*Баран Е. Д.* – старший преподаватель кафедры систем сбора и обработки данных Новосибирского государственного технического университета;
*Романов А. Ю.* – канд. техн. наук, доцент Московского института электроники и математики им. А. Н. Тихонова Национального исследовательского университета «Высшая школа экономики».

**Баран Е.Д., Романов А.Ю.**
Проектирование реконфигурируемых систем в LabVIEW FPGA. – М.: ДМК Пресс, 2023. – 646 с.: ил.

**ISBN 978-5-93700-170-2**

В этой книге изложены основы проектирования реконфигурируемых систем в среде графического программирования LabVIEW, дополненной модулем LabVIEW FPGA. Приведен обзор разновидностей ПЛИС, модулей ввода-вывода классической архитектуры и модулей с реконфигурируемыми каналами ввода-вывода производства National Instruments. Рассмотрены основные компоненты и инструменты среды проектирования, на реальных примерах показана методика разработки и отладки распределенных и встраиваемых систем измерения, управления и тестирования. Описан полный цикл проектирования, включая этапы создания исполняемых файлов и инсталляторов, а также развертывания систем на целевых устройствах.

Содержание книги отражает опыт разработок, выполненных в учебном центре «Центр технологий National Instruments» Новосибирского государственного технического университета, опыт обучения на курсах повышения квалификации и преподавания соответствующих дисциплин студентам.

Книга, которую вы держите в руках, продолжает серию «Книжная полка Истового Инженера», которая издается при поддержке компании YADRO. Это издание подготовлено к публикации Московским институтом электроники и математики им. А. Н. Тихонова НИУ ВШЭ совместно с «ДМК Пресс».



ISBN 978-5-93700-170-2

# Содержание

# Отзывы о книге

Современный мир невозможно представить без сложных цифровых устройств, которые претерпели существенные изменения за последнее столетие. Задачи, для решения которых пару десятков лет назад требовались огромные вычислительные машины, занимавшие целые здания, сейчас способен решить процессор мобильного телефона, который может уместиться в ладони. Сейчас логические устройства встраиваются в различную технику, окружающую человека: автомобили способные управляться без участия человека, умные бытовые устройства способные анализировать поведения человека и предлагать ему разнообразные сценарии использования и т.д. Поэтому задача разработки архитектуры новых логических устройств является важной и актуальной.

Одним из инструментов разработки логических цифровых устройств является ПЛИС (программируемая логическая интегральная схема). Благодаря своей реконфигурируемой архитектуре ПЛИС может быть встроена как логический узел в различные вычислительные устройства. Одним из инструментов работы с платой ПЛИС является расширение среды LabVIEW под названием LabVIEW FPGA. Данное расширение позволяет провести полный цикл разработки логического устройства с ПЛИС в среде LabVIEW, вплоть до его переноса и развертывания в других САПР.

Стоит отметить, что на русском языке материалов по работе с данным расширением мало и они не систематизированы, поэтому рассматриваемая книга является большим подспорьем начинающим инженерам и профессионалам, которые хотят освоить принципы работы с платами ПЛИС в среде LabVIEW. LabVIEW FPGA может рассматриваться как средство проектирования для встраиваемых модулей с ПЛИС от компании National Instruments, так и как средство высокоуровневого синтеза для подготовки проекта к реализации на других аппаратных платформах.

Книга «Проектирование реконфигурируемых систем в LabVIEW FPGA» хорошо структурирована, изложена грамотным русским языком, легким для чтения. В ней подробно и полно раскрыто проектирование в LabVIEW встраиваемых систем на ПЛИС.

Эта книга будет полезна как специалистам, так и начинающим разработчикам.

*Иванников А.Д.,*
*д.т.н., проф., Главный научный сотрудник Института*
*проблем проектирования в микроэлектронике*
*Российской академии наук.*

*Дорогой Читатель!*

Ты держишь в руках книгу по проектированию реконфигурируемых систем измерения и управления, написанную профессионалами-практиками: разработчиком измерительных систем и разработчиком систем на кристалле; оба руководят вузовскими исследовательскими лабораториями и имеют большой преподавательский опыт – так что не сомневайся, они точно знают, как и о чём нужно писать в учебной литературе.

Книга стала логическим развитием предыдущего издания, увидевшего свет уже почти 15 лет назад, и, конечно, требующего замены. Это учебное пособие будет крайне полезно в качестве входного портала для начинающих разработчиков, интересующихся студентов, а также и в качестве концентрированного справочника для практикующих специалистов. Оно удачно и систематически сочетает в себе 3 важнейших аспекта:

1) методология проектирования – включая необходимые сведения о ПЛИС как таковых, устройствах ввода/вывода, архитектуре конечных систем, интерфейсе для работы с периферийными устройствами;

2) введение в среду разработки (инструмент проектирования) – включая необходимые описания пользовательского интерфейса LabVIEW, правила и приёмы программирования на встроенном графическом языке G, необходимые описания компонентов встроенной библиотеки FPGA;

3) множество простых и комплексных практических примеров (практику проектирования), необходимых для закрепления материала, демонстрирующих нюансы использования имеющихся возможностей.

Читатель! Эта книга даст тебе то самое «знание того, куда нужно ударить». Не теряй времени, воспользуйся короткой дорогой к профессионализму!

*Самбурский Лев Михайлович,*
*к.т.н., доцент МИЭМ НИУ ВШЭ,*
*преподаватель курсов «Электроника»,*
*«Компоненты инфокоммуникационных устройств» и др.*
*Уже больше 10 лет использует LabVIEW*
*в преподавании и проектной работе.*

# Об авторах

**Баран Ефим Давыдович** – старший преподаватель кафедры систем сбора и обработки данных (ССОД) Новосибирского государственного технического университета (НГТУ, до 1992 года – Новосибирский электротехнический институт – НЭТИ).

В 1969 г. защитил на кафедре информационно-измерительной техники (ныне ССОД) дипломный проект и работал в научно-исследовательском секторе на должностях от инженера до с.н.с., в качестве ассистента-совместителя вел на кафедре лабораторные и практические занятия по нескольким техническим дисциплинам.

С 1984 г. – старший преподаватель, основные курсы: «Микропроцессоры и микроконтроллеры», «Измерительные информационные системы», «Системы контроля и диагностики», «SCADA-системы», «Программирование в LabVIEW».

С 2004 г. руководитель созданного на базе кафедре ССОД НГТУ учебного центра «Центр технологий National Instruments»: *https://nitec.nstu.ru*. В лабораториях центра обучаются студенты нескольких факультетов НГТУ, проводятся курсы повышения квалификации для сотрудников и руководителей НИИ и КБ, промышленных предприятий, преподавателей образовательных учреждений высшего и среднего профессионального образования.

Баран Е.Д. разрабатывал и руководил разработками приборов и систем различного назначения, в том числе приборов и систем для функционального контроля и диагностики средств микропроцессорной техники, систем автоматизации научного эксперимента, учебных лабораторных стендов и лабораторных практикумов по ряду дисциплин. Работы в этих направлениях продолжаются и в настоящее время.

Автор 55 публикаций и 15 изобретений.

**Романов Александр Юрьевич** – доцент Московского института электроники и математики им. А.Н. Тихонова Национального исследовательского университета «Высшая школа экономики» (МИЭМ НИУ ВШЭ). В 2009 г. закончил магистратуру в Харьковском политехническом институте, работал в Киевском политехническом институте им. Сикорского. С 2014 г. работает в МИЭМ НИУ ВШЭ, где возглавляет лабораторию САПР (*https://miem.hse.ru/edu/ce/cadsystem*), специализирующуюся на проектной деятельности, а также разработке цифровых систем на ПЛИС / микроконтроллерах, робототехнических комплексов, аппаратных реализаций систем искусственного интеллекта, многопроцессорных систем, систем удаленного доступа к лабораторному оборудованию и т.д. В 2015 г. защитил диссертацию в Институте проблем проектирования в микроэлектронике РАН (г. Зеленоград), является автором более 150 научных статей, патентов и книг. Более подробно об учебном процессе в лаборатории можно узнать из интервью: *https://miem.hse.ru/news/364316102.html*.

# Предисловие Е.Д. Барана

Система LabVIEW корпорации National Instruments заслуженно пользуется популярностью среди научных сотрудников и инженеров, которые разрабатывают системы автоматизации экспериментальных исследований, системы автоматизированного управления производственными процессами и испытаний объектов в различных отраслях промышленности и т.д. Основные достоинства LabVIEW – интуитивно понятный язык графического программирования, развитые прикладные библиотеки функций и простота интеграции с оборудованием ведущих производителей. позволяют в кратчайшие сроки решать сложные задачи, причем для этого не обязательно обладать знаниями и опытом профессионального программиста.

Один из модулей LabVIEW – модуль LabVIEW FPGA предоставляет возможность создавать реконфигурируемые системы, каналы ввода-вывода которых и встроенные специализированные процессоры обработки данных разрабатываются в едином процессе интеграции технических средств и программного обеспечения. Расширение функционала и улучшение технических характеристик систем достигаются благодаря конфигурированию структуры FPGA – программируемой логической интегральной схемы (ПЛИС), встроенной в готовые функциональные блоки, из которых собирается система. Чрезвычайно важно, что и разработка структуры FPGA и разработка прикладного программного обеспечения для процессора классической архитектуры выполняются в одной среде проектирования с помощью одних и тех же инструментальных средств.

Концепция реконфигурируемых систем, основные их элементы и техника проектирования таких систем описаны в первом издании книги, которое вышло еще в 2009 году. За прошедшее с тех пор время радикально изменилась элементная база: увеличились степень интеграции и быстродействие FPGA, значительно расширился состав функциональных узлов. Теперь в один кристалл интегрированы многоядерные процессоры, аналого-цифровые и цифро-аналоговые преобразователи, стандартные интерфейсные порты и т.п. Программируемые матрицы простых логических элементов эволюционировали в системы на кристалле (Systems-on-Chip).

Для программирования систем на кристалле компания Xilinx разработала новый инструментарий – в системе Vivado традиционные текстовые языки программирования адаптированы к задачам проектирования электронных схем. Благодаря этому профессиональные программисты теперь могут разрабатывать не только надстройку над предоставленной им технической базой, но и саму базу. В Vivado усовершенствован и процесс проектирования, что позволяет существенно сократить время, необходимое для конфигурирования FPGA.

Корпорация National Instruments, используя современные микроэлектронные компоненты, разрабатывает все более совершенные функциональные блоки для систем измерения, тестирования и управления. На основе новых ре-

конфигурируемых модулей ввода-вывода, контроллеров и модульных измерительных приборов стало возможным создавать системы, обладающие большей производительностью и заметно расширенным функционалом. Библиотеки модуля LabVIEW FPGA пополнились новыми программными компонентами для реализации сложных алгоритмов обработки данных, представленных не только целыми числами, но и данными, представленными в формате чисел с фиксированной и плавающей запятой. Появились библиотеки функций для решения прикладных задач цифровой радиосвязи, обработки изображений, электроэнергетики, усовершенствованы средства работы с каналами ввода-вывода, инструменты отладки кода, выполняемого в FPGA, функции для упрощения интерфейса с программным обеспечением верхнего уровня и многое другое. Хотя процесс проектирования структуры FPGA в новой системе проектирования Vivado существенно изменился, тем, кто разрабатывает реконфигурируемые системы в LabVIEW FPGA, по-прежнему не обязательно знать специфические особенности этого процесса. Достаточно щелчка кнопки, чтобы инициировать компиляцию созданной структуры кристалла, а LabVIEW FPGA выберет необходимый компилятор – ISE Xilinx или Vivado, и спустя необходимое для компиляции время двоичный файл будет загружен и начнет выполняться в FPGA.

В предлагаемой читателям книге рассмотрены далеко не все упомянутые новшества LabVIEW FPGA, это потребовало бы значительно увеличить ее объем. Первое издание [В-1] было напечатано ограниченным тиражом, и не все желающие смогли ознакомиться с принципами проектирования реконфигурируемых систем. Поэтому в этом издании сохранены главы, в которых описаны предпосылки появления этого перспективного класса систем, техника программирования в LabVIEW, технические средства и особенности разработки на их основе систем традиционной архитектуры. Содержание этих глав обновлено и переработано.

Системы цифровой радиосвязи, а также вопросы программно-технического моделирования при разработке сложных объектов, кратко описанные в главе «Примеры применения технологий реконфигурируемого ввода-вывода» в первом издании книги, в настоящее издание не вошли. Принципы построения реконфигурируемых систем и инструментарий, с помощью которого их разрабатывают, одни и те же для любых областей техники, а специфические особенности, связанные преимущественно с алгоритмами обработки данных и областью применения систем, заслуживают более полного изложения в отдельных книгах.

В дополнительной к предыдущему изданию 14-й главе описывается заключительный этап проектирования систем – этап создания и развертывания исполняемых приложений.

Немного о том, как выбирался материал, и о стиле его оформления. Вначале приводятся краткие сведения об аппаратных и/или программных компонентах систем, инструментах, необходимых для их разработки. Затем на простых примерах подробно рассматривается процесс разработки и отладки типовых структур, после чего приводится описание средств и результатов испытаний созданных приложений.

В каждом примере раскрываются только те свойства используемых компонентов и те особенности их применения, которые необходимы для решения

соответствующей задачи. В последующих примерах представление о доступных свойствах тех же компонентов расширяется. При этом описание техники программирования сводится не к абстрактному перечислению компонентов, свойств и методов, пусть даже упорядоченному, а к описанию практических приемов решения конкретных задач. Такой формат не предполагает изложения всей информации по каждой из рассматриваемых тем и всех вариантов решения той или иной задачи (это превратило бы книгу в сборник статей справочной системы LabVIEW Help и во много раз увеличило бы ее объем).

Ознакомившись с базовыми понятиями и разобравшись с тем, как реализован пример, читатель, при необходимости, сможет самостоятельно узнать об иных способах и средствах решения задачи. Опыт преподавания профильных дисциплин студентам Новосибирского технического университета, слушателям курсов повышения квалификации, проводимых для сотрудников научных и образовательных организаций, а также для специалистов промышленных предприятий, подтверждает эффективность подобной методики обучения технологиям проектирования [В-2].

Предложение подготовить второе издание книги поступило в 2015 году. Но работу над ним пришлось отложить, т.к. к тридцатилетию LabVIEW (в 2016 г.) ожидался выпуск более совершенной системы нового поколения LabVIEW NXG. Это не означало свертывания проекта LabVIEW предыдущего поколения. Корпорация National Instruments продолжила его развивать, и те, кто приобретал LabVIEW, получали LabVIEW NXG бесплатно, что должно было способствовать ускорению внедрения этой системы. Но в 2016 году в составе первой версии LabVIEW NXG модуль LabVIEW FPGA еще не был включен, а его расширение в последующие годы отставало от продолжающего развиваться модуля LabVIEW FPGA предыдущего поколения. Поэтому, чтобы не терять время, было решено бóльшую часть второго издания книги посвятить уже привычной, завоевавшей широкую популярность редакции системы LabVIEW, дополняя соответствующие разделы информацией, касающейся особенностей LabVIEW NXG.

Системы LabVIEW и LabVIEW NXG очень близки по своим функциональным возможностям и производительности, а усовершенствования в LabVIEW NXG, судя по всему, не столь принципиальны, чтобы тратить ресурсы на поддержку и развитие двух очень сложных систем. И в 2021 году корпорация National Instruments объявила о завершении работы над проектом LabVIEW NXG и прекращении расширенной поддержки всех пяти версий в 2022 году [В-3]. Мы уверены, что и корпорация, и многочисленные пользователи ее продукции от этого только выиграют – LabVIEW и все ее модули теперь будут развиваться ускоренными темпами.

От начала работы над рукописью до издания книги прошло много времени. Многим читателям уже доступна более актуальная версия LabVIEW, чем та, на которую ссылается автор. Но это совершенно не важно – главные принципы, объекты и инструменты LabVIEW, без сомнения, остались неизменными. И это является еще одним свидетельством прочности фундамента, на котором выстроена система LabVIEW.

Представляется, что информации, предлагаемой читателю этой книги, будет достаточно для ускоренного освоения перспективной технологией про-

ектирования реконфигурируемых информационно-измерительных и управляющих систем. Книга может быть рекомендована инженерам, начинающим применять LabVIEW и особенно LabVIEW FPGA в своей практической деятельности, а в качестве учебного пособия – студентам соответствующих специальностей.

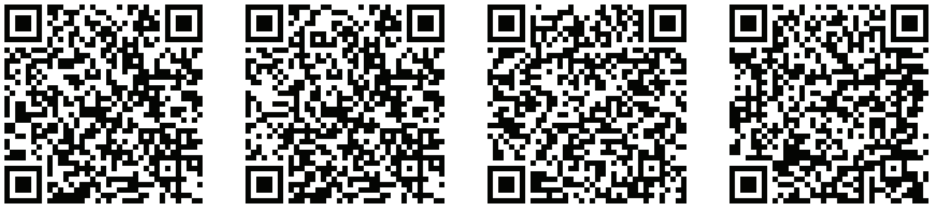
*Ефим Давыдович Баран,*
*преподаватель кафедры систем сбора и обработки данных*
*Новосибирского государственного технического университета,*
*руководитель авторизованного регионального учебного центра*
*«Центр технологий National Instruments»*
*г. Новосибирск, Россия*

# Предисловие А.Ю. Романова

**Дорогие друзья!**

Книга, которую вы держите в руках, продолжает серию книг «Книжная полка истового инженера», которую выпускает издательство «ДМК Пресс» совместно с Московским институтом электроники и математики им. А. Н. Тихонова НИУ ВШЭ при поддержке группы компаний YADRO (yadro.com).

Если вы интересуетесь цифровой электроникой, разработкой на ПЛИС, проектированием на языках описания аппаратуры Verilog или VHDL, то вы, скорее всего, уже знакомы с книгами по цифровому синтезу, такими как: Д. Харрис и С.Л. Харрис «Цифровая схемотехника и архитектура компьютера», «Цифровая схемотехника и архитектура компьютера: RISC-V», «Цифровая схемотехника и архитектура компьютера. Дополнение по архитектуре ARM»; «Цифровой синтез: практический курс» (под ред. А.Ю. Романова и Ю.В. Панчула), Ф. Бруно «Программирование FPGA для начинающих» и др. (воспользуйтесь QR-кодами, чтобы узнать об упомянутых книгах).



Эта книга расширяет тематику и раскрывает особенности разработки на FPGA в среде проектирования от National Instruments LabVIEW, а именно в ее расширении LabVIEW FPGA.

## О чем эта книга?

В **первой главе** кратко рассказывается о ПЛИС и их разновидностях. Во **второй главе** дается краткий экскурс в устройства ввода-вывода, используемые в продукции National Instruments. **Третья глава** посвящена виртуальным измерительным приборам, создаваемым в LabVIEW. В **четвертой и пятой главах** дается общее представление об организации среды проектирования LabVIEW и принципах разработки в этой среде. В **шестой главе** рассказывается, для чего нужны реконфигурируемые системы, а в **седьмой главе** – какие средства LabVIEW предназначены для разработки реконфигурируемых систем. В **восьмой главе** описан типичный маршрут разработки реконфигурируемых систем. **Девятая глава** посвящена описанию, как управлять FPGA VI и создавать Host VI. В **десятой главе** на примерах показано, как можно обрабатывать

данные в LabVIEW FPGA. В **одиннадцатой главе** – как тестировать разработанные приложения, как разрабатывать стандартные интерфейсы для связи с периферийными устройствами. **Двенадцатая глава** посвящена модулям R-серии, а также вопросам измерения неэлектрических величин. В **тринадцатой главе** приведен пример разработки встраиваемой системы на платформе cRIO. А в **четырнадцатой главе** кратко рассказывается, как создать сборку проекта и инсталлятор для развертывания приложений на пользовательских компьютерах. В завершающей п**ятнадцатой главе** рассказывается, как проекты, разработанные в LabVIEW FPGA, перенести в другие САПР-ы и подготовить для компиляции под ПЛИС других производителей.

## Что послужило причиной появления этой книги?

Дело в том, что на русском языке книг по LabVIEW FPGA нет. Существовала книга Е.Д. Барана **«LabVIEW FPGA. Реконфигурируемые измерительные и управляющие системы»** 2009 г., но она была выпущена небольшим тиражом и за более чем 10 лет уже порядком устарела. В то же время, LabVIEW широко преподается в российских вузах, и материалов по данной тематике не хватает. У нас в МИЭМ есть отдельная лаборатория, оборудованная продукцией National Instruments, в т.ч. myRIO на основе FPGA. На них студенты изучают основы цифровой электроники, прежде чем попадают в мою лабораторию Систем автоматизированного проектирования (САПР), где уже знакомятся с проектированием на отладочных платах с ПЛИС от компаний Altera (Intel FPGA)/ Xilinx и их САПР-ами. Но, на самом деле, я всегда считал такое несколько подчиненное положение LabVIEW –несправедливым. Потому что LabVIEW – это очень мощная и разносторонняя САПР, обладающая большим функционалом и возможностями. Главный ее недостаток – это жесткая привязка к продуктам National Instruments. Но вот как раз в случае с проектами на ПЛИС в LabVIEW FPGA в последние несколько лет появилась надстройка IP Export Utility. Этот инструментарий превращает LabVIEW в мощный инструмент высокоуровневого синтеза (HLS – High Level Synthesis) наподобие MATLAB, но только со своим языком графического проектирования и возможностью аппаратной отладки проектов и создания цифровых панелей управления, после чего проекты могут быть экспортированы на нужную разработчику платформу или даже скомпилированы под ASIC с помощью открытого маршрута проектирования.

Поэтому, когда ко мне обратились из издательства «ДМК Пресс» с предложением выступить редактором нового переиздания книги Е.Д. Барана по LabVIEW FPGA, я отложил работу над другими книгами и с огромным энтузиазмом взялся за работу. Оказалось, что не все так просто: книга претерпела глубокую переработку, получив тысячи правок и дополнений, обзавелась новой главой (как раз про LabVIEW FPGA IP Export Utility) стала современней и более дружественной для читателя. И теперь, по прошествии 9 месяцев работы, я, теперь уже в статусе второго автора, рад представить ее на ваш суд.

**Что почитать еще?**

Если Вы осилите эту книгу, то я советую также ознакомиться с книгой Terry Stratoudakis «Introduction to LabVIEW FPGA for RF, Radar, and Electronic Warfare Applications» [B-4]. Она сама по себе интересна, но еще в ней есть обширный спи-

сок литературы, который собран в репозитарии: *https://github.com/LVFPGABOOK* [В-5]. Также обратите внимание на список литературы, который есть у этой книги. Там собраны практически все мануалы и инструкции, а также другие книги преимущественно в открытом доступе, которые Вам могут пригодиться.

## Насколько сложно сделать хорошую книгу?

Об этом я рассказываю в своем выступлении на IV конференции FPGA разработчиков FPGA Systems 2023.1: *https://www.youtube.com/watch?v=7K5l_rIL8-o*

Надеюсь, эта книга тоже вышла хорошей и **будет интересна широкому кругу читателей** от студентов технических вузов до разработчиков научного и промышленного оборудования, которые стремятся расширить свой кругозор и познакомиться с новыми САПР. Для знакомства с ней достаточно знать основы программирования, иметь хотя бы некоторое понимание, что такое ПЛИС и для чего они нужны, а также стремление развиваться.

**В добрый путь!**

*Александр Юрьевич Романов,*
*к. т. н., доцент ДКИ МИЭМ НИУ ВШЭ,*
*преподаватель курсов «Проектирование систем на кристалле»,*
*«Системное проектирование цифровых устройств»*
*и «Системы искусственного интеллекта»*
*г. Москва, Россия*

# Благодарности

Прежде всего, благодарим читателей первого издания книги, приславших свои отзывы и замечания, мы постарались их учесть.

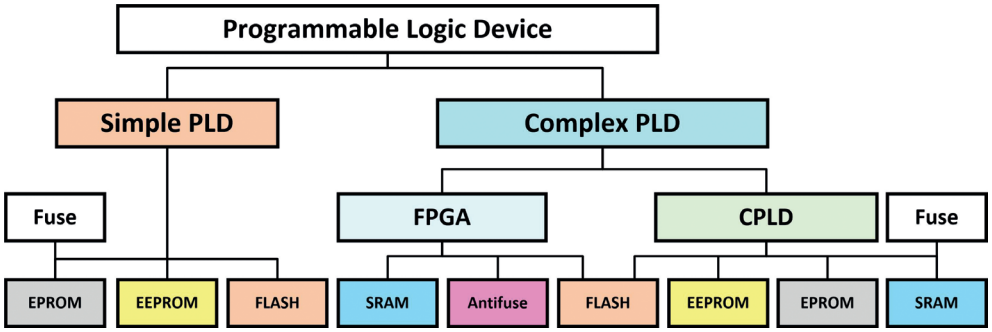
Особая благодарность всем сотрудникам Российского представительства корпорации National Instruments, оказывавшим содействие в практическом внедрении технологий реконфигурируемого ввода-вывода, инициировавшим работы по популяризации этого перспективного направления путем издания серии книг о LabVIEW, которую пополнит и настоящая книга.

Рукопись книги, которую вы сейчас читаете, подготовили и прочитали два автора, научный редактор, один корректор и еще несколько рецензентов, но, безусловно, книга не идеальна. Мы будем очень признательны тем внимательным читателям, которые обнаружат в данном издании какие-либо ошибки или опечатки и сообщат о них авторам: **baran@corp.nstu.ru**, **a.romanov@hse.ru** или **dmkpress@gmail.com** (книги постоянно перепечатываются, и в каждом новом тираже все найденные ошибки и недочеты исправляются).

Авторы благодарны всем, кто помогал готовить эту книгу. Вот некоторые из тех, кто помогал сделать эту книгу лучше: **Максим Колпаков**, **Оксана Батонова**, **Денис Палуха**, **Максим Ельцов**.


# Глава 1

# Программируемые логические интегральные схемы




В эту главу входят следующие параграфы:

С развитием микроэлектронных технологий и увеличением степени интеграции элементов на кристалле все очевиднее становилось противоречие между потребностями создавать компактные специализированные устройства из небольшого количества компонентов и нерентабельностью расширения номенклатуры выпускаемых массовым тиражом типовых логических микросхем и функциональных блоков. Действительно, уже с конца 60-х годов прошлого века стало возможным на одном кристалле размещать десятки и сотни логических вентилей, но чем выше становилась плотность размещения компонентов на новом кристалле, тем больше возрастала его специализация и тем меньше становилась относительная доля объема его выпуска.

Появление универсальных логических устройств, микропроцессоров, конечная функция которых определялась загружаемой в них программой, казалось, сблизили возможности изготовителей микросхем, владеющих высокотехнологичными средствами производства, с одной стороны, и потребности и возможности разработчиков прикладных устройств и систем – с другой. Но инженеры по-прежнему вынуждены были для решения своих задач использовать компоненты высокой (по меркам того времени) степени интеграции. Процессор был основным функциональным блоком, а десятки логических элементов малой степени интеграции использовались для того, чтобы состыковать между собой микросхемы разных классов и типов, а также, чтобы реализовать на аппаратном уровне специфические для каждого конкретного случая узлы.

Например, контроллер выполнялся на основе 4- или 8-разрядного микропроцессора. А несколько микросхем памяти, содержащих тысячи логических вентилей, и до десятка схем более низкой степени интеграции (дешифраторов, регистров, логических элементов), необходимых для реализации системной шины, каналов ввода-вывода и т.п. Для изготовления такого контроллера требовалась печатная плата с площадью, достаточной для установки микросхем повышенной степени интеграции (микропроцессор, память), размещения большого количества микросхем малой степени интеграции и всех необходимых печатных проводников. Сборка относительно несложных логических компонентов контроллера проводилась с помощью паяльника, в то время как существенно более сложные микропроцессор и блоки памяти поступали в виде готовых микросхем. При этом стоимость монтажа оказывалась непропорциональной сложности устройства, а его надежность определялась надежностью элементов малой степени интеграции, количеством их выводов и соединительных проводников на печатной плате.

Таким образом, разработчики цифровой аппаратуры, получив возможность использовать достаточно мощные и универсальные процессорные компоненты, вынуждены были по-прежнему применять простые и тоже универсальные элементарные логические схемы, собирая их в специализированные блоки, без которых невозможно было спроектировать и изготовить устройство, решающее задачу измерений и обработки информации, управления или тестирования.

Назревала необходимость усовершенствования технологии проектирования и производства конечных изделий, для того чтобы приблизить ее к тех-

нологии проектирования и производства микроэлектронных компонентов. Массовое производство микросхем, где связи между элементами в кристалле создавались с помощью масок, было передовым. Но использовать эту технологию в многочисленных лабораториях разработчиков цифровых устройств невозможно, т.к. изготовление масок – весьма трудоемкий и дорогой процесс. Необходимо было придумать более доступный механизм произвольного соединения элементов кристалла между собой и с внешними выводами микросхемы, а разместить на кристалле необходимый набор типовых логических элементов (десятков или сотен) к тому времени проблемой уже не являлось.

Другими словами, разработчикам требовалась технология, позволяющая более простыми и в то же время более эффективными средствами **создавать собственную микросхему**, которая могла бы заменить горсть универсальных микросхем малой степени интеграции, объединяемых в специализированный блок. Такая возможность была реализована в программируемых логических интегральных структурах – **ПЛИС** (**PLD – Programmable Logic Device**). На кристалле ПЛИС размещались простые логические вентили различных типов и объединяющие их регулярные шины проводников. Подобная микросхема представляла собой некоторую универсальную заготовку, в которой разработчик доступными средствами мог удалить ненужные связи между вентилями, изменяя тем самым функцию ПЛИС и, по существу, создавая уникальное, необходимое только ему специализированное логическое устройство, сохраняя почти все преимущества его интегрального выполнения.

## 1.1. Простые программируемые логические устройства

Первые кристаллы, структуру и функции которых мог определять пользователь, – программируемые постоянные запоминающие устройства – **ППЗУ** (**PROM– Programmable Read Only Memory**). ППЗУ состоит из двух матриц логических элементов – матрицы элементов «И» и матрицы элементов «ИЛИ».

Входы элементов «И» соединяются с внешними входами микросхемы через повторитель или инвертор, а входы элементов «ИЛИ» – с выходами элементов «И» [1-1]. Таким образом, на выходах элементов «ИЛИ», являющихся внешними выходами микросхемы, формируется логическая сумма произведений входных переменных.

В показанной на рис. 1-1 схеме переменные, подаваемые на входы адреса, в дешифраторах строк образуют все возможные конъюнкции – функции «И» от всех входных переменных. Объединением выходов конъюнкторов в столбцах матрицы реализуются логические функции «ИЛИ» – отключением определенных строк от столбцов определяются выходные функции ППЗУ.

От изготовителя ППЗУ поступало «чистым» – были подключены все входы всех элементов «И» и «ИЛИ», но пользователь мог удалять ненужные соединения, адресно разрушая их в специальном режиме. В качестве разрушаемых соединений (перемычек) использовались выжигаемые током проводники или пробиваемые *p-n*-переходы (диоды), сохраняемые же связи и являлись запоминающими элементами и определяли логическую функцию для каждого выхода

ППЗУ. Следует отметить, что использовались и иные способы программирования ППЗУ – не удалением ненужных из предварительно созданных межсоединений, а наоборот – созданием нужных соединений. Но подобные технологические нюансы для разработчика прикладных устройств были непринципиальны.

Различают ППЗУ двух основных типов:

- микросхемы, получившие название **PROM**, в которых можно изменять только подключения входов элементов «ИЛИ». Матрица связей элементов «И» в них фиксирована (пример на рис. 1-1);
- микросхемы типа **PAL/GAL** (**Programmable/Generic Array Logic**). В них программируются связи элементов «И», а соединения элементов «ИЛИ» изменению не подлежат.

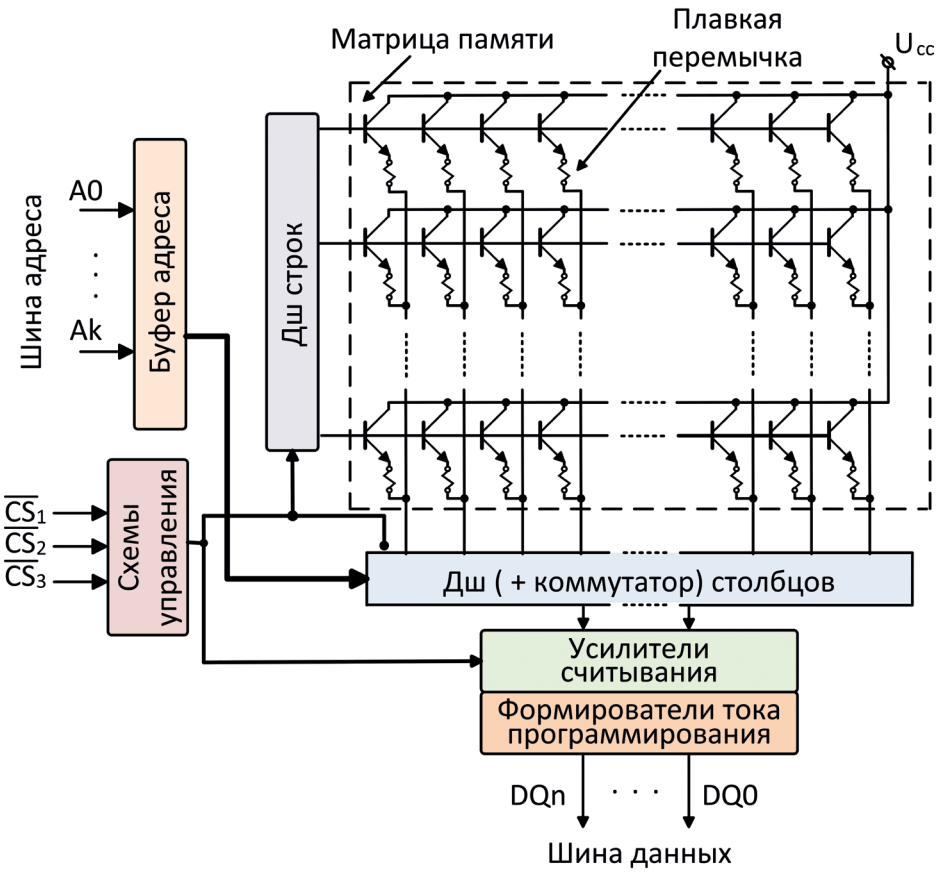



**Рис. 1-1.** Структура ППЗУ

Набор из $N$ произвольных логических функций от $k$ входных переменных может быть реализован в одной микросхеме ППЗУ, содержащей $N$ $k$-входовых логических элементов. Для этого достаточно записать эти функции в совершенной дизъюнктивной нормальной форме и задать их таблицы истинности в программаторе.

В программируемых ПЗУ площадь кристалла расходуется неэкономно – для всех возможных конъюнкций и дизъюнкций входных и промежуточных переменных должны быть зарезервированы входы элементов «И» и «ИЛИ», значительная часть которых при программировании отключается.

Более рационально логические функции реализуются на базе **ПЛМ** – программируемых логических матриц (**PLA – Programmable Logic Array**) [1-2, 1-3]. В ПЛМ могут быть запрограммированы обе матрицы логических элементов – «И» и «ИЛИ», при этом количество входов конъюнкторов и дизъюнкторов может быть уменьшено без существенных потерь сложности реализуемых логических функций (рис. 1-2).

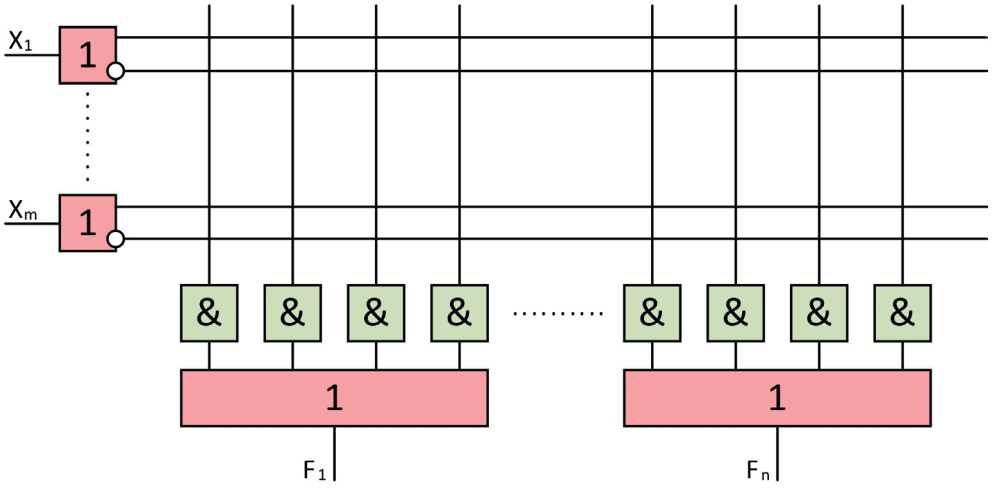



**Рис. 1-2.** Структура ПЛМ

ПЛИС, рассмотренные выше (PROM, PAL, GAL, PLA) дали разработчикам определенную свободу в реализации проектов, обеспечили возможность уменьшения количества компонентов, повышения надежности изделий и снижения их стоимости. Вычислительные устройства стали создавать не только на основе микропроцессоров с жесткой системой команд, но и на базе микропроцессорных секций с микропрограммным управлением, причем набор команд уже определялся самим разработчиком, который записывал микропрограммы в ПЛИС.

Таким же образом проектировались специализированные комбинационные устройства – преобразователи кодов, шифраторы и дешифраторы, а ПЛИС совместно с элементами памяти (триггерами, регистрами, счетчиками) стали применять для разработки быстродействующих устройств микропрограммного управления различными объектами.

Дальнейшее развитие технологии ПЛИС позволило включить в них элементы памяти. Это перевело разработку на качественно новый уровень, когда стало возможно создавать не только простые комбинационные устройства, а и законченные последовательностные цифровые автоматы, полностью реализующие требуемую диаграмму состояний-переходов без использования дополнительных элементов малой и средней степени интеграции. Достаточно наглядное представление о возможностях, предоставляемых ПЛИС с памя-

тью, дает приведенная на рис. 1-3 схема популярной микросхемы **PAL22V10**, клоны которой выпускают до сих пор многие компании [1-4, 1-5].

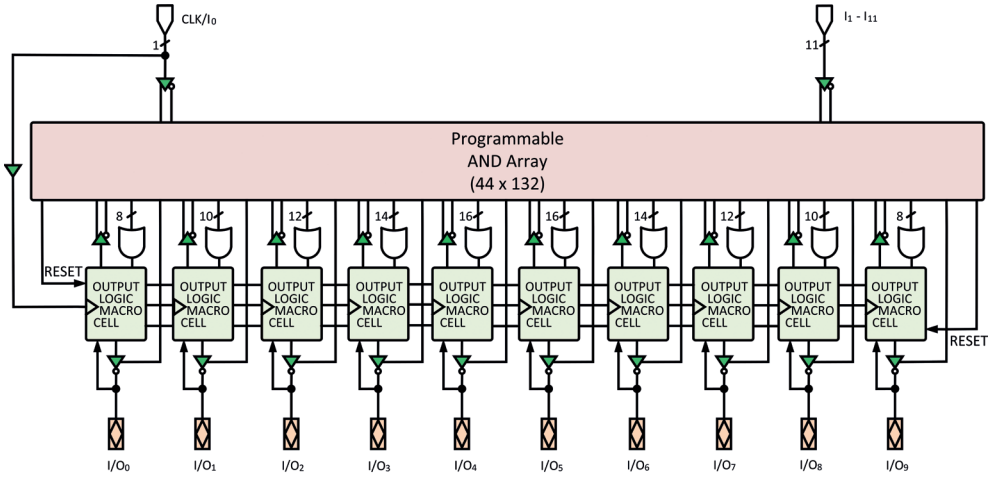
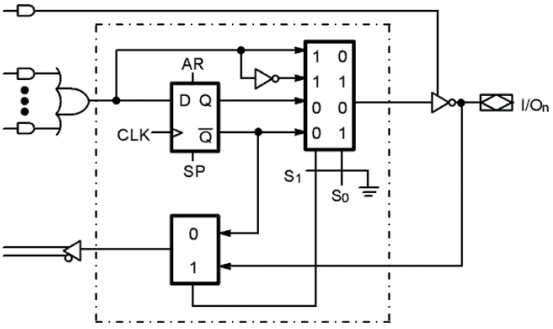



**Рис. 1-3.** Структура ПЛИС с элементами памяти

Эта микросхема, относящаяся к классу PAL, содержит программируемую матрицу элементов «И» (**Programmable AND Array**), на входы которых $\mathbf{I_1 \div I_{11}}$ поданы 11 переменных (или их инверсии) с внешних входов ПЛИС и 10 внутренних переменных (или их инверсии), ассоциируемых с внешними входами/выходами ПЛИС. Выходы элементов «И» объединяются в линейке из 10 непрограммируемых элементов «ИЛИ» с фиксированным количеством входов – от 8 до 16. Сигналами с выходов элементов «ИЛИ» управляются выходные макроячейки (**OUTPUT LOGIC MACROCELL**), каждая из которых содержит тактируемый D-триггер с входами сброса и установки.

Макроячейка может быть сконфигурирована для работы в одном из 4 режимов, выбираемых с помощью программируемых перемычек **S0** и **S1** (рис. 1-4). Этими перемычками основной мультиплексор ячейки настраивается таким образом, что на выходной контакт ПЛИС поступает сигнал, реализуемый комбинационной логикой (матрицами «И» и «ИЛИ»), или тот же сигнал, зафиксированный D-триггером по приходу синхроимпульса **CLK**. Выходной сигнал ($\mathbf{I/O_n}$) с помощью дополнительного мультиплексора может быть возвращен во входную комбинационную схему в качестве сигнала обратной связи для текущей макроячейки или в качестве дополнительной логической переменной – для остальных макроячеек.

Любой выход может быть переведен в высокоимпедансное состояние и использован как вход или двунаправленный вход/выход ПЛИС. Перечень полезных свойств этой микросхемы расширяют возможности сброса элементов памяти в исходное состояние по включению питания, загрузки в регистр произвольного кода через внешние контакты (что улучшает тестопригодность создаваемых на основе этой ПЛИС устройств), а также возможность установки защиты от несанкционированного копирования внутренней структуры ПЛИС – интеллектуальной собственности (**Intellectual Property**) разработчика.




| S0 | S1 | Выходной сигнал |
|---|---|---|
| 0 | 0 | С выхода триггера, инверсный |
| 0 | 1 | С выхода триггера, прямой |
| 1 | 0 | Комбинационный, инверсный |
| 1 | 1 | Комбинационный, прямой |

**Рис. 1-4.** Макроячейка PAL22V10

Микросхемы типа PAL с элементами памяти, вместе с ПЛИС комбинационного типа PROM, PAL, GAL, в последующем были отнесены к классу простых программируемых логических устройств – **SPLD** (**Simple Programmable Logic Device**).

Отличительные признаки SPLD:

- каждая макроячейка имеет свой внешний выход и свой собственный триггер;
- в микросхеме SPLD реализовано не менее 2 макроячеек;
- обычно все макроячейки одинаковые;
- логическая функция макроячейки описывается одним логическим термом;
- логическая функция макроячейки реализуется матрицами элементов «И» и «ИЛИ».

К достоинствам SPLD обычно относят простоту проектирования специализированных устройств, постоянное и, как правило, одинаковое время прохождения сигналов с входов на выходы, возможность замены одной или несколькими микросхемами SPLD достаточно большого количества типовых микросхем малой и средней степени интеграции.

Основные недостатки SPLD – неэффективное использование ресурсов (логических вентилей) и, как следствие, проблематичность создания на их основе сложных цифровых устройств.

## 1.2. Технологии программирования ПЛИС

Последовательное увеличение степени интеграции микросхем и функциональной насыщенности реализуемых в кристалле ячеек, которые мог сконфигурировать разработчик, сопровождалось совершенствованием технологии программирования ПЛИС. Поэтому прежде чем продолжить обзор архитектур ПЛИС, в этом разделе будут рассмотрены технологии программирования, которые развивались параллельно с технологиями производства микросхем.

В первых семействах SPLD (PROM, PAL / GAL, PLA), выпускаемых на основе биполярных полупроводниковых элементов, однажды реализованная пользователем функция не могла быть изменена, структура соединений логических вентилей в ПЛИС записывалась однократно [1-5, 1-6]. Из-за этого, если в процессе отладки вновь разработанного устройства обнаруживались ошибки, то

для их исправления необходимо было , запрограммированную («прошитую)» с ошибками ПЛИС извлечь из печатной платы и заменить на исправную. То есть по-прежнему разработчик был вынужден использовать паяльник (хоть и в меньшей степени, чем раньше) или, если это допускалось с точки зрения надежности изделия, монтировать микросхему ПЛИС на печатной плате в гнезде (сокете), позволяющем при необходимости оперативно ее заменить.

Совершенно новое качество проектирования было достигнуто после перехода на кристаллы с униполярными полупроводниковыми компонентами, в которых вместо **однократно** разрушаемых (создаваемых) связей-перемычек стали применять запоминающие элементы на МОП-транзисторах с изолированным затвором. Заряд, создаваемый на затворе МОП-транзистора при программировании таких структур, сохранялся годами и не исчезал при отключении питания. Заряд этот можно снять, восстановив тем самым исходное состояние ячейки памяти, а при необходимости вновь записать – т.е. появилась возможность **неоднократного** перепрограммирования ПЛИС. Технологии производства таких изделий разработаны в 1971г. фирмой **Intel**, а в 1974г. – фирмой **Toshiba** и реализованы в **перепрограммируемых ПЗУ** (**ППЗУ**).

Запись информации в РППЗУ производится в специальном режиме импульсами напряжения повышенной амплитуды, а стирание осуществляется двумя способами:

- ультрафиолетовым излучением. Такие микросхемы называются **UV-EPROM** (**Ultraviolet Erasable PROM**), УФ РППЗУ;
- электрическим напряжением противоположной полярности – микросхемы **EPROM** (**Erasable Programmable ROM**). В отечественной литературе их обычно называют ЭСППЗУ (электрически стираемые программируемые ПЗУ).

Для стирания информации ультрафиолетовым излучением в корпусах микросхем РППЗУ создается закрытое кварцевым стеклом окошко, через которое облучается кристалл.

Теперь инженеру не нужно было иметь запас однократно программируемых устройств, чтобы заменить неправильно спроектированную микросхему. Вновь разработанное прикладное изделие можно было запускать в производство, комплектуя его набором универсальных типовых элементов и достаточно дешевыми микросхемами «собственного изготовления» – специфическими только для данного изделия и запрограммированными самим разработчиком устройства. При этом по-прежнему необходимо было предусматривать возможность извлечения микросхемы РППЗУ из печатной платы для стирания и последующего перепрограммирования, а самое главное – надо было оснастить рабочее место специальным программатором и устройством для стирания ультрафиолетовым излучением.

Следующий шаг в развитии программируемых пользователем компонентов был сделан, когда технология позволила реализовать стирание/перезапись информации внутри кристалла без использования внешнего специального оборудования. Подобные микросхемы – **Electronically Erasable PROM**– были разработаны фирмой **Intel** в 1979 г. и, по существу, исключили из процесса разработки, изготовления и отладки макетных образцов аппаратуры демонтаж исправных, но неправильно запрограммированных инженером компонентов.

Наилучшими качествами в семействе микросхем класса Electronically Erasable PROM обладает энергонезависимая память типа **Flash**, предложенная в 1984 г. фирмой **Toshiba** (а с 1988 г. развиваемая фирмой **Intel** и рядом других фирм), которая широко применяется сейчас в твердотельных накопителях информации большой емкости. Flash-память характеризуется большим количеством циклов перезаписи (до $10^6$ и более), высоким быстродействием. Некоторые разновидности такой памяти допускают возможность стирания/модификации содержимого произвольной ячейки. Благодаря этим качествам, высокой технологичности в производстве, а, следовательно, и более низкой стоимости, Flash-память практически вытеснила остальные разновидности программируемой энергонезависимой памяти, используемых при разработке цифровых устройств и систем.

## 1.3. Пограничное сканирование и JTAG-интерфейс

Упомянем еще о некоторых задачах, которые приходилось решать в процессе развития микроэлектроники и системотехники. С увеличением степени интеграции ПЛИС, как, собственно, и любых других интегральных схем, обострялась и становилась все более важной проблема тестирования компонентов и систем, создаваемых на их основе. Радикально усложнились отладка и верификация цифровых автоматов при массовом использовании ПЛИС в новых изделиях и массовом производстве систем, содержащих микросхемы высокой степени интеграции. Тестирование подобных компонентов и устройств требует значительного времени как на подготовку тестов, так и на их проведение. Необходимо также специализированное дорогостоящее оборудование и программное обеспечение, сопоставимое по сложности и трудоемкости разработки с прикладным (целевым) программным обеспечением.

Эффективным решением проблемы тестирования стала взятая на вооружение разработчиками концепция проектирования **тестопригодных** или легко тестируемых компонентов и систем (**Design For Testability – DFT**). Из нескольких опробованных на практике подходов к созданию тестопригодных устройств наиболее универсальным и экономным был признан подход, основанный на методе тестирования, получившем название **метода пограничного сканирования** (**Boundary-Scan**). Суть метода заключается в том, что в режиме тестирования обеспечиваются возможности:

- изоляции любого компонента от всех остальных компонентов системы;
- доступа к любому изолированному компоненту для его автономного тестирования;
- выполнения отдельной процедуры проверки межсоединений компонентов.

При этом в основном режиме работы и компоненты, и система в целом по-прежнему функционируют по прямому назначению.

Концепция пограничного сканирования в 1990 г. была закреплена международным стандартом **IEEE-1149.1 – IEEE Standard Test Access Port and Boundary-Scan Architecture**, определяющим архитектуру устройств с пограничным сканированием, а также структуру и функционирование специального тестового порта (**Test Access Port – TAP-порт**) [1-7]. Этот порт обычно на-

зывают **JTAG**-интерфейсом по наименованию объединенной рабочей группы по тестированию **Joint Test Action Group**.

Стандарт стал руководством к действию подавляющего большинства производителей электронных компонентов, разработчиков и изготовителей цифровых систем. Теперь в каждую микросхему, разработанную в соответствии со стандартом IEEE-1149.1, встроены TAP-порт и совокупность **JTAG-ячеек**, по одной на каждый из выводов микросхемы. Они образуют регистр пограничного сканирования. JTAG-ячейка позволяет:

- подключить ядро микросхемы (**Internal Core Logic**) к внешним выводам для основного режима работы микросхемы в составе устройства;
- подключить к внешним выводам микросхемы регистр пограничного сканирования и отключить от них ядро микросхемы для тестирования соединений между микросхемами с помощью внешнего тестера;
- отключить внешние выводы микросхемы от ядра и подключить к ядру регистр пограничного сканирования для тестирования ядра.

Таким образом, регистр пограничного сканирования может работать в нескольких режимах. В основном режиме работы микросхемы регистр «прозрачен» для внешних сигналов, а в тестовом – ядро микросхемы изолировано от внешней среды. При этом на входы ядра тестовые сигналы поступают с выходов регистра пограничного сканирования, а выходные сигналы ядра могут быть записаны в этот же регистр. В тестовом режиме загрузка и считывание данных в регистр осуществляются последовательно через TAP-порт (рис. 1-5).

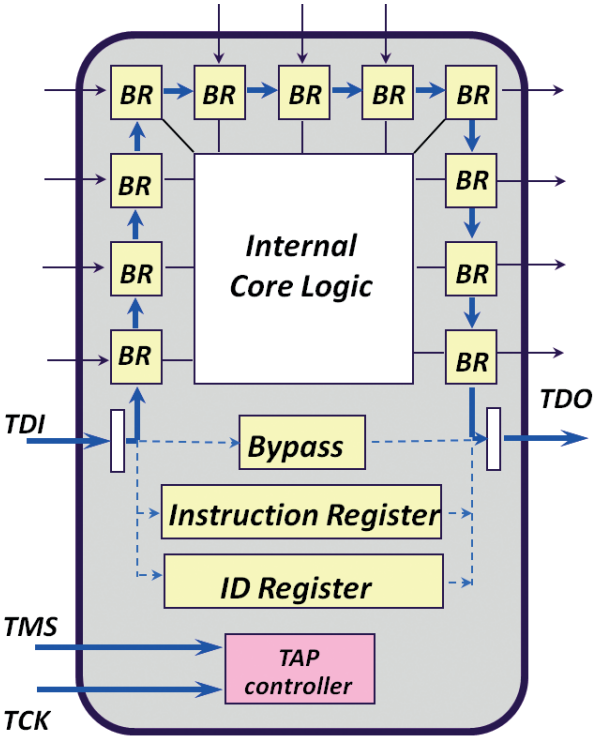



**Рис. 1-5.** TAP-порт

TAP-порт содержит 4 обязательные и одну дополнительную линию для передачи сигналов:

- **TDI** – **Test Data Input** – вход последовательного потока данных. В зависимости от состояния интерфейса в микросхему передаются либо команды для управления JTAG-ячейками, либо тестовые данные;
- **TDO** – **Test Data Output** – выход последовательного потока данных. Через эту линию в тестер передаются результаты выполнения команд и состояния JTAG-ячеек;
- **TCK** – **Test Port Clock** – последовательность тактовых импульсов для синхронизации последовательного обмена. Этот сигнал вырабатывается тестером и необязательно должен быть фиксированной частоты. Максимальная частота следования импульсов TCK для многих микросхем находится в пределах 5÷10 МГц. Стандарт не ограничивает частоту тактирования снизу;
- **TMS** – **Test Mode Select** – этот сигнал управляет режимами JTAG-интерфейса;
- **TRST*** – **Test Port Reset** – сброс узлов тестовой логики (сигнал необязателен). Этот сигнал не заменяет сигнал сброса ядра микросхемы, поскольку JTAG-логика управляет только JTAG-ячейками и не влияет на работу ядра.

Ячейки регистра пограничного сканирования (**Boundary Register – BR**) всех микросхем могут объединяться в один общий последовательный регистр, чтобы обеспечить доступ к любой микросхеме устройства. Для этого в каждой микросхеме предусмотрен однобитовый регистр обхода (**Bypass**). Кроме того, TAP-порт содержит **TAP Controller** и 4-битовый регистр команд **Instruction Register** для управления режимами работы, а также регистр идентификатора микросхемы **ID Register**.

Стандарт IEEE-1149.1 определяет только 4 обязательные команды, остальные 12 возможных кодов команд зарезервированы для дальнейших расширений, и их действие определяется изготовителем микросхем.

В целом реализация метода пограничного сканирования предельно проста, как с точки зрения необходимых аппаратных ресурсов, так и с точки зрения программирования. В микросхеме может потребоваться не более 4÷5 дополнительных выводов для TAP-порта и от нескольких десятков до нескольких тысяч дополнительных логических элементов и триггеров, что обычно составляет не более единиц процентов от общего количества логических вентилей на кристалле.

Исключительно важно, что тестирование методом пограничного сканирования не требует сложного и дорогого внешнего тестера [1-8] – его функции может выполнять персональный компьютер с адаптером, позволяющим использовать обычный последовательный или параллельный порт компьютера в качестве TAP-порта (рис. 1-6).

Благодаря стандарту IEEE-1149.1 качественное тестирование стало возможным не только в крупных компаниях, которые смогли заплатить миллионы долларов за современный тестер, но и в любой лаборатории, где тестером становится персональный компьютер со свободным стандартным портом.

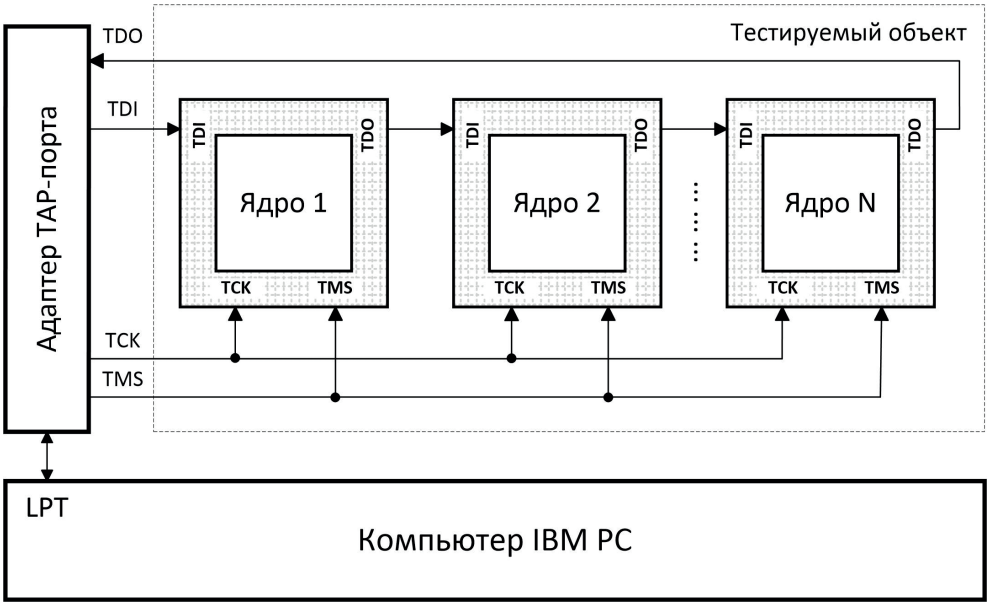



**Рис. 1-6.** Тестирование методом пограничного сканирования

Одновременно существенно упростилась и разработка программ тестирования вследствие регулярности структуры ПЛИС и доступности большого количества внутренних узлов синтезируемых в ПЛИС автоматов. С целью облегчения проектирования тестов и унификации тестового обеспечения были созданы специальные языки программирования высокого уровня – **Boundary Scan Description Language** (**BSDL**, язык описания устройств с пограничным сканированием), **Jam** и др.

В настоящее время JTAG-интерфейс и созданные для работы с TAP-портом языки программирования могут быть использованы **не только для тестирования ПЛИС, но и для их конфигурирования** [1-9]. Собрав схему, подобную изображенной на рис. 1-6, разработчик в считанные минуты может «загрузить» в чистую ПЛИС только что спроектированное устройство, провести его тестирование (прототипирование), а в дальнейшем, при необходимости, оперативно изменять структуру ПЛИС, находясь в одной и той же среде проектирования. Аналогично, если на печатной плате вместе с ПЛИС смонтированы другие микросхемы, поддерживающие стандарт IEEE-1459.1, например, микропроцессоры или запоминающие устройства, то и в них можно загрузить программу, воспользовавшись TAP-портом, осуществить тестирование отдельных компонентов или системы в целом.

Таким образом, концепция пограничного сканирования дала в руки инженеру универсальное средство реконфигурирования аппаратных компонентов, программирования микропроцессорных и иных устройств, а также тестирования. При использовании этих средств стала доступной **технология программирования в системе** (**In-Circuit Programming**) – без паяльника и гнезд под микросхемы, благодаря которой не только упростился процесс проектирова-

ния и производства, но стало доступней и проще и необходимое специальное оборудование. Исчезла необходимость в отдельных устройствах – программаторах, устройствах стирания информации в ПЛИС, сложном и дорогостоящем тестовом оборудовании.

## 1.4. Сложные программируемые логические устройства

Вернемся к обзору архитектур программируемых логических схем. Вслед за простыми, SPLD, появился класс сложных программируемых логических устройств – **Complex Programmable Logic Device** (**CPLD**).

Одной из первых подобные изделия под названием **Classic Programmable Logic Device** (аббревиатура та же) выпустила на рынок компания **Altera**[1] (сейчас, Intel FPGA). В качестве примера на рис. 1-7 показана структурная схема одной из микросхем семейства **CPLD – EP610** [1-10].

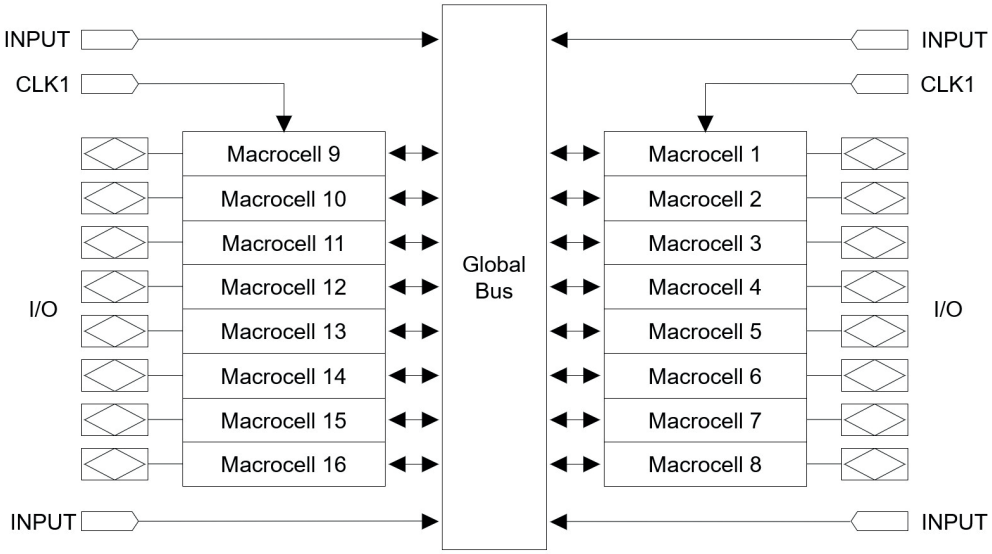



**Рис. 1-7.** Структурная схема Classic PLD Altera EP610

Чем сложные ПЛИС отличаются от простых?

В SPLD логические макроячейки, содержащие триггер, соединялись преимущественно с выходными контактами ПЛИС, причем количество макроячеек было невелико.

Сложные ПЛИС при бóльшем количестве макроячеек (в самой простой CPLD микросхеме типа EP610 их уже 16) обеспечивают возможность практически произвольного соединения макроячеек (**Macrocell**) друг с другом и с внешними выводами микросхемы (**I / O**) и позволяют создавать существенно более

[1] В декабре 2015 г. компания Altera приобретена корпорацией Intel за 16,7 млрд. долларов. В настоящее время она является подразделением Intel и называется Intel FPGA, входит в Programmable Solutions Group (PSG).

сложные последовательностные устройства. Для организации связей внутри ПЛИС используется программируемая матрица (шина) глобальных проводников (**Global Bus**). В ПЛИС EP610, кроме 16 двунаправленных выводов (**I/O**), имеются также 4 дополнительных логических входа (**Input**) и 2 глобальных входа синхронизации (**Clk1** и **Clk2**).

Возможности реализации логических функций в CPLD, в отличие от SPLD, определяются не только связями между макроячейками через разветвленную общую шину коммутации Global Bus, но и связями, создаваемыми внутри каждой макроячейки (рис. 1-8). Элемент памяти макроячейки (**Programmable Register**) может быть сконфигурирован как D, T, SR или JK триггер с индивидуальным управлением установки в «0» или «1».

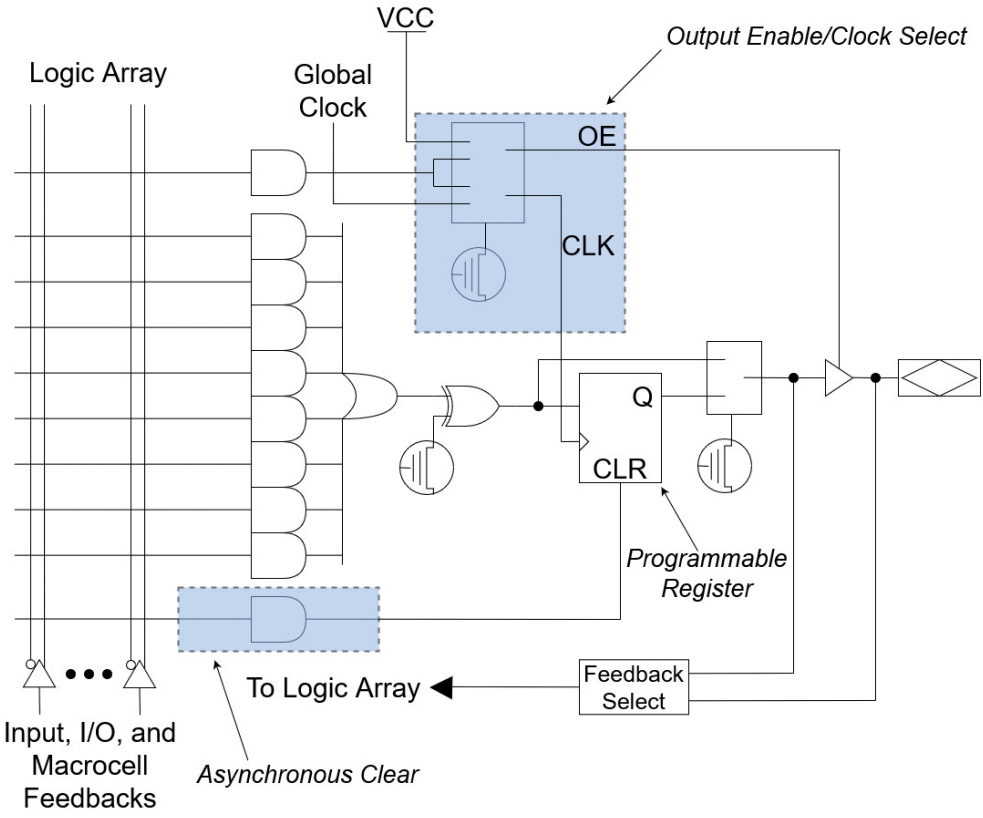



**Рис. 1-8.** Макроячейка EP610

Комбинаторная логическая функция макроячейки образуется программированием линейки элементов «И» на первом уровне при фиксированных соединениях с 8-входовым элементом «ИЛИ» на втором уровне. Возможно программирование цепей синхронизации, входов/выходов, использование сигналов обратной связи и т.п.

В этом первом семействе Classic PLD пока еще отсутствует TAP-порт и внутрисхемное программирование невозможно. Но уже в следующих семействах – **MAX 3000** (5000, 7000, 9000 и др.) компании **Altera**, так называемых «настоящих» сложных программируемых устройств (**Complex PLD**), концепция пограничного сканирования поддерживается и тем самым обеспечивается достижение всех связанных с этой концепцией преимуществ – простоты тестирования и программирования структуры ПЛИС.

Общее представление об особенностях CPLD этих семейств дает рис. 1-9, на котором представлена архитектура самой простой микросхемы MAX 3000A [1-11, 1-12].

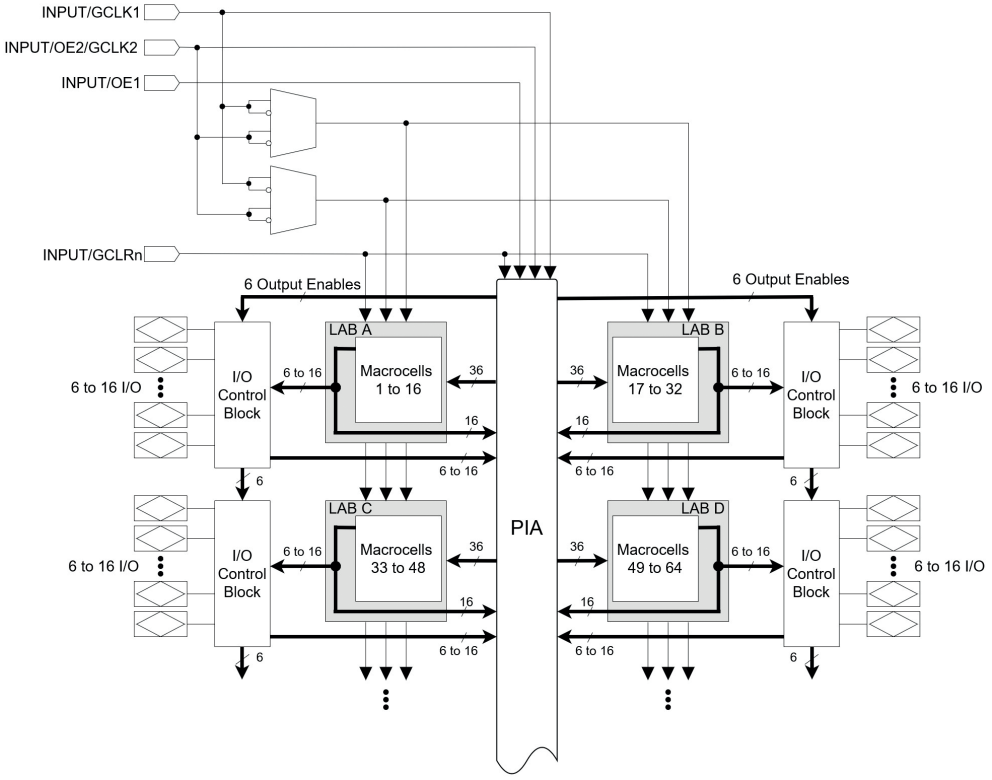



**Рис. 1-9.** Архитектура CPLD MAX 3000A

Основным крупным компонентом MAX 3000 является блок логических матриц – **Logical Array Blocks** (**LAB**), каждый из которых содержит по 16 макроячеек (**Macrocell**), подобных рассмотренным выше. Макроячейки и логические блоки соединяются между собой, с внешними программируемыми контактами ввода-вывода (**I/O**), а также с внешними общими линиями управления и синхронизации (**Input/GClkX**, **Input/OEX**, **Input/GClrn**) с помощью глобальной программируемой матрицы межсоединений – **Programmable Interconnect Array** (**PIA**) (рис. 1-10).

Эта матрица представляет собой набор переключателей, управляемых ячейкой памяти с электрическим стиранием (EEPROM), которые позволяют подсо-

единить любой из 36 входов логического блока (LAB) к источнику любого из сигналов (внешнего, внутреннего обратной связи и т.д.), подключенного к вертикальным линиям матрицы.

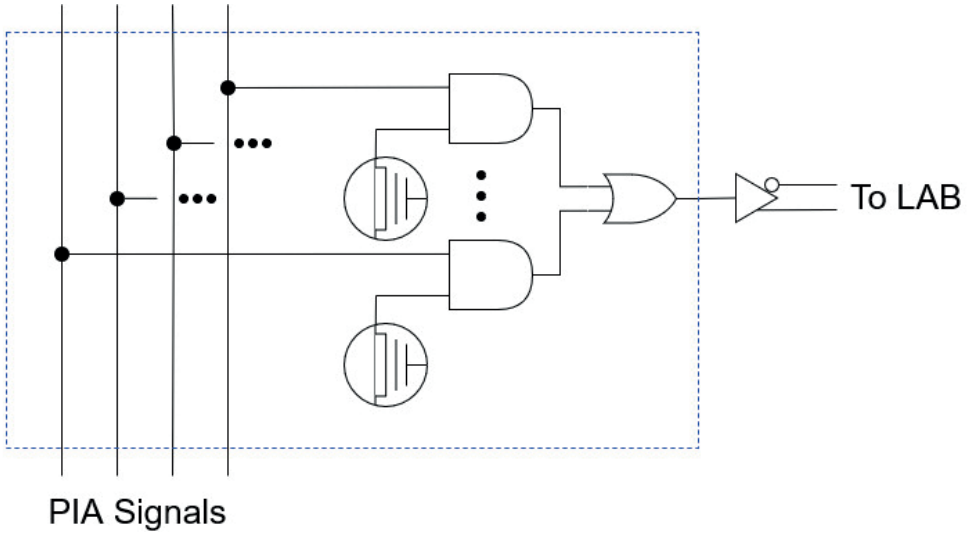



**Рис. 1-10.** PIA – программируемая матрица межсоединений CPLD MAX 3000

Для увеличения гибкости и сложности реализуемых в рассматриваемых CPLD схем, предусмотрены возможности расширения комбинаторной логики макроячеек. В каждой макроячейке некоторый промежуточный терм логической функции через расширитель общих ресурсов (**Shareable Expander**) может быть подключен к линии матрицы межсоединений и использован в других логических блоках. Соответственно, в логическую функцию «ИЛИ» в каждой макроячейке с помощью схемы расширения может быть добавлен терм, созданный в других макроячейках. Тем самым практически снимаются ограничения на сложность логических схем проектируемых устройств при одновременной минимизации расходуемых ресурсов (логических вентилей).

Не останавливаясь на схемотехнических особенностях макроячейки, коротко рассмотрим, как выполнен блок управления внешними выводами MAX 3000A (**I/O Control Block**) (рис. 1-11).

Каждый контакт ввода-вывода микросхемы (**I/O pin**) конфигурируется как входной, выходной или двунаправленный, и может устанавливаться в высокоимпедансное состояние. Индивидуальное управление логикой контакта осуществляется с помощью мультиплексора выбора режима (**OE Select Multiplexer**) шестью глобальными сигналами разрешения (**Global Output Signals**), поступающим из матрицы межсоединений (**PIA**), и константными уровнями «0» и «1». Дополнительно функциональность контакта расширяется сигналом перевода выхода в режим с открытым стоком (**Open-Drain Output**) и сигналом управления скоростью изменения выходного напряжения (**Slew-Rate Control**).

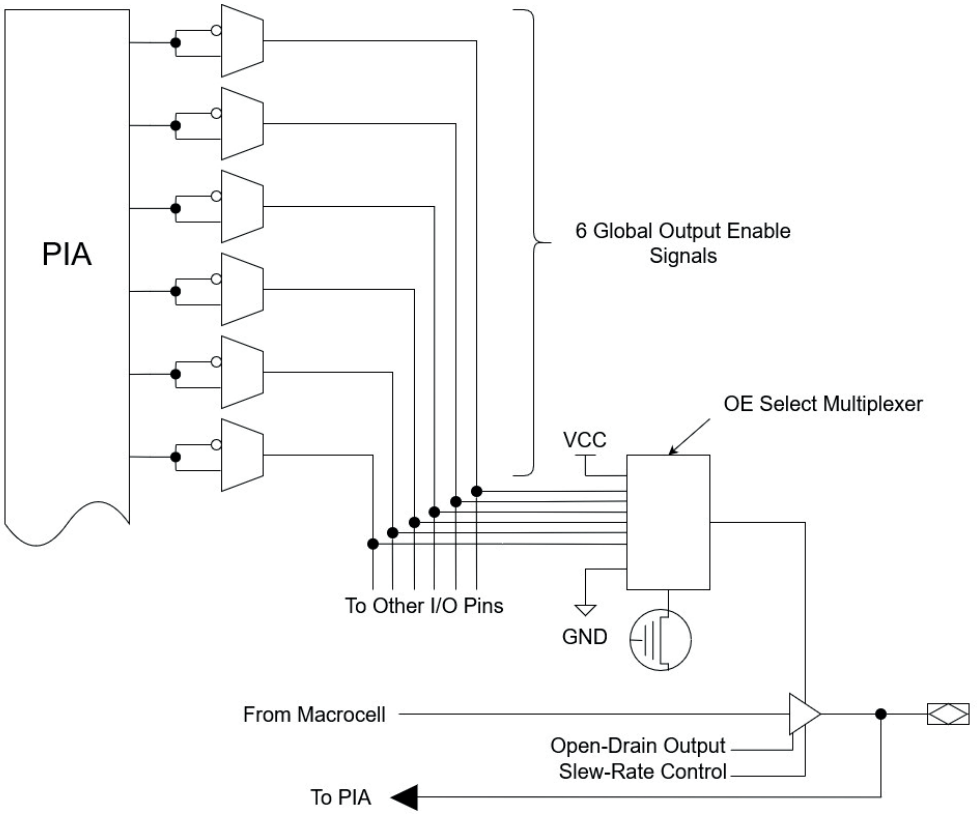



**Рис. 1-11.** Блок управления внешними выводами CPLD MAX 3000

Дальнейшее развитие архитектуры программируемых логических устройств класса Complex PLD происходит в направлении усложнения логики макроячеек, расширения возможностей конфигурирования внутренних связей, повышения быстродействия и снижения потребляемой мощности, реализации дополнительных пользовательских функций и т.п. Так, например, компания Altera выпускает CPLD (MAX II, MAX V) с тактовой частотой свыше 300 МГц, количеством эквивалентных макроячеек до 2210, количеством каналов ввода-вывода до 272, возможностью выбора уровней и типа сигналов TTL-логики, блоком конфигурационной памяти Flash-типа, блоком пользовательской энергонезависимой (Flash) памяти емкостью 8 кбит. В этих CPLD реализованы функции управления потребляемой мощностью и быстродействием, полная поддержка технологии многократного внутрисхемного программирования и тестирования по принципу пограничного сканирования и многое другое.

CPLD с аналогичными характеристиками выпускают и другие компании. Подробно ознакомиться с возможностями и характеристиками современных CPLD-устройств можно по посвященным этой теме книгам и технической документации различных фирм. Здесь же приведем обобщенные характеристики ПЛИС этого класса.


Основные свойства и отличительные признаки CPLD:

- в одной микросхеме находится не менее двух логических блоков;
- в одном логическом блоке находится не менее двух макроячеек;
- обычно все макроячейки одинаковые;
- каждая макроячейка имеет свой собственный триггер;
- логическая функция реализуется в макроячейке;
- связи между логическими блоками конфигурируются с помощью глобальной матрицы межсоединений.

Достоинства CPLD – регулярность и универсальность топологии кристалла, благодаря чему обеспечиваются высокая степень использования ресурсов и высокое быстродействие. Это также позволяет легко рассчитать время задержки распространения сигналов.

Основной недостаток CPLD является продолжением их достоинств – для проектирования сложных цифровых автоматов необходима сложная глобальная матрица межсоединений, которая утилизирует много ресурсов кристалла. Другими словами, универсальность структуры ПЛИС становится ограничением для количества пользовательских компонентов.

## 1.5. Оперативно программируемые логические матрицы

Существует еще одна широко распространенная разновидность сложных программируемых логических устройств, получившая название **Field Programmable Gate Array** (**FPGA**). Слово Field в определении этой разновидности устройств не характеризует принципиальное отличие FPGA от ПЛИС других типов. Они практически все программируются на рабочем месте разработчика аппаратуры, а не на заводе, как, например, непрограммируемые ПЗУ или заказные интегральные схемы высокой степени интеграции, конфигурируемые на завершающих стадиях производства (**ASIC**, **Applications Specific Integrated Circuit**). В литературе многие из рассмотренных разновидностей ПЛИС имеют альтернативные названия, в том числе содержащие слово Field, которые им присваивают разработчики или изготовители часто из конъюнктурных соображений. Более существенное отличие FPGA от других программируемых устройств заключается в том, что их структура хранится в оперативном запоминающем устройстве. Это позволяет перепрограммировать FPGA непосредственно в процессе функционирования системы.

В данной книге будут использоваться те наименования и аббревиатуры, которые применяются чаще всего и впервые введены фирмами-пионерами в данной области, которые создали соответствующий тип ПЛИС. Одной из таких фирм является компания **Xilinx**, выпустившая на рынок в **1984** году новую разновидность сложных программируемых логических устройств под названием **FPGA** [1-6]. Основные особенности FPGA рассмотрим на примере микросхем семейства **Virtex II** компании **Xilinx** [1-13, 1-14], архитектура которых в общем виде представлена на рис. 1-12.

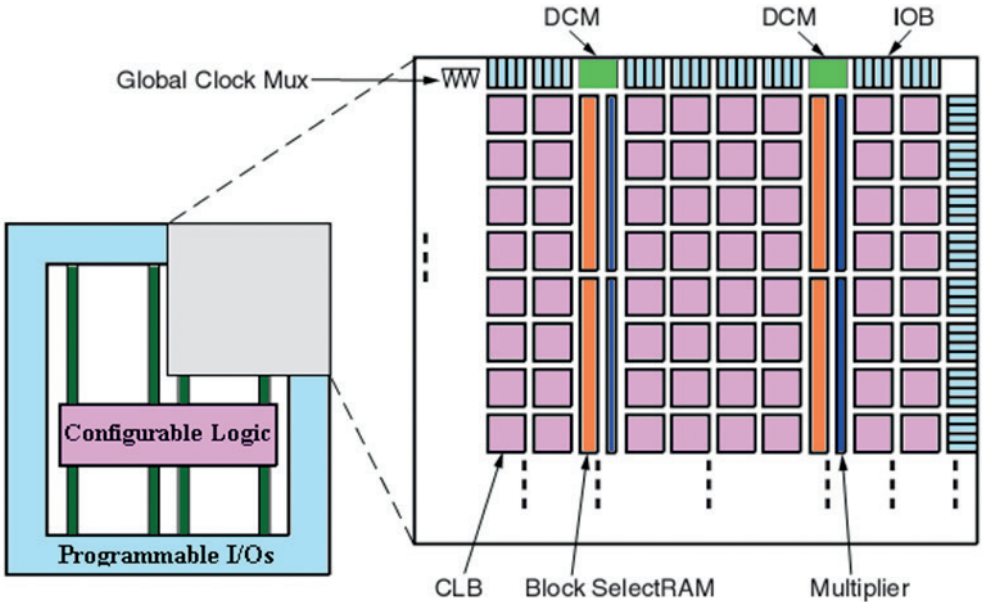



**Рис. 1-12.** Архитектура FPGA семейства Virtex II

В FPGA применяется многоуровневое конфигурирование различных по сложности логических компонентов, основным из которых является конфигурируемый логический блок (**Configurable Logic Block**, **CLB**). На рис. 1-12 они показаны квадратами розового цвета. Эти логические блоки подключаются к внешним контактам микросхемы через программируемые блоки ввода-вывода (**Input/Output Block, IOB** – голубого цвета).

Кроме упомянутых универсальных логических компонентов, FPGA Virtex II содержит несколько специализированных функциональных компонентов:

- отдельные блоки двухпортовой оперативной памяти общего назначения (**Block Select RAM**), емкостью до 18 кбит (оранжевого цвета);
- отдельные блоки 18-разрядных умножителей (синего цвета);
- модули цифрового управления синхронизацией (**Digital Clock Manager**) (зеленого цвета);
- мультиплексор глобальных цепей синхронизации (**Global Clock Mux**).

Для конфигурирования Virtex II используется технология активных иерархических буферизируемых межсоединений – **Active Interconnect Technology** (рис. 1-13).

В приведенном фрагменте FPGA вертикальными (зелеными) и горизонтальными (синими) линиями показаны шины матрицы глобальных межсоединений. Конфигурируемые логические блоки, блоки ввода-вывода и другие основные компоненты FPGA могут подключаться к шинам глобальной синхронизации (по 8 в каждом квадранте), а также к вертикальным и горизонтальным глобальным шинам, проходящим вдоль каждого столбца и каждой строки, для создания логических связей второго уровня. Соединения первого

уровня устанавливаются между любыми блоками FPGA и глобальными шинами через матрицы коммутации (**Switch Matrix**).

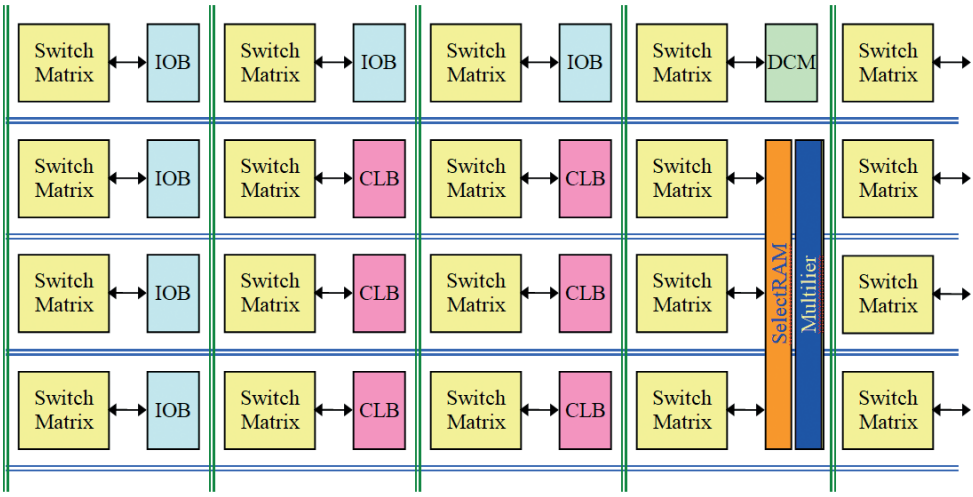



**Рис. 1-13.** Обобщенная схема соединений компонентов FPGA семейства Virtex II

Существует несколько типов глобальных шин. К шинам, состоящим из 24 «длинных» двунаправленных линий (**long Lines**), могут подключаться любые блоки из соответствующих столбца или строки матрицы блоков. Второй тип шин (**hex Lines**) состоит из 120 линий, позволяющих соединять в шахматном порядке блоки, расположенные в ячейках матрицы блоков с номерами, кратными четырем. Шины из 40 линий (**double Lines**) также предназначены для селективного соединения блоков – каждого второго в строке или столбце. И, наконец, еще 16 линий могут быть использованы для непосредственного соединения соседних блоков по всем направлениям – по горизонтали, по вертикали или по любой диагонали.

Возможности реализации логических функций в FPGA определяются структурой конфигурируемого логического блока – CLB (рис. 1-14).

Каждый блок состоит из четырех секций (**Slice**), связанных с коммутирующей матрицей (**Switch Matrix**), в том числе с возможностью организации локальных обратных связей, а также цепями быстрых подключений с соседними блоками (**Fast Connects to neighbors**). Кроме того логические схемы секций имеют входы **CIN** и выходы **COUT** для образования цепей ускоренного переноса.

В составе каждого CLB есть два драйвера линий с тремя состояниями (**TBUF**), позволяющие создавать двунаправленные магистральные связи между блоками.

Каждая секция (рис. 1-15) содержит два 4-входовых генератора логических функций, схему арифметической логики с цепями переноса для организации многоразрядных вычислителей, набор мультиплексоров и два элемента памяти.

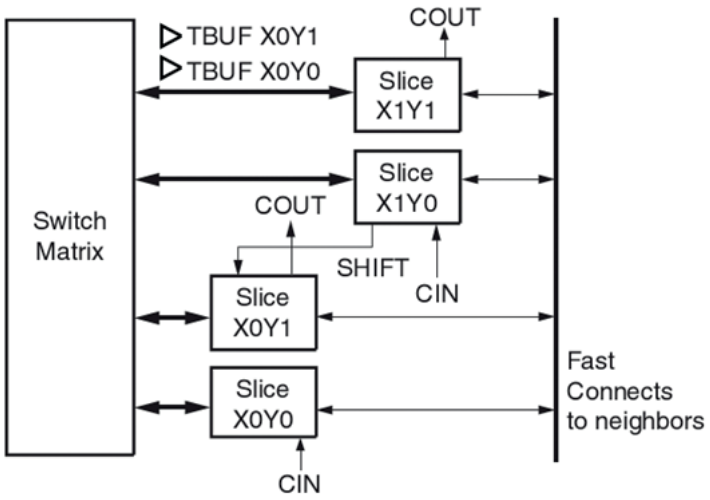



**Рис. 1-14.** Структура CLB

Генератор логических функций может быть сконфигурирован как 4-входовая таблица преобразования (**Look-Up-Table**, **LUT**), 16-битовая распределенная память (**RAM16**) или как 16-разрядный сдвиговый регистр (**SRL16**). Сигналы на основные выходы секций могут поступать непосредственно с выходов генераторов функций или через элементы памяти (**Register**). Таблицы преобразования (**LUTF** и **LUTG**) реализуют произвольные логические комбинационные функции от 4 переменных, а с помощью мультиплексоров **MUXFx** и **MUXF5** количество переменных генерируемых функций может быть увеличено до 8.

При конфигурировании генератора логических функций в качестве устройств памяти (**RAM16**) на базе нескольких логических блоков можно создать блок распределенной оперативной памяти с различной организацией: от 16×8 бит до 128×1 бит для однопортового доступа и от 16×4 до 64×1 – для двухпортового доступа.

За подробным описанием всех возможностей конфигурирования CLB, включая режимы сдвигового регистра, организации арифметических операций, мультиплексирования и т.п., рекомендуем обратиться к технической документации компании Xilinx.

Особого внимания заслуживает блок ввода-вывода (**Input/Output Block**, **IOB**). На рис. 1-16 показана упрощенная функциональная схема блока ввода-вывода FPGA Virtex II.

Каждый внешний контакт микросхемы соединен с тремя буферными узлами ввода (**Input**), вывода (**Output**) и двунаправленного ввода-вывода (**3-State**) блока IOB. Эти узлы предназначены для согласования внешних устройств, подключаемых к FPGA, по электрическим уровням сигналов и по импедансам источника и приемника сигналов. Каждый из упомянутых узлов содержит также по два элемента памяти (**Reg ICK** и **Reg OCK**), необходимых для буферизации данных.

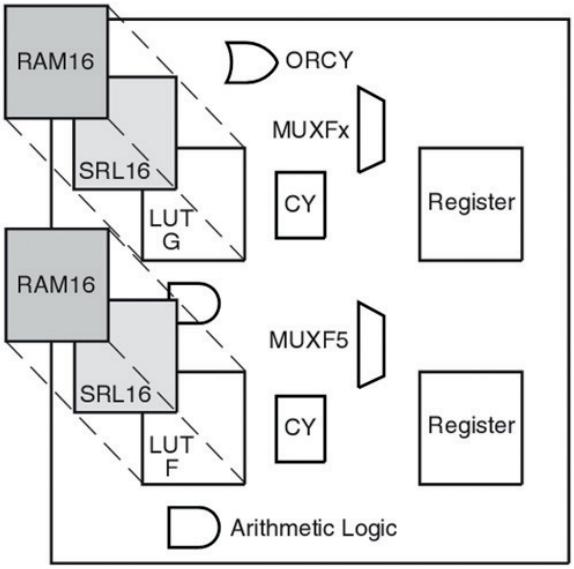



**Рис. 1-15.** Структура секции Slice

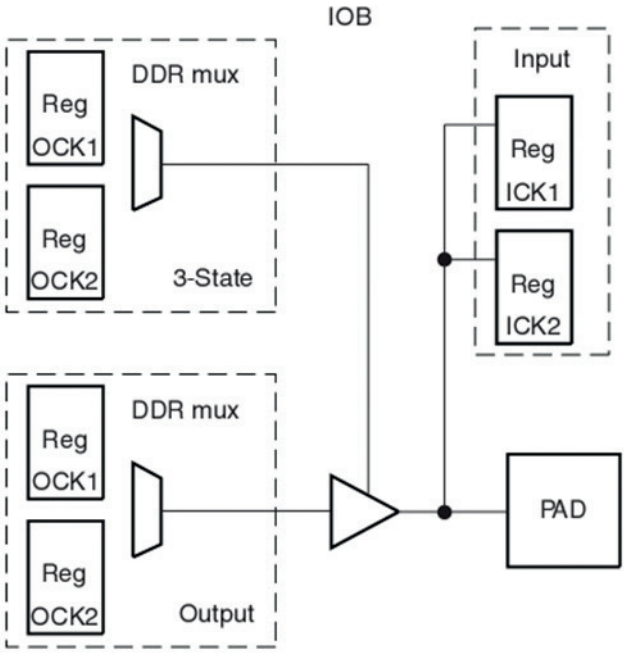



**Рис. 1-16.** Упрощенная схема блока IOB

Все блоки ввода-вывода компонуются в банки и располагаются по периметру кристалла. Каждый блок может быть сконфигурирован как входной, выходной или двунаправленный контакт FPGA для несимметричной линии связи. Два бло-

ка IOB используются в качестве драйвера дифференциальной линии связи. Поддерживаются свыше 30 стандартов обмена данными по несимметричным или дифференциальным линиям связи, в том числе сигналами с низковольтовыми уровнями ТТЛ- и КМОП-логики (LVTTL, LVCMOS, LVDS, BLVDS и др.), обеспечивается совместимость со стандартными шинами PCI, PCI-X, AGP, CardBus и т.д.

Предусмотрена возможность конфигурирования синхронизации буферных элементов памяти по фронту или уровню синхроимпульсов. Реализована поддержка механизма обмена данными с удвоенной скоростью (**Double Data Rate – DDR**), настройка параметров выходных каскадов по нагрузочной способности и многое другое.

При проектировании каскадируемых цифровых устройств важно обеспечить точность и стабильность синхронной работы всех каскадов. Для этих целей в FPGA Virtex II служит модуль цифрового управления синхронизацией (**Digital Clock Manager**, **DCM**), который позволяет синтезировать широкую сетку частот синхроимпульсов, синхронизировать их с внутренним или внешним задающим генератором, формировать требуемые сдвиги фаз, автоматически компенсировать задержки прохождения сигналов и т.п.

В современные ПЛИС (как в FPGA, так и в CPLD) включают не только регулярные многофункциональные конфигурируемые устройства вроде CLB или Macrocell, но и специализированные блоки, позволяющие создавать законченные сложные системы без применения дополнительных микросхем общего назначения. Так, в состав FPGA Virtex II входят упоминавшиеся ранее отдельные блоки оперативной памяти общего назначения – **Block Select RAM** емкостью 18 кбит, которые могут быть использованы как одно- или двухпортовая память с организацией 16K×1, 8K×2, 4K×4, 2K×9, 1K×18 или 512×36 бит.

Второй из специализированных блоков в Virtex II – блок **Multiplier** аппаратного умножения 18-разрядных чисел. Эти умножители применяются совместно с блоками памяти или независимо от них и предназначены для реализации алгоритмов цифровой обработки сигналов.

В табл. 1-1 приведены количественные оценки основных ресурсов FPGA семейства Virtex II, доступные для разработчика специализированной цифровой аппаратуры.

Таким образом, на кристалле FPGA Virtex II может размещаться от 64 до более чем 10 000 конфигурируемых логических блоков, 4 ÷ 168 аппаратных 18-разрядных умножителя, 8 кбит ÷ 1,5 Мбит распределенной и 72 кбит ÷ 3 Мбит блочной памяти. При этом количество несимметричных линий ввода-вывода составляет от 88 (для младшего представителя семейства) до 1108. Их попарное конфигурирование образует дифференциальные линии для ускоренной помехозащищенной передачи данных. Основная тактовая частота ПЛИС этого семейства равна 420 МГц.

Компания Xilinx впервые использовала для хранения конфигурации ПЛИС статическую оперативную память, что позволяет не только быстрее изменять структуру связей, но и снимает принципиальные ограничения на количество циклов реконфигурирования. Таким образом, становится возможным создавать системы, функциональное назначение и характеристики которых могут в полном смысле слова изменяться «на лету» в процессе эксплуатации изделия конечным потребителем.

**Таблица 1-1.** Доступные ресурсы FPGA семейства Virtex II

| Модель FPGA Virtex II | Количество логических элементов | Количество конфигурируемых логических блоков CLB (1 CLB = 4 Slices = 128 bits) | | | Количество блоков умножения | Блочная память (SelectRAM) | | Количество модулей цифрового управления синхронизацией (DCM) | Количество контактов ввода-вывода |
|---|---|---|---|---|---|---|---|---|---|
| | | Размер матрицы CLB (строк x столбцов) | Количество секций (Slices) | Объем распределенной памяти, Кбит | | Количество блоков SelectRAM | Объем распределенной памяти, Кбит | | |
| XC2V40 | 40K | 8 x 8 | 256 | 8 | 4 | 4 | 72 | 4 | 88 |
| XC2V80 | 80K | 16 x 8 | 512 | 16 | 8 | 8 | 144 | 4 | 120 |
| XC2V250 | 250K | 24 x 16 | 1536 | 48 | 24 | 24 | 432 | 8 | 200 |
| XC2V500 | 500K | 32 x 24 | 3072 | 96 | 32 | 32 | 576 | 8 | 264 |
| XC2V1000 | 1M | 40 x 32 | 5120 | 160 | 40 | 40 | 720 | 8 | 432 |
| XC2V1500 | 1,5M | 48 x 40 | 7680 | 240 | 48 | 48 | 864 | 8 | 528 |
| XC2V2000 | 2M | 56 x 48 | 10752 | 336 | 56 | 56 | 1008 | 8 | 624 |
| XC2V3000 | 3M | 64 x 56 | 14336 | 448 | 96 | 96 | 1728 | 12 | 720 |
| XC2V4000 | 4M | 80 x 72 | 23040 | 720 | 120 | 120 | 2160 | 12 | 912 |
| XC2V6000 | 6M | 96 x 88 | 33792 | 1056 | 144 | 144 | 2592 | 12 | 1104 |
| XC2V8000 | 8M | 112x104 | 46592 | 1456 | 168 | 168 | 3024 | 12 | 1108 |

Конфигурирование FPGA Virtex II осуществляется через стандартный JTAG-порт или через дополнительные линии специального последовательного порта. При этом несколько микросхем, соединенных в цепочку, могут быть сконфигурированы последовательно из одного источника данных конфигурации, например из ПЗУ с последовательным интерфейсом. Кроме того, в этом семействе FPGA предусмотрен режим быстрой загрузки конфигурационного файла через 8-разрядную шину данных.

Принято считать, что для FPGA характерны следующие отличительные признаки:

- матрица конфигурируемых логических блоков может состоять из узлов разных типов, т.е. конфигурируемая матрица может быть неоднородной;
- FPGA может содержать специализированные цифровые узлы;
- существуют FPGA с «крупнозернистой» или «мелкозернистой» архитектурой, отличающиеся сложностью логических блоков и возможностями связи между ними;
- в «крупнозернистых» FPGA логические блоки содержат, по меньшей мере, один комбинационный логический элемент и один элемент памяти;

- в «мелкозернистых»FPGA логические блоки обычно содержат отдельный комбинационный логический элемент или один элемент памяти;
- каждый логический блок может быть соединен с другим логическим блоком или блоком ввода-вывода;
- логические функции реализуются с помощью глобальных коммутационных матриц и иерархических локальных связей.

К главным достоинствам FPGA относят возможность создания на их основе систем высокой сложности и эффективное использование ресурсов, а также высокое быстродействие.

Вместе с тем FPGA свойственны и недостатки:

- несмотря на то, что в FPGA применяется гибкая система организации связей между блоками, реализовать произвольную логическую структуру оказывается достаточно сложно;
- для проектирования систем необходимо использовать высококачественные инструментальные средства;
- существуют проблемы с обеспечением предсказуемости временны́х задержек.

## 1.6. Сравнение архитектур ПЛИС

Завершая рассмотрение основных типов и архитектур ПЛИС, приведем их укрупненную классификацию в наглядном графическом виде (рис. 1-17), одновременно показав используемые технологии программирования [1-6, 1-15, 1-16].

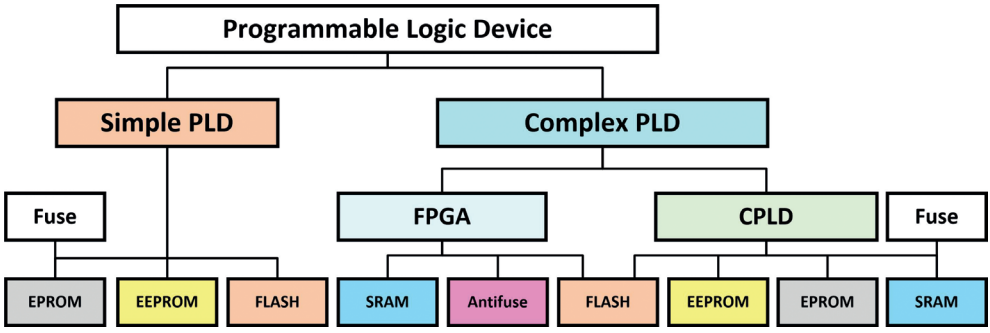



**Рис. 1-17.** Классификация ПЛИС

Из показанных на рис. 1-17 технологий программирования ранее не упоминался термин **Antifuse**, относящийся к технологии конфигурирования путем создания соединений (перемычек), подобной применяемой в самых первых однократно программируемых ПЗУ. Также используется и «инверсия» этой технологии – выжигание перемычек (**Fuse**). Технология **Antifuse** отличается радиационной стойкостью, повышенной надежностью сохранения конфигурации в экстремальных условиях и применяется в компонентах и системах специального назначения.

Выбор между SPLD и CPLD сделать сравнительно просто. Он определяется логической сложностью проектируемого изделия.

Выбор между CPLD или FPGA не столь очевиден. Даже главные конкуренты и законодатели мод в ПЛИС – компания Intel FPGA (бывшая Altera) и компания Xilinx– производят изделия обоих типов с сопоставимыми характеристиками. При этом в процессе развития ПЛИС и совершенствования технологий многие достижения одной архитектуры заимствуются или реализуются тем или иным способом в другой архитектуре. В табл. 1-2 приведены результаты сравнения свойств CPLD и FPGA, отмечаемые в многочисленных публикациях на эту тему.

**Таблица 1-2.** Сравнение свойств ПЛИС[1]

| Свойство | FPGA | CPLD |
|---|---|---|
| Уровень интеграции | ++ | + |
| Структура межсоединений | Сегментированная | Непрерывная (однородная) |
| Временны́е соотношения | Неодинаковые (непредсказуемые) | Фиксированные (предсказуемые) |
| Производительность | + | ++ |
| Использование ресурсов | + | ++ |
| Доводка конфигурации «вручную» | Может потребоваться | Не требуется |
| Возможность реконфигурирования | Есть | Есть |
| Компиляция | + | ++ |
| Стоимость | ++ | + |

Принципиальные свойства архитектур этих двух классов ПЛИС определяют области их предпочтительного применения. Как правило, CPLD эффективнее использовать при реализации сложных логических функций, а FPGA дают лучшие результаты при создании систем обработки данных. Но редко задачи обработки данных могут быть решены без использования сугубо логических алгоритмов, поэтому, в конечном счете, выбор элементной базы должен проводиться самим разработчиком прикладной системы с учетом не только особенностей решаемой задачи, назначения и области применения системы, но и ряда других факторов, вплоть до собственного опыта создания подобных систем.

[1] Большему количеству знаков «+» соответствует более высокое значение свойства.

Следует отметить, что существует еще одна проблема – как выбрать производителя ПЛИС? Например, дискуссии о достоинствах и недостатках FPGA компаний Altera и Xilinx велись много лет [1-17, 1-18]. Сравнение проводилось с использованием разнообразных методик и критериев, но далеко не всегда весьма убедительные по отдельности, но противоречащие друг другу результаты сравнения зачастую не могут послужить надежным основанием для выбора производителя и семейства FPGA. Как это часто бывает, в подобных случаях надо опираться на свой опыт разработки, самим сравнивать отдельные типы FPGA и принимать решение, исходя из конкретных требований к конечному изделию.

Компания National Instruments (NI), о продукции которой пойдет речь далее, для создания устройств измерения, обработки данных и управления, названных **устройствами реконфигурируемого ввода-вывода** (**RIO**), выбрала FPGA производства **Xilinx**. В первых модулях и контроллерах NI RIO использовались FPGA семейств **Virtex II** и **Spartan 3**, содержащие до 14,3 и 20,5 тыс. секций (Slices) соответственно (эквивалентно 2 и 3 млн. логических вентилей). Каждая секция этих FPGA содержит по две 4-входовые таблицы преобразования (LUT) и два триггера. Этих ресурсов оказалось достаточно, чтобы реализовать новые функциональные возможности, радикально повысить детерминизм и скорость обработки данных систем реального времени. Непрерывному расширению номенклатуры и улучшению характеристик изделий RIO способствовало появление все более мощных программируемых интегральных схем и средств их программирования.

Ниже приводится краткий обзор некоторых новых компонентов Xilinx.

## 1.7. Современные программируемые логические схемы Xilinx FPGA 7 серии

В эту серию входят FPGA семейств **Spartan**, **Artix**, **Kintex** и **Virtex**, выполненные по технологии 28 нм. По сравнению с FPGA предыдущих серий у них увеличена производительность, количество логических секций и других традиционных компонентов ПЛИС, объем памяти, а также снижена потребляемая мощность. Поскольку каждая секция состоит из четырех 6-входовых таблиц преобразования и восьми триггеров, можно реализовать более сложные логические функции, а увеличение разрядности вычислительных блоков позволяет повысить точность вычислений и расширить динамический диапазон обрабатываемых данных [1-19, 1-20].

Не меньший интерес представляют разнообразные функциональные блоки, реализованные в этих FPGA: мультигигабитные трансиверы для последовательного обмена данными, интегрированные порты интерфейса PCIe, секции DSP. Почти во всех моделях FPGA 7 серии есть блок XADC, в состав которого входят сдвоенный 12-разрядный 17-канальный аналого-цифровой преобразователь (АЦП) с частотой преобразования 1 МГц. Этот блок позволяет контролировать состояние кристалла с помощью интегрированных датчиков (температура, питание), а также предоставляет возможность использовать FPGA для работы с аналоговыми сигналами.