**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Кубанский Государственный Университет**

Кафедра информационных технологий

ОТЧЕТ

о выполнении лабораторной работы №2

по дисциплине «Системы реального времени»

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

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

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

доц. каф. ИТ Полетайкин А.Н. \_ \_

Краснодар

2024

**Изучение команд арифметических и логических операций**

Цель: изучение команд арифметических и логических операций, приобретение практических навыков при их использовании в различных вычислительных задачах.

**Задание**

1. Составить программу для расчета заданного арифметического выражения. Длину и значение переменных A, B, C выбрать самостоятельно. Константы, заданные в выражении, использовать в кодовом сегменте. Программа должна корректно работать при любых допустимых значениях переменных.

2. Описать команды умножения и деления, используемые в программе на предмет длины операндов, участвующих в операции. Охарактеризовать длину результата и место его хранения.

3. Получить загрузочный модуль и протестировать выполнение программы в отладчике.

4. На основе составленной программы выполнить следующие действия:

− загрузить в аккумулятор маскирующее слово, позволяющее определить заданную характеристику содержимого регистра DX;

− выполнить заданную проверку и ее результат сохранить в переменной RESULT, объявленной в сегменте данных.

5. Перекомпилировать загрузочный модуль и протестировать выполнение программы в отладчике.

6. Произвести расчет времени выполнения программы.

7. Сделать выводы.

Индивидуальные условия лабораторной работы представлены в таблице (Таблица 1):

|  |  |  |
| --- | --- | --- |
| Номер варианта | Заданные операции | |
| Арифметическая | Логическая |
| 7 | A\*64/(B+81)+C/42 | Определение наличия 0 в разряде 6 |

Таблица 1 – индивидуальные условия лабораторной работы

**Ход работы**

Код программы:

.486

.model flat

.data

const\_64 DB 64

const\_81 DB 81

const\_42 DB 42

param\_A DB 2

param\_B DB 19

param\_C DW 168

res\_A64 DW ?

res\_B81 DB ?

res\_A64B81 DB ?

res\_C42 DB ?

total\_res DW ?

res\_check DB ?

RESULT DB ?

.code

\_start:

mov al, param\_A

mov bl, const\_64

mul bl

mov res\_A64, ax

mov ah, param\_B

mov al, const\_81

add ah, al

mov res\_B81, ah

mov ax, res\_A64

mov bl, res\_B81

div bl

mov res\_A64B81, al

mov ax, param\_C

mov bl, const\_42

div bl

mov res\_C42, al

mov ah, res\_A64B81

mov al, res\_C42

add al, ah

mov bl, 0

adc bl, 0

mov ah, bl

mov total\_res, ax

and ax, 0040h

jz res\_0

mov res\_check, 0

jmp end\_res\_check

res\_0:

mov res\_check, 1

end\_res\_check:

mov ax, 0040h

and dx, ax

jz dx\_0

mov RESULT, 0

jmp end\_dx\_check

dx\_0:

mov RESULT, 1

end\_dx\_check:

ret

end \_start

Проверка корректности программы:

1. A\*64=2\*64=128 (80h)

2. B+81=19+81=100 (64h)

3. A\*64/(B+81)=128/100=1 (1h, остаток игнорируем)

4. C/42=168/42=4 (4h)

5. A\*64/(B+81)+C/42=1+4=5 (5h)

6. 5h = 0000 0000 0000 0101 -> 0 в разряде 6 есть (результат проверки 1)

7. 1000h = 0001 0000 0000 0000 -> 0 в разряде 6 есть (результат проверки 1)

На рисунках ниже представлен ход работы программы:

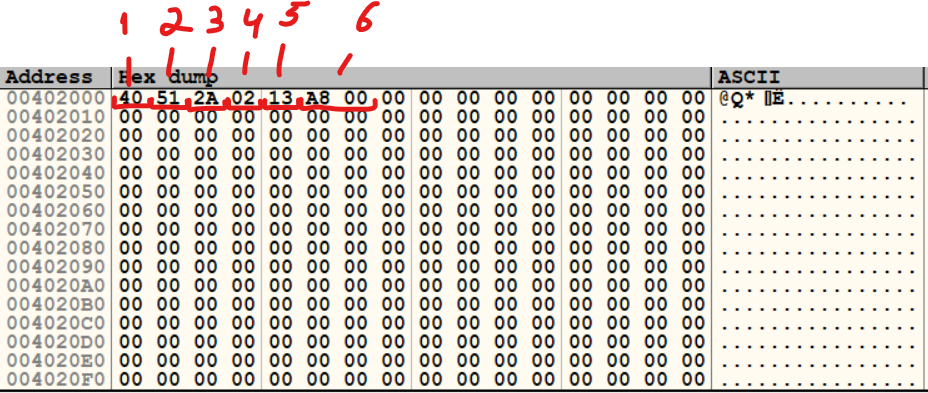


Рисунок 1 – старт работы программы

1, 2, 3 – константы (64, 81, 42)

4, 5, 6 – свободные параметры A, B, C (2, 19, 168)

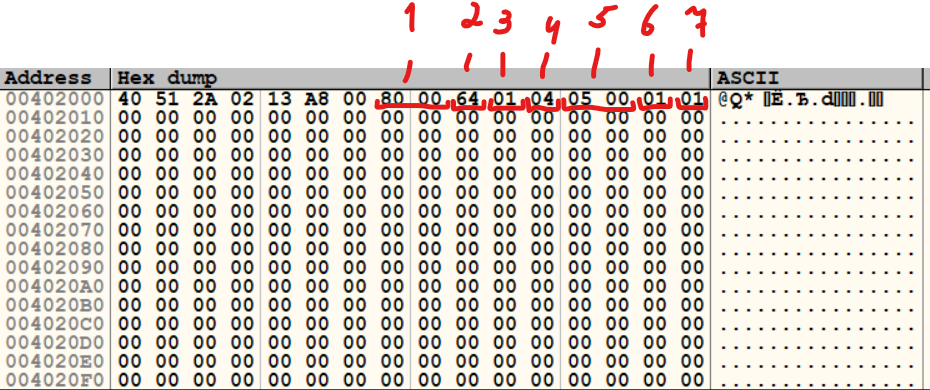


Рисунок 2 – окончание работы программы

1 – A\*64=2\*64=128 (80h)

2 – B+81=19+81=100 (64h)

3 – A\*64/(B+81)=128/100=1 (1h, остаток игнорируем)

4 – C/42=168/42=4 (4h)

5 – A\*64/(B+81)+C/42=1+4=5 (5h)

6, 7 – логические проверки результата операции и DX

Ниже представлен подсчёт времени выполнения программы:

Итоговое кол-во тактов: 1032

Время выполнения одного такта (тактовая частота процессора – 2.50 ГГц):



Время выполнения в наносекундах: 

Время выполнения в секундах: 

Вывод:

Ассемблер позволяет относительно удобно работать с арифметическими операциями и выполнять логические проверки. Для арифметики он поддерживает все основные операции (сложение, вычитание, умножение, деление, также имеет возможность проверки флагов переноса и заёма). Однако нужно предельно внимательно работать с форматами значений, так как не все операторы одинаково себя ведут при работе со значениями разных форматов.