САМОСИНХРОННАЯ ЭЛЕКТРОНИКА:
НАПРАВЛЕНИЯ РАЗВИТИЯ

Бурное развитие электроники предъявляет жесткие требования к разработчикам микросхем. Необходимо постоянно увеличивать их сложность и быстродействие. При таких условиях начинают проявляться недостатки синхронного проектирования, связанные с тем, что большое число элементов микросхемы трудно синхронизировать с помощью единого тактового сигнала: увеличивается расфазировка и дрожание фронта тактового сигнала, энергопотребление цепей распространения тактового сигнала может достигать 30% общего потребления микросхемы. Все это снижает скорость работы микросхемы. Решить эту проблему можно с помощью самосинхронной схемотехники, лишенной указанных недостатков.

А.Руткевич, В.Стешенко, к.т.н., Г.Шишкин
info@dsol.ru

Самосинхронный подход является одним из методов локального решения проблемы тактирования микросхемы. В отличие от синхронного подхода, где все события в цифровой схеме подчиняются единому тактовому сигналу, в самосинхронной схеме каждый комбинационный блок сам синхронизирует свою работу с соседними блоками за счет отслеживания момента окончания вычислений. Как только блок заканчивает вычисления, он вырабатывает сигнал, по которому начинает работать следующий блок, а на вход этого блока могут поступать новые данные. Таким образом, обеспечивается логическое упорядочивание событий в схеме. Данные между логическими блоками передаются с применением некоторого протокола передачи, который отвечает за синхронизацию информации. В задачу протокола входит сопровождение передаваемых данных парой "запрос/подтверждение", которая обеспечивает синхронизацию на локальном участке схемы. В зависимости от реализации сигнал запроса передается отдельно или встраивается непосредственно в данные.

К достоинствам самосинхронных схем можно отнести меньшее потребление энергии, низкий уровень перекрестных наводок и электромагнитного излучения, большую стойкость к технологическому разбросу параметров элементов, температуры и напряжения питания. Но проектировать самосинхронные схемы сложнее, чем другие. Если при синхронном проектировании разработчик имеет дело с отдельными сигналами и работа схемы синхронизируется за счет общего тактового сигнала, то в самосинхронных схемах этот подход не применим. При проектировании таких схем разработчику необходимо использовать специальные каналы, с помощью которых передаются данные между блоками и синхронизируется их работа. Соединить вместе несколько блоков невозможно без дополнительных специальных компонентов, синхронизирующих их функционирование. Кроме этого, для синхронизации работы схемы в целом необходимы дополнительные средства, которые при синхронном подходе были не нужны. Все это требует изменения мышления разработчика и новых средств автоматического проектирования. Сейчас предпринимаются попытки сделать автоматический синтез самосинхронных схем из синхронного описания, но результат оставляет желать лучшего, хотя такой подход имеет право на жизнь.

Разработки самосинхронных схем велись с 50-х годов прошлого века, но тогда победил синхронный подход, менее трудоемкий с точки зрения проектирования. Возвращение к самосинхронной схемотехнике связано с появлением ультрабольших интегральных схем, в которых трудно обеспечить распространение общего тактового сигнала. Многие компании, в том числе Intel, IBM и Motorola, проводили исследования в области самосинхронных вычислений.

Первый самосинхронный процессор был разработан профессором Элэйном Мартином в Калифорнийском технологическом институте [1], а сама идея принадлежит одному из создателей компьютерной графики – Айвану Сюзерланду, написавшему первую статью о нетактируемой логике. В 1990 году в Университете Манчестера в Англии по этому направлению была создана рабочая группа, а в 1994 году был разработан первый самосинхронный чип Amulet1 для сотовых телефонов [2] (рис.1а). Процессор Amulet1 был создан по кремниевой технологии 1,0 мкм с двумя слоями металлизации и содержал 60000 транзисторов. Для коммуникаций внутри ядра процессора, основанного на микроархитектуре Сатэрланда [3], использовался двухфазный протокол передачи данных. Микропроцессор выполнял набор команд процессора ARM6, за исключением умножения с накоплением. Производительность процессора со-

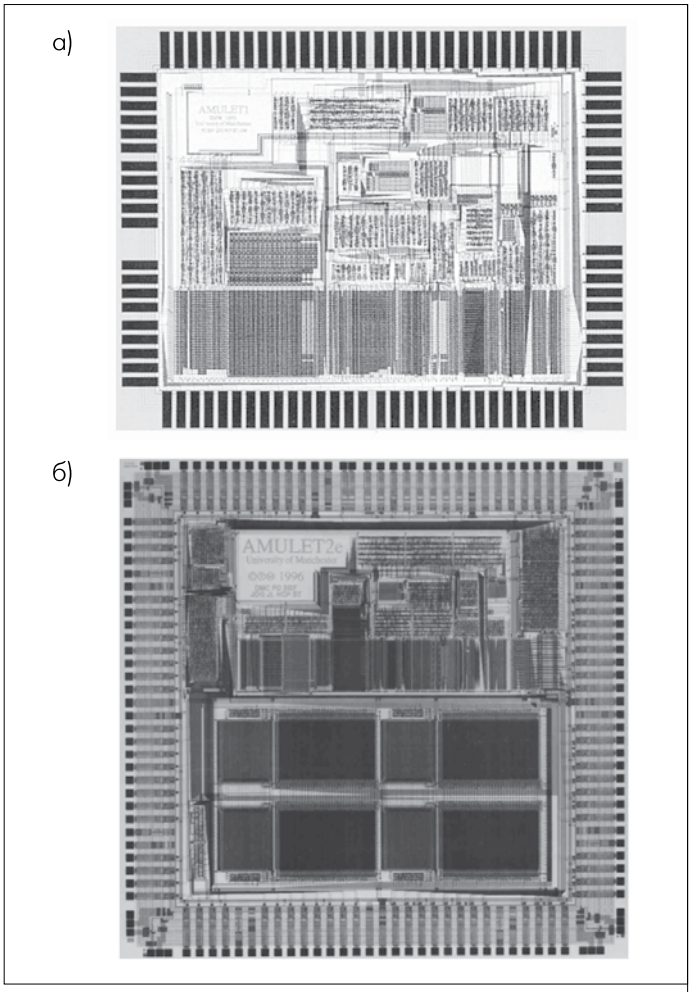

Рис.1. Топология микропроцессоров Калифорнийского технологического института: а – Amulet1; б – Amulet2e

ставляла примерно половину от ARM6, изготовленного с помощью того же технического процесса и обладающего аналогичной энергетической эффективностью (MIPS/Вт).

На смену Amulet1 в 1996 году пришел процессор Amulet2e [4] (рис.1б), совместимый с полным набором команд ARM7. Помимо центрального ядра процессор содержал синхронную кэш-память и гибкий внешний интерфейс, позволяющий подключать внешние функциональные блоки. Была добавлена возможность прогнозирования ветвления для увеличения скорости вычислений. Внутри ядра микропроцессора также использовался двухфазный протокол передачи данных. Amulet2e содержал 450000 транзисторов (главным образом в кэш-памяти), был создан по кремниевой технологии 0,5 мкм с применением трех слоев металлизации. Производительность Amulet2e была приблизительно в три раза выше, чем у Amulet1, но составляла только половину от производительности аналогичного синхронного микропроцессора.

В 2000 году был разработан процессор Amulet3i [5], совместимый с системой команд ARM9. Он предназначался для работы в составе системы на кристалле (СнК) и не являлся автономным устройством. Кристалл процессора был создан по кремниевой технологии 0,35 мкм с тремя слоями металлизации и содержал приблизительно 800000 транзисторов. Устройство включало центральный процессор, 8-Кбайт двухпортовое ОЗУ, 16-Кбайт ПЗУ, диспетчер прямого доступа к памяти и интерфейс внешней памяти/тестирования, объединенные с помощью самосинхронной шины MARBLE.

Amulet3i обладал такой же производительностью, как и современный ему синхронный микропроцессор на основе ARM9, но выигрывал в энергопотреблении. Ядро Amulet3i задействовано в СнК DRACO (DECT Radio Communications Controller), предназначенной для работы в телекоммуникациях.

Известно еще несколько примеров реализации самосинхронных процессорных устройств. Калифорнийский технологический институт произвел два чипа: Caltech (1989) [6], который стал первым самостоятельным самосинхронным 16-битным RISC-процессором, и MiniMIPS [7], выполненный по архитектуре R3000.

Еще один микропроцессор, TITAC-2, был разработан Токийским университетом в 1997 году [8] и основан на архитектуре R2000. В процессе проектирования использовалась ручная трассировка кристалла.

Из самосинхронных DSP-процессоров известен также ASPRO-216, разработанный в 1998 году в Гренобле (рис.2).

Работы по проектированию самосинхронных устройств проводились и в нашей стране [9, 10]. Их основоположником является доктор технических наук В.И.Варшавский, под руководством которого работал коллектив специалистов ИПИАН (Институт проблем информатики Академии наук, теперь – ИПИ РАН).

Все упомянутые выше устройства являлись опытными образцами, не получившими коммерческого применения.

В 1998 году с использованием системы синтеза "Tangram" собственной разработки научно-исследовательской лабораторией Philips был создан самосинхронный микроконтроллер на базе системы команд 80C51 [11]. Он применялся в пейджерах, где низкое энергопотребление и электромагнитная совместимость критически важны.

Самосинхронный микропроцессор Sharp DDMP (Data-Driven Media Processor) [12], созданный в 1997 году, был предназначен для мультимедийных приложений и содержал множество параллельных блоков обработки сигналов.

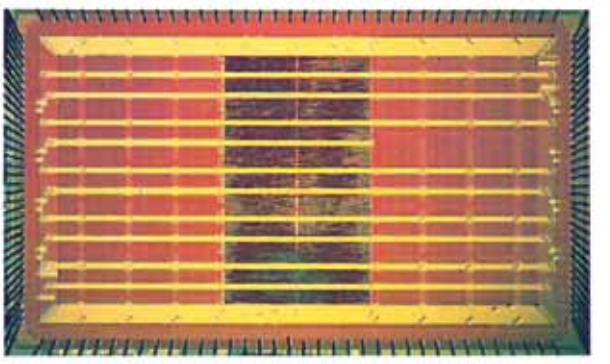

Рис.2. Топология микропроцессора ASPRO-216

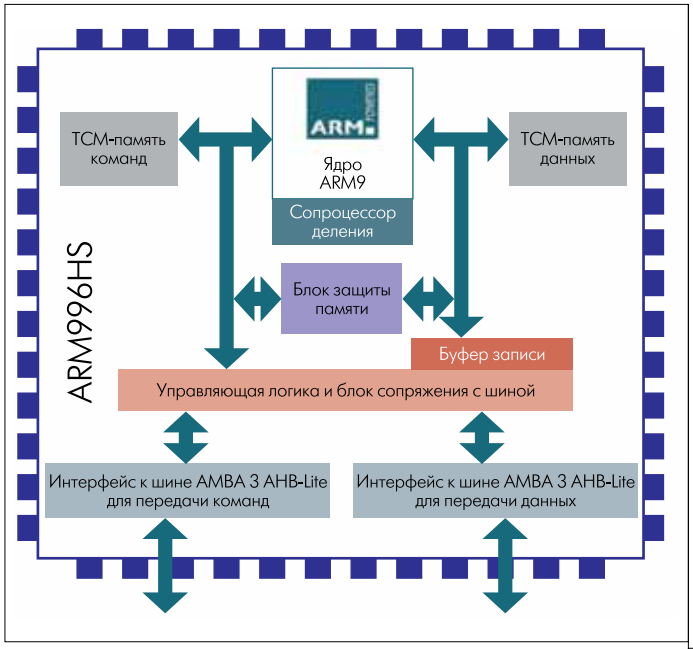

Рис.3. Функциональная схема процессора ARM996HS

Сегодня наиболее широко распространены микропроцессоры семейства ARM. Первым самосинхронным устройством коммерческого назначения из этого семейства стал ARM996HS, созданный в 2006 году компаниями ARM Ltd и Handshake Solutions (подразделение Royal Philips Electronics). Процессор расходует очень мало энергии и демонстрирует низкий уровень электромагнитных помех, причем такие параметры достигаются без снижения производительности. Это позволяет успешно использовать процессор в автомобильных, медицинских и встраиваемых приложениях. ARM996HS – первый промышленно выпускаемый самосинхронный процессор, предназначенный для работы в отказоустойчивых и быстродействующих устройствах.

Процессор ARM996HS включает в себя самосинхронное 32-битовое ядро RISC-архитектуры ARMv5TE с набором команд Thumb, блок защиты памяти MPU, аппаратный делитель, шину Dual AMBA AHB-Lite, быстрый 32-битный MAC, стандартные синхронные интерфейсы (рис.3). В архитектуре процессора используются самосинхронизирующиеся цепи, разработанные компанией Philips, которые ранее применялись в смарт-картах, пейджинговых устройствах, мобильных телефонах. Отсутствие внешнего источника синхронизации делает процессоры более устойчивыми к радиопомехам.

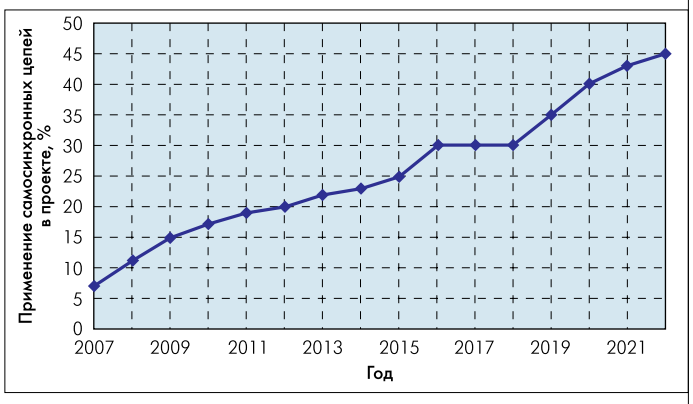

Рис.4. Прогноз доли самосинхронной логики в разрабатываемых микросхемах

В СнК процессор ARM996HS может использоваться как в синхронных, так и в самосинхронных проектах. В настоящее время процессор ARM996HS доступен всем желающим для лицензирования. Средства разработки приложений для процессора – семейство САПР ARM RealView DEVELOP, в том числе пакеты RealView Development Suite, RealView ICE и RealView Trace. Разработчики также могут воспользоваться библиотеками ARM Metro (для приложений с низким энергопотреблением) и ARM Advantage (для высокопроизводительных применений).

Прогнозы ассоциаций производителей микроэлектроники (США, ЕС, Японии, Кореи и Тайваня), представленные в виде международного плана развития индустрии полупроводников (International Technology Roadmap for Semiconductors – ITRS) [13], показывают четкую тенденцию к сдвигу в сторону самосинхронной парадигмы проектирования (рис.4). Планируется, что к 2012 году применение самосинхронных цепей в проекте составит 20% от всей схемы, а к 2020 году – уже 40%.

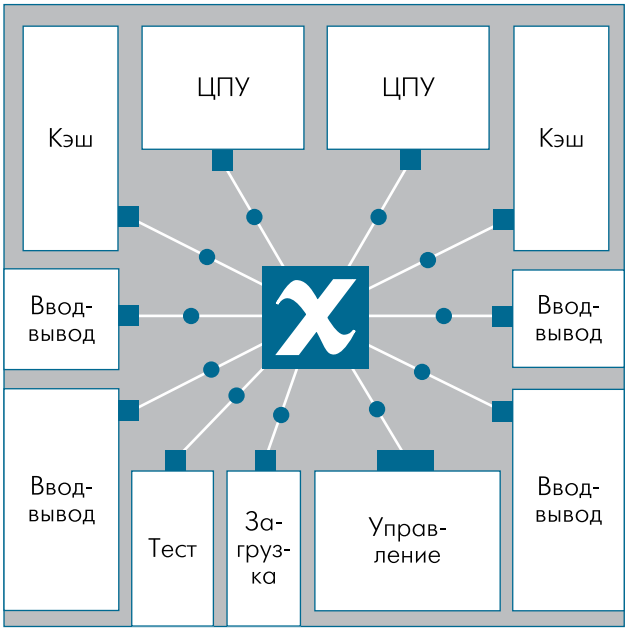

Рис.5. Матрица соединений Nexus Crossbar

Сегодня самосинхронная схемотехника наиболее перспективна для реализация каналов передачи данных между синхронными блоками (глобально самосинхронное/локально синхронное проектирование). Такое решение позволяет уменьшить размеры тактовых областей, одновременно увеличив скорость их работы. Одним из лидеров в данной области является компания Fulcrum Microsystems, которая использует самосинхронную реализацию матрицы соединений (Nexus Crossbar, рис.5) в линейке микросхем, предназначенных для построения высокоскоростных маршрутизаторов [14]. С помощью данного подхода удалось значительно улучшить характеристики конечного устройства, а

именно – в 10 раз уменьшить задержку передаваемых данных и в четыре раза снизить энергопотребление [15].

Законченное решение для построения самосинхронных каналов передачи данных внутри микросхемы – сетей на кристалле (Network-on-Chip) – предлагает компания Silistix. Ее продукт CHAINworks, включающий средства разработки и IP-ядра, позволяет создавать высокоскоростные сети пакетной передачи данных между блоками [16], в которых реализуется как синхронный подход CHAINworks Sync, так и глобально самосинхронный/локально синхронный CHAINworks GALS (рис.6). Разработчику достаточно описать требования к шине на специальном языке, после чего он получает HDL-описание сгенерированной программным обеспечением сети. За счет встроенной поддержки наиболее популярных шин передачи данных (например, AMBA) процесс разработки заметно упрощается. Несмотря на то, что сеть на кристалле занимает большую площадь, она обеспечивает более высокое быстродействие и меньшее энергопотребление по сравнению с синхронными шинами.

Отдельным направлением самосинхронной схемотехники являются программируемые логические интегральные схемы (ПЛИС). Лидер в этой области – компания Achronix Semiconductor [17]. Она выпустила семейство самосинхронных ПЛИС Achronix Speedster FPGA. Микросхемы и средства разработки стали доступны в начале 2009 года. Данное семейство включает четыре микросхемы, отличающиеся логической емкостью. ПЛИС представляет собой (рис.7) вычислительное ядро, которое состоит из набора реконфигурируемых логических блоков (РЛБ) и матрицы коммутирующих блоков (КБ), соединенных линиями связи. Вокруг логического ядра находится периферийная область, содержащая преобразователи синхронных интерфейсов в самосинхронные и дополнительные интерфейсные блоки.

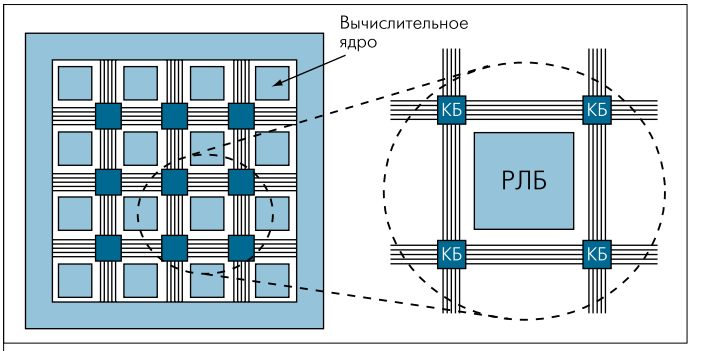

Рис.7. ПЛИС Achronix Speedster FPGA

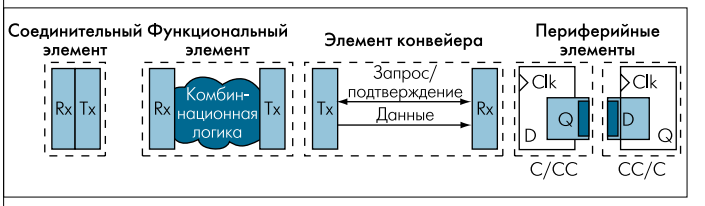

Рис.8. Логические элементы ПЛИС Achronix Speedster FPGA

Элементы, из которых состоит логическое ядро, можно представить в виде четырех логических блоков (рис.8):

- соединительного элемента, являющегося самосинхронным элементом памяти;
- функционального элемента, выполняющего логическую функцию;
- элемента конвейера;
- периферийных элементов: преобразователя синхронного в самосинхронный сигнал (C/CC) и самосинхронного в синхронный (CC/C).

С помощью этих элементов можно строить самосинхронные конвейерные схемы (рис.9) и реализовывать самосинхронные схемы в ПЛИС (рис.10). Соединительные эле-

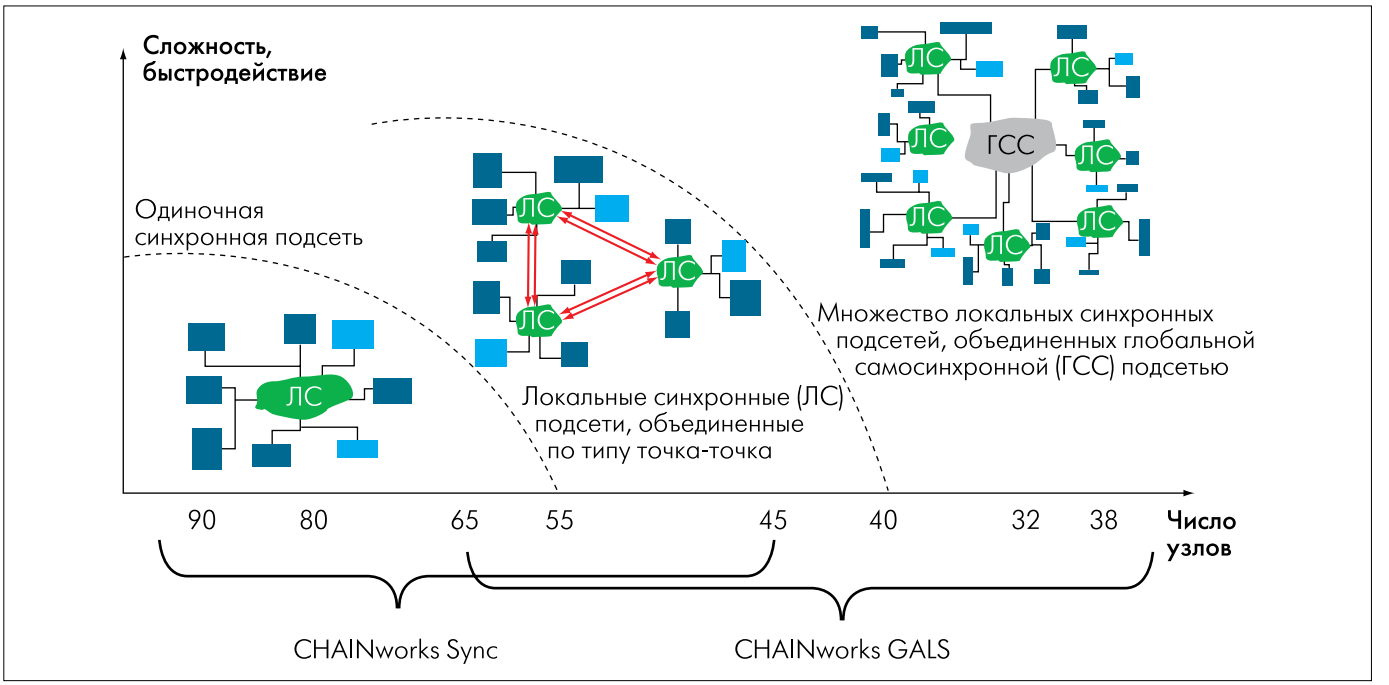

Рис.6. Зависимость метода проектирования от количества соединяемых блоков и быстродействия всей системы

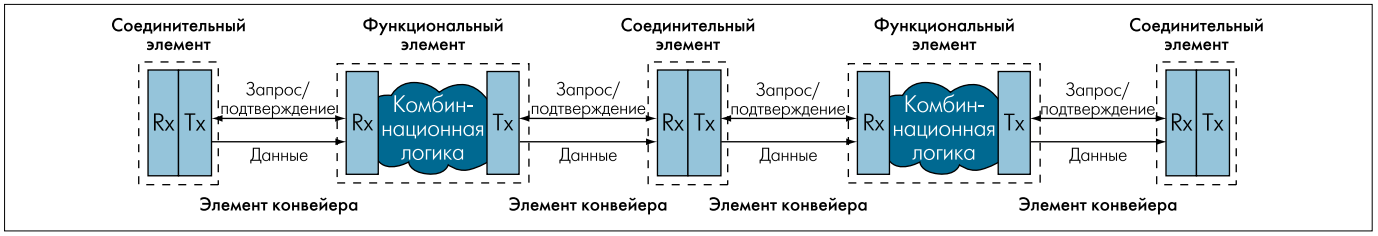

Рис.9. Пример самосинхронной конвейерной схемы

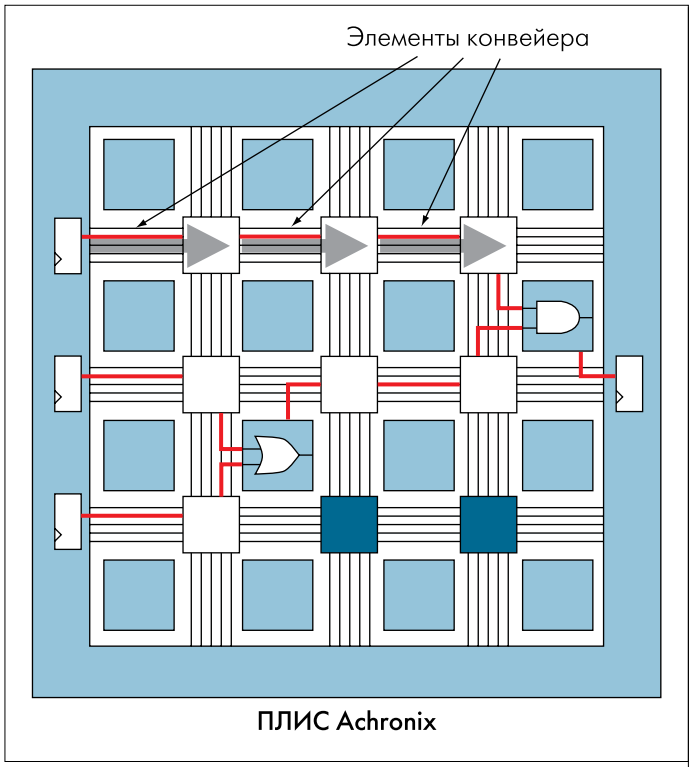

Рис.10. Реализация самосинхронной схемы в ПЛИС Achronix Speedster FPGA

менты выполняют на элементах матрицы коммутирующих блоков, функциональные – на реконфигурируемых логических блоках.

Achronix предлагает следующий маршрут разработки самосинхронных ПЛИС:

- RTL-описание схемы на языках описания аппаратуры с использованием обычных синхронных методов проектирования;
- логический синтез с помощью специальных версий программ Synplicity Synplify Pro или Mentor Graphics Precision Synthesis с использованием библиотеки синхронных элементов Achronix;
- размещение и разводка с помощью Achronix CAD Environment (ACE). Кроме этого на данном этапе программное обеспечение ACE выполняет преобразование синхронной схемы в самосинхронную и позволяет выполнить временной анализ и моделирование.

Achronix максимально скрыл от пользователей самосинхронный подход к проектированию. Исходными данными для проектирования является синхронное RTL-описание схемы, все преобразования в самосинхронную схему выполняются внутри ACE и скрыты от пользователя.

Основное преимущество микросхем Achronix Speedster – большая скорость обработки данных. Если сравнить скорость работы блока шифрования AES-128, реализованного в самосинхронной ПЛИС Achronix Speedster и в обычной синхронной ПЛИС Xilinx Virtex-5 [18], обнаруживается более чем двукратный выигрыш в быстродействии в случаях, когда для решения данной задачи применялась самосинхронная ПЛИС (см. таблицу). При этом цены указанных микросхем различаются незначительно.

Таким образом, самосинхронные микросхемы обладают рядом преимуществ. К сожалению, сдерживающим фактором развития самосинхронной схемотехники является отсутствие средств автоматического проектирования. Существующие методы получения самосинхронной схемы из синхронного описания не позволяют максимально использовать все имеющиеся преимущества. Специализированные средства разработки самосинхронных схем только начинают появляться и слабо интегрированы в существующие маршруты проектирования и производства микросхем. Кроме того, необходимо учитывать, что идеология проектирования отличается от используемой в синхронных схемах, что также является ограничением, поскольку потребует времени на переобучение разработчиков.

Скорости работы блока шифрования AES-128, реализованного в ПЛИС Achronix Speedster и Xilinx Virtex-5

Платформа	Ресурсы		Задержка, тактов	Темп обработки, бит/такт	Максимальная частота, МГц	Производительность, Гбит/с
	Логические блоки	Блоки памяти				
Achronix Speedster	19,100	100	55	128	950	120
Xilinx Virtex-5	11,800	100	55	128	425	54

ЛИТЕРАТУРА

1. Асинхронные процессоры. – chernykh.net/content/view/431/638.
2. **Woods J.V., Day P., Furber S.B.** et al. AMULET1: An asynchronous ARM processor. – IEEE Transactions on Computers, 1997, 46(4), p.385.
3. **Suherland I.E.** Micropipelines. – Communications of the ACM, 1989, 32(6), p.720.
4. **Furber S.B., Garside J.D., Riocreux P.** et al. AMULET2e:

An asynchronous embedded controller. – Proceedings of the IEEE, 1999, 87(2), p.243.
5. **Garside J.D., Bainbridge W.J., Bardsley A.** et al. AMULET31 – an asynchronous system-on-chip. – Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems: IEEE Computer Society Press, 2000, p.162.
6. **Marin A.J., Burns S.M., Lee T. K.** et al. The design of an asynchronous microprocessor. – Advanced Research in VLSI: MIT Press, 1989, p.351.
7. **Marin A.J., Lines A., Manohar R., M.** et al. The design of an asynchronous MIPS R3000. – Proceedings of the 17h Conference on Advanced Research in VLSI: MIT Press, 1997, p.164.
8. **Takamura A., Kuwako M., Imai M.** et al. TITAC-2: An asynchronous 32-bi microprocessor based on scalable-delay-insensitive model. – Proc. International Conf. Computer Design (ICCD'97): MIT Press, 1997, p.288.
9. Автоматное управление асинхронными процессами в ЭВМ и дискретных системах/ Под. Ред. В.И. Варшавского. – М.: Наука, 1986.
10. **Varshavsky V., Kishinevsky M., Marakhovsky V.** et al. Self-timed Control of Concurrent Proc-esses. – Kluver Academic Publishers, 1990.
11. **H. van Gageldonk, Baumann D., C.H. van Berkel** et al. An asynchronous low-power 80c51 mi-croconroller. – Proc. Inernaional Symposium on Advanced Research in Asynchronous Circuits and Systems: IEEE Computer Society Press, 1998, p.96.
12. **Terada H., Miyaa S., Iwaa M.** DDMPs: Self-timed superpipelined data-driven mulimedia processors. – Proceedings of the IEEE, 1999, 87(2), p.282.
13. International Technology Roadmap for Semiconductors 2007. Design. – www.itrs.net.
14. **Lines A.** Nexus: An Asynchronous Crossbar for Synchronous SoC Designs – Fulcrum Microsys-tems, 2003. www.fulcrummicro.com/documents/technology/hoti_03_nexus.pdf.
15. Scaling the Cloud. Using the FocalPoint Fat Tree Architecture. – Fulcrum Microsystems, July, 2009. www.fulcrummicro.com/product_library/applications/Scaling_The_Cloud.pdf.
16. CHAIN works. – Silistix 2009. www.silistix.com/Papers/CHAINworks.pdf
17. Speedster FPGA Family DS001 Rev. 1.0 – Achronix Semiconductor Corporation, 2008. http://www.achronix.com.
18. **Lewis J. R.** Advanced Encryption Standard IP Cores. Product brief. – Signali Corporation, 2009. www.signalicorp.com/products/pdf/ProductBrief_AES.pdf.