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. 9382 _____ Пя С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить режимы адресации на примере готовой программы lr2_comp.asm на Ассемблере, объяснить неправильность допущенных ошибок и откорректировать код.

Задание:

5 Вариант

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.

Ход работы:

Часть 1.

Файл диагностических сообщений корректный в приложении А.

Пояснения имеющихся неправильных конструкций:

mov mem3, [bx]

LR2.ASM(42): error A2052: Improper operand type

Неверный тип операндов

Не получится одновременно считывать из памяти и вписывать в нее.

mov cx, vec2[di]
LR2.ASM(49): warning A4031: Operand types must match
Типы операндов должны совпадать
Однако размер элементов массива ‘vec2’ - 1 байт, а ‘cx’ - 2 байта

mov cx, matr[bx][di]
LR2.ASM(53): warning A4031: Operand types must match
Типы операндов должны совпадать
Размер элементов матрицы ‘matr' - 1 байт, а ‘cx' - 2 байта

mov ax, matr[bx*4][di]
LR2.ASM(54): error A2055: Illegal register value
Незаконное значение регистра
Bx – 16-битный регистр, на который нельзя умножать
Типы операндов должны совпадать
Размер элементов матрицы ‘matr' - 1 байт, а ‘ax' - 2 байта

mov ax, matr[bp+bx]
LR2.ASM(73): error A2046: Multiple base registers
Несколько базовых регистров
Нельзя использовать несколько базовых регистров
Типы операндов должны совпадать
Размер элементов матрицы ‘matr' - 1 байт, а ‘ax' - 2 байта

mov ax, matr[bp+di+si]
LR2.ASM(74): error A2047: Multiple index registers
Несколько индексных регистров
Нельзя использовать несколько индексных регистров

Слишком много регистров

Нельзя использовать более двух регистров

Типы операндов должны совпадать

Размер элементов матрицы ‘matr' - 1 байт, а ‘ax' - 2 байта

push mem1

push mem2

семантическая ошибка: чтобы завершить программу и вернуться в DOS нужно, чтобы вершина стека имела смещение и сегмент начала PSP. Однако при выполнении вышеуказанных команд она имеет mem2 и mem1, при выполнении ret 2 программа не сможет правильно завершиться.

Содержимое сегментных регистров до старта программы: CS:1A0A, DS:19F5, ES:19F5, SS:1A05, HS:19F5, FS:19F5

Табл.1.

Адрес Команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(SP) = 0018 (IP) = 0000 Stack +0 0000 +2 0000 +4 0000	(SP)=0016 (IP) = 0001 Stack +0 19F5 +2 0000 +4 0000
0001	SUB AX, AX	2BC0	(IP) = 0001	(IP) = 0003
0003	PUSH AX	50	(SP)=0016 (IP) = 0003 Stack +0 19F5 +2 0000 +4 0000	(SP) = 0014 (IP) = 0004 Stack +0 0000 +2 19F5 +4 0000
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	(CX) = 00A8 (IP) = 000C	(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) = 000E	(BX) = CE24 (IP) = 0012
0012	MOV [0002], FFCE	C7060200CEFF	(IP) = 0012	(IP) = 0018
0018	MOV BX, 0006	BB0600	(BX) = 0000 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	MOV [0000], AX	A30000	(IP) = 001B	(IP) = 001E
001E	MOV AL, [BX]	8AO7	(AX) = 01F4 (IP) = 001E	(AX) = 010B (IP) = 0020
0020	MOV AL, [BX+03]	8A4703	(AX) = 010B (IP) = 0020	(AX) = 010E (IP) = 0023
0023	MOV CX, [BX+03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 120E (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [000E+DI]	8A850E00	(AX) = 010E (IP) = 0029	(AX) = 01F6 (IP) = 002D
002D	MOV BX, 0003	BB0300	(BX) = 0006 (IP) = 002D	(BX) = 0003 (IP) = 0030
0030	MOV AL, [0016+BX+DI]	8A811600	(AX) = 01F6 (IP) = 0030	(AX) = 0104 (IP) = 0034
0034	MOV AX, 1A07	B8071A	(AX) = 0104 (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	(SP) = 0014 (IP) = 0041 Stack +0 0000 +2 19F5 +4 0000	(SP) = 0012 (IP) = 0042 Stack +0 1A07 +2 0000 +4 19F5
0042	POP ES	07	(SP) = 0012 (IP) = 0042 Stack +0 1A07 +2 0000 +4 19F5	(SP) = 0014 (IP) = 0043 Stack +0 0000 +2 19F5 +4 0000
0043	MOV CX, ES:[BX-01]	268B4FFF	(CX) = 120E (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	(IP) = 004B	(IP) = 004E
004E	MOV BP, SP	8BEC	(BP) = 0014 (IP) = 004E	(BP) = 0014 (IP) = 0050
0050	MOV BP, SP	8BEC	(BP) = 0014 (IP) = 0050	(BP) = 0014 (IP) = 0052
0052	MOV DX, [BP+02]	8B5602	(DX) = 0000 (IP) = 004E	(DX) = 19F5 (IP) = 0055
0055	RET Far 0002	CA0200	(SP) = 0014 (CS) = 1A0A (IP) = 0055 Stack +0 0000 +2 19F5 +4 0000	(SP) = 001A (CS) = 19F5 (IP) = 0000 Stack +2 0000 +4 0000 +6 01F4

0000	INT 20	CD20	(AX) = FFCE (BX) = 0003 (DX) = 19F5 (DI) = 0002 (BP) = 0014 (SP) = 001A (CS) = 19F5 (DS) = 1A07 (ES) = 1A07 Stack +2 0000 +4 0000 +6 01F4	(AX) = 0000 (BX) = 0000 (DX) = 0000 (DI) = 0000 (BP) = 0000 (SP) = 0018 (CS) = 1A0A (DS) = 19F5 (ES) = 19F5 Stack +0 7244 +2 0000 +4 0000

Выводы.

В результате выполнения лабораторной работы был исправлен код готовой программы, освоена базовая теория о режимах адресации.

Приложение.

Текст файла LR2_COMP.LST

```
__Microsoft (R) Macro Assembler Version 5.10                 10/21/20 00:58:1
                                                             Page     1-1


                        ; Программа изучения режимов адресации процессо
                        ра 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 данных
 0000  0000             mem1 DW 0
 0002  0000             mem2 DW 0
 0004  0000             mem3 DW 0
 0006  0B 0C 0D 0E 12 11   vec1 DB 11,12,13,14,18,17,16,15
       10 0F
 000E  0A 14 F6 EC 1E 28   vec2 DB 10,20,-10,-20,30,40,-30,-40
       E2 D8
 0016  01 02 FC FD 03 04   matr DB 1,2,-4,-3,3,4,-2,-1,5,6,-8,-7,7,8,-6,-5
       FE FF 05 06 F8 F9
       07 08 FA FB
 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 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
 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
__Microsoft (R) Macro Assembler Version 5.10                 10/21/20 00:58:1
```

```
 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 сегмента
                            ; ------ 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 стека
                             ;push mem1
                             ;push mem2
 0050  8B EC                 mov bp,sp
 0052  8B 56 02                     mov dx,[bp]+2
 0055  CA 0002                      ret 2
 0058                       Main ENDP
 0058                       CODE ENDS
                             END Main
__Microsoft (R) Macro Assembler Version 5.10                 10/21/20 00:58:1
                                                             Symbols-1


Segments and Groups:

                N a m e           Length  Align Combine Class

ASTACK . . . . . . . . . . . . .        0018   PARA   STACK
CODE . . . . . . . . . . . . . .        0058   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 = 0058
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  LR2_COMP
@VERSION . . . . . .            TEXT  510


     83 Source  Lines
     83 Total   Lines
     19 Symbols

  47800 + 461507 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```

Текст 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 11,12,13,14,18,17,16,15
vec2 DB 10,20,-10,-20,30,40,-30,-40
matr DB 1,2,-4,-3,3,4,-2,-1,5,6,-8,-7,7,8,-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
```