**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Организация ЭВМ и систем»**

Тема: Изучение режимов адресации основной памяти.

|  |  |  |
| --- | --- | --- |
| Студент гр. 3388 |  | Титкова С.Д. |
| Преподаватель |  | Фирсов М.А. |

Санкт-Петербург

2024

**Цель работы.**

Изучение режимов адресации основной памяти в языке Ассемблер x86.

**Основные теоретические положения.**

В Ассемблере x86 существует несколько режимов адресации основной памяти, например:

**Регистровая адресация:** При регистровой адресации в инструкции указывается имя регистра, в котором хранятся данные, с которыми будет работать инструкция. Например, MOV DS,AX копирует содержимое регистра AX в DS.

**Прямая адресация:** При прямой адресации эффективный адрес ячейки памяти записывается непосредственно в инструкции. Например, MOV AX, 0003 перемещает значение по адресу памяти 0003 в регистр AX.

**Косвенная адресация:** При косвенной адресации эффективный адрес находится в SI, DI, BX или BP. Например, MOV AX, [DI] перемещает содержимое ячейки памяти, на которую указывает регистр DI, в регистр AX.

**Базированная адресация:** При базовой адресации адрес смещения операнда задается суммой содержимого регистров BX/BP и 8-битного/16-битного смещения. Например, MOV DX, [BX+4] перемещает в регистр DX значение, находящееся в ячейке памяти, вычисленное путем прибавления 4 к содержимому регистра BX. Также существуют и другие виды записей: MOV DX, [BX]+4 или MOV DX, 4[BX] (все они эквивалентны)

**Индексированная адресация:** При индексированной адресации адрес смещения операнда находится путем сложения содержимого регистра SI или DI и 8-битного/16-битного смещения. Например, MOV AX, [BX+DI] перемещает в регистр AX значение, находящееся в ячейке памяти, вычисленной путем сложения содержимого регистра DI с содержимым регистра BX.

**Адресация с базированием и индексированием:** Это комбинация базовой адресации и индексированной адресации. В этом режиме смещение операнда вычисляется путем сложения содержимого базового регистра, содержимого индексного регистра и 8- или 16-битного смещения. Например, MOV AX, [BP+SI+0016] перемещает в регистр AX значение в ячейке памяти, вычисленное путем сложения содержимого регистров SI, BP и смещения 0016.

**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). 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 22:

і vec1 і 4,6,8,10,13,11,9,7 i

22 і vec2 і 80,30,-80,-30,-60,-50,60,50 i

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

**Выполнение работы.**

1. Значения исходных данных (массивов) vec1, vec2 и matr из файла lr2.dat получены у преподавателя и занесены вместо значений, указанных в приведённой для образца программе.

2. Программа протранслирована с созданием файла диагностических сообщений, объяснены обнаруженные ошибки (error) и предупреждения (warning). После чего закомментированы операторы с ошибками в тексте программы, а операторы с предупреждениями оставлены без изменения. (Рис. 1)



Рисунок 1

* Ошибка LR2\_COMP.ASM(54): error A2052: Improper operand type (Неверный тип операнда)

**Строка 54:** mov mem3,[bx]

В x86 ассемблере нельзя напрямую перемещать значения из одной области памяти в другую. Вместо этого, нужно использовать регистры в качестве промежуточных переменных. В данном случае необходимо перевести информацию из памяти в регистр, а затем уже перевести информацию из регистра в необходимый сегмент.

* Предупреждение LR2\_COMP.ASM(61): warning A4031: Operand types must match (Несоответствие типов операндов)

**Строка 61:** mov cx,vec2[di]

Несоответствие типов операндов, cx – 1 слово (2 байта), элемент vec2 – 1 байт.

* Предупреждение LR2\_COMP.ASM(65): warning A4031: Operand types must match (Несоответствие типов операндов)

**Строка 65:** mov cx,matr[bx][di]

Несоответствие типов операндов, cx – 1 слово (2 байта), элемент matr – 1 байт.

* Ошибка LR2\_COMP.ASM(66): error A2055: Illegal register value (Незаконное использование регистра)

**Строка 66:** mov ax,matr[bx\*4][di]

Здесь используется базово-индексная адресация. Такая форма адресации используется в тех случаях, когда в регистре находится адрес начала структуры данных, а доступ надо осуществить к какому-нибудь элементу этой структуры. При данном типе адресации надо сначала изменить значение регистра, затем уже переводить информацию. Необходимо сначала умножить bx на 4, а затем производить действия.

* Ошибка LR2\_COMP.ASM(86): error A2046: Multiple base registers (Несколько базовых регистров)

**Строка 86:** mov ax,matr[bp+bx]

Нельзя складывать регистры bp и bx. Так как здесь оба регистра базовые, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра. Необходимо сначала в регистр bp занести общую сумму, затем уже производить смещение.

* Ошибка LR2\_COMP.ASM(87): error A2047: Multiple index registers (Несколько индексных регистров)

**Строка 87:** mov ax,matr[bp+di+si]

Нельзя складывать регистры di и si. Так как здесь два индексных регистра, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра. Необходимо сначала в регистр di занести общую сумму, затем уже производить смещение.

* Ошибка LR2\_COMP.ASM(94): error A2006: Phase error between passes

**Строка 94:** Main ENDP

Объяснение:

Ошибка указывает на то что в функции Main содержатся ошибки.

3. Программа снова была протранслирована и скомпонован загрузочный модуль.(Рис. 2)

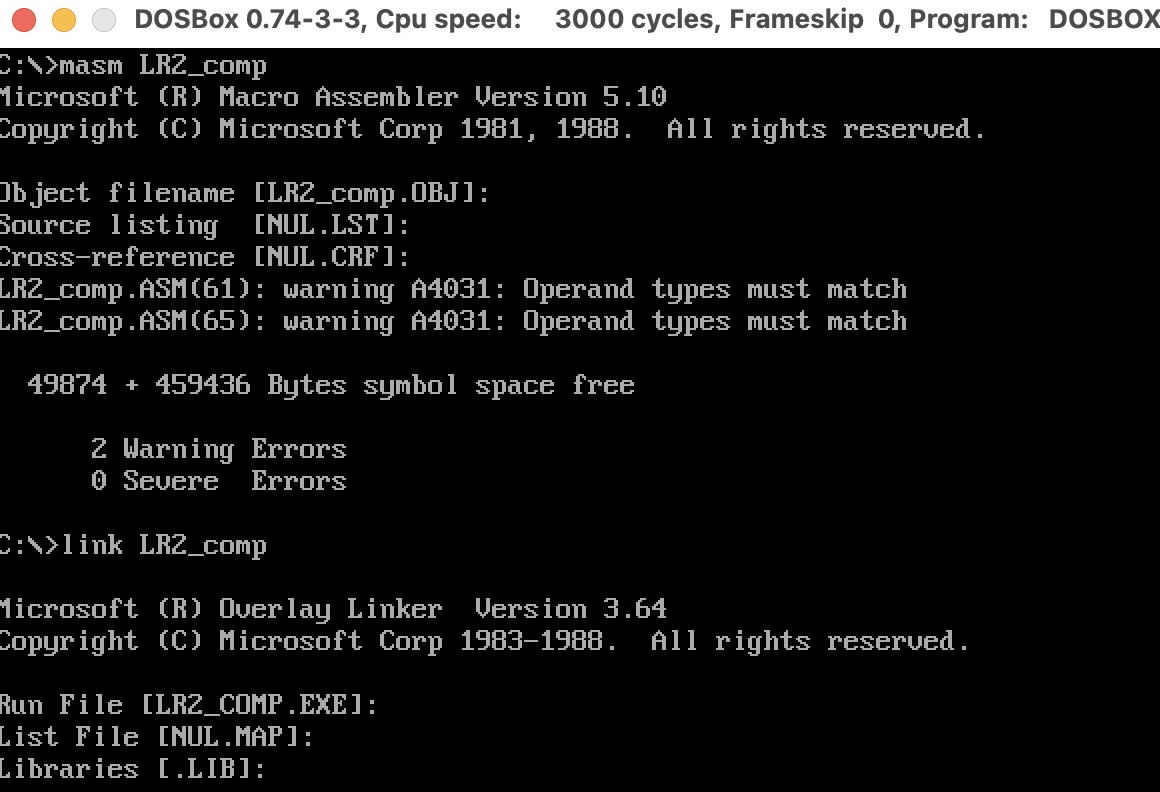


Рисунок 2

4. Программа выполнена в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения каждой команды. Результаты прогона программы под управлением отладчика представлены в Табл. 1, аналогично указанной в лаб.работе №1.

Содержимое используемых регистров и ячеек памяти до выполнения 1-й команды(Рис. 3)

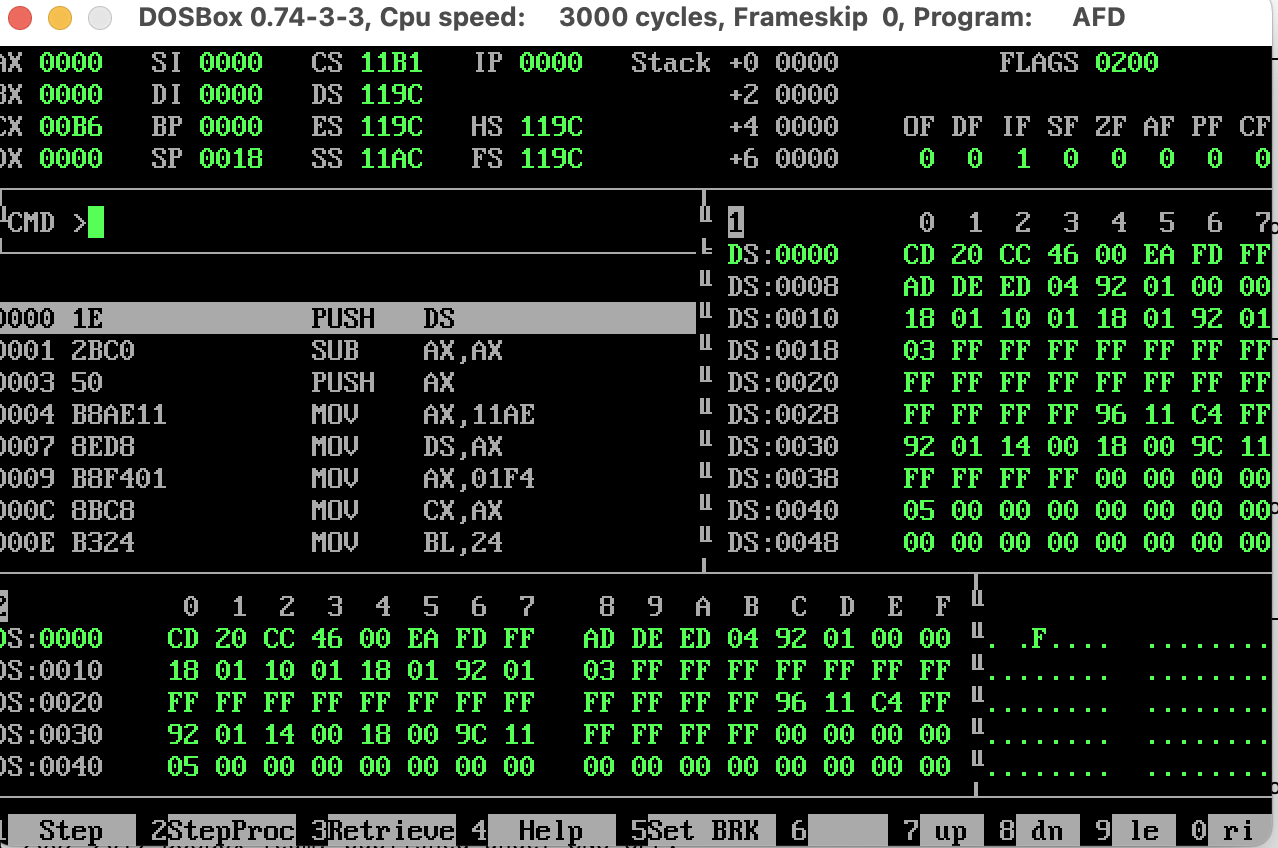


Рисунок 3

Табл.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес команды | Символьный 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  +6 0000 | (SP)=0016  (IP)=0001  Stack:  +0 119C  +2 0000  +4 0000  +6 0000 |
| 0001 | SUB AX, AX | 2BC0 | (IP)=0001  (AX) = 0000 | (IP)=0003  (AX) = 0000 |
| 0003 | PUSH AX | 50 | (SP)= 0016  (IP)= 0003  Stack:  +0 119C  +2 0000  +4 0000  +6 0000 | (SP)= 0014  (IP)= 0004  Stack:  +0 0000  +2 119C  +4 0000  +6 0000 |
| 0004 | MOV AX, 11AE | B8AE11 | (AX)=0000  (IP)=0004 | (AX)= 11AE  (IP)=0007 |
| 0007 | MOV DS, AX | 8ED8 | (DS)=119C  (IP)= 0007  DS:0000  CD 20 CC 46 00 EA FD FF AD DE ED 04 92 01 00 00 | (DS)=11AE  (IP)= 0009  DS:0000  00 00 00 00 00 04 06 08 0A 0D 0B 09 07 50 32 1E |
| 0009 | MOV AX, 01F4 | B8F401 | (AX)=11AE  (IP)=0009 | (AX)=01F4  (IP)=000C |
| 000C | 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 | (BX)=0024  (IP)=0010 | (BX)=CE24  (IP)=0012 |
| 0012 | MOV [0002], FFCE | C7060200CEFF | (IP)=0012  DS:0002 = 00  DS:0003 = 00 | (IP)=0018  DS:0002 = CE  DS:0003 = FF |
| 0018 | MOV BX, 0006 | BB0600 | (BX)=CE24  (IP)=0018 | (BX)=0006  (IP)=001B |
| 001B | MOV [0000], AX | A30000 | (IP)=001B  (AX)=01F4  DS:0000 = 00  DS:0001 = 00 | (IP)=001E  (AX)=01F4  DS:0000 = F4  DS:0001 = 01 |
| 001E | MOV AL, [BX] | 8A07 | (AX)=01F4  (BX)=0006  (IP)=001E | (AX)=0104  (BX)=0006  (IP)=0020 |
| 0020 | MOV AL, [BX+03] | 8A4703 | (AX)=0104  (BX)=0006  (IP)=0020 | (AX)=010A  (BX)=0006  (IP)=0023 |
| 0023 | MOV CX, [BX+03] | 8B4F03 | (CX)=01F4  (BX)=0006  (IP)=0023 | (CX)=0D0A  (BX)=0006  (IP)=0026 |
| 0026 | MOV DI, 0002 | BF0200 | (DI)=0000  (IP)=0026 | (DI)=0002  (IP)=0029 |
| 0029 | MOV AL, [DI+000E] | 8A850E00 | (AX)=010A  (DI)=0002  (IP)=0029 | (AX)=01B0  (DI)=0002  (IP)=002D |
| 002D | MOV CX, [DI+000E] | 8B8D0E00 | (CX)=0D0A  (DI)=0002  (IP)=002D | (CX)=E2B0  (DI)=0002  (IP)=0031 |
| 0031 | MOV BX, 0003 | BB0300 | (BX)=0006  (IP)=0031 | (BX)=0003  (IP)=0034 |
| 0034 | MOV AL, [BX+DI+0016] | 8A811600 | (AX)=01B0  (BX)=0003  (DI)=0002  (IP)=0034 | (AX)=0102  (BX)=0003  (DI)=0002  (IP)=0038 |
| 0038 | MOV CX, [BX+DI+0016] | 8B891600 | (CX)=E2B0  (BX)=0003  (DI)=0002  (IP)=0038 | (CX)=FF02  (BX)=0003  (DI)=0002  (IP)=003C |
| 003C | MOV AX, 11AE | B8AE11 | (AX)=0102  (IP)=003C | (AX)=11AE  (IP)=003F |
| 003F | MOV ES, AX | 8EC0 | (ES)=11AE  (AX)=11AE  (IP)=003F | (ES)=11AE  (AX)=11AE  (IP)=0041 |
| 0041 | MOV AX, ES:[BX] | 268B07 | (AX)=11AE  (ES)=11AE  (BX)=0003  (IP)=0041 | (AX)=00FF  (ES)=11AE  (BX)=0003  (IP)=0044 |
| 0044 | MOV AX, 0 | B80000 | (AX)=00FF  (IP)=0044 | (AX)=0000  (IP)=0047 |
| 0047 | MOV ES, AX | 8EC0 | (ES)=11AE  (AX)=0000  (IP)=0047 | (ES)=0000  (AX)=0000  (IP)=0049 |
| 0049 | PUSH DS | 1E | (IP)=0049  (SP)= 0014  Stack:  +0 0000  +2 119C  +4 0000  +6 0000 | (IP)=004A  (SP)= 0012  Stack:  +0 11AE  +2 0000  +4 119C  +6 0000 |
| 004A | POP ES | 07 | (IP)=004A  (SP)= 0012  (ES)=0000  Stack:  +0 11AE  +2 0000  +4 119C  +6 0000 | (IP)=004B  (SP)= 0014  (ES)=11AE  Stack:  +0 0000  +2 119C  +4 0000  +6 0000 |
| 004B | MOV CX, ES:[BX-01] | 268B4FFF | (CX)=FF02  (ES)=11AE  (BX)=0003  (IP)=004B | (CX)=FFCE  (ES)=11AE  (BX)=0003  (IP)=004F |
| 004F | XCHG AX, CX | 91 | (CX)=FFCE  (AX)=0000  (IP)=004F | (CX)=0000  (AX)=FFCE  (IP)=0050 |
| 0050 | MOV DI, 0002 | BF0200 | (DI)=0002  (IP)=0050 | (DI)=0002  (IP)=0053 |
| 0053 | MOV ES:[BX+DI], AX | 268901 | (BX)=0003  (DI)=0002  (IP)=0053  ES:0005 = 00  ES:0006 = 04 | (BX)=0003  (DI)=0002  (IP)=0056  ES:0005 = CE  ES:0006 = FF |
| 0056 | MOV BP, SP | 8BEC | (BP)=0000  (SP)= 0014  (IP)=0056 | (BP)=0014  (SP)= 0014  (IP)=0058 |
| 0058 | PUSH [0000] | FF360000 | (IP)=0058  (SP)=0014  DS:0000 = F4  DS:0001 = 01  Stack:  +0 0000  +2 119C  +4 0000  +6 0000 | (IP)=005C  (SP)=0012  DS:0000 = F4  DS:0001 = 01  Stack:  +0 01F4  +2 0000  +4 119C  +6 0000 |
| 005C | PUSH [0002] | FF360200 | (IP)=005C  (SP)=0012  DS:0002 = CE  DS:0003 = FF  Stack:  +0 01F4  +2 0000  +4 119C  +6 0000 | (IP)=0060  (SP)=0010  DS:0002 = CE  DS:0003 = FF  Stack:  +0 FFCE  +2 01F4  +4 0000  +6 119C |
| 0060 | MOV BP, SP | 8BEC | (IP)=0060  (SP)=0010  (BP)=0014 | (IP)=0062  (SP)=0010  (BP)=0010 |
| 0062 | MOV DX, [BP+02] | 8B5602 | (DX)=0000  (BP)=0010  (IP)=0062 | (DX)=01F4  (BP)=0010  (IP)=0065 |
| 0065 | RET Far | CB | (IP)=0065  (SP)=0010  (CS)=11B1  Stack:  +0 FFCE  +2 01F4  +4 0000  +6 119C | (IP)=FFCE  (SP)=0014  (CS)=01F4  Stack:  +0 0000  +2 119C  +4 0000  +6 0000 |

**Вывод.**

По итогам лабораторной работы были изучены режимы адресации основной памяти в языке Ассемблер x86.

**ПРИЛОЖЕНИЕ**

Код программы 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(?); 24 байта

AStack ENDS

; Данные программы

DATA SEGMENT

; Директивы описания данных

mem1 DW 0

mem2 DW 0

mem3 DW 0

vec1 DB 4,6,8,10,13,11,9,7

vec2 DB 80,30,-80,-30,-60,-50,60,50

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

DATA ENDS

; Код программы

CODE SEGMENT

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

; Головная процедура

Main PROC FAR

push DS; сохраняет текущее значение регистра DS в стеке.

sub AX,AX; очищает регистр AX.

push AX; сохраняет очищенный регистр AX в стеке.

mov AX,DATA; загружает адрес сегмента данных в AX.

mov DS,AX; устанавливает регистр DS на адрес сегмента данных.

; 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

Листинг LR2\_COMP.LST(до)

Microsoft (R) Macro Assembler Version 5.10 9/27/24 12:59:14

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 04 06 08 0A 0D 0B vec1 DB 4,6,8,10,13,11,9,7

09 07

000E 50 1E B0 E2 C4 CE vec2 DB 80,30,-80,-30,-60,-50,60,50

3C 32

0016 FA 04 03 F9 F8 02 matr DB -6,4,3,-7,-8,2,-1,1,-2,-3,-4,8,

7,6,5,-5

FF 01 FE FD FC 08

07 06 05 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 СМЕЩЕНИ

Й

; Регистровая адресация

0009 B8 01F4 mov ax,n1

Microsoft (R) Macro Assembler Version 5.10 9/27/24 12:59:14

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 9/27/24 12:59:14

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

47798 + 459462 Bytes symbol space free

2 Warning Errors

5 Severe Errors

Листинг LR2\_COMP.LST(после)

Open procedures: Microsoft (R) Macro Assembler Version 5.10 9/25/24 16:18:52

Page 1-1

MAIN . . . . . . . . . . . . . .

Microsoft (R) Macro Assembler Version 5.10 9/25/24 16:18:52

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 04 06 08 0A 0D 0B vec1 DB 4,6,8,10,13,11,9,7

09 07

000E 50 1E B0 E2 C4 CE vec2 DB 80,30,-80,-30,-60,-50,60,50

3C 32

0016 FA 04 03 F9 F8 02 matr DB -6,4,3,-7,-8,2,-1,1,-2,-3,-4,8,

7,6,5,-5

FF 01 FE FD FC 08

07 06 05 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 СМЕЩЕНИ

Й

; Регистровая адресация

0009 B8 01F4 mov ax,n1

Microsoft (R) Macro Assembler Version 5.10 9/25/24 16:18:52

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

;Main ENDP

0066 CODE ENDS

END Main

Open procedures:

MAIN . . . . . . . . . . . . . .

Microsoft (R) Macro Assembler Version 5.10 9/25/24 16:18:52

Page 1-3

Microsoft (R) Macro Assembler Version 5.10 9/25/24 16:18:52

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 = 0000

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

47798 + 459462 Bytes symbol space free

2 Warning Errors

1 Severe Errors

LR2\_COMP.MAP

Start Stop Length Name Class

00000H 00017H 00018H ASTACK

00020H 00045H 00026H DATA

00050H 000B5H 00066H CODE

Program entry point at 0005:0000