

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.

Студент гр. 1303

Бутыло Е.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить работу с режимами адресации на языке программирования Ассемблер.

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.

Выполнение работы

1. У преподавателя получен вариант набора значений исходных данных (массивов) `vec1`, `vec2` и `matr` из файла `lr2.dat`, приведенного в каталоге Задания и свои данные занесены вместо значений, указанных в приведенной ниже программе.

2. Программа протранслирована с созданием файла диагностических сообщений; обнаруженные ошибки объяснены и закомментированы соответствующие операторы в тексте программы.

```
source _comp.asm(41): error A2502: Improper operand type  
mov mem3,[bx]
```

Машинные команды не могут манипулировать одновременно двумя операндами, находящимися в оперативной памяти, то есть в команде только 1 операнд может указывать на ячейку памяти, другой операнд должен быть либо регистром, либо непосредственным значением.

```
source _comp.asm(43): warning A4001: Extra characters on line 7  
Лишний нелогичный символ.  
source _comp.asm(49): warning A4031: Operand types must match  
mov cx,vec2[di]
```

Разные типы операндов, `cx` – слово, а `vec2[di]` – размерность 1 байт

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

```
mov cx,matr[bx][di]
```

Разные типы операндов, cx – слово, а matr[bx][di] – размерность 1 байт

source _comp.asm(54): error A2055: Illegal register value

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

В непосредственной адресации с базированием и индексированием для вычисления исполнительного адреса берется сумма базового и индексного регистра, к которым добавляется непосредственно фигурирующее в команде смещение. Там не фигурирует умножение.

source _comp.asm(73): error A2046: Multiple base register

```
mov ax,matr[bp+bx]
```

В косвенной адресации с индексированием исполнительный адрес берется в виде суммы адресов, находящихся в базовом и индексном регистрах, а в данной строке оба регистра базовые.

source _comp.asm(74): error A2047: Multiple index register

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

В непосредственной адресации с базированием и индексированием берется сумма базового и индексного регистра, к которым добавляется непосредственно фигурирующее в команде смещение, а в данной строке фигурируют 2 индексных регистра и 1 базовый.

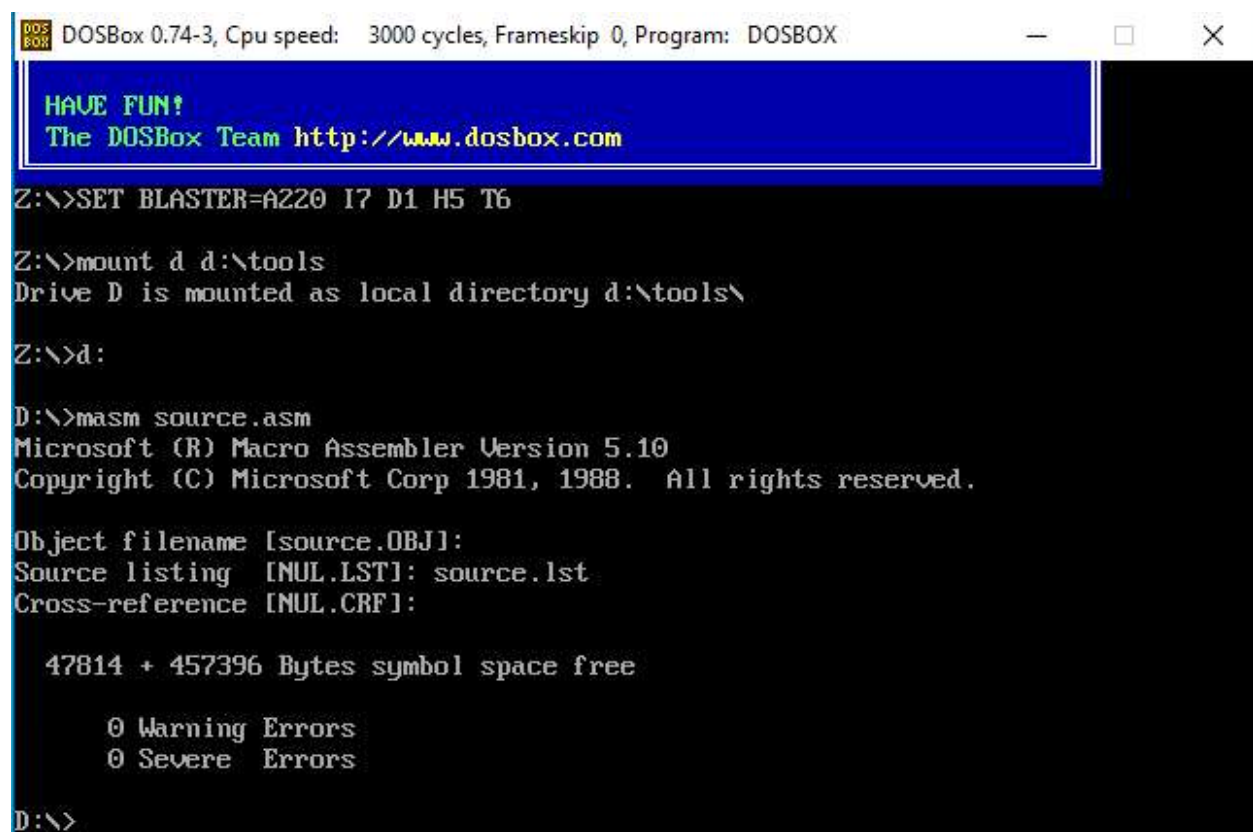
source _comp.asm(81): error A2006: Phase error between passes

```
Main ENDP
```

Ошибка говорит о том, что в функции Main допущены ошибки.

3. Снова протранслирована программа и скомпонован загрузочный модуль.

Трансляция программы после исправления ошибок



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount d d:\tools
Drive D is mounted as local directory d:\tools\

Z:\>d:

D:\>masm source.asm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [source.OBJ]:
Source listing [NUL.LST]: source.lst
Cross-reference [NUL.CRF]:

47814 + 457396 Bytes symbol space free

0 Warning Errors
0 Severe Errors

D:\>
```

4. Программа выполнена в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения команды.

source.exe

Адрес команды	Символический код команды	16-ричный код команды	Изменяемые данные	
			до	после
0000	PUSH DS	1E	STACK(+0)=0000 IP = 0000 SP=0018	STACK(+0)=19F5 IP = 0001 SP=0016
0001	SUB AX, AX	2BC0	AX=0000	AX=0000

			IP = 0001	IP = 0003
0003	PUSH AX	50	STACK(+0)=19F5 STACK(+2)=0000 IP = 0003 SP=0016	STACK(+0)=0000 STACK(+2)=19F5 IP = 0004 SP=0014
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	B8F401	AX = 1A07 IP = 0009	AX = 01F4 IP = 000C
000C	MOV CX,AX	8BC8	IP=000C CX=00B0	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	C7060200C EFF	IP = 0012	IP = 0018

0018	MOV BX,0006	BB0600	BX = CE24 IP = 0018	BX = 0006 IP = 001B
001B	MOV [0000],AX	A30000	IP = 001B	IP = 001E
001E	MOV AL,[BX]	8A07	AX = 01F4 IP = 001E	AX = 0101 IP = 0020
0020	MOV AL,[BX+03]	8A4703	IP = 0020 AX = 0101	IP = 0023 AX = 0104
0023	MOV CX, [BX+03]	8B4F03	CX = 01F4 IP = 0023	CX = 0804 IP = 0026
0026	MOV DI, 0002	BF0200	DI = 0002 IP = 0026	DI = 0002 IP = 0029
0029	MOV AL, [000E+DI]	8A850E00	AX = 0104 IP = 0029	AX = 010A IP = 002D
002D	MOV BX, 0003	BB0300	IP = 002D BX = 0006	IP = 0030 BX = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	IP = 0030 AX = 010A	IP = 0034 AX = 01FD
0034	MOV AX, 1A07	B8071A	AX = 01FD	AX = 1A07

			IP = 0034	IP = 0037
0037	MOV ES, AX	8EC0	ES = 19F5 IP= 0037	ES = 1A07 IP= 0039
0039	MOV AX, ES:[BX]	268B07	AX = 1A07 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 = 1A07 IP= 003F	ES= 0000 IP= 0041
0041	PUSH DS	1E	IP= 0041 SP= 0014 STACK (+0) = 0000 STACK (+2) =19F5 STACK (+4) =0000	IP= 0042 SP= 0012 STACK(+0)=1A07 STACK (+2)=0000 STACK (+4)=19F5
0042	POP ES	07	SP= 0012 ES=0000 IP= 0042 STACK (+0) = 1A07 STACK (+2) = 0000 STACK (+4) =19F5	SP = 0014 ES=1A07 IP= 0043 STACK (+0)=0000 STACK (+2)=19F5 STACK (+4)=0000

0043	MOV CX, ES:[BX—01]	268B4FFF	CX = 0804 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	IP = 0048 DI=0002	IP = 004B DI=0002
004B	MOV ES:[BX+DI], AX	268901	IP = 004B	IP = 004E
004E	MOV BP, SP	8BEC	IP = 004E BP = 0010	IP = 0050 BP = 0014
0050	PUSH [0000]	FF360000	IP = 0050 SP=0014 STACK (+0) = 0000 STACK (+2) = 19F5 STACK (+4) =0000	IP = 0054 SP=0012 STACK (+0) = 01F4 STACK (+2) = 0000 STACK (+4) =19F5
0054	PUSH [0002]	FF360200	IP = 0054 SP = 0012 STACK (+0) = 01F4 STACK (+2) = 0000 STACK (+4) =19F5 STACK (+6) = 0000	IP = 0058 SP = 0010 STACK (+0) = FFCE STACK (+2) = 01F4 STACK (+4) =0000 STACK (+6) = 19F5

0058	MOV BP, SP	8BEC	IP = 0058 BP = 0014	IP = 005A BP = 0010
005A	MOV DX, [BP+02]	8B5602	IP = 005A DX = 01F4	IP = 005D DX = 01F4
005D	RET Far 0002	CA0200	IP = 005D SP = 0010 CS=1A0A STACK (+0) = FFCE STACK (+2) = 01F4 STACK (+4) = 0000 STACK (+6) = 19F5	IP = FFCE SP= 0016 CS=01F4 STACK (+0) = 19F5 STACK (+2) = 0000 STACK (+4) = 0000 STACK (+6) = 0000

Выводы

В ходе выполнения лабораторной работы были получены основные навыки работы с режимами адресации на языке программирования Ассемблер.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: source.asm

```
; Учебная программа N2  цикла лаб.раб. по дисциплине
;   "Организация и функционирование ЭВМ"
;
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 vec2
           mov    mem1,ax

;   Косвенная адресация
           mov    al,[bx]

;           mov    mem3,[bx] ----- некорректная конструкция
;   Базированная адресация
           mov    al,[bx]+3
           mov    cx,3[bx]

;   Индексированная адресация
           mov    di,ind
           mov    al,vec2[di]

;           mov    cx,vec2[di] ----- некорректная конструкция
;   Адресация с базированием и индексированием
           mov    bx,3
           mov    al,matr[bx][di]

;           mov    cx,matr[bx][di] ----- некорректная конструкция
;           mov    ax,matr[bx*4][di] ----- некорректная конструкция

;   ПРОВЕРКА 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 ax, matr[bp+bx] ----- некорректная конструкция
;        mov ax, matr[bp+di+si] ----- некорректная конструкция
; Использование сегмента стека
        push mem1
        push mem2
        mov bp, sp
        mov dx, [bp]+2
        ret 2
Main     ENDP
CODE     ENDS
        END Main

```

Название файла: source.lst

Microsoft (R) Macro Assembler Version 5.10

10/4/22 16:23:47

Page

1-1

```

; Учебная программа N2  цикИ
»а лаб.раб. по дисциплине
; "Организация и функцион
ирование ЭВМ"
;

```

```

= 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 данных

```
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
```

; 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

Microsoft (R) Macro Assembler Version 5.10

10/4/22 16:23:47

Page

1-2

ЦИИ 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 000E R           mov   bx,OFFSET vec2
001B  A3 0000 R           mov   mem1,ax

```

; Косвенная адресация

```

001E  8A 07              mov   al,[bx]      ; ---

```

записывѣ

°ем значение лежащее по ад

ресу bx

```

;          mov   mem3,[bx]          -----

```

----- некорректнѣ

°ая конструкция

; Базированная адресация

```

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

```

; Индексированная адресаѣ

ия

```

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

```

```

;          mov  cx,vec2[di]          -----
----- некорректный
°ая конструкция
;  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]      -----
----- некорректный
°ая конструкция
;          mov  ax,matr[bx*4][di]    -----
----- некорректный
°ая конструкция

;  ПРОВЕРКА АДРЕСАЦИИ С УЧЁТ
ТОМ 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

```

1-3

```

0048 BF 0002                                mov di,ind
004B 26: 89 01                            mov es:[bx+di],ax
; ----- вариант 4
;          mov ax,matr[bp+bx]             -----
----- некорректный
ая конструкция
;          mov ax,matr[bp+di+si]          -----
----- некорректный
ая конструкция
;  Использование сегмента
стека
004E FF 36 0000 R                          push mem1
0052 FF 36 0002 R                          push mem2
0056 8B EC                                mov bp,sp
0058 8B 56 02                             mov dx,[bp]+2
005B CA 0002                             ret 2
005E                                Main      ENDP
005E                                CODE     ENDS
                                END Main

```

Symbols-1

Segments and Groups:

Class	N a m e	Length	Align	Combine
	ASTACK	0018	PARA	STACK
	CODE	005E	PARA	NONE
	DATA	0026	PARA	NONE

Symbols:

N a m e	Type	Value	Attr
EOL	NUMBER	0024	
IND	NUMBER	0002	
MAIN Length = 005E	F PROC	0000	CODE
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	source	
@VERSION	TEXT	510	

96 Source Lines

96 Total Lines

19 Symbols

47814 + 457396 Bytes symbol space free

0 Warning Errors

0 Severe Errors