

**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. 0382

\_\_\_\_\_

Куликов М.Д.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2021

### **Цель работы.**

Изучение режимов адресации процессора IntelX86 в процессе отладки программы.

### **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.

### **Выполнение работы.**

Была протранслирована программа `lab2.asm`, содержащая ошибки, был создан диагностический файл `lab2.lst`.

Было найдено 6 ошибок:

1. В строке `«mov mem3,[bx]»` - Improper operand type.

Вызвано попыткой перемещения из памяти в память, что запрещено.

2. В строке `«mov cx,vec2[di]»` - Operand typer must match.

Вызвано попыткой перенести 1-байтовый элемент массива в 2-байтовый регистр.

3. В строке «mov cx,matr[bx][di]» - Operand typer must match.

Вызвано попыткой перемещения 1-байтовый элемент массива в 2-байтовый регистр.

4. В строке «mov ax,matr[bx\*4][di]» - Illegal register value.

Вызвано тем, что 2-байтовые регистры запрещено умножать.

5. В строке «mov ax,matr[bp+bx]» - Mutiple base registers.

Вызвано тем, что запрещено использовать более 1 базового регистра для адресации.

6. «В строке mov ax,matr[bp+di+si]» - Multiple index registers.

Вызвано тем, что запрещено использовать более 1 индексного регистра для адресации.

Протокол отладки программы предоставлен в таблице 1.

Начальные сегментных регистров:

CS = 1A0A

DS = 19F5

SS = 1A05

ES = 19F5

Адрес команды	Символический код команды	16- ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	IP = 0000 SP = 0018 STACK + 0 0000 + 2 0000	IP = 0001 SP = 0016 STACK + 0 19F5 +2 0000
0001	SUB AX,AX	2BC0	IP = 0001 AX = 0000	IP = 0003 AX = 0000

0003	PUSH AX	50	IP = 0003	IP = 0004
0004	MOV AX,1A07	B8071A	IP = 0004 AX = 0000	IP = 0007 AX = 1A07
0007	MOV DS,AX	8ED8	IP = 0007 DS = 19F5	IP = 0009 DS = 1A07
0009	MOV AX,01F4	B8F401	IP = 0009 AX = 1A07	IP = 000C AX = 01F4
000C	MOV CX,AX	8BC8	IP = 000C CX = 00B0	IP = 000E CS = 01F4
000E	MOV BL,24	B324	IP = 000E BL = 00	IP = 0010 BL = 24
0010	MOV BH,CE	B7CE	IP = 0010 BH = 00	IP = 0012 BH = CE
0012	MOV [0002],FFCE	C7060200 CEFF	IP = 0012 DS: 0000 00 00 00 00 00 00	IP = 0018 DS: 0000 00 00 CE FF 00 00
0018	MOV BX,0006	BB0600	IP = 0018 BX = CE24	IP = 001B BX = 0006
001B	MOV [0000],AX	A30000	IP = 001B DS: 0000 00 00 CE FF 00 00	IP = 001E DS: 0000 F4 01 CE FF 00 00
001E	MOV AL,[BX]	8A07	IP = 001E AL = F4	IP = 0020 AL = 08
0020	MOV AL,[BX+03]	8A4703	IP = 0020 AL = 08	IP = 0023 AL = 05
0023	MOV CX,[BX+03]	8B4F03	IP = 0023 CX = 01F4	IP = 0026 CX = 0105

0026	MOV DI,0002	BF0200	IP = 0026 DI = 0002	IP = 0029 DI = 0002
0029	MOV AL,[000E+DI]	8A850E00	IP = 0029 AL = 05	IP = 002D AL = 1E
002D	MOV BX,0003	BB0300	IP = 002D BX = 0006	IP = 0030 BX = 0003
0030	MOV AL,[0016+BX+DI]	8AB11600	IP = 0030 AL = 1E	IP = 0034 AL = 03
0034	MOV AX,1A07	B8071A	IP = 0034 AX = 0103	IP = 0037 AX = 1A07
0037	MOV ES,AX	8EC0	IP = 0037 ES = 19F5	IP = 0039 ES = 1A07
0039	MOV AX,ES:[BX]	268B07	IP = 0039 AX = 1A07	IP = 003C AX = 00FF
003C	MOV AX,0000	B80000	IP = 003C AX = 00FF	IP = 003F AX = 0000
003F	MOV ES,AX	8EC0	IP = 003F ES = 1A07	IP = 0041 ES = 0000
0041	PUSH DS	1E	IP = 0041 SP = 0014 STACK +0 0000 +2 19F5 +4 0000	IP = 0042 SP = 0012 STACK +0 1A07 +2 0000 +4 19F5
0042	POP ES	07	IP = 0042 SP = 0012 ES = 0000 STACK +0 1A07 +2 0000	IP = 0043 SP = 0014 ES = 1A07 STACK +0 0000 +2 19F5

			+4 19F5	+4 0000
0043	MOV CX,ES:[BX-01]	268B4FFF	IP = 0043 CX = 0105	IP = 0047 CX = FFCE
0047	XCHG AX,CX	91	IP =0047 AX = 0000 CX = FFCE	IP = 0048 AX = FFCE CX = 0000
0048	MOV DI,0002	BF0200	IP = 0048 DI = 0002	IP = 004B DI = 0002
004B	MOV ES:[BX+DI],AX	268901	IP = 004B DS : 0000 F4 01 CE FF 00 00 00	IP = 004E DS : 0000 F4 01 CE FF 00 CE FF
004E	MOV BP,SP	8BEC	IP = 004E BP = 0000	IP = 0050 BP = 0014
0050	PUSH [0000]	FF360000	IP = 0050 SP = 0014 STACK +0 0000 +2 19F5 +4 0000	IP = 0054 SP = 0012 STACK +0 01F4 +2 0000 +4 19F5
0054	PUSH [0002]	FF360200	IP = 0054 SP = 0012 STACK +0 01F4 +2 0000 +4 19F5 +6 0000	IP = 0058 SP = 0010 STACK +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	MOV BP,SP	8BEC	IP = 0058 BP = 0014	IP = 005A BP = 0010

005A	MOV DX,[SP+02]	8B5602	IP = 005A DX = 0000	IP = 005D DX = 01F4
005D	RET FAR 0002	CA0200	IP = 005D SP = 0010 STACK +0 FFCE +2 01F4 +4 0000 +6 19F5	IP = FFCE SP = 0016 STACK +0 0000 +2 19F5

### **Выводы.**

В ходе лабораторной работы была изучена работа режимов адресации процессора IntelX86.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab2.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 -5,-6,-7,-8,4,3,2,1,-1,-2,-3,-4,8,7,6,5

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**  
**ДИАГНОСТИЧЕСКИЙ ФАЙЛ**

#MICROSOFT (R) MACRO ASSEMBLER VERSION 5.10

10/30/21

22:20:4

Page 1-1

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

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

0018	AStack ENDS
------	-------------

0000	DATA SEGMENT
------	--------------

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 FB FA F9 F8 04 03    matr DB -5,-6,-7,-8,4,3,2,1,-1,-2,-3,-4,8,7,6,5

02 01 FF FE FD FC
08 07 06 05
0026                      DATA ENDS

0000                      CODE SEGMENT
                        ASSUME CS:CODE, DS:DATA, SS:AStack

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

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

0012 C7 06 0002 R FFCE          mov mem2,n2

```

```
0018 BB 0006 R      mov bx,OFFSET vec1
001B A3 0000 R      mov mem1,ax

001E 8A 07          mov al,[bx]
                   ;mov mem3,[bx]

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

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

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]

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

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
0048 BF 0002	mov di,ind
004B 26: 89 01	mov es:[bx+di],ax
004E 8B EC	mov bp,sp
	;mov ax,matr[bp+bx]
	;mov ax,matr[bp+di+si]
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

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

88 Source Lines

88 Total Lines

19 Symbols

47812 + 459448 Bytes symbol space free

0 Warning Errors

0 Severe Errors