

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»**

Студент гр. 0383

Желнин М.Ю.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучить механизм работы трансляции, отладки и выполнении программ на языке Ассемблер.

Формулировка задания.

Необходимо скомпилировать и выполнить две программы на ассемблере: hello1.asm и hello2.asm, заменив данные выводимой строки на свои. Для выполнения программ нужно скачать DosBox, компилятор nasm и отладчик afdpro.

Далее следует разобраться в структуре обеих программ, и после запуска в отладчике записать в таблицу изменения содержимого регистров.

Ход выполнения.

1. Скомпилировав asm файл и слинковав obj файл, удалось запустить программу hello1.



Настроить вывод русского языка не удалось.

Произведен запуск программы под управлением отладчика afdpro с фиксацией используемых регистров до и после выполнения каждой команды в таблицу 1. Начальное содержимое системных регистров:

(CS) = 1A05; (DS) = 19F5; (ES) = 19F5; (SS) = 1A0C;

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			до выполнения	после выполнения
0010	mov ax,1A07	B8071A	(AX) = 0000 (DS) = 19F5 (IP) = 0010	(AX) = 1A07 (DS) = 19F5 (IP) = 0013
0013	mov ds, ax	8ED8	(AX) = 1A07 (DS) = 19F5	(AX) = 1A07 (DS) = 1A07

			(IP) = 0013	(IP) = 0015
0015	mov dx, 0000	BA0000	(AX) = 1A07 (DS) = 1A07 (IP) = 0015	(AX) = 1A07 (DS) = 1A07 (IP) = 0018
0018	mov ah, 09	B409	(AX) = 1A07 (DS) = 1A07 (IP) = 0018	(AX) = 0907 (DS) = 1A07 (IP) = 001A
001A	int 21	CD21	(AX) = 0907 (DS) = 1A07 (IP) = 001A	(AX) = 0907 (DS) = 1A07 (IP) = 001C
001C	mov ah, 4c	B44C	(AX) = 0907 (DS) = 1A07 (IP) = 001C	(AX) = 4C07 (DS) = 1A07 (IP) = 001E
001E	int 21	CD21	(AX) = 4C07 (DS) = 1A07 (IP) = 001E	(AX) = 0000 (DS) = 19F5 (IP) = 0010

Табл. 1

Детальнее рассмотрим некоторые исполняемые команды:

- 1) команда `mov xx, yy` помещает содержимое регистра `xx` в `yy`, например в строке `mov DS, AX` мы помещаем в `DS` содержимое `AX`, так как напрямую изменить содержимое `DS` нельзя. Также можно помещать смещение на начало сегмента при помощи команды `OFFSET`.
- 2) Прерывание `int 21h` содержит в себе множество функций, но чтобы выбрать конкретную, нам необходимо передать её номер в `ah`, что мы и делаем `mov ah, 09` или `mov ah, 4c`, где первая выводит сообщение на экран, а вторая завершает программу.
2. Аналогичным образом удалось запустить программу `hello2`.

```
C:\>hello2.exe
Hello Worlds!
Student from 0383 - Zhelnin Maksim
```

Далее запускаем программу в отладчике afdbg с фиксацией используемых регистров до и после выполнения каждой команды в таблицу 2. Начальное содержимое системных регистров:

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

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			до выполнения	после выполнения
0005	PUSH DS	1E	(AX) = 0000 (DX) = 0000 (DS) = 19F5 (CS) = 1A0A (SP) = 0018 Stack +0 0000 (IP) = 0005	(AX) = 0000 (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	(AX) = 1A07 (DX) = 0000 (DS) = 19F5 (CS) = 1A0A (SP) = 0014

			Stack +0 0000 Stack +2 19F5 (IP) = 0009	Stack +0 0000 Stack +2 19F5 (IP) = 000C
000C	Mov DS, AX	8ED8	(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, 0000	BA0000	(AX) = 1A07 (DX) = 0000 (DS) = 1A07 (CS) = 1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 (IP) = 000E	(AX) = 1A07 (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	(AX) = 0907 (DX) = 0000

			(DS) = 1A07 (CS) = 1A0A (SP) = 0012 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0002	(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 Stack +0 0014 Stack +2 0000 Stack +4 19F5 (IP) = 0004	(AX) = 0907 (DX) = 0000 (DS) = 1A07 (CS) = 1A0A (SP) = 0014 Stack +0 0000 Stack +2 19F5 Stack +4 0000 (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	(AX) = 0907 (DX) = 0010 (DS) = 1A07 (CS) = 1A0A (SP) = 0012 Stack +0 001A

			Stack +2 0000 Stack +4 19F5 (IP) = 0000	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 19F5 (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	-

Табл. 2

Детальнее рассмотрим некоторые исполняемые команды:

- 1) При помощи ASSUME мы сообщаем ассемблеру через какой регистр сегмента будет осуществляться доступ к информации.
- 2) Используя EOFLine EQU '\$', мы определяем какой символ будет свидетельствовать об окончании строки вывода.
- 3) (DS): Директивы описания данных - HELLO и GREETING
- 4) (CS): Описание процедуры печати строк

Вывод.

Был изучен механизм работы трансляции, отладки и выполнении программ на языке Ассемблер.

Приложение А.

hello1.asm

```
; 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 ДОО "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 ДОО
.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 - Желнин М.Ю.',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 ДОО печати строки
int 21h ; вывод на экран приветствия
mov ah, 4ch ; # функции ДОО завершения программы
int 21h ; завершение программы и выход в ДОО
END
```

hello2.asm

; 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 15 DUP('') ; Отводится 15 слов памяти

AStack ENDS

; Данные программы

DATA SEGMENT

; Директивы описания данных

HELLO DB 'Hello Worlds!', 0AH, 0DH, EOFLine

GREETING DB 'Student from 0383 - Zhelnin Maksim\$'

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


```

P·PSP°PεPsPj "$";
;          - C, CЪPμP±CfPμC, P·P°PrP°PSPëCЦ
PI CЪPμPiPëCfC, CЪPμ ah PSPsPjPμCЪP° C,, CfPSPεC†P
ëPë=09h,
;          P° PI CЪPμPiPëCfC, CЪPμ dx - C
ÍPjPμC%PμPSPëCЦ P°PrCЪPμCfP° PIC<PIPsPrPëPjPs
P№
;          CfC, CЪPsPePë;
;          - PëCfPíPsP»CЪP·CfPμC, CЪPμPiPëC
fC, CЪ ax Pë PSPμ CfPsC...CЪP°PSCЦPμC, PμPiPs
;          CfPsPrPμCЪP¶PëPjPsPμ.
; *****
; *****
*****

```

```

DOSSEG
; P—P°PrP°PSPëPμ CfPμPiPjPμPSC, PsPI PíPsPr P”Ph
PŸ
.MODEL SMALL
; PЪPsPrPμP»CЪ PíP°PjCЦC, Pë-SMALL(PЪP°P»P°CЦ)
.STACK 100h
; PhC, PIPμCfC, Pë PíPsPr PŸC, PμPe 256 P±P°P№C,
.DATA
; PќP°C‡P°P»Ps CfPμPiPjPμPSC, P° PrP°PSPSC<C...
Greeting LABEL BYTE
; PŸPμPeCfC, PíCЪPëPIPμC, CfC, PIPëCЦ
DB 'P°P°Cf PíCЪPëPIPμC, CfC, PICfPμC, CfC, .PiC
Ъ.0383 - P—PμP»PSPëPS PЪ.P®, 13, 10, '$'

```

0000

0000 D0 92 D0 B0 D1 81

20 D0 BF D1 80 D0

B8 D0 B2 D0 B5 D1

82 D1 81 D1 82 D0

B2 D1 83 D0 B5 D1

82 20 D1 81 D1 82

2E D0 B3 D1 80 2E

30 33 38 33 20 2D

Microsoft (R) Macro Assembler Version 5.10

9/15/21 11:25:28

Page 1-2

20 D0 96 D0 B5 D0

BB D0 BD D0 B8 D0

BD 20 D0 9C 2E D0
AE 2E 0D 0A 24

```

        .CODE                                ; PkP°C
        ‡P°P»Ps CÍPµPiPjPµPSC,P° PePsPrP°
0000 B8 ---- R      mov ax, @data            ; P—P°
                    PiCṪCfP·PeP° PI DS P°PrCṪPµCÍP° PSP°C‡P°P»P°
0003 8E D8          mov ds, ax                ; CÍPµ
                    PiPjPµPSC,P° PrP°PSPSC<C...
0005 BA 0000 R      mov dx, OFFSET Greeting    ; P—P°
                    PiCṪCfP·PeP° PI dx CÍPjPµC%oPµPSPëCİİ
                                ; P°PrC
                    ṪPµCÍP° C,PµPeCÍC,P° PiCṪPëPIPµC,CÍC,PİPëCİİ
0008               DisplayGreeting:
0008 B4 09          mov ah, 9                  ; # C,,
                    CfPSPeC†PëPë P”PhPŸ PiPµC‡P°C,Pë CÍC,CṪPsPePë
000A CD 21          int 21h                    ; PIC<P
                    IPsPr PSP° CÍPeCṪP°PS PiCṪPëPIPµC,CÍC,PİPëCİİ
000C B4 4C          mov ah, 4ch                ; # C,,
                    CfPSPeC†PëPë P”PhPŸ P·P°PIPµCṪC€PµPSPëCİİ PiCṪPs
                    PiCṪP°PjPjC<
000E CD 21          int 21h                    ; P·P°P
                    IPµCṪC€PµPSPëPµ PiCṪPsPiCṪP°PjPjC< Pë PIC<C...PsP
                    r PI P”PhPŸ
                    END
Microsoft (R) Macro Assembler Version 5.10      9/15/21 11:25:28
                    Symbols-1

```

Segments and Groups:

N a m e	Length	Align	Combine Class
DGROUP		GROUP	
_DATA	0047	WORD	PUBLIC'DATA'
STACK	0100	PARA	STACK 'STACK'
_TEXT	0010	WORD	PUBLIC'CODE'

Symbols:

N a m e	Type	Value	Attr
---------	------	-------	------

DISPLAYGREETING L NEAR0008 _TEXT

GREETING L BYTE 0000 _DATA

@CODE TEXT _TEXT

@CODESIZE TEXT 0

@CPU TEXT 0101h

@DATASIZE TEXT 0

@FILENAME TEXT hello1

@VERSION TEXT 510

33 Source Lines

33 Total Lines

19 Symbols

47994 + 459266 Bytes symbol space free

0 Warning Errors

0 Severe Errors

hello2.lst

Microsoft (R) Macro Assembler Version 5.10

9/15/21 11:26:45

Page 1-1

```
; HELLO2 - PJC:PμP±PSP°Cμ PιCτPσPιCτP°PjPjP° N2
P»P°P±.CτP°P±.#1 PιPs PrPëCΓC†PëPιP»PëPSPμ "P
hCτC...PëC,PμPeC,CrCτP° PePsPjPιCτCτC,PμCτP°"
;      PμCτPσPιCτP°PjPjP° PëCΓPιPsP»CHP·CrP
μC, PιCτPσC†PμPrCrCτCτ PrP»Cμ PιPμC†P°C,Pë CΓC,
CτPσPePë
;
;      PÿP•PъPÿPÿ PμP PhP“P PђPђPђP«
```

= 0024

```
EOFLine EQU '$' ; PhPιCτPμPrPμP»PμPSP
ëPμ CΓPëPjPIPsP»CHPSPσPM PëPsPSCΓC,P°PSC,C<
;      "PъPsPSPμC† CΓC
,CτPσPePë"
```

```

; PŸC,PμPε PïCḐPsPiCḐP°PjPjC<

ASSUME CS:CODE, SS:AStack

0000 AStack SEGMENT STACK
0000 000F[ DW 15 DUP('!') ; PhC,PIPsPrPëC,CÍC
        0021  15 CÍP»PsPI PïP°PjC¼C,Pë
        ]

001E AStack ENDS

; P°P°PSPSC<Pμ PïCḐPsPiCḐP°PjPjC<

0000 DATA SEGMENT

; P°PëCḐPμPεC,PëPIC< PsPïPëCÍP°PSPëC¼ PrP°PSPS
C<C...

0000 48 65 6C 6C 6F 20 HELLO DB 'Hello Worlds!', 0AH, 0DH,EOFLine
        57 6F 72 6C 64 73
        21 0A 0D 24
0010 53 74 75 64 65 6E GREETING DB 'Student from 0383 - Zhelnin Maksi
        m$'
        74 20 66 72 6F 6D
        20 30 33 38 33 20
        2D 20 5A 68 65 6C
        6E 69 6E 20 4D 61
        6B 73 69 6D 24
0033 DATA ENDS

; PḐPsPr PïCḐPsPiCḐP°PjPjC<

0000 CODE SEGMENT
; PμCḐPsC†PμPrCfCḐP° PïPμC‡P°C,Pë CÍC,CḐPsPePë
0000 WriteMsg PROC NEAR
0000 B4 09 mov AH,9
0002 CD 21 int 21h ; P°C<P·PsPI C,,CfPSPεC†PëP
        ë DOS PïPs PïCḐPμCḐC<PIP°PSPëCḐ
0004 C3 ret

```

```

0005                                WriteMsg ENDP

                                ; P“PsP»PsPIPSP°C¼ PiCThPsC†PµPrCfCThP°
0005                                Main  PROC FAR
0005 1E                                push DS      ;\ PŸPsC...CThP°PSPµPSPë
                                Pµ P°PrCThPµCÍP° PSP°C‡P°P»P° PSP PI CÍC,PµPePµ
0006 2B C0                            sub  AX,AX    ; > PrP»C¼ PiPsCÍP»PµP
                                rCfCThC%PµPiPs PIPsCÍCÍC,P°PSPsPIP»PµPSPëC¼ PiPs
0008 50                                push AX      ;/ PePsPjP°PSPPrPµ ret
                                , P·P°PIPµCThC€P°CThC%PµPNë PiCThPsC†PµPrCfCThCf.
0009 B8 ---- R                        mov  AX,DATA    ; P—P°PiCThC
                                íP·PeP° CÍPµPiPjPµPSC,PSPsPiPs
000C 8E D8                            mov  DS,AX      ; CThPµPiPëC
                                ÍC,CThP° PrP°PSPSC<C....
000E BA 0000 R                        mov  DX, OFFSET HELLO ; P’C<PIPsP
                                r PSP° CÍPeCThP°PS PiPµCThPIPsPNë
0011 E8 0000 R                        call WriteMsg    ; CÍC,CThPsP
                                ePë PiCThPëPIPµC,CÍC,PIPëC¼.
0014 BA 0010 R                        mov  DX, OFFSET GREETING ; P’C<PIPsP
                                r PSP° CÍPeCThP°PS PIC,PsCThPsPNë
0017 E8 0000 R                        call WriteMsg    ; CÍC,CThPsP
                                ePë PiCThPëPIPµC,CÍC,PIPëC¼.
001A CB                                ret            ; P’C<C...PsP
                                r PI DOS PiPs PePsPjP°PSPPrPµ,
                                ; PSP°C...PsP
                                rC¼IC%PµPNëCÍC¼ PI 1-PsPj CÍP»PsPIPµ PSP.
001B                                Main  ENDP
001B                                CODE  ENDS
                                END Main

```

Segments and Groups:

N a m e	Length	Align	Combine	Class
---------	--------	-------	---------	-------

ASTACK	001E	PARA	STACK
CODE	001B	PARA	NONE
DATA	0033	PARA	NONE

Symbols:

N a m e	Type	Value	Attr
EOFLINE	NUMBER		0024
GREETING	L BYTE	0010	DATA
HELLO	L BYTE	0000	DATA
MAIN	F PROC	0005	CODE Length = 0016
WRITEMSG	N PROC	0000	CODE Length = 0005
@CPU	TEXT	0101h	
@FILENAME	TEXT	hello2	
@VERSION	TEXT	510	

52 Source Lines

52 Total Lines

13 Symbols

47986 + 459271 Bytes symbol space free

0 Warning Errors

0 Severe Errors