

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.

Студентка гр. 1381

Новак П. И.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Развитие навыков работы 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.

Ход 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.

```
lr2_comp.asm(42): 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.

```
lr2_comp.asm(44): warning A4001: Extra characters on line  
7
```

Лишний нелогичный символ.

```
lr2_comp.asm(50): warning A4031: Operand types must match  
mov cx,vec2[di]
```

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

lr2_comp.asm(54): warning A4031: Operand types must match

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

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

lr2_comp.asm(55): error A2055: Illegal register value

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

Нельзя использовать масштабирование без подключения директивы .286.

lr2_comp.asm(74): 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.

lr2_comp.asm(75): 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.

lr2_comp.asm(82): 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 lr2_comp.asm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [lr2_comp.OBJ]:
Source listing [NUL.LST]: lr2_comp.lst
Cross-reference [NUL.CRF]:

47800 + 459460 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.

lr2_comp.asm

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 IP = 0001	AX=0000 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	C7060200CEFF	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=0115 IP=0020
0020	MOV AL,[BX+03]	8A4703	IP= 0020 AX=0115	IP=0023 AX=0118
0023	MOV CX, [BX+03]	8B4F03	CX= 01F4 IP = 0023	CX = 1C18 IP= 0026
0026	MOV DI, 0002	BF0200	DI= 0000 IP= 0026	DI= 0002 IP= 0029
0029	MOV AL, [000E+DI]	8A850E00	AX= 0118 IP = 0029	AX= 01D8 IP= 002D
002D	MOV BX, 0003	BB0300	IP = 002D BX = 0006	IP = 0030 BX = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	IP = 0030	IP = 0034

			AX =01D8	AX =0108
0034	MOV AX, 1A07	B8071A	AX= 0108 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 = 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 = 1C18 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	IP = 005A

			BP = 0003	BP = 0010
005A	MOV DX, [BP+02]	8B5602	IP = 005A DX = 0000	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

5. Результаты прогона программы под управлением отладчика представлены в отчете.

Выводы.

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

Приложение А. Код программ.

Имя файла: lr2_comp.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 21,22,23,24,28,27,26,25

vec2 DB 40,50,-40,-50,20,30,-20,-30

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

; 7 .44 this лишний символ

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. Листинг успешной трансляции
программами.**

Имя файла: lr2_comp.asm

```
                                ;      PµCḡPsPiCḡP°PjPjP°      PëP·CrC‡PµPSPëCḡ
CḡPµP¶PëP

                                jPsPI                                P°PrCḡPµCḡP°C‡PëPë
PiCḡPsC‡PµCḡCḡPsCḡP° I

                                ntelX86

= 0024                        EOL EQU '$'

= 0002                        ind EQU 2

= 01F4                        n1 EQU 500

=-0032                        n2 EQU -50

                                ; PŸC,PµPë PiCḡPsPiCḡP°PjPjC<

0000                          AStack SEGMENT STACK

0000 000C[                     DW 12 DUP(?)

                                ????

                                ]

0018                          AStack ENDS

                                ; P”P°PSPSC<Pµ PiCḡPsPiCḡP°PjPjC<

0000                          DATA SEGMENT

                                ;      P”PëCḡPµPëC,PëPIC<      PsPiPëCḡP°PSPëCḡ
```

PrP°PSPSC

⋈C...

0000 0000 mem1 DW 0

0002 0000 mem2 DW 0

0004 0000 mem3 DW 0

0006 15 16 17 18 1C 1B vec1 DB 21,22,23,24,28,27,26,25

1A 19

000E 28 32 D8 CE 14 1E vec2 DB 40,50,-40,-50,20,30,-20,-30

EC E2

0016 05 06 F8 F9 07 08matr DB 5,6,-8,-7,7,8,-6,-5,1,2,-4,-3,3,4,-2,-1

FA FB 01 02 FC FD

03 04 FE FF

0026 DATA ENDS

; PљPsPr PiCћPsPiCћP°PjPjC⋈

0000 CODE SEGMENT

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

; P“PsP»PsPIPSP°CЀ PiCћPsC†PμPrCѓCћP°

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

; P₄P P_hP'P•P P₇P_h P P•P–P P_hP_hP' P_hP''P P•P_YP_hP

|P P P_kP_h PJP P_hP'P_kP• P_YP_hP•P©P•P_kP P™

; P P_μPiP_eC_fC,C_hP_sPIP°C P°PrC_hP_μC_fP°C†P_eC

0009 B8 01F4 mov ax,n1

000C 8B C8 mov cx,ax

000E B3 24 mov bl,EOL

0010 B7 CE mov bh,n2

; P₄C_hC_hPjP°C P°PrC_hP_μC_fP°C†P_eC

0012 C7 06 0002 R FFCE mov mem2,n2

0018 BB 0006 R mov bx,OFFSET vec1

001B A3 0000 R mov mem1,ax

; P₇P_sC_fPIP_μPSPSP°C P°PrC_hP_μC_fP°C†P_eC

001E 8A 07 mov al,[bx]

mov mem3,[bx]

lr2a.asm(42): error A2052: Improper operand type

; P°P•P•P_eC_hP_sPIP°PSPSP°C

P°PrC_hP_μC_fP°C†P_eC

; 7.44 this P»PëC€PSPëPNº CÍPëPjPIPsP»

0020 8A 47 03 mov al,[bx]+3

0023 8B 4F 03 mov cx,3[bx]

; P PPSPrPµPëCÍPSP°C P°PrCṪPµCÍP°C†PëC P

0026 BF 0002 mov di,ind

0029 8A 85 000E R mov al,vec2[di]

002D 8B 8D 000E R mov cx,vec2[di]

lr2a.asm(50): warning A4031: Operand types must match

; PḣPrCṪPµCÍP°C†PëC P CÍ

P±P°P·PëCṪPsPIP°PSPëPµP

j Pë PëPSPPrPµPëCÍPëCṪPsPIP°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]

lr2a.asm(54): warning A4031: Operand types must match

003C 8B 85 0022 R mov ax,matr[bx*4][di]

lr2a.asm(55): error A2055: Illegal register value

; P P PḣP’P·P P Pḣ P P·P–P PḣPḣP’ PḣP”P P·PŸPḣP

‘P P PŸ PJP§P·PŸPḣPḣ PŸP·P“PḣP·P PŸPḣP’

;
CÍPµPiPjPµPSC,

P°

; ----- PIP°CḡPëP°PSC, 1

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

; ----- PIP°CḡPëP°PSC, 2

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

; ----- PIP°CḡPëP°PSC, 3

0054 BF 0002 mov di,ind

0057 26: 89 01 mov es:[bx+di],ax

; ----- PIP°CḡPëP°PSC, 4

005A 8B EC mov bp,sp

005C 3E: 8B 86 0016 R mov ax,matr[bp+bx]

lr2a.asm(74): error A2046: Multiple base registers

0061 3E: 8B 83 0016 R mov ax,matr[bp+di+si]

lr2a.asm(75): error A2047: Multiple index registers

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 PROC	0000	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 lr2a

@VERSION TEXT 510

84 Source Lines

84 Total Lines

19 Symbols

47828 + 459432 Bytes symbol space free

2 Warning Errors

5 Severe Errors