

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ
по лабораторной работе №8
по дисциплине «Организация ЭВМ и систем»
ТЕМА: ОБРАБОТКА ВЕЩЕСТВЕННЫХ ЧИСЕЛ. ПРОГРАММИРОВАНИЕ
МАТЕМАТИЧЕСКОГО СОПРОЦЕССОРА.

Студент гр. 1303

Насонов Я.К.

Преподаватель

Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Получить навыки программирования на языке Ассемблера. Изучить работу с вещественными числами на языке Ассемблера.

Задание.

Разработать подпрограмму на языке Ассемблера, обеспечивающую вычисление заданной математической функции с использованием математического сопроцессора. Подпрограмма должна вызываться из головной программы, разработанной на языке С. При этом должны быть обеспечены заданный способ вызова и обмен параметрами. Альтернативный вариант реализации: разработать на языке Ассемблера фрагмент программы, обеспечивающий вычисление заданной математической функции с использованием математического сопроцессора, который включается по принципу in-line в программу, разработанную на языке С.

Выполнить трансляцию программы с подготовкой ее ассемблерной версии и отладочной информации. Для выбранного контрольного набора исходных данных прогнать программу под управлением отладчика. При этом для каждой команды сопроцессора следует фиксировать содержимое используемых ячеек памяти, регистров ЦП и численных регистров сопроцессора до и после выполнения этой команды. Проверить корректность выполнения вычислений для нескольких наборов исходных данных.

Вариант 15:

* function

Name poly - generates a polynomial from arguments

Usage double poly(double x, int n, double c []);

Prototype in math.h

Description poly generates a polynomial in x, of degree n, with coefficients c[0], c[1], ..., c[n].

For example, if n=4 the generated polynomial is $c[4].x^4 + c[3].x^3 + c[2].x^2 + c[1].x + c[0]$

The polynomial is calculated using Horner's method:

$$\text{polynom} = (..((x.c[n] + c[n-1]).x + c[n-2])..).x + c[0]$$

Return value poly returns the value of the polynomial as evaluated for the given x.

Выполнение работы.

На языке Си была разработана программа, в которой сначала происходит считывание необходимых данных от пользователя (значения x, значения массива констант coefficients[]). Далее на языке Ассемблера был разработан фрагмент программы, обеспечивающий вычисление заданной математической функции с использованием математического сопроцессора, который включается по принципу in-line в программу.

Сначала на вершину математического стека кладем значение x. Далее в цикле по количеству констант по методу Горнера вычисляем значение полинома: значение вершины математического стека (st(0)) умножается на следующий за ним элемент в стеке (st(1)) с помощью инструкции fmul. С помощью инструкции fadd складываем значение вершины математического стека с текущей константой (это значение присваивается вершине математического стека). Далее из верхушки стека записываем значение в переменную result с помощью инструкции fst.

Входные данные: x = 1.1 n = 3 constants = 1.1, 1.2, 1.3

Таблица 1 – результат прогона ассемблерного модуля в отладчике

Символический код команды	Содержимое регистров и ячеек памяти	
	До выполнения	После выполнения
fld qword ptr x	EIP = 005126C0 ST0 = +0.0000000000000000e+0000 STAT = 0000 TAGS = FFFF	EIP = 005126C3 ST0 = +1.1000000000000000e+0000 STAT = 3800 TAGS = 3FFF
fldz	EIP = 005126C3 ST0 = +1.1000000000000000e+0000 ST0 = +0.0000000000000000e+0000	EIP = 005126C5 ST0 = +0.0000000000000000e+0000 ST1 = +1.1000000000000000e+0000

	STAT = 3800 TAGS = 3FFF	STAT = 3000 TAGS = 1FFF
mov edi, amount	EDI = 00B6F5E0 EIP = 005126C3	EDI = 00000003 EIP = 005126C8
mov esi, coefficients	ESI = 00B6F5F8 EIP = 005126C8	ESI = 00EB1280 EIP = 005126CB
test edi, edi	EIP = 005126CB	EIP = 005126CD
je skip	EIP = 005126CD	EIP = 005126CF
mov ecx, edi	ECX = 00000000	ECX = 00000003
fmul st(0), st(1)	EIP = 005126D1	EIP = 005126D3
fadd qword ptr[esi + ecx * 8 - 8]	EIP = 005126D3 TAGS = 1FFF ST0 = +0.0000000000000000e+0000	EIP = 005126D7 TAGS = 0FFF ST0 = +1.3000000000000000e+0000
loop poly	EIP = 005126D7 ECX = 00000003	EIP = 005126D1 ECX = 00000002
fmul st(0), st(1)	EIP = 005126D1 ST0 = +1.3000000000000000e+0000 STAT = 3000	EIP = 005126D3 ST0 = +1.4300000000000001e+0000 STAT = 3020
fadd qword ptr[esi + ecx * 8 - 8]	EIP = 005126D3 ST0 = +1.4300000000000001e+0000	EIP = 005126D7 ST0 = +2.6299999999999998e+0000
loop poly	EIP = 005126D7 ECX = 00000002	EIP = 005126D1 ECX = 00000001
fmul st(0), st(1)	EIP = 005126D1 ST0 = +2.6299999999999998e+0000 STAT = 3020	EIP = 005126D3 ST0 = +2.8930000000000002e+0000 STAT = 3220
fadd qword ptr[esi + ecx * 8 - 8]	EIP = 005126D3 ST0 = +2.8930000000000002e+0000 STAT = 3220	EIP = 005126D7 ST0 = +3.9930000000000003e+0000 STAT = 3020
loop poly	EIP = 005126D7 ECX = 00000002	EIP = 005126D9 ECX = 00000000

fst qword ptr result	EIP = 005126D9	EIP = 005126DC
----------------------	----------------	----------------

Исходный код программы см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 2.

Таблица 2 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	5 4 1.4 4.8 2.7 -7.2	-807.1	
2.	2 2 9.9 -8.1	-6.3	
3.	9.1 4 1 2 3 4	3281.91	

Выводы.

В ходе выполнения лабораторной работы были получены навыки программирования на языке Ассемблера.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab8.cpp

```
#include <iostream>
#include <stdlib.h>

using namespace std;

int main() {
    double x;
    cout << "Enter x:\n";
    cin >> x;

    int amount;
    cout << "Enter amount of coefficients:\n";
    cin >> amount;

    double* coefficients = new double[amount];
    cout << "Enter coefficients:\n";
    for (int i = 0; i < amount; ++i) {
        cout << i + 1 << " ";
        cin >> coefficients[i];
    }

    double result = 0;

    __asm {
        fld qword ptr x ; load a real number into the stack FPU
        fldz ; load the +0.0 into the stack FPU
        mov edi, amount
        mov esi, coefficients
        test edi, edi
        je skip ; amount = 0
        mov ecx, edi
        poly:
            fmul st(0), st(1)
            fadd qword ptr[esi + ecx * 8 - 8]
            loop poly
        skip:
            fst qword ptr result
    };

    cout << "Result: " << result;
    delete[] coefficients;

    return 0;
}
```