МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ

БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет автоматики и вычислительной техники

Кафедра радиоэлектронных средств

Отчет по дисциплине

«Цифровые устройства и микропроцессоры»

Лабораторная работа №3

«ИСПОЛЬЗОВАНИЕ МАТЕМАТИЧЕСКОГО СОПРОЦЕССОРА»

Вариант №5

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИКТб-3301 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | А.В.Герасименко |
| Проверил: доцент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | М.А. Земцов |

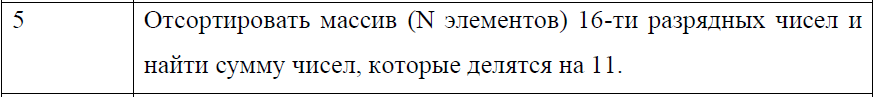
Киров 2023

**Цель работы:** изучение принципов выполнения арифметических команд с помощью математического сопроцессора FPU микропроцессоров с архитектурой x86.

**Ход работы:**

Условие варианта представлено в таблице 1.

Таблица 1 – Условие варианта



**Листинг кода программы:**

.686

.model flat,stdcall

.stack 100h

.data

Sum dd 0.0

N dd 10

F dd 11.0

array dd 83.0, 11.0, 121.0, 33.0, 41.0, 5.0, 10.0, 18.0, 17.0, 100.0 ;массив

.code

ExitProcess PROTO STDCALL :DWORD

Start:

;сортировка массива пузырьковым методом на убывание

xor ecx, ecx

push n

pop ecx

cycl\_1: push ecx ; Основной цикл

mov eax, n

dec eax

push eax

pop ecx

lea esi, array

cycl\_2: mov eax, [esi] ; Вложенный цикл

mov ebx, [esi+4]

cmp eax, ebx

js swap

just\_: add esi, 4 ; Метка для удобства после swap

loop cycl\_2

pop ecx

loop cycl\_1

jmp l

swap: mov [esi], ebx ; Обмен элементами

mov [esi+4], eax

jmp just\_

l:

FINIT ;инициализация математического сопроцессора

mov esi, 0 ; для индексации массива

mov ecx, N ; счетчик итераций

; нахождение суммы элементов, делящихся на 11

@cycl3:

FNCLEX

FLDZ ; загрузка нуля

FLD F ; загрузка числа 11

FLD array[esi\*4] ; загрузка элемента массива

FPREM1 ; остаток от деления

FCOMI ST, ST(2) ; сравнение остатка с нулем

je L1 ; условные переходы по метке

jg G

jl G

L1: ; если остаток равен нулю

FLD Sum ;загрузка переменной суммы

FLD array[esi\*4] ; загрузка элемента массива

FADD ST, ST(1) ; суммирование

FST Sum

FFREE ST(0) ; очистка регистра 1

FFREE ST(1)

FFREE ST(2)

FFREE ST(3)

FFREE ST(4)

FFREE ST(5)

FFREE ST(6)

FFREE ST(7)

FNCLEX

G: ; все остальные условия

inc esi ; для перемещения по массиву

loop @cycl3

exit:

Invoke ExitProcess,1

End Start

**Верификация программы:**

**1)** Расчет вручную:

Начиная с начала массива просматриваются попарно по 2 элемента (первый со вторым, второй с третим, третий с четвертым и т.д.). Если второй элемент в паре больше первого элемента, он ставится на место первого, а первый на место второго. Это делается для всех элементов. После того, до конца массив был пройден до конца (произведено сравнение предпоследнего и последнего элемента и сделан обмен, если нужно), происходит проверка, был ли хотя бы один обмен. Если да, значит массив не отсортирован и начинается все сначала. Повторение таких проходов до тех пор, пока не будет так, что проверены попарно все элементы от начала до конца, а обмена ни одного не было. Таким образом элементы с самыми маленькими значениями потихоньку перемещаются слева направо.

Дан массив из 10 чисел: 18, 11, 121, 33, 41, 5, 10, 55, 17, 100.

Массив после сортировки по убыванию: 121, 100, 83, 41, 33, 18, 17, 11, 10, 5.

Далее суммирование элементов, делящихся на 11. Числа 121, 55, 33, 11 делятся на 11. Их сумма равна 220.

**2)** Расчет в программе:

Дан массив из 10 чисел: 18, 11, 121, 33, 41, 5, 10, 55, 17, 100.

Ниже представлены числа отсортированного массива по убыванию.



Рисунок 1 – 1 элемент массива



Рисунок 2 – 2 элемент массива



Рисунок 3 – 3 элемент массива



Рисунок 4 – 4 элемент массива



Рисунок 5 – 5 элемент массива



Рисунок 6 – 6 элемент массива



Рисунок 7 – 7 элемент массива



Рисунок 8 – 8 элемент массива



Рисунок 9 – 9 элемент массива



Рисунок 10 – 10 элемент массива

Далее находится сумма чисел, делящихся на 11. На скрине ниже представлено вычисление остатка.

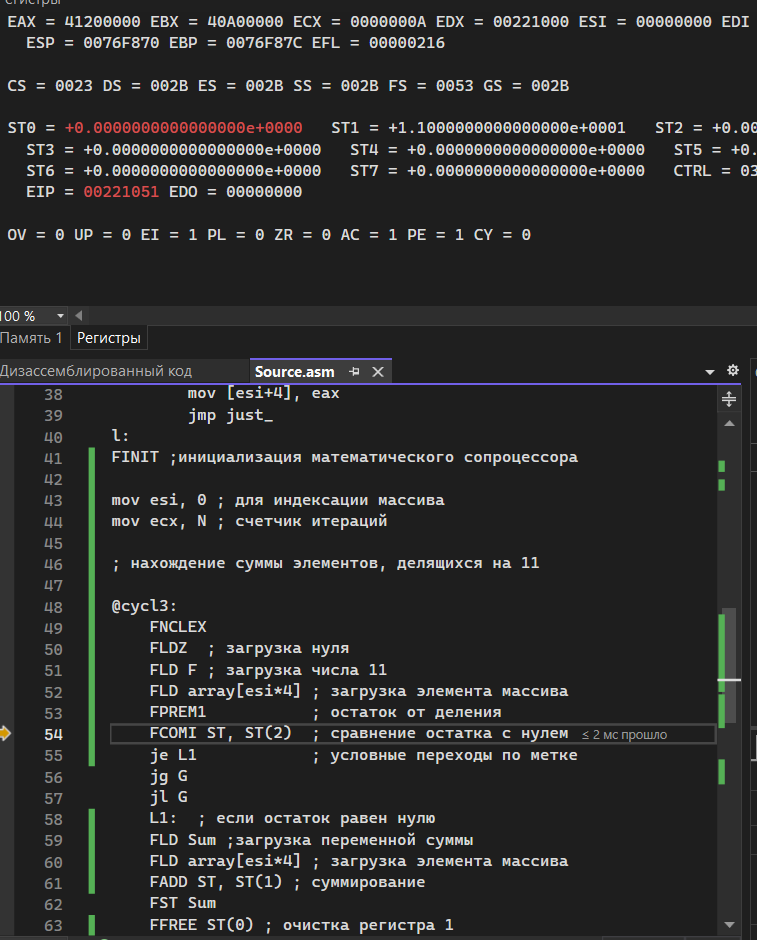


Рисунок 11 – Сравнение остатка от деления с нулем

Ниже представлен скрин, когда остаток равен нулю. Суммируются элементы, делящиеся на 11.

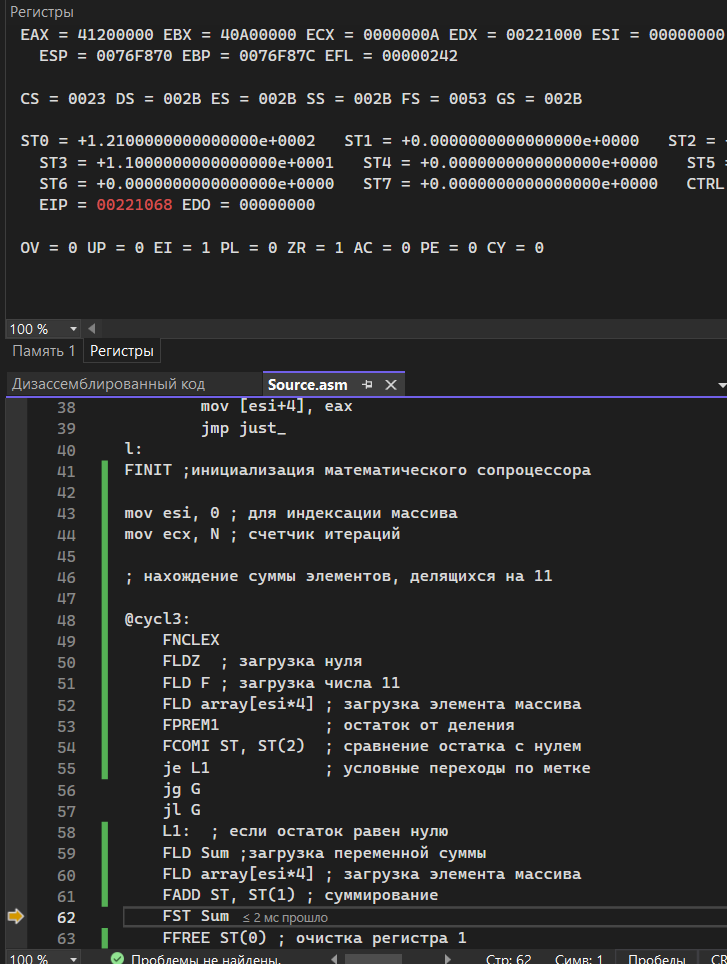


Рисунок 12 – Суммирование элементов, когда остаток равен нулю

Ниже представлен скрин, когда число не делится нацело на 11.

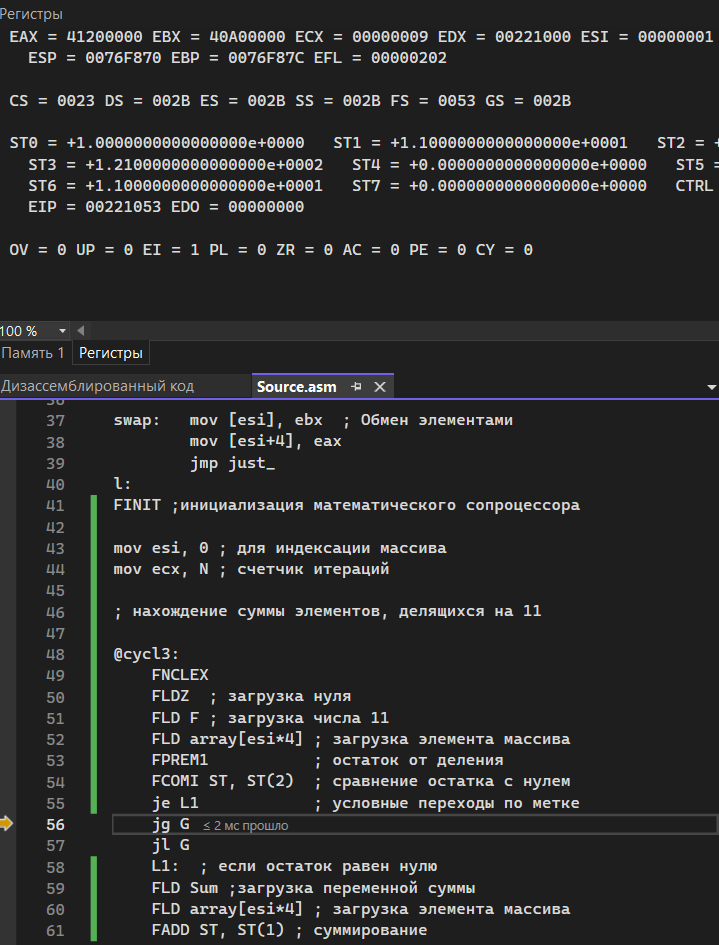


Рисунок 13 – Случай, когда остаток от деления не равен нулю

Сумма элементов массива равна 220 в программном расчете



Рисунок 14 – Результат

**Вывод:** в ходе данной лабораторной работы изучены команды математического сопроцессора, служебные регистры swr, cwr, twr, а также способ задания циклов и ветвлений в сопроцессоре, программа верифицирована, и программа показала верный результат.

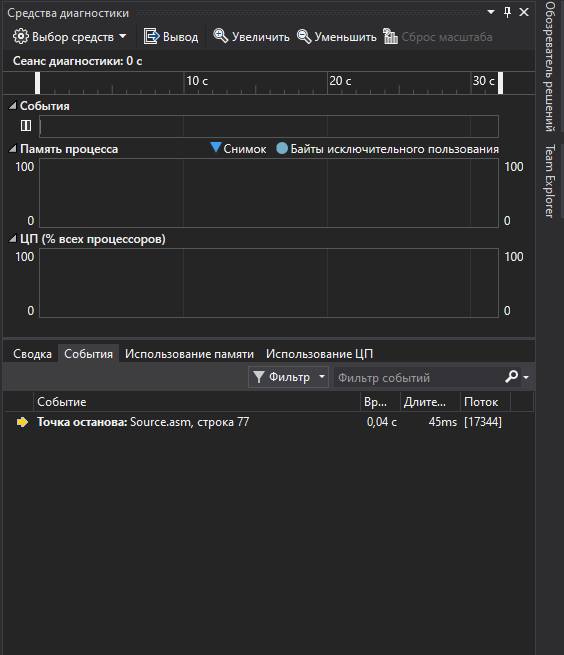


Рисунок 16 – Время выполнения программы, написанной на Ассемблере

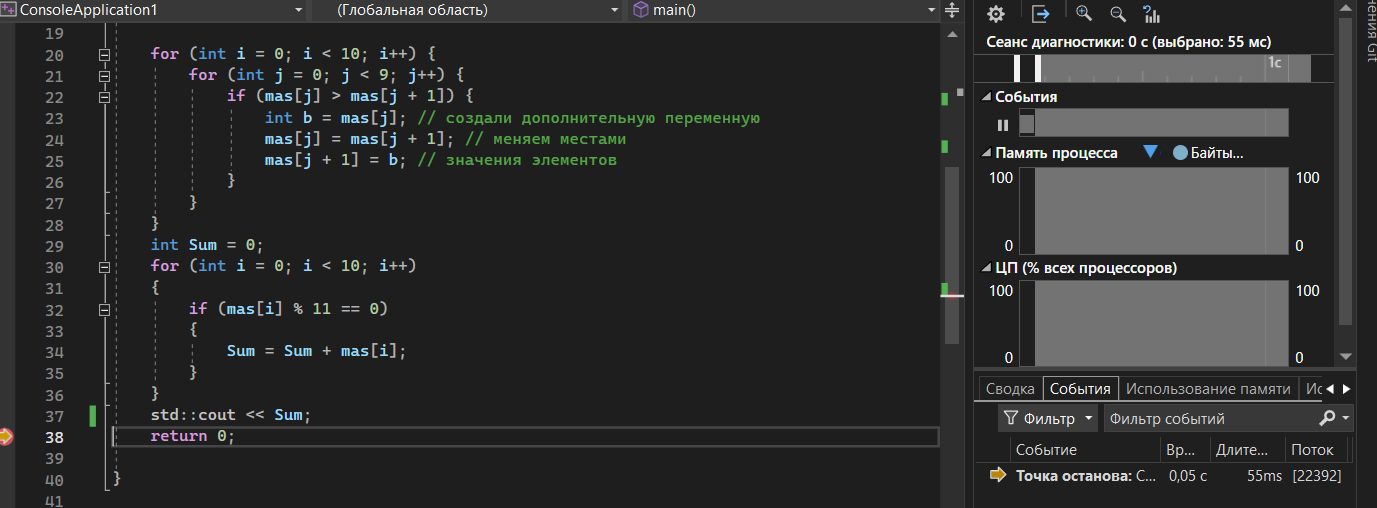


Рисунок 16 – Время выполнения программы, написанной на C++

Скорость выполнения программы на языке Ассемблере выше по сравнению с написанным на С++. Это связано с тем, что код на Ассемблере легче, чем код С++, т.к. там есть дополнительные операторы. Чем больше команд математического сопроцессора в коде, тем он более производителен.