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. 0383 _____ Смирнов И.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучение режимов адресации и формирования 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.

Вар. 5

vec1 11,12,13,14,18,17,16,15

vec2 10,20,-10,-20,30,40,-30,-40

matr 1,2,-4,-3,3,4,-2,-1,5,6,-8,-7,7,8,-6,-5

1. Попытка протранслировать программу lab2.asm привела к возникновению следующих ошибок:
    1. mov mem3,[bx], lab2.asm(42): error A2052: Improper operand type. Нельзя перемещать данные из одной ячейки памяти в другую (только между регистрами или между регистром и ячейкой памяти).

2. mov cx,vec2[di], lab2.asm(49): warning A4031: Operand types must match. Попытка поместить данные размером в 1 байт а регистр размером 2 байта.
3. mov cx,matr[bx][di], lab2.asm(53): warning A4031: Operand types must match. Попытка поместить данные размером в 1 байт а регистр размером 2 байта.
4. mov ax,matr[bx*4][di], lab2.asm(54): error A2055: Illegal register value. Недопустимое значение регистра.
5. mov ax,matr[bp+bx], lab2.asm(73): error A2047: Multiple base registers. Попытка использовать несколько базовых регистров для адресации.
6. mov ax,matr[bp+di+si], lab2.asm(74): error A2047: Multiple index registers. Попытка использовать несколько индексных регистров для адресации.

2. После того, как строки с ошибками были закомментированы, файл был протранслирован без ошибок и предупреждений. Был создан диагностический файл lab2.lst и объектный файл lab2.obj. Был собран lab2.exe и запущен в отладчике.

   (CS) = 1A0A

   (DS) = 19F5

   (ES) = 19F5

   (SS) = 1A05

   (CX) = 00B0

   (BP) = 0000

   (DX) = 0000

Табл. 1

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(AX) = 0000 (DS) = 19F5 (IP) = 0000 (SP) = 0018 Stack +0 0000	(AX) = 0000 (DS) = 19F5 (IP) = 0001 (SP) = 0016 Stack +0 19F5
0001	SUB AX, AX	2BC0	(AX) = 0000 (DS) = 19F5 (IP) = 0001	(AX) = 0000 (DS) = 19F5 (IP) = 0003
0003	PUSH AX	50	(AX) = 0000 (DS) = 19F5 (IP) = 0003 (SP) = 0016 Stack +0 19F5 +2 0000	(AX) = 0000 (DS) = 19F5 (IP) = 0004 (SP) = 0014 Stack +0 0000 +0 19F5
0004	MOV AX, 1A07	B8071A	(AX) = 0000 (DS) = 19F5 (IP) = 0004	(AX) = 1A07 (DS) = 19F5 (IP) = 0007
0007	MOV DS, AX	8ED8	(AX) = 1A07 (DS) = 19F5 (IP) = 0007	(AX) = 1A07 (DS) = 1A07 (IP) = 0009
0009	MOV AX, 01F4	B8F401	(AX) = 1A07 (DS) = 1A07 (IP) = 0009	(AX) = 01F4 (DS) = 1A07 (IP) = 000C
000C	MOV CX, AX	8BC8	(AX) = 01F4 (DS) = 1A07 (IP) = 000C (CX) = 00B0	(AX) = 01F4 (DS) = 1A07 (IP) = 000E (CX) = 01F4
000E	MOV BL, 24	B324	(AX) = 01F4 (DS) = 1A07 (IP) = 000E	(AX) = 01F4 (DS) = 1A07 (IP) = 0010

			(BX) = 0000	(BX) = 0024
0010	MOV BH, CE	B7CE	(AX) = 01F4 (DS) = 1A07 (IP) = 0010 (BX) = 0024	(AX) = 01F4 (DS) = 1A07 (IP) = 0012 (BX) = CE24
0012	MOV [0002], FFCE	C7060200CEF F	(AX) = 01F4 (DS) = 1A07 (IP) = 0012 Data seg +2 0000	(AX) = 01F4 (DS) = 1A07 (IP) = 0018 Data seg +2 CEFF
0018	MOV BX, 00076	BB0600	(AX) = 01F4 (DS) = 1A07 (BX) = CE24 (IP) = 0018	(AX) = 01F4 (DS) = 1A07 (BX) = 0006 (IP) = 001B
001B	MOV [0000], AX	A30000	(AX) = 01F4 (DS) = 1A07 (IP) = 001B Data seg +0 0000	(AX) = 01F4 (DS) = 1A07 (IP) = 001E Data seg +0 F401
001E	MOV AL, [BX]	8A07	(AX) = 01F4 (DS) = 1A07 (IP) = 001E	(AX) = 010B (DS) = 1A07 (IP) = 0020
0020	MOV AL, [BX+03]	8A4703	(AX) = 010B (DS) = 1A07 (IP) = 0020	(AX) = 010E (DS) = 1A07 (IP) = 0023
0023	MOV CX, [BX+03]	8B4F03	(AX) = 010E (DS) = 1A07 (IP) = 0023 (CX) = 01F4	(AX) = 010E (DS) = 1A07 (IP) = 0026 (CX) = 120E
0026	MOV DI, 0002	BF0200	(AX) = 010E (DS) = 1A07 (IP) = 0026 (DI) = 0000	(AX) = 010E (DS) = 1A07 (IP) = 0029 (DI) = 0002
0029	MOV AL, [000E+DI]	8A850E00	(AX) = 010E (DS) = 1A07 (IP) = 0029	(AX) = 01F6 (DS) = 1A07 (IP) = 002D

			(DI) = 0002	(DI) = 0002
002D	MOV BX, 0003	BB0300	(AX) = 01F6 (DS) = 1A07 (IP) = 002D (BX) = 0006	(AX) = 01F6 (DS) = 1A07 (IP) = 0030 (BX) = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	(AX) = 01F6 (DS) = 1A07 (IP) = 0030 (DI) = 0002 (BX) = 0003	(AX) = 0104 (DS) = 1A07 (IP) = 0034 (DI) = 0002 (BX) = 0003
0034	MOV AX, 1A07	B8071A	(AX) = 0104 (DS) = 1A07 (IP) = 0034	(AX) = 1A07 (DS) = 1A07 (IP) = 0037
0037	MOV ES, AX	8EC0	(AX) = 1A07 (DS) = 1A07 (IP) = 0037 (ES) = 19F5	(AX) = 1A07 (DS) = 1A07 (IP) = 0039 (ES) = 1A07
0039	MOV AX, ES:[BX]	268B07	(AX) = 1A07 (DS) = 1A07 (IP) = 0039 (ES) = 1A07	(AX) = 00FF (DS) = 1A07 (IP) = 003C (ES) = 1A07
003C	MOV AX, 0000	B80000	(AX) = 00FF (DS) = 1A07 (IP) = 003C	(AX) = 0000 (DS) = 1A07 (IP) = 003F
003F	MOV ES, AX	BEC0	(AX) = 0000 (DS) = 1A07 (IP) = 003F (ES) = 1A07	(AX) = 0000 (DS) = 1A07 (IP) = 0041 (ES) = 0000
0041	PUSH DS	1E	(AX) = 0000 (DS) = 1A07 (IP) = 0041 (SP) = 0014 Stack +0 0000	(AX) = 0000 (DS) = 1A07 (IP) = 0042 (SP) = 0012 Stack +0 1A07
0042	POP ES	07	(AX) = 0000	(AX) = 0000

			(DS) = 1A07 (IP) = 0042 (SP) = 0012 (ES) = 0000 Stack +0 1A07 +0 0000	(DS) = 1A07 (IP) = 0043 (SP) = 0014 (ES) = 1A07 Stack +0 0000 +2 19F5
0043	MOV CX, ES:[BX-01]	268B4FFF	(AX) = 0000 (DS) = 1A07 (IP) = 0043 (ES) = 1A07 (BX) = 0003 (CX) = 120E	(AX) = 0000 (DS) = 1A07 (IP) = 0047 (ES) = 1A07 (BX) = 0003 (CX) = FFCE
0047	XCHG AX, CX	91	(AX) = 0000 (DS) = 1A07 (IP) = 0047 (CX) = FFCE	(AX) = FFCE (DS) = 1A07 (IP) = 0048 (DI) = 0002
0048	MOV DI, 0002	BF0200	(AX) = FFCE (DS) = 1A07 (IP) = 0048 (DI) = 0002	(AX) = FFCE (DS) = 1A07 (IP) = 004B (DI) = 0002
004B	MOV ES:[BX+DI], AX	268901	(AX) = FFCE (DS) = 1A07 (IP) = 004B (DI) = 0002 (BX) = 0003 (ES) = 1A07	(AX) = FFCE (DS) = 1A07 (IP) = 004E (DI) = 0002 (BX) = 0003 (ES) = 1A07
004E	MOV BP, SP	8BEC	(AX) = FFCE (DS) = 1A07 (IP) = 004E (BP) = 0000 (SP) = 0014	(AX) = FFCE (DS) = 1A07 (IP) = 0050 (BP) = 0014 (SP) = 0014
0050	PUSH [0000]	FF360000	(AX) = FFCE (DS) = 1A07 (IP) = 0050	(AX) = FFCE (DS) = 1A07 (IP) = 0054

			(SP) = 0014 Stack +0 0000 +2 19F5 +4 0000	(SP) = 0012 Stack +0 01F4 +2 0000 +4 19F5
0054	PUSH [0002]	FF360200	(AX) = FFCE (DS) = 1A07 (IP) = 0054 (SP) = 0012 Stack +0 01F4 +2 0000 +4 19F5 +6 0000	(AX) = FFCE (DS) = 1A07 (IP) = 0058 (SP) = 0010 Stack +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	MOV BP, SP	8BEC	(AX) = FFCE (DS) = 1A07 (IP) = 0058 (SP) = 0010 (BP) = 0014	(AX) = FFCE (DS) = 1A07 (IP) = 005A (SP) = 0010 (BP) = 0010
005A	MOV DX, [BP+02]	8B5602	(AX) = FFCE (DS) = 1A07 (IP) = 005A (SP) = 0010 (BP) = 0010 (DX) = 0000	(AX) = FFCE (DS) = 1A07 (IP) = 005D (SP) = 0010 (BP) = 0010 (DX) = 01F4
005D	RET Far 0002	CA0200	(AX) = FFCE (DS) = 1A07 (IP) = 005D (SP) = 0010 Stack +0 FFCE +2 01F4	(AX) = FFCE (DS) = 1A07 (IP) = FFCE (SP) = 0016 Stack +0 19F5 +2 0000

Программа не завершила работу, так как на стек были записаны ненужные данные. Чтобы исправить эту ошибку, нужно закомментировать две строки, которые добавляют эти значения на стек.

Выводы.

По ходу выполнения данной лабораторной работы мы изучили режимы адресации и формирование исполнительного адреса.

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 11,12,13,14,18,17,16,15
vec2 DB 10,20,-10,-20,30,40,-30,-40
matr DB 1,2,-4,-3,3,4,-2,-1,5,6,-8,-7,7,8,-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

ФАЙЛ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ ПРОГРАММЫ

Название файла: lab2.lst

```
Microsoft (R) Macro Assembler Version 5.10                  10/13/21
14:50:3
                                                             Page     1-1


                        ; Программа изучения режимов адресации процессо
                        ра 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  0B 0C 0D 0E 12 11     vec1 DB 11,12,13,14,18,17,16,15
       10 0F
 000E  0A 14 F6 EC 1E 28     vec2 DB 10,20,-10,-20,30,40,-30,-40
       E2 D8
 0016  01 02 FC FD 03 04     matr DB 1,2,-4,-3,3,4,-2,-1,5,6,-8,-7,7,8,-6,-
5
       FE FF 05 06 F8 F9
       07 08 FA FB
 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 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

Microsoft (R) Macro Assembler Version 5.10                  10/13/21
14:50:3
                                                             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 СЕГМЕНТОВ
                      ; 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                  10/13/21
14:50:3
                                                             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  lab2
@VERSION . . . . . . . . . . .      TEXT  510


     83 Source  Lines
     83 Total   Lines
     19 Symbols

  47828 + 461479 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```