

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Институт прикладной математики и компьютерных наук
Кафедра «Вычислительная техника»

Утверждено на заседании кафедры
«Вычислительная техника»
«26» января 2022 г., протокол № 7

Заведующий кафедрой

_____ А.Н. Ивутин

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению лабораторных работ
по дисциплине (модулю)**

«Операционные системы и среды»

**основной профессиональной образовательной программы
высшего образования – программы бакалавриата**

по направлению подготовки
09.03.01 Информатика и вычислительная техника

с направленностью (профилем)
Электронно-вычислительные машины, комплексы, системы и сети

Форма обучения: заочная

Идентификационный номер образовательной программы: 090301-02-22

Тула 2022 год

Разработчик(и):

Андриянова М.А., доцент каф. ВТ, доцент, канд. техн. наук

(ФИО, должность, ученая степень, ученое звание)

(подпись)

Лабораторная работа №1 Работа с файлами в Linux

Цель работы: знакомство с командной строкой Linux, командами по работе с файловой системой и командами обработки текстовых файлов.

Часть 1. Знакомство с каталогами Linux

Объекты и средства исследования: Объекты исследования: **pwd, cd, ls, mkdir**. Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **pwd, cd, ls, mkdir**

Программа работы:

Начальные условия: Командная строка после входа в систему.

1. Получить имя текущего каталога.
2. Перейти в корневой каталог. Отметить, как изменилась строка приглашения.
3. Посмотреть содержимое корневого каталога.
4. Сравнить с использованием «расширенного» вывода содержимого каталога.
5. Посмотреть содержимое домашнего каталога.
6. Вернуться в домашний каталог.
7. Создать каталог test_n, где n – номер студента по журналу.
8. Посмотреть обновлённое содержимое домашнего каталога.
9. Создать подкаталог subtest в каталоге test_n.
10. Посмотреть содержимое домашнего каталога и его подкаталогов с помощью ключа рекурсивного просмотра **-R**.

Часть 2. Изучение типов файлов в Linux

Объекты и средства исследования: Объекты исследования: **touch, echo, cat, alias, cp, mv, ln**. Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **touch, echo, cat, alias, cp, mv, ln**.

Программа работы:

Начальные условия: Командная строка. Каталог test_n.

1. Создать пустой текстовый файл.
2. Добавить строку текста в конец файла с помощью команды **echo** и перенаправления вывода.
3. Посмотреть содержимое файла.
4. Посмотреть расширенную информацию о каталоге, используя ключ **-l** команды **ls**.
5. Добавить псевдоним командной оболочки, чтобы сократить размер команды.
6. Посмотреть глубже на каталог test_n, используя ключи **-a** и **-i**. Обратить внимание: первый столбец — номера индексных узлов файловой системы; третий столбец — число жёстких ссылок файла.
7. Скопировать файл. Обратить внимание: новый файл имеет свой собственный индексный узел.
8. Переименовать исходный файл.
9. Создать жёсткую ссылку на переименованный файл. Обратить внимание: добавилась ещё одна ссылка на тот же файл, число ссылок увеличилось на 1.
10. Создать символическую ссылку. Обратить внимание: новый файл имеет новый индексный узел и размер.
11. Сравнить содержимое файлов при обращении к ним по имени. К одним и тем же данным обратиться и через жесткую и через символическую ссылку.

Часть 3. Поиск системных журналов

Объекты и средства исследования: Объекты исследования: **find**. Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **find**.

Программа работы:

Начальные условия: Командная строка, каталог test_n.

1. Вывести список всех файлов и каталогов в текущей директории, включая содержимое подкаталогов.
2. Найти все файлы и директории в текущем каталоге и ее подкаталогах, которые начинаются на «о».
3. Найти все файлы и каталоги в каталоге /etc и его подкаталогах, которые начинаются на «о». Обратить внимание: список найденных файлов может быть слишком большим и для его просмотра можно воспользоваться «прокруткой» терминала с помощью клавиш **Shift-PgUp** и **Shift-PgDn**.
4. Множество сообщений об ошибках доступа («Permission denied») необходимо подавить с помощью перенаправления ошибок.
5. Найти все каталоги в /etc, которые начинаются на «о»
6. Найти все обычные файлы в каталоге /var и его подкаталогах, заканчивающиеся на «log».
7. Создать каталог logs_n, где n – номер студента по списку.
8. Скопировать найденные файлы в локальный каталог с помощью параметра -ehes команды find. Обратить внимание: были скопированы только файлы, на которые есть права на чтение.

Часть 4. Архивирование и деархивирование файлов и каталогов

Объекты и средства исследования: Объекты исследования: **tar**. Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих программ: **tar**, **gzip**, **bzip2**

Программа работы:

Начальные условия: Командная строка.

1. Перейти в домашний каталог.
2. Создать архив с именем test_n.tar.gz с применением сжатия, содержащий каталог test_n.
3. Создать новый каталог для содержимого архива.
4. Перейти в новый каталог.
5. Развернуть содержимое архива в текущий каталог.
6. Убедиться, что содержимое распакованного архива совпадает с оригинальным каталогом.

Лабораторная работа №2

Работа с файлами в Linux

Цель работы: знакомство с командной строкой Linux, командами по работе с файловой системой и командами обработки текстовых файлов.

Часть 5. Создание новых текстовых файлов

Объекты и средства исследования: Объекты исследования: **echo**, **cat**.

Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **echo**, **cat**

Программа работы:

Начальные условия: Командная строка.

1. Создать текстовый файл **second.txt**, содержащий одну строку текста.
2. Добавить в конец файла **second.txt** строку «**A line**».
3. Вывести содержимое файла на экран.
4. С помощью команды **cat** можно создавать многострочные файлы, если переопределить вывод программы в файл и вводить текст до нажатия **Ctrl-D** (конец ввода). Создать файл **multiline.txt**, введя соответствующую команду и набрав текст.
5. Убедиться, что содержимое файла совпадает с введенным текстом, включая все переводы строки.
6. Основным назначением команды **cat** является объединение файлов, имена которых передаются как аргументы командной строки. Объединить созданные файлы в файл **big.txt**.
7. Убедиться, что новый файл содержит строки из перечисленных файлов.

Часть 6. Разрезание и склеивание файлов

Объекты и средства исследования: Объекты исследования: **cat**, **sort**.

Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **cat**, **sort**.

Программа работы:

Начальные условия: Командная строка.

1. Посмотреть содержимое файла **/etc/passwd**, в котором содержится информация о пользователях системы. Обратить внимание: каждая строка хранит запись об отдельном пользователе. Запись состоит из полей, разделённых символом «:».
2. Выделить первый столбец в каждой записи, установив разделитель в «:».
3. Отсортировать список пользователей по алфавиту.
4. Выделить седьмой столбец в каждой записи, соответствующий командной оболочке пользователя. Обратить внимание: список содержит много повторяющихся строк.
5. Удалить повторяющиеся строки. Примечание: некоторые повторяющиеся строки остались, так как сравниваются только рядом стоящие строки.
6. Добиться полной уникальности, предварительно отсортировав строки с помощью команды **sort**.
7. Выделить первый и седьмой столбец файла **/etc/passwd** в отдельные файлы **users** и **shells**.
8. Объединить два получившихся файла (должно получиться два столбца).

Часть 7. Быстрый анализ текстов

Объекты и средства исследования: Объекты исследования: **wc, head, tail**.

Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **wc, head, tail**.

Программа работы:

Начальные условия: Командная строка.

1. Получить число строк в файле **/etc/passwd**.
2. Получить число строк в файле **/etc/passwd** другим способом: посчитав число слов в созданном ранее файле **users**.
3. Посмотреть первые несколько строк длинного файла.
4. Аналогично, посмотреть последние несколько строк.

Примечание: Это бывает полезно при просмотре системных журналов — в них самые последние события содержатся в конце файлов.

Часть 8. Поиск строк по регулярным выражениям

Объекты и средства исследования: Объекты исследования: **less, grep**.

Средства исследования: операционная систем Linux

Подготовка к работе

Изучить описание и синтаксис следующих команд: **less, grep**

Программа работы:

Начальные условия: Командная строка. Каталог **test/log** с найденными системными журналами.

1. Среди найденных системных журналов посмотреть файл журнала загрузки графической системы **XFree86.0.log**. *Примечание:* выйти из просмотрщика можно нажатием клавиши **q**.
2. Найти все предупреждения в журнале: строка, начинающаяся с «(WW)».
3. Получить все строки, содержащие следующий шаблон: последовательность из одной или более цифры или буквы, затем символ собаки, затем еще одна последовательность одной или более цифры или буквы или точки, затем символ точки, затем от двух до четырёх букв (имя домена первого уровня). Также служебный вывод ошибок удаляется (перенаправляется в нулевое устройство).
Примечание: при работе с несколькими файлами **grep** по умолчанию использует вывод имени файла перед найденной строкой.
4. Избавиться от имени файла в начале строки с помощью ключа **-h**.
5. Для вывода только найденной подстроки используется ключ **-o**.
6. Получить все файлы каталога **/etc/**, игнорируя ошибки.
7. Для каждого из найденных файлов произвести поиск подстроки «**nameserver**».

Для большего удобства добавить вывод имени файла (ключ **-H**) и номера строки (ключ **-n**), на которой было найдено совпадение. *Обратить внимание:* команда **grep** может эффективно использоваться вместе с командой **find** для нахождения информации в файлах.

Лабораторная работа № 3
ОПРЕДЕЛЕНИЕ КОНТЕКСТА ПРОЦЕССА.
ЧАСТЬ 1.

РАЗРАБОТКА И ОТЛАДКА ПРОГРАММ ПОД УПРАВЛЕНИЕМ ОТЛАДЧИКА DEBUG

Цель и задачи работы: Целью работы является закрепление теоретических знаний о возможностях программирования на языке Ассемблер. Задачами работы является приобретение навыков трансляции и отладки программ под управлением отладчика DEBUG.

ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Команды отладчика DEBUG:

R [имя регистра] изобразить содержимое одного или всех регистров

T [число команд] исполнить заданное число команд и изобразить содержимое регистров на каждом шаге

G [смещение] исполнить программу до точки, заданной смещением

D адрес изобразить содержимое ячеек памяти. Адрес задается в виде: регистр: смещение от начала сегмента(обычно смещение равно 0). Регистр название регистра, где хранится начальный адрес сегмента.

Q выйти из отладчика

ОБОРУДОВАНИЕ

IBM совместимые ЭВМ; эмулятор DOSBox, создающий DOS-окружение; оболочки Norton Commander, Volcov Commander, FAR, Windows Commander, программы Masm, Link, Debug.

ЗАДАНИЕ НА РАБОТУ

4.1. Повторить шестнадцатичную систему счисления.

4.2. Изучить структуру программы на Ассемблере.

4.3. Изучить команды отладчика DEBUG.

4.4. Внимательно ознакомиться с порядком выполнения работы.

4.5. Разобраться в представленной ниже программе, записывающей исходные данные типа байт из области source в обратном порядке в область dest.

title ex_prog - заголовок программы

page ,132

stack segment para stack 'stack' - заголовок сегмента стека

db 64 dup('stack ')

stack ends - конец сегмента стека

dseg segment para public 'data' - заголовок сегмента данных

source db 10,20,30,40 - исходные данные

dest db 4 dup(?)

dseg ends - конец сегмента данных

cseg segment para public 'code' - заголовок сегмента команд

prog proc far - оформление программы в виде процедуры

assume cs:cseg,ds:dseg,ss:stack- соответствия между регистрами и сегментами

push ds - запоминание контекста предыдущего процесса

mov ax,0

push ax

mov ax,dseg

mov ds,ax

```

mov    dest,0      - обнуление дополнительной области данных
mov    dest+1,0
mov    dest+2,0
mov    dest+3,0

mov    al,source   - запись первого числа области source на четвертую
позицию области dest
mov    dest+3,al
mov    al,source+1 - запись второго числа области source на третью
позицию области dest
mov    dest+2,al
mov    al,source+2 - запись третьего числа области source на вторую
позицию области dest
mov    dest+1,al
mov    al,source+3 - запись четвертого числа области source на первую
позицию области dest
mov    dest,al
ret     - возврат в запомненное ранее состояние
prog endp - конец процедуры
cseg ends - конец сегмента команд
end our_prog - конец программы

```

4.6. Скопировать программу PRIMER.ASM под другим именем.

4.7. Внести изменения в программу в зависимости от номера варианта.

№ варианта	Кол-во входных данных	Значения входных данных
1	4	25, 33, 45, 56
2	4	20, 40, 60, 70
3	5	15, 20, 25, 30, 35
4	5	10, 11, 12, 13, 14
5	4	5, 15, 45, 55
6	4	15, 16, 17, 63
7	5	21, 23, 25, 27, 31
8	5	45, 55, 56, 67, 70
9	5	40, 50, 60, 100, 200
10	4	30, 130, 230, 150
11	5	41, 42, 43, 44, 45
12	5	71, 72, 73, 81, 83
13	4	101, 102, 103, 110
14	5	51, 55, 61, 65, 71
15	4	49, 59, 69, 79

4.7. Внести изменения в программу, поменяв тип исходных данных на слово.

4.8. Внести изменения в программу, поменяв тип исходных данных на двойное слово.

Программа работы

5.1. Создать на диске C: папку asm, переписать в нее файлы masm, link, debug, PRIMER.ASM и папку vc.

5.2. Запустить DOSBox, примонтировать папку asm как логический диск, сделать примонтированный диск текущим.

5.3. Запустить файловый менеджер командой `vs\vc`.

5.4. В любом редакторе исправить программу, вводя исходные данные своего варианта. В первой программе тип данных - байт.

5.5. Оттранслировать исходный модуль. Для этого набрать команду:

`>masm {имя файла},`

где {имя файла} - имя файла, содержащего исходный модуль, без расширения.

На запросы ответить так:

`object filename[ ]`: нажать клавишу перевода строки

`source listing[ ]`: набрать имя файла без расширения

`cross reference[ ]`: нажать клавишу перевода строки

В результате создается объектный файл и листинг трансляции.

5.6. Просмотр листинга трансляции. Для этого вызвать на экран файл с расширением LST клавишей F3. Листинг состоит из двух частей. В левой части находится столбец шестнадцатиричных значений смещения адреса (в байтах) от начала сегмента и столбцы объектного кода каждого оператора. В правой части - текст исходной программы.

Особое внимание обратить на смещение оператора RET.

5.7. Создать исполняемый файл. Для ввода загрузчика ввести команду:

`>link {имя файла}`

На все запросы нажимать клавишу перевода строки.

В результате создается исполняемый файл с расширением EXE.

5.8. Выполнить программу. Для этого используется отладчик DEBUG. Для начала ввести команду:

`>debug {имя файла}.exe`

Просмотреть содержимое регистров. Сделать 5-6 шагов. Выполнить программу до конца, используя смещение команды RET.

5.9. Проверить правильность работы программы. Просмотреть содержимое сегмента данных.

5.10. Просмотреть содержимое сегмента команд.

5.11. Выйти из отладчика.

5.12. Повторить пункты 5.4-5.11, переопределив тип данных – слово.

Не забудьте исправить текст программы.

5.13. Повторить пункты 5.4-5.11, переопределив тип данных – двойное слово и исправив текст программы.

6. ОФОРМЛЕНИЕ ОТЧЕТА.

Отчет должен включать для трех программ:

6.1. Текст листинга трансляции (2 страницы).

6.2. Содержимое регистров после команд отладчика R и T.

6.3. Содержимое сегмента данных и сегмента команд.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ.

7.1. Как перевести число из одной системы счисления в другую?

7.2. Из каких основных частей состоит программа на Ассемблере?

7.3. Как оформляется сегмент данных?

7.4. Как оформляется сегмент команд?

7.5. Какие псевдооператоры данных вы знаете?

7.6. Как работает оператор MOV?

7.7. Как получить листинг трансляции?

7.8. Как получить исполняемый модуль?

7.9. В каких случаях используется отладчик DEBUG?

7.10. Перечислите команды отладчика.

- 7.11. Как просмотреть содержимое сегментов?
- 7.12. Сколько и какие регистры вы знаете?
- 7.13. В чем особенность регистров данных?

ЧАСТЬ 2. ИССЛЕДОВАНИЕ ФОРМИРОВАНИЯ РЕГИСТРА ФЛАГОВ

1. Цель и задачи работы: Целью и задачей работы является приобретение навыков работы с регистром флагов при выполнении команд сложения и сравнения.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

2.1. Флаги

В 16-ти битовом регистре флагов фиксируется информация о текущем состоянии дел (см.рис.10.1).

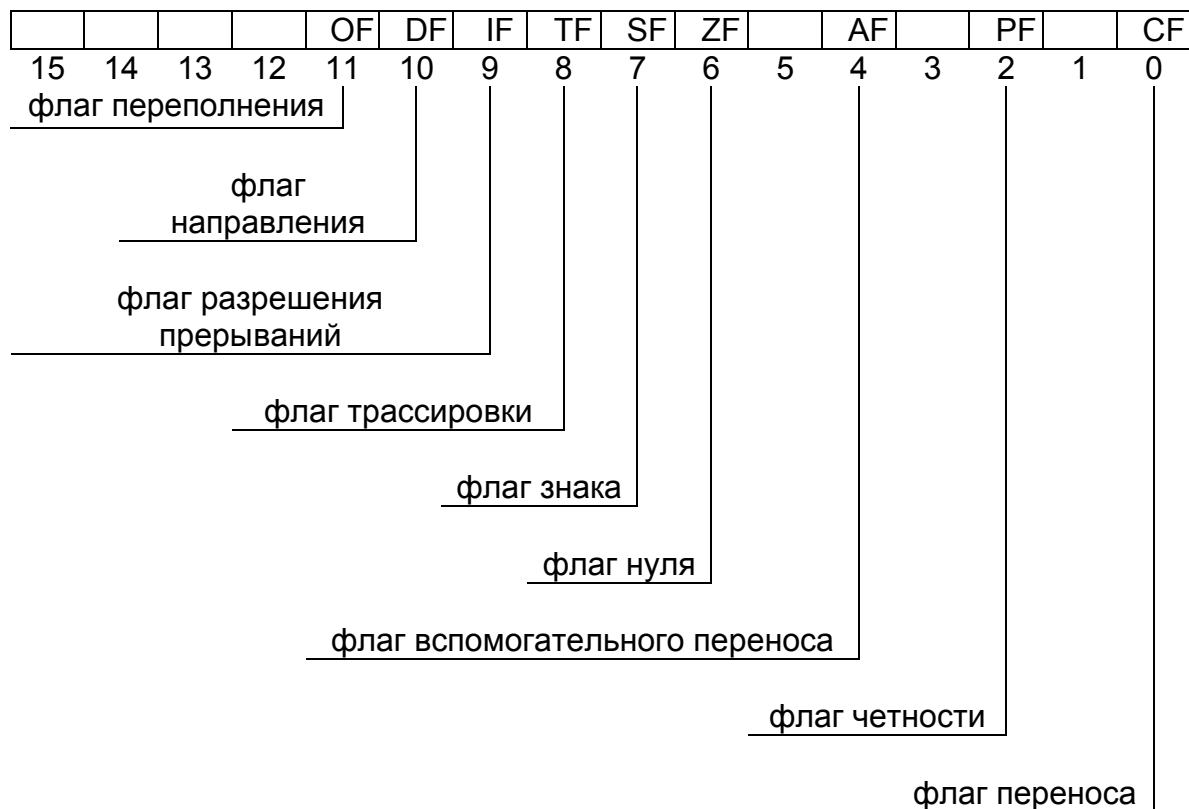


Рисунок 10.1 - Регистр флагов

С помощью команды LAHF (Load AH from Flags – загрузить регистр AH флагами) можно сохранить флаги CF,PF,AF,ZF,SF в соответствующих битах регистра AH (0,2,4,6,7).

2.2. Команды сложения.

ADD приемник, источник (add-сложить)

ADC приемник, источник (add with carry - сложить с переносом)

Команды могут складывать как 8-, так и 16-битовые операнды.

Если при выполнении сложения ADD используются 8-битовые регистры, то 8 младших битов ответа будут занесены в регистр приемника, а 9-й бит (если он есть) - во флаг переноса CF, например

```
TABL DB 250,10
```

```
.....
```

```
MOV AL, TABL
```

```
MOV BL, TABL+1
```

```
ADD AL, BL
```

.....

Команда ADD AX,BX складывает 16-битовые значения регистров AX,BX.

Команды

ADD AX,CX

ADC BX,DX

складывают 32-битовое число, находящееся в регистрах CX, DX с 32 -битовым числом, находящимся в регистрах AX и BX .

2.3. Команда сравнения CMP (compare).

Команда CMP вычитает операнд-источник из операнда-приемника, устанавливает флаги (см. табл.7.1), но не сохраняет результат вычитания.

Таблица 7.1 - Результаты исполнения команды CMP

условие	SF	ZF	CF
приемник > источник	0	0	0
приемник = источник	0	1	0
приемник < источник	1	0	1

3. ОБОРУДОВАНИЕ

Объект исследования - программа на языке Ассемблера. Средства исследования - программные модули Masm/Tasm, Link/Tlink, Debug/Turbo Debug, и IBM совмещенная ЭВМ.

4. ЗАДАНИЕ НА РАБОТУ

4.1. Повторить команды сложения (ADD и ADC), сравнения (CMP) и сохранения флагов (LAHF).

4.2. Составить программу, состоящую из трех частей. В первой части сложить первое и второе числа из последовательности входных данных в соответствии с вариантом и посмотреть состояние флагов. Во второй части сравнить 2 и 3, 2 и 4, 3 и 4 числа из последовательности и также посмотреть состояния флагов после каждого сравнения. Тип данных и используемые регистры выбрать самостоятельно, обосновав выбор.

№ варианта	Входные данные
1	240, 40, 32, 40
2	-60, 20, 10, 20
3	31100, 2010, 50, 2010
4	3015, 105, 100, 105
5	250, -40, 5, -40
6	60000, 6000, 20, 6000
7	-100, 50, 100, 50
8	200, 70, 150, 70
9	2000, 150, 200, 150
10	100, 180, 200, 180

5. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

- 5.1. Набрать программу.
- 5.2. Откомпилировать и отладить программу.
- 5.3. Оформить отчет.

6. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать:

- 6.1. Вариант задания.
- 6.2. Текст программы.
- 6.3. Содержание регистров после выполнения операции сложения и результат сложения в шестнадцатиричном, двоичном и десятичном виде.
- 6.4. Содержание регистра АН в шестнадцатиричном и двоичном виде и состояние флагов после каждого выполнения команды LARH.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ

- 7.1. Какие команды пересылки флагов вы знаете?
- 7.2. Какие команды сложения вы знаете?
- 7.3. Что из себя представляет команда сравнения CMP?
- 7.4. Какими способами можно узнать состояние флага CF?
- 7.5. Что означают состояния флагов?

Лабораторная работа № 4

СПОСОБЫ АДРЕСАЦИИ

1. Цель и задачи работы: Целью и задачей работы является приобретение навыков составления и отладки простейших программ на языке Ассемблера с использованием различных типов адресации.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Смещение, которое вычисляется операционным блоком для доступа к находящемуся в памяти операнду, называется исполнительным адресом (ИА) операнда.

Типы адресации и их форматы представлены в таблице 4.1

Таблица 4.1

Тип адресации	Формат	Пример
1. Регистровая	регистр, регистр	MOV AX,CX
2. Непосредственная	регистр, значение	MOV CL,500
3. Прямая	регистр, ИА регистр, ИА+С	T DB 5, 4 MOV AL,T
4. Косвенная регистровая	регистр, [BX] регистр, [BP] регистр, [SI] регистр, [DI]	T DB 5 T1 DB 4 MOV BX,OFFSET T MOV AL,[BX] MOV BX,OFFSET T1 MOV AL,[BX]
5. По базе	регистр, [BX]+С регистр, [BP]+С	T DB 5,4 ... MOV BX,OFFSET T MOV AL,[BX] MOV AL,[BX]+1
6. Прямая с индексированием	регистр, ИА[DI] регистр, ИА[SI]	TABL DB 5,4,3,2 ... MOV DI,2 MOV AL,TABL[DI]
7. По базе с индексированием	регистр, [BP+ИР+С]	T DB 5,4,3,2,1 MOV DI,1 MOV BX,OFFSET T MOV AL,[BX+DI+2] MOV AL,[DI+BX+2] MOV AL,[BX+2][DI] MOV AL,[BX][DI+2]

3. ОБОРУДОВАНИЕ

Объектом исследования является программа на языке Ассемблер, IBM совместимые ЭВМ, операционная система MS DOS/Windows, оболочки Norton Commander, Volcov Commander, FAR, Windows Commander, программы Masm/Tasm, Link/Tlink, Debug/Turbo Debugger.

4. ЗАДАНИЕ НА РАБОТУ

- 4.1. Повторить типы адресации и команду пересылки MOV языка Ассемблер.
- 4.2. Повторить порядок отладки программы.
- 4.3. Задание:

В сегменте данных определить три области: А – 1 байт под одно число; В – по 1 байту под три числа; С – по 1 байту под шесть чисел.

В программе

- Обнулить область А.
- В область В записать три любых числа (прямая адресация).
- В первые три ячейки области С записать числа из области В в обратном порядке (адресация по базе).
- В оставшиеся ячейки области С записать три любых числа.
- Просмотреть сегмент данных.
- В области С поменять местами 1 и 4, 2 и 5, 3 и 6 числа (адресация по базе с индексированием). В качестве буфера можно использовать или регистр ВХ или область А.

5. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

- 5.1. Набрать программу в любом редакторе.
- 5.2. Откомпилировать и отладить программу.
- 5.3. Оформить отчет

6. ОФОРМЛЕНИЕ ОТЧЕТА.

Отчет должен включать:

- 6.1. Вариант задания
- 6.2. Тексты программ
- 6.3. Строку сегмента данных с полученным результатом.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ.

- 7.1. Что такое прямая адресация?
- 7.2. Что такое косвенная регистровая адресация?
- 7.3. Что такое адресация по базе?
- 7.4. Что такое адресация по базе с индексированием?
- 7.5. Что такое прямая адресация с индексированием?
- 7.6. Что такое исполнительный адрес?
- 7.7. Чем прямая адресация отличается от косвенной?
- 7.8. Как работает операция OFFSET?