**Turbo Debugger**

**Введение**

Турбо отладчик Turbo Debugger представляет собой набор инструментальных средств, позволяющий отлаживать программы на уровне исходного текста и предназначенный для программистов, использующих семейство компиляторов Borland. В пакет отладчика входят набор выполняемых файлов, утилит, справочных текстовых файлов и примеров программ.

Turbo Debugger позволяет отлаживать программы для Microsoft Windows, Windows 32s, Windows NT и DOS. Многочисленные перекрывающие друг друга окна, а также сочетание спускающихся и раскрывающихся меню обеспечивают быстрый, интерактивный пользовательский интерфейс. Интерактивная, контекстно-зависимая справочная система обеспечит вас подсказкой на всех стадиях работы. Кроме того, Turbo Debugger полный набор средств отладки:

* Вычисление любых выражений языка Си, C++, Pascal и Assemb ler.
* Полное управление выполнением программы, включая программ ную анимацию.
* Доступ на нижнем уровне к регистрам процессора и системной памяти.
* Полные средства проверки данных.
* Развитые возможности задания точек останова и регистрации.
* Трассировка сообщений Windows, включая точки останова по сообщениям.
* Обратное выполнение.
* Поддержка удаленной отладки, в том числе для Windows.
* Полная поддержка объектно-ориентированного программирова ния, включая просмотр классов и проверку объектов.
* Макрокоманды в виде последовательности нажатий клавиш, ус коряющие выполнение команд.
* Копирование и вставка между окнами и диалогами.
* Контекстно-зависимые меню.
* Возможность отладки больших программ.
* Диалоговые окна, позволяющие настроить параметры от ладчика.
* Возможность отладчик 16- и 32-разрядных программ Windows (для 32-разрядной отладки имеется отладчик TD32).
* Обработка исключительных ситуаций операционной системы, а также С и С++.
* Сохранение сеанса.
* Поддержка нитей для мультинитевого программирования Win dows NT.
* Возможность подключения готовых к выполнению в Windows процессов.
* Возможность выбора для элементов, выводимых в Turbo Debug ger, национального порядка сортировки.

Для работы Turbo Debugger требуются те же аппаратные средства, что и для компилятора языка Borland. Кроме того, Turbo Debugger поддерживает графические адаптеры CGA, EGA, VGA, Hercules (монохромный графический режим), Super VGA, TIGA и 8414.

**Замечания по терминологии**

Для удобства и краткости изложения несколько терминов в данном руководстве используется в более широком смысле, чем обычно. К ним относятся термины "модуль", "функция" и "аргумент".

Термин "модуль" в данном руководстве обозначает эквивалент модуля (module) Си и ассемблера, а также модуля (unit) Паскаля.

Термин "функция" в данном руководстве означает функцию в языке Си и то, что в языке Паскаль называется подпрограммой, то есть охватывает процедуры, функции и объекто-ориентированные правила. В языке Си функция может возвращать значение (так же, как и функция в Паскале), но может и не возвращать (как процедура в Паскале). (В Си функция, которая не возвращает значения, называется пустой (void) функцией). Для краткости изложения термин "функция" часто используется для обозначения как функций языка Си, так и функций и процедур языка Паскаль, конечно, кроме тех разделов руководства, которые посвящены конкретным языкам.

Термин "аргумент" используется в данном руководстве как синоним термина "параметр". Он означает как аргументы (параметры) командной строки, так и аргументы (параметры), передаваемые в процедуры и функции.

**Установка и настройка Turbo Debugger**

В данном разделе описывается, как инсталлировать Turbo Debugger и задать используемые по умолчанию параметры у режимы вывода. Здесь рассказывается также о некоторых файлах, входящих в состав данного продукта.

**Установка Turbo Debugger**

Программа INSTALL, поставляемая с компилятором Borland, полностью устанавливает пакет Turbo Debugger, включая выполняемые файлы, файлы конфигурации, утилиты, справочные текстовые файлы и примеры программ. Эта установочная программа создает пиктограммы для компилятора Borland и инструментальных средств языка, помещая их в новую программную группу Windows. Полный перечень файлов, инсталлируемых программой INSTALL.EXE, содержится в файле FILELIST.DOC (этот файл копируется программой инсталляции в основной каталог компилятора).

**Файлы, входящие в состав пакета Turbo Debugger**

|  |  |
| --- | --- |
| **Имя файла** | **Описание** |
| DUAL8514.DLL | Видео-DLL, которые поддерживают отладку с двумя мониторами для мониторов 8514. |
| STB.DLL | Видео-DLL, поддерживающая видео-адаптеры STB. |
| TD.EXE | Отладчик для отладки приложений DOS. |
| TDDEBUG.386 | Этот драйвер используется TDW.EXE для доступа к специальным отладочным регистрами процессора 80386 (или старше). |
| TDHELP.TDH | Справочных файл для TD.EXE. |
| TDKBDW16.DLL | Файл поддержки для Windows 32s. |
| TDKBDW32.DLL | Файл поддержки для Windows 32s. |
| TDREMOTE.EXE | Драйвер, используемый в удаленной системе для поддержки удаленной отладки в DOS. |
| TD32.ICO | Пиктограмма, используемая для TD32.EXE. |
| TD32HELP.TDH | Справочный файл для TD32.EXE. |
| TDVIDW16.DLL | Файл поддержки для Windows 32s. |
| TDVIDW32.DLL | Файл поддержки для Windows 32s. |
| TDW.EXE | Выполняемая программа для отладки 16-разрядных программ Windows. |
| TDW.INI | Файл инициализации, используемый TDW.EXE и |
| TD32.EXE. | Он создается программой инсталляции и помещается в основной каталог Windows. |
| TDWGUI.DLL | Видео-DLL, используемая для вывода окна отладчика TDW в Windows 3.х или TD32 в Windows 32s. |
| TDWHELP.TDH | Справочный файл для TDW.EXE. |
| TDWINTH.DLL | Поддерживающая DLL для TDW.EXE. |
| WREMOTE.EXE | Драйвер для удаленной отладки в Windows. |
| TDINST.EXE | Создает и модифицирует файл конфигурации TDCONFIG.TD. |
| TDMEM.EXE | Выводит на экран доступную память компьютера, включая дополнительную и расширенную. |
| TDRF.EXE | Утилита передачи файлов, используемая для передачи файлов в удаленную систему. |
| TD32INST.EXE | Создает и модифицирует файл конфигурации TD32 - TDCONFIG.TD32. |
| TDSTRIP.EXE | Удаляет из файлов .EXE и .DLL используемую отладчиком отладочную информацию (таблицу идентификаторов) без перекомпоновки файлов. |
| TDSTRP32.EXE | Удаляет из файлов .EXE и .DLL используемую отладчиком отладочную информацию (таблицу идентификаторов) без перекомпоновки файлов. |
| TDUMP.EXE | Выводит на экран структуру 16- или 32-раз- рядных файлов .EXE, .DLL и .OBJ, а также содержимое отладочную информацию об идентификаторах. |
| TDWINI.EXE | Позволяет изменить и настроить параметры видеодрайвера отладчика. |
| TDWINI.HLP | Справочный файл для TDWINI.EXE. |
| TDWINST.EXE | Создает и модифицирует файл конфигурации |
| TDCONFIG.EXE. | Настраивает для TDW параметры вывода и цвета экрана. |
| WRSETUP.EXE | Файл конфигурации для утилиты WREMOTE - драйвера удаленной отладки. |
| TD\_ASM.TXT | Файл с информацией по отладке программ на Turbo Assembler, которая полезна также при отладке программ со встроенным ассебмлером. |
| TD\_HELP!.TXT | Файл с ответами на общие вопросы. Среди прочего в нем обсуждаются синтаксические отличия между анализом выражений в Turbo Debugger и компиляторах, отладка программ на нескольких языках др. |
| TD\_HDWMP.TXT | Файл с информации о настройке конфигурации отладчика для использования аппаратных отладочных регистров. |
| TD\_RDME.TXT | Содержит последнюю информацию, отсутствую- щую в руководстве. |
| TD\_UTILS.TXT | Описывает утилиты отладчика, работающие в режиме командной строки: TDSTRIP, TDUMP, TDWINST, TD32INST, TDSTRP32, TDMEM, TDMAP и TDUMP32. |
| MAKEFILE | Формирующий файл, используемый в примерах программ. |
| TDWDEMO.BUG | Исходный код примера программы с ошибками (для отладки). |
| TDWDEMO.H | Файл заголовка, используемых примером прог- раммы. |
| TDWDEMO.ICO | Пиктограмма примера программы. |
| TDWDEMO.IDE | Файл проекта для примера программы. |
| TDWDEMO.RC | Файл ресурса для примера программы. |
| S\_PAINT.C | Исходный код примера программы. |
| S\_PAINT.EXE | Пример программы. |

**Настройка Turbo Debugger**

Вы можете конфигурировать параметры вывода Turbo Debugger и программные установки с помощью файлов конфигурации и меню Option отладчика. Параметры, установленные в файлах конфигурации, начинают действовать после загрузке Turbo Debugger. Чтобы изменить параметры после загрузки, используйте команды в меню Options.

**Файлы конфигурации**

При запуске Turbo Debugger использует следующие файлы конфигурации, инициализации и сеанса:

- TDCONFIG.TD

- TFCONFIG.TDW

- TDCONFIG.TD2

- TDW.INI

- XXXX.TR

- XXXXTRW

- XXXX.TR2

Файлы конфигурации TDCONFIG.TD, TDCONFIG.TDW и TDCONFIG.TD2 создаются с помощью отладчиков TD, TDW и TD32 соответственно. Параметры, установленные в этих файлах, переопределяют параметры, используемые данными отладчиками по умолчанию. Вы можете модифицировать файлы конфигурации с помощью инсталляционных программ TDINST.EXE. TDWINST.EXE и TD32.EXE.

TDW.INI - это файл инициализации, используемый TDW.EXE и TD2.EXE. Он содержит установки для используемого отладчиком видеодрайвера, расположение файла TDWINTH.DLL (динамически компонуемой библиотеки, применяемой для отладчик в Windows), и параметры удаленной отладчик для WRSETUP.EXE.

Программа установки отладчика помещает TDW.INI в основной каталог Windows. В этой копии TDW.INI параметр видеодрайвера ([VideoDLL]) устанавливается в SVGA.DLL, а установка DebuggerDLL указывает маршрут к TDWINTH.DLL. Полное описание TDW.INI можно найти в файле TD\_HELP!.TXT. Файлы с расширениями .TR, .TRW и .TR2 содержат параметры состояния сеанса отладчиков.

Когда вы запускаете Turbo Debugger, он ищет файлы конфигурации в следующей последовательности:

* в текущем каталоге;
* в каталоге, заданном в установке Turbo Directory программы установки Turbo Debugger;
* в каталоге, содержащем выполняемый файл отладчика.

Если Turbo Debugger находит файл конфигурации, то параметры, заданные в этом файле, переопределяют встроенные по умолчанию установки. Если при запуске Turbo Debugger вы указываете параметры командной строки, то они переопределяют соответствующие значения по умолчанию и значения, заданные в файле конфигурации.

Для поддержки доступных типов видеоадаптеров и мониторов TDW и TD32 используют различные типы видео-DLL. При инсталляции Turbo Debugger запустите программу установки TDWINI.EXE, которая поможет выбрать или модифицировать используемые отладчиками видео-DLL. По умолчанию TDW и TD32 используют драйвер SVGA.DLL, который поддерживает большинство видеоадаптеров и мониторов. Подробнее об этом рассказывается в оперативном справочнике Help программы TDWINI.EXE.

Кроме того, Turbo Debugger поддерживает в TD, TDW и в TD32 отладку с двумя мониторами. Для этого потребуется цветной монитор и видеоадаптер и монохромный монитор и видеоадаптер. При отладке с двумя мониторами Turbo Debugger выводится на монохромном мониторе, а отлаживаемая программа - на цветном. Это позволяет видеть во время отладки вывод программы. Для загрузки TD или TDW в режиме с двумя мониторами используйте параметр командной строки -do. При работе с TD32 в Windows 32s нужно использовать видеобиблиотеку SVGA.DLL. В файл TDW.INI в раздел [VideoOptions] для этого нужно включить mono=yes. Для установки параметров видеоадаптера используйте утилиту TDWINI.EXE.

**Меню Options**

|  |  |  |
| --- | --- | --- |
| Язык | Language... | Source |
| Макрокоманды | Macros | > |
| Параметры дисплея | Display options... |  |
| Маршрут доступа к исходному файлу | Path for source... |  |
| Параметры сохранения | Save options... |  |
| Параметры восстановления | Restore options... |  |

Это меню содержит команды, с помощью которых вы можете управлять выводом отладчика. С помощью команды Options Language вы можете выбрать язык, используемый в Turbo Debugger для вычисления выражений. Команда Options Dispay Options открывает диалоговое окно параметров вывода Display Options. Параметр этого окна можно использовать для управления выводом Turbo Debugger. Отладчик TD32 имеет несколько дополнительных параметров для поддержки многозадачной операционной системой Windows NT.

[\*] Display options

Display swapping Integer format

( ) None ( ) Hex

(.) Smart ( ) Decimal

( ) Always (.) Both

Screen lines Tab size

(.) 25 ( ) 43/50 8

Background delay User screen delay

10 3

OK Cancel Help

Кнопки с зависимой фиксацией Display Swapping (Переключение дисплея) позволяет выбрать один из трех способов управления переключением между экраном Turbo Debugger и экраном вашей программы, а именно:

|  |  |
| --- | --- |
| None (отсутствует) | Нет переключения между экранами. Используйте данный параметр, если вы отлаживаете программу, которая не выводит никакой информации на экран. |
| Smart (эффективное) | Переключение на экран пользователя выполняется только тогда, когда может произойти вывод на экран. Отладчик будет выполнять переключение экранов всякий раз когда вы проходите программу или выполняете инструкцию исходного кода, в которых осуществляется чтение или запись в видеопамять. Этот параметр используется по умолчанию. |
| Always (постоянное) | Переключение в экран пользователя происходит при каждом выполнении программы пользователя. Используйте этот параметр, если параметр Smart не позволяет перехватить все случаи вывода информации на экран вашей программой. Если вы выберете этот параметр, экран будет "мерцать" при каждом шаге выполнения вашей программы, так как на короткое время экран Турбо отладчика будет заменяться на экран вашей программы. |

**Переключатель Integer Format**

Кнопки с зависимой фиксацией Integer Format (Формат целых чисел) позволяет определить один из трех форматов, управляющих выводом целых чисел:

|  |  |
| --- | --- |
| Decimal (Десятичный) | Целые числа выводятся, как обычные десятичные значения. |
| Hex (Шестнадцатиричный) | Целые числа выводятся в шестнадцатиричном виде в формате, принятом в соответствующем языке. |
| Both (Оба) | Целые числа выводятся и как десятичные, и как шестнадца тиричные значения (которые указываются в скобках после шестнадцатиричных значений). |

Переключатель Screen Lines (Размер экрана) можно использовать для того, чтобы определить, использует ли Turbo Debugger обычный 25-строчный режим экрана или 40- или 50-строчный режим, доступный при работе с адаптерами EGA и VGA.

Поле Tab Size (Размер табуляции) позволяет определить позиции при каждой табуляции. Вы можете уменьшить число позиций табуляции, чтобы можно было видеть больше исходного текста в файлах, выравнивание кода выполнено с помощью табуляции. Размер позиции табуляции можно установить в значения от 1 до 32.

Поле Background Delay, которое выводится только в TD32.EXE, позволяет задать, как часто обновляются экраны отладчика. При использовании этого параметра в сочетании с командой Run Wait for Child вы можете наблюдать действия программы в экранах Turbo Debugger при ее выполнении.

Поле User Screen Delay, которое выводится только в TD32, позволяет задать время вывода экрана программы при нажатии Alt+F5 (команда Windows User Screen). Это полезно использовать при работе с TD32 в режиме полного экрана, когда нужно видеть окна приложения. Определив задержку, вы можете задать, как должно будет выводиться экран программы, прежде чем управление вернется к Turbo Debugger.

Команда Path for Source (Маршрут доступа к исходному файлу) задает каталоги, в которых Turbo Debugger будет искать исходные файлы. Чтобы задать несколько каталогов, разделите их точкой с запятой. Хотя поле ввода Enter Source Directory Path может содержать максимум 256 символов, для задания более длинных маршрутов вы можете определить файл ответов, содержащий одну строку с определением каталогов. Чтобы задать такой файл в данном поле ввода, введите символ @, затем задайте имя файла.

Команда Save Options (Сохранить параметры) открывает диалоговое окно, с помощью которого вы можете сохранить текущие параметры на диске в файле конфигурации. В этом файле сохраняются:

* ваши макрокоманды (кнопка Options);
* текущая схема окон и форматы областей окон (Layout);
* все значения параметров, заданные в меню Options (кнопка Options).

[\*] Save configuration

[X] Options OK

[ ] Layout Cancel

[ ] Macros

Save to Help

tdconfig.tdw

Поле ввода Save To позволяет также задать имя файла конфигурации. По умолчанию TDW.EXE использует TDCONFIG.TDW, а TD32.EXE TDCONFIG.TD2.

Команда Restore Options позволяет восстановить параметры из файла на диске. Вы можете создать несколько файлов конфигурации, записав в них различные макрокоманды, схемы окон и т.д. Требуется задавать файл параметров, созданный с помощью команды Options Save Options или утилиты установки отладчика.

**Выполнение программ с отладчиком**

**Подготовка программ для отладки**

Когда вы выполняете компиляцию и компоновку с помощью одного из языков фирмы Borland, следует указать компилятору, что нужно генерировать полную информацию для отладки. Если вы скомпилируете объектные модули своей программы без информации для отладки, придется перекомпилировать все эти модули, чтобы можно было полностью использовать все средства отладки на уровне исходного кода. Можно также сгенерировать информацию для отладки только для отдельных модулей (это позволит сократить объем программы), но потом будет крайне неприятно попасть в модуль, где информация для отладки недоступна. Поэтому мы рекомендуем перекомпилировать все модули, если, конечно, это позволяет имеющаяся память. В случае нехватки памяти или уверенности в правильной работе отдельных модулей можно перекомпилировать только конкретные модули. При компиляции программ для отладки лучше исключить оптимизацию, иначе вы запутаетесь при отладке отдельных частей кода, оптимизированных компилятором.

При компиляции из интегрированной среды для включения в файлы .OBJ отладочной информации выберите команду Options Project (для вывода блокноте Style Sheet), в блоке списка Topic выберите Compiler Debugging и включите в OBJs кнопку с независимой фиксацией Debug. Чтобы включить отладочную информацию в выполняемые файлы, выберите команду Options Project, затем команду Linker General в блоке списка Topic. Выводятся кнопки с независимой фиксацией General. Включите кнопку Debug Information.

При компиляции программ с использованием компилятора режима командной строки используйте для включения отладочной информации директиву компилятора -v.

После полной отладки программы вы можете скомпилировать и скомпоновать ее заново с оптимизацией и исключением отладочной информации.

**Отладка программ ObjectWindows**

Если вы применяете TDW для отладки программ, использующих ObjectWindows 1.0х, то нужно конфигурировать отладчик, чтобы он распознавал систему диспетчеризации сообщений Objecwindows DDVT. Для этого запустите TDWINST, для вывода диалогового окна Source Debugging выберите команду Options Source Debugging, включите кнопку с независимой фиксацией OWL 1.0X Window Messages, затем сохраните конфигурацию и выйдите из TDWINST.

**Запуск отладчика**

После компиляции и компоновки программ с включением отладочной информации вы можете начать процесс отладки, запустив Turbo Debugger и загрузив с ним программу. При этом вы можете использовать один из трех отладчиков: TD.EXE для отладки 16-разрядных приложений DOS, TDW.EXE для отладки 16-разрядных приложений Windows и TD32.EXE для отладки 32-разрядных приложений Windows.

Отладчики для Windows запускаются в Windows из группу компиляторов Borland в Program Manager выбором пиктограмм TDW или TD32, из интегрированной среды компиляторов выбором команды Tool Turbo Debugger (программы будут отлаживаться в активном окне Edit), из диалогового окна Program Manager File Run (в поле ввода Command наберите TDW или TD32 и параметры) или из File Manager двойным щелчком "мышью" на пиктограмме выполняемого файла TDW.EXE или TD32.EXE из каталога, содержащего Turbo Debugger.

При запуске Turbo Debugger из командной строки можно задать параметры запуска и режимы отладки. Эта командная строка имеет следующий синтаксис:

TD TDW TD32 [параметры] [имя\_программы [аргументы]]

Элементы в квадратных скобках не обязательны. При запуске отладчика задавайте корректный маршрут программы и ее аргументы. Параметры Turbo Debugger перечислены в следующей таблице:

|  |  |
| --- | --- |
| **Параметр** | **Функция** |
| -ar# | Подключает к процессу с идентификационным номером # и продолжает выполнение. |
| -as# | Подключает к процессу с идентификационным номером и передает управление Turbo Debugger. |
| -cимя\_файла | Файл конфигурации, активизирующийся при загрузке. |
| -do | Выводит TD.EXE или TDW.EXE на втором дисплее. |
| -dp | Переключение страниц для TD.EXE. |
| -ds | Переключение на содержимое экрана пользователя. |
| -h | Вывод справочного экрана. |
| -? | Вывод справочного экрана. |
| -ji | Игнорирование старой информации сохранения. |
| -jn | Не использовать информацию сохраненного состоя- ния. |
| -ip | Вывод подсказки, если информация сохраненного состояния старая. |
| -ju | Использовать информацию сохраненного состояния, даже если она старая. |
| -k | Разрешает запись нажатий клавиш. |
| -l | Запуск кода инициализации ассемблера. |
| -p | Разрешает работать с "мышью". |
| -r | Отладка на удаленных системах (с параметрами по умолчанию. |
| -rnлок;удал | Разрешает сетевую отладку. |
| -rp# | Задает порт для удаленной отладки. |
| -rs# | Скорость связи: 1 - медленная, 2 - средняя, 3 - быстрая. |
| -sc | Отмена проверки букв на верхний/нижний регистр. |
| -sdкат;[кат] | Каталог исходного файла. |
| -tкаталог | Задает каталог для поиска информации о конфигурации и выполняемых файлов. |
| -vg | Полное сохранение графики (только для TD.EXE). |
| -vn | Запрет режима 43/50 строк для TD.EXE. |
| -vp | Разрешение сохранения палитры EGA/VGA для TD.EXE. |
| -wc | Разрешает/запрещает сообщение о возможном крахе системы. |
| -wd | Разрешает проверку на наличие всех DLL вашей программы (по умолчанию разрешена). |

Если вы запускаете программу, используя пиктограммы TDW или TD32, то можете задать параметры с помощью диалогового окна Pro perties пиктограммы. При этом параметры сохраняются вместе с ус тановленными значениями характеристик пиктограммы. В окне Proper ties вы можете также задать свою программу и ее аргументы. После этого она будет загружаться при двойном щелчке "мышью" на пиктог рамме отладчика. Чтобы задать для пиктограммы значения Property, щелкните на ней "мышью", затем выберите в Program Manager команду File Properties. В поле ввода Command Line наберите имя отладчика с параметрами командной строки. После этого щелкните "мышью" на OK.

Для запуска Turbo Debugger из интегрированной среды Borland С++ for Windows, то для задания параметров командной строки може те сделать следующее:

* Для вывода диалогового окна Tools выберите команду Opti ons Tools интегрированной среды.
* В списке окна Tools выберите TDStartup.
* Чтобы открыть диалоговое окно Tools Options, щелкните "мышью" на командной кнопке Edit.
* В поле Commands Line после макрокоманды $TD введите пара метры командной строки отладчика.

Макрокоманда $ARG в поле Command Line позволяет задать аргу менты, передаваемые программе. Чтобы задать аргументы, выберите для открытия диалогового окна Enviroment Options команду Opti ons Enviroment. Затем выберите в блоке списка Topics Debugger и введите в блоке списка Run Arguments аргументы программы.

**Выполнение отладчика**

При выполнении TDW (или TD32 в Windows 32s) отладчик откры вает полноэкранное текстовое окно. Однако, в отличие от других приложений, вы не можете использовать в Turbo Debugger клавиши Windows Alt+Esc или Ctrl+Esc, то есть смена задач здесь запреще на. Однако в Windows NT TD32 активизирует окно с командной подс казкой, и доступны все обычные средства приложения Windows.

**Загрузка программы в отладчик**

Программу в Turbo Debugger вы можете загрузить из командной строки или после запуска отладчика. Чтобы загрузить в отладчик новую программу (или сменить загруженную), используйте команду File Open. Эта команда открывает набор диалоговых окон, первое из которых называется Load a Program to Debug. В TD и TDW это окно содержит дополнительную командную кнопку Session, которая исполь зуется для поддержки средств удаленной отладки.

В поле ввода Program Name задайте имя выполняемого файла программы и нажмите Enter. Чтобы выполнить поиск программы по ка талогам, щелкните "мышью" на кнопке Browse. Откроется второе диа логовое окно - Enter Program Name to Load. В блоке Files этого окна выводятся файлы в текущем выбранном каталоге. Введя в блоке File Name маску файлов (например, \*.EXE), вы можете задать список нужных файлов.

Для перемещения по каталогам вы можете использовать двойной щелчок "мышью" на записях окна Directories. После выбора каталога выберите загружаемый файл в блоке Files. Для быстрого поиска фай ла наберите в блоке Files его имя.

После задания программы вы можете определить, требуется ли выполнять в отладчике ее код запуска. Если вы выберите кнопку с независимой фиксацией Execute Startup Code, Turbo Debugger выпол няет программный код до процедуры main программы (или ее эквива лента). В противном случае при загрузке программы никакой код вы полняться не будет.

Для поддержки удаленной отладки TDW содержит дополнительный набор переключателей. Если вы выберите в группе Session окна Load a New Program to Debug кнопку с зависимой фиксацией Remote, это позволяет задать отладку на удаленной системе. Кнопка Local опре деляет локальную отладку.

При загрузке программы с включенной в нее отладочной инфор мацией Turbo Debugger открывает окно CPU, в котором показывает дизассемблированные инструкции ассемблера. При выполнении прог раммы под управлением отладчика должны быть доступны все ее ис ходные файлы. Кроме того, в том же каталоге должны находиться все файлы .EXE и .DLL приложения. Исходный код программы отладчик ищет в следующем порядке:

* в том каталоге, где компилятор нашел исходные файлы;
* в каталоге, заданном в команде Options Path for Source (или в параметре командной строки -sd);
* в текущем каталоге;
* в том каталоге, где находятся файлы .EXE и .DLL.

После загрузки программы в отладчик вы можете с помощью ко манды Run Arguments задать или изменить аргументы программы. Их можно также задать после имени программы в командной строке.

При выходе из Turbo Debugger он сохраняет состояние текущего сеанса в файле сеанса. При перезагрузке программы из этого ката лога отладчик восстанавливает параметры последнего сеанса. По умолчанию в файле сеанса сохраняются все списки протоколов, вы ражения просмотра, элементы буфера, установки исключительных си туаций операционной системы, установки выражений Си и С++. Эти файлы называются XXXX.TR (отладчик TD), XXXX.TRW (TDW) и XXXX.TR2 (TD32), где XXXX - имя отлаживаемой программы. Если при выходе из отладчика программа не загружена, то XXXX - это имя отладчика.

Команда Options Set Restart открывает диалоговое окно пара метров рестарта Restart Options, где вы можете настроить обработ ку в Turbo Debugger файлов сеанса. Кнопка с независимой фиксацией Restore at Restart определяет, какие параметры отладчика вы хоти те сохранять в файле состояния сеанса, а кнопка а зависимой фик сацией Use Restart задает, когда следует загружать файл сеанса:

|  |  |
| --- | --- |
| Always | Файл состояния сеанса используется всегда. |
| Ignore if old | Если программа перекомпилирована, файл состояния сеанса не используется. |
| Prompt if old | Turbo Debugger запрашивает, хотите ли вы использовать файл состояния сеанса после изменения программы. |
| Never | Не использовать файл состояния сеанса. |

**Управление выполнением программы**

В процессе отладки управление периодически передается между вашей программой и отладчиком. Когда управление передается Turbo Debugger, он может использовать свои средства для поиска по исходному коду и структурам данных программы и выявления причины неправильного выполнения программы. Для этого можно использовать меню и окна отладчика. Отладчик предоставляет много способов управления выполнением программы. Вы можете:

* выполнять программу по шагам (по одной машинной инструкции или строке исходного кода);
* выполнять как один шаг вызовы функций;
* выполнять программу до заданного места;
* выполнять программу до возврата из текущей функции;
* трассировать программу;
* выполнять программу в обратном направлении;
* выполнять программу до точки останова;
* выполнять программу до появления определенного сообщения Windows;
* приостанавливать программу при возникновении исключитель ной ситуации С++ или Си.

Кроме точек останова, сообщений Windows и исключительных си туаций С++ все механизмы управления выполнением находятся в меню Run.

**Меню Run**

Меню Run (Выполнение) содержит несколько параметров для выполнения различных частей вашей программы. Поскольку эти параметры часто используются, им соответствуют функциональные клавиши.

|  |  |  |
| --- | --- | --- |
| Run | F9 | Выполнение |
| Go to cursor | F4 | Выполнение до курсора |
| Trace into | F7 | Трассировка |
| Step over | F8 | Шаг с пропуском |
| Execute to... | Alt-F9 | Выполнение до... |
| Until return | Alt-F8 | Выполнение до возврата |
| Animate... |  | Автоматизировать |
| Back trace | Alt-F4 | Обратная трассировка |
| Instruction trace | Alt-F7 | Трассировка инструкций |
| Arguments... |  | Аргументы |
| Program reset | Ctrl-F2 | Сброс программы |
| Next Pending Status |  | Следующий ждущий |
| Wait for Child |  | Ожидание дочернего |

Команда Run запускает вашу программу на выполнение. При наступлении одного из следующих событий управление передается отладчику.

* ваша программа завершила выполнение;
* обнаружена точка останова с действием прерывания;
* прервали выполнение с помощью клавиши прерывания;
* олнение программы остановлено из-за ошибки;
* никли отмеченные исключительные ситуации Си или С++.

Команда Go to Cursor выполняет программу до той строки, где находится курсор (в текущем окне Module или области Code окна CPU). Если текущим окном является окно Module, курсор должен находиться на строке исходного кода внутри функции.

Команда Trace Into выполняет одну строку исходного кода или машинную инструкцию. Если текущая строка содержит вызов процедуры или функции, то отладчик выполняет трассировку этой процедуры. Однако, если текущим окном является окно CPU, то выполняется одна машинная инструкция. Если текущим является окно Module, то выполняется строка исходного кода.

Turbo Debugger интерпретирует методы объектов и функции-элементы классов, как все другие процедуры и функции. Клавиша F7 позволяет трассировать их исходный код (если он доступен).

Если вы выполняете эту команду для одной машинной инструкции, отладчик интерпретирует некоторые инструкции, как одну инструкцию, даже они приводят к выполнению нескольких инструкций. Это инструкции CALL, INT, LOOP, LOOPZ и LOOPNZ.

Это справедливо и для префиксов REP, REPNZ или REPZ, за которыми следуют инструкции CMPS, CMPSW, LODSB, MOVS, MOVSB, MOVSW, SCAS, SCASB, SCASW, STOS, STOSB или STOSW.

Команда Step Over выполняет одну строку исходного кода или машинную инструкцию, минуя трассировку вызываемой процедуры или функции. При этом обычно выполняется одна строка исходного текста программы. Если вы используете Step Over при расположении указателя на инструкции вызова, то Turbo Debugger полностью отрабатывает эту функции и переводит вас к оператору после вызова функции.

Если вы выполняете эту команду для одной исходной строки, отладчик интерпретирует любой вызов процедуры или функции на этой строке, как часть самой строки, поэтому при завершении вы не окажетесь в начале одной из этих функций. Вместо этого вы перейдете к следующей строке текущей подпрограммы или к предыдущей программе, которая вызвала данную.

Команда Run Step Over интерпретирует вызов метода объекта или функцию-элемент класса как один оператор, и выполняет для него такие же действия, как при любом другом вызове процедуры или функции.

Команда Execute To выполняет вашу программу до адреса, который вы ввели в ответ на подсказку в диалоговом окне Enter Code Address to Execute To. Программа может не достичь этого адреса, если встречается точка останова или вы прерываете выполнение.

Команда Until Return выполняет текущую процедуру или функцию, пока она не возвратит управление вызывающей программе. Это полезно использовать при двух обстоятельствах: если вы случайно вошли в процедуру или функцию, выполнение которой вас не интересует (с помощью команды Run Trace вместо команды Run Step), или когда вы определили, что текущая функция работает правильно, и не хотите медленно проходить по шагам ее оставшуюся часть.

Команда Animate выполняет непрерывную последовательность команд Trace. Это позволяет наблюдать за текущим адресом в исходном коде и видеть изменение значений переменных. Прервать выполнение данной команды можно нажатием любой клавиши. После выбора команды Run Animate выведется подсказка для ввода значения интервала временной задержки между последовательными трассировками (в десятых долях секунды). По умолчанию используется значение 3.

Если вы выполняете трассировку программы (с помощью оперативных клавиш F7 или Alt-F7), то команда Back Trace изменяет порядок выполнения на обратный. Это средство удобно использовать, если вы проскочили место предполагаемой ошибки и хотите вернуться к этой точке. Данная команда позволяет выполнить программу "в обратном порядке" по шагам или до заданной (подсвеченной) точки в области инструкций окна Execution History. В окне CPU обратное выполнение доступно всегда, а для исходного кода в окне Full History параметр Execution History нужно установить в On.

Команда Instruction Trace выполняет одну инструкцию. Ее можно использовать, когда вы хотите трассировать прерывание, или когда вы находитесь в окне Module и хотите выполнять трассировку процедуры или функции, которая находится в модуле без отладочной информации (например, библиотечной подпрограмме). Так как вы больше не будете находиться в начале строки исходного теста, эта команда обычно переводит вас в окно CPU.

Команда Arguments позволяет задать новые аргументы программы. Введите аргументы программы, как они задаются после имени программы в командной строке. После этого отладчик запрашивает, хотите ли вы перезагрузить отладчик с диска. Следует ответить "Yes".

Команда Program Reset перезагружает отлаживаемую программу с диска. Ее можно использовать в следующих случаях:

* да выполнение "зашло слишком далеко", то есть пройдено то место, где имеется ошибка;
* да ваша программа завершила работу и вы хотите запустить ее снова;
* если вы работаете в окне CPU, приостановили выполнение программы с помощью прерывания и хотите завершить ее и начать сначала (убедитесь однако, что вы не прервали выполнения программы в коде ядра Windows);
* если вы хотите перезагрузить DLL, которая уже загружена, установите для нужной DLL в Yes параметр Startup Option в диалоговом окне Load Module Source или DLL Symbols и сбросьте программу.

Если вы выбрали команду Program Reset и находитесь в окне Module или CPU, то отладчик устанавливает Instruction Pointer на начало программы, но экран остается там, где вы были при выборе команды Program Reset. Такое поведение облегчает установку курсора на то место, где вы были, и выполнение программы до данной строки. Если вы выбрали команду Program Reset только потому, что зашли на один оператор дальше нужного места, вы можете переместить курсор в файле исходного кода вверх на несколько строк и нажать клавишу F4, чтобы выполнить программу до этого места.

Команда Next Pending Status (доступная при отладке в Windows NT) может использоваться при установке в Yes команды Run Wait for Child. Если Wait for Child установлена в No (и ваша программа при обращении к отладчику работает в фоновом режиме), то команду Next Pending Status можно использовать для получения сообщения о статусе программы. Чтобы указать на наличие такого сообщения, индикатор активности отладчика выводит PENDING.

Команду Wait for Child (которая используется исключительно отладчиком TD32 для отладки программ Windows NT) можно переключать в Yes и No. В состоянии No вы можете обращаться к отладчику во время выполнения программы, не дожидаясь, пока она дойдет до точки останова. Эта команда полезна при отладке интерактивных программ (она позволяет, например, перейти в отладчик при ожидании программой ввода с клавиатуры).

**Прерывание выполнения программы**

При выполнении программы вы можете получить доступ к отладчику, нажав клавишу прерывания программы. Используемые клавиши зависят от типа отлаживаемого приложения:

* при отладке программ Windows 3.х используйте клавиши Ctrl+Alt+SysRq;
* при отладке программ Windows 32s используйте клавиши Ctrl+Alt+F11;
* при отладке программ Windows NT используйте клавишу F12;
* при отладке программ DOS используйте клавиши Ctrl+Break.

Это полезно использовать, когда в программе не установлены точки останова.

Если при возврате в Turbo Debugger вы увидите окно CPU без соответствующих программе инструкций, то возможно вы находитесь в коде ядра Windows. При этом следует установить точку останова в том месте, где должна выполняться ваша программа. Затем выполните программу до точки останова (F9). После этого можно возобновить отладку. Находясь в ядре Windows, не следует пытаться выполнять программу по шагам или пытаться перезагрузить приложение. Это может привести к краху системы.

**Обратное выполнение**

Каждую выполненную инструкцию Turbo Debugger регистрирует в протоколе выполнения (при трассировки программы). С помощью окна протокола выполнения Execution History вы можете просмотреть выполненные инструкции и вернуться в нужную точку программы. Команда обратного выполнения Reverse Execute выполняется по клавишам Alt+F4. Turbo Debugger может регистрировать около 400 инструкций. Здесь действуют следующие правила:

* Регистрируются только те инструкции, которые выполнены с помощью команды Trace Into (F7) или Instruction Trace (Alt+F7). Однако, если не выполняются отдельные инструкции (перечисленные ниже), то регистрируются также команды Step Over.
* Инструкция INT приводит к стиранию протокола выполнения. Если вы не трассируете прерывание с помощью Alt+F7, то об ратное выполнение этой инструкции невозможно.
* После выполнения команды Run или выполнения после прерыва ния протокол удаляется. (Регистрация начинается после во зобновления трассировки.)
* При выполнении вызова функции без ее трассировки обратное выполнение за инструкцию после возврата невозможно.
* Обратное выполнение инструкций работы с портами невозможно (отменить чтение и запись нельзя).
* Невозможно также обратное выполнение вызываемого програм мой кода Windows (если только вы не находитесь в окне CPU и не отлаживаете DLL).

В окне CPU обратное выполнение доступно всегда, а для обратного выполнения исходного кода нужно установить Full History в On (в меню Execution History). Меню Execution History содержит также команды Inspect и Reverse Execute. Команда Inspect переводит вас к команде, подсвеченной в области Instruction. Если это строка исходного кода, она выводится в окне Module. При отсутствии исходного кода открывается окно CPU и подсвечивается инструкция в области Code. Действие инструкций IN, INSB, INSW, OUT, OUTSB, OUTSW отменить невозможно, поэтому их обратное выполнение может давать побочные эффекты.

TD.EXE имеет в окне Execution History дополнительную область, позволяющую вернуться в нужную точку программы при случайной потере протокола. Область Keystroke Recording в нижней части этого окна активизируется при разрешении регистрации нажатий клавиш (это можно сделать с помощью TDINST или параметра -k командной строки).

Область Keystroke Recording показывает причину передачи управления отладчику (например, точка останова) и текущий адрес программы с соответствующей строкой исходного кода или машинной инструкцией. Turbo Debugger регистрирует все нажимаемые клавиши и записывает их в файл XXXX.TDK, где XXXX - это имя отлаживаемой программы. Локальное меню этой области содержит команды Inspect и Keystroke Restore. По команде Inspect отладчик активизирует окно Model или CPU, в котором курсор позиционирован на ту строку, где нажата клавиша. Команда Keystroke Restore перезагружает программу и выполняет ее до строки, подсвеченной в области Keystroke Recording.

**Завершение программы**

При завершении работы программы управление передается в Turbo Debugger, который выводит код выхода программы. После этого любая команда меню Run перезагружает программу. После завершения программы проверить или модифицировать ее переменные нельзя.

При выполнении программы в отладчике легко случайно пропустить нужное место. В этом случае вы можете возобновить сеанс отладки с помощью команды Run Program Reset (Ctrl+F2), которая перезагружает программу с диска. Перезагрузка программы не влияет на точки останова и параметры просмотра.

**Выход из отладчика**

Завершить сеанс отладки и вернуться в администратор программ Windows вы можете в любое время (за исключением передачи управления в программу или работы с диалоговым окном) с помощью клавиш Alt+X. Можно также выбрать команду File Quit.