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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Организация ЭВМ и систем»

Тема: Изучение режимов адресации и формирования исполнительного адреса

Студент гр. 1381 _____ Сагидуллин Э.Р.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить основные принципы трансляции, отладки и выполнения программ на языке Ассемблера. Разобраться в режимах адресации и полученных результатах.

Задание

Лабораторная работа 2 предназначена для изучения режимов адресации, использует готовую программу lr2_comp.asm на Ассемблере, которая в автоматическом режиме выполняться не должна, так как не имеет самостоятельного функционального назначения, а только тестирует режимы адресации. Поэтому ее выполнение должно производиться под управлением отладчика в пошаговом режиме.

В программу введен ряд ошибок, которые необходимо объяснить в отчете по работе, а соответствующие команды закомментировать для прохождения трансляции. Необходимо составить протокол выполнения программы в пошаговом режиме отладчика по типу таблицы 1 предыдущей лабораторной работы и подписать его у преподавателя. На защите студенты должны уметь объяснить результат выполнения каждой команды с учетом используемого вида адресации. Результаты, полученные с помощью отладчика, не являются объяснением, а только должны подтверждать ваши объяснения.

Порядок выполнения работы.

1. Получить у преподавателя вариант набора значений исходных данных (массивов) vec1, vec2 и matr из файла lr2.dat, приведенного в каталоге Задания и занести свои данные вместо значений, указанных в приведенной ниже программе.

2. Протранслировать программу с созданием файла диагностических сообщений; объяснить обнаруженные ошибки и закомментировать соответствующие операторы в тексте программы.

3. Снова протранслировать программу и скомпоновать загрузочный модуль.

4. Выполнить программу в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения команды.

5. Результаты прогона программы под управлением отладчика должны быть подписаны преподавателем и представлены в отчете.

Ход выполнения работы

Значения vec1, vec2, matr были изменены в соответствие с вариантом. Далее, программа была протранслирована с созданием файла диагностических сообщений.

Объяснение ошибок:

1) строка 45 mov mem3, [bx] --- были использованы косвенный и прямой адрес в mov. Тип операнда, нельзя читать из памяти и писать в память одной командой. Необходимо перевести информацию из памяти в регистр, а затем из регистра в необходимый сегмент.

2) строка 57 mov ax,matr[bx*4][di] --- ошибка использования базово-индексной адресации. Такая форма адресации используется в тех случаях, когда в регистре находится адрес начала структуры данных, а доступ надо осуществить к какому-нибудь элементу этой структуры.

3) строка 76 mov ax,matr[bp+bx] --- нельзя складывать регистры bp и bx. Так как оба регистра базовые, необходимо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра.

4) строка 77 mov ax,matr[bp+di+si] --- нельзя складывать регистры di и si. Так как используются два индексных регистра, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра.

5) строка 84 ret 2 --- берется неверный адрес.

Строки программы вызвавшие ошибки компиляции были обозначены комментариями. Была произведена повторная трансляция программы и компоновка загрузочного модуля. Далее, программа была выполнена в пошаговом режиме с помощью отладчика. Значения используемых регистров и ячеек памяти до и после выполнения команд зафиксированы в таблицы.

Таблица 1. Начальное значение регистров

CS	DS	ES	SS
1A0A	19F5	19F5	1A05

Таблица 2. Протокол работы программы lr2_comp

Адрес команды	Символьный код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(DS)=19F5 (IP)=0000 (SP)=0018 Stack +0 0000 +2 0000 +4 0000 +6 0000	(DS)=19F5 (IP)=0001 (SP)=0016 Stack +0 19F5 +2 0000 +4 0000 +6 0000
0001	SUB AX, AX	2BC0	(IP)=0001 (AX)=0000	(IP)=0003 (AX)=0000
0003	PUSH AX	50	(AX)=0000 (IP)=0003 (SP)=0016 Stack +0 19F5 +2 0000 +4 0000 +6 0000	(AX)=0000 (IP)=0004 (SP)=0014 Stack +0 0000 +2 19F5 +4 0000 +6 0000

0004	MOV AX, 1A07	B8071A	(AX)=0000 (IP)=0004	(AX)=1A07 (IP)=0007
0007	MOV DS, AX	8ED8	(AX)=1A07 (DS)=19F5 (IP)=0007	(AX)=1A07 (DS)=1A07 (IP)=0009
0009	MOV AX, 01F4	B8F401	(AX)=1A07 (IP)=0009	(AX)=01F4 (IP)=000C
000C	MOV CX, AX	8BC8	(CX)=0000 (IP)=000C	(CX)=01F4 (IP)=000E
000E	MOV BL, 24	B324	(BX)=0000 (IP)=000E	(BX)=0024 (IP)=0010
0010	MOV BH, CE	B7CE	(BX)=0024 (IP)=00010	(BX)=CE24 (IP)=0012
0012	MOV [0002],FFCE	C7060200CEFF	DS:0000 +2 00 +3 00 (IP)=012	DS:0000 +2 CE +3 FF (IP)=0018
0018	MOV BX, 0006	BB0600	(BX)=CE24 (IP)=0018	(BX)=0006 (IP)=001B
001B	MOV [0000], AX	A30000	DS:0000 +0 00 +1 00 (IP)=001B	DS:0000 +0 F4 +1 01 (IP)=001E
001E	MOV AL, [BX]	8A07	(AX)=01F4 (IP)=002E	(AX)=0126 (IP)=0020
0020	MOV AL, [BX+03]	8A4703	(AX)=0126 (BX)=0006 (IP)=0020	(AX)=0123 (BX)=1A07 (IP)=0023
0023	MOV CX, [BX+03]	8B4F03	(CX)=01F4 (IP)=0023	(CX)=1F23 (IP)=0026
0026	MOV DI, 0002	BF0200	(DI)=0000 (IP)=0026	(DI)=0002 (IP)=0029
0029	MOV AL, [000E+DI]	8A850500	(AX)=0123 (DI)=0002 (IP)=0029	(AX)=01BA (DI)=0002 (IP)=002D
002D	MOV BX, 0003	BB0300	(BX)=0006 (IP)=002D	(BX)=0003 (IP)=0030

0030	MOV AL, [0016+BX+DI]	8A811600	(AX)=01BA (IP)=0030	(AX)=01F9 (IP)=0034
0034	MOV AX, 1A07	B8071A	(AX)=01F9 (IP)=0034	(AX)=1A07 (IP)=0037
0037	MOV ES, AX	8EC0	(ES)=19F5 (AX)=1A07 (IP)=0037	(ES)=1A07 (AX)=1A07 (IP)=0039
0039	MOV AX, ES:[BX]	268B07	(AX)=1A07 (ES)=1A07 (BX)=0003 (IP)=0039	(AX)=00FF (ES)=1A07 (BX)=0003 (IP)=003C
003C	MOV AX, 0000	B80000	(AX)=00FF (IP)=003C	(AX)=0000 (IP)=003F
003F	MOV ES, AX	8EC0	(AX)=0000 (ES)=1A07 (IP)=003F	(AX)=0000 (ES)=0000 (IP)=0041
0041	PUSH DS	1E	(SP)=0014 (IP)=0041 Stack +0 0000 +2 19F5 +4 0000 +6 0000	(SP)=0012 (IP)=0042 Stack +0 1A07 +2 0000 +4 19F5 +6 0000
0042	POP ES	07	(ES)=0000 (SP)=0012 (IP)=0042 Stack +0 1A07 +2 0000 +4 19F5 +6 0000	(ES)=1A07 (SP)=0014 (IP)=0043 Stack +0 0000 +2 19F5 +4 0000 +6 0000
0043	MOV CX, ES:[BX-01]	268B4FFF	(CX)=03F9 (ES)=1A07 (BX)=0003 (IP)=0043	(CX)=FFCE (ES)=1A07 (BX)=0003 (IP)=0047
0047	XCHG AX, CX	91	(AX)=0000 (CX)=FFCE (IP)=0047	(AX)=FFCE (CX)=0000 (IP)=0048
0048	MOV DI, 0002	BF0200	(DI)=0002 (IP)=0048	(DI)=0002 (IP)=004B
004B	MOV ES:[BX+DI], AX	268901	ES:0000 +5 00 +6 26 (BX)=0003 (DI)=0002 (IP)=004B	ES:0000 +5 CE +6 FF (BX)=0003 (DI)=0002 (IP)=004E

004E	MOV BP. SP	8BEC	(BP)=0000 (SP)=0014 (IP)=004E	(BP)=0014 (SP)=0014 (IP)=0050
0050	PUSH [0000]	FF360000	(SP)=0014 (IP)=0050 Stack +0 0000 +2 19F5 +4 0000 +6 0000	(SP)=0012 (IP)=0054 Stack +0 01F4 +2 0000 +4 19F5 +6 0000
0054	PUSH [0002]	FF360200	(SP)=0012 (IP)=0054 Stack +0 01F4 +2 0000 +4 19F5 +6 0000	(SP)=0010 (IP)=0058 Stack +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	MOV BP, SP	8BEC	(BP)=0014 (SP)=0010 (IP)=0058	(BP)=0010 (SP)=0010 (IP)=005A
005A	MOV DX, [BP+02]	8B5602	(DX)=0000 (BP)=0010 (IP)=005A	(DX)=01F4 (BP)=0010 (IP)=005D
005D	RET Far 0002	CA0200	(IP)=005D (CS)=1A0A (SP)=0010 Stack +0 FFCE +2 01F4 +4 0000 +6 19F5	(IP)=FFCE (CS)=01F4 (SP)=0016 Stack +0 19F5 +2 00000 +4 0000 +6 0000

Вывод

В ходе выполнения лабораторной работы были освоены базовые навыки программирования на ассемблере, изучены основные режимы адресации памяти.

ПРИЛОЖЕНИЕ А
ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ

Файл lr2_comp.asm:

```
; Программа изучения режимов адресации процессора IntelX86
     EOL EQU '$'
     ind EQU 2
     n1 EQU 500
     n2 EQU -50

; Стек программы
AStack SEGMENT STACK
          DW 12 DUP(?)
AStack ENDS

; Данные программы
DATA SEGMENT
; Директивы описания данных
     mem1 DW 0
     mem2 DW 0
     mem3 DW 0
     vec1 DB 38,37,36,35,31,32,33,34
     vec2 DB 70,80,-70,-80,50,60,-50,-60
     matr DB -2,-1,5,6,-8,-7,3,4,-4,-3,7,8,-6,-5,1,2
DATA ENDS

; Код программы
CODE SEGMENT
     ASSUME CS:CODE, DS:DATA, SS:AStack
; Головная процедура
Main      PROC FAR
     push DS
     sub AX,AX
     push AX
     mov AX,DATA
     mov DS,AX
; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
; Регистровая адресация
     mov ax,n1
     mov cx,ax
```

```
    mov bl,EOL
    mov bh,n2
; Прямая адресация
    mov mem2,n2
    mov bx,OFFSET vec1
    mov mem1,ax
; Косвенная адресация
    mov al,[bx]
    mov mem3,[bx]
; Базированная адресация
    mov al,[bx]+3
    mov cx,3[bx]
; Индексная адресация
    mov di,ind
    mov al,vec2[di]
    mov cx,vec2[di]
; Адресация с базированием и индексированием
    mov bx,3
    mov al,matr[bx][di]
    mov cx,matr[bx][di]
    mov ax,matr[bx*4][di]
; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегмента
; ------ вариант 1
    mov ax, SEG vec2
    mov es, ax
    mov ax, es:[bx]
    mov ax, 0
; ------ вариант 2
    mov es, ax
    push ds
    pop es
    mov cx, es:[bx-1]
    xchg cx,ax
; ------ вариант 3
    mov di,ind
    mov es:[bx+di],ax
; ------ вариант 4
    mov bp,sp
    mov ax,matr[bp+bx]
    mov ax,matr[bp+di+si]
```

```
; Использование сегмента стека
     push mem1
     push mem2
     mov bp,sp
     mov dx,[bp]+2
     ret 2
Main      ENDP
CODE ENDS
          END Main
```

ПРИЛОЖЕНИЕ Б
ФАЙЛЫ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

```
#Microsoft (R) Macro Assembler Version 5.10              10/9/22 19:38:06
                                                        Page     1-1


                                ; РµСЂРsРiСЂР°РjРjР° РёР·СґС‡РµРSРёСЏ СЂРµР¶РёР
                                jPsPI Р°РrСЂРµСЃР°С†РёРё РїСЂРsС†РµСЃСґРsСЂР° I
                                ntelX86
= 0024                                  EOL EQU '$'
= 0002                                  ind EQU 2
= 01F4                                  n1 EQU 500
=-0032                                  n2 EQU -50

                                ; РЎС‚РµРє РїСЂРsРiСЂР°РjРjС‹
0000                            AStack SEGMENT STACK
0000  000C[                             DW 12 DUP(?)
        ????
                    ]

0018                            AStack ENDS

                                ; Р”Р°РSPSC‹Рµ РїСЂРsРiСЂР°РjРjС‹
0000                            DATA SEGMENT
                                ; Р”РёСЂРµРєС‚РёРIС‹ РsРїРёСЃР°РSРёСЏ РrР°РSPSC
                                ‹С…
0000  0000                              mem1 DW 0
0002  0000                              mem2 DW 0
0004  0000                              mem3 DW 0
0006  26 25 24 23 1F 20                  vec1 DB 38,37,36,35,31,32,33,34
    21 22
000E  46 50 BA B0 32 3C                 vec2 DB 70,80,-70,-80,50,60,-50,-60
    CE C4
0016  FE FF 05 06 F8 F9                 matr DB -2,-1,5,6,-8,-7,3,4,-4,-3,7,8,-
                                6,-5,1,2
    03 04 FC FD 07 08
    FA FB 01 02
0026                            DATA ENDS

                                ; РљРsРr РїСЂРsРiСЂР°РjРjС‹
0000                            CODE SEGMENT
                                        ASSUME CS:CODE, DS:DATA, SS:AStack
                                ; Р“РsР»РsРIРSР°СЏ РїСЂРsС†РµРrСґСЂР°
0000                            Main    PROC FAR
0000  1E                               push DS
0001  2B C0                            sub AX,AX
0003  50                               push AX
0004  B8 ---- R                        mov AX,DATA
0007  8E D8                            mov DS,AX
                                ; РµР ЋР’Р•Р РљРђ Р Р•Р–Р□РњРћР’ РђР”Р Р•РЎРђР
                                ¦Р□Р□ РќРђ РЈР ЋР’РќР• РЎРњР•Р©Р•РќР□Р™
```

```
                        ; P РµPiPëCЃC,CЂPsPIP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0009  B8 01F4                   mov ax,n1
000C  8B C8                 mov cx,ax
000E  B3 24                 mov bl,EOL
0010  B7 CE                 mov bh,n2
                        ; РџCЂCЏPjP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0012  C7 06 0002 R FFCE     mov mem2,n2
0018  BB 0006 R             mov bx,OFFSET vec1
001B  A3 0000 R             mov mem1,ax

#Microsoft (R) Macro Assembler Version 5.10          10/9/22 19:38:06
                                Page     1-2


                        ; РљPsCЃPIPµPSPSP°CЏ P°PrCЂPµCЃP°C†PëCЏ
001E  8A 07                 mov al,[bx]
                        ;mov mem3,[bx]
                        ; P‘P°P·PëCЂPsPIP°PSPSP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0020  8A 47 03                   mov al,[bx]+3
0023  8B 4F 03                   mov cx,3[bx]
                        ; P□PSPrPµPєCЃPSP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0026  BF 0002                    mov di,ind
0029  8A 85 000E R          mov al,vec2[di]
002D  8B 8D 000E R                mov cx,vec2[di]
lr2_comp.asm(52): warning A4031: Operand types must match
                        ; PђPrCЂPµCЃP°C†PëCЏ CЃ P±P°P·PëCЂPsPIP°PSPëPµP
                        j Pë PëPSPrPµPєCЃPëCЂPsPIP°PSPëPµPj
0031  BB 0003                    mov bx,3
0034  8A 81 0016 R          mov al,matr[bx][di]
0038  8B 89 0016 R          mov cx,matr[bx][di]
lr2_comp.asm(56): warning A4031: Operand types must match
                        ;mov ax,matr[bx*4][di]
                        ; РџP РћP’P•P РљPћ P P•P–P□PњPћP’ PћP”P P•PЎPћP
                        ¦P□P□ PЎ PJP§P•PўPћPњ PЎP•P“PњP•PќPўPћP’
                        ; РџPµCЂPµPsPїCЂPµPrPµP»PµPSPëPµ CЃPµPiPjPµPSC,
                        P°
                        ; ------ PIP°CЂPëP°PSC, 1
003C  B8 ---- R              mov ax, SEG vec2
003F  8E C0                  mov es, ax
0041  26: 8B 07              mov ax, es:[bx]
0044  B8 0000                    mov ax, 0
                        ; ------ PIP°CЂPëP°PSC, 2
0047  8E C0                  mov es, ax
0049  1E                     push ds
004A  07                     pop es
004B  26: 8B 4F FF           mov cx, es:[bx-1]
004F  91                     xchg cx,ax
                        ; ------ PIP°CЂPëP°PSC, 3
0050  BF 0002                    mov di,ind
0053  26: 89 01              mov es:[bx+di],ax
                        ; ------ PIP°CЂPëP°PSC, 4
0056  8B EC                  mov bp,sp
```

```
                        ;mov ax,matr[bp+bx]
                        ;mov ax,matr[bp+di+si]
                      ; P□CЃPїPsP»CЊP·PsPIP°PSPёPµ CЃPµPiPjPµPSC,P° C
                      ЃC,PµPєP°
0058  FF 36 0000 R             push mem1
005C  FF 36 0002 R             push mem2
0060  8B EC                    mov bp,sp
0062  8B 56 02                       mov dx,[bp]+2

                        ;ret 2
0065                  Main  ENDP
0065                  CODE  ENDS
                              END Main
```

#Microsoft (R) Macro Assembler Version 5.10 10/9/22 19:38:06
Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine Class
ASTACK	0018	PARA	STACK
CODE	0065	PARA	NONE
DATA	0026	PARA	NONE

Symbols:

N a m e	Type	Value	Attr
EOL	NUMBER	0024	
IND	NUMBER	0002	
MAIN	F PROC	0000	CODE Length = 0065
MATR	L BYTE	0016	DATA
MEM1	L WORD	0000	DATA
MEM2	L WORD	0002	DATA
MEM3	L WORD	0004	DATA
N1	NUMBER	01F4	
N2	NUMBER	-0032	
VEC1	L BYTE	0006	DATA
VEC2	L BYTE	000E	DATA
@CPU	TEXT	0101h	
@FILENAME	TEXT	lr2_comp	
@VERSION	TEXT	510	

87 Source Lines

87 Total Lines
19 Symbols

47798 + 455365 Bytes symbol space free

2 Warning Errors