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 6

Студентка гр. 1383 _____ Самулевич С.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Изучение режимов адресации на языке Ассемблера.

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.Были изменены значения vec1, vec2, matr согласно варианту.

2.Были обнаружены следующие ошибки:

1) mov mem3,[bx] (lab_2.asm(42): error A2052: Improper operand type) Неподходящий тип операнда. Невозможно одновременное считывание из памяти и запись данных в память.

2) mov cx,vec2[di] (lab_2.asm(49): warning A4031: Operand types must match) Несоответствие типов операндов. Операнды должны быть одинакового размера. Размерность регистра ‘cx’ - 2 байта, а ‘vec2’ 1 байт.

3) mov cx,matr[bx][di](lab_2.asm(53): warning A4031: Operand types must match) Несоответствие типов операндов. Размер элементов операнда ‘matr' 1 байт, а ‘cx' - 2 байта

4) mov ax,matr[bx*4][di](lab_2.asm(54): error A2055: Illegal register value) Незаконное использование регистра. Нельзя масштабировать 16-битные регистры.

5) mov ax,matr[bp+bx](lab_2.asm(73): error A2046: Multiple base registers) Слишком много базовых регистров. Нельзя использовать более одного базового регистра.

6) mov ax,matr[bp+di+si](lab_2.asm(74): error A2047: Multiple index registers) Слишком много индексных регистров. Нельзя использовать более одного индексного регистра.

3.Строки с ошибками были закомментированы.

4.Начальное (содержимое сегментных регистров) состояние режимов (CS) = 1A0A, (DS) = 19F5, (ES) = 19F5, (SS) = 1A05.

Таблица 2- исполнение файла LR2.asm

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	Stack: +0 0000 +2 0000 +4 0000 +6 0000 (SP) = 0018 (IP) = 0000	Stack: +0 19F5 +2 0000 +4 0000 +6 0000 (SP) = 0016 (IP) = 0001
0001	SUB AX, AX	2BC0	(AX) = 0000	(AX) = 0000

			(IP) = 0001	(IP) = 0003
0003	PUSH AX	50	Stack: +0 19F5 +2 0000 +4 0000 +6 0000 (SP) = 0016 (IP) = 0003	Stack: +0 0000 +2 19F5 +4 0000 +6 0000 (SP) = 0014 (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	(CX) = 00B0 (IP) = 000C	(CX) = 01F4 (IP) = 000E
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) = 0112 (IP) = 0020
0020	MOV AL, [BX	8A4703	(AX) = 011F	(AX) = 010F

	+ 03]		(IP) = 0020	(IP) = 0023
0023	MOV CX, [BX, + 03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 0B0F (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [000E+ DI]	8A850E00	(AX) = 010F (IP) = 0029	(AX) = 01E2 (IP) = 002D
002D	MOV CX, [000E+DI]	8B8D0E00	(CX) = 0B0F (IP) = 2D	(CX) = D8E2 (IP) = 31
0031	MOV BX, 0003	BB0300	(BX) = 0006 (IP) = 0031	(BX) = 0003 (IP) = 0034
0034	MOV AL, [0016 + BX + DI]	8A811600	(AX) = 01E2 (IP) = 0034	(AX) = 01FF (IP) = 0038
0038	MOV CX, [0016+BX+DI]	8B891600	(CX) = D8E2 (IP) = 0038	(CX) = 03FF (IP) = 003C
003C	MOV AL, [BX+03]	8A4703	(AX) = 01FF (IP) = 003C	(AX) = 0112 (IP) = 003F
003F	MOV CX, [BX+03]	8B4F03	(CX) = 03FF (IP) = 003F	(CX) = 1112 (IP) = 0042
0042	MOV AX, 1A07	B8071A	(AX) = 0112 (IP) = 0042	(AX) =1A07 (IP) = 0045
0045	MOV ES, AX	8EC0	(ES) = 19F5 (IP) = 0045	(ES) = 1A07 (IP) = 0047
0047	MOV AX, ES: [BX]	268B07	(AX) =1A07 (IP) = 0047	(AX) =00FF (IP) = 004A
004A	MOV AX, 0000	B80000	(AX) =00FF (IP) = 004A	(AX) =0000 (IP) = 004D
004D	MOV ES, AX	8EC0	(IP) = 004D (ES) = 1A07	(IP) = 004F (ES) = 0000

004F	PUSH DS	1E	Stack: +0 0000 +2 19F5 +4 0000 +6 0000 (SP) = 0014 (IP) = 004F	Stack: +0 1A07 +2 0000 +4 19F5 +6 0000 (SP) = 0012 (IP) = 0050
0050	POP ES	07	Stack: +0 1A07 +2 0000 +4 19F5 +6 0000 (SP) = 0012 (ES) = 0000 (IP) = 0050	Stack: +0 0000 +2 19F5 +4 0000 +6 0000 (SP) = 0014 (ES) = 1A07 (IP) = 0051
0051	MOV CX, ES: [BX - 01]	268B4FFF	(IP) = 0051 (CX) = 1112	(IP) = 0055 (CX) = FFCE
0055	XCHG AX, CX	91	(AX) = 0000 (IP) = 0055 (CX) = FFCE	(AX) = FFCE (IP) = 0056 (CX) = 0000
0056	MOV DI, 0002	BF0200	(IP) = 0056	(IP) = 0059
0059	MOV ES: [BX + DI], AX	268901	(IP) = 0059	(IP) = 005C
005C	MOV BP, SP	8BEC	(IP) = 005C (BP) = 0000	(IP) = 005E (BP) = 0014

005E	PUSH [0000]	FF360000	Stack: +0 0000 +2 19F5 +4 0000 +6 0000 (IP) = 005E (SP) = 0014	Stack: +0 01F4 +2 0000 +4 19F5 +6 0000 (IP) = 0062 (SP) = 0012
0062	PUSH [0002]	FF360200	Stack: +0 01F4 +2 0000 +4 19F5 +6 0000 (IP) = 0062 (SP) = 0012	Stack: +0 FFCE +2 01F4 +4 0000 +6 19F5 (IP) = 0066 (SP) = 0010
0066	MOV BP, SP	8BEC	(BP) = 0014 (IP) = 0066	(BP) = 0010 (IP) = 0068
0068	MOV DX, [BP + 02]	8B5602	(DX) = 0000 (IP) = 0068	(DX) = 01F4 (IP) = 006B
006B	RET Far 0002	CA0200	Stack: +0 FFCE +2 01F4 +4 0000 +6 19F5 (SP) = 0010 (CS) = 1A0A	Stack: +0 19F5 +2 0000 +4 0000 +6 0000 (SP) = 0016 (CS) = 01F4

			(IP) = 006B	(IP) = FFCE

Программный код см. в приложении А.

Файлы диагностических сообщений см. в приложении Б.

Выводы.

В ходе выполнения работы были изучены режимы адресации на языке Ассемблера.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: LR2.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 18,17,16,15,11,12,13,14
vec2 DB 30,40,-30,-40,10,20,-10,-20
matr DB -4,-3,1,2,-2,-1,3,4,5,6,7,8,-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]
 mov al,[bx]+3
 mov cx,3[bx]
; 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
ФАЙЛЫ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: LR2.lst

```
Microsoft (R) Macro Assembler Version 5.10
11/26/22 20:42:1
Page 1-1


1 ; РµСЂРsРiСЂР°РjРjР° РёР·СѓС‡РµРSРёСЏ С
ЂРµР¶РёРjРsРI Р°РґСЂРµСЃР°С†РёРё РїСЂРs
С†РµСЃСѓРsСЂР° IntelX86
2 = 0024 EOL EQU '$'
3 = 0002 ind EQU 2
4 = 01F4 n1 EQU 500
5 =-0032 n2 EQU -50
6 ; РЎС,РµРє РїСЂРsРiСЂР°РjРjС‹
7 0000 AStack SEGMENT STACK
8 0000 000C[ DW 12 DUP(?)
9 ????
10 ]
11
12 0018 AStack ENDS
13 ; Р”Р°РSРSС‹ Рµ РїСЂРsРiСЂР°РjРjС‹
14 0000 DATA SEGMENT
15 ; Р”РёСЂРµРєС,РёРIС‹ РsРїРёСЃР°РSРёСЏ Р
ґР°РSРSС‹ С…
16 0000 0000 mem1 DW 0
17 0002 0000 mem2 DW 0
18 0004 0000 mem3 DW 0
19 0006 12 11 10 0F 0B 0C vec1 DB 18,17,16,15,11,12,13,14
20 0D 0E
21 000E 1E 28 E2 D8 0A 14 vec2 DB 30,40,-30,-40,10,20,-10,-20
22 F6 EC
23 0016 FC FD 01 02 FE FF matr DB -4,-3,1,2,-2,-1,3,4,5,6,7,8,-8,
-7,-6,-5
24 03 04 05 06 07 08
25 F8 F9 FA FB
26 0026 DATA ENDS
27 ; РъРsРґ РїСЂРsРiСЂР°РjРjС‹
28 0000 CODE SEGMENT
29 ASSUME CS:CODE, DS:DATA, SS:AStack
30 ; Р"РsР»РsРIРSР°СЏ РїСЂРsС†РµРґСѓСЂР°
```

```
31 0000                    Main PROC FAR
32 0000  1E                  push DS
33 0001  2B C0               sub AX,AX
34 0003  50                  push AX
35 0004  B8 ---- R           mov AX,DATA
36 0007  8E D8               mov DS,AX
37                  ; PµP PħP’P•P PљPђ P P•P–P□PњPħP’ PђP”P
P•PЎPђP¦P□P□ PќPђ PЈP PħP’PќP• PЎPњP•P
©P•PќP□P™
38                  ; P PµPiPëCЃC,CЂPsPIP°CЏ P°PґCЂPµCЃP°C†
PëCЏ
39 0009  B8 01F4             mov ax,n1
40 000C  8B C8               mov cx,ax
41 000E  B3 24               mov bl,EOL
42 0010  B7 CE               mov bh,n2
43                  ; PµCЂCЏPjP°CЏ P°PґCЂPµCЃP°C†PëCЏ
44 0012  C7 06 0002 R FFCE  mov mem2,n2
45 0018  BB 0006 R          mov bx,OFFSET vec1
46 001B  A3 0000 R          mov mem1,ax
47                  ; PљPsCЃPIPµPSPSP°CЏ P°PґCЂPµCЃP°C†PëCЏ
```

```
          48 001E  8A 07              mov al,[bx]
           49                       ;mov mem3,[bx]
50                    ; P‘P°P·PëCЪPsPIP°PSPSP°CЏ P°PґCЪPµCЃP°
                           C†PëCЏ
       51 0020  8A 47 03                mov al,[bx]+3
       52 0023  8B 4F 03                mov cx,3[bx]
53                    ; P□PSPґPµPєCЃPSP°CЏ P°PґCЪPµCЃP°C†PëCЏ
          54 0026  BF 0002             mov di,ind
        55 0029  8A 85 000E R       mov al,vec2[di]
        56 002D  8B 8D 000E R       mov cx,vec2[di]
    lr2.asm(49): warning A4031: Operand types must match
57                    ; PђPґCЪPµCЃP°C†PëCЏ CЃ P±P°P·PëCЪPsPIP
          °PSPëPµPj Pë PëPSPґPµPєCЃPëCЪPsPIP°PSPë
                           PµPj
           58 0031  BB 0003             mov bx,3
       59 0034  8A 81 0016 R       mov al,matr[bx][di]
       60 0038  8B 89 0016 R       mov cx,matr[bx][di]
    lr2.asm(53): warning A4031: Operand types must match
        61                     ;mov ax,matr[bx*4][di]
       62 003C  8A 47 03                mov al,[bx]+3
        63 003F  8B 4F 03                mov cx,3[bx]
64                    ; PµP PћP’P•P PъPђ P P•P–P□PњPћP’ PђP”P
          P•PЎPђP¦P□P□ PЎ PJP§P•PўPћPњ PЎP•P“PњP
                           •PќPўPћP’
65                    ; PµPµCЪPµPsPїCЪPµPґPµP»PµPSPëPµ CЃPµPi
                           PjPµPSC,P°
      66                    ; ------ PIP°CЪPëP°PSC, 1
        67 0042  B8 ---- R          mov ax, SEG vec2
           68 0045  8E C0               mov es, ax
         69 0047  26: 8B 07          mov ax, es:[bx]
           70 004A  B8 0000             mov ax, 0
       71                    ; ------ PIP°CЪPëP°PSC, 2
           72 004D  8E C0               mov es, ax
            73 004F  1E                 push ds
```

```
                74 0050  07                  pop es
               75 0051  26: 8B 4F FF        mov cx, es:[bx-1]
                   76 0055  91                   xchg cx,ax
              77                     ; ------ PIP°CЂPëP°PSC, 3
                   78 0056  BF 0002              mov di,ind
               79 0059  26: 89 01           mov es:[bx+di],ax
              80                     ; ------ PIP°CЂPëP°PSC, 4
                    81 005C  8B EC                mov bp,sp
                 82                       ;mov ax,matr[bp+bx]
               83                       ;mov ax,matr[bp+di+si]
84                    ; P□CЃPïPsP»CЊP·PsPIP°PSPëPµ CЃPµPiPjPµ
                       PSC,P° CЃC,PµPєP°
                    85 005E  FF 36 0000 R         push mem1
                    86 0062  FF 36 0002 R         push mem2
                    87 0066  8B EC                mov bp,sp
              88 0068  8B 56 02                    mov dx,[bp]+2
                  89 006B  CA 0002                ret 2
                 90 006E                        Main ENDP
                 91 006E                        CODE ENDS
                    92                        END Main
```

Microsoft (R) Macro Assembler Version 5.10
11/26/22 20:42:1
Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	006E	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 = 006E
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	lr2	
@VERSION	TEXT	510	

```
    85 Source  Lines
    85 Total   Lines
       19 Symbols

47316 + 459944 Bytes symbol space free

    2 Warning Errors
    0 Severe  Errors
```