Продолжение. Начало в № 3`2008

Краткий курс HDL.
Часть 4. Расположение шаблонов текстовых файлов в программных инструментах различных фирм-производителей

Иосиф КАРШЕНБОЙМ
iosifk@narod.ru

Часто можно услышать такие высказывания: «Если я работаю в схемном проекте, то мне только надо взять из меню символ компонента и поместить на поле чертежа. Пара проводов — и готово. А в случае текстового описания для каждого триггера или даже для каждого вентиля надо написать целый кусок текста...»

На самом деле это не совсем так. При выполнении проекта на языках HDL разработчик может использовать программные инструменты, представляемые фирмами-изготовителями, третьими фирмами, а также специализированные текстовые редакторы. Такие текстовые редакторы не только позволяют выполнять обычное редактирование, но и имеют целый ряд дополнительных функций. В качестве примера автор может привести редактор EditPlus2. В этом редакторе пользователь может определить и установить свои собственные шаблоны текста. А это значит, что по двойному клику на шаблоне в редактируемый файл вставится фрагмент текста, в котором уже будут необходимые вам имена проводов и комментарии. Таким образом пользователь сможет быстро произвести все необходимые действия и создать файл в полуавтоматическом режиме.

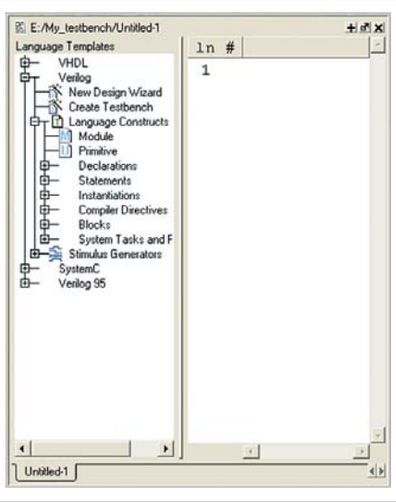

Рис. 1. Окно программы ModelSim, в котором находятся шаблоны конструкций языков Verilog и VHDL

Рассмотрим программные инструменты фирм-производителей. Современные программные инструменты, как поставляемые фирмами-производителями, так и независимыми фирмами, имеют в своем составе визарды, позволяющие генерировать довольно сложные блоки, такие как FIFO, UART и даже микроконтроллеры. Но, вместе с тем, все эти программные инструменты также имеют возможность вставлять в проект пользователя шаблоны обычных функций, таких как триггеры, вентили и т. д.

Как же добраться до шаблонов текстового редактора в различных программных инструментах?

В программном инструменте — симуляторе ModelSim

Для того чтобы попасть в раздел шаблонов в симуляторе ModelSim, необходимо открыть для редактирования любой текстовый файл. Далее, на поле текстового файла нажимаем правую клавишу мыши и в выпадающем меню выбираем пункт Language Template. Программа просит немного подождать и открывает окно шаблонов, показанное на рис. 1.

В левой части окна пользователь может выбрать язык и требуемый шаблон. При выделении какого-либо шаблона и двойному клику по этой надписи, в правую половину окна будет выведен текст этого шаблона.

В разделе Language Constructs есть визард, который помогает сделать описание счетчика:

а) вводим название и разрядность счетчика;
б) вводим название синхроимпульсов и указываем их активный фронт;
в) вводим название сигнала сброса и указываем его активный фронт.

Внешний вид окон для ввода параметров счетчика показан на рис. 2.

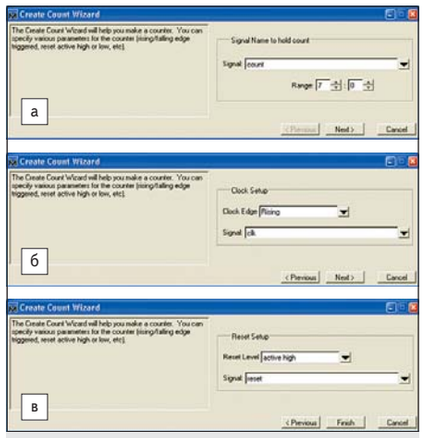

Рис. 2. Визард в окне Language Constructs, который помогает сделать описание счетчика

В примере 1 показан результат работы визарда по созданию описания счетчика.

```
wire clk;
wire reset;
reg[7:0] count;

always @(posedge clk)
begin
   if (reset == 1)
      count <= 'b0;
   else
      count <= count + 1;
end
```

Пример 1. Результат работы визарда по созданию описания счетчика

В программном инструменте ISE для работы с микросхемами фирмы Xilinx

Для того чтобы попасть в раздел шаблонов в программном инструменте ISE, необходимо выбрать в главном меню пункт Edit → Language Template. Откроется окно, внешний вид которого представлен на рис. 3. В левой части окна пользователь может выбрать язык и требуемый шаблон. При выделении какого-либо шаблона и двойному клику по этой надписи в правую половину окна будет выведен текст этого шаблона.

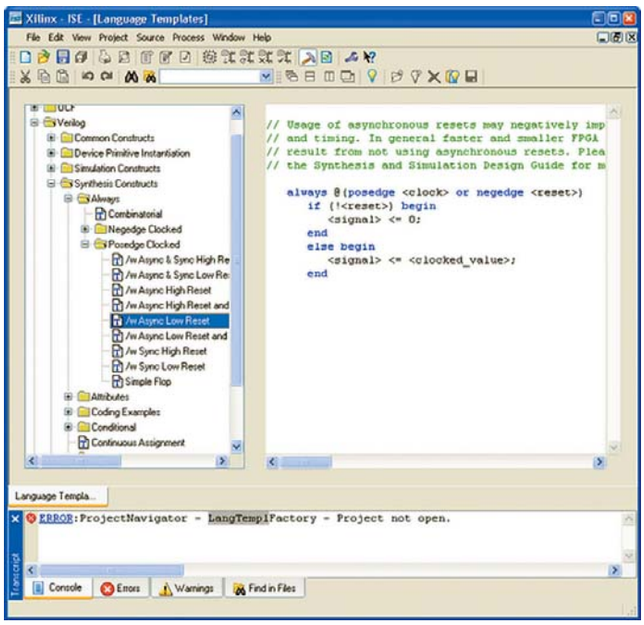

Рис. 3. Окно программы ISE, в котором находятся шаблоны конструкций языков Verilog и VHDL

В программном инструменте ISE для работы с микросхемами фирмы Altera

Автор благодарит Р. Золотухо (roman@efo.ru), сотрудника фирмы «ЭФО», за помощь в создании описания по работе с Quartus.

В Quartus нужно создать новый или открыть существующий текстовый файл. После этого в меню Edit выбрать пункт Insert Template. Откроется окно выбора шаблонов для текстового описания проектов с использованием различных языков описания аппаратуры: AHDL, VHDL, Verilog, SystemVerilog, а также команд TCL. Внешний вид этого окна показан на рис. 4.

Это же окно можно открыть, щелкнув по пиктограмме “Insert Template” в строке инструментов окна открытого текстового файла.

В программном инструменте Libero IDE для работы с микросхемами фирмы Actel

Автор благодарит С. Карпова (karpov@actel.ru), сотрудника фирмы «Актел.ру», за помощь в создании описания по работе с Libero.

Что касается таких программ, как ISE или Quartus, то сейчас уже создано довольно много книг, статей и прочих документов, в которых приводятся описания работы данных программ, а также описания их интерфейса. Поэтому автор ограничился только кратким описанием «путеводителя» по шаблонам языковых конструкций для этих программ. Но по использованию программы Libero IDE русскоязычных материалов гораздо меньше, поэтому в этом разделе описание Libero сделано более подробным, нежели описания предыдущих программ.

В программе Libero IDE имеется инструмент SmartGen, предназначенный для работы с примитивами, шаблонами и IP-компонентами (рис. 5).

В окне SmartGen есть четыре закладки:

- Cores — IP-компоненты.
- Templates — HDL-шаблоны на языках VHDL и Verilog.
- Bus Defenitions — перечень используемых интерфейсных шин и их параметры (информационная закладка).
- Actel Cell Library — библиотека примитивов Actel. Может использоваться для разработки блоков в SmartGen.

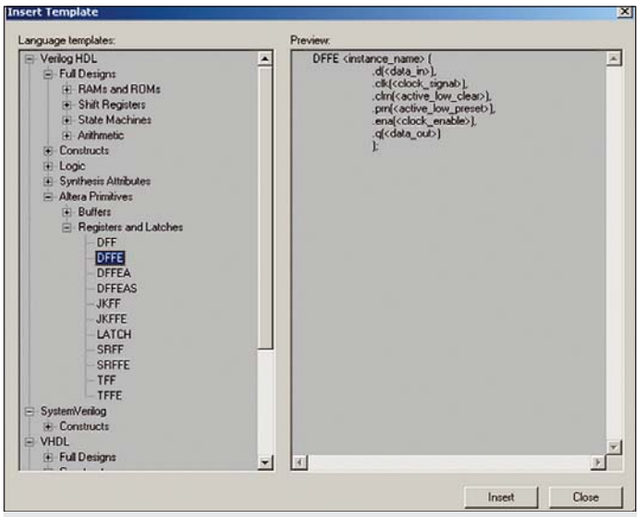

Рис. 4. Окно программы Quartus, в котором находятся шаблоны конструкций языков Verilog и VHDL

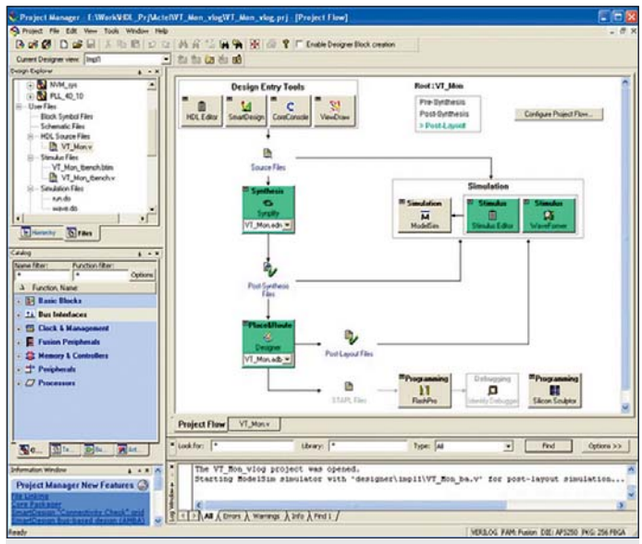

Рис. 5. Окно Actel Libero IDE

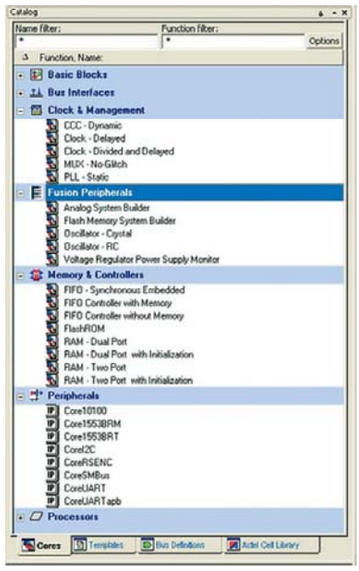

Рис. 6. Список каталогов для семейства Fusion

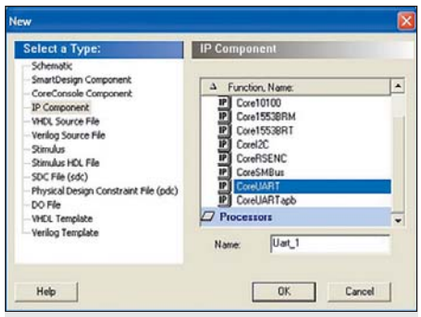

Рис. 8. Задание имени IP-компонента (UART)

Рис. 9. Настройка параметров IP-компонента (UART)

Закладка "Core"

В этом окне приводится список доступных для использовании в проекте IP-компонентов. При настройке можно установить, в каком виде этот список будет представлен — как алфавитный или иерархический с разделением на группы. По умолчанию список представлен в иерархическом виде. Количество и состав групп и компонентов зависит от выбранного семейства и микросхемы. Первым в списке идет группа базовых блоков (Basic Blocks). Эта группа присутствует всегда. В ней перечислены основные базовые блоки — регистры, счетчики, сумматоры, мультиплексоры и т. д. Наличие и состав остальных групп определяется семейством микросхем FPGA. В этих группах есть компоненты для конфигурации интегрированных контроллеров, периферии, памяти (PLL, FIFO/SRAM, RTC (только семейство Fusion) и т. д.), а также программных IP-компонентов (CoreUART, CoreI²C, CorePCI), включая процессорные ядра (CoreMP7 (ARM7) и CoreM1(Cortex-M1)). На рис. 6 показано окно программы со списком каталогов для семейства Fusion.

Для использования компонента надо раскрыть соответствующую группу, нажав мышкой на [+] напротив группы или два раза кликнув мышью по названию. В раскрывшемся списке дважды кликните по требуемому компоненту. Если выбран базовый блок или интегрированная периферия, память или контроллер, то откроется панель конфигурации выбранного компонента. Установите требуемые параметры и нажмите кнопку "Generate…" (рис. 7).

В том случае, если выбран программный IP-компонент, сначала откроется панель, где необходимо в поле "Name" назначить ему имя, с которым он будет применяться в проекте пользователя (рис. 8), и pfntv нажать кнопку "OK".

Будет сгенерирован шаблон и появится панель настройки (рис. 9).

Далее необходимо установить требуемые параметры и нажать кнопку "OK". Будет сгенерирован программный код на языке, выбранном для данного проекта, и описание этого компонента появится в списке компонентов в окне файлов проекта. Сгенерированный код можно посмотреть и, при необходимости, отредактировать.

Закладка "Templates"

Здесь присутствуют HDL-шаблоны языка Verilog и VHDL, среди которых есть основные конструкции языка, такие как директивы компилятора, модули, операторы и т. д., а также конструкции для тестбенча и синтеза (рис. 10).

Для использования этого режима необходимо раскрыть список, выбрать нужную конструкцию и кликнуть по ней два раза мышкой. Выбранный шаблон будет скопирован в буфер обмена. Затем можно перейти к файлам проекта и вставить в них этот шаблон.

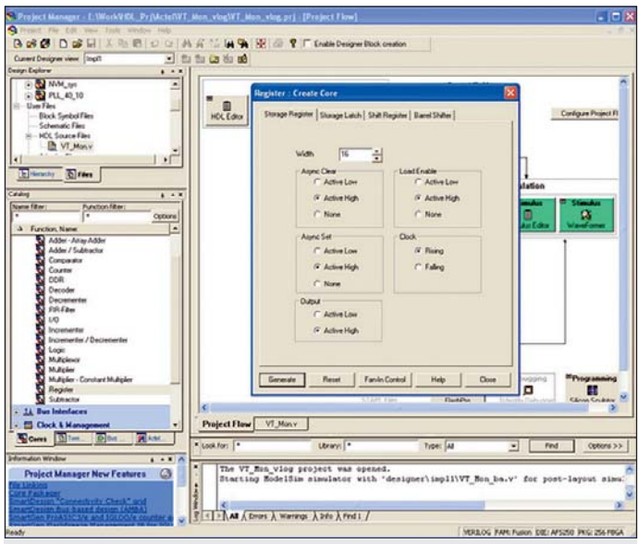

Рис. 7. Генерация 16-разрядного регистра

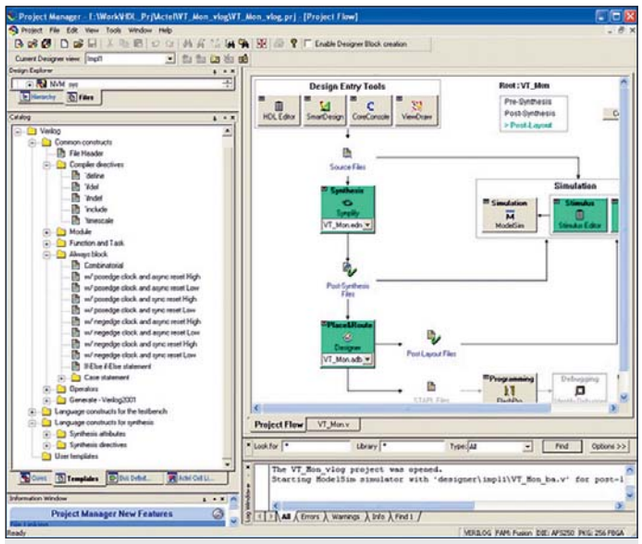

Рис. 10. Закладка "Templates". Шаблоны языка Verilog и VHDL

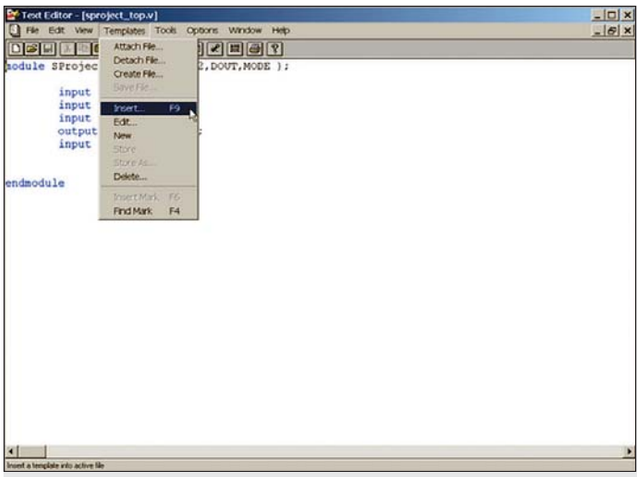

Рис. 11. Внешний вид окна текстового редактора

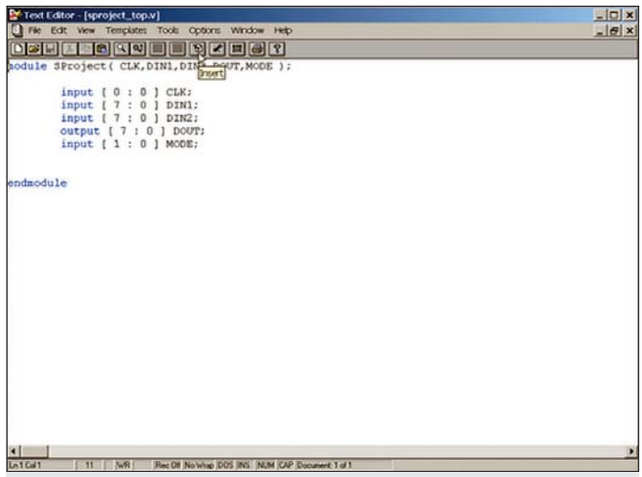

Рис. 12. Внешний вид окна текстового редактора

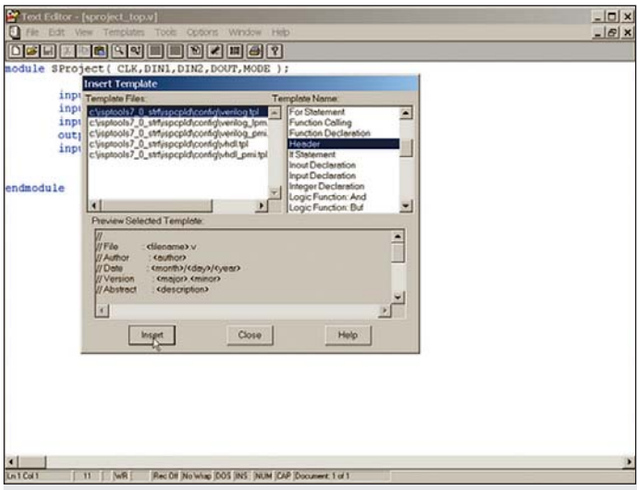

Рис. 13. Внешний вид окна для выбора шаблона в текстовом редакторе

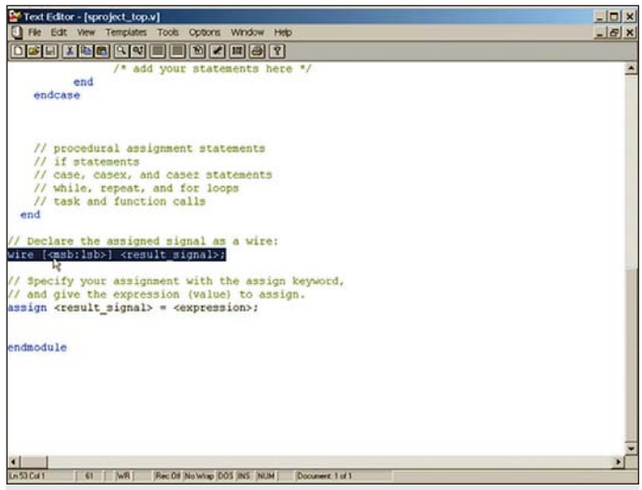

Рис. 14. Внешний вид окна текстового редактора с созданным шаблоном

В программном инструменте ispLEVER для работы с микросхемами фирмы Lattice

Автор благодарит А. Киселева (kiselev@ microem.ru), сотрудника фирмы «МикроЭм», за помощь в создании описания по работе с ispLEVER.

Представим, что мы создаем новый проект. В основном окне программы нажимаем **File/New** Project. В открывшемся окне вводим название проекта, его размещение, тип исходных файлов и синтезатор. Нажимаем «Далее». В следующем окне выбираем целевую микросхему. Нажимаем «Далее». В следующем окне никаких готовых файлов включать не будем. Еще раз нажимаем «Далее». Проверяем, все ли правильно. Нажимаем «Готово». Правой кнопкой мыши нажимаем на проект. Выбираем вкладку “new”. Указываем, что это новый модуль. Нажимаем “OK”. Открывается окно текстового редактора. В нем вводим название модуля, имя файла, описываем входные/выходные сигналы (это можно и не делать, тогда потом нужно будет сделать это вручную). Нажимаем “OK”.

Вот теперь мы добрались до текстового редактора (рис. 11).

В текстовом редакторе выбираем **templates/insert F9** или нажимаем соответствующую иконку. В окно редактора выводится фрагмент текста (рис. 12).

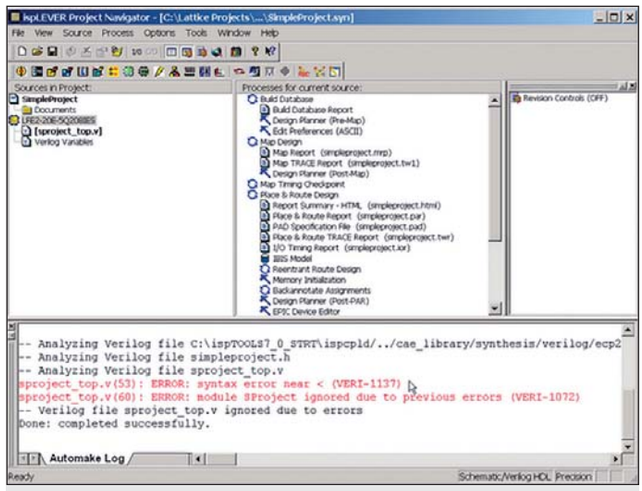

Рис. 15. Внешний вид окна с сообщениями об ошибках

На панели **template files** выбираем **verilog.tpl**, на панели **template name — header**. На панели **preview selected template** можно просмотреть тело шаблона. Вставляем шаблон (рис. 13).

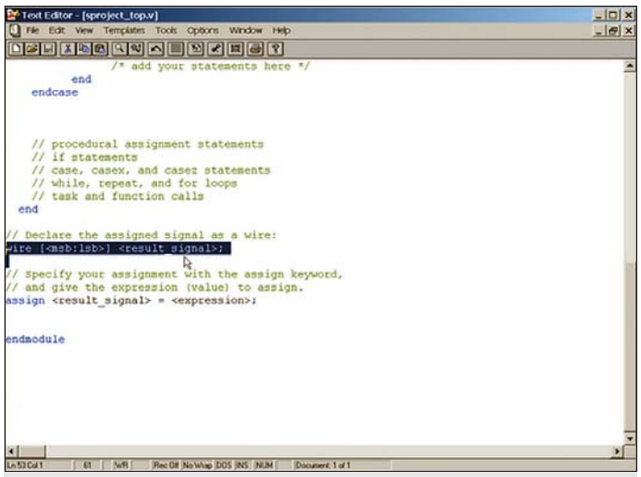

Рис. 16. Внешний вид окна текстового редактора и строка, содержащая ошибку

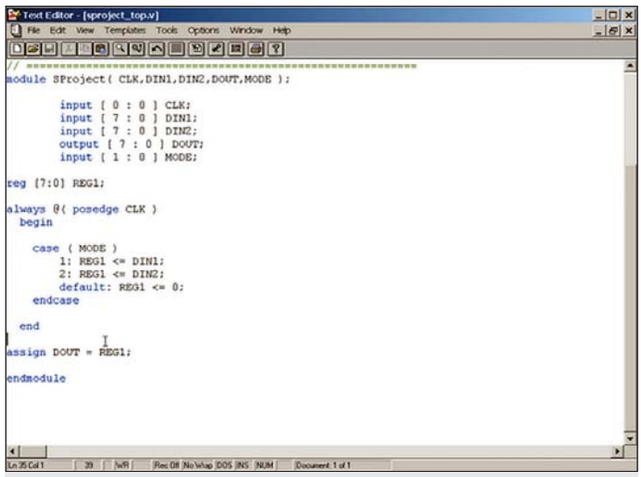

Рис. 17. Внешний вид окна текстового редактора с исправленным текстом

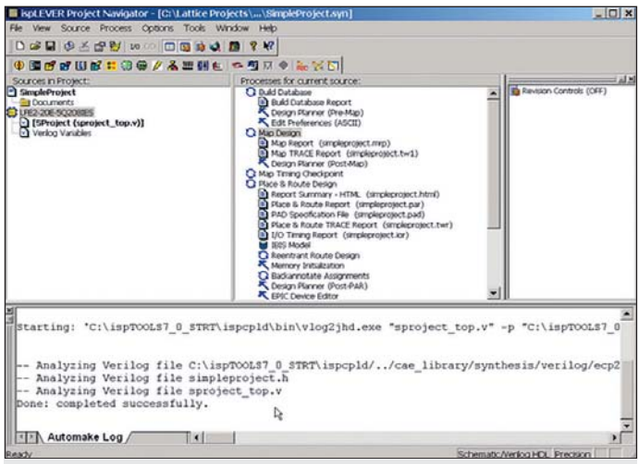

Рис. 18. Внешний вид окна навигатора, в том случае, когда ошибок не обнаружено

Далее, не закрывая окошко шаблонов, правим заголовок. Вставляем шаблон **register description**. Точно так же вставляем шаблон **always block**. Устанавливаем курсор на список событий и вставляем шаблон **rising edge**. Затем в тело процесса вставляем **case statement**. И после процесса вставляем **continuous assignment**. Закрываем окно шаблонов. Для того чтобы проиллюстрировать работу программы ispLEVER, преднамеренно создадим текст с ошибкой и посмотрим, как отреагирует ispLEVER. Итак, вставленный текст явно содержит ошибки (рис. 14).

Сохраняем сделанный нами фрагмент кода и переходим в окно **project navigator**.

При сохранении программа проверила текст и обнаружила лексические ошибки, о чем и сообщила пользователю (рис. 15).

При двойном нажатии на ошибку откроется окно текстового редактора и будет выделена строка, содержащая ошибку (рис. 16).

Правим текст и сохраняем файл еще раз (рис. 17).

В окошке навигатора видим, что ошибок не обнаружено (рис. 18).

В следующем разделе будут представлены описания типовых узлов — триггеров, счетчиков, дешифраторов.