

2 СИСТЕМА ПРОЕКТИРОВАНИЯ *QUARTUS II*

2.1 Общее описание САПР Quartus II

Программа Quartus II относится к средствам автоматического проектирования (САПР, EDA — Electronic Design Automation), предназначенным для работы с ПЛИС фирмы Altera. САПР Quartus II разработано фирмой Altera и поддерживает весь цикл проектирования цифровых устройств на основе программируемой и реконфигурируемой логики. До появления САПР Quartus разработчики использовали САПР MAX+PLUS II, также разработанный фирмой Altera. САПР MAX+PLUS II является более простым в освоении по сравнению с Quartus II. В настоящее время этот САПР фирма Altera не развивает и рекомендует переходить на систему Quartus II. САПР Quartus II является основной системой проектирования, которую фирма Altera активно развивает. САПР поддерживает все новые семейства микросхем и обладает особенностями, которых нет в MAX+PLUS II. Следует сказать пару слов о фирме Altera. Фирма Altera была основана в 1983 г [2]. На сегодняшний день Altera является одной из крупнейших корпораций, занимающейся разработкой программируемых логических интегральных схем. Основными изделиями являются ПЛИС. Также фирма предоставляет услуги по преобразованию проектов под ПЛИС в ASIC (Application-Specific Integrated Circuit — специализированные заказные интегральные схемы) для массового производства. Компания выпускает специальное программное обеспечение, куда входят средства разработки проектов для ПЛИС, а также компиляторы под ядра процессоров собственной разработки [2].

Quartus II содержит в своем составе определенные средства. Для разработки устройств имеется специальный графический схемный редактор, а также текстовый редактор для ввода описания схемы на языках VHDL, Verilog HDL, Altera HDL. Имеется компилятор и специальный редактор для размещения уже разработанной схемы на логику целевого устройства. В Quartus II входят средства анализа

временных характеристик устройства, таких как время задержки сигнала между входом и выходом и максимальная тактовая частота работы устройства, редактор временных диаграмм для тестирования и отладки разрабатываемого устройства, программатор для переноса конфигурации устройства из проекта в интегральные схемы (ИС). А также в Quartus II имеются дополнительные программные модули для выполнения специальных настроек проекта и целевого устройства.

Аналогом и конкурентом САПР Quartus II является САПР ISE WebPACK фирмы Xilinx. Существует бесплатная версия Quartus II — Quartus II Web Edition, которая несколько функционально ограничена и может использоваться для знакомства с программой, а также в академических целях и для научных исследований. Средства, предоставляемые этим САПР достаточны для проведения лабораторных работ. Эту версию САПР мы и будем использовать для выполнения лабораторных работ.

На рисунке 2.1 представлена общая структурная схема проектирования в САПР Quartus II.

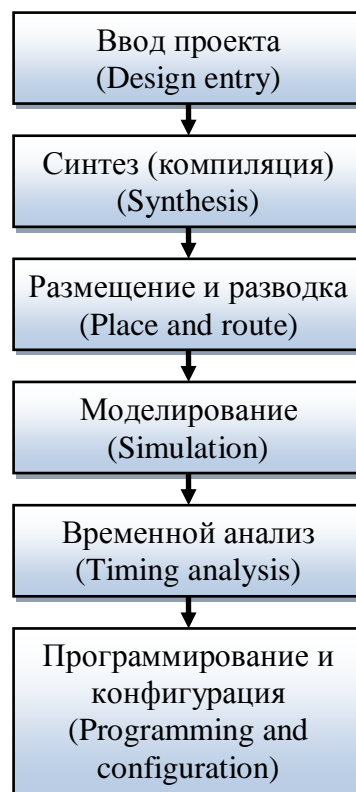


Рисунок 2.1 — Общая структурная схема проектирования в Quartus II

Под термином «проект» в рамках САПР Quartus II понимается набор файлов, определяющих проектируемое устройство, и из этого набора файлов выделяют две группы файлов:

- ✓ логические файлы, описывающие алгоритм работы устройства (Design Files);
- ✓ вспомогательные файлы (Ancillary Files).

Проект может содержать один логический файл либо несколько логических файлов, образующих иерархическое описание проектируемого модуля. При иерархическом описании среди множества логических файлов различают:

- ✓ файл верхнего уровня в иерархии описаний (Top-level Design File);
- ✓ файлы нижних (одного или нескольких) уровней иерархии (Low-level Design Files).

В файле верхнего уровня задаётся архитектура модуля, определяется набор модулей, входящих в его состав как компоненты, и их взаимосвязь. Описания этих модулей содержатся в логических файлах более низкого уровня иерархии. В их

состав, в свою очередь, в виде компонентов также могут входить модули, описания которых приведены в логических файлах еще более низкого уровня иерархии, и т. д. Имена модулей нижних уровней иерархии, должны совпадать с именами файлов, в которых они описаны.

Логический файл — это файл одного из следующих типов.

- Block Diagram/Schematics File (расширение — bdf). Эти файлы содержат блок-схемы или непосредственно принципиальные схемы устройств, созданных в графическом схемном редакторе Quartus II
- VHDL File (расширение — vhd). Содержит текстовое описание модуля проекта на языке VHDL.
- Verilog HDL File (расширение — v). Содержит текстовое описание модуля проекта на языке Verilog HDL.
- AHDL File (расширение — tdf). Содержит текстовое описание модуля проекта на языке Altera HDL.
- Vector Waveform File (расширение — vwf). Файл содержит временные диаграммы входных сигналов для симуляции работы модулей проектов.

Вспомогательные файлы хранят дополнительную информацию о проекте.

2.2 Стратегия проектирования

САПР Quartus II позволяет реализовать стратегии восходящего и нисходящего проектирования. И та и другая стратегии подразумевают использование поведенческих и структурных описаний модулей. При структурном описании модуль представляется в виде совокупности взаимосвязанных компонентов более низкого уровня в иерархии описаний. При поведенческом же описании задается алгоритм работы модуля на уровне элементарных вентилей.

Восходящее проектирование (проектирование снизу вверх) применимо в том случае, когда для создаваемого устройства имеется детальное структурное описание (принципиальная схема), выполненное в каком-то определённом элементном базисе.

Таким образом, в процессе проектирования разработчик сначала создает модули нижнего уровня в иерархии описаний, а затем — модуль верхнего уровня. Отсюда и название стратегии проектирования.

Стратегия нисходящего проектирования применяется в том случае, когда задан алгоритм работы (поведенческое описание) создаваемого устройства и набор системных требований. При этом поведенческое описание может быть как формализованным (блок-схема алгоритма, граф, таблица истинности и т. д.), так и неформализованным (словесное описание). Реализация нисходящего проектирования базируется на итерационном выполнении структурной декомпозиции.

Упрощенно, процедура нисходящего проектирования выглядит следующим образом.

- ✓ Разработка архитектуры. Исходное поведенческое описание преобразуется в структурное описание (блок-схему), элементами которого являются архитектурные модули.

- ✓ Архитектурные модули либо описываются на поведенческом уровне (например, с помощью языка VHDL), либо осуществляется их структурная декомпозиция и создается структурное описание, элементами которого являются функциональные модули. Далее процедура итерационно повторяется до тех пор, пока все функциональные модули не будут описаны на поведенческом уровне.

- ✓ После этого осуществляется функциональное моделирование модулей, имеющих поведенческие описания.

- ✓ Функциональное моделирование модулей, имеющих структурное описание (модули, имеющие поведенческое описание, входят в них как компоненты).

- ✓ Моделирование и отладка устройства в целом.

Таким образом, в процессе проектирования разработчик опускается с верхнего уровня иерархии описаний, к нижним уровням. Отсюда и название стратегии проектирования.

Следует отметить, что стратегия нисходящего проектирования имеет безусловные преимущества как по временным затратам на разработку, так и по качеству проработки проекта.

2.3 Ввод описания проекта в среде Quartus II

2.3.1 Базовое окно пакета Quartus II

Запускаем САПР Quartus II и перед нами появляется окно этой среды разработки. Окно САПР Quartus II без загруженных проектов представлено на рисунке 2.2.

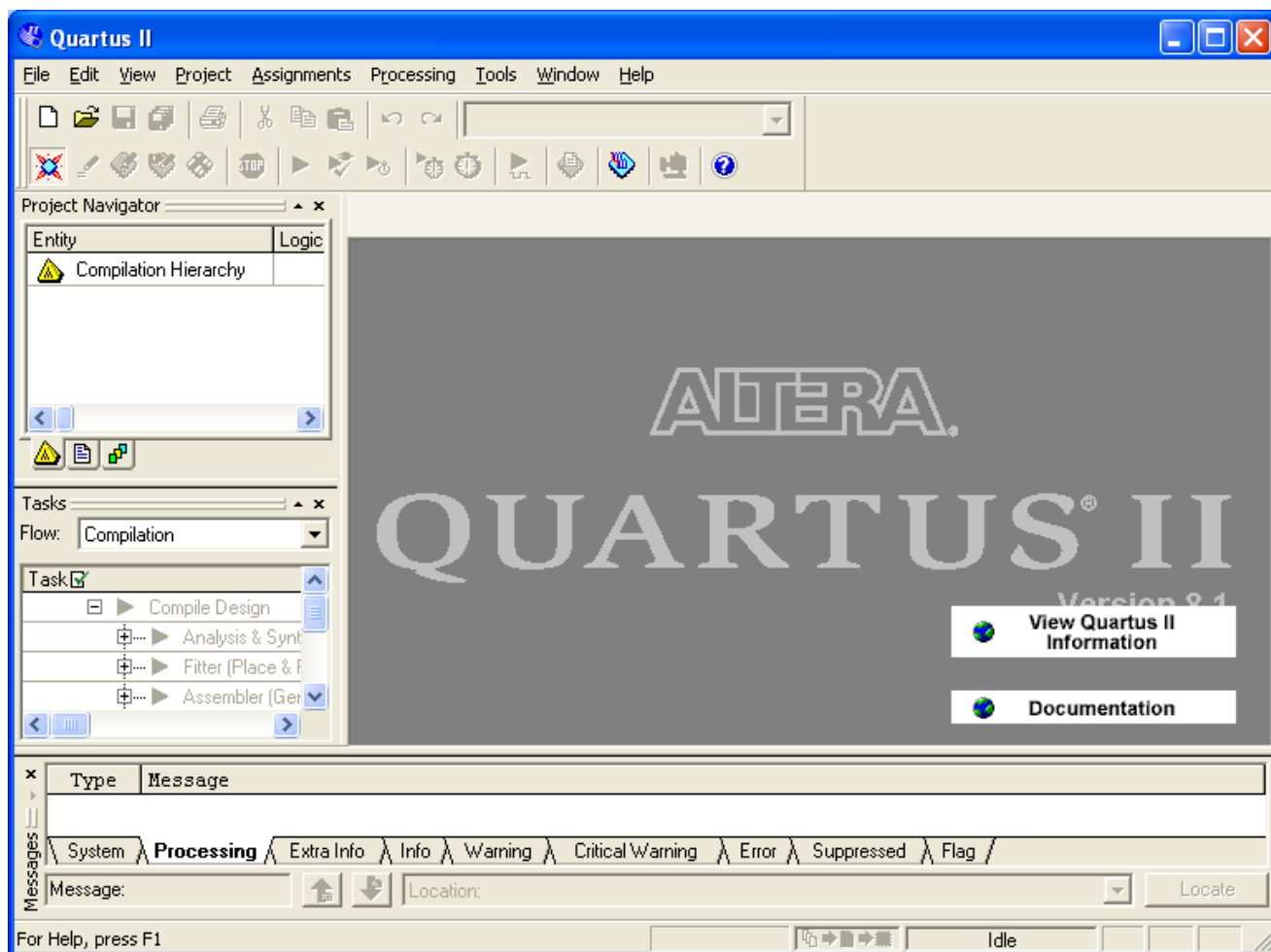


Рисунок 2.2 — Окно САПР Quartus II при отсутствии загруженного

На рисунке 2.3 представлено окно среды Quartus II с загруженным проектом. В самом верху находится строка меню, в разделах которой можно сделать все необходимые настройки, вызвать требуемые редакторы, всевозможные рабочие окна. Далее идет панель инструментов. В левой части окна среды Quartus II находится менеджер проекта, называемый Project Navigator. В самом низу находится окно процессора сообщений с множеством различных вкладок. По вкладкам группируются сообщения определенного типа, например во вкладке error отображаются ошибки проекта, во вкладке warning — предупреждения.

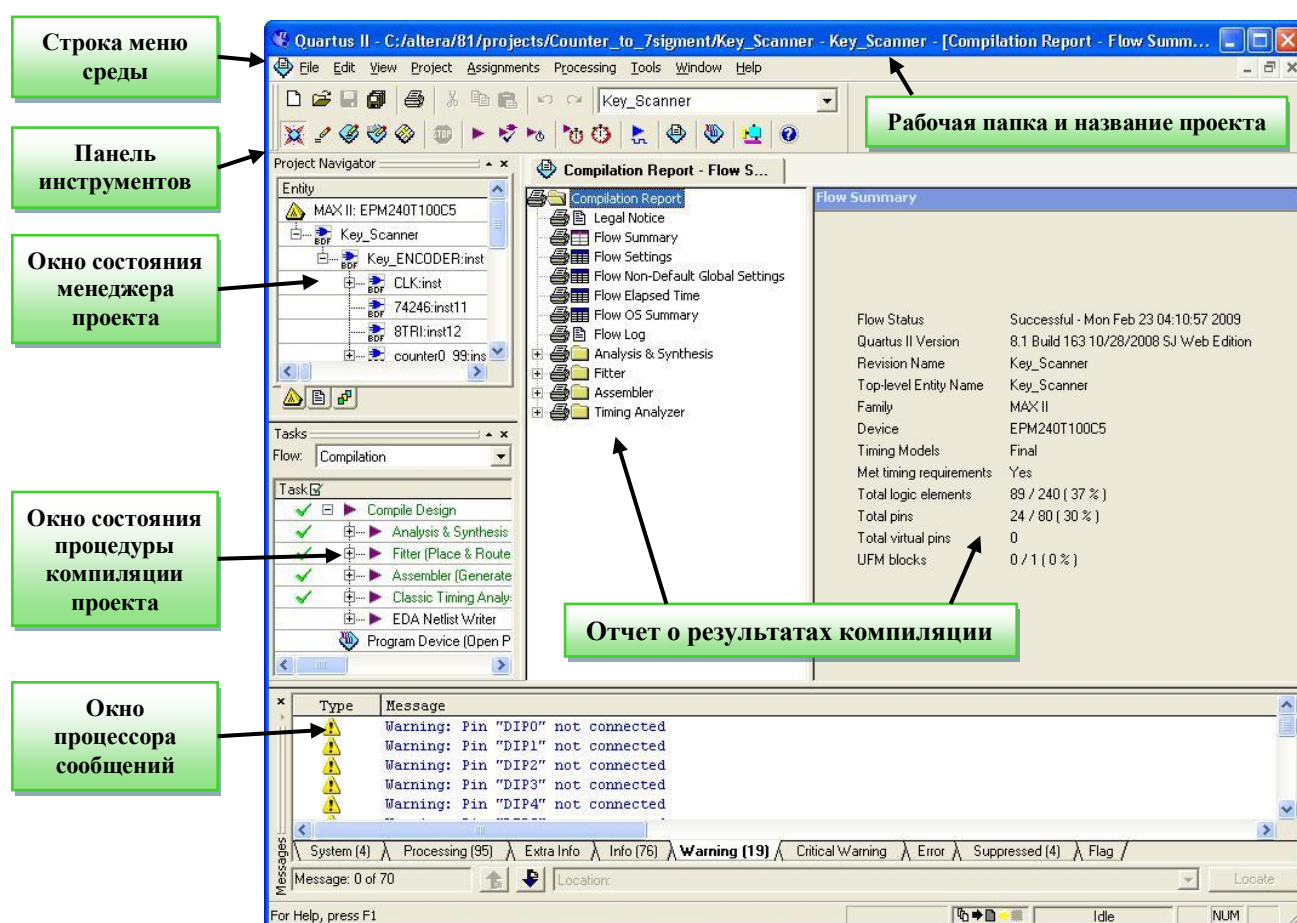


Рисунок 2.3 — Окно САПР Quartus II с загруженным проектом

2.3.2 Создание проекта

Проект в системе Quartus II создаётся с помощью встроенного мастера создания проекта (New Project Wizard). Выбираем в меню File пункт New Project Wizard в результате чего появляется приветствующее окно мастера проектов. Окно представлено на рисунке 2.4.

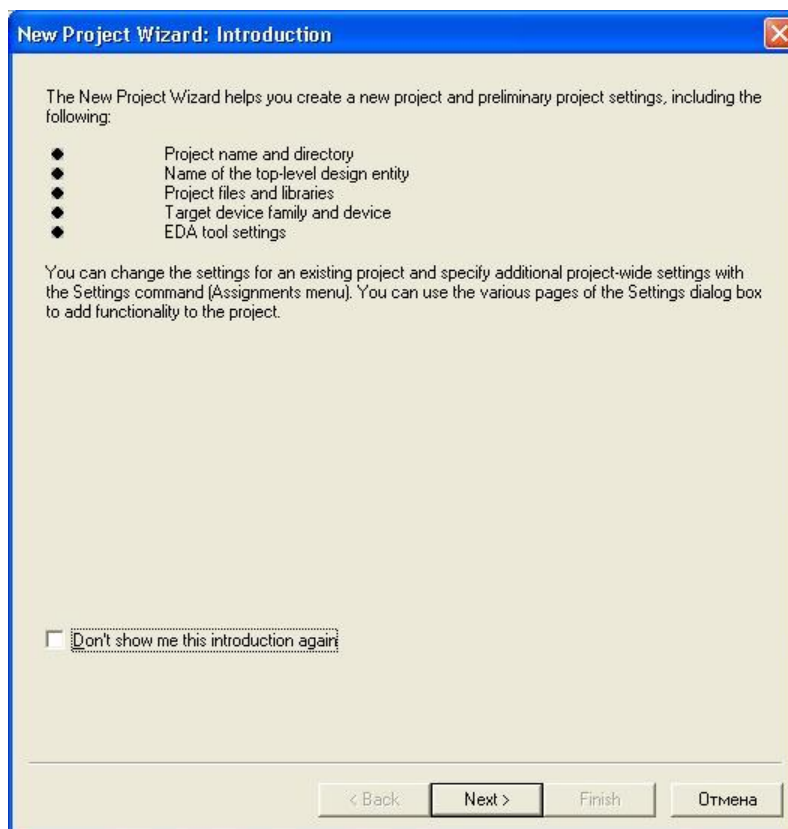


Рисунок 2.4 — Приветствующее окно мастера проектов

Нажимаем Next и в следующем появившемся окне задаем директорию, где будет располагаться наш проект, имя нашего будущего проекта и имя модуля самого верхнего уровня иерархии (top-level design entity). Диалоговое окно для указания директории, имени проекта и имени модуля верхнего уровня иерархии представлено на рисунке 2.5.

Существует некоторое правило, которого следует придерживаться при работе со всеми IDE. Оно способствует удобству в работе с файлами и быстрому

ориентированию в проектах. Правило заключается в том, что для каждого проекта должна быть своя папка, в которой необходимо располагать все файлы проекта. Имя папки проекта должно совпадать с именем самого проекта. Так вот, создаем папу, например LogicExample. Если вводится имя несуществующего каталога, Quartus II создаст его автоматически. Во второй строчке указываем имя нашего проекта. Позже, по окончании работы мастера проектов будет создан файл с этим именем и расширением *.qpf (Quartus II Project File). Это будет файл проекта. Придерживаясь вышеизложенного правила, назовем проект именем — LogicExample.

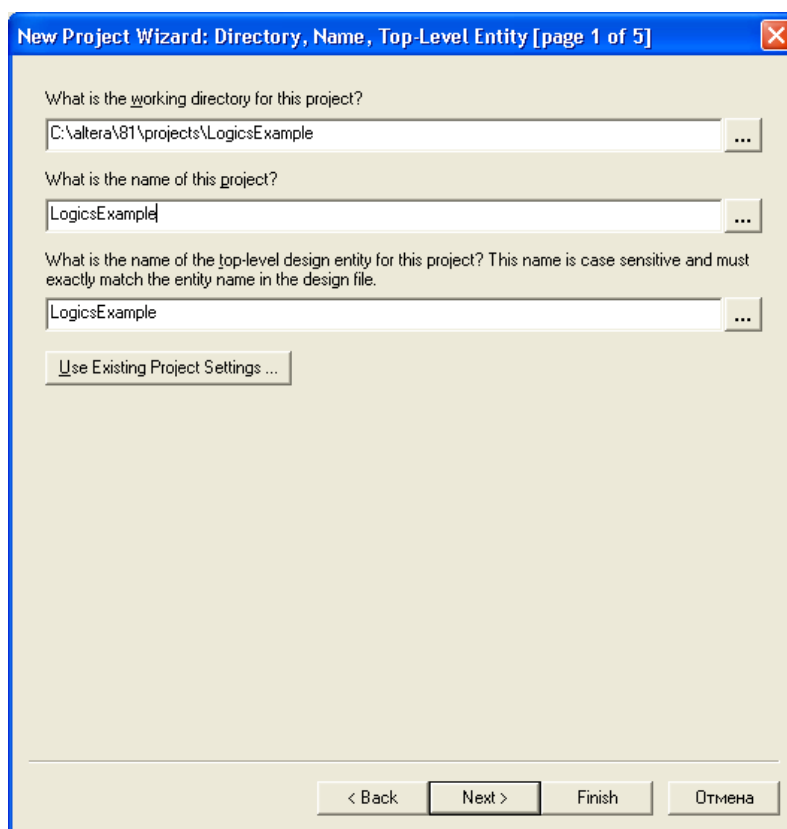


Рисунок 2.5 — Окно для указания директории, имени проекта и имени модуля верхнего уровня иерархии

В третьей строчке необходимо ввести имя, которое будет использоваться для модуля самого верхнего уровня иерархии (top-level entity). Файл, описывающий проект на самом верхнем уровне иерархии, создается пользователем чуть позже. Причем сохранить этот файл необходимо будет с тем же именем, какое было указано в третьей строчке. Конечно, можно задать этому файлу любое другое имя,

но тогда в настройках проекта нужно будет указать, что этот файл является модулем верхнего уровня иерархии всего проекта. Задаем имя модуля верхнего уровня иерархии LogicExample (по умолчанию, заданное имя проекта, автоматически присваивается модулю верхнего уровня иерархии).

Нажимаем Next. Следующее диалоговое окно предназначено для подключения к Вашему проекту ранее созданных файлов, например файлов, содержащих описания устройств из каких либо других проектов. Диалоговое окно Add Files представлено на рисунке 2.6. Путь к дополнительно подключаемым файлам может быть найден при помощи кнопки «Обзор» (Browse — (...)).

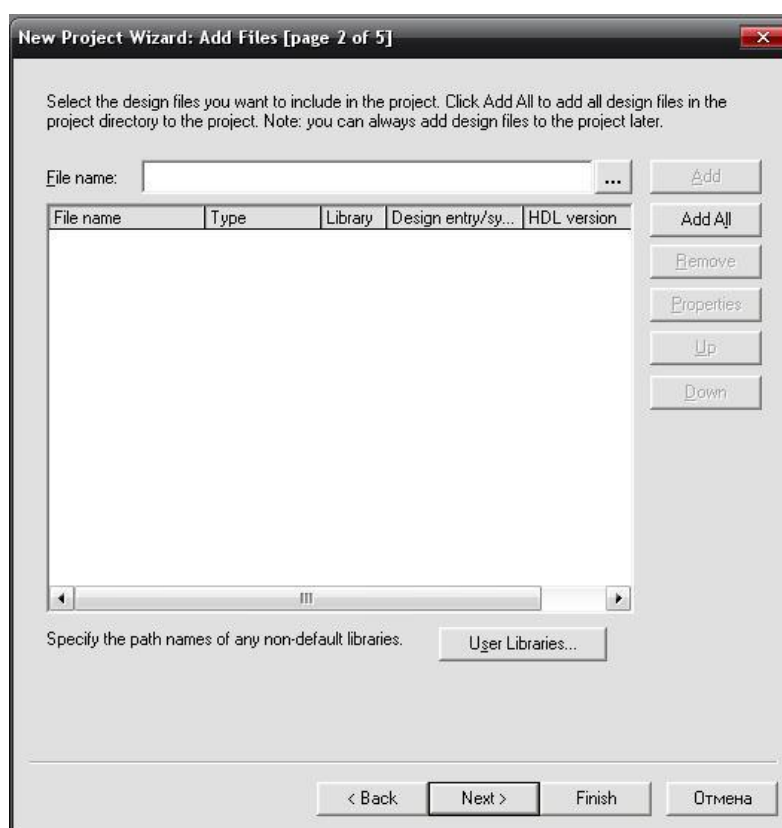


Рисунок 2.6 — Окно для подключения логических файлов и пользовательских библиотек

Нажав кнопочку User Libraries, можно вызвать другое диалоговое окно, предназначенное для подключения собственных библиотек (созданных разработчиком) или каких либо библиотек специализированных функций.

Ни того, ни другого у нас пока нет, поэтому оставляем поля пустыми и нажимаем Next.

Далее появляется окно выбора семейства и типа ПЛИС, которая будет использована для создания проекта. Окно представлено на рисунке 2.7. Мы будем изучать ПЛИС семейства MAXII. В соответствующих окнах указываем семейство (family) MAXII, тип корпуса TQFP (thin quad flat package), количество выводов (pin count) которое имеет микросхема — 100, и класс скорости (speed grade) — 5. Выбираем из списка Available devices микросхему EPM240T100C5. Для завершения операции подбора ПЛИС нажимаем кнопку Next.

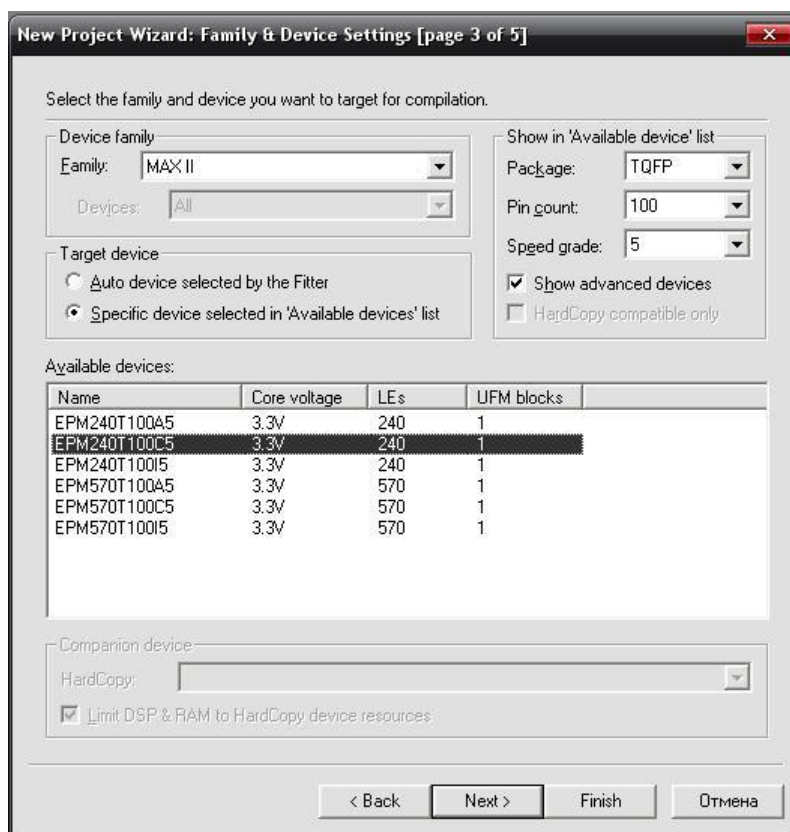


Рисунок 2.7 — Окно выбора семейства и типа ПЛИС

Следующим появляется окно, позволяющее подключить к системе Quartus II при создании данного проекта других средств EDA. Окно представлено на рисунке 2.8. При выполнении лабораторных работ в своих проектах мы не будем использовать дополнительные средства EDA, поэтому, не отмечая ни какие дополнительные средства, нажимаем кнопку Next.

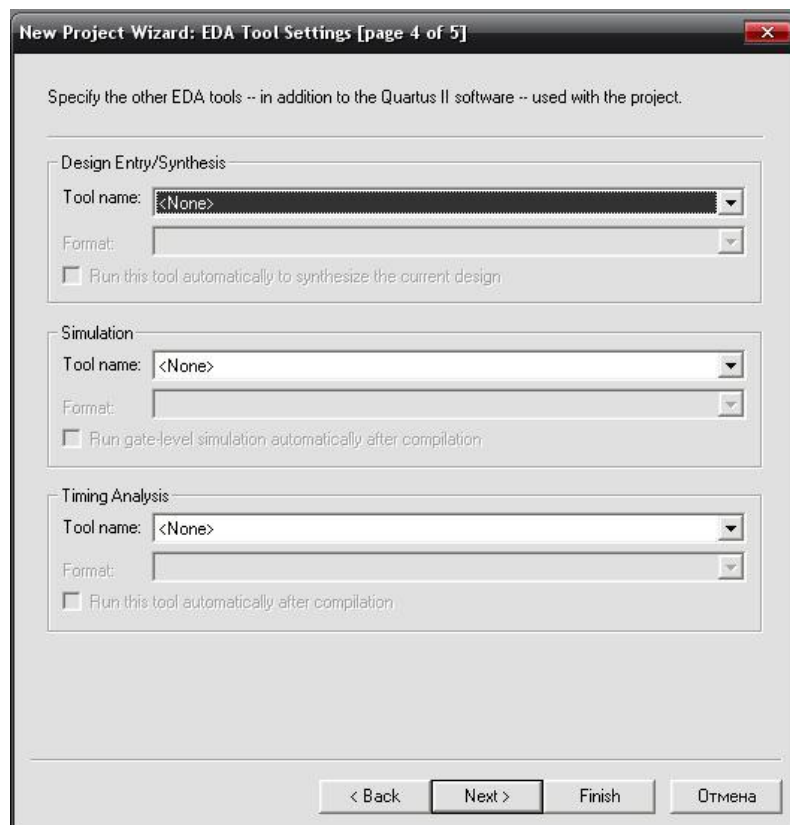


Рисунок 2.8 — Окно для подключения дополнительных средств EDA

Последнее окно, представленное на рисунке 2.9, содержит полную информацию о сделанных настройках. При необходимости можно сделать исправления в настройках, вернувшись к предыдущим окнам, используя клавишу Back. Для завершения создания нового проекта необходимо нажать кнопку Finish.

После создания проекта в окне программы Project Navigator появится имя используемого семейства ПЛИС с названием микросхемы и имя модуля верхнего уровня иерархии разрабатываемого проекта. Одновременно в верхней части основного окна проекта появятся путь к проекту и название проекта. Смотрите рисунок 2.10.

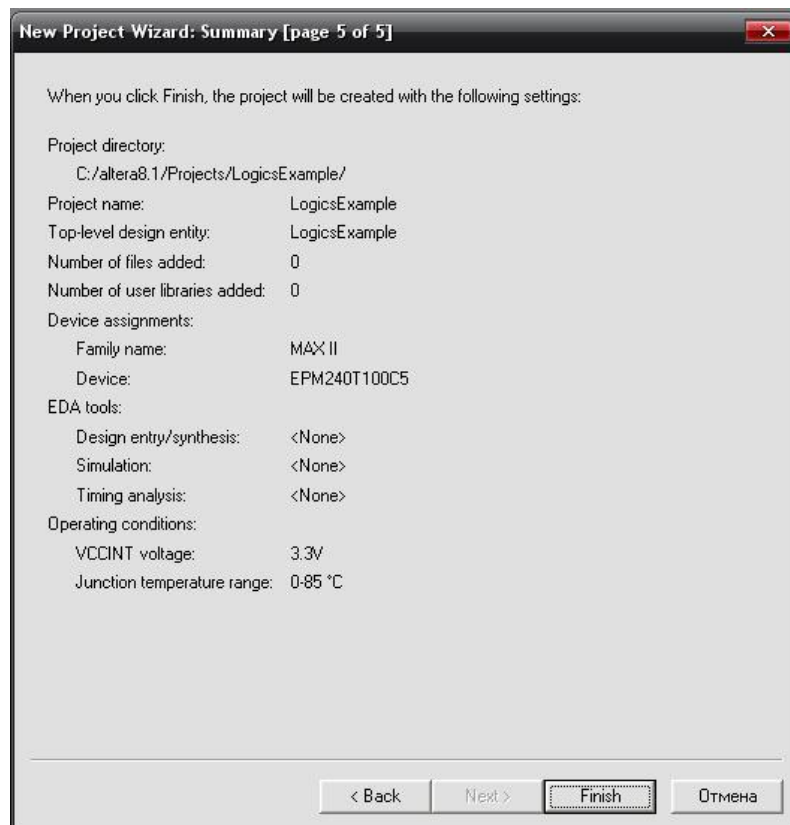


Рисунок 2.9 — Полная информация о сделанных настройках.

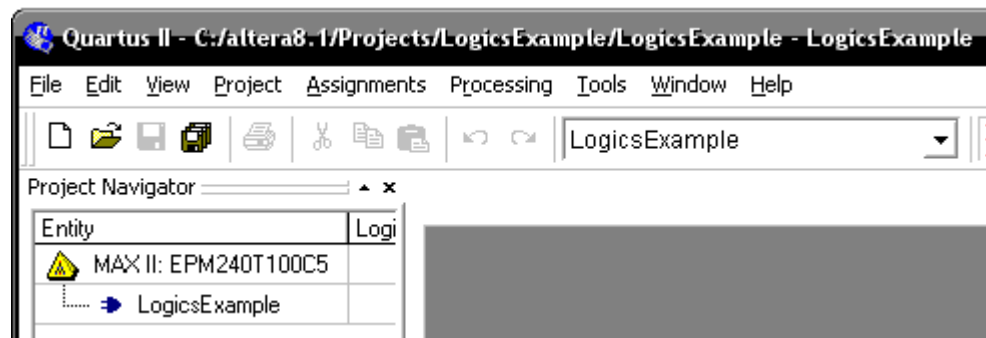


Рисунок 2.10 — Окно Project Navigator с созданным проектом

2.3.3 Создание файла верхнего уровня иерархии проекта.

Создадим файл верхнего уровня иерархии проекта — файл блок схемы устройства. Для этого выберем в меню File команду New. Появляется окно New, позволяющее выбрать тип создаваемого файла. Окно New показано на рисунке 2.11. В открывшемся окне выбираем пункт Block Diagram/Schematic File. Нажимаем кнопку ОК. Откроется окно графического редактора блок-схемы. Таким же образом, используя окно New, создается любой другой файл проекта.

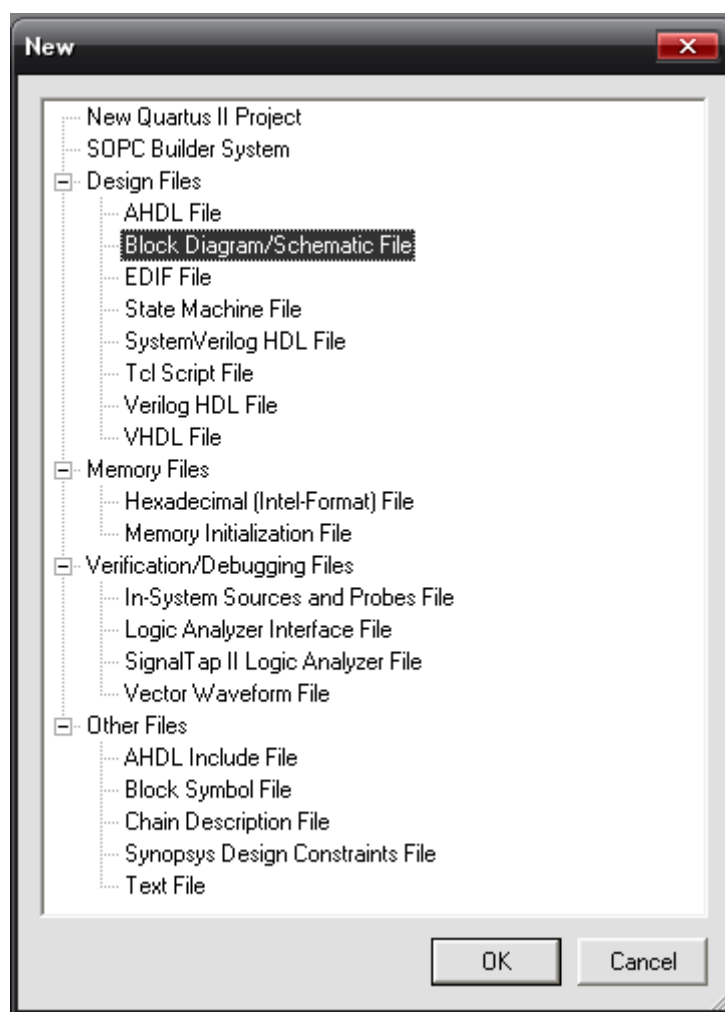


Рисунок 2.11 — Окно New для создания нового файла

Сохраним наш файл как модуль верхнего уровня иерархии. В меню File выбираем команду Save As и вводим имя файла LogicsExample. Мы можем ввести

любое другое имя, но тогда модуль не будет по умолчанию считаться модулем верхнего уровня иерархии, и необходимо будет сделать соответствующие настройки, чтобы он был таковым.

Если поставить галочку около сообщения Add file to current project, то файл после сохранения будет добавлен в текущий проект. Мы так и сделаем.

Окно графического редактора блок-схемы представлено на рисунке 2.12.

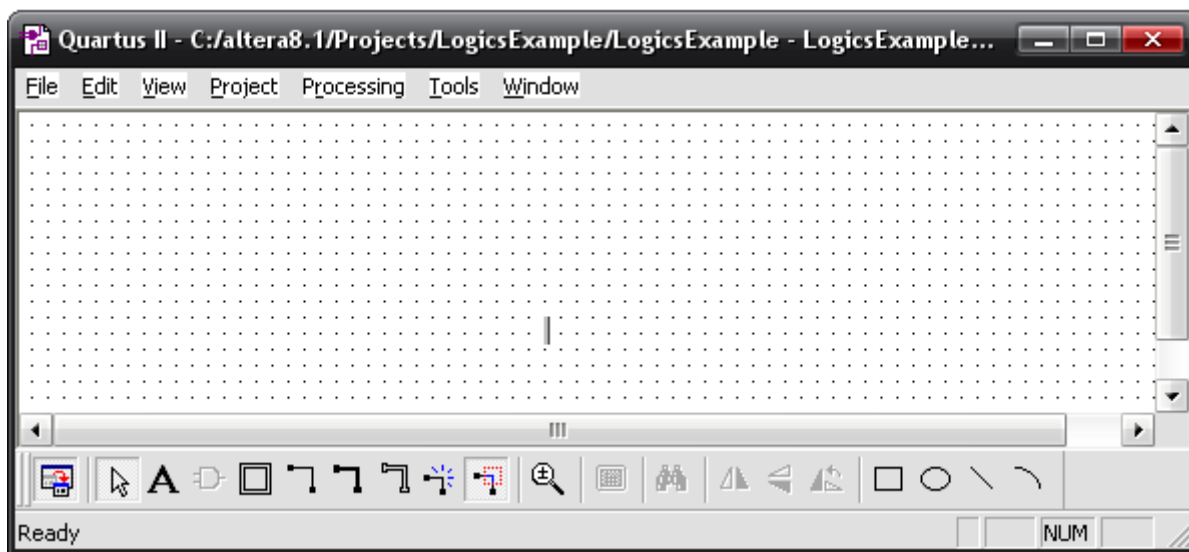


Рисунок 2.12 — Окно графического редактора

Большую часть экрана графического редактора блок-схемы занимает рабочая область, где и будем создавать схему нашего устройства. Как видно в рабочей области по умолчанию имеется специальная сетка, к которой будет осуществляться привязка логики, блоков, проводников и других элементов, определенных в графическом редакторе. По краю рабочей области располагается панель инструментов, с помощью которой мы будем создавать нашу схему (блок-схему).

Рассмотрим панель инструментов графического редактора. Назначение различных инструментов панели приведено на рисунке 2.13.

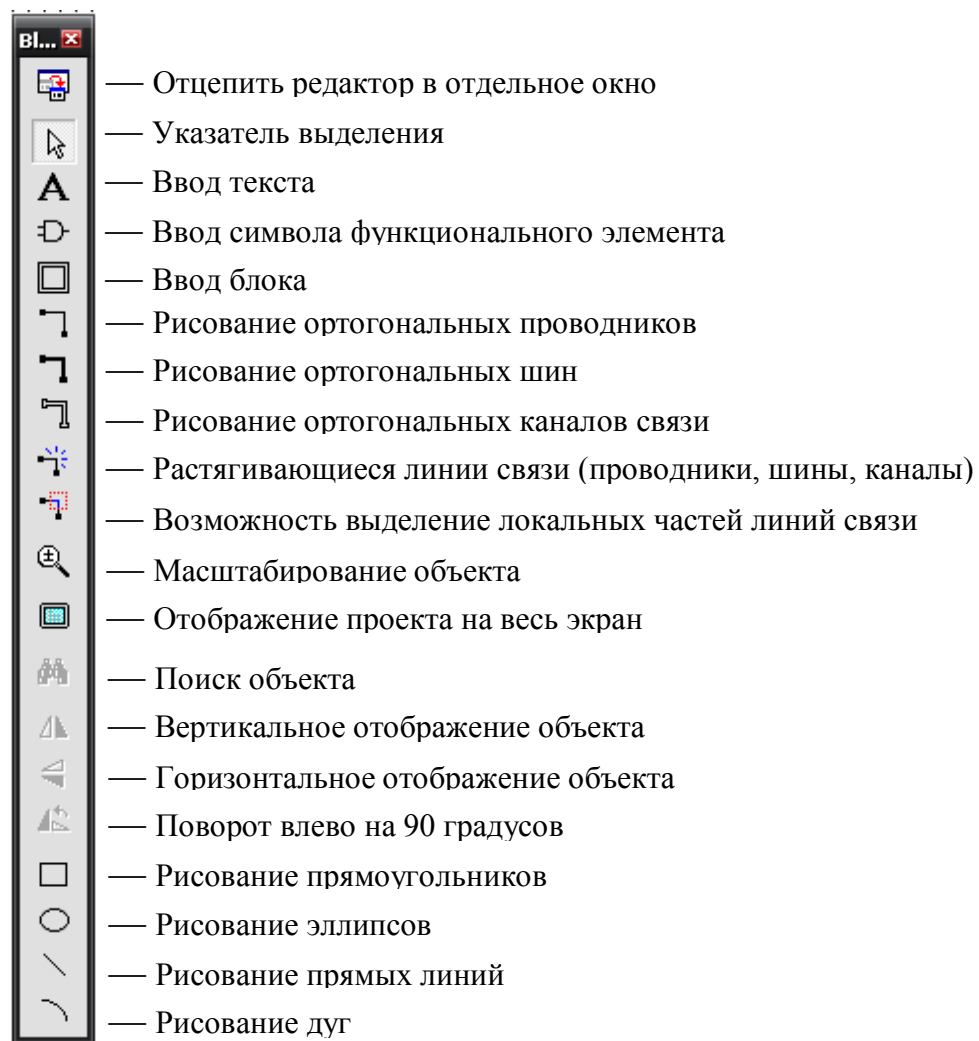


Рисунок 2.13 — Панель инструментов графического редактора блок-схем

Теперь вставим два логических вентиля 2И, а результат их операции объединим вентилем 2ИЛИ. Для того чтобы вставить в рабочую область графического редактора блок-схем вентиль необходимо нажать кнопку ввода символа функционального элемента. При этом появится диалоговое окно предлагающее выбрать требуемый элемент. Окно Symbol представлено на рисунке 2.14. В левом верхнем углу окна находится сгруппированный по субдиректориям список библиотечных элементов и мегафункций. Чуть ниже находится окно Name, с помощью которого можно быстро найти нужный элемент по его имени или же указать путь к другим элементам (кнопка browse), не входящим в стандартную библиотеку (к элементам созданных разработчиком).

Кнопка MegaWizard Plug-In Manager вызывает менеджер, с помощью которого можно создать свою мегафункцию или изменить библиотечную. Мегафункции это параметризованные блоки (полуфабрикаты) позволяющие на своей основе спроектировать функционально законченные узлы проекта. Библиотечные мегафункции представлены в субдиректории megafunctions окошка Libraries диалогового окна Symbol.

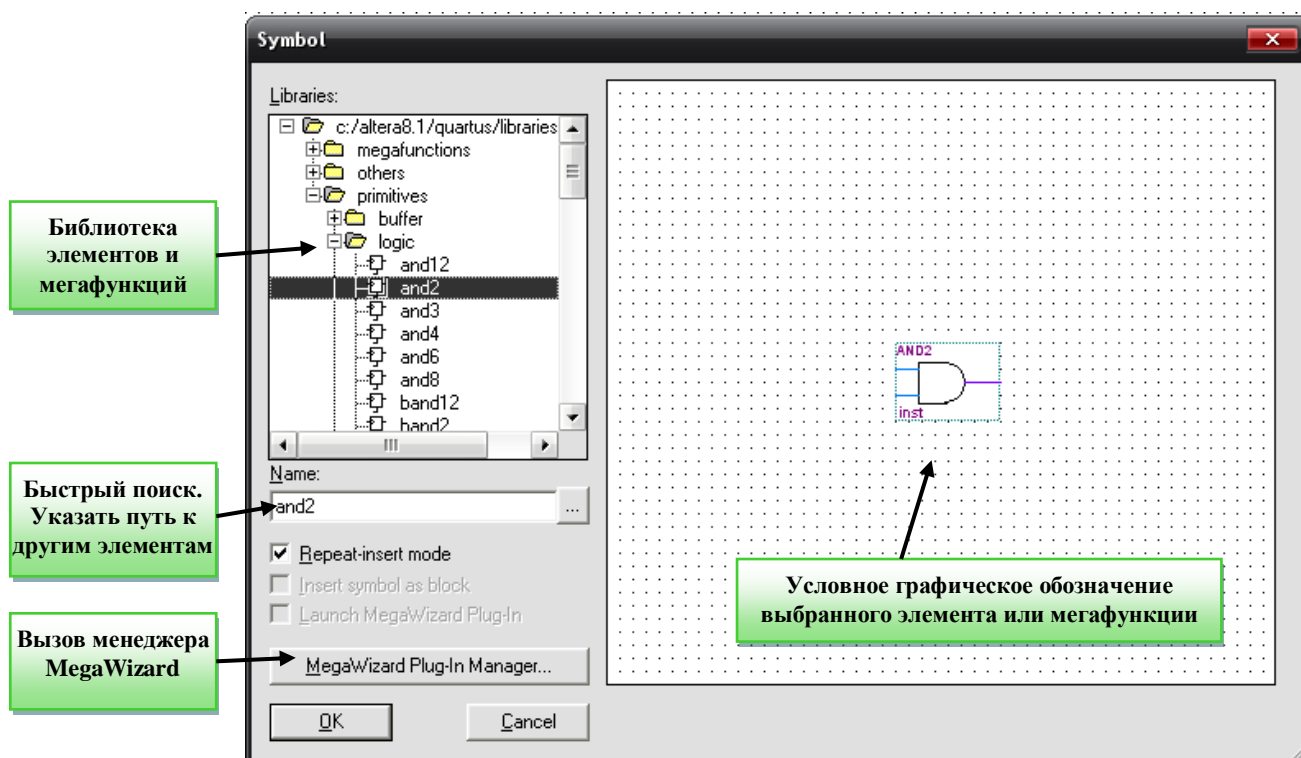


Рисунок 2.14 — Окно для выбора символа функционального элемента Symbol

Нажимаем ОК и вставляем два символа элемента 2И (AND2) в рабочую область графического редактора блок-схем. При вставке символа элемента автоматически включается режим «Повторного ввода» (Repeat-insert-mode) при котором один символ можно вставить в несколько мест проекта. Введенный символ привязывается к курсору. Теперь при нажатии левой кнопки мыши символ вводится на указанное в данный момент место схемы. Далее его можно перевести в другое место схемы и там его аналогичным способом зафиксировать. Для завершения вставки достаточно нажать на клавиатуре клавишу ESC или на правую кнопку мыши. Появляется окно, в котором можно выбрать одну из двух команд: вставить

здесь (Insert Here) и выход (Cancel). При выборе команды «Вставить здесь» процесс вставки продолжается. При выборе команды «Выход» — процесс ввода символа элемента завершается.

Таким же образом находим в библиотеке и вставляем элемент 2ИЛИ (OR2). Для того чтобы потом можно было соединить нашу схему с выводами микросхемы или же сделать её содержимым блока или символом, необходимо добавить в нашу схему соответствующие порты. Используя тот же инструмент — ввод символа функционального блока, вставим в рабочую область четыре входных и один выходной порт. Элемент «порт» находится в субдиректории pin директории primitives. Смотри рисунок 2.15.

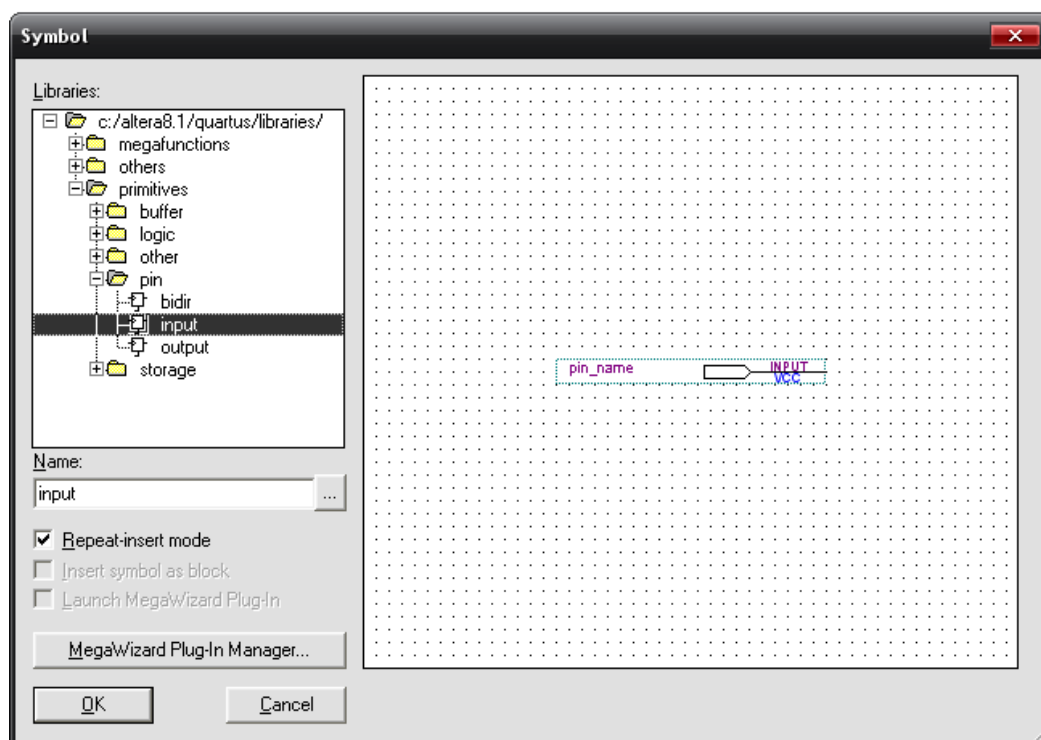


Рисунок 2.15 — Добавление элемента — порта

В результате рабочая область графического редактора блок-схем будет выглядеть так, как представлено на рисунке 2.16.

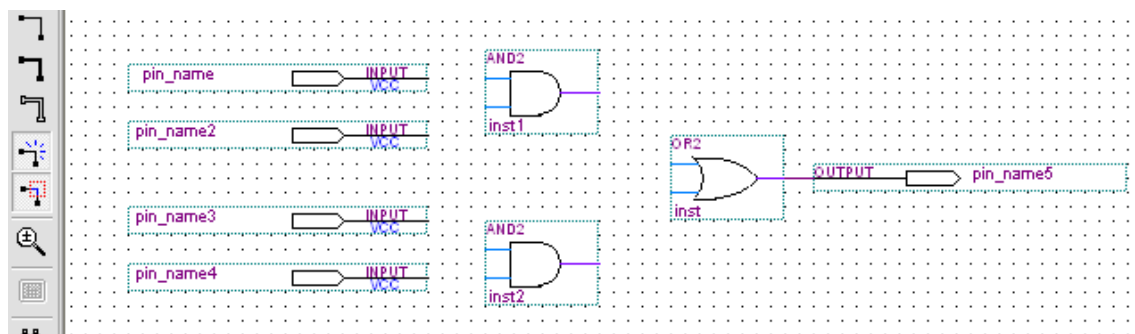


Рисунок 2.16 — Окно графического редактора и часть панели инструментов

Зададим собственные имена для портов, для этого двойным щелчком по порту вызываем диалоговое окно, в поле Pin name(s) которого вводим новое имя. Диалоговое окно представлено на рисунке 2.17. Введем имена: a, b, c, d — для входов и F — для выхода.

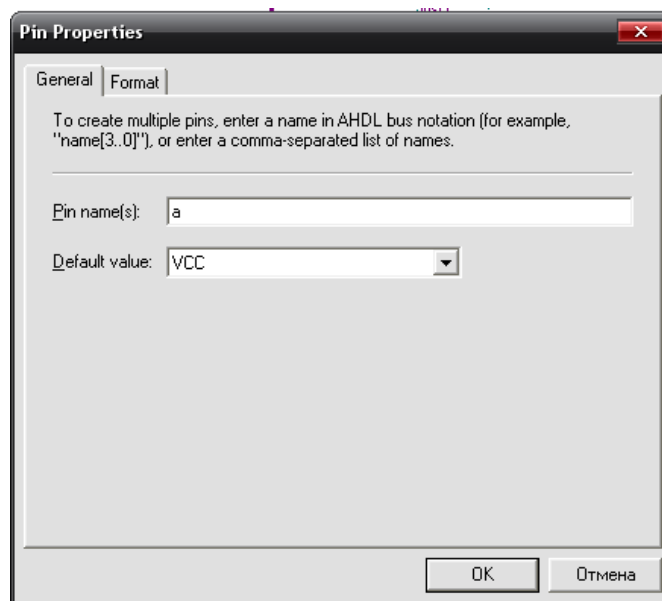


Рисунок 2.17 — Окно Pin Properties

В разделе «Значение по умолчанию» (Default value) выставляется исходное значение переменной, либо логическая единица (VCC), либо логический ноль (GND). Выбор команды «Формат» (Format) позволяет изменить цвета линий и текста. Завершается назначение имени порту нажатием клавиши OK.

По аналогии вводятся имена всех входных и выходных выводов проекта.

Разведем линии связи между элементами. Можно сразу в панели инструментов включить такие опции как растягивание проводников и возможность их локального выделения. Включенные инструменты представлены на рисунке 2.16. Использование этих опций делает разводку линий связи между символами элементов и блоками более удобной. Затем выберем инструмент «рисование ортогональных проводников» и нарисуем связи. Итак, теперь наша схема имеет вид, представленный на рисунке 2.18.

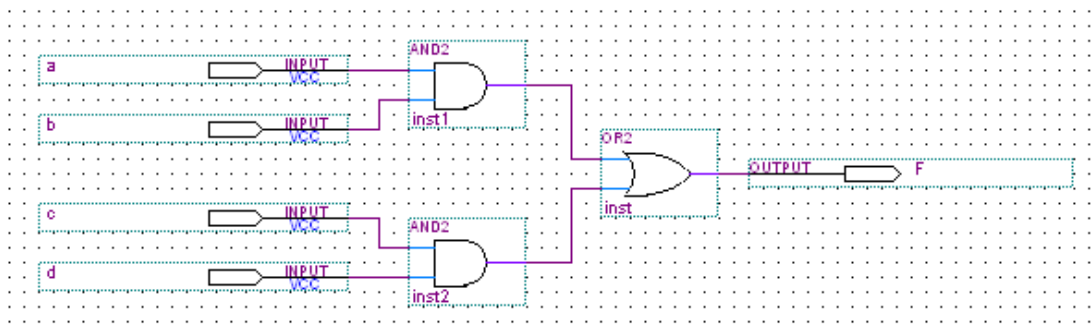


Рисунок 2.18 — Схема устройства выполняющего логическую функцию

$$F = (a \text{ and } b) \text{ or } (c \text{ and } d)$$

Проводникам, шинам, линиям в канале тоже можно присваивать имена. Например, чтобы не загромождать проект линиями связи (если их много) то можно не соединять их непосредственно и лишь задать одинаковое имя, так как это сделано на рисунке 2.19. Но в этом случае ухудшается наглядность схемы. Для задания проводнику имени необходимо щелкнуть по нему правой кнопкой мыши, в выпадающем меню выбрать properties и в появившемся диалоговом окне ввести имя.

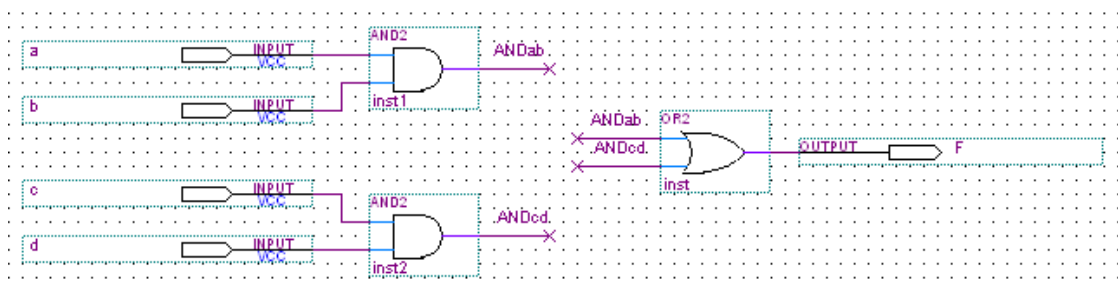


Рисунок 2.19 — Пример разводки линий связи

В окне будет три вкладки на первой вкладке (General) вводится имя проводника, во второй и третьей (Font и Format) выбирается шрифт, цвет проводника и текста.

В графическом редакторе также можно использовать шины, если мы хотим подключить к какому-нибудь блоку не все разряды шины и какую-то их часть, то шины обязательно следует поименовать определенным образом, чтобы конкретно определить какие разряды шины подключаются. Для шины задается имя и её разряды. В примере, представленном на рисунке 2.20, мы создали шину DataBUS[7..0]. Шина имеет имя DataBUS, и является 8-разрядной. Разрядность задается в квадратных скобках через две точки. Для обращения отдельно к каждому разряду (проводнику) шины, указывается имя шины и номер проводника, например DataBUS5.

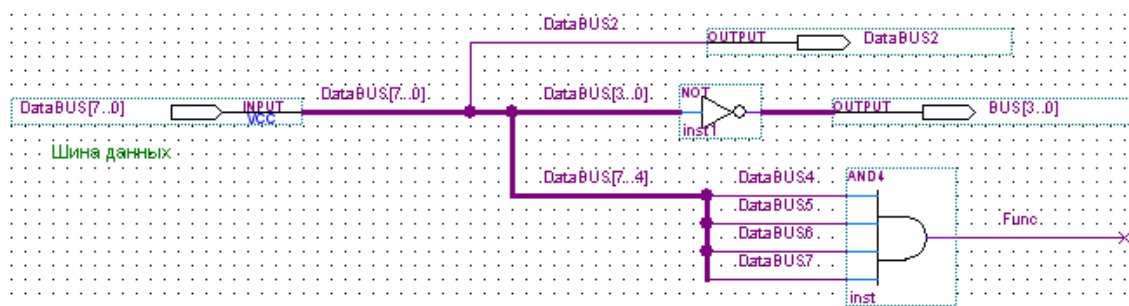


Рисунок 2.20 — Пример использования шин

Корректировать свои действия можно используя команды отмены (Undo) и повтора (Redo), действий меню «Правка» (Edit). При необходимости, элементы и проводники можно расположить в рабочей области графического редактора наиболее удобным образом. Для этого на панели инструментов необходимо выбрать команду «указатель выделения» — стрелку. Выбрать данную команду можно также нажав клавишу **ESC**. Навести указатель мыши на передвигаемый элемент, нажать левую кнопку мыши и, удерживая её, переместить элемент на новое место.

После создания схемы необходимо записать проект (команда Save в меню File). Далее можно приступить к компиляции. Для этого в меню Processing выбираем Start Compilation, или находим и нажимаем соответствующую кнопочку на панели инструментов главного окна САПР Quartus II.

2.3.4 Настройка свойств экрана графического редактора

Если необходимо, то в режиме редактора блок-схем можно изменить основные свойства главного окна программы. Для этого в меню Tools основного окна программы необходимо выбрать команду Options. В открывшемся окне из списка Category необходимо выбрать строку Block/Symbol Editor. Окно Options, с выбранной категорией Block/Symbol Editor представлено на рисунке 2.21.

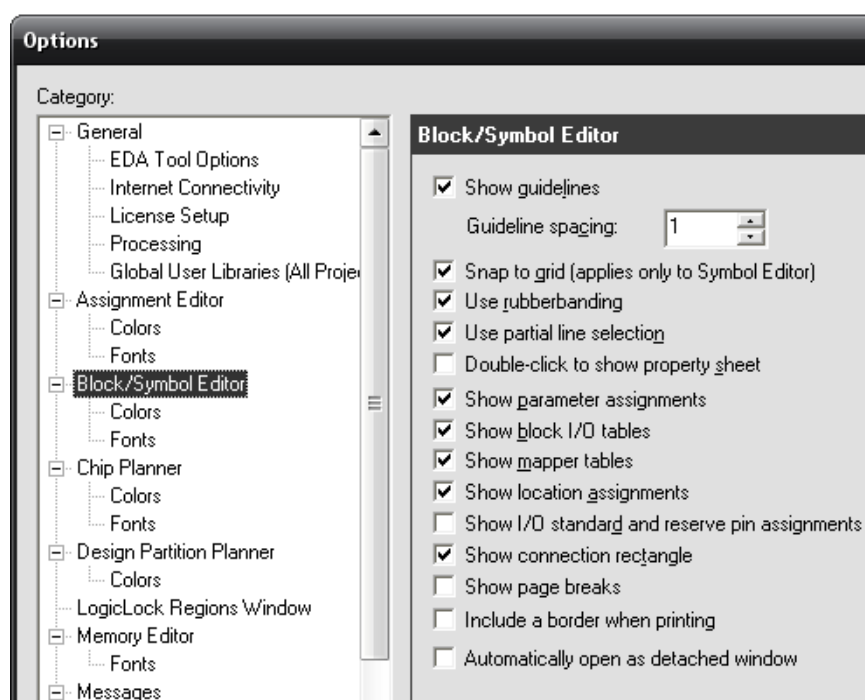


Рисунок 2.21 — Окно настроек среды Quartus II

Окно позволяет редактировать следующие свойства графического редактора блок-схем.

- ✓ *Show Guidelines* — показать сетку;
- ✓ *Guidelines spacing* — шаг сетки;
- ✓ *Snap to grid (applies only to Symbol Editor)* — привязка к сетке (только в режиме редактора символов);
- ✓ *Use rubberbanding* — использование растягивающихся линий связей;
- ✓ *Use partial line selection* — использовать возможность локального выделения линий связи.

Последние две настройки присутствуют в виде кнопок на панели инструментов графического редактора.

- ✓ *Double-click to show property sheet* — разрешить вызов окна свойств двойным щелчком мыши;
- ✓ *Show parameter assignment* — показывать назначение параметров;
- ✓ *Show block I/O tables* — показывать для блоков таблицы входов-выходов;
- ✓ *Show location assignment* — показывать разводку на выводы микросхемы (какие номера выводов микросхемы соответствуют портам схемы реализованной в графическом редакторе блок-схем);
- ✓ *Show I/O standard and reserve pin assignments* — показывать назначенные стандарты ввода-вывода для выводов микросхемы и зарезервированные выводы. Если галочка установлена, и соответствующие выводы микросхемы настроены на работу с определенными стандартами (например, 3.3-V LVTTL, 3.3-V LVCMOS, и т. д.), то в графическом редакторе соответствующие порты будут сопровождаться информационным окном, в котором будет указан этот стандарт.
- ✓ *Show connection rectangle* — показывать прямоугольники в местах соединения;
- ✓ *Show page breaks* — показывать границы проекта в рабочей области графического редактора блок-схем.
- ✓ *Include a boarder when printing* – включить границы при печати проекта.
- ✓ *Automatically open as detached windows* — автоматически открывать графический редактор блок-схем отдельным окном. Если галочка не стоит, то при создании файла *.bdf графический редактор будет открываться в виде вкладки, интегрированной основное окно Quartus II.

Изменить используемые в редакторе цветовую гамму и шрифт, можно с помощью подсписков Colors и Fonts списка Block/Symbol Editor.

Уменьшение или увеличение изображения осуществляется с использованием соответствующего значка на панели инструментов или командами «Увеличить» (Zoom In), «Уменьшить» (Zoom Out) или «Вписать изображение в окно» (Fit in Window). Команды находятся в меню «Вид» (View).

2.3.5 Настройка проекта

Для того чтобы задать определенные настройки нашему проекту необходимо выбрать из меню Assignments пункт Settings. Появится соответствующее диалоговое окно Settings, представленное на рисунке 2.22. В категории General должен быть указан файл самого верхнего уровня иерархии проекта.

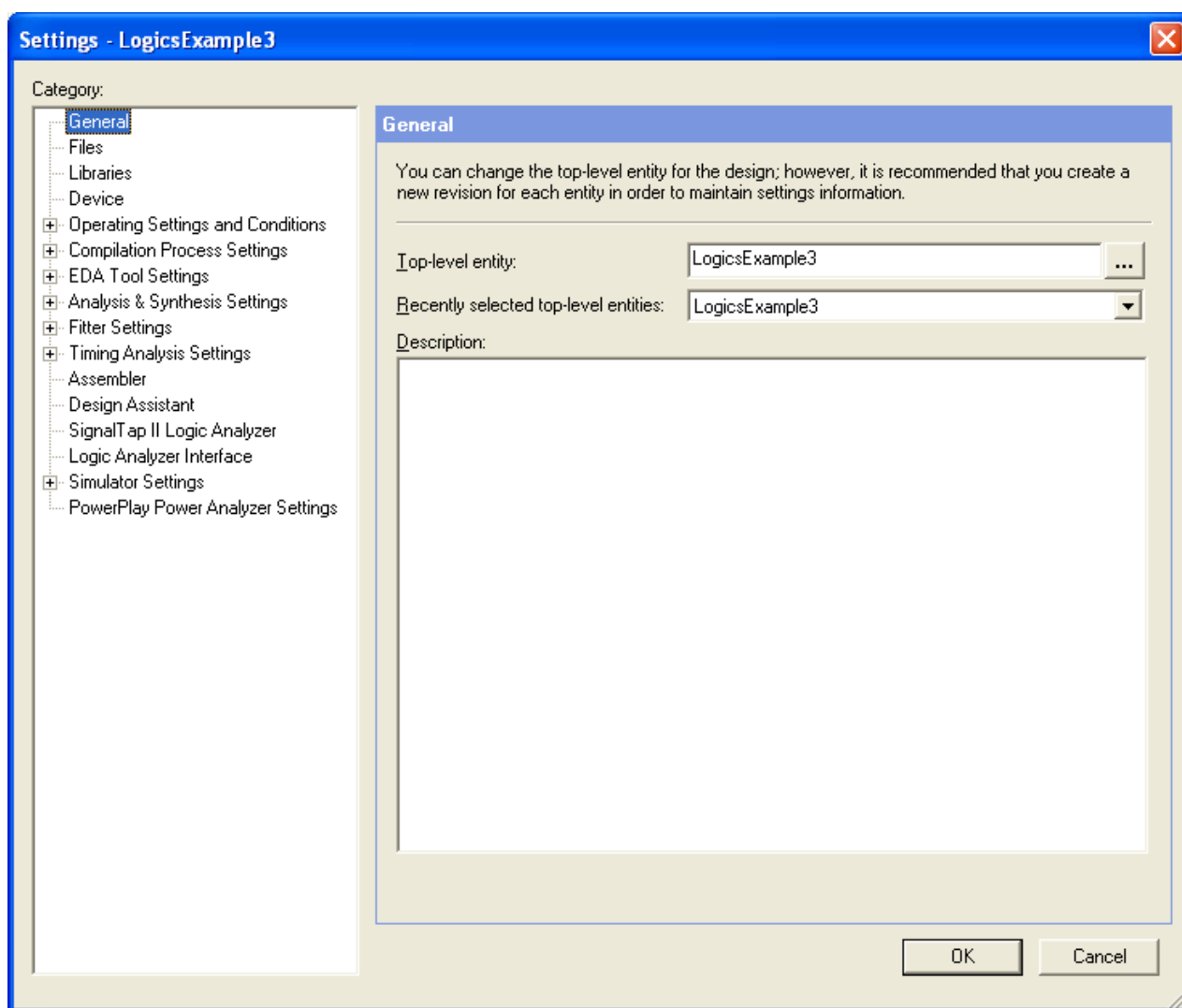


Рисунок 2.22 — Окно настройки проекта, категория General

В категории Files в виде списка указываются логические файлы, входящие в проект. Окно настройки проекта с категорией Files представлено на рисунке 3.23. При необходимости можно удалить файл из проекта, удалив его только из этого

списка. Логические файлы, не включенные в проект, не компилируются (хотя при этом они могут находиться в директории проекта).

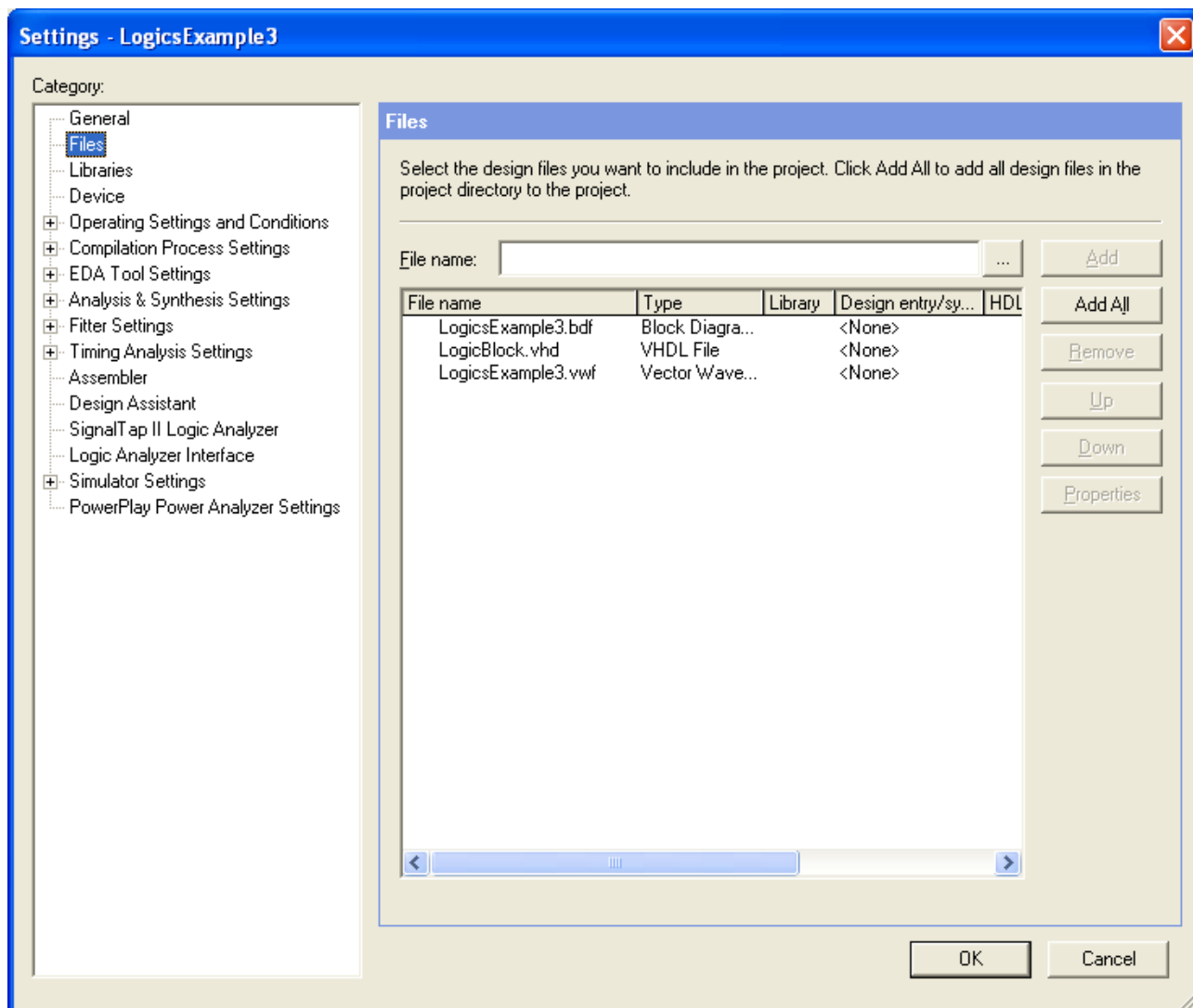


Рисунок 2.23 — Окно настройки проекта, категория Files

В категории Libraries можно подключить дополнительные библиотеки, в категории Device выбрать микросхему, для которой создается проект. Окно настроек с категорией Device представлено на рисунке 2.24. При создании своего проекта обязательно необходимо сделать следующую настройку. Необходимо нажать кнопку Device and Pin Options и в появившемся диалоговом окне, представленном на рисунке 2.25, установить галочку Enable device-wide output enable (DEV OE), для того чтобы системная логика лабораторного стенда могла переводить лабораторную ПЛИС в Z-состояние.

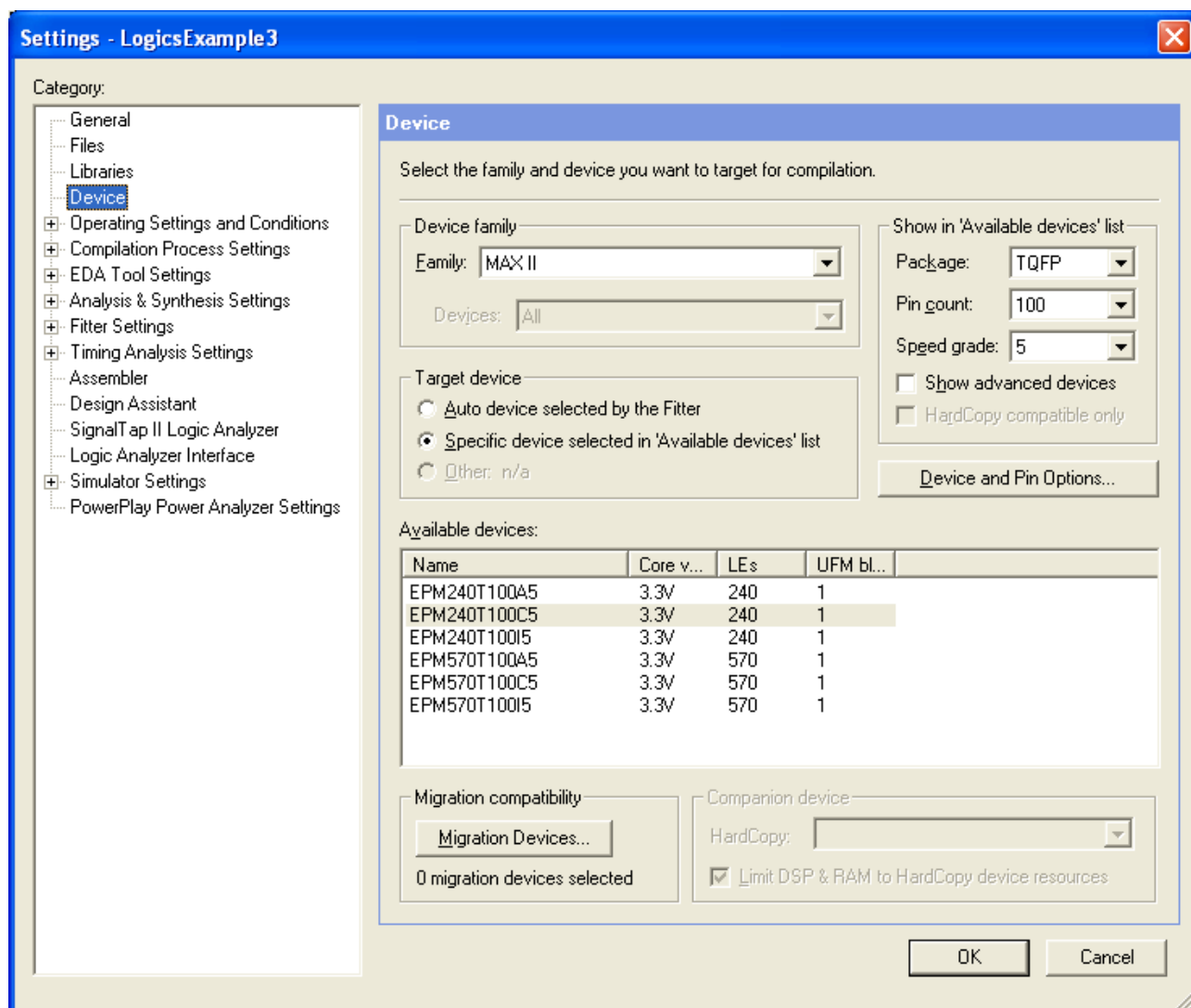


Рисунок 2.24 — Окно настройки проекта, категория Device

Так же используя окно Settings можно задать и некоторые другие настройки: настройки компиляции, симуляции и т. п.

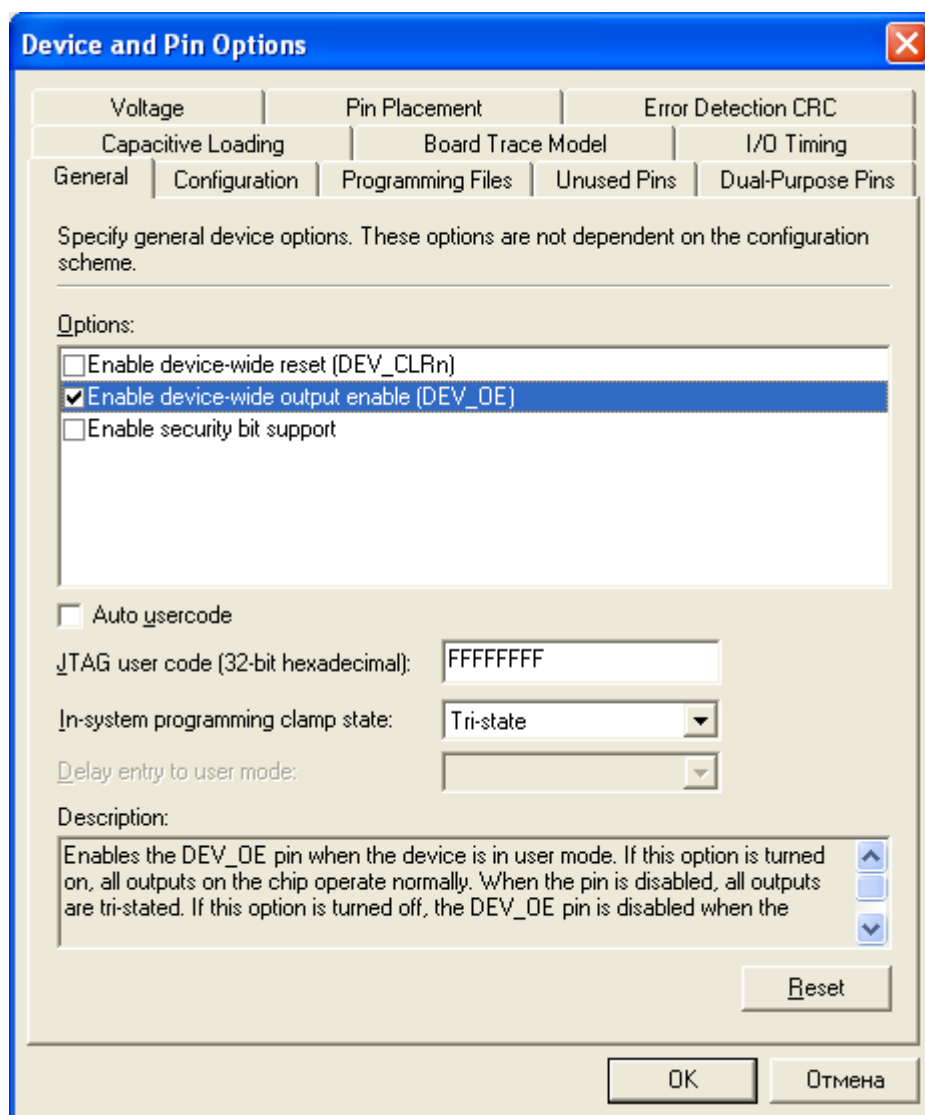


Рисунок 2.25 — Окно настройки Device and Pin Options

2.3.6 Создание иерархии проекта

Для создания иерархии проекта в графическом редакторе блок-схем САПР Quartus II используются специальные элементы: *символы* и *блоки*. Как уже было сказано, проект в системе Quartus II должен состоять обязательно из файла верхнего уровня иерархии и необязательных файлов нижних уровней иерархии (если проект простой, и использование иерархии не требуется). При реализации иерархии в

графический файл верхнего уровня встраиваются *блоки* и *символы*, которые содержат описание своего функционирования в файлах нижнего уровня иерархии.

Рассмотрим, пример создания собственного *символа*. Для примера возьмем мультиплексор 2 в 1, но не простой, а синхронный и с третьим состоянием (Z-состояние). Можно сразу определить какие входы и выходы будут у синхронного мультиплексора. Мультиплексор будет иметь два информационных входа, с одного из которых сигнал будет проходить на выход. Назовем входы *A* и *B*, а выход *F*. Какой из входных сигналов пройдет на выход будет определяться входом *S* (Select). Так как у нас мультиплексор будет синхронный то он должен иметь сигнал синхронизации. Назовем его *Clk*. Для перевода выхода *F* мультиплексора в Z-состояние добавим управляющий сигнал с названием *OE* (Output Enable).

Теперь мы уже можем создать *символ* нашего мультиплексора. Для этого в меню File выбираем New. В открывшемся диалоговом окне в разделе Other Files выбираем Block Symbol File (*.bsf). В результате откроется окно редактора символов, представленное на рисунке 2.26. Двойным щелчком по границе самого символа (пунктирной линии) добавляем входы и выходы. Имя входов и выходов задается в появляющемся диалоговом окне Port Properties. Диалоговое окно представлено на рисунке 2.27. Размеры создаваемого символа можно менять. Используя панель инструментов редактора символов можно рисовать прямоугольники, овалы, прямые, дуги. Например, для инверсных входов или выходов для наглядности следует рисовать кружки. С помощью кнопки Text Tool панели инструментов укажем, что созданный нами символ является мультиплексором (MUX). В результате символ мультиплексора в окне редактора символов может выглядеть, так как показано на рисунке 2.28. Сохраняем наш символ, например, с именем SynchroMUX.bsf.

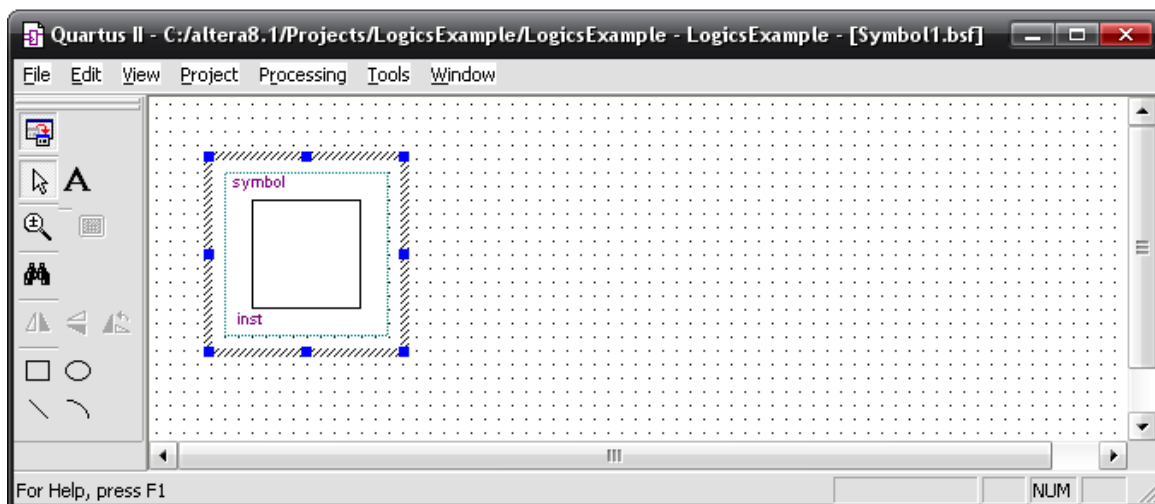


Рисунок 2.26 — Редактор символов САПР Quartus II

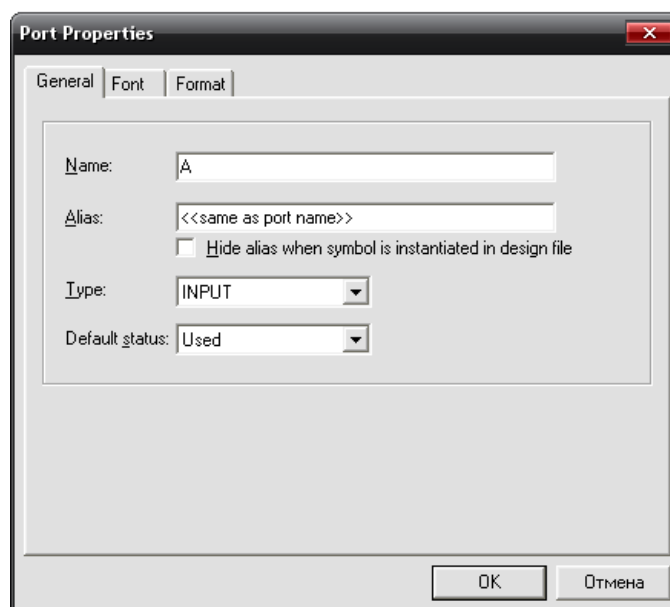


Рисунок 2.27 — Окно Port Properties

Теперь наполним наш символ содержанием, т. е. создадим описание нашего символа. Описание символа может быть сделано как в графическом виде, т. е. с использованием графического редактора блок-схем, так и на специальном языке описания аппаратуры (VHDL, Verilog, AHDL). Создадим описание этого символа с использованием графического редактора. Следует учесть следующее: чтобы связать созданный символ с его описанием, необходимо чтобы имя файла описывающего функционирование символа совпадало с именем файла самого символа (bsf).

Создаем файл иерархией ниже, чем LogicExample.bdf, и называем его так же как называется сам символ — SynchroMUX.bdf. Вводим описание этого символа в графическом редакторе. Описание символа представлено на рисунке 2.29.

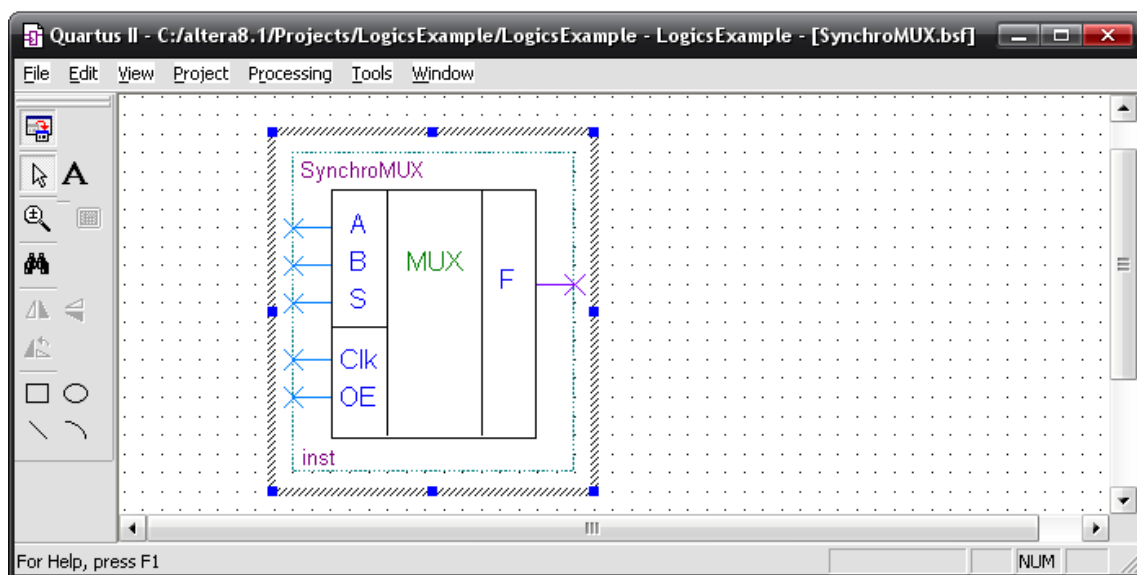


Рисунок 2.28 — Символ мультиплексора

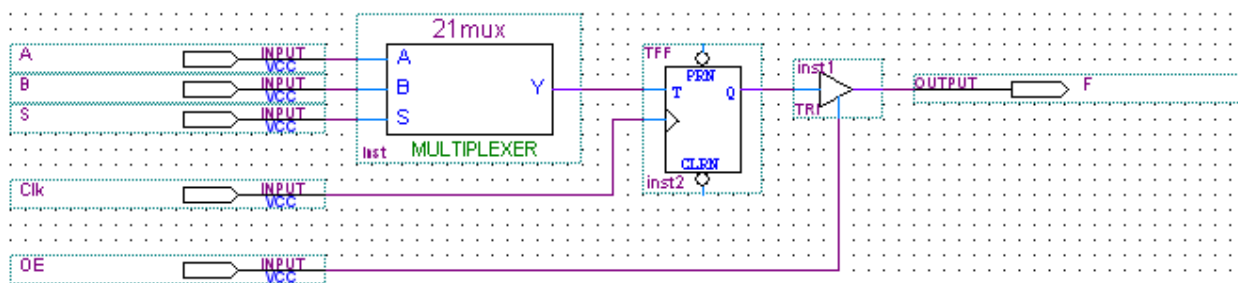


Рисунок 2.29 — Описание синхронного мультиплексора с Z-состоянием

Для описания использовались стандартные символы из библиотеки Quartus II: мультиплексор (21mux), синхронизируемый по фронту Т-триггер (TFF) и 3-стабильный буфер (TRI). Сохраняем описание.

Теперь в графическом файле иерархией выше, в нашем случае это файл LogicExample, используя инструмент «ввод символа функционального элемента», вставляем созданный нами символ. Причём в поле Libraries диалогового окна Symbol появится папочка Project, в которой будет находиться созданный нами

символ SynchroMUX (смотрите рисунок 2.30). В эту папочку автоматически помещаются все созданные пользователем символы текущего проекта.

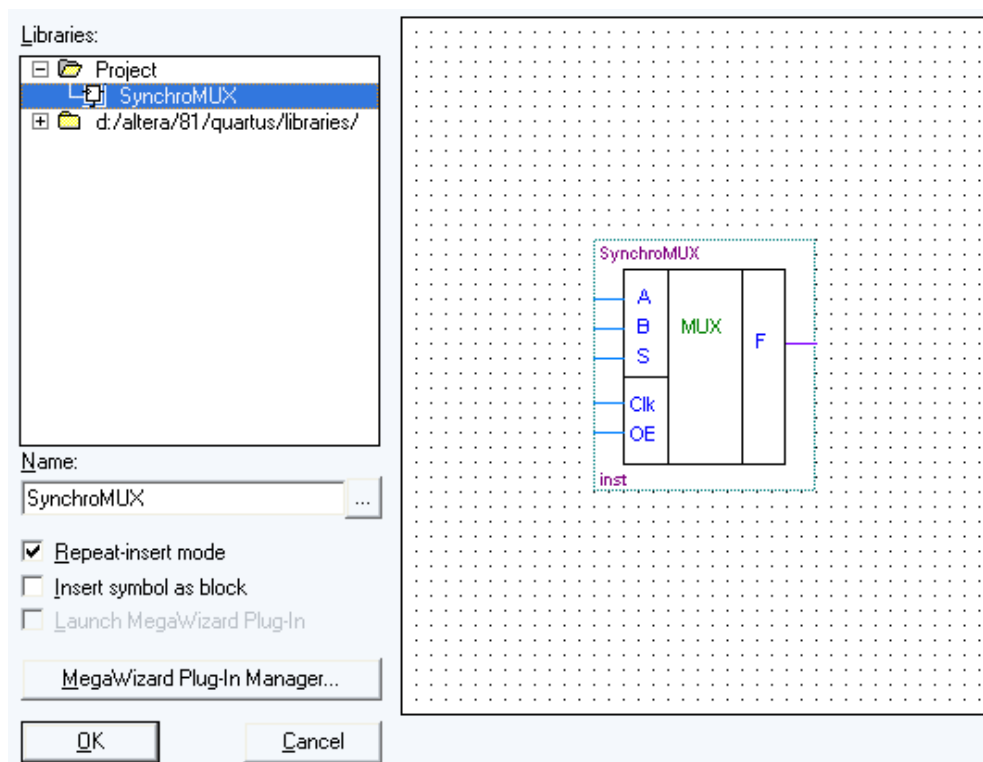


Рисунок 2.30 — Вставка символа SynchroMUX в проект

Использование символа мультиплексора SynchroMUX в графическом файле LogicExample, показано на рисунке 2.31. Компилируем проект. Для того чтобы посмотреть описание символа необходимо выделить его двойным щелчком.

В данном случае мы рассмотрели создание символа и затем его описание, такая стратегия используется при проектировании сверху вниз. Но можно наоборот, сначала создать описание символа, а затем сгенерировать символ, соответствующий описанию. Для этого, создаем описание символа (графический файл bdf). За тем в меню File выбираем раздел Create/Update, в выпадающем списке выбираем Create Symbol File for Current File. После чего среда Quartus II предлагает ввести имя файла, и сохранить его. Внешний вид символа можно отредактировать: щелкаем по нему правой кнопкой мыши, и в выпадающем списке выбираем Edit Selected Symbol or Block. Запускается знакомый нам редактор символов.

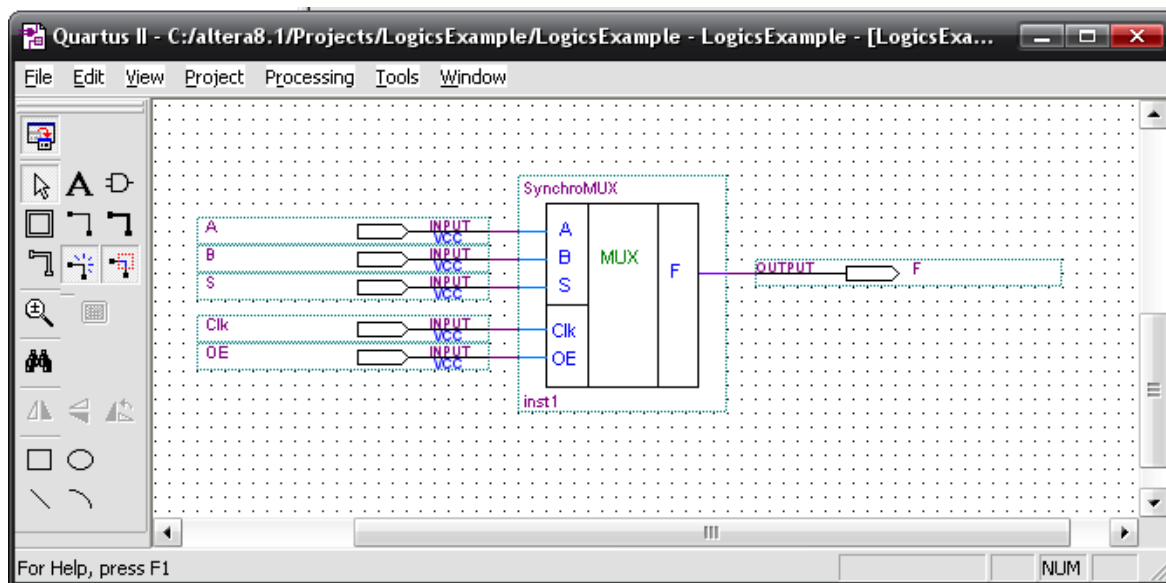


Рисунок 2.31 — Использование символа мультиплексора SynchroMUX в графическом файле иерархией выше LogicExample

Описание символа может быть выполнено на HDL-языке (Hardware Description Language) — языке описания аппаратуры. В качестве примера рассмотрим создание описания символа на языке VHDL. Схема, которую мы будем описывать представлена на рисунке 2.32.

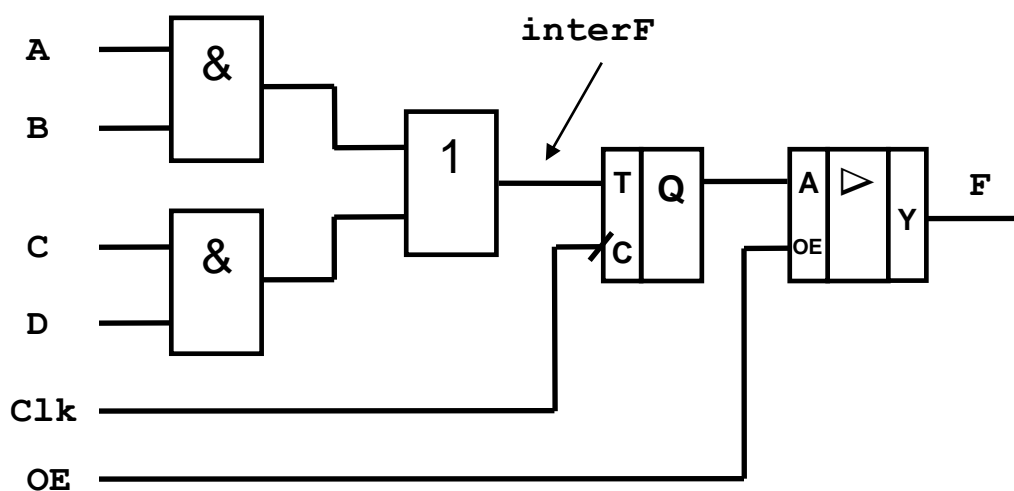


Рисунок 2.32 — Пример простой логической схемы

В меню File выбираем New, в открывшемся диалоговом окне в разделе Design Files выбираем VHDL File. В результате откроется текстовый редактор языка VHDL.

Вводим описание функционирования нашей схемы на языке VHDL. Описание схемы на языке VHDL имеет следующий вид.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY LogicBlock IS
    PORT
    (
        B      :    IN   STD_LOGIC;
        A      :    IN   STD_LOGIC;
        C      :    IN   STD_LOGIC;
        D      :    IN   STD_LOGIC;
        Clk    :    IN   STD_LOGIC;
        OE     :    IN   STD_LOGIC;
        F      :    OUT  STD_LOGIC
    );
END LogicBlock;

ARCHITECTURE Gates OF LogicBlock IS
BEGIN
    PROCESS(Clk, OE)
        VARIABLE interF : STD_LOGIC;
    BEGIN
        IF (OE = '1') THEN
            IF (RISING_EDGE(Clk)) THEN
                interF := (A and B) or (C and D);
            END IF;
            F <= interF;
        ELSE F <= 'Z';
        END IF;
    END PROCESS;
END Gates;
```

Здесь мы не будем объяснять работу приведенного VHDL-кода, а рассмотрим только принцип создания символа на основе этого кода. Основы языка VHDL рассматриваются в четвертом разделе данного методического пособия. Чтобы разобраться в работе этого кода изучите четвертый раздел. Для получения начальных навыков работы с САПР, приведенный пример VHDL-кода можно скопировать в текстовый редактор Quartus II и проделать все те операции, которые здесь описаны, самостоятельно.

Сохраняем файл с текстовым описанием схемы на языке VHDL (файл имеет расширение vhd). Компилируем файл. Затем в меню File выбираем раздел Create/Update, из которого выбираем Create Symbol Files for Current File, после чего система создаст соответствующий VHDL-описанию символ. Теперь можем использовать наш символ в графическом редакторе. Пример использования символа, сгенерированного автоматический системой показан на рисунке 2.33.

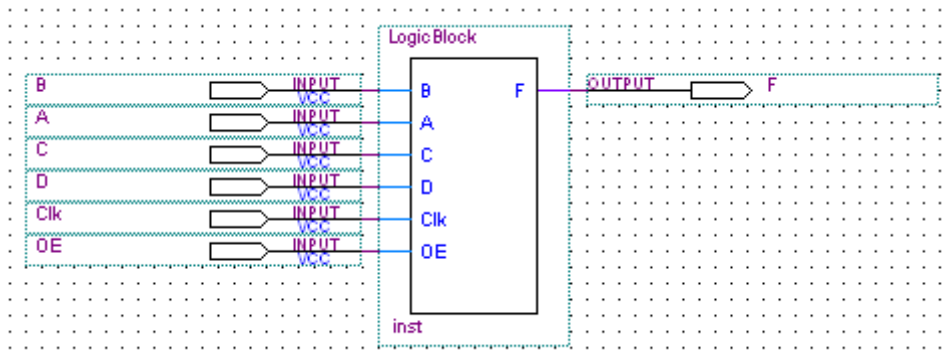


Рисунок 2.33 — Пример использования символа, созданного автоматический системой на основе VHDL-описания

Для того чтобы разработчик сразу по внешнему виду символа понимал какие функции он выполняет, чтобы схема была читабельна и наглядна, необходимо её соответствующим образом оформлять. Для этого можно использовать редактор символов. В этом случае приведенная выше схема, после редактирования будет выглядеть, так как представлено на рисунке 2.34.

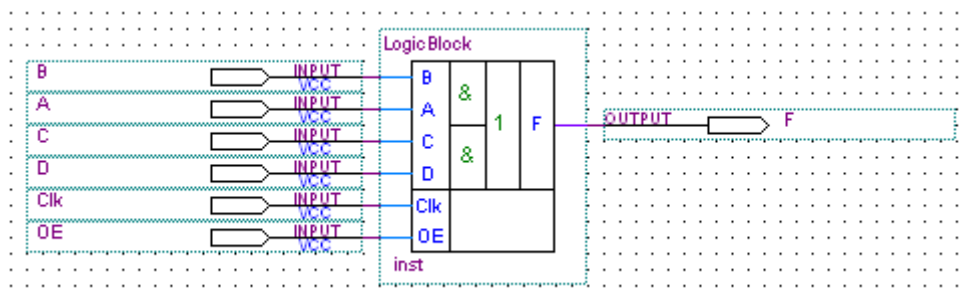


Рисунок 2.34 — Пример использования символа, отредактированного в редакторе символов

2.4 Моделирование проекта

2.4.1 Основные положения

Моделирование (Simulation) позволяет убедиться в правильности функционирования разработанного проекта. Анализируется результат работы проекта в соответствии с заданными входными воздействиями.

Исходными данными для моделирования являются внешние воздействия, заданные в виде некоторого входного вектора (набора входных параметров). Подсистема моделирования (Simulator) среды Quartus II, в соответствие с алгоритмом проекта, синтезирует выходные сигналы в ответ на входные.

Модель достаточно точно отображает работу схемы, которая будет реализована непосредственно в микросхеме ПЛИС. Симулятор учитывает особенности реальной работы проекта в ПЛИС: особенности данной микросхемы, задержки распространения сигнала и т. п. Поэтому не стоит удивляться, когда значение сигнала на выходе меняется через какой-то определённый промежуток времени после изменения входного сигнала.

Обычно, в типовых задачах разработчик задает наборы входных векторов и анализирует полученные в результате моделирования выходные сигналы.

В зависимости от поставленной цели подсистема моделирования позволяет выполнить:

- функциональное моделирование проекта (Functional Simulation) при котором проверяется правильность описания и логического функционирования проекта;
- моделирование с учетом временных параметров реальной ПЛИС (Timing Simulation), позволяющее проверить не только правильность логического функционирования проекта, но и его работу с учетом реальных параметров выбранной ПЛИС в самых жестких условиях эксплуатации.

2.4.2 Создание вектора входных воздействий

Файлы вектора входных воздействий в системе Quartus II задаются в графической форме (в виде временных диаграмм) с использованием редактора временных диаграмм (Waveform Editor). Файлы имеют расширение *.vwf (Vector Waveform File).

Создание файла *.vwf.

Рассмотрим создание файла (*.vwf), содержащего временные диаграммы. В меню File выбираем команду New и в открывшемся диалоговом окне в группе Verification/Debugging Files выделяем строку Vector Waveform Files (файл вектора временных диаграмм) и нажимаем кнопку OK. После чего откроется пустое окно редактора временных диаграмм с именем по умолчанию Waveform1.vwf. Редактор временных диаграмм представлен на рисунке 2.35. Созданный файл необходимо сохранить, используя команду Save As меню File. Программа автоматически предложит сохранить файл с именем, совпадающим с именем файла верхнего уровня проекта, присвоив ему расширение .vwf. Для завершения процесса создания файла необходимо нажать кнопку Save. При этом необходимо обратить внимание на наличие флажка около надписи Add file to current project (добавить файл к текущему проекту). Если флажок поставлен, то система автоматически включит созданный файл в текущий проект.

Для удобства, на область временных диаграмм нанесена временная сетка, предназначенная для визуальной привязки сигналов к конкретным временным интервалам. Используя команду Grid Size меню Edit можно изменить шаг временной сетки. Используя команду End Time меню Edit можно изменить время окончания моделирования, т.е. изменить время всего процесса моделирования.

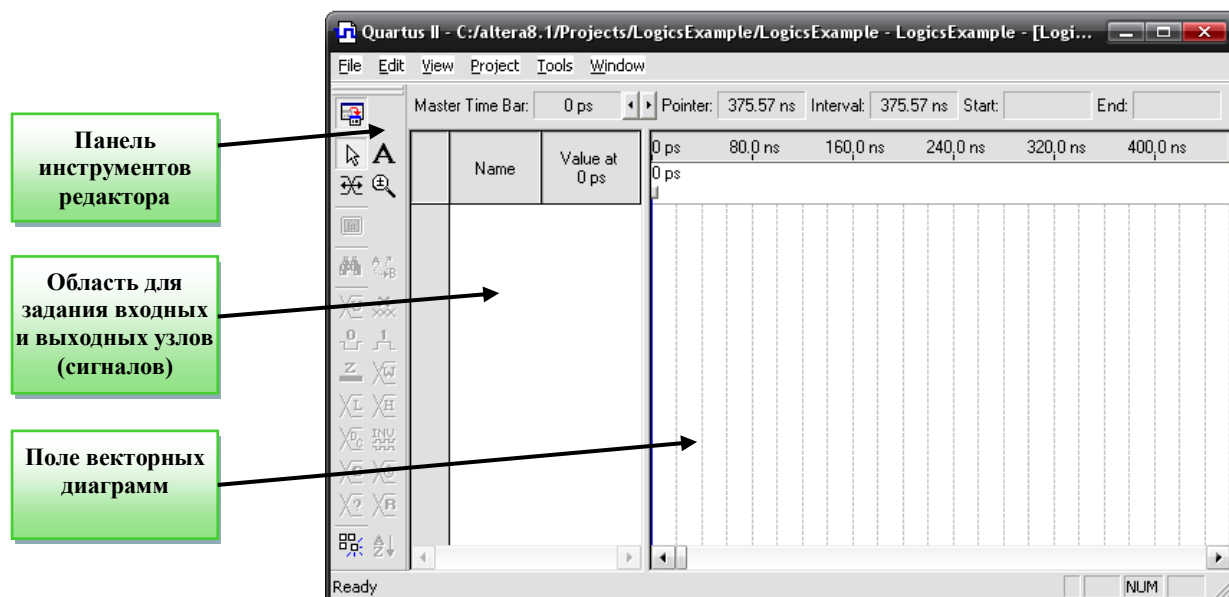


Рисунок 2.35 — Редактор временных диаграмм среды Quartus II

Добавление входных, выходных и промежуточных сигналов.

Далее в созданный файл необходимо ввести входные и выходные сигналы, присутствующие в проекте. Для этого в меню Edit выбираем строку Insert Node or Bus (вставить узел или шину), или же делаем двойной щелчок по области входных и выходных. В результате появляется диалоговое окно, представленное на рисунке 2.36.

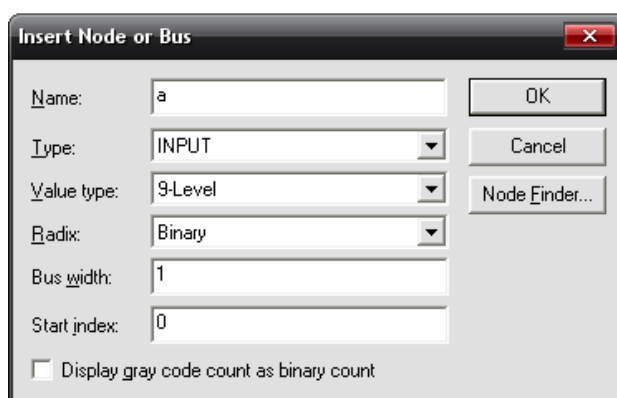


Рисунок 2.36 — Диалоговое окно Insert Node or Bus

В поле Name вписываем имя первого сигнала (например, сигнал порта a). В

следующих полях вписываем параметры сигнала. В поле Type для входных сигналов указываем тип INPUT, для выходных тип OUTPUT, для двунаправленных тип BIDIR. В поле Value type (тип значения) указываем 9-Level. Этот тип определяет набор значений, которые может принимать проводник, всего 9 значений:

'U' — Uninitialized	(не инициализирован)
'X' — Forcing unknown	(сильное неизвестное состояние)
'0' — Forcing zero	(сильный ноль)
'1' — Forcing one	(сильная единица)
'Z' — Z-state	(z-состояние, оно же третье состояние)
'W' — Weak unknown	(слабое неизвестное состояние)
'L' — Weak zero	(слабый ноль)
'H' — Weak one	(слабая единица)
'-' — Don't care	(не имеет значения)

В языке описания аппаратуры VHDL, которому посвящен следующий раздел данного методического пособия, для представления состояния проводников имеется специальный тип данных `std_logic` (переменная которого может принимать 9 вышеперечисленных значений).

В поле Radix указываем основание системы счисления. В нашем случае указываем binary (двоичная). В поле Bus width (ширина шины) ставим единицу т.к. у нас нет шин и проводники одиночные. В поле Start index оставляем ноль. Нажимаем ОК, и в области входных и выходных сигналов появится имя нашего первого сигнала. Но чтобы не вводить каждый сигнал отдельно можно воспользоваться системой поиска узлов (портов) Node Finder. Нажимаем кнопку Node Finder окна Insert Node or Bus. Смотрите рисунок 2.36. Открывается окно системы поиска узлов проекта, позволяющее ввести в файл временных диаграмм узлы текущего проекта. Окно Node Finder представлено на рисунке 2.37. Нажимаем кнопку List, после чего в поле Node Found отобразятся все узлы проекта. Перемещаем в поле Selected Nodes те узлы, которые необходимы для выполнения моделирования. В нашем примере необходимы все, поэтому нажимаем кнопочку с двойной правой стрелочкой, а затем кнопочку OK. Появляется окно Insert Node or Bus, в котором тоже необходимо

нажать кнопку ОК. Система Node Finder позволяет вести поиск узлов по критериям, используя специальные фильтры. Использование Node Finder делает работу в среде удобной, тем более, когда проект включает в себя сотни узлов. Мы неоднократно будем использовать его и в других редакторах САПР Quartus II.

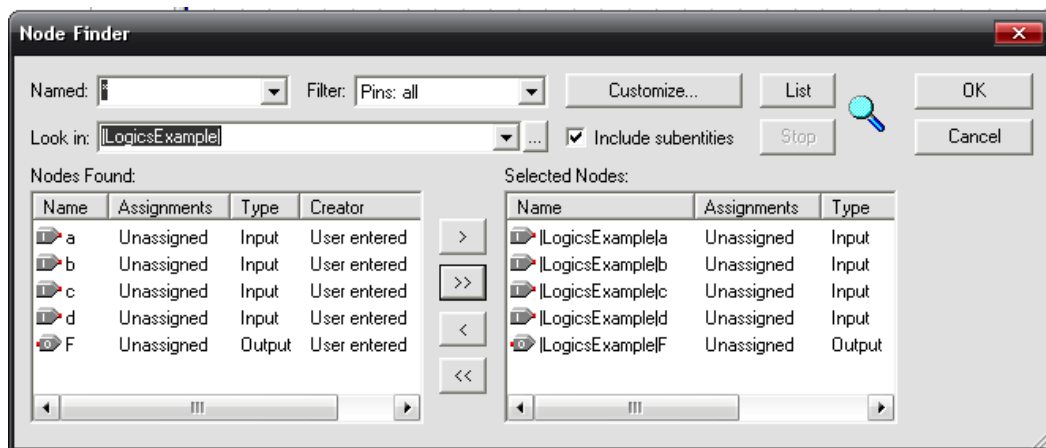


Рисунок 2.37 — Диалоговое окно Node Finder

После этого в поле временных диаграмм редактора появляются оси для всех вышеуказанных сигналов. Осям временных диаграмм входных сигналов задаём требуемые значения. Оси временных диаграмм выходных сигналов имеют неизвестное на данный момент состояние, и поэтому им по умолчанию присвоено значение X (forcing unknown). Редактор векторных диаграмм с узлами и осями векторных диаграмм представлен на рисунке 2.38.

Работа с временными маркерами

На временных диаграммах присутствует вертикальная линия временного маркера, который изображен в виде сплошной цветной линии (главная маркерная линия). Положение этого маркера можно изменять, используя курсор. Значение сигналов, соответствующее текущему положению курсора, отображается в соответствующем столбце рядом с именем узла. Столбец называется Value at XX, где XX — время, соответствующее текущему положению маркерной линии. Если на временных диаграммах необходимо отметить некоторые базовые моменты времени, это можно сделать, используя команду Insert Time Bar (вести временную метку) из

меню Edit. Откроется окно Insert Time Bar показанное на рисунке 2.39.

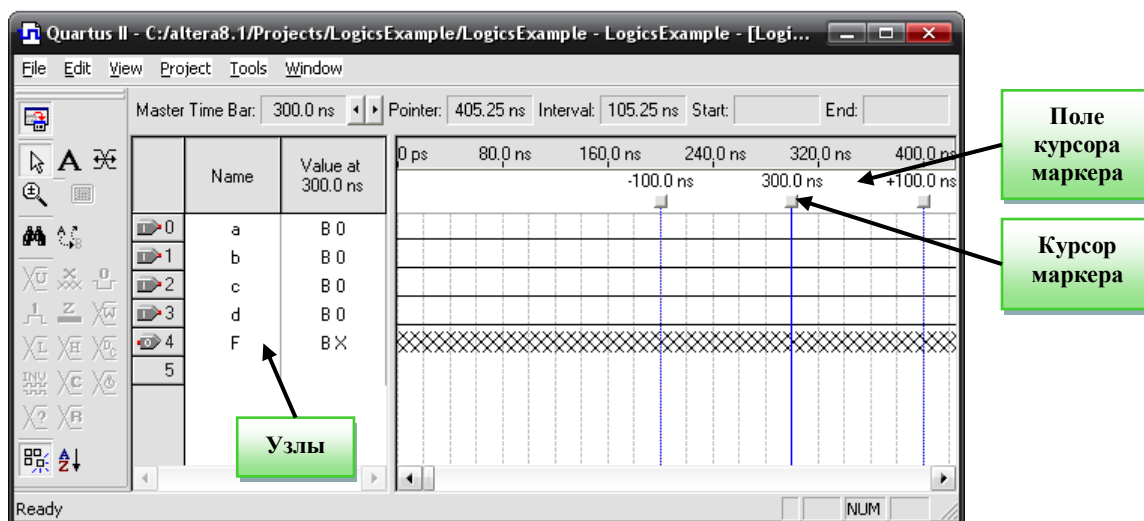


Рисунок 2.38 — Редактор временных диаграмм среды с загруженными узлами

В данном окне необходимо выбрать время, соответствующее базовому моменту времени и единицу его измерения. После нажатия кнопки ОК пунктирная линия (дополнительная маркерная линия), соответствующая введенному времени, появится на временных диаграммах. Эту линию также можно установить, кликнув двойным щелчком по белому полю расположенному над диаграммами (полю курсора маркера).

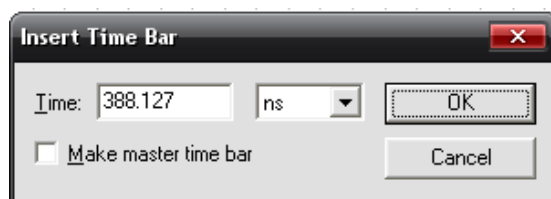


Рисунок 2.39 — Диалоговое окно Insert Time Bar

Теперь при перемещении главной маркерной линии над ней будет отображаться текущее время моделирования, а над линиями базовых моментов времени (дополнительными маркерными линиями), их расстояние (длительность временного интервала) от главной маркерной линии.

При необходимости, любую из введенных линий базовых моментов времени, можно преобразовать в главную маркерную линию. Для этого необходимо правой

кнопкой мыши нажать на курсор маркера расположенного в верхней части линии маркера. Появится выпадающий список, представленный на рисунке 2.40.

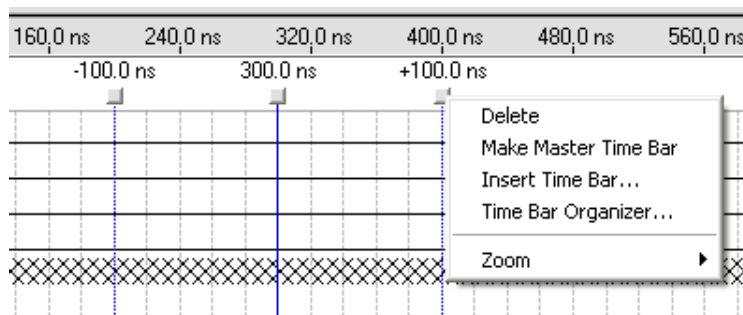


Рисунок 2.40 — Выпадающее окно работы с маркером

Из списка можно выполнить следующие команды:

- ✓ Delete — удалить линию времени,
- ✓ Make Master Time Bar — использовать данную линию как главную маркерную,
- ✓ Insert Time Bar — ввести дополнительную маркерную линию (линию базовых моментов времени),
- ✓ Time Bar Organizer — вызвать окно органайзера временных линий, позволяющее переназначить основные параметры линий времени.
- ✓ Zoom — выполнить масштабирование временных диаграмм.

Рассмотренная методика позволяет ввести в файл временных диаграмм любое количество используемых при моделировании проекта входных и выходных сигналов.

Редактирование временных диаграмм входных сигналов

Следует сразу отметить, что редактированию могут быть подвержены только входные сигналы, присутствующие в текущем проекте. Необходимый для моделирования вектор входных воздействий, вводится путем задания для выбранных временных интервалов логических уровней, соответствующих значениям входной переменной в заданном узле проекта. Для этого необходимо выполнить следующую последовательность действий. Если необходимо задать одно

и то же значение логического сигнала на всем интервале моделирования, необходимо щелкнуть левой кнопкой мыши на требуемом узле, области узлов. В этом случае будет выделена вся ось сигнала. Если необходимо задать значение логического сигнала на определенном интервале моделирования, в области диаграмм курсор необходимо установить на ось требуемого сигнала в точке начала задания логического уровня (сигнала) и, удерживая левую кнопку мыши, переместить курсор в конец требуемого временного интервала. В этой точке кнопку мыши следует отпустить. В результате будет выделен только требуемый временной интервал. После выделения нужной области сигнала в левой части главного окна системы Quartus II появляется набор инструментов для введения входных воздействий, причем, каждый вариант входного воздействия обозначен соответствующей пиктограммой. Панель инструментов графического редактора и расшифровка каждой пиктограммы показаны на рисунке 2.41. Например, если мы хотим ввести тактовый сигнал, то выделяем соответствующий узел и нажимаем пиктограмму панели инструментов Overwrite Clock (сигнал тактирования). Появится диалоговое окно, которое представлено на рисунке 2.42.

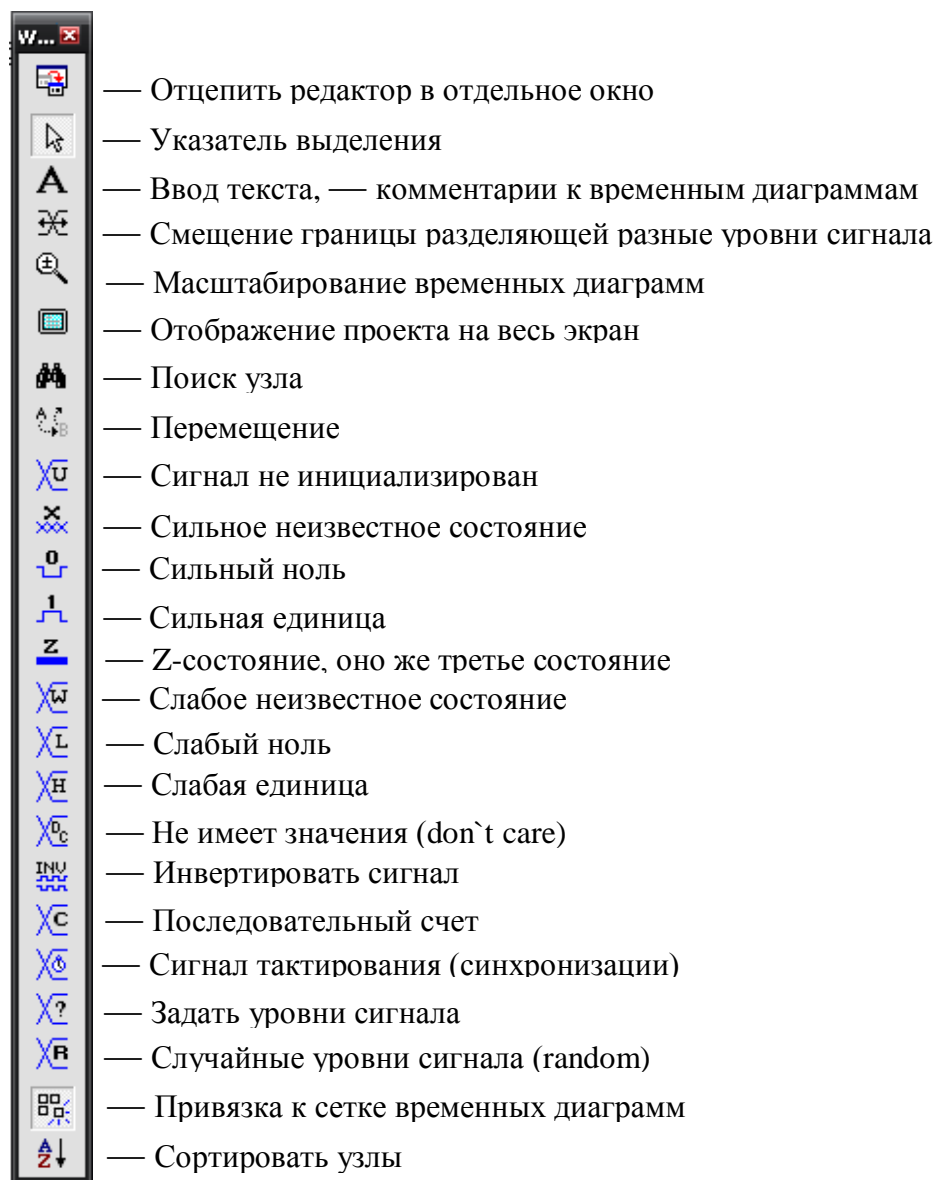


Рисунок 2.41 — Панель инструментов редактора временных диаграмм

В области Time Range в полях Start Time и End Time указывается время начала и окончания сигнала тактирования. По умолчанию временной интервал действия сигнала тактирования определен на всём диапазоне моделирования. Чуть ниже в области Base waveform on в поле Period задаем период сигнала тактирования, в поле Offset — смещение (начальную фазу), в поле Duty Cycle — заполнение цикла «единицей» в процентном соотношении. Ввод сигнала тактирования завершается нажатием кнопки ОК. Аналогичным образом вводятся все необходимые для моделирования входные сигналы. После того как все входные сигналы заданы

необходимо сохранить файл временных диаграмм.

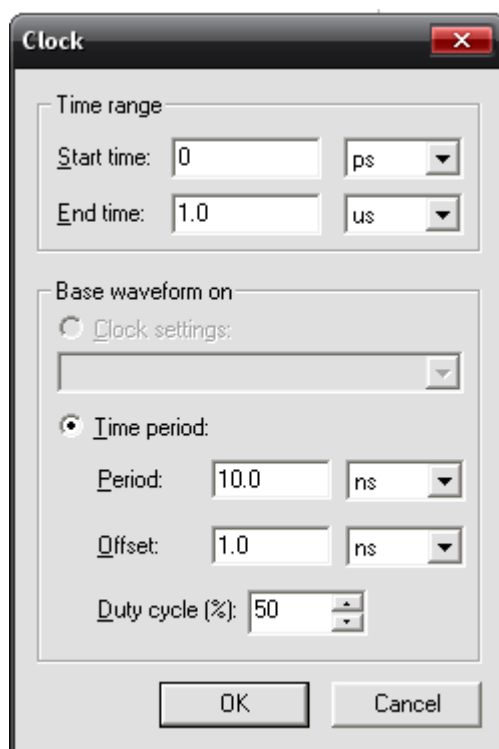


Рисунок 2.42 — Окно настройки сигнала тактирования Clock

2.4.3 Определение параметров моделирования

Система Quartus II позволяет выполнить моделирование, как всего проекта, так и его любой составной части. Типовые параметры по умолчанию задаются системой моделирования автоматически при создании нового проекта. При необходимости, эти параметры можно отредактировать. Для этого в системе Quartus II имеется специальный инструмент — Simulator Tool, вызываемый из меню Tools главной командной строки системы. Окно Simulator Tool представлено на рисунке 2.43.

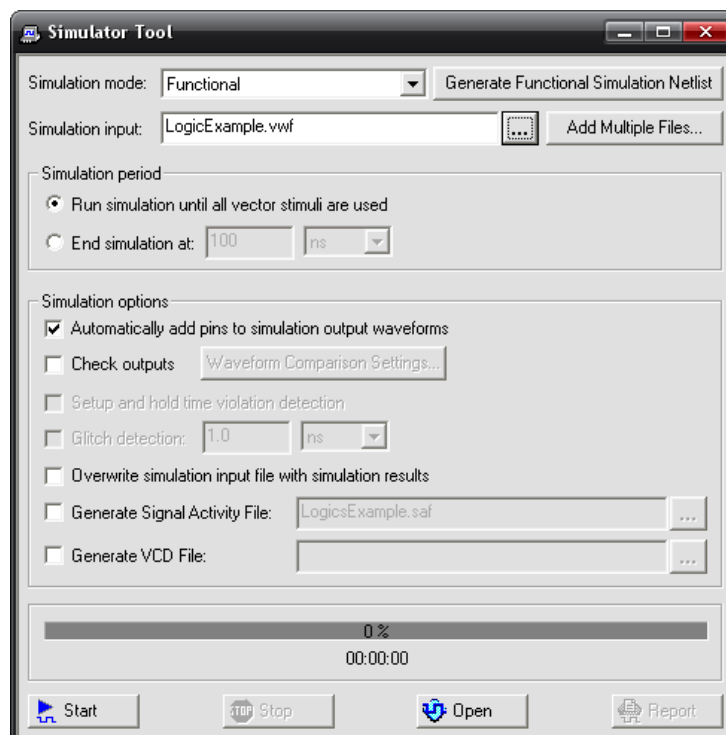


Рисунок 2.43 — Окно настройки симулятора Simulator Tool

В поле Simulation Mode необходимо выбрать тип моделирования проекта — функциональное (Functional) или учитывающее временные параметры выбранного типа ПЛИС (Timing). При указании Timing будет выполнено моделирование с учетом особенностей микросхемы и приближенных реальных временных задержек, которые будут существовать уже в микросхеме.

Если выполняется функциональное моделирования, то перед запуском системы моделирования необходимо создать список соединений проекта. Для этого в окне Simulation Tools необходимо нажать кнопку Generate Functional Simulations Netlist (создать файл со списком соединений для функционального моделирования). После его создания запускаем систему на моделирование. Если выполняется моделирование с учетом временных параметров выбранной ПЛИС, создание файла со списком соединений не требуется и сразу запускается система моделирования.

В следующем поле с названием Simulation Input необходимо ввести имя файла вектора входных воздействий (*.vwf), который мы уже создали. По умолчанию, это

имя файла верхнего уровня проекта. При необходимости, имя требуемого файла можно найти, используя кнопку в правой части окна имени. Область Simulation Period позволяет задать моделирование либо на всем заданном интервале Run simulation until all vector stimuli are used, либо задать время окончания моделирования не совпадающее с временем, определенным в файле временных диаграмм End simulation at. В последнем случае задаются значение времени окончания моделирования и единица его измерения.

Раздел Simulation options позволяет определить следующие опции.

Automatically add pins to simulation output waveform — режим автоматического добавления к временным диаграммам выходных выводов проекта,

Check outputs — режим проверки выходов.

Overwrite simulations input file with simulations results — режим перезаписи исходного файла моделирования с учетом результатов его выполнения. Если этот режим не задан, то файл входных воздействий сохраняется неизменным.

Generate signal activity file — режим генерации файла активных сигналов. В этом случае создается файл с расширением *.saf (Signal Activity File). Это текстовый файл в формате ASCII, содержащий информацию о частоте переключения и данные о статической вероятности для проекта. Этот файл используется при анализе энергетических характеристик проекта модулем PowerPlay Power Analyzer системы.

2.4.4 Запуск на моделирование и отчет о моделировании

Запустить процесс моделирования проекта можно несколькими способами:

- ✓ В окне Simulation Tool нажать кнопку Start.
- ✓ В меню Processing (обработка) вызвать команду Start Simulation (запуск моделирования).
- ✓ Нажать кнопку с соответствующей пиктограммой на панели инструментов главного окна среды Quartus II.

После запуска начинается процесс моделирования, который, в случае успешного выполнения, заканчивается появлением окна, показанного на рисунке 2.44.



Рисунок 2.44 — Сообщение об успешном моделировании проекта

При этом в главном окне Quartus II появляется отчет о моделировании — Simulation Report, правой части которого приведены временные диаграммы, описывающие работу проекта при заданных входных воздействиях. Если для запуска на моделирование использовалась кнопка Start окна Simulation Tool, то после моделирования для вывода отчёта моделирования необходимо будет нажать кнопку Report этого же окна.

Пример отчета о функциональном моделировании (Functional) представлен на рисунке 2.45. Пример отчета о моделировании, учитывающем временные параметры, представлен на рисунке 2.46. Предлагается сравнить временные диаграммы этих рисунков и объяснить их. На временных диаграммах рисунка 2.46, наблюдаются некоторые всплески длительностью меньше 1 нс. Предлагается объяснить их возникновение при моделировании.

Если посмотреть временные диаграммы, то видно, что наш проект в виде простой логической схемы функционирует правильно, — в соответствии с таблицей истинности нашего логического устройства.

Во время моделирования, так же как и при компиляции, работает процессор сообщений, формируя информационные сообщения, предупреждения и сообщения об ошибках.

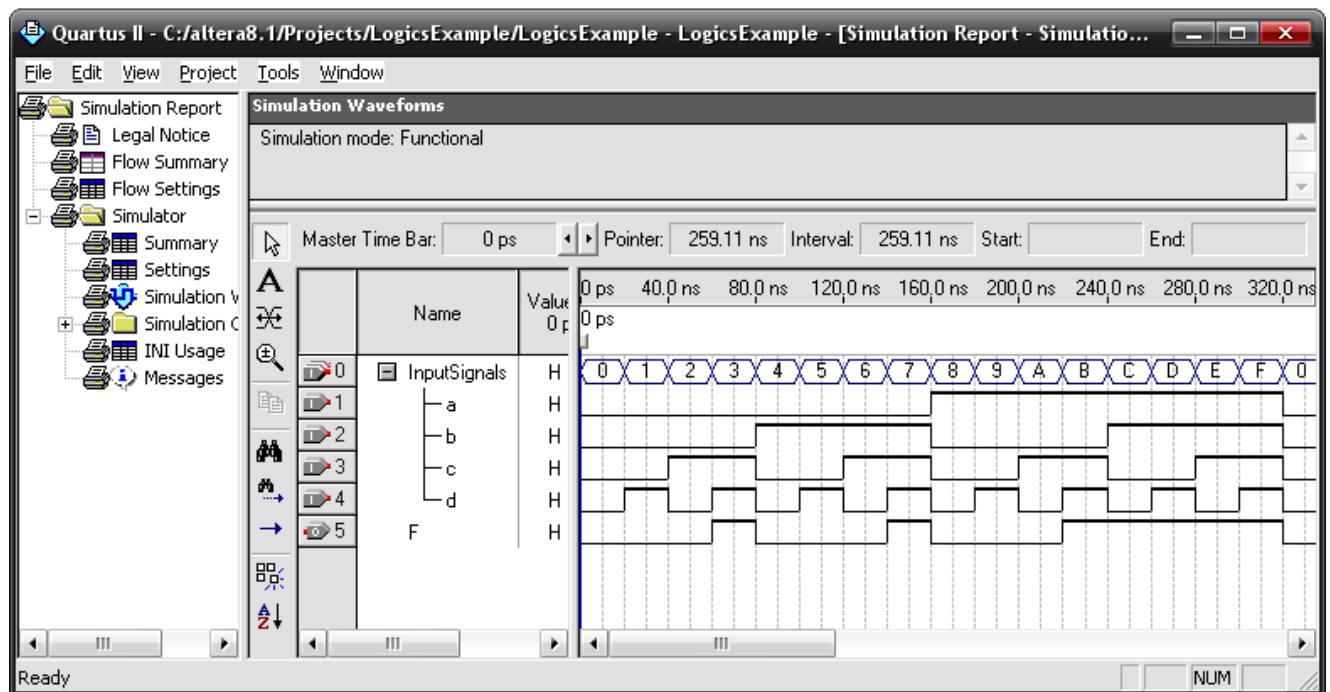


Рисунок 2.45 — Отчет о функциональном моделировании (Functional)

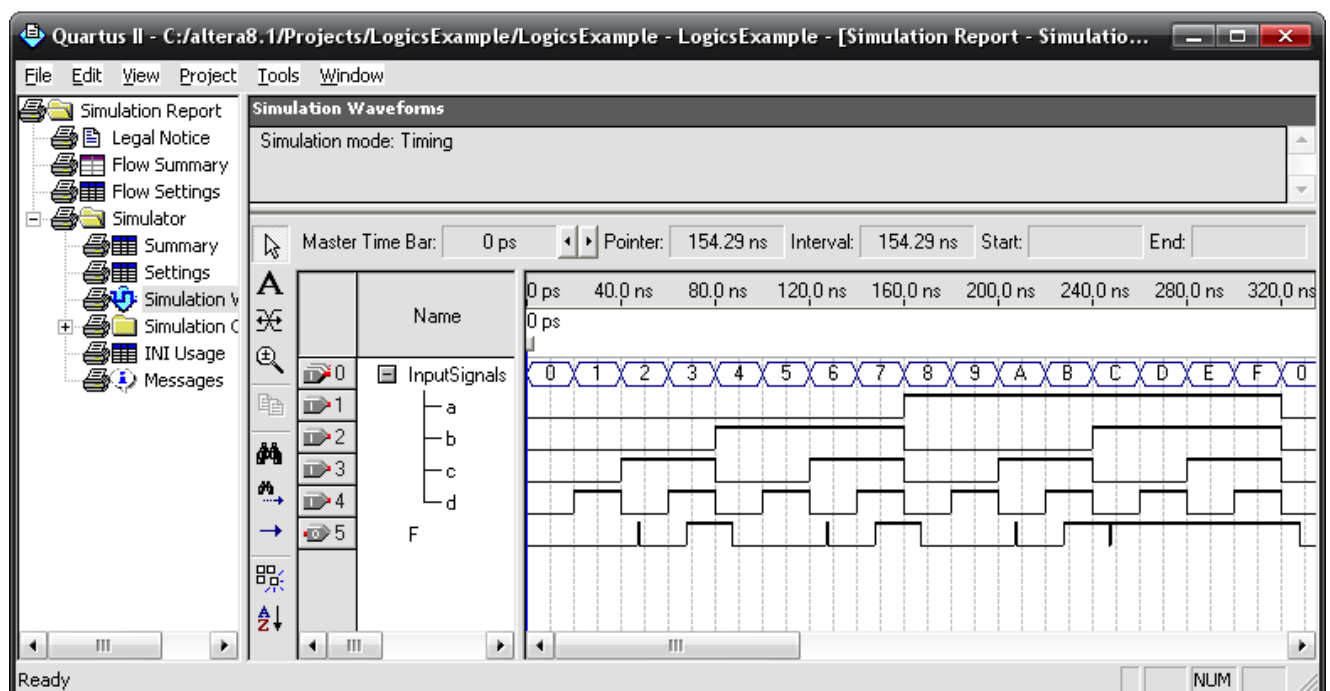


Рисунок 2.46 — Отчет о моделировании, учитывающем временные параметры (Timing)

Если в поле названий сигналов нажать правую кнопку мыши, появится окно, представленное на рисунке 2.47, позволяющее выполнять элементы редактирования.

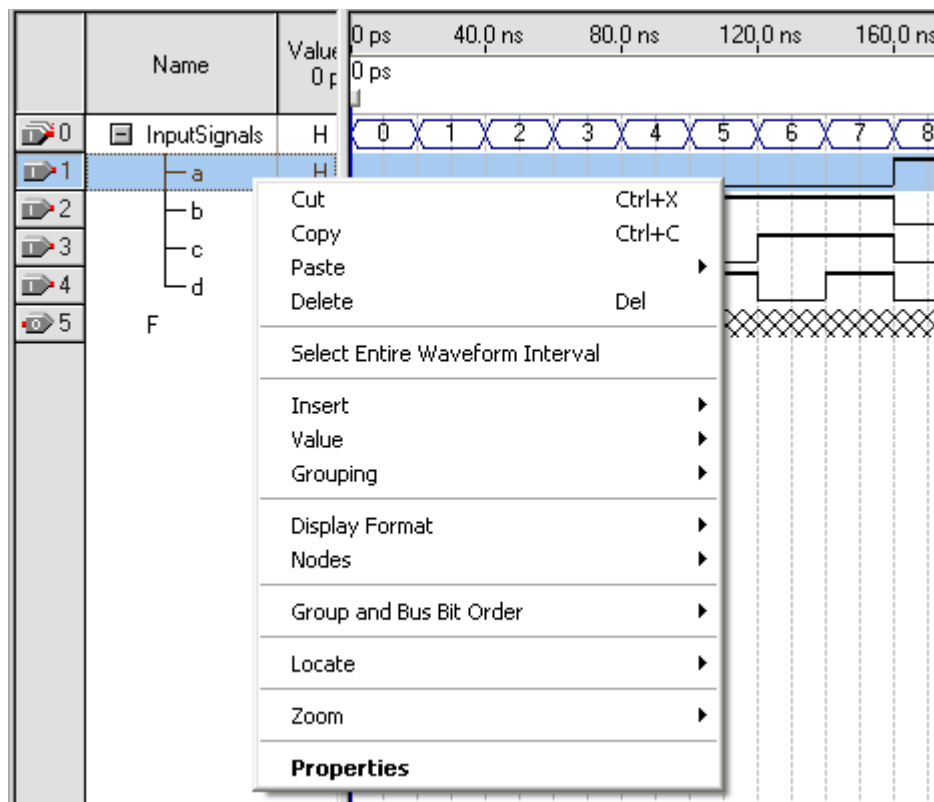


Рисунок 2.47 — Выпадающее окно редактирования сигналов

Доступны операции копирования, удаления создания, групп сигналов и т. д. В нашем примере была создана группа сигналов, названная InputSignals которая включает в себя сигналы a, b, c, d.

После того как правильность функционирования устройства проверена, можно приступить к настройке параметров устройства.

2.5 Задание параметров устройства

2.5.1 Редактор назначений — Assignment Editor

Редактор назначений (Assignment Editor) — это интерфейс, предназначенный для задания требований к разрабатываемому устройству на уровне проекта в целом. Он позволяет предварительно оговорить такие требования к устройству, как требования к его размещению на кристалле ПЛИС, стандарту ввода-вывода информации, временным параметрам, логическим опциям (logic option), параметрам моделирования (симуляции), а так же выполнить предварительное назначение контактов ПЛИС. Общий вид окна редактора назначений приведен на рисунке 2.48.

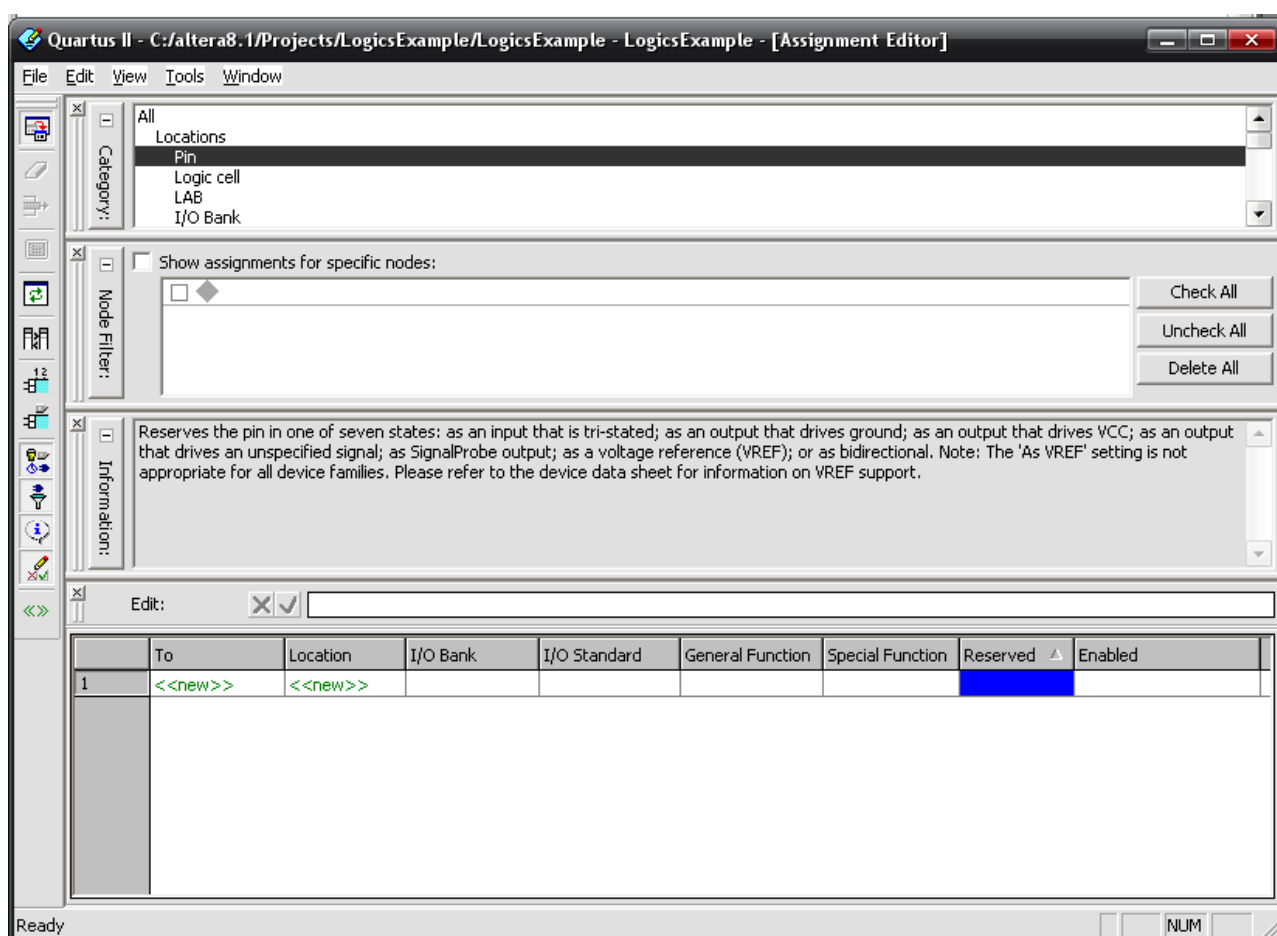


Рисунок 2.48 — Окно редактора назначений

В общем случае окно редактора состоит из пяти областей. Первая область называется Category — позволяет выбрать параметр, изменение или задание которого Вы хотите произвести. Вторая — Node Filter (фильтр узлов) позволяет найти требуемые узлы, с целью проверки их параметров или задания новых параметров. Третья область называется Information, она предназначена для выдачи справочной информации по выбранной категории параметров устройства. Четвертая область, называемая Edit, предназначена для редактирования параметров. В пятой области в виде таблицы (таблицы назначений) отображаются сами узлы и задаваемые параметры устройства. Таблицы назначений в Assignment Editor представляют собой ниспадающие списки с возможностью ввода или выбора необходимой информации о назначениях. По мере ввода, редактирования и удаления назначений.

При желании любая из перечисленных закладок схемы может быть закрыта или вновь открыта с использованием управляющих кнопок, расположенных в левой части окна проекта, на панели инструментов. Панель инструментов редактора назначений представлена на рисунке 2.49.

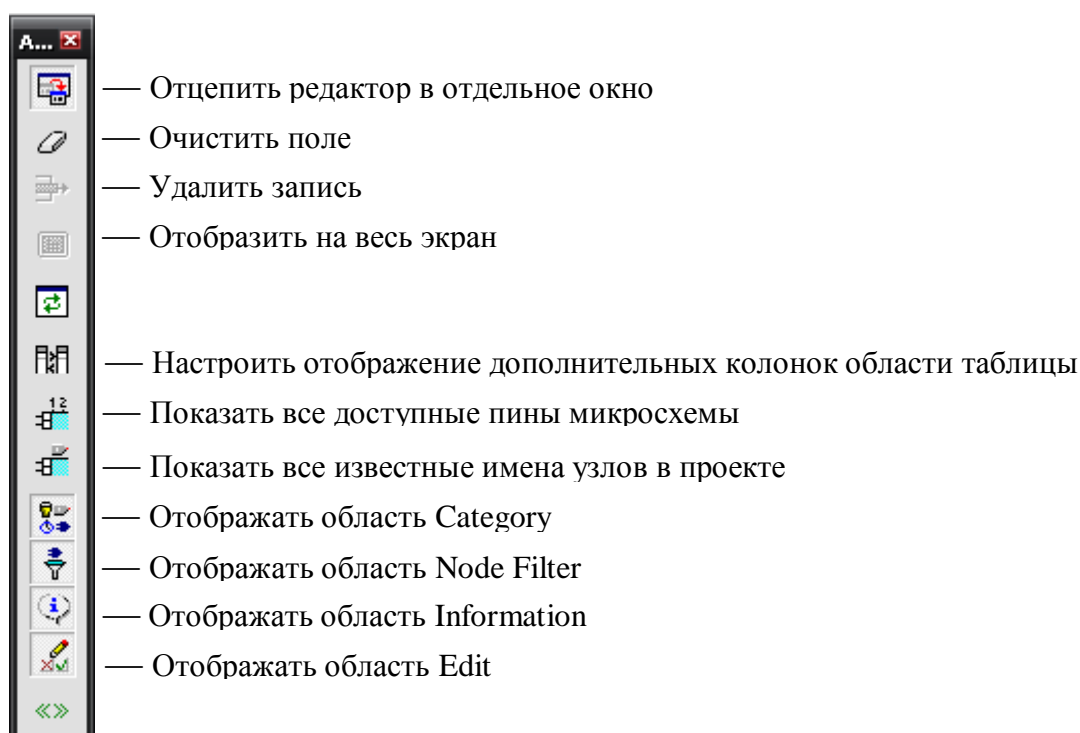
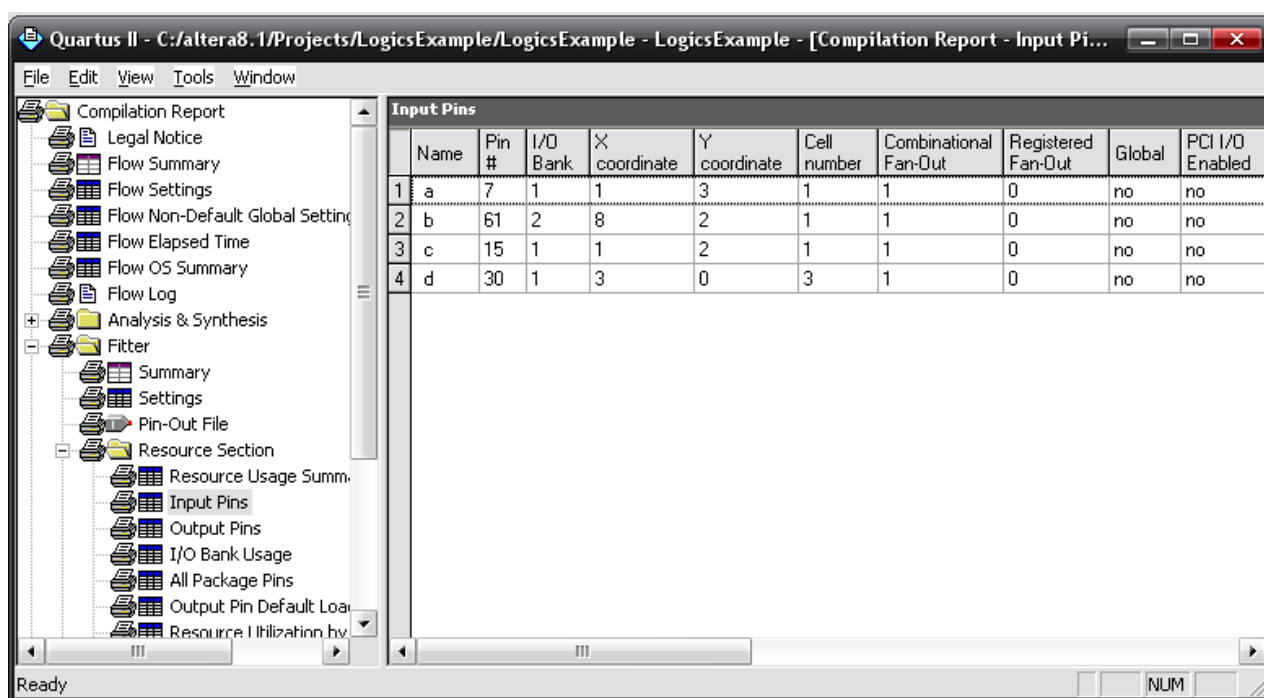


Рисунок 2.49 — Панель инструментов редактора назначений

Если перед выполнением компиляции выводы ПЛИС, на которые необходимо вывести входные и выходные сигналы, не были заданы, компилятор автоматически выполняет соответствующие назначения. Эти назначения можно посмотреть в отчете компилятора. Доступ к этим данным возможен следующим образом Tools → Compiler Tool кнопка Report, далее появится окно, в каталоге которого находим Fitter → Resource Section и далее разделы Input Pins, Output Pins или All Package Pins. На рисунке 2.50. показано окно отчёта о компиляции, отображающее информацию раздела Input Pins. Как видно по умолчанию узлы a, b, c, d закреплены за выводами микросхемы с номерами 7, 61, 15, 30 соответственно.



	Name	Pin #	I/O Bank	X coordinate	Y coordinate	Cell number	Combinational Fan-Out	Registered Fan-Out	Global	PCI I/O Enabled
1	a	7	1	1	3	1	1	0	no	no
2	b	61	2	8	2	1	1	0	no	no
3	c	15	1	1	2	1	1	0	no	no
4	d	30	1	3	0	3	1	0	no	no

Рисунок 2.50 — Элементы отчета о компиляции

Используя отчет можно получить информацию не только о номерах, присвоенных выводам, но и о других параметрах сигналов. Для изменения этой информации будем использовать редактор назначения. Проиллюстрируем работу с редактором на примере назначения выводов ПЛИС.

Запускаем Assignment Editor. В области категорий выбираем раздел Pin. На панели инструментов нажимаем кнопку «показать все известные имена узлов в проекте» и в самой нижней области в виде таблицы отобразятся наши входные и

выходные узлы. Если теперь выделить какую-нибудь ячейку в этой таблице, она становится доступной для редактирования в области Edit. Это происходит только с окнами, параметры которых доступны для редактирования. Также непосредственно отредактировать ячейку можно двойным щелчком по ней, с последующим указанием параметра.

В колонке Location нашей таблицы, необходимо задать соответствующие номера выводов микросхемы ПЛИС, за которыми будут закреплены наши узлы. Двойным щелчком по ячейке находящейся в колонке Location для сигнала *a* (первая строка таблицы) вызываем выпадающий список. В этом списке представлены все доступные для использования номера выводов микросхемы. Выбираем интересующий нас вывод. Смотри рисунок 2.51.

	To	Location	I/O Bank	I/O Standard	General Function	Special Function	Reserved	Enabled
1	a			3.3-V LVTTL				Yes
2	b							Yes
3	c							Yes
4	d	EDGE_BOTTOM						Yes
5	F	EDGE_LEFT						Yes
6	<<new>>	EDGE_RIGHT						Yes
		EDGE_TOP						
		IOBANK_1						
		IOBANK_2						
		PIN_1	I/O Bank 2	Column I/O				
		PIN_2	I/O Bank 1	Row I/O				
		PIN_3	I/O Bank 1	Row I/O				
		PIN_4	I/O Bank 1	Row I/O				
		PIN_5	I/O Bank 1	Row I/O				
		PIN_6	I/O Bank 1	Row I/O				
		PIN_7	I/O Bank 1	Row I/O				
		PIN_8	I/O Bank 1	Row I/O				
		PIN_12	I/O Bank 1	Row I/O	GCLK0p			
		PIN_14	I/O Bank 1	Row I/O	GCLK1p			
		PIN_15	I/O Bank 1	Row I/O				
		PIN_16	I/O Bank 1	Row I/O				
		PIN_17	I/O Bank 1	Row I/O				

Рисунок 2.51 — Назначение узлов определенным выводам микросхемы

Аналогичная настройка выполняется и для других узлов. На рисунке 2.52 показан редактор назначений после выполнения настроек для всех узлов. В конце выполнения всех настроек необходимо сохранить все изменения выбрав в меню File команду Save. Теперь посмотрим в окно графического редактора блок-схем, там для каждого порта будет показана информационная сноска о том, к какому выводу микросхемы подключен данный порт. На рисунке 2.53 показана наша схема после выполнения настроек в редакторе назначений.

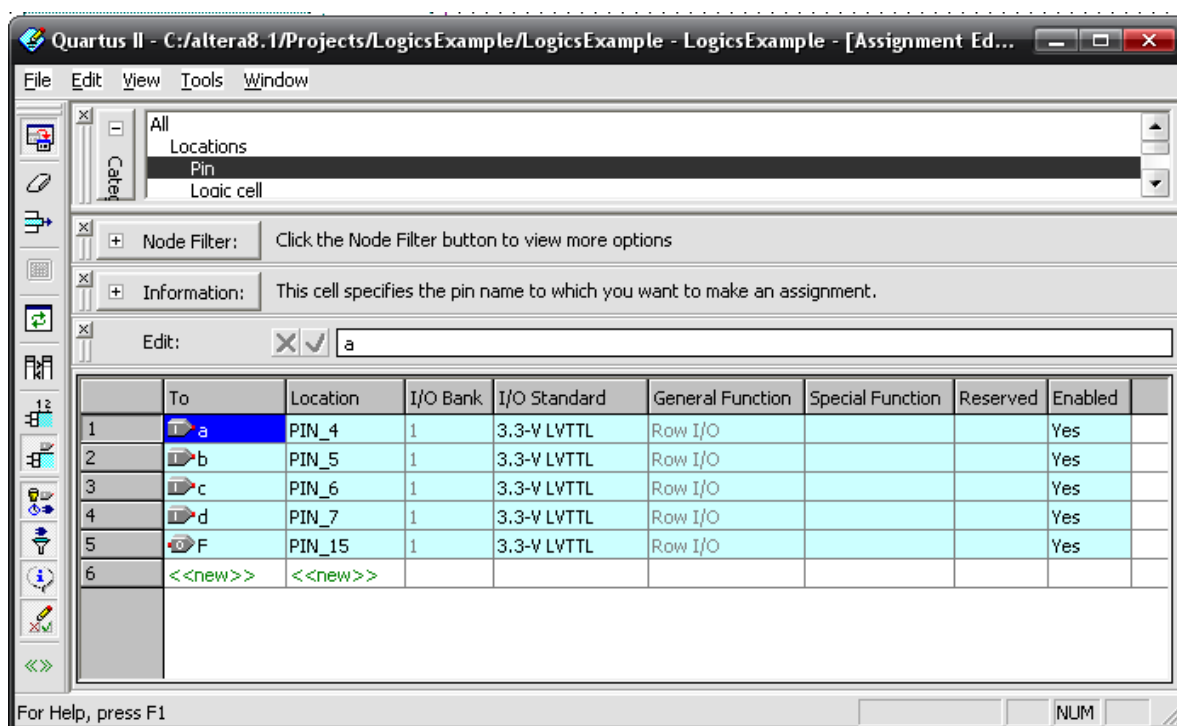


Рисунок 2.52 — Назначение узлов определенным выводам микросхемы

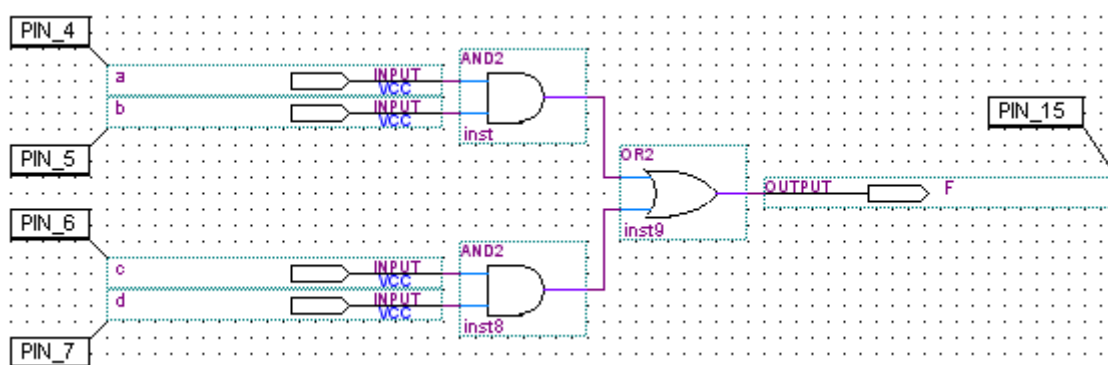


Рисунок 2.53 — Логическая схема с назначенными выводами

У микросхемы есть специализированные выводы. Так, например выводы с номерами 12, 14, 62 и 64 в программируемой логической интегральной схеме MAX II, которую мы будем изучать на лабораторном стенде, являются специализированным и предназначены для введения сигнала синхронизации (GCLK — Global Clock). Шина, связанная с этими выводами, является глобальной и без сегментирования проходит через весь кристалл микросхемы, что обеспечивает

минимизацию задержек срабатывания всех элементов (регистров, триггеров и т.д.) в ПЛИС. Так как данный вывод является узкоспециализированным, при разработке реального устройства его не желательно переназначать не специализированным выводам. Дополнительно в столбцах General Function (основные функции) и Special Function (специальные функции) появляется дополнительная информация об этом выводе. При выборе специализированного вывода, в столбце Special Function отображается его специальное предназначение. При выполнении лабораторных работ, следует учесть, что сигнал тактирования (например, с генератора) назначается на специализированный вывод. Если сделанные назначения не корректны, то при попытке их записи появляется окно-предупреждение. В этом случае следует разобраться и исправить назначение или же снять назначение.

На выводы микросхемы можно настроить следующие опции: открытый коллектор, подтягивающие к питанию резисторы, триггер Шмитта, схему фиксации потенциала, задержки, стандарт напряжения, силу тока и т. д. Например, чтобы установить подтягивающие резисторы, необходимо в области Category редактора назначений выбрать раздел Logic options. В область таблицы узлов, необходимо добавить узлы, для которых хотим сделать подтяжку к питанию. Для этого используем уже хорошо знакомую нам систему поиска узлов — Node Finder. Для вызова Node Finder в колонке с названием To правой кнопкой щелкаем по ячейке. В выпадающем окне выбираем Node Finder, с помощью которого добавляем в таблицу необходимые узлы. Смотри рисунок 2.54. В соответствии с примером на рисунке, для узлов (входов) a, b, c, d (установленных на выводы микросхемы) назначены подтягивающие к питанию резисторы — Weak Pull-Up Resistor. Узел (выход) F сконфигурирован как выход с открытым коллектором — Auto Open-Drain Pins. Причем, чтобы активировать эту опцию в столбце Value для каждого узла необходимо установить значение On. Для назначения одного и того же параметра сразу нескольким узлам используется область Edit редактора назначений. Выделяется диапазон ячеек столбца, и в окне области Edit указывается параметр, который автоматический назначается всем выделенным ячейкам.

Аналогичные процедуры выполняются и при назначении или изменении

других параметров проекта.

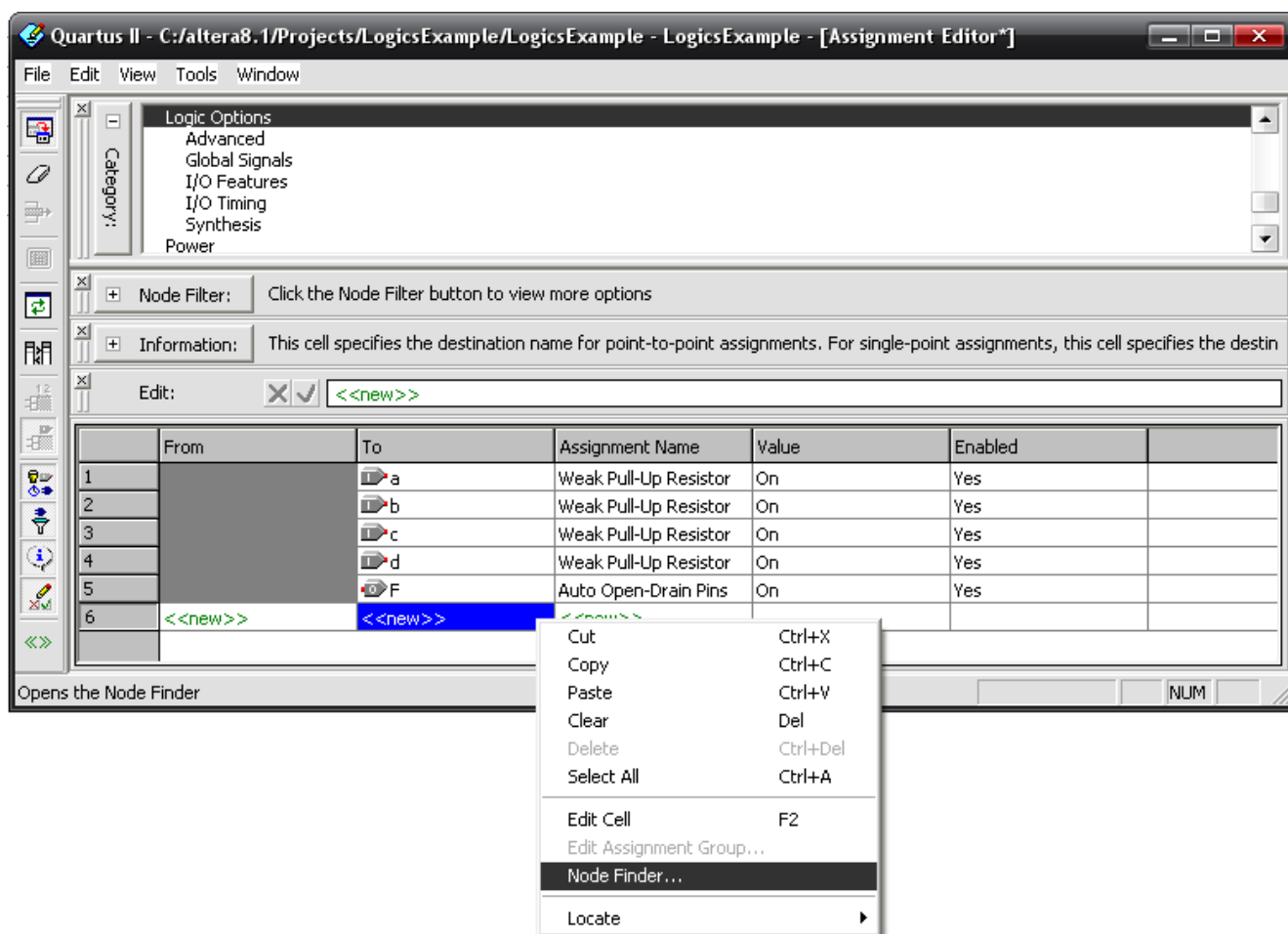


Рисунок 2.54 — Назначение узлам дополнительных настроек

Если в процессе назначений были допущены ошибки, не замеченные системой непосредственно на этапе назначения, то они будут выявлены на этапе компиляции проекта. В этом случае, например при назначении на один и тот же вывод нескольких различных сигналов, компилятор остановит работу и выдаст сообщение о невозможности реализации проекта. Соответствующие указания на ошибки будут отображены в окне процессора сообщений системы.

2.5.2 Планировщик выводов — Pin Planer

Для назначения узлам определенных выводов микросхемы, и задания некоторых других настроек, помимо редактора назначений можно использовать систему Pin Planer (планировщик выводов). Планировщик выводов представлен на рисунке 2.55. Планировщик выводов обеспечивает наглядность при назначении настроек выводам, что в некоторых случаях бывает очень удобно. Удобство заключается в том, что мы видим непосредственно саму микросхему и все недоступные и доступные пользователю выводы, причем двойным щелчком по выводу можно вызвать диалоговое окно для закрепления за этим выводом определенного узла, и назначения на работу с требуемым стандартом напряжения (по умолчанию устанавливается стандарт 3.3-V LVTTL). Выводы ПЛИС в основном окне Pin Planer имеют разные условные обозначения, в зависимости от их функциональной принадлежности. Все возможные условные обозначения выводов представлены на рисунке 2.56.

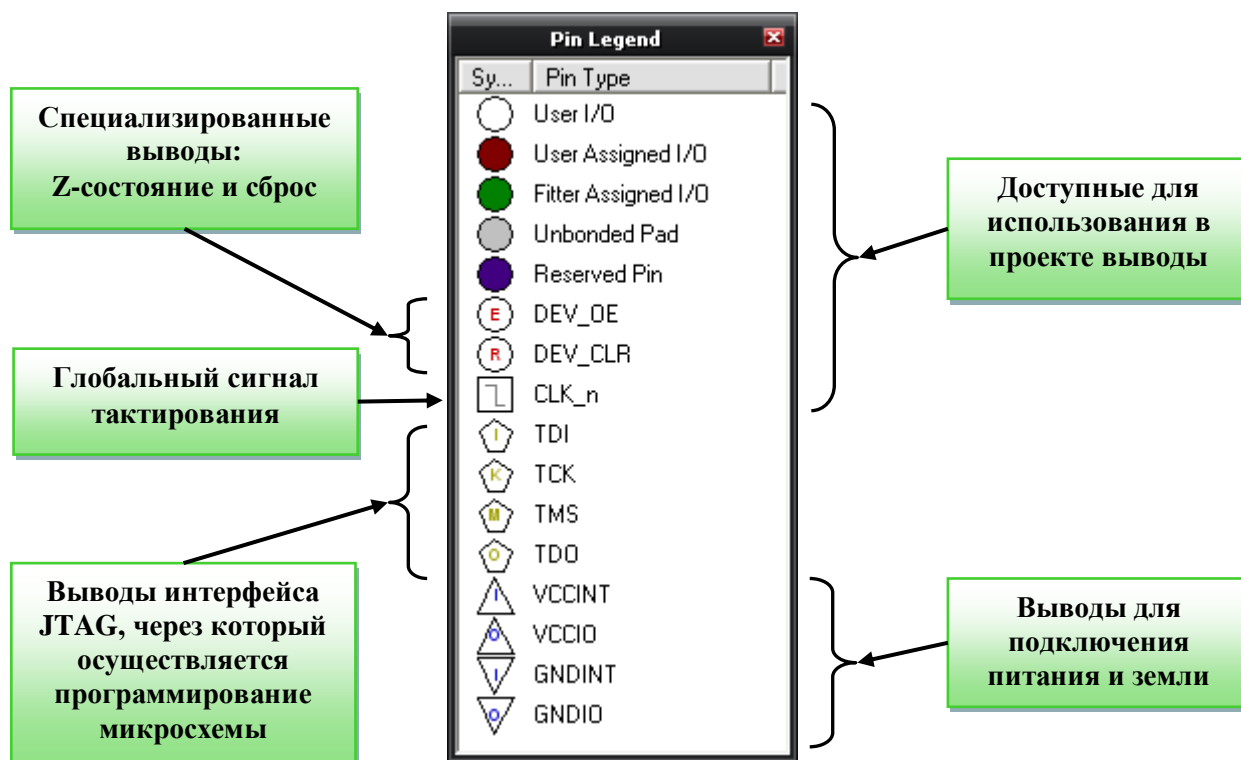


Рисунок 2.56 — Условное обозначение выводов микросхемы

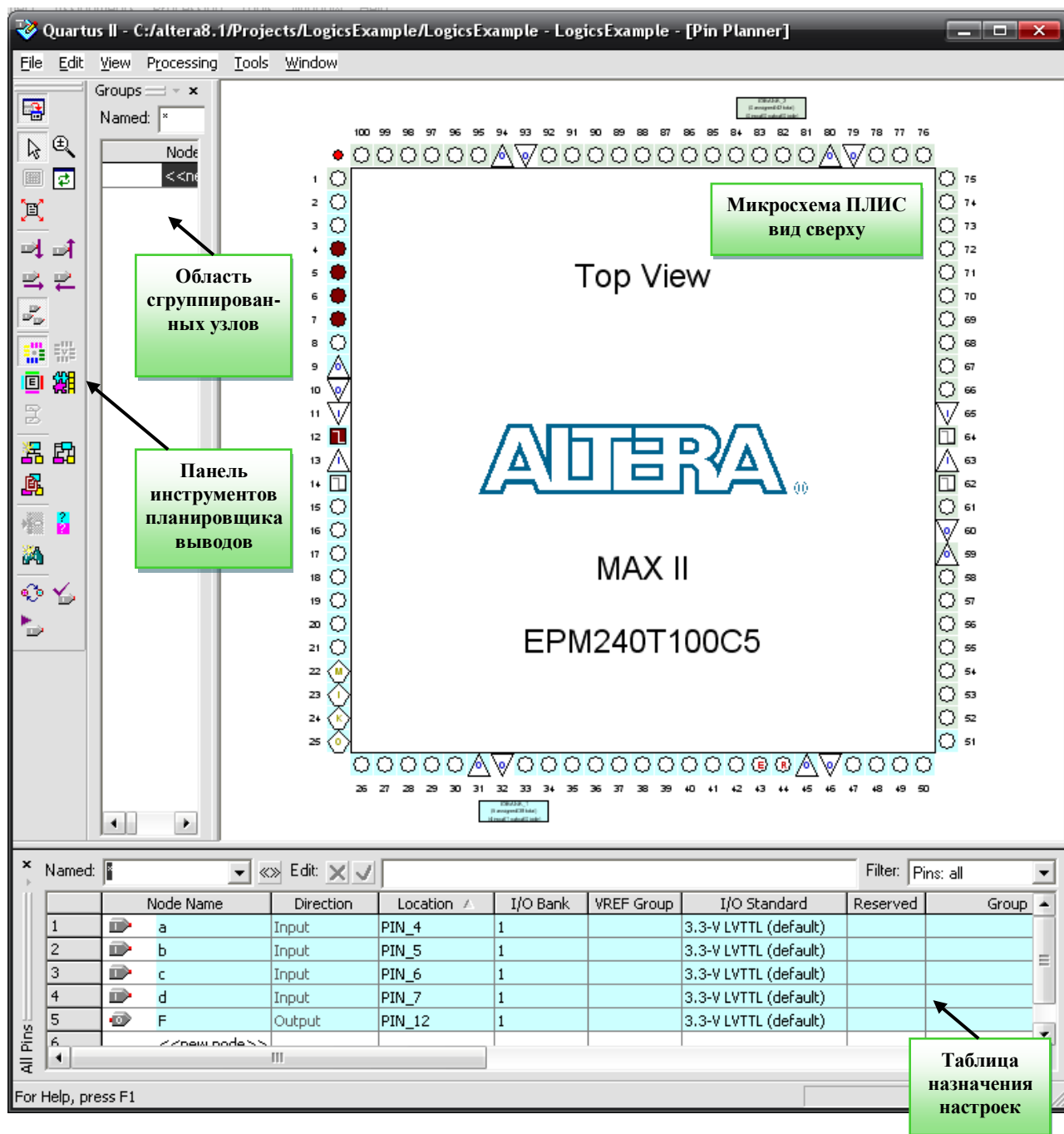


Рисунок 2.55 — Окно Pin Planner

2.6 Конфигурирование микросхемы

2.6.1 Общие сведения о программировании ПЛИС

Завершающим этапом изготовления специализированной БИС на основе ПЛИС является загрузка конфигурации в микросхему.

На сегодняшний день ведущие производители ПЛИС выпускают микросхемы по технологии EEPROM или FLASH, а также по технологии статического ОЗУ (SRAM), программируемые непосредственно на печатной плате (ISP — in system programmable) благодаря встроенному порту стандарта JTAG (IEEE Std. 1149.1 Joint Test Action Group). Программирование и стирание таких ПЛИС осуществляется через специальные загрузочные кабели, обычно подключаемые к последовательному или параллельному портам компьютера.

ПЛИС, изготовленных по технологии статического ОЗУ сохраняют конфигурацию только при наличии напряжения питания и выдерживают неограниченное число циклов перепрограммирования. Поэтому при применении таких устройств необходимо предусмотреть возможность загрузки конфигурации в ПЛИС после включения устройства. Обычно для таких микросхем ПЛИС используется внешний flash-конфигуратор (микросхема flash-памяти, хранящая конфигурацию ПЛИС). Во время включения питания ПЛИС начинает считывать конфигурацию из микросхемы внешней flash-памяти.

Микросхемы ПЛИС выполненные по технологии EEPROM или FLASH, хранят свою конфигурацию в конфигурационной flash-памяти, которая реализована на кристалле ПЛИС.

Как правило, существуют два протокола процедуры конфигурирования: активный и пассивный. В активном режиме данные хранятся в последовательном или параллельном ПЗУ или флэш-памяти, а процессом конфигурации управляет сама ПЛИС. В пассивном режиме функции загрузки обеспечивает внешнее интеллектуальное устройство (процессор или микроконтроллер), частным случаем

которого может быть и сам компьютер. К специализированным аппаратным средствам для загрузки конфигурации в ПЛИС с компьютера относятся загрузочные кабели фирмы Altera: Master Blaster, Byte Blaster MV, Byte Blaster II, USB-Blaster, Ethernet Blaster.

2.6.2 Программирование в среде Quartus II

Во время компиляции модуль **Assembler** системы Quartus II генерирует файлы, которые модуль программирования Quartus II (**Programmer** — о котором будет сказано чуть позже) использует для конфигурирования устройства. Ассемблер автоматически преобразует результаты компоновки логических ячеек и назначения контактов в программный образ устройства в виде одного или нескольких объектных файлов **Programmer Object** (.pof) или объектных **SRAM** файлов (.sof). При этом модуль **Assembler** системы Quartus II можно запускать либо в рамках полной компиляции проекта, либо запускать его отдельно.

Модуль **Programmer** использует **POF** и **SOF** файлы, сгенерированные ассемблером для программирования или конфигурирования всех устройств Altera, поддерживаемых САПР Quartus II. Теперь рассмотрим, как выполняется программирование ПЛИС MAX II в САПР Quartus II.

Для подготовки к программированию или конфигурированию ПЛИС необходимо открыть окно программатора. Для этого в меню **Tools** выбираем пункт **Programmer**. Окно **Programmer** представлено на рисунке 2.57. Если компиляция проекта была выполнена успешно, то обязательно должен быть сгенерирован файл конфигурации, в нашем случае файл с расширением *.pof. При запуске программатора, по умолчанию файл конфигурации должен отобразиться в основном окне программатора в виде строки. А также дополнительно будет отображено еще две строки с названием **CFM** (**Configuration Flash Memory**) и **UFM** (**User Flash Memory**). Для хранения своей конфигурации ПЛИС MAX II использует конфигурационную память **CFM**, в неё мы и будем загружать наш проект. Память