Министерство образования Республики Беларусь

Учреждение образования   
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра информатики

Отчет по лабораторным работам №4-6  
«Арифметические операции с числами с плавающей точкой»

Выполнил:  
студент гр. 153504  
Климкович Н.В.

Проверила:

Калиновская А.А.

Минск 2023

# 1. Цель работы

Изучить арифметические операции над числами с плавающей точкой различными алгоритмами выполнения. Рассмотреть представление чисел в формате с плавающей точкой в двоичном коде.

# 2. Постановка задачи

***Задание к лабораторной работе 4***

Написать программу эмулятора АЛУ, реализующего *операции сложения и вычитания с плавающей* точкой над двумя введенными числами, с возможностью пошагового выполнения алгоритмов.

***Задание к лабораторной работе 5***

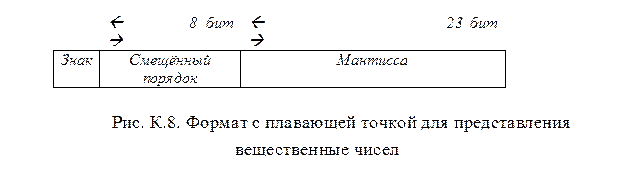
Написать программу эмулятора АЛУ, реализующего *операцию умножения с плавающей точкой* над двумя введенными числами, с возможностью пошагового выполнения алгоритмов.

***Задание к лабораторной работе 6***

Написать программу эмулятора АЛУ, реализующего *операцию деления с плавающей точкой* над двумя введенными числами, с возможностью пошагового выполнения алгоритмов.

# 3. Теоретические сведения

### *Представление чисел в формате с плавающей точкой*



Обратите внимание на следующие особенности:

·        знак сохраняется в старшем бите слова;

·        первый разряд мантиссы всегда равен 1, и в поле мантиссы не хранится;

·        к действительному значению порядка прибавляется смещение 127 и в поле порядка хранится эта сумма;

·        основание характеристики равно 2.

При сложении и вычитании необходимо предварительно уравнять порядки операндов, что требует сдвига положения разделительной точки в мантиссе. Операции умножения и деления уравнивания порядков операндов не требуют

При выполнении операций могут возникнуть следующие особые ситуации.

·        **Переполнение порядка.**Положительный порядок результата превышает максимальное значение, предусмотренное форматом. В некоторых системах такой результат может трактоваться как величина +∞ или -∞.

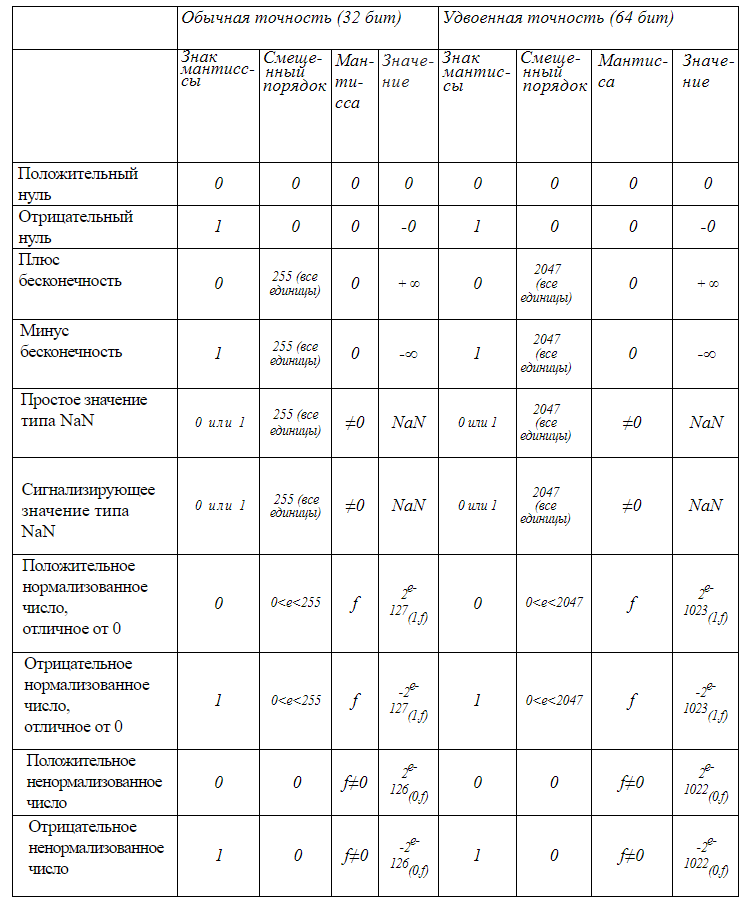
·        **Потеря значимости порядка.**Отрицательный порядок результата меньше минимального значения, допускаемого принятым форматом (например, получен порядок -200, который меньше разрешенного -127). Это означает, что получен очень малый результат, который можно считать равным нулю.

·        **Потеря значимости мантиссы.**В процессе уравнивания порядков мантисса сдвигается настолько сильно вправо, что старший значащий ее разряд выходит за пределы разрядной сетки.

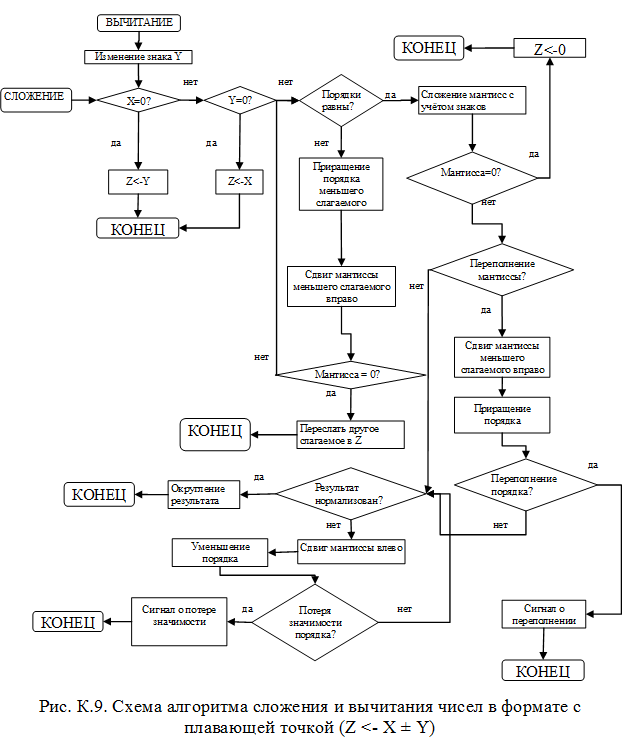
·        **Переполнение мантиссы.**При сложении мантисс с одинаковыми знаками возможно появление переноса из старшего разряда

### Стандарт IEEE формата с плавающей точкой

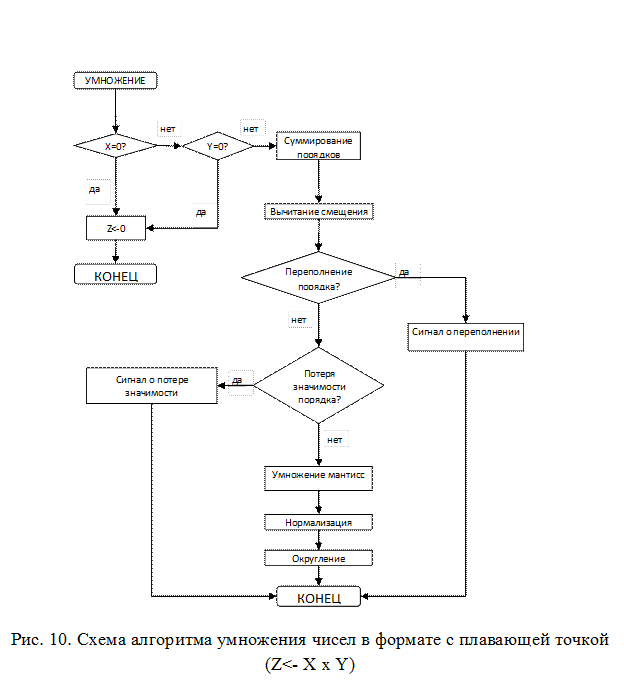
Для унификации формата представления чисел с плавающей точкой, что явля­ется необходимым условием переносимости программного обеспечения, Институтом инженеров по электротехнике и радиоэлектронике IEEE разработан стандарт 754. В последнее десятилетие практически все процессоры и арифметические сопроцессоры проектируются с учетом требований этого стандарта.

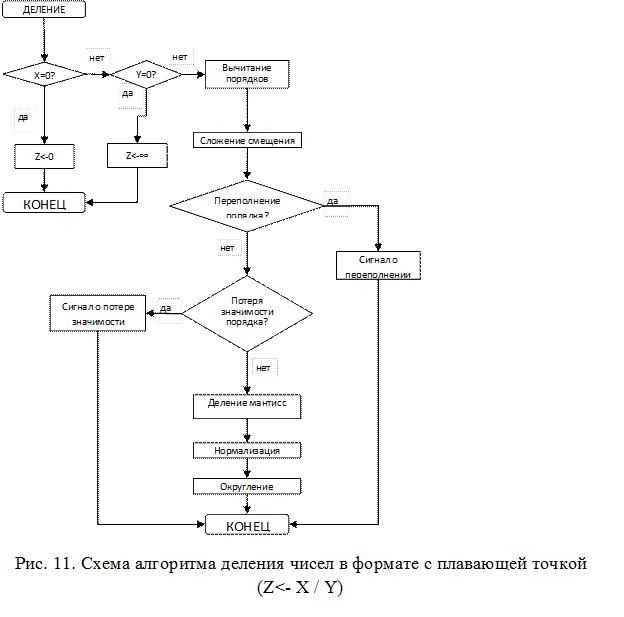
**

***Сложение и вычитание***

******

***Умножение и деление***

******



# 4. Программная реализация

В качестве языка для реализации эмулятора АЛУ был выбран язык Python.

Для работы с операциями сложения, вычитания, умножения и деления предусмотрено хранение двоичных чисел в 64-битном формате. Предусмотрена возможность обработки ситуаций переполнения с выводом информации о проблеме.

На рисунках представлен результат работы программы. Выполняется сложение и вычитание, умножения и деления с плавающей точкой. Показана подробная обработка арифметических операций.

**x1: -234765.12384**

**x1: 10-е в 2-е: 234765.12384 = 111001010100001101.00011111101100111111**

**x2: 632456.732166**

**x2: 10-е в 2-е: 632456.732166 = 10011010011010001000.10111011011011110011**

**Представление x1: 1 10000010001 1100101010000110100011111101100111111000000000000000**

**Представление x2: 0 10000010011 0011010011010001000101110110110111100110000000000000**

**==================================================**

**Сложение:**

**x1=0? Нет.**

**x2=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000010010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 01110010101000011010001111110110011111100000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000010011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00111001010100001101000111111011001111110000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Да.**

**Сложение мантисс с учётом знака. Результат: 001100001000101111011100110111011101101000000000000000**

**Мантисса=0? Нет.**

**Переполнение мантиссы? Нет.**

**Результат нормализован? Нет.**

**Сдвиг мантиссы влево. Результат мантиссы: 11000010001011110111001101110111011010000000000000000**

**Уменьшение порядка. Результат порядка: 10000010010**

**Потеря значимости порядка? Да**

**Результат нормализован? Да.**

**Ответ: 0 10000010010 1000010001011110111001101110111011010000000000000000**

**Ответ: +397691.60832595825**

**==================================================**

Рис.12 Пример 1. Сложение чисел: -234765,12384 и 632456,732166.

**Вычитание:**

**Изменение знака x2. Результат x2: 1100000100110011010011010001000101110110110111100110000000000000**

**x1=0? Нет.**

**x2=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000010010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 01110010101000011010001111110110011111100000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000010011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00111001010100001101000111111011001111110000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Да.**

**Сложение мантисс с учётом знака. Результат: 111010011101110010101110110110010001100100000000000000**

**Мантисса=0? Нет.**

**Переполнение мантиссы? Нет.**

**Результат нормализован? Да.**

**Ответ: 1 10000010011 1010011101110010101110110110010001100100000000000000**

**Ответ: -867221.856004715**

**==================================================**

**Умножение:**

**x1=0? Нет.**

**x2=0? Нет.**

**Суммирование порядков. Результат: 00000100100**

**Вычитанием смещения. Результат: 10000100101**

**Переполнение порядка? Нет.**

**Потеря значимости порядка? Нет.**

**Умножение мантисс Результат: 11000101001001000000110011011100100100111011000000000110010010101000110110011010000000000000000000000000000**

**Нормализация. Результат: 1 10000100101 000101001001000000110011011100100100111011000000000110010010101000110110011010000000000000000000000000000**

**Округление. Результат: 1 10000100101 0001010010010000001100110111001001001110110000000001**

**Ответ: 1 10000100101 0001010010010000001100110111001001001110110000000001**

**Ответ: -148478783049.84375**

**==================================================**

**Деление:**

**x1=0? Нет.**

**x2=0? Нет.**

**Вычитание порядков. Результат: 11111111110**

**Сложение смещения. Результат: 01111111110**

**Переполнение порядка? Нет.**

**Потеря значимости порядка? Нет.**

**Деление мантисс. Результат: 11011111000001101010100000000000000000000000000000000**

**Нормализация. Результат: 1 01111111110 011111000001101010100000000000000000000000000000000**

**Округление. Результат: 1 01111111110 011111000001101010100000000000000000000000000000000**

**Ответ: 1 01111111110 011111000001101010100000000000000000000000000000000**

**Ответ: -0.37119531631469727**

Рис.13 Пример 1. Вычитание, умножение и деление чисел: -234765,12384 и 632456,732166.

**x1: 0.02130100434**

**x1: 10-е в 2-е: 0.02130100434 = 0.00000101011100111111**

**x2: 0**

**x2: 10-е в 2-е: 0 = 0.00000000000000000000**

**Представление x1: 0 01111111010 0101110011111100000000000000000000000000000000000000**

**Представление x2: 0 01111111111 0000000000000000000000000000000000000000000000000000**

**==================================================**

**Сложение:**

**x1=0? Нет.**

**x2=0? Да. Результат: x1**

**Ответ: 0 01111111010 0101110011111100000000000000000000000000000000000000**

**Ответ: +0.021300315856933594**

**==================================================**

**Вычитание:**

**Изменение знака x2. Результат x2: 1011111111110000000000000000000000000000000000000000000000000000**

**x1=0? Нет.**

**x2=0? Да. Результат: x1**

**Ответ: 0 01111111010 0101110011111100000000000000000000000000000000000000**

**Ответ: +0.021300315856933594**

**==================================================**

**Умножение:**

**x1=0? Нет.**

**x2=0? Да.**

**Ответ: 0**

**==================================================**

**Деление:**

**x1=0? Нет.**

**x2=0? Да.**

**Ответ: infinity**

Рис.14 Пример 2. Сложение, вычитание, умножение и деление чисел: 0,02130100434 и 0.

**x1: 0.0123**

**x1: 10-е в 2-е: 0.0123 = 0.00000011001001100001**

**x2: 234.521**

**x2: 10-е в 2-е: 234.521 = 11101010.100001010110000001**

**Представление x1: 0 01111111001 1001001100001000000000000000000000000000000000000000**

**Представление x2: 0 10000000111 1101010100001010110000001000000000000000000000000000**

**==================================================**

**Умножение:**

**x1=0? Нет.**

**x2=0? Нет.**

**Суммирование порядков. Результат: 00000000000**

**Вычитанием смещения. Результат: 10000000001**

**Переполнение порядка? Нет.**

**Потеря значимости порядка? Нет.**

**Умножение мантисс Результат: 01011100010011011101001010101011111100001000000000000000000000000000000000000000000000000000000000000000000**

**Нормализация. Результат: 0 10000000001 011100010011011101001010101011111100001000000000000000000000000000000000000000000000000000000000000000000**

**Округление. Результат: 0 10000000001 0111000100110111010010101010111111000010000000000000**

**Ответ: 0 10000000001 0111000100110111010010101010111111000010000000000000**

**Ответ: +2.884499870127911**

Рис.15 Пример 3. Умножение чисел: 0,0123 и 234,521.

**==================================================**

**Сложение:**

**x1=0? Нет.**

**x2=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 01100100110000100000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00110010011000010000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111100**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00011001001100001000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111101**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00001100100110000100000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111110**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000110010011000010000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111111**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000011001001100001000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000000**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000001100100110000100000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000001**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000110010011000010000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000011001001100001000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000001100100110000100000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000100**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000110010011000010000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000101**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000011001001100001000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000110**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000001100100110000100000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000111**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000000110010011000010000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Да.**

**Сложение мантисс с учётом знака. Результат: 011101010100010001000011001010000000000000000000000000**

**Мантисса=0? Нет.**

**Переполнение мантиссы? Нет.**

**Результат нормализован? Да.**

**Ответ: 0 10000000111 1101010100010001000011001010000000000000000000000000**

**Ответ: +234.53329944610596**

**==================================================**

Рис.15 Пример 3. Сложение чисел: 0,0123 и 234,521.

**Вычитание:**

**Изменение знака x2. Результат x2: 1100000001111101010100001010110000001000000000000000000000000000**

**x1=0? Нет.**

**x2=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 01100100110000100000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00110010011000010000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111100**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00011001001100001000000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111101**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00001100100110000100000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111110**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000110010011000010000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 01111111111**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000011001001100001000000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000000**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000001100100110000100000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000001**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000110010011000010000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000010**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000011001001100001000000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000011**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000001100100110000100000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000100**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000110010011000010000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000101**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000011001001100001000000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000110**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000001100100110000100000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Нет.**

**Приращение порядка меньшего слагаемого. Результат порядка x1: 10000000111**

**Сдвиг мантиссы меньшего слагаемого вправо. Результат мантиссы x1: 00000000000000110010011000010000000000000000000000000**

**Мантисса=0? Нет.**

**Порядки равны? Да.**

**Сложение мантисс с учётом знака. Результат: 111101010100000100011101000110000000000000000000000000**

**Мантисса=0? Нет.**

**Переполнение мантиссы? Нет.**

**Результат нормализован? Да.**

**Ответ: 1 10000000111 1101010100000100011101000110000000000000000000000000**

**Ответ: -234.50870037078857**

**==================================================**

Рис.15 Пример 3. Вычитание чисел: 0,0123 и 234,521.

**Деление:**

**x1=0? Нет.**

**x2=0? Нет.**

**Вычитание порядков. Результат: 11111110010**

**Сложение смещения. Результат: 01111110010**

**Переполнение порядка? Нет.**

**Потеря значимости порядка? Нет.**

**Деление мантисс. Результат: 00110110111111100011000000000000000000000000000000000**

**Нормализация. Результат: 0 01111110001 101101111111000110000000000000000000000000000000000**

**Округление. Результат: 0 01111110001 101101111111000110000000000000000000000000000000000**

**Ответ: 0 01111110001 101101111111000110000000000000000000000000000000000**

**Ответ: +5.244533531367779e-05**

**==================================================**

Рис.15 Пример 3. Деление чисел: 0,0123 и 234,521.

# 5. Выводы

В ходе лабораторной работы были изучены принципы работы арифметических операций в АЛУ. Были подробно изучены понятия стандарта IEEE-754. Изучены алгоритмы сложения, вычитания, умножения и деления над числами с плавающей точкой.

# Список литературы

1. Волорова Н. А. Лабораторный практикум по курсу «Архитектура вычислительных систем» для студентов специальности «Информатика» / 93-444-487-2- Мн.: БГУИР, 2003. – 32с.:ил.

# Приложение 1. Текст программы

import re

import math

P = 11

M = 52

A = 20

# дополнительный код

def additional\_code(x):

if x[0] == '0':

return x

buf = "1"

for i in range(1, len(x)):

if x[i] == '1':

buf += '0'

else:

buf += '1'

return sum\_fix(buf, '0' \* (len(buf) - 1) + '1', True)

# двоичное в десятичное

def decimal(a):

buf = 0

minus = 1

if a[0] == '1':

minus = -1

a = '0' + a[1:]

for i in range(len(a)):

buf += float(a[len(a) - 1 - i]) \* float(pow(2, i))

return buf \* minus

# сумма

def sum\_fix(a, b, of):

buf = 0

rez = ""

for i in range(len(a) - 1, -1, -1):

if int(a[i]) + int(b[i]) + buf > 1:

rez += str((int(a[i]) + int(b[i]) + buf) % 2)

buf = 1

else:

rez += str(int(a[i]) + int(b[i]) + buf)

buf = 0

rez = rez[::-1]

if (a[0] == b[0]) and (a[0] != rez[0]):

if of:

return 'OverFlow'

return rez

# Отрицание

def negation(x):

buf = ""

for i in range(len(x)):

if x[i] == '1':

buf += '0'

else:

buf += '1'

return sum\_fix(buf, '0' \* (len(buf) - 1) + '1', False)

# вычитание

def subtraction(a, b, of):

b = negation(b)

return sum\_fix(a, b, of)

# значение после запятой в двоичную систему

def afterdot\_to\_bun(a):

buf = ""

a = '0.' + a

for i in range(A):

a = float(a) \* 2.0

if float(a) < 1.0:

buf += '0'

else:

buf += '1'

a = '0' + str(a)[1:]

if int(buf) != 0:

for i in range(A - 1, -1, -1):

if buf[i] == '0':

buf = buf[:i]

else:

break

return buf

# ввод

def input\_num(num):

while True:

a = str(input('x' + str(num) + ': '))

x1\_inp = a

try:

if -pow(2, M - A) + 1 < float(a) < pow(2, M - A) - 1:

break

else:

print('Error')

except Exception as e:

print(e)

minus = '0'

if a[0] == '-':

minus = '1'

a = a[1:]

befordot = bin(math.floor(float(a)))[2:]

afterdot = afterdot\_to\_bun(a[len(str(math.floor(float(a)))) + 1:])

print('x' + num + ': 10-е в 2-е:', a, '=', befordot + '.' + afterdot)

# если меньше 1

buf\_p = 0

new\_p = bin(0)[2:]

if int(befordot) == 0 and int(afterdot) != 0:

for i in range(len(afterdot)):

if afterdot[0] == '0':

afterdot = afterdot[1:]

buf\_p += 1

else:

break

afterdot = afterdot[1:]

else:

if int(befordot) != 0:

new\_p = bin(len(befordot))[2:]

offset = bin(pow(2, P - 1) - 1 - buf\_p)[2:]

return x1\_inp, minus + \

sum\_fix('0' \* (P - len(offset)) + offset, '0' \* (P - len(new\_p)) + new\_p, False) + \

befordot[1:] + afterdot + '0' \* (M - len(befordot[1:] + afterdot))

# сумма и вычитание

def sum\_or\_sub(sum, x1, x2, x1\_inp, x2\_inp):

print('=' \* 50)

if sum:

print("Сложение:")

else:

print("Вычитание:")

print('Изменение знака x2.', end=' ')

if x2[0] == '0':

x2 = '1' + x2[1:]

else:

x2 = '0' + x2[1:]

print('Результат x2:', x2)

x1\_znak = x1[0]

x1\_paradok = x1[1:P + 1]

x1\_mantisa = '1' + x1[P + 1:]

x2\_znak = x2[0]

x2\_paradok = x2[1:P + 1]

x2\_mantisa = '1' + x2[P + 1:]

print('x1=0?', end=' ')

if float(x1\_inp) == 0.0:

print('Да. Результат: x2')

return x2

print('Нет.')

print('x2=0?', end=' ')

if float(x2\_inp) == 0.0:

print('Да. Результат: x1')

return x1

print('Нет.')

while True:

print('Порядки равны?', end=' ')

if x1\_paradok == x2\_paradok:

print('Да.')

print('Сложение мантисс с учётом знака.', end=' ')

rez\_sum = sum\_fix(additional\_code(x1\_znak + x1\_mantisa), additional\_code(x2\_znak + x2\_mantisa), True)

if rez\_sum != 'OverFlow':

rez\_sum = additional\_code(rez\_sum)

print('Результат:', rez\_sum)

print('Мантисса=0?', end=' ')

if rez\_sum != 'OverFlow':

if int(rez\_sum[1:]) == 0:

print('Да. Результат: 0')

return '0' + '0' \* P + '0' \* M

print('Нет.')

print('Переполнение мантиссы?', end=' ')

if rez\_sum == 'OverFlow':

print('Да.')

print('Сдвиг мантиссы меньшего слагаемого вправо.', end=' ')

if int(decimal(x1\_mantisa)) > int(decimal(x2\_mantisa)):

x2\_mantisa = '0' + x2\_mantisa[:-1]

print('Результат, сдвиг x2:', x2\_mantisa)

print('Приращение порядка.', end=' ')

x2\_paradok = sum\_fix(x2\_paradok, '0' \* (P - 1) + '1', True)

print('Результат, порядок x2:', x2\_paradok)

else:

x1\_mantisa = '0' + x1\_mantisa[:-1]

print('Результат, сдвиг x1:', x1\_mantisa)

print('Приращение порядка.', end=' ')

x1\_paradok = sum\_fix(x1\_paradok, '0' \* (P - 1) + '1', True)

print('Результат, порядок x1:', x1\_mantisa)

print('Переполнение порядка?', end=' ')

if x1\_paradok == 'OverFlow' or x2\_paradok == 'OverFlow':

print('Да.', end=' ')

print('Сигнал о переполнении.')

return 'OverFlow'

print('Нет.')

print('Нет.')

while True:

print('Результат нормализован?', end=' ')

if rez\_sum[1] == '1':

print('Да.')

return rez\_sum[0] + x1\_paradok + rez\_sum[2:]

print('Нет.')

print('Сдвиг мантиссы влево.', end=' ')

rez\_sum = rez\_sum[0] + rez\_sum[2:] + '0'

print('Результат матиссы:', rez\_sum[1:])

print('Уменьшение порядка.', end=' ')

# x1\_paradok = subtraction(x1\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False)

x1\_paradok = subtraction((x1\_paradok), '0' \* (P - 1) + '1', True)

# x1\_paradok = sum\_fix(x1\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False)

print('Результат порядка:', x1\_paradok)

print('Потеря значимости порядка?', end=' ')

if x1\_paradok == 'OverFlow':

print('Да.', end=' ')

print('Сигнал о потере значимости.')

return 'OverFlow'

print('Да')

print('Нет.')

print('Приращение порядка меньшего слагаемого.', end=' ')

if int(decimal(subtraction(x1\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False))) > int(

decimal(subtraction(x2\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False))):

x2\_paradok = sum\_fix(x2\_paradok, '0' \* (P - 1) + '1', False)

print('Результат порядка x2:', x2\_paradok)

print('Сдвиг мантиссы меньшего слагаемого вправо.', end=' ')

x2\_mantisa = '0' + x2\_mantisa[:-1]

print('Результат мантиссы x2:', x2\_mantisa)

else:

x1\_paradok = sum\_fix(x1\_paradok, '0' \* (P - 1) + '1', False)

print('Результат порядка x1:', x1\_paradok)

print('Сдвиг мантиссы меньшего слагаемого вправо.', end=' ')

x1\_mantisa = '0' + x1\_mantisa[:-1]

print('Результат мантиссы x1:', x1\_mantisa)

print('Мантисса=0?', end=' ')

if int(x1\_mantisa) == 0 or int(x2\_mantisa) == 0:

print('Да.', end=' ')

print('Переслать другое слагаемое в ответ.')

if int(x1\_mantisa) == 0:

return x2\_znak + x2\_paradok + x2\_mantisa[1:]

return x1\_znak + x1\_paradok + x1\_mantisa[1:]

print('Нет.')

# перевод из M в M\*2 разряда доп код

def \_M\_to\_M2\_additional\_code(x):

if x[0] == '1':

return '1' \* (M + 2) + x

return '0' \* (M + 2) + x

# умножение

def multiplication\_fix(x1\_, x2\_):

m = \_M\_to\_M2\_additional\_code(x1\_)

a = '0' \* ((M + 2) \* 2)

q = \_M\_to\_M2\_additional\_code(x2\_)

q\_1 = '0'

for i in range((M + 2) \* 2):

if q[-1] + q\_1 == '01':

a = sum\_fix(a, m, False)

elif q[-1] + q\_1 == '10':

a = subtraction(a, m, False)

q\_1 = q[-1]

q = a[-1] + q[:-1]

a = a[0] + a[:-1]

return q

# умножение с плавающей точкой

def mult(x1, x2, x1\_inp, x2\_inp):

print('=' \* 50)

print('Умножение:')

print('x1=0?', end=' ')

if float(x1\_inp) == 0:

print('Да.')

return ' 0'

print('Нет.')

print('x2=0?', end=' ')

if float(x2\_inp) == 0:

print('Да.')

return ' 0'

print('Нет.')

x1\_znak = x1[0]

x1\_paradok = x1[1:P + 1]

x1\_mantisa = '1' + x1[P + 1:]

x2\_znak = x2[0]

x2\_paradok = x2[1:P + 1]

x2\_mantisa = '1' + x2[P + 1:]

print('Суммирование порядков.', end=' ')

rez\_paradok = sum\_fix(x1\_paradok, x2\_paradok, False)

print('Результат:', rez\_paradok)

print('Вычитанием смещения.', end='')

rez\_paradok = subtraction(rez\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False)

print('Результат:', rez\_paradok)

print('Переполнение порядка?', end=' ')

if rez\_paradok == 'OverFlow':

print('Да.')

print('Сигнал о переполнениию')

return 'OverFlow'

print('Нет.')

print('Потеря значимости поядка?', end=' ')

if rez\_paradok == 'OverFlow':

print('Да.')

print('Сигнал о потере значимости.')

return 'OverFlow'

print('Нет.')

print('Умножение мантисс', end=' ')

rez\_mult\_2 = additional\_code(

multiplication\_fix(additional\_code(x1\_znak + x1\_mantisa), additional\_code(x2\_znak + x2\_mantisa)))

rez\_mult\_2 = rez\_mult\_2[0] + rez\_mult\_2[2:]

print('Результат:', rez\_mult\_2)

print('Нормализация.', end=' ')

while True:

if rez\_mult\_2[1] == '1':

break

rez\_mult\_2 = rez\_mult\_2[0] + rez\_mult\_2[2:] + '0'

rez\_paradok = subtraction(rez\_paradok, '0' \* (P - 1) + '1', True)

if x1\_paradok == 'OverFlow':

print('Сигнал о потере значимости.')

return 'OverFlow'

print('Результат:', rez\_mult\_2[0], rez\_paradok, rez\_mult\_2[2:])

print('Округление.', end=' ')

print('Результат:', rez\_mult\_2[0], rez\_paradok, rez\_mult\_2[2:M + 2])

return rez\_mult\_2[0] + rez\_paradok + rez\_mult\_2[2:M + 2]

# деление фикс

def division\_fix(a, b):

znak1 = a[0]

znak2 = b[0]

a = decimal('0' + a[1:])

b = decimal('0' + b[1:])

buf = str(a / b)

befordot = bin(math.floor(float(buf)))[2:]

afterdot = afterdot\_to\_bun(buf[len(str(math.floor(float(buf)))) + 1:])

buf = befordot + afterdot

buf = buf + '0' \* (M - len(buf))

if znak1 == znak2:

return '0' + buf

return '1' + buf

# деление с плавающей точкой

def div(x1, x2, x1\_inp, x2\_inp):

print('Деление:')

print('x1=0?', end=' ')

if float(x1\_inp) == 0:

print('Да.')

return ' 0'

print('Нет.')

print('x2=0?', end=' ')

if float(x2\_inp) == 0:

print('Да.')

return 'infinity'

print('Нет.')

x1\_znak = x1[0]

x1\_paradok = x1[1:P + 1]

x1\_mantisa = '1' + x1[P + 1:]

x2\_znak = x2[0]

x2\_paradok = x2[1:P + 1]

x2\_mantisa = '1' + x2[P + 1:]

print('Вычитание порядков.', end=' ')

rez\_paradok = subtraction(x1\_paradok, x2\_paradok, False)

print('Результат:', rez\_paradok)

print('Сложение смещения.', end='')

rez\_paradok = sum\_fix(rez\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False)

rez\_paradok = sum\_fix(rez\_paradok, '0' \* (P - 1) + '1', False)

print('Результат:', rez\_paradok)

print('Переполнение порядка?', end=' ')

if rez\_paradok == 'OverFlow':

print('Да.')

print('Сигнал о переполнениию')

return 'OverFlow'

print('Нет.')

print('Потеря значимости порядка?', end=' ')

if rez\_paradok == 'OverFlow':

print('Да.')

print('Сигнал о потере значимости.')

return 'OverFlow'

print('Нет.')

print('Деление мантисс.', end=' ')

rez\_div\_2 = division\_fix((x1\_znak + x1\_mantisa), (x2\_znak + x2\_mantisa))

print('Результат:', rez\_div\_2)

print('Нормализация.', end=' ')

while True:

if rez\_div\_2[1] == '1':

break

rez\_div\_2 = rez\_div\_2[0] + rez\_div\_2[2:] + '0'

rez\_paradok = subtraction(rez\_paradok, '0' \* (P - 1) + '1', True)

if x1\_paradok == 'OverFlow':

print('Сигнал о потере значимости.')

return 'OverFlow'

print('Результат:', rez\_div\_2[0], rez\_paradok, rez\_div\_2[2:])

print('Округление.', end=' ')

print('Результат:', rez\_div\_2[0], rez\_paradok, rez\_div\_2[2:M + 2])

return rez\_div\_2[0] + rez\_paradok + rez\_div\_2[2:M + 2]

# перевод с 2 в 10

def \_2\_to\_10(x):

x\_paradok = x[1:P + 1]

x\_mantisa = '1' + x[P + 1:]

count = 0

minus = 1

if x\_paradok[0] == '1':

x\_paradok = subtraction(x\_paradok, '0' + bin(pow(2, P - 1) - 1)[2:], False)

else:

x\_paradok = subtraction('0' + bin(pow(2, P - 1) - 1)[2:], x\_paradok, False)

minus = 0

while True:

if int(x\_paradok) == 0:

break

x\_paradok = subtraction(x\_paradok, '0' \* (P - 1) + '1', False)

count += 1

buf = 0.0

count -= minus

if minus == 0:

x\_mantisa = '0' \* (count + 1) + x\_mantisa

count \*= minus

for i in range(M):

# print(x\_mantisa[i] + '\*2^' + str(count - i), buf)

buf += int(x\_mantisa[i]) \* pow(2, count - i)

print('Ответ:', '-' + str(buf) if x[0] == '1' else '+' + str(buf))

x1\_inp, x1 = input\_num('1')

x2\_inp, x2 = input\_num('2')

print('\nПредставление x1:', x1[0], x1[1:P + 1], x1[P + 1:])

print('Представление x2:', x2[0], x2[1:P + 1], x2[P + 1:])

summ = sum\_or\_sub(True, x1, x2, x1\_inp, x2\_inp)

if summ == 'OverFlow':

print('Ответ: OverFlow')

else:

print('Ответ:', summ[0], summ[1:P + 1], summ[P + 1:])

\_2\_to\_10(summ)

sub = sum\_or\_sub(False, x1, x2, x1\_inp, x2\_inp)

if sub == 'OverFlow':

print('Ответ: OverFlow')

else:

print('Ответ:', sub[0], sub[1:P + 1], sub[P + 1:])

\_2\_to\_10(sub)

multi = mult(x1, x2, x1\_inp, x2\_inp)

if multi[0] != '0' and multi[0] != '1':

print('Ответ:', multi)

else:

print('Ответ:', multi[0], multi[1:P + 1], multi[P + 1:])

\_2\_to\_10(multi)

print('=' \* 50)

divv = div(x1, x2, x1\_inp, x2\_inp)

if divv[0] != '0' and divv[0] != '1':

print('Ответ:', divv)

else:

print('Ответ:', divv[0], divv[1:P + 1], divv[P + 1:])

\_2\_to\_10(divv)

print('=' \* 50)