МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Организация ЭВМ и Систем»

Тема: Трансляции, отладка и выполнение программ на Ассемблере.

Студентка гр. 0383 Рудкова Ю.В.

Преподаватель:

Ефремов М.А.

Санкт-Петербург, 2021

Цель работы.

Изучить механизм работы трансляции, отладки и выполнения программ на Ассемблере.

Задание.

Лабораторная работа 1 использует 2 готовых программы на ассемблере: hello1 – составлена с использованием сокращенного описания сегментов и hello2 – составлена с полным описанием сегментов и выводом строки, оформленным как процедура. Выполнение работы состоит из двух частей, по каждой из которых необходимо представить протокол с фиксацией всех выполняемых действий и полученных результатов, и подписать его у преподавателя.

Уточнение задания следует посмотреть в файле lr1_comp.txt каталога Задания.

Часть 1

1. Просмотреть программу hello1.asm, которая формирует и выводит на экран приветствие пользователя с помощью функции ОС MSDOS, вызываемой через прерывание с номером 21H (команда Int 21h).

Выполняемые функцией действия и задаваемые ей параметры - следующие:

- обеспечивается вывод на экран строки символов, заканчивающейся знаком "$";
- требуется задание в регистре ah номера функции, равного 09h, а в регистре dx - смещения адреса выводимой строки;
- используется регистр ax и не сохраняется его содержимое.

2. Разобраться в структуре и реализации каждого сегмента программы. Непонятные фрагменты прояснить у преподавателя. Строку-приветствие преобразовать в соответствии со своими личными данными.

3. Загрузить файл hello1.asm из каталога Задания в каталог Masm.

4. Протранслировать программу с помощью строки

> masm hello1.asm

с созданием объектного файла и файла диагностических сообщений (файла листинга). Объяснить и исправить синтаксические ошибки, если они будут обнаружены транслятором. Повторить трансляцию программы до получения объектного модуля.

5. Скомпоновать загрузочный модуль с помощью строки

> link hello1.obj

с созданием карты памяти и исполняемого файла hello1.exe.

6. Выполнить программу в автоматическом режиме путем набора строки

> hello1.exe

убедиться в корректности ее работы и зафиксировать результат выполнения в протоколе.

7. Запустить выполнение программы под управлением отладчика с помощью команды

> afd hello1.exe

4

Записать начальное содержимое сегментных регистров CS, DS, ES и SS. Выполнить программу в пошаговом режиме с фиксацией используемых регистров и ячеек памяти до и после выполнения каждой команды. Обычные команды выполняются по F1 (Step), а вызовы обработчиков прерываний (Int) - по F2 (StepProc), чтобы не входить внутрь обработчика прерываний. Продвижение по сегментам экранной формы отладчика выполняется с помощью клавиш F7 – F10 (up, down, left, right). Перезапуск программы в отладчике выполняется клавишей F3 (Retrieve). Выход из отладчика - по команде Quit.

Результаты прогона программы под управлением отладчика должны быть представлены в виде, показанном на примере одной команды в табл.1, и подписаны преподавателем.

Табл.1

Адрес Команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			до выполнения .	После выполнения
0003	Mov DS, AX	8E D8	(AX) = 2D87 (DS) = 2D75 (IP) = 0003	(AX) = 2D87 (DS) = 2D87 (IP) = 0005

Часть 2

Выполнить пункты 1 - 7 части 1 настоящего задания применительно к программе hello2.asm, приведенной в каталоге Задания, которая выводит на экран приветствие пользователя с помощью процедуры WriteMsg, а также использует полное определение сегментов. Сравнить результаты прогона под управлением отладчика программ hello1 и hello2 и объяснить различия в размещении сегментов.

Отчет по работе должен содержать:

1) текст задания;

2) тексты исходных файлов программ hello1 и hello2;

3) тексты файлов диагностических сообщений hello1.lst и hello2.lst;

4) протокол работы на компьютере, включающий основные действия по пунктам 1 - 6 и протоколы пошагового исполнения каждой из программ под управлением отладчика в виде таблицы 1 (черновики протоколов должны быть подписаны преподавателем).

5) выводы по работе.

Выполнение работы.

Часть1. Работа с файлом hello1.asm.

Выполнила транслирование программы, скомпоновала, запустила программу в автоматическом режиме. Программа работает корректно.

Произвела запуск программы под управлением отладчика, фиксируя изменения используемых регистров.

Начальные значения системных регистров:

(CS)=1A05, (DS)=19F5 , (ES)=19F5, (SS)=1A0A.

Табл. 2:

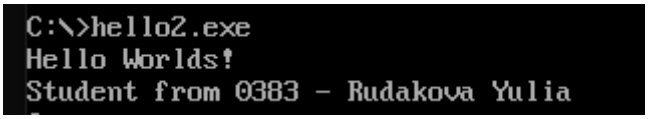
Адрес команды	Символический код команды	16-ричный код команды	Содержание регистров и ячеек памяти	
			до выполнения	после выполнения
0010	MOV, AX, 1A07	B8071A	(AX) = 0000 (DX)= 0000 (DS)= 19F5 (IP) = 0010	(AX)= 1A07 (DX)= 0000 (DS)= 19F5 (IP) = 0013
0013	MOV, DS, AX	8ED8	(AX) =1A07 (DX)= 0000 (DS)= 19F5 (IP) = 0013	(AX) =1A07 (DX)= 0000 (DS)= 1A07 (IP) = 0015
0015	MOV, DX, 0000	BA0000	(AX) =1A07 (DX)= 0000 (DS)= 1A07 (IP) = 0015	(AX) =1A07 (DX)= 0000 (DS)= 1A07 (IP) = 0018
0018	MOV, AH, 09	B409	(AX) =1A07 (DX)= 0000 (DS)= 1A07 (IP) = 0018	(AX) =0907 (DX)= 0000 (DS)= 1A07 (IP) = 001A

001A	INT, 21	CD21	(AX) =0907 (DX)= 0000 (DS)= 1A07 (IP) = 001A	(AX) =0907 (DX)= 0000 (DS)= 1A07 (IP) = 14A0
001C	MOV, AH,4C	B44C	(AX) =0907 (DX)= 0000 (DS)= 1A07 (IP) = 14A0	(AX) =4C07 (DX)= 0000 (DS)= 1A07 (IP) = 001E
001E	INT, 21	CD21	(AX) =4C07 (DX)= 0000 (DS)= 1A07 (IP) = 001E	(AX) =0000 (DX)= 0000 (DS)= 19F5 (IP) = 0010

Разработанный программный код см. в приложении А.

Часть 2. Работа с файлом hello2.asm

Выполнила транслирование программы, скомпоновала, запустила программу в автоматическом режиме. Программа работает корректно.



Произвела запуск программы под управлением отладчика, фиксируя изменения используемых регистров.

Начальные значения системных регистров:

(CS)=1A0B, (DS)=19F5 , (ES)=19F5, (SS)=1A05.

Адрес команды	Символический код команды	16-ричный код команды	Содержание регистров и ячеек памяти	
			до выполнения	после выполнения
0005	PUSH DS	1E	(AX) = 0000	(AX) = 0000

			(DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0018 Stack +0 0000 (IP) = 0005	(DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0016 Stack +0 19F5 (IP) = 0006
0006	SUB AX, AX	2BC0	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0016 Stack +0 19F5 (IP) = 0006	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0016 Stack +0 19F5 (IP) = 0008
0008	PUSH AX	50	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0016 Stack +0 19F5 (IP) = 0008	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0009
0009	Mov AX, 1A07	B8071A	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0009	(AX) = 1A07 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 000C
000C	Mov DS, AX	8EDD8	(AX) = 1A07 (DX) = 0000 (DS) = 19F5 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 000C	(AX) = 1A07 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 000E
000E	Mov DX,	BA 0000	(AX) = 1A07	(AX) = 1A07

	0000		(DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 000E	(DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0011
0011	CALL 0000	E8ECFF	(AX) = 1A07 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0011	(AX) = 1A07 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0000
0000	MOV AH, 09	B409	(AX) = 1A07 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0000	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0002
0002	Int 21	CD21	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0002	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0004
0004	RET	C3	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0012	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014

			Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0004	Stack +0 0000 Stack +2 19F5 (IP) = 0014
0014	Mov DX, 0010	BA1000	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0014	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0017
0017	CALL 0000	E8E6FF	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 0017	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0000
0000	Mov AH, 09	B409	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0000	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0002
0002	INT 21	CD21	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0002	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0004

0004	RET	C3	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0012 Stack +0 001A Stack +2 0000 Stack +4 19F5 (IP) = 0004	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 0000 (IP) = 001A
001A	RET FAR	CB	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =1A0A (SP) = 0014 Stack +0 0000 Stack +2 0000 (IP) = 001A	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =19F5 (SP) = 0018 Stack +0 0000 (IP) = 0000
0000	Int 20	CD20	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) =19F5 (SP) = 0018 Stack +0 0000 (IP) = 0000	

Разработанный программный код см. в приложении А.

Результаты прогона программы под управлением отладчика hello1.asm:

1. ASSUME определяет через какой регистр сегмента происходит доступ к информации
2. Требуется обязательное задание модели памяти, в которой используется эта программа..MODEL - модель памяти. Она накладывает ограничения на комбинирования сегментов.
3. Greeting LABEL BYTE - определение метки типа byte

4. CS: в регистр AX помещается смещение сегмента, в котором хранятся данные
5. В регистр DX помещается значение смещение начала сообщения.

   Результаты прогона программы под управлением отладчика hello2.asm:

   1. для доступа к информации используется сегмент кода (CS) и сегмент стека (SS)
   2. (DS) директива описания данных - HELLO и GREETING
   3. (CS) описание печати строк
   4. Загрузка сегментного регистра данных, как в hello1.asm
   5. Вызов сроки HELLO и GREETING
   6. Завершение программы

Выводы.

В ходе лабораторной работы я изучила механизм работы трансляции, отладки и выполнила программу на Ассемблере в иммитаторе DOSBOX.

Приложение А

Исходный код программы

```
; HELLO1.ASM -  упрощенная версия учебной программы лаб.раб. N1
;               по дисциплине "Архитектура компьютера"
; ***************************************************************
; Назначение: Программа формирует и выводит на экран приветствие
;       пользователя с помощью функции ДОС "Вывод строки"
;       (номер 09 прерывание 21h), которая:
;               - обеспечивает вывод на экран строки символов,
;               заканчивающейся знаком "$";
;               - требует задания в регистре ah номера функции=09h,
;               а в регистре dx -  смещения  адреса  выводимой
;               строки;
;               - использует регистр  ax  и  не сохраняет его
;               содержимое.
;
***************************************************************

  DOSSEG                        ; Задание сегментов под ДОС
  .MODEL  SMALL                 ; Модель памяти-SMALL(Малая)
  .STACK  100h                  ; Отвести под Стек 256 байт
  .DATA                         ; Начало сегмента данных
Greeting  LABEL  BYTE                   ; Текст  приветствия
  DB 'Вас приветствует ст.гр.0383 - Rudakova Yulia',13,10,'$'
  .CODE                         ; Начало сегмента кода
  mov  ax, @data                ; Загрузка в DS адреса начала
  mov  ds, ax                   ; сегмента данных
  mov  dx, OFFSET Greeting              ; Загрузка в dx смещения
                                ; адреса текста приветствия
DisplayGreeting:
  mov  ah, 9                    ; # функции ДОС печати строки
  int  21h                      ; вывод на экран  приветствия
  mov  ah, 4ch                  ; # функции ДОС завершения программы
```

```
  int  21h                    ; завершение программы и выход в ДОС
  END


; HELLO2 - Учебная программа N2  лаб.раб.#1 по дисциплине "Архитектура компьютера"
;       Программа использует процедуру для печати строки
;
;       ТЕКСТ  ПРОГРАММЫ

EOFLine  EQU  '$'         ; Определение символьной константы
                    ;       "Конец строки"

; Стек  программы

ASSUME CS:CODE, SS:AStack

AStack      SEGMENT  STACK
        DW 12 DUP(?)     ; Отводится 12 слов памяти
AStack      ENDS

; Данные программы

DATA        SEGMENT

;  Директивы описания данных

HELLO       DB 'Hello Worlds!', 0AH, 0DH,EOFLine
GREETING  DB 'Student from 0383 - Rudakova Yulia$'
DATA        ENDS

; Код программы

CODE        SEGMENT
; Процедура печати строки
WriteMsg  PROC  NEAR
```

```
        mov   AH,9
        int   21h  ; Вызов функции DOS по прерыванию
        ret
WriteMsg  ENDP

; Головная процедура
Main          PROC  FAR
        push  DS     ;\  Сохранение адреса начала PSP в стеке
        sub   AX,AX          ; > для последующего восстановления по
        push  AX     ;/  команде ret, завершающей процедуру.
        mov   AX,DATA              ; Загрузка сегментного
        mov   DS,AX                ; регистра данных.
        mov   DX, OFFSET HELLO     ; Вывод на экран первой
        call  WriteMsg             ; строки приветствия.
        mov   DX, OFFSET GREETING ; Вывод на экран второй
        call  WriteMsg             ; строки приветствия.
        ret                  ; Выход в DOS по команде,
                             ; находящейся в 1-ом слове PSP.
Main          ENDP
CODE          ENDS
        END Main
```