**Глава 1. Числовая система компьютера**

**1.1 Понятие кодирования**

Кодирование представляет собой процесс преобразования информации из исходного формата, удобного для восприятия человеком, в формат, который более подходит для хранения, передачи и обработки компьютером. Обратный процесс называется декодированием.

Основные цели кодирования включают:

1. Удобство технической реализации;
2. Удобство восприятия информации;
3. Высокая скорость передачи и обработки информационных сообщений;
4. Экономичность, достигаемая за счёт уменьшения избыточности в сообщениях;
5. Надежность, обеспечивающая защиту информации от случайных искажений;
6. Сохранность информации и защита от несанкционированного доступа.

На различных этапах работы с информацией могут возникать разные цели, что приводит к необходимости многократного перекодирования. При этом следует учитывать, что указанные цели могут быть взаимно противоречивыми.

**1.2 Хранение чисел в памяти компьютера**

Хранение чисел в электронных вычислительных машинах (ЭВМ) осуществляется с использованием цифрового двоичного кода. Это связано с тем, что электронные элементы компьютера могут находиться только в двух устойчивых состояниях — 0 или 1. Основной единицей информации является бит, а 8 бит формируют 1 байт памяти. Каждый байт имеет свой порядковый номер, который называется адресом.

Элементы памяти объединяются в ячейки, которые называются машинными словами [34]. Адрес машинного слова определяется адресом его младшего байта. Объем информации, который может быть представлен в компьютере, ограничен ёмкостью его памяти, что влияет на точность представления числовой информации, зависящей от архитектуры системы.

Например, если компьютер имеет оперативную память объемом 2 килобайта, то это соответствует 2048 байтам. Важно отметить, что в компьютерной терминологии 1 килобайт (КБ) равен 1024 байтам, так как используется двоичная система счисления. Таким образом, 2 КБ = 2 × 1024 = 2048 байтов. Поскольку нумерация начинается с нуля, адрес последнего байта будет равен 2047.

Для закрепления материала приведено решение задач:

Задача:

В оперативной памяти компьютера содержится 163840 машинных слов, что эквивалентно 0.625 мегабайтам. Необходимо определить длину машинного слова в байтах.

Решение:

* + - 1. Необходимо перевести мегабайты в байты:

0.625 МБ = 0.625 × 1024 КБ = 640 КБ

640 КБ = 640 × 1024 Б = 655360 Б

* + - 1. Для вычисления длины машинного слова необходимо выполнить следующее действие:

Длина машинного слова = Общее количество байтов/Количество машинных слов. Длина машинного слова = 655360 Б/163840 слов = 4 Б

Ответ: Длина машинного слова составляет 4 байта.

Задача:

Объем оперативной памяти равен 1 мегабайту, а адрес последнего машинного слова составляет 1048574. Необходимо определить размер машинного слова.

Решение:

Перевод объема памяти в байты:

Объем оперативной памяти равен 1 мегабайту. Переведем это значение в байты:

1 МБ = 1024 КБ = 1024 × 1024 Б = 1048576 Б

Определение адреса последнего байта:

Учитывая, что нумерация байтов начинается с 0, адрес последнего байта будет на единицу меньше общего количества байтов:

Адрес последнего байта = 1048576 - 1 = 1048575

Анализ адреса последнего машинного слова:

Адрес последнего машинного слова равен 1048574, что подразумевает, что последнее машинное слово занимает адреса 1048574 и 1048575.

Определение размера машинного слова:

Чтобы выяснить, сколько байтов занимает последнее машинное слово, рассмотрим два адреса: последнее машинное слово охватывает адреса 1048574 и 1048575. Следовательно, размер машинного слова составляет 2 байта.

Ответ: Размер машинного слова составляет 2 байта.

Задача:

Определите адрес последнего байта оперативной памяти объемом 4 килобайта (КБ).

Решение:

1. Перевод объема оперативной памяти в байты:

Объем оперативной памяти равен 4 килобайтам:

4 КБ = 4 × 1024 Б = 4096 Б

1. Определение адреса последнего байта:

Поскольку адресация начинается с нуля, адрес последнего байта можно определить следующим образом:

Адрес последнего байта = Общий объем памяти – 1 = 4096 – 1 = 4095

Ответ: Адрес последнего байта оперативной памяти объемом 4 килобайта составляет 4095.

**1.3 Форматы данных компьютера**

Форматы данных в компьютерах играют ключевую роль в обработке и хранении информации. Основные форматы данных представлены логическими кодами, числами с фиксированной точкой и числами с плавающей точкой. Логические коды представляют собой символьные величины, числа без знака и битовые величины.

Для обмена символьной информацией широко используется американский стандартный код ASCII. Этот код генерируется внешними устройствами, такими как принтеры, и служит для передачи данных между внешними устройствами и оперативной памятью компьютера [24]. Каждый символ в коде ASCII занимает 1 байт памяти, при этом седьмой бит всегда равен 0. Поскольку 1 байт состоит из 8 бит, таблица ASCII включает 28 символов, всего 256.

В таблице ASCII стандартными являются первые 128 символов, которые нумеруются от 0 до 127. В этом диапазоне хранятся буквы латинского алфавита, цифры, знаки препинания и служебные символы. Символы от 128 до 255 используются для представления символов национальных алфавитов, научных символов и псевдографики.

Пример: слово из 14 символов занимает 14 байт памяти, что соответствует 112 битам (14 байт × 8 бит).

Для закрепления материала приведено решение задач:

Задача:

Символ 'i' в ASCII имеет код 105₁₀. Определите слово, зашифрованное в последовательности 108 105 110 107₁₀.

Решение:

Нумерация символов в таблице ASCII идет по алфавиту. Необходимо соотнести символы с их кодами и получить слово "link".

Ответ: link.

Задача:

Слово "stop" в двоичной системе счисления необходимо представить в шестнадцатеричной системе.

Решение:

Чтобы представить слово "stop" в шестнадцатеричной системе счисления, сначала необходимо перевести каждую букву в двоичный код, а затем в шестнадцатеричный.

1. Перевод букв в ASCII:

's' = 115

't' = 116

'o' = 111

'p' = 112

1. Перевод ASCII в двоичный код:

's' = 01110011

't' = 01110100

'o' = 01101111

'p' = 01110000

1. Объединение двоичного кода:

01110011 01110100 01101111 01110000

1. Перевод двоичного кода в шестнадцатеричный:

Для перевода в данную систему счисления двоичный код разбивается на группы по 4 бита, начиная с правого края.

0010 0011 0111 0100 0110 1111 0111 0000

1. Перевод каждой группы в шестнадцатеричное значение:

0010 = 2

0011 = 3

0111 = 7

0100 = 4

0110 = 6

1111 = F

0111 = 7

0000 = 0

1. Объединение шестнадцатеричных значений: 23746F70

Ответ: 23746F70.

Таким образом, понимание форматов данных и их кодирования является важным аспектом работы с компьютерной информацией.

**1.4 Формат чисел с фиксированной точкой в компьютерах**

Формат чисел с фиксированной точкой используется для представления целых чисел в компьютерах. Диапазон целых чисел, который может быть представлен, зависит от размера ячеек памяти, известных как машинные слова. Если для хранения целого числа используется k-разрядное машинное слово, то оно содержит k бит памяти. Таким образом, в k-разрядном машинном слове возможно хранение 2k различных целых чисел.

Рассмотрим 16-разрядное машинное слово. В таком случае можно хранить 216 различных целых чисел, что составляет 65536 значений.

Положительные числа: Если хранятся только положительные числа, диапазон возможных значений будет от 0 до 65535 (от 0 до 2k - 1).

Положительные и отрицательные числа: Если в памяти хранятся как положительные, так и отрицательные числа в равном количестве, диапазон значений составит от -32768 до 32767 (от -2k-1 до 2k-1 - 1).

Таким образом, формат чисел с фиксированной точкой позволяет эффективно представлять целые числа, обеспечивая разнообразие диапазонов в зависимости от используемого количества бит.

**1.5 Алгоритмы представления целых чисел в компьютерах**

В современных вычислительных системах целые числа могут представляться в различных форматах, в том числе в двоичной и шестнадцатеричной системах счисления. Рассмотрим алгоритмы представления как положительных, так и отрицательных целых чисел в двухбайтовом машинном слове.

**Алгоритм представления целого положительного числа:**

1. Определение разрядности: Пусть число хранится в k-разрядном машинном слове.
2. Перевод в двоичную систему: Заданное целое положительное число N переводится в двоичную систему счисления.
3. Дополнение нулями: Полученный двоичный результат дополняется слева незначащими нулями до k-разрядов, что представляет число в компьютере.
4. Перевод в шестнадцатеричную систему: Двоичное число переводится в шестнадцатеричную систему, так как часто данные хранятся в упакованном виде именно в этом формате.

Например, для числа 160710, которое хранится в двухбайтовом машинном слове, алгоритм будет следующим: число переводится в двоичную систему, дополняется нулями до нужного количества разрядов и затем конвертируется в шестнадцатеричную систему (рисунок 1.1).

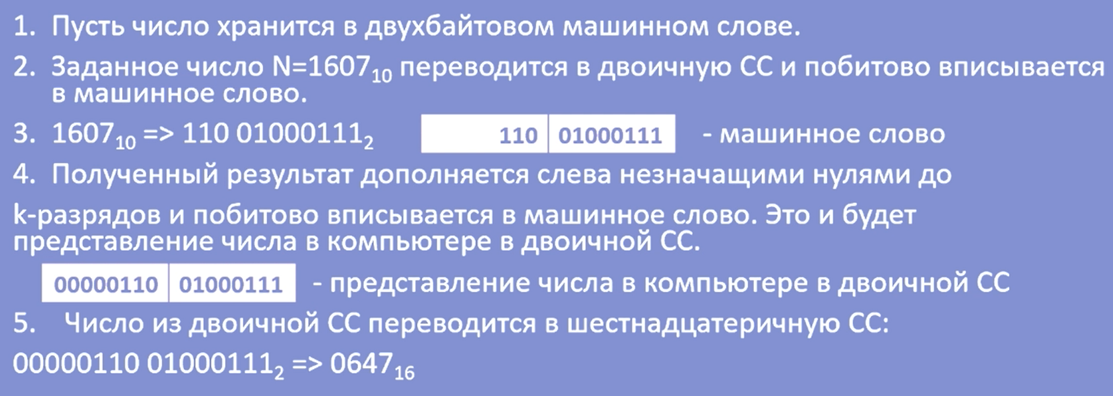


Рисунок 1.1 – Алгоритм представления целого положительного числа 160710 в формате с фиксированной точкой

**Алгоритм представления целого отрицательного числа:**

1. Получение внутреннего представления: Модуль заданного отрицательного числа N переводится в двоичную систему счисления.
2. Получение обратного кода: Обратный код получается заменой всех 0 на 1 и наоборот.
3. Получение дополнительного кода: К числу, полученному на предыдущем шаге, добавляется единица.
4. Перевод в шестнадцатеричную систему: Число из двоичной системы переводится в шестнадцатеричную, чтобы обеспечить его хранение в упакованном виде.

При рассмотрении алгоритма представления отрицательного числа -160710 в двухбайтовом машинном слове, компьютер проводит следующие шаги (рисунок 1.2):

* + - 1. Система вычисляет модуль отрицательного числа -1607, исключая отрицательный знак и получая положительное число 1607.
      2. Полученное положительное число преобразуется из десятичной системы в двоичную. Путем деления на 2 и сохранения остатков, число 1607 переводится в двоичную систему как 11001000111.
      3. Учитывая ограничения двухбайтового машинного слова, число дополняется нулями и принимает форму 0000011001000111.
      4. Затем происходит инвертирование битов этого числа для получения обратного кода. Преобразование в обратный код превращает число 0000011001000111 в 1111100110111000.
      5. Следующим этапом является получение дополнительного кода, который достигается путем добавления единицы к обратному коду. При добавлении 1 к обратному коду 1111100110111000 получается дополнительный код 1111100110111001.
      6. Для удобства представления, вычисленный двоичный результат 1111100110111001 конвертируется в шестнадцатеричную систему, где он разбивается на группы по четыре бита (1111 1001 1011 1001) и заменяется соответствующими шестнадцатеричными символами. В итоге представленное в памяти значение числа -1607 будет иметь вид F9B9 в шестнадцатеричной системе.

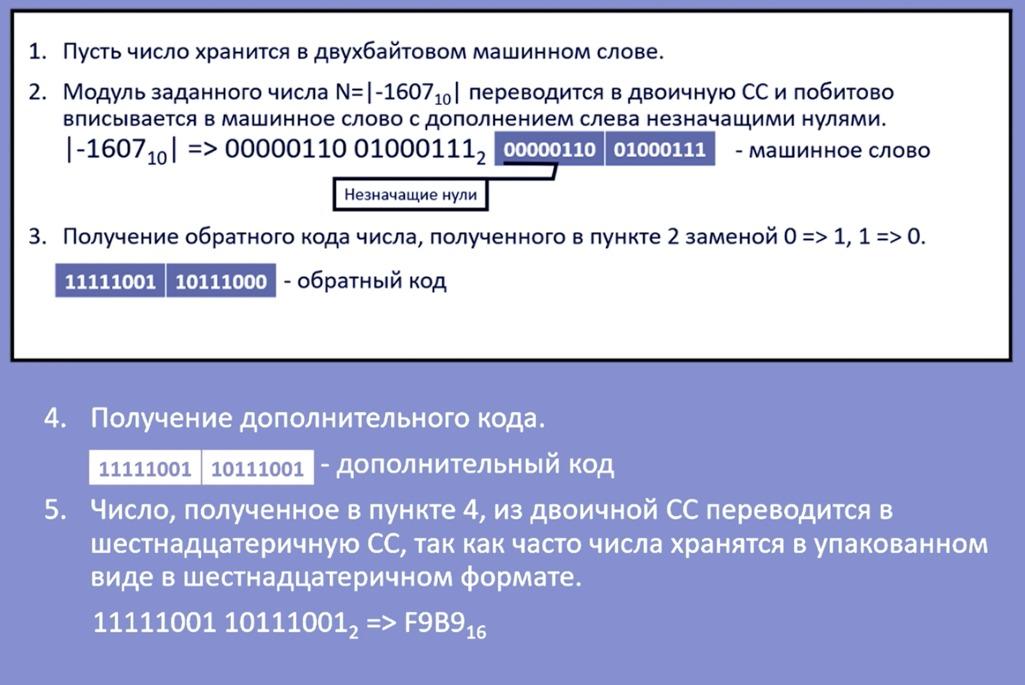


Рисунок 1.2 – Алгоритм представления целого отрицательного числа 160710 в формате с фиксированной точкой

**1.6 Алгоритм представления вещественного числа в компьютере**

В математике представление вещественных чисел неоднозначно, поэтому используется искусственная форма представления вещественного числа в компьютере в нормализованном виде. Оно основано на формуле:*k = m + np*, где k – вещественное число; m – мантисса числа; n – основание системы счисления; p – математический порядок числа.

Например, число 25.3240 можно представить в виде нескольких выражений:

25.3240 = 2.5324 × 10 = 0.25324 × 102 = 0.0025324 × 104 = 2532.4 × 10-2

1. Нормализованное представление вещественных чисел обеспечивает компактное использование памяти, сохраняя лишь самые важные элементы: мантиссу, порядок числа, знак числа и знак порядка действительного числа. В этом представлении не хранится целая часть числа, а также не нужно хранить основание системы счисления. Знак числа указывает на его положительность или отрицательность, в то время как знак порядка определяет положительный или отрицательный порядок имеет исходное число.

Примеры нормализованных записей дробных чисел в двоичной системе счисления:

101.11 = 0.10111 × 1011

0.01 = 0.1 × 10-10

1. Чем длиннее машинное слово, тем более точно можно представить вещественное число в памяти компьютера. При хранении в памяти компьютера мантисса представляется как целое число, включающее только значимые цифры перед и после запятой до первой значащей цифры мантиссы. Следовательно, внутреннее представление вещественного числа сводится к хранению двух целых чисел: мантиссы и порядка. Разные типы компьютеров используют разные способы представления чисел в формате с плавающей точкой. Обычно число в формате с плавающей точкой занимает в памяти либо четыре байта (для чисел стандартной точности), либо восемь байт (для чисел с двойной точностью).

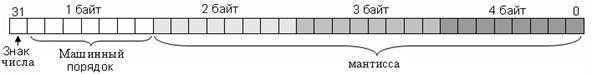


Рисунок 1.3 – Хранение вещественного числа в памяти компьютера

Из рисунка 1.3 видно, что в старшем бите первого байта хранится знак числа: 0 (плюс) или 1 (минус). Следующие 7 битов первого байта содержат машинный порядок. Значащие цифры мантиссы (24 разряда) хранятся в последующих трех байтах.

В 7-ми разрядах машинного порядка помещаются двоичные числа в интервале от 00000002 (010) до 11111112 (12710), составляющие 128 различных значений. Учитывая возможность наличия как положительного, так и отрицательного порядка, эти 128 значений равномерно распределяются между положительными и отрицательными значениями порядка, охватывая диапазон от -64 до 63 – математический порядок (рисунок 1.4).

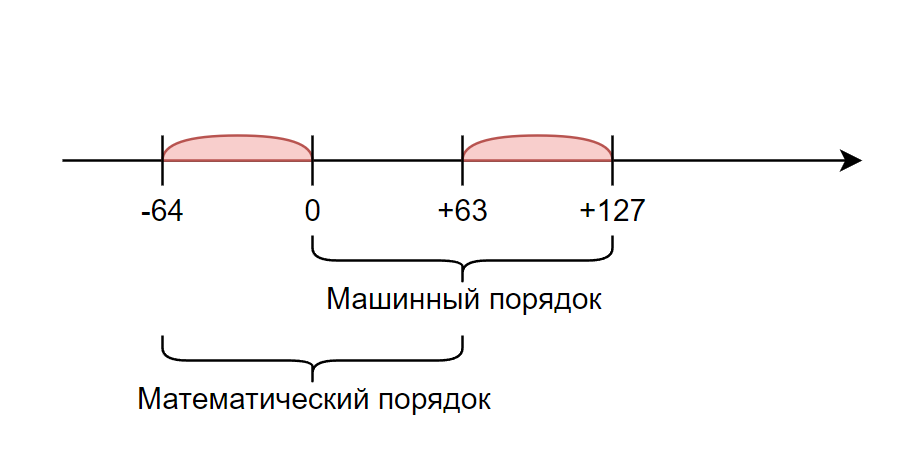


Рисунок 1.4 – Смещение машинного порядка относительно математического

Машинный порядок смещен относительно математического и содержит только положительные значения. Смещение выбирается таким образом, чтобы минимальному математическому значению порядка соответствовал нулевой машинный порядок.

Формулы для пересчета порядков: *Mp = p + 64; Mp2 = p2 + 10000002.*

**Алгоритм представления вещественного числа в памяти компьютера:**

1. Модуль числа переводится в двоичную систему с 24 значащими цифрами мантиссы. Если значащие цифры занимают меньше разрядов, то дополняются нулями справа до конца машинного слова.
2. Двоичное число нормализуется по формуле *k = m + np*.

Рассчитывается машинный порядок числа в двоичной системе с учетом формулы *Mp2 = p2 + 10000002*.

1. Знак числа (0 для положительного числа и 1 для отрицательного) вместе с машинным порядком вписываются в 4-байтовое машинное слово, образуя единое целое с мантиссой (рисунок 1.3).
2. Полученное представление вещественного числа выводится в шестнадцатеричной системе счисления.

Пример представления вещественного числа 250.187510 в четырехбайтовом машинном слове, компьютер выполняет следующие шаги:

1. Перевод числа в двоичную систему счисления с 24 значащими цифрами: 11111010.00110000000000002.
2. Запись в форме нормализованного двоичного числа с плавающей точкой: 0.11111010001100000000 0000 × 101000.

Вычисление машинного порядка в двоичной системе счисления по формуле *Mp2 = p2 + 10000002*: Мр2 = 10002 + 10000002 = 10010002.

1. Запись числа в 4-байтовой ячейке памяти с учетом знака:

Описание: https://konspekta.net/studopediaru/baza18/386191637022.files/image130.jpg

Рисунок 1.5 – Представление вещественного числа 250.187510 в четырехбайтовом машинном слове

1. Шестнадцатеричная форма: 48FA3000

Таким образом, компьютер успешно представляет вещественное число 250.187510 в четырехбайтовом машинном слове, используя указанный алгоритм, и выводит результат в формате шестнадцатеричной системы счисления.

В рамках практического задания необходимо получить внутреннее компьютерное представление следующих чисел в двухбайтовом машинном слове:

+19510

-19510

+26310

-26310

+345.06510

-164.07810