УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
июль–август 2022 Том 22 № 4 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
Jule–August 2022 Vol. 22 No 4 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

doi: 10.17586/2226-1494-2022-22-4-716-724
УДК 004.2

Усиление роли микроархитектурных этапов проектирования встраиваемых систем

Максим Вячеславович Кольчурин[1], Василий Юрьевич Пинкевич[2]✉, Алексей Евгеньевич Платунов[3]

[1,3] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[2] ООО «ЛМТ», Санкт-Петербург, 199034, Российская Федерация

[1] maxim.kolchurin@gmail.com, https://orcid.org/0000-0002-7061-9357
[2] vasiliy.pinkevich@yandex.ru✉, https://orcid.org/0000-0002-8635-5026
[3] aeplatunov@gmail.com, https://orcid.org/0000-0003-3003-3949

Аннотация

Предмет исследования. Растущее разнообразие вычислительных систем, стремительное увеличение их сложности, интеграции в объекты и процессы физического мира требуют резкого повышения производительности труда их создателей. Отмечено, что качество, сроки и степень повторного использования результатов проектирования в области информационных технологий сильно зависят от методологий и маршрутов проектирования на этапах выбора и/или создания стеков платформ, технологий и инструментов. Важнейшую роль в этом играют способы описания организации вычислительной системы на различных уровнях и применяемые системы абстракций. Проблема заполнения семантического разрыва между концептуальным (архитектурным) уровнем и уровнями реализации по-прежнему стоит очень остро. Следовательно, требуется создание промышленных методик и инструментов проектирования на «промежуточных» уровнях. **Метод.** В работе предложены способы представления проектных решений, которые направлены на целостное, сквозное описание как логики организации вычислительного процесса, так и шагов, технологий и инструментов процесса проектирования. **Основные результаты.** Подробно объяснено наполнение и обоснована необходимость этапов микроархитектурного проектирования вычислительных систем. Введена классификация проектов в области информационных технологий по степени вариативности проектной платформы. Предложен ряд понятий для представления набора абстракций микроархитектурного проектирования в рамках проектов с большой внутренней вариативностью. Подробно описаны следующие абстракции: проектное и аспектное пространства, проектные платформы и кросс-уровневые механизмы. **Практическая значимость.** Рассмотрены примеры представления ряда предложенных абстракций (рабочих инструментов документирования) микроархитектурных этапов проектирования, которые наиболее актуальны в проектировании вычислительных систем в модели «ограниченных ресурсов»: встраиваемые, киберфизические системы, «граничные» и «туманные» уровни систем интернета вещей.

Ключевые слова

встраиваемая система, микроархитектура, исследование пространства проектных решений, аспектное проектирование, кросс-уровневые механизмы

Ссылка для цитирования: Кольчурин М.В., Пинкевич В.Ю., Платунов А.Е. Усиление роли микроархитектурных этапов проектирования встраиваемых систем // Научно-технический вестник информационных технологий, механики и оптики. 2022. Т. 22, № 4. С. 716–724. doi: 10.17586/2226-1494-2022-22-4-716-724

Strengthening the role of microarchitectural stages of embedded systems design

Maxim V. Kolchurin[1], Vasiliy Yu. Pinkevich[2]✉, Alexey E. Platunov[3]

[1,3] ITMO University, Saint Petersburg, 197101, Russian Federation
[2] LMT Ltd., Saint Petersburg, 199034, Russian Federation

[1] maxim.kolchurin@gmail.com, https://orcid.org/0000-0002-7061-9357
[2] vasiliy.pinkevich@yandex.ru✉, https://orcid.org/0000-0002-8635-5026
[3] aeplatunov@gmail.com, https://orcid.org/0000-0003-3003-3949

Abstract

The growing variety of computing systems, the rapid increase in their complexity, their integration into objects and processes of the physical world require a dramatic increase in the productivity of their creators. It is noted that the quality, timing, and degree of reuse of design results in the field of information technologies strongly depend on design methodologies and routes at the stages of choosing and/or creating stacks of platforms, technologies and tools. The most important role belongs to the ways of describing the organization of the computing system at various levels and to the used systems of abstractions. The problem of filling the semantic gap between the conceptual (architectural) level and the implementation levels is still very acute. So, it requires the creation of industrial techniques and design tools at these "intermediate" levels. The paper suggests ways of presenting design solutions that are aimed at a holistic, end-to-end description of both the logic of the computing process organization and the steps, technologies, and tools of the design process. The content and the necessity of the stages of microarchitectural design of computing systems is justified and explained in detail. Classification of projects in the field of information technologies according to the degree of variability of the project platform is introduced. Several concepts representing a set of abstractions for microarchitectural design within projects with great variability are suggested. The following abstractions are described in detail: project, design and aspect spaces, project platforms and cross-level mechanisms. Examples of several proposed abstractions presentations (design documentation tools) of microarchitectural design stages are discussed that are most relevant in the design of computing systems in the "limited resources" model: embedded systems, cyber-physical systems, "edge" and "fog" levels of Internet of Things systems.

Keywords

embedded system, microarchitecture, design space exploration, aspect-based design, cross-level mechanisms, cross-cutting mechanisms

For citation: Kolchurin M.V., Pinkevich V.Yu., Platunov A.E. Strengthening the role of microarchitectural stages of embedded systems design. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2022, vol. 22, no. 4, pp. 716–724 (in Russian). doi: 10.17586/2226-1494-2022-22-4-716-724

Введение

Заказное проектирование встраиваемых систем, к которым можно отнести любые специализированные вычислительные системы [1], необходимо в тех случаях, когда прикладную задачу невозможно решить с требуемым качеством на базе готовых компонентов и платформ прикладного программирования. Таким образом, в заказном проектировании ключевую роль играет создание эффективной платформы, с использованием которой можно выполнить все требования к системе.

Проектные платформы разного уровня гранулярности предоставляемого набора функций следует считать одной из важнейших абстракций в создании встраиваемых систем. Под платформой понимается вычислительная сущность (программная, аппаратная), которая имеет зафиксированный разработчиком набор выполняемых операций (в общем случае — набор технических решений), фиксируется как важное («опорное») решение в текущем проекте и/или нацелена на повторное использование [2–4]. Платформа обычно обеспечивает новый уровень абстракции относительно тех средств, с помощью которых она сама построена.

Внешние требования к платформе (набор операций, производительность, ограничения на использование вычислительных ресурсов и т. п.) обычно известны и следуют из особенностей решаемых прикладных задач. При этом требования к внутренней организации платформы определяются самим разработчиком. Выбор и разработка проектных платформ — наиболее трудоемкая часть заказного проектирования встраиваемых систем.

По аналогии с делением, принятым в проектировании микропроцессоров, будем относить проектирование, не затрагивающее внутреннее устройство платформ, к архитектурному уровню, а проектирование самой платформы — к микроархитектурному.

Число методик и инструментальных средств, направленных на повышение производительности труда разработчиков в области микроархитектурного проектирования крайне невелико. В настоящее время значительные успехи имеются в развитии языков программирования, средств проектирования заказных микросхем, печатных плат и других важных, но все же частных задач. Методики и инструментальные средства проектирования системного уровня также сконцентрированы в основном на частных задачах (анализ требований, функциональное моделирование, верификация), и не предлагают полноценного автоматизированного маршрута [5–7][1].

При этом ситуация на рынке систем и услуг в сфере информационных технологий показывает, что заказное проектирование встраиваемых систем массово выполняется небольшими командами разработчиков в условиях ограниченных бюджетов. Методики и инструменты, включающие архитектурный/системный уровень про-

[1] Bonnet S., Lestideau F., Voirin J.-L. Arcadia and Capella on the Field: Real-World MBSE Use Cases // MBSE Symposium, Canberra, October 27th, 2014.

ектирования, применяются в недостаточном объеме из-за их высокой стоимости, трудоемкости использования, недостаточной квалификации разработчиков (высокий «порог вхождения») [1, 8, 9]. В основном проектирование выполняется непосредственно в элементах и шаблонах, навязанных языками, технологиями и инструментами рабочего проектирования. Это приводит к слабому исследованию пространства проектных решений (потенциальные варианты организации вычислительной системы генерируются и анализируются крайне ограниченно), невозможности проведения полноценной верификации архитектурных решений, и, как следствие, к снижению общего качества проектирования и достигаемых характеристик систем. Отметим, что существующие средства системного проектирования малопригодны для массового разработчика из-за слишком высокого отношения стоимости их применения к эффективности.

Для повышения эффективности проектирования требуется создание иерархии описаний разной степени детализации и использование итеративного последовательного приближения. В массовом проектировании этот подход недооценивается многими, в том числе отечественными разработчиками, крайне слабо поддержан методиками и инструментальными средствами. Заказное проектирование сложных встраиваемых систем с полным циклом (full-stack) во многом остается творческим процессом, результат которого в наибольшей степени зависит от опыта и квалификации членов команды.

Это ставит общую задачу расширения этапа высокоуровневого [10] проектирования встраиваемых систем. Одним из решений видится включение микроархитектурного этапа проработки проекта в процесс проектирования в качестве обязательной части.

В рамках данного этапа необходимы относительно простые и доступные для массового применения методики и автоматизированные средства проектирования встраиваемых систем, закрывающие семантический разрыв между высокоуровневым архитектурным представлением и реализацией. Заметим, что может быть необходимо говорить о целом ряде этапов в проектировании микроархитектуры.

В работе представлены модели, механизмы и шаблоны для этапов микроархитектурного проектирования, а также примеры их представления и применения.

Проблема выбора платформ для реализации встраиваемых систем

Введем классификацию вычислительных платформ по внутренней вариативности — размеру доступного для исследования пространства проектных решений, т. е. по количеству различных вариантов реализации системы, удовлетворяющих конкретному техническому заданию, которые можно создать с использованием данной платформы: малая (Группа 1), средняя (Группа 2) и высокая внутренняя вариативность (Группа 3).

Приведем примеры платформ для данных групп.

Группа 1 — программируемые логические контроллеры, а также универсальные вычислительные машины вообще, которые поддерживают стандартные высокоуровневые языки программирования и стандартные интерфейсы подключения устройств.

Группа 2 — контроллеры с программированием в среде операционных систем реального времени. Их внутренняя вариативность остается для пользователя относительно небольшой за счет широкого, но все же ограниченного набора предусмотренных механизмов управления вычислениями и драйверов устройств.

Группа 3 — заказные устройства с заказным программным обеспечением, возможностью использования специализированных архитектур на программируемой логике и заказных микросхемах.

Данная классификация демонстрирует один из важнейших компромиссов при выборе платформы реализации встраиваемых систем — соотношение между уровнем операций, реализуемых платформой (вычислительные абстракции, система команд, программный интерфейс приложения (API) или элементная база проектировщика), и эффективностью вычислителя/платформы, реализующей этот уровень (рис. 1) [11]. Отметим, что в отличие от примеров проектов Группы 1, примеры проектов Групп 2 и 3 предполагают смешанные аппаратно-программные разработки.

Повышение уровня абстракции позволяет сконцентрироваться на реализации прикладных требований, увеличить переносимость, а также снизить требования к квалификации разработчика. Часто это проявляется в реализации наиболее простых, шаблонных, а также «лобовых» вариантов в модели «условно неограниченных ресурсов», что оказывается приемлемо для многих применений. Если же такой сравнительно простой путь решения задачи не дает требуемого результата, необходимо переходить на сценарии полномасштабного заказного проектирования. При увеличении доли заказного проектирования возможно формирование более эффективных прикладных платформ и проведение «глубокой» оптимизации их микроархитектуры, реализация более сложных и эффективных решений. При этом необходим учет ограниченности доступных вычислительных ресурсов для их полноценного использования.

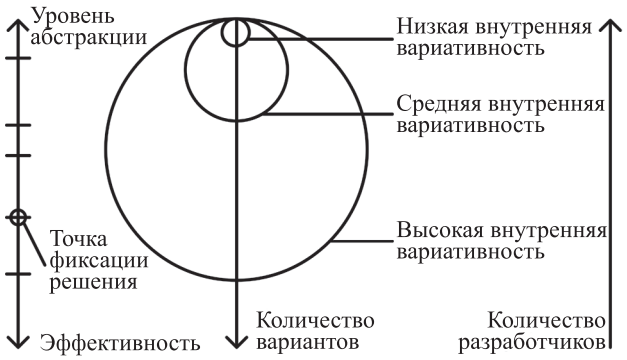

Рис. 1. Классификация проектных платформ по степени внутренней вариативности

Fig. 1. Classification of project platforms according to the degree of internal variability

Чем выше внутренняя вариативность платформы, тем сложнее ее использование, меньше экосистема и число разработчиков, обладающих необходимыми компетенциями, ниже уровень автоматизации проектирования и документирования объектов повторного использования.

Микроархитектурное проектирование

Основные задачи микроархитектурного проектирования: формирование, визуализация и документирование пространства проектных решений; генерация проектных решений и исследование пространства проектных решений; верификация и отбор проектных решений.

В основе предлагаемого подхода к микроархитектурному проектированию лежит аспектный подход. Он помогает определить структуру пространства проектных решений за счет выделения сквозных «зон ответственности» в системе и сквозных задач в процессе проектирования. В данном пространстве существуют вычислительные и иные системные механизмы — основные единицы, из которых создаются более сложные платформы и целые вычислительные системы.

Природа большинства вычислительных систем имеет ярко выраженную уровневую организацию. Механизмы и платформы как основные абстрактные единицы проектирования могут принадлежать к конкретному уровню или охватывать несколько уровней организации вычислительной системы. Для документирования уровневой организации вычислительной системы и/или ее пространства проектных решений мы предлагаем использовать граф актуализации и производные модели.

Выделение корневых компонентов («кернелов») в проекте системы позволяет в явном виде определять и балансировать трудозатраты на проектирование компонент «внутреннего» и «внешнего» назначений, группировать информацию о сквозных механизмах и подсистемах, представлять описания разного уровня детализации для последующего повторного использования.

Структура проектного и аспектного пространства системы

Назовем проектным пространством системы совокупность активностей и артефактов на фазе проектирования в жизненном цикле системы.

В процессе проектирования исходные требования технического задания дополняются внутренними требованиями коллектива разработчиков. Эти требования могут быть производными от исходных, а могут быть навязаны какими-либо сторонними обстоятельствами (необходимость экономии времени, отсутствие некоторых компетенций, создание задела на будущее и др.). В результате формируется полный набор требований к создаваемой встраиваемой системе и сопутствующим артефактам.

Выделим две основные группы требований:

1) «поведенческие» требования — непосредственно касаются функционирования системы во всех проявлениях (реализуемый вычислительный процесс, временные характеристики, надежность, энергопотребление, тепловыделение и др.);
2) «организационные» требования — касаются вопросов, связанных с разными стадиями жизненного цикла как конкретных экземпляров системы, так и совокупности активностей по созданию таких систем. Включают требования к организации процессов проектирования, производства, тестирования, обслуживания, ремонта, вывода системы из эксплуатации, возможностей устаревания, модернизации, повторного использования, развития системы, экономической эффективности и др.

Поведенческие требования можно разделить на прикладные (связанные с выполнением прикладной целевой функции системы, выраженной в исходных требованиях заказчика) и системные (относящиеся к решению задач, поставленных разработчиками: организация платформы для решения непосредственно прикладных задач, инструментально-диагностическая инфраструктура системы и др.).

Всем перечисленным выше видам требований соответствуют аспекты проектирования системы. Аспект определяется как сегмент проектного пространства и/или жизненного цикла системы и проекта, связанный с какой-либо проблемой или группой требований. Аспекты могут быть концептуальными (актуальными на протяжении всего жизненного цикла) и локальными, выделяемыми на какое-то время. Их совокупность определяет аспектное пространство проекта. Организационные аспекты порождают новые требования в поведенческих аспектах и ограничивают пространство проектных решений, в котором проектировщик создает систему.

Маршрут проектирования сложных вычислительных систем должен поддерживать в единой системе абстракций иерархическое представление вычислительных платформ, языков описания функциональности и их трансляторов, прикладной надстройки. Центральной идеей выступает последовательное уточнение/проработка целевой системы через иерархию шагов маршрута проектирования с понижением степени абстракции [12].

Аспекты делят проектное пространство «вертикально», пронизывая все компоненты и уровни организации системы, которые, в свою очередь, задают «горизонтальное» деление. Аспекты жизненного цикла пронизывают различные этапы во времени.

Для обеспечения эффективного сочетания уровневого представления с аспектным подходом мы предлагаем проектировщику работать со следующими моделями: аспектные стеки логических, функциональных, структурных уровней организации анализируемой или проектируемой системы (аспекты организации системы); аспектные стеки технологических уровней создания системы (инструментально-технологические аспекты); аспектные временные треки в рамках маршрута проектирования, где шаги могут быть сопоставлены с проработкой уровней организации системы (отражается логика проектирования). При этом последовательность проработки уровней может осознанно меняться проектировщиком.

Сложность современных встраиваемых систем, требования к их масштабируемости (горизонтальной и вертикальной) делают актуальными модели с несколькими аспектными уровневыми стеками, «растущими» с выделенного уровня (горизонтальная модель организации в пределах одного или нескольких уровней системы).

Кросс-уровневые механизмы в проектировании встраиваемых систем

Реализацию встраиваемой системы представим как совокупность вычислительных и иных механизмов — особых объектов в составе системы, решающих конкретную задачу известным способом [10, 12]. Механизмы могут быть разной сложности и гранулярности (от простых математических функций до виртуальных машин), применяться для обеспечения как функциональных, так и нефункциональных требований к системе (надежности, работы в реальном времени и др.). Такое представление наиболее эффективно на этапах микроархитектурного проектирования.

Задачи, решаемые большинством встраиваемых систем, характеризуются высокими требованиями к безопасности, надежности и времени исполнения. Это заставляет создавать значительную часть принципиально важных механизмов встраиваемых систем: кросс-уровневыми — реализованными с использованием более чем одной проектной платформы; кросс-доменными — сочетающими в себе элементы разных моделей вычислений в пределах одного уровня; кросс-аспектными — отражающимися на разных группах требований к системе (на производительности, энергопотреблении и др.).

Комплексный характер механизмов влияет на способ и качество подачи информации в документах на компоненты, особенно повторно используемые. В сегменте проектов с малой вариативностью (Группа 1) для вычислителей с условно безграничными ресурсами поставщики предлагают описания высокого качества для «прикладного» разработчика. В данном случае предоставляются абстрактные модели, которые напрямую поддерживаются наборами библиотек (например, Windows Driver Kit включает Kernel-Mode Driver Architecture Design Guide, Windows Driver Model (WDM) и др.). В проектах Группы 2 и особенно Группы 3 разработчики встраиваемых систем и аппаратных вычислительных платформ вынуждены использовать фрагментарные и преимущественно низкоуровневые описания отдельных вычислителей (групп вычислителей), периферийных контроллеров, IP-ядер или программных компонентов с целью реализации в равной степени функциональных и нефункциональных требований, опираясь на модель максимального использования ресурсов проектируемой системы.

Наибольшие сложности при проектировании вызывает именно кросс-уровневая природа механизмов. Описание такого механизма в рамках единой платформы и единого языка отсутствует, что порождает целый ряд проблем: необходимо использовать несколько фрагментов описания механизма в рамках разных моделей вычислений; при отсутствии документации на реализацию системы свойства механизма и даже само его существование для разработчика часто являются неочевидными (требуется обратная разработка системы, reverse engineering); затруднена проверка корректности реализации механизма; при замене даже одной из платформ на новую, несовместимую со старой (например, переход на другую линейку микроконтроллеров), необходимо пересматривать реализацию множества стеков механизмов или кросс-уровневых механизмов для сохранения корректности и эффективности функционирования, что может привести к необходимости пересмотра микроархитектуры системы в целом.

Примеры микроархитектурного представления проектных решений

На рис. 2 показан пример комплексной **аспектной диаграммы** типичного проекта встраиваемой системы с аспектными стеками уровней и временными треками. Диаграмма приведена в общем виде и по необходимости может быть детализирована, дополнена локальными аспектами.

Корневые механизмы (кернелы) представляют собой базовые, критически важные кросс-уровневые и кросс-доменные механизмы или их совокупности, предназначенные для активного повторного использования. Мы предлагаем фиксировать информацию о кернелах на трех основных уровнях документирования: функциональный интерфейс (функциональность и способы сопряжения с компонентами того же уровня), микроархитектурная спецификация в терминах механизмов и конечная реализация. Это обеспечивает эффективное повторное использование кернелов на микроархитектурных этапах проектирования. Совокупности кернелов, образующие цельную платформу, назовем микроархитектурными шаблонами. На рис. 3 показан пример представления кернелов в составе микроархитектурного шаблона с указанием охваченных уровней (платформ) встраиваемых систем. Вопросы вариантов реализации кернелов выходят за рамки настоящей работы, некоторые примеры можно найти в работе [12].

Граф актуализации вычислительного процесса [3, 4] в едином стиле в виде последовательности унифицированных трансляторов описывает создаваемую встраиваемую систему, используемый при этом инструментарий и реализуемый процесс проектирования. Это позволяет в рамках одной модели продемонстрировать взаимосвязи между аппаратной и программной частями, а также фазами проектирования, конфигурирования и исполнения вычислительного процесса (design-, config- и run-time).

Приведем пример использования графа актуализации для демонстрации зависимостей, которые возникают при реализации кросс-уровневых механизмов. На рис. 4, *а* показан ряд вариантов кросс-уровневой реализации одного из кернелов — драйвера UART. Если при проектировании драйвера закладывать возможность использования аппаратного буфера, то необходимо предусматривать программную адаптацию к буферам разного размера. Если же игнорировать возможности аппаратной буферизации, драйвер будет

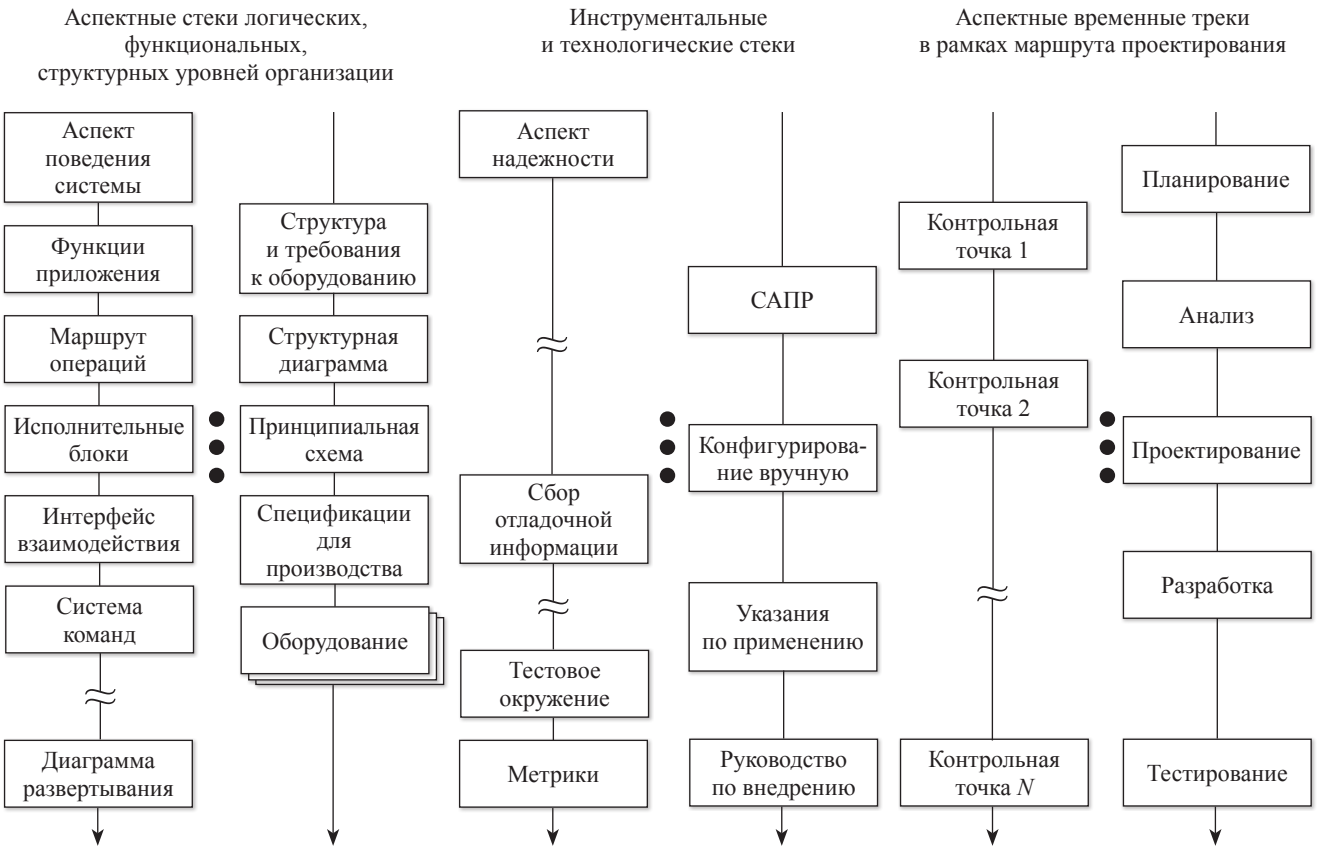

Рис. 2. Пример аспектной диаграммы проекта встраиваемой системы

Fig. 2. Example of an aspect diagram of an embedded system project

менее эффективно использовать процессорное время, но его проще переносить на микроконтроллер с другой периферией. В пределе аппаратная часть UART может быть сведена к набору дискретных портов ввода-вывода. На рис. 4, *b* представлена реализация виртуального датчика освещенности на базе цифровой камеры. Такая реализация не всегда переносима на другую модель камеры даже при использовании общего API, поскольку для получения информации об освещенности матрицы используются команды прямого доступа к регистрам модуля камеры.

Коллектив, который представляют авторы, в формализованном виде успешно использовал представленный подход в целом ряде проектов, среди которых автоматизированная система кросс-уровневого тестирования [13], фреймворк микроархитектурного проектирования

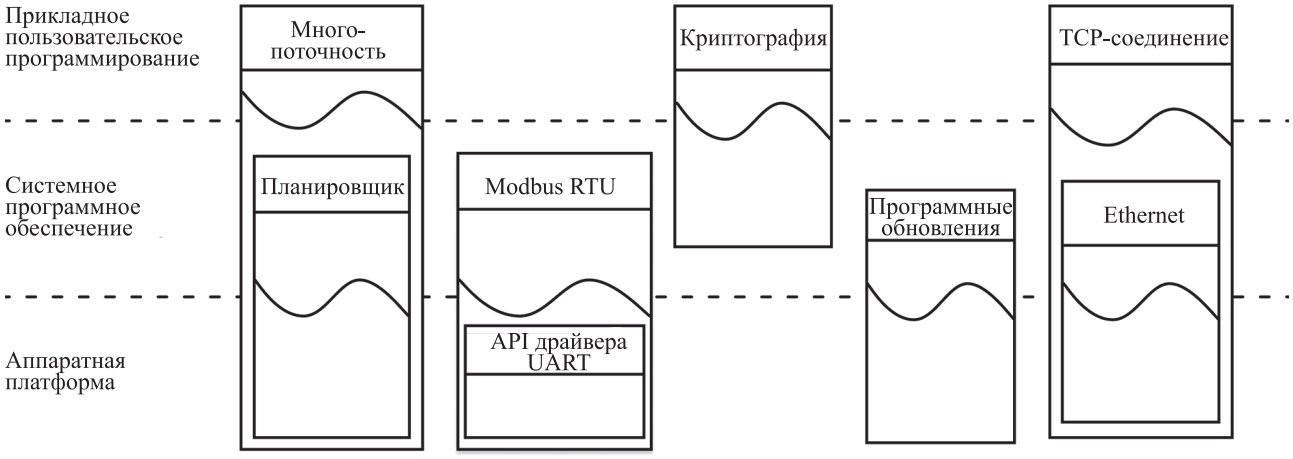

Рис. 3. Пример распределения кернелов по уровням микроархитектурного шаблона (UART — контроллер последовательного ввода-вывода)

Fig. 3. Example of kernels mapping by levels of microarchitectural pattern (UART — serial input-output controller)

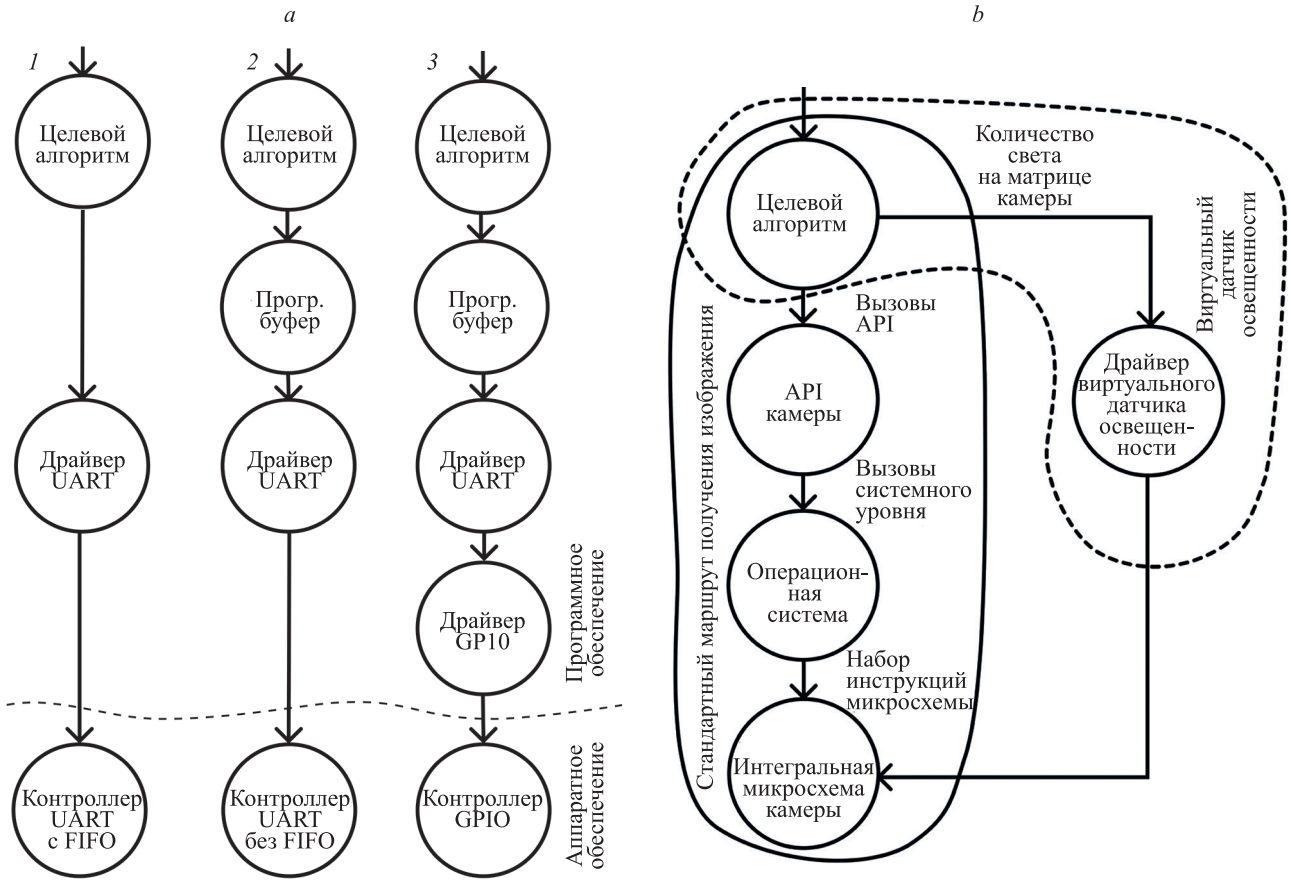

Рис. 4. Варианты графа актуализации для драйвера UART; варианты программно-аппаратного разделения функциональности: с использованием аппаратной буферизации данных (*1*) и только программной буферизации (*2*), с полностью программной реализацией приемопередатчика (*3*) (*a*); виртуального датчика освещенности (*b*)

Fig. 4. Variants of the actualization graph for UART driver; variants of hardware-software partitioning of functionality: using hardware data buffering (*1*) and only software buffering (*2*), with a fully software implementation of the transceiver (*3*) (*a*) and virtual illumination sensor (*b*)

процессорных ядер СнК [14], инструменты проектов NL3 и NITTA [15].

Заключение

В области информационных технологий по-прежнему остро стоит проблема повышения эффективности труда разработчиков. Разумный баланс между готовыми («с полки») и заказными решениями — «головная боль» всех архитекторов и тим-лидеров. Как показывает практика, в данной отрасли внедрение новых приемов и методов на концептуальных, системных этапах проектирования для выработки и принятия не организационных, а непосредственно технических решений, занимает 10–20 и более лет. И это определяется не инертностью индустрии, а сложным процессом подготовки («созревания») специалистов. Тем важнее новые предложения в данной сфере, больше потребность в их обсуждении и экспериментальном внедрении.

В сегменте встраиваемых систем предложенное деление проектов на ряд категорий позволило выделить группу, для которой заказные решения однозначно являются предпочтительными. Однако для таких проектов катастрофически не хватает эффективных технологий и инструментов проектирования.

Повысить степень формализации проектирования позволяет приведенный в работе общий взгляд на этапы высокоуровневого проектирования ответственных (критичных) вычислительных систем в сочетании с конкретными предложениями в части инструментов описания вычислительной системы на разных уровнях абстракции.

Представленные абстракции могут быть использованы как на уровне методологических решений в качестве «ручных приемов», так и в качестве алгоритмической основы в составе фреймворков и систем автоматизированного проектирования.

Исследования по теме работы находятся в активной фазе. Одна из актуальных задач — применение техники кросс-уровневых механизмов и микроархитектурных шаблонов для анализа сложного встроенного программного обеспечения (различные операционные системы реального времени, стеки коммуникационных протоколов, дистанционное конфигурирование и обновление, и др.).

Литература

1. Платунов А.Е., Пинкевич В.Ю. Создание киберфизических систем: проблемы подготовки ИТ-специалистов // Control Engineering Россия. 2021. № 3. С. 64–70.
2. Sangiovanni-Vincentelli A., Martin G. Platform-based design and software design methodology for embedded systems // IEEE Design and Test of Computers. 2001. V. 18. N 6. P. 23–33. https://doi.org/10.1109/54.970421
3. Platunov A., Penskoi A., Kluchev A. The architectural specification of embedded systems // Proc. of the 3rd Mediterranean Conference on Embedded Computing (MECO). 2014. P. 48–51. https://doi.org/10.1109/MECO.2014.6862656
4. Пинкевич В.Ю., Платунов А.Е. Тестирование и отладка встраиваемых вычислительных систем на основе уровневых моделей // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 5. С. 801–808. https://doi.org/10.17586/2226-1494-2018-18-5-801-808
5. Teich J. Hardware/software codesign: the past, the present, and predicting the future // Proceedings of the IEEE. 2012. V. 100. P. 1411–1430. https://doi.org/10.1109/JPROC.2011.2182009
6. Broman D., Lee A., Tripakis S., Törngren M. Viewpoints, formalisms, languages, and tools for cyber-physical systems // Proc. of the 6th International Workshop on Multi-Paradigm Modeling (MPM'12). 2012. P. 49–54. https://doi.org/10.1145/2508443.2508452
7. Clements P., Bachmann F., Bass L., Garlan D., Ivers J., Little R., Merson P., Nord R., Stafford J. Documenting Software Architectures. Views and Beyond / 2nd ed. Pearson Education, 2011. 608 p.
8. Masin M., Palumbo F., Myrhaug H., de Oliveira Filho J.A., Pastena M., Pelcat M., Raffo L., Regazzoni F., Sanchez A.A., Toffetti A., de la Torre E., Zedda K. Cross-layer design of reconfigurable cyber-physical systems // Proc. of the 20th Design, Automation and Test in Europe Conference and Exhibition (DATE). 2017. P. 740–745. https://doi.org/10.23919/DATE.2017.7927088
9. Pohlmann U. A Model-driven Software Construction Approach for Cyber-physical Systems. Universität Paderborn, 2018.
10. Platunov A., Kluchev A., Penskoi A. Expanding design space for complex embedded systems with HLD-methodology // Proc. of the 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). 2014. P. 157–164. https://doi.org/10.1109/ICUMT.2014.7002096
11. Bailey B., Martin G. ESL Models and their Application: Electronic System Level Design and Verification in Practice. New York: Springer Publication, 2010. XXIV, 446 p. https://doi.org/10.1007/978-1-4419-0965-7
12. Pinkevich V., Platunov A., Gorbachev Y. Design of embedded and cyber-physical systems using a cross-level microarchitectural pattern of the computational process organization // CEUR Workshop Proceedings. 2020. V. 2893.
13. Pinkevich V., Platunov A. Model-driven functional testing of cyber-physical systems using deterministic replay techniques // Proc. of the 1st IEEE International Conference on Industrial Cyber-Physical Systems (ICPS). 2018. P. 141–146. https://doi.org/10.1109/ICPHYS.2018.8387650
14. Antonov A., Kustarev P., Bikovsky S. MLIP cores: Designing hardware generators with programmable microarchitectural mechanisms // Proc. of the 52nd IEEE International Symposium on Circuits and Systems (ISCAS). 2020. P. 9180593. https://doi.org/10.1109/ISCAS45731.2020.9180593
15. Pinkevich V.Y., Platunov A.E., Penskoi A.V. The approach to design of problem-oriented reconfigurable hardware computational units // Proc. of the Wave Electronics and its Application in Information and Telecommunication Systems (WECONF). 2020. P. 9131512. https://doi.org/10.1109/WECONF48837.2020.9131512

References

1. Platunov A.E., Pinkevich V.Yu. Creation of cyber-physical systems: problems of it specialists training. *Control Engineering Russia*, 2021, no. 3, pp. 64–70. (in Russian)
2. Sangiovanni-Vincentelli A., Martin G. Platform-based design and software design methodology for embedded systems. *IEEE Design and Test of Computers*, 2001, vol. 18, no. 6, pp. 23–33. https://doi.org/10.1109/54.970421
3. Platunov A., Penskoi A., Kluchev A. The architectural specification of embedded systems. *Proc. of the 3rd Mediterranean Conference on Embedded Computing (MECO)*, 2014, pp. 48–51. https://doi.org/10.1109/MECO.2014.6862656
4. Pinkevich V.Yu., Platunov A.E. Testing and debugging of embedded computing systems based on level models. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2018, vol. 18, no. 5, pp. 801–808. (in Russian). https://doi.org/10.17586/2226-1494-2018-18-5-801-808
5. Teich J. Hardware/software codesign: the past, the present, and predicting the future. *Proceedings of the IEEE*, 2012, vol. 100, pp. 1411–1430. https://doi.org/10.1109/JPROC.2011.2182009
6. Broman D., Lee A., Tripakis S., Törngren M. Viewpoints, formalisms, languages, and tools for cyber-physical systems. *Proc. of the 6th International Workshop on Multi-Paradigm Modeling (MPM'12)*, 2012, pp. 49–54. https://doi.org/10.1145/2508443.2508452
7. Clements P., Bachmann F., Bass L., Garlan D., Ivers J., Little R., Merson P., Nord R., Stafford J. *Documenting Software Architectures*. Views and Beyond. 2nd ed. Pearson Education, 2011, 608 p.
8. Masin M., Palumbo F., Myrhaug H., de Oliveira Filho J.A., Pastena M., Pelcat M., Raffo L., Regazzoni F., Sanchez A.A., Toffetti A., de la Torre E., Zedda K. Cross-layer design of reconfigurable cyber-physical systems. *Proc. of the 20th Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2017, pp. 740–745. https://doi.org/10.23919/DATE.2017.7927088
9. Pohlmann U. *A Model-driven Software Construction Approach for Cyber-physical Systems*. Universität Paderborn, 2018.
10. Platunov A., Kluchev A., Penskoi A. Expanding design space for complex embedded systems with HLD-methodology. *Proc. of the 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2014, pp. 157–164. https://doi.org/10.1109/ICUMT.2014.7002096
11. Bailey B., Martin G. *ESL Models and their Application: Electronic System Level Design and Verification in Practice*. New York, Springer Publication, 2010, XXIV, 446 p. https://doi.org/10.1007/978-1-4419-0965-7
12. Pinkevich V., Platunov A., Gorbachev Y. Design of embedded and cyber-physical systems using a cross-level microarchitectural pattern of the computational process organization. *CEUR Workshop Proceedings*, 2020, vol. 2893.
13. Pinkevich V., Platunov A. Model-driven functional testing of cyber-physical systems using deterministic replay techniques. *Proc. of the 1st IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, 2018, pp. 141–146. https://doi.org/10.1109/ICPHYS.2018.8387650
14. Antonov A., Kustarev P., Bikovsky S. MLIP cores: Designing hardware generators with programmable microarchitectural mechanisms. *Proc. of the 52nd IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 9180593. https://doi.org/10.1109/ISCAS45731.2020.9180593
15. Pinkevich V.Y., Platunov A.E., Penskoi A.V. The approach to design of problem-oriented reconfigurable hardware computational units. *Proc. of the Wave Electronics and its Application in Information and Telecommunication Systems (WECONF)*, 2020, pp. 9131512. https://doi.org/10.1109/WECONF48837.2020.9131512

Авторы

Кольчурин Максим Вячеславович — аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, SC 57225126062, https://orcid.org/0000-0002-7061-9357, maxim.kolchurin@gmail.com

Пинкевич Василий Юрьевич — кандидат технических наук, инженер, ООО «ЛМТ», Санкт-Петербург, 199034, Российская Федерация, SC 56951052900, https://orcid.org/0000-0002-8635-5026, vasiliy.pinkevich@yandex.ru

Authors

Maxim V. Kolchurin — PhD Student, ITMO University, Saint Petersburg, 197101, Russian Federation, SC 57225126062, https://orcid.org/0000-0002-7061-9357, maxim.kolchurin@gmail.com

Vasiliy Yu. Pinkevich — PhD, Engineer, LMT Ltd., Saint Petersburg, 199034, Russian federation, SC 56951052900, https://orcid.org/0000-0002-8635-5026, vasiliy.pinkevich@yandex.ru

Платунов Алексей Евгеньевич — доктор технических наук, профессор, профессор, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, SC 35318291200, https://orcid.org/0000-0003-3003-3949, aeplatunov@gmail.com

Alexey E. Platunov — D. Sc., Full Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, SC 35318291200, https://orcid.org/0000-0003-3003-3949, aeplatunov@gmail.com

Статья поступила в редакцию 15.05.2022
Одобрена после рецензирования 30.05.2022
Принята к печати 12.07.2022

Received 15.05.2022
Approved after reviewing 30.05.2022
Accepted 12.07.2022