НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
март–апрель 2026 Том 26 № 2 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
March–April 2026 Vol. 26 No 2 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

doi: 10.17586/2226-1494-2026-26-2-357-366
УДК 621.382

Последовательно-параллельная архитектура для реализации на программируемых логических интегральных схемах нейронных сетей, обучаемых в реальном времени по алгоритму обратного распространения ошибки

Инна Владимировна Ушенина✉

Пензенский государственный технологический университет, Пенза, 440039, Российская Федерация
ivl23@yandex.ru✉, https://orcid.org/0000-0001-9674-2604

Аннотация

Введение. На сегодняшний день предложено несколько вычислительных архитектур, реализуемых на программируемых логических интегральных схемах, используемых для обучения нейронных сетей в реальном времени по алгоритму обратного распространения ошибки. Реализуемые архитектуры рассчитаны на нейронные сети небольших размеров, или в них наблюдается значительное снижение максимальной тактовой частоты с ростом размеров обучаемых нейронных сетей. В настоящей работе предложены решения задач обеспечения предсказуемости максимальной тактовой частоты и минимизации ее снижения с увеличением размеров сетей. Представленная архитектура решает эти задачи на уровне организации вычислений. **Метод.** Архитектура представляет собой массив вычислительных блоков, основанных на блоках цифровой обработки сигналов программируемых логических интегральных схем, которые выполняют большую часть вычислений в нейронах параллельно. Архитектура содержит также общий блок, последовательно обрабатывающий результаты вычислений в блоках массива. Получены формулы, показывающие линейную зависимость латентности вычислений от размеров нейронных сетей. **Основные результаты.** По результатам реализации на программируемой логической интегральной схеме отдельного вычислительного блока массива, общего блока и содержащих их нейронных сетей различных размеров оценены полученные временные характеристики. Установлено, что основным фактором, ограничивающим максимальную тактовую частоту нейронных сетей, являются задержки распространения сигналов по шинам, соединяющим массив вычислительных блоков с общим блоком. Максимальная тактовая частота нейронных сетей при 3–240 вычислительных блоках в массиве составляет от 112 до 77 МГц. **Обсуждение.** По сравнению с ближайшим аналогом, в предложенной архитектуре критические пути внутри вычислительных блоков сокращены за счет перевода части вычислений в последовательный режим, но при этом латентность вычисления локальных градиентов нейронов скрытых слоев может оказаться выше. При возрастании количества вычислительных блоков в массиве с 3 до 128 максимальная тактовая частота снижается на 27 % против 52 % у ближайшего аналога. Возрастание количества вычислительных блоков со 120 до 240 в представленной архитектуре снижает тактовую частоту не более чем на 5 %. Нейронные сети с разработанной архитектурой, реализованные на программируемых логических интегральных схемах, могут использоваться для решения задач, требующих обучения в реальном времени — идентификации систем и отслеживания объектов.

Ключевые слова

архитектура нейронной сети, обучение, ПЛИС, алгоритм обратного распространения ошибки, режим реального времени

Благодарности

Исследование выполнено за счет гранта Российского научного фонда и Пензенской области № 24-21-20100, https://rscf.ru/project/24-21-20100/.

Ссылка для цитирования: Ушенина И.В. Последовательно-параллельная архитектура для реализации на программируемых логических интегральных схемах нейронных сетей, обучаемых в реальном времени по алгоритму обратного распространения ошибки // Научно-технический вестник информационных технологий, механики и оптики. 2026. Т. 26, № 2. С. 357–366. doi: 10.17586/2226-1494-2026-26-2-357-366

Series-parallel architecture for the FPGA implementation of neural networks trainable in real-time using the error backpropagation algorithm

Inna V. Ushenina✉

Penza State Technological University, Penza, 440039, Russian Federation

ivl23@yandex.ru✉, https://orcid.org/0000-0001-9674-2604

Abstract

To date, several Field-Programmed Gate Array (FPGA) implementable computational architectures have been proposed that can be used for neural network training in real-time by the backpropagation algorithm. However, they are intended for small neural networks or have a significant reduction in maximum clock frequency as network sizes increase. The novelty of this work lies in addressing the problems of ensuring a predictable maximum clock frequency and minimizing its degradation when scaling the computational architecture. The proposed architecture solves these problems at the level of computational organization. The architecture comprises an array of computational blocks which are based on FPGA digital signal processing blocks and perform most computations in parallel. The architecture also contains the shared block that sequentially processes the computation results received from the array blocks. The equations were derived showing that the latency of computations increases linearly with neural network sizes. After a computational block instance, the shared block and neural networks containing various numbers of computational blocks had been implemented on the FPGA, their timing characteristics were assessed. It has been determined that the data path delays of the buses connecting the shared block with the array blocks are the primary factors constraining the maximum clock frequencies of neural networks. When the number of the array blocks lies in the range 3–240, the maximum clock frequency is from 112 down to 77 MHz. Compared to the closest counterpart, the critical paths in the proposed architecture are shortened because some computations are transferred to the sequential mode; however, this transfer may increase the latency of calculating the local gradients of the hidden layers neurons. When the number of the array computational blocks grows from 3 to 128, the maximum clock frequency decreases by 27 % compared to 52 % for the closest counterpart. Growing the number of computational blocks in the proposed architecture from 128 to 240 reduces the maximum clock frequency by no more than 5 %. FPGA-based neural networks of the proposed architecture are suitable for object tracking and system identification, which are typical applications of neural networks trained in real-time mode.

Keywords

neural network architecture, training, FPGA, error backpropagation algorithm, real-time mode

Acknowledgements

This work was supported by the Russian Science Foundation and Penza region under Grant № 24-21-20100, https://rscf.ru/project/24-21-20100/.

For citation: Ushenina I.V. Series-parallel architecture for the FPGA implementation of neural networks trainable in real-time using the error backpropagation algorithm. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2026, vol. 26, no. 2, pp. 357–366 (in Russian). doi: 10.17586/2226-1494-2026-26-2-357-366

Введение

Аппаратная реализация нейронных сетей и их обучение в реальном времени требуются, когда необходима адаптация к быстро меняющимся условиям, в частности, в задачах идентификации систем [1–3], отслеживания объектов [4–7] и др. В качестве нейронной сети или ее части может использоваться многослойный персептрон, обучающийся по алгоритму обратного распространения ошибки [1, 2, 5–7].

Обучение нейронных сетей может выполняться на различных аппаратных платформах — как специализированных [4, 5, 8], так и стандартных: графических процессорах [8], микроконтроллерах [9] или программируемых логических интегральных схемах (ПЛИС) [7–13]. ПЛИС сочетают в себе доступность, возможность параллельных вычислений и возможность управления архитектурой реализуемых устройств за счет программирования соединений между ресурсами.

Несмотря на большое количество ресурсов у ПЛИС, поддержка всех возможных уровней параллелизма вычислений [10] оказывается доступной только для нейронных сетей небольших размеров. В работах [11, 12] выделение отдельных ресурсов ПЛИС на все виды вычислений и все слои нейронных сетей позволило реализовать многослойные персептроны, обучающиеся в реальном времени по алгоритму обратного распространения ошибки, с размерами соответственно 32 × 32 × 32 и 7 × 7 × 4 (в обозначениях размеров нейронных сетей цифры слева направо соответствуют количеству нейронов в слоях сети от входного слоя к выходному). С увеличением размеров нейронных сетей приходится отказываться от некоторых уровней параллелизма вычислений и разделять ресурсы. В [9], где на ПЛИС реализованы многослойные персептроны с размерами до 60 × 15 × 10 × 5, на каждый слой и каждый нейрон выделены отдельные ресурсы, но они используются на этапах и прямого, и обратного распространений.

В [13] на ПЛИС реализованы нейронные сети с размерами до 784 × 128 × 64 × 10. При этом сохранен параллелизм только на уровне нейронов, когда один и тот же массив вычислительных блоков используется всеми слоями сети при прямом и обратном распространениях. Предложенная в [13] архитектура нейронных сетей предназначена не для их обучения в реальном времени, а для широкого применения в составе встраиваемых систем. Отдельные решения, принятые в [13], приводят к выраженной зависимости временных характеристик сетей от их размеров, а также к длинным критическим путям в вычислительных блоках.

Новизна настоящей работы заключается в том, что в ней ставятся задачи обеспечения предсказуемости максимальной тактовой частоты архитектуры нейронных сетей и минимизации ее снижения с увеличением количества входных сигналов и нейронов в слоях, от единиц до нескольких сотен. Для архитектуры в виде массива вычислительных блоков решить эти задачи можно, если минимизировать критические пути внутри вычислительных блоков и устранить зависимость их длины от размеров сети, а затем минимизировать снижение максимальной тактовой частоты архитектуры при ее масштабировании. Первым шагом к решению этих задач является организация вычислений, т. е. разработка вычислительной архитектуры, а последующими шагами — ее размещение и конвейеризация с учетом особенностей логических и трассировочных ресурсов ПЛИС.

В работе предложена и реализована на ПЛИС архитектура обучаемых в реальном времени по алгоритму обратного распространения ошибки нейронных сетей типа «многослойный персептрон» с настраиваемым количеством входных сигналов, слоев и нейронов в каждом слое, которая решает поставленные задачи на уровне организации вычислений. Получены формулы расчета латентности вычислений для нейронных сетей заданных размеров. После реализации на ПЛИС нейронных сетей различных размеров оценены их временные характеристики: максимальная тактовая частота и задержки распространения сигналов по критическим путям. Выполнено сравнение предложенной архитектуры с ближайшим аналогом [13].

Предлагаемая архитектура

Вычисления в нейронной сети. Каждая итерация алгоритма обучения нейронной сети состоит из прямого распространения сигналов от первого слоя к последнему, позволяющего вычислить выходные сигналы сети и сигналы ошибок, и обратного распространения ошибок от последнего слоя к первому, дающего возможность обновить синаптические коэффициенты нейронов [1].

Прямое распространение сигналов описывается следующим образом:

$$v_k^i(n) = b_k^i(n) + \sum_{j=1}^{K_{i-1}} w_{kj}^i(n) x_{kj}^i(n), \quad (1)$$

$$\varphi_k^i(n) = y_k^i(n) = (1 + \exp(-v_k^i(n)))^{-1}, \quad (2)$$

$$e_k^L(n) = d_k^L(n) - y_k^L(n), \quad (3)$$

где $v_k^i(n)$ — сумма взвешенных входных сигналов нейрона; $b_k^i(n)$ и $w_{kj}^i(n)$ — смещение и синаптический коэффициент нейрона; $x_{kj}^i(n)$ — входной сигнал нейрона; n — номер шага дискретизации (итерации алгоритма); i, k и j — номера слоя, нейрона в слое и входного сигнала нейрона; K_i, K_{i-1} — количество нейронов в i-м или $(i-1)$-м слое, а при $(i-1)=0$ K_0 — количество входных сигналов нейронной сети; $\varphi_k^i(n)$ — функция активации нейрона; $y_k^i(n)$ — выходной сигнал нейрона; $d_k^L(n)$, $y_k^L(n)$ и $e_k^L(n)$ — требуемый сигнал нейрона выходного слоя, его фактический выходной сигнал и сигнал ошибки; L — количество слоев сети, не считая входного.

К обратному распространению сигналов относятся следующие вычисления:

$$\varphi_k'^i(n) = \varphi_k^i(n)(1 - \varphi_k^i(n)), \quad (4)$$

$$\delta_k^L(n) = e_k^L(n)\varphi_k'^L(n), \quad (5)$$

$$\delta_k^{i-1}(n) = \varphi_k'^{i-1}(n) \sum_{k=1}^{K_i} \delta_k^i(n) w_{kj}^i(n), \quad (6)$$

$$w_{kj}^i(n+1) = w_{kj}^i(n) + \eta x_{kj}^i(n) \delta_k^i(n), \quad (7)$$

где $\varphi_k'^i(n)$ и $\delta_k^i(n)$ — соответственно производная функции активации и локальный градиент нейрона в i-м слое сети при $i \in [1; L]$; η — коэффициент, задающий скорость обучения. Так как $w_{kj}^i(n)$ масштабируют выходные сигналы нейронов $(i-1)$-го слоя, поступающие на нейроны i-го слоя, в (6) индекс k при $\delta_k^{i-1}(n)$ и $\varphi_k'^{i-1}(n)$ соответствует индексу j при $w_{kj}^i(n)$.

Архитектура нейронной сети и порядок вычислений. Предлагаемая архитектура поддерживает параллелизм вычислений на уровне нейронов, т. е. резервирует отдельные вычислительные ресурсы для каждого нейрона в пределах слоя нейронной сети. При этом одни и те же ресурсы обслуживают по очереди все слои сети как при прямом, так и при обратном распространениях. Такое решение оправдано ограниченным количеством ресурсов ПЛИС, а также тем, что $\delta_k^{i-1}(n)$ не могут быть вычислены, пока не рассчитаны $\delta_k^i(n)$ в формулах (5) и (6).

Архитектура состоит из массива вычислительных блоков (ВБ) (рис. 1) и общего блока (ОБ), используемого всеми ВБ (рис. 2). Количество K ВБ в массиве определяется слоем нейронной сети с максимальным количеством нейронов. Основу ВБ составляют: предварительный сумматор, принимающий входные сигналы A_k и D_k соответственно от мультиплексора A_k и регистра δRg_k; умножитель, принимающий входные сигналы от предварительного сумматора и мультиплексора B_k, и основной сумматор, который совместно с регистром aRg_k может работать в режиме аккумулятора.

Основной сумматор ВБ принимает результат умножения и сигналы a_k от aRg_k; $CASI_k$ от предыдущего ВБ; C_k, равный $w_{kj}^i(n)$ или $b_k^i(n)$, от хранящего их оперативного запоминающего устройства (ОЗУ) RAM_k. Сумматоры и умножитель могут работать в разных режимах, синхронизируя и обрабатывая конкретную комбинацию входных сигналов и отключая остальные. Каждый ВБ содержит также L регистров δRg_k, хранящих локальные градиенты обслуживаемых им нейронов; блоки V_k и W_k для ограничения разрядности соответственно $v_k^i(n)$ и $w_{kj}^i(n+1)$; регистры vRg_k.

При прямом распространении сигнала каждый ВБ выполняет вычисления согласно (1), принимая последовательность $x_{kj}^i(n)$ от мультиплексора X ОБ: если $i=1$, $x_{kj}^1(n)$ поступают извне; если $2 \le i \le L$, в качестве $x_{kj}^i(n)$ выступают $y_k^{i-1}(n)$. Полученный массивом ВБ набор $v_k^i(n)$ поступает от аккумуляторов через блоки V_k и цепочку регистров vRg_k на ОБ, где блоком φ, согласно (2), вычисляется последовательность значений $\varphi_k^i(n) = y_k^i(n)$, далее по очереди поступающих в качестве $x_{kj}^{i+1}(n)$ на

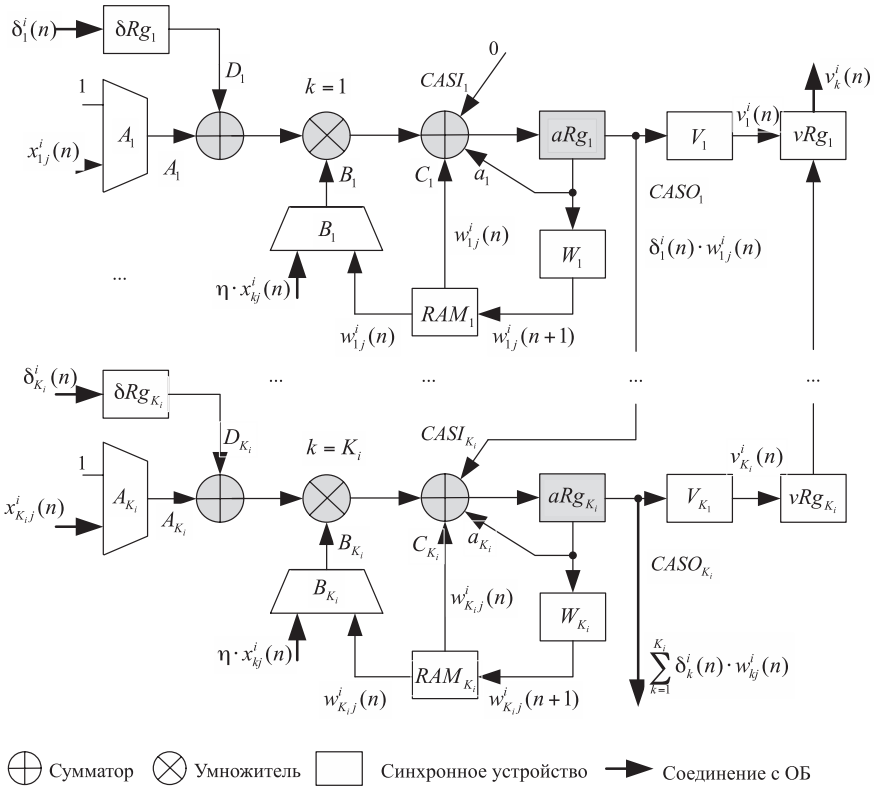

Рис. 1. Массив вычислительных блоков $K = K_i$.

Ресурсы блоков цифровой обработки сигналов программируемой логической интегральной схемы выделены серым цветом. $CASl_k$, $CASO_k$ — линии, объединяющие вычислительные блоки в каскад; C_k — выходной сигнал блока памяти RAM_k; V_k, W_k — блоки ограничения разрядности результатов вычислений; 0, 1 — константы, представленные в используемом формате вычислений

Fig. 1. Computational blocks array, $K = K_i$.

The resources of the FPGA signal processing blocks are colored gray. The notation used in the figure is as follows: $CASI_k$, $CASO_k$ are the lines joining the computational blocks into the cascade; C_k is the output signal of the RAM_k block; V_k, W_k are the blocks restricting the bit widths of the computation results; 0, 1 are the constants represented in the computation format used

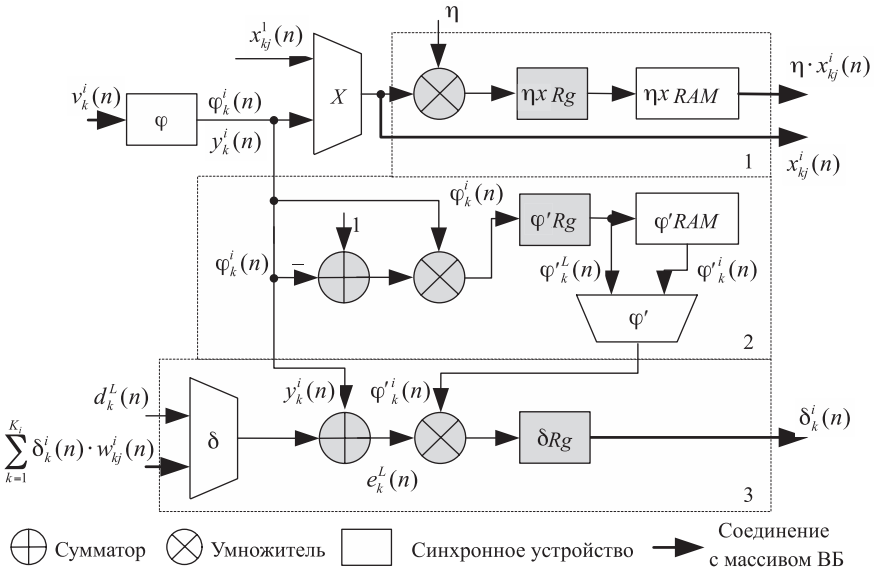

Рис. 2. Общий блок. Ресурсы блоков цифровой обработки сигналов программируемой логической интегральной схемы выделены серым цветом.

φ — блок вычисления функции активации; *X* — мультиплексор, формирующий входные сигналы для массива вычислительных блоков; 1–3 — блоки, входящие в состав общего блока

Fig. 2. Shared block. The resources of the FPGA signal processing blocks are colored gray.

The notation used in the figure is as follows: φ is the activation computation block; *X* is the multiplexer generating the input signals for the computational blocks array; 1–3 are the blocks that are included in the shared block

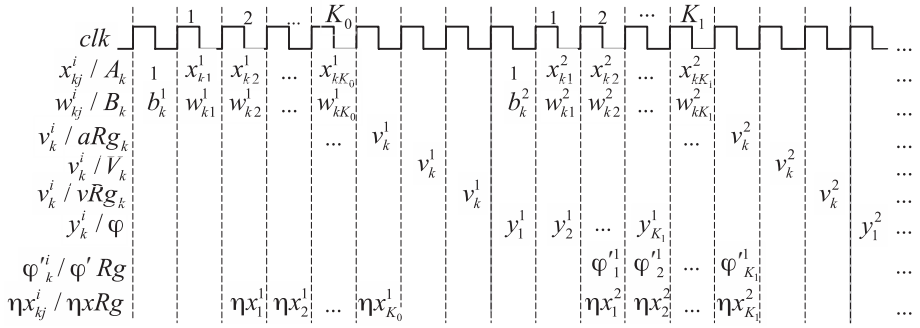

Рис. 3. Временная диаграмма работы общего блока и одного вычислительного блока массива при прямом распространении. Обработка сигналов первого и второго слоев нейронной сети

Fig. 3. Timing diagram of the shared block and one computational block of the array in the forward pass stage. Processing the signals of the first and second network layers

мультиплексоры A_k ВБ. При добавлении $b_k^i(n)$ к $v_k^i(n)$ $A_k = 1$. Работа одного ВБ массива и ОБ при прямом распространении представлена на рис. 3 и 4. На рис. 3–6 индексы n при обозначениях сигналов опущены.

При обратном распространении ВБ работают в двух режимах. Во-первых, ВБ, объединенные в цепочку шинами *CASI-CASO* (рис. 1), совместно участвуют в вычислении локальных градиентов нейронов скрытых слоев (рис. 5). Получаемые на выходе последнего ВБ в цепочке суммы $\sum_{k=1}^{K_i} \delta_k^i(n) w_{kj}^i(n)$ поочередно поступают в ОБ, где умножаются на значения $\varphi_k'^{i-1}(n)$ соответствующих нейронов (6), а полученные $\delta_k^{i-1}(n)$ отправляются на хранение в регистры δRg_k ВБ (рис. 1 и 2).

Во-вторых, ВБ используются для расчетов $w_{kj}^i(n+1)$ согласно (7), как показано на рис. 6. При этом ВБ работают параллельно и обращаются к ОБ для чтения $\eta x_{kj}^i(n)$.

В ОБ (рис. 2) вынесены операции, последовательное выполнение которых или снижает количество мультиплексируемых входов сумматоров и умножителей ВБ и тем самым оптимизирует критические пути, или экономит ресурсы, и не повышает при этом латентность вычислений. Так, ОБ содержит ресурсоемкий блок φ: $x_{kj}^i(n)$ поступают на массив ВБ по очереди, и последовательное вычисление $\varphi_k^i(n)$ не повышает латентность.

В состав ОБ входят еще три блока (рис. 2). Блок 1 содержит умножитель для масштабирования $x_{kj}^{i+1}(n)$ с коэффициентом η. Результаты масштабирования, проходя через регистр ηxRg, сохраняются в блок памяти $\eta xRAM$. При обратном распространении $\eta x_{kj}^i(n)$ считываются из $\eta xRAM$ и поступают на ВБ массива для расчетов по (7). Блок 2 ОБ, содержащий умножитель, предварительный сумматор, регистр $\varphi' Rg$, блок памяти $\varphi' RAM$ и мультиплексор φ′, вычисляет (4) и сохраняет $\varphi_k'^i(n)$. На рис. 3–6 показано, что расчеты $\varphi_k'^i(n)$ и $\eta x_{kj}^i(n)$, выполняемые при прямом распространении, не повышают латентность вычислений.

Блок 3 ОБ (рис. 2), предназначенный для расчета $\delta_k^i(n)$, содержит мультиплексор δ, предварительный сумматор, умножитель и регистр δRg. $\delta_k^L(n)$ начинают вычисляться в ОБ по (5), как только готовы $y_1^L(n)$ и $\varphi_k'^L(n)$ (рис. 4). При этом на входы предварительного сумматора поступают $y_k^L(n)$ и, через мультиплексор δ, $d_k^L(n)$. На умножитель, вычисляющий $\delta_k^L(n)$, попадают $e_k^L(n)$, вычисленные предварительным сумматором по (3), и $\varphi_k'^L(n)$ от регистра $\varphi' Rg$.

При прямом распространении вычисления в скрытых слоях (рис. 3 и 4) разделяются на циклы по $K_i + 4$ такта: от момента поступления $b_k^i(n)$ на умножители ВБ до появления $v_k^i(n)$ на выходах vRg_k. Во время вычисления $y_k^L(n)$, $\varphi_k'^L(n)$ и $\delta_k^L(n)$ этапы прямого и обратного

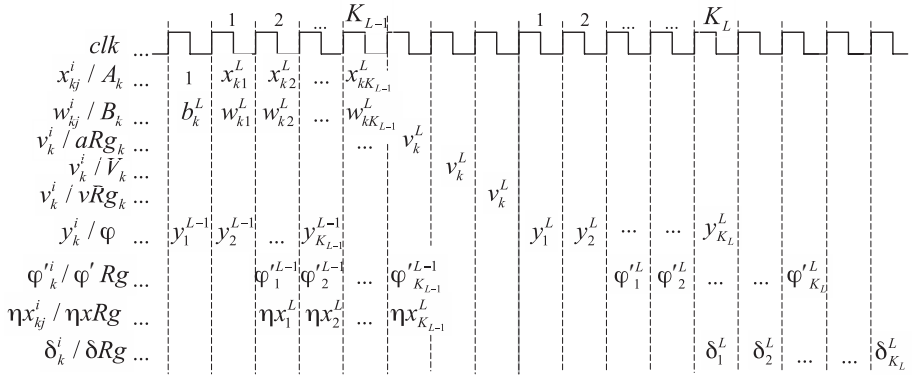

Рис. 4. Временная диаграмма работы общего блока и одного вычислительного блока массива при прямом распространении. Обработка сигналов предпоследнего и последнего слоев нейронной сети

Fig. 4. Timing diagram of the shared block and one computational block of the array in the forward pass stage. Processing the signals of the second last and last network layers

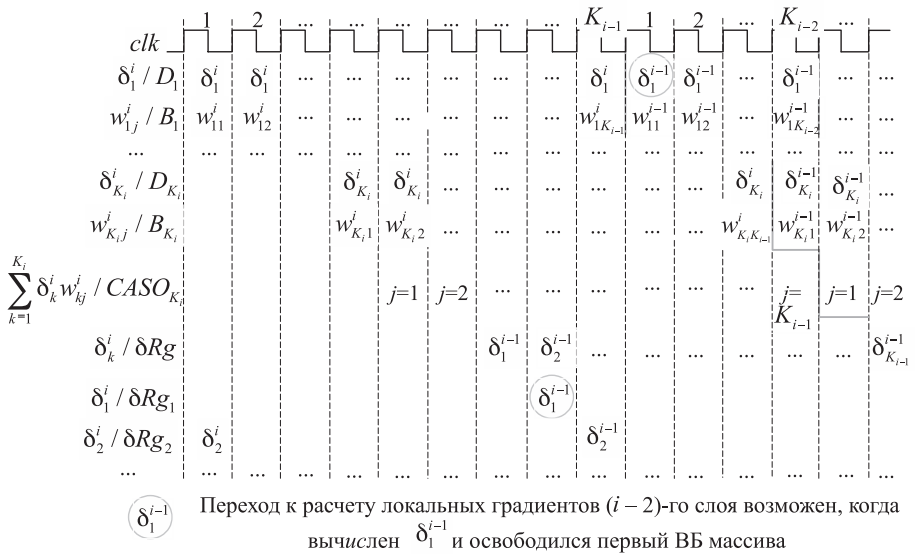

Рис. 5. Временная диаграмма работы общего блока и массива вычислительных блоков при обратном распространении. Вычисление локальных градиентов нейронов скрытых слоев

Fig. 5. Timing diagram of the shared block and computational blocks array in the backward pass stage. Computation of the local gradients of hidden layers neurons

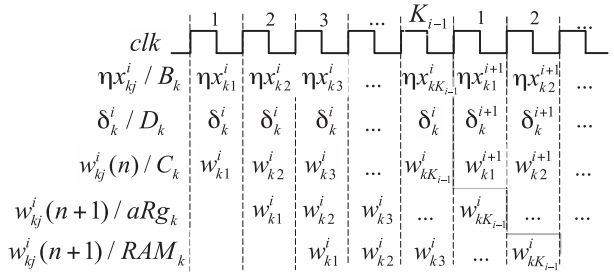

Рис. 6. Временная диаграмма работы одного вычислительного блока массива при обратном распространении. Обновление синаптических коэффициентов нейронов

Fig. 6. Timing diagram of a computational block of the array in the backward pass stage. Updating the synaptic coefficients of neurons

распространений частично перекрываются (рис. 4 и 5). Цикл вычисления $y_k^L(n)$, $\varphi_k'^L(n)$ и $\delta_k^L(n)$ в ОБ занимает $K_L + 4$ такта (рис. 4), но через 5 тактов от начала цикла $\delta_1^L(n)$ сохраняется в регистр δRg_1 первого ВБ в массиве, и начинаются вычисления $\delta_k^{L-1}(n)$ согласно (6) (рис. 5). Если отнести первые четыре такта этого цикла к прямому распространению, латентность последнего составит $4(L + 1) + \sum_{i=0}^{L-1} K_i$.

При обратном распространении в первом ВБ $\delta_1^L(n)$ поочередно умножается на K_{L-1} значений $w_{1j}^L(n)$, а полученные произведения по шине $CASO_1$ передаются на сумматор второго ВБ, где на такт позже начинаются аналогичные вычисления, и т. д. (рис. 5).

Расчет суммы $\sum_{k=1}^{K_L} \delta_k^L(n) w_{k1}^L(n)$ выполняется K_L ВБ массива за $K_L + 1$ тактов, после чего она поступает в ОБ для расчета $\delta_1^{L-1}(n)$. Расчет $\delta_1^{L-1}(n)$ и его пересылка в регистр δRg_1 первого ВБ занимает еще три такта. Вычисление полного набора $\delta_k^{L-1}(n)$ займет $K_L + K_{L-1} + 4$ тактов с начала этапа обратного распространения.

Если $L = 2$, спустя $K_L + K_{L-1} + 4$ тактов, требуемых для вычисления набора $\delta_k^1(n)$, будут вычисляться синаптические коэффициенты. Если $L > 2$, вычисление набора $\delta_k^{L-2}(n)$ может быть начато, когда $\delta_1^{L-1}(n)$ вычислен и записан в δRg_1, а первый ВБ массива освободился. На рис. 5 видно, что при $K_{i-1} \geq K_i + 4$ вычисление набора $\delta_k^{i-2}(n)$ начинается спустя K_{i-1} тактов с начала вычисления набора $\delta_k^{i-1}(n)$; при $K_{i-1} < K_i + 4$ — спустя $K_i + 4$ такта. Таким образом, если $L > 2$, вычисление набора $\delta_k^1(n)$ займет $K_1 + K_2 + 4$ тактов, а наборов $\delta_k^{i-1}(n)$ для нейронов остальных скрытых слоев — $\sum_{i=3}^{L} \max(K_{i-1}, K_i + 4)$ тактов.

После вычисления всех $\delta_k^i(n)$ обновляются $w_{kj}^i(n)$ (рис. 6). На пересчет синаптических коэффициентов нейронов i-го слоя требуется $K_{i-1} + 1$ тактов, и еще такт необходим для записи новых значений в RAM_k. Обновление всех $w_{kj}^i(n)$ займет $2 + \sum_{i=0}^{L-1} K_i$ тактов. Итого,

латентность T нейронной сети с $L \geq 3$, K_0 входными сигналами и $K_1 \ldots K_L$ нейронами в слоях определяется согласно:

$$T = 4L + 10 + K_1 + K_2 + 2\sum_{i=0}^{L-1} K_i + \\ + \sum_{i=3}^{L} \max(K_{i-1}, K_i + 4), \quad (8)$$

а в частном случае нейронной сети с одним скрытым слоем, т. е. при $L = 2$ по формуле:

$$T = 18 + 2K_0 + 3K_1 + K_2. \quad (9)$$

Результаты реализации нейронных сетей на ПЛИС

Временные характеристики нейронных сетей различных размеров определены после их реализации на ПЛИС Arora V и выполнения временного анализа результатов реализации в среде проектирования Gowin FPGA Designer 1.9[1]. Для описания нейронных сетей использован язык VHDL.

Переменные, обрабатываемые нейронными сетями, представлены в формате с фиксированной запятой. Разрядности переменных выбраны в соответствии с анализом, проведенным в [14]: входные, выходные, требуемые сигналы, аргументы функций активации представлены 10 разрядами; синаптические коэффициенты и локальные градиенты нейронов — 16 разрядами.

В качестве функции активации нейронов использован сигмоид (2), вычисление которого выполнялось методом поразрядного отображения. Этот метод и схемы вычислителей функции активации описаны в [15].

Сумматоры, аккумуляторы и умножители ВБ и ОБ реализованы на блоках цифровой обработки сигналов (ЦОС) ПЛИС; значения $w_{kj}^{i}(n)$, $\eta x_{kj}^{i}(n)$ и $\varphi_k'^{i}(n)$ хранятся в блоках ОЗУ ПЛИС (рис. 1 и 2). Остальные устройства ВБ и ОБ реализованы на программируемой логике ПЛИС — табличных преобразователях (Look-Up Table, LUT) и регистрах. В выбранной ПЛИС доступно 298 блоков ЦОС, 340 блоков ОЗУ емкостью 18 кбит и по 138 240 LUT и регистров. Результаты оценки предельных размеров нейронных сетей с предложенной архитектурой, реализуемых на выбранной ПЛИС, приведены в табл. 1. Количество ВБ в массиве (K) ограничивается запасом блоков ЦОС, а суммарное количество нейронов в скрытых слоях сети и входных сигналов, $\sum_{i=0}^{L-1} K_i$ — объемом памяти блоков ОЗУ, хранящих синаптические коэффициенты.

Временной анализ показал, что в отдельном ВБ критический путь проходит через мультиплексор B_k, а максимальная тактовая частота $F_{\max}$ ВБ равна 169 МГц. Внутри ОБ критический путь проходит через блок φ, ограничивая $F_{\max}$ на уровне 113 МГц. Таким образом, без дополнительной конвейеризации ОБ $F_{\max}$ нейронных сетей будет составлять не более 113 МГц, и с этим значением следует сравнивать $F_{\max}$ нейронных сетей при различных размерах массива ВБ.

Временные характеристики схем нейронных сетей при различных K приведены в табл. 2. Показано, что критические пути, ограничивающие $F_{\max}$, соответствуют шинам, соединяющим ОБ и массив ВБ.

Обсуждение

Сравним предложенную архитектуру с ближайшим аналогом [13] в контексте поставленных задач для архитектуры нейронных сетей, обучающихся в реальном времени. В [13] операции, выполняемые массивом ВБ, распараллелены максимально. Но при этом ВБ получают входные переменные через каскады многовходовых мультиплексоров, удлиняющие критические пути и усложняющие их минимизацию. В представленной архитектуре часть вычислений перенесена в ОБ, выполняющий их последовательно, за счет чего на сумматоры и умножители ВБ переменные подаются напрямую или через один двухвходовой мультиплексор. Исключение составляют каскады, образуемые мультиплексором X ОБ с мультиплексорами A_k ВБ. Но сокращение критических путей через них облегчается тем, что: мультиплексор X и блок φ расположены рядом; на один из входов мультиплексора A_k подается константа, формируемая рядом с этим входом.

Перевод части вычислений в последовательный режим частично компенсируется тем, что $\varphi_k'^{i}(n)$ и $\eta x_{kj}^{i}(n)$ рассчитываются в ОБ на этапе прямого распространения, не повышая латентность. Также, сумматоры ВБ объединены в цепочку, давая возможность рассчитать локальные градиенты нейронов всех скрытых слоев, кроме первого, за K_{i-1} или $K_i + 4$ тактов. Это сопоставимо с [13], где те же вычисления занимают $K_i + 8$ тактов. Однако, в предложенной архитектуре дольше

[1] Gowin Semiconductor. Официальная документация [Электронный ресурс]. Режим доступа: https://www.gowinsemi.com/en/support/, свободный. Яз. англ. (дата обращения: 10.08.2025).

Таблица 1. Ресурсоемкость и ограничения размеров нейронных сетей

Table 1. Resource capacity and size limitations of neural networks

Ресурс	В общем блоке, шт.	В вычислительном блоке, шт.	Ограничение
Блоки ЦОС	3	1	$K \leq 295$
Блоки ОЗУ	2	1	$\sum_{i=0}^{L-1} K_i \leq 1152$
LUT	849	87	$K \leq 1579$
Регистры	30	101	$K \leq 1368$

Таблица 2. Временные характеристики схем нейронных сетей при различных размерах массива ВБ
Table 2. Timing characteristics of neural network circuits under various neural network sizes

Характеристики схем нейронных сетей		Количество блоков в массиве, шт.						
		3	20	40	60	120	180	240
F_{max}, МГц		112	101	92	91	81	78	77
t_d, нс	$x_{kj}^i(n)$	4,769	8,939	10,136	10,842	12,234	12,376	12,909
	$\eta x_{kj}^i(n)$	4,841	6,934	10,875	10,943	12,231	12,399	12,905
	$\delta_k^i(n)$	3,678	6,772	10,286	10,933	12,354	12,886	12,237
	$v_k^i(n)$	8,928	9,834	10,888	10,947	12,425	12,552	13,032
	$\sum_{k=1}^{K_L} \delta_k^L(n) w_{kj}^L(n)$	4,773	5,471	9,418	10,854	12,124	12,361	12,473

Примечание: t_d — задержки распространения сигналов по шинам; $x_{kj}^i(n)$, $\eta x_{kj}^i(n)$, $\delta_k^i(n)$, $v_k^i(n)$, $\sum_{k=1}^{K_L} \delta_k^L(n) w_{kj}^L(n)$ — обозначения шин (рис. 1 и 2).

вычисляются локальные градиенты нейронов первого скрытого слоя, а в [13] — нейронов последнего слоя.

В [13] для коммутирования выходов K ВБ на вход разделяемого ими вычислителя функции активации используется мультиплексор. При этом с возрастанием K увеличивается объем комбинационной логики и длина соединений между массивом ВБ и вычислителем функции активации. Вместо этого в предложенной архитектуре из K выходных сигналов ВБ регистрами vRg_k формируется очередь для подачи на блок φ. Последний регистр цепочки сдвига соединяется с вычислителем функции активации напрямую. В итоге повышение K с 3 до 128 снижает F_{max} на 31 МГц (табл. 2), но это снижение связано с увеличением расстояния между ОБ и ВБ на кристалле.

В [13] реализация нейронных сетей выполнялась на нескольких более скоростных ПЛИС по сравнению с выбранной в настоящей работе, и для $K \le 50$ получены более высокие F_{max} — от 218 до 114 МГц. Однако с повышением K с 3 до 128 F_{max} в [13] снижается как минимум на 115 МГц, составив 103 МГц для нейронной сети с размерами 784 × 128 × 64 × 10.

В табл. 3 приведены результаты сравнения предложенной архитектуры с архитектурой [13] по критериям латентности T и требуемого количества ЦОС-блоков, которое составляет для [13] $K + 2$, а в разработанной архитектуре — $K + 3$. Для каждого значения T в скобках уточняется, сколько тактов требуют вычисления на этапах прямого распространения, вычисления локальных градиентов нейронов и обновления синаптических коэффициентов соответственно. Из табл. 2 и табл. 3 можно заключить, что при латентности и ресурсоемкости, сопоставимых с полученными в [13], предложенная архитектура позволяет остановить снижение F_{max} с ростом K.

Сопоставляя результаты, представленные в (8), (9) и в табл. 2, с условиями, типичными для идентификации систем [2, 3] и отслеживания объектов [4, 7], можно заключить, что в обоих случаях итерация алгоритма обучения будет занимать незначительную часть периода дискретизации и оставлять запас времени для других необходимых операций.

Заключение

Таблица 3. Сравнение предложенной архитектуры с архитектурой в [13]
Table 3. Comparison of the proposed architecture and the architecture in [13]

Размеры сети ($K_0 \times K_1 \times \ldots \times K_L$), нейронов в слоях	Архитектура [13]		Предложенная архитектура	
	Количество тактов, шт.	Количество блоков ЦОС, шт.	Количество тактов, шт.	Количество блоков ЦОС, шт.
10 × 3 × 1	59 (28 + 11 + 20)	5	48 (25 + 8 + 15)	6
10 × 50 × 1	192 (75 + 11 + 106)	52	189 (72 + 55 + 62)	53
50 × 10 × 10 × 5	207 (95 + 33 + 79)	12	192 (86 + 34 + 72)	13
60 × 15 × 10 × 5	237 (110 + 33 + 94)	17	227 (101 + 39 + 87)	18
784 × 128 × 64 × 10	2080 (1001 + 94 + 985)	130	2230 (992 + 260 + 978)	131
912 × 240 × 240	2802 (1167 + 482 + 1153)	242	2802 (1164 + 484 + 1154)	243

Архитектура, разработанная для решения поставленных в работе задач, позволила минимизировать критические пути внутри вычислительных блоков и зависимость максимальной тактовой частоты реализуемых схем нейронных сетей от количества вычислительных блоков в массиве. С возрастанием размеров схем нейронных сетей происходит снижение на 27 %, а затем стабилизация их максимальной тактовой частоты. Для схем нейронных сетей с предложенной архитектурой имеется возможность повысить максимальную тактовую частоту отдельных блоков и сократить ее снижение для нейронных сетей в целом за счет оптимизации размещения блоков на программируемой логической интегральной схеме и конвейеризации на критических путях.

Литература

1. Хайкин С. Нейронные сети: полный курс. СПб.: Диалектика, 2020. 1104 с.
2. Zhao G., Zhang P., Ma G., Xiao W. System identification of the nonlinear residual errors of an industrial robot using massive measurements // Robotics and Computer-Integrated Manufacturing. 2019. V. 59. P. 104–114. https://doi.org/10.1016/j.rcim.2019.03.007
3. Nelles O. Nonlinear System Identification: from Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes. Springer, 2020. 1253 p. https://doi.org/10.1007/978-3-030-47439-3
4. Han D., Yoo H.-J. On-Chip Training NPU-Algorithm, Architecture and SoC Design. Springer, 2023. 260 p. https://doi.org/10.1007/978-3-031-34237-0
5. Han D., Lee J., Lee J., Choi S., Yoo H.-J. A 141.4 mW low-power online deep neural network training processor for real-time object tracking in mobile devices // Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS). 2018. P. 1–5. https://doi.org/10.1109/iscas.2018.8351398
6. Varagula J., Kulproma P., Itob T. Object detection method in traffic by on-board computer vision with time delay neural network // Procedia Computer Science. 2017. V. 112. P. 127–136. https://doi.org/10.1016/j.procs.2017.08.185
7. Heidaryan M., Karimi G. FPGA implementation of two multilayer perceptron neural network in cascade for efficient real time hand gestures tracking // Microprocessors and Microsystems. 2023. V. 100. P. 104849. https://doi.org/10.1016/j.micpro.2023.104849
8. Seng K.P., Ang L.M. Embedded intelligence: State-of-the-art and research challenges // IEEE Access. 2022. V. 10. P. 59236–59258. https://doi.org/10.1109/access.2022.3175574
9. Ortega-Zamorano F., Jerez J.M., Munoz D.U., Luque-Baena R.M., Franco L. Efficient implementation of the backpropagation algorithm in FPGAs and microcontrollers // IEEE Transactions on Neural Networks and Learning Systems. 2015. V. 27. N 9. P. 1840–1850. https://doi.org/10.1109/tnnls.2015.2460991
10. Ушенина И.В., Данилов Е.А. Реализация на ПЛИС модуля искусственного нейрона для последовательно-параллельных архитектур нейронных сетей с прямой связью // Цифровая обработка сигналов. 2025. № 1. С. 78–84.
11. Zhang M.J., Garcia S., Terre M. Real-time fast learning hardware implementation // International Journal for Simulation and Multidisciplinary Design Optimization. 2023. V. 14. P. 1. https://doi.org/10.1051/smdo/2023001
12. Tisan A., Chin J. An end-user platform for FPGA-based design and rapid prototyping of feedforward artificial neural networks with on-chip backpropagation learning // IEEE Transactions on Industrial Informatics. 2016. V. 12. N 3. P. 1124–1133. https://doi.org/10.1109/tii.2016.2555936
13. Zhang Z., Wang G., Wang K., Gan B., Chen G. Efficient on-chip learning of multi-layer perceptron based on neuron multiplexing method // Electronics. 2023. V. 12. N 17. P. 3607. https://doi.org/10.3390/electronics12173607
14. Holt J.L., Hwang J.-N. Finite precision error analysis of neural network hardware implementations // IEEE Transactions on Computers. 1993. V. 42. N 3. P. 281–290. https://doi.org/10.1109/12.210171
15. Ушенина И.В. Реализация на современных ПЛИС вычислителя сигмоидной функции активации нейронных сетей табличным методом // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. 2024. № 69. С. 124–133. https://doi.org/10.17223/19988605/69/13

References

1. Haykin S. *Neural networks: A Comprehensive Foundation*. Prentice Hall, 1998. 842 p.
2. Zhao G., Zhang P., Ma G., Xiao W. System identification of the nonlinear residual errors of an industrial robot using massive measurements. *Robotics and Computer-Integrated Manufacturing*, 2019, vol. 59, pp. 104–114. https://doi.org/10.1016/j.rcim.2019.03.007
3. Nelles O. *Nonlinear System Identification: from Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes*. Springer, 2020, 1253 p. https://doi.org/10.1007/978-3-030-47439-3
4. Han D., Yoo H.-J. *On-Chip Training NPU-Algorithm, Architecture and SoC Design*. Springer, 2023, 260 p. https://doi.org/10.1007/978-3-031-34237-0
5. Han D., Lee J., Lee J., Choi S., Yoo H.-J. A 141.4 mW low-power online deep neural network training processor for real-time object tracking in mobile devices. *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5. https://doi.org/10.1109/iscas.2018.8351398
6. Varagula J., Kulproma P., Itob T. Object detection method in traffic by on-board computer vision with time delay neural network. *Procedia Computer Science*, 2017, vol. 112, pp. 127–136. https://doi.org/10.1016/j.procs.2017.08.185
7. Heidaryan M., Karimi G. FPGA implementation of two multilayer perceptron neural network in cascade for efficient real time hand gestures tracking. *Microprocessors and Microsystems*, 2023, vol. 100, pp. 104849. https://doi.org/10.1016/j.micpro.2023.104849
8. Seng K.P., Ang L.M. Embedded intelligence: State-of-the-art and research challenges. *IEEE Access*, 2022, vol. 10, pp. 59236–59258. https://doi.org/10.1109/access.2022.3175574
9. Ortega-Zamorano F., Jerez J.M., Munoz D.U., Luque-Baena R.M., Franco L. Efficient implementation of the backpropagation algorithm in FPGAs and microcontrollers. *IEEE Transactions on Neural Networks and Learning Systems*, 2015, vol. 27, no. 9, pp. 1840–1850. https://doi.org/10.1109/tnnls.2015.2460991
10. Ushenina I.V., Danilov E.A. FPGA implementation of the artifical neuron module for series-parallel architectures of feedforward neural networks. *Digital Signal Processing*, 2025, no. 1, pp. 78–84. (in Russian)
11. Zhang M.J., Garcia S., Terre M. Real-time fast learning hardware implementation. *International Journal for Simulation and Multidisciplinary Design Optimization*, 2023, vol. 14, pp. 1. https://doi.org/10.1051/smdo/2023001
12. Tisan A., Chin J. An end-user platform for FPGA-based design and rapid prototyping of feedforward artificial neural networks with on-chip backpropagation learning. *IEEE Transactions on Industrial Informatics*, 2016, vol. 12, no. 3, pp. 1124–1133. https://doi.org/10.1109/tii.2016.2555936
13. Zhang Z., Wang G., Wang K., Gan B., Chen G. Efficient on-chip learning of multi-layer perceptron based on neuron multiplexing method. *Electronics*, 2023, vol. 12, no. 17, pp. 3607. https://doi.org/10.3390/electronics12173607
14. Holt J.L., Hwang J.-N. Finite precision error analysis of neural network hardware implementations. *IEEE Transactions on Computers*, 1993, vol. 42, no. 3, pp. 281–290. https://doi.org/10.1109/12.210171
15. Ushenina I.V. Realization of the sigmoid activation function for neural networks on current FPGAs by the table-driven method. *Tomsk State University Journal of Control and Computer Science*, 2024, no. 69, pp. 124–133. (in Russian). https://doi.org/10.17223/19988605/69/13

Автор

Ушенина Инна Владимировна — кандидат технических наук, доцент, доцент, Пензенский государственный технологический университет, Пенза, 440039, Российская Федерация, sc 57208836904, https://orcid.org/0000-0001-9674-2604, ivl23@yandex.ru

Author

Inna V. Ushenina — PhD, Associate Professor, Associate Professor, Penza State Technological University, Penza, 440039, Russian Federation, sc 57208836904, https://orcid.org/0000-0001-9674-2604, ivl23@yandex.ru

Статья поступила в редакцию 25.08.2025
Одобрена после рецензирования 18.02.2026
Принята к печати 17.03.2026

Received 25.08.2025
Approved after reviewing 18.02.2026
Accepted 17.03.2026