УДК 004.052.42

Средства системной отладки рекуррентного вычислителя

Д.Ю. Степченков, В.С. Петрухин, Н.В. Морозов

Федеральное государственное бюджетное учреждение науки Институт проблем информатики Российской академии наук (ИПИ РАН), stepchenkov@mail.ru

***Аннотация* — Проведен анализ средств отладки САПР Quartus II, в соответствии с выбранными критериями определен состав средств системной отладки рекуррентного вычислителя. Предложена оптимальная структура аппаратных средств отладки рекуррентного вычислителя.**

***Ключевые* слова — потоковая архитектура, отладочные средства, рекуррентность.**

I. Введение

В Институте проблем информатики ведутся работы по созданию вычислителя нетрадиционной рекуррентной архитектуры, предназначенного для реализации параллельных вычислений в области цифровой обработки сигналов [1, 2]. Причиной работ в этой области послужили ограничения традиционной фон–неймановской архитектуры, в частности то, что возможности дальнейшего повышения производительности микропроцессоров за счет наращивания числа транзисторов и уменьшения технологических норм оказались исчерпанными. Дальнейшее наращивание объемов кэш-памяти и числа запускаемых на выполнение команд не дает должного прироста производительности вычислительных систем, соотношение затрат аппаратных ресурсов и энергии к росту производительности ухудшается и наращивание тактовой частоты экономически не оправданно [3, 4]. Согласно указанному в [2]: "Для преодоления ограничений фоннеймановской архитектуры используется ряд подходов, в том числе применяются специализированные ускорители, однородные и неоднородные многоядерные структуры и программируемые массивы. Однако эффективность использования этих подходов носит ограниченный характер в силу ориентации, опять же, на основополагающие принципы фон–неймановской архитектуры. Дальнейшее усложнение архитектуры не приносит ожидаемого выигрыша в производительности."

По сравнению с традиционными подходами разрабатываемая в ИПИ РАН [5] архитектура рекуррентного вычислителя имеет два резерва повышения производительности. Первый – в сокращении количества шагов для выполнения отдельной инструкции. В фоннеймановских и потоковых архитектурах оно равно пяти (например, выборка инструкции, дешифрация, чтение данных, выполнение инструкции, запись результата). В рекуррентном вычислителе их всего три: сравнение тегов или дешифрация инструкции, выполнение инструкции и запись результата. Второй – в ориентации рекуррентного компьютера на самосинхронную схемотехнику. Цикл работы самосинхронной схемотехники определяется реальными задержками элементов при обработке данных в рамках реальных условий – напряжения питания, температуры и емкостных нагрузок. Синхронная аппаратура вынуждена ориентироваться на худшие условия и, следовательно, худшее быстродействие, поэтому время реального цикла функционально идентичной самосинхронной аппаратуры меньше (в 1,5 – 2 раза), чем цикл синхронной [6, 7].

Однако кроме перечисленных недостатков фоннеймановской организации вычислений она имеет и ряд достоинств. Так в работе [8] авторы указывают, что силой фон–неймановской архитектуры "является универсальный характер ее программной организации. Она универсальна не только в смысле машины Тьюринга, но способна эффективно поддерживать спектр стилей программирования и эффективно моделировать многообразие структур алгоритмов". В результате в ходе проводимых в ИПИ РАН работ было установлено, что предпочтительный вариант организации рекуррентного обработчика сигналов – это создание гибридной архитектуры, в рамках которой были бы использованы сильные стороны как традиционной, фоннеймановской архитектуры, так и архитектуры рекуррентного компьютера.

Для гибридной архитектуры ключевыми особенностями являются, во–первых, объединение потока команд и потока данных в единый поток. Во–вторых, управляющий уровень управляется потоком команд. Управляющий уровень реализуется в виде одного синхронного процессора средней производительности с фон–неймановской архитектурой. В–третьих, поток самодостаточных данных рекуррентно свернут на стадии компиляции и разворачивается в процессе выполнения задач при помощи преобразователя тегов. Таким образом, операционный уровень реализуется на базе рекуррентного подхода и самосинхронной схемотехники [2, 9]. При этом данная архитектура является универсальной и может использоваться в процессорах произвольного назначения.

Завершающим этапом процесса разработки рекуррентного вычислителя с потоковой архитектурой [10] является верификация проекта на физической модели или системная отладка, то есть отладка с использованием реальных аппаратных и программных средств.

Функционирование аппаратных и программных средств при этом осуществляется в реальном масштабе времени, что существенно повышает трудоемкость процесса отладки.

В настоящее время можно выделить два основных метода внутрисхемной отладки ПЛИС [11]:

– применение встроенных в ПЛИС отладочных средств на основе JTAG;

– использование внешнего контрольно–измерительного оборудования, такого как осциллографы смешанных сигналов и логические анализаторы. Использование внешнего оборудования связано со значительными материальными затратами.

В качестве элементной базы реализации сигнального процессора выбраны ПЛИС фирмы Альтера. Изначально планировалась реализация на базе семейства микросхем Excalibur [12], целесообразность чего была доказана в [13], однако фирма Альтера объявила о прекращении производства данных микросхем с рекомендуемой заменой последних семействами ПЛИС Cyclone. Семейство Cyclone V [14], а именно плата Cyclone V GX FPGA Development Board, было выбрано из них как наиболее продвинутое на момент начала завершающего этапа разработки рекуррентного вычислителя. Это отладочное средство, помимо головной ПЛИС 5CGXFC7D6F31C7NES с широкими характеристиками и максимальной емкостью до 150 тысяч логических элементов, обладает большим объемом памяти SDRAM и набором мультиплексоров. Содержит энергонезависимые ПЛИС MAX2 и MAX5 с пониженным энергопотреблением, которые обеспечивают широкий набор микросхем с разными типами корпусов, имеют массив программируемой логики, встроенный RC-генератор, пользовательскую Flash–память, встроенный линейный регулятор напряжения. Все это дает возможность снизить общую стоимость разрабатываемой системы [15, 16]. Для проектирования процессора использовалась интегрированная среда разработки Quartus II, содержащая готовые компоненты для проведения проектирования и отладки систем. Использование таких сред позволяет значительно сократить время разработки и уменьшить временные затраты на ее отладку.

Чтобы облегчить выбор средств системной отладки, необходимо разработать соответствующие критерии. Апробация новых архитектурных решений проводилась в условиях, связанных со значительными временными и материальными ограничениями, поэтому основными критериями при выборе состава отладочных средств было максимальное использование реализованных в Quartus II стандартных средств отладки и минимальное привлечение дополнительной аппаратуры.

В соответствии с выбранными критериями проведен анализ и выбор встроенных отладочных средств Quartus II на предмет их использования для отладки устройства гибридной архитектуры рекуррентного обработчика сигналов (ГАРОС). Реализованы дополнительные отладочные средства. Встроенные и дополнительно реализованные отладочные средства использовались совместно.

II. Средства отладки Quartus II

Среда разработки Quartus II имеет следующие встроенные средства отладки [17], [18]: редактор отладочных выводов (SignalProbe Pins); редактор интерфейса для внешнего логического анализатора (Logic Analyzer Interface Editor); редактор содержимого памяти в системном окружении (In-System Memory Content Editor); встраиваемый логический анализатор SignalTap II (SignalTap II Logic Analyzer); редактор исходников и пробников (In-System Sources and Probes Editor); виртуальный JTAG (Virtual JTAG); отладочные средства процессора Nios II.

Редактор отладочных выводов, использующий незанятые в проекте выводы ПЛИС для отображения внутренних сигналов проекта, может быть использован для отладки отдельных блоков проекта из-за простоты использования, отсутствия дополнительных аппаратных затрат, сохранения логического ресурса ПЛИС. Однако для комплексной отладки ГАРОС он не подходит ввиду того, что не имеет доступа к содержимому памяти и потребуется задействовать дополнительные выводы ПЛИС.

Редактор интерфейса для внешнего логического анализатора позволяет при помощи набора управляемых мультиплексоров отобрать на вход внешнего логического анализатора нужные сигналы анализируемого проекта. Этот интерфейс позволяет решить проблему недостаточности внешних выводов, присутствующую в средстве SignalProbe Pins, ценой аппаратных затрат на реализацию интерфейса и привлечения внешнего оборудования. Привлечение дорогостоящего внешнего логического анализатора не отвечает критериям выбора средств отладки сигнального процессора, к тому же использование внешнего анализатора может повлечь изменение характеристик пользовательского проекта. Поэтому для отладки ГАРОС этот редактор не подходит.

Редактор содержимого памяти в системном окружении позволяет анализировать данные в блоках памяти ПЛИС, что может существенно упростить отладку ГАРОС. При помощи соответствующего конфигурирования редактор позволяет выбирать нужную область памяти по именам или характеристикам, читать оттуда данные, отображать их пользователю и даже редактировать. Редактор содержимого памяти не требует дополнительной аппаратуры и не использует ресурсы ПЛИС, что определило его эффективное использование при отладке ГАРОС.

Встраиваемый логический анализатор SignalTap II позволяет выбирать сигналы проекта для наблюдения, осуществлять запись логических состояний сигналов, осуществлять совместную работу с САПР Quartus II, подключаясь к нему через JTAG интерфейс, захватывать наблюдаемые сигналы в реальном времени на частотах свыше 300 МГц., дает возможность событий-

ной отладки. Он позволяет создать сложную систему условий и управлять ей, внося в процессе отладки минимальные искажения в наблюдаемые сигналы и дает возможность анализа изменений сигналов. Ввиду мощности и простоты данное средство было выбрано основным для отладки ГАРОС.

Редактор исходников и пробников состоит из мегафункции altsource_probe и интерфейсной графической оболочки, которая позволяет контролировать все элементы этой мегафункции внутри отлаживаемого проекта в реальном времени. Каждый элемент мегафункции дает для отображения исходные выходные порты и отводы входных портов. Интерфейсная графическая оболочка показывает все доступные в реальном времени контролируемые элементы и позволяет создать виртуальный пульт отладки. При помощи такого пульта можно управлять отладкой сигнального процессора формируя набор виртуальных кнопок и индикаторов, что позволяет подавать различные сигналы на входы схем сигнального процессора, следить за состоянием выбранных выходов и проводить ручное тестирование отдельных схем сигнального процессора.

Такой редактор не требует внешних устройств и в сочетании с встроенным логическим анализатором и редактором памяти дает наибольшую свободу в контроле над сигналами и создании виртуальных входов, что позволяет сократить время верификации проекта. Это определяет выбор данного средства для отладки ГАРОС.

Интерфейс виртуального JTAG обеспечивает обмен данными между платой и компьютером через кабель USB_blaster. Управление интерфейсом VJTAG осуществляется программой quartus_stp, входящей в состав САПР Quartus II, при помощи сценариев на языке Tcl. Конечные устройства (контроллеры памяти, устройства управления светодиодами и другие) обмениваются данными через декодер с мегафункцией VJTAG [19]. Работая с отладочной схемой, использующей VJTAG, можно выделить три типа фрагментов этой схемы: встроенные в ПЛИС (порождающие некоторые ограничения по использованию мегафункций), обязательные компоненты VJTAG (не допускающие вариаций) и требующие ручного управления и создания пользователем [20]. С одной стороны, это одно из наиболее мощных встроенных средств отладки, гибкое, позволяющее создавать схемы, зависящие от конкретных требований. Но из-за необходимости создания дополнительных аппаратных затрат и повышенной трудоемкости их проектирования виртуальный JTAG не отвечает критерию выбора отладочных средств и его использование при отладке ГАРОС нецелесообразно. Основной проблемой при потенциальном использовании данного средства является то, что фирма Альтера не раскрывает интерфейс, по которому VJTAG общается с компьютером.

Дополнительные отладочные средства в Quartus II появляются в случае использования процессора Nios II, отладочные средства которого построены на основе модуля JTAG.

Программные инструменты отладки связываются с отладочным модулем JTAG и предлагают следующие средства: загрузка программы в память; пуск и остановка исполнения программы; установка программных и аппаратных точек останова и точек просмотра; анализ регистров и памяти процессора; накопление следов (данных) исполнения программ в реальном времени.

Серьезно расширяет возможности отладки с помощью встроенного логического анализатора SignalTap II плагин Nios II. Он дополняет отладочные средства SignalTap II, позволяя захватывать коды операций процессора Nios II. Он разрешает запоминать инструкции, а также анализировать данные, доступные процессору Nios II. Можно задать триггер инструкции, который позволяет защелкивать встроенный логический анализатор SignalTap II, когда процессор обращается по определенному адресу. Плагин Nios II автоматически позволяет работать с символьными именами или прямыми адресами программы.

Отладочные средства процессора Nios II построены на основе отладочного модуля JTAG. В настоящее время такое построение традиционно для реализации средств отладки микроконтроллеров и микропроцессоров. Архитектура средств отладки Nios II поддерживает отладочный модуль JTAG, который обеспечивает управление процессом отладки программ с ПК.

В процессе разработки системы на основе Nios II можно варьировать состав средств отладки, используя обилие средств отладочного ядра, и удаляя потом отдельные компоненты, чтобы сэкономить на логических ресурсах. После завершения системной отладки отладочный модуль JTAG может быть уменьшен функционально или полностью удален.

Расположенные на компьютере программные инструменты отладки связываются с отладочным модулем JTAG и предлагают следующие средства:

– загрузка программы в память;

– пуск и остановка исполнения программы;

– установка программных и аппаратных точек останова и точек просмотра;

– анализ регистров и памяти процессора;

– накопление следов (данных) исполнения программ в реальном времени.

Загрузка программы связана со средством загрузки исполняемого кода и данных в память процессора через соединение JTAG. После загрузки программы в память отладочный модуль JTAG может выйти из режима отладки и переместить исполнение на начало исполняемого кода.

В целом процессор Nios II является мощным инструментом отладки, требующим, однако, больших затрат на подготовку и программирование и, к сожалению, не дающий полностью доступа ко всем входам и выходам нашего сигнального процессора, в связи с чем

использование процессора Nios II должно рассматриваться только как дополнительное.

III. Отладка ГАРОС

Фирма Альтера не раскрывает протокол взаимодействия с отладочной платой. Поэтому разработчик вынужден использовать средства, встроенные в Quartus, или проектировать свои (уникальные) отладочные средства, например, на основе виртуального JTAG. Использование компромиссного варианта (сочетание встроенных и дополнительно спроектированных элементов) позволяет оптимизировать временные и финансовые затраты на реализацию средств отладки.

Разработанный рекуррентный вычислитель ГАРОС реализован в виде гибридного двухуровневого варианта с ведущим фон–неймановским процессором на верхнем управляющем уровне (УУ) и рядом потоковых процессоров на нижнем уровне – рекуррентном операционном устройстве (РОУ). Управляющий уровень реализован в виде программы на основе процессора Nios II. РОУ представляет собой VHDL–описания для синтеза на ПЛИС Cyclone V.

Обмен данными между этими уровнями осуществляется через буферную память (БП). Для упрощения процесса изложения материала буферная память отнесена к РОУ.

Структура аппаратных средств ГАРОС с учетом его реализации на основе платы Cyclone V GX FPGA Development Board фирмы Альтера изображена на рис. 1.

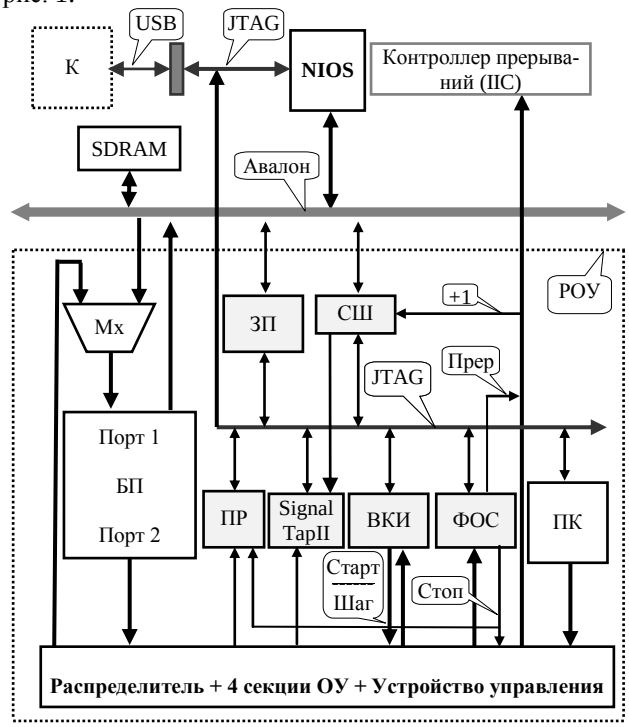

Рис. 1. Структура аппаратных средств ГАРОС

УУ состоит из следующих основных модулей: вычислительного ядра Nios II с внутренним контроллером прерываний (IIC); системной шины Авалон; памяти процессора Nios II.

В состав РОУ входят следующие функциональные модули: двухпортовая буферная память (БП); мультиплексор порта 1 записи в БП (Мх); память констант (ПК); распределитель; 4 секции обрабатывающего устройства (ОУ); устройство управления; средства отладки.

Все аппаратные средства ГАРОС взаимодействуют с персональным компьютером (К) через интерфейс USB 2.0.

Для обеспечения доступа ко всем функциональным элементам ГАРОС используется стандартный интерфейс диагностики и отладки аппаратных и программных средств JTAG, встроенный логический анализатор SignalTap II и редактор содержимого памяти в системном окружении.

В свою очередь, дополнительно разработанные средства отладки РОУ состоят из следующих функциональных модулей: зеркальной памяти (ЗП); счетчика шагов (СШ); памяти содержимого регистров (ПР); виртуальных клавиш и индикаторов (ВКИ); формирователя отладочных событий (ФОС).

Встроенные средства отладки САПР Quartus II не обеспечивают возможность чтения и редактирования содержимого оперативной двухпортовой памяти. В нашем случае БП является двухпортовой. Поэтому в состав РОУ введена однопортовая ЗП, отображающая содержимое БП. ЗП по сравнению с БП невелика по объему и является окном в БП в диапазоне заданных адресов. Содержимое ячеек ЗП можно редактировать с помощью редактора памяти. Перемещение данных между БП и ЗП осуществляется программным способом. Диапазон адресов (окно) задается непосредственно в ЗП с помощью редактора памяти. Инициализация осуществляется виртуальными кнопками в окне виртуального пульта отладки. Таким образом, имеется возможность перемещать данные между ЗП и БП. На виртуальном окне отладки указывается нужный диапазон адресов в БП и дается команда на заполнение ЗП или перенос данных из ЗП в БП.

Зеркальная память используется в режиме отладки для просмотра содержимого адресного окна карты памяти и загрузки капсул в буферную память. В обычном режиме ее можно использовать для хранения программ Nios II, критичных по времени выполнения. Зеркальная память состоит из адресного окна карты памяти и окна условий отладки РОУ.

В окне условий отладки РОУ кроме начального и конечного адреса карты окна памяти задаются также другие условия отладки. Операции чтения и записи содержимого ЗП и выполнение условий отладки осуществляются программой управляющего уровня. Содержимое ЗП доступно приложению Редактор содержимого памяти, что обеспечивает возможность про-

смотра и редактирования ее содержимого. Таким образом, можно осуществлять изменение содержимого любой ячейки памяти, расположенной в адресном пространстве Nios (в том числе и БП).

Счетчик шагов исполнения РОУ предназначен для осуществления контроля за процессом отладки и анализа исполняемой капсулы. Особенно актуален СШ для пошаговой отладки.

Необходимость просмотра внутренних регистров РОУ в процессе отладки для контроля за ходом процесса вычислений обеспечивается дополнительно введенной памятью содержимого внутренних регистров. Программные способы чтения содержимого регистров искажают (нарушают) процесс исполнения капсулы, поэтому реализован аппаратный способ чтения содержимого регистров.

Для этого в процессе отладки содержимое регистров РОУ копируется в ПР аппаратным способом по команде при очередном шаге или по отладочному событию. Поскольку содержимое ПР доступно приложению Редактор содержимого памяти, то это обеспечивает возможность наблюдения за содержимым регистров РОУ стандартными средствами Quartus II.

Приложение SignalTap II имеет расширенные возможности и позволяет разработчику создавать и встраивать в РОУ определенное число логических анализаторов, оперативно изменять условия фиксации данных в их памяти и отображать эти данные на экране компьютера. Особенность этого подхода состоит в исследовании поведения внутренних сигналов без использования дополнительных контактов ввода–вывода и какого–либо внешнего оборудования. Можно подсоединять его входы к различным точкам схемы РОУ, задавать условия фиксации сигналов и далее просматривать временные диаграммы.

Управление процессом отладки РОУ осуществляется с виртуального пульта. Для этого с помощью редактора источников и пробников сформирован набор виртуальных кнопок и индикаторов – виртуальный пульт. В процессе отладки с него можно подавать различные сигналы на входы схем РОУ и следить за состоянием выбранных выходов схем. С помощью виртуального пульта можно проводить ручное тестирование отдельных схем РОУ.

Архитектура РОУ существенно отличается от традиционной архитектуры, поэтому процесс отладки носит событийный характер. Для обеспечения возможности настройки и управления процессом отладки РОУ используется формирователь отладочных событий. Посредством редактора содержимого памяти и редактора источников и пробников формируется набор отладочных событий. В процессе отладки, например, при обнаружении отладочной ситуации, ФОС останавливает функционирование РОУ и сообщает УУ о ее возникновении. Набор событий в ФОС обеспечивает трассировку процесса исполнения капсул в РОУ.

Рассмотрим процесс отладки РОУ с использованием модулей ЗП, СШ, ПР и ФОС. Отладка РОУ может осуществляться в пошаговом или автоматическом режимах. Мы рассмотрим отладку в автоматическом режиме. В любом случае для отладки необходимо сформировать виртуальный пульт с набором кнопок и индикаторов с помощью мегафункции altsource_probe. Для работы необходимы, как минимум, следующие кнопки: обнуление РОУ (RST); пуск (Start); режим автомат/шаг (A/S); обнуление СШ (RC); запись капсулы в БП (WB). Непосредственное формирование пульта осуществляется в окне приложения Редактор источников и пробников.

Понадобится также логический анализатор SignalTap II. В окне STP редактора необходимо настроить логический анализатор и сформировать условия захвата нужных данных.

Далее виртуальной кнопкой WB осуществляется загрузка в БП капсулы, находящейся в ЗП, и выбор автоматического режима работы РОУ (кнопкой A/S). Формирование отладочных событий осуществляется в окне приложения Редактор содержимого памяти. РОУ приводится в начальное состояние кнопками RST и затем RC. Запуск на исполнение капсулы осуществляется нажатием кнопки Start.

Если в результате исполнения капсулы возникло отладочное событие, то ФОС посылает сигнал Стоп в устройство управления РОУ и сигнал Прер на вход контроллера прерываний Nios II (рис. 1). Эти сигналы вызывают останов исполнения капсулы в РОУ, запись содержимого регистров в ПР и вызов отладочной процедуры в УУ. Можно осуществить анализ процесса исполнения капсулы и изменение состояния ячеек памяти путем вызова соответствующих средств Quartus II. Возобновление процесса активируется кнопкой Start на виртуальном пульте.

Представленный набор отладочных средств обеспечивает возможность быстрой совместной отладки аппаратных и программных средств РОУ.

IV. Выводы

1. Процесс отладки рекуррентного вычислителя показал, что встроенные средства системной отладки САПР Quartus II удобны в использовании и позволяют существенно упростить процесс верификации проектов на основе ПЛИС в реальном аппаратном окружении. В соответствии с выбранными критериями встроенные средства отладки использовались максимальным образом и позволили решить большинство задач. Однако оказалось, что для отладки рекуррентного вычислителя в рамках значительных временных и материальных ограничений одних средств системной отладки САПР Quartus II недостаточно.

2. В результате в ГАРОС были реализованы дополнительные отладочные средства, такие как зеркальная память, счетчик шагов, память содержимого регистров, виртуальные клавиши и индикаторы, формирователь отладочных событий. В сочетании со встроенны-

ми средствами системной отладки Quartus II их оказалось достаточно для поддержки эффективного процесса совместной отладки аппаратных и программных средств. В соответствии с выбранными критериями отладки внешняя дополнительная аппаратура не использовалась. Предположение о том, что использование компромиссного варианта (сочетания встроенных и дополнительно спроектированных, уникальных средств отладки) позволит оптимизировать временные и финансовые затраты на реализацию средств отладки, получило свое подтверждение.

3. Принципы, лежащие в основе дополнительно разработанных средств отладки РОУ, могут использоваться не только в рамках ГАРОС, но и для отладки устройств в рамках традиционной архитектуры.

Поддержка

Работа выполнена при частичной финансовой поддержке по Программам фундаментальных исследований ОНИТ РАН за 2013 г. (проект 1.5) и Президиума РАН (проект 16) и РФФИ в рамках научного проекта № 13-07-12068 офи_м.

Литература

[1] Волчек В.Н., Степченков Ю.А., Петрухин В.С., Прокофьев А.А., Зеленов Р.А. Цифровой сигнальный процессор с нетрадиционной рекуррентной потоковой архитектурой // Проблемы разработки перспективных микро- и наноэлектронных систем - 2010. Сборник трудов / под общ. ред. академика А.Л.Стемпковского. М.: ИППМ РАН, 2010. С. 412–417.

[2] Степченков Ю.А., Петрухин В.С. Особенности гибридного варианта реализации на ПЛИС рекуррентного обработчика сигналов // Системы и средства информатики. Доп. вып. М.: ИПИ РАН, 2008. С. 118–129.

[3] Волков Дмитрий. Реальность и фантазии // Открытые системы №05. 2006. 1 с. URL: http://www.osp.ru/os/ 2006 /05/2449740/ (дата обращения: 08.02.2014).

[4] Леонид Черняк. Микропроцессоры: все только начинается // Открытые системы. 2006. №05. 28 с. URL: http://www.osp.ru/os/2006/05/2449832/ (дата обращения: 08.02.2014).

[5] Степченков Ю.А., Петрухин В.С., Филин А.В. Рекуррентное операционное устройство для процессоров обработки сигналов // Сборник Системы и средства информатики. М.: Наука, 2001. Вып. 11. С. 283-315.

[6] Степченков Ю.А., Дьяченко Ю.Г., Петрухин В.С., Филин А.В. Цена реализации уникальных свойств самосинхронных схем // Сборник "Системы и средства информатики". М.: Наука, 1999. Вып. 9. С. 261-292.

[7] P. Beerel, J. Cortadella, and A. Kondratyev Bridging the gap between asynchronous design and designers (Tutorial) // in VLSI Design Conference, (Mumbai). 2004.

[8] Anant Agarwal, Ben-Hong Lim, David Kranz, and John Kubiatowicz APRIL: A processor architecture for multiprocessing // Proc. 17th Annual Intl. Symp. on Computer Architecture. Seattle, Washington, U.S.A. May 28-31 1990. P. 104-114. URL: http://groups.csail.mit.edu/cag/papers/pdf/isca-april.pdf pdf (дата обращения: 29.01.2014).

[9] Волчек В.Н., Зеленов Р.А., Прокофьев А.А. Средство автоматизированного тестирования вычислительного блока рекуррентного операционного устройства // Проблемы разработки перспективных микро- и наноэлектронных систем - 2012. Сборник трудов / под общ. ред. академика А.Л.Стемпковского. М.: ИППМ РАН, 2012. С. 137–142.

[10] Шнейдер А.Ю., Петрухин В.С., Степченков Ю.А. Принципы построения средств отладки рекуррентного вычислителя // Проблемы разработки перспективных микро- и наноэлектронных систем - 2012. Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2012. С. 133–136. URL:http://www.mes-conference.ru/data/year2012/pdf/ D169.pdf (дата обращения: 28.01.2014).

[11] Упрощение отладки ПЛИС Xilinx и Altera // Каталог оборудования 2012-2013. Решения в области контрольно-измерительной аппаратуры. С. 386–395. URL:http://www.tehencom.com/Companies/Tektronix/Tektronix_Catalog_2012_Rus.pdf (дата обращения: 28.01.2014).

[12] About Excalibur Embedded Processor solutions.. URL: http://www.altera.com/products/devices/excalibur/exc-index.html/ (дата обращения: 27.02.2014).

[13] Степченков Ю.А., Петрухин В.С. Перспективы развития потоковых сигнальных процессоров и возможная реализация рекуррентного обработчика сигналов // Сборник "Методы и средства разработки информационно-вычислительных систем и сетей" (специальный выпуск). М.: Наука, 2004. С. 89-133.

[14] Cyclone V GX FPGA Development Board Reference Manual. URL: http://www.altera.com/literature/manual/rm_cvgx_fpga_dev_board.pdf (дата обращения: 05.12.2013).

[15] Петрухин В.С., Волчек В.Н., Прокофьев А.А., Зеленов Р.А. Особенности реализации рекуррентного обработчика сигналов с гибридной архитектурой // Системы и средства информатики. Доп. вып. М.: ИПИ РАН, 2008. С. 130–148.

[16] Петрухин В.С., Хилько Д.В. Выбор языковых средств представления параллельных алгоритмов для рекуррентного обработчика сигналов // Системы и средства информатики. Доп. вып. М.: ИПИ РАН, 2008. С. 149–158.

[17] Quartus II Handbook Version 12.0. Volume 3: Verification. Altera. 101 Innovation Drive San Jose, CA 95134. www.altera.com. QII5V3-12.0.0. URL: http://www.altera.com/literature/hb/qts/quartusii_handbook.pdf (дата обращения: 28.01.2014).

[18] Антонов Александр, Филиппов Алексей, Золотухо Роман. Средства системной отладки САПР Quartus II // Компоненты и технологии. 2008. № 12. — URL: http://kit-e.ru/articles/plis/2008_12_53.php (дата обращения: 05.12.2013).

[19] Гребенников А. Интерфейс VJTAG для отладочной платы DK-START-3C25N // Современная Электроника. 2010. № 9. – URL: http://www.soel.ru/issues/?id=343889 (дата обращения: 05.12.2013).

[20] Грушвицкий Р., Михайлов М. Проектирование в условия временных ограничений: отладка проектов // Компоненты и технологии. 2007. № 9. – URL: http://kit-e.ru/articles/plis/2007_09_133.php (дата обращения: 05.12.2013).