"КОНВЕРГЕНЦИЯ КРЕМНИЯ" И БУДУЩЕЕ СИСТЕМНОГО ПРОЕКТИРОВАНИЯ

Денни Биран,
первый вице-президент по стратегическому планированию, Altera

Рост степени интеграции микросхем для системных разработчиков означает как новые возможности, так и новые проблемы. С увеличением степени интеграции разработчики чипов получают возможность размещать на одном кристалле большее количество компонентов – процессоров, ускорителей, памяти и контроллеров периферии. Большее число компонентов в одном чипе – это более высокая производительность, меньшая потребляемая мощность и более компактные размеры. Но рост интеграции приводит к тому, что многие решения, которые раньше принимались системными разработчиками, теперь принимаются разработчиками чипов, т.е. последние берут на себя отчасти задачу дифференциации продуктов и реализацию инновационных решений. Для системных разработчиков очень важно понять идеи разработчиков чипов и оставить за собой широкий выбор дифференцированных свойств своих продуктов. В предлагаемой статье на примере одной из важных категорий приложений исследована эволюция архитектуры чипов с учетом новых потребностей рынка.

КАТЕГОРИИ ПРИЛОЖЕНИЙ

Многие из наиболее важных приложений на российском рынке электроники (в том числе системы видеонаблюдения, проводные и беспроводные коммуникации и усовершенствованные системы промышленного управления) построены по единому принципу. В таких приложениях система обеспечивает прием широкополосных сигналов, обрабатывает эти сигналы с целью выделения из них полезной информации, выполняет требующий большого объема вычислений анализ и принимает соответствующие решения, а затем реализует их.

Например, система наблюдения получает 1080-строчную видеоинформацию с прогрессивной разверткой с видеокамеры. Эта система может выполнять обработку видеопотока для выделения контуров, идентификации объектов и разделения объектов, представляющих потенциальный интерес. Для реализации таких функций обычно используются стандартизованные, сравнительно простые, однако требующие большого объема вычислений алгоритмы.

На следующем этапе более мощные блоки обработки анализируют объекты, например, с целью регистрации несанкционированного проникновения или идентификации лиц, представляющих интерес. Эти алгоритмы могут быть специализированными и часто меняться. Наконец, с помощью анализа определяется, требует ли данная ситуация подачи аварийного сигнала, защиты входа или уведомления службы безопасности о нарушении режима.

На практике разработчики используют три разных способа реализации таких систем.

Решение 1: программное обеспечение. Этот подход заключается в разработке программного обеспечения и запуске его на микропроцессоре, специализированной микросхеме (ASSP) или мощном 32-разрядном микроконтроллере. Разработчики выполняют отладку программного обеспечения, уточняют алгоритм, а затем начинают проверку функционирования системы. Если проверка выполняется слишком медленно, скорость повышают двумя путями – задача переносится на отдельный процессор или, если в интегральной схеме предусмотрена подходящая высокопроизводительная подсистема (например, DSP-ядро или векторный процессор), – то на этот ускоритель. Когда все задания начинают соответствовать всем требованиям к временным параметрам, система считается готовой для окончательной проверки функционала, синхронизации и энергопотребления.

В нашем примере с системой видеонаблюдения управляющее программное обеспечение нужно было бы запускать на центральном процессоре. Стандартные алгоритмы обработки изображений могли бы работать на базе стандартных библиотечных программ на DSP-ядре, в то время как более сложные специализированные алгоритмы необходимо было бы разрабатывать вручную, чтобы обеспечить их параллельное исполнение на всех доступных ядрах процессора.

Такой способ проектирования имеет важные преимущества. Он ориентирован на программное обеспечение и, следовательно, на функциональные возможности системы. И поскольку большая часть функций системы реализована программно, систему сравнительно легко модернизировать в случае возникновения ошибок или изменения системных требований. Но обычно программное обеспечение, запущенное на процессоре или DSP-ядре, – наиболее медленный и энергозатратный способ реализации алгоритма. Поэтому программно-ориентированный подход – это не самый лучший вариант реализации системы, так как имеет ограничения по производительности и эффективности. И, более того, поскольку различные функции системы являются программно реализуемыми, они легко могут копироваться конкурентами или изыматься обманным путем злоумышленниками – теми, кто имеет доступ к этим аппаратным средствам.

Решение 2: аппаратные средства. Этот противоположный подход к проектированию системы заключается в разработке аппаратных средств непосредственно на базе системных требований. Параллельно создается программное обеспечение, которое должно быть запущено на этой аппаратуре. Именно так разрабатывают специализированные интегральные схемы (ASIC). Сначала системный архитектор определяет, какие потребуются процессоры, ускорители, память, контроллеры и затем передает эти требования группе разработчиков чипа, которые приступают к проектированию ASIC.

В нашем примере системный архитектор мог бы выбрать пару ARM-ядер для запуска системного программного обеспечения, получить лицензию на процессор обработки изображений от стороннего производителя для решения задач предварительной обработки изображений и разработать специализированный микропрограммный DSP-конвейер для реализации сложных алгоритмов. В процессе проектирования интегральной схемы группа разработчиков программного обеспечения будет вынуждена иметь дело с тремя наборами программных инструментов разработки и отладки для трех совершенно разных подсистем.

Аппаратно-ориентированный подход имеет определенные преимущества: он обеспечивает максимальное быстродействие системы и максимальную энергоэффективность для любой технологии. Но этот метод требует наличия опытной команды разработчиков интегральных схем и, при переходе на новый технологический процесс, значительных инвестиций. Кроме того, после завершения разработки ASIC внесение изменений в аппаратные средства, коррекция ошибок и модернизация при изменении требований к системе будут затруднены, связаны с большими затратами и потребуют много времени. Решение проблемы программным путем может сэкономить день (правда, в ущерб скорости и потребляемой мощности), что делает подход на основе ASIC достаточно привлекательным.

Таким образом, аппаратно-ориентированный подход является наилучшим выбором для проектов, требующих высокой производительности и минимального энергопотребления. Однако на практике разработчики создают ASIC только в том случае, когда ожидается их крупносерийное производство, что может оправдать затраты и риск, или тогда, когда известно, что требования к аппаратным средствам, вероятнее всего, не претерпят изменений в процессе эксплуатации продукта. Во многих случаях команды разработчиков, столкнувшиеся с критичными проблемами в проекте, часто обращаются к подходу на основе ASIC

или приобретают ASSP, приблизительно отвечающие требованиям системной ИС, которую они не могут спроектировать самостоятельно.

Решение 3: FPGA ориентированное. Это третий альтернативный путь. Во многих отношениях FPGA являются средним вариантом реализации системы между программно-ориентированным подходом на базе процессора и аппаратно-ориентированной альтернативой на базе ASIC. Алгоритм, реализованный на FPGA, не так просто модифицировать, как в программе, но изменение конфигурации FPGA намного проще, чем выпуск новой версии ASIC, даже если изменения касаются всего нескольких слоев металлизации. С другой стороны, задание на FPGA может выполняться намного быстрее и при меньшей мощности, чем то же самое задание, реализованное программным способом. Но по сравнению с эквивалентной ASIC версия на FPGA, как правило, работает медленнее и потребляет больше мощности.

Вследствие этого, системные разработчики обращаются к FPGA, когда только одно программное решение не способно обеспечить требования по быстродействию и потребляемой мощности либо когда нет возможности найти ASSP с подходящими функциями, либо когда ограниченный бюджет, малый предполагаемый объем выпуска или наличие вероятности внесения изменений исключают применение ASIC. Надо сказать, что так как данная ситуация возникает достаточно часто (что выгодно поставщикам FPGA), то в последние годы продажи FPGA растут более быстрыми темпами, чем продажи альтернативных вариантов (рис.1).

В примере с системой видеонаблюдения разработчики могли бы найти оптимальную комбинацию, состоящую из стандартного микропроцессора, на котором запущено системное программное обеспечение, стандартного IP-блока обработки изображений и специализированного DSP-конвейера для сложных вычислений. Таким образом, проект на FPGA напоминал бы проект на ASIC на уровне функциональных блоков, хотя на вентильном уровне реализация может быть совершенно другой.

Поиск оптимального решения. Оптимальное решение – это когда системному разработчику не нужно выбирать один путь вместо другого, и у него есть возможность выбрать наилучший способ

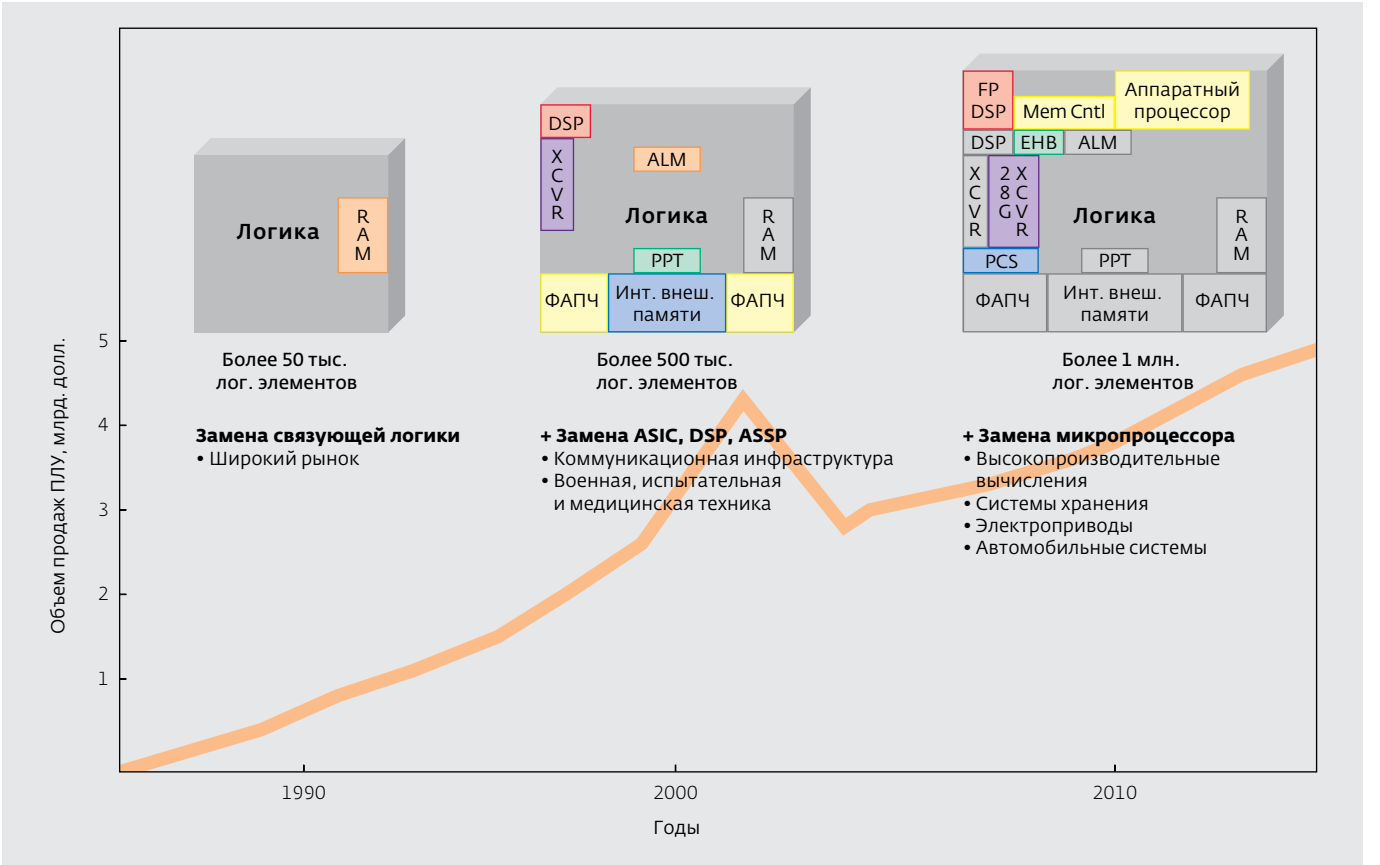

Рис.1. Развитие архитектуры FPGA и продажи программируемых логических устройств

реализации для каждой задачи. Некритичные задания могли бы быть реализованы программным путем и запущены на центральном процессоре, а критичные (с точки зрения производительности или потребляемой мощности) задания – определены стандартами и потому не изменялись и были бы привязаны к аппаратуре. Задачи, которые требуют аппаратной поддержки, но могут быть модифицированы, можно реализовать с помощью программируемой логики на FPGA.

Еще совсем недавно это была общепринятая практика. Степень интеграции была невелика, так что микропроцессоры, ускорители, сложные контроллеры интерфейса и FPGA были отдельными чипами. Но когда технология кристаллов подошла к 90-нм рубежу, системы на кристалле (СнК) стали включать в себя все эти функции, кроме программируемой матрицы FPGA. И большая часть решений по реализации приложения стала приниматься разработчиками СнК, а не системными разработчиками. Разработчики системы могли обеспечить дифференциацию устройства только путем выбора оптимальной СнК, создавая уникальное программное обеспечение и, по возможности, продуманный интерфейс сопряжения FPGA с СнК.

Теперь ситуация снова меняется – разработчикам чипов доступно огромное количество транзисторов, расположенных на одном кристалле. Эта тенденция в компании Altera получила название "конвергенция кремния". В мощные микроконтроллеры были добавлены специализированные аппаратные средства, поэтому они стали напоминать ASIC. ASIC и ASSP могут включать в себя мощные 32-разрядные процессоры, поэтому они стали походить на микроконтроллеры высоко класса. И FPGA, такие, например, как семейство компании Altera, объединяют в себе как многоядерные процессоры, так и специализированные аппаратные блоки, что на практике дает возможность системному разработчику выбирать программное обеспечение, специализированные аппаратные средства или программируемую логику в зависимости от характера реализуемой задачи (рис.2).

В нашем примере разработчики могли бы использовать такой "конвергированный" чип, чтобы организовать запуск системного программного обеспечения и многопотоковой части алгоритма по обработке изображений на двух мощных процессорных ядрах. Оставшуюся часть алгоритма можно было бы реализовать на комбинации аппаратных DSP-ядер и программируемой матрицы, которые также входят в состав чипа.

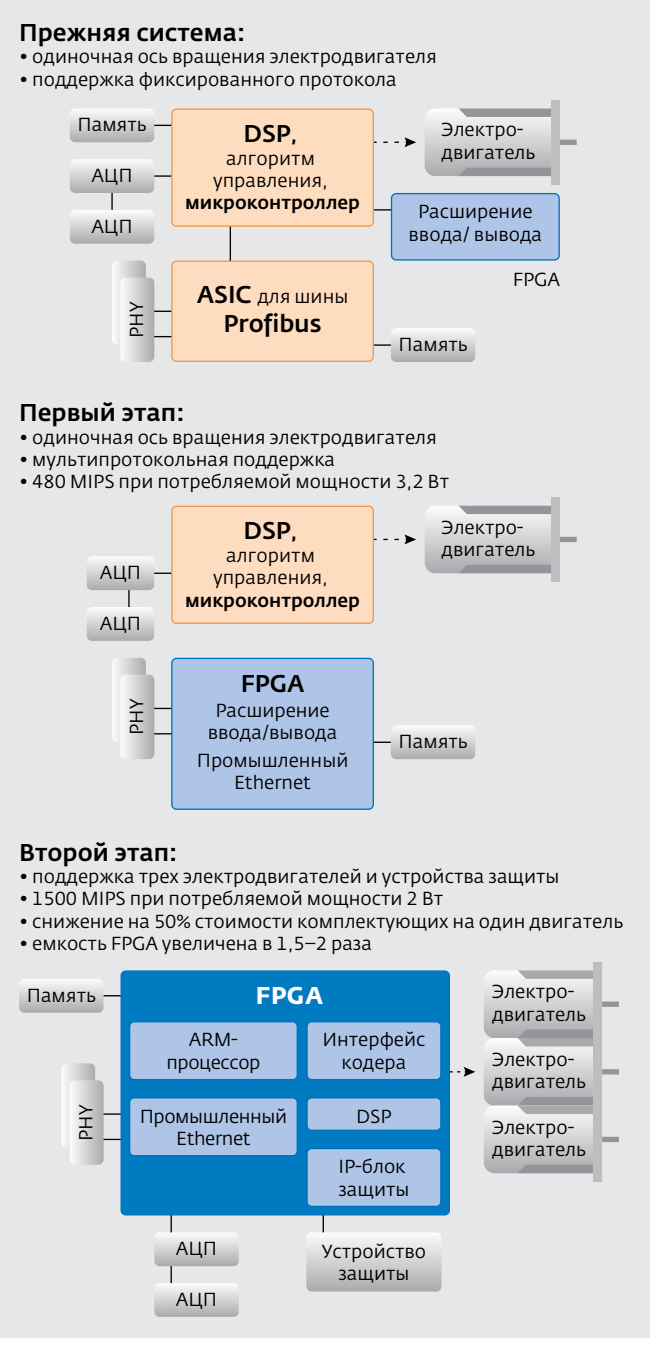

Рис.2. Применение FPGA для реализации энергоэффективных электроприводов

Поскольку увеличивающаяся стоимость разработки ограничивает применение ASIC, тенденция на "конвергенцию кремния" позволяет объединить оставшиеся три системных решения. Микроконтроллеры, ASSP и FPGA становятся все более похожими друг на друга. Но есть очень важное положение: по причинам технологического характера и в силу закона об интеллектуальной собственности только FPGA способны предложить

самую современную технологию программируемой логики. Поэтому только FPGA могут всецело поддерживать стратегию системного разработчика на дифференциацию устройств вплоть до аппаратного уровня.

БУДУЩЕЕ ДИФФЕРЕНЦИАЦИИ КОНЕЧНЫХ ПРОДУКТОВ

"Конвергенция кремния" определит направление развития системной разработки на следующие несколько лет. С одной стороны, микроконтроллеры высшего класса и ASSP будут основой систем, аппаратная часть которых практически станет стандартным продуктом, а дифференциация между системами на рынке переместится в сферу программного обеспечения. С другой стороны, аппаратно-дифференцированные системы на базе FPGA будут отличаться от всех остальных продуктов.

Такое расхождение увеличится вследствие развития двух перспективных технологий: интегральных 3D-схем и гетерогенных систем программирования. Технология 3D-схем позволит интегрировать совершенно разные технологии (например, FPGA, микропроцессоры, DRAM и радиочастотные схемы) в виде слоев кристаллов, что радикально улучшит временные параметры и энергопотребление по сравнению с отдельными чипами. Одним из первых примеров развития такой технологии стал процессор Intel Atom серии E6x5C, в котором интегрированы центральный процессор Atom и FPGA компании Altera. Процессор Atom обеспечивает стандартную в отрасли архитектуру для запуска программного обеспечения, а FPGA – возможность создания специализированных ускорителей и контроллеров интерфейса.

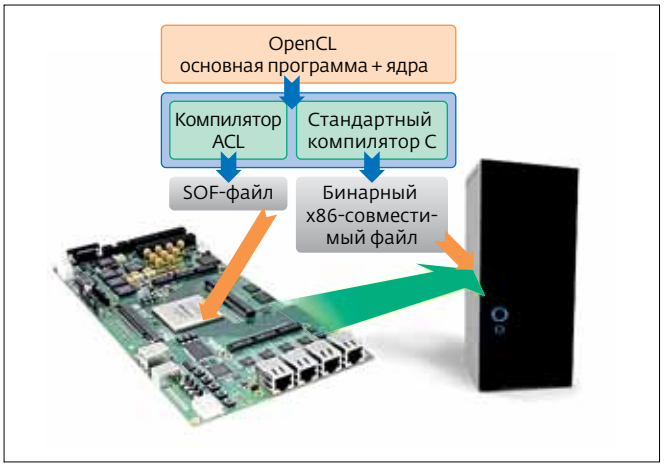

Рис.3. Единая среда разработки на базе OpenCL-FPGA

Серия процессоров E6x5C также подтверждает потребность в развитии второго перспективного направления – гетерогенной программной среды. Оптимально, когда системные разработчики могут начать просто с создания и отладки программного обеспечения для одного центрального процессора. Платформа разработки в дальнейшем могла бы обеспечить поддержку для разработчиков при определении критичных участков кода, распределяя задачи по нескольким ядрам процессора с использованием кэш-памяти и создавая аппаратные ускорители для критичных программных ядер. Таким образом, команда разработчиков могла бы оптимизировать проект до тех пор, пока не будут достигнуты требуемые показатели по временным параметрам и энергопотреблению.

Примером такой среды разработки является проект OpenCL-FPGA, который в настоящее время находится в процессе развития (рис.3). Цель данной работы – формирование единой среды, в которой системные разработчики могли бы создавать программы на диалекте языка C, отделять программные ядра, требующие большого объема вычислений, генерировать параллельные аппаратные подсистемы для ускорения программных ядер и интегрировать полученную аппаратно-программную систему.

Обусловленная ростом степени интеграции кремниевых чипов "конвергенция кремния" позволяет собрать все основные электронные блоки в одном корпусе, ограничивая возможности системных разработчиков в полной мере дифференцировать свои конечные продукты. Однако FPGA, внешне все больше напоминая ASSP и микроконтроллеры, в действительности увеличивает потенциал системных разработчиков по дифференциации аппаратных средств. Развитие перспективных технологий – 3D-схем и гетерогенных программных сред разработки – позволит ускорить отделение интегральных схем системного уровня на базе FPGA от традиционного мира микроэлектроники.

В заключение можно сказать. Рассмотрены преимущества и недостатки трех различных способов проектирования системы: программного, аппаратного и FPGA ориентированного. Показано, что реализация приложения на базе FPGA позволяет системным разработчикам принимать оптимальные решения с точки зрения быстродействия и энергопотребления, и, в то же время, сохранять достаточную гибкость и возможность модификации системы.

Публикуемый материал представляет собой перевод статьи Danny Biran. Silicon Convergence and the Future of System Design (www.altera.com).