**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Архитектура компьютера»**

**Тема: Освоение трансляции, выполнения и отладки программ на языке Ассемблера процессора Intel X86**

|  |  |  |
| --- | --- | --- |
| Студент гр. 5383 |  | Допира В. Е. |
| Преподаватель |  | Кирьянчиков В. А. |

Санкт-Петербург

2016

**Цель работы :**

а) отредактировать строки приветствия, указав собственные данные;

б) выполнить трансляцию с выводом файла листинга и получением объектного файла;

если объектный файл не сформирован, найти по листингу и устранить ошибки;

в) выполнить компоновку с сохранением карты памяти и получением exe-файлов;

г) запустить программы в автоматическом режиме и просмотреть результаты

на экране пользователя, открываемом по Ctrl^O ;

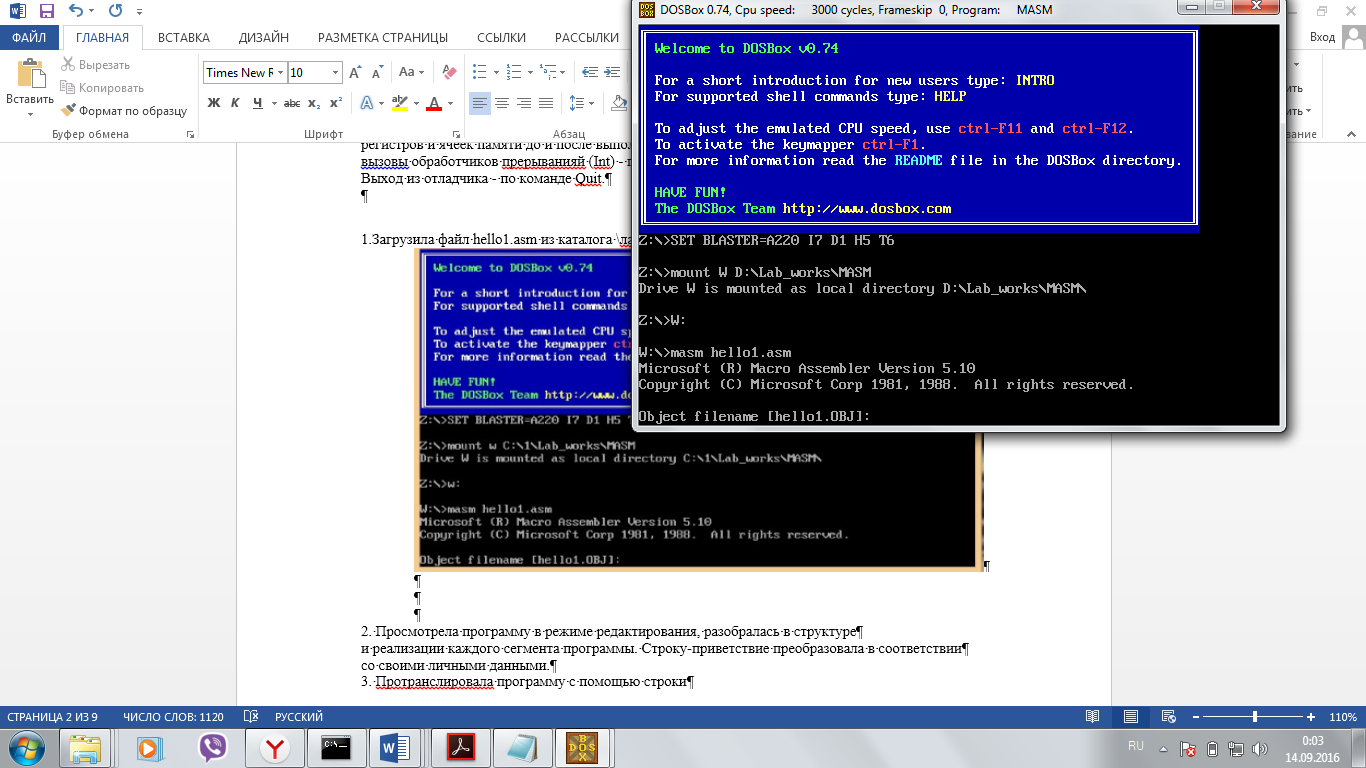
д) запустить выполнение в пошаговом режиме под управленим отладчика afd с фиксацией используемых регистров и ячеек памяти до и после выполнения каждой команды. Обычные команды выполняются по F1, а вызовы обработчиков прерыванияй (Int) - по F2, чтобы не входить внутрь обработчика прерываний.

Выход из отладчика - по команде Quit.

**Ход работы :**

1.Загружен файл «hello1.asm» из каталога \лаборат\_работы в каталог \MASM.

2. Программа просмотрена в режиме редактирования c помощью файлового менеджера Far, разобрана в структуре и реализации каждого сегмента программы. Строка-приветствие преобразована в соответствии со своими личными данными.

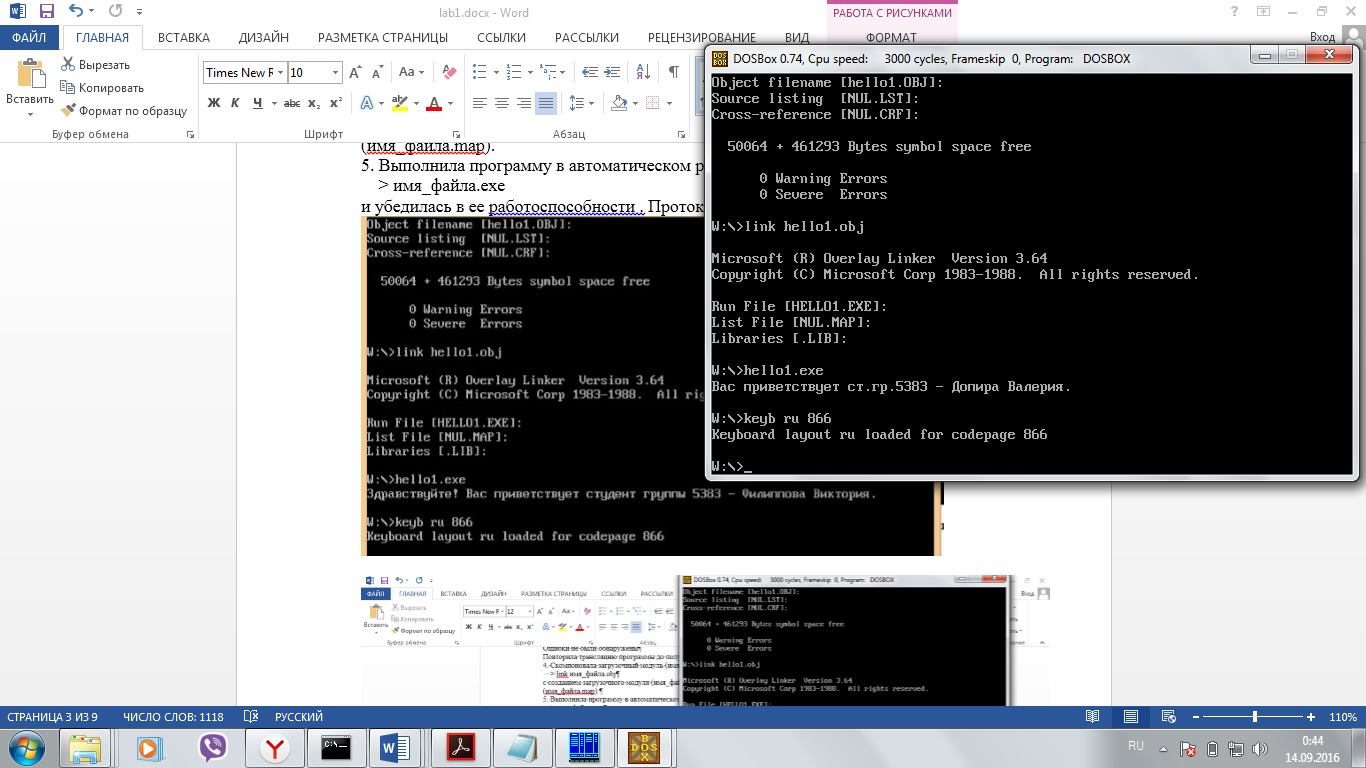


3. Запущен DOSBOX. Протранслирована программа с помощью строки «> masm hello1.asm», создан объектный файл «hello1.obj» и файл диагностических сообщений (файла листинга) «hello1.lst».

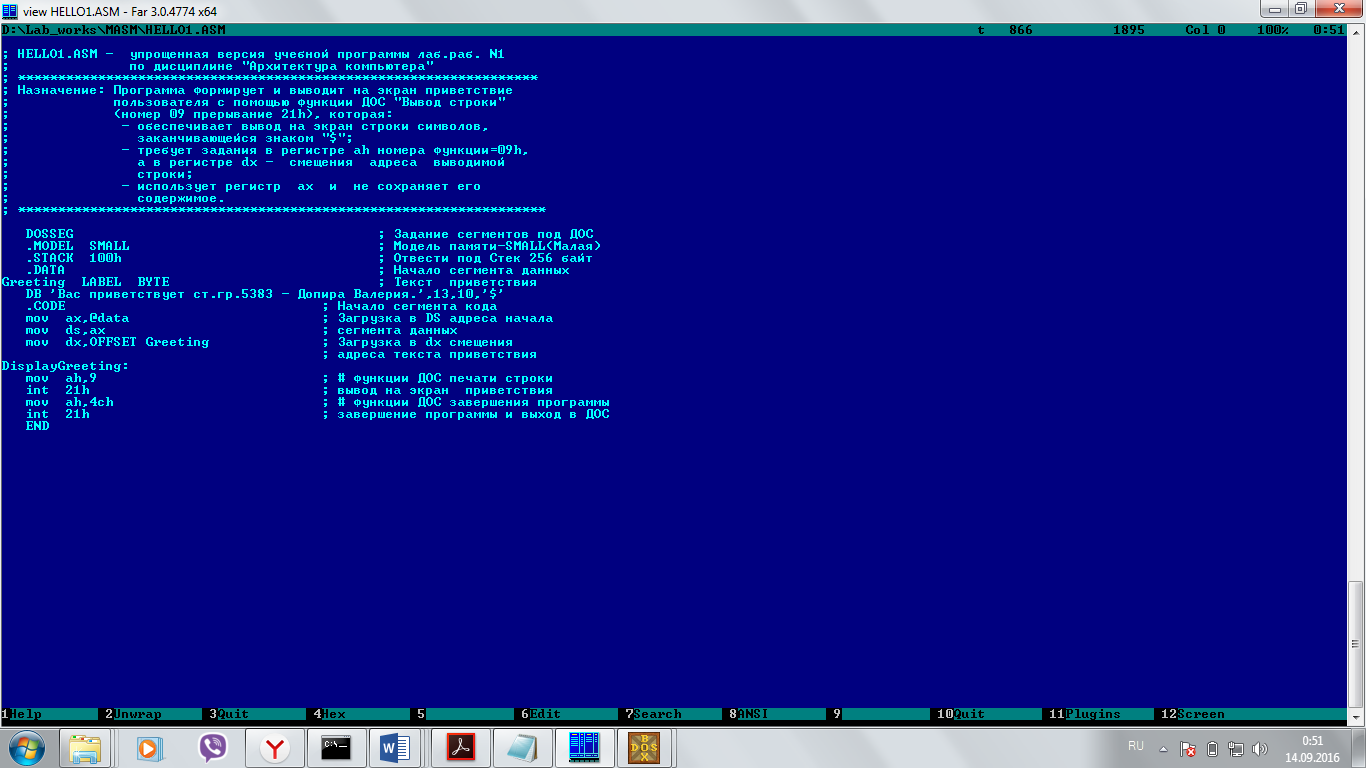
Ошибки не были обнаружены.

4. Скомпонован загрузочный модуль с помощью строки «> link hello1.obj», создан загрузочный модуль (hello1.exe) и файл карты памяти (hello1.map).

5. Выполнена программа в автоматическом режиме путем набора строки «>hello1.exe» и работоспособна. Протокол наблюдений представлен ниже.

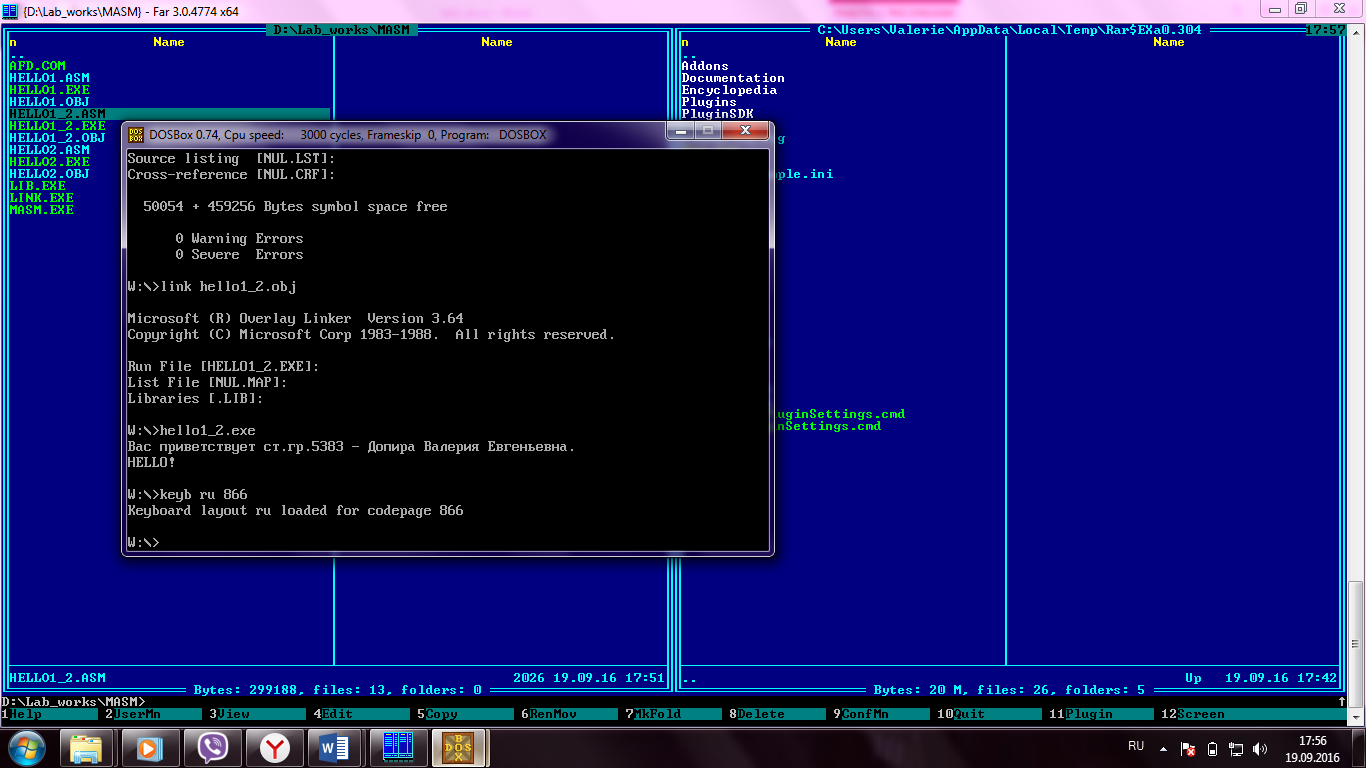


Исходный код: HELLO1.asm

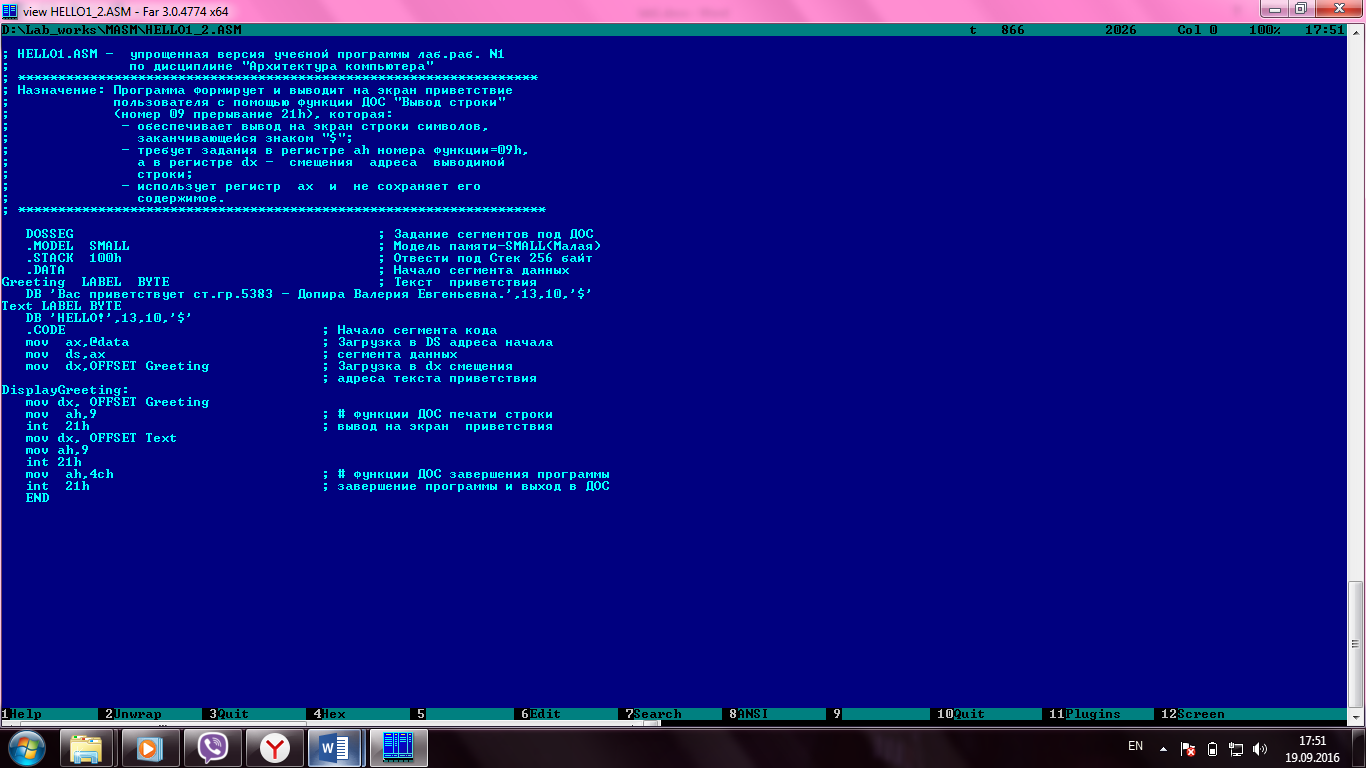


6. Создана версия hello1\_2 программы, дополненную еще одним оператором вывода другой строки приветствия.

7. Протранслирована программа hello1\_2, скомпонован загрузочный модуль, выполнен в автоматическом режиме и работает правильно.

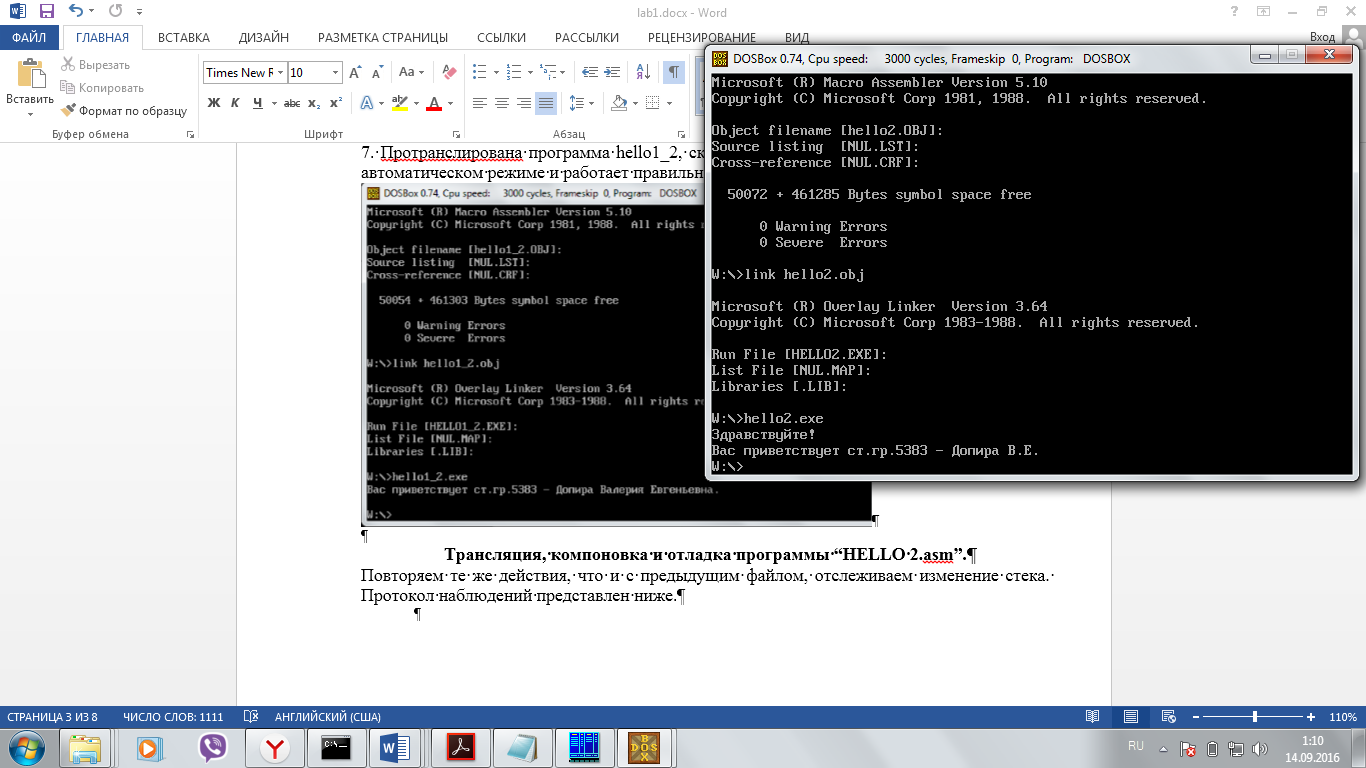


Исходный код: HELLO1\_2.asm

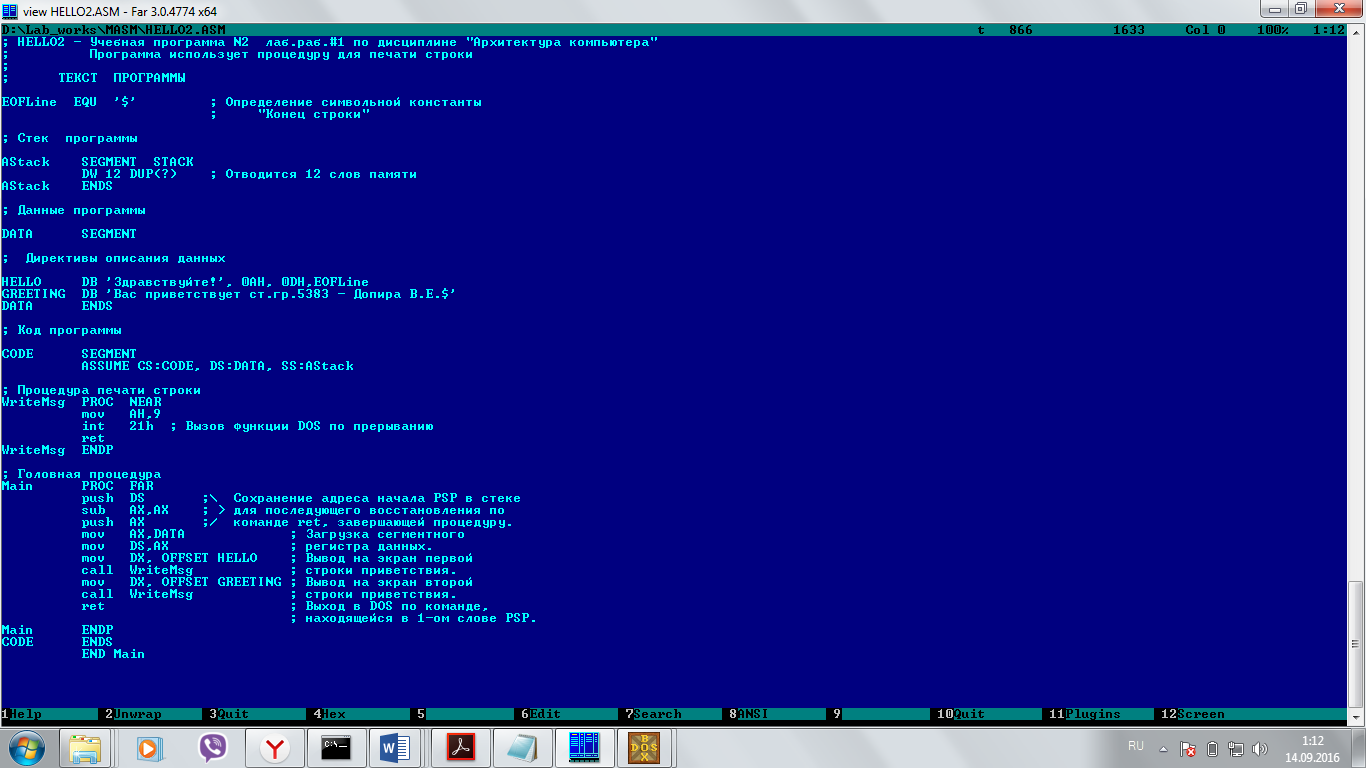


**Трансляция, компоновка и отладка программы “HELLO 2.asm”.**

Повторяем те же действия, что и с предыдущим файлом, отслеживаем изменение стека. Протокол наблюдений представлен ниже.



Исходный код: HELLO2.asm



**Регистры и сегменты**

AX – Accumulator, регистр данных, 16 битов

DX – Data Register, регистр данных, 16 битов

SP – Stack Pointer , регистр стека, отображает состояние (положение) стека, 16 битов

IP – Instruction Pointer, регистр , указывающий на смещение (адрес) инструкций в сегмента кода.

DS – Data Segment, сегментый регистр, указывающий на сегмент данных

CS – Code Segment, сегментый регистр, указывающий на сегмент кода.

**Команды, встретившиеся в лабораторной работе.**

**MOV**- приемник, источник – базовая команда пересылки данных. Копирует содержимое источника в приемник, источник не изменяется. Действует аналогично оператору присваивания на языках высокого уровня, но позволяет работать не только с переменными в памяти, но и со всеми регистрами процессора. Назначение пересылка данных.

**INT** – помещает в стек содержимое регистров EFLAGS , СS и EIP , после чего передает управление программе, называемой обработчиком прерываний с указанным в качестве операнда номером( число от 0 до OFFh). В DOS вызовы прерываний используются для выполнения большинства системных функций – работы с файлами, вводом/выводом и т.д ( Например, завершение выполнения программы и возвращения управления DOS, как и в лабораторной работе ( INT 21h)). Назначение – вызовы прерывания.

**PUSH**- помещает содержимое источника ( операнд) в стек. Источником может быть регистр, сегментный регистр, непосредственный операнд или переменная . Назначение : поместить данные в стек .

**POP** – помещает в приемник (операнд) слово или двойное слово, находящееся в вершине стека, увеличивая ESP(SP) на 2 или 4 соответственно POP выполняет действие , полностью обратное PUSH, и практически всегда используется в паре с ним. Приемником может быть регистр общего назначения, сегментный регистр, кроме CS( чтобы загрузить CS из стека, надо воспользоваться командой RET), или переменная. Назначение считать данные из стека .

**SUB** -приемник, источник – вычитает источник из приемника и помещает разность в приемник. Приемник может быть регистром или переменной, источник – числом, регистром или переменной, но нельзя использовать переменную одновременно и для источника, и для приемника. Является арифметической командой (работает с числами со знаком и без знака). Назначение: вычитание.

**CALL** – сохраняет адрес (IP) в стеке и передает управление по адресу, указанному в операнде. В HELLO2 сохранял адрес следующей команды в стек. Назначение: вызов процедуры

**RET** – считывает из стека слово и загружает его в IP (или EIP ), выполняя тем самым действия, обратные ближнему вызову процедуры командой CALL или загружает из стека в IP(EIP) и CS, возвращаясь из дальней процедуры. Назначение: возврат из процедуры.

**Протокол программы**

HELLO1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Адрес команды** | **Символьный код команды** | **16-ричный код команды** | **Содержимое регистров и ячеек памяти** | |
| **До выполнения** | **После выполнения** |
| 0010 | MOV AX, 11 AE | B8A311 | (AX)=0000  (IP)=0010 | (AX)=11AE  (IP)=0010 |
| 0013 | MOV DS, AX | 8ED8 | (DS)=119C  (IP)=0013 | (DS)=11AE  (IP)=0013 |
| 0015 | MOV DX, 0000 | BA0000 | (DX)=0000  (IP)=0015 | (DX)=0000  (IP)=0015 |
| 0018 | MOV AH,09 | B409 | (AX)=11AE  (IP)=0018 | (AX)=09AE  (IP)=0018 |
| 001А | INT 21 | CD21 | (IP)=001A | (IP)=001A |
| 001С | MOV AH, 4C | B44C | (AX)=09AE  (IP)=001C | (AX)=4CAE  (IP)=001C |
| 001Е | INT 21 | CD21 | - | - |

HELLO 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Адрес команды** | **Символьный код команды** | **16-ричный код команды** | **Содержимое регистров и ячеек памяти** | |
| **До выполнения** | **После выполнения** |
| 0005 | PUSH DS | 1E | (SP)=018  (IP)=0005 | (SP)=016 |
| 0006 | SUB AX, AX | 2BC0 | - | - |
| 0008 | PUSH AX | 50 | (SP)=016  (IP)=0008  Стек +0 119С  +2 0000  +4 0000  +6 0000 | (SP)=014  Стек +0 0000  +2 119С  +4 0000  +6 0000 |
| 0009 | MOVAX, 11AE | B8AE11 | (AX)=0000  (IP)=0009 | (AX)=11AE |
| 000C | MOV DS, AX | 8ED8 | (DS)=119C | (DS)=11AE- |
| 000E | MOV DX, 0000 | BA0000 |  |  |
| 0011 | CALL 0000 | E8ECFF | (SP)=014 | (SP)=012- |
| 0000 | MOV AH, 09 | B409 | (AX)=11AE | (AX)=09AE |
| 0002 | INT 21 | CD21 |  |  |
| 0004 | RET | C3 | (SP)=0014  Стек +0 0014  +2 0000  +4 119С  +6 0000 | (SP)=000С  Стек +0 0004  +2 11В2  +4 0244  +6 0014 |
| 0014 | MOV DX,0010 | BA1000 | (DX)=0000 | (DX)=0010 |
| 0017 | CALL 0000 | E8E6FF | (SP)=0014  Стек +0 0000  +2 119С  +4 0000  +6 0000 | (SP)=0012  Стек +0 001А  +2 0000  +4 119С  +6 0000 |
| 0000 | MOV AH, 09 | B409 |  |  |
| 0002 | INT 21 | CD21 |  |  |
| 0004 | RET | C3 | (SP)=0012  Стек +0 001А  +2 0000  +4 119С  +6 0000 | (SP)=0014  Стек +0 0004  +2 11В2  +4 0244  +6 001А |
| 001A | RET Far | CB | (SP)=0014  (CS)=11B3  Стек +0 0000  +2 119С  +4 0000  +6 0000 | (SP)=0018  (CS)=119C  Стек +0 0000  +2 0000  +4 0000  +6 0000 |
| 0000 | INT 20 | CD20 | (AX)=09AE  (CX)=008B  (CS)=119C  (DS)=11AE | (AX)=0000  (CX)=0000  (CS)=11B3  (DS)=119C |

**Изменение стека в программе HELLO2**

|  |  |
| --- | --- |
| **Адрес команды** | **Стек** |
| Начальное положение | +0 0000  +2 0000  +4 0000  +6 0000 |
| 0005 | +0 119С  +2 0000  +4 0000  +6 0000 |
| 0008 | +0 0000  +2 119С  +4 0000  +6 0000 |
| 0011 | +0 0014  +2 0000  +4 119С  +6 0000 |
| 0004 | +0 0000  +2 119С  +4 0000  +6 0000 |
| 0017 | +0 001А  +2 0000  +4 119С  +6 0000 |
| 0004 | +0 0000  +2 119С  +4 0000  +6 0000 |
| 001А | +0 0000  +2 0000  +4 0000  +6 0000 |
| 0000 | +0 0000  +2 0000  +4 0000  +6 0000 |

**Вывод**: В ходе выполнения данной лабораторной работы применены практические навыки работы с методами создания и отладки программ, написанных на языке ассемблера. Результат выполнения работы содержится в программах hello1.exe, hello1\_2.exe и hello2.exe. При отладке ошибок не обнаружено.