Лекция 3.

3.1 Информация

Информация - любые сведения, являющиеся объектом хранения, преобразования и передачи.

В широком смысле слова ***информация*** является отражением реального мира. Информация - это единственный неубывающий ресурс жизнеобеспечения. Более того: ее объем в настоящее время удваивается ежегодно. Информация, подготовленная для обработки на компьютерах, называется ***данными***. Информационный процесс включает в себя такие этапы:

1. сбор информации от различных источников и представление ее в форме, необходимой для ввода в компьютер;
2. передачу (пересылку) информации от источника к приемнику;
3. хранение — процесс передачи информации во времени;
4. обработку — систематическое выполнение операций над данными;
5. выдачу результата обработки пользователю.

На всех этих этапах используют средства компьютерной схемотехники. К информации предъявляют следующие требования:

- корректность (однозначность восприятия);
- ценность (полезность) и оперативность (актуальность);
- точность, достоверность и устойчивость (способность реагировать на изменения исходных данных);
- достаточность (полнота) — наличие минимально необходимого объема информации для принятия правильного решения.

Структуру и общие свойства информационных процессов изучают в информатике, которая включает:

- теорию информации;
- алгоритмические, программные и компьютерные средства обработки информации;
- архитектуру компьютеров, системы искусственного интеллекта, вычислительные сети и т.д.

В теории информации изучают процессы передачи, преобразования и хранения информации, в том числе:

- методы определения количества информации в сообщении;
- рациональные способы представления информации с помощью различных символов (букв, цифр и т.д.);

- способы формирования, обнаружения и оценки параметров информационных процессов.

Упорядоченную последовательность символов (букв, цифр, математических знаков, предназначенных для передачи информации), закодированную в материальной форме, называют сообщением. Информационное сообщение всегда связано с источником и приемником информации, соединенных каналом передачи (рис. 3.1).

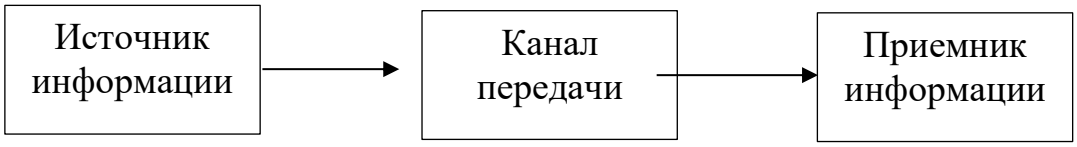


Рис. 3.1. Информационная модель канала передачи

Источником и приемником информации могут быть как люди, так и технические устройства (компьютеры, датчики, индикаторы и др.). Каналом передачи (связи) называется совокупность устройств, имеющих один вход и один выход, предназначенных для передачи информации на расстояния. Сообщения могут иметь различные формы: звука, текста, изображения, электрического напряжения от датчиков (например, от термопар). Информационные сообщения размещают на машинных носителях информации. ***Носитель информации*** - это любая запоминающая предметная среда, предназначенная для записи и хранения информации с целью ее непосредственного ввода в компьютер. Носитель информации является промежуточным звеном между компьютером и первичным документом, содержащим числовые данные, текстовые материалы, схемы, графики, различные измерения. Хранение сообщений - это, как правило, три вида операций: запись, собственно хранение и считывание. Информация записывается в носитель посредством изменения физических или механических свойств запоминающей среды. Данные считываются посредством преобразования их в электрические сигналы. Считают, что в носителе информации сигнал хранится в закодированной форме. Носители информации различаются по следующим признакам:

- средой накопления: непрерывные (магнитные ленты и диски) и дискретные, где каждой хранимой единице данных отводится свой дискретный участок или элемент (ферритовые сердечники, перфокарты, перфоленты, триггеры, криотроны и т.д.);
- типом материала: бумага с текстом или рисунком; бумажные перфоленты, перфокарты; магнитные пленки, магнитные ленты и диски;

- способом считывания данных: механические, оптические, магнитные, электрические;
- конструктивным исполнением: ленточные, дисковые, электронные и др.

В теории передачи и преобразования информации установлены информационные меры количества и качества информации - ***семантические, структурные, статистические***.

Семантический подход позволят выделить полезность или ценность информационного сообщения.

3.1.1 Структурная мера информации

Информация всегда представляется в виде сообщения. Элементарная единица сообщений - символ. Символы, собранные в группы - слова. Сообщение, оформленное в виде слов всегда передается в материально-энергетической среде.

Структурный подход используют для оценки возможностей информационных систем вне зависимости от условий их применения. При использовании структурных мер информации учитывают только дискретное строение сообщения, количество содержащихся в нем информационных элементов, связей между ними.

При структурном подходе различают ***геометрическую, комбинаторную и аддитивную*** меры информации. Геометрическая мера определяет параметры геометрической модели информационного сообщения (длина, площадь, объем) в дискретных единицах. Эту меру применяют как для оценки информационной емкости всей модели, так и для оценки количества информации в одном сообщении.

В комбинаторной мере количество информации I определяют количеством комбинаций элементов (символов), которые совпадают с числом:

- сочетаний из q элементов по n; например, для множества цифр 1, 2, 3, 4 можно составить шесть сочетаний по две цифры: 12, 13, 14, 23, 24, 34;
- перестановок $I = q!$; например, для множества букв *а, в, с* можно получить шесть перестановок: *авс, асв, вас, вса, сав, сва*;
- размещений с повторениями из q элементов по n, $I = q^n$. Например, для $q = 0, 1$ и $n = 3$ имеем: 000, 001, 010-, 011, 100, 101, 110, 111.

Аддитивная мера (мера Хартли) - в соответствии с которой количество информации измеряется в двоичных единицах - ***битах*** - наиболее распространена. Вводится понятие ***глубины q числа*** и ***длины n числа***.

Глубина q числа - количество символов, принятых для представления информации. В каждый момент времени реализуется только один какой-либо символ.

Длина n числа - количество позиций, необходимых и достаточных для представления чисел заданной величины.

При заданной глубине и длине числа количество возможных сообщений длины *n* равняется числу размещений с повторениями

$$N = q^n. \qquad (3.1)$$

Эту меру наделяют свойством аддитивности, чтобы она была пропорциональна длине сообщения и позволяла складывать количество информации ряда источников сообщений. Хартли предложил логарифмическую функцию как меру количества информации, определяемую как количество информации при минимальной глубине и минимальной длине числа, т.е. при g=2 и n=1, которая получила название - ***бит***.

$$I(g) = \log_2 N = n \log_2 g \qquad (3.2)$$

Следовательно, **1** ***бит*** информации соответствует одному символу двоичной системы счисления. С другой стороны, как будет видно далее, 1 бит соответствует элементарному событию, которое может произойти или не произойти. Такая мера количества информации удобна тем, что она обеспечивает возможность оперировать мерой как числом. Количество информации при этом эквивалентно количеству двоичных символов.

При наличии нескольких источников информации общее количество информации определяется как их сумма:

$$I(g_1, g_2, \cdots g_n) = \sum_{i=1}^{n} I(g_i) \qquad (3.3)$$

3.1.2 Статистическая мера информации

В статистической теории информации вводится более общая мера количества информации, в соответствии с которой рассматривается не само событие, а информация о нем[1].

Как показал К. Шеннон, в теории информации любое событие, с которым мы имеем дело, представляет собой совокупность сведений о некоторой физической системе. Очевидно, если бы состояние физической системы было бы известно заранее, не было бы смысла передавать сообщения.

[1] К. Шеннон “Избранные труды по теории информации”.

Рассмотрим некую гипотетическую систему X, которая может оказаться в том или ином состоянии, т.е. систему с заведомо присущей ей степенью неопределенности.

$$X \to X_1, X_2, \cdots X_i \qquad (3.4)$$

Очевидно, сведения, получаемые о системе, будут тем больше и ценнее, чем больше была неопределенность системы до получения этих сведений (априори). Возникает естественный вопрос, можно ли измерить степень неопределенности системы. Чтобы ответить на этот вопрос, рассмотрим три следующих примера:

1. Возьмем систему из одной монеты. В результате бросания монеты она может оказаться в одном из двух состояний;
2. Система из игральной кости. В результате бросания кости она может оказаться в одном из шести возможных состояний (“1”, “2”, “3”, “4”, “5”, “6”);
3. Рассмотрим некоторое устройство, которое может находиться в двух состояниях: “исправно”, “неисправно” соответственно с априорными вероятностями 0.99 и 0.01.

Сравнивая первый и второй пример, можно сделать вывод, что во втором случае степень неопределенности больше. Т.е. неопределенность должна оцениваться числом возможных исходов (состояний), в которых может оказаться система.

Сравнивая 1 и 3, можно также сделать вывод, что неопределенность первой системы гораздо больше. Т.е. при одинаковом числе состояний системы неопределенность должна зависеть от вероятности поступления событий.

В качестве меры априорной неопределенности системы в теории информации принимается величина (для дискретных систем)

$$H(x) = -\sum_{i=1}^{n} p(X_i) \cdot \log_a p(X_i) \qquad (3.5)$$

которую Шеннон назвал ***энтропией***, используя формальное ее сходство с энтропией в термодинамике. Основание логарифма влияет лишь на удобство вычисления.

В случае оценки энтропии

1. в двоичных единицах:
   $H(x) = -\sum_{i=1}^{n} p(X_i) \cdot \log_2 p(X_i)$ ***бит/символ***;
2. в десятичных единицах:
   $H(x) = -\sum_{i=1}^{n} p(X_i) \cdot \log p(X_i)$ ***дит /символ*** (1 бит ≈ 0.3 дит);
3. в натуральных единицах:

$H(x) = -\sum_{i=1}^{n} p(X_i) \cdot \ln p(X_i)$ ***нит/символ*** (1 бит ≈ 0.693 нит).

Как видно, 1 бит информации соответствует одному символу двоичной системы счисления, которая повсеместно используется в цифровых устройствах, и в частности –в ЭВМ

Например, для рассмотренных выше примеров получаем:

1. $H(x) = -\sum_{i=1}^{2} p(X_i) \cdot \log_2 p(X_i) = -0.5 \cdot \log_2 0.5 - 0.5 \cdot \log_2 0.5 =$ $= 0.5+0.5 = 1$бит.
2. $H(x) = -\sum_{i=1}^{6} p(X_i) \cdot \log_2 p(X_i) = -6 \cdot \frac{1}{6} \cdot \log_2 \frac{1}{6} =$ 2.58 бит.
3. $H(x) = -\sum_{i=1}^{2} p(X_i) \cdot \log_2 p(X_i) = -0.99 \cdot \log_2 0.99 -$ $0.01 \cdot \log_2 0.01$ = 0.08079 бит.

Если все элементы сообщения несут одинаковое количество информации, что часто имеет место в технических системах, то в этом случае энтропия - это количество информации, которое приходится на один элемент сообщения:

$$H = \frac{I}{n} = \log q. \qquad (3.6)$$

Как было уже показано, основание логарифма зависит от выбора единицы количества информации. Если для алфавита используют двоичные цифры 0 и 1, то за основание логарифма принимают q = 2, в результате чего $I = n \log_2 2 = n$. При длине n = 1 получают количество информации I = 1, которую называют ***битом***. Как видно из примера 1, передача сообщения длиной n=1 эквивалентна выбору одного из двух возможных равновероятных сообщений - одно из них равно единице, другое - нулю. Двоичное сообщение длины n содержит n битов информации. Если основание логарифма равно 10, то количество информации измеряется в десятичных единицах - ***дитах***, причем 1 ***дит*** = 3,32 бита. Например, текст составлен из 32 букв алфавита и передается последовательно по каналу связи в двоичном коде. При этом количество информации приходящее на один символ $I = \log_2 32 = \log_2 2^5 = 5$ бит. В общем случае сообщения появляются с разной вероятностью.

Статистическая мера использует вероятностный подход к оценке количества информации. Как было показано, каждое сообщение характеризуется вероятностью появления, и чем она меньше, тем больше в сообщении информации. Вероятность конкретных типов сообщений устанавливают на основе статистического анализа. Пусть сообщения образуются последовательной передачей букв некоторого алфавита $x_1, x_2, \cdots x_i, \cdots x_k$ с вероятностью появления каждой буквы $p(x_1) = p_1$, $p(x_2) = p_2$, $\cdots p(x_i) = p_i$, $\cdots p(x_k) = p_k$, при этом выполняется условие: $p_1 + p_2 + \cdots + p_i + \cdots + p_k = 1$. (Множество с известным распределением элементов называют ***ансамблем***). Согласно Шеннону количество

информации, которое содержится в сообщении x_i, рассчитывают по формуле (3.7).

$$I(x_i) = \log\frac{1}{p_i} \qquad (3.7)$$

Для абсолютно достоверных сообщений $p_i = 1$, и, следовательно, количество информации $I(x_i) = 0$. При уменьшении значения p_i количество информации увеличивается. Пусть в ансамбле все буквы алфавита $x_1, x_2, \cdots x_i, \cdots x_m$ равновероятны, то есть $p_1, p_2, \cdots p_i, \cdots p_m = \frac{1}{m}$, и статистически независимы. Тогда количество информации в сообщении длиной *n* букв с учетом выражения (3.7)

$$I = \sum_{i=1}^{n} I(x_i) = \sum_{i=1}^{n} \log\frac{1}{p_i} = \log\frac{1}{p_1} + \log\frac{1}{p_2} + \cdots + \log\frac{1}{p_m} = n\log m,$$

что совпадает с мерой Хартли в соответствии с выражениями (3.1) и (3.2).

Согласно Шеннону информация - это снятие неопределенности, что понимают следующим образом. До опыта событие (например, появление буквы x_i) характеризуют малой начальной вероятностью $p_{\text{н}}$, которой соответствует большая неопределенность. После опыта неопределенность уменьшается, поскольку конечная вероятность $p_{\text{к}} > p_{\text{н}}$. Уменьшение неопределенности рассчитывают как разность между начальным $I_{\text{н}}$ и конечным $I_{\text{к}}$ значениями количества информации. Например, для $p_{\text{н}}$= 0.1 и $p_{\text{к}}$ = 1 получим:

$$\Delta I = I_{\text{н}} - I_{\text{к}} = \log_2\frac{1}{p_{\text{н}}} - \log_2\frac{1}{p_{\text{к}}} = \log_2 10 - \log_2 1 = 3.32 \text{ бит.}$$

Пусть сложное сообщение характеризуется алфавитом из букв $x_1, x_2, \cdots x_i, \cdots x_k$, с их вероятностями $p_1, p_2, \cdots p_i, \cdots p_k$ и частотой появления каждой буквы $m_1, m_2, \cdots m_i, \cdots m_k$. Все сообщения статистически независимы, при этом $m_1 + m_2 + \cdots + m_i + \cdots + m_k = m$. Общее количество информации для всех *k* типов сообщений с учетом выражения (3.7)

$$I_{\Sigma} = \sum_{i=1}^{k} m_i \log\frac{1}{p_i} \quad . \qquad (3.8)$$

Среднее значение количества информации на одно сообщение (энтропия) согласно формуле Шеннона (3.7)

$$H = I_{\Sigma} = \sum_{i=1}^{k} \frac{m_i}{m}\log\frac{1}{p_i} = \sum_{i=1}^{k} p_i \log\frac{1}{p_i} \qquad (3.9)$$

где при большом значении m отношение m_i/m характеризует вероятность p_i каждой буквы. Выражение $\log 1/p_i$ рассматривают как частную энтропию, которая характеризует информативность буквы x_i, а энтропию H — как среднее значение частных энтропий. При малых значениях p_i частная энтропия велика, а с приближением p_i к единице она приближается к нулю.

3.1.3 Основные свойства энтропии

1. Энтропия есть величина вещественная, ограниченная и неотрицательная
2. Энтропия равна нулю тогда и только тогда, когда все вероятности кроме одной равны нулю и эта единственная вероятность равна единице.
3. Энтропия максимальна, если все состояния равновероятны:

$$H(X) = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i = -n \cdot \frac{1}{n} \cdot \log_2 \frac{1}{n} = \log_2 n. \qquad (3.10)$$

Доказательство:

$H(X) = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i$; $\varphi(P) = 1 - \sum_{i=1}^{n} P_i$; $\sum_{i=1}^{n} P_i = 1$.

Найдем условный экстремум методом неопределенных множителей Лагранжа (вариационная задача), для чего составим функционал:

$$\Phi(P,\lambda) = H + \lambda\varphi = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i + \lambda(1 - \sum_{i=1}^{n} P_i);$$

$$\frac{d\Phi(P,\lambda)}{dP_k} = -\log_2 P_k - \log_2 e - \lambda = 0;$$

$$\log_2 P_k = -\lambda - \log_2 e,$$

т.е. P_k не зависит от индекса k, и следовательно $P_1 = P_2 = \ldots = P_n = \frac{1}{n}$, ч.т.д.

В этом случае оценки количества информации по Хартли и Шеннону совпадают.

4. Энтропия объединения статистически независимых сообщений определяется суммой энтропий каждого сообщения (теорема сложения энтропий):

$$H(x,y) = -\sum_i \sum_j P(x_i, y_j) \cdot \log P(x_i, y_j) = \langle P(x_i, y_j) = P(x_i) \cdot P(y_j) \rangle =$$

$$= -\sum_i \sum_j P(x_i) \cdot P(y_j) \cdot [\log P(x_i) + \log P(y_j)] =$$

$$= -\sum_i P(x_i) \cdot \log_2 P(x_i) \cdot \sum_j P(y_j) - \sum_i P(x_i) \cdot \sum_j P(y_j) \cdot \log_2 P(y_j) =$$

$$= -\sum_i P(x_i) \cdot \log_2 P(x_i) - \sum_j P(y_j) \cdot \log_2 P(y_j) = H(x) + H(y). \qquad (3.11)$$

5. Энтропия объединения статистически зависимых сообщений определяется через условную энтропию одной из систем.

В этом случае $P(x_i, y_j) = P(x_i) \cdot P(\frac{y_j}{x_i})$, и

$$H(x,y) = H(x) + H\left(\frac{y}{x}\right) = H(y) + H\left(\frac{x}{y}\right), \qquad (3.12)$$

где $H\left(\frac{x}{y}\right) = -\sum_i \sum_j P(x_i, y_j) \cdot \log_2 P\left(\frac{x_i}{y_j}\right) -$ условная энтропия.

3.2 Количество информации

Количество информации есть неопределенность, снимаемая при получении сообщения, исходя из чего количество информации может быть определено как произведение общего числа сообщений K на среднюю энтропию H, приходящая на одно сообщение:

$$I = K \quad \text{(бит)}, \qquad (3.13)$$

или

$$I = K \cdot \sum_{i=1}^{n} P(x_i) \cdot \log_2 P(x_i) \ \text{(бит)}. \qquad (3.14)$$

Количество информации может быть также определено как снятая неопределенность при передаче сообщения, т.е.

$$I = H_{apriori} - H_{aposteriori}. \qquad (3.15)$$

Это определение является наиболее общим и окончательно разрешает вопрос об общности и различии понятий количества информации и энтропии: энтропия - это понятие первичное (исходное), а количество информации - это понятие вторичное (производное).

Имеется еще одна количественная оценка информации - ***объем информации***, который может быть вычислен как

$$Q = k \cdot l, \qquad (3.16)$$

где k - количество переданных символов, l - средняя длина кодовых комбинаций вторичного алфавита. Т.е. ***информационным объемом сообщения*** (объемом информации) называется количество битов в этом сообщении. Подсчет информационного объема сообщения является чисто техническим заданием, так как при таком подсчете содержание сообщения не играет никакой роли.

Пример 3.1. Определить объем информации, содержащийся на 1 странице машинописного текста, если на одной странице умещается по 40 строк и в каждой из которых содержится по 70 символов. Для кодирования символов используется 8-ми разрядный код ASCII.

Решение:

Определим количество символов на одной стороне листа:

40 строк · 70 символов = 2800 символов.

Количество информации = 2800 символов · 1 байт = 2800 байт =

=2800:1024 = 2.74 Кбайт. = 2800·8 =22400 бит = 21.88 Кбит.

Пример 3.2. Какое количество вопросов необходимо задать, чтобы наверняка «угадать» студента из группы численностью 25 человек?

Решение:

$2^n = N$, где $N = 25$. Тогда $n = \log_2 N = 4.64$. Берем ближайшее большее $\log_2 N$, т.е. n=5.

Значит, чтобы определить из 25 человек необходимо получить 5 бит информации, т. е. задать 5 вопросов.

3.3 Основные информационные характеристики

1. ***Скорость передачи информации*** (в отсутствии помех)

$$C = n \cdot H \quad \text{(бит/сек.)}, \qquad (3.17)$$

где n - количество символов, вырабатываемых источником сообщений в единицу времени; H - энтропия, снимая при получении одного символа сообщений, вырабатываемых источником.

Скорость передачи может быть представлена также следующим образом:

$$C = \frac{H}{\tau} \quad \text{(бит/сек.)}, \qquad (3.18)$$

где τ - время передачи одного двоичного символа.

2. ***Пропускная способность*** (емкость канала связи) - максимальная скорость передачи информации по данному каналу связи, т.е. пропускная способность характеризуется максимальной энтропией для данного алфавита (первичного)

$$П = C_{max} = \frac{1}{\tau} \cdot H_{max} \quad \text{(бит/сек.)} \qquad (3.19)$$

Это в отсутствии помех. При наличии помех пропускная способность канала связи вычисляется как произведение количества принятых в секунду знаков n на разность энтропий источника сообщений и источника сообщений относительно принятого сигнала (условная энтропия)

$$П = n \cdot \left[H(A) - H\left(\frac{A}{B}\right)\right] \quad \text{(бит/сек.)} \qquad (3.20)$$

Скорость передачи бит за одну секунду называется 1 Бодом.

3. ***Избыточность сообщений***.

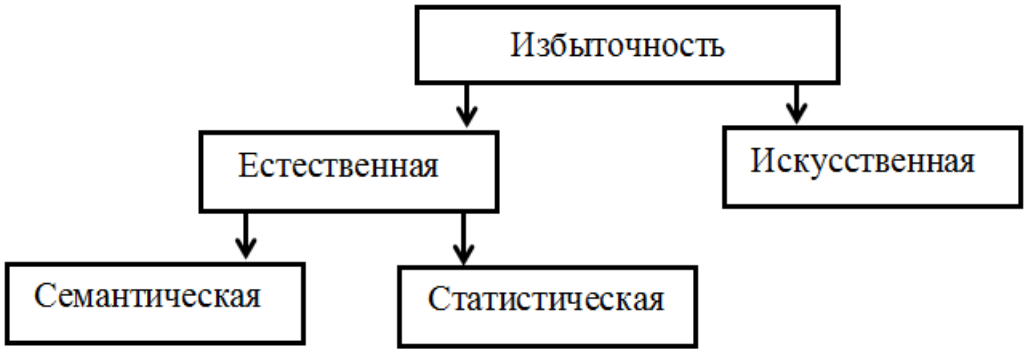


Рис 3.1 Классификация избыточности сообщений

Естественная избыточность присуща любому языку и служит для надежности процесса коммуникации. Она может быть семантической и статистической.

Семантическая избыточность заключается в том, что сообщение может быть выражено короче (телеграмма)

Статистическая избыточность обуславливается либо неравномерным распределением символов в сообщении, либо статистической связью между символами.

Если энтропия источника сообщений не равна максимальной для данного алфавита, то абсолютная недогруженность на символ сообщений такого источника определяется как:

$$\Delta D = (H_{max} - H) \text{ (бит/символ)}, \qquad (3.21)$$

а избыточность определяется как

$$D = \frac{\Delta D}{H_{max}} = \frac{H_{max} - H}{H_{max}} = 1 - \mu, \qquad (3.22)$$

где μ - коэффициент снятия.

H и H_{max} берутся для одного и того же алфавита.

3. ***Помехоустойчивость*** – способность канала связи (системы в целом) противостоять вредному действию помех. Различают атмосферные помехи, индустриальные помехи, импульсные помехи, и т.п. Увеличение помехоустойчивости связано с введением определенной избыточности, т.е. с увеличением объема передаваемой информации (объема сигнала). Если емкость канала связи это допускает, могут быть приняты меры, повышающие надежность передачи. Отметим некоторые из них:
   - Увеличение мощности сигнала;
   - Применение помехоустойчивого кодирования;
   - Применение помехоустойчивых методов модуляции сигнала;
   - Применение помехоустойчивых методов приема;
   - Применение каналов с обратной связью.

Как правило, уменьшение избыточности увеличивает эффективность связи, но ухудшает ее помехоустойчивость.

Лекция 4.

4. Представление числовой информации в цифровых автоматах

Определение: Система счисления (СС) - совокупность приемов и правил для записи чисел цифровыми знаками.

Любая предназначенная для практического использования система счисления должна обеспечивать:

- возможность представления любого числа в рассматриваемом диапазоне величин;
- единственность представления;
- простоту оперирования с числами.

Самый простой способ записи чисел может быть описан выражением:

$$A_D = D_1 + D_2 + \cdots + D_k = \sum_{i=1}^{k} D_i, \qquad (4.1)$$

где A_D -запись числа A в системе счисления D, D_i -символы системы, образующие базу $D=\{D_1, D_2, \ldots\ldots D_k\}$. По этому принципу построены непозиционные системы счисления, для которых значение символа не зависит от его положения в числе. Принцип построения таких систем не сложен. Для их образования используют в основном операции сложения и вычитания. Классическим примером непозиционной системы счисления является Римская система счисления.

4.1 Позиционные системы счисления

В общем случае системы счисления можно построить по следующему принципу:

$$A_B=a_1B_1+a_2B_2+\ldots\ldots+a_nB_n, \qquad (4.2)$$

где A_B - запись числа A в системе счисления с основанием B_i, a_i - символ системы счисления с основанием B_i, B_i - ***база*** или ***основание*** системы счисления.

Если предположить, что $B_i= q^i$, то $B_i=q_iB_{i-1}$. Система счисления, удовлетворяющая равенству $B_i=q_iB_i$, называется ***позиционной*** системой счисления (ПСС). В качестве общего определения ПСС можно использовать следующее утверждение: ***В позиционной системе счисления значения символа числа зависит от его местоположения в числе***. Т.е. один и тот же знак (символ)принимает различное значение в зависимости от его положения в числе. Действительно, например, в числе 888 значение символа (цифры) “8” имеет разный вес в зависимости от его положения в записи числа: восемь, восемьдесят, восемьсот.

Естественная позиционная система счисления имеет место, если q -целое, положительное число. Любая позиционная система характеризуется основанием.

Определение: Основание (или базис) q естественной ПСС - есть количество символов, используемых для изображения числа в данной системе счисления.

Например, для 10-ной системы счисления имеем q=10 и имеем десять символов, принятых для обозначения чисел: 0,1,2,3,4,5,6,7,8,9. Вполне понятно, что возможно бесконечное множество позиционных систем счисления.

Для позиционной системы счисления справедливо равенство:

$$A_q = \sum_{i=-m}^{n} a_i q^i \qquad (4.3)$$

или $A_q = a_n q^n + a_{n-1} q^{n-1} + \ldots\ldots + a_1 q^1 + a_0 q^0 + a_{-1} q^{-1} + a_{-2} q^{-2} + \ldots\ldots + a_{-m} q^{-m}$

На практике используют сокращенную запись чисел:

$$a_n a_{n-1} a_{n-2} \ldots\ldots a_0 a_{-1} a_{-2} \ldots\ldots a_{-m}.$$

В таблице 4.1 приведены числа в десятичной системе счисления и их эквиваленты в других системах счисления.

Таблица 4.1

q=10	Эквиваленты десятичных чисел в других системах счисления					
	q=2	q=3	q=4	q=5	q=8	q=16
0	00000	0000	000	000	00	0
1	00001	0001	001	001	01	1
2	00010	0002	002	002	02	2
3	00011	0010	003	003	03	3
4	00100	0011	010	004	04	4
5	00101	0012	011	010	05	5
6	00110	0020	012	011	06	6
7	00111	0021	013	012	07	7
8	01000	0022	020	013	10	8
9	01001	0100	021	014	11	9
10	01010	0101	022	020	12	A
11	01011	0102	023	021	13	B
12	01100	0110	030	022	14	C
13	01101	0111	031	023	15	D
14	01110	0112	032	024	16	E
15	01111	0120	033	030	17	F
16	10000	0121	100	031	20	10
17	10001	0122	101	032	21	11
18	10010	0200	102	033	22	12
19	10011	0201	103	034	23	13
20	10100	0202	110	040	24	14

21	10101	0210	111	041	25	15
22	10110	0211	112	042	26	16
23	10111	0212	113	043	27	17
24	11000	0220	120	044	30	18
25	11001	0221	121	100	31	19
26	11010	0222	122	101	32	1A

Для любой ПСС справедливо, что основание изображается числом “10” в своей системе, т.е. любое число можно записать в виде:

$$A_q=a_n\cdot 10^n + a_{n-1}\cdot 10^{n-1} + a_{n-2}\cdot 10^{n-2} + \ldots + a_1\cdot 10^1 + a_0\cdot 10^0 + \\ + a_{-1}\cdot 10^{-1} + a_{-2}\cdot 10^{-2} + \ldots + a_{-m}\cdot 10^{-m} \qquad (4.4)$$

Вес разряда p_i числа в ПСС выражается соотношением:

$$p_i=q^i/q^0=q^i , \qquad (4.5)$$

где i -номер разряда.

Если разряд имеет вес $p_i=10^i$, то следующий разряд (старший) имеет вес $p_{i+1}=10^{i+1}$, а соседний младший - $p_{i-1}=10^{i-1}$. Такая взаимосвязь разрядов приводит к необходимости передачи информации между ними. Если в данном разряде накопилось значение единиц равное или больше q, то должна происходить передача единицы в соседний старший разряд (перенос).

Количество позиций (разрядов) в записи числа называется **длиной числа.** Для разных систем счисления характерна разная длина разрядной сетки, необходимая для записи одного и того же числа. Например, $100_{10}=64_{16}=144_8=1210_4=10201_3= 1100100_2$. Очевидно, что одно и тоже число в разных ПСС будет иметь меньшую длину для систем с большим основанием. В техническом аспекте длина числа интерпретируется как **длина разрядной сетки**. Если длина разрядной сетки задана, то это ограничивает максимальное по абсолютному значению число, которое может быть записано в данной разрядной сетке. Если длина разрядной сетки равна n, то

$$A_{q\,max}=q^n-1; \qquad A_{q\,min}=-(q^n-1) \qquad (4.6)$$

<u>Определение:</u> Интервал представления чисел, заключенный между $A_{q\ max}$ и $A_{q\ min}$ называется диапазоном представления чисел (ДП).

$$A_{q\,\min} \le \text{ДП} \le A_{q\ max}$$

Правильный выбор системы счисления - важный практический вопрос. При выборе системы счисления необходимо учитывать:

- основание системы счисления определяет количество устойчивых состояний, которое должен иметь функциональный элемент, выбранный для представления разрядов числа;
- длина числа существенно зависит от основания системы счисления;

- система счисления должна обеспечивать простые алгоритмы выполнения арифметических и логических операций.

Для оценки эффективности использования ПСС введем показатель экономичности системы

$$C=q\cdot N,$$

где q - основание ПСС, N - длина разрядной сетки.

Если принять, что каждый разряд числа представлен не одним элементом с q устойчивыми состояниями, а q элементами, каждый из которых имеет одно устойчивое состояние, то показатель экономичности укажет условное количество оборудования, которое необходимо затратить на представление чисел в этой системе.

Итак, $A_{q\,max}=q^n-1$.

Нужно найти длину разрядной сетки.

$$A_{q\,max}+1=q^n,$$

откуда

$$n=\log_q(A_{q\,max}+1).$$

Тогда для любой системы счисления

$$C=q\cdot N=q\cdot \log_q(A_{q\,max}+1).$$

Для сравнения СС введем относительный показатель экономичности:

$$F=\frac{q\log_q(A_{q\ max}+1)}{2\log_2(A_{2\ max}+1)},$$

позволяющий сравнить любую систему счисления с двоичной. В таблице 4.2 приведены расчетные значения относительного показателя экономичности ПСС. График функции $F(q)$ приведен на рисунке 4.1

Таблица 3.2

q	*F*
2	1.0
3	0.946
4	1.0
6	1.148
8	1.333
10	1.505

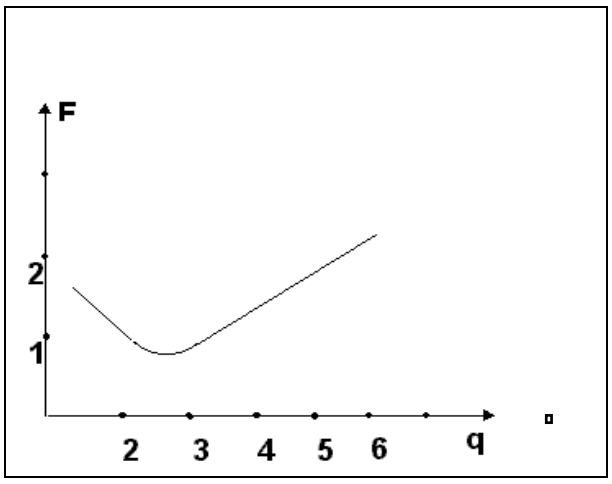


Рис. 4.1

Нижняя точка соответствует минимуму функции, определяемому из условия $\partial F/\partial q=0$, что соответствует значению $q=e$. Следовательно, с точки зрения минимальных затрат условного оборудования, наиболее экономичной является система счисления с основанием e=2.728...

4.2 Перевод числовой информации из одной ПСС в другую

В соответствии с определением, числа в разных ПСС можно представить следующим образом:

$$A_{q_1} = \sum_{i=-m}^{n} a_i q_1^i = \sum_{j}^{k} b_j q_2^j = A_{q_2} \qquad (4.7)$$

Т.е. в общем виде задачу перевода из системы счисления с основанием q_1 в систему с основанием q_2 можно представить как задачу определения коэффициентов (символов) b_j нового ряда, изображающих это число в системе q_2.

Решать эту задачу можно и подбором коэффициентов b_j. Но основная трудность - выбор максимальной степени. Необходимо также иметь в виду, что в обеих системах должны выполняться условия:

$$0 \le a_i \le q_1 - 1 \quad \text{и} \quad 0 \le b_j \le q_2 - 1.$$

4.2.1 Перевод целых чисел делением на основание q_2 новой системы счисления

Целое число A_{q1} в системе с основанием q_2 записывается в виде:

$$A_{q_2} = b_k q_2^k + b_{k-1} q_2^{k-1} + \cdots + b_1 q_2^1 + b_0 q_2^0.$$

Переписав это выражение по схеме Горнера, получим:

$$A_{q_2} = \left(\cdots\left((b_k q_2 + b_{k-1})q_2 + b_{k-2}\right)q_2 + \cdots b_1\right)q_2 + b_0. \qquad (4.8)$$

Разделив выражение на величину основания q_2, получим целую часть

$$\left(\cdots\left((b_k q_2 + b_{k-1})q_2 + b_{k-2}\right)q_2 + \cdots b_1\right)$$

и первый остаток b_0. Разделив целую часть на q_2, получим второй остаток b_1. Повторяя процесс деления k+1 раз получим последнее частное b_k, которое по условию меньше основания системы q_2 и является старшей цифрой числа.

Например, переведем десятичное число 100 в позиционные системы счисления с основаниями q_2=2, 3, 4, 5, 6, 7, 8, 9, 16:

q_2=2

100	2
100	50
0	50
	0

50	2
25	24
24	1
1	

25	2
12	12
12	0
0	

12	2
6	6
6	0
0	

6	2
3	2
2	1
1	

3	2
2	1
1	

q_2=3

100	3
99	33
1	33
	0

33	3
11	9
9	2
2	

11	3
9	3
3	3
3	0
0	

3	3
3	1
1	

q_2=4

100	4
100	25
0	24
	1

25	4
24	6
6	4
4	1
2	

6	4
4	1
1	

q_2=5

100	5
100	20
0	20
	0

20	5
20	4
4	

q_2=6

100	6
96	16
4	12
	4

16	6
12	2
4	

q_2=7 q_2=8 q_2=9 q_2=16

```
 100 | 7          100 | 8          100 | 9          100 | 16
 98  | 14 | 7     96  | 12 | 8     99  | 11 | 9     96  | 6
 2   | 14 | 2     4   | 8  | 1     1   | 9  | 1     4
     | 0          |   4|            |   2|
```

В результате перевода получили:

$100_{10}=1100100_2=10201_3=1210_4=400_5=244_6=202_7=144_8=121_9=64_{16}$

4.2.2 Перевод правильных дробей умножением на основание q_2 новой системы счисления

Пусть исходное число, записанное в системе с основанием q_1:

$$A_{q_1} = a_{-1}q_1^{-1} + a_{-2}q_1^{-2} + \cdots + a_{-m}q_1^{-m}$$

Тогда в новой системе данное число запишется в виде:

$$A_{q_2} = b_{-1}q_2^{-1} + b_{-2}q_2^{-2} + \cdots b_{-s}q_2^{-s}.$$

Переписав это выражение по схеме Горнера, получим:

$$A_{q_2} = q_2^{-1}(b_{-1} + q_2^{-1}(b_{-2} + \cdots + q_2^{-1}b_{-s})\cdots) \qquad (4.9)$$

Если правую часть выражения умножить на q_2, то получится новая неправильная дробь, в целой части которой будет число b_{-1}. Умножив затем оставшуюся дробную часть на величину основания q_2, получим дробь, в целой части которой будет b_{-2}, и т.д. Повторяя процесс умножения s раз, находим все s символов в новой системе счисления. При этом все действия должны выполняться по правилам q_1 арифметики и, следовательно, в целой части получающихся дробей будут проявляться эквиваленты цифр новой системы счисления. Переведем десятичную дробь 0.125 в системы счисления с основаниями, например, q_2=2, 3, 4, 5, 6:

q_2=2 q_2=3 q_2=4 q_2=5 q_2=6

```
0. | 125      0. | 125      0. | 125      0. | 125      0. | 125
×  |  2       ×  |  3       ×  |  4       ×  |  5       ×  |  6
0  | 25       0  | 375      0  |  5       0  | 625      0  | 750
×  |  2       ×  |  3       ×  |  4       ×  |  5       ×  |  6
0  |  5       1  | 125      2  |  0       3  | 125      4  | 500
×  |  2       ×  |  3                    ×  |  5       ×  |  6
1  |  0       0  | 375                   0  | 625      3  | 000
   |          ×  |  3                    ×  |  5
   |          1  | 125                   3  | 125
```

В результате получили следующие результаты:

$0.125_{10}=0.001_2=0.010101_3=0.02_4=0.030303_5=0.043_6$

Для перевода неправильных дробей из одной ПСС в другую необходим раздельный перевод целой и дробной частей по вышеописанным правилам. Например, число из пятеричной системы необходимо перевести в шестеричную систему, т.е. имеем q_1=5, q_2=6, A_{q_1} =431.124_5. Сначала переводим только целую часть числа 431_5 делением на 6, которое в пятеричной системе счисления записывается как 11, т.е. $11_5=6_{10} = 10_6$. Все вычисления делаются по законам пятеричной арифметики:

```
 431 | 11        11 | 11
 33  | 34        33 |  3
 101 | 33
 44  |  1
  2
```

Т.е. получили, что $431_5=312_6$. Затем переводим дробную часть числа 0.124_5 умножением на 11, выполняя все действия по законам пятеричной арифметики:

```
 0. | 124
  × |  11
  1 | 414
  × |  11
 10 | 104
  × |  11
  1 | 144
  × |  11
  2 | 134
и т.д.
```

Отметим, что в результате умножения 0.414×11 получен результат 10.104 (т.е. 0.414×11=10.104), где пятеричное целое число 10_5 записывается уже в символике шестеричной системы счисления, т.е. как 5_6. (Напомним, что в любой позиционной системе счисления, число, равное основанию системы счисления записывается как 10. Поэтому $10_5=5_6$). Итак, $0.124_5=0.1512_6$.

Полученные значения соединяем “десятичной” точкой и получаем результат: $431.124_5=312.1512_6$.

4.2.3 Табличный метод перевода

В самом простейшем виде табличный метод перевода заключается в том, что имеется таблица всех чисел одной ПСС с соответствующими эквивалентами из другой ПСС, и перевод числа заключается в нахождении по таблице соответствующего эквивалента числа в нужной системе счисления. Использование подобных таблиц эквивалентности чисел нецелесообразно,

т.к. они громоздки, требуют большой объем памяти для хранения, неудобны для использования.

Другой вид табличного метода перевода чисел из одной ПСС в другую заключается в том, что имеются таблицы эквивалентов в каждой системе только для цифр этих систем и степеней основания. Задача сводится к тому, что в выражении $A_q = \sum_{i=-m}^{i=n} a_i q^i$ для исходной системы счисления надо подставить эквиваленты из новой системы для всех цифр и степеней основания и произвести соответствующие действия по правилам q_2 арифметики. Полученный результат будет изображать число в новой (q_2) системе счисления.

Например, вышеописанным методом переведем десятичное число 123_{10} в двоичную систему счисления:

Десятичное число	Двоичный эквивалент десятичного числа
1	1
2	10
3	11
10^1	1010
10^2	1100100

Подставляя в десятичное число 123 его двоичные эквиваленты цифр и весов разрядов, и выполнив действия по законам двоичной арифметики, получаем:

$123_{10} = 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0 = 1 \cdot 1100100 + 10 \cdot 1010 + 11 \cdot 1 = 1100100 + 10100 + 11 = 1111011_2$.

Действительно, если перевести 123_{10} в двоичную систему методом деления получается тот же самый результат.

Пример 4.1 Табличным методом перевести число 431.124_5 в шестеричную систему счисления.

Для вычисления нам будет необходимо знать значения символов и весов разрядов пятеричной системы и их эквивалентные значения в представлении шестеричной системы счисления. Пусть имеется такая таблица эквивалентов:

Символы и веса разрядов пятеричной системы (q=5)	Значения символов и весов разрядов для q=5 в шестеричной системе счисления
0	0
1	1
2	2

3	3
4	4
10^0	1
10^1	5
10^2	41
10^3	325
10^{-1}	0.111111…11
10^{-2}	0.0123501…
10^{-3}	0.001433…

$431.124_5=4\cdot41+3\cdot5+1\cdot1+1\cdot0.11111+2\cdot0.01235+4\cdot0.001433=$
$244+23+1+0.11111+0.02514+0.01050=312.15115_6$.

Как видно, результат получился тот же самый, что и при переводе по схеме Горнера.

Сразу отметим, что табличный метод очень удобен при переводе чисел из различных позиционных систем счисления в десятичную систему счисления, т.к. нужно выполнять операции в десятичной арифметике, которые нам более привычны.

4.2.4 Использование промежуточной системы счисления

Этот метод применяют для перевода из десятичной системы в двоичную и наоборот. В качестве промежуточной системы используется шестнадцатеричная или восьмеричная системы счисления. Суть метода заключается в том, что десятичное число переводят, например, в шестнадцатеричную систему счисления, а затем каждый символ полученного числа в шестнадцатеричной системе счисления заменяют четырехразрядным (тетрадой) двоичным эквивалентом. Это следует из того, что шестнадцатеричная система счисления связана с двоичной системой счисления соотношением $16^n=(2^4)^n$. Аналогично для перевода можно использовать восьмеричную систему счисления, но при этом в полученном выражении каждый символ числа заменяется трехразрядным (триадой) двоичным эквивалентом, т.к. восьмеричная система счисления связана с двоичной соотношением: $8^n=(2^3)^n$. В качестве промежуточной системы можно использовать так же и четверичную систему счисления.

Пример 4.2. Число 12345_{10} перевести в двоичную систему счисления.

Решение.

1. Данное число переводим в 16-ричную систему: $12345_{10}=3039_{16}$.

2. Каждый символ полученного числа заменяем двоичной тетрадой и получаем искомый результат: 11000000111001_2.

4.3 Формы представления чисел

Любое число имеет многообразие форм своего представления. Действительно, число 0.02345 можно записать так: $2345 \cdot 10^{-5}$, или $2.345 \cdot 10^{-2}$, или $0.2345 \cdot 10^{-1}$ и т.д. Разнообразие форм в записи одного числа может послужить причиной затруднений для работы цифрового устройства. Во избежание этого нужно либо создать специальные алгоритмы распознавания числа, либо указывать каждый раз форму его записи, что является более просто.

Существует две формы записи чисел: естественная и нормальная.

При естественной форме число записывается в естественном натуральном виде, в соответствии со смысловым значением: 123 - целое число, 0.02345 - правильная дробь, 12.345 - неправильная дробь.

При нормальной форме запись одного числа может принимать разный вид, в зависимости о ограничений, накладываемых на ее форму. Например, $123 = 1.23 \cdot 10^2 = 0.123 \cdot 10^3 = 0.123000 \cdot 10^{-3}$ и т.д.

Определение: Автоматное (машинное) изображения числа- представление числа А в разрядной сетке цифрового автомата.

Условно автоматное изображение числа A обозначается $[A]$. Справедливо соотношение: $A=[A] \cdot K_A$, где K_A - коэффициент, величина которого зависит от формы представления числа в автомате.

4.3.1 Представление чисел в формате с фиксированной (запятой) точкой

Естественная форма представления числа в ЭВМ (или ином цифровом автомате) характеризуется тем, что положение его разрядов в автоматном изображении остается всегда постоянным, независимо от самого числа. Т.е. длина разрядной сетки строго фиксирована, а так же фиксирована ДРС целой и дробной частей. В связи с этим существует так же и другое название этой формы представления чисел - **представление чисел в форме с фиксированной запятой (точкой)**. В ЭВМ эта форма используется преимущественно для представления целых чисел. Так как числа бывают положительные и отрицательные, то в разрядной сетке при их машинном представлении один разряд отводится под знак числа, а остальные образуют поле числа. Знаковый разряд может располагаться как в начале, так и в конце числа. Принято, что “0” - соответствует положительному, а “1” - отрицательному числам.

Если поле числа включает n разрядов, то диапазон представления целых чисел в этом случае ограничивается значениями $-(2^n-1) \div +(2^n-1)$. Для приведенного здесь примера имеем n=11, и тогда диапазон представления целых чисел: -2047 ÷ +2047.

При записи на бумаге принято целую часть числа от дробной отделять запятой (в США - точкой). Также первоначально стали представлять числа и в ЭВМ, но оказалось, что в достаточно длинном 32-разрядном машинном слове помещается число, которое после перевода в десятичную систему счисления содержит не более десяти цифр:

$$\pm(2^{31}-1) = \pm 2\ 147\ 483\ 647.$$

При инженерных и научных расчётах часто требуется более трёх знаков после запятой. Если под дробную часть отвести четыре разряда, т. е. зафиксировать запятую перед четвёртым разрядом справа, то самое большое по модулю число будет меньше 220 000:

$$214748.3647\ .$$

Этого совершенно недостаточно для практических расчётов, поэтому для представления действительных чисел используют форму представления чисел с плавающей точкой.

Целые числа в программах, в основном, используются для нумерации, индексации и диапазона

$$-2\ 147\ 483\ 647 \div +2\ 147\ 483\ 647,$$

что вполне хватает.

Итак, под числами с фиксированной запятой понимают целые числа. На практике применяют 8-ми, 16-и, 32-х и 64-х разрядные форматы (со знаком или без знака) числа с фиксированной запятой. Пример записи числа $-50010 = 111110100_2$ в 16-битном формате со знаком

1	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0

Крайний левый разряд отведён под знак числа: 0 - плюс, 1 - минус.

Для того, чтобы определить, сколько двоичных разрядов будет занимать целое положительное число N после перевода из десятичной системы счисления в двоичную, нужно вычислить $k = log_2 N$ и взять ближайшее большее, чем k, целое число.

Примеры

$\log_2 15 = 3.90$ $k = 4$ $15_{10} = 1111_2$

$\log_2 16 = 4$ k = 5 $16_{10} = 1\ 0000_2$
$\log_2 500 = 8{,}97$ k = 9 $500_{10} = 1\ 1111\ 0100_2$
$\log_2 1000 = 9.96$ k = 10 $1000_{10} = 11\ 1110\ 1000_2$
$\log_2 1024 = 10$ k = 11 $1024_{10} = 100\ 0000\ 0000_2$

4.3.2 Представление чисел в форме с плавающей запятой (точкой)

Как известно, для записи очень больших, или очень маленьких по модулю чисел используют полулогарифмическую (нормальную) форму представления чисел. Например, расстояние от Земли до Солнца удобнее записывать не в виде 149 600 000 км, а в виде $1.496 \cdot 10^8$ км, или даже с меньшей точностью в виде $1.5 \cdot 10^8$ км.

Многие физические величины практически можно записать только в нормальной форме. Например, массу электрона m_e:

$$m_e = 9.1091 \cdot 10^{-31} \text{ кг.}$$

В нормальной форме число A запишется как

$$A_н = m_A \cdot p_{A,} = m_A \cdot q^p , \qquad (4.10)$$

где m_A - мантисса числа A, q – основание, p – порядок, q^p характеристика числа A.

Как видно из ранее изложенного, такое представление чисел не однозначно; для определения обычно вводят некоторые ограничения. Наиболее распространено для представления ограничения вида:

$$q^{-1} \le |m_A| < 1, \qquad (4.11)$$

где q - основание системы счисления.

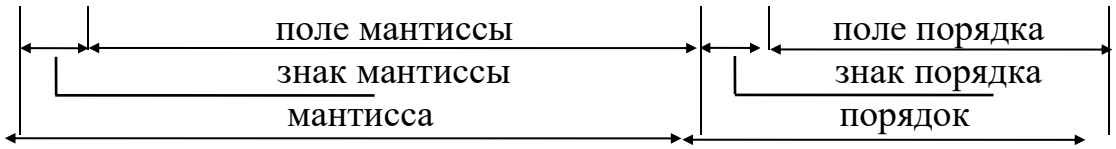
Применительно к ЭВМ условие (4.11) часто записывают в виде:

$$0.5 \le |m_A| < 1$$

Определение: Форма представления чисел, для которой справедливо условие $q^{-1} \le |m_A| < 1$, называется нормализованной формой представления числа.

Поскольку в этом случае абсолютное значение мантиссы лежит в пределах от q^{-1} до $1\text{-}q^{-n}$, где n - количество разрядов для изображения мантиссы без знака, положение разрядов числа в его автоматном представлении не постоянно. Поэтому такую форму представления называют **формой представления с плавающей точкой**. Формат машинного изображения числа с плавающей точкой должен содержать знаковые части и поля для мантиссы и порядка. Выделяются специальные разряды для изображения знака числа (мантиссы) и знака порядка. Например, для 16-разрядной сетки условно формат можно представить в виде:

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



Рассмотрим пример записи чисел в формате с плавающей точкой. Пусть в разрядной сетке необходимо записать двоичные числа:

1. $A_1 = -\ 101100.1111_2$
2. $A_2 = 0.000110010111_2$

Прежде всего запишем эти числа в нормальной форме. Порядок чисел выбираем таким образом, чтобы для них выполнялось условие 4.11. Тогда получаем:

$A_1 = -\ 0.101101111 \cdot 10^{101}$;

$A_2 = \ 0.110010111 \cdot 10^{-11}$.

Поскольку для изображения выделено, например, пять разрядов и один для знака, то их машинные изображения и машинные изображения мантисс будут иметь вид:

$[p_{A_1}] = 000101$ $\qquad$ $[p_{A_2}] = 100011$

$[m_{A_1}] = 1.101101111$ $\qquad$ $[m_{A_2}] = 0.110010111$

Число A_1:

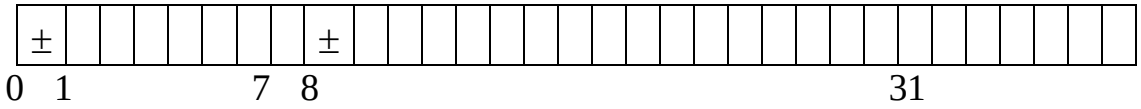
1	1	0	1	1	0	1	1	1	1	0	0	0	1	0	1

Число A_2:

0	1	1	0	0	1	0	1	1	1	1	0	0	0	1	1

Часто формат с плавающей точкой представляют и в другом виде, при котором, например, сначала записывается порядок, а затем – мантисса.

Например, простейший 32-разрядный формат числа с плавающей запятой имеет вид:



В разряды 0 - 7 записывается порядок числа. В разряде 0 - знак порядка. В разряды с 8-го по 31-й помещается мантисса. В 8-м разряде - знак мантиссы.

Представим, например, в форме с плавающей точкой число $A = 12.5_{10} = 1100.1_2$. В нормализованном виде

$$A = (\ (12.5/2^4) \cdot 2^4\ )_{10} = (0.78125 \cdot 2^4)_{10} = (0.11001 \cdot 10^{100})_2$$

(Конечно, это число можно перевести в 2-ю по схеме Горнера и далее записать его в нормализованном виде по правилам двоичной арифметики: $A=(1100.1=0.11001 \cdot 10^{100})_2$).

Тогда в 32-разрядной ячейке памяти число A выглядит так:

0	0000100	0	11001000000000000000000
+	4	+	0.78125

Большинство людей за всю свою жизнь ни разу не сталкивалось с числом, состоящим из такого большого количества цифр, что для его записи не хватало одной строки листа бумаги. К сожалению, в ЭВМ числа часто не помещаются в заданный формат.

Самое большое число с плавающей точкой определяется характеристикой $A = q^p$.

При $q = 2$ и $p_{\max} = 2^7-1 = 127$ получаем $A_{max}=2^{127}=10^{38}$.

Данная оценка удобна, но не совсем точна. Запишем в рассмотренном выше 32-разрядном формате самое большое число:

0	1111111	0	11111111111111111111111

Переведём полученное число в десятичную систему счисления. Сначала переведём мантиссу:

$$0.111\ 1111\ 1111\ 1111\ 1111\ 1111_2 = 1 - 2^{-23}.$$

Умножив мантиссу на характеристику, получим в десятичной системе счисления самое большое число, которое можно записать в 32-разрядном формате с плавающей точкой:

$$(1 - 2^{-23})\cdot 2^{127} = 1{,}70141\cdot 10^{38}$$

Самое маленькое положительное число, которое можно записать в рассматриваемом формате, отличается по форме только знаковым разрядом порядка и в десятичной системе счисления оно имеет вид:

$$(1 - 2^{-23})\cdot 2^{-127} = 5{,}87747\cdot 10^{-39}.$$

Итак, модуль числа A в рассматриваемом формате должен находиться в диапазоне:

$$5{,}87747\cdot 10^{-39} \le |A| \le 1{,}70141\cdot 10^{38}$$

4.3.3 Представление отрицательных чисел

Основной операцией двоичного сумматора является операция сложения. Операцию арифметического вычитания заменяют операцией алгебраического сложения:

$$A\text{-}B = A+(\text{-}B)$$

Для машинного представления отрицательных чисел используют коды: **прямой, дополнительный, обратный**.

Прямой код числа $A=\text{-}0a_1a_2.....a_n$- есть машинное изображение этого числа в виде $[A]_{np=}1a_1a_2....a_n$. В прямом коде все цифровые разряды отрицательного числа остаются неизменными, а в знаковой части записывается единица. Положительное число в прямом коде не меняет своего изображения. Например: если A=-0101110 , то $[A]_{np}$=1101110, т.е. в знаковой части записывается единица; если A= 0110111, то $[A]_{np}$=0110111, т.е. положительное число не меняет своего изображения.

В прямом коде в разрядную сетку можно записать следующее максимальное по абсолютному значению число: $A_{пр\ мах}$=01111....1111=2^n-1, где n - количество разрядов разрядной сетки. Диапазон изменения машинных изображений для прямого кода составит: -(2^n-1)≤ $[A]_{пр}$ ≤ (2^n-1).

Если положить, что для представления чисел используется формат с фиксированной точкой, и при этом числа представляются в виде правильной дроби, т.е. -1<$[A]_ф$<1 , то в этом случае максимальное по абсолютному значению число в прямом коде запишется как $A_{пр\ мах}$=0.1111...1111=1-2^n, где n -количество разрядов разрядной сетки цифрового автомата. И тогда правила преобразования чисел в прямой код можно сформулировать в виде:

$$[A]_{\text{пр}} = \begin{cases} A, \text{если} \quad A \geq 0, \\ 1 + |A|, \text{если} \quad A < 0 \end{cases} \qquad (4.12)$$

Обратный код числа A=-$0a_1a_2....a_n$ - есть такое машинное изображение этого числа $[A]_{\text{об}} = 1\bar{a}_1\bar{a}_2 \cdots \bar{a}_n$, для которого $\bar{a}_i = 0$, если $a_i = 1$, и $\bar{a}_i = 1$, если $a_i = 0$. Из определения следует, что обратный код двоичного числа является инверсным изображением самого числа, в котором все разряды исходного числа принимают инверсное (обратное) значение, т.е. все нули заменяются на единицы, а единицы на нули. Например, если A=-01001100111, то $[A]_{об}$=1 0110011000 .

Для обратного кода чисел, представленных в форме с запятой, фиксированной перед старшим разрядом (т.е. – правильная дробь), справедливо соотношение:

$$|A| + [A]_{\text{об}} = q - q^{-n}, \qquad (4.13)$$

где $|A|$ - абсолютная величина A, n - количество разрядов после запятой в изображении числа.

Правила преобразования чисел в обратный код можно сформулировать следующим образом:

$$[A]_{\text{об}} = \begin{cases} A, \text{если} \quad A \geq 0, \\ q - q^{-n} - |A|, \text{если} \quad A < 0. \end{cases} \qquad (4.14)$$

В обратном коде можно изображать максимальное положительное число $[A]_{\text{об мах}}$ =0.111...111=(1-2^{-n}) и наибольшее отрицательное число $[A]_{\text{об min}}$ = - 0.111...111= -(1-2^{-n}) . Следует иметь в виду неоднозначное изображение нуля в обратном коде: +0 изображается 0.000...000, то -0 изображается 1.111...111.

Если необходимо преобразовать в обратный код целое число, то это можно выполнить в соответствии с формулой (3.15), которая получается из (4.14).

$$[A]_{\text{об}} = \begin{cases} A, \text{если} \quad A \geq 0, \\ 2^n - 1 - |A|, \text{если} \quad A < 0, \end{cases} \qquad (4.15)$$

где n – количество разрядов в формате числа A (в n входит и знаковый разряд), или проинвертировать все числовые разряды слова, т.е. заменить все единицы в двоичном коде на нули, а нули на единицы.

Например, для представления числа $-100_{10}=-1100100_2$ в 10-разрядной сетке двоичного кода имеем: 2^{10}-1-100=1024-1-100=923=$39B_{16}$ = 1 110011011.

Конечно, намного проще получить обратный код двоичного числа в заданной разрядной сетке инвертированием его прямого кода (только цифровой части). $[A]_{пр}$ = 1 001100100. Тогда $[A]_{об}$=1 110011011.

Дополнительный код числа $A=-0a_1a_2....a_n$ -есть такое машинное изображение этого числа $[A]_д = 1\bar{a}_1\bar{a}_2\cdots\bar{a}_n$, которого $\bar{a}_i = 0$, если $a_i = 1$, и $\bar{a}_i = 1$, если $a_i = 0$, за исключением последнего значащего разряда, для которого $\bar{a}_n = 1$ при $a_n = 1$.

Для дополнительного кода чисел, представленных в форме с запятой, фиксированной перед старшим разрядом (т.е. – правильная дробь), справедливо соотношение:

$$|A| + [A]_д = q. \qquad (4.16)$$

Т.е., дополнительный код числа является математическим дополнением основанию системы счисления.

Т.к. положительные числа не меняют своего изображения в дополнительном коде, то правила преобразования в дополнительный код можно записать следующим образом:

$$[A]_д = \begin{cases} A, \text{если } A \geq 0, \\ q - |A|, \text{если } A < 0. \end{cases} \qquad (4.17)$$

Сравнивая 4.16 и 4.17 видно, что

$$[A]_д = [A]_{об} + q^{-n}.$$

Это выражение используют для получения дополнительного кода отрицательного числа следующим образом:

Сначала инвертируется цифровая часть исходного кода, в результате чего получается его обратный код, затем добавляется единица в младший разряд цифровой части числа и тем самым получается дополнительный кода исходного изображения.

Пример 4.3. Найти обратный и дополнительный код числа A= - 0.11011100110.

Решение. Используя определение обратного кода, получаем $[A]_{об}$=100100011001.

Для нахождения дополнительного кода прибавим единицу к младшему разряду полученного изображения:

$$\begin{array}{r} 100100011001 \\ + \qquad\qquad 1 \\ \hline [A]_д = 100100011010 \end{array}$$

Ответ: $[A]_{об}$ = 100100011001; $[A]_д$ = 100100011010

Для получения двоичного дополнительного кода целого отрицательного используется формула:

$$[A]_{д} = [A]_{об} + 1 = 2^n - |A|. \qquad (4.18)$$

Например, для представления числа $-100_{10}=-1100100_2$ в дополнительном коде в 10-разрядной сетке в соответствии с (3.18) получаем:

$$[A]_{д} = 2^{10}\text{-}100=1024\text{-}100=924=39C_{16} = 1\ 110011100$$

Использование различных способов изображения отрицательных чисел в цифровом устройстве (например, ЭВМ) обуславливает целый ряд особенностей выполнения операции алгебраического сложения двоичных чисел.

4.3.4 Погрешности представления чисел

Представление цифровой информации в ЭВМ, как правило, влечет за собой появление погрешностей (ошибок), величина которых зависит от формы представления чисел и от длины разрядной сетки цифрового автомата.

Абсолютная погрешность представления – разность между истинным значением входной величины А и ее значением, полученным из машинного изображения A_M, т.е. $\Delta[A]=A-A_M$.

Относительная погрешность представления – величина, определяемая соотношением:

$$\delta[A] = \frac{\Delta[A]}{A_M} \qquad (4.19)$$

Пример 4.4. Часто принимают значение π равным 3.14. Однако эта величина может быть получена и с большей точностью. Если принять, что точное значение π=3.141159265, то абсолютная погрешность будет равна в данном случае $\Delta[\pi]$=0.00159265, а относительная погрешность - $5.072133\cdot10^{-4}$.

Часто некоторая величина в одной системе счисления имеет конечное значение, а в другой системе счисления становится бесконечной дробью. Например, число 0.1_{10} имеет конечное десятичное представление, но в двоичной системе это число выражается бесконечной (в данном случае периодической) дробью: $0.0001100110011..._2$. Следовательно, при переводе чисел из одной системы счисления в другую неизбежно возникают погрешности, оценить которые нетрудно, если известны истинные значения входных чисел.

Если учесть, что для машинного представления чисел справедливо соотношение: $A=[A]\cdot K_A$, где K_A – коэффициент, величина которого зависит от формы представления числа в цифровом автомате, и при этом масштабный коэффициент K_A выбирают так, чтобы абсолютное значение машинного изображения числа A в системе счисления с основанием q=2 было всегда меньше 1. Для представления чисел в ЭВМ в общем случае примем K_A=1. Также принимая во внимание ограниченность разрядной сетки для представления правильной дроби ($-1<[A]_Ф<1$, где $[A]_Ф$ – машинное изображение числа в формате с фиксированной точкой), то с учетом вышесказанного будем иметь:

$$A_q = K_A\left[a_{-1}q^{-1} + a_{-2}q^{-2} + \cdots + a_{-n}q^{-n} + a_{-(n+1)}q^{-(n+1)} + \cdots\right] = \\ = \sum_{i=-1}^{\infty} a_i q^i \qquad (4.20)$$

Т.к. длина разрядной сетки равна n разрядов после запятой, то абсолютная погрешность перевода десятичного числа в систему с основанием q будет равна:

$$\Delta[A] = a_{-(n+1)}q^{-(n+1)} + \cdots + a_{-(n+m)}q^{-(n+m)} + \cdots = \sum_{i=-(n+1)}^{\infty} a_i q^i. \quad (4.21)$$

Оценим погрешность представления чисел в двоичной системе счисления. Имеем q=2. Максимальное значение погрешности будет, если a_i=1, и тогда имеем:

$$\Delta[A]_{max} = \sum_{i=-(n+1)}^{\infty} 1\cdot 2^i = 2^{-n}\sum_{i=-1}^{\infty} 2^i = 2^{-n} \qquad (4.22)$$

Т.е. максимальная погрешность перевода десятичного числа в двоичное представление не будет превышать единицы младшего разряда разрядной сетки. Вполне очевидно, что минимальная погрешность перевода будет равна нулю. Усредненная погрешность перевода чисел, равная среднему арифметическому минимальной и максимальной погрешностей, будет равна:

$$\Delta[A] = \frac{0+2^{-n}}{2} = 0.5\cdot 2^{-n}. \qquad (4.23)$$

Для представления чисел в формате с фиксированной точкой абсолютное значение машинного изображения числа (для правильной дроби) равно:

$$2^{-n} \le \left|[A]_ф\right| \le 1 - 2^{-n}.$$

Следовательно, относительная погрешность представления минимального значения числа составит:

$$\delta[A]_{ф\ min} = \frac{\Delta[A]}{[A]_{ф\ max}} = \frac{0.5\cdot 2^{-n}}{1-2^{-n}}. \qquad (4.24)$$

Для ЭВМ, как правило, n=16÷64, поэтому $1>>2^{-n}$ и тогда получаем:

$$\delta[A]_{ф\ min} = \frac{\Delta[A]}{[A]_{ф\ max}} = \frac{0.5\cdot 2^{-n}}{1} = 0.5\cdot 2^{-n}. \qquad (4.25)$$

Аналогично, для оценки максимального значения получаем:

$$\delta[A]_{\text{ф}\ max} = \frac{\Delta[A]}{[A]_{\text{ф}\ min}} = \frac{0.5\cdot 2^{-n}}{2^{-n}} = 0.5. \qquad (4.26)$$

Из последнего выражения видно, что погрешности представления малых чисел в формате с фиксированной точкой могут быть очень значительными. Поэтому формат с фиксированной точкой используется в ЭВМ только для представления целых чисел.

Для представления чисел в формате с плавающей точкой абсолютное значение мантиссы должно удовлетворять условию:

$$2^{-1} \le |[m_A]| \le 1 - 2^{-n}. \qquad (4.27)$$

Погрешность $\Delta[A]_{max}=2^{-n}$ есть погрешность мантиссы. Для нахождения погрешности представления числа в формате с плавающей точкой величину этой погрешности нужно умножить на величину порядка числа p_A, и тогда получаем:

$$\delta[A]_{\text{П}\ max} = \frac{0.5\cdot 2^{-n} p_A}{2^{-1} p_A} = 2^{-n} \qquad (4.28)$$

$$\delta[A]_{\text{П}\ min} = \frac{0.5\cdot 2^{-n} p_A}{(1-2^{-n}) p_A} = 2^{-(n+1)}, \qquad (4.29)$$

где n – количество разрядов для представления мантиссы числа. Из последних выражений следует, что погрешность представления чисел в формате с плавающей точкой не зависит от величины числа и определяется длиной разрядной сетки представления мантиссы числа.

Рассмотрим пример по оценке представления чисел в ЭВМ в формате с плавающей точкой.

Как известно из алгебры, в 10-ной системе счисления дробь, знаменатель которой не может быть приведён к виду 10^n, n = 1, 2, ..., не может быть представлена точно в виде десятичной дроби. Например,

$$\frac{7}{50} = \frac{14}{100} = \frac{14}{10^2} = 0.14;$$

$$\frac{5}{24} = \frac{5}{(3\cdot 2\cdot 2\cdot 2\cdot 2)} = \quad 0.208333(3);$$

$$\frac{2}{3} = \quad 0.666(6)$$

Аналогичным свойством обладает и двоичная система счисления.

Точно в формате с плавающей запятой представляется число, модуль которого можно записать в виде следующей несократимой дроби:

$\frac{A}{2^n}$, где A и n – целые положительные числа.

Может показаться неожиданным, что 0,1 (одна десятая) не может быть представлена точно в формате с плавающей запятой, а 0,25 – представима:

$$0.1_{10} = \frac{1}{(2\cdot 5)} = 0.00011001100\ldots = 0.00011(0011)_2;$$

$$0.125_{10} = \frac{1}{8} = \frac{1}{2^3} = 0.001_2.$$

Количество двоичных знаков в числе *x* определяется количеством разрядов в мантиссе. В рассматриваемом нами 32-разрядном формате длина мантиссы, как мы определили ранее, составляет 23 двоичных разрядов. Тогда количество десятичных знаков

$$R = \log_{10} 2^{23} = 23 \cdot \log_{10} 2 \cong 6.92369 \cong 7.$$

Целое число *N* не имеет погрешности в формате с плавающей запятой, если в двоичном виде номер его самого младшего разряда, содержащего единицу, меньше или равен числу разрядов в мантиссе. Например, $N = 2^{33}$ представляется точно, а $N = 2^{33} - 1$, как показано в приведённом ниже примере, имеет погрешность.

Теория погрешностей вычислений в формате с плавающей запятой очень сложна. Приведем примеры возникновения таких погрешностей.

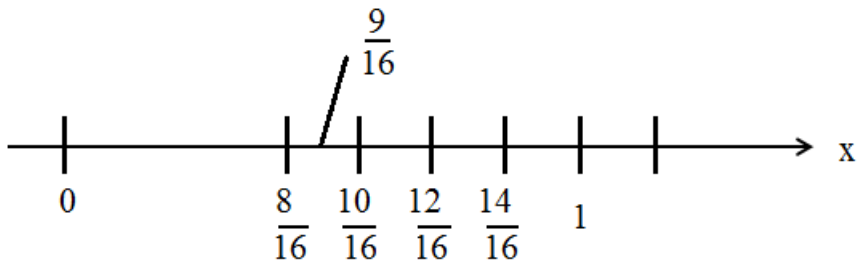
Для простоты рассмотрим семибитный формат числа с плавающей запятой:

±			±			
0	1	2	3	4	5	6

В этом формате при значении порядка *p* = 0 точно представимы только 4 числа: 1/2=4/8; 5/8; 6/8 и 7/8. В памяти ЭВМ эти числа хранятся так:

0	0	0	**0**	1	0	0
0	0	0	**0**	1	0	1
0	0	0	**0**	1	1	0
0	0	0	**0**	1	1	1

Разместим эти числа на числовой оси:



Все числа, заключённые между 1/2 и 5/8 в выбранном нами формате не могут быть представлены точно. Они будут округлены либо в меньшую сторону до 1/2, либо в большую до 5/8. Попытаемся представить в выбранном формате 9/16 = $(0.1001)_2$:

0	0	0	**0**	1	0	0	1

Младший разряд вышел за пределы формата и вместо 9/16 в памяти ЭВМ будет храниться 8/16. В общем случае погрешность представления числа с плавающей запятой

$0 \le \Delta x < 2^{-m} \cdot 2^{p}$, где m-число разрядов в мантиссе, исключая знаковый.

На практике для представления чисел с плавающей запятой чаще всего используются машинные слова длиной либо 32 либо 64 бита. Запишем в 32-разрядное слово число $x = 2^{33} - 1$. В двоичной системе счисления - это число из 33-х единиц. Найдём мантиссу и порядок этого числа в нормализованном виде:

$x = \frac{2^{33}-1}{2^{33}} \cdot 2^{33} = 0.111\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 111 \cdot 10^{100001}$.

Для того чтобы, не теряя точности, разместить это число, требуется слово длиною 41 бит (7 бит под порядок и 34 бита под мантиссу). Разместив это число в 32-битном формате с 24-разрядной мантиссой, считая знак, мы потеряем 10 значащих цифр исходного числа x:

0	0100001	0	11111111111111111111111	1 1 1 1 1 1 1 1 1 1
0	1 7	8	31	10 разрядов

23 разряда

Вычислим погрешность числа x в 32-разрядном двоичном формате с плавающей запятой:

$$\Delta x = x - m_x \cdot p_x = (2^{33} - 1) - (1 - 2^{-23}) \cdot 2^{33} = 2^{10} - 1 = 1023,$$

где m_x – 24-разрядная мантисса.

В десятичной системе счисления

$2^{33} = 8\ 589\ 934\ 591;$

$(1- 2^{-23}) \cdot 2^{33} = 8\ 589\ 933\ 568,$

т.е. в форме с плавающей точкой точно представлены только 6 старших разрядов исходного числа. Все числа в диапазоне $8\ 589\ 933\ 568 \le x \le 8\ 589\ 934\ 591$ в 32-разрядном двоичном формате с плавающей запятой будут представлены одним числом - 8 589 933 568.

Можно оценить погрешность представления числа N в формате с плавающей запятой без перевода из десятичной системы счисления в двоичную. Для простоты рассмотрим пример, в котором мантисса в формате с плавающей запятой имеет семь двоичных разрядов, а N=1234. В двоичном представлении N имеет 11 разрядов:

$$]\log_2 1234[+ 1 = 11, \text{где }]x[- \text{целая часть числа } x.$$

Так как мантисса имеет семь разрядов, то из числа N при записи в ячейку будут отброшены 11 – 7 = 4 разряда. Далее

- разделив N на 2^4 получаем 1234/16 = 77.125;
- вычислим погрешность: $\Delta = 0{,}125 \cdot 16 = 2$.

Проверим результат прямым переводом N в двоичную систему счисления:

1234 = 1024 + 128+64+16+2= 10011010010_2;

Мантисса m = 0, 1001101_2;

Порядок $p = 1011_2$.

0	1	0	1	1	**0**	1	0	0	1	1	0	1	0	0	1	0
+	1			4	+	6				12		остаток				

Переведём полученное число с плавающей запятой в десятичную систему счисления:

$N_{пл}$=0.1001101·10^{1011}=10011010000=2^{10}+2^7+2^6+2^4=

=1024+128+64+16=1232.

Δ=N-$N_{пл}$=1234-1232=2.

Следует сразу отметить, что погрешности возникают и при выполнении арифметических операциях над числами в формате с плавающей точкой, что будет рассмотрено позже.

Лекция 5.

5. Основы двоичной арифметики

5.1 Формальные правила двоичной арифметики

Простота выполнения арифметических действий в двоичной системе счисления очевидна:

Сложение	Вычитание	Умножение
0+0=0	0-0=0	0×0=0
0+1=1	1-0=1	0×1=0
1+0=1	1-1=0	1×0=0
1+1=(1)0 (единица переноса в старший разряд)	0-1=(1)1 (заем единицы в старшем разряде)	1×1=1

В основе арифметическо-логического устройства любой ЭВМ лежит сумматор, для которого разработаны алгоритмы выполнения арифметических операций. В общем случае выполнение арифметических действий сумматором представляется простым выражением:

$$C=A \nabla B,$$

где ∇ - знак арифметического действия. Так как ЭВМ, как и любой другой цифровой автомат, оперирует только с машинным (автоматным) изображением чисел, то правильнее бы стоило записать:

$$[C]=[A]\nabla[B],$$

где [ ] –обозначено машинное представление операндов. (**Операнды** – это числа, непосредственно участвующие в арифметической операции).

Рассмотрим формальные правила выполнения арифметических операций сложения на уровне разрядов операндов.

На основе правил двоичной арифметики можно записать правила сложения двоичных цифр так, как показано в таблице 5.1 и 5.2, где a_i, b_i – разряды операндов A и B соответственно, c_i – разряд суммы, n_i –перенос из данного разряда в соседний (старший) разряд.

Двоичный полусумматор – устройство, выполняющее арифметические действия в соответствии с таблицей 5.1

Как видно из таблицы 4.1 появление единицы переноса при сложении двух разрядов несколько изменяет правила сложения двоичных цифр. Обобщая вышеизложенное, можно сформулировать правила поразрядных действий при сложении операндов A и B:

$$a_i+b_i+n_{i-1}=c_i+n_{i,}$$

где n_{i-1} – перенос из (i-1) –го разряда, n_i – перенос в (i+1) разряд.

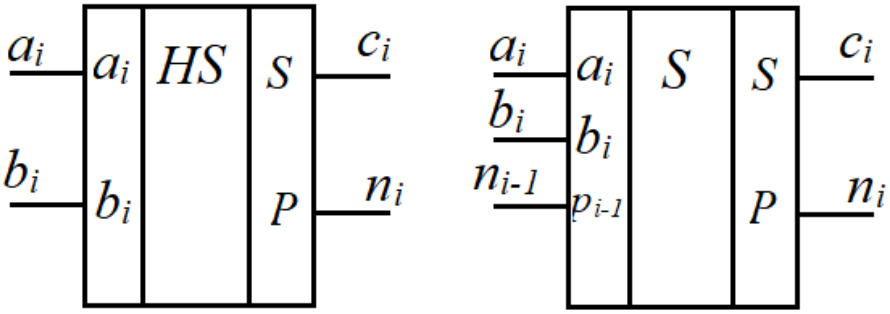
Таблица 5.1

a_i	b_i	c_i	n_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Таблица 5.2

a_i	b_i	n_{i-1}	c_i	n_i
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

Двоичный сумматор – устройство, выполняющее арифметические действия в соответствии с таблицей 5.2. Условные обозначения полусумматора и сумматора приведены на рисунке 5.1а и 5.1б соответственно



а б

Рис. 5.1

5.2 Сложение чисел в форме с фиксированной запятой на двоичных сумматорах

5.2.1 Двоичный сумматор прямого кода (ДСПК)

Двоичный сумматор, в котором отсутствует цепь поразрядного переноса между старшим цифровым и знаковым разрядом, называется двоичным сумматором прямого кода – ДСПК. Условная схема ДСПК может быть представлена в виде, показанном на на рис. 5.2.

Как видно из таблицы 5.1 появление единицы переноса при сложении двух разрядов несколько изменяет правила сложения двоичных цифр. Обобщая вышеизложенное, можно сформулировать правила поразрядных действий при сложении операндов A и B:

$$a_i+b_i+n_{i-1}=c_i+n_{i,}$$

где n_{i-1} – перенос из (i-1) –го разряда, n_i – перенос в (i+1) разряд.

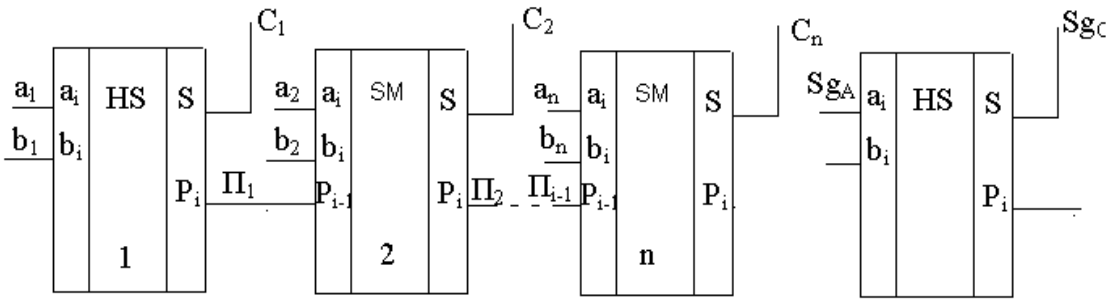


Рис. 5.2.

На ДСПК можно складывать только числа, имеющие одинаковые знаки, т.е. такой сумматор не может выполнять операцию алгебраического сложения.

Пусть заданы операнды:

$$[A]_{пр}=Sg_A a_1 a_2 \ldots . a_n \ , \qquad [B]_{пр}=Sg_B b_1 b_2 \ldots . . b_n,$$

где Sg_A, Sg_B – соответственно содержимое знаковых разрядов изображений для A и B (символ происходит от английского слова sign – знак); a_i, b_i – цифровые разряды изображений. Если $Sg_A=Sg_B$, то сумма чисел будет иметь знак любого из слагаемых, а цифровая часть результата получится после сложения цифровых частей операндов.

Пример 5.1. Сложить числа A=0.1001 и B=0.0110 на ДСПК.

$[A]_{пр}$=0,1001	Sg_A=0	$[A]_{пр}$=1,1001	Sg_A=1
$[B]_{пр}$=0,0110	Sg_B=0	$[B]_{пр}$=1,0110	Sg_B=1
$[C]_{пр}$=0,1111	Sg_C=0	$[C]_{пр}$=1,1111	Sg_C=1

Если абсолютное значение суммы будет больше единицы, то имеет место переполнение разрядной сетки ДСПК. Как видно, признак переполнения разрядной сетки ДСПК – появление единицы переноса из старшего разряда цифровой части сумматора. В этом случае должен вырабатываться сигнал переполнения φ=1, по которому происходит корректировка масштабных коэффициентов или автоматическая остановка машины.

5.2.2 Двоичный сумматор дополнительного кода (ДСДК)

ДСДК – сумматор, оперирующий изображениями чисел в дополнительном коде. Характерная особенность ДСДК – наличие цепи поразрядного переноса из старшего разряда цифровой части в знаковый разряд. Условная схема ДСДК представлена на рис. 5.3.

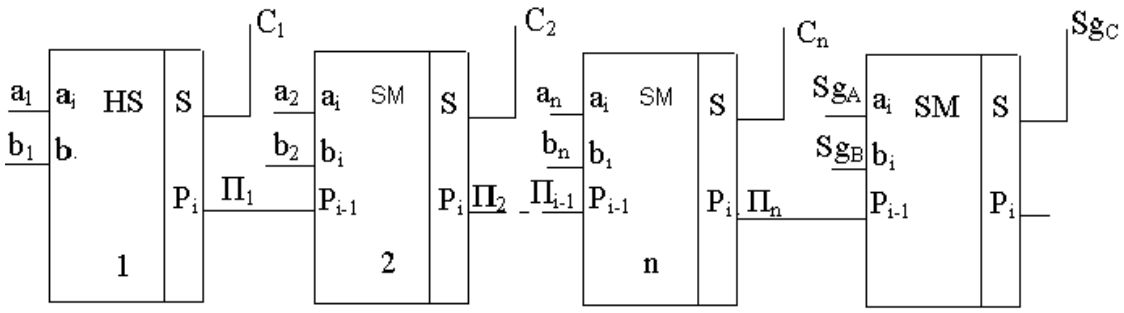


Рис.5.3

Теорема: ***Сумма дополнительных кодов чисел есть дополнительный код результата.***

Доказательство: Предполагая, что числа представлены в формате с фиксированной точкой, стоящей перед старшим разрядом, рассмотрим возможные случаи.

1) $A>0, B>0, \quad A+B<1.$

Так как $[A]_Д=A$, $[B]_Д=B$, то $[A]_Д+[B]_Д=A+B=[A+B]_Д$ – положительный результат.

2) $A<0, \quad B>0, \quad |A|>B.$

Так как в этом случае $[A]_Д=A+q, \quad [B]_Д=B$, то $[A]_Д+[B]_Д=A+B+q=[A+B]_Д$ – отрицательный результат (в знаковом разряде будет “1”).

3) $A<0, \quad B>0, \quad |A|<B.$

В этом случае имеем: $[A]_Д=A+q, \ [B]_Д=B$. И тогда $[A]_Д+[B]_Д=A+B+q$. Так как значение полученной в этом случае суммы больше q, то появляется единица переноса из знакового разряда, что равносильно изъятию из суммы q единиц. С учетом этого получаем результат: $[A]_Д+[B]_Д=A+B=[A+B]_Д$ – результат положительный, т.к. в знаковом разряде будет “0”.

4) $A<0, B<0, \quad |A+B|<1.$

Здесь $[A]_Д=A+q, \quad [B]_Д=B+q$. Тогда $[A]_Д+[B]_Д=A+B+q+q$. Появляется единица переноса из знакового разряда, что эквивалентно изъятию из суммы q единиц. В результате получается: $[A]_Д+[B]_Д=A+B+q=[A+B]_Д$ – отрицательный результат.

Таким образом, теорема справедлива для всех случаев, в которых не возникает переполнения разрядной сетки, что позволяет складывать машинные изображения чисел по правилам двоичной арифметики, не разделяя знаковую и цифровую части изображений.

Пример 5.2. Найти сумму чисел A=0.10110, B=0.00101, используя сумматор дополнительного кода

Решение. Складываются машинные изображения этих чисел

$$\begin{array}{rl} & [A]_Д = 0.10110 \\ + & [B]_Д = 0.00101 \\ \hline & [C]_Д = 0.11011 \end{array}$$

Ответ: C=0.11011

Пример 5.3. Найти сумму A=-0.10110, B=0.00101, используя сумматор дополнительного кода

Решение

$$\begin{array}{rl} & [A]_Д = 1.01010 \\ + & [B]_Д = 0.00101 \\ \hline & [C]_Д = 1.01111 \end{array}$$

Ответ: C=-0.10001.

Пример 5.4. Найти сумму чисел A=0.10110, B=-0.00101, используя сумматор дополнительного кода

Решение. Складывая машинные изображения чисел, представленных в дополнительном коде, получаем:

$$\begin{array}{rl} & [A]_Д = 0.10110 \\ + & [B]_Д = 1.11011 \\ \hline & [C]_Д = 0.10001 \end{array}$$

Ответ: *C*=0.10001.

Пример 5.5. Найти сумму чисел A=-0.10110, B=-0.00101, используя сумматор дополнительного кода

Решение.

$$\begin{array}{rl} & [A]_Д = 1.01010 \\ + & [B]_Д = 1.11011 \\ \hline & [C]_Д = 1.00101 \end{array}$$

Ответ: *C*=-0.11011.

5.2.3 Двоичный сумматор обратного кода (ДСОК)

ДСОК – сумматор, оперирующий изображением чисел в обратном коде. Характерная особенность ДСОК – наличие цепи кругового, или циклического, переноса из знакового разряда в младший разряд цифровой

части. Структурная схема ДСОК может быть представлена в виде, приведенном на рис. 5.4.

Теорема. ***Сумма обратных кодов есть обратный код результата.***

Доказательство. Аналогично предыдущему случаю рассмотрим основные случаи:

1) $A>0, \quad B>0, \quad A+B<1.$

Так как в этом случае $[A]_{об}=A, \quad [B]_{об}=B$, то $[A]_{об}+[B]_{об}=A+B=[A+B]_{об}$ – результат положительный.

2) $A<0, \quad B>0, \quad |A|>B.$

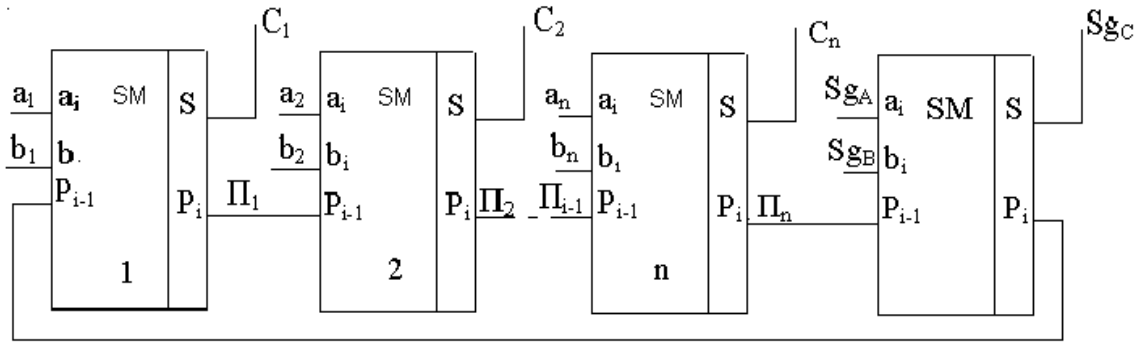


Рис. 5.4.

В этом случае имеем: $[A]_{об}=q-q^{-n}+A, \quad [B]_{об}=B$. И тогда

$[A]_{об}+[B]_{об}=q+q^{-n}+A+B=[A+B]_{об}$ – результат отрицательный.

3) $A<0, \; B>0, \quad |A|<B.$

В этом случае также имеем: $[A]_{об}=q-q^{-n}+A$, $[B]_{об}=B$, и

$[A]_{об}+[B]_{об}=q+q^{-n}+A+B.$

Но в данном случае сумма $[A]_{об}+[B]_{об}$ положительна, и правая часть выражения становится больше q, что вызывает появление единицы переноса из знакового разряда. Поскольку в ДСОК существует цепь переноса из знакового разряда в младший разряд (величина переноса из знакового разряда равна $q-q^{-n}$), то $[A]_{об}+[B]_{об}=[A+B]_{об}$ – результат положительный.

4) $A<0, \quad B<0, \quad |A+B|<1.$

Здесь имеем: $[A]_{об}=q-q^{-n}+A$, $[B]_{об}=q-q^{-n}+B$. Следовательно, сумма в этом случае будет иметь вид:

$$[A]_{об}+[B]_{об}=q-q^{-n}+A+q-q^{-n}+B=q-q^{-n}+q-q^{-n}+A+B.$$

Вполне очевидно, что здесь появляется единица переноса из знакового разряда, что эквивалентно изъятию из суммы величины $q-q^{-n}$. Тогда с учетом этого имеет место соотношение: $[A]_{об}+[B]_{об}=q-q^{-n}+A+B$ - результат отрицательный, т.е. $[A]_{об}+[B]_{об}=[A+B]_{об}$.

5) $|A|=B, \quad A<0, \quad B>0.$

Тогда в данном случае получаем: $[A]_{об}=q-q^{-n}+A$, $[B]_{об}=B$, и

$[A]_{об}+[B]_{об}=q-q^{-n}+A+B=q-q^{-n}$ – одно из изображений нуля в обратном коде.

Таким образом доказано, что на ДСОК машинные изображения чисел складываются по правилам, приведенным в таблице 5.2

Пример 5.6. Найти сумму двоичных чисел A=0.10110 и B=0.00101 используя сумматор обратного кода.

Решение. Складывая машинные изображения чисел, представленных в обратном коде, получаем:

$$\begin{array}{ll} & [A]_{об}=0.10110 \\ + & [B]_{об}=0.00101 \\ \hline & [C]_{об}=0.11011 \end{array}$$

Ответ: C=0.11011.

Пример 5.7. Найти сумму двоичных чисел A=-0.10110 и B=0.00101 используя сумматор обратного кода.

Решение.

$$\begin{array}{ll} & [A]_{об}=1.01001 \\ + & [B]_{об}=0.00101 \\ \hline & [C]_{об}=1.01110 \end{array}$$

Ответ: C=-0.10001.

Пример 5.8. Найти сумму чисел A=0.10110 и B=-0.00101 используя сумматор обратного кода.

Решение.

$$\begin{array}{ll} & [A]_{об}=0.10110 \\ + & [B]_{об}=1.11010 \\ \hline & [C]_{об}=0.10000 \\ & + \quad \text{- единица переноса} \\ & 1 \\ \hline & [C]_{об}=0.10001 \end{array}$$

Ответ: C=0.10001.

Пример 5.9. Найти сумму двоичных чисел A=-0.10110 и B=-0.00101 используя сумматор обратного кода.

Решение.

$$
\begin{array}{l}
[A]_{об} = 1.01001 \\
+\ [B]_{об} = 1.11010 \\
\hline
[C]_{об} = 1.00011 \\
\quad + \qquad\qquad \text{- единица переноса} \\
\quad 1 \\
\hline
[C]_{об} = 1.00100
\end{array}
$$

Ответ: C=-0.11011.

5.3 Переполнение разрядной сетки

При сложении чисел одинакового знака, представленных в форме с фиксированной точкой, может возникнуть переполнение разрядной сетки. Каждый сумматор обладает своим признаком переполнения разрядной сетки, анализируя который можно корректировать дальнейшие действия.

1. Признаком переполнения разрядной сетки ДСПК является появление единицы переноса из старшего разряда цифровой части числа.

Пример 5.10. Выполнить сложение на ДСПК:

1. A=0.10110, B=0.10001; 2. A=-0.10110, B=-0.10001.

$$
\begin{array}{l}
+\ [A]_{пр} = 0.10110 \\
\quad [B]_{пр} = 0.10001 \\
\hline
[C]_{пр} \neq 0.00111 \\
\quad \swarrow \\
\text{1- единица переноса}
\end{array}
\qquad
\begin{array}{l}
+\ [A]_{пр} = 1.10110 \\
\quad [B]_{пр} = 1.10001 \\
\hline
[C]_{пр} \neq 1.00111 \\
\quad \swarrow \\
\text{1- единица переноса}
\end{array}
$$

2. Признаком переполнения разрядной сетки ДСДК и разрядной сетки ДСОК является получение отрицательного результата при сложении положительных чисел, и получение положительного результата при сложении отрицательных чисел, т.е. при переполнении РС знак результата противоположен знакам операндов.

Пример 5.11 . Выполнить сложение двоичных чисел на ДСДК:

1. A=0.10110, B=0.10001; 2. A=-0.10110, B=-0.10001.

$$
\begin{array}{l}
+\ [A]_{д} = 0.10110 \\
\quad [B]_{д} = 0.10001 \\
\hline
[C]_{д} \neq 1.00111
\end{array}
\qquad
\begin{array}{l}
+\ [A]_{д} = 1.10110 \\
\quad [B]_{д} = 1.10001 \\
\hline
[C]_{д} \neq 0.11001
\end{array}
$$

Пример 5.12. Выполнить сложение двоичных чисел на ДСОК:

1. A=0.10110, B=0.10001; 2. A=-0.10110, B=-0.10001.

$$+\begin{array}{l}[A]_{об}=0.10110\\ [B]_{об}=0.10001\\ \hline [C]_{об}\neq 1.00111\end{array} \qquad +\begin{array}{l}[A]_{об}=1.10110\\ [B]_{об}=1.10001\\ \hline [C]_{об}\neq 0.10111\end{array}$$

Для обнаружения переполнения разрядной сетки в составе цифрового автомата должны быть предусмотрены аппаратные средства, автоматически вырабатывающие признак переполнения – сигнал φ.

Чтобы обнаружить переполнение разрядной сетки ДСДК и ДСОК, вводится вспомогательный разряд в знаковую часть изображения числа, который называется **разрядом переполнения**. Такой формат представления данных называется **модифицированным**. На рис.5.5 изображен модифицированный формат представления числа.

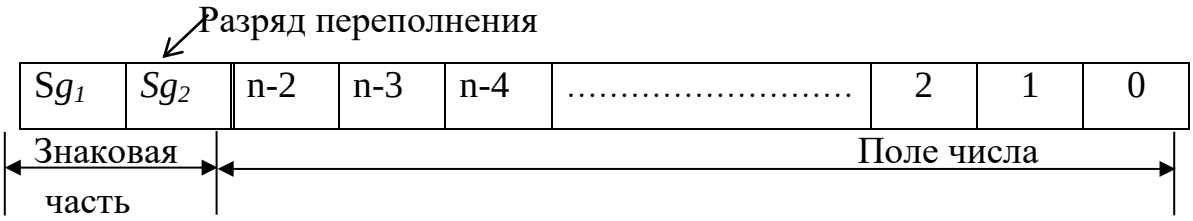


Рис. 5.5 Модифицированный формат представления данных

При представлении положительных чисел в модифицированном формате в знаковой части разрядной сетки помещаются нули (рис. 5.6 а), а при представлении отрицательных чисел – единицы (рис.5.6 б).

0	**0**	1	1	0	1	0	0	1	0	1	1

Рис. 5.6 а Представление положительного числа в модифицированном формате

1	**1**	1	1	0	1	0	0	1	0	1	1

Рис.5.6 б Представление отрицательного числа в модифицированном формате

Тогда в случае переполнения разрядной сетки при сложении чисел в знаковой части разрядной сетки будут находиться разные символы - | **0** | **1** | , или | **1** | **0** |. В этом случае функция φ=1, сигнализирующая о наличии переполнения разрядной сетки, будет равна единице, т.е.:

$$\varphi = \overline{Sg_1} \cdot Sg_2 \vee Sg_1 \cdot \overline{Sg_2} = (Sg_1 \oplus Sg_2) \qquad (5.1)$$

Пример 5.13. Выполнить сложение двоичных чисел на ДСОК, ДСПК, ДСДК, если:

1. A=0.10110, B=0.10001.

Решение

$[A]^{м}_{об} = 00.10110 \rightarrow$ модифицированное представление операнда A

$+\underline{[B]^{м}_{об} = 00.10001 \rightarrow \text{модифицированное представление операнда } B}$

$[C]^{м}_{об} = 01.00111$

$\rightarrow 01$ в знаковых разрядах – признак переполнения,

т.е. φ=1.

2. A=-0.10110, B=-0.10001.

Решение

$[A]^{м}_{об} = 11.01001 \rightarrow$ модифицированное представление операнда A

$+\underline{[B]^{м}_{об} = 11.01110 \rightarrow \text{модифицированное представление операнда } B}$

$[C]^{м}_{об} = 10.11000$

$\rightarrow 10$ в знаковых разрядах – признак переполнения,

т.е. φ=1.

3. A=0.10110, B=0.10001.

Решение

$[A]^{м}_{д} = 00.10110 \rightarrow$ модифицированное представление операнда A

$+\underline{[B]^{м}_{д} = 00.10001 \rightarrow \text{модифицированное представление операнда } B}$

$[C]^{м}_{д} = 01.00111$

$\rightarrow 01$ в знаковых разрядах – признак переполнения,

т.е. φ=1.

4. A=-0.10110, B=-0.10001.

Решение

$[A]^{м}_{д} = 11.01010 \rightarrow$ модифицированное представление операнда A

$+\underline{[B]^{м}_{д} = 11.01111 \rightarrow \text{модифицированное представление операнда } B}$

$[C]^{м}_{д} = 10.11001$

$\rightarrow 10$ в знаковых разрядах – признак переполнения,

т.е. φ=1.

5.4 Сложение чисел в формате с плавающей точкой

Как было уже сказано, числа, представленные в формате с плавающей точкой, изображаются двумя частями – мантиссой и порядком. При операции алгебраического сложения действия, выполняемые над мантиссами и порядками, различны. Следовательно, в цифровом устройстве должны быть два раздельных устройства для обработки мантисс и для обработки порядка.

Для чисел с плавающей точкой справедливо условие (4.11):

$$(q^{-1} \leq |m_A| < 1),$$

и всякий результат, не удовлетворяющий этому условию, должен быть приведен в соответствие с формулой (4.11). Операцию приведения результата (числа) к нормализованному виду называют операцией нормализации. Вполне очевидно, что операция нормализации числа состоит из проверки условия (4.11) и сдвига изображения мантиссы вправо или влево в пределах разрядной сетки.

Различают сдвиги: **логический, циклический, арифметический**.

Логический сдвиг – смещение всей числовой последовательности слова, включая знаковый разряд. При этом в освободившихся k разрядов записываются нули.

Пример 5.14 . Имеем числовую последовательность:

1	1	0	1	1	0	0	1	1	1

После логического сдвига на 3 разряда вправо получаем:

0	0	0	1	1	0	1	1	0	0

Циклический сдвиг – смещение всей числовой последовательности, при которой значения разрядов, выходящих за пределы разрядной сетки снова вводятся в освобождающиеся позиции слова.

Пример 5.15. Имеем числовую последовательность:

1	1	0	1	1	0	0	1	1	1

После циклического сдвига на 3 разряда вправо получаем:

1	1	1	1	1	0	1	1	0	0

Арифметический сдвиг – сдвиг всей числовой последовательности без изменения позиции знака числа. Различают **простой** и модифицированный **арифметический** сдвиги.

Простой арифметический сдвиг эквивалентен умножению числа, представленного в определенной системе счисления, на основание этой системы счисления, возведенное в степень, равную величине сдвига.

Простой арифметический сдвиг влево для разных кодов осуществляется по-разному.

Для прямого кода влево сдвигается только цифровая часть, младшие разряды заполняются нулями. Например, для исходного числа, представленного в прямом коде

1	0	0	0	1	0	1	1	0	1	1

После сдвига на 3 разряда влево получим:

1	1	0	1	1	0	1	1	0	0	0

Для дополнительного кода (ДК) и обратного кода (ОК) сдвигается влево вся числовая последовательность, освобождающиеся младшие разряды для ДК заполняются нулями, для ОК – единицами. Для ДК выдвигающаяся знаковая единица теряется. Например, дополнительный код числа имеет вид:

1	0	0	1	1	1	0	0	1	1	1

После сдвига на 1 разряд влево получаем:

0	0	1	1	1	0	0	1	1	1	0

Для обратного кода выдвигающаяся знаковая единица переносится в младший разряд цифровой части (как для циклического сдвига влево). Например, обратный код числа имеет вид:

1	0	0	1	1	1	0	0	1	1	1

После сдвига на один разряд влево в этом случае получаем:

0	0	1	1	1	0	0	1	1	1	1

Простой арифметический сдвиг вправо:

Для прямого кода вправо сдвигается только цифровая часть.

Например, для числа, представленного в прямом коде

1	0	0	1	1	1	0	0	1	1	1

после сдвига на один разряд вправо получаем:

1	0	0	0	1	1	1	0	0	1	1

Для ДК и ОК вправо сдвигается вся числовая последовательность. Цифра знакового разряда перемещается в старший цифровой разряд и в то же время восстанавливается в знаковом разряде. Например, в исходном состоянии имеем:

1	0	0	1	1	1	0	0	1	1	1

После сдвига на один разряд вправо получаем:

1	1	0	0	1	1	1	0	0	1	1

Модифицированный арифметический сдвиг используется для чисел, представленных в формате с плавающей точкой. Для такого сдвига величина исходного числа не меняется, т.е. в этом случае наряду со смещением мантиссы числа одновременно изменяется и его порядок так, чтобы величина числа осталась без изменений.

5.5 Операция нормализации

Невыполнение условия (4.11) ($q^{-1} \leq |m_A| < 1$) – есть нарушение нормализации числа. Приведение числа к нормализованной форме

представления – операция нормализации, которая заключается в последовательном выполнении модифицированного арифметического сдвига, для которого каждый сдвиг мантиссы числа на один разряд влево будет сопровождаться вычитанием единицы из порядка числа. Такой процесс продолжается до тех пор, пока число не примет нормализованный вид.

Так как условие (4.11) содержит два неравенства, то может быть нарушение как правого условия, так и левого (нарушение справа и слева).

Признак нарушения нормализации числа справа γ(т.е. $|m_A|\geq 1$) – наличие разных символов в знаковых разрядах сумматора.

$$\gamma = 1, \text{если } \overline{Sg_1}\&Sg_2 \vee Sg_1\&\overline{Sg_2} = 1 \qquad (5.2)$$

Соотношение γ=1 указывает на необходимость сдвига числа вправо на один разряд.

Признак нарушения нормализации числа слева δ (когда $|m_A|<q^{-1}$) – наличие одинаковых символов в разряде переполнения и в старшем разряде цифровой части сумматора (p_1).

$$\delta = 1, \text{если } Sg_2\&p_1 \vee \overline{Sg_2}\&\bar{p}_1 = 1. \qquad (5.3)$$

Соотношение δ=1 указывает на необходимость сдвига числа влево на один разряд.

Таким образом, операция нормализации числа состоит из совокупности сдвигов и проверки наличия признаков нарушения.

Рассмотрим сложение чисел, представленных в нормализованной форме: $C=A+B$, где $A=m_A \cdot p_A$, $B=m_B \cdot p_B$., где мантиссы m_A и m_B удовлетворяют условию (4.11).

1.Пусть $p_A=p_B$. Сложение мантисс осуществляется на соответствующем сумматоре по правилам, определяемым типом сумматора. Если после сложения, мантисса результата удовлетворяет условию нормализации, то к этому результату приписывается порядок любого из операндов. В противном случае необходимо выполнить нормализацию числа.

Пример 5.16. На ДСДК, включающим 6 разрядов мантиссы и 4 разряда для порядка, выполнить действие: $C=A+B$, если $A=0.1001\cdot 2^{-4}$ и $B=-0.1101\cdot 2^{-4}$.

Решение: $[m_A]^{\text{м}}_{\text{д}} = 00.1001;\ [p_A]_{\text{д}} = 1.100;$

$[m_B]^{\text{м}}_{\text{д}} = 11.0011;\ [p_B]_{\text{д}} = 1.100.$

Складывая мантиссы, получаем:

$$\begin{array}{r} [m_A]^{\text{м}}_{\text{д}} = 00.1001 \\ +\ [m_B]^{\text{м}}_{\text{д}} = 11.0011 \\ \hline [m_C]^{\text{м}}_{\text{д}} = 11.1100 \end{array}$$

Здесь получаем, что

$$\delta = Sg_2 \& p_1 \vee \overline{Sg}_2 \& \bar{p}_1 = 1 \cdot 1 \vee 1 \cdot 0 = 1 \vee 0 = 1.$$

$$\gamma = \overline{Sg}_1 \& Sg_2 \vee Sg_1 \& \overline{Sg}_2 = 0 \cdot 1 \vee 1 \cdot 0 = 0 \vee 0 = 0.$$

Т.е. имеется нарушение нормализации слева и, следовательно, необходимо осуществить модифицированный сдвиг влево. После сдвига на один разряд получаем:

$[m_C^*]_{\text{д}}^{\text{м}}$=11.1000. Т.е. здесь опять имеем δ=1

Одновременно со сдвигом осуществляется коррекция порядка, т.е. уменьшение порядка в данном случае на единицу ($[p_C^*]_{\text{д}} = [p_C]_{\text{д}} - 1$), т.е.

$$\begin{array}{r} [p_C]_{\text{д}} = 1.100 \\ +\underline{[-1]_{\text{д}} = 1.111} \\ [p_C^*]_{\text{д}} = 1.011 \end{array}$$

Т.к. после первого сдвига снова имеет место нарушение нормализации, то снова нужна коррекция результата, т.е. еще раз осуществляется сдвиг мантиссы и коррекция порядка:

$[m_C^{**}]_{\text{д}}^{\text{м}}$=11.0000. (Теперь получаем, что γ=0, δ=0).

$$\begin{array}{r} [p_C^*]_{\text{д}} = 1.011 \\ +\underline{[-1]_{\text{д}} = 1.111} \\ [p_C^{**}]_{\text{д}} = 1.010 \end{array}$$

Теперь можно записать результат: $[m_C]_{\text{д}}^{\text{м}} = [m_C^*]_{\text{д}}^{\text{м}} = 11.0000$; $[p_C^{**}]_{\text{д}} = 1.010$. C=-1.0000·10^{-110}=-2^{-6}_{10}.

Пример 5.17. На ДСОК, включающим 6 разрядов мантиссы и 4 разряда для порядка, выполнить действие: C=A+B, если A=0.1001·2^{-4} и B=-0.1101·2^{-4}. (т.е. выполним сложение чисел из предыдущего примера на ДСОК).

Решение: $[m_A]_{\text{об}}^{\text{м}} = 00.1001$; $[p_A]_{\text{об}} = 1.011$;

$[m_B]_{\text{об}}^{\text{м}} = 11.0010$; $[p_B]_{\text{об}} = 1.011$.

Складывая мантиссы, получаем:

$$\begin{array}{r} [m_A]_{\text{д}}^{\text{м}} = 00.1001 \\ +\underline{[m_B]_{\text{д}}^{\text{м}} = 11.0010} \\ [m_C]_{\text{д}}^{\text{м}} = 11.1011 \end{array}$$

Здесь получаем, что

$$\delta = Sg_2 \& p_1 \vee \overline{Sg}_2 \& \bar{p}_1 = 1 \cdot 1 \vee 0 \cdot 0 = 1 \vee 0 = 1.$$

$$\gamma = \overline{Sg}_1 \& Sg_2 \vee Sg_1 \& \overline{Sg}_2 = 0 \cdot 1 \vee 1 \cdot 0 = 0 \vee 0 = 0.$$

Следовательно, необходим сдвиг влево. После сдвига получаем: $[m_C^*]_{\text{д}}^{\text{м}}$=11.0111, и

$$[p_C]_{об} = 1.011$$
$$+\underline{[-1]_{об} = 1.110}$$
$$[p_C^*]_{об} = 1.010$$

Теперь нарушения нормализации числа нет, и конечный результат будет иметь вид:

$[m_C]^{м}_{об}=[m_C^*]^{м}_{об} = 11.0111;$ $[p_C^*]_{об} = 1.010$. $C=-0.1\cdot10^{-101}=-2^{-6}{}_{10}$.

2. Пусть теперь $p_A \neq p_B$. Для операции сложения чисел необходимым условием является соответствие разрядов операндов друг другу. Следовательно, нужно уровнять порядки, что повлечет за собой, естественно, временное нарушение нормализации одного из операндов. Выравнивание порядков означает, что порядок меньшего числа надо увеличить на величину $\Delta p=|p_A-p_B|$, что в свою очередь означает сдвиг мантиссы меньшего числа вправо на количество разрядов, равное Δp. Если $p_A-p_B>0$, то $p_A>p_B$ и сдвигается на Δp разрядов вправо мантисса числа B; если же $p_A-p_B<0$, то $p_A<p_B$ и сдвигается вправо на Δp разрядов мантисса числа A. Далее выполняется операция сложения чисел по вышеописанным правилам.

Операция сложения и вычитания чисел в форме с плавающей запятой осуществляется во всех современных машинах по изложенным выше правилам.

Пример 5.18. На 10-разрядном ДСОК, включающим 6 разрядов мантиссы и 4 разряда для порядка, выполнить действие:

$C=A+B$, если $A=0.1001\cdot2^{-2}$ и $B=-0.1011\cdot2^{-3}$.

Решение:

$[m_A]^{м}_{об} =$ 00.1001; $[p_A]_{об} = 1.101;$

$[m_B]^{м}_{об} = 11.0100;$ $[p_B]_{об} = 1.100.$

$[\Delta p]_{об} = [p_A]_{об} - [p_B]_{об} = [p_A]_{об}(-[p_B]_{об}) = 1.101 + 0.011 = 0.001.$

Т.к. $\Delta p > 0$, следовательно, $p_A>p_B$, и нужно сдвинуть мантиссу числа B на один разряд. После сдвига получаем: $[m_B]^{м}_{об} = 11.1010.$ Складывая мантиссы, получаем:

Складывая мантиссы, получаем:

$$[m_A]^{м}_{об} = 00.1001$$
$$+\underline{[m_B]^{м}_{об} = 11.1010}$$
$$[m_C]^{м}_{об} = 00.0011$$
$$+ 1 \text{ – единица переноса из знакового разряда}$$
$$[m_C]^{м}_{об} = 00.0100$$

Здесь необходимо выполнить операцию нормализации результат, т.к. имеет место нарушение нормализации слева:

$$\delta = Sg_2 \& p_1 \vee \overline{Sg}_2 \& \bar{p}_1 = 0 \cdot 0 \vee 1 \cdot 1 = 0 \vee 1 = 1.$$

$$\gamma = \overline{Sg}_1 \& Sg_2 \vee Sg_1 \& \overline{Sg}_2 = 1 \cdot 0 \vee 0 \cdot 1 = 0 \vee 0 = 0.$$

Следовательно, необходим сдвиг влево. После сдвига получаем: $[m_C^*]_{\text{об}}^{\text{м}}$=00.1000, и

$$\begin{array}{r} [p_C]_{\text{об}} = 1.101 \\ +\underline{[-1]_{\text{об}} = 1.110} \\ [p_C^*]_{\text{об}} = 1.100 \end{array}$$

Теперь δ=0, γ=0, и можно записать результат:

$[m_C]_{\text{об}}^{\text{м}} = [m_C^*]_{\text{об}}^{\text{м}}$=00.1000; $[p_C]_{\text{об}} = [p_C^*]_{\text{об}}$=1.100;

m_C =0.1000; p_C=-011; C=$0.1 \cdot 10^{-11_2}$.

5.6 Оценка точности выполнения арифметических операций

Выбор системы счисления и длины разрядной сетки, а также формы представления числа в машине тесно связаны с обеспечением заданной точности вычислений. Важное значение имеет оценка точности арифметических вычислений в форме с фиксированной и плавающей точкой. При операциях сложения и вычитания чисел, представленных в формате с фиксированной точкой, можно считать, что они выполняются точно.

Для чисел, представленных в форме с плавающей точкой, при операциях сложения и вычитания необходимо выравнивать порядки, что ведет к потере значений некоторых разрядов мантиссы при сдвиге. Поэтому, при нормализованной форме представления чисел сама операция алгебраического сложения чисел также является источником погрешностей.

Таким образом, причинами погрешностей вычислений в ЭВМ являются:

- неточное задание исходных данных;
- погрешности перевода (представления) чисел;
- погрешности, вызванные ограничением ДРС;
- использование приближенных методов вычислений;
- округление результатов элементарных операций;
- сбои в работе ЭВМ.

5.6.1 Погрешности выполнения арифметических операций

Произведем действия над числами *A* и *B*, заданными с абсолютными погрешностями *ΔA* и *ΔB* соответственно: *A*=[*A*]+ *ΔA*, *B*=[*B*]+ *ΔB*.

При выполнении операции сложения имеем:

$$A+B=[A]+\Delta A+[B]+\Delta B=[A]+[B]+(\Delta A+\Delta B)=[A]+[B]+\Delta(A+B), \quad (5.4)$$

где $\Delta(A+B)= \Delta A+\Delta B$ – абсолютная погрешность суммы.

При выполнении операции вычитания:

$$A-B=[A]+\Delta A-[B]-\Delta B=[A]-[B]+(\Delta A-\Delta B)=[A]-[B]+\Delta(A-B), \quad (5.5)$$

где $\Delta(A-B)= \Delta A-\Delta B$ – абсолютная погрешность разности.

При выполнении операции умножения;

$$A\cdot B=([A]+\Delta A)\cdot([B]+\Delta B)=[A]\cdot[B]+B\cdot\Delta A+[A]\cdot\Delta B+\Delta A\cdot\Delta B. \quad (5.6)$$

Так как $\Delta A\cdot\Delta B$ на два порядка меньше чисел A и B, то этим произведением можно пренебречь, и, следовательно,

$$A\cdot B\approx[A]\cdot[B]+B\cdot\Delta A+[A]\cdot\Delta B=[A]\cdot[B]+\Delta(A\cdot B), \quad (5.7)$$

где $\Delta(A\cdot B)$- абсолютная погрешность произведения.

При выполнении операции деления:

$$\frac{A}{B}=\frac{[A]+\Delta A}{[B]+\Delta\ B}=\frac{[A]+\Delta A}{[B]}\cdot\left(\frac{1}{1+{}^{\Delta B}/_{B}}\right). \quad (5.8)$$

Второй сомножитель разложим в ряд и после преобразований получим:

$$\frac{A}{B}=\frac{[A]}{[B]}-\frac{[A]\cdot\Delta B}{[B]^2}+\frac{[A]\cdot\Delta B^2}{[B]^3}+\frac{\Delta A}{[B]}-\frac{\Delta A\cdot\Delta B}{[B]^2}+\cdots \quad (5.9)$$

Пренебрегая членами второго порядка малости, получаем:

$$\frac{A}{B}=\frac{[A]}{[B]}+\frac{\Delta A}{[B]}-\frac{\Delta A\cdot\Delta B}{[B]^2}=\frac{[A]}{[B]}+\Delta\left(\frac{A}{B}\right), \quad (5.10)$$

где $\Delta\left(\frac{A}{B}\right)$ - абсолютная погрешность частного.

Аналогично можно вывести выражения для относительных погрешностей:

$$\delta_{\pm}=\frac{[A]}{[A]+[B]}\cdot\frac{\Delta A}{[A]}\pm\frac{[B]}{[A]+[B]}\cdot\frac{\Delta B}{[B]} \quad (5.11)$$

$$\delta_{A\cdot B}=\frac{\Delta A}{[A]}+\frac{\Delta B}{[B]}; \quad (5.12)$$

$$\delta_{\frac{A}{B}}=\frac{\Delta A}{[A]}-\frac{\Delta B}{[B]} \quad (5.13)$$

5.6.2 Погрешности округления

Если предположить, что исходная информация не содержит никаких ошибок и все вычислительные процессы конечны и не приводят к ошибкам, то все равно присутствует третий тип ошибок – ошибки округления.

Пример5.19 . Имеем гипотетическую машину с пятиразрядной сеткой. Необходимо сложить 9.2654 и 7.1625. Примем, что эти числа точные. Но их сумма, равная 16.4279, содержит шесть разрядов. Поэтому шестиразрядный результат будет округлен до значения 16.428.

Погрешность округления имеет смысл только для действительных чисел, т.к. ЭВМ автоматически выравнивает порядки действительных чисел при сложении и вычитании.

Для чисел, представленных в форме с плавающей точкой, справедливы выражения: $A_q = m_A \cdot q^k$ и $q^{-1} \le |m_A| < 1$.

Если для представления мантиссы используется только *n* разрядов, то изображение числа разбивается на две части:

$$A_q = [m_A] \cdot q^n + [A_0] \cdot q^{r-n}, \qquad (5.14)$$

где $[A_0] \cdot q^{r-n} = A_0$ - ‘хвост’ числа, не попавший в разрядную сетку.

В зависимости от того, как учитывается величина A_0 в машинном изображении, существует несколько способов округления.

1. **Отбрасывание A_0.** При этом возникает относительная погрешность:

$$\delta_{\text{окр}} = \frac{|A_0| \cdot q^{k-n}}{|m_A| \cdot q^k}. \qquad (5.15)$$

Т.к. $q^{-1} \le |m_A| < 1$ и $0 \le |A_0| < 1$, то

$$\delta_{\text{окр}} = \frac{1 \cdot q^{k-n}}{q^{-1} \cdot q^k} = q^{-(n-1)}, \qquad (5.16)$$

т.е. не зависит от величины самого числа, а зависит только от количества разрядов в машине для любой системы счисления.

2. **Симметричное округление**. При этом производится анализ величины A_0. Принимается, что

$$[A] = \begin{cases} [m_A] \cdot q^n, & \text{если} \quad |A_0| < q^{-1}, \\ [m_A] \cdot q^n + q^{k-n}, & \text{если} \quad |A_0| \ge q^{-1}. \end{cases}$$

(5.17)

При условии $|A_0| \ge q^{-1}$ производится прибавление единицы к младшему разряду мантиссы. Абсолютная погрешность округления при этом будет равна:

$$\Delta_{\text{окр}} = \begin{cases} |A_0| \cdot q^{k-n}, \\ |1 - A_0| \cdot q^{k-n}. \end{cases} \qquad (5.18)$$

Максимально возможное значение модуля абсолютной погрешности равно $0.5q^{k-n}$. Способ симметричного округления наиболее часто используется на практике.

3. **Округление по дополнению**. В этом случае для округления берется информация, содержащая в $(n+1)$-м разряде. В случае двоичной системы счисления, если в $(n+1)$-м разряде содержится единица, в n разряд добавляется единица; если в $(n+1)$-м разряде содержится нуль, содержимое разрядов правее n-го отбрасывается.

4. **Случайное округление**. Для такого округления необходимо иметь датчик случайных величин (1 или 0), который выдает единицу в самый младший разряд машинного изображения числа.