УДК 004.4’24, 004.3, 004.89, 004.942

А.Ю. Дорошенко, В.М. Шимкович, Т.А. Мамедов, О.А. Яценко

АВТОМАТИЗОВАНЕ ПРОЄКТУВАННЯ ШТУЧНОГО НЕЙРОНА ДЛЯ ПРОГРАМОВАНИХ ЛОГІЧНИХ ІНТЕГРАЛЬНИХ СХЕМ НА ОСНОВІ АЛГЕБРО-АЛГОРИТМІЧНОГО ПІДХОДУ

Дорошенко Анатолій Юхимович

Інститут програмних систем НАН України, м. Київ,

doroshenkoanatoliy2@gmail.com

Шимкович Володимир Миколайович

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»,

shymkovych.volodymyr@gmail.com

Мамедов Турал Алірзайович

Інститут програмних систем НАН України, м. Київ,

tural.mamedov1@gmail.com

Яценко Олена Анатоліївна

Інститут програмних систем НАН України, м. Київ,

oayat@ukr.net

Нейромережеві системи керування є високотехнологічним напрямком теорії керування та відносяться до класу нелінійних динамічних систем. Висока швидкодія за рахунок розпаралелювання вхідної інформації в поєднанні зі здатністю до навчання нейронних мереж робить цю технологію вельми привабливою для створення пристроїв керування в автоматичних системах. Забезпечення швидкодії мереж у реальному часі здійснюється шляхом їх реалізації на програмованих логічних інтегральних схемах (ПЛІС). Одним із прикладів апаратної реалізації нейронних мереж є проєктування штучного нейрона та його нелінійних функцій активації. Технологія розробки додатків для ПЛІС ґрунтується на поданні алгоритму мовою опису апаратури, наприклад VHDL, і автоматичному перекладі цього опису в специфікацію на рівні логічних таблиць та інших функціональних компонентів інтегральних схем. Програмування мовою VHDL досить складне, тому постає питання про розробку спеціальних засобів автоматизації, які дозволили б ефективно генерувати високопродуктивний програмний код. У статті запропоновано засоби автоматизованого проєктування та генерації програм для ПЛІС, що ґрунтуються на алгебрі алгоритмів. Створені засоби застосовано для проєктування штучного нейрона. Розроблено метод конструювання штучного нейрона з сигмоїдальною функцією активації на ПЛІС, який відрізняється від аналогічних підходів тим, що коефіцієнти кусково-лінійної апроксимації функції активації зберігаються в пам’яті лише для

додатних або лише для від'ємних значень аргументів. Це дозволило оптимізувати кількість використовуваних обчислювальних ресурсів і підвищити продуктивність нейронної мережі. Даний підхід застосовано для розробки системи з нейромережевим контролером для балансування кульки на платформі, реалізованим на ПЛІС.

Ключові слова: автоматизоване проєктування, алгебра алгоритмів, апроксимаційні обчислення, генерація програм, нейронна мережа, програмовані логічні інтегральні схеми, система керування.

Вступ

Керування сучасними технічними системами вимагає розробки нових методів керування, оскільки модифікація та вдосконалення традиційних підходів не завжди забезпечує виконання жорстких вимог до показників якості. Нейромережеві системи керування є високотехнологічним напрямком теорії керування та відносяться до класу нелінійних динамічних систем. Висока швидкодія за рахунок розпаралелювання вхідної інформації в поєднанні зі здатністю до навчання нейронних мереж робить цю технологію вельми привабливою для створення пристроїв керування в автоматичних системах. Забезпечення швидкодії мереж в реальному часі здійснюється шляхом їх реалізації на програмованих логічних інтегральних схемах (ПЛІС).

Одним із прикладів апаратної реалізації нейронних мереж є реалізація штучного нейрона та його нелінійних функцій активації. Існуючі підходи до реалізації нелінійних функцій використовують різні методи апроксимації [1–3], такі як ряд Тейлора, табличний метод, кусково-лінійна апроксимація тощо. Ряди Тейлора потребують численних операцій множення і тому не є оптимальними для реалізації на ПЛІС, оскільки блок множення займає багато ресурсів. Тому найкращою для реалізації нелінійних функцій активації є кусково-лінійна апроксимація.

Технологія розробки додатків для ПЛІС грунтується на поданні алгоритму мовою опису апаратури, наприклад VHDL [4], і автоматичному перекладі цього опису в специфікацію на рівні логічних таблиць та інших функціональних компонентів ПЛІС. Елементарні функції в ПЛІС зазвичай реалізуються як окремі проєкти або модулі, що містять інформацію про швидкість передачі даних і внутрішню структуру системи.

Програмування мовою VHDL досить складне, тому постає питання про розробку спеціальних програмних засобів автоматизації, які дозволили б ефективно генерувати високопродуктивний код для програмованих логічних інтегральних схем. У даній статті застосовано алгебро-алгоритмічний підхід для автоматизованого проєктування апаратної реалізації нейронної мережі на ПЛІС. Програми проєктуються у вигляді високорівневих специфікацій у системах алгоритмічних алгебр Глушкова [5], поданих у природно-лінгвістичній формі. Підхід застосовано для розробки нейроконтролера, реалізованого на ПЛІС для системи балансування кульки на платформі.

Підхід, описаний в даній статті, пов'язаний, зокрема, з роботами, присвяченими автоматизованій генерації програм мовою VHDL [6–9]. У [6] запропоновано бібліотеку мовою Java для автоматизованої обробки та генерації коду VHDL. У [7] описано методологію та інструментарій генерації програм для ПЛІС на основі специфікацій Xilinx System Generator. В [8] запропоновано підхід до генерації VHDL програм із використанням кінцевого автомата Мура та графу потоку даних. В роботі [9] представлено генератор коду мовою VHDL на основі предметно-орієнтованої мови для мережевих пристроїв. Основна відмінність даного підходу полягає у використанні природно-лінгвістичного подання схем в системах алгоритмічних алгебр для автоматизованого проєктування програм для ПЛІС.

Подібні до нашого підходу реалізації штучних нейронів розглядаються в роботах [10, 11]. У [10] представлено апаратну реалізацію багатовходового нейрона з нелінійною функцією активації із використанням ПЛІС та VHDL для опису архітектури системи. Відмінністю даного методу є те, що коефіцієнти кусково-лінійної апроксимації функції активації зберігаються в пам'яті лише для додатних або лише для від'ємних значень аргументів, що дозволяє оптимізувати кількість використовуваних обчислювальних ресурсів і підвищити продуктивність нейронної мережі. У [11] запропоновано реалізацію штучної нейронної мережі на ПЛІС з використанням мови Verilog. На відміну від нашої роботи, реалізація функції лінійної активації повільніша та використовує більше ресурсів ПЛІС.

Контролери на основі нейронних мереж для задачі балансування кульки на платформі розглянуті в [12–14]. Зокрема, в [12] запропоновано контролер зі зворотним зв'язком на основі нейронної мережі для компенсації помилок, спричинених використанням апроксимованої динамічної моделі при проєктуванні контролера. Контролер складається з двох паралельно працюючих субрегуляторів: базового лінійного контролера та ПІД-компенсатора на основі нейронної мережі. У [13, 14] для навчання нейронних мереж, що використовуються в контролері, застосовано метод оптимізації роїв частинок.

1. Алгебро-алгоритмічні засоби проєктування програм

Для проєктування програм для ПЛІС у даній роботі використовуються високорівневі схеми, подані в системах алгоритмічних алгебр (САА) Глушкова [5]. Основними об'єктами мови САА є абстракції предикатів (умов) і операторів, що поділяються на базисні та складені. Базисні предикати та оператори вважаються первинними, атомарними, неподільними конструкціями в САА-схемах. Складені предикати та оператори будуються з базисних за допомогою логічних та операторних операцій САА, зокрема:

- диз'юнкції: ‘condition 1’ or ‘condition 2’;
- кон'юнкції: ‘condition 1’ and ‘condition 2’;
- заперечення: not(‘condition’);
- послідовного виконання операторів: «operator 1»; «operator 2»;
- розгалуження: IF ‘condition’ THEN «operator 1» ELSE «operator 2»;
- циклу: WHILE ‘condition’ «operator» END OF LOOP.

Ідентифікатори умов вказуються в одинарних лапках, а оператори — в подвійних. Суперпозиція операцій та базисних елементів САА називається САА-схемою. Основною відмінністю САА від інших процедурних мов програмування є можливість проєктувати програми як в алгебраїчній, так і в природно-лінгвістичній формах, а також виконувати формальні перетворення програм. Конструкції САА для проєктування програм для ПЛІС детально розглянуті в [15].

Автоматизоване конструювання САА-схем та генерацію відповідних Послідовних і паралельних програм цільовими мовами програмування забезпечує Інтегрований інструментарій проєктування та синтезу програм (Integrated toolkit for Design and Synthesis of programs — IDS) [5, 15]. Процес розробки програми в даній системі зображено на рис. 1. Основна ідея інструментарію полягає в розробці схем зверху вниз шляхом вибору операцій САА зі списку та додавання їх до дерева конструювання схеми. Описи конструкцій зберігаються в базі даних ін-

струментарію. На основі розробленого дерева система автоматично генерує програмний код однією з цільових мов програмування (C++, C#, Java, VHDL).

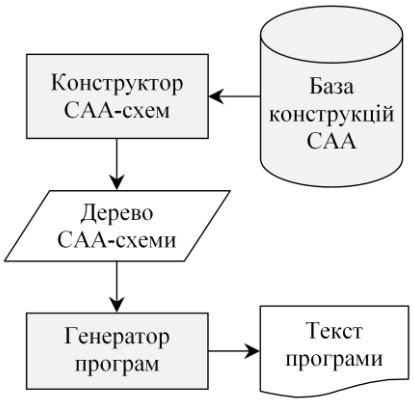

Рис. 1

Наведена нижче САА-схема — приклад проєктування логічної схеми для булевого виразу $y = (\overline{a \wedge b}) \wedge (\overline{a \vee b}) \wedge c$ (рис. 2). Вона складається з опису входів і виходів проєкту (entity) та його функціонування (architecture) і є основою для автоматичної генерації коду мовою VHDL.

```
SCHEME BOOL_EXPR
  ENTITY bool_expr IS
    PORT (
      «Input bit signals (a, b, c)»;
      «Output bit signal (y)»);
  END OF ENTITY;
  ARCHITECTURE arch1 of bool_expr IS
   (y <= (not(a and b)) and (not(a or b)) and c);
  END OF ARCHITECTURE
END OF SCHEME BOOL_EXPR
```

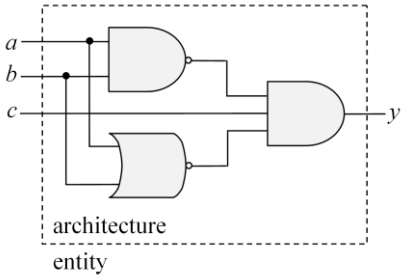

Рис. 2

У даній роботі інструментарій IDS застосовано для проєктування схеми апаратної реалізації нейрона на ПЛІС.

2. Проєктування штучного нейрона для ПЛІС

У даному розділі розглядається модель нейрона та проєктування його функцій активації на ПЛІС. Проєктування нейрона виконано з використанням САА-схем з подальшою генерацією коду мовою VHDL.

Модель нейрона працює таким чином. Вхідні сигнали a_{ki} надходять на блоки, що реалізують функцію синапсів, кожен з яких характеризується своїм

ваговим коефіцієнтом (синаптичною вагою). Зважені вхідні сигнали подаються на лінійний суматор, після чого результат підсумовування потрапляє в блок функції активації $f(.)$ і після відповідної обробки подається на вихід як сигнал q_k. Як правило, функція активації обмежує вихідний сигнал нейрона в діапазоні $[0, 1]$ або $[-1, 1]$. Модель нейрона також містить початкове зміщення b_k, яке додається до вхідного сигналу блоку функції активації. Функціональна схема моделі штучного нейрона безперервного типу наведена на рис. 3.

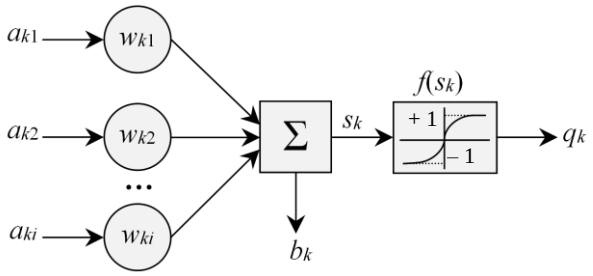

Рис. 3

Математично модель нейрона описується наступними залежностями:

$$q_k = f(S_k) = f\left(\sum_{i=1}^{n} w_{ki} a_{ki} + b_k\right),$$

де q_k — вихідний сигнал k-го нейрона; $f(.)$ — функція активації; a_{ki} — вхідні сигнали нейрона; w_{ki} — синаптична вага; b_k — зміщення k-го нейрона.

Функція активації $f(.)$ реалізує нелінійне перетворення, здійснюване нейроном. Найпоширенішим типом функції активації є сигмоїдальна функція. Такі функції є монотонно зростаючими, неперервними та диференційованими. Диференційованість сигмоїдальних функцій — важлива властивість деяких методів навчання та аналізу. Вони також мають універсальні апроксимаційні властивості [1, 3]. Особливістю нейронів з такою функцією активації є те, що вони значно менше підсилюють сильні сигнали, ніж слабкі, оскільки області сильних сигналів відповідають пологим ділянкам характеристики. Це дозволяє запобігти насиченню від сильних сигналів.

Сигмоїдальні функції описуються виразом

$$f(x, k, b, T, c) = k + \frac{c}{1 + be^{Tk}}, \tag{1}$$

де x, k, b, T, c — параметри; $k, b \in R$; $b > 0$; T, $c \in R \setminus \{0\}$.

Якщо $k = 0$, $c = 1$, $b = 1$ і $T = -1$, то вираз (1) набуде вигляду, що називається «класичним» сигмоїдом:

$$f(x, 0, 1, -1, 1) = 0 + \frac{1}{1 + 1e^{-1x}} = \frac{1}{1 + e^{-x}}. \tag{2}$$

Активною областю визначення функції активації нейрона є область значень вхідних параметрів, де значення функції суттєво змінюються. Для сигмоїдальної функції інтервал $[-4; 4]$ використовується як активна область визначення. У цьому випадку функція приймає значення в інтервалі (0,018; 0,982), що становить 96,4 % від усього діапазону значень.

Пропонуємо наступний метод проєктування функцій активації нейрона на ПЛІС. Вхідними даними методу є функції, описані виразом (1), або ті, які можна записати з їх допомогою, а також точність, з якою має бути реалізована функція активації.

На першому етапі досліджується функція активації щодо симетричності відносно осей. Розглянемо функцію (2)

$$f(-x)=1-f(x)=1-\frac{1}{1+e^{-x}}=1-\frac{1}{1+\frac{1}{e^{x}}}=1-\frac{e^{x}}{e^{x}+1}=\frac{e^{x}+1}{e^{x}+1}-\frac{e^{x}}{e^{x}+1}=\frac{1}{e^{x}+1},$$

Отримаємо

$$1-\frac{1}{1+e^{-x}}=\frac{1}{1+e^{x}} \text{ або } 1-f(x)=f(-x). \tag{3}$$

Функцію $f(x)$ можна розглядати лише для додатних аргументів. Для від'ємних значення можна знайти за формулою (3), що, у свою чергу, прискорить обчислення функції та зменшить використовуваний ресурс ПЛІС.

На другому етапі на кожному з інтервалів визначається кусково-лінійна функція $(-\infty; x_1)$, $(x_1; x_2)$, ..., $(x_n; +\infty)$ окремою формулою

$$f(x)=\begin{cases} k_0x+b_0,\ x<x_1, \\ k_1x+b_1,\ x_1<x<x_2, \\ \quad\cdots \\ k_nx+b_n,\ x_n<x. \end{cases} \tag{4}$$

На третьому етапі $f(x)$ обчислюється для попередньо розрахованого значення x. Коефіцієнти k і b вибираються з пам'яті.

Таким чином, на основі лінійних формул можна знайти апроксимацію функції сигмоїдального типу для будь-якого аргументу із заданою точністю. Розроблений метод побудови нелінійних функцій активації штучного нейрона на ПЛІС відрізняється від наведеного в [10] тим, що коефіцієнти кусково-лінійної апроксимації функції активації зберігаються в пам'яті лише для додатних або лише для від'ємних значень аргументів, що дозволяє оптимізувати кількість використовуваного обчислювального ресурсу та підвищити продуктивність нейронної мережі.

Алгоритм реалізації штучного нейрона з класичною сигмоїдальною функцією складається з таких кроків.

Крок 1. Встановлюються ваги синапсів нейрона. Кожному нейрону надається блок пам'яті, де зберігаються синаптичні ваги.

Для встановлення ваг використовуються такі сигнали:

- synaddr (synapse address) — вибір синапсу, вага якого буде читатися або записуватися за його номером у двійковому коді;
- synsetw (synapse set weight) — значення ваги, яке буде записане в синапс;
- synwren (synapse write enabled) — коли вхідний сигнал дорівнює 1, нейрон записує значення з synsetw у вибраний синапс, а коли сигнал дорівнює 0, нічого не відбувається.

Ці змінні необхідні для навчання нейронів і нейронних мереж.

Крок 2. Значення вхідного вектора подаються на входи штучного нейрона. Встановлюється змінна, яка дорівнює виходу суматора

$$x = \sum_{i=1}^{N} w_i \cdot a_i,$$

де a_i — входи нейрона; w_i — синаптичні ваги нейрона.

Розрахунок виконується за допомогою чисел з фіксованою комою. Кожне число займає 16 біт (9 біт для цілої частини і 7 біт для дробової).

Крок 3. Обчислюється модуль аргументу сигмоїдальної функції. Задається змінна $x' = |x|$.

Крок 4. Сигмоїдальна функція активації розбивається на лінійні частини, і визначаються коефіцієнти k та b лінійних рівнянь. Для цього використовується базова модель кусково-лінійної апроксимації із заданою кількістю лінійних сегментів [16]. Розбиття та похибка показані на рис. 4. Коефіцієнти лінійних рівнянь визначаються наступним чином:

$$k,\ b = \begin{cases} [0,234; 0,500], & \text{якщо } 0 \le x' < 1; \\ [0,129;\ 0,605], & \text{якщо } 1 \le x' < 2,5; \\ [0,234;\ 0,500], & \text{якщо } 2,5 \le x' < 4; \\ [0,009;\ 0,946], & \text{якщо } x' \ge 4. \end{cases}$$

Крок 5. Змінна f визначається за формулою $f = k \cdot x' + b.$

Крок 6. Якщо $x < 0$, значення локальної змінної обчислюються за формулою $f = 1 - f.$

Крок 7. Значення вихідного сигналу нейрона встановлюється рівним значенню змінної f.

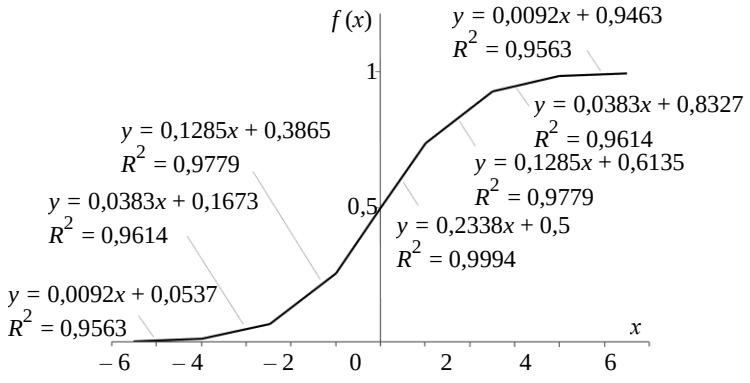

Рис. 4

На рис. 5 зображено класичну сигмоїдальну функцію (*a*) за формулою (2) та функцію, реалізовану на ПЛІС (*b*) на основі розробленого алгоритму. Реалізований блок сигмоїдальної функції на ПЛІС відображає функцію з достатньою точністю для подальшої реалізації штучних нейронних мереж.

Блок штучного нейрона з чотирма входами показаний на рис. 6. Він розроблений автоматизованим способом у формі САА-схеми з використанням інструмен-

тарію IDS, описаного в розд. 1. Схема використана для автоматичної генерації коду VHDL. Приклад опису сутності (entity) створеного проєкту в САА наведено нижче.

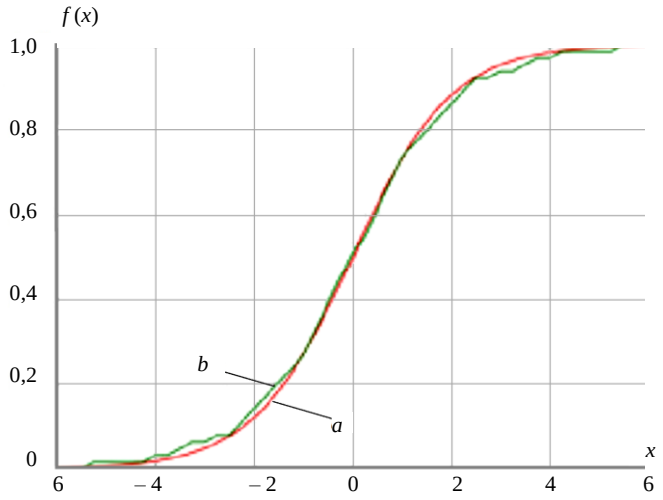

Рис. 5

```
ENTITY neur4sigm IS
  PORT (
    «Input logic vector signals (x1, x2, x3, x4) of range (15) down to (0)»;
    «Input logic vector signal (synaddr) of range (1) down to (0)»;
    «Input logic signal (synwren)»;
    «Input logic vector signal (synsetw) of range (15) down to (0)»;
    «Output logic vector signal (syngetw) of range (15) down to (0)»;
    «Output logic vector signal (y) of range (15) down to (0)» )
END OF ENTITY
```

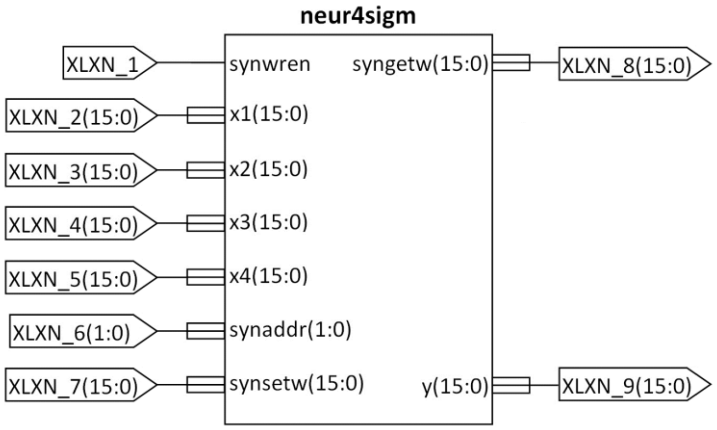

Рис. 6

3. Результати експериментів та практичне застосування запропонованого підходу

Реалізація на ПЛІС штучного нейрона з чотирма входами та сигмоїдальною функцією активації з використанням 16-розрядних чисел із фіксованою комою зайняла 672 LUTs (Look Up Tables — вентильні логічні матриці). Продуктивність (загальна затримка комбінаторної схеми) нейронного блоку 75,6 НС. Абсолютна похибка ± 0,005, а точність реалізації сигмоїдальної функції була наведена раніше в розд. 2 (див. рис. 4).

ISSN 2786-6491

У таблиці наведено порівняння результатів реалізації сигмоїдальної функції активації нейрона з найбільш подібними відомими аналогами [10, 11] на ПЛІС Xilinx Spartan 6 та Xilinx Spartan 3. Як видно, реалізація нейронної мережі на основі розроблених методу й алгоритму є швидшими і потребують менше ресурсів ПЛІС, максимальне відхилення також було зменшене. У даній роботі використовуються нелінійні сигмоїдальні функції на відміну від лінійних, використаних у [10].

Таблиця

Характеристика реалізації	Стаття [10]	Дана робота	Стаття [11]	Дана робота
Серія ПЛІС	Xilinx Spartan 6		Xilinx Spartan 3	
Ресурс ПЛІС (LUTs)	108	60	336	75
Продуктивність, НС	22,12	17,5	120,1	30,2
Максимальне відхилення, %	0,50	0,45	0,556	0,45

Запропонований підхід застосовано для розробки нейромережевого контролера для системи балансування кульки на платформі в режимі реального часу з апаратно-програмною реалізацією на ПЛІС. Платформа регулює положення кульки для приведення її в стан рівноваги шляхом нахилу по горизонтальних осях (x та y) за допомогою двох серводвигунів. Положення кульки фіксується відеокамерою. Спрощене зображення фізичної моделі руху кульки по платформі наведене на рис. 7, де L — відстань від середини до краю платформи (0,35 м); r — відстань від центру кульки до кінця платформи; α — кут повороту платформи; d — довжина подовження серводвигуна (0,05 м); θ — кут його повороту.

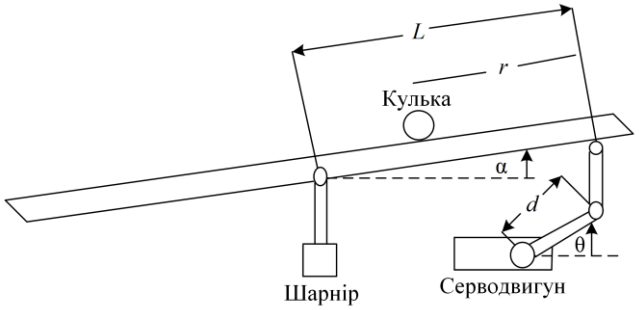

Рис. 7

Для керування рухом кульки за кутом нахилу розроблено нейромережеву систему керування з інверсною моделлю керованого об’єкта та зворотним зв’язком (рис. 8). Була обрана тришарова штучна нейронна мережа з двома нейронами у вхідному шарі, вісьмома нейронами у прихованому шарі та двома нейронами у вихідному шарі, оскільки ця топологія повторює рух кульки з найменшою похибкою. Початкове навчання нейронної мережі проводилось на даних, отриманих у [17] при моделюванні ПІД-регулятора.

Результати моделювання, отримані на платформі з контролером нейронної мережі та традиційними ПІД-регуляторами, представлені на рис. 9 (*а* — положення кульки r від часу; *б* — кута повороту θ серводвигуна від часу, на платформі з нейроконтролером (*n*) та на платформі із звичайними ПІД-регуляторами (*p*) [17]). Як показано на графіках, нейроконтролер адаптується та усуває всі припущення та невизначеності в моделюванні та розрахунках, а також підлаштовується до змін

в умовах його роботи, наприклад, при зміні параметрів кульки та до внесення в систему збурюючих впливів. У результаті кулька встановлюється в задану точку швидше і з меншим відхиленням.

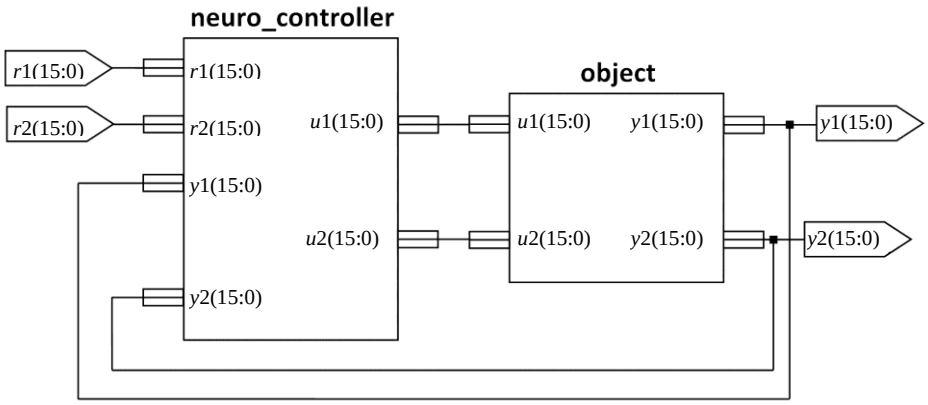

Рис. 8

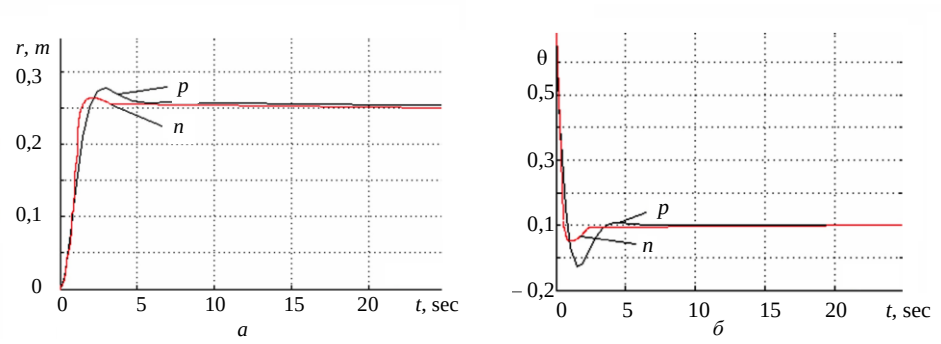

Рис. 9

Висновок

Розроблено засоби автоматизованого проєктування та генерації програм для ПЛІС, що грунтуються на алгебро-алгоритмічних схемах, та застосовано для автоматизованого проєктування штучного нейрона. Схеми використовуються для генерації тексту програми мовою VHDL, який далі виконується на ПЛІС. Розроблено метод конструювання штучного нейрона з сигмоїдальною функцією активації на ПЛІС, який відрізняється від аналогічних підходів тим, що коефіцієнти кусково-лінійної апроксимації функції активації зберігаються в пам’яті лише для додатних або лише для від’ємних значень аргументів. Це дозволило оптимізувати обсяг використовуваних обчислювальних ресурсів і підвищити продуктивність нейронної мережі. Розроблений підхід застосовано для розробки нейромережевого контролеру для системи балансування кульки на платформі, реалізованого на ПЛІС.

ISSN 2786-6491

AUTOMATED DESIGN OF AN ARTIFICIAL NEURON FOR FIELD-PROGRAMMABLE GATE ARRAYS BASED ON AN ALGEBRA-ALGORITHMIC APPROACH

Anatoliy Doroshenko

Institute of Software Systems of National Academy of Sciences of Ukraine, Kyiv,

doroshenkoanatoliy2@gmail.com

Volodymyr Shymkovych

National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute»,

shymkovych.volodymyr@gmail.com

Tural Mamedov

Institute of Software Systems of National Academy of Sciences of Ukraine, Kyiv,

tural.mamedov1@gmail.com

Olena Yatsenko

Institute of Software Systems of National Academy of Sciences of Ukraine, Kyiv,

oayat@ukr.net

Neural network control systems are a high-tech branch of control theory and belong to the class of nonlinear dynamic systems. High speed due to the parallelization of input information combined with the ability to train neural networks makes this technology very attractive for creating control devices in automatic systems. The high-speed operation of networks in real time is provided by implementing them on field-programmable gate arrays. An example of the hardware implementation of neural networks is the design of an artificial neuron and its nonlinear activation functions on an FPGA. The technology of developing applications for FPGAs is based on the presentation of the algorithm in the hardware description language, for example, VHDL, and the automatic translation of this description into a specification at the level of logic tables and other functional components of the FPGA. Programming in the VHDL language is quite complex, so the question arises about the development of special software automation tools that would allow the efficient generation of high-performance code. The paper proposes the facilities of automated design and generation of programs for FPGAs based on the algebra of algorithms. The developed tools are applied for the automated design of an artificial neuron. A method of constructing an artificial neuron with a sigmoidal activation function on an FPGA is developed, which differs from similar approaches in that the coefficients of the piecewise linear approximation of the activation function are stored in memory only for positive or only for negative values of the arguments. This made it possible to optimize the number of utilized computing resources and increase the performance of the neural network. The developed approach is applied to the development of a system with a neural controller for balancing a ball on a platform implemented on an FPGA.

Keywords: automated design, algebra of algorithms, approximate computing, control system, field-programmable gate array, neural network, program generation.

REFERENCES

1. Barron A.R. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions on Information Theory*. 1993. Vol. 39, N 3. P. 930–945. http://doi.org/https://doi.org/10.1109/18.256500.
2. Jhang J.-Y., Tang K.-H., Huang C.-K., Lin C.-J., Young K.-Y. FPGA implementation of a functional neuro-fuzzy network for nonlinear system control. *Electronics.* 2018. Vol. 7, N 145. P. 1–22. http://doi.org/10.3390/electronics7080145.
3. Costarelli D., Spigler R. Approximation results for neural network operators activated by sigmoidal functions. *Neural Networks*. 2013. Vol. 44. P. 101–106. http://doi.org/10.1016/j.neunet.2013.03.015.
4. Godse A.P., Godse D.A. VHDL programming: concepts, modeling styles and programming. Seattle: Amazon Digital Services LLC, 2020. 206 p.
5. Doroshenko A., Yatsenko O. Formal and adaptive methods for automation of parallel programs construction: emerging research and opportunities. Hershey : IGI Global, 2021. 279 p. http://doi.org/10.4018/978-1-5225-9384-3.
6. Pohl C., Paiz C., Porrmann M. vMAGIC — automatic code generation for VHDL. *International Journal of Reconfigurable Computing*. 2009. Vol. 2009, Article ID 205149. P. 1–9. http://doi.org/10.1155/2009/205149.
7. Martín P., Bueno E., Rodríguez Fco. J., Machado O., Vuksanovic B. An FPGA-based approach to the automatic generation of VHDL code for industrial control systems applications: a case study of MSOGIs implementation. *Mathematics and Computers in Simulation*. 2013. Vol. 91. P. 178–192. http://doi.org/10.1016/j.matcom.2012.07.004.
8. de Bulnes D.R.F., Maldonado Y. VHDL code generation as state machine from a data flow graph. *Proc. 2016 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC 2016) (09–11 November 2016, Ixtapa, Mexico).* New York : IEEE, 2016. P. 1–6. http://doi.org/10.1109/ROPEC.2016.7830518.
9. Benáček P., Puš V., Kubátová H., Čejka T. P4-To-VHDL: automatic generation of high-speed input and output network blocks. *Microprocessors and Microsystems*. 2018. Vol. 56. P. 22–33. http://doi.org/10.1016/j.micpro.2017.10.012.
10. Goel K., Arun U., Sinha A. K. An efficient hardwired realization of embedded neural controller on System-On-Programmable-Chip (SOPC). *International Journal of Engineering Research & Technology*. 2014. Vol. 3, N 1. P. 276–284.
11. Singh S., Sanjeevi S., Suma V., Talashi A. FPGA implementation of a trained neural network. *IOSR Journal of Electronics and Communication Engineering*. 2015. Vol. 10, N 3. P. 45–54.
12. Mohammadi A., Ryu J.-C. Neural network-based PID compensation for nonlinear systems: ball-on-plate example. *International Journal of Dynamics and Control*. 2020. Vol. 8. P. 178–188. http://doi.org/10.1007/s40435-018-0480-5.
13. Shaheer M., Hashmi H., Khan S., Atif M., Shabbir Z., Ali A., Kamal K., Zafar T., Awan A. Control of a ball-bot using a PSO trained neural network. *Proc. 2nd International Conference on Control, Automation and Robotics (ICCAR 2016) (28–30 April 2016, Hong Kong, China).* New York : IEEE, 2016. P. 24–28. http://doi.org/10.1109/ICCAR.2016.7486692.
14. Han K., Tian Y., Kong Y., Li J.,. Zhang Y. Tracking control of ball and plate system using an improved PSO on-line training PID neural network. *Proc. 2012 IEEE International Conference on Mechatronics and Automation (5–8 August 2012, Chengdu, China).* New York : IEEE, 2012. P. 2297–2302.
15. Doroshenko A., Shymkovych V., Yatsenko O., Mamedov T. Automated software design for FPGAs on an example of developing a genetic algorithm. *Proc. 17th International Conference «ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer» (ICTERI 2021)* (28 September – 2 October 2021, Kherson, Ukraine). 2021. P. 74–85.
16. Camponogara E., Nazari L.F. Models and algorithms for optimal piecewise-linear function approximation. *Mathematical Problems in Engineering*. 2015. Vol. 2015, Article ID 876862. P. 1–9. http://doi.org/10.1155/2015/876862.
17. Shymkovych V., Samotyy V., Telenyk S., Kravets P., Posvistak T. A real time control system for balancing a ball on a platform with FPGA parallel implementation. *Technical Transactions.* 2018. Vol. 5. P. 109–117. http://doi.org/10.4467/2353737XCT.18.077.8559.

Отримано 28.11.2022