УДК 681.51

П. І. Кравець, к. т. н.;
В. А. Жеребко, асп.;
Х. С. Василевська, студ.;
О. О. Степанчук, студ.

ПІДХІД ДО ПОБУДОВИ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ ТА ЇХ РЕАЛІЗАЦІЯ НА АПАРАТНІЙ ПЛАТФОРМІ COMPACTRIO

Розглянуто питання побудови типової математичної моделі штучної нейронної мережі в середовищі графічного програмування LabVIEW. Розроблено приклад нейромережевої структури для подальшої технічної реалізації на апаратній платформі промислового контролеру CompactRIO. Нейромережева структура та алгоритм навчання інтегруються у FPGA-контролер.

Під час проектування та побудови сучасних інтелектуальних систем автоматичного управління постає важливе питання — практична реалізація окремих синтезованих моделей системи на апаратному рівні. В рамках цієї роботи розглядається підхід до апаратної реалізації нейромережевих структур (НМС) моделей системи, побудованих на штучних нейронних мережах (ШНМ).

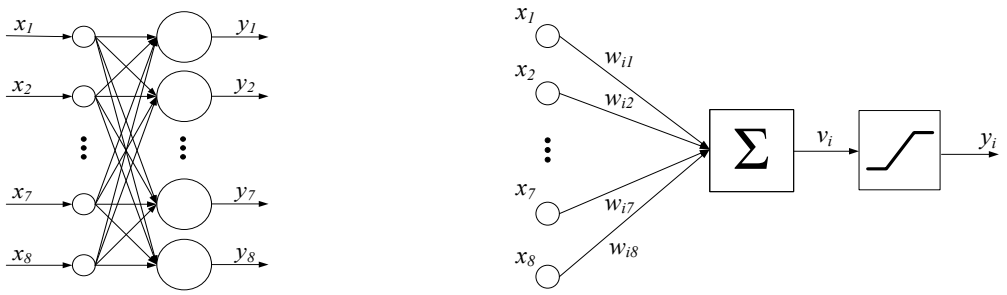
Реалізувати ШНМ можна на нейрообчислювачах (апаратне моделювання структури ШНМ) або на нейроемуляторах (програмне моделювання НМС) [1]. Програмне моделювання ШНМ можна легко здійснити, наприклад, на сучасному персональному комп'ютері (ПК) або на PC-based контролері. Такий спосіб реалізації ШНМ часто називають емуляцією ШНМ на ПК. Але окрім завдання користувача, що виконує, наприклад, нейромережевий алгоритм управління, операційна система на ПК виконує безліч службових та сервісних програм, які призводять до переривань в роботі програми користувача. Тому апаратна реалізація алгоритмів та структур користувача із паралельними обчислювальними процесами доцільніша ніж програмна.

Як відомо [2], ШНМ — це алгоритмічні процесори, які використовуються для паралельних обчислень. «Чистий» паралелізм реалізувати на комп'ютерних обчислювачах неможливо, тобто сама архітектура процесорного ядра ПК апріорі обмежує можливість повного розпаралелювання обчислювальних процесів. Тому, у якості паралельного обчислювального ядра для побудови систем управління на ШНМ доцільно використовувати програмовані логічні інтегральні схеми (ПЛІС, англ. — FPGA). Структура ПЛІС надає можливість гнучкої конфігурації взаємозв'язків між внутрішніми логічними блоками і, тим самим забезпечує реалізацію паралельного виконання декількох процесів одночасно.

У якості передової апаратної платформи для реалізації НМС, вибрана компактна система CompactRIO компанії National Instruments (NI). Це високопродуктивна платформа, заснована на технології RIO (reconfigurable I/O technology) і графічного програмування LabVIEW. CompactRIO (cRIO) складається із FPGA-контролера (паралельний апаратний цілочисельний обчислювач) та Real-Time контролера (PC-based контролер реального часу).

Підхід до побудови НМС був відпрацьований на моделі cRIO-9004 (64 Мб оперативної пам'яті, 512 Мб енергонезалежної пам'яті, для підключення до ПК та програмування використовувався Ethernet Port 10/100 BaseTX) і 9104 (шасі з 8-ма слотами та ядром ПЛІС). Необхідно підкреслити, що платформа cRIO програмується виключно мовою G в середовищі LabVIEW.

В цій роботі використана модель одношарової ШНМ прямого розповсюдження (Feed-Forward), яка має 8 входів та 8 виходів. Схематично модель такої ШНМ зображена на рис. 1а. При підрахунку кількості шарів не враховують нейрони на вході, бо ніяких обчислень вони не виконують [2]. На рис. 1б зображено модель одного нейрона.



а б

Рис. 1. Нейромережева структура:

а — модель одношарової нейронної мережі прямого розповсюдження
із 8 входами та 8 виходами; б — модель одного нейрона із функцією активації

Нейрон працює під управлінням вектора вхідного сигналу $X(n)$, що отримується від попереднього шару нейронів (в нашому випадку — вхідного), а n — номер кроку ітеративного процесу (епохи) корегування вагових коефіцієнтів. Вектор вихідного сигналу $Y(n)$ визначається таким чином:

$$y_i(n) = f(v_i), \tag{1}$$

де $f(v_i)$ — функція активації нейрону, що описується

$$f(v_i) = \begin{cases} 0, & v_i > 0,5; \\ v_i + 0,5, & -0,5 < v_i < 0,5; \\ 1, & v_i < -0,5. \end{cases} \tag{2}$$

А v_i описується таким виразом:

$$v_i = \sum_{1}^{8} x_j(n)\, w_{ij}, \tag{3}$$

де w_{ij} — поточне значення вагового коефіцієнта (i — номер нейрона, j — номер входу).

Навчання ШНМ проходить за «дельта-правилом». Вихідний сигнал $y_i(n)$ порівнюється із бажаним значенням виходу нейрона $d_i(n)$. В результаті розраховується значення помилки

$$e_i(n) = d_i(n) - y_i(n). \tag{4}$$

Сигнал помилки $e_i(n)$ ініціює механізм управління (навчання), метою якого є застосування послідовності корегувань до значень вагових коефіцієнтів w_{ij}. Таким чином, отримуємо покрокове наближення вихідного сигналу $y_i(n)$ до бажаного $d_i(n)$. Корегування продовжуються до тих пір, поки система не досягає стійкого стану. Після цього процес навчання завершується.

У відповідності до «дельта-правила», зміна вагового коефіцієнта $\Delta w_{ij}(n)$ задається виразом

$$\Delta w_{ij}(n) = \eta e_i(n)\, x_i(n), \tag{5}$$

де η — деяка додатна константа, яка визначає швидкість навчання (Learning Rate Parameter), і використовується при переході від одного кроку навчання до іншого. Для забезпечення стійкості та збіжності ітеративного процесу навчання необхідно ретельно підбирати цей параметр [2].

Фактично «дельта-правило» можна визначити таким чином: корегування, що застосовується до вагового коефіцієнта w_{ij}, пропорційне добутку сигналу помилки на вхідний сигнал, що його викликав.

Середовище графічного програмування LabVIEW не має допоміжних інструментів для проектування НМС. Але, як було зазначено вище, будь-яка ШНМ складається із сполучених і взаємодіючих між собою штучних нейронів (див. рис. 1), реалізація яких на мові G в пакеті LabVIEW не викликає труднощів [3].

Для роботи з контролером CompactRIO необхідно встановити до базового пакету LabVIEW додаткові програмні модулі LabVIEW FPGA та LabVIEW Real-Time. При проектуванні НМС та кодуванні їх мовою G на платформі CompactRIO FPGA потрібно враховувати такі обмеження середовища LabVIEW FPGA:

— відсутність підтримки багатовимірних масивів;
— кількість елементів одновимірних масивів фіксована;
— відсутність типу даних із плаваючою комою (Floating-Point);
— не реалізовано генератор псевдовипадкових чисел.

У LabVIEW FPGA версії 8.5 була введена підтримка формату даних із фіксованою комою (Fixed-Point). Операції суми та множення виконуються за допомогою стандартних математичних операцій LabVIEW FPGA. А от операція ділення взагалі відсутня в LabVIEW FPGA. Тому для коректного оперування даними Fixed-Point доцільно встановити допоміжний модуль LabVIEW FPGA Fixed-Point Math. Також він дозволяє жорстко проставляти розмірність цілої та дробової частин кожного константного числа та/або змінної в коді програми.

Базовим прикладом для розробки та проектування НМС було обрано бібліотеку з відкритим кодом neural_net_learner.llb, що розрахована на комп'ютерну емуляцію ШНМ. Допоміжні модулі та приклади отримано з офіційного сайту NI [4].

Головним результатом роботи є розробка мовою G в середовищі LabVIEW FPGA програмно-апаратної нейромережевої структури (ПАНМС) для платформи CompactRIO, яка складається із таких елементів:

— підпрограма початкової ініціалізації вагових коефіцієнтів (випадковими числами);
— підпрограма підрахунку суми вхідних сигналів;
— підпрограма, що реалізує функцію активації нейрону;
— підпрограма, що реалізує алгоритм навчання «дельта-правило».

Реалізація програми навчання ШНМ показана на рис. 2 і складається із послідовності спрацьовування ШНМ із подальшим коригуванням вагових коефіцієнтів мережі у відповідності з «дельта-правилом», згідно формули (5). Спрацьовування ШНМ виконується послідовним розрахунком виходу кожного нейрону.

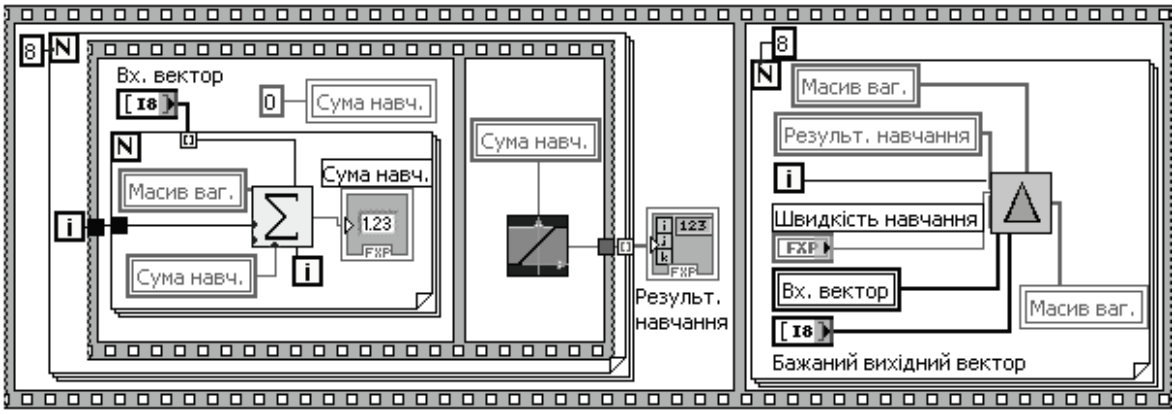


Рис. 2. Фрагмент програми навчання ШНМ в LabVIEW FPGA

Програма роботи ШНМ показана на рис. 3.

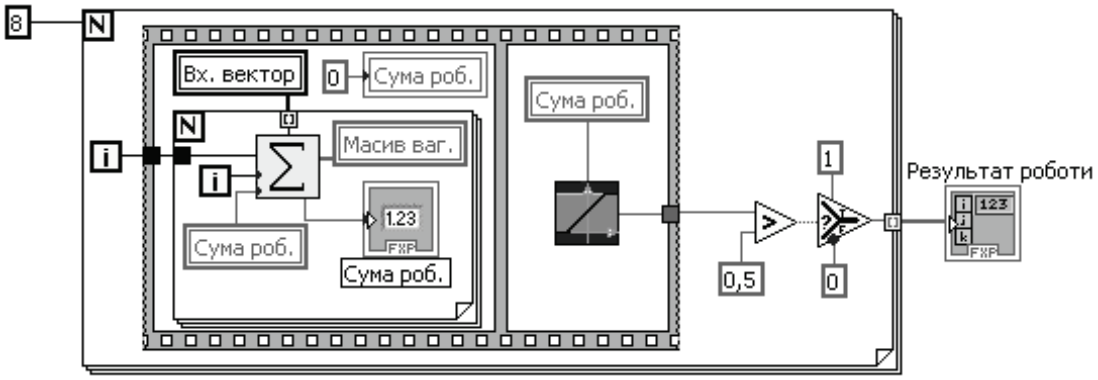


Рис. 3. Фрагмент програми роботи ШНМ в LabVIEW FPGA

Спочатку розраховується сума добутків вхідного сигналу на вагові коефіцієнти у відповідності до формули (3). Далі застосовується активаційна функція, що описується формулою (2).

Враховуючи те, що активаційна функція забезпечує не лінійність, а процес навчання — наближення до максимальної відповідності бажаному відклику ШНМ, на виході мережі матимемо лише наближені значення, а не бажані. Тому в програмі використовується додатковий логічний блок порівняння з порогом, підібраним для розв'язання прикладу задачі визначення належності цілого числа до певного діапазону (див. далі).

Інтерфейс програми ПАНМС показано на рис. 4.

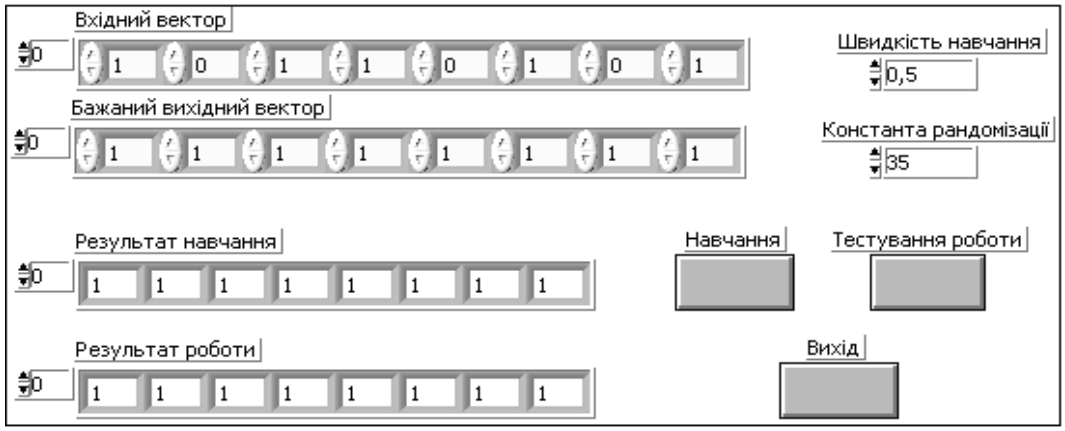


Рис. 4. Інтерфейс програми ПАНМС в LabVIEW FPGA

На початку роботи задаються значення швидкості навчання та константа рандомізації. Константа рандомізації задає максимальне відхилення при ініціалізації вагових коефіцієнтів випадковими числами.

Навчання ШНМ полягає у тому, що спочатку задаються пара вхідного та бажаного вихідного вектору. Після натискання кнопки «Навчання» у рядку «Результат навчання» виводиться поточна реакція ШНМ на вхідний вектор. Вона буде змінюватися протягом навчання ШНМ в залежності від зміни вагових коефіцієнтів.

Тестування роботи ШНМ проводиться шляхом натискання кнопки «Тестування роботи», при цьому корегування вагових коефіцієнтів не здійснюється, а результат роботи мережі буде відображений у рядку «Результат роботи».

Для реалізації двовимірного масиву (8×8) вагових коефіцієнтів, в роботі було використано одновимірний масив фіксованої довжини із 64-х елементів. Формат змінних типу Int вибирають I8, а формат Fixed-Point (FXP) — U8.8. Якщо розмір кластеру або масиву даних не перевищує 32 біти, то на реалізацію кожного біту даних буде витрачено одна логічна одиниця (flip-flop) ядра FPGA. Наприклад, масив із 4-х елементів типу U8 займатиме 32 біти. У разі перевищення цього розміру на кожен біт даних буде використано вже не одна, а пара логічних одиниць (slice).

Демонстрація та перевірка роботи ШНМ програми ПАНМС була опрацьована на прикладі простої задачі, за умови якої необхідно визначити чи є довільне ціле число з проміжку 0..255, більшим за число 120. Програма ПАНМС, в такому випадку, працює таким чином. На вході ШНМ формуються цілі числа у двійковому форматі, а на виході — всі нулі (у випадку, коли задане число менше 120), або всі одиниці (якщо задане число більше 120).

Далі проводиться ітеративне покрокове навчання ШНМ. Кількість ітерацій не задана, всі маніпуляції проходять в ручному режимі. Після закінчення процедури навчання ШНМ, на входи ШНМ подаються деякі тестові бінарні послідовності (еквіваленти цілих чисел із проміжку 0..255) та отримують результати, наведені в таблиці.

Вхідний вектор ШНМ	Вихідний вектор ШНМ
$0000\ 0001_2 = 1_{10}$	0000 0000
$1000\ 0000_2 = 128_{10}$	1111 1111
$1000\ 1001_2 = 137_{10}$	1111 1111
$0101\ 1000_2 = 88_{10}$	0000 0000

Код програми ПАНМС завантажується лише на цільову платформу cRIO FPGA. Інтерфейси всіх підпрограм, за виключенням головного вікна, вимкнено для економії витрат ресурсів ПЛІС.

Процес компіляції коду в LabVIEW FPGA може тривати від декількох хвилин до годин. В процесі компіляції код на мові G конвертується в код на мові VHDL, а потім він компілюється в бітфайл конфігурації ПЛІС за допомогою сервера компіляції Xilinx ISE.

ПЛІС cRIO (ядро Virtex-II 3000) має біля 28 тисяч тригерів та більше ніж 3 млн логічних одиниць (вентилів). Код розробленої програми ПАНМС був скомпільований на ПК (процесор з подвійним ядром, частота 3ГГц, 2 Гб оперативної пам'яті) за проміжок часу близько 40 хвилин. В результаті об'єм витрачених пар логічних одиниць склав приблизно 80 %, що говорить про неоптимальне використання ресурсів ПЛІС. Програма оптимізації, а також суттєве зменшення розміру коду програми (біт-файлу) виходять за межі цієї роботи.

Платформа CompactRIO відкриває широкі можливості для подальшого розвитку програмно-апаратного проектування та оперування НМС. Реалізація різноманітних динамічних алгоритмів навчання/перенавчання, налагодження, корегування структури ШНМ, тощо в реальному часі покладатиметься на Real-Time-контролер, а структури ШНМ мають завантажуватись для виконання лише у FPGA-контролер. В такому випадку використання ресурсів ПЛІС буде мінімальним.

СПИСОК ЛІТЕРАТУРИ

1. Власов А. И. Аппаратная реализация нейровычислительных управляющих систем / А. И. Власов // Приборы и системы управления. — 1999. — № 2. — С. 61—65.

2. Хайкин С. Нейронные сети: полный курс. : пер. с англ.. — 2-е изд. / С. Хайкин. — М. : Издательский дом «Вильямс», 2006. — 1104 с.

3. Pogula Sridhar Sriram. Developing Neural Network applications using LabVIEW / Sriram Pogula Sridhar. Adviser: Robert W. McLaren // Electrical Engineering, MS SS. — 2005. — 115 p.

4. The National Instruments CompactRIO PAC [Електронний ресурс] // National Instruments, 2008. — Режим доступу: http://www.ni.com/compactrio.

Рекомендована кафедрою комп'ютерних систем управління

Надійшла до редакції 21.10.08
Рекомендована до друку 20.11.08

Кравець Петро Іванович — доцент, ***Жеребко Валерій Анатолійович*** — аспірант,
Кафедра автоматики та управління в технічних системах ;
Василевська Христина Сергіївна — студентка, ***Степанчук Олена Олександрівна*** — студентка.
Національний технічний університет України «КПІ», м. Київ