621.382.8(076.5)
Р851

№ 3297

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Таганрогский государственный радиотехнический университет

КАФЕДРА КОНСТРУИРОВАНИЯ ЭЛЕКТРОННЫХ СРЕДСТВ

РУКОВОДСТВО К ЛАБОРАТОРНОЙ РАБОТЕ

РАЗРАБОТКА VHDL-ОПИСАНИЙ СБИС В ПОДСИСТЕМЕ RENOIR САПР FPGA ADVANTAGE

Для студентов специальностей 220500, 201900, 200200, 200800, 220100, 220300 и направлений 551100, 550700

ФЭП

Таганрог 2002

УДК 621.382.8:658.512.2.011.5(076.5)

Составители: Е.А. Рындин, В.Г. Ивченко, А.В. Ковалев

Руководство к лабораторной работе «Разработка VHDL-описаний СБИС в подсистеме Renoir САПР FPGA Advantage». Таганрог: Изд-во ТРТУ, 2002. 24 с.

Цикл лабораторных работ по освоению студентами методов проектирования специализированных сверхбольших интегральных схем (СБИС) на основе ПЛИС и в виде заказных микросхем подготовлен сотрудниками кафедры конструирования электронных средств (КЭС) Таганрогского государственного радиотехнического университета (ТРТУ).

В работе излагаются сведения, необходимые для описания проектов специализированных СБИС на языке VHDL в подсистеме Renoir САПР FPGA Advantage Mentor Graphics. Описаны возможности подсистемы Renoir, графический интерфейс, файловая организация проектов, маршруты синтеза и генерации VHDL-описаний, особенности работы с VHDL-библиотеками, связь подсистемы Renoir с подсистемами ModelSim и Leonardo Spectrum САПР FPGA Advantage.

Ил. 13. Библиогр.: 3 назв.

Рецензент В.Ф. Гузик, д-р техн. наук, профессор, заведующий кафедрой вычислительной техники ТРТУ.

ВВЕДЕНИЕ

Основными проблемами проектирования современных сверхбольших интегральных схем (СБИС), содержащих миллионы полупроводниковых структур на кристалле, являются обеспечение бездефектности и сокращение времени проектирования. Учитывая крайне высокую функциональную сложность СБИС, решение данных проблем возможно лишь посредством использования различных методов автоматизации в системах автоматизированного проектирования (САПР), опирающихся на мощную вычислительную базу.

На данный момент практически все СБИС проектируются с использованием специальных языков высокого уровня для описания электронной аппаратуры, являющихся формальными записями, предназначенными для описания функции и организации электронных систем. Функция системы определяется как преобразование значений на входах в значения на выходах, а организация задается перечнем связанных компонентов [1, 2].

В зависимости от метода последующей реализации проектируемой СБИС (на основе программируемой логической интегральной схемы (ПЛИС), базового матричного кристалла (БМК), заказной микросхемы) выбирается комплекс аппаратно-программных средств САПР и наиболее удобный язык описания проекта. Следует отметить, что все современные САПР (например, Mentor Graphics, Cadence, Altera MAX+plus II и др.) используют только языки описания, принятые в качестве международных стандартов. К ним относятся VHDL, Verilog, AHDL и др. Наиболее гибким и универсальным из них является VHDL (Very high speed integrated circuits Hardware Description Language - язык описания аппаратуры на основе высокопроизводительных интегральных схем), используемый для описания вычислительных систем любого уровня сложности и конструктивной иерархии (микросхема, печатный модуль, блок, ЭВМ, вычислительный комплекс и т.д.). Язык VHDL может быть использован на всех этапах разработки электронных систем: проектирование, верификация, синтез, тестирование, передача данных о проекте [2].

Одним из основных отличий языков описания аппаратуры и, в частности, VHDL от традиционных языков программирования (Паскаль, C++ и др.) является возможность ветвления, распараллеливания, конвейеризации потоков данных, без чего, как правило, невозможно проектирование высокопроизводительных СБИС. В результате при проектировании систем высокого уровня сложности возникает необходимость использования наряду с текстовыми VHDL-описаниями графического интерфейса, позволяющего визуализировать параллельные потоки информации и тем самым сократить количество ошибок, время проектирования, значительно облегчить труд разработчика. Именно таким интерфейсом, а также рядом дополнительных возможностей обладает подсистема Renoir САПР FPGA Advantage Mentor Graphics, о которой и пойдет речь в данной работе.

1. ОБЩАЯ ОРГАНИЗАЦИЯ САПР FPGA ADVANTAGE

САПР FPGA Advantage Mentor Graphics предназначена для автоматизированного проектирования СБИС с использованием языков VHDL или Verilog с последующей реализацией на основе ПЛИС и/или в виде заказных микросхем.

В состав САПР FPGA Advantage входят три основных подсистемы:

- Renoir - подсистема синтеза, генерации и верификации VHDL/Verilog-описаний проекта;
- ModelSim - подсистема функционально-логического моделирования проекта, в том числе с учетом задержек в элементах и соединительных линиях, осуществляемого после размещения и трассировки (back-annotate моделирование);
- Leonardo Spectrum - компилятор VHDL/Verilog-описаний проектов СБИС в стандартные форматы описания структуры (файлы формата EDIF) и задержек в элементах и линиях связи (файлы формата SDF) для последующей реализации в виде ПЛИС или заказных микросхем с использованием любых соответствующих САПР (например, Altera MAX+plus II, Tanner Pro, Cadence и т.д.). Причем передача данных о проекте и запуск следующей САПР происходит автоматически по заданным настройкам системы [3].

Данные подсистемы объединены файловой оболочкой проектов Design Browser, позволяющей создавать, открывать, закрывать, копировать файлы проектов и библиотек, а также осуществлять запуск подсистем Renoir, ModelSim и Leonardo Spectrum для файлов любого уровня иерархии проектов.

Существуют реализации программного обеспечения САПР FPGA Advantage под операционные системы UNIX и Windows 95-2000/Windows NT. Сведения о минимальной конфигурации аппаратных средств для работы с САПР FPGA Advantage приведены в табл. 1 [3].

Таблица 1

Минимальная конфигурация аппаратных средств для САПР FPGA Advantage

Процессор	Pentium
Оперативная память	32 MB, для проектов СБИС - не менее 256 MB
Дисковая память	261 MB, в том числе: - Renoir - 51 MB; - ModelSim - 78 MB; - Leonardo Spectrum - 79 MB; - Файлы документации - 3 MB; - Файлы проектов - 50 MB
Видеопамять	2 MB
Монитор	Диагональ - 17”; Количество цветов - 2^{16}; Разрешение - 1024 x 768

Следует отметить, что конфигурация аппаратных средств для САПР определяется сложностью проекта.

2. СИНТЕЗ VHDL-ОПИСАНИЙ ПРОЕКТОВ СБИС В ПОДСИСТЕМЕ RENOIR

2.1. Запуск системы. Оболочка Design Browser

После запуска системы FPGA Advantage на экране монитора появляется оболочка Design Browser, представленная на рис. 1 и содержащая меню команд, кнопки для быстрого вызова команд, окно Sourse со списком каталогов с файлами проектов, полученными в подсистеме Renoir, окно HDL со списком каталогов с файлами VHDL/Verilog-описаний, окно Downstream с закладками ModelSim (список каталогов с исходными, управляющими и результирующими файлами моделирования проектов) и Leonardo (список каталогов с файлами, полученными в результате компиляции), а также строку подсказки (в нижней части экрана).

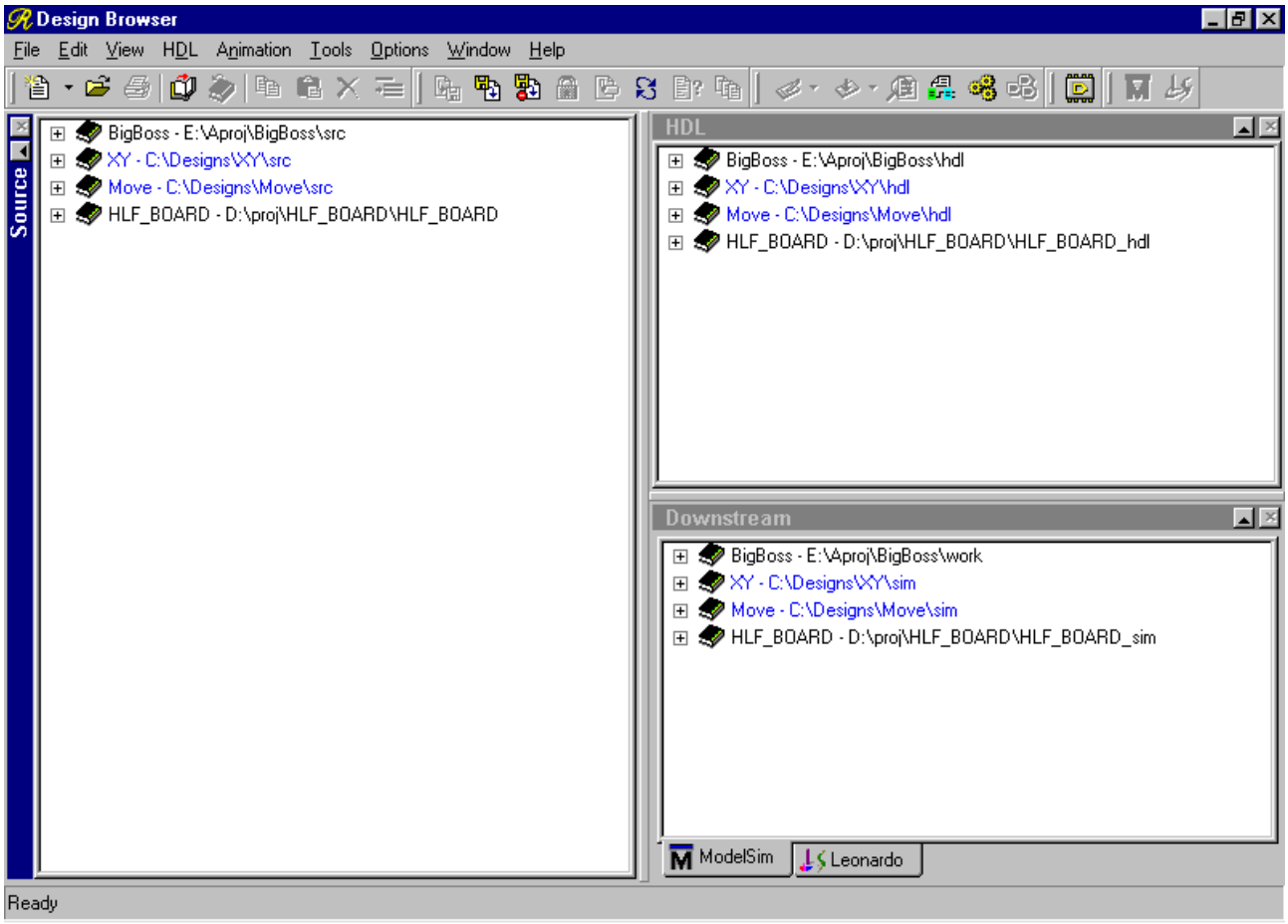


Рис. 1. Оболочка Design Browser

2.2. Создание библиотеки проекта

Прежде всего необходимо создать библиотеку проекта с помощью команды File/New Library главного меню. В появившемся диалоговом окне Add New Library Mapping (рис. 2) необходимо ввести имя библиотеки проекта в поле Library Name (в приведенном примере - FILTR), корневой каталог библиотеки проекта в поле Root Directory (в приведенном примере - D:\proj), установить флажок Open Library after Add (если он не установлен) и нажать кнопку «Advanced» с помощью манипулятора «мышь».

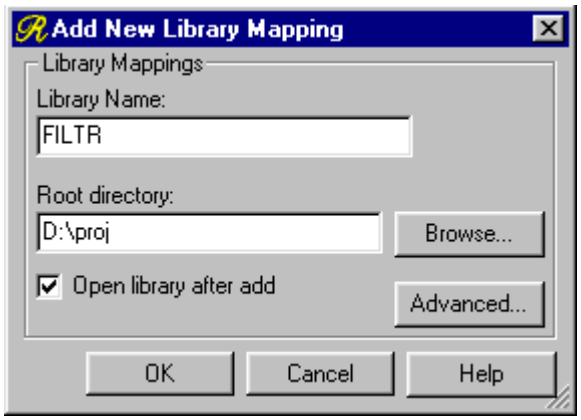


Рис. 2. Диалоговое окно Add New Library Mapping

Следует особо отметить, что все имена объектов проекта, файлов и каталогов должны состоять только из латинских букв и не содержать пробелов. Перед запуском FPGA Advantage необходимо установить в свойстве операционной системы Windows «Язык и стандарты» региональный стандарт «Английский (США)» (или убедиться в его установке). Кроме того, оговоримся, что при дальнейшем изложении под словом «кнопка» будет подразумеваться кнопка на экране монитора, активизируемая при помощи манипулятора «мышь», а под словом «клавиша» - элемент клавиатуры.

После активизации кнопки «Advanced» на экране появится окно Advanced Add Library Mapping (рис. 3), в котором будут автоматически сформированы имя библиотеки в поле Library Name (FILTR), имя каталога с файлами проекта, полученными в подсистеме Renoir, в поле SOURCE - Renoir Design Data Directory (D:\proj\FILTR\src), имя каталога с VHDL/Verilog-файлами проекта в поле HDL - Generated HDL Directory (D:\proj\FILTR\hdl). Необходимо с помощью манипулятора «мышь» скопировать путь к корневому каталогу библиотеки проекта (например, из поля SOURCE - Renoir Design Data Directory без последнего подкаталога \src - D:\proj\FILTR) и вставить его в пустое окошко поля DOWNSTREAM - Compiled Data Directories под заголовком ModelSim, добавив в конце подкаталог \sim, после чего выполнить аналогичные действия, активизировав опускающееся меню справа от заголовка ModelSim, выбрав заголовок Leonardo и добавив в конце подкаталог \leo. После этого нажать кнопку «ОК».

2.3. Открытие библиотеки проекта

Для того, чтобы открыть вновь созданную (или ранее созданную) библиотеку проекта, необходимо воспользоваться командой основного меню File/Open Library или активизировать кнопку  .

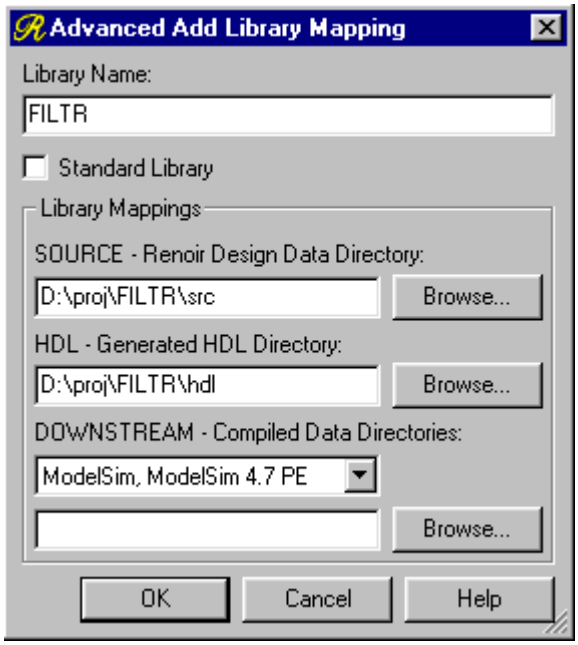


Рис. 3. Диалоговое окно Advanced Add New Library Mapping

В появившемся окне Open Library необходимо выделить имя открываемой библиотеки (в данном примере - FILTR), после чего нажать на кнопку «Open». При этом имена соответствующих каталогов библиотеки проекта появятся в окнах Sourse (FILTR - D:\proj\FILTR\src), HDL (FILTR - D:\proj\FILTR\hdl) и Downstream (FILTR - D:\proj\FILTR\sim на закладке ModelSim и FILTR - D:\proj\FILTR\leo на закладке Leonardo) оболочки Design Browser и будут подсвечены синим цветом, свидетельствующем о том, что данные каталоги пока не содержат файлов проекта. При создании файлов проекта в соответствующих каталогах их имена в оболочке Design Browser изменят цвет на черный.

2.4. Создание файлов проекта в подсистеме Renoir

После того, как библиотека проекта создана и открыта, можно приступать к созданию файлов проекта в подсистеме Renoir с помощью команды основного меню File/New. В опускающемся меню появится список представлений верхнего (в данном случае) уровня иерархии проекта. В принципе, проект может иметь всего один уровень иерархии, но для проектов СБИС это не характерно.

Приведем данный список с краткими пояснениями по каждому пункту:

Block Diagram - структурная схема, представление структурных компонентов в виде условных графических обозначений (УГО), соединенных между собой сигнальными линиями и шинами, передающими многоразрядные сигналы, объединенные общим именем с соответствующей индексацией (рис. 4). Каждый структурный компонент может быть описан любым из приведенных в данном списке способов. Сложность структурной схемы и число уровней иерархии ограничивается возможностями аппаратных средств САПР, в основном, объемом оперативной и дисковой памяти (см. табл. 1);

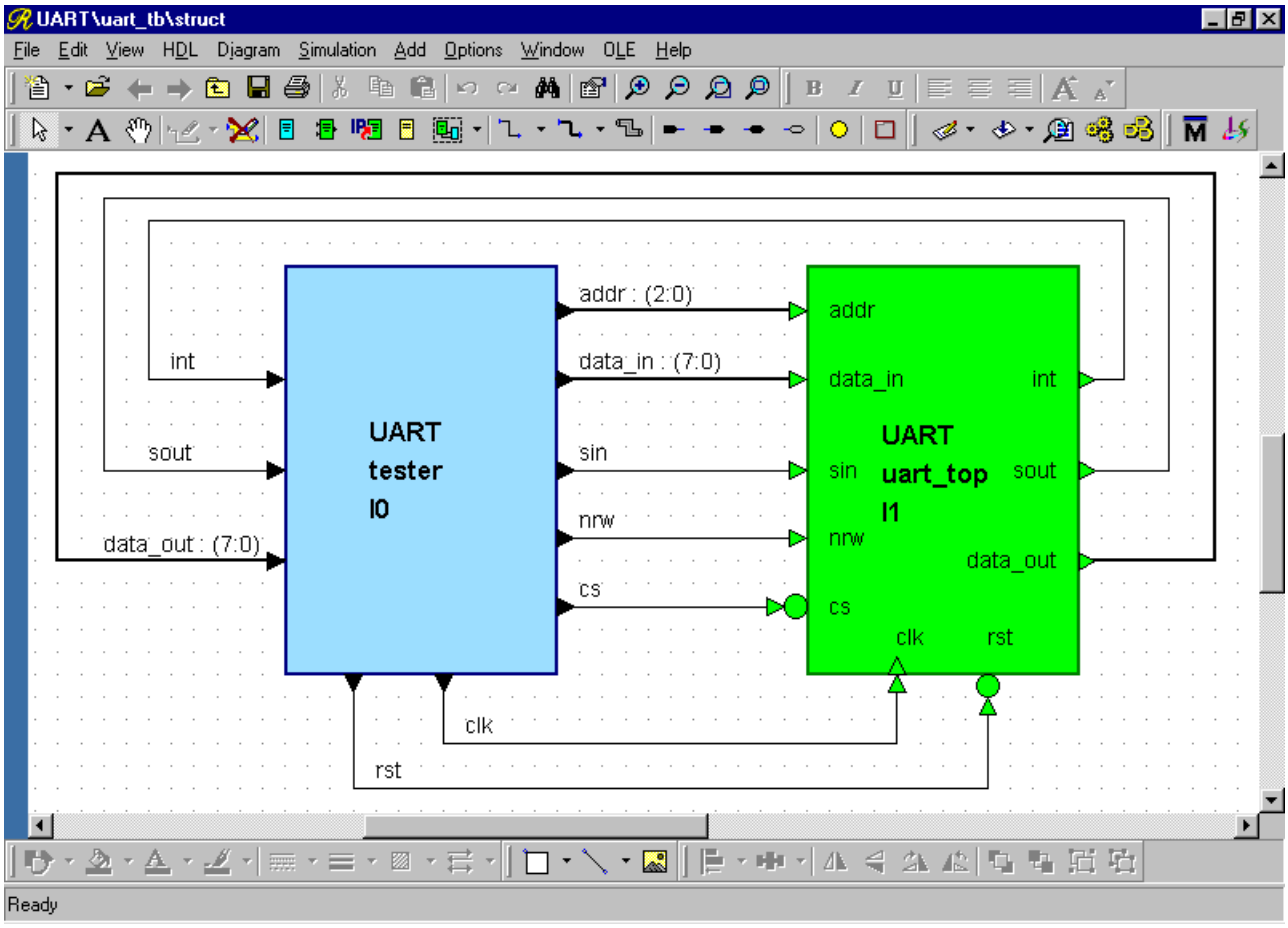


Рис. 4. Пример структурного представления Block Diagram

Flow Chart - графическое представление алгоритма функционирования проекта в виде блок-схемы (рис. 5). Каждый из блоков представлен соответствующим VHDL-описанием;

State Diagram - представление конечных автоматов - функциональное представление в виде направленного графа (рис. 6), вершины которого (круги) определяют различные состояния системы (текущие значения всех сигналов и переменных, записанные с помощью VHDL-назначений), а ребра (линии, соединяющие круги) - переходы между состояниями. Переходы могут быть условными или безусловными и сопровождаться выполнением определенных функций обработки данных.

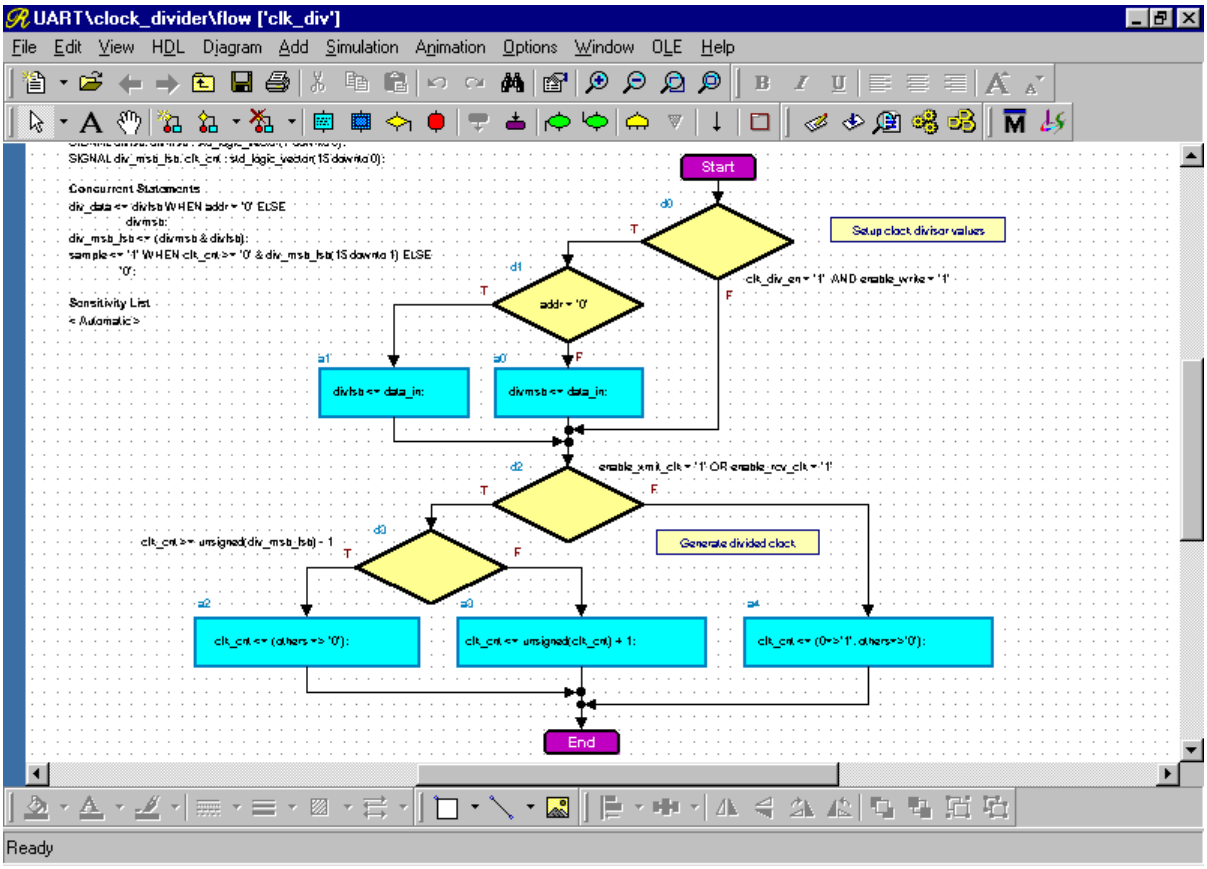


Рис. 5. Пример функционального представления Flow Chart

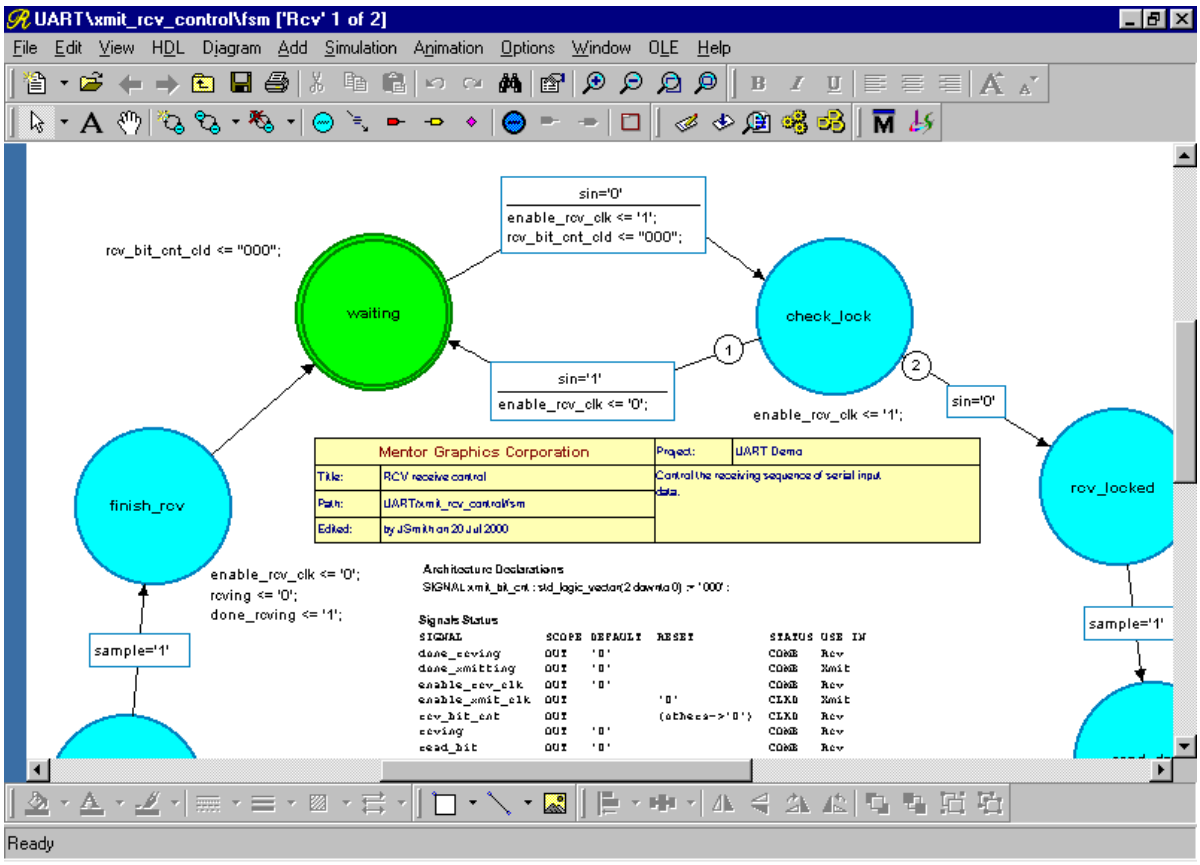


Рис. 6. Пример функционального представления State Diagram

Эти условия и функции представлены VHDL-описаниями, помещенными в прямоугольники, относящимися к соответствующим ребрам. Очередность проверки условий для различных переходов из одного состояния определяется уровнем приоритета, задаваемым разработчиком и выводимым в виде числа в маленьком круге, расположенном на каждом ребре. При составлении State Diagram допускаются петли, т.е. переходы из текущего состояния в него же, сопровождающиеся определенными действиями при определенных условиях. Состояния могут быть иерархическими, т.е. текущее состояние может быть представлено, в свою очередь, конечным автоматом более низкго уровня иерархии. Работа конечного автомата синхронизируются тактовыми импульсами (например, Clock). При этом важно помнить, что при входе в состояние по фронту сигнала Clock выполняются действия, относящиеся к переходу, а при выходе - действия, относящиеся к состоянию, из которого осуществляется выход;

Symbol - создание условного графического обозначения компонента проекта (рис. 7).

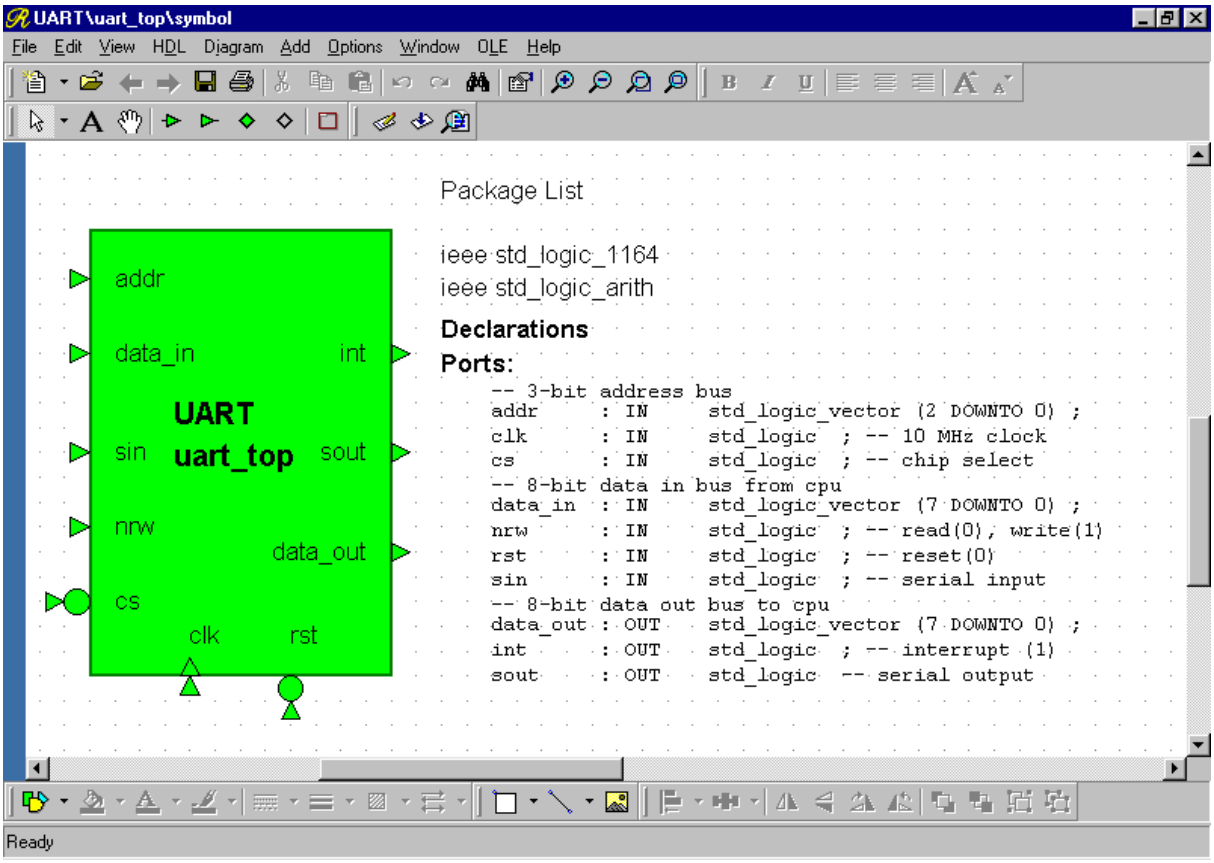


Рис. 7. Пример условного графического обозначения Symbol

При активизации данной команды появляется окно редактора с прямоугольным УГО без внешних выводов (портов). Условные обозначения портов расставляются по периметру УГО с помощью манипулятора «мышь» при активизации следующих команд меню: Add/Input Port - добавление входного порта, Add/Output Port - добавление выходного порта, Add/InOut Port - добавление двунаправленного порта. Фиксация обозначения порта на периметре УГО осу-

ществляется щелчком левой кнопки «мыши». Для ввода параметров порта необходимо дважды щелкнуть левой кнопкой «мыши» на его обозначении и в появившемся диалоговом окне ввести имя порта в поле Name, тип сигнала в поле Type (можно выбрать из списка, щелкнув левой кнопкой «мыши» на стрелке справа от данного поля), способ задания диапазона допустимых значений в поле Constraint (Index - диапазон индексов линий шины, Range - диапазон значений сигнала, None - для однобитных сигналов), диапазон индексов шины или значений сигнала в поле Bounds, после чего последовательно нажать кнопки «Apply» и «OK». Если необходимо ввести параметры данного компонента (generic), необходимо навести курсор «мыши» на УГО, щелкнуть правой кнопкой и активизировать команду контекстного меню Object Properties. В появившемся диалоговом окне (рис. 8) в закладке Generic Declarations следует ввести имя, тип и значение параметра в трех полях в нижней части окна под заголовками Name, Type и Value соответственно, после чего нажать кнопку «Add».

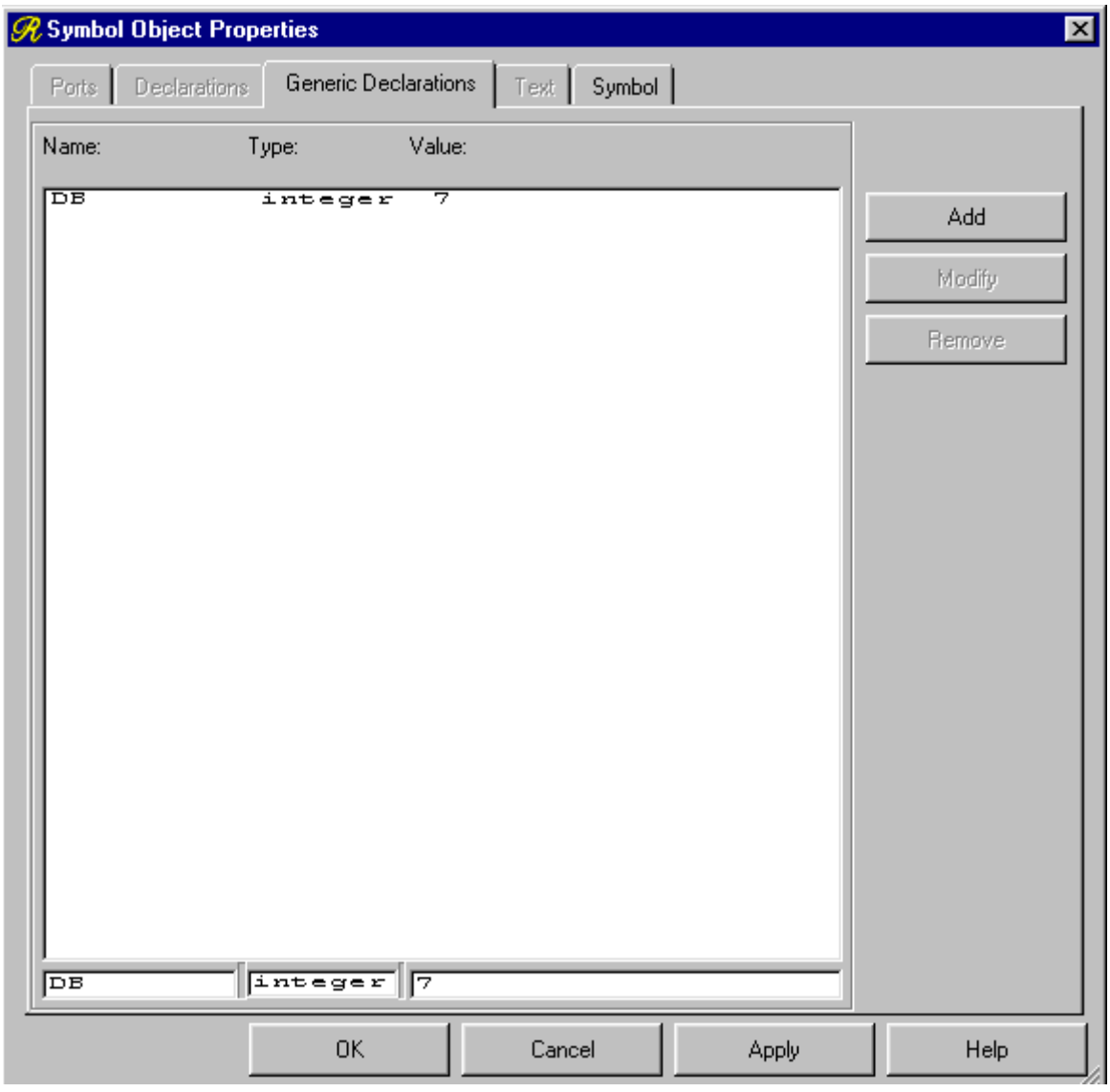
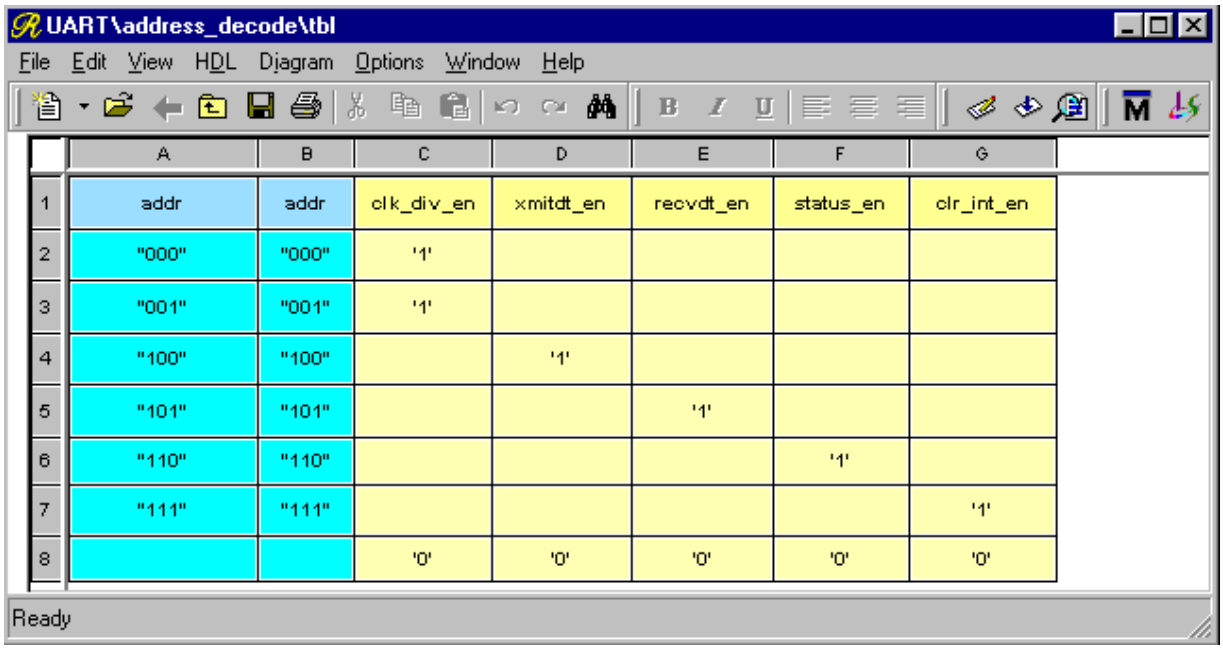


Рис. 8. Диалоговое окно Symbol Object Properties

При этом введенная информация появится в верхней части основного поля диалогового окна. При необходимости изменить введенные данные следует в основном окне выделить с помощью «мыши» строку с требуемым параметром и в нижних полях ввести нужные исправления, после чего нажать кнопку «Modify». Кнопка «Remove» служит для удаления выделенной строки из списка в основном поле. После ввода всех портов и параметров необходимо сохранить УГО с помощью команды File/Save. Следует отметить, что при первом сохранении появится диалоговое окно, в котором будет необходимо указать библиотеку, в которой будет храниться данный компонент, а также его имя. Для ввода содержимого нового компонента следует закрыть окно редактирования УГО, перейти в оболочку Design Browser, раскрыть содержимое библиотеки, в которой сохранен компонент, щелчком левой кнопки «мыши» на значке «+» слева от имени библиотеки в поле Source и дважды щелкнуть левой кнопкой на имени введенного компонента. В появившемся диалоговом окне необходимо выбрать способ ввода содержимого компонента (структурная схема, блок-схема, конечный автомат, таблица истинности, VHDL-описание), после чего осуществляется ввод информации;

Truth Table - представление в виде таблицы истинности (рис. 9);



	A	B	C	D	E	F	G
1	addr	addr	clk_div_en	xmitdt_en	recvdt_en	status_en	clr_int_en
2	"000"	"000"	'1'				
3	"001"	"001"	'1'				
4	"100"	"100"		'1'			
5	"101"	"101"			'1'		
6	"110"	"110"				'1'	
7	"111"	"111"					'1'
8			'0'	'0'	'0'	'0'	'0'

Рис. 9. Пример таблицы истинности Truth Table

Verilog Include, Verilog Module - описания на языке Verilog;

VHDL Architecture/Entity - VHDL-описание, включающее два обязательных модуля проекта: объявление объекта проекта Entity и архитектурное тело Architecture Body, т.е. определяющее как интерфейс (внешние порты) проекта, так и выполняемые функции. Кроме объявления объекта проекта и архитектурного тела, в VHDL-описании могут использоваться еще три модуля проекта: объявление конфигурации, объявление пакета и тело пакета [2, 3]. Объявление конфигурации (Configuration Declaration) применяется для задания объектов,

использованных при создании проекта. В языке VHDL предусмотрен механизм пакетов для часто используемых описаний, констант, типов, сигналов. Эти описания помещаются в объявлении пакета (Package Declaration). Если пользователь осуществляет нестандартные операции или функции, их интерфейсы описываются в объявлении пакета, а тела содержатся в теле пакета (Package Body);

VHDL Entity only - VHDL-описание, включающее только объявление объекта проекта, т.е. определяющее интерфейс;

VHDL External Package - VHDL-описание, предусматривающее подключение пакета из внешней библиотеки;

VHDL Package Body - VHDL-описание тела пакета;

VHDL Package Header - объявление пакета;

Plain Text File - создание текстового файла.

Файл верхнего уровня иерархии проекта СБИС, как правило, наиболее целесообразно создавать в виде структурной схемы с помощью команды File/New/Block Diagram, при вызове которой появляется окно редактора с соответствующим меню команд (см. рис. 4). Помимо традиционных команд работы с файлами и их содержимым (File/New - «Создать новый файл», File/Open - «Открыть файл», File/Save - «Сохранить файл», File/Close Window - «Закрыть», File/Exit - «Выйти из редактора», Edit/Undo - «Отменить последнее действие», Edit/Redo - «Выполнить последнее отмененное действие», Edit/Cut - «Вырезать выделенный фрагмент», Edit/Copy - «Скопировать выделенный фрагмент», Edit/Past - «Вставить скопированный фрагмент», Edit/Delete - «Удалить выделенный фрагмент», Edit/Select All - «Выделить все содержимое файла», Edit/Find - «Найти фрагменты по ключевому признаку», Edit/Replace - «Заменить найденные по ключевому признаку фрагменты на другие» и др.) меню содержит ряд специфических команд, на которых следует остановиться несколько подробнее:

Add/Block - добавление элемента структурной схемы, называемого «Блок», выполняющего определенную функцию обработки или хранения информации, которая может быть задана, в свою очередь, структурной схемой, блок-схемой алгоритма, конечным автоматом, таблицей истинности или непосредственно VHDL-описанием, содержащим как объявление объекта проекта (модуль Entity), так и архитектурное тело (модуль Architecture). После активизации данной команды при наведении курсора «мыши» на поле редактора появляется контур УГО нового блока, который перемещается вместе с курсором. Положение УГО фиксируется нажатием левой кнопки «мыши». Отмена команды осуществляется нажатием правой кнопки «мыши». После добавления блока и сохранения файлов проекта в окне Source оболочки Design Browser в списке объектов соответствующей библиотеки проекта появится ссылка на новый блок с указанным именем;

Add/Component - добавление элемента структурной схемы, называемого «Компонент». Отличие «Компонента» от «Блока» состоит в том, что компонент имеет собственное УГО (Symbol), сохраненное в определенной библиотеке проекта в виде соответствующего текстового файла с расширением *.sb, что

позволяет непосредственно и многократно использовать компоненты как из библиотеки данного проекта, так и из других доступных библиотек. В отличие от команды добавления блока, предусматривающей добавление УГО нового блока, а затем определение его содержимого, при активизации команды добавления компонента вначале появляется диалоговое окно Add Component с закладкой Renoir, содержащее три поля (рис. 10). Необходимо выбрать из списка в поле Library имя библиотеки, содержащей требуемый компонент, навести на него курсор «мыши» и щелкнуть левой кнопкой. При этом в поле Design Unit появится список компонентов данной библиотеки. Необходимо с помощью «мыши» аналогично выбрать имя компонента и нажать на кнопку «ОК». В результате на поле редактора появляется контур УГО компонента, который перемещается вместе с курсором. Положение УГО фиксируется нажатием левой кнопки «мыши». Отмена команды осуществляется нажатием правой кнопки «мыши». Таким образом, добавить можно лишь предварительно созданный компонент;

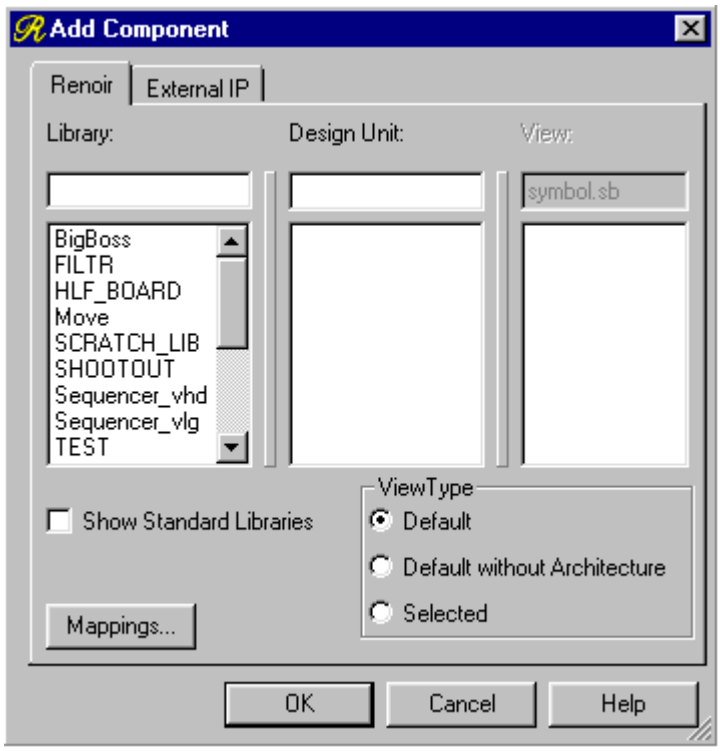


Рис. 10. Диалоговое окно Add Component с закладкой Renoir

Add/IP - добавление внешнего сложного функционального (СФ) блока (IP-core). При активизации данной команды появляется диалоговое окно Add Component с закладкой External IP, показанное на рис. 11.

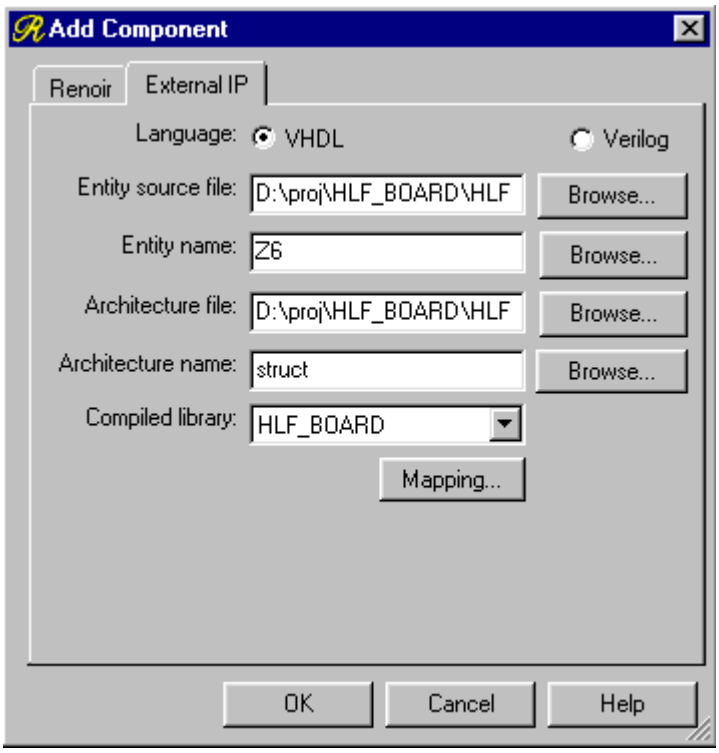


Рис. 11. Диалоговое окно Add Component с закладкой External IP

Необходимо указать путь к файлу VHDL-описания СФ-блока (*.vhd) в поле Entity source file и имя библиотеки в поле Compiled library (при этом остальные поля будут заполнены автоматически) и нажать «ОК». В результате на поле редактора появляется контур УГО СФ-блока, который перемещается и фиксируется аналогично блоку и компоненту. Следует отметить, что в данном случае, в отличие от компонента, файл УГО отсутствует;

Add/Embedded Block - добавление так называемого встроенного блока, который не может быть самостоятельной единицей проекта, так как представляет собой описание лишь отдельных функций, входящих в архитектурное тело. Иными словами, VHDL-код, соответствующий встроенному блоку не содержит ни модуля Entity, ни модуля Architecture. Поэтому, в отличие от блока, содержимое встроенного блока может быть задано одним из четырех способов: в виде блок-схемы алгоритма (Flow Chart), в виде конечного автомата (State Diagram), в виде таблицы истинности (Truth Table) и в виде текстового VHDL-описания (Text). После активизации данной команды при наведении курсора «мыши» на поле редактора появляется контур УГО нового встроенного блока, который перемещается и фиксируется аналогично блоку и компоненту;

Add/Frame - добавление рамки условного синтеза. Если в проекте предусмотрены параметры (Generic), отдельные объекты структурной схемы проекта могут охватываться этими рамками и компилироваться в соответствующие топологические файлы и файлы для моделирования только при выполнении определенных условий, которые задаются пользователем и отражаются в текстовом виде над рамкой. На рис. 12 показан пример использования рамки условного синтеза IF;

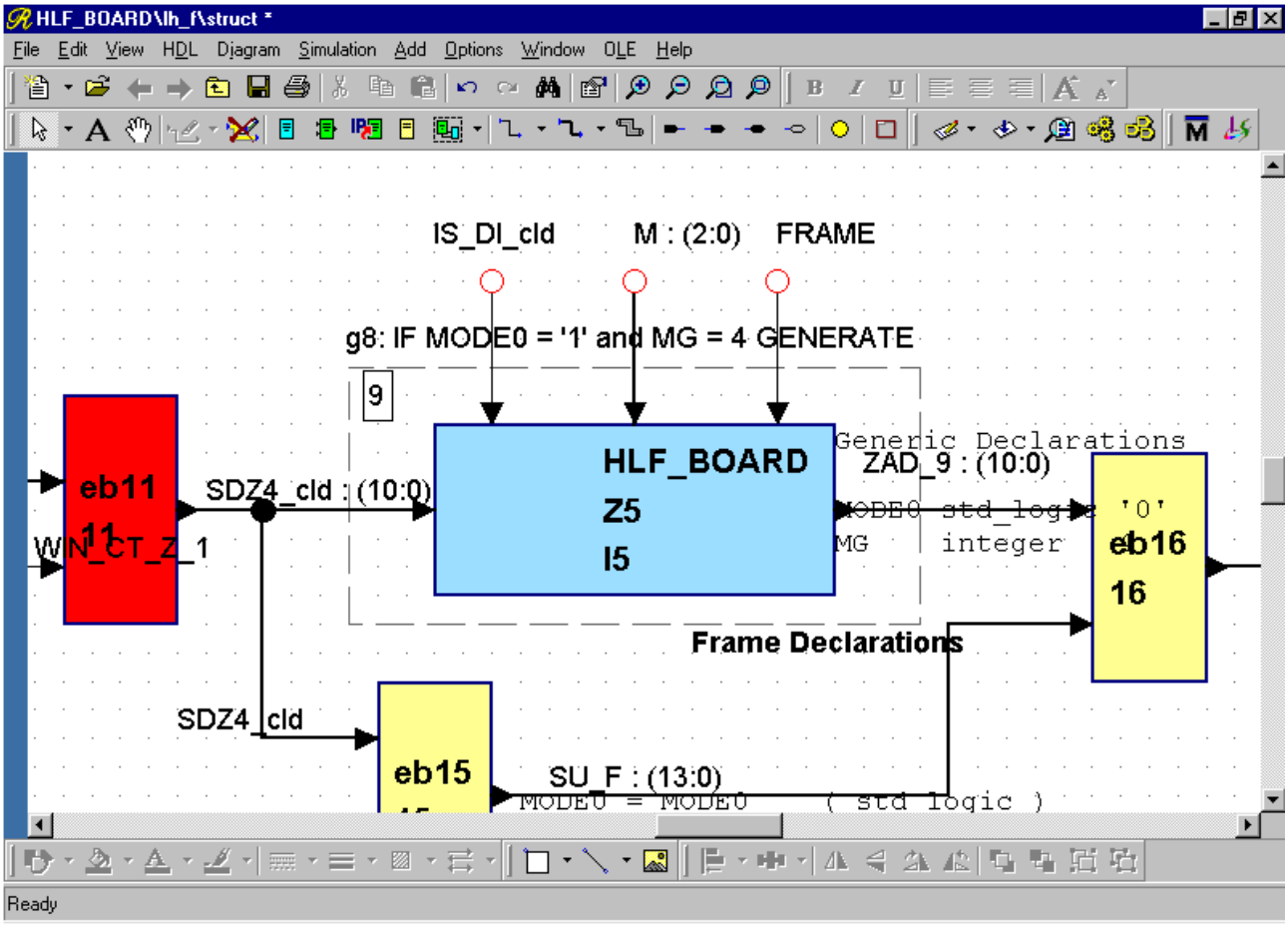


Рис. 12. Пример использования рамки условного синтеза

Add/Signal - добавление сигнала. После активизации данной команды следует навести курсор «мыши» в точку начала сигнальной линии, щелкнуть левой кнопкой и вести курсор до точки предполагаемого окончания линии, щелкая левой кнопкой при необходимости поворота линии. По достижении конечной точки сигнальной линии необходимо один раз щелкнуть левой кнопкой «мыши», если конечная точка является изображением входа элемента структурной схемы, и дважды щелкнуть левой кнопкой в противном случае (при этом на конце сигнальной линии появится красный кружок). Затем необходимо нажать правую кнопку «мыши», навести курсор на изображение линии и дважды щелкнуть левой кнопкой, после чего в диалоговом окне ввести имя сигнала (в поле Name), тип сигнала (в поле Type, например, std_logic) и последовательно нажать кнопки «Apply» и «ОК». Если указанное имя сигнала уже используется в проекте, после нажатия кнопки «Apply» будет выдано соответствующее предупреждение. Пользователь должен подтвердить или отменить указанное назначение. Следует отметить, что в данном редакторе линии сигналов не должны обязательно начинаться и заканчиваться на изображениях выходов и входов элементов структурной схемы. В этом случае соединение инициализируется согласованием имен отдельных отрезков линии данного сигнала. То же относится и к шинам, которые будут описаны ниже;

Add/Bus - добавление шины (нескольких сигналов, имеющих общее имя с соответствующей индексацией). Все необходимые действия аналогичны добавлению сигнала. Но при идентификации шины необходимо, кроме имени и типа, указать также разрядность в поле Bounds диалогового окна (например, 7 downto 0 для типа std_logic_vector);

Add/Global Connector - добавление глобального сигнала (объявленного и используемого во всех блоках проекта, например, тактового сигнала). После фиксации УГО, вызываемого данной командой (круг желтого цвета), сигнал, линия которого будет с ним соединена, станет глобальным;

Add/Port In - добавление входного внешнего порта;

Add/Port Out - добавление выходного внешнего порта;

Add/Port InOut - добавление двунаправленного внешнего порта.

Для наполнения добавленных блоков и встроенных блоков или редактирования содержимого компонентов необходимо дважды щелкнуть левой кнопкой «мыши», наведя курсор на соответствующее условное графическое изображение.

При описании проекта в виде блок-схемы алгоритма функционирования (команда File/New/Flow Chart) на экране монитора появляется окно редактора Renoir с соответствующим набором команд меню Add:

Add/Start Point - добавление начального элемента блок-схемы;

Add/End Point - добавление конечного элемента блок-схемы;

Add/Action Box - добавление функционального блока;

Add/Decision Box - добавление условного перехода;

Add/Case - добавление условного перехода по набору значений переменной, сигнала, константы или параметра, стоящего в условии;

Add/Wait Box - добавление блока ожидания до выполнения определенного условия, заданного пользователем;

Add/Loop - добавление блока начала тела цикла;

Add/End Loop - добавление блока окончания тела цикла;

Add/Flow - добавление направленной линии, соединяющей блоки;

Add/Hierarchical Action Box - добавление иерархического функционального блока, содержимое которого представляется отдельной вложенной блок-схемой.

Для наполнения добавленных блоков необходимо дважды щелкнуть левой кнопкой «мыши», наведя курсор на изображение блока.

При описании проекта в виде конечного автомата (команда File/New/State Diagram) на экране монитора появляется окно редактора Renoir со следующим набором команд меню Add:

Add/State - добавление нового состояния, отображаемого на экране в виде круга с именем, определяемым пользователем или по умолчанию. Круг, соответствующий начальному состоянию, по умолчанию выделяется зеленым цветом с двойным контуром. Остальные - голубым цветом;

Add/Transition - добавление перехода между состояниями. Точки плавного изгиба линии перехода фиксируются щелчками левой кнопки «мыши»;

Add/Hierarchical State - добавление иерархического состояния, содержимое которого определяется вложенной диаграммой состояний;

Add/Entry Point - добавление точки входа в диаграмму состояний (только для вложенных диаграмм);

Add/Exit Point - добавление точки выхода из диаграммы состояний (только для вложенных диаграмм).

Определение содержания добавленных состояний и переходов осуществляется после двойного щелчка левой кнопкой «мыши» на изображении состояния или линии перехода.

При описании проекта в виде таблицы истинности (команда File/New/Truth Table) на экране монитора появляется окно редактора Renoir со следующим набором команд Add, которые не вынесены в данном случае в главное меню, а доступны лишь в контекстном меню, появляющемся при нажатии правой кнопки «мыши» на выбранной клетке таблицы:

Add/Column - добавление нового столбца таблицы истинности;

Add/Row - добавление новой строки таблицы истинности.

Имеются также команды Delete Column, Delete Row для удаления столбца или строки, на пересечении которых находится выделенная ячейка таблицы. Столбцы входов подсвечены голубым, а выходов - желтым цветом. Для заполнения клеток таблицы необходимо установить курсор «мыши» на нужную клетку, щелкнуть левой кнопкой, после чего вводить необходимые данные (имена и значения входных и выходных сигналов) с клавиатуры.

Описание содержимого каждого из введенных элементов структурной схемы проекта может, в свою очередь, быть выполнено любым из выше перечисленных способов, что позволяет создавать иерархические описания любого уровня сложности (определяемого аппаратными ресурсами САПР).

Ввод и редактирование параметров объектов проекта (блоков, компонентов, встроенных блоков и др.) можно осуществить, наведя курсор «мыши» на УГО объекта, щелкнув правой кнопкой и выбрав в контекстном меню команду Object Properties. Команда Reconcile Interface из этого же меню позволяет автоматически согласовать различия в описаниях интерфейсов (внешних выводов) объектов на текущем и нижнем уровнях иерархии.

2.5. Подключение библиотек ресурсов в подсистеме Renoir

Помимо рабочей библиотеки, в которой размещаются все файлы проекта, VHDL предусматривает возможность использования ссылок на объекты из внешних библиотек (так называемых библиотек ресурсов) [2].

Подключение и отключение библиотек ресурсов в подсистеме Renoir осуществляется с помощью команды меню Diagram/Package References, при активизации которой выводится соответствующее диалоговое окно (рис. 13).

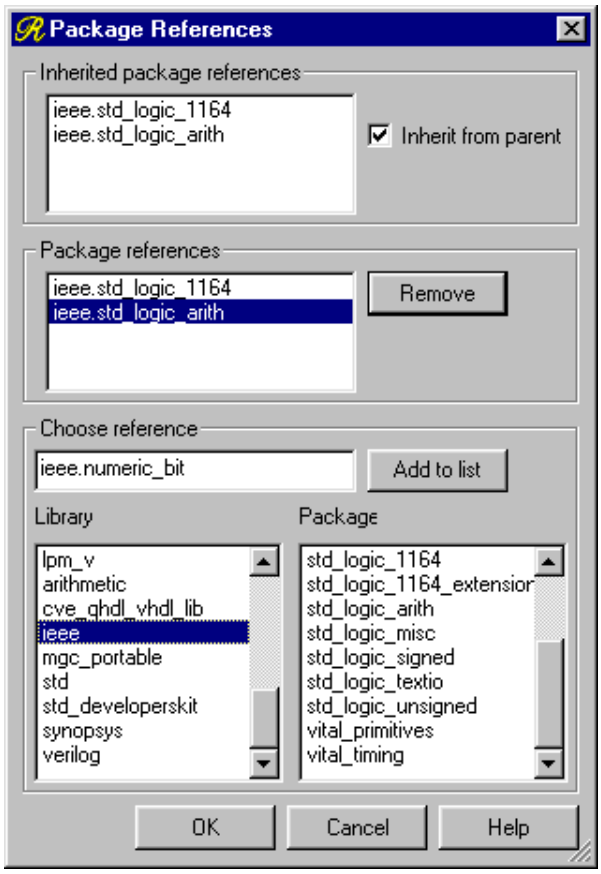


Рис. 13. Диалоговое окно подключения библиотек ресурсов

В верхнем поле Inherent Package References отображается список пакетов, «унаследованных» от объектов верхних уровней иерархии. Наследование может быть отменено при удалении значка справа от данного поля (см. рис. 13).

В поле Package References отображается список подключенных в данный момент пакетов. Удаление отдельных пакетов из данного списка выполняется при помощи кнопки «Remove» справа от поля списка.

Для добавления к списку нового пакета необходимо выделить требуемую библиотеку из списка в поле Library, выделить пакет из списка в поле Package и нажать кнопку «Add to list». При этом имя пакета появится в списке Package References, после чего необходимо нажать кнопку «ОК».

2.6. Генерация файлов VHDL-описаний в подсистеме Renoir

Следует отметить, что каким бы из вышеперечисленных способов не было задано содержимое блоков и компонентов проекта, в конечном итоге, после автоматической генерации, проект будет представлен соответствующими VHDL-файлами.

Генерация VHDL-описаний может быть выполнена на любом уровне иерархии проекта, причем как для текущего, так и для всех нижележащих уровней. Вне зависимости от способа описания проекта в подсистеме Renoir, генерация описаний осуществляется с помощью одной из следующих команд меню:

HDL/Generate/Single Level - генерация VHDL-файла только текущего уровня описания проекта;

HDL/Generate/Hierarchy - генерация VHDL-описаний проекта с учетом иерархии блоков;

HDL/Generate/Hierarchy Through Components - полная генерация VHDL-описаний проекта с учетом иерархии блоков и содержимого компонентов.

При успешном выполнении генерации, т.е. при отсутствии грамматических и синтаксических ошибок, правильном с точки зрения стандарта VHDL описании всех модулей проекта, в окне Log Window появится сообщение: «Generation completed successfully». В противном случае будут выведены сообщения об ошибках. Причем можно быстро перейти к структурному элементу или строке VHDL-описания, содержащим ошибки, выделив конкретное сообщение об ошибке и щелкнув левой кнопкой «мыши» на одной из кнопок  .

3. КОМПИЛЯЦИЯ VHDL-ОПИСАНИЙ В ПОДСИСТЕМЕ RENOIR

После успешного завершения генерации VHDL-файлов необходимо устранить все ошибки, связанные не с нарушением грамматики или синтаксиса, а с некорректным с точки зрения технического задания описанием выполняемых устройством функций. С этой целью выполняют функционально-логическое моделирование проекта, для выполнения которого необходимо провести компиляцию VHDL-описаний, в результате которой будут автоматически выявлены некоторые ошибки и после их устранения созданы файлы в специальном формате для подсистемы ModelSim.

Компиляция VHDL-описаний осуществляется с помощью следующих команд:

HDL/Compile/Single Level - компиляция VHDL-файла только текущего уровня описания проекта;

HDL/Compile/Hierarchy - компиляция VHDL-описаний проекта с учетом иерархии блоков;

HDL/Compile/Hierarchy Through Components - полная компиляция VHDL-описаний проекта с учетом иерархии блоков и содержимого компонентов.

При успешном выполнении компиляции в окне Log Window появится сообщение: «Data preparation step completed, check transcript...». В противном случае будут выведены сообщения об ошибках.

После компиляции VHDL-описаний можно осуществить функционально-логическое моделирование проекта в подсистеме ModelSim, нажав кнопку  на верхней панели.

4. СИНТЕЗ ФАЙЛОВ ТОПОЛОГИИ

Синтез топологии проектируемой СБИС осуществляется в отдельной САПР, выбираемой в зависимости от метода реализации СБИС (например, САПР MAX+plus II для реализации на ПЛИС фирмы Altera, САПР Tanner Pro для реализации в виде заказной СБИС и т.д.). Подготовка файлов стандартных форматов описания структуры (например, файлы формата EDIF) и задержек в элементах и линиях связи (файлы формата SDF) осуществляется в подсистеме Leonardo Spectrum. Для выполнения этих операций необходимо нажать кнопку  на верхней панели.

5. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

1. Получить у преподавателя вариант задания.
2. Ознакомиться с общей организацией САПР FPGA Advantage по данному методическому пособию.
3. Изучить маршруты проектирования СБИС в САПР FPGA Advantage по данному методическому пособию.
4. Выбрать один из маршрутов проектирования для выполнения задания и согласовать его с преподавателем.
5. Проанализировать набор реализуемых функций, составить список используемых библиотек ресурсов и согласовать его с преподавателем.
6. Запустить САПР FPGA Advantage и ввести описание проекта в соответствии с заданием и согласованным с преподавателем маршрутом.
7. Выполнить генерацию VHDL-файлов проекта. Исправить возможные ошибки.
8. Выполнить компиляцию VHDL-файлов проекта. Исправить возможные ошибки.
9. Показать преподавателю результаты выполнения задания и ответить на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение САПР FPGA Advantage.
2. Основные подсистемы САПР FPGA Advantage и их назначение.
3. Назначение и интерфейс оболочки Design Browser.
4. Состав и назначение библиотеки проекта и библиотек ресурсов.
5. Маршрут подключения библиотек ресурсов.
6. Понятия «Блок», «Компонент», «Встроенный блок».
7. Понятия «Сигнал», «Шина», «Порт».
8. Способы описания объектов проекта.
9. Особенности ввода структурной схемы.
10. Особенности ввода блок-схемы.

11. Особенности ввода конечного автомата.
12. Особенности ввода таблицы истинности.
13. Маршрут создания компонента.
14. Особенности подключения внешних СФ-блоков.
15. Назначение параметров компонентов.
16. Маршрут ввода и редактирования параметров компонентов.
17. Назначение и маршрут ввода рамки условного синтеза.
18. Назначение процедуры генерации VHDL-файлов.
19. Назначение процедуры компиляции VHDL-файлов.
20. Особенности синтеза топологии в САПР FPGA Advantage.

ВАРИАНТЫ ЗАДАНИЙ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Разработать линию задержки входного сигнала DI типа std_logic_vector(M downto 0) на L периодов тактового сигнала CLK (активный фронт передний). Сброс регистров линии задержки в нулевое состояние, асинхронный по активному уровню сигнала RESET = ‘1’. Входные данные сопровождаются сигналом подсвета IS_DI, при пассивном уровне которого (IS_DI = ‘0’, RESET = ‘0’) все регистры линии задержки сохраняют текущие значения. Параметры L и M задаются преподавателем и могут быть описаны как параметры синтеза.
2. Разработать конвейерный сумматор N сигналов типа std_logic_vector(M downto 0). Тактовый сигнал для регистров конвейера CLK (активный фронт передний). Сброс регистров в нулевое состояние, асинхронный по активному уровню сигнала RESET = ‘1’. Входные данные сопровождаются сигналом подсвета IS_DI, при пассивном уровне которого (IS_DI = ‘0’, RESET = ‘0’) все регистры обнуляются по переднему фронту сигнала CLK. Параметры N и M задаются преподавателем и могут быть описаны как параметры синтеза.
3. Разработать накапливающий сумматор для N последовательно поступивших данных по шине DI типа std_logic_vector(M downto 0). Тактовый сигнал CLK (активный фронт передний). Данные выдаются по переднему фронту сигнала CLK сплошным потоком или с промежутками произвольной длительности. Сброс регистра суммы в нулевое состояние, асинхронный по активному уровню сигнала RESET = ‘1’. Входные данные сопровождаются сигналом подсвета IS_DI. Параметры N и M задаются преподавателем и могут быть описаны как параметры синтеза.
4. Разработать счетчик-формирователь сигнала WIN, управляющего мультиплексором двух сигналов DI_1, DI_2 типа std_logic_vector(M downto 0) на выходную шину DO. Данные по шинам DI_1, DI_2 выдаются по переднему фронту тактового сигнала CLK сплошным потоком или с промежутками произвольной длительности из 2 блоков внешней памяти (в задание не входит) с матричной организацией (NK строк, MK столбцов) построчно. Сигнал WIN принимает активный уровень WIN = ‘1’ в диапазоне номеров строк от

N_MIN до N_MAX и столбцов от M_MIN до M_MAX. При WIN = ‘0’ на выход DO выдается сигнал DI_1, в противном случае - DI_2. Параметры M, NK, MK, N_MIN, N_MAX, M_MIN, M_MAX задаются преподавателем и могут быть описаны как параметры синтеза.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Рындин Е.А. Проектирование специализированных СБИС. Конспект лекций. Таганрог: Изд-во ТРТУ, 1999. 115 с.
2. Коноплев Б.Г., Рындин Е.А., Ивченко В.Г. Описание проектов СБИС с использованием языка VHDL. Методическое руководство. Таганрог: Изд-во ТРТУ, 1998. 28 с.
3. Mentor Graphics FPGA Advantage Documentation Bookcase, 2000.

СОДЕРЖАНИЕ