УДК 004.31.85 EDN: KHNNVS

АРХИТЕКТУРЫ АППАРАТНЫХ УСКОРИТЕЛЕЙ ГЛУБОКИХ НЕЙРОННЫХ СЕТЕЙ НА БАЗЕ ПРОГРАММИРУЕМЫХ ПОЛЬЗОВАТЕЛЕМ ВЕНТИЛЬНЫХ МАТРИЦ

О.В. Зобов zobovov@student.bmstu.ru
В.А. Шахнов shakhnov@bmstu.ru

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация

Аннотация

Стремительное развитие технологий глубокого обучения и их широкое внедрение в различных областях требует эффективных решений для аппаратного ускорения вычислительно сложных моделей нейронных сетей. В качестве аппаратной платформы для акселерации задач глубокого обучения особый интерес представляют программируемые пользователем вентильные матрицы, сочетающие гибкость перепрограммирования и эффективность аппаратной реализации. Вентильные матрицы обеспечивают возможность тонкой настройки вычислительных конвейеров и совершенствования иерархии памяти, что позволяет достичь существенного снижения латентности и повышения энергоэффективности при выполнении фазы обучения и логического вывода. Приведены теоретические и практические достижения в усовершенствовании компонентов и архитектуры программируемых пользователем вентильных матриц для эффективного ускорения алгоритмов глубокого обучения. Рассмотрены различные подходы к построению акселераторов: от структурно-фиксированных ускорителей до программно-конфигурируемых аппаратных ускорителей, обеспечивающих баланс между производительностью и адаптивностью ускорителя. Особое внимание уделено усовершенствованию компонентов программируемых пользователем вентильных матриц и их специализации для эффективной реализации базовых операций глубокого обучения, включая матричные вычисления и операции умножения с накоплением различной точности

Ключевые слова

Глубокое обучение, аппаратные ускорители, структурно-фиксированные ускорители, программно-конфигурируемые аппаратные ускорители, архитектура вычислительных систем

Поступила 14.02.2025
Принята 29.08.2025

Отдельные результаты работы получены в рамках выполнения государственного задания (FSFN-2024-0086)

Введение. Глубокое обучение является подразделом машинного обучения, изучающим искусственные нейронные сети со множеством слоев, которые автоматически обучаются иерархическим представлениям данных. Иерархические представления позволяют нейронной сети поэтапно выявлять абстракции: от простых паттернов, таких как линии или цвета, до сложных концепций — объектов или текста, что важно для анализа многомерных данных и достижения высокой точности в задачах, требующих понимания контекста и структуры [1].

В настоящее время выделяют две доминирующие архитектуры глубокого обучения: глубокие нейронные сети (ГНС) и трансформерные нейронные сети. Среди ГНС обычно различают три основных типа: многослойные перцептроны (МСП), сверточные и рекуррентные нейронные сети (СНС и РНС) [2].

Многослойные перцептроны представляют собой полносвязные сети с прямым распространением сигнала [3]. Сверточные нейронные сети используют сверточные слои для эффективного извлечения признаков, что делает их особенно полезными в задачах обработки изображений [4]. Рекуррентные нейронные сети, благодаря наличию внутренней памяти, применяются для обработки последовательных данных, особенно в задачах обработки естественного языка [5].

Трансформерная нейронная сеть — это архитектура модели глубокого обучения, основанная на механизме внимания, которая позволяет модели анализировать зависимости между всеми элементами входной последовательности одновременно, без использования рекуррентных или сверточных слоев. Трансформеры стали основой для современных языковых моделей и применяются в задачах обработки естественного языка, а также в задачах компьютерного зрения [6].

Выбор конкретной архитектуры зависит от целевого применения и ресурсных ограничений вычислительной системы. Несмотря на различия, ГНС и трансформерные сети разделяют многие базовые принципы глубокого обучения.

Основные требования к аппаратным акселераторам для логического вывода ГНС. Требования к аппаратной реализации ГНС определяются условиями развертывания модели и характером решаемых задач, что формирует комплекс ограничений по производительности, энергоэффективности и экономической целесообразности [7].

Производительность. Эффективность ускорителей ГНС оценивается по двум основным метрикам: латентности, определяющей время обработки единичного пакета, и пропускной способности, характеризующей количе-

ство обрабатываемых входных пакетов за единицу времени. Пропускная способность измеряется в гига- или тераоперациях в секунду. Под операциями понимается совокупность умножений и накоплений (УНК), составляющих основу вычислений в задачах ускорения ГНС. Существенное различие между пиковой пропускной способностью ускорителя, определяемой его аппаратными характеристиками, и эффективной пропускной способностью обусловлено практической невозможностью достижения полной загрузки УНК-блоков [8]. В целях максимизации пропускной способности применяется механизм пакетной обработки данных, позволяющий улучшить вычислительные ресурсы за счет многократного использования весовых коэффициентов. Тем не менее рост пропускной способности, достигнутый за счет пакетной обработки, приводит к увеличению задержки, что создает необходимость поиска компромисса между этими метриками в зависимости от требований конкретного приложения [9].

Энергоэффективность и экономическая целесообразность — основные факторы при совершенствовании ускорителей ГНС для их внедрения в автономные устройства. При выполнении логического вывода ГНС на устройствах с батарейным питанием действуют жесткие ограничения энергопотребления, что требует применения высокоэффективного вычислительного оборудования. Специализированные заказные микросхемы позволяют достичь лучших показателей энергоэффективности, однако они имеют существенные ограничения — отсутствие гибкости в адаптации к различным системам и алгоритмам, значительные невозвратные затраты на проектирование и увеличенное время разработки, производства и тестирования, что может оказаться критичным в ряде применений [10].

Ускорители ГНС, кроме высокой производительности и энергоэффективности, должны быть адаптивны к быстро эволюционирующим алгоритмам, темпы развития которых значительно превосходят темпы разработки аппаратных средств [11]. Реализация адаптивности на программном уровне сопряжена с дополнительными затратами энергии и производительности по сравнению со специализированными ускорителями с фиксированной функциональностью. Во встраиваемых системах ускоритель ГНС функционирует как элемент комплексной архитектуры, что требует его эффективного взаимодействия с разнообразными датчиками и исполнительными устройствами. Многообразие протоколов обмена данными и требований к их обработке обусловливает необходимость адаптивности не только вычислительного ядра, но и подсистемы взаимодействия с периферийными устройствами.

Типы аппаратных ускорителей на базе программируемых пользователем вентильных матриц. Структурно-фиксированные аппаратные ускорители. Аппаратное обеспечение с фиксированной функциональностью, создаваемое традиционными средствами высокоуровневого синтеза, предполагает разработку исходного кода на языках описания аппаратуры. Последующая обработка кода системами автоматизированного проектирования обеспечивает генерацию описания на уровне регистровых передач и в виде функционального блока, пригодного для использования в базисе программируемых пользователем вентильных матриц (ППВМ). Ускорители, реализованные по данной методологии, характеризуются высокой пропускной способностью и эффективным использованием ресурсов ППВМ, но ограничены ускорением только заданного алгоритма без возможности адаптации к новым задачам [12]. Существенными недостатками подхода являются необходимость владения низкоуровневыми языками описания аппаратуры и отсутствие стандартизации интерфейсов между управляющей системой и ускорителем в различных проектах. Современной тенденцией в разработке структурно-фиксированных ускорителей становится использование библиотек шаблонов, где пользователь описывает алгоритм с помощью примитивов, предопределенных программным интерфейсом, или доменно-ориентированным языком программирования [13]. Компилятор транслирует код доменно-ориентированного языка в низкоуровневые аппаратные шаблоны на языках описания аппаратуры с последующим преобразованием в функциональные блоки средствами высокоуровневого синтеза. Такой подход обеспечивает унификацию интерфейсов между управляющей системой и ускорителем, а также позволяет использовать единый инструментальный комплекс для ускорения различных алгоритмов в пределах доступной библиотеки шаблонов.

Программно-конфигурируемые аппаратные ускорители. Структурно-фиксированные ускорители, подвергшиеся ручной доработке, обеспечивают высокую производительность и эффективность, но их разработка требует значительных невозвратных затрат на проектирование при ограниченной адаптивности. В противовес этому ускорители на основе генераторов аппаратных средств используют параметризуемые инструментальные комплексы и средства автоматизации проектирования для сокращения времени разработки и повышения адаптивности, часто за счет уменьшения производительности и эффективности использования ресурсов.

Преимущества, проблемы и компромиссы этих парадигм проанализированы в некоторых работах. С помощью автоматизированных инструментов, таких как DNNBuilder [14], AccDNN [15] и FP-DNN [16],

заложены основы проектирования ускорителей на базе генераторов, существенно снижены невозвратные затраты и обеспечена адаптивность для различных платформ ППВМ и моделей ГНС (включая СНС, РНС). Интеграция в данные инструменты методов исследования пространства проектных решений типа DNNExplorer [17] и GANDSE [18] дополнительно сократила разность производительностей структурно-фиксированных ускорителей и ускорителей на основе генераторов. Параллельно применение инструментальных комплексов типа AutoDNNchip [19] позволяет расширить возможности адаптации мультиплатформенных проектов (ППВМ и заказные искусственные сети), демонстрируя потенциал методологий на основе генераторов для создания эффективных решений с минимальным участием человека.

Алгоритм генерации аппаратной архитектуры в составе комплекса AutoDNNchip приведен на рис. 1.

Новые подходы, например, адаптация с использованием искусственного интеллекта [18], гибридные методы RTL-HLS [16], стремятся сбалансировать автоматизацию и производительность для достижения более конкурентоспособных результатов. Несмотря на достижения, структурно-фиксированные ускорители стабильно демонстрируют более высокую производительность и эффективность использования ресурсов благодаря целевым модификациям под конкретные архитектуры нейронных сетей, что подтверждается исследованиями fpgaConvNet [20] и DeepBurning [21]. Однако циклы разработки таких ускорителей остаются неприемлемо длительными, ограничивая их практическое применение для эволюционирующих моделей или новых платформ ППВМ.

Для преодоления разрыва между адаптивностью программно-конфигурируемых и производительностью структурно-фиксированных ускорителей появились гибридные подходы, объединяющие прецизионную настройку с автоматизацией, например FP-DNN [22]. Эти методы, наряду с инновационными методами совершенствования, например, смешанное целочисленное программирование [23], отражают продолжающиеся усилия по снижению невозвратных затрат при достижении производительности, близкой к производительности структурно-фиксированных ускорителей, для широкого спектра моделей ГНС и платформ ППВМ.

Программно-конфигурируемые ускорители демонстрируют динамическую адаптацию к различным вычислительным задачам без необходимости полного аппаратного ресинтеза. В [24–26] рассмотрены программируемые доменно-ориентированные архитектуры RSN, DLA и Vis-TOP

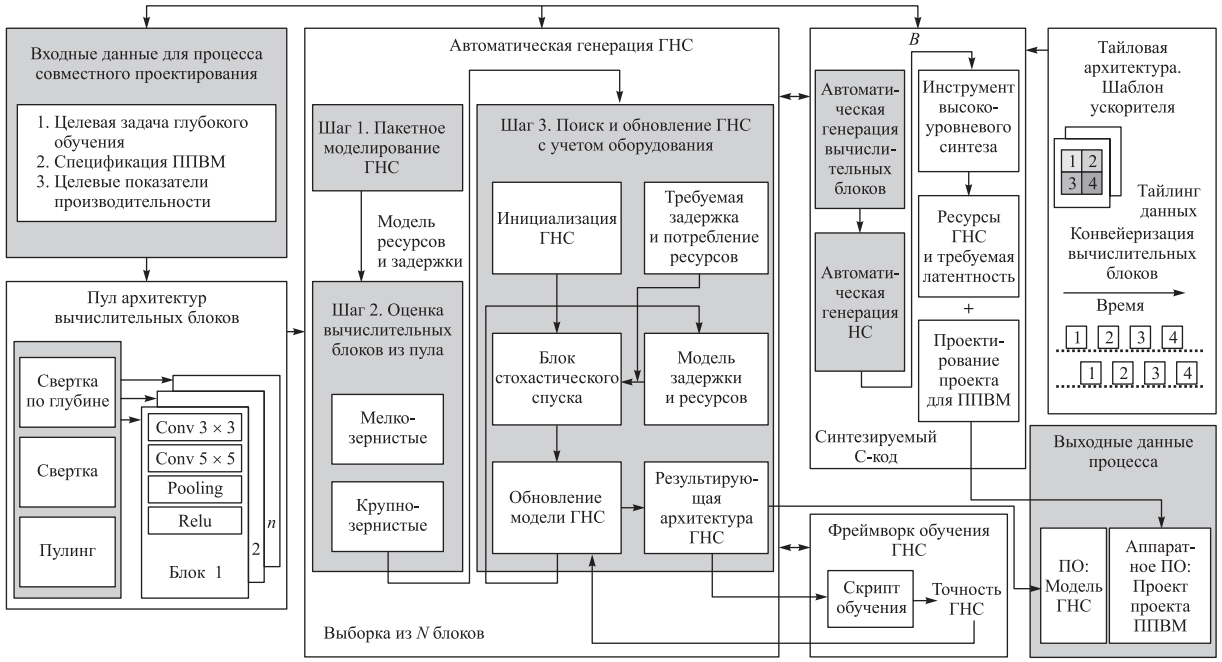

Рис. 1. Алгоритм генерации аппаратной архитектуры в составе комплекса AutoDNNchip

ускорителей. В частности, в [24] предложена адаптивная конвейерная архитектура RSN, обеспечивающая реконфигурируемость на уровне потока данных для сверточных и трансформерных нейронных сетей. В [25] рассмотрена архитектура DLA ускорителей со сверхдлинным командным словом в сочетании с предметно-ориентированным компилятором графов для поддержки различных семейств нейронных сетей, включая сверточные и рекуррентные. В [26] архитектура Vis-TOP развивает эти концепции применительно к визуальным трансформерам, вводя модульные вычислительные ядра трех типов (блок линейной проекции, трансформерный блок и блок пулинга с настраиваемым числом гиперпараметров), что позволяет использовать различные модели трансформерных сетей без необходимости изменения синтезированного аппаратного обеспечения.

Наряду с реконфигурируемостью важным направлением становится программное задание функциональности. Эти подходы используют высокоуровневые инструментальные комплексы, предметно-ориентированные языки программирования или компиляторы для автоматического преобразования программных описаний моделей в аппаратные ускорители. В инструментальных комплексах FP-DNN [22] и DNNBuilder [27] объединены высокоуровневый синтез и параметризуемые аппаратные шаблоны в целях повышения эффективности ускорителей сверточных сетей. В более современных работах [24, 26] предлагается программно-определяемая реконфигурация, обеспечивающая динамическую адаптацию для различных моделей. В [28] архитектура OPU, ориентированная на сверточные сети, использует аналогичный программно-ориентированный подход, применяя управление на основе инструкций для исключения необходимости повторного синтеза аппаратной конфигурации ППВМ.

Несмотря на достигнутый прогресс, сохраняются ключевые проблемы в расширении ускорителей для одновременной поддержки СНС и трансформерных нейронных сетей. Хотя такие архитектуры, как RSN [24] и DLA [25], стремятся поддерживать универсальные вычислительные задачи, большинство существующих систем остаются узкоспециализированными. В литературе по программно-конфигурируемым аппаратным ускорителям, разработанным для СНС, например, [20] fpgaConvNet и [28] OPU, акцентируется внимание на мелкозернистом параллелизме и переиспользовании ресурсов, однако не рассматривается обеспечение аппаратно-программной адаптации под требования трансформеров. В [29] отмечено, что NPE и аналогичные решения для трансформеров специализируются на ускорении обработки механизмов внимания и функций софтмакс, не затрагивая архитектурные особенности, критичные для СНС.

Основные компоненты архитектуры программируемых пользователем вентильных матриц и их модификации для задач ускорения вывода логической ГНС. Таблицы поиска (основные логические блоки ППВМ) изначально проектируются для реализации универсальных булевых функций, однако современные исследования доказали возможность их целенаправленной адаптации под выполнение арифметических операций в специализированных вычислительных задачах. Эта тенденция особенно актуальна для нейронных сетей с весами пониженной точности, в которых интенсивно используются арифметические операции, такие как матричные умножения и операции умножения с накоплением, часто реализуемые с квантованными (например, 4-битными или бинарными) весами и активациями.

Исследования в этой области можно разделить на совершенствование архитектуры таблиц поиска, использование их возможностей для реализации энергоэффективной арифметики чисел с плавающей запятой пониженной точности и согласование ресурсов ППВМ с вычислительными требованиями квантованных нейронных сетей. Современные подходы концентрируются на встраивании арифметических операций непосредственно в таблицы поиска через предварительно вычисленные статические таблицы или специализированные арифметические умножители. Например, компактные умножители, работающие на таблицах поиска [30], обеспечивают ресурсоэффективное 4-битное умножение, встраивая предварительно вычисленные результаты в таблицы поиска, исключая зависимость от блоков цифровой обработки сигналов (ЦОС) при поддержке высокопараллельных вычислений. Аналогично такие архитектурные модификации, как табличные УНК [31], объединяют статические архитектуры на основе таблиц поиска с адаптивной маршрутизацией, что позволяет масштабировать выполнение операций в нейронных сетях с квантованием 2–4 бита.

Еще одно из направлений исследований сфокусировано на приближенной или реконфигурируемой арифметике для вычислений с пониженной точностью. Исследования [32] DyRecMul и умножителей с сохранением переноса [33] предлагают динамически конфигурируемые архитектуры, имеющие незначительные потери точности и существенную экономию площади и энергопотребления, что хорошо согласуется с допустимостью ошибок в нейронных сетях с весами пониженной точности. Подходы на основе гибридных архитектур [34] демонстрируют комбинированные методы улучшения, где производительность ЦОС дополняется программируемостью таблиц поиска, достигая баланса между вычислительной эффективностью и масштабируемостью.

Ключевым результатом является интеграция модификаций архитектуры таблиц поиска и методов квантованного проектирования нейронных сетей, позволяющая реализовывать целостные квантованные модели на ППВМ. В [35, 36] LUTNet и [37, 38] LogicShrinkage рассмотрены прореживание с учетом разреженности и совместное проектирование ПО и аппаратуры для отображения булевых операций логического вывода непосредственно в таблицах поиска при существенном уменьшении площади и энергопотребления для бинарных и субвосьмибитных сетей. С помощью этих методов согласуются разреженность нейронных сетей и специфика ППВМ, достигая передовых результатов по латентности и энергоэффективности на эталонных наборах данных CIFAR-10 и ImageNet [37, 39]. Параллельно, такие методы, как алгоритм Винограда для свертки в квантованных нейронных сетях [40, 41] и динамическая регулировка точности во время выполнения [32, 42], решают ключевые проблемы, связанные с масштабированием ресурсов и адаптивностью к вычислительным задачам.

Блоки цифровой обработки сигналов. В ППВМ и заказных искусственных сетях они предназначены для арифметики с фиксированной высокой точностью (например, умножение 27 × 18 бит в XilinxDSP48E2), что приводит к неэффективному использованию ресурсов при выполнении вычислений с пониженной точностью, характерных для задач ускорения ГНС (например, INT8, INT4 или бинарные операции). Это несоответствие стимулирует исследования в области фрагментации умножителей блоков ЦОС (разделение крупных блоков умножения на меньшие независимо функционирующие модули) для повышения эффективности использования аппаратных ресурсов, энергоэффективности и производительности логического вывода ГНС.

Современные исследования направлены на интегрированное применение фрагментации блоков ЦОС, реконфигурируемой логики и квантования смешанной точности для эффективного решения специфических задач глубокого обучения. Методы динамического масштабирования точности позволяют адаптировать разрядность вычислений (например, 4, 8 или 16 бит) в зависимости от требований конкретных слоев [43–46]. Методы уплотнения блоков ЦОС максимизируют использование ресурсов путем объединения нескольких операций умножения пониженной точности в одном блоке ЦОС [47–50], например, метод двойного умножения с накоплением (двойной УНК) [43], обеспечивающий параллельное выполнение двух 8-битных операций УНК в одном блоке ЦОС ППВМ. Настройка режимов функционирования блоков ЦОС и совершенствование распределения ресурсов позволили достичь двукратного повышения эффективности использования

блоков на уровне слоев СНС и общего прироста производительности до 80 % на уровне сети без значительных потерь точности вычислений. Структурная схема модифицированного блока ЦОС для DSP48 [30] приведена на рис. 2.

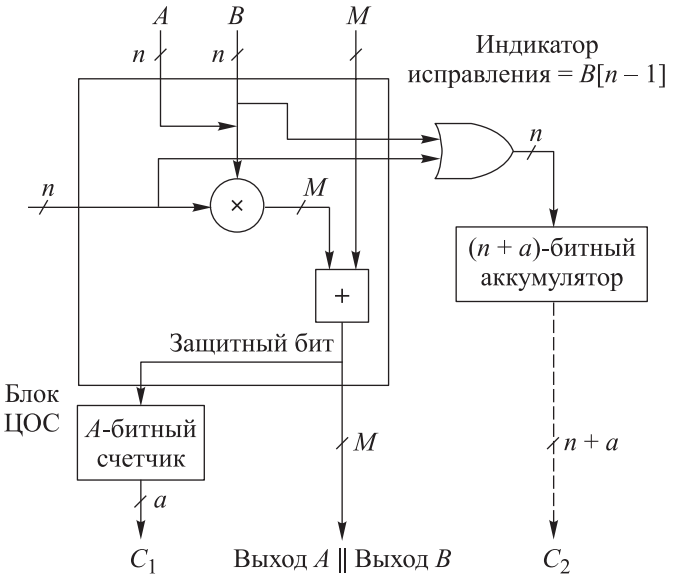

Рис. 2. Структурная схема модифицированного блока ЦОС для DSP48 [30]

Метод уплотнения блоков ЦОС [47] обобщает процесс фрагментации для произвольной разрядности, в то время как подходы, типа переуплотнения, имеют небольшие потери точности (< 0,4 % средней абсолютной погрешности) при значительном повышении эффективности использования блоков ЦОС (шесть умножений 4-битных чисел на один 48-разрядный блок ЦОС).

Квантование смешанной точности обеспечивает более тонкий контроль над точностью в различных слоях, максимизируя вычислительную эффективность без ущерба для точности модели. Системы типа PIR-DSP [51] и Uint-Packing [48] обеспечивают гибкую реконфигурацию архитектур умножителей во время выполнения. В [43, 52] исследованы методы интеграции аппаратных средств пониженной точности с квантованными ГНС. Эти разработки проверяются на эталонных сверточных моделях (ResNet, VGG, AlexNet), показано увеличение в 3–15 раз производительности при сохранении точности [53–55].

Энергетическая и экономическая эффективности являются центральными темами многих исследований, рассматривающих фрагментированные умножители, включающие в себя методы приближенных вычислений

(усечение, стробирование) для дальнейшего снижения энергопотребления. Например, в [56] предложен механизм стробирования для весов пониженной точности при достижении уменьшения энергопотребления на ~ 35 %, в [57–59] приведены приближенные умножители, работающие с незначительной потерей точности (< 1 %) и экономией до 67 % площади и энергии.

Несмотря на эти достижения, сохраняются проблемы, связанные с накладными расходами на маршрутизацию, сложностью управления и масштабированием для различных архитектур ГНС. Попытки решить эти проблемы прослеживаются в архитектурных инновациях, таких как систолические массивы в сочетании с фрагментированными умножителями [54, 55] и методы вычислений в памяти, подобные M4BRAM [60], которые переносят вычисления в ресурсы памяти. Кроме того, тесная интеграция модификаций блоков ЦОС с платформами автоматического поиска архитектур нейронных сетей [54, 61] иллюстрирует преимущества совместного проектирования алгоритмов и аппаратных решений для значительного улучшения быстродействия ГНС.

Встроенные блоки оперативных запоминающих устройств. Ключевой особенностью ППВМ являются встроенные блоки оперативной памяти (т. е. блочные ОЗУ), обеспечивающие распределенное низколатентное хранение данных на кристалле. В последнее время блочные ОЗУ исследуются не только как модули памяти, но и как активные вычислительные элементы, реализующие обработку данных внутри себя. Алгоритмы семейства BRAMAC, адаптированные под низкоразрядные (2–8 бит) целочисленные вычисления, реализуют комбинированную битпоследовательно-параллельную архитектуру, направленную на максимизацию производительности логического вывода ГНС [62].

Структурная диаграмма модифицированной ячейки блочного ОЗУ на базе Intel M20K BRAM [49] приведена на рис. 3.

В то же время проект M4BRAM расширяет возможности вычислений в памяти для задач смешанной точности, которые становятся все более популярными в нейронных сетях, поддерживая как хранение, так и параллельные вычисления для матричных умножений [60].

Проекты IMAGine [63] и TheBRAMistheLimit [64] предлагают масштабируемые парадигмы проектирования, достигающие полного использования тактовой частоты блочного ОЗУ (737 МГц) и линейного масштабирования плотности обработки, обеспечивая до 64 тыс. параллельных блоков УНК. Эти результаты опровергают предыдущие предположения о том,

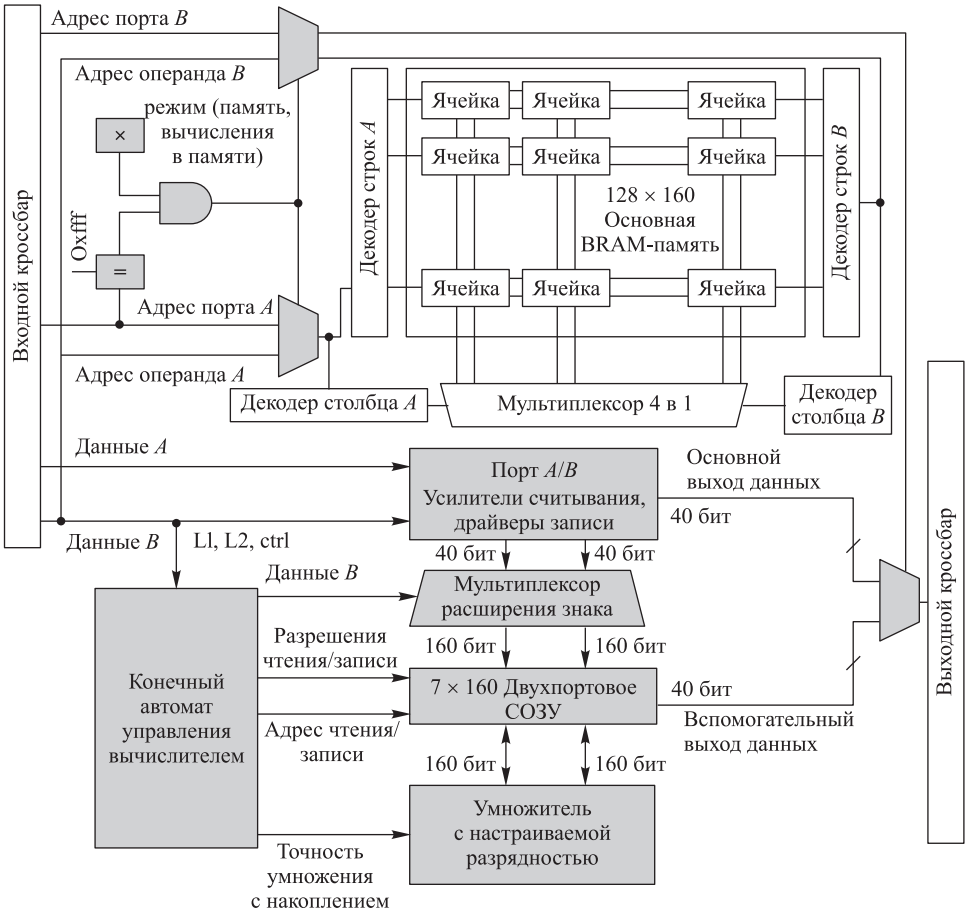

Рис. 3. Структурная диаграмма модифицированной ячейки блочного ОЗУ на базе Intel M20K BRAM [49]

что архитектуры внутрипамятной обработки на основе BRAM неизбежно страдают от снижения тактовой частоты при высоких нагрузках.

Заключение. Рассмотрены современные тенденции в области аппаратных ускорителей для задач глубокого обучения, реализуемых на базе программируемых пользователем вентильных матриц. Проанализирована эволюция подходов к ускорению вычислений — от традиционных методов машинного обучения к глубоким нейронным сетям, что обусловило необходимость разработки специализированных аппаратных решений.

Приведена классификация типов ускорителей, включающая в себя структурно-фиксированные и программно-конфигурируемые архитектуры. Рассмотрены характеристики каждого подхода: высокая производительность, но ограниченная гибкость структурно-фиксированных ускорителей сопоставлены с адаптивностью программно-конфигурируемых решений

при их относительно меньшей эффективности. Отмечено развитие гибридных архитектур, стремящихся объединить достоинства обоих подходов.

Рассмотрены методы модернизации базовых компонентов ППВМ. Усовершенствованы таблицы поиска для эффективного выполнения арифметических операций пониженной точности. Приведены методы фрагментации блоков цифровой обработки сигналов, обеспечивающие увеличение производительности операций с пониженной битностью. Показано использование блоков встроенной памяти для реализации вычислений внутри памяти. Проанализировано влияние этих модификаций на увеличение производительности и энергоэффективности аппаратных ускорителей.

ЛИТЕРАТУРА

[1] LeCun Y., Bengio Y., Hinton G. Deep learning. *Nature*, 2015, vol. 521, no. 7553, pp. 436–444. DOI: https://doi.org/10.1038/nature14539

[2] Schmidhuber J. Deep learning in neural networks: an overview. *Neural Netw.*, 2015, vol. 61, pp. 85–117. DOI: https://doi.org/10.1016/j.neunet.2014.09.003

[3] Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.*, 1958, vol. 65, no. 6, pp. 386–408.
DOI: https://doi.org/10.1037/h0042519

[4] LeCun Y., Bottou L., Bengio Y., et al. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998, vol. 86, no. 11, pp. 2278–2324.
DOI: https://doi.org/10.1109/5.726791

[5] Rumelhart D.E., Hinton G.E., Williams R.J. Learning internal representations by error propagation. *Technical Report ICS-8504*. San Diego, University of California, Institute for Cognitive Science, 1985.

[6] Vaswani A., Shazeer N., Parmar N., et al. Attention is all you need. *Proc. 31st. NIPS*, 2017, pp. 6000–6010. DOI: https://doi.org/10.1007/s11704-025-50480-3

[7] Шахнов В.А., Власов А.И., Поляков Ю.А. и др. Нейрокомпьютеры: архитектура и схемотехника. М., Машиностроение, 2000. EDN: RVYJUX

[8] Левин И.И., Дордопуло А.И., Каляев И.А. и др. Высокопроизводительные реконфигурируемые вычислительные системы на основе ПЛИС Virtex-7. *Труды Института математики и информатики Национальной академии наук Беларуси*, 2014, № 6, с. 3–7.

[9] Каляев И.А., Левин И.И. Реконфигурируемые мультиконвейерные вычислительные системы для решения потоковых задач. *Известия Южного федерального университета. Технические науки*, 2011, № 2, с. 12–22. EDN: OZQLKX

[10] Ахметов Н.Р., Власов А.И., Димитров Д.А. и др. Перспективная элементная база для СМАРТ-систем в условиях цифровой трансформации промышленности. *Датчики и системы*, 2021, № 1, с. 9–17.
DOI: https://doi.org/10.25728/datsys.2021.1.2

[11] Дордопуло А.И., Каляев И.А., Левин И.И. и др. Высокопроизводительные реконфигурируемые вычислительные системы нового поколения. *Вычислительные методы и программирование*, 2011, т. 12, № 4, с. 82–89. EDN: OJAZNN

[12] Власов А.И. Аппаратная реализация нейровычислительных управляющих систем. *Приборы и системы. Управление, контроль, диагностика*, 1999, № 2, с. 61–65. EDN: TEKPVZ

[13] Sozzo E.D., Conficconi D., Zeni A., et al. Pushing the level of abstraction of digital system design: a survey on how to program FPGAs. *ACM Comput. Surv.*, 2023, vol. 55, no. 5, art. 106. DOI: https://doi.org/10.1145/3532989

[14] Zhang X., Wang J., Zhu C., et al. DNNBuilder: an automated tool for building high-performance DNN hardware accelerators for FPGAs. *ICCAD'18*, 2018, art. 56. DOI: https://doi.org/10.1145/3240765.3240801

[15] Zhang X., Wang J., Zhu C., et al. AccDNN: an IP-Based DNN generator for FPGAs. *IEEE 26th FCCM*, 2018, p. 210. DOI: https://doi.org/10.1109/FCCM.2018.00044

[16] Guan Y., Liang H., Xu N., et al. FP-DNN: an automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. *IEEE 25th FCCM*, 2017, pp. 152–159. DOI: https://doi.org/10.1109/FCCM.2017.25

[17] Zhang X., Ye H., Wang J., et al. DNNExplorer: a framework for modeling and exploring a novel paradigm of FPGA-based DNN accelerator. *ICCAD'20*, 2020, art. 61. DOI: https://doi.org/10.1145/3400302.3415609

[18] Feng L., Liu W., Guo C., et al. GANDSE: generative adversarial network-based design space exploration for neural network accelerator design. *ACM TODAES*, 2023, vol. 28, no. 3, art. 35. DOI: https://doi.org/10.1145/3570926

[19] Xu P., Zhang X., Hao C., et al. AutoDNNchip: an automated DNN chip predictor and builder for both FPGAs and ASICs. *FPGA'20*, 2020, pp. 40–50. DOI: https://doi.org/10.1145/3373087.3375306

[20] Venieris S.I., Bouganis C. fpgaConvNet: a framework for mapping convolutional neural networks on FPGAs. *IEEE 24th FCCM*, 2016, pp. 40–47. DOI: https://doi.org/10.1109/FCCM.2016.22

[21] Wang Y., Xu J., Han Y., et al. DeepBurning: automatic generation of FPGA-based learning accelerators for the neural network family. *DAC'16*, 2016, art. 110. DOI: https://doi.org/10.1145/2897937.2898003

[22] Guan Y., Liang H., Xu N., et al. FP-DNN: an automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. *IEEE 25th FCCM*, 2017, pp. 152–159. DOI: https://doi.org/10.1109/FCCM.2017.25

[23] Ding Y., Wu J., Gao Y., et al. Model-platform optimized deep neural network accelerator generation through mixed-integer geometric programming. *IEEE 31st FCCM*, 2023, pp. 83–93. DOI: https://doi.org/10.1109/FCCM57271.2023.00018

[24] Wang C., Zhang X., Cong J., et al. Addressing architectural obstacles for overlay with stream network abstraction. *ArXiv:2411.17966.* URL: https://arxiv.org/abs/2411.17966v1

[25] Abdelfattah M., Han D., Bitar A., et al. DLA: compiler and FPGA overlay for neural network inference acceleration. *28th FPL*, 2018, pp. 411–417.
DOI: https://doi.org/10.1109/FPL.2018.00077

[26] Hu W., Xu D., Fan Z., et al. Vis-TOP: visual transformer overlay processor. *ArXiv:2110.10957*. DOI: https://doi.org/10.48550/arXiv.2110.10957

[27] Zhang X., Wang J., Zhu C., et al. DNNBuilder: an automated tool for building high-performance DNN hardware accelerators for FPGAs. *ICCAD'18*, 2018, art. 56.
DOI: https://doi.org/10.1145/3240765.3240801

[28] Bai Y., Zhou H., Zhao K., et al. LTrans-OPU: a low-latency FPGA-based overlay processor for transformer networks. *33rd FPL*, 2023, pp. 283–287.
DOI: https://doi.org/10.1109/FPL60245.2023.00048

[29] Khan H., Khan A., Khan Z.F., et al. NPE: an FPGA-based overlay processor for natural language processing. *FPGA'21*, 2021, p. 227.
DOI: https://doi.org/10.1145/3431920.3439477

[30] Zhao B.-B., Wang Y., Zhang H., et al. 4-bit CNN Quantization method with compact LUT-based multiplier implementation on FPGA. *IEEE Trans. Instrum. Meas.*, 2023, vol. 72, art. 2008110. DOI: https://doi.org/10.1109/TIM.2023.3324357

[31] Gerlinghoff D., Choong B.C.M., Goh R., et al. Table-lookup MAC: scalable processing of quantised neural networks in FPGA soft logic. *FPGA'24*, 2024, pp. 235–245.
DOI: https://doi.org/10.1145/3626202.3637576

[32] Vakili S., Vaziri M., Zarei A., et al. DyRecMul: fast and low-cost approximate multiplier for FPGAs using dynamic reconfiguration. *arXiv:2310.10053*.
DOI: https://doi.org/10.48550/arXiv.2310.10053

[33] Awais M., Zahir A., Shah S.A.A., et al. Toward optimal softcore carry-aware approximate multipliers on xilinx FPGAs. *ACM TECS*, 2023, vol. 22, no. 4, art. 76.
DOI: https://doi.org/10.1145/3564243

[34] Haghi P., Kamal M., Afzali-Kusha A., et al. O^4-DNN: a hybrid DSP-LUT-based processing unit with operation packing and out-of-order execution for efficient realization of convolutional neural networks on FPGA devices. *IEEE Trans. Circuits Syst. I: Regul. Pap.*, 2020, vol. 67, no. 9, pp. 3056–3069.
DOI: https://doi.org/10.1109/TCSI.2020.2986350

[35] Wang E., Davis J.J., Cheung P., et al. LUTNet: rethinking inference in FPGA soft logic. *IEEE 27th FCCM*, 2019, pp. 26–34.
DOI: https://doi.org/10.1109/FCCM.2019.00014

[36] Wang E., Davis J.J., Cheung P., et al. LUTNet: learning FPGA configurations for highly efficient neural network inference. *IEEE Trans. Comput.*, 2020, vol. 69, no. 12, pp. 1795–1808. DOI: https://doi.org/10.1109/TC.2020.2978817

[37] Wang E., Auffret M., Stavrou G., et al. Logic shrinkage: learned connectivity sparsification for LUT-based neural networks. *ACM TRETS*, 2023, vol. 16, no. 4, art. 57.
DOI: https://doi.org/10.1145/3583075

[38] Wang E., Davis J.J., Stavrou G., et al. Logic shrinkage: learned FPGA netlist sparsity for efficient neural network inference. *FPGA'22*, 2022, pp. 101–111.
DOI: https://doi.org/10.1145/3490422.3502360

[39] Xie Y., Li Z., Diaconu D., et al. LUTMUL: exceed conventional FPGA roofline limit by LUT-based efficient multiplication for neural network inference. *ArXiv2411.11852*.
DOI: https://doi.org/10.48550/arXiv.2411.11852

[40] Cao Y., Wang C., Tang Y. Explore efficient LUT-based architecture for quantized convolutional neural networks on FPGA. *IEEE 28th FCCM*, 2020, p. 232.
DOI: https://doi.org/10.1109/FCCM48280.2020.00065

[41] Cao Y., Song C., Tang Y. Efficient LUT-based FPGA accelerator design for universal quantized CNN inference. *ASSE'21*, 2021, pp. 108–115.
DOI: https://doi.org/10.1145/3456126.3456140

[42] Neda N., Ullah S., Ghanbari A., et al. Multi-precision deep neural network acceleration on FPGAs. *27th ASP-DAC*, 2022, pp. 454–459.
DOI: https://doi.org/10.1109/asp-dac52403.2022.9712485

[43] Lee S., Kim D., Nguyen D., et al. Double MAC on a DSP: boosting the performance of convolutional neural networks on FPGAs. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2019, vol. 38, no. 5, pp. 888–897.
DOI: https://doi.org/10.1109/TCAD.2018.2824280

[44] Ding C. Dynamic precision multiplier for deep neural network accelerators. *IEEE 33rd SOCC*, 2020, pp. 180–184. DOI: https://doi.org/10.1109/socc49529.2020.9524752

[45] Neda N. Multi-precision deep neural network acceleration on FPGAs. *27th ASP-DAC*, 2022, pp. 454–459. DOI: https://doi.org/10.1109/asp-dac52403.2022.9712485

[46] Raees P.C.M., Akshayraj M.R., Gopi Varun P., et al. Dynamic precision scaling in MAC units for energy-efficient computations in deep neural network accelerators. *28th VDAT*, 2024. DOI: https://doi.org/10.1109/VDAT63601.2024.10705697

[47] Sommer J., Özkan A., Keszocze O., et al. DSP-packing: squeezing low-precision arithmetic into FPGA DSP blocks. *32nd FPL*, 2022, pp. 160–166.
DOI: https://doi.org/10.1109/FPL57034.2022.00035

[48] Zhang J., Zhang M., Cao X., et al. Uint-packing: multiply your DNN accelerator performance via unsigned integer DSP packing. *60th ACM/IEEE DAC*, 2023.
DOI: https://doi.org/10.1109/DAC56929.2023.10247773

[49] Kalali E., van Leuken R. Near-precise parameter approximation for multiple multiplications on a single DSP block. *IEEE Trans. Comput.*, 2022, vol. 71, no. 9, pp. 2036–2047. DOI: https://doi.org/10.1109/TC.2021.3119187

[50] Li R., Jiang B., Xu H. Mixed DSP packing method for convolutional neural network on FPGA. *Proc. SPIE*, 2023, vol. 12800. DOI: https://doi.org/10.1117/12.3004070

[51] Rasoulinezhad S., Zhou H., Wang L., et al. PIR-DSP: an FPGA DSP block architecture for multi-precision deep neural networks. *IEEE 27th FCCM*, 2019, pp. 35–44.
DOI: https://doi.org/10.1109/FCCM.2019.00015

[52] Liu Y., Rai S., Ullah S., et al. High-flexibility designs of quantized runtime reconfigurable multi-precision multipliers. *IEEE Embed. Syst. Lett.*, 2023, vol. 15, no. 4, pp. 194–197. DOI: https://doi.org/10.1109/LES.2023.3298736

[53] Liu X., Wu X., Shao H., et al. A flexible FPGA-based accelerator for efficient inference of multi-precision CNNs. *IEEE ISCAS*, 2024.
DOI: https://doi.org/10.1109/ISCAS58744.2024.10557882

[54] Huang M., Liu Y., Huang S., et al. Multi-bit-width CNN accelerator with systolic-in-systolic dataflow and single DSP multiple multiplication scheme. *FPGA'23*, 2023, p. 229.
DOI: https://doi.org/10.1145/3543622.3573209

[55] Huang M., Liu Y., Man C., et al. A high performance multi-bit-width booth vector systolic accelerator for NAS optimized deep learning neural networks. *IEEE Trans. Circuits Syst. I: Regul. Pap.*, 2022, vol. 69, no. 9, pp. 3619–3631.
DOI: https://doi.org/10.1109/TCSI.2022.3178474

[56] Zheng Y., Li Z., Sun K., et al. A 40 nm area-efficient effective-bit-combination-based DNN accelerator with the reconfigurable multiplier. *IEEE 5th AICAS*, 2023.
DOI: https://doi.org/10.1109/AICAS57966.2023.10168550

[57] Ghavami B., Sajadi M., Shannon L., et al. Boosting multiple multipliers packing on FPGA DSP blocks via truncation and compensation-based approximation. *IEEE ISVLSI*, 2024, pp. 222–227. DOI: https://doi.org/10.1109/ISVLSI61997.2024.00049

[58] Rehman A., Vakili S. A cost-effective FPGA-based approximate multiplier for machine learning acceleration. *IEEE 14th PAAP*, 2023.
DOI: https://doi.org/10.1109/PAAP60200.2023.10391619

[59] Ullah S., Rehman S., Prabakaran B., et al. Area-optimized low-latency approximate multipliers for FPGA-based hardware accelerators. *DAC'18*, 2018, art. 159.
DOI: https://doi.org/10.1145/3195970.3195996

[60] Chen Y., Dotzel J., Abdelfattah M. M4BRAM: mixed-precision matrix-matrix multiplication in FPGA block RAMs. *ICFPT*, 2023, pp. 69–78.
DOI: https://doi.org/10.1109/ICFPT59805.2023.00013

[61] Luo E., Huang H., Liu C., et al. DeepBurning-MixQ: an open source mixed-precision neural network accelerator design framework for FPGAs. *IEEE/ACM ICCAD*, 2023. DOI: https://doi.org/10.1109/ICCAD57390.2023.10323831

[62] Chen Y., Abdelfattah M. BRAMAC: compute-in-BRAM architectures for multiply-accumulate on FPGAs. *31st IEEE FCCM*, 2023, pp. 52–62.
DOI: https://doi.org/10.1109/FCCM57271.2023.00015

[63] Kabir M.A., Kamucheka T., Fredricks N., et al. IMAGine: an in-memory accelerated GEMV engine overlay. *34th FPL*, 2024, pp. 220–226.
DOI: https://doi.org/10.1109/FPL64840.2024.00038

[64] Kabir M.A., Kamucheka T., Fredricks N., et al. The BRAM is the limit: shattering myths, shaping standards, and building scalable PIM accelerators. *32nd IEEE FCCM*, 2024, p. 223. DOI: https://doi.org/10.1109/FCCM60383.2024.00045

Зобов Олег Валерьевич — аспирант кафедры «Проектирование и технология производства электронной аппаратуры» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Шахнов Вадим Анатольевич — чл.-корр. РАН, д-р техн. наук, профессор, заведующий кафедрой «Проектирование и технология производства электронной аппаратуры» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Просьба ссылаться на эту статью следующим образом:

Зобов О.В., Шахнов В.А. Архитектуры аппаратных ускорителей глубоких нейронных сетей на базе программируемых пользователем вентильных матриц. *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2025, № 4 (153), с. 78–101.
EDN: KHNNVS

FPGA-BASED ARCHITECTURES FOR DEEP LEARNING ACCELERATORS

O.V. Zobov zobovov@student.bmstu.ru
V.A. Shakhnov shakhnov@bmstu.ru

BMSTU, Moscow, Russian Federation

Abstract

The rapid development of deep learning technologies and their widespread adoption in various fields requires efficient solutions for hardware acceleration of computationally complex neural network models. As a hardware platform for accelerating deep learning tasks, field-programmable gate arrays are of particular interest, combining the flexibility of reprogramming with the efficiency of hardware implementation. They provide the ability to fine-tune computational pipelines and optimize memory hierarchy, which allows for significant reduction in latency and increase in energy efficiency when performing both the training phase and inference of models. The article presents theoretical and practical achievements in optimizing the architecture of field-programmable gate arrays for efficient acceleration of deep learning algorithms. Various approaches to building accelerators are considered — from structurally fixed accelerators to software-configurable hardware accelerators that provide a balance between performance and flexibility. Special attention is paid to the improvement of classical components of field-program-

Keywords

Deep learning, hardware accelerators, structurally fixed accelerators, software-configurable hardware accelerators, computer system architecture

mable gate arrays and their specialization for the efficient implementation of basic deep learning operations, including matrix computations and multiply-accumulate operations of various precisions

Received 14.02.2025
Accepted 29.08.2025

Individual results of the work were obtained within the framework of the State Assignment (FSFN-2024-0086)

REFERENCES

[1] LeCun Y., Bengio Y., Hinton G. Deep learning. *Nature*, 2015, vol. 521, no. 7553, pp. 436–444. DOI: https://doi.org/10.1038/nature14539

[2] Schmidhuber J. Deep learning in neural networks: an overview. *Neural Netw.*, 2015, vol. 61, pp. 85–117. DOI: https://doi.org/10.1016/j.neunet.2014.09.003

[3] Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.*, 1958, vol. 65, no. 6, pp. 386–408. DOI: https://doi.org/10.1037/h0042519

[4] LeCun Y., Bottou L., Bengio Y., et al. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998, vol. 86, no. 11, pp. 2278–2324. DOI: https://doi.org/10.1109/5.726791

[5] Rumelhart D.E., Hinton G.E., Williams R.J. Learning internal representations by error propagation. *Technical Report ICS-8504.* San Diego, University of California, Institute for Cognitive Science, 1985.

[6] Vaswani A., Shazeer N., Parmar N., et al. Attention is all you need. *Proc. 31st. NIPS*, 2017, pp. 6000–6010. DOI: https://doi.org/10.1007/s11704-025-50480-3

[7] Shakhnov V.A., Vlasov A.I., Polyakov Yu.A., et al. Neyrokompyutery: arkhitektura i skhemotekhnika [Neurocomputers: architecture and circuitry]. Moscow, Mashinostroenie Publ., 2000. EDN: RVYJUX

[8] Levin I.I., Dordopulo A.I., Kalyaev I.A., et al. High-performance reconfigurable computing systems based on Virtex-7 FPGAs. *Trudy Instituta matematiki i informatiki Natsionalnoy akademii nauk Belarusi* [Proceedings of the Institute of Mathematics of the National Academy of Sciences of Belarus], 2014, no. 6, pp. 3–7 (in Russ.).

[9] Kalyaev I.A., Levin I.I. Reconfigurable multipipeline computing systems for data-driven tasks of information handling and control solution. *Izvestiya Yuzhnogo federalnogo universiteta. Tekhnicheskie nauki* [Journal of Information Technologies and Computing Systems], 2011, no. 2, pp. 12–22 (in Russ.). EDN: OZQLKX

[10] Akhmetov N.R., Vlasov A.I., Dimitrov D.A., et al. A promising element base for smart systems in a digital transformation of industry. *Datchiki i sistemy* [Sensors & Systems], 2021, no. 1, pp. 9–17 (in Russ.). DOI: https://doi.org/10.25728/datsys.2021.1.2

[11] Dordopulo A.I., Kalyaev I.A., Levin I.I., et al. High-performance reconfigurable computer systems of new generation. *Vychislitelnye metody i programmirovanie* [Numerical Methods and Programming], 2011, vol. 12, no. 4, pp. 82–89 (in Russ.). EDN: OJAZNN

[12] Vlasov A.I. Hardware implementation of neurocomputing control systems. *Upravlenie, kontrol, diagnostika* [Instruments and Systems: Monitoring, Control, and Diagnostics], 1999, no. 2, pp. 61–65 (in Russ.). EDN: TEKPVZ

[13] Sozzo E.D., Conficconi D., Zeni A., et al. Pushing the level of abstraction of digital system design: a survey on how to program FPGAs. *ACM Comput. Surv.*, 2023, vol. 55, no. 5, art. 106. DOI: https://doi.org/10.1145/3532989

[14] Zhang X., Wang J., Zhu C., et al. DNNBuilder: an automated tool for building high-performance DNN hardware accelerators for FPGAs. *ICCAD'18*, 2018, art. 56. DOI: https://doi.org/10.1145/3240765.3240801

[15] Zhang X., Wang J., Zhu C., et al. AccDNN: an IP-Based DNN generator for FPGAs. *IEEE 26th FCCM*, 2018, p. 210. DOI: https://doi.org/10.1109/FCCM.2018.00044

[16] Guan Y., Liang H., Xu N., et al. FP-DNN: an automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. *IEEE 25th FCCM*, 2017, pp. 152–159. DOI: https://doi.org/10.1109/FCCM.2017.25

[17] Zhang X., Ye H., Wang J., et al. DNNExplorer: a framework for modeling and exploring a novel paradigm of FPGA-based DNN accelerator. *ICCAD'20*, 2020, art. 61. DOI: https://doi.org/10.1145/3400302.3415609

[18] Feng L., Liu W., Guo C., et al. GANDSE: generative adversarial network-based design space exploration for neural network accelerator design. *ACM TODAES*, 2023, vol. 28, no. 3, art. 35. DOI: https://doi.org/10.1145/3570926

[19] Xu P., Zhang X., Hao C., et al. AutoDNNchip: an automated DNN chip predictor and builder for both FPGAs and ASICs. *FPGA'20*, 2020, pp. 40–50. DOI: https://doi.org/10.1145/3373087.3375306

[20] Venieris S.I., Bouganis C. fpgaConvNet: a framework for mapping convolutional neural networks on FPGAs. *IEEE 24th FCCM*, 2016, pp. 40–47. DOI: https://doi.org/10.1109/FCCM.2016.22

[21] Wang Y., Xu J., Han Y., et al. DeepBurning: automatic generation of FPGA-based learning accelerators for the neural network family. *DAC'16*, 2016, art. 110. DOI: https://doi.org/10.1145/2897937.2898003

[22] Guan Y., Liang H., Xu N., et al. FP-DNN: an automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. *IEEE 25th FCCM*, 2017, pp. 152–159. DOI: https://doi.org/10.1109/FCCM.2017.25

[23] Ding Y., Wu J., Gao Y., et al. Model-platform optimized deep neural network accelerator generation through mixed-integer geometric programming. *IEEE 31st FCCM*, 2023, pp. 83–93. DOI: https://doi.org/10.1109/FCCM57271.2023.00018

[24] Wang C., Zhang X., Cong J., et al. Addressing architectural obstacles for overlay with stream network abstraction. *ArXiv:2411.17966*. Available at: https://arxiv.org/abs/2411.17966v1

[25] Abdelfattah M., Han D., Bitar A., et al. DLA: compiler and FPGA overlay for neural network inference acceleration. *28th FPL*, 2018, pp. 411–417. DOI: https://doi.org/10.1109/FPL.2018.00077

[26] Hu W., Xu D., Fan Z., et al. Vis-TOP: visual transformer overlay processor. *ArXiv:2110.10957*. DOI: https://doi.org/10.48550/arXiv.2110.10957

[27] Zhang X., Wang J., Zhu C., et al. DNNBuilder: an automated tool for building high-performance DNN hardware accelerators for FPGAs. *ICCAD'18*, 2018, art. 56.
DOI: https://doi.org/10.1145/3240765.3240801

[28] Bai Y., Zhou H., Zhao K., et al. LTrans-OPU: a low-latency FPGA-based overlay processor for transformer networks. *33rd FPL*, 2023, pp. 283–287.
DOI: https://doi.org/10.1109/FPL60245.2023.00048

[29] Khan H., Khan A., Khan Z.F., et al. NPE: an FPGA-based overlay processor for natural language processing. *FPGA'21*, 2021, p. 227.
DOI: https://doi.org/10.1145/3431920.3439477

[30] Zhao B.-B., Wang Y., Zhang H., et al. 4-bit CNN Quantization method with compact LUT-based multiplier implementation on FPGA. *IEEE Trans. Instrum. Meas.*, 2023, vol. 72, art. 2008110. DOI: https://doi.org/10.1109/TIM.2023.3324357

[31] Gerlinghoff D., Choong B.C.M., Goh R., et al. Table-lookup MAC: scalable processing of quantised neural networks in FPGA soft logic. *FPGA'24*, 2024, pp. 235–245.
DOI: https://doi.org/10.1145/3626202.3637576

[32] Vakili S., Vaziri M., Zarei A., et al. DyRecMul: fast and low-cost approximate multiplier for FPGAs using dynamic reconfiguration. *arXiv:2310.10053*.
DOI: https://doi.org/10.48550/arXiv.2310.10053

[33] Awais M., Zahir A., Shah S.A.A., et al. Toward optimal softcore carry-aware approximate multipliers on xilinx FPGAs. *ACM TECS*, 2023, vol. 22, no. 4, art. 76.
DOI: https://doi.org/10.1145/3564243

[34] Haghi P., Kamal M., Afzali-Kusha A., et al. O^4-DNN: a hybrid DSP-LUT-based processing unit with operation packing and out-of-order execution for efficient realization of convolutional neural networks on FPGA devices. *IEEE Trans. Circuits Syst. I: Regul. Pap.*, 2020, vol. 67, no. 9, pp. 3056–3069.
DOI: https://doi.org/10.1109/TCSI.2020.2986350

[35] Wang E., Davis J.J., Cheung P., et al. LUTNet: rethinking inference in FPGA soft logic. *IEEE 27th FCCM*, 2019, pp. 26–34.
DOI: https://doi.org/10.1109/FCCM.2019.00014

[36] Wang E., Davis J.J., Cheung P., et al. LUTNet: learning FPGA configurations for highly efficient neural network inference. *IEEE Trans. Comput.*, 2020, vol. 69, no. 12, pp. 1795–1808. DOI: https://doi.org/10.1109/TC.2020.2978817

[37] Wang E., Auffret M., Stavrou G., et al. Logic shrinkage: learned connectivity sparsification for LUT-based neural networks. *ACM TRETS*, 2023, vol. 16, no. 4, art. 57.
DOI: https://doi.org/10.1145/3583075

[38] Wang E., Davis J.J., Stavrou G., et al. Logic shrinkage: learned FPGA netlist sparsity for efficient neural network inference. *FPGA'22*, 2022, pp. 101–111.
DOI: https://doi.org/10.1145/3490422.3502360

[39] Xie Y., Li Z., Diaconu D., et al. LUTMUL: exceed conventional FPGA roofline limit by LUT-based efficient multiplication for neural network inference. *ArXiv2411.11852*. DOI: https://doi.org/10.48550/arXiv.2411.11852

[40] Cao Y., Wang C., Tang Y. Explore efficient LUT-based architecture for quantized convolutional neural networks on FPGA. *IEEE 28th FCCM*, 2020, p. 232. DOI: https://doi.org/10.1109/FCCM48280.2020.00065

[41] Cao Y., Song C., Tang Y. Efficient LUT-based FPGA accelerator design for universal quantized CNN inference. *ASSE'21*, 2021, pp. 108–115. DOI: https://doi.org/10.1145/3456126.3456140

[42] Neda N., Ullah S., Ghanbari A., et al. Multi-precision deep neural network acceleration on FPGAs. *27th ASP-DAC*, 2022, pp. 454–459. DOI: https://doi.org/10.1109/asp-dac52403.2022.9712485

[43] Lee S., Kim D., Nguyen D., et al. Double MAC on a DSP: boosting the performance of convolutional neural networks on FPGAs. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2019, vol. 38, no. 5, pp. 888–897. DOI: https://doi.org/10.1109/TCAD.2018.2824280

[44] Ding C. Dynamic precision multiplier for deep neural network accelerators. *IEEE 33rd SOCC*, 2020, pp. 180–184. DOI: https://doi.org/10.1109/socc49529.2020.9524752

[45] Neda N. Multi-precision deep neural network acceleration on FPGAs. *27th ASP-DAC*, 2022, pp. 454–459. DOI: https://doi.org/10.1109/asp-dac52403.2022.9712485

[46] Raees P.C.M., Akshayraj M.R., Gopi Varun P., et al. Dynamic precision scaling in MAC units for energy-efficient computations in deep neural network accelerators. *28th VDAT*, 2024. DOI: https://doi.org/10.1109/VDAT63601.2024.10705697

[47] Sommer J., Özkan A., Keszocze O., et al. DSP-packing: squeezing low-precision arithmetic into FPGA DSP blocks. *32nd FPL*, 2022, pp. 160–166. DOI: https://doi.org/10.1109/FPL57034.2022.00035

[48] Zhang J., Zhang M., Cao X., et al. Uint-packing: multiply your DNN accelerator performance via unsigned integer DSP packing. *60th ACM/IEEE DAC*, 2023. DOI: https://doi.org/10.1109/DAC56929.2023.10247773

[49] Kalali E., van Leuken R. Near-precise parameter approximation for multiple multiplications on a single DSP block. *IEEE Trans. Comput.*, 2022, vol. 71, no. 9, pp. 2036–2047. DOI: https://doi.org/10.1109/TC.2021.3119187

[50] Li R., Jiang B., Xu H. Mixed DSP packing method for convolutional neural network on FPGA. *Proc. SPIE*, 2023, vol. 12800. DOI: https://doi.org/10.1117/12.3004070

[51] Rasoulinezhad S., Zhou H., Wang L., et al. PIR-DSP: an FPGA DSP block architecture for multi-precision deep neural networks. *IEEE 27th FCCM*, 2019, pp. 35–44. DOI: https://doi.org/10.1109/FCCM.2019.00015

[52] Liu Y., Rai S., Ullah S., et al. High-flexibility designs of quantized runtime reconfigurable multi-precision multipliers. *IEEE Embed. Syst. Lett.*, 2023, vol. 15, no. 4, pp. 194–197. DOI: https://doi.org/10.1109/LES.2023.3298736

[53] Liu X., Wu X., Shao H., et al. A flexible FPGA-based accelerator for efficient inference of multi-precision CNNs. *IEEE ISCAS*, 2024.
DOI: https://doi.org/10.1109/ISCAS58744.2024.10557882

[54] Huang M., Liu Y., Huang S., et al. Multi-bit-width CNN accelerator with systolic-in-systolic dataflow and single DSP multiple multiplication scheme. *FPG'23*, 2023, p. 229.
DOI: https://doi.org/10.1145/3543622.3573209

[55] Huang M., Liu Y., Man C., et al. A high performance multi-bit-width booth vector systolic accelerator for NAS optimized deep learning neural networks. *IEEE Trans. Circuits Syst. I: Regul. Pap.*, 2022, vol. 69, no. 9, pp. 3619–3631.
DOI: https://doi.org/10.1109/TCSI.2022.3178474

[56] Zheng Y., Li Z., Sun K., et al. A 40nm area-efficient effective-bit-combination-based DNN accelerator with the reconfigurable multiplier. *IEEE 5th AICAS*, 2023.
DOI: https://doi.org/10.1109/AICAS57966.2023.10168550

[57] Ghavami B., Sajadi M., Shannon L., et al. Boosting multiple multipliers packing on FPGA DSP blocks via truncation and compensation-based approximation. *IEEE ISVLSI*, 2024, pp. 222–227. DOI: https://doi.org/10.1109/ISVLSI61997.2024.00049

[58] Rehman A., Vakili S. A cost-effective FPGA-based approximate multiplier for machine learning acceleration. *IEEE 14th PAAP*, 2023.
DOI: https://doi.org/10.1109/PAAP60200.2023.10391619

[59] Ullah S., Rehman S., Prabakaran B., et al. Area-optimized low-latency approximate multipliers for FPGA-based hardware accelerators. *DAC'18*, 2018, art. 159.
DOI: https://doi.org/10.1145/3195970.3195996

[60] Chen Y., Dotzel J., Abdelfattah M. M4BRAM: mixed-precision matrix-matrix multiplication in FPGA block RAMs. *ICFPT*, 2023, pp. 69–78.
DOI: https://doi.org/10.1109/ICFPT59805.2023.00013

[61] Luo E., Huang H., Liu C., et al. DeepBurning-MixQ: an open source mixed-precision neural network accelerator design framework for FPGAs. *IEEE/ACM ICCAD*, 2023. DOI: https://doi.org/10.1109/ICCAD57390.2023.10323831

[62] Chen Y., Abdelfattah M. BRAMAC: compute-in-BRAM architectures for multiply-accumulate on FPGAs. *31st IEEE FCCM*, 2023, pp. 52–62.
DOI: https://doi.org/10.1109/FCCM57271.2023.00015

[63] Kabir M.A., Kamucheka T., Fredricks N., et al. IMAGine: an in-memory accelerated GEMV engine overlay. *34th FPL*, 2024, pp. 220–226.
DOI: https://doi.org/10.1109/FPL64840.2024.00038

[64] Kabir M.A., Kamucheka T., Fredricks N., et al. The BRAM is the limit: shattering myths, shaping standards, and building scalable PIM accelerators. *32nd IEEE FCCM*, 2024, p. 223. DOI: https://doi.org/10.1109/FCCM60383.2024.00045

Zobov O.V. — Post-Graduate Student, Department of Electronic Equipment Design and Technology, BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Shakhnov V.A. — Corresponding Member of the RAS, Dr. Sc. (Eng.), Professor, Head of the Department of Electronic Equipment Design and Technology, BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Please cite this article in English as:

Zobov O.V., Shakhnov V.A. FPGA-based architectures for deep learning accelerators. *Herald of the Bauman Moscow State Technical University, Series Instrument Engineering*, 2025, no. 4 (153), pp. 78–101 (in Russ.). EDN: KHNNVS