# 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 основной памяти.**

Студентка гр. 3388 Глебова В.С

Преподаватель Фирсов М.А.

Санкт-Петербург 2024

# Цель работы.

Ознакомиться с режимами адресации основной памяти.

# Задание.

**Часть 1.**

1. Получить у преподавателя вариант выбора значений исходных данных (массивов) vec1, vec2 и matr из файла lr2+.dat и занести свои данные вместо значений, указанных в приведённой для образца программе.

2. Протранслировать программу с созданием файла диагностических сообщений и объяснить обнаруженные ошибки (error) и предупреждения (warning). Листинг привести в приложении к отчёту. Закомментировать операторы с ошибками в тексте программы, а операторы с предупреждениями оставить без изменения. Объяснения ошибок и предупреждений должны быть приведены в отчёте по лабораторной работе.

3. Снова протранслировать программу и скомпоновать загрузочный модуль. Учесть, что программа учебная и может выполняться только под отладчиком. В автоматическом режиме она выполняться не должна.

4. Выполнить программу в пошаговом режиме под управлением отладчика с фиксацией содержимого **используемых** регистров и ячеек памяти до и после выполнения каждой команды. Разобраться в используемых режимах адресации и получаемых результатах. Результаты прогона программы под управлением отладчика должны быть представлены в отчёте по лабораторной работе в табличном виде, аналогичном указанному в лаб.работе №1.

Вариант 1:

і vec1 і 1,2,3,4,8,7,6,5

1 і vec2 і 10,20,-10,-20,-30,-40,30,40

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

<Конец задания>

# Основные теоретические положения.

Процессоры Intel x86 используют несколько режимов адресации для указания местоположения операндов, обеспечивая гибкость работы с памятью и регистрами. Всего существует 8 режимов адресации:

1. **Регистровая адресация**: Операнд находится в регистре общего назначения или сегментном регистре. В команде на языке ассемблера указывается имя соответствующего регистра. Это самый быстрый и простой способ доступа к операндам.
2. **Непосредственная адресация**: Операнд задается прямо в коде команды. Например, в арифметических командах или командах пересылки один из операндов может быть указан непосредственно, что удобно для работы с константами.
3. **Прямая адресация**: Операнд находится в памяти, и его адрес известен заранее. Доступ осуществляется напрямую через указание этого адреса или имени переменной. Обычно используется в программах с директивами определения данных для обращения к статическим переменным.
4. **Косвенная адресация**: Вместо явного указания адреса операнда в памяти, он хранится в регистре. Например, адрес может быть записан в EAX, EBX, ECX и других регистрах, и доступ к памяти осуществляется через их содержимое. DS и SS используются для сегментации.
5. **Базовая и индексная адресация**: Доступ к элементам структур данных или массивов осуществляется через регистр, который содержит базовый адрес. Например, в регистрах хранится адрес начала массива, и смещение от этого адреса используется для доступа к отдельным элементам.
6. **Индексная адресация с масштабированием**: Этот метод адресации позволяет работать с массивами слов, двойных слов или учетверенных слов. Регистр, хранящий индекс элемента массива, может быть масштабирован (умножен на 2, 4 или 8), что соответствует размеру элемента массива.
7. **Адресация по базе с индексированием и масштабированием**: Самый универсальный метод адресации, который объединяет базовую и индексную адресацию с возможностью масштабирования индекса. Подходит для работы с любыми типами данных, включая сложные структуры.

# Выполнение работы Часть 1.

1. Изменим данные для файла LR2\_comp.ASM значениями, которые были получены в соответсвуещем варианте (см. рис. 1).

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Рисунок 1

1. Протранслируем программу LR2\_comp.ASM с созданием файла диагностических сообщений при помощи команды masm. В результате трансляции получаем 5 ошибок и 2 предупреждения (см. рис. 2.)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2

После трансляции были получены ошибки, в которых сначала приведено название файла, в котором была обнаружена ошибка, а в скобках указана строчка программы. Конкретно о каждой ошибке:

**Ошибка A2052: Improper operand type** – это ошибка неверного типа операнда. Она возникает в строке mov mem3,[BX], где пытаются перенести данные из области памяти, адресуемой через регистр BX, в другой участок памяти (mem3). Инструкция mov не поддерживает прямую передачу данных между двумя ячейками памяти. Один из операндов обязан быть регистром.

**Ошибка A2055: Illegal register value** – недопустимое значение регистра. Появляется в строке mov ax, matr[BX\*4][DI], поскольку здесь пытаются использовать масштабирование для регистра BX, умножая его на 4. Однако регистр BX не может быть использован с умножением, что приводит к ошибке.

**Ошибка A2046: Multiple base registers** – несколько базовых регистров. Возникает при попытке использовать два базовых регистра одновременно в строке mov ax, matr[bp+BX]. Это недопустимо, поскольку инструкция не позволяет иметь больше одного базового регистра.

**Ошибка A2047: Multiple index registers** – несколько индексных регистров. Встречается в строке mov ax, m, где задействованы сразу два индексных регистра (DI и si), что не разрешено.

**Ошибка A2006: Phase error between passES** – фазовая ошибка между проходами. Появляется при наличии сложных схем адресации или неправильного использования регистров, что затрудняет сборку программы на нескольких шагах. Эта ошибка исчезает после устранения всех предыдущих.

**Предупреждения:**

**Предупреждение A4031: Operand typES must match** – типы операндов должны совпадать. Возникает в строках mov CX, matr[BX][DI] и mov CX, vec2[DI], когда типы операндов не соответствуют друг другу. Это значит, что объем памяти, используемый операндами, не совпадает.

1. Протранслируем программу снова, ошибок больше не было обнаружено, остались только предупреждения (см. рис. 3).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3

Выполним программу в ручном режиме (см. рис. 4)

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 4

1. Пошагово выполним программу и зафиксируем измнения в регистрах и стеке после каждого шага.

Таблица 1 – Протокол пошагового исполнения lr2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес команды | Код команды | 16-ый код команды | Содержание регистров и ячеек памяти | |
| До выполнения | После выполнения |
| 0000 | PUSH DS | 1E | (SP) = 0018  (IP) = 0000  (DS) = 119C  Stack (+0) 0000 | \*(IP) = 0001  \*(SP) = 0016  (DS) = 119C  Stack \*(+0) 119C |
| 0001 | SUB AX, AX | 2BC0 | (AX) = 0000  (IP) = 0001 | (AX) = 0000  \*(IP) = 0003 |
| 0003 | PUSH AX | 50 | (SP) = 0016  (IP) = 0003  (AX) = 0000  Stack (+0) 119C  Stack (+2) 0000 | \*(SP) = 0014  \*(IP) = 0004  (AX) = 0000  Stack \*(+0) 0000  Stack \*(+2) 119C |
| 0004 | MOV AX, 11AE | B8AE11 | (AX) = 0000  (IP) = 0004 | \*(AX) = 11AE  \*(IP) = 0007 |
| 0007 | MOV DS, AX | 8ED8 | (DS) = 119C  (IP) = 0007 | \*(DS) = 11AE  \*(IP) = 0009 |
| 0009 | MOV AX, 01F4 | B8F401 | (AX) = 11AE  (IP) = 0009 | \*(AX) = 01F4  \*(IP) = 000C |
| 000С | MOV CX, AX | 8BC8 | (CX) = 00B6  (IP) = 000C | \*(CX) = 01F4  \*(IP) = 000E |
| 000E | MOV BL, 24 | B324 | (BX) = 0000  (IP) = 000E | \*(BX) = 0024  \*(IP) = 0010 |
| 0010 | MOV BH, CE | B7CE | (IP) = 0010  (BX) = 0024 | \*(IP) = 0012  \*(BX) = CE24 |
| 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) = 0101  \*(IP) = 0020 |
| 0020 | MOV AL, [BX+03] | 8A4703 | (AX) = 0101  (IP) = 0020 | \*(AX) = 0104  \*(IP) = 0023 |
| 0023 | MOV CX, [BX+03] | 8B4F03 | (CX) = 01F4  (IP) = 0023 | \*(CX) = 0804  \*(IP) = 0026 |
| 0026 | MOV DI, 0002 | BF0200 | (DI) = 0000  (IP) = 00026 | \*(DI) = 0002  \*(IP) = 0029 |
| 0029 | MOV AL, [DI+000E] | BF0200 | (AX) = 0110  (IP) = 0029 | \*(AX) = 01F2  \*(IP) = 002D |
| 002D | MOV CX, [DI+000E] | 8B8DOE00 | (IP) = 002D  (CX) = 0C10 | \*(IP) = 0031  \*(CX) = ECF6 |
| 0031 | MOV BX, 0003 | BB0300 | (BX) = 0006  (IP) = 0031 | \*(BX) = 0003  \*(IP) = 0034 |
| 0034 | MOV AL, [BX+DI+0016] | 8A811600 | (AX) = 01E2  (IP) = 0034 | \*(AX) = 01FD  \*(IP) = 0038 |
| 0038 | MOV CX, [BX+DI+0016] | 8B891600 | (CX) = C4E2  (IP) = 0038 | \*(CX) = FEFD  \*(IP) = 003C |
| 003C | MOV AX, 11AE | B8AE11 | (AX) = 0101  (IP) = 003C | \*(AX) = 11AE  \*(IP) = 003F |
| 003F | MOV ES, AX | 8EC0 | (ES) = 119C  (IP) = 003F | \*(ES) = 11AE  \*(IP) = 0041 |
| 0041 | MOV AX, ES:[BX] | 268B07 | (AX) = 11AE  (IP) = 0041 | \*(AX) = 00FF  \*(IP) = 0044 |
| 0044 | MOV AX, 0000 | B80000 | (AX) = 00FF  (IP) = 0044 | \*(AX) = 0000  \*(IP) = 0047 |
| 0047 | MOV ES, AX | 8EC0 | (ES) = 11AE  (IP) = 0047 | \*(ES) = 0000  \*(IP) = 0049 |
| 0049 | PUSH DS | 1E | (IP) = 0049  Stack (+0) 0000  Stack (+2) 119C  Stack (+4) 0000 | \*(IP) = 004A  Stack \*(+0) 11AE  Stack \*(+2) 0000  Stack \*(+4) 119C |
| 004A | POP ES | 07 | (SP) = 0012  (ES) = 0000  (IP) = 004A  Stack (+0) 11AE  Stack (+2) 0000  Stack (+4) 119C | \*(SP) = 0014  \*(ES) = 11AE  \*(IP) = 004B  Stack \*(+0) 0000  Stack \*(+2) 119C  Stack \*(+4) 0000 |
| 004B | MOV CX, ES:[BX-01] | 268B4FFF | (CX) = FD01  (IP) = 004B | \*(CX) = FFCE  \*(IP) = 004F |
| 004F | XCHG AX, CX | 91 | (AX) = 0000  (CX) = FFCE  (IP) = 004F | \*(AX) = FFCE  \*(CX) = 0000  \*(IP) = 0050 |
| 0050 | MOV DI, 0002 | BF0200 | (DI) = 0002  (IP) = 0050 | (DI) = 0002  \*(IP) = 0053 |
| 0053 | MOV ES;[BX+DI], AX | 268901 | (ES) = 11AE  (IP) = 0053 | (ES) = 11AE  \*(IP) = 0056 |
| 0056 | MOV BP, SP | 8BEC | (BP) = 0000  (IP) = 0056 | \*(BP) = 0014  \*(IP) = 0058 |
| 0058 | PUSH [0000] | FF360000 | (SP) = 0014  (IP) = 0058  Stack (+0) 0000  Stack (+2) 119C  Stack (+4) 0000 | \*(SP) = 0012  \*(IP) = 005C  Stack \*(+0) 01F4  Stack \*(+2) 0000  Stack \*(+4) 119C |
| 005C | PUSH [0002] | FF360200 | (SP) = 0012  (IP) = 005C  Stack (+0) 01F4  Stack (+2) 0000  Stack (+4) 119C  Stack (+6) 0000 | \*(SP) = 0010  \*(IP) = 0060  Stack \*(+0) FFCE  Stack \*(+2) 01F4  Stack \*(+4) 0000  Stack \*(+6) 119C |
| 0060 | MOV BP, SP | 8BEC | (BP) = 0014  (IP) = 0060 | \*(BP) = 0010  \*(IP) = 0062 |
| 0062 | MOV DX, [BP+02] | 8B5602 | (dx) = 0000  (IP) = 0062 | \*(dx) = 01F4  \*(IP) = 0065 |
| 0065 | RET Far | CB | (SP) = 0010  (cs) = 11B1  (IP) = 0065  Stack (+0) FFCE  Stack (+2) 01F4  Stack (+4) 0000  Stack (+6) 119C | \*(SP) = 0014  \*(cs) = 01F4  \*(IP) = FFCE  Stack:  Stack \*(+0) 0000  Stack \*(+2) 119C  Stack \*(+4) 0000  Stack \*(+6) 0000 |

**Вывод.**

В ходе выполнения работы было произведено ознакомление с режимами адресации основной памяти на языке Ассемблера, при помощи отладчика afd.

**ПРИЛОЖЕНИЕ A**

1. **lr2\_comp.asm (первый)**

; Учебная программа лабораторной работы №2 по дисциплине "Организация ЭВМ и С";

;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 1,2,3,4,8,7,6,5

vec2 DB 10,20,-10,-20,-30,-40,30,40

matr DB 1,2,3,4,-4,-3,-2,-1,5,6,7,8,-8,-7,-6,-55

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]

; Индексированная 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 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

Main ENDP

CODE ENDS

END Main

1. **LR2COMP.LST (первый)**

Microsoft (R) Macro Assembler Version 5.10 9/24/24 18:30:17

Page 1-1

; Учебная программа лабораторной работы №2 по

дисциплине "Организация ЭВМ и С";

;

= 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 13 12 11 10 0C 0D vec1 DB 19,18,17,16,12,13,14,15

0E 0F

000E 1E 3C E2 C4 0A 28 vec2 DB 30,60,-30,-60,10,40,-10,-40

F6 D8

0016 04 03 FF FE 02 01 matr DB 4,3,-1,-2,2,1,-3,-4,-5,-6,-7,-8

,8,7,6,5

FD FC FB FA F9 F8

08 07 06 05

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

Microsoft (R) Macro Assembler Version 5.10 9/23/24 18:28:17

Page 1-2

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]

lr2\_comp.ASM(54): error A2052: Improper operand type

; Базированная адресация

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

0023 8B 4F 03 mov CX,3[BX]

; Индексированная адресация

0026 BF 0002 mov DI,ind

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

002D 8B 8D 000E R mov CX,vec2[DI]

lr2\_comp.ASM(61): warning A4031: Operand typES must match

; Адресация с базированием и индексированием

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]

lr2\_comp.ASM(65): warning A4031: Operand typES must match

003C 8B 85 0022 R mov AX,matr[BX\*4][DI]

lr2\_comp.ASM(66): error A2055: Illegal register value

; ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ

; Переопределение сегмента

; ------ вариант 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

; ------ вариант 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

; ------ вариант 3

0054 BF 0002 mov DI,ind

0057 26: 89 01 mov ES:[BX+DI],AX

; ------ вариант 4

005A 8B EC mov bp,SP

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

lr2\_comp.ASM(86): error A2046: MultIPle base registers

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

lr2\_comp.ASM(87): error A2047: MultIPle index registers

; Использование сегмента стека

0066 FF 36 0000 R push mem1

006A FF 36 0002 R push mem2

006E 8B EC mov bp,SP

0070 8B 56 02 mov dx,[bp]+2

0073 CB ret

0074 Main ENDP

lr2\_comp.ASM(94): error A2006: Phase error between passES

0074 CODE ENDS

END Main

Microsoft (R) Macro Assembler Version 5.10 9/23/24 18:29:19

Symbols-1

Segments and Groups:

N a m e Length Align Combine Class

ASTACK . . . . . . . . . . . . . 0018 PARA STACK

CODE . . . . . . . . . . . . . . 0074 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 = 0074

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

96 Source LinES

96 Total LinES

19 Symbols

47806 + 459454 BytES symbol SPace free

2 Warning Errors

5 Severe Errors

**3. lr2\_comp.asm (второй)**

; Учебная программа лабораторной работы №2 по дисциплине "Организация ЭВМ и С";

;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 1,2,3,4,8,7,6,5

vec2 DB 10,20,-10,-20,-30,-40,30,40

matr DB 1,2,3,4,-4,-3,-2,-1,5,6,7,8,-8,-7,-6,-55

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]

; Индексированная 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 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

Main ENDP

CODE ENDS

END Main

**4. LR2.LST (после исправления)**

Microsoft (R) Macro Assembler Version 5.10 9/23/24 18:42:40

Page 1-1

; Учебная программа лабораторной работы №2 по дисциплине "Организация ЭВМ и С";

= 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 13 12 11 10 0C 0D vec1 DB 19,18,17,16,12,13,14,15

0E 0F

000E 1E 3C E2 C4 0A 28 vec2 DB 30,60,-30,-60,10,40,-10,-40

F6 D8

0016 04 03 FF FE 02 01 matr DB 4,3,-1,-2,2,1,-3,-4,-5,-6,-7,-8,8,7,6,5

FD FC FB FA F9 F8

08 07 06 05

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

Microsoft (R) Macro Assembler Version 5.10 9/23/24 18:35:55

Page 1-2

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]

; Индексированная адресация

0026 BF 0002 mov DI,ind

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

002D 8B 8D 000E R mov CX,vec2[DI]

lr2\_comp.ASM(61): warning A4031: Operand typES must match

; Адресация с базированием и индексированием

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]

lr2\_comp.ASM(65): warning A4031: Operand typES must match

; mov AX,matr[BX\*4][DI]

; ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ

; Переопределение сегмента

; ------ вариант 1

003C B8 ---- R mov AX, SEG vec2

003F 8E C0 mov ES, AX

0041 26: 8B 07 mov AX, ES:[BX]

0044 B8 0000 mov AX, 0

; ------ вариант 2

0047 8E C0 mov ES, AX

0049 1E push DS

004A 07 pop ES

004B 26: 8B 4F FF mov CX, ES:[BX-1]

004F 91 xchg CX,AX

; ------ вариант 3

0050 BF 0002 mov DI,ind

0053 26: 89 01 mov ES:[BX+DI],AX

; ------ вариант 4

0056 8B EC mov bp,SP

; mov AX,matr[bp+BX]

; mov AX,matr[bp+DI+si]

; Использование сегмента стека

0058 FF 36 0000 R push mem1

005C FF 36 0002 R push mem2

0060 8B EC mov bp,SP

0062 8B 56 02 mov dx,[bp]+2

0065 CB ret

0066 Main ENDP

0066 CODE ENDS

END Main

Microsoft (R) Macro Assembler Version 5.10 9/23/24 18:36:02

Symbols-1

Segments and Groups:

N a m e Length Align Combine Class

ASTACK . . . . . . . . . . . . . 0018 PARA STACK

CODE . . . . . . . . . . . . . . 0066 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 = 0066

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

96 Source LinES

96 Total LinES

19 Symbols

47822 + 459438 BytES symbol SPace free

2 Warning Errors

0 Severe Errors

**5. LR2.MAP**

Start Stop Length Name Class

00000H 00017H 00018H ASTACK

00020H 00045H 00026H DATA

00050H 000B5H 00066H CODE

Program entry point at 0005:0000