**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Кубанский Государственный Университет**

Кафедра информационных технологий

ОТЧЕТ

о выполнении лабораторной работы №3

по дисциплине «Системы реального времени»

Выполнил: \_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ст. гр. 44 Ерюшев И.А. \_\_\_\_\_\_\_\_

Проверил: \_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

доц. каф. ИТ Полетайкин А.Н. \_ \_

Краснодар

2024

**ЛАБОРАТОРНАЯ РАБОТА 3**

**ОРГАНИЗАЦИЯ ОБРАБОТКИ ЧИСЛОВЫХ ОДНОМЕРНЫХ МАССИВОВ**

Цель: изучение команд организации циклов и способов косвенной адресации данных памяти в микропроцессоре i486; приобретение практических навыков составления программ обработки одномерных массивов, освоение методов анализа трудоемкости и ресурсной сложности алгоритмов обработки одномерных числовых массивов.

Задание

1. На основании индивидуального задания (табл. 1) составить программу для обработки элементов одномерного массива. Длина элементов исходного массива равна DW. Значения элементов исходного массива задать в сегменте данных (см. пример выше). Длину элементов результирующего массива, если он необходим, выбрать самостоятельно.
2. Получить загрузочный модуль и протестировать выполнение программы.
3. Выполнить расчет времени выполнения программы.

*Таблица 1*

| № вар. | Задание на обработку |
| --- | --- |
| 7 | Из элементов массива А[20] сформировать массив В[19], элементы которого равны –1, если каждый предыдущий элемент массива А меньше последующего; 0, если элементы равны; 1, если А[i]>А[i+1]. Подсчитать количество нулевых элементов массива В |

**ХОД РАБОТЫ**

**Задача 1.**

Была составлена программа для обработки элементов массива (табл. 1). Код программы:

.486

INCLUDE \MASM32\INCLUDE\IO.ASM

.DATA

A DW 1, 1, 3, 4, 5, 6, 6, 6, 4, 3, 2, 1, 1, 2, 1, 3, 2, 2, 1, 1

B DW 19 DUP(?)

Len DW 20

CountZero DW 0

.code

Start:

XOR EAX, EAX ; текущий элемент массива

XOR EBX, EBX ; число нулевых элементов

SUB ESI, ESI ; индекс элемента

XOR CX, CX ; счетчик для перемещения по циклу

MOV CX, Len

printA: ; вывод массива A

MOV AX, A[SI]

ADD SI, 2

outint16 AX

newline

DEC CX

JNE printA

MOV CX, 19

SUB SI, SI

MOV BX, A[SI]

Loop\_start:

MOV AX, A[SI]

ADD SI, 2

CMP [A+SI+2], AX

JE Equal

JG Greater

JL Less

Continue:

JMP Loop\_condition

Equal:

MOV B[SI], 0

ADD BX, 1

JMP Continue

Greater:

MOV B[SI], -1

JMP Continue

Less:

MOV B[SI], 1

JMP Continue

Loop\_condition:

DEC CX

JNE Loop\_start

newline

XOR AX, AX

SUB SI, SI

MOV CX, 19

printB: ; вывод массива B

MOV AX, B[SI]

ADD SI, 2

outint16 AX

newline

DEC CX

JNE printB

newline

outint16 BX

end Start

**Задача 2.**

Протестируем полученную программу в отладчике:

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание

Рисунок 1 – вывод программы в консоль

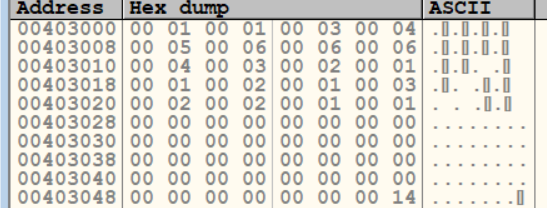


Рисунок 2 – дамп памяти до выполнения программы

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 3 – дамп памяти после выполнения программы

**Задача 3.**

Произведем подсчет времени выполнения программы, для этого рассчитаем количество процессорных тактов, необходимых для выполнения программы:

Start:

XOR EAX, EAX ; 3

XOR EBX, EBX ; 3

SUB ESI, ESI ; 3

XOR CX, CX ; 3

MOV CX, Len ; 8

printA: ; цикл выполнится 20 раз

MOV AX, A[SI] ; 3

ADD SI, 2 ; 4

outint16 AX

newline

DEC CX ; 2

JNE printA ; 15

MOV CX, 19 ; 8

SUB SI, SI ; 3

MOV BX, A[SI] ; 8

Loop\_start: ; цикл выполнится 19 раз

MOV AX, A[SI] ; 3

ADD SI, 2 ; 4

CMP [A+SI+2], AX ; 9

JE Equal ; 15 Переход, если равно/нуль

JG Greater ; 15 Переход, если больше/не меньше или равно

JL Less ; 15 Переход, если меньше/не больше или равно

Continue:

JMP Loop\_condition ; 15  Команда прыжка к метке Loop\_condition, если все элементы массива A уже были проверены.

Equal:

MOV B[SI], 0 ; 4

ADD BX, 1 ; 4

JMP Continue ; 15

Greater:

MOV B[SI], -1 ; 4

JMP Continue ; 15

Less:

MOV B[SI], 1 ; 4

JMP Continue ; 15

Loop\_condition:

DEC CX ; 2

JNE Loop\_start ; 15 Железострукторный команда, которая выполняет прыжок, если предыдущая операция была проведена над разными элементами массива A.

newline

XOR AX, AX ; 3

SUB SI, SI ; 3

MOV CX, 19 ; 4

printB: ; цикл выполнится 19 раз

MOV AX, B[SI] ; 8

ADD SI, 2 ; 4

outint16 AX

newline

DEC CX ; 2

JNE printB ; 15

newline

outint16 BX

end Start

C учетом выполнения ветки Equal для каждого элемента массива программа потребует процессорных 2391 тактов для полного выполнения. Что при частоте процессора в 2.1 ГГц займёт 1138 наносекунд.

**Вывод:** изучили команды организации циклов и способы косвенной адресации данных памяти в микропроцессоре i486; приобрели практические навыки составления программ обработки одномерных массивов, освоили методы анализа трудоемкости и ресурсной сложности алгоритмов обработки одномерных числовых массивов.

1. XOR EAX, EAX: Сбрасывает регистр EAX, чтобы использовать его в качестве счетчика или хранилища данных.
2. XOR EBX, EBX: Аналогично, сбрасывает регистр EBX.
3. SUB ESI, ESI: Сбрасывает индексный регистр ESI.
4. XOR CX, CX: Сбрасывает счетчик ЦИКЛА.
5. MOV CX, Len: Загружает значение из переменной Len (длина массива A) в счетчик ЦИКЛА.
6. outint16 AX: Выводит содержимое регистра AX на экран.
7. newline: Выводит символ переноса строки на экран.
8. JNE: Железострукторный команда, которая выполняет прыжок, если предыдущая операция была проведена над разными элементами массива A.
9. Loop\_start: Метка начала цикла сравнения и определения положения элементов массива A.
10. Loop\_condition: Метка конца цикла.
11. JNE Loop\_start: Железострукторный команда, которая выполняет прыжок, если предыдущая операция была проведена над разными элементами массива A.
12. JMP Loop\_condition: Команда прыжка к метке Loop\_condition, если все элементы массива A уже были проверены.
13. CMP [A+SI+2], AX: Сравнивает текущий элемент массива A с соседним элементом (следующим по массиву).
14. JE Equal: Если элементы равны, то выполняется команда прыжка к метке Equal.
15. JG Greater: Если текущий элемент больше, то выполняется команда прыжка к метке Greater.
16. JL Less: Если текущий элемент меньше, то выполняется команда прыжка к метке Less.
17. Continue: Метка, указывающая на продолжение работы цикла после выполнения команд Equal, Greater или Less.
18. Equal, Greater, Less: Метки, определяющие результаты сравнения элементов массива A.
19. outint16 BX: Выводит содержимое регистра BX на экран.
20. end Start: Завершает программу.