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 формировании исполнительного адреса

Студент гр. 1303 _____ Чернуха В.В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Изучить виды адресации, принцип их работы в 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.

Выполнение работы.

1. Согласно 6 варианту были взяты следующие значения:

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

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

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

2. Программа была протранслированна и были выявлены следующие ошибки:

- mov mem3,[bx]

  lr2.asm(45): error A2052: Improper operand type

  Мы не можем напрямую перемещать значения из одного сегмента памяти в другой.

- mov cx,vec2[di]

  lr2.asm(53): warning A4031: Operand types must match

  Мы не можем поместить в двухбайтный регистр однобайтое значение

- mov cx,matr[bx][di]

  lr2.asm(57): warning A4031: Operand types must match

  Ситуация аналогична — попытка поместить однобайтное значение в двухбайтный регистр.

- mov ax,matr[bx*4][di]

  lr2.asm(58): error A2055: Illegal register value

  Мы не можем умножать регистр bx

- mov ax,matr[bp+bx]

  lr2.asm(78): error A2046: Multiple base registers

  Мы не можем использовать несколько базовых регистров в операнде

- mov ax,matr[bp+di+si]

  lr2.asm(79): error A2047: Multiple index registers

  Мы не можем использовать несколько индексных регистров в операнде

  3. Исправленная программа была протранслированна и слинкована

  4. Программа запущена в пошаговом режиме в отладчике AFDPRO.

Начальные значения регистров:

AX = 0000	SI = 0000	CS = 1A0A	IP = 0000
BX = 0000	DI = 0000	DS = 19F5	HS = 19F5
CX = 00B0	BP = 0000	ES = 19F5	FS = 19F5
DX = 0000	SP = 0018	SS = 1A05	

Пошаговая работа программы

Адрес Команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	push DS	1E	SP = 0018 IP = 0000 Stack +0 0000	SP = 0016 IP = 0001 Stack +0 19F5
0001	sub AX, AX	2BC0	IP = 0001 AX = 0000	IP = 0003 AX = 0000
0003	push AX	50	SP = 0016 IP = 0003 Stack +0 19F5	SP = 0014 IP = 0004 Stack +0 0000 Stack +2 19F5
0004	mov AX, 1A07	B8071A	AX = 0000 IP = 0004	AX = 1A07 IP = 0007
0007	mov DS, AX	8ED8	DS = 19F5 IP = 0007	DS = 1A07 IP = 0009
0009	mov AX, 01F4		AX = 1A07	AX = 01F4

		B8F401	IP = 0009	IP = 000C
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	C7060200C EFF	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 DS:0000 = 00 DS:0001 = 00	IP = 001E DS:0000 = F4 DS:0001 = 01
001E	mov AL, [BX]	8A07	AX = 01F4 IP = 001E	AX = 0112 IP = 0020
0020	mov AL, [BX+03]	8A4703	AX = 0112 IP = 0020	AX = 010F 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 BX,0003	BB0300	BX = 0006 IP = 002D	BX = 0003 IP = 0030

0030	mov AL,[0016+BX+DI]	8A811600	AX = 01E2 IP = 0030	AX = 01FF IP = 0034
0034	mov AX, 1A07	B8071A	AX = 01FF IP = 0034	AX = 1A07 IP = 0037
0037	mov ES,AX	8EC0	ES = 19F5 IP = 0037	ES = 1A07 IP = 0039
0039	mov AX.ES:[BX]	268B07	AX = 1A07 IP = 0037	AX = 00FF IP = 003C
003C	mov AX,0000	B80000	AX = 00FF IP = 003C	AX = 0000 IP = 003F
003F	mov ES,AX	8EC0	ES = 1A07 IP = 003F	ES = 0000 IP = 0041
0041	push DS	1E	SP = 0014 IP = 0041 Stack +0 = 0000 Stack +2 = 19F5 Stack +4 = 0000	SP = 0012 IP = 0042 Stack +0 = 1A07 Stack +2 = 0000 Stack +4 = 19F5
0042	pop ES	07	SP = 0012 IP = 0042 ES = 0000 Stack +0 = 1A07 Stack +2 = 0000 Stack +4 = 19F5	SP = 0014 IP = 0043 ES = 1A07 Stack +0 = 0000 Stack +2 = 19F5 Stack +4 = 0000
0043	mov CX,ES:[BX-01]	268B4FFF	CX = 0B0F IP = 0043	CX = FFCE IP = 0047

0047	xchg AX,CX	91	CX = FFCE AX = 0000 IP = 0047	CX = 0000 AX = FFCE IP) = 0048
0048	mov DI,0002	BF0200	DI = 0002 IP = 0048	DI = 0002 IP = 004B
004B	mov ES:[BX+DI],AX	268901	DS:0005 = 00 DS:0006 = 12 IP = 004B	DS:0005 = CE DS:0006 = FF IP = 004E
004E	mov BP,SP	8BEC	BP = 0000 IP = 004E	BP = 0014 IP = 0050
0050	push [0000]	FF360000	SP = 0014 IP = 0050 Stack +0 = 0000 Stack +2 = 19F5 Stack +4 = 0000	SP = 0012 IP = 0054 Stack +0 = 01F4 Stack +2 = 0000 Stack +4 = 19F5
0054	push [0002]	FF360200	SP = 0012 IP = 0054 Stack +0 = 01F4 Stack +2 = 0000 Stack +4 = 19F5 Stack +6 = 0000	SP = 0010 IP = 0058 Stack +0 = FFCE Stack +2 = 01F4 Stack +4 = 0000 Stack +6 = 19F5
0058	mov BP,SP	8BEC	BP = 0014 IP = 0058	BP = 0010 IP = 005A
005A	mov DX,[BP+02]	8B5602	DX = 0000 IP = 005A	DX = 01F4 IP = 005D
005D	ret Far 0002	CA0200	SP = 0010 CS = 1A0A	SP = 0016 CS = 01F4

			IP = 005D Stack +0 = FFCE Stack +2 = 01F4 Stack +4 = 0000 Stack +6 = 19F5	IP = FFCE Stack +0 = 19F5 Stack +2 = 0000 Stack +4 = 0000 Stack +6 = 0000

Выводы.

В результате были изучены виды адресации, их принцип работы, распространённые ошибки при работе с адресацией.

ПРИЛОЖЕНИЕ 1

lr2.asm

```
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
mem1DW    0
mem2DW    0
mem3DW    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]
; 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
```

```
= 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 данно́
                        x
0000  0000              mem1DW  0
0002  0000              mem2DW  0
0004  0000              mem3DW  0
0006  12 11 10 0F 0B 0C    vec1  DB   18,17,16,15,11,12,13,14
    0D 0E
000E  1E 28 E2 D8 0A 14    vec2  DB   30,40,-30,-40,10,20,-10,-20
    F6 EC
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
    03 04 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 АДРЕСА□
                        ИИ 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              10/6/22 01:37:04
                                        Page     1-2


                        ;mov mem3,[bx]
                        ; 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]
                        ; Адресация с базирование□
                         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 АДРЕСА□
                        ИИ С УЧЕТОМ СЕГМЕНТОВ
                        ; Переопределение сегмент
                        а
                        ;       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 ύ
                        тека
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                 10/6/22 01:37:04
                                        Symbols-1


Segments and Groups:

                N a m e         Length         AlignCombine 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  lr2ispr2
@VERSION . . . . . .      TEXT  510


    88 Source  Lines
    88 Total   Lines
    19 Symbols

 47800 + 459460 Bytes symbol space free

     0 Warning Errors
     0 Severe  Errors
```