АРХИТЕКТУРНЫЕ АБСТРАКЦИИ В ТЕХНОЛОГИИ СКВОЗНОГО ПРОЕКТИРОВАНИЯ ВСТРОЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

А.Е. Платунов

В работе рассматриваются вопросы высокоуровневого проектирования встроенных вычислительных систем (embedded systems). В рамках создания архитектурных моделей таких систем анализируются перспективные абстракции, как известные, так и предлагаемые автором. Формулируется задача создания "действующей" модели встроенной вычислительной системы, инвариантной к способу аппаратно-программной реализации.

Введение

Проектирование встроенных вычислительных систем (ВсС) [1] представляет собой сложную техническую задачу. Процесс проектирования направлен на выбор приемлемого (в смысле целевого критерия) решения при существовании потенциально огромного числа вариантов и ограниченном количестве пробных реализаций. На практике число анализируемых разработчиком вариантов, включая прототипы, ограничивается единицами. Это определяется сжатыми сроками и бюджетами разработок на фоне высокой сложности проектируемой системы, отсутствием эффективных технологий и инструментальных средств [2].

Общая задача в области проектирования ВсС состоит в создании сквозной технологии проектирования ВсС, основанной на формальном представлении системы от этапа исходных спецификаций до реализации.

Задача носит комплексный характер. Ее решение в значительной мере "буксует" из-за отсутствия эффективных моделей архитектурного уровня, которые адекватно отображали бы особенности ВсС [3]. Существующие модели носят последовательный характер и слабо учитывают фактор времени. Они применимы для вычислительных систем (ВС) общего назначения. Использование таких моделей при создании ВсС порождает проблемы в части реактивных свойств системы, ее функциональной надежности, возможности дальнейшей модификации.

В работе анализируются подходы к проектированию ВсС, вводится и обсуждается понятийная база технологии сквозного проектирования ВсС.

Технологии проектирования ВсС

Традиционные подходы к проектированию ВсС. Ключевыми чертами традиционного процесса проектирования микропроцессорных вычислительных систем следует считать:

- ручное разбиение системы на аппаратную и программную части (на основе опыта разработчика) на начальном шаге проектирования;
- раздельное моделирование и последовательное проектирование аппаратуры и программы;
- ручную интеграцию аппаратной и программной частей проекта;
- исправление (компенсацию) выявившихся в процессе отладки ошибок за счет изменения программы (с ухудшением характеристик системы);
- повторное выполнение цикла проектирования при невозможности компенсировать ошибки за счет программной части.

Особенность данного процесса состоит в раннем делении системы на аппаратную и программную составляющие с последующим изолированным их проектированием.

Такой подход приводит к высокой избыточности реализации, выявлению ошибок только в конце – на этапе объединения аппаратуры и программы, устранению ошибок путем практически полного повторного проектирования.

Распространенные технологии проектирования ВсС можно разделить на группы по степени влияния применяемой готовой элементной базы на архитектуру создаваемой системы. Можно говорить о следующих группах: а) проектирование в рамках высокоуровневой системной платформы (на готовой вычислительной системе); б) проектирование на основе набора вычислительных модулей (полузаказное); в) проектирование в условиях свободного выбора элементной базы (заказное).

На практике сегодняшнее массовое проектирование ВсС сводится к полузаказному (модульному) способу или к использованию готовой вычислительной системы.

Заказное проектирование ВсС применяется в случаях создания систем с "нестандартными" характеристиками и при создании образцов для дальнейшего серийного и массового производства.

Дальнейшее применение морально устаревших традиционных технологий проектирования ВсС делает разработки уже в сегодняшних условиях неконкурентоспособными.

Альтернативой являются потенциально реализуемые в современных условиях технологии сквозного параллельного проектирования. Результатом их применения должна стать экономия вычислительных ресурсов (на порядки величины) в сочетании с повышением надежности проектирования и сокращением сроков.

Технология сквозного проектирования ВсС. Центральной идеей сквозного проектирования ВсС является перенос основного объема проектирования на абстрактный уровень. Соответствующая технология может складываться из: а) иерархии архитектурных моделей ВС, инвариантных к аппаратно-программной реализации; б) методики проектирования, направленной на перенос основного объема проектирования на уровень абстрактных вычислительных механизмов; в) инструментальной инфраструктуры, обеспечивающей корректное и эффективное преобразование формальных описаний различного уровня и назначения.

В основе методики сквозного проектирования ВсС [1] лежат следующие положения:

- унификация представления аппаратной и программной частей проекта;
- унификация процесса проектирования аппаратной и программной частей проекта;
- использование сквозного формализованного описания проекта;
- оценка качества проекта на уровнях его абстрактного представления;
- выполнение основного объема отладки проекта на этапах абстрактного представления;
- дополнение архитектурной модели рядом нетрадиционных для этого уровня представления параметров;
- использование расширенной трактовки вычислительной элементной базы.

Важной особенностью технологии сквозного проектирования ВсС является ее принадлежность к заказной модели проектирования в рамках предложенной выше классификации. При этом вторичность выбора элементной базы присутствует даже в тех случаях, когда элемент представляет собой высокоуровневую системную платформу и фактически целиком "закрывает" задачу.

Архитектурные абстракции сквозного проектирования ВсС

Далее мы рассмотрим ряд ключевых абстракций и некоторые важные модели, перспективные, по мнению автора, в контексте развития и конкретизации технологии сквозного проектирования ВсС. Варианты методики сквозного проектирования и инструментальной инфраструктуры оставлены за рамками данной работы.

Вычислительная платформа (ВП). Термин *платформно-ориентированное проектирование* (platform-based design) был введен совсем недавно специалистами в

контексте технологии Codesign [4, 5]. Существуют его различные трактовки. Мы этим термином будем обозначать фиксированную функционально завершенную часть архитектуры ВсС, инвариантную к способу реализации.

Специфицируя ВП на уровне технического задания (ТЗ) или используя этот объект на уровне элементной базы, разработчик может в значительной степени формализовать процесс проектирования ВсС.

ВП можно рассматривать, как: а) класс элементной базы, наряду с микропроцессорами (МП), интерфейсами, операционными системами реального времени (ОСРВ) и другими объектами; б) способ обеспечения программной и аппаратной совместимости; в) базу для повторного использования разработанных компонент; г) абстрактное ядро ряда совместимых ВсС.

Расширенный взгляд на элементную базу ВсС. Привычное понятие элементной базы ВсС в контексте сквозного проектирования нуждается в пересмотре. Повышая уровень абстракции проектирования, разработчик должен сосредоточиться на тех классах элементов, которые значимы, в первую очередь, в рамках архитектурного представления ВсС.

Необходимо изменить подходы к представлению и оценке параметров элементов. На передний план должны выноситься системные параметры, которые позволяют оценить такие стороны использования элемента, как функциональность, системная производительность, надежность, переносимость, степень стандартности. Большое значение приобретают "не технические" в обычном понимании параметры, позволяющие учесть организационные и экономические риски применения элемента, оценить необходимую для применения инфраструктуру.

В качестве новых классов элементной базы ВсС следует рассматривать алгоритмы, программные компоненты (например, драйверы устройств), операционные системы, протоколы, интерфейсы, вычислительные платформы.

Задача описания элементов для использования на высоких уровнях абстракции предполагает унификацию представления независимо от того, к какой из традиционно выделяемых групп – аппаратной или программной, они относятся. Более того, важнейшее значение приобретает категория виртуальных элементов, которые можно определить как параметризуемые объекты с заданной функциональностью и незафиксированным способом реализации.

Встроенное программное обеспечение (embedded software). Важным шагом в области проектирования ВсС явилось определение термина *встроенное программное обеспечение* [3], который фиксирует особенности этой отрасли программирования. Как отмечает один из идеологов этого направления Э.А. Ли в своей работе [6], встроенное программное обеспечение (ПО) предназначено не для трансформации данных (как ПО "обычных" ВС), но для взаимодействия ВС с реальным миром.

Рассматривая организацию встроенного ПО на уровне архитектуры, Э.А. Ли выделяет в качестве основных абстракций программирования своевременность (timeliness), параллельность, живучесть (оригинальное liveness) и неоднородность (heterogeneity).

Существует проблема "программистского перекоса" в проектировании ВсС. Взгляд на проектирование систем реального времени только в виде традиционного программирования (пускай и со всеми этапами, начиная от архитектуры программной надстройки), безусловно, вреден.

Современные аппаратные возможности позволяют реально распространить "программистские" технологии на проектирование аппаратной компоненты ВсС, что является предпосылкой для унификации процесса проектирования системы в целом.

Проектирование систем на кристалле (system on chip - SoC). Одной из составляющих технологии создания ВсС является проектирование сложных цифровых

интегральных микросхем. В этой сфере в наибольшей степени ощущается потребность в технологиях повторного использования разрабатываемого продукта.

Ведущие фирмы в области создания цифровых микросхем и схемотехнических САПР представили ряд стандартов в рамках reuse-технологий [7, 8]. Такие технологии в своей основе содержат иерархию моделей для описания функциональных элементов на последовательных стадиях создания.

Следует отметить, что в reuse-стандартах на сложные аппаратные элементы в наибольшей степени проработаны схемы специфицирования, высокоуровневые модели функционирования и интерфейсные модели. Введены понятия виртуальных компонент и вычислительных моделей, что позволяет разработчикам накапливать не реализации, а механизмы функционирования в абстрактном виде с различной степенью детализации.

Технология создания SoC тесно смыкается с технологией проектирования ВсС. Анализ reuse-технологий этой области выявляет недостаточное развитие в них средств обслуживания программной компоненты. Результатом является "ущемленное положение" программных абстракций по сравнению с аппаратными в инструментарии проектировщика. Это проявляется в виде поиска решения преимущественно в навязанной инструментом "аппаратной" области.

Программирование как универсальный инструмент проектирования ВсС. Поднять эффективность проектирования ВсС возможно за счет формализации высокоуровневых этапов проектирования и унификации вычислительных абстракций. Перспективным следует считать распространение хорошо развитых приемов традиционного программирования на весь цикл проектирования ВсС.

Работы в этом направлении в мире ведутся очень активно. Прежде всего следует отметить, что уже сегодня сколько-нибудь серьезное аппаратное проектирование выполняется языковыми средствами, однако эти средства заметно отличаются от языков высокого уровня для процессоров с последовательной интерпретацией команд. Известны, например, реализации САПР аппаратуры, в которых входное описание очень близко к языку Си. Однако стандартом де-факто является представление результата работы САПР системного уровня в виде совокупности Си и VHDL – текстов, соответственно, для описания программной и аппаратной частей проекта.

Большой интерес представляют в этом смысле синхронный язык Esterel [9] и язык многопланового системного описания SLDL [10], предоставляющие разработчику конструкции, не ориентированные на тот или иной способ реализации непосредственно.

Архитектурные агрегаты (АА). Архитектурная модель ВсС в рамках методики сквозного проектирования [1] может формироваться в двух вариантах: в терминах функциональных преобразователей и виртуальных машин (Ф-модель); в терминах архитектурных агрегатов (А-модель).

Ф-модель может быть представлена в структурном или автоматном стиле и позволяет отразить алгоритм функционирования ВсС, но без многих важных аспектов реализации.

Более полное представление о проектируемой ВсС можно получить посредством А-модели. Базис АА позволяет сгруппировать объекты различных классов по трем направлениям: непосредственно функциональные преобразователи; технические приемы (алгоритмы или механизмы); "не вычислительные" свойства.

Рассмотрим более подробно элементы, лежащие в основе Ф- и А-модели ВсС. Функциональный преобразователь можно определить как объект, который имеет информационные входы, информационные выходы, управляющие входы и характеризуется законом преобразования. Можно различать абстрактные и реальные ФП.

Абстрактный ФП может быть определен как абстрактный вычислительный механизм. Примеры: умножитель, механизм прерывания вычислительного процесса, сторожевой таймер.

Виртуальной машиной будем называть ФП, который позволяет решать логически завершенную задачу пользователя или разработчика ВС. В качестве примеров ВМ можно рассматривать Java-машину, диспетчер вычислительных процессов ОС.

Архитектурные агрегаты – одно из центральных понятий предлагаемой методики. Определенной близостью к АА обладают понятия IP – Intellectual Property modules и VC – Virtual Components [2, 7]. АА рассматривается как абстрактное представление технического решения, которое обеспечивает определенное качество или свойство ВМ. Примеры: арифметико-логический блок, память команд/данных, избыточность кодирования, моноблочная конструкция.

Абстрактные ФП представляют собой одну (центральную) из групп АА. Функциональная классификация ФП, таким образом, рассматривается как базовая классификация "вычислительных" АА. В рамках А-модели базовая классификация дополняется группами, которые позволяют непосредственно отобразить в проекте важные "не вычислительные" аспекты.

Практическое использование А-модели нуждается в формальном описании АА и методике получения метрик. Однако использование А-модели даже без должной степени формализации позволяет разработчику ВсС получить интересные результаты. Далее кратко будут описаны результаты двух проектов, в которых неформальным образом использовались обсуждающиеся в данной работе абстракции и модели.

Построение "действующих" моделей ВсС, инвариантных к способу реализации. Уровень представления (детализации) архитектурной модели ВсС непосредственно определяет ряд важных характеристик проекта. Например, в зависимости от выбранного базиса (конкретный набор АА) будет определяться фактический объем архитектурного этапа проектирования и степень "привязки" результирующей модели к стилю реализации.

Предлагается создавать детальное представление проекта в терминах стандартных механизмов вычислительной системы (стандартные АА). Такое представление может быть получено в результате эволюции архитектурной модели ВсС до определенного уровня детализации многими известными методами.

Целевой функцией такого преобразования является декомпозиция исходной спецификации на элементы базиса вычислительных механизмов с определенными свойствами, с указанием для каждого из элементов конкретного набора параметров. Элементы такого "свободного" базиса должны обладать "прозрачностью" функционирования (т.е. механизмы должны быть стандартными, предполагающими хотя бы одну известную реализацию), но не навязывать конкретный стиль (аппаратный или программный в традиционном понимании) реализации.

В качестве способа описания свободного базиса может быть принят верхний уровень одного из стандартов reuse-технологий или формальное представление АА в рамках А-модели. Формирование такого базиса является на сегодня открытой задачей.

Внедрение "действующих" А-моделей в практику проектирования ВсС предполагает наряду с формальным описанием АА создание плеера (симулятора) для "прогона" свободных базисных моделей ВсС, обладающего широкими возможностями в части параллелизма и временного масштабирования работы модели. Частичным прототипом такой системы можно считать продукты серии "Virtual Component Co-Design" (VCC) фирмы Cadence, однако отсутствие развернутой информации о них не позволяет делать точные оценки.

При всей масштабности задач автоматизации архитектурного проектирования ВсС автору кажется вполне реальным создание экспериментальных САПР этого уровня силами ограниченного научного коллектива.

Опыт использования архитектурных абстракций в проектировании ВсС

Положения и элементы технологии сквозного проектирования ВсС отрабатываются в рамках научно-производственной деятельности фирмы "ЛМТ" (http://lmt.cs.ifmo.ru) при непосредственном участии автора на протяжении ряда лет. Рассмотрим кратко два проекта, о которых дополнительную информацию можно найти на интернет-сайте фирмы и в литературе.

Коммуникационный вычислитель М3М с высокой безопасностью функционирования. Исходные условия проектирования данного изделия формулировались следующим образом: создать одноплатный вычислитель с безопасным процессорным ядром, широким набором двоированных последовательных интерфейсов для сопряжения с модулями верхнего и нижнего уровней иерархии, возможностью ограниченного расширения системных ресурсов, возможностью функционирования стандартной ОСРВ. При этом параметры целевой задачи (например, интенсивности информационных потоков и сложность их обработки) определялись приблизительно.

Было принято решение разработать гетерогенную многопроцессорную ВП со значительными возможностями перераспределения вычислительной нагрузки между "программными" и "аппаратными" процессорами, оснащенную базовой инструментальной инфраструктурой.

Созданная аппаратная ВП, именуемая М3М, содержит три процессора для встроенных применений с архитектурой X86, два массива оперативно программируемой логики фирмы Altera, выделенные CAN-контроллеры и TCP-стеки, ряд массивов оперативной и Flash-памяти, слоты интерфейсов расширения с программируемыми функциями, набор стандартных устройств поддержки реального времени, сервисный процессор питания и термоконтроля.

В основу инструментальной инфраструктуры ВП положен принцип вложенной отладки распределенных ВС, чем обеспечивается эффективный транзитный доступ ко всем программируемым элементам с инструментальной консоли (например, с ПК).

Предложенная аппаратная ВП позволила прикладным проектировщикам реализовать в ограниченные сроки ряд прототипных и целевых вариантов центрального вычислителя линейного пункта системы диспетчерской централизации железнодорожного движения [11]. Следует отметить серьезное разнообразие архитектур разработанных версий вычислителя. Например, объем исходных алгоритмов, реализуемых на аппаратных акселераторах, по отношению к общему объему алгоритмов в различных реализациях изменялся от 10 до 50%.

ВП М3М, дополненная ОС QNX, продолжает активно использоваться в экспериментах по ряду направлений, в частности, на ней отрабатываются адаптивные механизмы надежности и безопасности функционирования.

Многофункциональный контроллер для встроенных применений MEC5091/ADEPT. Экспериментальный проект по созданию такого контроллера начат в 2001 г. Одной из основных задач проекта является практическая проверка эффективности положений и приемов технологии сквозного проектирования ВсС.

Разработана архитектурная модель контроллера, в значительной степени представленная средствами языка UML. Выбран состав периферийных интерфейсов и варианты ресурсов вычислительного ядра. Определены параметры информационных потоков в контроллере. Разработана модель инструментальной инфраструктуры.

Обеспечить создание архитектуры контроллера, независимой в полной мере от доступной элементной базы, к сожалению, в рамках проекта не удалось. Однако перечень вопросов, решенных на этапе архитектурного проектирования, был значительно расширен.

Контроллер создан в виде набора из двух модулей формата PC104 с шифрами MEC5091 и ADEPT. В сумме они представляют собой масштабируемую в широких пределах системную ВП с набором резидентных и внешних (для работы на ПК) программных драйверов, оснащенную развитой инструментальной инфраструктурой.

В настоящее время на ВП MEC/ADEPT реализован контроллер сканирующего зондового микроскопа. Ведется разработка еще ряда целевых систем на базе ВП MEC/ADEPT.

Заключение

Растущая потребность в ВсС в мире определяет направленность усилий специалистов на создание промышленной технологии их проектирования.

На повестке дня стоят задачи формализации ранних стадий проектирования ВсС, совершенствования и развития абстракций такого проектирования.

К сожалению, активность отечественных специалистов в данном направлении, судя по публикациям, отсутствует.

Автор надеется, что высказанные в работе мысли будут способствовать как решению конкретных технических проблем, так и интеграции российских специалистов в процесс создания передовых технологий не только в области офисного программного обеспечения.

Литература

1. Архитектурная модель цифровых вычислительных систем для встроенных применений. / Платунов А.Е. // Изв. вузов. Приборостроение. 2001. Т.44. №3. С. 8–15.
2. Hardware-Software Codesign. // IEEE Design & Test of Computers, January – March 2000, pp. 92 – 99.
3. Embedded Software – An Agenda for Research. / E.A. Lee // Technical Memorandum UCB/ERL M99/63, University of California, Berkeley, CA 94720, December 15, 1999.
4. System Design: Traditional Concepts and New Paradigms. / A. Ferrari, A. Sangiovanni-Vincentelli // Proceedings of the 1999 Int. Conf. On Comp. Des., Austin, Oct. 1999.
5. System Level Design: Orthogonalization of Concerns and Platform-Based Design. / K. Keutzer, S. Malik, R. Newton, J.Rabaey, A. Sangiovanni-Vincentelli // IEEE Transactions on Computer-Aided Design of Circuits and Systems, Vol. 19, No. 12, December, 2000.
6. Embedded Software. / E.A. Lee // Technical Memorandum UCB/ERL M01/26, University of California, Berkeley, CA 94720, November 1, 2001.
7. VSI Alliance. Model Taxonomy. Version 2.1 / URL: http://www.vsi.org.
8. URL: http://www.gigascale.org.
9. The Esterel synchronous programming language: Design, semantics, implementation. / G. Berry, G. Gonthier // Science of Programming, 1992, vol. 19, № 2, pp. 87 – 152.
10. Get a handle on design languages. / G. Moretti // URL: http://www.ednmag.com/ednmag/reg/2000/06052000/12cs.htm
11. Комплекс технических средств для систем железнодорожной автоматики / Гавриков В.О., Платунов А.Е., Никифоров Н.Л. // Автоматика, телемеханика и связь. 1998. №11. С. 5–10.