Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра информатики

Отчет по практическому занятию №3

Программирование арифметического сопроцессора

4

Выполнил:

студент гр. 153503

Вергасов В. М.

Проверила:

Калиновская А. А.

Минск 2023

**Цель работы**

Ознакомиться с предназначением арифметического сопроцессора. Рассмотреть строение и основные принципы работы сопроцессора. Изучить команды, доступные при использовании сопроцессора. Научиться работать с сопроцессором в ходе выполнения лабораторного задания.

1. **Постановка задачи**

**Задание к лабораторной работе 7**

Написать программу, находящую решение квадратного уравнения с помощью сопроцессора.

**Задание к лабораторной работе 8**

Значение аргумента изменяется от до с шагом . Для каждого найти значения функции , суммы и число итераций , при котором достигается требуемая точность . Результат вывести в виде таблицы. Значения и вводятся с клавиатуры.

1. **Теоретические сведения**

Использование сопроцессора позволяет значительно ускорить работу программ, выполняющих расчеты с высокой точностью, тригонометрические вычисления и обработку информации, которая должна быть представлена в виде действительных чисел. Сопроцессор подключается к системной шине параллельно с центральным процессором (CPU) и может работать только совместно с ним. Все команды попадают в оба процессора, а выполняет каждый свои. Сопроцессор не имеет своей программы и не может осуществлять выборку команд и данных. Это делает центральный процессор. Сопроцессор перехватывает с шины данные и после этого реализует конкретные действия по выполнению команды. Два процессора работают параллельно, что повышает эффективность системы. Но возникают ситуации, когда их работа требует синхронизации (из-за разницы во времени выполнения команд).

1. **Программная реализация**

В качестве средств для написания программы используются язык C++ и среда разработки Microsoft Visual Studio.

**Результаты:**

*Лабораторная работа 7.*

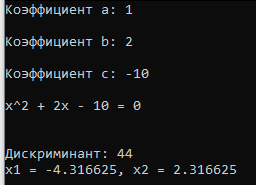
****

Рисунок 1. Квадратное уравнение с двумя корнями

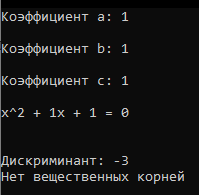


Рисунок 2. Квадратное уравнение, не имеющее корней

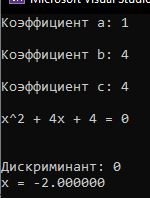


Рисунок 3. Квадратное уравнение с одним корнем

*Лабораторная работа 8.*

Была выбрана функция .

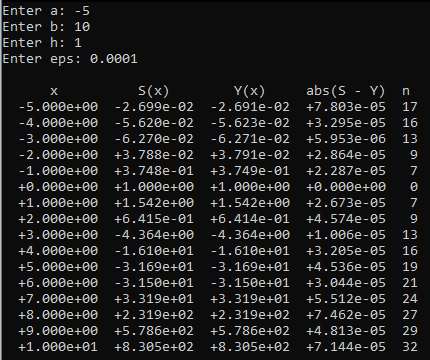


Рисунок 4. Результат выполнения программы

1. **Выводы**

В ходе выполнения лабораторной работы были изучены понятия арифметического сопроцессора, сопроцессорных конфигураций, программной модели сопроцессора. Выяснены режимы работы и состояния арифметического сопроцессора, система команд сопроцессора, особенности задания команд, различные группы команд сопроцессора (команды управления, команды передачи данных, команды загрузки, команды запоминания, арифметические команды).

**Приложение 1. Исходный код программы (ЛР 7)**

**#include <windows.h>**

**#include <iostream>**

**#include <string>**

**#include <functional>**

**#include <regex>**

**std::string Solve(long double a, long double b, long double c) {**

**const long double LD\_4 = 4.0L;**

**if (a == 0) {**

**return "Уравнение не является квадратным";**

**}**

**long double D;**

**\_\_asm**

**{**

**FLD b**

**FLD b**

**FMUL**

**FLD LD\_4**

**FLD a**

**FLD c**

**FMUL**

**FMUL**

**FSUB**

**FSTP D**

**}**

**std::cout << "\nДискриминант: " << D << '\n';**

**if (!isfinite(D)) {**

**return "Дискриминант не является конечным числом";**

**}**

**if (D < 0.0L) {**

**return "Нет вещественных корней";**

**}**

**if (D == 0.0L) {**

**long double x;**

**\_\_asm**

**{**

**FLD b**

**FCHS**

**FLD a**

**FLD a**

**FADD**

**FDIV**

**FSTP x**

**}**

**return "x = " + std::to\_string(x);**

**}**

**if (D > 0.0L) {**

**long double x1, x2;**

**\_\_asm**

**{**

**FLD b**

**FCHS**

**FLD D**

**FSQRT**

**FSUB**

**FLD a**

**FLD a**

**FADD**

**FDIV**

**FSTP x1**

**}**

**\_\_asm**

**{**

**FLD b**

**FCHS**

**FLD D**

**FSQRT**

**FADD**

**FLD a**

**FLD a**

**FADD**

**FDIV**

**FSTP x2**

**}**

**return "x1 = " + std::to\_string(x1) + ", " +**

**"x2 = " + std::to\_string(x2);**

**}**

**return "Что-то пошло не так";**

**}**

**int main() {**

**SetConsoleCP(1251);**

**SetConsoleOutputCP(1251);**

**long double a, b, c;**

**std::cout << "Коэффициент а: ";**

**std::cin >> a;**

**std::cout << "\nКоэффициент b: ";**

**std::cin >> b;**

**std::cout << "\nКоэффициент c: ";**

**std::cin >> c;**

**std::cout << '\n';**

**if (a != 1) {**

**std::cout << ((a == -1) ? '-' : a);**

**}**

**std::cout << "x^2 " << ((b < 0) ? '-' : '+') << ' ' << std::abs(b) << "x "**

**<< ((c < 0) ? '-' : '+') << ' ' << abs(c) << " = 0\n\n" << Solve(a, b, c) << '\n';**

**return 0;**

**}**

**Приложение 2. Исходный код программы (ЛР 8)**

**#include <windows.h>**

**#define \_USE\_MATH\_DEFINES**

**#include <cmath>**

**#include <functional>**

**#include <iomanip>**

**#include <iostream>**

**#include <regex>**

**#include <string>**

**void Solve(long double a, long double b, long double h, long double eps) {**

**std::cout << '\n';**

**std::cout << " x S(x) Y(x) abs(S - Y) n " << '\n';**

**for (long double x = a; x < b + h / 2; x += h) {**

**int n = 0;**

**long double Y, S, f, p, intp, fracp, trash, fi = M\_PI / 4, x\_powed = 1;**

**// Y = pow(exp, x \* cos(fi)) \* cos(x \* sin(fi));**

**\_\_asm**

**{**

**FLD fi**

**FCOS**

**FLD x**

**FMUL**

**FLDL2E**

**FMUL**

**FSTP p**

**}**

**\_\_asm {**

**FLD p**

**FLD1**

**FSCALE**

**FSTP intp**

**FSTP trash**

**}**

**\_\_asm**

**{**

**FLD1**

**FLD p**

**FPREM**

**F2XM1**

**FADD**

**FSTP fracp**

**}**

**\_\_asm {**

**FLD intp**

**FLD fracp**

**FMUL**

**FLD fi**

**FSIN**

**FLD x**

**FMUL**

**FCOS**

**FMUL**

**FSTP Y**

**}**

**\_\_asm**

**{**

**FLD1**

**FSTP S**

**FLD1**

**FSTP f**

**}**

**while (abs(S - Y) > eps) {**

**n++;**

**\_\_asm**

**{**

**FLD f**

**FILD n**

**FMUL**

**FSTP f**

**FLD x\_powed**

**FLD x**

**FMUL**

**FSTP x\_powed**

**}**

**\_\_asm {**

**FLD S**

**FILD n**

**FLD fi**

**FMUL**

**FCOS**

**FLD f**

**FDIV**

**FLD x\_powed**

**FMUL**

**FADD**

**FSTP S**

**}**

**}**

**std::cout << std::setprecision(3) << std::scientific << std::showpos << " " << x << " "**

**<< S << " " << Y << " " << abs(S - Y) << " " << std::noshowpos**

**<< std::right << std::setw(2) << n << " \n";**

**}**

**}**

**int main() {**

**float a;**

**float b;**

**float h;**

**float eps;**

**std::cout << "Enter a: ";**

**std::cin >> a;**

**std::cout << "Enter b: ";**

**std::cin >> b;**

**std::cout << "Enter h: ";**

**std::cin >> h;**

**std::cout << "Enter eps: ";**

**std::cin >> eps;**

**Solve(a, b, h, eps);**

**return 0;**

**}**