УДК 004.43 DOI: 10.14529/cmse150202

СРЕДСТВА ПРОГРАММИРОВАНИЯ РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ НА ОСНОВЕ ПЛИС VIRTEX-7 С ИСПОЛЬЗОВАНИЕМ СОФТ-АРХИТЕКТУР[1]

И.И. Левин, А.И. Дордопуло, В.Б. Коваленко, В.А. Гудков, А.А. Гуленок

В статье рассматриваются существующие средства проектирования цифровых устройств в программируемых логических интегральных схемах (ПЛИС), языки программирования реконфигурируемых вычислительных систем и возможность их использования при программировании многокристальных реконфигурируемых вычислительных систем. Также рассмотрены разработанные в НИИ МВС ЮФУ язык программирования высокого уровня COLAMO и комплекс средств разработки многокристальных решений на реконфигурируемых вычислительных системах. Особое внимание уделено новому подходу в программировании, заключающемуся в разработке и использовании настраиваемых проблемно-ориентированных софт-архитектур, которые позволяют сократить количество трансляций конфигурационных файлов ПЛИС при отладке параллельных программ на реконфигурируемых вычислительных системах. Проблемно-ориентированные софт-архитектуры дают возможность без перезагрузки файлов конфигурации ПЛИС вычислительного поля с помощью программной настройки изменять коммутацию между устройствами и создавать необходимые вычислительные структуры для решения прикладных задач пользователя, что существенно сокращает время отладки параллельных прикладных программ.

Ключевые слова: реконфигурируемые вычислительные системы, параллельное программирование, софт-архитектура, программирование РВС, проблемно-ориентированные софт-архитектуры.

Введение

За последние годы значительно вырос интерес к программируемым логическим интегральным схемам. Различные отечественные и зарубежные компании создают как отдельные ускорители с одним-двумя кристаллами программируемых логических интегральных схем (ПЛИС), так и целые вычислительные комплексы. Реконфигурируемые вычислительные системы (РВС) на базе ПЛИС показали свою эффективность при решении вычислительно трудоемких задач из различных областей науки и техники. Такие фирмы как Nallatech [1] и Pico Computing [2] выпускают ряд ускорителей и несущих плат с небольшим числом (до четырех) кристаллов ПЛИС, которые используются в создании серверов и гибридных кластерных систем фирмами HP и IBM. Компании Convey [3] и Maxeler Technologies [4] создают гибридные суперкомпьютеры на основе собственных гетерогенных кластерных узлов, каждый из которых может содержать от 1 до 4 кристаллов ПЛИС и несколько универсальных процессоров. Похожее решение используется и компанией SRC [5], которая выпускает узлы, названные MAP processor, для стойки (MAPstation) форм-фактором 1U, 2U и 4U. MAPstation 1U содержит один MAP processor. MAPstation 2U содержит до трех MAP processor. MAPstation 4U может содержать до 10 различных модулей — MAP

[1] Статья рекомендована для публикации Программным комитетом конференции «Научный сервис в сети Интернет — 2014»

processor, модуль с универсальным микропроцессором или модуль памяти. При этом наиболее рациональным способом построения реконфигурируемой вычислительной системы является использование в качестве основного вычислительного элемента множества кристаллов ПЛИС, объединенных в единое вычислительное поле высокоскоростными каналами передачи данных, что позволяет получить мощный вычислительный ресурс, достаточный для эффективного решения задачи.

Статья организована следующим образом. В первом разделе рассмотрены существующие в настоящее время системы проектирования, позволяющие создавать структурные решения прикладных задач на ПЛИС, отмечен их общий основной недостаток — невозможность автоматизированного создания схемотехнических решений для РВС, содержащих множество кристаллов ПЛИС, соединенных между собой пространственной коммутационной системой.

Решение этой проблемы возможно при использовании языка программирования высокого уровня с неявным описанием параллелизма COLAMO, краткое описание основных принципов которого представлено во втором разделе.

Третий раздел содержит описание концепции программирования многокристальных РВС с помощью софт-архитектур, которая позволяет без перезагрузки файлов конфигурации ПЛИС РВС с помощью программной настройки изменять коммутацию между устройствами и создавать необходимые вычислительные структуры для решения прикладных задач пользователя, что позволяет существенно сократить время отладки прикладной программы.

В заключении представлены практические результаты, полученные при решении задачи обработки спекл-изображений по методу Лабейри с помощью предложенной концепции софт-архитектур.

1. Существующие средства программирования РВС

Наиболее популярными как в качестве отдельного средства разработки, так и в составе комплексов являются программы-синтезаторы, предоставляемые фирмами-изготовителями ПЛИС: ISE и Vivado фирмы Xilinx [6], Quartus II фирмы Altera [7] и Actel Libero IDE фирмы Actel [8]. Данные средства, помимо непосредственно среды проектирования цифровых устройств, включают в себя ряд вспомогательных утилит: анализаторы временных характеристик, редакторы размещения, модули программирования ПЛИС, системы моделирования цифровых устройств и др. Благодаря широкому инструментарию эти системы проектирования обеспечивают полный цикл разработки цифровых устройств в пределах одного кристалла ПЛИС: создание исходного описания проекта, синтез, моделирование, размещение, трассировку, конфигурирование кристалла.

В связи с постоянным ростом емкости кристаллов ПЛИС проектирование решений прикладных задач в кристаллах ПЛИС с помощью языков описания аппаратуры (VHDL, AHDL, Verilog и др.), а также разработка цифровых устройств в графических редакторах становится все более трудоемкой. Поэтому в настоящее время ведущие производители ПЛИС и реконфигурируемых вычислителей ориентируются на языки высокого уровня. Так, в новой среде разработки Vivado фирмы Xilinx добавлен новый инструмент проектирования Vivado HLS [9], основанный на языке высокого уровня, а ком-

пания Altera для своих ПЛИС предлагает инструментарий разработки Altera SDK [10] для нового стандарта параллельного программирования гетерогенных систем OpenCL. В данных решениях используются трансляторы C-подобных языков, генерирующие из программы на аналоге языка программирования высокого уровня C код на языках описания аппаратуры на уровне регистровых передач (RTL-уровень, трансляторы C-to-RTL).

Несмотря на схожесть синтаксисов данных C-подобных языков с самим языком C, подобный подход вовсе не означает, что исходный код на C, написанный под персональный компьютер или кластерную вычислительную систему, будет понят трансляторами C-to-RTL. Выбор языка C в качестве основы обусловлен широкой распространенностью данного языка, что существенно упрощает освоение новых инструментариев разработки решений прикладных задач для ПЛИС.

Также при использовании трансляторов C-to-RTL весь код программы либо явно указанные процедуры транслируются в RTL-описания отдельных кристаллов ПЛИС. В подобных системах разработки отсутствует инструментарий, обеспечивающий автоматические разбиения параллельной программы на множество связанных кристаллов ПЛИС.

В Vivado HLS проект разрабатывается в рамках одного кристалла ПЛИС, и если программисту недостаточно аппаратного ресурса кристалла, то он вынужден самостоятельно распределять вычисления между несколькими проектами для каждой ПЛИС и синхронизировать управляющие и информационные потоки между ними.

Стандарт OpenCL используется компанией Nallatech (разработчиком реконфигурируемых вычислителей) и подразумевает использование нескольких ПЛИС в одном проекте. Программирование решений в кристаллах ПЛИС в данном случае осуществляется с помощью вызова функций из библиотек инструментария разработки Altera SDK, и в каждом задействованном кристалле ПЛИС выполняются вычисления, описанные отдельным участком кода. Таким образом, программа, написанная в стандарте OpenCL, представляет собой основной код, написанный под традиционные процессоры, и отдельные участки кода, написанные под ПЛИС, задействованные как сопроцессоры. В данном случае задача синхронизации данных также возлагается на самого программиста.

Еще одним известным средством программирования ПЛИС является комплекс, разработанный компанией Mitrionics, состоящий из виртуального процессора Mitrion Virtual Processor (MVP), программирование которого осуществляется на языке высокого уровня Mitrion-C, и библиотеки функций для построения хост-программ MithalAPI, входящих в пакет разработки Mitrion SDK [6].

Mitrion-C является специализированным языком высокого уровня семейства C, но работает с потоком данных, а не с потоком команд, реализуя параллелизм на уровне инструкций и циклов. Параллелизм на уровне инструкций является неявным, поскольку анализ зависимостей данных происходит автоматически на этапе компиляции программы. Порядок выполнения инструкций в языке не определен и зависит от степени готовности данных для каждой конструкции. Таким образом, параллелизм на этом уровне обеспечивается независимостью данных.

Основным средством распараллеливания на языке Mitrion-C является распараллеливание на уровне циклов. Параллелизм на уровне циклов является явным и определяется ключевым словом «Foreach», что дает возможность программисту самому определять, какие циклы он хочет распараллеливать.

Наличие неявного параллелизма на уровне инструкций требуют от пользователя особого внимания при разработке параллельной программы, а явное указание распараллеливаемых циклов может приводить к значительным трудностям при модификации параллельной программы или ее переносимости на другие архитектуры ПЛИС.

В целом язык Mitrion-C обеспечивает пользователя достаточно простыми средствами распараллеливания, но предъявляет серьезные требования к разработке параллельного алгоритма и упрощает процесс разработки параллельных программ незначительно.

Разрабатываемая на языке Mitrion-C программа должна быть полностью реализована на одном виртуальном процессоре MVP, что не позволяет программировать многокристальные РВС, а следовательно, существенно снижает эффективность использования программного комплекса компании Mitrionics. Для программирования многокристальных РВС со связями между ПЛИС программисту необходимо самому реализовать интерфейс (протокол) обмена данными между ПЛИС и решить вопросы, связанные с синхронизацией потоков данных. В этом случае программа для РВС вырождается в программу для кластера (множество процессоров MVP), реализованного на ПЛИС, что существенно снижает эффективность реализации задач на многокристальных РВС.

2. Язык COLAMO и комплекс средств программирования многокристальных РВС

Альтернативный подход к программированию РВС предлагается в Научно-исследовательском институте многопроцессорных вычислительных систем Южного федерального университета (НИИ МВС ЮФУ), занимающемся разработкой многокристальных реконфигурируемых вычислительных систем различной архитектуры и конфигурации уже более 15 лет.

Опыт решения различных классов задач в НИИ МВС ЮФУ показал, что для эффективного решения современных трудоемких задач программисту необходимы средства программирования, обеспечивающие следующие основные возможности:

- программирование на языке высокого уровня;
- поддержка многокристального программирования;
- обеспечение высокой частоты работы ПЛИС;
- высокая плотность заполнения кристалла ПЛИС;
- поддержка конвейерной и макроконвейерной организации вычислений.

В НИИ МВС ЮФУ разработан и широко используется программный комплекс, включающий в себя следующие компоненты:

- *транслятор языка программирования COLAMO*, осуществляющий трансляцию исходного кода на COLAMO в информационный граф параллельной прикладной программы;
- *синтезатор масштабируемых схемотехнических решений* на уровне логических ячеек ПЛИС Fire!Constructor, осуществляющий отображение полученного от транслятора языка программирования COLAMO информационного графа на архитектуру РВС, размещение отображенного решения по кристаллам ПЛИС и автоматическую синхронизацию фрагментов информационного графа в разных кристаллах ПЛИС;
- *библиотеку IP-ядер*, соответствующих операторам языка COLAMO (функционально-законченных структурно-реализованных аппаратных устройств) для раз-

личных предметных областей, и интерфейсов для согласования скорости обработки информации и связи в единую вычислительную структуру;

– *средства отладки* программ, *средства доступа и мониторинга* состояния РВС.

Язык высокого уровня COLAMO предназначен для описания реализации параллельного алгоритма и создания на основе принципов структурно-процедурной организации вычислений [11] в архитектуре РВС специализированной вычислительной структуры, которая предполагает последовательную смену структурно (аппаратно) реализованных фрагментов информационного графа задачи, каждый из которых является вычислительным конвейером потока операндов. Таким образом, приложение (прикладная задача) для РВС состоит из структурной составляющей, представленной в виде аппаратно реализованных фрагментов вычислений, и процедурной составляющей, представляющей собой единую для всех структурных фрагментов управляющую программу последовательной смены вычислительных структур и организации потоков данных. Для представления такой организации вычислений в языке используется понятие «кадр». Кадром является программно-неделимый компонент, представляющий собой совокупность операторов, которые реализуются в виде арифметико-логических команд и команд чтения/записи, выполняемых на различных функциональных устройствах, соединенных между собой в соответствии с информационной структурой алгоритма.

В языке COLAMO отсутствуют явные формы описания параллелизма, а распараллеливание достигается с помощью объявления типов доступа к переменным и индексации элементов массивов, что характерно для языков потока данных. Для обращения к данным используется два основных метода доступа: параллельный доступ (задаваемый типом Vector) и последовательный доступ (задаваемый типом Stream). Степень параллелизма определяется по минимальному значению параметра распараллеливания. Для типа доступа Stream степень параллелизма равна 1, а для типа доступа Vector определяется наименьшим значением векторной составляющей каждого массива, участвующего в вычислениях. Для параллельного типа доступа возможна одновременная обработка всех размерностей массивов, заданных типом Vector, при этом повышается аппаратный ресурс на обработку, но снижается время обработки.

Многомерные массивы данных могут иметь множество измерений, каждое из которых может иметь последовательный или параллельный тип доступа, задаваемый ключевым словом Stream или Vector соответственно. Изменение типа доступа позволяет достаточно просто управлять как степенью распараллеливания вычислений на уровне описания структур данных, так и скоростью обработки и занимаемым ресурсом, что позволяет программисту описывать различные виды параллелизма в достаточно сжатом виде.

Помимо типа доступа, для переменной в языке COLAMO определены также и типы ее хранения: мемориальный (Mem), регистровый (Reg) и коммутационный (Com).

Мемориальной переменной называется величина, хранящаяся в ячейке распределенной памяти и, следовательно, сохраняющая свое значение до очередного переприсваивания. Для мемориальной переменной возможно одновременное выполнение только одного процесса. Поэтому в семантике языка COLAMO для мемориальной переменной в теле кадра действуют правило однократного присваивания и правило единственной подстановки. Правило однократного присваивания указывает на то, что мемориальная переменная в кадре изменяет свое значение только один раз. Правило единственной подста-

новки определяет, что переменная в кадре может использоваться только для одного процесса чтения или записи.

Для описания связей между элементами информационного графа задачи в языке COLAMO предназначена коммутационная переменная. Поскольку коммутационная переменная описывает информационные связи, то она не требует никакого вычислительного аппаратного ресурса для своей реализации. Доступ к значению коммутационной переменной после выполнения кадра невозможен. Коммутационная переменная необходима транслятору для указания информационных зависимостей при построении вычислительной структуры задачи. На коммутационную переменную, как и на мемориальную переменную, действует правило однократного присваивания, но для нее не выполняется правило единственной подстановки. Использование коммутационных переменных позволяет легко разветвлять и дублировать потоки данных, но не позволяет реализовать рекурсию.

Для организации рекурсии в языке COLAMO используется регистровая переменная, которая представляет собой регистр на аппаратном уровне и используется для хранения промежуточных данных, полученных в процессе вычислений. Единственным ограничением для регистровой переменной в теле кадра является правило однократного присваивания.

Трансляция программы на языке высокого уровня COLAMO состоит в создании схемотехнической конфигурации вычислительной системы (структурной составляющей) и параллельной программы, управляющей потоками данных (потоковой и процедурной составляющих) [12]. Создание структурной составляющей заключается в построении вычислительного графа, соответствующего описанным на COLAMO информационным зависимостям между результатами вычислений. При этом для каждой используемой в тексте программы операции подставляется специализированный вычислительный блок в зависимости от типа доступа к переменным, типа данных, их разрядности и т.д. Синтезированный информационный граф задачи передается в синтезатор Fire!Constructor для отображения на аппаратный ресурс многокристальной РВС [13].

Задача автоматического отображения параллельной программы на аппаратный ресурс многокристальной РВС состоит из трех подзадач: разбиения информационного графа на непересекающиеся подграфы, размещения сформированных подграфов в ПЛИС РВС и трассировки внешних связей размещенных подграфов в системе коммутаций РВС.

Результатом работы синтезатора Fire!Constructor являются файлы VHDL-описаний и файлы временных и топографических ограничений (User Constraints Files). Файлы VHDL описывают структурные реализации фрагментов параллельной программы. На основании этих файлов и библиотеки схемотехнических элементов формируются проекты для синтезатора ISE под каждую отдельную ПЛИС. Далее с помощью синтезатора ISE формируются конфигурационные файлы ПЛИС. Полученные конфигурационные файлы загружаются в РВС.

Программа на языке COLAMO разрабатывается в едином проекте и может быть транслирована на любую РВС, описание которой и соответствующие библиотеки присутствуют в комплексе программирования РВС. В отличие от других существующих систем разработки решений прикладных задач на ПЛИС пользователю не требуется в тексте программы указывать, какие фрагменты программы в каких ПЛИС будут вы-

полняться. Синтезатор Fire!Constructor обеспечивает автоматическое разбиение вычислительной структуры программы на языке COLAMO на несколько проектов в синтезаторе Xilinx ISE, при этом обеспечивается синхронизация информационных потоков как внутри кристалла ПЛИС, так и между ними.

3. Использование софт-архитектур для программирования РВС на базе ПЛИС VIRTEX-7

С ростом емкости кристаллов ПЛИС растет и время синтеза конфигурационных файлов для загрузки в ПЛИС. Современные ПЛИС фирмы Xilinx семейства Virtex-7 имеют емкость от 33 до 200 млн. эквивалентных вентилей. Время синтеза конфигурационного файла с большим заполнением (от 80%) для одного такого кристалла может составлять от нескольких часов до нескольких суток, что существенно увеличивает время отладки параллельной программы для многокристальной РВС. Рассмотренные ранее средства программирования РВС лишь частично сокращают процесс отладки, позволяя моделировать формируемые виртуальные вычислительные структуры для отдельных кристаллов ПЛИС, и не имеют средств для отладки для многокристальных решений.

Для сокращения времени отладки многокристальных решений задач предметной области разработана [14] и реализована [15] концепция софт-архитектур, которая позволяет без перезагрузки файлов конфигурации ПЛИС вычислительного поля с помощью программной настройки изменять коммутацию между устройствами и создавать необходимые вычислительные структуры для решения прикладных задач пользователя.

Софт-архитектура — это созданная вычислительная структура, содержащая макрообъекты. Макрообъект — это совокупность вычислительных устройств, выполняющих определенную группу команд и соединенных между собой коммутационной системой. Для каждого класса задач, решаемых на РВС, можно подобрать определенный набор оптимальных вычислительных структур (макрообъектов), наиболее эффективно решающие задачи данного класса. Для макрообъекта допустимо изменение количества функциональных вычислительных устройств и их параметров (разрядности операндов, числа информационных каналов, системы команд и др.), но недопустимо изменение их назначения. Таким образом, макрообъект с точки зрения прикладного программиста является «заготовкой», которая может доопределяться им при создании конкретного технического решения, а затем тиражироваться в нужном количестве в ПЛИС вычислительных модулей и соединяться с подобными или другими макрообъектами в вычислительные структуры, которые оптимально соответствуют структуре решаемой задачи. На основе макрообъектов возможно создание «софт-архитектуры РВС», под которой понимается созданная схемотехником вычислительная структура, содержащая макрообъекты, в которой можно без перезагрузки файлов конфигурации ПЛИС вычислительного поля с помощью программной настройки изменять коммутацию между устройствами и создавать необходимые вычислительные структуры для решения прикладных задач пользователя.

Использование софт-архитектур позволяет сократить время разработки прикладной программы РВС за счет уменьшения времени трансляции. При этом производительность вычислительной системы остается на уровне, сопоставимом с производительностью специализированных вычислительных структур. Это достигается путем создания и предварительной загрузки в РВС настраиваемых проблемно-ориентированных софт-

архитектур, способных решать задачи определенного класса. Опыт создания вычислительных структур на РВС показывает, что для каждого класса задач можно выделить конечный набор вычислительных узлов, при помощи которых может быть построена вычислительная архитектура, эффективно решающая задачи данного класса. Вычислительные узлы, управляемые посредством системы команд и связанные в единую вычислительную структуру, составляют софт-архитектуру.

Это обеспечивает при сохранении преемственности принципов программирования и использования языка высокого уровня для программирования РВС возможность простой адаптации программных компонентов средств разработки для РВС при переходе на новые топологии ПВМ без внесения существенных изменений в код программных компонентов комплекса, а также позволяет сократить время решения прикладных задач.

Для создания софт-архитектур разработан язык SADL (Soft-Architecture Development Language) [14], который транслируется в виртуальную архитектуру вычислительной системы для синтезатора Fire!Constructor, на которую отображается информационный граф прикладной задачи. Для создания софт-архитектуры выполняются:

- разработка описания софт-архитектуры на языке SADL;
- трансляция описания софт-архитектуры в промежуточное представление при помощи синтезатора конфигураций параллельно-конвейерных вычислительных структур Fire!Constructor;
- размещение элементов софт-архитектуры на аппаратной платформе при помощи синтезатора масштабируемых параллельно-конвейерных процедур Steam!Constructor.

Транслятор языка SADL преобразует текст программы в промежуточное представление, используемое синтезатором FireConstructor для размещения на аппаратной платформе РВС. Результатом размещения софт-архитектуры на аппаратную платформу являются модифицированный файл промежуточного представления и конфигурационные файлы для ПЛИС, участвующих в размещении софт-архитектуры на аппаратной платформе РВС. После того как софт-архитектура была размещена на аппаратной платформе РВС, она может быть использована для решения различных прикладных задач заданной проблемной области.

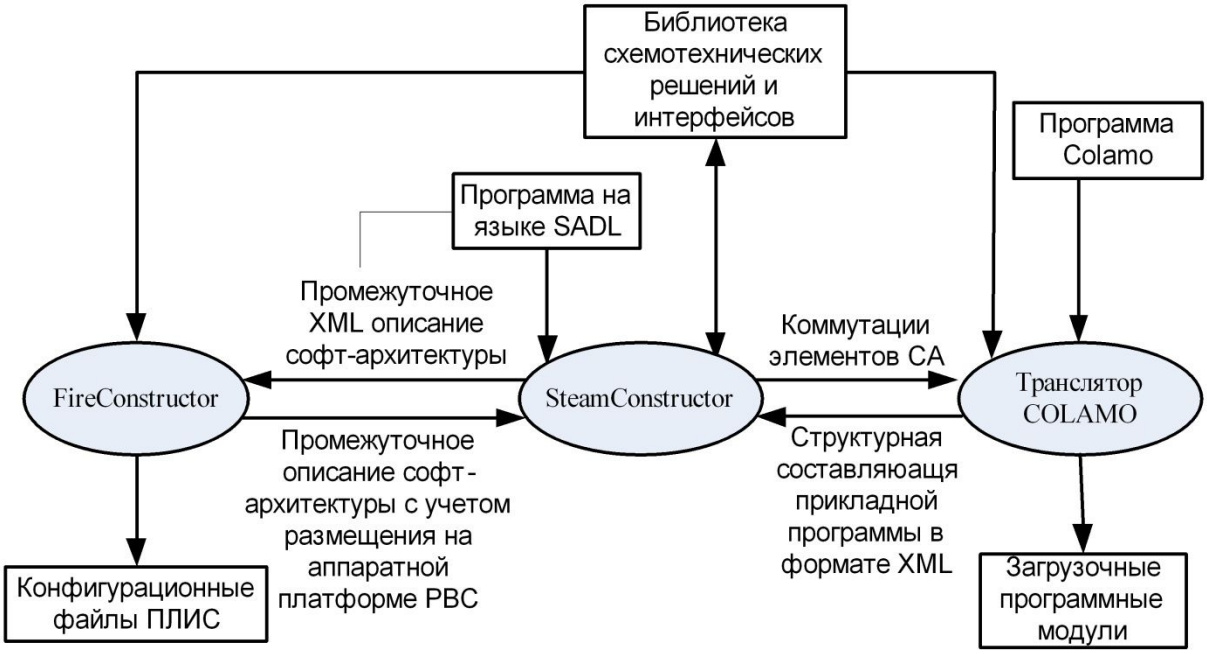
Разработка прикладной программы для софт-архитектуры РВС состоит из следующих основных этапов:

1) разработка параллельной программы на языке высокого уровня COLAMO;
2) трансляция параллельной программы, преобразование информационного графа в структурный и процедурный компоненты;
3) отображение структурного компонента параллельной программы на софт-архитектуру при помощи синтезатора конфигураций параллельно-конвейерных вычислительных структур SteamConstructor;
4) трансляция процедурного компонента параллельной программы на уровень команд устройств софт-архитектуры;
5) формирование загрузочного out-файла, содержащего команды элементов софт-архитектуры;

6) загрузка конфигурационных файлов ПЛИС, полученных в результате размещения элементов софт-архитектуры на аппаратной платформе реконфигурируемой системы;
7) загрузка out-файла;
8) загрузка в софт-архитектуру исходных данных решаемой задачи;
9) запуск программы на исполнение и выгрузка результатов.

Взаимодействие средств разработки прикладных программ при разработке софт-архитектуры показано на рисунке.

Благодаря разработанным программным средствам разработка и модификация софт-архитектур не требуют привлечения высококвалифицированного специалиста-схемотехника для изменения вычислительной структуры. При этом время, затрачиваемое на создание или модификацию софт-архитектуры, значительно сокращается, а получаемые многокристальные архитектурные решения сравнимы по эффективности с решениями, выполненными специалистами-схемотехниками вручную. В соответствии с основными принципами языка COLAMO параллельные прикладные программы могут быть легко модифицированы для адаптации к доступному вычислительному ресурсу.



Структура взаимодействия средств разработки

Автоматизация отображения графов на ресурс РВС позволяет разработчикам прикладных задач мыслить не несколькими ПЛИС, а одной виртуальной ПЛИС с большим логическим объемом.

Комплекс разработанных программных средств позволяет программисту РВС разрабатывать и выполнять отладку прикладных параллельных программ для РВС, не вникая в особенности архитектуры РВС, а также самостоятельно создавать и модифицировать различные софт-архитектуры, ориентируясь на предметную область, к которой принадлежит решаемая задача.

4. Заключение

Благодаря разработанной концепции программирования РВС с использованием софт-архитектур программисту не требуется при каждом изменении программы синтезировать новые конфигурационные файлы ПЛИС. Текст программы транслируется в команды настройки, заранее загруженной в РВС софт-архитектуры, на выполнение тех или иных вычислений. Если разработанная софт-архитектура не имеет достаточного количества вычислительных блоков определенного типа, или блоки данного типа совсем отсутствуют в ней, то программист с помощью языка SADL может модернизировать софт-архитектуру и лишь в данном случае потребуется синтезировать новые конфигурационные файлы ПЛИС.

Современные ПЛИС, такие как кристаллы семейства Virtex-7 фирмы Xilinx, имеют большую емкость и время синтеза конфигурационных файлов занимает существенную часть на этапе отладки параллельной программы. Таким образом, снижение количества выполнения этапа синтеза конфигурационных файлов ПЛИС приводит к снижению времени отладки параллельной программы и времени разработки решения прикладной задачи в целом. При этом производительность вычислительной системы при использовании софт-архитектур остается на уровне, сопоставимом с производительностью специализированных вычислительных структур.

Предложенная концепция программирования РВС была апробирована на РВС на основе ПЛИС Xilinx Virtex-7, при решении задач из области цифровой обработки сигнала. В частности были решены задача многоканальной цифровой фильтрации для мониторинга систем цифровой связи третьего и четвертого поколений (КИХ-фильтр высокого порядка), задача нахождения спектра комплексного сигнала с использованием алгоритма быстрого преобразования Фурье (БПФ) и задача обработки спекл-изображений по методу Лабейри. Использование софт-архитектур позволило сократить время отладки в 2–3 раза, при этом снижение производительности по сравнению с производительностью специализированных вычислительных структур решающих данные задачи, не превышает 15–20%.

Работа выполнена при финансовой поддержке Министерства образования и науки РФ по Соглашению о предоставлении субсидии №14.578.21.0006 от 05.06.2014, уникальный идентификатор RFMEFI57814X0006, НИР регистрационный №114061040060 и НИР регистрационный №01201460286.

Литература

1. http://www.nallatech.com/(дата обращения 25.12.2014)
2. http://picocomputing.com/(дата обращения 25.12.2014)
3. http://www.conveycomputer.com/(дата обращения 25.12.2014)
4. http://www.maxeler.com/(дата обращения 25.12.2014)
5. http://www.srccomp.com/(дата обращения 25.12.2014)
6. Зотов, В.Ю. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPACK ISE / В.Ю. Зотов. — М.: Горячая линия-Телеком. 2003. — 624 с.
7. Quartus II Handbook Version 10.1 Volume 1: Design and Synthesis. Altera Corporation 2010 — 130 p.

8. Libero IDE v9.1 User's Guide. Actel Corporation 2010. — 633 p.
9. Проектирование для ПЛИС Xilinx с применением языков высокого уровня в среде Vivado HLS / Компоненты и технологии, 2013. — №12. — С. 10—17.
10. http://www.altera.com/literature/lit-opencl-sdk.jsp/(дата обращения 25.12.2014)
11. Каляев, И.А. Реконфигурируемые мультиконвейерные вычислительные структуры / И.А. Каляев, И.И. Левин, Е.А. Семерников, В.И. Шмойлов / Изд. 2-е, перераб. и доп. / Под общ. ред. И.А. Каляева. — Ростов-на-Дону: Изд-во ЮНЦ РАН, 2009. — 344 с.
12. Гудков, В.А. Расширение языка высокого уровня COLAMO для программирования реконфигурируемых вычислительных систем на уровне логических ячеек ПЛИС / В.А. Гудков, И.И. Левин // Вестник компьютерных и информационных технологий. — М.: Машиностроение, 2010. — № 12. — С. 10—17.
13. Гудков, В.А. Средства программирования реконфигурируемых многопроцессорных вычислительных систем / В.А. Гудков, А.А Гуленок, А.И. Дордопуло, Л.М. Сластен // Известия ТРТУ. Тематический выпуск «Интеллектуальные и многопроцессорные системы». — Таганрог: Изд-во ТРТУ, 2006. — № 16 (71). Специальный выпуск. — С. 16—20.
14. Семерников, Е.А. Организация многоуровневого программирования реконфигурируемых вычислительных систем / Е.А. Семерников, В.Б. Коваленко // Вестник компьютерных и информационных технологий. — М.: Машиностроение, 2011. — № 9. — С. 3—10.
15. Gudkov, V.A. / V.A. Gudkov, A.A., Gulenok, V.B. Kovalenko, L.M. Slasten Multi-level Programming of FPGA-based Computer Systems with Reconfigurable Macroobject Architecture / Preprints of the 12th IFAC Conference on Programmable Devices and Embedded Systems PDES, 2013.— Technical University of Ostrava, Czech Republic. P. 65—70.

Левин Илья Израилевич, доктор технических наук, профессор, зам. директора по науке, НИИ многопроцессорных вычислительных систем имени академика А.В. Каляева, Южный федеральный университет (Таганрог, Российская Федерация), levin@mvs.tsure.ru.

Дордопуло Алексей Игоревич, кандидат технических наук, старший научный сотрудник, Южный научный центр РАН (Ростов-на-Дону, Российская Федерация), scorpio@mvs.tsure.ru.

Коваленко Василий Борисович, кандидат технических наук, младший научный сотрудник, Южный научный центр РАН, (Ростов-на-Дону, Российская Федерация), vereten@hotbox.ru.

Гудков Вячеслав Александрович, кандидат технических наук, старший научный сотрудник, НИИ многопроцессорных вычислительных систем имени академика А.В. Каляева, Южный федеральный университет (Таганрог, Российская Федерация), Slava_Gudkov@mail.ru.

Гуленок Андрей Александрович, кандидат технических наук, старший научный сотрудник, НИИ многопроцессорных вычислительных систем имени академика А.В. Каляева, Южный федеральный университет (Таганрог, Российская Федерация), andrei_gulenok@mail.ru.

Поступила в редакцию 29 декабря 2014 г.

Bulletin of the South Ural State University
Series "Computational Mathematics and Software Engineering"
2015, vol. 4, no. 2, pp. 20–32

DOI: 10.14529/cmse150202

PROGRAMMING TOOLS FOR RECONFIGURABLE COMPUTER SYSTEMS BASED ON VIRTEX-7 FPGAS WITH USING SOFT-ARCHITECTURES

I.I. Levin, Academician A.V. Kalyaev SRI multiprocessor computer system at Southern Federal University (Taganrog, Russian Federation) levin@mvs.tsure.ru,

A.I. Dordopulo, Southern Scientific Centre of the RAS (Rostov-on-Don, Russian Federation) scorpio@mvs.tsure.ru,

V.B. Kovalenko, Southern Scientific Centre of the RAS (Rostov-on-Don, Russian Federation) vereten@hotbox.ru,

V.A. Gudkov, Academician A.V. Kalyaev SRI multiprocessor computer system at Southern Federal University (Taganrog, Russian Federation) Slava_Gudkov@mail.ru,

A.A. Gulenok, Academician A.V. Kalyaev SRI multiprocessor computer system at Southern Federal University (Taganrog, Russian Federation) andrei_gulenok@mail.ru

The paper covers the existing design tools of digital devices in FPGAs, programming languages of reconfigurable computer systems and ability of their use for programming of multichip reconfigurable computer systems. Besides it deals with the high-level programming language COLAMO and the multichip solution development suite for reconfigurable computer systems that are developed in SRI MCS SFU. A particular attention is paid to a new programming approach, which consists in development and use of adjustable special-purpose soft-architectures which help to reduce the number of translations of FPGA bitstream files during debugging of parallel programs on reconfigurable computer systems. Owing to special-purpose soft-architectures it is possible to change communication links between devices and create required computing structures for user applications only by means of program adjustment and without re-loading FPGA bitstream files of the computational field. It considerably reduces the debugging time of parallel applications.

Keywords: reconfigurable computer systems, parallel programming, soft-architecture, RCS programming, special-purpose soft-architectures.

References

1. http://www.nallatech.com/(accessed:25.12.2014)
2. http://picocomputing.com/(accessed: 25.12.2014)
3. http://www.conveycomputer.com/(accessed: 25.12.2014)
4. http://www.maxeler.com/(accessed: 25.12.2014)
5. http://www.srccomp.com/(accessed: 25.12.2014)
6. Zotov V.Y. Proijektirovanije tsifrovykh ustroistv na osnove PLIS firmy XILINX v SAPR WebPACK ISE [The Design (большая буква) of Digital Devices Based on XILINX FPGAs Using CAD WebPACK ISE]. M.: Goryachaya liniya-Telekom. [Moscow: Hot-Line-Telecom]. 2003. 624 P.

7. Quartus II Handbook Version 10.1 Volume 1: Design and Synthesis. Altera Corporation, 2010. 130 P.
8. Libero IDE v9.1 User's Guide. Actel Corporation. 2010. 633 P.
9. Proijektirovanije dlya PLIS Xilinx s primenenijem yazikov visokogo urovnya v srede Vivado HLS [Design for a Xilinx FPGAs Using High-level Languages in Vivado HLS IDE] / Komponenty i tekhnologii. [Components and technologies]. 2013. Vol. 12. P. 10—17.
10. http://www.altera.com/literature/lit-opencl-sdk.jsp/(accessed: 25.12.2014)
11. Kaliaev I.A., Levin I.I., Semernikov E.A., Shmoilov V.I. Rekonfiguriruijemyie multikonveyijernyie vichislitelniye struktury [Reconfigurable Multipipeline Computational Structures] / Izd. 2nd, pererabotannoye i dopolnennoye / Pod obshei redaktsiyei I.A. Kaliaeva. [Second Edition, Rev. and Suppl./ Editor I.A. Kaliaev]. Rostov-on-Don, Publishing of SSC RAS, 2009. 344 P.
12. Gudkov V.A.,(шр) Levin I.I. Rasshireniye yazika visokogo urovnya COLAMO dlya programmirovaniya rekonfiguriruijemykh vichislitelnykh sistem na urovnie logicheskikh yacheek PLIS [High-level Language COLAMO Extension for Reconfigurable Computing Systems Programming at the Level of the FPGA Logic Cells]. Vestnik komputernykh i informatsionnykh tekhnologii. M.: Mashinostroyeniye. [Herald of Computer and Information Technologies]. Moscow, Engineering , 2010. Vol. 12. P. 10—17.
13. Gudkov V.A., Gulenok A.A., Dordopulo A.I., Slasten L.M. Sredstva programmirovaniya rekonfiguriruijemykh mnogoprotsessornykh vichislitelnykh sistem [Software Tools for Reconfigurable Multiprocessor Computing Systems]. Izvestia TRTU. Tematicheskiy vipusk "Intellectualnie i mnogoprocessornie systemi". [Proceedings of TSURE. Subject issue "Intelligent and multiprocessor systems"]. Taganrog, TSURE Publishing, 2006. Vol. 16 (71). Special issue. P. 16—20.
14. Semernikov E.A., Kovalenko V.B. Organizatsiya mnogourovnevogo programmirovaniya rekonfiguriruijemykh vichislitelnykh sistem [Organization Multi-level Programming of Reconfigurable Computing Systems]. Vestnik komputernykh i informatsionnykh tekhnologii. M.: Mashinostroyeniye. [Herald of Computer and Information Technologies.] Moscow, Engineering, 2011. Vol. 9. P. 3—10.
15. Gudkov V.A., Gulenok A.A., Kovalenko V.B., Slasten L.M. Multi-level Programming of FPGA-based Computer Systems with Reconfigurable Macroobject Architecture. Preprints of the 12th IFAC Conference on Programmable Devices and Embedded Systems PDES-2013, Technical University of Ostrava, Czech Republic (24—29 September 2013). P. 65—70.

Received December 29, 2014.