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

Студентка гр. 1381		Демчук П. Д.
Преподаватель		Ефремов М. А.

Санкт-Петербург

2022

Цель работы.

Изучить режимы адресации и формирование 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.

1. Получен у преподавателя вариант набора значений исходных данных (массивов) vec1, vec2 и matr из файла lr2.dat, приведенного в каталоге Задания и занесены свои данные (вариант 5) вместо значений, указанных в приведенной ниже программе.

2. Протранслирована программа с созданием файла диагностических сообщений; объяснены обнаруженные ошибки и закомментированы соответствующие операторы в тексте программы:

1) LR2_comp.asm(49): error A2052: Improper operand type
   mov mem3,[bx]
   Нельзя читать из памяти и писать в память одной командой. Перемещать данные можно между регистрами либо регистром и памятью.
2) LR2_comp.asm(56): warning A4031: Operand types must match
   mov cx,vec2[di]
   Несовпадение размеров операндов (первый - 2 байта, второй - 1 байт).
3) LR2_comp.asm(60): warning A4031: Operand types must match
   mov cx,matr[bx][di]
   Несовпадение размеров операндов (первый - 2 байта, второй - 1 байт).
4) LR2_comp.asm(61): error A2055: Illegal register value
   mov ax,matr[bx*4][di]
   Нельзя масштабировать адрес на 086 наборе инструкций.
5) LR2_comp.asm(81): error A2046: Multiple base registers
   mov ax,matr[bp+bx]
   Нельзя использовать более одного базового регистра в одной команде.
6) LR2_comp.asm(82): error A2047: Multiple index registers
   mov ax,matr[bp+di+si]
   Нельзя использовать более одного индексного регистра в одной команде.
7) LR2_comp.asm(89): error A2006: Phase error between passes
   Main ENDP
   Ошибки в Main.

3. Снова протранслирована программа и скомпонован загрузочный модуль.

4. Выполнена программа 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:

Начальное содержание сегментных регистров: (CS)=1A0A, (DS)=19F5, (ES)=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 (DS) = 19F5 (IP) = 0000 Stack +0 0000	(SP) = 0016 (DS) = 19F5 (IP) = 0001 Stack +0 19F5
0001	SUB AX, AX	2BC0	(AX) = 0000 (IP) = 0001	(AX) = 0000 (IP) = 0003
0003	PUSH AX	50	(AX) = 0000 (SP) = 0016 (IP) = 0003 Stack +0 19F5 +2 0000	(AX) = 0000 (SP) = 0014 (IP) = 0004 Stack +0 0000 +2 19F5
0004	MOV AX, 1A07	B8071A	(AX) = 0000 (IP) = 0004	(AX) = 1A07 (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	(AX) = 01F4

			(IP) = 0009	(IP) = 000C
000C	MOV CX, AX	8BC8	(AX) = 01F4 (CX) = 00B0 (IP) = 000C	(AX) = 01F4 (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	DS:0002 = 00 DS:0003 = 00 (IP) = 0012	DS:0002 = CE DS:0003 = FF (IP) = 0018
0018	MOV BX, 0006	BB0600	(BX) = CE24 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	MOV [0000], AX	A30000	DS:0000 = 00 DS:0001 = 00 (AX) = 01F4 (IP) = 001B	DS:0000 = F4 DS:0001 = 01 (AX) = 01F4 (IP) = 001E
001E	MOV AL, [BX]	8A07	(AX) = 01F4 (BX) = 0006 (IP) = 001E	(AX) = 010B (BX) = 0006 (IP) = 0020
0020	MOV AL, [BX+03]	8A4703	(AX) = 010B (BX) = 0006 (IP) = 0020	(AX) = 010E (BX) = 0006 (IP) = 0023
0023	MOV CX, [BX+3]	8B4F03	(AX) = 010E (CX) = 01F4 (IP) = 0023	(AX) = 010E (CX) = 120E (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [000E+DI]	8A850E00	(AX) = 010E (DI) = 0002 (IP) = 0029	(AX) = 01F6 (DI) = 0002 (IP) = 002D

002D	MOV CX, [000E+DI]	8B8D0E00	(CX) = 120E (DI) = 0002 (IP) = 002D	(CX) = ECF6 (DI) = 0002 (IP) = 0031
0031	MOV BX, 0003	BB0300	(BX) = 0006 (IP) = 0031	(BX) = 0003 (IP) = 0034
0034	MOV AL, [0016+BX+DI]	8A811600	(AX) = 01F6 (BX) = 0003 (DI) = 0002 (IP) = 0034	(AX) = 0104 (BX) = 0003 (DI) = 0002 (IP) = 0038
0038	MOV CX, [0016+BX+DI]	8B891600	(CX) = ECF6 (BX) = 0003 (DI) = 0002 (IP) = 0038	(CX) = FE04 (BX) = 0003 (DI) = 0002 (IP) = 003C
003C	MOV AX, 1A07	B8071A	(AX) = 0104 (IP) = 003C	(AX) = 1A07 (IP) = 003F
003F	MOV ES, AX	8EC0	(AX) = 1A07 (ES) = 19F5 (IP) = 003F	(AX) = 1A07 (ES) = 1A07 (IP) = 0041
0041	MOV AX, ES:[BX]	268B07	(AX) = 1A07 (BX) = 0003 (ES) = 1A07 (IP) = 0041	(AX) = 00FF (BX) = 0003 (ES) = 1A07 (IP) = 0044
0044	MOV AX, 0000	B80000	(AX) = 00FF (IP) = 0044	(AX) = 0000 (IP) = 0047
0047	MOV ES, AX	8EC0	(AX) = 0000 (ES) = 1A07 (IP) = 0047	(AX) = 0000 (ES) = 0000 (IP) = 0049
0049	PUSH DS	1E	(SP) = 0014 (DS) = 1A07 (IP) = 0049 Stack +0 0000 +2 19F5 +4 0000	(SP) = 0012 (DS) = 1A07 (IP) = 004A Stack +0 1A07 +2 0000 +4 19F5
004A	POP ES	07	(SP) = 0012	(SP) = 0014

			(ES) = 0000 (IP) = 004A Stack +0 1A07 +2 0000 +4 19F5	(ES) = 1A07 (IP) = 004B Stack +0 0000 +2 19F5 +4 0000
004B	MOV CX, ES:[BX-01]	268B4FFF	(BX) = 0003 (CX) = 120E (ES) = 1A07 (IP) = 004B	(BX) = 0003 (CX) = FFCE (ES) = 1A07 (IP) = 004F
004F	XCHG AX, CX	91	(AX) = 0000 (CX) = FFCE (IP) = 004F	(AX) = FFCE (CX) = 0000 (IP) = 0050
0050	MOV DI, 0002	BF0200	(DI) = 0002 (IP) = 0050	(DI) = 0002 (IP) = 0053
0053	MOV ES:[BX+DI], AX	268901	(AX) = FFCE (BX) = 0003 (DI) = 0002 (ES) = 1A07 (IP) = 0053	(AX) = FFCE (BX) = 0003 (DI) = 0002 (ES) = 1A07 (IP) = 0056
0056	MOV BP, SP	8BEC	(BP) = 0000 (SP) = 0014 (IP) = 0056	(BP) = 0014 (SP) = 0014 (IP) = 0058
0058	PUSH [0000]	FF360000	(SP) = 0014 (IP) = 0058 Stack +0 0000 +2 19F5 +4 0000 +6 0000	(SP) = 0012 (IP) = 005C Stack +0 01F4 +2 0000 +4 19F5 +6 0000
005C	PUSH [0002]	FF360200	(SP) = 0012 (IP) = 005C Stack +0 01F4 +2 0000 +4 19F5	(SP) = 0010 (IP) = 0060 Stack +0 FFCE +2 01F4 +4 0000

			+6 0000	+6 19F5
0060	MOV BP, SP	8BEC	(BP) = 0014 (SP) = 0010 (IP) = 0060	(BP) = 0010 (SP) = 0010 (IP) = 0062
0062	MOV DX, [BP+02]	8B5602	(DX) = 0000 (BP) = 0010 (IP) = 0062	(DX) = 01F4 (BP) = 0010 (IP) = 0065
0065	RET Far 0002	CA0200	(SP) = 0010 (CS) = 1A0A (IP) = 0065 +0 FFCE +2 01F4 +4 0000 +6 19F5	(SP) = 0016 (CS) = 01F4 (IP) = FFCE +0 19F5 +2 0000 +4 0000 +6 0000

Выводы.

В ходе лабораторной работы были изучены режимы адресации на практике на языке Ассемблер.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ПРОГРАММ

Название файла: LR2_comp.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
```