УДК 004.031.6
DOI: 10.17586/0021-3454-2015-58-7-527-532

АРХИТЕКТУРНОЕ ДОКУМЕНТИРОВАНИЕ ВСТРОЕННЫХ СИСТЕМ С МНОГОУРОВНЕВОЙ КОНФИГУРАЦИЕЙ

А. В. Пенской

Университет ИТМО, 197101, Санкт-Петербург, Россия
E-mail: aleksandr.penskoi@gmail.com

Представлено описание разработанного архитектурного стиля и нотация на базе UML, предназначенные для специфицирования и анализа встроенных систем применительно к организации многоуровневой конфигурации и инструментальной составляющей. Приведен анализ взаимосвязей аппаратной составляющей, программного обеспечения и этапов жизненного цикла системы.

Ключевые слова: *конфигурация, встроенные системы, архитектура, архитектурный стиль, архитектурная нотация.*

Введение. Одним из перспективных направлений разработки встроенных систем (ВсС) является применение реконфигурируемых архитектур [1], где конфигурирование рассматривается как универсальный механизм управления сложностью и функциональностью систем. Возможность изменения конфигурации позволяет создавать системы, адаптируемые для решения новых задач. Это сокращает требуемые на разработку системы технологические и временные затраты за счет повторного использования ее компонентов и оптимизации аппаратной составляющей [2]. В то же время наблюдается заметное увеличение степени конфигурируемости систем в целом, что проявляется как в росте числа уровней конфигураций в системе, так и в мере их влияния на ее конечные характеристики.

Как правило, конфигурация ВсС реализуется посредством введения формальных языков (либо общего назначения, либо специализированных), позволяющих задать свойства системы на различных уровнях ее представления. Уровней конфигураций в системе может быть множество (к примеру, для проекта [3] — языки Java и C, язык описания потоков обработки данных, конфигурация спецпроцессора и FPGA, конфигурация системы ввода—вывода). На рис. 1 представлена схема системы с одним уровнем конфигурации и реализуемого ею вычислительного процесса. Отсутствие ограничений на тип механизма конфигурирования системы и на тип базового вычислителя делают данную схему универсальной.

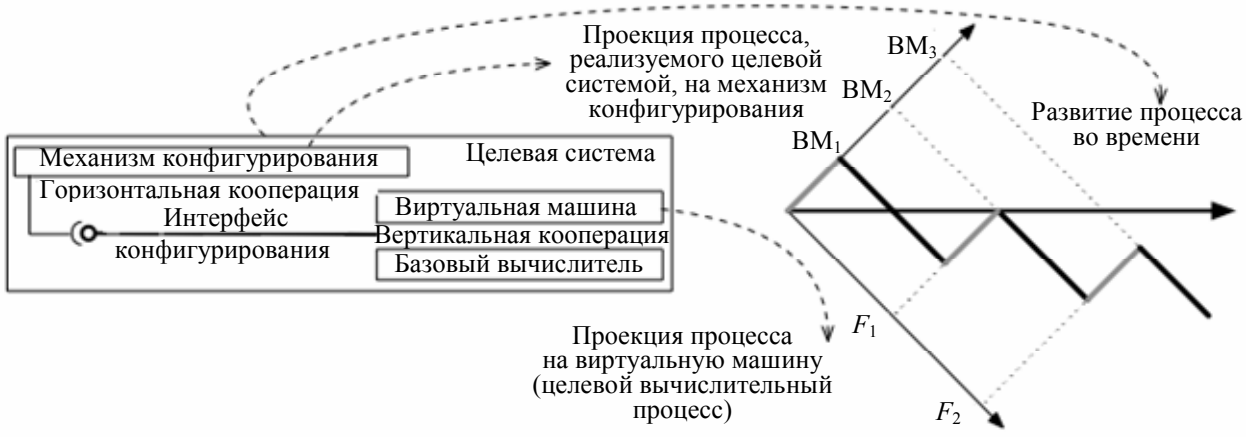

Рис. 1

Применение множества разнородных видов конфигураций в системе приводит к возрастанию сложности инструментальной цепочки, межуровневого взаимодействия и процесса

пусконаладки. Это обусловливает особую значимость средств концептуального уровня для работы над ВсС и ее инструментальной составляющей [4].

Архитектурные стили. Организация многоуровневой конфигурации ВсС требует комплексного рассмотрения всей системы как в абстрактном виде, так и в приближенном к исполняемому (executable) представлении. Такие задачи решаются в рамках архитектурного анализа ВсС. Основную роль в процессе работы на данном уровне абстракции играют архитектурные стили [5], определяющие способы рассмотрения ВсС и систему понятий. Комбинация стилей для каждого проекта выбирается индивидуально, в зависимости от его технических особенностей и организации работы над ним [4]. Выбор используемой нотации является вторичным, так как в значительной степени определяется конкретной ситуацией и традициями, сложившимися в коллективе разработчиков проекта.

Конфигурирование классифицируется:

— по предмету конфигурации [6] — программному обеспечению (программирование) и аппаратному обеспечению (реконфигурация);

— по фазе жизненного цикла — фазе подготовки к эксплуатации (design-time — компиляция, синтез, кодогенерация) и фазе эксплуатации (run-time — интерпретация, реконфигурация ПЛИС).

Правильное распределение функций между видами конфигурации оказывает значительное влияние на характеристики системы, трудозатраты на ее разработку и на эффективность системы в целом. Примерами таких систем являются следующие проекты: процессорная архитектура NISC [7]; реконфигурируемые архитектуры [2]; проекты mbeddr и steps, методологическое направление Hardware Software Codesign [8].

Для многих проектов характерны следующие проблемы [4, 6]:

— суженное пространство проектных решений (Design Space [6]) относительно вычислительных платформ, излишне шаблонное проектирование, фиксация принятых решений в качестве пунктов технического задания — это приводит к росту рисков проекта, разрешающихся лишь на финальных этапах, а также к неоптимальным решениям;

— неэффективное распределение функций и ресурсов между аппаратной и программной составляющими системы, фазами ее жизненного цикла, целевой ВсС и ее инструментальным компонентом — это приводит к неоправданному росту сложности разработки;

— невозможность разделения аппаратной и программной составляющих ВсС на поздних стадиях проекта, что приводит к повышению проектных рисков;

— потеря информации об устройстве ВсС концептуального уровня, что приводит к увеличению стоимости поддержки.

В значительной мере перечисленные проблемы вызваны отсутствием развитых архитектурных стилей и нотаций, ориентированных на работу с многоуровневой конфигурацией ВсС и взаимосвязями уровней. Наиболее подходящими архитектурными стилями являются уровневый стиль и модель актуализации.

Уровневый стиль (Layered Style [5]) ориентирован на модульную декомпозицию системы и, как следствие, не позволяет непосредственно работать с конфигурацией на фазе run-time, кроме того, он не предоставляет понятийного аппарата для работы с виртуальными машинами и вычислителями.

Модель актуализации вычислительного процесса [9]. Данный стиль ориентирован на представление ВсС как совокупности трансляторов. Основными недостатками стиля являются излишняя общность, затрудняющая определение границ его применимости, и совместное рассмотрение процессов конфигурирования и целевой обработки данных. Это приводит к неоднозначности спецификаций и не позволяет сфокусировать внимание на вопросах конфигурирования.

В настоящей статье представлено краткое описание разработанного автором архитектурного стиля и нотации, предназначенных для организации многоуровневой конфигурации в ВсС.

Архитектурный стиль „модель — процесс — вычислитель“. Отличительной особенностью архитектурного стиля, разработанного для специфицирования ВсС с многоуровневой конфигурацией, является масштабируемость относительно количества уровней конфигураций. Возможности предлагаемого стиля продемонстрируем на примере процессорной архитектуры Transport Triggered Architecture (TTA) [10], реализуемой на программируемой логике (ПЛИС). Данная архитектура позволяет организовать следующие уровни конфигураций:

— блоков обработки данных (БОД);

— состава БОД в конкретном процессоре;

— целевого алгоритма.

Данная спецификация основана на гипотезе, что ВсС можно определить как тройку (M, P, C), где M — *модель* вычислительного процесса (конечный автомат, программа на языке СИ или конфигурация ПЛИС); P — *вычислительный процесс* (ВП), конкретный ВП фазы run-time всегда является уникальным; C — *вычислитель* (ПЛИС, процессор, виртуальная машина). Работа с этими компонентами производится взаимонезависимо, в соответствии с результатами в области онтологического моделирования [11]. Это позволяет достичь высокой гибкости спецификаций по сравнению с традиционным подходом. Между этими тремя компонентами (M, P, C) определены следующие отношения.

Трансляция — формальное соответствие двух моделей по заданному критерию (как правило, поведенческому). Например: компиляция с языка СИ в исполняемый код; трансляция архитектурных спецификаций в целевую систему, выполняемая командой разработчиков.

Актуализация — задание/выделение структуры ВП через его модель (выделение промежуточных состояний, шагов и компонентов ВП); может производиться по построению ВП (модель в виде исполняемого кода выполнена на процессоре) или путем внешнего сопоставления (верификация на соответствие ВП, полученного исполнением кода на языке СИ, его спецификации в виде конечного автомата).

Виртуализация — абстракция над вычислительным процессом, формирующая вычислитель или виртуальную машину, которая определяет полное множество атомарных шагов ВП и позволяет описать любой валидный ВП (в случае если ВП не может быть выражен в рамках вычислителя, это свидетельствует либо о сбое, либо о некорректно выбранной абстракции). Каждому атомарному компоненту ВП соответствует вычислительный механизм (ВМх), обеспечивающий его развертку во времени; ВМх является компонентом вычислителя и реализуется на уровне выделение и вниз до уровня физических процессов (выходящих за границы компетенции специалиста по вычислительной технике). Характер взаимодействия вычислительных механизмов не регламентируется и должен определяться в рамках иных архитектурных стилей [5].

На рис. 2 приведен пример спецификации, выполненной в рассматриваемом архитектурном стиле, а также описание элементов предложенной нотации. Разработанная нотация основана, главным образом, на диаграммах деятельности (Activity Diagram) и диаграммах развертывания (Deployment Diagram) унифицированного языка моделирования (UML). Данный выбор обусловлен наличием развитого инструментария для UML.

Как видно на схеме, конфигурирование блоков обработки данных может осуществляться взаимонезависимо на языке описания аппаратуры (HDL — Hardware Description Language). Их интеграция в TTA-процессор производится на основании описания конфигурации, транслируемого в HDL-код соответствующими инструментальными средствами, что позволяет абстрагироваться от излишних деталей аппаратуры и сосредоточиться на требуемых

функциональных характеристиках процессора. Работа с целевым алгоритмом производится не в терминах организации ВП TTA (пересылки данных между БОД), а на языке высокого уровня.

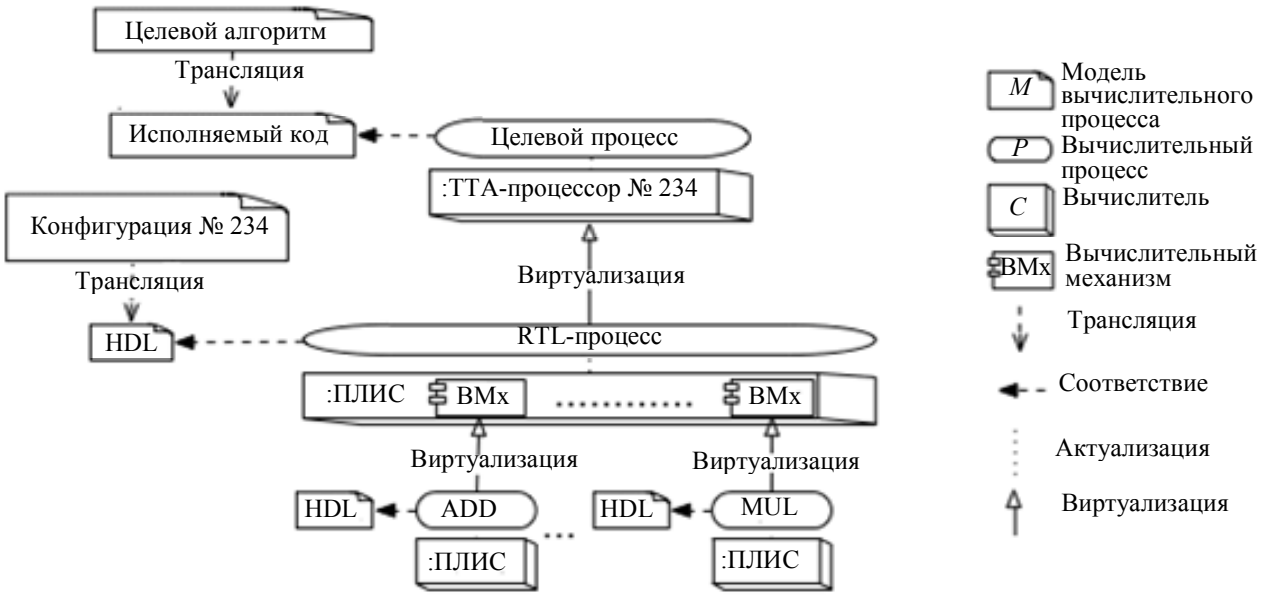

Рис. 2

Разработанный архитектурный стиль позволяет:

— абстрагироваться от разделения на программную и аппаратную составляющие, следовательно, решить проблему раннего их разделения при проектировании;

— различать процессы фазы design-time (трансляции и верификации) и фазы run-time (актуализации и виртуализации) и работать со множеством моделей одного вычислительного процесса, следовательно, проектировать в комплексе целевую ВсС и ее инструментальную цепочку;

— отображать формирование из компонентов нижележащих уровней вычислителей и вычислительных платформ, а также влияние BMx на целевой вычислительный процесс с учетом элементов ВП, которые не могут быть выражены в рамках верхнего уровня.

В совокупности возможности предложенного архитектурного стиля делают его эффективным инструментом для анализа и документирования процесса организации многоуровневых конфигурируемых ВсС. Кроме того, данная спецификация имеет большой потенциал в области построения САПР. В настоящее время ведутся работы по формализации данного архитектурного стиля в виде математического аппарата САПР для построения инструментальных цепочек специализированных процессоров, включающих компиляторы, симуляторы, средства профилирования, средства статической и динамической верификации.

Заключение. Рост степени конфигурируемости ВсС должен быть поддержан соответствующими инструментальными средствами архитектурного уровня, ориентированными на представление взаимосвязей между уровнями конфигурации целевой системы и ее инструментальной цепочкой, программной и аппаратной составляющими системы, фазами ее жизненного цикла. Представленный в статье архитектурный стиль специфицирования ВсС и соответствующая нотация отвечают этим требованиям. Способ декомпозиции ВП на шаги и вычислителя на соответствующие им BMx, используемый в данном архитектурном стиле, позволил положить его в основу САПР для создания инструментальных цепочек специализированных вычислителей, работы над которыми ведутся в настоящее время. Предлагаемая же нотация была апробирована на реальных проектах.

СПИСОК ЛИТЕРАТУРЫ

1. *Qureshi T. N., Törngren M., Persson M., Chen D., Sjöstedt C.-J.* Towards harmonizing multiple architecture description languages for real-time embedded systems // Proc. of Real-Time in Sweden (RTiS'11), June 13—14, 2011. Västerås, Sweden.

2. *Hartenstein R. A* decade of reconfigurable computing: a visionary retrospective // Proc. of the Conf. on Design, Automation and Test in Europe (DATE '01): Piscataway, NJ: IEEE Press, 2001. P. 642—649.

3. *Болгаров И. С., Маковецкая Н. А., Платунов А. Е., Постников Н. Н.* Проектирование приборных контроллеров // Изв. вузов. Приборостроение. 2012. Т. 55, № 10. С. 73—77.

4. *Platunov A., Kluchev A., Penskoi A.* HLD Methodology: The role of architectural abstractions in embedded systems design // Proc. of the 14th Intern. Multidisciplinary Scientific GeoConference on Informatics, Geoinformatics and Remote Sensing (SGEM 2014), Albena, Bulgaria. 2014. P. 209—218.

5. *Garlan D., Bachmann F., Ivers J., Stafford Ju., Bass L., Clements P., Merson P.* Documenting Software Architectures: Views and Beyond. Addison-Wesley Professional, 2010.

6. *Platunov A., Kluchev A., Penskoi A.* Expanding design space for complex embedded systems with HLD-methodology // Proc. of the 6th Intern. Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 6—8 Oct., 2014, St. Petersburg. P. 157—164.

7. STEPS Toward Expressive Programming Systems // Progress Report Submitted to the National Science Foundation (NSF), Oct. 2010; Viewpoints Research Institute, Los Angeles [Электронный ресурс]: <http://www.vpri.org/pdf/tr2010004_steps10.pdf>.

8. *Teich J.* Hardware/Software Codesign: the past, the present, and predicting the future // Proc. of the IEEE. 2012. Vol.100, Special Centennial Iss. P. 1411—1430.

9. *Platunov A., Penskoi A., Kluchev A.* The architectural specification of embedded systems // Proc. of the 3rd Mediterranean Conf. on Embedded Computing (MECO 2014 — Including ECyPS 2014), 2014. P. 48—51.

10. On Efficiency of Transport Triggered Architectures in DSP Applications // Advances in Systems Engineering, Signal Processing and Communications; Ed. *N. Mastorakis*. New York, NY: WSES Press, 2002. P. 25—29.

11. *Partridga C.* Business Object: Re-Engineering for Re-Use. London: BORO Centre, 2005.

Сведения об авторе

Александр Владимирович Пенской — аспирант; Университет ИТМО; кафедра вычислительной техники; E-mail: aleksandr.penskoi@gmail.com

Рекомендована кафедрой вычислительной техники

Поступила в редакцию 13.03.15 г.

Ссылка для цитирования: *Пенской А. В.* Архитектурное документирование встроенных систем с многоуровневой конфигурацией // Изв. вузов. Приборостроение. 2015. Т. 58, № 7. С. 527—532.

ARCHITECTURAL SPECIFICATION OF EMBEDDED SYSTEMS WITH MULTI-LEVEL CONFIGURATION

A. V. Penskoi

ITMO University, 197101, Saint Petersburg, Russia
E-mail: aleksandr.penskoi@gmail.com

An architecture style is developed, description of the style and notation based on UML for specification and analysis of embedded systems in terms of multi-level organization and hardware. Analysis of relationships between the hardware and software components, and the system life cycle stages is presented.

Keywords: configuration, embedded system, architecture, architecture style, architectural notation.

Data on author

Aleksander V. Penskoi — Post-Graduate Student; ITMO University, Department of Computation Technologies; E-mail: aleksandr.penskoi@gmail.com

Reference for citation: *Penskoi A. V.* Architectural specification of embedded systems with multi-level configuration // Izvestiya Vysshikh Uchebnykh Zavedeniy. Priborostroenie. 2015. Vol. 58, N 7. P. 527—532 (in Russian).

DOI: 10.17586/0021-3454-2015-58-7-527-532