Russian Technological Journal.
2026;14(2):29–41

ISSN 2782-3210 (Print)
ISSN 2500-316X (Online)

УДК 004.2
https://doi.org/10.32362/2500-316X-2026-14-2-29-41
EDN XHLRAX

НАУЧНАЯ СТАТЬЯ

О проектировании гетерогенных вычислительных систем с аппаратным ускорением массово-параллельной потоковой обработки данных

А.С. Зуев [@], П.Н. Советов, И.Е. Тарасов

МИРЭА – Российский технологический университет, Москва, 119454 Россия
[@] Автор для переписки, e-mail: zuev_a@mirea.ru

• **Поступила:** 09.06.2025 • **Доработана:** 23.09.2025 • **Принята к опубликованию:** 05.02.2026

Резюме

Цели. Необходимость ускорения вычислений и достижения высоких показателей энергоэффективности приводит к расширению сфер использования специализированных гетерогенных вычислительных систем, имеющих в своем составе аппаратные ускорители с массовым параллелизмом вычислений. Целью настоящей работы является создание методики анализа и оценки вариантов реализации аппаратных ускорителей для задач массово-параллельной потоковой обработки данных, отражающей все направления совершенствования характеристик применяемых аппаратных ускорителей.

Методы. Использованы методы проектирования и моделирования цифровых систем.

Результаты. Предложен метод сравнительной оценки эффективности архитектуры гетерогенной вычислительной системы на основе аппаратных ускорителей с массовым параллелизмом вычислений, выполняемых независимо программируемыми узлами. Введен коэффициент ускорения вычислений, объединяющий три направления совершенствования характеристик применяемых аппаратных ускорителей – математическое обеспечение и микроархитектура, инструментарий проектирования, технологии изготовления (литография). Предложена основанная на решении оптимизационной задачи методика анализа и оценки вариантов реализации применяемых аппаратных ускорителей.

Выводы. Полученные авторами формулы расчета коэффициента ускорения вычислений и пропускной способности совокупности аппаратных ускорителей учитывают многоканальную и поблочную массово-параллельную обработку потоков данных. В отличие от известных подходов к поиску архитектурных решений, предлагаемая оценка вариантов реализации аппаратных ускорителей может быть проведена на самых ранних этапах проектирования с учетом версий алгоритма и альтернатив их реализации, влияющих на оптимизацию аппаратной архитектуры. Предложенная методика анализа и оценки вариантов реализации аппаратных ускорителей может применяться при разработке технических заданий на их изготовление, при их проектировании в соответствии с заданными требованиями, для обоснования решений относительно их конфигурации, а также при составлении заданий на научно-исследовательские и опытно-конструкторские работы с целью достижения целевых значений характеристик для конкретных задач массово-параллельной потоковой обработки данных и функциональных возможностей систем автоматизированного проектирования.

Ключевые слова: процессор, аппаратный ускоритель, сопроцессор, спецпроцессор, архитектура, компилятор

Для цитирования: Зуев А.С., Советов П.Н., Тарасов И.Е. О проектировании гетерогенных вычислительных систем с аппаратным ускорением массово-параллельной потоковой обработки данных. *Russian Technological Journal*. 2026;14(2):29–41. https://doi.org/10.32362/2500-316X-2026-14-2-29-41, https://www.elibrary.ru/XHLRAX

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

RESEARCH ARTICLE

Heterogeneous computing systems with hardware acceleration of massively parallel stream processing design

Andrey S. Zuev [@], Peter N. Sovietov, Ilya E. Tarasov

MIREA – Russian Technological University, Moscow, 119454 Russia
[@] Corresponding author, e-mail: zuev_a@mirea.ru

• **Submitted:** 09.06.2025 • **Revised:** 23.09.2025 • **Accepted:** 05.02.2026

Abstract

Objectives. The growing demand for higher computational performance and energy efficiency has motivated the increasing adoption of specialized heterogeneous computing systems incorporating hardware accelerators with massive parallelism. This paper aims to develop a methodology for the analysis and evaluation of hardware accelerator implementation strategies for large-scale parallel stream data processing which systematically captures all major directions of performance improvement.

Methods. The study employs established techniques of digital system design and modeling.

Results. A comparative evaluation method is introduced to assess the efficiency of heterogeneous computing architectures based on massively parallel hardware accelerators composed of independently programmable nodes. A computational acceleration ratio is defined which consolidates three key dimensions of accelerator improvement: algorithmic support and microarchitecture; design automation tools; and fabrication technologies (lithography). Furthermore, the study proposes an optimization-based methodology for the systematic analysis and evaluation of the alternatives for hardware accelerator implementation.

Conclusions. The expressions derived herein for calculating the computational acceleration ratio and the aggregate throughput of hardware accelerators account for both multichannel and block-based massively parallel data stream processing. In contrast to conventional architectural exploration approaches, the evaluation method proposed herein enables hardware accelerator design alternatives to be assessed at the earliest stages of the design cycle. This incorporates variations in algorithmic versions and implementation strategies which influence hardware architecture optimization. The proposed methodology for analyzing and evaluating implementation options for hardware accelerators can be used to develop technical specifications for their manufacture, design them according to specified requirements, and justify configuration decisions. It can also support research and development assignments to achieve target characteristics for certain domain-specific tasks of massively parallel stream data processing and CAD capabilities.

Keywords: processor, hardware accelerator, coprocessor, special-purpose processor, architecture, compiler

For citation: Zuev A.S., Sovietov P.N., Tarasov I.E. Heterogeneous computing systems with hardware acceleration of massively parallel stream processing design. *Russian Technological Journal.* 2026;14(2):29–41. https://doi.org/10.32362/2500-316X-2026-14-2-29-41, https://www.elibrary.ru/XHLRAX

Financial disclosure: The authors have no financial or proprietary interest in any material or method mentioned.

The authors declare no conflicts of interest.

ВВЕДЕНИЕ

Проектирование высокопроизводительных вычислительных систем в настоящее время основывается на двух важных направлениях – параллельных архитектурах и специализации отдельных вычислительных узлов. Широко распространенным примером является сочетание процессоров общего назначения и графических ускорителей, где графические ускорители представляют собой микросхемы с большим количеством специализированных вычислительных узлов, ориентированных на выполнение преобразований, характерных для построения изображений трехмерной графики. Подобные архитектурные решения с сочетанием универсальных и специализированных вычислительных подсистем применимы также для задач в сферах искусственного интеллекта, сетевой безопасности и анализа больших данных.

При разработке вычислительных систем с гетерогенной архитектурой необходимо решить комплекс взаимосвязанных задач, включающий:

- разработку программной модели и прототипов специализированных компиляторов для выбранной предметно-ориентированной задачи;
- разработку архитектуры и основных компонентов вычислительных устройств на уровне регистровых передач;
- реализацию аппаратных подсистем в выбранном технологическом базисе, в качестве которого могут выступать как программируемые логические интегральные схемы (ПЛИС), так и вновь разрабатываемые сверхбольшие интегральные схемы (СБИС).

Важной задачей, решение которой обуславливает показатели качества вычислительной системы, является комплексная оптимизация по устанавливаемым критериям. В зависимости от назначения системы оптимизация может проводиться как по абсолютным (энергопотребление, себестоимость, производительность и т.д.), так и по производным показателям (например, удельное энергопотребление или стоимость на единицу производительности). С учетом широкого диапазона характеристик, определяемых имеющимися в микроэлектронной отрасли поколениями технологических процессов, необходимо производить как взаимную оптимизацию программной модели и архитектурных решений, так и согласование схемотехнических подходов с возможностями технологического базиса. Следовательно, можно расширить подход, обозначаемый как «совместное проектирование аппаратного и программного обеспечения» (hardware & software co-design), включив в него совместную оптимизацию с технологическим уровнем проектирования компонентов устройства.

В настоящее время существует ряд общих положений, описывающих ситуацию и тенденции в области микроэлектроники. В то время как закон Мура указывает на технико-экономическую сторону процесса роста производительности вычислений, закон масштабирования размеров транзистора, сформулированный Деннардом [1], на ранних стадиях развития микроэлектроники объяснял этот рост экстенсивным процессом повышения плотности компонентов на кристалле и уменьшением задержки распространения сигнала. Эти факторы обусловили существенный рост тактовой частоты процессоров в период 1980–2000 гг., когда для массовых процессоров настольных компьютеров наблюдался переход от 4 МГц в начале 1980-х гг. до 4 ГГц у Intel Pentium 4.

При этом для многих технических характеристик цифровых микросхем наблюдается эффект насыщения по мере перехода к новым поколениям технологических процессов из производства. Сравнительные характеристики некоторых параметров поколений технологических процессов приведены в табл. 1.

Таблица 1. Сравнительные характеристики некоторых параметров поколений технологических процессов

Год	2013	2015	2017	2019	2021
Обозначение	16/14	10	7	5	3
Шаг координатной сетки (half pitch), нм	40	32	25	20	16
Ширина транзистора FinFET[1], нм	7.6	7.2	6.8	6.4	6.1
Число логических вентилей 4И на кв. мм, млн	4	6.4	10.1	16.1	25.5
Длина затвора транзистора, нм	20	17	14	12	10

[1] Fin field-effect transistor – МОП-транзистор (металл-оксид-полупроводник) с трехмерной структурой. [A FinFET (fin field-effect transistor) is a type of MOSFET (metal–oxide–semiconductor field-effect transistor) with a three-dimensional structure.]

Из приведенных сведений видно, что уменьшение размеров транзистора приостановилось, при этом основным фактором улучшения характеристик выпускаемых микросхем становится повышение плотности компонентов на единицу площади полупроводникового кристалла. Таким образом, дальнейшее улучшение транзисторов может происходить не в соответствии с законом Деннарда, а по мере перехода к новым материалам или изменения конструкции затвора (например, FinFET [2], GAAFET[2] [3], MBCFET[3] [4] и т.д.). В связи с этим эволюция вычислительных устройств претерпевала ряд изменений, отражающих особенности новых поколений технологических процессов производства интегральных микросхем.

В работе Паттерсона и Хеннесси [5] дается исторический обзор основных тенденций проектирования процессоров. В частности, отмечается, что переход к микросхемам высокой степени интеграции позволил в период 1980–1990 гг. перейти от простых процессорных ядер к конвейеризованным архитектурам типа RISC[4], причем вплоть до технологических норм 90-65 нм глубина конвейера возрастала. Прекращение действия закона Деннарда с одновременным увеличением количества компонентов на одном кристалле позволило перейти к многоядерным процессорным устройствам, что дало возможность увеличения общей производительности процессора не за счет роста его тактовой частоты, а за счет параллельной работы нескольких ядер.

В то же время, рост производительности вычислений в параллельной архитектуре ограничивается законом Амдала [6], который учитывает зависимость от данных в исполняемом алгоритме. В настоящее время ограничения закона Амдала для многих применений (в частности, настольных и серверных) не являются критичными вследствие того, что независимо выполняемые задачи не имеют зависимости от данных, и, таким образом, могут в полной мере использовать имеющиеся в наличии процессорные ядра.

Необходимость ускорения вычислений в пределах одной задачи требует применения соответствующих архитектурных решений, не сводящихся только к увеличению количества параллельно работающих вычислительных узлов. В качестве важного фактора роста производительности вычислений может рассматриваться глубокая специализация вычислительного узла, в т.ч. специализация его тракта данных и подсистемы памяти, применение системы с гетерогенной архитектурой, состоящей из комбинации процессоров общего назначения и специализированных узлов, ускоряющих вычисление определенных алгоритмов или подклассов алгоритмов для типовых, многократно решаемых предметно-ориентированных задач.

Методика, предложенная в настоящей статье, позволяет организовать проектирование специализированных вычислительных устройств для работы в составе программно-аппаратных комплексов различного назначения. Особенностью методики является комплексное рассмотрение уровней проектирования от системной модели до топологической реализации с использованием обратных связей от расположенных ниже уровней проектирования, что позволяет проводить комплексную оптимизацию характеристик вычислительного устройства по устанавливаемым критериям. Предложены математические критерии эффективности ускорителя, основанные на введенном коэффициенте ускорения вычислений по сравнению с процессором общего назначения. Также сформулирована задача оптимизации архитектуры аппаратного ускорителя в виде задачи поиска в пространстве архитектурных вариантов.

В отличие от известных аналитических моделей, таких как Roofline [7] и LogCA [8], рассматриваемые в статье формулы расчета коэффициента ускорения вычислений и пропускной способности учитывают многоканальную и поблочную массово-параллельную обработку потоков данных.

В отличие от известных подходов к поиску в пространстве архитектурных вариантов [9] предложенная оценка вариантов реализации аппаратных ускорителей может быть проведена на самых ранних этапах проектирования. Она учитывает как специфику массово-параллельной потоковой обработки данных, так и возможность выбора среди различных вариантов ускоряемого алгоритма и его реализаций, с учетом которых происходит оптимизация аппаратной архитектуры.

Гетерогенные вычислительные системы на основе специализированных аппаратных ускорителей с массовым параллелизмом вычислений

На рис. 1 представлена интерпретация авторами работы Паттерсона и Хеннесси [5] в части тенденций изменения архитектуры процессорных систем.

[2] Gate-all-around field-effect transistor – транзистор, каналы в котором окружены затворами со всех четырех сторон. [A GAAFET (gate-all-around field-effect transistor) is a type of transistor where the channels are surrounded by gates on all four sides.]

[3] Multi-bridge channel field-effect transistor – многоканальный полевой транзистор. [MBCFET is a multi-bridge channel field-effect transistor.]

[4] Reduced instruction set computer – компьютерная архитектура с сокращенным набором команд. [RISC (reduced instruction set computer) is a computer architecture that has a reduced instruction set.]

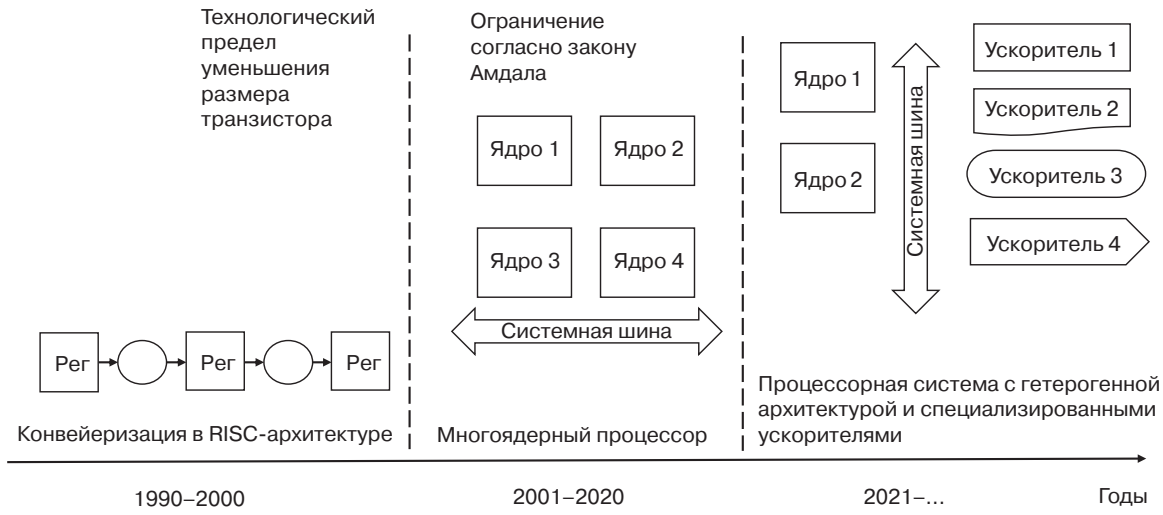

Рис. 1. Иллюстрация тенденций изменения архитектуры процессорных систем. Рег – регистр

Представленная иллюстрация предлагает вариант архитектурного решения для предложенного Паттерсоном и Хеннесси направления «предметно-ориентированных архитектур» (DSA, domain-specific architecture). Формально, специализация вычислительной системы может быть произведена множеством способов. В данной публикации авторами предложено и рассмотрено решение, предусматривающее сохранение процессора общего назначения (в т.ч. в многоядерном исполнении) и добавление специализированных ускорителей, работающих в составе системы в качестве независимых параллельно функционирующих с ним устройств.

Для конкретизации роли процессоров общего назначения и ускорителей можно рассмотреть диаграмму, представленную на рис. 2, где оси представляют сложность одной итерации вычислений в циклическом алгоритме и количество таких итераций, требуемых для решения задачи.

На рис. 2 можно видеть четыре основные зоны, соответствующие комбинации простых и сложных

Рис. 2. Диаграмма соотношения сложности итераций и их количества с предпочтительными архитектурами вычислительных устройств

итераций (имеющих, в первую очередь, зависимости по данным, затрудняющие распараллеливание вычислений), а также их малого и большого количества. Простой случай в виде несложного алгоритма с небольшим количеством итераций (частным случаем является линейный алгоритм с единственной итерацией) не представляет сложности при его реализации на базе процессора общего назначения, поэтому создание специализированного ускорителя для такого случая нецелесообразно. При увеличении сложности алгоритма реализующее его вычислительное устройство должно обладать развитым набором функциональных узлов и достаточными ресурсами памяти для размещения относительно длинной последовательности команд. Также немаловажно, что сложные алгоритмы часто могут требовать коррекции в ходе эксплуатации системы (например, обновление протоколов работы устройства), поэтому выделение ключевой вычислительной подсистемы для ее аппаратной реализации в виде ускорителя в подобном случае является сложной задачей. При этом небольшое количество итераций такого алгоритма не создает существенной вычислительной нагрузки на процессор общего назначения.

Важным вопросом является интеграция ускорителя в состав вычислительной системы. Повышение степени интеграции цифровых микросхем привело к появлению эффекта «стены интерфейсов» [10], который проявляется в виде опережающего роста производительности вычислений по сравнению с производительностью внешних интерфейсов (подсистем ввода-вывода). На практике это означает, что размещение в ускорителе большого количества простых вычислительных узлов, требующих интенсивного обмена данными с процессором общего назначения, приведет к перегрузке интерфейсов обмена данными

и снижению итоговой производительности вычислений из-за невозможности обеспечить требуемый поток входных данных для ускорителей. В качестве примера можно привести эволюцию характеристик ПЛИС с распространенными архитектурами – FPGA[5], APSoC[6] и ACAP[7], которые в настоящее время являются характерными представителями высокоинтегрированных цифровых систем, совмещающих подсистемы различного назначения. В табл. 2 приведены сравнительные характеристики производительности вычислительной и интерфейсной подсистем некоторых семейств ПЛИС производства компании AMD/Xilinx (США).

Таблица 2. Сравнительные характеристики производительности вычислительной и интерфейсной подсистем некоторых семейств ПЛИС AMD

Серия ПЛИС	Серия 7	UltraScale+	Versal
Норма технологического процесса, нм	28	16	7
Логических ячеек, тыс.	2000	8900	7350
Блоков цифровой обработки сигналов	3600	12288	14352
Суммарная производительность вычислений, трлн оп./с с данными int8	15	38.3	100
Суммарная производительность передачи данных, Тбит/с	2	4.1	17.6
Соотношение производительности вычислений и передачи данных	60	74	45

Из табл. 2 можно видеть, что между производительностью вычислений и передачи данных в целом сохраняется соотношение порядка 50–70. Это означает, что при эффективной загрузке вычислительных мощностей с переданными данными производится не менее 50–70 операций обработки. Данный диапазон значений является условным и не учитывает практические аспекты реализации операций, однако можно отметить, что чрезмерно простой ускоритель снизит эффективность системы как таковой, поскольку каждая операция потребует интенсивного обмена данными между ускорителем и другими компонентами системы. Негативный сценарий использования ускорителя вычислений с ограничением по пропускной способности интерфейса показан на рис. 3.

Таким образом, на архитектурном уровне следует предпринимать усилия по обеспечению соответствующего уровня производительности вычислений, гарантирующего эффективное использование ресурсов ускорителя при заданной пропускной способности его интерфейсов. Варианты таких сценариев показаны на рис. 4.

Показанные варианты предусматривают, что ускоритель выполняет множество операций с передаваемыми данными. Вариант 1 – при реализации относительно сложного автономного алгоритма, выполняющего итерации вычислений без необходимости передачи постоянного потока данных. Вариант 2 – при параллельной обработке данных, в которой большое количество операций реализуется последовательно соединенными стадиями конвейерного вычислителя. Авторами рассматривается показанный на рис. 4 вариант 2 – гетерогенная вычислительная система с аппаратным ускорением массово-параллельной потоковой обработки данных.

Задачи и алгоритмы массово-параллельной потоковой обработки данных

Задачи потоковой обработки данных возникают в различных областях, к которым, в частности относятся: цифровая обработка сигналов, сжатие данных, криптография, аналитика реального времени и анализ сетевого трафика.

Потоковая обработка предполагает поэлементную обработку потенциально бесконечного потока данных $x_1, x_2, \ldots, x_t, \ldots$, где x_t – элемент, доступный для обработки в момент времени t. Элементы являются однотипными и могут представлять собой как отдельные биты или байты, так и сложноструктурированные сущности.

Для алгоритмов потоковой обработки характерно их выполнение за один проход, без возможности перейти к уже просмотренным или еще не обработанным элементам. Обработка осуществляется в условиях ограниченной локальной памяти с пространственной сложностью $O(1)$, не зависящей от количества элементов потока.

К основным характеристикам систем потоковой обработки относится пропускная способность $R = N/t_N$, где N – число элементов, а t_N – общее время их обработки. При этом $R \le R_{max}$, где R_{max} представляет собой максимально допустимую пропускную

[5] Field-programmable gate array – программируемая пользователем вентильная матрица. [A field-programmable gate array is a user-programmable gate array.]

[6] All programmable system-on-chip – тип интегральной схемы, программируемая система на чипе. [All programmable system-on-chip is a type of programmable integrated circuit.]

[7] Adaptive compute acceleration platform.

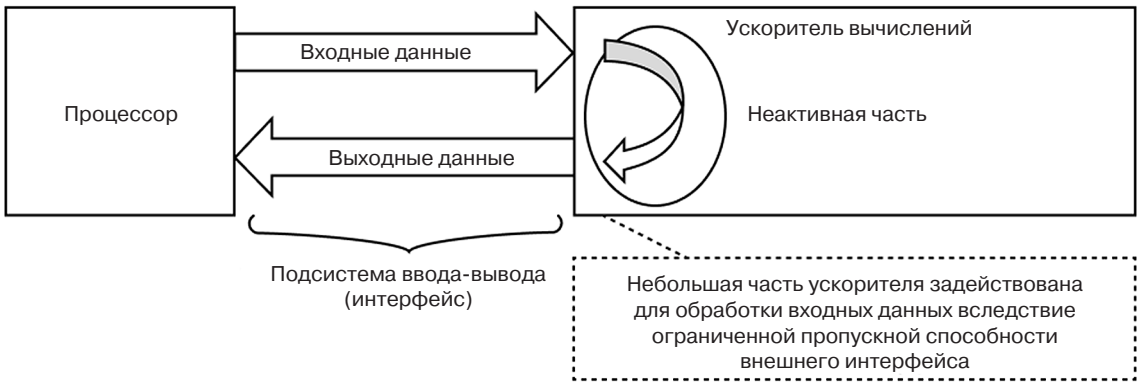

Рис. 3. Негативный сценарий использования ускорителя вычислений с ограничением пропускной способности подсистемы ввода-вывода (интерфейса)

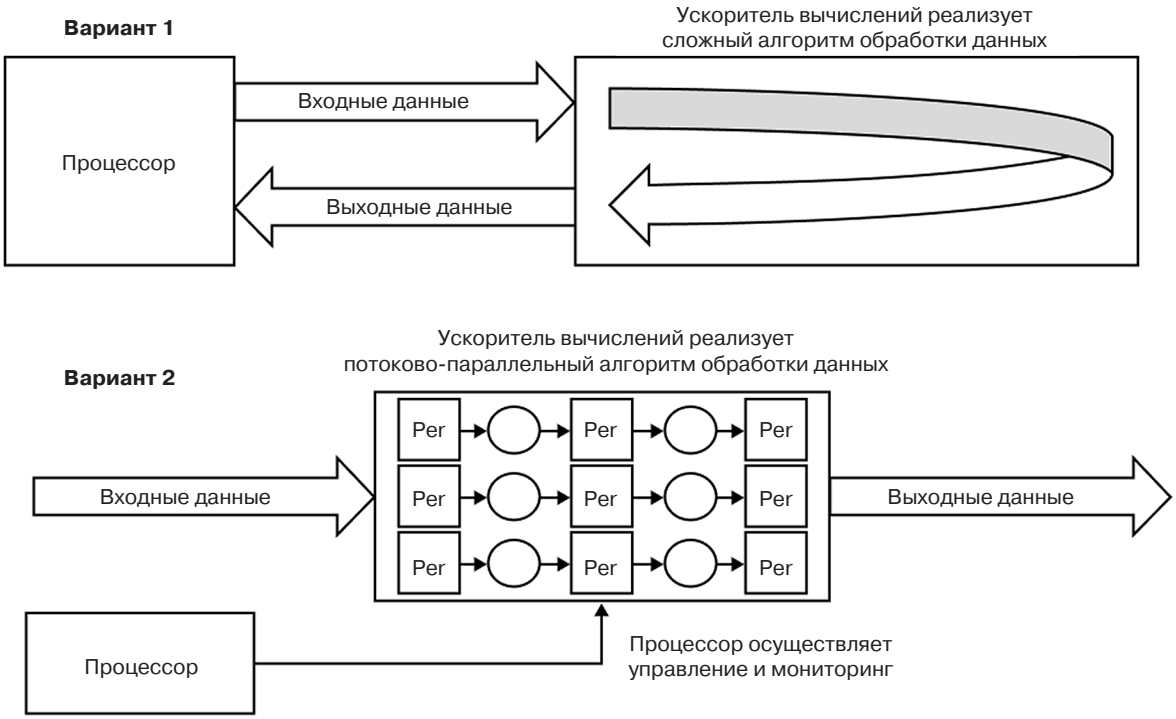

Рис. 4. Варианты сценариев использования ускорителя вычислений, эффективно использующего ограниченную пропускную способность интерфейсов (используется конвейеризация вычислений)

способность, ограниченную, в частности, возможностями подсистемы ввода-вывода. В условиях работы в реальном времени задержка на обработку одного элемента должна быть также ограничена: $L \leq L_{\max}$.

Аппаратное ускорение позволяет использовать в потоковой обработке данных параллелизм на различных уровнях. В частности, параллелизм уровня команд определяется, в первую очередь, интенсивностью вычислений, т.е. преобладанием вычислительных операций над операциями обращения к памяти при обработке элемента потока. Параллелизм задач (параллелизм каналов) возможен при условии обработки нескольких каналов одновременно.

Во многих случаях потоковый алгоритм необходимо модифицировать с учетом конкретной архитектуры аппаратного ускорителя. Например, если размер таблиц, используемых в реализуемом алгоритме, превышает размер локальной памяти аппаратного ускорителя, то таблицы потребуется заменить дополнительными вычислениями.

В литературе описаны аппаратные ускорители потоковой обработки данных для различных областей и предметно-ориентированных задач. Существуют, в частности, аппаратные ускорители для быстрой проверки принадлежности элемента множеству (фильтр Блума) [11] и задачи оценки числа различных элементов

в мультимножестве (HyperLogLog) [12]. Для ускорения работы различных алгоритмов на строках также спроектированы аппаратные ускорители: вычисление редакционного расстояния [13], поиск по регулярным выражениям [14], разбор формата JSON[8] [15]. Вопросы аппаратного ускорения сжатия и декомпрессии данных описаны в [16]. В [17] описан аппаратный ускоритель, объединяющий декомпрессию данных с разбором формата JSON.

Важной областью для применения аппаратных ускорителей являются различные алгоритмы баз данных. В [18] описано аппаратное ускорение операций реляционной базы данных: выборки, проекции, соединения, сортировки и различных агрегатных функций. Аппаратное ускорение операции сопоставления с образцом для графовой базы данных предложено в [19]. Перспективным для аппаратного ускорения является направление потоковой аналитики. В частности, в [20] описаны основные алгоритмы из этой области, допускающие аппаратную реализацию. Существуют также подходы к потоковой обработке для ряда алгоритмов машинного обучения. На этой основе реализованы аппаратные ускорители для градиентного бустинга [21], а также для задач кластеризации [22].

Сравнительная оценка эффективности вычислительных систем потоковой обработки данных: с процессором общего назначения и с применением аппаратного ускорения

Количественная оценка эффективности ускорителя может быть проведена на основе сравнения T_{soft} – времени выполнения задачи с использованием только процессора общего назначения и T_{hard} – времени выполнения задачи с использованием ускорителя, т.е.:

$$K = \frac{T_{soft}}{T_{hard}}.$$

При оценке времени выполнения программы удобно воспользоваться формулой, которая подробно прокомментирована в [23]:

$$\text{time} = \frac{\text{instructions}}{\text{program}} \cdot \frac{\text{cycles}}{\text{instruction}} \cdot \frac{T_{period}}{\text{cycle}}.$$

В этой формуле отдельные множители, представленные в виде дробей, соответствуют характеристикам компилятора, микроархитектуры и технологического процесса:

1. Количество инструкций (instructions) на одну итерацию программы определяется способностью компилятора порождать оптимизированный машинный код по показателю производительности.
2. Количество тактов (cycles) на инструкцию определяется микроархитектурой процессора.
3. Время выполнения одного такта (период тактового сигнала) – T_{period} в основном определяется технологическим процессом изготовления процессора.

Таким образом, общее время выполнения программы может быть улучшено различными способами, сочетания которых могут быть использованы в задачах проектирования специализированных ускорителей.

Для ускорителя, подключаемого к процессору с помощью одного из интерфейсов, необходимо также учитывать время T_{conf} его конфигурирования – передачи входных параметров задачи и приема результатов ее решения.

При условии, что одна и та же задача требует выполнения b итераций, отношение времени работы процессора общего назначения к времени работы ускорителя принимает вид:

$$K = \frac{bT_{soft}}{T_{conf} + bT_{hard}}.$$

В данном виде формула не отражает наличия в ускорителе нескольких вычислительных узлов, которые могли бы работать параллельно. Под узлами потоковой обработки данных будем понимать количество одновременно выполняемых в ускорителе экземпляров алгоритма обработки данных. Например, в случае обработки потока данных с целью поиска по регулярным выражениям количество узлов потоковой обработки данных соответствует количеству регулярных выражений, по которым анализ каждого блока данных выполняется одновременно и параллельно.

Пусть t_{hard} – время выполнения алгоритма с использованием единственного узла аппаратного ускорителя. Если обозначить количество параллельно работающих узлов, задействованных в решении данной задачи, как p, то формула принимает вид:

$$K = \frac{bT_{soft}}{T_{conf} + bt_{hard}/p}.$$

Значением T_{conf} можно пренебречь, т.к. вследствие потоковой обработки данных конфигурирование ускорителя производится однократно для продолжительных периодов его работы, в течение которых выполняется обработка большого (потенциально – бесконечного) количества однотипных блоков данных, и коэффициент ускорения вычислений стремится к:

[8] JavaScript object notation – текстовый формат обмена данными, основанный на синтаксисе JavaScript. [JavaScript object notation is a text-based data exchange format based on JavaScript syntax.]

$$K = \frac{pT_{soft}}{t_{hard}} = \frac{pn_{soft}f_{hard}}{n_{hard}f_{soft}}, \quad (1)$$

где n_{hard}, n_{soft} – количество тактов, требуемых ускорителю и процессору соответственно для решения задачи, а f_{hard}, f_{soft} – соответствующие тактовые частоты.

Формула (1) показывает, что аппаратное ускорение работы вычислительной системы в целом основано на двух факторах:

- увеличение количества параллельно работающих узлов;
- улучшение эффективности одного узла путем повышения его тактовой частоты и уменьшения количества тактов, требуемых для реализации задачи.

Методика разработки специализированных процессорных ядер была ранее рассмотрена в [24]. В ситуации, когда разработка специализированной электронной компонентной базы (ЭКБ) затруднена отсутствием доступа к современным технологическим процессам и необходимостью быстрой адаптации вновь разрабатываемых схем к технологическому базису, достижение высокой тактовой частоты ускорителя объективно затруднено. Поэтому соотношение частот аппаратного ускорителя и серийно выпускаемых процессоров общего назначения может быть невысоким (по сути, меньше 1). В связи с этим следует обращать особенное внимание на снижение количества тактов, требуемых аппаратному ускорителю для обработки одного блока данных, т.е. уменьшение параметра n_{hard} при одновременном сохранении достаточно высокой (в общем случае, без учета фактора энергопотребления – максимально возможной) тактовой частоты f_{hard}. Это может быть достигнуто с использованием специализированных параллельных алгоритмов и фаз оптимизирующего компилятора, учитывающих аппаратные особенности ускорителя.

Таким образом, предложенный авторами коэффициент ускорения вычислений (1) объединяет три направления совершенствования характеристик аппаратных ускорителей гетерогенных систем массово-параллельной потоковой обработки данных: математическое обеспечение и микроархитектура ускорителя (p и n_{hard}), инструментарий проектирования (система автоматизированного проектирования, САПР) и технологии изготовления (литография) ЭКБ (p и f_{hard}).

Для ускорителя с p узлами параллельной потоковой обработки данных, с учетом (1), время $T_{hard}(n)$ обработки n блоков данных может быть определено как

$$T_{hard}(n) = T_{conf} + n \cdot \max\left(T_{block}, \frac{T_{soft}}{K}\right), \quad (2)$$

где T_{block} – время приема-передачи очередного блока данных, которое определяется возможностями подсистемы ввода-вывода и поэтому не может быть сокращено; p – количество экземпляров алгоритма, параллельно обрабатывающих очередной блок данных.

В случае возможности реализации на одной микросхеме нескольких рассмотренных выше параллельно функционирующих ускорителей можно говорить о параллельной обработке C потоков блоков данных и считать итоговым вариантом ускорителя их данную совокупность. Тогда пропускная способность R_{hard} совокупности из C параллельно работающих ускорителей (с характеристикой $T_{hard}(n)$ по формуле (2)) имеет вид:

$$R_{hard} = \frac{Cn}{T_{hard}(n)} = \frac{C}{\frac{T_{conf}}{n} + \max\left(T_{block}, \frac{T_{soft}}{K}\right)} \approx \frac{C}{\max\left(T_{block}, \frac{T_{soft}}{K}\right)}.$$

Величина C ограничена числом физически доступных интерфейсов в реализуемой подсистеме ввода-вывода, а также спецификацией решаемой прикладной задачи. Таким образом, в процессе проектирования аппаратного ускорителя, помимо принятия решений о технологических процессах его изготовления и технических характеристиках, требуется одновременная разработка инструментальных средств, в частности, специализированного компилятора, который может использоваться для совместной оптимизации программно-аппаратного обеспечения.

Анализ и оценка вариантов реализации аппаратных ускорителей гетерогенных вычислительных систем массово-параллельной потоковой обработки данных

Изложенный в предыдущем разделе материал содержит описание метода оценки эффективности применения для массово-параллельной потоковой обработки данных процессора общего назначения относительно специализированного ускорителя, обладающего следующими характеристиками:

T_{block} – время приема-передачи блока данных, определяемое подсистемой ввода-вывода (вариантами реализации сопряжения ускорителя с вычислительной системой);

f_{hard} – тактовая частота, зависящая от доступных вариантов приобретения или изготовления ускорителя, фактически определяется возможностями инструментария его проектирования (САПР)

и технологий изготовления (ПЛИС или заказная микросхема);

n_{hard} – количество тактов, требуемых ускорителю (узлу потоковой обработки данных) для решения задачи, зависит от состава применяемых специализированных параллельных алгоритмов и фаз оптимизирующего компилятора, учитывающих аппаратные особенности ускорителя, фактически определяется вариантами математического обеспечения решения задачи ускорителем и его микроархитектурой;

p – количество параллельно работающих в составе ускорителя узлов потоковой обработки данных, задействованных в решении задачи.

Заметим, что доступные варианты значений f_{hard} определяются характеристиками доступных для применения ПЛИС и СБИС в случае приобретения готовой ЭКБ, а также вариантами их изготовления, т.е. доступными в производстве ЭКБ технологическими процессами и библиотеками стандартных ячеек.

При проектировании ускорителя к перечисленным выше характеристикам целесообразно добавить также следующие параметры:

T_{task} – время обработки одного блока данных ускорителем, определяемое требованиями технического задания (ТЗ);

$C_{\max}$ – максимально доступное число каналов подсистемы ввода-вывода;

C ($C \le C_{\max}$) – размещаемое на кристалле количество экземпляров ускорителя;

S ($S \le S_{\max}$) – площадь кристалла, ограниченная в его конкретном исполнении.

Конкретные значения некоторых из перечисленных выше параметров зависят от особенностей реализации решения задачи на ускорителе – от конкретного варианта соответствующего алгоритма обработки данных. Пусть a_i, $i = \overline{1,n}$ – множество альтернативных алгоритмов, т.е. вариантов реализации решения задачи на ускорителе, каждый из которых характеризуется следующим набором параметров:

n_{hard}^i – количество тактов, требуемых одному вычислительному узлу ускорителя для решения задачи;

P_i – максимально возможное количество параллельно работающих в составе ускорителя экземпляров алгоритма;

local_i – площадь кристалла, занимаемая объемом локальной памяти m_i ускорителя, требуемой для выполнения алгоритма a_i;

alu_i – площадь кристалла, занимаемая одним узлом потоковой обработки данных;

glob_i – площадь кристалла, выделяемая для хранения данных, используемых всеми экземплярами алгоритма a_i (может быть константой, например для случая регулярных выражений, а также зависеть от пропускной способности подсистемы ввода-вывода, требований к загрузке процессора общего назначения и т.п.).

Тогда $S_i = (\text{local}_i + \text{alu}_i)$ – площадь кристалла, выделяемая для одного узла потоковой обработки данных ускорителя, т.е. для функционирования одного экземпляра алгоритма a_i. Для ускорителя с p_i узлами потоковой обработки данных, реализующими отдельные экземпляры алгоритма a_i, требуемая площадь кристалла может быть рассчитана по формуле:

$$U_i = S_i p_i = (\text{local}_i + \text{alu}_i)p_i,\ p_i \le P_i.$$

При размещении на кристалле C_i экземпляров ускорителя, каждый из которых реализует p_i экземпляров алгоритма a_i требуемая площадь S_i кристалла может быть определена следующим образом:

$$S_i = \text{glob}_i + U_i C_i = \text{glob}_i + S_i p_i C_i = \\ = \text{glob}_i + (\text{local}_i + \text{alu}_i)p_i C_i,\ p_i \le P_i,\ S_i \le S.$$

Формула (2) позволяет не только определить время $T_{\text{hard}}(n)$ обработки n блоков данных ускорителем с p узлами, но и обеспечивает возможность анализа и оценки вариантов его реализации, исходя из имеющихся (доступных) технологических возможностей. Определим ограничение на доступные значения характеристики быстродействия ускорителя – время T_{fix} обработки им одного блока данных:

$$T_{\text{fix}} = \min(T_{\text{block}}, T_{\text{task}}).$$

Время T_{fix}, определяемое возможностями подсистемы ввода-вывода или ТЗ на разработку ускорителя, можно считать однозначно детерминированным и задающим естественные ограничения на целесообразные комбинации его характеристик:

$$\frac{T_{\text{soft}}}{K} \approx T_{\text{fix}}. \qquad (3)$$

Варианты реализации ускорителя, соответствующие условию (3) со знаком «<», являются нецелесообразными, т.к. его избыточная эффективность будет ограничена возможностями подсистемы ввода-вывода. Варианты реализации ускорителя, соответствующие условие (3) со знаком «>», не обеспечивают максимум производительности.

В качестве критерия оптимальности варианта реализации ускорителя с учетом соответствующего ТЗ варианта исполнения подсистемы ввода-вывода и исходя из имеющихся технологических возможностей может использоваться выражение

$$\left| \frac{T_{\text{soft}}}{K} - T_{\text{fix}} \right| \to \min,$$

которое с учетом (1) и возможностей реализации на ускорителе множества альтернативных алгоритмов $a_i, i = \overline{1,n}$ принимает следующий вид:

$$\left| \frac{n^i_{\text{hard}}}{p_i f_{\text{hard}}} - T_{\text{fix}} \right| \to \min. \quad (4)$$

С учетом того, что параметр f_{hard} может принимать различные значения (описываться вектором значений) может быть сформулирована следующая оптимизационная задача проектирования ускорителя: для каждого варианта алгоритма решения задачи a_i, $i = \overline{1,n}$ определить такие комбинации значений p_i, C_i и f^i_{hard}, при которых:

$$S_i = \text{glob}_i + (\text{local}_i + \text{alu}_i) p_i C_i,$$

$$p_i \le P_i,$$

$$S_i \le S,$$

$$\frac{n^i_{\text{hard}}}{p_i f^i_{\text{hard}}} \le T_{\text{fix}},$$

$$T_{\text{fix}} - \frac{n^i_{\text{hard}}}{p_i f^i_{\text{hard}}} \to \min. \quad (5)$$

В результате для алгоритмов a_i, $i = \overline{1,n}$ могут быть определены соответствующие множества $|\text{A}_i| = r_i$ комбинаций значений рассматриваемых параметров $(p^k_i, C^k_i, f^{i,k}_{\text{hard}})$, $k = \overline{1,r_i}$, являющиеся альтернативными вариантами изготовления ускорителя. Выбор проектного(ых) варианта(ов) ускорителя может быть выполнен по результатам их имитационного моделирования средствами САПР, а также с учетом дополнительных критериев, таких как энергопотребление, себестоимость, вероятность выхода из строя и т.п.

Решение представленной выше задачи определения характеристик ускорителя в соответствии с параметрами реализуемого алгоритма целесообразно автоматизировать. Этому направлению будут посвящены дальнейшие исследования авторов, однако ниже представлен общий подход к решению данной задачи.

Для перечисленных выше параметров p_i, C_i и f^i_{hard} каждого из a_i, $i = \overline{1,n}$ альтернативных алгоритмов могут быть составлены универсальные множества вариантов значений P, C, F, которые вместе с множеством A ($a_i \in \text{A}$, $i = \overline{1,n}$) формируют четырехмерный массив W исходных данных рассматриваемой задачи. Элементы соответствующего массива будут содержать значение «0» в случае несовместности конкретного варианта значений четырех рассматриваемых параметров или значение выражения (5) – в случае допустимости их совместной реализации и использования для его вычислений параметров n^i_{hard}, local_i, alu_i, glob_i соответствующего алгоритма a_i, $i = \overline{1,n}$.

Заметим, что множества P, C, F в общем случае детерминированы, а количество вариантов алгоритмов решения конкретной задачи, как правило невелико. Следовательно, массив W универсален в контексте множеств P, C, F, а в контексте конкретной задачи обработки данных, как правило, не обладает большой размерностью. Поэтому задачи расчета и перебора значений его элементов, позволяющие анализировать варианты реализации гетерогенных вычислительных систем с аппаратным ускорением массово-параллельной потоковой обработки данных, не будут характеризоваться высокой вычислительной сложностью.

Изложенные выше постановка задачи и порядок определения соответствующих ей исходных данных позволяют сформировать методику анализа и оценки вариантов реализации аппаратных ускорителей гетерогенных вычислительных систем массово-параллельной потоковой обработки данных. Предложенная методика, по результатам ее дальнейшего развития, может претендовать на универсальность – применяться при разработке ТЗ на изготовление аппаратных укорителей, при их проектировании в соответствии с заданными требованиями, для обоснования принимаемых решений относительно конфигурации аппаратного ускорителя, а также при составлении заданий на научно-исследовательские и опытно-конструкторские работы с целью достижения целевых значений f_{hard}, n_{hard}, $C_{\max}$ и $S_{\max}$ для конкретных предметно-ориентированных задач массово-параллельной обработки данных и функциональных возможностей САПР.

Развитие представленной методики будет изложено в следующих работах авторов, в т.ч. в контексте применения к работе с массивом W технологий интерактивной аналитической обработки данных (OLAP, online analytical processing).

Если рассматривать гетерогенные системы массово-параллельной обработки данных как отдельное перспективное направление развития вычислительной техники, то предложенная методика, основываясь на требованиях к параметрам решения наиболее востребованных предметно-ориентированных задач, может найти применение как инструмент научно-технической политики, позволяющий определять и обосновывать потребности в достижении целевых доступных значений:

- характеристик системных шин и интерфейсов сопряжения ускорителей с вычислительными системами (T_{block}),
- схемотехнических решений и технологий литографии (f_{hard}),
- технологий корпусирования интегральных схем (p),
- алгоритмов решения задач и математического обеспечения компиляторов (n_{hard}),
- систем высокоуровневого проектирования и программного моделирования.

ЗАКЛЮЧЕНИЕ

В целях повышения эффективности решения и расширения круга задач, решаемых за счет программирования массово-параллельных ускорителей, авторами предложены архитектура и методика проектирования гетерогенных вычислительных систем, использующих специализированные аппаратные ускорители с массовым параллелизмом вычислений и независимо программируемыми специализированными процессорными ядрами. Такое решение дополняет вычислительные системы на базе процессоров общего назначения и архитектурно отличается от устройств класса «графический процессор» вследствие возможности независимого программирования отдельных узлов. Показано, что повышение производительности достигается путем увеличения количества параллельно работающих узлов и их специализации, которую предлагается выполнять на основе анализа графа вычислений, формируемого специально разрабатываемым компилятором, ориентированным на подкласс задач целевой предметной области. Предложенный авторами коэффициент ускорения вычислений обосновывает целесообразность выполнения в процессе разработки ускорителей комплексной оптимизации с учетом трех уровней представления проекта: программной модели, схемотехнического описания и топологического представления в выбранном технологическом базисе. Определить технические характеристики требующегося ускорителя позволяет предложенная методика анализа и оценки вариантов реализации аппаратных ускорителей гетерогенных вычислительных систем массово-параллельной потоковой обработки данных. Полученная в результате методика проектирования вычислительных систем данного класса призвана обеспечить регулируемый процесс оптимизации характеристик по требуемым критериям.

Вклад авторов

Изложенные в статье исследования и результаты получены авторами совместно по итогам организации работы научной школы «Специализированные гетерогенные вычислительные системы» Института информационных технологий ФГБОУ ВО «МИРЭА – Российский технологический университет».

Основное содержание раздела «Анализ и оценка вариантов реализации аппаратных ускорителей гетерогенных вычислительных систем массово-параллельной потоковой обработки данных», в т.ч. задача оптимизации архитектуры аппаратного ускорителя в виде задачи поиска в пространстве архитектурных вариантов и методика анализа и оценки вариантов реализации аппаратных ускорителей, составлено А.С. Зуевым.

Authors' contributions

The research and results presented in the article were obtained jointly by the authors as an outcome of the scientific school "Specialized Heterogeneous Computing Systems" organized by the Institute of Information Technologies at the MIREA – Russian Technological University.

The main content of the section "Analysis and evaluation of implementation options for hardware accelerators in heterogeneous computing systems for massively parallel stream data processing," including the task of optimizing the hardware accelerator architecture formulation as a search problem in the space of architectural options, as well as the methodology for analyzing and evaluating implementation options of hardware accelerators, was developed by A.S. Zuev.

СПИСОК ЛИТЕРАТУРЫ / REFERENCES

1. Dennard R.H., Gaensslen F.H., Yu H.-N., Rideout V.L., Bassous E., LeBlanc A.R. Design of ion-implanted MOSFET's with very small physical dimensions. *IEEE Journal of Solid-State Circuits*. 1974;9(5):256–268. https://doi.org/10.1109/JSSC.1974.1050511
2. Jain P.U., Tomar V.K. FinFET Technology: As A Promising Alternatives for Conventional MOSFET Technology. In: *2020 International Conference on Emerging Smart Computing and Informatics* (*ESCI*). 2020. P. 43–47. https://doi.org/10.1109/ESCI48226.2020.9167646
3. Yakimets D., Eneman G., Schuddinck P., et al. Vertical GAAFETs for the Ultimate CMOS Scaling. *IEEE Transactions on Electron Devices*. 2015;62(5):1433–1439. https://doi.org/10.1109/TED.2015.2414924
4. Lee S.-Y., Kim S.-M., Yoon E.-J., et al. A Novel Multibridge-Channel MOSFET (MBCFET): Fabrication Technologies and Characteristics. *IEEE Transactions on Nanotechnology*. 2003;2(4):253–257. https://doi.org/10.1109/TNANO.2003.820777
5. Hennessy J.L., Patterson D.A. *Computer Architecture: A Quantitative Approach* (*The Morgan Kaufmann Series in Computer Architecture and Design*). 6th ed. 2017, 936 p.
6. Annaratone M. MPPs, Amdahl's Law, and Comparing Computers. In: *Proceedings of The Fourth Symposium on the Frontiers of Massively Parallel Computation.* 1992. P. 465–470. https://doi.org/10.1109/FMPC.1992.234879
7. Verhelst M., Benini L., Verma N. How to keep pushing ML accelerator performance? Know your rooflines! *IEEE Journal of Solid-State Circuits*. 2025;6(60):1888–1905. https://doi.org/10.1109/JSSC.2025.3553765
8. Altaf M.S.B., Wood D.A. LogCA: A high-level performance model for hardware accelerators. *ACM SIGARCH Computer Architecture News*. 2017;45(2):375–388. https://doi.org/10.1145/3079856.3080216
9. Molina R.S., Gil-Costa V., Crespo M.L., et al. High-level synthesis hardware design for FPGA-based accelerators: Models, methodologies, and frameworks. *IEEE Access*. 2022;10:90429–90455. https://doi.org/10.1109/ACCESS.2022.3201107

10. A New Golden Age for Computer Architecture: Domain-Specific Hardware/Software Co-Design, Enhanced Security, Open Instruction Sets, and Agile Chip Development. In: *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture* (*ISCA*). 2018. P. 27–29. https://doi.org/10.1109/ISCA.2018.00011
11. Liu K., Lu A., Fang Z. BitBlender: Scalable Bloom Filter Acceleration on FPGAs with Dynamic Scheduling. In: *34th International Conference on Field-Programmable Logic and Applications* (*FPL*). 2024. P. 325–331. https://doi.org/10.1109/FPL64840.2024.00052
12. Kulkarni A., Chiosa M., Preuber T.B., et al. HyperLogLog Sketch Acceleration on FPGA. In: *30th International Conference on Field-Programmable Logic and Applications* (*FPL*). 2020. P. 47–56. https://doi.org/10.1109/FPL50879.2020.00019
13. Marchisio A., Teodonio F., Rizzi A., et al. ISMatch: A Real-Time Hardware Accelerator for Inexact String Matching of DNA Sequences on FPGA. *Microprocess. Microsyst*. 2023;97:104763. https://doi.org/10.1016/j.micpro.2023.104763
14. Zhang C., Tang X., Peng Y. Enhancing Regular Expression Processing through Field-Programmable Gate Array-Based Multi-Character Non-Deterministic Finite Automata. *Electronics.* 2024;13(9):1635. https://doi.org/10.3390/electronics13091635
15. Dann J., Wagner R., Ritter D., et al. PipeJSON: Parsing JSON at Line Speed on FPGAs. In: *Proceedings of the 18th International Workshop on Data Management on New Hardware*. 2022;Article 3:1–7. https://doi.org/10.1145/3533737.3535094
16. Karandikar S., Udipi A.N., Choi J., et al. CDPU: Co-Designing Compression and Decompression Processing Units for Hyperscale Systems. In: *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 2023;Article 39:1–17. https://doi.org/10.1145/3579371.3589074
17. Hahn T., Wildermann S., Teich J. JSON-CooP: A JSON Decompression/Parsing Co-Design for FPGAs. In: *34th International Conference on Field-Programmable Logic and Applications* (*FPL*). 2024. P. 11–18. https://doi.org/10.1109/FPL64840.2024.00012
18. Fang J., Mulder Y., Hidders J., et al. In-Memory Database Acceleration on FPGAs: A Survey. *The VLDB Journal.* 2020;29(10):33–59. https://doi.org/10.1007/s00778-019-00581-w
19. Dann J., Götz T., Ritter D., et al. GraphMatch: Subgraph Query Processing on FPGAs. *arXiv.* arXiv:2402.17559. 2024.
20. Kejariwal A., Kulkarni S., Ramasamy K. Real Time Analytics: Algorithms and Systems. *arXiv.* arXiv:1708.02621. 2017. https://doi.org/10.48550/arXiv.1708.02621
21. Alcolea A., Resano J. FPGA Accelerator for Gradient Boosting Decision Trees. *Electronics*. 2021;10(3):314. https://doi.org/10.3390/electronics10030314
22. Graf J.R., Perera D.G. Optimizing Density-Based Ant Colony Stream Clustering Using FPGA-Based Hardware Accelerator. In: *2023 IEEE International Symposium on Circuits and Systems* (*ISCAS*). 2023. https://doi.org/10.1109/ISCAS46773.2023.10181665
23. Shen J.P., Lipasti M.H. *Modern Processor Design: Fundamentals of Superscalar Processors.* Waveland Press; 2013, 658 p.
24. Тарасов И.Е., Советов П.Н., Люлява Д.В., Мирзоян Д.И. Методика проектирования специализированных вычислительных систем на основе совместной оптимизации аппаратного и программного обеспечения. *Russian Technological Journal*. 2024;12(3):37–45. https://doi.org/10.32362/2500-316X-2024-12-3-37-45
[Tarasov I.E., Sovietov P.N., Lulyava D.V., Mirzoyan D.I. Method for designing specialized computing systems based on hardware and software co-optimization. *Russian Technological Journal*. 2024;12(3):37–45. https://doi.org/10.32362/2500-316X-2024-12-3-37-45]

Об авторах

Зуев Андрей Сергеевич, к.т.н., доцент, заведующий кафедрой квантовых информационных технологий, практической и прикладной информатики, Институт информационных технологий, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: zuev_a@mirea.ru. Scopus Author ID 23977152300, SPIN-код РИНЦ 6737-5778, https://orcid.org/0000-0002-1797-7585

Советов Петр Николаевич, к.т.н., доцент, кафедра корпоративных информационных систем, Институт информационных технологий, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: peter.sovietov@gmail.com. Scopus Author ID 57221375427, SPIN-код РИНЦ 9999-1460, http://orcid.org/0000-0002-1039-2429

Тарасов Илья Евгеньевич, д.т.н., доцент, профессор кафедры корпоративных информационных систем, Институт информационных технологий, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: tarasov_i@mirea.ru. Scopus Author ID 57213354150, SPIN-код РИНЦ 4628-7514, http://orcid.org/0000-0001-6456-4794

About the Authors

Andrey S. Zuev, Cand. Sci. (Eng.), Associate Professor, Head of the Department of Quantum Information Technologies, Practical and Applied Informatics, Institute of Information Technologies, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: zuev_a@mirea.ru. Scopus Author ID 23977152300, RSCI SPIN-code 6737-5778, https://orcid.org/0000-0002-1797-7585

Peter N. Sovietov, Cand. Sci. (Eng.), Associated Professor, Department of Corporate Information Systems, Institute of Information Technologies, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: peter.sovietov@gmail.com. Scopus Author ID 57221375427, RSCI SPIN-code 9999-1460, http://orcid.org/0000-0002-1039-2429

Ilya E. Tarasov, Dr. Sci. (Eng.), Associated Professor, Professor, Department of Corporate Information Systems, Institute of Information Technologies, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: tarasov_i@mirea.ru. Scopus Author ID 57213354150, RSCI SPIN-code 4628-7514, http://orcid.org/0000-0001-6456-4794