# 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 основной памяти**»

Студент гр. 3388 Лексин М.В.

Преподаватель Фирсов М.А.

Санкт-Петербург 2024

# Цель работы.

Изучить режимы адресации основной памяти в ассемблере Intel 8086.

# Задание.

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 12:

і vec1 DB 6,7,8,9,13,12,11,10

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

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

# Основные теоретические положения.

Процессоры Intel X86 обычно работают с данными, называемыми операндами. Эти операнды могут быть расположены в нескольких местах:

1. В регистрах процессора общего назначения
2. Непосредственно в самой инструкции
3. В ячейках памяти, адрес которых указан прямо или косвенно
4. В портах ввода-вывода

Для определения местоположения операнда используется восемь различных методов адресации:

1. **Адресация через регистры**

Операнды могут находиться в любом регистре общего назначения или сегментном регистре. В ассемблерном коде просто указывается имя нужного регистра.

1. **Непосредственная адресация**

Некоторые команды позволяют указать один из операндов прямо в инструкции. Это применимо к командам пересылки и большинству арифметических операций (кроме деления).

1. **Прямая адресация памяти**

Если известен точный адрес операнда в памяти, его можно использовать напрямую. В реальных программах часто применяются директивы определения данных, позволяющие обращаться к статическим переменным по имени, а не по адресу.

1. **Косвенная адресация**

Адрес операнда может храниться в регистре. По умолчанию используется сегментный регистр DS, кроме случаев с регистрами ESP, EBP или BP, когда применяется SS.

**5-6. Базовая и индексная адресация**

Эти методы полезны при работе со структурами данных или при доступе к параметрам в стеке из подпрограммы. Также они применяются для работы с одномерными массивами.

**7. Индексная адресация с масштабированием**

Этот метод похож на предыдущий, но позволяет работать с массивами слов, двойных или учетверенных слов, умножая индекс на соответствующий коэффициент (1, 2, 4 или 8).

**8. Комбинированная адресация**

Это наиболее полный метод, включающий все предыдущие как частные случаи. Он позволяет использовать базу, индекс и масштабирование одновременно.

Важно отметить, что выбор сегментного регистра по умолчанию зависит от используемых регистров: если применяются EBP или ESP, используется SS, в остальных случаях - DS.

# Выполнение работы

1. Был получен вариант выбора значений исходных данных у преподавателя. Полученные данные были занесены вместо значений, указанных в приведенной для образца программе lr2\_comp.asm.

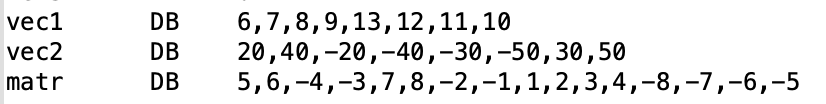


Рисунок 1

1. Программа балы протранслирована с созданием файла диагностических сообщений. При транслировании программы нашлось 5 ошибок и 2 предупреждения.

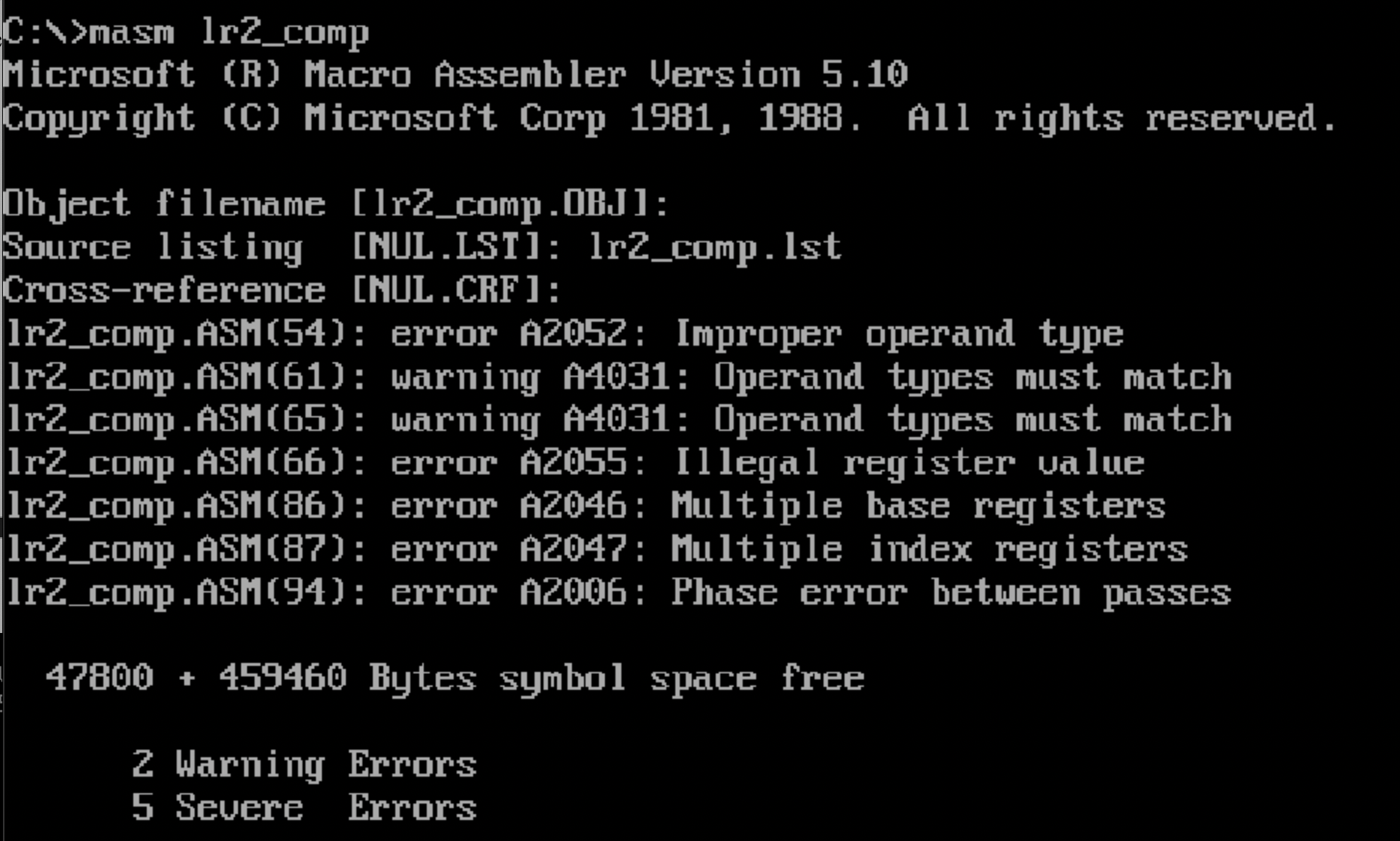


Рисунок 2

Операторы с ошибками в тексте программы были закомментированы, а операторы с предупреждениями остались без изменений.



Рисунок 3

**Объяснение ошибок и предупреждений:**

* *Строка 54: error A2052: Improper operand type*. Эта команда переводит информацию из сегмента памяти в другой, что невозможно в языке Assembler**.** В данном случае необходимо перевести информацию из памяти в регистр, а затем уже в необходимый сегмент информацию перевести из регистра.
* *Строка 61: warning A4031: Operand types must match.* Объем регистра cx составляет 2 байта, в то время, как вес элемента в массиве составляет 1 байт, в данном случае надо вместо регистра cx использовать, например, cl.
* *Строка 65: warning A4031: Operand types must match.* Та же проблема, что и в строке 61. matr - байтовый массив, cx - 16-битный регистр.
* *Строка 66: error A2055: Illegal register value.* Здесь используется базово-индексная адресация. При данном типе адресации надо сначала изменить значение регистра, затем уже переводить информацию.
* *Строка 86: error A2046: Multiple base registers.* В базированной адресации необходимо указывать базовый регистр, затем произволить смещение с помощью индексного. Так как здесь оба базовые, надо сначала сложить значения регистров, затем уже передавать информацию указателю из одного регистра.
* *Строка 87: error A2047: Multiple index registers.* Эта ошибка похода на ошибку выше. В таком случае необходимо сначала в регистр di занести общую сумму, затем уже производить смещение.
* *Строка 94: error A2006: Phase error between passes.* Эта строка не должна быть закомментирована. Она необходима для правильного завершения процедуры Main. После того, как строка была раскомментирована, программа была протранслирована без ошибок.

1. Программа снова была протранслирована и скомпонован загрузочный модуль.
2. Программа была выполнена в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения каждой команды.

Таблица 1 - Протокол пошагового исполнения lr2\_comp

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес  команды | Символический  код команды | 16 - ричный  код команды | Содержимое регистров и ячеек памяти | |
| До выполнения | После выполнения |
| 0000 | PUSH DS | 1E | Stack(+0) = 0000  (IP) = 0000  (SP) = 0018 | Stack(+0) = 119C (IP) = 0001  (SP) = 0016 |
| 0001 | SUB AX, AX | 2BC0 | (IP) = 0001  (AX) = 0000 | (IP) = 0003  (AX) = 0000 |
| 0003 | PUSH AX | 50 | (IP) = 0003  (SP) = 0016 Stack(+0) = 119C Stack(+2) = 0000 | (IP) = 0004  (SP) = 0014  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 | (IP) = 0009 (AX) =11AE | (IP) = 000C  (AX) = 01F4 |
| 000C | MOV CX, AX | 8BC8 | (IP) = 000C  (CX)=0000 | (IP) = 000E  (CX)=01F4 |
| 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  DS:0000=00 00 00 00 | (IP) = 0018  DS:0000=00 00 CE FF |
| 0018 | MOV BX,0006 | BB0600 | (IP) = 0018  (BX) = CE24 | (IP) = 001B  (BX) = 0006 |
| 001B | MOV [0000], AX | A30000 | (IP) = 001B  DS:0000=00 00 CE FF | (IP) = 001E  DS:0000=F4 01 CE FF |
| 001E | MOV AL, [BX] | 8A07 | (IP) = 001E  (AX) = 01F4 | (IP) = 0020  (AX) = 0106 |
| 0020 | MOV AL, [BX+03] | 8A4703 | (IP) = 0020  (AX)=0106 | (IP) = 0023  (AX)=0109 |
| 0023 | MOV CX, [BX+03] | 8B4F03 | (IP) = 0023  (CX)=01F4 | (IP) = 0026  (CX)=0D09 |
| 0026 | MOV DI, 0002 | BF0200 | (IP) = 0026  (DI) = 0000 | (IP) = 0029  (DI) = 0002 |
| 0029 | MOV AL, [DI+000E] | 8A850E00 | (AX) = 0109  (IP) = 0029 | (AX) =01EC  (IP) = 002D |
| 002D | MOV BX,0003 | 8B8D0E00 | (BX) = 0006  (IP) = 002D | (BX)=0003  (IP) = 0030 |
| 0030 | MOV AL,[BX+DI+0016] | 8A811600 | (AX) = 01EC  (IP) = 0030 | (AX)=0108  (IP)=0034 |
| 0034 | MOV AX,11A4 | B8AE11 | (AX) =0108  (IP) = 0034 | (AX)=11AE  (IP)=0037 |
| 0037 | MOV ES,AX | 8EC0 | (ES) = 119C  (IP) = 0037 | (ES)=11AE  (IP)=0039 |
| 0039 | MOV AX,ES:[BX] | 268B07 | (AX) = 11AE  (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) = 11AE  (IP) = 003F | (ES)=0000  (IP)=0041 |
| 0041 | PUSH DS | 1E | Stack(+0) = 0000  Stack(+2) = 119C  (IP) = 0041  (SP) = 0014 | Stack(+0) =11AE  Stack(+2)=0000  Stack(+4) = 119C  (IP) = 0042  (SP) = 0012 |
| 0042 | POP ES | 07 | Stack(+0)=11AE  Stack(+2)=0000  Stack(+4) =119C  (IP) = 0042  (SP) = 0012  (ES) = 0000 | Stack(+0) = 0000  Stack(+2) = 119C  Stack(+4) = 0000  (IP) = 0043  (SP) = 0014  (ES) = 11AE |
| 0043 | MOV CX,ES:[BX-01] | 268B4FFF | (CX) = 0D09  (IP) = 0043 | (CX)=FFCE  (IP)=0047 |
| 0047 | XCHG AX, CX | 91 | (AX) = 0000  (CX) = FFCE  (IP)=0047 | (CX) = 0000  (AX) = FFCE  (IP)=0048 |
| 0048 | MOV DI,0002 | BF0200 | (DI) = 0002  (IP)=0048 | (DI) = 0002  (IP)=004B |
| 004B | MOV ES:[BX+DI], AX | 268901 | (ES) = 11AE  (IP)=004B  ES:0000=F4 01 CE FF 00 00 06 07 | (ES) = 11AE  (IP)=004E  ES:0000=F4 01 CE FF 00 CE FF 07 |
| 004E | MOV BP, SP | 8BEC | (BP) = 0000  (IP)=004E | (BP) = 0014  (IP)=0050 |
| 0050 | PUSH [0000] | FF360000 | Stack(+0) = 0000  Stack(+2) = 119C  Stack(+4) = 0000  (IP)=0050 | Stack(+0) = 01F4  Stack(+2) = 0000  Stack(+4) = 119C  (IP)=0054 |
| 0054 | PUSH [0002] | FF360200 | Stack(+0) = 01F4  Stack(+2) = 0000  Stack(+4) = 119C  (IP) = 0054 | Stack(+0) = FFCE  Stack(+2) = 01F4  Stack(+4) = 0000  Stack(+6) = 119C  (IP) = 0058 |
| 0058 | MOV BP, SP | 8BEC | (BP) = 0014  (IP)=0058 | (BP) = 0010  (IP) = 005A |
| 005A | MOV DX, [BP+02] | 8B5602 | (DX) = 0000  (IP) = 005A | (DX) = 01F4  (IP) = 005D |
| 005D | RET Far | CB | Stack(+0) = FFCE | Stack(+2) = 0000 |

# Вывод.

В ходе выполнения лабораторной работы были изучены режимы адресации основной памяти в ассемблере Intel 8086.

# ПРИЛОЖЕНИЕ А

1. lr2\_comp.ASM

­

; 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 6,7,8,9,13,12,11,10

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

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

; 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

2. lr2\_comp.lst

Microsoft (R) Macro Assembler Version 5.10 9/27/24 01:34:02

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 06 07 08 09 0D 0C vec1 DB 6,7,8,9,13,12,11,10

0B 0A

000E 14 28 EC D8 E2 CE vec2 DB 20,40,-20,-40,-30,-50,30,50

1E 32

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

-7,-6,-5

FE FF 01 02 03 04

F8 F9 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

Microsoft (R) Macro Assembler Version 5.10 9/27/24 01:34:02

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\_2.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\_2.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\_2.ASM(65): warning A4031: Operand types must match

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

lr2\_comp\_2.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\_2.ASM(86): error A2046: Multiple base registers

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

lr2\_comp\_2.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\_2.ASM(94): error A2006: Phase error between passes

0074 CODE ENDS

END Main

Microsoft (R) Macro Assembler Version 5.10 9/27/24 01:34:02

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\_2

@VERSION . . . . . . . . . . . . TEXT 510

96 Source Lines

96 Total Lines

19 Symbols

47786 + 459474 Bytes symbol space free

2 Warning Errors

5 Severe Errors