УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
март–апрель 2019 Том 19 № 2 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
March–April 2019 Vol. 19 No 2 ISSN 2226-1494 http://ntv.ifmo.ru/en

УДК 004.273

СИСТЕМА ВЫСОКОУРОВНЕВОГО СИНТЕЗА НА ОСНОВЕ ГИБРИДНОЙ РЕКОНФИГУРИРУЕМОЙ МИКРОАРХИТЕКТУРЫ

А.В. Пенской[a], А.Е. Платунов[a], А.О. Ключев[a], Я.Г. Горбачев[a], Р.И. Яналов[b]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[b] ООО «Люксофт Профешнл», Санкт-Петербург, 195027, Российская Федерация
Адрес для переписки: aeplatunov@corp.ifmo.ru

Информация о статье
Поступила в редакцию 01.02.19, принята к печати 28.02.19
doi: 10.17586/2226-1494-2019-19-2-306-313
Язык статьи – русский

Ссылка для цитирования: Пенской А.В., Платунов А.Е., Ключев А.О., Горбачев Я.Г., Яналов Р.И. Система высокоуровневого синтеза на основе гибридной реконфигурируемой микроархитектуры // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 2. С. 306–313. doi: 10.17586/2226-1494-2019-19-2-306-313

Аннотация

Предмет исследования. Исследованы методы создания специализированных программно-аппаратных комплексов реального времени на базе программируемой логики и систем на кристалле, предложена система высокоуровневого синтеза на основе гибридной реконфигурируемой микроархитектуры NISC/TTA. **Метод исследования.** Используются инструменты анализа и синтеза вычислительных микроархитектур и методов их проектирования в методологиях модель-ориентированной инженерии, высокоуровневого синтеза (HLS), программно-аппаратного совместного проектирования (HW/SW Codesign). **Основные результаты.** Проанализированы средства разработки программно-аппаратных комплексов на базе программируемой логики и систем на кристалле. Определены ключевые факторы, препятствующие широкому внедрению средств и технологий высокоуровневого синтеза специализированных вычислителей в промышленности. Разработана концепция и архитектура системы высокоуровневого синтеза на базе оригинальной реконфигурируемой NISC/TTA вычислительной микроархитектуры. Реализован прототип системы автоматизированного проектирования, демонстрирующий вариант синтезатора и интерфейс управления процессом синтеза, позволяющие провести анализ цепочки принятых системой автоматизированного проектирования решений. Система обеспечивает в автоматизированном или ручном режиме управление процессом синтеза. Реализован интерфейс визуализации спланированного системой автоматизированного проектирования целевого вычислительного процесса, позволяющий наглядно представлять его многоуровневую организацию. Построена система сквозного тестирования, позволяющая верифицировать соответствие целевой системы ее функциональной модели. **Практическая значимость.** Реализованные в составе прототипа системы инструменты, сделав процесс синтеза прозрачным и управляемым для оператора, продемонстрировали возможность нахождения компромиссных проектных решений в автоматизированном режиме. Удалось гибко управлять эвристиками в работе синтезатора, не только сокращая итерации проектирования, но и делая процесс сходящимся в принципе, чего во многих случаях не обеспечивают альтернативные технологии. Важную роль в этом играет NISC/TTA-микроархитектура. Решение ряда тестовых задач показало, что данная проектная платформа может быть рекомендована для реализации алгоритмов управления в системах реального времени и для использования в задачах моделирования системной динамики.

Ключевые слова
встроенные системы, САПР, системы на кристалле, высокоуровневый синтез, многоуровневая реконфигурация, ПЛИС, совместное проектирование, NISC, TTA

HIGH-LEVEL SYNTHESIS SYSTEM BASED ON HYBRID RECONFIGURABLE MICROARCHITECTURE

A.V. Penskoi[a], A.E. Platunov[a], A.O. Kluchev[a], Ya.G. Gorbachev[a], R.I. Yanalov[b]

[a]ITMO University, Saint Petersburg, 197101, Russian Federation
[b]Luxoft Professional LLC, Saint Petersburg, 195027, Russian Federation
Corresponding author: aeplatunov@corp.ifmo.ru

Article info
Received 01.02.19, accepted 28.02.19
doi: 10.17586/2226-1494-2019-19-2-306-313
Article in Russian

For citation: Penskoi A.V., Platunov A.E., Kluchev A.O., Gorbachev Ya.G., Yanalov R.I. High-level synthesis system based on hybrid

reconfigurable microarchitecture. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2019, vol. 19, no. 2, pp. 306–313 (in Russian). doi: 10.17586/2226-1494-2019-19-2-306-313

Abstract

Subject of Research. The paper presents research of state-of-the-art methods of real-time hardware/software systems development based on FPGAs and SoCs. High-level synthesis system based on the hybrid reconfigurable NISC/TTA microarchitecture is proposed. **Method.** The work is based on analysis and synthesis of computer architectures and their design methods within Model-Driven Engineering, High-Level Synthesis and Hardware/Software Codesign methodologies. **Main Results.** Analysis of academic and commercial tools for development of hardware/software systems based on FPGAs and SoCs is performed. The key factors are determined limiting the wide introduction of high-level synthesis tools in industry. The ideology and architecture of the high-level synthesis system are developed on the basis of the original reconfigurable NISC/TTA microarchitecture. The prototype of the considered EDA system is developed. The synthesis tool and the synthesis process control interface are among the most valuable outcomes and give the possibility to explore the results of synthesis decisions made by the tool, and control the process either manually or automatically. The visualization interface of the target computational process is implemented that allows for effective representation of its multi-level organization. The end-to-end testing system is developed enabling verification of compliance between a synthesized system and its functional model. **Practical Relevance.** The tools implemented as part of the CAD prototype have made the synthesis process transparent and manageable for the designer and demonstrated the possibility of finding compromise design solutions in semi-automatic mode. We managed to control flexibly heuristics in the synthesizer operation, not only reducing design iterations, but also making the process convergent in principle, which is not provided by alternative technologies in many cases. NISC/TTA microarchitecture played an important role in this process. Solution of a number of test problems has shown that this design platform can be recommended for implementation of control algorithms in real-time systems and for the application in system dynamics modeling.

Keywords

embedded systems, CAD, systems on chip, high level synthesis (HLS), multilevel reconfiguration, FPGA, hardware/software and architecture/compiler CoDesign, NISC, TTA

Введение

Разработка специализированных и встроенных вычислительных систем [1] для управления динамическими объектами, обработки цифровых сигналов и моделирования в режиме реального времени сопряжена с большим количеством сложностей, среди которых особое значение занимают вопросы производительности и детерминированности работы во времени, энергопотребления, массогабаритных характеристик, надежности. Во многих случаях применение готовых микропроцессоров не позволяет обеспечить требуемые характеристики, а разработка заказной микросхемы ASIC (Application-Specific Integrated Circuit) для рассматриваемого класса задач, как правило, экономически нецелесообразна.

Для реализации описанных выше систем широко используются программируемые логические интегральные схемы (ПЛИС) и системы на кристалле (СнК). Они предоставляют возможность создать специализированный вычислитель, поддерживающий все необходимые вычислительные блоки, интерфейсы и периферийные устройства, с характеристиками, значительно лучшими, чем при реализации на массовых микропроцессорах и микроконтроллерах [2]. Заказное проектирование на уровне аппаратной составляющей также позволяет внедрить в систему дополнительные механизмы обеспечения надежности с минимальными накладными расходами. Кроме того, есть и принципиальное преимущество перед ASIC – реконфигурируемость системы в процессе эксплуатации [3]. Это позволяет снизить риски за счет возможности корректировки аппаратной составляющей и повысить степень использования ресурсов посредством перепрограммирования ПЛИС в соответствии с текущими потребностями.

Действующим промышленным стандартом разработки на ПЛИС является представление проекта на уровне регистровых передач (RTL, Register-Transfer Level) с применением языков Verilog и VHDL (рис. 1, *а*) с последующим получением реализации средствами САПР. Несмотря на серьезные достижения в данной области [4, 5] специалистами отмечаются следующие проблемы:

1) разработка на уровне RTL требует глубокого понимания принципов цифровой схемотехники и навыков работы с САПР для ПЛИС;
2) ключевые микроархитектурные решения принимаются на ранних этапах и подчинены прикладному алгоритму, что приводит к высоким проектным рискам [6];
3) длительный процесс синтеза (единицы часов), медленная симуляция [4].

Преодолеть указанные проблемы позволяет высокоуровневый синтез (HLS, High Level Synthesis) [7]. HLS автоматизирует процесс трансляции прикладного алгоритма, описанного на языке высокого уровня, в спецификацию уровня RTL на основе информации о целевой платформе, предоставляя разработчику механизм воздействия на микроархитектуру и характеристики реализации посредством настроек САПР [8]. Это позволяет вынести значительный объём проектирования на этап создания микроархитектуры, а также в автоматическом режиме варьировать соотношение между площадью кристалла, производительностью и потребляемой мощностью на поздних стадиях проекта.

Немаловажным достоинством применения HLS является сокращение времени тестирования,

достигаемое за счет быстрого моделирования высокоуровневого описания по сравнению с симуляцией RTL, повторного использования высокоуровневого тестового окружения для верификации RTL.

В [7] представлен обзор текущего состояния HLS и приводится сравнение более 30 различных HLS-систем. В [9] показаны основные болевые точки и проблемы, существующие в данной области. Из работ [7, 9] видно, что современные средства HLS (рис. 1, *б*) обладают рядом ограничений:

– для их эффективного использования по-прежнему необходимо глубокое понимание принципов проектирования цифровой схемотехники;
– высокая сложность, непрозрачность, а зачастую и нестабильность работы САПР, что выражается в невозможности прогнозирования последствий даже незначительных изменений в прикладном алгоритме, что приводит к необходимости глубокого понимания инструмента и его особенностей;
– большинство средств HLS накладывают серьезные ограничения на пользовательские вычислительные блоки с точки зрения интеграции, интерфейса обмена данными и интерфейса управления;
– типовые шаблоны синтеза и ориентированность на системы обработки данных (data-dominated systems), например, системы цифровой обработки сигналов, для которых пропускная способность гораздо важнее времени реакции, что значительно сужает область эффективного применения HLS-инструментов;
– промышленные инструменты высокоуровневого синтеза (например, Vivado HLS), как правило, входят в состав крупных САПР и неотчуждаемы от них, что в совокупности с несовместимостью на уровне прикладного языка привязывает к конкретному поставщику инструментальных средств.

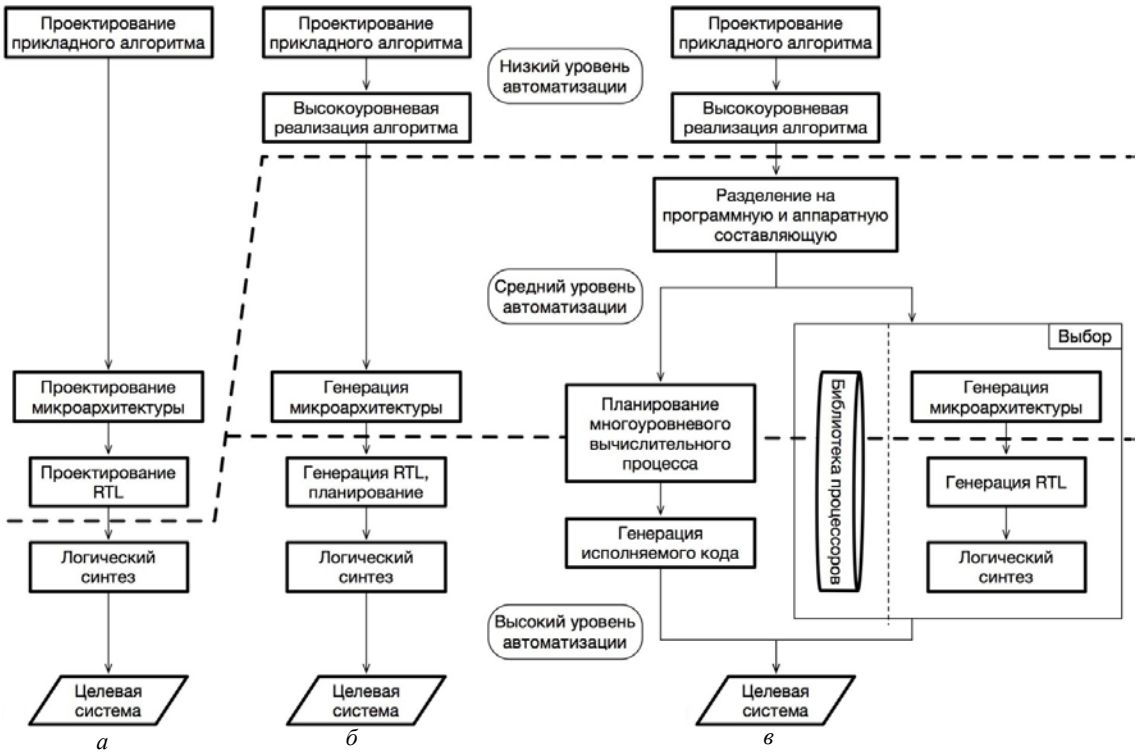

Рис. 1. Маршрут проектирования: *а*) на уровне RTL; *б*) с применением HLS; *в*) с применением вычислительной платформы на основе NISC/TTA

Таким образом, можно утверждать, что сформировался осознанный запрос на инструменты высокоуровневого синтеза [7], но перечисленные выше недостатки препятствуют широкому принятию какой-либо одной конкретной САПР в качестве промышленного стандарта и ограничивают возможности интеграции HLS-инструментов. При этом необходимо отметить, что решение обозначенных проблем в значительной степени лежит в области научных задач, а не инженерии [3, 10].

Гибридная реконфигурируемая микроархитектура NISC/TTA

Авторы работают над преодолением обозначенных проблем в проекте по созданию системы высокоуровневого синтеза на основе гибридной реконфигурируемой NISC/TTA (No Instruction Set Computing/Transport Triggered Architecture) микроархитектуры. Данная платформа ориентирована на реализацию высокоскоростных алгоритмов управления с элементами цифровой обработки сигналов и позволяет прикладным специалистам активно участвовать в процессе разработки на всех ее этапах. Типовую трудоемкость решаемых на данной платформе прикладных задач можно оценить по [11, 12].

Важнейшим исследовательским приоритетом проекта является обеспечение прозрачности и управляемости процесса синтеза, поддержка реконфигурируемости и повторного использования артефактов, принадлежащих различным уровням представления проекта, а также максимальное использование для выбранной микроархитектуры вычислительных ресурсов ПЛИС.

Определяющую роль в технологиях HLS играет выбор используемой микроархитектуры – набора принципов и архитектурных шаблонов, описывающих организацию аппаратной составляющей целевой системы и определяющих, в частности: возможности по масштабированию и реконфигурации; способы реализации специализированных вычислительных блоков; уровень и тип параллелизма. Кроме того, микроархитектура оказывает значительное влияние на метод синтеза, модель целевого вычислительного процесса и класс реализуемых прикладных алгоритмов.

В работе предлагаются гибридная реконфигурируемая микроархитектура NISC/TTA – без системы команд, где вычислительный процесс строится на основе транзакций между вычислительными блоками, и средства для ее автоматической генерации. Микроархитектура сочетает элементы зарекомендовавших себя решений:

1) NISC (No Instruction Set Computing [13]) – широко применяемый в области высокоуровневого синтеза подход, направленный на использование в качестве инструкций горизонтальных микрокоманд. Это позволяет минимизировать аппаратную составляющую, задержки и накладные расходы, генерировать эффективный высокопараллельный программный код для вычислителя, сочетающего разнотипные вычислительные блоки;
2) TTA (Transport Triggered Architecture [14]) – подход к разработке высокопараллельных гетерогенных вычислителей, управляемых при помощи команд передачи данных между вычислительными блоками. Этот подход позволяет обеспечить высокий уровень параллелизма аппаратной составляющей, определяет пути для масштабирования вычислителя, позволяет вычислительным блокам взаимодействовать непосредственно, минуя промежуточные регистры, и самое главное – формирует простую и наглядную модель описания вычислительного процесса и аппаратной составляющей.

Прототипом для микроархитектуры NISC/TTA стал процессор NL3[1] [6, 15]. NL3 успешно применяется в различных областях, в том числе в сложных аналитических приборах [6, 16]. Основные преимущества его гибридной микроархитектуры заключаются в применении динамического потока управления и масштабируемости коммуникационной инфраструктуры.

Описанные свойства NISC/TTA обеспечивают простую процессорную микроархитектуру (рис. 2), где вся обработка и хранение данных, взаимодействие с периферийным оборудованием и задачи организации вычислительного процесса сосредоточены в вычислительных блоках (ВБ). При этом на шины управления ВБ не накладывается принципиальных ограничений, а микрокод генерируется статически в процессе синтеза. В целом микроархитектурная платформа поддерживает модель программирования со следующими свойствами:

– вычислительный процесс носит циклический характер, где на каждом цикле производятся чтение входных данных, их обработка и формирование выходных данных, часть из которых по логическим обратным связям поступает на вход алгоритма вновь;
– длительность вычислительного цикла задается настройками инструментария и гарантируется в случае успешного синтеза, что позволяет эффективно решать задачи реального времени.

Дополнительно необходимо отметить следующие достоинства гибридной микроархитектуры NISC/TTA:

– масштабируемость по количеству ВБ и наличие механизмов компенсации узких мест (коммуникационная подсистема, ширина хранимых инструкций);
– высокая параллельность вычислительного процесса с минимальными накладными расходами на его организацию;
– микропрограммное управление разнородными ВБ без ограничений на их внутреннюю организацию, состав реализуемых ими функций и уровень внутреннего параллелизма.

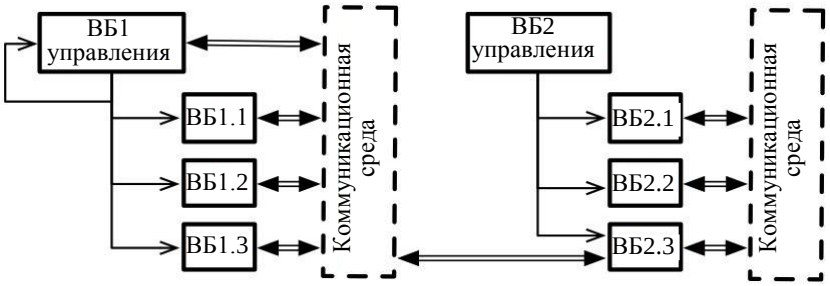

Рис. 2. Гибридная микроархитектура NISC/TTA

[1] Разработка дизайн-центра «ЛМТ» при участии ряда авторов данной статьи. В настоящее время процессор используется в зондовых сканирующих микроскопах NanoTutor, MicProbe, ProBeam производства NT-Spb, см. https://nano.ifmo.ru

Обратной стороной простой аппаратной составляющей является относительно высокая сложность САПР.

Принципы организации системы автоматизированного проектирования

Основными отличительными особенностями предложенной системы от других систем высокоуровневого синтеза являются:

а) простое описание целевого вычислительного процесса. В работе решено отказаться от идеи абстрагировать прикладного специалиста от аппаратной составляющей и принципов ее работы. Вместо этого ему предоставляется простая и понятная модель вычислительной системы и вычислительного процесса. Это позволяет не только значительно снизить количество неявных преобразований и оптимизаций в инструментарии, но также дает язык и предмет для взаимодействия прикладных специалистов со специалистами по разработке на ПЛИС;

б) управляемость процессом синтеза и глубокая реконфигурируемость. Одним из важнейших свойств САПР является предсказуемость получаемого результата. Как правило, это достигается за счет большого опыта использования, что сегодня труднодостижимо в такой динамично развивающейся и сложной области, как HLS.

В работе используется максимальная открытость внутренних алгоритмов принятия решений. Это выражается в возможности остановить процесс синтеза в произвольном месте, проанализировать, какие варианты уже были рассмотрены и почему не были приняты, какие варианты рассматриваются в текущий момент, а какие будут рассмотрены позднее, почему выбрана та или иная стратегия поиска решения.

Выстраивание вычислительной платформы на принципах глубокой реконфигурируемости позволяет принимать решения параллельно на разных уровнях организации вычислительного процесса. Основными являются:

– уровень микроархитектуры аппаратной составляющей, в рамках которого определяется коммуникационная подсистема и состав используемых вычислительных блоков (какие функции могут вычисляться и с какой скоростью);
– уровень планирования потока управления и потока данных целевого вычислительного процесса;

в) повторное использование аппаратной составляющей и регламент для совместного проектирования.

В большинстве случаев результатом высокоуровневого синтеза является RTL-проект, в котором присутствуют неразделимые, глубоко интегрированные между собой компоненты аппаратной и микропрограммной составляющей. Следствием этого является невозможность изменения прикладного алгоритма без повторного прохождения всего маршрута проектирования, что не всегда удобно по причине длительности процесса синтеза и необходимости иметь в наличии САПР для ПЛИС.

В работе выбран иной подход, в рамках которого происходит явное раннее разделение функциональности проектируемой системы на программную и аппаратную составляющие в соответствии с принципами совместного проектирования (HW/SW CoDesign [17]), что позволяет активно использовать аппаратную составляющую при синтезе повторно (рис. 1, *в*). Особую важность этому свойству придает гибкое управление процессом синтеза, так как становится возможным создать избыточный вычислитель (проблемно-ориентированную вычислительную платформу) с целью его повторного использования для выбранного класса прикладных алгоритмов и задач.

Это принципиально меняет сценарий использования разрабатываемой вычислительной платформы, делая работу прикладного специалиста в известной мере независимой от работы специалиста по вычислительной технике.

Прототип системы высокоуровневого синтеза

Любые достоинства системы высокоуровневого синтеза и используемой вычислительной микроархитектуры могут быть нивелированы неудачным (медленным, непредсказуемым, неуправляемым, не масштабируемым и т.п.) методом синтеза.

Важно учитывать, что решаемая задача имеет огромную вычислительную сложность, ввиду чего критически важны выбор стратегии поиска решения и используемые эвристики. В процессе синтеза решаются следующие задачи:

1) оптимизация прикладного алгоритма с учетом конфигурации процессора;
2) формирование конфигурации процессора (коммуникационная и управляющая инфраструктура, состав и схема подключения ВБ);
3) распределение функций прикладного алгоритма по ВБ, доступным в конкретной конфигурации процессора;
4) планирование потока управления и потока данных;
5) планирование вычислительного процесса для прикладного алгоритма с учетом передачи данных между ВБ.

При этом можно видеть, что перечисленные задачи взаимосвязаны. Для того чтобы сделать столь комплексный и ресурсоемкий процесс прозрачным и управляемым предложен итеративный эвристический метод синтеза с поздним «связыванием». Метод рассматривает процесс синтеза как последовательное принятие решений системой автоматизированного проектирования, при котором можно проанализировать причины принятия любого решения, отменить или изменить его, ознакомиться с альтернативными вариантами. Для ранжирования вариантов решений используются эвристики, основанные на анализе спланированного процесса, нераспределенной части прикладного алгоритма, уровня загрузки ВБ, зависимостей по данным и т.д.

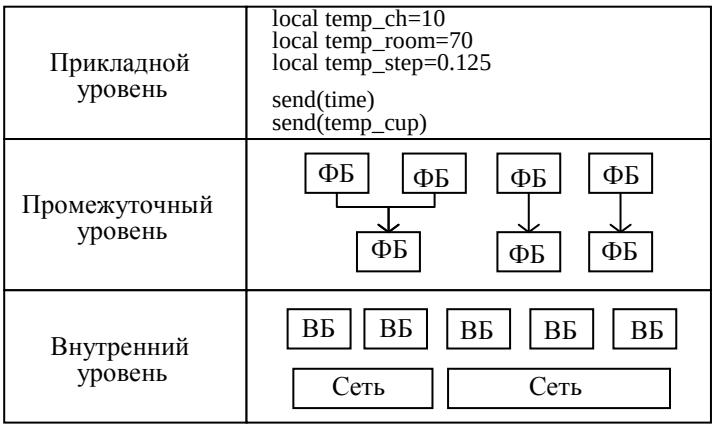

Рис. 3. Три верхних уровня микроархитектуры NISC/TTA (ФБ – функциональный блок)

Кроме того, решения принимаются параллельно на всех уровнях организации вычислительного процесса (рис. 3) [18], так как возможность свободных многократных переходов между уровнями позволяет перейти от классического поэтапного синтеза к постепенному и более гибкому формированию целевого вычислительного процесса, когда объем принятых решений на всех уровнях сводится к минимально достаточному. Важную роль в этом процессе играет многоуровневое представление целевого вычислительного процесса [19], позволяющее качественно анализировать эффективность полученного решения.

Моделирование ВБ для планирования вычислительного процесса и генерации микрокода осуществляется посредством реализации для каждого типа ВБ узкоспециализированной САПР, включаемой в состав описываемой системы высокоуровневого синтеза. Такой подход позволяет описывать не только простейшие ВБ, обеспечивающие хранение данных, выполнение простейших математических функций или обмен данными с периферией, но также и многофункциональные ВБ со сложным поведением, с переменным числом входов и выходов, требующие сложного конфигурирования перед работой и обладающие внутренним параллелизмом.

Разработанный прототип системы высокоуровневого синтеза [20] реализует четыре уровня представления проекта.

1. Прикладной уровень (Frontend level) – язык прикладного программирования и его инструментальные средства (IEC 61131, C, Lua, стрелочная модель вычислений [21], XMILE и др.).
2. Промежуточный уровень (Intermediate level) – универсальное промежуточное представление для вычислительной платформы NISC/TTA, непосредственно используемое в качестве входных данных. На промежуточном уровне создается множество ФБ (рис. 3). ФБ соединяются друг с другом в рамках модели вычислений, ближайшим аналогом которой является модель синхронных потов данных.
3. Внутренний уровень (Internal level) – совокупность конфигурации процессора (состав ВБ, схемы подключения), распределения ФБ по вычислительным ресурсам и многоуровневого описания вычислительного процесса.
4. Уровень реализации (Backend level) – совокупность программного и аппаратного обеспечения (их спецификаций), позволяющая получить реализацию целевой системы.

В текущей версии системы реализованы следующие компоненты:

1) система синтеза с входными языками Lua и XMILE, транслируемыми в промежуточное представление, и выходным (проект «под ключ») – для системы логического синтеза;
2) система сквозного тестирования, позволяющая верифицировать соответствие целевой системы (включая ввод/вывод) и ее функциональной модели;
3) графический интерфейс управления процессом синтеза, позволяющий провести анализ цепочки принятых САПР решений, а также в автоматизированном или ручном режиме управлять процессом синтеза;

4) графический интерфейс визуализации целевого вычислительного процесса, отражающий его многоуровневую структуру.

Прототип HLS-САПР включает всю необходимую инструментальную инфраструктуру, в том числе средства аппаратного прототипирования на ПЛИС фирм Xilinx, Altera.

Заключение

Система высокоуровневого синтеза на основе гибридной реконфигурируемой NISC/TTA микроархитектуры находится на стадии действующего прототипа. Проходит опытная эксплуатация и апробация на алгоритмах управления технологическими процессами и на задачах моделирования динамических систем [22] совместно с проектом SdCloud[1]. Отмечены следующие достоинства вычислительной платформы и прототипа системы автоматизированного проектирования:

– повышение прозрачности и контролируемости процесса высокоуровневого синтеза по сравнению с такими инструментами, как Xilinx Vivado HLS, Mentor Graphics Catapult HLS, LegUP;
– повышение скорости разработки прикладного алгоритма за счет сокращения временных затрат на процесс синтеза, что достигается в результате выборочного сохранения готовых вариантов аппаратной составляющей;
– прирост эффективности работы вычислительной системы за счет использования специализированных вычислительных блоков;
– обеспечение детерминированности вычислений в реальном времени для высокоскоростных алгоритмов.

Несмотря на полученные положительные результаты, для широкого использования системы необходимо сформировать требования к стандартной библиотеке вычислительных блоков, к библиотеке процессоров, к рабочему окружению и интеграции с существующими системами автоматизированного проектирования.

Литература

1. Lee E.A., Seshia S.A. Introduction to Embedded Systems: A Cyber-Physical Systems Approach. 2nd ed. MIT Press, 2017. 565 p.
2. Hemsoth N., Morgan T.P. FPGA Frontiers: New Applications in Reconfigurable Computing. Next Platform Press, 2017. 182 p.
3. Hartenstein R. SE Curricula are Unqualified to Cope with the Data Avalanche. [Электронный ресурс]. 2017. 23 p. Режим доступа: hartenstein.de/publications/CS.pdf свободный. Яз. англ. (дата обращения: 09.03.2018).
4. Pelcat M. et al. Design productivity of a high level synthesis compiler versus HDL // Proc. Int. Conf. on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS). 2016. P. 140–147. doi: 10.1109/samos.2016.7818341
5. Хэррис Д., Хэррис С. Цифровая схемотехника и архитектура компьютера. Morgan Kaufman, 2013. 1627 с.
6. Платунов А.Е. Теоретические и методологические основы высокоуровневого проектирования встраиваемых вычислительных систем. СПб: Университет ИТМО, 2010. 477 с.
7. Nane R. et al. A survey and evaluation of FPGA high-level synthesis tools // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2016. V. 35. N 10. P. 1591–1604. doi: 10.1109/tcad.2015.2513673
8. Fingeroff M. High-Level Synthesis Blue Book. Xlibris, 2010. 332 p.
9. Bailey B. ESL Flow is Dead [Электронный ресурс]. Semiconductor Engineering. 2016. URL: semiengineering.com/esl-flow-is-dead (дата обращения: 09.03.2018).
10. Teich J., Henkel J., Herkersdorf A. et al. Invasive computing: an overview // Multiprocessor System-on-Chip. 2011. P. 241–268. doi: 10.1007/978-1-4419-6460-1_11
11. Дмитроченко Л.А., Сачков Г.П. Функциональные алгоритмы и уравнения ошибок определения параметров ориентации в инерциальных навигационных системах // Труды МАИ. 2015. № 80. С. 1–24.
12. Зайцев Д.Ю., Неретин Е.С., Рамзаев А.М. Разработка архитектуры универсального модульного контроллера авионики // Труды МАИ. 2016. № 85. С. 1–29.

References

1. Lee E.A., Seshia S.A. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. 2nd ed. MIT Press, 2017.
2. Hemsoth N., Morgan T.P. *FPGA Frontiers: New Applications in Reconfigurable Computing*. Next Platform Press, 2017, 182 p.
3. Hartenstein R. *SE Curricula are Unqualified to Cope with the Data Avalanche*. 2017. 23 p. Available at: hartenstein.de/publications/CS.pdf (accessed: 09.03.2018).
4. Pelcat M. et al. Design productivity of a high level synthesis compiler versus HDL. *Proc. Int. Conf. on Embedded Computer Systems: Architectures, Modeling and Simulation, SAMOS*, 2016, pp. 140–147. doi: 10.1109/samos.2016.7818341
5. Harris D., Harris S. *Digital Design and Computer Architecture*. 2nd ed. Morgan Kaufman, 2012.
6. Platunov A.E. *Theoretical and Methodological Bases of High-Level Design of Embedded Computing Systems*. St. Petersburg, ITMO University Publ., 2010, 477 p. (in Russian)
7. Nane R. et al. A survey and evaluation of FPGA high-level synthesis tools. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2016, vol. 35, no. 10, pp. 1591–1604. doi: 10.1109/tcad.2015.2513673
8. Fingeroff M. *High-Level Synthesis Blue Book*. Xlibris, 2010, 332 p.
9. Bailey B. *ESL Flow is Dead*. Semiconductor Engineering, 2016. Available at: semiengineering.com/esl-flow-is-dead (accessed: 09.03.2018).
10. Teich J., Henkel J., Herkersdorf A. et al. Invasive computing: an overview. *Multiprocessor System-on-Chip*, 2011, pp. 241–268. doi: 10.1007/978-1-4419-6460-1_11
11. Dmitrochenko L.A., Sachkov G.P. Functional algorithms and attitude determination error equations for strap-down inertial navigation systems. *Trudy MAI*, 2015, no. 80, 24 p. (in Russian)
12. Zaytsev D.Y., Neretin E.S., Ramzaev A.M. Universal avionics modular controller architecture development. *Trudy MAI*, 2016, no. 85, 29 p. (in Russian)
13. Henkel J., Parameswaran S. *Designing Embedded Processors: A Low Power Perspective*. Springer, 2007, 550 p. doi: 10.1007/978-1-4020-5869-1
14. Corporaal H., Arnold M. Using transport triggered architectures for embedded processor design. *Integrated Computer-Aided Engineering*, 1998, vol. 5, no. 1, pp. 19–38. doi: 10.3233/ica-

[1] SdCloud project – Режим доступа: https://sdCloud.io/, свободный. Яз. англ. (дата обращения 06.02.2019)

13. Henkel J., Parameswaran S. Designing Embedded Processors: A Low Power Perspective. Springer, 2007. 550 p. doi: 10.1007/978-1-4020-5869-1
14. Corporaal H., Arnold M. Using transport triggered architectures for embedded processor design // Integrated Computer-Aided Engineering. 1998. V. 5. N 1. P. 19–38. doi: 10.3233/ica-1998-5103
15. Ковязин Р.Р., Постников Н.П. Разработка проблемно-ориентированных процессоров // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. 2010. № 6 (70). С. 81–85.
16. Голубок А.О., Платунов А.Е., Сапожников И.Д. Система управления сканирующим зондовым микроскопом // Научное приборостроение. 2003. Т. 13. № 3. С. 25–31.
17. Teich J. Hardware/Software codesign: the past, the present, and predicting the future // Proceedings of the IEEE. 2012. V. 100. P. 1411–1430. doi: 10.1109/JPROC.2011.2182009
18. Kluchev A., Platunov A., Penskoi A. HLD - methodology in embedded systems design with a multilevel reconfiguration // Proc. 3rd Mediterranean Conference on Embedded Computing (MECO-2014). Budva, Montenegro, 2014. P. 36–39. doi: 10.1109/meco.2014.6862729
19. Пенской А.В. Архитектурное документирование встроенных систем с многоуровневой конфигурацией // Изв. вузов. Приборостроение. 2015. № 7. С. 527–532.
20. Penskoi A., Gaiosh A., Platunov A., Kluchev A. Specialised computational platform for system dynamics // Proc. 18th International Multidisciplinary Scientific GeoConference. 2018. V. 18. N 2.1. P. 709–716. doi: 10.5593/sgem2018/2.1/s07.090
21. Hughes J. Generalizing monads to arrows // Science of Computer Programming. 2000. V. 37. N 1-3. P. 67–111. doi: 10.1016/s0167-6423(99)00023-4
22. Perl I., Mulyukin A., Kossovich T. Continuous execution of system dynamics models on input data stream // Proc. 20th Conference of Open Innovations Association (FRUCT). 2017. P. 371–376. doi: 10.23919/fruct.2017.8071336

Авторы

Пенской Александр Владимирович – кандидат технических наук, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 56429383100, ORCID ID: 0000-0002-9842-3276, aleksandr.penskoi@corp.ifmo.ru

Платунов Алексей Евгеньевич – доктор технических наук, профессор, профессор, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 35318291200, ORCID ID: 0000-0003-3003-3949, aeplatunov@corp.ifmo.ru

Ключев Аркадий Олегович – кандидат технических наук, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 56429748500, ORCID ID: 0000-0002-3892-8424, kluchev@gmail.com

Горбачев Ярослав Георгиевич – инженер, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, ORCID ID: 0000-0001-5419-6422, yaroslav-go@yandex.ru

Яналов Роман Игоревич – старший программист, ООО «Люксофт Профешнл», Санкт-Петербург, 195027, Российская Федерация, Scopus ID: 56951112000, ORCID ID: 0000-0002-0980-9733, yanalov.roman@gmail.com

Authors

Alexander V. Penskoi – PhD, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 56429383100, ORCID ID: 0000-0002-9842-3276, aleksandr.penskoi@corp.ifmo.ru

Alexey E. Platunov – D.Sc., Full Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 35318291200, ORCID ID: 0000-0003-3003-3949, aeplatunov@corp.ifmo.ru

Arkady O. Kluchev – PhD, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 56429748500, ORCID ID: 0000-0002-3892-8424, kluchev@gmail.com

Yaroslav G. Gorbachev – engineer, ITMO University, Saint Petersburg, 197101, Russian Federation, ORCID ID: 0000-0001-5419-6422, yaroslav-go@yandex.ru

Roman I. Yanalov Senior programmer, Luxoft Professional LLC, Saint Petersburg, 195027, Russian Federation, Scopus ID: 56951112000, ORCID ID: 0000-0002-0980-9733, yanalov.roman@gmail.com