

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. 0383

Позолотин К.С.

Преподаватель

Ефремов М.А.

Санкт-Петербург 2021

Цель работы.

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. 6

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

Вариант 10:

vec1 38,37,36,35,31,32,33,34

vec2 70,80,-70,-80,50,60,-50,-60

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

Выполнение работы.

При трансляции программы были обнаружены ошибки:

- `mov mem3,[bx] lr2.asm(46): error A2052: Improper operand type`
Попытка положить данные из одной ячейки памяти в другую, что недопустимо. Перемещать данные можно только между регистрами или между регистрами и ячейками памяти.
- `mov cx,vec2[di] lr2.asm(53): warning A4031: Operand types must match`
Попытка положить данные из ячейки памяти размером 1 байт в регистр размером 2 байт. Размеры операндов не совпадают.
- `mov cx,matr[bx][di] lr2.asm(57): warning A4031: Operand types must match`
Попытка положить данные из ячейки памяти размером 1 байт в регистр размером 2 байт. Размеры операндов не совпадают.
- `mov ax,matr[bx*4][di] lr2.asm(58): error A2055: Illegal register value`
Недопустимое значение регистра
Попытка использовать несколько индексных регистров для адресации, что недопустимо.
- `mov ax,matr[bp+bx] lr2.asm(78): error A2046: Multiple base registers`

Попытка использовать несколько базовых регистров для адресации, что недопустимо.

- `mov ax,matr[bp+di+si] lr2.asm(79): error A2047: Multiple index registers`

6.

Попытка использовать несколько индексных регистров для адресации, что недопустимо.

Начальное содержимое сегментных регистров: (CS) = 1A0A, (DS) = 19F5, (ES) = 19F5, (SS) = 1A05

Строки, содержащие ошибки, были закомментированы в файле `lr2_fixed.asm`.

Таблица 2. Протокол выполнения программы `lr2_fixed.asm`

| Адрес команды | Символический код команды | 16-ричный код команды | Содержимое регистров и ячеек памяти | |
|---------------|---------------------------|-----------------------|---|---|
| | | | До выполнения | После выполнения |
| 0000 | push ds | 1E | (AX) = 0000 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 (CS) = 1A0A (ES) = 19F5 (SP) = 0018 (IP) = 0000 Stack +0 0000 | (AX) = 0000 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 (CS) = 1A0A (ES) = 19F5 (SP) = 0016 (IP) = 0001 Stack +0 0000 |
| 0001 | sub ax, ax | 2BC0 | (AX) = 0000 | (AX) = 0000 |

| | | | | |
|--|--|--|-------------|-------------|
| | | | (DX) = 0000 | (DX) = 0000 |
| | | | (CX) = 0000 | (CX) = 0000 |
| | | | (BX) = 0000 | (BX) = 0000 |
| | | | (DI) = 0000 | (DI) = 0000 |
| | | | (DS) = 19F5 | (DS) = 19F5 |
| | | | (CS) = 1A0A | (CS) = 1A0A |
| | | | (ES) = 19F5 | (ES) = 19F5 |
| | | | (SP) = 0016 | (SP) = 0016 |

| | | | | |
|------|--------------|--------|---|--|
| | | | (IP) = 0001 Stack +0 19F5 | (IP) = 0003 Stack +0 19F5 |
| 0003 | push ax | 50 | (AX) = 0000 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 (CS) = 1A0A (ES) = 19F5 (SP) = 0016 (IP) = 0003 Stack +0 19F5 | (AX) = 0000 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0004 Stack +0 0000 Stack +2 19F5 |
| 0004 | mov ax, 1A07 | B8071A | (AX) = 0000 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 | (AX) = 1A07 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 19F5 |

| | | | | |
|------|------------|------|--|--|
| | | | (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0004 Stack +0 0000 Stack +2 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) = 0000 | (AX) = 1A07 (DX) = 0000 (CX) = 0000 |

| | | | | |
|------|--------------|--------|--|--|
| | | | (BX) = 0000 (DI) = 0000 (DS) = 19F5 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0007 Stack +0 0000 Stack +2 19F5 | (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) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0009 | (AX) = 01F4 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 000C |

| | | | | |
|------|------------|------|--|--|
| | | | Stack +0 0000 Stack +2 19F5 | Stack +0 0000 Stack +2 19F5 |
| 000C | Mov cx, ax | 8BC8 | (AX) = 01F4 (DX) = 0000 (CX) = 0000 (BX) = 0000 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 | (AX) = 01F4 (DX) = 0000 (CX) = 01F4 (BX) = 0000 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 |

| | | | | |
|------|------------|------|--|--|
| | | | (SP) = 0014 (IP) = 000C Stack +0 0000 Stack +2 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 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 000E Stack +0 0000 Stack +2 19F5 | (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 |
| 0010 | Mov bh, CE | B7CE | (AX) = 01F4 (DX) = 0000 | (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 | (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 | (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 | (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 | (AX) = 01F4 (DX) = 0000 (CX) = 01F4 (BX) = 0006 (DI) = 0000 (DS) = 1A07 |

| | | | | |
|------|----------------|--------|--|--|
| | | | (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0018 Stack +0 0000 Stack +2 19F5 | (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 | (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 | (CS) = 1A0A (ES) = 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 | (AX) = 011F (DX) = 0000 (CX) = 01F4 (BX) = 0006 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0020 |

| | | | | |
|------|-----------------|--------|---|---|
| | | | Stack +0 0000 Stack +2 19F5 | Stack +0 0000 Stack +2 19F5 |
| 0020 | Mov al, [bx+03] | 8A4703 | (AX) = 011F (DX) = 0000 (CX) = 01F4 (BX) = 0006 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0020 Stack +0 0000 | (AX) = 0122 (DX) = 0000 (CX) = 01F4 (BX) = 0006 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0023 Stack +0 0000 |

| | | | | |
|------|-----------------|--------|--|--|
| | | | Stack +2 19F5 | Stack +2 19F5 |
| 0023 | Mov cx, [bx+03] | 8B4F03 | (AX) = 0122 (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) = 0122 (DX) = 0000 (CX) = 2622 (BX) = 0006 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0026 Stack +0 0000 Stack +2 19F5 |
| 0026 | Mov di, 0002 | DF0200 | (AX) = 0122 (DX) = 0000 | (AX) = 0122 (DX) = 0000 |

| | | | | |
|------|-------------------|----------|--|--|
| | | | (CX) = 2622 (BX) = 0006 (DI) = 0000 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0026 Stack +0 0000 Stack +2 19F5 | (CX) = 2622 (BX) = 0006 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0029 Stack +0 0000 Stack +2 19F5 |
| 0029 | Mov al, [000E+di] | 8A850E00 | (AX) = 0122 (DX) = 0000 (CX) = 2622 (BX) = 0006 | (AX) = 01CE (DX) = 0000 (CX) = 2622 (BX) = 0006 |

| | | | | |
|------|--------------|--------|--|--|
| | | | (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0029 Stack +0 0000 Stack +2 19F5 | (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 002D Stack +0 0000 Stack +2 19F5 |
| 002D | Mov bx, 0003 | BB0300 | (AX) = 01CE (DX) = 0000 (CX) = 2622 (BX) = 0006 (DI) = 0002 (DS) = 1A07 | (AX) = 01CE (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 |

| | | | | |
|------|-------------------------|----------|---|---|
| | | | (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 002D Stack +0 0000 Stack +2 19F5 | (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0030 Stack +0 0000 Stack +2 19F5 |
| 0030 | Mov al, [0016+bx+di] | 8A811600 | (AX) = 01CE (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 | (AX) = 01FF (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 |

| | | | | |
|------|--------------|--------|--|--|
| | | | (IP) = 0030 Stack +0 0000 Stack +2 19F5 | (IP) = 0034 Stack +0 0000 Stack +2 19F5 |
| 0034 | Mov ax, 1A07 | B8071A | (AX) = 01FF (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0034 | (AX) = 1A07 (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 19F5 (SP) = 0014 (IP) = 0037 |

| | | | | |
|------|-----------------|--------|--|--|
| | | | Stack +0 0000 Stack +2 19F5 | Stack +0 0000 Stack +2 19F5 |
| 0037 | Mov es, ax | 8ECO | (AX) = 1A07 (DX) = 0000 (CX) = 2622 (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) = 2622 (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] | 268B07 | (AX) = 1A07 (DX) = 0000 | (AX) = 00FF (DX) = 0000 |

| | | | | |
|------|--------|--------------|--|--|
| | | | (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0014 (IP) = 0039 Stack +0 0000 Stack +2 19F5 | (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0014 (IP) = 003C Stack +0 0000 Stack +2 19F5 |
| 003C | B80000 | Mov ax, 0000 | (AX) = 00FF (DX) = 0000 | (AX) = 0000 (DX) = 0000 |

| | | | | |
|------|------------|------|--|--|
| | | | (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0014 (IP) = 003C Stack +0 0000 Stack +2 19F5 | (CX) = 2622 (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) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A | (AX) = 0000 (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A |

| | | | | |
|------|---------|----|--|--|
| | | | (ES) = 1A07 (SP) = 0014 (IP) = 003F Stack +0 0000 Stack +2 19F5 | (ES) = 0000 (SP) = 0014 (IP) = 0041 Stack +0 0000 Stack +2 19F5 |
| 0041 | Push ds | 1E | (AX) = 0000 (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 | (AX) = 0000 (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 |

| | | | | |
|------|--------|----|---|---|
| | | | (CS) = 1A0A (ES) = 0000 (SP) = 0014 (IP) = 0041 Stack +0 0000 Stack +2 19F5 | (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) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 0000 (SP) = 0012 (IP) = 0042 Stack +0 1A07 | (AX) = 0000 (DX) = 0000 (CX) = 2622 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0014 (IP) = 0043 Stack +0 0000 |
| | | | Stack +2 0000 Stack +4 19F5 | Stack +2 19F5 |

| | | | | |
|------|--------------------|----------|--|--|
| 0043 | Mov cx, es:[bx-01] | 268B4FFF | (AX) = 0000 (DX) = 0000 (CX) = 2622 (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 (CX) = 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 | (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 | (AX) = FFCE (DX) = 0000 (CX) = 0000 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 |

| | | | | |
|------|-------------|----------|---|--|
| | | | (SP) = 0014 (BP) = 0000 (IP) = 004E Stack +0 0000 Stack +2 19F5 | (SP) = 0014 (BP) = 0014 (IP) = 0050 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 (BP) = 0014 (IP) = 0050 Stack +0 0000 Stack +2 19F5 | (AX) = FFCE (DX) = 0000 (CX) = 0000 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0012 (BP) = 0014 (IP) = 0054 Stack +0 01F4 Stack +2 0000 Stack +4 19F5 |

| | | | | |
|------|-------------|----------|--|--|
| 0054 | Push [0002] | FF360200 | (AX) = FFCE (DX) = 0000 (CX) = 0000 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0012 (BP) = 0014 | (AX) = FFCE (DX) = 0000 (CX) = 0000 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0010 (BP) = 0014 |
| | | | (IP) = 0054 Stack +0 01F4 Stack +2 0000 Stack +4 19F5 | (IP) = 0058 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 (BP) = 0014 (IP) = 0058 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 (BP) = 0010 (IP) = 005A 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 | (AX) = FFCE (DX) = 01F4 (CX) = 0000 (BX) = 0003 (DI) = 0002 (DS) = 1A07 (CS) = 1A0A (ES) = 1A07 (SP) = 0010 |

| | | | | |
|------|--------------|--------|---|--|
| | | | (BP) = 0010 (IP) = 005A Stack +0 FFCE Stack +2 01F4 Stack +4 0000 Stack +6 19F5 | (BP) = 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 (ES) = 1A07 (SP) = 0010 (BP) = 0010 (IP) = 005D 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) = 01F4 (ES) = 1A07 (SP) = 0016 (BP) = 0010 (IP) = FFCE Stack +0 19F5 |

Тексты исходных файлов программ см. в приложении А.

Тексты файлов диагностических сообщений см. в приложении Б.

Выводы.

В ходе выполнения данной лабораторной работы был изучен механизм работы с адресами Ассемблера.

ПРИЛОЖЕНИЕ А

ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **lr2.asm**

; Программа изучения режимов адресации процессора IntelX86

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 38,37,36,35,31,32,33,34

vec2 DB 70,80,-70,-80,50,60,-50,-60

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

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

```

Название файла: **lr2_fixed.asm**

; Программа изучения режимов адресации процессора IntelX86

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 38,37,36,35,31,32,33,34
vec2      DB 70,80,-70,-80,50,60,-50,-60
matr  DB      -2,-1,5,6,-8,-7,3,4,-4,-3,7,8,-6,-5,1,2
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
ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **lr2.lst**

Microsoft (R) Macro Assembler Version 5.10

9/29/21 23:08:41

Page 1-1

; Программа изучения режии
¼ов адресации процессора I
ntelX86

| | |
|--------|--------------|
| = 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 данны |
| x | |
| 0000 0000 | mem1 DW 0 |

```

0002 0000          mem2   DW   0
0004 0000          mem3   DW   0
0006 26 25 24 23 1F 20vec1  DB  38,37,36,35,31,32,33,34
      21 22
000E 46 50 BA B0 32 3C      vec2  DB  70,80,-70,-80,50,60,-50,-60
      CE C4
0016 FE FF 05 06 F8 F9      matr  DB  -2,-1,5,6,-8,-7,3,4,-4,-3,7,8,-
      6,-5,1,2
      03 04 FC FD 07 08
      FA FB 01 02
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 АДРЕСА
ЦИИ НА УРОВНЕ СМЕЩЕНИЙ

; Регистровая адресация

```

0009 B8 01F4          mov ax,n1
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

```

Microsoft (R) Macro Assembler Version 5.10

9/29/21 23:08:41

Page 1-2

```

001B A3 0000 R          mov mem1,ax

```

; Косвенная адресация

```

001E 8A 07              mov al,[bx]
                        mov mem3,[bx]

```

lr2.asm(46): error A2052: Improper operand type

; Базированная адресация

```

0020 8A 47 03          mov al,[bx]+3
0023 8B 4F 03          mov cx,3[bx]

```

; Индексная 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.asm(53): warning A4031: Operand types must match

; Адресация с базирование

м 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.asm(57): warning A4031: Operand types must match
003C 8B 85 0022 R      mov ax,matr[bx*4][di]

```

lr2.asm(58): error A2055: Illegal register value

```

; ПРОВЕРКА РЕЖИМОВ АДРЕСА
ЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегментов
□a

```

; ----- 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.asm(78): error A2046: Multiple base registers

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

lr2.asm(79): 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 CA 0002 ret 2

0076 Main ENDP

lr2.asm(86): error A2006: Phase error between passes

0076 CODE ENDS

 END Main

Microsoft (R) Macro Assembler Version 5.10

9/29/21 23:08:41

Symbols-1

Segments and Groups:

| N a m e | Length | Align | Combine | Class |
|--------------|--------|-------|---------|-------|
| ASTACK | 0018 | PARA | | STACK |
| CODE | 0076 | 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 = 0076 |
| 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 | 1r2 | |
| @VERSION | TEXT | 510 | |

88 Source Lines

88 Total Lines

19 Symbols

47842 + 459418 Bytes symbol space free

2 Warning Errors

5 Severe Errors

```

; Программа изучения режИЙ
;¼ов адресации процессора I
ntelX86

= 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 данны
x
0000 0000            mem1    DW  0
0002 0000            mem2    DW  0
0004 0000            mem3    DW  0
0006 26 25 24 23 1F 20vec1  DB  38,37,36,35,31,32,33,34
    21 22
000E 46 50 BA B0 32 3C    vec2  DB  70,80,-70,-80,50,60,-50,-60
    CE C4
0016 FE FF 05 06 F8 F9    matr  DB  -2,-1,5,6,-8,-7,3,4,-4,-3,7,8,-
    6,-5,1,2
    03 04 FC FD 07 08
    FA FB 01 02
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 АДРЕСА
ЦИИ НА УРОВНЕ СМЕЩЕНИЙ

```

```

; Регистровая адресация

```

```

0009 B8 01F4          mov ax,n1
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

```

```

Microsoft (R) Macro Assembler Version 5.10

```

9/29/21 23:29:25

Page 1-2

```

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]

```

```

; Индексная адресация

```

```

0026 BF 0002          mov di,ind
0029 8A 85 000E R     mov  al,vec2[di]

```

```

;mov cx,vec2[di]

```

```

; Адресация с базирование

```

```

м и индексированием

```

```

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]

```

```

; ПРОВЕРКА РЕЖИМОВ АДРЕСА

```

ЦИИ С УЧЕТОМ СЕГМЕНТОВ

; Переопределение сегментов

□ a

; ----- вариант 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

; ----- вариант 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

; ----- вариант 3

0048 BF 0002 mov di, ind

004B 26: 89 01 mov es:[bx+di], ax

; ----- вариант 4

004E 8B EC mov bp, sp

 ;mov ax, matr[bp+bx]

 ;mov ax, matr[bp+di+si]

; Использование сегмента

стека

0050 FF 36 0000 R push mem1

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

9/29/21 23:29:25

Symbols-1

Segments and Groups:

| N a m e | Length | Align | Combine | 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_fix1 | |
| @VERSION | TEXT | 510 | |

88 Source Lines

88 Total Lines

19 Symbols

47800 + 459460 Bytes symbol space free

0 Warning Errors

0 Severe Errors