МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Организация ЭВМ и систем»
Тема: Изучение режимов адресации в Intel8086

Студент гр. 9382 _____ Русинов Д.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Изучить режимы адресации, указать на ошибки в программе и объяснить их.

Основные теоретические положения.

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.

Исходный код программы:

```
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50

AStack          SEGMENT STACK
        DW 12 DUP(?)
AStack          ENDS

DATA   SEGMENT

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

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

Main    PROC FAR
        push DS
        sub AX,AX
        push AX
        mov AX,DATA
        mov DS,AX

        mov ax,n1
        mov cx,ax
        mov bl,EOL
```

```
mov bh,n2

mov mem2,n2
mov bx,OFFSET vec1
mov mem1,ax

mov al,[bx]
; mov mem3,[bx]

mov al,[bx]+3
mov cx,3[bx]

mov di,ind
mov al,vec2[di]
; mov cx,vec2[di]

mov bx,3
mov al,matr[bx][di]
; mov cx,matr[bx][di]
; mov ax,matr[bx*4][di]

mov ax, SEG vec2
mov es, ax
mov ax, es:[bx]
mov ax, 0

mov es, ax
push ds
pop es
mov cx, es:[bx-1]
xchg cx,ax

mov di,ind
```

```
        mov es:[bx+di],ax

        mov bp,sp
        ; mov ax,matr[bp+bx]
        ; mov ax,matr[bp+di+si]

        push mem1
        push mem2
        mov bp,sp
        mov dx,[bp]+2
        ret 2
Main    ENDP
CODE    ENDS
END Main
```

Экспериментальные результаты.

Листинг трансляции программы:

```
Microsoft (R) Macro Assembler Version 5.10              10/10/20 14:40:5
                                                        Page     1-1


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

0000                        AStack          SEGMENT STACK
0000  000C[                            DW 12 DUP(?)
          ????
                            ]

0018                        AStack          ENDS

0000                        DATA    SEGMENT

0000  0000                  mem1    DW 0
0002  0000                  mem2    DW 0
0004  0000                  mem3    DW 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
```

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

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

0009  B8 01F4                          mov ax,n1
000C  8B C8                     mov cx,ax
000E  B3 24                     mov bl,EOL
0010  B7 CE                     mov bh,n2

0012  C7 06 0002 R FFCE         mov mem2,n2
0018  BB 0006 R                 mov bx,OFFSET vec1
001B  A3 0000 R                 mov mem1,ax

001E  8A 07                     mov al,[bx]
                                mov mem3,[bx]
CHECK.ASM(40): error A2052: Improper operand type

0020  8A 47 03                          mov al,[bx]+3
0023  8B 4F 03                          mov cx,3[bx]

0026  BF 0002                           mov di,ind
0029  8A 85 000E R              mov al,vec2[di]
```

```
002D  8B 8D 000E R                mov cx,vec2[di]
CHECK.ASM(47): warning A4031: Operand types must match

0031  BB 0003                         mov bx,3
0034  8A 81 0016 R                mov al,matr[bx][di]
0038  8B 89 0016 R                mov cx,matr[bx][di]
CHECK.ASM(51): warning A4031: Operand types must match
003C  8B 85 0022 R                mov ax,matr[bx*4][di]
CHECK.ASM(52): error A2055: Illegal register value

0040  B8 ---- R                   mov ax, SEG vec2
0043  8E C0                       mov es, ax
0045  26: 8B 07                   mov ax, es:[bx]
0048  B8 0000                         mov ax, 0

004B  8E C0                       mov es, ax
004D  1E                          push ds
004E  07                          pop es
004F  26: 8B 4F FF                mov cx, es:[bx-1]
0053  91                          xchg cx,ax

0054  BF 0002                         mov di,ind
0057  26: 89 01                   mov es:[bx+di],ax

005A  8B EC                       mov bp,sp
005C  3E: 8B 86 0016 R                 mov ax,matr[bp+bx]
CHECK.ASM(69): error A2046: Multiple base registers
0061  3E: 8B 83 0016 R                 mov ax,matr[bp+di+si]
CHECK.ASM(70): error A2047: Multiple index registers
```

```
0066  FF 36 0000 R                    push mem1
006A  FF 36 0002 R                    push mem2
006E  8B EC                           mov bp,sp
0070  8B 56 02                              mov dx,[bp]+2
0073  CA 0002                               ret 2
0076                            Main  ENDP
CHECK.ASM(77): error A2006: Phase error between passes
0076                            CODE  ENDS
                                END Main
Microsoft (R) Macro Assembler Version 5.10          10/10/20 14:40:5
                                                    Symbols-1


Segments and Groups:

                N a m e         Length  Align   Combine Class

ASTACK . . . . . .              0018    PARA    STACK
CODE . . . . . .                0076    PARA    NONE
DATA . . . . . .                0026    PARA    NONE

Symbols:

                N a m e         Type     Value   Attr

EOL  . . . . . .                NUMBER          0024

IND  . . . . . .                NUMBER          0002

MAIN . . . . . .                F PROC0000      CODE    Length = 0076
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  CHECK
@VERSION . . . . . .    TEXT  510


   79 Source  Lines
   79 Total   Lines
   19 Symbols

 47812 + 461495 Bytes symbol space free

    2 Warning Errors
    5 Severe  Errors
```

Были обнаружены и закомментированы 6 ошибок:

```
mov mem3, [bx]
6mov al, [bx]+3
mov cx, vec2[di]
mov cx, matr[bx][di]
mov ax, matr[bx*4][di]
mov ax, matr[bp+bx]
mov ax, matr[bp+di+si]
```

Обработка результатов эксперимента.

mov mem3, [bx]

Ошибка: “Improper operand type”

Мы не можем передавать объекты из памяти в память напрямую. Чтобы переместить данные из ячейки [bx], то нужно сделать это через регистр.

mov cx, vec2[di]

Ошибка: “Operand types must match”

Массив vec2 хранит элементы размеров DB – 1 байт. А регистр CX хранит 2 байта. Необходимо, чтобы запись производилась между операндами одного размера.

mov cx, matr[bx][di]

Ошибка: “Operand types must match”

Та же самая ошибка, которая возникла при операции “mov cx, vec2[di]”

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

Ошибка: “Illegal register value”

Умножение на число применимо только к регистрам, начинающимся с “e”.

mov ax, matr[bp+bx]

Ошибка: “Multiple base registers”

Нельзя использовать более одного базового регистра

И более того, это выражение вызовет ошибку несовместимости размеров данных. AX – 2 байта, элемент matr – 1 байт.

mov ax, matr[bp+di+si]

Ошибка: “Multiple index registers”

Нельзя использовать более одного индекс-регистра.

И более того, это выражение вызовет ошибку несовместимости размеров данных. AX – 2 байта, элемент matr – 1 байт.

Выводы.

Были изучены режимы адресации, определены ошибки в программе, и было дано объяснение ошибкам.

ПРОТОКОЛ

Начальные значения регистров:

CS = 30C5, DS=30B0, ES=30B0, SS=30C0

Адрес команды	**Символический код команды**	**16-ричный код команды**	**Содержимое регистров и ячеек памяти**	
			До выполнения	**После выполнения**
0000	PUSH DS	1E	DS= 30B0 SP=0018 STACK=+0|0000 IP=0000	DS= 30B0 SP=0016 STACK=+0| 30B0 IP=0001
0001	SUB AX, AX	2BC0	AX=0000 IP=0001	AX=0000 IP=0003
0003	PUSH AX	50	AX=0000 SP=0016 STACK=+0|30B0 IP=0003	AX=0000 SP=0014 STACK=+0|0000 +2|30B0 IP=0004
0004	MOV AX,30C2	B8C230	AX=0000 IP=0004	AX=30C2 IP=0007
0007	MOV DS, AX	8ED8	AX=30C2 DS= 30B0 IP=0007	AX=30C2 DS=30C2 IP=0009
0009	MOV AX,01F4	B8F401	AX=30C2 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	C7060200CE FF	IP=0012 DS[0002]=00	IP=0018 DS[0002]=CE

			DS[0003]=00	DS[0003]=FF
0018	MOV BX, 0006	BB0600	BX=CE24 IP=0018	BX=0006 IP=001B
001B	MOV [0000], AX	A30000	AX=01F4 IP=001B DS[0000]=00 DS[0001]=00	AX=01F4 IP=001E DS[0000]=F4 DS[0001]=01
001E	MOV AL, [BX]	8A07	AX=01F4 DS[BX]= DS[0006]=01 IP=001E	AX=0101 DS[BX]= DS[0006]=01 IP=0020
0020	MOV AL, [BX+03]	8A4703	AX=0101 DS[BX+03]= DS[0009]=04 IP=0020	AX=0104 DS[BX+03]= DS[0009]=04 IP=0023
0023	MOV CX, [BX+03]	8B4F03	CX=01F4 DS[BX+03]= DS[0009]=04 DS[000A]=08 IP=0023	CX=0804 DS[BX+03]= DS[0009]=04 DS[000A]=08 IP=0026
0026	MOV DI, 0002	BF0200	DI=0000 IP=0026	DI=0002 IP=0029
0029	MOV AL, [000E+DI]	8A850E00	AX=0104 DS[000E+DI]= DS[0010]=0A IP=0029	AX=010A DS[000E+DI]= DS[0010]=0A IP=002D
002D	MOV BX,0003	BB0300	BX=0006 IP=002D	BX=0003 IP=0030
0030	MOV AL, [0016 + BX + DI]	8A811600	AX=010A DS[0016+BX+DI]= DS[001B]=FD IP=0030	AX=01FD DS[0016+BX+DI]= DS[001B]=FD IP=0034
0034	MOV AX,30C2	B8C230	AX=01FD	AX=30C2

			IP=0034	IP=0037
0037	MOV ES, AX	8EC0	ES=30B0 AX=30C2 IP=0037	ES=30C2 AX=30C2 IP=0039
0039	MOV AX, ES:[BX]	268B07	AX=30C2 ES=30C2 ES[BX]=ES[0003]=FF ES[0004]=00 IP=0039	AX=00FF ES=30C2 ES[BX]=ES[0003]=FF ES[0004]=00 IP=003C
003C	MOV AX,0000	B80000	AX=00FF IP=003C	AX=0000 IP=003F
003F	MOV ES, AX	8EC0	ES=30C2 AX=0000 IP=003F	ES=0000 AX=0000 IP=0041
0041	PUSH DS	1E	DS=30C2 SP=0014 STACK=+0|0000 +2|30B0 IP=0041	DS=30C2 SP=0012 STACK=+0|30C2 +2|0000 +4|30B0 IP=0042
0042	POP ES	07	SP=0012 ES=0000 STACK=+0|30C2 +2|0000 +4|30B0 IP=0042	SP=0014 ES=30C2 STACK=+0|0000 +2|30B0 IP=0043
0043	MOV CX, ES:[BX-01]	268B4FFF	CX=0804 ES=30C2 ES[BX-01]= ES[0002]=CE ES[0003]=FF IP=0043	CX=FFCE ES=30C2 ES[BX-01] =ES[0002]=CE ES[0003]=FF IP=0047
0047	XCHG AX, CX	91	AX = 0000 CX = FFCE	AX=FFCE CX=0000

			IP=0047	IP=0048
0048	MOV DI,0002	BF0200	DI=0002 IP=0048	DI=0002 IP=004B
004B	MOV ES:[BX+DI],AX	268901	ES=30C2 ES[BX+DI] = [0005]= 00 ES[0006] = 01 AX=FFCE IP=004B	ES=30C2 ES[0005] = CE ES[0006] = FF IP=004E
004E	MOV BP, SP	8BEC	BP=0000 SP=0014 IP=004E	BP=0014 SP=0014 IP=0050
0050	PUSH [0000]	FF360000	DS[0000] = F4 DS[0001] = 01 SP = 0014 STACK = +0|0000 +2|30B0 IP=0050	DS[0000] = F4 DS[0001] = 01 SP = 0012 STACK= +0|01F4 +2|0000 +4|30B0 IP=0054
0054	PUSH [0002]	FF360200	DS[0002] = CE DS[0003] = FF SP = 0012 STACK=+0|01F4 +2|0000 +4|30B0 IP=0054	DS[0002] = CE DS[0003] = FF SP = 0010 STACK=+0|FFCE +2|01F4 +4|0000 +6|30B0 IP=0058
0058	MOV BP, SP	8BEC	SP=0010 BP=0014 IP=0058	SP=0010 BP=0010 IP=005A
005A	MOV DX, [BP+02]	8B5602	DX=0000 SS[BP+02] = SS[0012]=F4 SS[0013]=01 IP=005A	DX=01F4 SS[BP+02] = SS[0012]=F4 SS[0013] = 01 IP=005D

005D	RET FAR 0002	CA0200	CS=30C5 SP=0010 IP=005D STACK=+0|FFCE +2|01F4 +4|0000 +6|30B0	CS=01F4 SP=0016 IP=FFCE STACK=+0|30B0