

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
ВАРИАНТ №4

Студент гр. 1383

Манучарова А.С.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить режимы адресации и формирование исполнительного адреса.

Задание.

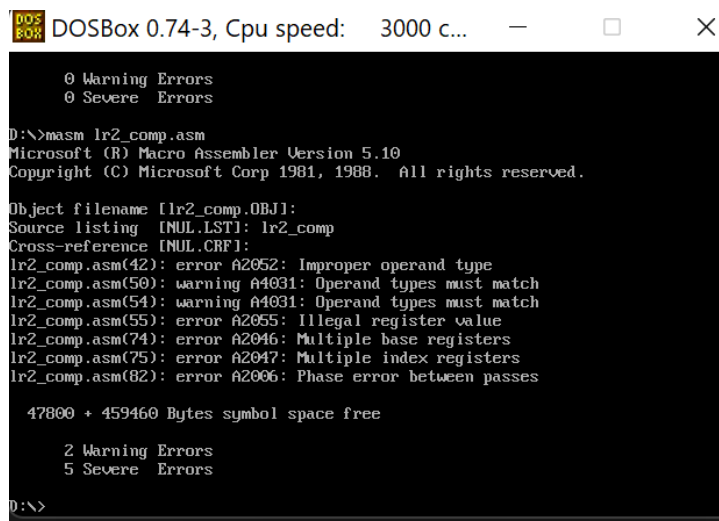
Лабораторная работа 2 предназначена для изучения режимов адресации, использует готовую программу `lr2_comp.asm` на Ассемблере, которая в автоматическом режиме выполняться не должна, так как не имеет самостоятельного функционального назначения, а только тестирует режимы адресации. Поэтому ее выполнение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

Выполнение работы.

1. Были изменены значения `vec1`, `vec2`, `matr` согласно варианту.
2. Была проведена попытка трансляции файла с получением ошибок, которые показаны на рисунке 1.



```
DOSBox 0.74-3, Cpu speed: 3000 c...
0 Warning Errors
0 Severe Errors
D:\>masm lr2_comp.asm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [lr2_comp.OBJ]:
Source listing [NUL.LST]: lr2_comp
Cross-reference [NUL.CRF]:
lr2_comp.asm(42): error A2052: Improper operand type
lr2_comp.asm(50): warning A4031: Operand types must match
lr2_comp.asm(54): warning A4031: Operand types must match
lr2_comp.asm(55): error A2055: Illegal register value
lr2_comp.asm(74): error A2046: Multiple base registers
lr2_comp.asm(75): error A2047: Multiple index registers
lr2_comp.asm(82): error A2006: Phase error between passes

47800 + 459460 Bytes symbol space free

2 Warning Errors
5 Severe Errors
D:\>
```

Рисунок 1 — Ошибки при первой трансляции файла

3. Были закомментированы строки:

- a) `mov mem3, [bx]` – обращение к области памяти возможно только через регистр.
- b) `mov cx, vec2[di]` – различие в величине операндов. Регистр `cx` имеет размер 2б, а `vec2[di]` – 1б.
- c) `mov cx, matr[bx][di]` – разная длина операндов. `cx` – 2б, `matr[bx][di]` – 1б.
- d) `mov ax, matr[bx*4][di]` – в базово-индексной адресации не предусмотрено масштабирование. Оно уместно в тех случаях, когда массив состоит не из байт, а из слов. В таком случае применяется базово-индексная адресация с масштабированием.
- e) `mov ax, matr[bp+bx]` – базовый регистр должен быть один.
- f) `mov ax, matr[bp+di+si]` – индексный регистр должен быть один.
- g) для очищения стека были добавлены `pop ax`, `pop bx`

4. Выполнена трансляция файла.

5. Начальное состояние режимов:

`CS = 1A0A, DS = 19F5, ES = 19F5, SS = 1A05`

Таблица 1. Протокол lr2_compr.exe

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	push dx	1E	IP = 0000 SP = 0018 +0 0000	IP = 0001 SP = 0013 +0 19F5
0001	sub ax, ax	2B C0	IP = 0001 SP = 0013	IP = 0003 SP = 0016
0003	push ax	50	IP = 0003 SP = 0016 +0 19F5 +2 0000	IP = 0004 SP = 0014 +0 0000 +2 19F5
0004	mov ax, 1A07	B8 07 1A	IP = 0004 AX = 0000	IP = 0007 AX = 1A07
0007	mov ds, ax	8E D8	IP = 0007 DS = 19F5	IP = 0009 DS = 1A07
0009	mov ax, 01F4	B8 F4 01	AX = 1A07 IP = 0009	AX = 01F4 IP = 000C
000C	mov cx, ax	8B C8	CX = 0000 IP = 000C	CX = 01F4 IP = 000E
000E	mov bl, 24	B3 24	BX = 0000 IP = 000E	BX = 0024 IP = 0010
0010	mov bh, ce	B7 CE	BX = 0024 IP = 0010	BX = CE24 IP = 0012
0012	mov [0002], FFCE	C7 06 02 00 CE FF	IP = 0012	IP = 0018
0018	mov bx, 0006	BB 06 00	BX = CE24 IP = 0018	BX = 0006 IP = 001B
001B	mov [0000], ax	A3 00 00	IP = 001B	IP = 001E

001E	mov al, bx	8A 07	AX = 01F4 IP = 001E	AX = 010C IP = 0020
0020	mov al, [bx + 3]	8A 47 03	IP = 0020 AX = 010C	IP = 0023 AX = 0109
0023	mov cx, [bx + 3]	8B 4F 03	IP = 0023 CX = 01F4	IP = 0026 CX = 0509
0026	mov di, 0002	BF 02 00	DI = 0000 IP = 0026	DI = 0002 IP = 0029
0029	mov al, [000E + di]	8A 85 0E 00	AX = 0109 IP = 0029	AX = 0128 IP = 002D
002D	mov bx, 0003	BB 00 03	BX = 0006 IP = 002D	BX = 0003 IP = 0030
0030	mov al, [0016 + bx + di]	8A 81 16 00	AX = 0128 IP = 0030	AX = 07F9 IP = 0034
0034	mov ax, 1A07	B8 07 1A	AX = 07F9 IP = 0034	AX = 1A07 IP = 0037
0037	mov es, ax	8EC0	ES = 19F5 IP = 0037	ES = 1A07 IP = 0039
0039	mov ax, es:[bx]	26 8B 07	AX = 1A07 IP = 0039	AX = 00FF IP = 003C
003C	mov ax, 0000	B8 00 00	AX = 00FF IP = 003C	AX = 0000 IP = 003F
003F	mov es, ax	8E C0	ES = 1A07 IP = 003F	ES = 0000 IP = 0041
0041	push ds	1E	IP = 0041 +0 0000 +2 19F5 +4 0000	IP = 0042 +0 1A07 +2 0000 +4 19F5
0042	pop es	07	ES = 0000 IP = 0042 +0 1A07	ES = 1A07 IP = 0043 +0 0000

			+2 0000 +4 19F5	+2 19F5 +4 0000
0043	mov cx, es:[bx-01]	26 8B 4F FF	CX = 0509 IP = 0043	CX = FFCE IP = 0047
0047	xchg ax,cx	91	AX = 0000 CX = FFCE IP = 0047	AX = FFCE CX = 0000 IP = 0048
0048	mov di, 0002	BF 02 00	DI = 0002 IP = 0048	DI = 0002 IP = 004B
004B	mov es:[bx + di], ax	26 89 01	IP = 004B	IP = 004E
004E	mov bp, sp	8B EC	BP = 0000 IP = 004E	BP = 0014 IP = 0050
0050	push [0000]	FF 36 00 00	SP = 0014 IP = 0050 +0 0000 +2 19F5 +4 0000	SP = 0012 IP = 0054 +0 01F4 +2 0000 +4 19F5
0054	push [0002]	FF 36 02 00	SP = 0012 IP = 0054 +0 01F4 +2 0000 +4 19F5 +6 0000	SP = 0010 IP = 0058 +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	mov bp, sp	8B EC	BP = 0014 IP = 0058	BP = 0010 IP = 005A
005A	mov dx, [bp + 02]	8B 56 02	DX = 0000 IP = 005D	DX = 01F4 IP = 005A
005D	pop ax	58	AX = FFCE IP = 005D +0 FFCE +2 01F4 +4 0000	AX = FFCE IP = 005E +0 01F4 +2 0000 +4 19F5

			+6 19F5	+6 0000
005E	pop bx	5B	BX = 0000 IP = 005E	BX = 01F4 IP = 005F
005F	ret far	CB	IP = 005F CS = 1A0A SP = 0014 +0 01F4 +2 0000 +4 19F5 +6 0000	IP = 0000 CS = 19F5 SP = 0018 +0 0000 +2 0000 +4 000 +6 0000
0000	int 20	cd 20		

Выводы.

Были изучены режимы адресации и формирование исполнительного адреса на языке Ассемблер.

ПРИЛОЖЕНИЕ А

КОД ПРОГРАММЫ

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 12,11,10,9,5,6,7,8
vec2 DB -40,-50,40,50,-20,-30,20,30
matr DB 5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,-3,-2,-1
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
pop ax
pop bx
ret
Main ENDP
CODE ENDS
END Main

```

ПРИЛОЖЕНИЕ В

ЛИСТИНГ ПРОГРАММЫ

lr2_comp.lst:

Microsoft (R) Macro Assembler Version
5.10 10/9/22 13:26:48

Page 1-1

```
; Программа изучения режи?
?ов адресации процессора I
ntelX86

= 0024          EOL EQU '$'
= 0002          ind EQU 2
= 01F4          n1 EQU 500
=-0032          n2 EQU -50

; Стек программы

0000          AStack SEGMENT STACK
0000 000C[          DW 12 DUP(?)
          ????
          ]

0018          AStack ENDS

; Данные программы

0000          DATA SEGMENT

; Директивы описания данн?
?x

0000 0000          mem1 DW 0
0002 0000          mem2 DW 0
```

```

0004 0000 mem3 DW 0
0006 0C 0B 0A 09 05 06 vec1 DB
12,11,10,9,5,6,7,8
07 08
000E D8 CE 28 32 EC E2 vec2 DB -40,-50,40,50,-
20,-30,20,30
14 1E
0016 05 06 07 08 F8 F9 matr DB 5,6,7,8,-8,-7,-
6,-5,1,2,3,4,-4,-3,-2,-1
FA FB 01 02 03 04
FC FD FE FF
0026 DATA ENDS
; Код программы
0000 CODE SEGMENT
ASSUME CS:CODE, DS:DATA, SS:AStack
; Головная процедура
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
; ПРОВЕРКА РЕЖИМОВ АДРЕСА?
?ИИ НА УРОВНЕ СМЕЩЕНИЙ
; Регистровая адресация
0009 B8 01F4 mov ax,n1
000C 8B C8 mov cx,ax
000E B3 24 mov bl,EOL
0010 B7 CE mov bh,n2
; Прямая адресация

```

```

0012  C7 06 0002 R FFCE      mov mem2,n2
0018  BB 0006 R              mov bx,OFFSET vec1
001B  A3 0000 R              mov mem1,ax
                                ; Косвенная адресация
001E  8A 07                  mov al,[bx]
                                ;mov mem3,[bx]
                                ; Базированная адресация

```

```

Microsoft      (R)      Macro      Assembler      Version
5.10           10/9/22 13:26:48

```

Page 1-2

```

0020  8A 47 03              mov al,[bx]+3
0023  8B 4F 03              mov cx,3[bx]
                                ; Индексная адресация
0026  BF 0002              mov di,ind
0029  8A 85 000E R          mov al,vec2[di]
                                ;mov cx,vec2[di]
                                ; Адресация с базирование?
                                ? и индексированием
002D  BB 0003              mov bx,3
0030  8A 81 0016 R          mov al,matr[bx][di]
                                ;mov cx,matr[bx][di]
                                ;mov ax,matr[bx*4][di]
                                ; ПРОВЕРКА РЕЖИМОВ АДРЕСА?
                                ?ИИ С УЧЕТОМ СЕГМЕНТОВ
                                ; Переопределение сегмент
a

```

```

; ----- вариант 1
0034 B8 ---- R      mov ax, SEG vec2
0037 8E C0          mov es, ax
0039 26: 8B 07      mov ax, es:[bx]
003C B8 0000        mov ax, 0
; ----- вариант 2
003F 8E C0          mov es, ax
0041 1E            push ds
0042 07            pop es
0043 26: 8B 4F FF    mov cx, es:[bx-1]
0047 91            xchg cx,ax
; ----- вариант 3
0048 BF 0002        mov di,ind
004B 26: 89 01      mov es:[bx+di],ax
; ----- вариант 4
004E 8B EC          mov bp,sp
;mov ax,matr[bp+bx]
;mov ax,matr[bp+di+si]
; Использование сегмента ?
?тека
0050 FF 36 0000 R    push mem1
0054 FF 36 0002 R    push mem2
0058 8B EC          mov bp,sp
005A 8B 56 02        mov dx,[bp]+2
005D 58            pop ax
005E 5B            pop bx
005F CB            ret
0060              Main ENDP
0060              CODE ENDS
END Main

```

Microsoft (R) Macro Assembler Version
 5.10 10/9/22 13:26:48

Symbols-1

Segments and Groups:

					N	a	m
	Length	Align	Combine	Class			
ASTACK			0018	PARA	S	
TACK							
CODE			0060	PARA	N	
ONE							
DATA			0026	PARA	N	
ONE							

Symbols:

					N	a	m
	Type	Value	Attr				
EOL			NUMBER	0024		
IND			NUMBER	0002		
MAIN						F
PROC	0000	CODE	Length = 0060				
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

```

```

86 Source Lines
86 Total Lines
19 Symbols

```

47800 + 459460 Bytes symbol space free

```

0 Warning Errors
0 Severe Errors

```