

**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 №3**

Студент гр. 8381

\_\_\_\_\_

Луценко Д.А.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2019

### **Цель работы.**

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

### **Задание.**

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 работы.**

1. Изменение набора значений исходных данных (массивов) `vec1`, `vec2` и `matr` из файла `lr2.dat`, согласно своему варианту.

2. Трансляция программы с созданием файла диагностических сообщений.
- Объяснение обнаруженных ошибок и предупреждений и закомментирование операторов с ошибками в тексте программы.

```
C:\>LB2.LST
Illegal command: LB2.LST.

C:\>MASM.EXE LB2.ASM
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [LB2.OBJ]:
Source listing [NUL.LST]: lb2
Cross-reference [NUL.CRF]:
LB2.ASM(42): error A2052: Improper operand type
LB2.ASM(49): warning A4031: Operand types must match
LB2.ASM(53): warning A4031: Operand types must match
LB2.ASM(54): error A2055: Illegal register value
LB2.ASM(73): error A2046: Multiple base registers
LB2.ASM(74): error A2047: Multiple index registers
LB2.ASM(81): error A2006: Phase error between passes

47842 + 461465 Bytes symbol space free

2 Warning Errors
5 Severe Errors

C:\>_
```

- Ошибка lb2.asm(42): error A2052: Improper operand type (Неверный тип операнда)

**Строка 42:** mov mem3, [bx]

Тип операнда, нельзя читать из памяти и писать в память одной командой. В данном случае необходимо перевести информацию из памяти в регистр, а затем уже перевести информацию из регистра в необходимый сегмент.

- Предупреждение lb2.asm(49): warning A4031: Operand types must match (Несоответствие типов операндов)

**Строка 49:** mov cx, vec2[di]

Несоответствие размеров операндов, cx – 1 байта, элемент vec2[di] – 1 байт.

- Предупреждение lb2.asm(53): warning A4031: Operand types must match (Несоответствие типов операндов)

**Строка 53:** `mov cx, matr[bx][di]`

Несоответствие размеров операндов, cx – 2 байта, элемент matr[bx][di] – 1 байт.

- Ошибка lb2.asm(54): error A2055: Illegal register value (Незаконное использование регистра)

**Строка 54:** `mov ax, matr[bx*4][di]`

Здесь используется базово-индексная адресация. 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. При данном типе адресации надо сначала изменить значение регистра, затем уже переводить информацию.

- Ошибка lb2.asm(73): error A2046: Multiple base registers (несколько базовых регистров)

**Строка 73:** `mov ax, matr[bp+bx]`

Нельзя складывать регистры bp и bx. Так как здесь оба регистра базовые, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра.

- Ошибка lb2.asm(74): error A2047: Multiple index registers (несколько индексных регистров)

**Строка 74:** `mov ax, matr[bp+di+si]`

Нельзя складывать регистры di и si. Так как здесь два индексных регистра, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра.

- Ошибка lb2.asm(81): error A2006: Phase error between passes

### Строка 95: Main ENDP

Данная ошибка свидетельствует о том, что в функции main содержатся ошибки.

3. Повторная трансляция программы и компоновка загрузочного модуля.

```
Z:\>SET BLASTER=A220 17 D1 H5 T6

Z:\>MOUNT C "D:\Assembler\DOSBox-0.74-3\comp_arch_materials\labs\tools"
Drive C is mounted as local directory D:\Assembler\DOSBox-0.74-3\comp_arch_mater
ials\labs\tools\

Z:\>C:

C:\>LB2.ASM
Illegal command: LB2.ASM.

C:\>masm LB2.ASM
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [LB2.OBJ]: lb2
Source listing [NUL.LST]: lb2Now
Cross-reference [NUL.CRF]:

47832 + 461475 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>
```

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

Начальные значения регистров:

AX = 0000 SI = 0000 CS = 1A0A IP = 0000 Stack = 0000

BX = 0000 DI = 0000 DS = 19F5

CX = 00B0 BP = 0000 ES = 19F5

DX = 0000 SP = 0018 SS = 1A05

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

0003	PUSH AX	50	(SP)= 0016 (IP)= 0003 (AX)=0000 Stack = 19F5	(SP)= 0014 (IP)= 0004 (AX)=0000 Stack(+0)= 0000 Stack(+2)=19F5
0004	MOV AX, 1A07	B9071A	(AX)=0000 (IP)=0004	(AX)=1A07 (IP)=0007
0007	MOV DS,AX	8ED8	(DS)=19F5 (IP)= 0007 (AX) = 1A07	(DS)= 1A07 (IP)= 0009 (AX) = 1A07
0009	MOV AX, 01F4	B8F401	(AX)=1A07 (IP)=0009	(AX)= 01F4 (IP)= 000C
000C	MOV CX,AX	8BC8	(IP)=000C (CX)=00B0 (AX) = 01F4	(IP)=000E (CX)=01F4 (AX)=01F4
000E	MOV BL,24	B324	(BX)=0000 (IP)=000E	(BX)=0024 (IP)=0010
0010	MOV BH,CE	B7CE	(BX)=0024 (IP)=0010	(BX)=CE24 (IP)=0012
0012	MOV [0002],FFCE	C7060200CEFF	(IP)=0012	(IP)=0018
0018	MOV BX,0006	BB0600	(BX)=CE24 (IP)=0018	(BX)=0006 (IP)=001B
001B	MOV [0000],AX	A30000	(IP)=001B	(IP)=001E
001E	MOV AL,[BX]	8A07	(AX)=01F4 (IP)=001E	(AX)=0108 (IP)=0020
0020	MOV AL,[BX+03]	8A4703	(IP) = 0020 (AX) = 0108	(IP)= 0023 (AX) = 0105
0023	MOV CX, [BX+03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 0105 (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [DI+ 000E]	8A850E00	(AX) = 0105 (IP) = 0029	(AX)= 011E (IP)= 002D

002D	MOV BX, 0003	BB03000	(IP) = 002D (BX) = 0006	(IP) = 0030 (BX) = 0003
0030	MOV AL, [BX+DI+DI]	8A811600	(IP) = 0030 (AX) = 011E	(IP) = 0034 (AX) = 0107
0034	MOV AX, 1A07	B8071A	(AX) = 0107 (IP) = 0034	(AX) = 1A07 (IP) = 0037
0037	MOV ES, AX	8EC0	(ES) = 19F5 (IP) = 0037	(ES) = 1A07 (IP) = 0039
0039	MOV AX, ES:[BX]	268B07	(AX) = 1A07 (IP) = 0039	(AX) = 00FF (IP) = 003C
003C	MOV AX, 0000	B80000	(AX) = 00FF (IP) = 003C	(AX) = 0000 (IP) = 003F
003F	MOV ES, AX	8EC0	(ES) = 1A07 (IP) = 003F	(ES) = 0000 (IP) = 0041
0041	PUSH DS	1E	(IP) = 0041 (SP) = 0014 Stack(+0) = 0000 Stack(+2) = 19F5 Stack(+4) = 0000	(IP) = 0042 (SP) = 0012 Stack(+0) = 1A07 Stack(+2) = 0000 Stack(+4) = 19F5
0042	POP ES	07	(SP) = 0012 (ES) = 0000 (IP) = 0042 Stack(+0) = 1A07 Stack(+2) = 0000 Stack(+4) = 19F5	(SP) = 0014 (ES) = 1A07 (IP) = 0043 Stack(+0) = 0000 Stack(+2) = 19F5 Stack(+4) = 0000
0043	MOV CX, ES:[BX—01]	268B4FFF	(CX) = 0105 (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	BF0200	(IP) = 0048	(IP) = 004B
004B	MOV ES:[BX+DI], AX	268901	(IP) = 004B	(IP) = 004E
004E	MOV BP, SP	8BEC	(IP) = 004E (BP) = 0000	(IP) = 0050 (BP) = 0014
0050	PUSH [0000]	FF360000	(IP) = 0050 (SP)=0014 Stack(+0)=0000 Stack(+2) =19F5 Stack(+4) =0000	(IP) = 0054 (SP)=0012 Stack(+0)=01F4 Stack(+2) =0000 Stack(+4) =19F5
0054	PUSH [0002]	FF360200	(IP) = 0054 (SP) = 0012 Stack(+0)=01F4 Stack(+2) =0000 Stack(+4) =19F5 Stack(+6)=0000	(IP) = 0058 (SP) = 0010 Stack(+0)=FFCE Stack(+2) =01F4 Stack(+4) =0000 Stack(+6)= 19F5
0058	MOV BP, SP	8BEC	(IP) = 0058 (BP) = 0014	(IP) = 005A (BP) = 0010
005A	MOX DX, [BP+02]	8B5602	(IP) = 005A (DX) = 0000	(IP) = 005D (DX) = 01F4
005D	RET Far	CB	(IP) = 005D (SP) = 0010 (CS)=1A0A Stack(+0)=FFCE Stack(+2) =01F4	(IP) = FFCE (SP)= 0016 (CS)=01F4 Stack(+0)= 19F5 Stack(+2) =0000



			Stack(+4) =0000 Stack(+6)= 19F5	Stack(+4) =0000 Stack(+6)=0000
--	--	--	------------------------------------	-----------------------------------

### **Выводы.**

В ходе выполнения лабораторной работы были приобретены знания о режимах адресации и формировании исполнительного адреса в языке Ассемблера.

## Приложение А. Код программы lb2.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 8,7,6,5,1,2,3,4
vec2 DB -30,-40,30,40,-10,-20,10,20
matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-8,4,3,2,1
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
ret 2
Main ENDP
CODE ENDS

```

END Main

### Приложение Б. Листинг.

\_\_Microsoft (R) Macro Assembler Version 5.10

10/16/22 12:35:3

Page 1-1

; Программа изучения режимов адресации процессора

pa IntelX86

```

= 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 данных
0000  0000          mem1 DW 0
0002  0000          mem2 DW 0
0004  0000          mem3 DW 0
0006  08 07 06 05 01 02 vec1 DB 8,7,6,5,1,2,3,4
          03 04
000E  E2 D8 1E 28 F6 EC vec2 DB -30,-40,30,40,-10,-20,10,20
          0A 14
0016  FF FE FD FC 08 07 matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-
8,4,3,2,1
          06 05 FB FA F9 F8
          04 03 02 01

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 АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕ-
НИЙ

; 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
0020  8A 47 03          mov al,[bx]+3
0023  8B 4F 03          mov cx,3[bx]

; Индексная адресация
__Microsoft (R) Macro Assembler Version 5.10
10/16/22 12:35:3

Page      1-2

0026  BF 0002          mov di,ind
0029  8A 85 000E R      mov al,vec2[di]

;mov cx,vec2[di]

; Адресация с базированием 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 АДРЕСАЦИИ С УЧЕТОМ СЕГМЕН-
ТОВ

; Переопределение сегмента
; ----- 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 стека
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 CA 0002          ret 2
0060                Main ENDP
0060                CODE ENDS
END Main

```

\_\_Microsoft (R) Macro Assembler Version 5.10

10/16/22 12:35:3

Symbols-1

Segments and Groups:

	N a m e	Length	Align	Combine
Class				

```

ASTACK . . . . . 0018 PARA STACK
CODE . . . . . 0060 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 = 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	LB2	
@VERSION . . . . .		TEXT	510	

83 Source Lines

83 Total Lines

19 Symbols

47832 + 461475 Bytes symbol space free

0 Warning Errors

0 Severe Errors