МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Организация систем и ЭВМ»
Тема «Изучение режимов адресации и формирования исполнительного адреса»

Студент гр. 1303 _____ Кропотов Н.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Целью лабораторной работы №2 является изучения режимов адресации.

Задание.

Лабораторная работа 2 предназначена для изучения режимов адресации, используя готовую программу lr2c.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. 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.

LR2C.ASM(41): error A2052: 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.

LR2C.ASM(48): warning A4031: Operand types must match

mov cx, vec2[di]

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

LR2C.ASM(52): warning A4031: Operand types must match

mov cx, matr[bx][di]

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

LR2C.ASM(53): 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.

LR2C.ASM(72): error A2046: Multiple base registers

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.

LR2C.ASM(73): error A2047: Multiple index registers

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.

LR2C.ASM(80): error A2006: Phase error between passes

Main ENDP

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

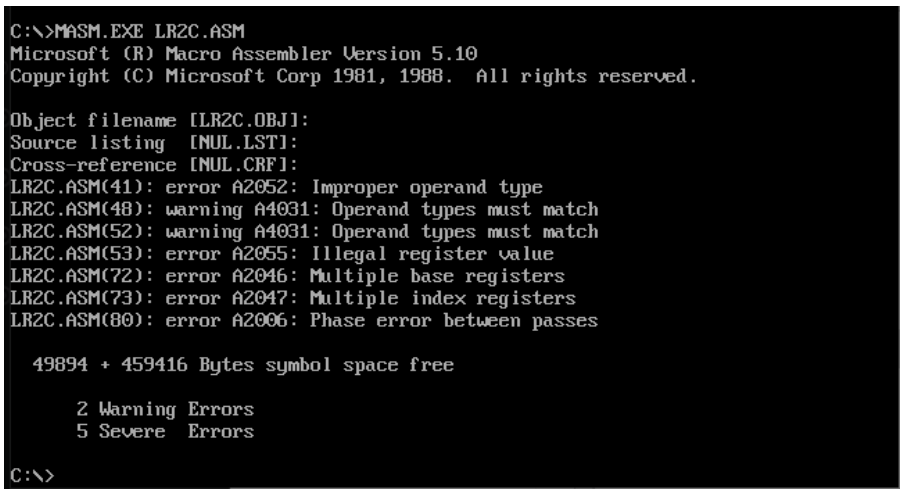

Рис. 1 – Трансляция программы с созданием файла диагностических сообщений

3. Снова протранслировать программу и скомпоновать загрузочный модуль.

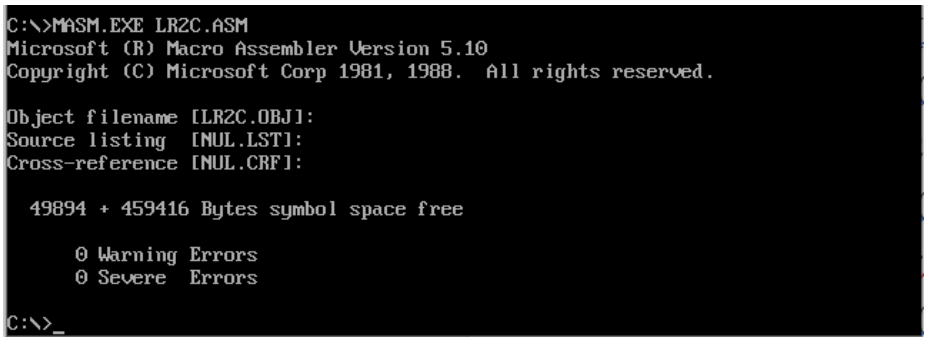

Рис. 2 – Трансляция программы после исправления ошибок

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

Таблица 1 – Протокол пошагового исполнения lr2c.exe

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(IP) = 0000 (SP) = 0018 (CX) = 00B0 STACK(+0)=0000	(IP) = 0001 (SP) = 0016 (CX) = 00B0 STACK(+0)=19F5
0001	SUB AX, AX	2BC0	(IP) = 0001 (SP) = 0016 (AX) = 0000 STACK(+0)=19F5	(IP) = 0003 (SP) = 0016 (AX) = 0000 STACK(+0)=19F5
0003	PUSH AX	50	(IP) = 0003 (SP) = 0016 (AX) = 0000 STACK(+0)=19F5 STACK(+2)=0000	(IP) = 0004 (SP) = 0014 (AX) = 0000 STACK(+0)=0000 STACK(+2)=19F5
0004	MOV AX, 1A07	B8071A	(IP) = 0004 (SP) = 0014 (AX) = 0000 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0007 (SP) = 0014 (AX) = 1A07 STACK(+0)=0000 STACK(+2)=19F5
0007	MOV DS, AX	8ED8	(IP) = 0007 (SP) = 0014 (DS) = 19F5 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0009 (SP) = 0014 (DS) = 1A07 STACK(+0)=0000 STACK(+2)=19F5
0009	MOV AX, 01F4	B8F401	(IP) = 0009 (SP) = 0014 (AX) = 1A07 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 000C (SP) = 0014 (AX) = 01F4 STACK(+0)=0000 STACK(+2)=19F5
000C	MOV CX, AX	8BC8	(IP) = 000C (SP) = 0014 (CX) = 00B0 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 000E (SP) = 0014 (CX) = 01F4 STACK(+0)=0000 STACK(+2)=19F5
000E	MOV BL, 24	B324	(IP) = 000E (SP) = 0014 (BX) = 0000 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0010 (SP) = 0014 (BX) = 0024 STACK(+0)=0000 STACK(+2)=19F5
0010	MOV BH, CE	B7CE	(IP) = 0010 (SP) = 0014 (BX) = 0024 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0012 (SP) = 0014 (BX) = CE24 STACK(+0)=0000 STACK(+2)=19F5
0012	MOV [0002], FFCE	C7060200CE FF	(IP) = 0012 (SP) = 0014 (BX) = CE24	(IP) = 0018 (SP) = 0014 (BX) = CE24

			STACK(+0)=0000 STACK(+2)=19F5	STACK(+0)=0000 STACK(+2)=19F5
0018	MOV BX, 0006	BB0600	(IP) = 0018 (SP) = 0014 (BX) = CE24 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 001B (SP) = 0014 (BX) = 0006 STACK(+0)=0000 STACK(+2)=19F5
001B	MOV [0000], AX	A30000	(IP) = 001B (SP) = 0014 (BX) = 0006 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 001E (SP) = 0014 (BX) = 0006 STACK(+0)=0000 STACK(+2)=19F5
001E	MOV AL, [BX]	8A07	(IP) = 001E (SP) = 0014 (AX) = 01F4 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0020 (SP) = 0014 (AX) = 010B STACK(+0)=0000 STACK(+2)=19F5
0020	MOV AL, [BX+03]	8A4703	(IP) = 0020 (SP) = 0014 (AX) = 010B STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0023 (SP) = 0014 (AX) = 010E STACK(+0)=0000 STACK(+2)=19F5
0023	MOV CX, [BX+03]	8A4703	(IP) = 0023 (SP) = 0014 (CX) = 01F4 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0026 (SP) = 0014 (CX) = 120E STACK(+0)=0000 STACK(+2)=19F5
0026	MOV DI, 0002	BF0200	(IP) = 0026 (SP) = 0014 (DI) = 0000 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0029 (SP) = 0014 (DI) = 0002 STACK(+0)=0000 STACK(+2)=19F5
00029	MOV AL, [000E+DI]	8A850E00	(IP) = 0029 (SP) = 0014 (AX) = 010E STACK(+0)=0000 STACK(+2)=19F5	(IP) = 002D (SP) = 0014 (AX) = 01F6 STACK(+0)=0000 STACK(+2)=19F5
002D	MOV BX, 0003	BB0300	(IP) = 002D (SP) = 0014 (BX) = 0006 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0030 (SP) = 0014 (BX) = 0003 STACK(+0)=0000 STACK(+2)=19F5
0030	MOV AL, [0016+BX+DI]	8A811600	(IP) = 0030 (SP) = 0014 (AX) = 01F6 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0034 (SP) = 0014 (AX) = 0104 STACK(+0)=0000 STACK(+2)=19F5
0034	MOV AX, 1A07	B8071A	(IP) = 0034 (SP) = 0014 (AX) = 0104 STACK(+0)=0000	(IP) = 0037 (SP) = 0014 (AX) = 1A07 STACK(+0)=0000

			STACK(+2)=19F5	STACK(+2)=19F5
0037	MOV ES, AX	8EC0	(IP) = 0037 (SP) = 0014 (ES) = 19F5 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0039 (SP) = 0014 (ES) = 1A07 STACK(+0)=0000 STACK(+2)=19F5
0039	MOV AX, ES:[BX]	268B07	(IP) = 0039 (SP) = 0014 (AX) = 1A07 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 003C (SP) = 0014 (AX) = 00FF STACK(+0)=0000 STACK(+2)=19F5
003C	MOV AX, 0000	B80000	(IP) = 003C (SP) = 0014 (AX) = 00FF STACK(+0)=0000 STACK(+2)=19F5	(IP) = 003F (SP) = 0014 (AX) = 0000 STACK(+0)=0000 STACK(+2)=19F5
003F	MOV ES, AX	8EC0	(IP) = 003F (SP) = 0014 (ES) = 1A07 STACK(+0)=0000 STACK(+2)=19F5	(IP) = 0041 (SP) = 0014 (ES) = 0000 STACK(+0)=0000 STACK(+2)=19F5
0041	PUSH DS	1E	(IP) = 0041 (SP) = 0014 (ES) = 0000 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 0042 (SP) = 0014 (ES) = 0000 STACK(+0)=1A07 STACK(+2)=0000 STACK(+4)=19F5
0042	POP ES	07	(IP) = 0042 (SP) = 0014 (ES) = 0000 STACK(+0)=1A07 STACK(+2)=0000 STACK(+4)=19F5	(IP) = 0043 (SP) = 0012 (ES) = 1A07 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000
0043	MOV CX, ES:[BX-01]	268B4FFF	(IP) = 0043 (SP) = 0012 (CX) = 120E STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 0047 (SP) = 0012 (CX) = FFCE STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000
0047	XCHG AX, CX	91	(IP) = 0047 (AX) = 0000 (CX) = FFCE STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 0048 (AX) = FFCE (CX) = 0000 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000
0048	MOV DI, 0002	BF0200	(IP) = 0048 (AX) = FFCE (DI) = 0002 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 004B (AX) = FFCE (DI) = 0002 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000

004B	MOV ES:[BX+DI], AX	268901	(IP) = 004B (AX) = FFCE (DI) = 0002 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 004E (AX) = FFCE (DI) = 0002 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000
004E	MOV BP, SP	8BEC	(IP) = 004E (AX) = FFCE (BP) = 0010 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 0050 (AX) = FFCE (BP) = 0014 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000
0050	PUSH [0000]	FF360000	(IP) = 0050 (SP) = 0014 (BP) = 0014 STACK(+0)=0000 STACK(+2)=19F5 STACK(+4)=0000	(IP) = 0054 (SP) = 0012 (BP) = 0014 STACK(+0)=01F4 STACK(+2)=0000 STACK(+4)=19F5
0054	PUSH [0002]	FF360200	(IP) = 0054 (SP) = 0012 (BP) = 0014 STACK(+0)=01F4 STACK(+2)=0000 STACK(+4)=19F5 STACK(+6)=0000	(IP) = 0058 (SP) = 0010 (BP) = 0014 STACK(+0)=FFCE STACK(+2)=01F4 STACK(+4)=0000 STACK(+6)=19F5
0058	MOV BP, SP	8BEC	(IP) = 0058 (SP) = 0010 (BP) = 0014 STACK(+0)=FFCE STACK(+2)=01F4 STACK(+4)=0000 STACK(+6)=19F5	(IP) = 005A (SP) = 0010 (BP) = 0010 STACK(+0)=FFCE STACK(+2)=01F4 STACK(+4)=0000 STACK(+6)=19F5
005A	MOV DX, [BP+02]	8B5602	(IP) = 005A (SP) = 0010 (DX) = 0000 STACK(+0)=FFCE STACK(+2)=01F4 STACK(+4)=0000 STACK(+6)=19F5	(IP) = 005D (SP) = 0010 (DX) = 01F4 STACK(+0)=FFCE STACK(+2)=01F4 STACK(+4)=0000 STACK(+6)=19F5
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

Вывод.

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

ПРИЛОЖЕНИЕ А

Тексты исходных файлов программ lr2.asm.

```
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
```