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 № 14*

Выполнил:

Студент группы

КТбо2-8

|  |  |  |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Нестеренко П. А. |
|  | *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, использующих циклические алгоритмы работы и условные переходы, 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 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 сумму всех отрицательных чисел 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. Массива «arr», в котором расположено 10 2-байтных слов
2. Неизвестной переменной «sum», которую необходимо найти с помощью алгоритма программы.

Общая логика работы алгоритма сводится к циклическому суммированию всех отрицательных элементов массива в регистре «DX».

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 BEG», которая «переадресовывает» выполнение программы на строку с меткой «BEG:». Цикличная переадресация будет происходить, пока в регистре «СX» не будет содержаться нулевого значения. За каждый пройденный цикл значение в «CX» уменьшается на 1, а первоначальное значение «10».

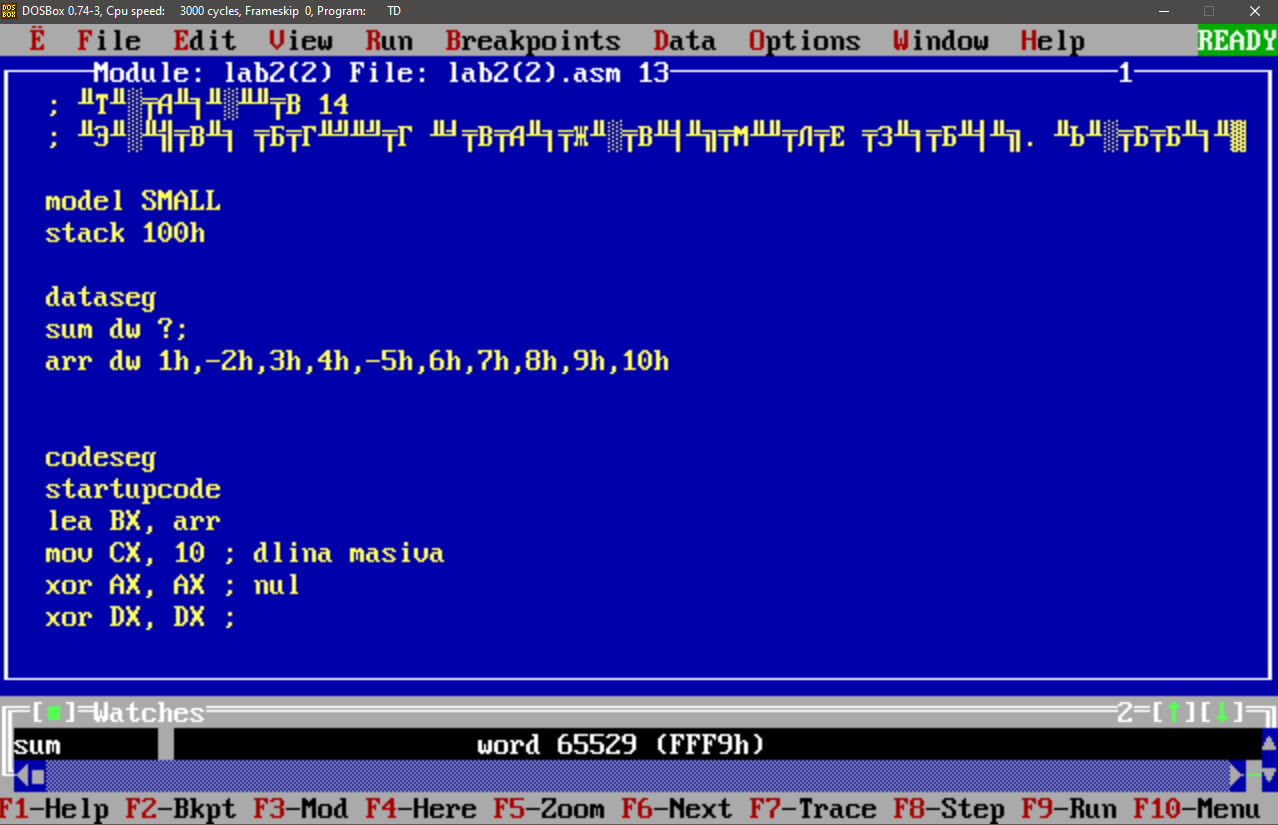
В цикле с помощью мнемоника «cmp» мы сравниваем число с 0. «Сmp» возвращает флаг, который мы анализируем с помощью мнемокода «jns». Если число отрицательное, то накапливаем сумму в регистре DX, иначе проверяем массив дальше.

После совершения операций значение из регистра «DX» вносится в переменную «sum», после чего программа завершает работу с кодом «0». Значение «sum» в дополнительном коде.

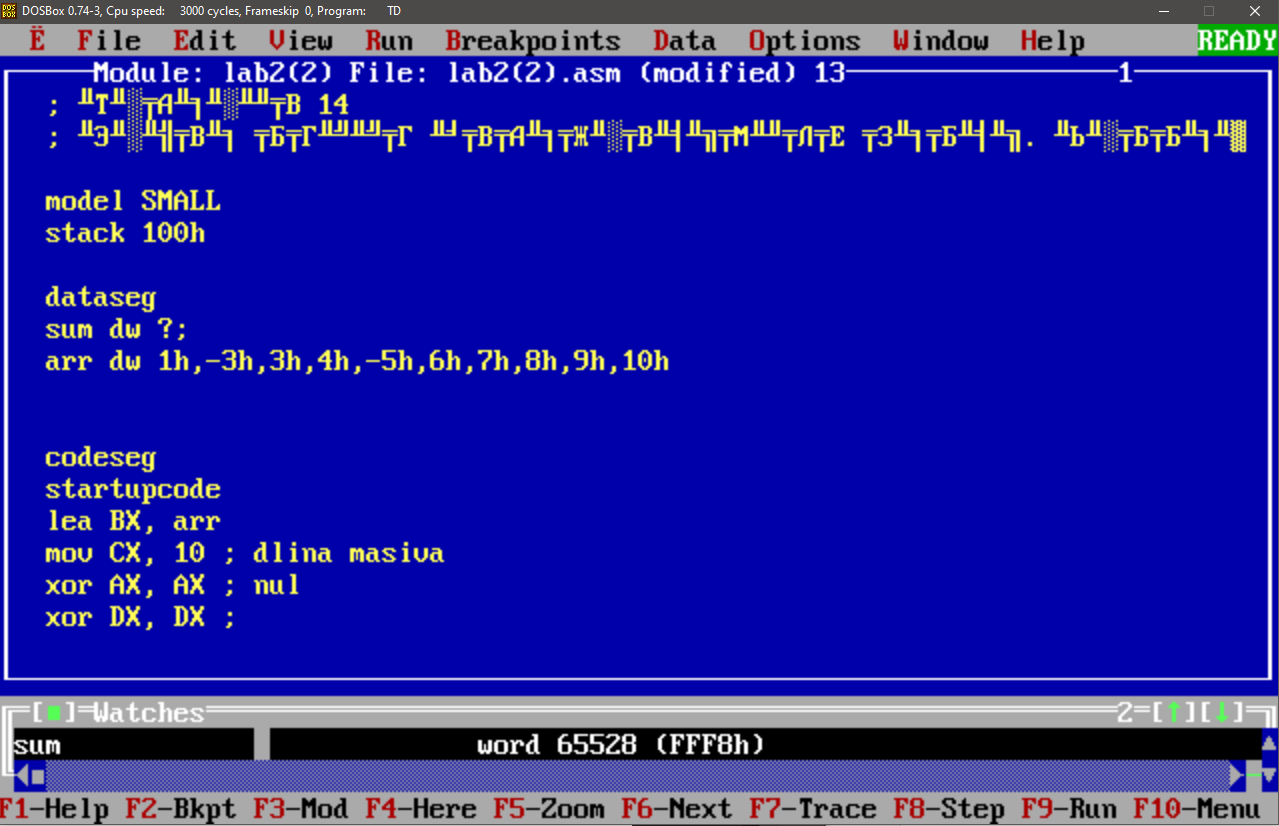
Полный листинг программы с комментариями расположен в *«Приложение А».*

**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 2)*

# **4 ВЫВОДЫ**

**4.1 Полученные знания, навыки, умения**

В ходе выполнения лабораторной работы была разработан и отлажен программный модуль, который находит сумму отрицательных элементов массива из 2-байтовых слов. Получены навыки работы с метками, флагом «CX», мнемониками «div», «cmp», «jns» и «loop», для создания циклов в программах на языке «Ассемблер».

**ПРИЛОЖЕНИЕ А**

model SMALL

stack 100h

dataseg

sum dw ?;

arr dw 1h,-2h,3h,4h,-5h,6h,7h,8h,9h,10h

codeseg

startupcode

lea BX, arr

mov CX, 10 ; dlina masiva

xor AX, AX ; nul

xor DX, DX ;

BEG: cmp [BX],AX

jns NO

add DX,[BX] ;esli -

NO: add BX,2 ; esli +

loop BEG

mov sum,DX ;v dop code

QUIT: exitcode 0

end

;65529 = -7

;65528 = -8