ основывается на выполнении арифметических операций в новой системе счисления, второй – в старой.

3.1. Основные способы перевода чисел

Ниже рассматриваются указанные способы перевода чисел из одной системы в другую и формируются общие правила перевода чисел.

1-й способ.

Прежде чем дать формулу перевода в общем виде, рассмотрим несколько примеров.

Пример 1. Число 254_6 , заданное в шестеричной системе счисления, перевести перевод $6 \rightarrow 10$. Представим это число в виде полинома в соответствии с формулой (1):

$$254_6 = 2*6^2 + 5*6^1 + 4*6^0,$$

в котором каждая цифра записана в новой системе счисления (в данном случае в десятичной). Выполним требуемые вычисления, получим:

$$254_6 = 106_{10}.$$

Пример 2. Число $254,24_8$, заданное в восьмеричной системе счисления, перевести в двоичную систему счисления, т.е. сделать перевод $8 \rightarrow 2$. Представим это число в виде полинома:

$$254,24_8 = 2*8^2 + 5*8^1 + 4*8^0 + 2*8^{-1} + 4*8^{-2},$$

в котором затем каждую цифру запишем в новой системе счисления (в данном случае в двоичной). Получим:

$$10*1000^{10} + 101*1000^1 + 100*1000^0 + 10*1000^{-1} + 100*1000^{-10}.$$

Выполнив вычисления в новой системе счисления, будем иметь

$$254,24_8 = 10101100,010100_2.$$

Так как арифметические операции определены однозначно во всех системах счисления, то каждому числу в заданной системе счисления Q соответствует одно и только одно число в новой системе счисления P .

Рассмотренный способ перевода числа из **Q-ичной** системы счисления в **P-ичную** сводится к следующим правилам:

- 1) записать переводимое число в виде полинома в старой системе счисления (Q);
- 2) в полученном полиноме заменить основание Q и все коэффициенты числами в новой системе счисления;
- 3) выполнить арифметические операции в новой системе счисления.

Пример 3. Дано число $9B3_{17}$. Представить его в десятичной системе счисления:

$$9B3_{17} = 9 \cdot Q^2 + B \cdot Q + 3 = 9 \cdot 17^2 + 11 \cdot 17 + 3 = 2791_{10}.$$

При необходимости перевода чисел из десятичной системы счисления в любую другую ($10 \rightarrow Q$) вычисления удобнее выполнять в десятичной системе счисления, поэтому первый способ перевода становится неудобным.

2-й способ.

Вычисления требуется выполнять в старой системе счисления. Сначала разберем несколько примеров.

Пример 4. Представить десятичное число 941 в шестнадцатеричной системе счисления. Искомое число имеет вид:

$$(q_n q_{n-1} \dots q_1 q_0)_{16}.$$

Задача сводится к определению q_i , принадлежащих базису шестнадцатеричной системы счисления. Используя представления числа в виде полинома, будем иметь:

$$941_{10} = q_n 16^n + q_{n-1} 16^{n-1} + \dots + q_1 16 + q_0.$$

Числа 941 и 16 заданы в десятичной системе счисления. Легко видеть, что при делении 941 на 16 полученный остаток, выраженный соответствующей цифрой базиса шестнадцатеричной системы счисления, равен последнему слагаемому нашего многочлена:

$$\begin{array}{r|l} 941 & 16 \\ \hline - 80 & 58 \\ \hline 141 & \\ - 128 & \\ \hline 13 & \end{array}$$

Следовательно,

$$q_0 = 13_{10} = D_{16} \text{ и}$$

$$941_{10} = (q_n 16^{n-1}_{10} + q_{n-1} 16^{n-2}_{10} + \dots + q_1) 16_{10} + q_0;$$

$$58_{10} = q_n 16^{n-1}_{10} + q_{n-1} 16^{n-2}_{10} + \dots + q_1.$$

Рассуждая аналогично, получим

$$\begin{array}{r|l} 58 & 16 \\ - 48 & 4 \\ \hline 10 & \end{array}$$

Следовательно, $q_1 = 10_{10} = A_{16}$; $q_2 = 4$.

Таким образом, $941_{10} = 4AD_{16}$.

Пример 5. Представить десятичное число 0,50390625 в шестнадцатеричной системе счисления. Искомое число будет иметь вид:

$$(0, q_{-1} q_{-2} \dots q_{-v} \dots)_{16}.$$

Наша задача сводится к определению q_i , принадлежащих базису шестнадцатеричной системы счисления. Используя представление числа в виде полинома, получаем:

$$0,50390625_{10} = q_{-1} 16^{-1}_{10} + q_{-2} 16^{-2}_{10} + \dots + q_{-v} 16^{-v}_{10} \dots$$

Умножив обе части этого равенства на 16, получим

$$16 * 0,50390625_{10} = q_{-1} + q_{-2} 16^{-1}_{10} + \dots + q_{-v} 16^{-v+1}_{10} \dots$$

Очевидно, что полученная при умножении 0,50390625 на 16 целая часть, замененная соответствующей цифрой базиса шестнадцатеричной системы счисления, равна q_{-1} , т.е. найдена первая цифра в представлении искомого числа. В данном случае:

$$\begin{array}{r} 0,50390625 \\ \times \quad 16 \\ \hline 3 \ 02343750 \\ + 5 \ 0390625 \\ \hline 8,06250000 \end{array}$$

и $q_{-1} = 8$.

Для получения следующих цифр поступаем аналогично:

$$\begin{array}{r}
 0,0625 \\
 \times 16 \\
 \hline
 3750 \\
 + 625 \\
 \hline
 1,0000
 \end{array}$$

Таким образом $q_{-2}=1$, и окончательно получаем:

$$0,50390625_{10}=0,81_{16}.$$

При переводе целых чисел выполнялись операции деления, а при переводе дробной части – операции умножения, следовательно, при переводе смешанных чисел следует отдельно переводить целую и дробную части числа.

Пример 6. Представить десятичное число 291,25 в восьмеричной системе исчисления.

1. Переведем целую часть числа в восьмеричную систему счисления:

$$\begin{array}{r|l}
 291 & 8 \\
 \hline
 -24 & 36 \\
 \hline
 51 & -32 & 8 \\
 \hline
 -48 & 4 & 4 \\
 \hline
 3 & &
 \end{array}$$

Получим $291_{10}=443_8$.

2. Переведем дробную часть в восьмеричную систему счисления:

$$\begin{array}{r}
 0,25 \\
 \times 8 \\
 \hline
 2,00
 \end{array}$$

Получим $0,25_{10}=0,2_8$.

3. В итоге будем иметь $291_{10}=443_8$.

Данный способ не всегда позволяет при переводе дробной части получить конечную величину. Например, при переводе десятичного числа 0,3 в восьмеричную систему счисления последовательно получаем

$$\begin{array}{r} 0,3 \\ \times 8 \\ \hline 2,4 \end{array} \quad \begin{array}{r} 0,4 \\ \times 8 \\ \hline 2,4 \end{array} \quad \begin{array}{r} 0,2 \\ \times 8 \\ \hline 1,6 \end{array} \quad \begin{array}{r} 0,6 \\ \times 8 \\ \hline 4,8 \end{array} \quad \begin{array}{r} 0,8 \\ \times 8 \\ \hline 6,4 \end{array} \quad \begin{array}{r} 0,4 \\ \times 8 \\ \hline 3,2 \end{array}$$

т.е. имеем периодическую дробь $0,3_{10}=0,2(3146)_8$.

Рассмотрим второй способ перевода чисел из системы счисления с основанием P ($Q \rightarrow P$) в общем виде. Так как при переходе от одной системы счисления к другой следует переводить отдельно целую и дробную части, то опишем их перевод отдельно

3.2. Перевод целых чисел

Пусть дано целое число $N_q \geq 0$. Требуется перенести в P -ичную систему счисления, т.е. найти $N_p = (p_s \ p_{s-1} \ . \ . \ . \ p_1 \ p_0)$, для чего определить p_i , принадлежащие базису $(0, 1, 2, \dots, P-1)$; i изменяется от 0 до s .

Перевод целого из Q -ичной системы счисления в P -ичную сводится к следующим правилам.

1. Данное число разделить на основание (P) новой системы счисления. Остаток от деления записать в новой системе счисления.

Это младшая цифра (p_0) искомого числа.

2. Если частное равно 0, то искомое число получено.

3. Если частное не равно 0, то разделить его на P . Остаток от деления записать в новой системе счисления. Это предыдущая цифра искомого числа. Процесс получения цифр продолжать до тех пор, пока частное не станет равным нулю.

Пример 7.

Дано число 2791_{10} в десятичной системе счисления. Представить его в семнадцатеричной системе счисления.

$$\begin{array}{r} 2791 \big| 17 \\ -17 \quad \quad \quad 164 \\ \hline 109 \end{array} \quad \begin{array}{r} 164 \big| 17 \\ -153 \quad \quad \quad 9 \\ \hline 11=B \end{array} \quad \begin{array}{r} 9 \big| 17 \\ -0 \quad \quad \quad 9 \\ \hline 9 \end{array}$$

$$\begin{array}{r} -102 \\ \hline 71 \\ -68 \\ \hline 3 \end{array}$$

Получим $2791_{10}=9B3_{17}$.

3.3. Перевод правильной дроби

Пусть дано число X_Q ($0 < X_Q < 1$). Требуется перенести его в P -ичную систему счисления, т.е. найти $X_P = 0, p_{-1} p_{-2} \dots p_{-i} \dots$. Для этого определим p_i , принадлежащие базису $(0, 1, \dots, P-1)$; i принимает значения $-1, -2, \dots$

Перевод дроби из Q -ичной системы счисления в P -ичную сводится к следующим правилам.

1. Умножить данное число на P – основание новой системы счисления. Цифра, получившаяся в целой части произведения, является первой цифрой после запятой в искомом числе.

2. Дробную часть произведения умножить на P . Цифра, получившаяся в целой части произведения, является следующей цифрой в искомом числе.

Если дробная часть произведения равна нулю или достигнута заданная точность представления числа в новой системе счисления, то результат получен. Иначе повторить п. 2.

Пример 8.

Дано число 0.225_{10} . Перевести его в восьмеричную систему счисления, сохранив в результате четыре цифры после запятой:

$$0.225 * 8 = 1.800$$

$$0.800 * 8 = 6.4$$

$$0.4 * 8 = 3.2$$

$$0.2 * 8 = 1.6$$

$$0.6 * 8 = 4.8$$

$$\text{Получим } 0.225_{10} = 0.1631_8$$

Пример 9. Дробь 0.31_{10} перевести в двоичную систему счисления:

0.31	0.96
$\times \underline{2}$	$\times \underline{2}$
0.62	1.92
$\times \underline{2}$	$\times \underline{2}$
1.24	1.84
$\times \underline{2}$	$\times \underline{2}$
0.48	1.68
$\times \underline{2}$	
0.96	

в результате $0.31_{10} \approx 0.010011_2$

Из последних примеров 8 и 9 следует, что перевод дробей может представлять собой бесконечный процесс, а результат перевода – приближённый.

Число цифр в числе, представленном в системе счисления с основанием p , определяется из условия, что точность числа в системе счисления с основанием q .

Перевод двоичной дроби в десятичную можно осуществлять сложением всех цифр со степенями 2, соответствующими позициям разрядов исходной двоичной дроби, в которых цифры равны 1.

Пример 10. Дробь, 0,11012 перевести в десятичную систему счисления.

Представить исходное число в виде:

$$0.1101_2 = [1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4}]_{10} = [0.5 + 0.25 + 0.0625]_{10} = 0.8125_{10}$$

3.4 Перевод произвольных чисел

Числа, имеющие целую и дробные части, переводятся в два этапа: вначале целая часть, а затем дробная. В результате получаем число, записанное в новой системе счисления.

Пример 11. Перевести неправильную дробь 118.37510 и в двоичную систему счисления.

Для перевода в двоичную систему сначала переведём последовательным делением целую часть:

$$\begin{array}{r}
 118 \quad | \quad 2 \\
 \hline
 118 \quad 59 \quad | \quad 2 \\
 \hline
 0 \quad 58 \quad 29 \quad | \quad 2 \\
 \hline
 \quad 1 \quad 28 \quad 14 \quad | \quad 2 \\
 \hline
 \quad \quad 1 \quad 14 \quad 7 \quad | \quad 2 \\
 \hline
 \quad \quad \quad 0 \quad 6 \quad 3 \quad | \quad 2 \\
 \hline
 \quad \quad \quad \quad 1 \quad 2 \quad 1 \quad | \quad 2 \\
 \hline
 \quad \quad \quad \quad \quad 1 \quad 0 \quad 0 \\
 \hline
 \quad \quad \quad \quad \quad \quad 1
 \end{array}$$

В результат имеем:

$$118_{10} = 1110110_2$$

Далее переведем в двоичную систему счисления дробную часть методом последовательного умножения:

$$\begin{array}{r}
0,375 \\
\times 2 \\
\hline
0,750 \\
\times 2 \\
\hline
1,500 \\
\times 2 \\
\hline
1,000 \\
\times 2 \\
\hline
0,000
\end{array}$$

В результате имеем: $0,375_{10}=0,011_2$

Окончательный результат имеет вид: $118,375_{10}=1110110,011_2$

3.5 Смешанная система счисления

В смешанной системе счисления каждая цифра, заданного в Q-ичной системе счисления, заменяется соответствующим ее представлением в P-ичной счисления. Такая система называется P-Q-ичной.

Рассмотрим двоично-десятичную систему счисления, как пример смешанной системы счисления, где каждая цифра числа, представленного в десятичной системе счисления, заменяется соответствующим ее представлением в двоичной системе счисления. Двоично-десятичная система счисления нашла широкое применение во многих ЭВМ.

Например, десятичное число 931 в двоично-десятичной системе счисления запишется в виде:

$$1001 \quad 0011 \quad 0001_{2-10}.$$

Т. е. каждая цифра десятичного числа (от 0 до 9) заменяется тетрадой в двоичной системе счисления.

Рассмотрим еще один пример.

Пример. Число $273,59_{10}$ перевести в двоично-десятичную систему счисления.

Перевод осуществим следующим образом:

$$\begin{array}{ccccc}
2 & 7 & 3, & 5 & 9 \\
0010 & 0111 & 0011, & 0101 & 1001
\end{array}$$

т. е. $273,59_{10} = 001001110011,01011001_{2-10}.$

Двоично-десятичную запись числа используют непосредственно или как промежуточную форму записи между обычной десятичной его записью и машинной двоичной. Вычислительная машина сама по специальной программе переводит двоично-десятичные числа в двоичные и обратно.

Для смешанных систем счисления характерно, что количество разрядов, отводимых под каждую цифру, определяется максимальной цифрой базиса Q -ичной системы счисления. В десятичной системе счисления это цифра 9, для представления которой в двоичной системе требуется четыре разряда. Особый интерес представляет случай, когда $Q=P^m$, где m -целое число. В этом случае представление числа в P -ичной системе счисления совпадает с его представлением в смешанной P - Q -ичной системе счисления. Приводить доказательство этого утверждения мы не будем, покажем его только на примере.

Пример. Перевести восьмеричное число 741 в двоичную и двоично-восьмеричную системы счисления. Сначала переведем число 741 в двоичную систему счисления, получим:

$$741_8 = 111\ 100\ 001_2.$$

Затем переведем по отдельности каждую цифру числа 741 в двоичную систему счисления, получим:

$$741_8 = 111\ 100\ 001_2 = 111\ 100\ 001_{2-8}.$$

Как видим, представления числа в двоичной и двоично-восьмеричной системах счисления совпадают.

Отметим, что, приведенные способы перевода чисел из одной системы счисления в другую представляют собой достаточно трудоемкий процесс. Однако в современных ЭВМ такой перевод осуществляется самой машиной автоматически по стандартным подпрограммам. К ручному переводу обращаются в исключительных случаях.

4. Формы представления чисел в ЭВМ

В цифровых вычислительных машинах применяются две формы представления чисел:

- с фиксированной запятой (точкой);
- с плавающей запятой (точкой).

В зависимости от того, какой из этих двух способов принят для представления чисел, различают два режима работы цифровых вычислительных машин: с фиксированной запятой и с плавающей точкой. Рассмотрим подробнее сущность представления чисел в обеих формах.

4.1 Представление чисел в форме с фиксированной точкой

При работе ЭВМ в режиме с фиксированной запятой место запятой, отделяющей целую часть числа от дробной, остается постоянным для всех чисел. При конструировании машин, работающих в режиме с фиксированной запятой, заранее устанавливают, какое количество разрядов отводится для целой части числа, а какое – дробной части.

Поясним сказанное примером. Допустим, что цифровая машина рассчитана на представление шестиразрядного десятичного числа, причем три разряда отводятся на целую часть числа, а три – на дробную, или, как принято говорить, запятая фиксирует после третьего цифрового разряда. В этом случае в машине могут быть представлены следующие числа:

+999.999,
+999.998,
.....
+000.001,
000.000,
-000.001,
.....
-999.999.

Отметим сразу некоторые недостатки, присущие способу представления чисел с фиксированной запятой. В машине, работающей в режиме с фиксированной запятой, диапазон представляемых чисел, отличных от нуля, сравнительно невелик (в нашем примере – от 0.001 до 999.999). Всякое число, меньшее по абсолютной величине минимального положительного числа, представляемого машиной, будет записано в машине в виде нуля. Это так называемый машинный нуль. Кроме того, любое число, получающееся в результате вычислений, не должно превышать по абсолютной величине максимального числа, которое может быть представлено машиной (применительно к нашему примеру результат не должен превышать 999.999: в противном случае старшие разряды числа будут потеряны, а результат вычисления – искажен. Такое явление называется переполнением разрядной сетки.

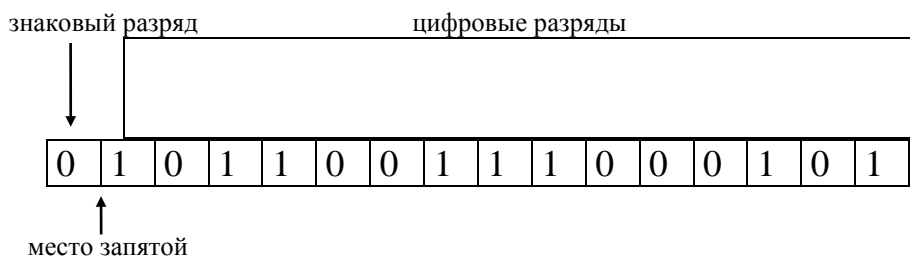
Из-за перечисленных недостатков режим с фиксированной запятой оказывается неудобным для решения задач, включающих в себя разнообразные и обширные вычисления, проводимые над числами из достаточно широкого диапазона. При работе в режиме с фиксированной запятой необходимо заранее (еще при подготовке задачи к решению на машине) учитывать возможность переполнения разрядной сетки. Чтобы не допустить переполнения, входящие в задачу исходные данные, приходится

умножать на соответствующие масштабные коэффициенты (масштабировать). Масштабные множители подбираются таким образом, чтобы все участвующие в арифметических операциях величины после масштабирования оказались в допустимых для данной машины пределах. При этом следует еще предусмотреть, чтобы результаты вычислений также оказались в допустимых пределах.

Кроме того, масштабные множители должны по возможности приближать числа, реально участвующие в вычислениях, к верхнему допустимому пределу. Это требование объясняется тем, что при работе машины в режиме с фиксированной запятой малые по абсолютной величине числа могут быть представлены со значительно меньшим количеством значащих цифр, - то есть с меньшей относительной точностью, чем числа, близкие к верхней границе. Поэтому масштабные коэффициенты для различных величин, участвующих в вычислениях, могут оказаться различными. Более того, эти коэффициенты, возможно, потребуются менять в ходе вычислений.

Введение масштабных множителей, как правило, сильно усложняет расчетные формулы, входящие в алгоритм решения задачи. Подбор этих множителей является делом очень сложным и в большей степени зависит от опытности математика, подготавливающего задачу для решения на машине. Поэтому режим работы с фиксированной запятой обычно применяется только в малых и специализированных ЭВМ; такие машины являются менее сложными и, следовательно, более дешевыми, чем машины, которые могут работать в режиме с плавающей запятой.

Ячейка памяти машины, содержащая число с фиксированной запятой, имеет знаковый разряд и цифровые разряды. Разряды ячейки нумеруются слева направо, начиная с нуля. Знак числа указывается в нулевом (знаковом) разряде ячейки следующим образом: плюс изображается нулем, а минус – единицей. Двоичная запись числа помещается в цифровые разряды ячейки (с первого по (n-1)-ый в n-разрядной ячейке), причем каждый разряд числа записывается в строго определенном месте в зависимости от его удаления от запятой. Чаще всего запятая фиксируется перед первым (старшим) цифровым разрядом. Тогда, например, двоичное число $+0.101100111000101$ в шестнадцатиразрядной ячейке ЭВМ запишется следующим образом:



Фиксация запятой перед старшим цифровым разрядом требует, чтобы все величины, участвующие в решении задачи, были меньше единицы. В этом случае несколько облегчается подбор масштабных коэффициентов, поскольку при выполнении операции умножения (встречающейся очень часто) переполнение разрядной сетки исключается, так как в этом случае произведение не превосходит по абсолютной величине ни один из сомножителей.

Представление целых чисел.

Часто в машине требуется хранить точные значения некоторых чисел, например, значения индексом переменной, номеров ячеек и т.п. Такие числа обычно записывают в ячейки памяти как числа с фиксированной запятой, предполагая, что запятая стоит после последнего ((n-1)-го разряда. Эту форму записи иногда называют записью в виде условного целого числа. Например, число $100_{10} = 1100100_2$ в форме условного целого числа так запишется в 16-разрядной ячейке:

0 000 000 001 100 100.

В восьмеричной системе эта запись выглядит так:

000 144.

Действия с условными целыми числами также удобно производить в режиме с фиксированной запятой, поскольку при выполнении сложения и вычитания в этом режиме не теряется ни один из разрядов, представляющих данные числа в машине.

Вся информация, обрабатываемая в машине (числа с фиксированной запятой, числа с плавающей запятой и т. д.), в конечном счете представляет собой некоторые последовательности нулей и единиц, размещенные в ячейках памяти ЭВМ. При выполнении над содержимым ячейки той или иной операции, находящийся в этой ячейке набор двоичных разрядов расшифровывается в соответствии с типом операции. Сама команда также представляет собой набор двоичных разрядов. В частности, команды, записанные в ячейках памяти машины, можно рассматривать как условные целые числа и преобразовывать их с помощью операций над такими числами.

4.2. Представление чисел в форме с плавающей запятой

Запись чисел с плавающей запятой в ячейки памяти ЭВМ основывается на нормальной форме представления чисел. Положительное число x считается представленным в нормальной форме, если оно записано в виде произведения:

$$x=q^p \cdot M,$$

где q – основание системы счисления; p – целое число (порядок); M – правильная положительная дробь (мантисса). Соответственно, если $x < 0$, то его нормальная форма такова:

$$x=-q^p \cdot M.$$

Различают нормализованные и ненормализованные нормальные числа. Если в первом разряде мантиссы стоит цифра, отличная от нуля, то есть $1/q \leq M < 1$, то нормальное число называют нормализованным. Если же цифра первого разряда мантиссы является нулем, то нормальное число называют ненормализованным. Желательно хранить числа в машине в нормализованном виде, так как при этом не теряются последние разряды мантиссы; говоря в дальнейшем о нормальных числах, мы всегда будем иметь в виду нормализованные числа.

Примеры записи десятичных чисел в нормальной форме:

$$\begin{aligned} 1 &= 10^1 \cdot 0.1 & (p=1, M=0.1); \\ 37.2 &= 10^2 \cdot 0.372 & (p=2, M=0.372); \\ 0.00035 &= 10^{-3} \cdot 0.35 & (p=-3, M=0.35); \\ 0.56 &= 10^0 \cdot 0.56 & (p=0, M=0.56). \end{aligned}$$

Аналогично записываются числа в нормальной форме в восьмеричной системе:

$$\begin{aligned} 6_8 &= \left(8^1 \cdot \frac{6}{8} \right)_{10} = 10^1 \cdot 0.6_8 \quad (p=1, M=0.6_8), \\ \left(\frac{1}{16} \right)_{10} &= \left(8^{-1} \cdot \frac{1}{2} \right)_{10} = 10^{-1} \cdot 0.4_8 \quad (p=-1, M=0.4_8). \end{aligned}$$

В двоичной системе:

$$\begin{aligned} 1_{10} &= \left(2^1 \cdot \frac{1}{2} \right)_{10} = 10^1 \cdot 0.1_2 \quad (p=1, M=0.1_2); \\ 8_{10} &= \left(2^4 \cdot \frac{1}{2} \right)_{10} = 10^{100} \cdot 0.1_2 \quad (p=100_2, M=0.1_2); \\ 50_{10} &= \left(2^6 \cdot \frac{25}{32} \right)_{10} = 10^{110} \cdot 0.11001_2 \quad (p=110_2, M=0.11001_2); \\ \frac{3}{32_{10}} &= \left(2^{-3} \cdot \frac{3}{4} \right)_{10} = 10^{-11} \cdot 0.11_2 \quad (p=-11_2, M=0.11_2). \end{aligned}$$

Приведенное выше определение нормальной формы числа относится ко всем числам, кроме нуля. Число **0** в нормальной форме записывается особым образом. Естественно считать мантиссу нуля равной нулю (хотя при этом и не выполняется условие $q \leq M$).

Тогда $0 = q^p \cdot 0$. Это равенство выполняется при любом p . Условимся считать $p = -1\ 000\ 000_2$.

Рассмотрим один из способов записи в памяти машины числа с плавающей запятой в предположении, что для записи каждого числа отводится 32-разрядная ячейка. Знак числа записывается в нулевом разряде ячейки (плюс изображается нулем, минус – единицей). В следующие семь разрядов (с первого по седьмой) записывается:

$$1\ 000\ 000_2 + p_2.$$

В остальные ряды записываются цифры мантиссы, следующие после запятой.

Запишем в форме с плавающей запятой число 5. Прежде всего, представим число 5 в нормальной форме в двоичной системе: $5_{10} = 101_2 = (10^{11} \cdot 0.101)_2$ ($p = 11_2$, $M = 0.101_2$). Таким образом, число 5 в форме с плавающей запятой так запишется в 32-разрядной ячейке:

0	1	0	0	0	0	1	1	1	0	1	0	0	0	...	0	0	0
1 000 000 + p									мантисса								

Аналогичная запись числа (-5) отличается от приведенной записи числа $(+5)$ только знаком: в нулевом разряде ячейки мы должны записать в этом случае 1. Таким образом, число -5 в форме с плавающей запятой так запишется в 32-разрядной ячейке:

$$\begin{array}{ccccccc} 31 & 30 & & 24 & 23 & & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{array}$$

разряды

Записью числа 0 в форме с плавающей запятой является нулевой код:

$$00\ 000\ 000\ 000 \dots 000.$$

Заметим, что при рассмотренной нами форме записи чисел с плавающей запятой в машине не допускаются отрицательные порядки, то есть число

$$1\ 000\ 000 + p$$

должно быть всегда не меньше нуля.

Числа с порядком, меньшим $-1\ 000\ 000_2$,

мы не сможем записать в ячейку: если в результате каких-то действий получится такое число, то машина воспримет это число как ноль, то есть машинными нулями являются все числа, меньшие по абсолютной величине, чем $(10^{-1\,000\,000} \cdot 0.1)_2 = (2^{-128})_{10}$.

Числа с порядком, большим $+1\,000\,000_2$, мы также не сможем записать в ячейку, так как для записи порядка отводится лишь 7 двоичных разрядов. Если же машина получит такое большое число ($\sim 10^{19}$) в результате действий, то она остановится (произойдет переполнение разрядной сетки).

Рассмотренная нами форма записи чисел с плавающей запятой обеспечивает единственность представления каждого числа в машине.

Если бы, например, число записывалось так: в нулевом разряде знак числа, в первом порядке знак порядка, потом – сколько-то разрядов было отведено под порядок и остальные – под мантиссу, то двойка записывались бы числа с нулевым порядком ($+0 = -0$), то есть несовпадающие содержимые ячеек могли бы означать одно и то же число.

Режим с плавающей запятой обеспечивает достаточно широкий диапазон представления чисел в машине без применения масштабных коэффициентов. Недостатком такого способа представления чисел (в сравнении с представлением чисел с фиксированной запятой) является увеличение количества элементов, изображающих число в машине: кроме элементов, необходимых для записи мантиссы, требуются еще элементы для записи порядка. Отдать предпочтение какой-либо одной форме представления чисел нельзя. В больших универсальных цифровых машинах, предназначенных для решения широкого класса задач, обычно используются оба способа представления чисел, а при решении конкретной задачи выбирается способ, более удобный для данного вида задачи.

Примеры записи чисел в форме с плавающей запятой

1. Число 2 в нормальной форме записывается следующим образом:

$$2_{10} = 2^2 \cdot (1/2) = 10^{10} \cdot 0.1_2.$$

Соответствующая запись числа 2 в 32-разрядной ячейке памяти машины такова:

$$\begin{array}{ccccccc} & 31 & 30 & & 24 & 23 & & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0. \end{array}$$

$$2. -3_{10} = -2^2 \cdot (3/4) = -10^{10} \cdot 0.11_2.$$

Следовательно, число -3 так запишется в 32-разрядной ячейке:

$$11\,000\,010\,110\,000 \dots 000.$$

$$3. 3/64 = 2^{-4} \cdot (3/4) = 10^{-100} \cdot 0.11_2.$$

Запись числа $3/64$ в 32-разрядной ячейке памяти ЭВМ:

00 111 100 110 000 . . . 000.

4. $-(13/4096) = -2^{-8} \cdot (13/16) = -10^{-1000} \cdot 0.1101_2$.

Исходя из нормальной формы числа $-(13/4096)$, получим следующую запись числа в 32-разрядной ячейке памяти:

10 111 000 110 100 000 . . . 000.

5. Арифметические операции в ЭВМ

Все современные ЭВМ имеют достаточно развитую систему команд, включающую десятки и сотни машинных операций. Однако выполнение любой операции основано на использовании простейших микроопераций типа сложения и сдвиг. Это позволяет иметь единое арифметико-логическое устройство для выполнения любых операций, связанных с обработкой информации. Правила сложения двоичных цифр двух чисел А и В представлены в табл. 5.1.

Таблица 5.1.

Правила сложения двоичных цифр

Значения двоичных чисел А и В			Разряд Суммы S_i	Перенос в следующий разряд P_i
a_i	b_i	P_{i-1}		
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Здесь показаны правила сложения двоичных цифр a_i , b_i одноименных разрядов с учетом возможных переносов из предыдущего разряда p_{i-1} .

Подобные таблицы можно было бы построить для любой другой арифметической и логической операции (вычитание, умножение и т.д.), но именно данные этой таблицы положены в основу выполнения любой операции ЭВМ. Под знак чисел отводится специальный знаковый разряд. Знак “+” кодируется двоичным нулем, а знак “-” - единицей. Действия над прямыми кодами двоичных чисел при выполнении операций создают большие трудности, связанные с необходимостью учета значений знаковых разрядов:

- во-первых, следует отдельно обрабатывать значащие разряды чисел и разряды знака;
- во-вторых, значение разряда знака влияет на алгоритм выполнения операции (сложение может заменяться вычитанием и наоборот).

Во всех ЭВМ без исключения все операции выполняются над числами, представленными специальными машинными кодами. Их использование позволяет обрабатывать знаковые разряды чисел так же, как и значащие разряды, а также заменять операцию вычитания операцией сложения,

Различают **прямой код (П)**, **обратный код (ОК)** и **дополнительный код (ДК)** двоичных чисел.

5.1. Машинные коды

Прямой код двоичного числа образуется из абсолютного значения этого числа и кода знака (ноль или единица) перед его старшим числовым разрядом.

Пример.

$A_{10}=+10$ $A_2=+1010$ $[A_2]_П=0:1010;$

$B_{10}=-15$ $B_2=-1111$ $[B_2]_П=1:1111.$

Точечной вертикальной линией отмечена условная граница, отделяющая знаковый разряд от значащих.

Обратный код двоичного числа образуется по следующему правилу. Обратный код положительных чисел совпадает с их прямым кодом. Обратный код отрицательного числа содержит единицу в знаковом разряде числа, а значащие разряды числа заменяются на инверсные, т.е. нули заменяются единицами, а единицы - нулями.

Пример.

$A_{10}=+5$ $A_2=+101$ $[A_2]_П=[A_2]_{ОК}=0:101;$

$B_{10}=-13$ $B_2=-1101$ $[B_2]_{ОК}=1:0010.$

Свое название обратный код чисел получил потому, что коды цифр отрицательного числа заменены на инверсные. Укажем наиболее важные свойства обратного кода чисел:

- сложение положительного числа S с его отрицательным значением в обратном коде дает так называемую машинную единицу $ME_{ОК}= 1: 111... 11$, состоящую из единиц в знаковом и значащих разрядах числа;
- нуль в обратном коде имеет двойное значение. Он может быть положительным - $0: 00...0$ и отрицательным числом - $1; 11... 11$. Значение отрицательного нуля совпадает с $ME_{ОК}$. Двойственное представление нуля явилось причиной того, что в современных ЭВМ все числа представляются не обратным, а дополнительным кодом.

Дополнительный код положительных чисел совпадает с их прямым кодом. Дополнительный код отрицательного числа представляет собой результат суммирования обратного кода числа с единицей младшего разряда (2^0 - для целых чисел, 2^{-k} - для дробных).

Пример.

$$A_{10}=+19 \quad A_2=+10011 \quad [A_2]_{\Pi}=[A_2]_{\text{OK}}=[A_2]_{\text{ДК}}=0:10011;$$

$$B_{10}=-13 \quad B_2=-1101 \quad [B_2]_{\text{ДК}}=[B_2]_{\text{OK}}+2^0=1:0010+1=1:0011.$$

Укажем основные свойства дополнительного кода:

• сложение дополнительных кодов положительного числа C с его отрицательным значением дает так называемую машинную единицу дополнительного кода:

$$ME_{\text{ДК}}=ME_{\text{OK}}+2^0=10:00\dots00, \text{ т.е. число } 10 \text{ (два) в знаковых разрядах числа};$$

• дополнительный код получил такое свое название потому, что представление отрицательных чисел является дополнением прямого кода чисел до машинной единицы $ME_{\text{ДК}}$.

Модифицированные обратные и дополнительные коды двоичных чисел отличаются соответственно от обратных и дополнительных кодов удвоением значений знаковых разрядов. Знак “+” в этих кодах кодируется двумя нулевыми знаковыми разрядами, а “-” - двумя единичными разрядами.

Пример.

$$A_{10}=9 \quad A_2=+1001 \quad [A_2]_{\Pi}=[A_2]_{\text{OK}}=[A_2]_{\text{ДК}}=0:1001$$

$$[A_2]_{\text{МОК}}=[A_2]_{\text{МДК}}=00:1001;$$

$$B_{10}=-9 \quad B_2=-1001 \quad [B_2]_{\text{OK}}=1:0110 \quad [B_2]_{\text{ДК}}=1:0111$$

$$[B_2]_{\text{МОК}}=11:0110 \quad [B_2]_{\text{МДК}}=11:0111.$$

Целью введения модифицированных кодов являются фиксация и обнаружение случаев получения неправильного результата, когда значение результата превышает максимально возможный результат в отведенной разрядной сетке машины. В этом случае перенос из значащего разряда может исказить значение младшего знакового разряда. Значение знаковых разрядов “01” свидетельствует о положительном переполнении разрядной сетки, а “10” - об отрицательном переполнении. В настоящее время практически во всех моделях ЭВМ роль удвоенных разрядов для фиксации переполнения разрядной сетки играют переносы, идущие в знаковый и из знакового разряда.

5.2. Арифметические операции над числами с фиксированной точкой

Сложение (вычитание). Операция вычитания приводится к операции сложения путем преобразования чисел в обратный или дополнительный код. Пусть числа $A \geq 0$ и $B \geq 0$, тогда операция алгебраического сложения выполняется в соответствии с табл. 5.2.

Таблица 5.2.

Таблица преобразования кодов при алгебраическом сложении

Требуемая операция	Необходимое преобразование
$A+B$	$A+B$
$A-B$	$A+(-B)$
$-A+B$	$(-A)+B$
$-A-B$	$(-A)+(-B)$

Скобки в представленных выражениях указывают на замену операции вычитания операцией сложения с обратным или дополнительным кодом соответствующего числа. Сложение двоичных чисел осуществляется последовательно, поразрядно в соответствии с табл. 4.2.

При выполнении сложения цифр необходимо соблюдать следующие правила.

1. Выравнивание слагаемых. Слагаемые должны иметь одинаковое число разрядов. Для выравнивания разрядной сетки слагаемых можно дописывать незначащие нули слева к целой части числа и незначащие нули справа к дробной части числа.

2. Знаковые разряды чисел участвуют в сложении так же, как и значащие.

3. Необходимые преобразования кодов производятся с изменением знаков чисел. *Приписанные незначащие нули изменяют свое значение при преобразованиях по общему правилу.*

4. При образовании единицы переноса из старшего знакового разряда, в случае использования ОК, эта единица складывается с младшим числовым разрядом. При использовании ДК единица переноса теряется. **Знак результата формируется автоматически**, результат представляется в том коде, в котором представлены исходные слагаемые.

Пример 1. Сложить два числа $A_{10}=7$ $B_{10}=16$

$A_2=+111;$

$B_2=+10000$

Исходные числа имеют различную разрядность, необходимо провести выравнивание разрядной сетки:

$[A_2]_п=[A_2]_{ок}=[A_2]_{дк}=0: 00111;$

$[B_2]_п=[B_2]_{ок}=[B_2]_{дк}=0: 10000.$

Сложение в обратном или дополнительном коде дает один и тот же результат

$$\begin{array}{r} 0: 00111 \\ + 0: 10000 \\ \hline C_2 = 0: 10111 \\ C_{10} = +23. \end{array}$$

Обратим внимание, что при сложении цифр отсутствуют переносы в знаковый разряд и из знакового разряда, что свидетельствует о получении правильного результата.

Пример 2. Сложить два числа $A_{10} = +16$ $B_{10} = -7$ в ОК и ДК. В соответствии с табл. 4.3 должна быть реализована зависимость $A+(-B)$, в которой второй член преобразуется с учетом знака

$$\begin{aligned} [A_2]_{\Pi} &= 0; 10000 = 0; 10000 & [A_2]_{OK} &= 0; 10000 & [A_2]_{ДК} &= 0; 10000; \\ [B_2]_{\Pi} &= 1; 111 = 1; 00111 & [B_2]_{OK} &= 1; 11000 & [B_2]_{ДК} &= 1; 11001. \end{aligned}$$

Сложение в ОК

$$\begin{array}{r} [A_2]_{\text{ОК}} = 0; 10000 \\ + [B_2]_{\text{ОК}} = 1; 11000 \\ \hline \quad \quad \quad \overbrace{0; 01000} \\ + \quad \quad \quad \underbrace{\quad \quad \quad 1} \\ \hline \quad \quad \quad 0; 01001 \end{array}$$

$C_2 = 0; 01001$
 $C_{10} = +9$

Сложение в ДК

$$\begin{array}{r} [A_2]_{\text{ДК}} = 0; 10000 \\ + [B_2]_{\text{ДК}} = 1; 11001 \\ \hline \quad \quad \quad \underbrace{}_{0}; \underbrace{}_{01001} \end{array}$$

$C_2 = 0; 01001$

$C_{10} = +9$

При сложении чисел в ОК и ДК были получены переносы в знаковый разряд и из знакового разряда. В случае ОК перенос из знакового разряда требует дополнительного прибавления единицы младшего разряда (см.п.4 правил). В случае ДК этот перенос игнорируется.

Умножение. Умножение двоичных чисел наиболее просто реализуется в прямом коде. Рассмотрим, каким образом оно приводится к операциям сложения и сдвигам.

Пример 3. Умножить два числа $A_{10}=7$ $B_{10}=5$.

Перемножим эти числа, представленные прямыми двоичными кодами, так же, как это делается в десятичной системе.

$$\begin{array}{r}
[A_2]_n = 111 \text{ – множимое} \\
* \quad * \\
[B_2]_n = 101 \text{ – множимое} \\
\quad 111 \text{ – множимое (сдвиг на 0 разрядов)} \\
\quad +000 \text{ – умножение на 0 (сдвиг на 1 разряд)} \\
\quad \underline{111} \text{ – множимое (сдвиг на 2 разряда)} \\
[C_2]_n = 100011 \text{ – произведение} \\
C_{10} = 35
\end{array}$$

Нетрудно видеть, что произведение получается путём сложения частных произведений, представляющих собой разряды множимого, сдвинутые влево в соответствии с позициями разрядов множителя. Частные произведения, полученные умножением на нуль, игнорируются. Важной особенностью операции умножения n -разрядных сомножителей является увеличение разрядности произведения до $n+n=2n$. **Знак произведения формируется путём сложения знаковых разрядов сомножителей. Возможные переносы из знакового разряда игнорируются.**

Деление. Операция деления, как и в десятичной арифметике, является обратной операции умножения. Покажем, что и эта операция приводится к последовательности операций сложения и сдвига.

Пример 4. Разделить два числа $A_{10}=45$ $B_{10}=5$

$$\begin{array}{r}
[A_2]_n = 101101 \\
[B_2]_n = 101 \\
\text{Делимое} \quad \text{Делитель} \\
101101 \quad \underline{101} \\
-101 \quad \quad 1001 \text{ – частное} \\
\hline
0101 \\
-101 \\
\hline
0 \\
[C_2]_n = 1001 \\
C_{10} = 9
\end{array}$$

Деление произведено так же, как это делается обычно в десятичной системе. Сначала проверяется, можно ли вычесть значение делителя из старших разрядов делимого. Если возможно, то в разряде частного записывается единица и определяется частная разность. В противном случае в частное записывается нуль и разряды делителя сдвигаются вправо на один разряд по отношению к разрядам делимого. К полученной предыдущей разности сносится очередная цифра делимого, и данный процесс повторяется, пока не будет получена необходимая точность. Если учесть, что все вычитания в ЭВМ заменяются сложением в ОК или в ДК (см. табл.4.2.), то действительно операция деления приводится к операциям сложения и сдвигам вправо разрядов делителя относительно разрядов делимого. Отметим, что

делимое перед операцией деления должно быть приведено к $2n$ -разрядной сетке. Только в этом случае при делении на n -разрядный делитель получается n -разрядное частное.

Знак частного формируется также путем сложения знаковых разрядов делимого и делителя, как это делалось при умножении.

5.3. Арифметические операции над двоичными числами с плавающей точкой

В современных ЭВМ числа с плавающей точкой хранятся в памяти машин, имея мантиссу и порядок (характеристику) в прямом коде и нормализованном виде. Все арифметические действия над этими числами выполняются так же, как это делается с ними, если они представлены в полулогарифмической форме (мантисса и десятичный порядок) в десятичной системе счисления. **Порядки и мантиссы обрабатываются раздельно.**

Сложение (вычитание). Операция сложения (вычитания) производится в следующей последовательности.

1. Сравниваются порядки (характеристики) исходных чисел путем их вычитания $\Delta p = p_1 - p_2$. При выполнении этой операции определяется, одинаковый ли порядок имеют исходные слагаемые.

2. Если разность порядков равна нулю, то это значит, что одноименные разряды мантисс имеют одинаковые веса (двоичный порядок). В противном случае должно проводиться выравнивание порядков.

3. Для выравнивания порядков число с меньшим порядком сдвигается вправо на разницу порядков Δp . Младшие выталкиваемые разряды при этом теряются.

4. После выравнивания порядков мантиссы чисел можно складывать (вычитать) в зависимости от требуемой операции. Операция вычитания заменяется операцией сложения в соответствии с данными таблицы 5.2. Действия над слагаемыми производятся в ОК или ДК по общим правилам.

5. Порядок результата берется равным большему порядку $p_{рез} = \max(p_1, p_2)$.

6. Если мантисса результата не нормализована, то осуществляются нормализация и коррекция значений порядка.

Пример. Сложить два числа $A_{10} = +1.375$; $B_{10} = -0.625$.

$$A_2 = +1.011 = 0:1011 \cdot 10^1; B_2 = -0.101 = 1:101 \cdot 10^0.$$

В нормализованном виде эти числа будут иметь вид:

<i>Порядок</i>	<i>Мантисса</i>
$[A_2]_{II} = 0:1$	$\uparrow 0:1011$
	<i>знак числа</i>
$[B_2]_{II} = 0:0$	$\downarrow 1:101$

1. Вычитаем порядки $\Delta p = p_1 - p_2 = 1 - 0 = 1$. В машине эта операция требует операции сложения с преобразованием порядка чисел в дополнительный код:

$$\begin{array}{rcl} p_1 = 0:1 & & [p_1]_{ДК} = 0:1 \\ p_2 = 0:0 & + & [p_2]_{ДК} = 0:0 \\ \hline \Delta p & = & 0:1 \end{array}$$

Определяем, что $\Delta p \neq 0$.

2. Порядок первого числа больше порядка второго числа на единицу. Требуется выравнивание порядков.

3. Для выравнивания порядков необходимо второе число сдвинуть вправо на один разряд.

$$[B_2]_{\text{исх}} = 0:0 \ 1: 101$$

после сдвига

$$[B_2]_{\text{п}} = 0:1 \ 1:0101$$

мантисса числа В в дополнительном коде: $[m'_B]_{\text{дк}} = 1: 1011$

4. Складываем мантииссы.

$$\begin{array}{r} [m_A]_{\text{дк}} = 0:1011 \\ + [m'_B]_{\text{дк}} = 1:1011 \\ \hline [m_C]_{\text{дк}} = 0:0110 \end{array}$$

Мантисса числа С - положительная.

5. Порядок числа С равен порядку числа с большим порядком, т.е. $p_c = +1$.

$$[C_2]_{\text{п}} = 0:1 \ 0: 0110.$$

Видно, что мантисса результата не нормализована, так как старшая цифра мантииссы равна нулю.

6. Нормализуем результат путем сдвига мантииссы на один разряд влево и соответственно вычитаем из значения порядка единицу:

$$\begin{array}{l} [C_2]_{\text{п}} = 0:00:110 \\ C_{10} = +0.75. \end{array}$$

Умножение (деление). Операция умножения (деления) чисел с плавающей точкой также требует разных действий над порядками и мантииссами. Алгоритмы этих операций выполняются в следующей последовательности.

1. При умножении (делении) порядки складываются (вычитаются) так, как это делается над числами с фиксированной точкой.

2. При умножении (делении) мантииссы перемножаются (делятся).

3. Знаки произведения (частного) формируются путем сложения знаковых разрядов сомножителей (делимого и делителя). Возможные переносы из знакового разряда игнорируются.

Пример. Перемножить два числа $A_{10}=+1.25$; $B_{10}=-0.4$; $C_{10} = - 0.50$

В двоичной системе: $A_2=+1.010=0:101\times10^1$; $B_2= -0.011=1:11\times10^{-1}$.

1. Сложение порядков: $(1+(-1))\ 0:1 + 1:1$

2. Перемножение мантисс $101\times11=01111$

3. Определение знака результата: $0+1=1$, т.е. результат отрицательный

5.4. Арифметические операции над двоично-десятичными кодами чисел

При обработке больших массивов экономической информации переводы чисел из десятичной системы в двоичную и обратно могут требовать значительного машинного времени. Некоторые образцы ЭВМ поэтому имеют или встроенные, или подключаемые блоки, которые обрабатывают десятичные целые числа в их двоично-десятичном представлении. Действия над ними также приводятся к операции алгебраического сложения отдельных цифр чисел, представленных дополнительными кодами.

Приведем один из **алгоритмов сложения**, который получил довольно широкое распространение.

1. Сложение чисел начинается с младших цифр (тетрад) и производится с учетом возникающих переносов из младших разрядов в старшие.

2. Знак суммы формируется специальной логической схемой по знаку большего слагаемого.

3. Для того чтобы при сложении двоично-десятичных цифр возникали переносы, аналогичные при сложении чисел в десятичном представлении, необходимо проводить так называемую **десятичную коррекцию**. Для этого к каждой тетраде первого числа прибавляется дополнительно по цифре $6_{10}=0110_2$, что позволяет исключить шесть неиспользуемых комбинаций $(1010-1111)_2$, так как они кодируют шестнадцатеричные цифры A-F (числа $10-15_{10}$).

4. После операции суммирования осуществляется корректировка суммы. Из тех тетрад суммы, из которых не было переносов, изымаются ранее внесенные избытки $6_{10}=0110_2$. Для этого проводится **вторая коррекция**. Операция вычитания заменяется, как и обычно, операцией сложения с числом -6, представленным дополнительным кодом $[-6]_{\text{дк}}=1010_2$, но только в тех разрядах, в которых отсутствовали переносы. При этой второй коррекции **переносы из тетрад блокируются**.

5. **Операция вычитания** выполняется по общему правилу сложения: к тетрадам числа с большим модулем прибавляются **дополнительные коды тетрад**

другого числа. После операции суммирования осуществляется корректировка суммы, а именно тех тетрад суммы, из которых не было переносов (**вторая коррекция** - операция сложения с числом -6, представленным дополнительным кодом $[-6]_{\text{ДК}}=1010_2$). При коррекции **переносы из тетрад блокируются**. В качестве знака результата берется знак числа с большим модулем.

Пример. Сложить два числа $A_{10}=177$; $B_{10}=418$

$A_{2-10}=$	0001	0111	0111	
	+			1-ая коррекция
		0110	0110	0110
A'	0111	1101	1101	
	+			Сложение $A'+B$
$B_{2-10}=$	0100	0001	1000	
		1011	1111	0101
				Результат C'
	+			
		1010	1010	2-ая коррекция
C_{2-10}	0101	1001	0101	результат
$C_{10}=$	595			

Пример. Сложить два числа $A_{10}=-177$; $B_{10}=418$

Задание для самостоятельного выполнения:

1. Выполнить сложение двоичных чисел с плавающей точкой: $11011,1+11,111$
 $-11101,11 -11,101$
2. Выполнить сложение «2-10» кодированных чисел: 245-79