МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждения высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ

**ЛАБОРАТОРНАЯ РАБОТА № 2**

по дисциплине

**«Машинно-ориентированное программирование»**

на тему:

**«Циклические программы»**

*Вариант № 3*

Выполнил:

Студент группы

КТбо2-8

|  |  |  |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Жалнин Д. И. |
|  | *подпись* |  |
|  |  |  |

Проверил:

ассистент кафедры

МОП ЭВМ

|  |  |  |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гуляев Н. А. |
|  | *подпись* |  |

Оценка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Таганрог 2020

# **1 ЦЕЛЬ РАБОТЫ**

* 1. **Дидактическая цель работы**

Ознакомление с основными методами составления программ, используемых циклические алгоритмы работы, на языке «Ассемблер», а также с наиболее часто используемыми для этого мнемониками.

**1.2 Практическая цель работы**

В рамках лабораторной работы необходимо разработать программу на языке ассемблера, алгоритм которой выполняет задачу согласно описанному индивидуальному заданию, скомпилировать и запустить код программы с помощью программного пакета «TASM».

# **2 ВАРИАНТ ЗАДАНИЯ**

**2.1 Общие требования**

Для всех вариантов требуется выполнить разработку программного модуля при помощи СРПО «Turbo Assembler», реализующего некоторую обработку массива, состоящего из байтов или 2-байтных слов. Выполнить отладку разработанного программного модуля при помощи программного отладчика «Turbo Debugger».

**2.2 Индивидуальное задание, вариант № 3**

Разработать программу, которая находит среднее арифметическое значение массива, состоящего из 2-байтных слов.

# **3 ХОД РАБОТЫ**

**3.1 Описание высокоуровневой реализации**

В ход работы была составлена программа на языке программирования «Ассемблер», используя модель памяти «small», в котором допускается наличие одного сегмента кода и единственного сегмента данных. Размер стека программы – 256 байт.

Перед началом основного сегмента кода располагается инициализация:

1. Константы «LEN», которая хранит в себе длину инициализируемого массива
2. Массива «MASS», в котором расположено 7 2-байтных слов
3. Неизвестной переменной «AVER», которую необходимо найти с помощью алгоритма программы.

Общая логика работы алгоритма сводится к циклическому суммированию всех элементов массива в регистре «AX» с последующим делением значения в регистре на длину массива.

Перед началом цикла в регистр «BX» заносится адрес 1го элемента массива «MASS». Для доступа к значению по адресу, который хранится в регистре, используются квадратные скобки ([BX]). Для перехода к следующему слову необходимо прибавить 2 к текущему значению адреса.

Цикл реализован с помощью конструкции «loop X», которая «переадресовывает» выполнение программы на строку с меткой «X:». Цикличная переадресация будет происходить, пока в регистре «CX» не будет содержаться нулевого значения. За каждый пройденный цикл значение в «CX» уменьшается на 1, а первоначальным значением является константа «LEN».

Для того, чтобы в результате деления (командами «div» или «idiv») на 2-байтовое слово ответом было 2-байтовое слово, нужно представить делимое как 4-байтовое слово и поместить его в регистры «DX» и «AX» (в «DX» старшие разряды, а в «AX» младшие). Для выполнения данной операции с минимальным шансом допустить ошибку, рекомендуется использовать команду «CWD».

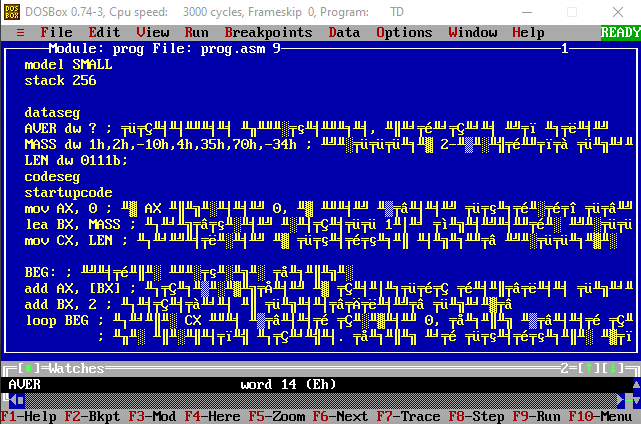
Деление с сохранением знака происходит с помощью команды «idiv», аргументом которой является только делитель, который содержится в регистре или константе. Результаты деления вносятся в регистры «AX» и «DX» (в «AX» – целочисленное частное, в «DX» – остаток деления).

После совершения операции деления значение из регистра «AX» вносится в переменную «AVER», после чего программа завершает работу с кодом «0».

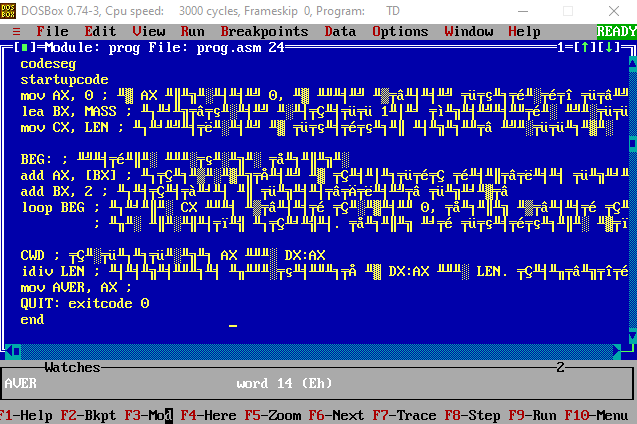
Полный листинг программы с комментариями расположен в *«Приложение А».*

**3.4 Описание полученных результатов**

Программный модуль был скомпилирован, запущен и отлажен в рамках среды «DOS BOX» с разными входными данными. При вызове «TASM» были заданы ключи «-L -ZI», которые позволили получить отладочные файлы. При вызове «TLINK» были использованы ключи «-V». С целью отладки был запущен отладчик «TD», в котором было проведено пошаговое исполнение программ. На *рисунках 1 и 2* можно наблюдать результат выполнения программы в отладчике – значение переменной, которую необходимо вычислить расположено в секции «Watches». Результат работы программы совпал с ожидаемым – программа отработала корректно.

**

*Рисунок 1 – результат работы программы в «Turbo Debugger» (часть 1)*

**

*Рисунок 2 – результат работы программы в «Turbo Debugger» (часть 1)*

# **4 ВЫВОДЫ**

**4.1 Полученные знания, навыки, умения**

В ходе выполнения лабораторной работы была разработан и отлажен программный модуль, который считает среднее арифметическое значение заданного массива из 2-байтовых слов. Получены навыки работы с метками, флагом «CX», мнемониками «div», «idiv», «cwd» и «loop», для создания циклов в программах на языке «Ассемблер».

**ПРИЛОЖЕНИЕ А**

model SMALL

stack 256

dataseg

AVER dw ? ; среднее значение, которое мы ищем

MASS dw 1h,2h,-10h,4h,35h,70h,-34h ; масссив 2-байтных слов

LEN dw 0111b;

codeseg

startupcode

mov AX, 0 ; в AX кладем 0, в нем будем считать сумму эл-ов

lea BX, MASS ; получаем адресс 1го элемента массива и кладем его в BX

mov CX, LEN ; помещаем в счетчик длину массива

BEG: ; метка начала цикла

add AX, [BX] ; прибавляем в регистр текущее слово из массива

add BX, 2 ; переход к следующему слову

loop BEG ; пока CX не будет равен 0, цикл будет работать

; за каждый пройд. цикл от счетчика вычитается 1

CWD ; расписали AX на DX:AX

idiv LEN ; деление значения в DX:AX на LEN. результат в AX остаток в DX

mov AVER, AX ;

QUIT: exitcode 0

end