СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
SIBERIAN FEDERAL UNIVERSITY

ПРОГРАММИРУЕМЫЕ ЛОГИЧЕСКИЕ ИНТЕГРАЛЬНЫЕ СХЕМЫ

Учебное пособие

**ИНСТИТУТ КОСМИЧЕСКИХ
И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

УДК 621.3.049.77:004.42(07)
ББК 32.844.15я73
П784

Р е ц е н з е н т ы:

В. А. Хабаров, кандидат технических наук, ведущий инженер-конструктор отдела систем управления АО «НПП “Радиосвязь”»;

В. В. Прудков, кандидат технических наук, начальник группы отдела проектирования и испытания бортовой РЭА систем управления КА АО «ИСС» им. акад. М. Ф. Решетнева

П784 **Программируемые логические интегральные схемы** : учеб. пособие / Н. Ю. Сиротинина, О. В. Непомнящий, А. И. Постников, Д. А. Недорезов. – Красноярск : Сиб. федер. ун-т, 2020. – 224 с.

ISBN 978-5-7638-4244-9

Рассмотрены архитектуры программируемых логических интегральных схем, маршруты проектирования на их основе, средства разработки ПЛИС-проектов на языке описания аппаратуры Verilog. Представлены практические работы, позволяющие освоить разработку ПЛИС-проектов.

Предназначено для студентов бакалавриата всех форм обучения по направлению подготовки 09.03.01 «Информатика и вычислительная техника» в качестве основной литературы по дисциплине «Программируемые логические интегральные схемы». Может использоваться в качестве основной и дополнительной литературы при изучении дисциплин «Программируемые логические интегральные схемы. Дополнительные главы», «Прикладная теория цифровых автоматов», «Схемотехника ЭВМ» и др.

Электронный вариант издания см.:
http://catalog.sfu-kras.ru

УДК 621.3.049.77:004.42(07)
ББК 32.844.15я73

Разработано в рамках проекта Erasmus+ 573545-EPP-1-2016-1-DE-EPPKA2-CBHE-JP Applied curricula in space exploration and intelligent robotic systems / Прикладные образовательные программы в области освоения космоса и интеллектуальных робототехнических систем.

Co-funded by the Erasmus+ Programme of the European Union: Joint project Capacity Building in the field of Higher Education 573545-EPP-1-2016-1-DE-EPPKA2-CBHE-JP Applied curricula in space exploration and intelligent robotic systems.

ISBN 978-5-7638-4244-9

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Цель преподавания дисциплины «Программируемые логические интегральные схемы» (ПЛИС) – обучение студентов использованию методов и средств проектирования цифровых устройств на базе программируемых интегральных схем.

Разработка и создание цифровых устройств различного назначения на базе ПЛИС является одним из наиболее активно развивающихся направлений развития цифровой элементной базы. Основными областями применения ПЛИС в настоящее время являются:

- цифровая обработка сигналов;
- высокопроизводительные вычисления;
- криптографические системы, системы защиты информации;
- мосты (коммутаторы) между системами с различной логикой, напряжением питания, с несовместимыми или нестандартными интерфейсами;
- нейронные сети и прочие разработки из области искусственного интеллекта;
- системы с перестраиваемой (реконфигурируемой) архитектурой;
- прототипирование заказных интегральных схем.

Обширный список областей применения ПЛИС позволяет предположить, что специалисты в области разработки ПЛИС-проектов будут востребованы на рынке труда.

Пособие содержит базовый объем информации, достаточный для того, чтобы студент получил представление об особенностях организации современных ПЛИС с различными архитектурами, их преимуществах и недостатках, применимости для решения различных типов прикладных задач, проблемах эффективного использования. Практическая часть пособия позволяет освоить базовые навыки применения средств разработки и отладки ПЛИС-проектов.

После освоения курса студент должен уметь:

- выполнять декомпозицию проектируемого цифрового устройства на модули;
- описывать отдельные модули и ПЛИС-проект в целом средствами схемного ввода и с использованием языка описания аппаратуры Verilog;

- разрабатывать тестовое окружение ПЛИС-проекта;
- выполнять моделирование, тестирование и отладку ПЛИС-проектов средствами среды разработки Quartus и ModelSim;
- программировать целевую ПЛИС;
- тестировать полученное изделие.

Пособие состоит из пяти глав и практической части.

Первая глава содержит общие сведения о ПЛИС, предпосылки появления ПЛИС в контексте истории развития вычислительной техники. Приведена классификация ПЛИС, и охарактеризованы основные области их применения. Особое внимание уделено архитектурам ПЛИС класса FPGA. Перечислены основные производители ПЛИС, и дана краткая характеристика их продукции. Дано представление о маршруте проектирования цифровых устройств на базе ПЛИС. Введено понятие языков описания аппаратуры.

Во второй главе рассмотрены общие вопросы разработки ПЛИС-проектов и создания синтезируемых описаний на языке Verilog. Рассмотрены основные конструкции Verilog, соглашения о порядке именования. Описаны основные типы переменных, как универсальные, так и специфические для языков описания аппаратуры, такие как цепи и регистры. Перечислены основные операции Verilog. Рассмотрена структура ПЛИС-проекта. Введены понятия модуля и экземпляра модуля (инстанса), даны основные способы и правила описания модулей.

Третья глава посвящена вопросам синтезируемого описания асинхронных (комбинационных) схем. Рассмотрены операция непрерывного присваивания (назначения) assign и особенности ее применения в синтезируемых ПЛИС-проектах , способы описания условной логики в комбинационных схемах.

Четвертая глава посвящена синтезируемым описаниям синхронных схем. Дано понятие поведенческого блока always, рассмотрены особенности описания синхронных и асинхронных управляющих сигналов. Особое внимание уделено специфическим операциям блокирующего и неблокирующего присваивания. Рассмотрены синтезируемые описания типовых синхронных схем, используемых в ПЛИС-проектах: модулей памяти, регистров с расширенной функциональностью, таймер-счетчиков. Описаны подходы к проектированию цифровых автоматов Мура и Мили.

В последней главе дано общее представление о возможности повторного использования разработанных компонентов, о библиотечных и IP-ядрах различной сложности, в частности, о soft-процессорах.

Курс практических работ состоит из двух видов заданий: тематических практических работ и заключительного проекта.

Цель тематических работ – изучение особенностей организации ПЛИС, средств и инструментов разработки ПЛИС-проектов, подходов к описанию модулей различных типов.

Цель итогового проекта – получение навыков применения инструментов и приемов разработки ПЛИС-проекта в комплексе.

Для того чтобы успешно усвоить материал курса, необходимо иметь навыки программирования на языке высокого уровня, представление о математической логике, прикладной теории цифровых автоматов, схемотехнике ЭВМ.

Материал курса далее будет востребован в дисциплинах «Микропроцессорные системы», может оказаться полезным при выполнении выпускной квалификационной работы и при дальнейшем обучении в магистратуре. Пособие может использоваться в качестве основной и дополнительной литературы для таких дисциплин, как «Программируемые логические интегральные схемы. Дополнительные главы» (магистратура), «Прикладная теория цифровых автоматов», «Схемотехника ЭВМ» и др.

1. ОБЩИЕ СВЕДЕНИЯ О ПЛИС

1.1. История вычислительной техники и предпосылки появления ПЛИС

В 1937 году американский математик, специалист в области теории шифрования Клод Шеннон (Claude Elwood Shannon) показал, что существует соответствие «один к одному» между конструкциями булевой логики и некоторыми электронными схемами, которые получили название «логические вентили». Он продемонстрировал, что комбинационные схемы – логические вентили, объединенные связями, – эквивалентны выражениям булевой алгебры. Своей работой “A Symbolic Analysis of Relay and Switching Circuits” он создал основу для практического проектирования цифровых схем, которые могли осуществлять вычисления [1, 2].

Одним из основных кандидатов на роль первой электронной вычислительной машины, воплотившей принципы Шеннона, считается программируемый электронный калькулятор общего назначения ENIAC (Electronic Numerical Integrator and Computer – электронный цифровой интегратор и вычислитель), который был создан в период с 1942 по 1946 год в университете Пенсильвании математиками и инженерами Джоном Мочли (John J. Mauchly) и Преспером Эккертом (J. Presper Eckert). Важной особенностью этой вычислительной системы являлось то, что порядок выполнения вычислений задавался схемой коммутации компонентов на 40 наборных полях. Таким образом, для каждой прикладной задачи вычислительная система фактически конфигурировалась заново. Процесс коммутации занимал длительное время – от нескольких часов до нескольких суток [3]. В проекте принимал активное участие американский физик и математик Джон фон Нейман (John von Neumann). Фон Нейман в своей работе представил научному миру свои идеи по организации компьютера [4]. Именно эти идеи, получившие название «принципы фон Неймана», в основном определили развитие вычислительной техники на длительный период. Одним из принципов фон Неймана является принцип программного управления.

Принцип программного управления гласит, что все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов – команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, т. е. в порядке их положения в программе. При необходимости эта последовательность может быть изменена с помощью специальных команд. Была сформирована архитектура вычислительной машины, выполненной согласно принципам фон Неймана. Позднее в архитектуре ЭВМ был выделен процессор (рис. 1).

Процессор (центральный процессор, ЦП; от англ. central processing unit, CPU) – электронный блок, исполняющий машинные команды, главная часть аппаратного обеспечения компьютера.

В то же время Аланом Тьюрингом было доказано, что программируемая вычислительная машина является универсальным инструментом, т. е. способна реализовать любой алгоритм обработки информации. Далее история развития цифровой техники стала прежде всего историей развития программируемых электронных вычислительных машин. Эту историю принято разбивать на поколения [3].

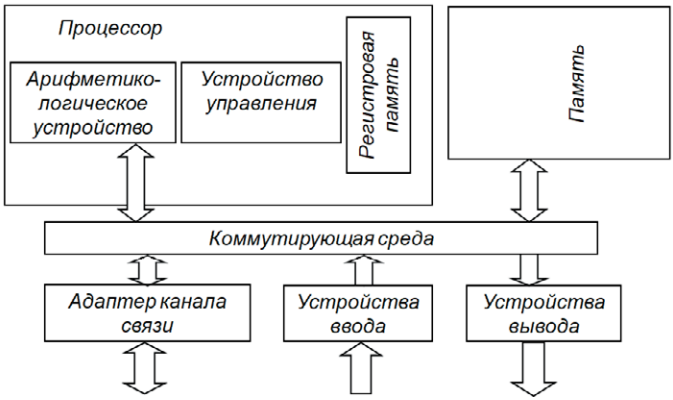

Рис. 1. Архитектура фон Неймана

Первое поколение (1945–1954), представителем которого был упомянутый ENIAC, было реализовано на электронных лампах.

Во втором поколении (1955–1964) вместо электронных ламп использовались транзисторы, а в качестве устройств памяти стали применяться магнитные сердечники и магнитные барабаны.

В третьем поколении (1965–1974) впервые стали использоваться интегральные схемы, выполняющие как функции обработки информации, так и функции ее хранения.

Интегральная схема (микросхема) – микроэлектронное изделие, выполняющее определенную функцию преобразования, обработки сигнала, накапливания информации и имеющее высокую плотность электрически соединенных элементов (или элементов и компонентов), которые с точки зрения требований к испытаниям, приемке, поставке и эксплуатации рассматриваются как единое целое.

Для оценки сложности цифровых СБИС используется единица «эквивалентный вентиль» (например, 2-входовый элемент «И-НЕ»), который соответствует четырем эквивалентным транзисторам (рис. 2).

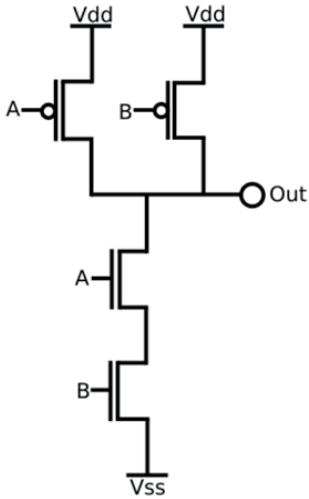

Рис. 2. Пример эквивалентного вентиля 2-входового И-НЕ

На первых этапах для реализации процессоров использовались микросхемы низкой и средней степени интеграции и сам процессор представлял собой сложный многокомпонентный модуль. По мере развития технологии и роста степени интеграции микросхем количество элементов и соединений между ними, реализуемых на одном кристалле, постоянно росло. В 1970-е годы интегральные схемы содержали уже тысячи транзисторов, что позволило реализовать в одной микросхеме отдельные элементы цифровой схемотехники: логические элементы, затем более сложные модули – регистры, счётчики, сумматоры. Позднее появились микросхемы, содержа-

щие функциональные блоки процессора: микропрограммное устройство, арифметико-логическое устройство, устройства работы с шинами данных и команд [2, 3].

Наконец, в 1971 году фирма Intel выпустила первый микропроцессор, который положил начало четвертому поколению ЭВМ.

Микропроцессор – это программно-управляемое устройство, которое отвечает за выполнение арифметических, логических операций и операций управления, записанных в машинном коде, конструктивно выполненное в виде одной большой интегральной схемы (БИС) [5].

Многие исследователи считают, что на этом смена поколений ЭВМ прекратилась, прогресс идет в основном по пути развития того, что уже изобретено. Основным направлением развития долгое время оставалось уменьшение размеров электронных компонент, увеличение их числа на отдельном кристалле. К настоящему моменту это направление развития вплотную приблизилось к физическому пределу, а каждый следующий шаг становится все более затруднительным. В то же время запросы потребителей и рынка к вычислительной технике постоянно растут. В основном это касается таких характеристик, как производительность, автономность и энергонезависимость, компактность вычислительных устройств. Ведется поиск новых архитектур, в основном в направлении параллельной организации вычислительных систем, но это не снимает остроты проблемы. Ситуация требует поиска резервов для повышения перечисленных характеристик и принципиально новых решений.

Программно-управляемые устройства универсальны, т. е. способны решить практически любую задачу за счет реализации принципа программного управления, описанного выше. Программное управление позволяет изменить алгоритм работы и выполняемые функции за счет замены программного обеспечения, не затрагивая аппаратную часть. Однако выполнение программы требует значительных затрат времени на обработку машинных команд, обмен с памятью команд и данных и прочие этапы командного цикла, а сами программно-управляемые системы имеют достаточно сложную архитектуру. Большинство специальных задач может быть успешно решено с использованием программно-управляемых устройств (ЭВМ, микроконтроллеров) за счет разработки и оптимизации соответствующего программного обеспечения. Но если

накладываются жесткие критические ограничения на объем аппаратуры и время выполнения вычислений, программно-управляемые устройства часто не обеспечивают нужные характеристики. Аппаратная реализация функций в подавляющем большинстве случаев является более быстродействующей, хотя и лишена свойства универсальности. По этой причине перед разработчиками вычислительной техники, наряду с задачей проектирования универсальных вычислительных машин, всегда стояли проблемы создания специализированной аппаратуры, предназначенной для выполнения конкретных, специфических операций. Долгое время для подобного применения разрабатывались специализированные многокомпонентные устройства из отдельных элементов и микросхем малой и средней степени интеграции, а в настоящее время с этой целью применяются специализированные интегральные микросхемы.

Специализированные (заказные) интегральные микросхемы (Application Specific Integrated Circuit, ASIC) – микросхемы, которые имеют жестко заданную на этапе разработки структуру, проектируются и изготавливаются под заказ и предназначены для выполнения конкретной задачи или круга задач [6]. Специализированные интегральные схемы выполняют строго ограниченные функции, характерные только для данного устройства, причем эффективность реализации этих функций обычно очень высока.

Микросхемы класса ASIC получили широкое распространение, поскольку они являются практически единственным приемлемым решением при реализации сложных изделий микроэлектроники для портативной и носимой аппаратуры. ASIC широко используются в различных отраслях микроэлектроники, включая телеметрию, коммуникационные системы, глобальные навигационные системы, мультимедиа, автомобильную и бытовую электронику и т. д. Основное преимущество таких микросхем – быстродействие. При этом современные ASIC настолько сложны, что их проектирование даже с применением современных инструментов САПР является серьезной проблемой и часто требует значительных трудозатрат и недопустимо много времени. При мелкосерийном и единичном производстве ASIC сто́ят значительно дороже массово изготавливаемых микросхем, а в случае даже незначительных изменений в алгоритме функционирования требуют повторной разработки в полном объеме. Поэтому применение ASIC становится эконо-

мически оправданным в случае достаточно большого тиража продукции из-за высоких затрат на разработку и отладку проекта [6, 7].

Сравнительно новым решением, которое может во многих случаях заменить ASIC, являются программируемые логические интегральные схемы.

Программируемая логическая интегральная схема (ПЛИС) – электронный компонент, логика работы которого не определяется при изготовлении, а задаётся посредством программирования связей между элементами [7, 8].

Важный момент, на который стоит еще раз обратить внимание: при программировании ПЛИС не пишется программа, определяющая последовательность операций, а задаются (программируются) связи между элементами. Можно сказать, что ПЛИС представляет собой конструктор, с помощью которого фактически собирается аппаратная реализация требуемой функции.

В зависимости от сложности ПЛИС реализуемые функции могут варьироваться от довольно простых, соответствующих одной или нескольким микросхемам низкого уровня интеграции, до весьма сложных устройств. На ПЛИС можно реализовать алгоритмы цифровой обработки сигналов (ЦОС) и другие сложные задачи обработки данных и управления, включая и возможность сборки своего собственного процессора (soft-процессора) с уникальной архитектурой.

История ПЛИС пишется прямо сейчас. Хотя первые реализации микросхем, которые можно отнести к ПЛИС, относятся к середине 70-х годов прошлого века, заметное влияние на современную вычислительную технику они начали оказывать только последние 10–15 лет. В настоящее время ПЛИС – современная, активно развивающаяся, удобная в освоении и применении элементная база. Развитие ПЛИС связано прежде всего с увеличением количества элементов на кристалле. Общепринятой оценкой логической емкости ПЛИС является число эквивалентных вентилей, определяемое как среднее число вентилей «И-НЕ» на два входа, необходимых для реализации эквивалентного проекта. Эта оценка очень приблизительна, однако для проведения сравнительного анализа различных проектов ее можно использовать. Широко выпускаемые ПЛИС с эквивалентной емкостью более 1 миллиона логических вентилей предоставляют разработчику достаточно ресурсов для реали-

зации сложных и интересных проектов. Цены производителей на ПЛИС неуклонно падают. Существуют и также постоянно совершенствуются инструменты и системы автоматизированного проектирования на базе ПЛИС. Все это вызвало потребность в подготовке специалистов, способных проводить разработку аппаратуры на базе ПЛИС, владеющих основными методами и средствами проектирования, ориентирующихся в современной элементной базе и программном обеспечении.

1.2. Классификация ПЛИС

Отметим, что в области ПЛИС, которая возникла сравнительно недавно и активно развивается, терминология не является устоявшейся. Разные авторы предпочитают разные варианты перевода англоязычных терминов или заимствований; как правило, это не создает особых проблем. Другая сложность состоит в том, что иногда нет однозначного соответствия между русско- и англоязычными терминами.

Наиболее точно термину «ПЛИС» соответствует Programmable Logic Device, PLD. Но нужно иметь в виду, что часто в русскоязычной литературе термину «ПЛИС» перевод как FPGA – Field Programmable Gate Array, дословно – программируемый пользователем массив логических вентилей. Это соответствие, строго говоря, не вполне точно. FPGA представляют собой один из типов ПЛИС, наиболее распространенный в настоящее время.

Основными критериями такой классификации являются наличие, вид и способы коммутации элементов логических матриц. По этим признакам можно выделить следующие классы ПЛИС (рис. 3).

Рассмотрим приведенные на рис. 3 типы ПЛИС с указанием исторической последовательности их появления.

Согласно приведенной классификации на верхнем уровне ПЛИС подразделяются на логические устройства, программируемые в заводских условиях (factory programmable), и устройства, программируемые пользователем (field programmable) [9].

К первым можно отнести микросхемы энергонезависимой памяти, обычно используемые для хранения неизменяемых программ и данных – ROM (read-only memory), а также устройства с масочным

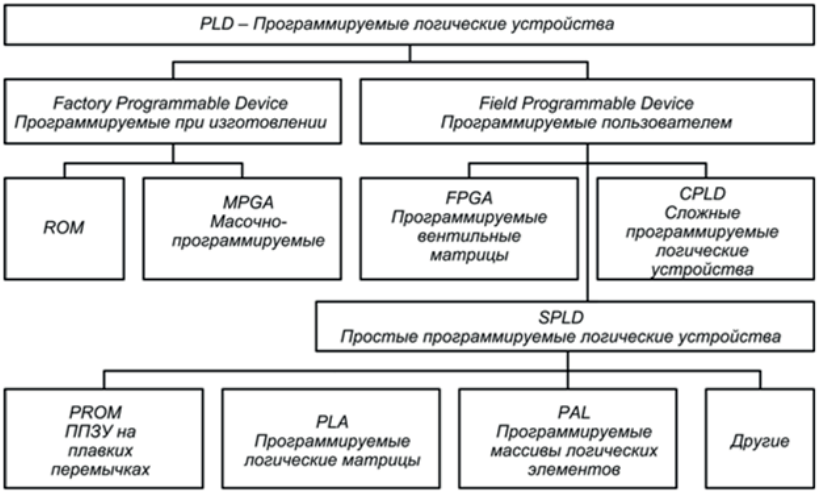

Рис. 3. Семейства программируемых логических интегральных схем

программированием (mask programmable gate array). Программирование масочных схем происходит в процессе изготовления БИС. Обычно на кристалле полупроводника вначале создаются все запоминающие элементы (ЗЭ), а затем, на заключительных технологических операциях, с помощью фотошаблона слоя коммутации реализуются связи между линиями адреса, данных и собственно запоминающим элементом. Этот шаблон (маска) выполняется в соответствии с техническим заданием заказчика.

Класс устройств, программируемых при изготовлении, является достаточно обширным; к нему могут быть, в частности, отнесены некоторые разновидности специализированных заказных микросхем. Но устройства этого типа в данном курсе не рассматриваются подробно.

Первые из появившихся программируемых пользователем логических устройств относятся к классу простых программируемых логических устройств (ППЛУ, Simple Programmable Logic Devices, SPLD) [9, 10].

История развития ПЛИС начинается с появления в начале 70-х годов прошлого века программируемых постоянных запоминающих устройств (ППЗУ, Programming Read-Only Memory, PROM), входящих

в класс программируемых логических устройств (ПЛУ, Programmable Logic Device, PLD). Изначально основной функцией ППЗУ являлось хранение программ, констант, таблиц знакогенераторов и т. п. При этом на вход схемы поступает адрес ячейки памяти, а на выход – хранимые данные.

В то же время если предположить, что входные сигналы схемы – это значения аргументов логической функции, а в ячейки памяти занесены соответствующие им значения этой функции, то мы получаем фактически ее табличную реализацию. ППЗУ реализует функции в виде комбинационной схемы, соответствующей совершенной дизъюнктивной нормальной форме (СДНФ). Поскольку слово памяти имеет обычно несколько разрядов, а схема ППЗУ соответственно несколько выходов, то может быть реализовано сразу несколько логических функций.

Конструктивно такая микросхема представляет собой фиксированную матрицу элементов «И», в которой реализуются все возможные конъюнкции входных сигналов, подсоединенную к программируемому набору логических функций «ИЛИ». Незапрограммированное ППЗУ имеет полные соединения между шинами входных и выходных переменных (на рис. 4 отмечены косыми крестиками) на всех пересечениях матриц «И» и «ИЛИ». Программирование ППЗУ, т. е. занесение в него данных, осуществляется путем электрического разрушения ненужных соединений. Таким образом, на выходе можно реализовать любую функцию от трех переменных в виде СДНФ.

В качестве примера рассмотрим ППЗУ с тремя входами (a, b и c) и тремя выходами (w, x и y) (рис. 4). На программируемом массиве элементов «ИЛИ» в процессе программирования оставляются связи, отмеченные кружками, остальные разрушаются.

Схема, приведенная на рис. 4, реализует следующие функции:

$$\text{выход } w = a\&b\&\overline{c}\ Va\&b\&c = a\&b;$$

$$\text{выход } x = \overline{a}\,\&\,\overline{b}\,\&\,\overline{c}\ V\overline{a}\,\&\,\overline{b}\,\&cV\overline{a}\,\&b\&\overline{c}\ V\overline{a}\,\&b\&cVa\&\overline{c}\,\&\overline{c}\ Va\&\overline{b}\,\&c =$$

$$= \overline{a}\,\&\,\overline{b}\ V\overline{a}\,\&bVa\&\overline{b}\ = \overline{a}\ V\overline{b}\,;$$

$$\text{выход } y = \overline{a}\,\&\,\overline{b}\,\&cV\overline{a}\,\&b\&cVa\&\overline{b}\,\&cVa\&b\&\overline{c}\ = \overline{a}\,\&cV\overline{b}\,\&cVa\&b\&\overline{c}\,.$$

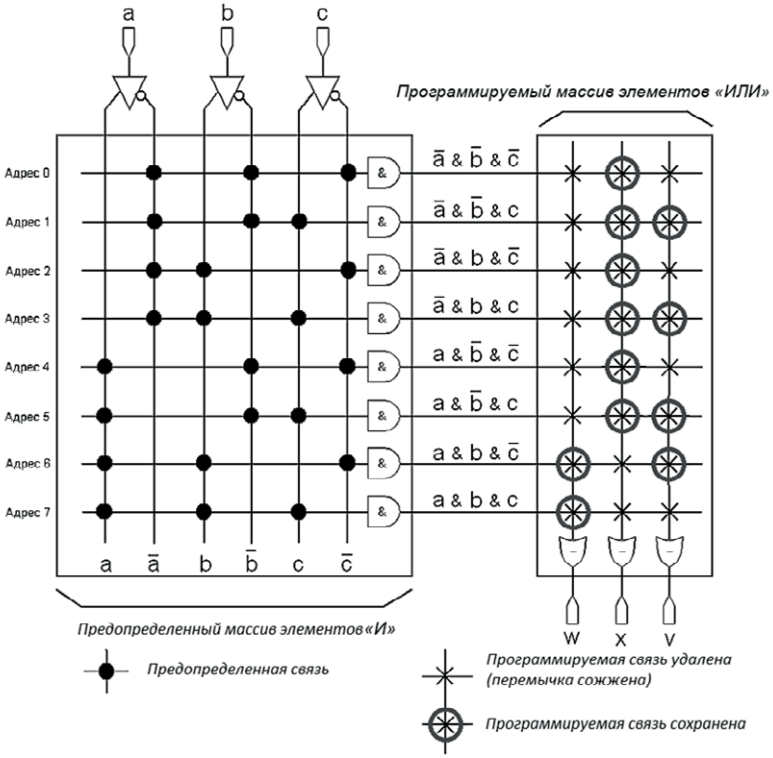

Рис. 4. Пример использования ППЗУ для реализации простых логических функций

Такой способ программирования позволяет использовать микросхемы ППЗУ в качестве простых программируемых устройств для реализации простых логических функций. Для более сложных устройств он не подходит, поскольку сложность схемы с увеличением числа входных сигналов растет экспоненциально: если добавить один входной сигнал (аргумент функции), сложность схемы увеличится примерно в два раза. Время прохождения сигналов через схему также возрастет, а значит, снизится ее быстродействие. Такой способ реализации не оптимизируется. По этим причинам использование ППЗУ в качестве ПЛИС оправдано для реализации устройств с небольшим числом входов (аргументов функции) и большим числом реализуемых функций.

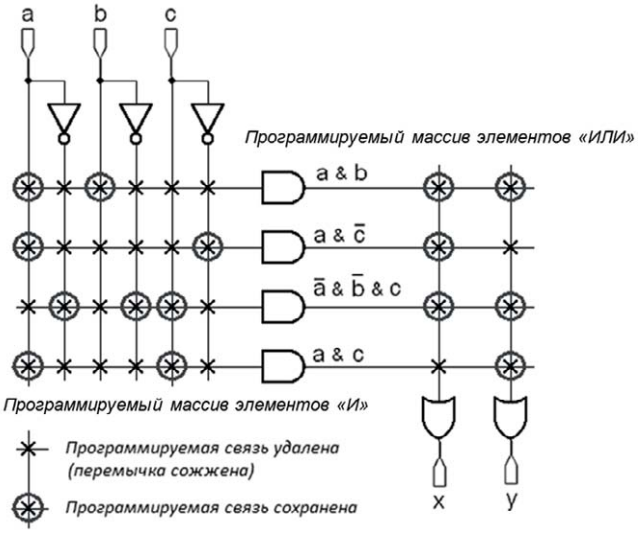

Рис. 5. Пример использования ПЛМ (PLA) для реализации простой логической функции

Исторически следующими появились программируемые логические матрицы (ПЛМ), созданные в 1975 году [9, 10].

Программируемые логические матрицы (ПЛМ, Programmable Logic Array, PLA) являются универсальными логическими схемами, предназначенными для реализации систем логических функций, заданных в дизъюнктивной нормальной форме [11, 12].

ПЛМ состоит из двух программируемых логических матриц (матрицы «И» и матрицы «ИЛИ») и вспомогательных схем, обеспечивающих сопряжение и программирование микросхемы. Термин «матрица» здесь используется в смысле «регулярная структура», набор однородных элементов и связей между ними. Кроме того, схемные элементы подобных изделий удобнее всего располагать по строкам и столбцам, обеспечивая тем самым регулярность структуры БИС [12].

Схема, приведенная на рис. 5, реализует следующие функции:

$$\text{выход } x = a\&bVa\&\overline{c}\,V\overline{a}\,\&\,\overline{b}\,\&c;$$

$$\text{выход } y = a\&bV\overline{a}\,\&\,\overline{b}\,\&cVa\&c.$$

Такие микросхемы обеспечивали более широкие возможности реконфигурирования, но в то же время имели более высокую цену. Недостаточно использовались ресурсы программируемой матрицы «ИЛИ». Решением проблемы явились устройства, основанные на архитектуре программируемой матричной логики (ПМЛ, Programmable Array Logic, PAL), которые появились в конце 70-х годов XX века и были выделены в отдельный класс. В ПМЛ программируемой является логическая матрица элементов «И», а матрица элементов «ИЛИ» является фиксированной (рис. 6) [13, 14].

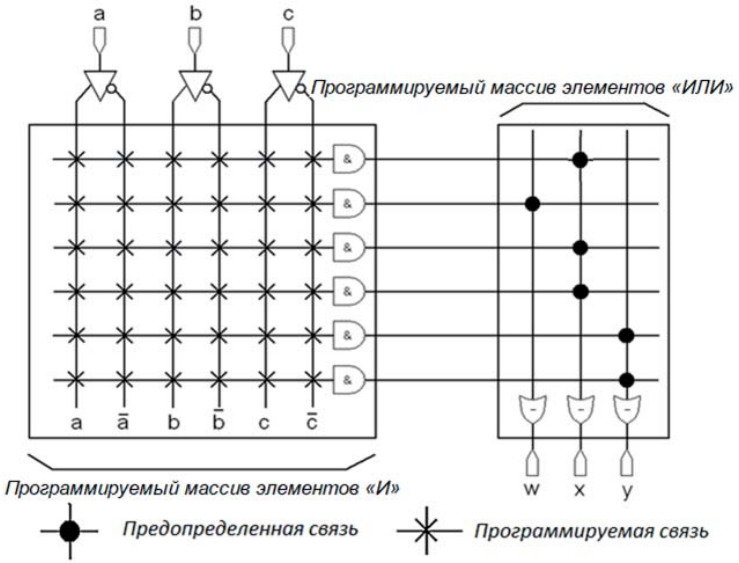

Рис. 6. Незапрограммированное устройство PAL

Эти устройства сочетают достаточно высокую гибкость, сравнимую с характеристиками, свойственную ПЛМ, и быстродействие, свойственное ППЗУ. При этом матрицы логических компонентов имеют значительно меньший размер. Но функциональность программируемого массива логики ограничена, так как его архитектура позволяет выполнять операцию логического сложения только над ограниченным набором логических произведений.