Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
« ___» _____ 2019 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Метод аппаратной реализации рекуррентных нейронных сетей
тема

09.04.01 «Информатика и вычислительная техника»
код и наименование направления

09.04.01.06 «Микропроцессорные системы»
код и наименование магистерской программы

Научный руководитель	_____ подпись, дата	проф., зав. каф. ВТ, канд. техн. наук должность, ученая степень	О.В.Непомнящий инициалы, фамилия
Выпускник	_____ подпись, дата		А.Г.Хантимиров инициалы, фамилия
Рецензент	_____ подпись, дата	Professor, PhD Luleå tekniska universitet должность, ученая степень	Evgeny Osipov инициалы, фамилия
Нормоконтроллер	_____ подпись, дата		А.И.Постников инициалы, фамилия

Красноярск 2019

РЕФЕРАТ

Магистерская диссертация по теме «Метод аппаратной реализации рекуррентных нейронных сетей» содержит 57 страниц текстового документа, 25 иллюстраций, 44 использованных источников, 5 формул, 2 приложения.

НЕЙРОННЫЕ СЕТИ, СЕТИ ЭХО-СОСТОЯНИЙ, РЕКУРРЕНТНЫЕ НЕЙРОННЫЕ СЕТИ, ПЛИС, ОПТИМИЗАЦИЯ.

Задачи:

– выполнить исследования способов и методов резервуарных вычислений в классе рекуррентных нейронных сетей;

– предложить метод реализации рекуррентных нейронных сетей на аппаратном уровне;

– разработать и выполнить сравнительный анализ моделей нейронных сетей, предназначенных для аппаратной реализации;

– разработать программную модель и выполнить исследование целочисленной сети эхо-состояний в базисе ПЛИС.

Основные полученные результаты:

– проведены исследования способов и методов резервуарных вычислений в классе рекуррентных нейронных сетей. Рассмотрен подход аппаратной реализации рекуррентных нейронных сетей;

– предложен метод эффективной реализации нейронных сетей эхо-состояний на аппаратном уровне, основанный на принципе многомерных вычислений. Вычисления сетей эхо-состояний реализованы простыми арифметическими операциями (сложение/объединение, перестановки). Уменьшена разрядность вычислений резервуара, уменьшено количество соединений, увеличено быстродействие сети. Предложена архитектура целочисленной сети эхо-состояний, по принципу функционирования аналогичная традиционной сети эхо-состояний, однако требующая значительно меньшие ресурсы при аппаратной реализации;

– предложены модели сетей эхо-состояний. Проведены эксперименты, показывающие способность целочисленных сетей эхо-состояний справляться с задачами традиционных сетей эхо-состояний. Проведен сравнительный анализ занимаемого на кристалле места для традиционной и предложенной моделей;

– реализован набор модулей для создания целочисленных сетей эхо-состояний в виде параметризируемых сложно-функциональных блоков на языке описания аппаратуры VerilogHDL.

СОДЕРЖАНИЕ

Введение 5
1 Резервуарные вычисления в классе рекуррентных нейронных сетей 8
1.1 Задача организации аппаратных вычислений для нейронных сетей 8
1.2 Рекуррентные нейронные сети 9
1.3 Резервуарные вычисления 11
1.4 Сети эхо-состояний 13
1.5 Вывод 15
2 Реализация рекуррентных нейронных сетей на аппаратном уровне 16
2.1 Основы многомерных вычислений 16
2.2 Целочисленные сети эхо-состояний 18
2.3 Вывод 20
3 Разработка и исследование моделей нейронных сетей, предназначенных для аппаратной реализации 22
3.1 Модель традиционной сети эхо-состояний 22
3.2 Модель целочисленной сети эхо-состояний 23
3.3 Результаты экспериментальных исследований 25
3.4 Вывод 31
4 Реализация и исследование целочисленной сети ЭС на базе программируемой логической интегральной схемы 32
4.1 СФ-блок объектной памяти 33
4.2 СФ-блок резервуара 34
4.3 СФ-блок интерпретатора 36
4.4 СФ-блок умножителя 37
4.5 СФ-блок нормализатора 38
4.6 СФ-блок функции активации 39
4.7 СФ-блок управления 40
4.8 Вывод 41
Заключение 42
Список сокращений 43
Список использованных источников 44
Приложение А Программный код моделей эхо-сетей 49
Приложение Б Фрагмент разработанного ПЛИС-проекта 53

ВВЕДЕНИЕ

На сегодняшний день, работы в области резервуарных вычислений (РВ) [1, 2] показывают, что рекуррентные нейронные сети с постоянными связями могут запоминать и генерировать сложные пространственно-временные последовательности. РВ являются мощным инструментом для решения задач моделирования и предсказания состояния динамических систем.

Однако аппаратная реализация классических нейронных сетей является нетривиальной задачей. Это обусловлено высокими требованиями к вычислительным мощностям аппаратной платформы, большими ресурсными затратами, временем обучения и пр.

Таким образом, **актуальной задачей** является исследование и создание метода архитектурной организации и аппаратного инструментария для практической реализации нейронных сетей предназначенных для решения задач управления динамическими объектами в режиме реального времени.

Последние работы в области теории и практики применения нейронных сетей прямого распространения показывают, что бинаризация фильтров в сверточных нейронных сетях может привести к большому выигрышу в памяти и вычислительной эффективности [3]. Сокращение памяти, занимаемой каждым нейроном или синапсом с 32 бит до нескольких бит или даже двух, сохраняет вычисление с минимальной потерей производительности [4, 5]. Это расширяет сферу применения таких сетей.

В процессе исследований были обнаружены некоторые сходства между операциями в РВ и многомерными вычислениями [6]. Многомерные вычисления являются парадигмой для нейронно-символьных представлений [7, 8], вычислений и «рассуждений по аналогии». Отличие многомерных от традиционных вычислений состоит в том, что все сущности (объекты, фонемы, символы) представлены случайными векторами очень большой размерности – несколько тысяч бит. В данном случае, комплексные структуры данных и рассуждения по аналогии реализуются простыми арифметическими

операциями (сложение/объединение, перестановки) [7]. Исследования показали, что РВ и многомерные вычисления связаны следующими основными принципами:

– случайные проекции входных значений на резервуар (которые, по сути, являются многомерными векторами) соединяют случайные представления многомерных вычислений, хранящиеся в суперпозиции;

– обновление резервуара посредством использования случайной рекуррентной матрицы схоже с операциями многомерных вычислений, такими как компоновка/перестановка;

– нелинейность резервуара может быть достигнута установлением порога при сложении целых чисел в многомерных вычислениях.

Это позволяет перейти к созданию архитектуры целочисленной сети эхо-состояний (ЭС), которая функционирует так же, как традиционная сеть ЭС, однако использует меньше памяти и использует меньше вычислительной мощности. В такой архитектуре резервуар сети содержит только целые числа, ограниченные несколькими битами для каждого нейрона, что позволяет значительно сократить занимаемые вычислительные ресурсы. Достижение рекуррентности резервуара посредством перемножения матриц может быть заменено операцией перестановки посредством циклического сдвига, что позволит значительно уменьшить выделяемые для вычислительной логики ресурсы кристалла.

Таким образом, цель данной работы состоит в тестировании и отработке метода аппаратной реализации сетей ЭС на основании целочисленной архитектуры.

Для достижения поставленной цели определены основные задачи исследования:

– выполнить исследования способов и методов резервуарных вычислений в классе рекуррентных нейронных сетей;

– предложить метод реализации рекуррентных нейронных сетей на аппаратном уровне;

– разработать и выполнить сравнительный анализ моделей нейронных сетей, предназначенных для аппаратной реализации;

– разработать программную модель и выполнить исследование целочисленной сети ЭС в базисе ПЛИС.

Предполагаемая **научная новизна** исследования заключается в предложенном методе архитектурной организации нейронных сетей для реализации в виде заказной (полузаказной) интегральной схемы, или ПЛИС, базирующемся на использовании целочисленной сети ЭС, позволяющим минимизировать требуемые ресурсы кристалла и увеличить скорость обработки данных.

1 Резервуарные вычисления в классе рекуррентных нейронных сетей

1.1 Задача организации аппаратных вычислений для нейронных сетей

В настоящее время происходит стремительное развитие нейросетевых технологий. Подавляющее большинство сервисов по распознаванию текста, аудио, изображений, сервисов по переводу текстов и другие работают на основе нейросетей [9, 10]. Используемые нейросети функционируют на программном уровне, так как реализация нейросетей на аппаратном уровне является сложной задачей [11].

Одним из определяющих факторов для аппаратной реализации является ограниченность ресурсов целевой платформы. Для сложных задач невозможно вместить на кристалл или расположить в памяти ЭВМ достаточное количество нейронов. Кроме того, большое количество связей между нейронами, так же накладывает серьезные аппаратные ограничения. Усугубляется ситуация растущими требованиями к вычислительной мощности, которые растут с увеличением сложности сети.

Таким образом, требуется решение трех основных задач:

– уменьшение занимаемого нейронами ресурса;

– уменьшение связей между нейронами;

– переход к более простой модели вычислений.

Это позволит значительно увеличить скорость работы сети, что является ключевым фактором при работе с системами динамическими системами реального времени, критичными ко времени реакции на внешние события.

Это обуславливает использование рекуррентных нейронных сетей, поскольку именно такой подход будет лучшим решением для создания однокристального нейросетевого регулятора для динамических систем реального времени.

1.2 Рекуррентные нейронные сети

Искусственные рекуррентные нейронные сети (РНС) представляют большой и изменяющийся класс вычислительных моделей, разработанных по аналогии с биологической моделью мозга [12]. Характерной особенностью РНС, которая отличает их от остальных сетей, использующих обратную связь заключается в том, что топология их соединений содержит циклы [13]. Наличие в структуре сети циклов приводит к следующему:

– РНС могут развить самоподдерживающуюся динамику временных изменений при помощи рекуррентных соединений, даже при отсутствии входного сигнала. Математически РНС можно представить в виде динамических систем, в то время как сети с прямым распространением – функциями;

– при наличии входного сигнала, РНС сохраняет во внутреннем состоянии нелинейное отображение истории входных сигналов – другими словами, имеет динамическую память и возможность обрабатывать временную контекстную информацию.

Отметим, что РНС являются основным элементом для задач машинного обучения, теории вычислений, нелинейной обработки сигналов и управления [14].

Известны два класса РНС. Модели из первого класса характеризуются минимизацией энергетических затрат на стохастическую динамику и симметричные связи. Наиболее известны сети Хопфилда [15, 16], машины Больцмана [17, 18], сети глубокого доверия [19]. Обучение таких сетей в основном происходит по модели «обучение без учителя». Типичными задачами для этого направления являются ассоциативная память, компрессия данных, неконтролируемое моделирование распространения данных и распознавание статических образов [20].

Типичной особенностью второго класса моделей РНС является детерминированная динамика обновлений и точечные соединения [21].

Системы из этого класса используются в качестве нелинейных фильтров, которые преобразуют входные временные последовательности в выходные временные последовательности. В основе таких сетей лежат нелинейные динамические системы. Эти системы обучаются по модели «обучение с учителем».

Не смотря на высокий потенциал и некоторое количество успешных академических и практических применений, вклад РНС в нелинейное моделирование остается небольшим. Главной причиной этого является сложность обучения РНС, так как для таких сетей не подходят обычные методы обучения на основе градиентного спуска, направленные на уменьшение ошибки обучения. Известен ряд алгоритмов обучения РНС, однако все они имеют следующие недостатки [22]:

– постепенное изменение параметров сети в процессе обучения стимулирует динамику сети через бифуркации [23]. В таких случаях, градиентная информация размывается и может стать недостаточно определенной. Как следствие, сходимость не может быть гарантирована;

– обновление единственного параметра может быть вычислительно затратным и может потребоваться множество циклов обновления. Это приводит к увеличению времени обучения, соответственно делает возможным обучение только сравнительно малых РНС;

– в действительности, тяжело обучить зависимости, требующие долгосрочной памяти, так как необходимая градиентная информация экспоненциально растворяется со временем [24];

– продвинутые алгоритмы обучения используют достаточно большое количество глобальных параметров, которые непросто оптимизировать. В результате, успешное применение таких алгоритмов требует существенных навыков и опыта.

1.3 Резервуарные вычисления

В 2001 году был предложен новый подход к созданию и обучению РНС [25]. Независимо друг от друга, Вольфганом Маассом и Гербертом Йегером были предложены схожие подходы для реализации РНС. В первом случае – «машина неустойчивого состояния» [26], во втором – «сеть ЭС» [27]. Этот подход в данный момент известен под общим названием «Резервуарные вычисления» (РВ). Парадигма РВ избегает сложностей обучения РНС, основанных на методах градиентного спуска, в связи с построением РНС по следующим принципам:

– РНС задается случайно и остается неизменной на протяжении обучения. Такая РНС называется резервуаром. Она пассивно возбуждается входным сигналом и поддерживает в своем состоянии нелинейное преобразование входной истории;

– желательный выходной сигнал генерируется как линейная комбинация нейронных сигналов резервуара. Эта линейная комбинация получается при помощи линейной регрессии, используя обучающий сигнал как целевой.

На рисунках 1, 2 изображено различие в подходах традиционной РНС и РНС построенной на основе РВ.

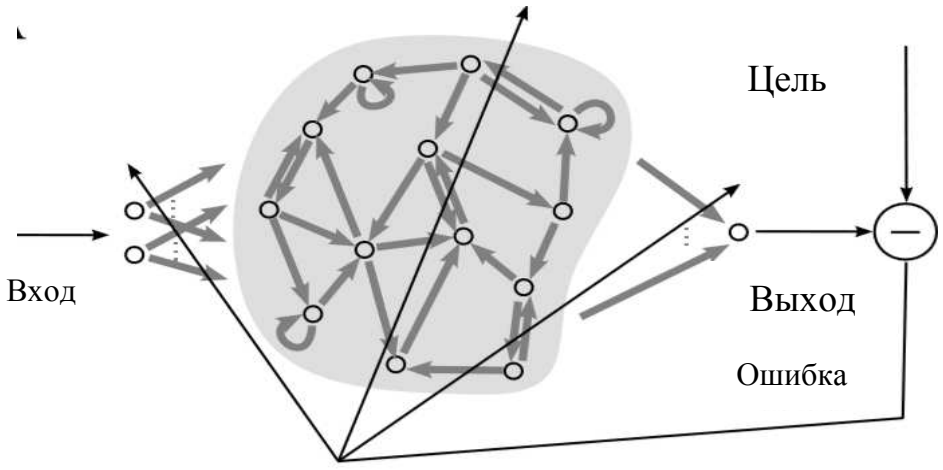

Рисунок 1 – Традиционная рекуррентная нейронная сеть, при обучении изменяются весовые коэффициенты всех связей

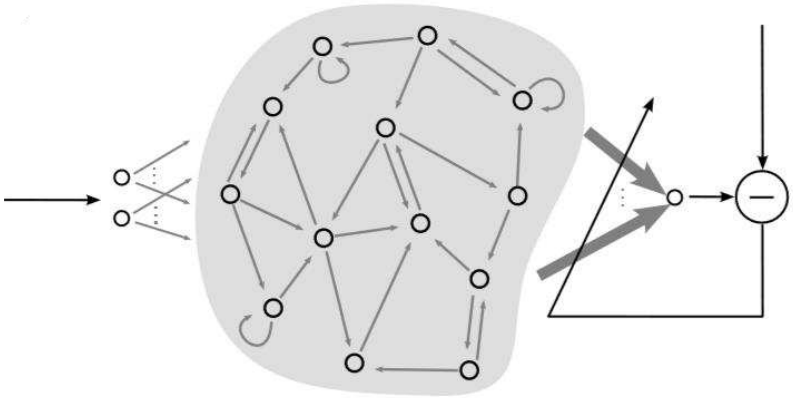

Рисунок 2 – Рекуррентная нейронная сеть, построенная по методу резервуарных вычислений, при обучении изменяются весовые коэффициенты выходного слоя, остальные не изменяются

Методы РВ быстро стали популярны, и на сегодняшний день они являются одной из основных парадигм РНС [28]. Основными причинами развития РВ являются:

– точность моделирования. РВ значительно превосходят предыдущие методы идентификации нелинейных систем, предсказания и классификации, например, в предсказании хаотической динамики [29], нелинейной беспроводной коррекции канала [29], финансовом прогнозировании, распознавании отдельных разговорных цифр;

– объем моделирования. РВ являются универсальными для продолжительных по времени, продолжительных по значению систем реального времени, моделируемых с ограниченными ресурсами (включая время и разрешение значения);

– биологическая достоверность. Архитектура и динамические характеристики РВ схожа со строением биологического мозга;

– расширяемость и экономность. Сложной задачей исследования нейронных сетей является проблема расширение прежде обученных моделей новыми объектами без ухудшения или уничтожения прежде изученных представлений. РВ предлагают простое решение – новые объекты

представляются новыми выходными слоями, которые надстраиваются над резервуаром. Так как слои независимы, они не мешают работе друг друга.

1.4 Сети эхо-состояний

Рассмотрим функционирование традиционной сети ЭС, согласно [30]. На рисунке 3 изображена архитектура традиционной сети ЭС, включающая в себя три слоя нейронов. Входной слой из K нейронов содержит текущее значение входного сигнала, обозначенного **u**(n). Выходной слой (L нейронов) производит выходное значение сети (обозначается y(n)) в течение операционной фазы. Скрытый слой сети с N нейронами называется резервуаром, его состояние в момент времени n обозначается как x(n).

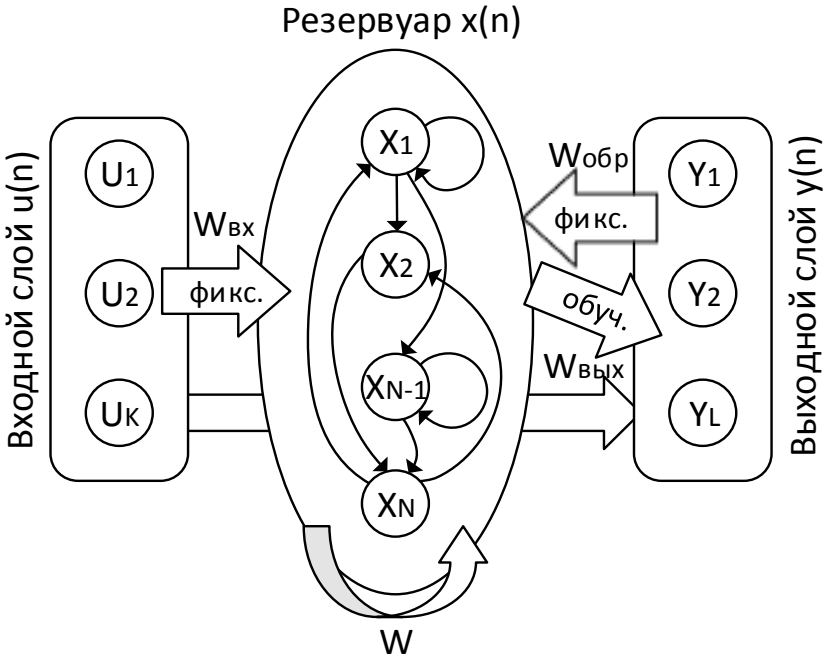

Рисунок 3 – Архитектура традиционной сети эхо-состояний

В общем случае, соединения в сети ЭС описываются четырьмя матрицами. $W^{вх}$ описывает соединения между входным слоем и резервуаром, $W^{обр}$ описывает соединения резервуара с выходным слоем. Обе матрицы проецируют текущие входные и выходные значения на резервуар. Сеть ЭС

обладает памятью благодаря рекуррентным соединениями между нейронами в резервуаре, которые описаны матрицей резервуара W. Матрица $W^{вых}$ преобразует текущие значения во входном слое и резервуаре (u(n) и x(n), соответственно) в выход сети y(n). Матрицы ($W^{вх}$, $W^{обр}$, и W) задаются случайно при инициализации сети и остаются неизменными в течение жизненного цикла сети. Таким образом, процесс обучения сфокусирован на изучении считывающей матрицы $W^{вых}$. Не существует жестких ограничений для генерации проецирующих матриц $W^{вх}$ и $W^{обр}$. Обычно они задаются случайно с нормальным или равномерным распределением и масштабируются, как показано ниже. Матрица соединений резервуара, тем не менее, имеет ограничения согласно свойства эхо-состояния. Это свойство достигается, когда спектральный радиус матрицы W меньше либо равен единице. Например, W может быть сгенерирована при помощи нормального распределения и нормализована по ее максимальному собственному значению. Пока не указано другого, в данной работе будет использоваться ортогональная матрица для описания соединений резервуара; такая матрица создана применением QR-разложения к случайной матрице, сгенерированной при помощи стандартного нормального распределения. Также, W может быть масштабирована параметром обратной связи, см. (1).

Обновление резервуара сети со временем n описано следующим уравнением:

$$x(n) = \tanh\left(\rho W x(n-1) + \beta W^{вх} u(n) + \beta W^{обр} y(n-1)\right), \quad (1)$$

где β и ρ обозначают коэффициент проекции и параметр обратной связи, соответственно. Обратите внимание, предполагается, что спектральный радиус матрицы соединений резервуара W равен единице. Также обратите внимание, что на каждом шаге нейроны в резервуаре принимают тангенс как функцию активации. Нелинейность предотвращает сеть от вырождения ограничением

области возможных значений от -1 до 1. Функция выходного слоя вычисляется как:

$$\hat{y}(n) = g\left(W^{\text{обр}}[x(n); u(n)]\right), \quad (2)$$

где точка с запятой обозначает конкатенацию двух векторов и g() обозначает функцию активации выходных нейронов, например, линейная или победитель получает все.

1.5 Вывод

Рассмотрена задача аппаратной реализации нейронных сетей. Определено, что реализация классических нейронных сетей на аппаратном уровне является нетривиальной задачей.

Выделен класс рекуррентных нейронных сетей, допускающих эффективную аппаратную реализацию.

Рассмотрены способы создания таких сетей.

Предложен метод аппаратной реализации, базирующийся на использовании принципа резервуарных вычислений.

Определено, что реализованные на аппаратном уровне сети ЭС найдут применение в задачах предсказания и идентификации для нелинейных динамических систем. Следовательно, на их основе возможно построение систем, предназначенных для управления сложными многокомпонентными объектами в реальном времени. В связи с развитием технологии производства микрочипов, такой нейронный регулятор может быть реализован на платформе ПЛИС или СБИС класса “Система на кристалле”. Тем не менее, остается проблема нехватки аппаратных ресурсов при расширении нейронной сети. Решение может быть найдено при использовании целочисленных эхо-сетей, это подразумевает разработку нового способа архитектурной организации таких сетей.

2 Реализация рекуррентных нейронных сетей на аппаратном уровне

Основой целочисленных сетей ЭС является парадигма многомерных вычислений.

2.1 Основы многомерных вычислений

В текущем представлении, которое используется во всех современных цифровых компьютерах, для того, чтобы в полной мере интерпретировать представление, необходима группа бит [31]. В многомерных вычислениях все сущности (объекты, фонемы, символы) представляются в виде векторов большой размерности – тысячи бит. Информация используется в распределенном представлении, в котором, вопреки текущему представлению, любое подмножество бит может быть интерпретировано. Вычисления с распределенным представлением используют статистические свойства векторных пространств с большой размерностью, что позволяет проводить приблизительные, устойчивые к шуму, высоко-параллельные вычисления. Объектная память необходима для восстановления составных представлений, присвоенных к сложным данным. Существует несколько видов многомерных вычислений с распределенным представлением, различимых случайным распределением элементов вектора, которые могут быть действительными числами [32 – 35], комплексными числами [36], двоичными числами [37, 38] или биполярными [33, 39].

Канерва [37] предложил использовать распределенные представления, содержащие N = 10000 бинарных элементов (называемых многомерными векторами). Значения каждого элемента многомерного вектора независимы и равновероятны, следовательно, они также называются плотно-распределенным представлением. Сходство между двумя бинарными многомерными векторами характеризуется расстоянием Хемминга, которое (для двух векторов) измеряет количество элементов, в которых они различны. При очень большой

размерности, расстояние Хемминга (нормализованное размерностью N) между любым произвольным многомерным вектором и всеми остальными многомерными векторами в том же пространстве, приближается к 0,5 [37, 40].

Бинарные многомерные вектора могут быть эквивалентно перенесены к биполярному представлению, где каждый элемент вектора кодируется как “-1” или “+1”. Такое определение более удобно в чисто вычислительных причинах. Показателем расстояния в биполярном представлении является точечное произведение:

$$dist = x^{\top} y. \tag{3}$$

Основными символами в многомерных вычислениях являются элементарные многомерные вектора. Они задаются случайно и независимо, и, благодаря большой размерности, будут близки к ортогональным с очень большой вероятностью, то есть мера сходства (произведение точек) между такими многомерными векторами стремится к нулю. Упорядоченная последовательность символов может быть закодирована в комплексный многомерный вектор, используя элементарные многомерные вектора, перестановку (например, циклический сдвиг в качестве особого случая перестановки) и операторы объединения. Такой вектор кодирует целую последовательность в сложный многомерный вектор и напоминает нейронный резервуар.

Обычно, в многомерных вычислениях восстановление компонентов элементарных многомерных векторов из комплексных многомерных векторов выполняется поиском наиболее похожих векторов, хранящихся в объектной памяти. Тем не менее, чем больше векторов смешаны вместе, тем сильнее взаимный шум и вероятность восстановления правильных элементарных многомерных векторов снижается.

Работа [6] показывает влияние взаимного шума и показывает, что различные виды многомерных вычислений имеют универсальный объем

памяти. Таким образом, различные виды многомерных вычислений могут заменяться без понижения производительности. При понимании этого, возможно разработать намного более эффективную сеть резервуарных вычислений для цифровой аппаратуры.

2.2 Целочисленные сети эхо-состояний

Архитектура целочисленной сети ЭС изображена на рисунке 4. Такая сеть идентична по структуре обычной сети ЭС (см. рисунок 3) с тремя слоями нейронов: входной (u(n), K нейронов), выходной (y(n), L нейронов) и резервуар (x(n), N нейронов). Отметим, что обучение считывающей матрицы $W^{вых}$ для целочисленной сети ЭС, аналогично обучению выходной матрицы традиционной сети ЭС.

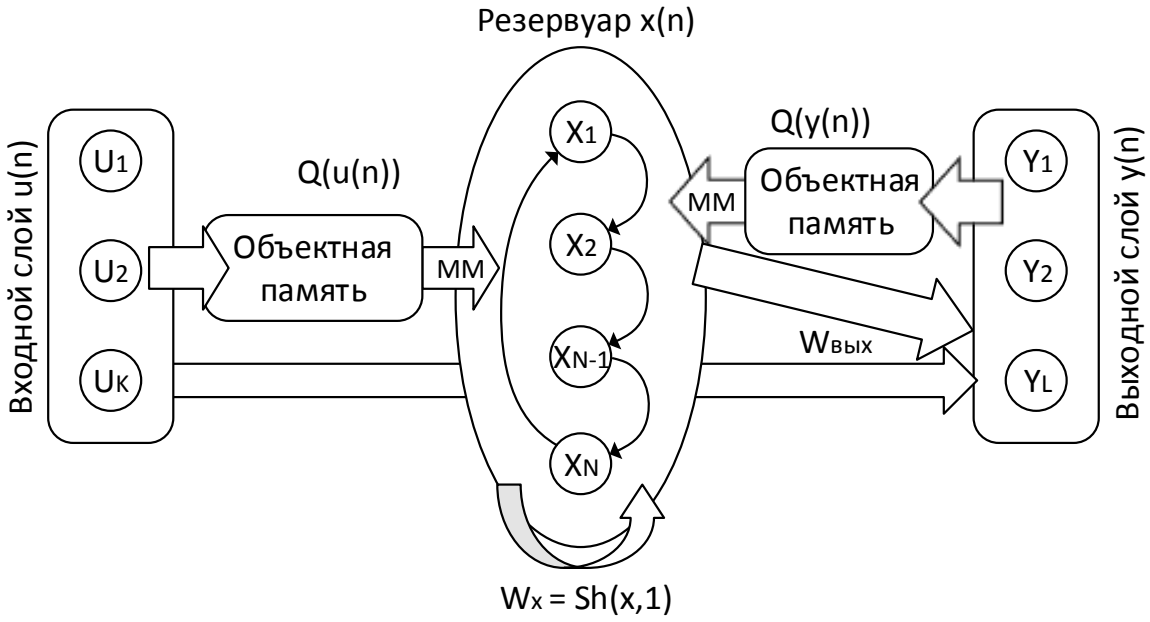

Рисунок 4 – Архитектура целочисленной сети эхо-состояний

Тем не менее, остальные компоненты целочисленной сети ЭС отличаются от обычной сети ЭС. Во-первых, активации входного и выходного слоев проецируются на резервуар в форме многомерных биполярных векторов [34]

размера N (отмеченных как $u^{HD}(n)$ и $y^{HD}(n)$). Для проблем где входные и выходные данные описаны конечным алфавитом и каждый символ может быть интерпретирован независимо, отображение N-мерного пространства достигается простым присвоением случайного многомерного биполярного вектора каждому символу в алфавите и хранению их в объектной памяти [7]. В случае непрерывных данных (например, действительных чисел), мы производим квантование непрерывных данных в конечный алфавит. Схема квантования и точность квантования зависят от решаемой проблемы. В дополнение, когда появляется надобность в сохранении схожести между уровнями квантования, применяются схемы сохранения расстояния (см. пример [41, 42]), который может сохранить, например, линейную или нелинейную схожесть между уровнями.

Продолжительные значения также могут быть представлены в виде многомерных векторов варьированием их плотности распределения. Другой особенностью целочисленной сети ЭС является метод, по которому создается рекуррентность в резервуаре. Вместо перемножения матриц, рекуррентность реализована при помощи перестановок вектора резервуара. Перестановки вектора могут быть описаны в матричной форме, которая может играть роль матрицы W в целочисленной сети ЭС. Согласно [27], одним из основных условий задания такой матрицы является равенство ее спектрального радиуса единице. Тем не менее, эффективная реализация перестановки может быть достигнута в особом случае – циклическим сдвигом (отмеченным как Sh()). Рисунок 4 показывает рекуррентные соединения нейронов в резервуаре с рекуррентностью посредством циклического сдвига на одну позицию. В таком случае, векторно-матричное перемножение Wx(n) эквивалентен Sh(x(n), 1).

Для хранения целочисленных значений нейронов, целочисленная сеть ЭС использует различные нелинейные функции активации для резервуара – ограничение (4). Самая простая операция объединения – это поэлементное сложение. Тем не менее, при использовании поэлементного сложения, функция

резервуара (т.е. комплексный многомерный вектор) более не биполярна. С точки зрения реализации целесообразно хранить значения элементов многомерного вектора в ограниченном диапазоне, используя пороговое значение (отмеченное как k).

$$f_k(x) = \begin{cases} -k, & x \leq -k \\ x, & -k < x < k. \\ k, & x \geq k \end{cases} \quad (4)$$

Ограничение порогом k регулирует нелинейное поведение резервуара и ограничивает диапазон значений функции. В целочисленной сети ЭС резервуар обновляется только целочисленными биполярными векторами и после ограничения значения нейронов целочисленные в диапазоне от –k до k. Таким образом, каждый нейрон представлен с помощью $\log_2(2k+1)$ битов памяти. Например, если k = 7, существует 15 различных значений нейрона, хранящихся всего в 4 битах.

Подводя итог приведенных различий, обновление целочисленной сети ЭС можно описать как:

$$x(n) = f_k\big(Sh(x(n-1), 1) + u^{HD}(n) + y^{HD}(n-1)\big). \quad (5)$$

2.3 Вывод

Для разработки нового метода архитектурной организации целочисленных сетей ЭС, рассмотрены основы многомерных вычислений. Результаты позволили перейти от традиционной сети ЭС к целочисленной сети ЭС. Такой переход основывается на изменении устройства резервуара, а именно:

– случайные проекции входных значений на резервуар (которые, по сути, являются многомерными векторами) соединяют случайные представления многомерных вычислений, хранящиеся в суперпозиции;

– обновление резервуара посредством использования случайной рекуррентной матрицы схоже с операциями многомерных вычислений, такими как компоновка/перестановка;

– нелинейность резервуара может быть достигнута установлением порога при сложении целых чисел в многомерных вычислениях.

Описанный выше метод реализации сетей ЭС позволит сократить затраты занимаемой памяти внутри резервуара на 70-80%, объема используемых вычислительных элементов на 90-95% по сравнению с традиционным методом реализации резервуара. Однако следует разработать модель функционирования и выяснить, возможно ли эффективное использование предлагаемой архитектуры.

3 Разработка и исследование моделей нейронных сетей, предназначенных для аппаратной реализации

Выполнено моделирование сетей эхо-состояния, это позволило:

– определить эффективность рекуррентности резервуара целочисленной сети ЭС при сравнении с традиционной по критерию точности расшифровки значения резервуара через определенное количество итераций;

– определить целесообразность реализации традиционной эхо-сети на аппаратном уровне;

– определить влияние параметров ЭС на эффективность рекуррентности.

3.1 Модель традиционной сети эхо-состояний

Модель традиционной сети ЭС разработана на языке Matlab (Рисунок Рисунок 3). Текст программы приведен в приложении А.

При инициализации сети задаются случайные значения связей внутри резервуара, матрицы проекции входных значений (W, $W^{вх}$ соответственно).

Следующим шагом является настройка матрицы выходных проекций резервуара, что является обучением традиционной сети ЭС. Обучение производится с помощью линейной регрессии выходных значений обучающей выборки. Для последующего определения эффективности способности “запоминать” информацию в краткосрочном периоде, производится обучение массива матриц выходных весов, соответствующим количеству циклов работы резервуара. Таким образом, на модели становится возможно определить точность работы нейронной сети через определенное количество циклов.

Заключительным шагом является тестирование обученной нейронной сети на массиве данных с целью выявления точности определения правильного значения на различных значениях задержки.

Блок-схема алгоритма работы модели традиционной сети ЭС приведена на рисунке 5.

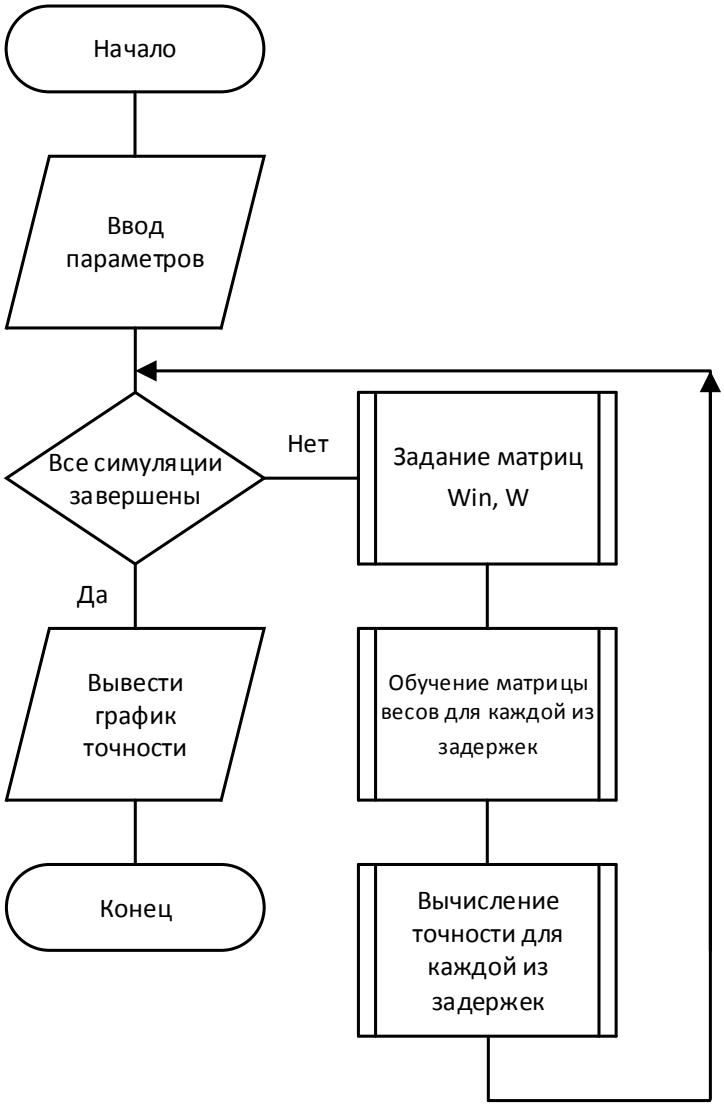

Рисунок 5 – Блок-схема алгоритма работы модели традиционной сети эхо-состояний

3.2 Модель целочисленной сети эхо-состояний

Модель целочисленной сети ЭС построена в соответствии с описанием, приведенным в пункте 2.2 данной работы. Архитектура сети представлена на рисунке Рисунок 4. Текст программы приведен в приложении Б. Основной целью рассмотрения модели является выявление влияния параметров сети на

изменение характеристики рекуррентности по сравнению с традиционной сетью ЭС.

На первом этапе работы с моделью задаются параметры сети. С учетом заданных параметров происходит инициализация объектной памяти случайными биполярными векторами.

Затем производится обучение выходных матриц весов для каждой из задержек посредством вычисления линейной регрессии.

Блок-схема алгоритма работы целочисленной сети ЭС приведена на рисунке Рисунок 6.

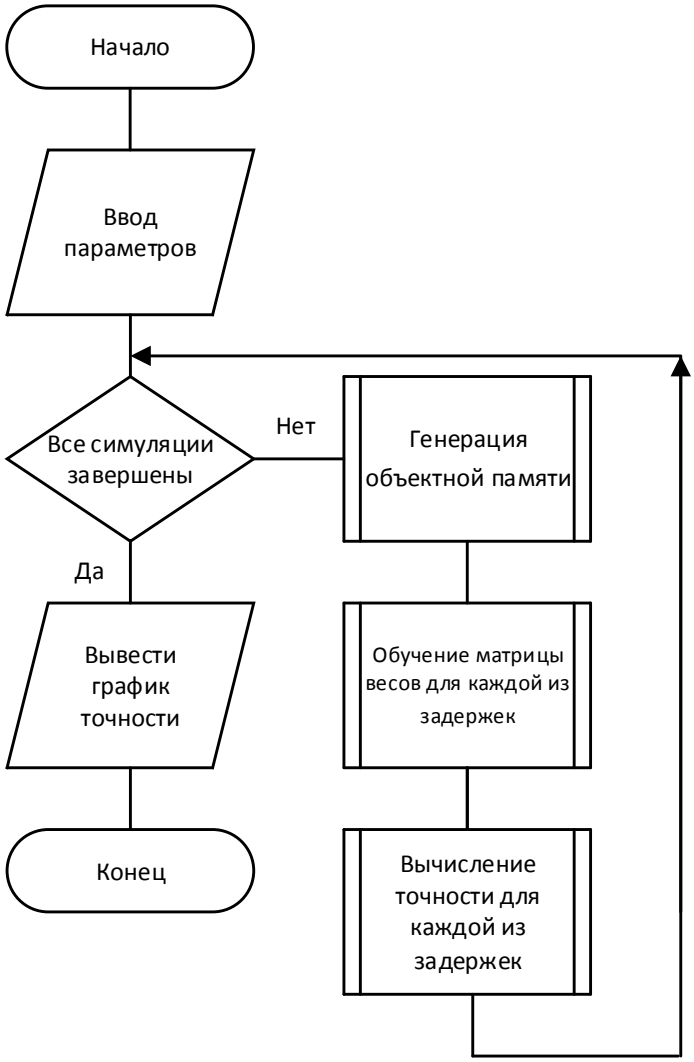

Рисунок 6 – Блок-схема алгоритма работы модели целочисленной сети эхо-состояний

Как и для модели традиционной сети ЭС вычисляется точность интерпретации состояния резервуара на каждой из задержек, что позволяет провести сравнение.

3.3 Результаты экспериментальных исследований

В качестве первого эксперимента вычисления точности декодирования, использовалась стандартная модель целочисленной сети ЭС и традиционная модель с полносвязной архитектурой. Под понятием полносвязная архитектура в данном контексте подразумевается архитектура, выходы нейронов резервуара которой поступают на вход всех нейронов резервуара. Такая архитектура крайне невыгодна с точки зрения аппаратной реализации, однако наиболее полно отображает свойства сетей эхо-состояния.

Результаты моделирования представлены на рисунке 7, по оси ординат приведена средняя по всем симуляциям точность определения входного символа, по оси абсцисс количество циклов задержки.

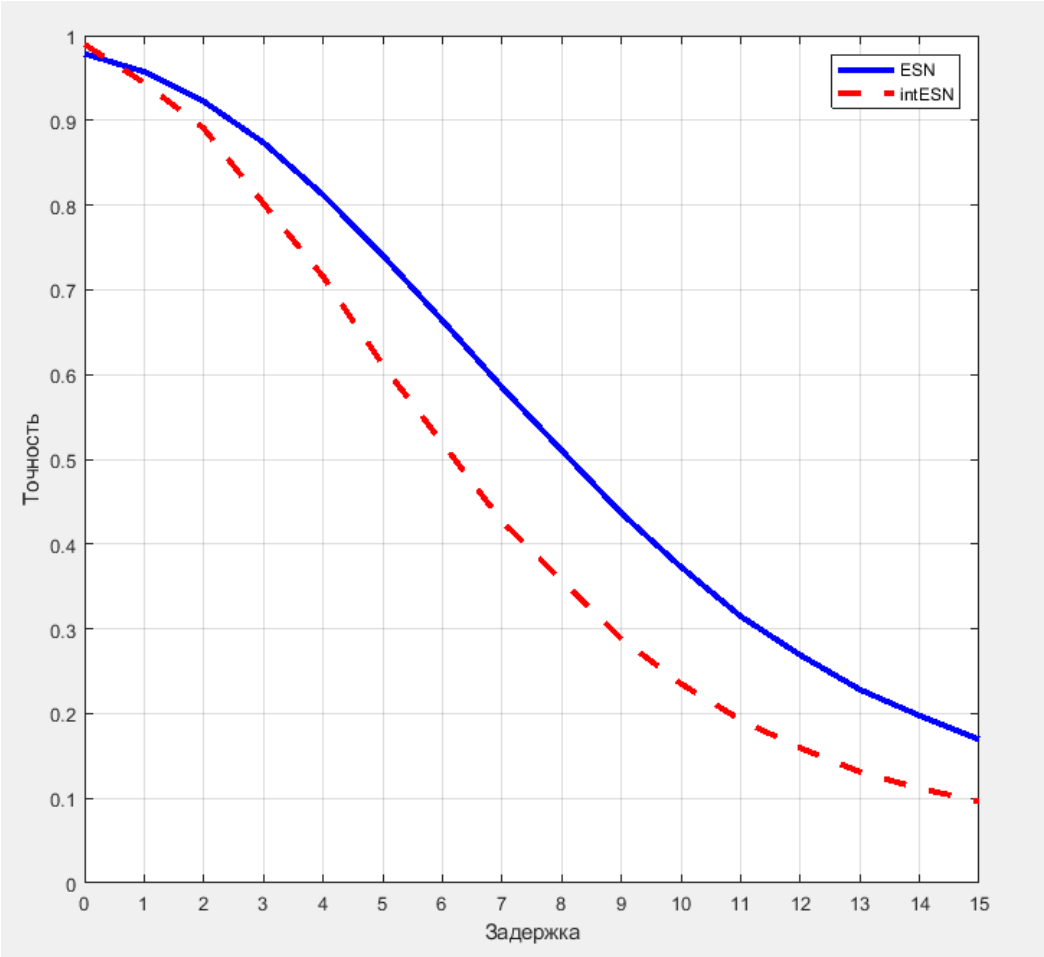

Рисунок 7 – Графики зависимости точности распознавания символа от количества циклов задержки при размере резервуара 100 нейронов

Из рисунка 7 следует, что точность определения входного символа у целочисленной сети ЭС при размере резервуара 100 нейронов ниже чем у традиционной сети на всех временных задержках. На рисунке 8 показаны результаты аналогичного эксперимента, с увеличенным в 10 раз количеством нейронов резервуара. Из рисунка видно, что обе архитектуры данной мощности одинаково справляются с задачей примерно до десятого цикла задержки, после чего возможности целочисленной сети ЭС начинают снижаться.

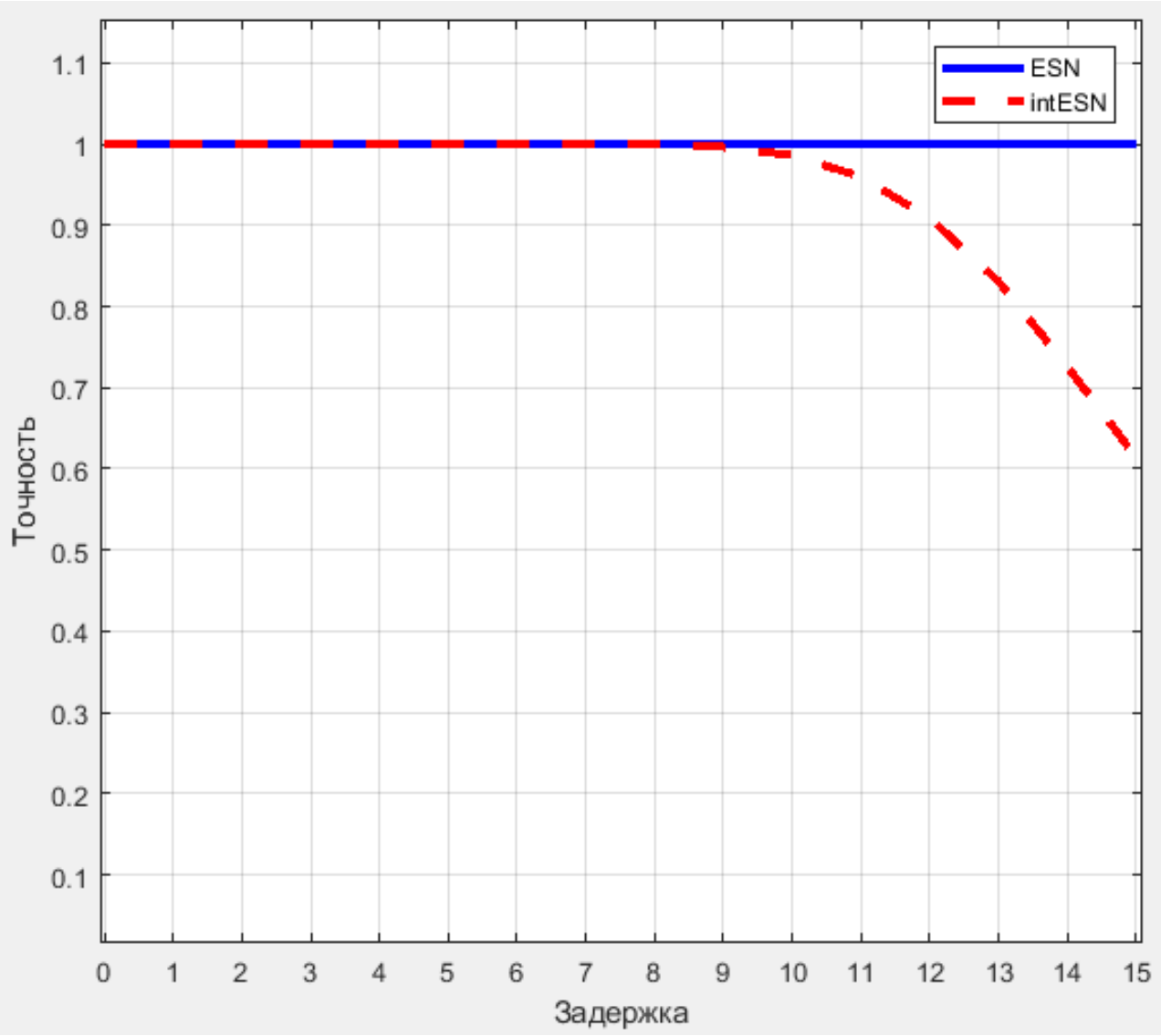

Рисунок 8 – Графики зависимости точности распознавания символа от количества циклов задержки при размере резервуара 1000 нейронов

Однако, при размещении архитектуры в аппаратуре, нейроны традиционной нейронной сети ЭС занимают больше места, следовательно, необходимо рассчитать и сравнить точность расшифровки символов при определенном количестве нейронов, занимающих одинаковый объем памяти и вычислительной логики.

Для выполнения этой задачи на языке описания аппаратуры Verilog были разработаны модуль традиционного нейрона и модуль резервуара целочисленной ЭС. Подсчет занимаемых логических вентилей проводился в

системе разработки Quartus Prime© для ПЛИС серии CycloneIV E производства корпорации Intel [44]. Результат представлен на рисунке 9. Как видно на приведенном графике, объем занимаемых логических вентилей резервуара целочисленной эхо-сети меньше резервуара традиционной на 2-3 порядка при малом количестве нейронов.

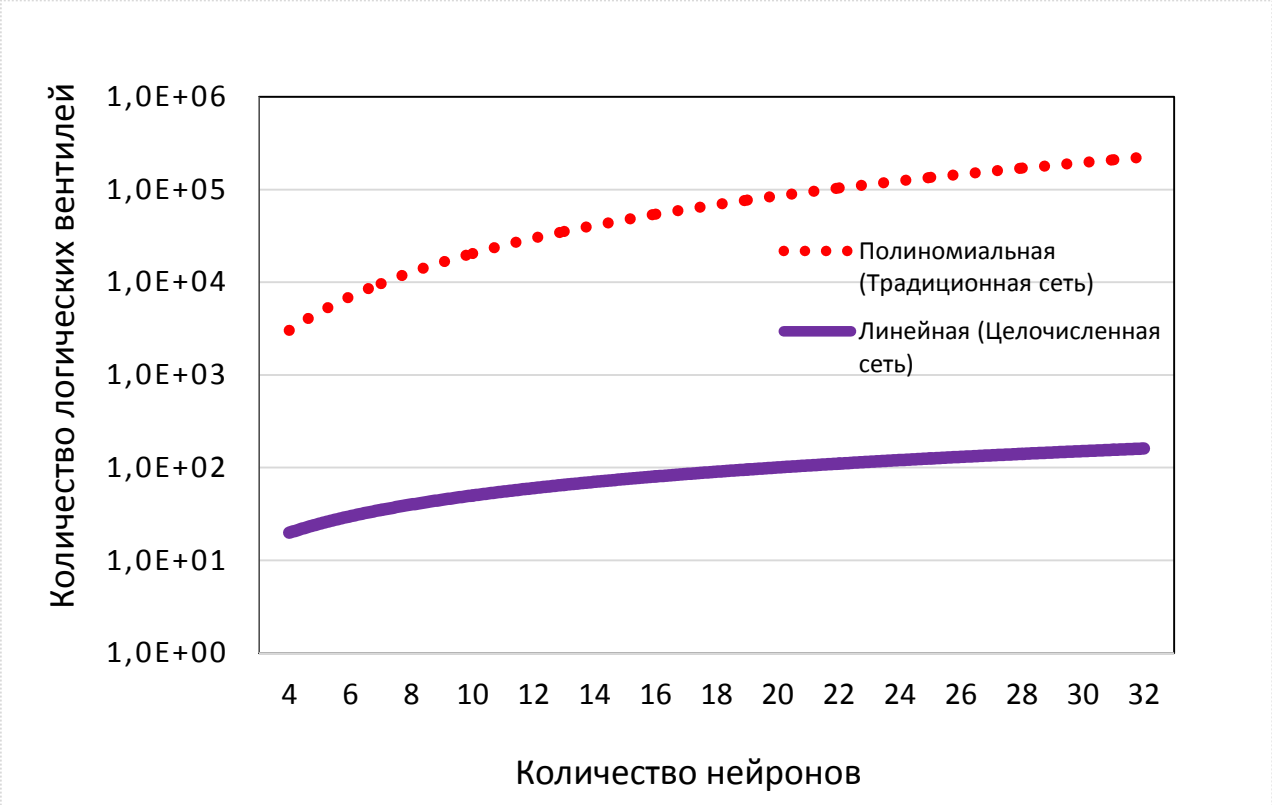

Рисунок 9 – График зависимости количества нейронов резервуара от количества логических вентилей. Точками обозначена традиционная эхо-сеть, сплошной линией целочисленная эхо-сеть

В следующем эксперименте количество нейронов резервуара целочисленной нейронной эхо-сети больше традиционной в 50 раз. При этом резервуар целочисленной эхо-сети занимает меньше места чем резервуар традиционной эхо-сети. Результат эксперимента приведен на рисунке 10.

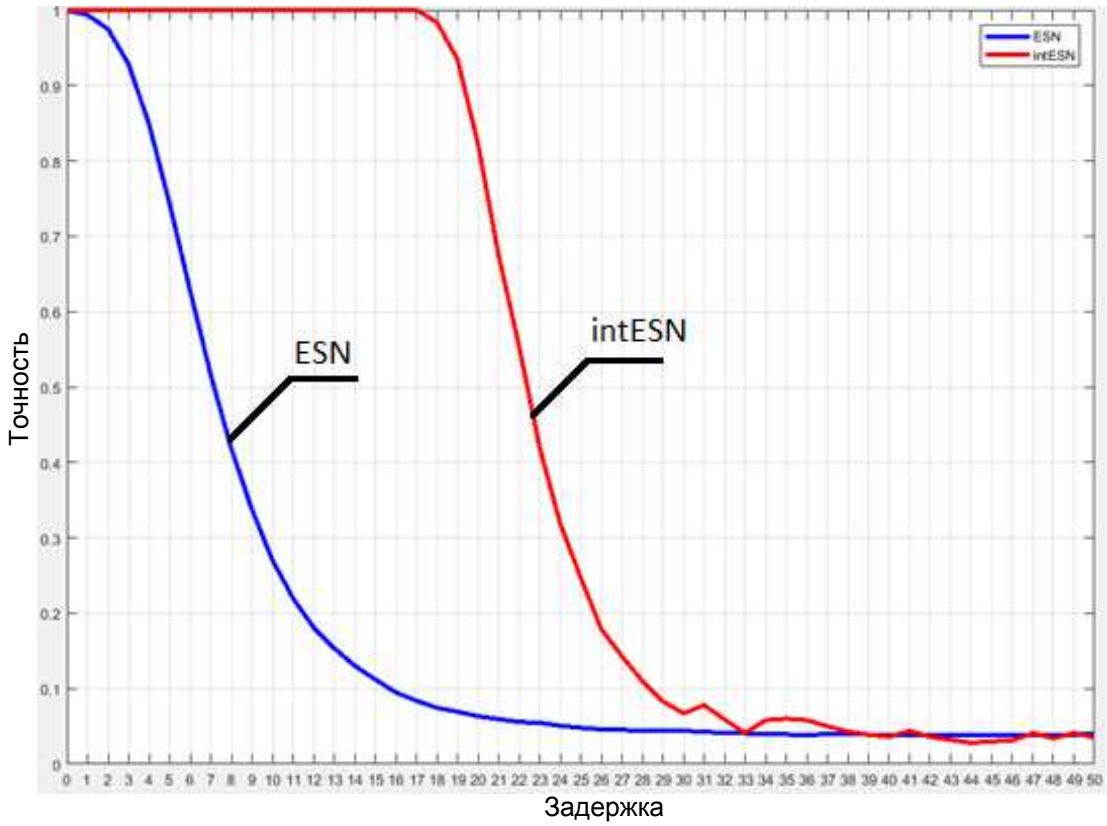

Рисунок 10 – Графики зависимости точности распознавания символа от количества циклов задержки при размере резервуара традиционной эхо-сети 100 нейронов и целочисленной эхо-сети 5000 нейронов

Объем, занимаемый резервуаром традиционной ЭС, возрастает быстрее чем у целочисленной ЭС. Это обуславливается сложностью нормализации весов нейрона. Так как при эксперименте использовалась структура резервуара, при которой каждый из нейронов резервуара связан с каждым из нейронов резервуара, при увеличении количества нейронов в резервуаре, происходит расширение глубины каскада сумматоров, решающих задачу нормализации весов.

В изначальной структуре традиционной ЭС предполагаются случайные связи нейронов внутри резервуара, что ведет к сложностям в аппаратной реализации, так как скорость параллельного вычисления значений нейронов в таком случае будет равна скорости вычисления значения нейрона с наибольшим количеством связей. Также увеличивается сложность разработки

такой сети, так как для каждого нейрона необходимо отдельно разрабатывать структуру работы с разрядностями.

Проблему можно решить, создав регулярную структуру, в которой каждому нейрону соответствует одинаковое количество связей. Следующим экспериментом является проверка изменения работоспособности такой структуры традиционной эхо-сети. На рисунке 11 приведены графики зависимости точности определения входного значения в зависимости от количества итераций для традиционной сети ЭС с различным количеством связей внутри резервуара.

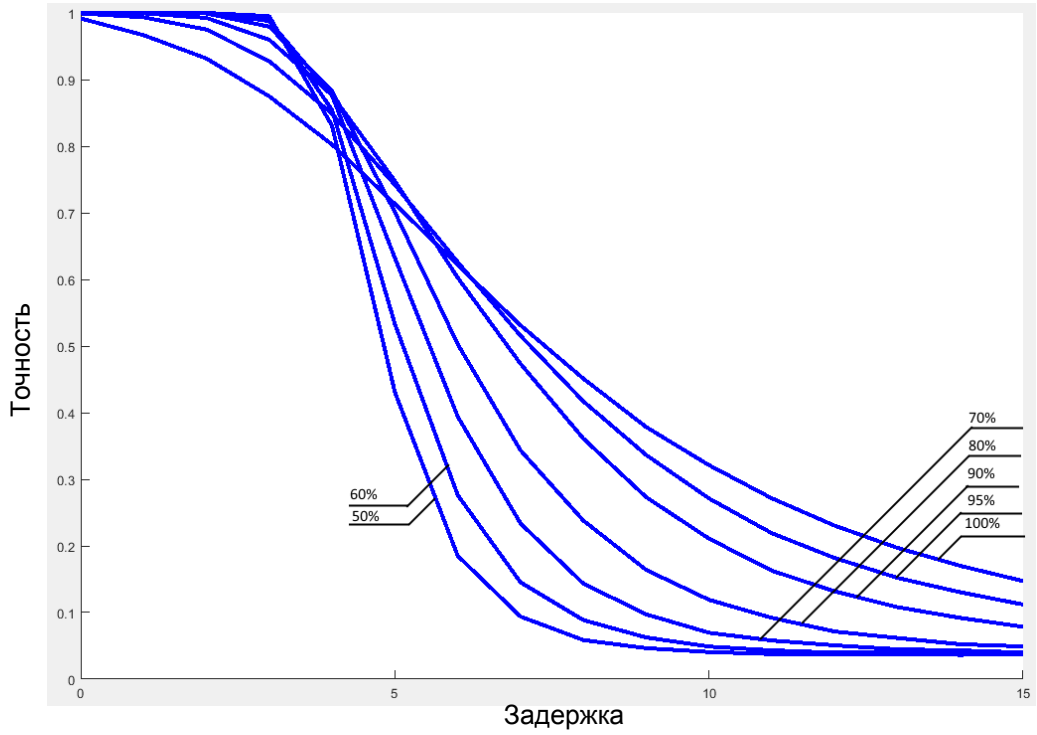

Рисунок 11 - Графики зависимости точности распознавания символа от количества циклов задержки при размере резервуара традиционной эхо-сети 100 нейронов и различным количеством связей внутри резервуара

В данном случае 100% означает полносвязный резервуар, в то время, как 50% – резервуар, в котором каждый нейрон связан с половиной нейронов резервуара.

Из рисунка видно значительное уменьшение точности распознавания входного значения традиционной эхо-сетью при уменьшении количества связей.

3.4 Вывод

Согласно проведенному сравнительному анализу, подтверждена возможность эффективного использования целочисленной сети ЭС. Возможный выигрыш по занимаемым аппаратным ресурсам резервуаром составляет 2-3 порядка уже при минимальном количестве нейронов резервуара, при этом, разница увеличивается при увеличении количества нейронов резервуара, в связи с нелинейным возрастанием аппаратных затрат традиционной сети ЭС. Количество связей внутри резервуара традиционной сети ЭС также увеличивается нелинейно, при этом, уменьшение количества связей резко ухудшает точность распознавания входных символов спустя несколько итераций.

Следующим шагом является разработка и отладка ПЛИС-проекта целочисленной сети ЭС.

4 Реализация и исследование целочисленной сети эхо-состояний на базе программируемой логической интегральной схемы

Разработка целочисленной сети ЭС выполнена в виде параметризируемых сложно-функциональных блоков (СФ-блоков) на языке описания аппаратуры VerilogHDL. Тестирование работоспособности блоков проводилось в программе моделирования Modelsim. Полное описание разработанных блоков приведено в Приложении Б.

Структурная схема модели сети ЭС приведена на рисунке 12. Интерпретатором является выходной слой нейронов, интерпретирующий значение резервуара согласно поставленной задаче. Поскольку модули являются параметризируемыми, объем резервуара, количество нейронов и прочие параметры могут быть изменены. Кроме того, имеется возможность добавления новых слоев. Однако, следует учитывать ресурсы целевого кристалла при конечной реализации.

Ввиду сложности и громоздкости схем СФ-блоков, они приведены в приложении В и на электронном носителе к ВКР.

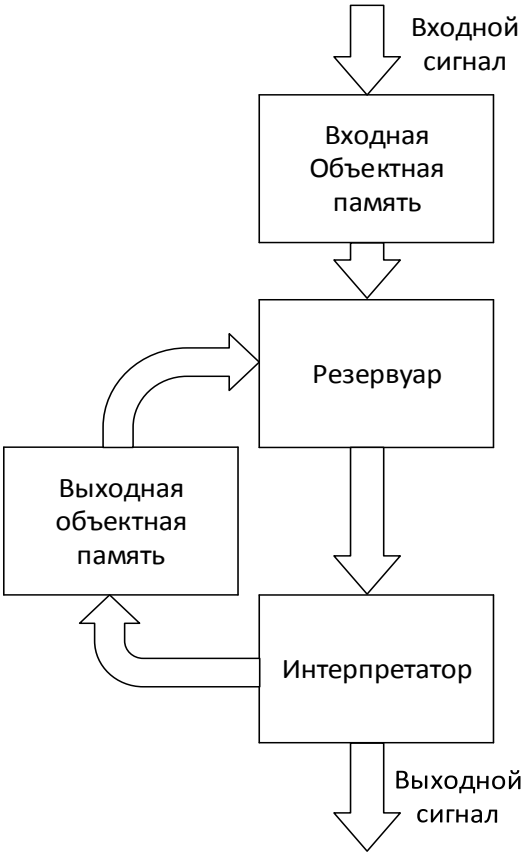

Рисунок 12 – Структурная схема целочисленной сети эхо-состояний

4.1 СФ-блок объектной памяти

Объектная память предназначена для хранения случайного бинарного многомерного вектора для каждого из входных значений. Генерация набора векторов производится при инициализации сети, следовательно, необходимо задать векторы и определить соответствие векторов входным значениям перед компиляцией проекта.

Разрядности входа addr и выхода q зависят от заданных при инициализации сети параметров.

На рисунке 13 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

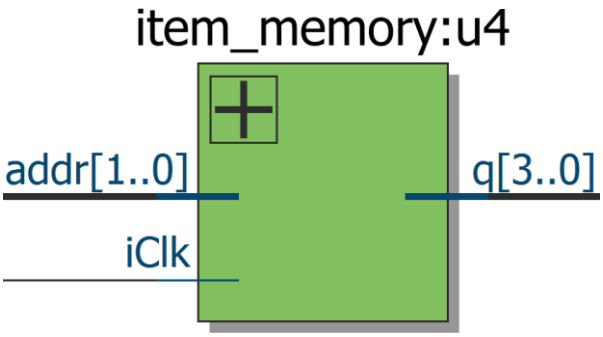

Рисунок 13 – Символьное изображение модуля объектной памяти

Вход addr принимает значение входного сигнала и при нарастающем фронте сигнала iClk выдает на выход q определенный для данного входного значения бинарный вектор.

Инициализация модуля осуществляется занесением в память набора векторов для соответствующих значений. Пример тестирования модуля приведен на рисунке 14.

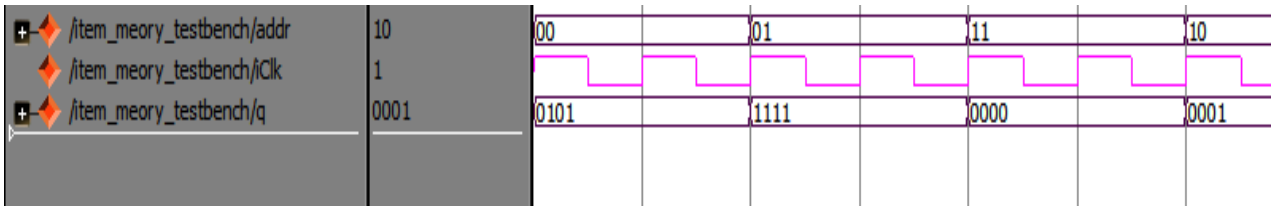

Рисунок 14 – Временная диаграмма работы модуля объектной памяти

4.2 СФ-блок резервуара

Блок резервуара осуществляет функционирование резервуара целочисленной нейронной сети ЭС, в которое входит прибавление значения соответствующего элемента вектора значению нейрона резервуара. Осуществляет операцию циклического сдвига значений нейронов, а также операцию ограничения.

На рисунке 15 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

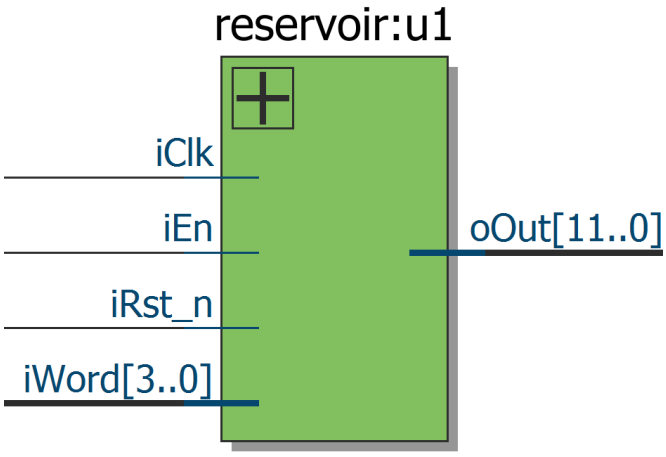

Рисунок 15 – Символьное изображение модуля резервуара

На вход iWord поступает многомерный вектор из объектной памяти, каждый элемент вектора подается на соответствующий нейрон, также на нейрон подается значение выхода соседнего нейрона. Вход iEn отвечает за разрешение счета. Вход iRst_n сбрасывает значения нейронов в ноль. На вход iClk подается сигнал синхронизации. В результате работы на выход oOut поступает вектор значений нейронов резервуара.

В целях уменьшения аппаратных затрат, арифметические операции резервуара проводятся с использованием дополнительного кода.

При разработке модуля было проведено моделирование с различными параметрами и входными значениями. На рисунке 16 приведен пример временной диаграммы работы модуля резервуара.

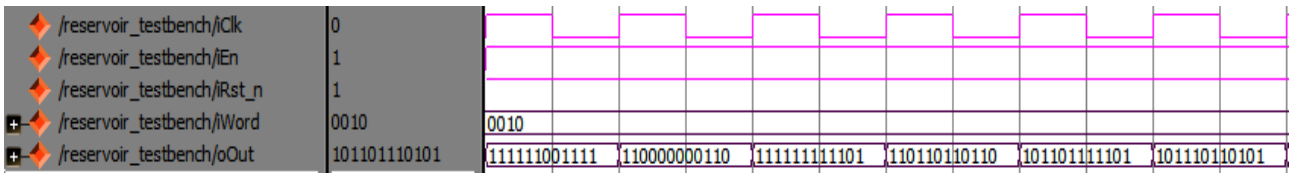

Рисунок 16 – Временная диаграмма работы модуля резервуара

4.3 СФ-блок интерпретатора

Модуль интерпретатора представляет собой классический персептрон. В архитектуре целочисленной нейронной сети ЭС персептроны присутствуют в выходном слое. Модуль интерпретатора включает в себя блок перемножения, блок нормализации, блок функции активации и блок управления.

Символьное представление блока с обозначенными входами и выходами приведено на рисунке 17. На вход модуля подается сигнал синхронизации iClk, сигнал разрешения счета iEn, на вход iData подается набор значений выхода резервуара. Выход oIntRdy генерирует высокий уровень, когда на выход oValue поступает результат работы модуля.

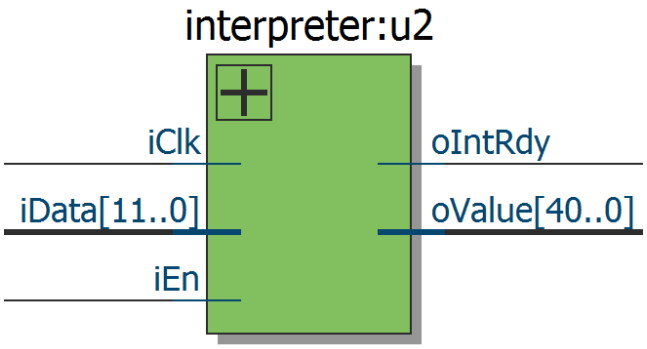

Рисунок 17 – Символьное обозначение модуля интерпретатора

Описание каждого блока, входящего в модуль интерпретатора приведено в пунктах 4.4, 4.5, 4.6. На рисунке 18 приведена временная диаграмма работы модуля при 4-х нейронах резервуара, каждый из которых разрядностью 3 бит, разрядности весовых коэффициентов 4 бит.

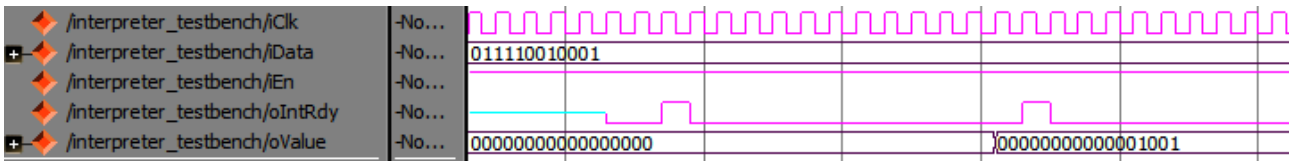

Рисунок 18 – Временная диаграмма работы модуля интерпретатора

4.4 СФ-блок умножителя

Данный модуль входит в состав интерпретатора и осуществляет перемножение набора значений резервуара и набора весовых коэффициентов. Значения весов вводятся при инициализации нейронной сети и не меняются на протяжении работы. Дальнейшим усовершенствованием является реализация возможности перенастройки весовых коэффициентов в ходе работы сети.

На рисунке 19 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

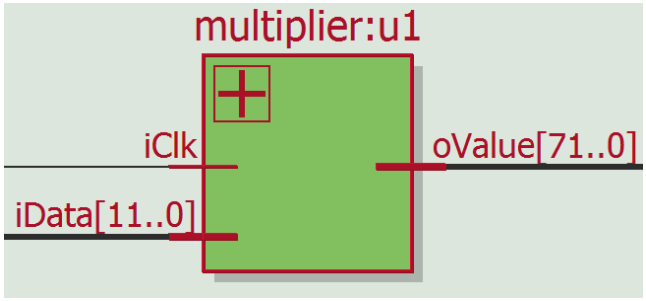

Рисунок 19 – Символьное обозначение блока умножителя

На вход iClk подается синхросигнал, на вход iData подается набор значений резервуара. Значение произведения подается на выход oValue. Разрядность выходного сигнала зависит от количества нейронов в резервуаре, разрядности нейронов резервуара, разрядности весов.

На рисунке 20 приведен пример тестирования модуля умножителя.

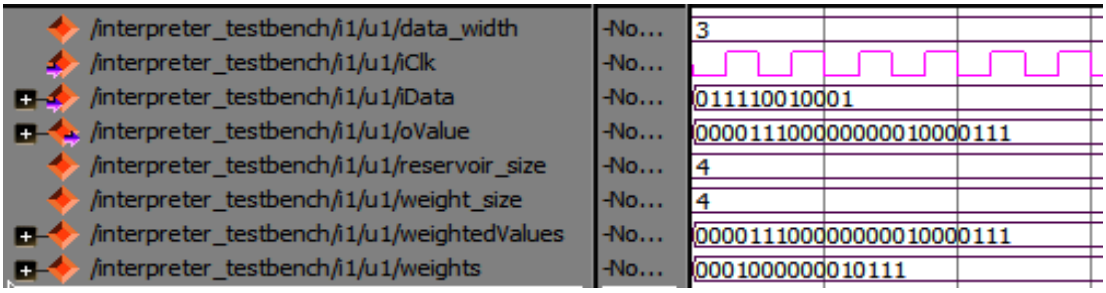

Рисунок 20 – Временная диаграмма работы модуля умножителя

4.5 СФ-блок нормализатора

Данный блок входит в состав интерпретатора и предназначен для сложения всех взвешенных значений резервуара. Реализован в виде каскада сумматоров в целях повышения скорости работы. К сожалению, при такой реализации значительно увеличиваются используемые аппаратные ресурсы. На данном этапе исследования предполагается, что задачи, решаемые при помощи аппаратной реализации сети критичны ко времени, поэтому оптимизация проводилась по времени работы.

Одним из возможных направлений дальнейших исследований является оптимизация занимаемого данным модулем места на кристалле, что неизбежно приведет к замедлению работы модуля. Необходимо выяснить, какой из параметров оптимизации важнее при решении реальных задач реального времени.

На рисунке 21 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

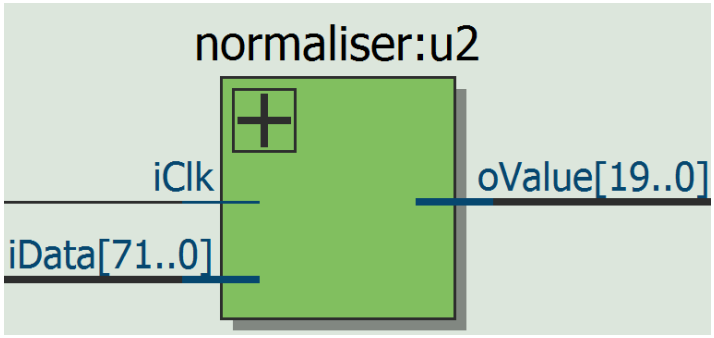

Рисунок 21 – Символьное обозначение блока умножителя

На рисунке 22 приведен пример тестирования модуля умножителя. Параметр demention обозначает разрядность слагаемых. Сложение происходит в прямом коде. Сложение происходит в один такт, в связи с использованием каскада сумматоров.

Данное решение нерационально по количеству занимаемого места на кристалле. Также возможны проблемы при повышении тактовой частоты. Дальнейшее направление исследований может заключаться в тестировании модуля на реальной задаче, с целью определения необходимой скорости срабатывания.

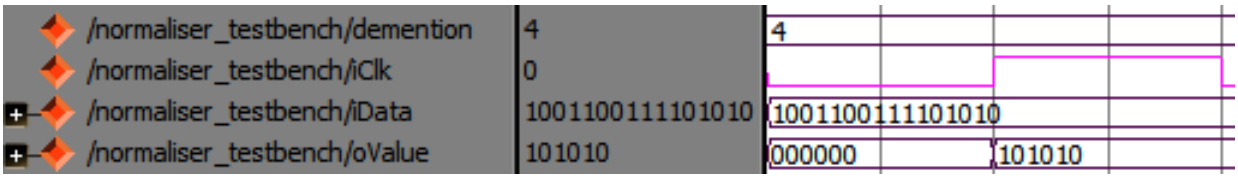

Рисунок 22 – Временная диаграмма работы модуля умножителя

4.6 СФ-блок функции активации

Данный блок входит в состав модуля интерпретатора и предназначен для вычисления функции активации. В архитектуре целочисленной сети ЭС данный блок не используется, так как нелинейность сети достигается посредствам операции ограничения на этапе обработки информации в резервуаре. Однако, при необходимости использования в архитектуре сети классических нейронов, необходимо включить данный блок в состав интерпретатора.

Функция активации, вычисляемая в данном СФ-блоке, приведена в формуле 6.

$$y = x/(0{,}5 + |x|), \tag{5}$$

где y – входное значение блока, на рисунке 23 обозначено как iData;

x – выходное значение блока, на рисунке 23 обозначено как oData.

Приведенная выше функция активации выбрана как одна из менее затратных с точки зрения аппаратной реализации.

На рисунке 23 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

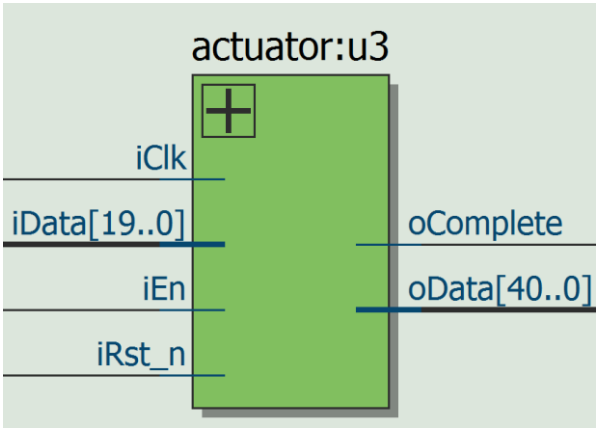

Рисунок 23 – Символьное обозначение модуля функции активации

Проведено тестирование модуля. Пример временной диаграммы приведен на рисунке 24.

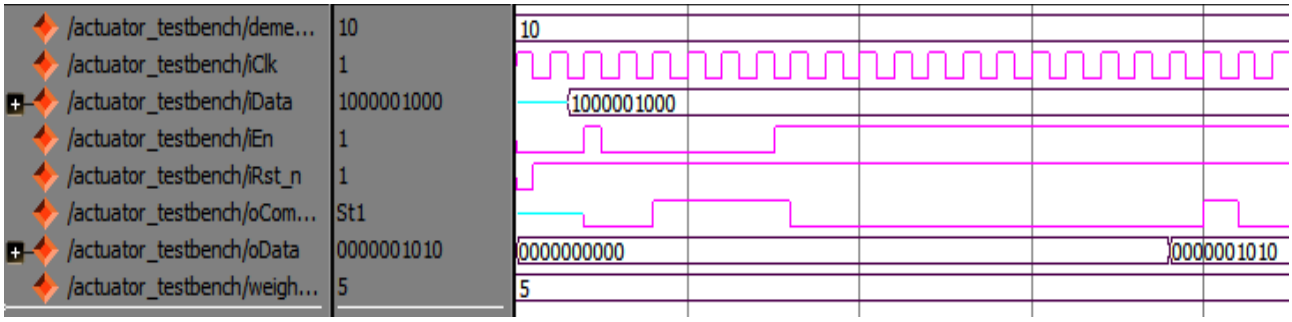

Рисунок 24 – Временная диаграмма работы модуля функции активации

4.7 СФ-блок управления

Данный модуль предназначен для управления функционированием сети. Представляет собой управляющий автомат.

На рисунке 25 приведено символьное изображение, а также обозначение входов и выходов СФ-блока.

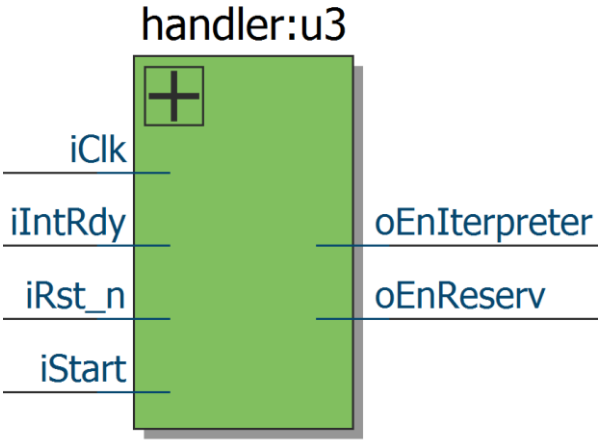

Рисунок 25 – Символьное обозначение модуля функции активации

4.8 Вывод

Разработан комплект параметризируемых, масштабируемых СФ-блоков целочисленной нейронной сети ЭС. Проведено тестирование разработанных блоков в среде моделирования ModelSim.

По результатам моделирования сделан ряд выводов. Предложенная архитектура модуля нормализатора оптимизирована по времени работы, при этом крайне неоптимизирована по занимаемому на кристалле месту. Благодаря такой архитектуре скорость работы целочисленной нейронной сети составляет порядка 3-4 тактов. Возможная частота тактового сигнала, а также возможность решения задач управления в реальном времени нуждаются в дальнейших исследованиях.

ЗАКЛЮЧЕНИЕ

1. Проведены исследования способов и методов резервуарных вычислений в классе рекуррентных нейронных сетей. В целях управления целевым объектом в динамической среде в реальном времени, рассмотрен подход аппаратной реализации рекуррентных нейронных сетей.

2. Предложен метод эффективной реализации нейронных сетей ЭС на аппаратном уровне, основанный на принципе многомерных вычислений. Особенностью вычислений, проводимых по принципу многомерных вычислений состоит в том, что все сущности (объекты, фонемы, символы) представлены случайными векторами очень большой размерности – несколько тысяч бит. Вычисления сетей ЭС реализованы простыми арифметическими операциями (сложение/объединение, перестановки). Уменьшена разрядность вычислений резервуара, уменьшено количество соединений, увеличено быстродействие сети. Составлена архитектура целочисленной сети ЭС, по принципу функционирования схожей с традиционной сетью ЭС, однако более компактную с точки зрения аппаратной реализации.

3. Составлены модели сетей ЭС. Проведены эксперименты, показывающие способность целочисленных сетей ЭС справляться с задачами традиционных сетей ЭС. Проведен сравнительный анализ занимаемого на кристалле места для традиционной и предложенной моделей. Согласно анализу, скрытый слой предложенной архитектуры может занимать, в зависимости от размеров сети, в десятки или даже сотни раз меньше места на кристалле.

4. Проведена реализация набора модулей целочисленной сети ЭС в виде параметризируемых сложно-функциональных блоков на языке описания аппаратуры VerilogHDL. Разработанный комплект модулей позволяет простое масштабирование и конфигурацию сети для различных задач.

СПИСОК СОКРАЩЕНИЙ

РВ	–	Резервуарные вычисления
РНС	–	Рекуррентные нейронные сети
СФ-блок	–	Сложно-функциональный блок
Сеть ЭС	–	Сеть Эхо-состояний

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Lukosevicius, M. Reservoir computing approaches to recurrent neural network training / M. Lukosevicius, H. Jaeger // Computer Science Review. – 2009. – Vol. 3. – № 3. – P. 127–149.

2. Sussillo, D. Generating coherent patterns of activity from chaotic neural networks / D. Sussillo, L. Abbott // Neuron. – 2009. – Vol. 63. – № 4. – P. 544–557.

3. Rastegari, M. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks / M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi // European Conference on Computer Vision. ¬ 2016. – Vol. 9908. – P. 525–542.

4. Courbariaux, M. Binarized neural networks: Training neural networks with weights and activations constrained to +1 or -1 / M. Courbariauxa, I. Hubara, D. Soudry, R. El-Yaniv, Y. Bengio // arXiv:1602.02830. – 2016. – P. 1–11.

5. Hubara, I. Quantized neural networks: Training neural networks with low precision weights and activations / I. Hubara, M. Courbariauxa, D. Soudry, R. El-Yaniv, Y. Bengio // arXiv:1609.07061. – 2016. – P. 1–29.

6. Frady, E.P. A theory of sequence indexing and working memory in recurrent neural networks / E.P. Frady, D. Kleyko, F.T. Sommer // Neural Computation. – 2018. – Vol. 0. – № 0. – P. 1–65.

7. Kanerva, P. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors // Cognitive Computation. – 2009. – Vol. 1. – № 2. – P. 139–159.

8. Javier, G. Frame-by-frame language identification in short utterances using deep neural networks. / G.D. Javier, L.M. Ignacio, J.M. Pedro, J. Gonzalez-Rodriguez // Neural Networks. – 2015. – Vol. 64. – P. 49–58.

9. Golovin, D. Google Vizier: A Service for Black-Box Optimization. / D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, D. Sculley // Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. – 2017. – P. 1487–1495.

10. Frances-Villora, J.V. Hardware implementation of real-time Extreme Learning Machine in FPGA: Analysis of precision, resource occupation and performance. / J.V. Frances-Villora, A. Rosado-Muñoz, J.M. Martínez-Villena, M. Bataller-Mompean, J.F. Guerrero, M. Wegrzyn // Computers & Electrical Engineering. – 2016. – Vol. 51. – P. 139–156.

11. Chang, J.R. Neuroscience-inspired recurrent network for object recognition. / J.R. Chang, P.C. Kuo, Y.S. Chen // 2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS). – 2017. – P.729–734.

12. Abraham, A. Artificial Neural Networks. // Handbook of Measuring System Design. – 2005. – P. 901–908.

13. Ku, C.C., Diagonal Recurrent Neural Networks for Dynamic Systems Control. / C.C. Ku, K. Y. Lee // IEEE Transactions on Neural Networks. – 1995. – Vol. 6. – № 1. – P. 144–156.

14. Gayler, R. Vector Symbolic Architectures Answer Jackendoff ’s Challenges for Cognitive Neuroscience. // Proceedings of the Joint International Conference on Cognitive Science. ICCS/ASCS. – 2003. – P. 133–138.

15. Hopfield, J.J. Neurons with graded response have collective computational properties like those of two-state neurons. // Proceedings of the National Academy of Sciences. – 1984. – Vol. 81. – P. 3088–3092.

16. Hopfield, J.J. Neural networks andphysical systems with emergent collective computational abilities // Proceedings of National Academy of Sciences of United States of America. – 1982. – Vol. 79. – P. 2554–2558.

17. Geoffrey, E. Hinton Boltzmann machine // Scholarpedia. – 2007. – Vol. 2. – P. 1–7.

18. Ackley, D.H. A learning algorithm for Boltzmann machines / H. D. Ackley, G.E. Hinton, T.J. Sejnowski // Cognitive Science. – 1985. – Vol. 9. – P. 145–169.

19. Hinton, G.E. Redusing the dimensionality of data with neural networks / G.E. Hinton, R. Salakhudinov // Science. – 2006. – Vol. 313. – P. 504–507.

20. Mohiuddin, K.M. Artificial Neural Networks: A Tutorial. / K.M. Mohiuddin, J. Mao, A.K. Jain // IEEE Computer Society. – 1996. – Vol. 29. – P. 31–44.

21. Nutzel, K. The length of attractors in asymmetric random neural networks with deterministic dynamics. // Journal of Physics A: Mathematical and General. – 1996. – Vol. 24. – №3. – P. 47–55.

22. Pascanu, R. On the difficulty of training recurrent neural networks. / R. Pascanu, T. Mikolov, Y. Bengio // Conference on machine learning. – 2013. – P. 1–12.

23. Funahashi, K. Approximation of dinamical systems by continious time reccurent neural networks. / K. Funahashi, Y. Nakamura // Neural Networks. – 1993. – Vol. 6. – P. 801–806.

24. Doya, K. Burification in the learning of reccurent neural networks. // Proceedings of IEEE International Symposium on Circuits and System. – 1992. – Vol. 6. – P. 2777–2780.

25. Bengio, Y. Lerning long-term dependencies with gradient descent is difficult. / Y. Bengio, P. Simard, P. Frasconi // IEEE Transactions on Neural Networks. – 1994. – Vol. 5. – P. 157–166.

26. Maass, W. Real-time computing without stable states: A new framework for neural computation based on pertubations. / W. Maass, T. Natschlager, H. Markram // Neural Computation. – 2002. – Vol. 14. – P. 2531–2560.

27. Jaeger, H. The “echo state” approach to analysing and training reccurent neural networks // Technical Report GMD Report 148, German National Research Center for Informational Technology. – 2001. – P.1–47.

28. Jaeger, H. Echo state network. // Scholarpedia. – 2007. – Vol. 2. – P. 2330–2356.

29. Jaeger, H. Optimization and applications of echo state networks with leaky-integrator neurons. / H. Jaeger, M. Lukosevicius, D. Popovici, U. Siewert // Neural Networks. – 2007. – Vol. 20. – P. 335–352.

30. Lukosevicius, M. A Practical Guide to Applying Echo State Networks. // Neural Networks: Tricks of the Trade, ser. Lecture Notes in Computer Science. – 2012. – Vol. 7700. – P. 659–686.

31. Plate, T. Holographic reduced representations. // IEEE Transactions on Neural Networks. – 1995. – Vol. 6. – №. 3. – P. 623–641.

32. Gallant, S. I. Representing objects, relations, and sequences. / S.I. Gallant, T.W. Okaywe // Neural Computation. – 2013. – Vol. 25. – № 8. – P. 2038–2078.

33. Gayler, R. Multiplicative binding, representation operators & analogy. / R. Gayler, D. Gentner, K. J. Holyoak, B. N. Kokinov // Advances in analogy research: Integration of theory and data from the cognitive, computational, and neural sciences, New Bulgarian University, Sofia, Bulgaria. – 1998. – P. 1–4.

34. Gallant, S. I. Positional binding with distributed representations. / S.I. Gallant, P. Culliton // International Conference on Image, Vision and Computing (ICIVC). – 2016. – P. 108–113.

35. Plate, T. Holographic reduced representations: Distributed representation for cognitive structures. // Center for the Study of Language and Information (CSLI). – 2003. – P.1–21.

36. Kanerva, P. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. // Cognitive Computation. – 2009. – Vol. 1. – № 2. – P. 139–159.

37. Rachkovskij, D.A. Representation and Processing of Structures with Binary Sparse Distributed Codes. // IEEE Transactions on Knowledge and Data Engineering. – 2001. – Vol. 3. – № 2. – P. 261–276.

38. Rahimi, A. Hyperdimensional biosignal processing: A case study for emg-based hand gesture recognition. / A. Rahimi, S. Benatti, P. Kanerva, L. Benini, J. M. Rabaey // IEEE International Conference on Rebooting Computing (ICRC). – 2016. – P. 1–8.

39. Kanerva, P. Sparse Distributed Memory. // The MIT Press. – 1988. – P.1–27.

40. Rachkovskij, D. A. Sparse Binary Distributed Encoding of Scalars. / D.A. Rachkovskij, S.V. Slipchenko, E.M. Kussul, T.N. Baidyk // Journal of Automation and Information Sciences. – 2005. – Vol. 37. – № 6. – P. 12–23.

41. Widdows, D. Reasoning with vectors: A continuous model for fast robust inference. / D. Widdows, T. Cohen // Logic Journal of the IGPL. – 2015. – Vol. 23. – № 2. – P. 141–173.

42. Wang, H. Reccurent Neural Networks: Associative Memory and Optimization. / H. Wang, Y. Wu, B. Zhang, K.L. Du // Information Technology & Software Engineering – 2019 – Vol. 1. – Issue 2 – P. 1–15.

43. Rahimi, A. High-dimensional computing as a nanoscalable paradigm. / A. Rahimi, S. Datta, D. Kleyko, E. P. Frady, B. Olshausen, P. Kanerva, J. M. Rabaey // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2017. – Vol. 64. – Issue 9 – P. 2508–2521.

44. Cyclone IV E series [Электронный ресурс]: Официальный сайт Altera. – Режим доступа: https://www.intel.ru/content/dam/www/programmable/us/en/pdfs/literature/pt/cyclone-iv-product-table.pdf. (Дата посещения: 27.06.2019)

ПРИЛОЖЕНИЕ А

Программный код моделей эхо-сетей

Файл Start_simulation.m

```
ESN_tokens_orthogonal; %simulation for ESN
intESN_tokens;  %simulation for intESN

%Update values for figure
box on;
grid on;
xlabel('Задержка')
xticks(0:laten)
ylabel('Точность')
legend({'ESN', 'intESN'})
```

Файл ESN_initialization.m

```
function [Wвх,W] = ESN_initialization(rho,scaling, N, K)

%Inputs
%rho- feedback strength of the reservoir connection matrix
% scaling  %projection gain.
%N number of neurons in reservoir.
%K - number of inputs
%Outputs
% W - reservoir connection matrix
% Wвх - input projection matrix


Wвх=scaling*(-1+2*rand(N,K));% random matrix for projection from input to
reservoir
W=randn(N); % random matrix
[W,~] = qr(W); % random orthgonal matrix for reservoir connections
W=rho* W; % this is the final reservoir matrix which will be used later. It is
scaled by the feedback strength
end
```

Файл ESN_tokens_orthogonal

```
clear all
%Set parameters for the simulation
simul=50; % number of independent simulations to run
Dic=27; %dictionary size of tokens
Len=3000; %total length of the sequence
T=Len-1000;% length of the training sequence
Tcut=500; %cut off first steps for training
laten=50;%Size of delay to learn
N=100; % number of units in reservoir
K=Dic;% number of input neurons
L=Dic; %number of outpur neurons
rho=0.94; % feedback strength of the reservoir connection matrix
scaling=0.1; %projection gain.
```

```
acc=zeros(laten+1,simul); % store the the descoding accuracy for each delay and
each simulation

for sim=1:simul
    disp(sim)
    [Wbx,W] = ESN_initialization(rho,scaling, N, K); % initialize ESN random
parameters
    TRACE=randi([1,Dic],1,Len,'single'); %randomly generate a sequence of tokens
to write to the network

    U=zeros(K,Len); %  create one hot encoding for TRACE
    for i=1:Len
        U(TRACE(1,i),i)=1;
    end

    %% Start training
    X=zeros(N,T); %it will store values of reservoir at each time step
    X(:,1)=tanh(Wbx*U(:,1)); %compute the reservoir state at the first step
    for n=2:T %compute the reservoir states for steps up to T
        X(:,n)=tanh(W*X(:,n-1)+Wbx*U(:,n)); %compute the reservoir state at the
current step
    end

    % Learn matrix from reservoir to the outpur  by linear regression
    M=X(:,Tcut+1:T)'; % reservoir states from step Tcut+1 to T
    WOUT=cell(1,laten+1); % it stores the readout matrices for each delay step
    for i=0:laten
        Td= U(:,Tcut+1-i:T-i)'; % correct answers from the ground truth
        Wout=(pinv(M)*Td)'; %solving linear regression by pseudo-inverse matrix
        WOUT{1,i+1}=Wout; % store the trained readout matrix
    end

    % END of training
    %% Operational phase
    Yh=zeros(laten+1,Len); %output values produced by the reservoir
    Y=zeros(laten+1,Len); %output values from the true data
    Xo=zeros(N,Len); %it will store values of reservoir at each time step

    for n=T+1:Len %for all steps from T to Len
        if n~=T+1
            Xo(:,n)=tanh(W*Xo(:,n-1)+Wbx*U(:,n));  %compute the reservoir state
at the current step
        else % special case at the first step feed with the ground truth state
of the reservoir
            Xo(:,n)=tanh(W*X(:,T)+Wbx*U(:,n) );
        end
        %Decoding from the reservoir
        for i=0:laten %for all delays
            [~,Yh(i+1,n)]=max(WOUT{1,i+1}*Xo(:,n)); % estimate the output using
the learned matrix Wout
            Y(i+1,n)= TRACE(1,n-i); %get the true value from TRACE
        end
    end

    Yh=Yh(:,T+1:Len); %cut to the length of testing data
    Y=Y(:,T+1:Len); %cut to the length of testing data
    acc(:,sim)=sum(Y==Yh,2)/(Len-T); %calculate the decoding accuracy from the
current simulation
```

```
end
acc_m=mean(acc,2);% Take mean values across all simulations
%plot results
figure(1)
hold on
plot(0:laten,acc_m, 'b', 'LineWidth', 3 )
```

Файл intESN_initialization.m

```
function [HD] = intESN_initialization(N, K)

%Inputs
%N number of neurons in reservoir.
%K - number of inputs
%Outputs
% HD - input projection matrix for tokens

%create HD vector for mapping
HD=2*(-0.5+  randi([0,1],K,N,'single')); %random bipolar vectors
end
```

Файл intESN_tokens.m

```
clear all
%Set parameters for the simulation
simul=50; % number of independent simulations to run
Dic=27; %dictionary size of tokens
Len=3000; %total length of the sequence
T=Len-1000;% length of the training sequence
Tcut=500; %cut off first steps for training
laten=50;%Size of delay to learn
N=100; % number of units in reservoir
K=Dic;% number of input neurons
L=Dic; %number of outpur neurons
thr=3; %clipping threshold for resrvoir

acc=zeros(laten+1,simul); % store the the descoding accuracy for each delay and
each simulation

for sim=1:simul
    disp(sim)
    [HD] = intESN_initialization(N, K); % initialize intESN random parameters

    TRACE=randi([1,Dic],1,Len,'single'); %randomly generate a sequence of tokens
to write to the network
    U=zeros(K,Len); %  create one hot encoding for TRACE
    for i=1:Len
        U(TRACE(1,i),i)=1;
    end

    %% Start training

    X=zeros(N,T); %it will store values of reservoir at each time step
    X(:,1)=HD(TRACE(1,1),:)'; %compute the reservoir state at the first step.
%do not need clippping as it is the first vector an it is in a linear range
    for n=2:T %compute the reservoir states for steps up to T
        HGN=circshift(X(:,n-1), [1,0])+HD(TRACE(1,n),:)'; %update reservoir
        %Clipping
        HGN(HGN>thr)=thr; %clip reservoir value above threshold
```

```
        HGN(HGN<-thr)=-thr; %clip reservoir value below threshold
        X(:,n)=HGN; % store the current resevoir state
    end

    % Learn matrix from reservoir to the outpur  by linear regression
    M=X(:,Tcut+1:T)'; % reservoir states from step Tcut+1 to T
    WOUT=cell(1,laten+1); % it stores the readout matrices for each delay step
    for i=0:laten
        Td= U(:,Tcut+1-i:T-i)'; % correct answers from the ground truth
        Wout=(pinv(M)*Td)'; %solving linear regression by pseudo-inverse matrix
        WOUT{1,i+1}=Wout; % store the trained readout matrix
    end
    % END of training
    %% Operational phase
    Yh=zeros(laten+1,Len); %output values produced by the reservoir
    Y=zeros(laten+1,Len); %output values from the true data
    Xo=zeros(N,Len); %it will store values of reservoir at each time step
    for n=T+1:Len %for all steps from T to Len
        if n~=T+1
            HGN=circshift(Xo(:,n-1), [1,0])+HD(TRACE(1,n),:)'; %compute the
reservoir state at the current step
        else % special case at the first step feed with the ground truth state
of the reservoir
            HGN=circshift(X(:,n-1), [1,0])+HD(TRACE(1,n),:)';
        end
        %Clipping
        HGN(HGN>thr)=thr; %clip reservoir value above threshold
        HGN(HGN<-thr)=-thr; %clip reservoir value below threshold
        Xo(:,n)=HGN; % store the current resevoir state

        %Decoding from the reservoir
        for i=0:laten %for all delays
            [~,Yh(i+1,n)]=max(WOUT{1,i+1}*Xo(:,n)); % estimate the output using
the learned matrix Wout
            Y(i+1,n)= TRACE(1,n-i); %get the true value from TRACE
        end

    end

    Yh=Yh(:,T+1:Len); %cut to the length of testing data
    Y=Y(:,T+1:Len); %cut to the length of testing data
    acc(:,sim)=sum(Y==Yh,2)/(Len-T); %calculate the decoding accuracy from the
current simulation
end
acc_m=mean(acc,2);% Take mean values across all simulations

%plot results
figure(1)
hold on
plot(0:laten,acc_m,'--r', 'LineWidth', 3 )
```

ПРИЛОЖЕНИЕ Б

Фрагмент разработанного ПЛИС-проекта

В приложении приведены заголовки основных СФ-блоков, большая часть кода приведена на CD-диске.

Топ-модуль проекта:

```
module intESN(
iClk,
iItem,
iEn,
iRst_n,
iStart,
oValue
);

parameter		reservoir_size = 4;
parameter		data_width = 3;
parameter		weight_size = 16;
parameter		idata_demention = 2;
parameter		layer = 1;

input iClk;
input iEn;
input [idata_demention-1:0]iItem;
input iRst_n;
input iStart;
output [(data_width+weight_size+layer)*2:0] oValue;
.
.
.
Endmodule
```

Модуль reservoir:

```
module reservoir
(
        iClk,
        iWord,
        //iY,
        iRst_n,
        iEn,
        oOut
);
parameter               reservoir_size = 4;
parameter               data_width = 3;

input iClk;
input iRst_n;
input iEn;
input [reservoir_size-1:0] iWord;
//input [reservoir_size*2-1:0] iY;

output [(data_width*reservoir_size-1):0]oOut;
.
.
.
Endmodule
```

Модуль interpreter:

```
module interpreter
(
        iClk,
        iEn,
        iData,
        oValue,
        oIntRdy,
```

```
        iRst_n
);

parameter           layer = 1;
parameter           data_width = 3;
parameter           weight_size = 4;
parameter           reservoir_size = 4;
parameter           WAIT            = 4'b0001,
                    MULTIPLIER      = 4'b0010,
                    SUMMARISE       = 4'b0100,
                    ACTIVATION      = 4'b1000;
input       iClk;
input               iRst_n;
input               iEn;
input               [reservoir_size*data_width-1:0]     iData;
output      [(data_width+weight_size+layer)*2:0] oValue;
output      oIntRdy;
.
.
.
Endmodule
```

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
«04» 07 2019 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Метод аппаратной реализации рекуррентных нейронных сетей
тема

09.04.01 «Информатика и вычислительная техника»
код и наименование направления

09.04.01.06 «Микропроцессорные системы»
код и наименование магистерской программы

Научный руководитель	_____ подпись, дата 02.07.19	проф., зав. каф. ВТ, канд. техн. наук должность, ученая степень	О.В.Непомнящий инициалы, фамилия
Выпускник	_____ 02.07.19 подпись, дата		А.Г.Хантимиров инициалы, фамилия
Рецензент	_____ подпись, дата 02.07.19	Professor, PhD Luleå tekniska universitet должность, ученая степень	Evgeny Osipov инициалы, фамилия
Нормоконтроллер	_____ 03.07.19 подпись, дата		А.И.Постников инициалы, фамилия

Красноярск 2019