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»
Тема: Изучение режимов адресации и
формирования исполнительного адреса
Вариант 7

Студент гр. 0382 _____ Ильин Д.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучить работу режимов адресации процессора Intel X86, используя программу 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.

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.

Вариант №7:

vec1 DB 21,22,23,24,28,27,26,25

vec2 DB 40,50,-40,-50,20,30,-20,-30

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

Выполнение работы.

Описание ошибок, обнаруженных при первоначальной трансляции:

Ошибка	Объяснение
mov mem3,[bx]	Нельзя в сегмент данных отправлять данные напрямую, только через регистры
mov cx,vec2[di]	cx- DW, vec2[di]- DB
mov cx,matr[bx][di]	cx- DW, vec2[di]- DB
mov ax,matr[bx*4][di]	Нельзя маштабировать 2-х байтные регистры
mov ax,matr[bp+bx]	Нельзя в индексации использовать несколько базовых регистров
mov ax,matr[bp+di+si]	Нельзя в индексации использовать несколько сегментных регистров

ПРОТОКОЛ

Таблица 1. Результат выполнения программы в пошаговом режиме.

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 Stack:+0 0000	(SP) = 0016 (DS) = 19F5 Stack: +0 19F5
0001	SUB AX, AX	2BC0	(AX) = 0000	(AX) = 0000
0003	PUSH AX	50	(SP) = 0016 (AX) = 0000 Stack: +0 19F5	(SP) = 0014 (AX) = 0000 Stack: +0 0000 +2 19F5
0004	MOV AX, 1A07	B8071A	(AX) = 0000	(AX) = 1A07
0007	MOV DS, AX	8ED8	(DS) = 19F5	(DS) = 1A07
0009	MOV AX, 01F4	B8F401	(AX) = 1A07	(AX) = 01F4
000C	MOV CX, AX	8BC8	(CX) = 00B0	(CX) = 01F4
000E	MOV BL, 24	B324	(BX) = 0000	(BX) = 0024
0010	MOV BH, CE	B7CE	(BX) = 0024	(BX) = CE24
0012	MOV [0002], FFCE	C7060200CE FF		
0018	MOV BX, 0006	BB0600	(BX) = CE24	(BX) = 0006
001B	MOV [0000], AX	A30000		
001E	MOV AL, [BX]	8A07	(AX) = 01F4	(AX) = 0126
0020	MOV AL, [BX+03]	8A4703	(AX) = 011F	(AX) = 0123
0023	MOV CX, [BX+03]	8B4F03	(CX) = 01F4	(CX) = 1F23
0026	MOV DI, 0002	BF0200	(DI) = 0000	(DI) = 0002

0029	MOV AL, [000E+DI]	8A850E00	(AX) = 0123	(AX) = 01BA
002D	MOV BX, 0003	BB0300	(BX) = 0006	(BX) = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	(AX) = 01BA	(AX) = 01F9
0034	MOV AX, 1A07	B8071A	(AX) = 01F9	(AX) = 1A07
0037	MOV ES, AX	8EC0	(ES) = 19F5	(ES) = 1A07
0039	MOV AX, ES:[BX]	268B07	(AX) = 1A07	(AX) = 00FF
003C	MOV AX, 0000	B80000	(AX) = 00FF	(AX) = 0000
003F	MOV ES, AX	8EC0	(ES) = 1A07	(ES) = 0000
0041	PUSH DS	1E	(DS) = 1A07 (SP) = 0014 Stack: +0 0000 +2 19F5	(DS) = 1A07 (SP) = 0012 Stack: +0 1A07 +2 0000 +4 19F5
0042	POP ES	07	(ES) = 0000 (SP) = 0012 Stack: +0 1A07	(ES) = 1A07 (SP) = 0014 Stack: +0 0000
0043	MOV CX, ES:[BX-01]	268B4FFF	(CX) = 1F23	(CX) = FFCE
0047	XCHG AX, CX	91	(AX) = 0000 (CX) = FFCE	(AX) = FFCE (CX) = 0000
0048	MOV DI, 0002	BF0200	(DI) = 0002	(DI) = 0002
004B	MOV ES:[BX+DI], AX	268901	5 6 DS:00 26	5 6 DS:CE FF
004E	MOV BP, SP	8BEC	(BP) = 0010	(BP) = 0014
0050	PUSH 01F4	FF360000	(SP) = 0014 Stack: +0 0000 +2 19F5 +4 0000	(SP) = 0012 Stack: +0 01F4 +2 0000 +4 19F5
0054	PUSH FFCE	FF360200	(SP) = 0012	(SP) = 0010

			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	(BP) = 0010
005A	MOV DX, [BP+02]	8B5602	(DX) = 0000	(DX) = 01F4
005D	RET far 0002	CA0200	(CS) = 1A0A (SP) = 0010 Stack: +0 FFCE +2 01F4 +4 0000 +6 19F5	(CS) = 01F4 (SP) = 0016 Stack: +0 19F5 +2 0000 +4 0000 +6 0000

Вывод.

В результате работы была изучена работа режимов адресации с использованием программы на языке Ассемблера.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

LB2.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 21,22,23,24,28,27,26,25
    vec2 DB 40,50,-40,-50,20,30,-20,-30
    matr DB 5,6,-8,-7,7,8,-6,-5,1,2,-4,-3,3,4,-2,-1

    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 В

LB2.lst

```
#Microsoft    (R)    Macro    Assembler    Version    5.10
10/6/21 02:10:02

Page     1-1


                        ;                    Ð Ñ ÐŸÐ³Ñ Ð°ÐŒÐŒÐ°
ÐžÐ·Ñ Ñ ÐµÐœÐžÑ Ñ ÐµÐ¶ÐžÐ
                        ŒÐŸÐ²                 Ð°ÐŽÑ ÐµÑ Ð°Ñ ÐžÐž
Ð¿Ñ ÐŸÑ ÐµÑ Ñ ÐŸÑ Ð° I
                        ntelX86

      = 0024                EOL EQU '$'
      = 0002                ind EQU 2
      = 01F4                n1 EQU 500
      =-0032                n2 EQU -50

                        ; Ð¡Ñ(ÐµÐ° Ð¿Ñ(ÐŸÐ³Ñ(Ð°ÐŒÐŒÑ)
      0000                  AStack SEGMENT STACK
      0000  000C[               DW 12 DUP(?)
            ????
                  ]

      0018                  AStack ENDS

                        ;                          Ð Ð°ÐœÐœÑ Ðµ
Ð¿Ñ ÐŸÐ³Ñ Ð°ÐŒÐŒÑ
      0000                  DATA SEGMENT

                        ;                    Ð ÐžÑ ÐµÐ°Ñ ÐžÐ²Ñ
ÐŸÐ¿ÐžÑ Ð°ÐœÐžÑ  ÐŽÐ°ÐœÐœÑ
                        Ñ
      0000  0000            mem1 DW 0
      0002  0000            mem2 DW 0
      0004  0000            mem3 DW 0
      0006  15 16 17 18 1C 1B vec1                       DB
21,22,23,24,28,27,26,25

            1A 19
      000E  28 32 D8 CE 14 1E vec2      DB      40,50,-40,-
50,20,30,-20,-30
            EC E2
```

```
     0016  05 06 F8 F9 07 08 matr  DB  5,6,-8,-7,7,8,-6,-
5,1,2,-4,-3,3,4,-2,-1
           FA FB 01 02 FC FD
           03 04 FE FF

     0026                DATA ENDS

                       ; Ð¤ÐŸÐŽ Ð¿Ñ€ÐŸÐ³Ñ€Ð°ÐŒÐŒÑ‚

     0000                CODE SEGMENT
                         ASSUME     CS:CODE,     DS:DATA,
SS:AStack

                       ;                    Ð©ÐŸÐ»ÐŸÐ²ÐœÐ°Ñ‰
Ð¿Ñ€ÐŸÑ‘ÐµÐŽÑ€Ñ‹Ñ€Ð°
     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

                       ;                Ð Ð Ð¦Ð¤Ð Ð¤Ð
Ð Ð Ð¾Ð★Ð©Ð¦Ð¤ Ð¬Ð«Ð Ð Ð¡Ð¬Ð
                       ŠÐ★Ð★     Ð¦Ð¬     Ð£Ð Ð¦Ð¤Ð¦Ð Ð
Ð¡Ð©Ð Ð©Ð Ð¦Ð★Ð✍

                       ;          Ð ÐµÐ³ÐžÑ€Ñ€Ñ€ÐŸÐ²Ð°Ñ‰
Ð°ÐŽÑ€ÐµÑ€Ð°Ñ‘ÐžÑ‰
     0009  B8 01F4              mov ax,n1
     000C  8B C8           mov cx,ax
     000E  B3 24           mov bl,EOL
```

```
0010  B7 CE              mov bh,n2
                      ;                    Ð Ñ Ñ ÐŒÐ°Ñ
Ð°ÐŽÑ ÐµÑ Ð°Ñ Ðž
0012  C7 06 0002 R FFCE      mov mem2,n2
0018  BB 0006 R          mov bx,OFFSET vec1
001B  A3 0000 R          mov mem1,ax
                      ;              Ð ÐŸÑ Ð²ÐµÐœÐ°Ñ
Ð°ÐŽÑ ÐµÑ Ð°Ñ Ðž
001E  8A 07              mov al,[bx]
                         ;mov mem3,[bx]        ;ÐŒÑ  ÐœÐµ
ÐŒÐŸ
                      Ð¶ÐµÐŒ       Ð²       Ñ ÐµÐ³ÐŒÐµÐœÑ
ÐŽÐ°ÐœÐœÑ Ñ  ÐŸÑ Ð¿Ñ Ð
                      °Ð²Ð»Ñ Ñ Ñ           ÐŽÐ°ÐœÐœÑ Ðµ
ÐœÐ°Ð¿Ñ Ñ ÐŒÑ Ñ , Ñ ÐŸ
                      Ð»Ñ Ð°ÐŸ Ñ ÐµÑ(ÐµÐ· Ñ(ÐµÐ³ÐžÑ)Ñ(Ñ)
                      ;          Ð Ð°Ð·ÐžÑ ÐŸÐ²Ð°ÐœÐœÐ°Ñ
Ð°ÐŽÑ ÐµÑ Ð°Ñ Ðž
0020  8A 47 03              mov al,[bx]+3
0023  8B 4F 03              mov cx,3[bx]
                      ;              Ð ÐœÐŽÐµÐ°Ñ ÐœÐ°Ñ
Ð°ÐŽÑ ÐµÑ Ð°Ñ Ðž
0026  BF 0002              mov di,ind
0029  8A 85 000E R          mov al,vec2[di]
                         ;mov cx,vec2[di] ;cx- DW, vec2[d
                      i]- DB
                      ;      Ð ÐŽÑ ÐµÑ Ð°Ñ ÐžÑ      Ñ
Ð±Ð°Ð·Ñ ÐŸÐ²Ð°ÐœÐžÐµÐ
                      Œ                                  Ðž
ÐžÐœÐŽÐµÐ°Ñ ÐžÑ ÐŸÐ²Ð°ÐœÐžÐµÐŒ
002D  BB 0003              mov bx,3
0030  8A 81 0016 R          mov al,matr[bx][di]
                         ;mov cx,matr[bx][di] ;cx-    DW,
matr[b
                      x][di]- DB
                         ;mov          ax,matr[bx*4][di]
    ;Ñ ÐŒÐœÐŸÐ¶Ð°Ñ
                      Ñ     ÐœÐ°    Ð°ÐŸÐœÑ)Ñ(Ð°ÐœÑ(Ñ)    ÐŒÑ)
ÐŒÐŸÐ¶ÐµÐŒ Ñ(ÐŸ
```

```
                        Ð»Ñ‹Ð°ÐŸ                ÐžÐœÐŽÐµÐ°Ñ)ÐœÑ)Ðµ
Ñ ÐµÐ³ÐžÑ Ñ Ñ Ñ Ñ‚, Ñ
                        (.Ðµ. si, di ?????

                        ;                    Ð Ð Ð Ð Ð Ð Ð Ð
Ð Ð Ð Ð Ð Ð Ð Ð¡Ð Ð Ð Ð Ð Ð¡Ð Ð
                        ŠÐ Ð          Ð¡        Ð£Ð§Ð Ð¢Ð Ð Ð
Ð¡Ð Ð Ð Ð Ð Ð Ð¢Ð Ð Ð
                        ;    Ð ÐµÑ ÐµÐŸÐ¿Ñ ÐµÐŽÐµÐ»ÐµÐœÐžÐµ
Ñ ÐµÐ³ÐŒÐµÐœÑ
                        Ð°
                        ; ------ Ð²Ð°Ñ ÐžÐ°ÐœÑ  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
                        ; ------ Ð²Ð°Ñ ÐžÐ°ÐœÑ  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
                        ; ------ Ð²Ð°Ñ ÐžÐ°ÐœÑ  3
     0048  BF 0002                mov di,ind
     004B  26: 89 01          mov es:[bx+di],ax
                        ; ------ Ð²Ð°Ñ ÐžÐ°ÐœÑ  4
     004E  8B EC              mov bp,sp
                           ;mov ax,matr[bp+bx]   ;ÐŒÑ  ÐœÐµ
ÐŒÐŸ
                        Ð¶ÐµÐŒ  Ð²  Ñ Ð°Ñ Ñ ÐžÑ ÐµÐœÐœÐŸÐ¹
ÐžÐœÐŽÐµÐ°Ñ Ð°
                        Ñ Ð Ð»Ð°ÐŽÑ Ð²Ð°Ñ (Ñ 
ÐœÐµÑ Ð°ÐŸÐ»Ñ Ð°ÐŸ
                        Ñ ÐµÐ³ÐžÑ Ñ Ñ Ñ ÐŸÐ²
```

```
                        ;mov ax,matr[bp+di+si]
                    ;       Ð☆Ñ▌Ð¿ÐŸÐ»Ñ◙Ð·ÐŸÐ²Ð°ÐœÐžÐµ
Ñ▌ÐµÐ³ÐŒÐµÐœÑ▌Ð° Ñ
                        ▌Ñ▌ÐµÐ°Ð°
     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
10/6/21 02:10:02

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


     92 Source  Lines
     92 Total   Lines
     19 Symbols

  47808 + 459452 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```