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. 0382 _____ Азаров М.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

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.

Порядок выполнения работы.

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.

Ход работы

1. Описание обнаруженных при первоначальной трансляции ошибок и их объяснение :
    - (47) ***mov mem3, [bx]*** - Нельзя читать из памяти и писать в память одной командой .
    - (54) ***mov cx, vec2[di]*** - Несоответствие типов операндов.
    - (58) ***mov cx, matr[bx][di]*** - Несоответствие типов операндов.
    - (59) ***mov ax, matr[bx*4][di]*** - Нельзя умножать 16-битные регистры, нужно использовать регистры EXX.
    - (79) ***mov ax, matr[bp+bx]*** - Нельзя использовать более одного базового регистра.
    - (80) ***mov ax, matr[bp+di+si]*** - Нельзя использовать более одного индексного регистра.

2. Запуск программы под управлением отладчика с пошаговым выполнением и занесением данных в таблицу 1:

Начальное значение сегментных регистров:

CS = 1A0A ; DS = 19F5 ;

ES = 19F5 ; SS = 1A05 ;

Таблица 1 - Отладка LR2_COMP.EXE

Адрес Команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			до выполнения	После выполнения
0000	PUSH DS	1E	DS = 19F5 IP = 0000 SP = 0018 Stack +0 0000 +2 0000 +4 0000 +6 0000	DS = 19F5 IP = 0001 SP = 0016 Stack +0 19F5 +2 0000 +4 0000 +6 0000
0001	SUB AX, AX	2BC0	AX = 0000 IP = 0001	AX = 0000 IP = 0003
0003	PUSH AX	50	AX = 0000 IP = 0003 SP = 0016 Stack +0 19F5 +2 0000 +4 0000 +6 0000	AX = 0000 IP = 0004 SP = 0014 Stack +0 0000 +2 19F5 +4 0000 +6 0000
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	AX = 1A07 IP = 0009	AX = 01F4 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	IP = 0012 DS:0002 = 00 DS:0003 = 00	IP = 0018 DS:0002 = CE DS:0003 = FF
0018	MOV BX, 0006	BB0600	BX = CE24 IP = 0018	BX = 0006 IP = 001B
001B	MOV [0000], AX	A30000	IP = 001B AX = 01F4 DS:0000 = 00 DS:0001 = 00	IP = 001E AX = 01F4 DS:0000 = F4 DS:0001 = 01
001E	MOV AL, [BX]	8A07	AX = 01F4 BX = 0006 IP = 001E	AX = 0101 BX = 0006 IP = 0020
0020	MOV AL, [BX+03]	8A4703	AX = 0101 BX = 0006 IP = 0020	AX = 0104 BX = 0006 IP = 0023
0023	MOV CX, [BX+03]	8B4F03	CX = 01F4 BX = 0006 IP = 0023	CX = 0804 BX = 0006 IP = 0026
0026	MOV DI, 0002	BF0200	DI = 0000 IP = 0026	DI = 0002 IP = 0029
0029	MOV AL,[000E+DI]	8A850E00	AX = 0104 DI = 0002 IP = 0029 DS:0010 = 0A	AX = 010A DI = 0002 IP = 002D DS:0010 = 0A
002D	MOV BX, 0003	BB0300	BX = 0006 IP = 0002D	BX = 0003 IP = 00030
0030	MOV AL,[0016+BX+DI]	8A811600	AX = 010A	AX = 01FD

			DX = 0003 DI = 0002 IP = 0030 DS:001B = FD	DX = 0003 DI = 0002 IP = 0034 DS:001B = FD
0034	MOV AX, 1A07	B8071A	IP = 0034 AX = 01FD	IP = 0037 AX = 1A07
0037	MOV ES,AX	8EC0	ES = 19F5 AX = 1A07 IP = 0037	ES = 1A07 AX = 1A07 IP = 0039
0039	MOV AX, ES:[BX]	268B07	AX = 1A07 ES = 1A07 BX = 0003 IP = 0039 DS:0003 = FF DS:0004 = 00	AX = 00FF ES = 1A07 BX = 0003 IP = 003C DS:0003 = FF DS:0004 = 00
003C	MOV AX, 0000	B80000	IP = 003C AX = 00FF	IP = 003F AX = 0000
003F	MOV ES, AX	8EC0	IP = 003F ES = 1A07 AX = 0000	IP = 0041 ES = 0000 AX = 0000
0041	PUSH DS	1E	DS = 1A07 IP = 0041 SP = 0014 Stack +0 0000 +2 19F5 +4 0000 +6 0000	DS = 1A07 IP = 0042 SP = 0012 Stack +0 1A07 +2 0000 +4 19F5 +6 0000
0042	POP ES	07	ES = 0000 IP = 0042 SP = 0012 Stack +0 1A07 +2 0000 +4 19F5	ES = 1A07 IP = 0043 SP = 0014 Stack +0 0000 +2 19F5 +4 0000

0043	MOV CX, ES:[BX-01]	268B4FFF	CX = 0804 IP = 0043 BX = 0003 ES =1A07 DS:0002 = CE DS:0003 = FF	CX = FFCE IP = 0047 BX = 0003 ES =1A07 DS:0002 = CE DS:0003 = FF
0047	XCHG AX,CX	91	AX = 0000 CX = FFCE IP = 0047	AX = FFCE CX = 0000 IP = 0048
0048	MOV DI, 0002	BF0200	DI = 0002 PI = 0048	DI = 0002 PI = 004B
004B	MOV ES:[BX+DI], AX	268901	AX= FFCE IP = 004B BX = 0003 ES =1A07 DI = 0002 DS:0005 = 00 DS:0006 = 01	AX= FFCE IP = 004E BX = 0003 ES =1A07 DI = 0002 DS:0005 = CE DS:0006 = FF
004E	MOV BP, SP	8BEC	SP = 0014 BP = 0000 PI = 004E	SP = 0014 BP = 0014 PI = 0050
0050	PUSH [0000]	FF360000	IP = 0050 SP = 0014 DS:0000 = F4 DS:0001 = 01 Stack +0 0000 +2 19F5 +4 0000 +6 0000	IP = 0054 SP = 0012 DS:0000 = F4 DS:0001 = 01 Stack +0 01F4 +2 0000 +4 19F5 +6 0000
0054	PUSH [0002]	FF360000	IP = 0054 SP = 0012 DS:0002 = CE DS:0003 = FF	IP = 0058 SP = 0010 DS:0002 = CE DS:0003 = FF

			Stack +0 01F4 +2 0000 +4 19F5 +6 0000	Stack +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	MOV BP, SP	8BEC	BP = 0014 SP = 0010 IP = 0058	BP = 0010 SP = 0010 IP = 005A
005A	MOV DX, [BP+02]	8B5602	BP = 0010 DX = 0000 IP = 005A SS:0012 = 01F4	BP = 0010 DX = 01F4 IP = 005D SS:0012 = 01F4
005D	RET Far 0002	CA0200	IP = 005D SP = 0010 CS = 1A0A Stack +0 FFCE +2 01F4 +4 0000 +6 19F5	IP = FFCE SP = 0016 CS = 01F4 Stack +0 19F5 +2 0000 +4 0000 +6 0000

Вывод.

Изучены режимы адресации и формирование исполнительного адреса.

В ходе работы был исправлен и пошагово отлажен исходных файл.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

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 1,2,3,4,8,7,6,5
vec2 DB -10,-20,10,20,-30,-40,30,40
matr DB 1,2,3,4,-4,-3,-2,-1,5,6,7,8,-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] ;Нельзя читать из памяти и писать в память одной
командой
```

```
; Базированная адресация
      mov al,[bx]+3
      mov cx,3[bx]
; Индексная адресация
      mov di,ind
      mov al,vec2[di]
      ;mov cx,vec2[di] ;Несоответствие типов операндов
; Адресация с базированием и индексированием
      mov bx,3
      mov al,matr[bx][di]
      ;mov cx,matr[bx][di] ;Несоответствие типов операндов
      ;mov ax,matr[bx*4][di] ;Нельзя умножать 16-битные регистры, нужно
использовать регистры EXX

; 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] ;Нельзя использовать более одного базового
регистра
      ;mov ax,matr[bp+di+si] ;Нельзя использовать более одного
индексного регистра
; Использование сегмента стека
      push mem1
      push mem2
      mov bp,sp
      mov dx,[bp]+2
      ret 2
Main ENDP
CODE ENDS
      END Main
```

Название файла: LR2.LST

```
     #Microsoft (R) Macro Assembler Version 5.10
10/19/21 14:33:0
                                                                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  01 02 03 04 08 07      vec1 DB 1,2,3,4,8,7,6,5
       06 05
 000E  F6 EC 0A 14 E2 D8      vec2 DB -10,-20,10,20,-30,-40,30,40
       1E 28
 0016  01 02 03 04 FC FD      matr DB 1,2,3,4,-4,-3,-2,-1,5,6,7,8,-8,-7,-
6,-5
       FE FF 05 06 07 08
       F8 F9 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]
#Microsoft (R) Macro Assembler Version 5.10                  10/19/21
14:33:0
                                                             Page
1-2


                           ;mov mem3,[bx] ;Нельзя читать из памяти
                       и писать в память одной командой
                      ; 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] ;Несоответствие типов
                      операндов
                      ; 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] ;Несоответствие ти
                      пов операндов
                           ;mov ax,matr[bx*4][di] ;Нельзя умножать
                       16-битные регистры, нужно использовать регистр
                      ы EXX

                      ; 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] ;Нельзя использоват
                      ь более одного базового регистра
                           ;mov ax,matr[bp+di+si] ;Нельзя использо
```

```
                        вать более одного индексного регистра
                        ; 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 Mai
#Microsoft (R) Macro Assembler Version 5.10                  10/19/21
14:33:0
                                                             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  LR2_COMP
@VERSION . . . . . .            TEXT  510


     89 Source  Lines
     89 Total   Lines
     19 Symbols

  47796 + 459464 Bytes symbol space free
```

```
0 Warning Errors
0 Severe  Errors
```