ГЛАВА 4

АРХИТЕКТУРА ПЛИС

Предупреждение

Данная глава посвящена особенностям построения ПЛИС. При этом часть вопросов, таких, например, как сравнение технологий на основе метода наращиваемых перемычек или на основе ячеек памяти статического ОЗУ, в явном виде не рассматривается. Некоторые поставщики ПЛИС специализируются на одной из упомянутых технологий, другие предлагают различные семейства устройств, основанные на обеих технологиях. Большинство рассуждений в этой главе относится к устройствам на основе ячеек статического ОЗУ.

При использовании в составе ПЛИС встраиваемых блоков, таких как мультиплексоры, сумматоры, память и микропроцессорные ядра, каждый поставщик предлагает свое «меню» этих блоков с различным «составом» ингредиентов. Аналогично различным сортам печенья с шоколадной крошкой, которые отличаются количеством шоколадных крошек, разные ПЛИС будут содержать большие/лучшие/худшие встроенные блоки ОЗУ или будут отличаться параметрами умножителей либо поддерживаемыми стандартами ввода/вывода, либо чем-нибудь ещё.

Проблема заключается в том, что характеристики, поддерживаемые каждым производителем и каждым семейством устройств, меняются едва ли не каждый день. Это значит, что при выборе микросхемы по определенным параметрам необходимо провести небольшое исследование, чтобы выяснить, устройства каких поставщиков, максимально удовлетворяют вашим запросам.

Небольшое введение

Перед тем как погрузиться в эту главу, уточним определения некоторых терминов, чтобы убедиться в том, что мы говорим на одном и том же языке. Например, рассмотрим термин *структура*, который встречается в этой книге. Если этот термин используется в контексте описания кремниевого кристалла, это значит, что он имеет отношение к основной структуре устройства (например, как в выражении «структура цивилизованного общества»).

Когда кто-то впервые слышит этот термин, произнесенный в данном контексте, возможно, он покажется несколько вычурным или даже высокопарным. Между прочим, некоторые инженеры действительно считают, что это не что иное, как очередной рекламный трюк. Другими словами, они считают, что таким образом поставщики специализированных микросхем и ПЛИС раскручивают свои устройства, создавая им имидж более сложных, чем они есть на самом деле.

Но стоит только привыкнуть к этому термину, как сразу понимаешь, что он даже очень полезный.

Когда речь идет о *проектных нормах* микросхем, подразумеваются размеры конкретных структур, сформированных на кристалле, — аналогично тому, как под словом *канал* мы понимаем часть *полевого транзистора*. Эти структуры невероятно малы. В начале и середине 80-х устройства основывались на 3-х микронной геометрии, т. е. их наименьшие структуры составляли по размеру 3 мкм (три микрометра = три миллионных части метра = три микрона). Звучать это будет так: «Эта микросхема выполнена по трехмикронной технологии».

Разработку каждых новых проектных норм называют *технологическим процессом*. К 90-м стали появляться устройства, основанные на 1-микронной технологии, и на протяжении последующих 10 лет характеристики размеров продолжали падать. С наступлением 21-го века появились высокопроизводительные ИС, выполненные по технологии менее 0.18 мкм. К 2002 году этот показатель сократился до 0.13 мкм, и в 2003 году появятся устройства с технологией 0.09 мкм.

Технологический процесс менее чем примерно 0.5 мкм, называется *глубоким субмикронном (DSM — deep submicron)*. С некоторого момента, а именно с какого точно не определено, поскольку существует несколько определений в зависимости от того, о чём идет речь, мы перемещаемся в область *сверхглубокого субмикрона (USMC — ultra deep submicron)*.

Возникает некомфортная ситуация: когда показатели конфигурации опускаются ниже 1 мкм, приходится выговаривать что-то вроде «ноль целых тринадцать сотых микрона». По этой причине общепринятым стало употреблять приставку «*нано*». Один нано, сокращение от «нанометр», соответствует одной тысячной микрона, что, в свою очередь, соответствует одной тысячной от одной миллионной метра. Поэтому вместо того чтобы бормотать «ноль целых девять сотых микрона» (0.09 мкм), можно просто четко произнести «девяносто нанометров» и покончить с этим. Естественно, оба выражения имеют одинаковый смысл, но если захочется пообщаться с друзьями на эту тему, лучше говорить на общепринятом языке и показать себя знающим и ультрасовременным, чем стариком минувшего тысячелетия.

Наращиваемые перемычки, статическое ОЗУ и прочее...

Устройства на основе статического ОЗУ

Большинство ПЛИС использует для хранения конфигурации ячейки памяти статического ОЗУ, которые могут быть многократно перепрограммируемы. Главным преимуществом этой технологии является то, что они позволяют легко и быстро реализовать и протестировать все новые идеи, относительно легко подстраиваясь под новые стандарты и протоколы. Кроме того, при включении системы такая ПЛИС может быть изначально запрограммирована для выполнения определенных функций, например, самотестирования или тестирования всей системы, а затем может быть перепрограммирована для выполнения своей главной задачи.

Другое существенное преимущество использования ячеек статического ОЗУ заключается в том, что эта технология является передовой. На поставщиков ПЛИС также действует тот факт, что многие компании, специализирующиеся на устройствах памяти, расходуют огромные ресурсы на исследование и развитие ячеек статического ОЗУ. Более того, эти ячейки памяти создаются по такой же КМОП-технологии, как и остальные части ПЛИС, т. е. для создания этих компонентов не требуются какие-то специальные технологии.

Раньше для апробирования производства микросхем по новым технологиям часто использовались микросхемы памяти. В последнее время совокупность таких характеристик, как размер, сложность и бесперебойность последних поколений ПЛИС сделали возможным их применение для решения и этих задач. В отличие от устройств памяти ПЛИС позволяют легко идентифицировать и обнаруживать дефекты структуры, т. е. установить факт ошибки и даже определить, что и где произошло. Например, когда компании IBM и UMC внедряли технологию 0.09 мкм (90 нм), ПЛИС фирмы Xilinx стали первыми устройствами, изготовленными по такой технологии.

К сожалению, ничего не дается даром, за все надо платить. Недостатком устройств на основе статического ОЗУ является то, что их нужно переконфигурировать каждый раз при включении системы. Для этого приходится использовать специальную внешнюю память, что увеличивает стоимость системы и требует места на печатной плате, или использовать встроенный микропроцессор, или реализовать какие-то другие варианты (*см.* гл. 5).

Защита интеллектуальной собственности в устройствах на основе статического ОЗУ

При использовании ячеек статического ОЗУ может оказаться довольно сложно защитить интеллектуальную собственность, реализованную в данном устройстве. Это обусловлено тем, что конфигурационный файл, используемый для программирования устройства, в том или ином виде хранится во внешней памяти.

В настоящее время существуют некоммерческие инструменты, с помощью которых можно прочитать содержимое конфигурационного файла и сгенерировать соответствующую схему или таблицу соединений. Говоря об этом следует иметь в виду, что понимание и извлечение логики из конфигурационного файла пока не является тривиальной задачей, но и не находится выше предела следующей суммы: сообразительность человека + мощность современного компьютера.

Давайте не будем забывать, что по всему миру существуют компании обратного проектирования, которые специализируются на вскрытии «средств интеллектуальной собственности». Правительства некоторых стран закрывают глаза на воровство интеллектуальной собственности, получая от этого немалые доходы. Надеюсь, все понимают, кого я имею в виду. Если изделие приносит большие доходы, готов поспорить, что найдутся такие ребята, которые будут стремиться повторить его у вас за спиной.

В действительности проблема связана не с личностью, ворующего интеллектуальную собственность с помощью содержимого конфигурационного файла. Все зависит от его способности клонировать устройство независимо от понимания принципов его работы. Используя

легко доступные технологии, любой может взять печатную плату, положить её на испытательную установку с матрицей подпружиненных игольчатых контактов и быстро получить полную таблицу соединений на плате. Такая таблица может быть использована для воспроизводства печатной платы. Теперь единственное, что остается сделать этому гнусному подонку — скопировать конфигурационный файл из загрузочного ППЗУ (или СППЗУ, ЭСППЗУ и так далее), и... он получает дубликат всего устройства.

Радует то, что некоторые современные ПЛИС на основе статического ОЗУ поддерживают концепцию *побитного поточного шифрования*. В этом случае окончательная версия конфигурационных данных шифруется перед сохранением в устройство внешней памяти. Ключ шифрования загружается в специальный регистр на ячейках статического ОЗУ в ПЛИС через порт JTAG (*см.* гл. 5). Совместно с небольшим количеством логики этот ключ позволяет дешифровать входящий зашифрованный конфигурационный поток битов во время загрузки устройства.

Процесс загрузки зашифрованного потока автоматически делает невозможным считывание записанной в ПЛИС информации. Это значит, что обычно можно использовать незашифрованные конфигурационные данные в процессе разработки (когда необходимо использовать считывание записанной информации), а в процессе производства просто перейти к использованию зашифрованных потоков. Разработчик может в любое время загрузить незашифрованный поток. Следовательно, сначала можно загрузить тестовую конфигурацию, а затем перегрузить зашифрованную версию.

Недостатком этой схемы является то, что для её работоспособности на печатной плате необходимо использовать батарею резервного питания, и таким образом сохранить содержимое регистра ключа шифрования, когда ПЛИС будет обесточена. Эта батарея будет иметь продолжительное время жизни, исчисляемое годами или десятилетиями, так как необходимо поддерживать работу только одного регистра в устройстве. Однако её присутствие приводит к увеличению размеров, веса, сложности и стоимости печатной платы.

Устройства на основе наращиваемых перемычек

В отличие от устройств, основанных на ячейках статического ОЗУ, которые программируются при включении системы, устройства на основе наращиваемых перемычек программируются в выключенном состоянии с помощью так называемого *программатора*.

Сторонники ПЛИС на основе наращиваемых перемычек гордятся разнообразием их преимуществ. Во-первых, эти устройства энергонезависимы, т. е. их конфигурационные данные не стираются при отключении питания системы. Это значит, что они готовы к работе сразу после включения системы. Будучи энергонезависимыми, эти устройства не требуют дополнительной микросхемы внешней памяти для хранения конфигурационных данных. Это позволяет уменьшить стоимость всей системы и сохранить свободное место на печатной плате.

Одно из преимуществ ПЛИС на основе наращиваемых перемычек заслуживает особого внимания: структура их внутренних соединений является действительно «сверхжесткой», и значит, такие устройства относительно устойчивы к радиации. Это свойство перемычек может представлять определенный интерес для военных и космических при-

ложений. Состояние конфигурационных ячеек компонентов статического ОЗУ может быть «переключено» в произвольное состояние, если эти ячейки будут поражены *радиацией*, которой так много в космосе. В сравнении с ними, однажды запрограммированная наращиваемая перемычка не может быть изменена таким способом. При этом, следует иметь в виду, что любые триггеры в этом устройстве остаются чувствительными к воздействию радиации, поэтому микросхемы, предназначенные для работы в условиях радиационного воздействия, должны иметь защиту триггеров в виде *системы тройного резервирования*. Другими словами, в системе должно быть три копии каждого регистра, и решение о значении выходного сигнала принимается большинством голосов. В идеальном случае все три регистра будут содержать одинаковое значение, но если один из регистров «переключается» так, что два из них содержат 0, а третий — 1, выходным значением будет 0, и наоборот, если два регистра содержат 1, а третий 0, выходным значением будет считаться 1.

Радиация может проявляться в виде гамма-лучей (высокоэнергетических фотонов), бета-частиц (высокоэнергетических электронов) и альфа-частиц. Следует заметить, что устойчивые к радиации устройства не ограничены только технологией наращиваемых перемычек. Такими свойствами обладают и другие компоненты, в том числе на основе ячеек статического ОЗУ, но в специальных радиационно-устойчивых корпусах и с поддержкой рассмотренной системы тройного резервирования.

Но, возможно, самым важным преимуществом ПЛИС на основе наращиваемых перемычек является то, что их конфигурационные данные скрыты глубоко внутри. По определению программатор может прочитать эти данные, поскольку это часть его работы. После наращивания каждой перемычки программатор должен провести тестирование, чтобы убедиться, что элемент успешно запрограммирован, и лишь затем перейти к следующей перемычке. Кроме того, программатор может использовать автоматическую проверку успешного конфигурирования устройства. Такой подход оправдан в случае устройства, содержащего 50 миллионов элементов не считая программируемых. Чтобы это сделать, программатору потребуется произвести чтение состояния наращиваемых перемычек и полученные результаты сравнить с требуемым состоянием, которое определяется в конфигурационном файле.

После программирования устройства следует воспользоваться возможностью наращивать специальную защиту перемычек, и, таким образом, обеспечить защиту от считывания любых запрограммированных данных из устройства (в форме наличия или отсутствия перемычек). Даже если устройство будет вскрыто, запрограммированные и незапрограммированные перемычки остаются идентичными, к тому же тот факт, что наращиваемые перемычки утоплены во внутренних слоях металлизации, делает конструкцию недоступной для обратного проектирования.

Поставщики ПЛИС на основе наращиваемых перемычек могут рекламировать и два других преимуществ, связанных с мощностью потребления и скоростью, и, если при этом вы не будете очень дотошным, не исключено, что ваш собственный выбор вас разочарует. Например, поставщики могут завлекать вас тем, что устройства на наращиваемых перемычках потребляют в режиме покоя, примерно, всего лишь 20% мощности по сравнению с аналогичными устройства-

ми на статическом ОЗУ, что их рабочая мощность потребления также существенно ниже, а задержка на внутренних соединениях меньше. Они могут также ненароком упомянуть, что наращиваемые перемычки очень маленькие и, таким образом, занимают значительно меньше места на кристалле, чем эквивалентные им ячейки памяти статического ОЗУ. При этом они могут игнорировать тот факт, что устройство на наращиваемых перемычках требуют внешних программируемых цепей, включая большие корпусные транзисторы. Они будут продолжать, утверждая, что, если ваше устройство содержит десятки миллионов конфигурируемых элементов, использование наращиваемых перемычек позволит сделать устройство более компактным. Это, в свою очередь, позволяет уменьшить задержки на внутренних соединениях, делая устройство быстрее, чем его аналоги на статическом ОЗУ.

1821 год. *Англия. Майкл Фарадей (Michael Faraday) изобрел первый электрический двигатель.*

И оба этих утверждения будут справедливыми... при сравнении двух устройств, выполненных по одной технологии. Но это невозможно, так как технология наращиваемых перемычек требует использования приблизительно трех дополнительных технологических шагов после завершения основного технологического процесса. В силу этих, а также других причин устройства на наращиваемых перемычках всегда отстают на одно, а обычно на несколько поколений технологических процессов от компонентов на статическом ОЗУ, которые обладают преимуществами в скорости и потребляемой мощности по сравнению с другими микросхемами.

Устройства на наращиваемых перемычках являются однократно программируемыми, и это их главный недостаток, так как, если эта маленькая штучка уже была однажды запрограммирована, изменить её конфигурацию уже невозможно.

Следует заметить, что когда появятся устройства, основанные на ячейках магнитного ОЗУ, о которых говорилось в гл. 2, то положение среди ПЛИС может существенно измениться. Причиной тому может быть то, что перемычки магнитного ОЗУ будут намного меньше, чем ячейки статического ОЗУ, что приведёт к увеличению плотности элементов на кристалле и уменьшению задержек, при этом такие устройства будут потреблять меньше энергии. Более того, устройства на магнитном ОЗУ могут быть предварительно запрограммированы, подобно микросхемам на наращиваемых перемычках, и впоследствии перепрограммированы, как устройства на ячейках статического ОЗУ.

Устройства на основе СППЗУ

Этот предельно короткий и чрезвычайно приятный раздел, поскольку в настоящее время никто не делает и даже не планирует делать ПЛИС на основе СППЗУ.

Устройства на основе ЭСППЗУ и Flash

Конфигурационные ячейки ПЛИС на основе ЭСППЗУ или Flash-памяти так же, как и ячейки памяти статического ОЗУ, образуют длинную цепочку подобно сдвиговому регистру. Эти устройства могут программироваться в отключенном состоянии с помощью программатора. Некоторые версии устройств можно программировать не

1821 г. *Англия. Майкл Фарадей нарисовал магнитное поле вокруг проводника.*

выключая питание, но при этом время их программирования примерно в три раза превышает время программирования устройств на статическом ОЗУ.

После программирования содержащиеся в устройствах данные будут энергонезависимы, т. е. эти устройства будут сразу готовы к работе после подачи напряжения на систему. Что касается защиты, то некоторые из этих устройств используют принцип мультибитного ключа, размер которого приблизительно от 50 до нескольких сотен бит. После программирования устройства можно загрузить в него личный ключ, т. е. битовую комбинацию, для защиты конфигурационных данных. Дело в том, что, если загрузить ключ, прочитать данные из устройства или записать в него новые данные можно будет только, загрузив копию ключа через JTAG-порт (гл. 5). Если при этом учесть, что JTAG порт современных устройств работает на частоте 20 МГц, взлом ключа методом изнурительного перебора каждого возможного значения займет миллионы лет.

Двухтранзисторные ЭСППЗУ- и Flash-ячейки памяти приблизительно в два с половиной раза больше по размеру своих однотранзисторных СППЗУ братьев, но они существенно меньше, чем ячейки статического ОЗУ. Это значит, что устройство может быть выполнено более компактным, с меньшими задержками на внутренних соединениях.

Вместе с тем эти устройства требуют примерно пять дополнительных технологических шагов после завершения стандартного технологического цикла КМОП. Именно в этом кроется причина их отставания от устройств на статическом ОЗУ на одно или несколько поколений технологических процессов. И последнее, но, как обычно, не менее важное. Эти устройства имеют тенденцию сохранять относительно высокую мощность потребления в статическом режиме из-за содержащегося в них большого количества нагрузочных резисторов.

Гибридные устройства на ячейках Flash и статического ОЗУ

Так уж повелось в этой жизни, что всегда найдется желающий добавить еще какие-нибудь ингредиенты в котелок с супом. Применительно к ПЛИС некоторые производители часто предлагают весьма экзотические комбинации технологий программирования. Рассмотрим, например, устройство, каждый конфигурационный элемент которого представляет собой комбинацию Flash- (или ЭСППЗУ-) ячейки памяти и связанной с ней ячейки памяти статического ОЗУ.

В этом случае Flash-элементы могут быть предварительно запрограммированы. Затем, после включения системы, содержимое Flash-ячеек памяти массово параллельно копируется в соответствующие им ячейки памяти статического ОЗУ. Такая технология обеспечивает энергонезависимость, присущую устройствам на основе наращиваемых перемычек, и это значит, что устройство будет немедленно готово к работе после включения системы. Но, в отличие от устройств на основе наращиваемых перемычек, ячейки памяти статического ОЗУ можно использовать для перепрограммирования устройства во включенном состоянии. Аналогично можно перепрограммировать Flash-ячейки памяти устройства, не снимая напряжения питания, либо выполнить процедуру перепрограммирования с помощью программатора после выключения системы.

Выводы

В **Табл. 4.1** кратко перечислены наиболее важные характеристики рассмотренных технологий программирования.

Таблица 4.1. **Характеристики технологий программирования**

Характеристика	Статическое ОЗУ	Наращиваемые перемычки	ЭСППЗУ/Flash
Технологический процесс	Современный уровень развития	Отставание на одно или несколько поколений	Отставание на одно или несколько поколений
Перепрограммирование	Да (внутрисистемно)	Нет	Да (внутрисистемно или в выключенном состоянии)
Скорость перепрограммирования (включая стирание)	Быстрая	—	В 3 раза медленнее статического ОЗУ
Энергозависимость (возможность программирования при включении питания)	Да	Нет	Нет (возможно при необходимости)
Потребность во внешнем конфигурационном файле	Да	Нет	Нет
Пригодность для изготовления прототипов устройств	Да (высокая пригодность)	Нет	Да (приемлемо)
Готовность к работе сразу после включения	Нет	Да	Да
Защита прав интеллектуальной собственности	Приемлемая	Очень хорошая	Очень хорошая
Размер конфигурационной ячейки	Большой (шесть транзисторов)	Очень малый	Малый (два транзистора)
Мощность потребления	Средняя	Низкая	Средняя
Устойчивость к радиации	Нет	Да	Нет

Мелко-, средне- и крупномодульные архитектуры

В общем случае ПЛИС подразделяются на *мелкомодульные* и *крупномодульные*. Чтобы понять эту классификацию, напомню, что главной особенностью ПЛИС является их внутренняя структура, которая преимущественно состоит из большого количества простых программируемых логических блоков-«островов» в «море» программируемых внутренних связей (**Рис. 4.1**).

В действительности подавляющее большинство конфигурационных ячеек в ПЛИС связано с внутренними соединениями в отличие от конфигурируемых логических блоков. Поэтому инженеры в шутку говорят, что поставщики ПЛИС на самом деле продают только внутренние связи, а логика поставляется бесплатно.

В мелкомодульной архитектуре каждый логический блок может использоваться для реализации только очень простой функции. Например, блок можно сконфигурировать для работы в качестве 3-входо-

1821 год. *Англия. Чарльз Уитстон (Sir Charles Wheatstone) впервые воспроизвёл звук.*

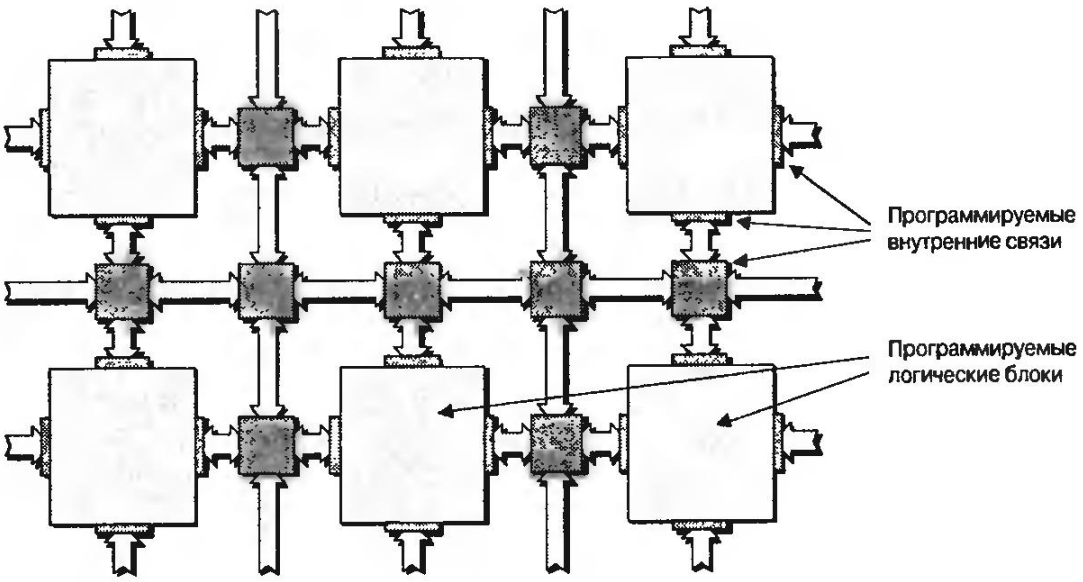


Рис. 4.1. Внутренняя структура ПЛИС

вого простого логического элемента (И, ИЛИ, И-НЕ и так далее) или элемента памяти (триггер D-типа, защелка D-типа и так далее).

Мелкомодульные структуры используются при реализации связующей логики и неоднородных структур, подобных конечным автоматам. Мелкомодульные структуры также эффективны при реализации *систолических алгоритмов* (функции, которые чрезвычайно эффективны за счет реализации массового параллелизма). Эти структуры обладают определённым преимуществом при использовании технологии традиционного логического синтеза, которая базируется на мелкомодульной архитектуре заказных микросхем.

Пик интереса к мелкомодульной архитектуре ПЛИС был отмечен в середине 90-х. Однако со временем подавляющее большинство представителей этого семейства канули в лету, и на плаву остались лишь представители крупномодульной архитектуры. В подобной архитектуре каждый логический блок содержит относительно большое количество логики по сравнению с мелкомодульными представителями. Так, например, логический блок может содержать четыре 4-входовых таблицы соответствия, четыре мультиплексора, четыре D-триггера, и некоторое количество логики быстрого переноса.

Для «мелкомодульных» ПЛИС характерно большое количество соединений внутри блоков и между ними. По мере увеличения модульности устройств до среднемодульных и выше количество соединений в блоках уменьшается. Это важное свойство, так как внутренние связи определяют величину подавляющего большинства задержек, связанных с прохождением сигналов через ПЛИС.

Своеобразной ложкой дегтя в бочке классификации ПЛИС было появление компаний, создающих крупномодульные устройства. Эти устройства содержат массивы узлов, где каждый узел представляет собой сложный элемент, реализующий алгоритмические функции, например *быстрое преобразование Фурье* (БПФ), или даже ядро микропроцессора общего назначения (*см.* гл. 6 и 23). Суть заключается в том, что на самом деле эти устройства мутят воду, поскольку не классифицируются как ПЛИС. По этой причине архитектуру ПЛИС на основе таблиц соответствия (LUT) часто классифицируют как среднемодульную, тем самым, освобождая термин *крупномодульный* для обозначения таких устройств, как устройства на основе узлов.

Логические блоки на мультиплексорах и таблицах соответствия

Существуют два основных способа реализации программируемых логических блоков, используемых для формирования среднемодульных устройств на основе мультиплексоров (MUX — от *multiplexer)* и на основе таблиц соответствия (LUT — от *lookup table*).

Устройства на основе мультиплексоров

В качестве примера реализации устройств на основе мультиплексоров рассмотрим 3-входовую функцию *y = (a & b) | c*, реализованную с помощью блока, содержащего только мультиплексоры (**Рис. 4.2**).

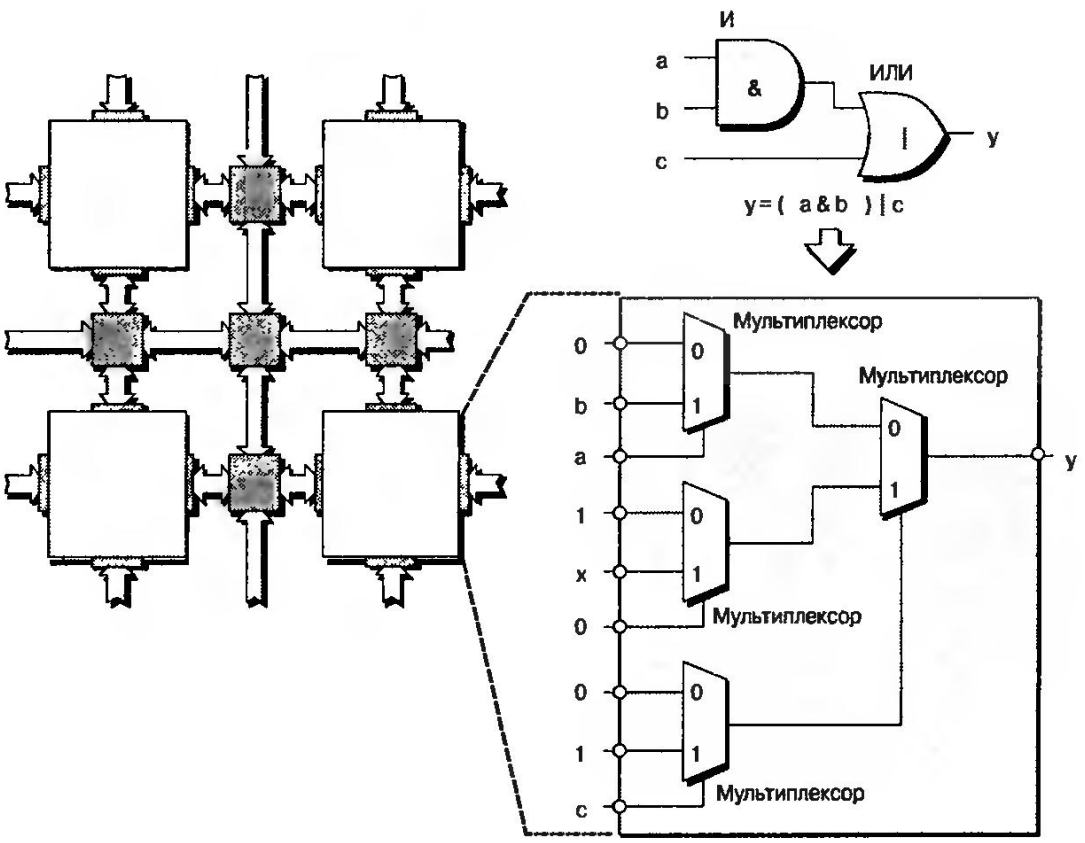


Рис. 4.2. Логический блок на мультиплексорах

Устройство может быть запрограммировано таким образом, что на каждый его вход может подаваться логический 0 либо логическая 1, либо истинное, либо инверсное значение входного сигнала (в нашем случае *a*, *b* или *c*), приходящего с другого блока или с входа микросхемы. Такой подход позволяет для каждого блока создавать огромнейшее количество вариантов конфигурирования для выполнения разнообразнейших функций (*x* на входе центрального мультиплексора на **Рис. 4.2** обозначает, что на вход можно подавать любой сигнал — 0 или 1).

Устройства на основе таблиц соответствия

Основная концепция таблиц соответствия относительно проста. В таких микросхемах группа входных сигналов используется в качестве индекса (указателя, или адреса ячейки) таблицы соответствия. Содержимое этой таблицы организовано таким образом, что ячейки, указываемые каждой входной комбинацией, содержат требуемое выходное значение. Допустим, что требуется реализовать функцию y = (a & b) | c, *см.* **Рис. 4.3**.

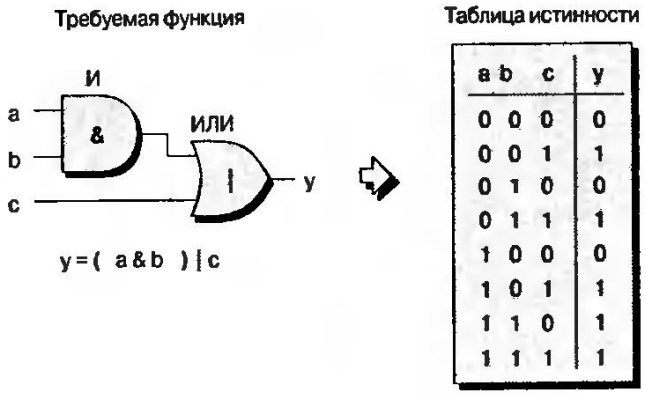


Рис. 4.3. Требуемая функция и соответствующая ей таблица истинности

Если взять группу логических вентилей глубиной в несколько слоёв, таблица соответствия может быть весьма эффективной с точки зрения использования ресурсов и задержки распространения сигнала. Здесь под «глубиной» подразумевается количество логических элементов между входом и выходом цепочки, так на **Рис. 4.3** глубина составляет два слоя. Однако недостатком архитектуры на таблицах соответствия является то, что если с их помощью реализовать небольшую функцию, например 2-входовый логический элемент И, для этого придётся использовать всю таблицу. Результирующая задержка для такой простой функции окажется весьма большой.

Для этого надо загрузить 3-входовую таблицу соответствующими значениями. А теперь допустим, что таблица соответствий формируется из ячеек памяти статического ОЗУ. Она может также быть сформирована наращиваемыми перемычками, ЭСППЗУ- или Flash-ячейками памяти. Для выбора требуемой ячейки ОЗУ с помощью каскада передаточных вентилей используются входные сигналы, как показано на **Рис. 4.4**. При этом ячейки памяти статического ОЗУ для загрузки конфигурационных данных должны быть соединены в длинную цепочку, но эти цепи на **Рис. 4.4** не показаны с целью его упрощения.

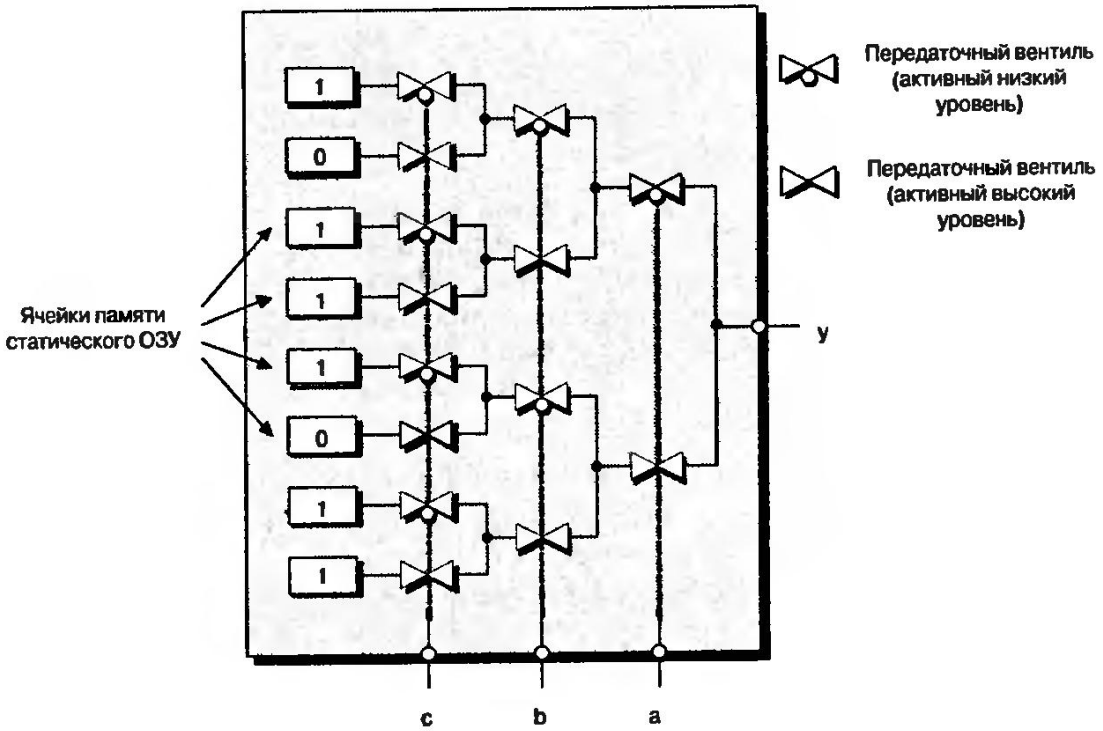


Рис. 4.4. Таблица соответствия на основе передаточных вентилей

На схеме открытый, или активный, передаточный вентиль пропускает сигнал с входа на выход. Закрытый вентиль электрически отключает свой выход от проводника, к которому он подсоединен.

Передаточные вентили, на обозначениях которых изображен небольшой «кружок», активируются при подаче на управляющий вход логического 0. И наоборот, вентили, на обозначениях которых нет кружка, активируются при подаче на управляющий вход уровня логической 1. Основываясь на этом, легко проследить, как различные входные комбинации могут использоваться для выбора содержимого требуемой ячейки памяти.

Мультиплексоры или таблицы соответствия?

До появления современных систем автоматизированного проектирования, когда инженеры вручную изготавливали схемы, некоторые утверждали, что применение архитектуры на основе мультиплексоров позволяет достичь лучших результатов. К сожалению, они, как правило, не удосуживались пояснить, почему эти результаты лучше. По сути дела ответ на этот вопрос перекладывался на плечи других. Говорят также, что мультиплексорные архитектуры имеют преимущество при разработке управляющей логики в стиле «если этот вход имеет значение *истина*, а этот вход имеет значение *ложь*, то на выходе будет *истина*...»[1]. Однако некоторые из этих архитектур не обеспечивают работу высокоскоростных цепочек логического переноса. Их аналоги на таблицах соответствия остаются лидерами во всех приложениях, в которых предусмотрены арифметические действия.

В сравнении с таблицами соответствия, при использовании мультиплексорной архитектуры, содержащей смесь мультиплексоров и логических вентилей, часто удаётся получить доступ к промежуточным значениям сигналов, проходящих между логическими элементами и мультиплексорами. В этом случае при реализации небольших функций ненужные (лишние) логические блоки могут быть отключены. Следовательно, мультиплексорная архитектура может иметь преимущества с точки зрения производительности и использования ресурсов кристалла при реализации устройств, содержащих большое количество простых независимых логических функций.

В 90-х ПЛИС получили широкое применение в области сетей передачи данных и связи. Обе сферы подразумевают передачу больших потоков данных, в которых архитектуры на основе таблиц соответствия зарекомендовали себя очень хорошо. По мере того как устройства и их пропускная способность становились больше, и технология синтеза микросхем становилась сложнее, ручное изготовление схем вместе с мультиплексорной архитектурой постепенно уходили со сцены. В итоге, как будет показано, большинство современных архитектур ПЛИС основывается на таблицах соответствия.

3-, 4-, 5- или 6-входовые таблицы соответствия

Важной особенностью многовходовой таблицы соответствия является то, что она может реализовать любую *n*-входовую комбинационную, или комбинаторную[2], логическую функцию. С помощью боль-

---

[1] Некоторые мультиплексорные архитектуры, такие как выпускаемые компанией QuickLogic (www.quicklogic.com), состоят из логических блоков с множеством мультиплексоров, к выходам которых подключены простые логические вентили, например реализующие функцию **И**. Такая схема обеспечивает большую нагрузочную способность, что дает преимущество в приложениях декодирования адреса и в конечных автоматах.

[2] Как уже упоминалось в гл. 3, некоторые называют комбинационную логику комбинаторной.

1822 г. *Англия. Чарльз Бэббидж (Charles Babbage) приступил к построению механической вычислительной машины, так называемой* ***разностной машины.***

шего количества входов можно реализовать более сложные функции, но при этом следует учитывать, что добавление каждого нового входа сопровождается удвоением количества ячеек статического ОЗУ.

Первые ПЛИС основывались на 3-входовых таблицах соответствия. Впоследствии поставщики ПЛИС и студенты некоторых учебных заведений провели анализ сравнительных эксплуатационных характеристик 3-, 4-, 5- и даже 6-входовых таблиц соответствия (чем бы вы ни занимались, никогда не ввязывайтесь в разговоры проектировщиков ПЛИС). Результаты исследований показали, что оптимальному равновесию всех «за» и «против» отвечают 4-входовые таблицы соответствия.

В прошлом некоторые поставщики использовали в своих устройствах таблицы соответствия разных размеров, например, на три и на четыре входа, так как они сулили наиболее оптимальное использование устройства. Однако это решение не пришлось по душе большинству инженеров, которые предпочли использовать программы синтеза схем. Поэтому в настоящее время все действительно удачные архитектуры основываются только на 4-входовых таблицах соответствия. Это вовсе не означает, что архитектуры на смешанных по размеру таблицах соответствия не появятся вновь, так как программное обеспечение систем проектирования продолжает усложняться.

Таблицы соответствия, распределенное ОЗУ, сдвиговые регистры

Ядро таблицы соответствия в устройстве на статическом ОЗУ использует для своей работы несколько ячеек памяти. Это позволяет использовать некоторые интересные возможности. Помимо основного назначения, т. е. формирования таблицы соответствия, устройства некоторых поставщиков позволяют использовать ячейки, формирующие таблицу, в качестве небольших блоков оперативной памяти. Например, 16 ячеек памяти, формирующих 4-входовую таблицу, могут выступать в роли блока ОЗУ 16×1. Такие участки памяти называются *распределенным ОЗУ*, так как, во-первых, таблицы соответствия разбросаны (распределены) по всей поверхности кристалла, а во-вторых, это название отличает их от больших блоков ОЗУ.

Другой вариант альтернативного использования таблиц основан на том, что все конфигурационные ячейки, включая и те, которые формируют таблицу соответствия, эффективно связаны вместе в одну длинную цепочку (**Рис. 4.5**).

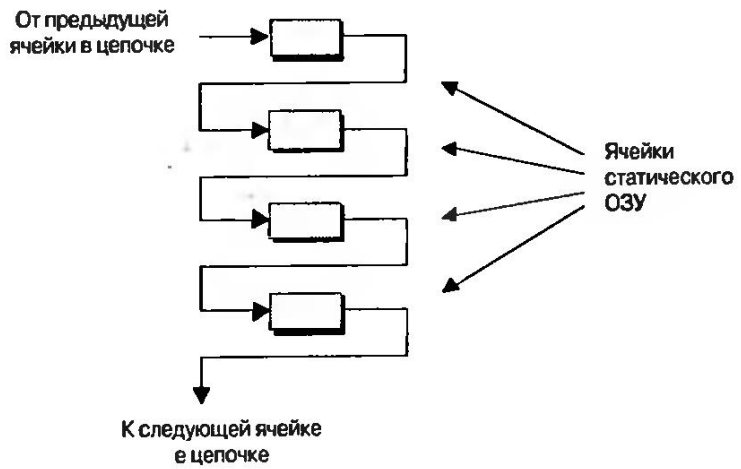


Рис. 4.5. Конфигурационные ячейки, связанные в цепочку

Дело в том, что в некоторых устройствах ячейки памяти, формирующие таблицу соответствия, после программирования могут рассматриваться независимо от главной структуры цепочки и использоваться в качестве сдвигового регистра. Таким образом, каждую таблицу соответствия можно рассматривать как многофункциональный компонент (**Рис. 4.6**).

1822 г. *Франция. Андре Ампер (Andre Ampere) открыл, что два проводника, по которым течет электрический ток, притягиваются друг к другу.*

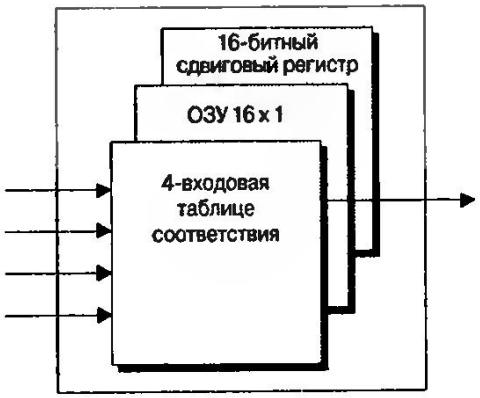


Рис. 4.6. Многофункциональная таблица соответствия

Конфигурируемые логические блоки, блоки логических массивов, секции

«Человек не может жить только таблицами соответствия», непременно сказал бы Шекспир, будь он разработчиком ПЛИС. И действительно, помимо одной или нескольких таблиц соответствия логический блок может содержать другие элементы, такие как мультиплексоры и регистры. Но перед тем как мы начнем копаться в этом разделе, нам необходимо настроить наши мозги на некоторую терминологию.

Логические ячейки фирмы Xilinx

Это может быть показаться странным, но каждый производитель ПЛИС дает им собственное название. Начнем с того, что блоки, из которых состоит современная ПЛИС фирмы Xilinx, называются *логическими ячейками (logic cell)*. Кроме всего прочего, *логическая ячейка* содержит 4-входовую таблицу соответствия, которая может работать как ОЗУ 16×1 или как 16-битный сдвигающий регистр, а также мультиплексор и регистр (**Рис. 4.7**).

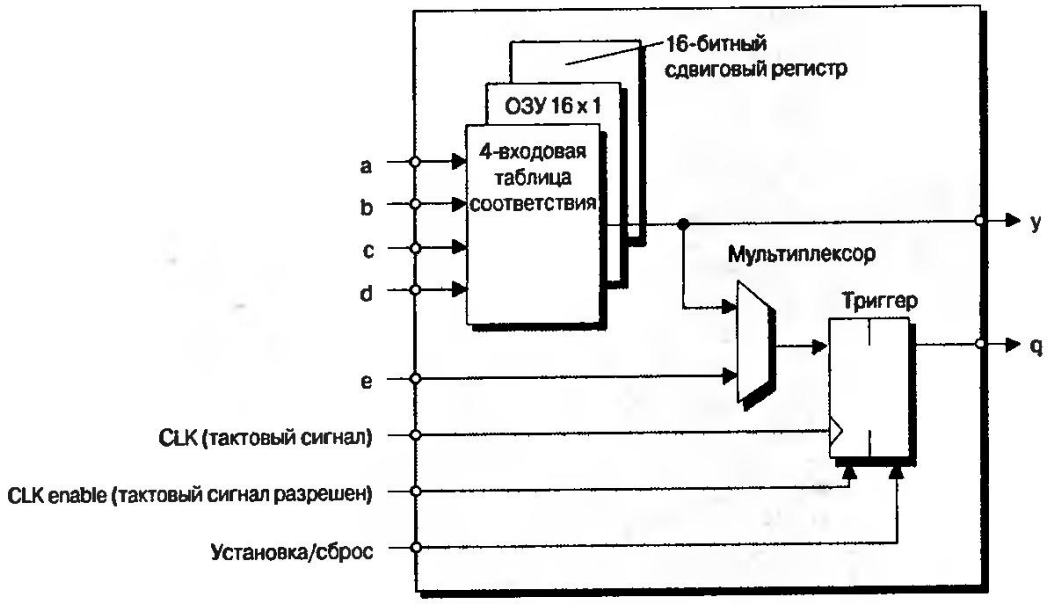


Рис. 4.7. Упрощенный вид логической ячейки Xilinx

1827 г. *Англия. Чарльз Уитстон (Sir Charles Wheatstone) разработал микрофон.*

Схема, представленная на **Рис. 4.7**, сильно упрощена, но, тем не менее, она удовлетворяет контексту рассматриваемого материала. Регистр может быть сконфигурирован для работы в качестве триггера **(Рис. 4.7)** или в качестве защелки. Полярность тактового сигнала (реакция триггера на фронт или спад синхроимпульса) может задаваться программно, так же как полярность сигналов «тактовый сигнал разрешен» и «установка/сброс» (активный высокий или низкий уровень).

Кроме таблиц соответствия, мультиплексоров и регистров, логические ячейки содержат небольшое количество других элементов, включая специальную логику быстрого переноса для использования в арифметических действиях.

Логические элементы компании Altera

Блоки, из которых состоят ПЛИС компании Altera, называются *логическими элементами (logic element)*. Между логическими ячейками Xilinx и логическими элементами Altera существует ряд отличий, но в целом их концепции очень похожи.

Секции и логические ячейки

Следующей ступенью в иерархии построения микросхем программируемой логики является так называемая, по определению фирмы Xilinx, *секция (slice)*. Компания Altera и другие поставщики, по-видимому, называют эту ступень как-то иначе, используя собственные определения. Почему же *секция*? Ну, хорошо, надо же было как-то назвать эту ступень. Так почему же, не назвать его секцией, тем более что бы там не говорили, а *секция* — именно то, что нужно. Во время написания этой книги, *секция* содержала две логические ячейки **(Рис. 4.8)**.

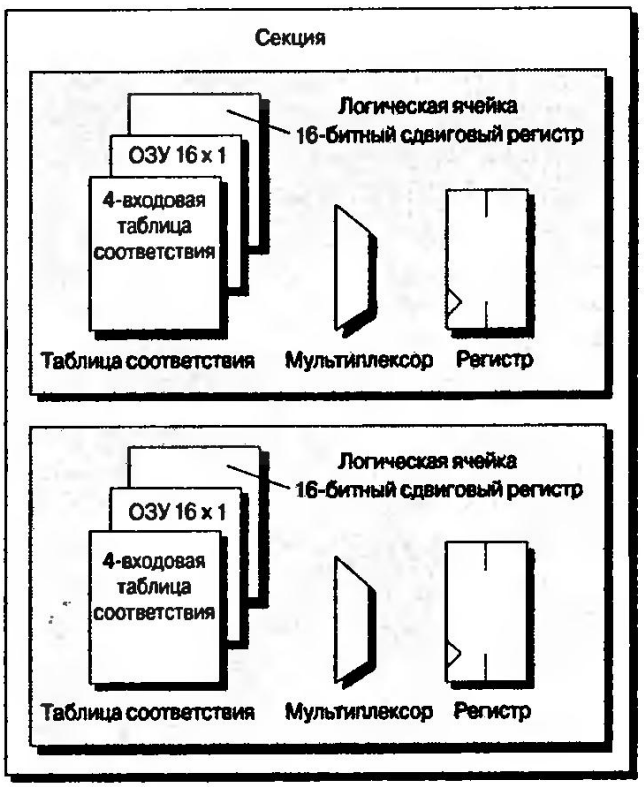


Рис. 4.8. Секция, содержащая две логических ячейки

Приходится использовать слова «во время написания», так как эти определения не только могут меняться, но и меняются с каждым сезоном. На **Рис. 4.8** для его упрощения не показаны внутренние связи, так-

же необходимо заметить, что хотя таблицы соответствия, мультиплексоры и регистры каждой логической ячейки имеют собственные входы и выходы данных. Секция имеет общие тактовые сигналы, разрешения тактовых сигналов и установки/сброса для обеих логических ячеек.

1827 г. *Германия. Георг Ом (Georg Ohm) исследовал электрическое сопротивление и сформулировал закон Ома.*

Конфигурируемые логические блоки CLB и блоки логических массивов LAB

Поднимаясь по иерархической лестнице, мы достигаем уровня, который компания Xilinx называет *конфигурируемый логический блок КЛБ* или (*CLB — configurable logic block*). Компания Altera, в свою очередь, называет его *блоком логических массивов* или *LAB* (*LAB — logic array block*). Другие поставщики ПЛИС, очевидно, дают им свои эквивалентные названия, которые представляют интерес только в случае, когда пользователь имеет дело именно с такими устройствами.

Определение конфигурируемого логического блока (КЛБ) меняется с каждым годом. На заре своего существования КЛБ содержали две 3-входовых таблицы соответствия и один регистр. Позже в них стали включать две 4-входовых таблицы соответствия и два регистра. Теперь каждый блок содержит две или четыре секции, каждая из которых состоит из двух 4-входовых таблиц соответствия и двух регистров, а в недалёком будущем...

Используя конфигурационные логические блоки в качестве примера, заметим, что некоторые ПЛИС фирмы Xilinx содержат по две секции в каждом блоке, другие устройства по четыре секции в каждом блоке. Во время написания этой книги логические блоки визуально можно было представить в виде «островов» программируемой логики в «море» программируемых соединений (**Рис. 4.9**).

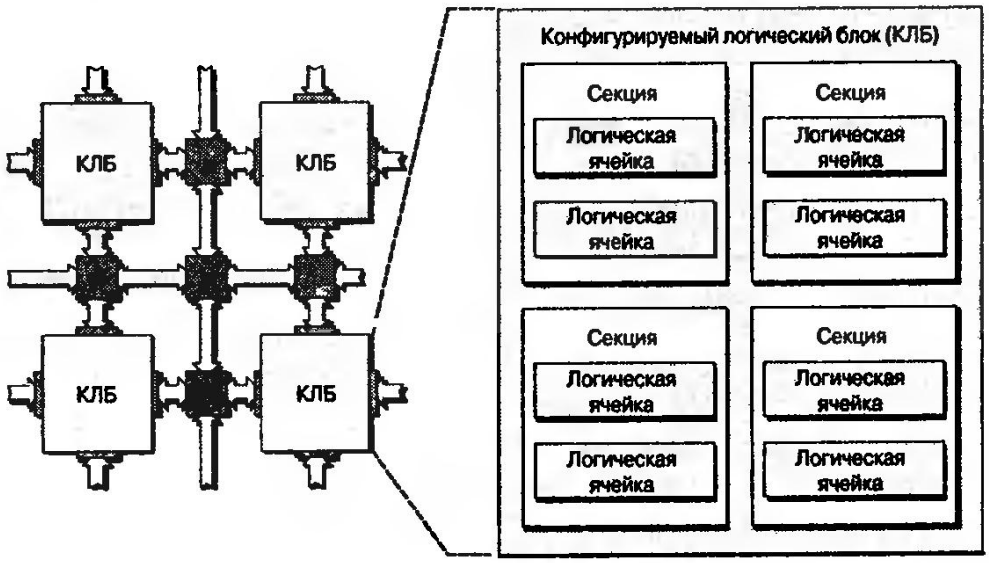


Рис. 4.9. Конфигурируемый логический блок, содержащий четыре секции (количество секций зависит от семейства ПЛИС)

Внутри логического блока находятся быстрые программируемые внутренние соединения. Эти проводники используются для соединения соседних секций (на **Рис. 4.9** для простоты изложения они не показаны).

Причина существования такой логико-блочной иерархии, т. е. логическая ячейка→ секция с двумя логическими ячейками→ конфигурируемый логический блок с четырьмя секциями, заключается в том, что она дополняется эквивалентной иерархией внутренних соедине-

ний. Другими словами, существуют быстрые внутренние соединения между логическими ячейками внутри секции, затем менее быстрые соединения между секциями внутри логического блока и соединения между блоками. Подобная иерархия отражает пошаговое достижени оптимального компромисса между простотой соединения внутренних структур и чрезмерными задержками сигнала на внутренних соединениях.

Распределенное ОЗУ и сдвиговые регистры

Как уже отмечалось, каждая 4-входовая таблица соответствия может использоваться в качестве блока ОЗУ 16×1. Если взять за основу четырёхсекционный конфигурируемый логический блок, показанный на **Рис. 4.9**, все таблицы соответствия внутри этого блока могут быть сконфигурированы для реализации следующих функций:

- Однопортовый блок ОЗУ 16×8 бит.
- Однопортовый блок ОЗУ 32×4 бита.
- Однопортовый блок ОЗУ 64×2 бита.
- Однопортовый блок ОЗУ 128×1 бит.
- Двухпортовый блок ОЗУ 16×4 бита.
- Двухпортовый блок ОЗУ 32×2 бита.
- Двухпортовый блок ОЗУ 64×1 бит.

Набор сигналов управления и данных, рассматриваемых как единое целое, обычно называют *портом*. В однопортовом ОЗУ данные записываются и считываются из ячейки через общую шину данных. В двухпортовом ОЗУ данные записываются и считываются через различные шины (порты). На практике в этом случае операции чтения и записи, как правило, работают со своими адресными шинами (используемыми для указания необходимой ячейки внутри ОЗУ), т. е. операции чтения и записи в двухпортовом ОЗУ могут выполняться одновременно.

Аналогично, каждая 4-битная таблица соответствий может использоваться в качестве 16-битного сдвигового регистра. Для этого существуют специальные соединения между логическими ячейками внутри секции и между секциями, которые позволяют соединить последний бит одного сдвигового регистра с первым битом другого регистра без привлечения к этому процессу выходных сигналов таблиц соответствия. Последние также могут использоваться для просмотра содержимого определенного бита в 16-битном регистре. Это позволяет при необходимости соединить вместе таблицы соответствия внутри одного логического блока и реализовать сдвиговый регистр величиной до 128 бит.

Схемы ускоренного переноса

Ключевой особенностью современных ПЛИС является то, что они содержат специальную логику и внутренние соединения, необходимые для реализации схем ускоренного переноса. В контексте программируемых логических блоков, рассмотренных в предыдущем разделе, следует упомянуть о том, что каждая логическая ячейка содержит специальную логику переноса. Эта логика дополняется специальными внутренними соединениями между двумя логическими ячейками в пределах каждой секции, между секциями в рамках каждого логического блока и между блоками.

Специальная логика быстрого переноса и выделенная маршрутизация способствуют выполнению логических функций, таких как

счетчики, и арифметических функций, таких как сумматоры. Возможности схем ускоренного переноса совместно с возможностями других средств, аналогичных сдвиговым регистрам на основе таблиц соответствия, встроенным умножителям и другим блокам обеспечивают необходимый набор средств для использования ПЛИС в приложениях цифровой обработки сигналов (ЦОС).

Встроенные блоки ОЗУ

В процессе реализации большинства приложений возникает необходимость использовать ячейки памяти, поэтому современные ПЛИС содержат довольно большие блоки встроенной памяти, называемые *блоками встроенного ОЗУ*. В зависимости от архитектуры микросхемы эти блоки могут быть расположены по периметру кристалла, разбросаны по его поверхности и относительно изолированы друг от друга или организованы в столбцы, как показано на **Рис. 4.10**.

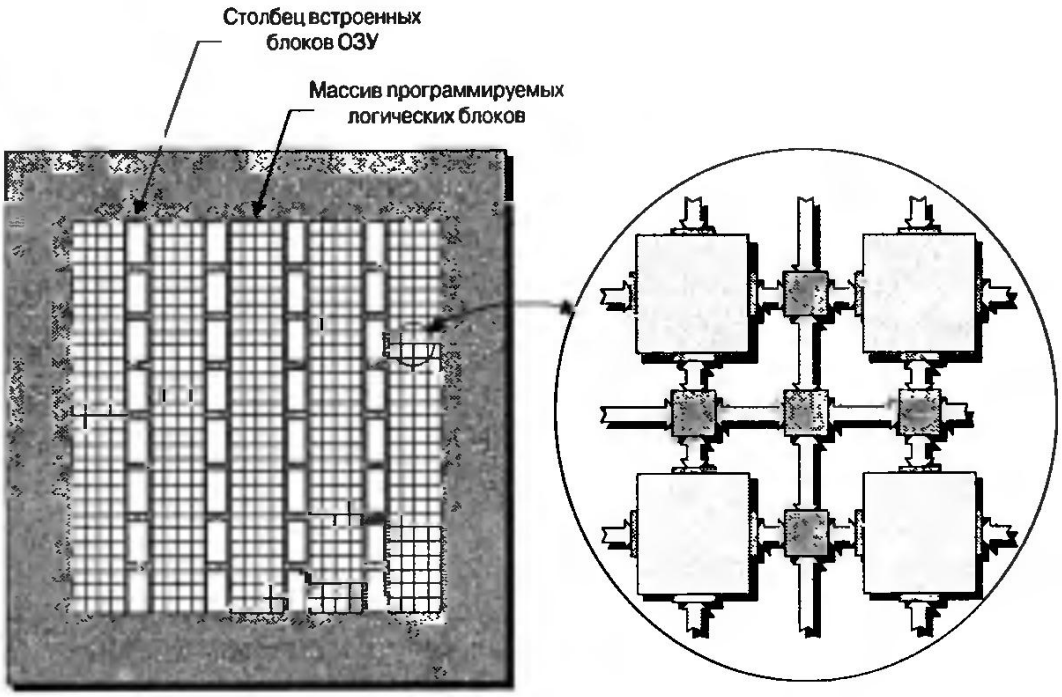


Рис. 4.10. **Вид на кристалл со столбцами встроенных блоков ОЗУ**

В зависимости от устройства размер блоков ОЗУ может меняться от нескольких тысяч до нескольких десятков тысяч бит. Каждая микросхема может содержать от нескольких десятков до нескольких сотен таких блоков. Таким образом, полная ёмкость простирается от нескольких сотен тысяч бит до нескольких миллионов бит.

Каждый блок ОЗУ может использоваться либо как независимое запоминающее устройство, либо находиться в связке с несколькими блоками для реализации массивов памяти большого объема. Блоки могут использоваться для различных целях, например, как стандартные одно- и двух-портовые блоки ОЗУ, *очереди FIFO (first-in first-out — первый пришёл, первый вышел)*, конечные автоматы и так далее.

Встроенные умножители, сумматоры, блоки умножения с накоплением и др.

Некоторые типы функций, например умножители, по своей сути являются довольно медленными, если их реализовывать с помощью большого количества программируемых логических блоков соединенных вместе. Поскольку эти функции используются в многочисленных

приложениях, многие ПЛИС содержат специальные аппаратные блоки умножения. Эти блоки обычно расположены в непосредственной близости от блоков встроенного ОЗУ, которые рассматривались в предыдущем разделе, так как они часто используются вместе (**Рис. 4.11**).

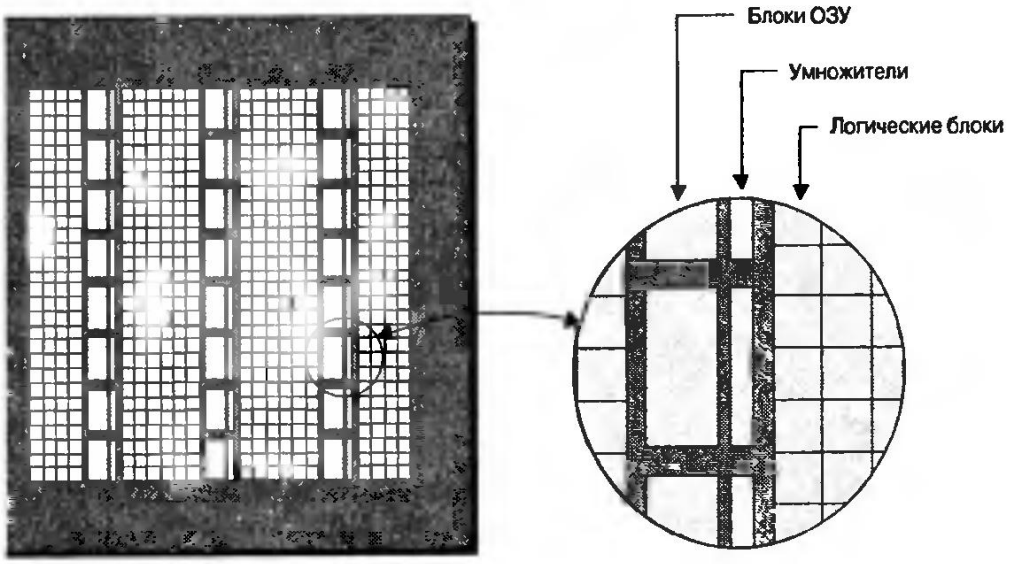


Рис. 4.11. Вид на кристалл со столбцами встроенных умножителей и блоков ОЗУ

Некоторые производители ПЛИС также предлагают выделенные сумматоры. В то же время, одной из самых распространенных операций, применяемых в приложениях цифровой обработки сигналов, является *умножение с накоплением* (*multiply-and-accumulate* или *MAC*), **Рис. 4.12**. Как подсказывает название, эта функция перемножает два числа и суммирует результат с текущим числом, сохранённым в аккумуляторе.

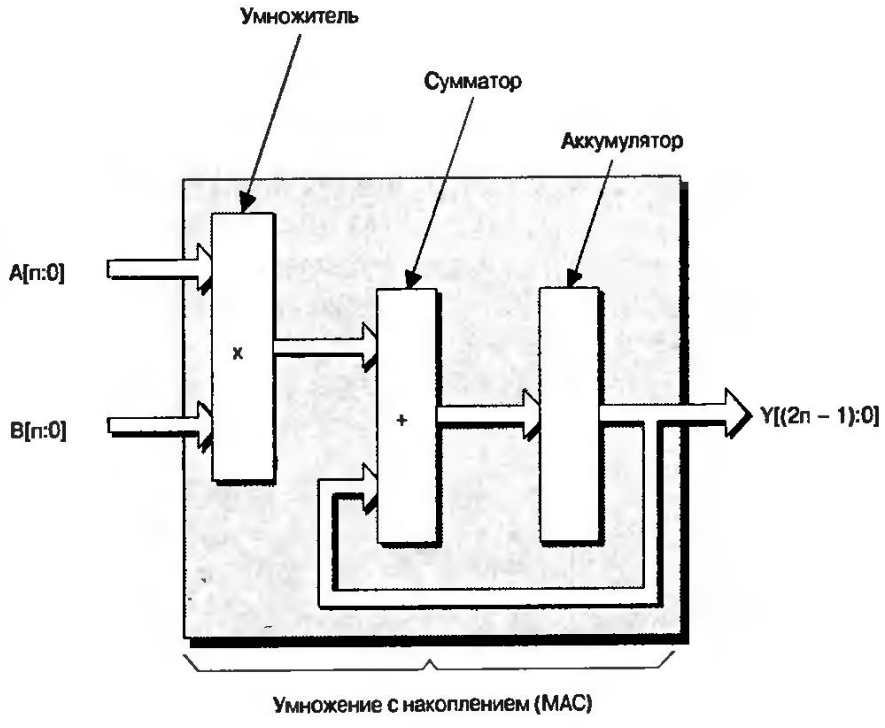


Рис. 4.12. Функции, формирующие операцию умножения с накоплением

При работе с ПЛИС, которая содержит только *встроенные умножители*, для реализации этой функции необходимо соединить умножитель с сумматором, сформированным из нескольких программируемых логических блоков. Результат будет сохраняться в триггерах логи-

ческих блоков, либо в блоках встроенного ОЗУ, либо в распределенном ОЗУ. Жизнь становится проще, когда ПЛИС уже содержат *встроенные сумматоры*, а отдельные их виды даже поддерживают *встроенные умножители с накоплением*.

1829 г. *Англия. Чарльз Уитстон (Sir Charles Wheatstone) разработал концертино — шестигранную гармонику.*

Аппаратные и программные встроенные микропроцессорные ядра

Важной особенностью ПЛИС является то, что в них почти все части электронного устройства могут быть реализованы аппаратно (с использованием логических вентилей, регистров и т. д.) или программно (в виде инструкций микропроцессора). Одним из главных частных критериев выбора между аппаратной и программной реализацией функции является время, за которое эта функция должна выполнять свои задачи:

- **Пикосекундная и наносекундная логика** — должна работать безумно быстро и реализуется аппаратно (в структуре ПЛИС).
- **Микросекундная логика** — умеренно быстрая и может реализовываться как аппаратно, так и программно. Тип логики, при котором основная масса времени тратится на определение того, каким путём пойти.
- **Миллисекундная логика** — используется при реализации интерфейсов, таких как опрос состояния переключателей или зажигание светодиодов. Основные усилия будут направлены на замедление аппаратной части при реализации этих функций; например, используя громаднейшие счетчики для генерации задержек. Таким образом, зачастую такие задачи лучше реализовывать в микропроцессорном коде, так как процессор позволяет снизить скорость по сравнению с аппаратной частью, но при этом задача может оказаться очень сложной.

Фактически большинство устройств в той или иной форме используют микропроцессоры. До последнего времени микропроцессоры представляли собой дискретные элементы, располагаемые на печатной плате. В последнее время появились высокотехнологичные ПЛИС, которые содержат один или несколько встроенных микропроцессоров, которые обычно называются *микропроцессорными ядрами*. При применении таких микросхем часто имеет смысл переложить все задачи, выполняемые внешним микропроцессором, на «плечи» встроенного ядра. Такой подход обеспечивает ряд преимуществ, не последними из которых будут снижение стоимости, устранение большого количества дорожек, посадочных мест и выводов на печатной плате, а также уменьшение размера и веса печатной платы.

Аппаратные микропроцессорные ядра

Аппаратные микропроцессорные ядра изготавливаются как отдельные предопределённые блоки. Существует два способа интеграции таких ядер в ПЛИС. Первый предусматривает расположение ядра в виде *полосы* (*полоса* — stripe) вдоль одной из сторон главной части, или главной структуры ПЛИС (**Рис. 4.13**).

При таком подходе все компоненты обычно формируются на одном кремниевом кристалле, хотя они могут быть выполнены и на двух кристаллах и размещены в виде *многокристального модуля (Multichip module — MCM)*. Главная часть ПЛИС также включает встроенные блоки ОЗУ, умножители и другие рассмотренные выше блоки, которые на этом рисунке не показаны с целью его упрощения.

1831 г. *Англия. Майкл Фарадей (Michael Faraday) создал первую электрическую динамомашину.*

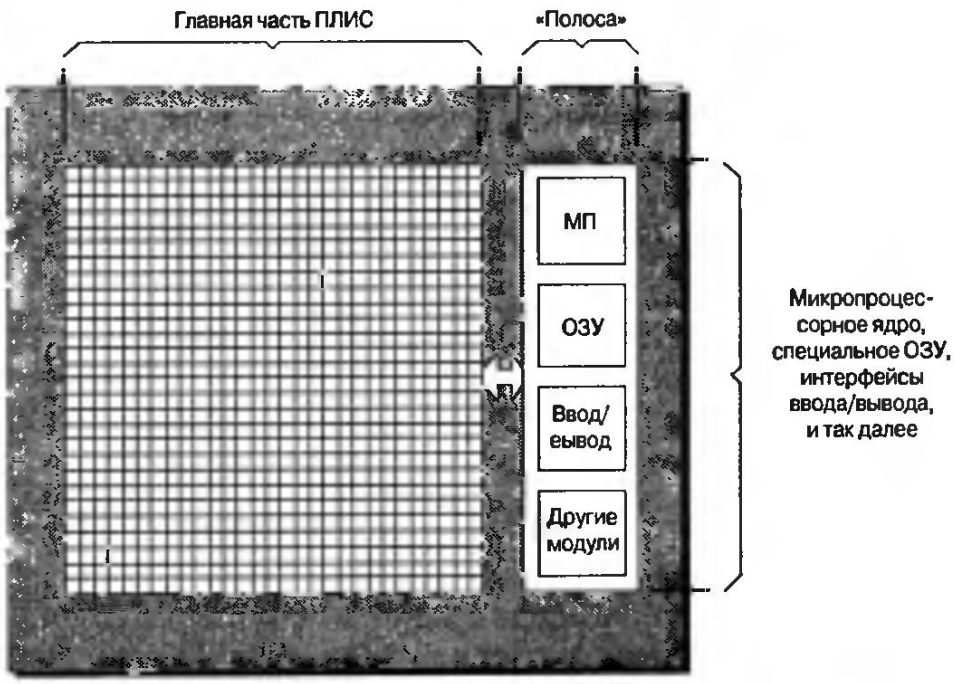
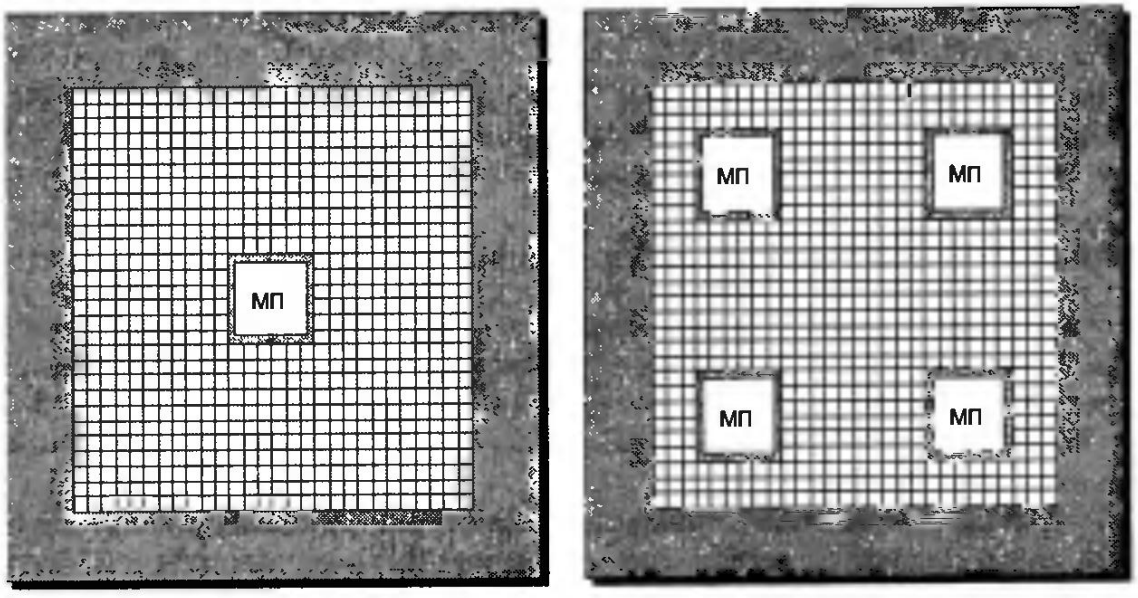


Рис. 4.13. Вид на кристалл со встроенным ядром, находящимся за пределами главной части

Преимущество такой реализации проявляется в главной части (структуре) ПЛИС, которая получается идентичной для устройств со встроенным и без встроенного микропроцессорного ядра. Это может существенно упростить работу инженеров со средствами разработки. Другое преимущество заключается в том, что поставщики ПЛИС могут связать все дополнительные функции, такие как память, устройства ввода-вывода и другие, в одну полосу для дополнения микропроцессорного ядра.

Одним из возможных решений является встраивание одного или более микропроцессорных ядер прямо в главную часть ПЛИС. Во время написания этих строк существовали микросхемы с одним, двумя и даже четырьмя ядрами (**Рис. 4.14**).

Хочу повториться и повторяюсь, главная часть ПЛИС включает встроенные блоки ОЗУ, умножители, и другие рассмотренные выше блоки, которые на этом рисунке не показаны с целью его упрощения.



а) Одно встроенное ядро ***б)*** Четыре встроенных ядра

Рис. 4.14. Вид на кристаллы с ядрами, встроенными внутрь главной части

При использовании этого способа используемые средства проектирования должны учитывать присутствие микропроцессоров в структуре микросхемы. Память, используемая ядром, формируется из встроенных блоков ОЗУ, а любые функции сопряжения реализуются с помощью групп программируемых логических блоков общего назначения. Сторонники этой схемы утверждают, что размещение микропроцессорного ядра в непосредственной близости к главной части (структуре) ПЛИС обеспечивает ей преимущество в скорости.

1831 г. *Англия. Майкл Фарадей (Michael Faraday) создал первый электрический трансформатор.*

Программные микропроцессорные ядра

Кроме физического встраивания микропроцессора в структуру кристалла, можно сконфигурировать группу программируемых логических блоков для работы в качестве микропроцессора. Такую группу блоков обычно называют *программным ядром*, но более точно они могут быть классифицированы как *программные* и *микропрограммные* в зависимости от способа, с помощью которого функциональность микропроцессора реализована логическими блоками. Программные ядра проще и медленнее, чем их аппаратные аналоги. Однако у них есть одно преимущество — при необходимости можно реализовать ядро или несколько ядер в том объеме, который можно достичь пока не будут исчерпаны все ресурсы в виде программируемых логических блоков.

Скорость работы программного ядра обычно составляет 30...50% от скорости аппаратного ядра.

Дерево синхронизации и диспетчеры синхронизации

Все синхронные элементы внутри ПЛИС, например регистры внутри программируемого логического блока, сконфигурированные для работы в виде триггеров, необходимо синхронизировать с помощью тактового сигнала. Тактовые сигналы обычно вырабатываются за пределами микросхемы и поступают в неё через специальные входы синхронизации, а затем распределяются через специальные устройства и подаются на соответствующие регистры.

Дерево синхронизации

Рассмотрим упрощенное изображение дерева синхронизации (**Рис. 4.15**, программируемые логические блоки не показаны).

Название *дерево синхронизации* возникло потому, что главный синхросигнал разветвляется подобно ветвям дерева, при этом триггеры могут рассматриваться как «листья» на концах веток. Такая структура вселяет уверенность в том, что все триггеры увидят свои тактовые сигналы одновременно, на сколько это возможно. Если бы тактовые сигналы распространялись по одному длинному проводнику, синхронизируя все триггеры поочередно, триггер, расположенный ближе к выводу синхронизации микросхемы увидел бы синхроимпульс намного раньше, чем последние триггеры в этой цепочке. Подобная ситуация называется *фазовым сдвигом*, и она порождает целый ряд проблем. Даже при использовании дерева синхронизации может возникнуть некоторый сдвиг фаз между регистрами, находящимися на одной ветви, а также между ветвями.

Дерево синхронизации реализуется с помощью специальных проводников, которые отделены от внутренних соединений общего назначения. Принцип действия, рассмотренный выше, на самом деле сильно упрощен. На практике в микросхемах существуют множество

1831 г. *Англия. Майкл Фарадей (Michael Faraday) открыл силовые линии магнитного поля.*

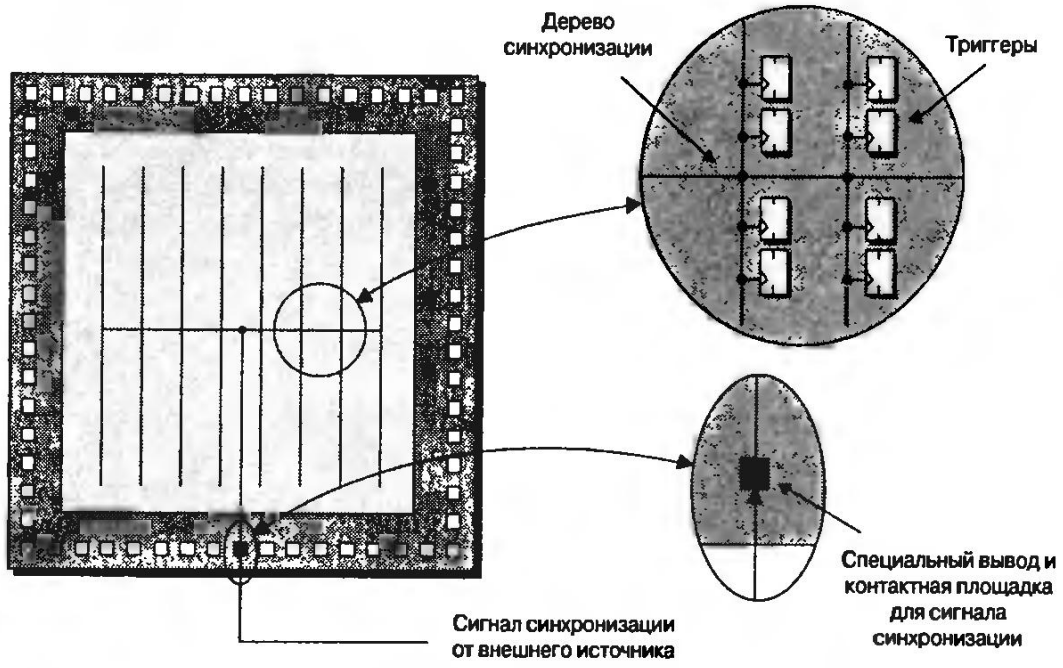


Рис. 4.15. Простое дерево синхронизации

выводов синхронизации (неиспользуемые выводы синхронизации могут быть использованы как выводы общего назначения), а внутри устройства имеются множественные домены синхронизации (деревья синхронизации).

Диспетчер синхронизации

Устройство управления синхронизацией (диспетчер синхронизации) по терминологии компании Xilinx называется DCM *(digital clock manager)*.

Вывод синхронизации микросхемы может быть подключен к дереву синхронизации. Однако, как правило, этот вывод подключают не напрямую к дереву синхронизации, а к входу устройства (или блока) управления синхронизацией, называемого *диспетчером синхронизации*, который генерирует *дочерние тактовые сигналы* (**Рис. 4.16**).

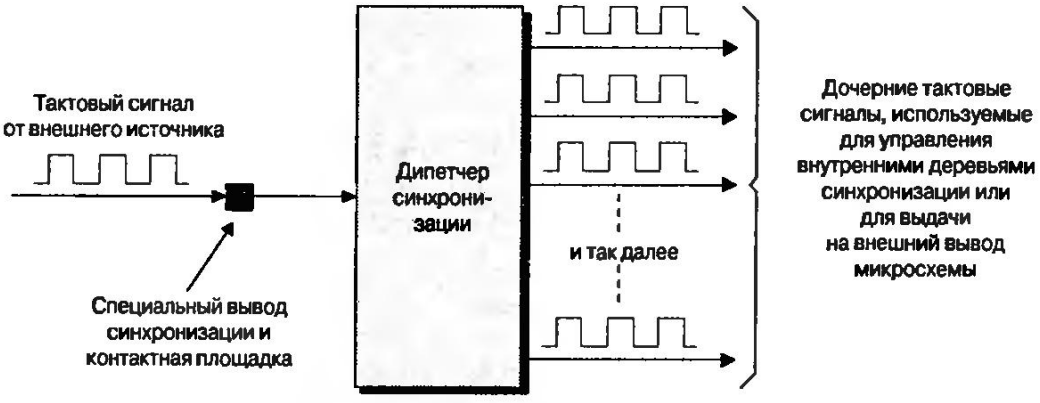


Рис. 4.16. Диспетчер синхронизации, генерирующий дочерние тактовые сигналы

Дочерние тактовые сигналы могут использоваться для управления внутренними деревьями (доменами) синхронизации или для выдачи сигналов на внешние выводы микросхемы, которые, в свою очередь, можно использовать для синхронизации других устройств, расположенных на печатной плате. Каждое семейство микросхем ПЛИС располагает собственным типом диспетчера синхронизации, и в одном устройстве могут находиться множество модулей диспетчера синхронизации. Различные диспетчеры синхронизации могут поддерживать все или только некоторые из следующих свойств.

Устранение флуктуаций. Для упрощения примера допустим, что сигнал синхронизации имеет частоту 1 МГц. На практике, конечно же, эта частота во много-много раз выше. В идеальном случае каждый фронт тактового сигнала от внешней системы синхронизации приходит ровно через одну миллионную долю секунды после предыдущего. Однако в реальной жизни фронт импульса может прийти немного раньше или немного позже.

Для визуализации этого эффекта, называемого *флуктуацией*, изобразим фронты импульсов один под другим, чтобы получить их суперпозицию; в результате может получиться смазанный тактовый сигнал (**Рис. 4.17**).

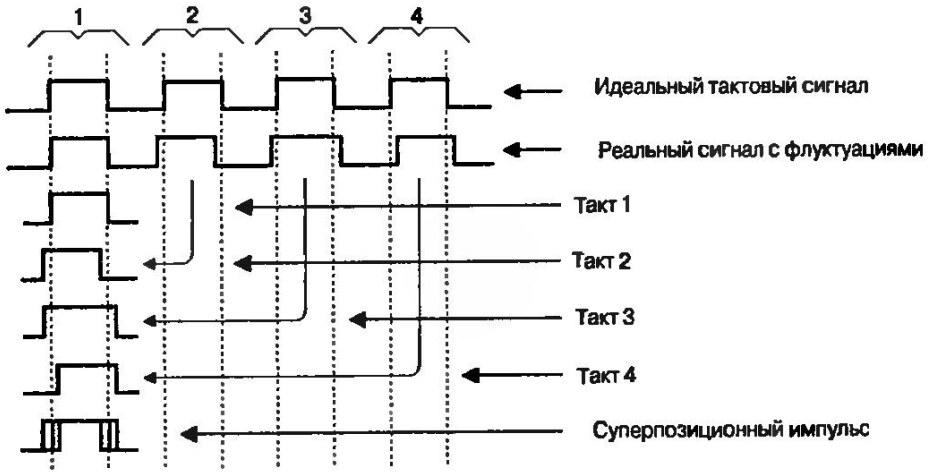


Рис. 4.17. Флуктуация как результат смазывания тактового сигнала

Диспетчер синхронизации ПЛИС может использоваться для обнаружения и коррекции подобных флуктуаций и обеспечивать очистку дочерних тактовых сигналов для их использования внутри устройства (**Рис. 4.18**).

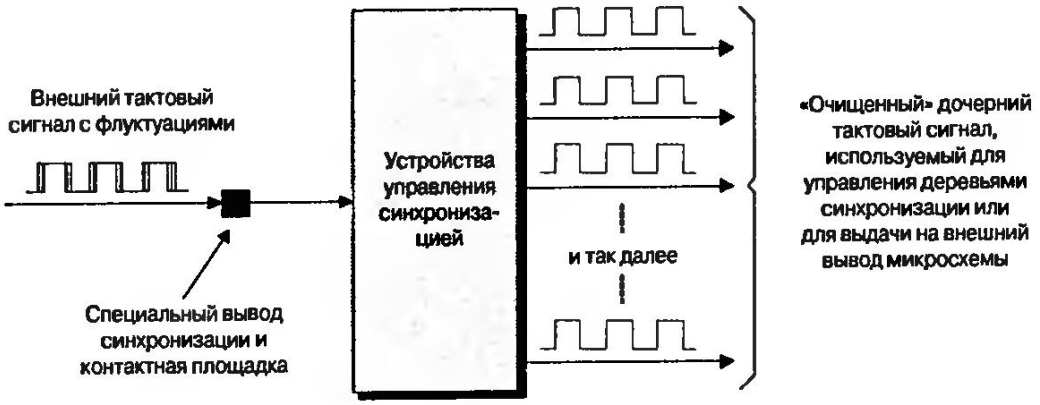


Рис. 4.18. Диспетчер синхронизации может убрать флуктуацию тактового сигнала

Частотный синтез. Может случиться так, что частота тактового сигнала, поступающего на ПЛИС от внешнего источника синхронизации, не соответствует частоте, необходимой для решения поставленных задач. В этом случае диспетчер синхронизации может генерировать дочерние тактовые сигналы, частота которых является производной величиной от исходного сигнала и получается путем его деления или умножения.

Единица измерения герц (Гц) названа в честь Генриха Герца (Heinrich Hertz), немецкого физика из университета Карлсруэ (Karlsruhe) в Германии, который впервые в 1888 г. осуществил передачу и приём радиоволн в лабораторных условиях. Один герц (1 Гц) равен одному колебанию в секунду, 1 мегагерц (1 МГц) равен одному миллиону Герц.

В качестве простого примера рассмотрим три дочерних тактовых сигнала: первый с частотой, эквивалентной исходной частоте тактового сигнала, второй с частотой, равной частоте исходной последовательности умноженной на два, и третий с частотой, равной половине частоты исходного сигнала (**Рис. 4.19**).

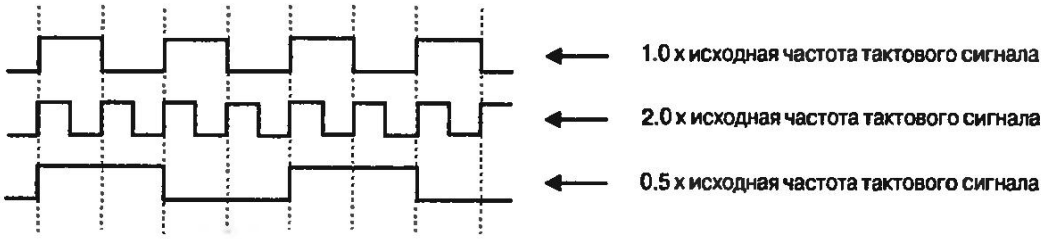


Рис. 4.19. Использование диспетчера синхронизации для частотного синтеза

И еще раз повторюсь, что на **Рис. 4.19** представлен очень простой пример. На практике ПЛИС могут синтезировать все типы внутренних тактовых сигналов, например четыре пятых от частоты первоначальной последовательности тактового сигнала.

Фазовый сдвиг. Некоторые изделия требуют использования тактовых сигналов, фаза в которых сдвинута (задержана) относительно друг друга. Некоторые диспетчеры синхронизации позволяют выбирать общие фиксированные значения фазовых сдвигов, например 120° и 240° для трехфазной системы синхронизации или 90°, 180° и 270° для четырёхфазной системе синхронизации. Другие диспетчеры позволяют конфигурировать точное значение фазового сдвига по требованию пользователя для каждого дочернего тактового сигнала.

Предположим, что мы получаем четыре внутренних тактовых сигнала из главной последовательности тактовых сигналов, где первый находится в фазе с исходным сигналом, второй сдвинут по фазе на 90°, третий на 180° и т. д. (**Рис. 4.20**).

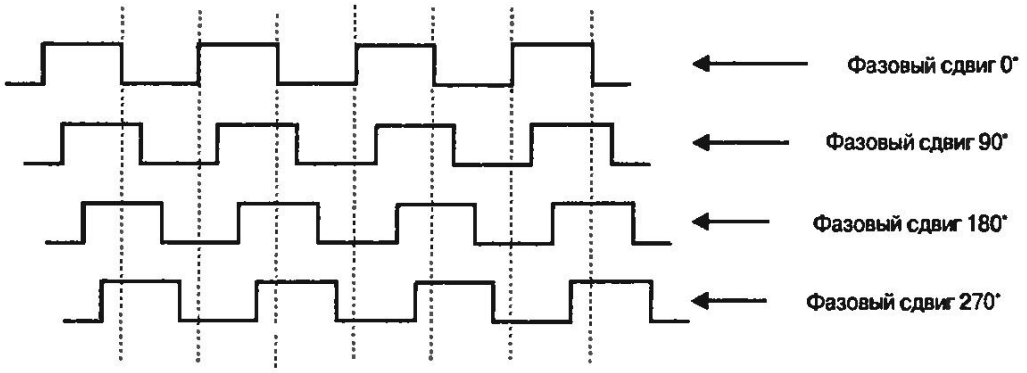


Рис. 4.20. Использование диспетчера синхронизации для фазового сдвига дочерних тактовых сигналов

Автокоррекция сдвига фаз. В целях упрощения предположим, что речь идет о дочерних тактовых сигналах, сформированных на той же частоте и с той же фазой, что и главный тактовый сигнал, приходящий на вход ПЛИС. Однако диспетчер синхронизации, по определению, будет добавлять к сигналам некоторую задержку. Кроме того, еще большие задержки добавляют логические вентили и внутренние соединения, используемые в распределении тактовых сигналов. Итак, если не предпринять корректирующих мер, дочерние тактовые сигналы будут отставать от входных тактовых сигналов на некоторую величину. Напомню, что разница между фазами двух сигналов называется *фазовым сдвигом*.

В зависимости от того, как главные и дочерние тактовые сигналы используются ПЛИС и остальными элементами печатной платы, сдвиг фаз может стать причиной различных проблем. Поэтому диспетчер синхронизации может содержать специальный вход для подачи на него дочернего тактового сигнала. В этом случае диспетчер синхронизации сравнивает два сигнала и точно добавляет определенную задержку к дочерним сигналам, достаточную для выравнивания их с исходным тактовым сигналом (**Рис. 4.21**).

Таким способом будет очищен только первичный, т. е. с нулевым фазовым сдвигом, дочерний сигнал, а все другие дочерние сигналы будут фазированы уже относительно этого сигнала.

1831 г. *Англия. Майкл Фарадей (Michael Faraday) открыл принцип электромагнитной индукции.*

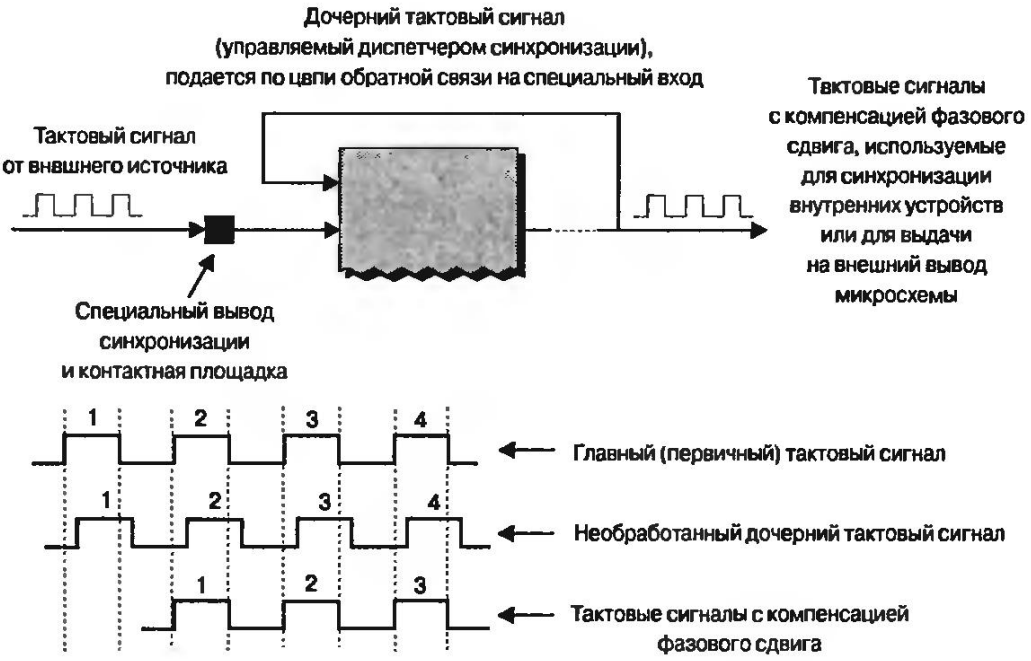


Рис. 4.21. Компенсация фазового сдвига относительно первичного тактового сигнала

Некоторые диспетчеры синхронизации ПЛИС работают по принципу *фазовой автоподстройки частоты* (ФАПЧ), другие по принципу *цифровой автоподстройки по задержке (digital delay-locked loop — DLL)*. С 40-х гг. ФАПЧ используются в аналоговых устройствах, но сегодня в связи с широким внедрением цифровых методов желанной становится цифровая автоподстройка. Системы ФАПЧ могут быть реализованы аналоговым или цифровым способом, а системы цифровой автоподстройки по задержке по своей природе могут быть только цифровыми. Сторонники ФАПЧ утверждают, что эти системы имеют преимущества в точности, стабильности и потребляемой мощности, нечувствительны к шумам и флуктуациям.

Ввод/вывод общего назначения

Современные ПЛИС могут содержать 1000 и более выводов, которые располагаются по всему корпусу микросхемы. Аналогично они подходят к кристаллу внутри корпуса микросхемы. Внутрь корпуса кристалл устанавливают в перевернутом виде. Это позволяет подсоединять общий провод, проводники питания, синхронизации и сигналов ввода/вывода к любой точке на его поверхности. Однако в рамках наших рассуждений, предположим, что все выводы на кристалле расположены по его контуру, как и было на самом деле много лет назад.

1831 г. *Англия. Майкл Фарадей (Michael Faraday) открыл, что переменное магнитное поле вызывает электрический ток.*

Конфигурируемые стандарты ввода/вывода

Давайте посмотрим на электронное устройство с точки зрения инженеров, разрабатывающих печатную плату. В зависимости от того что они делают, от типов используемых устройств, от окружающей среды, в которой будет работать плата и так далее, разработчики будут выбирать определенный стандарт передачи данных. (В этом контексте понятие «стандарт» относится к электрическим параметрам сигналов, таким как уровень логического 0 или логической 1 в вольтах.)

Проблема в том, что существует большое разнообразие стандартов, и было бы трудно создавать специальные ПЛИС для поддержки каждого варианта. По этой причине *ввод/вывод общего назначения* ПЛИС может быть сконфигурирован для приема и передачи сигналов, соответствующих любому стандарту. Эти сигналы ввода/вывода разделяются на несколько банков. Будем полагать, что существует восемь банков, пронумерованных от 0 до 7 (**Рис. 4.22**).

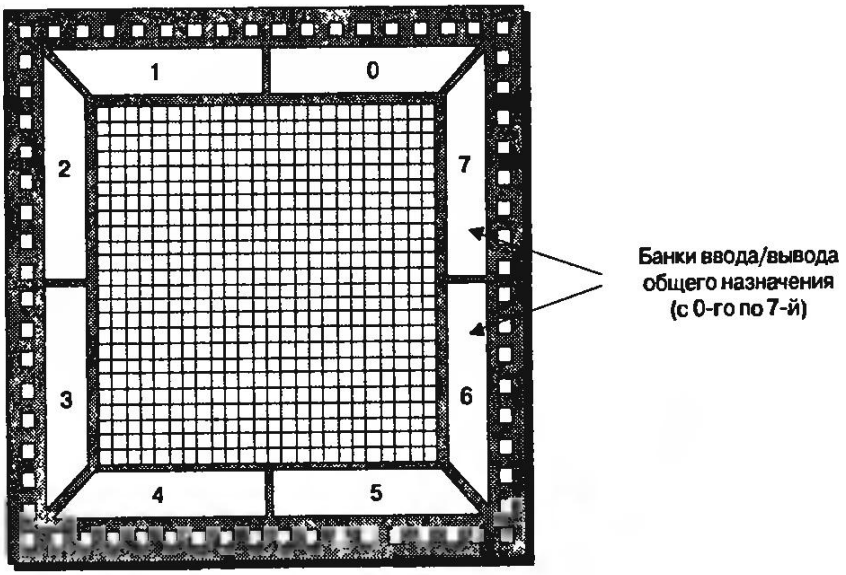


Рис. 4.22. Вид на кристалл с банками ввода/вывода

Интересно, что каждый банк может быть индивидуально сконфигурирован для поддержки определенных стандартов ввода/вывода. Таким образом, мало того, что ПЛИС имеет возможность работать с устройствами, используя многочисленные стандарты ввода/вывода, эти микросхемы могут служить интерфейсом между различными стандартами ввода/вывода, а также осуществлять связь между различными протоколами, которые могут основываться на частных электрических стандартах.

Согласование ввода/вывода

Сигналы в современных печатных платах часто обладают быстрым *временем переключения*. Имеется в виду время, которое необходимо сигналу для переключения с одного логического уровня на другой. Чтобы предотвратить отражение сигналов, приводящее к возникновению пульсаций, к выводам микросхемы, т. е. к входу или выходу, необходимо подключить соответствующие согласующие резисторы, т. е. терминаторы.

В прошлом эти резисторы применялись как отдельные компоненты, которые размещались на печатной плате за пределами ПЛИС. Однако такой подход становился всё более проблематичным, поскольку

количество выводов у микросхемы увеличивалось, а шаг, т. е. расстояние между ними, уменьшался. В связи с этим современные ПЛИС позволяют использовать внутренние согласующие резисторы, или внутренние терминаторы, сопротивление которых может задаваться пользователем, для согласования различного оборудования на печатной плате и различных стандартов ввода/вывода.

1832 г. *Англия. Чарльз Бэббидж (Charles Babbage) создал первый механический компьютер, так называемую аналитическую машину.*

Напряжение ядра и напряжение ввода/вывода

В былые времена, где-то с 1965 по 1995 год, большинство цифровых микросхем использовало напряжение земли 0 В и напряжение питания +5 В. Кроме того, сигналы ввода/вывода переключались между уровнями 0 В (логический 0) и +5 В (логическая 1), что значительно упрощало жизнь.

Со временем размеры структур на кремниевых кристаллах становились меньше, поскольку транзисторы меньших размеров были дешевле, а также быстрее и потребляли меньше энергии. Однако для этого требовалось понизить напряжения питания. Напряжение питания продолжало падать с каждым годом (**Табл. 4.2**).

Таблица 4.2. **Зависимость напряжения питания от технологического процесса**

Год	Источник питания (напряжение ядра) [В]	Технологический процесс [нм]
1998	3.3	350
1999	2.5	250
2000	1.8	180
2001	1.5	150
2003	1.2	130

Источник питания, указанный в таблице, используется для питания внутренней логики ПЛИС, поэтому это напряжение называется *напряжением ядра* (на самом деле для подключения питания и «земли» задействуются несколько выводов микросхемы). Различные стандарты ввода/вывода используют сигналы с напряжениями, уровни которых значительно отличаются от напряжения ядра, поэтому каждый банк ввода/вывода данных может иметь свой дополнительный вывод питания.

Интересно, что, начиная с 350 нм процесса, напряжение ядра изменялось прямо пропорционально технологическому процессу. Однако существуют физические причины, не позволяющие опуститься значительно ниже 1 вольта (эти причины основаны на технологических аспектах, таких как входной порог переключения транзисторов и перепад напряжений), поэтому эта «лестница напряжений» в недалеком будущем может закончиться.

Гигабитные приёмопередатчики

Традиционный способ передачи больших объемов данных предусматривает использование *шины*, которая представляет собой набор сигналов, передающих данные и выполняющие некоторые общие функции (**Рис. 4.23**).

Примерно с 1975 г. для передачи информации в микропроцессорных системах стали использоваться 8-битные шины. С ростом потребности в передачи большего объема данных и увеличении скорости их

1832 г. *Англия. Джозеф Генри (Joseph Henry) открыл явление самоиндукции, или индуктивности.*

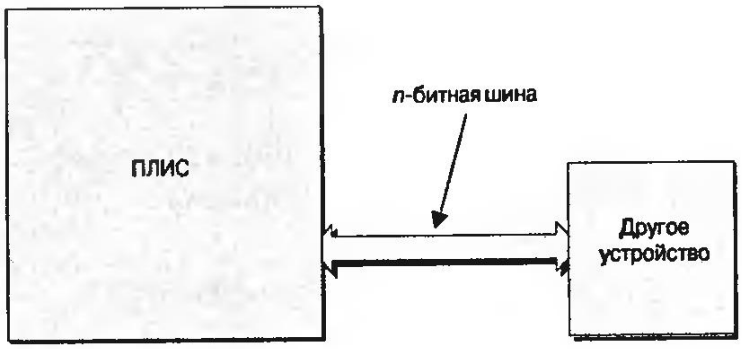


Рис. 4.23. Использование шины для связи устройств

передачи разрядность шин выросла до 16 бит, затем до 32 бит, затем до 64 бит и так далее. Проблема заключается в том, что такой рост требовал большого количества выводов в микросхеме и большого количества дорожек на печатной плате для соединения компонентов. Прокладка дорожек таким образом, чтобы дорожки были одинаковой длины и имели одно и то же сопротивление, становится всё более трудной задачей по мере роста сложности печатных плат. Кроме того, усложнялось управление параметрами целостности сигналов, такими, как чувствительность к шумам, особенно при наличии большого количества проводников в шине

По этой причине современные высокотехнологичные ПЛИС снабжены встроенными блоками гигабитных приёмопередатчиков. Эти блоки используют одну пару дифференциальных сигналов для *передачи (TX)* и вторую пару для *приема (RX)* данных **(Рис. 4.24)**. Это значит, что пара сигналов всегда несет противоположные логические значения.

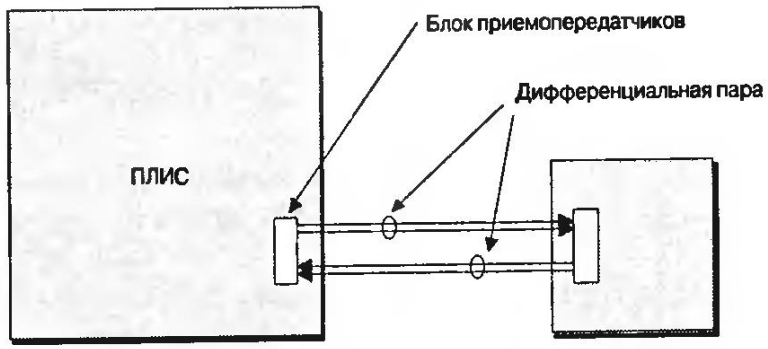


Рис. 4.24. Использование высокоскоростных приёмопередатчиков для связи устройств

Эти приёмопередатчики работают на невероятно высоких скоростях, которые позволяют им передавать и принимать миллиарды бит данных в секунду. Кроме того, каждый блок практически поддерживает несколько, скажем четыре, таких передатчиков, и каждая ПЛИС может содержать несколько таких передающих блоков.

IP — блоки аппаратной, программной и микропрограммной интеллектуальной собственности

Каждый поставщик ПЛИС предлагает собственный набор *аппаратных, микропрограммных* и *программных блоков интеллектуальной собственности (IP — intellectual property)*. Аппаратные блоки интеллектуальной собственности (аппаратные IP) представляют собой уже реа-

1833 г. *Англия. Майкл Фарадей (Michael Faraday) сформулировал законы электролиза.*

лизованные блоки, такие как микропроцессорные ядра, гигабитные интерфейсы, умножители, сумматоры, функции умножения с накоплением и им подобные. Эти блоки разрабатываются так, чтобы они были максимально эффективны и с точки зрения потребляемой мощности, и с точки зрения производительности, и с точки зрения площади, занимаемой на кристалле. Для каждого семейства ПЛИС характерны различные комбинации и различное количество таких блоков.

Программные блоки интеллектуальной собственности (программные IP) представляют собой библиотеки исходных кодов высокоуровневых функций, которые могут использовать разработчики конечных устройств. Эти функции выражаются языком описания аппаратных средств *HDL (HDL — hardware description language)*, таким как Verilog или VHDL, терминами *уровня регистровых передач (RTL — register transfer level)*. Все функции программных блоков интеллектуальной собственности, используемые инженерами, являются частью основного устройства, которое также описывается в терминах RTL, и затем синтезируется в группу программируемых логических блоков, возможно в комбинации с некоторыми аппаратными блоками интеллектуальной собственности, такими как умножители и другими.

Среднее положение между аппаратными и программными занимают *микропрограммные блоки интеллектуальной собственности* (микропрограммные IP — firm IP), которые также реализуются в виде библиотеки высокоуровневых функций. В отличие от своих программных эквивалентов эти функции уже оптимально преобразованы, размещены и объединены в группы программируемых логических блоков (возможно в комбинации с некоторыми аппаратными блоками интеллектуальной собственности, такими как, например, умножители). При необходимости, в конструкцию микросхемы могут быть включены несколько предопределенных блоков микропрограммной IP.

Проблема состоит в том, что довольно трудно провести границу между функциями, которые лучше реализовывать в виде аппаратных IP, и функциями, которые следует реализовывать как программные или микропрограммные блоки интеллектуальной собственности, используя набор программируемых логических блоков общего назначения. Что касается умножителей, сумматоров, и функций умножения с накоплением, которые уже рассматривались в этой главе, они используются в большинстве практических приложений и будут всегда востребованы. Вместе с тем, некоторые ПЛИС содержат специализированные блоки для управления специфичными интерфейсными протоколами, например, стандарт PCI. Это, конечно, может существенно облегчить жизнь пользователя, но при условии, что на печатной плате найдется интерфейс, с которым он пожелаете соединить своё устройство. При необходимости использовать другой интерфейс специализированный PCI-блок окажется только растратой свободного места, и будет затруднять передачу данных и бездумно потреблять энергию.

Поэтому, если поставщики ПЛИС добавляют в свои микросхемы подобные специфичные функции, это свидетельствует о том, что они уже определили для этого компонента его нишу. Иногда потребители вынуждены использовать подобные функции для достижения желаемых показателей или параметров разрабатываемых систем, что является классической проблемой, так как следующее поколение устройств очень часто способно достигать необходимых показателей в своей главной (программируемой) структуре без специфичных блоков.

Системный вентиль и реальный вентиль

Одним из показателей, используемых для измерения размера устройства в мире заказных микросхем, является *эквивалентный вентиль*. Дело в том, что различные поставщики предоставляют различные библиотеки своих функций, причем реализация каждой функции требует различного количества транзисторов. Это затрудняет сравнение относительной ёмкости и сложности двух устройств.

Решение заключается в назначении каждой функции устройства эквивалентного количества вентилей по принципу: «Функция А считается равной пяти эквивалентным вентилям; функция Б считается равной трем эквивалентным вентилям...». На следующем этапе необходимо подсчитать количество экземпляров каждой функции, преобразовать его в значение в эквивалентных вентилях, суммировать все значения вместе и с гордостью заявить: «Моя микросхема содержит 10 миллионов эквивалентных вентилей, что существенно больше, чем в вашей микросхеме!»

К сожалению, не всё так просто, так как определение, что действительно представляет собой эквивалентный вентиль, существенно зависит от того, кто об этом говорит. Имеется договоренность, согласно которой 2-входовая функция И-НЕ представляется собой один эквивалентный вентиль. Между тем, некоторые поставщики эквивалентный вентиль приравнивают к некоторому количеству транзисторов. И ещё более специфичная договоренность, понятная только посвященным, определяет эквивалентный вентиль эмиттерно-связанной логики: «одна одиннадцатая минимального количества логики, требуемого для реализации полного однобитного сумматора» (кому же это пришло в голову?). В подобной ситуации перед реализацией каких-либо планов за кровно заработанные деньги следует убедиться, что все вкладывают один и тот же смысл в обсуждаемые понятия.

Вернемся к ПЛИС. Проблема, с которой постоянно сталкиваются поставщики ПЛИС, возникает при попытке сравнить их устройство с заказной интегральной микросхемой. Например, кто-то располагает устройством на заказной микросхеме с 500000 эквивалентных вентилей и желает перенести его на микросхему ПЛИС. Как в этом случае определить, можно ли эту разработку реализовать на конкретной ПЛИС? Тот факт, что каждая 4-входовая таблица соответствия может использоваться в виде блока, содержащего от одного до 20 и более 2-входовых примитивных вентилей, делает такое сравнение довольно сложным.

Чтобы устранить эти разногласия, в начале 90-х поставщики ПЛИС начали поговаривать о *системном вентиле*. Некоторые расценивают это как благородное желание использовать терминологию, понятную разработчикам заказных микросхем, другие утверждают, что это был маркетинговый ход, не нашедший соответствующей поддержки.

К сожалению, как оказалось, в то время не существовало точного определения системного вентиля. Ситуация осложнялась тем, что тогда ПЛИС в основном состояли из программируемой логики общего назначения в форме таблиц соответствия и регистров. Даже нельзя было получить ответ на такой вопрос, как: «Можно ли впихнуть функциональность, реализованную на определенной заказной микросхеме, содержащей *x* эквивалентных вентилей, в некоторую ПЛИС, содержащую *y* системных вентилей?». Проблема состояла в том, что некоторые устройства на заказных микросхемах были, как правило, комбинационными, в то же время как на других устройствах было довольно тяже-

ло использовать регистры. И в том, и в другом случае перенос устройства на ПЛИС не был в полной мере оптимальным.

Проблема обострилась, когда ПЛИС стали содержать встроенные блоки ОЗУ, так как некоторые функции могли быть более эффективно реализованы в памяти, чем с помощью логики общего назначения. А тот факт, что таблица соответствия могла работать как распределённое ОЗУ, ещё больше замутил воду. Так, например, по утверждению одного из поставщиков: «Предположительно, от 20 до 30% таблиц соответствия используется в качестве распределенных ОЗУ». При рассмотрении ПЛИС со встроенными процессорными ядрами и подобными функциями ситуация усугубляется до такой степени, что некоторые поставщики говорят: «Количество системных вентилей для этих устройств не определено».

Существуют ли какие-нибудь практические приемы для перевода системных вентилей в эквивалентные вентили и наоборот? Конечно таких способов очень много. Некоторые говорят, если вы оптимист, вам следует разделить количество системных вентилей на 3. В этом случае, например, 3 000000 системных вентилей ПЛИС-устройств будут соответствовать 1 000000 эквивалентных вентилей заказных интегральных микросхем. Если же вы скорее пессимист, чем оптимист, можете разделить количество системных вентилей на 5. В этом случае 3 000000 системных вентилей будут соответствовать 600000 эквивалентных вентилей.

Однако кое-кто может сказать, что изложенное выше справедливо, если допустить, что число системных логических элементов включает все возможные функции, которые могут быть реализованы при использовании как программируемой логики общего назначения, так и блоков ОЗУ. Этот же кое-кто будет утверждать, что при удалении из расчетов блоков ОЗУ, следует делить число системных вентилей на 10. В этом случае 3 000000 миллионам системных вентилей будут соответствовать всего лишь 300000 эквивалентных вентилей), но в этом случае блок ОЗУ по-прежнему будет играть свою роль... Брррррррррр!

В конечном счете, эта тема становится сложной настолько, что даже поставщики ПЛИС отчаянно пытаются не заводить разговоров о системных вентилях. Когда ПЛИС только появились, дискомфорта по отношению к эквивалентным вентилям не существовало, чего нельзя сказать о таблицах соответствия, секциях и тому подобному. Однако спустя годы, широкое распространение ПЛИС привело к тому, что сегодня разработчики более свободно ориентируются и в этих понятиях. По этой причине я предпочел бы определять и сравнивать ПЛИС по таким показателям:

- количество логических ячеек или логических элементов или других элементов (которые эквивалентны 4-входовым таблицам соответствия и связанными с ними триггерами/защелками);
- количество (и размер) встроенных блоков ОЗУ;
- количество (и размер) встроенных умножителей;
- количество (и размер) встроенных сумматоров;
- количество (и размер) встроенных блоков операций умножения с накоплением;
- и т. д.

Почему же все так сложно? На самом деле было бы очень полезно взять другой набор реально существующих заказных интегральных микросхем и определить значения эквивалентных вентилей с учетом триггеров или защёлок, примитивных логических элементов и других более сложных функций. Затем соотнести каждое из этих устройств с

1837 г. *Америка. Самуил Финли Бриз Морзе (Samuel Finley Breese Morse) изобрёл электромеханический телеграфный аппарат.*

1837 г. *Англия. Чарльз Уитстон (Sir Charles Wheatstone) совместно с Вильямом Куком (Sir William Fothergill Cooke) запатентовали 5-стрелочный электрический телеграф.*

1842 г. *Англия. Джозеф Генри (Joseph Henry) открыл явление, согласно которому электрическая искра между двумя проводниками вызывает намагничивание стальных игл. Этот эффект обнаруживался на расстоянии 30 метров.*

1842 г. *Шотландия. Александр Бейл (Alexander Bail) продемонстрировал первое электромеханическое средство, позволяющее вводить, передавать и воспроизводить изображения.*

количеством таблиц соответствия, триггеров/защелок, необходимых для реализации эквивалентного устройства на базе ПЛИС с учетом количества встроенных блоков памяти ОЗУ и ряда других встроенных функций.

Конечно, это далеко не идеальное решение, поскольку общая тенденция в проектировании имеет различные подходы для ПЛИС и заказных интегральных микросхем, но, по крайней мере, надо же с чего-то начать.

Возраст ПЛИС

Все мы знаем, что один год жизни собаки соответствует семи годам человеческой жизни. Следовательно, 10-летняя дворняга, в пересчете на человеческий возраст, это 70-летний человек. Это, разумеется, ничего не меняет. Но, тем не менее, этот пример демонстрирует прекрасную систему отсчета. Так, если во время продолжительной прогулки гончая не будет поспевать за хозяином, вполне справедливо будет сказать: «Этого стоило ожидать, ведь тебе, дружище, уже почти 100 лет». Или что-то в этом роде.

В случае с ПЛИС рассмотренная ситуация может натолкнуть на мысль, что один год для этих устройств соответствует 15 годам человеческой жизни. Поэтому, если пользователь имеет дело с микросхемой, которая появилась на рынке в прошлом году, её можно сравнивать с подростком. Если имеются большие виды на будущее, возможно, в будущем он или она станет лауреатом Нобелевской премии мира или займет пост президента Соединенных Штатов Америки. Или же объект вашей любви будет, как обычно, обладать некоторыми капризами, к которым придется привыкать и учиться их использовать.

Если ПЛИС присутствует на рынке в течение 2-х лет, что соответствует 30 годам в пересчете на человеческий возраст, можно воспринимать ее как достаточно зрелого и всесторонне одаренного человека, который находится в самом расцвете сил. После трехлетнего возраста (45 лет) ПЛИС становится уравновешенным человеком средних лет, а через четыре года (60-летний возраст) следует относиться к ней с уважением, а не как к ломовой лошади!