Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра электронных вычислительных средств

ПРОЕКТИРОВАНИЕ ЭВС С ДИНАМИЧЕСКИ РЕКОНФИГУРИРУЕМОЙ АРХИТЕКТУРОЙ

Лабораторный практикум
для студентов специальности 1-40 02 02
«Электронные вычислительные средства»
дневной формы обучения

Минск БГУИР 2008

УДК 681.3.06 (076)
ББК 32.973.26-02 я7
П 79

Р е ц е н з е н т
канд. техн. наук В. Н. Левкович

А в т о р ы:
А. А. Петровский, А. Е. Новиков, Д. А. Машеров, М. И. Вашкевич,
М. З. Лившиц, Д. С. Лихачев

П 76 **Проектирование** ЭВС с динамически реконфигурируемой архитектурой : лаб. практикум для студ. спец. 1-40 02 02 «Электронные вычислительные средства» днев. формы обуч. / А. А. Петровский [и др.]. – Минск : БГУИР, 2008. – 64 с. : ил.
ISBN 978-985-488-318-2

Представлены краткие теоретические сведения о лабораторной установке на базе отладочного модуля Xilinx ML401.

Темы лабораторных работ отражают основные положения курса «Проектирование ЭВС с динамически реконфигурируемой архитектурой». Особое внимание уделяется синтезу устройств, реализующих базовые алгоритмы цифровой обработки сигналов с использованием распределенной арифметики. Лабораторный практикум содержит пять лабораторных работ, в каждой из которых приведены краткие теоретические сведения и рекомендации по выполнению.

УДК 681.3.06 (076)
ББК 32.973.26-02 я7

ISBN 978-985-488-318-2

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

FPGA (field programmable gate arrays), или ПЛИС, представляют собой цифровые интегральные схемы, состоящие из программируемых логических блоков и программируемых соединений между этими блоками. Возможность конфигурировать эти устройства позволяет инженерам-разработчикам решать множество различных задач.

Словосочетание «field programmable», содержащееся в расшифровке аббревиатуры FPGA, означает, что программирование FPGA-устройств выполняется на месте (в отличие от устройств, внутренняя функциональность которых жестко прописана производителем), т.е. возможность модификации функций устройства в лабораторных условиях.

ПЛИС занимают промежуточное положение между программируемыми логическими устройствами (ПЛУ) и заказными интегральными схемами. С одной стороны, их функциональность может быть задана непосредственно на месте в соответствии с требованиями заказчика-пользователя. С другой стороны, они могу содержать миллионы логических вентилей и, следовательно, реализовывать чрезвычайно большие и сложные функции, которые изначально могли быть реализованы только с помощью заказных интегральных микросхем.

Стоимость ПЛИС намного ниже стоимости заказных интегральных схем (хотя окончательная версия заказной микросхемы при массовом производстве оказывается более дешевой). В случае использования ПЛИС внесение изменений в устройство не вызывает особых затруднений и существенно сокращаются сроки выхода таких устройств. Все это делает ПЛИС привлекательными не только для крупных разработчиков, но и для небольших новаторских конструкторских бюро.

ЛАБОРАТОРНАЯ УСТАНОВКА

В качестве лабораторной установки используется отладочная плата Xilinx ML401 (рис В.1).

Отладочная плата характеризируется следующими параметрами:

• используемая ПЛИС Virtex-4: XC4VLX25-FF668-10;

• 64-Мб DDR SDRAM с 32-битным интерфейсом, работающим на частоте передачи данных до 266 МГц;

• одна входная дифференциальная пара синхронизации и выходная дифференциальная пара синхронизации с разъёмами SMA;

• один генератор синхроимпульсов с частотой 100 МГц (с панелью) и один дополнительный генератор синхроимпульсов на 3,3 В с открытой панелью;

• DIP-переключатели общего назначения, светодиоды и кнопки;

• разъёмы расширения с 32 несимметричными контактами ввода–вывода, 16 LVDS-совместимыми дифференциальными парами, 14 дополнительными контактами ввода–вывода, подключенными к кнопкам и светодиодам, разъёмом для подключения питания, средствами расширения JTAG-цепочки, а также расширителем IIC-шины;

• аудиокодек Stereo AC97 с линейным входом и выходом, 50 мВт наушниками и гнездами (моно) для микрофона;

• последовательный порт RS-232;

• ЖКИ на две строки по 16 символов в каждой;

• один 4-Кбитный IIC EEPROM;

• выход VGA: 50 МГц/24-битный видео ЦАП;

• 2 порта PS/2 для подключения мыши и клавиатуры;

• конфигурационный контроллер System ACE Compact Flash с разъёмом Type I/II Compact Flash.

• синхронная ZBT SRAM: 9 Мбит SRAM на 32-битной шине данных с четырьмя битами четности;

• микросхемы Intel Strata Flash ёмкостью 8 Мб (или совместимые с ними);

• трёхрежимный (10/100/1000) приёмопередатчик Ethernet PHY;

• микросхема USB-интерфейса (Cypress CY7C67300) с основным (host) и периферийными портами;

• Xilinx CPLD XC95144XL, позволяющая использовать микросхемы флэш-памяти для конфигурирования FPGA Virtex-4;

• память для хранения конфигурации: Xilinx XCF32P Platform Flash;

• конфигурационный порт JTAG, позволяющий использовать кабели Parallel Cable III или Parallel Cable IV;

• встроенные источники питания для всех используемых значений напряжения;

• адаптер переменного тока напряжением 5 В и силой тока 3 А;

• светодиодный индикатор питания.

Рис. B.1. Отладочная плата Xilinx ML401 (лицевая сторона)

На рис. B.2 показана блок-схема отладочной платформы ML401.

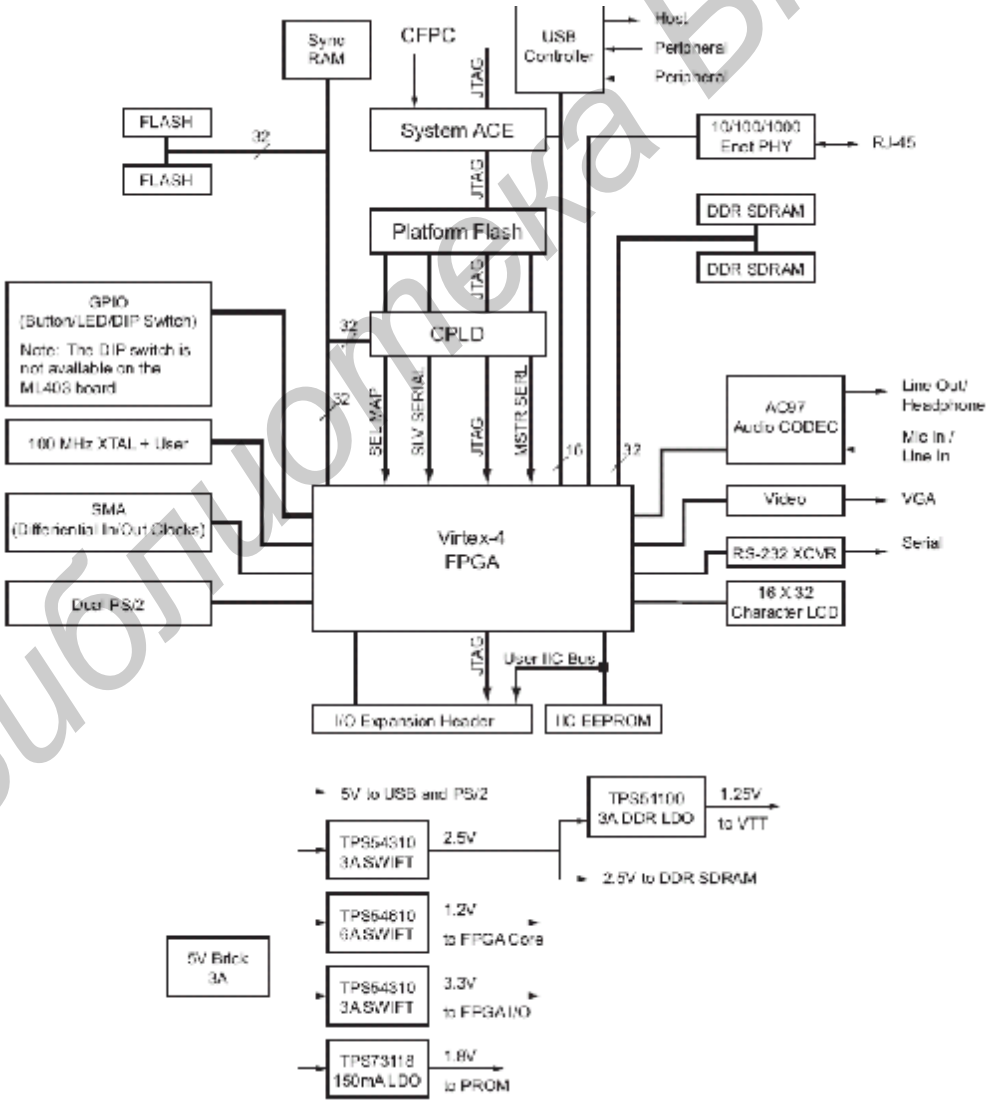

Рис. B.2. Блок-схема отладочной платформы Virtex-4 ML401

Наиболее простым способом обмена данными между персональным компьютером и отладочным модулем ML401 является использование последовательного интерфейса – COM-порта (COMmunication port). COM-порт обеспечивает асинхронный обмен по стандарту RS-232. Реализуются COM-порты на микросхемах универсальных асинхронных приемопередатчиков (UART – Universal Asynchronous Receiver Transmitter). UART обеспечивает полную дуплексную связь по последовательным линиям интерфейса RS-232 (рис. В.3).

Физически в контроллер интерфейса RS-232 входят две линии ***Rx*** и ***Tx*** – приемника и передатчика соответственно. В состоянии покоя на линиях выставляется уровень логической единицы. Данные передаются последовательно на заданной скорости обмена. О начале передачи сигнализирует старт-бит, во время которого линия переводится с состояние логического нуля, далее следуют 8 передаваемых бит (начиная с младшего), завершает передачу стоп-бит, гарантирующий паузу между посылками. Формат посылки приведен на рис. В.4.

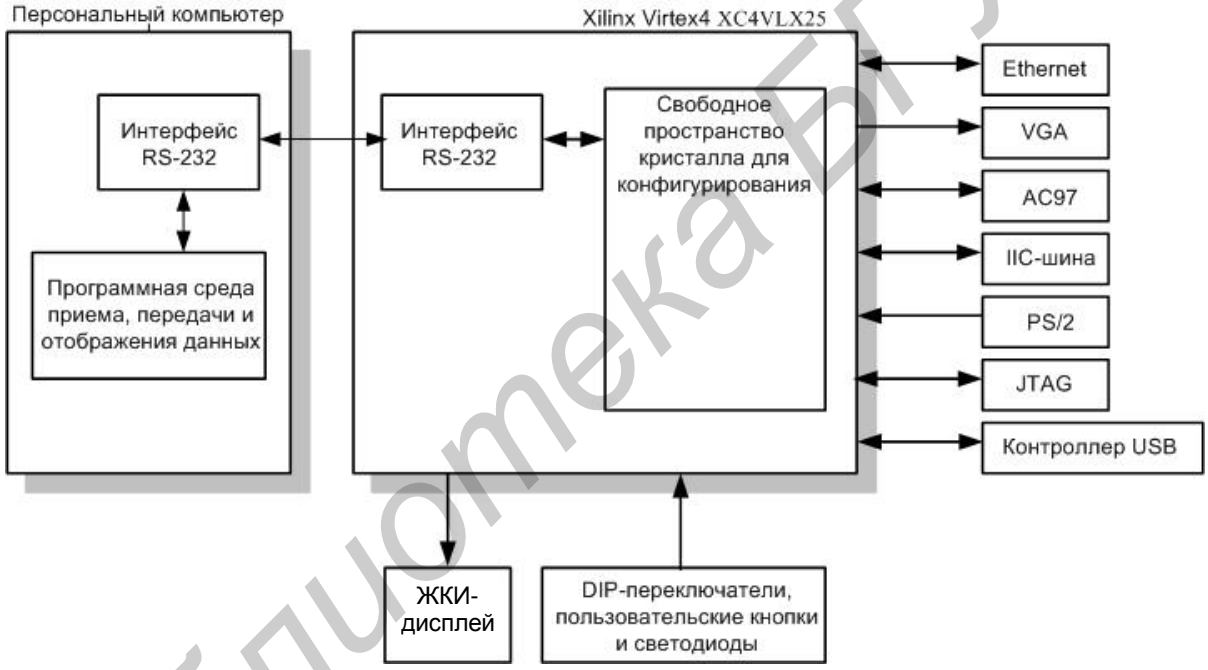

Рис. В.3. Обмен данными между пользователем и отладочной платой посредством интерфейса RS-232

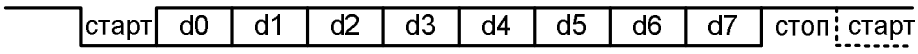

Рис. В.4. Формат передачи в последовательном интерфейсе

Контроллер интерфейса RS-232 позволяет производить обмен данными на заданной скорости между выполняющимся на FPGA пользовательским приложением и персональным компьютером. COM-порт имеет ряд стандартных скоростей обмена: 115 200, 57 600, 38 400, 19 200, 9 600, 4 800, 2 400, 1 200, 600, 300, 150 бит/с. Интерфейс контроллера приведен на рис. В.5.

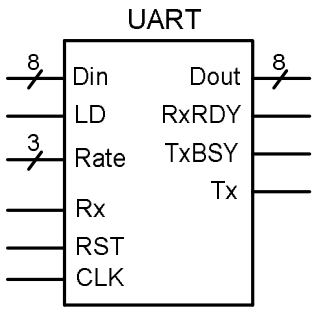

Рис. В.5. Контроллер последовательного порта

Описание входных и выходных сигналов контроллера сведено в табл. В.1.

Таблица В.1

Назначение сигналов контроллера UART

Название	Назначение сигнала
Din[7:0]	8 бит данных для пересылки в ПК
LD	Строб для записи данных со входа Din в контроллер. Длительность сигнала равна одному периоду CLK
Rate[2:0]	3 бита, управляющие скоростью обмена. Поддерживаются скорости обмена от 115 200 до 1 200 бит/с. Код 000 соответствует скорости 115 200, 001 – скорости 57 600 и т.д.
Rx	Вход приемника
RST	Асинхронный сброс, активный высокий уровень
CLK	Тактовый сигнал (100 МГц)
Dout	8 бит данных, переданных от ПК
RxRDY	Сигнал длительностью один период CLK появляется, когда контроллер принял данные от ПК, находящиеся на выходе Dout
TxBSY	Устанавливается в «1» на весь цикл передачи данных, если в это время приходит сигнал LD, то он будет проигнорирован
Tx	Выход передатчика

Типичная схема использования контроллера UART для связи отладочного модуля ML401 с ПК показана на рис. В.6.

Из рис. В.6 видно, что для передачи данных от приложения к ПК через контроллер UART необходимо выставить данные на выход ADout и стробировать их сигналом LD_A. Полученные от ПК данные приходят на вход ADin и стробируются сигналом RxRDY.

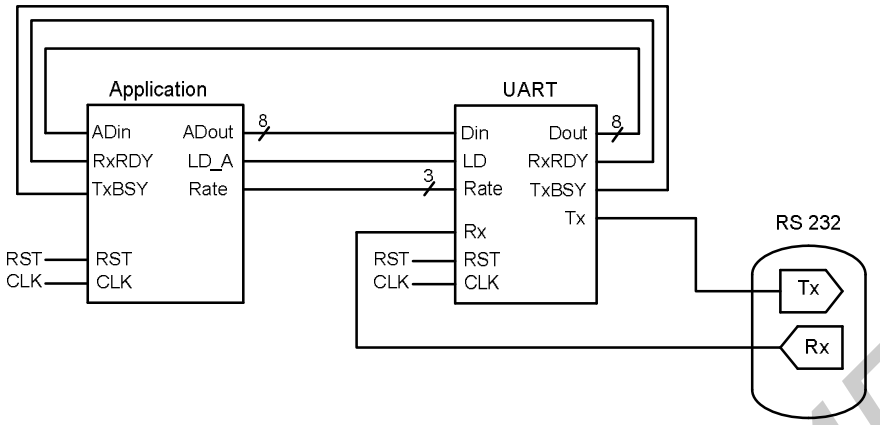

Рис. В.6. Схема подключения контроллера UART

Для использования контроллера необходимо подключить VHDL-файлы с описанием контроллера к проекту. Затем в головном файле прописать контроллер как компонент схемы, как показано ниже:

```
component UART
    Port ( Din : in  STD_LOGIC_VECTOR (7 downto 0);
           Rx  : in  STD_LOGIC;
           LD  : in STD_LOGIC;
           Rate: in STD_LOGIC_VECTOR (2 downto 0);
           clk : in  STD_LOGIC;
           rst : in  STD_LOGIC;
           Dout: out STD_LOGIC_VECTOR (7 downto 0);
           RxRDY: out STD_LOGIC;
           TxBSY: out STD_LOGIC;
           Tx :  out STD_LOGIC);
end component;
```

Затем при помощи оператора *port map* назначить входам и выходам контроллера нужные сигналы. Далее при помощи утилиты *Assign Package Pins* назначить входу приемника *Rx* и выходу передатчика *Tx* выводы микросхемы *FPGA – W2* и *W1* соответственно.

Для приема и отправки данных по интерфейсу RS-232 со стороны ПК можно использовать программу HyperTerminal, входящую в состав Microsoft Windows. Также для этих целей подходит специально разработанная программа RS-232 tool, основные возможности которой включают прием и отправку данных через выбранный COM-порт (номера от 1 до 32), выбор скорости обмена, сохранение принятых данных в файл.

1. ЛАБОРАТОРНАЯ РАБОТА №1 КОНФИГУРИРОВАНИЕ ОТЛАДОЧНОЙ ПЛАТЫ ML401 ЧЕРЕЗ КОНТРОЛЛЕР SYSTEM ACE В СРЕДЕ XILINX ISE 8.1

Цель работы: изучить основные этапы конфигурирования отладочной платы ML401 через контроллер System ACE в среде Xilinx ISE 8.1.

1.1. Теоретические сведения

1.1.1. Основы работы в среде Xilinx ISE 8.1, создание проекта

Integrated Software Environment (ISE) – основная САПР ПЛИС фирмы Xilinx, обеспечивающая весь процесс проектирования цифровых устройств на базе ПЛИС от ввода исходного описания до программирования микросхем.

Версия 8.1 ISE поддерживает следующие семейства ПЛИС:

- Virtex/Virtex-E/Virtex-ll/Virtex-ll Pro/Virtex-4;
- Spartan-1 I/Spartan-11E/Spartan-3/Spartan-3E;
- XC9500/XC9500XL/XC9500XV;
- CoolRunner/CoolRunner-ll.

САПР ISE работает под управлением ОС Windows 2000 + SP2 или Windows XP. Кроме того, возможна работа под управлением Solaris 2.8 и Linux (Red Hat Enterprise 3).

Все конфигурации ISE имеют интерфейсы к САПР Synplify и Leonardo Spectrum.

Пакет ISE WebPack представляет собой бесплатную версию САПР, доступную для загрузки с сайта Xilinx (www.xilinx.com). Он имеет ограничение по максимально поддерживаемому объему ПЛИС, но, начиная с версии 8.1, функциональные возможности WebPack полностью соответствуют ISE Foundation.

Внешний вид главного окна оболочки проектирования показан на рис. 1.1.

Главное окно ISE называется «Навигатор проектов» (*Project Navigator*). В этом окне расположены инструментальные панели и консоль сообщений. В отличие от предыдущих версий отчет о состоянии проекта, занятых ресурсах ПЛИС и результатах выполнения основных шагов трансляции хорошо структурирован и разбит на группы.

Панели, расположенные слева, представляют собой соответственно список файлов текущего проекта и список процессов (операций), доступных для текущего выбранного файла.

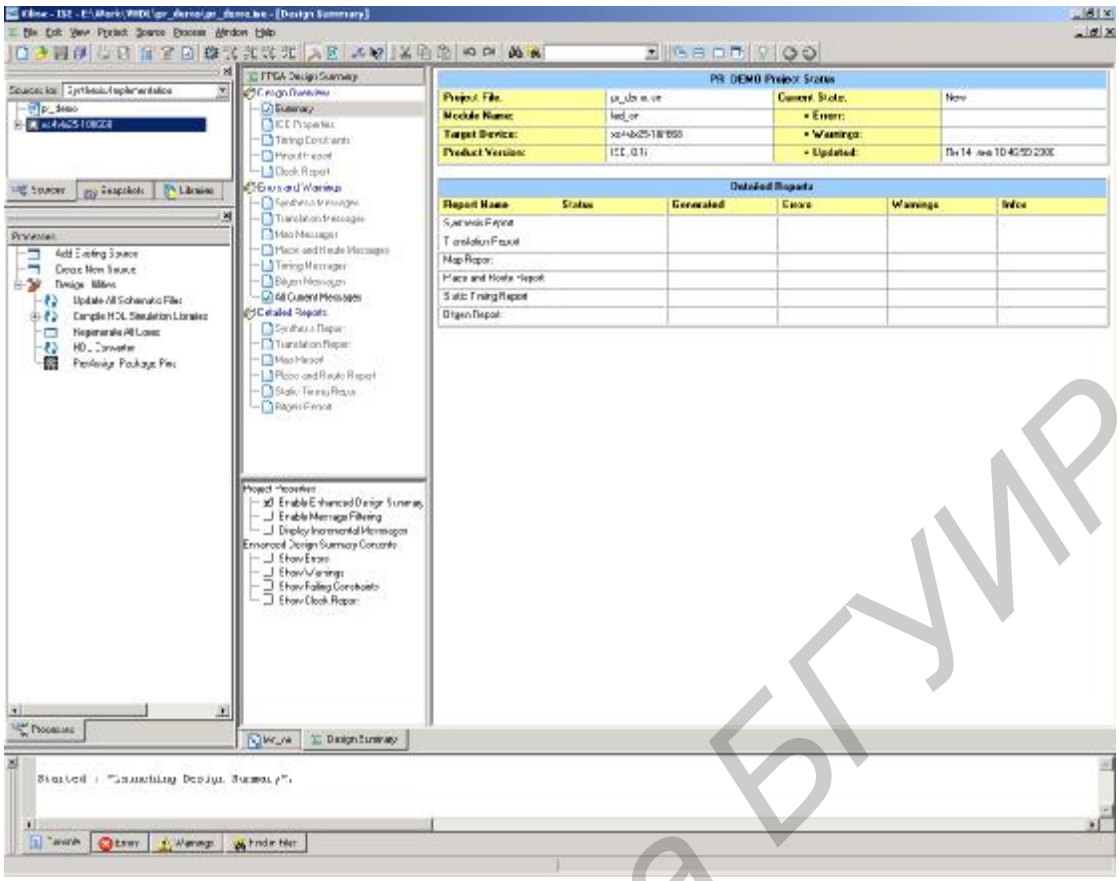

Рис. 1.1. Главное окно САПР ICE

Для создания нового проекта необходимо выбрать соответствующий пункт главного меню (*File —> New Project*). После этого открывается мастер создания нового проекта, как показано на рис. 1.2–1.3.

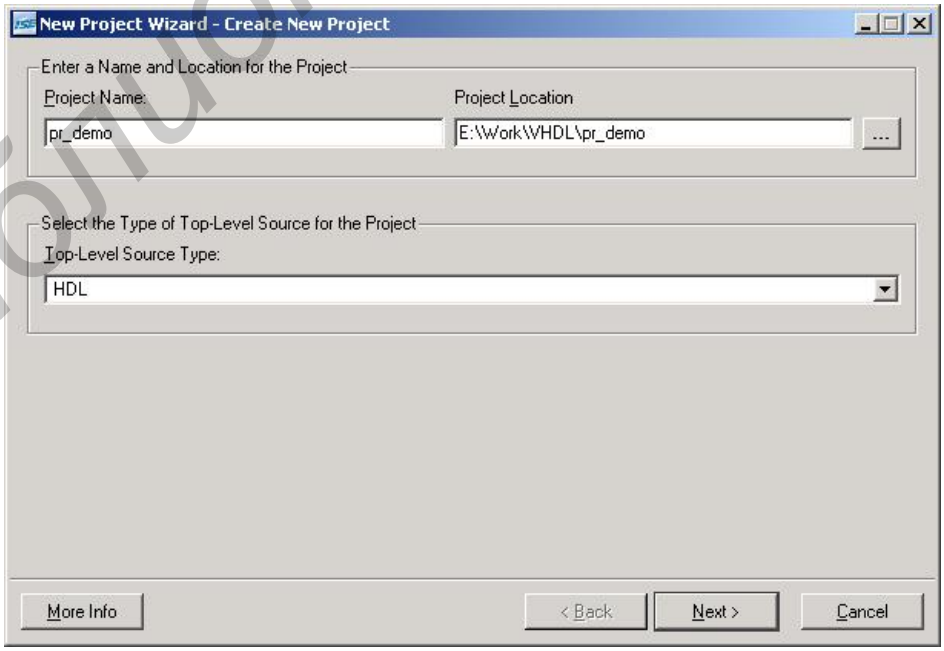

Рис. 1.2. Диалоговое окно создания нового проекта

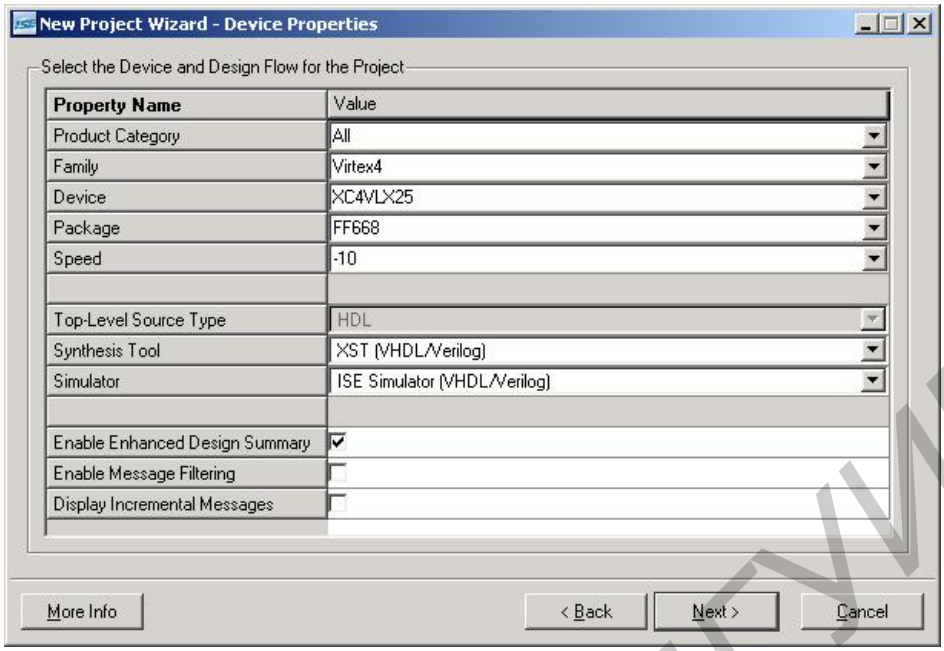

Рис. 1.3. Диалоговое окно выбора ПЛИС и маршрута проектирования

В первом окне открывшегося мастера необходимо задать имя проекта и размещение папки с материалами. Настоятельно рекомендуется выбрать для хранения проектов отдельную папку, не располагающуюся внутри папки, в которую был установлен пакет. В показанном на рис. 1.2 примере проекты размещены в *E:\Work\VHDL*.

Кроме этого, в представленном окне необходимо задать параметр *Top-Level Source Type*, т.е. тип представления модуля верхнего уровня. При этом типу HDL соответствует описание на одном из языков описания аппаратуры, Schematic задает графическое описание, а форматы EDIF и NGC/NGO предусматривают использование результатов работы внешних средств синтеза.

Описание на HDL существенно эффективнее графического ввода, поэтому выберем его.

В следующем окне мастера (рис. 1.4) можно создать новый модуль с последующим его добавлением в проект при помощи кнопки *New Source*. После создания проекта аналогичным способом можно будет создать модуль при помощи команды *Project->New Source.*

После нажатия на кнопку *New Source* будет вызвано диалоговое окно создания новых модулей (рис 1.5).

Следующее диалоговое окно (рис. 1.6) позволяет добавить к проекту уже существующие модули. Это действие можно выполнить и позже: *Project->Add Source* или *Project->Add Copy of Source.* Удобно также то, что в окне процессов размещены два пункта для быстрого добавления новых модулей или подключения к проекту существующих.

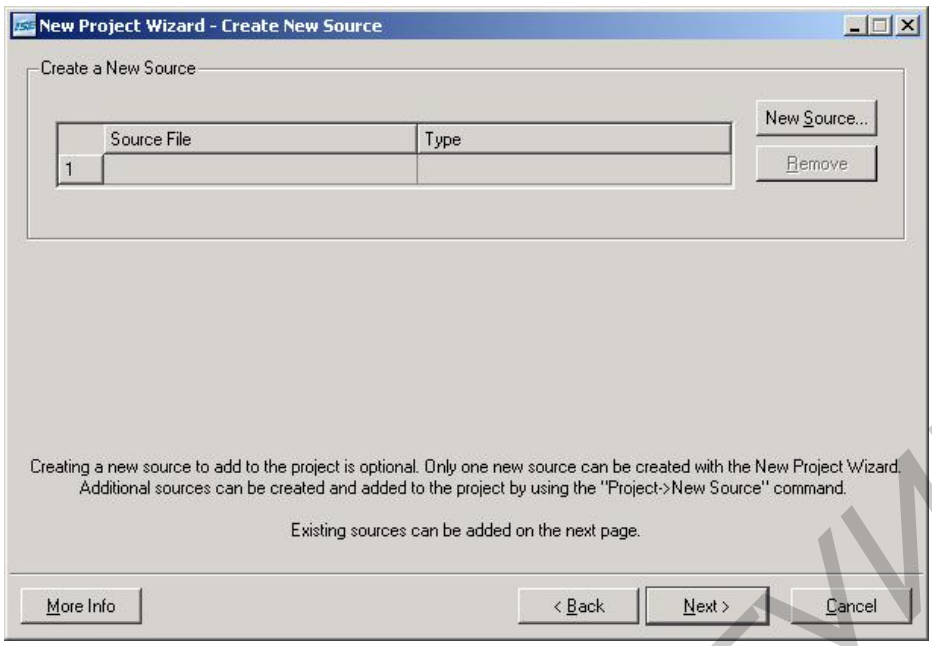

Рис. 1.4. Создание нового модуля описания проекта

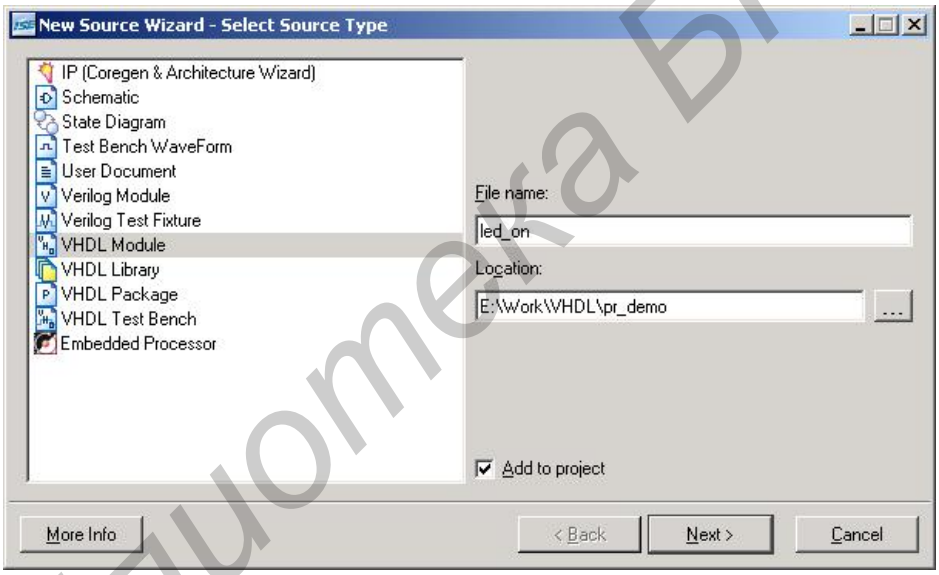

Рис. 1.5. Диалоговое окно создания новых модулей

После завершения работы мастера и создания шаблона в основном окне ISE будет показано содержимое только что созданного компонента (рис. 1.7).

Создадим простейший проект, по которому на отладочной платформе ML401 будет загораться светодиод *Центр*. Светодиод будет загораться при подаче на него ‘1’, т.е. в текстовом редакторе необходимо следующее:

c<='1';

Представленный текст вводится после ключевого слова ***begin*** в разделе architecture. После набора текста необходимо проверить описание компонента на наличие ошибок. Для этого компонент должен быть выбран в списке файлов проекта. После выбора в окне доступных процессов появится список операций, возможных для выбранного файла (рис. 1.8). После чего следует выполнить операцию *Synthesize – XST -> Check Syntax*.

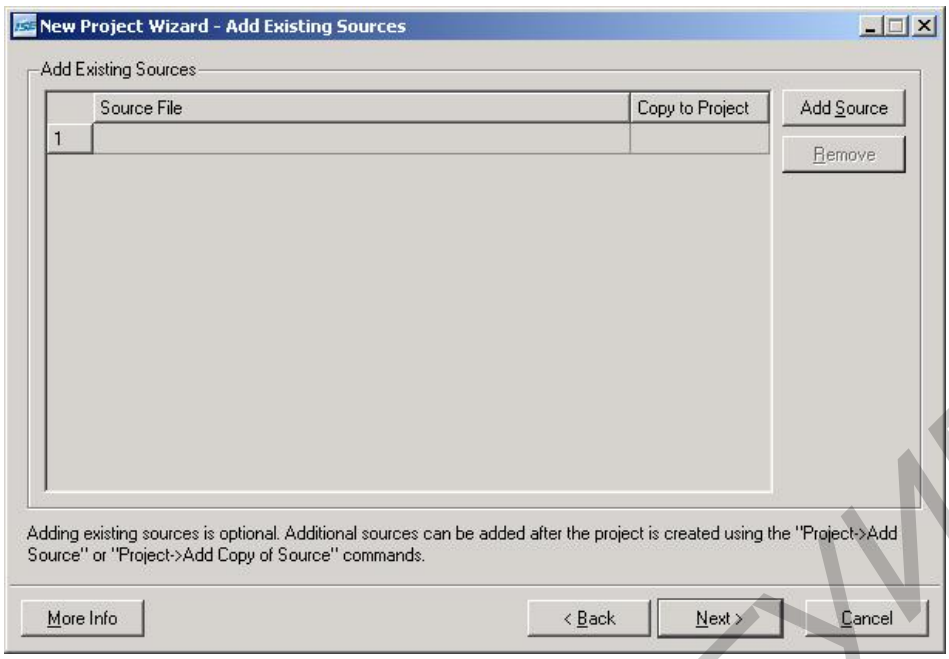

Рис. 1.6. Добавление к проекту существующих модулей

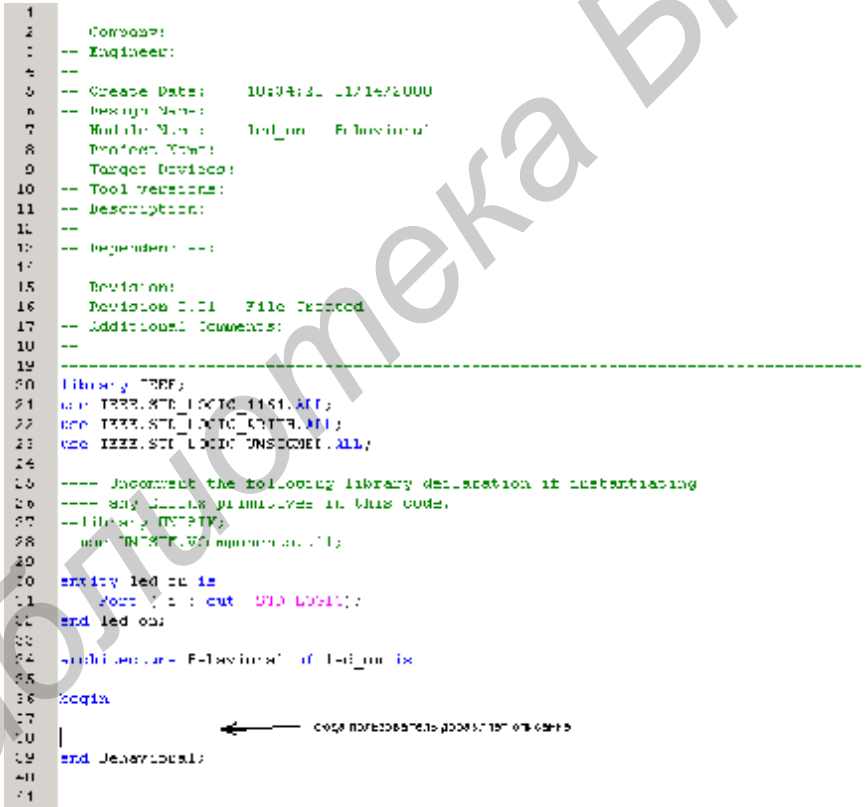

Рис. 1.7. Вид текстового редактора после завершения работы мастера

Далее требуется ассоциировать выводы разрабатываемого модуля и выводов ПЛИС (рис. 1.9). Для этого следует запустить утилиту PACE (выполнив следующие действия *User Constraints->Assign Package Pins*).

Перед открытием этого приложения *Навигатор проектов* предварительно создаст список выводов проекта и их типов. Для этого выполняются несколько шагов трансляции проекта.

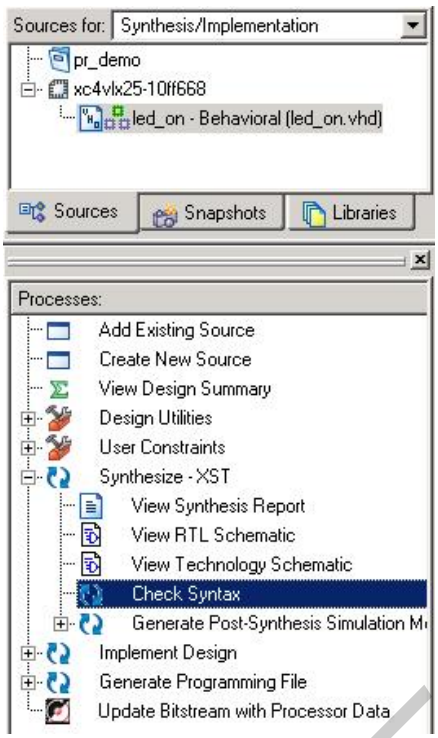

Рис. 1.8. Проверка модуля на ошибки

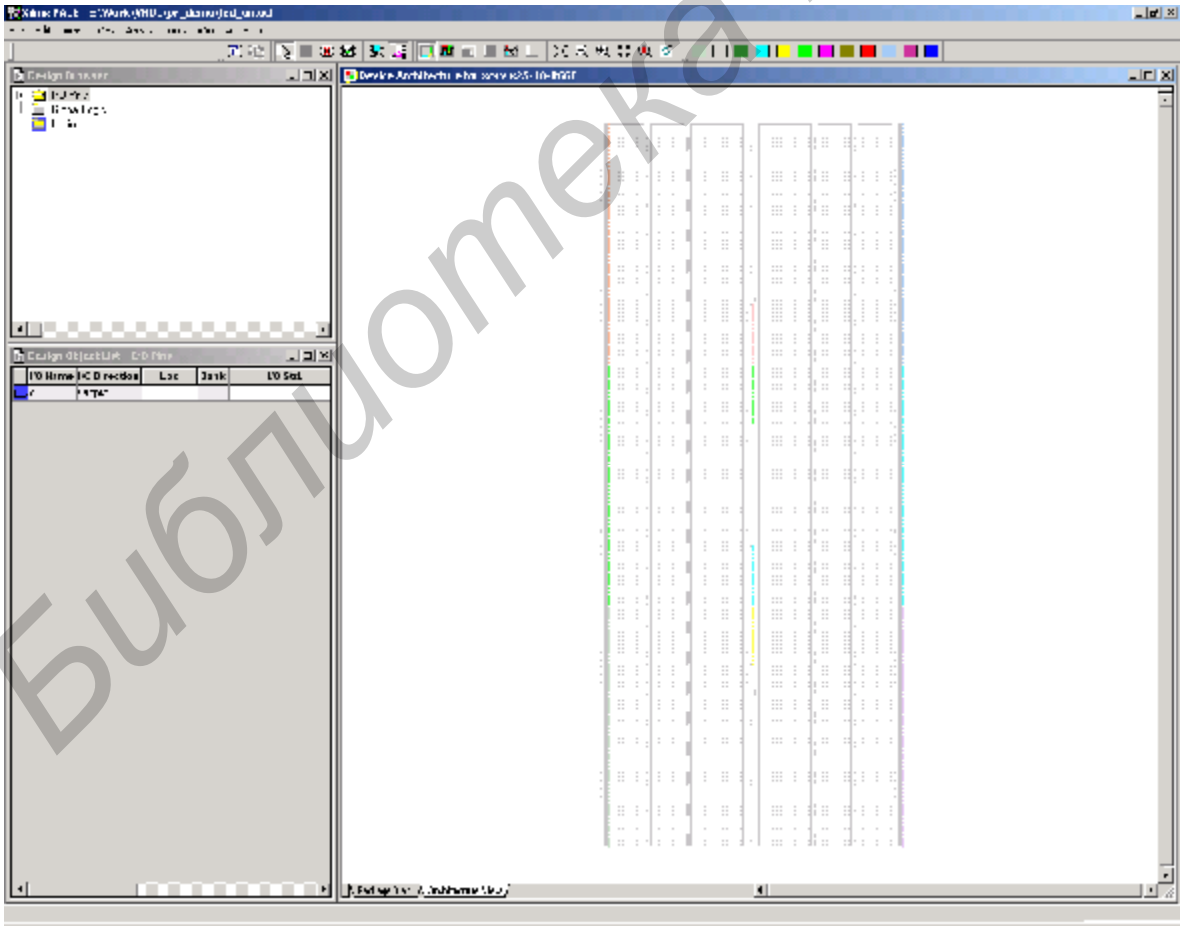

Рис. 1.9. Окно редактора выводов

В основной части окна этого приложения показано схематическое изображение кристалла ПЛИС с отмеченными выводами. В левой части расположены собственно панели редактирования номеров выводов (рис. 1.10). Рассмотрим эту часть более подробно.

Рис. 1.10. Назначение выводов ПЛИС

В графе *Loc* (расположение) необходимо ввести номер вывода ПЛИС, к которому нужно подключить соответствующий сигнал. В соседней колонке размещается название банка ввода–вывода, к которому относится данное имя. Это сделано для того, чтобы избежать ситуации, когда в одном банке оказываются линии, требующие разного напряжения питания. Значения номеров контактов также можно взять из документации к плате, которая доступна на сайте *xilinx.com*.

Колонка *I/O Std* позволяет установить для вывода один из электрических интерфейсов, которые реализуются аппаратно путем активизации соответствующих буферов ввода–вывода. В соседних колонках можно задать опорное напряжение (требуется не для всех типов интерфейса), напряжение питания линий ввода–вывода *Vcco,* максимальный выходной ток, тип внутренней подтяжки, скорость нарастания и задержку.

После выполнения функционального модулирования можно приступать непосредственно к трансляции проекта. Для трансляции необходимо указать файл *led_on.vhd* (попытка трансляции проекта без выбора главного файла не является ошибкой, но приведет к трансляции одного из библиотечных компонентов пользователя), после чего становится доступным список процессов для этого файла (рис. 1.11).

Трансляция проекта включает три основных этапа: *Synthesize, Implement Design* и *Generate Programming File.* На рис. 1.11 эти процессы показаны в «развернутом» виде. Пиктограммы около имен отдельных компонентов соответствуют типу приложений, которые будут запущены при их активизации. Значком документа показаны файлы отчета, которые при активизации будут выведены в основное окно. Однако для составления этих отчетов может потребоваться запуск каких-либо этапов синтеза или трассировки. Двойными стрелками отмечены процессы, исполняемые в основном окне ISE. Прочие пункты соответствуют внешним утилитам, открывающимся в своем окне, и значки представляют собой пиктограммы соответствующих приложений.

Для трансляции всего проекта достаточно запустить процесс *Generate Programming File.* Недостающие для его выполнения процессы будут запущены автоматически.

Ход трансляции будет показан в консоли, при появлении ошибки трансляция прекращается. При этом зеленые «галочки» напротив каждого процесса показывают его успешное завершение, желтые восклицательные знаки говорят о том, что один из вложенных процессов не был запущен (это является нормальной ситуацией), критические ошибки показываются красными крестиками. Знаки вопроса сигнализируют, что результаты выполнения этого процесса устарели и требуется его повторный запуск.

Рис. 1.11. Процессы, доступные для проекта

Настройки параметров трансляции производятся из контекстного меню для каждого процесса. Интерес представляют настройки процессов *Synthesize, Implementaion* и *Generate Programming File*. Общее количество настроек довольно велико, к тому же точный состав меняется от версии к версии САПР и от одной серии ПЛИС к другой. Поэтому здесь будут перечислены только некоторые особенности настройки параметров трансляции и синтеза, которые оказывают существенное влияние на процесс проектирования.

Рассмотрим настройку свойств синтеза проекта. Все окна настройки свойств процесса (рис. 1.12) имеют два уровня отображения: *Standard* и *Advanced*. На уровне *Standard* показаны наиболее часто используемые свойства, но ряд довольно важных настроек становится доступным только в режиме *Ad-*

vanced. Поэтому комментарии к настройкам будут сделаны именно для данного режима, но не для всех пунктов диалоговых окон.

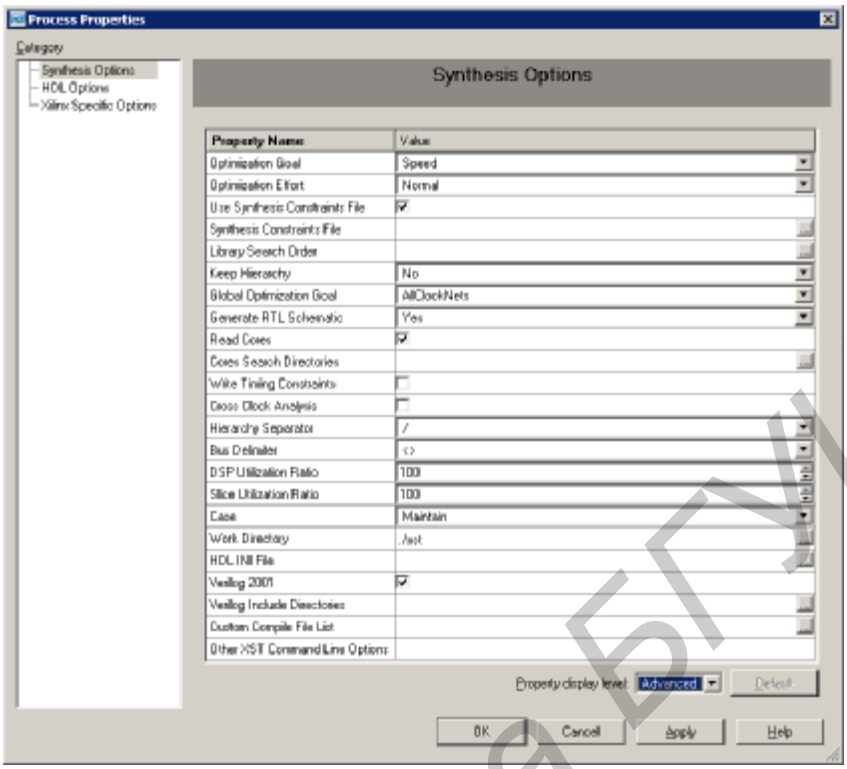

Рис. 1.12. Диалоговое окно настройки свойств трансляции

Итак, в представленном на рис. 1.12 окне нас интересуют следующие пункты.

Optimization Goal имеет два варианта: *Speed* и *Area.* Эти варианты соответствуют общей направленности алгоритмов на оптимизацию с целью достижения максимальной производительности или минимальной занимаемой площади на кристалле.

Optimization Effort регулирует «интенсивность» алгоритмов оптимизации *(effort* дословно означает «усилие»). Значение *normal* соответствует меньшему времени работы, значение *high* – более высокому качеству получаемых результатов.

Напротив параметра *Use Synthesis Constraints File* необходимо оставить флажок, поскольку только в этом случае ограничения по взаимному размещению компонентов на кристалле и подключения к выводам ПЛИС будут использоваться ISE.

Параметр *Keep Hierarchy* (варианты *Yes/No/Soft)* регулирует уровень соответствия структуры ПЛИС тому описанию, которое было введено пользователем. После эквивалентных преобразований логических выражений, упрощений и оптимизации получаемая структура логических блоков может существенно отличаться от той схемы, которая была введена разработчиком. Разумеется, функционирование при этом полностью соответствует описанию проекта. Если параметр имеет значение *No*, САПР имеет возможность провести глубо-

кую оптимизацию проекта, не сохраняя в точности ту иерархию цифровых блоков, которая задается в описании.

Параметр *Global Optimization Goal* имеет множество вариантов:

• AIICIockNets – проект оптимизируется таким образом, чтобы обеспечить максимальную частоту тактовых сетей. Вариант может использоваться в случаях, когда обработка сигнала сосредоточена в ПЛИС и выполняется большим количеством цифровых блоков, тактируемых внешним сигналом высокой частоты;

• InPad to OutPad – минимизируется время распространения сигналов от входных линий микросхемы к выходным. Вариант удобен в случаях, когда ПЛИС выступает в качестве контролера в системе, включающей несколько микросхем;

• Offset In Before – минимизируется время установления сигнала на входах цифровых блоков относительно последующего фронта тактового сигнала;

• Offset Out After – минимизируется время установления сигнала на выходах цифровых блоков относительно предыдущего фронта тактового сигнала;

• Maximum Delay – минимизируется максимальная задержка распространения. Может быть использован в проектах с большим удельным весом комбинаторной логики и сложной трассировкой.

В ряде случаев изменение параметра *Global Optimization Goal* может существенно улучшить временные характеристики проекта, поэтому рекомендуется использовать данную возможность в случаях, когда не удается удовлетворить требованиям по производительности.

На рис. 1.13 показан вид диалогового окна настройки свойств синтеза HDL. В данном окне рекомендуется оставлять показанные настройки, за исключением случаев, когда требуется воспользоваться определенными типом физических ресурсов ПЛИС для реализации той или иной функции. Например, умножение может быть реализовано как с помощью аппаратных умножителей (если они присутствуют в данной ПЛИС), так и на базе логических ячеек, что может быть явно указано в настройках. Аналогично ПЛИС Xilinx допускают использование как блочной, так и распределенной памяти. Особенно много вариантов существует для выбора способа кодирования состояний конечных автоматов. В автоматическом режиме для каждого их этих случаев выбирается вариант, соответствующий оптимальному использованию ресурсов ПЛИС (для памяти – в зависимости от объема требуемого блока, для конечных автоматов – в зависимости от числа состояний).

На рис. 1.14 показана последняя вкладка: *Xilinx Specific Options*. На ней представляют интерес следующие параметры.

Max Fanout – задает максимальный «коэффициент разветвления» для сигналов. Если один выход внутреннего блока должен быть подключен к множеству входов, трассировка программируемых соединений может существенно усложниться. Речь в данном случае не идет о превышении максимальной нагрузочной способности по протекающему току, поскольку внутри ПЛИС существуют многочисленные устройства буферизации. Дело в том, что при необхо-

димости распространения важного для проекта сигнала по множеству ячеек может оказаться полезным создать его копию, с тем чтобы каждый из блоков, распространяющих свою копию сигнала, делал это для ограниченной площади кристалла ПЛИС. Таким способом можно добиться уменьшенной задержки распространения за счет использования небольшого количества дополнительных ресурсов.

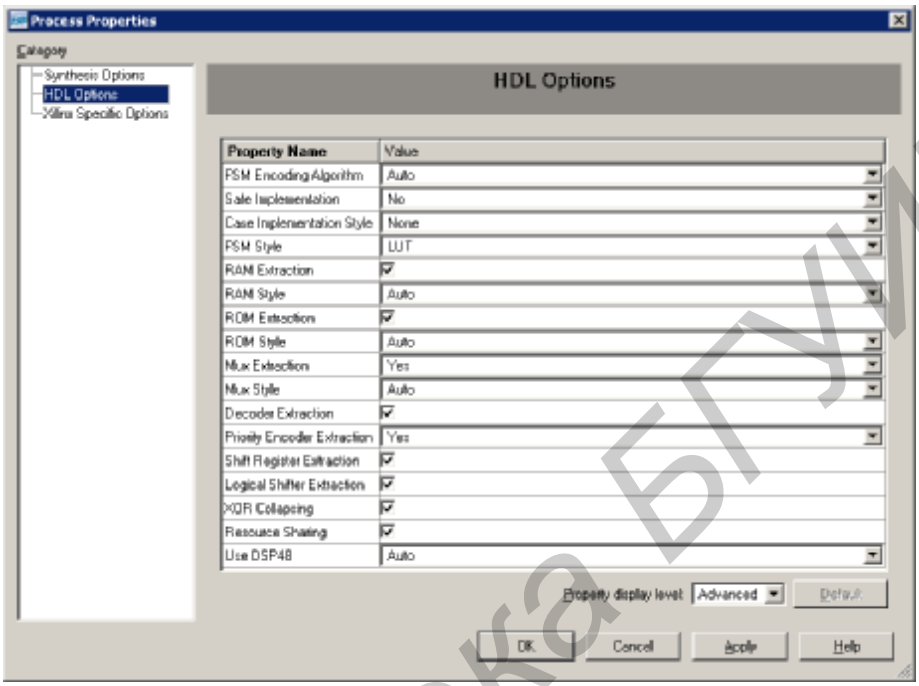

Рис. 1.13. Диалоговое окно настройки свойств синтеза HDL

Той же цели способствует параметр *Register Duplication.* При разрешении дублирования регистров САПР будет создавать их копии для тех областей ПЛИС, которые находятся на большом расстоянии от тех ячеек, которые формируют важный для проекта сигнал.

Следует убедиться, что установлены флажки *Add I/O Buffers* (автоматически размещает соответствующие буферы ввода–вывода) и *Pack I/O Registers into IOBs* (позволяет использовать регистры, имеющиеся в блоках ввода–вывода, что положительно влияет как на производительность проекта, так и на объем ресурсов).

Далее рассмотрим свойства процесса Implement Design (рис. 1.15). Доступ к диалоговому окну настройки этих свойств осуществляется также по контекстному меню, вызываемому правой кнопкой мыши.

В данном окне необходимо обратить внимание на флажок *Use LOC Constraints,* который подтверждает необходимость следовать ограничениям, наложенным разработчиком на размещение компонентов проекта. Также важным может оказаться сохранение иерархии субмодулей *(Preserve Hierarchy on Sub Module),* если данные модули были разработаны ранее и относительное размещение компонентов для них уже оптимизировано. В подобных случаях повторная трансляция для проекта «в целом» может ухудшить производительность.

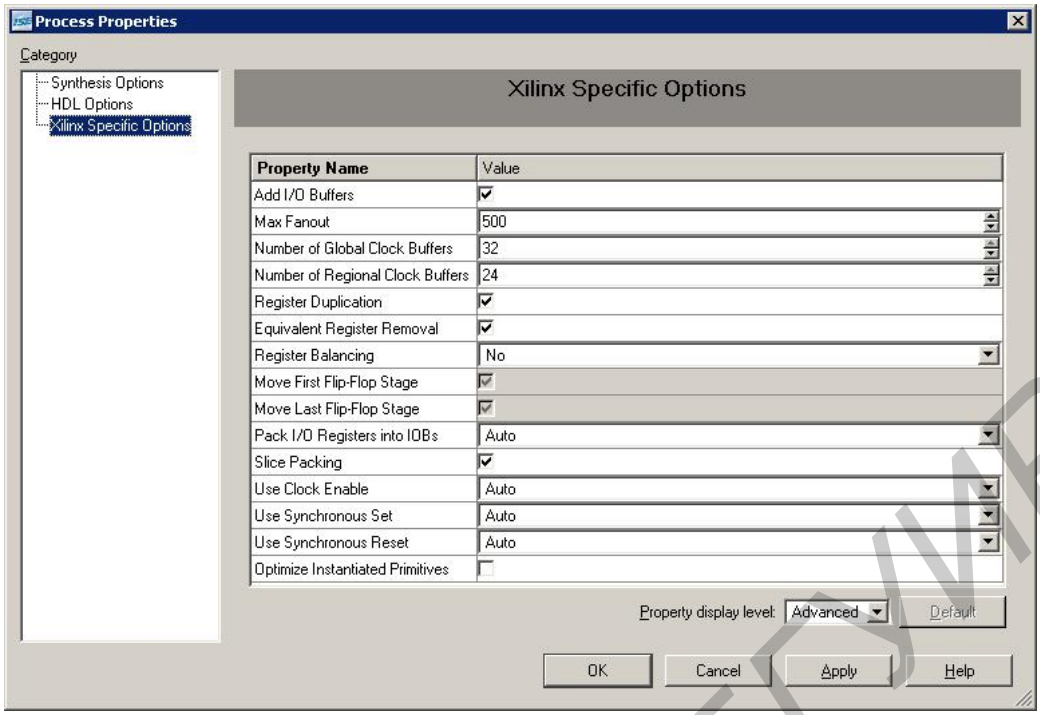

Рис. 1.14. Диалоговое окно настройки специальных параметров

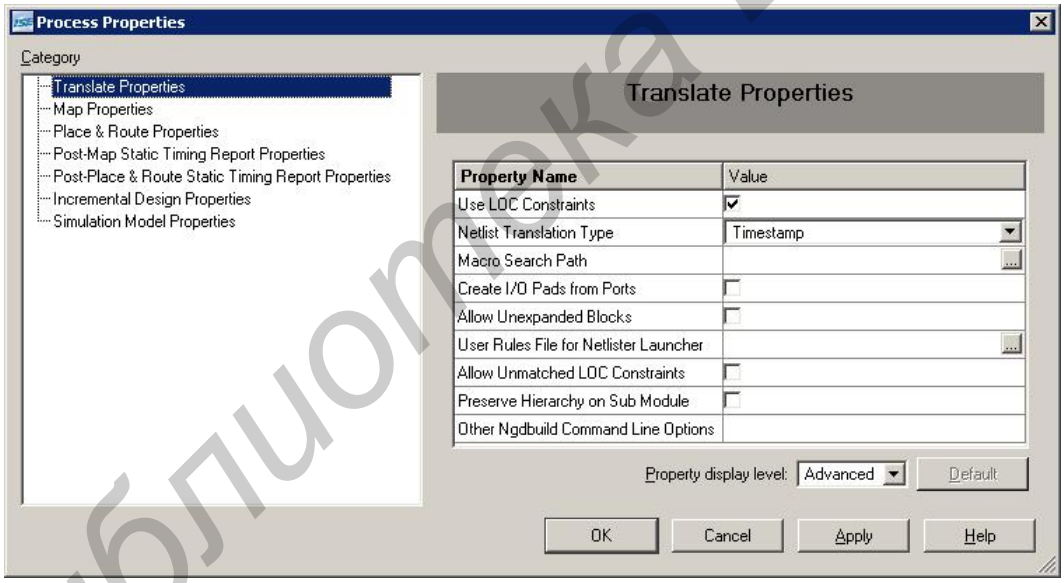

Рис. 1.15. Диалоговое окно настройки параметров трансляции

На вкладке *Map Properties* (рис. 1.16) устанавливаются свойства процесса отображения описания проекта на ресурсы ПЛИС. Свойство *Trim Unconnected Signals* позволяет исключить из проекта неиспользуемые цепи. Установленный флажок напротив *Replicate Logic To Allow Logic Level Reduction* в ряде случаев позволяет достаточно эффективно управлять процессом синтеза. Смысл этого флажка – разрешить средствам САПР «копировать» фрагменты логических блоков с целью уменьшения количества последовательно соеди-

ненных логических ячеек. Это приводит к некоторому повышению быстродействия за счет использования дополнительных ресурсов.

Установленное свойство *Allow Logic Optimization Across Hierarchy* разрешает, как это видно из названия, оптимизировать логические функции, скомбинированные из выражений, содержащихся в разных модулях проекта. Установка этого флажка может как улучшить, так и ухудшить общую производительность, поскольку в ряде случаев используется раздельная оптимизация субмодулей с последующим их объединением в ПЛИС большого объема. При этом разрешение на «перемешивание» отдельных компонентов с последующей оптимизацией может свести на нет усилия по оптимизации отдельных компонентов проекта.

Свойство *Map To Input Functions* позволяет выбрать максимально используемое число входов логических генераторов. Это значение желательно оставить выбранным по умолчанию.

Параметр *Optimization Strategy* довольно часто подвергается изменениям. Возможные значения стратегии размещения – оптимизация по используемым ресурсам *(Area)* и оптимизация по быстродействию *(Speed)*. Вариант *Balanced* представляет собой промежуточное решение.

Интересен также последний из параметров, представленных на вкладке, – *Map Slice Logic into Unused Block RAMs*. Поскольку блочная память может выполнять функции мощного логического генератора, часть многовходовых логических функций может быть реализована с помощью этих ресурсов. Установка флажка явно разрешает подобное использование этого дефицитного ресурса.

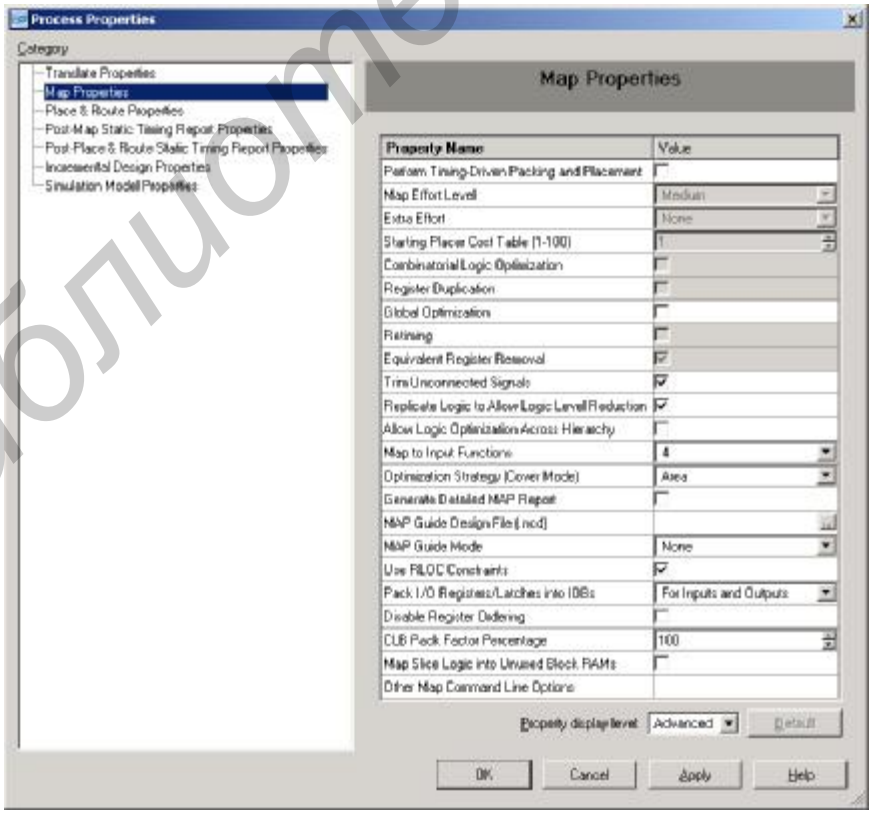

Рис. 1.16. Диалоговое окно настройки параметров отображения

Настройки процесса *Place &, Route* (рис. 1.17) позволяют установить параметры процесса взаимного размещения ресурсов ПЛИС и трассировки связей между ними. «Строгость» проверки различных условий размещения задается свойством *Place & Route Effort Level*, имеющем две отдельные настройки для этапов *Place* (размещение) и *Route* (трассировка).

Существует режим *Guide Mode*. Понятие *guide* в данном случае означает, что трассировщик берет за основу ранее разведенный вариант создаваемого проекта. Этот режим полезен, если в проект были внесены незначительные изменения, но в целом его трассировка является вполне удачной. В таком случае включение *Guide Mode* (с указанием соответствующего файла в строке *PAR Guide Design File)* позволяет сохранить удачное размещение основной части проекта.

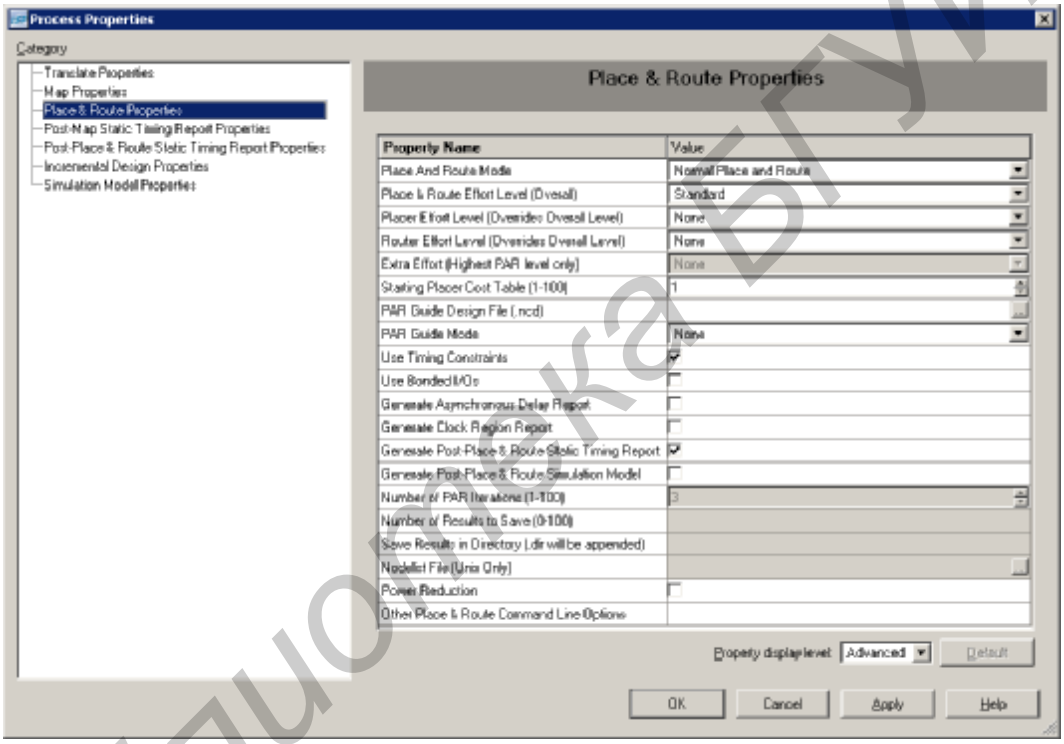

Рис. 1.17. Диалоговое окно настройки параметров размещения и трассировки

Следующие вкладки (рисунки для них не приведены) задают свойства генерации отчетов о моделировании временных характеристик проекта. Эти характеристики определяются отдельно после трансляции *(Post-Map)* трассировки *(Post-Place & Route)*. Характеристики, определенные после трассировки, соответствуют реальному размещению проекта с учетом дополнительных задержек, появившихся из-за неудачного размещения и/или трассировки.

Наконец, процесс *Generate Programming File* (генерация файла программирования ПЛИС) содержит относительно небольшое количество настраиваемых параметров.

В окне на рис. 1.18 может оказаться важной установка флажков *«Create Binary Configuration File»* и *«Create ASCII Configuration File»*. Все эти файлы

являются разновидностями форматов одной и той же конфигурационной последовательности. Однако конфигурация ПЛИС может быть загружена не только с помощью САПР ISE, но и процессором или микроконтроллером. Для этого может оказаться удобным двоичный формат (binary file) или даже текстовый формат, в котором загружаемые биты записаны обычными символами в кодах ASCII.

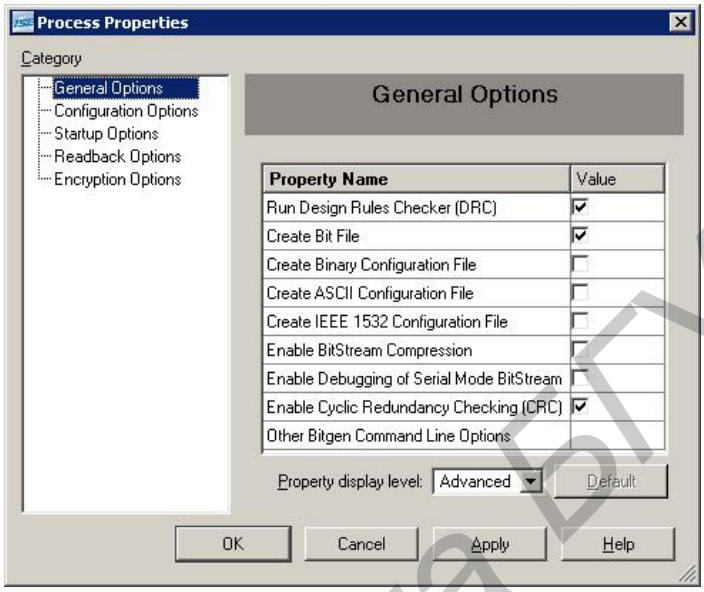

Рис. 1.18. Диалоговое окно настроек процесса формирования файла конфигурации ПЛИС

Следующая вкладка, *Configuration Options*, позволяет задать настройки процесса загрузки конфигурации ПЛИС в части установки притягивающих резисторов на выводах ПЛИС. Изменение этих настроек не представляется важным для нормальной работы.

На вкладке *Startup Options* (рис. 1.19) можно задать порядок выполнения ПЛИС внутренних операций в процессе инициализации. Последние такты загрузки конфигурационной последовательности отводятся на выполнение подготовительных к нормальной работе действий. На данной вкладке можно назначить каждому из этих действий конкретный такт из завершающего цикла. На рисунке видно, что ПЛИС выполняет:

- установку вывода Done;
- разрешение выходов (перевод их из высокоимпедансного состояния в рабочее, заданное конфигурационным файлом);
- перевод внутренней схемы сброса/установки в неактивное состояние;
- разрешения записи в регистры;
- разрешение работы устройств DLL.

Установленные по умолчанию параметры полностью соответствуют нормальной работе ПЛИС. В данном проекте не используются сигналы синхронизации, поэтому следует установить значение параметра *FPGA Start-Up*

Clock в *JTEG Clock*. Их изменение желательно производить в случае, когда цель таких изменений и ожидаемый результат понятны разработчику.

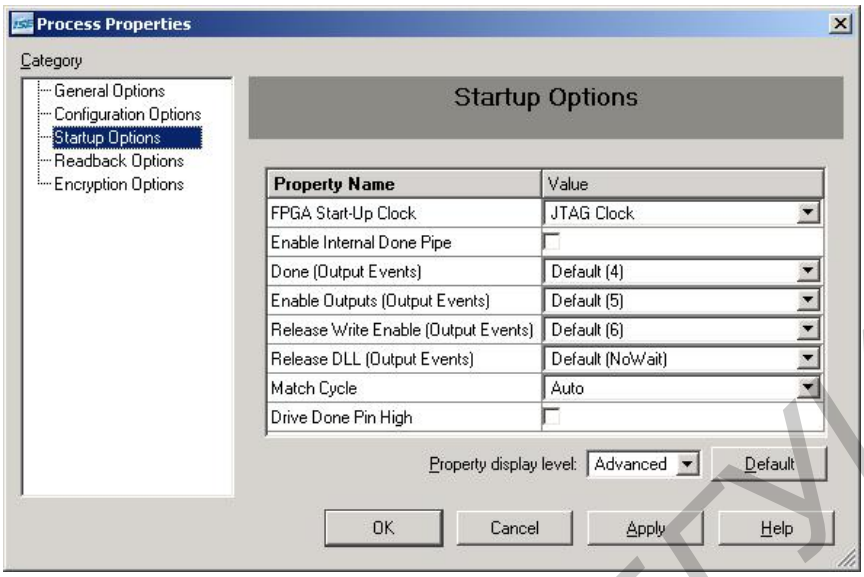

Рис. 1.19. Диалоговое окно настроек создания файла конфигурации (порядок загрузки и инициализации)

На последней вкладке можно установить очень важный параметр *Security*, отвечающий за возможность обратного считывания конфигурации ПЛИС. Запрещение обратного считывания *(Readback)* полностью защищает проект от несанкционированного копирования.

Приведенная информация о продукте ISE не является исчерпывающей, хотя и дает примерное представление о порядке работы с этой довольно эффективной версией САПР.

1.1.2. Конфигурирование отладочной платы ML401 с помощью карты памяти CompactFlash

Для трансляции всего проекта запустим процесс *Generate Programming File*. Недостающие для его выполнения процессы будут запущены автоматически.

Далее следует запустить программу iMPACT для создания конфигурационных файлов. Для этого следует выполнить следующую последовательность действий *Generate Programming File-> Generate PROM, ACE, or JTEG File*. После этих действий запустится программа конфигурирования кристалла iMPACT и откроется окно мастера определения параметров конфигурирования кристалла (рис. 1.20).

Конфигурирование платы будет производиться через контроллер *System ACE*, поэтому выбираем пункт *Prepare System ACE File* и нажимаем *Next*.

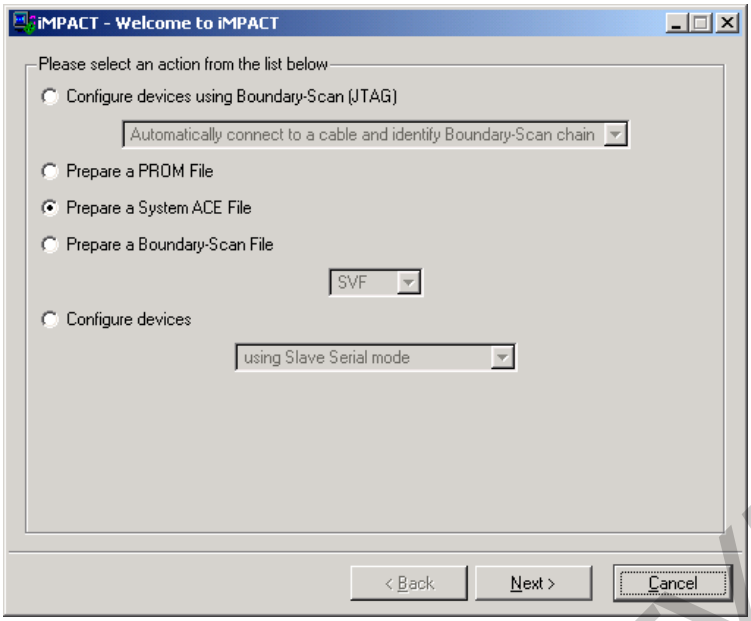

Рис. 1.20. Выбор типа создаваемых конфигурационных файлов

Подготовку файлов к конфигурированию будем осуществлять в режиме для «новичка» (Novice). Выберем данный пункт в диалоговом окне (рис. 1.21) и перейдем к следующему пункту.

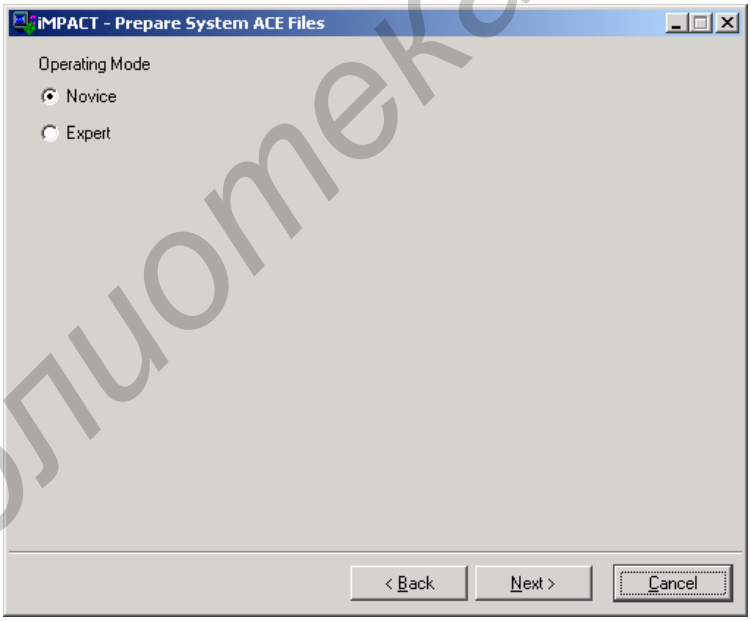

Рис. 1.21. Выбор режима создания конфигурационных файлов

В следующем окне «помощника» требуется выбрать размер карты памяти, используемой для конфигурирования платы (рис. 1.22). В комплекте с поставляемой идет карта памяти объемом 32 Мб, что соответствует 256 Мбит. Резервирование пространства не требуется. Выберем данные параметры и перейдем дальше.

В следующем окне надо задать имя создаваемой коллекции конфигураций и выбрать её расположение на диске (рис. 1.23). Имя коллекции не может быть больше 8 символов.

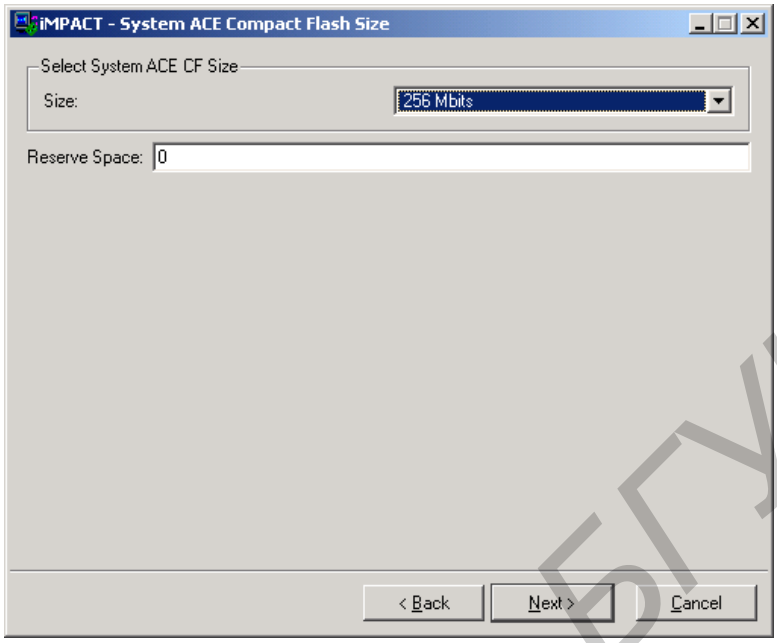

Рис. 1.22. Задание размера карты памяти CF

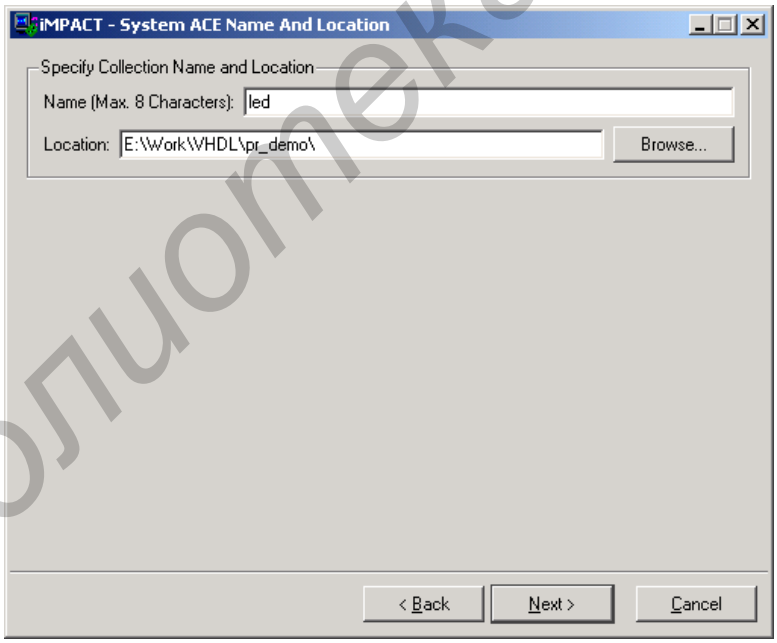

Рис. 1.23. Создание коллекции

Следующим этапом будет задание адресов и имен для конфигураций, которые будут входить в коллекцию. Так как в коллекцию у нас будет входить только одна конфигурация, то зададим конфигурирование только нулевого адреса (рис. 1.24).

В следующем окне «помощника» отобразится суммарная информация (рис. 1.25). Если вся отображаемая информация верна, то следует нажать *Finish*.

После завершения работы мастера появится сообщение о начале ассоциирования конфигурации к нулевому адресу, затем диалоговое окно выбора файлов дизайна (рис. 1.26). В данном окне надо выбрать файл led_on.bit.

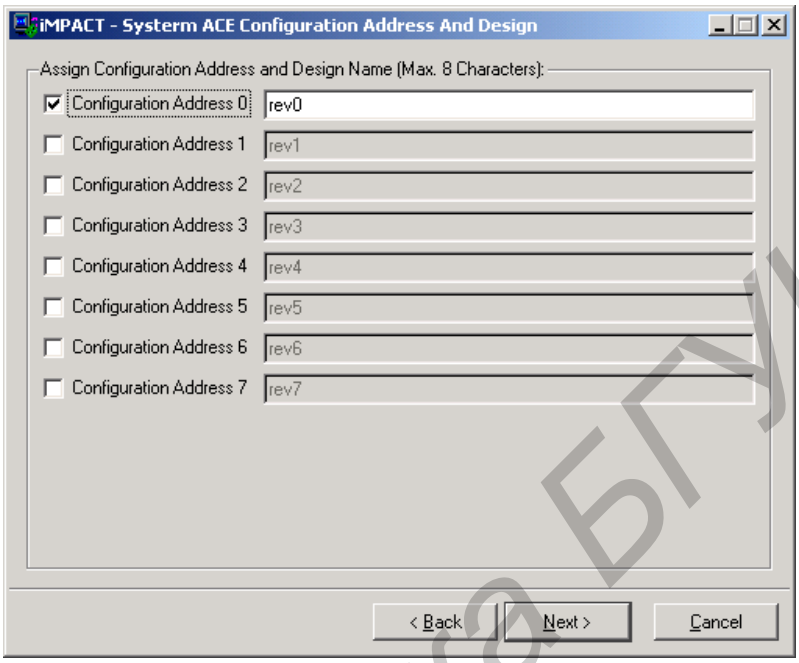

Рис. 1.24. Задание адресов конфигураций, входящих в коллекцию

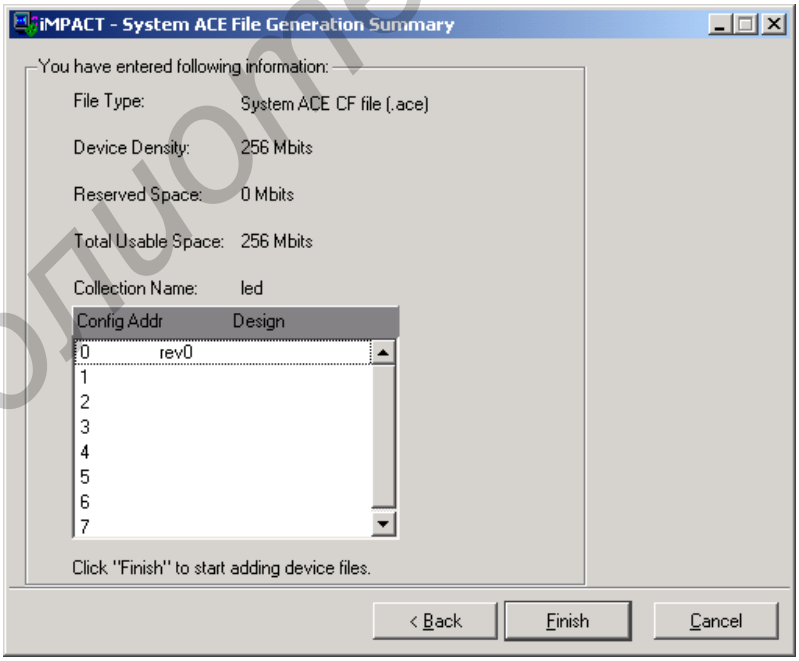

Рис. 1.25. Суммарная информация по созданной коллекции

После выбора конфигурационного файла появится сообщение с предложением перевыбрать конфигурационный файл. Если файл выбран правильно, то надо нажать на кнопку *No*.

Так как в JTEG-цепочке находятся ещё устройства, то их надо добавить для правильной работы отладочной платы. Для этого надо кликнуть в свободной области слева от нашей конфигурации правой кнопкой мыши и добавить файл описания (в появившемся контекстном меню пункт *Add Xilinx Device*) *xcf32p_vo48.bsd*, находящийся в папке *<каталог Xilinx Ise>\xcpf\data* (рис. 1.27– 1.28).

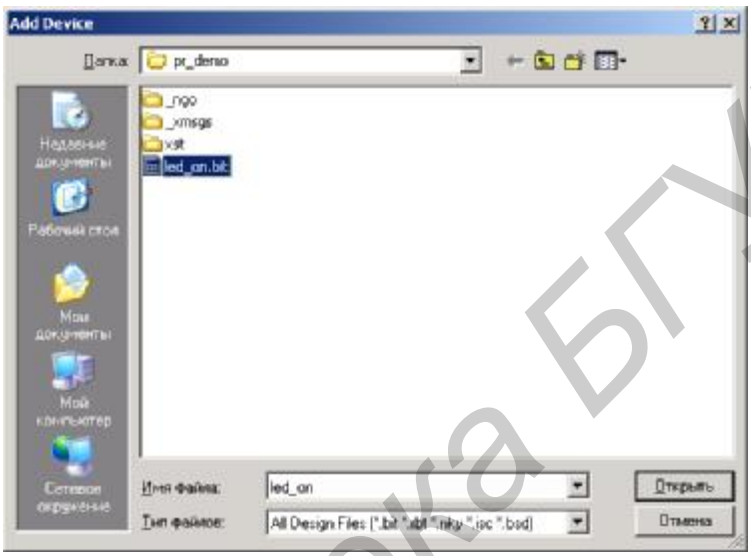

Рис. 1.26. Выбор файлов дизайна

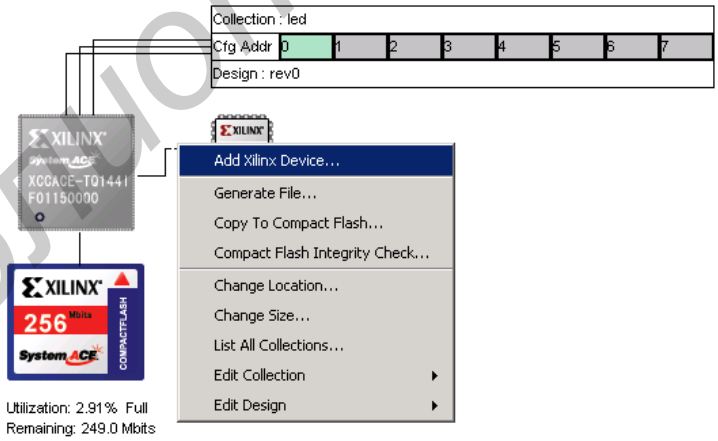

Рис. 1.27. Добавление файла описания

Аналогичным способом, только уже справой стороны от нашей конфигурации, добавляем файл описания *xc95144xl_tq100.bsd*, расположенный в папке *<каталог Xilinx Ise>\xc9500xl\data* (рис. 1.28).

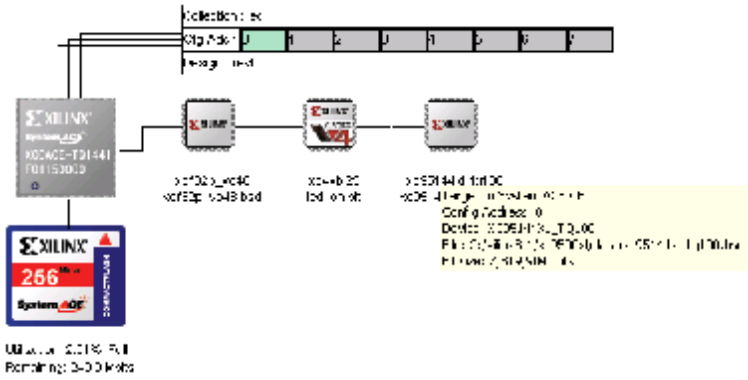

Рис. 1.28. Готовый проект для конфигурирования ПЛИС

После выполнения всех рассмотренных выше действий надо сгенерировать файлы для записывания их на *CompactFlash*. Для этого в любом месте рабочей области надо кликнуть правой кнопкой мыши и из контекстного меню выбрать пункт *Generate File*. После чего появится диалоговое окно выбора коллекции, для которой будут генерироваться файлы. Выбираем текущую коллекцию и нажимаем *OK* (рис. 1.29).

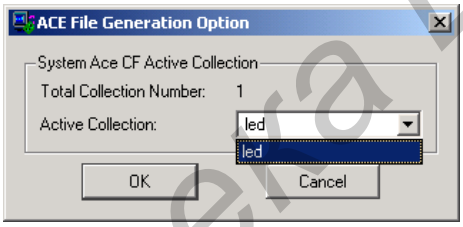

Рис. 1.29. Выбор текущей коллекции

После успешной генерации файлов на экране появится сообщение *ACE File Generation Succeeded* (рис. 1.30).

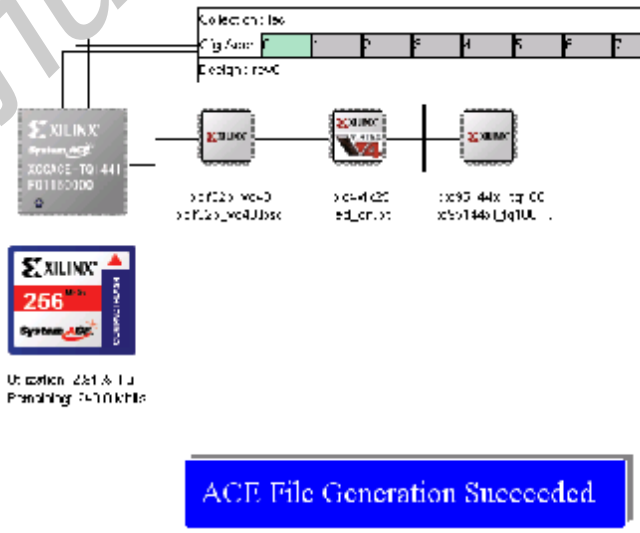

Рис. 1.30. Успешное завершение создания ACE-файлов

Далее подключим через card reader карту памяти CompactFlash к компьютеру.

Перед записью файлов конфигураций на карту памяти CompactFlash при необходимости нужно сохранить хранящиеся на ней конфигурационные данные, а затем удалить все находящиеся на ней файлы. Это следует сделать для предотвращения скапливания на ней ненужных данных и сохранения рабочих конфигураций.

Для копирования коллекции на CompactFlash надо кликнуть правой кнопкой мыши в рабочем пространстве проекта и выбрать пункт *Copy to Compact Flash*. После чего появится диалоговое окно, изображенное на рис. 1.31. Выбираем коллекцию led, затем нажимаем на кнопку *Add to Flash*. После этого нажимаем на активированную кнопку *Copy to CF*, что приведет к копированию коллекции на карту памяти. После выполнения этих действий должно появиться сообщение «*Copy to compact flash completed*».

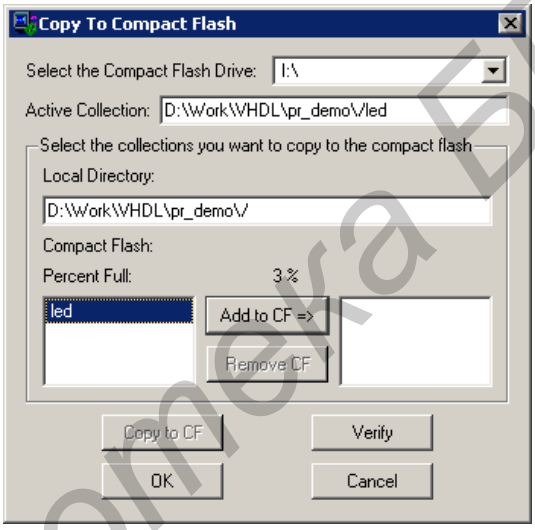

Рис. 1.31. Копирование проекта на CF-карту памяти

Теперь произведем подготовку платы к её программированию. Для этого требуется произвести ряд действий:

1) перевести переключатель включения питания на плате в положение *OFF*;

2) вставить CF-карту памяти в слот;

3) подключить источник питание к плате;

4) установить DIP-переключатели в положение 000111 (загрузка с нулевого адреса, режим конфигурирования подчиненный последовательный);

5) переключатель выбора источника конфигурации должен быть установлен в положение *SYS ACE*.

После выполнения рассмотренных выше действий можно переводить переключатель включения питания в положение *ON*. При правильных установках контроллер System ACE начинает программировать ПЛИС в момент вклю-

чения питания. Нажатие кнопки сброса System ACE также вызывает программирование ПЛИС от контроллера System ACE.

После программирования ПЛИС светодиод *Центр* должен загореться зеленым цветом.

Таблица 1.1

Варианты заданий

Вариант	Задание
1	Мигание светодиодов *Север* и *Юг* в противофазе
2	Поочередное мигание светодиодов *Север*, *Юг*, *Запад*, *Восток*
3	Управление загоранием светодиодов *Север* и *Юг* при помощи DIP-переключателей общего назначения
4	Мигание светодиода *Центр* с периодом 1 с

1.2. Порядок выполнения работы

1. Изучить теоретические сведения по теме лабораторной работы.
2. Получить у преподавателя задание для выполнения практической части работы (табл. 1.1).
3. На языке VHDL составить описание требуемого устройства в соответствии с вариантом.
4. Синтезировать устройство, проверить его работу. Оценить количество занимаемых в ПЛИС ресурсов.
5. Показать результат работы устройства преподавателю.
6. Оформить и защитить отчет по лабораторной работе.

1.3. Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения.
4. Описание устройства на VHDL с комментариями.
5. Временные диаграммы симуляции работы устройства.
6. Листинг отчета синтезатора о количестве занятых ресурсов.
7. Выводы по работе.

2. ЛАБОРАТОРНАЯ РАБОТА №2
MAC-ПРОЦЕССОР

Цель работы: составить VHDL-описание MAC-процессора, получить временные диаграммы его работы, оценить объем занимаемых ресурсов и быстродействие.

2.1. Теоретические сведения

Одна из основных операций, выполняемых цифровыми процессорами обработки сигналов (ЦПОС), – цифровая фильтрация:

$$y(k) = \sum_{i=0}^{N-1} h(i) \cdot x(k-i). \qquad (2.1)$$

Анализ данного выражения показывает, что базовой операцией цифровой фильтрации является умножение с аккумулированием результата:

$$S = S + A \cdot B. \qquad (2.2)$$

Функциональная схема одного из вариантов реализации устройства показана на рис. 2.1.

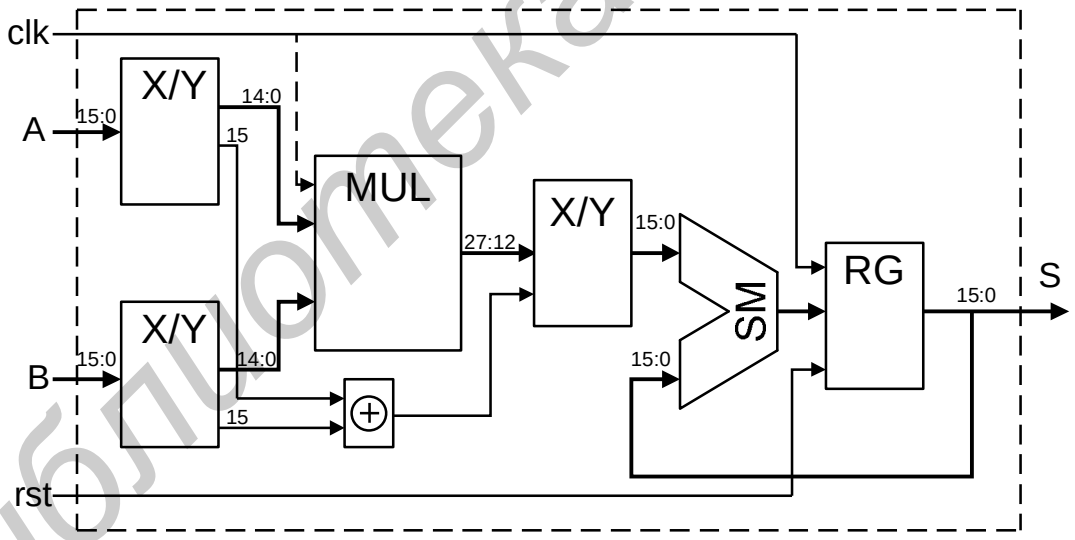

Рис. 2.1. Функциональная схема устройства умножения–накопления

На вход устройства умножения–накопления (MAC-ядра) поступают 16-разрядные числа со знаком. Знаковые разряды складываются по модулю 2, формируя знак результата. Далее результат умножения складывается с содержимым регистра-накопителя, сумма сохраняется в регистре-накопителе *RG*. Управляющий сигнал rst сообщает регистру *RG* о необходимости выдачи результата, после чего регистр обнуляется.

Основным функциональным блоком этого устройства является умножитель двоичных чисел. От вида умножителя зависят такие параметры устройства, как быстродействие, требования к площади кристалла и энергопотребление. Лабораторная работа включает реализацию устройства умножения–накопления, содержащего умножитель в соответствии с вариантом (табл. 2.1).

Таблица 2.1

Варианты заданий

Вариант	Умножитель
1	Матричный умножитель
2	Умножение, начиная с младших разрядов множителя, со сдвигом суммы частичных произведений вправо при неподвижном множимом
3	Умножение, начиная со старших разрядов множителя, со сдвигом суммы частичных произведений влево при неподвижном множимом
4	Умножитель по модифицированному алгоритму Бута (сразу на 2 разряда)

2.2. Порядок выполнения работы

1. Изучить теоретические сведения по теме лабораторной работы.
2. Получить у преподавателя задание для выполнения практической части работы.
3. На языке VHDL составить структурное описание устройства умножения–накопления в соответствии с вариантом.
4. Синтезировать устройство, проверить его работу. Оценить быстродействие, количество занимаемых в ПЛИС ресурсов и энергопотребление.
5. Показать результат работы устройства преподавателю.
6. Оформить и защитить отчет по лабораторной работе.

2.3. Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения.
3. Схема синтезируемого устройства.
4. Описание устройства на VHDL с комментариями, листинг файла теста.
5. Временные диаграммы симуляции работы устройства.
6. Листинг отчета синтезатора о быстродействии, количестве занятых ресурсов.
7. Выводы по работе.

3. ЛАБОРАТОРНАЯ РАБОТА №3
ИССЛЕДОВАНИЕ БАЗОВЫХ АЛГОРИТМОВ ЦОС НА MAC-АРХИТЕКТУРЕ

Цель работы: составить VHDL-описание устройства в соответствии с вариантом на базе MAC-ядра, получить временные диаграммы его работы, оценить объем занимаемых ресурсов и быстродействие.

3.1. Теоретические сведения

MAC-ядро является основной составной частью многих устройств ЦОС. Например, для реализации КИХ-фильтра к MAC-ядру необходимо добавить буфер входных отсчетов, буфер для хранения коэффициентов, счетчик и другие элементы. Аналогичным образом реализуется БИХ-фильтр.

MAC-ядро может применяться в генераторах сигналов синусоидальной формы. Существует несколько аналитических (вычислительных) методов формирования синусоидальных сигналов. Остановимся на двух из них. Первый из таких методов основан на формулах суммы двух углов:

$$\begin{aligned} &sin(\alpha+\beta)=sin(\alpha)\cdot cos(\beta)+cos(\alpha)sin(\beta),\\ &cos(\alpha+\beta)=cos(\alpha)\cdot cos(\beta)+sin(\alpha)sin(\beta). \end{aligned} \qquad (3.1)$$

Этот метод применим для последовательной генерации отсчетов одной синусоиды и позволяет вычислить очередной отсчет функции, например $sin(a\cdot n) = sin(a\cdot(n-1)+a)$, через предыдущий $sin(a\cdot(n-1))$. Использовав $a\cdot(n-1)$ в качестве α и a в качестве β и обозначив $S1 = sin(a)$, $C1 = cos(a)$, $Sn(n) = sin(a\cdot n)$, $Cn(n) = cos(a\cdot n)$, получим следующие выражения для вычисления очередных значений синуса и косинуса:

$$\begin{aligned} &Sn(n) = C1\cdot Sn(n-1) + S1\cdot Cn(n-1),\\ &Cn(n) = C1\cdot Cn(n-1) - S1\cdot Sn(n-1)\ . \end{aligned} \qquad (3.2)$$

Исходными данными для вычисления первой гармоники (с минимальной возможной частотой) являются следующие значения: $Sn(0) = 0$, $Cn(0) = 1$ – значения синуса и косинуса для нулевого аргумента; N – длина выборки (число отсчетов в первой гармонике); $a = 2\pi/N$, – минимальное приращение аргумента и $S1 = \sin(2\pi/N)$, $C1 = cos(2\pi/N)$ – значения синуса и косинуса для минимального аргумента. На рис. 3.1 показан пример синусоиды с периодом 8 отсчетов ($N = 8$).

Учитывая, что исходными значениями для k-й гармоники являются начальный угол $\beta = a\cdot k$, $Sk = \sin(a\cdot k)$ и $Ck = \cos(a\cdot k)$, для их вычисления можно использовать те же самые формулы:

$$\begin{aligned} &Sk(k) = C1\cdot Sk(k-1) + S1\cdot Ck(k-1),\\ &Ck(k) = C1\cdot Ck(k-1) - S1\cdot Sk(k-1)\ . \end{aligned} \qquad (3.3)$$

Исходными данными для вычисления, как и в выражении (3.2), являются значения: $Sk(0) = 0$ и $Ck(0) = 1$.

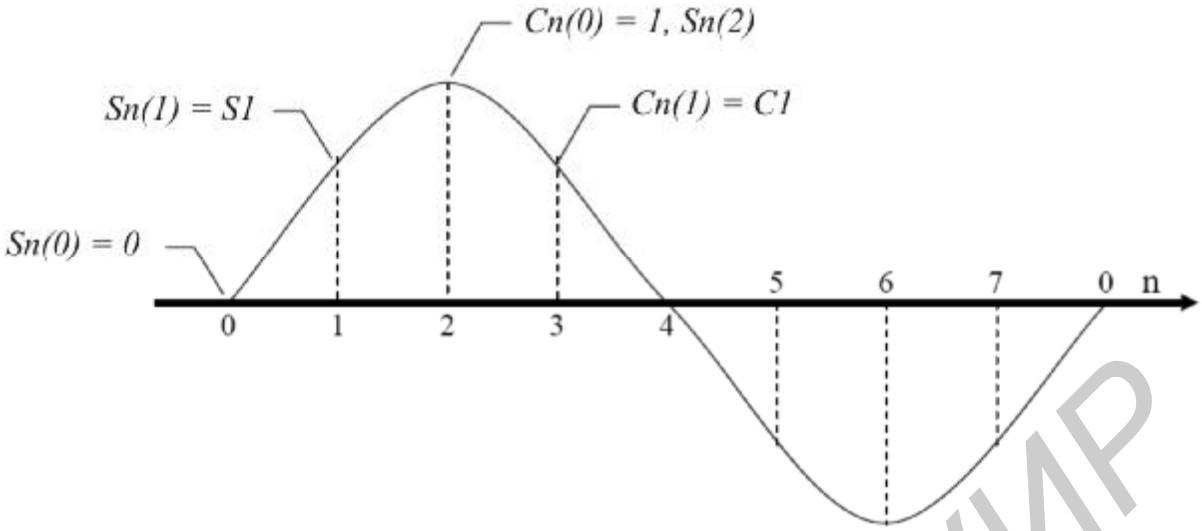

Рис. 3.1. Синусоида с начальными значениями SIN и COS

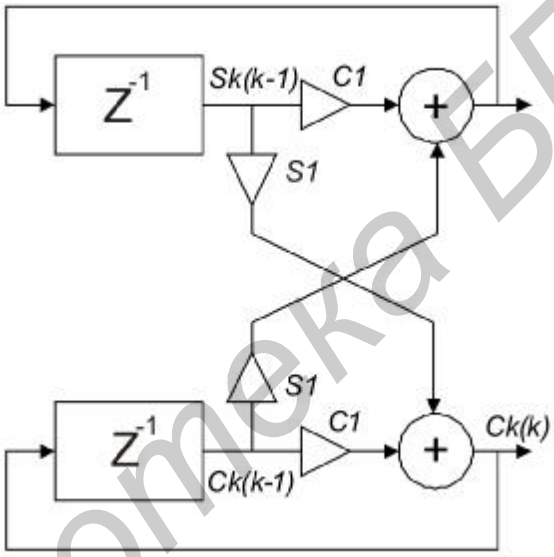

Рис. 3.2. Генератор сигналов SIN и COS

Второй способ заключается в использовании неустойчивого звена второго порядка. За основу построения генератора синусоидальных колебаний (рис. 3.2) взято простейшее рекурсивное звено 2-го порядка – цифровой резонатор, передаточная функция которого имеет вид

$$T_{Ц}(z) = \frac{1}{1 - b_{1z}^{-1} - b_{2z}^{-1}}. \qquad (3.4)$$

Условиями устойчивости такого звена является выполнение соотношений $|b_1|<2$; $b_2<0$; $|b_2|<1$. Резонансные свойства цифрового резонатора характеризуются полосой пропускания (по уровню 0,707 от максимального):

$$2\Delta w = 2 \times 2p\Delta f = \frac{1}{T_{Д}} 2 \arcsin \frac{1 + b_2}{2\sqrt{-b_2}}, \qquad (3.5)$$

и добротностью

$$Q=\omega_0/2\Delta\omega, \qquad (3.6)$$

где ω_0 – резонансная частота.

$$w_0 = 2pf_0 = \frac{1}{T_Д} 2 \arcsin \frac{b_1}{2\sqrt{|b_2|}}. \qquad (3.7)$$

Видно, что при $b_2 = -1$ звено становится неустойчивым:

$$2\Delta w = \frac{1}{T_Д} 2 \arcsin \frac{1-1}{2\sqrt{1}} = 0, Q = \frac{w_0}{0} \to \infty, \qquad (3.8)$$

и начинают генерироваться колебания при получении начального возбуждения.

Разностное уравнение $y(kT)=y_T(k)=x_T(k)+b_1y_T(k–1)+b_2y_T(k–2)$ при $x_T(k)=0$, $b_2= –1$ получает вид $y(kT)=y_T(k)=b_1y_T(k–1)–y_T(k–2)$. Схема генератора показана на рис. 3.3.

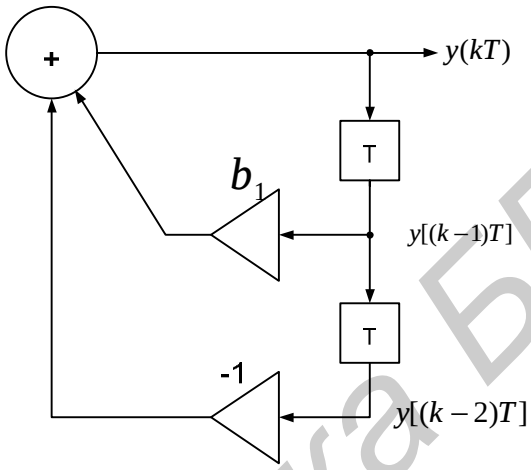

Рис. 3.3. Структурная схема генератора синусоидальных колебаний

Для работы генератора необходимо сформировать начальное возбуждение, т.е. ввести начальные значения $y_T(k–2)$ и $y_T(k–1)$ (рис. 3.4).

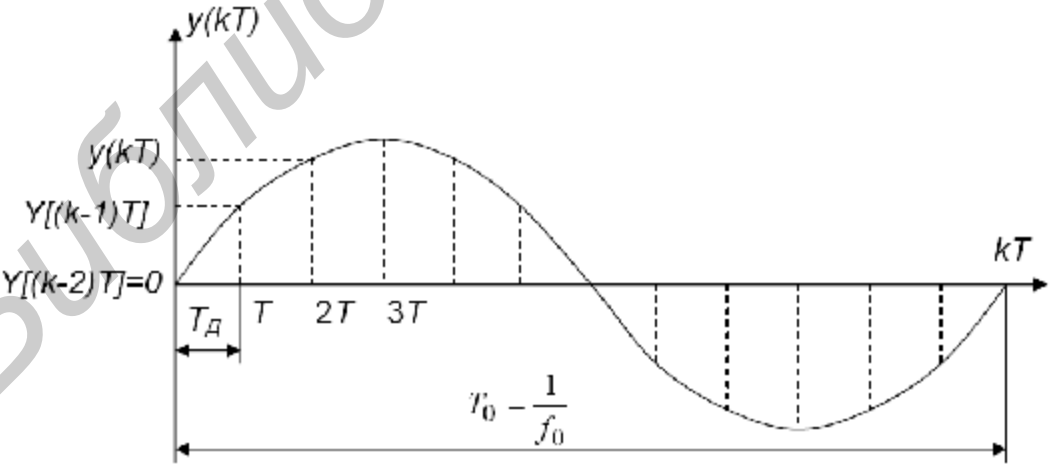

Рис. 3.4. Пояснение к формированию начальных условий

Из рис. 3.4 видно, что можно ввести любые два соседних отсчета. Для удобства примем $y_T(k-2)=0$, тогда

$$y_T(k-1) = \sin 2p \frac{T_Д}{T_0} = \sin 2p \frac{f_0}{f_Д}. \quad (3.9)$$

Таблица 4.1

Варианты заданий

Вариант	Устройство
1	КИХ-фильтр, порядок 8
2	БИХ-фильтр, порядок 4
3	Генератор синусоиды на основе рекурсивных вычислений
4	Генератор синусоиды на основе звена второго порядка

3.2. Порядок выполнения работы

1. Изучить теоретические сведения по теме лабораторной работы.
2. Получить у преподавателя задание для выполнения практической части работы (табл. 3.1).
3. На языке VHDL составить структурное описание устройства в соответствии с вариантом.
4. Синтезировать устройство, проверить его работу. Оценить быстродействие, количество занимаемых в ПЛИС ресурсов и энергопотребление.
5. Показать результат работы устройства преподавателю.
6. Оформить и защитить отчет по лабораторной работе.

3.3. Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения.
3. Схема синтезируемого устройства.
4. Описание устройства на VHDL с комментариями, листинг файла теста.
5. Временные диаграммы симуляции работы устройства.
6. Листинг отчета синтезатора о быстродействии, количестве занятых ресурсов.
7. Выводы по работе.

4. ЛАБОРАТОРНАЯ РАБОТА №4
ПРОЦЕССОР НА РАСПРЕДЕЛЕННОЙ АРИФМЕТИКЕ

Цель работы: составить VHDL-описание КИХ-фильтра на базе распределенной арифметики, получить временные диаграммы его работы, оценить объем занимаемых ресурсов и быстродействие.

4.1. Теоретические сведения

4.1.1. Прямой механизм распределенной арифметики

В основе распределенной арифметики (DA – distributed arithmetic) лежат последовательные побитовые вычислительные операции, осуществление которых позволяет получать выходной результат из пары входных векторов на каждом отдельном этапе вычисления. Преимуществом распределенной арифметики является исключительная эффективность реализации. Недостатком может быть названа невысокая скорость получения результата, из-за изначально присущего этому механизму последовательного способа поступления входных данных.

Прямой механизм распределенной арифметики предполагает вычисление выражения следующего вида:

$$y = \sum_{k=1}^{K} A_k x_k , \tag{4.1}$$

где K – количество частичных произведений;

$A_к$ – постоянные коэффициенты;

x_k – входные слова данных.

Если каждое входное слово данных x_k является нормированным двоичным числом в дополнительном коде, т.е. выполняется условие, что $|x_k| \leq 1$, то данное слово можно записать в следующем виде:

$$x_k = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} , \tag{4.2}$$

где b_{k0} – знаковый разряд;

b_{kn} – значения бит;

N – разрядность входных чисел.

Подставив выражение (4.2) в (4.1), можно получить:

$$y = \sum_{k=1}^{K} A_k \left[-b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \right] = \sum_{k=1}^{K} A_k \left(-b_{k0} \right) + \sum_{n=1}^{N-1} \left[\sum_{k=1}^{K} A_k b_{kn} \right] 2^{-n} . \tag{4.3}$$

Так как каждый бит b_{kn} может принимать только 2 значения (0 или 1), то сумма $\sum A_k b_{kn}$ может принимать 2^K возможных значений. Аналогично и сумма $\sum A_k(-b_{k0})$ имеет 2^K возможных значений. Подсчитав все значения, можно сохранить их в памяти размером 2×2^K слов. Разрядность слов памяти при этом будет

определяться разрядностью постоянных коэффициентов $A_к$ (в большинстве случаев также равна N).

Для вычисления выражения (4.3) достаточно сложить вместе N соответствующих значений, хранимых в памяти, которые сдвигаются влево на необходимое число разрядов (что соответствует присутствующему в рассматриваемом выражении умножению на 2^{-n}). При этом используется только одно значение из части памяти, хранящей суммы вида $\sum A_k(-b_{k0})$, и N –1 значений из части памяти, хранящей суммы вида $\sum A_k b_{kn}$. Для выполнения данного сложения достаточно одного сумматора.

Если сопровождать появление знаковых разрядов специальным сигналом T_S, то его можно использовать для выбора требуемой части памяти. В этом случае входные данные, вводимые последовательно (начиная с младших битов), можно использовать для прямой адресации памяти. Тогда в каждом такте с выхода памяти может сниматься требуемая сумма $\sum A_k b_{kn}$ или $\sum A_k(-b_{k0})$, которая должна суммироваться с предыдущим значением, сдвинутым на 1 разряд влево (умноженным на 2^{-1}). После N таких циклов на выходе сумматора будет содержаться требуемый результат. Таким образом, сумматор необходимо охватить петлей обратной связи, содержащей элемент сдвига влево на 1 разряд, а также переключателем, который будет коммутировать выход сумматора либо на цепь обратной связи (в процессе вычисления), либо на выход схемы (при готовности результата).

Структура схемы, реализующей выражение (4.1) на распределенной арифметике в соответствии с вышерассмотренной методикой, приведена на рис. 4.1. Содержимое блока памяти, определенное в соответствии с выражением (4.3) сведено в табл. 4.1.

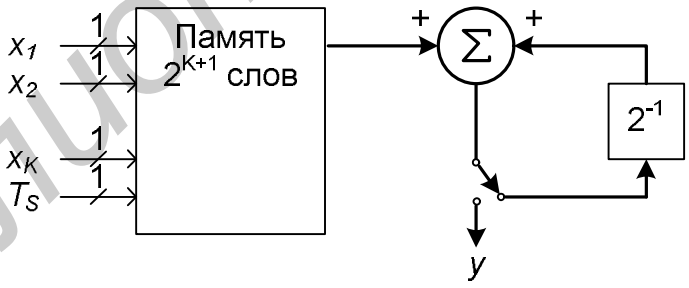

Рис. 4.1. Реализация вычисления на распределенной арифметике (память 2^{K+1} слов)

4.1.2. Уменьшение размера памяти

Уменьшение размера используемой памяти в два раза (до 2^K слов) можно достичь путем замены сумматора на рис. 4.1 сумматором/вычитателем. В данном случае сигнал T_S можно использовать в качестве управляющего сигнала для переключения режимов сложения/вычитания, как показано на рис. 4.2.

Таблица 4.1

Содержимое блока памяти (2^{K+1} слов)

Входной код					Содержимое памяти	Входной код					Содержимое памяти
T_s	b_{1n}	b_{2n}	…	b_{Kn}		T_s	b_{1n}	b_{2n}	…	b_{Kn}	
	0	0	0	0	0		0	0	0	0	0
	1	0	0	0	A_1		1	0	0	0	$-A_1$
	0	1	0	0	A_2		0	1	0	0	$-A_2$
	1	1	0	0	A_1+A_2		1	1	0	0	$-(A_1+A_2)$
	…				…		…				…
0	0	0	0	1	A_K	1	0	0	0	1	$-A_K$
	1	0	0	1	A_1+A_K		1	0	0	1	$-(A_1+A_K)$
	0	1	0	1	A_2+A_K		0	1	0	1	$-(A_2+A_K)$
	1	1	0	1	$A_1+A_2+A_K$		1	1	0	1	$-(A_1+A_2+A_K)$
	...				...		...				...
	1	1	1	1	$A_1+A_2+...+A_K$		1	1	1	1	$-(A_1+A_2+...+A_K)$

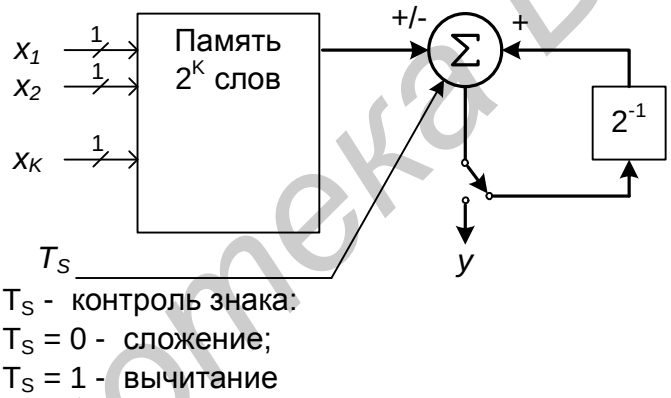

Рис. 4.2. Реализация вычисления на распределенной арифметике (память 2^K слов)

Размер таблицы памяти можно уменьшить до 2^{K-1} слов с помощью использования смещенного двоичного кода. Для этого входное слово данных x_k представляется в следующем виде:

$$x_k = \frac{1}{2}\left[x_k - (-x_k)\right], \tag{4.4}$$

где $-x_k$ – отрицательное значение слова данных x_k (в дополнительном коде):

$$-x_k = -\bar{b}_{k0} + \sum_{n=1}^{N-1} \bar{b}_{kn} 2^{-n} + 2^{-(N-1)}. \tag{4.5}$$

Здесь $\bar{b}_{kn}, \bar{b}_{k0}$ – инверсные значения бит b_{kn}, b_{k0}.

С учетом выражений (4.2) и (4.5) выражение (4.4) примет следующий вид:

$$x_k = \frac{1}{2}\left[-b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} - \left(-\bar{b}_{k0} + \sum_{n=1}^{N-1} \bar{b}_{kn} 2^{-n} + 2^{-(N-1)} \right) \right] =$$

$$= \frac{1}{2}\left[-(b_{k0} - \bar{b}_{k0}) + \sum_{n=1}^{N-1} (b_{kn} - \bar{b}_{kn}) 2^{-n} - 2^{-(N-1)} \right]. \quad (4.6)$$

Для упрощения обозначений целесообразно ввести новые переменные, значения которых лежат в пределах ±1:

$$c_{kn} = b_{kn} - \bar{b}_{kn},\ c_{k0} = -(b_{k0} - \bar{b}_{k0}). \quad (4.7)$$

С учетом выражения (4.7) можно записать

$$x_k = \frac{1}{2}\left[\sum_{n=1}^{N-1} c_{kn} 2^{-n} - 2^{-(N-1)} \right]. \quad (4.8)$$

Подставив данное выражение в (4.1), можно получить уравнение для вычисления требуемой суммы произведений:

$$y = \frac{1}{2}\sum_{k=1}^{K} A_k \left[\sum_{n=0}^{N-1} c_{kn} 2^{-n} - 2^{-(N-1)} \right] = \sum_{n=0}^{N-1} Q(b_n) 2^{-n} + 2^{-(N-1)} Q(0), \quad (4.9)$$

где функции $Q(b_n)$ и $Q(0)$ задаются следующим образом:

$$Q(b_n) = \sum_{k=1}^{K} \frac{A_k}{2} c_{kn},\ Q(0) = -\sum_{k=1}^{K} \frac{A_k}{2}. \quad (4.10)$$

Функция $Q(b_n)$ имеет 2^K возможных значений, которые могут быть размещены в памяти в соответствии с содержимым табл. 4.2.

Значения Q в нижней части таблицы являются зеркальным отражением верхней части, но с противоположным знаком. Из приведенной таблицы можно увидеть, что, выполняя сложение по модулю два бит b_{1n} с оставшимся набором b_{2n}, b_{3n}, ... b_{Kn}, можно получить адрес для чтения из памяти соответствующего значения Q (без учета знака). Используя сигнал b_{1n} в качестве сигнала управления сложением/вычитанием, можно получить требуемый знак считываемого из памяти значения. Во время появления на входах знаковых разрядов сигнал управления сложением/вычитанием должен быть проинвертирован, что может быть реализовано путем сложения по модулю сигналов b_{1n} и T_S. Соответствующая структурная схема приведена на рис. 4.3. Благодаря такой реализации размер таблицы памяти уменьшается до 2^{K-1} слов.

Когда младшие биты слов x_k осуществляют адресацию, значение, считываемое из памяти, в соответствии с выражением (4.9) должно быть скорректировано на величину $Q(0)$. Это осуществляется путем коммутации выхода регистра, содержащего значение $Q(0)$, с помощью переключателя SW_B на вход аккумулятора. Последующие значения, считываемые из памяти, суммируются со сдвинутой вправо на 1 разряд суммой.

Объем требуемой памяти можно уменьшить и другим способом. На рис. 4.4 показан КИХ-фильтр в обычной реализации. Для такой реализации фильтра 4 порядка требуется 16 ячеек памяти.

Таблица 4.2

Содержимое блока памяти (2^K слов)

Входной код					Содержимое памяти, Q
b_{1n}	b_{2n}	b_{3n}	…	b_{Kn}	
0	0	0	0	0	$-1/2\cdot(A_1+A_2+A_3+\ldots+A_K)$
0	0	0	0	1	$-1/2\cdot(A_1+A_2+A_3+\ldots-A_K)$
0	0	1	0	0	$-1/2\cdot(A_1+A_2-A_3+\ldots+A_K)$
0	0	1	0	1	$-1/2\cdot(A_1+A_2-A_3+\ldots-A_K)$
0	1	0	0	0	$-1/2\cdot(A_1-A_2+A_3+\ldots+A_K)$
0	1	0	0	1	$-1/2\cdot(A_1-A_2+A_3+\ldots-A_K)$
0	1	1	0	0	$-1/2\cdot(A_1-A_2-A_3+\ldots+A_K)$
0	1	1	0	1	$-1/2\cdot(A_1-A_2-A_3+\ldots-A_K)$
0	1	1	…	1	…
0	1	1	1	1	$-1/2\cdot(A_1-A_2-A_3-\ldots-A_4)$
1	0	0	0	0	$1/2\cdot(A_1-A_2-A_3-\ldots-A_K)$
1	0	0	0	1	$1/2\cdot(A_1-A_2-A_3-\ldots+A_K)$
1	0	1	0	0	$1/2\cdot(A_1-A_2+A_3-\ldots-A_K)$
1	0	1	0	1	$1/2\cdot(A_1-A_2+A_3-\ldots+A_K)$
1	1	0	0	0	$1/2\cdot(A_1+A_2-A_3-\ldots-A_K)$
1	1	0	0	1	$1/2\cdot(A_1+A_2-A_3-\ldots+A_K)$
1	1	1	0	0	$1/2\cdot(A_1+A_2+A_3-\ldots-A_K)$
1	1	1	0	1	$1/2\cdot(A_1+A_2+A_3-\ldots+A_K)$
1	1	1	..	1	…
1	1	1	1	1	$1/2\cdot(A_1+A_2+A_3+\ldots+A_K)$

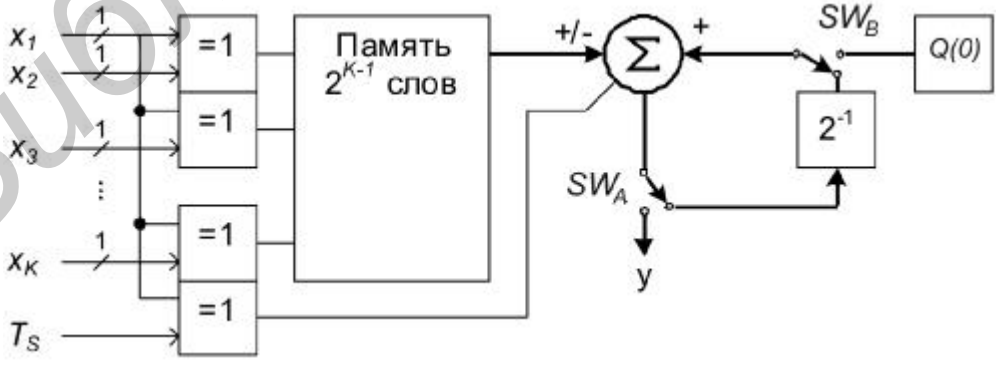

Рис. 4.3. Реализация вычисления на распределенной арифметике (память 2^{K-1} слов)

С увеличением порядка фильтра экспоненциально возрастает и объем требуемой памяти. Например, для КИХ-фильтра 128 порядка потребуется память на 2^{128} ячеек. Эта проблема может быть решена путем разбиения фильтра на меньшие подфильтры и суммирования их выходов. Фильтр порядка K может быть разделен на m меньших фильтров (рис. 4.5), порядок каждого равен k ($K = m \times k$). Теперь общий порядок фильтра K не имеет решающего значения для латентности фильтра.

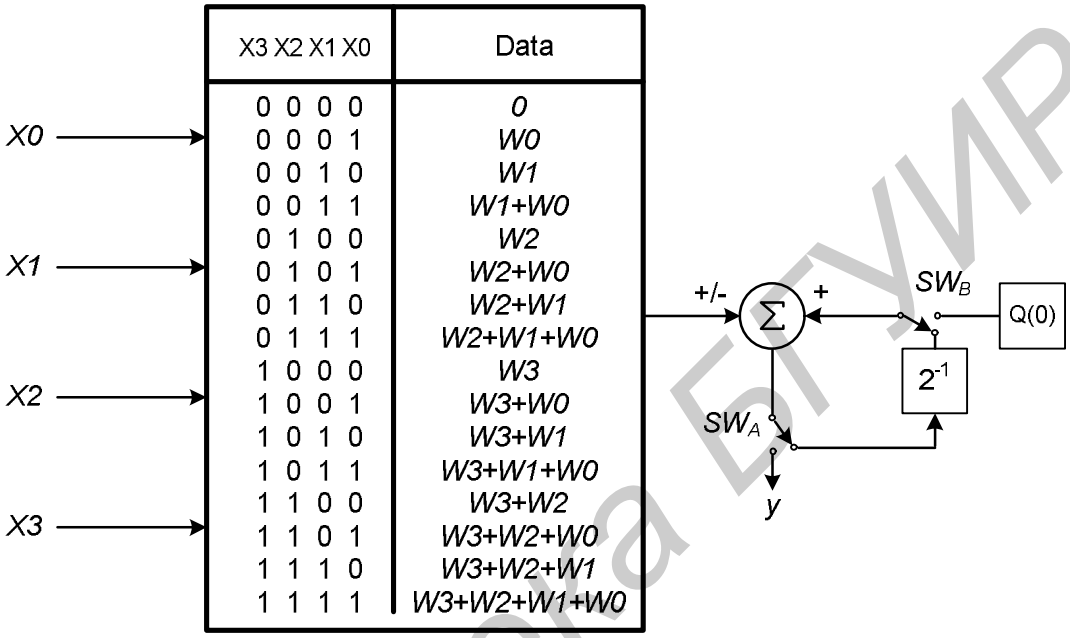

Рис. 4.4. Обычная реализация КИХ-фильтра 4-го порядка на распределенной арифметике

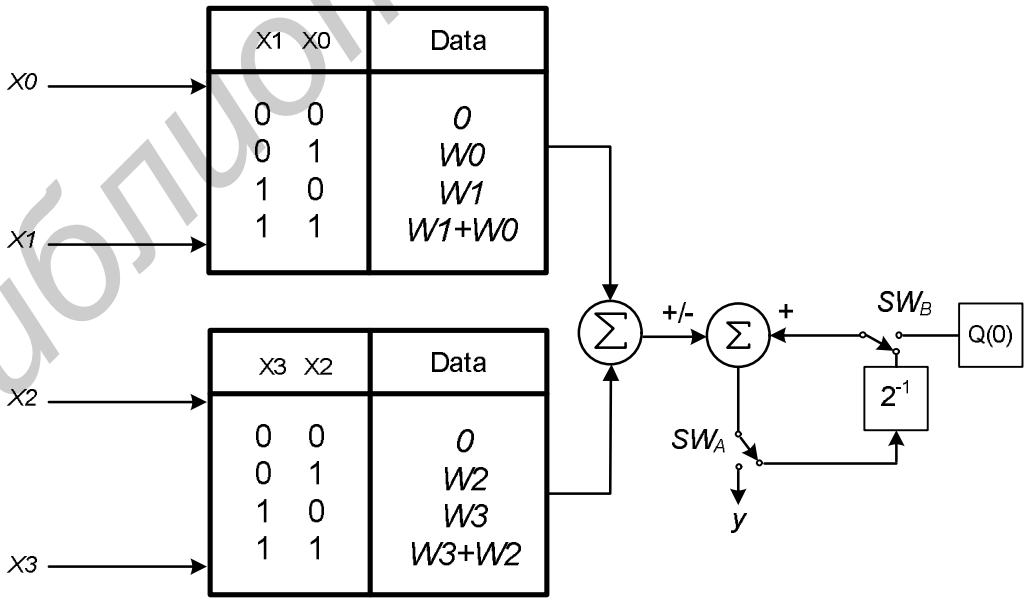

Рис. 4.5. КИХ-фильтр 4 порядка с 2 таблицами

Например, если $K = 128$, то в обычной реализации понадобится 2^{128} ячеек, а выбрав $m = 32$ и $k = 4$, можно снизить требования к памяти до 512 элементов.

4.1.3. Увеличение производительности распределенной арифметики

Увеличение скорости вычисления выражения (4.1) с помощью распределенной арифметики может быть достигнуто двумя способами, которые базируются на разделении входных слов на L частей (причем N должно быть кратно L):

- с использованием большего количества блоков памяти;
- путем экспоненциального увеличения размера памяти.

Следовательно, в первом случае необходимо использовать L блоков памяти одинакового размера (2^{K-1} слов). Во втором случае блок памяти один, но емкость его возрастает до значения $½ \cdot 2^{KL}$ и длина входных переменных увеличивается пропорционально значению $log_2 L$.

Поскольку во втором случае резкое увеличение занимаемой памяти для ускорения алгоритма приводит к потере выигрыша в занимаемой площади, то дальнейшее внимание будет уделено первому случаю. Использование первого метода для увеличения частоты работы приблизительно в 2 раза ($L = 2$) потребует двукратного увеличения емкости памяти. Данный подход, по сути, является типичным распараллеливанием процесса вычисления.

Поскольку в схемах, приведенных на рис. 4.1–4.3, поступление данных происходит по 1 биту за 1 такт, то такие схемы называются 1-BAAT (one-bit-at-a-time). Аналогично при одновременном поступлении подаче L бит схему можно обозначить как L-BAAT.

Выражения (4.9) и (4.10) для произвольного варианта L-BAAT примут следующий вид:

$$y = \sum_{n=0}^{(N/L)-1} Q(b_n) 2^{-nL} + 2^{-(B-L)} Q(0), \tag{4.11}$$

где функции $Q(b_n)$ и $Q(0)$ задаются следующим образом:

$$Q(b_n) = \sum_{k=1}^{K} \sum_{l=0}^{L-1} \frac{A_k}{2} 2^{-l} c_{k,nL+l}\ ,\ Q(0) = -2^{-L} \sum_{k=1}^{K} A_k\ . \tag{4.12}$$

Структурная схема, получаемая для $L = 2$ (2-BAAT), приведена на рис. 4.6.

Как можно видеть из полученной структуры, для данного варианта реализации необходимо использовать сумматор/вычитатель с 3 входами, 2 из которых управляются соответствующими сигналами, а также элемент сдвига на 2 разряда в цепи аккумулирования. Кроме того, необходимо введение элемента сдвига на 1 разряд выходного значения второго блока памяти. Все остальные блоки по сравнению со схемой 1-BAAT остались без изменения.

Дополнительный элемент сдвига может не использоваться, если длину слов, хранимых во втором блоке памяти, уменьшить на 1 разряд, как это показано на рис. 4.7. Однако это изменение незначительно ухудшит точность получаемых результатов.

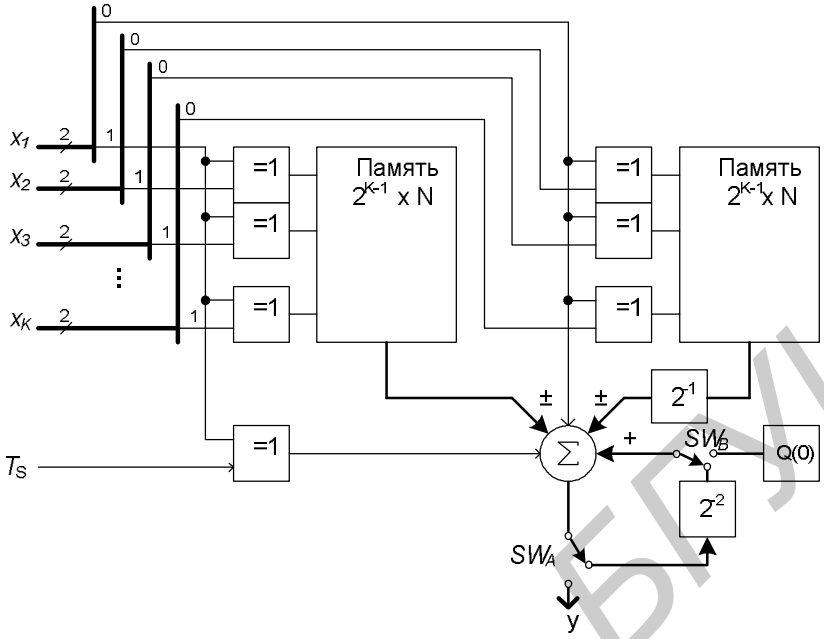

Рис. 4.6. Реализация вычисления на распределенной арифметике (2-ВААТ)

Аналогичным образом можно реализовать структуры для больших значений *L*. Однако при этом необходимо учитывать, что кроме ожидаемого увеличения размера занимаемой памяти это вызовет увеличение снижения частотных характеристик из-за роста задержек, вносимых сумматором/вычитателем, реализуемого на практике с помощью дерева сумматоров. Следовательно, для каждого непосредственного применения необходима комплексная оценка эффективности реализации по нескольким наиболее важным параметрам.

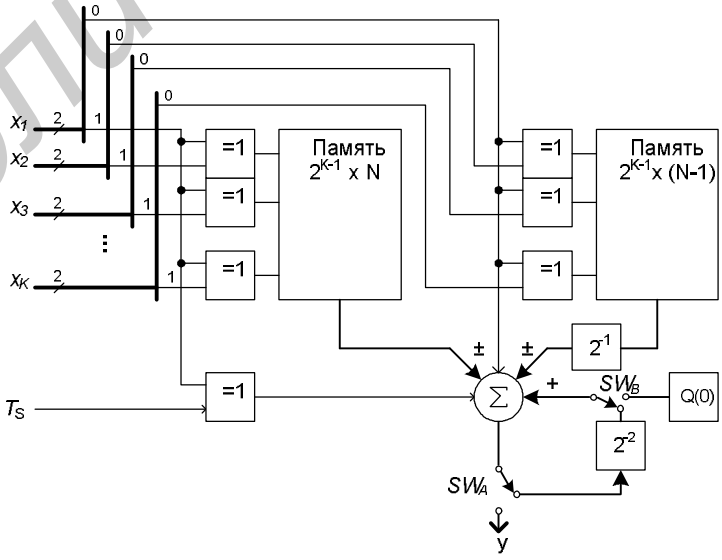

Рис. 4.7. Реализация вычисления на распределенной арифметике с сокращенным размером памяти (2-ВААТ)

Таблица 4.3

Варианты заданий

Вариант	Фильтр
1	Порядок фильтра 4, количество таблиц 1
2	Порядок фильтра 6, количество таблиц 3
3	Порядок фильтра 8, количество таблиц 2
4	Порядок фильтра 12, количество таблиц 4

4.2. Порядок выполнения работы

1. Изучить теоретические сведения по теме лабораторной работы.
2. Получить у преподавателя задание для выполнения практической части работы (табл. 4.3).
3. На языке VHDL составить структурное описание КИХ-фильтра в соответствии с вариантом.
4. Синтезировать КИХ-фильтр, проверить его работу. Оценить быстродействие, латентность, количество занимаемых в ПЛИС ресурсов и энергопотребление.
5. Показать результат работы устройства преподавателю.
6. Оформить и защитить отчет по лабораторной работе.

4.3. Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения.
3. Схема синтезируемого устройства.
4. Описание устройства на VHDL с комментариями, листинг файла теста.
5. Временные диаграммы симуляции работы устройства.
6. Листинг отчета синтезатора о быстродействии, количестве занятых ресурсов.
7. Выводы по работе.

5. ЛАБОРАТОРНАЯ РАБОТА №5
ИССЛЕДОВАНИЕ БАЗОВЫХ АЛГОРИТМОВ ЦОС НА РАСПРЕДЕЛЕННОЙ АРИФМЕТИКЕ

Цель работы: составить VHDL-описание устройств на базе распределенной арифметики в соответствии с вариантом (табл. 5.1), получить временные диаграммы его работы, оценить объем занимаемых ресурсов и быстродействие.

Таблица 5.1

Варианты заданий

Вариант	Устройство
1	КИХ-фильтр, порядок 8
2	БИХ-фильтр, порядок 4
3	Генератор синусоиды на основе рекурсивных вычислений
4	Генератор синусоиды на основе звена второго порядка

5.1. Порядок выполнения работы

1. Изучить теоретические сведения по теме лабораторной работы.
2. Получить у преподавателя задание для выполнения практической части работы.
3. На языке VHDL составить структурное описание КИХ-фильтра в соответствии с вариантом.
4. Синтезировать КИХ-фильтр, проверить его работу. Оценить быстродействие, латентность, количество занимаемых в ПЛИС ресурсов и энергопотребление.
5. Показать результат работы устройства преподавателю.
6. Оформить и защитить отчет по лабораторной работе.

5.2. Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения.
3. Схема синтезируемого устройства.
4. Описание устройства на VHDL с комментариями, листинг файла теста.
5. Временные диаграммы симуляции работы устройства.
6. Листинг отчета синтезатора о быстродействии, количестве занятых ресурсов.
7. Выводы по работе.

ПРИЛОЖЕНИЕ *А*

А.1. Описание отладочной платы ML401

А.1.1. Общий вид платы ML401

Отладочная плата ML401 показана на рис. А.1 (лицевая сторона) и на рис. А.2 (обратная сторона). Компоненты платы пронумерованы, и после рисунков следует подробное описание каждой секции.

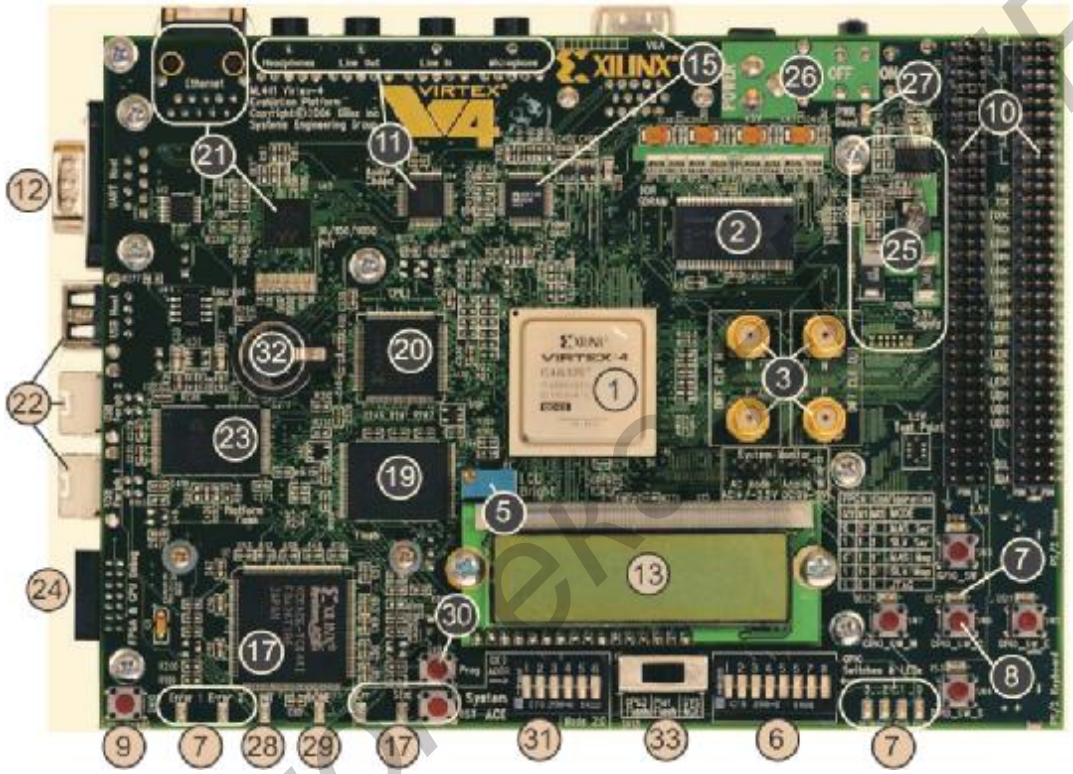

Рис. А.1. Подробное описание компонентов отладочной платы Virtex-4 ML401 (лицевая сторона)

А.1.2. Внешние запоминающие устройства

DDR SDRAM (позиция 2 на рис. А.2)

На плате установлены две микросхемы DDR SDRAM (U4 и U5) Infineon HYB25D256160BT-7 (или совместимых с ними) суммарной ёмкостью 64 Мб. Каждая микросхема имеет 16-разрядную шину данных. Вместе они формируют 32-битную шину данных, способную работать на частоте до 266 МГц. Все сигналы DDR SDRAM подключены к дополнительному опорному напряжению (V_{TT}) 1,25 В через ограничительные резисторы 47 Ом. Плата спроектирована таким образом, чтобы совпадали размеры трассировочных путей для всех сигналов управления DDR-памятью и сигналов данных, кроме сигналов синхронизации и DDR Loop trace (см. «DDR Clock Signal» и «DDR Loop Signal»).

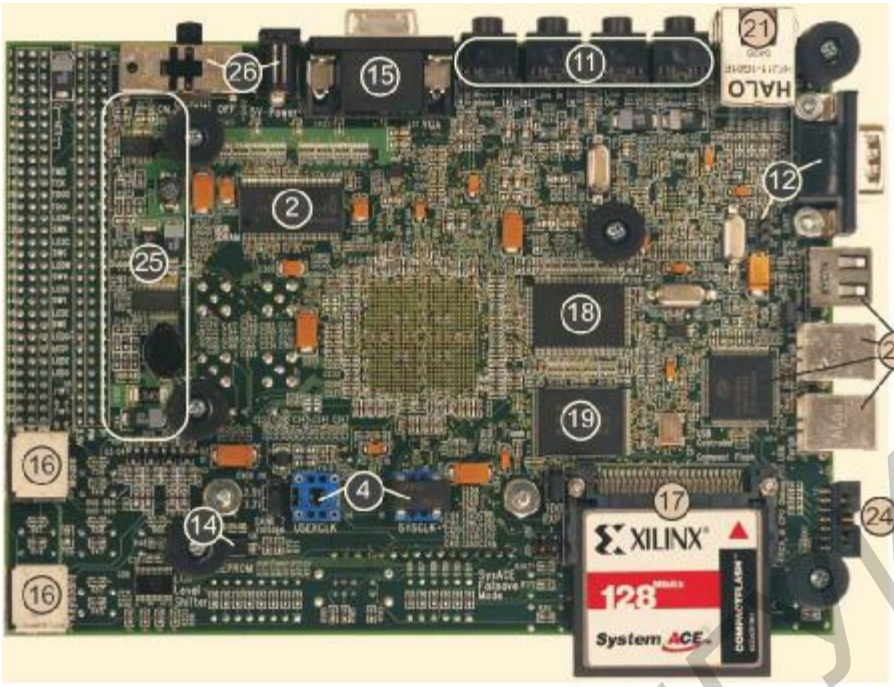

Рис. А.2. Подробное описание компонентов отладочной платы Virtex-4 ML401 (обратная сторона)

Сигнал синхронизации DDR-памяти передается из ПЛИС как одна дифференциальная пара, которая управляет обеими микросхемами DDR-памяти. Линии синхронизации спроектированы таким образом, чтобы задержки на них совпадали с задержками других сигналов управления DDR-памятью и сигналов данных. Сигнал синхронизации также подается по цепи обратной связи на ПЛИС для устранения разбега фронтов при помощи DCM в Virtex-4. Плата спроектирована таким образом, что сигнал синхронизации DDR-памяти достигает контакта цепи обратной связи ПЛИС одновременно с тем, как он поступает на входы микросхем DDR-памяти.

На плате организована обратная связь. Сигнал обратной связи управляется трассами сигналов синхронизации и затем поступает в ПЛИС с задержкой, равной сумме задержек трассировки сигналов синхронизации. Обратная связь может быть использована в высокоскоростных контроллерах памяти для того, чтобы помочь скомпенсировать физические задержки прохождения сигналов между микросхемами ПЛИС и DDR-памяти.

Синхронная ZBT SRAM (позиция 18 на рис. А.2)

Синхронная ZBT SRAM (Cypress CY7C1354B) является высокоскоростной, с коротким временем запаздывания внешней памятью для ПЛИС. Организация памяти соответствует 256К x 36 бит. Подобная организация обеспечивает поддержку 4 бит четности для 32-битной шины данных.

Примечание. Для памяти SRAM и флэш-памяти используется одна общая шина данных.

Микросхемы линейной флэш-памяти (позиция 19 на рис. А.2)

На плату установлены две 32-мегабайтные микросхемы линейной флэш-памяти Micron MT28F320J3RG-11 ET, совместимые с Intel StrataFlash. Эта память является энергонезависимой памятью и может использоваться для хранения данных, программ и конфигураций. Каждая микросхема флэш-памяти имеет 16-битный выход. Вместе они формируют 32-битную шину данных, которая также используется SRAM. В сочетании с CPLD флэш-память может также использоваться для загрузки ПЛИС.

Примечание. Вход сброса кодека AC97 объединен с входом сброса микросхем флэш-памяти. Сигнал сброса формируется при включении питания или сбросе системы.

Память для хранения конфигурации – Xilinx XCF32P Platform Flash (позиция 23 на рис. А.1)

Память для хранения конфигурации Platform Flash XCF32P представляет собой удобное и простое решение хранения конфигурации ПЛИС. Она может хранить до четырех различных вариантов конфигурации, доступ к каждому из этих вариантов конфигурации осуществляется с помощью переключателей адреса конфигурации. Чтобы использовать память Platform Flash для загрузки конфигурации в ПЛИС, переключатель выбора конфигурации (SW12) должен быть установлен в положение Plat Flash.

Память Platform Flash может конфигурировать ПЛИС Virtex-4 в режимах: ведущем последовательном, ведущем параллельном, подчиненном последовательном и подчиненном параллельном. Запись конфигурационных файлов в память Platform Flash осуществляется с помощью программы iMPACT через JTAG-цепь платы.

IIC-шина с 4 Кбит EEPROM (позиция 14 на рис А.2)

IIC EEPROM (Microchip Technology 24LC04B-I/ST) предназначена для хранения неизменяющихся данных, таких как Ethernet MAC-адреса. Аппаратная защита EEPROM от записи отключена. IIC-шина работает с сигналами напряжением 2,5 В при частоте до 400 кГц. На плате расположены также pull up-резисторы IIC-шины.

IIC-шина подключена к разъёму расширения, что позволяет пользователю подключать дополнительные IIC-устройства и управлять ими с помощью IIC-контроллера в ПЛИС. Для использования IIC-шины расширения необходимо наличие на плате расширения дополнительных pullup-резисторов. Наличие на плате расширения транзисторов двустороннего сдвига уровня позволяет при передаче сигналов по IIC использовать напряжение от 2,5 до 5 В.

А.1.3. ПЛИС

ПЛИС XC95144XL (позиция 20 на рис. А.1)

ПЛИС типа CPLD XC95144XL подсоединена к флэш-памяти и конфигурационному интерфейсу ПЛИС Virtex-4. Это соединение CPLD поддерживает приложения, в которых программирование ПЛИС осуществляется из флэш-памяти. CPLD программируется через главную JTAG-цепочку платы. CPLD подключена таким образом, что она может конфигурировать ПЛИС Virtex-4 в режимах: ведущем последовательном, ведущем параллельном, подчиненном последовательном и подчиненном параллельном. Для конфигурирования ПЛИС через CPLD из флэш-памяти переключатель выбора конфигурации (SW12) необходимо поставить в положение CPLD Flash.

ПЛИС Virtex-4 (позиция 1 на рис. А.1)

На плате ML401 установлена ПЛИС типа FPGA Virtex-4 XC4VLX24-FF668-10.

Блоки ввода–вывода в ПЛИС Virtex-4, установленные на плате, организованы в виде групп, называемых банками ввода-вывода, каждый банк имеет раздельное питание V_{CCO}. Всего 11 банков, из которых используются только первые 10. Последний банк подключен к питанию, но не используется. Значения напряжения ввода–вывода для каждого банка даны в табл. А.1.

Таблица А.1

Напряжения питания на банках ввода–вывода

Банк ПЛИС	Значение напряжения V_{CCO}
0	3,3 В
1	3,3 В
2	3,3 В
3	2,5 В
4	3,3 В
5	2,5 В
6	2,5 В
7	2,5 В или 3,3 В (выбирается пользователем при помощи перемычки J16)
8	3,3 В
9	3,3 В
10	3,3 В (подключен к питанию, но контакты ввода–вывода не используются)

Некоторые банки ПЛИС Virtex-4 поддерживают DCI (цифровое управление импедансом).

A.1.4. Сопряжение с внешними устройствами

System ACE контроллер и CompactFlash разъём (позиция 17 на рис. А.1)

Конфигурационный контроллер System ACE позволяет программировать ПЛИС из карты CompactFlash через порт JTAG. Через порт JTAG можно загружать как аппаратные, так и программные данные. Контроллер System ACE поддерживает до восьми вариантов конфигурации в одной карте CompactFlash. Переключатели адреса конфигурации позволяют пользователю выбрать для работы один из восьми вариантов конфигурации.

Светодиоды ошибки и статуса System ACE отображают состояние контроллера System ACE:

• мигающий красный светодиод ошибки сигнализирует об отсутствии карты CompactFlash;

• непрерывное свечение красного светодиода ошибки сигнализирует об ошибке конфигурирования;

• мигающий зеленый светодиод статуса показывает, что процесс конфигурирования продолжается;

• непрерывное свечение зеленого светодиода статуса показывает, что загрузка успешно завершена.

При установке карты CompactFlash в разъём System ACE конфигурирование начинается автоматически. При нажатии кнопки перезагрузки System ACE начинается перепрограммирование ПЛИС.

Плата также поддерживает отказоустойчивый режим System ACE, при котором в случае регистрации неудачной попытки конфигурирования происходит автоматический возврат к предварительно установленному варианту конфигурации. Отказоустойчивый режим приводится в действие с помощью установки двух перемычек J29 и J30 (в горизонтальном или вертикальном положении).

Внимание. Осторожно устанавливайте карту CompactFlash с открытыми металлическими поверхностями. Неправильная установка может привести к короткому замыканию с металлизированными дорожками или компонентами на плате.

Порт MPU System ACE подсоединен к ПЛИС. Это соединение позволяет ПЛИС использовать контроллер System ACE для реконфигурирования системы или для доступа к карте CompactFlash как к общей файловой системе FAT. Шина данных порта MPU System ACE соединена с шиной данных USB-контроллера.

Для конфигурирования с помощью контроллера System ACE переключатель выбора конфигурации (SW12) должен быть поставлен в положение SYS ACE.

USB-контроллер с главным и периферийными портами (позиция 22 на рис. А.1)

Встроенный USB-контроллер Cypress CY7C67300 обеспечивает подключение к плате устройств по USB-интерфейсу. USB-контроллер поддерживает главный и периферийный режимы работы. USB-контроллер имеет два процессора последовательного интерфейса (SIE), которые могут использоваться независимо. SIE1 подключен к разъёму USB – главный 1 (J19) – и к разъёму USB – периферийный 1 (J2). SIE2 подключен только к разъёму USB периферийный 2.

Примечание. Если используется SIE1, то порт при загрузке может быть сконфигурирован на подключение либо к главному, либо к периферийному разъёму, но не к обоим разъёмам одновременно.

USB-контроллер имеет внутренний микропроцессор, который осуществляет выполнение USB-команд. Микропрограмма для этого процессора может храниться в ее собственной, специально предназначенной для этой цели памяти IIC EEPROM (U17) либо может быть загружена из главного компьютера через периферийный разъём. Последовательный порт USB контроллера подключен к J27 через трансивер RS232, что помогает в процессе отладки.

Разъёмы расширения (позиция 10 на рис. А.1)

Отладочная плата имеет разъёмы расширения, предназначенные для расширения или адаптации платы для работы с внешними устройствами. В разъёмах расширения используются стандартные 0,1-дюймовые штырьки. Разъёмы расширения соединены с несимметричными и дифференциальными парами ввода–вывода ПЛИС, заземлением, питанием 2,5 /3,3 / 5 В, JTAG-цепью и IIC-шиной.

Путем соответствующей установки перемычек разъём расширения позволяет также соединить JTAG-цепь платы с платой расширения.

IIC-шина платы также может быть расширена с помощью разъёма расширения с целью подключения дополнительных IIC-устройств к данной шине. При расширении IIC-шины пользователь должен установить нагрузочные IIC pullup-резисторы на плату расширения. Транзисторы двунаправленного сдвига уровня позволяют плате расширения передавать по IIC-шине сигналы напряжением от 2,5 до 5 В.

Соединения источника питания с разъёмами расширения обеспечивают наличие напряжений 2,5, 3,3, 5 В и «земли» на контактах разъёма. Если плата расширения отбирает значительную часть мощности у платы ML401, пользователь должен убедиться, что плата способна обеспечить эту мощность.

Последовательный порт RS-232 (позиция 12 на рис. А.1)

На плате ML40x расположен один последовательный порт RS-232 DB-9(male) типа, что позволяет ПЛИС передавать последовательные данные на другие устройства. Последовательный порт является в данном случае главным устройством. Таким образом, для подключения платы к последовательному порту ком-

пьютера обычно используется нуль-модемный кабель. Последовательный порт работает в диапазоне до 115 200 бод. Интерфейсная микросхема используется для сдвига уровня напряжения для сигналов ПЛИС и RS-232.

Примечание. ПЛИС подсоединена только к информационным выводам TX и RX последовательного порта. Таким образом, другие сигналы типа RS-232, в том числе сигналы аппаратного контроля передачи, не используются. При обмене данными с компьютером контроль передачи необходимо отключить.

Вторичный последовательный интерфейс можно использовать через разъём J27 с целью отладки микросхемы USB-контроллера. Через разъём J27 выведены TX и RX сигналы RS-232 и «земля».

Порты PS/2 для мыши и клавиатуры(позиция 16 на рис. А.2)

На плате ML401 расположены два порта PS/2: для мыши (J17) и для клавиатуры (J18). Транзисторы двунаправленного смещения уровня позволяют согласовать интерфейсные сигналы ввода/вывода ПЛИС (2,5 В) с сигналами ввода/вывода портов PS/2 (5 В). Порты PS/2 отладочной платы запитываются непосредственно от основного разъёма питания (5 В), от которого запитывается и остальная плата.

Внимание. Убедитесь, что потребляемая мощность всех подключенных к PS/2 устройств не перегружает адаптер переменного тока платы

Аудиокодек Stereo AC97 (позиция 11 на рис. А.1)

На плате ML401 установлена микросхема аудиокодека AC97 (U14). Аудиокодек National Semiconductor LM4550 поддерживает 16-битный стереосигнал с частотой выборки до 48 кГц. Частота выборки для записи и воспроизведения может различаться.

Примечание. Вход сброса кодека AC97 объединен с входом сброса микросхем флэш-памяти. Сигнал сброса формируется при включении питания или сбросе системы.

На плате установлены отдельные разъёмы для микрофона, линейного входа, линейного выхода и наушников. Все разъёмы, кроме разъёма для микрофона, предназначены для передачи стереосигналов. Разъём для наушников управляется внутренним усилителем кодека (50-мВт). В табл. А.2 приведены данные по всем разъёмам.

Таблица А.2

Аудиоразъёмы платформы ML401

Обозначение	Функция	Стерео/Моно
J11	Микрофон – вход	Моно
J12	Аналоговая линия – вход	Стерео
J13	Аналоговая линия – выход	Стерео
J14	Наушники – выход	Стерео

Трехскоростной (10/10/1000) трансивер Ethernet (позиция 21 на рис. А.1)

На плате ML401 установлена микросхема физического уровня Marvell Alaska PHY(88E1111), работающая на скоростях 10/100/1000 Мбит/с. Плата поддерживает интерфейсные режимы MII, GMII и RGMII (при работе с ПЛИС). PHY подсоединен к разъёму RJ-45 Halo HFJ11-1G01E. Кварцевый генератор (25 МГц) подает синхроимпульсы на PHY.

JTAG-разъём (позиция 24 на рис. А.1)

JTAG-разъём (J20) позволяет программировать и осуществлять отладку конфигурации ПЛИС. JTAG порт совместим с загрузочным кабелем Parallel Cable IV фирмы Xilinx. JTAG-цепь также может быть дополнена платой расширения после соответствующей установки перемычки J26.

Выход VGA (позиция 15 на рис. А.1)

Порт выхода VGA (P2) предназначен для подключения внешнего видеомонитора. В таблице А.3 приведены основные параметры используемой ЦАП.

Таблица А.3

Параметры ЦАП

Скорость	Описание	Видеомонитор
50 МГц	24-битовая шина видеоданных, подсоединенная к ПЛИС	Аналоговые устройства ADV7125KST50

А.1.5. Переключатели и устройства отображения информации

DIP-переключатели обего назначения (позиция 6 на рис. А.1)

Восемь DIP-переключателей общего назначения подсоединены к пользовательским контактам ввода–вывода ПЛИС. В табл. А.4 обозначены эти соединения.

Таблица А.4

Подключения DIP-переключателей (SW1) к ПЛИС

SW1	Контакт ПЛИС
1	R20
2	R19
3	T26
4	U26
5	U23
6	V23
7	U25
8	U24

DIP-переключатели адреса конфигурации и режима

Этот 6-позиционный DIP-переключатель управляет адресом конфигурации и режимом конфигурирования ПЛИС.

С помощью трех самых левых переключателей выбирается один из восьми адресов конфигурации. Эти три DIP-переключателя позволяют контроллеру System ACE и CPLD использовать до восьми различных вариантов конфигурации в соответствии с их положениями. Память Platform Flash поддерживает до четырех различных вариантов конфигурации.

Три самых правых DIP-переключателя устанавливают контакты конфигурации режима ПЛИС M2, M1 и M0 в положения, указанные в табл. А.5.

Таблица А.5

Положения DIP-переключателей режима конфигурации

	M	M	Режим
0	0	0	Master Serial (Ведущий последовательный)
1	1	1	Slave Serial (Подчиненный последовательный)
0	1	1	Master Parallel (Ведущий параллельный) (SelectMAP)
1	1	0	Slave Parallel (Подчиненный параллельный) (SelectMAP)
1	0	1	JTAG

Переключатель Program(позиция 30 на рис. А.1)

Данный переключатель при нажатии заземляет контакт ПЛИС Prog. Эта операция приводит к очистке ПЛИС

Переключатель выбора источника конфигурации (позиция 33 на рис. А.1)

Переключатель (SW12) выбора источника конфигурации позволяет выбрать способ программирования ПЛИС из следующих: System ACE, Platform Flash и линейная флэш-память/CPLD. При любом из выбранных способов убедитесь, что переключатели выбора режима конфигурации ПЛИС поставлены в положение, соответствующее выбранному способу. Разъём PC4 позволяет осуществлять загрузку через JTAG, а также проводить отладку платы в независимости от положения переключателя выбора источника конфигурирования.

Пользовательские светодиоды и светодиоды индикации ошибок (позиция 7 на рис. А.1)

На плате расположено 11 светодиодов, которые непосредственно управляются микросхемой Virtex-4:

- четыре зеленых светодиода общего назначения расположены в ряд;
- пять зеленых светодиодов расположены по соседству с кнопками, ориентированными по принципу *Север-Восток-Юг-Запад-Центр*;
- два красных светодиода предназначены для индикации ошибочных состояний (таких, как ошибки шины), но могут быть также использованы для каких-либо других целей.

В табл. А.6 показаны состояния светодиодов и их подключения.

Светодиод INIT (позиция 28 на рис. А.1)

Светодиод INIT загорается после включения системы, тем самым сигнализируя о том, что подача напряжения на ПЛИС совершена и все внутренние переходные процессы при включении питания завершены.

Таблица А.6

Пользовательские светодиоды и светодиоды индикации

Обозначение	Маркировка	Цвет	Контакт ПЛИС
DS14	Светодиод *Север*	Зеленый	E2
DS11	Светодиод *Восток*	Зеленый	E10
DS3	Светодиод *Юг*	Зеленый	A5
DS13	Светодиод *Запад*	Зеленый	F9
DS12	Светодиод *Центр*	Зеленый	C6
DS15	GPIO светодиод 0	Зеленый	G5
DS4	GPIO светодиод 1	Зеленый	G6
DS5	GPIO светодиод 2	Зеленый	A11
DS6	GPIO светодиод 3	Зеленый	A12
DS205	Error 1	Красный	V6
DS2065	Error 2	Красный	L24

Светодиод DONE (позиция 29 на рис. А.1)

Светодиод DONE показывает статус контакта DONE в ПЛИС. При успешном завершении конфигурирования ПЛИС светодиод DONE загорается.

Светодиод индикатора питания (позиция 27 на рис. А.1)

Светодиод PWR Good загорается, когда подача напряжений 1,2, 2,5 и 3,3 В проходит в номинальном режиме. Выключенный, мигающий или слабо горящий светодиод PWR Good сигнализирует о возможном наличии ошибки, такой, как короткое замыкание или перегрузка.

Кнопки (активное состояние соответствует нажатой кнопке, позиция 8 на рис. А.1)

Пять кнопок общего назначения расположены по принципу *Север–Восток–Юг–Запад–Центр* (только кнопка *Центр* отмечена на рис. А.2).

В табл. А.7 указаны подключения пользовательских кнопок к выводам ПЛИС.

Кнопка сброса состояния процессора (активное состояние соответствует ненажатой кнопке, позиция 9 на рис. А.1)

Кнопка сброса состояния процессора предназначена для системного или пользовательского сброса. Данная кнопка подключена только к контакту ввода-

вывода ПЛИС, поэтому она может использоваться также в качестве кнопки общего назначения (табл. А.8).

Таблица А.7

Подключения кнопок к выводам ПЛИС

Обозначение	Маркировка/Определение	Контакт ПЛИС
SW3	GPIO переключатель *Север*	E7
SW5	GPIO переключатель *Восток*	F10
SW4	GPIO переключатель *Юг*	A6
SW7	GPIO переключатель *Запад*	E9
SW6	GPIO переключатель *Центр*	B6

Таблица А.8

Подключение кнопки сброса состояния процессора

Обозначение	Маркировка/Определение	Контакт ПЛИС
SW10	FPGA CPU RESET	D6

ЖКИ с двумя строками по 16 символов (позиция 13 на рис. А.1)

На плате ML401 расположен ЖКИ с двумя строками по 16 символов (Lumex LCM-S01602DTR/M) для вывода текстовой информации. Потенциометр R1 предназначен для настройки контрастности ЖКИ. Интерфейс данных ЖКИ подсоединен к ПЛИС и поддерживает только 4-битный режим. Микросхема преобразования уровня обеспечивает сдвиг уровня напряжения между ПЛИС и ЖКИ.

А.1.6. Синхронизация и питание

Дифференциальный вход и выход синхронизации с SMA-разъёмами (позиция 3 на рис. А.1)

Высокостабильные сигналы синхронизации могут быть поданы на ПЛИС с использованием дифференциальных сигналов синхронизации, поступающих через SMA-разъёмы (50 Ом). Это позволяет внешнему источнику синхроимпульсов управлять дифференциальными входами синхронизации таким образом, чтобы синхроимпульсы напрямую поступали на глобальные тактовые входы ПЛИС.

Выходные дифференциальные сигналы синхронизации от ПЛИС могут быть выведены наружу через вторую пару SMA-разъёмов. Это позволяет ПЛИС управлять подачей высокостабильных сигналов синхронизации на внешнее устройство, например образец тестового оборудования. В табл. А.9 приведены номера контактов ПЛИС, подключенных к SMA-разъёмам.

Таблица А.9

Подключения дифференциальных сигналов синхронизации к SMA-разъёмам

Маркировка	Наименование синхросигнала	Контакт ПЛИС
J10	SMA_DIFF_CLK_IN_N	C12
J7	SMA_DIFF_CLK_IN_P	C13
J8	SMA_DIFF_CLK_OUT_N	D7
J9	SMA_DIFF_CLK_OUT_P	D8

Панели для генераторов синхроимпульсов (позиция 4 на рис. А.2)

Отладочная плата ML40x имеет две панели для кварцевых генераторов синхроимпульсов. Панели предназначены для установки в них стандартных тактовых генераторов типа LVTTL половинного размера. В одну предустановлен 100 МГц генератор (панель X1). В табл. А.10 показано подключение сигналов к контактам синхронизации ПЛИС. На генераторы подается напряжение питания 3,3 В.

Таблица А.10

Подключения панели генератора

Маркировка	Наименование синхросигнала	Контакт ПЛИС
X1	SYSCLK	AE14
X6	USERCLK	AD12

Источники питания на плате (позиция 25 на рис. А.1)

На плате реализована схема подачи напряжения питания на компоненты платы, вырабатывающая напряжения 1,2, 1,25, 1,8, 2,5 и 3,3 В. Подача напряжения 1,2, 2,5 и 3,3 В осуществляется импульсными преобразователями напряжения питания. Когда от этих регуляторов поступает сигнал о том, что они вырабатывают номинальные значения напряжения, включается светодиод PWR Good.

Адаптер переменного тока и переключатель подачи внешнего питания (позиция 26 на рис. А.1)

Плата ML401 поставляется с (5 В x 3 А) адаптером переменного тока (15 Вт). Разъём питания размером 2,1 x 5,5 мм имеет цилиндрическую форму (внутренняя часть имеет положительное напряжение). Кнопка питания осуществляет включение и выключение платы путем управления подачей 5 В на плату.

Батарея ключа кодирования (позиция 32 на рис. А.1)

Гнездо батареи на плате подсоединено к контакту V_{BATT} ПЛИС для хранения ключа кодирования конфигурационной последовательности ПЛИС.

А.2. Конфигурирование отладочной платы ML401

Общие сведения

В отладочной плате ML401 ПЛИС может быть сконфигурирована с помощью четырех основных устройств:

1) параллельного кабеля Parallel Cable IV (JTAG) или Parallel Cable III;
2) контроллера System ACE (JTAG);
3) памяти Platform Flash;
4) линейной флэш-памяти + CPLD.

JTAG (параллельный кабель Parrallel Cable IV(PC4), Parrallel Cable III и контроллер System ACE)

ПЛИС, память Platform Flash и CPLD могут быть сконфигурированы через JTAG порт. JTAG-цепь платы изображена на рис. А.3.

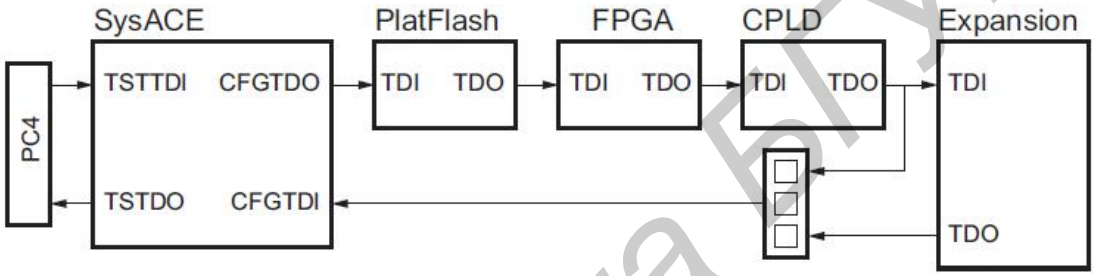

Рис. А.3. JTEG-цепь

Цепь начинается на разъёме для кабеля PC4 и проходит через контроллер System ACE, память Platform Flash, ПЛИС, CPLD и дополнительное расширение цепи для платы расширения. Перемычка J26 определяет, расширяется ли JTAG-цепь платой расширения.

JTAG-цепь может использоваться для программирования ПЛИС и доступа к ПЛИС с целью аппаратной и программной отладки. JTAG-цепь также используется для программирования памяти Platform Flash и CPLD.

PC4 JTAG-соединение позволяет загрузить конфигурацию из компьютера в ПЛИС с помощью программы iMPACT. PC4 позволяет также отладочным средствам, таким, как ChipScope Pro Analyzer или программным отладчикам, осуществлять доступ к ПЛИС.

Контроллер System ACE также может программировать ПЛИС через JTAG-порт. С помощью установленной карты CompactFlash или запоминающего устройства Microdrive конфигурационная информация может быть записана в ПЛИС и приведена в действие. Контроллер System ACE поддерживает до восьми вариантов конфигурации, которые выбираются с помощью трех DIP-переключателей адреса конфигурации.

Переключатель выбора источника конфигурации при необходимости использования контроллера System ACE должен быть установлен в положение SYS ACE.

При правильных установках контроллер System ACE начинает программировать ПЛИС в момент включения питания, если карта CompactFlash уже установлена в свой разъём, или непосредственно после того, как карта CompactFlash вставляется в разъём. Нажатие кнопки сброса System ACE также вызывает программирование ПЛИС от контроллера System ACE, если при этом карта CompactFlash уже вставлена.

Память Platform Flash

Память Platform Flash также может использоваться для программирования ПЛИС. Память Platform Flash может хранить до четырех вариантов конфигурации, выбор из которых осуществляется в зависимости от состояния двух младших разрядов адреса конфигурации на DIP-переключателях.

Плата сконструирована таким образом, что конфигурационные данные могут загружаться в память Platform Flash в следующих режимах: Master Serial (ведущий последовательный), Slave Serial (подчиненный последовательный), Master SelectMAP (ведущий параллельный), Slave SelectMAP (подчиненный параллельный). Используя программу iMPACT для программирования памяти Platform Flash, пользователь имеет возможность выбрать один из этих четырех режимов программирования ПЛИС. DIP-переключатели режима конфигурации на плате должны быть установлены таким образом, чтобы их состояние соответствовало выбранному режиму программирования памяти Platform Flash.

Переключатель выбора источника конфигурации при необходимости использования памяти Platform Flash должен быть установлен в состояние Plat Flash. При правильных установках память Platform Flash программирует ПЛИС в момент включения питания или при каждом нажатии кнопки Prog.

Линейная флэш-память и CPLD

Хранящиеся в линейной флэш-памяти данные могут быть считаны микросхемой CPLD и использованы для программирования ПЛИС Virtex-4. В зависимости от проекта в CPLD теоретически могут поддерживаться до восьми вариантов конфигурации.

Плата сконструирована таким образом, что конфигурационные данные могут загружаться в память Platform Flash в следующих режимах: Master Serial (ведущий последовательный), Slave Serial (подчиненный последовательный), Master SelectMAP (ведущий параллельный), Slave SelectMAP (подчиненный параллельный).

Переключатель выбора источника конфигурации при необходимости использования CPLD и линейной флэш-память должен быть установлен в состояние CPLD Flash.

При правильных установках CPLD программирует ПЛИС в момент включения питания или при каждом нажатии кнопки Prog.

ЛИТЕРАТУРА

1. ML401/ML402/ML403 Evalution Platform User Guide [Электронный ресурс]. – 2006. – Режим доступа: http://www.xilinx.com/support/documentation/.

2. ML40x Getting Started Tutorial [Электронный ресурс]. – 2006. – Режим доступа: http://www.xilinx.com/support/documentation/.

3. Virtex-4 User Guide [Электронный ресурс]. – 2007. – Режим доступа: http://www.xilinx.com/support/documentation/.

4. Гук, М. Аппаратные интерфейсы ПК. Энциклопедия / М. Гук. – СПб. : Питер, 2002. – 528 с.

5. Зотов, В. Ю. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WEBPACK ISE / В. Ю. Зотов. – М. : Горячая линия – Телеком, 2003. – 520 с.

6. Потехин, Д. С. Разработка систем цифровой обработки сигналов на базе ПЛИС / Д. С. Потехин, И. Е. Тарасов. – М. : Горячая линия – Телеком, 2007. – 248 с.

7. Максфилд, К. Проектирование на ПЛИС. Курс молодого бойца / К. Максфилд – М. : Издательский дом «Додека-XXI», 2007. – 408 с.

8. White, S. A. Application of distributed arithmetic to digital signal processing: a tutorial review / S. A. White // IEEE ASP Mag. – 1989. – №6.

Св. план 2008, поз. 20

Учебное издание

Петровский Александр Александрович
Новиков Алексей Евгеньевич
Машеров Дмитрий Александрович и др.

ПРОЕКТИРОВАНИЕ ЭВС С ДИНАМИЧЕСКИ РЕКОНФИГУРИРУЕМОЙ АРХИТЕКТУРОЙ

Лабораторный практикум
для студентов специальности 1-40 02 02
«Электронные вычислительные средства»
дневной формы обучения

Редактор Н. В. Гриневич
Корректор Е. Н. Батурчик

Подписано в печать 11.09.2008.	Формат 60×84 1/16.	Бумага офсетная.
Гарнитура «Таймс»	Печать ризографическая.	Усл. печ. л. 3,84.
Уч.-изд. л. 3,2.	Тираж 100 экз.	Заказ 195

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0056964 от 01.04.2004, ЛП №02330/0133666 от 30.04.2004.
220013, Минск, П. Бровки, 6.