УДК 681.5.13

КРАВЕЦЬ П.І.,
ШИМКОВИЧ В.М.,
ОМЕЛЬЧЕНКО П.

НЕЙРОМЕРЕЖЕВІ КОМПОНЕНТИ СИСТЕМ КЕРУВАННЯ ДИНАМІЧНИМИ ОБЄКТАМИ З ЇХ АПАРАТНО-ПРОГРАМНОЮ РЕАЛІЗАЦІЄЮ НА FPGA

В даній роботі розроблені та досліджені нейромережеві компоненти систем керування динамічними об'єктами з їх апаратно програмною реалізацією на FPGA, а саме розроблено метод синтезу таких типових моделей як пряма та інверсна модель об'єкта управління та алгоритм апаратно-програмної реалізації фільтра Калмана.

In this paper neural networks are designed and research components of control systems dynamic objects with their hardware program implementation on FPGA, namely those developed typical model as direct and inverse model of the object management and algorithm for hardware-software implementation of Kalman filter.

Нейромережеві системи управління являють собою новий високотехнологічний напрямок в теорії управління та відносяться до класу нелінійних динамічних систем [1,2]. Висока швидкодія за рахунок розпаралелювання вхідної інформації в поєднанні зі здатністю до навчання нейронних мереж робить цю технологію вельми привабливою для створення пристроїв управління в автоматичних системах [2]. Нейронні мережі можуть бути використані для побудови регулюючих та коректуючих пристроїв, еталонної, адаптивної, номінальної та інверсно-динамічної моделей об'єкта, на основі яких виконується спостереження та оцінка параметрів об'єкта керування (ОК), спостереження та оцінка величини діючих в системі збурень, пошук або обчислення оптимальної програми зміни керуючого впливу, ідентифікація ОК, прогнозування стану ОК та інше [2,3]. Здатність до навчання на будь-який заданий принцип функціонування дозволяє створити системи автоматичного керування, оптимальні по швидкодії, по енергоспоживанню і т. д., при цьому, природно, можлива реалізація декількох принципів функціонування та перехід з одного на інший. Навчені нейронні мережі не потребують для обчислень значних часових затрат, а тому системи з нейронними мережами мають значно кращу динаміку. Вони є універсальним засобом для моделювання складних нелінійних ОК та знаходження рішень в некоректних задачах [2-5].

Засоби реалізації нейромережевих систем управління повинні орієнтуватися на широке застосування в промислових умовах, бути універсальними і гнучкими, навчатися і адаптуватися в реальному часі, бути простими і дешевими, тому найбільш перспективними засобами можна вважати FPGA [6,7]. З появою FPGA проектування цифрових мікросхем перестало бути долею виключно великих підприємств з обсягами випуску в десятки і сотні тисяч кристалів. Проектування і випуск невеликої партії унікальних цифрових пристроїв став можливий в умовах проектно-конструкторських підрозділів промислових підприємств, в дослідницьких і навчальних лабораторіях і навіть в умовах домашніх радіоаматорських місць. Промислово випускаються «заготовки» програмованих мікросхем з електричним програмуванням і автоматизованим процесом перекладу схеми користувача в послідовність імпульсів програмування роблять проектування нових цифрових пристроїв порівнянним з розробкою програмного забезпечення [8-10].

Розглянуті далі моделі систем керування включають елементарні схеми, які можуть бути базовими для структурного синтезу функціонально більш складних систем керування. Властивості ШНМ з динамічними алгоритмами навчання дозволяють моделювати складні нелінійні динамічні об'єкти управління – у вигляді прямих та інверсних моделей по вимірах «вхід-вихід» цього об'єкта. Обидві моделі використаються для обчислення векторів стану об'єкта і формування функції управління ним. Також важливим завданням в теорії управління є побудова ефективних алгоритмів оцінювання стану динамічної системи [11]. Один з підходів до оцінювання вектора стану в умовах невизначеності – імовірнісний, згідно з яким обурення і перешкоди є випадковими величинами з відомими функціями розподілу. Завдання фільтрації, оцінки стану динамічної системи за вимірюваннями, при випадкових збуреннях допус-

кає практично вичерпне рішення за допомогою фільтра Калмана [12].

Метою роботи є розробка методів та алгоритмів апаратно-програмної реалізації нейромережевих компонентів систем керування динамічними об'єктами на FPGA.

Пряму модель об'єкта управління по вимірах вхідних $u(k)$ та вихідних $y(k)$ даних можна отримати по схемі зображеній на рис. 1. Відома в теорії ідентифікації як схема з налагоджуваною моделлю [13], реалізується в даному випадку рекурентною штучною мережею, що навчається на основі помилки $\hat{e}(t)$:

$$\hat{y}(k+1) = \mathbf{F}\begin{pmatrix} u_k, z^{-1}u_k, ..., z^{-m}u_k; \\ \hat{y}_k, z^{-1}\hat{y}_k, z^{-n}\hat{y}_k; \\ \mathbf{w}_i^{(l)} \end{pmatrix} \quad (1)$$

В якості вектора стану мережі обрано вектор $col(u, z^{-1}y, ..., z^{-n}y) = col(\hat{x}_n(k), \hat{x}_{n-1}(k), ..., \hat{x}_1(k))$, де z^{-1} – оператор затримки. Результатом ідентифікації динамічної моделі поведінки реального ОК в сенсі близькості функцій виходів $\hat{y}(t)$ та $y(t)$ з точністю до помилки навчання $\hat{e}(t) = y(t) - \hat{y}(t)$, або мінімуму деякого функціонала $J(e(t))$, можуть бути налаштовані значення вагових коефіцієнтів $\mathbf{w}_i^{(l)}$ у шарах $l = \overline{1, K}$ та оцінки вектора стану об'єкта, який в загальному випадку може бути представлений параметрично недовизначеним нелінійним диференціальним рівнянням:

$$y(k+1) = f\begin{bmatrix} y(k), y(k-1), ..., y(k-n+1); \\ u(k), ..., u(k-m+1) \end{bmatrix}.$$

Структурна схема динамічної нейромережі з одним входом та одним виходом, що відповідає схемі прямої моделі навчання з використанням вимірів «вхід-вихід об'єкта» керування зображена на рис. 1. Схема прямої моделі навчання з використанням вимірів «вхід-вихід об'єкта» керування зображена на рис. 2.

Засновані на ШНМ дискретні ідентифікаційні моделі називаються нейроемуляторами або предикторами. У загальному вигляді вони описуються нелінійним рівнянням (1).

Для одержання прямої моделі в заданому класі функцій $\mathbf{u}(t) \in \mathbf{U}$ не потрібно повної апріорної інформації про структуру зв'язків й їхніх операторів для ОК, окрім інформації про стійкість та обмеженість всіх траєкторій $\mathbf{y}(t)$ для $t \geq 0$. Варто зауважити, що навчена мережа побічно «враховує» вплив на реальний об'єкт управління зовнішніх збурень.

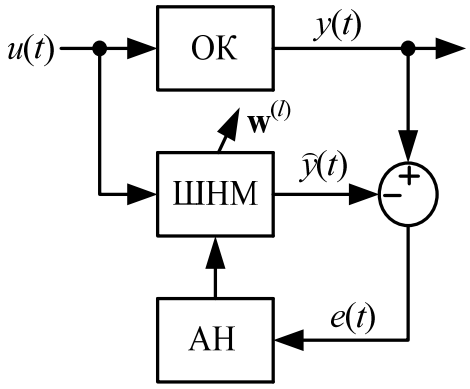

Рис. 1. Структурна схема адаптивної ідентифікації

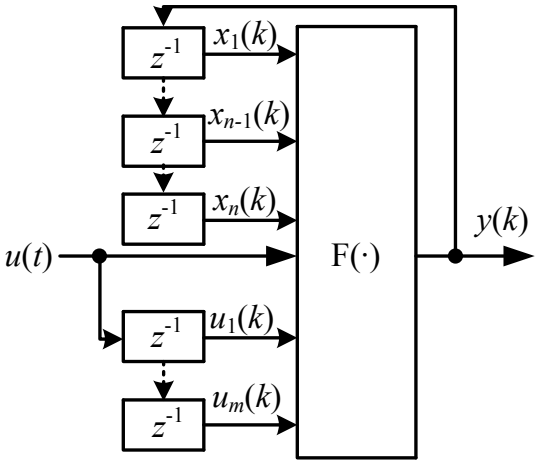

Рис. 2. Структурна схема прямої моделі ОК

Для побудови інверсної моделі об'єкта управління можуть бути використані два підходи навчання ШНМ: узагальнене та спеціалізоване інверсне навчання

В схемі узагальненого інверсного навчання, рис. 3, в якості вхідних даних використається тестовий сигнал $\mathbf{u}^*(t)$. Це може бути, наприклад, значення відомої функції оптимального керування об'єктом. Вихід реального ОК $\mathbf{y}(t)$ при підключенні виходу навченої мережі до входу об'єкта відтворює значення функції $\mathbf{r}(t)$, тобто якщо A – оператор ОК, то при $\mathbf{e}_u(t)=0$ нейромережа відображає інверсний оператор A^{-1}, що й пояснює термін «інверсне навчання».

Тестовий сигнал $\mathbf{u}^*(t)$ повинен задовольняти вимозі повного перекриття можливого діапазону зміни керуючого впливу в реальній системі керування.

Другий підхід до інверсного моделювання об'єкта управління реалізується схемою спеціа-

лізованого інверсного навчання, зображеною на рис 4. Ця ж схема дозволяє відтворювати задану функцію **r**(*t*).

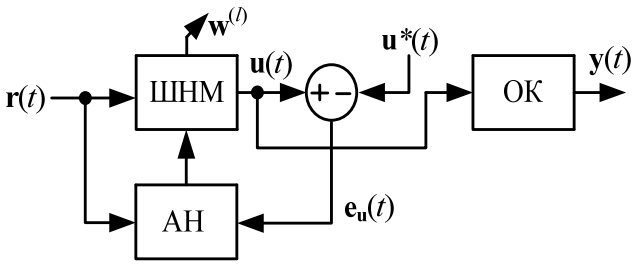

Рис. 3. Структурна схема загального інверсного навчання ШНМ

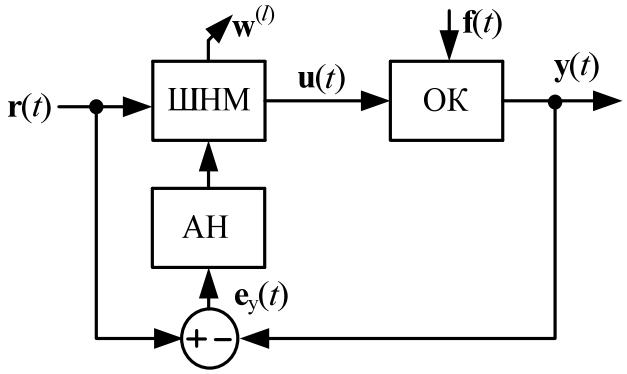

Рис. 4. Структурна схема спеціалізованого інверсного навчання ШНМ

У даній схемі сигнал **r**(*t*) виконує роль тестового в завданні інверсного моделювання об'єкта керування за допомогою нейромережі і його форма відповідає класу відтворених у завданні управління функцій.

На відміну від узагальненої схеми в спеціалізованій використовується помилка між заданою функцією **r**(*t*) і виходом **y**(*t*), нейронна мережа налагоджується за динамічним алгоритмом навчання. Якщо нейромережа навчена і помилка $\mathbf{e}_y(t)=0$, то функція **r**(*t*) у точності відтворюється на виході об'єкта. В [17] стверджується, що схема спеціалізованого інверсного навчання дозволяє, відтворити точну інверсну модель об'єкта управління при використанні реального виходу **y**(*t*).

Інверсні моделі самі по собі можуть бути застосовні для управління динамічними об'єктами, у тому числі і для адаптивного управління [2,17-19], але прямі й інверсні моделі можуть служити типовими «будівельними блоками» для синтезу багатомірних систем управління об'єктами з більш складною структурою.

Апаратно-програмна реалізація моделей та елементів систем управління на основі ШНМ на прикладі реалізації узагальненої нейромережевої моделі об'єкта управління рис.2, виконується за наступним алгоритмом:

Крок 1. Проведення експерименту.

Основним завданням проведення експерименту є побудова інформативної множини даних *Z*, придатної для побудови працездатної моделі:

$$Z^N = \{[u(t),\, y(t)], t = \overline{1, N}\}, \qquad (2)$$

де: $u(t)$ – тестовий сигнал;

$y(t)$ – реакція об'єкту на тестовий сигнал;

t – час;

N – розмірність множини *Z*.

Побудова інформативної множини даних *Z* може бути виконана шляхом імітаційного моделювання (у випадку якщо модель об'єкту відома) або шляхом проведення серії експериментів з реальним об'єктом (у випадку, якщо модель об'єкту не відома.).

Для нелінійних об'єктів надзвичайно важливо, що б в множині експериментальних даних Z^N були представлені всі можливі комбінації амплітуд і частот з робочого діапазону системи. Один з можливих варіантів тестового сигналу, що задовольняє вказаним вимогам, описується виразом:

$$u(t) = u(t-N) + e(\mathrm{int}\left[\frac{t-1}{N}\right]+1),\ t = 1,2,\ldots \qquad (3)$$

де: $e(t)$ – білий шум з дисперсією σ_e^2.

Іншим варіантом тестового сигналу є синусоїда з наростаючою частотою ω_c і амплітудою A_c, що змінюється:

$$u(t) = u_0 + A_c \sin(\omega_c t T),$$
$$\omega_c = \omega_н + \frac{(\omega_к - \omega_н)}{N}, \qquad (4)$$
$$A_c = A_н + \frac{(A_к - A_н)}{N},$$

де $\omega_н$, $\omega_к$ – відповідно початкове і кінцеве значення частоти синусоїдального сигналу;

$A_н$, $A_{\hat{e}}$ – відповідно початкове і кінцеве значення амплітуди синусоїдального сигналу.

Для ефективного використання отриманої тестової множини експериментальних даних необхідно виконати його попередню обробку (фільтрацію, видалення надмірних даних і викидів, масштабування), метою якого є отримання найбільш значущої інформації і приведення її до певного вигляду, необхідного для навчання нейромережевої моделі.

Крок 2. Вибір області можливих модельних структур.

Під модельною структурою розумітимемо структуру нейронної мережі, яка може бути умовно розділена на «внутрішню структуру» і «зовнішню структуру».

Внутрішня структура нейронної мережі визначається: топологією мережі, кількістю прихованих шарів, числом нейронів і видом активаційних функцій в кожному шарі.

Зовнішня структура нейронної мережі визначається вектором входу $\varphi(t)$ (регресором). Вектор входу $\varphi(t)$ нейронної мережі можна представити як:

$$\phi(t) = [\phi_1 \dots \phi_k]^T = \\ = [\phi_1(t-1) \dots \phi_1(t-d_1) \dots \phi_k(t-1) \dots \phi_k(t-d_k)]^T, \quad (5)$$

де: φ_k – k-а компонента регресора;

d_k – «глибина» регресора.

Під вибором регресора мається на увазі визначення компонент регресора φ_k і глибини регресії d_k, тобто кількості d значень k-ої компоненти регресора в попередні значення часу. У якості компонент регресора зазвичай використовуються ті параметри системи (процесу), які можуть бути безпосередньо виміряні (або оцінені) в режимі функціонування. Наприклад, для одновимірних об'єктів в якості компонентів регресора використовується значення входу $u(t)$ і виходу $y(t)$ об'єкту. Вибір глибини регресії визначається динамікою об'єкта .

Таким чином, завдання вибору структури нейронної мережі зводиться до визначення «глибини» регресора $\varphi(t)$ («зовнішня» структура) і кількості нейронів прихованого шару n («внутрішня» структура) багатошарової нейронні мережі, що містить один прихований шар нейронів з сигмоїдальними функціями активації і вихідний шар з лінійними функціями активації.

Визначення області «можливих» модельних структур $V(n, \varphi_1, \dots, \varphi_k)$ в області модельних структур $M(n, \varphi_1, \dots, \varphi_k)$, виконується на основі аналізу апріорної інформації про об'єкт і його динаміку.

Крок 3. Оцінка моделей з області «можливих» модельних структур.

У роботі [15] розглядається комплексний формалізований підхід до реалізації багатоетапної процедури побудови нейромережевих моделей складних динамічних об'єктів, розроблено програмний пакет «MIMO-Plant» в середовищі MatLab. Для кожної моделі з області $V(n, \varphi_1, \dots, \varphi_k)$ обчислюється значення критерію адекватності, у якості котрого можна використовувати значення середньоквадратичної помилки. Представлено детальний опис програмного пакету та приклади дослідження.

Крок 4. Реалізація нейромережевої моделі на ПЛІС.

При апаратно-програмній реалізації нейромережевих моделей на FPGA використовується метод апаратно-програмної реалізації ШНМ описаний в [14].

Крок 5. Оцінка «вартості» реалізації моделі на FPGA.

Для вибраної моделі обчислюється значення критерію «вартості» апаратної реалізації нейромережевої моделі на ПЛІС. Під «вартістю» реалізації C в даному випадку розуміється частина ресурсів (Slices), кількість D–тригерів (Flip Flops), об'єм вбудованої блокової пам'яті (BRAM), кількість чотиревходових таблиць перетворення (Look-up Table)), необхідних для реалізації отриманої моделі на вибраному типі кристала ПЛІС

$$C = \max\left(\frac{\sum_{i=1}^{I} S_i}{S_X}, \frac{\sum_{i=1}^{I} F_i}{F_X}, \frac{\sum_{i=1}^{I} R_i}{R_X}, \frac{\sum_{i=1}^{I} L_i}{L_X} \right) \quad (6)$$

де: S_i, F_i, R_i, L_i – кількість Slices, Flip Flops, BRAM, Look-up Table, відповідно, що необхідна для реалізації нейромережевої моделі;

S_X, F_X, R_X, L_X – кількість Slices, Flip Flops, BRAM, Look-up Table, відповідно, яка міститься у вибраному кристалі ПЛІС;

I – сумарна кількість елементів (ваги, зсуви, функції активації, затримки) тих, що реалізовують нейронну мережу.

Умовою реалізуємості нейромережевої моделі на вибраному кристалі ПЛІС є виконання нерівності

$$C < 0.7 . \quad (7)$$

Крок 6. Прийняття рішення про можливість практичної реалізації.

Якщо отримана модель не задовольняє якомусь критерію, то необхідно виконати попередні кроки алгоритму побудови моделі, аж до проведення нової серії експериментів.

Відповідно до даного алгоритму, якщо умова (7) виконується, то дану модель можна реалізувати на вибраному кристалі ПЛІС, якщо ж умо-

ва (7) не виконується, то модель на такому кристалі ПЛІС реалізувати неможливо. В цьому випадку забезпечити виконання умови (7) можна наступним чином: 1) виконати процедуру оптимізації структури моделі; 2) вибрати кристал ПЛІС більшої ємності і/або іншого сімейства; 3) збільшити число кристалів ПЛІС.

Крок 7. Оптимізація структури моделі

Оптимізація структури нейромережевої моделі проводиться шляхом видалення ряду малозначних вагових коефіцієнтів. Для реалізації цього можуть бути використані такі алгоритми:

- алгоритм послідовного зменшення структури;
- алгоритм послідовного збільшення структури;
- генетичний алгоритм;
- Optimal Brain Damage (OBD);
- Optimal Brain Surgeon (OBS) [5].

Останній алгоритм є найбільш відомим і широко поширеним. Після виконання мінімізації структури необхідно заново оцінити адекватність моделі і виконати кроки 5, 6.

Розглянемо приклад реалізації прямої та інверсної моделі такого динамічного об'єкту як індукційна піч. Динаміка індукційної печі детально описана в [20].

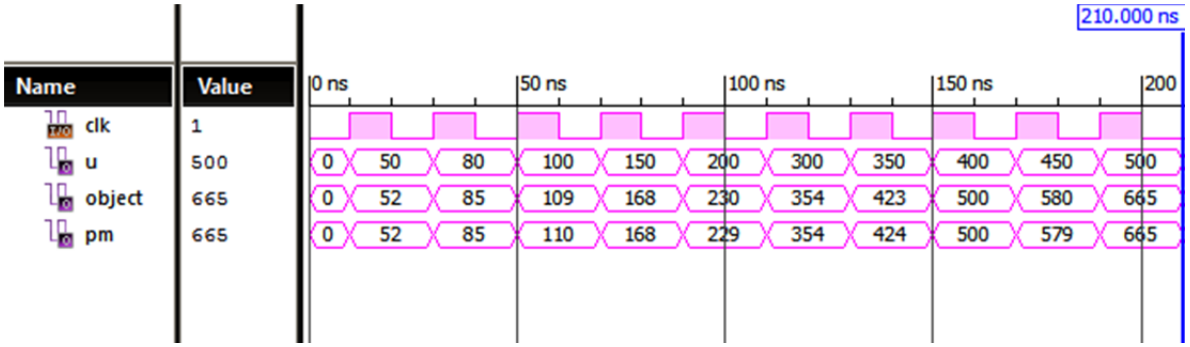

Рис. 5. Моделювання роботи прямої моделі

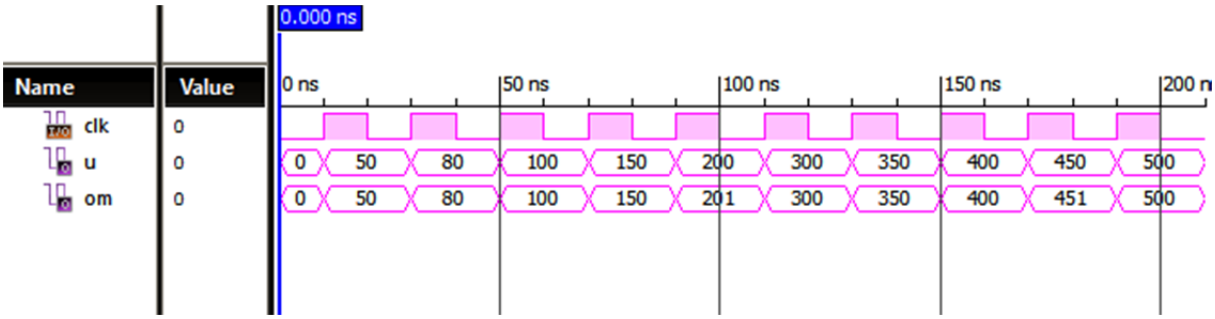

Рис. 6. Моделювання роботи інверсної моделі

На рисунках 5 та 6 представлено моделювання роботи структурних схем зображених на рисунках 1 та 4 відповідно. Обидві схеми побудовані в середовищі ISE Design Suite 13.2 та промодельовані в середовищі ISim. Як видно з рисунків 5 та 6 отримані нейромережеві моделі є адекватними.

Фільтр Калмана призначено для рекурсивного дооцінювання вектора стану апріорно відомої динамічної системи, тобто для розрахунку поточного стану системи необхідно знати поточний вимір, а також попередній стан самого фільтра. Фільтр Калмана реалізований в тимчасовому, а не в частотному поданні. Наочний приклад можливостей фільтра – отримання точних, безперервно обновлюваних оцінок положення і швидкості деякого об'єкта за результатами часового ряду неточних вимірювань його місця розташування. Фільтр Калмана оперує поняттям вектора стану системи і його статистичними описом. Вони базуються на дискретизованих за часом лінійних динамічних системах. Стан системи описується вектором кінцевої розмірності – вектором стану. У кожен такт часу лінійний оператор діє на вектор стану і переводить його в інший вектор стану, додається деякий вектор нормального шуму і в загальному випадку вектор управління, що моделює вплив системи управління. Опис фільтра Калмана представлено в [20]. Апаратно-програмна реалізація фільтра Калмана на FPGA дозволить синтезувати велику кількість паралельно пра-

цюючих фільтрів на одному кристалі, що в свою чергу дає можливість синтезу систем управління багатовимірними об'єктами(MIMO).

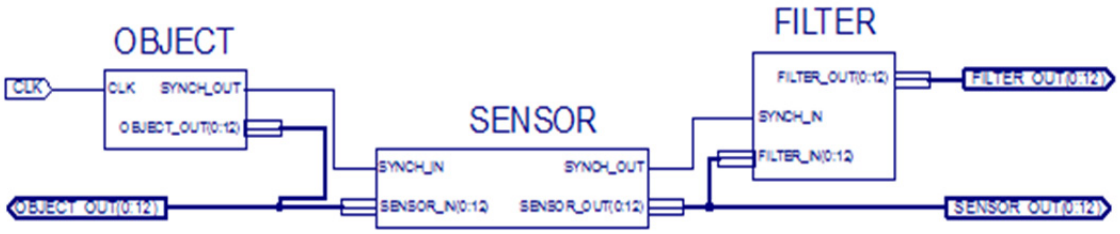

Рис. 7. Схема електрична принципова моделі для дослідження роботи фільтра Калмана

Апаратно-програмна реалізація фільтра Калмана на FPGA, виконується за наступним алгоритмом:

Крок 1: Виконати ініціалізацію змінних.

Задати дисперсію похибки датчика *sigmaeta*, середнє значення квадрату похибки на *k*-й ітерації *eopt_prev*, дисперсію похибки моделі *sigmaksi*. Задати *a,* коефіцієнт деякої відомої керуючої функції $u_k=a\cdot k$, де *k* – номер ітерації. При обчисленні використовувались десяткові числа, проте, так як в мові VHDL немає вбудованих засобів представлення не цілих десяткових чисел, то в програмному коді фільтра застосовуються коефіцієнти-змінні h=10000 і l=10 для перенесення коми, множення не цілих змінних в формулах. Таким чином отримуються великі цілі числа типу integer, з якими можна за допомогою вбудованих арифметичних операцій виконати необхідні обчислення. Отже в обчисленнях використовуються змінні з точністю до 0.0001. Результат представлений тільки цілою частиною.

Крок 2: Якщо на синхровхід надходить імпульс(перехід з 0 в 1) – перейти до **Крок 3**, інакше – чекати, поки не надійде імпульс (повторити **Крок 2**).

Крок 3: Зчитати з входу сигнал FILTER_IN покази датчика *z*<=FILTER_IN. *z* – покази датчика.

Крок 4: Обчислити квадрат середнього значення квадрату похибки

eopt:=*divide*(
(*sigmaeta*^{2})·(*eopt_prev*+*sigmaksi*^{2}),
(*sigmaeta*^{2}+*eopt prev*+*sigmaksi*^{2}))

Вбудованої операції ділення двох десяткових чисел в VHDL не передбачено, тому операцію ділення (ф-я *divide*) в вищенаведеній формулі було реалізовано відновлювальним алгоритмом ділення [21]. Ця функція приймає два числа типу integer (ділене та дільник) і приводить їх до типу unsigned. Тому отримавши два двійкових числа, дуже легко знайти частку маніпулюючи бітовими зсувами, ця операція швидка й використовує манімальну кількість ресурсів.

Крок 5: Обчислити коефіцієнти підсилення Калмана

$$k:=divide((eopt\cdot h),(sigmaeta^2));$$

В цій формулі також використовується перенесення коми і функція діленні *divide*.

Крок 6: Обчислити оптимальне відфільтроване значення

$$xopt:=(divide(xopt\_prev\cdot l, h)+a\cdot t\cdot l)\cdot(1\cdot h-k)/l+k\cdot z;$$

де *z*, як було вказано вище, покази датчика.

Крок 7: Ділимо на коефіцієнт *h*

$$xoptout:=xopt/h;$$

і результат фільтра (*xoptout*). Дрібна частина з точністю до 0.0001 враховується в обчисленнях, але в результаті нехтується.

Крок 8: Видати результат на вихід фільтра *FILTER_OUT*:

FILTER_OUT <= xoptout

Крок 9: Інкремент кроку ітерації $k=k+1$.

Розглянемо приклад реалізації фільтра Калмана на FPGA. Для моделювання його роботи побудуємо схему електричну принципову в програмному забезпеченні Xilinx ISE Design Suite 13.2, на якій в свою чергу змоделюємо роботу ОК, датчика та фільтра рис.6. На рис. 6 блок Object – модель об'єкта. Входи: CLK – вхід синхронізації. Виходи: SYNCH_OUT – вихід синхронізації; OBJECT_OUT – вихідна величина об'єкта. Блок SENSOR – модель датчика. Входи: SYNCH_IN – вхід синхронізації (з об'єктом); SENSOR_IN – вхід, на який надходить вимірювана величина. Виходи: SYNCH_OUT – вихід синхронізації(з фільтром); SENSOR_OUT – вихід, «виміряна» величина. Блок FILTER – фільтр Калмана. Отримує результати вимірювання датчика та за методом Калмана генерує значення, які ближчі до реальної вимірюваної величини**.** Входи: SYNCH_IN – синхронізація; FILTER_IN – вхід фільтра (для датчика з похибкою). Виходи: FILTER_OUT–

оптимальне згенероване значення вимірюваної величини.

Нехай динаміка об'єкта управління описується фізичним законом руху. Характеризується деякою вихідною величиною, яка змінюється за заданим законом $x_{k+1} = x_k + v_k dt$, де $k \in [0,100]$ – номер ітерації; $x_{k+1} \in [0,4950]$ – вихідна(вимірювана) величина на ітерації k+1; a=1 – коеф. нелінійного закону; $v_k = a \cdot k$ – відома керуюча функція. В побудованій моделі похибка датчика не перевищує $\eta_k = 50$, генерується псевдовипадкова послідовність значень.

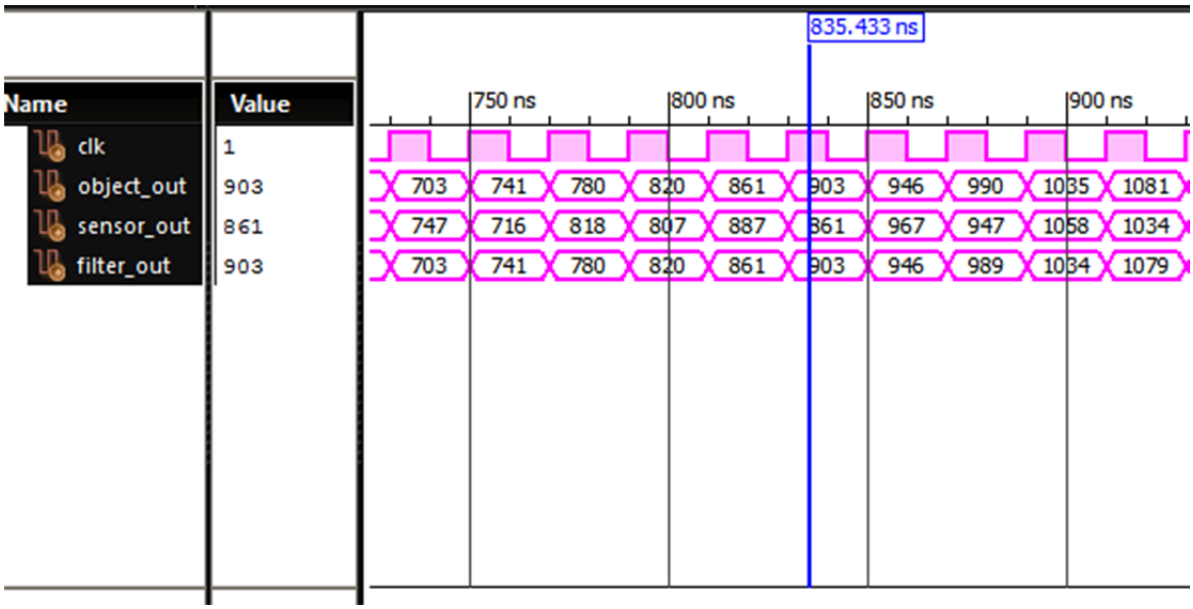

Рис. 8. Моделювання роботи фільтра Калмана в програмному пакеті ISim

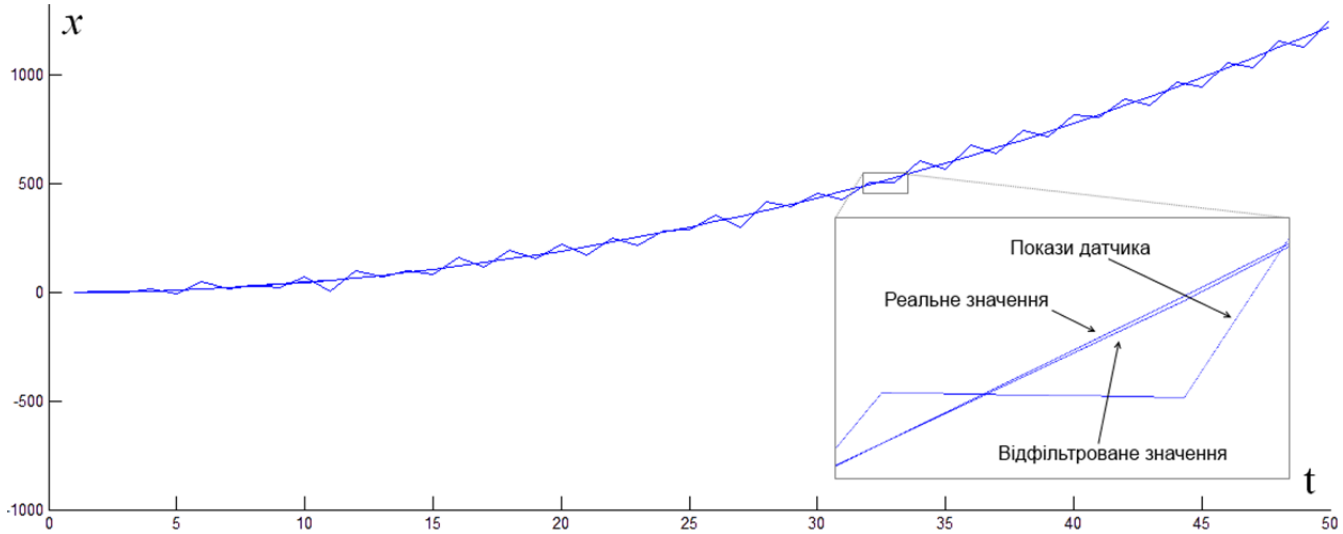

Рис. 9. Графік фільтрації сигналу по Калману

На часовій діаграмі, рис. 7, видно, що сигнал на виході датчика (sensor_out) спотворений похибкою, видно сигнал без похибки та роботу фільтра Калмана. При точності 0.0001 використовуваній в обчисленнях та досить великому проміжку значень вимірюваної величини, точність на виході фільтра знаходиться в межах 0.0001. Наглядний графік фільтрації сигналу по Калману(рис. 8) датчика (sensor_out), відфільтрованого значення(filter_out) та реальної величини(object_out) побудований в Matlab.

Висновок

В даній роботі розроблені та описані методи апаратно-програмної реалізації нейромережевих компонентів систем керування динамічними об'єктами. Розроблено метод апаратно-програмної реалізації прямої та інверсної моде-

лі об'єкта управління. На прикладі показана реалізація моделей такого об'єкта як індукційна піч. Отримані нейромережеві моделі є адекватними. Розроблено алгоритм реалізації фільтра Калмана, розглянуто приклад його реалізації та промодельовано його роботу. Розроблені в даній роботі нейромережеві компоненти дозволять будувати системи управління, які функціонують та адаптуються в режимі реального часу та формують функції управління будь-якої складності. Розробку та моделювання виконано на програмному забезпеченні Xilinx ISE Design Suite 13.2 та чіпі сімейства Spartan 3 – XC3S200.

Перелік посилань

1. Саймон Хайкин. Нейронные сети. Полный курс. – М.: Вильямс, 2006. – 1104с.
2. Терехов В.А., Ефимов Д.В., Тюкин И.Ю. Нейросетевые системы управления: Учеб. пособие для вузов. – М: Высшая школа. 2002. -183с.
3. Сигеру Омату. Нейроуправление и его приложения. – М.: ИПРЖР, 2000. – 272с.
4. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И.Д. Рудинского. – М.: Горячая линия – Телеком, 2007. – 452 с.
5. Егупов Н.Д. Методы робастного, нейро-нечеткого и адаптивного управления. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 744 с.
6. Гильгурт С.Я. Анализ применения реконфигурируемых вычислителей на базе ПЛИС для реализации нейронных сетей // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ НАН України. – Вип. 37. – Київ: 2006. – С. 168-174 с.
7. Логовский А. Технология ПЛИС и ее применение для создания нейрочипов. // Открытые системы. – 2000. – № 4. – С. 100-102 с.
8. Соловьев В. Проектирование цифровых систем на основе ПЛИС. – М.: Радио и связь, 2003. – 376с.
9. Тарасов И.Е. Разработка цифровых устройств на основе ПЛИС Xilinx® с применением языка VHDL. – М.: Горячая линия – Телеком, 2005. – 252 с.
10. Сергиенко А.М. VHDL для проектирования вычислительных устройств – К ЧП «Корнейчук», ООО «ТИД «ДС», 2003. – 208 с.
11. Кунцевич В.М. Управление в условиях неопределенности: гарантированные результаты в задачах управления и идентификации. Монография. — К.: Наук. думка, 2006. — 264 с.
12. Kalman R.E. Identification of Noisy Systems. Russian Mathematical Surveys, 1985, no. 40 (4), pp. 25–42.
13. Льюнг Л. Идентификация систем. Теория для пользователя. – М.: Наука, 1991.
14. Кравець П.І., Шимкович В.М., Зубенко Г.А. Технологія апаратно-програмної реалізації штучного нейрона та штучних нейронних мереж засобами FPGA / Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2012. – №55. 174-180с.
15. Кравець П.І., Лукіна Т.Й., Шимкович В.М., Ткач І.І. Розробка та дослідження технології оцінювання показників нейромережевих моделей MIMO-об'єктів управління / Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2012. – №57. 144-150с.
16. Кравец П.И., Шимкович В.Н. Метод оптимизации весовых коэффициентов нейронных сетей с помощью генетического алгоритма при реализации на программируемых логических интегральных схемах / Международный научно-технический журнал «Электронное моделирование». – 2013. – 35, №3. – С. 65-75.
17. Neural networks for control systems: A survey / K. J. Hunt, D. Sbarbaro, R. Zbikowski, P. J. Gawthrop // Automatica. 1992. Vol. 28. № 6. P. 1083 – 1112.
18. Горбат А. Н. Обучение нейронных сетей. – М.: СП «ParaGraph», 1990. – 160с.
19. Intelligent control systems: Theory and applications. Madan M. Gupta, Naresh K. Sinha. 1995. P. 820.
20. Пузырев В.А. Управление технологическими процессами производства микроэлектронных приборов. – М.: Радио и связь, 1984 – 160 с.
21. http://habrahabr.ru/post/166693
22. http://en.wikipedia.org/wiki/Division_algorithm#Restoring_division