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. 9382 _____ Кузьмин Д. И.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург
2020

Цель работы.

Изучить режимы адресации в языке ассемблер.

Основные теоретические положения.

Выполнение работы производилось при помощи эмулятора операционной системы MS-DOS, DosBox. Для отладки программы использовался инструмент afdpro.

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) Описание ошибок:

mov mem3, [bx] – некорректно, потому что операндами инструкции mov не могут быть два участка памяти.

mov cx,vec2[di] - Размер элементов массива ‘vec2’ 1 байт, а ‘cx’ - 2 байта

mov cx,matr[bx][di] - Размер элементов матрицы ‘matr' 1 байт, а ‘cx' - 2 байта

mov ax,matr[bx*4][di] —некорректно, потому что в режиме индексной адресации с масштабированием нельзя использоваться регистр bx

mov ax,matr[bp+bx] — некорректно, потому что база не может определяться двумя регистрами

mov ax,matr[bp+di+si] — некорректно, потому что индекс не может определяться тремя регистрами

push mem1, push mem2 – программа не завершится, если в вершине стека не содержится соответствующая команда(int 20) (в данном случае там содержится mem2, поэтому после выполнения ret 2 программа не завершится)

2) Операторы, содержащие ошибки, были закомментированы.
3) Затем программа была протранслирована с создание файла листинга.
4) Далее был скомпанован загрузочный модуль
5) Затем с помощью отладчика afdpro было реализовано выполнение программы по шагам. Каждый шаг указан в табл.1

Исходный код файлов main.asm и main.lst представлен в приложении А.
Результат отладки main.exe показаны в табл. 1

Табл. 1 отладка файла main.exe
Начальное содержимое сегментных регистров
(CS) = 1A05, (ES) = 19F5, (DS) = 19F5, (SS) = 1A05

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(SP) = 0018 (IP) = 0000	(SP) = 0016 (IP) = 0001
0001	SUB AX, AX	2BC0	(IP) = 0001	(IP) = 0003
0003	PUSH AX	50	(SP) = 0016 (IP) = 0003	(SP) = 0014 (IP) = 0004
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	(CX) = 00B8 (IP) = 000C	(CX) = 01F4 (IP) = 000E
000E	MOV BL, 24	B324	(BL) = 00 (IP) = 000E	(BL) = 24 (IP) = 0010
0010	MOV BH, CE	B7CE	(BH) = 00 (IP) = 0010	(BH) = CE (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	(AL) = F4 (IP) = 001B	(AL) = 05 (IP) = 0020
0020	MOV AL, [BX + 03]	8A4703	(AL) = 05 (IP) = 0020	(AL) = 08 (IP) = 0023
0023	MOV CX, [BX + 03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 0C08 (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [000E + DI]	8A850E00	(AL) = 08 (IP) = 0029	(AL) = 14 (IP) = 002D
002D	MOV BX, 0003	BB0300	(BX) = 0006 (IP) = 002D	(BX) = 0003 (IP) = 0030
0030	MOV AL, [0016 + BX + DI]	8A811600	(AL) = 14 (IP) = 0030	(AL) = 03 (IP) = 0034
0034	MOV AX, 1A07	B8071A	(AX) = 0103 (IP) = 0034	(AX) = 1A07 (IP) = 0037
0037	MOV ES, AX	8EC0	(ES) = 19F5 (IP) = 0037	(ES) = 1A07 (IP) = 0029
0039	MOV AX, ES: [BX]	268B07	(AX) = 1A07 (IP) = 0039	(AX) = 00FF (IP) = 003C
003C	MOV AX, 0000	B80000	(AX) = 1A07 (IP) = 003C	(AX) = 0000 (IP) = 003F
003F	MOV ES, AX	8EC0	(ES) = 1A07 (IP) = 003F	(ES) = 0000 (IP) = 0041
0041	PUSH DS	1E	(SP) = 0014 (IP) = 0041	(SP) = 0012 (IP) = 0042

0042	POP ES	07	(SP) = 0012 (IP) = 0042	(SP) = 0014 (IP) = 0043
0043	MOV CX, ES: [BX – 01]	268B4FFF	(CX) = 0203 (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	(DI) = 0002 (IP) = 0048	(DI) = 0002 (IP) = 004B
004B	MOV ES:[BX + DI], AX	268901	(ES) = 1A07 (IP) = 004B	(ES) = 1A07 (IP) = 0056
004E	MOV BP, SP	8BEC	(BP) = 0000 (IP) = 004E	(BP) = 0014 (IP) = 0050
0050	MOV BP, SP	8BEC	(BP) = 0000 (IP) = 0050	(BP) = 0014 (IP) = 0052
0052	MOV DX, [BP + 02]	8B5602	(DX) = 0000 (IP) = 0052	(DX) = 19F5 (IP) = 0055
0055	RET FAR 0002	CA0200	(SP) = 0010	(SP) = 0016
0000	CD20	INT 20		

Выводы.

Были изучены различные режимы адресации языка ассемблер.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

Файл main.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 5,6,7,8,12,11,10,9
vec2 DB -20,-30,20,30,-40,-50,40,50
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
```

Файл main.lst

```
#Microsoft (R) Macro Assembler Version 5.10                 10/22/20
07:45:5
                                                            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?
                        ?х
 0000  0000                   mem1 DW 0
 0002  0000                   mem2 DW 0
 0004  0000                   mem3 DW 0
 0006  05 06 07 08 0C 0B     vec1 DB 5,6,7,8,12,11,10,9
       0A 09
 000E  EC E2 14 1E D8 CE     vec2 DB -20,-30,20,30,-40,-50,40,50
       28 32
 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
                        ; 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
```

```
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                mov 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
                       ;push mem1
                       ;push mem2
0050  8B EC                mov bp,sp
0052  8B 56 02             mov dx,[bp]+2
0055  CA 0002              ret 2
0058                  Main ENDP
0058                  CODE ENDS
                       END Main
```

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	0058	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 = 0058
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	main		
@VERSION	TEXT	510		

```
     83 Source  Lines
     83 Total   Lines
     19 Symbols

  47828 + 459432 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```