**ПЛАН ЗАНЯТИЯ**

**Дисциплина:** МДК.01.04 Системное программирование

**Преподаватель:** ГалузинА.Б.

**Курс:** 4

**Группа:** П-40

**Специальность:** 09.02.07 Информационные системы и программирование

**Дата:** 30.09.25

**Время проведения:** 1 пара

**Тема:** Работа с консолью в программах на ассемблере

**Цель занятия:**

**дидактическая:** изучить особенности работы с консолью в программах на ассемблере **развивающая**: развивать абстрактное мышление, логику

**Вид занятия** лекция

**Литература**

Юров В.И. Assembler. Учебник для вузов. - 2-ое изд. – СПб.: Питер, 2003, стр. 149-156.

**Интернет-ресурсы:**

<http://datadump.ru/fasm-console-application/?ysclid=m1grrz386o288102383>

<https://www.youtube.com/watch?v=cohLTLLIyIQ>

**ЗАДАНИЕ**: законспектировать лекцию с учетом контрольных вопросов.

***КОНСПЕКТ ЛЕКЦИИ***

План

1. Консольные и оконные приложения.
2. Работа с консолью в программах на ассемблере.
3. Работа с отладчиком программ TURBO DEBUGGER

*Консольное приложение* представляет собой единственное окно, работающее в текстовом режиме с минимальным интерфейсом. Как правило, интерфейс приложения представляет собой ввод и вывод текстовых строк.

Консольное приложение операционной системы Windows обеспечивает взаимодействие с пользователем через так называемое окно консоли. Примером подобных консольных приложений могут являться: окно командной строки (cmd), файловые менеджеры (например, Far Commander), и ряд типовых системных консольных утилит (рис. 1):

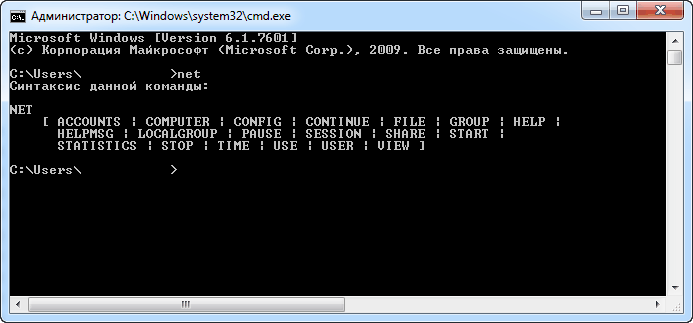


Рис. 1

**Объекты консольного приложения**

Выводить текст на консоль, осуществлять ввод символов, а также совершать любые иные действия с консолью можно лишь ассоциировав с ней некие системные сущности (объекты), посредством которых можно обеспечивать обмен данными. Каждая консоль состоит из следующих основных объектов:

* [единственный] входной буфер - область данных (события/сигналы/данные) для ввода (передачи на консоль);
* [несколько] экранный выходной буфер - область данных (символы/атрибуты) для вывода (отображения на экране);
* Окно консоли - область экрана, отображающая часть выходного буфера;
* Текущая позиция курсора - маркера вывода, обозначающий текущую позицию вывода.

Пример консольного приложения — это командный интерпретатор, который есть в любой операционной системе. В Windows 95/98/ME — это программа command.com. В операционных системах семейства Windows NT (начиная с Windows NT 3.1 и заканчивая Windows 11 / Windows Server 2012) и OS/2 интерпретатором командной строки является программа cmd.exe. В Unix-совместимых операционных системах популярный командный интерпретатор bash.

Оконное приложение — это привычная всем программа Windows. То есть это окошко с разными кнопочками и полями для ввода-вывода данных. На сегодняшний день это, пожалуй, самый распространённый вид программ. Именно оконные приложения создают большинство программистов.

Отличие консольного приложения от оконного заключается в том, что консольное приложение всегда создает консоль.

Создание проекта [консольного](https://prog-cpp.ru/vs-console/) или [оконного](https://prog-cpp.ru/vs-win/) Windows-приложения не отличается от языков программирования Си и C++.

После того, как в Visual Studio появилось окно проекта (в левой части появившегося окна отображается **Обозреватель решений**), для добавления нового файла программы в проект выбираем по правой кнопке мыши на папке **Файлы исходного кода** меню **Добавить->Создать элемент**.

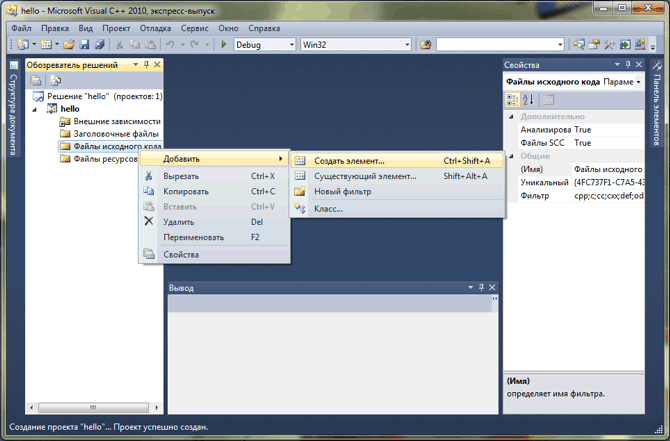


Рис. 1

**2. Работа с консолью в программах на ассемблере**

На практике редко возникает необходимость разработки программы как «вещи в себе». В подавляющем большинстве случаев программа должна взаимодействовать с пользователем, получая от него данные посредством клавиатуры и выдавая результаты своей работы на экран. При знакомстве с новым языком программирования одним из первых вопросов, на которые ищет ответа программист, является выяснение средств этого языка для выполнения операций обмена с консолью (консоль — клавиатура и монитор). Что касается языка ассемблера, то собственных средств обмена с консолью у него нет. Чтобы выполнить подобную операцию, программа использует возможности самого компьютера (прерывания BIOS) и операционной системы, в среде которой эта программа работает. Каждый программист самостоятельно ищет решение проблемы обмена с консолью. Так как эта задача актуальна всегда, то есть необходимость на конкретных примерах показать порядок использования средств BIOS и ОС для обмена с консолью в программах на ассемблере. Примеры не очень сложны, и вы легко сможете быстро встроить их в свои программы.

**Функции BIOS для работы с консолью**

В контексте нашего изложения ROM BIOS (Read Only Memory Basic Input Output System) представляет собой совокупность программ в энергонезависимой памяти компьютера, одной из задач которых является устранение специфики аппаратных компонент компьютера для функционирующего на нем программного обеспечения, включая операционную систему. Обслуживание клавиатуры и монитора выполняют программы BIOS, называемые драйверами. Структурно драйверы состоят из ряда подпрограмм, называемых функциями, каждая из которых выполняет определенные действия. Обращение к функциям BIOS производится аналогично обращению к функциям MS DOS. Для работы с клавиатурой и экраном BIOS содержит два программных прерывания (П — одна из базовых концепций вычислительной техники, которая заключается в том, что при наступлении какого-либо события происходит передача управления специальной процедуре, называемой обработчиком прерываний) — 16h и 10h, обращение к которым, исходя из вышесказанного, является обращением к драйверам этих устройств. Для вызова этих прерываний, как обычно, используется команда INT — int 16h или int 10h. Для выполнения определенной операции в регистре АН указывается номер функции. При необходимости в других регистрах может указываться дополнительная (параметрическая) информация. Ниже рассмотрим подробнее возможности BIOS для работы с консолью.

**Функции BIOS для работы с клавиатурой**

Прерывание 16 BIOS имеет функции для различных типов клавиатур: обычной — 84 клавиши и двух типов расширенной клавиатуры — 101\102 и 122-клавишной. Выяснить функциональные возможности клавиатуры позволяет функция 09h:

Вход: АН - 09h.

Выход: AL = битовое поле, установленные биты которого обозначают поддерживаемые функции: 7 - резерв; 6 — поддержка клавиатуры со 122 клавишами (и функций 20h-22h (int 16h)); 5 — поддержка расширенной клавиатуры со 101-102 клавишами (и функций 10h-12h (int 16h)); 4 — поддержка функции 0Ah (int 16h); 3 — поддержка функции 0306h (int 16h); 2 — поддержка функции 0305h (int 16h); 1 — поддержка функции 0304h (int 16h); 0 — поддержка функции 0300h (int 16h). Прежде чем вызывать эту функцию, необходимо удостовериться в том, что она поддерживается данной версией BIOS. Сделать это можно, вызвав функцию 0c0h прерывания int 15h.

Вход: АН = 00h получить конфигурацию.

Выход: CF = 1 — BIOS не поддерживает эту функцию; CF - 0 — в случае успеха: ES:BX — адрес конфигурационной таблицы в ROM-памяти; АН = состояние (00h — успех; 86h — функция не поддерживается).

**Формат конфигурационной ROM-таблицы:**

|  |  |  |
| --- | --- | --- |
| **Смещение** | **Размер** | **Описание** |
| 00h | 2 байта | Число байтов в этой таблице |
| 02h | 1 байт | Модель BIOS |
| 03h | 1 байт | Подмодель BIOS |
| 04h | 1 байт | Издание BIOS: 0 — 1-я редакция,1 — 2-я редакция и т. д. |
| 05h | 1 байт | 1-й байт свойств |
| 06h | 1 байт | 2-й байт свойств |
| 07h | 1 байт | 3-й байт свойств |
| 08h | 1 байт | 4-й байт свойств |
| 09h | 1 байт | 5-й байт свойств |

# 3. Работа с отладчиком программ TURBO DEBUGGER

Отладчик TURBO DEBUGGER позволяет по шагам проследить процесс выполнения программы на уровне регистров процессора и ячеек памяти. Внешний вид окна отладчика представлен на рисунке 2.

Нижнее меню в отладчике - меню функциональных клавиш.

Значения некоторых функциональных клавиш:

1) **F7** – трассировка программы.

2) **F8** – выполнение программы по шагам т.е. по программе перемещается полоса выбора (синяя), и будет выполнена та команда, на которой эта полоса размещена.

**Примечание.** Трассировка по F7 отличается от пошагового выполнения по F8 тем, что при наличии подпрограмм при трассировке будет по шагам выполняться не только основная программа, но и каждая подпрограмма, которая вызывается из основной программы. А при пошаговом выполнении по F8 по шагам выполняется только основная программа, а каждая подпрограмма выполняется как единый оператор.

После выполнения команды на экране появляется содержимое регистров, флагов и адрес следующей на очереди команды (соответствующие регистры подсвечиваются белым цветом).

3) **F10** - выход в главное, верхнее меню.

Запускаются команды или с помощью мыши или с помощью клавиш перемещения курсора на клавиатуре. Курсором выбирается нужная команда и нажимается клавиша ENTER или нажимается левая кнопка мыши, если выбор выполнялся с помощью мыши. Выбор группы верхнего меню также может выполняться с помощью мыши или с клавиатуры (ALT+ горячая клавиши соответствующей группы).

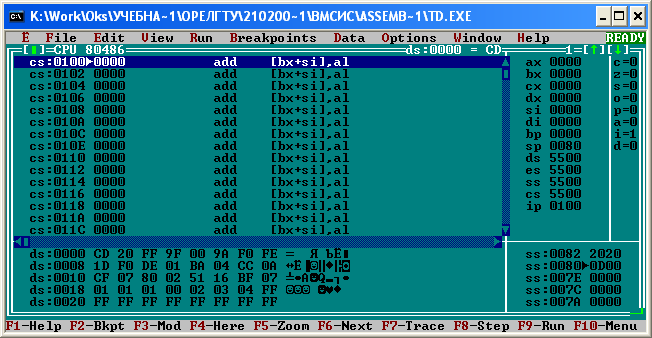


Рисунок 2 – Окно отладчика

В верхнем меню по команде FILE можно открыть любой файл, если он не был указан в команде TD при запуске отладчика.

По команде VIEW появляется еще меню, в котором DUMP - команда получения содержимого памяти по соответствующему адресу, заданному в регистре DS, т.е. содержимое данных определенных в нашей программе. Данные начинаются с нулевого относительного адреса. Эти данные можно изменять.

REGISTERS - после запуска этой команды появляется окно с регистрами, и данные, находящиеся в этих регистрах, можно изменять. Курсором или мышкой выбрать изменяемый регистр и перевести курсор на изменяемое данное, на клавиатуре в появившемся окне набрать новое данное и нажать ENTER.

Выход из отладчика по нажатию ALT+X.

Выход из любой команды по нажатию клавиши ESC.

Закрыть появившееся окно можно или нажать ALT+F3 или надо перевести мышкой курсор в левый угол окна на зеленый квадрат и нажать левую кнопку мыши.

Верхнее и нижнее меню обрамляют отдельные окна, содержащие следующую информацию.

1) О сегменте кода текущей программы. В этом окне отражается смещение команды относительно начала сегмента кода (регистра CS), код команды, мнемоника команды и операнды в шестнадцатеричной системе счисления.

2) О сегменте данных текущей программы. В этом окне отражается смещение данных относительно начала сегмента (регистра DS), их представление в шестнадцатеричном виде и в виде символа таблицы ASCII.

3) О сегменте стека текущей программы. В этом окне отражается смещение вершины стека (регистра SP) относительно начала сегмента стека (регистра SS) и элементы стека в шестнадцатеричном виде.

4) О регистре флагов процессора. В этом окне перечислены все флаги (кроме TF) и в процессе выполнения команд программы отображается их реакция на выполненную команду.

5) О регистрах микропроцессора. В этом окне перечислены регистры процессора и в процессе выполнения команд программы отображается их содержимое после очередной выполненной команды.

Анализируя информацию из этих окон, можно делать вывод о правильности выполнения текущей программы.

Далее рассмотрим, что представляет собой программа на языке ассемблера.

**Контрольные вопросы**

1. Что собой представляет консольное приложение?
2. Назовите объекты консольного приложения
3. В чем заключается отличие консольного приложения от оконного?
4. Какие программные прерывания используют для работы с клавиатурой и экраном BIOS?
5. Опишите окно отладчика TURBO DEBUGGER.