

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

Кафедра защиты информации

С.Л. ПРИЩЕПА, Е.А. ИЛЬИНА

***ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ СХЕМ С ПОМОЩЬЮ САПР
MAX+PLUS II ФИРМЫ ALTERA***

Учебно-методическое пособие
по курсу
«САПР цифровых устройств»
для студентов специальности «Телекоммуникационные системы»
дневной формы обучения

Минск 2005

УДК 004.312(075.8)
ББК 32.844 я 73
П 79

Рецензент:
старший преподаватель кафедры СиУТ БГУИР С.М. Лапшин

Прищепа С.Л.

П 79 Проектирование цифровых схем с помощью САПР MAX+PLUS II фирмы Altera: Учебно-метод. пособие по курсу «САПР цифровых устройств» для студ. спец. «Телекоммуникационные системы» дневной формы обуч. / С.Л. Прищепа, Е.А. Ильина. – Мн.: БГУИР, 2005. – 52 с.: ил. ISBN 985-444-793-6

В учебно-методическом пособии произведен обзор существующих архитектур ПЛИС, приведены основные этапы проектирования цифровых устройств на базе ПЛИС, рассмотрены модули САПР MAX+PLUS II фирмы Altera, а также основные принципы работы в среде данного САПР.

Настоящее пособие предназначено для студентов, обучающихся по специальности «Телекоммуникационные системы».

УДК 004.312(075.8)
ББК 32.844 я 73

ISBN 985-444-793-6

© Прищепа С.Л., Ильина Е.А.,
2005
© БГУИР, 2005

Содержание

1. Обзор существующих архитектур ПЛИС
 - 1.1. Классификация ПЛИС по структурной организации
 - 1.2. Стандартные ПЛИС
 - 1.3. Макроматрицы (МАСН-устройства)
 - 1.4. Матричные таблицы (МАХ-устройства)
 - 1.5. Программируемые пользователем вентиляемые матрицы (FPGA)
 - 1.6. Сложные PLD (Complex PLD – CPLD)
 - 1.7. СБИС программируемой логики смешанной архитектуры (FLEX)
2. Основные сведения о САПР MAX+ plus II
 - 2.1. Проектирование с помощью САПР MAX+ plus II
 - 2.2. Составные части (программные модули) MAX+ plus II
 - 2.3. Файлы проекта, вспомогательные файлы и проекты
 - 2.4. Справочная система MAX+ plus II
 - 2.5. Меню Help
3. Основные этапы разработки цифрового устройства
4. Лабораторная работа № 1. Парные электронные «игральные кости»
 - 4.1. Структурная схема устройства
 - 4.2. Компиляция проекта
 - 4.3. Построение временных диаграмм
 - 4.4. Вычисление временных задержек
5. Лабораторная работа № 2. Проектирование ЦУ ввода-вывода данных
 - 5.1. Структурная схема устройства
 - 5.2. Создание автомата управления AvtOutBt
 - 5.3. Настройка параметризованных модулей
 - 5.4. Компиляция проекта
 - 5.5. Построение временных диаграмм
 - 5.6. Вычисление временных задержек
6. Контрольные вопросы
- Литература

1. Обзор существующих архитектур ПЛИС

1.1. Классификация ПЛИС по структурной организации

Программируемые логические интегральные схемы (ПЛИС – Programmable Logic Devices – PLD) представляют собой новую элементную базу, обладающую гибкостью заказных БИС и доступностью традиционной «жесткой» логики. Главным отличительным свойством ПЛИС является возможность их настройки на выполнение заданных функций самим пользователем. Современные ПЛИС характеризуются низкой стоимостью (1 – 2 дол. США), высоким быстродействием (до 3,5 нс), значительными функциональными возможностями (одна ПЛИС может заменить несколько сот корпусов традиционной «жесткой» логики), многократностью перепрограммирования, низкой потребляемой мощностью (позволяющей использовать их в изделиях с батарейным питанием), гибкостью архитектуры и др.

Доказательством перспективности новой элементной базы служит ежегодное появление новых, имеющих более совершенную архитектуру поколений ПЛИС, а также постоянно растущий объем выпуска ПЛИС ведущими зарубежными производителями микросхем: Advanced Micro Devices (AMD), Altera, Xilinx, Atmel, Intel, Texas Instruments и др.

Процесс проектирования цифрового устройства на основе ПЛИС заключается в описании его функционирования на входном языке используемого программного средства, выполнении автоматизированного синтеза, проведении моделирования и настройке выбранной ПЛИС с помощью программатора. При этом время разработки даже достаточно сложных проектов может составлять всего несколько часов. Для того чтобы изменить алгоритм работы устройства, достаточно перепрограммировать ПЛИС, причем отдельные ПЛИС допускают программирование (перепрограммирование) уже после их установки на плату. По существу разработка устройств на основе ПЛИС представляет собой новую технологию проектирования электронных схем, включая их изготовление и сопровождение.

История развития программируемой логики начинается с появления программируемых постоянных запоминающих устройств (ППЗУ -Programmable Read Only Memory – PROM) в начале 70-х гг. Первое время PROM использовались исключительно для хранения данных, позже их стали применять для реализации логических функций. Неудобство использования PROM в качестве логических преобразователей заключается в том, что логические функции перед записью в PROM необходимо приводить к совершенной дизъюнктивной нормальной форме (СДНФ), кроме того, емкость PROM не позволяла реализовать функции большого числа переменных. Специально для реализации систем булевых функций (СБФ) большого числа переменных были разработаны и с 1971 г. стали выпускаться промышленностью программируемые логические матрицы (ПЛМ – Programmable Logic Arrays – PLAs). Именно PLA можно считать первыми программируемыми логическими устройствами (Programmable Logic Devices – PLD). PLA получили очень широкое распространение в качестве универсальной элементной базы.

Совершенствование архитектуры PLA привело к появлению программируемых матриц логики (ПМЛ – Programmable Array Logics – PALs), которые на долгие годы определили наиболее популярную архитектуру PLD. С момента своего появления PAL стали успешно конкурировать с PLA и благодаря ряду присущих им положительных свойств практически полностью вытеснили программируемые пользователем PLA.

Дальнейшее совершенствование технологии производства интегральных схем, повышение степени интеграции, успехи в создании корпусов с большим числом внешних выводов в начале 90-х гг. привели к возможности реализации на одном кристалле нескольких PAL, объединяемых программируемыми соединениями. Подобные архитектуры получили название сложных PLD (Complex PLD – CPLD), соответственно все разработанные ранее PLD стали называть стандартными PLD (Standart PLD – SPLD) или классическими PLD (Classic PLD).

Основу всех рассмотренных выше устройств составляют программируемые матрицы. Поэтому эти устройства еще называют программируемыми логическими устройствами, имеющими матричную структуру.

Параллельно с PLD также развивались архитектуры вентиляльных матриц (Gate Array – GA) или матриц логических ячеек (Logic Cell Array – LCA), в русскоязычной литературе получившие название базовых матричных кристаллов (БМК).

В настоящее время ПЛИС принято делить на три больших класса: стандартные PLD (Standart PLD – SPLD) или классические PLD (Classic PLD), сложные PLD (Complex PLD – CPLD) и программируемые пользователем вентиляльные матрицы (Field Programmable Gate Array – FPGA) (рис. 1.1).

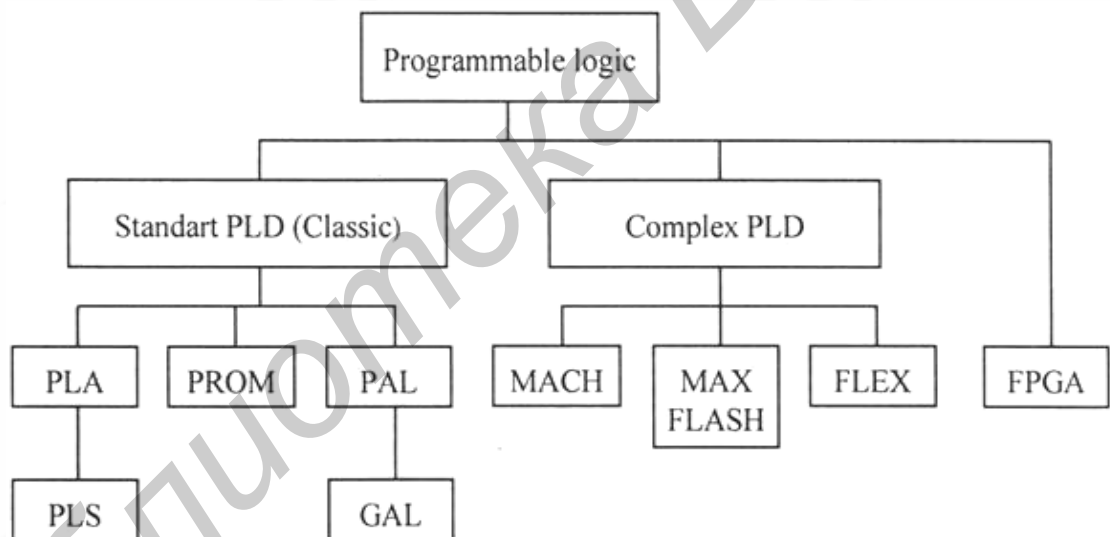


Рис. 1.1

1.2. Стандартные ПЛИС

Структуру большинства стандартных PLD условно можно представить в виде совокупности двух матриц взаимно ортогональных проводников: матрицы И и матрицы ИЛИ. Входные сигналы обычно поступают на парафазные входы матрицы И, которая на ортогональных шинах позволяет реализовать любые конъюнкции входных переменных. Выходы матрицы И соединены со входами

матрицы ИЛИ, которая на выходах реализует дизъюнкции поступающих сигналов. Совокупность выходных шин матрицы И и входных шин матрицы ИЛИ образует множество промежуточных шин PLD (product terms). В зависимости от того, какая матрица программируется: матрица И или матрица ИЛИ, SPLD принято делить на три класса: программируемые логические матрицы (ПЛМ – Programmable Logic Arrays – PLAs), программируемые постоянные запоминающие устройства (ППЗУ – Programmable Read Only Memory – PROM) и программируемые матрицы логики (ПМЛ – Programmable Array Logics – PALs).

В PLA (рис. 1.2) программируются обе матрицы: матрица И и матрица ИЛИ, что делает их наиболее гибкими из всех устройств типа PLD. Основным недостатком является снижение быстродействия и повышение цены. В PROM (рис. 1.3) матрица И постоянно настроена на функции полного дешифратора. Использование PROM наиболее эффективно для реализации устройств с небольшим числом входов и большим числом логических произведений. В структуре PAL (рис. 1.4), наоборот, матрица ИЛИ имеет фиксированную настройку, а программируется только матрица И.

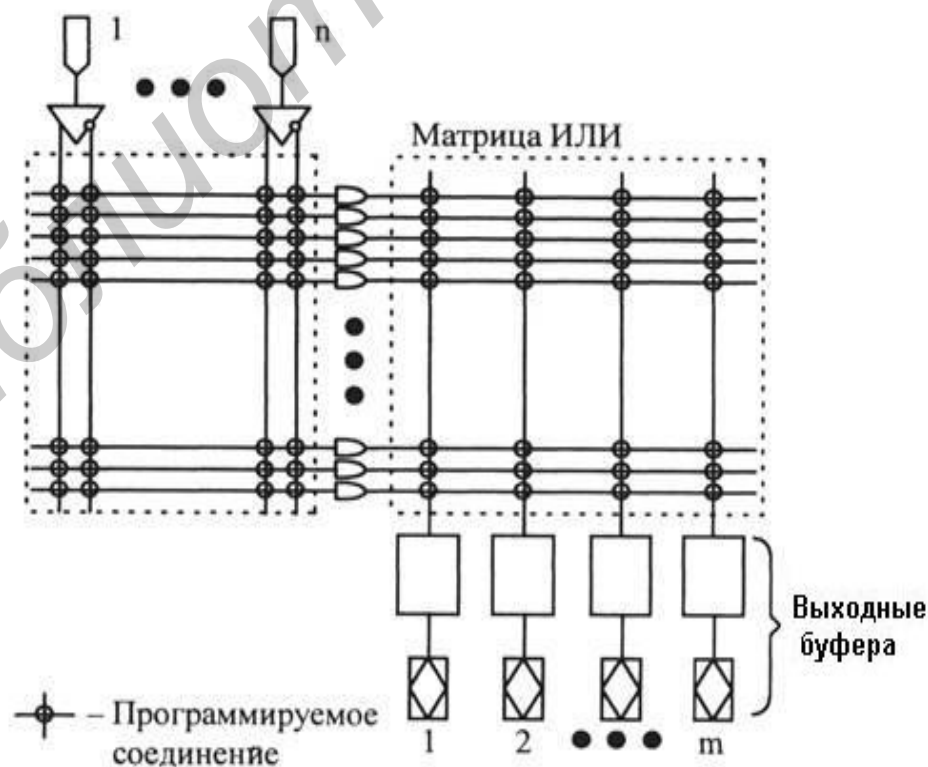


Рис. 1.2

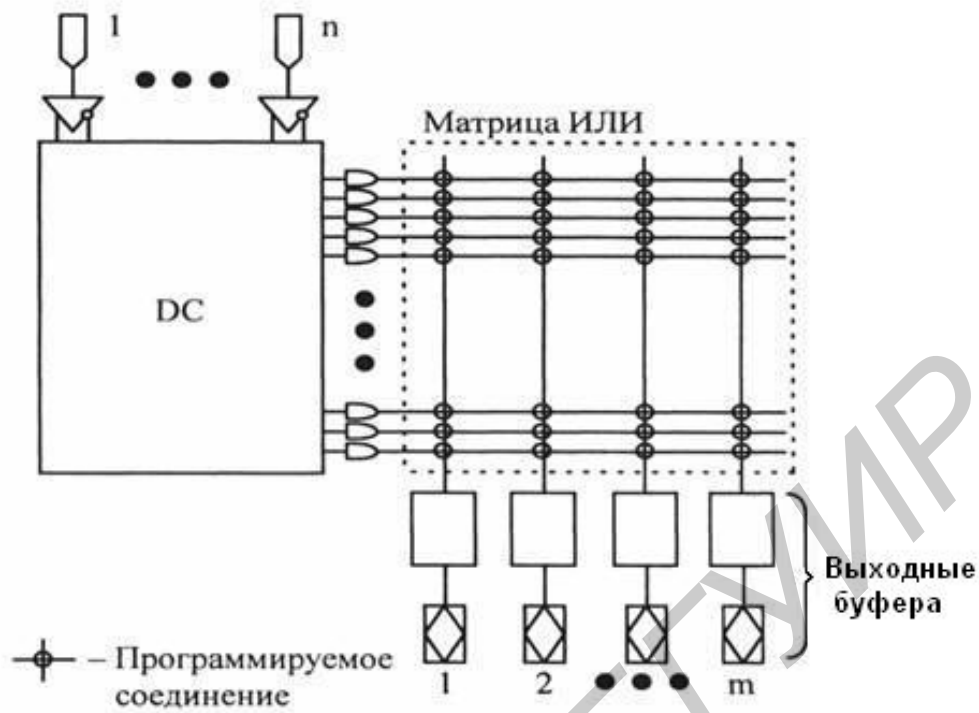


Рис. 1.3

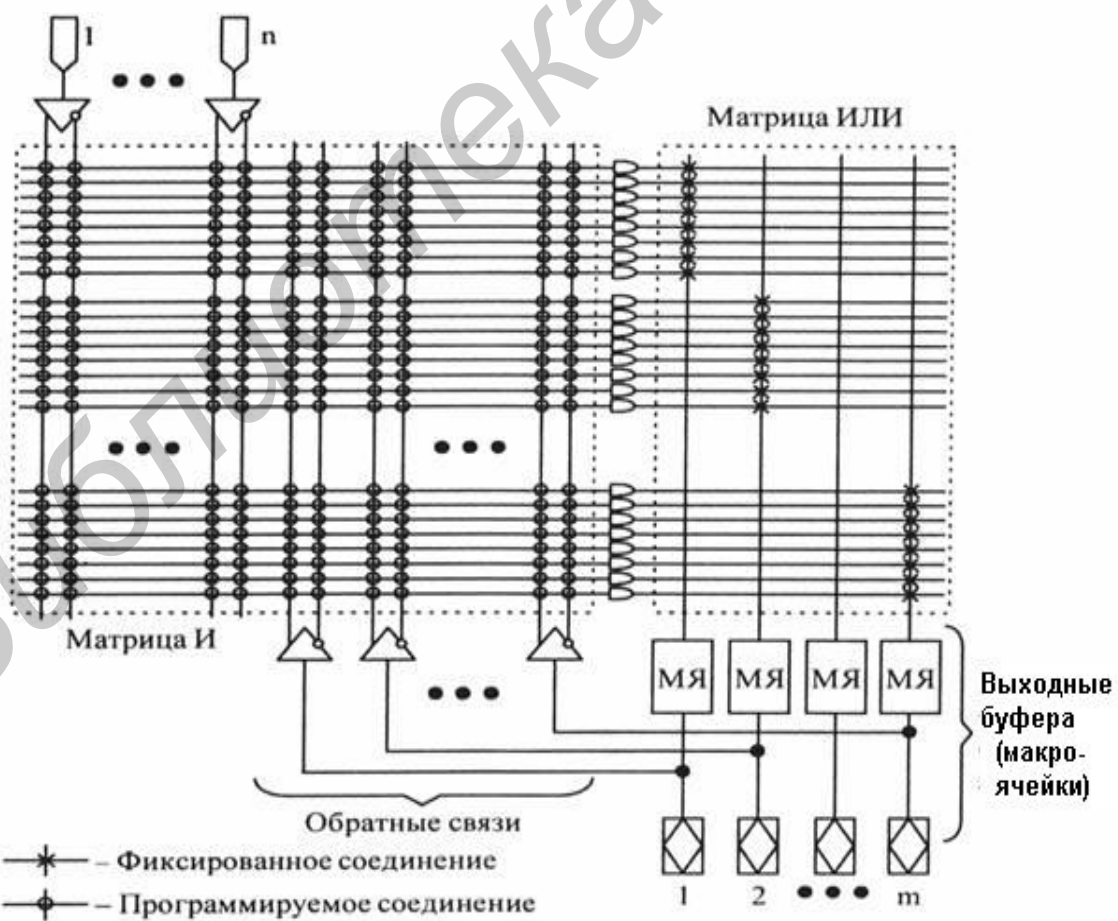


Рис. 1.4

Устройства PAL сочетают гибкость, свойственную PLA, с быстродействием, свойственным PROM. Они содержат n парафазных входов, матрицу И, матрицу ИЛИ, m выходных буферов и цепи обратной связи. Матрица И является программируемой и на своих выходах позволяет получать любые элементарные конъюнкции переменных, значения которых поступают на ее входы. Выходы матрицы И подсоединены ко входам матрицы ИЛИ, которая образует дизъюнкции элементарных конъюнкций, сформированных матрицей И. Выходы матрицы И называются промежуточными шинами (product terms), или термами (terms).

Программирование только одной матрицы И значительно упрощает структуру PAL и, как следствие, приводит к снижению стоимости устройства и повышению его быстродействия. Кроме того, упрощение матрицы ИЛИ позволило добавить в структуру PAL цепи обратной связи и выходные буферы, благодаря чему PAL приобрели новые качества.

Выходные буферы PAL представляют собой программируемые макроячейки, которые и определяют архитектуру PAL. Макроячейки PAL могут включать выходной инвертор с тремя состояниями, триггеры различного типа, вентили «исключающее ИЛИ» и др. В соответствии с типом выходных макроячеек PAL делят на функциональные группы: комбинационные, регистровые, универсальные, асинхронные и др. В свою очередь производители делят PAL на семейства, в которых выделяют серии устройств, причем в одной и той же серии могут встречаться устройства из разных функциональных групп. Более того, различные типы выходных макроячеек могут одновременно встречаться в одной PAL.

1.3. Макроматрицы (MACH-устройства)

Дальнейшее увеличение степени интеграции PLD и успехи в создании корпусов микросхем с большим числом выводов привели к появлению устройств, которые получили название КМОП – макроматрицы высокой плотности (Macro Array CMOS High-density), или MACH-устройства. MACH-

устройства можно рассматривать как несколько PAL (называемых PAL-блоками) на одном кристалле, которые могут соединяться между собой при помощи матрицы переключений. Каждый PAL-блок имеет свое множество двунаправленных внешних выводов. В MACH-устройствах также имеется некоторое число «чистых» входов, значения сигналов с которых через матрицу переключений могут быть поданы на вход любого PAL-блока. Обобщенная структура MACH-устройств показана на рис. 1.5.

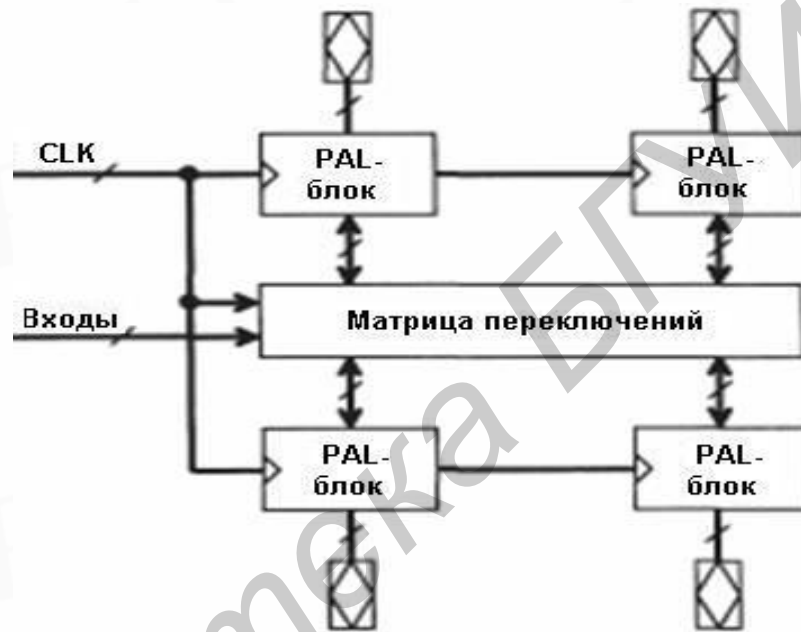


Рис. 1.5

Характерным для MACH-устройств является то, что любой сигнал со входа на выход может пройти через матрицу переключений. Благодаря этому сохраняется постоянная задержка прохождения сигнала с любого входа к любому выходу. В настоящее время фирмой Advanced Micro Devices (AMD) разработано три поколения MACH-устройств. Первое составляют семейства MACH1 и MACH2, второе – MACH3 и MACH4, а третье – MACH5.

1.4. Матричные таблицы (MAX-устройства)

Фирмой Altera разработаны сложные PLD, которые получили название «многократные матричные таблицы» (Multiple Array Matrix – MAX). Обобщенная структура простых MAX-устройств показана на рис. 1.6. Она представляет

собой совокупность PAL-подобных модулей, называемых блоками логических матриц (Logic Array Blocks – LAB), или LAB-модулями. Совокупность LAB-модулей объединяется в одно устройство с помощью программируемой матрицы межсоединений (Programmable Interconnect Array – PIA).

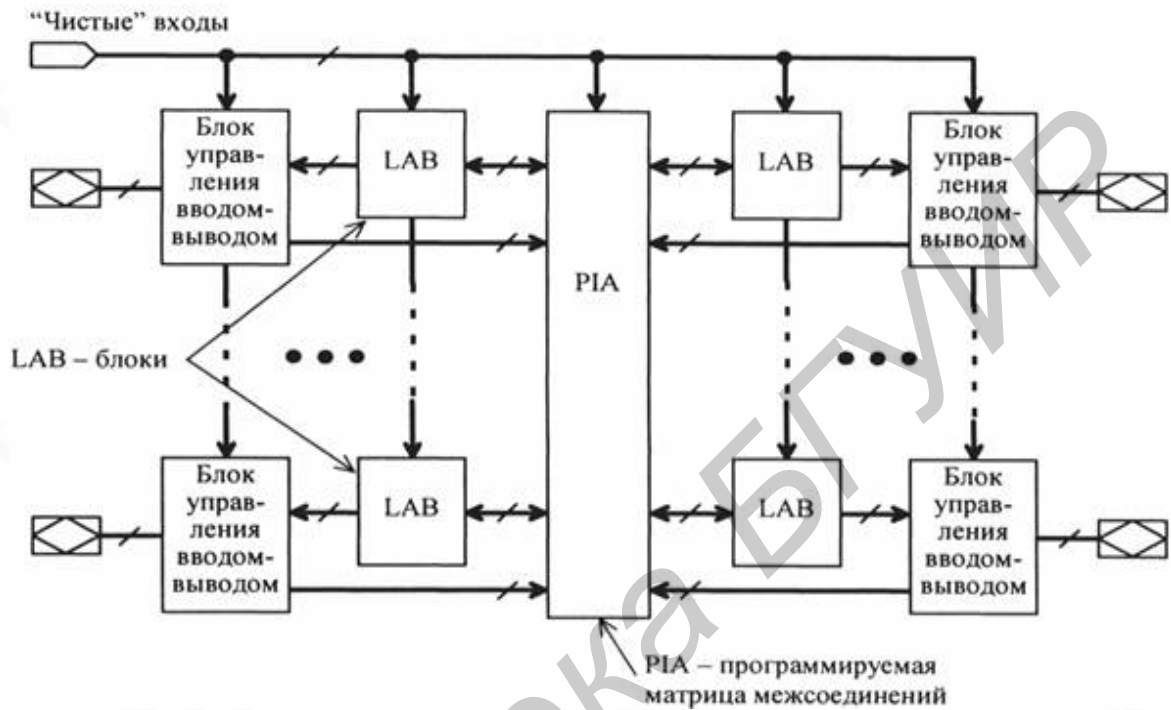


Рис. 1.6

В настоящее время разработаны три поколения MAX-устройств: MAX5000, MAX7000, MAX9000. В семействе MAX9000 LAB-модули организованы в виде матрицы, а соединения между ними осуществляются с помощью вертикальных и горизонтальных каналов межсоединений.

Каждый LAB-модуль семейств MAX5000 и MAX7000 имеет свое множество двунаправленных внешних выводов. В семействе MAX9000 LAB-модули связаны с двунаправленными внешними выводами при помощи матрицы межсоединений и элементов ввода-вывода. Кроме того, специальные управляющие сигналы, значения которых поступают с отдельных «чистых» входов, проходят через все LAB-модули, что характерно для всех семейств MAX-устройств. Эти сигналы обычно используются для синхронизации и сброса регистров, а также для разрешения выходов.

Важным отличием всех МАХ-устройств от МАСН-устройств является то, что в МАХ-устройствах не гарантируется постоянная задержка прохождения сигнала с любого входа на любой выход. В МАХ-устройствах задержка сигнала зависит от конфигурации макроячеек, способа получения входных сигналов, а также от пути, по которому входной сигнал достигает выхода. Поскольку для разных сигналов пути прохождения через устройство различаются, то различными будут и задержки сигналов. Поэтому разработка проектов на основе МАХ-устройств неизбежно связана с необходимостью выполнения временного тестирования и анализа. С другой стороны, гибкость в выборе пути прохождения сигналов в ряде случаев позволяет повысить быстродействие проекта по сравнению с его реализацией на МАСН-устройствах.

1.5. Программируемые пользователем вентиляльные матрицы (FPGA)

Основу FPGA составляет матрица логических элементов, между которыми располагается поле межсоединений. По краям кристалла находятся блоки ввода-вывода (рис. 1.7).

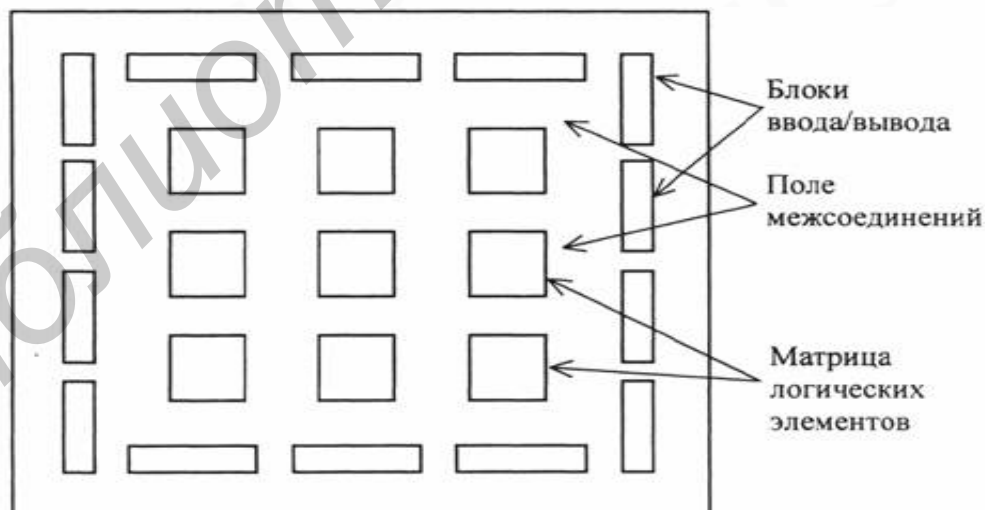


Рис. 1.7

Логические элементы FPGA (ЛЭ)

В качестве ЛЭ используются:

1. Транзисторные пары, простые логические вентили И-НЕ, ИЛИ-НЕ и т.п. Такие ЛЭ называют SLC – Simple Logic Cells.

2. Логические модули на основе мультиплексоров.

3. Логические модули на основе программируемых ПЗУ, такие ЛЭ называют LUTs — Look-Up Tables.

Важной характеристикой ЛЭ является их «зернистость» (Granularity). Другой важной характеристикой считается «функциональность» (Functionality).

Первое свойство связано с тем, насколько «мелкими» будут те части, из которых можно «собирать» нужные схемы, второе – с тем, насколько велики логические возможности ЛЭ.

Блоки ввода-вывода

Блоки ввода-вывода (БВВ) необходимы для согласования внутренних и внешних уровней логических сигналов, усиления сигналов до необходимой нагрузочной способности, защиты внутренних цепей FPGA от электрических повреждений и др. В FPGA блоки ввода-вывода могут программироваться для согласования либо с ТТЛ, либо с КМОП уровнями сигналов.

Системы межсоединений FPGA

Системы межсоединений (системы коммутации), как и логические элементы, реализуются в широком диапазоне архитектурных и технологических решений. Линии связей в FPGA обычно сегментированы, т.е. составлены из проводящих сегментов (участков, не содержащих ключей) различной длины, соединяемых друг с другом программируемым элементом связи (ключом). Малое количество сегментов ведет к недостаточно эффективному использованию логических блоков, слишком большое – к появлению большого числа программируемых ключей в линиях связи, что увеличивает затраты площади кристалла и вносит дополнительные задержки сигналов.

Короткие сегменты затрудняют реализацию длинных связей, длинные – коротких. Поэтому целесообразна иерархическая система связей с несколькими типами межсоединений для передач на разные расстояния. Целью построения системы связей является обеспечение максимальной коммутируемости блоков при минимальном количестве ключей и задержек сигналов, а также предсказуемость последних, облегчающая проектирование.

1.6. Сложные PLD (Complex PLD – CPLD)

Сложные программируемые логические ИС (СПЛИС) архитектурно произошли от структур PLD (PAL, GAL).

Архитектурно CPLD состоит из центральной коммутационной матрицы, множества функциональных блоков (ФБ) (именуемых также макроэлементами, макроэлементами и др.) и блоков ввода-вывода на периферии кристалла. Архитектура CPLD показана на рис. 1.8, где через ПМС обозначена программируемая матрица соединений (PIA, Programmable Interconnect Array).

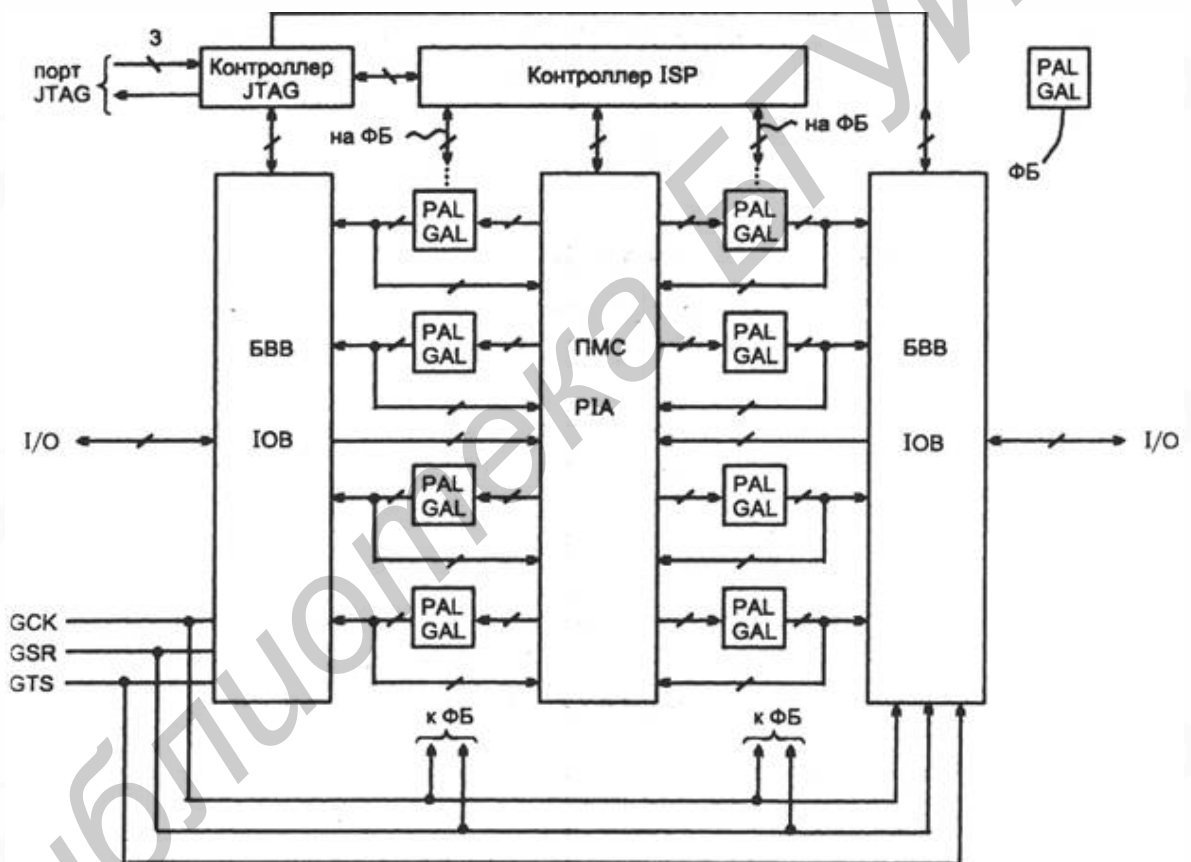


Рис. 1.8

Функциональные блоки CPLD

Эти блоки подобны PLD и содержат многовходовую (wide) программируемую матрицу элементов И, вырабатывающую конъюнктивные термы из поступающих на ее входы переменных, группу элементов ИЛИ, между которыми распределяются выработанные термы, и некоторые другие элементы, обогащающие функциональные возможности ФБ. Функциональные блоки реализуют

двухуровневую логику с вариантами формируемого результата (прямой или инверсный, комбинационный выход или регистровый выход и т.д.).

Системы коммутации CPLD

В CPLD используется непрерывная или одномерно непрерывная система связей, причем все связи идентичны, что дает хорошую предсказуемость задержек сигналов в связях – важное достоинство для схем высокого быстродействия. Типичная ПМС (рис. 1.9) позволяет соединять выход любого ФБ со входами других. Входы ФБ связаны с горизонтальными линиями, пересекающимися все вертикальные линии. Любой вход может быть подключен к любому выходу программированием точек связи между вертикальными и горизонтальными линиями. Таким образом обеспечивается полная коммутируемость блоков.

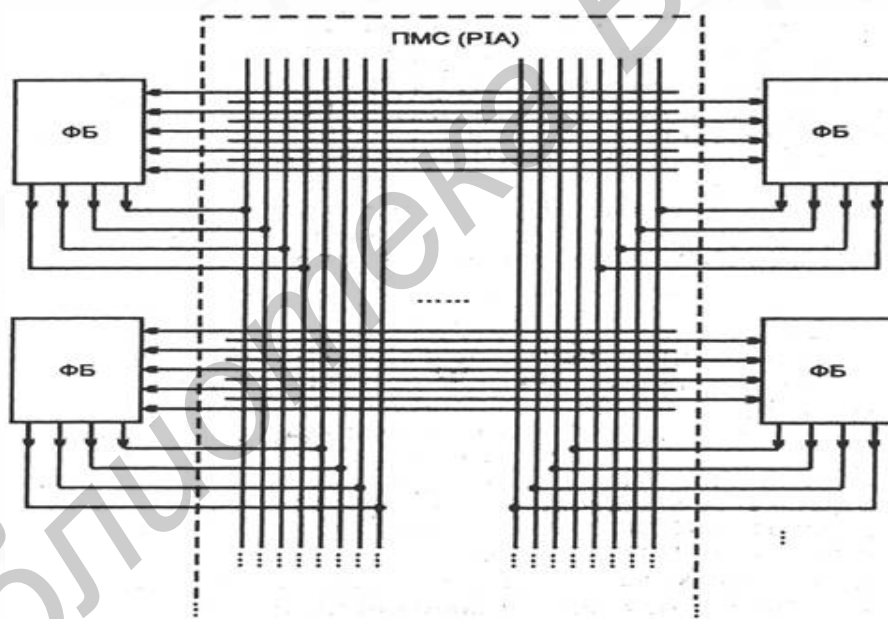


Рис. 1.9

В рамках каждого семейства потребителю предлагается ряд представителей, различных по сложности, стоимости и другим параметрам, с целью гибкого обслуживания разнообразных потребностей. Общее число разновидностей CPLD оказывается очень большим.

CPLD и СБИС ПЛ смешанной архитектуры производятся многими фирмами. К ведущим фирмам относятся Altera (семейства MAX, FLEX, APEX и

др.), Atmel (семейство ATF1500 и др.), Vantis (ранее была известна как фирма AMD, семейство MACH), Xilinx (семейство XC9500), Philips и т.д.

1.7. СБИС программируемой логики смешанной архитектуры (FLEX)

По архитектуре микросхемы типа FLEX (рис. 1.10) занимают промежуточное положение между классическими вариантами CPLD и FPGA. Сохранив ряд качеств предшествующих разработок CPLD, микросхемы FLEX в то же время имеют логические элементы табличного типа (LUT), их логические блоки расположены в виде матрицы, причем трассировочные каналы проходят горизонтально и вертикально между ЛБ, что характерно для FPGA. Одновременно с этим трассы в каналах не сегментированы, а непрерывны, что типично для CPLD и дает хорошую предсказуемость и малые величины задержек.

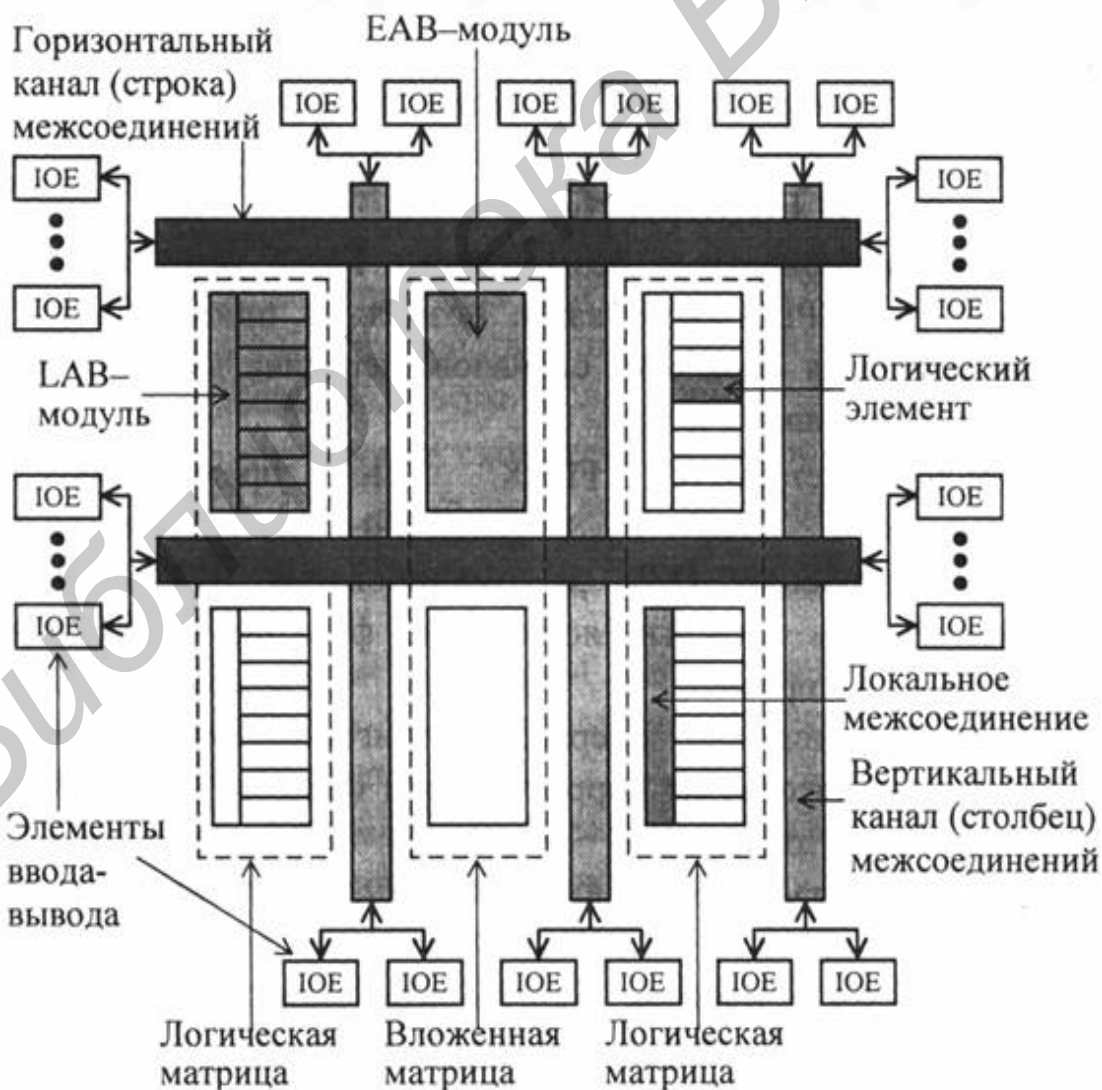


Рис. 1.10

Структура FLEX-устройств представляет матрицу блоков логических элементов (Logic Array Block – LAB) или LAB-модулей, между которыми располагаются строки (row) и столбцы (column) каналов межсоединений (Fast Track Interconnect – FTI). Устройства семейства FLEX10K дополнительно содержат блоки вложенной матрицы (Embedded Array Blocks – EAB).

FTI-каналы являются непрерывными и проходят через все устройство: горизонтальные – по ширине, вертикальные – по высоте устройства. Это позволяет, в отличие от FPGA, значительно сократить число точек коммутации сигнала и легко предсказывать задержки формируемых сигналов. Окончания вертикальных и горизонтальных линий каналов матрицы межсоединений подключаются к элементам ввода-вывода (I/O Elements – IOE).

Каждый LAB-модуль состоит из восьми логических элементов (Logic Elements – LEs), объединенных локальными межсоединениями. Логические функции в логических элементах реализуются с помощью функционального генератора (Look-Up Tables – LUTs) и программируемых регистров.

Высокой производительности FLEX-устройств также способствуют глобальные цепи с большим коэффициентом расширения, используемые для передачи сигналов глобального управления и синхронизации всего FLEX-устройства. EAB-модули устройств FLEX10K представляют собой быстродействующее ОЗУ с изменяемой конфигурацией. Кроме возможностей построения ОЗУ и эмулирования ПЗУ, EAB-модули позволяют реализовать сложные логические функции с очень высоким быстродействием.

Настройка FLEX-устройств производится при включении питания системы от отдельного ППЗУ, ОЗУ, микропроцессора или через специальные устройства последовательного и параллельного интерфейса, называемые фирмой Altera кабелями BitBlaster и ByteBlaster соответственно.

На сегодняшний день микросхемы типа FLEX являются оптимальными PLD для проектирования и реализации цифровых устройств средней и повышенной сложности и получили наибольшее распространение среди CPLD.

2. Основные сведения о САПР МАХ+plus II

2.1. Проектирование с помощью САПР МАХ+plus II

САПР МАХ+plus II (Multiple Array Matrix Programmable Logic User) представляет собой архитектурно-независимую среду проектирования, которая легко приспособляется к конкретным проектным требованиям и может работать на различных компьютерных платформах. МАХ+plus II предоставляет различные способы ввода проекта, быструю компиляцию и непосредственное программирование микросхем.

Система проектирования МАХ+plus II, как показано на рис. 2.1, является функционально полной САПР для реализации проектов на микросхемах программируемой логики фирмы Altera семейств Classic, МАХ3000, МАХ5000, МАХ7000, МАХ9000, FLEX6000, FLEX8000, FLEX10K, АСЕХ1К.

МАХ+plus II предоставляет полный спектр возможностей для проектирования цифрового устройства: различные способы ввода проекта, логический синтез, компиляцию с заданными временными ограничениями, функциональное и временное моделирование, разделение проекта на части и моделирование проекта, выполненного на нескольких микросхемах, временной анализ, автоматическое определение ошибок, программирование и верификацию микросхем. САПР МАХ+plus II позволяет вводить и сохранять проекты в виде файлов АНDL (язык описания аппаратуры фирмы Altera), EDIF, Verilog HDL, VHDL, OrCAD. МАХ+plus II может использовать netlist файлы фирмы Xilinx для ввода проекта, может создавать выходные SDF (Standard Delay Format) файлы для обеспечения удобного интерфейса с другими САПР промышленного стандарта САЕ.

МАХ+plus II имеет удобный графический интерфейс и простую в использовании справочную систему, содержащую всю необходимую разработчику информацию. МАХ+plus II состоит из 11 полностью интегрированных программных модулей, которые используются при создании проекта (рис. 2.2). Задание многих параметров и выполнение команд, например: открытие файлов, выбор микросхем, назначение контактов и логических ячеек, компиляция теку-

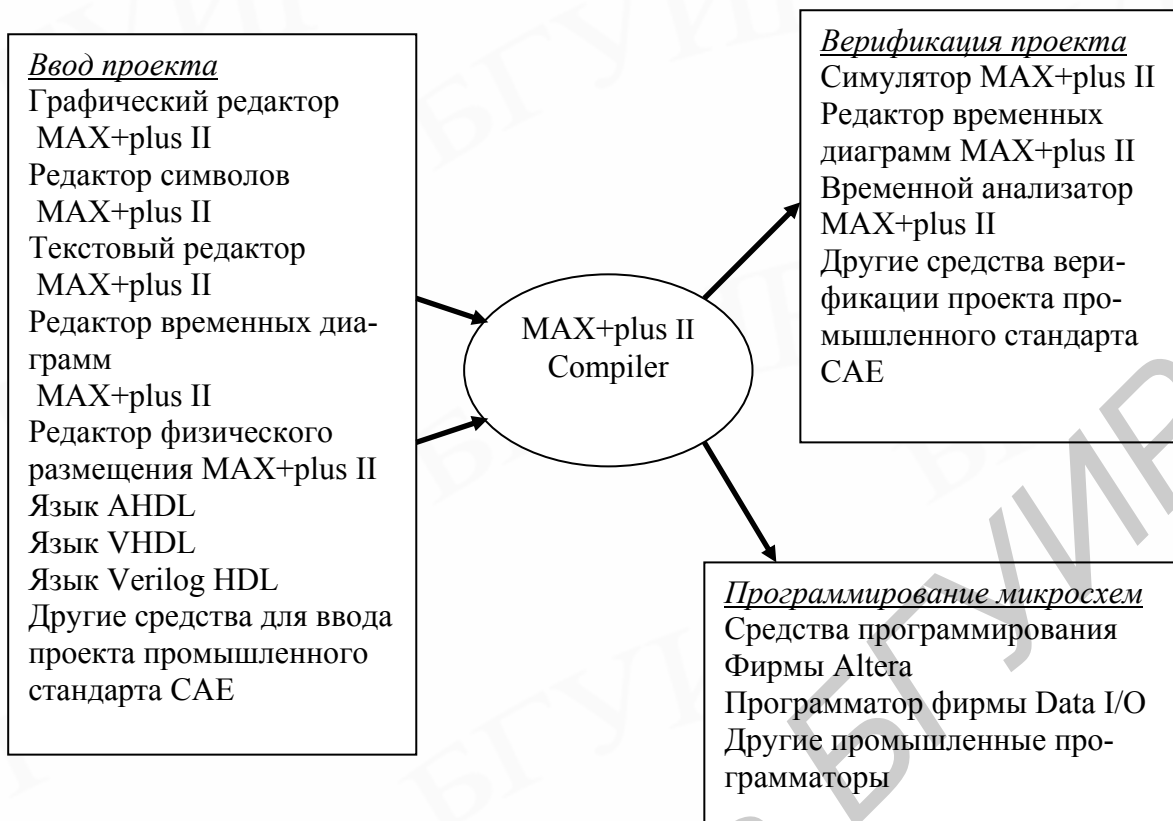


Рис. 2.1

щего проекта происходит во многих или во всех программных модулях САПР MAX+plus II одинаково. Редакторы проекта (графический, текстовый, временных диаграмм) и вспомогательные редакторы (редактор физического размещения и редактор символов) также имеют много общих параметров и команд. Каждый редактор проекта позволяет выполнять поиск сигнала, или символа, причем во всех редакторах это делается одинаково. В одном иерархическом проекте можно сочетать различные типы файлов ввода проекта, выбирая такой способ ввода, который больше всего подходит для каждого функционального блока. Обширная библиотека разработанных компанией Altera мега- и макрофункций, включая операционные устройства из библиотеки параметризованных модулей (Library of Parameterized Modules), обеспечивает широкий спектр возможностей для ввода проекта. САПР MAX+plus II позволяет работать с несколькими программными модулями одновременно.

Компилятор является основным модулем системы MAX+plus II и обеспечивает эффективную обработку проекта. Для достижения наилучшей реализа-

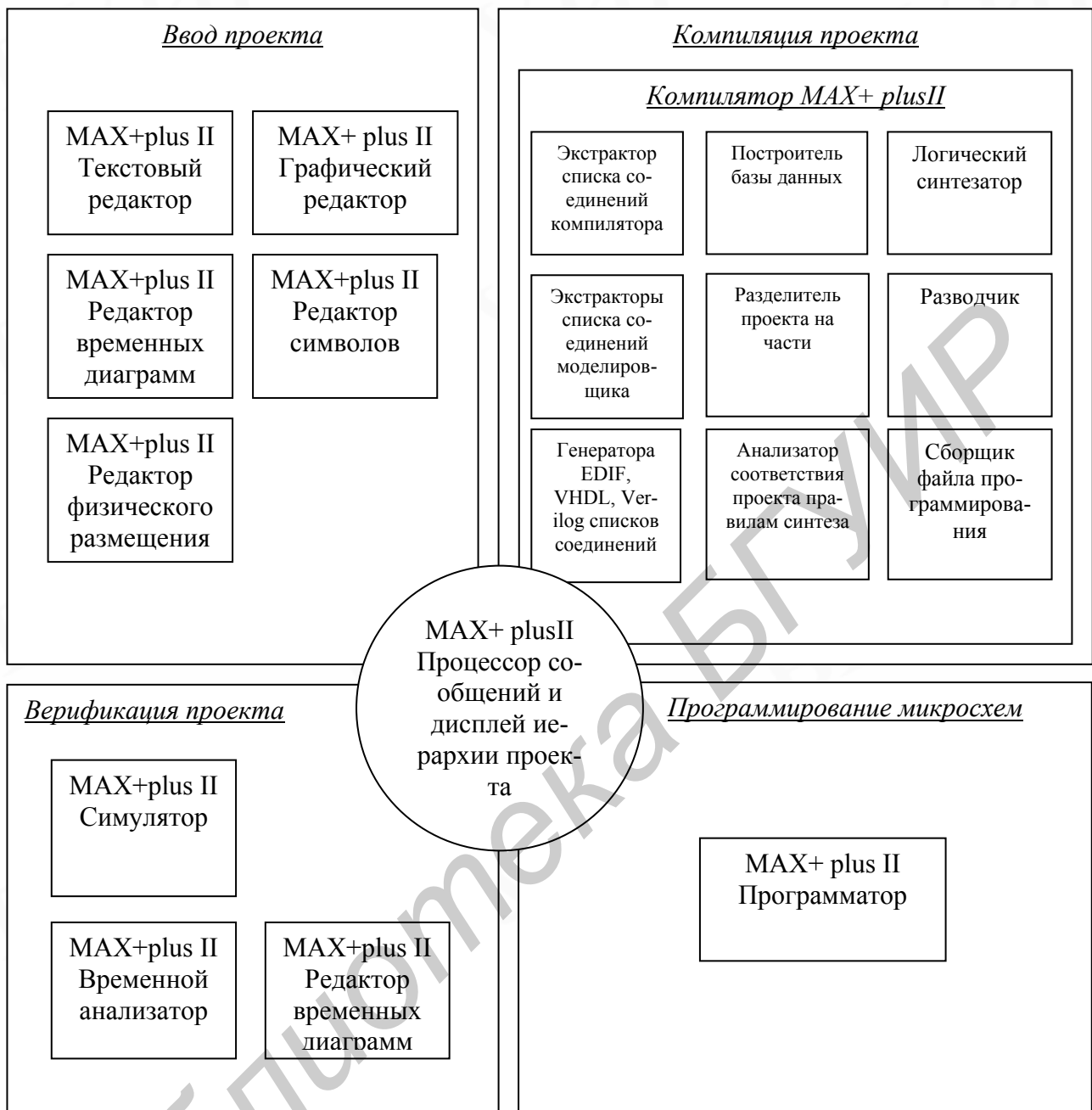


Рис. 2.2

ции проекта компилятору задаются параметры в выбранной микросхеме. Автоматическое определение местоположения ошибок и обширная документация по ошибкам и предупреждающим сообщениям делают формальную отладку проектов легкой и быстрой.

Также возможно создание различных выходных файлов для функционального и временного моделирования, моделирования проекта, реализованного

на нескольких микросхемах, временного анализа, программирования микросхемы.

Интеграция программных модулей MAX+ plus II позволяет максимально повысить эффективность и производительность, предоставляя полный контроль над средой проектирования.


2.2. Составные части (программные модули) MAX+plus II





Программное обеспечение MAX+ plus II состоит из 11 программных модулей и MAX+ plus II Manager. Одновременно могут быть активизированы несколько приложений ввода проекта, что позволяет переключаться между ними посредством щелчка мыши или выбора команды из меню. Также возможно запустить один из программных модулей в фоновом режиме. Команды, совместно используемые различными программными модулями, работают одинаково, что облегчает задачу проектирования на логическом уровне.






В табл. 2.1 приведено описание программных модулей MAX+ plus II и соответствующие им «иконки».

Таблица 2.1

Программные модули MAX+ plus II

«Иконки»	Программные модули
1	2
	<p>Hierarchy Display (дисплей иерархии проекта)—вывод на экран текущей иерархии файлов в виде иерархического дерева, ветвями которого являются подпроекты. С его помощью можно определить, является ли данный файл проекта графическим или текстовым или это проект, введенный с помощью редактора временных диаграмм, какие файлы открыты в данный момент, какие вспомогательные файлы, редактируемые пользователем, доступны для проекта. Также возможно открыть или закрыть один или несколько файлов прямо в иерархическом дереве и ввести для них назначения ресурсов</p>

1	2
	<p>Graphic & Symbol Editors (графический и символьный редакторы). Graphic Editor – позволяет выполнить схемный ввод проекта. Наряду с предоставляемыми фирмой Altera базовыми элементами (primitives), мегафункциями и макрофункциями, используемыми для построения стандартных блоков, также применяются созданные пользователем символы функциональных узлов. Symbol Editor – позволяет редактировать существующие и создавать новые символы</p>
	<p>Text Editor (текстовый редактор) – текстовый редактор позволяет создавать и редактировать текстовые файлы проекта на уровне функционально-логического описания на языке AHDL и VHDL. С помощью текстового редактора создаются, просматриваются и редактируются другие ASCII файлы, используемые различными модулями MAX+ plus II. Создание AHDL и VHDL файлов посредством текстового редактора MAX+ plus II по сравнению с другими текстовыми редакторами имеет ряд преимуществ: контекстуально зависящая справочная система, синтаксическая раскраска, шаблоны AHDL и VHDL</p>
	<p>Waveform Editor (редактор временных диаграмм) – выступает в двух ролях: как инструмент ввода проекта и как инструмент для введения тестовых векторов и просмотра результатов моделирования</p>
	<p>Floorplan Editor (редактор физического размещения) – позволяет назначать размещение физических контактов микросхем и ресурсов логических ячеек, используя графическую среду, а также редактировать как расположение контактов, так и назначать расположение отдельных логических ячеек (LC) в логических блоках (LAB). Также доступен просмотр результатов размещения, выполненных при последней компиляции</p>

1	2
	<p>Compiler (компилятор) – производит логическую обработку проектов и размещение их на микросхемах семейств Altera Classic, MAX5000, MAX3000, MAX7000, MAX9000, FLEX8000, FLEX10K, ACEX1K. Большинство задач он выполняет автоматически. Тем не менее все или часть параметров, управляющих процессом компиляции, могут задаваться пользователем</p>
	<p>Simulator (симулятор) – позволяет тестировать логику работы и временные характеристики схемы. Возможно функциональное моделирование, моделирование с учетом временных соотношений и моделирование проекта, выполненного на нескольких микросхемах</p>
	<p>Timing Analyzer (временной анализатор) – анализирует временные характеристики цифрового устройства после того, как оно синтезировано и оптимизировано компилятором</p>
	<p>Programmer (программатор) – позволяет программировать микросхемы Altera и тестировать созданные устройства</p>
	<p>Message Processor (процессор сообщений) – выводит на экран сообщения об ошибках, предупреждающие и информирующие сообщения о состоянии проекта и позволяет автоматически определять местоположение источника сообщения в исходных файлах проекта, во вспомогательных файлах и в назначениях физического уровня размещения</p>

2.3. Файлы проекта, вспомогательные файлы и проекты

До начала работы с MAX+ plus II следует обозначить разницу между файлами проекта, вспомогательными файлами и проектами.

Файлы проектов

Файл проекта – это файл, содержащий функционально-логическое описание проекта для MAX+ plus II и служащий исходным описанием для компиля-

тора. Файл проекта может быть графическим, текстовым или файлом в виде временной диаграммы, созданным соответствующим редактором MAX+ plus II или каким-либо другим стандартным графическим или текстовым редактором или генератором EDIF VHDL файлов.

Компилятор может автоматически обрабатывать следующие файлы проекта:

- графические файлы проекта (.gdf);
- текстовые файлы проекта (.tdf);
- файлы проекта временных диаграмм (.wdf);
- VHDL файлы проекта (.vhd);
- Verilog файлы проекта (.v);
- графические файлы OrCAD (.sch);
- входные файлы EDIF (.edf);
- файлы формата списка соединений Xilinx (.xnf);
- файлы проекта Altera (.adf);
- файлы конечного автомата (.smf).

Вспомогательные файлы

Вспомогательные файлы – это файлы, связанные с проектом MAX+ plus II, но не являющиеся частью иерархического дерева проекта. Большинство вспомогательных файлов не содержат описаний логики проекта. Некоторые из этих файлов автоматически создаются приложением MAX+ plus II, некоторые вводятся пользователем. Примерами вспомогательных файлов являются файлы назначения и конфигурации (.act), символьные файлы (.sym), файлы сообщений (.rpt) и векторные файлы (.vec).

Проекты

Проект состоит из всех файлов и иерархии проекта, включая вспомогательные входные и выходные файлы. Проект называется так же, как и файл проекта на самом высоком уровне иерархического дерева, но без расширения имени файла. MAX+ plus II осуществляет компиляцию, моделирование, временной анализ и программирование в данный момент времени только одного

проекта, существует возможность редактировать файлы, относящиеся к другому проекту.

2.4. Справочная система MAX+plus II

Справочная система MAX+ plus II обеспечивает полную, обновленную документацию программного обеспечения MAX+ plus II и содержит основные сведения об инструментах программных модулей MAX+plus II, командах, процедурах, сокращениях, золотых правилах, сообщениях, базовых элементах, мегафункциях, макрофункциях, AHDL, VHDL. Справочная система также предоставляет информацию обо всех микросхемах фирмы Altera и адаптерах, что позволяет пользователю выбрать подходящую микросхему до начала функционально-логического проектирования.

Каждый раздел справочной системы содержит одно или несколько выделенных слов, называемых переходами, которые позволяют перейти к другим темам раздела или к дополнительной информации по текущей теме.

2.5. Меню Help




Строка меню каждого программного модуля MAX+ plus II обеспечивает доступ к меню Help (меню справочной системы).







В табл. 2.2 приведено описание всех пунктов меню справочной системы (Help) и, показаны соответствующие им «иконки».




Таблица 2.2

Пункты меню справочной системы MAX+plus II

Пункт меню	«Иконки»
1	2
Search for Help on (поиск по справочной системе) – открывает диалоговое окно, названное Search (поиск), которое позволяет быстро отыскать нужную информацию по обширному индексу справочной системы	–

1	2
<p>MAX+ plus II Table of Contents (таблица содержания MAX+ plus II) – полная таблица содержания, в которой перечислены все главные темы, представленные в справочной системе MAX+ plus II. К ней также можно получить доступ через кнопку Contents на панели кнопок в верхней части окна справочной системы</p>	–
<p><Application name>Help (справочная система по <Имя программного модуля>) – открывает подменю по темам справочной системы для текущего программного модуля MAX+ plus II</p>	<p>В табл. 2.1 показаны «иконки» всех приложений</p>
<p>Table of Contents – таблица содержания для текущего программного модуля</p>	–
<p>Introduction (введение) – краткий обзор текущего программного модуля, включая иллюстрации и информацию о том, как начать работу с использованием этого модуля</p>	
<p>Basic Tools (основные инструменты) – подробное описание элементов, находящихся в окне программного модуля, а также входных и выходных файлов, сопровождаемое иллюстрациями и примерами. В категории основных инструментов в справочной системе также представлена информация о базовых элементах и макрофункциях, о кнопках, полях, «иконках» и т.д.</p>	
<p>Commands (команды) – описание каждой из команд текущего программного модуля, сопровождаемое иллюстрациями и примерами. Иллюстрации диалоговых окон команд включают появляющиеся объяснения для каждой опции и кнопки диалогового окна</p>	

1	2
<p>Procedures (процедуры) – пошаговые инструкции, сопровождаемые иллюстрациями и примерами решения специфических задач в текущем программном модуле</p>	
<p>Golden Rules (золотые правила) – подборка важнейших советов и правил по использованию текущего программного модуля</p>	
<p>Shortcuts (горячие клавиши и «иконки») – последовательность нажатия клавиш клавиатуры или мыши, «иконок» панели инструментов для быстрого выполнения команд и процедур текущего программного модуля</p>	
<p>AHDL – справочная система по AHDL, включающая подробные инструкции по разработке проекта, описанию основных элементов структуры проекта, синтаксиса AHDL и руководство по стилю проектирования</p>	
<p>VHDL – справочная система по VHDL, включающая инструкции по разработке проекта с помощью MAX+ plus II VHDL, описания поддерживаемых конструкций языка VHDL, синтаксиса VHDL и руководство по стилю проектирования</p>	
<p>Verilog HDL – справочная система по Verilog HDL, включающая инструкции по разработке проекта с помощью MAX+plus II, описания поддерживаемых конструкций языка Verilog HDL, синтаксиса и руководство по стилю проектирования</p>	
<p>Megafunction/LPM (мегафункция/БПМ) – список мегафункций из библиотеки параметризованных модулей и функции Altera MegaCore/OpenCore. Для конкретной мегафункции выводится на экран ее описание, правила обращения, AHDL Function Prototype (AHDL прототип функции), VHDL Component Declaration (средства включения в описания на языке VHDL) и информация о необходимых для этой мегафункции ресурсах микросхемы</p>	<p>–</p>

1	2
<p>Old-Style Macrofunctions (макрофункции микросхемы серии 74XXXX) – перечень операционных узлов – функциональных аналогов микросхем серии 74XXXX. Можно выбрать один из перечисленных узлов для вывода на экран всех названий макрофункций, которые его реализуют. При выборе конкретной макрофункции будут выведены на экран ее описание и логические уровни сигналов по умолчанию</p>	–
<p>Primitives (базовые элементы) – алфавитный список базовых элементов</p>	–
<p>Device & Adapters (микросхемы и адаптеры) – список всех текущих микросхем фирмы Altera, поддерживаемых MAX+ plus II, и адаптеров для их программирования. Также включены руководства по выбору микросхем для каждого семейства фирмы Altera. При выборе одной из микросхем, перечисленных в списке, на экран выводятся ее характеристики, информация о расположении логических ячеек и назначении контактов для каждого типа корпуса</p>	
<p>Messages (сообщения) – алфавитный список всех сообщений MAX+ plus II: информационных, сообщений-предупреждений и сообщений об ошибках. Все сообщения сопровождаются подробными объяснениями возможных причин их появления и рекомендуемыми действиями. В объяснениях сообщений предусмотрены переходы к дополнительной полезной информации</p>	
<p>Glossary (словарь) – полный список терминов MAX+ plus II и их определений</p>	
<p>READ.ME (прочти) – копия файла read.me, предоставляемого с MAX+ plus II. В нем находится информация о системных требованиях, известных проблемах и способах их решения</p>	–



1	2
New Features in This Release (новые возможности данной версии) – описание всех новых возможностей в текущей версии MAX+ plus II, включая поддержку новых микросхем и интерфейсы с другими средствами проектирования стандарта САЕ	
How to Use help (как пользоваться справочной системой) – информация о механизмах использования приложений справочной системы Windows	–
How to Use MAX+plus II Help (как пользоваться справочной системой MAX+plus II) – подробная информация о том, как пользоваться справочной системой MAX+plus II, включая описание пунктов меню справочной системы и документацию по условным обозначениям	
About MAX+plus II (о MAX+plus II) – отображает номер версии MAX+plus II, номер версии текущего приложения, информацию об авторских правах	–

Таблица 2.3

Кнопочная панель окна справочной системы

Название	Функция
1	2
Contents (содержание)	Показывает таблицу содержания справочной системы MAX+plus II
Index (индекс)	Открывает диалоговое окно Help Topics (темы справочной системы)
Back (вернуться)	Возвращает ранее просмотренную информацию, т.е. восстанавливает путь по темам, которые уже были просмотрены

1	2
Print (печать)	Выводит на печать одну или несколько копий текущей темы справочной системы
Glossary (словарь)	Показывает список терминов MAX+ plus II и их определения

3. Основные этапы разработки цифрового устройства

1. Составление содержательной граф-схемы алгоритма или функциональной блок-схемы устройства.

Первый этап включает переход от технического задания (ТЗ) к формализованному описанию проектируемого устройства, представляющему собой разбиение задачи на отдельные функционально обособленные подзадачи (этап декомпозиции). Способ и средства разбиения определяются непосредственно проектировщиком и в редких случаях являются predetermined.

2. Разработка общей структуры операционного блока (ОБ).

Основа этапа – выбор допустимых для данного уровня иерархии элементов, определение связей между ними и, если параметры элементов являются настраиваемыми, то и их настройка.

3. Описание работы управляющего автомата (УА).

На этом этапе определяется функционирование УА, обеспечивающее требуемое взаимодействие элементов ОБ. Следует подчеркнуть, что два последних этапа сильно взаимосвязаны, и, если они не разрабатываются параллельно, то обычно выполняются итерационно.

Формы и средства описания автомата разнообразны. Современная тенденция состоит в переходе от записи логических выражений, ограниченных правилами ТЗ, к графической форме. Описание в виде граф-схемы переходов (диаграммы состояний) становится одним из самых распространенных вариантов задания автоматов (в английской терминологии State Machines). Графические редакторы для создания автоматов включаются в состав средств задания

исходных проектов современных САПР (например, в САПР Foundation фирмы Xilinx разработки фирмы ALDEC).

4. Компиляция проекта.

После составления проекта и всех его частей переходят непосредственно к компиляции проекта. На этом этапе проявляются все скрытые ошибки и нестыковки. Компиляция разбивается на ряд последовательных подэтапов: сборка базы данных проекта, контроль соединений, логическая минимизация проекта, формирование загрузочного (конфигурационного) файла и др. На любом подэтапе могут возникать ошибки, требующие повторной компиляции после их коррекции. Результатом компиляции является загрузочный файл, т. е. конфигурационная информация для выбранной микросхемы ПЛ. Помимо этого, обычно создается файл отчета, содержащий всю информацию как о процессе компиляции, так и о его результатах.

5. Тестирование проекта.

Тестирование разработанного устройства – один из важнейших этапов проектирования, поскольку практически не бывает бездефектных проектов. Обнаружение дефектов проекта сложнейшая задача. Скорость и тщательность тестирования во многом зависят от разработчика.

В современных САПР наиболее распространено тестирование путем работы с редакторами временных диаграмм. Эти редакторы делятся на компилирующие и интерпретирующие. В многооконных САПР интерпретирующего типа просто отображаются результаты моделирования для текущего момента модельного времени во всех видах отображения проекта (сигналы в электрических схемах, топологии), легко изменить ход эксперимента и состав отображаемых сигналов. Достоинством компилирующих систем моделирования является минимизация временных затрат.

6. Определение временных характеристик разработанного устройства.

Современные САПР имеют внутри себя полную информацию о структуре проектируемого устройства и временных параметрах всех его компонентов, и это позволяет автоматизировать процесс вычисления разнообразных времен-

ных характеристик проекта. В САПР MAX+ plus II предусмотрено автоматическое вычисление трех основных классов временных параметров:

- минимальных и максимальных задержек между источниками (входными сигналами) и приемниками (выходными сигналами), информация о которых выдается в виде матрицы задержек;
- максимально возможной производительности устройства (пропускной способности) в виде максимальной частоты тактирования элементов памяти, используемых в проекте;
- времен предустановки и выдержки сигналов, гарантирующих надежную работу схем при фиксации сигналов в синхронных элементах памяти.

Многие САПР позволяют также выделять критические пути передачи и преобразования информации для схемного или топологического представления проекта.

7. Организация натурных экспериментов.

Последним этапом проектирования является этап экспериментальной проверки спроектированного устройства. При всей тщательности выполнения предыдущих этапов всегда существует вероятность того, что в проекте имеются дефекты, которые могут проявиться на этапе внедрения или даже штатного использования устройства и повлечь за собою тяжкие последствия.

Выполнение натурных экспериментов существенно увеличивает вероятность выпуска бездефектной продукции. Средства ускорения работ на этом этапе и возможности их переноса на ранние этапы разработки, т.е. до того момента, когда будет закончено изготовление конечного продукта, известны – это прототипные системы и средства проведения экспериментов с ними. Широкий спектр прототипных плат, содержащих микросхемы программируемой логики и дополнительную аппаратуру (прежде всего микросхемы быстродействующих ОЗУ), выпускается и поставляется различными зарубежными фирмами. Здесь можно указать средства фирм Altera (Demo Board); PLD Applications (платы PCI Bus Evaluation Board); Xilinx, Virtual Computer Corp Video Software (платы HOT PCI Design Kit) и др.

4. Лабораторная работа № 1.

Парные электронные «игральные кости»

4.1. Структурная схема устройства

Электронная версия игры в кости является классикой и входит во все каталоги изготовителей наборов для радиолюбителей и сборников схем устройств.

Как правило, требуется по крайней мере четыре или пять корпусов КМОП или ТТЛ, чтобы воспроизвести эквивалент одной единственной кости, снабженной семью светодиодами. Использование ПЛИМ решительно меняет ситуацию в лучшую сторону. Одного корпуса ПЛИМ вполне достаточно, чтобы создать на этот раз уже парные «кости», в то время как существует возможность сделать не копируемой внутреннюю схему, которую там запрограммировали. Схема парных «игральных костей», представленная на рис. 4.1, показывает, до какой степени использование ПЛИМ упрощает логические системы: одной единственной интегральной схемой ПЛИМ типа 16R8 и одного мультивибратора на двух транзисторах хватает для реализации устройства, имеющего уже довольно сложный алгоритм функционирования. Результатом разработки стала плата, основную часть которой занимают 14 светодиодов, образующих две «игральные кости». Каждое нажатие на кнопку «Вброс» должно вызывать новое выбрасывание костей.

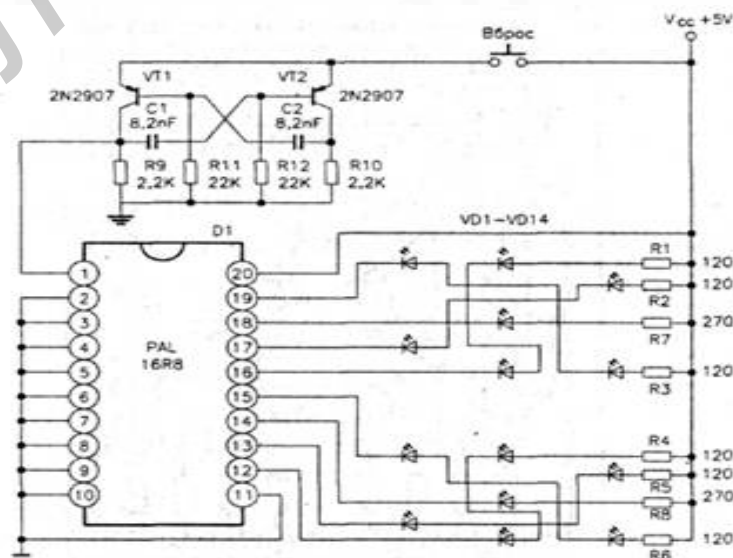


Рис. 4.1

В данной лабораторной работе для реализации парных электронных "интегральных костей" программируется ПЛИС фирмы Altera. Программирование ПЛИС данной фирмы производится с использованием САПР MAX+plus II.

Построение схемы производится в графическом редакторе (подменю Graphic Editor меню MAX+plus II либо «иконка» New на панели инструментов), расширение файла *.gdf. Окно редактора имеет ряд дополнительных пунктов основного меню и панель инструментов редактора, расположенную вертикально с левой стороны окна.

Для добавления символа используется диалоговое окно Enter Symbol (рис. 4.2) из меню Symbol либо меню, вызываемое правой кнопкой мыши, либо команда Double-Click. Выбрать и установить необходимый элемент можно двумя способами:

1. Набрать имя элемента (примитива, мега- или макрофункции) в окне Symbol Name диалогового окна Enter Symbol и нажать ОК.

2. Выбрать необходимую библиотеку в окне Symbol Libraries диалогового окна Enter Symbol и двойным щелчком левой клавиши мыши открыть её. Затем аналогичным образом выбрать необходимый элемент в окне Symbol File.

Для построения структурной схемы (рис. 4.3) необходимы следующие элементы:

Библиотека примитивов(PRIM):

D-триггер (DFF);

вход (INPUT);

выход (OUTPUT);

логический элемент НЕ (NOT);

логический элемент 2И (AND2);

логический элемент 2ИЛИ (OR2).

Для соединения функциональных элементов используется панель инструментов, расположенная в левой части главного окна.

Далее необходимо сохранить новый файл проекта (через меню File → Save) под именем COSTI (расширение будет присвоено автоматически).

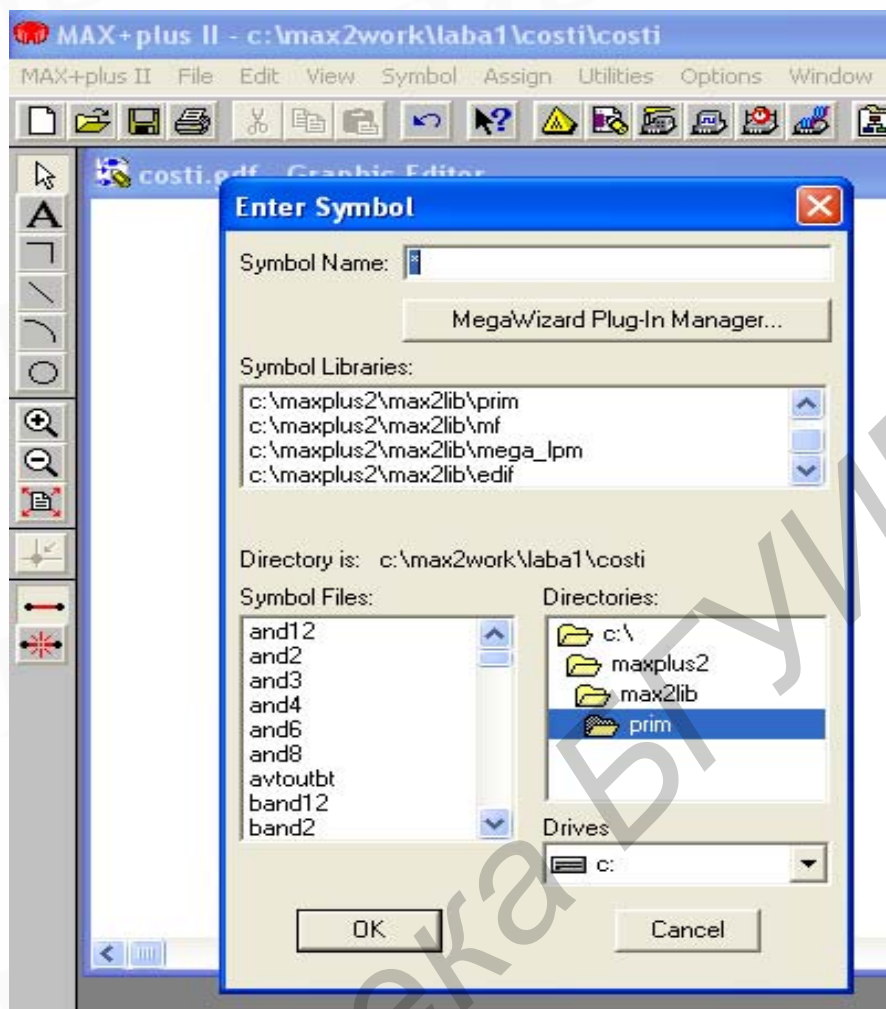


Рис. 4.2

Имя файла проекта следует обязательно привязать к имени проекта – это делается при выборе пункта Set Project to Current File (в подменю Project меню File главного меню рабочего окна).

4.2. Компиляция проекта

После построения структурной схемы выполняется компиляция проекта. Результатом компиляции является загрузочный файл, т.е. конфигурационная информация для выбранной микросхемы.

В процессе компиляции проявляются скрытые ошибки, которые были допущены при построении структурной схемы. После устранения ошибок проект необходимо повторно перекомпилировать. Проверка проекта осуществляет-

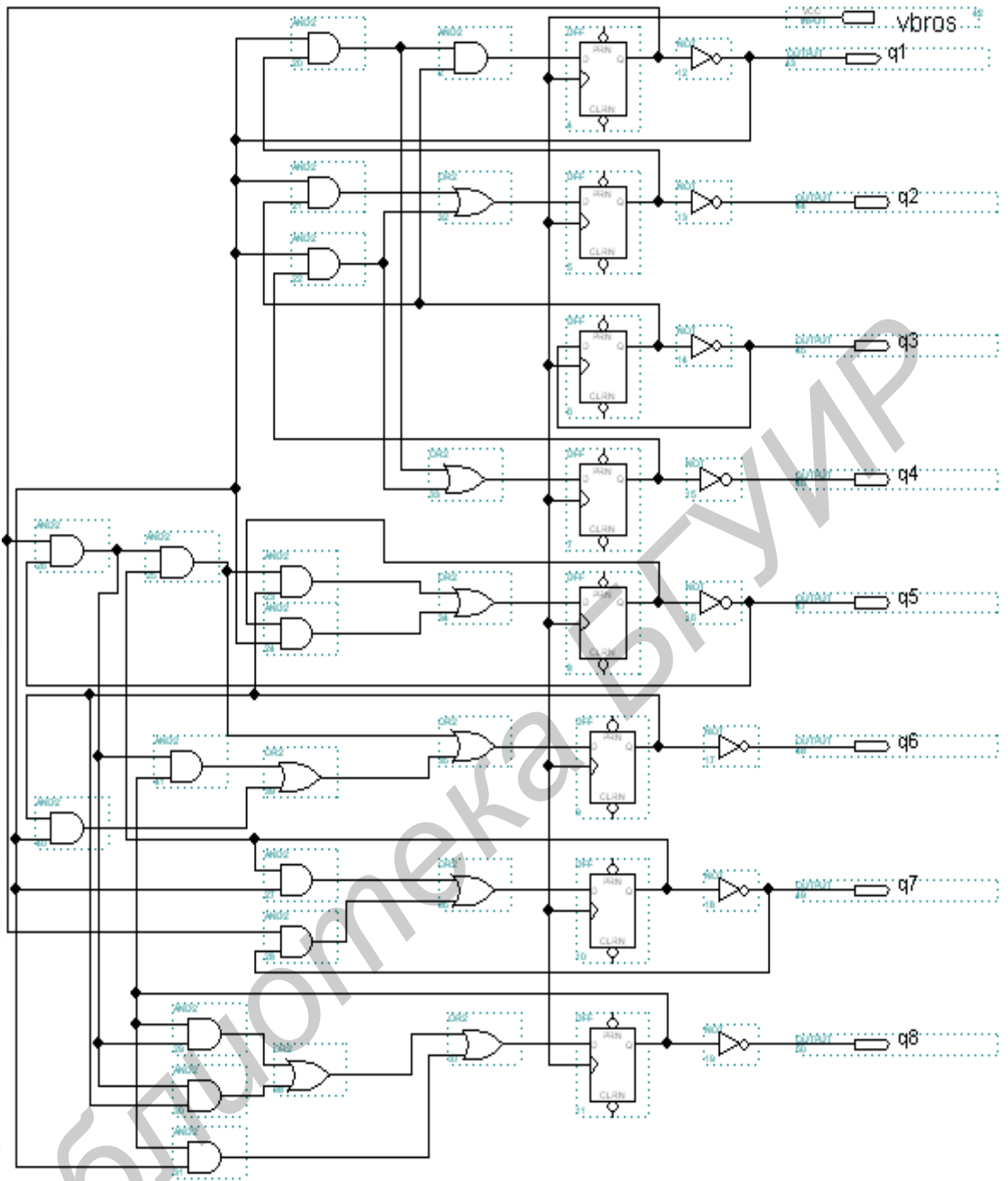


Рис. 4.3

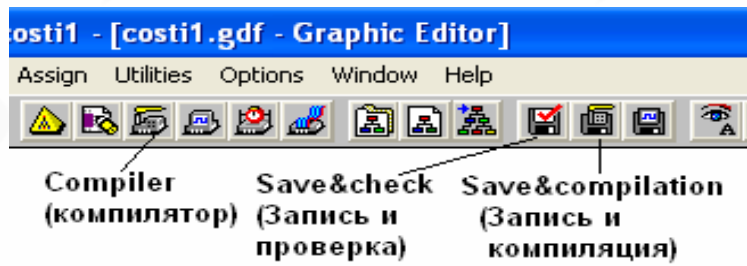


Рис. 4.4

ся командой Save&Check подменю Project меню File либо кнопкой на панели инструментов (рис. 4.4).

Для запуска компилятора используется подменю Compiler из меню MAX+plus II, либо команда Save&Compile подменю Project из меню File, либо кнопки, расположенные на панели инструментов.

Для начала компиляции необходимо нажать кнопку START (рис. 4.5). Для получения дополнительной информации об ошибках и возможных способах их устранения используются Messages – Compiler и Help on Message.

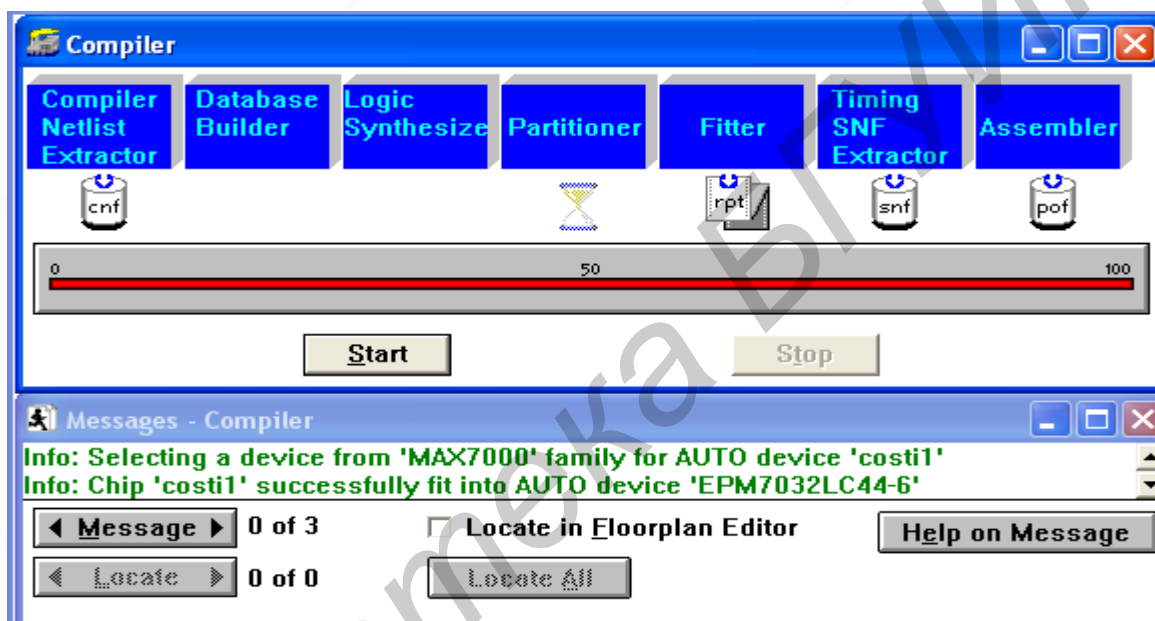


Рис. 4.5

4.3. Построение временных диаграмм

Для тестирования разработанного устройства в системе MAX+plus II используется редактор временных диаграмм (Waveform Editor). В начале работы с Waveform Editor необходимо создать Waveform Editor files (*.scf). Для этого после открытия Waveform Editor (MAX+plus II → Waveform Editor) в предложенном окне отображаются все входы (input) и выходы (output), для чего используется команда Insert Node меню Node либо меню, вызываемое нажатием правой кнопки мыши. Далее файл с расширением *.scf необходимо сохранить. Для анализа работы разрабатываемого устройства на входы подаются предполагаемые входные сигналы, а выходы задаются неопределенными (рис. 4.6).

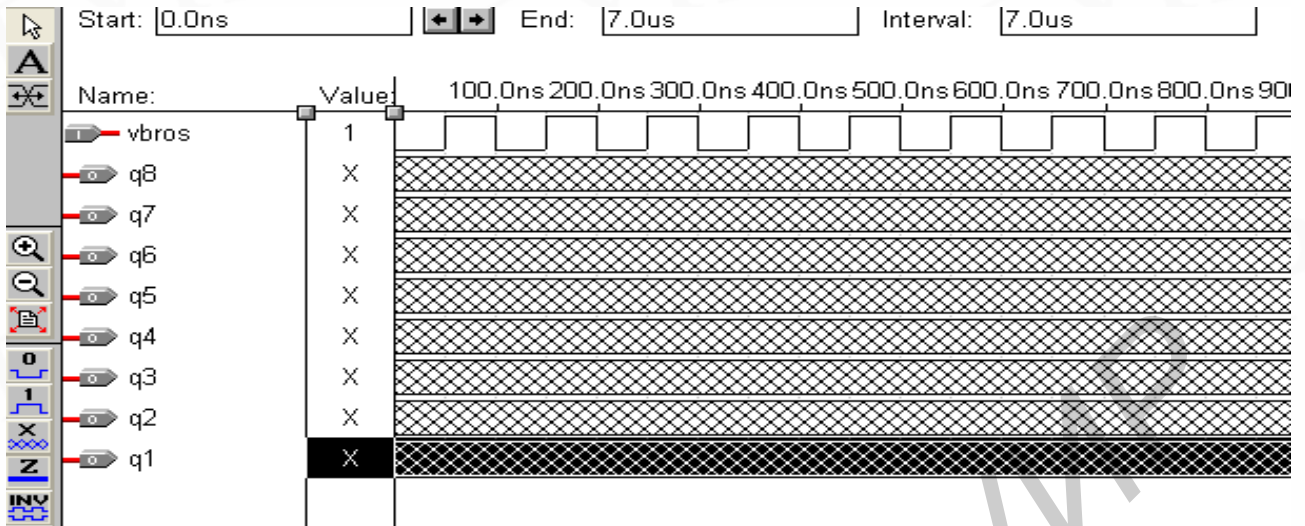


Рис. 4.6

Входные сигналы задаются командами на панели инструментов либо командами Overwright, Grow/Shrink, вызываемыми правой кнопкой мыши.

Для определения выходных сигналов используется Simulator, вызов которого осуществляется использованием подменю Simulator меню MAX+plus II либо кнопкой на панели инструментов (рис. 4.7). Запуск Simulator осуществляется нажатием кнопки START.

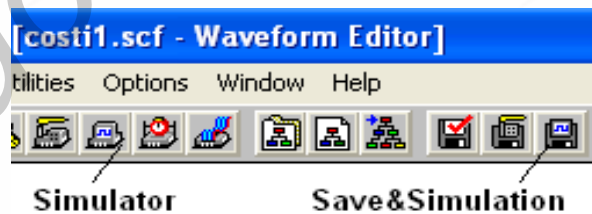


Рис. 4.7

4.4. Вычисление временных задержек

Важнейшими характеристиками современных цифровых устройств являются временные характеристики. Современные САПР позволяют полностью автоматизировать процесс вычисления временных задержек, т.к. содержат внутри себя полную информацию о структуре проектируемого устройства и временных параметрах всех его компонентов.

Для вычисления временных задержек в САПР MAX+plus II используется Timing Analyzer, вызов которого осуществляется из меню MAX+plus II. В САПР MAX+plus II предусмотрено автоматизированное вычисление трех основных видов временных задержек (для вызова которых используется панель инструментов (рис. 4.8)):

- минимальных и максимальных задержек между источниками (входными сигналами) и приемниками (выходными сигналами), информация о которых выдается в виде матрицы задержек (Delay Matrix);
- максимально возможной производительности устройства (пропускной способности) в виде максимальной частоты тактирования элементов памяти, используемых в проекте (Registered Performance);
- времен предустановки и выдержки сигналов, гарантирующих надежную работу схем при фиксации сигналов в синхронных элементах памяти (Setup/Hold Time Analysis).

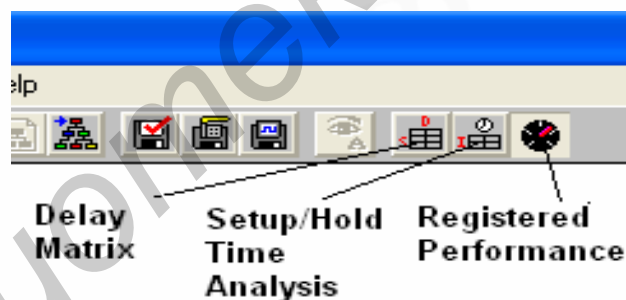


Рис. 4.8

5. Лабораторная работа № 2.

Проектирование ЦУ ввода-вывода данных

5.1. Структурная схема устройства

Разрабатываемое устройство позволяет либо записывать по запросу параллельный восьмиразрядный код в буферное ОЗУ, либо выводить байт из заданного адреса буферного ОЗУ в форме последовательного кода.

Перечисленные ниже требования определяют основные блоки проектируемого устройства и их взаимодействие:

- объем буферного ОЗУ 256 восьмиразрядных слов;
- запись в ОЗУ осуществляется сигналами внешнего управляющего устройства;
- внешнее чтение статуса устройства позволяет определить состояние его выходного регистра (пуст или полон);
- вывод последовательного кода осуществляется по запросу приемника последовательного кода и сопровождается стробирующими сигналами со стороны разрабатываемого устройства.

Блок-схема устройства приведена на рис. 5.1. Схема отображает следующие основные процессы.

Внешнее управляющее устройство (процессор) обеспечивает запись байтов в ОЗУ, подавая на него помимо данных также 8-разрядный адресный код и строб записи Write. Преобразователь параллельного кода в последовательный представляет в своей основе сдвигающий регистр, загружаемый параллельно байтом из ОЗУ, и затем по командам из устройства управления выводящий последовательные данные во внешнее устройство – приемник последовательного кода. Появление разрядов последовательного кода отмечается во времени сигналами стробирования Strob. Загрузка данных в преобразователь параллельного кода инициируется процессором по сигналу Read, адрес загружаемых данных должен быть перед этим задан процессором.

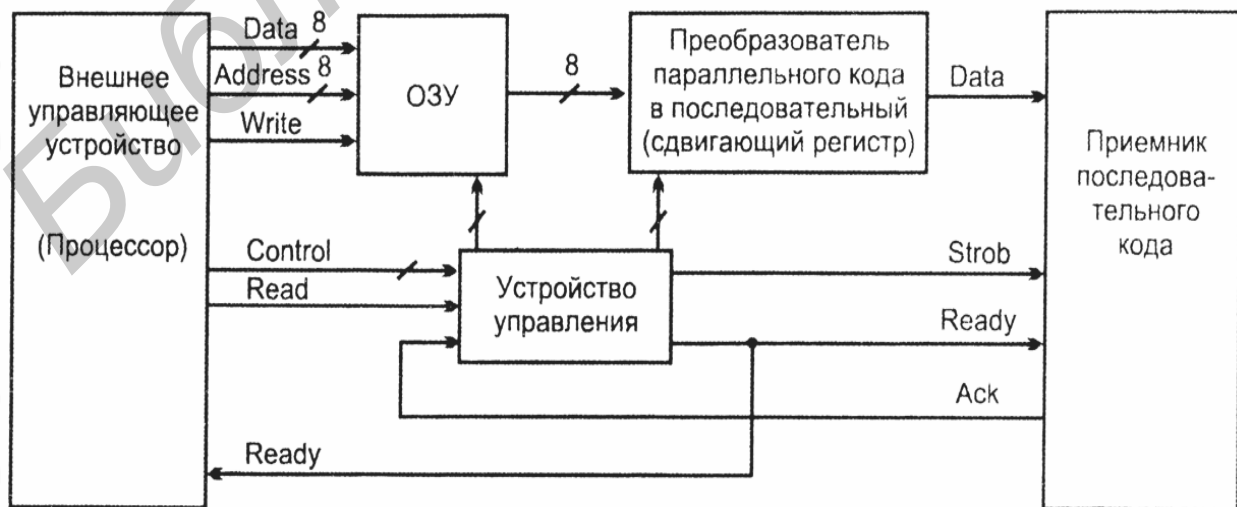


Рис. 5.1

Готовность преобразователя кода к передаче данных индуцируется сигналом Ready, поступающим как на приемник последовательных данных, так и на соответствующий вход процессора. Асинхронный характер обмена с приемником данных обеспечивается сигналом разрешения передачи Ask со стороны приемника. В состав сигналов шины Control входят сигналы тактирования, сброса и другие сигналы.

Сопоставляя функциональный состав библиотеки САПР MAX+plus II и блоков структурной схемы, представленной на рис. 5.1, для реализации рассматриваемого устройства из состава библиотеки САПР можно использовать следующий набор библиотечных параметризуемых модулей (LPM):

- блок ОЗУ (LPM_RAM_DQ) с организацией 256x8;
- 8-разрядный сдвигающий регистр выходного кода (LPM_SHIFTREG);
- счетчик сдвигов регистра вывода на 8 состояний (LPM_COUNTER).

Структурная схема устройства, включающая эти операционные блоки и автомат, управляющий считыванием кода из ОЗУ и его преобразование в последовательную форму (AvtOutBt), имеет вид, приведенный на рис. 5.2.

Кроме указанных выше базовых блоков, в схеме присутствует ряд дополнительных элементов. Условные обозначения всех элементов схемы соответствуют стандарту, принятому в САПР MAX+plus II. Необходимость введения дополнительных элементов (инвертора, четырех D-триггеров и двух схем 2ИЛИ-НЕ) диктуется требованиями временной аппаратной совместимости отдельных блоков схемы.

Для создания проектируемого устройства ввода-вывода данных используется графический редактор (подменю Graphic Editor из меню MAX+ plus II либо «иконка» New на панели инструментов) расширение файла *.gdf.

Окно редактора имеет ряд дополнительных пунктов основного меню и панель инструментов редактора, расположенную вертикально с левой стороны окна. Сохранение нового файла проекта осуществляется через меню File → Save под именем PROJECT (расширение будет присвоено автоматически) в каталоге \Laba2. Имя файла проекта следует обязательно привязать к имени – это

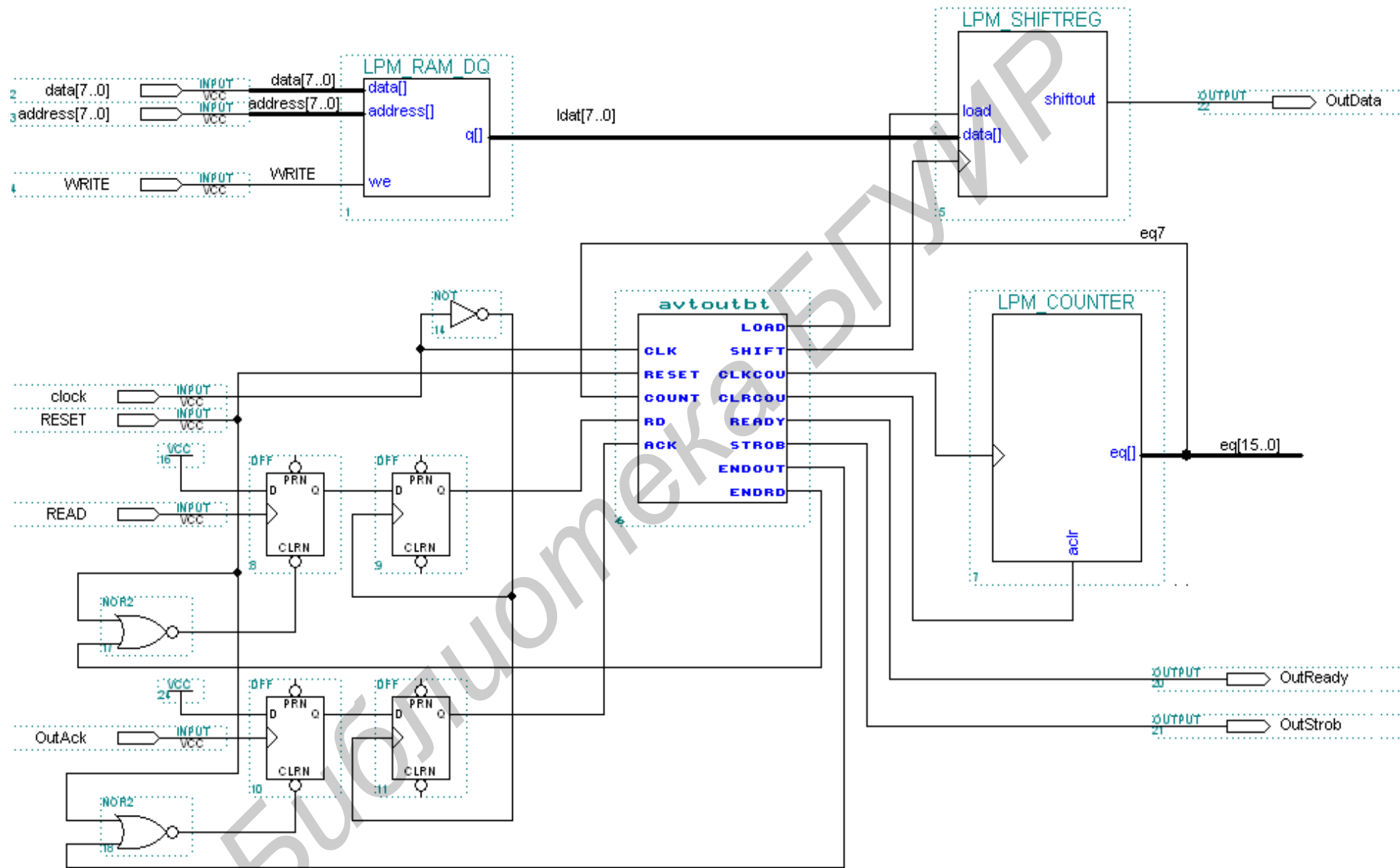


Рис. 5.2

делается при выборе пункта Set Project to Current File (в подменю Project из меню File главного меню рабочего окна).

Для создания графического проекта рекомендуется использовать библиотеки примитивов (\MAXPLUS\MAX2LIB\PRIM), макрофункций (\MAXPLUS\MAX2LIB\MF) и параметризованных мегафункций (\MAXPLUS\MAX2LIB\MEGA_LPM), а также созданный элемент управления AvtOutBt (\MAX2WORK\Laba2).

Для построения структурной схемы используются следующие элементы:

Библиотека созданных элементов:

Элемент управления (AvtOutBt).

Библиотека параметризуемых модулей (LPM):

Блок ОЗУ (LPM_RAM_DQ);

8-разрядный сдвигающий регистр (LPM_SHIFTRREG);

счетчик сдвигов регистра вывода на восемь состояний (LPM_COUNTER).

Библиотека примитивов (PRIM):

D-триггер (DFF);

вход (INPUT);

выход (OUTPUT);

логический элемент НЕ (NOT);

логический элемент 2ИЛИ-НЕ (NOR);

логическая единица (VCC).

Для размещения элементов в окне редактора используется диалоговое окно Enter Symbol, которое можно открыть через меню Symbol основного меню редактора (пункт Enter Symbol). Однако удобнее вызывать это окно двойным щелчком левой клавиши мыши по свободному пространству открытого окна редактора, после чего выбранный элемент будет размещён именно в этом месте (позиционирование элемента производится по верхнему левому углу условного обозначения элемента).

5.2. Создание автомата управления AvtOutBt

Для создания автомата управления AvtOutBt используется текстовый редактор (подменю Text Editor меню Max+plus II). Для описания работы управляющего автомата используется язык AHDL.

Основу алгоритма составляют два последовательно выполняемых блока. Первый блок по сигналу Rd автомата, образуемому из сигнала READ внешнего управляющего устройства, считывает байт из памяти, а затем загружает его в регистр сдвига. Второй блок содержит последовательность действий, которая в ответ на запускающий внешний сигнал OutAck, преобразуемый в сигнал Ack автомата, осуществляет циклический вывод на выходной контакт OutData данных из регистра сдвига и при этом сопровождает каждый бит стробирующим сигналом OutStrob (контакт автомата Strob). Количество требуемых итераций цикла подсчитывает счетчик сдвигов. Необходимую для правильной работы автомата синхронизацию асинхронных сигналов READ и OutAck выполняют дополнительно введенные в схему цепочки D-триггеров.

Тело программы, описывающей работу управляющего автомата, представлено ниже.

```
SUBDESIGN AvtOutBt
(CLK, Reset, count, RD, ACK:INPUT;
   load, shift, clkcou, clrcou, ready, strob, ENDOut, ENDRd : OUTPUT;)
VARIABLE
TempReady:NODE;
sreg:MACHINE WITH STATES (Idle, BegLdBt, LdBt, EndLdBt,
   WaitOut, StateStbOut, StateCount, Quit);
Begin
   sreg.clk=CLK;
   ready=DFF(TempReady,CLK,VCC,VCC);
   IF (Reset) THEN
     sreg=Idle;
     load=GND; shift=GND; clkcou=GND; clrcou=GND;
```

```

    TempReady=GND; strob=GND; ENDOut=GND; ENDRd=GND;
ELSE
CASE sreg IS
    WHEN Idle=>
        load=GND; shift=GND; clkcou=GND; clrcou=GND;
        TempReady=GND; strob=GND; ENDOut=GND; ENDRd=GND;
        IF (RD) THEN sreg=BegLdBt; END IF;
        IF (!RD) THEN sreg=Idle;END IF;
    WHEN BegLdBt=>
        load=VCC; shift=GND; clkcou=GND; clrcou=GND;
TempReady=GND; strob=GND; ENDOut=GND; ENDRd=GND;
        sreg=LdBt;
    WHEN LdBt=>
        load=VCC; shift=VCC; clkcou=GND; clrcou=GND;
TempReady=GND; strob=GND; ENDOut=GND; ENDRd=GND;
        sreg=EndLdBt;
    WHEN EndLdBt=>
        load=GND; shift=GND; clkcou=GND; clrcou=GND;
TempReady=GND; strob=GND; ENDOut=GND; ENDRd=VCC;
        IF (RD) THEN sreg=EndLdBt; END IF;
        IF (!RD) THEN sreg=WaitOut; END IF;
    WHEN WaitOut=>
        load=GND; shift=GND; clkcou=VCC; clrcou=VCC;
TempReady=VCC; strob=GND; ENDOut=GND; ENDRd=GND;
        IF (ACK) THEN sreg=StateStbOut; END IF;
        IF (!ACK) THEN sreg=WaitOut; END IF;
    WHEN StateStbOut=>
        load=GND; shift=GND; clkcou=GND; clrcou=GND;
TempReady=VCC; strob=VCC; ENDOut=GND; ENDRd=GND;
        IF (count) THEN sreg=Quit; END IF;

```

```

        IF (!count) THEN sreg=StateCount; END IF;
    WHEN StateCount=>
        load=GND; shift=VCC; clkcou=VCC; clrcou=GND;
    TempReady=VCC; strob=GND; ENDOut=GND; ENDRd=GND;
        sreg=StateStbOut;
    WHEN Quit=>
        load=GND; shift=GND; clkcou=GND; clrcou=GND;
    TempReady=GND; strob=GND; ENDOut=VCC; ENDRd=GND;
        IF (ACK) THEN sreg=Quit; END IF;
        IF (!ACK) THEN sreg=Idle; END IF;
    END CASE;
END IF;
END;

```

Далее проект необходимо откомпилировать. После компиляции осуществляется создание символа проекта AvtOutBt.

Создание символа проекта осуществляется через подменю Project, в котором следует выбрать пункт Create Default Symbol – этот пункт становится доступным только после закрытия окна компилятора. Созданный символ будет помещён в каталог проекта. Использование созданных символов так же, как и элементов других библиотек, производится через диалоговое окно Enter Symbol.

5.3. Настройка параметризованных модулей

Понятие параметризованных модулей соответствует возможности настроить выбранный библиотечный элемент на определенный режим функционирования, на определенную разрядность данных, их полярность и т. д.

Параметры модулей используемых при проектировании устройства ввода-вывода данных, приведены ниже.

Таблица 5.1

Параметры блока ОЗУ (LPM_RAM_DQ)

PORTS		
Name	Status	Inversion
address[LPM_WIDTHHAD-1..0]	Used	None
data[LPM_WIDTH-1..0]	Used	None
inclock	Unused	None
outclock	Unused	None
q[LPM_WIDTH-1..0]	Used	None
we	Used	None
PARAMETERS		
Name	Value	
LPM_ADDRESS_CONTROL	"UNREGISTERED"	
LPM_FILE	<none>	
LPM_INDATA	"UNREGISTERED"	
LPM_NUMWORDS	<none>	
LPM_OUTDATA	"UNREGISTERED"	
LPM_WIDTH	8	
LPM_WITHAD	8	

Таблица 5.2

Параметры 8-разрядного сдвигающего регистра (LPM_SHIFTREG)

PORTS		
Name	Status	Inversion
aclr	Unused	None

aset	Unused	None
clock	Used	None
data[LPM_WIDTH-1..0]	Used	None
enable	Unused	None
load	Used	None
q[LPM_WIDTH-1..0]	Used	None
sclr	Unused	None
shiftin	Unused	None
shiftout	Used	None
sset	Unused	None
PARAMETRES		
Name		Value
LPM_AVALUE		<none>
LPM_DIRECTION		"RIGHT"
LPM_SVALUE		<none>
LPM_WIDTH		8

Таблица 5.3

Параметры счетчика сдвигов (LPM_COUNTER)

PORTS		
Name	Status	Inversion
1	2	3
aclr	Used	None
aconst	Unused	None
aload	Unused	None
aset	Unused	None
cin	Unused	None
clc_en	Unused	None
clock	Used	None

cnt_en	Unused	None
count	Unused	None
data[LPM_WIDTH-1..0]	Unused	None
eq[15..0]	Used	None
q[LPM_WIDTH-1..0]	Unused	None
sclr	Unused	None
sconst	Unused	None
sload	Unused	None
sset	Unused	None
updown	Unused	None
PARAMETRES		
Name	Value	
LPM_AVALUE	<none>	
LPM_DIRECTION	<none>	
LPM_MODULUS	<none>	
LPM_SVALUE	<none>	
LPM_WIDHT	16	

5.4. Компиляция проекта

После построения структурной схемы необходима компиляция проекта. Результатом компиляции является загрузочный файл, т.е. конфигурационная информация для выбранной микросхемы. Перед компиляцией проекта необходимо выбрать тип PLD, для которого осуществляется компиляция.

Объем ОЗУ проектируемого устройства ввода-вывода данных делает целесообразным выбор в качестве БИС PLD семейства FLEX10K. Выбор семейства ПЛИС осуществляется в подменю Device меню Assign (Device Family→FLEX10K; Devices→AUTO).

Перед компиляцией необходимо выполнить проверку корректности введенного проекта. Проверка осуществляется через подменю Project (меню File

главного меню рабочего окна) путём выбора пункта Save&Check или щелчком левой кнопки мыши на «иконке» соответствующего инструмента основной панели инструментов (рис. 5.3). После устранения ошибок проект необходимо повторно перекомпилировать.

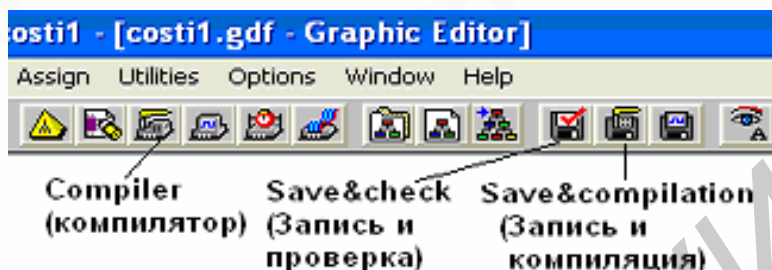


Рис. 5.3

Компиляция осуществляется через подменю Project путём выбора пункта Save&Compile или вызовом подменю Compile меню MAX+plusII либо с помощью соответствующего инструмента основной панели инструментов. Для начала компиляции необходимо нажать кнопку START. Для получения дополнительной информации об ошибках и возможных способах их устранения используются Messages – Compiler и Help on Message.

5.5. Построение временных диаграмм

Для тестирования разработанного устройства в системе MAX+plus II используется редактор временных диаграмм (Waveform Editor). В начале работы с Waveform Editor необходимо создать Waveform Editor files (*.scf). Файл создается после открытия Waveform Editor (Max+plus II→Waveform Editor) отображением используемых входов (Input) и выходов (Output) (используется команда Insert Node меню Node либо меню, вызываемое нажатием правой кнопки мыши). Далее файл необходимо сохранить.

Для анализа работы разработанного устройства на входы подаются предполагаемые входные сигналы, а выходы задаются неопределенными сигналами. Для проектируемого устройства ввода-вывода данных входные сигналы могут быть заданы следующим образом (рис. 5.4).

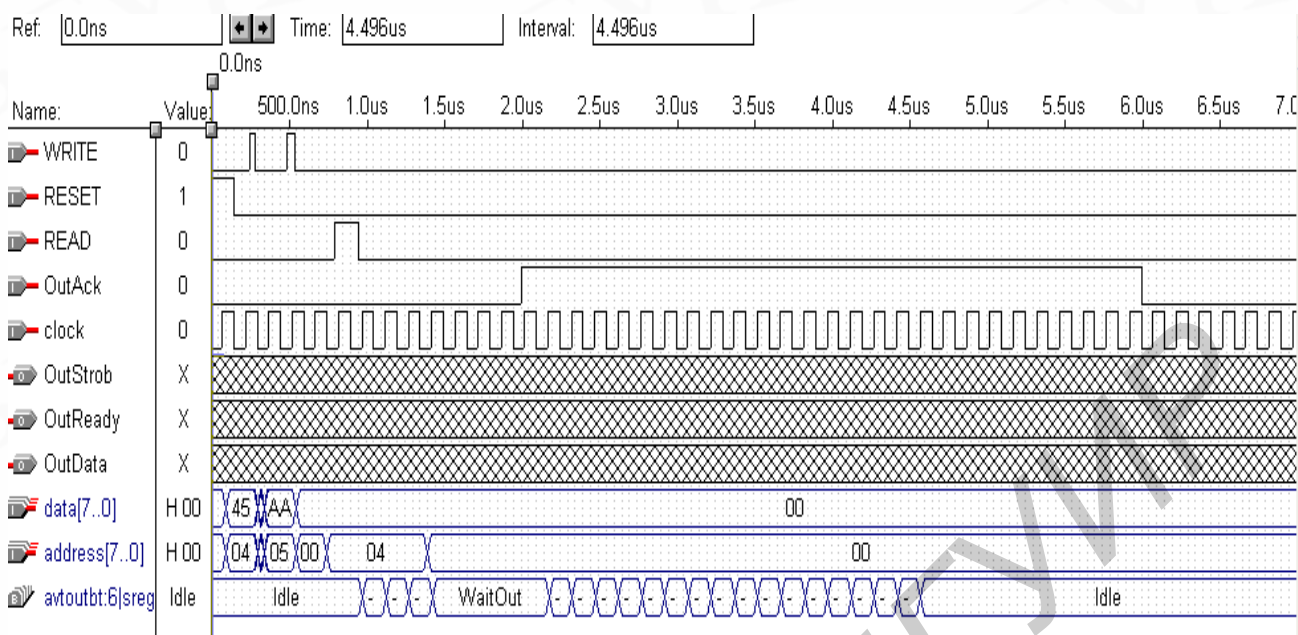


Рис. 5.4

Входные сигналы задаются, используя команды на панели инструментов либо команды Overwright, Grow/Shrink, вызываемые правой кнопкой мыши.

Для определения выходных сигналов используется Simulator, вызов которого осуществляется использованием подменю Simulator меню MAX+plus II либо кнопкой на панели инструментов. Запуск Simulator осуществляется нажатием кнопки START.

5.6. Вычисление временных задержек

Вычисление временных задержек осуществляется аналогично п. 4.4.

6. Контрольные вопросы

1. Классификация ПЛИС по структурной организации.
2. Стандартные ПЛИС.
3. Макроматрицы (МАСН-устройства).
4. Матричные таблицы (МАХ-устройства).
5. Программируемые пользователем вентиляльные матрицы.

6. Сложные PLD (Complex PLD).
7. СБИС программируемой логики смешанной архитектуры.
8. Файлы проекта, вспомогательные файлы, проекты MAX+plus II.
9. Программные модули MAX+plus II.
10. Основные этапы разработки цифрового устройства в САПР MAX+plus II.

Литература

1. Соловьев В.В., Васильев А.Г. Программируемые логические интегральные схемы и их применение. – Мн.: Беларуская навука, 1998. – 270 с.
2. Угрюмов Е.П. Цифровая схемотехника: Учеб. пособие. – СПб.: БХВ – Петербург, 2000. – 528 с.
3. Комолов Д.А., Мьяльк Р.А., Зобенко А.А., Филиппов А.С. Системы автоматизированного проектирования фирмы Altera MAX+plus II и Quartus II. – М.: РадиоСофт, 2002. – 355 с.

Учебное издание

Прищеп Сергей Леонидович,
Ильина Екатерина Алексеевна

*ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ СХЕМ С помощью САПР
MAX+PLUS II ФИРМЫ ALTERA*

Учебно-методическое пособие
по курсу
«САПР цифровых устройств»
для студентов специальности «Телекоммуникационные системы»
дневной формы обучения

Редактор Н.В. Гриневич

Корректор Е.Н. Батурчик

Подписано в печать 09.02.2005.
Гарнитура «Таймс».
Уч.-изд. л. 2,0.

Формат 60×84 1/16.
Печать ризографическая.
Тираж 100 экз.

Бумага офсетная.
Усл. печ. л. 3,26.
Заказ 22.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Лицензия на осуществление издательской деятельности №02330/0056964 от 01.04.2004.
Лицензия на осуществление полиграфической деятельности №02330/0133108 от 30.04.2004
220013, Минск, П. Бровки, 6.