

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 и формирования
исполнительного адреса
Вариант 6

Студентка гр.1381

Рымарь М.И.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

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.

Выполнение работы.

1. В соответствии с номером студенческого билета, был получен вариант набора значений исходных данных (массивов) `vec1`, `vec2` и `matr` из файла `lr2.dat`, приведённый в каталоге Задания. Были занесены свои данные вместо значений, указанных в приведённой программе.

Вариант 6:

`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 протранслирована с созданием файла диагностических сообщений. Были обнаружены следующие ошибки и закомментированы:

1). lab2.asm(48): error A2052: Improper operand type

mov mem3,[bx] – одной командой нельзя читать из памяти и писать в память одновременно;

2). lab2.asm(55): warning A4031: Operand types must match

mov cx,vec2[di] – у операндов разные типы, cx – двухбайтовое слово, vec2[di] – однобайтовый элемент;

3). lab2.asm(59): warning A4031: Operand types must match

mov cx,matr[bx][di] - у операндов разные типы, cx – двухбайтовое слово, matr[bx][di] – однобайтовый элемент;

4). lab2.asm(60): error A2055: Illegal register value

mov ax,matr[bx*4][di] – нельзя одновременно изменять регистр и обращаться к элементу, нужно сделать умножение отдельной операцией;

5). lab2.asm(80): error A2046: Multiple base registers

mov ax,matr[bp+bx] – нельзя использовать более одного базового регистра для адресации;

6). lab2.asm(81): error A2047: Multiple index registers

mov ax,matr[bp+di+si] – нельзя использовать более одного индексного регистра;

7). lab2.asm(88): Phase error between passes

Main ENDP – в Main содержатся ошибки, эту строку не нужно комментировать.

3. Программа была повторно протранслирована, был скомпонован загрузочный модуль.

4. Далее программа была выполнена в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения команды. Результаты выполнения программы приведены в таблице 1.

Начальные значения:

SP=0018

IP = 0000

DS = 119C

CX = 00B0

Адрес команды	Символьный код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(SP) = 0018 (IP) = 0000 (DS) = 119C Stack: +0 0000	(SP) = 0016 (IP) = 0001 (DS) = 119C Stack: +0 119C
0001	SUB AX, AX	2BC0	(AX) = 0000 (IP) = 0001	(AX) = 0000 (IP) = 0003
0003	PUSH AX	50	(AX) = 0000 (SP) = 0016 (IP) = 0003 Stack: +0 119C +2 0000	(AX) = 0000 (SP) = 0014 (IP) = 0004 Stack: +0 0000 +2 119C
0004	MOV AX, 11AE	B8AE11	(AX) = 0000 (IP) = 0004	(AX) = 11AE (IP) = 0007
0007	MOV DS, AX	8ED8	(AX) = 11AE (DS) = 119C (IP) = 0007	(AX) = 11AE (DS) = 11AE (IP) = 0009
0009	MOV AX, 01F4	B8F401	(AX) = 11AE (IP) = 0009	(AX) = 01F4 (IP) = 000C
000C	MOV CX, AX	8BC8	(AX) = 01F4 (IP) = 000C (CX) = 00B0	(AX) = 01F4 (IP) = 000E (CX) = 01F4
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	(IP) = 0018
0018	MOV BX, 0006	BB0600	(BX) = CE24 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	MOV [0000], AX	A30000	(AX) = 01F4 (IP) = 001B	(AX) = 01F4 (IP) = 001E
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, [DI+000E]	8A850E00	(AX) = 010F (IP) = 0029	(AX) = 01E2 (IP) = 002D
002D	MOV BX, 0003	BB03000	(IP) = 002D (BX) = 0006	(IP) = 0030 (BX) = 0003

0030	MOV AL, [BX+DI+0016]	8A811600	(IP) = 0030 (AX) = 01E2	(IP) = 0034 (AX) = 01FF
0034	MOV AX, 11AE	B8AE11	(AX) = 01FF (IP) = 0034	(AX) = 11AE (IP) = 0037
0037	MOV ES, AX	8EC0	(AX) = 11AE (ES) = 119C (IP) = 0037	(AX) = 11AE (ES) = 11AE (IP) = 0039
0039	MOV AX, ES:[BX]	268B07	(AX) = 11AE (IP) = 0039	(AX) = 00FF (IP) = 003C
003C	MOV AX, 0000	B80000	(AX) = 00FF (IP) = 003C	(AX) = 0000 (IP) = 003F
003F	MOV ES, AX	8EC0	(ES) = 11AE (AX) = 0000 (IP) = 003F	(ES) = 0000 (AX) = 0000 (IP) = 0041
0041	PUSH DS	1E	(DS) = 11AE (SP) = 0014 (IP) = 0041 Stack: +0 0000 +2 119C +4 0000	(DS) = 11AE (SP) = 0012 (IP) = 0042 Stack: +0 11AE +2 0000 +4 119C
0042	POP ES	07	(SP) = 0012 (IP) = 0042 (ES) = 0000 Stack: +0 11AE +2 0000 +4 119C	(SP) = 0014 (IP) = 0043 (ES) = 11AE Stack: +0 0000 +2 119C +4 0000
0043	MOV CX, ES:[BX-01]	268B4FFF	(CX) = 0B0F (IP) = 0043	(CX) = FFCE (IP) = 0047
0047	XCHG AX, CX	91	(AX) = 0000 (CX) = FFCE (IP) = 0047	(AX) = FFCE (CX) = 0000 (IP) = 0048
0048	MOV DI, 0002	BF0200	(DI) = 0002 (IP) = 0048	(DI) = 0002 (IP) = 004B
004B	MOV ES:[BX+DI], AX	268901	(AX) = FFCE (IP) = 004B	(AX) = FFCE (IP) = 004E
004E	MOV BP, SP	8BEC	(BP) = 0000 (SP) = 0014 (IP) = 004E	(BP) = 0014 (SP) = 0014 (IP) = 0050
0050	PUSH [0000]	FF360000	(SP) = 0014 (IP) = 0050 Stack: +0 0000 +2 119C +4 0000	(SP) = 0012 (IP) = 0054 Stack: +0 01F4 +2 0000 +4 119C
0054	PUSH [0002]	FF360200	(SP) = 0012 (IP) = 0054 Stack: +0 01F4 +2 0000 +4 119C +6 0000	(SP) = 0010 (IP) = 0058 Stack: +0 FFCE +2 01F4 +4 0000 +6 119C
0058	MOV BP, SP	8BEC	(BP) = 0014 (SP) = 0010	(BP) = 0010 (SP) = 0010

			(IP) = 0058	(IP) = 005A
005A	MOV DX, [BP+02]	8B5602	(DX) = 0000 (IP) = 005A	(DX) = 005D (IP) = 005D
005D	RET Far 002	CA0200	(IP) = 005D (SP) = 0010 (CS) = 11B1 Stack: +0 FFCE +2 01F4 +4 0000 +6 119C	(IP) = FFCE (SP) = 0016 (CS) = 01F4 Stack: +0 119C +2 0000 +3 0000 +4 0000

Таблица 1 – Результаты прогона программы в отладчике

Выводы.

В ходе выполнения лабораторной работы были изучены режимы адресации и формирование исполнительного запроса.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММ

Название файла: *lab2.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 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

```

Название файла: *lab2.lst*

Microsoft (R) Macro Assembler Version 5.10
01:10:22

10/9/22

Page 1-1

```

; Программа
изучения режимов адресации процессора 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 даннх
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

; Код программы
0000      CODE SEGMENT
            ASSUME CS:CODE,
DS:DATA, SS:AStack

```

процедура		; Головная
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	
Microsoft (R) Macro Assembler Version 5.10		10/9/22
01:10:22		
		Page 1-2
0018 BB 0006 R	mov bx,OFFSET vec1	
001B A3 0000 R	mov mem1,ax	
	; Косвенная	
адресация		
001E 8A 07	mov al,[bx]	
	; 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	
базированием 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
АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ		; Переопределение
сегмент		
	a	
	; ----- 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 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
01:10:22
                        10/9/22
                        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 lab2
@VERSION	TEXT 510

90 Source Lines
 90 Total Lines
 19 Symbols

47828 + 459432 Bytes symbol space free

0 Warning Errors
 0 Severe Errors