# ГЛАВА 2

# Основы технологии проектирования ИСПС и структуры систем автоматизированного проектирования

Задача данной книги — помочь в освоении современных подходов к проектированию систем, ориентированных на широкое использование в них новой элементной базы — микросхем с программируемой структурой (ИСПС) и их наиболее ярких представителей — конфигурируемых систем на кристалле (SOPC).

Краеугольными составляющими любого проектирования являются:

- элементы, из которых строится проектируемый объект;
- инструментарий, при помощи которого выполняются отдельные этапы изготовления документации и собственно результирующего объекта;
- методика применения инструментария для получения конечных результатов;
- абстракции (определения, понятия и т. д.), вводимые для упрощения описания процесса проектирования.

Строительными "кирпичиками" интересующих нас систем являются рассмотренные в *гл. 1* элементы с конфигурируемой структурой (по сути, представляющие собой конфигурируемую аппаратуру, для которой в английской терминологии обычно используется термин HardWare, HW) и программируемым поведением (в основе которого лежит программное обеспечение, в английской терминологии SoftWare, SW). Термины HW и SW, благодаря своей краткости (при сохранении однозначности трактовки), получили широкое распространение в отечественной литературе и будут применяться в последующих разделах. Основными абстрактными понятиями, которыми приходится пользоваться в процессе проектирования, являются языки опи-

сания аппаратуры, рассматриваемые в *гл. 3*, и средства описания программной части проектируемых систем (универсальные языки программирования), которые достаточно подробно разобраны в существующей литературе. Инструментарий проектирования и методика его использования составляют содержание данной главы. Основным инструментом проектирования для рассматриваемой прикладной области (проектирование на основе ИСПС) является программное обеспечение ЭВМ, в комплексе образующее системы автоматического проектирования, САПР. Наиболее значимым для проектирования на основе ИСПС (и отличающим его от многих других) является не столько резкое увеличение возможностей, предоставляемых современными САПР, сколько взаимопроникновение и взаимное влияние составляющих процесса проектирования.

Стремительное увеличение степени интеграции современной элементной базы и постоянное улучшение характеристик ЭВМ (не в последнюю очередь благодаря этой элементной базе) вызывает адекватное наращивание как количественных, так и качественных возможностей САПР, т. е. инструментария проектирования. Динамика изменения возможностей САПР предопределяет изменение методологии их применения и все большую интеграцию в рамках одной САПР смешанных методик и объектов проектирования. Темпы изменений всех составляющих процесса проектирования электронных систем заставляют периодически пересматривать и переосмысливать различные аспекты проектирования. Материал данной главы посвящен рассмотрению возможностей и особенностей использования современных методик проектирования цифровых устройств с ориентацией на существующие САПР.

В области структурной организации схем с программируемой структурой и средств их проектирования состояние весьма подвижно. Если базовые архитектурные и структурные решения для схем ПЛИС в значительной мере уже определились и перешли в стадию простого наращивания количественных характеристик, то для схем ПАИС и SOPC архитектурные и структурные решения еще только развиваются. О степени подвижности этого аспекта проектирования говорит тот факт, что новые семейства БИС ПЛИС появляются с частотой приблизительно раз в два года, а принципиально новые архитектурные решения появляются не чаще чем раз в пять лет.

Что касается инструментария проектирования — САПР и самой методики проектирования, то эта составляющая в настоящий момент находится в состоянии стремительного нарастания своей интеллектуализации, предоставляемых средств и возможностей автоматизации. Подтверждением интенсивности изменений в методике проектирования является тот факт, что новые версии САПР поставляются фирмами с частотой раз в квартал. Вместе с тем, в отечественной литературе вопросам создания и даже выбора современных средств проектирования (если не считать переводов или переложений фирменных руководств по использованию САПР) уделяется явно недостаточное внимание. Последние издания отечественной литературы по этому направлению [1, 2] вышли более 10 лет назад. Поэтому материал данной главы, кроме изложения общих концепций, содержит и справочные данные. Авторы надеются, что включение справочного материала (несмотря на ожидаемое быстрое устаревание) поможет читателю лучше сориентироваться в основных направлениях и тенденциях развития современных методик проектирования.



## 2.1. Общие сведения о процессе проектирования

*Проектирование* — комплекс работ, целью которого является получение технической документации, позволяющей реализовать или изготовить новый или модернизируемый объект с заданными свойствами и с заданным функционированием в заданных условиях. В общем случае, объектами проектирования могут быть изделия (например, мобильный телефон, ЭВМ, стиральная машина) или процессы (например, технологические, вычислительные). Сущность процесса проектирования изложена в работах [12, 16]. В контексте рассматриваемой проблематики нас интересуют процессы, связанные с созданием электронных систем.

*Стратегия проектирования* — функциональная декомпозиция. Для системы в целом и ее блоков используется концепция "черного ящика". Для "черного ящика" разрабатывается функциональная спецификация, включающая внешнее описание блока (входы и выходы) и внутреннее описание — функцию или алгоритм работы: $F = \Phi(X, t)$, где $X$ — вектор входных величин, $F$ — вектор выходных величин, $t$ — время. При декомпозиции функция $\Phi$ разбивается на более простые функции $\Phi_1 ... \Phi_K$, между которыми должны быть установлены определенные связи, соответствующие принятому алгоритму реализации функции $\Phi$. В результате разбиения, в конечном счете, получается структура. *Переход от функции к структуре — синтез.*

Синтез неоднозначен. Выбор наилучшего варианта осуществляется по результатам анализа, когда проверяется правильность работы и некоторые показатели, характеризующие устройство. Процедуры синтеза и анализа постоянно чередуются.

Декомпозиция функций блоков выполняется до тех пор, пока не получатся типовые функции, каждая из которых может быть реализована на элементах выбранного уровня иерархии.

*Процесс проектирования* — многоуровневый, многошаговый и итерационный, с возвратами назад и пересмотром ранее принятых решений. Комплекс проектных работ, как правило, включает в себя теоретические и экспериментальные исследования, расчеты и конструирование.

Различие теоретической базы и понятийного аппарата, используемых на разных стадиях проектирования, приводит к тому, что традиционным является разбиение процесса проектирования как электронных систем, так и БИС/СБИС, на этапы, приведенные на рис. 2.1.


**Рис. 2.1.** Традиционные этапы проектирования

На этапе *системного* проектирования определяется архитектура, состав компонентов и основные характеристики будущей системы, выбирается элементная база для ее построения. В ходе работ на этом этапе наиболее сложным, трудно формализуемым и, следовательно, слабо поддающимся автоматизации, является, например, принятие решения о разделении функций между программной и аппаратной или между цифровой и аналоговой частями системы. Большую роль здесь играют индивидуальные аспекты и опыт разработчика. Весьма сомнительной представляется возможность введения какого-либо обобщенного критерия качества, позволяющего однозначно определить наилучшую архитектуру проектируемой вычислительной системы.

При *структурно-алгоритмическом* проектировании определяются алгоритмы функционирования аппаратных и программных компонентов системы.

На этапе *функционально-логического* проектирования разрабатываются функциональные и принципиальные электрические схемы, программы, подготавливаются тестовые и контрольные данные.

На *конструкторском* этапе производится привязка элементов проекта к конструктивным элементам.



Разбиение процесса проектирования на этапы связывают с различием технических средств, как привлекаемых для создания проекта в роли инструментария, так и используемых в качестве компонентов проекта, а также и технологических особенностей реализации конечного продукта. Хотя общая методология процесса проектирования не зависит от варианта разбиения процесса проектирования на отдельные уровни, содержание, а также методы и средства проектирования для различных уровней оказываются очень специфичными и существенно зависят как от типа применяемой элементной базы, так и от способа реализации (изготовления) конечного продукта.

Представленная последовательность действий характерна для всех уровней проектирования. Для каждого уровня декомпозиция заканчивается при получении типовых функций, соответствующих этому уровню иерархии. Так, например, при многоплатной реализации проекта декомпозиция заканчивается на верхнем уровне иерархии при представлении проекта в виде отдельных плат, на следующем уровне — в виде отдельной платы (типового элемента замены), еще ниже декомпозиция осуществляется до реализации функций при помощи той или иной микросхемы. При ориентации на программируемые (разрабатываемые) пользователем микросхемы процедура декомпозиции осуществляется уже для этой микросхемы в соответствии с составом функциональных библиотек программируемых БИС/СБИС.

На любом этапе проектирования может быть выявлена ошибочность или не оптимальность выбранного ранее варианта реализации или принятого решения. Такая ситуация требует оценки целесообразности возврата и пересмотра решений. С учетом возможностей современных САПР проектирование может считаться законченным после верификации проекта в целом, когда завершена отладка готового изделия.

Последовательная декомпозиция проекта на отдельные фрагменты (с определением функций каждого фрагмента и его интерфейса) характерна для любого этапа проектирования и применяется при разработке широкого спектра цифровых устройств, начиная от устройства целиком и кончая проектированием отдельных БИС/СБИС. Такая методология проектирования отображает процесс проектирования *"сверху вниз"*: от технического задания до электрических схем, файлов прошивки ПЗУ и конфигурации программируемых приборов, а также конструкции устройства в целом.

Другая последовательность, соответствующая методологии *"снизу вверх"*, предусматривает объединение простейших модулей в более сложную структуру до тех пор, пока, в конце концов, не будет создан конечный проект. Исходные модули — это решения, созданные проектировщиком на более ранних этапах работы или в ходе работ над другими проектами, или доступные проектировщику и входящие в состав имеющихся библиотек САПР.

## 2.1.1. Факторы, влияющие на методику проектирования электронных устройств

Первым фактором, влияющим на специфику проектирования и, как следствие, на возможные САПР, является *тип обрабатываемой информации* и связанные с ним методы и способы ее обработки. Проект или его отдельные фрагменты могут включать аналоговые, аналого-цифровые и/или цифроаналоговые элементы, строиться на основе дискретных (цифровых) компонентов или опираться на встроенные микропроцессорные средства. Отсюда следует многообразие вариантов проектирования, которые в современных технологиях часто называют *потоком проектирования* (Design Flow). Поток проектирования при этом определяется тем, какие компоненты превалируют в проекте.

Следующим определяющим фактором является выбор *технической базы* для реализации фрагментов проекта, а также *технологического способа реализации* самого проекта. Как правило, одно и то же электронное изделие может быть реализовано различными способами. Здесь должен быть дан ответ на вопрос — будет ли проект построен на стандартных микросхемах, будут ли использоваться те или иные специализированные ИС или комбинация различных типов ИС. Более того, в разные моменты жизненного цикла проекта в зависимости от тиражности изделия в составе проекта могут меняться используемые типы ИС.

В наибольшей степени появление и широкое распространение БИС с программируемой структурой и конфигурируемых систем на кристалле повлияло на два соседних иерархических уровня проектирования: проектирование собственно ИС с программируемой структурой (ИСПС) и проектирование печатных плат, содержащих такие БИС.

Хотя во многих чертах проектирование для этих двух уровней близко, специфика проектирования и, соответственно, применяемые САПР заставляют рассматривать их, в зависимости от анализируемой проблемы, раздельно или последовательно.

## 2.1.2. Области применения СпИС различных типов

Все типы СпИС имеют свои целесообразные области применения, поскольку каждому типу свойственно определенное соотношение таких параметров, как сложность (достижимый уровень интеграции), быстродействие, стоимость. На выбор типа СпИС для реализации проекта влияет совокупность свойств. Основные соображения можно пояснить с позиций экономики, обратившись



к формуле стоимости ИС, изготовляемой в соответствии с уже освоенным технологическим процессом:

$$C_{ис} = C_{изг} + C_{пр}/N,$$

где $C_{изг}$ — стоимость изготовления ИС (стоимость кристалла и других материалов, стоимость технологических операций по изготовлению ИС, контрольных испытаний). Затраты на изготовление относятся к каждой ИС, т. е. повторяются столько раз, сколько ИС будет произведено; $C_{пр}$ — стоимость проектирования ИС, т. е. однократные затраты для данного типа ИС; $N$ — объем производства (тиражность), т. е. число ИС, которое будет произведено.

Стоимость проектирования БИС/СБИС велика и может достигать сотен миллионов долларов. Для дорогостоящих вариантов проектирования БИС/СБИС производство становится рентабельным только при большом объеме их продаж.

Затраты $C_{пр}$ и $C_{изг}$ находятся во взаимосвязи. Рост затрат на проектирование, как правило, ведет к снижению $C_{изг}$, поскольку чем совершеннее проект, тем рациональнее используется площадь кристалла и другие его ресурсы. Отсюда видно, *что выигрыш по экономичности могут получать те или иные типы СпИС в зависимости от тиражности их производства и сложности.*

Применительно к микросхемам программируемой логики справедливы следующие положения. Простые устройства со сложностью в сотни эквивалентных вентилей целесообразно реализовывать на PLD (PAL, GAL, PLA). При росте сложности проекта естественен переход к FPGA и CPLD, если тиражность ИС сравнительно невелика. Рост тиражности (приблизительно свыше десятков тысяч) ведет к преимуществам реализаций на БМК, т. к. стоимость изготовления небольшого числа шаблонов для создания межсоединений разложится на большое число микросхем, а стоимость изготовления каждой ИС уменьшится благодаря исключению из схемы программируемых связей и средств их программирования.

При еще большей тиражности выгодным оказывается метод стандартных ячеек (СЯ), позволяющий дополнительно улучшить параметры схемы, плотнее разместить ее элементы на кристалле, т. е. уменьшить $C_{изг}$ и улучшить быстродействие. При этом, слагаемое $C_{пр}/N$ в формуле стоимости ИС не окажется слишком большим благодаря большой величине $N$, хотя необходимость проектировать весь комплект шаблонов для технологических процессов приводит к большим затратам $C_{пр}$.

Полностью заказное проектирование для СпИС не характерно. Оно стоит настолько дорого, что применяется практически только для создания стандартных БИС/СБИС массового производства.

Диаграмма областей целесообразного применения разных типов СпИС в зависимости от их сложности и тиражности приведена на рис. 2.2.

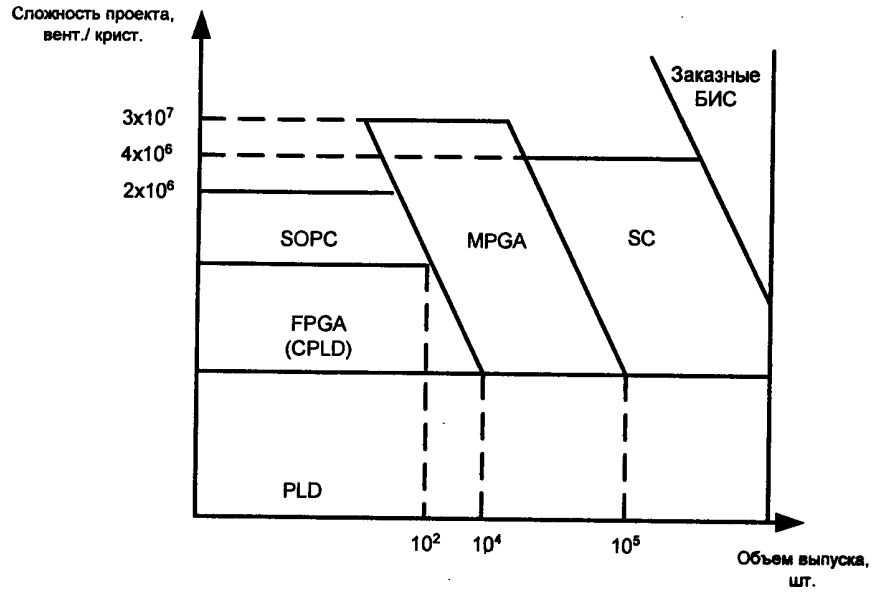


**Рис. 2.2.** Рациональные объемы выпуска СпИС разных технологий проектирования и изготовления

## 2.1.3. Место БИС с программируемой структурой в процессе создания современной аппаратуры

Проектирование стандартных ИС массового применения, как и проектирование заказными методами вообще, — удел крупных специализированных фирм. На долю системотехников приходятся главным образом другие разработки: цифровые устройства малой сложности на МИС и СИС, микропроцессорные системы для целей управления техническими объектами и технологическими процессами, малотиражная аппаратура либо прототипы систем на основе ИС с программируемой структурой.

*Проектирование на основе МИС* и *СИС* — наиболее традиционный процесс, в котором используются как эвристические подходы, так и формализованные методики. Проектировщик задает структуру устройства на базе своих знаний, идей и освоения опыта предшественников, а при определении функций отдельных блоков пользуется и формальными методами. Требуется знание функциональных возможностей ИС из выбранных наборов ИС, их свойств и параметров. В современных условиях, когда наблюдается тенденции снижения стоимости схем с программируемой структурой, следует ожидать замещения систем, построенных на МИС/СИС, на системы с ИСПС. При этом МИС/СИС останутся в тех фрагментах схемы, где требуются специфические характеристики (оптронная развязка, повышенное выходное напряжение и т. д.).



*Микропроцессорная система* создается в результате разработки комплекса программно-аппаратных средств. Проектирование аппаратной части сводится к компоновке системы из типовых модулей: центрального процессорного элемента, различных видов памяти, адаптеров, контроллеров и внешних устройств. Способы подключения модулей к шинам микропроцессорной системы, описания основных модулей, *сведения о типах и методике их выбора, о методике их программирования и применения изложены в литературе* [27]. Ключевой проблемой при проектировании микропроцессорных систем была и остается проблема разработки программного обеспечения.

Помимо проектирования микропроцессорных систем различного уровня и функционального назначения, львиной долей инженерных разработок аппаратуры в условиях современной России, по-видимому, явится *использование схем с программируемой структурой* как для создания автономных устройств, так и их применения в составе микропроцессорных систем. Более того, в соответствии с ростом возможностей построения систем на одном кристалле сами микропроцессорные системы могут оказаться одним из элементов, входящих в состав БИС с программируемой структурой. В подобной ситуации уже трудно определить — являются ли подобные СБИС программируемой логикой со встроенным микропроцессором или это МП-система со встроенной программируемой периферией. Также трудно провести четкую грань между проектированием собственно МП, проектированием его периферии и, тем более, проектированием связи между МП-ядром и периферией. *Сведения о средствах и методике отладки ПО, входящего в состав SOPC, приведены в данной главе.*

Целесообразность использования того или иного типа СпИС может определяться различными соображениями. Наиболее распространенной ситуацией оказывается случай, когда с целью обеспечения минимального времени выхода конечного изделия на рынок (time-to-market), производимые приборы на начальном этапе жизненного цикла изделия строятся на основе схем с программируемой структурой, затем (по мере готовности) продукция переводится на использование полузаказных БИС. На рис. 2.3 показана такая зависимость. Выбор типа полузаказных БИС (БМК или стандартные ячейки) может определяться необходимостью сокращения времени выпуска серийной продукции (time-to-silicon).

На рисунке также отражена ситуация, соответствующая случаю, когда на заключительных этапах жизненного цикла могут оказаться нецелесообразными заказы новых партий полузаказных БИС, а экономически обоснованным будет возврат к использованию ИСПС. Кроме соображений быстрого выхода на рынок, причиной ориентации первых опытных партий конечной продукции на использование ИСПС может служить простота корректиро-

вок внутренней конфигурации БИС по результатам эксплуатации опытной партии.

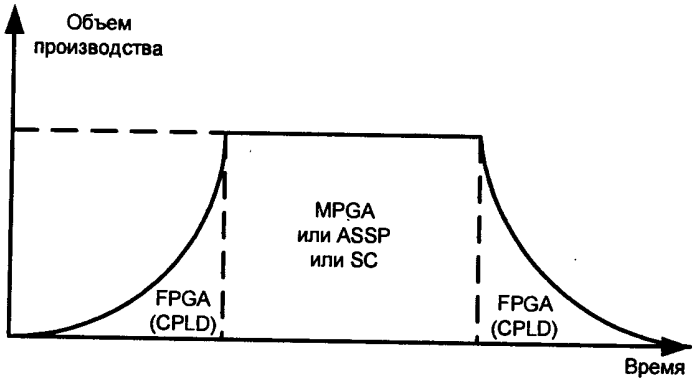


**Рис. 2.3.** Зависимость применяемого типа СпИС от времени в жизненном цикле устройства

Достоинством применения БИС типа ИСПС, и в том числе типа SOPC, является не только легкость корректировок как макетов, так и опытных партий продукции, но и определенная секретность разработок. В отличие от традиционной реализации на стандартных дискретных элементах, при воплощении системы на ИСПС структура ее оказывается невидимой, и этим соответственно затрудняется несанкционированное повторение разработок, а тем более выпуск их усовершенствованных вариантов.

Широкое внедрение БИС типа SOPC в настоящее время сдерживается не только и не столько экономическими соображениями (БИС пока относительно дороги), сколько издержками новизны (недостаточная подготовленность кадров, малое количество у разработчиков апробированных решений, недостаточное качество и количество методических и учебных материалов). В силу сказанного, отечественным разработчикам следует уделять особое внимание вопросам проектирования схем с программируемой структурой.

## 2.2. Основы организации проектной процедуры для ИСПС

Приведенная выше последовательность этапов проектирования соответствовала самому общему взгляду на эту процедуру и не учитывала специфики работ, выполняемых на каждом из этапов. Практически для любой современной системы содержание работ и используемый при этом инструментарий на всех этапах проектирования тесно связаны с выбором элементной базы. В настоящее время, независимо от примененной элементной базы,



проектирование выполняется с помощью систем автоматизированного проектирования САПР. САПР для проектирования и комплексной отладки программного обеспечения МП-систем обычно называют *интегрированной средой разработки* или *оболочкой*. Для единства терминологии, кроме случаев, узко ориентированных на разработку ПО МП-систем, будем использовать общий термин "САПР" для любых систем автоматизированного проектирования. *Для схем с программируемой структурой, даже для ПЛИС не очень высокой сложности, проектирование связано с обязательным применением тех или иных САПР.*

Для лучшего уяснения специфики современного проектирования произведем параллельный сравнительный анализ традиционной и современной методик проектирования. Чтобы охватить наибольшее число возможных сценариев проектирования и, тем самым, обеспечить общность дальнейшего рассмотрения, предположим необходимость проектирования традиционной многоплатной системы, требующей как программной, так и аппаратной реализации, необходимость обработки аналоговой информации и последующей стыковки с цифровыми фрагментами системы. Сравнение станем производить с учетом как уже существующей элементной базы, так и современных тенденций ее развития. Будем предполагать проектирование, опирающееся на возможности современных БИС с программируемой структурой (для реализации как цифровых, так и аналоговых фрагментов системы) и возможности современных САПР. Укрупненная *структура организации процесса проектирования* для подобных систем показана на рис. 2.4.

Хотя относительно традиционных методик процедура проектирования внешне осталась той же, изменилось содержание и взаимодействие отдельных ее этапов.

*Проектирование на концептуальном уровне* возлагается на проектировщика и слабо связано с автоматизацией. Сложность автоматизации этого этапа связана со спецификой работ для различных прикладных областей и систем. Автоматизация касается скорее современных способов получения, копирования и отображения исходной информации для проектирования и автоматизации вычислительных работ. Исходные данные для проектирования на этом этапе содержат требования к основным технико-экономическим показателям (производительности, энергопотреблению, стоимости, надежности), конструктивным и другим параметрам. Кроме того, для управляющих фрагментов системы должны быть определены реализуемые алгоритмы управления, для типовых дискретных фрагментов — классы выполняемых задач, для аналоговых — требуемые преобразования и их свойства (точность, скорость). Это уровень принятия концепций, используемых абстракций, методик решения и т. д.

Только в отдельных случаях, для тех приложений, которые характеризуются единством методики решения и используемого математического аппарата, возможно *сквозное применение средств автоматизации*. Целый ряд фирм раз-

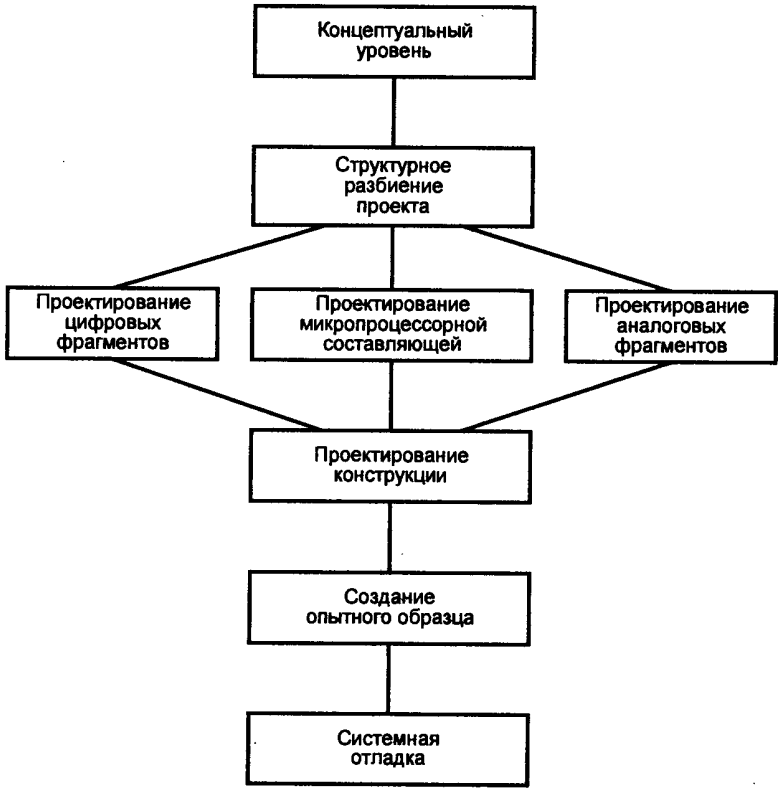


**Рис. 2.4.** Структура автоматизированного проектирования

рабатывает средства и методы автоматизации системного уровня проектирования. Решения, принятые на этом самом первом этапе проектирования, на 80% предопределяют цену конечного продукта, затраты на реализацию проекта и скорость появления конечного продукта на рынке. Примерами прикладных областей, где обеспечивается достаточно легкая стыковка средств автоматизации системного и последующих этапов проектирования, является цифровая обработки сигналов (DSP), анализ и синтез динамических систем. Фирма Elanix (**www.elanix.com**) предлагает редактор SystemView, обеспечивающий взаимодействие с проектным потоком фирмы Xilinx, используемым для разработки DSP-систем. Работа редактора ориентируется на применение схем ПЛИС фирмы Xilinx и предлагаемых этой же фирмой стандартных аппаратных решений, оформленных в виде модулей интеллектуальной собственности (DSP IP-core) и настраиваемых на конкретные условия применения специальной программой — генератором ядер (Core Generator). Редактор SystemView обеспечивает не только решение задач системного уровня проектирования DSP-систем, но и выполняет функции управления взаимо-



действием отдельных проектных средств. В результате SystemView позволяет проектировать, оптимизировать и тестировать DSP-алгоритмы. На рис. 2.5 приведены традиционный и предлагаемый фирмой Elanix процессы проектирования.

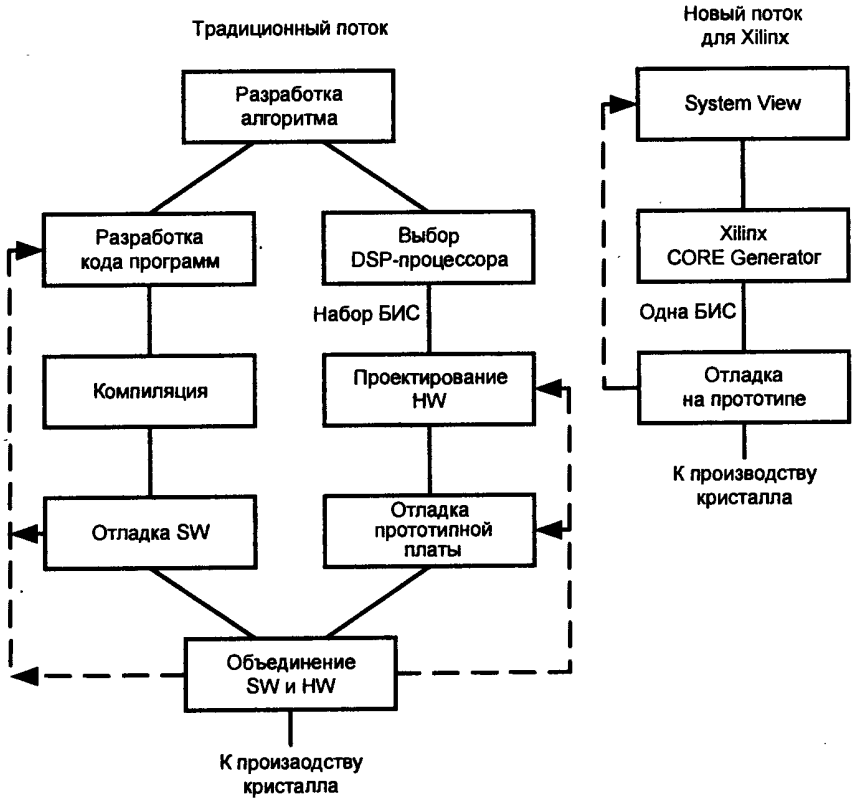


**Рис. 2.5.** Начальные этапы проектирования DSP-систем на базе схем фирмы Xilinx

Другой проблемной областью, имеющей хорошо отработанную математическую базу, и для которой возможно использование САПР, начиная с самых начальных этапов проектирования, является разработка вычислительных комплексов, управляющих динамическими системами. Примером САПР, поддерживающей проектирование подобного класса приложений, может служить пакет Professional VisSim фирмы Visual Solutions, Inc. (**www.vissim.com**). Программный пакет VisSim объединяет в своем составе интуитивно простой блочный диаграммный интерфейс с мощными моделирующими программами. Структура взаимодействия основных компонентов пакета приведена на рис. 2.6. В состав Professional VisSim входит блок VisSim Viewer, который по-

зволяет разработчику еще до покупки лицензии определить необходимый и целесообразный набор средств пакета.

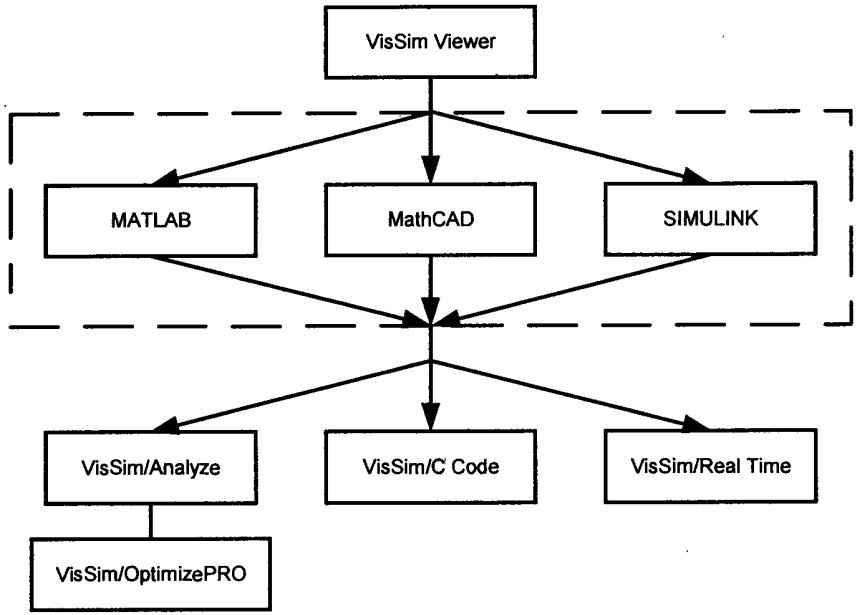


**Рис. 2.6.** Основные компоненты пакета Professional VisSim

Все версии Professional VisSim включают интегрированные с ним известные пакеты MATLAB, SIMULINK и MathCAD. Задачей блока VisSim/Analyze является линеаризация нелинейных систем. В основе работы блока лежит анализ корневого годографа и частотный анализ непрерывных и дискретных систем. Блок VisSim/OptimizePRO — вычисляет оптимальные значения основных характеристик и параметров проекта исходя из ограничений, установленных пользователем, и начиная от заданных им приближенных значений. Блок VisSim/C Code автоматически преобразует модели пакета в оптимизированный на языке С код, который может быть скомпилирован для работы на большинстве вычислительных платформ. Блок VisSim/Real Time обладает способностью объединять системные модели с реальным процессом или контроллером для моделирования в реальном масштабе времени работы проектируемого устройства управления. Ряд дополнительно подключаемых блоков, таких как VisSim/ModelWizard и VisSim/matLab Compiler, еще больше упрощает работу проектировщика. Фирма выпускает также программные прототипные пакеты для ускорения разработок. Графический метод доступа к моделям элементов позволяет легко создавать, модифицировать и комбинировать состав будущей системы. Пакет позволяет разработчику быстро создать программное обеспечение или виртуальный прототип



системы или процесса, чтобы определить основные характеристики и параметры проекта до его физической реализации.

Конечно, далеко не для всех прикладных областей ситуация складывается столь удачно, и поэтому проектирование и средства, используемые на системном уровне проектирования, требуют значительных временных затрат, плохо автоматизируются и предполагают высокий профессионализм проектировщика.

Следующий этап проектирования связан с *предварительной (концептуальной) проработкой проекта*. На этом этапе, исходя из требуемого функционирования системы, проектировщик осуществляет разбиение проекта на физически реализуемые части, определяет множества входных и выходных сигналов (как системы в целом, так и ее составных частей), их характер и взаимосвязь, а также выбирает способы конструкторской реализации отдельных составных частей. Основным результатом этапа является распределение алгоритмов работы системы между программным и стандартным аппаратным обеспечением выбранного типа МП-ядра. Другим важным результатом может стать выделение задач, требующих для своей реализации разработки нетипового оборудования (как цифрового, так и аналогового). Отличия между традиционным и современным подходом намечаются именно на этом этапе. *Выбор проектировщиком элементной базы будущего проекта и привлекаемой САПР предопределяет весь дальнейший процесс проектирования.* Чаще всего работы этого этапа выполняются одним проектировщиком, называемым проектным менеджером.

При традиционном подходе вопрос об используемой элементной базе решался отдельно для каждой из ветвей (направлений) проектирования. Более того, конструктивная реализация в форме отдельных печатных плат для аналогового фрагмента, МП-фрагмента и цифрового фрагмента на дискретных компонентах позволяли разделить во времени процедуру проектирования на отдельно выполняемые работы, включая конструкторскую разработку отдельных плат, их индивидуальную реализацию и автономное тестирование. И лишь на заключительных этапах производилась совместная отладка многоплатной реализации системы.

Существующая тенденция интеграции проектных решений приводит к возможности построения всей системы на одной печатной плате, а возможно даже в одной БИС (вся система реализуется на одном кристалле SOPC). При современном уровне развития техники и технологии изготовления БИС вопрос выбора элементной базы оказывается значительно более важным, чем раньше. Например, анализ возможных вариантов реализации может привести к выводу о целесообразности создания смешанной аппаратно-программной системы (для МП, совмещенного с ПЛИС) или гибридной системы (для ПЛИС, совмещенной с ПАИС) в форме одиночной БИС. После определения варианта реализации системы проектировщик для выполнения последующих этапов должен выбрать САПР из жестко предопреде-

денного набора. Таким образом, выбор элементной базы, осуществленный на этом этапе, может задавать стиль, методы и средства решения задач по требуемым направлениям проектирования.

Для многих приложений БИС теперь начинают конкурировать с традиционными решениями и вытеснять не только многоплатные конструкции, но даже вычислительные комплексы на базе персональных компьютеров. Те задачи, которые ранее решались с помощью подключения разрабатываемого устройства к ПК, теперь могут быть реализованы в одной БИС. Например, целый ряд сервисных задач, который раньше мог быть решен только компьютером, подключенным к телефонной линии, сейчас может быть реализован в одной БИС, встроенной в мобильный телефон.

Если предположить, что проектируемая система требует интеллектуальности, достижимой только при использовании МП или МК, то, прежде всего, должен быть решен основной вопрос этого этапа — вопрос о технической реализации микропроцессорного ядра. Возможные варианты реализации сводятся либо к ориентации на автономный МП, либо на МП, встроенный в БИС ПЛИС. Если планируется использовать МП, встроенный в БИС ПЛИС, то возникает очередной вопрос — каким образом будем встраивать МП? Будет ли МП-ядро представлять собой библиотечный элемент (Softcore) для ПЛИС типа generic, или это будет аппаратное процессорное ядро (Hardcore) в ПЛИС типа SOPC.

Результаты выполнения работ двух рассмотренных этапов позволяют перейти к последующим параллельным этапам проектирования. В отличие от традиционного подхода, когда порядок работы сильно зависел от конструкторской реализации и достаточно жестко увязывался во времени с окончательной готовностью макетных плат отдельных проектных направлений, современный подход позволяет выполнять *проектирование одновременно для нескольких направлений*. Современные средства и технологии проектирования позволили разрушить существовавшую ранее жесткую взаимосвязь отдельных направлений и обеспечить возможность выполнения проектных работ в произвольных взаимных сочетаниях. Даже этап конструкторско-технологического проектирования, благодаря легкости перепрограммирования ИСПС, может начинаться (а иногда даже заканчиваться) до получения окончательных результатов по всем параллельным ветвям проектирования.

Процедуры проектирования по всем параллельным ветвям в САПР для ИСПС сходны. Как разработка программного обеспечения для МП(МК)-ядра, так и разработка дискретной и аналоговой частей проекта могут рассматриваться как последовательность трех основных этапов:

- ввод исходной информации (спецификации проекта);
- компиляция проекта;
- тестирование полученных результатов.



Конкретное содержание этапов для аппаратной и программной частей проекта, цифровой и аналоговой частей, естественно, различное. Компиляция аппаратной части проекта приводит к синтезу устройства (или устройств) на базисе выбранных элементов (со стандартной и/или программируемой структурой), а компиляция программной части проекта приводит к синтезу кодового представления программ.

Полученные результаты требуют тщательной проверки, поэтому за этапом синтеза следует этап тестирования, выполняемого моделированием и/или реальными экспериментами. Моделирование, как правило, имеет несколько уровней с разной степенью отображения свойств реального объекта. Оно может быть как функционально-логическим, проверяющим правильность функционирования устройства или программы, так и функционально-временным, учитывающим задержки элементов, составляющих проект. Поведение реальной системы будет зависеть еще от ряда факторов, в том числе результатов окончательной трассировки межсоединений или времен исполнения отдельных программных фрагментов. В результате тестирования могут быть выявлены ошибки, которые требуют исправлений, это придает процессу проектирования *итеративный характер с возвратами к прежним этапам и введением в проект необходимой коррекции.*

По мере отработки решений по отдельным ветвям проектирования отрабатываются и вопросы связи между этими ветвями (например, между программной и аппаратной частями проекта), хотя комплексный анализ и отладка могут быть выполнены только после завершения проектирования соответствующих ветвей процедуры проектирования. Естественно, такое последовательное представление о проектировании является условным и в реальных условиях выполняется последовательно-параллельно и с многократными итерационными возвратами к началу проектных процедур.

Окончание проектирования по отдельным ветвям создает исходные данные для завершающего конструкторско-технологического этапа проектирования, результатом которого явится создание реальной системы. Физическая реализация проекта, в свою очередь, создает основу для комплексной отладки решений, полученных на отдельных ветвях проектирования.

Современный этап развития САПР направлен на *ускорение* работ по всем направлениям. Для этого разрабатываются средства, методы и методология их использования, позволяющие сдвигать работы каждого этапа работ в направлении их более ранней реализации и совмещения с другими направлениями или даже опережения по определенным направлениям. Изменения произошли по всем составляющим проектных процедур. Существующая элементная база позволяет легко трансформировать проектные решения в рамках одной ветви проектирования и перемещать решения между параллельными ветвями.

Современный инструментарий проектирования дает возможность в рамках одной САПР или за счет передачи информации в другие САПР (легко реа-

лизуемой как на конечном этапе, так и на промежуточных) выполнять этапы всех типов работ с недостижимой ранее скоростью.

Даже для экспериментальной проверки совсем не обязательно ждать реализации будущей системы целиком. Возможна практическая проверка всего устройства или его отдельных фрагментов с помощью либо специально разработанных устройств, либо предлагаемых различными фирмами достаточно универсальных наборов отладочных средств, включающих, как правило, отладочную плату, программные средства и методические материалы. Названия таких средств отражают целевую направленность отладочного средства. Спектр отладочных средств включает следующие разновидности:

- средства, предназначенные для предварительного знакомства с БИС рассматриваемого класса (обычно это комплекс HW- и SW-средств, называемых *стартовый набор*, Starter Kit);
- средства для оценки применимости проектных решений (обычно включают так называемую *оценочную плату* или *демонстрационную плату*, Evaluation Board);
- средства для отладки прикладных проектных решений (обычно базируются на *макетной плате* или *проектной плате*, Development Board);
- средства, замещающие на начальных этапах выпуска готовой продукции оборудование, которое еще находится в конструкторско-технологической разработке (обычно называемые *прототипными платами*, Prototype Plate).

Несмотря на некоторые отличия по имеющимся ресурсам и предлагаемым возможностям, любой тип плат позволяет выполнять широкий круг экспериментов. Все платы позволяют также изменять программное обеспечение МП для средств отладки МП-систем, либо содержимое памяти конфигурации для микросхем с программируемой структурой, либо и то и другое для БИС типа SOPC.

Естественно, что допустимый объем выполняемых экспериментов значительнее у более дорогих средств. В стартовых наборах обычно удается реализовать и проверить работоспособность лишь отдельных фрагментов будущей системы.

После успешного завершения экспериментальных работ файлы с программным обеспечением или файлы памяти конфигурации могут использоваться либо для изготовления требуемых ИСПС, либо для записи в соответствующие виды промежуточных ПЗУ. Файлы отчетов о результатах компиляции обычно содержат информацию о конкретных данных по монтированию проекта в реальную систему или БИС. Поэтому уже после этапа компиляции проектов возможен переход к разработке технологической реализации проекта, например, к разработке топологии печатных плат, являющихся, как правило, конечной продукцией проектирования.

Ввиду легкости перепрограммирования как программной, так и аппаратной части проекта, этап экспериментальных работ с ним может быть отложен до



ершения конструкторской разработки печатной платы. *Даже значитель- изменения схемы на основе ИСПС обычно не влекут за собой столь ката- рофических последствий, как в случаях использования жесткой стандартной ики.* Для упрощения реализации этого подхода фирмы-изготовители едлагают набор схем, отличающихся логической мощностью, но имеющих инаковое расположение выходных контактов, что делает модификацию оектов вопросом скорее экономическим (более мощные БИС стоят доро- ), чем техническим. Поэтому экспериментальные работы целесообразно водить на различных отладочных прототипных системах исключительно я ускорения общего процесса проектирования, поскольку для большинст- ИСПС возможно совмещение во времени этапов конструкторской разра- тки и экспериментальных работ.

терационные возвраты к повторным процедурам компиляции в ходе кон- рукторско-технологического этапа проектирования возникают в том слу- е, когда, исходя из соображений повышения помехоустойчивости или бо- е эффективной разводки соединений БИС на печатной плате между со- й, целесообразно изменить расположение входных и/или выходных нтактов ИСПС и их подключение к выходным разъемам печатной платы. одобная возможность следует из способности современных ИСПС обеспе- вать различные варианты монтирования одного и того же проекта в одну у же БИС.

овременные методы проектирования продолжают сосуществовать с тради- онными. И те, и другие сильно зависят от варианта реализации проекта тви реализации для рис. 2.4) и в некоторых деталях отличаются друг от уга, поэтому далее рассмотрим параллельные ветви проектирования по дельности.

## 2.1. Проектирование цифровых фрагментов а заказных ИС и стандартных искретных компонентах

еобходимость проектирования цифровой части современных систем обычно отвечающей за логику функционирования) на дискретных компо- нтах малой и средней интеграции, несмотря на рост интеграции ИС, со- анилась, хотя объем такой продукции существенно уменьшился и в на- тоящий момент составляет около 10—20%. В любом проекте может воз- икнуть ситуация, когда часть ИС не может быть включена в БИС общего азначения и целесообразно использование элементов МИС. В методике роектирования при ориентации только на этот тип элементной базы за по- леднее время особых изменений не произошло. Наиболее существенные зменения произошли в самом подходе к созданию проектов, замещающих ранее созданные системы, которые были реализованы на дискретных ком- понентах или включали значительные фрагменты с подобной реализацией.

При решении задачи перевода реализации таких фрагментов на современную элементную базу разработчик ставится перед выбором наиболее рационального варианта. Необходимость перевода диктуется не только и не столько престижными соображениями, сколько экономическими. Стоимость ИС старого типа (из разных соображений и причин) не только не уменьшается, а иногда даже и увеличивается (малый спрос, например, приводит к сокращению или даже прекращению выпуска неперспективных СИС и МИС). Вместе с тем, во многих проектах остаются фрагменты, требующие использования специализированных ИС. Примерами фрагментов, которые обычно не включаются в состав общецелевых БИС, являются фрагменты, решающие следующие общесистемные проблемы: создание ИС осцилляторов частоты со средними или высокими требованиями к точности реализации, гальваническая развязка на основе оптронных элементов, отдельные мощные элементы, элементы сопряжения различных систем элементов и т. д.

Поэтому наряду с выпуском ИС с увеличенной логической мощностью и функциональными или эксплуатационными возможностями продолжается выпуск МИС, реализующих отдельные логические функции. Например, фирма Fairchild Semiconductor производит семейство ИС под названием TinyLogic, которое представляет собой реализацию в одном корпусе (различных габаритов) одной единственной функции, например функции 2И-НЕ или аналогового ключа.

Для разработчиков цифровых фрагментов фирмы-производители электронного оборудования предлагают различные варианты технологической реализации. Если исключить разработку полностью заказных схем, то у разработчика остаются направления, показанные на рис. 2.7.

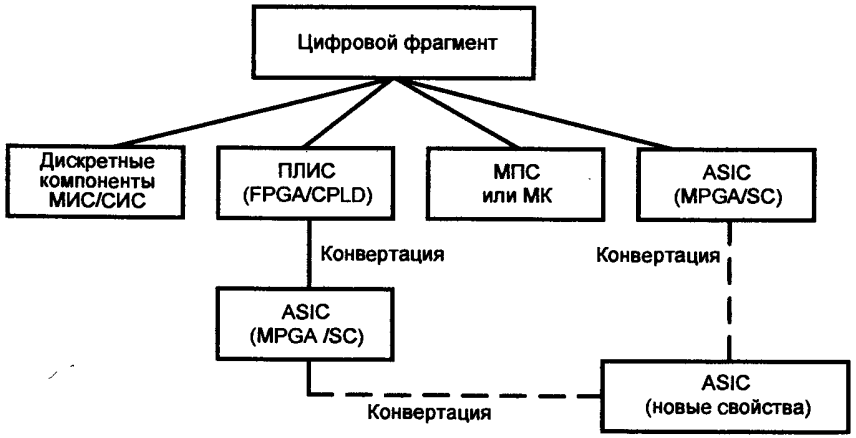


**Рис. 2.7.** Технологии проектирования и реализации цифровых фрагментов



Возможно проектирование по четырем основным направлениям: традиционная реализация на дискретных схемах СИС/МИС, реализация на ПЛИС, программная интерпретация цифровой схемы на МК и реализация в форме полузаказного кристалла того или иного типа. Естественно, при физической реализации конечной продукции может оказаться целесообразным применение любых комбинаций перечисленных вариантов. Более того, реализация ряда комбинаций может оказаться целесообразной и в рамках единой БИС.

Так, например, уже появлялись сообщения [43] и [46] об объединении в одной БИС схемы ASIC и фрагмента ПЛИС. Этот вариант иногда считают средним путем реализации полузаказных схем. Его же ряд производителей называет вариантом ПЛИС со встроенными стандартными блоками. Последнее определение представляется предпочтительным, т. к. в большей мере отражает способ создания структуры такой комбинированной БИС. Следует ожидать, что номенклатура блоков, относимых к стандартным, будет постоянно увеличиваться, и у разработчика появится большая свобода выбора реализации БИС.

В тех случаях, когда при переходе к реализации в другом типе БИС проект не меняет своего функционирования, а сама процедура перевода выполняется в автоматическом или полуавтоматическом режимах, процесс носит название *конвертация проекта*. На рис. 2.7 показаны такие варианты переходов от одного способа реализации к другому.

Каждый базовый вариант проектирования характеризуется своим специфическим потоком проектирования, более того, маршруты проектирования у разных фирм могут отличаться. Представляется целесообразным более подробно остановиться на потоках проектирования, характерных для основных вариантов реализации.

## Реализация в базисе дискретных элементов типа МИС и СИС

Рассмотрим традиционную реализацию в базисе дискретных элементов типа МИС и СИС. На рис. 2.8 показан маршрут проектирования, характерный для такого способа реализации цифровых фрагментов.

При традиционном проектировании основные изменения коснулись предоставляемой элементной базы и возможностей САПР. Эти изменения взаимосвязаны, например, увеличение объема доступной к использованию номенклатуры ИС потребовало включения характеристик этих элементов в состав библиотек САПР. Целый ряд фирм — Mentor Graphics Corporation, Synplicity, Innoveda (прежде ViewLogic, Inc.) — предлагают САПР, ориентированные на задачи проектирования систем класса ASIC. Приведенное на рисунке распределение функций, естественно, является условным. В некоторых случаях фирма-изготовитель готова взять на себя бóльшую долю проектных работ, чем изображено на рисунке.

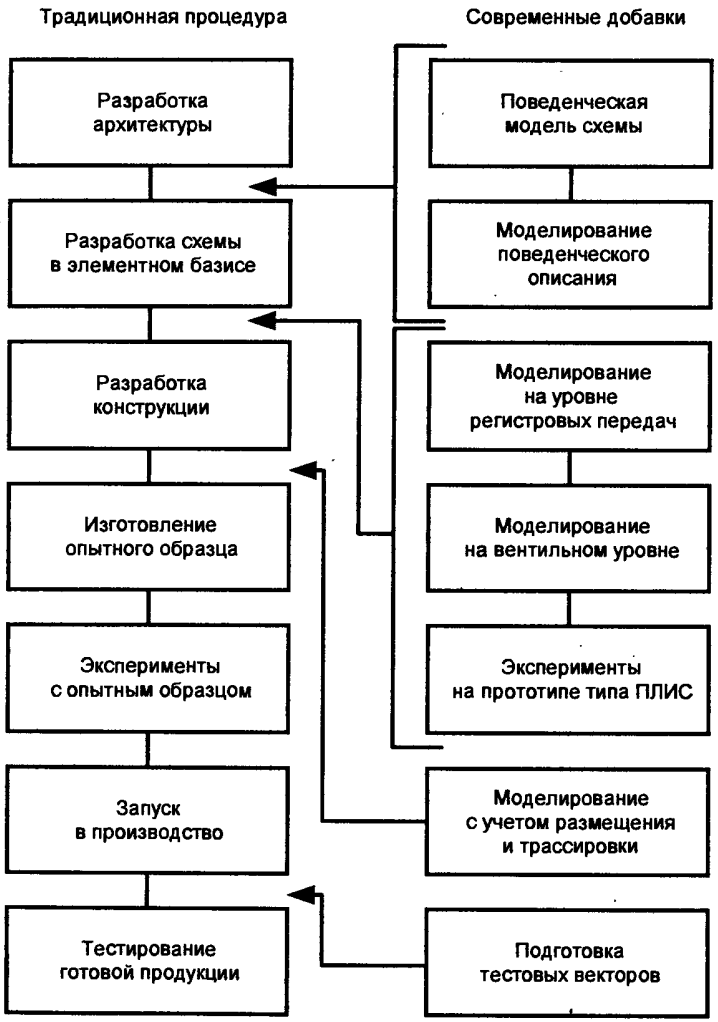


**Рис. 2.8.** Типовая последовательность проектирования цифровых фрагментов

Для рассматриваемого направления САПР изменяют и сам подход к проектированию, существенно расширяя и добавляя к нему новые возможности. Наибольшее внимание в современных САПР при этом уделяется проблеме верификации и тестирования разработки, где главным инструментом служит моделирование. Основной проблемой в моделировании раньше являлась ограниченная вычислительная мощность компьютеров, на которых устанавливались САПР. Производительность современных компьютеров (даже персональных) настолько возросла, что стало возможно моделирование сложных систем (содержащих большое число элементов) за приемлемые времен-



ные сроки, при этом не только увеличена адекватность результатов моделирования, но и скорость и простота получения результатов, теперь почти не связанных со сложностью проекта. В САПР последних поколений не только увеличены объемы библиотек стандартных элементов, но и предпринимаются значительные усилия по упрощению корректировки состава и параметров этих библиотек, включая корректировки через ресурсы сети Интернет. Современные САПР, в отличие от САПР предыдущего поколения, предполагают выполнение значительно большего числа сервисных операций. Значительная часть улучшений сосредоточена в сфере моделирования свойств будущего проекта. Например, САПР позволяют автоматически определять: критические пути (фрагмент схемы, имеющий максимальный путь с критическими по скорости распространения сигнала задержками); максимально допустимую тактовую частоту; места возникновения рисков; места и условия возникновения самопроизвольной генерации фрагмента; вычисление требуемых значений времен предустановки или удержания сигналов и т. д. Увеличивается число сервисных функций САПР, предназначенных для выпуска пригодной к тестированию аппаратуры (например, выполняется автоматическая генерация тестовых векторов как для моделирования, так и для организации экспериментов с реальной аппаратурой).

## Реализация цифровых фрагментов на ПЛИС

возможным, а зачастую целесообразным вариантом реализации системы, является ее полное или частичное воплощение на ПЛИС. Поскольку проектированию цифровых фрагментов систем на основе ПЛИС и положительным эффектам, связанным с такой реализацией, будут уделены последующие разделы книги, здесь отметим только те моменты, которые могут, по нашему мнению, препятствовать переводу проектов на этот вариант изготовления. Проектирование на ПЛИС является существенно новым направлением, которое еще недостаточно поддержано выпуском учебной и научной литературы. Одним из моментов отказа от реализации на ПЛИС может стать необходимость при проектировании обязательно (хотя бы на этапах программирования) пользоваться специальными САПР и, конечно, требование модернизации на начальных этапах парка средств сопровождения разработок. Отсутствие отечественных разработок современных ПЛИС — еще один фактор, в определенных ситуациях останавливающий реализацию конечной продукции на этом типе БИС. Нехватка соответствующих квалифицированных кадров, и отсутствие у разработчиков навыков грамотного схемотехнического проектирования схем на основе ПЛИС может также явиться причиной нежелания создания продукции на этом типе ИС. Специфика внутренней организации ПЛИС требует замены целого ряда традиционных схемотехнических решений на другие. Наиболее важной представляется ориентация на организацию тактированной работы отдельных узлов ПЛИС, и непривычным для многих разработчиков оказывается невозмож-

ность построения различных времязадающих цепей на основе резисторов и конденсаторов. Помощь даже мощной САПР не заменяет опыта проектировщика, а только упрощает и ускоряет его работу, и далеко не всегда САПР может отличить хороший проект от плохого.

## Реализация цифровых фрагментов в форме однокристального микроконтроллера

Выполнение цифрового фрагмента в микроконтроллерной форме также во многих случаях может быть удачным решением задачи, стоящей перед разработчиком. В момент появления первых МК многим проектировщикам показалось, что понадобится совсем немного времени, и все цифровые проекты окажутся реализованными на однокристальных МК или, в худшем случае, на МПС. В действительности этого не произошло. Конечно, определенный (и не малый) сектор рынка электронной продукции этот способ реализации цифровых фрагментов захватил и продолжает удерживать. Однако подобная реализация требует от проектировщика знаний и умений по разработке программного обеспечения, привлечения специального оборудования для целей отладки ПО и т. д.

Кроме того, опыт использования МК-техники показал, что далеко не все проекты удачно ложатся в русло традиционной однокристальной реализации. Существенным ограничением, например, является трудность создания цифровых фрагментов, реализующих быстрое взаимодействие нескольких параллельных процессов управления. Для таких приложений воплощение на ПЛИС может оказаться более предпочтительной. Как и в варианте перевода МК-проекта на реализацию в форму SOPC желательно, чтобы проектировщик совмещал в одном лице и схемотехника, и программиста. Более подробно современный подход к реализации фрагментов, обрабатывающих цифровую информацию с помощью МП-систем, будет рассмотрен в *разд. 2.2.2*. А средства, предназначенные для проектирования на основе кристаллов SOPC, объединяющих в себе плюсы ПЛИС и МК, будут обсуждаться в *разд. 2.5*.

## Реализация цифровых фрагментов в форме полузаказных кристаллов

Помимо перевода продукции на реализацию в форме ПЛИС (как видно из рис. 2.6 и как указывалось в *разд. 2.1*), существуют и другие варианты технологического воплощения аппаратных фрагментов с цифровым представлением информации в форме конечного продукта. Выбор каждого варианта на начальном этапе проектирования является скорее вопросом экономическим и стратегическим и лишь после этого переходит в плоскость технической реализации. Как уже отмечалось выше, при достаточной тиражности проектируемой продукции целесообразной оказывается ориентация на технологию класса ASIC с реализацией по одной из возможных технологий изготовления полузаказной БИС. Выбор способа реализации зависит не только от сроков изготовления, тиражности, объема начальных затрат, но и от функционального состава проекта.



Общий объем продукции, выполненной по технологии стандартных ячеек (СЯ), за последние годы резко увеличился и составляет значительный и постоянно увеличивающийся сектор конечной электронной продукции. Общее увеличение доли продукции на базе СЯ связано с улучшением технологии проектирования и изготовления устройств этого класса. Уменьшаются сроки разработки, уменьшается риск выпуска неработоспособной продукции, упрощается взаимодействие с изготовителем ИС. Существенным аргументом в пользу реализации на том или ином виде полузаказной БИС являются лучшие окончательные характеристики конечной БИС. В табл. 2.1 приведены сравнительные данные для различных способов реализации межсоединений. В качестве базового варианта сравнения принят вариант полностью заказного кристалла.

***Таблица 2.1.*** *Сравнение характеристик для различных технологий задания системы межсоединений*

| | Laser-programmable gate array | MPGA/SC | FPGA/CPLD |
|---|---|---|---|
| Время до выпуска первого кристалла | Дни | Недели | Минуты или часы |
| Тактовая частота | В 1,1 раза меньше | В 1,25 раз меньше | В 3–5 раз меньше |
| Площадь | В 3,3 раза больше | В 1,6 раз больше | В 10–20 раз больше |

Если разработчик ориентируется на реализацию проекта в виде полузаказной БИС, то, независимо от способа ее изготовления, в процессе разработки требуется выполнение согласованных и/или совмещенных работ заказчика (или проектировщика) и фирмы-изготовителя ИС. Формы взаимодействия и используемое оборудование зависят от многих факторов, в том числе от фирмы-изготовителя и от формы заказа. Далее рассмотрим проектные процедуры для некоторых типичных примеров, отличающихся технологией изготовления конечного продукта.

Фирма American Microsystems, Inc. (AMI) обеспечивает производственный выпуск семейства БИС типа AMI3HS с проектными нормами 0,35 мкм, относящихся к классу ASIC и с технологией изготовления по *методу стандартных ячеек* (Standard Cell).

Архитектура БИС, предлагаемая фирмой AMI, строится на основе следующих основных фрагментов: индивидуальных функциональных ячеек, мегаячеек (megacells), информационных функциональных блоков (datapath functions) и блоков памяти, которые с целью уменьшения размеров и повышения быстродействия являются оптимизированными и предварительно скомпилированными. Фирма специализируется на проектировании дополнительных схем заказчика, связанных с выполнением логических и даже простых аналоговых функций. Библиотека мегаячеек фирмы весьма обширна, чрезвычайно гибка и включает: процессорные ядра от 8 до 32 разрядов, набор периферийных элементов от таймеров до дисковых контроллеров, информационные фрагменты от сдвигающих регистров и сумматоров до умножителей, различные виды буферных блоков FIFO.

Рассмотрим типовую процедуру проектирования БИС этого типа. БИС позволяют реализовать проекты заказчика, содержащие до 4 млн. вентилей в кристалле плюс емкость встраиваемой памяти. Основные этапы проектной процедуры, принятой фирмой, приведены на рис. 2.9.

Из рисунка видно, что проектирование выполняется в тесном взаимодействии заказчика и изготовителя. Работы требуют стыковки САПР, используемых заказчиком и изготовителем. Фирма AMI имеет собственную САПР под названием ACCESS Design Tools, которая интегрирует программные средства для верификации цифровых ASIC-проектов, для конвертации FPGA- или PLD-проектов в ASIC-проекты или для корректировок и конвертаций ASIC-проектов в другие формы. Пакет и методология его использования обеспечивают значительное сокращение времени проектных работ, уменьшают величину однократных (начальных) денежных вложений заказчика в проект (в английской терминологии NRE, Non Recurrent Expends), увеличивают вероятность успешности выпуска уже первых образцов кристалла. Пакет поддерживает методологию компактного проектирования, хорошо согласованного с производственным процессом фирмы, а сама фирма предлагает помощь разработчикам на любых этапах проектирования.

Для ускорения и упрощения импортирования описаний из САПР различных фирм, AMI ориентируется на языки Verilog и VHDL. Стандартизация описаний предопределяет простоту переводов проектов в различные технологические реализации, включая перевод в MPGA, другие классы ASIC и т. д.

Заказчик ИС для выполнения своей части работ может использовать САПР фирм Cadence, Mentor Graphics, Synopsys, Viewlogic, Veribest и некоторых других. Проектирование начинается с передачи изготовителем ИС заказчику проектного набора, содержащего библиотеку ячеек, их символьного изображения, моделей для симуляции, а также программного обеспечения для проектной верификации, вычисления временных соотношений и генерации списка соединений. Временное моделирование до разводки опирается на среднестатистические фирменные данные о значениях емкостей и сопротивлений межсоединений. После детальной разводки, выполняемой фирмой-



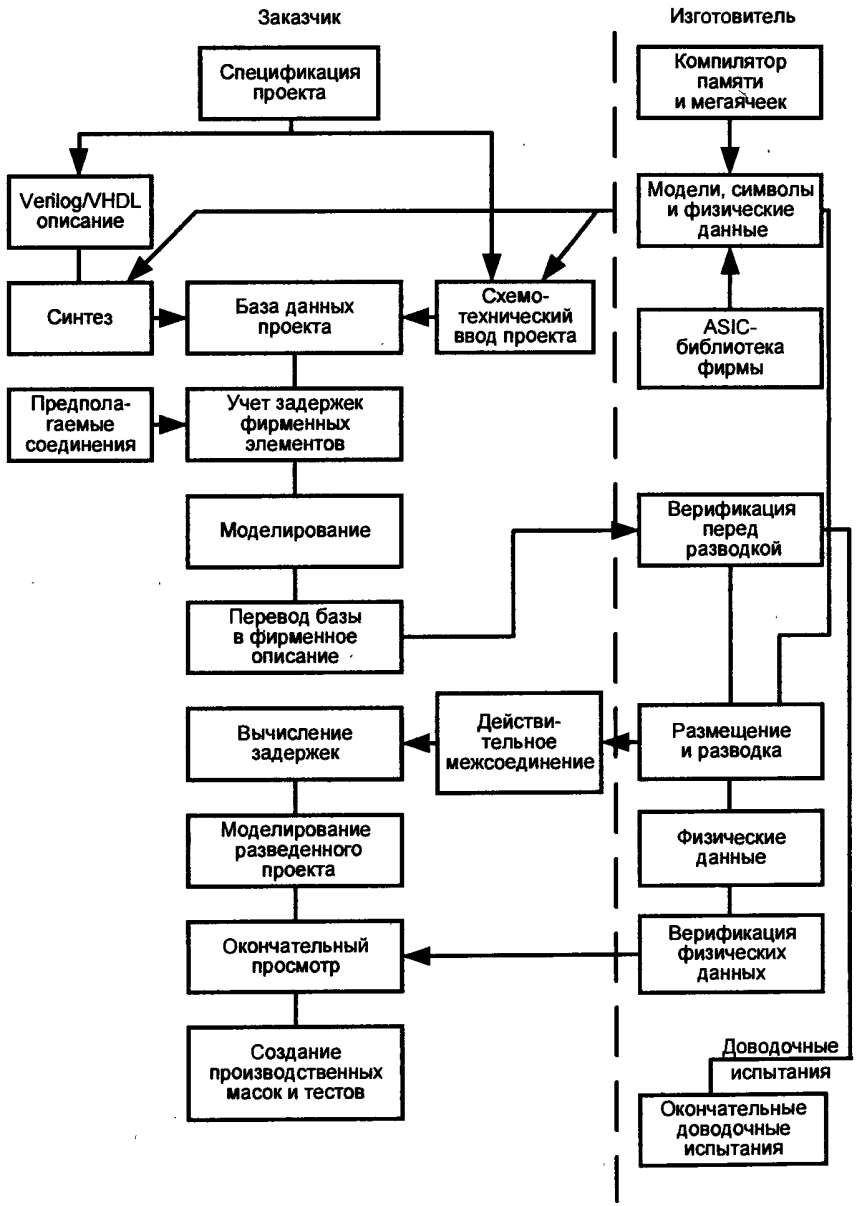


**Рис. 2.9.** Этапы проектирования продукции класса ASIC SC, используемые фирмой American Microsystems, Inc.

изготовителем, таблицы емкостей и сопротивлений действительных соединений передаются разработчику и используются им для определения ожидаемых фактических временных параметров прибора. Одновременно с работой над собственно проектом, заказчик и изготовитель подготавливают тестовые процедуры для проверки доводочных модификаций и готовых изделий. К доводочным испытаниям прибегают после вставок элементов граничного сканирования JTAG/SCAN, выполнения оптимизации размещения In-Place Optimization (IPO), а также после выполнения процедуры балансировки времен синхронизации (BCT) отдельных фрагментов. Только после получения удовлетворительных результатов моделирования, произведенных заказчиком, фирма-изготовитель приступает к выпуску масок и подложек. Тестовые программы разрабатываются заказчиком и изготовителем совместно на базе САПР и тестируются на прототипах перед их окончательной сдачей.

Существенными факторами, которые предопределяют ориентацию разработчика на продукцию той или иной фирмы, являются логическая мощность кристаллов, тактовая частота или задержка на вентиль, а также предоставляемый фирмой-изготовителем набор библиотечных элементов, сроки изготовления, стоимость при различных объемах партии, потребляемая мощность, возможность управления различными режимами потребления и т. д.

Приведенная выше процедура проектирования остается справедливой и при реализации в форме матричных кристаллов. Фирма AMI, кроме технологии стандартных ячеек, поддерживает выпуск кристаллов класса MPCA. Производимые фирмой БМК-кристаллы семейства AMI5HG с проектными нормами 0,5 мкм (от 4,6 до 1320 тыс. вентилей в кристалле) выполняются по бесканальной технологии типа "море вентилей", разрабатываются на той же САПР ACCESS Design Tools и по такой же технологической схеме. Некоторые отличия наблюдаются только в составе библиотек.

Другой фирмой, выпускающей БИС класса ASIC с технологией изготовления по *методу базовых матричных кристаллов* MPCA, является фирма LSI Logic Corporation. Фирма является ведущей в мире по числу вентилей в кристалле (до 33 млн. вентилей). В качестве примера рассмотрим проектный процесс при ориентации на кристаллы типа G10p с проектными нормами 0,35 мкм.

В основе проектной процедуры лежит использование фирменной технологии под названием FlexStream, которая интегрирует в своем составе как специфические собственные проектные средства, так и встроенные лучшие САПР других фирм, таких как Avanti, Cadence, Mentor Graphic и Synopsys. Подобное объединение различных средств в единой оболочке позволяет использовать эту технологию на всех уровнях проектирования (системном, логическом и физическом).



В состав собственных средств фирмы, помимо библиотек, включаются проектные средства, которые позволяют осуществлять проектный анализ, интерфейс со стандартными САПР, вычисление задержек, генерацию модулей, контроль проектных норм и т. д. Так же, как и фирма AMI, фирма LSI Logic в процессе работ над выпуском кристалла предполагает тесное взаимодействие с заказчиком-проектировщиком. Основное различие состоит в ориентации фирмы LSI Logic на описания проектов на уровне регистровых передач. В процедуре проектирования фирма выделяет семь основных контрольных точек. Содержание этапов и взаимодействие разработчика и изготовителя приведены на рис. 2.10.

Рассмотрим последовательно маршрут проектирования и кратко охарактеризуем работы, выполняемые на отдельных этапах.

- Этап 1. Начало работ над проектом. Заказчиком создается описание проекта на блочном уровне и спецификация на уровне регистровых передач (RTL). В проекте могут использоваться стандартные решения фирмы LSI Logic — CoreWare. Описание вводится в САПР, САПР контролирует проект и выполняет синтез до вентильного уровня. Одновременно с синтезом производится проверка функционирования и соблюдения указанных временных ограничений. Фирма LSI Logic просматривает спецификации проекта и может осуществлять консультирование и обучение разработчиков. Процесс контроля и синтеза итерационный. Результатом работ этого этапа является предварительное разбиение проекта на части и составление списка соединений.
- Этап 2. Начальное физическое размещение. Список соединений и ограничения анализируются совместно, чтобы быть уверенными в том, что проект хорошо структурирован и не содержит фрагментов, которые смогут создать проблемы (во временной области) при физической реализации. Осуществляется статический временной анализ и функциональная верификация.
- Этап 3. Пробная разводка. Блоки разводятся, размещаются шины питания и синхронизации, определяются места максимального скопления трасс и цепи, нарушающие временные ограничения. Информация о топологическом распределении элементов и граф их соединений объединяются в общий отчет для просмотра.
- Этап 4. Если параметры межсоединений и временные характеристики оказываются приемлемыми, тогда переходят к окончательному размещению и трассировке. В случае неудовлетворительных результатов предыдущего этапа производится повторное разбиение проекта на части, их новая организация или производится синтез с измененными ограничениями. После завершения размещения выдаются данные для верификации законченного изделия. Приступают к генерации тестирующих программ.

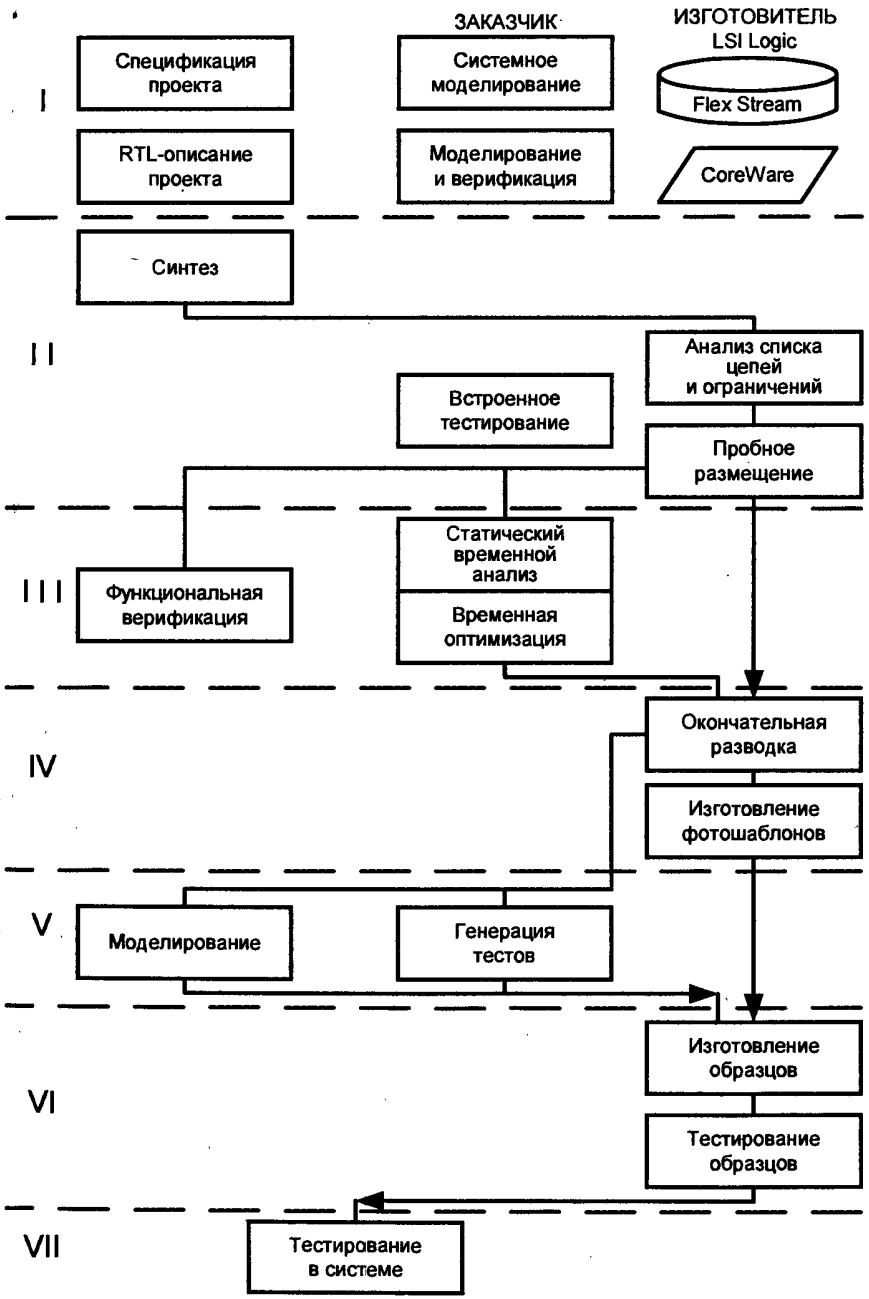


**Рис. 2.10.** Маршрут проектирования MPGA, используемый фирмой LSI Logic



- Этап 5. Заказчик анализирует данные, полученные после окончательной разводки изготовителем. Производится просмотр результатов разводки на моделях. Осуществляется разработка тестовых программ. При выполнении всех требований проект утверждается и передается на изготовление.
- Этап 6. Фирма-изготовитель выпускает фотошаблоны, изготавливает и проверяет подложки. Готовность прототипных образцов и тестов позволяют приступить к системному тестированию.
- Этап 7. Удачное завершение системных тестов, производимых заказчиком, служит признаком принятия ИС и завершения проекта.

Основное внимание при выборе технологического процесса и используемых проектных средств фирма уделяет уменьшению числа итерационных возвратов к процедуре синтеза проекта.

Важным средством, которое существенно улучшает методологию проектирования схем класса ASIC в форме MPGA/SC, можно считать *добавление этапа прототипирования* схемы MPGA схемами FPGA/CPLD. Для обеспечения эффективности процесса прототипирования желательно выполнение двух условий. Во-первых, чтобы для конечного и прототипного вариантов продукции либо использовался совпадающий способ задания исходной информации, либо взаимные переводы осуществлялись простыми методами. При этом методы и средства задания исходной информации могут быть различными. Общение между САПР изготовителя MPGA и САПР изготовителя FPGA/CPLD может осуществляться при помощи различных стандартных языков описания аппаратуры (EDIF, Verilog или VHDL). Вторым условием является обеспечение структурно-архитектурной близости между двумя вариантами реализации, которые гарантировали бы преемственность функциональных характеристик проекта. Описание на уровне регистровых передач в наибольшей степени позволяет сохранить близость поведения оригинала и прототипа.

Максимальное совпадение между характеристиками ПЛИС и БМК может быть обеспечено при совмещении в одном лице фирмы-изготовителя MPGA и фирмы-изготовителя FPGA/CPLD и при наличии у нее соответствующих типов БИС. В таком случае наиболее рациональным представляется предварительная реализация проекта в форме ПЛИС. Ведущие фирмы-производители схем ПЛИС Xilinx и Altera стараются поддержать такое направление. И та и другая фирма выпускают БМК, соответствующие определенным схемам ПЛИС. В качестве примера проектной процедуры конвертации проекта на рис. 2.11 приведена процедура переноса проекта, реализованного на БИС фирмы Altera типа APEX20KE или APEX20KC, в форму БМК. Из рисунка видно, как много проверок содержит конвертационная процедура, включая проверку основных проектных правил и ограничений DRC (Design Rule-Check). В том числе, показано формирование файлов для автоматической генерации тестовых последовательностей для оборудования как общего

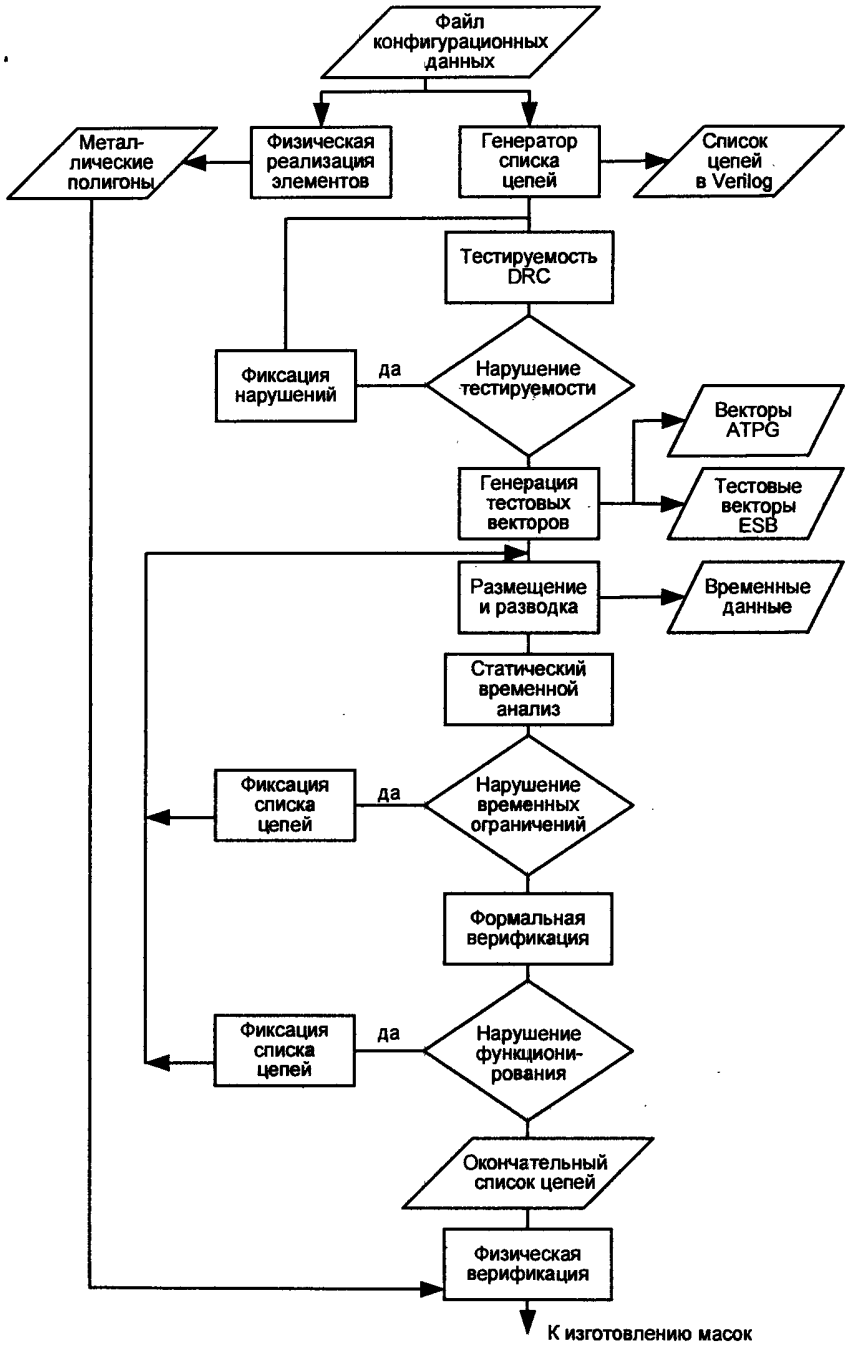


**Рис. 2.11.** Процедура конвертации проектоа для БИС типа APEX



азначения ATPG (Automatic Test Pattern Generation), так и встроенных локов памяти ESB (Embedded System Blocks).

Фирма Altera анонсировала в 2001 году выпуск БМК типа HardCopy. Семейство БМК этого типа позволяют поддерживать методологию конвертации проектов из ПЛИС типа APEX с минимальным риском иметь расхождение поведении прототипа и конечного продукта (при условии, что логическая мкость приборов лежит в диапазоне от 400 тыс. до 1,5 млн. вентилей). Минимизация достигается сохранением всех основных ресурсов ПЛИС типа APEX при переводе ее реализации в жесткие соединения вместо соединений, определяемых памятью конфигурации.

Особенностью процедуры конвертации является отсутствие необходимости проектировщика вмешиваться в ее процесс. Этот процесс не требует от проектировщика разработки и создания специальных средств тестирования типа генерации Test-Bench, методика создания и использования которых удет более подробно рассмотрена в *гл. 3*), разработки тестовых векторов ли временного и функционального моделирования. Естественно, что определенные тестовые проверки разработчик выполнял при разработке и проверке реализации своего проекта на ПЛИС типа APEX. Для проведения игрции требуются выходные файлы работоспособного проекта, сгенерированного САПР Quartus II для БИС типа APEX. На выполнение процесса онвертации и изготовление функциональных прототипов фирма Altera затрачивает порядка 8 недель, для выпуска промышленной партии требуется около 16—18 недель.

Экономическая выгода применения БМК типа HardCopy видна из следующих сравнительных данных. Стоимость изготовления комплекта масок для БИС с проектными нормами 0,13 мкм и с шестью слоями металла, которая выполняется по технологии стандартных ячеек, превышает 750 тыс. долларов. Поскольку к этой стоимости необходимо добавить дополнительные расходы фирмы-изготовителя БМК на подготовительные операции к проектированию и производственному процессу, то общая сумма расходов превышает 1 млн. долларов. Для проектирования БМК типа HardCopy фирма Altera обещает затраты на уровне 300 тыс. долларов [33].

Фирмы, специализирующиеся на производстве полузаказных БИС, также стараются поддерживать технологии конвертации проектов, выполненных на схемах типа ПЛИС, что делает целесообразным проектирование по ветви FPGA-ASIC. Однако временное отставание фирм-изготовителей БИС БМК в выпуске схем, соответствующих определенному классу схем ПЛИС, от темпов снижения цены на перспективные типы ПЛИС фирмами-производителями БИС ПЛИС часто создает ситуацию, когда экономически невыгодно отказываться от выпуска продукции на основе схем ПЛИС.

Целесообразность конвертации проектов из ПЛИС в форму БМК может следовать не только из экономических соображений. Как правило, помимо

большей уверенности в работоспособности проекта, этот подход дает и определенные дополнительные выгоды, например, допускает работу на больших системных частотах, меньшую стоимость, меньшую потребляемую мощность и т. д.

Еще одним важным моментом, который следует учитывать при поиске рациональной формы реализации конечного продукта, является отработанность у фирм технологии конвертации проектов ASIC-ASIC. Наличие таких технологий создает предпосылки для сохранения у проекта резервов за счет простоты перевода в реализации с другими и более перспективными характеристиками. Необходимость таких переводов может быть связана как с экономическими соображениями (схемы с меньшими топологическими нормами, как правило, имеют лучшие эксплуатационные характеристики и стоят дешевле), так и с возможностью реализовать проект с другими электрическими или эксплуатационными характеристиками (примером может служить перевод проекта на современные значения напряжения питания, включение проекта в систему со смешанными выходными напряжениями и т. д.).

Современное возрастание сложности проектов заставляет искать технологические варианты реализации проектов с гарантированными свойствами. Одним из основных путей увеличения надежности проектирования при условии сокращения времени на проектирование является увеличение доли уже отработанных и проверенных или просто стандартных решений.

Варианты реализации стандартных решений и способы их использования проектировщиком могут существенно отличаться. Непрерывно увеличивается число фирм, специализирующихся на разработке библиотек стандартных фрагментов для различных типов и видов ASIC продукции (IP-core). Процесс добавления такого стандартного фрагмента может иметь различную технологическую реализацию. Чаще всего фрагменты разрабатываются и записываются на одном из языков описания аппаратуры и вставляются в проект перед процедурой синтеза.

Реже используются варианты добавления стандартных фрагментов на этапах топологического проектирования, хотя именно топологическая оптимизация может резко улучшить временные характеристики фрагмента при сохранении его размеров. Для БМК фирма-разработчик IP-core должна обеспечить простоту переноса топологии стандартного фрагмента (систему межсоединений) на произвольное место среди массива вентилей. Для технологии стандартных ячеек может использоваться как перенос топологии стандартной ячейки на любое место массива ячеек, так и фиксированное размещение в кристалле стандартных фрагментов. Фирма AMI для фрагментов первого типа, в зависимости от их сложности, использует название мегаячейка (megacell) или информационный функциональный блок (datapath function), а для второго типа — скомпилированный блок (compiled block). Скомпилированные блоки (в подавляющем большинстве случаев это те или иные вари-



анты блоков памяти) оптимизированы по площади, скорости и потребляемой мощности, но могут размещаться только в заранее определенных местах кристалла.

## 2.2.2. Проектирование микропроцессорных фрагментов систем

Для большого числа современных проектов (от 60 до 80%) основой функционирования является работа программного обеспечения на платформах либо общецелевых, либо заказных (чаще DSP) процессоров. Причины появления МП в системе могут быть различными — от необходимости использовать сложные программные системы в минимальном конструктивном исполнении, до просто престижных соображений разработчика. Несмотря на то, что использование 8-битовых микроконтроллеров в разнообразных встроенных системах насчитывает уже почти 20-летнюю историю, не приходится сомневаться, что существующая практика ориентироваться на встроенные микропроцессорные решения сохранится и в будущем. Однако в свете новейших технологических и архитектурных достижений микроэлектроники, и особенно в связи с развитием ИС с программируемой структурой, целесообразные варианты построения и проектирования современных систем явно изменились и желателен их пересмотр.

Широкое практическое распространение двух альтернативных способов оперативного изменения поведения системы на уровне корректировки памяти программы МП или на уровне корректировки памяти конфигурации ПЛИС привели к возникновению новых архитектурных решений. На рис. 2.12 приведены некоторые возможные варианты.

На крайних полюсах этих решений располагаются, с одной стороны, обычная микропроцессорная система или однокристальный микроконтроллер МК, а с другой — обычная ПЛИС. При определенных условиях оба варианта могут решать одну и ту же задачу. Ранее замена реализации системы из дискретных компонентов на МПС (МК) диктовалась либо требованием миниатюризации конструкции, либо способностью к корректировкам поведенческой реакции без каких-либо конструкционных изменений на любых этапах жизни конструкции. В современных условиях оба варианта отвечают этим требованиям.

Вариант решения в форме связанных между собой БИС МП и БИС ПЛИС соответствует реализации с очень гибкими возможностями модификации организации всей интерфейсной части микропроцессорной системы, а по гибкости значительно опережает традиционную архитектуру микропроцессора, соединенного с набором программируемых периферийных БИС. Подобная организация системы позволяет сформировать периферию МП в произвольных сочетаниях — ограничениями являются не столько логиче-

ская мощность ПЛИС (обычно можно в том же корпусе установить более мощную ПЛИС), сколько количество конструктивно предусмотренных контактов ввода/вывода. Классические МПС могут достигать таких возможностей только при многоплатных или мезонинных технологиях, когда за счет изменения типа платы, подключаемой к основной плате, создается конфигурация системы с заданными ресурсами.

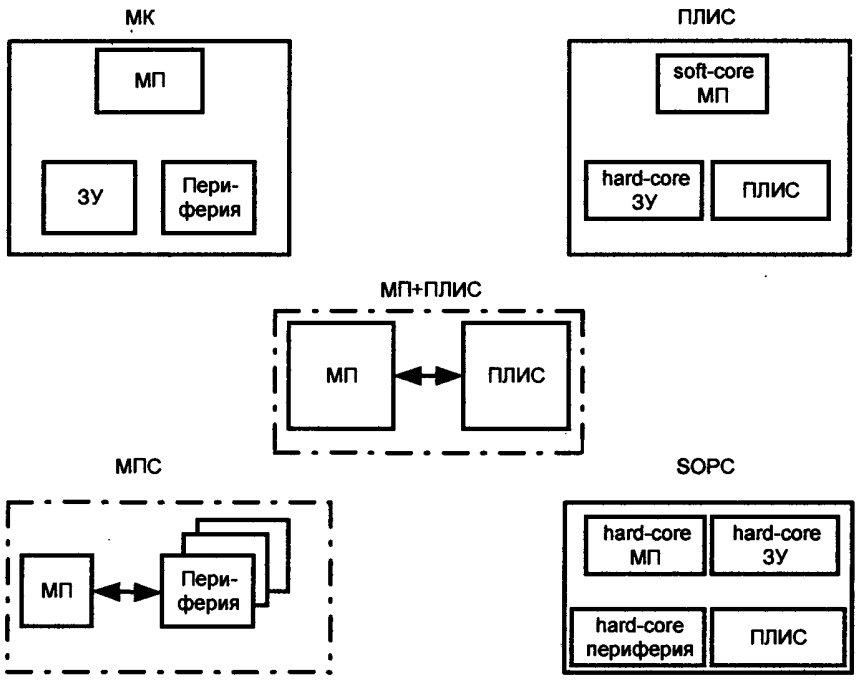


**Рис. 2.12.** Современная организация систем с оператиано корректируемым поведением

Еще бóльшими потенциальными возможностями обладают системы, совмещающие в одном кристалле оба подхода к реализации изменений поведения проектируемой или модифицируемой системы. Вопросы технической реализации БИС этого типа подробно рассматривались в предыдущей главе, такие БИС носят обобщенное название SOPC. Однако объединение в одном кристалле устройств, поддерживающих два разных подхода к модификации поведения, не приводит к чисто арифметическому сложению их возможностей, а вызывает и будет вызывать поистине революционные изменения в методике проектирования электронных систем. Два основных момента требуют подробного рассмотрения процедуры проектирования традиционных микропроцессорных систем. Во-первых, в связи с тем, что в архитектуре современных и будущих SOPC присутствует микропроцессорная состав-



яющая, рассмотрение процедуры проектирования традиционных МПС ажно для последующего анализа процедуры проектирования SOPC. Во-вторых, широкое распространение ИСПС оказало и оказывает существенное лияние на различные аспекты процедуры проектирования традиционных МПС. Таким образом, тесная взаимосвязь и фактическая близость проблем роектирования традиционных МПС и SOPC заставляют более подробно становиться на особенностях реализации микропроцессорных фрагментов современной аппаратуре. Вместе с тем, следует учитывать, что методы, ешения и средства, которые создавались и продолжают разрабатываться ля систем на кристалле SOPC, далеко не всегда могут быть применены традиционных микропроцессорных системах.

оскольку методика проектирования микропроцессорной составляющей в ИС SOPC в значительной мере опирается на опыт, полученный при проктировании традиционных МПС, приводимый ниже материал посвящен нализу влияния ПЛИС на различные этапы разработки традиционных МПС. А специфика и проблематика проектирования микропроцессорной оставляющей в SOPC будет рассмотрена в *разд. 2.5.*

ажнейшим аспектом современного проектирования является ориентация а привлечение проектных средств, поддерживающих проектирование микропроцессорных ядер. Методика привлечения заключается либо во включении этих средств в состав САПР ПЛИС, либо в организации связи инструментальных средств проектирования программных средств и САПР ПЛИС. Методика проектирования микропроцессорных систем на традиционной лементной базе к настоящему времени в достаточной мере стабилизировалась. Состав и назначение отдельных этапов проектирования при ориентации на традиционные реализации сохранились, однако произошли изменения содержания отдельных этапов.

Эти изменения связаны с тем, что теперь не обязательно ожидать реализации аппаратной части проектируемой системы перед переходом к проектированию программной части системы, как это приходилось делать до последнего времени. Это положение наглядно иллюстрируется рис. 2.13.

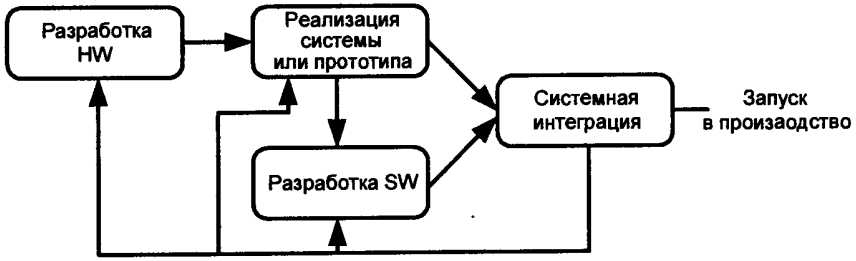


**Рис. 2.13.** Традиционное временное распределение процессов проектирования

Стремление уйти от этого периода ожидания потребовало изменения инструментария проектирования, методологии его использования и применения. В проектировании систем, базирующихся на успехах современной электронной технологии, намечается явное перераспределение целей и задач, этапов, инструментария и методологии, появляется ряд новых подходов. Далеко не все из них окончательно оформились к настоящему времени. Некоторые отличия видны из сравнения современной и традиционной методологии проектирования, приведенного на рис. 2.14.

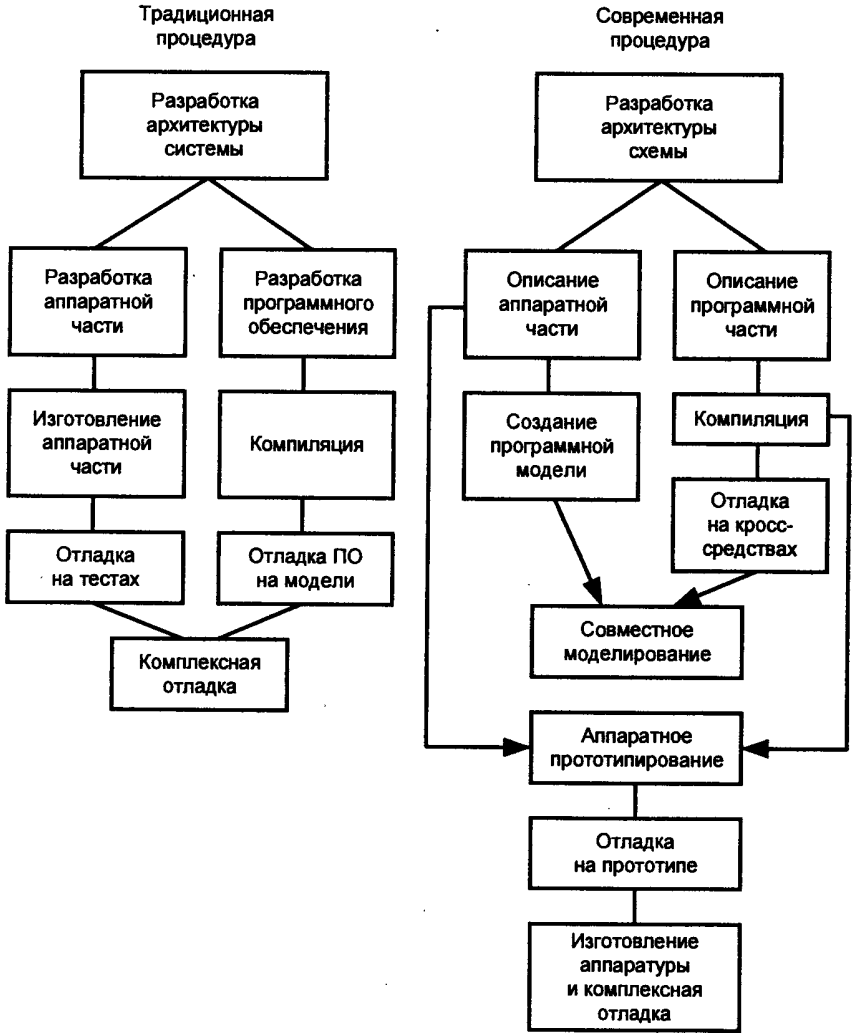


**Рис. 2.14.** Проектирование микропроцессорных фрагментов систем



в рисунка видно, что возросло число этапов работы. Вновь введенные эта- позволяют проектировщикам обнаружить на этих новых этапах моде- рования и верификации большее число ошибок и нестыковок между рограммами и аппаратурой. За счет увеличения эффективности работы со- еменных средств проектирования, этап спецификации и разработки про- аммного обеспечения существенно упростился. Те ошибки, которые пре- де обнаруживались только на этапах комплексной отладки, теперь, как равило, обнаруживаются на более ранних этапах проектирования в процес- моделирования. Рассмотрим последовательно основные этапы проекти- вания фрагментов, ориентированных на микропроцессорную реализацию.

## тап выбора типа МП

ервым этапом является *этап выбора* типа МП. Для традиционного подхода истема команд и архитектура МК или МП всегда носили предопределяю- ий характер и оказывали исключительное влияние на эксплуатационные рактеристики проектируемой системы, поэтому этап выбора типа процес- рного ядра был особенно важен. Выбор процессорного ядра предопреде- ет все составляющие проектного процесса: доступные аппаратные ресур- , имеющиеся программные заготовки (библиотеки стандартных программ, аличие разработанных систем реального времени), наличие разработанного нструментария проектирования и отладки, методики проектирования. Это оложение было особенно существенно для микроконтроллерной реализа- ии, поскольку выбор семейства предопределяет и допустимые варианты роцессорной периферии (объемов и типов памяти, скоростей и типов дру- х периферийных устройств, достаточного количества входных/выходных ортов и т. д.). К настоящему времени острота проблемы несколько умень- илась как со стороны HW, так и со стороны SW. В части HW, во-первых, роизошла стандартизация свойств и параметров периферийных блоков, а о-вторых, в них появилась гибкость и возможность выполнения в опреде- енных пределах корректировок внутренней организации. В части SW раз- аботчиками постоянно предпринимаются усилия, чтобы облегчить перено- имость его с одной аппаратной платформы на другую.

ем не менее, выбор МП или МК был и остается важнейшим этапом про- ктирования. Для оценки возможных направлений поиска обратимся к абл. 2.2 и 2.3, составленным на основании данных, приведенных в [41, 42].

***Таблица 2.2.*** *Количество фирм, производящих МП/МК (кроме процессоров DSP и сетевых)*

| | 8 разрядов | 16 разрядов | 32 разряда | 64 разряда |
|---|---|---|---|---|
| **Число фирм** | 20 | 9 | 28 | 6 |
| **Число типов МП/МК** | 38 | 15 | 71 | 13 |

*Таблица 2.3. Количество фирм, производящих процессоры DSP и сетевые*

| | 16 разрядов | 32 разряда | Специализированные ядра |
|---|---|---|---|
| **Число фирм** | 10 | 3 | 10 |
| **Число типов МП/МК** | 13 | 5 | 12 |

В табл. 2.2 приведено количество фирм, производящих МП/МК выбранной разрядности и общее число выпускаемых типов МП/МК. В таблицу не включены процессоры DSP и сетевые процессоры. Табл. 2.3 содержит информацию о DSP-процессорах и специализированных DSP-ядрах.

После 2000 года процессоры DSP по рыночным показателям (объем продаж) переместились на первое место среди процессорных ядер, сменив традиционного лидера — 8-разрядные МП-ядра. Содержание таблиц подтверждает следующие выводы. МП и МК, имеющие 8-разрядную организацию, сохраняют большую часть уже завоеванного рынка продукции, причем (из данных тех же источников) более 40% фирм поддерживают выпуск клонированных MCS-51. Стремительно развивающийся сектор телекоммуникационных приложений существенно увеличил выпуск 32-разрядных МП, а 16-разрядные решения и для МК, и для МП имеют достаточно ограниченную и стабильную сферу приложений. Для традиционной МП-реализации определенное распространение получил промежуточный вариант — 16- или даже 32-разрядное процессорное ядро и 8-разрядная организация периферии. Отмеченная ситуация еще более четко прослеживается на отечественном рынке.

Сложившаяся ситуация возникла потому, что затраты при замене 8-разрядной шины на 16-разрядную или, тем более, 32-разрядную достаточно велики. В то же время, вполне возможна эмуляция особенностей 16/32-разрядного процессорного ядра, опираясь на дешевое 8-разрядное ядро. Поэтому проектировщикам приходится тщательно оценивать соотношение экономического выигрыша и потерь скорости и увеличения объемов исполняемого кода. Для отечественных разработчиков на выбор МП-ядра, по-видимому, существенное влияние оказывает широкое распространение персональных компьютеров (ПК) класса PC для реализации разнообразных задач. Необходимость использования в этих компьютерах плат расширения с 32/64-разрядной организацией шинного интерфейса, составляющих существенную часть отечественных разработок, создает предпосылки для переноса центра тяжести перспективных разработок в область больших разрядностей.

Сокращается число употребляемых процессорных ядер. Лишь незначительное число систем команд и базовых архитектур МП получают широкое распространение. Большинство фирм, выпускающих МП и МК, пытаются



беспечить распространение своей продукции не за счет оригинальности нутренней архитектуры МП или, тем более, системы команд, а за счет бо-ее эффективных сопровождающих решений (например, удобные типораз-еры корпусов, малое потребление мощности, малая стоимость, удобный абор встроенных блоков памяти и т. д.). Такие клонированные МП в неко-орых случаях полностью замещают продукцию фирмы-разработчика исход-ой архитектуры МП. Стремление разработчиков максимальным образом беспечить преемственность и применимость решений из ранних разрабо-ок не создают плацдарма для появления новых типов МП и МК.

новь разрабатываемые БИС SOPC также ориентируются на включение в вой состав наиболее распространенных типов МП и МК. Это позволяет экономить усилия при создании новых вариантов фрагментов програм-ного обеспечения, базируясь на модификации блоков, ранее разработанных отлаженных в предшествующих реализациях. Многие из фирм, учтенные в аблицах, специализируются на выпуске интеллектуальной собственности Intellectual Property, IP), соответствующих архитектуре наиболее популярных роцессорных ядер. Для реализации SOPC можно ожидать постепенного мещения центра тяжести на использование МП с большей разрядностью.

Для 32-разрядных приложений в настоящее время наибольшее распростра-ение получили два типа RISC-процессорных ядер — это разработка фирмы RM Limited (система команд ARMv4T с расширениями для семейства RM7/ARM7TDMI или семейства ARM9 Thumb) и разработка фирмы MIPS echnologies (MIPS32 4KcB). Достаточно широкое распространение имеют рхитектуры и системы команд таких МП, как Power PC, 68K/CPU32, ColdFire, MiniRISC, TinyRISC,OakDSPCore.

## Этап выбора периферии

Выбор периферии, необходимой для реализации многих функций МП-си-темы, является следующим шагом проектирования. Основной проблемой была и остается проблема создания нестандартных устройств ввода/вывода. Наряду с традиционным вариантом реализации периферийного оборудова-ния на поставляемых различными фирмами дискретных схемах, новым под-ходом является реализация на ПЛИС всего интерфейсного блока системы (исключение могут составлять экономически обоснованная целесообразность установки действительных схем СБИС-памяти большой емкости, очень сложных и логически емких контроллеров). Широкая номенклатура выпус-каемых БИС ПЛИС с различным количеством выводов и логической мощ-ностью позволяет реализовать в одном корпусе практически любой вариант набора стандартных периферийных схем (включающей таймеры, последова-тельные и/или параллельные регистры и т. д.). Выгодами реализации на од-ной БИС ПЛИС всего интерфейсного блока, помимо традиционных пре-имуществ интеграции, таких как уменьшение потребления, габаритов и т. д., является легкость добавления или удаления из проекта отдельных элементов.

Другим вариантом является интегрирование в кристалле МП, интерфейсных средств и блока программируемой логики, что соответствует традиционной идеологии SOPC и будет рассматриваться в *разд. 2.5*. Промежуточным вариантом является идеология фирмы Agere Systems. В выпускаемые этой фирмой ПЛИС семейства ORCA4, помимо элементов программируемой логики, включен определенный набор стандартных встроенных периферийных устройств, а также интерфейсных средств, упрощающих связь с целевым МП (не входящим в состав БИС). Возможность настройки интерфейса на подключение различных типов МП порождает достаточно широкие перспективы подобного подхода.

Реализация в форме БИС SOPC имеет, помимо обычно перечисляемых достоинств, еще одну особенность — согласованность характеристик МП и встроенных блоков памяти. Последнее позволяет существенно поднять тактовую частоту встроенного МП, например, для БИС E5 фирмы Triscend тактовая частота типового МП-семейства 8032 поднята до 40 МГц.

## Этап разработки программного обеспечения

После выбора стандартной аппаратуры и разработки специфического оборудования ввода/вывода (по крайней мере, после четкой спецификации свойств и характеристик этого оборудования) переходят к этапу разработки и отладки программного обеспечения. Временное запаздывание начала работ по разработке программного обеспечения (отраженное на рис. 2.13) является основной болевой точкой проектирования аппаратно-программных систем. В *разд. 2.5*, посвященном проектированию БИС SOPC, будет показано, как эта проблема решается при современном подходе. Однако и для традиционной реализации МПС появление ПЛИС существенно повлияло на применяемые средства и методы разработки и отладки программного обеспечения. На рис. 2.15 показан состав средств, поддерживающих типовую процедуру разработки программного обеспечения. Если отладку программного обеспечения предполагается производить сразу в конечной системе, то этой отладке должна предшествовать отладка аппаратуры, и только затем отладка аппаратных и программных средств может производиться совместно. Необходимость подобного совмещения отражена на рис. 2.13.

Существенным вопросом, решаемым в начале этого этапа, является выбор технических средств, поддерживающих разработку и отладку программного обеспечения. На разных этапах отладки может использоваться различный набор отладочных средств. Стремление ускорить проектирование может приводить к тому, что одним из оснований выбора типа МП-ядра может оказаться ориентация на определенную САПР и спектр поддерживаемых ею отладочных средств. Конечно, при этом приходится учитывать и другие характеристики САПР, такие как доступность, цена, эффективность, удобство работы и пр. Для проектирования программной части МП-фрагментов набор используемых программ, как правило, объединяется в единый комплекс



управляется программой, обычно называемой *оболочкой проектирования*. Для комплексной отладки набор используемого оборудования не ограничивается ЭВМ и находящимся в ней программным обеспечением, а требует дополнительного привлечения достаточно большого количества разнообразного вспомогательного оборудования. Поэтому для такого комплекса средств, как правило, используется понятие *система разработки*. Причем, для всех этих средств разработки и отладки характерна большая ориентация на возможности и ресурсы, предоставляемые основным (Host) компьютером. Иногда его называют инструментальным или отладочным компьютером. Отлаживаемую систему, соединенную с Host-компьютером, называют в этой связке Target-системой.

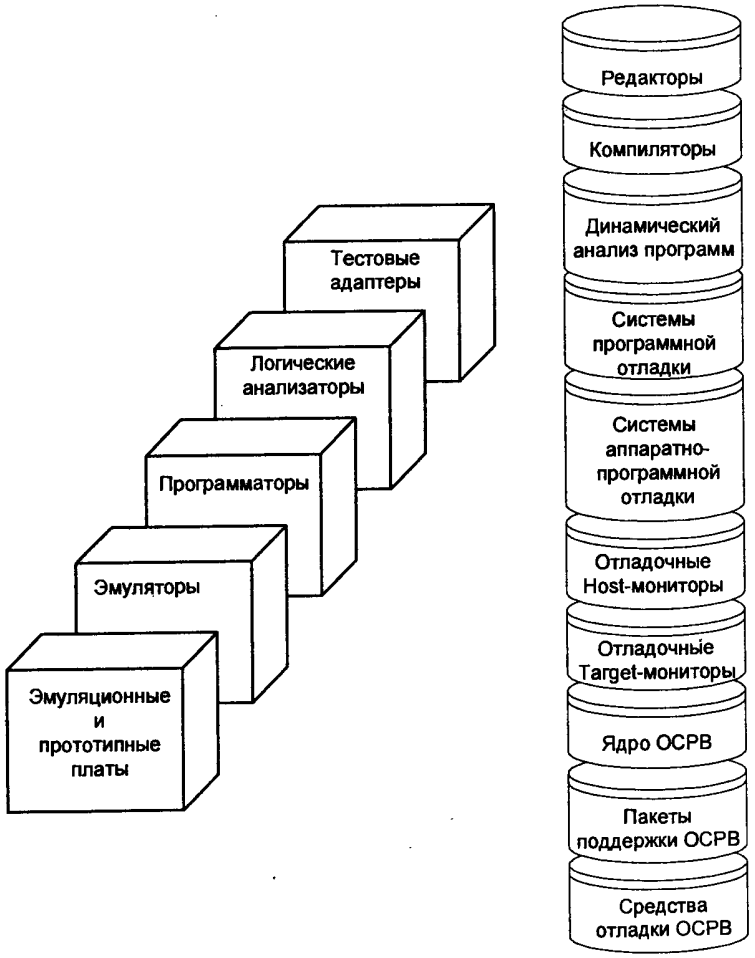


**Рис. 2.15.** Состав современной системы разработки МПС

Большое значение имеет возможность прототипирования проектируемой системы. Традиционно используемые типы прототипных плат и их назначение были рассмотрены в *разд. 2.2.*

## Средства поддержки создания программного обеспечения

Рассмотрение состава средств разработки ПО микропроцессорных систем целесообразно начать с важнейшей составляющей этих систем — средств упрощения и ускорения процесса создания программного обеспечения. Вопросам методологии и технологии создания программного обеспечения МПС посвящено достаточно большое количество работ [7, 10]. Необходимость ускорения разработки специализированного ПО вызвана тем, что от этого зависят сроки выхода на рынок конечной продукции. Скорость создания ПО при ориентации на традиционные методы проектирования не может быть существенно увеличена из-за трудности распараллеливания процесса разработки, обусловленного ограниченными возможностями проектировщиков управлять многосвязными проблемами. Вместе с тем, в реальных приложениях число параллельно решаемых задач может лежать в диапазоне от десятков до сотен. Скорость проектирования, достигаемая при применении существующих методов разработки прикладного ПО, вступает в противоречие с технологией создания новых аппаратных решений. Наряду с разработкой ПО по классическим схемам (линейный подход, процедурный подход и использование концепций ОСРВ) идет поиск новых подходов к проектированию прикладного ПО для МПС. В настоящее время наибольшее распространение получили два подхода. Один заключается в использовании систем визуального проектирования, суть другого — применение стандартных библиотек классов для проблемно ориентированного программирования. В различных странах идет интенсивный поиск методов и средств, переводящих процесс разработки прикладного ПО для МПС из творческого и уникального процесса в стандартный поток выпуска серийной продукции [5].

Не останавливаясь на всей проблематике создания ПО МПС, отметим только, что одним из основных направлений быстрой и надежной реализации программных проектов является широкое использование предшествующих наработок. Они по источнику возникновения могут быть либо результатом собственных предыдущих разработок проектировщика, либо входить в состав библиотек стандартных программ.

В состав любой современной системы проектирования включаются стандартные библиотеки, а также средства для создания собственных библиотек пользователей. Стандартные библиотеки могут иметь различную прикладную направленность. Наиболее удобными для использования являются библиотеки, органически входящие в состав систем проектирования. Боль-



инство фирм, предлагающих инструментальные средства разработки ПО, дновременно предлагают и обширные библиотеки, ориентированные на аиболее распространенные типы МПС и МК. Программные средства под-ерживают все стадии разработки пользовательских приложений. Интегри-ованные среды для МП и МК выпускаются большим числом фирм. Наи-ольшее распространение получили среды разработок SingleStep фирмы DS, MULTI фирмы GreenHills Software, VADS, IDE фирмы Keil Software, Code Composer фирмы Texas Instruments для DSP-разработок.

начительное число областей приложения микропроцессорных систем свя-ано с реализацией аппаратно-программных комплексов, работающих в ре-име реального времени. Поэтому существенное место в стандартных сред-твах проектирования занимают вопросы упрощения и ускорения разрабо-ок именно таких систем.

## Управление в реальном времени

При использовании МПС в качестве управляющих систем (а этот спектр родукции представляет весьма значительную часть всей МП-продукции) адачей программного обеспечения является контроль за физическими про-ессами и управление ими. Одной из наиболее важных проблем при этом тановится проблема управления процессами в реальном времени. Хотя по-троение систем, ориентированных на решение таких проблем, в современ-ых источниках освещено достаточно широко (много информации об ОСРВ, включая ссылки на публикации по разработке систем реального вре-ени, можно получить на сайте **www.realtime-magazine.com**), целесообразно становиться на отдельных вопросах реализации подобных систем.

словно МП-системы, связанные с управлением реальными процессами, огут быть разделены на три категории: большой сложности, средней слож-ости и малой сложности. В системах большой сложности основной про-лемой является *распределение ресурсов МП между задачами* (распределение ремени). В этих системах могут использоваться ОС общего назначения как тандартные (такие как Linux, Windows NT), так и расширенные средствами правления в реальном времени (такие как RT Linux, RTAI). Приобретение амой ОС и средств разработки для Windows NT, Windows CE или RTX по-ребует затрат более 1 тыс. долларов. Время реактивности системы с такими ОС может лежать в пределах более 1 мс для ОС общего назначения и более 100 мкс для ОС типа RT Linux.

В системах средней сложности основной задачей является *время реакции на события*, происходящие в управляемой среде, поэтому здесь применение ОСРВ оказывается необходимым. Дорогие ОСРВ класса VxWorks обходятся потребителю в 20 тыс. долларов, а более дешевые типа QNX, OS9 — в 10 тыс. долларов. Время реактивности таких систем лежит в диапазоне

от 10 до 100 мкс. Малыми временами реакции обладает поставляемая фирмой WindRiver ОСРВ VxWorks, что позволяет рекомендовать ее для использования в компактных системах реального времени. С 2001 года разработчики МПС могут бесплатно через Интернет загрузить комплекс Tornado Prototyper. Комплекс включает в себя полную инструментальную среду Tornado и моделировщик операционной системы VxWorks, работающие в среде Windows. При таком подходе проект может быть выполнен в рамках Tornado Prototyper и только на этапе производства конечной продукции потребуется приобретение стандартной версии Tornado с включенными средствами генерации VxWorks.

Требование в системах малой сложности иметь времена реакции порядка 10 мкс оказывается предельным для стандартных СРВ и заставляет отказаться от их применения. В последнем случае необходимо ориентироваться на *аппаратное решение задач управления*. И здесь, прежде всего, окажется целесообразным использование ПЛИС и SOPC. Разумное распределение функций между аппаратной и программной частями SOPC является вариантом решения возникающих проблем.

Достаточно распространенной ситуацией, особенно для систем средней и малой сложности, является случай, когда разработчикам ПО представляется нецелесообразным использование в качестве основы ПО стандартной операционной системы реального времени ОСРВ. Подобное решение чаще всего возникает из-за того, что на начальном этапе работ отсутствует или недостаточно четко специфицирована постановка задач, решаемых программным обеспечением. Постоянное добавление в ходе разработки новых функций системы или более сложной ее реакции на совокупности внешних событий приводит к существенному "разбуханию" программного обеспечения, и в результате могут возникать ошибки, которые весьма трудно обнаружить, или даже резкие провалы работоспособности подобной самодеятельной разработки системы управления в реальном времени. В отличие от этого подхода ориентация на *стандартные средства и механизмы ОСРВ* позволяет проектировщику не только значительно упростить процесс написания программ, но, что очень важно, приводит к созданию более четко структурированной системы, что существенно упрощает ее отладку. В ряде случаев разработчик может остановить свой выбор на той или иной ОСРВ из-за поставляемых одновременно с ОСРВ средств отладки и полезных утилит отладки. По сравнению с обычными средствами разработки МПС средства разработки ОСРВ функционально богаче. Как правило, они включают средства удаленной отладки, средства временного анализа исполнения, средства эмуляции целевого процессора, средства отладки взаимодействующих задач.

При ориентации на встраивание в состав своей системы стандартной ОСРВ или ее ядра разработчику необходимо решить вопросы экономической целесообразности. Предлагаемый различными фирмами набор ОСРВ весьма зна-



телен (более пятидесяти систем упомянуто только в обзорных статьях). реди ОСРВ можно выделить группу наиболее популярных систем (соответственно, у них больше и объем продаж). Это системы VxWorks фирмы indRiver Systems, OS-9 фирмы Microware Systems, pSOS фирмы Integrated stems, QNX фирмы Quantum Software Systems, LynxOS фирмы Lynx Real-me Systems и VRTX фирмы Ready Systems.

реди ОСРВ традиционно выделяют подкласс ОСРВ, реализующих идеи *ра*. В этот подкласс включаются ОСРВ с монолитным ядром. Внутри компактного ядра содержится реализация всех основных механизмов реального емени. Механизмы эти тщательно отработаны и обеспечивают решение ирокого спектра проблем, ядро построено по модульному принципу, система легко масштабируется (настраивается на условия конкретного применения), компактна и ее поведение предсказуемо. Примерами этих систем ляются системы OS-9 и QNX.

угим вариантом решения задач реального времени для систем малой ожности (при условии максимального использования стандартных решений) может являться построение своей ОСРВ на основе отдельных стандартных решений. Примером фирмы, поддерживающей проектирование льзовательских ОСРВ, является компания Real-Time User Support International (**www.rtusi.com**).

омимо экономических соображений разработчик должен учесть технические критерии выбора ОСРВ. В основе выбора лежит рассматриваемая далее почка конкретных временных расчетов. На основании анализа событий, торые могут происходить на управляемых объектах, разработчик выделяет итические события (события, отсутствие реакции на которые может привести к невосполнимым потерям в системе). С каждым событием сопоставляется критическое время реакции на это событие. После этого ищется дшая комбинация одновременно возникших критических событий. Цель кого поиска состоит в определении ситуаций, в которых к ОСРВ предъявляются наиболее жесткие требования. Знание времен реакции ОСРВ на события позволяет достаточно точно определить пригодность той или иной СРВ для управления заданными объектами.

точки зрения рассматриваемой нами проблемы применения ИСПС существует достаточно тесная взаимосвязь с реализацией ОСРВ в SOPC. Такая лизость задач, решаемых ПЛИС и ОСРВ, связана с целевой направленностью обоих направлений. И то и другое направление связаны с повышением быстродействия. Одна из основных задач ОСРВ — обеспечение гарантированной временной реакции системы на возникшее событие — может ешаться с новой степенью гибкости, благодаря мобильности аппаратуры. Возможность построения реконфигурируемых систем на кристалле, а тем более на динамически реконфигурируемых SOPC, открывает новые перспективы построения систем управления реального времени.

## Этапы кодирования и отладки программного обеспечения

После решения общих принципов построения программного обеспечения МП-систем разработчики переходят к написанию и отладке ПО. Основные изменения в методике и средствах отладки связаны с увеличением взаимосвязи между Host- и Target-системами (при этом не важно, является ли Target-система прототипом или конечным продуктом).

Изменения начинаются с первых шагов проектирования.

Первый этап работы — *создание исходных файлов*. Создание исходного файла независимо от языка программирования поддерживается современными редакторами. Встроенные в проектную оболочку или внешние подключаемые (типа Prism Editor) редакторы существенно упрощают процедуру набора и предварительного контроля вводимых программ. Для этого редактор поддерживает контекстную цветовую окраску всех синтаксических конструкций программы. Непрерывно проверяется синтаксическая корректность программы. При сомнениях в синтаксических конструкциях можно вызвать не только подсказку (Help по конструкции), но и включить в текст составляемой программы шаблон выбираемой конструкции (Template). Можно вывести и просмотреть все последующие предложения, содержащие изменения выбранной переменной. Большое количество редакционных действий над группами объектов значительно упрощает работу с исходным текстом программы.

Очередной этап работы — *компиляция программ* обычно поддерживается для двух основных языков: языка ассемблера и языка C или C++. В последнее время использование конструкций ассемблера все чаще допускается прямо среди операторов языка C. В комплекты обычно включаются программы редактирования связей (компоновщики), оптимизированные библиотеки, архиваторы/библиотекари и набор различных утилит. Современная компиляция с языка C обычно поддерживает создание различных вариантов программного кода, например, возможности компиляции от строгого ANSI C до ANSI C с различными расширениями. Программист имеет возможность настраивать компилятор на работу с различной степенью локальных или глобальных оптимизаций, создающих программный код, оптимизированный по скорости и/или размерам.

Следующий этап — один из самых ответственных этапов проектирования — *отладка программы*. В различные периоды развития средств разработки ПО использовались и получили распространение различные подходы к отладке программного обеспечения. Традиционное временное распределение взаимоотношений между разработкой программного и аппаратного обеспечения было показано на рис. 2.13.

Неправильная работа может быть связана с ошибками программы и аппаратуры. Поэтому, чтобы обеспечить быстроту и качество отладки, желательно



ользоваться различными методами и аппаратурой. Отладка может происхоить сразу на конечной продукции, а может начинаться на прототипах. Еще льшое опережение отладки достигается, если на предварительном этапе спользуются моделирующие системы, последний вариант обычно называется *кросс-отладкой*. Положительный эффект от нее связан не только с тем, о появление кросс-системы в распоряжении проектировщика может во емени значительно опережать завершение работ по изготовлению проекруемой системы, но и с тем, что работоспособность отладочной кроссистемы не вызывает сомнений, в отличие от аппаратуры разрабатываемой истемы.

а следующем этапе работ нужны средства, которые помогут обнаружить *рограммные ошибки*. Программные ошибки, в свою очередь, делятся на шибки программирования (скорее, кодирования алгоритма) и смысловые емантические). *Семантические ошибки* связаны с ошибками программной нтерпретации требуемых системных действий и в большинстве случаев обаруживаются на этапе комплексной аппаратно-программной отладки.

шибки программирования, в свою очередь, делятся на статические и диамические. Статическими являются ошибки синтаксиса, несоответствие ипов, необъявленные идентификаторы. Подавляющее большинство этих шибок обнаруживается на этапе трансляции. *Динамические ошибки* проявляются на этапе работы (исполнения) программы. К таким ошибкам относится: выход за пределы стековой памяти, обращение к недействительному ементу массива, неправильная работа с указателем и т. п. Требование эффективности построения компилированного кода приводит к тому, что обаружение ошибок этого типа при отладке затруднено. Современные средства выполнения компиляции могут добавлять для режимов отладки кодые фрагменты, которые обнаруживают во время исполнения программы явление динамических ошибок заданного типа, локализуют их и связывают с вызывающим их участком исходной программы.

## Отладка аппаратуры

ппаратная отладка, так же как отладка программного обеспечения, может ыполняться на всех трех видах реализации системы: виртуального прототиа, реального прототипа и реальной системы. Наличие в распоряжении проктировщика моделей отдельных фрагментов проектируемой системы позволяет легко создавать на экране дисплея желаемые конфигурации системы моделировать ее будущее поведение. Для ИСПС сгенерированная модель может использоваться при создании конфигурации реальной системы.

Все виды ошибок проектирования моделирование позволяет обнаружить в редких случаях, поэтому только эксперименты с реальными образцами могут дать уверенность в достигнутых характеристиках системы. Далеко не все проектируемые системы содержат такое количество собственных ресурсов,

которое позволяет легко и просто организовать проведение экспериментальных работ, осуществлять быструю коррекцию аппаратных и программных решений. Поэтому широкое распространение получают *методы удаленной отладки*. Отладочные средства сконцентрированы при такой организации экспериментов около компьютера ведущей системы (Host-системы), а соединение ведущей системы с отлаживаемой системой (Target-системой) позволяет достаточно просто создавать любые исследовательские комбинации.

## Методы комплексной аппаратно-программной отладки

Исторически первым появился и использовался принцип EPROM-прототипирования. Идея метода иллюстрируется рис. 2.16. Отладка сводится к тому, что исследуемая программа, оттранслированная в ПК, при помощи программатора зашивалась в БИС программируемой тем или ином способом схемы ПЗУ (EPROM, OTP и т. д.). После этого ПЗУ помещалось в отлаживаемую систему, на ней выполнялись эксперименты, на основе данных экспериментов принималось решение о целесообразных направлениях изменений разрабатываемой программы, и процесс повторялся.

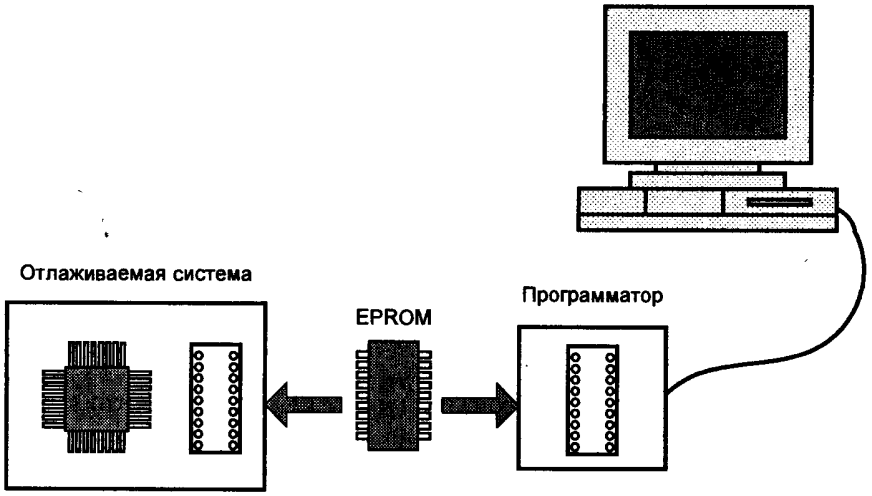


**Рис. 2.16.** Метод EPROM-прототипирования

Метод применялся и для памяти программ, внешней относительно МП/МК, и для встроенной в МК. Более перспективной представляется модификация метода, приведенная на рис. 2.17. Она опирается на широкое распространение в современных программных системах электрически перепрограммируемой памяти типа Flash и методов внутрисистемного программирования (In System Programmability, ISP) подробно рассматриваемых в *разд. 2.6.*



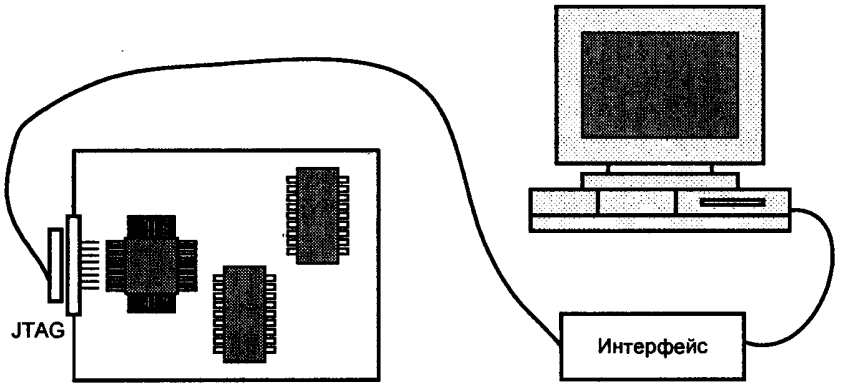


**Рис. 2.17.** Современная модификация метода EPROM-прототипирования

том методе объединение отлаживаемой системы с отладочным ПК через терфейсный блок позволяет многократно производить вышеописанную адочную процедуру без необходимости вынимать из отлаживаемой сис-ы программируемый элемент (ПЗУ или МК). Вариант интерфейсного ка определяется протоколом, который поддерживается программируе-м элементом. Еще совсем недавно практически отсутствовали средства утрисхемного программирования и соответствующие методики. Это при-о к распространению промежуточного варианта реализации, носящего вание *эмуляция ПЗУ*. В основе метода (рис. 2.18) лежит использование циального оборудования — эмулятора ПЗУ. Базой эмулятора является ИС ОЗУ, содержимое которого загружается из отладочного ПК. В режиме уляции, когда разъемный элемент эмулятора вставляется в цоколь схемы У (расположенный на отлаживаемой плате), ОЗУ эмулятора выполняет нкции удаленного на момент отладки ПЗУ. В зависимости от сложности пользуемого эмулятора, входящее в его состав дополнительное оборудова-е позволяет выполнять ряд сервисных функций. В состав отладочных йствий может входить, например, сохранение и передача в отладочный мпьютер трассы исполняемых команд (их последовательность). Эмулятор жет фиксировать значения данных, появляющихся на шинах адреса, и нных отлаживаемой системы, а также формировать сигналы прерывания я отлаживаемой системы при выполнении условий, задаваемых разработ-иком из инструментального ПК. Реакция на прерывания при этом опреде-ется содержимым обрабатывающих подпрограмм, временно подставляе-ых эмулятором.

современных условиях диапазон отладочных функций может быть суще-твенно расширен, а их содержание усложнено, если в качестве элемента, правляющего работой эмулятора, используется ПЛИС, конфигурация кото-ой определяется отладочным ПК и может оперативно им модифицировать-я. Недостатком метода является необходимость выполнения целого ряда

условий, включающих требование доступности памяти программ, допустимость блокировки штатной памяти программ и возможность подключения эмулятора. Перспективным направлением, опирающимся на специфику современных ПЛИС, является встраивание элементов эмулятора ПЗУ в БИС ПЛ, реализующую функции набора интерфейсных схем микропроцессорной системы. Для отладочных целей в качестве интерфейсной схемы может быть выбрана БИС с логическими ресурсами, превышающими последующую серийную реализацию.

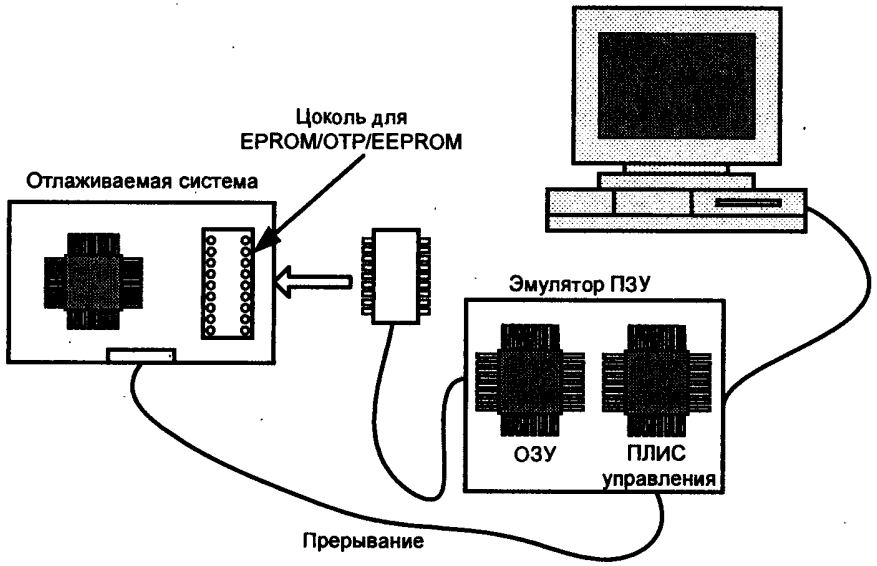


**Рис. 2.18.** Метод эмуляции ПЗУ

Недостатки, свойственные предыдущим методам, заставили разработчиков в ряде ситуаций ориентироваться на *методы встроенного монитора*. Идея метода иллюстрируется рис. 2.19. Хотя внешне соединение отладочного ПК и отлаживаемой системы похоже на соединение, характерное для метода внутрисхемного программирования, принципы и процедура отладки другие. Во-первых, Host-система соединяется не со схемой ПЗУ, а с портом процессора, как правило, при этом используется стандартное соединение по последовательному каналу (типа RS232). Во-вторых, основные отладочные действия выполняются специальной программой-монитором, включенной в состав программного обеспечения отлаживаемой системы. Ресурсы отладочного компьютера используются для отображения контролируемых параметров отлаживаемой программы, для корректировки (при необходимости) этих параметров и для управления запуском/остановкой отлаживаемой программы.



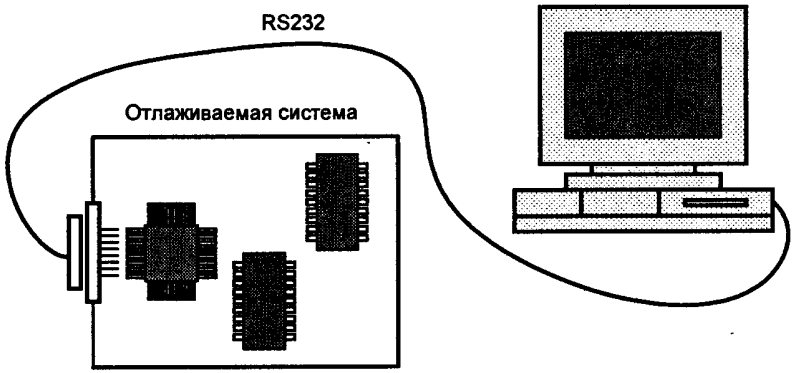


**Рис. 2.19.** Метод встроенного монитора

авным требованием к отлаживаемой системе (помимо, естественно, нали- свободного последовательного порта) является возможность организа- прерывания работы отлаживаемого программного обеспечения в зада- мых разработчиком местах. Прерывание происходит в тот момент, когда полнение программы достигает одной из задаваемых из Host-системы то- останова (break point). Удобнее всего для организации точек останова пользовать команды программных прерываний, вставляемые вместо ис- няемой команды. Это относительно легко реализуется при отладке про- мм, помещенных в оперативную память. Программное прерывание пере- т управление монитору. Набор выполняемых монитором действий опре- яется далее разработчиком и зависит от возникшей ситуации.

тод удобен для тех систем, у которых не предусмотрены или отсутствуют бственные средства ввода/вывода, которые могли бы служить для получе- я отладочной информации. Долгое время этот метод оставался практиче- и единственным дешевым и допустимым методом для отладки микроконоллерных систем, у которых все исполнение основной программы проис- дит без обращения к каким-либо внешним ресурсам (соответственно, нет, пример, внешнего ПЗУ команд, и поэтому невозможно использовать эму- торы ПЗУ и т. д.).

временный подход к реализации встроенных мониторов базируется на траивании в систему отладочных агентов. Отладочные агенты представляют бой управляемые программные вставки, которые в определенных условиг могут перехватывать работу основной программы и передавать управле- е отладочным фрагментам. Основной проблемой является влияние отла- чных фрагментов на штатную работу системы. Наличие вставок позволяет нтролировать работу системы, однако может приводить к некоторой поте- производительности. Удаление/вставление отладочных агентов может из- енять временные соотношения для отлаживаемых ситуаций. Приемлемым читается постоянное присутствие отладочных агентов при потере произво-

дительности, не превышающей диапазон значений от 1 до 5%. Наличие отладочных агентов оказывается единственным источником информации при полных отказах системы.

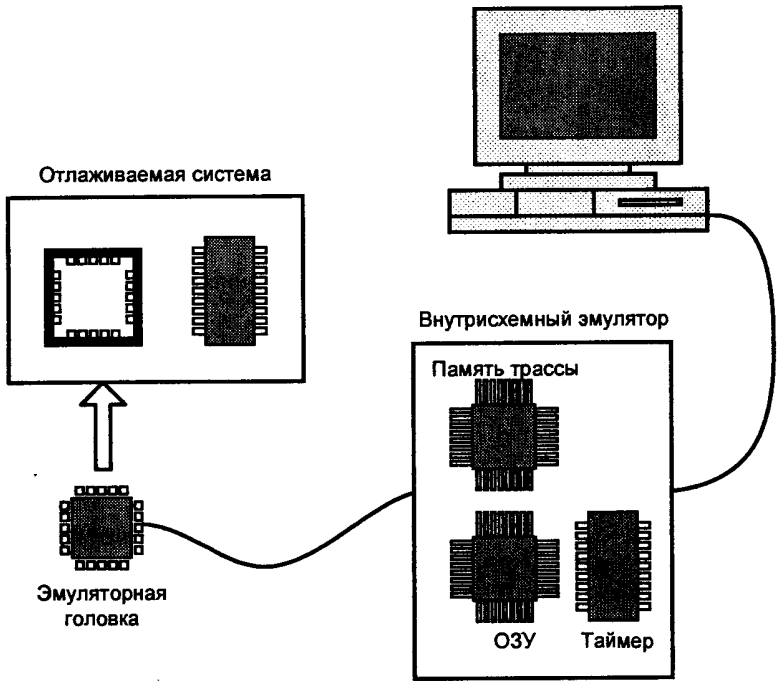


**Рис. 2.20.** Методы внутрисхемной эмуляции

Сложность отладки программ управляющих систем в реальном времени привели к широкому распространению *методов внутрисхемной эмуляции* (ICE, In Circuit Emulator). Реализация метода показана на рис. 2.20. Метод опирается на тесное объединение ресурсов отлаживаемой и отлаживающей систем. Объединение осуществляется за счет помещения в цоколь МП или МК специальной эмуляторной головки, содержащей схему, эмулирующую МП/МК отлаживаемой системы, и соединенной далее с собственно эмулятором. Метод состоит в управлении со стороны отладочного МП направлением подключения шин отлаживаемого МП. Внутрисхемная эмуляция может выполняться с различной полнотой эмулируемых ресурсов. Содержимое эмулятора позволяет оперативно модифицировать состав элементов ведущей системы, берущих на себя работу (эмулирующих работу) отлаживаемой системы. В результате выполнение отлаживаемого программного фрагмента может осуществляться, опираясь на любую задаваемую разработчиком комбинацию ресурсов отлаживаемой и отлаживающей систем. В предельном случае содержимое эмулятора заменяет даже процессор отлаживаемой системы, для этого эмулятор должен содержать симулятор системы команд МП отлаживаемой системы. Наиболее сложно решается проблема использования методов ICE для однокристальных микроконтроллеров, поскольку большинство деталей поведения МК остается внутри кристалла и недоступно для наблюдения. Поэтому в некоторых случаях фирмы выпускали помимо базового варианта МК специальную версию (для общецелевого использования слишком дорогую) эмуляционного кристалла. Основным отличием кристалла (кроме стоимости) является наличие дополнительных контактов, на которые выводятся сигналы внутренних шин (адреса, данных и управления).



Высокая стоимость традиционных приборов внутрисхемной эмуляции заставляет проектировщиков искать более дешевые решения. Интересным и перспективным представляется использование в качестве основы внутрисхемного эмулятора ИСПС. Способность ПЛИС изменять структуру создает возможность менять тип эмулируемого МП за минуты, легко добавлять новые типы МП, конфигурировать саму систему эмулятора.

Наиболее перспективным методом отладки встроенных микропроцессорных устройств (в общем случае не только их) являются методы встроенной внутрикристальной отладки. Эти методы наибольшее распространение получили в современных 32-разрядных МП. Структура кристалла, содержащего встроенные средства отладки аппаратно-программных ресурсов, приведена на рис. 2.21. Но даже у 8-разрядных процессоров известны варианты встраивания отладочных средств в кристалл. Примером может служить продукция фирмы Cygnal Integrated Products (**www.cygnal.com**), в которой комбинация свойства внутрисистемной программируемости (ISP), внутренней Flash-памяти и встроенных в кристалл специальных отладочных средств существенно облегчает процедуру отладки. Другим примером того же подхода является продукция фирмы Atmel (**www.atmel.com**). Выпускаемые этой фирмой AVR-процессоры имеют 8-битовую архитектуру и поддерживают JTAG-интерфейс. В состав кристалла включены отладочные средства, которые существенно упрощают процесс отладки.

Более простыми вариантами являются методы соединения отладочных Host-ПК с отлаживаемой системой через параллельные или последовательные порты. В качестве приемника в целевой системе могут служить специальные ИС или выделенные для этих целей контакты МП. В качестве стандарта подключения используется либо стандарт JTAG, либо разработанный фирмой Motorola интерфейс со встроенным в МП отладочным модулем (в терминологии фирмы режим отладки на его базе носит название BDM, Background Debug Mode). Основными элементами внутрикристальной отладки являются интерфейсный блок, блок формирования точек останова и средства доступа к отладочной (процессорной) информации из интерфейсного блока.

Данные по эффективности различных отладочных действий для рассмотренных методов организации отладки сведены в табл. 2.4. Сравниваемыми

показателями являются возможность быстрого и оперативного изменения объектного кода отлаживаемой программы, доступность и удобство наблюдения за состоянием основных программных и аппаратных объектов, эффективность управления (легкость корректировок) вышеуказанных объектов, возможность определения аналоговых эффектов на дискретных сигналах и, наконец, цена отладочного средства.

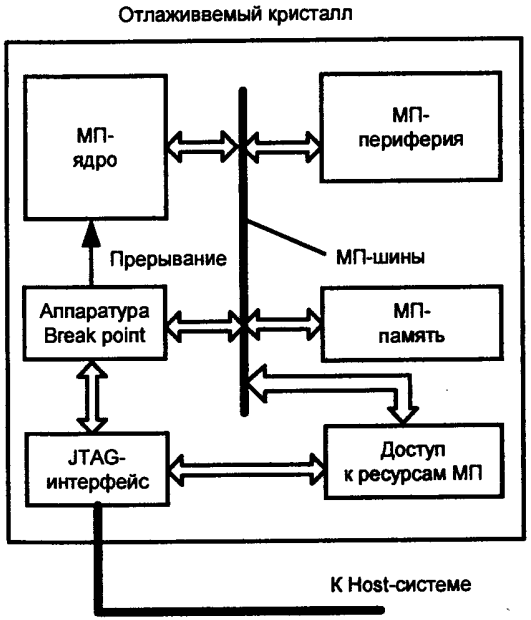


**Рис. 2.21.** Методы внутрикристальной отладки

***Таблица 2.4.*** *Данные по эффективности различных типов отладки*

| | Изменение кода | Наблюдаемость | Управляемость | Аналоговые эффекты | Цена |
|---|---|---|---|---|---|
| EPROM-прототипирование | Нет | Низкая | Отсутствует | Низкая | Низкая |
| Монитор в системе | Нет | Средняя | Средняя | Низкая | Низкая |
| Внутрисхемная эмуляция | Есть | Очень высокая | Полная | Высокая | Высокая |
| Внутрикристальная отладка | Есть | Высокая | Полная | Низкая | Низкая |



Для организации и проведения более тонких экспериментов незаменимыми оказываются такие средства, как запоминающие осциллографы, цифровые осциллографы, логические анализаторы или специализированные тестирующие системы. Основным недостатком этого класса оборудования является большая стоимость при эпизодичности использования. Современные приборы с перестраиваемой структурой создают альтернативное направление развития отладочных средств. Например, программа и небольшая плата расширения превращает ПК в многоканальный цифровой или даже аналоговый запоминающий осциллограф. Связка — малоресурсная отлаживаемая система (содержащая средства внутрикристальной отладки) и мощная компьютеризованная отладочная система — обеспечивает эффективность процедуры отладки, почти не уступающую эффективности отладки при помощи внутрисхемных эмуляторов. Широкая номенклатура выпускаемых прототипных плат и их разнообразие — еще один существенный фактор ускорения процесса проектирования. Платы выпускаются практически одновременно с первыми образцами кристаллов.

## 2.2.3. Проектирование систем, связанных с обработкой аналоговых сигналов

Проектирование фрагментов систем, связанных с обработкой аналоговых сигналов, было и остается одной из важнейших и сложнейших проблем. Окружающий нас мир является в значительной мере аналоговым (непрерывным), и не удивительно, что для 10% проектов необходима обработка аналоговой информации. По некоторым оценкам к 2006 году количество таких проектов достигнет 30%. Работы этого направления требуют решения как проблем чисто *аналоговых* (масштабирование, фильтрация и т. д.), так и проблем стыковки проектов с аналоговым и дискретным представлением информации, т. е. проблем аналого-цифрового и цифроаналогового преобразований.

Так же, как и для других направлений проектирования, базовыми этапами являются этапы ввода информации, компиляции и моделирования. Проектирование по этой ветви характеризуется следующими отличительными особенностями.

За последние годы существенно изменилась элементная база. Успехи микроэлектронной промышленности привели к тому, что набор широкой номенклатуры ИС переместился в область более высокой точности и более быстродействующих приборов. Современные АЦП и ЦАП при интегральной реализации достаточно уверенно дают 12—14-разрядную точность при времени преобразования в единицы и доли микросекунд. Точность и быстродействие многих аналоговых датчиков и исполнительных механизмов не требует больших значений указанных параметров. Поскольку выбор элементной базы предопределяет не только эксплуатационные характеристики проекта, но и в значительной мере его конструктивную реализацию, то су-

щественно возросли и требования к выполнению конструкторского этапа проектирования.

## Проектирование аналоговых фрагментов

В основе современных САПР (уже более 30 лет) для моделирования аналоговых фрагментов остается пакет Spice в различных модификациях. Большинство фирм-разработчиков систем моделирования сохраняют совместимость своих разработок со стандартами пакета Spice. Изменения касаются адекватности и точности отражения в моделях свойств реальных элементов. Объемы библиотек постоянно увеличиваются, а их содержимое модернизируется. Возрастает качество работы алгоритмов моделирования (сходимость результатов и их адекватность), уменьшается время моделирования. Как следствие, уменьшаются ограничения на сложность моделируемых систем. Конечно, улучшаются интерфейсные функции пакетов.

На окончательные характеристики аналоговых систем конструктивная реализация оказывает большое существенное влияние, чем при реализации цифровых систем. Возможность получить неработоспособную плату (успешно прошедшую испытания в прототипной реализации) приводит к необходимости моделирования проектов с учетом эффектов, вносимых конкретной реализацией. При моделировании учитываются паразитные сопротивления, паразитные емкости монтажа. Более мощные САПР учитывают распределенный характер этих паразитных элементов.

Проблемой остается конструктивная реализация аналоговых фрагментов. Имеется весьма существенная специфика проектирования и изготовления печатных плат. При конструктивной реализации требуется сохранить точность и быстродействие и к минимуму свести влияние паразитных факторов (электромагнитные наводки, паразитное падение напряжения, перекрестные помехи и т. д.). Маршруты проектирования аналоговых фрагментов систем на современных САПР при ориентации на дискретные компоненты или современные схемы ПАИС показаны на рис. 2.22.

Для проектирования аналоговых фрагментов систем на базе дискретных аналоговых компонентов могут использоваться САПР различных фирм. Наибольший интерес среди этой продукции представляют САПР, позволяющие совместить выполнение конструкторской разработки печатной платы с моделированием проекта. Среди отечественных разработчиков наибольшее распространение получили разработки фирмы MicroSim Corporation (с 1998 г. входит в состав фирмы OrCAD Corporation). Привлекательность последних версий ее САПР — DesignLab обусловлена двумя основными факторами.

Во-первых, система позволяет моделировать аналоговые фрагменты, разводить печатные платы и моделировать поведение фрагмента с учетом паразитных параметров разводки. Более того, САПР позволяет выполнять моде-



лирование для цифровых (включая простейшие схемы ПЛИС) и смешанных аналого-цифровых устройств. Дополнительно может использоваться программа Filter Designer, осуществляющая синтез пассивных и активных аналоговых фильтров и фильтров на переключаемых конденсаторах. Несомненным достоинством САПР является наличие библиотеки, включающей 40 тыс. графических обозначений символов и около 10 тыс. математических моделей компонентов.

Во-вторых, работа с DesignLab хорошо изложена в отечественной литературе [22].

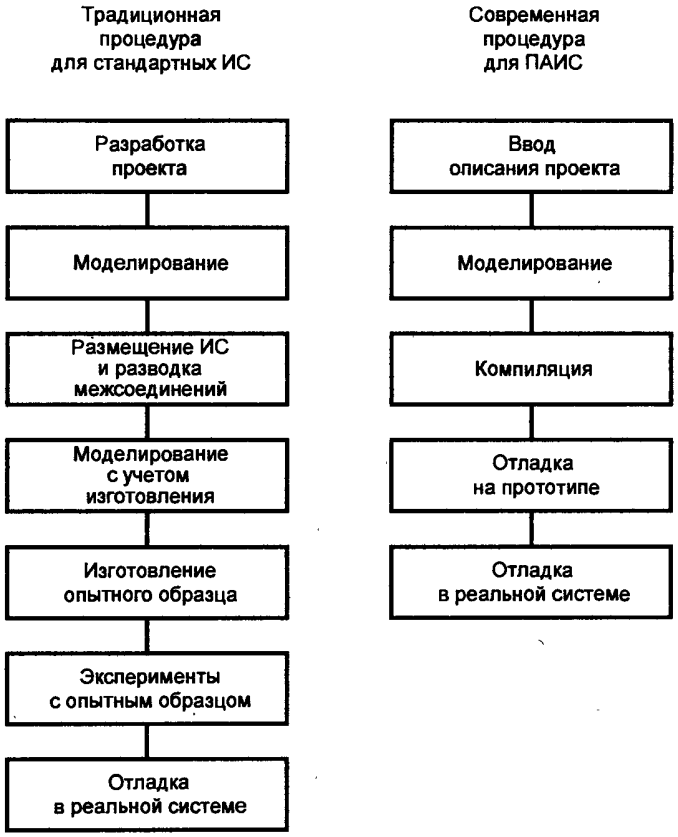


**Рис. 2.22.** Маршруты проектирования для аналоговых или аналого-цифровых фрагментов систем

При реализации аналоговой части проекта в виде отдельной БИС ПАИС существенно снижаются требования к профессиональным качествам проектировщика, поскольку паразитные эффекты размещения и разводки компо-

нентов фрагмента учтены разработчиками БИС ПАИС. Для проектирования таких ИС фирмы-разработчики БИС ПАИС выпускают специальные САПР. Для проектирования на базе ПАИС ispPAC фирма Lattice Semiconductor выпустила САПР под названием PAC Designer. Функционирование аналоговых программируемых блоков микросхемы CY8C25/26, выпускаемых фирмой Cypress Semiconductor, задается при конфигурировании кристалла. Фирма Cypress Semiconductor обеспечивает поддержку проектирования своей продукции на базе различных версий САПР Warp. Наиболее мощная версия пакета Warp Enterprise (2995 долларов) поддерживает все стадии проектного потока, включая сервисные функции типа конвертеров графического ввода в HDL-описания, моделирование поведения на уровне исходного описания, интерактивной отладки и автоматической генерации создания тестовых последовательностей (Test-Bench).

Еще более интересным представляется подход фирмы ZETEX к проектированию и конвертации аналоговых средств обработки. Выпускаемое фирмой ZETEX семейство ПАИС TRAC (Totally Re-configurable Analog Circuit) относится к семейству кристаллов с настраиваемой структурой отдельных аналоговых ячеек. Загрузка ОЗУ конфигурации ПАИС TRAC выполняется за счет подключения схем TRAC к стандартным ПЗУ через специальную схему TRAC-S2. Образование требуемой аналоговой схемы производится путем внешней коммутации, обычно выполняемой на печатной плате. Подобная организация соединения ячеек между собой снимает проблему разработки фирмой специального САПР, решающего вопросы размещения ячеек и трассировки межсоединений. Чтобы уменьшить влияние межсоединений ячеек кристалла TRAC на окончательные характеристики системы, фирма ZETEX организовала выпуск аналоговых схем ZXF36Lxx, относящихся к классу Computational Application Specific Integrated Circuit (CASIC). Схема ZXF36Lxx состоит из аналоговых ячеек, аналогичных по структуре ячейкам кристаллов семейства ZXF36Lxx. Функции, выполняемые ячейками, и конфигурация межсоединений кристалла определяются топологией двух слоев металлизации. Маршрут проектирования, используемый фирмой ZETEX при конвертации проектов из представления в форме ПАИС в форму полузаказной схемы, приведен на рис. 2.23.

Маршрут проектирования опирается на комбинирование TRAC-технологии с технологией изготовления ASIC-продукции. Проектирование по технологии предполагает предварительное макетирование проекта на схемах TRAC-семейства. Сохранение в полузаказной схеме структуры ячеек обеспечивает при конвертации совпадение характеристик прототипа и конечного продукта, а также делает конвертационный процесс практически свободным от угрозы нереализуемости. Вместе с тем, как видно из рисунка, конвертационный процесс фирмы включает не только имитационное моделирование, но и предполагает тестирование изготовленной продукции на двух уровнях.



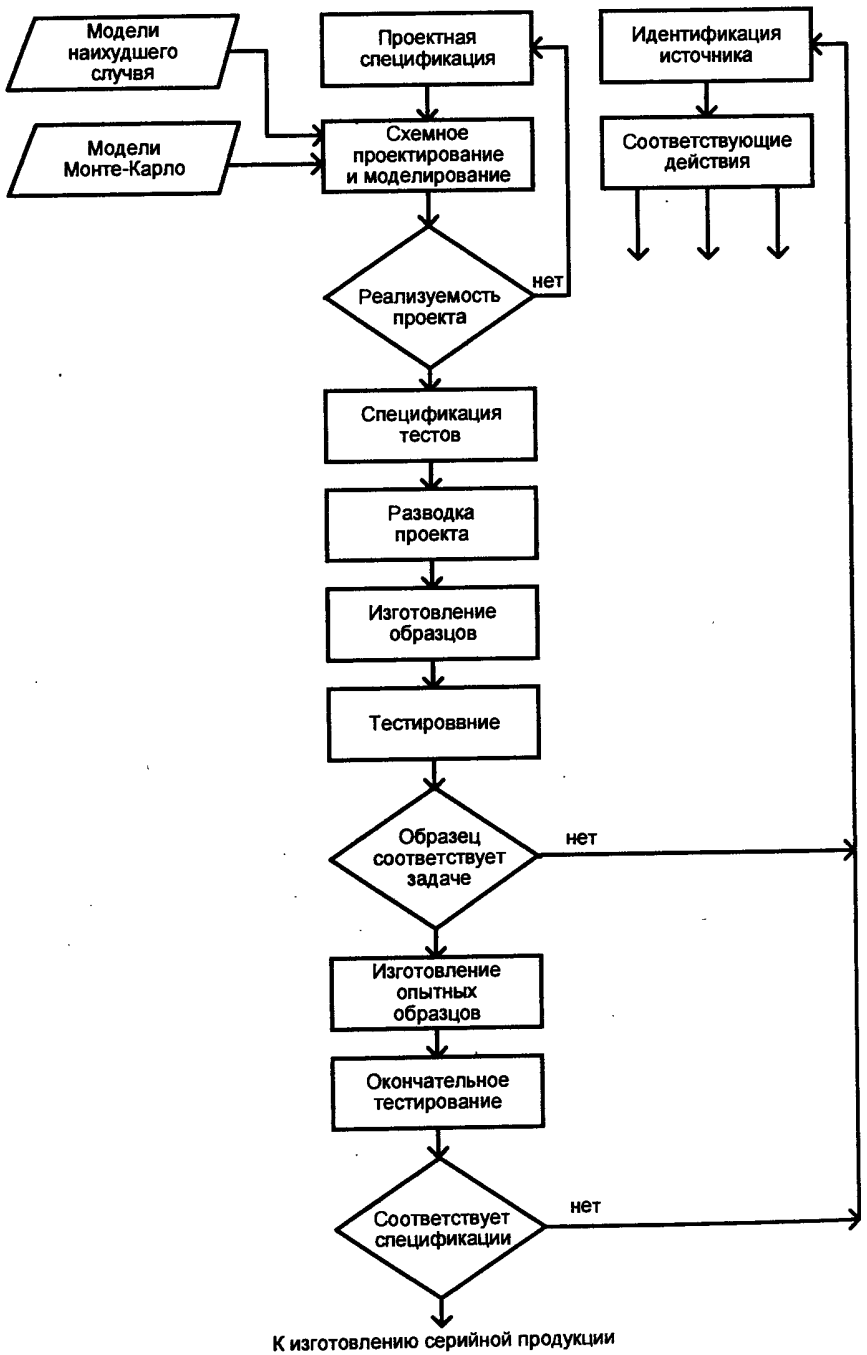


**Рис. 2.23.** Проектный поток конвертации проектов фирмы ZETEX

## Проектирование фрагментов со смешанным представлением сигналов

Наибольшие изменения в последнее время (по мнению многих исследователей — за последние два года) произошли среди средств проектирования систем со смешанным представлением сигналов, причем, основные изменения — в сфере моделирования и в области создания и применения средств тестирования. Большое количество ссылок можно найти в обзорной статье [47].

Изменения в методике проектирования прежде всего касаются тщательной верификации проекта на схемотехническом уровне. Одновременное моделирование цифровых и аналоговых сигналов уже значительно сложнее, чем чисто цифровое или чисто аналоговое моделирование. К проблемам моделирования добавляется проблема использования языков описания систем такой смешанной природы (хотя здесь имеются значительные успехи — это рассмотрим чуть ниже). В части эффективности средства проектирования и моделирования систем со смешанными сигналами значительно уступают аналогичным средствам для цифровых систем. Но современное состояние характеризуется значительной динамикой, и можно ожидать существенного прогресса как в области создания новых типов SOPC, так и в области разработки средств проектирования.

Информация о продукции фирм, связанных с поддержкой различных этапов проектирования аналоговых и аналогово-цифровых систем, может быть найдена в обзорных статьях интернет-журналов. Анализ этих статей демонстрирует неослабевающий интерес различных фирм к данному направлению. Сложность решения проблем для систем со смешанным представлением сигналов находит свое отражение, прежде всего, в уровне цен на поставляемые САПР. Стоимость САПР значительно превышает стоимость САПР аналогичной целевой направленности для цифровых систем. Приведенные данные показывают, что усилия разработчиков направлены на автоматизацию различных этапов проектирования.

Поддержка проектировщиков начинается с момента анализа путей реализации проекта. Уже на этом этапе фирмы предлагают свои готовые решения, поставляемые в форме моделей стандартных решений или генераторов моделей.

Основные усилия последнего времени направлены на повышение эффективности совместного моделирования цифровых и аналоговых схем. Методы событийного моделирования (подробно рассмотренные в *разд. "Моделирование и реальное время" гл. 3*), весьма эффективные для моделирования цифровых схем, оказываются неприемлемыми для аналоговых и аналого-цифровых, ибо в последних изменениях сигналов (события) происходят непрерывно. Системы моделирования устройств со смешанным представлением информации часто строят, используя объединение событийного подхода для представления цифровой подсхемы и численного интегрирования дифференциальных уравнений, являющегося основой пакета Spice. Необходимость такого подхода становится особенно важной для проектов, в которых в одном кристалле объединяются сотни тысяч вентилей программируемой логики и аналоговые цепи. Типовыми проблемами для таких проектов являются, например, задачи проектирования для АЦП высокого разрешения или ускоренного моделирования работы фазово-управляемых схем (в английской терминологии Phase-Locked Loop, PLL).



Даже беглый анализ данных, приведенных в обзорных статьях интернет-журнала EDN (**www.ednmag.com**), показывает, что одной из ведущих фирм в области создания средств проектирования систем со смешанным представлением сигналов является фирма Antrim Design System — разработчик САПР Antrim-A/MS. Этот пакет (рис. 2.24) помимо традиционных для систем

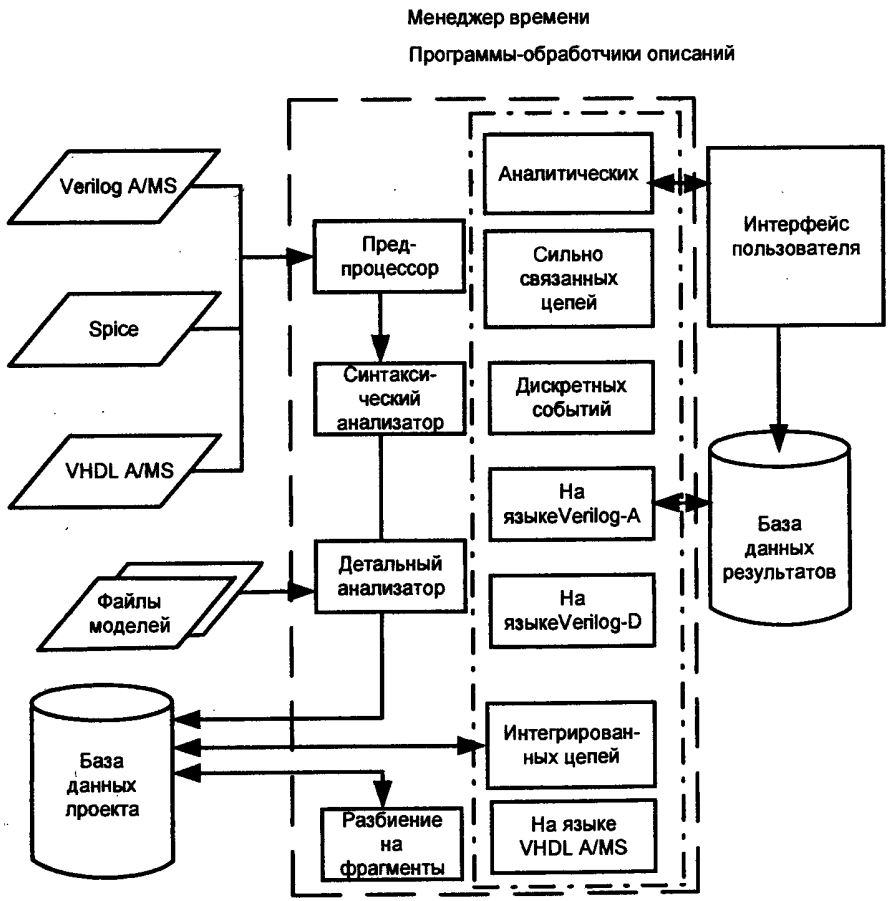


**Рис. 2.24.** Состав САПР Antrim-A/MS фирмы Antrim Design System

моделирования блоков содержит богатый набор программ-обработчиков для различных способов описания аналоговых, цифровых и смешанных блоков. В состав этого набора входят: релаксационные событийно управляемые средства моделирования для КМОП цифровых цепей, описанных на вентильном уровне; средства интерпретации Verilog-описаний для цифровых блоков; программы-обработчики описаний межсоединений и шин питания; средства моделирования сильносвязанных аналоговых цепей, реализующие решение систем дифференциальных уравнений по методу Ньютона—Ральфсона, и средства моделирования аналоговых блоков, описываемых поведенческими моделями на языке Verilog-A.

Смешанные способы описания и многоуровневое моделирование поддерживаются не только САПР Antrim-A/MS, но и разработками других фирм. Выгоды многоуровневого подхода состоят в возможности моделировать одну часть схемы на уровне физических моделей (например, операционные усилители), а другую на событийном (например, схемы логического управления).

Очень полезным свойством систем рассматриваемого класса, хотя присущим, к сожалению, не всем, является совместимость процедур моделирования и автоматического размещения и разводки. В отличие от нисходящего проектирования цифровых систем при их реализации в форме микросхемы, когда процесс проектирования от архитектурного определения до физического расположения на кристалле легко поддается автоматизации и опирается на ранее разработанные САПР, что существенно ускоряет процесс проектирования, аналоговое проектирование не является столь простым и легким. Иллюстрацией этого положения является приведенный на рис. 2.25 типичный маршрут проектирования аналогового блока, встраиваемого в кристалл. Из рисунка видно, что проектирование без ручной доводки проекта обходится в редких случаях.

Наибольший интерес представляют попытки разработки средств, автоматизирующих наиболее ответственный этап проектирования — этап синтеза устройств, исходя из описания их функционирования. Различие алгоритмов, необходимых для синтеза аналого-цифрового преобразователя или операционного усилителя, делает задачу разработки САПР, поддерживающих синтез систем со смешанным представлением информации, весьма проблематичной. Вместе с тем, уже упоминавшаяся выше фирма Antrim анонсировала выпуск САПР, поддерживающей синтез аналоговых приборов.

Не меньше проблем возникает и при попытках автоматизировать этапы проектирования, отвечающие за создание пригодной к тестированию конечной продукции, а также за тесты и аппаратуру для такого тестирования, однако на этом направлении полученных положительных результатов несколько больше. Интерес представляют разработки фирм OpMaxx и LogicVision в части разработки САПР, генерирующих тесты для проверки аналоговых и



гибридных систем. Фирма SZ Testsysteme традиционно поддерживает выпуск аппаратуры, предназначенной для тестирования аналоговых и смешанных систем различного класса.

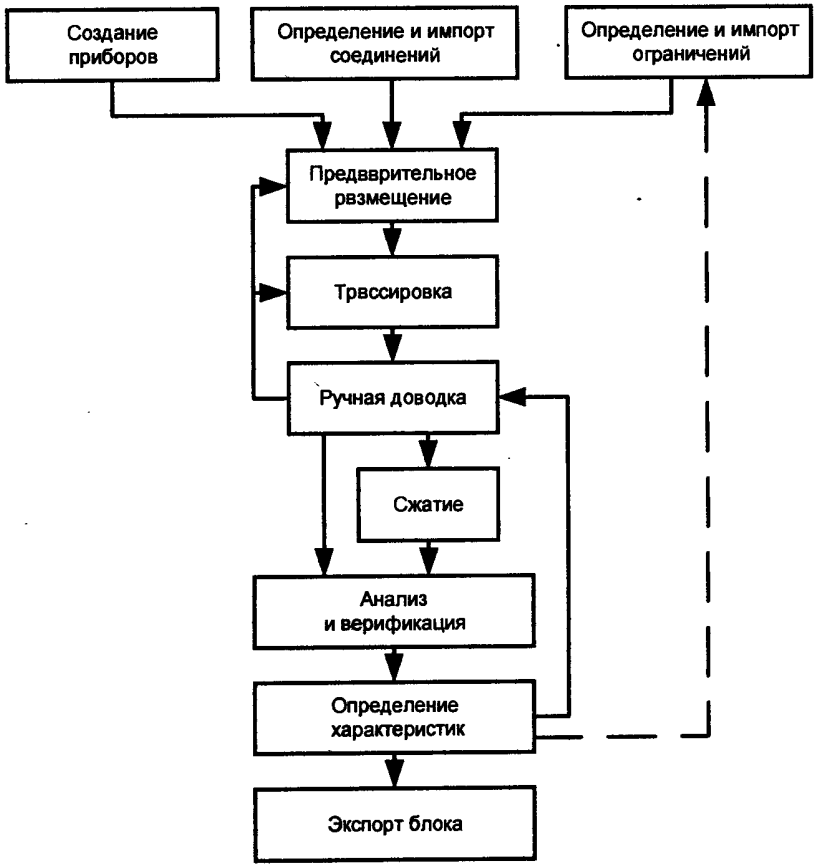


**Рис. 2.25.** Маршрут проектирования аналогового блока

Существенное значение для развития средств автоматизации различных этапов проектирования имеют работы, направленные на создание базы такой автоматизации — разработки языков, предназначенных для описания смешанного представления сигналов.

Широкое распространение получает использование для описания систем со смешанным представлением сигналов расширенных вариантов языка Verilog и VHDL ("чисто цифровые" версии этих языков рассмотрены далее). После почти десятилетней подготовки в 1999 году были приняты предложения по стандартам Verilog-AMS и VHDL-AMS. Принятые расширения этих языков позволят проектировщикам использовать для своих аналоговых блоков и

блоков со смешанным представлением информации поведенческие модели, которые оказываются совместимыми с поведенческими моделями цифровых блоков, описываемых стандартными средствами Verilog и VHDL. Прежде чем стать стандартами, новые варианты языков должны были преодолеть ограничения, обусловленные правами собственности ряда фирм, в том числе решениями, уже введенными в такие пакеты, как MAST фирмы Analogy, SpectreHDL фирмы Cadence, HDL-A фирмы Mentor Graphics, Diablo фирмы VeriBest. Подобно их цифровым прототипам аналоговые и смешанные поведенческие модели обеспечивают большую скорость моделирования, чем модели приборного или вентильного уровня. Дополнительной выгодой поведенческих моделей является свойственная им относительная секретность, поскольку они не показывают, как будет реализован описываемый ими блок.

После долгих дебатов в 1999 году был принят стандарт IEEE Standard 1076.1-1999 языка VHDL-AMS, являющийся расширением известного стандарта IEEE 1076 языка VHDL. Язык позволяет описывать сложные проекты, используя комбинацию дифференциальных выражений, алгебраических ограничений и логических условий. Смешение аналоговых и событийно управляемых элементов в одном языке оказывается более эффективным, чем использование макромоделей в Spice. Поскольку VHDL-AMS позволяет описывать поведение и неэлектрических элементов, то он может использоваться для систем, смешивающих электрические и неэлектрические компоненты, например, электромеханические и жидкостные электрические системы. Многие компании-разработчики САПР, в том числе ведущие фирмы Analogy и Mentor, объявили о своей поддержке нового стандарта VHDL-AMS и после его утверждения готовы выпустить новые САПР, основанные на этом стандарте.

Аналогичная ситуация складывается и с языком Verilog. Его расширенный вариант стандарта — Verilog-AMS standard 1.3, выпущенный Open Verilog International organization (**www.ovi.org**), также был принят в 1999 году. Подобно VHDL-AMS язык Verilog-AMS поддерживает включение в описание системы неэлектрических компонентов (механических, оптических и т. д.). Большинство фирм разработчиков САПР для смешанных систем, включая Analogy, Antrim, Apteq, Cadence, Mentor, Transcendent, и Innoveda, объявили о включении в будущую выпускаемую продукцию средств поддержки языков Verilog-AMS, VHDL-AMS или обоих.

## Проектирование аналоговых и аналого-цифровых SOPC

Существующая в настоящее время в микроэлектронике тенденция интеграции в одном кристалле всех элементов современных систем, находящая свое воплощение в форме выпуска "систем на кристалле" (SOPC), окажется не



полной, если не будет включать в свой состав аналоговых и/или аналого-цифровых блоков. Включение ПАИС в SOPC дополнительно дает следующие выгоды, характерные для всех ИСПС — возможность модернизаций и корректировок в весьма значительных пределах, определенная секретность разработок и их защита от несанкционированного копирования. Дополнительная особенность ПАИС состоит в том, что модернизации и корректировки фрагмента допустимы не только в форме изменения взаимного соединения элементов (т. е. структуры фрагмента), но и форме изменения номиналов используемых элементов. Таким образом может достигаться весьма существенное изменение как характеристик, так и поведения фрагмента.

Соединение в одной системе цифровых, аналоговых и цифроаналоговых фрагментов имеет свою историю развития. Первые успешные попытки объединения в одном кристалле цифровых и цифроаналоговых фрагментов осуществлялись фирмой Intel еще в середине 80-х годов, дальнейшее развитие это направление получило у фирмы Analog Device. Сложности реализации требуемой совмещенной технологии не обеспечивают желаемых темпов развития этого направления развития микроэлектроники. Очень немногие фирмы сейчас могут поддерживать (на условиях экономической рентабельности) выпуск устройств со смешанной технологией.

Наибольших успехов к настоящему моменту достигла фирма Cypress MicroSystems, Inc. (филиал Cypress Semiconductor). Производимые фирмой схемы семейства CY8C25***/26*** представляют собой БИС, в основе которых микропроцессорная часть (8-разрядный процессор M8C) и программируемая пользователем часть (набор реконфигурируемых аналоговых и цифровых блоков).

Фирма выпустила САПР — PSoC Designer IDE, являющуюся, по сути, интегрированным пакетом средств разработки встраиваемых SOPC-систем. Пакет обеспечивает доступ ко всем традиционным средствам отладки микропроцессорных систем, включая полный отладчик, поддерживающий внутрисхемную эмуляцию ICE. На предварительном этапе проектирования и для моделирования проектов, содержащих смешанное аналого-цифровое представление сигналов, фирма Cypress предлагает пользоваться ПО фирмы Antirim Design System (Antirim-ACV, Antirim-MSS). Между этими фирмами достигнуты соответствующие лицензионные соглашения, позволяющие надеяться на объединение усилий обеих фирм в направлении интеграции проектных средств в форме единого пакета.

Системы автоматизированного проектирования аналоговых и аналого-цифровых ИСПС еще находятся на начальном этапе своего становления. Следует ожидать возрастания объема и качества работ, выполняемых этими САПР.

# 2.3. Структура и организация САПР

## 2.3.1. Связь процедуры проектирования и САПР БИС программируемой логики

Как уже отмечалось, разработка современных вычислительных систем, содержащих как микропроцессорные решения, так и БИС с программируемой структурой, немыслима без широкого привлечения САПР. Все методики проектирования таких вычислительных систем ориентированы на использование САПР. Правильный выбор САПР предопределяет успешность всей процедуры проектирования и является важнейшим условием эффективного проектирования. При хорошем выборе может быть достигнуто существенное ускорение выпуска продукции, т. е. сокращение времени от спецификации проекта до выпуска первых продажных образцов (Time-to-Market), во многих случаях удается сократить и время, необходимое для выпуска серийной продукции.

Методы и средства проектирования тесно связаны с выбором САПР и, наоборот, выбор САПР определяет допустимые и целесообразные методы и средства проектирования, так что эти вопросы нельзя рассматривать в отрыве друг от друга. Взаимосвязь между САПР и выбранным типом БИС ПЛ значительно более тесная, чем между средствами проектирования МП-систем и используемым типом МП. Выбор элементной базы для проекта в значительной мере предопределяет требуемую или требуемые для реализации проекта САПР.

В общем случае, при выборе САПР приходится учитывать целый ряд соображений, а именно:

- распространенность САПР;
- цену САПР, ее сопровождения и модификаций;
- поддержку выбранной элементной базы;
- широту охвата разнообразных этапов проектирования и эффективность их выполнения;
- наличие широкой библиотечной поддержки стандартных решений;
- возможность и простоту стыковки с другими САПР;
- удобство работы с САПР и ее дружественность;
- легкость изучения;
- возможности корпоративной работы.

Очевидно, трудно найти САПР, которая удовлетворяла бы сразу всем перечисленным условиям, как ввиду их взаимной противоречивости, так и исходной неопределенности варианта реализации конечного продукта.



## 2.3.2. Связь проектной проблемы с выбором САПР

Как было показано в предыдущих разделах (рис. 2.7 и 2.12), один и тот же проект может быть реализован с ориентацией на различную элементную базу. Из рисунков видно, например, что реализация одного и того же проекта возможна как в виде раздельных проектов МП и ПЛИС, так и в форме проекта, реализуемого в БИС класса SOPC. Аналогичная ситуация складывается с выбором САПР. На первоначальных шагах проектирования можно пользоваться любым вариантом. Программное обеспечение, например, разрабатывать на ранее приобретенных версиях компиляторов, а аппаратуру проектировать при помощи оценочной версии САПР. Однако на заключительных этапах проектирования, как правило, все равно придется вернуться к версии САПР той фирмы, на элементной базе которой решено реализовать проект. Поэтому на первом же шаге целесообразно определиться — пользоваться ли САПР фирмы-производителя выбранного класса программируемой БИС ПЛ или SOPC (Xilinx, Altera, Atmel, Actel и др.) или воспользоваться услугами системы проектирования, созданной одной из ведущих фирм-производителей САПР, таких как Synopsys, Cadence, Mentor Graphics, Exemplar Logic, Viewlogic, Symplicity и др.

Чтобы отличать САПР фирм-производителей БИС ПЛ (Vendors) от САПР фирм, поддерживающих проектирование ПЛИС различных производителей, для последних обычно используется термин *сторонние САПР* (САПР независимых фирм). В английской терминологии для обозначения таких фирм принято использовать термин Third Party EDA Tools. Возможность совместного использования САПР различных фирм базируется на том, что эти САПР состоят из отдельных частей, и работа на них носит явно выраженный этапный характер (ввод, компиляция, моделирование, программирование). Тот факт, что проектные средства состоят из отдельных частей позволяет одной САПР вклиниваться в проектный поток другой. Производители схем ПЛИС, а тем более БИС SOPC, чаще всего монополизируют возможность и право монтирования проекта в конкретный тип БИС и поэтому при применении САПР сторонних фирм, как минимум, на заключительных этапах приходится пользоваться услугами САПР производителей схем ПЛИС. Ноу-хау (know-how) этих фирм обычно является взаимное соответствие между структурой соединений внутри кристалла и загрузочным файлом конфигурации.

Предельным проявлением такой стратегии является, например, САПР фирмы Agere Systems — *ORCA* Foundry 2000. САПР ориентируется на использование стандартных проектных средств сторонних фирм для решения задач большинства этапов проектирования. Разработчик использует САПР таких фирм для ввода, синтеза и моделирования проектов. Наличие в составе *ORCA* Foundry определенных интерфейсных средств позволяет загрузить в

него созданную проектную информацию. После этого основной задачей собственно САПР является разработка топологии межсоединений элементов структуры FPGA из списка цепей (netlist), ориентированного на конфигурирование FPGA. САПР размещает блоки на кристалле, осуществляет их настройку и решает задачу их взаимного соединения, используя встроенные средства (timing-driven tools). Встроенный в пакет анализатор временных задержек элементов используемой БИС (static timing analysis tool) обеспечивает определение скорости работы БИС по отдельным соединениям и возвращает аннотированный список цепей БИС, который позволяет организовать моделирование и временной анализ. Выходные файлы ORCA Foundry являются совместимыми со многими средствами моделирования сторонних фирм. Это позволяет исследовать и анализировать окончательные временные характеристики проекта. Генератор программной информации (bit stream generator) создает выходной файл, интерпретирующий структуру БИС в битовый информационный поток, который может использоваться для программирования либо БИС ПЛ, либо специальных конфигурационных БИС.

Если проектировщик решился на совместную работу с несколькими САПР (что авторам представляется в современных условиях более предпочтительным), то их выбор все равно остается достаточно многовариантной проблемой. Ни одна из САПР не является предпочтительной для любых пользовательских приложений.

Стратегия взаимодействия фирм, производящих БИС ПЛ, и фирм, специализирующихся на разработке САПР, модифицировалась во времени. До начала 90-х годов фирмы ориентировались на узконаправленную деятельность. Каждая фирма имела свою узкую специфическую область, в которой и концентрировались все ее усилия, чтобы обеспечить свое лидирующее положение в выбранной нише. В 90-х годах началось более тесное взаимодействие фирм между собой. Практически все САПР стали предусматривать возможности взаимного обмена (Import/Export Design) проектной информацией на любом этапе проектирования. Технической основой этого информационного обмена явилось введение и поддержка практически всеми фирмами стандартных языков обмена. Промышленным стандартом стал язык EDIF с версиями 2.0.0 и 3.0.0. В большинстве САПР, однако, предусмотрено взаимодействие на основе различных языков описания аппаратуры (в том числе VHDL, Verilog и т. д.), а не на основе какого-то одного языка.

С середины 90-х годов начался этап совместных разработок в области САПР. В настоящее время эта тенденция сохраняется и заключается в кооперации усилий различных фирм при разработке САПР, при этом каждая фирма реализует тот этап проектирования и, соответственно, фрагмент САПР, в котором она является общепризнанным лидером. Техническая реализация такого объединения осуществляется как в форме единой САПР, так и в форме взаимосвязанных САПР. Важно, что при этом создается *единый*



*информационный поток проектирования*, и все этапы проектирования оказываются тесно взаимосвязанными.

Способ объединения естественно влияет на окончательные характеристики САПР. Создание единой САПР задача более сложная, чем просто объединение нескольких САПР под одной управляющей оболочкой, но зато (помимо единообразия правил общения) облегчается информационный обмен между ними и, с точки зрения пользователя, значительно увеличивается объем сервисных услуг. Форма единой САПР создает более тесное общение различных САПР между собой. При этом, например, результаты моделирования работы какого-либо фрагмента схемы могут отображаться не только в диалоговом окне временного моделирования, но и в окне изображения электрической схемы, которая создавалась на этапе ввода информации о проекте.

Задача создания программ-оболочек, интегрирующих совместную работу САПР различных фирм, в последнее время несколько упростилась, поскольку разработчики САПР теперь обычно включают в них интерфейсные возможности, предоставляемые идеологией языка Tcl. В результате можно организовать совокупность взаимодействующих САПР, каждая из которых будет выступать как источник команд, так и как исполнитель (приемник) этих команд. Для этого САПР компонуется как пакет встроенных библиотек. Библиотека содержит синтаксический анализатор языка Tcl и подпрограммы, интерпретирующие как действия команды языка с параметрами, предусмотренными в САПР. Команды языка Tcl могут считываться из источников различной природы, таких как командная строка, параметры вызова из другой программы (САПР), меню выбора и т. д. Соответственно, в САПР-приемнике определено выполнение команд.

Примером *кооперационной САПР* является САПР Foundation для продукции фирмы Xilinx, которая представляет собой результат совместной разработки фирм Xilinx, Aldec и Synopsys.

Другим способом кооперации является включение в состав САПР не полной, а усеченной и специализированной версии пакета. Примером может служить технология, используемая фирмой Model Technology. Модификации моделирующей программы этой фирмы под названием ModelSim используются как составная часть САПР других фирм. Несмотря на усеченность и специализацию, версии программы предоставляют широкий набор средств для функционального и временного моделирования проектов, написанных на языках VHDL и Verilog.

Еще одной формой кооперации является включение фирмой-производителем ПЛИС в комплект поставляемого программного обеспечения оценочных версий пакетов сторонних фирм. Например, поставляемая фирмой Atmel САПР Integrated Development System (IDS) Ver. 7.2, включает оценочные версии LeonardoSpectrum (Exemplar Logic) и ModelSim (Model Technology).

Фирма Altera также позволяет (а последнее время даже рекомендует) для большей эффективности процедуры компиляции использовать на предварительных этапах САПР сторонних фирм. Проектировщик может выбрать для повышения эффективности компиляции проектов, написанных на языках VHDL или Verilog HDL, либо компилятор FPGA Express фирмы Synopsys, либо компилятор LeonardoSpectrum фирмы Mentor Graphics, а для эффективного моделирования ориентироваться на пакет ModelSim фирмы Model Technology (подразделение фирмы Mentor Graphics).

Подобная кооперация усилий различных фирм является вполне обоснованной и ожидаемой, а тенденция обращения к специализированным САПР (на определенных этапах проектирования и в определенных ситуациях) будет, по-видимому, сохраняться и в дальнейшем.

Еще более критичным оказывается вариант выбора САПР при желании обеспечить *сквозное проектирование*, когда в рамках одной САПР должны совмещаться средства различных ветвей или уровней проектирования (например, проектирования конфигурации ПЛИС и проектирования печатной платы, на которой БИС ПЛ будет находиться, или проектирование конфигурации ПЛИС и моделирование ее работы совместно с цифро-аналоговыми элементами).

Типичным представителем САПР для *смешанного или иерархического проектирования* являются САПР фирм Innoveda (бывшая Viewlogic) или OrCAD. Рассмотрим в качестве типового примера состав, назначение и особенности реализации средств, входящих в САПР Workview Office фирмы Viewlogic.

WorkView Office является законченной системой программного обеспечения (для PC-платформ, работающих под Windows 95, 98 или NT), ориентированной на автоматизацию различных уровней проектирования электронных схем. WorkView Office имеет программу-оболочку *навигатор проектов* (в других САПР программа может называться *менеджером проектов*), дающую возможность эффективной организации потока проектирования. Навигатор позволяет управлять как последовательностью выполнения частей пользовательского проекта, так и выбором используемых программ и стандартных библиотек. С информационной точки зрения САПР представляет собой классическую трехступенчатую систему (ввод, обработка и верификация), предназначенную для различных видов обрабатываемой информации на каждой стадии проектирования. Так, вводимая информация может описывать схему электронного блока, а с помощью навигатора проекта далее выбирается желаемый вид обработки: моделирование поведения этой схемы (цифровой, аналоговой или смешанной), компиляция в выбранный тип ПЛИС или разводка печатной платы. В зависимости от выбора на следующем этапе обработки информации проектировщик пользуется соответствующей компиляционной процедурой разводки печатной платы, компиляции в выбранный тип БИС ПЛ или подготовки к моделированию.



И, наконец, на третьем этапе осуществляется верификация готового проекта. Анализ поведения разработанной схемы на временной диаграмме, просмотр и анализ результатов топологической разработки печатной платы или анализ файла отчета о результатах размещения схемы в выбранный кристалл ПЛИС.

Хотя интегрированные САПР безусловно удобны в работе, значительная стоимость их приобретения заставляет многих проектировщиков оставаться на позициях использования САПР, решающих локальную проблему сегодняшнего дня. При необходимости моделирования работы аналоговых или смешанных (аналого-цифровых) систем обычно используются современные версии профессиональной программы PSpice A/D, входящей в состав пакета DesignLab. Достоинством ее является наличие отечественной литературы, описывающей основные приемы работы в ней [22]. Распространение имеют и другие программы (например, Micro-Cap фирмы Spectrum Software [20], APLAC 7.0, Electronics Workbench 5.0 и т. д.). Программный комплекс корпорации MicroSim под названием DesignLab 8.0 используется для сквозного проектирования аналоговых, цифровых и смешанных аналого-цифровых устройств, синтеза устройств программируемой логики и аналоговых фильтров. Введение в комплекс интерфейса с другими САПР ПЛ позволил создать систему, дающую возможность разрабатывать схемы, включающие CPLD и FPGA различных фирм, моделировать на ПК их поведение совместно с другими аналоговыми и цифровыми компонентами, разрабатывать печатные платы и на заключительном этапе повторять моделирование с учетом паразитных эффектов, соответствующих свойствам реальных образцов выходной продукции.

Таким образом, если предполагается совместное проектирование цифровых и аналоговых (в том числе цифроаналоговых и аналого-цифровых) фрагментов или комплексный (сквозной) подход к проектированию, то проектировщик стоит перед выбором САПР фирм MicroSim, Viewlogic или OrCAD [21].

На выбор САПР может оказывать желание разработчика иметь возможность использования результатов проектирования для реализации конечной продукции в других (кроме заданной базовой) *технологических формах*. Различные варианты технологической реализации, как правило, будут требовать и различных проектных процедур, и различных САПР. Так, если после этапа выпуска опытных образцов на базе БИС ПЛ предполагается дальнейшая реализация проекта в виде заказной БИС, то переход от ПЛ к такой форме будет более простым, если при разработке и той и другой формы проектировщик будет ориентироваться на САПР одной и той же фирмы. Такие возможности (с гарантией работоспособности проекта при другой технологии изготовления) предоставляют фирмы Synopsys, Cadence или Mentor Graphics (выбор конкретной фирмы может определяться как типом заказной БИС, так и просто симпатиями проектировщика).

Большое количество фирм специализируется на выпуске полузаказных БИС. Основные варианты различаются. Возможна ситуация, когда исходный про-

ект имел прототип в форме БИС ПЛ и, соответственно, проект был описан средствами САПР выбранного типа ПЛИС, а для реализации проекта в форме полузаказной БИС потребуется использование САПР фирмы-производителя БМК. Если же проект сразу ориентировался на реализацию в форме БМК, то дополнительных преобразований может не потребоваться.

Важнейшей характеристикой САПР БИС ПЛ является эффективность компиляции. Хотя рекомендуется всегда стараться при проектировании занимать под проект не более 90% ресурсов используемой БИС ПЛ (т. е. оставлять минимальные резервы для возможных модификаций), стоимость БИС следующего варианта логической мощности (а иногда и отсутствие БИС с требуемым быстродействием) заставляет разработчика пытаться "уложиться" в ресурсы БИС минимально допустимой логической мощности. Одним из возможных вариантов, способствующих достижению этой цели, может оказаться использование САПР фирм Exemplar Logic и Symplicity, обеспечивающих для проектов, написанных на языках VHDL или Verilog HDL, как правило, самые высокие показатели по эффективности компиляции (минимальность затрачиваемых логических ресурсов и быстродействие проекта).

Существенное влияние выбор САПР оказывает и на эффективность верификации проектов. Этап отладки готового проекта традиционно (как было показано для типовых МП-систем) поддерживался средствами САПР, не является исключением и отладка проекта, загруженного в БИС ПЛ. Современная тенденция заключается во введении в перечень возможностей САПР функций, способствующих упрощению процедуры отладки готового проекта. Например, в САПР Quartus фирмы Altera предусматривается наличие всех трех составляющих такой процедуры: отладочных средств, помещаемых в отлаживаемую БИС/СБИС; информационно-транспортировочных средств, связывающих отлаживаемую БИС и ПК с САПР; программных средств в составе САПР, управляющих и отображающих результаты отладки. Средства так называемого SignalTap Logic Analysis позволяют регистрировать состояния не только на контактах ПЛ, но и во внутренних точках ПЛ в реальном масштабе времени, занося эту информацию в память ПЛ, передавать сохраненную информацию в компьютер с помощью интерфейса JTAG и отображать ее в редакторе временных диаграмм (Waveform Editor) для просмотра, анализа и отладки схемы в БИС.

Еще более сложные средства отладки требуются для БИС, совмещающих в одном кристалле устройства различной природы. Например, реализованных по схеме МП+FPGA, как в приборах семейства E5 типа SOPC ("системы на кристалле") фирмы Triscend. Здесь необходимо отметить возможность обеспечения одинаково эффективной отладки не только МП-ядра (на базе MCS-52) и аппаратного ядра системной логики (типа FPGA), но и их связи между собой за счет введения в архитектуру специального устройства отладки (Hardware Breakpoint Unit). Организация такой смешанной отладки опи-



рается на возможности, предоставляемые как архитектурой кристалла, так и собственно САПР Triscend FastChip.

Не менее сложные средства отладки потребуются для отладки смешанных БИС, например, в анонсируемых фирмой Atmel приборах, реализующих в одном кристалле интерфейс PCI и логику FPGA.

Внешне значительно проще выглядит отладка систем типа SOPC, относящихся к классу generic, (например, схем фирмы Altera), в которых объединяются МП-ядра того или иного типа (начиная от 8-разрядных MCS 8052 и кончая 32-разрядными RISC-архитектурами) и другие части проекта, реализованные в рамках единой технологии программируемой логики. Однако, поскольку реально верификация совместной работы МП, программы и элементов периферийных устройств осуществляется в рамках типового редактора временных диаграмм САПР MAX+PLUS II, формирование даже относительно короткой последовательности команд МП может быть серьезно затруднено сложностью подготовки исходной информации.

Следующим фактором является наличие или возможность использования стандартных решений. Этот момент также может оказаться решающим при выборе САПР. Ситуация несколько улучшается и сглаживается с помощью создания переносимых проектных решений (например, записи функционирования на одном из вариантов языков описания аппаратуры VHDL, Verilog или EDIF). Однако особенности внутренней организации БИС ПЛ могут приводить в таком случае к получению после компиляции не самых эффективных решений (если, конечно, в спецификации проекта эти особенности не учтены).

## 2.4. Основные этапы проектирования БИС программируемой логики

Порядок разработки системы, содержащей БИС ПЛ, укрупненно был приведен в *разд. 2.1* (см. рис. 2.1). Рассмотрим более детально маршрут проектирования, соответствующий разработке конфигурации БИС ПЛ с использованием САПР.

### 2.4.1. Этап 1. Выбор элементной базы и САПР

На первом этапе проектирования на основании анализа технического задания (ТЗ) выявляются специфические требования проекта, позволяющие остановить свой выбор на определенной фирме, выпускающей БИС ПЛ, и на определенном семействе ПЛИС этой фирмы. Отбор осуществляется на основе анализа характеристик как самой БИС — логических, конструктивных, эксплуатационных, стоимостных, так и на анализе свойств требуемой или допустимой САПР. Зачастую выбор предопределяется уже имеющимся практическим заделом и опытом работы проектировщика с продукцией и

САПР определенной фирмы. В этом случае уточняется семейство, архитектурные и эксплуатационные характеристики которого удовлетворяют требованиям ТЗ. Выбор семейства может существенно зависеть от специфических требований проекта, например, необходимости соответствия определенным интерфейсным стандартам, требования наличия скоростной встроенной памяти значительного объема, повышенной радиационной стойкости и т. д. На выбор могут влиять и такие характеристики, как условия поставок, объявления о разработке перспективных модификаций семейства и многие другие соображения. Одним из самых неприятных (и дорогостоящих по совокупности последствий) фактов является выяснение в ходе выполнения проекта невозможности его реализации на продукции выбранной фирмы.

Анализ интерфейсных требований к проекту позволяет конкретизировать количество внешних контактов, необходимых для реализации проекта, т. е. типоразмер корпуса БИС выбранного семейства ПЛИС. Сложность проекта или определенные требования к проекту (например, скоростные характеристики) могут приводить к целесообразности использования на начальных этапах проектирования группы САПР сторонних фирм.

## 2.4.2. Этап 2. Спецификация проекта

Задача этого этапа — переход от технического задания к формализованному описанию проектируемого устройства. Как правило, ТЗ является смесью словесного и технического описания, его формализация приводит к выявлению основных блоков устройства (или алгоритма) и определению их связей и/или взаимодействия. В сущности, именно в этот момент реализуются начальные действия второго этапа. Формально же содержание работ этого этапа — разбиение задачи на отдельные функционально обособленные подзадачи — этап декомпозиции. Способ и средства разбиения чаще всего определяются именно функциональной завершенностью и обособленностью отдельных фрагментов, хотя в значительной степени здесь большую роль играют просто симпатии проектировщика, и лишь иногда разбиение является полностью предопределенным. Сама форма ТЗ может провоцировать проектировщика на использование тех или иных средств, хотя не исключено, что более эффективным мог бы быть другой метод описания проекта или его фрагментов. Декомпозиция может сводиться к составлению схем алгоритмов функционирования фрагментов или к функциональной схеме устройства и его частей. Возможным вариантом для достаточно сложных систем будет разумное совмещение и поведенческого, и структурного разбиения проекта. Разбиение осуществляется не только в рамках одного уровня иерархии. Для большинства проектов производится и разбиение на иерархически организованные уровни.

Существенной задачей, решаемой на этом этапе, является уточнение и согласование с заказчиком интерфейсных функций проекта. Уточняется реализация протоколов внешнего обмена. Именно требуемые временные характеристики и правила взаимодействия с внешними приборами определяют допустимую организацию и структуру внутренних узлов проекта.



Использование САПР на этом этапе проектирования пока еще явление достаточно редкое, хотя для реализации современных очень сложных проектов (несколько сотен тысяч вентилей) все чаще используются специальные блочные редакторы, позволяющие осуществлять декомпозицию проекта без детализации составных частей. Примером может служить САПР Quartus фирмы Altera, включающая в свой состав специальное средство, редактирующее проект на уровне блоков (block-level editing).

## 2.4.3. Этап 3. Разработка общей структуры проекта

Основные задачи этапа — выбор допустимых для реализации каждого уровня иерархии элементов, определение связей между ними, и если параметры элементов являются настраиваемыми, то и их настройка. Ряд моментов является для этапа определяющим: с одной стороны, это источник набора допустимых элементов, а с другой — средства описания соединений элементов между собой, а при необходимости, и возможность описания новых (специфических для этого проекта) элементов.

Как уже указывалось, возможно *как только временное (поведенческое), так и только пространственное (архитектурно-структурное) описание проекта. Однако обычно целесообразно совмещать обе возможности.* При разработке устройств с цифровым представлением информации бывает естественным разбиение их на два блока: операционный и управления. Операционный блок (ОБ) выполняет преобразование данных и строится из стандартных частей (частей с определенным поведением), а блок управления (устройство управления, УУ) обеспечивает необходимую последовательность операций, выполняемых в ОБ (одном или нескольких). Для этого УУ передает на входы ОБ управляющие сигналы. Последовательность действий и, следовательно, управляющих сигналов зависит от результатов операций в ОБ и внешних воздействий. Отсюда видно, что *УУ удобно задавать в форме конечного автомата с памятью* (АП) того или иного типа.

В сложных проектах возможно разделение УУ на несколько функционально слабо связанных пар ОБ-УУ на одном уровне иерархии или создание пары, иерархически погруженной в ОБ (реже в УУ).

Ресурсы и возможности, предоставляемые современными САПР, заставляют несколько по-новому относиться к проектированию автоматов. Здесь необходимо отметить два основных момента.

Во-первых, вопросы оптимального кодирования состояний и условий переходов автоматов (подробно анализируемые многими авторами в различных работах) начинают приобретать для разработчиков скорее теоретический интерес, чем практический. Разработчик чаще всего задает только способ

кодирования автомата, а конкретное кодирование выполняет САПР. Лишь в отдельных случаях (чаще всего, исходя из соображений скорости реализации отдельных определенных переходов или формирования выходного сигнала) целесообразен ручной способ кодировки автомата и навязывание САПР принятого варианта. Возможность подобного жесткого определения кодового соответствия поддерживается или присвоением фиксированных назначений определенным элементам проектируемой структуры в САПР, или использованием более детализированного способа описания структуры проекта.

Во-вторых, организация и структура ячеек ПЛИС (а также практическое отсутствие ограничений на сложность связей автомата) позволяют существенно расширить спектр синтезируемых САПР структурных схем автоматов. Канонические структуры автоматов типа Мили и Мура при синтезе оказываются переплетенными в различных комбинациях *(см. разд. 3.2.8 и 3.4.6)*.

## 2.4.4. Этап 4. Содержательное описание проекта и его частей

Внедрение САПР позволяет создавать эффективные, наглядные, управляемые и контролируемые описания проектов и их частей. Причем одно и то же устройство может быть описано с помощью различных средств САПР. Используемые способы обычно пригодны как для описания проекта в целом, так и для описания его отдельных фрагментов. Методы описания, допустимые к применению исключительно для определенных отдельных фрагментов устройства, относятся к числу редких. Более того, большинство САПР позволяют трансформировать один вид описания в другой.

В настоящее время к наиболее распространенным универсальным способам описания проекта, применимым для любого уровня его иерархии, относят *графический и текстовый*. Реже используются непосредственная разводка схем FPGA в редакторе топологии, описания в виде требуемых временных диаграмм и др. Каждый из способов имеет свои достоинства и недостатки. Близость выразительных средств выбранного способа описания в САПР и внутренней организации или поведения разрабатываемого устройства способна существенно сократить время создания проекта, поскольку может упростить его создание и тестирование, а описание, как правило, окажется более наглядным и понятным.

Графическое представление проекта в современных САПР может создаваться как в базисе графических символов проектировщика, так и в базисе допустимых для выбранной САПР библиотечных элементов, например элементов стандартной серии ТТЛ(Ш). Допустимо смешивание этих двух базисов в различных комбинациях. Главные достоинства графического способа — его традиционность и наглядность, связанные с привычностью разработчиков к восприятию изображений схем. Конечно, эти преимущества проявляются только при правильном иерархическом и структурном разбиении проекта. В большинстве случаев графическое представление не заменяет текстовое представление, а только предваряет его.



Графическую форму задания информации о проекте широко использует, например, фирма Mentor Graphics Corporation в пакете Renoir (в последней редакции эта программа носит название HDL Designer). В зависимости от целевого назначения (и, соответственно, используемых синтаксических конструкций) разработчик должен выбрать тот или иной редактор. Пакет поддерживает ввод графической информации: для структурного описания устройства (Block diagram), для описания комбинационных схем на основе таблиц истинности (Truth table), для потокового описания поведения (Flow chart), для описания в терминах и понятиях автоматов с памятью (в фирменной документации используется термин "диаграммы состояний", State diagram). Результатом работы редакторов является создание описания проекта или его фрагментов на одном из языков описания аппаратуры. Даже если САПР не поддерживает графического ввода, она часто может отображать текстовое описание в графическую форму. Примером может служить самостоятельное обнаружение САПР Synplify (9—24 тыс. долларов) фирмы Synplicity фрагментов, соответствующих по построению описанию автоматов с памятью, с возможностью их дальнейшего графического представления.

К недостаткам графического представления можно отнести отсутствие четких стандартов соответствия текстового и графического представления. В отличие от текстовых, графические способы представления проекта обычно узко специализированы и требуют особых средств для переноса информации о проекте в другую среду, для чего могут быть применены специальные универсальные языки передачи информации о проекте (типа языка EDIF, Electronic Design Interchange Format).

Современные языки описания аппаратуры (HDL, Hardware Description Languages) допускают описание проектируемого устройства как с точки зрения его *поведения*, так и с точки зрения его *структуры*. Эти возможности делают распространенным представление проекта в форме текстового описания алгоритмов функционирования его фрагментов в сочетании с текстовым же описанием межблочных соединений для сложных проектов. Достоинства текстового способа описания проекта заключаются в его компактности и относительной простоте автоматизации любых преобразований, включая начальную генерацию описания проекта. Очень важна возможность использования стандартных универсальных языков типа HDL, обеспечивающая простоту переноса проекта с одной аппаратной платформы на другую и переход от одной САПР к другой.

### Иерархия языков проектирования дискретных устройств

Формализованные текстовые описания достаточно давно применяются при проектировании цифровых устройств. Описания фрагментов систем в форме логических выражений и таблиц использовались от самого зарождения циф-

ровой автоматики. В семидесятые годы было разработано много языков, соответствующих уровню представления регистровых передач (Register Transfer Language, RTL). В этом ряду можно отметить популярный в свое время и ставший прообразом многих современных языков проектирования язык CDL [29], выполненные в Советском Союзе разработки — язык ПРОЕКТ Института кибернетики АН УССР [11], Ф-язык (иначе, "Язык функционального микропрограммирования") Ленинградского института точной механики и оптики [15]. Все эти языки предусматривали определение набора микроопераций, выполняемых в операционном блоке, и описание функционирования в форме микропрограмм. В современных терминах можно говорить об использовании структурно-поведенческого описания проекта. Однако отмеченные языки не были ориентированы ни на машинное моделирование, ни, тем более, на прямую компиляцию в аппаратуру. Возможности автоматизации проектирования в тот период практически ограничивались логическим синтезом и разводкой плат. Языковое представление использовалось, главным образом, для спецификации проектов и сопровождения ручного проектирования. Под сопровождением понималось наличие компактного и наглядного описания для лучшей "обозримости" проектов сложных систем, планирования работ, а также для обеспечения взаимопонимания между разработчиками подсистем.

Один из первых, и до сих пор один из наиболее распространенных языков, VHDL, создавался в начале восьмидесятых так же, как язык спецификации проектов. Однако очень скоро были созданы программы моделирования систем на основании описания в терминах этого языка, а в начале девяностых уже и прямые компиляторы VHDL-программ в аппаратные реализации различных классов.

К настоящему времени сложилась определенная иерархия языков проектирования дискретных устройств. Критерием отнесения языка к определенному уровню является степень абстракции используемых в языке конструкций. В этом смысле можно наблюдать определенное соответствие типов языков проектирования с языками программирования, которое иллюстрируется табл. 2.5.

***Таблица 2.5. Соответствие типов языков проектирования языкам программирования***

| Уровень языка | Языки программирования | Языки проектирования дискретных устройств |
|---|---|---|
| Языки реализации | Программный код | Таблицы соединений |
| Приборно (машинно)-ориентированные языки | Языки ассемблера: | |
| | простой ассемблер | PLDASM, |
| | макроассемблер | ABEL, AHDL... |



***Таблица 2.5** (окончание)*

| Уровень языка | Языки программирования | Языки проектирования дискретных устройств |
|---|---|---|
| Процедурно-ориентированные языки | FORTRAN, Pascal, C | VHDL, Verilog HDL |
| Объектно-ориентированные языки | C++, Prolog, Java | Hardware-C |

*Языки реализации* наиболее близко отражают результат проектирования, будь то исполняемая программа или проектируемое изделие. В языках программирования это программный (машинный) код, представляющий последовательность команд в том виде, в котором она считывается из памяти и интерпретируется процессором. Из языков проектирования этому уровню наиболее близко соответствуют языки описания соединений (таблицы соединений). Элементарные конструкции таких языков определяют набор блоков проектируемого изделия и порядок их соединений в том виде, в котором они будут воспроизведены в устройстве. Тесная связь языков низкого уровня с аппаратными средствами проектируемой БИС заставляет компиляторы создавать проекты со структурой, заданной проектировщиком (ручное проектирование может обеспечить получение более выигрышных параметров). Платой за это обычно является жесткая ориентация на определенную аппаратуру и производящую ее фирму. Общий недостаток языков этого уровня — большая трудоемкость разработки, включая трудности поиска ошибок.

*Простейшие машинно-ориентированные языки* (язык простого ассемблера) сохраняют структуру программного кода, т. е. используют представление "одна машинная команда — один оператор". Однако применяется мнемоническая запись операций, а также символическое представление операндов, что делает текст более наглядным и контролируемым. Подобный принцип заложен в язык PLDASM (аббревиатура расшифровывается как PLD Assembler — ассемблер для ПЛИС). Оператор языка PLDASM задает операцию, выполняемую ячейкой ПЛИС, и имена входных и выходных сигналов ячеек, фактически операндов [28].

В языки уровня *макроассемблера* вводятся конструкции (макрокоманды и макроопределения), которые интерпретируются не одной командой, а некоторыми достаточно стандартными последовательностями команд. Могут вводиться и групповые данные. Одна из важных особенностей языков этого класса — ориентация на определенный процессор и сохраняющаяся близость языкового описания и скомпилированной программы. Этим обеспечивается большее по сравнению с языками высокого уровня влияние разработчика на результирующую программу, что иногда обеспечивает лучшие

показатели критических участков программ, прежде всего, производительность и затраты памяти.

Подобные свойства характерны для развитых *приборно-ориентированных* языков. Обычно это входные языки САПР фирм изготовителей ПЛИС. Такие языки, как правило, содержат опции, ориентированные на эффективное использование архитектурных особенностей определенных типов ПЛИС, а также широкий набор библиотечных модулей, оптимизированных с учетом этих особенностей. Наиболее известные приборно-ориентированные языки (Device Specific Languages) это AHDL фирмы Altera и ABEL, разработанный фирмой Aldec и принятый в САПР фирмы Xilinx. Перенос разработок, выполненных на языках ассемблерного уровня, в другие среды затруднен.

*Процедурно-ориентированные языки* программирования (в дальнейшем назовем их просто процедурными) могут лишь косвенно отражать содержание будущей реализации, используя специфические синтаксические конструкции для описания порядка преобразований. Наибольшее распространение получили языки VHDL и Verilog. Процедурные языки проектирования позволяют описывать устройства через алгоритм их функционирования, в том числе в реальном времени и во взаимодействии с физическим окружением. Эти языки, как и другие алгоритмические языки высокого уровня, в принципе позволяют описать любой алгоритм в последовательной форме, т. е. через последовательность операторов присваивания и принятия решений. Основное их отличие в способности отражать также и параллельно исполняемые в аппаратуре действия, представляемые отдельными параллельно выполняемыми процессами с общим инициализирующим воздействием. Кроме того, процедурные языки проектирования расширяются операторами, позволяющими описывать структуру проектируемого изделия. Впрочем, и структурное описание во многом подобно структуризации в традиционном программировании — включение модуля в структуру можно рассматривать с точки зрения представления в HDL-программе как вызов подпрограммы. На сегодня процедурные языки являются наиболее универсальным аппаратом описания цифровых устройств и покрывают диапазоны представления от уровня элементарных логических ячеек до блочного описания сложных вычислительных устройств. Практические все современные САПР имеют встроенные компиляторы процедурных языков и обеспечивают возможность моделирования поведения и непосредственного преобразования описания в файлы конфигурации ПЛИС.

*Объектно-ориентированные языки* (ООЯ) основаны на абстракциях еще более высокого уровня — классах, объектах. Программа на ООЯ строится как последовательность вызовов объектов и методов классов. Преимущества объектного подхода наиболее проявляются при реализации сложных программных проектов, ибо этот подход позволяет использовать такие важные свойства классов, как наследование и инкапсуляцию.

Развития ООЯ для проектирования только еще началось. Наиболее развитым представляется язык Hardware-C. Он построен как расширение C++



путем введения специфических классов, необходимых для представления аппаратных средств. В частности, введены классы, отражающие параллельные процессы, реальное время, поведение реальных сигналов в цифровых системах.

Такой подход представляется очень перспективным, ибо еще более сближает методологию и средства проектирования программ и аппаратуры. К сожалению, до сих пор не принят стандарт Hardware-C, а фирмы-разработчики ПЛИС не включили в состав своих САПР компиляторы с этого языка. Поэтому авторы сочли включение подробного изложения объектно-ориентированных языков проектирования в настоящую книгу преждевременным.

Итак, при содержательном описании проекта и его частей необходимо, чтобы проектировщик выбрал средства и методы описания трех составляющих этого проекта: описания структуры операционного блока, описания поведения элементов и описания функционирования устройства управления.

## Описание структуры операционного блока

Архитектурно-структурное описание операционного блока базируется на задании структуры соединений отдельных элементов. Традиционный графический способ представления структуры соединений, оставаясь наиболее наглядным способом, в современных САПР сопровождается текстовым способом описания. Входящие в состав САПР программные утилиты обычно обеспечивают автоматическое прямое и обратное преобразование описаний.

Элементный состав операционного блока зависит от состава используемой библиотеки. Большинство САПР поддерживает иерархическое описание проектов. На любом уровне проект представляется как совокупность элементов этого уровня. Набор функциональных возможностей библиотечных элементов, предлагаемых стандартными САПР, чрезвычайно широк, а по составу и происхождению библиотеки разделяют на следующие:

- стандартные библиотеки фирмы-разработчика САПР, содержимое которой соответствует той или иной распространенной серии схем МИС и СИС (например, ИС типа 74 серии);
- стандартные элементы вычислительной техники (n-входовые логические элементы, дешифраторы, мультиплексоры, счетчики и т. д.). Параметры элементов при этом фиксированы;
- типовые элементы вычислительной техники (счетчики, регистры, мультиплексоры и т. д.), ряд конкретных параметров которых (разрядность, полярность управляющих сигналов и т. д.) могут назначаться проектировщиком произвольно;
- типовые узлы вычислительных систем (периферийные устройства, аппаратные ядра микроконтроллеров и микропроцессоров), часть параметров которых варьируется проектировщиком (как правило, разработка кон-

фигурации этих узлов выполняется либо фирмой-разработчиком узла, либо в содружестве с ней);

- узлы вычислительных систем (периферийные устройства, аппаратные ядра микроконтроллеров и микропроцессоров), часть параметров которых варьируется проектировщиком. Узлы обычно разрабатываются фирмами, специализирующимися на выпуске этого вида узлов (результаты разработки носят название IP, Intellectual Property, а содержимое поставляемого описания носит название ядра интеллектуальной собственности — core IP);
- элементы, созданные проектировщиком и объединенные в его библиотеку проектировщика.

Любой проект может быть использован в качестве подпроекта в более сложном проекте. Единую библиотеку модулей проектировщика можно и не создавать (хотя желательно иметь некий путеводитель по собственным проектным модулям, а в лучшем случае, базу данных о выполненных проектах).

Как правило, на любом уровне иерархии базовый набор элементов операционного блока (этого уровня) дополняется требуемым для их функционирования набором регистров, логических схем (обычно многофункциональных и управляемых), буферных схем и коммутируемых связей между ними. Важно, чтобы на более низких иерархических уровнях описания проекта была однозначная трактовка функционирования всех элементов ОБ.

Несмотря на оптимизацию схем, соответствующих наиболее распространенным сериям ИС, самым неправильным (с точки зрения авторов) представляется практикуемый некоторыми проектировщиками прямой перенос схемотехнических решений из реализации в форме СИС и МИС в реализацию на БИС ПЛ. Получаемые при этом результаты, как правило, не только не оптимальны, но зачастую не обеспечивают желаемого функционирования проекта.

## Описание поведения элементов операционного блока

В тех случаях, когда поведение того или иного элемента операционного блока не соответствует поведению имеющихся в распоряжении проектировщика стандартных элементов, разработчик вынужден создавать свой элемент. Так же, как и при описании структуры устройства, описание поведения отдельных элементов может осуществляться с привлечением как *графических, так и текстовых средств*. Сравнительные характеристики двух способов описания достаточно подробно обсуждались выше. Как правило, для компиляторов входным описанием является текстовое описание на одном из языков описания аппаратуры.

Вопросы разработки описаний поведения различных функциональных фрагментов цифровых и логических устройств подробно рассматриваются в *гл. 3*.



## Описание работы устройства управления (УУ)

На этом этапе определяется функционирование УУ, обеспечивающее требуемое взаимодействие элементов ОБ. Следует подчеркнуть, что все составляющие рассматриваемого этапа проектирования сильно взаимосвязаны, и если не разрабатываются параллельно, то чаще всего требуют итерационного выполнения.

Формы и средства описания автоматов разнообразны. Вопросы и примеры описания поведения автоматов в текстовой форме на различных языках представлены в последующих главах.

Для повышения наглядности поведения автоматов большинство современных САПР поддерживает задание этого поведения в *графической форме*. Описание в виде схемы переходов (диаграммы состояний) становится одним из самых распространенных вариантов задания автоматов (в английской терминологии State Machins). Графические редакторы для создания автоматов включаются в состав средств задания исходных проектов современных САПР (например, в САПР Foundation фирмы Xilinx разработки фирмы Aldec). Фирма Mentor Graphics для создания текстов на языках VHDL или Verilog предлагает использовать специальный набор графических редакторов под названием Renoir (с 2001 года HDL Designer Series). Редактор позволяет создавать в графической форме описания не только автоматов, но и других форм задания поведения проектов.

Редакторы разных фирм-производителей СБИС ПЛИС имеют особенности, но для всех них характерны исключительная простота, естественность и дружественность интерфейса с пользователем, а также отсутствие жесткой необходимости знания выходного языка редактора. Наиболее совершенные версии программ типа StateCAD пакета Workview Office фирмы Viewlogic обладают полным набором средств для выполнения всей проектной процедуры разработки управляющих автоматов (УА), позволяющих реализовать следующие операции:

- рисовать граф переходов, включая наименование состояний, направления, условия и приоритеты условий переходов, формируемые сигналы и способы их образования;
- проверять корректность составленного графа переходов (повторение имен, неоднозначность перехода, некорректность перехода и т. д.);
- компилировать проект (формировать выходной текстовый файл) в выбранном языковом базисе;
- моделировать поведение автомата в интерактивном или компиляционном режиме.

Важное достоинство программы типа StateCAD — возможность широкого выбора форм представления результата (описания на языках высокого уровня VHDL, Verilog и C, а также на языках низкого уровня ABEL, AHDL).

Заметим, что *специфика продукции той или иной фирмы сказывается и на языках высокого уровня*, выражаясь, прежде всего, в отличиях в библиотеках, требуемых для работы, и в сложности и вариантности допустимых синтаксических конструкций для компиляторов. Конечные результаты компиляции одной и той же исходной схемы автомата или последующей компиляции одной и той же программы с языка высокого уровня в загрузочный файл микросхемы ПЛИС, полученные от компиляторов разных фирм, могут существенно различаться и иметь различную эффективность. Программа StateCAD пакета Workview Office удобна тем, что перед трансляцией графа переходов можно задать не только желательное языковое представление (VHDL, AHDL, Verilog, ABEL и т. д.), но и фирменные атрибуты, что позволяет оптимизировать запись автомата и избежать применения синтаксических конструкций, недопустимых для компиляторов соответствующих фирм.

Как уже отмечалось, при использовании графических редакторов от пользователя не требуется обязательное владение выходным языком редактора. Однако в определенных случаях такое владение исключительно полезно. Полезность ориентации в языковых конструкциях проявляется, например, в ситуациях, когда автомат должен быть минимизирован по тем или иным параметрам, прежде всего, по временным интервалам между формируемыми выходными сигналами, что может приводить к временным состязаниям сигналов. Именно в этих случаях владение языком и искусство проектировщика облегчают получение наилучших результатов.

## 2.4.5. Этап 5. Компиляция проекта

После составления проекта и его наиболее существенных частей можно приступать к самому ответственному этапу проектирования — компиляции проекта. Компиляции может подвергаться как весь проект, так и его основные части. Компиляция отдельных фрагментов, с одной стороны, упрощает проектирование, поскольку уменьшает размерность анализируемой проблемы, но, с другой стороны, удлиняет процедуру проектирования, а самое главное, требует учета различия функционирования внутренних ресурсов и внешних (согласующих) элементов.

При компиляции проекта целиком удается обнаружить большинство скрытых ошибок и нестыковок, которые проявляются при попытке объединить отдельные фрагменты. Реально процесс компиляция состоит из ряда последовательно выполняемых действий: сборки базы данных проекта, контроля соединений, логической минимизации проекта, монтирования проекта в заданную или выбранную схему, формирования загрузочного (конфигурационного) файла и т. д. На любом этапе этих работ могут возникать ошибки, требующие повторной компиляции после их коррекции. Процесс ком-



пиляции в САПР фирм-производителей БИС ПЛИС отличается от компиляции в САПР сторонних фирм. Как уже отмечалось, основное отличие состоит в отсутствии у последних стадии формирования загрузочного файла.

Результатом работы компиляторов сторонних фирм, как правило, является структура проекта в базисе выбранного семейства ПЛИС без конкретной привязки к ресурсам ПЛИС. Чаще всего это носит название представления *на уровне регистровых передач*. Программные пакеты сторонних фирм не только создают такое представление, но дают возможности его просмотра и автоматизированного анализа.

Классическим примером такого компилятора является пакет LeonardoSpectrum фирмы Exemplar Logic, Inc. (подразделение Mentor Graphics). Пакет может поставляться в различных вариантах: стартовая цена первого уровня тыс. долларов, второго — 8950 долларов, третьего — 17,5 тыс. долларов. LeonardoSpectrum является набором высокоуровневых средств, предназначенных для синтеза проектов, ориентированных на реализацию в форме одиночной БИС CPLD, FPGA или в форме ASIC-проектов. Поддерживает проектирование для ПЛИС фирм Actel, Altera, Agire Systems, QuickLogic, Xilinx и др. Обеспечивает ввод проектов на языках VHDL и Verilog, отладку на уровне регистровых передач, оптимизацию, базирующуюся на задаваемых проектировщиком ограничениях, анализ временных характеристик проекта, формирование выходной информации, необходимой для размещения и разводки проекта в выбранную БИС, просмотр проекта на вентильном уровне.

Результат компиляции при использовании САПР фирм-производителей БИС ПЛ — загрузочный файл, т. е. конфигурационная информация для выбранной микросхемы ПЛИС. Кроме этого, обычно создается и файл отчета, содержащий всю информацию как о процессе компиляции, так и о его результатах. Имеется существенное различие для компиляционных процедур схем типа CPLD и FPGA. Для FPGA помимо автоматического размещения и трассировки соединений, как правило, допустимо и ручное вмешательство проектировщика в процесс на любых его этапах. Этой цели служат *редакторы топологии* (Floorplanner), позволяющие изменять структуру проекта на кристалле и повышать производительность проектируемых устройств. А для схем типа CPLD влияние проектировщика на структуру (а следовательно, и характеристики) скомпилированной схемы, чаще всего, возможно только путем косвенного воздействия, за счет либо изменения формы описания проекта, либо изменения устанавливаемых перед компиляцией опций. Ручная трассировка выбранных цепей возможна, например, в топологическом редакторе FPGA Editor, входящим в состав САПР Foundation фирмы Xilinx. Только после успешной синтаксической компиляции проекта или его частей можно переходить к их верификации.

## 2.4.6. Этап 6. Верификация проекта

Верификация разработанного устройства, а в мало-мальски сложных проектах и отдельных его фрагментов — один из важнейших этапов проектирования, поскольку практически не бывает бездефектных проектов, созданных с чистого листа. Обнаружение дефектов проекта — сложнейшая задача. Скорость и тщательность верификации во многом зависят от искусства разработчика.

Важность этапа верификации приводит к тому, что работы по тестированию стремятся упростить, включая в состав САПР различные программы, автоматизирующие работы этого этапа. В современных САПР наибольшее распространение получила верификация, базирующаяся на моделировании работы ПЛИС при различных внешних воздействиях. Для упрощения создания желаемой последовательности входных или контролируемых (выходных) сигналов в состав САПР вводят *редакторы временных диаграмм*. Редакторы делятся на компилирующие и интерпретирующие. Редакторы интерпретирующего типа позволяют упростить процедуру отладки проектов и обнаружить их дефекты, связанные с неправильной трактовкой разработчиком структурной или поведенческой реализации системы либо особенностей реализации используемой элементной базы.

В многооконных САПР интерпретирующего типа легко демонстрируются результаты моделирования для текущего момента модельного времени во всех видах отображения проекта (сигналы в электрических схемах, в топологии), в них также легко изменить ход эксперимента и состав представляемых сигналов. В случаях схемотехнического описания проекта упрощена трассировка сигналов.

Достоинством компилирующих систем моделирования является минимизация временных затрат. Проблематичными представляются отладка программно реализованных проектов и установление соответствия между строками текста и состояниями отдельных сигналов.

Распространение получили два подхода к генерации внешних относительно проекта воздействий. Один подход заключается в формировании этих воздействий путем задания временной последовательности входных сигналов в редакторе временных диаграмм (графический способ задания для компилирующих систем моделирования или формульный для интерпретирующих). Другой подход (особенно удобный при языковом задании проекта и обычно используемый на этапе его функциональной верификации) состоит в написании специальной тестирующей программы. Программы для тестирования в этом случае строятся на основе архитектурно-поведенческого тела, в котором проектируемый модуль представлен как структурный компонент, а генератор воздействия — в поведенческой форме (подробнее о создании программ Test-Bench *см. разд. 3.1*).

В большинстве реальных цифровых устройств после подачи на них некоторых начальных данных выполняются несколько повторяющихся циклов.



Поэтому целесообразна проверка работы устройства на ряде наборов однотипных данных. Можно рекомендовать организацию следующей последовательности работы программного модуля (процесса), представляющего тестовое воздействие: генерация сигналов начальной установки, затем реализация двух вложенных циклов, причем внутренний цикл последовательно формирует тестирующие сигналы для выполнения действий на одном наборе входных данных, а внешний обеспечивает их изменение.

## 2.4.7. Этап 7. Определение временных характеристик разработанного устройства

Наличие после компиляции проекта в САПР полной модели структуры проектируемого устройства и знание временных параметров всех компонентов этой структуры позволяет автоматизировать процесс вычисления различных временных характеристик проекта.

Например, в САПР MAX+PLUS II (фирма Altera) предусмотрено автоматическое вычисление трех основных классов временных параметров:

- минимальных и максимальных задержек между источниками (входными сигналами) и приемниками (выходными сигналами), информация о которых выдается в виде матрицы задержек;
- максимально допустимой частоты тактирования элементов памяти, используемых в проекте;
- времен предустановки и удержания сигналов, гарантирующих надежное срабатывание схем при фиксации сигналов в синхронных элементах памяти.

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

Хотя выполнение перечисленных вычислений не гарантирует обнаружения всех ошибок проектировщика, связанных с временными процессами в проекте, оно существенно уменьшает число таких ошибок или, как минимум, позволяет обнаружить в проекте места, опасные с точки зрения сбоев, а также выяснить причины непредусмотренного поведения проектируемой системы.

## 2.4.8. Этап 8. Организация натурных экспериментов

Одним из заключительных этапов проектирования является экспериментальная проверка спроектированного устройства. При всей тщательности выполнения предыдущих этапов всегда существует далеко не нулевая вероятность того, что в проекте имеются дефекты, которые могут проявиться

при внедрении или даже штатном использовании устройства и вызвать нежелательные последствия.

Выполнение натурных экспериментов существенно увеличивает вероятность выпуска бездефектной продукции. Средства ускорения работ на этом этапе и возможности его переноса на ранние этапы разработки, т. е. до того момента, когда будет закончено изготовление конечного продукта, известны — это прототипные системы и средства проведения экспериментов с ними. Прототипные платы широко использовались и ранее, в частности, при создании микропроцессорных систем. Аналогична и ситуация при разработке систем и устройств на основе средств программируемой логики. Прототипная плата содержит одну или несколько микросхем программируемой логики и дополнительную аппаратуру, связанную с ее целевым назначением, например, средства управления и отображения, микросхемы быстродействующих ОЗУ. Широкий спектр прототипных плат выпускается и поставляется различными отечественными и зарубежными фирмами. Здесь можно указать средства фирм Altera (Demo Board), PLD Applications (платы PCI Bus Evaluation Board), Xilinx, Virtual Computer Corp., Video Software (платы HOT PCI Design Kit) и др.

Как при использовании прототипирования, так и при тестировании конечного изделия необходимо привлечение аппаратных и программных средств для загрузки конфигурации, генерации воздействий и контроля правильности поведения исследуемого устройства. Важный момент проведения экспериментов — генерация тестирующих воздействий. В этом плане БИС ПЛ обеспечивают новые, не достижимые ранее возможности. Дополнительная микросхема, а в некоторых случаях и определенная часть ресурсов отлаживаемой БИС, могут использоваться как программируемые генераторы сигналов. При этом содержание тестов легко модифицируется в процессе проведения экспериментов в зависимости от результатов предыдущих шагов экспериментальных работ. Полезным инструментом отладки могут стать средства передачи данных о состоянии тестируемого объекта в процессе выполнения эксперимента в инструментальный компьютер с целью визуализации и детального анализа (некоторые возможности представлены в *разд. 2.6*). Многие ПЛИС и соответствующие САПР поддерживают такое взаимодействие. Но во многих случаях может потребоваться разработка специального программного обеспечения для ускорения анализа поведения объекта тестирования. Естественно, в определенных ситуациях допускается пользоваться серийной аппаратурой типа многолучевых осциллографов, логических анализаторов и т. п.

### 2.4.9. Этап 9. Подготовка к производственному выпуску

После успешного завершения натурных экспериментов с прототипным или макетным образцом проектировщик должен обеспечить выпуск опытной партии разработанного изделия. Важнейшей задачей при этом является



обеспечение качественного сопровождения продукции, выпускаемой в форме ИСПС.

Поскольку принципиальная работоспособность проекта к рассматриваемому моменту проектной процедуры уже не подвергается сомнениям (как правило, почти не подвергается), то задача тестового оборудования, применяемого на этом этапе, отличается от задач оборудования, использованного на предыдущем этапе. Тестовое оборудование и методика его применения должны отсеивать неисправную продукцию за минимальное время и с минимальными затратами и выполняться персоналом при минимальных профессиональных требованиях. И лишь для небольшого объема продукции целесообразно использовать тестовые средства, позволяющие локализовать места неисправностей.

Необходимо разделять требования к тестовому оборудованию и требования к самому проекту. Эффективность работы тестового оборудования в значительной мере зависит от предусмотрительности проектировщика. Процедура разработки с первых шагов должна ориентироваться на необходимость тестирования конечной продукции, и только если проект исходно обладает определенными ресурсами (более подробно рассматриваемыми в *разд. 2.6*), процедура тестирования может быть существенно упрощена и ускорена.

Если проектировщиком (или заказчиком проекта) была правильно выбрана цель проектирования, то после выпуска опытных образцов изделия может возникнуть задача организации выпуска серийной продукции. Основные усилия разработчиков при этом направлены на проектирование и изготовление аппаратных и программных средств, позволяющих удешевить и ускорить выпуск серийной продукции. При условии появления признаков повышенного спроса на выпускаемую продукцию и реальных рынков сбыта может рассматриваться вопрос о модернизации продукции и, в том числе, о переводе на более дешевую элементную базу СпИС. Это направление связано с разработкой стратегии и тактики конвертации проектов в реализацию на новую элементную базу или перехода к новым технологиям изготовления конечной продукции.

## 2.5. Проблемы и методы проектирования SOPC

Если еще совсем недавно схемы БИС и СБИС уже вмещали в себя большинство ИС, находящихся на одной печатной плате, то теперь один кристалл SOPC вмещает в себя несколько СБИС (таких как МП, память, блоки ввода/вывода). Задачи, встающие перед проектировщиком, собирающимся использовать SOPC, внешне не отличаются от задач, уже рассмотренных для МПС и ПЛИС. Основное отличие заключается в том, что в одном кристалле должна быть спроектирована и аппаратура, и программное обеспечение.

Однако известно базовое положение теории систем (впервые опубликованное в работах L. Von Bertalanffy) о несовпадении свойств целого и суммы его частей вполне применимо для SOPC и поэтому требует пересмотра процедуры проектирования SOPC практически по всем направлениям. С появлением кристаллов SOPC требуются и уже начали происходить определенные изменения практически на всех маршрутах проектных процедур.

Современное требование уменьшения времени, необходимого для выпуска конечного продукта, заставляет искать пути сокращения как времени выполнения любых отдельных этапов проектирования, так и устранения причин, вызывающих необходимость итерационных возвратов к предыдущим этапам проектирования. Традиционная процедура проектирования, приведенная на рис. 2.26, характеризовалась существенным запаздыванием разработки программ относительно разработки аппаратуры.

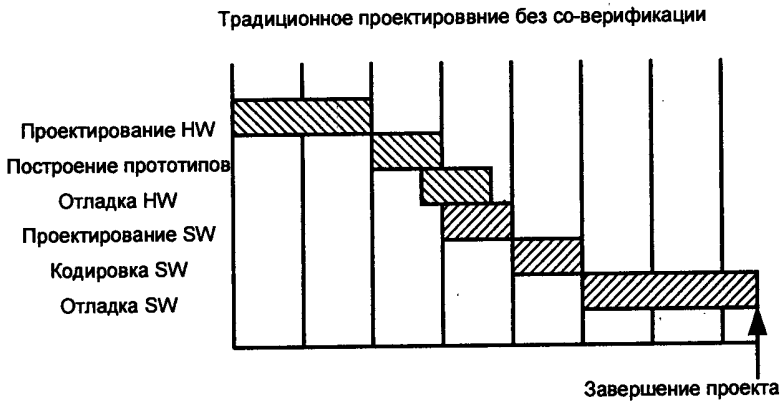


**Рис. 2.26.** Традиционная процедура проектирования

Трудности проектирования SOPC связаны как с необходимостью выполнять проекты в сжатые сроки, так и с постоянно возрастающей сложностью систем. По существующим оценкам, средний объем современных проектов превышает 50 тыс. вентилей и имеет постоянную тенденцию удваиваться каждые два года, и в окончательной стоимости проекта все большую роль играют разработанные программные и "зашитые" в ПЗУ аппаратные ресурсы. Ожидается, что в ближайшие несколько лет проектами, содержащими более 100 тыс. вентилей и МП-ядро (core-based), будут заниматься более 200 тыс. разработчиков. Возрастание сложности современных проектов как в части объемов HW (в количестве логических вентилей), так и в части объемов SW (в строках программного текста), делает проблему поддержки данного направления актуальной.



## 2.5.1. Специфические технико-технологические особенности реализации систем типа SOPC

Как уже отмечалось выше, совмещение в одном кристалле программного и аппаратного исполнения алгоритмов приводит не к арифметическому сложению их потенциальных возможностей. Если на первом этапе появления кристаллов типа SOPC основное внимание уделялось мобильности периферийной части МПС, и именно это выдвигалось в качестве основного достоинства SOPC, то далее стало понятно, что меняется отношение и к другим составляющим МПС. Основным фрагментом, определяющим интеллектуальность, становится не процессор, а содержимое памяти, не периферия как таковая, а память ее конфигурации. Процессор, а вернее его конкретная организация, перестает быть доминирующим фактором, определяющим производительность системы. Поведение или структуру процессора в современных системах можно достаточно легко изменить или просто поменять процессор целиком (см. Agere Systems, **http://www.agere.com**). Перетрансляция, а вернее эмуляция нового типа процессора (если процессор это Softcore) может позволить без значительных временных потерь работать с программой, исходно разработанной для другого процессора. При этом свойство переносимости написанных на языках C и C++ программ — фактор, существенно облегчающий подобные модификации.

Поскольку целью любого подхода к проектированию являются попытки упрощения составления, а при необходимости и корректировки структуры системы, то естественным будет желание построения системы по модульному принципу. Тогда построение системы при помощи САПР сведется к извлечению необходимых строительных кирпичиков из библиотеки и взаимной подгонке их друг к другу для получения выбранной архитектуры. Приблизительно так компоновались печатные платы из набора дискретных компонентов ИС. Элементы плат объединялись в модульную структуру с шинными связями.

Естественно, что этот же прием разработчики электронной аппаратуры попытались сохранить и при переходе от размещения элементов в форме корпусов ИС на печатной подложке к размещению и трассировке виртуальных компонентов в плоскости кристалла. (Термин "виртуальность" будет использоваться для того, чтобы подчеркнуть отличие физического представления компонента в кристалле от его логического обозначения или описания в проектной документации). Объединение блоков, в том числе и виртуальных компонентов, независимо от их типа (HW-core, SW-core, firm-core) осуществляется с помощью шин. Однако подобный прямой перенос игнорирует особенности и выгоды размещения элементов на одном кристалле.

При печатном монтаже ключевой стратегией являлась минимизация числа шинных сигналов, поскольку рост числа контактов и сигналов непосредственно отражался в размерах корпусов ИС, размерах платы и, тем самым,

в стоимости печатной платы. Такая тактика приводила к увеличению размеров корпуса и уменьшению плотности монтажа. Шины общесистемного уровня должны были, кроме того, поддерживать межплатные соединения на кросс-платах. При этом размеры разъемов и окончательная цена изделия оказывались прямо связанными с числом сигналов. Именно по этому традиционными для системных шин были решения, базирующиеся на введении высокоимпедансных Z-состояний для сигналов, а если была возможность, то прибегали к мультиплексированию (например, шин адреса и данных).

Многие проектировщики перенесли приемы, свойственные печатным платам, на уровень размещения в кристалле и не учли некоторых особенностей размещения компонентов в кристалле. Проводник на кристалле не расходует кремниевую зону и поэтому оказывает незначительное влияние на размеры кристалла БИС, цену корпуса и т. д. Современные промышленные технологии изготовления кристаллов предполагают наличие большого числа слоев металлических соединений (свыше 5) и следует ожидать, что добавление линий или обеспечение переходов между слоями не будут вызывать значительного удорожания продукции. Более того, большинство фирм-производителей ИС с большой осторожностью подходят к формированию внутренних элементов с тремя состояниями. Очень редко в состав ПЛИС вводятся такие элементы, и, чаще всего, синтезирующие средства заменяют их на мультиплексоры с соответствующим количеством входов. Отказываться от применения элементов с тремя состояниями разработчиков структур ПЛИС заставляют возможности и ограничения синтезирующих средств. Ограничения САПР связаны со сложностью (большим объемом вычислений) определения состояний узлов, объединяющих сигналы от различных источников на одной шине. Большие сложности вызывает выполнение точного статического временного анализа для различных комбинаций разрешающих и запрещающих сигналов.

Модульность создает предпосылки и для перемещения решений из области аппаратного подхода в программные решения или наоборот.

## 2.5.2. Идеи и методы сопряженного проектирования

Рассмотренные выше проблемы и современные требования к проектированию систем и, прежде всего, типа SOPC привели к появлению таких новых подходов к проектированию, как сопряженное проектирование, сопряженная верификация и сопряженное моделирование.

Со-проектирование (сопряженное проектирование) — процесс параллельного проектирования аппаратных и программных средств, при котором осуществляется оценка целесообразности выбора аппаратной или программной реализации определенного фрагмента проекта. Этот процесс также дает проектировщикам возможность, увидеть (на абстрактном уровне), как система могла бы работать с данным разделением аппаратных средств ЭВМ и программного обеспечения.



Со-верификация (сопряженная верификация) — анализ совместной работы программного обеспечения и аппаратных средств ЭВМ с целью определения, будут ли они функционировать правильно вместе. С другой стороны, во время этого процесса анализируется, будут ли одинаково решаться специфические задачи проекта при аппаратной или программной реализации. Результатом работы является фиксация выбранного варианта реализации.

Со-симуляция (сопряженное моделирование) — совместное моделирование работы аппаратных и программных средств, при котором аппаратные средства ЭВМ описаны, например, в форме программных моделей (обычно это VHDL- или Verilog-модели).

Возможны две трактовки понятия сопряженного проектирования: либо это весь процесс проектирования, а со-верификация и со-симуляция его отдельные подэтапы, либо это начальный этап проектирования, результатом которого является декомпозиция проблемы, а со-верификация и со-симуляция это последующие и самостоятельные этапы проектирования. Разные трактовки приводят к тому, что иногда отдельные решения в данной проблемной области ряд авторов выдает за решение общей проблемы. Реально совместные процедуры выполняются в различных взаимных комбинациях на различных этапах проектирования.

В целом задача современного проектирования SOPC — выполнять каждый этап быстрее, с более надежными и достоверными (не требующими пересмотра и итерационных возвратов) результатами. Вопрос контроля проекта становится одним из узловых. Если удается хорошо протестировать каждый этап проекта и получить оценки его эффективности, то резко уменьшается число итерационных возвратов в проектной процедуре. Основные этапы процедуры сопряженного проектирования, базирующиеся на идеях со-верификации, приведены на рис. 2.27.

Проблемы и методы их решения, опирающиеся на идеи сопряженного проектирования, при совпадении их общих подходов отличаются и оказываются различными для разных этапов проектирования.

Узловой проблемой проектирования микропроцессорных систем всегда являлась проблема принятия решения о разделении функций между программной и аппаратной частями системы. От успешности отнесения решаемого фрагмента системной задачи к той или иной реализации зачастую зависит успех или неуспех всего проекта.

На начальном этапе проектирования требуется правильно (эффективно, по конечной оценке) разделить решаемые задачи между аппаратной и программной реализациями. Перемещение фрагмента от одной реализации

к другой на заключительных стадиях проектирования может вообще оказаться невозможным, а чаще приводит к существенным сдвигам окончательной готовности проекта из-за необходимости возврата к предшествующим этапам проектирования. Основной подход для начального этапа проектирования — поиск способа спецификации проекта, подходящего для описания и аппаратной, и программной части. В конце этого этапа проектировщика интересуют вопросы организации со-верификационной процедуры, которая позволила бы оценить правильность разбиения системы.

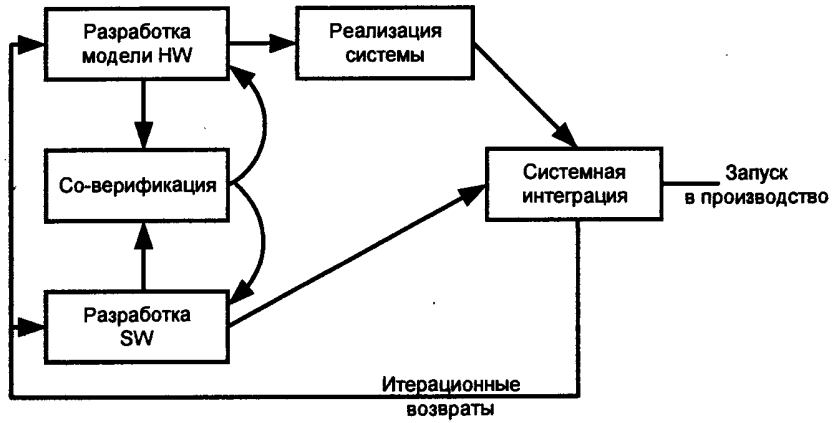


**Рис. 2.27.** Варианты верификации аппаратных и программных решений

На этапе проектирования отдельных составляющих проектируемой системы сопряженное проектирование обозначает параллельное проектирование HW- и SW-средств с взаимными и координированными изменениями тех шагов разработки, где предусмотрено их взаимодействие. Необходимость перераспределения функций или действий между HW и SW может оказаться целесообразной на любом шаге этого этапа проектной процедуры, такое перераспределение может быть исключительно полезным при составлении частей проекта и его моделей. На заключительных этапах разработки HW и SW необходимо убедиться в том, что получены правильно функционирующие части. Основной метод здесь со-верификация. Если совместное HW/SW-моделирование выполняется достаточно быстро, этот процесс может использоваться в качестве основы со-проектирования и со-верификации.

Задачей комплексной отладки HW и SW является быстрое устранение ошибок проектирования. На этом этапе наиболее сложной становится стыковка двух разных подходов. Здесь у проектировщиков, представляющих противоположные стороны (HW и SW), появляются противоречия в трактовке причин и источников отклонений характеристик системы от запланированных или ожидаемых. Локализация причины отклонения и способы ее устранения могут требовать очередного перераспределения функций между HW и SW. Удобнее, чтобы стыковка произошла как можно раньше, и отладка выполнялась средствами, одинаково эффективными и для аппаратуры, и для программного обеспечения. Не менее важно, чтобы эти средства были одинаково удобны для разработчиков HW и SW.



Вопросы разбиения или перераспределения задач между HW и SW могут возникать даже на этапе встраивания проекта в ИС или на этапах отладки готового изделия. Однако чем позже, тем сложнее и дороже дается такое перераспределение.

Идея сопряженного проектирования в SOPC приводит к двум основным направлениям решения этой глобальной задачи:

- Откладывание вопроса о выделении конкретного исполнителя (для программной или аппаратной реализации) определенного поведенческого фрагмента системы на как можно более удаленный момент при условии, что описание поведения фрагмента существует и может использоваться для моделирования, а возможно даже для синтеза фрагмента. Программная реализация может быть достаточно компактной. После реализации основных функций ядра МП-системы добавление еще одной программы, как правило, будет очень слабо влиять на объем памяти, уже заложенной в систему, и не будет требовать замены типа используемой БИС памяти. Даже существенная модернизация функционирования системы (при правильно заложенных ресурсах модернизации) не требует изменений конструкции печатной платы при переходе к следующему более емкому корпусу ИС памяти. Основным результатом безоглядного наращивания числа параллельно исполняемых задач на одном и том же процессоре может оказаться ситуация, когда программная реализация не сможет вовремя реагировать на внешние события. Аппаратная реализация, в отличие от программной, хотя и требует значительных ресурсов, но зато не накладывает столь жестких ограничений на поведение одновременно работающих фрагментов.
- Возможность не принимать окончательное решение о разделении функций до как можно более позднего момента. Это позволяет отодвигать решение вопроса о целесообразной физической реализации фрагмента на более поздний срок. Такую возможность дает реализация нетиповой части проекта в форме БИС программируемой логики. При этом модификация структурной организации этой части системы может не требовать конструктивного пересмотра. Перераспределение функций между аппаратной и программной частями системы может быть выполнено в любой момент времени жизни спроектированной системы. Предельным вариантом является реализация в форме кристалла типа SOPC, когда конструктивная реализация спрятана внутри одного кристалла.

"Системы на кристалле" оказываются прекрасной технической основой сопряженного проектирования — поскольку системы оказываются в значи-

тельных пределах конструктивно инвариантными к аппаратным и программным решениям. Переход от одного способа реализации к другому не требует существенной конструктивной переработки. Технологической основой сопряженного проектирования являются САПР, интегрирующие в себе средства проектирования и отладки аппаратно-программных систем. Кроме того, работа как самих САПР, так и проектировщиков существенно упрощается, если возможно использование инвариантных способов описания аппаратной и программной реализации.

До последнего времени реальное использование методов сопряженного проектирования, а тем более соответствующих технических средств, сдерживалось целым рядом факторов и оказывалось по силам лишь небольшим группам проектировщиков, связанных с разработкой новых проектов. Из приведенных определений ясна общая постановка проблемы, проектная процедура в таком случае может иметь вид, приведенный на рис. 2.28, однако вопрос состоит в том, как технически организовать эту действительно совмещенную процедуру.

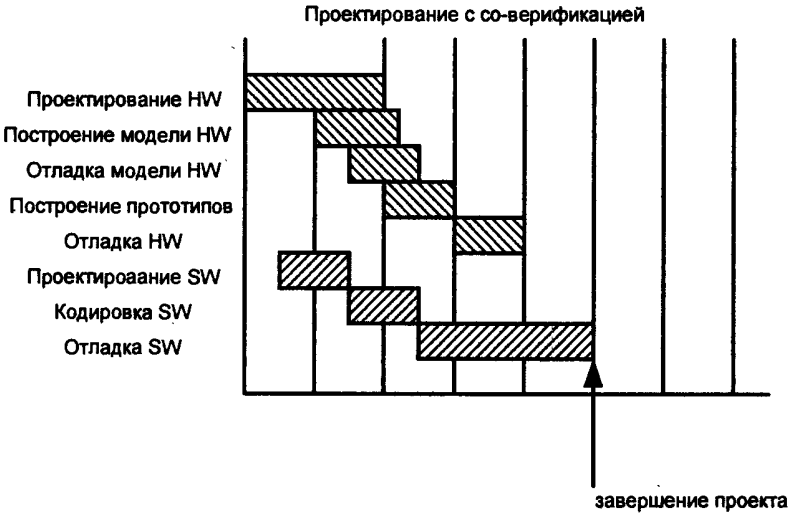


**Рис. 2.28.** Процедура сопряженного проектирования

Целый ряд причин тормозит решение в такой общей постановке. Отсутствует техническая база. В проектах еще достаточно редко используются современные кристаллы SOPC. Проектировщики вынуждены опираться на старые САПР, новые САПР оказываются очень дорогими. Нет единых проектных средств самого первого уровня — системного — инвариантных к реализации. Новые технологии проектирования требуют существенных изменений и в мышлении самих проектировщиков.



Проектированием сложных аппаратно-программных систем традиционно занимаются специалисты трех профилей: системные инженеры, инженеры-программисты и инженеры-схемотехники. Методы и средства автоматизации, используемые этими разработчиками, существенно отличаются — программисты в своей работе больше внимания уделяли вопросам эффективного описания программного обеспечения с точки зрения его поведения, в то время как схемотехники традиционно тяготели к средствам, позволяющим эффективно описывать структуру проектируемой системы.

Уже начиная с системного уровня проектирования, современные инженеры пересматривают свои взгляды на то, как проекты надо сертифицировать, как их надо разбивать на части, как их надо верифицировать и отлаживать.

До последнего времени системные инженеры и программисты опирались в своей работе на языки C и C++, в то время как их партнеры по разработке аппаратуры предпочитали схемотехническое представление проектов и лишь в последнее время стали описывать свои разработки на языках VHDL и Verilog. Расхождение во взглядах на целесообразность использования того или иного способа описания проекта наносит явный ущерб не только успешности взаимного общения специалистов (разрабатывающих начинку одного кристалла), но и, что более существенно, замедляет процесс проектирования и ухудшает его качество.

Пересмотр позиций, как будет видно из дальнейшего материала, необходим почти всюду. Целесообразно сосредоточить усилия на решении задач по четырем ключевым направлениям:

- создание новых средств и методологии проектирования;
- систематическое повторное использование базовых элементов проектов;
- использование передовых САПР;
- введения новых и модернизация существующих стандартов с целью повышения их эффективности.

## Методология проектирования

Новые проектные подходы потребовались не только из-за огромной сложности проектируемых схем (десятки миллионов вентилей), но и из-за таких проблем, как временная корреляция между логическими и физическими областями или достоверность верификации столь больших систем. Многие традиционные проектные средства и методологии не срабатывают на таких масштабах работы. Каждый шаг проектной процедуры, начиная от ввода проекта и кончая физической загрузкой, должен обеспечивать получение определенных промежуточных результатов. Если результаты проектирования на каком-либо этапе не соответствуют требованиям технического задания или существующим технологическим ограничениям, то результаты проектирования не могут быть переданы на изготовление.

Рассмотрим, к примеру, геометрические размеры. При их величинах на уровне квадратного микрометра и ниже задержка распространения сигнала определяется уже не столько процессами в транзисторах, сколько длиной межсоединения. Отсюда резко возрастает роль этапа размещения и разводки (устранение ошибок здесь зачастую выполнить сложнее, чем устранить ошибку в схеме на транзисторах). Проектировщик, для того чтобы убедиться в правильности работы схемы, должен дождаться топологических результатов и на их основе определить действительные задержки в приборе. Чем раньше эта информация будет получена, тем раньше можно вернуться (при необходимости) к более ранним этапам проектного потока и использовать данные для коррекции произведенных стадий работы.

Вопросы конструкторско-технологической реализации проекта в форме современных БИС ASIC становятся значимыми уже для первых шагов проектирования. Многие неявно видимые связи между физическим и схемотехническим воплощениями проекта оказываются весьма существенными.

Использование идей сопряженного проектирования для SOPC требует решения двух основных проблем. Первой проблемой является введение таких средств описания проекта, которые позволили бы сгладить противоречия между HW- и SW-реализациями. Второй проблемой (способ решения которой оказывается в значительной мере связанным с первой) является разработка и использование новой техники верификаций и моделирования. Существовавшие до последнего времени моделирующие средства не в силах за приемлемые сроки обработать последовательность событий в столь сложных системах или могут давать искаженную информацию. Выполнение моделирующих процедур на традиционных средствах может занять более года. Требуется введение определенных дополнительных шагов в верификационную процедуру, чтобы как можно раньше обнаружить те проектные ошибки, которые способны проявиться только после реализации проекта в форме кристалла, если они остаются незамеченными раньше.

## Систематическое повторное использование

Заполнение столь значительных объемов кристаллов (несколько миллионов вентилей) в ограниченные сроки (8—10 месяцев), определяемые новыми требованиями рынка (в первую очередь, мирового, да и российского тоже), требует существенного пересмотра строительных кирпичей, из которых будет возводиться здание. С одной стороны, хорошо бы, чтобы модули были как можно большего размера, их тогда меньше понадобится, но при этом возникает риск потерять управляемость свойствами проекта, и, соответственно, утратить шансы получить высокие характеристики. Нужен достаточно широкий выбор строительных элементов, набор должен быть очень тщательно проверен в разных условиях (отсюда желание, чтобы элементы уже применялись в других успешных проектах). Требуется исключительно высокая надежность и достоверность используемых модулей — после первого же



отказа их не будут больше применять. Многие авторы используют термин Reused, чтобы специально подчеркнуть возможность применения модуля не только в данном проекте, но и в других. Эффективность здесь заключается не столько в долгой жизни IP, а скорее в обеспечении совместимости и применимости их в последующих проектах. Для этого модули должны допускать возможность модификации и изменяемости параметров в достаточно широких пределах.

Отсюда основная проблема, где брать такие строительные кирпичи? Существуют два основных варианта: использование фрагментов собственных проектов или приобретение чужих разработок. Последний вариант является покупкой интеллектуальной собственности (IP) сторонних фирм, специализирующихся на выпуске IP-core, которые могут быть HW или SW. Потенциально возможен и третий источник приобретения — создание добровольной международной инициативы, собирающей и отбирающей лучшие разработки (по типу Linux, CPM, OpenC и т. д.). Однако для практической реализации такой инициативы требуется определенное время и достаточное количество желающих участвовать. Независимо от источника приобретения, основной проблемой является надежность модуля и возможность его достаточно простого встраивания в свой проект.

Значительный интерес представляет направление, предложенное в 1996 году фирмой Cadence. Суть его состоит в разработке компонентов, подключение которых к другим элементам системы будет определяться стандартом. Для этого стандартом вводится понятие *виртуальных компонентов* (Virtual Component, VC) и определяются правила их взаимного подключения. Компонент может быть отнесен к классу виртуальных, если он будет удовлетворять стандарту Virtual Socet Interface Alliance (VSIA).

Элементы набора могут существовать в разных формах:

- hard blocks, которые могут быть использованы для прямого физического размещения в кристалле;
- firm blocks, которые должны соответствовать определенным физическим требованиям фирмы-производителя кристалла;
- soft blocks, которые могут быть синтезированы и помещены в часть кристалла, отведенную для ПЛИС.

Очевидно, что HW-блоки должны допускать определенную подгонку (настройку параметров) в рамках семейства кристаллов. Более подробно вопросы практического использования этого подхода будут рассмотрены при анализе соответствующих САПР.

## Современные проектные средства

К современным проектным средствам относятся, прежде всего, новые САПР с новыми возможностями и построенные на новых принципах. Однако не следует ожидать, что наличие хороших САПР решит все проблемы.

От проектировщиков требуется отказ от устаревших методов и средств и овладение новыми. Понимание описаний, создаваемых автоматическими проектными средствами, необходимо для современного квалифицированного разработчика.

Внешняя простота общения с графическими интерфейсами современных САПР не должна заслонять важность уяснения проектировщиком организации и принципов работы узлов, являющихся результатами его творчества. Задание поведения узла, выполненное на одном из языков описания аппаратуры, или описание того же блока, но уже на уровне регистровых передач, должны восприниматься проектировщиком так же, как и схемотехническое задание структуры этого узла. Проектные средства, используемые для разработок SOPC, должны не только органически объединить два ранее разделенных проектных маршрута HW и SW, но и включить новые проектные средства. Следует учитывать, что стоимость подобных средств будет на начальных этапах весьма значительной (до нескольких сотен тысяч долларов).

### Эффективность стандартов

Стандарты требуются для всех уже рассмотренных участников проектного потока (включая и структуру самих SOPC). И даже для самих стандартов обычно не мешает ввести определенные ограничения, лучше в форме стандартов.

Возникновение и развитие методологии проектирования SOPC требует разработки или модернизации стандартов, которые создадут более простые, надежные, защищенные, закрытые от ошибок связи между различными уровнями проектирования: системными, логическими и физическими. Одним из наиболее широко используемых сейчас стандартов этого класса является стандарт SDF (Standard Delay Format). Заглядывая вперед, следует ожидать того же и от стандартов, поддерживающих обмены между проектными средствами не только в структурной, но и во временной областях. Соответственно, для физического уровня того же можно ожидать для библиотечных и проектных стандартов.

Связь между различными проектными средствами вполне может поддерживаться уже хорошо зарекомендовавшими себя в других областях (не SOPC) языками Verilog HDL, VHDL и EDIF, хотя некоторые изменения или расширения, диктуемые спецификой SOPC, могут потребоваться.

## 2.5.3. Стиль сопряженного проектирования и язык описания проекта

Все рекомендации, изложенные выше, окажутся эффективными только, если нацеленность на рациональный стиль сопряженного проектирования начинается с самого его начала, и всеми его идеями и подходами можно будет



пользоваться (без удвоения усилий) не только как можно дольше в проектной процедуре, но и начиная с наиболее ранних моментов проектирования.

Здесь все исследователи подчеркивают важность самого начального этапа проектирования, а именно этапа спецификации проекта, когда описание функционирования системы может еще не опираться на отнесение частей к аппаратной или программной реализации. Основным инструментом проверки и верификации становятся методы и средства системного уровня автоматизации (Electronic System Design Automation, ESDA). И первым вопросом становится выбор средства описания (спецификации) проекта. Важным оказывается возможность объединить разработку HW и SW, начиная с самых ранних этапов работы.

Несмотря на определенную дуальность представления системы в форме аппаратуры и в форме программного обеспечения МПС, существует огромный разрыв между уже имеющимися наработками и средствами, подготовленными на C и C++ для программного обеспечения, и наработками на этих же средствах для аппаратуры. Мир разработок в области аппаратных средств обладает не меньшим заделом, но на языках описания аппаратуры HDL.

Поэтому серьезным претендентом на роль языка системного уровня проектирования является язык C. Наследство от программных разработок, написанных на C, достаточно обширно, в то время как наследство аппаратных разработок в лучшем случае доступно на уровне регистровых передач (языки VHDL и Verilog). Для широкого использования язык системного уровня должен:

- сохранить систему понятий, использованных проектировщиками при разработке SW на языке C;
- добавить необходимую временную привязку к конструкциям языка C, чтобы обеспечить их функционирование в аппаратуре;
- обеспечить сосуществование (для целей со-проектирования и со-моделирования) с существующей аппаратурой, написанной на языках HDL (VHDL и Verilog).

Направление использования языков C и C++ в качестве основы подобного языка поддерживается рядом компаний. Наиболее яркий представитель этого направления фирма Synopsys. Поддерживаемая ею разработка под названием SystemC, будет более подробно рассматриваться ниже.

Вместе с тем, ряд компаний расценил создание SystemC как неудачную попытку использования традиционного программного языка для проектирования и моделирования, которая не дала ничего нового. Поэтому многие перешли к собственным разработкам и достигли определенных результатов.

Язык низкого уровня RTL хорошо определен и является широко используемым языком описания HW, поскольку трансляция из RTL в аппаратуру также хорошо определена и поддерживается многими производителями.

Вместо многих дней, необходимых для разводки ИС при ориентации на уровень индивидуальной транзисторной ячейки, даже язык низкого уровня RTL обеспечивает существенное ускорение процесса проектирования. Еще большей результативности можно достичь, если опираться на языки более высокого уровня.

Языки более высокого уровня могут поддерживать конструкции низкого уровня — уровня RTL. Исходно Very High-Speed IC (VHSIC) Design Languages включал VHDL (VHSIC HDL) и Verilog (Verifying Logic) HDL, причем Verilog планировался как язык моделирования. В настоящий момент оба языка VHDL International (VI) и Open Verilog International (OVI) сохранили и свои группы разработчиков, и своих пользователей, но организационно объединились под именем Accellera. Можно пытаться уменьшить различия между этими двумя подходами, но ни один из них не может (по крайней мере, в обозримом будущем) исчезнуть совсем.

Согласиться на один какой-то вариант или обеспечить (взаимное) конвертирование описаний проектов из одного вида в другой (на различных этапах), вот, пожалуй, и весь набор возможных вариантов. В настоящий момент отсутствует какое-то общепринятое мнение по этому вопросу, и поэтому далее мы более подробно попытаемся проанализировать плюсы и минусы различных подходов.

Фирма Co-Design Automation Inc. (**www.co-design.com**) разработала свое собственное подмножество языка Verilog под названием Superlog. Язык, в действительности, является смешением Verilog и C. Тем самым поддерживается простота общения с конструкциями обоих языков, но требуется языково-специфический компилятор. Выгодами Superlog является возможность прямого использования всех IP-языка Verilog без каких-либо модификаций.

Примером попытки совместить достоинства разных языков путем конвертации описаний с одного языка на другой является пакет CynLib фирмы CynApps (**www.cynapps.com**), ориентированный на C++. Библиотека пакета CynLib доступна в режиме on-line. Продажным продуктом фирмы являются интерфейс Verilog Co-simulation и пакет CynSuite, который имеет синтезатор (Cynthesizer), транслирующий коды C++/Cynlib в описания Verilog RTL.

Разрабатываются проекты, базирующиеся на расширениях и совмещениях существующих языков (HW и SW) с попыткой объединить в них достоинства обоих подходов. Типичным представителем этого пути стала фирма Cynergy Systems Design Inc. (**www.cae-plus.com**). Основной идеей здесь является создание языка, запатентованного под именем RTLC, который расширяет язык регистровых передач, используя язык C. Целый ряд компаний поддержал эту инициативу и работает с языком RTLC.

Пакет ArchGen содержит графическую оболочку, позволяющую генерировать описание проекта на языке RTLC, описание проекта может легко моделироваться и, базируясь на графической анимации элементов проекта,



упрощать решение проблем верификации и отладки. Другой продукт, Builder, этой же фирмы Application Specification Virtual Prototype (ASVP) также опирается на язык RTLC. Трансляцию (конвертацию) с языка RTL на язык RTLC может выполнять пакет Afterburner.

## Методы, устраняющие связь сложности систем и скорости моделирования

Ключевым моментом для любой из рассмотренных выше методологий является проблема моделирования, именно она предопределяет практическую применимость идей сопряженного проектирования и со-верификации. Небольшие проекты и проекты средней сложности могут с приемлемыми временными затратами моделироваться практически при любом виде входного описания. Для очень больших же систем SOPC потребуются также существенные вычислительные мощности или применение методов, ускоряющих сам процесс моделирования. Проектировщики двигаются различными путями, чтобы обеспечить компромисс между сложностью систем, точностью моделирования и временами как подготовки к моделированию, так и к его проведению.

Наиболее осторожные разработчики стараются использовать любые технические средства и приемы, чтобы обеспечить проведение как можно большего числа тестовых испытаний на уровне как фрагментов, так и целой системы, но осуществленных до ее изготовления. Как правило, при этом на первых шагах используются полные функциональные модели процессоров, а затем тестирование выполняется на уровне прототипных образцов аппаратных средств. Процедура проектирования, характерная для подобного подхода, была приведена на рис. 2.28.

Компромисс по скорости моделирования достигается при использовании в качестве составных частей предварительно тщательно проверенных IP, таких как процессорные ядра. Кроме того, как правило, у разработчика имеется несколько модификаций IP, отличающихся подробностью уровня описания, это позволяет выбирать уровень описания того же уровня, что и остальной проект. Однако зачастую более эффективным представляется моделирование процессорного ядра на высшем допустимом уровне представления. Например, моделирование на уровне кодирования логических команд может оказаться эффективнее, чем на вентильном уровне.

Для моделирования чаще всего используется подход, называемый моделированием *управляемым событиями* (Event-Driven Simulation, EDS). Подробно принципы EDS рассматриваются в *гл. 3*. Процесс моделирования больших проектов (например, современные ASIC содержат свыше трех млн. вентилей или более 20 млн. транзисторов) при использовании таких традиционных средств может растянуться до года. Специально для того, чтобы снять ограничения эффективности, свойственные моделировщикам, построенным на

принципах управления событиями, распространение получают средства моделирования, построенные на принципах *циклобазированного моделирования*.

Циклобазированное моделирование (Cycle-Based Simulation, CBS) использует специальные алгоритмы, которые исключают несущественные вычисления, чтобы поднять производительность расчетов. Основными ограничениями метода CBS являются:

- результаты моделирования вычисляются только на фронтах тактового сигнала;
- игнорируется поведение сигналов внутри фаз тактовых сигналов;
- для сигналов вычисляются только два логических состояния (0 и 1).

В результате, ограничивая объем вычислений, циклобазированное моделирование может обеспечить существенный выигрыш во времени моделирования.

Традиционные методы моделирования, управляемые событиями, наоборот, жертвуют эффективностью, чтобы обеспечить большую функциональность моделирования. Поэтому значения каждого активного сигнала вычисляются для каждого элемента в течение всего такта. Полные симуляторы EDS-типа обычно поддерживают:

- от 4 до 28 различных состояний сигналов;
- моделируют поведение элементов, представляющих собой транзистор, вентиль, RTL- или HDL-описание;
- выполнение временных вычислений для всех элементов схемы;
- полный стандарт HDL-языков.

Циклобазированное моделирование сфокусировано на функциональном поведении проекта и, следовательно, может быть оптимизировано для достижения этих целей. В результате, эффективность моделирующих средств этого типа превышает эффективность работы традиционных моделировщиков в 10—1000 раз.

Вместе с тем, отмечая положительные стороны метода CBS, не следует противопоставлять один метод верификации другому. Использование каждого из методов оказывается целесообразным для определенных условий применения. На рис. 2.29 приведен график, показывающий области целесообразного использования в процессе проектирования БИС высокой плотности различных верификационных средств.

На рисунке процесс проектирования условно поделен на три основных этапа отладки, на каждом из которых целесообразно применение специфических средств моделирования и верификации.

*Этап отладки отдельных модулей.* На начальном этапе верификационной процедуры целесообразно привлечение средств, построенных на принципах



событийного моделирования. Средства этого типа используются, поскольку они являются прекрасным инструментом для тщательного исследования отдельных модулей большой системы, когда требуется высокая степень интерактивности и детальность моделирования поведения исследуемых модулей. Как правило, число ошибок и нестыковок в проекте на этом этапе очень велико и может исчисляться десятками или сотнями.

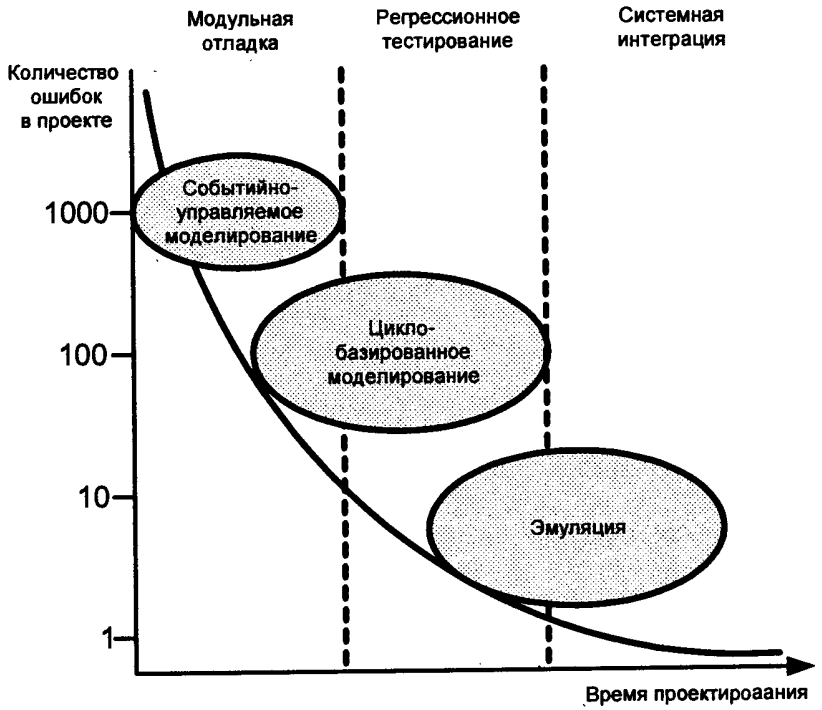


**Рис. 2.29.** Области применения различных средств верификации

*Этап регрессионного тестирования*, который требует периодических возвратов к своему началу после устранения очередной ошибки проектирования. Это этап верификации системы, образованной объединением отдельных модулей. Число погрешностей проекта уменьшено по сравнению с предшествующим этапом. Однако сложность проекта существенно увеличилась. Тестирующие средства этого этапа целесообразно строить на основе циклобазированного моделирования.

На последнем этапе — *системной интеграции*, необходимо выполнять комплексную проверку системы. Число необнаруженных ошибок проекта, как правило, уменьшено до единиц, но их выявление и локализация из-за резкого увеличения размерности решаемых задач серьезно затруднено. Це-

лесообразным инструментом проектировщика становятся эмуляционные средства.

Вариантом, пытающимся объединить различные подходы, является аппаратно-ассистирующее моделирование. В этом случае используется прототипирующая плата, содержащая такое количество элементов программируемой логики, что на этом наборе возможна эмуляция проекта с производительностью, в десятки или сотни раз превышающей производительность традиционных программных средств. Такая смешанная аппаратно-программная эмуляция может быть более комплексной, чем простая программная эмуляция.

Рассмотрев общие проблемы проектирования SOPC, можно перейти к разбору особенностей их проектирования на примерах конкретных схем и САПР.

## 2.5.4. Пакеты и САПР, поддерживающие проектирование SOPC

Фирмы-производители кристаллов SOPC (а в большинстве случаев это фирмы, ранее выпускавшие более простые схемы программируемой логики) сохраняют и традиционное разделение функций со сторонними фирмами, т. е., как минимум, оставляют за собой заключительные этапы проектирования кристалла — размещение элементов и трассировку межсоединений.

Однако, в отличие от обычных БИС ПЛ, проектирование схем SOPC требует проектирования и отладки программного продукта. Если проектирование SW еще может выполняться на средствах проектирования сторонних фирм, то отладка (по крайней мере, эффективная) требует применения совмещенных средств. В современных проектных средствах большое внимание уделяется включению средств разработки программных средств в проектные потоки разработки аппаратуры. Следует подчеркнуть, что это не просто механическое подключение еще одного программного средства, а обеспечение тесной интеграции всех средств проектирования для организации совместной работы с HW и SW и обеспечение их совместной отладки. Для этого в современных SOPC предусматривается отладка в рамках системы, интегрированной с исходной системой проектирования (Altera, Atmel, Triscend).

Фирма Atmel (**http://www.atmel.com**) для проектирования систем на базе семейства FPSLIC (AT94) предлагает использовать свою САПР: System Designer Ver. 2.1. Эта САПР в отличие от САПР других фирм не подключает, а включает в свой состав специальные версии программных средств, разработанные фирмой Mentor Graphic (синтезатор Leonardo, моделировщик ModelSim и со-верификационное средство Coverification). Описание проектирования для БИС этой фирмы можно найти в литературе [14].

Фирма Altera (**http://www.altera.com**) для схем семейства Excalibur (EPXA, EPXM) на последних этапах проектирования, включая отладку, предлагает



использовать свою САПР Quartus II. Для работы с программным обеспечением САПР обеспечивает переключение своего традиционного рабочего потока в специальный режим проектирования программного обеспечения (SoftMode), который подключает компилятор C/C++ и отладчик. Наиболее рациональной, однако, фирма считает совместную работу со средствами сторонних фирм. Примером подобной интеграции могут служить полные функциональные САПР Quartus II и MAX+PLUS II, синтезирующие средства от Synopsys и Exemplar и моделирующие средства от Model Technology. Более подробно организация корпоративной работы с продукцией фирмы Mentor Graphic будет рассмотрена ниже.

Фирма Triscend (**http://www.triscend.com**) для проектирования на основе производимых схем семейств E5 и A7 предлагает использование САПР: Fast Chip Triscend Ver. 2.3. Проектный поток, рекомендуемый фирмой, будет приведен при рассмотрении практического примера в *гл. 4*.

Разработку SOPC корпоративными усилиями разных фирм можно наблюдать, рассматривая САПР и проектные потоки, используемые фирмами QuickLogic, Cypress MicroSystems и Xilinx.

Фирма QuickLogic (**http://www.quicklogic.com**) для поддержки проектирования своих БИС QuickMIPS ESP (на 80% встроенная аппаратура, включающая ЦП MIPS 4Kc, 16 Кбайт память команд и 16 Кбайт данных и интерфейсные узлы, и на 20% FPGA с перемычками типа ViaLink) использует САПР QuickMIPS development environment для процесса сопряженного проектирования, включая отладку. САПР этой же фирмы QuickWorks Ver. 8.21 поддерживает все этапы проектирования БИС: схемотехнический ввод проекта, синтез и компиляцию, временной анализ, функциональное и временное моделирование, конфигурирование. Имеет лицензионные соглашения на подключение пакетов САПР других фирм (компилятора Simplify фирмы Simplicity, HDL-редактора TurboWriter, моделировщика фирмы VeriBest).

Поскольку выпускаемые фирмой Cypress MicroSystems (**http://www.cypress.com**) БИС SOPC семейства CY8C25***/26*** содержат не только типичные для всех SOPC части (МП и ПЛИС), но и аналоговые блоки, то весьма интересна организация фирменной САПР: PSoC Designer IDE. САПР представляет собой интегрированный пакет средств разработки встраиваемых PSoC-систем. Пакет обеспечивает доступ ко всем традиционным средствам, включая полный отладчик, поддерживающий внутрисхемную эмуляцию In-Circuit Emulator (ICE). Основой IDE является редактор проекта (Device Editor), обеспечивающий легкий в использовании графический интерфейс для реконфигурируемости подключаемых к МК устройств (MCU). В состав Designer IDE входят редактор приложений, библиотеки, ассемблер и компилятор языка C, загрузчик, линкер, отладчик и программатор.

Однако на предварительном этапе проектирования для моделирования проектов, содержащих смешанные аналого-цифровые цепи, фирма Cypress

предлагает использовать ПО лидирующей фирмы — Antirim Design System (Antirim-ACV, Antirim-MSS).

Для проектирования кристаллов типа SOPC фирмы Xilinx (**www.xilinx.com**) может использоваться САПР этой фирмы ES Series. Интерес представляет сообщение о выпуске САПР, поддерживающей полный цикл проектирования, ориентированного на описание и SW, и HW на языке C. Примером одного из первых коммерческих вариантов реализации этого подхода к сопряженному проектированию на базе языка C является анонсированная фирмой Celoxica Limited проектная САПР под именем DK1. Помимо обычных возможностей пакет поддерживает еще одно важное свойство будущей системы: обеспечивает не только быстрое (за недели и дни вместо традиционных лет) создание электронной продукции, но и реализует фундаментально новый подход к проектированию электронной аппаратуры, позволяя лучше управлять жизненным периодом продукции за счет поддержки средствами Интернета корректировки аппаратуры во время ее эксплуатации.

Пакет DK1 фирмы Celoxica Limited разрешает использовать навыки разработчиков программного обеспечения в создании аппаратуры. Разработчик записывает сложные алгоритмы поведения проектируемой системы на языке Handel-C (языке высокого уровня, базирующемся на стандарте ANSI C) для их преобразования в аппаратуру. В отличие от подходов, когда требовалась перекомпиляция C-подобных программ на промежуточный язык описания аппаратуры, пакет DK1 обеспечивает прямое преобразование программы, написанной на языке C, в аппаратную реализацию. DK1 позволяет на последующих стадиях осуществлять перевод проверенных рынком проектов, реализованных в форме FPGA, в форму традиционных ASIC-решений. Хотя некоторые фирмы-производители конечной продукции могут продолжать выпуск продукции в форме FPGA, чтобы сохранить потенциальные возможности корректирования и улучшения продукции, опираясь на ресурсы сети Интернет. Пользователи, использующие Internet Reconfigurable Logic (IRL), могут преобразовывать C-ориентированные проекты прямо в список соединений FPGA семейства Virtex фирмы Xilinx без перевода их описаний на языки VHDL, Verilog или в схематическое представление. По мнению разработчиков DK1, в результате проектирование аппаратуры не будет требовать привлечения специалистов-схемотехников, а окажется доступным для прикладных специалистов, включая системных инженеров и инженеров-программистов. Помимо оболочки, Integrated Development Environment (IDE), пакет DK1 содержит весь набор проектных средств, включая компилятор, моделировщик и отладчик. Этот набор средств обеспечивает выполнение всех типовых проектных этапов, включая компиляцию, моделирование, оптимизацию и отладку проектов, написанных на высокоуровневом языке Handel-C. Проектировщик может ориентироваться на использование поставляемых фирмой прототипных плат и универсальных печатных плат (типа RC1000), которые готовы к размещению пользовательских проектов,



ориентированных на FPGA семейства Virtex, и могут прямо вставляться в аппаратуру разработчика.

Пакет работает на IBM PC-совместимых компьютерах под ОС Microsoft Windows 98, 2000 или NT4.0 и поддерживает БИС фирм Xilinx и Altera. Начальная цена 25 тыс. долларов в год для трехгодичной лицензии.

Количество фирм, поддерживающих различные аспекты и этапы проектирования кристаллов SOPC, весьма велико, предлагаемые ими решения представляются интересными и перспективными и могут являться предметом самостоятельного исследования. Однако основной тон, так же как и в других производственных сферах, задают крупные фирмы, материальные, технические и другие ресурсы которых позволяют им разработать и предложить не отдельное решение, а группу согласованных решений, образующих интегрированную систему. Наибольший интерес представляют те предложения, которые могут в ближайшем будущем получить широкое распространение и останутся надолго. Как правило, такими решениями являются разработки крупных фирм, специализирующихся на выпуске САПР. Среди фирм, уделяющих значительное внимание проблемам создания проектных средств в рассматриваемой области, наибольший вес имеют фирмы Mentor Graphics, Cadence и Synopsys. Поэтому ниже и рассматриваются основные программные продукты этих фирм, поддерживающие проектирование систем SOPC.

## Фирма Mentor Graphics

Одним из основных современных средств со-проектирования, получивших широкое распространение, является пакет Seamless-CVE фирмы Mentor Graphics. Его задача состоит в построении виртуального прототипа аппаратуры для работы прикладного SW. Наличие у фирмы программных моделей большинства процессорных ядер различных фирм позволяет приступить к написанию и отладке программ практически сразу после выбора типа микропроцессорного ядра. Большое значение имеет и тот факт, что могут использоваться модели разной степени детализации. Пакет обеспечивает разбиение проекта на части, со-верификация выполняется с высокой производительностью, с использованием моделировщика системы команд, когда моделируется процессорное ядро.

Основное отличие пакета состоит как раз в том, что если моделирование аппаратной части SOPC выполняется традиционными средствами пакета ModelSim, то моделировщик процессорного ядра построен на принципах, отличных от принципов, используемых в традиционных средствах моделирования. Благодаря этим изменениям скорость со-моделирования аппаратных и программных частей системы удалось увеличить в несколько сот раз при сохранении детальности анализа работы аппаратной части системы.

На рис. 2.30 приведен проектный поток, используемый при разработке схем класса Excalibur фирмы Altera. Проектирование предполагает совместную

работу САПР разных фирм. Как видно из рисунка, пакет Seamless используется в со-верификационной процедуре. Такое же применение (но во встроенном исполнении) пакет находит в проектном САПР фирмы Atmel.

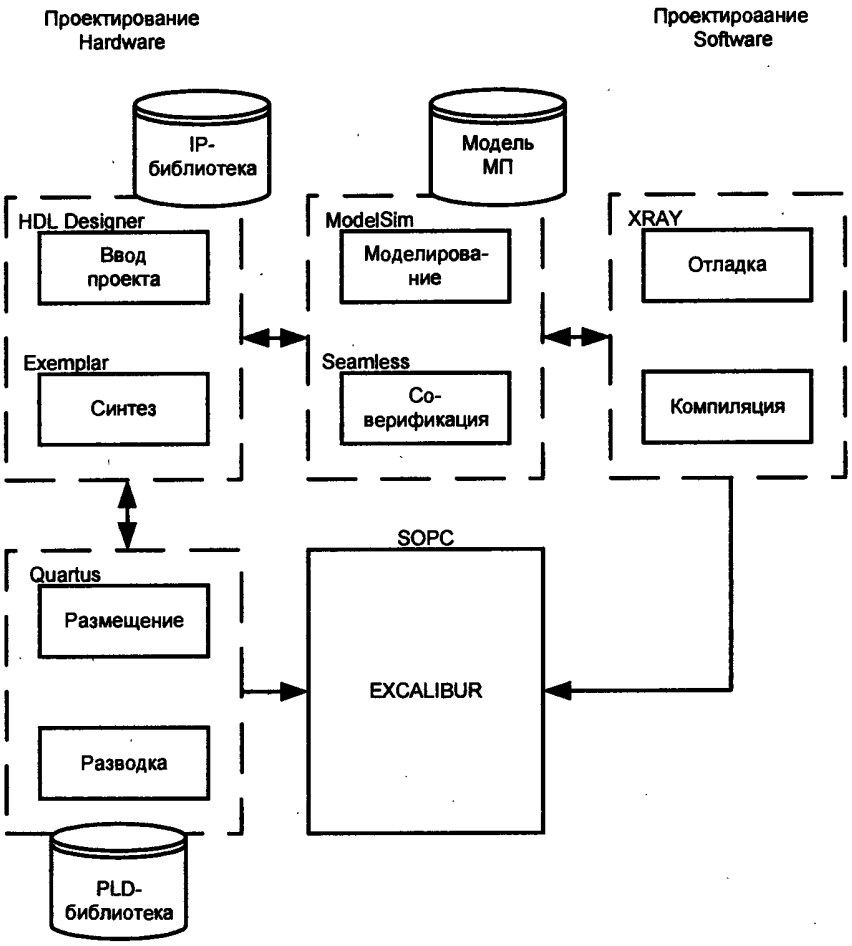


**Рис. 2.30.** Проектный поток для SOPC фирмы Altera

## Фирма Cadence

Фирма Cadence Design Systems, Inc. (**http://www.cadence.com**) выпускает большое количество САПР, отличающихся целевым назначением, стоимостью, технической и программной платформой и т. д. С точки зрения сопряженного проектирования наибольший интерес могут представлять два направления работ этой фирмы.



Первым является направление, связанное с использованием понятия *"виртуальные компоненты"*. Упрощения создания систем на идеях сопряженного проектирования фирма Cadence пытается добиться в пакете Virtual Component Codesign (VCC). Пакет поддерживает спецификацию проектов на различных языках, включая C, C++, MATLAB и SDL, а также систему SPW (Signal-Processing Work system), ускоряющую проектирование на системном уровне сложных систем цифровой обработки сигналов. Пакет предназначен для распределения задач между HW и SW, для анализа загруженности шины и процессора и способен контролировать процедуру распределения задач и состязания за ресурсы в системах реального времени RTOS. Коммуникативные ссылки помогают конвертировать абстрактные интерфейсные описания гексемного уровня в действительный уровень сигналов интерфейса. Со-верификация VCC включает симуляцию, так же как и поддержку Affirma HW/SW верификатора фирмы Cadence.

Вторым важнейшим направлением деятельности фирмы Cadence можно считать работы над созданием САПР, ускоряющих моделирование проектируемых систем. Фирмой выпущена САПР SpeedSim — программный пакет спроектированный специально для уменьшения времени и цены со-верификации больших проектов. Пакет был успешно использован для производственной верификации проекта, содержащего более 5 млн. вентилей. Построен он на основе метода *циклобазированного моделирования* для того, чтобы снять ограничения эффективности, свойственные традиционным моделировщикам, базирующимися на принципах управления событиями. Результирующая эффективность пакета в десятки или тысячи раз превышает эффективность последних. Существенного ускорения работы пакета удалось достичь путем его реализации в форме однопроходного компилятора с языка Verilog, как показано на рис. 2.31.

Для получения высокой эффективности при создании SpeedSim в него были заложены различные новейшие технологии. Можно выделить пять основных особенностей реализации пакета.

- *Технология BDF* (Boolean Data flow Engine). Пакет генерирует исключительно эффективный программный код за счет ориентации на внутренний код моделирующей ЭВМ.
- *Минимизация памяти.* Объемы требуемой памяти по сравнению с пакетами на базе EDS уменьшены от 5 до 50 раз.
- *Совместное тестирование.* На одной рабочей станции с одной копией проекта может одновременно моделироваться до 32 различных тестов.
- *Режим работы Symmetric Multi-Processing.* Предусмотрено функционирование пакета в мультипроцессорной системе, когда он может распараллелить решение задачи между 8 процессорами, работающими с одним сервером.

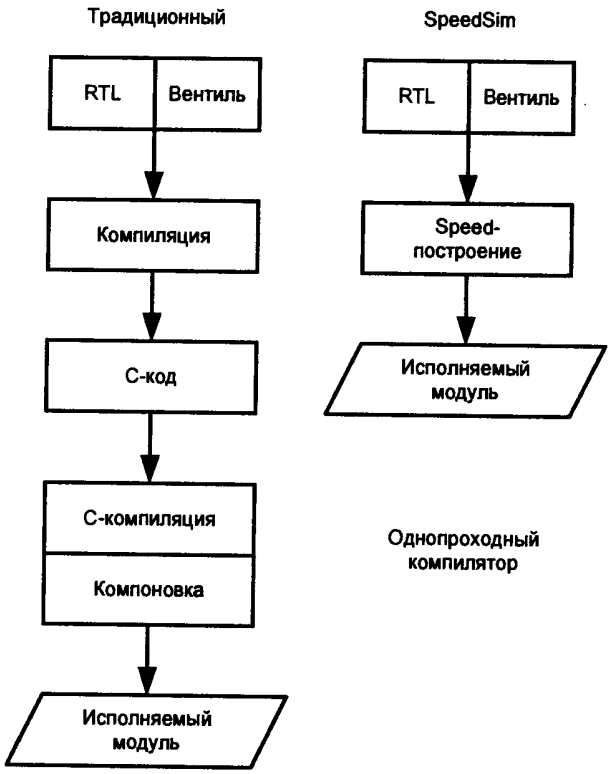


**Рис. 2.31.** Варианты построения компиляционных процедур

- *Быстрые возвратные итерации.* В пакете учитывается итерационность процесса тестирования: а) необходимо как можно быстрее обнаружить ошибку и локализовать ее; б) надо как можно быстрее исправить ошибку и перезапустить верификационную процедуру с того места, где существовавшая до того ошибка искажала результат.

В результате пакет SpeedSim на относительно простых вычислительных системах в практически приемлемые сроки позволяет моделировать проекты, содержащие несколько миллионов вентилей.

## Проектный поток в SystemC

Ориентация большинства системных инженеров на описание систем на языке C и некоторые другие обстоятельства привели к тому, что на системном уровне проектирования наиболее распространенным оказался в настоящий момент вариант, ориентированный на язык C. В методологии C/C++ и программное обеспечение, и аппаратура описываются на языке C. Один язык описания позволяет после компиляции проще организовать со-



местную верификацию. Большая распространенность такого подхода опреляется тем, что проектировщик может опираться на ряд уже созданных оектных средств. Этот ряд включает проект SystemC, создание которо было инициировано фирмой Synopsys, пакет N2C фирмы CoWare ww.coware.com) и пакет Ptolemy — для быстрого виртуального прототипивания.

### бзор проектного потока и базовых средств SystemC

сентябре 1999 года ряд ведущих компаний, разрабатывающих САПР, IP и О для системных и встраиваемых приложений, объявил об инициативе здания Открытой Системы C (Open SystemC Initiative, OSCI) и немедленой ее доступности и свободной загрузке. Моделирующая C++ платформа ыла названа SystemC. Управляющая группа инициативы включала такие рупные фирмы, как ARM, CoWare Inc., Cygnus Solutions, Ericsson, Fujitsu, nfineon, Lucent Technologies, Sony Corporation, STMicroelectronics, Synopsys nc. и Texas Instruments. (За основу рассмотрения возьмем версию SystemC 0, анонсированную в марте 2000 г.).

ундаментом системы является специализированная библиотека классов ++ и проектный маршрут. Назначением подготовительного этапа проекрования является создание моделей, соответствующих поведению процесора и интерфейсных компонентов аппаратуры на цикловом уровне. Базоым принципом был принят иерархический подход, использование которого ожет распространяться от абстракций достаточно высокого уровня, до низоуровневой логики. В основу SystemC положили ряд классовых шаблонов, пределяющих небольшой набор новых понятий. К ним относятся асинронность временных операций, методика соединений, эффективность отлеживания поведения объектов программы, регистрация и отладка событий проекте. Знание адекватных понятий VHDL или Verilog очень помогает ри изучении SystemC: несмотря на расхождение в синтаксисе, в семантике се же больше совпадений, чем расхождений.

се строительные блоки в SystemC являются объектами (classes). Фундаменальным строительным блоком является процесс (process). Process подобен ункции в языках C и C++, описывающей некоторое поведение. Описание аконченной системы представляет собой множество параллельных процесов, связанных друг с другом посредством сигналов (signal). Функционироание процессов может определяться тактовыми сигналами (clock), используемыми для формирования событий или синхронизации процессов.

ля эффективного моделирования HW требуются специальные типы данных, также являющиеся частью библиотеки SystemC. В состав библиотеки ходят описания одних и тех же модулей, но на разном уровне абстракции именно это позволяет обменивать скорость трансляции и моделирования на точность и адекватность). На высшем уровне иерархии HW системы может быть описано исключительно на функциональном уровне, для более

точного моделирования поведения аппаратуры потребуется либо функциональное ее описание, либо описание на уровне регистровых передач. Программная часть системы пишется на средствах традиционных C и C++. Что касается интерфейса между HW и SW, или между HW- и HW-блоками, то он может описываться на следующих уровнях — на уровне обменов транзакциями (transaction-accurate level) или на уровне цикловых обменов (cycle-accurate level). В пределах одного проекта допускается произвольное смешение уровней детализации описаний. То же самое относится и к созданию программных тестов (Test-Bench). Для получения окончательных программ, подаваемых на вход любого компилятора, совместимого с ANSI C++, SystemC содержит не только стандартные библиотеки классов, но и компоновочную библиотеку, которая включает моделирующее ядро. Программное обеспечение системы разработки (рис. 2.32), в составе которой обычно используется моделирующая SystemC, позволяет эффективно создавать, модифицировать и отлаживать проект разработчика.

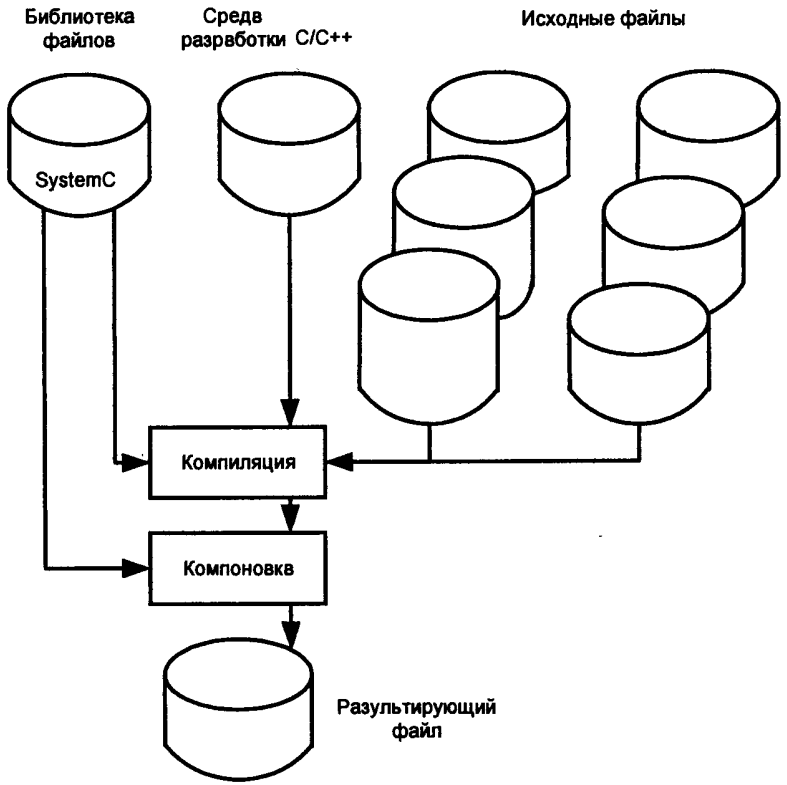


**Рис. 2.32.** Структурный состав среды разработки с SystemC



Подробное описание SystemC можно найти в Reference Manual Release 1.0. Здесь же остановимся на кратком описании тех базовых понятий, добавление которых к универсальному языку программирования позволило в совмещенном варианте моделировать совместное поведение HW и SW. В общем случае, набор этих понятий мог бы иметь другое синтаксическое оформление, однако важным представляется их целевое назначение и, как следствие, обязательное присутствие подобных конструкций в языке, претендующем на системный уровень спецификации аппаратно-программных систем.

- Modules — набор, содержащий классы. Модули допускают иерархическую организацию и могут включать другие модули и/или процессы. Модули имеют (однонаправленные или двунаправленные) порты, которые, по сути, являются функциональными интерфейсами, обеспечивающими соединение модулей между собой. Наличие интерфейсов позволяет скрывать внутри модуля детали его реализации и удобны для включения в состав модулей различных блоков IP.
- Processes — процессы, которые используются для описания выполняемых функций. Процессы могут быть как одиночными объектами, так и содержать внутренние модули.
- Signals — SystemC поддерживает два типа сигналов: resolved и unresolved. Сигналы типа resolved, в отличие от сигналов типа unresolved, могут управляться более чем одним драйвером (в том числе шиной).
- Rich set of signal type — для поддержания моделирования на различных уровнях абстракции (от функционального до уровня регистровых передач), включая и поддержку моделирования SW, SystemC поддерживает значительное число типов сигналов. Этим язык отличается от языка типа Verilog, который поддерживает только простой и векторный битовый типы сигналов. SystemC поддерживает и двузначный, и четырехзначный типы сигналов.
- Rich set of data type — SystemC содержит обширный ряд типов данных, что позволяет поддерживать множественные проектные наследования и абстрактные уровни. Типы данных с фиксированной точностью позволяют ускорять моделирование при сохранении приемлемой точности его результатов. Типы данных с произвольной точностью (ограничения на задаваемую точность отсутствуют) могут использоваться для вычислений с повышенной точностью и моделирования больших шин. Система поддерживает и двузначные и четырехзначные типы данных. Система включает и богатый ряд перегружаемых операторов и типов, а также механизмов преобразований для этих типов данных.
- Clocks — SystemC содержит определение тактовых сигналов (Clocks) как специального типа сигналов. Такты хронометрируют систему в течение моделирования. Поддерживается множественность тактовых сигналов с различными фазовыми взаимоотношениями.

- Reactivity — для моделирования реактивности поведения SystemC имеет механизмы ожидания фронтов и спадов тактовых сигналов, событий или сигнальных изменений. Система также поддерживает ожидание определенных событий, не связанных с выполнением какой-либо определенной стадии процесса (наиболее типичным примером является ожидание асинхронных сигналов сброса).
- Multiple abstraction levels — SystemC поддерживает моделирование на различных уровнях абстракции, выполняя моделирование от высших уровней функциональных моделей до детализированных моделей на уровне RTL. Система поддерживает итеративное уточнение моделей от высших к низшим уровням абстракции.
- Creating functional models — для работы с абстрактными функциональными моделями SystemC вводит понятие коммуникационного примитива, называемого каналом (Channel). Канал представляет собой специальный тип сигнала, при помощи которого могут взаимодействовать между собой синхронные или асинхронные процессы. Канальное взаимодействие отличается от обычного сигнального взаимодействия. Для обеспечения записи в канал или чтения из канала требуется определенное число тактовых циклов, однако затраты времени требуемого на обеспечение соединения не фиксируются, поскольку не являются существенным свойством канала. Обычно каналы являются средством, используемым на начальных стадиях моделирования, когда основной интерес представляет системное функционирование. В подобных ситуациях каналы являются механизмом, призванным гарантировать надежность обмена данными. Каналы в неявной форме производят необходимое квитирование обмена для гарантирования корректности доставки данных от передатчика к приемнику. В последующих версиях системы понятие каналов расширено до уровня многоуровневой коммуникационной семантики, что позволяет описывать как внутренние (SOPC), так внешние (ввода/вывода) протоколы с различным уровнем коммуникационной абстракции (уровень передачи данных, уровень шинного цикла, уровень тактового цикла).
- Cycle-based simulation — для обеспечения высокой скорости моделирования SystemC включает в свой состав ядро моделирования на уровне циклов моделируемой HW-системы. При этом сохраняются механизмы для управления моделированием из любой точки входной спецификации.
- Debugging support and waveform tracing — классы, введенные в SystemC, имеют встроенные средства контроля ошибок реального времени исполнения (run-time error checking), которые могут зависеть от течения компиляции. Ядро системы содержит базовые программы, которые позволяют запомнить временные диаграммы внутренних сигналов моделируемой системы в выходных файлах (с форматами типа VCD, WIF или ISDB), что позволяет в дальнейшем просматривать их содержимое типичными редакторами временных диаграмм.



## Процесс проектирования в системе SystemC

Процесс проектирования начинается с того, что разработчики создают *функциональную спецификацию* системы. Целью является проверка функционирования алгоритмов и системы. Функциональная спецификация представляет собой описание связей, соединяющих при помощи сигналов и каналов взаимодействующие процессы. Процессы соответствуют определенным действиям и могут требовать подключения различных архитектурных блоков, которые могут быть реализованы в форме HW или SW. Поскольку функциональная работоспособность системы может быть проверена путем компиляции и последующего исполнения программы C++, то функциональная спецификация может быть преобразована в архитектурную спецификацию.

В *архитектурной спецификации* процессы отображают работу таких действительных аппаратных блоков, как процессоры, память, блоки прямого доступа к памяти или оборудование ввода/вывода. Взаимодействие между процессами осуществляется при помощи сигналов, соответствующих реальным соединениям в аппаратуре. Таким образом, в архитектурной спецификации происходит разделение на программные и аппаратные компоненты, но поскольку для описания любых компонентов использован один и тот же язык, то возможны любые архитектурные перестановки и изменения, вероятны даже изменения и замещения между программными и аппаратными реализациями.

Завершение архитектурной спецификации и выполнение ее компиляции позволяет не только убедиться в работоспособности проекта, но и определить основные характеристики производительности проектируемой системы. В соответствии с ранее данным определением это и есть один из вариантов реализации сопряженной верификации и сопряженного моделирования. Основное назначение действий этого этапа — проверить эффективность различных вариантов архитектурного описания, поэтому для ускорения перебора различных вариантов реализации желательно, чтобы модели этого уровня описания проекта были максимально простыми и отражали наиболее существенные характеристики моделируемых блоков. Именно поэтому на описываемом уровне для процессоров, например, используется модель, соответствующая шинной организации функционирования. Такой же максимально абстрактный уровень описания функционирования применяется и для других блоков системы.

Если основные вопросы построения архитектуры системы решены, то естественен переход к следующему этапу проектирования. На этом уровне все аппаратные и программные блоки должны быть уточнены и доопределены добавлением необходимых деталей и введением конкретных ограничений на реализацию. Поскольку HW- и SW-средства работают совместно (параллельно-последовательно) со-верификационные действия должны выпол-

няться непрерывно, чтобы обеспечить правильную работу системы после подобных корректирующих действий. Усложнение и детализация описаний аппаратных и программных блоков заставляет переходить к более подробным и детальным описаниям модулей. Для процессора, например, целесообразно подключение к модели функционирования шины Bus-Functional Model (BFM) еще и модели работы системы команд Instruction Set Simulator (ISS), для многих других блоков потребуются более подробные поведенческие модели или описания на уровне регистровых передач (RTL). Основной проблемой при этом является, как уже неоднократно отмечалось выше, проблема резкого увеличения временных затрат на со-моделирование.

Как правило, начиная с этого момента, работа с HW и SW разделяется. Для доведения аппаратуры до вентильного уровня должны использоваться соответствующие синтезирующие средства. Чаще всего требуется перевод описания на языки описания аппаратуры HDL (VHDL или Verilog), а уже затем синтезирующая процедура. Работа над обоими направлениями не всегда может идти синхронно. Достаточно часто SW является источником противоречия между повышенной производительностью системы и минимизацией ресурсных требований. Для устранения противоречий необходимо проанализировать большинство фрагментов системы при их реализации в форме HW- и SW-проектов. Окончательное решение определяется таким выбором HW, которое обеспечит максимальную эффективность работы SW в соответствии с системными ограничениями.

Между ошибками, допущенными в HW и SW, существует большая разница. Если проектировщик допустил ошибку в аппаратуре, то и программное обеспечение будет работать с ошибками. Корректность работы обоих частей проекта является крайне важной. Вместе с тем, средства разработки SW оказываются относительно независимыми от средств сопряженного проектирования, и поэтому разрабатываемый SW-модуль (код) может много раз с исправлениями возвращаться в верификационную процедуру. Таким образом, поступают многие средства со-проектирования, чтобы обеспечить механизм итерационных возвратов.

SystemC не является единственным вариантом решения проблемы и был выбран исключительно из-за своей распространенности, например, другим С-базированным решением является пакет CycleC фирмы C-Level Design Automation (**www.cleveldesign.com**), написанный на ANSI C.

## 2.5.5. Понятие платформенно-базированных проектов SOPC

Значительное количество современных разработчиков поддерживают платформенно-базированное проектирование SOPC. Основной задачей такого подхода является сокращение времени проектирования путем сосредоточе-



ия внимания проектировщика только на тех аспектах проекта, которые тличают его проект от любых других. Подход не отрицает и не противоре-ит большинству подходов, рассмотренных выше, а только добавляет новые войства. Обычно под платформенно-базированным проектированием (далее ПБ-проектирование) понимают проектирование, построенное на стандарт-ом ряде модулей интеллектуальной собственности (IP), разработанных той ли иной компанией. Как правило, такой набор включает набор процессо-ов, память с различной организацией, определенные блоки периферии, онтроллеры прерываний и ПДП. Этот подход является одним из вариантов опытки создать свой собственный проект SOPC без существенных старто-ых издержек (сюда не входит стоимость приобретения прав на IP).

ыгоды подхода связаны с тем, что базовый набор, как правило, хорошо пределен, протестирован и не требует значительных переделок для других одификаций (если они не выходят за рамки подключения этого IP к дру-им IP данного набора. Очевидно, что здесь допустимы и известные из-ержки, связанные, например, с тем, что не все стандартные компоненты абора платформы будут соответствовать пожеланиям проектировщика, а одернизация IP, особенно если они зашифрованы, может оказаться делом олее сложным, чем самостоятельная разработка.

есмотря на начальный этап развития направления ПБ-проектирования OPC, уже наметились некоторые различия в подходах у разных фирм. Од-ой из ведущих выступила фирма CoWare с пакетом N2C.

ерьезным и интересным конкурентом выступает фирма Virtio Inc. **ww.virtio.com**), использующая свою САПР MagicC как компилятор HDL-оделей. Самым интересным здесь является то, что моделирование и тести-ование осуществляется в режиме on-line на сайте фирмы.

роведем сначала краткий анализ самих платформ и общих понятий и про-ем, а затем рассмотрим особенности их проектирования.

латформа Xtensa фирмы Tensilica Inc. (**www.tensilica.com**) относится к клас-ПБ-проектирования SOPC (может быть отнесена к подклассу проек-руемых и оперативно настраиваемых SOPC) и является 32-разрядной 00 МГц архитектурой с дополнительным DSP. Отличительной особен-остью процессора Xtensa является наличие специального процессорного енератора Processor Generator, который поддерживает введение в систему овых команд, определяемых пользователем и задаваемых в формате фир-енного языка Tensilica Instruction Extension Language (TIEL).

ругим вариантом платформы являются реконфигурируемые SOPC, типич-ый представитель которых — семейство AT94K класса приборов FPSLIC фирмы Atmel или реконфигурируемый коммуникационный процессор Reconfigurable Communications Processor (RCP) CS2000 фирмы Chameleon Systems Inc. (**www.chameleonsystems.com**).

Представители фирмы STMicroelectronics, Geneva (**www.isdmag.com /story /oeg20010228s0084**) предлагают вариант гибкого ПБ-проектирования в рамках консорциума Virtual Socet Interchange Alliance (VSIA), активным членом которого является STMicro.

Технологию конфигурируемой SOC платформенно-базированного проектирования активно разрабатывают и представители фирмы Palmchip (**www.palmchip.com**).

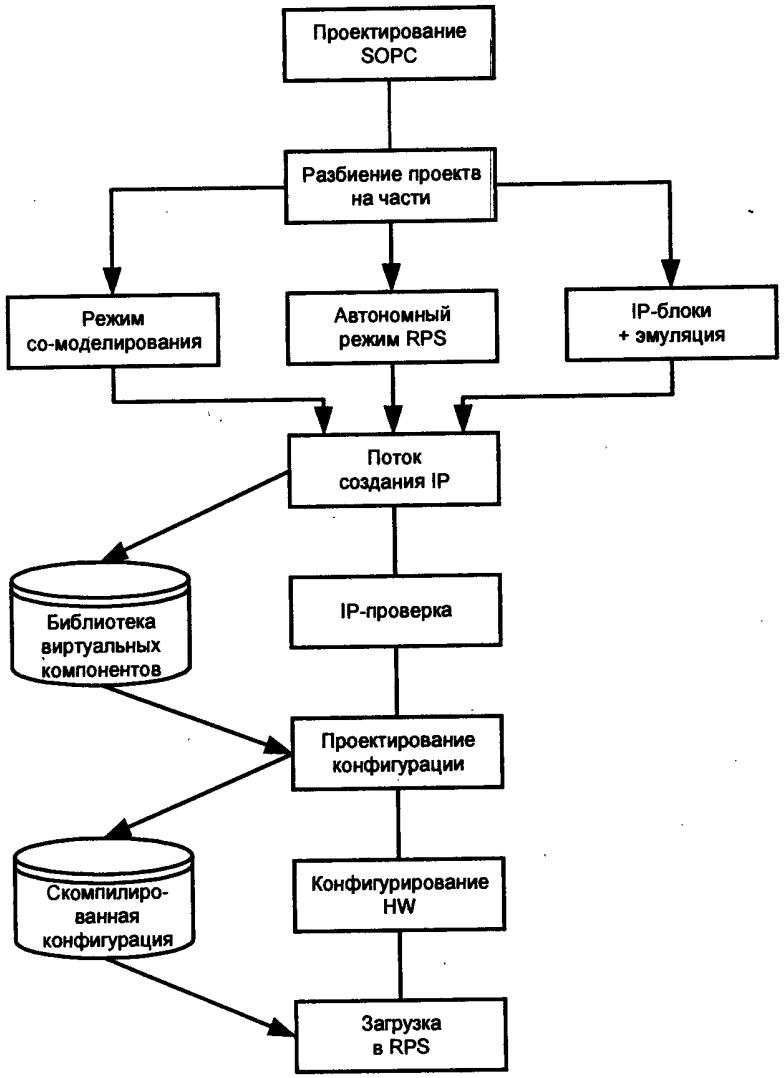


**Рис. 2.33.** Верификация в платформенно-базированном проектировании



дно из интересных решений проблемы верификации предлагает фирма uickturn Design System (**www.quickturn.com**) в своей разработке под названи- Rapid Prototyping System (RPS). RPS представляет собой реконфигури- емую прототипную систему, содержащую индивидуальные логические оки, или IP-core. Блоки оформлены в форме небольших плат расширения , называемых платы ядер и коммутируемых на общей плате, собственно и яющейся RPS. На одной коммуникационной плате может размещаться 30 плат ядер, каждая из которых может содержать IP в форме bonded-out re, реальным кристаллом или SW-моделью, помещенной в FPGA. Про- дура верификации при ориентации на продукцию фирмы Quickturn Design stem может иметь вид, представленный на рис. 2.33.

смотря на обилие приведенных выше работ в плане создания САПР для -проектирования, со-верификации и со-моделирования, следует иметь в ду, что, в основном, это еще поисковые исследовательские работы. Какие оектные средства получат наиболее широкое распространение в будущем едсказать достаточно трудно, поскольку успех тех или иных решений ле- т в плоскости пересечения различных требований, возможностей и вос- ебованности.

## 2.6. Некоторые аспекты технологии производства систем с ПЛИС

ажнейшей задачей, встающей перед разработчиками современной элек- онной аппаратуры, является создание (или использование готового) тех- логического инструментария и методологии его применения для различ- ых этапов жизни электронного изделия. Важность этих вопросов связана с м, что системная верификация и генерация тестовых последовательностей жет требовать до 70% проектного времени. Традиционно выделяются сле- ющие этапы, связанные с разработкой и использованием специального хнологического оборудования: макетирование проекта, изготовление опыт- ых образцов, изготовление серийных образцов, модификация и замена таревших версий изделия. Каждый из этапов характеризуется как специ- ическим перечнем задач, так и требуемым инструментарием, специальной валификацией персонала и методологией работы. На современном уровне азвития технологии разработки электронных компонентов необходимость рименения специального технологического оборудования смещается в об- асть все более ранних этапов проектирования. Теперь даже не всегда про- то различить, где инструментарий для проектирования, а где части проек- тируемого изделия.

Уже на этапе спецификации проекта разработчик должен заложить в буду- щую конструкцию целый ряд технологических "зацепок", отсутствие кото- рых на более поздних этапах может оказать крайне негативное влияние.

Задачей *макетирования* является проверка реализуемости выбранного способа решения и получение экспериментальных данных для оценки достижимых параметров будущей системы.

Целью *изготовления опытного образца* является экспериментальная проверка работоспособности системы в предполагаемых условиях эксплуатации. Как правило, эта цель достигается путем организации тестирования опытных образцов в различных условиях использования. Качество тестирования, его полнота и адекватность выводов и принимаемых решений определяют возможность перехода к выпуску серийных изделий.

*Изготовлению серийных образцов* предшествует разработка технологии изготовления с учетом ее стоимостной оптимизации при повышении эффективности, а также разработка и изготовление необходимого тестового инструментария. В понятие "тестовый инструментарий" входят как специфическая аппаратура, так и набор специальных тестовых процедур, алгоритмов тестирования и т. п. Важной предпосылкой эффективности выпуска серийной продукции является включение в разрабатываемый проект специфических элементов тестирования на наиболее ранних этапах проектирования, а также обеспечение возможности подключения соответствующего тестирующего оборудования.

Для повышения экономической эффективности разработок следует обеспечить преемственность проектов. В основе этапа *модификации и замены устаревших версий изделия* лежат результаты деятельности по сопровождению выпуска готовых изделий. В ходе сопровождения осуществляется сбор и анализ информации об использовании выпущенной продукции, накопление возможных претензий со стороны пользователей и потребителей, и, при необходимости, принимается решение о целесообразности модификации или замены выпускаемого изделия.

Для проведения перечисленных работ обычно требуется специализированное оборудование, а если оно отсутствует, производится разработка и изготовление как самого такого оборудования, так и методики его применения. Экономически целесообразна унификация средств и методов, используемых на различных этапах жизненного цикла изделий. Так же как ранее удлинение жизненного цикла продукции, содержащей МП, было возможно ввиду простоты корректировок содержания программной памяти, так и теперь значительное повышение "средней продолжительности жизни" изделий обеспечивается простотой замены содержимого памяти конфигурации схем с конфигурируемой структурой.

В большинстве случаев желательно, чтобы оценка эффективности проекта на всех этапах (особенно на начальных) выполнялось в сжатые сроки и с минимальными финансовыми затратами. Подобная задача особенно актуальна в рамках отечественного производства, где не приходится ориентироваться на большую вероятность серийного производства проектируемой



системы, а в большинстве случаев выпуск характеризуется сравнительно небольшими объемами производства. Поэтому для отечественных разработчиков более характерен подход, предусматривающий большие людские и временные трудозатраты и меньшие затраты на исследовательское оборудование. В такой ситуации предпочтительным оказывается применение оборудования, обеспечивающего возможность решить самый широкий спектр задач за счет простых перенастроек. Включение в состав технологического оборудования прототипных плат, построенных на БИС с конфигурируемой структурой, представляется перспективным и относительно дешевым методом достижения сформулированной цели.

Рассмотренные этапы проектирования серьезно тормозят появление на рынке новой продукции, поэтому современные идеи, методы, элементная база и отладочный инструментарий направлены на форсирование работ на этих этапах. Элементы — это схемы с конфигурируемой структурой и "системы на кристалле", уже рассмотренные в *гл. 1*. Идеи — различные варианты расширения возможностей JTAG-интерфейса, граничного сканирования, методы со-проектирования и со-верификации. Отладочный инструментарий — различные виды программного обеспечения, позволяющие оценить интересующие характеристики системы, создать аппаратуру, имеющую упрощенные процедуры ее тестирования, библиотеки и методики, ориентированные на современные методы тестирования, различные типы оценочных и отладочных плат.

Приступая к рассмотрению вопросов упрощения процедуры тестирования аппаратуры, следует сразу четко определить цели, средства и назначение тестовой процедуры. Ключевыми моментами являются:

- средства тестирования проекта (верификация, наблюдаемость);
- средства тестирования создаваемой БИС;
- выпуск пригодных к тестированию приборов и средств их тестирования;
- создание и контроль тестового оборудования;
- генерация тестовых последовательностей.

Здесь следует подчеркнуть, что попытка объединить в одном тестовом оборудовании и одной тестовой процедуре и верификацию проекта, и локализацию неисправностей может привести разработчика к значительным временным потерям.

Существенных выгод, получаемых как на этапе отладки опытных образцов, так и на этапе выпуска серийной продукции, можно достичь за счет ориентации на современные методики и, прежде всего, методы, ориентированные на применение JTAG-интерфейса.

Поэтому и рассмотрим идеи, заложенные в стандарт интерфейса, аппаратуру, связанную с ним, а также методику использования этих средств на раз-

ных этапах разработки. Заметим, что эти принципы широко поддерживаются современными САПР.

## 2.6.1. JTAG-интерфейс

### Предпосылки возникновения

Методы и средства отладки устройств, содержащих современные БИС, претерпели за последние десятилетия существенные изменения. К середине 70-х годов используемые методы визуального, а в лучшем случае, ручного тестирования межсоединений печатных плат, особенно при крупносерийном производстве, полностью себя исчерпали. Широкое распространение получило структурное тестирование печатных плат, выполняемое с привлечением методов технологии "ложе щупов" (в английской литературе — bed of nails). Эта технология базируется на физическом доступе тестирующей аппаратуры к контролируемой плате с помощью механических щупов (подпружиненных контактов), прижимаемых к печатному монтажу. Идею метода можно понять из рис. 2.34.

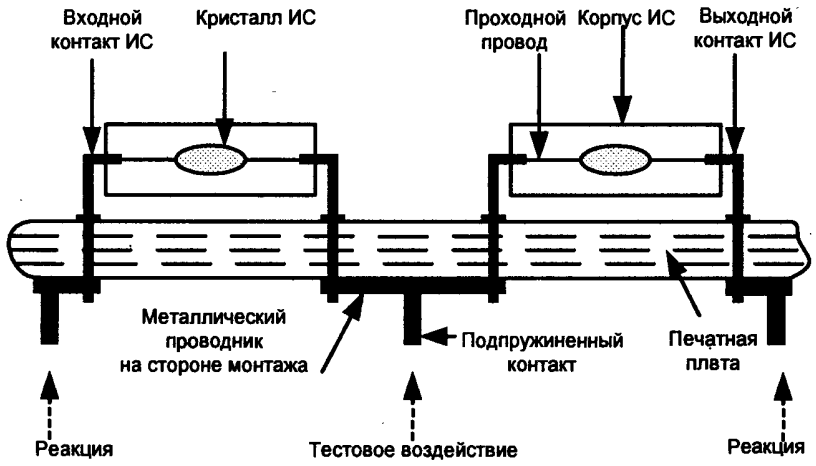


**Рис. 2.34.** Тестирование на основании метода контактирующих щупов

Технология выдвигает специфические требования к конструкции печатных плат. Должен быть обеспечен доступ к местам, где необходимо сформировать тестовые воздействия (стимулы), и местам, где нужно контролировать результаты воздействия тестовых сигналов (реакции). Эти требования легко выполнялись, если элементы установлены только с одной стороны печатной платы (односторонний монтаж), при двухслойном же печатном монтаже доступ к исследуемым контактам монтируемых БИС осуществляется со сто-



роны монтажа через сквозные отверстия в печатной плате. Контролируемая цепочка, как правило, содержит достаточно много разветвлений, что далеко не всегда позволяет этим способом однозначно определить действительное состояние межсоединений. Изготовление тестирующего ложа из контактных щупов может быть экономически оправданным только при промышленном и серийном выпуске печатных плат. Поэтому раньше при невыполнении любого из перечисленных условий приходилось довольствоваться методами ручного просмотра и индивидуального ручного контроля соединений.

Необходимость изменения технологии тестирования печатных плат (причем, как непосредственно после изготовления печатной подложки, так и после установки на нее навесных элементов и их пайки) возникла не только из-за вышеперечисленных неудобств, но из-за того, что:

- резко возросла цена погрешности изготовления печатных плат или монтажа устанавливаемых на них достаточно дорогих БИС (даже в случаях единичного изготовления). При этом произошло увеличение вероятности отказов, связанное с уменьшением физических размеров как контактных площадок для ножек БИС, так и других последствий высокоплотного размещения БИС на печатных подложках;
- сегодня изготовление ориентируется на многослойные печатные платы, используются БИС с технологией SMD (surface mount device) и, как следствие, применяется двусторонняя установка элементов, контактные площадки (при числе контактов современных БИС, превышающих 500) размещаются под корпусом БИС (корпуса типа Ball Grid Array (BGA) и FineLine BGA);
- появилась возможность более широкого применения встраиваемых в БИС специальных тестирующих фрагментов (затраты на добавление тестирующих элементов существенно меньше затрат на настройку). При этом процедура разработки последовательности тестирования и организация процесса тестирования должны учитывать новые возможности современной элементной базы (в том числе специфические возможности микросхем программируемой логики).

Современные системы оказываются исключительно сложными, например, телекоммуникационные и сетевые приложения требуют разработки плат с числом внутренних соединений более 5 тысяч. Такое количество цепей на одной печатной подложке кладет практический предел для применения коммерческого автоматизированного оборудования Automatic Test Equipment (ATE). Методы граничного сканирования, рассматриваемые ниже, позволяют вводить практически неограниченное количество виртуальных тестовых каналов и, тем самым, снимают указанную проблему.

Сегодня каждый серьезный производитель телекоммуникационного, сетевого и другого аналогичного оборудования, ориентирующийся на сверх-

плотный монтаж многоконтактных БИС, должен опираться в своих проектах на методологию граничного сканирования.

Многие плюсы граничного сканирования могут быть использованы даже в тех случаях, когда только одна БИС на проектируемой плате поддерживает граничное сканирование. Однако наибольшая эффективность процедур тестирования, базирующихся на методах граничного сканирования, достигается в случаях, когда большинство приборов на плате объединены в единую сканируемую цепочку.

Важной характеристикой современных приборов (не только МП и ПЛИС, но и других типов БИС) является наличие в их составе и поддержка ими JTAG-интерфейса.

## JTAG-интерфейс и метод граничного сканирования

Исторически интерфейс JTAG (Joint Test Action Group) появился как развитие работ европейской группы (JETAG) исследователей проблем разработки пригодной для тестирования аппаратуры в рамках специальной международной группы, созданной по инициативе фирмы Texas Instrument, для выработки стандарта на производство пригодных для тестирования БИС. Результатом работы этой группы явился принятый в 1990 году стандарт IEEE Std.1149.1 и его усовершенствованная версия стандарт IEEE Std.1149.1a (1993 год).

Предложенный стандарт имел два различных аспекта. Один состоял в разработке протокола и принципов обмена информацией между БИС, соединенными в последовательную цепочку. В дальнейшем для определения этого аспекта стандарта будем пользоваться термином "транспортный механизм". А другой — в специальной (ориентированной на тестирование) организации связи между основными схемами кристалла БИС и ее внешними контактами. Такая организация этой связи позволяет передавать значения сигналов на выходных контактах в транспортный механизм цепочки и наоборот. Это дает возможность использовать границы БИС для задач тестирования их межсоединений без физического доступа к каждому ее выводу. Такой подход получил название метода граничного сканирования (Boundary Scan Testing, BST). Термин "граничное сканирование" представляется более точно соответствующим задаче сканирования состояния границы между основными элементами БИС и оборудованием, расположенным вне БИС, чем используемый иногда термин "периферийное сканирование".

Рассмотрим более детально идеи метода граничного сканирования. На рис. 2.35 схематически показан вариант подачи стимулов и снятия реакций с "виртуальных" щупов, которые находятся на границах кристалла и конструктивно входят в состав БИС.

Рис. 2.36 показывает структурную организацию БИС, поддерживающей метод BST, и позволяет понять основную концепцию граничного сканирования.



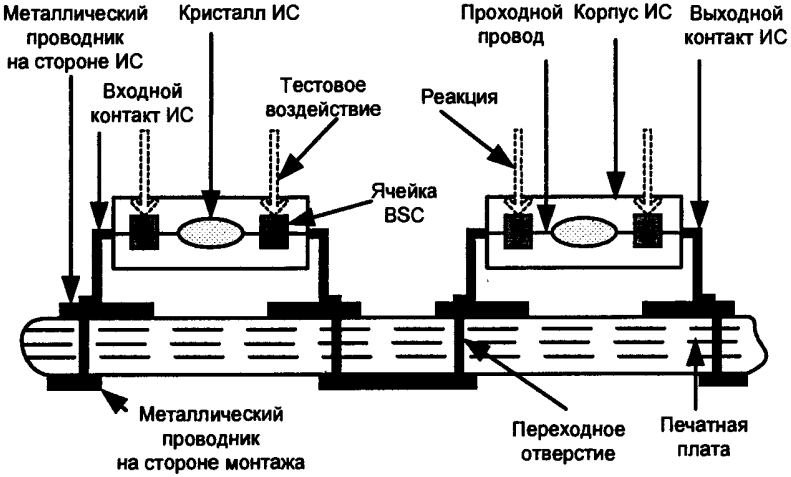


**Рис. 2.35.** Внешний вид печатной платы с БИС, поддерживающими метод граничного сканирования

Ячейки сканирования (Boundary Scan Cells, BSC) размещены между каждым внешним выводом микросхемы и схемами кристалла, образующими собственно БИС. Ячейки, с одной стороны, обеспечивают прием, сохранение или выдачу тестовой информации JTAG-цепочки, а с другой — различные режимы взаимодействия между внешними контактами БИС, запоминающими триггерами ячейки BSC и основными (внутренними) схемами кристалла. Реализация команд тестирования базируется на соответствии длины регистров приема/передачи информации (числа BSC-ячеек) и числа тестируемых контактов БИС, а также на настройке взаимодействия разрядов этих регистров с внутренней структурой БИС или состоянием внешних контактов БИС.

Метод BST задумывался для выполнения следующих тестовых процедур:

- проверки функциональной работоспособности БИС с помощью встроенных в них тестовых цепей;
- проверки качества соединений между контактами различных БИС, смонтированных на печатной плате;
- считывания или установки сигналов на выходных контактах БИС в штатном режиме работы БИС.

Выполнение тестовых процедур предполагает совместную работу трех основных компонентов:

- источника тестовых команд и данных (тестового прибора), которым обычно является программа ПК. Этот же ПК тогда выступает и в качестве анализатора результатов тестирования;

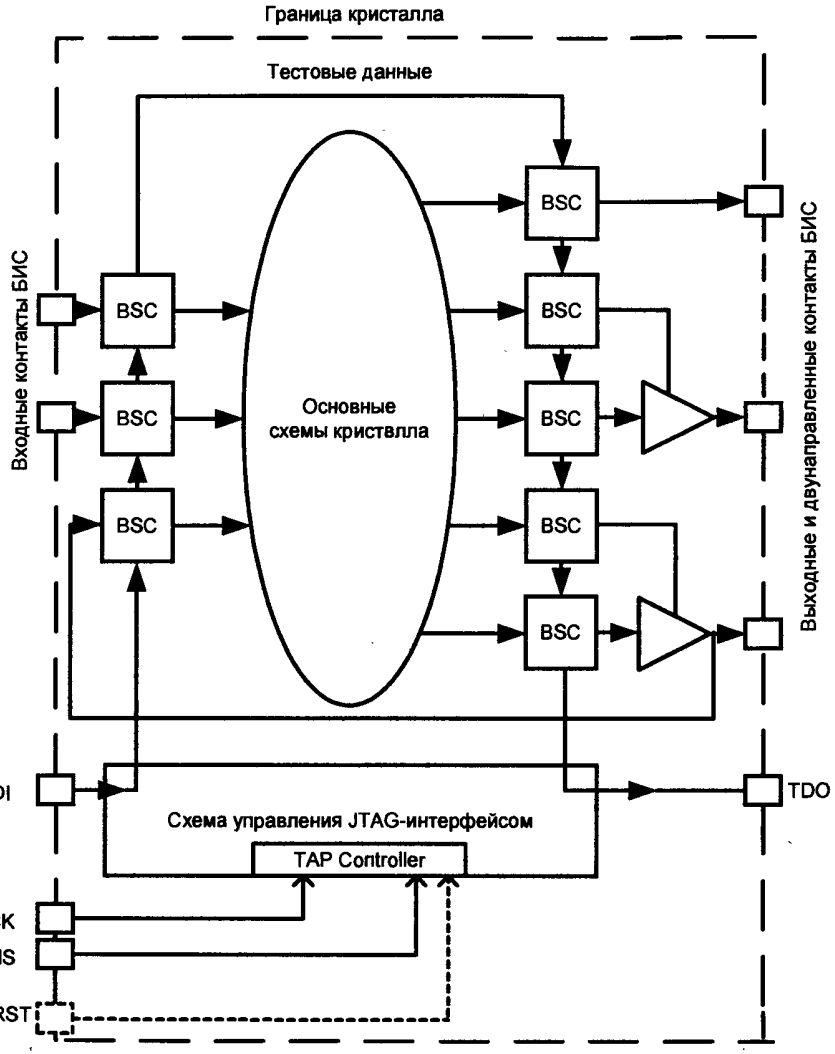


**Рис. 2.36.** Организация БИС, использующей метод граничного сканирования

- механизма информационной связи тестируемых БИС и тестирующего ПК. Транспортный механизм интерфейса JTAG как раз и предусматривает последовательное перемещение тестовых команд и данных от выходных цепей ПК через цепочку последовательно соединенных БИС к входной цепи ПК;
- схем управления JTAG-интерфейсом, встроенных в каждую тестируемую БИС и обеспечивающих соответствующую интерпретацию BSC-ячейками



тестовых команд ПК (реализация собственно метода граничного сканирования). Если предполагается тестирование работоспособности внутренних схем отдельных БИС печатной платы, то в архитектуру БИС должны быть дополнительно встроены аппаратура и специальные тестовые процедуры самотестирования Bilt In Self Test (BIST). Тогда запуск этой процедуры (автоматически при включении питания или/и по подаче внешней команды) позволит судить о работоспособности внутренних схем БИС. Информация об исправности или неисправности БИС будет передаваться по линиям JTAG-интерфейса.

Правильное понимание организации тестовых процедур предполагает рассмотрение взаимодействия всех трех компонентов. Тестовый прибор является не только источником всех тестовых процедур, но и устройством, задающим и синхронизирующим любые действия в цепочке и в тестируемых приборах. Многие особенности организации работы тестового прибора и построения схем управления JTAG-интерфейсом отдельных БИС определяются организацией транспортного механизма. Вместе с тем, реализуемые ПК тестовые эксперименты определяются возможностями, заложенными в структуру отдельных БИС. Поэтому дальнейшее рассмотрение идет от транспортного механизма к анализу тестовых возможностей отдельных БИС и далее к комплексной процедуре тестирования печатной платы.

## Транспортный механизм JTAG-интерфейса

Интерфейс JTAG проектировался для организации информационной связи между произвольным количеством БИС на печатной плате, в приборе и т. д. Основное требование при этом состояло в минимизации числа контактов БИС, необходимого для организации информационного обмена. Обычно используется четыре (реже пять) выделенных для JTAG-интерфейса контакта БИС. Эти контакты образуют так называемый порт доступа (Test Access Port, TAP) контроллера управления портом доступа (TAP Controller).

Контакты порта доступа:

- TCK (Test Clock Input) — синхронизация передачи данных и команд;
- TMS (Test Mode Select) — выбор режима передачи (считывание по переднему фронту CK);
- TDI (Test Data Input) — вход данных и команд (считывание по переднему фронту TCK);
- TDO (Test Data Output) — выход данных, команд или состояния (изменение по заднему фронту TCK);
- TRST (Test ReSeT) — сброс в исходное состояние контроллера (TAP Controller).

Если на печатной плате или в устройстве установлено несколько БИС, поддерживающих JTAG-интерфейс, то эти схемы могут быть объединены в так называемую JTAG-цепочку (рис. 2.37).

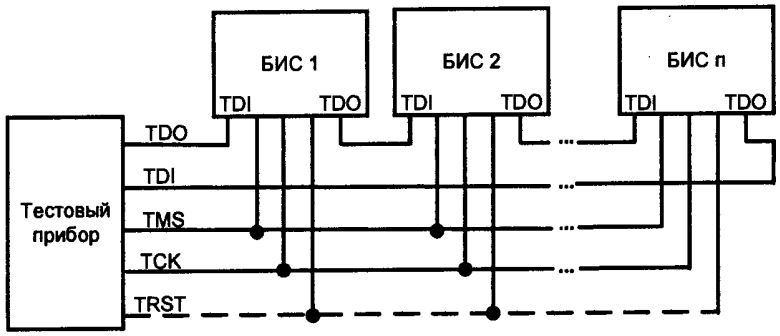


**Рис. 2.37.** Структура JTAG-цепочки

Особенностью исполнения любых действий в цепочке является последовательная форма передачи по ней команд и данных. Устройство управления JTAG-цепочкой (входящее в состав тестирующего прибора) путем управления состояниями всего двух контактов TMS и TCK может устанавливать автоматы TAP-контроллеров всех БИС в любое требуемое состояние — исходное состояние, загрузки заданных команд в БИС цепочки, загрузки требуемых данных в регистры или чтения данных из них. В режимах передачи команд или данных каждому импульсу TCK соответствует сдвиг на один разряд информации в совокупности регистров, расположенных в момент передачи от выходного контакта TDO тестирующего прибора до его же входного контакта TDI. Соединение нескольких БИС в последовательную цепочку заставляет, если необходимо, настроить одну из БИС цепочки на выполнение определенной команды, образовать и вдвинуть в цепочку последовательность битов с длиной, соответствующей набору команд для всех БИС цепочки. Аналогично, если требуется передать данные в определенную БИС цепочки, длина цепочки последовательности битов должна соответствовать совокупной длине сканирующих регистров БИС цепочки.

### Механизм граничного сканирования

Механизм граничного сканирования определяется организацией сканирующих BSC-ячеек. Они обеспечивают реализацию перечисленных ниже режимов.

- Режимы самотестирования БИС, режимы программирования или чтения внутрисхемных ЗУ и т. д. В этих режимах внешние контакты отключаются от внутренних схем БИС, внутрь которой передается информация из BSC-ячеек. Информация соответствует поступившим из JTAG-цепочки командам или данным. Дальнейшая последовательность действий определяется поступившей командой и, в частности, может соответствовать фиксации в ячейках BSC результирующей информации, которую также можно передать в JTAG-цепочку.
- Режимы тестирования соединения БИС между собой. В этом режиме, так же как и в предыдущем случае, внешние контакты отключаются от



внутренних схем БИС. Однако, в отличие от предыдущего режима, информация из ячеек BSC поступает не внутрь БИС, а наружу на ее внешние выходные контакты. Фиксация в ячейках BSC сигналов, поступивших на входные (или двунаправленные) контакты, позволяет судить о наличии или отсутствии реального соединения соответствующей группы контактов. Анализ этой информации осуществляется тестирующим прибором путем ее передачи по JTAG-цепочке.

- Режим тестирования штатной работы БИС. В этом режиме сохраняется требуемое соединение внешних контактов БИС и внутренних схем кристалла. Фиксация в ячейках BSC значений сигналов от всех контактов БИС (в задаваемый из JTAG-цепочки момент времени) и последующая их передача в тестирующий прибор позволяет проектировщику получить интересующую его информацию. Значения внутренних сигналов тестируемой системы становятся известными без организации физического доступа к контактам контролируемых БИС.

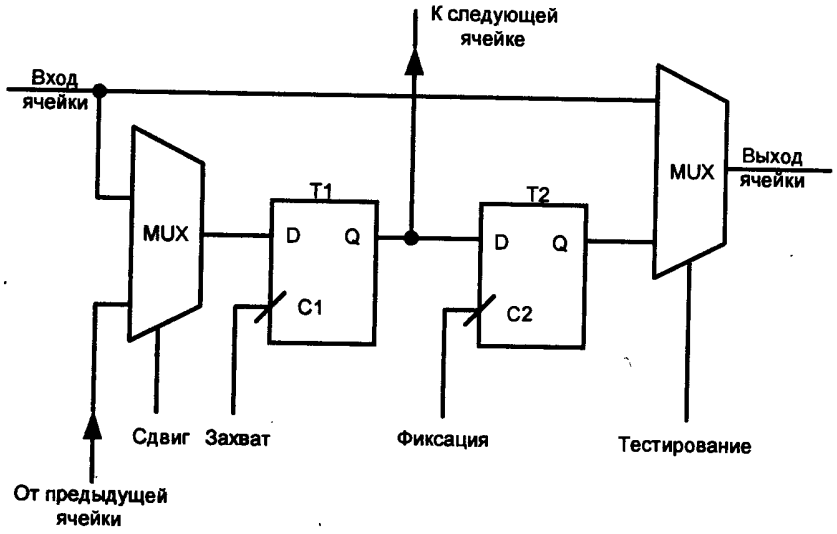


**Рис. 2.38.** Структура сканирующей ячейки

Структурная схема сканирующих ячеек, позволяющих обеспечить реализацию описанных выше режимов, в своей основе имеет вид, приведенный на рис. 2.38. Схема содержит два мультиплексора и два D-триггера. Один из них, T1, является триггером в сдвигающем регистре данных JTAG-цепочки: в состоянии ввода/вывода данных информация сдвигается по цепочке от предыдущей к следующей BSC-ячейке. В этих же триггерах (при другом состоянии входного мультиплексора) может фиксироваться входная информация BSC-ячеек. Триггер T2 является триггером-защелкой, буферизирующим

данные основного сдвигающего регистра. Работа ячейки зависит от режима использования. В рабочем режиме информация со входа ячейки передается на выход, соединяя выходной контакт БИС с внутренними ресурсами БИС, но при этом входная информация ячейки может фиксироваться в триггере T1 сдвигающего регистра. В большинстве тестовых режимов вход и выход ячейки разъединены. Входная информация ячейки может при этом фиксироваться в триггере T1, а выходная информация будет определяться содержимым триггера T2.

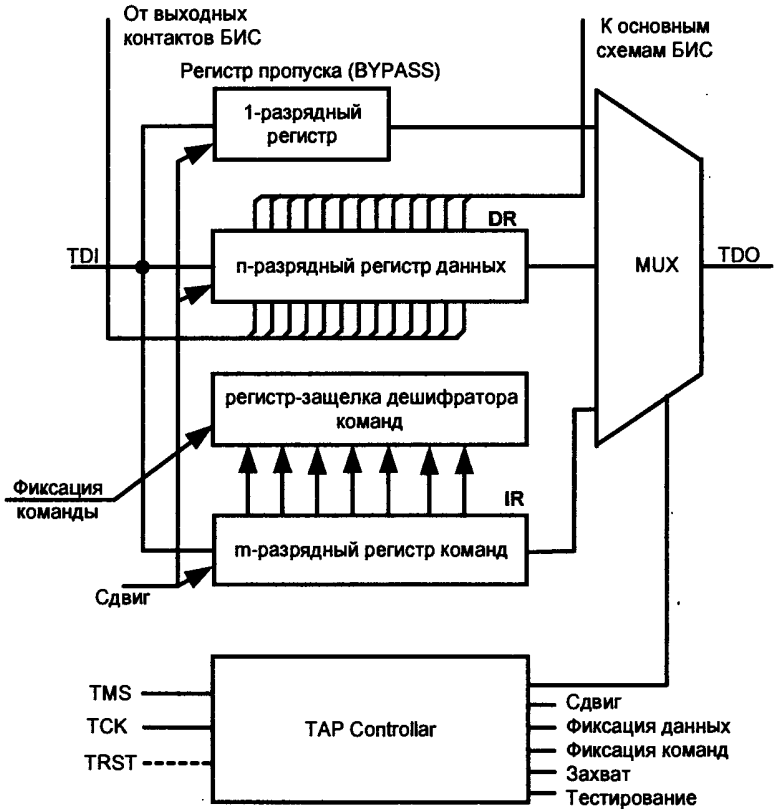


**Рис. 2.39.** Структура устройства управления граничным сканированием

Интерпретацию команд, поступающих из тестирующего прибора, и настройку БИС на выполнение определенной тестовой процедуры осуществляет *устройство управления граничным сканированием* (ГС). По сути, устройство управления является интерфейсным элементом между BSC-ячейками и тестирующим прибором. Структурная схема устройства управления приведена на рис. 2.39. Основные и обязательные элементы устройства управления это три регистра (регистр команд (IR), регистр пропуска (Bypass) и ре-



гистр данных (DR), выходной мультиплексор (MUX) и контроллер управления (TAP Controller).

Основным регистром является регистр данных (часто называемый сканирующим регистром), образованный из последовательно соединенных триггеров T1 (см. рис. 2.38) сканирующих ячеек. Регистр данных служит источником или приемником данных при выполнении в JTAG-цепочках любых команд. С точки зрения устройства управления ГС регистр данных является одним из трех сдвигающих регистров, включаемых между контактом для подачи входной информации (контакт TDI) и контактом для получения выходной информации (контакт TDO).

Организацию различных режимов работы устройства управления ГС обеспечивает дешифратор команд. В цикле записи команд в устройство управления ГС в регистре-защелке дешифратора команд фиксируется следующая команда к исполнению. Код очередной команды вдвигается в сдвигающий регистр — m-разрядный регистр команд (как правило, после включения питания в этот же регистр заносится идентификационный код БИС).

Одноразрядный сдвигающий регистр, регистр пропуска, предназначен для ускорения работы JTAG-интерфейса. В режимах загрузки/выгрузки данных регистр обеспечивает обходной путь для сдвигов многоразрядных данных, не относящихся к данной БИС.

## Команды граничного сканирования

Структура сканирующих ячеек и организация устройства управления граничным сканированием позволяют соотнести с каждой требуемой тестовой процедурой выполнение определенной команды, поступающей из тестирующего прибора. Стандартом предусмотрена обязательность поддержания четырех команд граничного сканирования: EXTEST, SAMPLE/PRELOAD, BYPASS и INTEST. Вместе с тем, допускается введение дополнительных команд, упрощающих работу с JTAG-цепочками или тестируемыми БИС.

Для решения *задач самотестирования* БИС предусмотрены две команды стандарта JTAG — INTEST и RUNBIST. Они предназначены для запуска процедуры автономного тестирования содержимого кристалла.

Команда RUNBIST производит тестирование кристалла без привлечения каких-либо внешних данных. Выполнение команды приводит к запуску встроенной в кристалл тестирующей схемы. Результаты процедуры тестирования заносятся в регистр DR. Данные, зафиксированные в регистре DR после завершения команды, позволяют однозначно определить, исправна БИС или нет.

Команда INTEST проверяет функционирование БИС путем подачи данных от тестирующего прибора. Запуск процедуры самотестирования, основанной на данных зафиксированных в регистрах сканирования DR, вызывает вы-

полнение тестовой процедуры основных схем кристалла. Завершение тестовой процедуры приводит к фиксации в регистре DR результата тестирования, который позволяет однозначно определить, исправна БИС или нет.

Для тестирования внешних соединений БИС на печатной плате предусмотрено использование команды EXTEST. Стандарт оговаривает, что код операции этой команды должен содержать все нули. Исполнение команды предполагает отключение внутренних схем кристалла от внешних контактов (оно осуществляется установкой сигнала "тестирование" на рис. 2.38). Состояния выходных контактов БИС определяются информацией, находящейся в регистрах данных DR БИС. В начале выполнения команды в буферных регистрах DR на фронте сигнала "захват" фиксируется состояние сигналов на внешних контактах БИС, после этого в следующей фазе этой команды зафиксированные данные (при включенном сигнале "сдвиг" на мультиплексоре 1) могут выдвигаться из БИС и замещаться вновь подаваемыми данными, и, наконец, на завершающей фазе команды, обновляется состояние регистров DR (а значит, и состояния выходных контактов БИС) на фронте сигнала "фиксация". Хотя команда EXTEST позволяет производить тестирование межсоединений без привлечения каких-либо дополнительных команд, чаще всего она используется совместно с нижерассматриваемой командой SAMPLE/PRELOAD, выполняющей роль команды, загружающей информацию в регистр данных.

Для контроля поведения БИС в рабочих режимах тестируемой системы стандартом предусмотрено сканирование значений сигналов, присутствующих на внешних контактах БИС. Эту задачу выполняет команда SAMPLE/PRELOAD. Поскольку ее исполнение не предполагает отключение внутренних схем кристалла от внешних контактов (инверсное состояние сигнала "тестирование" на рис. 2.38), то в начале выполнения команды в буферных регистрах DR по фронту сигнала "захват" фиксируется состояние сигналов на внешних контактах БИС. Значения этих сигналов соответствуют штатному режиму работы БИС. Интерпретация содержания такого мгновенного "снимка состояния" сигналов на границе тестируемой БИС зависит от состояния других БИС, связанных с тестируемой. Если все БИС находятся в штатном режиме, то содержание "снимка" отражает протекание рабочих режимов в системе и обычно используется для целей отладки. Возможность задания произвольной конфигурации тестируемой системы (одна группа БИС системы формирует тестовые выходные сигналы, а другая группа БИС находится в рабочем режиме), позволяет организовывать совместное тестирование межсоединений и функционирования внутренних схем БИС.

Команды CLAMP и HIGHZ позволяют расширить возможности тестирования межсоединений или состояния и работоспособности выбираемых пользователем групп БИС в проверяемой системе.

Команда HIGHZ переводит все выходные контакты управляемой БИС в Z-состояние, а устройство управления ГС — в состояние BYPASS. При ми-



нимальности аппаратных затрат ресурсов БИС на реализацию команды ее применение особенно эффективно при тестировании систем с шинной организацией. Для БИС микропроцессоров реализация такого состояния их выходных контактов носит название "In-Circuit emulation" и упрощает использование внешних отладочных эмуляционных средств.

Исполнение команды CLAMP предполагает отключение внутренних схем кристалла от внешних контактов, ее выполнение приводит к установке на выходных контактах БИС значений, соответствующих данным, зафиксированным в регистре DR, и переводу устройства управления ГС в состояние BYPASS.

Команда BYPASS позволяет установить устройство управления JTAG-интерфейсом в состояние сквозного пропуска данных через БИС. Стандартом предусмотрено, что код операции команды должен содержать все единицы. Команда соответствует отключению БИС от процедур тестирования без изъятия ее из JTAG-цепочки.

## Устройство управления JTAG-интерфейса

Основу любого устройства управления JTAG-командами составляет контроллер тестирующего порта доступа (TAP Controller). Поведением контроллера управляет синхронный конечный автомат, состояния и переходы которого определяются значениями управляющего сигнала JTAG-цепочки (сигналом TMS, фиксируемым по переднему фронту тактового сигнала цепочки — TCK). Контроллер формирует выходные сигналы, управляющие поведением BSC-ячеек, — сдвиг, захват, тестирование.

Диаграмма состояний автомата приведена на рис. 2.40. Условие перехода — значение сигнала TMS — показано на диаграмме цифрами 0 или 1. В диаграмме состояний можно выделить четыре базовых фрагмента:

- состояние TEST-LOGIC_RESET (сброс логики тестирования);
- RUN-TEST\IDLE (состояние ожидания или выполнения внутренних тестов);
- режим управления вводом/выводом данных (семь состояний от SELECT-DR-SCAN до UPDATE-DR);
- режим управления вводом команд и чтения состояния (семь состояний от SELECT-IR-SCAN до UPDATE-IR).

Основу режимов управления образуют три состояния: состояние фиксации в выдвигаемых регистрах данных на момент начала фрагмента (состояние CAPTURE-DR или CAPTURE-IR), состояние выдачи из БИС зафиксированной информации и получения новой информации в БИС (состояние SHIFT-DR или SHIFT-IR) и состояние фиксации новой информации (состояние UPDATE-DR или UPDATE-IR) на момент завершения фрагмен-

та. Состояния PAUSE-DR и PAUSE-IR позволяют приостанавливать продвижение информации в цепочке на произвольное количество тактов синхронизации (например, для выполнения каких-либо действий в БИС с внешним тактированием).

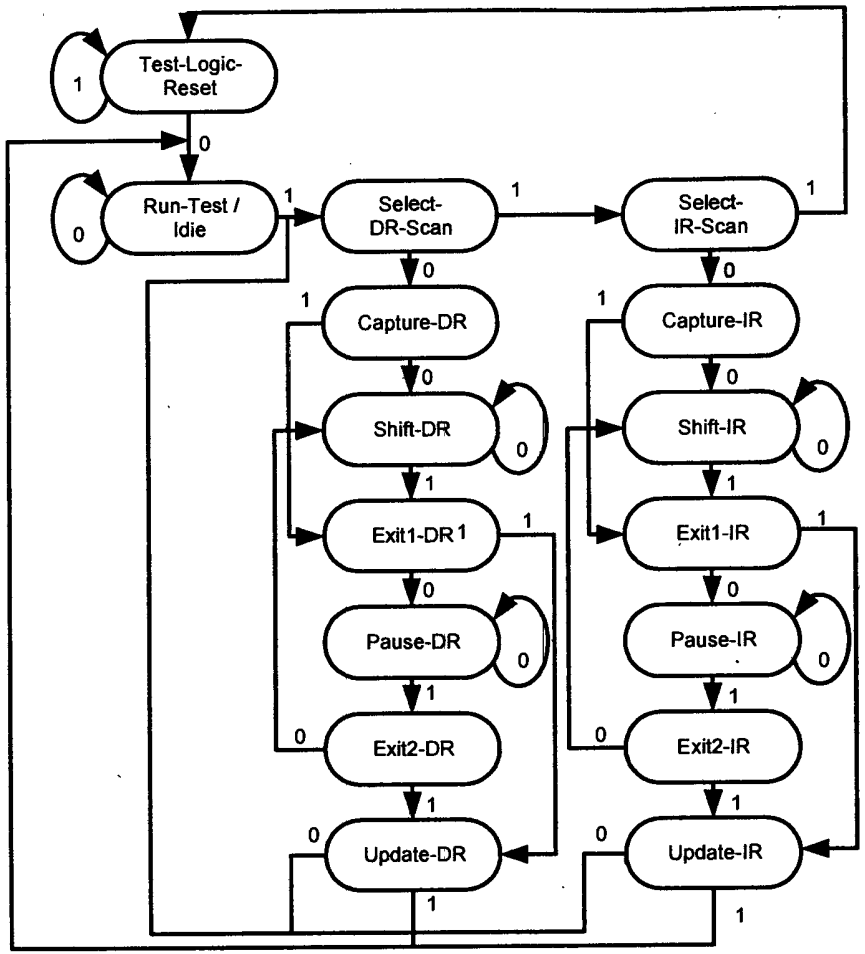


**Рис. 2.40.** Диаграмма состояний автомата контроллера

Состояния EXIT1-DR, EXIT2-DR, EXIT1-IR и EXIT2-IR имеют вспомогательный характер (в том числе для реализации альтернативных переходов), но могут оказаться полезными при реализации устройств управления расширенных вариантов интерфейса JTAG, соответствующих новым вводимым командам.



## Стандарты, связанные с JTAG-интерфейсом

Для широкого использования идей и методов граничного сканирования BST и принципов организации JTAG-интерфейса предложения различных фирм были включены в ряд стандартов. Наличие стандартов позволяет организовать не только обмен информацией между разработчиками БИС, разработчиками тестирующих средств и проектировщиками аппаратуры, но и обеспечить согласованную совместную работу тестируемой аппаратуры и тестирующего программного обеспечения. Стандарты позволяют однообразно описывать пользовательские системы, поддерживающие ГС и JTAG-интерфейс. К настоящему времени наибольшее распространение получили рассматриваемые ниже стандарты.

### Язык BSDL

BSDL — Boundary-Scan Description Language — язык описания организации и архитектуры граничного сканирования в рамках одиночной БИС. Язык входит в состав стандарта IEEE Std.1149.1 и построен как подмножество языка VHDL. Одной из основных задач, возлагаемых на язык, является создание программных средств, автоматизирующих процесс создания тестов. Файлы в формате языка описывают соответствие логических сигналов и физических контактов, набор команд, поддерживаемых TAP контроллером, размер сканирующих JTAG-цепочек, соответствие порядка ячеек ГС в цепочке номеру внешнего контакта и тип управления внешним контактом.

### Язык HSDL

HSDL — Hierarchical Scan Description Language (разработка фирмы Texas Instruments) — язык, ориентированный на описание информационных потоков в группе БИС, объединенных JTAG-интерфейсом. HSDL является расширением языка BSDL, совместим с ним, входит в состав стандарта IEEE Std.1149.1.a и построен как подмножество языка VHDL. Средства языка позволяют определить JTAG-цепочку любого уровня, т. е. последовательность БИС, объединяемых в модуле, в системе или на плате, а также допускают описание динамических или реконфигурируемых цепочек. Язык позволяет описать свойства как отдельной БИС, так и группы БИС внутри тестируемого модуля Unit Under Test (UUT). Модуль — это элемент любого уровня архитектуры выше уровня БИС (может быть платой или системой).

Язык позволяет:

- создавать определяемые пользователем имена для описания команд сканирования и групп данных;
- объединять отдельные биты тестовых последовательностей в легко управляемые подмножества битов с назначаемыми пользователем именами;
- задавать условия разнообразных изменений, определяя свойства цепочек сканирования (статических, динамических или внешних относительно UUT).

Основной файл BSDL приборного уровня может быть аргументом соответствующего предложения файла HSDL и использоваться для организации интерактивной отладки или верификации тестируемого модуля UUT.

#### Язык SVF

Язык Serial Vector Format (SVF) был разработан фирмами Texas Instruments и Teradyne в 1991 году. Широко используется для описания тестовых векторов, посылаемых в приборы цепочки, для получения результирующих данных и маскирования любых данных. Необходимость введения формата диктовалась желанием фирм иметь шаблон данных, легко воспринимаемый широким спектром моделирующих и тестирующих программ и оборудования. Основным требованием являлась возможность использования языка на всех этапах проектирования от проектной верификации до окончательной диагностики.

#### Другие стандарты

Ряд фирм (например, ASSET InterTech, Inc.) разработали форматы макроязыков, существенно упрощающих написание различных программ. Макроязыки позволяют писать программы как для управления различными системами, так и для контроля и тестирования входящего в них оборудования. Основным достоинством макроязыков является способность расширять свои собственные возможности путем введения и определения новых понятий.

Большая избыточность тестовых или загрузочных кодовых последовательностей, циркулирующих в JTAG-цепочках, и, как следствие последовательного характера передачи, значительные временные затраты на обмен, заставили разработчиков искать варианты сжатия информации. И хотя издержками такого подхода является необходимость введения языкового стандарта на метод сжатия, установка аппаратуры, свертывающей и развертывающей последовательности, целесообразность такого подхода оказалась очевидной. Все три составляющих метода сжатия получают все большее распространение. Появился стандарт на язык JAM, структуры и конструкции свертывающей и развертывающей аппаратуры типа JAM Player.

## 2.6.2. Проблемы и методология создания пригодной для тестирования аппаратуры

Создание аппаратуры, которую можно было бы быстро продвинуть на рынок, — это задача не только быстро сделать проект, но и обеспечить быстрое производство этой аппаратуры. Кроме того, требуется сделать эту аппаратуру технологичной, а не только правильно работающей. Как обычно, при построении большого и сложного здания здесь важны все компоненты:



строительные элементы, инструментарий и разумная методология строительства, опирающаяся на стандартизацию всех составляющих.

Строительные элементы тестирования могут быть отнесены к одной из двух групп, отличающихся как средствами, так и моментом применения. Первая группа — это схемы тестирования, которые входят в состав конечной продукции (БИС, печатной платы, системы и т. д.) и должны быть включены в систему до ее перехода в стадию реализации, и внешние относительно конечной продукции средства (программные пакеты и САПР), поддерживающие процедуру создания и включения тестирующих фрагментов. Вторая группа — это аппаратура и программное обеспечение, входящие в состав тестирующего оборудования, которое используется для тестирования готовой продукции. Очевидно, что эффективность работы аппаратуры второй группы оказывается зависящей не только от оборудования, используемого на этом этапе работ, но и от наличия схем тестирования, включенных в продукцию еще на этапе проектирования.

Перед разработкой пригодной к тестированию аппаратуры проектировщик должен, прежде всего, определиться с применяемым инструментарием и методикой его использования. Так же как и при решении других проблем проектирования, допустимыми являются два подхода — ориентация на инструментарий автономной поддержки рассматриваемого этапа проектирования или комплексный подход, включающий работы всех этапов в один общий процесс проектирования. При очевидной целесообразности использования последнего варианта, необходимо учитывать сложности, ему сопутствующие (рост размерности решаемых задач, дороговизну требуемых САПР, резкое возрастание сложности реализации). Вследствие этих причин в настоящий момент широкого распространения комплексный подход к проблеме создания пригодной к тестированию аппаратуры еще не получил, и работы находятся на своей начальной стадии. Чуть ниже мы более подробно остановимся на основных особенностях реализации этого направления. Поскольку при комплексном подходе интегрируются решения, используемые при индивидуальном подходе, далее последовательно рассмотрим решения, характерные для отдельных этапов проектирования.

Если проектировщик аппаратуры планирует опираться при выпуске конечного изделия (печатной платы или даже отдельной БИС) на возможность использования тестовых процедур граничного сканирования, то начальным и важнейшим для него действием является поиск средств, автоматизирующих выполнение соответствующих этапов проектирования. Эффективность процедуры тестирования в значительной мере будет определяться используемым программным обеспечением. Целый ряд фирм предлагает использовать для автоматизации различных этапов проектирования пригодной к тестированию аппаратуры разнообразные программные пакеты. Разобраться в наборе этих средств уже не самая простая задача. Поэтому в начале рассмотрим целесообразную организацию процедуры проектирования, а затем остановимся на возможностях, предоставляемыми САПР ведущих фирм.

## Работы, выполняемые проектировщиком при создании проекта

Стремление разработчиков использовать в своих проектах возможности, предоставляемые JTAG-интерфейсом (и граничным сканированием, в частности), должно начинаться с первых шагов проектирования.

Если тестирование проекта предполагается строить на базе JTAG-интерфейса, то, прежде всего, необходим *выбор элементов*, обеспечивающих транспортный механизм интерфейса.

Эти элементы должны иметь возможность подключаться к соответствующему оборудованию. Транспортная система может быть организована по разному — от простой цепочки (одиночная БИС) до сложных адресуемых, перестраиваемых и реконфигурируемых цепочек. Существует достаточно большое количество фирм, которые предоставляют широкую номенклатуру БИС данного целевого направления. Для более удобной организации сканирования может быть использована, например, специальная ИС адресуемого скан-порта SN74LVT8980, предназначенная для установки на отдельную плату, а для системного уровня может быть использована ИС контроллера тестовой шины SN74LVT8996.

После того, как проектировщик определил средства и способы подачи и получения JTAG-последовательностей, он переходит к решению вопросов встраивания средств граничного сканирования или самотестирования в систему (это может быть отдельная БИС, печатная плата или набор таких плат).

Если *тестирование межсоединений* в готовой продукции планируется построить на основе идей граничного сканирования, то в системе (на плате) должен быть установлен хотя бы один элемент с возможностью ГС. Интегральные схемы, поддерживающие граничное сканирование, могут быть любого типа — стандартными, полузаказными или полностью заказными.

Практически любой тип стандартной продукции имеет альтернативные варианты реализации, которые, в отличие от стандартных, включают средства ГС. Подавляющее число выпускаемых в настоящее время БИС ПЛ поддерживает методы ГС либо в варианте "жестко" встроенной реализации (HW-core), либо в форме стандартных фрагментов, встраиваемых в ходе компиляции проекта (SW-core). Для реализации методов ГС в состав стандартных библиотек фирм, выпускающих схемы класса ASIC по технологии стандартных ячеек, включаются контроллеры JTAG-интерфейса и регистры ГС. Если разрабатываются полностью заказные БИС или схемы ASIC, изготавливаемые по технологии БМК, то появляется необходимость добавления к описанию проекта средств тестирования. Добавляемые фрагменты, как правило, выбираются проектировщиком со способом описания, принятым в основном проекте. Распространение получили описания на уровне регистровых передач и на уровне языков высокого уровня.



аличие во многих приложениях фрагментов, для которых затруднена или кономически не оправдана реализация тестирования на принципах ГС, заавляет разработчиков анализировать возможность архитектурного пороения системы, позволяющего охватывать тестированием и те фрагменты стемы, в которых нет элементов, поддерживающих методы ГС. В настояее время интенсивно разрабатываются формальные методы построения истем с возможностью контроля фрагментов, не охваченных ГС.

ругим направлением создания надежной продукции (не отменяющим, а полняющим методы граничного сканирования, и также опирающимся транспортные механизмы интерфейса JTAG) является направление, свянное с таким проектированием БИС, при котором в них оказываются *троенными средства самотестирования* Built-In-Self-Test (BIST). Достаточ- большое число фирм занимается этим направлением, и они предлагают едства, позволяющие автоматизировать создание как аппаратуры, так и мих тестирующих последовательностей. Наибольших успехов при этом к и следовало ожидать) удалось достичь в разработке средств, тестирующих работоспособность регулярных структур, например блоков ОЗУ.

ирмы, разрабатывающие тестовые фрагменты Hardcore или Softcore (в засимости от технологии изготовления проектируемой БИС), не только педают их в форме интеллектуальной собственности IP, но и одновременно этим поставляют оборудование и программное обеспечение, существенно коряющие процесс проектирования. САПР таких фирм, как Analogy, dence, Intusoft, Mentor Graphics и Synopsys позволяют не только встраить эти фрагменты в проекты, но и учитывают их в синтезирующих и моирующих пакетах. В результате разработчик имеет возможность выполть полный цикл проектных верификационных работ, убеждающих его, добавление тестовых фрагментов не привело к искажению штатной раты проекта.

## аботы, выполняемые и тестировании готовой продукции

осле получения готовой продукции в форме БИС или печатной платы разботчик оказывается перед необходимостью выполнения настроечных и падочных работ, для ускорения и упрощения которых и вводились средст, ориентированные на использование методов и средств граничного скарования и JTAG-интерфейса.

роцедура отладки и привлекаемое для этого оборудование, прежде всего, пределяются видом тестируемой продукции. Необходимо разделять работы, ыполняемые на этапе отладки лабораторных макетов и серийных образцов родукции (хотя и то, и другое базируется на возможностях JTAG-инерфейса). В простейших случаях, для проведения экспериментов с лабораорными образцами или первыми образцами опытных партий процедура

тестирования может опираться на обычный ПК, снабженный соответствующим программным обеспечением и имеющий загрузочный кабель, подключенный, например, к его параллельному порту. Промышленные установки, предназначенные для тестирования серийной продукции, представляют собой достаточно большие и дорогие агрегаты, а целесообразность их применения диктуется большой тиражностью выпускаемой продукции, что, в свою очередь, требует сокращения времени, необходимого для тестирования оборудования.

Рассмотрим сценарий отладки лабораторного макета. Сложность проведения экспериментальных работ связана с тем, что отклонение поведения макета от ожидаемого может быть связано как с дефектом его изготовления, так и с ошибкой самого проекта. Более того, в ряде случаев может оказаться дефектным само тестирующее оборудование. Поэтому процедура выяснения причины отклонения оказывается более сложной, чем процедура, используемая при анализе серийной продукции, когда причина отклонений обусловлена только технологическими дефектами. Для проведения экспериментов проектировщик может пользоваться простейшими тестирующими программами, опирающимися на ресурсы граничного сканирования и JTAG-интерфейса.

На первом шаге работ проектировщик должен убедиться в *работоспособности* элементов, образующих JTAG-цепочку. Тестовые программы автоматически сравнивают идентификационные коды ИС, полученные из цепочки, с кодами, взятыми из справочных данных фирм-производителей ИС (фирмы поставляют такие файлы с расширением bsd). Отсутствие данных из цепочки должно заставить проектировщика использовать программы, локализующие причину и место нарушения, а несоответствие данных требует анализа и адекватных действий.

На следующем шаге могут использоваться (если они включены в состав ИС) *встроенные средства самотестирования*. Отладочное оборудование при этом должно через JTAG-интерфейс инициировать процедуры самопроверки в проверяемых ИС. Информация, полученная из JTAG-интерфейса, после завершения процедуры самотестирования позволяет убедиться в работоспособности схем, входящих в отлаживаемую систему.

После проверки работоспособности отдельных элементов печатной платы можно переходить к *тестированию соединений* этих элементов между собой. Если все элементы цепочки работают исправно, и в цепочке имеются ИС, поддерживающие команды и методы граничного сканирования, то существующие программы могут автоматически проверять исправность межсоединений и определять места нарушения контактов или наличия паразитных соединений. При хорошей организации тестовых последовательностей возможна локализация существующей неисправности с точностью до отдельной сканирующей ячейки. Проверяться могут не только соединения, расположенные между сканируемыми контактами, но и соединения, разделенные



достаточно простыми логическими цепями, не поддерживающими метод ГС. В последнем случае одновременно проверяются соединения и функционирование этих цепей.

Убедившись в отсутствии ошибок межсоединений, проектировщик может переходить к тестированию оборудования, не поддерживающего ГС. Проверка работоспособности такого оборудования может выполняться на ресурсах JTAG-интерфейса, если проверяемые фрагменты своими входными и выходными цепями соединены с контактами сканируемых ИС. Эффективность тестирования определяется всей совокупностью задействованных элементов. Структура тестируемого фрагмента, места его подключения к источникам тестирующих сигналов и места снятия с него контрольных сигналов, организация теста — все эти вопросы должны решаться проектировщиком. Не в последнюю очередь рациональный выбор этих составляющих определяется возможностями их автоматизации.

Различные фирмы предлагают здесь огромный спектр услуг, и в частности комплексы аппаратуры, программного и методического обеспечения, отличающиеся объемом и уровнем решения проблем, возникающих при автоматизации различных этапов проектирования пригодной к тестированию аппаратуры. Наибольшее распространение получили пакеты, автоматически генерирующие тестовые последовательности и шаблоны правильных реакций. Продукция различных фирм отличается и допустимой сложностью тестируемых объектов, и оборудованием, используемым для организации тестирования. Сложнее дело с пакетами, автоматизирующими процесс синтеза такой структуры фрагмента, которая может быть эффективно протестирована.

Хотя построение аппаратуры, допускающей проведение тестирования в процессе ее эксплуатации, несомненно, является перспективным, удорожание при этом самой аппаратуры и процедуры ее проектирования заставляет рассматривать и альтернативные варианты. Если аппаратура отлаживаемой системы дает возможность изменения или неоднократной загрузки информации в программируемые фрагменты системы, то в этих фрагментах на этапе проверки работоспособности оборудования может размещаться тестовая информация. Раньше тестовые примеры размещались в памяти команд микропроцессорных систем, и это существенно упрощало проведение тестирования. Появление в современных системах приборов с программируемой структурой расширяет этот подход. Занесение в память конфигурации ПЛИС структуры тестового оборудования может производиться в любой требуемый момент. Поэтому конфигурация, соответствующая тестовым экспериментам, может применяться для отладки не только опытных образцов, но и серийной продукции. Тестирование опытных и серийных образцов, естественно, различаются полнотой, длительностью проведения и т. д.

После того, как разработчик убедился в правильности монтажных соединений и работоспособности отдельных используемых ИС, в микросхемы памяти программируемых и конфигурируемых приборов может заноситься рабочая информация. Наступает *этап проверки правильности функционирова-*

*ния проекта* в целом. Методика проведения работ на этом этапе в основных чертах совпадает с методикой проверки функционирования отдельных фрагментов.

Однако, как бы тщательно ни выполнялось проектирование, всегда остается небольшое количество (не более 5%) ошибок, нахождение которых составляет основную трудность отладки. Чаще всего это соответствует случаю, когда стабильное и "правильное" (с точки зрения разработчика) поведение проекта лишь изредка нарушается "неверным" поведением. Обычно в таких ситуациях сложнее всего обнаружить причину нарушения и выяснить условия ее возникновения. Потенциальные возможности, заложенные в стандарт граничного сканирования, и правильная отладочная тактика проектировщика могут помочь обнаружить любую, даже очень редко возникающую, ситуацию. Здесь находят применения следующие сценарии отладки, опирающиеся на возможности аппаратуры с JTAG-интерфейсом.

1. Получение мгновенных "снимков" со всех контактов всех БИС для выбранного момента времени. Информация, находящаяся на входах сканирующих ячеек, в любой момент времени может быть зафиксирована в сканирующих регистрах, а затем извлечена из системы. Все это может производиться без прерывания штатной работы системы. Помимо отладки, базирующейся на анализе граничных сигналов, все большее распространение в современных ПЛИС и SOPC получают методы внутрикристальной отладки. Существуют два подхода. Первый заключается во встраивании в проект контролирующих цепей, подключенных к цепочке регистров, функционирование которых определяется командами автомата, совпадающего по поведению и интерфейсу с TAP-контроллером. Все эти цепи реализуются за счет использования общецелевых ресурсов ПЛИС и в окончательной конструкции устройства могут быть убраны. При втором подходе отладочные регистры уже исходно включены в состав штатных ресурсов кристалла. Чтобы минимизировать влияние контролирующих цепей на поведение основных фрагментов проекта, в реальных схемах подключение осуществляется только к глобальным и длинным линиям ПЛИС. Наличие ограничений и фиксация мест подключения контрольных точек приводят к тому, что поведение проекта может наблюдаться только в определенных местах. Применение ПЛИС с подобной организацией оказывается эффективным в тех случаях, когда отладочные средства интегрированы с САПР, занятой монтированием проекта в БИС. В этой ситуации САПР позволяет контролировать состояние сигналов, имена которых были заданы при спецификации проекта. Основным недостатком всех подходов является, как правило, достаточно длинная во времени процедура извлечения данных из цепочки, поэтому частота получения снимков оказывается не очень высокой. Другой задачей, существенной для обоих рассматриваемых подходов, является синхронизация моментов времени, связанных с фиксацией данных в сканирующих регистрах и с изменением контролируемых данных. Длина



сканирующих регистров делает проблематичной отладку в реальном масштабе времени. Наиболее сложной задачей здесь является реализация цепей, определяющих момент времени, интересующий проектировщика, и формирующих сигнал, соответствующий этому моменту.

Симуляция любых комбинаций сигналов на сканируемых контактах, причем сигналы могут быть направлены как наружу, так и внутрь БИС. Создаваемая комбинация сигналов (во времени и в пространстве) может быть и такой, какая не бывает в штатном режиме работы системы (или, по крайней мере, предполагается, что в системе реально такой комбинации не может быть). Подобная возможность позволяет экспериментально определить реакцию системы на "нештатную" комбинацию сигналов.

Если сигналы тактового генератора системы в состоянии управляться или замещаться сигналами, формируемыми тестирующим оборудованием, а проектировщика интересует в поведении отлаживаемой системы только функциональная последовательность смены состояний, то частота подачи тестирующей последовательности, передаваемой по JTAG-интерфейсу, может быть соотнесена со скоростью съема данных из регистра граничного сканирования. Таким образом формируется полностью контролируемая и управляемая тестовая процедура.

овершенно очевидно, что эффективность рассмотренных выше методик стирования системы будет сильно зависеть от конструктивной реализации роекта. В некоторых случаях только интерактивное участие проектировщика в создании тестирующих последовательностей обеспечивает увеличение олноты тестирования. Например, проектировщик может ввести соединение татно неиспользуемого контакта БИС ПЛ с электрической цепью, которая о того не могла быть протестирована. Наличие подобного соединения позволяет при подаче тестовой последовательности на фиктивное соединение обиться определенности тестирования.

осле того, как проект на тестовых примерах заработал правильно и стабильно, можно переходить к подготовке тестов для рабочих испытаний аппаратуры. Как уже отмечалось выше, требования к программам тестирования аппаратуры, в функциональной работоспособности которой практически отсутствуют сомнения, отличаются от требований, предъявляемым программам, тестирующим серийную продукцию. Соответственно, отличаются аппаратные средства и САПР, привлекаемые для подготовки и проведения экспериментов этого этапа работ.

## 2.6.3. Обзор средств поддержки JTAG-интерфейса ведущими фирмами

Поддержка, развитие и внедрение идей JTAG-интерфейса осуществляется различными фирмами практически по всем тем направлениям, которые были рассмотрены выше.

Прежде всего, аппаратная поддержка JTAG-интерфейса — средства, встраиваемые в выпускаемую продукцию и включающие: транспортные средства, расширяющие возможности JTAG-интерфейса, интеллектуальные средства, расширяющие возможности ГС, средства BIST, встраиваемые в СпИС и ИС.

Затем средства, образующие саму тестирующую аппаратуру.

И, наконец, программные средства, подготавливающие тестовые последовательности, программные и аппаратные средства, организующие тестирование на уровне БИС, печатных плат или систем.

## Аппаратная поддержка JTAG-интерфейса

Наибольшие усилия в части выпуска стандартной аппаратуры, обеспечивающей работу с JTAG-интерфейсом, прикладываются фирмами Texas Instrument (**www.ti.com**) и JTAG Technologies B.V. (**www.jtag.com**). Только TI выпускает более 40 типов коммерческой продукции этого направления. В том числе ИС, отличающиеся от стандартных только наличием встроенных средств JTAG-интерфейса. Например, вместо обычных магистральных приемопередатчиков (типа АП4, АП5, АП6) могут быть выбраны приемопередатчики фирмы Texas Instrument, поддерживающие режимы граничного сканирования. Выбор предлагаемых ИС достаточно широк — могут быть выбраны октальные приборы типа BCT и ABT или ИС, предусматривающие работу с шинами (18/20 бит) типа ABT/ABTH для питающего напряжения 5 В или LVTH для напряжения 3,3 В.

Для поддержки расширенных вариантов транспортного механизма JTAG-интерфейса фирма предлагает использовать ИС адресуемого скан-порта Addressable Scan Port (ASP) типа SN74LVT8996, который обычно устанавливается на отдельной плате, а для системного уровня встраивания — контроллер тестовой шины embedded Test Bus Controller (eTBC) типа SN74LVT8980.

Фирмы-производители ПЛИС в большинстве выпускаемых схем предусматривают использование JTAG-интерфейса. Однако полнота поддержки JTAG-интерфейса (число и типы поддерживаемых команд) коррелирована со сложностью ПЛИС. Иногда приходится включать в проект схему PLD из семейства БИС с превышением логической мощности по сравнению с требуемой только из-за того, что она поддерживает необходимые команды граничного сканирования. У некоторых схем фирмы Xilinx элементы JTAG-интерфейса в явном виде отсутствуют и являются опционными (при их встраивании в структуру кристалла затрачиваются ресурсы PLD).

Направление деятельности ряда фирм — добавление фрагментов граничного сканирования в базовую структуру проекта. Основные фрагменты — контроллер тестового доступа и ячейки сканирующих регистров. Подобные фрагменты могут встраиваться в подготовленный проект не только для



ЛИС, но и при изготовлении проектов по заказной или полузаказной технологии.

ак, например, программный пакет фирмы Synopsys (САПР носит название SD Compiler) автоматизирует процедуру добавления фрагментов граничного сканирования к основному проекту, сопровождая ее выпуском тестов я последующего тестирования конечной продукции. САПР BSD Compiler зволяет автоматизировать процессы синтеза и верификации логики граичного сканирования в ИС, выпускаемых по заказной или полузаказной хнологии. Для синтеза BSD Compiler использует описание пользователього проекта на уровне регистровых передач (RTL) и добавляет требуемые AG-компоненты из фирменной библиотеки (DesignWare). Затем САПР томатически проверяет комплексный проект на совместимость со станартом IEEE 1149.1. И на последнем этапе автоматически создает файлы на ыке BSDL (Boundary Scan Description Language), предназначенные для стирования на уровне печатных плат и для генерации функциональных параметрических векторов для производственного тестирования.

ирма Alternative System Concepts Inc., ASC (**www.ascinc.com**) предлагает пает VBIT (VHDL Built Test), который позволяет проектировщику автоматиески вставлять средства JTAG-интерфейса в проекты высокой плотности, аписанные на языке VHDL, еще до начала их логического синтеза.

ругой разновидностью аппаратных добавлений, увеличивающих возможноти тестирования БИС, является добавление к базовому проекту средств строенного самоконтроля (BIST). Работы по этому направлению характерны для таких фирм, как Diagonal Systems, Inc. (**www.diagonal.com**), Fluence echnology (**www.fluence.com**), Genesys Testware, InterHDL (**www.interhdl.com**), ogicVision (**www.logicvision.com**) и Syntest. Фирмы выпускают средства, автоматизирующие различные этапы проектирования. Проекты, предназначенные для добавления средств BIST, могут использовать описание своей труктуры на одном из стандартных языков описания аппаратуры, таких как erilog, VHDL, EDIF или Spice. Продукция перечисленных фирм обеспечивает добавление BIST-функций, автоматическую генерацию тестовых векторов или моделирование отказов.

дним из представителей, поддерживающих это направление, является фирма LogicVision. Она предлагает широкий набор блоков IP, соответствующих различным видам встраиваемой тестовой продукции. Тестовая продукция поддерживает реализацию большого числа типичных приложений и содержит в том числе такие средства, как:

- Memory BIST — встраиваемый IP-блок для встраиваемых динамических и статических ОЗУ;
- IC Memory BIST — законченное IP BIST-решение, представляющее собой контроллер и тестирующее обрамление элементов ОЗУ, которое ав-

томатически генерируется как синтезируемый RTL-объект для проектов на языках VHDL или Verilog;

- PLL BIST — встраиваемые элементы тестовых цепей для цепей фазового управления Phase-Locked Loops;
- Logic BIST — встраиваемые фрагменты комбинационной логики.

Другим видом продукции этой фирмы является пакет Chip Test Assemble, который создает и собирает всю тестовую инфраструктуру кристалла в соответствии со стандартом IEEE 1149.1, включая контроллер и регистр граничного сканирования.

## Программная поддержка тестирования

Естественно, что все описанные выше возможности реализуются в том случае, когда управление БИС, входящими в состав JTAG-цепочки, выполняется достаточно интеллектуальным контроллером, обеспечивающим подачу предварительно разработанных тестовых последовательностей и контроль последовательностей, получаемых в результате тестирования. Тестовые последовательности, как правило, создаются специальным программным обеспечением. Программное обеспечение, автоматизирующее процесс подготовки тестовых последовательностей, опирается на систему межсоединений, реализованных на плате и на тестирующие возможности БИС, расположенных на этой плате.

Целый ряд фирм разрабатывает и поддерживает программные пакеты, автоматизирующие указанные выше тестовые процедуры. Предлагаются аппаратно-программные комплексы, выполняющие загрузку тестовых последовательностей, получение контролирующих последовательностей и анализ полученных результатов. Среди этих фирм можно указать на фирмы ASSET Intertech (**www.asset-intertech.com**), Corelis (**www.corelis.com**), GenRad, Inc. (**www.genrad.com**), Flynn Systems Comporation (**www.flynn.com**), Hewlett-Packard.

Фирма JTAG Technologies B.V. предлагает, например, пакет JTAGLINK, который обеспечивает прямое соединение между схемными редакторами типовых САПР и средствами фирмы, предназначенными для автоматической генерации тестов Automatic Test Pattern Generation (ATPG). Пакет обеспечивает стыковку с большинством САПР — такими как Mentor Graphics, Cadence, Viewlogic, OrCAD, Accel, P-CAD, Synopsis, Sunario и др.

Большинство фирм предлагает более простые (и, соответственно, более дешевые) пакеты. Например, фирма ACUGEN Software, Inc. (**www.acugen.com**) выпускает программное обеспечение, выполняющее автоматическую генерацию тестов Automatic Test Generation (ATG) для приборов программируемой логики (PLD, EPLD, FPGA и небольших GATE Array), а также приборов, поддерживающих язык описания граничного сканирования BSDL.



Примером фирмы, предлагающей комплекс технических средств, является фирма Teradyne (**www.teradyne.com**). Фирма разработала виртуальный тестовый набор, включающий пять средств: Virtual Interconnect Test, Virtual Component Cluster Test, Boundary Functional Test, Boundary-Scan Intelligent Diagnostics и Access Analyzer. Первые четыре модуля, генерирующих тесты, дают проектировщику свободу в выборе тестовой стратегии, базирующуюся на его тестовых объектах и с различной степенью тестового доступа. Диагностический модуль быстро изолирует тестовую неисправность, идентифицирует отказавший прибор, тип неисправности и вовлеченные цепи. Анализатор доступа обычно используется перед разводкой для помощи проектировщику разделить сканируемые и не сканируемые дефекты.

## Комплексная организация поддержки тестирования

Как уже отмечалось, наибольших успехов при создании пригодных к тестированию проектов Design-For-Testability (DFT) можно достичь, только соблюдая единство подхода ко всем составляющим процесса проектирования на всех этапах создания продукции.

Естественно, что такой подход могут предложить только крупные фирмы, имеющие богатый опыт разработки и выпуска САПР. Примером фирм, стоящих на позициях комплексного решения задачи проектирования пригодной к тестированию аппаратуры, являются Mentor Graphics Corporation (**www.mentor.com**) и Synopsys (**www.synopsys.com**).

Фирма Synopsys при производстве пригодных к тестированию БИС предлагает автоматизировать процесс проектирования, опираясь на последовательную работу разработанных ею трех пакетов: DFT Compiler, BSD Compiler и TetraMAX ATPG. Первый пакет поддерживает создание целевого проекта, содержащего требуемые тестовые вставки, второй пакет синтезирует устройство, содержащее целевую систему, фрагменты, тестирующие ее работу и, наконец, фрагменты, поддерживающие граничное сканирование, третий пакет, опираясь на результаты синтеза собранного проекта, автоматически генерирует тестовые последовательности Automatic-test-pattern-generation.

До настоящего времени синтез тестов для проектирования пригодных к тестированию проектов выполнялся путем дополнения тестовых фрагментов к основной части ASIC-проекта, и это приводило к тому, что улучшение свойств пригодности к тестированию требовало возврата к началу проектной процедуры. На рис. 2.41 показана традиционная методика добавления тестовых фрагментов к базовому проекту. Чтобы успешно решать все проблемы времени, площади и рассеиваемой мощности требуется одновременное и совместное использование новейших средств САПР для создания законченных DFT-проектов.

Для успешности этой процедуры необходимо, чтобы проектировщики RTL-описания проекта и проектировщики DFT работали в согласии и с единым

взглядом и видением проекта, используя объединенные проектные средства и потоки. Конечно, это должно предполагать, что средства DFT не будут оказывать негативного воздействия на критические временные моменты основной обработки данных в проекте.

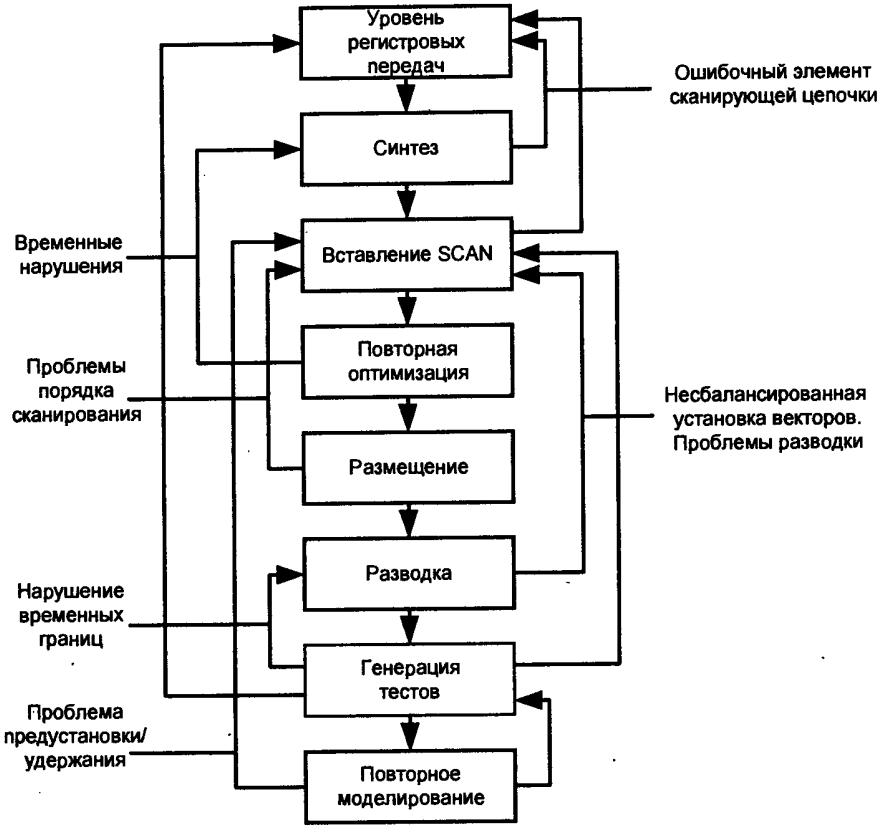


**Рис. 2.41.** Традиционная методика проектирования пригодной к тестированию аппаратуры

Требуемая технология должна удовлетворять ряду важнейших требований:

- тестовая САПР должна начинать работу с уровня регистровых передач и подключаться к основному описанию для совместного синтеза;
- при разводке синтезируемые для теста фрагменты должны хорошо интегрироваться с физическими проектными средствами;
- тестовые фрагменты должны позволять синтезировать DFT с полной оптимизацией по введенным ограничениям.



Последовательность комплексной разработки пригодных к тестированию проектов, предлагаемая и используемая фирмой Synopsys, приведена на рис. 2.42. Из рисунка видно, что процедура проектирования с первого же этапа предполагает совместное описание устройства и средств его тестирования.


**Рис. 2.42.** Последовательность разработки пригодного к тестированию ASIC- или SOPC-проекта

## Другие подходы

Приведенный выше материал был посвящен рассмотрению возможностей JTAG-интерфейса, что определялось, во-первых, его эффективностью при простоте использования, а во-вторых, слабой освещенностью этого подхода в отечественной литературе. Чтобы избежать односторонности рассмотрения, стоит упомянуть и об альтернативных вариантах организации тестовых процедур.

Ряд фирм продолжает выпуск средств, опирающихся на традиционные методы и устройства. Среди них такая фирма, как SZ Testsysteme, уже более 20 лет специализирующаяся на выпуске гаммы оборудования, тестирующего системы со смешанным (цифровым и аналоговым) представлением информации и с широким диапазоном возможностей по скорости, диапазону выходных напряжений.

Помимо обычных методов тестирования разрабатываются методы тестирования, базирующиеся на современных технологиях и нетрадиционных методах. Так, фирма Teradyne выпускает аппаратуру для автоматической оптиче-

ской проверки печатных плат Automated Optical Inspection (AOI). Фирма OptEm Engineering Inc. (**www.optem.com**) предлагает аппаратуру и программное обеспечение, которые анализируют печатные платы, затем отображают корпуса ИС, сопротивления и емкости для всех их внешних соединений и выдают отчет о местах потенциальных замыканий и цепях, где можно ожидать временных проблем при передаче сигналов.

## 2.6.4. Системные функции на основе JTAG-интерфейса

Реализация возможностей, предоставляемых JTAG-интерфейсом, определяется правильной политикой на этапе подготовки использования опционных контактов БИС, т. е. на этапе выпуска проектной документации для печатных плат. Поскольку для некоторых типов БИС ПЛ контакты, предназначенные для организации JTAG-интерфейса, являются опционными, желательно, как минимум, оставлять их резервными, а лучше выводить на дополнительные контактные площадки, чтобы, при необходимости, можно было объединить БИС, находящиеся на печатной плате и поддерживающие методы граничного сканирования в единую JTAG-цепочку. Для некоторых типов ПЛИС характерно отсутствие постоянно существующих ячеек граничного сканирования и необходимо на этапах контроля межсоединений загружать в подобные БИС специально подготовленную тестовую конфигурацию, которая и позволит воспользоваться методами граничного сканирования. Подобные мероприятия дадут возможность, в случае необходимости, организовать тестирование межсоединений на изготовленной плате.

### Конфигурирование БИС ПЛИС

Транспортный аспект стандарта JTAG-интерфейса (стандарт IEEE Std.1149.1) позволил стандартизировать, во-первых, прием (передачу) информации в (из) БИС, входящих в JTAG-цепочку, а во-вторых, исполнение команд JTAG контроллерами, входящими в состав каждой БИС-цепочки. В результате он оказался прекрасным инструментом для создания *расширенных версий JTAG-интерфейса* (enhanced JTAG interface), позволив тем самым использовать JTAG-интерфейс для решения не только задач тестирования, но и широкого круга других задач. Возможность адресного направления информации внутрь БИС, расположенной на печатной плате, и одновременное указание о запоминании этого потока информации как содержимого ПЗУ этой БИС, привел к введению специального термина для такой возможности применения JTAG-интерфейса — внутрисхемное программирование (In System Programmability, ISP). При употреблении термина ISP безразлично, является ПЗУ памятью команд МП или памятью конфигурации БИС ПЛ. Если же ОЗУ находится в БИС ПЛ, и оно используется как память конфигурации схемы ПЛИС, то для такого применения JTAG-интерфейса



одится термин "внутрисхемное реконфигурирование" (In System Reconfirability, ISR). Если БИС программируемой логики расположена на плате сширения с интерфейсом PCI (и поскольку по стандарту среди контактов ких плат предусмотрено наличие и контактов JTAG-интерфейса), то появ-ется возможность репрограммировать конфигурацию БИС платы расши-ния, не вынимая самой платы из компьютера. Такая возможность получи-название In Site Programmability, ISP — с совпадением аббревиатуры для утрисхемного программирования.

### звитие средств конфигурирования схем ИСПС

рма Xilinx анонсировала выпуск первой производственной системы, ре-ющей конфигурационные проблемы на системном уровне: технологии tem ACE, System Advantaged Configuration Environment. Гибкость и ем-ть позволяет одной системе ACE CF, содержащей 16-мегабайтную схему яти (CF, CompactFlash) и управляющий контроллер, конфигурировать ату, наполненную БИС FPGA, или целый набор плат, соединенных объ-нительной платой. Подобная централизация упрощает процедуру кон-гурирования систем, содержащих сотни БИС FPGA, позволяет заменять ссив конфигурационных БИС памяти новым содержимым либо осущест-ть выборочную корректировку. Чтобы изменить или корректировать фигурацию системы, можно воспользоваться либо заменой модуля памя-(8 Гбайт), либо корректировать содержимое внутрисистемными средства-через МП-порт, либо загрузить новую конфигурацию, ориентируясь на евые средства, которые используют, например, технологию Internet Re-figurable Logic (IRL).

проектировщиков часто возникает необходимость иметь различные моди-кации конфигурации. В процессе проектирования могут быть созданы очие, тестовые и отладочные конфигурации. Все эти конфигурации мо-оказаться востребованными в различных ситуациях жизни конечной одукции.

### страивание отладочных приборов и инструментов ПЛИС

наконец, широкое распространение в последнее время находит использо-ние транспортного механизма JTAG-интерфейса для обмена информацией фрагментами БИС, предназначенными для отладки схем, расположенных нутри кристалла, что позволяет говорить о внутрисхемной или внутрикри-тальной отладке. Ввиду исключительной важности этого свойства для эф-ективного проектирования современных систем рассмотрим его несколько одробнее. Возможность использования связана с исключительно большими огическим ресурсами (по крайней мере, при постановке мощных ПЛИС опытные образцы и возврате к более дешевым вариантам в серийном ис-полнении).

Определенные отладочные ресурсы закладывались и ранее, например, фирмой Motorola в выпускаемых ею контроллерах (в *разд. 2.3* упоминался блок BDM). В схемах микроконтроллеров этой фирмы широко использовалась методика внешнего доступа к элементам программного обеспечения. Сейчас огромные внутренние ресурсы ПЛИС создают платформу для развития старых и создания новых методик.

Конечно, требуемое теперь время для модификации современной схемы, включая ее перепрограммирование прямо внутри исследуемой системы, намного короче традиционной процедуры перепайки контактов или добавления ИС на свободном монтажном поле и т. д. Но есть различие поведения сигналов, выведенных на наружный контакт (для дополнительного контроля, например), и сигналов схем, находящихся внутри кристалла. Методика отладки должна учитывать возможные модификации схемы, вносимые синтезатором ПЛИС, и поэтому предпочтение стоит отдавать средствам, расположенным непосредственно около исследуемого фрагмента схемы.

Выбор различных вариантов контроля поведения сигналов реализуется, например, в САПР Quartus фирмы Altera. Проектировщик может ориентироваться при отладке как на связь со встраиваемым логическим анализатором внутри БИС программируемой логики этой же фирмы APEX20K (SignalTap embedded logical analisis), так и на результаты защелкивания в соединительном устройстве (MasterBlaster) внешних относительно БИС значений сигналов (до 32 сигналов), присутствующих на произвольных соединительных линиях между БИС.

Для БИС ПЛ класса SOPC фирмы Triscend транспортный механизм JTAG-интерфейса может использоваться не только для загрузки конфигурации и программного кода, но и для совмещенной отладки аппаратных и программных ресурсов. Через JTAG-интерфейс возможна установка аппаратных точек останова для встроенного в БИС специального блока управления точками останова (Hardware Breakpoint Unit) микроконтроллера и доступ к его основным программно-доступным элементам. К ним относятся не только стандартные предопределенные аппаратные ресурсы типа регистров таймера и т. д., но и фрагменты, построенные на программируемых элементах. Наблюдение за контролируемыми параметрами и сигналами возможно не только после остановки системы, но и в процессе ее штатной работы. В последнем случае разработчик задает временной интервал обновления отладочной информации (из-за большой длины сканирующих регистров темп такого обновления не превышает долей секунды).

Фирма TI (Texas Instruments) значительное внимание уделяет проблеме анализа в реальном времени (Real-time Analysis) для отладки своих приложений, связанных с цифровой обработкой (DSP Solutions). Отладка выполняется в реальном масштабе времени. При этом решаются следующие задачи: анализ логики работы ПО, получение информации, необходимой для анализа производительности и оценки эффективности работы ПО (в том числе и



бор статистической информации), управление, при необходимости, повением проекта.

ребования к такому подходу — встраивание предопределенных (базовых) стовых ресурсов и ресурсов, назначаемых пользователем (разработчиком), томатическое (без постоянного управления проектировщиком) поступлеие требуемых данных в отладочные средства, использование интерфейса ального времени между исследуемым устройством и отладочным средством: Real-Time Data Transfer (RTDT), JTAG или другого определяемого льзователем интерфейса. Основными требованиями к базовому тестовому борудованию являются минимальное и предсказуемое паразитное воздействия на выполнение основных функций оборудования и его оптимальность. Точно так же, от встраиваемого разработчиком оборудования требует наличие в нем механизмов управления его встраиванием и включение его состав средств условного включения/выключения.