**Федеральное агентство связи**

**БФ ГОУ ВПО СибГУТИ**

**КУРСОВАЯ РАБОТА**

**По дисциплине:** ОСНОВЫ СИСТЕМНОГО ПРОГРАММИРОВАНИЯ

**Выполнил: Проверил:**

**ст-ка гр. преподаватель**

**Кудинов Н.Н.**

**Содержание**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Разраб.

Провер.

Кудинов Н.Н.

Программирование на языке Assembler

Лит.

Листов

20

Т-291

Задание на курсовой проект…………………………………………………….3

Введение………………………………………………………………………….4

Теоретическая часть.

1. Состав и назначения ОС………………………………………………………5

2. Характеристики современных ОС……………………………………………7

3. ОС MS DOS……………………………………………………………...…….8

4. Язык Ассемблера………………………………………………………...…....9

5. Разработка программы на языке ассемблер………………………………..10

6. Компиляция откладка программы………………………………….……….11

Практическая часть.

Задание 1……………………………………………………………...……..…..13

Задание 2………………………………………………………………....….…..15

Блок схема………………………………………………………………..……..17

Заключение……………………………………………………………..…...…..19

Список литературы…………………………………….……………….………20

**Задания на курсовую работу**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

1. Создать командный файл, который выполняет следующие действия:

* + - 1. Очистка экрана
      2. Создание директории C:\TMP
      3. Копирование в директорию C:\TMP файлов из корневого каталога.
      4. Создание нового файла NEW.TXT с консоли в директории C:\TMP.
      5. Создание копий файлов \*.TXT, сменив расширение на bat и оставив имя.
      6. Сравнение копий с оригиналами.
      7. Просмотр постранично содержимого директории C:\TMP.

2. Написать и отладить программу на языке ассемблера. В программе описать процедуру, которая выводит на экран элемент массива с заданным номером. Параметры передавать следующим образом:

в ВХ – смещение массива;

в СХ – число элементов в массиве;

в АХ – заданный номер элемента.

В основной программе вызвать описанную процедуру для двух разных массивов.

**ВВЕДЕНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

4

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Прогресс компьютерных технологий определил процесс появления новых разнообразных знаковых систем для записи алгоритмов – языков программирования. Существует множество языков программирования, но мы остановимся на Ассемблере.

***Актуальность.***

Несмотря на то, что в настоящее время программист может использовать большое количество языков программирования, которые гораздо проще Ассемблера, знание Ассемблера и умение программировать на нем никогда не будут лишними. Этот факт подтверждается следующими доводами:

1. На других языках программирования не всегда возможно написать приложение, которое полно бы удовлетворяло требованиям. А в некоторых случаях полностью невозможно. На языке программирования Ассемблер можно написать любое приложение.

2. Иногда языки программирования высокого уровня не могут обеспечить требуемое быстродействие. А приложение созданное на Ассемблере всегда быстродействующее.

3. Размер приложений созданных на языках высокого уровня имею гораздо больший размер, чем приложения созданные на Ассемблере.

4. Язык Ассемблер позволяет работать непосредственно с аппаратными средствами, что в некоторых случаях дает программисту преимущество и требуемый результат. Приложение на языке высокого уровня работает с аппаратными устройствами через написанные модули, т.е. не позволит программисту что-либо изменить, а, следовательно, получить требуемый результат.

5. Знание языка Ассемблер дает большее преимущество перед теми, кто программирует только на языках высокого уровня. Знающий Ассемблер знает и структуру компьютера, и структуру аппаратных устройств.

***Состав и назначение ОС***

*Операционная система* (ОС) – это комплекс программ, входящих в состав программного обеспечения компьютера, обеспечивающих управление работой аппаратных средств компьютера, обменом данных между различными аппаратными узлами ПК, а также организующих диалог компьютера и человека. При параллельной работе процессора, памяти и внешних устройств ОС обеспечивает разделение ресурсов, что предотвращает возникновение конфликтов между компонентами компьютерной системы. ОС – неотъемлемая часть любого компьютера. Ни один из компонентов программного обеспечения, за исключением самой ОС, не имеет доступа к аппаратуре компьютера.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

5

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Наиболее важными частями ОС являются файловая система (ФС), драйверы внешних устройств, загрузчик, системная библиотека, сервисные программы (или утилиты), справочная система. Кроме того, в состав ОС могут входить такие программы, как текстовые редакторы, редакторы связей, системные мониторы, трансляторы и т.д.

*Файловая система* представляет собой способ организации хранения файлов в дисковой памяти. Тип ФС и организация хранения данных на носителях внешней памяти определяют удобство работы пользователя, скорость доступа к файлам, организацию многозадачной работы, возможность создания хороших баз данных. Файлы – это программы, тексты, данные, любая информация, хранящаяся в памяти, которая имеет уникальное имя. Имя каждого файла и основные сведения о нем хранятся в каталоге (директории), который упрощает поиски доступ к информации. Каталоги – это специальное место на диске, организованное для хранения имен файлов и сведений о них. Исходный каталог, в состав которого входят все остальные каталоги называется корневым каталогом. ФС поддерживает операции чтения, переименования, удаления файлов.

*Задачи ОС* заключаются в том, чтобы:

- облегчить проектирование, программирование, отладку и сопровождение программ, обеспечить их взаимодействие с аппаратурой;

- распределить ресурсы ЭВМ таким образом, чтобы обеспечить эффективную работу всех ее компонентов (центрального процессора, устройств ввода/вывода и т.п.);

- предоставить пользователям возможности общего управления машиной.

В рамках первой задачи ОС обеспечивает взаимодействие программ с внешними устройствами и друг с другом, распределение оперативной памяти, выявление различных событий, возникающих в процессе работы, и соответствующую реакцию на них (например, при ошибочных ситуациях). Общее управление машиной осуществляется на основе командного языка (языка директив), с помощью которого человек может осуществлять различные операции, например, такие, как разметка дисков, копирование файлов, запуск программ, установка режимов работы дисплея, принтера и т.п.

Главное назначение ОС – управление ресурсами компьютера. Операционная система управляет следующими *основными ресурсами*: процессорами, памятью, устройствами ввода/вывода, данными. При этом операционная система реализует следующие *функции*:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

6

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

* определяет интерфейс пользователя;
* обеспечивает разделение аппаратных средств между пользователями;
* планирует доступ пользователей к общим ресурсам;
* обеспечивает эффективное выполнение операций ввода-вывода;
* осуществляет восстановление информации и вычислительного процесса в случае ошибок;
* обеспечивает сохранность данных и защиту одной программы от другой;

Операционная система *взаимодействует с*: операторами ЭВМ, прикладными программистами, системными программистами, административным персоналом, программами, аппаратными средствами, пользователями.

*Операторы ЭВМ* – это специально подготовленные люди, которые контролируют работу ОС и в случае необходимости (поступление запроса) вмешиваются в работу компьютера для устранения каких-либо препятствий.

*Системные программисты* занимаются сопровождением ОС, осуществляют ее настройку применительно к требованиям конкретной машины и при необходимости доработку для обслуживания новых типов устройств.

*Администраторы систем* устанавливают порядок работы на ЭВМ и взаимодействуют с ОС, чтобы обеспечить соблюдение принятого порядка.

*Программы* обращаются к ОС при помощи специальных команд (вызов монитора, супервизора и т.п.), не нарушающих ее целостности и работоспособности.

*Пользователи* – это абоненты вычислительной сети.

Операционной системе, как правило, присваивается статус самого полномочного пользователя. Она имеет возможность доступа ко всем видам аппаратных ресурсов, всем программам пользователя, данным и т.п.

Пользователь взаимодействует ПК через внешний интерфейс, организуемый ОС. Он вводит задания (команды) и получает результаты их выполнения. Существует два типа диалоговых интерфейсов – текстовый (MS DOS) и графический (Windows). В графических интерфейсах информация и команды представляются в виде пиктограмм (значков), и пользователь выполняет те или иные действия, указывая на эти пиктограмм и совершая с ними определенные действия.

***Характеристики современных ОС***

Изм.

Лист

№ докум.

Подпись

Дата

Лист

7

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Современные операционные системы имеют следующие особенности:

* дружественный интерфейс, ориентированный на неподготовленного пользователя и при помощи меню предоставляющий пользователю ряд альтернатив, выраженных на естественном языке;
* использование концепции виртуальных машин, благодаря которой пользователь избавлен от необходимости знать физические особенности машин и систем; он имеет дело с функциональным эквивалентом компьютера, создаваемым для него операционной системой и называемым виртуальной машиной;
* распределенная обработка данных: гораздо целесообразнее иметь вычислительные мощности там, где они необходимы, вместо того, чтобы передавать данные для обработки в вычислительные центры.

Различные ОС обладают теми или иными возможностями по обслуживанию компонентов компьютера и организации диалога с пользователем. К числу основных характеристик ОС относят: разрядность поддержка многопроцессорности, многозадачность, поддержка многопользовательского режима.

*Разрядность ОС* определяет, какую разрядность внутренней шины данных ЦП может поддерживать ОС. Все современные ОС поддерживают 32-разрядный интерфейс прикладных программ. ОС может поддерживать два режима работы ЦП: реальный и защищенный. В *реальном режиме*, характерном для системы MS-DOS, все программы и данные располагаются в одной области ОП. Таким образом пользователь может войти в любую системную программу и испортить ее. 32-разрядные ОС поддерживают *защищенный режим* работы ЦП, который позволяет хранить программы и данные раздельно, в соответствии с их важностью в системе.

*Многопроцессорность* – это способность ОС, ЦП и системных контроллеров компьютера поддерживать одновременную работу нескольких процессоров над выполнением одной и той же задачи. ОС могут быть ориентированы на одновременное обслуживание нескольких процессов (задач). Такое свойство ОС называется *многозадачностью*. Многозадачность могут поддерживать все современные процессоры и чипсеты ПК. ЦП в определенные кванты времени выполняет работу над отдельными фрагментами задач. У пользователя складывается впечатление одновременности их выполнения. ОС обеспечивает переключение ЦП и других устройств с выполнения одной задачи на другую, распределяет между задачами системные ресурсы и синхронизирует задачи между собой.

*Переносимость* ОС – это возможность ОС работать на компьютерах, базирующихся на ЦП с различной архитектурой.

***ОС MS DOS***

Изм.

Лист

№ докум.

Подпись

Дата

Лист

8

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Операционная система MS-DOS за годы своего существования прошла путь от простого загрузчика до универсальной системы для персональных компьютеров, построенных на базе микропроцессоров Intel 8086/8088.

Операционная система MS-DOS была разработана компанией Microsoft в 1981 г. и впервые использована фирмой IBM для персональных компьютеров – тогда она была названа PC DOS 1.0. В последующие годы MS-DOS многократно перерабатывалась и приобретала новые важные функции (версии 2.0 в 1983 г., 3.0 в 1984 г., 3.2 в 1986 г., 3.3 в 1987 г., 4.0 в 1988 г., 5.0, …, 6.22). Например, в версии 4.0 появилась графическая оболочка пользователя DOS-Shell, в версии 5.0 – целый ряд сервисных программ (редактор командной строки DOSKEY, страничный редактор EDIT, и т.д.).

MS-DOS является однозадачной однопользовательской системой. Она требует относительно небольшого объема памяти, легко адаптируется к разнообразным аппаратным конфигурациям и поэтому достаточно популярна среди пользователей. Кроме того, ее можно считать основой для изучения операционных систем и их возможностей – знакомство с общей структурой системы MS-DOS очень полезно для понимания поведения вычислительной системы в целом. Многие ОС, созданные позднее, имеют пользовательский интерфейс, совместимый с MS-DOS. Например, это система MS OS/2, в которой реализованы многозадачный и защищенный режим, система виртуальной памяти.

Система MS-DOS разбита на несколько уровней, которые служат для разделения логики ядра ОС и восприятия системы пользователем от технических средств, реализующих ее работу. К этим уровням относятся:

* *BIOS* (базовая система ввода-вывода),
* *Ядро* системы DOS,
* *Командный процессор* (оболочка).

*Модуль BIOS* индивидуален для каждой вычислительной системы и поставляется ее изготовителем. В этом модуле по умолчанию резидентно

содержатся аппаратно-зависимые драйверы следующих устройств:

* Консольный дисплей с клавиатурой (CON);
* Устройство построчной печати (PRN);
* Последовательный канал связи (AUX);
* Часы/календарь (CLOCK$);
* Дисковое устройство начальной загрузки.

Ядро системы MS-DOS взаимодействует с драйверами этих устройств с помощью пакетов запросов ввода-вывода. Затем драйверы переводят эти запросы в сами команды для различных аппаратных контроллеров.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

9

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

*Ядро DOS* реализует MS-DOS, как она видится прикладным программам. *Ядро*– это специальная программа, которая включает набор аппаратно-независимых сервисных программ, называемых *системными функциями*. К ним можно отнести следующие функции:

* управление файлами и записями;
* управление памятью;
* символьно-ориентированное устройство ввода-вывода;
* порождение других задач;
* доступ к часам реального времени.

Программы могут обращаться к системным функциям путем загрузки регистров параметрами функций и последующей передачи управления операционной системе посредством программного прерывания.

*Командный процессор* (или оболочка) – это интерфейс пользователя с операционной системой. Он отвечает за анализ синтаксиса и выполнение команд пользователя, в том числе и за загрузку и выполнение других программ, находящихся на диске.

***Язык ассемблера***

*Понятие ассемблера*

Язык ассемблера позволяет лучше понять взаимодействие всех функциональных узлов компьютера с операционной системой. *Язык ассемблера* – это специфический язык программирования со взаимно однозначным соответствием между его операторами и командами процессора. Язык ассемблера существует для каждого типа процессоров или целого семейства процессоров, поскольку команды на языке ассемблера должны иметь взаимно однозначное соответствие с системой машинных команд и должны быть согласованы с архитектурой компьютера. В данном курсе рассматривается система команд для 16-разрядного 8086-88 процессоров производства Intel. Микропроцессоры 8086-88 характеризуются основным адресным пространством объемом 1 (MB) мегабайт, из которого первые 640 KB (килобайт) отведены под основную память (RAM) и адресным пространством ввода/вывода объемом 65536 байтов.

*Ассемблер* – это программа, преобразовывающая исходные коды языка ассемблера в машинные команды. Ассемблерные программы могут быть очень эффективными. Из программистов, с равными навыками и способностями, работающий на языке Ассемблера создаст программу более компактную и быстродействующую, чем такая же программа, написанная на языке высокого уровня. Это так практически для всех небольших или средних программ. Программы на языке Ассемблера очень точны. Поскольку этот язык позволяет программисту непосредственно работать со всем аппаратным обеспечением, ассемблерная программа может делать то, что недоступно никакой другой программе.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

10

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Хотя разработка и отладка программы на языке ассемблера занимают много времени, при этом получаются небольшие исполняемые модули, занимающие мало места в памяти и позволяющие достичь приемлемой скорости даже на медленных компьютерах. Язык ассемблера используется в основном для написания отдельных сегментов прикладных программ (для повышения скорости работы и прямого доступа к оборудованию), а также встроенных системных программ, которые хранятся в программируемой памяти отдельных устройств.

Главный недостаток языка ассемблера состоит в том, что написанная для одного типа компьютеров программа не может быть перекомпилирована и использована на компьютерах других типов, поскольку для каждого семейства компьютеров используется свой язык ассемблера. Если создаваемая программа должна использоваться на различных компьютерах, то ее необходимо разрабатывать на языках высокого уровня, которые скрывают от программиста специфику архитектуры компьютера для удобства использования и получения переносимого кода.

***Разработка программы на языке ассемблера***

Разработка программ на языке ассемблера отличается от написания программ на языках высокого уровня тем, что требует большого внимания и аккуратности при отслеживании содержимого памяти и регистров. При этом следует соблюдать следующие этапы разработки программы:

* постановка задачи и составление проекта программы;
* создание файла с текстом программы с помощью любого текстового редактора;
* трансляция программы с помощью ассемблера, при обнаружении ошибок – исправить их в текстовом редакторе и оттранслировать заново;
* преобразование результата работы ассемблера в исполняемый модуль с помощью компоновщика;
* запуск программы на исполнение;
* проверка результатов. В случае не соответствия необходимо найти ошибки с помощью отладчика.

Программа, написанная в кодах ассемблера, называется *исходной программой*, а ее преобразованный вид в команды микропроцессора – *объектной программой или объектным модулем*. Компоновщик позволяет создать исполняемый файл или *исполняемый модуль*. *Отладчик* – это программа, позволяющая отображать на экране значения необходимых переменных, получать состояния всех регистров и ячеек памяти при пошаговом исполнении программы, вносить изменения в программу, указывать точки останова и многое другое.

Изм.

Лист

№ докум.

Подпись

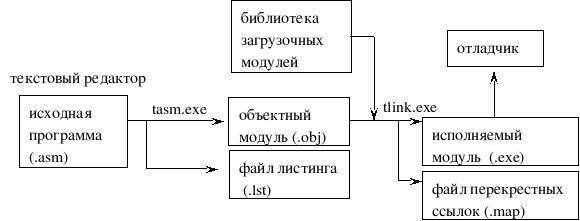
Дата

Лист

11

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Для создания программ на языке ассемблера в данном курсе используются программный продукт Турбо Ассемблер фирмы Borland Int. Компилятор Турбо Ассемблера – это выполняемая программа, размещенная в файле TASM.EXE, а компоновщик содержится в файле TLINK.EXE. Отладчик содержится в файле TD.EXE. Процесс компиляции и компоновки программы на языке ассемблера выглядит следующим образом.



**Рисунок 1 Этапы разработки программы на языке ассемблера**

***Компиляция и отладка программы***

Для удобства работы с компилятором можно создать командный файл, который содержит в себе вызов компилятор и компоновщика с соответствующими опциями. Это можно сделать следующим образом.

Создайте в текстовом редакторе (в Far Manager) текстовый файл с расширением .bat. Назовите его, например, a.bat. Наберите следующие строки (компилятор и компоновщик находятся в директории C:\ASM):

**@echo off**

**C:\ASM\tasm /zi %1.asm, %1.obj, %1.lst**

**C:\ASM\tlink /v %1.obj**

Во второй строчке вызывается компилятор с нужными опциями для файла с исходной программой, который будет указан в командной строке. В третьей строчке вызывается компоновщик для создания ехе-файла. Для компиляции программы в командной строке необходимо набрать **a.bat Myprog** (если исходная программа хранится в файле Myprog.asm)

После трансляции в текущей директории появляются файлы с расширением .lst, .map, .exe. Файл с расширением .lst содержит информацию об ошибках. Обязательно нужно просматривать файл листинга на наличие ошибок в программе, после этого необходимо исправить их и заново оттранслировать программу. Если ошибки не найдены, то можно запустить на исполнение созданный ехе-файл.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

12

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Для исправления ошибок можно воспользоваться отладчиком Turbo Debugger. В командной строке необходимо набрать **C:\АSM\td Myprog.exe** Можно оформить этот вызов в виде командного файла для удобства. После выполнения этой команды вызывается отладчик.

Просмотр содержимого регистров осуществляется командой Veiw\CPU. Пошаговое исполнение программы – F8. Выход из отладчика – ALT-X.

***Задание 1:***

Изм.

Лист

№ докум.

Подпись

Дата

Лист

13

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Выполнение:

В командной строке прописываем следующие команды:

а. cls - *очистка экрана*

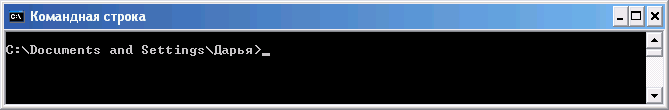


Рис.2 Очистка экрана

b. md C:\TMP – *создания директории*

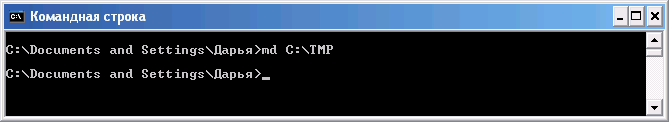


Рис.3 Создание директории

c. copy C:\\*.\* C:\Tmp - *Копирование в директорию C:\TMP файлов из корневого каталога*

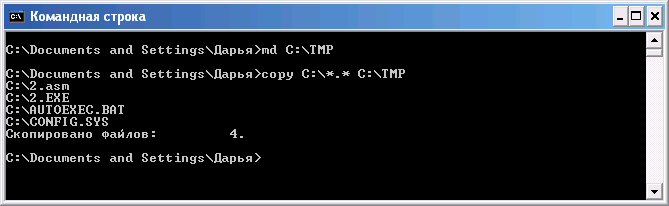


Рис.4 Копирование файлов

d. copy con C:\Tmp\new.txt – *создания файла*

*ctrl+Z – для завершения*

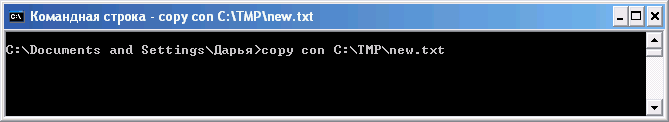


Рис.5 Создание файла

Изм.

Лист

№ докум.

Подпись

Дата

Лист

14

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

e. copy С:\Tmp\new.txt C:\Tmp\new.bat – *создания копии файла .txt c расширением .bat*

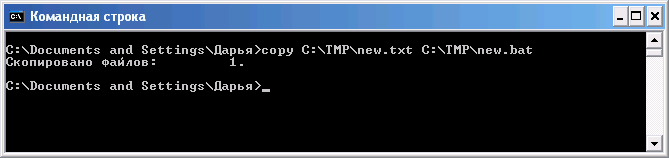


Рис.6 Создание копии файла с другим расширением (.bat)

f. comp С:\Tmp\new.txt C:\Tmp\new.bat *– сравнения файлов .txt c .bat*

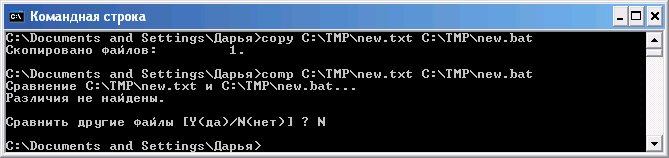


Рис.7 Сравнение файлов с различными расширениями

g. dir C:\tmp /p – *просмотр директории постранично*

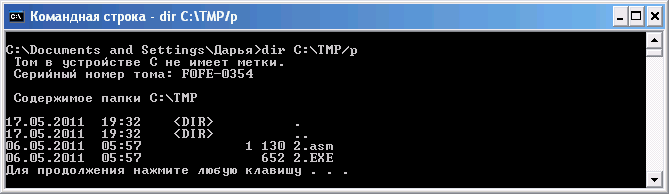


Рис.8 Просмотр директории постранично

***Задание 2:***

Изм.

Лист

№ докум.

Подпись

Дата

Лист

15

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Title main proga

.model SMALL

.data

mas dw 9,9,8,7,6,5,4,3,2,1,3,4,5 ; *первый массив*

mas1 dw 1,2,3,4,5,6,7,8,9,9 ; *второй массив*

me db "Index Massiva$"

.stack 100h ; *размер стэка*

.code

print proc ; *процедура печати - начало описания*

cmp ax,cx

jle rep1

mov dx,offset me ; *заносим смещение сообщения в регистр дх*

mov ah,09h ; *функция вывода строки*

int 21h; *прерывание вызываем*

jmp p1

rep1:

mov cx,ax ; *ставим счетчик цикла в значение ax*

rep2:

add bx,2 ; *смещаемся на следуещее слово*

loop rep2

mov dx,[bx]; *выводим элемент под нужным номером*

add dx,48 ; *переводим число в соответствующий символ*

mov ah,2 ; *функция вывода символа*

int 21h ; *вызываем прерываение*

p1:

mov ah,02h ; *выводим конец строки в конце работы процедуры*

mov dl,0Dh

int 21h

mov dl,0Ah

int 21h

ret

print endp

main proc

mov ax,@data

mov ds,ax

xor ax,ax

mov bx,offset mas ; *смещение 1го массива*

mov cx,13 ; *размер 1го массива*

mov ax,10

call print ; *печетаем*

mov bx,offset mas1 ; *смещение 2го массива*

mov cx,10 ; *и размер его*

Изм.

Лист

№ докум.

Подпись

Дата

Лист

16

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

mov ax,5

call print

mov ah,4ch ; *возврат в опер. сист*.

int 21h

main endp

end main

Блок-схема

Изм.

Лист

№ докум.

Подпись

Дата

Лист

17

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Да

Ax<Cx

Нет

Cx=Ax

Rep2

Bx=Bx[i+1]

Вывод cообщения «Индекс>Размера массива»

Rep2

Dx=Знач.по.адр(Bx)

Вывод Значения DX

Вывод перевода строки

Изм.

Лист

№ докум.

Подпись

Дата

Лист

18

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

.

Print(ax,cx,bx)

Ax=Индекс

Cx=Size(mas2)

Bx=адрес(mas2)

Print(ax,cx,bx)

Ax=Индекс

Cx=Size(mas1)

Bx=адрес(mas1)

**ЗАКЛЮЧЕНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

19

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

Язык Ассемблера - мощное средство программирования. Он позволяет программисту осуществлять всестороннее управление аппаратными средствами ЭВМ. Однако такое управление заставляет программиста вникать в детали, далекие от основного содержания программы. Все преимущества языка Ассемблера оборачиваются подчас пустой тратой времени на многочисленные детали.

Несмотря на то, что Ассемблер является машинно-ориентированным языком, то есть языком низкого уровня, программист может применять его для работы, как на высоком.

*К преимуществам Ассемблера можно отнести:*

1. Данный язык программирования позволяет создавать приложения, которые будут более эффективны, чем аналогичные приложения, написанные на языке высокого уровня, т.е. приложения будут более короткими и при этом более быстро выполнимыми.

2. Язык Ассемблера позволяет программисту выполнять действия, которые либо вообще нельзя реализовать на других языках и в частности на языках высокого уровня, либо выполнение которых займет слишком много машинного времени в случае привлечения дорогих средств языка высокого уровня.

*К недостаткам языка следует отнести:*

1. По мере увеличения своего размера программа на Ассемблере теряет наглядность. Это связано с тем, что в ассемблерных программах следует уделять много внимания деталям. Язык требует планирования каждого шага ЭВМ. Конечно, в случае небольших программ это позволяет сделать их оптимальными с точки зрения эффективности использования аппаратных средств. В случае же больших программ бесконечное число деталей может помешать добиться оптимальности программы в целом, несмотря на то, что отдельные фрагменты программы будут написаны очень хорошо.

2. Для программирования на данном языке необходимо очень хорошо знать структуру компьютера и работу аппаратных устройств, так как Ассемблер работает непосредственно с устройствами.

Можно сделать вывод, что на языке Ассемблера можно сделать любое приложение, любую программу, но для написания больших программ лучше использовать языки высокого уровня.

***Список использованной литературы:***

Изм.

Лист

№ докум.

Подпись

Дата

Лист

20

БФ ГОУ ВПО СибГУТИ ХХХХХХ.ХХХ ПЗ

* 1. Методическое пособие «Основы системного программирования»
  2. Электронный учебник по Ассемблеру

3. Абель П. Язык Ассемблера для IBM PC и программирования- М.: Высш.шк.,1992.

4. Скэнлон Л. Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера –М.: Радио и связь, 1991.

5. Зубков С.В. Ассемблер. Язык неограниченных возможностей