

Продолжение. Начало в № 3 `2009

Александр ШАЛАГИНОВ
shalag@vt.cs.nstu.ru

На первом уроке мы уже узнали об инструменте, который называется **Design Flow Manager** (менеджер маршрута проектирования). Это не просто картинка. Под курсором мыши отдельные фрагменты рисунка меняют цвет (на рис. 1 выделена левая кнопка), значит, под ними спрятаны ссылки на ту или иную программу или диалоговую панель. А картинка показывает, в каком порядке мы должны действовать.



Рис. 1. Простейший маршрут проектирования в среде Active-HDL 7.1

Пиктограмма **HDE** запуска текстового редактора **HDL Editor** стоит первой слева. Значит, сначала нужно щелкнуть по этой кнопке, вызвав основной инструмент создания проекта. Заметим, что в ранних версиях пакета никаких других средств описания проекта пользователю и не предлагалось.

Появится небольшая панель (рис. 2), где нам следует выбрать язык описания проекта (VHDL, Verilog или SystemC). По умолчанию переключатель установлен на языке проектирования VHDL, и мы согласимся с предлагаемым выбором.

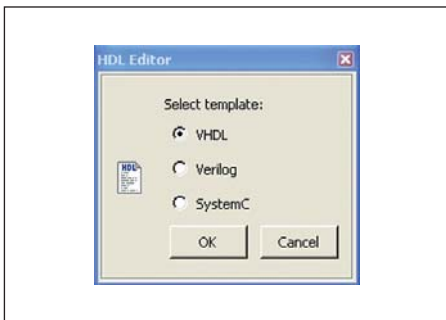


Рис. 2. Выбираем язык описания проекта VHDL

После нажатия на кнопку **OK** запускается мастер создания нового исходного файла, и на экране появляется его диалоговая панель **New Source File Wizard** (рис. 3). По умолчанию на ней установлен флажок **Add the generated file to the Design** («Добавить создаваемый файл к проекту»). Если вы не знаете, как поступить, то во всех случаях лучше сле-

Изучаем Active-HDL 7.1. Урок 5. Создание проекта в текстовом формате

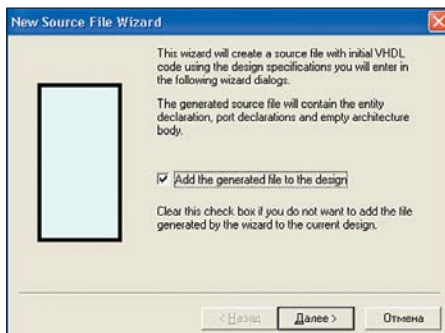


Рис. 3. Первая диалоговая панель мастера нового исходного файла

довать простому правилу — соглашаться с установками, заданными по умолчанию.

Нажав на кнопку «Далее», увидим следующую панель мастера (рис. 4), где требуется ввести три имени: имя исходного файла (**source file**), имя объекта проекта (**entity**) и имя его архитектурного тела (**architecture**).

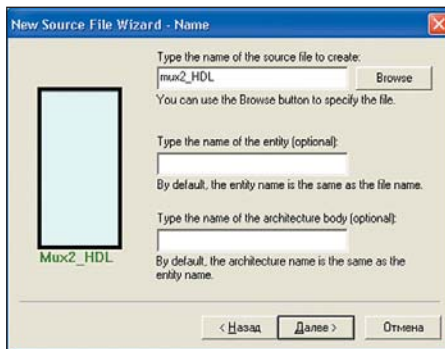


Рис. 4. На второй панели вводим имя исходного файла mux2_HDL

Впрочем, можно ограничиться только именем исходного файла. Остальные имена будут позаимствованы от него по умолчанию.

Далее уже описанным ранее способом (урок 1) вводим имена внешних сигналов (портов), задаем им направление передачи данных (входной или выходной порт) и нажимаем на кнопку «Готово» (рис. 5).

По этой команде управление будет передано текстовому редактору **HDL Editor**. В окне просмотра проекта **Design Browser** появится имя только что созданного файла **mux2_HDL.vhd**, а его содержимое отобра-

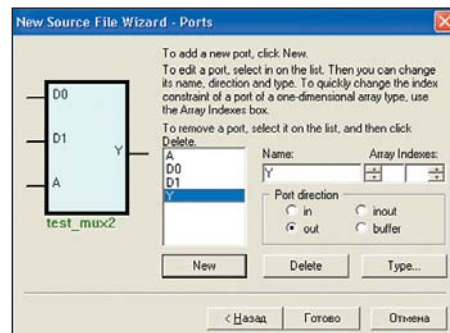


Рис. 5. Вводим имена портов и задаем направление передачи сигналов (входы, выходы)

зится на новой закладке в главном окне среды проектирования **Active-HDL 7.1**.

Окно текстового редактора HDL Editor

Обратите внимание, в окне текстового редактора (рис. 6) мы видим не пустой файл, а фактически заготовку (шаблон) будущего проекта.

Для удобства дальнейшей работы закладку можно распахнуть на весь экран (кнопка в верхнем правом углу экрана) или открыть файл с **HDL** описанием в отдельном окне (кнопка там же). Тот же результат получится, если щелкнуть правой кнопкой мыши на имени закладки **mux2_HDL.vhd** и в открывшемся контекстном меню (рис. 7) выполнить команду **Undock Window** («Отстыковать окно»).

Однако удобнее всего использовать для этих целей горячие клавиши **Alt+F9**. Заметим, что описанные манипуляции применяются для любой активной закладки в рабочем окне пакета **Active-HDL 7.1**.

На рис. 6 видны две основные инструментальные панели текстового редактора: **HDE Tools** и **HDE Edit**. По умолчанию они вытянуты в линейку и размещены в верхней части рабочего окна. При желании вы можете легко изменить их местоположение и конфигурацию.

Инструменты редактора HDL Editor

На рис. 8 показаны основные кнопки панели инструментов **HDE Tools** (мы переместили ее на рабочую область редактора).

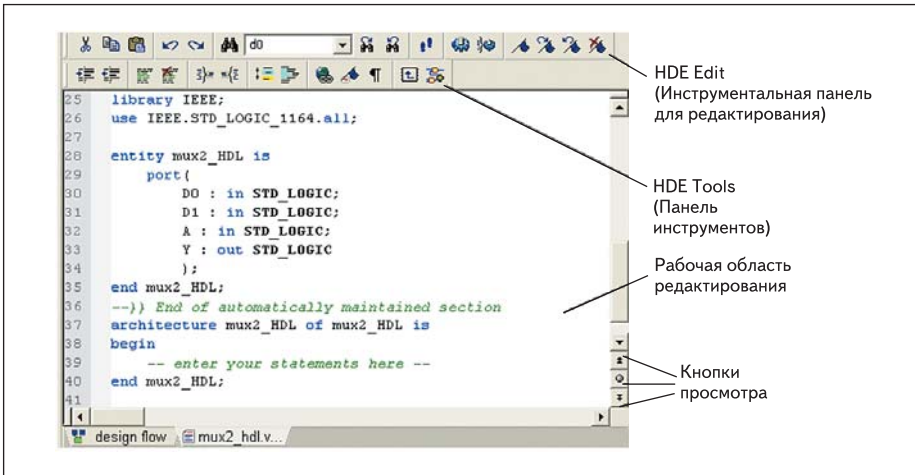


Рис. 6. Окно текстового редактора HDL Editor с шаблоном будущего проекта

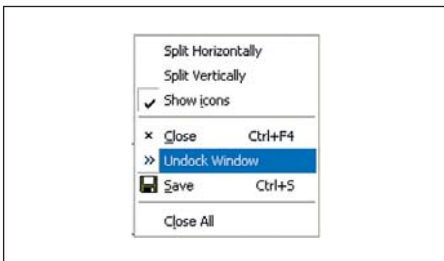


Рис. 7. Команда Undock Window

В скобках рядом с иностранным названием пиктограммы указаны горячие клавиши для быстрого вызова соответствующей команды. Любой программист при написании программы заботится о том, чтобы созданный им продукт был удобен для восприятия. И неважно, кто его будет читать — сам создатель или другой пользователь. Удобочитаемость кода заметно повышается, если он должным образом структурирован, форматирован и документирован (снабжен комментарием).

Под структурированием в пакете **Active-HDL** понимается процесс разбиения текста программы на блоки (группы, group), содержащие синтаксически законченные конст-

рукции языка описания аппаратуры, и раскраска их. Редактор **HDL Editor** поддерживает иерархию описаний, когда один блок может быть вложен в другой.

Перед структурированием полезно выполнить пробную компиляцию, чтобы убедиться, что написанный отрезок кода не содержит синтаксических ошибок.

Если вы хотите автоматически структурировать весь документ, то следует нажать на кнопку **Generate Structure**. За неимением более серьезного документа, применим этот инструмент к шаблону модели мультиплектора, показанной на рис. 6.

Прежде всего, обратим внимание на две вещи: раскраску кода и разбиение его на группы (рис. 9).

Например, блок **entity** выделен сиреневым цветом и раскрыт на верхнем уровне, тогда как список портов **port** имеет более темный оттенок и, наоборот, свернут до одной (первой) строки. Пустой блок **architecture** тоже раскрыт и имеет светло-зеленую окраску. Заметим, что цвет группы и ее первоначальное состояние (раскрыта или свернута в строку) можно настраивать по своему желанию (команда **Preferences** из меню **Tools**).

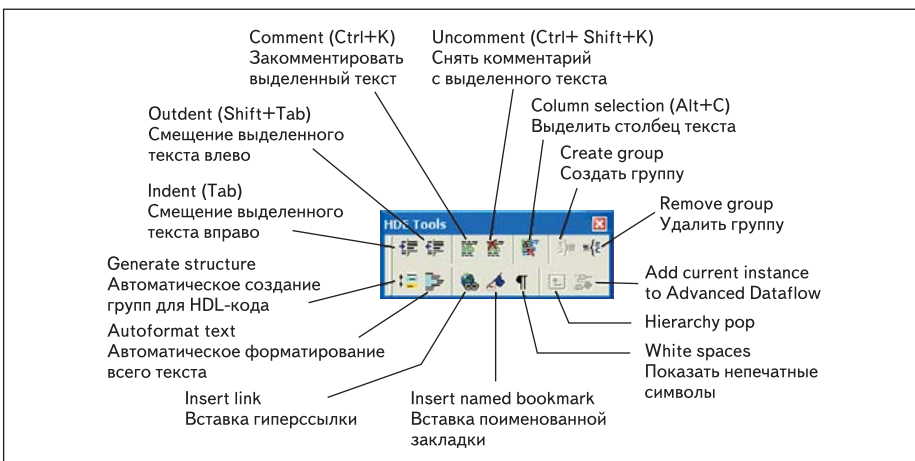


Рис. 8. Назначение кнопок панели инструментов HDE Tools

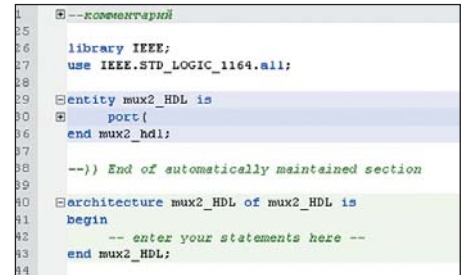


Рис. 9. Структурирование VHDL-кода разбивает его на группы (блоки) и раскрашивает их разными цветами и оттенками

Чтобы убрать все следы структурирования кода (разбиение на группы и их раскраску), достаточно нажать на кнопку **Remove groups**.

В принципе вы можете отказаться от автоматического структурирования и выполнить эту процедуру вручную, манипулируя кнопками **Create group** и **Remove groups**. Именно таким способом создана группа **комментарий**, показанная на рис. 9 (вверху). Здесь появляются дополнительные возможности: можно создавать группы из любого числа строк и даже нарушать границы синтаксически завершенных конструкций языка описания аппаратуры. Более того, ручное структурирование возможно и поверх автоматически структурированного текста.

Обратите внимание, структурирование текста программы не включает в себя операции форматирования, то есть перемещения строк друг относительно друга, с целью выделить основные элементы синтаксических конструкций, отразить иерархию описания и повысить, таким образом, удобочитаемость кода.

Для этого в пакете **Active-HDL** существует другая процедура, называемая автоматическим форматированием текста. Она выполняется в редакторе **HDL Editor** нажатием на кнопку **Autoformat text**. Выделите весь код (горячие клавиши **Ctrl+A**) и нажмите несколько раз на кнопку **Outdent**, чтобы выровнять текст по левому краю (рис. 10а).

А теперь выполните автоматическое форматирование, результат которого показан на рис. 10б.

Если вы предпочитаете ручную форматировать текст, то вам пригодятся кнопки **Indent** и **Outdent**, перемещающие выделенный фрагмент кода на одну позицию табуляции вправо или влево. При этом, правда, придется навсегда забыть про существование

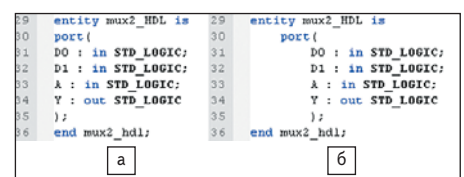


Рис. 10. Результат автоматического форматирования фрагмента VHDL-кода

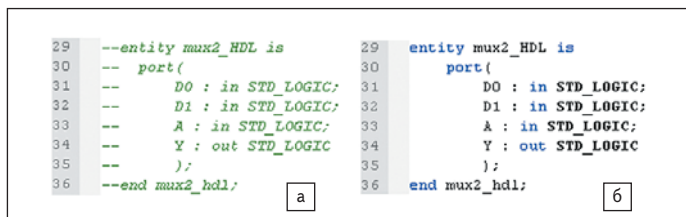


Рис. 11. Редактор HDL Editor позволяет с легкостью создавать и снимать комментарий с любого фрагмента VHDL-кода

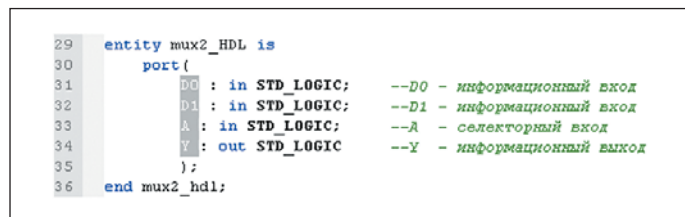




Рис. 12. Выделением столбца с именами портов просто и быстро создаем комментарий


команды автоматического форматирования, иначе вы рискуете потерять всю доселе выполненную работу по ручному форматированию своего кода.


Дело в том, что автоматическое форматирование выполняется над всем текстом, и совмещать оба режима у вас не получится.


В языке VHDL нет инструмента, позволяющего «одним махом» закрыть комментарий большой отрезок кода. Приходится ставить символы комментария (два дефиса подряд) в начале каждой строки.


Однако при отладке программных VHDL-моделей такая необходимость возникает довольно часто. Поэтому в редакторе HDL Editor имеется специальная кнопка **Comment** , позволяющая автоматизировать названную процедуру.

Выделите фрагмент текста, который требуется закомментировать, и нажмите кнопку **Comment** . В начале каждой строки выделенного текста редактор автоматически поставит символы комментария (рис. 11а).

Чтобы снять комментарий, укажите выделением желаемый фрагмент закомментированного кода и нажмите кнопку противоположного действия **Uncomment**  (рис. 11б).

Пожалуй, стоит обратить ваше внимание еще на одну кнопку инструментальной панели HDE Tools (рис. 8). Она называется **Column selection**  и позволяет включить (а при повторном нажатии — выключить) режим выделения столбца. На рис. 12 показано, как с помощью этого инструмента созданы четыре строки комментария. Выполним и мы эту работу.

Нажмем на кнопку **Column selection** , выделим столбец с именами входных и выходных портов D0, D1, A, Y (рис. 12) и скопируем его в буфер обмена. Переместим указатель мыши вправо и вставим копируемый столбец.

Нажмем на кнопку **Comment** , превращая введенный столбец в комментарий. Добавим сюда необходимые пояснения. Просто и быстро, не правда ли?

Настройка параметров редактора HDL Editor

Для настройки параметров текстового редактора следует исполнить команду **Tools/Preferences** и в разделе **Category** выбрать строку **HDL Editor** и язык **VHDL**. На экране по-

явится соответствующая диалоговая панель, фрагмент которой показан на рис. 13.

В верхней части панели находится переключатель **Auto Complete** («Автоматическое завершение»). По умолчанию он установлен в положение **Complete Word**. Это означает, что при вводе ключевых слов редактор будет предлагать вам автоматически дописать слово, распознавая его по нескольким первым буквам, которые вы успели напечатать.

Например, при вводе ключевого слова **entity** вы напечатали только две первые буквы. Редактор предложит завершить слово, автоматически допечатав его по шаблону **entity**. Если редактор угадал слово, достаточно нажать клавишу «Пробел» (**Space**).

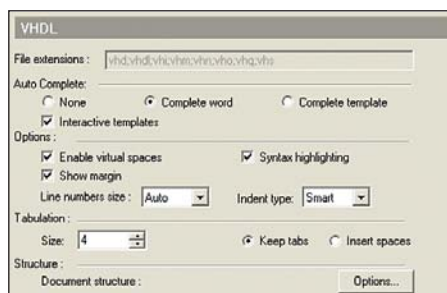


Рис. 13. Фрагмент диалоговой панели для настройки параметров HDL-редактора

В противном случае продолжайте ввод нужного вам слова до его ручного завершения или пока редактор не сделает другой прогноз. Например, вам нужно напечатать ключевое слово **process**. Введя первые две буквы, вы увидите, что редактор хочет закончить слово, предлагая вариант: **procedure**. Но такой исход нас не устраивает, поэтому продолжаем ввод, пока не дождемся нового прогноза: **process**.

А теперь сделаем еще один эксперимент. Снова введем две первые буквы ключевого слова **entity**, и когда редактор предложит завершить слово, нажмем вместо **Space** комбинацию клавиш **Ctrl+Enter**.

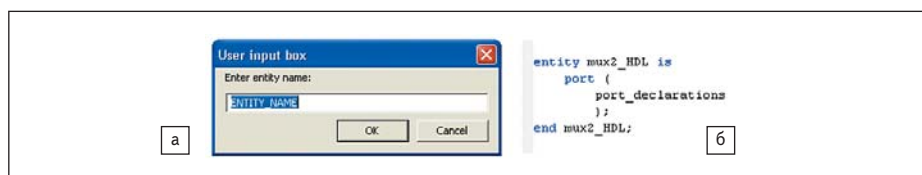


Рис. 14. В интерактивном режиме редактор сконструировал целый блок VHDL-кода

Откроется диалоговая панель (рис. 14а), на которой редактор HDL-кода запросит у вас имя объекта проекта **ENTITY_NAME**. Введем требуемое имя, например, **mux2_HDL**, и нажмем на кнопку **OK**. В рабочей области редактора появится законченная синтаксическая конструкция языка VHDL, в данном случае — интерфейс объекта проекта (рис. 14б).

На этом примере мы познакомились еще с одной полезной особенностью редактора — умением завершать не только ключевые слова, но и целые блоки VHDL-кода, причем в интерактивном режиме с системой.

Можно «заставить» редактор автоматически генерировать шаблоны VHDL-кода, избавив себя от необходимости вступать с ним в диалог. Для этого на панели **VHDL** (рис. 13) надо поставить переключатель **Auto Complete** в положение **Complete template** и сбросить флажок **Interactive templates**.

Вы уже обратили внимание, что редактор HDL Editor по умолчанию раскрашивает отдельные категории слов в различные цвета. Например, ключевые слова помечаются им синим цветом, комментарий — зеленым курсивом, а типы данных выделяются черным жирным шрифтом (рис. 6).

В принципе можно отказаться от цветовой гаммы и созерцать монохромный текст, если сбросить флажок **Syntax highlighting** на панели **Preferences**. Кстати, здесь же вы можете снять еще один флажок **Enable virtual spaces** и тем самым лишить себя возможности помещать точку вставки текста в любое место на рабочей области редактора. Отныне указатель мыши не удастся поставить правее невидимого символа перевода строки.

А вот сбрасывать установленный умолчанию флаг **Show margin** (рис. 13) нежелательно. Как только вы это сделаете, исчезнет полоса показа, расположенная вдоль левого края рабочей области редактора (рис. 15). Информация, размещаемая на этой полосе, весьма насыщена. Именно здесь указываются порядковые номера строк, а немного пра-

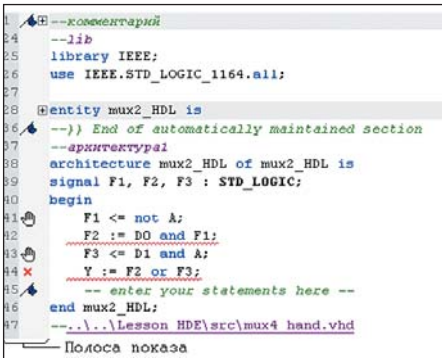














Рис. 15. Полоса показа с элементами управления и навигации

вее — символы закладок , контрольные точки прерывания (останова) , маркер ошибки , управляющие элементы, раскрывающие и сворачивающие  группу строк.

Сказанное, прежде всего, касается больших документов, в которых затруднена навигация и быстрый доступ к желаемому фрагменту текста. Самое простое, что приходит в голову, — запомнить номер строки, на которую надо вернуться, и при необходимости нажать кнопку **Go to (Ctrl+G)** , а затем ввести номер нужной строки. Но что делать, если таких фрагментов несколько?

В этом случае на помощь приходят закладки , показанные на рис. 15. Чтобы сделать такую закладку, надо указать мышью на нужную строку и щелкнуть по кнопке **Toggle bookmark** . Таких закладок можно поставить сколько угодно. В дальнейшем вы сможете легко перемещаться по документу, «опираясь» на закладки нажатием на кнопки **Next bookmark**  и **Previous bookmark** . Все вышеназванные кнопки находятся на инструментальной панели для редактирования **HDE Edit** (рис. 6).

Для удаления закладок предназначена иконка **Clear all bookmarks** , но с ее помощью вы избавитесь сразу от всех закладок. Чтобы удалить одну закладку, надо указать строку, которую она маркирует, и повторно нажать на кнопку **Toggle bookmark** .

Если закладок много, то работа с ними становится не очень удобной, потому что закладки «безымянные» и просматривать их можно только в режиме сканирования. В этом случае предпочтительнее выглядят именованные закладки. Две такие закладки показаны на рис. 15 (строки 24 и 37). Для размещения именованной закладки достаточно щелкнуть по кнопке **Insert named bookmark**  и ввести ее имя. Редактор маскирует такую закладку под комментарий, но ее выдает фиолетовый цвет. Обратите внимание, для задания имен закладок можно использовать и кириллицу (рис. 15, строка 37).

Следует отметить еще одну очень полезную особенность редактора **HDL Editor**: он интегрирован с компилятором и модельером пакета **Active-HDL 7.1**. Другими слова-

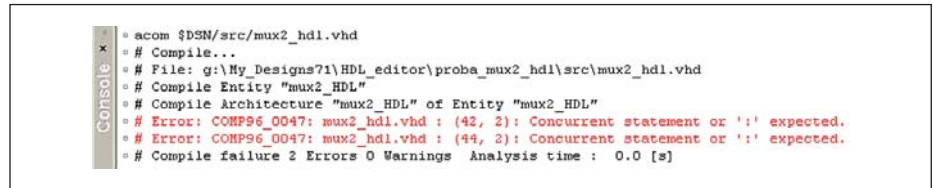


Рис. 16. Результаты компиляции отображаются в окне сообщений Console (красным цветом выделены обнаруженные ошибки в строках с номерами 42 и 44)

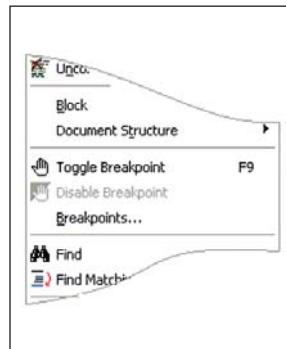


Рис. 17. Контекстное меню

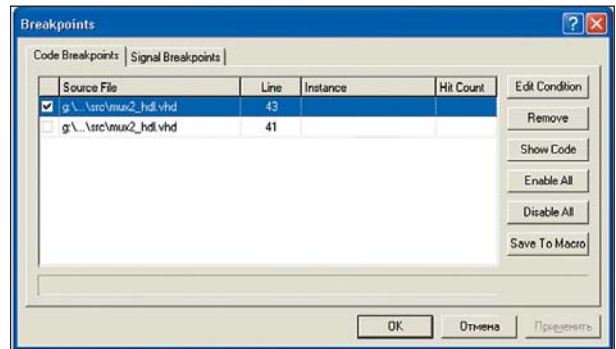




Рис. 18. Диалоговая панель Breakpoints раскрывает дополнительные возможности по управлению точками останова

ми, вы можете, написав небольшой отрезок кода, тут же проверить, нет ли в нем ошибок. Нажимаем на кнопку **Compile** , предлагая пакету откомпилировать активный документ (рис. 15), и тут же видим результат (рис. 16). В четырех написанных нами параллельных операторах назначения сделано две ошибки. Редактор подчеркивает строки с ошибками линией красного цвета, а в окне сообщений **Console** поясняет характер ошибки и уточняет ее местоположение.

Для большего эффекта закроем файл **mux2_HDL.hdl** и в окне **Console** дважды щелкнем мышью на нижней строке с ошибкой. Откроется редактор **HDL Editor**, и в его окне отобразится фрагмент текста со строкой, содержащей ошибку. При этом слева на полосе показа появится красный крестик — маркер выделенной ошибки (рис. 15). Теперь в окне **Console** укажите мышью на верхнюю строку с ошибкой, и вы увидите, что маркер ошибки переместился на строку с номером 42.

В процессе отладки VHDL-кода или верификации VHDL-модели нередко возникает необходимость «тормознуть» процесс моделирования, чтобы посмотреть текущее состояние моделируемого объекта, значения сигналов на его входах и реакцию на выходах. Для достижения этих целей в исходном коде следует расставить в желаемых местах так называемые контрольные точки прерывания (называемые также точками останова). Пиктограммами такая команда не поддерживается, поэтому щелкнем правой кнопкой мыши на строке, перед которой вы хотите остановить моделирование, и исполним команду **Toggle Breakpoint** из контекстного меню (рис. 17).

Конечно, удобнее пользоваться горячей клавишей **F9**, повторное нажатие на которую удаляет точку останова. Если вы не уверены, что она вам больше не понадобится, то лучше ее просто заблокировать, активизировав команду **Disable Breakpoint** в том же меню. Блокированная точка останова останется на своем месте, но изменит цвет заливки , а главное, ее в любое время можно «оживить».

Другой способ управления точками останова призывает на помощь диалоговую панель **Breakpoints** (рис. 18). Здесь появляются дополнительные возможности, но мы не станем погружаться в детали.

Режимы просмотра исходного кода

Ранее упоминалось, что при работе с большими документами возникает проблема размерности. Процедура структурирования в какой-то мере снижает остроту этой проблемы, потому что целые блоки текста можно свернуть до размеров одной строки. Но полностью избавиться от прокрутки документа не удастся. И тогда на помощь приходят механизмы быстрого просмотра исходного кода.

В среде проектирования **Active-HDL 7.1** они объединяются в инструмент, управляющие кнопки которого находятся в нижнем правом углу окна **HDL-редактора** (рис. 19).

Похожая организация просмотра нам уже встречалась, когда мы обсуждали работу с редактором временных диаграмм **Waveform Editor** (урок 4). Правда, там требовалась прокрутка длинных временных диаграмм.

Чтобы выбрать способ просмотра редактируемого кода, следует щелкнуть мышью на

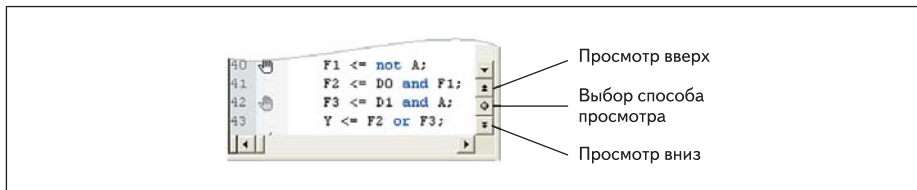


Рис. 19. Управляющие кнопки для эффективного просмотра больших документов

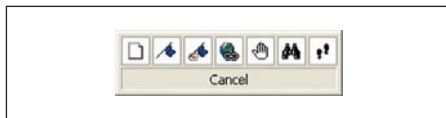









Рис. 20. Возможные режимы просмотра текста

средней кнопке **Browse select** («Выбор способа просмотра»). Редактор покажет возможные режимы быстрого просмотра документа (рис. 20).

Большинство кнопок нам уже знакомо, а две, обойденные вниманием, — общеизве-

стны. Вы можете «пролистывать» документ по страницам (экранам) , продвигаться по тексту, опираясь на закладки  или точки останова , сразу попадать на нужный фрагмент кода, отмеченный поименованной закладкой , гиперссылкой  или порядковым номером строки .

Понятно, что редактор HDL-кода поддерживает и классический поиск по заданному шаблону . Обратите внимание на цвет кнопок перемещения. Он зависит от выбранного режима, например, при движении по закладкам цвет кнопок синий, а при перемещении по точкам останова — красный. ■

Продолжение следует