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.
Вариант 3

Студент гр. 1383 _____ Куликов М.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Изучение режимов адресации процессора IntelX86 в процессе отладки программы.

Задание.

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.

Была протранслирована программа lab2.asm , содержащая ошибки, был создан диагностический файл lab2.lst.

Было найдено 6 ошибок:

1. В строке «mov mem3,[bx]» - Improper operand type.

Вызвано попыткой перемещения из памяти в память, что запрещено.

2. В строке «mov ax,matr[bx*4][di]» - Illegal register value.

Вызвано тем, что 2-байтовые регистры запрещено масштабировать с набором инструкций 086.

3. В строке «mov ax,matr[bp+bx]» - Mutiple base registers.

Вызвано тем, что запрещено использовать более 1 базового регистра для адресации.

4. «В строке mov ax,matr[bp+di+si]» - Multiple index registers.

Вызвано тем, что запрещено использовать более 1 индексного регистра для адресации.

Протокол отладки программы предоставлен в таблице 1.

Начальные сегментных регистров:

CS = 1A0A

DS = 19F5

SS = 1A05

ES = 19F5

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеер памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	IP = 0000 SP = 0018 STACK + 0 0000 + 2 0000	IP = 0001 SP = 0016 STACK + 0 19F5 +2 0000
0001	SUB AX,AX	2BC0	IP = 0001 AX = 0000	IP = 0003 AX = 0000
0003	PUSH AX	50	IP = 0003 STACK + 0 19F5 +2 0000	IP = 0004 STACK + 0 0000 +2 19F5
0004	MOV AX,1A07	B8071A	IP = 0004	IP = 0007

			AX = 0000	AX = 1A07
0007	MOV DS,AX	8ED8	IP = 0007 DS = 19F5	IP = 0009 DS = 1A07
0009	MOV AX,01F4	B8F401	IP = 0009 AX = 1A07	IP = 000C AX = 01F4
000C	MOV CX,AX	8BC8	IP = 000C CX = 00B0	IP = 000E CS = 01F4
000E	MOV BL,24	B324	IP = 000E BL =00	IP = 0010 BL = 24
0010	MOV BH,CE	B7CE	IP = 0010 BH = 00	IP = 0012 BH = CE
0012	MOV [0002],FFCE	C7060200 CEFF	IP = 0012 DS: 0000 00 00 00 00 00 00	IP = 0018 DS: 0000 00 00 CE FF 00 00
0018	MOV BX,0006	BB0600	IP =0018 BX = CE24	IP = 001B BX = 0006
001B	MOV [0000],AX	A30000	IP = 001B DS: 0000 00 00 CE FF 00 00	IP = 001E DS: 0000 F4 01 CE FF 00 00
001E	MOV AL,[BX]	8A07	IP =001E AL =F4	IP = 0020 AL = 08
0020	MOV AL,[BX+03]	8A4703	IP = 0020 AL =08	IP = 0023 AL =05
0023	MOV CX,[BX+03]	8B4F03	IP =0023 CX = 01F4	IP = 0026 CX = 0105

0026	MOV DI,0002	BF0200	IP = 0026 DI = 0000	IP = 0029 DI = 0002
0029	MOV AL,[000E+DI]	8A850E00	IP = 0029 AL = 05	IP = 002D AL = 1E
002D	MOV CX,[000E+DI]	8B8D0E00	IP = 002D CX = 0105	IP = 0031 CX = 281E
0031	MOV BX,0003	BB0300	IP = 0031 BX = 0006	IP = 0034 BX = 0003
0034	MOV AL,[0016+BX+DI]	8AB11600	IP = 0034 AL = 1E	IP = 0038 AL = 07
0038	MOV CX, [0016+BX+DI]	8B891600	IP = 0038 CX = 281E	IP = 003C CX = 0607
003C	MOV AX,1A07	B8071A	IP = 003C AX = 0107	IP = 003F AX = 1A07
003F	MOV ES,AX	8EC0	IP = 003F ES = 19F5	IP = 0041 ES = 1A07
0041	MOV AX,ES:[BX]	268B07	IP = 0041 AX = 1A07	IP = 0044 AX = 00FF
0044	MOV AX,0000	B80000	IP = 0044 AX = 00FF	IP = 0047 AX = 0000
0047	MOV ES,AX	8EC0	IP = 0047 ES = 1A07	IP = 0049 ES = 0000
0049	PUSH DS	1E	IP = 0049 SP = 0014 STACK +0 0000	IP = 004A SP =0012 STACK +0 1A07

			+2 19F5 +4 0000	+2 0000 +4 19F5
004A	POP ES	07	IP = 004A SP = 0012 ES = 0000 STACK +0 1A07 +2 0000 +4 19F5	IP = 004И SP = 0014 ES = 1A07 STACK +0 0000 +2 19F5 +4 0000
004B	MOV CX,ES:[BX-01]	268B4FFF	IP = 004B CX = 0607	IP = 004F CX = FFCE
004F	XCHG AX,CX	91	IP =004F AX = 0000 CX = FFCE	IP = 0050 AX = FFCE CX = 0000
0050	MOV DI,0002	BF0200	IP = 0050 DI = 0002	IP = 0053 DI = 0002
0053	MOV ES:[BX+DI],AX	268901	IP = 0053 DS : 0000 F4 01 CE FF 00 00 00	IP = 0056 DS : 0000 F4 01 CE FF 00 CE FF
0056	MOV BP,SP	8BEC	IP = 0056 BP = 0000	IP = 0058 BP = 0014
0058	PUSH [0000]	FF360000	IP = 0058 SP = 0014 STACK +0 0000 +2 19F5 +4 0000	IP = 005C SP = 0012 STACK +0 01F4 +2 0000 +4 19F5

005C	PUSH [0002]	FF360200	IP = 005C SP = 0012 STACK +0 01F4 +2 0000 +4 19F5 +6 0000	IP = 0060 SP = 0010 STACK +0 FFCE +2 01F4 +4 0000 +6 19F5
0060	MOV BP,SP	8BEC	IP = 0060 BP = 0014	IP = 0062 BP = 0010
0062	MOV DX,[SP+02]	8B5602	IP = 0062 DX = 0000	IP = 0065 DX = 01F4
0065	RET FAR 0002	CA0200	IP = 0065 SP = 0010 STACK +0 FFCE +2 01F4 +4 0000 +6 19F5	IP = FFCE SP = 0016 STACK +0 19F5

Выводы.

В ходе лабораторной работы была изучена работа режимов адресации процессора IntelX86.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
; 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 8,7,6,5,1,2,3,4
vec2 DB -30,-40,30,40,-10,-20,10,20
matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-8,4,3,2,1
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 Б

ДИАГНОСТИЧЕСКИЙ ФАЙЛ

```
Microsoft (R) Macro Assembler Version 5.10              10/15/22 17:07:1
                                                        Page     1-1


                        ;       PµCЂPsPiCЂP°PjPjP°      PëP·CґC‡PµPSPëCЏ
CЂPµP¶PëP
                        jPsPI                           P°PґCЂPµCЃP°C†PëPë
PїCЂPsC†PµCЃCЃPsCЂP° I
                        ntelX86
        = 0024                  EOL EQU '$'
        = 0002                  ind EQU 2
        = 01F4                  n1 EQU 500
        =-0032                  n2 EQU -50

                        ; PЎC,PµPє PїCЂPsPiCЂP°PjPjC‹
        0000            AStack SEGMENT STACK
        0000  000C[             DW 12 DUP(?)
                ????
                            ]

        0018            AStack ENDS

                        ; P”P°PSPSC‹Pµ PїCЂPsPiCЂP°PjPjC‹
        0000            DATA SEGMENT

                        ;       P”PëCЂPµPєC,PëPIC‹      PsPїPëCЃP°PSPëCЏ
PґP°PSPSC
```

```
                        ‹C…
0000  0000               mem1 DW 0
0002  0000               mem2 DW 0
0004  0000               mem3 DW 0
0006  08 07 06 05 01 02 vec1 DB 8,7,6,5,1,2,3,4
    03 04
000E  E2 D8 1E 28 F6 EC      vec2 DB -30,-40,30,40,-10,-20,10,20
    0A 14
0016  FF FE FD FC 08 07      matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-8,4,3,2,1

    06 05 FB FA F9 F8
    04 03 02 01
0026                    DATA ENDS

                        ; РљРsРґ РїСЂРsРiСЂР°РjРjС‹
0000                    CODE SEGMENT
                        ASSUME CS:CODE, DS:DATA, SS:AStack

                        ; Р“РsР»РsРIРSР°СЏ РїСЂРsС†РµРґСѓСЂР°
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
                        ;          РџР РћР’Р•Р РљРђ          Р Р•Р–Р   РњРћР’
РђР”Р Р•РЎРђР
                        ¦Р   Р    РќРђ РJР РћР’РќР• РЎРњР•Р©Р•РќР   Р™
                        ;                              Р РµРiРёСЃ‚С‚РЂРsРIР°СЏ
Р°РґСЂРµСЃР°С†РёСЏ
```

```
0009  B8 01F4                   mov ax,n1
000C  8B C8                     mov cx,ax
000E  B3 24                     mov bl,EOL
0010  B7 CE                     mov bh,n2
                        ; PµCЂCЏPjP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0012  C7 06 0002 R FFCE         mov mem2,n2
Microsoft (R) Macro Assembler Version 5.10          10/15/22 17:07:1
                                                    Page     1-2


0018  BB 0006 R                 mov bx,OFFSET vec1
001B  A3 0000 R                 mov mem1,ax
                        ; PљPsCЃPIPµPSPSP°CЏ P°PrCЂPµCЃP°C†PëCЏ
001E  8A 07                     mov al,[bx]
                                ;mov mem3,[bx]
                        ;                       P‘P°P·PëCЂPsPIP°PSPSP°CЏ
P°PrCЂPµCЃP°C†PëCЏ
0020  8A 47 03                  mov al,[bx]+3
0023  8B 4F 03                  mov cx,3[bx]
                        ; P   PSPrPµPєCЃPSP°CЏ P°PrCЂPµCЃP°C†PëCЏ
0026  BF 0002                   mov di,ind
0029  8A 85 000E R              mov al,vec2[di]
002D  8B 8D 000E R              mov cx,vec2[di]
      LAB2.ASM(54): warning A4031: Operand types must match
                        ;               PђPrCЂPµCЃP°C†PëCЏ          CЃ
P±P°P·PëCЂPsPIP°PSPëPµP
                        j Pë PëPSPrPµPєCЃPëCЂPsPIP°PSPëPµPj
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]
```

```
LAB2.ASM(58): warning A4031: Operand types must match
                                ;mov ax,matr[bx*4][di]
                        ;       РџР РћР’Р•Р РЉРЋ        Р Р•Р–Р  РњРћР’
РЋР”Р Р•РЎРЋР
                        ¦Р  Р   РЎ РЈРЈР§Р•РўРћРњ РЎР•Р“РњР•РќРўРћР’
                        ;           РџРµСЂРµРѕРїСЂРµРґРµР»РµРЅРёРµ
СЃРµРіРјРµРЅС,
                        Р°
                        ; ------ РІР°СЂРёР°РЅС, 1
003C  B8 ---- R                 mov ax, SEG vec2
003F  8E C0                             mov es, ax
0041  26: 8B 07                 mov ax, es:[bx]
0044  B8 0000                           mov ax, 0
                        ; ------ РІР°СЂРёР°РЅС, 2
0047  8E C0                             mov es, ax
0049  1E                        push ds
004A  07                        pop es
004B  26: 8B 4F FF                      mov cx, es:[bx-1]
004F  91                        xchg cx,ax
                        ; ------ РІР°СЂРёР°РЅС, 3
0050  BF 0002                           mov di,ind
0053  26: 89 01                 mov es:[bx+di],ax
                        ; ------ РІР°СЂРёР°РЅС, 4
0056  8B EC                             mov bp,sp
                                ;mov ax,matr[bp+bx]
                                ;mov ax,matr[bp+di+si]
                        ;                       Р  СЃРїРѕР»СЊР·РѕРІР°РЅРёРµ
СЃРµРіРјРµРЅС,Р° С
                        ЃС,РµРєР°
0058  FF 36 0000 R                     push mem1
```

```
005C  FF 36 0002 R                push mem2
0060  8B EC                       mov bp,sp
0062  8B 56 02                   mov dx,[bp]+2
0065  CA 0002                     ret 2
0068                    Main ENDP
0068                    CODE ENDS
                              END Main
```


Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA		STACK
CODE	0068	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 = 0068
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  LAB2
@VERSION . . . . . .    TEXT  510


    88 Source  Lines
    88 Total   Lines
    19 Symbols

 47812 + 459448 Bytes symbol space free

     2 Warning Errors
     0 Severe  Errors
```