

**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 ЯЗЫКЕ**  
**АССЕМБЛЕРА.**

Студент гр. 1381

Дудко М.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

## **Цель работы.**

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

## **Ход 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. (8 ВАРИАНТ)
2. Протранслировать программу с созданием файла диагностических сообщений; объяснить обнаруженные ошибки и закомментировать соответствующие операторы в тексте программы.

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
2 Warning Errors
3 Severe Errors

C:\>MASM.EXE LAB2.ASM
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [LAB2.OBJ]:
Source listing [NUL.LST]: Lab2.LST
Cross-reference [NUL.CRF]:
LAB2.ASM(41): error A2052: Improper operand type
LAB2.ASM(48): warning A4031: Operand types must match
LAB2.ASM(52): warning A4031: Operand types must match
LAB2.ASM(53): error A2055: Illegal register value
LAB2.ASM(72): error A2046: Multiple base registers
LAB2.ASM(73): error A2047: Multiple index registers
LAB2.ASM(80): error A2006: Phase error between passes

47828 + 459432 Bytes symbol space free

2 Warning Errors
5 Severe Errors
```

1. LAB2.ASM(41): error A2052: Improper operand type: `mov mem3,[bx]`  
Нельзя одновременно считывать и записывать тип операнда. Для корректной работы мы должны сначала записать его в регистр, и считывать уже от туда.
2. LAB2.ASM(48): warning A4031: Operand types must match: `mov cx,vec2[di]`  
Типы данных не совпадают нельзя записать в слово – cx 1 байт – `vec2[di]`
3. LAB2.ASM(52): warning A4031: Operand types must match: `mov cx,matr[bx][di]`  
Типы данных не совпадают нельзя записать в слово – cx 1 байт – `matr[bx][di]`
4. LAB2.ASM(53): error A2055: Illegal register value: `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.
5. LAB2.ASM(72): error A2046: Multiple base registers `mov ax,matr[bp+bx]`  
Нельзя складывать данные регистры, ведь они оба базовые. Сначала нужно суммировать их значения, а уже затем передавать информацию указателю из одного регистр. Сначала нужно передать сумму в регистр `bp`, а затем производить смещение

6. LAB2.ASM(73): error A2047: Multiple index registers:

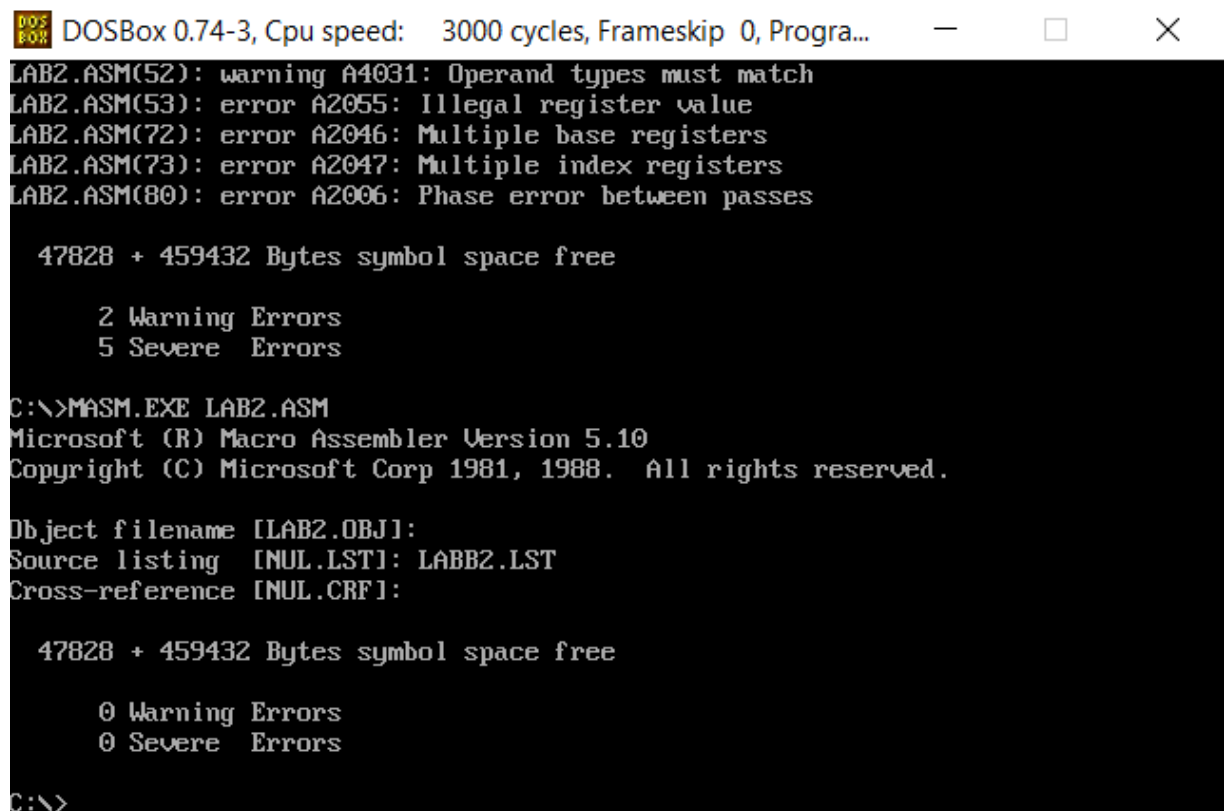
```
mov ax,matr[bp+di+si]
```

Аналогично прошлой ошибки только с регистрами индекса.

7. LAB2.ASM(80): error A2006: Phase error between passes Main ENDP

Данная ошибка сообщает нам о наличии ошибок в функции main.

3. Повторная трансляция программы и компоновка загрузочного модуля.



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
LAB2.ASM(52): warning A4031: Operand types must match
LAB2.ASM(53): error A2055: Illegal register value
LAB2.ASM(72): error A2046: Multiple base registers
LAB2.ASM(73): error A2047: Multiple index registers
LAB2.ASM(80): error A2006: Phase error between passes

47828 + 459432 Bytes symbol space free

2 Warning Errors
5 Severe Errors

C:\>MASM.EXE LAB2.ASM
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [LAB2.OBJ]:
Source listing [NUL.LST]: LABB2.LST
Cross-reference [NUL.CRF]:

47828 + 459432 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 команд

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 (Stack) = +0 0000 +2 0000 +4 0000	(SP)=0016 (IP)=0001 (Stack) = +0 19F5 +2 0000 +4 0000

			+6 0000	+6 0000
0001	SUB AX, AX	2BC0	(IP)=0001	(IP)=0003
0003	PUSH AX	50	(SP)= 0016 (IP)= 0003 (Stack) = +0 19F5 +2 0000 +4 0000 +6 0000	(SP)= 0014 (IP)= 0004 (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	(DS)= 19F5 (IP)= 0007	(DS)= 1A07 (IP)= 0009
0009	MOV AX, 01F4	B8F401	(AX)= 1A07 (IP)=0009	(AX)= 01F4 (IP)= 000C
000C	MOV CX, AX	8BC8	(IP)=000C (CX)=00B0	(IP)=000E (CX)=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)=010C (IP)=0020
0020	MOV AL, [BX+03]	8A4703	(IP) = 0020 (AX) = 010C	(IP)= 0023 (AX) = 0109

0023	MOV CX, [BX+03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 0509 (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [DI+000E]	8A850E00	(AX) = 0109 (IP) = 0029	(AX) = 0128 (IP) = 002D
002D	MOV BX, 0003	BB03000	(IP) = 002D (BX) = 0006	(IP) = 0030 (BX) = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	(IP) = 0030 (AX) = 0128	(IP) = 0034 (AX) = 01F9
0034	MOV AX, 1A07	B8071A	(AX) = 01F9 (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	(IP) = 0042
			(SP) = 0014 (Stack) = +0 0000 +2 19F5 +4 0000 +6 0000	(SP) = 0012 (Stack) = +0 1A07 +2 0000 +4 19F5 +6 0000

0042	POP ES	07	(SP)= 0012 (ES)= 0000 (IP)= 0042 (Stack) = +0 1A07 +2 0000 +4 19F5 +6 0000	(SP) = 0014 (ES)= 1A07 (IP)= 0043 (Stack) = + 0000 +2 19F5 +4 0000 +6 0000
0043	MOV CX, ES:[BX— 01]	268B4FFF	(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	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) = + 0000 +2 19F5	(IP) = 0054 (SP)=0012 (Stack) = + 01F4 +2 0000



			+4 0000	+4 19F5
			+6 0000	+6 0000
0054	PUSH [0002]	FF360200	(IP) = 0054 (SP) = 0012 (Stack) = + 01F4 +2 0000 +4 19F5 +6 0000	(IP) = 0058 (SP) = 0010 (Stack) = + FFCE +2 01F4 +4 0000 +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	CA0200	(IP) = 005D (SP) = 0010 (CS)=1A0A (Stack) = + FFCE +2 01F4 +4 0000 +6 19F5	(IP) = FFCE (SP)= 0016 (CS)=01F4 (Stack) = +0 19F5 +2 0000 +4 0000 +6 0000

### Выводы.

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

## Приложение А. Код программы lr2.asm

```
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 28,27,26,25,21,22,23,24
vec2 DB 20,30,-20,-30,40,50,-40,-50
matr DB -8,-7,3,4,-6,-5,1,2,-4,-3,7,8,-2,-1,5,6

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 успешной трансляции  
программы с закомментированными ошибочными  
операторами**

Microsoft (R) Macro Assembler Version 5.10

10/6/22 13:18:05

Page 1-1

```
= 0024          EOL EQU '$'  
= 0002          ind EQU 2
```

```

= 01F4                                n1 EQU 500
=-0032                                n2 EQU -50

; PŸC,PµPe PiCᵀPsPiCᵀP°PjPjC<

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

0018                                AStack ENDS

; P°P°PSPSC<Pµ PiCᵀPsPiCᵀP°PjPjC<

0000                                DATA SEGMENT

; P°PᵀCᵀPµPeC,PḂPIC< PsPiPḂCÍP°PSPḂCµ PrP°PSPSC
<C...

0000 0000                            mem1 DW 0
0002 0000                            mem2 DW 0
0004 0000                            mem3 DW 0
0006 1C 1B 1A 19 15 16              vec1 DB 28,27,26,25,21,22,23,24
    17 18
000E 14 1E EC E2 28 32              vec2 DB 20,30,-20,-30,40,50,-40,-50
    D8 CE
0016 F8 F9 03 04 FA FB              matr DB -8,-7,3,4,-6,-5,1,2,-4,-3,7,8,-2,-1,5,6
    01 02 FC FD 07 08
    FE FF 05 06

0026                                DATA ENDS

; PµPsPr PiCᵀPsPiCᵀP°PjPjC<

0000                                CODE SEGMENT

    ASSUME CS:CODE, DS:DATA, SS:AStack

; P°PsP»PsPIPSP°Cµ PiCᵀPsC†PµPrC†CᵀP°

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

; PµP PhP'P•P PµPh P P•P-PḂPḂPhP' PhP°P P•PŸPhP
|PḂPḂ PḂPh PJP PhP'PḂP• PŸPhP•PḂP•PḂPḂP™
; P PµPiPḂCÍC,CᵀPsPIP°Cµ P°PrCᵀPµCÍP°C†PḂCµ

```

```

0009  B8 01F4                mov ax,n1
000C  8B C8                  mov cx,ax
000E  B3 24                    mov bl,EOL
0010  B7 CE                    mov bh,n2

                                ; PµCṪCṪPjP°CṪ P°PrCṪPµCṪP°C†PḔCṪ
0012  C7 06 0002 R FFCE      mov mem2,n2
0018  BB 0006 R              mov bx,OFFSET vec1
001B  A3 0000 R              mov mem1,ax

                                ; PṼPsCṪPIPµPSPSP°CṪ P°PrCṪPµCṪP°C†PḔCṪ
001E  8A 07                  mov al,[bx]

                                ; mov mem3,[bx]
                                ; P`P°P·PḔCṪPsPIP°PSPSP°CṪ P°PrCṪPµCṪP°C†PḔCṪ
0020  8A 47 03              mov al,[bx]+3
0023  8B 4F 03              mov cx,3[bx]

                                ; PṼPSPPrPµPeCṪPSP°CṪ P°PrCṪPµCṪP°C†PḔCṪ

```

```

0026 BF 0002          mov di,ind
0029 8A 85 000E R      mov al,vec2[di]
                      ; mov cx,vec2[di]
                      ; PĥPrCĤPμCÍP°C†PěCĤ CÍ P±P°P•PěCĤPsPIP°PSPěPμP
j Pě PěPSPPrPμPeCÍPěCĤPsPIP°PSPěPμPj

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]
                      ; PμP PĥP'P•P PĥPĥ P P•P-PĥPĥPĥP' PĥP"P P•PŸPĥP
|PĥPĥ PŸ PJPSP•PŸPĥPĥ PŸP•P"PĥP•PĶPŸPĥP'
                      ; PμPμCĤPμPsPİCĤPμPrPμP»PμPSPěPμ CÍPμPİPjPμPSC,
P°
                      ; ----- PIP°CĤPěP°PSC, 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
                      ; ----- PIP°CĤPěP°PSC, 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
                      ; ----- PIP°CĤPěP°PSC, 3

0048 BF 0002          mov di,ind
004B 26: 89 01         mov es:[bx+di],ax
                      ; ----- PIP°CĤPěP°PSC, 4

004E 8B EC            mov bp,sp
                      ; mov ax,matr[bp+bx]
                      ; mov ax,matr[bp+di+si]
                      ; PĥCÍPİPsP»CĥP•PsPIP°PSPěPμ CÍPμPİPjPμPSC,P° C
ÍC,PμPeP°

```

```
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
```



## 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	LAB2	
@VERSION . . . . .	TEXT	510	

82 Source Lines

82 Total Lines

19 Symbols

47828 + 459432 Bytes symbol space free

0 Warning Errors

0 Severe Errors