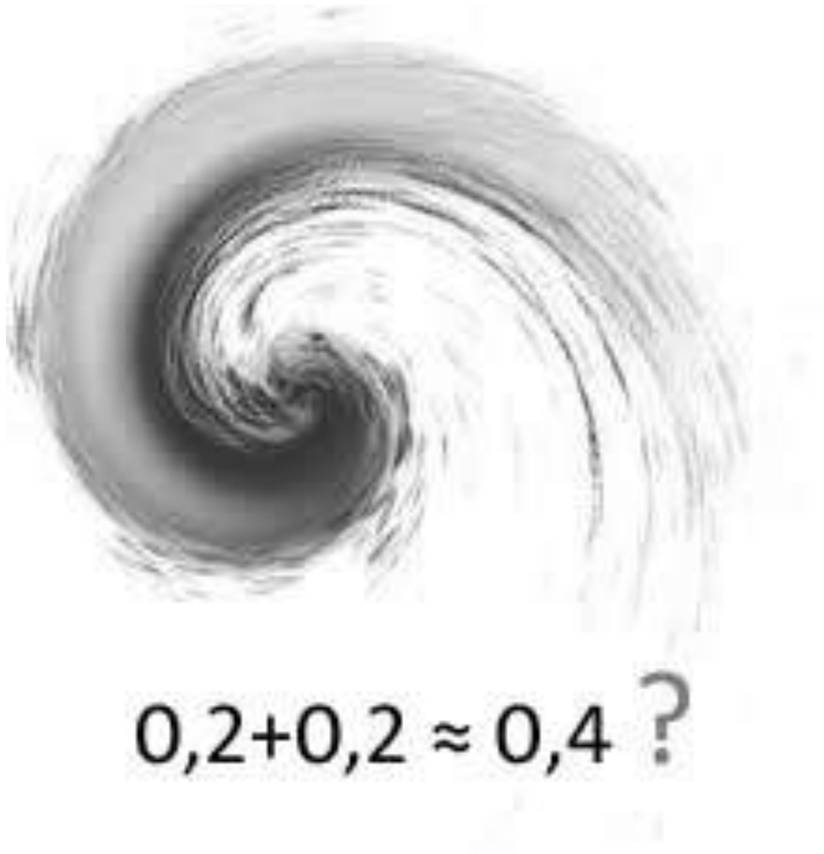
УНИВЕРСИТЕТ ИТМО

А. Г. Зыков, В. И. Поляков

АРИФМЕТИЧЕСКИЕ ОСНОВЫ ЭВМ

Учебное пособие по дисциплине «Дискретная математика»



Санкт-Петербург

2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

А. Г. Зыков, В. И. Поляков

АРИФМЕТИЧЕСКИЕ ОСНОВЫ ЭВМ

Учебное пособие по дисциплине «Дискретная математика»

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург

2016

Зыков А.Г., Поляков В.И. Арифметические основы ЭВМ. – СПб: Университет ИТМО, 2016. – 140 с.

Учебное пособие охватывают раздел «Арифметические основы ЭВМ» дисциплины «Дискретная математика». В пособии рассмотрены особенности представления чисел с фиксированной и плавающей запятой в разных форматах, принятых в персональных ЭВМ, в ЭВМ общего назначения и в микро ЭВМ. Описаны методы выполнения арифметических операций (сложения, умножения и деления) над числами с фиксированной и плавающей запятой, над десятичными числами

Для закрепления теоретического материала необходимо выполнить предлагаемые в пособии домашние задания. Каждое задание содержит свою формулировку, основные положения, знание которых необходимо студенту для его выполнения, а также большое количество примеров, поясняющих способы представления чисел в ЭВМ и принципы выполнения операций над ними. Примеры снабжены необходимыми комментариями. В приложении 1 приводится большое количество вариантов задания, что позволяет в полной мере решить проблему их индивидуализации.

В приложении 2 приводится именной указатель ученых, внесших значительный вклад в развитие математики.

Учебное пособие предназначено для бакалавров, обучающихся по направлениям "Информатика и вычислительная техника" и "Программная инженерия", а также для студентов других технических направлений и специальностей.

УНИВЕРСИТЕТ ИТМО

Университет ИТМО – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

СОДЕРЖАНИЕ

Прежде чем решать задачу – прочитай условие.
Жак Адамар

ПРЕДИСЛОВИЕ

Раздел «Арифметические основы ЭВМ» дисциплины «Дискретная математика» является одним из первых специальных курсов, которые формируют у студентов понимание основополагающих вопросов организации ЭВМ, принципы построения отдельных устройств ЭВМ, их взаимосвязь. Он должен сформировать начальные знания для лучшего понимания последующих спецдисциплин.

Основная цель настоящего учебного пособия – помочь студенту, приступившему к изучению арифметики ЭВМ, приобрести теоретические знания и практические навыки представления чисел и выполнения основных арифметических операций.

Рассматриваемый в пособии теоретический материал сопровождается большим количеством примеров, что делает более понятным излагаемый материал и упрощает выполнение домашних заданий.

Следует отметить, что в последние годы литература, освещающая арифметику ЭВМ, не выпускалась. Пособие, в некоторой части, устраняет этот информационный пробел.

В Приложениях приводятся варианты домашних заданий и именной обзор известных математиков, внесших вклад в формирование арифметики как математической науки.

1. ИСТОРИЯ АРИФМЕТИКИ

1.1. Возникновение арифметики

Арифметика — наука о числах, их свойствах и отношениях — является одной из основных математических наук. Она тесно связана с алгеброй и теорией чисел. Слово «арифметика» происходит от греческого arithmos, что значит «число».

Причиной возникновения арифметики стала практическая потребность в счёте, простейших измерениях и вычислениях. Первые достоверные сведения об арифметических знаниях обнаружены в исторических памятниках Вавилона и Древнего Египта, относящихся к III—II тысячелетиям до н. э. Большой вклад в развитие арифметики внесли греческие математики, в частности пифагорейцы, которые пытались с помощью чисел определить все закономерности мира. В Средние века основными областями применения арифметики были торговля и приближённые вычисления. Арифметика развивалась в первую очередь в Индии и странах ислама и только затем пришла в Западную Европу.

Индийский математик Брахмагупта (VII век) дал определение нуля как результат вычитания из числа самого числа. Он одним из первых установил правила арифметических операций над положительными и отрицательными числами и нулём, рассматривая при этом положительные числа как имущество, а отрицательные числа как долг. Вот как Брахмагупта излагал правила сложения и вычитания:

«Сумма двух имуществ есть имущество».

«Сумма двух долгов есть долг».

«Сумма имущества и долга равна их разности».

Далее Брахмагупта пытался расширить арифметику дав определение деления на ноль. Согласно Брахмагупте:

- Деление нуля на нуль есть нуль.
- Деление положительного или отрицательного числа на нуль есть дробь с нулём в знаменателе.
- Деление нуля на положительное или отрицательное число есть нуль.

Брахмагупта предложил три метода умножения многозначных чисел в столбик (основной и два упрощённых), которые близки к тем, что используются в настоящее время.

Брахмагупта также предложил метод приближённого вычисления квадратного корня, эквивалентный итерационной формуле Ньютона (Newton-Raphson), метод решения некоторых неопределённых квадратных уравнений вида $ax^2+c=y^2$, метод решения неопределённых линейных уравнений вида $ax+c=by$, используя метод последовательных дробей.

Он определил сумму квадратов и кубов первых n чисел через сумму первых n чисел, утверждая, что «Сумма квадратов есть сумма чисел умноженная на удвоенное число шагов, увеличенное на единицу, и делённая на три. Сумма кубов есть квадрат суммы чисел до одного и того же числа».

Брахмагупта предложил интерполяционную формулу второго порядка, являющуюся частным случаем выведенной более чем через 1000 лет интерполяционной формулы Ньютона— Стирлинга.

В XVII веке мореходная астрономия, механика, более сложные коммерческие расчёты поставили перед арифметикой новые запросы к технике вычислений и дали толчок к ее дальнейшему развитию.

Теоретические обоснования представления о числе связаны в первую очередь с определением натурального числа и аксиомами Д. Пеано, сформулированными в 1889 году. За ними последовали строгие определения рациональных, действительных, отрицательных и комплексных чисел. Дальнейшее расширение понятия числа возможно только при отказе от одного из арифметических законов.

Если в двух множествах (наборах предметов) каждый элемент одного набора имеет единственную пару в другом наборе, то эти множества равномощны. Такое фактическое сравнение, когда предметы раскладывались в два ряда, использовалось ещё первобытными племенами при обмене, оно даёт возможность устанавливать количественные соотношения между группами объектов и не требует понятия числа.

В дальнейшем появились естественные эталоны счёта, например, пальцы рук, а затем и множества-эталоны, такие как руки. С появлением эталонов, символизирующих конкретные числа, и связывают возникновение понятия числа. При этом число предметов сравнивали с Луной в небе, количеством глаз, количеством пальцев на руке. Позднее многочисленные эталоны заменились на один наиболее удобный, обычно им становились пальцы рук и/или ног.

Следующим шагом было появление общего понятия натурального числа, отделённого от конкретных предметов. Натуральное число возникло как идеализация конечного множества однородных, устойчивых и неделимых предметов (людей, овец, дней и т. п.); соответственно и действия с числами первоначально отражали реальные действия с такими множествами (объединение, разделение и т. п.). Для праиндоевропейского языка, использовавшего десятеричную систему счисления, уже реконструированы названия числительных до ста включительно. Французский математик Анри Леон Лебег по этому поводу заметил: «*Возможно, что, если бы люди имели одиннадцать пальцев, была бы принята одиннадцатиричная система счисления*».

Для записи результатов счёта использовали зарубки на дереве или костях, узелки на верёвках — искусственные эталоны счёта.

Арифметика возникла в странах Древнего Востока: Вавилоне, Китае, Индии, Египте. Например, египетский папирус Ринда (названный по имени его владельца Г. Ринда) относится к XX в. до н.э. Среди прочих сведений он содержит разложения дроби на сумму дробей с числителем, равным единице, например,

$$\frac{2}{73} = \frac{1}{60} + \frac{1}{219} + \frac{1}{292} + \frac{1}{365}.$$

Накопленные в странах Древнего Востока сокровища математических знаний были развиты и продолжены учеными Древней Греции. Много имен ученых, занимавшихся арифметикой в античном мире, сохранила нам история - Анаксагор и Зенон, Евклид, Архимед, Эратосфен и Диофант. Яркой звездой сверкает здесь имя Пифагора (VI в. до н.э.). Пифагорейцы (ученики и последователи Пифагора) преклонялись перед числами, считая, что в них заключена вся гармония мира. Отдельным числам и парам чисел приписывались особые свойства. В большом почете были числа 7 и 36, тогда же было обращено внимание на так называемые совершенные числа, дружественные числа и т. п.

В средние века развитие арифметики также связано с Востоком: Индией, странами арабского мира и Средней Азии. От индийцев пришли к нам цифры, которыми мы пользуемся, нуль и позиционная система счисления; от аль-Каши (XV в.), работавшего в Самаркандской обсерватории Улугбека, - десятичные дроби.

Благодаря развитию торговли и влиянию восточной культуры начиная с XIII в. повышается интерес к арифметике и в Европе. Следует вспомнить имя итальянского ученого Леонардо Пизанского (Фибоначчи), сочинение которого «Книга абака» знакомило европейцев с основными достижениями математики Востока и явилось началом многих исследований в арифметике и алгебре.

Вместе с изобретением книгопечатания (середина XV в.) появились первые печатные математические книги. Первая печатная книга по арифметике была издана в Италии в 1478 г. В «Полной арифметике» немецкого математика М. Штифеля (начало XVI в.) уже есть отрицательные числа и даже идея логарифмирования.

Примерно с XVI в. развитие чисто арифметических вопросов влилось в русло алгебры – в качестве значительной вехи можно отметить появление работ ученого из Франции Франсуа Виета, в которых числа обозначены буквами. Начиная с этого времени основные арифметические правила осознаются уже окончательно с позиций алгебры.

Основной объект арифметики – число. Натуральные числа, т.е. числа 1, 2, 3, 4, . и т. д., возникли из счета конкретных предметов. Прошло много тысячелетий, прежде чем человек усвоил, что два фазана, две руки, два человека и т.д. можно назвать одним и тем же словом «два». Важная задача арифметики – научиться преодолевать конкретный смысл названий считаемых предметов, отвлекаться от их формы, размера, цвета и т. п. Уже у Фибоначчи есть задача: «Семь старух идут в Рим. У каждой по 7 мулов, каждый мул несет по 7 мешков, в каждом мешке по 7 хлебов, в каждом хлебе по 7 ножей, каждый нож в 7 ножнах. Сколько всех?» Для решения задачи придется складывать вместе и старух, и мулов, и мешки, и хлеба.

Под «наукой чисел» понимаются две науки: практическая и теоретическая. Практическая изучает числа постольку, поскольку речь идет о числах считаемых. Эту науку применяют в рыночных и гражданских делах. Теоретическая наука чисел изучает числа в абсолютном смысле, отвлеченные разумом от тел и всего, что поддается в них счету.

В Центральной Америке в основном использовалась двадцатеричная система счисления. Жрецы майя с Юкатана создали её искусственно и использовали для календарных расчётов. В ней второй разряд был неполным и доходил только до **19**. В качестве дополнительного основания использовалось число **5**. Календарь майя представлял собой позиционную систему, где на каждой позиции располагалось божество с определённым количеством знаков. При письме божества не изображали, а для обозначения пустого разряда использовали символ в виде открытой раковины или глаза. В Южной Америке для записи чисел использовалась узловая нумерация, или кипу.

Арифметические расчёты проводились с помощью юпаны, которая представляет собой аналог абака, однако в связи с особенностями системы счисления, арифметика, не связанная с астрономическими расчётами, получила слабое развитие.

В эпоху раннего феодализма в Западной Европе потребности в науке не выходили за пределы вопросов практической арифметики и геометрии. Книги содержали начальные сведения о семи свободных искусствах, включая арифметику. Наиболее популярными были сочинения Боэция, датируемые VI веком, который в числе прочего перевёл на латинский язык «Арифметику» Никомаха (1-я пол. 2 в. н. э.) с собственными числовыми примерами и часть «Начал» Евклида без строгих доказательств.

Через Испанию и Сицилию в X веке начали завязываться научные связи с арабским миром. В это время Каталонию посетил учёных монах Герберт, ставший позднее папой Сильвестром II. Ему приписываются такие сочинения, как «Книжка о делении чисел» и «Правила счёта на абаке». В обеих книгах числа пишутся словами или римскими цифрами. Герберт называл вычислителей на абаке «абацистами».

В XII—XIII веках в Европе появились латинские переводы арабских книг по арифметике. Основные переводы были сделаны с арабского на территории Пиренейского полуострова в Толедо под покровительством архиепископа Раймонда I, а также в Барселоне и Сеговии. Приверженцы представленной в книгах десятичной позиционной нумерации стали называться «алгористами» по имени математика ал-Хорезми в латинской форме. Постепенно новая система взяла верх. Основным её преимуществом явилось упрощение арифметических операций. Вместе с тем в Германии, Франции и Англии новые цифры не употреблялись до конца XV века.

Далее переводов пошёл итальянец Леонардо Пизанский (Фибоначчи), живший в XIII веке. В своём основном труде «Книга абака», написанном в 1202 году, он выступил сторонником индийской системы нумерации и считал приёмы абацистов отклонением от верного пути. Пять глав книги посвящены арифметике целых чисел. Фибоначчи использовал нуль как настоящее число, проводил проверку с помощью девятки, знал признаки делимости на 2, 3, 5, 9, приводил дроби к общему знаменателю с помощью наименьшего общего кратного знаменателей, излагал тройное правило, правила пяти, семи, девяти величин и другие правила пропорций, решал задачи на смешение, оперировал суммированием рядов, включая один из возвратных рядов, или ряд Фибоначчи, разъяснял способы приближённого вычисления квадратных и кубических корней. В «Книге абака» приводятся вместе с доказательствами разнообразные методы и задачи, которые широко использовались в сочинениях поздних математиков.

Преподавателю Оксфордского университета магистру Томасу Брадвардину (начало XIV века), ставшему впоследствии архиепископом Кентерберийским, принадлежит книга «Теоретическая арифметика», которая является сокращённым вариантом «Арифметики» Боэция. Кроме того, этот мыслитель в своих работах по механике использовал «половинное» отношение, на основе которого французский математик Николай Орем развил учение о дробных показателях степеней в своём трактате «Алгоризм отношений», а также подошёл к понятию иррационального показателя которое можно заключать между достаточно близкими целыми и дробными, и осуществил обобщение возведения в степень на положительные дробные показатели. Работы Орема были напечатаны только в XIX веке.

В 1484 году увидела свет рукопись французского бакалавра медицины Никола Шюке «Наука о числах в трёх частях», в которой он, в частности, сопоставляет произведение членов арифметической прогрессии и сумму членов геометрической прогрессии, предвосхищая логарифмы, предлагает число считать корнем первой степени из себя самого, а также использует отрицательные и нулевой показатели степени. В 1487 году Фра Лука Бартоломео де Пачоли написал свою «Сумму [знаний] по арифметике, геометрии, отношениям и пропорциональности». В книге, изданной в Венеции в 1494 году, Пачоли изложил различные приёмы арифметических действий, пользуясь при этом алгебраическими символами. Сложение Пачоли обозначал знаком $\tilde{p}$, а вычитание — $\tilde{m}$. Кроме того, он использовал для отрицательного числа выражение «меньше нуля» и сформулировал правило, по которому меняются знаки при умножении чисел.

В работе Джероламо Кардано «Великое искусство» в XVI веке было введено понятие мнимых величин, или софистических. Хотя сам Кардано считал их бесполезными, они были использованы Рафаэлем Бомбелли для решения кубических уравнений, который также ввёл правила умножения мнимых и действительных чисел. В том же веке в Европе получают распространение десятичные дроби. Они появляются в работах Франсуа Виета, Иммануила Бонфиса, Симона Стевина. В 1585 году в книге «Десятая» последний агитировал за повсеместное использование десятичных дробей. В том же году в работе «Арифметика» он дал принципиально новое определение иррациональному числу как «с помощью чего выражается количество всякой вещи». Стевин считал иррациональные и отчасти отрицательные числа такими же настоящими, как и дроби, а также полагал делимой единицу.

М. Штифель в своей «Полной арифметике» вводит определение и алгоритм деления отношения на отношение, он также даёт геометрическое толкование отрицательных чисел («ниже, чем ничего») и проводит аналогию между введением отрицательных и иррациональных чисел. В 1569 году французский профессор Пётр Рамус, которому указом короля было запрещено выступать с критикой Аристотеля, написал «Курс математики в тридцать одной книге», в котором пытался дать математике новое обоснование, основанное не на геометрии, а на арифметике.

1.2. Десятичная арифметика и расширение понятия числа

В XVII веке мореходная астрономия, механика, более сложные коммерческие расчёты поставили перед арифметикой новые запросы к технике вычислений и дали толчок к дальнейшему развитию.

Значительному изменению подверглось понятие числа. Если ранее к области чисел в большинстве своём относили только положительные рациональные числа, то начиная с XVI века всё более признавались иррациональные и отрицательные числа. В «Геометрии» Рене Декарта 1637 года устанавливается связь между арифметикой и геометрическими построениями, причём числовые величины, вопреки Евклиду, фактически лишаются размерности и отделяются от геометрии. Отношение любой величины к единичному эталону является в данном случае эквивалентом действительного числа, при этом рассуждения оставались верны как для соизмеримых, так и для несоизмеримых отрезков, последние сам Декарт называл «глухими числами» (*nombres sourds*). Ньютон в своих лекциях также делит числа на три вида: целые (измеряются единицей), дробные (кратные доли единицы) и иррациональные (несоизмеримые с единицей). С 1710 года такое определение числа прочно входит во все учебники.

Дроби появились в глубокой древности, когда древний человек решил разделить добычу с себе подобным. При разделе добычи, при измерениях величин, да и в других похожих случаях люди столкнулись с необходимостью делить что-то на равные части, т.е. наряду с необходимостью считать предметы у людей с древних времён появилась потребность измерять длину, площадь, объём, время и другие величины. Результат измерений не всегда удавалось выразить натуральным числом, приходилось учитывать и части употребляемой меры. Так возникли дроби. Так русское слово *дробь*, как и его аналоги в других языках, происходят от латинского слова *fractura*, которое, в свою очередь, является переводом арабского с тем же значением: *ломать, раздроблять*. Первой дробью, с которой познакомились люди, была половина. Следующей дробью была треть. В древности у разных народов использовались разные дроби и разные записи дробей.

У римлян основной единицей измерения массы, а также и денежной единицей служил «асс». Асс делился на 12 равных частей - унций. Из них складывали все дроби со знаменателем 12, то есть $^1/_{12}$, $^2/_{12}$, $^3/_{12}$… Со временем унции стали применяться для измерения любых величин.

Так возникли римские *двенадцатеричные дроби*, то есть дроби, у которых знаменателем всегда было число **12**. Вместо $^{1}/_{12}$ римляне говорили «одна унция», $^{5}/_{12}$ – «пять унций» и т.д. Три унции назывались четвертью, четыре унции – третью, шесть унций – половиной.

На протяжении многих веков египтяне именовали дроби “ломаным числом”, а первая дробь, с которой они познакомились была 1/2. За ней последовали 1/4, 1/8, 1/16, …, затем 1/3, 1/6, …, т.е. самые простые дроби называемые единичными или *основными дробями*. У них числитель всегда единица. Лишь значительно позже у греков, затем у индийцев и других народов стали входить в употребление и дроби общего вида, называемые обыкновенными, у которых числитель и знаменатель могут быть любыми натуральными числами.

В Древнем Египте архитектура достигла высокого развития. Для того, чтобы строить грандиозные пирамиды и храмы, чтобы вычислять длины, площади и объемы фигур, необходимо было знать арифметику.

Из расшифрованных сведений на папирусах ученые узнали, что египтяне 4 000 лет назад имели десятичную (но не позиционную) систему счисления, умели решать многие задачи, связанные с потребностями строительства, торговли и военного дела.

Одним из первых известных упоминаний о египетских дробях является математический папирус Ринда. Три более древних текста, в которых упоминаются египетские дроби — это Египетский математический кожаный свиток, Московский математический папирус и Деревянная табличка Ахмима. Папирус Ринда включает таблицу египетских дробей для рациональных чисел вида $2/n$, а также 84 математические задачи, их решения и ответы, записанные в виде египетских дробей.

Жители древнего Вавилона примерно за три тысячи лет до нашей эры создали систему мер, аналогичную нашей метрической, только в основе её лежало не число 10, а число 60.

Исследователи по-разному объясняют появление у вавилонян шестидесятеричной системы счисления. Скорее всего здесь учитывалось основание 60, которое кратно 2, 3, 4, 5, 6, 10, 12, 15, 20, 30 и 60, что значительно облегчает всякие расчеты.

Шестидесятые доли были привычны в жизни вавилонян. Они пользовались *шестидесятеричными* дробями, имеющими знаменателем всегда число 60 или его степени: 60^2, 60^3 и т.д. В этом отношении шестидесятеричные дроби можно сравнить с нашими десятичными дробями.

Вавилонская математика оказала влияние на греческую математику. Следы вавилонской шестидесятеричной системы счисления удержались в современной науке при измерении времени и углов. До наших дней сохранилось деление часа на 60 мин., минуты на 60 с, окружности на 360 градусов.

Вавилоняне внесли ценный вклад в развитие астрономии. Шестидесятеричными дробями пользовались в астрономии ученые всех народов до XVII века, называя их *астрономическими* дробями. В отличие от них, дроби общего вида, которыми пользуемся мы, были названы *обыкновенными.*

В китайской «Математике в девяти разделах» уже имеют место сокращения дробей и все действия с дробями.

У индийского математика Брахмагупты можно найти достаточно развитую систему дробей. У него встречаются разные дроби: и основные, и производные с любым числителем. Числитель и знаменатель записываются так же, как и у нас сейчас, но без горизонтальной черты, а просто размещаются один над другим.

Арабы первыми начали отделять чертой числитель от знаменателя.

Леонардо Пизанский уже записывает дроби, помещая в случае смешанного числа целое число справа, но читает так, как принято у нас. Иордан Неморарий (XIII в.) выполняет деление дробей с помощью деления числителя на числитель и знаменателя на знаменатель, уподобляя деление умножению. Для этого приходится члены первой дроби дополнять множителями.

В XV – XVI столетиях учение о дробях приобретает уже знакомый нам вид и оформляется приблизительно в те самые разделы, которые встречаются в наших учебниках.

Периодические дроби появились ещё в работе «Десятичный счёт» (*Logistica decimalis*) Иоганн Гартман Бейера в 1603 году. Работу над ними продолжил Джон Валлис в «Трактате по алгебре» в 1685 году, где он определил, что для несократимой дроби число цифр периода меньше или равно. Валлис, кроме того, показал конечность дроби со знаменателем вида, он также знал, что невозможно иррациональные числа выразить периодическими дробями.

В начале XVII века Д. Непер изобрёл логарифмы. Применение логарифмов и десятичных дробей, включение в арифметику понятия иррационального числа как последовательности рациональных приближений расширили область применения арифметики к концу XVII века и определили фундаментальное значение науки для изучения непрерывных величин.

В XVIII веке продолжились работы с десятичными дробями, в частности с бесконечными и периодическими десятичными дробями. Тот факт, что любая периодическая дробь является рациональным числом, а также, что любая несократимая дробь, содержащая в знаменателе отличные от двух и пяти простые делители, разлагается в периодическую, доказал в середине XVIII века И. Г. Ламберт. В работе И. Гаусса «Арифметические исследования» с помощью теории степенных вычетов представлены более глубокие свойства периодических дробей. Вместе с тем в учебниках того времени десятичные дроби затрагиваются мимоходом или не упоминаются вовсе. Непрерывными дробями занимался Л. Эйлер, который впервые представил приёмы преобразования бесконечных непрерывных дробей в бесконечные ряды, а затем посвятил им целую главу в первом томе своего «Введения в анализ бесконечных» в 1748 году. Эйлеру принадлежит доказательство того, что всякое рациональное число может быть представлено в виде конечной непрерывной дроби, а также, что периодическая непрерывная дробь с единицами в числителях является корнем квадратного уравнения. Обратное было доказано Ж. Л. Лагранжем в 1768 году. В XVIII веке у Эйлера и его учеников арифметика приобретает современные формы.

А. Жирар и Р. Декарт геометрически интерпретировали отрицательные числа противоположно направленными отрезками. Несмотря на то, что уже Декарт считал отрицательные корни уравнений наряду с положительными действительными корнями (в противовес мнимым), некоторые свойства отрицательных чисел долгое время оставались неясными. 1 сентября 1742 года Эйлер в письме Бернулли впервые высказал утверждение, что корни любого алгебраического уравнения имеют вид $a+b\sqrt{-1}$. В 1747 году в «Размышлениях об общей причине ветров» Ж. Даламбер показал, что $(a+b\sqrt{-1})^{g+h\sqrt{-1}} = A+B\sqrt{-1}$. В «Исследованиях о мнимых корнях» Эйлер тем не менее определяет мнимое число как такое, которое «ни больше нуля, ни меньше нуля, ни равно нулю», а «нечто невозможное». При этом он доказывает теорему, что всякое мнимое число образовано суммой действительного числа M и произведения действительного числа N на $\sqrt{-1}$. Задача решалась для отдельных функций, круг операций над мнимыми числами очерчен не был. Кроме того, были проблемы с геометрическим толкованием мнимых чисел. Первую попытку сделал Валлис, который полагал мнимые числа отрезками, перпендикулярными вещественным, затем была работа Генриха Кюна в 1753 году, в которой он считал мнимым числом сторону квадрата с отрицательной площадью. Развить определение Валлиса удалось К. Весселю и Ж. Р. Аргану только на рубеже XVIII—XIX веков.

В 30-х годах XVII века Ферма выделил теорию чисел как отдельную область арифметики, по его мнению, лишь слегка затронутую Евклидом и, возможно, Диофантом. П. Ферма занимался решением диофантовых уравнений и делимостью целых чисел. Он сформулировал ряд утверждений без доказательства, в частности малую и великую теоремы Ферма. Ферма не написал никакого специального труда по теории чисел, его предложения сохранились лишь в переписке, а также в виде замечаний к «Арифметике» Диофанта.

Только через 70 лет работы Ферма привлекли внимание Эйлера, который занимался теорией чисел несколько десятилетий. Ей посвящено четыре с половиной тома 30-томной математической серии Эйлера. Эйлер занимался обобщением малой теоремы Ферма, а также доказательством великой теоремы Ферма для случая $n = 3$. Эйлер первым начал применять для задач теории чисел аппарат других разделов математики, в первую очередь математического анализа. Он сформулировал метод производящих функций, тождество Эйлера, а также задачи, связанные со сложением простых чисел.

Считается, что именно после работ Эйлера теория чисел стала отдельной наукой.

1.3. Обоснования арифметики

С работами Н. И. Лобачевского по геометрии связан процесс критического пересмотра основ математики, который случился в XIX веке. Ещё в XVIII веке начались попытки дать теоретические обоснования представлениям о числе. Поначалу это касалось только арифметики натуральных чисел, для которой применялись различные аксиомы и определения, зачастую избыточные и недостаточные одновременно, во многом заимствованные из «Начал» Евклида. Также обстояло дело с основными законами арифметики: коммутативный и ассоциативный законы для умножения и сложения упоминались довольно часто, дистрибутивный закон относительно сложения для умножения — реже, а все пять законов — крайне редко. Г. В. Лейбниц первый поставил задачу дедуктивного построения арифметики и, в частности, показал необходимость доказательства равенства «два плюс два равно четыре» в своих «Новых опытах о человеческом разуме» в 1705 году. В попытках решить этот вопрос свои аксиомы представили Х. Вольф в 1770 году, И. Шульц в 1790 году, Г. С. Ом в 1822 году, Г. Грассман в 1861 году и, наконец, Д. Пеано в 1889 году.

Сложность выделения основных положений арифметики связана с простотой её начальных положений. Только в середине XIX века Грассман выбрал систему основных аксиом, определяющих сложение и умножение. Система позволяла вывести остальные положения арифметики как логическое следствие из аксиом. На основе аксиом были доказаны коммутативный, ассоциативный и дистрибутивный законы сложения и умножения, введено понятие дроби как пары целых чисел с определёнными законами сравнения и действий. Работа Грассмана была продолжена Пеано. Были и дальнейшие попытки приблизиться к полному теоретическому обоснованию арифметики натуральных чисел, в частности работы Д. Гильберта, пока в 1932 году К. Гёдель не доказал теорему о неполноте.

Аналогичным образом были попытки дать теоретическое обоснование рациональным дробям, для которых выделялись две концепции: равные доли единицы или отношение двух однородных величин. Для рациональных дробей необходимо было доказать верность равенства $\frac{am}{bm} = \frac{a}{b}$ (m - натуральное число), которые использовались при сложении, вычитании и сокращении дробей. Равенство было тривиальным в теории отношений, но совсем не очевидным в независимой от неё концепции. Вместе с тем его просто считали верным. Арифметика дробей была обоснована Ж. Таннери в 1894 году, в его модели дроби представлялись парами целых чисел.

В 1758 году в «Первых основаниях арифметики, геометрии, плоской и сферической тригонометрии и перспективы» А. Кестнер выступил за обоснование всех арифметических понятий через целое число. Таким образом, он определил, в порядке следования в книге, натуральные числа, дроби, отрицательные числа, десятичные дроби, иррациональные числа и только затем теорию отношений. Операции над иррациональными числами стали исследовать, опираясь на их приближения рациональными дробями. При этом существование иррациональных чисел принималось заранее, а сами они трактовались как пределы последовательности рациональных чисел. Для иррациональных чисел пользовались определением И. Ньютона как отношения несоизмеримых величин (подобное определение дал и Л. Эйлер). Аналогичным образом трактовал иррациональные числа П. А. Рахманов в «Новой теории содержания и пропорции геометрически соизмеримых и несоизмеримых количеств, и в последнем случае основанной на теории пределов». И только во второй половине XIX века появляются строгие теории действительного числа, сформулированные Ш. Мере, Г. Кантором, Р. Дедекиндом и К. Вейерштрассом.

В формировании теории отрицательных чисел основную проблему составляло утверждение, что отрицательное число меньше нуля, то есть меньше, чем ничего. Строгое определение отрицательных чисел отсутствовало, при этом были попытки сформулировать правила знаков («минус на плюс даёт минус» и «минус на минус даёт плюс»). Французский математик Л. Карно в 1813 году писал: «*Метафизика правила знаков при более глубоком изучении её обнаруживает, пожалуй, бо́льшие трудности, чем метафизика бесконечно малых количеств; это правило никогда не было доказано вполне удовлетворительным образом, и, по-видимому, оно даже не может быть доказано достаточно удовлетворительно*». Первые попытки сформулировать теорию отрицательных чисел были сделаны в середине XIX века и принадлежат У. Гамильтону и Г. Грассману.

Полное геометрическое толкование комплексных чисел было предложено Каспаром Весселем в «Опыте об аналитическом представлении направления и его применениях, преимущественно к решению плоских и сферических многоугольников» в 1799 году. Вессель хотел работать с направленными отрезками на плоскости с помощью алгебраических операций, но для вещественных чисел они позволяли только изменить направление на противоположное, а не задать произвольное направление. Вессель использовал основные единицы +1, -1, $+\varepsilon$ $-\varepsilon$ и, используя правила умножения, заключил, что $\varepsilon=\sqrt{-1}$. Работы Весселя оставались незамеченными около 100 лет. За это время своё толкование мнимых чисел представили Жан Робер Арган в 1813—1814 годах, Шайсс в 1831 году в «Теории биквадратичных вычетов», а также Гамильтон в 1832 году, который построил арифметическую теорию, рассматривая комплексные числа как пары действительных.

Вессель пытался обобщить теорию на трёхмерное пространство, но это ему не удалось. Вопрос оставался открытым до тех пор, пока Гамильтон не построил теорию кватернионов, при умножении которых не выполняется коммутативный закон. При этом исследования К. Вейерштрасса, Ф. Фробениуса и Б. Пирса показали, что отказаться от какого-либо из арифметических законов придётся при любом расширении понятия числа за пределы комплексных чисел.

1.4. История арифметики в России

На Руси использовали аналог древнегреческой нумерации с использованием букв кириллицы или глаголицы. Вместе с тем, в отличие от многих народов, которые придали числовые значения новым буквам, на Руси за малым исключением продолжали использовать буквы греческого алфавита или похожие. Числа писались в том же порядке, что и произносились, то есть в числе 15 сначала шёл знак для пяти, а потом для десятка, в то время как в числе 25 — сначала для 2, а потом для 5. Наибольшее распространение получила кириллическая нумерация. Арифметика в России называлась *щётная мудрость*, или *«Чёрная книга»*, откуда произошло *чернокнижие*. Книги по арифметике мало кто мог прочитать и понять, так как они содержали арифметические правила и выкладки и были составлены из малопонятных знаков.

XI веком датируются математические задачи из юридического сборника «Русская Правда» — первый дошедший до нас математический документ Древней Руси, содержащий задачи о приплоде скота, количестве зерна и сена, собираемого с определённой площади. Дальнейшее развитие науки было остановлено монголо-татарским нашествием. В конце XVI века появилась «Книга, рекома по гречески Арифметика, по-немецки Алгорисма, а по-русски — Цифирная счетная мудрость», которая, по мнению Николая Михайловича Карамзина, и была первой русской арифметикой.

Считается, что арабские цифры были введены в России после первого заграничного путешествия Петра I, когда он в 1698 году привёз из Лондона морских офицеров. Одним из офицеров был Фергарсон, который, как полагают, ввёл в России арабские цифры. Но на самом деле они пришли в Россию до Петра, в 1647 году в Москве по указу царя Алексея Михайловича был напечатан русский воинский устав, в котором использовались арабские цифры. Книги же, напечатанные на русском языке за пределами России, содержали арабские цифры с начала XVI века. При этом в тексте использовалась славянская нумерация, а для вычислений — арабская.

В 1682 году в Москве была напечатана первая книга математического содержания «Считание удобное, которым всякий человек купующий или продающий зело удобно изыскати может, число всякие вещи», которая содержала таблицы умножения до 100 и использовала славянскую нумерацию. Второе издание этой книги, выпущенное в 1714 году в Петербурге, было напечатано гражданским шрифтом и арабскими цифрами. В 1699 году в Амстердаме вышла книга «Краткое и полезное руковедение в аритметыку, или во обучение и познание всякого счёту в сочетании всяких вещей» — первый учебник арифметики на русском языке. Книга была составлена Ильёй Фёдоровичем Копиевичем (или Копиевским) по заказу архангельских купцов. Она не удовлетворила заказчиков и распространения не получила.

В русских рукописных арифметиках XVII века дроби называли долями, позднее «ломаными числами». В старых руководствах находим следующие названия дробей на Руси:

$\frac{1}{2}$ – половина, полтина	$^1/_3$ – треть
$\frac{1}{4}$ – четь	$^1/_6$ – полтреть
$^1/_8$ – полчеть	$^1/_{12}$ –полполтреть
$^1/_{16}$ – полполчеть	$^1/_{24}$ – полполполтреть (малая треть)
$^1/_{32}$ – полполполчеть (малая четь)	$^1/_5$ – пятина
$^1/_7$ – седьмина	$^1/_{10}$ – десятина

Славянская нумерация употреблялась в России до XVI века, затем в страну начала постепенно проникать десятичная позиционная система счисления. Она окончательно вытеснила славянскую нумерацию при Петре I.

В России первый учебник арифметики Леонтия Магницкого был напечатан в 1703 году. В «Арифметике» Магницкого, вслед за остальной Европой, используется счёт по числу пальцев на руках: числа от 1 до 9 названы «перстами», нуль — «низачто», десятки — «составами», а остальные числа — «сочинениями».

В «Арифметике» Л. Ф. Магницкого даются уже принятые сейчас термины для больших чисел (миллион, биллион, триллион, квадриллион), дойдя до 10^{24}, автор заявляет, что больших чисел не потребуется.

Русская математическая терминология не является чем-то вечным, застывшим. Лет 200-300 назад она весьма сильно отличалась от современной. Русская терминология обогащалась как за счет создания новых названий математических понятий с русскими корнями, так и за счет заимствования некоторых иностранных терминов. Возможно, вы не замечали одной странности в образовании названий чисел: одиннадцать=один-на-дцать=один-надесять, двенадцать = два-на-десять и так далее, двадцать = два-десять, тридцать = три-десять, пятьдесят, шестьдесят и так далее вообще понятно. Но в отличие от общего правила числа 40 и 90 получили названия «сорок» и «девяносто», а не «четырьдцать» или «четырьдесят» и «девятьдесят».

Известно, что в древние времена в качестве денег употреблялись меха - позднее кожаные деньги (кусочки кожи с клеймами). Указ Петра I еще в 1700 году утверждает, что «в Калуге и в иных городах вместо серебряных денежек торгуют кожаными». По «Толковому словарю живого великорусского языка» Владимира Даля, 40 собольих мехов составляли полную шубу и вкладывались в «чехол или в сорочку». Отсюда название числа «сорок». Аналогичное образование имен числительных наблюдается и в других языках. В Дании, например, продают рыбу партиями в 80 голов, надетыми на жердь. Название «жердь» стало у них и названием числа 80.

Числительное «девяносто» производят от слов «девять до ста»: между числами 90 и 100 в натуральном ряду стоят девять чисел.

В начале XIX в. В России было создано Министерство народного просвещения, возникли учебные округа, и гимназии стали открываться во всех крупных городах России. При этом содержание курса математики было довольно обширным — алгебра, тригонометрия, приложения к физике и др. Начали открываться новые университеты — в Казани и Харькове (1804), в Петербурге (1819), в Киеве (1834). Все они в обязательном порядке имели физико-математический факультет. В XIX веке молодая российская математика уже выдвинула учёных мирового уровня. Михаил Васильевич Остроградский разрабатывал преимущественно прикладной математический анализ. В его работах исследуется распространение тепла, волновое уравнение, теория упругости, электромагнетизм. Занимался также теорией чисел. Академик пяти мировых академий. Николай Иванович Лобачевский прославился своей самоотверженной борьбой против догмата евклидовости пространства. Он построил геометрию Лобачевского и глубоко исследовал её необычные свойства. Виктор Яковлевич Буняковский — чрезвычайно разносторонний математик, изобретатель, признанный авторитет по теории чисел и теории вероятностей. Автор фундаментального труда «Основания математической теории вероятностей». Пафнутий Львович Чебышев, выдающийся русский математик-универсал. Он сделал множество открытий в самых разных, далёких друг от друга, областях математики — теории чисел, теории вероятностей, теории приближения функций. Учитель А. М. Ляпунова и А. А. Маркова.

Во второй половине 19 века российская математика, при общем прикладном уклоне, публикует и немало фундаментальных результатов. Несколько важных открытий общего характера сделала Софья Ковалевская.

Наиболее важные исследования относятся к теории вращения твёрдого тела. Ковалевская открыла третий классический случай разрешимости задачи о вращении твёрдого тела вокруг неподвижной точки.

В конце XIX — начале XX века выходит на историческую сцену новое поколение крупных российских математиков: Дмитрий Александрович Граве, А. Н. Крылов, А. М. Ляпунов, В. И. Смирнов, В. А. Стеклов и другие.

Андрей Андреевич Марков— выдающийся русский математик, внёс большой вклад в теорию вероятностей, математический анализ и теорию чисел.

А. А. Марков является первооткрывателем обширного класса стохастических процессов с дискретной и непрерывной временной компонентой, названных его именем. Марковские процессы обладают следующим (Марковским) свойством: следующее состояние процесса зависит, вероятностно, только от текущего состояния. В то время, когда эта теория была построена, она считалась весьма абстрактной, однако в настоящее время практические применения данной теории чрезвычайно многочисленны. Теория цепей Маркова выросла в огромную и весьма важную область научных исследований — теорию Марковских случайных процессов, которая в свою очередь представляет основу общей теории стохастических процессов.

Советская математическая школа окончательно оформилась в 1930-е годы и вскоре стала одной из ведущих в мире. Больших успехов достигли советские математики как в традиционных, так и в новых областях математики — топология, теория меры, теория функций действительного и комплексного переменного, ряды Фурье, теория множеств, теория вероятностей и др.

Перечислим некоторые крупные открытия советских математиков.

А. Н. Колмогоров разработал аксиоматику теории вероятностей, сразу ставшую общепризнанным фундаментом этой науки.

И. М. Виноградов и Ю. В. Линник внесли определяющий вклад в решение «проблемы Варинга». Л. Г. Шнирельман и И. М. Виноградов в 1930-е годы далеко продвинули решение «проблемы Гольдбаха». А. О. Гельфонд решил 7-ю проблему Гильберта: всякое алгебраическое число, отличное от 0 и 1, будучи возведено в иррациональную степень, дает трансцендентное число. И. Р. Шафаревич доказал общий закон взаимности степенных вычетов.

А. Д. Александров, родоначальник так называемой геометрии Александрова (раздела метрической геометрии), развил синтетический подход к дифференциальной геометрии. Этот раздел повлиял на формирование геометрической теории групп, в частности теории гиперболических групп.

П. С. Александров создал теорию компактных топологических пространств. Л. С. Понтрягин стал одним из основоположников современной алгебраической топологии.

А. И. Мальцев нашёл необходимые и достаточные условия упорядочиваемости группы, доказал фундаментальную теорему о представлении произвольной группы Ли в виде прямого произведения её максимальной компактной подгруппы на евклидово пространство. Он же осуществил классификацию полупростых подгрупп классических групп Ли. Л. С. Понтрягин создал чрезвычайно общую теорию характеров топологических абелевых групп.

Н. Г. Чеботарёв и И. Р. Шафаревич успешно использовали теорию Галуа для решения множества алгебраических проблем. В частности, Шафаревич установил, что для поля алгебраических чисел конечной степени всегда существует алгебраическое расширение, имеющее заданную разрешимую группу Галуа.

С. Н. Бернштейн решил 19-ю проблему Гильберта. Д. Е. Меньшов доказал, что всякая конечная периодическая измеримая функция почти всюду представима сходящимся тригонометрическим рядом. Значительный вклад был внесен в теорию дифференциальных уравнений и функциональный анализ.

Владимир Игоревич Арнольд решил тринадцатую проблему Гильберта.

Десятую проблему решил Юрий Владимирович Матиясевич.

Григорий Яковлевич Перельман доказал гипотезу Пуанкаре.

2. ОСНОВНЫЕ ПОНЯТИЯ

Определение 1. *Системой счисления* называется совокупность цифровых знаков и правил их записи, применяемая для однозначного изображения чисел.

Определение 2. Под *системой счисления* понимается способ представления любого числа посредством некоторого алфавита символов, называемых «цифрами».

Определение 3. *Системой счисления* называют систему приемов и правил, позволяющих установить взаимно однозначное соответствие между любым числом и его представлением в виде конечного числа символов, называемых «цифрами».

Понятие системы счисления включает в себя:

- Алфавит, используемый для записи чисел (цифры, знаки);
- Способ записи чисел;
- Однозначность представления любого числа.

Системы счисления принято разделять на два класса:

- позиционные;
- непозиционные.

Для ***позиционных*** систем счисления значение каждой цифры однозначно определяется ее положением (позицией) в числе.

Для ***непозиционных*** систем счисления значение цифры не зависит от ее положения в числе.

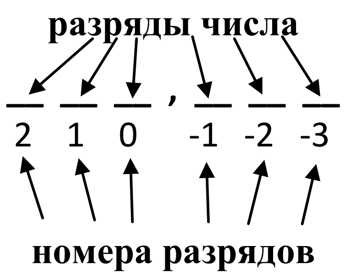
Пример. Римская системы счисления:

$$LXVI = (66)_{10}\ ;$$
$$XLIV = (44)_{10}\ .$$

Два числа состоят из одинаковых цифр (L = 50, X = 10, V = 5, I = 1), однако имеют разные значения.

Римскую систему счисления нельзя считать классической непозиционной.

Для позиционных систем счисления каждая позиция в числе, на которой может находиться цифра, называется ***разрядом числа***. Нумерацию разрядов принято производить влево и вправо от запятой следующим образом:



Под ***весом разряда*** принято понимать количественное значение цифры данного разряда в числе. Фактически, вес разряда представляет собой множитель, на который умножается цифра этого разряда при получении значения числа.

Для системы счисления с основанием $\boldsymbol{S}$ вес i-го разряда определяется в виде:

$$\boldsymbol{V_i = S^i}.$$

Основание системы счисления показывает, во сколько раз вес i-ого разряда числа больше веса предыдущего (i - 1)-ого разряда.

Под основанием системы счисления можно понимать:

- Количество разнообразных цифр, используемых при записи чисел.
- Основание степени для определения веса разряда.

Запись значения основания в любой системе счисления имеет вид:

$$S = 10.$$

При выборе системы счисления для применения в ЭВМ следует учитывать совокупность различных параметров, среди которых можно выделить:

– наличие физических элементов для представления и хранения цифр системы;

–экономичность системы, т.е. количество элементов, требуемых для представления и хранения многоразрядных чисел;

– быстродействие вычислительных устройств;

– высокую помехоустойчивость кодирования цифр.

Очевидно, что по первому и последнему критериям двоичная система счисления наиболее выгодна в применении, поскольку для кодирования двух цифр этой системы могут использоваться элементы, принимающие только два устойчивых состояния.

Введем показатель экономичности системы счисления $C = SN$, где N – , сетки, выбранной для представления числа. Максимальное число, которое можно представить в этой сетке $A_{Smax} = SN - 1$. Тогда можно найти длину разрядной сетки: $N = \log_S(A_{Smax} + 1)$. Тогда для любой системы счисления

$C = S\log_S(A_{Smax} + 1)$. Введем относительный показатель экономичности $F = S\log_S(A_{Smax}\_+ 1)/[2\log_2(A_{2max} + 1)]$, позволяющий сравнить любую систему счисления с двоичной. На рисунке 2.1. представлена зависимость величины F от основания системы счисления S.Нижняя точка графика соответствует минимуму функции F, определяемому из условия $dF/dt = 0$, что соответствует значению $S = e$. Следовательно, с точки зрения затрат условного оборудования, наиболее экономичной является система счисления с основанием, равным $e \approx 2{,}72$.

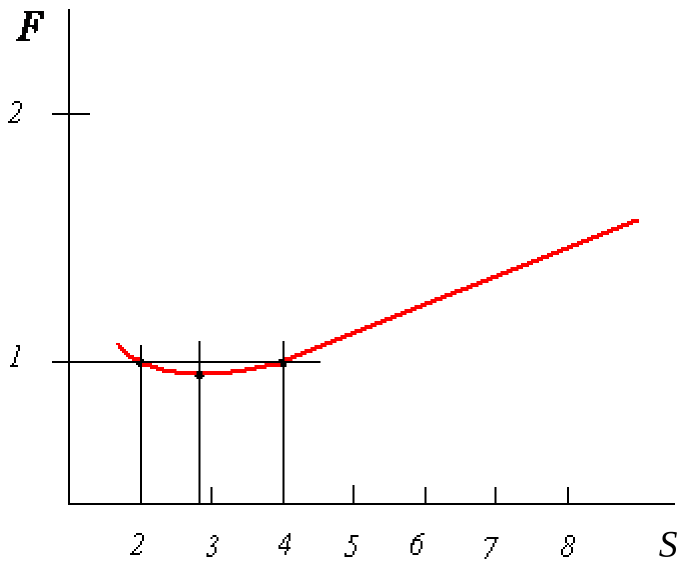


Рис. 2.1. Зависимость относительного показателя экономичности от основания системы счисления

В цифровых вычислительных устройствах наиболее выгодно применять систему счисления с основанием $S = 3$. Однако для хранения произвольной троичной цифры требуется элемент с тремя устойчивыми состояниями, уступающий в плане помехоустойчивости и простоты физической реализации элементу с двумя устойчивыми состояниями.

Таким образом, в повседневной жизни наиболее широко используется десятичная система счисления, а все современные вычислительные средства работают на базе двоичной системы. Использование двоичной системы счисления при построении вычислительных устройств объясняется следующими причинами:

1) в двоичной системе легче различить 2 состояния, чем, например, 10 десятичных;

2) большинство физических величин имеет два состояния: потенциал высокий или низкий; выключатель включен или выключен; ток есть или нет;

3) реализовать цифровую схему, имеющую два состояния значительно легче;

4) сравнительная простота реализации любого арифметического и логического действия, что вытекает из простоты представления чисел комбинацией лишь двух цифр.

При использовании двоичной системы счисления необходимо переводить входную информацию в двоичную систему и двоичную информацию в выходную информацию.

Первым два символа для кодирования информации применил известный философ XVII в. Ф. Бэкон, который использовал символы 0, 1.

Правило (алгоритм) перевода дробных чисел из десятичной системы в систему с основанием *S*

1. Перевод реализуется в виде последовательности шагов умножения исходного числа на первом шаге и получаемых в дальнейшем дробных частей произведения на основание новой системы *S*.

2. Произведение, получаемое на каждом шаге, содержит целую (возможно, нулевую) и дробную части. Целая часть произведения представляет собой очередную цифру дробного числа в системе с основанием *S*.

3. Цифры дробного числа в системе счисления *S* берутся по порядку их получения многократным умножением, то есть старшая цифра вырабатывается на первом шаге.

Процесс последовательных умножений завершается в одном из следующих случаев:

- на очередном шаге в дробной части произведения получен 0 (перевод осуществлен точно);
- при многократном умножении удалось выявить период дроби;
- когда получено необходимое количество цифр дробного числа.

Пример перевода десятичной дроби 0,375 в двоичную систему счисления:

```
 0,375
*    2
  0,750
*     2
  1,50
*    2
  1,0
```

$(0,375)_{10} = (0,011)_2$.

3. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ В ЭВМ

3.1. Классификация данных, используемых в ЭВМ

В отношении данных достаточно широко используется термин «***аппаратная поддержка***». Принято считать, что данные некоторого типа в определенных форматах являются аппаратно-поддерживаемыми в рамках определенной модели компьютера (процессора), если в системе команд процессора имеются команды для обработки данных этого типа в соответствующих форматах.

Например, в базовой ЭВМ аппаратно поддерживаются целые знаковые числа в 16-битном формате.

Классификация данных в виде дерева представлена на рисунке 3.1.

К основным типам нечисловых данных, обладающих аппаратной поддержкой, принято относить логические значения и символьные данные.

Для логических значений характерным свойством является их побитовая обработка, то есть каждый бит логического значения обрабатывается независимо от других битов формата. Как правило, отдельные биты логических значений объединяются в стандартные форматы, которые в терминологии фирмы

Intel имеют следующие наименования:

Byte (**B**) байт – 8 бит;

Word (**W**) слово – 16 бит;

Double Word (**DW**) двойное слово– 32 бита;

Quadro Word (**QW**) учетверенное слово– 64 бита (квадрослово).

Аппаратная поддержка логических значений реализуется на уровне логических команд, таких как:

AND – поразрядная конъюнкция (логическое И);

OR – поразрядная дизъюнкция (логическое ИЛИ);

XOR – исключительное «или» (для двух операндов операция XOR совпадает со сложением по модулю 2);

NOT – инверсия (логическое НЕ).

Символьные данные используются для представления в ЭВМ разнообразной текстовой информации.

В современных компьютерах для представления символьных данных используется код **ASCII** – American Standard Code for Interchange Information (Американский стандартный код для обмена информацией).

Аппаратная поддержка символьных данных в процессорах Intel 80X86 реализуется на уровне специальных команд обработки строк, основными из которых являются:

MOVS – пересылка строк;

CMPS – сравнение строк;

SCAS – сканирование строки.

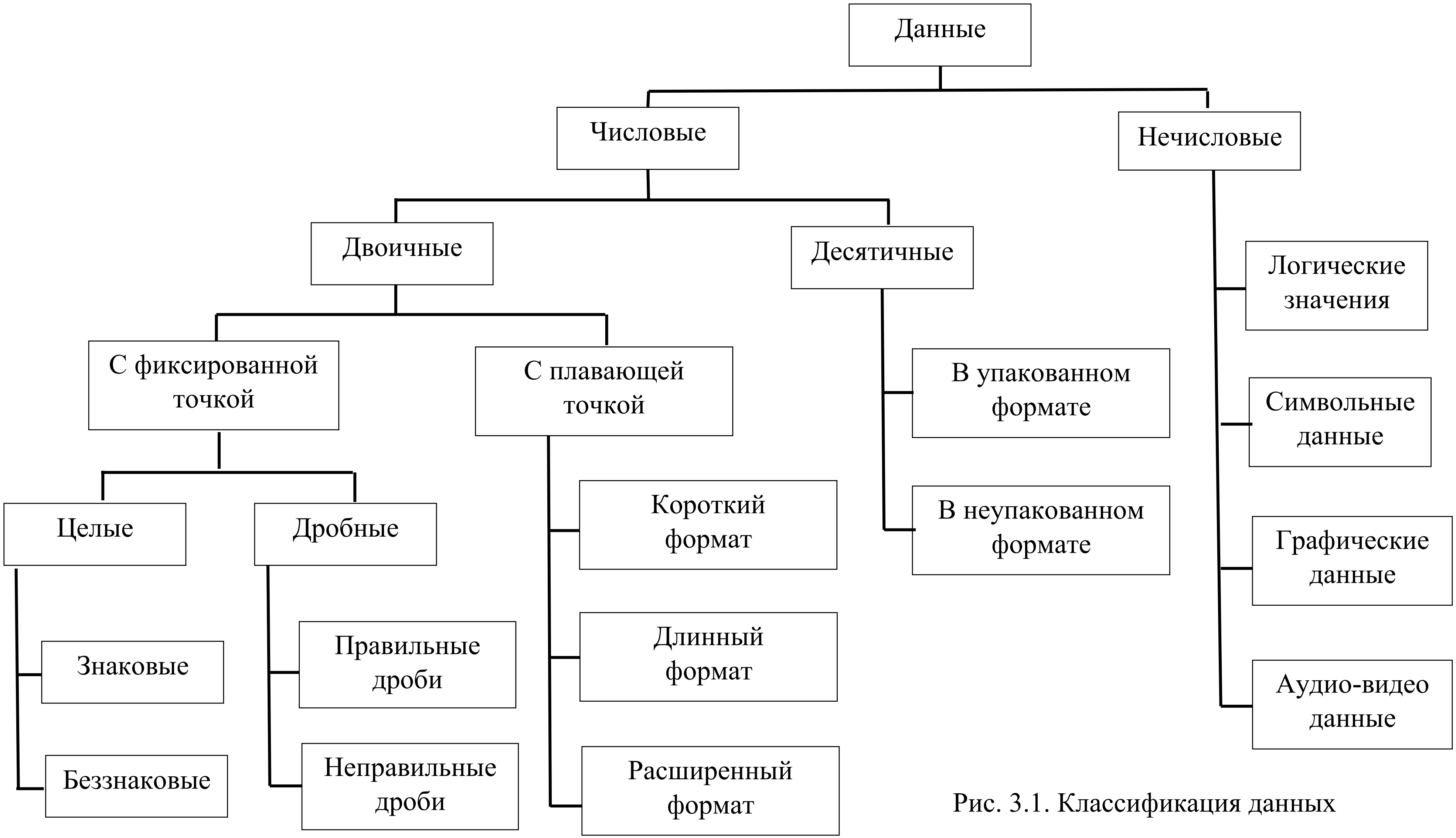


Рис. 3.1. Классификация данных

Каждая команда обработки строк рассчитана на обработку одного элемента строки длиной в байт, слово или двойное слово, однако использование перед этими командами специального префикса **REP** (повторение) позволяет осуществлять обработку строки произвольной длины с заданным числом элементов. Использование команд обработки строк с префиксом **REP** можно рассматривать как аппаратную поддержку элементарной структуры данных типа «строка».

Аппаратная поддержка числовых данных реализуется прежде всего на уровне арифметических команд, таких как:

ADD – сложение;

SUB – вычитание;

MUL, IMUL – умножение;

DIV, IDIV – деление.

3.2. Двоичные числа с фиксированной запятой. Знаковые и беззнаковые числа

Основной особенностью представления знаковых целых чисел является использование дополнительного кода. Дополнительный код позволяет значительно упростить основные арифметические операции (сложение, вычитание).

Дополнительный код числа:

$$[X]_{\text{доп.}} = \begin{cases} X, \text{ при } X \geq 0; \\ 2^{n} - |X|, \text{ при } X < 0. \end{cases}$$

Пример.

$n = 5$, $X = -13$.

$[X]_{\text{пр}} = 1.1101$

$[X]_{\text{доп}} = 1.0000 - \text{M} = 1.0011$

Основными форматами, в которых представляются целые числа, являются:

1 байт;

2 байта – слово;

4 байта – двойное слово;

8 байтов – учетверенное слово.

Диапазон представления знаковых целых чисел

$$\mathbf{-2^{n} \leq A_{зн} \leq 2^{n-1}}$$

$$1.\underbrace{00\ldots00}_{n-1} \qquad 0.\underbrace{11\ldots11}_{n-1}$$

Отрицательное число по модулю на единицу больше положительного.

Для байтного формата: $n = 8$

$$-128 \le A_{зн} \le 127.$$

Диапазон представления беззнаковых целых чисел

$$\mathbf{0 \le A_{бзн} \le 2^{n} - 1}$$

$$\underbrace{\mathbf{00\ldots0}}_{\boldsymbol{n}} \qquad \underbrace{\mathbf{11\ldots1}}_{\boldsymbol{n}}$$

Для байтного формата: $n = 8$

$$0 \le A_{зн} \le 255.$$

Диапазон представления дробных чисел

Для правильной n-разрядной дроби:

$$\mathbf{2^{-n} \le A_{др} \le 1 - 2^{-n}.}$$

Неправильная дробь в целой части содержит обязательную и единственную единицу.

$$\mathbf{1 \le A_{др.нпр.} \le 2 - 2^{-(n+1)}.}$$

3.3. Числа с плавающей запятой

Множество целых чисел бесконечно, но всегда можно подобрать число бит, необходимое для представления любого целого числа, необходимого для решения поставленной задачи. Множество действительных чисел не только бесконечно, но еще и непрерывно, поэтому, независимо от количества используемых бит, мы неизбежно столкнемся с числами, которые не имеют точного представления. Числа с плавающей запятой — один из возможных способов представления действительных чисел, который является компромиссом между точностью и диапазоном принимаемых значений.

Действительные числа в обобщенном виде представляются следующим образом:

$$A = (-1)^{signA} \cdot M_A \cdot S^{P_A},$$

где $\boldsymbol{signA}$ – знак числа $\boldsymbol{A}$ (0 – для положительных, 1 – для отрицательных),

$\boldsymbol{M_A}$ – мантисса числа $\boldsymbol{A}$,

$\boldsymbol{S}$ – основание порядка,

$\boldsymbol{P_A}$ – порядок числа $\boldsymbol{A}$.

Основными особенностями представления чисел с плавающей запятой (плавающая точка) в современных ЭВМ являются:

1. В качестве основания порядка в современных ЭВМ используются $\boldsymbol{S}$=2 (двоичное представление мантиссы) и $\boldsymbol{S}$=16 (шестнадцатеричное представление мантиссы). Основание 2 используется персональных компьютерах, а 16 – в компьютерах типа Мэйнфрейм (англ. Mainframe). Порядок числа представляет собой целое число со знаком.

2. Преимущественное использование так называемых нормализованных чисел.

Число с плавающей запятой называется ***нормализованным***, если старшая цифра его мантиссы является значащей (не ноль).

3. Порядок числа представляется не как целое число со знаком в явном виде, а в виде беззнакового числа, называемого смещенным порядком или характеристикой. При этом характеристика отличается от порядка на некоторую фиксированную для данного формата величину, называемую смещением (или смещением порядка)

$$\boldsymbol{X_A = P_A + d,}$$

где $\boldsymbol{X_A}$ – характеристика числа $\boldsymbol{A}$, $\boldsymbol{d}$ – смещение порядка.

Существует 2 подхода к выбору величины смещения:

1) Величина смещения равна весу старшего разряда смещенного порядка (характеристики).

2) Величина смещения равна весу старшего разряда смещения порядка, уменьшенного на 1.

Первый подход используется в основном в компьютерах типа Мэйнфрейм, второй в ПК.

В 60-е и 70-е годы не было единого стандарта представления чисел с плавающей запятой, способов округления, арифметических операций. В результате программы были не переносимы на разные ЭВМ. Но еще большей проблемой было то, что у разных компьютеров были свои «странности» и их нужно было знать и учитывать в программе. Например, разница двух не равных чисел возвращала ноль. В результате выражения «X=Y» и «X-Y=0» вступали в противоречие.

Инициатива создать единый стандарт для представления чисел с плавающей запятой совпала с попытками в 1976 году компанией Intel разработать «лучшую» арифметику для новых сопроцессоров к 80386 и i432.

Кроме Intel свои предложения представили компании DEC, National Superconductor, Zilog, Motorola. Производители Мейнфреймов Cray и IBM наблюдали со стороны.

Авторами предложенной Intel спецификации стали Уильям Кэхэн, Джероми Кунен и Гарольд Стоун и их предложение сразу прозвали «K-C-S». Практически сразу же были отброшены все предложения, кроме двух: VAX от DEC и «K-C-S» от Intel. Спецификация VAX была значительно проще, уже была реализована в компьютерах PDP-11, и было понятно, как на ней получить максимальную производительность. С другой стороны, в «K-C-S» содержалось много полезной функциональности, такой как «специальные» и «денормализованные» числа.

В «K-C-S» все арифметические алгоритмы заданы строго и требуется, чтобы в реализации результат с ними совпадал. Это позволяет выводить строгие выкладки в рамках этой спецификации. Если раньше математик решал задачу численными методами и доказывал свойства решения, не было никакой гарантии, что эти свойства сохранятся в программе. Строгость арифметики «K-C-S» сделала возможным доказательство теорем, опираясь на арифметику с плавающей запятой.

3.4. Стандарт IEEE 754

Данный стандарт разработан ассоциацией IEEE (*Institute of Electrical and Electronics Engineers*) и используется для представления действительных чисел (чисел с плавающей точкой) в двоичном коде. Это самый используемый стандарт для вычислений с плавающей точкой, он используется многими микропроцессорами и логическими устройствами, а также программными средствами.

Полное название стандарта в ассоциации IEEE: ***IEEE Standard for Binary Floating-Point Arithmetic*** (***IEEE стандарт для двоичной арифметики с плавающей точкой*.** В августе 2008 года ассоциация IEEE выпустила стандарт ***ANSI/IEEE Std 754****-2008* (***ANSI:*** *American National Standards Institute* – Американский национальный институт стандартов).

Стандарт IEEE 754 определяет:

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

Способы округления чисел по стандарту IEEE 754:

1. Округление к ближайшему целому.
2. Округление к нулю.
3. Округление к $+\infty$.
4. Округление к $-\infty$.

Таблица. Примеры округления чисел до десятых

исходное число	к ближайшему целому	к нулю	к $+\infty$	к $-\infty$
1,33	1,3	1,3	1,4	1,3
-1,33	-1,3	-1,3	-1,3	-1,4
1,37	1,4	1,3	1,4	1,3
-1,37	-1,4	-1,3	-1,3	-1,4
1,35	1,4	1,3	1,4	1,3
-1,35	-1,4	-1,3	-1,3	-1,4

В IEEE754 число «0» представляется значением со смещенным порядком, равным $X = X_{\min}$ -1 (для короткого формата это -127) и нулевой мантиссой. Введение нуля как самостоятельного числа (т.к. в нормализованном представлении нельзя представить ноль) позволило избежать многих странностей в арифметике. И хоть операции с нулем нужно обрабатывать отдельно, обычно они выполняются быстрее, чем с обычными числами.

Также в IEEE754 предусмотрено представление для специальных чисел, работа с которыми вызывает исключение. К таким числам относится бесконечность (±∞) и неопределенность (NaN). Эти числа позволяет вернуть адекватное значение при переполнении. Бесконечности представлены как числа с порядком $X = X_{\max} + 1$ и нулевой мантиссой. Получить бесконечность можно при переполнении и при делении ненулевого числа на ноль. Бесконечность при делении разработчики определили исходя из существования пределов, когда делимое и делитель стремятся к какому-то числу. Соответственно, $c/0 = \pm\infty$ (например, $3/0 = +\infty$, а $-3/0 = -\infty$), так как если делимое стремиться к константе, а делитель к нулю, предел равен бесконечности. При 0/0 предел не существует, поэтому результатом будет неопределенность.

Неопределенность или NaN (от *not a number*) – это представление, придуманное для того, чтобы арифметическая операция могла всегда вернуть какое-то не бессмысленное значение. В IEEE754 NaN представлен как число, в котором $X = X_{\max} + 1$, а мантисса не нулевая. Любая операция с NaN возвращает NaN. При желании в мантиссу можно записывать информацию, которую программа сможет интерпретировать. Стандартом это не оговорено и мантисса чаще всего игнорируется.

NaN можно получить одним из следующих способов:

- $\infty+(-\infty)$;
- $0 \times \infty$;
- $0/0$, ∞/∞;
- sqrt(x), где $x < 0$.

В представлении чисел с плавающей запятой существует два нуля, которые отличаются только знаком. Так, $3\cdot(+0) = +0$, а $3\cdot(-0) = -0$. Но при сравнении $+0=-0$. В стандарте знак сохранили умышленно, чтобы выражения, которые в результате переполнения или потери значимости превращаются в бесконечность или в ноль, при умножении и делении все же могли представить максимально корректный результат. Например, если бы у нуля не было знака, выражение $1/(1/x) = x$ не выполнялось бы верно при $x = \pm\infty$, так как $1/\infty$ и $1/-\infty$ равны.

Еще один пример:

$(+\infty/0) + \infty = +\infty$, тогда как $(+\infty/-0) +\infty =$ NaN.

Чем бесконечность в данном случае лучше, чем NaN? Тем, что, если в арифметическом выражении появился NaN, результатом всего выражения всегда будет NaN. Если же в выражении встретилась бесконечность, то результатом может быть ноль, бесконечность или обычное число с плавающей запятой. Например, $1/\infty = 0$.

В арифметике с плавающей запятой ассоциативный закон $(a*b)*c = a*(b*c)$ не выполняется для любых арифметических операций. Например,

$$(10^{20} + 1) - 10^{20} = 0 \neq (10^{20} - 10^{20}) + 1 = 1.$$

3.5. Представление чисел с плавающей запятой

Рассмотрим способы представления чисел с плавающей запятой (точкой) в различных форматах. Форматы, используемые в компьютерах типа Мэйнфрейм – (***формат Ф1***); формат, который раньше использовался в МиниЭВМ семейства VAX – (***формат Ф2***) (представление чисел в этом формате логически понятно, и он приводился исключительно в учебных целях); формат, используемый в персональных компьютерах (стандарт IEEE-754) – (***формат Ф3***).

В целях разумного компромисса между точностью представления и скоростью обработки данных в ЭВМ каждого типа используется несколько форматов:

• *короткий формат* (формат одинарной точности) - 32 разряда;

• *длинный формат* (формат двойной точности) – 64 разряда;

• *расширенный формат* (формат расширенной точности):
для MF и Мини ЭВМ – 128 разрядов;
для IEEE – 80 разрядов.

В классах MF и Мини ЭВМ происходит увеличение только разрядности мантиссы, в стандарте IEEE – увеличение мантиссы и порядка.

Диапазон представления чисел с плавающей запятой

$$\boldsymbol{M_{A\min}\, S^{P A\min} \le |\, A_{\text{п.т. норм}}\, | \le M_{A\max}\, S^{P\, A\max}}$$

Для классов:

1. MF (*Ф1*)

$S = 16,\ d = 64.$

±	характеристика	мантисса
0	1 7	8 31

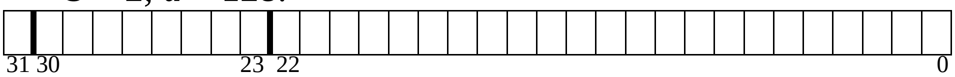
Характеристика: $0 \le X_A \le 127$;

Порядок: $-64 \le P_A \le 63$.

$$1/16 \bullet 16^{-64} \le |\, \boldsymbol{A_{\text{п.т. норм}}}\, | \le 1 \bullet 16^{63}$$

2. Мини ЭВМ (*Ф2*)

$S = 2;\ d = 128.$



Характеристика: $0 \le X_A \le 225$;

Порядок: $-128 \le P_A \le 127$.

$$1/2 \bullet 2^{-128} \le |\, \boldsymbol{A_{\text{п.т. норм}}}\, | \le 1 \bullet 2^{127}$$

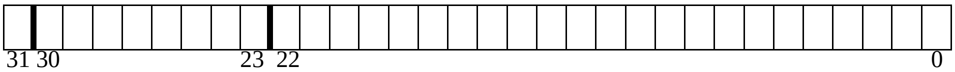
3. Стандарт IEEE (*Ф3*)

$S = 2;$

$d = 16383$ – для расширенного формата,

$d = 1023$ – для длинного формата,

$d = 127$ – для короткого формата.



Скрытая единица имеется только в коротком и длинном формате, в расширенном формате она представляется в явном виде.

Поэтому диапазон представления чисел в формате стандарта IEEE:

- короткий формат:

$$2^{-126} \leq |\boldsymbol{A}_{\text{п.т. норм}}| < 2^{128};$$

- длинный формат

$$10^{-308} \leq |\boldsymbol{A}_{\text{п.т. норм}}| < 10^{308};$$

- расширенный формат

$$10^{-4932} \leq |\boldsymbol{A}_{\text{п.т. норм}}| < 10^{4932}.$$

Точность представления чисел

Каждая десятичная представляется в виде бесконечной двоичной дроби, что в условиях ограниченного формата приводит к возникновению погрешности.

Максимальная абсолютная погрешность имеет место в том случае, когда все отбрасываемые разряды равны единице.

$$0\,.\,\underline{10\,\ldots\ldots\,110}_{\,n}\,|\,\underset{n+1}{1}11\,\ldots\ldots\underset{\infty}{1}$$

$$\Delta A_{\text{др.мах}} = \sum_{i=n+1}^{\infty} 2^{-i} = 2^{-n},$$

где n - разрядность числа.

Относительная погрешность

$$\boldsymbol{\delta}\ \boldsymbol{A}_{\text{др.min}} = \left|\frac{\Delta A\ \text{др max}}{A\text{др max}}\right| = \left|\frac{2^{-n}}{1-2^{-n}}\right| \approx 2^{-n}$$

$$\boldsymbol{\delta}\ \boldsymbol{A}_{\text{др.max}} = \left|\frac{\Delta A\ \text{др max}}{A\text{др min}}\right| = \left|\frac{2^{-n}}{2^{-n}}\right| \approx 1$$

Погрешность представления чисел с плавающей точкой определяется погрешностью мантиссы как дробного числа.

$$|A_{n.m.}| = |M_A| \cdot S^{P_A}$$

$$\delta\ \boldsymbol{A}_{\text{п.т.}} = \left|\frac{\Delta A\,n.m.}{A\,n.m.}\right| = \left|\frac{\Delta M_A \cdot S^{P_A}}{M_A \cdot S^{P_A}}\right| = \left|\frac{\Delta M_A}{M_A}\right| = \left|\frac{2^{-n}}{1/S}\right| = S \cdot 2^{-n}$$

Формула справедлива для правильных и неправильных дробей.

Точность представления чисел в различных форматах

1. MF (***Ф1***): $\delta\ A_{\text{п.т.}} = 16 \cdot 2^{-24} = 2^{-20} = 10^{-6}$;
2. Мини ЭВМ (***Ф2***): $\delta\ A_{\text{п.т}}. = 2 \cdot 2^{-24} = 2^{-23} = 10^{-7}$;
3. IEEE (***Ф3***): $\delta\ A_{\text{п.т}}. = 10^{-7}$.

3.6. ЗАДАНИЕ 1
ПРЕДСТАВЛЕНИЕ ЧИСЕЛ С ФИКСИРОВАННОЙ И ПЛАВАЮЩЕЙ ЗАПЯТОЙ В РАЗЛИЧНЫХ ФОРМАТАХ

1. Заданное число ***A*** представить в виде двоично-кодированного десятичного числа:

а) в упакованном формате (BCD);

б) в неупакованном формате (ASCII).

2. Заданное число ***A*** и ***–A*** представить в форме с фиксированной запятой.

3. Заданные числа ***A*** и ***B*** представить в форме с плавающей запятой в формате *Ф1*.

4. Заданные числа ***A*** и ***B*** представить в форме с плавающей запятой в формате *Ф2*.

5. Заданные числа ***A*** и ***B*** представить в форме с плавающей запятой в формате *Ф3*.

6. Найти значения чисел ***Y*** и ***Z*** по их заданным шестнадцатеричным представлениям ***R*** и ***S*** в форме с плавающей запятой в формате *Ф1*.

7. Найти значения чисел ***V*** и ***W*** по их заданным шестнадцатеричным представлениям ***R*** и ***S*** в форме с плавающей запятой в формате *Ф2*.

8. Найти значения чисел ***T*** и ***Q*** по их заданным шестнадцатеричным представлениям ***R*** и ***S*** в форме с плавающей запятой в формате *Ф3*.

Замечание. При выполнении п.п. 3–5 задания для дробного числа ***B*** в целях увеличения точности его представления произвести симметричное округление мантиссы.

Варианты задания приведены в табл. 1 (десятичные числа ***A*** и ***B***) и в табл. 2 (шестнадцатеричные числа ***R*** и ***S***) Приложения 1.

1. Представление чисел в виде двоично-кодированного десятичного числа

Десятичные числа представляются в ЭВМ в двоично-кодированной форме, при этом каждая десятичная цифра (или буква – для шестнадцатеричной системы) кодируется с помощью четверки двоичных разрядов (двоичной тетрады).

десятичная цифра	двоичная тетрада
0	0000
1	0001
2	0010
3	0011
4	0100

десятичная цифра	двоичная тетрада
5	0101
6	0110
7	0111
8	1000
9	1001

Десятичные числа представляются в форме с использованием либо упакованного (PACK), либо неупакованной (UNPACK) формата.

В упакованном формате в каждом байте числа кодируются две цифры, в неупакованном – одна.

Для кодирования десятичных цифр используется в основном естественный двоичный код, обычно называемый 8421.

В этом коде:

0 – 0000

1 – 0001

…

9 – 1001

Частным случаем неупакованного формата является код ASCII (American Standart Code for Interchange Information), используемый в PC. В этом коде десятичная цифра представляется в младшей тетраде байта, а старшая тетрада принимает стандартное значение 0011.

Упакованный формат обычно называют BCD-форматом (или BCD-числом – Binary Coded Decimal).

В упакованном формате в каждом байте кодируется две десятичные цифры, в неупакованном – одна.

Пример: $\boldsymbol{A}$=395.

а) в упакованном формате

0000.0011	1001.0101
3	9 5

б) В ASCII-формате код цифры помещается в младшую тетраду байта (в младший полубайт). Старшая тетрада байта имеет стандартное значение 0011.

0011.0011	0011.1001	0011.0101
3	9	5

2. Представление чисел с фиксированной запятой

Особенностью представления целых чисел со знаком в форме с фиксированной запятой в ЭВМ является использование прямого кода для положительных чисел и дополнительного кода – для отрицательных чисел.

Пример. $\boldsymbol{A}$ = 250.

2.1. Заданное десятичное число $\boldsymbol{A}$ переводится в двоичную систему счисления:

$(250)_{10} = (11111010)_2$.

Полученное двоичное число размещается в формате таким образом, чтобы его младший разряд совпадал с крайним правым (15-ым) разрядом формата. Старшие разряды формата, включая знаковый (нулевой разряд), заполняются нулями.

0	0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 0
15	1 0

В шестнадцатеричной системе счисления: (250)10 = (FA)16.

$$\boxed{0}\boxed{000\quad 0000\quad \overbrace{1111}^{F}\quad \overbrace{1010}^{A}}$$

15 … 1 0

2.2. Для представления отрицательного числа в дополнительном коде производится инвертирование цифровых разрядов прямого кода (получение обратного кода числа) с добавлением единицы в младший разряд. В знаковый разряд числа заносится единица (знак “–”).

$[–A]_{пр}$ = 1.000 0000 1111 1010 – прямой код,
$[–A]_{об}$ = 1.111 1111 0000 0101 – обратный код,
\+ 1
$[–A]_{доп}$ = 1.111 1111 0000 0110 – дополнительный код.

Для преобразования отрицательных чисел из прямого кода в дополнительный может использоваться следующее правило. Младшие нули прямого кода, включая младшую (крайнюю правую) единицу, сохраняются и в дополнительном коде, а остальные разряды прямого кода заменяются на противоположные (инвертируются) в дополнительном коде. Преобразование применяется только к цифровым разрядам числа, знаковый разряд не меняется. С использованием этого правила:

$$[–A]_{доп} = 1.\underbrace{111\ 1111\ 0000\ 01}_{\text{разряды, изменяемые инвертированием}}\underbrace{10}_{\text{сохраняемые разряды}}$$

Полученное представление числа в дополниительном коде записывается в формате:

1	1 1 1 1 1 1 1 0 0 0 0 0 1 1 0
15	14 … 1 0

3. Представление чисел с плавающей запятой в формате *Ф1*

Для представления числа в формате с плавающей запятой определяется его мантисса и порядок. В связи с тем, что в формате *Ф1* используется шестнадцатеричное представление порядка, для этой цели удобнее использовать число, представленное в шестнадцатеричной системе счисления. Для определения мантиссы и порядка производится перемещение запятой в шестнадцатеричном числе влево или вправо таким образом, чтобы она установилась перед старшей значащей цифрой. Полученное дробное число представляет мантиссу числа с плавающей запятой. Модуль порядка определяется количеством шестнадцатеричных цифр, на которое была перенесена запятая. Знак порядка определяется направлением, в котором переносилась запятая. При перенесении запятой влево порядок положителен, вправо – отрицателен.

Такой способ получения мантиссы и порядка дает нормализованное число, у которого старшая цифра мантиссы значащая.

Если в исходном числе запятая находится перед старшей значащей цифрой, то порядок этого числа равен нулю.

По значению порядка определяется характеристика (смещенный порядок) числа путем сложения порядка со смещением (для формата *Ф1* величина смещения равна 64), после чего двоичные значения знака, характеристики и мантиссы числа записываются в формат.

3.1. Для определения мантиссы и порядка числа ***A*** запятая переносится влево на две шестнадцатеричные цифры:

$$\boldsymbol{A}=250 \qquad A = (\mathrm{FA})_{16} = \underbrace{(0,\mathrm{FA})_{16}}_{\text{мантисса}} \times 16^{\overset{2}{\text{порядок}}}$$

Характеристика числа ***A***:

$X_A = P_A + 64 = (66)_{10} = (1000010)_2.$

Для получения двоичного значения характеристики, соответствующей положительному или нулевому порядку, в ее старший разряд записывается единица (вес этого разряда характеристики равен величине смещения – 64), а в младших разрядах характеристики представляется величина порядка

$$x_A = (1\,0\,0\,0\,0\,1\,0).$$

$$\underbrace{64}_{\text{смещение}} + \underbrace{2}_{\text{порядок}} = \underbrace{66}_{\text{характеристика}}$$

При записи числа в формате *Ф*1 шестнадцатеричная мантисса представляется в двоичной системе счисления. Младшие разряды мантиссы заполняются нулями. Представление числа ***A*** в формате *Ф1* имеет вид

		F	A	0	0	0	0
0	100 0010	1111	1010	0000	0000	0000	0000
0	1 7	8					31
знак “+”	характеристика	мантисса					

Старшая тетрада мантиссы нормализованного числа может содержать от одного до трех старших нулей. Например, число

$(1E8,5)_{16} = (0,1E85)_{16} \times 16^3$ представляется в виде

	64	3	1	E	8	5	0	0
0	100	0011	0001	1110	1000	0101	0000	0000
0	1	7	8					31

и содержит в старшей тетраде мантиссы три нуля.

3.2. Число $\boldsymbol{B}$ = 0,0025 переводится в шестнадцатеричную систему счисления. При переводе необходимо получить шесть цифр, не считая старших нулей.

В целях повышения точности представления числа рекомендуется получить еще одну (дополнительную) цифру, по значению которой производится симметричное округление этого числа.

$\boldsymbol{B} = (0,0025)_{10} = (0,00A3D70A3)_{16}$.

Дополнительная цифра числа, равная $(3)_{16} = (0011)_2$, содержит в старшем двоичном разряде ноль и поэтому не изменяет значения предыдущей цифры, равной $(A)_{16}$, при округлении числа.

При наличии старшей единицы в двоичном представлении дополнительной цифры, что соответствует значению, большему или равному $(8)_{16}$, при симметричном округлении предыдущая младшая цифра числа увеличивается на единицу.

Для определения мантиссы и порядка числа $\boldsymbol{B}$ запятая в его шестнадцатеричном представлении переносится вправо на две цифры, что определяет порядок числа, равный (–2):

$\boldsymbol{B} = (0,00A3D70A)_{16} = (0,A3D70A)_{16} \times 16^{-2}$.

Характеристика числа $\boldsymbol{B}$:

$X_B = P_B + 64 = -2 + 64 = 62 = (0111110)_2$.

Для получения двоичного значения характеристики, соответствующей отрицательному порядку, можно использовать следующее правило: в старший разряд характеристики записывается ноль, а в шести ее младших разрядах представляется дополнительный код порядка (дополнение до 64)

000010 – прямой код порядка,

111110 – дополнительный код порядка,

0111110 – характеристика.

Представление числа ***B*** в формате *Ф1* имеет вид

$$\begin{array}{|c|c|cccccc|}
\multicolumn{1}{c}{} & \multicolumn{1}{c}{} & A & 3 & D & 7 & 0 & \multicolumn{1}{c}{A} \\
\hline
0 & 011\ 1110 & \underbrace{1010} & \underbrace{0011} & \underbrace{1101} & \underbrace{0111} & \underbrace{0000} & \underbrace{1010} \\
\hline
\end{array}$$

0 1 7 8 31

4. Представление чисел с плавающей запятой в формате *Ф2*

Для представления чисел в форме с плавающей запятой в формате *Ф2* используется их двоичная запись, так как в этом формате основание порядка ***S***=2. Для определения мантиссы и порядка запятая переносится влево или вправо в двоичном числе до установления перед старшей единицей. Модуль порядка определяется количеством двоичных цифр (разрядов), на которое переносится запятая.

Характеристика (смещенный порядок) определяется путем сложения порядка со смещением, величина которого в формате *Ф2* равна 128 (в отличие от формата *Ф1* в формате *Ф2* под характеристику отводится 8 двоичных разрядов формата).

При записи числа необходимо учитывать, что во формате *Ф2* используются **только** нормализованные числа и, так как нормализация осуществляется с точностью до двоичной цифры, старший разряд мантиссы всегда равен единице, в связи с чем он в разрядной сетке не представляется (так называемый скрытый разряд). Кроме того, в формате *Ф2* принята нумерация разрядов в сетке справа налево (от младшего разряда к старшему).

4.1. Определение мантиссы и порядка числа ***A***:

$$\boldsymbol{A} = (250)_{10} = (FA)_{16} = (11111010)_2 = (0,\underbrace{1111101}_{\text{мантисса}})_2 \times 2^{\overset{\text{порядок}}{8}}.$$

Характеристика числа ***A***:

$$X_A = P_A + 128 = 136 = (\overset{128}{1}0\underbrace{001000}_{8})_2.$$

$$\underbrace{128}_{\text{смещение}} + \underbrace{8}_{\text{порядок}} = \underbrace{136}_{\text{характеристик}}$$

Число ***A*** представляется в формате *Ф2* в виде

0	10001000	11110100000000000000000
31	30 … 23	22 … 0
знак	характеристика	мантисса

4.2. Для определения мантиссы и порядка числа ***B*** запятая в его двоичном представлении переносится вправо на 8 двоичных разрядов, что определяет порядок числа, равный (–8).

$$\boldsymbol{B} = (0{,}0025)_{10} = (0{,}00A3D70A)_{16} =$$

$$= (0{,}\underbrace{0000}_{0}\ \underbrace{0000}_{0}\ \underbrace{1010}_{A}\ \underbrace{0011}_{3}\ \underbrace{1101}_{D}\ \underbrace{0111}_{7}\ \underbrace{0000}_{0}\ \underbrace{1010}_{A})_2 =$$

$$= \underbrace{(0{,}10100011110101110000101)_2}_{\text{мантисса}} \times 2^{\overset{\text{порядок}}{-8}}.$$

Характеристика числа ***B***:

$$X_B = P_B + 128 = 120 = (\underbrace{01111000}_{\text{дополнительный код порядка}})_2.$$

Представление числа ***B*** в формате *Ф2* имеет вид

0	01111000	01000111101011100001010
31	**30** … **23**	**22** … **0**
знак	характеристика	мантисса

Для данного примера дробное число *B* представлено в форматах *Ф1* и *Ф2* с одинаковой точностью. Это объясняется наличием единицы в старшем разряде мантиссы формата *Ф1*. Если же в старших разрядах мантиссы содержится хотя бы один ноль, то точность представления числа в формате *Ф2* будет больше за счет использования большего числа значащих цифр в мантиссе.

Например, шестнадцатеричная мантисса $(0{,}1A90BC7)_{16}$ будет представлена в формате *Ф1* в виде

$$\overset{8}{}\ \underbrace{0001}_{1}\ \underbrace{1010}_{A}\ \underbrace{1001}_{9}\ \underbrace{0000}_{0}\ \underbrace{1011}_{B}\ \underbrace{1100}_{C}\ \overset{31}{}\ \boxed{0}$$

разряд, по которому производится округление

а в формате *Ф2* - в виде

1	1 0 1 0 1 0 0 1 0 0 0 0 1 0 1 1 1 1 0 0 1 0 0	1
	22 … 0	
скрытый разряд		разряд, по которому производится округление

В данном случае в формате *Ф2* в представлении мантиссы используется на три разряда больше, чем в *Ф1*. Кроме того, за счет дополнительного разряда мантиссы, равного единице, производится добавление единицы к младшему разряду, в результате чего получается представление числа с избытком, в то время как в формате *Ф1* оно представлено с недостатком.

5. Представление чисел с плавающей запятой в формате *Ф3*

Представление чисел в формате *Ф3* во многом аналогично их представлению в формате *Ф2*. Основными отличиями являются:

1) величина смещения равна 127 (в формате *Ф2* – 128);

2) старшая единица мантиссы нормализованного числа является единицей целой части мантиссы, т.е. запятая в мантиссе фиксируется после старшей единицы (в формате *Ф2* запятая в мантиссе фиксируется перед старшей единицей).

5.1. Определение мантиссы и порядка числа ***A***:

$$\boldsymbol{A} = (250)_{10} = (FA)_{16} = (11111010)_2 = (\underbrace{1,111101}_{\text{мантисса}})_2 \times 2^{\overbrace{7}^{\text{порядок}}}.$$

Смещенный порядок числа ***A***:

$$X_A = P_A + 127 = 134 = (10000110)_2.$$

Число ***A*** представляется в формате *Ф3* в виде

0	10000110	11110100000000000000000
31	30 … 23	22 … 0
знак	смещенный порядок	мантисса

Следует отметить, что:

а) в отличие от представления чисел в форматах *Ф1* и *Ф2* в *Ф3* не принято называть смещенный порядок характеристикой;

б) по аналогии с представлением чисел в формате *Ф2* в *Ф3* используется скрытый разряд (единица целой части мантиссы в формате не представляется);

в) представление числа в формате *Ф3* отличается от представления в *Ф2* только значением смещенного порядка (его величина уменьшается на 2).

5.2. Определение мантиссы и порядка числа ***B***:

$$\boldsymbol{B} = (0{,}0025)_{10} = (0{,}00A3D70A)_{16} =$$

$$= (0{,}\underbrace{0000}_{0}\ \underbrace{0000}_{0}\ \underbrace{1010}_{A}\ \underbrace{0011}_{3}\ \underbrace{1101}_{D}\ \underbrace{0111}_{7}\ \underbrace{0000}_{0}\ \underbrace{1010}_{A})_2 =$$

$$= \underbrace{(1{,}0100011110101110000101)_2}_{\text{мантисса}} \times 2^{\underset{\substack{\uparrow \\ \text{порядок}}}{-9}}.$$

Смещенный порядок числа ***B***:

$$X_B = P_B + 127 = 118 = (01110110)_2.$$

Для чисел с отрицательным порядком значение смещенного порядка может быть получено по следующему правилу: старший разряд смещенного порядка равен нулю, а в остальных разрядах представляется обратный код порядка:

0001001 – прямой код порядка,
1110110 – обратный код порядка,
01110110 – смещенный порядок.

Представление числа ***B*** в формате *Ф3* имеет вид

0	01110110	01000111101011100001010
31	30 … 23	22 … 0

6. Определение значения числа с плавающей запятой по его представлению в формате *Ф1*

R = C318FC00, ***S*** = 3E600000.

6.1. Для определения значения числа ***Y*** производится наложение его шестнадцатеричного представления ***R*** на разрядную сетку формата *Ф1*:

C	3	1	8	F	C	0	0
1 100	0011	0001	1000	1111	1100	0000	0000

знак	характеристика	мантисса
1	100 0011	0001 1000 1111 1100 0000 0000
0	1 … 7	8 … 31

Из этого представления видно, что число ***Y*** – отрицательное (в знаковом разряде числа – единица).

Определим порядок числа ***Y*** по его характеристике:

$X_Y = 67 = \underbrace{64}_{\text{смещение}} + \underbrace{3}_{\text{порядок}},$

$P_Y = X_Y - 64 = 3.$

Представим число $\boldsymbol{Y}$ с помощью мантиссы и порядка:

$\boldsymbol{Y} = -\,(0{,}18FC)_{16} \times 16^3.$

Получили представление числа $\boldsymbol{Y}$ в нормальной (полулогарифмической) форме. Для приведения числа $\boldsymbol{Y}$ к естественной форме необходимо перенести запятую в мантиссе на количество шестнадцатеричных цифр, равное модулю порядка, вправо – при положительном или влево – при отрицательном порядке. В данном случае запятая переносится вправо:

$\boldsymbol{Y} = -\,(18F{,}C)_{16}.$

Переведем число $\boldsymbol{Y}$ из шестнадцатеричной в десятичную систему счисления с использованием весов разрядов:

$\boldsymbol{Y} = -\,(1 \times 16^2 + 8 \times 16^1 + 15 \times 16^0 + 12 \times 16^{-1}) =$

$= -\,(256 + 128 + 15 + 0{,}75) = -399{,}75.$

6.2. Для определения значения числа $\boldsymbol{Z}$ производится наложение его шестнадцатеричного представления $\boldsymbol{S}$ на разрядную сетку:

3		E	6	0	0	0	0	0
0	0 1 1	1 1 1 0	0 1 1 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
0	1	7	8					31

Порядок числа $\boldsymbol{Z}$:

$P_Z = X_Z - 64 = 62 - 64 = -2.$

Значение числа $\boldsymbol{Z}$:

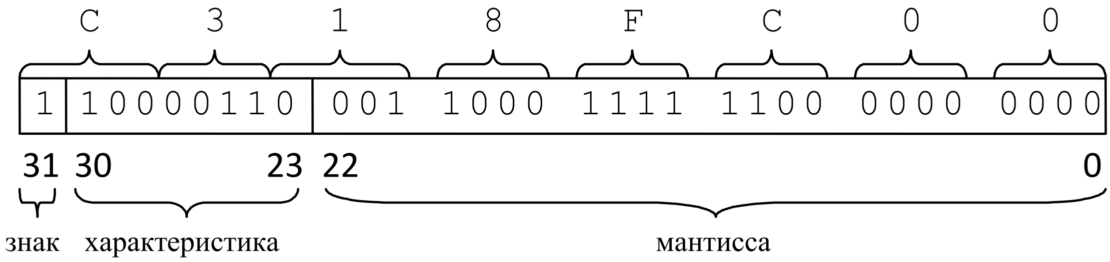
$\boldsymbol{Z} = (0{,}6)_{16} \times 16^{-2} = (0{,}006)_{16} = 6/16^3 = 6/2^{12} = 3/2^{11} =$

$= (3/2) \times (1/2^{10}) = (3/2) \times (1/1024) \approx 1{,}5 \times 10^{-3}.$

При переводе дробных чисел из двоичной системы счисления в десятичную можно считать: $2^{10} \approx 10^3$.

7. Определение значения числа с плавающей запятой по его представлению в формате *Ф2*

7.1. Представление числа $\boldsymbol{V}$ в формате *Ф2* имеет вид



Порядок числа ***V***:

$P_V = X_V - 128 = 134 - 128 = 6.$

Значение числа ***V*** в нормальной форме:

$$\boldsymbol{V} = -(0{,}\underbrace{10011000111111}_{\text{мантисса}})_2 \times 2^{6}.$$

скрытый разряд — порядок

При определении двоичного значения мантиссы производится восстановление ее скрытого старшего разряда, равного 1.

Для приведения числа ***V*** к естественной форме запятая в его мантиссе переносятся вправо на 6 двоичных разрядов:

$\boldsymbol{V} = -(100110{,}00111111)_2.$

Перевод числа ***V*** из двоичной системы в десятичную:

а) целая часть:
$(100110)_2 = 2^5 + 2^2 + 2^1 = 32 + 4 + 2 = 38;$

б) дробная часть:
первый способ перевода:

$(0{,}00111111)_2 = 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} =$

$= 1/8 + 1/16 + 1/32 + 1/64 + 1/128 + 1/256 = 63/256 \approx 0{,}246;$

второй способ перевода:

$(0{,}00111111)_2 = (111111)_2 \times 2^{-8} = 63 \times (1/256) \approx 0{,}246;$

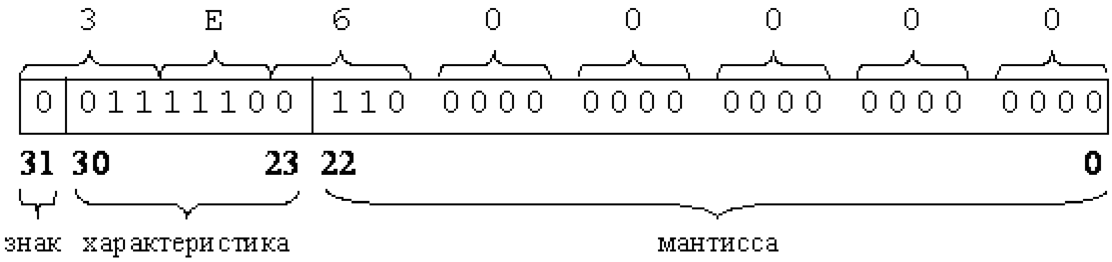
третий способ перевода:

$(0{,}0011111)_2 = (0{,}01)_2 - (0{,}00000001)_2 = 1/4 - 1/256 \approx$

$\approx 0{,}25 - 0{,}004 = 0{,}246.$

Значение числа ***V***:
$\boldsymbol{V} \approx -38{,}246.$

7.2. Представление числа $\boldsymbol{W}$ в формате:



Порядок числа $\boldsymbol{W}$:

$P_W = X_W - 128 = 124 - 128 = -4.$

Число $\boldsymbol{W}$ в нормальной форме:

$\boldsymbol{W} = (0{,}111)_2 \times 2^{-4}.$

Число $\boldsymbol{W}$ в естественной форме получается переносом запятой в мантиссе влево на четыре двоичных разряда:

$\boldsymbol{W} = (0{,}0000111)_2.$

Значение числа $\boldsymbol{W}$:

$\boldsymbol{W} = (0{,}0000111)_2 = (111)_2 \times 2^{-7} = 7 / 128 \approx 0{,}0547.$

8. Определение значения числа с плавающей запятой по его представлению в формате *Ф3*

8.1. Представление числа $\boldsymbol{T}$ в формате *Ф3* имеет тот же вид, что и для числа $\boldsymbol{V}$ в формате *Ф2* (п.7.1).

Порядок числа $\boldsymbol{T}$:

$\boldsymbol{P_T = X_T} - 127 = 134 - 127 = 7.$

Значение числа $\boldsymbol{T}$ в двоичной системе счисления:

$$\mathrm{T} = -\,(\underbrace{1,\,0011000111111}_{\text{мантисса}})_2 \times 2^{7}.$$

скрытый разряд порядок

Перевод числа $\boldsymbol{T}$ из двоичной системы счисления в десятичную:

целая часть:

$(10011000)_2 = 2^7 + 2^4 + 2^3 = 128 + 16 + 8 = 152.$

дробная часть:

$(0{,}111111)_2 = 1 - (0{,}000001)_2 = 1 - 1 / 64 \approx 0{,}984.$

Значение числа $\boldsymbol{T}$:

$\boldsymbol{T} \approx -152{,}984.$

По сравнению со значением числа **V**, *имеющего то же самое представление в формате Ф2, число* **T** *в формате Ф3 в четыре раза больше за счет* *большего на единицу значения порядка и за счет использования целой единицы в мантиссе.*

8.2. Представление числа $\boldsymbol{Q}$ в формате *Ф3* имеет тот же вид, что и для числа в *Ф2* (п.7.2.).

Порядок числа $\boldsymbol{Q}$:

$\boldsymbol{P_Q} = \boldsymbol{X_Q} - 127 = 124 - 127 = -3.$
Значение числа $\boldsymbol{Q}$:
$\boldsymbol{Q} = (1{,}11)_2 \times 2^{-3} = (0{,}00111)_2 = (111)_2 \times 2^{-5} = 7 / 32 \approx 0{,}219.$

4. СЛОЖЕНИЕ ЦЕЛЫХ ЧИСЕЛ

4.1. Регистр флагов

Регистр флагов — регистр процессора (FLAGS), отражающий текущее состояние процессора. Регистр флагов содержит группу флагов состояния (арифметические флаги) и флаги управления.

Арифметические флаги формируются арифметическими командами (ADD, SUB, MUL, DIV) и являются признаками их результата.

Флаги управления оказывают влияние на процесс выполнения программ.

К арифметическим флагам относятся:

• **(CF) Carry Flag –** флаг переноса, в нем фиксируется перенос из старшего разряда при сложении и заем в старший разряд при вычитании. При умножении CF показывает возможность (= 0) и невозможность (= 1) представления произведения в том же формате, что и операндов.

• **(PF) Parity Flag** – флаг паритета (четности). Устанавливается в единицу при наличии четного числа единиц в младшем байте результата, в противном случае - сбрасывается. PF используется в качестве аппаратной поддержки контроля по четности.

• **(AF) Auxiliary Carry Flag** – флаг вспомогательного переноса, в котором фиксируется межтетрадный перенос при сложении и межтетрадный заём при вычитании. Этот флаг используется командами десятичной арифметики.

• **(ZF) Zero Flag –** флаг нуля, устанавливается при нулевом значении результата, в противном случае сбрасывается.

• **(SF) Sign Flag –** флаг знака, в котором копируется старший разряд результата.

• **(OF) Overflow Flag** флаг переполнения. Устанавливается в командах сложения и вычитания, если результат не помещается в формате, при этом и операнды и результат интерпретируются как знаковые числа. Аппаратно он формируется совпадением переносов из двух старших разрядов при сложении и заёмов в два старших разряда при вычитании (если они совпадают, то флаг равен нулю).

Переполнение при сложении чисел возникает только в том случае, если операнды имеют одинаковые знаки. Переполнение фиксируется тремя способами:

• сравнение знаков операндов и суммы: если знак суммы отличается от знаков операндов, то фиксируется переполнение;

• сравнение переносов из двух старших разрядов: если они не совпадают, то фиксируется переполнение;

• использование модифицированного знака (под знак отводится два разряда, второй разряд дублирует знак).

Операции двоичного сложения реализуются поразрядно, начиная с младшего разряда, с учетом возникающих межразрядных переносов согласно следующей таблицы.

a_i	b_i	P_{i-1}	S_i	P_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

P_{i-1} – перенос из предыдущего разряда;
P_i – перенос из i-го разряда;
S_i – сумма i-го разряда.

4.2. ЗАДАНИЕ 2
СЛОЖЕНИЕ ЦЕЛЫХ ЧИСЕЛ

1. Для заданных чисел ***A*** и ***B*** выполнить операцию знакового сложения со всеми комбинациями знаков операндов. Для каждого примера:

a) проставить межразрядные переносы, возникающие при сложении;

b) дать знаковую интерпретацию (Зн) операндов и результатов. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой;

c) дать беззнаковую интерпретацию (БзИ) операндов и результатов, при получении неверного результата пояснить причину его возникновения;

d) показать значения арифметических флагов.

2. Сохранив значение первого операнда ***A***, выбрать такое значение ***B***, чтобы в операции сложения с одинаковыми знаками имел место особый случай переполнения формата. Выполнить два примера, иллюстрирующие эти случаи, для каждого из них проделать пункты a, b, c, d.

3. Сохранив операнд ***B***, подавать такое значение операнда ***A***, чтобы при сложении положительных операндов имело место переполнение формата, а при сложении отрицательных операндов результат был бы корректен. Выполнить два примера, иллюстрирующие этот случай. Для каждого из них проделать пункты a, b, c, d.

Варианты заданий приведены в табл. 3 Приложения 1.

Примеры выполнения задания:

1. $A = 57$, $B = 49$.

$A > 0$, $B > 0$.

									Интерпретации	
									Знаковая	Беззнаковая
+ $A_{пр.}$	0	0	1	1	1	0	0	1	+ 57	+ 57
$B_{пр.}$	0	0	1	1	0	0	0	1	49	49
$C_{пр.}$	0	1	1	0	1	0	1	0	106	106

CF=0, ZF=0, PF=1, AF=0, SF=0, OF=0.

$A > 0$, $B < 0$.

									Интерпретации	
									Знаковая	Беззнаковая
+ $A_{пр.}$	0	0	1	1	1	0	0	1	+ 57	+ 57
$B_{доп.}$	1	1	0	0	1	1	1	1	-49	207
$C_{пр.}$	0	0	0	0	1	0	0	0	8	8?

CF=1, ZF=0, PF=0, AF=0, SF=0, OF=0.

Для беззнаковой интерпретации результат неверен вследствие возникающего переноса из старшего разряда. Вес этого переноса составляет 256.

$A < 0$, $B > 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
$A_{доп}$	1	1	0	0	0	1	1	1	-57	199
+ $B_{пр.}$	0	0	1	1	0	0	0	1	49	49
$C_{доп.}$	1	1	1	1	1	0	0	0		248
$C_{пр.}$	1	0	0	0	1	0	0	0	-8	

CF=0, ZF=0, PF=0, AF=0, SF=1, OF=0.

$A < 0$, $B < 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
$A_{доп.}$	1	1	0	0	0	1	1	1	-57	199
+ $B_{доп.}$	1	1	0	0	1	1	1	1	-49	207
$C_{доп.}$	1	0	0	1	0	1	1	0		150?
$C_{пр.}$	1	1	1	0	1	0	1	0	-106	

CF=1, ZF=0, PF=1, AF=1, SF=1, OF=0.

Для БзИ результат неверен вследствие возникающего переноса из старшего разряда. Вес этого переноса составляет 256.

2. Правило для подбора выглядит следующим образом:
$A + B > 128$, значит $127 > B > 128 - A$, пусть $B = 96$.

	7							0
A	0	0	1	1	1	0	0	1
B	0	1	1	0	0	0	0	0

$A > 0$, $B > 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
$A_{пр.}$	0	0	1	1	1	0	0	1	57	57
+ $B_{пр.}$	0	1	1	0	0	0	0	0	96	96
$C_{доп.}$	1	0	0	1	1	0	0	1		153
$C_{пр.}$	1	1	1	0	0	1	1	1	-103?	

CF=0, ZF=0, PF=1, AF=0, SF=1, OF=1.

Для знакового сложения результат является некорректным вследствие переполнения формата.

$A < 0$, $B < 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
$A_{доп.}$	1	1	0	0	0	1	1	1	-57	199
+ $B_{доп.}$	1	0	1	0	0	0	0	0	-96	160
$C_{пр.}$	0	1	1	0	0	1	1	1	103?	103 ?

CF=1, ZF=0, PF=0, AF=0, SF=0, OF=1.

Для знаковой интерпретации результат неверен вследствие возникающего переполнения, для беззнаковой интерпретации результат неверен вследствие возникающего переноса из старшего разряда

3. Значение числа B фиксируем (B = 49), а значение A подбираем согласно формуле $A + B = 128$, по которой при сложении положительных чисел будет фиксироваться переполнение, а при сложении отрицательных этого не будет. Тогда A = 79.

A	0	1	0	0	1	1	1	1
	7							0

B	0	0	1	1	0	0	0	1
	7							0

$A > 0$, $B > 0$.

										Интерпретации	
										Знаковая	Беззнаковая
+	$A_{пр.}$	0	1	0	0	1	1	1	1	+ 79	+ 79
	$B_{пр.}$	0	0	1	1	0	0	0	1	49	49
	$C_{доп.}$	1	0	0	0	0	0	0	0		128
	$C_{пр.}$	1	0	0	0	0	0	0	0	-128?	

CF=0, ZF=0, PF=0, AF=1, SF=1, OF=1.

Для знаковой интерпретации результат некорректен вследствие возникающего переполнения.

$A < 0$, $B < 0$.

										Интерпретации	
										Знаковая	Беззнаковая
+	$A_{доп.}$	1	1	0	0	1	1	1	1	+ -79	+ 177
	$B_{доп.}$	1	0	1	1	0	0	0	1	-49	207
	$C_{доп.}$	1	0	0	0	0	0	0	0		128?
	$C_{пр.}$	1	0	0	0	0	0	0	0	-128	

CF=1, ZF=0, PF=0, AF=1, SF=1, OF=0.

Для беззнаковой интерпретации результат неверен вследствие возникающего переноса из старшего разряда, результат знаковой интерпретации корректен.

5. ВЫЧИТАНИЕ ЦЕЛЫХ ЧИСЕЛ

Вычитание можно проводить двумя способами:

1) Сведение вычитания к сложению, заменяя знак операнда *B* на противоположный.
2) Выполнение прямого (непосредственного) вычитания производится поразрядно, начиная с младших разрядов, с учетом возникающих межразрядных заёмов.

Вычитание реализуется по следующей таблице:

a_i	b_i	Z_{i-1}	r_i	Z_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Z_{i-1}- заём из *i*-го разряда;
r_i – разность;
Z_i – заём в *i*-й разряд из (*i*-1)-го разряда.

5.1. ЗАДАНИЕ 3
ВЫЧИТАНИЕ ЦЕЛЫХ ЧИСЕЛ

1. Для заданных чисел ***A*** и ***B*** выполнить операцию знакового вычитания со всеми комбинациями знаков операндов. Для каждого примера:

a) проставить межразрядные заёмы, возникающие при вычитании;

b) дать знаковую интерпретацию операндов и результатов. При получении отрицательного результата предварительно преобразовать его из дополнительного кода в прямой;

c) дать беззнаковую интерпретацию операндов и результатов, при получении неверного результата пояснить причину его возникновения;

d) показать значения арифметических флагов.

2. Сохранив значение первого операнда ***A***, выбрать такое значение ***B***, чтобы в операции вычитания с разными знаками имел место особый случай переполнения формата. Выполнить два примера, иллюстрирующие эти случаи, для каждого из них проделать пункты a, b, c, d.

3. Сохранив операнд ***B***, подобрать такое значение операнда ***A***, чтобы при вычитании отрицательного ***B*** из положительного ***A*** имело место переполнение формата, а при вычитании положительного ***B*** из отрицательного ***A*** результат был бы корректен. Выполнить два примера, иллюстрирующие этот случай. Для каждого из них проделать пункты a, b, c, d.

Варианты заданий приведены в табл. 3 приложения 1.

Переполнение при вычитании имеет место при разных знаках операндов.

Для знакового вычитания результат некорректен вследствие переполнения, о котором можно судить по одному из двух признаков:

- при разных знаках операндов знак результата отличается от знака первого операнда;
- несовпадение заёмов в два старшие разряда (один из них присутствует, а другой нет).

Примеры выполнения задания:

1. $A = 67$, $B = 51$.

$A > 0$, $B > 0$.

			Интерпретации	
			Знаковая	Беззнаковая
– $A_{пр.}$	0	1 0 0 0 0 1 1	– 67	– 67
$B_{пр.}$	0	0 1 1 0 0 1 1	51	51
$C_{пр.}$	0	0 0 1 0 0 0 0	16	16

CF=0, ZF=0, PF=0, AF=0, SF=0, OF=0.

$A < 0$, $B > 0$.

			Интерпретации	
			Знаковая	Беззнаковая
– $A_{доп.}$	1	0 1 1 1 1 0 1	– -67	– 189
$B_{пр.}$	0	0 1 1 0 0 1 1	51	51
$C_{доп.}$	1	0 0 0 1 0 1 0		138
$C_{пр.}$	1	1 1 1 0 1 1 0	-118	

CF=0, ZF=0, PF=0, AF=0, SF=1, OF=0.

$A > 0$, $B < 0$.

			Интерпретации	
			Знаковая	Беззнаковая
– $A_{пр.}$	0	1 0 0 0 0 1 1	– 67	– 67
$B_{доп.}$	1	1 0 0 1 1 0 1	-51	205
$C_{пр.}$	0	0 0 1 0 0 0 0	118	118?

CF=1, ZF=0, PF=0, AF=1, SF=0, OF=0.

Для беззнаковой интерпретации результат неверен вследствие возникающего заёма из разряда за пределами формата

$A < 0$, $B < 0$.

			Интерпретации	
			Знаковая	Беззнаковая
– $A_{доп.}$	1	0 1 1 1 1 0 1	– -67	– 189
$B_{доп.}$	1	1 0 0 1 1 0 1	-51	205
$C_{доп.}$	1	1 1 1 0 0 0 0		240?
$C_{пр.}$	1	0 0 1 0 0 0 0	-16	

CF=1, ZF=0, PF=1, AF=1, SF=1, OF=0.

Для беззнаковой интерпретации результат неверен вследствие возникающего заёма из разряда за пределами формата.

2. Правило для подбора выглядит следующим образом:

$$A + B > 128, \text{ значит } 128 - A < B < 127$$

A = 67 (неизменно), B = 64 (подбираем).

$A < 0$, $B > 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
− $A_{доп.}$	1	0	1	1	1	1	0	1	− -67	− 189
$B_{пр.}$	0	1	0	0	0	0	0	0	64	64
$C_{пр.}$	0	1	1	1	1	1	0	1	125?	125

CF=0, ZF=0, PF=1, AF=0, SF=0, OF=1.

Для знаковой интерпретации результат некорректен вследствие возникающего переполнения

$A > 0$, $B < 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
− $A_{пр.}$	0	1	0	0	0	0	1	1	− 67	− 67
$B_{доп.}$	1	1	0	0	0	0	0	0	-64	192
$C_{доп.}$	1	0	0	0	0	0	1	1		131?
$C_{пр.}$	1	1	1	1	1	1	0	1	-125?	

CF=1, ZF=0, PF=0, AF=0, SF=0, OF=1.

Для знаковой интерпретации результат некорректен вследствие возникающего переполнения, для беззнаковой интерпретации результат некорректен из-за возникающего заёма из старшего разряда

3. Значение числа B фиксируем (B = 51), а значение A подбираем согласно формуле $A + B = 128$, благодаря которой при вычитании из положительного числа отрицательного будет фиксироваться переполнение, а при вычитании из отрицательного числа положительного не будет. Тогда A = 77.

$A > 0$, $B < 0$.

									Интерпретации: Знаковая	Интерпретации: Беззнаковая
− $A_{пр.}$	0	1	0	0	1	1	0	1	− 77	− 77
$B_{доп.}$	1	1	0	0	1	1	0	1	-51	205
$C.$	1	0	0	0	0	0	0	0	-128?	128?

CF=1, ZF=0, PF=0, AF=0, SF=1, OF=1.

Результат беззнаковой интерпретации некорректен вследствие возникающего заёма из разряда за пределами формата, для знаковой интерпретации результат некорректен из-за переполнения.

$A < 0, B > 0.$

										Интерпретации	
										Знаковая	Беззнаковая
−	$A_{доп.}$	1	0	1	1	0	0	1	1	− -77	− 179
	$B_{пр.}$	0	0	1	1	0	0	1	1	51	51
	$C_{.}$	1	0	0	0	0	0	0	0	-128	128

CF=0, ZF=0, PF=0, AF=0, SF=1, OF=0.

Результаты знаковой и беззнаковой интерпретаций корректны.

6. ОПЕРАЦИЯ УМНОЖЕНИЯ ЦЕЛЫХ ЧИСЕЛ И ПРИНЦИПЫ ЕЕ РЕАЛИЗАЦИИ В ЭВМ

Умножение двоичных чисел состоит в последовательном умножении множимого на отдельные разряды множителя с суммированием результатов умножения. Результат умножения множимого на один разряд множителя принято называть *частным произведением*.

Пример.

A = 13, B = 11.

Первый способ:

$$
\begin{array}{cccccccc}
 & & & & 1 & 1 & 0 & 1 \\
 & & & & \underline{1} & \underline{0} & \underline{1} & \underline{1} \\
 & & & & 1 & 1 & 0 & 1 \\
 & & & 1 & 1 & 0 & 1 & \\
 & \underline{1} & \underline{1} & \underline{0} & \underline{1} & \underline{\ } & \underline{\ } & \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array}
\quad (143)_{10}
$$

Второй способ:

$$
\begin{array}{cccccccc}
 & 1 & 1 & 0 & 1 & & & \\
 & \underline{1} & \underline{0} & \underline{1} & \underline{1} & & & \\
 & 1 & 1 & 0 & 1 & & & \\
 & & & 1 & 1 & 0 & 1 & \\
 & \underline{\ } & \underline{\ } & \underline{\ } & \underline{1} & \underline{1} & \underline{0} & \underline{1} \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array}
$$

6.1. Особенности операции умножения целых чисел

При умножении целых чисел необходимо учитывать следующее:

• каждое частное произведение либо совпадает с множимым, либо равно нулю;

• формируемые частные произведения должны быть определенным образом сдвинуты друг относительно друга для их последующего суммирования;

• частное произведение можно формировать, начиная как от младших, так и от старших разрядов множителя;

• в общем случае для результата умножения требуется количество цифр, равное сумме количества цифр операндов. Если операнды имеют одинаковое количество цифр, то для суммы потребуется 2n разрядов.

6.1.1 Особенности реализации операций умножения в ЭВМ

Реализация операции умножения в ЭВМ имеет следующие особенности:

1. В операционном устройстве для умножения двоичных чисел должен использоваться многоразрядный двоичный сумматор, поэтому умножение реализуется в виде последовательного многошагового процесса, на каждом шаге которого проводится умножение на один разряд множителя. Для фиксации суммы частных произведений необходимо использовать $2n$-разрядный регистр при n-разрядных операндах. Перед началом операции этот регистр необходимо обнулить;

2. На каждом шаге умножения анализируется определенный разряд множителя. Если он равен 1, то на этом шаге проводится сложение суммы частных произведений с множимым. Если разряд равен 0, то сложение не проводится;

3. Каждый шаг умножения должен сопровождаться сдвигом суммы частных произведений (**СЧП**) относительно неподвижного множимого, или наоборот: сдвиг множимого относительно неподвижной СЧП;

4. Умножение можно начинать как с младших, так и со старших разрядов множителя;

5. В целях упрощения схемы умножения регистр множителя реализуется как сдвигающий, это дает возможность анализировать только один разряд регистра. При выполнении умножения, начиная от младших разрядов, схема анализа привязывается к младшему разряду регистра множителя, регистр при этом сдвигается вправо;

При реализации умножения, начиная от старших разрядов, схема анализа привязывается к старшему разряду множителя, реализуется сдвиг влево.

6. Для фиксации момента завершения операции в операционном устройстве умножения должен быть использован суммирующий или вычитающий счетчик, который считает количество разрядов множителя;

7. Так как возможно умножение, начиная от старших и с младших разрядов (то есть сдвигается множимое или сумма частных произведений) то можно использовать четыре способа (схемы) умножения.

6.1.2 Способы (схемы) реализации умножения в ЭВМ

Для учета особенностей реализации операции умножения можно использовать четыре способа (схемы) умножения:

1. начиная от младших разрядов множителя со сдвигом множимого влево;
2. начиная от младших разрядов со сдвигом СЧП вправо;
3. начиная от старших разрядов со сдвигом множимого вправо;
4. начиная от старших разрядов со сдвигом СЧП влево.

6.1.3 Анализ схем

1. В схемах умножения со сдвигом множимого для его представления требуется два n-разрядных регистра.

2. Для схем умножения со сдвигом СЧП для представления множимого требуется n-разрядный регистр.

3. В схемах умножения, начиная от старших разрядов со сдвигом множителя вправо, необходимо использовать $2n$-разрядный регистр.

4. Для схем умножения, начиняя от старших разрядов со сдвигом СЧП влево, требуется 2*n*-разрядный регистр для хранения суммы СЧП.

5. Для схем умножения, начиная от младших разрядов со сдвигом СЧП вправо, требуется *n*-разрядный регистр СЧП.

В целях экономии оборудования практически во всех ЭВМ реализована схема умножения, начиная от младших разрядов множителя со сдвигом СЧП вправо (вторая схема). Упрощенная схема операционного устройства для реализации умножения по этому способу представлена на рисунке 6.1.

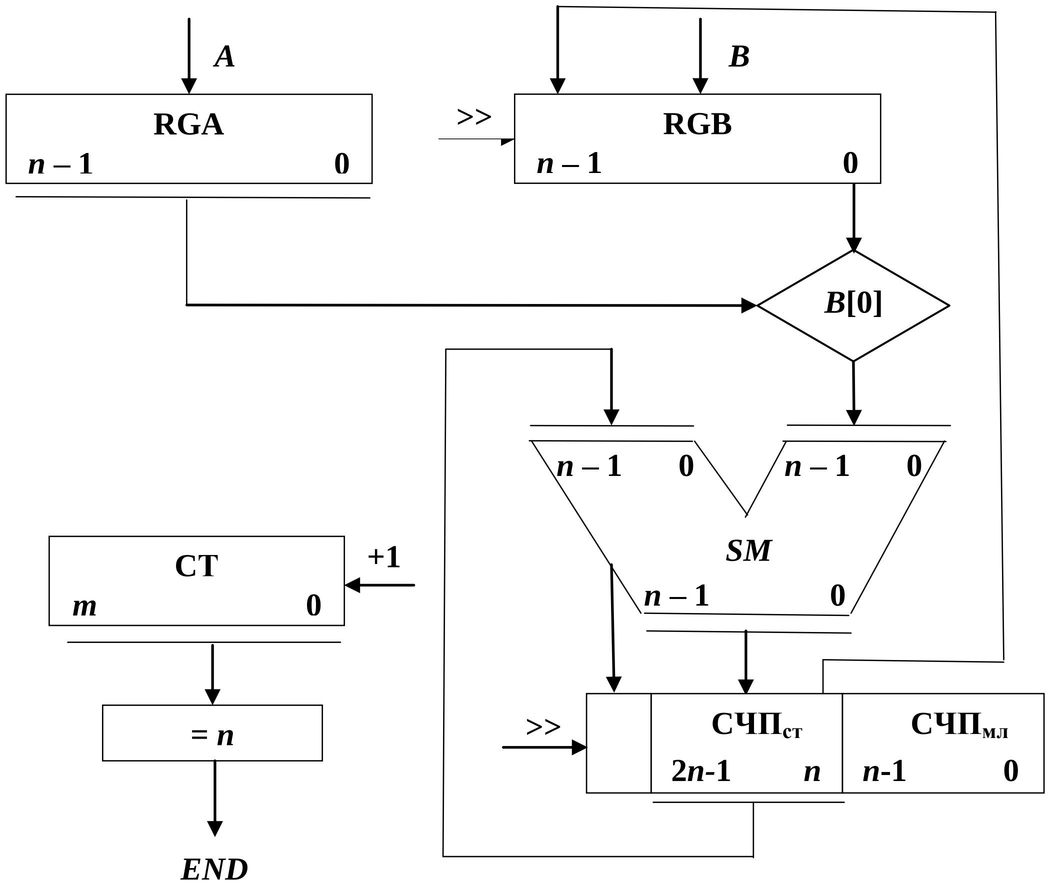


Рисунок 6.1. Схема операционного устройства умножителя по второму способу

6.2. ЗАДАНИЕ 4
УМНОЖЕНИЯ ЧИСЕЛ С ФИКСИРОВАННОЙ ЗАПЯТОЙ

1. В разрядной сетке длиной в байт (один разряд знаковый и семь – цифровых) выполнить операцию умножения заданных чисел A и B со всеми комбинациями знаков, используя метод умножения в дополнительных кодах с применением коррекции. При выполнении операции использовать способ умножения с поразрядным анализом множителя, начиная от его младших разрядов со сдвигом СЧП вправо. Результаты представить в десятичной системе и проверить их правильность.
2. В разрядной сетке длиной в байт (один разряд знаковый и семь – цифровых) выполнить операцию умножения заданных чисел A и B со всеми комбинациями знаков, используя метод умножения в дополнительных кодах без применения коррекции. При выполнении операции использовать способ умножения с поразрядным анализом множителя, начиная от его младших разрядов со сдвигом СЧП вправо. Результаты представить в десятичной системе и проверить их правильность.

Варианты заданий приведены в табл. 4 Приложения 1.

6.2.1. Основные положения

Использование метода умножения в дополнительных кодах базируется на представлении отрицательных операндов и участии их в операции в дополнительном коде. В отличие от метода умножения в прямых кодах не требуется выполнять преобразование отрицательных операндов из дополнительного кода в прямой, а отрицательного результата – из прямого кода в дополнительный. Результат операции получается в коде, зависящем от знака, т.е. положительный – в прямом, а отрицательный – в дополнительном. Знаковые разряды операндов участвуют в операции умножения точно так же, как и цифровые. Это означает, что в сложении с СЧП вступают все разряды множимого, включая знаковый, и в анализе разрядов множителя с целью определения последующих действий над СЧП участвует знаковый разряд, т.е. на него производится умножение, как и на любой цифровой.

1. Умножения в дополнительных кодах с применением коррекции.

При использовании традиционного метода умножения в дополнительных кодах только в случае положительных операндов результат получается в явном виде, в остальных же случаях он требует коррекции.

$$C = [A_{пр.}]\,[B_{пр.}].$$

При $A<0$, $B>0$ получаем псевдо-произведение:

$$C^* = [A_{доп.}]\cdot[B_{пр.}] = [2^n - |A|]\cdot[B_{пр.}] = 2^n \cdot B_{пр.} - |A|\,B_{пр.}$$

Должно быть: $C = 2^{2n} - |A|\,B_{пр.}$

При $A>0$, $B<0$:

$$C^* = A_{пр.} \cdot [2^n - |B|] = 2^n \cdot A_{пр} - A_{пр} \cdot |B|.$$

Должно быть: $C = 2^{2n} - A_{пр} \cdot |B|$.

При $A<0$, $B<0$:

$$C^* = (2^n - |A|) \cdot (2^n - |B|) = 2^{2n} - 2^n \cdot |B| - 2^n \cdot |A| + |A|\,|B|.$$

Должно быть: $C = |A|\,|B|$.

Сравнение псевдо-произведения с правильным результатом показывает, что при отрицательном множителе необходимо из старших разрядов СЧП вычесть множимое, а при отрицательном множителе из старших разрядов СЧП вычесть множимое. При отрицательных сомножителях, необходимы обе коррекции. Но из-за того, что для множителя используется сдвигающий регистр, к концу операции в регистре B хранятся старшие разряды СЧП. Поэтому применяются два вида коррекции: а) коррекция в ходе перемножения операндов; б) коррекция окончательного результата. Коррекция первого вида имеет место при отрицательном множимом и состоит в модифицированном сдвиге СЧП вправо, при котором в освобождающийся старший разряд СЧП вносится единица. Коррекция второго вида производится при отрицательном множителе и состоит в вычитании множимого из старших разрядов СЧП, которое может сводиться к сложению с дополнением множимого.

При умножении на младшие нули множителя в случае отрицательного множимого сдвиг нулевой СЧП производится обычным образом (не модифицированный), т.е. в освобождающийся старший разряд вносится нуль.

Пример. $A = 15$, $B = 13$.

Для иллюстрации метода используется укороченная по сравнению с заданием разрядная сетка для операндов (один разряд знаковый и 4 – цифровых) и результата (один разряд знаковый и 9 – цифровых). При выполнении примеров выделен анализируемый на каждом шаге разряд множителя, а также показано последовательное вытеснение множителя при его сдвиге вправо и заполнение его освобождающихся старших разрядов младшими разрядами СЧП. Таким образом, в начале операции СЧП занимает пять двоичных разрядов, а в конце – результат представлен десятью разрядами.

Представление операндов в разрядной сетке:

$[+A]_{пр} = 0.1111$; $[-A]_{доп} = 1.0001$;

$[+B]_{пр} = 0.1101$; $[-B]_{доп} = 1.0011$.

а) Множимое отрицательное ($A < 0$), множитель положительный ($B > 0$):

№ шага	Операнды и действия	СЧП (старшие разряды)	Множитель и СЧП (младшие разряды)	Пояснения
1	2	3	4	5
0	*СЧП*	0 0 0 0 0	$0\ 1\ 1\ 0\ \boxed{1}$	Обнуление старших разрядов СЧП
1	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\ 0\ 0\ 0\ 1}$ 1 0 0 0 1 1 1 0 0 0	 0 1 1 0 1 $1\ \vert\ 0\ 1\ 1\ \boxed{0}$	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
2	*СЧП*→	1 1 1 0 0	$0\ 1\ \vert\ 0\ 1\ \boxed{1}$	Модифицированный сдвиг СЧП и множителя вправо
3	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\ 0\ 0\ 0\ 1}$ 0 1 1 0 1 1 0 1 1 0	 $0\ 1\ \vert\ 0\ 1\ 1$ $1\ 0\ 1\ \vert\ 0\ \boxed{1}$	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
4	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\ 0\ 0\ 0\ 1}$ 0 0 1 1 1 1 0 0 1 1	 $1\ 0\ 1\ \vert\ 0\ 1$ $1\ 1\ 0\ 1\ \vert\ \boxed{0}$	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
5	*СЧП*→	1 1 0 0 1	1 1 1 0 1	Модифицированный сдвиг СЧП и множителя вправо

Полученный результат отрицателен и представлен в дополнительном коде:

$[C]_{доп} = [A]_{доп} \times [B]_{пр} = 1.100111101.$

Для проверки правильности результата необходимо предварительно перевести его в прямой код:

$[C]_{пр} = (1.011000011)_2 = (–195)_{10}.$

б) $A > 0$, $B < 0$:

1	2	3	4	5
0	*СЧП*	0 0 0 0 0	$1\ 0\ 0\ 1\ \boxed{1}$	Обнуление старших разрядов СЧП
1	$[A]_{пр}$ *СЧП* *СЧП*→	$\underline{0\ 1\ 1\ 1\ 1}$ 0 1 1 1 1 0 0 1 1 1	 1 0 0 1 1 $1\ \vert\ 1\ 0\ 0\ \boxed{1}$	Сложение СЧП с множимым Сдвиг СЧП и множителя вправо
2	$[A]_{пр}$ *СЧП* *СЧП*→	$\underline{0\ 1\ 1\ 1\ 1}$ 1 0 1 1 0 0 1 0 1 1	 $1\ \vert\ 1\ 0\ 0\ 1$ $0\ 1\ \vert\ 1\ 0\ \boxed{0}$	Сложение СЧП с множимым Сдвиг СЧП и множителя вправо
3	*СЧП*→	0 0 1 0 1	$1\ 0\ 1\ \vert\ 1\ \boxed{0}$	Сдвиг СЧП и множителя вправо
4	*СЧП*→	0 0 0 1 0	$1\ 1\ 0\ 1\ \vert\ \boxed{1}$	Сдвиг СЧП и множителя вправо
5	$[A]_{пр}$ *СЧП* *СЧП*→	$\underline{0\ 1\ 1\ 1\ 1}$ 1 0 0 0 1 0 1 0 0 0	 $1\ 1\ 0\ 1\ \vert\ 1$ 1 1 1 0 1	Сложение СЧП с множимым Сдвиг СЧП и множителя вправо
6	$[–A]_{доп}$ *СЧП*	$\underline{1\ 0\ 0\ 0\ 1}$ 1 1 0 0 1	 1 1 1 0 1	Коррекция результата сложение старших разрядов СЧП с дополнением множимого

Полученный результат отрицателен и представлен в дополнительном коде:

$[C]_{доп} = [A]_{пр} \times [B]_{доп} = (1.100111101)_2$,

$[C]_{пр} = (1.011000011)_2 = (–195)_{10}$.

в) Оба операнда отрицательные ($A < 0$, $B < 0$):

1	2	3	4	5
0	*СЧП*	0 0 0 0 0	1 0 0 1 $\boxed{1}$	Обнуление старших разрядов СЧП
1	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\,0\,0\,0\,1}$ 1 0 0 0 1 1 1 0 0 0	 1 0 0 1 1 1 | 1 0 0 $\boxed{1}$	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
2	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\,0\,0\,0\,1}$ 0 1 0 0 1 1 0 1 0 0	 1 | 1 0 0 1 1 1 | 1 0 $\boxed{0}$	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
3	*СЧП*→	1 1 0 1 0	0 1 1 | 1 $\boxed{0}$	Модифицированный сдвиг СЧП и множителя вправо
4	*СЧП*→	1 1 1 0 1	0 0 1 1 | $\boxed{1}$	Модифицированный сдвиг СЧП и множителя вправо
5	$[A]_{доп}$ *СЧП* *СЧП*→	$\underline{1\,0\,0\,0\,1}$ 0 1 1 1 0 1 0 1 1 1	 0 0 1 1 | 1 0 0 0 1 1	Сложение СЧП с множимым Модифицированный сдвиг СЧП и множителя вправо
6	$[–\boldsymbol{A}]_{пр}$ *СЧП*	$\underline{0\,1\,1\,1\,1}$ 0 0 1 1 0	 0 0 0 1 1	Коррекция результата сложение старших разрядов СЧП с дополнением множимого

Полученный результат положителен и представлен в прямом коде:

$[C]_{пр} = [A]_{доп} \times [B]_{доп} = (0.011000011)_2 = (195)_{10}$.

2. Умножение в дополнительных кодах без коррекции.

Наряду с традиционным методом умножения в дополнительных кодах, используемом в ЭВМ общего назначения и требующим коррекции результата, достаточно широкое применение в микроЭВМ находит метод Бута, при котором не требуется выполнять коррекцию. Особенность метода состоит в выполнении сложения или вычитания СЧП и множимого на каждом шаге умножения в зависимости от того, как после сдвига вправо изменяется младший разряд множителя. При его изменении с единицы на ноль производится сложение СЧП с множимым, а при изменении с нуля на единицу – вычитание множимого из СЧП, которое может быть реализовано как сложение с дополнением множимого. Если младший разряд множителя при сдвиге не изменяется, то на данном шаге не производится сложения (вычитания), а выполняется только сдвиг СЧП и множителя вправо.

При реализации этого метода происходит чередование сложений и вычитаний множимого и СЧП, вследствие чего старший разряд СЧП в явном виде представляет его знак. При сдвиге СЧП вправо значение знакового разряда сохраняется (арифметический сдвиг).

Необходимо отметить, что:

а) При умножении на младшую единицу множителя производится вычитание множимого из СЧП, поскольку считается, что происходит изменение младшего разряда множителя с нуля на единицу.

б) При умножении на младшие нули множителя осуществляется сдвиг нулевой СЧП и множителя вправо до появления единицы в младшем разряде множителя, после чего производится вычитание множимого из СЧП.

Пример. $A = 11$, $B = 15$.

Представление операндов в разрядной сетке:

$[+A]_{пр} = 0.1011$; $[-A]_{доп} = 1.0101$;

$[+B]_{пр} = 0.1111$; $[-B]_{доп} = 1.0001$.

а) Оба операнда положительные ($A > 0$, $B > 0$):

0	*СЧП*	0 0 0 0 0	0 1 1 1 $\boxed{1}$	Обнуление старших разрядов СЧП
1	$[-A]_{доп}$	$\underline{1\,0\,1\,0\,1}$		Младший разряд множителя равен 1: вы-
	СЧП	1 0 1 0 1	0 1 1 1 1	читание множимого из СЧП
	СЧП→	1 1 0 1 0	1 $\vert$ 0 1 1 $\boxed{1}$	Сдвиг СЧП и множителя вправо
2				При сдвиге младший разряд не изменился
	СЧП→	1 1 1 0 1	0 1 $\vert$ 0 1 $\boxed{1}$	Сдвиг СЧП и множителя вправо
3				При сдвиге младший разряд не изменился
	СЧП→	1 1 1 1 0	1 0 1 $\vert$ 0 $\boxed{1}$	Сдвиг СЧП и множителя вправо
4				При сдвиге младший разряд не изменился
	СЧП→	1 1 1 1 1	0 1 0 1 $\vert$ $\boxed{0}$	Сдвиг СЧП и множителя вправо
5	$[A]_{пр}$	$\underline{0\,1\,0\,1\,1}$		При сдвиге младший разряд множителя
	СЧП	0 1 0 1 0	0 1 0 1 $\vert$ 0	изменился с 1 на 0: сложение СЧП с множимым
	СЧП→	0 0 1 0 1	0 0 1 0 1	Сдвиг СЧП и множителя вправо

Полученный результат представлен в прямом коде и равен:

$$[C]_{пр} = 2^0 + 2^2 + 2^5 + 2^7 = 1 + 4 + 32 + 128 = 165.$$

Проверка: $C = 11 \times 15 = 165$.

б) Оба операнда отрицательные ($A < 0$, $B < 0$):

1	2	3	4	5
0	*СЧП*	0 0 0 0 0	1 0 0 0 $\boxed{1}$	Обнуление старших разрядов СЧП
1	$[-A]_{пр}$	$\underline{0\,1\,0\,1\,1}$		Вычитание множимого из СЧП
	СЧП	0 1 0 1 1	1 0 0 0 1	
	СЧП→	0 0 1 0 1	1 $\vert$ 1 0 0 $\boxed{0}$	Сдвиг СЧП и множителя вправо
2	$[A]_{доп}$	$\underline{1\,0\,1\,0\,1}$		Сложение СЧП с множимым
	СЧП	1 1 0 1 0	1 $\vert$ 1 0 0 0	
	СЧП→	1 1 1 0 1	0 1 $\vert$ 1 0 $\boxed{0}$	Сдвиг СЧП и множителя вправо
3	*СЧП*→	1 1 1 1 0	1 0 1 $\vert$ 1 $\boxed{0}$	Сдвиг СЧП и множителя вправо
4	*СЧП*→	1 1 1 1 1	0 1 0 1 $\vert$ $\boxed{1}$	Сдвиг СЧП и множителя вправо
5	$[-A]_{пр}$	$\underline{0\,1\,0\,1\,1}$		Сложение СЧП с множимым
	СЧП	0 1 0 1 0	0 1 0 1 $\vert$ 1	
	СЧП→	0 0 1 0 1	0 0 1 0 1	Сдвиг СЧП и множителя вправо

Полученный результат положителен и представлен в прямом коде:

$[C]_{пр} = [A]_{доп} \times [B]_{доп} = (0.010100101)_2 = (165)_{10}$.

7. ОПЕРАЦИЯ ЦЕЛОЧИСЛЕННОГО ДЕЛЕНИЯ И СПОСОБЫ ЕЕ РЕАЛИЗАЦИИ В ЭВМ

7.1. Особенности операции двоичного деления

Пример. $A = 130$, $B = 10$.

$A/B = C(R)$,

где A – делимое,

B – делитель,

C – частное,

R – остаток.

$A = (130)_{10} = (10000010)_2$

$B = (10)_{10} = (1010)_2$

```
_1 0 0 0 0 0 1 0  | 1 0 1 0
 1 0 1 0          | 0  1  1  0  1
 1 1 1 0 (R<0)
_1 0 0 0 0 0 1 0
   1 0 1 0
 0 0 1 1 0 (R>0)
   _1 1 0 0 1 0
     1 0 1 0
     0 0 1 0 (R>0)
     _0 1 0 1 0
       1 0 1 0
       1 0 1 1 (R<0)
     _0 1 0 1 0
         1 0 1 0
         0 0 0 0 (R≥0)
```

Из приведенного примера отчетливо проявляются следующие особенности двоичного деления:

1) Процесс деления сводится к последовательному вычитанию делителя первоначально из делимого, а далее – из получаемых текущих остатков (под текущим остатком будем понимать промежуточный результат вычитания делителя из делимого на первом шаге или очередного остатка на последующих шагах).

2) На начальном шаге делитель совмещается со старшими разрядами делимого, а затем на каждом шаге делитель сдвигается на одну позицию (разряд) вправо относительно неподвижного текущего остатка. На последнем шаге делитель совмещается с младшими разрядами текущего остатка.

3) Цифры частного, вырабатываемые на каждом шаге, определяются знаком текущего остатка. Для остатка большего или равного 0 цифра частного равна 1, для остатка, который меньше 0, цифра частного равна 0.

4) О знаке текущего остатка можно судить по наличию или отсутствию заёма в старший разряд делимого при вычитании. Если заём есть, то результат вычитания (т.е. текущий остаток) отрицателен. Если заём отсутствует, то результат вычитания и, соответственно, текущий остаток не отрицателен.

5) При получении отрицательного остатка на очередном шаге перед переходом к следующему шагу необходимо выполнить *восстановление остатка* путем сложения отрицательного остатка с делителем (это действие в примере опущено).

7.2. Особенности реализации целочисленного деления в ЭВМ

• Т.к. операция деления является обратной по отношению к умножению, а результат сложения *n*-разрядных сомножителей представляется в 2n-разрядном формате, то делимое по сравнению с делителем так же должно представляться в удвоенном формате.

• В качестве результата деления формируется не только частное, но и остаток. Операция деления с остатком должна удовлетворять следующему соотношению:

$$B \cdot C + R = A.$$

Для большинства современных процессоров (в том числе и фирмы Intel) по окончании операции деления частное занимает младшие разряды делимого, а остаток – старшие разряды делимого.

• В целях экономии оборудования на каждом шаге операции деления осуществляется не сдвиг делителя вправо относительно неподвижного текущего остатка, а сдвиг текущего остатка влево относительно неподвижного делителя. При этом делитель должен совмещаться со старшими разрядами сначала делимого, а на последующих шагах – текущего остатка.

• При получении отрицательного остатка на каком-либо шаге для перехода к следующему шагу требуется предварительное восстановление остатка путем его сложения с делителем. Подробный подход к реализации операции деления называется «метод деления с восстановлением остатка». В целях экономии времени выполнения операции деления в современных процессорах деление реализуется с использованием метода без восстановления остатка. Идея метода сводится к тому, что после получения отрицательного остатка на очередном шаге деления осуществляется его сдвиг влево так же, как и для положительного остатка, однако на следующем шаге производится не вычитание делителя из остатка, а сложение делителя с остатком.

7.3. Обоснование метода целочисленного деления

Допустим, что на i-ом шаге деления получен остаток R_i<0, тогда для получения очередного остатка R_{i+1} на (i+1)-ом шаге над текущим остатком R_i выполняются следующие действия:

а) По методу с восстановлением остатка:

$R_{i+1} = (R_i + B)\cdot 2 - B = 2R_i + 2B - B = 2R_i + B,$

$(R_i + B)$ – сложение с делителем;

$(R_i + B)\cdot 2$ – сдвиг влево на 1 разряд;

$(R_i + B)\cdot 2 - B$ – вычитание делителя.

б) По методу без восстановления остатка:

$R_{i+1} = 2R_i + B.$

$2R_i$ – сдвиг отрицательного остатка влево;

$2R_i + B$ – сложение с делителем.

При некоторых соотношениях между делимым и делителем может оказаться, что частное не помещается в формат делителя. Подобная ситуация возникает также, если делитель равен 0. Этот особый случай распознается и фиксируется на начальных шагах операции деления. Наличие подобного случая классифицируется как некорректность целочисленного деления и приводит к прерыванию выполняемой программы. Выход на это прерывание может быть зафиксирован при попытке разделить двухбайтное делимое, равное 1000, на однобайтный делитель, равный 1, 2, 3, при выполнении команды DIV (беззнаковое деление), а при выполнении команды IDIV также при B = 4,5.

7.4. Деление беззнаковых целых чисел

В основном беззнаковое деление реализуется по общим принципам, описанным ранее. Это означает, что цифра частного, вырабатываемая на каждом шаге деления, определяется знаком текущего остатка, получаемого на данном шаге. При отрицательном остатке вырабатывается цифра частного, равная 0; при положительном остатке – равная 1.

Проверка корректности беззнакового деления осуществляется следующим образом. Для n-разрядного делителя и, следовательно, n-разрядного частного условие корректности принимает вид:

$$A/B \le 2^n - 1 \;\Rightarrow\; A/B < 2^n$$

$$A < B \cdot 2^n$$

$$\underline{\mathbf{A - B \cdot 2^n < 0}}$$

Из полученного условия следует, что для проверки корректности беззнакового деления необходимо произвести вычитание делителя из старших разрядов делимого. Если полученная разность отрицательна, то деление корректно, т.е. частное помещается в формате делителя. Если же результат так называемого «пробного вычитания» положителен или равен 0, то результат деления в виде частного не может быть помещен в n-разрядный формат. При обнаружении такого случая генерируется требование прерывания выполняемой программы по причине *некорректности целочисленного деления*. Для процессоров Intel это прерывание является стандартным, ему присвоен номер, равный нулю.

Пример. $A = 141$, $B = 13$, $C = 10$, $R = 11$.

Минимальный формат для выполнения операции – $n = 4$ (формат делителя).

$A = (141)_{10} = (10001101)_2$,

$B = (13)_{10} = (1101)_2$.

N шага	Операнды	Действия: Делимое/остаток старш. разряды	Действия: Делимое/остаток младш. разряды	Комментарии
0	A B $R_0<0$	_1 0 0 0 1 1 0 1 1 0 1 1	1 1 0 1 1 1 0 1	Деление корректно
1	$\leftarrow R_0$ B $\boldsymbol{R_1>0}$	+ 0 1 1 1 1 1 0 1 0 1 0 0	1 0 1 ¦ 0 1 0 1 ¦ **1**	
2	$\leftarrow R_1$ B $\boldsymbol{R_2<0}$	_1 0 0 1 1 1 0 1 1 1 0 0	0 1 ¦ 1 0 0 1 ¦ 1 **0**	
3	$\leftarrow R_2$ B $\boldsymbol{R_3>0}$	+ 1 0 0 0 1 1 0 1 0 1 0 1	1 ¦ 1 0 0 1 ¦ 1 0 **1**	
4	$\leftarrow R_3$ B $\boldsymbol{R_4<0}$	_1 0 1 1 1 1 0 1 1 1 1 0	1 0 1 0 1 0 1 **0**	частное = $(10)_{10}$
5	B R	1 1 0 1 1 0 1 1		Коррекция остатка. Остаток = $(11)_{10}$

Пояснения к примеру

1. Формируемые на каждом шаге (кроме первого) цифры частного помещаются в освобождающийся при сдвиге влево младший разряд остатка.

2. Знак остатка определяется наличием или отсутствием переноса из старшего разряда при сложении или заёма в старший разряд при вычитании. Наличие переноса при сложении свидетельствует о получении положительного остатка, а его отсутствие – о получении отрицательного остатка. В свою

очередь наличие заёма при вычитании свидетельствует о получении отрицательного остатка, а его отсутствие – о получении положительного остатка.

3. По результату пробного вычитания на начальном (нулевом) шаге осуществляется проверка корректности деления, поэтому на данном шаге никакой цифры частного не формируется.

Пример фиксации некорректности деления.

$A = 141$, $B = 8$.

N шага	Операнды	Действия		Комментарии
		Делимое/остаток старш. разряды	Делимое/остаток младш. разряды	
0	A B $R_0 \geq 0$	_1 0 0 0 1 0 0 0 0 0 0 0	1 1 0 1	Заем отсутствует. Некорректность деления.

Для рассматриваемых делимого и делителя $C = 17$.

Аналогичная ситуация фиксации некорректности деления будет иметь место и при $B < 8$, в том числе и при $B = 0$.

7.5. Возможные модернизации метода деления

1) Для явного представления знака остатка можно использовать дополнительный старший бит как в регистре остатка, так и в сумматоре-вычитателе. Следует иметь в виду, что при сдвиге остатка влево может произойти искажение старшего бита остатка, интерпретируемого как знаковый. В связи с этим действия на очередном шаге алгоритма деления следует определять значением знакового бита остатка до сдвига, а не после.

Замечание: для устранения этого недостатка можно использовать модифицированный код с двумя битами для представления знаков. В этом случае старший знаковый бит даже после сдвига остатка влево будет сохранять верное значение знака остатка. Это замечание касается также и знакового деления целых чисел.

2) Для упрощения метода операцию вычитания делителя можно заменить операцией сложения с его дополнительным кодом.

В процессорах Intel беззнаковое деление реализуется командой DIV, а знаковое - командой IDIV.

7.6. Деление знаковых чисел

Аналогично операции умножения знаковое деление может быть реализовано одним из двух методов:

- метод деления в прямых кодах;
- метод деления в дополнительных кодах.

7.6.1. Основные особенности метода деления в прямых кодах

1. Отрицательные операнды предварительно преобразуются из дополнительного кода в прямой.
2. Деление модулей операндов осуществляется аналогично методу деления для беззнаковых чисел.
3. Знак частного вырабатывается отдельным действием как сумма по модулю 2 знаковых разрядов операндов.
4. Знаку остатка присваивается знак делимого. Исключением из правил 3, 4 является получение нулевого частного и(или) нулевого остатка, которым, независимо от знаков операндов, присваивается знак + (положительный ноль).
5. Отрицательные результаты в конце операции преобразуются из прямого кода в дополнительный.
6. Существенным отличием знакового деления в прямых кодах от беззнакового деления является проверка корректности деления, выполняемая на начальных шагах алгоритма.

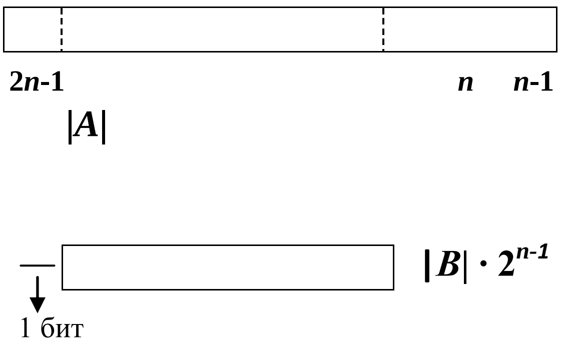
7.6.2. Обоснование метода проверки корректности деления

В связи с тем, что старший бит n-разрядного делителя и, соответственно, частного отводится для представления знака, условие корректности для деления модулей операндов имеет вид:

$|A| / |B| \leq 2^{n-1} - 1 \;\; => \;\; |A| / |B| < 2^{n-1}$

$$\mathbf{|A| / |B| \cdot 2^{n-1} < 0.}$$

Из полученного соотношения следует, что на этапе пробного вычитания следует делимое и делитель расположить друг под другом следующим образом:



В целях однообразия выполнения пробного вычитания с основным циклом деления, в котором делитель размещается под старшими разрядами остатка, необходимо перед выполнением пробного вычитания сдвинуть делимое на один разряд влево и после этого произвести вычитание делителя из его старших разрядов.

7.6.3. Основные особенности метода деления в дополнительных кодах

1. Цифры частного, формируемые на каждом шаге, определяются не только знаком остатка, но и знаком делителя. При их совпадении цифра частного равна 1, при несовпадении – 0.

2. Действие, выполняемое над текущим остатком на каждом шаге, определяется не только знаком остатка, но и знаком делителя. При их совпадении производится вычитание делителя из старших разрядов остатка, а при несовпадении – сложение делителя со старшими разрядами остатка. Вычитание делителя может заменяться сложением с его дополнительным кодом.

3. Коррекция остатка выполняется в конце операции (после выработки всех цифр частного) в том случае, если знак последнего остатка не совпадает со знаком делимого. Эта коррекция осуществляется действием над остатком (прибавление или вычитание делителя), аналогичным действиям в основном цикле деления (определяется сравнением знаков остатка и делителя).

4. Коррекция частного выполняется только при отрицательном делимом и нулевом остатке и состоит в инкременте (увеличении на единицу) для положительного частного и декременте для отрицательного частного.

5. Проверка корректности деления реализуется аналогично методу деления в прямых кодах только в случае положительных операндов. Для остальных комбинаций знаков имеют место следующие нюансы в обосновании методов проверки корректности:

1. $\boldsymbol{A > 0, B > 0.}$

$A/B < 2^{n-1}$;

$A - B{\cdot}2^{n-1} < 0.$

2. $\boldsymbol{A < 0, B < 0.}$

$A/B < 2^{n-1}$;

$A - B{\cdot}2^{n-1} > 0.$

Из полученных соотношений следует, что **проверка корректности при одинаковых знаках делимого и делителя** реализуется следующим образом:

a) выполняется предварительный сдвиг делимого на один разряд влево;

b) из старших разрядов сдвинутого делимого вычитается делитель;

c) знак полученного остатка сравнивается со знаком делимого. Если они совпадают, то деление некорректно, и операция завершается выходом на прерывание. Если знаки делимого и первого остатка не совпадают, то формируется старший разряд частного, интерпретируемый как знак, и осуществляется переход к основному циклу деления. Знак частного формируется по тем же правилам, что и любая его цифра, т.е. как результат сравнения знаков текущего остатка и делителя.

3. $\boldsymbol{A < 0, B > 0.}$

$A/B \geq - 2^{n-1}$;

$A/B \geq - 2^{n-1} - 1$;

$$A + B \cdot 2^{n-1} + B > 0.$$

4. $A > 0, B < 0$.

$$A/B > - 2^{n-1} - 1;$$

$$A + B \cdot 2^{n-1} + B < 0.$$

Из полученных соотношений следует, что **проверка корректности деления при разных знаках делимого и делителя** выполняется в такой последовательности:

1. сложение делителя с младшими разрядами делимого;
2. сдвиг полученного остатка влево;
3. сложение делителя со старшими разрядами остатка;
4. знак полученного остатка сравнивается со знаком делимого. Если они совпадают, то деление некорректно, и операция завершается выходом на прерывание. Если знаки делимого и первого остатка не совпадают, то формируется старший разряд частного, интерпретируемый как знак, и осуществляется переход к основному циклу деления. Знак частного формируется по тем же правилам, что и любая его цифра, т.е. как результат сравнения знаков текущего остатка и делителя.

7.7. З А Д А Н И Е 5
ДЕЛЕНИЕ ЧИСЕЛ С ФИКСИРОВАННОЙ ЗАПЯТОЙ

1. Выполнить операцию деления заданных целых чисел ***A*** и ***B*** со всеми комбинациями знаков, используя метод деления в дополнительных кодах. Для представления делимого (***A***) использовать 16 двоичных разрядов (один – знаковый и 15 – цифровых), для представления делителя (***B***) – 8 разрядов (один – знаковый и 7 – цифровых). Остаток от деления и частное представляются в той же разрядной сетке, что и делитель.

2. Результаты операции представить в десятичной системе счисления и проверить их правильность.

Варианты задания приведены в табл. 5 Приложения 1.

П р и м е р 1. Деление с ненулевым остатком.

$\boldsymbol{A} = 139, \quad \boldsymbol{B} = 13.$

Для иллюстрации метода используется укороченная по сравнению с заданием разрядная сетка: для делимого 10 разрядов (один знаковый и 9 – цифровых), для делителя, остатка и частного – 5 разрядов (один знаковый и 4 – цифровых).

Представление операндов с разными знаками в разрядной сетке:

$[+\boldsymbol{A}]_{пр} = 0.010001011; \quad [-\boldsymbol{A}]_{доп} = 1.101110101;$

$[+\boldsymbol{B}]_{пр} = 0.1101; \quad [-\boldsymbol{B}]_{доп} = 1.0011.$

а) Делимое положительное (***A*** > 0), делитель отрицательный (***B*** < 0):

№ шага	Операнды и действия	Делимое и остаток (старшие разряды)	Делимое и остаток (младшие разряды), частное	Пояснения
1	2	3	4	5
0	$[A]_{пр}$	0 0 1 0 0	0 1 0 1 1	Делимое
1	$[B]_{доп}$ R_1' $\overleftarrow{R_1'}$ $[B]_{доп}$ R_1	$\underline{1\ 1\ 1\ 1\ 1}$ 0 0 0 1 1 0 0 1 1 1 + $\underline{1\ 0\ 0\ 1\ 1}$ $\boxed{1}$ 1 0 1 0 1 1 0 1 0 ЗнR₁=ЗнB	$\underline{1\ 0\ 0\ 1\ 1}$ 1 1 1 1 0 1 1 1 0 | 0 1 1 1 0 | $\boxed{1}$	Сложение с делителем, выровненным по младшим разрядам Сдвиг остатка влево Сложение с делителем, выровненным по старшим разрядам Знак первого остатка не совпадает со знаком делимого–деление корректно Формирование знака частного

1	2	3	4	5
2	$\overleftarrow{R}_1$ $[-B]_{пр}$ R_2	$+\,10101$ $\underline{01101}$ 00010 $ЗнR_2 \neq ЗнB$	$110 \vert 10$ $110 \vert 1\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
3	$\overleftarrow{R}_2$ $[B]_{доп}$ R_3	$+\,00101$ $\underline{10011}$ 11000 $ЗнR_3 = ЗнB$	$10 \vert 100$ $10 \vert 10\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
4	$\overleftarrow{R}_3$ $[-B]_{пр}$ R_4	$+\,10001$ $\underline{01101}$ 11110 $ЗнR_4 = ЗнB$	$0 \vert 1010$ $0 \vert 101\boxed{1}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
5	$\overleftarrow{R}_4$ $[-B]_{пр}$ R_5	$+\,11100$ $\underline{01101}$ 01001 $ЗнR_5 \neq ЗнB$	10110 $1011\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

В результате выполнения операции получено отрицательное частное $[\boldsymbol{C}]_{доп} = (1.0110)_2$, $[\boldsymbol{C}]_{пр} = (1.1010)_2 = (-10)_{10}$ и положительный остаток $[\boldsymbol{R}]_{пр} = (0.1001)_2 = (+9)_{10}$, которые соответствуют истинным значениям: $(-10) \times (-13) + 9 = 139$.

б) Делимое отрицательное ($\boldsymbol{A} < 0$), делитель положительный ($\boldsymbol{B} > 0$):

1	2	3	4	5
0	$[A]_{доп}$	11011	10101	Делимое
1	$[B]_{пр}$ R'_1 $\overleftarrow{R}'_1$ $[B]_{пр}$ R_1	$\underline{00000}$ 11100 11000 $+\,\underline{01101}$ $\boxed{0}0101$ 00101 $ЗнR_1 = ЗнB$	$\underline{01101}$ 00010 $0010 \vert 0$ $0010 \vert \boxed{1}$	Сложение с делителем, выровненным по младшим разрядам Сдвиг остатка влево Сложение с делителем, выровненным по старшим разрядам Знак первого остатка не совпадает со знаком делимого–деление корректно Формирование знака частного
2	$\overleftarrow{R}_1$ $[-B]_{доп}$ R_2	$+\,01010$ $\underline{10011}$ 11101 $ЗнR_2 \neq ЗнB$	$010 \vert 10$ $010 \vert 1\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

3	$\overleftarrow{R}_2$ $[B]_{пр}$ R_3	+ 1 1 0 1 0 0 1 1 0 1 0 0 1 1 1 ЗнR_3=ЗнB	1 0 ∣ 1 0 0 1 0 ∣ 1 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
4	$\overleftarrow{R}_3$ $[–B]_{доп}$ R_4	+ 0 1 1 1 1 1 0 0 1 1 0 0 0 1 0 ЗнR_4=ЗнB	0 ∣ 1 0 1 0 0 ∣ 1 0 1 $\boxed{1}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
5	$\overleftarrow{R}_4$ $[–B]_{доп}$ R_5	+ 0 0 1 0 0 1 0 0 1 1 1 0 1 1 1 ЗнR_5≠ЗнB	1 0 1 1 0 1 0 1 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

В результате выполнения операции получено отрицательное частное $[\boldsymbol{C}]_{\boldsymbol{доп}} = (1.0110)_2$, $[\boldsymbol{C}]_{\boldsymbol{пр}} = (1.1010)_2 = (–10)_{10}$ и отрицательный остаток $[\boldsymbol{R}]_{\boldsymbol{доп}} = (1.0111)_2$, $[\boldsymbol{R}]_{\boldsymbol{пр}} = (1.1001)_2 = (–9)_{10}$, которые соответствуют истинным значениям: $(–10) \times 13 + (–9) = –139$.

в) Оба операнда отрицательные ($\boldsymbol{A} < 0$, $\boldsymbol{B} < 0$):

1	2	3	4	5
0	$[A]_{доп}$	1 1 0 1 1	1 0 1 0 1	Делимое
1	$[\overleftarrow{A}]_{доп}$ $[–B]_{пр}$ R_1	+ 1 0 1 1 1 0 1 1 0 1 $\boxed{0}$ 0 1 0 0 0 0 1 0 0 ЗнR_1≠ЗнB	0 1 0 1 ∣ 0 0 1 0 1 ∣ $\boxed{0}$	Сдвиг делимого влево Вычитание делителя Знак первого остатка не совпадает со знаком делимого–деление корректно Формирование знака частного
2	$\overleftarrow{R}_1$ $[B]_{доп}$ R_2	+ 0 1 0 0 0 1 0 0 1 1 1 1 0 1 1 ЗнR_2=ЗнB	1 0 1 ∣ 0 0 1 0 1 ∣ 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
3	$\overleftarrow{R}_2$ $[–B]_{пр}$ R_3	+ 1 0 1 1 1 0 1 1 0 1 0 0 1 0 0 ЗнR_3≠ЗнB	0 1 ∣ 0 1 0 0 1 ∣ 0 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

4	$\overleftarrow{R}_3$ $[B]_{доп}$ R_4	+ 0 1 0 0 0 1 0 0 1 1 1 1 0 1 1 ЗнR_4=ЗнB	1 │ 0 1 0 0 1 │ 0 1 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
5	$\overleftarrow{R}_4$ $[-B]_{пр}$ R_5	+ 1 0 1 1 1 0 1 1 0 1 0 0 1 0 0 ЗнR_5≠ЗнB	0 1 0 1 0 0 1 0 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
6	$[B]_{доп}$ R_6	1 0 0 1 1 1 0 1 1 1	0 1 0 1 0	Коррекция остатка: сложение с делителем Результат

В результате выполнения операции получено положительное частное $[\boldsymbol{C}]_{пр} = (0.1010)_2 = (+10)_{10}$ и отрицательный остаток $[\boldsymbol{R}]_{доп} = (1.0111)_2$, $[\boldsymbol{R}]_{пр} = (1.1001)_2 = (-9)_{10}$, которые соответствуют истинным значениям: $10 \times (-13) - 9 = -139$.

П р и м е р 2. Деление с нулевым остатком.

$\boldsymbol{A} = 72$, $\boldsymbol{B} = 6$.

Представление операндов в разрядной сетке.

$[+\boldsymbol{A}]_{пр} = 0.001001000$; $[-\boldsymbol{A}]_{доп} = 1.110111000$;

$[+\boldsymbol{B}]_{пр} = 0.0110$; $[-\boldsymbol{B}]_{доп} = 1.1010$.

а) Делимое отрицательное ($\boldsymbol{A} < 0$), делитель положительный ($\boldsymbol{B} > 0$):

1	2	3	4	5
0	$[A]_{доп}$	1 1 1 0 1	1 1 0 0 0	Делимое
1	$[B]_{пр}$ R_1' $\overleftarrow{R}_1'$ $[B]_{пр}$ R_1	0 0 0 0 0 1 1 1 0 1 1 1 0 1 1 + 0 0 1 1 0 $\boxed{0}$ 0 0 0 1 0 0 0 0 1 ЗнR_1=ЗнB	0 0 1 1 0 1 1 1 1 0 1 1 1 0 │ 0 1 1 1 0 │ $\boxed{1}$	Сложение с делителем, выровненным по младшим разрядам Сдвиг остатка влево Сложение с делителем, выровненным по старшим разрядам Знак первого остатка не совпадает со знаком делимого–деление корректно Формирование знака частного
2	$\overleftarrow{R}_1$ $[-B]_{доп}$ R_2	+ 0 0 0 1 1 1 1 0 1 0 1 1 1 0 1 ЗнR_2≠ЗнB	1 1 0 │ 1 0 1 1 0 │ 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

3	$\overleftarrow{R}_2$ $[B]_{пр}$ R_3	+ 1 1 0 1 1 0 0 1 1 0 $\boxed{0}$ 0 0 0 1 ЗнR_3=ЗнB	1 0 | 1 0 0 1 0 | 1 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
4	$\overleftarrow{R}_3$ $[-B]_{доп}$ R_4	+ 0 0 0 1 1 1 1 0 1 0 $\boxed{1}$ 1 1 0 1 ЗнR_4≠ЗнB	0 | 1 0 1 0 0 | 1 0 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
5	$\overleftarrow{R}_4$ $[B]_{пр}$ R_5	+ 1 1 0 1 0 0 0 1 1 0 $\boxed{0}$ 0 0 0 0 ЗнR_5=ЗнB	1 0 1 0 0 1 0 1 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
6	$[-1]_{доп}$	 0 0 0 0 0	1 1 1 1 1 1 0 1 0 0	Коррекция частного: вычитание единицы Результат

В результате выполнения операции получено отрицательное частное

$[\boldsymbol{C}]_{\boldsymbol{доп}}$ = $(1.0100)_2$, $[\boldsymbol{C}]_{\boldsymbol{пр}}$ = $(1.1100) = (-12)_{10}$ и нулевой остаток.

б) Оба операнда отрицательные ($\boldsymbol{A} < 0$, $\boldsymbol{B} < 0$):

1	2	3	4	5
0	$[A]_{доп}$	1 1 1 0 1	1 1 0 0 0	Делимое
1	$[\overleftarrow{A}]_{доп}$ $[-B]_{пр}$ R_1	+ 1 1 0 1 1 0 0 1 1 0 $\boxed{0}$ 0 0 0 1 0 0 0 0 1 ЗнR_1≠ЗнB	1 0 0 0 | 0 1 0 0 0 | $\boxed{0}$	Сдвиг делимого влево Вычитание делителя Знак первого остатка не совпадает со знаком делимого–деление корректно Формирование знака частного
2	$\overleftarrow{R}_1$ $[B]_{доп}$ R_2	+ 0 0 0 1 1 1 1 0 1 0 1 1 1 0 1 ЗнR_2≠ЗнB	0 0 0 | 0 0 0 0 0 | 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
3	$\overleftarrow{R}_2$ $[-B]_{пр}$ R_3	+ 1 1 0 1 0 0 0 1 1 0 0 0 0 0 0 ЗнR_3≠ЗнB	0 0 | 0 1 0 0 0 | 0 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного

4	$\overleftarrow{R}_3$ $[B]_{доп}$ R_4	$+$ 0 0 0 0 0 $\underline{1\ 1\ 0\ 1\ 0}$ 1 1 0 1 0 ЗнR_4≠ЗнB	0 | 0 1 0 0 0 | 0 1 0 $\boxed{1}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
5	$\overleftarrow{R}_4$ $[-B]_{пр}$ R_5	$+$ 1 0 1 0 0 $\underline{0\ 0\ 1\ 1\ 0}$ 1 1 0 1 0 ЗнR_5≠ЗнB	0 1 0 1 0 0 1 0 1 $\boxed{1}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
6	$[-B]_{пр}$ R_6	$\underline{0\ 0\ 1\ 1\ 0}$ 0 0 0 0 0	0 1 0 1 1	Коррекция остатка, совпадающего с делителем: вычитание делителя
7	$[+1]_{пр}$	 0 0 0 0 0	0 0 0 0 1 ――――― 0 1 1 0 0	Коррекция частного: сложение с единицей Результат

В результате выполнения операции получено положительное частное

$[\boldsymbol{C}]_{\boldsymbol{пр}}$ = $(0.1100)_2 = (+12)_{10}$ и нулевой остаток.

Пример 3. Получение максимального по модулю частного и фиксация некорректности деления.

$\boldsymbol{A} = 254$, $\boldsymbol{B} = 15$.

Представление операндов в разрядной сетке.

$[+\boldsymbol{A}]_{пр} = 0.011111110$; $[-\boldsymbol{A}]_{доп} = 1.100000010$;

$[+\boldsymbol{B}]_{пр} = 0.1111$; $[-\boldsymbol{B}]_{доп} = 1.0001$.

а) Оба операнда положительные ($\boldsymbol{A} > 0$, $\boldsymbol{B} > 0$):

1	2	3	4	5
0	$[A]_{пр}$	0 0 1 1 1	1 1 1 1 0	Делимое
1	$[\overleftarrow{A}]_{пр}$ $[-B]_{доп}$ R_1	$+$ 0 1 1 1 1 $\underline{1\ 0\ 0\ 0\ 1}$ $\boxed{0}$ 0 0 0 0	1 1 1 0 | 0	Сдвиг делимого влево Вычитание делителя Знак первого остатка совпадает со знаком делимого – деление некорректно

б) Делимое положительное ($\boldsymbol{A} > 0$), делитель отрицательный ($\boldsymbol{B} < 0$):

1	2	3	4	5
0	$[A]_{пр}$	0 0 1 1 1	1 1 1 1 0	Делимое
1	$[B]_{доп}$ R'_1 $\overleftarrow{R}'_1$ $[B]_{доп}$ R_1	$\underline{1\,1\,1\,1\,1}$ 0 0 1 1 1 + 0 1 1 1 0 1 0 0 0 1 $\boxed{1}$ 1 1 1 1 1 1 1 1 1 ЗнR_1=ЗнB	$\underline{1\,0\,0\,0\,1}$ 0 1 1 1 1 1 1 1 1 | 0 1 1 1 1 | $\boxed{1}$	Сложение с делителем, выровненным по младшим разрядам Сдвиг остатка влево Сложение с делителем, выровненным по старшим разрядам Знак первого остатка не совпадает со знаком делимого – деление корректно Формирование знака частного
2	$\overleftarrow{R}_1$ $[-B]_{пр}$ R_2	+ 1 1 1 1 1 $\underline{0\,1\,1\,1\,1}$ 0 1 1 1 0 ЗнR_2≠ЗнB	1 1 1 | 1 0 1 1 1 | 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
3	$\overleftarrow{R}_2$ $[B]_{доп}$ R_3	+ 1 1 1 0 1 $\underline{1\,0\,0\,0\,1}$ 0 1 1 1 0 ЗнR_3≠ЗнB	1 1 | 1 0 0 1 1 | 1 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
4	$\overleftarrow{R}_3$ $[B]_{доп}$ R_4	+ 1 1 1 0 1 $\underline{1\,0\,0\,0\,1}$ 0 1 1 1 0 ЗнR_4≠ЗнB	1 | 1 0 0 0 1 | 1 0 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
5	$\overleftarrow{R}_4$ $[B]_{доп}$ R_5	+ 1 1 1 0 1 $\underline{1\,0\,0\,0\,1}$ 0 1 1 1 0 ЗнR_5≠ЗнB	1 0 0 0 0 1 0 0 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного

В результате выполнения операции получено отрицательное частное

$[\boldsymbol{C}]_{\boldsymbol{доп}} = (1.0000)_2 = (-16)_{10}$ и положительный остаток

$[\boldsymbol{R}]_{\boldsymbol{пр}} = (0.1110)_2 = (+14)_{10}$, которые соответствуют истинным значениям $(-16) \times (-15) + 14 = 254$.

В данном примере на 3-ем, 4-ом и 5-ом шагах выполняется сложение с отрицательным делителем, так как остаток, полученный на предыдущем шаге, положителен, но в результате сдвига влево его знак оказался искаженным. Для того, чтобы знак остатка при сдвиге влево не искажался, может быть использован модифицированный код (см. следующий пример).

в) Делимое отрицательное (***A*** < 0), делитель положительный (***B*** > 0):

1	2	3	4	5
0	$[A]^{м}_{доп}$	1 1 1 0 0 0	0 0 0 1 0	Делимое
1	$[B]^{м}_{доп}$ R_1' $\overleftarrow{R}_1'$ $[B]^{м}_{пр}$ R_1	$\underline{0\,0\,0\,0\,0\,0}$ 1 1 1 0 0 0 + 1 1 0 0 0 1 $\underline{0\,0\,1\,1\,1\,1}$ $\boxed{0\,0}$ 0 0 0 0 0 0 0 0 0 0 ЗнR_1=ЗнB	$\underline{0\,1\,1\,1\,1}$ 1 0 0 0 1 0 0 0 1 | 0 0 0 0 1 | $\boxed{1}$	Сложение с делителем, выровненным по младшим разрядам Сдвиг остатка влево Сложение с делителем, выровненным по старшим разрядам Знак первого остатка не совпадает со знаком делимого – деление корректно. Формирование знака частного
2	$\overleftarrow{R}_1$ $[-B]^{м}_{доп}$ R_2	+ 0 0 0 0 0 0 $\underline{1\,1\,0\,0\,0\,1}$ 1 1 0 0 0 1 ЗнR_2≠ЗнB	0 0 1 | 1 0 0 0 1 | 1 $\boxed{0}$	Сдвиг остатка влево Вычитание делителя Формирование цифры частного
3	$\overleftarrow{R}_2$ $[B]^{м}_{пр}$ R_3	+ 1 0 0 0 1 0 $\underline{0\,0\,1\,1\,1\,1}$ 1 1 0 0 0 1 ЗнR_3≠ЗнB	0 1 | 1 0 0 0 1 | 1 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
4	$\overleftarrow{R}_3$ $[B]^{м}_{пр}$ R_4	+ 1 0 0 0 1 0 $\underline{0\,0\,1\,1\,1\,1}$ 1 1 0 0 0 1 ЗнR_4≠ЗнB	1 | 1 0 0 0 1 | 1 0 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного
5	$\overleftarrow{R}_4$ $[B]^{м}_{пр}$ R_5	+ 1 0 0 0 1 1 $\underline{0\,0\,1\,1\,1\,1}$ 1 1 0 0 1 0 ЗнR_5≠ЗнB	1 0 0 0 0 1 0 0 0 $\boxed{0}$	Сдвиг остатка влево Сложение с делителем Формирование цифры частного

В результате выполнения операции получено отрицательное частное $[\boldsymbol{C}]_{доп} = (1.0000)_2 = (-16)_{10}$ и отрицательный остаток $[\boldsymbol{R}]_{доп} = (1.0010)_2$, $[\boldsymbol{R}]_{пр} = (-14)_{10}$, которые соответствуют истинным значениям $(-16) \times 15 + (-14) = -254$.

г) Оба операнда отрицательные (***A*** < 0, ***B*** < 0):

1	2	3	4	5
0	$[A]_{доп}$	1 1 0 0 0	0 0 0 1 0	Делимое
1	$[\overleftarrow{A}]_{доп}$ $[-B]_{пр}$ R_1	$+$ 1 0 0 0 0 $\underline{0\ 1\ 1\ 1\ 1}$ $\boxed{1}$ 1 1 1 1	0 0 1 0 | 0	Сдвиг делимого влево Вычитание делителя Знак первого остатка совпадает со знаком делимого – деление некорректно

8. ОПЕРАЦИИ СЛОЖЕНИЯ И ВЫЧИТАНИЯ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

8.2. Основные положения

Рассмотрим основные нюансы операции сложения на примере для десятичных чисел.

$$\boldsymbol{A} = \underbrace{\mathbf{0,945}}_{\text{мантисса МА}} \cdot \mathbf{10}^{\overset{\mathbf{P}_A}{\mathbf{3}}}$$

$$\boldsymbol{B} = \underbrace{\mathbf{0,847}}_{\text{МВ}} \cdot \mathbf{10}^{\overset{\mathbf{P}_B}{\mathbf{2}}}$$

Первоначально необходимо привести операнды к одному порядку. Для этой цели мантиссу числа с меньшим порядком сдвигают вправо на один разряд (величина разности порядка).

$$\boldsymbol{B}' = \underbrace{\mathbf{0,0847}}_{\text{МВ}} \cdot \mathbf{10}^{\overset{\mathbf{P}_B}{\mathbf{3}}} = \mathbf{P}_A$$

$$\begin{array}{r} M_A = 0{,}945 \\ \underline{M'_B = 0{,}0847} \\ M_C = M_A + M_B = 1{,}0297 \end{array}$$

В результате сложения получается ненормализованный результат (сумма мантисс больше 1, при сложении мантисс происходит переполнение формата).

Для приведения результата к нормализованному виду необходимо сдвинуть его мантиссу вправо и увеличить порядок на 1. Таки образом результат сложения $C = 0{,}10297 \cdot 10^4$.

Таким образом, операция сложения двоичных чисел с плавающей запятой реализуется последовательностью этапов:

1) сравнение порядков;
2) выравнивание (уравнивание) порядков;
3) сложение мантисс;
4) нормализация результата.

Обязательными из этих этапов являются 1-й и 3-й, а 2-й и 4-й этапы могут быть опущены.

Этап 1

Сравнение порядков реализуется путем их вычитания. При этом в целях однозначности вычитание производится таким образом, что уменьшаемым всегда является порядок первого операнда (P_A), а вычитаемым – порядок второго операнда (P_B). При использовании смещенных порядков (характеристик) осуществляется их вычитание как беззнаковых целых чисел. В соответствии с этим о знаке полученной разности можно судить по наличию или отсутствию заёма в старший разряд из-за пределов формата. При получении отрицательной разности характеристик (наличие заёма в старший разряд при вычитании) результат будет представлен в беззнаковом дополнительном коде. Подобный результат переводится в прямой код.

Этап 2

Этот этап пропускается, если на предыдущем этапе получена нулевая разность характеристик (порядки операндов одинаковы). Выравнивание порядков реализуется путем сдвига мантиссы операнда с меньшим порядком вправо на число разрядов, равное модулю разности порядков. В общем плане понятие разряда зависит от основания порядка. Если в качестве основания используется S=2 (форматы *Ф2* и *Ф3*), то разряд является двоичным. Если основание порядка S=16 (формат *Ф1*), то разряд является шестнадцатеричным.

Например, если модуль разности порядков равен 2, то сдвиг мантиссы вправо в формате *Ф1* производится на 8 двоичных разрядов. Выбор мантиссы сдвигаемого операнда определяется знаком разности характеристик, полученным на первом этапе. При отрицательной разности сдвигается мантисса первого операнда ($М_A$), при положительной разности – мантисса второго операнда ($М_B$).

Замечание. При любых операциях с мантиссами в арифметическом блоке FPU предварительно производится восстановление скрытых (старших) разрядов (для форматов *Ф2* и Ф3). Сдвиг мантиссы операнда с меньшим порядком вправо может приводить к потере младших значащих разрядов, что сказывается на уменьшении точности полученного результата.

При большом значении модуля разности порядков мантисса операнда с меньшим порядком может при сдвиге вправо полностью выйти за пределы формата (обратиться в 0). В этом случае операция сложения может досрочно завершаться путем присвоения результату значения операнда с большим порядком.

Этап 3

При реализации этого этапа необходимо учитывать, что мантиссы чисел с плавающей запятой, независимо от их знака, представляются в **прямом** коде.

При сложении операндов с одинаковыми знаками осуществляется сложение их мантисс в прямом коде, и в дальнейшем знаку результата присваивается знак одного из операндов (для однозначности – первого). Единственный дополнительный момент, который может возникнуть при сложении мантисс в прямых кодах – это переполнение разрядной сетки мантиссы. При беззнаковом сложении мантисс это переполнение фиксируется наличием переноса из старшего разряда. При знаковом сложении переполнение может быть зафиксировано при получении знака суммы, отличного от знаков операндов (предполагается, что при любых знаках мантисс их сложение производится как для положительных чисел). Возникшее при сложении мантисс переполнение устраняется на последнем шаге.

Сложение мантисс с разными знаками фактически реализуется их прямым вычитанием. В принципе, вычитание может приводиться к сложению с дополнительным кодом вычитаемого. В отношении того, мантиссу какого операнда используют в качестве уменьшаемого при вычитании или представляют в прямом коде при сложении, могут использоваться различные подходы (варианты):

1) в качестве уменьшаемого используется мантисса положительного операнда;
2) в качестве уменьшаемого используется мантисса первого операнда;
3) в качестве уменьшаемого используется мантисса операнда с меньшим порядком.

Кроме того, различные варианты выполнения сложения (вычитания) мантисс с разными знаками могут иметь место в связи с использованием явного или неявного представления знаков операндов. Принцип реализации сложения мантисс с разными знаками определяет и принцип получения знака результата. Независимо от используемого принципа, отрицательный результат сложения или вычитания мантисс, получаемый в дополнительном коде, переводится в прямой код.

Замечание. При сложении мантисс чисел с разными знаками мантисса результата может содержать старшие нули, т.е. результат может оказаться ненормализованным.

Этап 4

Этот этап имеет место, если при сложении мантисс получается ненормализованный (денормализованный) результат. Будем различать два вида денормализации:

- денормализация влево;
- денормализация вправо.

Первый случай денормализации может иметь место только при одинаковых знаках операндов. Этот случай распознается по наличию переполнения при сложении мантисс. Нормализация такого результата осуществляется сдвигом мантиссы вправо и увеличением порядка (характеристики) на 1. При сдвиге мантиссы в старший разряд вносится 1. Для формата *Ф1* старшая двоичная тетрада мантиссы после сдвига принимает значение 0001. Если порядок большего операнда, используемый в качестве порядка результата, имеет максимальное значение, то его увеличение на 1 приводит к особому случаю переполнения порядка.

Денормализация вправо означает наличие старших нулей в мантиссе результата. Для формата *Ф1* денормализованный вправо результат должен содержать в старших разрядах мантиссы как минимум четыре нуля. Нормализация такого результата осуществляется сдвигом мантиссы влево до удаления всех старших нулей и соответствующим изменением порядка. Порядок (характеристика) уменьшается на значение числа разрядов (двоичных или шестнадцатеричных), на которые производится сдвиг влево мантиссы. При уменьшении порядка может иметь место особый случай исчезновения порядка (в терминологии фирмы Intel эта ситуация называется *антипереполнением*). Исключением из общих правил нормализации является получение нулевой мантиссы результата при сложении операндов с разными знаками. В этом случае результат не нормализуется и представляется кодом полного нуля (во всех разрядах числа – 0).

Операция вычитания сводится к сложению путем предварительного изменения знака второго операнда на противоположный.

8.3. ЗАДАНИЕ 5
СЛОЖЕНИЕ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

1. Заданные числа ***A*** и ***B*** представить в форме с плавающей запятой в разрядных сетках форматов *Ф1* и *Ф2* с укороченной мантиссой (12 двоичных разрядов).
   Примечание: общее число разрядов в формате – 20.
2. Выполнить операцию сложения заданных чисел со следующими комбинациями знаков операндов: “++”, “+–”, “–+” в разрядных сетках форматов *Ф1* и *Ф2*.
3. Результаты представить в форматах операндов, перевести в десятичную систему счисления и проверить их правильность.
4. Определить абсолютную и относительную погрешности результатов и обосновать их причину.
5. Сравнить погрешности результатов аналогичных операций для форматов Ф*1* и Ф*2* и объяснить причины их сходства или различия.
   Варианты задания приведены в табл. 6 Приложения 1.

Пример операции сложения.

***A* = 0,3; *B* = 15,8.**

1. Формат *Ф1* (число разрядов мантиссы *m* =12).

$A = (0,3)_{10} = \underbrace{(0,4(C))_{16}}_{MA} \cdot 16^0$

0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0	1	1	0	1
0	1					7	8												19

1 округление к ближайшему

$B = (15,8)_{10} = (F,(C))_{16} = \underbrace{(0,F(C))_{16}}_{MB} \cdot 16^1$

0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1	0	1
0	1					7	8												19

1 округление к ближайшему

1) X_A = _1 0 0 0 0 0 0
$\underline{X_B = 1\ 0\ 0\ 0\ 0\ 0\ 1}$
$(X_A\text{-}X_B)_{доп.}$ = 1 1 1 1 1 1 1

$(X_A\text{-}X_B)$. = -1; $X_C = X_B = 1$.

а) Оба операнда положительные (*A*>0, *B*>0):

2,3) $\overset{4}{\rightarrow}$ M_A = . 0 0 0 0 0 1 0 0 1 1 0 0
M_B = + . 1 1 1 1 1 1 0 0 1 1 0 1
M_C = 1. 0 0 0 0 0 0 0 1 1 0 0 1

Результат сложения денормализован влево.

4) $\overset{4}{\overrightarrow{\mathrm{M}_C}} = .\,0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1$

Т.к. выполнен сдвиг мантиссы влево, то характеристику результата нужно увеличить на 1 ($X_C = X_C + 1 = 2$).

C

0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1
0	1					7	8											19	

$C^* = \mathrm{M}_C \cdot 16^{\mathrm{P}c} = (0{,}101)_{16} \cdot 16^2 = (10{,}1)_{16} = 16{,}0625.$

Несмотря не то, что оба операнда за счет округления были представлены с избытком, результат получился представленным с недостатком. Этот факт можно объяснить потерей значащих младших разрядов сначала у первого операнда при выравнивании порядков, а затем и у результата при его нормализации.

В принципе, погрешность полученного результата можно объяснить следующими факторами:

• неточным представлением операндов;

• потерей значащих разрядов мантиссы одного из операндов при уравнивании порядков;

• потерей значащих разрядов мантиссы результата при его нормализации сдвигом мантиссы вправо.

$\Delta C = C_\mathrm{T} - C^* = 16{,}1 - 16{,}0625 = 0{,}0375,$

где ΔC –абсолютная погрешность;

C_T –точное значение;

C^* - приближенное значение.

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{0{,}0375}{16{,}1}\right| .\ 100\% = 0{,}22\%,$$

где δC – относительная погрешность.

б) $A<0$, $B>0$.

Сложение мантисс будем проводить их прямым вычитанием. В качестве уменьшаемого используем мантиссу положительного операнда (B);

$$\begin{array}{l} 2{,}3)\ \mathrm{M}_B = .\,1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1 \\ \underline{\overset{4}{\overrightarrow{\mathrm{M}_{\bar{A}}}} = .\,0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0} \\ \mathrm{M}_C = .\,1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1 \end{array}$$

Результат сложения нормализован.

C

0	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	1
0	1					7	8											19	

$C^* = \mathrm{M}_C \cdot 16^{\mathrm{P}c} = (0{,}\mathrm{F}81)_{16} \cdot 16^1 = (\mathrm{F}{,}81)_{16} = 15{,}5039.$

$\Delta C = C_\mathrm{T} - C^* = 15{,}5 - 15{,}5039 = -\,0{,}0039,$

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{0{,}0039}{15{,}5}\right| . \ 100\% = 0{,}025\%.$$

2. Формат *Ф2*.

$A = (0{,}3)_{10} = (0{,}4(C))_{16} = (0{,}100110011001100)_2 \cdot 2^{-1}$

0	0	1	1	1	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1
19	18							11	10										0

$B = (15{,}8)_{10} = (F,(C))_{16} = (0{,}111111001100)_2 \cdot 2^4$

0	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	1	1	0	0
19	18							11	10										0

1) X_A = _0 1 1 1 1 1 1 1
$\underline{X_B = 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0}$
$(X_A\text{-}X_B)_{\text{доп.}}$ = 1 1 1 1 1 0 1 1
$(X_A\text{-}X_B)_.$ = -5; $X_C = X_B = 4$.

а) Оба операнда положительные (A>0, B>0):

2,3) $\overset{\xrightarrow{5}}{M_A}$ = . 0 0 0 0 0 1 0 0 1 1 0 0
+
M_B = . 1 1 1 1 1 1 0 0 1 1 0 0
M_C = 1 . 0 0 0 0 0 0 0 1 1 0 0 0

Результат сложения денормализован влево.

4) $\overset{\xrightarrow{1}}{M_C}$ = . 1 0 0 0 0 0 0 0 1 1 0 0

Т.к. выполнен сдвиг мантиссы влево, то характеристику результата нужно увеличить на 1 ($X_C = X_C$ +1 = 5).

C

0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0
19	18							11	10										0

$C^* = M_C \cdot 2^{Pc} = (0{,}10000000111)_2 \cdot 2^5 = (10000{,}000111)_2 = 16{,}109375.$

$\Delta C = C_T - C^* = 16{,}1 – 16{,}\ 109375 = 0{,}009375,$

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{0{,}009375}{16{,}1}\right| . \ 100\% = 0{,}058\%.$$

б) A<0, B>0.

2,3) M_B = . 1 1 1 1 1 1 0 0 1 1 0 0
$\underline{\overset{\xrightarrow{5}}{M_A} = .\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0}$
M_C = . 1 1 1 1 1 0 0 0 0 0 0 0

C	0	1	0	0	0	0	1	0	0	1	1	1	1	0	0	0	0	0	0	0
	19	18							11	10										0

$C^* = \mathrm{M}_C \cdot 2^{\mathrm{P}c} = (0{,}11111_2 \cdot 2^4 = (1111{,}1)_2 = 15{,}5.$

$\Delta C = C_{\mathrm{T}} - C^* = 15{,}5 – 15{,}5 = 0.$

В формате *Ф2* результаты получились точнее из-за того, что операнды представлены точнее и при нормализации результата сдвиг производился на один двоичный разряд, а не на четыре.

9. УМНОЖЕНИЕ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

9.1. Основные положения

Операция умножения сводится к сложению порядков как целых чисел и перемножению мантисс сомножителей как дробных чисел с фиксированной запятой.

В связи с тем, что, независимо от знака, мантиссы чисел с плавающей запятой представляются в прямом коде, то для умножения мантисс целесообразно использовать метод умножения в прямых кодах. Т.к. порядки сомножителей представлены не в явном виде, а со смещением (в виде характеристик), то для получения характеристики результата (произведения) необходимо выполнить следующие операции над характеристиками операндов:

$$\begin{array}{r} X_A = \mathrm{P}_A + d \\ + \; \underline{X_B = \mathrm{P}_B + d} \end{array} \qquad C = A \cdot B$$

$$X_A + X_B = \underbrace{\mathrm{P}_A + \mathrm{P}_B}_{\mathrm{P}_C} + 2d.$$

$$X_C = \mathrm{P}_C + d;$$

$$\boldsymbol{X_C = X_A + X_B - d.}$$

Замечание: будем считать, что в операции умножения участвуют именно нормализованные числа с плавающей запятой.

При использовании правильных дробей для представления мантисс сомножителей результат перемножения мантисс может находиться в следующих пределах:

$$1/S \le \mathrm{M}_A < 1$$

$$1/S \le \mathrm{M}_B < 1$$

$$\mathbf{1/S^2 \le M_C = M_A \cdot M_B < 1}$$

Из полученного соотношения следует, что результат умножения двух нормализованных мантисс может оказаться денормализованным вправо, но максимум – на одну цифру.

Действительно:

$S = 2 \;\rightarrow\; \mathrm{M}_{C\min} = (0{,}01)_2;$

$S = 16 \rightarrow \; \mathrm{M}_{C\min} = (0{,}01)_{16} = (0{,}00000001)_2.$

Умножение чисел с плавающей точкой в стандарте IEEE-754 имеет ряд дополнительных нюансов.

9.2. Особые случаи при выполнении операции умножения

К ним относятся *переполнение* и исчезновение (*антипереполнение*) порядка.

1) Переполнение порядка имеет место при умножении очень больших сомножителей.

2) Антипереполнение порядка имеет место при очень маленьких (близких к нулю) сомножителях.

В принципе, наличие особого случая можно распознать на начальном этапе операции, связанном с формированием предварительной характеристики произведения. Так, например, если сумма характеристик операндов меньше величины смещения, то можно, не перемножая мантиссы, фиксировать особый случай исчезновения порядка.

Другой крайний случай можно зафиксировать, если после вычитания смещения из суммы характеристик операндов полученная таким образом характеристика произведения будет превышать максимально возможное значение, т.е. выходить за пределы формата характеристики.

Следует иметь в виду, что после перемножении мантисс операндов может понадобиться нормализация результата сдвигом влево и уменьшение порядка произведения на 1.

9.3. Методы ускорения операции умножения

В связи с тем, что операция умножения является достаточно массовой, по крайней мере при решении научно-технических задач (по статистике доля операций умножения в программах научно-технического профиля составляет порядка 5%), значительное внимание разработчиков ЭВМ уделяется различным способам ускорения операции умножения. Эти способы ускорения принято разделять на два класса (вида):

- *аппаратные* (схемные);
- *логические* (алгоритмические).

Схемные методы сводятся к построению быстродействующих схем суммирования, а также схем, называемых *матричными умножителями*. Матричный умножитель 8×8 позволяет выполнить умножение байтных сомножителей за один такт.

Логические методы ускорения основаны на одновременном анализе нескольких разрядов множителя и выполнении соответствующих действии над суммой частных произведений (СЧП), в зависимости от анализируемых комбинаций. В практике построения схем АЛУ в основном используются методы ускоренного умножения на 2 и 4 разряда множителя.

9.3.1. Ускоренное умножение на 2 разряда множителя

Для определенности будем считать, что реализация метода осуществляется с использованием способа умножения от младших разрядов множителя со сдвигом СЧП вправо.

Идея метода состоит в анализе пары младших разрядов множителя на каждом шаге умножения. После выполнения действий над СЧП, соответствующих анализируемой паре, производится сдвиг СЧП и множителя на 2 разряда вправо.

Возможны следующие комбинации пары разрядов:

00 –производится только сдвиг вправо на 2 разряда, СЧП сохраняется;

01 – производится сложение СЧП с множимым и последующий сдвиг на 2 разряда вправо;

10 – СЧП складывается с удвоенным множимым (сложение с удвоенным множимым соответствует сложению со сдвинутым на 1 разряд влево множимым), множимое подается на вход сумматора не прямо, а с перекосом влево;

11 – производится вычитание множимого из СЧП с последующим сдвигом на 2 разряда вправо. При этом используют так называемую корректирующую единицу, которую необходимо учитывать при умножении на следующую пару разрядов множителя путем сложения единицы с этой парой.

$11_2 = 100_2 - 01_2$ (в данной паре умножение производится на (-01_2), которое реализуется сложением СЧП с $(-A)$).

Другим способом реализации является вычитание множимого из старших разрядов СЧП.

В связи с необходимостью учета возможной добавки к очередной паре разрядов множителя в схему умножителя обычно вводится дополнительный бит для фиксации так называемого признака коррекции, поэтому выполняемое на каждом шаге действие над СЧП определяется не только парой младших разрядов множителя, но и значением этого признака. С учетом признака коррекции выполняемые действия можно свести к следующей таблице:

Пара младших разрядов множителя	Признак коррекции для этой пары	Действия, выполняемые над СЧП	Признак коррекции для следующей пары
00	0	-	0
01	0	$+A$	0
10	0	$+2A$	0
11	0	$-A$	1
00	1	$+A$	0
01	1	$+2A$	0
10	1	$-A$	1
11	1	-	1

Особенности реализации метода

• В связи с тем, что выполняемые над СЧП действия включают в себя как сложение, так и вычитание множимого, СЧП необходимо рассматривать как знаковое число (применительно к перемножению мантисс операндов с плавающей запятой).

• Поскольку одной из добавок к СЧП является $2A$, для представления которого понадобится один дополнительный старший разряд, необходимо для представления результата сложения расширить СЧП на 2 старших разряда (для сохранения переноса, который может иметь место при сложении с удвоенным множимым). С учетом же знакового представления СЧП для явного отображения знака понадобится еще один дополнительный старший разряд.

• В связи со знаковым представлением СЧП, его сдвиг вправо для корректного представления выполняется как арифметический. Это означает, что в освобождающиеся при сдвиге старшие разряды производится копирование знакового разряда СЧП.

• Если после завершения умножения сохраняется единичное значение признака коррекции, необходимо выполнить дополнительный шаг, на котором к СЧП прибавляется множимое (как для пары (01)), после чего **сдвиг не выполняется**.

Пример (умножение в формате *Ф2*).

Для примера ограничимся 8 разрядами мантиссы.

$A = (84{,}5)_{10} = 0{,}10101001 \cdot 2^7$;

0	1	0	0	0	0	1	1	1	0	1	0	1	0	0	1
15	14							7	6						0

$B = (45{,}75)_{10} = 0,\ 10110111 \cdot 2^6$.

0	1	0	0	0	0	1	1	0	0	1	1	0	1	1	1
15	14							7	6						0

Перемножаются только мантиссы, знаки формируются отдельно.

$\mathrm{Sign}C = \mathrm{Sign}A \oplus \mathrm{Sign}\,B$.

$X_A = \mathrm{P}_A + d$; $X_B = \mathrm{P}_B + d$;

$X_C = X_A + X_B - d$;

$\mathrm{P}_C + d = \underbrace{\mathbf{P}_A + \boldsymbol{d} + \mathbf{P}_B}_{\mathrm{P}_C} + d - d.$

$$
\begin{array}{rl}
\boldsymbol{X_A} = & 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1 \\
+\ \underline{\boldsymbol{X_B}} = & \underline{1\ 0\ 0\ 0\ 0\ 1\ 1\ 0} \\
\boldsymbol{X_A + X_B} = & 1\,0\ 0\ 0\ 0\ 1\ 1\ 0\ 1 \\
-\ \boldsymbol{d} = & \underline{1\ 0\ 0\ 0\ 0\ 0\ 0\ 0} \\
\boldsymbol{X_C} = & 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1
\end{array}
$$

$\mathrm{P}_C = 13$.

N	Операнды	СЧП (старшие разряды)											В/СЧП (младшие разряды)								Признак коррекции
0	СЧП	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	**1**	**1**	0
1	$[-M_A]_{доп}$	1	1	1	0	1	0	1	0	1	1	1								$-M_A$	1
	СЧП	1	1	1	0	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	
	СЧП→2	1	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	1	**0**	**1**	
2	$2A$	0	0	1	0	1	0	1	0	0	1	0								$2M_A$	0
	СЧП	0	0	1	0	0	1	0	0	1	1	1	1	1	1	0	1	1	0	1	
	СЧП→2	0	0	0	0	1	0	0	1	0	0	1	1	1	1	1	1	0	**1**	**1**	
3	$[-M_A]_{доп}$	1	1	1	0	1	0	1	0	1	1	1								$-M_A$	1
	СЧП	1	1	1	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	1	
	СЧП→2	1	1	1	1	1	1	0	1	0	0	0	0	0	1	1	1	1	**1**	**0**	
4	$[-M_A]_{доп}$	1	1	1	0	1	0	1	0	1	1	1								$- M_A$	1
	СЧП	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0	
	СЧП→2	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	
5	M_A	0	0	0	1	0	1	0	1	0	0	1									
	СЧП	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	1	1	1	1	
	$\leftarrow M_C$	0	0	0	1	1	1	1	0	0	0	1	1	0	0	1	1	1	1	0	

$X_C = X_C - 1.$

$C^* = (0.11110001)_2 \cdot 2^{12} = (111100010000)_2 = 3856.$

0	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1
15	14							7	6						0

$C_T = 3855{,}875.$

$\Delta C = C_T - C^* = 3855{,}875 - 3856 = - 0{,}125,$

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{-0{,}125}{3855{,}875}\right| \cdot 100\% = 0{,}0032\%.$$

9.3.2. Метод ускоренного умножения на 4 разряда множителя

Предполагается, что умножение реализуется, начиная от младших разрядов множителя со сдвигом СЧП вправо. На каждом шаге умножения анализируется младшая тетрада множителя, и, в зависимости от ее значения, выполняются соответствующие операции над текущим СЧП. Шаг завершается сдвигом СЧП и множителя на 4 разряда вправо. По аналогии с предыдущим методом сдвиг СЧП реализуется как арифметический (с учетом знака текущего СЧП).

Простейшая реализация метода умножения на 4 разряда сводится к разделению тетрады множителя на две пары и формированию соответствующей добавки (частного произведения) к СЧП для каждой из пар отдельно. При этом может иметь место случай формирования внутреннего признака коррекции для старшей пары тетрады при соответствующем значении младшей пары и возможным наличием внешнего признака коррекции, сформированного при умножении на предыдущую тетраду множителя.

При формировании частного произведения для старшей пары тетрады необходимо учитывать, что она имеет вес $(100)_2 = 4$ по сравнению с младшей парой тетрады. Таким образом, в общем случае формируется два ненулевых частных произведения, которые могут быть как положительными, так и отрицательными и должны быть прибавлены к текущему значению СЧП. В соответствии с этим, максимального ускорения при схемной реализации этого метода можно достичь путем использования трехвходового сумматора.

Таблица действий над СЧП:

Младшая тетрада множителя	Частные произведения	Признак коррекции для следующей тетрады
0000	$0A + 0A$	0
0001	$0A + 1A$	0
0010	$0A + 2A$	0
0011	$4A – A$	0
0100	$4A + 0A$	0
0101	$4A + A$	0
0110	$4A + 2A$	0
0111	$8A – A$	0
1000	$8A + 0A$	0
1001	$8A + A$	0
1010	$8A + 2A$	0
1011	$-4A – A$	1
1100	$-4A + 0A$	1
1101	$-4A + A$	1
1110	$-4A + 2A$	1
1111	$0A - A$	1

Особенности реализации метода

- Для получения положительных кратных множимого ($2A$, $4A$, $8A$) производится его сдвиг влево на 1, 2 или 3 разряда соответственно. Схемно подача кратных множимого реализуется путем «косой» передачи из регистра множимого на вход сумматора. При формировании кратных отрицательного множителя ($-2A$, $-4A$) осуществляется сдвиг отрицательного множимого, представленного в дополнительном коде, влево на 1 или 2 разряда соответственно.
- Для корректной реализации метода n-разрядный сумматор (n – разрядность мантисс сомножителей) необходимо расширить на 5 старших разрядов. Три из них необходимы для представления увосьмеренного множимого ($8A$), один – для сохранения возможного переноса при сложении и еще один – для явного представления знака СЧП.

• Остальные особенности такие же, как и у метода умножения на 2 разряда множителя.

Пример (умножение в формате *Ф1*).

$A = 84{,}5 = (1010100{,}1)_2 = (54{,}8)_{16} = (0{,}548)\cdot 16^2$;

0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0
0	1					7	8								15

$B = 45{,}75 = (101101{,}11)_2 = (2D{,}C)_{16} = (0{,}2DC)_{16}\cdot 16^2$.

0	1	0	0	0	0	1	0	0	0	1	0	1	1	0	1
0	1					7	8							15	

$$
\begin{array}{rl}
\boldsymbol{X_A} & 1\;0\;0\;0\;0\;1\;0 \\
+ & \\
\boldsymbol{X_B} & \underline{1\;0\;0\;0\;0\;1\;0} \\
\boldsymbol{X_A+X_B} & 1\;0\;0\;0\;0\;1\;0\;0 \\
- & \\
\boldsymbol{d} & \underline{1\;0\;0\;0\;0\;0\;0} \\
\boldsymbol{X_C} & 1\;0\;0\;0\;1\;0\;0
\end{array}
$$

N	Операнды	СЧП (старшие разряды)													В/СЧП (младшие разряды)								Признак коррекции
0	СЧП	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**1**	**1**	**0**	**1**	1
	M_A	0	0	0	0	0	0	1	0	1	0	1	0	0					$-4M_A$		M_A		
	$[-4M_A]_{доп}$	1	1	1	1	0	1	0	1	1	0	0	0	0									
1	СЧП	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	0
	СЧП→4	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	**0**	**0**	**1**	**0**	
	$4M_A$	0	0	0	0	1	0	1	0	1	0	0	0	0					$4M_A$		$-M_A$		
	$[-M_A]_{доп}$	1	1	1	1	1	1	0	1	0	1	1	0	0									
2	СЧП	0	0	0	0	0	1	1	1	0	1	1	0	0	0	1	0	0	0	0	1	0	
	СЧП→4	0	0	0	0	0	0	0	0	0	1	1	1	0	1	1	0	0	0	1	0	0	
	4←СЧП	0	0	0	0	0	1	1	1	0	1	1	0	0									

$C^* = (0{,}EC)_{16}\cdot 16^3 = (EC0)_{16} = 3776$.

$C_T = 3855{,}875$.

$\Delta C = C_T - C^* = 3855{,}875 - 3776 = 79{,}875$,

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{79{,}875}{3855{,}875}\right| \cdot 100\% = 2{,}0715\%.$$

Погрешности результатов вызваны неточным представлением операндов. В формате *Ф2* операнды представлены точнее и погрешность меньше.

9.4. ЗАДАНИЕ 6
УМНОЖЕНИЕ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

1. Заданные числа ***A*** (множимое) и ***B*** (множитель) представить в форматах *Ф*1 и *Ф2* с укороченной мантиссой (12 двоичных разрядов). Метод округления выбирается произвольно.

Примечание: общее число разрядов в формате – 20.

2. Выполнить операцию умножения операндов в формате *Ф1*, используя метод ускоренного умножения мантисс на два разряда множителя.
3. Выполнить операцию умножения операндов в формате *Ф2*, используя метод ускоренного умножения мантисс на четыре разряда множителя.
4. Результаты представить в форматах операндов, перевести в десятичную систему счисления и проверить их правильность.
5. Определить абсолютную и относительную погрешности результатов и обосновать их причину.
6. Сравнить погрешности результатов аналогичных операций для форматов *Ф1* и *Ф2* и объяснить причины их сходства или различия.

   Варианты задания приведены в табл. 7 Приложения 1.

10. ДЕЛЕНИЕ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

10.1. Основные положения

Операция деления сводится к вычитанию порядка делителя из порядка делимого как целых чисел и делению мантиссы делимого на мантиссу делителя как дробных чисел с фиксированной запятой. Для получения характеристики частного необходимо к разности характеристик делимого и делителя прибавить величину смещения:

$$\begin{array}{r} X_A = \mathrm{P}_A + d \\ - \; \underline{X_B = \mathrm{P}_B + d} \\ X_A - X_B = \underbrace{\mathrm{P}_A - \mathrm{P}_B}_{\mathrm{P}_C}. \end{array}$$

$$X_C = \mathrm{P}_C + d;$$

$$\boldsymbol{X_C = X_A - X_B + d.}$$

10.2. Особенности деления мантисс

2. В связи с тем, что мантиссы операндов представляются в прямом коде, независимо от их знака, при делении мантисс используется метод деления в прямых кодах.

3. В связи с тем, что мантиссы операндов (делимого и делителя) представляются в одинаковых форматах, предварительного сдвига мантиссы делимого влево (как при знаковом делении целых чисел) не требуется.

4. В операцию деления вступают нормализованные операнды. Для формата *Ф*1 это означает, что при ненормализованных операндах в начале операции производится их нормализация (сдвигом мантиссы влево).

5. На начальном шаге операции деления мантисс производится так называемое *пробное вычитание,* при котором из мантиссы делимого вычитается мантисса делителя. Если результат пробного вычитания не отрицателен, то для мантисс, представленных правильными дробями, это означает, что мантисса частного $M_C \geq 1$, т.е. выходит за границу диапазона дробных мантисс.

При использовании основания порядка, равного 2 (формат *Ф2*), подобный случай можно трактовать как получение старшей цифры мантиссы в виде единицы целой части.

Действительно, при делении двух нормализованных мантисс, представленных правильными дробями, имеют место следующие соотношения:

для делимого $1/2 \leq M_A < 1$;

для делителя $1/2 \leq M_B < 1$;

$$\mathbf{1/2 < M_C = M_A - M_B < 2 \quad (*).}$$

Из этого соотношения следует, что при S=2 в целой части мантиссы частного может находиться максимум одна единица.

Таким образом, при положительном результате пробного вычитания старшая цифра мантиссы частного равна 1, и для перенесения этой единицы из целой части мантиссы в дробную необходимо увеличить порядок (характеристику) частного на 1 (неявная нормализация результата сдвигом вправо).

При получении отрицательного результата пробного вычитания цифра частного, вырабатываемая как инверсия знакового разряда остатка, равна нулю, причем этот 0 находится в целой части мантиссы частного. Исходя из соотношения (***),** старшая цифра дробной части мантиссы частного, вырабатываемая на следующем шаге, должна быть равна 1.

При положительном результате пробного вычитания для выработки остальных цифр частного выполняется еще (n - 1)-й шаг (n – разрядность мантисс операндов).

При отрицательном результате пробного вычитания после него для получения всех n цифр мантиссы частного выполняется n шагов.

Замечание: для выполнения округления результата по методу к ближайшему необходимо получить дополнительную (n - 1)-ю цифру мантиссы частного.

При выполнении пробного вычитания в формате *Ф1* (S = 16) в случае положительного первого остатка определение числа двоичных цифр, оказывающихся в целой части мантиссы частного, достаточно затруднительно. Действительно имеют место следующие соотношения:

для делимого $1/16 \le M_A < 1$;

для делителя $1/16 \le M_B < 1$;

$$\mathbf{1/16 < M_C = M_A - M_B < 16},$$

т. е., число цифр в целой части может быть от 1 до 4). Поэтому рекомендуется на подобную ситуацию реагировать следующим образом:

а) выполнить восстановление мантиссы делимого путем сложения первого остатка с мантиссой делителя;

б) сдвинуть мантиссу делимого на 4 разряда (одну шестнадцатеричную цифру) вправо, при этом к порядку частного прибавляется единица.

Подобное действие можно расценивать как принудительную денормализацию делимого вправо. Чтобы не потерять точность при сдвиге мантиссы делимого вправо, ее младшая тетрада сохраняется и в дальнейшем при сдвиге остатка влево поочередно замещает его освобождающиеся младшие разряды. После денормализации выполняется пробное вычитание, результат которого будет обязательно отрицателен.

6. В отличие от реализации целочисленного деления, где формируемые на каждом шаге цифры частного помещались в младшие разряды делимого/остатка, освобождаемые при сдвиге влево, при делении мантисс необходимо использовать отдельный регистр для записи цифр мантиссы частного, формируемых на каждом шаге деления. Этот регистр обычно реализуется как сдвигающий влево, в связи с чем цифры частного вносятся в его младший разряд.

7. Знак частного формируется отдельным действием как сумма по модулю 2 знаковых разрядов операндов.

8. Остаток от деления как результат операции не сохраняется.

9. При выполнении операций могут иметь место следующие особые случаи, фиксируемые в блоке FPU с помощью специальных флагов:

- деление на 0 (мантисса делителя равна 0);
- переполнение порядка (делимое большое, делитель маленький);
- исчезновение порядка (делимое маленькое, а делитель большой).

10.3. ЗАДАНИЕ 7
ДЕЛЕНИЕ ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

1. Заданные числа ***A*** (делимое) и ***B*** (делитель) представить в форматах *Ф1* и *Ф2* с укороченной мантиссой (8 двоичных разрядов). Метод округления выбирается произвольно.

Примечание: общее число разрядов в формате – 16.

2. Выполнить операцию деления операндов в формате *Ф1*.
3. В случае положительного результата «пробного» вычитания сохранить младшую тетраду.
4. Выполнить операцию деления операндов в формате *Ф2*.
5. Результаты представить в форматах операндов, перевести в десятичную систему счисления и проверить их правильность.
6. Определить абсолютную и относительную погрешности результатов и обосновать их причину.

   Варианты задания приведены в табл. 8 Приложения 1.

Пример 1. Деление в формате *Ф2*.

$A = 7{,}7 = (111.10110011)_2 = (0.11110110)_2 \cdot 2^3$

$B = 0{,}028 = (0.0000011100101)_2 = (0.11100101)_2 \cdot 2^{-5}$

$X_C = X_A - X_B + d$

$$d + \mathrm{P}_C = \frac{\underline{\mathbf{P_A + \mathit{d} - P_B - d}}}{\mathbf{P}_C} + d$$

$X_C = 3 - (-5) + 128 = 136$

$\mathrm{P}_C = 8$

N шага	Действие	Делимое	Частное
0	M_A $[-M_B]_{доп}$ R_0	**0 1 1 1 1 0 1 1 0** **1 0 0 0 1 1 0 1 1** **0 0 0 0 1 0 0 0 1**	**0 0 0 0 0 0 0 0** **0 0 0 0 0 0 0 1**
1	$\leftarrow R_0$ $[-M_B]_{доп}$ R_1	**0 0 0 1 0 0 0 1 0** **1 0 0 0 1 1 0 1 1** **1 0 0 1 1 1 1 0 1**	**0 0 0 0 0 0 1 0** **0 0 0 0 0 0 1 0**
2	$\leftarrow R_1$ $M_{B\ пр}$ R_2	**0 0 1 1 1 1 0 1 0** **0 1 1 1 0 0 1 0 1** **1 0 1 0 1 1 1 1 1**	**0 0 0 0 0 1 0 0** **0 0 0 0 0 1 0 0**
3	$\leftarrow R_2$ $M_{B\ пр}$ R_3	**0 1 0 1 1 1 1 1 0** **0 1 1 1 0 0 1 0 1** **1 1 0 1 0 0 0 1 1**	**0 0 0 0 1 0 0 0** **0 0 0 0 1 0 0 0**
4	$\leftarrow R_3$ $B_{пр}$	**1 0 1 0 0 0 1 1 0** **0 1 1 1 0 0 1 0 1**	**0 0 0 1 0 0 0 0**

	R_4	0 0 0 1 0 1 0 1 1	0 0 0 1 0 0 0 1
5	$\leftarrow R_4$ $[-M_B]_{доп}$ R_5	0 0 1 0 1 0 1 1 0 1 0 0 0 1 1 0 1 1 1 0 1 1 1 0 0 0 1	0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0
6	$\leftarrow R_5$ $M_{B\ пр}$ R_6	0 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 1 1 1 0 0 0 1 1 1	0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0
7	$\leftarrow R_6$ $M_{B\ пр}$ R_7 $M_C \rightarrow$	1 1 0 0 0 1 1 1 0 0 1 1 1 0 0 1 0 1 0 0 1 1 1 0 0 1 1	1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 1

$C^* = (0.10001001)_2 \cdot 2^9 = (100010010)_2 = 274.$

$C^T = 275$ (точное значение).

$$\Delta C = C_T - C^* = 275 - 274 = 1,$$

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{1}{275}\right| \cdot 100\% = 0{,}36\%.$$

Погрешность вызвана неточным представлением операндов.

Пример 2. Деление в формате *Ф1*.

N шага	Действие	Делимое	Частное
0	M_A $[-M_B]_{доп}$ R_0 $M_A \rightarrow 4$ $[-M_B]_{доп}$ R_0	0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 1	0 0 0 0 0 0 0 0 **R_0>0** 1 0 1 1 0 0 0 0 1 0 1 1 0 0 0 0
1	$\leftarrow R_0$ $M_{B\ пр}$ R_1	1 0 0 1 0 1 0 1 1 0 0 1 1 1 0 0 1 0 1 1 0 0 1 1 1 0 1	0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0
2	$\leftarrow R_1$ $M_{B\ пр}$ R_2	1 0 0 1 1 1 0 1 0 0 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 0 0	1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0
3	$\leftarrow R_2$ $B_{пр}$ R_3	1 0 1 0 1 1 0 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1	1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0

4	$\leftarrow R_3$ $M_{B\ \text{пр}}$ R_4	**1 1 0 0 1 0 1 1 1** **0 0 1 1 1 0 0 1 0** **0 0 0 0 0 1 0 0 1**	**0 0 0 0 0 0 0 0** **0 0 0 0 0 0 0 1**
5	$\leftarrow R_4$ $[-M_B]_{\text{доп}}$ R_5	**0 0 0 0 1 0 0 1 0** **1 1 0 0 0 1 1 1 0** **1 1 0 1 0 0 0 0 0**	**0 0 0 0 0 0 1 0** **0 0 0 0 0 0 1 0**
6	$\leftarrow R_5$ $M_{B\ \text{пр}}$ R_6	**1 0 1 0 0 0 0 0 0** **0 0 1 1 1 0 0 1 0** **1 1 0 1 1 0 0 1 0**	**0 0 0 0 0 1 0 0** **0 0 0 0 0 1 0 0**
7	$\leftarrow R_6$ $M_{B\ \text{пр}}$ R_7	**1 0 1 1 0 0 1 0 0** **0 0 1 1 1 0 0 1 0** **1 1 1 0 1 0 1 1 0**	**0 0 0 0 1 0 0 0** **0 0 0 0 1 0 0 0**
8	$\leftarrow R_7$ $M_{B\ \text{пр}}$ R_8	**1 1 0 1 0 1 1 0 0** **0 0 1 1 1 0 0 1 0** **0 0 0 0 1 1 1 1 0**	**0 0 0 1 0 0 0 0** **0 0 0 1 0 0 0 1**

$C^* = (1,1)_{16}\ 16^3 = (110)_{16} = 272$

$\Delta C = C_{\text{Т}} - C^* = 275 - 272 = 3,$

$$\delta C = \left|\frac{\Delta C}{C_T}\right| \cdot 100\% = \left|\frac{1}{275}\right| .\ 100\% = 1{,}09\%.$$

Погрешность вызвана неточным представлением операндов и она больше, чем при делении в формате *Ф2*.

11. ОСНОВЫ ДЕСЯТИЧНОЙ АРИФМЕТИКИ

11.1. Десятичные числа

Десятичные числа используются в ЭВМ на этапе ввода исходных данных или этапе вывода результатов для поддержки удобного интерфейса с пользователем.

В ЭВМ десятичные числа представляются в двоично-кодированной форме, в связи с чем их достаточно часто называют *двоично-десятичными числами*. (см. Раздел 2).

В современных ЭВМ для кодирования десятичных цифр используется код **8421**, который характеризуется естественным представлением десятичных цифр с помощью двоичной тетрады:

0 – 0000	5 – 0101
1 – 0001	6 – 0110
2 – 0010	7 – 0111
3 – 0011	8 – 1000
4 – 0100	9 – 1001

Десятичные числа принято представлять в ЭВМ в одном из двух форматов:

- упакованном (PACK);
- неупакованном (UNPACK).

В *упакованном формате* в каждом байте числа содержатся две десятичные цифры. Обычно упакованный формат называют **BCD-форматом** (Binary Coded Decimal).

В *неупакованном (распакованном) формате* в каждом байте числа представляется только одна десятичная цифра. Типичным примером неупакованного формата является представление десятичных цифр в коде ASCII. В дальнейшем представление десятичных чисел в неупакованном формате будем называть **ASCII-форматом**. В данном формате для представления десятичной цифры отводится младшая тетрада байта (младший полубайт), старшая тетрада байта (старший полубайт) принимает стандартное значение $(0011)_2 = (3)_{10}$.

На этапе ввода числовых данных и вывода числовых результатов десятичные числа представляются в ASCII-формате. Их преобразование в BCD-формат может быть реализовано либо на аппаратном, либо на программном уровне.

В системе команд процессоров семейства Intel X86 отсутствуют команды преобразования десятичных чисел из одного формата в другой, значит, это преобразование приходится реализовать на программном уровне.

В связи с тем, что в любой ЭВМ поддерживаются как двоичные, так и десятичные числа, вполне естественно выглядит реализация не только двоичной, но и десятичной арифметики для обработки десятичных чисел.

Аппаратная поддержка преобразования десятичного числа в упакованном формате в двоичное число с плавающей запятой в расширенном формате (80 бит), а также обратное преобразование реализуются специальными командами **FPU**.

FBLD – Load Binary coded Decimal (загрузка с преобразованием). По этой команде осуществляется пересылка десятичного числа в BCD-формате из памяти во внутренний регистр данных FPU (вершину стека ST(0)) с преобразованием из десятичной системы счисления в двоичную форму числа с плавающей запятой, представленного в расширенном формате (внутренний формат FPU).

Внутренняя регистровая память данных блока FPU включает в себя восемь 80-битных регистров, организованных в виде так называемого ***кольцевого регистрового стека.***

Обратное преобразование из двоичной системы в десятичную реализуется командой FBSTP – Store BCD Integer and Pop. По этой команде осуществляется пересылка данных из вершины регистрового стека FPU в память с предварительным преобразованием двоичного числа с плавающей запятой, упакованной в десятичный формат. Выбираемый операнд удаляется из регистрового стека (POP).

Этими командами поддерживается 80-битный BCD-формат для представления упакованного десятичного числа. В данном формате представлено 18 десятичных цифр, которые занимают 72 младших разряда формата. Старший байт формата используется для представления знака, который занимает старший бит. Десятичное число предполагается целым. Как таковые, десятичные числа в BCD и ASCII – форматах имеют аппаратную поддержку в процессорах Intel на уровне так называемых команд десятичной BCD и ASCII – коррекции. К ним относят:

BCD-коррекция:

- DAA (Decimal Adjust after Addition. BCD –коррекция после сложения);
- DAS (Decimal Adjust after Subtraction, BCD –коррекция после вычитания).

ASCII-коррекция:

- AAA (ASCII Adjust after Addition, корректирует сумму двух неупакованных BCD-чисел);
- AAS (ASCII Adjust after Subtraction, корректирует разность двух неупакованных BCD-чисел);
- AAM (ASCII Adjust after Multiply, корректирует результат умножения двух неупакованных BCD-чисел;
- AAD (ASCII Adjust before Division, корректирует неупакованное BCD-число в регистре *ax* перед делением).

Команды коррекции используются в машинном коде для преобразования результатов соответствующей операции в корректную десятичную форму. При этом предшествующая арифметическая операция, выполняемая командами ADD, SUB, MUL, предполагает использование двоичной арифметики применительно к десятичным числам. Все команды коррекции поддерживают минимальный формат данных – 1 байт.

Для операции деления команда коррекции AAD в машинном коде предшествует команде DIV.

В принципе, можно считать, что в процессорах Intel десятичная арифметика поддерживается аппаратно не как таковая, а реализуется на основе двоичной арифметики с последующей коррекцией результата.

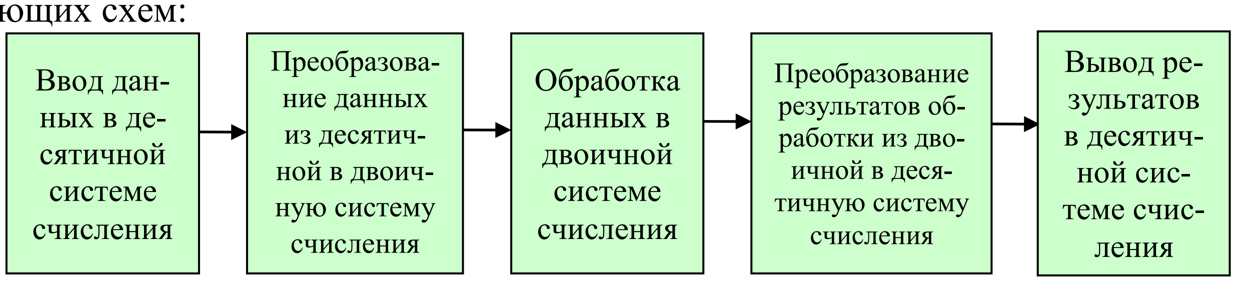
11.2. Обоснование необходимости использования в ЭВМ десятичной арифметики наряду с двоичной

В принципе, преобразование из исходной десятичной системы в двоичную при вводе и обратное преобразование при выводе может быть реализовано либо на аппаратном, либо на программном уровне. Использование аппаратного уровня предполагает наличие в системе команд процессора специальных команд, выполняющих это преобразование. Так, например, в вычислительной системе IBM/370 в системе команд имелись команды CDB (Convert Decimal to Binary) и обратная CBD.

В PC и, в частности, на базе процессоров Intel преобразование реализуется на программном уровне.

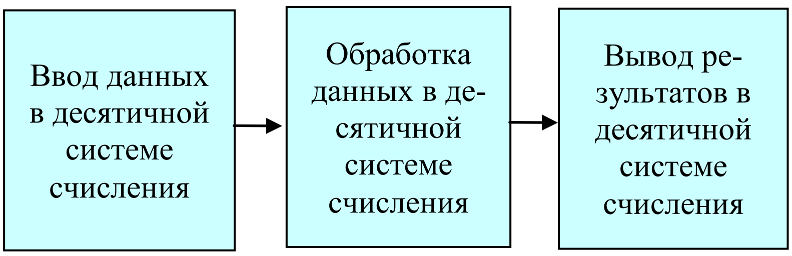
Для обработки данных в ЭВМ возможно применение одной из двух следующих схем:

1)



двоичная арифметика

2)



десятичная арифметика

Первую схему обработки данных целесообразно применять при сравнительно небольших объемах перерабатываемых данных и при достаточно большом объеме вычислений (операций), приходящихся на единицу данных. Подобным свойством обладают так называемые *научно-технические задачи*.

Вторую схему обработки целесообразно применять при больших объемах обрабатываемых данных и небольшом объеме вычислений, приходящихся на каждую единицу данных. Подобным свойством обладают так называемые *экономические (коммерческие) задачи*.

Аппаратная поддержка второй схемы обработки данных в ЭВМ предполагает наличие в составе процессора ***десятичного АЛУ.*** Наличием подобных АЛУ характеризуются ЭВМ, относящиеся к классам Mainframe. В ПК и рабочих станциях на базе практически всех современных процессоров десятичное АЛУ отсутствует.

Аппаратная поддержка десятичной арифметики связана с использованием десятичного АЛУ, естественно наряду с двоичным, в составе центрального процессора ЭВМ.

В принципе в большинстве ЭВМ принято разделять АЛУ на ряд функциональных блоков по типу обрабатываемых данных. В связи с этим используются отдельные АЛУ для:

1. целочисленной арифметики;
2. арифметики с ПЗ;
3. десятичной арифметики.

Кроме того, могут быть выделены в АЛУ отдельные блоки для выполнения логических операций и операций сдвигов.

11.3. Основные идеи десятичного беззнакового сложения

При реализации десятичной арифметики необходимо иметь в виду, что десятичные числа представляются в двоично-кодированной форме с помощью BCD-чисел (Binary Coded Decimal). В BCD-числах каждая десятичная цифра представляется с помощью соответствующей двоичной тетрады с естественными весами разрядов (код 8421):

0 – 0000

………..

9 - 1001

В принципе в ЭВМ используется одна из двух форм представления десятичных чисел:

1. упакованный формат
2. неупакованный формат.

В п. 1 в каждом байте числа содержится по две десятичных цифры, в п.2 - одна.

Частным случаем п.2 может служить ASCII - код десятичных чисел, в котором собственно цифра занимает правую тетраду байтов, в то время как в левой части код (0011).

В подавляющем большинстве десятичная арифметика реализуется в отношении упакованного формата.

11.3.1 Сложение десятичных чисел

При сложении двух десятичных цифр $c_i = a_i + b_i$ на двоичном сумматоре могут иметь место следующие случаи:

1. $c_i \leq 9$ (1001);

Пример.

$a_i = 3$ (0011); $b_i = 5$ (0101);

$$\begin{array}{rl} a_i = & 0\ 0\ 1\ 1\ (3) \\ + b_i = & \underline{0\ 1\ 0\ 1\ (5)} \\ c_i = & 1\ 0\ 0\ 0\ (8) \end{array}$$ - результат корректный.

2. $9 < c_i \leq 15$;

Пример.

$a_i = 5$ (0101); $b_i = 9$ (1001);

$$\begin{array}{rl} a_i = & 0\ 1\ 0\ 1\ (5) \\ + b_i = & \underline{1\ 0\ 0\ 1\ (9)} \\ c_i = & 1\ 1\ 1\ 0\ (14)\ ? \end{array}$$

$c_i = 1\ 0\ 1\ 0\ 0$ – правильный результат

3. $c_i > 15$;

Пример.

$a_i = 9$ (1001); $b_i = 7$ (0111);

$$\begin{array}{rl} a_i = & 1\ 0\ 0\ 1\ (9) \\ + b_i = & \underline{0\ 1\ 1\ 1\ (7)} \\ c_i = & 1\ 0\ 0\ 0\ 0\ (16). \end{array}$$

$c_i = 1\ 0\ 1\ 1\ 0$ – правильный результат

В примерах 2 и 3 получились некорректные результаты в связи с тем, что вес переноса для двоичной системы счисления равен 16, а при десятичном сложении он должен быть равен 10.

При простейшем подходе к сложению BCD - чисел с использованием двоичного сумматора необходимо выполнить потетрадную коррекцию полученного результата в следующих случаях:

1. Результат превышает значение десятичной цифры, т.е. от 10 до 15.
2. При сложении имеет место перенос из старшего разряда тетрады.

Коррекция результата осуществляется прибавлением корректирующего кода (КК) 6 (0110) в соответствующие тетрады.

11.3.2 Операция беззнакового десятичного сложения

Пример 1:

$A = 547$; $B = 456$;

$C = A + B = 1003$.

A = 0000 0101 0100 0111

\+ B = 0000 0100 0101 0110

C' = 0000 1001 1001 1101

\+ KK' = 0000 0000 0000 0110

\+ C'' = 0000 1001 1010 0011

KK'' = 0000 0000 0110 0000

C''' = 0000 1010 0000 0011

\+ KK''' = 0000 0110 0000 0000

C = 0001 0000 0000 0011 = 1003.

Из рассмотренного примера вполне очевидно вытекает недостаток приведённого метода коррекции, который состоит в возможном многократном повторении коррекции для нескольких последующих тетрад. В пределе - для всех.

Подобный метод коррекции реализован в процессорах фирмы Intel и осуществляется при выполнении команды DAA (Decimal Ajust after Addition). Свободным от этого недостатка является сложение BCD - чисел с представлением одного из операндов в "коде с избытком 6".

\+ A = 0000 0101 0100 0111

Изб.6 = 0110 0110 0110 0110

[A]иб = 0110 1011 1010 1101

\+ B = 0000 0100 0101 0110

\- C' = 0111 0000 0000 0011

KK = 0110 0000 0000 0000

C = 0001 0000 0000 0011 = 1003_{10}.

При использовании этого метода может понадобиться максимум однократная коррекция результата. Коррекция осуществляется для тех тетрад, из которых при сложении не возникало переноса, следовательно, остался избыток 6 и состоит в вычитании избытка из этих тетрад.

Реализация коррекции с помощью вычитания требует дополнительных затрат оборудования в десятичном АЛУ, т.к. наряду с сумматором требуется и вычитатель. В целях сокращения затрат оборудования можно использовать другой подход к коррекции, заменяя вычитание 6 сложением с дополнением 6:

[6] = 0110;

[6]доп = 1010 = [10].

При коррекции сложения на момент этой коррекции осуществляется блокировка межтетрадных переносов в сумматоре.

Пример.

A = 231; B = 768.

A = 0010 0011 0001;

B = 0111 0110 1000.

$$
\begin{array}{rcl}
 A & = & 0010\ 0011\ 0001 \\
+\ \text{Изб.6} & = & \underline{0110\ 0110\ 0110} \\
 [A]\text{иб} & = & 1000\ 1001\ 0111 \\
+\ B & = & \underline{0111\ 0110\ 1000} \\
 C & = & 1111\ 1111\ 1111 \\
+\ KK & = & \underline{1010\ 1010\ 1010} \\
 C & = & 1001\ 1001\ 1001 = 999_{10}.
\end{array}
$$

11.3.3 Операция знакового десятичного сложения

Как правило для кодирования знака десятичного числа используется специальный байт или полубайт (тетрада) в зависимости от используемого формата (неупакованный или упакованный) и от конкретной реализации ЭВМ. При этом знак может помещаться либо перед старшими цифровыми разрядами(слева), либо после младшего разряда числа (справа). Знак, как таковой, в операции сложения не участвует. Естественно, что двоично-десятичные числа независимо от их знака представлены в прямом коде.

Знаковое сложение осуществляется по следующим правилам:

1. Числа с одинаковыми знаками складываются точно также, как и беззнаковые, и результату присваивается знак одного из операндов (как правило первого);
2. Числа с разными знаками складываются с использованием дополнительного кода одного из операндов (как правило, отрицательного);
3. Дополнительный код BCD - числа содержит избыток 6 в каждой тетраде, следовательно, сложение чисел с разными знаками не требует добавления избытка 6 к одному из операндов.

Пример:

A = - 231; B = 768.

$$
\begin{array}{rcl}
 A & = & 0010\ 0011\ 0001 \\
 [A]_{\text{дк}} & = & 1101\ 1100\ 1111 \\
+\ B & = & \underline{0111\ 0110\ 1000} \\
 C & = & 0101\ 0011\ 0111 = 537.
\end{array}
$$

Наличие переноса из старшего разряда при сложении двух BCD - чисел с разными знаками свидетельствует о том, что результат сложения положителен.

Пример:

A = 231; B = - 768.

B = 0111 0110 1000

$[B]_{дк}$ = 1000 1001 1000

\+ A = 0010 0011 0001

C' = 1010 1100 1001

$КК$ = 1010 1010 1010

$[C]_{дк}$ = 0100 0110 0011

$[C]_{пк}$ = 1011 1001 1101

$КК$ = 1010 1010 1010

C = 0101 0011 0111 = - 537.

Отсутствие переноса из старшего разряда при сложении чисел с разными знаками свидетельствует о том, что результат отрицателен, и, следовательно, представлен в дополнительном коде.

Для уменьшения количества действий, связанных с коррекцией дополнительного кода результата и дальнейшего его преобразования в прямой код можно использовать следующий подход: результат сложения C' не корректируется, а преобразовывается из дополнительного кода в прямой. В прямом коде результата осуществляется коррекция для тех тетрад, из которых при сложении был перенос.

Пример:

A = -587; B = 329

A = 0101 1000 0111

$[A]_{дк}$= 1010 0111 1001

\+

B = 0011 0010 1001

$[C']_{дк}$= 1101 1010 0010

$[C']_{.пк}$=0010 | 0101 | 1110

\+

$КК$ = 0000 | 0000 | 1010

C = 0010 0101 1000 = 258.

Операция десятичного вычитания сводится к десятичному сложения путем предварительного изменения знака второго операнда (вычитаемого) на противоположный.

11.4. ЗАДАНИЕ 8
ЗНАКОВОЕ СЛОЖЕНИЕ ДЕСЯТИЧНЫХ ЧИСЕЛ

1. Выполнить операцию десятичного сложения заданных чисел ***A*** и ***B*** с использованием двух подходов к коррекции результата:

   а) сложение прямых BCD-кодов операндов с последовательной коррекцией результата;

   б) использование кода с избытком 6 для одного из операндов с последующей коррекцией результата;

2. Выполнить операцию десятичного сложения чисел –***A*** и ***B***.
3. Выполнить операцию десятичного сложения чисел ***A*** и **–*B***.

Результаты каждого примера представить в десятичной системе и проверить их корректность.

Формат результата и операндов выбрать таким, чтобы при сложении с одинаковыми знаками не возникал особый случай переполнения формата.

Исходные операнды ***A*** и ***B*** выбираются из табл. 3 Приложения 1 из двух последовательных вариантов, начиная с заданного. Например, для варианта 1 задания старшие разряды операндов берутся из варианта 1 (***A***=78, ***B***=47), а младшие из варианта 2 (***A***=46, ***B***=81) и задание выполняется для операндов ***A***=7846, ***B***=4781.

Вопросы и задачи по теме «Представление чисел в ЭВМ»

1. Обосновать использование в ЭВМ двоичной системы счисления. (6 баллов)

2. В чем состоит отличие интерпретации двоичного числа как знакового или беззнакового? (2 балла) Как проявляется это отличие на аппаратном уровне? (4 балла)

3. Почему в ЭВМ для представления знаковых чисел используется дополнительный код, а не прямой? (3 балла)

4. Изобразить классическую схему обработки данных в ЭВМ (3 балла) При решении задач какого класса эта схема оказывается целесообразной (1 балл) и почему? (3 балла)

5. Определить диапазон представления целых чисел со знаком в формате, содержащем 9 двоичных разрядов. (2 балла) Представить в этом формате число (-**87**). (3 балла)

65. Представить отрицательное число с порядком, равным -5, и максимальной мантиссой в формате *Ф3*. (2 балла) Определить значение этого числа. (2 балла)

7. Представить положительное число с максимальным порядком минимальной мантиссой в формате *Ф2*. (1 балл) Определить значение этого числа. (2 балла)

8. Представить положительное число с минимальным порядком максимальной мантиссой в формате *Ф*1. (1 балл) Определить значение этого числа. (2 балла)

9. Определить диапазон и точность представления чисел с плавающей запятой в формате, использующем семь двоичных разрядов для представления мантиссы и шесть разрядов для представления порядка. Порядок представляется со знаком в дополнительном коде. Основание порядка: S=8. (7 баллов)

10. Представить число ***A*=-22,3125** в форме с плавающей запятой в форматах *Ф1*, *Ф2* и *Ф3*. (за каждый 4 балла)

11. Определить значение числа с плавающей запятой по его шестнадцатеричному представлению **51A7B000** в форматах *Ф1*, *Ф2* и *Ф3*. (за каждый 3 балла)

12. Определить количество двоичных разрядов, необходимых для представления целых чисел со знаком, содержащих не более семи десятичных цифр. (3 балла) Определить фактический диапазон представления целых чисел со знаком в полученном формате. (2 балла)

13. Определить формат разрядной сетки для представления чисел с плавающей запятой из условия обеспечения заданного диапазона представления: $|A| < 10^{70}$ и заданной точности (максимальная относительная погрешность не превосходит величины 10^{-11}), считая, что порядок представляется со смещением, равным весу старшего разряда. Основание порядка: S=4. (8 баллов) Определить фактический диапазон представления чисел в полученном формате и фактическую точность их представления (относительную погрешность). (4 балла)

Вопросы и задачи по теме «Выполнение арифметических операций в ЭВМ»

1. Пояснить принципы установки флага CF в командах сложения - ADD (2) и вычитания – SUB (3). Как влияет установка CF на результат операции сложения и вычитания при беззнаковой интерпретации операции (3)?.
2. Выполнить операцию знакового сложения целых чисел $|A|=78$ и $|B|=50$ с одинаковыми знаками (2 примера) в байтном формате (4). Прокомментировать полученные результаты (3). Показать значения арифметических флагов для каждого из примеров (3). Дать беззнаковую интерпретацию для каждого из примеров (3).
3. Перечислить способы (схемы) реализации умножения в ЭВМ (3).
4. Выполнить операцию умножения целых чисел A=-5 (множимое) на B=10 (множитель), используя метод умножения в дополнительных кодах с коррекцией. Формат операндов выбрать самостоятельно (8). Дать обоснование используемых в данном примере методов коррекции (6).
5. Привести пример фиксации некорректности деления целых чисел по методу деления в дополнительных кодах для положительного делимого и делителя, равного -7. Формат операндов выбрать самостоятельно (4).
6. Выполнить операцию деления целых чисел (-26) на (-5), используя метод деления в дополнительных кодах. Формат операндов выбрать самостоятельно (10).
7. Привести пример операндов с плавающей запятой в формате *Ф1*, при сложении которых имеет место особый случай переполнения порядка. Порядок меньшего операнда не более 62 (4).
8. Привести пример операндов с плавающей запятой в формате *Ф2*, при уравнивании порядков которых в операции сложения мантисса одного из них целиком выходит за пределы разрядной сетки. Порядок меньшего операнда равен -28, а порядок большего операнда отрицателен (4).
9. Привести пример операндов в формате *Ф1*, при умножении которых имеет место особый случай переполнения порядка, если только результат умножения мантисс является нормализованным (6).
10. Привести пример операндов в формате *Ф2*, при делении которых имеет место особый случай исчезновения порядка. Порядок делимого равен 11 (4).
11. Привести пример операндов в формате *Ф1*, при делении которых имеет место особый случай переполнения порядка, только за счет увеличения порядка частного на начальном этапе деления мантисс (6).

Список литературы

1. Качинский М.В. Арифметические основы электронных вычислительных средств: учеб.-метод. Пособие / М.В. Качинский, В.Б. Клюс, А.Б. Давыдов. – Минск: БГУИР, 2014. – 64 с.: ил. ISBN 978-985-543-001-09.
2. Меннингер, К. История цифр. Числа, символы, слова. – М.: ЗАО Центрополиграф. 2011. - 543 с. ISBN 978592449787.
3. Цилькер Б. Я., Орлов С.А. Организация ЭВМ и систем. Учебник для вузов. — 2-е изд. — СПб.: Питер, 2011. — 688 с.:
ISBN 978-5-49807-862-5
4. Мейлахс А. Л. Практикум по математическим основам информатики. Метод. указания. Ч.1.:Системы счисления. Двоичная арифметика. Представление чисел в памяти ЭВМ. М.: Горная книга, 2012. - 63 с.
5. Жмакин А. П. Архитектура ЭВМ: 2-е изд., перераб. и доп.: учеб. пособие. — СПб.: БХВ- Петербург, 2010. — 352 с. — (Учебная литература для вузов) ISBN 978-5-9775-0550-5.
6. Андреева Е., Фалина И. Системы счисления и компьютерная арифметика. М. Бином, 2004. ISBN 5-94779-117-2.
7. Гашков, С. Б. Занимательная компьютерная арифметика: быстрые алгоритмы операций с числами и многочленами / С. Б. Гашков. - Москва : URSS, 2012. - 221 с. ISBN 978-5-397-02880-6
8. Таненбаум Э. Остин Т. Архитектура компьютера (6-е издание). СПб.: Питер, 2013. – 816 с.
9. Бурдинский И. Н. Системы счисления и арифметика ЭВМ : учеб. пособие / И. Н. Бурдинский. – Хабаровск : Изд-во Тихоокеан. гос. ун-та, 2008. – 79 с. ISBN 978-5-7389-0733-3.
10. Максимов Н.В., Партыка Т.Л., Попов И.И. Архитектура ЭВМ и вычислительных систем: 5-е изд., перераб. и доп. - М.:Форум, НИЦ ИНФРА-М, 2016. - 512 с.: 60x90 1/16. ISBN 978-5-91134-742-0.
11. Сайт «Все о системах счисления» http://numeration.ru/
12. Сайт "История развития вычислительной техники" –
http://istrasvvt.narod.ru/index.htm

ПРИЛОЖЕНИЕ 1

Таблица 1

№	*A*	*B*	№	*A*	*B*	№	*A*	*B*	№	*A*	*B*
1.	100	0,009	**31.**	850	0,105	**61.**	950	0,205	**91.**	570	0,225
2.	200	0,008	**32.**	1050	0,305	**62.**	1150	0,405	**92.**	1300	0,425
3.	300	0,007	**33.**	1250	0,505	**63.**	1990	0,011	**93.**	1925	0,041
4.	400	0,006	**34.**	1350	0,605	**64.**	125	0,022	**94.**	425	0,082
5.	500	0,005	**35.**	1450	0,705	**65.**	225	0,033	**95.**	275	0,063
6.	600	0,004	**36.**	1550	0,805	**66.**	325	0,044	**96.**	375	0,074
7.	700	0,003	**37.**	1650	0,905	**67.**	425	0,066	**97.**	475	0,056
8.	800	0,002	**38.**	1750	0,115	**68.**	525	0,077	**98.**	575	0,087
9.	900	0,001	**39.**	1850	0,125	**69.**	625	0,088	**99.**	675	0,038
10.	1000	0,09	**40.**	1950	0,135	**70.**	725	0,099	**100.**	775	0,049
11.	1100	0,08	**41.**	110	0,145	**71.**	825	0,111	**101.**	8725	0,131
12.	1200	0,07	**42.**	220	0,155	**72.**	925	0,222	**102.**	975	0,242
13.	1300	0,06	**43.**	330	0,165	**73.**	1025	0,333	**103.**	1075	0,373
14.	1400	0,05	**44.**	440	0,175	**74.**	1125	0,444	**104.**	1175	0,484
15.	1500	0,04	**45.**	550	0,185	**75.**	1225	0,555	**105.**	1275	0,515
16.	1600	0,03	**46.**	660	0,195	**76.**	1325	0,666	**106.**	1375	0,656
17.	1700	0,02	**47.**	770	0,205	**77.**	1425	0,777	**107.**	1475	0,737
18.	1800	0,01	**48.**	880	0,215	**78.**	1525	0,888	**108.**	1575	0,828
19.	1900	0,15	**49.**	990	0,225	**79.**	1625	0,999	**109.**	1675	0,929
20.	2000	0,25	**50.**	1110	0,235	**80.**	1725	0,12	**110.**	1775	0,17
21.	150	0,35	**51.**	1220	0,245	**81.**	1825	0,23	**111.**	1875	0,73
22.	250	0,45	**52.**	1330	0,255	**82.**	1925	0,34	**112.**	1975	0,84
23.	350	0,55	**53.**	1440	0,265	**83.**	1975	0,46	**113.**	1925	0,56
24.	450	0,65	**54.**	1550	0,275	**84.**	1875	0,56	**114.**	1825	0,76
25.	550	0,75	**55.**	1660	0,285	**85.**	1775	0,67	**115.**	1725	0,97
26.	650	0,85	**56.**	1770	0,295	**86.**	1675	0,78	**116.**	1625	0,38
27.	750	0,95	**57.**	1880	0,305	**87.**	1575	0,89	**117.**	1525	0,29
28.	670	0,123	**58.**	380	0,238	**88.**	560	0,076	**118.**	660	0,056
29.	460	0,064	**59.**	780	0,456	**89.**	870	0,231	**119.**	670	0,271
30.	780	0,172	**60.**	760	0,031	**90.**	890	0,654	**120.**	690	0,752

Таблица 2.

№	*R*	*S*	№	*R*	*S*	№	*R*	*S*
1.	41A40000	BC800000	**46.**	40B40000	BF3A0000	**91.**	40B40000	BF3A0000
2.	C1B80000	3C400000	**47.**	C1C50000	3E7B0000	**92.**	C1C50000	3E7B0000
3.	423E5000	BD200000	**48.**	41D60000	BD6C0000	**93.**	41D60000	BD6C0000
4.	C25C2000	3D100000	**49.**	C2E70000	3F9E0000	**94.**	C2E70000	3F9E0000
5.	4314DC00	BE300000	**50.**	42F8B000	BE5F0000	**95.**	42F8B000	BE5F0000
6.	C322FD00	3E500000	**51.**	C3A9C100	3D4C0000	**96.**	C3A9C100	3D4C0000
7.	419A0000	BF600000	**52.**	43BAD200	BFA40000	**97.**	43BAD200	BFA40000
8.	C1AB0000	3F700000	**53.**	C0CB0000	3EB60000	**98.**	C0CB0000	3EB60000
9.	424B3000	BC100000	**54.**	40DC0000	BDF90000	**99.**	40DC0000	BDF90000
10.	C23CF000	3C200000	**55.**	C1ED4C00	3FD10000	**100.**	C1ED4C00	3FD10000
11.	4312D500	BD400000	**56.**	41F68000	BEB30000	**101.**	41F68000	BEB30000
12.	C322E300	3D800000	**57.**	C29FA400	3DC50000	**102.**	C29FA400	3DC50000
13.	41FF0000	BE900000	**58.**	42AEC800	BFE70000	**103.**	42AEC800	BFE70000
14.	C1E08000	3EA00000	**59.**	C3BDE500	3EC80000	**104.**	C3BDE500	3EC80000
15.	426A7000	BFB00000	**60.**	43CC0700	BDD90000	**105.**	43CC0700	BDD90000
16.	C2A98800	3F800000	**61.**	C0EB0000	3FE10000	**106.**	C0EB0000	3FE10000
17.	43AAAA00	BCD00000	**62.**	40FA0000	BEF70000	**107.**	40FA0000	BEF70000
18.	C380F800	3C400000	**63.**	C119C000	3D780000	**108.**	C119C000	3D780000
19.	41FC0000	BD800000	**64.**	41288000	BFD70000	**109.**	41288000	BFD70000
20.	C1B10000	3D400000	**65.**	C237A000	3FA50000	**110.**	C237A000	3FA50000
21.	42ABC000	BEF00000	**66.**	42429000	BDB30000	**111.**	42429000	BDB30000
22.	C2FF0000	3EA00000	**67.**	C3598400	3FCB0000	**112.**	C3598400	3FCB0000
23.	43FA0000	BFD00000	**68.**	C0D40000	3E5A0000	**113.**	C0D40000	3E5A0000
24.	C11F0000	3EC00000	**69.**	436A7600	BDBC0000	**114.**	436A7600	BDBC0000
25.	C33A0000	3C200000	**70.**	C0780000	3F830000	**115.**	C0780000	3F830000
26.	41B90000	BE500000	**71.**	408C0000	BEA90000	**116.**	408C0000	BEA90000
27.	43AF0000	BEB00000	**72.**	C19DE000	3DF40000	**117.**	C19DE000	3DF40000
28.	43E40000	BED00000	**73.**	41AEC000	BEDF0000	**118.**	41AEC000	BEDF0000
29.	B51C0000	432ABE00	**74.**	C2BFD800	3F6B0000	**119.**	C2BFD800	3F6B0000
30.	C25A3000	3DD00000	**75.**	42D0E400	BEB90000	**120**	42D0E400	BEB90000
31.	417A7000	BE100000	**76.**	C311F200	3D570000	**121.**	C311F200	3D570000
32.	C33B8000	3FA00000	**77.**	43230100	BF910000	**122.**	43230100	BF910000
33.	40A30000	BCC00000	**78.**	40E50000	BE8C0000	**123.**	40E50000	BE8C0000
34.	C1F90000	3DF00000	**79.**	C1F61000	3DE50000	**124.**	C1F61000	3DE50000
35.	41CA0000	BD300000	**80.**	41E72000	BDC20000	**125.**	41E72000	BDC20000
36.	C26B6000	3E200000	**81.**	C2183000	3F350000	**126.**	C2183000	3F350000
37.	42597000	BE400000	**82.**	42294200	BE050000	**127.**	42294200	BE050000
38.	C320A600	3FF00000	**83.**	C33A5300	3D9B0000	**128.**	C33A5300	3D9B0000
39.	43619800	BF100000	**84.**	434B6480	BF8C0000	**129.**	434B6480	BF8C0000
40.	C0910000	3DA20000	**85.**	C03C0000	3E8A0000	**130.**	C03C0000	3E8A0000
41.	40AF0000	BFA10000	**86.**	41CF0000	BCD40000	**131.**	41CF0000	BCD40000
42.	C2F81000	3FB50000	**87.**	C0A65000	3CE70000	**132.**	C0A65000	3CE70000
43.	42BE6000	BEB20000	**88.**	43B70000	BCE70000	**133**	43B70000	BCE70000
44.	C3E34000	3EC60000	**89.**	C1C47000	3FF80000	**134.**	C1C47000	3FF80000
45.	43ED0000	BDC30000	**90.**	427C4B00	BEF60000	**135.**	427C4B00	BEF60000

Таблица 3

№	*A*	*B*	№	*A*	*B*	№	*A*	*B*	№	*A*	*B*
1.	78	41	**31.**	45	81	**61.**	45	70	**91.**	75	41
2.	81	43	**32.**	70	43	**62.**	61	44	**92.**	63	54
3.	100	26	**33.**	63	32	**63.**	92	25	**93.**	81	23
4.	73	49	**34.**	83	43	**64.**	103	17	**94.**	95	13
5.	51	62	**35.**	94	23	**65.**	72	51	**95.**	64	49
6.	65	53	**36.**	53	68	**66.**	90	35	**96.**	83	35
7.	107	15	**37.**	45	71	**67.**	56	57	**97.**	56	48
8.	56	61	**38.**	24	101	**68.**	80	23	**98.**	77	17
9.	61	24	**39.**	103	21	**69.**	35	77	**99.**	53	66
10.	42	80	**40.**	51	40	**70.**	83	37	**100.**	71	38
11.	102	19	**41.**	72	52	**71.**	62	59	**101.**	54	48
12.	37	88	**42.**	77	43	**72.**	49	63	**102.**	39	67
13.	31	91	**43.**	81	36	**73.**	53	55	**103.**	67	49
14.	109	17	**44.**	32	84	**74.**	93	31	**104.**	87	31
15.	54	66	**45.**	43	81	**75.**	37	70	**105.**	75	27
16.	54	67	**46.**	101	21	**76.**	62	52	**106.**	74	43
17.	36	89	**47.**	38	78	**77.**	65	45	**107.**	58	47
18.	51	69	**48.**	33	82	**78.**	49	58	**108.**	91	18
19.	41	76	**49.**	46	68	**79.**	68	50	**109.**	61	57
20.	27	93	**50.**	65	39	**80.**	71	53	**110.**	36	73
21.	70	47	**51.**	67	46	**81.**	57	62	**111.**	73	48
22.	85	34	**52.**	85	37	**82.**	109	15	**112.**	103	17
23.	92	28	**53.**	74	48	**83.**	49	66	**113.**	69	46
24.	56	61	**54.**	109	15	**84.**	92	28	**114.**	81	37
25.	34	83	**55.**	11	114	**85.**	76	38	**115.**	67	43
26.	67	51	**56.**	43	81	**86.**	53	71	**116.**	45	72
27.	68	54	**57.**	101	25	**87.**	43	81	**117.**	43	71
28.	32	95	**58.**	53	62	**88.**	35	64	**118.**	53	49
29.	75	36	**59.**	39	84	**89.**	77	21	**119.**	73	19
30.	48	48	**60.**	67	56	**90.**	18	70	**120.**	21	77

Таблица 4

№	A	B	№	A	B	№	A	B	№	A	B
1.	61	47	**31.**	117	14	**61.**	16	118	**91.**	61	125
2.	22	81	**32.**	19	101	**62.**	38	62	**92.**	78	52
3.	82	21	**33.**	119	20	**63.**	120	28	**93.**	119	18
4.	20	83	**34.**	46	63	**64.**	18	121	**94.**	21	111
5.	23	84	**35.**	102	17	**65.**	27	122	**95.**	71	103
6.	64	37	**36.**	123	26	**66.**	24	85	**96.**	41	57
7.	65	45	**37.**	67	50	**67.**	46	49	**97.**	36	41
8.	66	47	**38.**	36	86	**68.**	53	15	**98.**	37	21
9.	44	67	**39.**	29	124	**69.**	103	68	**99.**	93	46
10.	25	87	**40.**	51	45	**70.**	35	126	**100.**	34	117
11.	88	36	**41.**	104	11	**71.**	13	125	**101.**	23	95
12.	89	26	**42.**	43	69	**72.**	12	22	**102.**	43	72
13.	90	35	**43.**	106	21	**73.**	105	91	**103.**	113	41
14.	70	34	**44.**	12	107	**74.**	33	48	**104.**	73	28
15.	41	71	**45.**	52	63	**75.**	55	57	**105.**	75	31
16.	72	33	**46.**	92	27	**76.**	42	109	**106.**	41	93
17.	73	28	**47.**	13	108	**77.**	23	74	**107.**	73	43
18.	93	25	**48.**	44	56	**78.**	41	40	**108.**	53	50
19.	37	94	**49.**	14	170	**79.**	59	29	**109.**	93	21
20.	95	24	**50.**	75	42	**80.**	58	111	**110**	56	91
21.	15	96	**51.**	112	11	**81.**	17	97	**111.**	18	79
22.	76	32	**52.**	113	17	**82.**	35	50	**112.**	37	70
23.	38	77	**53.**	62	54	**83.**	48	43	**113.**	47	71
24.	78	40	**54.**	98	30	**84.**	60	115	**114.**	70	113
25.	31	79	**55.**	114	34	**85.**	31	80	**115.**	11	91
26.	18	99	**56.**	53	39	**86.**	39	41	**116.**	31	44
27.	19	100	**57.**	116	30	**87.**	125	94	**117.**	111	74
28.	52	61	**58.**	57	63	**88.**	24	37	**118.**	76	29
29.	34	85	**59.**	76	45	**89.**	91	76	**119.**	101	53
30.	20	98	**60.**	67	49	**90.**	51	63	**120.**	78	37

Таблица 5

№	*A*	*B*	№	*A*	*B*	№	*A*	*B*	№	*A*	*B*
1.	3338	33	**31.**	1571	23	**61.**	1716	26	**91.**	1162	35
2.	1682	24	**32.**	1536	22	**62.**	1410	15	**92.**	1110	41
3.	1318	19	**33.**	1303	20	**63.**	1449	21	**93.**	1390	51
4.	1382	18	**34.**	986	12	**64.**	1020	15	**94.**	1001	17
5.	1344	20	**35.**	1654	18	**65.**	1272	12	**95.**	1272	12
6.	1422	21	**36.**	2076	22	**66.**	1248	13	**96.**	1314	34
7.	1605	16	**37.**	964	12	**67.**	1152	16	**97.**	1217	26
8.	1630	17	**38.**	3328	33	**68.**	1904	28	**98.**	1715	37
9.	1834	26	**39.**	1182	12	**69.**	960	10	**99.**	851	12
10.	3072	31	**40.**	1145	17	**70.**	1920	30	**100**	1735	41
11.	2566	27	**41.**	2816	21	**71.**	940	10	**101.**	1105	15
12.	954	10	**42.**	1833	27	**72.**	1088	16	**102.**	1107	13
13.	982	15	**43.**	1436	19	**73.**	1380	15	**103.**	1138	25
14.	1916	26	**44.**	1644	23	**74.**	930	10	**104.**	703	20
15.	1804	25	**45.**	981	15	**75.**	960	12	**105.**	716	13
16.	1534	16	**46.**	1058	16	**76.**	1460	20	**106.**	1246	30
17.	1643	24	**47.**	974	11	**77.**	1080	12	**107.**	1178	22
18.	944	12	**48.**	1211	17	**78.**	1924	26	**108.**	1249	25
19.	1684	25	**49.**	1911	27	**79.**	1056	16	**109.**	1105	26
20.	1645	25	**50.**	1933	30	**80.**	938	14	**110.**	1003	15
21.	1146	17	**51.**	1654	18	**81.**	1540	22	**111.**	1150	23
22.	1271	18	**52.**	1022	15	**82.**	1634	19	**112.**	1345	21
23.	1228	17	**53.**	3076	33	**83.**	994	14	**113.**	1194	24
24.	1522	23	**54.**	1354	19	**84.**	924	11	**114.**	892	19
25.	1435	21	**55.**	1217	18	**85.**	1440	20	**115.**	1411	27
26.	1036	15	**56.**	1800	23	**86.**	1056	12	**116.**	1606	22
27.	1302	17	**57.**	1017	16	**87.**	1462	17	**117.**	1624	31
28.	3584	34	**58.**	2171	23	**88.**	1584	24	**118.**	915	25
29.	2367	25	**59.**	1885	16	**89.**	1944	18	**119.**	1047	23
30.	3106	29	**60.**	2043	19	**90.**	1242	27	**120.**	941	17

Таблица 6.

№	A	B	№	A	B	№	A	B	№	A	B
1.	16,32	7,14	**31.**	8,368	17,63	**61.**	103,4	161,8	**91.**	115,1	128,7
2.	161,4	94,61	**32.**	18,23	15,45	**62.**	0,632	0,594	**92.**	0,728	0,415
3.	5,891	2,964	**33.**	124,2	15,65	**63.**	8,524	9,435	**93.**	7,154	10,37
4.	140,25	84,3	**34.**	33,27	99,88	**64.**	115,7	170,1	**94.**	127,6	168,3
5.	118,3	252,5	**35.**	256,2	202,6	**65.**	10,18	9,343	**95.**	12,36	8,417
6.	241,5	34,81	**36.**	0,826	1,436	**66.**	14,68	13,77	**96.**	12,84	15,64
7.	9,87	22,68	**37.**	117,6	163,1	**67.**	17,67	15,89	**97.**	17,67	25,39
8.	16,385	8,964	**38.**	103,7	29,43	**68.**	0,342	0,299	**98.**	0,423	0,187
9.	83,42	59,82	**39.**	96,24	31,66	**69.**	176,1	97,86	**99.**	126,6	83,18
10.	30,84	11,45	**40.**	1,567	63,76	**70.**	130,2	128,6	**100.**	125,3	115,7
11.	5,134	9,354	**41.**	9,165	6,842	**71.**	25,62	21,93	**101.**	36,16	42,84
12.	0,632	8,287	**42.**	17,43	8,186	**72.**	0,821	0,797	**102.**	0,915	0,636
13.	25,25	46,55	**43.**	6,134	11,75	**73.**	12,34	15,07	**103.**	18,87	13,28
14.	33,42	32,66	**44.**	45,63	30,89	**74.**	198,3	189,5	**104.**	186,8	173,4
15.	21,68	11,41	**45.**	9,812	7,553	**75.**	276,2	201,2	**105.**	268,9	212,3
16.	62,76	103,5	**46.**	29,52	53,04	**76.**	0,245	0,786	**106.**	0,357	0,817
17.	256,8	244,6	**47.**	12,12	7,129	**77.**	234,4	228,9	**107.**	237,6	219,8
18.	9,415	16,53	**48.**	60,97	14,13	**78.**	67,54	72,34	**108.**	78, 84	82,39
19.	250,1	120,4	**49.**	120,3	152,6	**79.**	623,8	502,1	**109.**	54,7	62,8
20.	31,62	46,04	**50.**	0,464	1,321	**80.**	57,23	49,87	**110.**	35,13	41,76
21.	291,6	221,8	**51.**	0,725	0,484	**81.**	173,7	139,2	**111.**	164,9	140,3
22.	0,876	4,591	**52.**	67,18	62,25	**82.**	66,23	74,71	**112.**	71,31	65,62
23.	10,26	34,37	**53.**	231,2	260,8	**83.**	0,023	0,993	**113.**	0,047	0,874
24.	67,04	63,45	**54.**	3,735	15,82	**84.**	7,652	8,578	**114.**	8,731	7,963
25.	12,62	20,85	**55.**	5,475	9,732	**85.**	304,5	256,2	**115.**	301,6	265,1
26.	106,5	58,38	**56.**	346,7	273,9	**86.**	20,86	17,34	**116.**	21,67	18,43
27.	27,52	38,16	**57.**	56,65	64,86	**87.**	0,821	0,982	**117.**	0,962	0,734
28.	130,17	61,94	**58.**	0,782	128,2	**88.**	12,56	11,98	**118.**	12,72	14,89
29.	130,1	129,9	**59.**	62,84	58,92	**89.**	326,8	250,2	**119.**	236,7	180,3
30.	18,27	14,38	**60.**	124,2	146,3	**90.**	78,21	63,29	**120.**	93,31	73,21

Таблица 7.

№	*A*	*B*	№	*A*	*B*	№	*A*	*B*	№	*A*	*B*
1.	1,2	0,09	**31.**	3,6	0,042	**61.**	7,3	0,55	**91.**	6,7	0,36
2.	4,3	0,41	**32.**	9,6	0,074	**62.**	2,2	0,031	**92.**	3,1	0,045
3.	2,3	0,029	**33.**	9.1	0,01	**63.**	6,9	0,051	**93.**	7,3	0,072
4.	8,3	0,059	**34.**	2,5	0,015	**64.**	5,9	0,092	**94.**	6,3	0,084
5.	2,1	0,08	**35.**	3,8	0,043	**65.**	8,2	0,45	**95.**	7,3	0,37
6.	4,2	0,34	**36.**	9,8	0,075	**66.**	1,1	0,023	**96.**	2,4	0,035
7.	2,4	0,32	**37.**	1,9	0,03	**67.**	7,1	0,052	**97.**	6,9	0,078
8.	8,4	0,07	**38.**	1,5	0,016	**68.**	0,11	0,093	**98.**	0,24	0,082
9.	3,4	0,07	**39.**	3,9	0,44	**69.**	9,7	0,35	**99.**	7,9	0,44
10.	4,1	0,27	**40.**	5,1	0,076	**70.**	3,3	0,024	**100.**	4,6	0,037
11.	2,6	0,033	**41.**	2,8	0,025	**71.**	7,2	0,053	**101.**	8,3	0,064
12.	8,6	0,038	**42.**	1,4	0,017	**72.**	0,22	0,094	**102.**	0,33	0,085
13.	4,3	0,06	**43.**	6,1	0,045	**73.**	5,1	0,15	**103.**	3,7	0,24
14.	9,5	0,12	**44.**	5,2	0,077	**74.**	5,4	0,025	**104.**	4,7	0,031
15.	2,7	0,034	**45.**	3,7	0,015	**75.**	7,4	0,054	**105.**	5,7	0,063
16.	8,9	0,039	**46.**	1,3	0,018	**76.**	0,33	0,095	**106.**	0,34	0,073
17.	5,6	0,05	**47.**	6,2	0,046	**77.**	4,9	0,11	**107.**	5,7	0,34
18.	8,5	0,011	**48.**	5,3	0,078	**78.**	6,6	0,026	**108.**	5,2	0,035
19.	2,9	0,035	**49.**	4,6	0,95	**79.**	7,6	0,056	**109.**	8,9	0,071
20.	9,2	0,071	**50.**	1,6	0,019	**80.**	0,44	0,096	**110.**	0,38	0,073
21.	6,5	0,04	**51.**	6,3	0,047	**81.**	4,8	0,22	**111.**	4,8	0,76
22.	7,5	0,012	**52.**	5,4	0,087	**82.**	7,7	0,027	**112.**	6,6	0,052
23.	3,1	0,036	**53.**	5,5	0,85	**83.**	7,9	0,057	**113.**	8,3	0,044
24.	9,3	0,072	**54.**	1,7	0,021	**84.**	0,55	0,097	**114.**	0,47	0,063
25.	7,8	0,03	**55.**	6,7	0,048	**85.**	4,7	0,35	**115.**	6,3	0,56
26.	4,5	0,13	**56.**	5,7	0,089	**86.**	8,8	0,028	**116.**	7,9	0,036
27.	3,2	0,041	**57.**	6,4	0,65	**87.**	8,1	0,058	**117.**	9,2	0,071
28.	9,7	0,073	**58.**	1,8	0,022	**88.**	0,66	0,098	**118.**	0,54	0,083
29.	8,7	0,02	**59.**	6,8	0,049	**89.**	4,4	0,41	**119.**	3,5	0,52
30.	3,5	0,014	**60.**	5,8	0,091	**90.**	9,9	0,55	**120.**	7,6	0,43

Таблица 8.

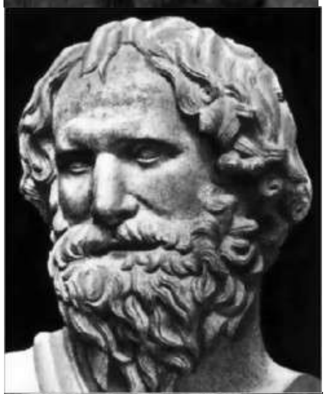
№	*A*	*B*	№	*A*	*B*	№	*A*	*B*	№	*A*	*B*
1.	2,8	0,025	**31.**	6,3	0,047	**61.**	6,5	0,04	**91.**	5,3	0,072
2.	1,4	0,017	**32.**	5,4	0,087	**62.**	7,5	0,012	**92.**	2,6	0,063
3.	6,1	0,045	**33.**	5,5	0,85	**63.**	3,1	0,036	**93.**	6,2	0,38
4.	5,2	0,077	**34.**	1,7	0,021	**64.**	9,3	0,072	**94.**	2,9	0,034
5.	3,7	0,15	**35.**	6,7	0,048	**65.**	7,8	0,03	**95.**	3,8	0,081
6.	1,3	0,018	**36.**	5,7	0,089	**66.**	4,5	0,013	**96.**	5,3	0,093
7.	6,2	0,046	**37.**	1,2	0,09	**67.**	3,2	0,041	**97.**	2,6	0,081
8.	5,3	0,078	**38.**	4,3	0,4	**68.**	9,4	0,073	**98.**	3,7	0,13
9.	7,3	0,055	**39.**	2,3	0,029	**69.**	4,4	0,6	**99.**	3,2	0,037
10.	2,2	0,031	**40.**	8,3	0,059	**70.**	9,9	0,55	**100.**	7,5	0,063
11.	6,9	0,051	**41.**	5,6	0,05	**71.**	7,4	0,054	**101.**	4,7	0,043
12.	5,9	0,092	**42.**	8,5	0,011	**72.**	0,33	0,095	**102.**	7,3	0,021
13.	8,2	0,45	**43.**	2,9	0,035	**73.**	9,7	0,35	**103.**	3,3	0,043
14.	1,1	0,23	**44.**	9,2	0,071	**74.**	3,3	0,024	**104.**	8,1	0,062
15.	7,1	0,52	**45.**	2,1	0,08	**75.**	7,2	0,053	**105.**	3,2	0,076
16.	0,11	0,092	**46.**	4,1	0,2	**76.**	0,22	0,086	**106.**	5,2	0,33
17.	4,8	0,1	**47.**	2,6	0,033	**77.**	6,4	0,65	**107.**	3,5	0,013
18.	6,6	0,026	**48.**	8,6	0,038	**78.**	1,8	0,024	**108.**	9,3	0,041
19.	7,6	0,056	**49.**	8,7	0,02	**79.**	6,8	0,049	**109.**	7,3	0,019
20.	0,44	0,096	**50.**	3,5	0,014	**80.**	5,8	0,091	**110.**	2,9	0,032
21.	4,7	0,2	**51.**	3,6	0,042	**81.**	1,9	0,03	**111.**	4,7	0,053
22.	7,7	0,028	**52.**	9,6	0,074	**82.**	1,5	0,016	**112.**	8,7	0,062
23.	7,9	0,057	**53.**	3,4	0,08	**83.**	3,9	0,044	**113.**	2,9	0,078
24.	0,55	0,098	**54.**	4,2	0,03	**84.**	5,1	0,076	**114.**	5,1	0,035
25.	4,9	0,3	**55.**	2,4	0,032	**85.**	9,8	0,02	**115.**	3,2	0,047
26.	8,8	0,027	**56.**	8,4	0,037	**86.**	2,6	0,015	**116.**	9,3	0,071
27.	8,1	0,058	**57.**	4,6	0,06	**87.**	3,8	0,043	**117.**	5,4	0,023
28.	0,66	0,098	**58.**	9,7	0,2	**88.**	3,6	0,075	**118.**	8,6	0,19
29.	4,6	0,95	**59.**	2,7	0,034	**89.**	5,1	0,15	**119.**	3,9	0,042
30.	1,6	0,019	**60.**	8,9	0,039	**90.**	5,4	0,025	**120.**	9,8	0,043

Приложение 2

Именной указатель

Анаксагор (др.-греч. Ἀναξαγόρας) **из Клазомен** (ок. 496 до н. э. — 428 до н. э.) — древнегреческий философ, математик и астроном. Основоположник афинской философской школы.

Жан Робер Арган (фр. *Jean-Robert Argand*) (18.07.1768 —13.08.1822) — французский непрофессиональный математик швейцарского происхождения. В 1806 году, управляя книжным магазином в Париже, он издал идею геометрической интерпретации комплексных чисел, известную сейчас как диаграмма Аргана. Позже он ввёл термин «модуль комплексного числа» (1814—1815).



Архимед (Ἀρχιμήδης; 287 до н. э. — 212 до н. э.)— древнегреческий математик, физик и инженер из Сиракуз. Сделал множество открытий в геометрии. Заложил основы механики, гидростатики, был автором ряда важных изобретений.

Рафаэль Бомбелли (итал. *Rafael Bombelli*; ок. 1526—1572) — итальянский математик, инженер-гидравлик. Известен тем, что ввёл в математику комплексные числа и разработал базовые правила действий с ними.

Томас Брадвардин (англ. *Thomas Bradwardine*; 1290—26.08.1349) — английский философ, математик и механик, старший представитель группы оксфордских калькуляторов из Мертон-колледжа, членом которого он был с 1323. В 1349 был выбран архиепископом Кентерберийским.

Эндрю Дональд Бут (англ. *Andrew Donald Booth*; 11.02.1918—29.11.2009) — британский инженер-электрик, физик и программист. Создатель «Автоматической релейной вычислительной машины» (1948 г.), в которой в качестве оперативной памяти использовался магнитный барабан емкостью 256 двоичных 21-разрядных слов. Изобрел алгоритм умножения Бута.

Брахмагупта (санскр. ब्रह्मगुप्त, ок. 598—670) — индийский математик и астроном. Руководил обсерваторией в Удджайне. Оказал существенное влияние на развитие астрономии в Византии и исламских странах, стал использовать алгебраические методы для астрономических вычислений, ввёл правила операций с нулём, положительными и отрицательными величинами.

Джон Валлис, точнее — **Уоллис** (англ. *John Wallis*; 23.11. (3.12) 1616—28.10. (8.11) 1703) — английский математик, один из предшественников математического анализа. В 1655 году Валлис издал большой трактат «Арифметика бесконечного», где ввёл придуманный им символ бесконечности. В книге он сформулировал строгое определение предела переменной величины, продолжил многие идеи Декарта, впервые ввёл отрицательные абсциссы, вычислил суммы бесконечных рядов — по существу интегральные суммы, хотя понятия интеграла тогда ещё не было.

Карл Теодор Вильгельм Вейерштрасс (нем. *Karl Theodor Wilhelm Weierstraß*; 31.10.1815—19.02.1897) — немецкий математик, «отец современного анализа». Исследования Вейерштрасса существенно обогатили математический анализ, теорию специальных функций, вариационное исчисление, дифференциальную геометрию и линейную алгебру.

Каспар Вессель (норв. *Caspar Wessel*; 8.06.1745—29.03.1818) — датско-норвежский математик, по профессии землемер. Автор сочинения «Об аналитическом представлении направлений» (1799), посвященного теории векторов на плоскости и в пространстве, в котором впервые дано геометрическое представление комплексных чисел. В течение столетия сочинение Весселя оставалось неизвестным, а его результаты открывались вновь.

Франсуа Виет, сеньор де ля Биготье (фр. *François Viète, seigneur de la Bigotière*; 1540—13.02.1603) — французский математик, положивший начало алгебре как науке о преобразовании выражений, о решении уравнений в общем виде, создатель буквенного исчисления.

Христиан фон Вольф (нем. *Christian Freiherr von Wolff*; 24.01.1679—09.04.1754), немецкий философ, популяризатор и систематизатор идей Лейбница, почетный член Петербургской АН (1725). Х. Вольфу принадлежат несколько руководств по математике, оказавших сильное влияние на организацию преподавания этой дисциплины в Германии и России. Особое значение имел изданный им в 1716 году «Математический лексикон» («Mathematisches Lexikon»). Труд Вольфа был первым математическим словарём по-настоящему энциклопедического охвата.

Сэр **Уильям Роуэн Гамильтон** (англ. *William Rowan Hamilton*; 4.08.1805—2.09.1865) — ирландский математик, механик-теоретик, физик-теоретик, «один из лучших математиков XIX века». Известен фундаментальными открытиями в математике (кватернионы, основы векторного анализа, вариационное исчисление, обоснование комплексных чисел), аналитической механике (гамильтонова механика) и оптике

Иоганн Карл Фридрих Гаусс (нем. *Johann Carl Friedrich Gauß*; 30.04.1777—23.092.1855) — немецкий математик, механик, физик, астроном и геодезист. Считается одним из величайших математиков всех времён, «королём математиков».

Курт Фридрих Гёдель (нем. *Kurt Friedrich Gödel*; 28.04. 1906—14.01.1978) — австрийский логик, математик и философ математики, наиболее известный сформулированной и доказанной им теоремой о неполноте.

Давид Гильберт (нем. *David Hilbert*; 23.01.1862—14.02.1943) — немецкий математик-универсал, внёс тельный вклад в развитие многих областей математики. В 1910—1920-е годы (после смерти Анри Пуанкаре) был признанным мировым лидером математиков. Гильберт разработал широкий спектр фундаментальных идей во многих областях математики, в том числе теорию инвариантов и аксиоматику евклидовой геометрии. Он сформулировал теорию гильбертовых пространств, одну из основ современного функционального анализа.

Дмитрий Александрович Граве (6.09.1863—19.12.1939) — украинский, российский и советский математик, создатель первой крупной русской математической школы; академик АН УССР (1919), почётный член АН СССР (1929).

Герман Гюнтер Грассман (нем. *Hermann Günther Grassmann*; 15.04.1809—26.09.1877) — немецкий физик, математик и филолог, выбрал систему основных аксиом, определяющих сложение и умножение.

Жан Лерон Д'Аламбер (**д'Аламбер**, **Даламбер**; фр. *Jean Le Rond D'Alembert, d'Alembert*; 16.11.1717—29.10.1783) — французский философ, математик и механик. Математические исследования Д'Аламбера относятся к теории дифференциальных уравнений, где он дал метод решения дифференциального уравнения 2-го порядка в частных производных, описывающего поперечные колебания струны (волнового уравнения).

Юлиус Вильгельм Рихард Дедекинд (нем. *Julius Wilhelm Richard Dedekind*; 6.10.1831—12.02.1916) — немецкий математик, известный работами по общей алгебре и основаниям вещественных чисел. Дедекинд, обобщив теорию многочленов и алгебраических чисел, вводит в математику абстрактные алгебраические структуры: кольца, идеалы и модули.

Рене Декарт (фр. *René Descartes*, лат. *Renatus Cartesius* — Картезий; 31.03.1596 — 11.02.1650) — французский философ, математик, механик, физик и физиолог, создатель аналитической геометрии и современной алгебраической символики, автор *метода радикального сомнения* в философии, механицизма в физике, предтеча рефлексологии.

Диофант Александрийский (др.-греч. Διόφαντος ὁ Ἀλεξανδρεύς; лат. *Diophantus*) — древнегреческий математик, живший предположительно в III веке н. э. Упоминается как «отец алгебры». Автор «Арифметики» — книги, посвящённой нахождению положительных рациональных решений неопределённых уравнений. Диофант был первым греческим математиком, который рассматривал дроби наравне с другими числами. Диофант также первым среди античных учёных предложил развитую математическую символику, которая позволяла формулировать полученные им результаты в достаточно компактном виде.

Евклид (др.-греч. Εὐκλείδης; ок. 365—300 до н. э.) — древнегреческий математик. Главный труд «Начала» (15 книг), содержащий основы античной математики, элементарной геометрии, теории чисел, общей теории отношений и метода определения площадей и объемов, включавшего элементы теории пределов, оказал огромное влияние на развитие математики. Работы по астрономии, оптике, теории музыки.

Альберт Жирар (фр. *Albert Girard*; 1595—8.12.1632) — голландский математик. Впервые (1629) высказал основную теорему алгебры, доказанную лишь в 1799 г. К. Гауссом. При решении уравнений, наряду с положительными корнями, рассматривал и отрицательные. Ввел (1629) употребляемый ныне знак корня и знак плюс-минус. В книге по тригонометрии первым дал построение синуса, тангенса и котангенса.

Зенон Элейский (Элеатский; др.-греч. Ζήνων ὁ Ἐλεάτης; ок. 490 до н. э.—ок. 430 до н. э.) — древнегреческий философ, ученик Парменида, представитель Элейской школы. Знаменит своими апориями, которыми он пытался доказать противоречивость концепций движения, пространства и множества.

Георг Кантор (нем. *Georg Ferdinand Ludwig Philipp Cantor*, 3.03.1845—6.01.1918) — немецкий математик. Создатель теории множеств, ставшей краеугольным камнем в математике. Кантор ввёл понятие взаимно-однозначного соответствия между элементами множеств, дал определения бесконечного и вполне-упорядоченного множеств и доказал, что действительных чисел «больше», чем натуральных. Теорема Кантора, фактически, утверждает существование «бесконечности бесконечностей». Он определил понятия кардинальных и порядковых чисел и их арифметику.

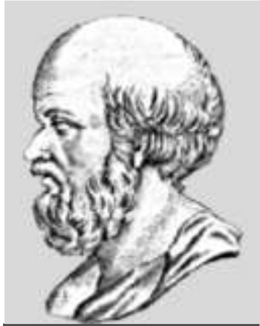
Лазар Карно (фр. *Lazare Nicolas Marguerite Carnot*; 13.05.1753—2.08.1823) — французский государственный и военный деятель, инженер и учёный. Первым предложил название «Комплексное число». Как учёный, Карно в основном занимался математическим анализом и геометрией.

Джероламо Кардано (лат. *Hieronymus Cardanus*, итал. *Girolamo Cardano*; 24.09.1501—21.09.1576) — итальянский математик, инженер, философ, медик и астролог. В его честь названы открытые Сципионом дель Ферро формулы решения кубического уравнения (Кардано был их первым публикатором), карданов подвес и карданный вал.

Гияс ад-Дин Джамшид ибн Масуд аль-Каши (перс. غياث‌الدين جمشيد كاشانى, англ. *Ghiyāth al-Dīn Jamshīd ibn Mas'ūd al-Kāshī*; 1380—22.06.1429) — один из крупнейших математиков и астрономов XV века, сотрудник Улугбека, один из руководителей Самаркандской обсерватории. В 1427 году написал книгу «Ключ к арифметике», в которой сформулировал основные правила действия с десятичными дробями. Вычислил величину числа π с точностью до 16-го знака после запятой.

Абрахам Готхельф Кестнер (нем. *Kästner Abraham Gotthelf*; 27.09.1719 - 20.06.1800). Немецкий математик и физик, иностранный почётный чл. Петербургской АН (с 23.10.1786), чл. Берлинской АН (1749). Учитель К. Гаусса. Основные труды по основаниям геометрии. Показал недоказуемость аксиомы параллельных.

Эратосфен Киренский (др.-гр. Ἐρατοσθένης ὁ Κυρηναῖος; 276 год до н. э.—194 год до н. э.) — греческий математик, астроном, географ, филолог и поэт. Ученик Каллимаха, с 235 г. До н. э. — глава Александрийской библиотеки. Первый известный учёный, вычисливший размеры Земли.



Алексей Николаевич Крылов (3.[15].08.1863—26.10.1945) — русский и советский кораблестроитель, механик и математик, академик Петербургской АН/РАН/АН СССР, генерал флота (06.12.1916), генерал для особых поручений при морском министре Российской империи (1911), лауреат Сталинской премии (1941), Герой Социалистического Труда (1943). А. Н. Крылов автор около 300 книг и статей. Они покрывают большой диапазон человеческого знания, включая судостроение, магнетизм, артиллерийское дело, математику, астрономию и геодезию.

Жозеф Луи Лагранж (фр. *Joseph Louis Lagrange*, итал. *Giuseppe Lodovico Lagrangia*; 25.01.1736—10.04.1813) — французский математик, астроном и механик итальянского происхождения. Наряду с Эйлером — крупнейший математик XVIII века. Особенно прославился исключительным мастерством в области обобщения и синтеза накопленного научного материала.

Иоганн Генрих Ламберт (нем. *Johann Heinrich Lambert*; 26.08.1728—25.09.1777) — физик, философ, математик и астроном; был академиком в Мюнхене и Берлине; один из основателей неевклидовой геометрии. Нашел разложение числа π в виде цепной дроби.

Анри Леон Лебег (фр. *Henri Leon Lebesgue*; 28.06.1875—26.07.1941) — французский математик, член Парижской АН (1922), член-корреспондент АН СССР (1929). Профессор Парижского университета (с 1910). Наиболее известен как автор теории интегрирования (т. н. интеграл Лебега), обобщающей обычное определение интеграла на более широкий класс функций. Интеграл Лебега нашёл широкое применение в теории вероятностей.

Готфрид Вильгельм Лейбниц (нем. *Gottfried Wilhelm von Leibniz*, 21.06.(1.07).1646—14.11.1716) — саксонский соф, логик, математик, механик, физик, юрист, историк, дипломат, изобретатель и языковед. Основатель и первый президент Берлинской Академии наук. Независимо от Ньютона, создал математический анализ — дифференциальное и интегральное исчисления, основанные на бесконечно малых. Создал комбинаторику как науку,' заложил основы математической логики, Описал двоичную систему счисления с цифрами 0 и 1. Сформулировал закон сохранения энергии.

Николай Иванович Лобачевский (20.11.(1.12.).1792—12 (24).02.1856) — русский математик, один из создателей неевклидовой геометрии, деятель университетского образования и народного просвещения.

Александр Михайлович Ляпунов (25.05.(6.06).1857—3.11.1918) —русский математик и механик, академик Петербургской Академии наук с 1901 года, член-корреспондент Парижской академии наук, член Национальной академии деи Линчеи .(Италия) и ряда других академий наук. А. М. Ляпунов создал теорию устойчивости равновесия и движения механических систем с конечным числом параметров. Работал в области: дифференциальных уравнений, гидродинамики, теории вероятностей.

Лещнтий Филиппович Магницкий (9.(19).06.1669—19 (30).1739) — русский математик, педагог. Преподаватель математики в Школе математических и навигацких наук в Москве (с 1701 по 1739), автор первого в России учебного пособия по математике.

Джон Непер (англ. *John Napier*; 1.02.1550—4.04.1617) — шотландский математик, один из изобретателей логарифмов, первый публикатор логарифмических таблиц. Для вычисления логарифмов предложил использовать устройство, называемое "палочками Непера", которые позволяли строго выполнять операции умножения и деления.

Сэр **Исаак Ньютон** (англ. *Isaac Newton*, 25.12.1642—20.03.1727) — английский физик, математик, механик и астроном, один из создателей классической физики. Автор фундаментального труда «Математические начала натуральной философии», в котором он изложил закон всемирного тяготения и три закона механики, ставшие основой классической механики. Разработал дифференциальное и интегральное исчисления, теорию цвета, заложил основы современной физической оптики, создал многие другие математические и физические теории.

Георг Симон Ом (нем. *Georg Simon Ohm*; 16.03.1787—6.07.1854) — немецкий физик. Он вывел теоретически и подтвердил на опыте закон, выражающий связь между силой тока в цепи, напряжением и сопротивлением (закон Ома).

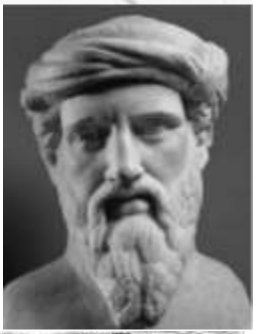
Николай Орем, или **Николай Орезмский** (фр. *Nicolas Oresme, Nicole Oresme*; до 1330 года—11.07.1382) французский философ, натурфилософ, математик, механик, астроном, теолог. Епископ города Лизьё. Его научные труды оказали влияние на Николая Кузанского, Коперника, Галилея и Декарта.

Фра Лука Бартоломео де Пачоли (устаревшее написание: **Пачиоли**, итал. *Fra Luca Bartolomeo de Pacioli*), (1445—19.06.1517) — итальянский математик, один из основоположников современных принципов бухгалтерии. Крупнейший европейский алгебраист XV в.

Джузеппе Пеано (итал. *Giuseppe Peano*; 27.08.1858—20.04.1932) — итальянский математик. Внёс вклад в математическую логику, аксиоматику, философию математики. Создатель вспомогательного искусственного языка латино-сине-флексионе. Более всего известен как автор стандартной аксиоматизации натуральной арифметики — арифметики Пеано.

Леонардо Пизанский (итал. *Leonardo Pisano*, около 1170 года — около 1250 года) — первый крупный математик средневековой Европы. Наиболее известен под прозвищем **Фибоначчи**. Пропагандист десятичной системы счисления и использования арабских цифр.

Бенджамин Пирс (англ. *Benjamin Peirce*; 04.04.1809–06.09.1880) — американский астроном и математик. Вычислил пертурбацию планет Урана и Нептуна. Пирс отвечал преимущественно за внедрение математики в американских исследовательских учреждениях. Он был известен своим вкладом в областях аналитической механики и линейной алгебры. Он сыграл роль в деле открытия планеты Нептун.



Пифагор Самосский (др.-греч. Πυθαγόρας ὁ Σάμιος, лат. *Pythagoras*, «пифийский вещатель»; 570—490 гг. до н. э.) — древнегреческий философ, математик и мистик, создатель религиозно-философской школы пифагорейцев.

Пьер де ла Раме, **Пётр Рамус** (фр. *Pierre de la Ramée*, лат. *Petrus Ramus*; 1515 — 26.08.1572) — французский философ, логик, математик, риторик, педагог. Снискал известность, выступив в 1536 году с тезисом *«всё, сказанное Аристотелем — ложно»*. Погиб от рук фанатиков в Варфоломеевскую ночь.

Сильвестр II (лат. *Silvester PP. II*), **Герберт Орильякский (Аврилакский)** (лат. *Gerbertus Aureliacus*; ок. 946—12.05.1003) — средневековый учёный, папа римский со 2.04.999 года по 12.05.1003 года. Он популяризировал арабские научные достижения в математике и астрономии в Европе. Возродил использование абака, армиллярной сферы и астролябии, забытые в Европе после падения Римской империи.

Симон Стевин (нидерл. *Simon Stevin*, 1548—1620) — фламандский математик, механик и инженер. Пропагандист десятичной системы. Стал известен прежде всего своей книгой Десятая (De Thiende) изданной в 1585 г. После неё в Европе началось широкое использование десятичных дробей.

Владимир Иванович Смирнов (29.05(10.06).1887—11.02.1974) — российский, советский математик, академик АН СССР. Герой Социалистического Труда. Лауреат Сталинской премии. Основные труды по теории функций комплексного переменного: униформизация многозначных аналитических функций, исследование фуксовых групп и фуксовых функций, исследование полноты системы многочленов, ных на спрямляемом замкнутом контуре, вопросы, связанные с предельными значениями аналитических функций. В. И. Смирнов разработал новый метод решения некоторых задач теории распространения волн в упругих средах с плоскими границами.

Владимир Андреевич Стеклов (28.12.1863 (9.01.1864)—30.05.1926) — русский математик и механик. Действительный член Петербургской Академии наук (1912), вице-президент АН СССР (1919—1926). Организатор и первый директор Физико-математического института РАН. После разделения Физико-математического института на институт математики и институт физики (в 1934 году) имя В. А. Стеклова было присвоено институту математики. Основные работы В. А. Стеклова (их насчитывается более 150) относятся к математической физике, механике, квадратурным формулам теории приближений, асимптотическим методам, теории замкнутости, ортогональным многочленам. Им получены значительные результаты в решении задач Дирихле и Неймана.

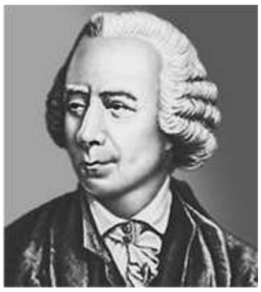
Жюль Таннери (24.03.1848—11.12.1910) — французский математик и философ. Занимался теорией функций действительного переменного и теорией множеств. Среди его работ: "Введение в теорию функций переменной", "Элементы теории эллиптических функций", "Введение в теорию чисел и высшей алгебры", "Трактат об арифметике", "Наука и философия".

Михаэль Штифель (нем. *Michael Stifel*, около 1487—19.04.1567) — немецкий математик, один из изобретателей логарифмов. Штифель написал на латинском языке «Полную арифметику» (Arithmetica integra, 1544) и на немецком языке «Немецкую арифметику» (Deutsche Arithmetik, 1545). Он дал содержательную теорию отрицательных чисел, возведения в степень, различных прогрессий и других последовательностей. Штифель впервые использовал понятия «корень» и «показатель степени», причём подробно анализировал и целые, и дробные показатели. Опубликовал правило образования биномиальных коэффициентов и составил их таблицы до 18-й степени.

Никола Шюке (фр. *Nicolas Chuquet*, 1445(?)—около 1488) — французский математик, оказавший влияние на развитие алгебры. Наиболее известен вводом в общее употребление названий больших чисел: *биллион*, *триллион* и т. д. Первым предложил обозначать показатели степени малыми литерами справа вверху.

Пьер де Ферма (фр. *Pierre de Fermat*, 17.08.1601—12.01.1665) — французский математик, один из создателей аналитической геометрии, математического анализа, теории вероятностей и теории чисел. По профессии юрист, с 1631 года — советник парламента в Тулузе. Блестящий полиглот. Наиболее известен формулировкой Великой теоремы Ферма.

Фердинанд Георг Фробениус (нем. *Ferdinand Georg Frobenius*; 26.10.1849—3.08.1917) — немецкий математик, известный за его вклад в теорию эллиптических функций, дифференциальных уравнений и теории групп. Он был первым, кто ввёл понятие рациональной аппроксимации функций (ныне известный как аппроксимации Паде), и дал первое полное доказательство теоремы Гамильтона — Кэли. Он внёс свой вклад в определение дифференциально-геометрических объектов в современной математической физике, известных ныне как многообразия Фробениуса.



Леонард Эйлер (нем. *Leonhard Euler*; 15.04.1707—7 (18).09.1783) — швейцарский, немецкий и российский математик и механик, внёсший фундаментальный вклад в развитие этих наук (а также физики, астрономии и ряда прикладных наук). Эйлер — автор более чем 850 работ (включая два десятка фундаментальных монографий) по математическому анализу, дифференциальной геометрии, теории чисел, приближённым вычислениям, небесной механике, математической физике, оптике, баллистике, кораблестроению, теории музыки и другим областям.

Миссия университета – генерация передовых знаний, внедрение инновационных разработок и подготовка элитных кадров, способных действовать в условиях быстро меняющегося мира и обеспечивать опережающее развитие науки, технологий и других областей для содействия решению актуальных задач.

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

О кафедре

Кафедра вычислительной техники Университета ИТМО создана в 1937 году и является одной из старейших и авторитетнейших научно-педагогических школ России.

Первоначально кафедра называлась кафедрой математических и счетно- решающих приборов и устройств и занималась разработкой электромеханических вычислительных устройств и приборов управления. Свое нынешнее название кафедра получила в 1963 году.

Кафедра вычислительной техники является одной из крупнейших в университете, на которой работают высококвалифицированные специалисты, в том числе 7 профессоров и 14 доцентов.

Кафедра имеет 4 компьютерных класса, объединяющих более 70 компьютеров в локальную вычислительную сеть кафедры и обеспечивающих доступ студентов ко всем информационным ресурсам кафедры и выход в Интернет. Кроме того, на кафедре имеются учебные и научно-исследовательские лаборатории по вычислительной технике, в которых работают студенты кафедры.

Чему мы учим

Традиционно на кафедре вычислительной техники Университета ИТМО основной упор в подготовке специалистов делается на фундаментальную базовую подготовку в рамках общепрофессиональных и специальных дисциплин, охватывающих наиболее важные разделы вычислительной техники: функциональная схемотехника и микропроцессорная техника, алгоритмизация и программирование, информационные системы и базы данных, мультимедиа технологии, вычислительные сети и средства телекоммуникации, защита информации и информационная безопасность. В то же время, кафедра предоставляет студентам старших курсов возможность специализироваться в более узких профессиональных областях в соответствии с их интересами.

Специализации на выбор

Кафедра вычислительной техники Университета ИТМО ведёт подготовку специалистов высшей квалификации в соответствии с Государственными образовательными стандартами 3-го поколения (ГОС-3) по двум направлениям:

09.04.01 «Информатика и вычислительная техника» (профиль подготовки «Вычислительные машины, комплексы, системы и сети»);

09.04.04 «Программная инженерия» (профиль подготовки «Разработка программно-информационных систем»); с присвоением степени (квалификации) бакалавр (срок обучения – 4 года).

Прием абитуриентов на указанные направления подготовки бакалавров осуществляется в соответствии с общими Правилами приема в Университет ИТМО.

Студенты, успешно завершившие обучение и получившие диплом бакалавра, могут продолжить обучение в магистратуре кафедры (срок обучения – 2 года) по следующим магистерским программам:

• «Безопасность вычислительных систем и сетей» – руководитель д.т.н., профессор Щеглов Андрей Юрьевич;

• «Вычислительные системы и сети» - руководитель д.т.н., профессор Алиев Тауфик Измайлович;

• «Информационно-вычислительные системы»- руководитель д.т.н., профессор Алиев Тауфик Измайлович;

• «Интеллектуальные информационные системы» – руководитель д.т.н., профессор Тропченко Александр Ювенальевич;

• «Проектирование встроенных вычислительных систем» - руководитель д.т.н., профессор Платунов Алексей Евгеньевич;

• «Системотехника интегральных вычислителей. Системы на кристалле» – руководитель д.т.н., профессор Платунов Алексей Евгеньевич;

• «Сетевые встроенные системы» - руководитель д.т.н., профессор Платунов Алексей Евгеньевич;

• «Технологии компьютерной визуализации» (совместно с базовой кафедрой Института Прикладной математики им. М.В. Келдыша) – руководитель д.т.н., профессор Galташев Тимур Турсунович.

В магистратуру на конкурсной основе принимаются выпускники других вузов, имеющие диплом бакалавра.

На кафедре вычислительной техники Университета ИТМО в рамках аспирантуры и докторантуры осуществляется подготовка научных кадров по следующим специальностям:

• 05.13.05 – Элементы и устройства вычислительной техники и систем управления (технические науки);

• 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей (технические науки);

• 05.13.12 – Системы автоматизации проектирования (приборостроение) (технические науки);

• 05.13.15 – Вычислительные машины, комплексы и компьютерные сети (технические науки);

• 05.13.17 – Теоретические основы информатики (технические науки);

• 05.13.18 – Математическое моделирование, численные методы и комплексы программ (технические науки);

• 05.13.19 – Методы и системы защиты информации, информационная безопасность (технические науки).

КАФЕДРА ИНФОРМАТИКИ И ПРИКЛАДНОЙ МАТЕМАТИКИ

О кафедре

В 1976 году из кафедры Вычислительной техники выделяется кафедра Прикладной математики, на которую возлагается задача по подготовке специалистов в области программирования и методов вычислений. Свое нынешнее название кафедра получила в 1994 году.

На кафедре работают высококвалифицированные специалисты, в том числе 2 профессора, доктора наук и 12 доцентов, к.т.н.

Кафедра проводит занятия в 6 компьютерных классах, объединяющих более 80 компьютеров в локальную вычислительную сеть и обеспечивающих доступ студентов ко всем информационным ресурсам кафедры и выход в Интернет. Кафедра осуществляет методическое руководство студенческим вычислительным центром. Кроме того, на кафедре имеются учебные и научно-исследовательские лаборатории по научным направлениям кафедры, в которых работают студенты, магистранты и аспиранты кафедры.

Основными научными интересами кафедры ИПМ являются:

* Онтологическое моделирование.
* Методы верификации вычислительных процессов.
* Информационное и программное обеспечение САПР.
* Семантические технологии.

Чему мы учим

Традиционно на кафедре информатики и прикладной математики Университета ИТМО основной упор в подготовке специалистов делается на фундаментальную базовую подготовку в рамках общепрофессиональных и специальных дисциплин, охватывающих наиболее важные разделы программной инженерии: алгоритмизация и программирование, информационные системы и базы данных, базы знаний, интернет-технологии, компьютерное моделирование и автоматизация проектирования.

Кафедра готовит бакалавров по программе «Разработка программно-информационных систем», которая ориентирована на охват практически всей области разработки программного обеспечения – от выбора и использования инструментария до создания архитектуры приложения и организации процесса разработки, поскольку цель этой программы – подготовка специалиста высокого класса, а не программиста, владеющего только парой-тройкой языков и имеющего опыт в создании приложений только какого-то определённого класса. Такая подготовка является высоким конкурентным преимуществом наших выпускников на рынке труда.

В то же время, кафедра предоставляет студентам старших курсов возможность специализироваться в более узких профессиональных областях в соответствии с их интересами.

Специализация

Кафедра информатики и прикладной математики Университета ИТМО ведёт подготовку специалистов высшей квалификации в соответствии с Государственными образовательными стандартами 3-го поколения (ГОС-3) по направлению:

09.04.04 «Программная инженерия» (профиль подготовки «Разработка программно-информационных систем»); с присвоением степени (квалификации) бакалавр (срок обучения – 4 года).

Прием абитуриентов на указанные направления подготовки бакалавров осуществляется в соответствии с общими Правилами приема в Университет ИТМО.

Студенты, успешно завершившие обучение и получившие диплом бакалавра, могут продолжить обучение в магистратуре кафедры (срок обучения – 2 года) по следующим магистерским программам:

* Математические модели и компьютерное моделирование – руководитель д.т.н., профессор Демин Анатолий Владимирович.
* Разработка программно-информационных систем - руководитель к.т.н., доцент Муромцев Дмитрий Ильич.

Целью образовательной программы «Математические модели и компьютерное моделирование» является подготовка системных аналитиков и разработчиков в области инжиниринга киберфизических систем. Специфика данной образовательной программы состоит в том, что основной упор делается на методы и средства математического и компьютерного моделирования, интеллектуализацию методов поиска решений и прогнозирования социальных и технических процессов. Выпускники, освоившие эту магистерскую программу способны проектировать и создавать киберфизические системы, включая программное обеспечение и протоколы обмена информации.

Целью программы «Разработка программно-информационных систем» является подготовка специалистов, удовлетворяющих потребностям отраслей промышленности, в которых требуется организация процесса разработки программного обеспечения, а также непосредственное участие в нем с применением навыков и знаний в области языков программирования, проектирования и тестирования ПО, информационного и математического моделирования. Выпускники, освоившие эту магистерскую программу, способны заниматься исследованиями в области создания инструментария для разработки ПО, распределенных систем, интеллектуального анализа данных, машинного обучения.

В магистратуру на конкурсной основе принимаются выпускники других вузов, имеющие диплом бакалавра.

На кафедре информатики и прикладной математики Университета ИТМО в рамках аспирантуры и докторантуры осуществляется подготовка научных кадров высшей квалификации по следующим специальностям:

• 05.13.12 – Системы автоматизации проектирования (приборостроение) (технические науки);

• 05.13.17 – Теоретические основы информатики (технические науки);

• 05.13.18 – Математическое моделирование, численные методы и комплексы программ (технические науки);

• 05.13.19 – Методы и системы защиты информации, информационная безопасность (технические науки).

Зыков Анатолий Геннадьевич
Поляков Владимир Иванович

Арифметические основы ЭВМ

Учебное пособие

В авторской редакции
Редакционно-издательский отдел Университет ИТМО
Зав. РИО Н.Ф. Гусарова
Подписано к печати …
Заказ №
Тираж
Отпечатано на ризографе