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

Студент гр. 1381 _____ Тарасов К.О.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Задание

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.

Порядок выполнения 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.

3. Снова протранслировать программу и скомпоновать загрузочный модуль.

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

5. Результаты прогона программы под управлением отладчика должны быть подписаны преподавателем и представлены в отчете

Вариант 1

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,-5

Ход работы:

При трансляции программы были обнаружены следующие ошибки:

1) mov mem3,[bx] lr2.asm(41): error A2052: Improper operand type

Неподходящий тип операнда. Способ использования некоторого операнда препятствует формированию операционного кода

2) mov cx,vec2[di] lr2.asm(49): warning A4031: Operand types must match

Типы операндов должны совпадать. Ассемблер обнаружил разные виды или размерности аргументов в той ситуации, в которой предполагается их соответствие

3) mov cx,matr[bx][di] lr2.asm(53): warning A4031: Operand types must match

Типы операндов должны совпадать. Ассемблер обнаружил разные виды или размерности аргументов в той ситуации, в которой предполагается их соответствие

4) mov ax,matr[bx*4][di] lr2.asm(54): error A2055: Illegal register value Недопустимое значение регистра

5) mov ax,matr[bp+bx] lr2.asm(73): error A2046: Multiple base registers

Попытка использовать несколько базовых регистров для адресации, что недопустимо

6) mov ax,matr[bp+di+si] lr2.asm(74): error A2047: Multiple index registers

Попытка использовать несколько индексных регистров для адресации, что недопустимо.

Начальное значение регистров:

CS = 1A0A

DS = 19F5

ES = 19F5

SS = 1A05

Все ошибки за комментированы в файле lr2_fix.asm

Результат прогона программы представлена в таблице 1:

Табл. 1

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памятие	
			До выполнения	После выполнения
0000	PUSH DS	1E	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0018 IP = 0000 Stack +0 0000	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0016 IP = 0001 Stack +0 19F5
0013	SUB AX,AX	2BC0	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0016 IP = 0001 Stack +0 19F5	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0016 IP = 0003 Stack +0 19F5
0003	PUSH AX	50	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0016 IP = 0003 Stack +0 19F5	AX = 0000 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0014 IP = 0004 Stack +0 0000 Stack +2 19F5
0004	MOV AX,1A07	B071A	AX = 0000 DX = 0000 CX = 00B0 BX = 0000	AX = 1A07 DX = 0000 CX = 00B0 BX = 0000

			DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0014 IP = 0004 Stack +0 0000 Stack +2 19F5	DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0014 IP = 0007 Stack +0 0000 Stack +2 19F5
0007	MOV DS,AX	8ED8	AX = 1A07 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 19F5 CS = 1A0A ES = 19F5 SP = 0014 IP = 0007 Stack +0 0000 Stack +2 19F5	AX = 1A07 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0009 Stack +0 0000 Stack +2 19F5
0009	MOV AX,01F4	B8F401	AX = 1A07 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0009 Stack +0 0000 Stack +2 19F5	AX = 01F4 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 000C Stack +0 0000 Stack +2 19F5
000C	MOV CX,AX	8BC8	AX = 01F4 DX = 0000 CX = 00B0 BX = 0000 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 000C Stack +0 0000 Stack +2 19F5	AX = 01F4 DX = 0000 CX = 01F4 BX = 0000 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 000E Stack +0 0000 Stack +2 19F5
000E	MOV BL,24	B324	AX = 01F4 DX = 0000 CX = 01F4 BX = 0000 DI = 0000 DS = 1A07	AX = 01F4 DX = 0000 CX = 01F4 BX = 0024 DI = 0000 DS = 1A07

			CS = 1A0A ES = 19F5 SP = 0014 IP = 000E Stack +0 0000 Stack +2 19F5	CS = 1A0A ES = 19F5 SP = 0014 IP = 0010 Stack +0 0000 Stack +2 19F5
0010	MOV BH,CE	B7CE	AX = 01F4 DX = 0000 CX = 01F4 BX = 0024 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0010 Stack +0 0000 Stack +2 19F5	AX = 01F4 DX = 0000 CX = 01F4 BX = CE24 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0012 Stack +0 0000 Stack +2 19F5
0012	MOV [0002], FFCE	C7060200CEFF	AX = 01F4 DX = 0000 CX = 01F4 BX = CE24 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0012 Stack +0 0000 Stack +2 19F5	AX = 01F4 DX = 0000 CX = 01F4 BX = CE24 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0018 Stack +0 0000 Stack +2 19F5
0018	MOV BX,0006	BB0600	AX = 01F4 DX = 0000 CX = 01F4 BX = CE24 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0018 Stack +0 0000 Stack +2 19F5	AX = 01F4 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 001B Stack +0 0000 Stack +2 19F5
001B	MOV [0000],AX	A30000	AX = 01F4 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5	AX = 01F4 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5

			SP = 0014 IP = 001B Stack +0 0000 Stack +2 19F5	SP = 0014 IP = 001E Stack +0 0000 Stack +2 19F5
001E	MOV AL,[BX]	8A07	AX = 01F4 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 001E Stack +0 0000 Stack +2 19F5	AX = 0101 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0020 Stack +0 0000 Stack +2 19F5
0020	MOV AL, [BX+03]	8A4703	AX = 0101 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0020 Stack +0 0000 Stack +2 19F5	AX = 0104 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0023 Stack +0 0000 Stack +2 19F5
0023	MOV CX, [BX+03]	8B4FO3	AX = 0104 DX = 0000 CX = 01F4 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0023 Stack +0 0000 Stack +2 19F5	AX = 0104 DX = 0000 CX = 0804 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0026 Stack +0 0000 Stack +2 19F5
0026	MOV DI,0002	BF0200	AX = 0104 DX = 0000 CX = 0804 BX = 0006 DI = 0000 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0026	AX = 0104 DX = 0000 CX = 0804 BX = 0006 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0029

			Stack +0 0000 Stack +2 19F5	Stack +0 0000 Stack +2 19F5
0029	MOV AL, [000E+DI]	8A850E00	AX = 0104 DX = 0000 CX = 0804 BX = 0006 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0029 Stack +0 0000 Stack +2 19F5	AX = 010A DX = 0000 CX = 0804 BX = 0006 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 002D Stack +0 0000 Stack +2 19F5
002D	MOV BX,0003	BB0300	AX = 010A DX = 0000 CX = 0804 BX = 0006 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 002D Stack +0 0000 Stack +2 19F5	AX = 010A DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0030 Stack +0 0000 Stack +2 19F5
0030	MOV AL, [0016+BX+DI]	8A811600	AX = 010A DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0030 Stack +0 0000 Stack +2 19F5	AX = 01FD DX = 0000 CX = 1804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0034 Stack +0 0000 Stack +2 19F5
0034	MOV AX,1A07	B8071A	AX = 01FD DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0034 Stack +0 0000 Stack +2 19F5	AX = 1A07 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0037 Stack +0 0000 Stack +2 19F5

0037	MOV ES,AX	8EC0	AX = 1A07 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 19F5 SP = 0014 IP = 0037 Stack +0 0000 Stack +2 19F5	AX = 1A07 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0039 Stack +0 0000 Stack +2 19F5
0039	MOV AX,ES: [BX]	268D07	AX = 1A07 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0039 Stack +0 0000 Stack +2 19F5	AX = 00FF DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 003C Stack +0 0000 Stack +2 19F5
003C	MOV AX,0000	B80000	AX = 00FF DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 003C Stack +0 0000 Stack +2 19F5	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 003F Stack +0 0000 Stack +2 19F5
003F	MOV ES,AX	8ECO	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 003F Stack +0 0000 Stack +2 19F5	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 0000 SP = 0014 IP = 0041 Stack +0 0000 Stack +2 19F5
0041	PUSH DS	1E	AX = 0000 DX = 0000	AX = 0000 DX = 0000

			CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 0000 SP = 0014 IP = 0041 Stack +0 0000 Stack +2 19F5	CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 0000 SP = 0012 IP = 0042 Stack +0 1A07 Stack +2 0000 Stack +4 19F5
0042	POP ES	07	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 0000 SP = 0012 IP = 0042 Stack +0 1A07 Stack +2 0000 Stack +4 19F5	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0043 Stack +0 0000 Stack +2 19F5
0043	MOV CX,ES: [BX-01]	268B4FFF	AX = 0000 DX = 0000 CX = 0804 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0043 Stack +0 0000 Stack +2 19F5	AX = 0000 DX = 0000 CX = FFCE BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0047 Stack +0 0000 Stack +2 19F5
0047	XCHG AX,CX	91	AX = 0000 DX = 0000 CX = FFCE BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0047 Stack +0 0000 Stack +2 19F5	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0048 Stack +0 0000 Stack +2 19F5
0048	MOV DI,0002	BF0200	AX = FFCE DX = 0000	AX = FFCE DX = 0000

			CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0048 Stack +0 0000 Stack +2 19F5	CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 004B Stack +0 0000 Stack +2 19F5
004B	MOV ES: [BX+DI],AX	268901	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 004B Stack +0 0000 Stack +2 19F5	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 004E Stack +0 0000 Stack +2 19F5
004E	MOV BP,SP	8BEC	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 004E BP = 0000 Stack +0 0000 Stack +2 19F5	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0050 BP = 0014 Stack +0 0000 Stack +2 19F5
0050	PUSH [0000]	FF360000	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0014 IP = 0050 BP = 0014 Stack +0 0000 Stack +2 19F5	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0012 IP = 0054 BP = 0014 Stack +0 01F4 Stack +2 0000 Stack +4 19F5
0054	PUSH [0002]	FF360200	AX = FFCE	AX = FFCE

			DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0012 IP = 0054 BP = 0014 Stack +0 01F4 Stack +2 0000 Stack +4 19F5	DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0010 IP = 0058 BP = 0014 Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5
0058	MOV BP,SP	8BEC	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0010 IP = 0058 BP = 0014 Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0010 IP = 005A BP = 0010 Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5
005A	MOV DX, [BP+02]	8B5602	AX = FFCE DX = 0000 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0010 IP = 005A Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5	AX = FFCE DX = 01F4 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A ES = 1A07 SP = 0010 IP = 005D Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5
005D	RET Far 0002	CA0200	AX = FFCE DX = 01F4 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 1A0A	AX = FFCE DX = 01F4 CX = 0000 BX = 0003 DI = 0002 DS = 1A07 CS = 01F4

			ES = 1A07 SP = 0010 IP = 005D Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5	ES = 19F5 SP = 0016 IP = FFCE Stack +0 19F5 Программа не завершилась

Выводы.

В ходе выполнения работы были изучены режимы адресации и формирования исполнительного адресации

ПРИЛОЖЕНИЕ А

Текст компонентов программы lr2-fix.asm

lr2-fix.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 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,-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
```

ПРИЛОЖЕНИЕ Б

Текст компонентов программы lr2-fix.lst

lr2-fix.lst:

```
#Microsoft (R) Macro Assembler Version 5.10              9/30/22 13:34:45
                                                        Page     1-1


                        ; PµCЂPsPiCЂP°PjPjP° PëP·CґC‡PµPSPëCЏ
CЂPµP¶PëP
                        jPsPI P°PґCЂPµCЃP°C†PëPë PїCЂPsC†PµCЃCЃPsCЂP°
I
                        ntelX86

= 0024                          EOL EQU '$'
= 0002                          ind EQU 2
= 01F4                          n1 EQU 500
=-0032                          n2 EQU -50

                        ; PЎC,PµPє PїCЂPsPiCЂP°PjPjC‹
0000                    AStack SEGMENT STACK
0000  000C[                             DW 12 DUP(?)
        ????
                    ]

0018                    AStack ENDS
                        ; P”P°PSPSC‹Pµ PїCЂPsPiCЂP°PjPjC‹

0000                    DATA SEGMENT
                        ; P”PëCЂPµPєC,PëPIC‹ PsPїPëCЃP°PSPëCЏ PґP°PSPSC
                        ‹C…
```

```
0000  0000                    mem1 DW 0
0002  0000                    mem2 DW 0
0004  0000                    mem3 DW 0
0006  01 02 03 04 08 07       vec1 DB 1,2,3,4,8,7,6,5
    06 05
000E  F6 EC 0A 14 E2 D8            vec2 DB -10,-20,10,20,-30,-40,30,40
    1E 28
0016  01 02 03 04 FC FD            matr DB 1,2,3,4,-4,-3,-2,-1,5,6,7,8,-8,
                        -7,-6,-5
    FE FF 05 06 07 08
    F8 F9 FA FB
0026                    DATA ENDS

                        ; PљPsPґ PїCЋPsPiCЋP°PjPjC‹
0000                    CODE SEGMENT
                            ASSUME CS:CODE, DS:DATA, SS:AStack

                        ; P“PsP»PsPIPSP°CЏ PїCЋPsC†PµPґCґCЋP°
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

                        ; PџP РћP’P•P РљРђ P P•P–PѕPњPћP’ РђP”P P•PЎРђP
                        ¦PѕPѕ РќРђ PJP РћP’РќP• PЎРњP•P©P•РќPѕP™
                        ; P PµPiPёCЃC,CЋPsPIP°CЏ P°PґCЋPµCЃP°C†PёCЏ
0009  B8 01F4                     mov ax,n1
000C  8B C8                       mov cx,ax
```

```
000E  B3 24                     mov bl,EOL
0010  B7 CE                     mov bh,n2

#Microsoft (R) Macro Assembler Version 5.10          9/30/22 13:34:45
                                                     Page     1-2


                        ; PџCЂCЏPjP°CЏ P°PґCЂPµCЃP°C†PëCЏ
0012  C7 06 0002 R FFCE        mov mem2,n2
0018  BB 0006 R                mov bx,OFFSET vec1
001B  A3 0000 R                 mov mem1,ax

                        ; PЉPsCЃPIPµPSPSP°CЏ P°PґCЂPµCЃP°C†PëCЏ
001E  8A 07                     mov al,[bx]
                        ;       mov mem3,[bx]

                        ; P‘P°P·PëCЂPsPIP°PSPSP°CЏ P°PґCЂPµCЃP°C†PëCЏ
0020  8A 47 03                  mov al,[bx]+3
0023  8B 4F 03                  mov cx,3[bx]

                        ; P℘PSPґPµPєCЃPSP°CЏ P°PґCЂPµCЃP°C†PëCЏ
0026  BF 0002                   mov di,ind
0029  8A 85 000E R              mov al,vec2[di]
                        ;       mov cx,vec2[di]

                        ; PђPґCЂPµCЃP°C†PëCЏ CЃ
P±P°P·PëCЂPsPIP°PSPëPµP
                        j Pë PëPSPґPµPєCЃPëCЂPsPIP°PSPëPµPj
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]

                        ; РџР РћР’Р•Р РљРђ Р Р•Р–РѕРњРћР’ РђР"Р Р•РЎРђР
                        ¦РѕРѕ РЎ PJP§Р•РўРћРњ РЎР•Р“РњР•РќРўРћР’
                        ; РџРµСЂРµРsРїСЂРµРтРµР»РµРSРёРµ
СЃРµРiPjPµPSC,
                        Р°
                        ; ------ PIP°СЂРёР°PSC, 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

                        ; ------ PIP°СЂРёР°PSC, 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

                        ; ------ PIP°СЂРёР°PSC, 3
0048  BF 0002                           mov di,ind
004B  26: 89 01                 mov es:[bx+di],ax

                        ; ------ PIP°СЂРёР°PSC, 4
004E  8B EC                             mov bp,sp
                        ;       mov ax,matr[bp+bx]
                        ;       mov ax,matr[bp+di+si]
```

```
                        ; P‰CЃPїPsP»CЊP·PsPIP°PSPёPµ CЃPµPiPjPµPSC,P° C
                        ЃC,PµPєP°
 0050  FF 36 0000 R                  push mem1
#Microsoft (R) Macro Assembler Version 5.10          9/30/22 13:34:45
                                     Page     1-3


 0054  FF 36 0002 R                  push mem2
 0058  8B EC                        mov bp,sp
 005A  8B 56 02                     mov dx,[bp]+2
 005D  CA 0002                      ret 2
 0060                  Main  ENDP
 0060                  CODE ENDS
                             END Main
#Microsoft (R) Macro Assembler Version 5.10          9/30/22 13:34:45
                                     Symbols-1


Segments and Groups:

                N a m e         Length         AlignCombine Class

ASTACK . . . . . .  .           0018  PARA         STACK
CODE . . . . . .  . . . .  .    0060  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 = 0060
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_fix
@VERSION . . . . . .    TEXT  510


    99 Source  Lines
    99 Total   Lines
    19 Symbols

 47798 + 459462 Bytes symbol space free

     0 Warning Errors
     0 Severe  Errors
```