Міністерство освіти і науки України
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Data sorting hardware means with real-time merger method

Студент гр. КІ - 21
Макух В. С.

Тернопіль - 2019

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

НВІС – надвелика інтегральна схема

ОП – операційними пристроями

ПЕ – процесорний елемент

ПЗ – програмне забезпечення

ФО – функціональні оператори

ПГА – потоковий граф алгоритму

БПП - багатопортова пам’ять.

CPU англ. Central Processing Unit – центральний процесор

CUDA англ. Compute Unified Device Architecture – програмно-апаратна архітектура паралельних обчислень

GPU англ. Graphics Processing Unit – графічний процесор

ВСТУП

Актуальність теми. Сучасний етап розвитку інформаційних технологій характеризується накопиченням великих масивів даних. При опрацюванні таких масивів найчастіше доводиться використовувати операції сортування, метою якого є прискорення пошуку необхідної інформації. Основним шляхом підвищення швидкодії сортування великих масивів даних є розпаралелювання процесу сортування та використання масово-паралельних обчислюваних засобів із великим обсягом пам'яті. До таких засобів відноситься графічні процесори (GPU – Graphics Processing Unit), які є процесорами класу SIMD (Single Instruction Multiple Data). Особливістю SIMD процесорів є те, що в них одна операція використовується одночасно для опрацювання множини незалежних даних. Для розробки програмного забезпечення сортування великих масивів даних, частина якого працює на CPU (центральний процесор), а частина на GPU доцільно використати кросплатформову систему компіляції та виконання програм CUDA.

Програмна реалізація сортування даних на GPU вимагає вдосконалення та розроблення нових паралельних алгоритмів сортування масивів даних. Алгоритми з паралелізаціюсортування масивів даних для реалізації на GPU повинні бути:

– гарно структурованими з налагодженим переміщенням масивів даних;

– сути структуровані на однотипних операціях з регулярними та локальними зв'язками;

– використовувати конвеєрний підхід та паралелізм у просторі.

У зв'язку з вищезазначенимт критеріями набуває проблема вдосконалення та розроблення нових алгоритмів паралелізації сортування чисел і їх організація на (GPU) на графічні процесори.

Мета і завдання дослідження. Метою роботи є вдосконалення технології сортування масивів чисел методом злиттям. Для виконання мети в роботі необхідно виконати такі завдання:

– дослідити алгоритми сигмоїдальної апроксимації;

– визначети та розглянути основні етапами розробки потокового графу;

– розробити орієнтований на архітектуру CUDA конкретизований потоковий граф паралельного апаратної реалізації використанням вдосконаленого методу;

Об'єкт дослідження – алгоритми та апаратна реалізація сигмоїдальних функцій активації.

Предмет дослідження- методи, сигмоїдальних функцій активацій.

Методи досліджень. Для розв'язання задач які представлені у кваліфікаційній роботі використано: дослідження методів і алгоритмів реалізації функцій активації, архітектуру процесора CUDA.

Наукова новизна одержаних результатів:

– модифіковано методи та розроблено структури апроксимації паралельного сортування масивів чисел, використовуючи кусково-лінійну апроксимацію і апроксимацію поліномом другого порядку;

– виконано оцінку точності апроксимації цими методами паралельного сортування масивів чисел та її похідної. Розглянуті методи реалізовані мовою C++ на CUDA;

– приведено порівняння по необхідних апаратних ресурсах і швидкодії.

Практичне значення. Розроблено орієнтований на архітектуру CUDA конкретизований потоковий граф паралельного апаратної реалізації використанням вдосконаленого методу

Публікації та апробація випускної кваліфікаційної роботи. Отримані результати апробовані в межах III науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі» Західноукраїнського національного економічного університету та опубліковано дві тези доповіді по темі роботи [1,2].

1 АНАЛІЗ ГАЛУЗЕЙ ЗАСТОСУВАННЯ, МЕТОДІВ ТА ЗАСОБІВ СИГМОЇДАЛЬНИХ ФУНКЦІЙ АКТИВАЦІЇ

1.1 Галузі застосування засобів сигмоїдальних функцій активації

В останні десятиліття зростає інтерес до апаратної реалізації штучних нейронних мереж (ARN). Про це свідчить кількість публікацій на цю тему. Це пов'язано насамперед із швидким розвитком елементної бази, яка використовується при реалізації цифрових ANN (надвеликі інтегральні схеми - VLIS). Однією з актуальних проблем, яка залишається, є збільшення швидкості роботи нейронних мереж. Одним із способів пришвидшити їх роботу за допомогою одночасності є впровадження їх на CUDA.

Швидкість роботи штучних нейронів залежить від ПАРАЛЕЛЬНОГО СОРТУВАННЯ МАСИВІВ ЧИСЕЛ активації. ЇЇ реалізація на CUDA вимагає значних апаратних ресурсів [1, 2, 3]. В роботі [1] приведено огляд основних методів, які використовуються при реалізації ПАРАЛЕЛЬНОГО СОРТУВАННЯ МАСИВІВ ЧИСЕЛ .

У сучасних комп'ютерах висока продуктивність обробки даних досягається за допомогою просторового та часового паралелізму. Особливий інтерес представляє концепція, за допомогою якої підвищення продуктивності комп'ютерного пристрою досягається шляхом наближення його структури до структури виконуваного алгоритму [1]. Трансформація алгоритму в його апаратну модель відбувається шляхом повного апаратного відображення графіку потоку виконуваного алгоритму (PGA) операційними пристроями (ОР), які виконують функціональні оператори (FO) алгоритму і взаємопов'язані згідно алгоритму графік [2]. Пристрої, структура яких адаптована до PGA, мають наступні переваги:

ви можете виконати операцію з усіма операндами, незалежно від стану інших операцій (синхронізація не потрібна);

обмін даними між операціями чітко визначений; операції контролюються передачею даних між ними; алгоритм виконується за один цикл, що забезпечує багаторазове прискорення конкретного завдання.

Але є багато питань, що стосуються, насамперед, способу побудови та зміни графіків алгоритмів для можливості їх апаратної реалізації [1, 2, 3, 4].

Розглянуто різні варіанти синтезу паралельно-обчислювальних сортувальних пристроїв та їх проектування від програмного опису алгоритму до апаратної реалізації з можливістю зміни ширини паралельної форми.

Доволі високою є доля сортування серед операцій, виконуваних комп'ютером, задача сортування методом злиття є однією з однотипних проблем покрокової обробки даних, і, звичайно, розуміється як завдання розміщення елементів неупорядковоного набору значень $\overline{X} = \{x_1, x_2, ..., x_N\}$ в порядку монотонного зростання або спадання $\overline{X} \approx \overline{Y} = \{(y_1, y_2, ..., y_N) : y_1 \le y_2 \le ... \le y_N\}$. Сортування даного типу будуються комутуючі мережі вони забезпечують обмін даними між компонентами комп'ютерної системи [2]. Алгоритми сортування в даному випадку зручний засіб для визначення позитивних сторін моделі та для порівняння між собою різнис моделей.

Трудомісткість визначення процедури є надзвичайно високою. Так, для деяких відомих простих методів кількість операцій які необхідні визначається залежністю від числа даних, що впорядковують $t_S \approx N \log_2 N$ [5]. Прискорення виконання алгоритму сортування виконується декількома операційними пристроями одночасно значно прискорює час опрацювання алгоритму. Серед алгоритмів сортування паралельні операції можуть виконуватися для алгоритмів, в яких послідовність операцій залежить лише від кількості вхідних даних і не залежить від значень їх ключів (неадаптивні алгоритми) [6]. Серед алгоритмів сортування є неадаптивні: алгоритм сортування методом «парно-непарної» перестановки [7], алгоритм сортування модифікованим методом «бульбашок» [6], алгоритм сортування методом Бутчера [5]]. Під час проектування були деякі недоліки, які зі складністю проектів стають досить багато суттєвими, одним із недоліків є те, що часто доводиться виконувати ручну роботу для паралелізації алгоритмів або зміни ширини паралельної фігури. Звідси випливає, що така робота вимагає додаткового часу, високої кваліфікації, і, тим не менше, не гарантує збереження простого рішення, отже, призводить до

того, що на якомусь етапі складності проект стає збитковим. Це змушує шукати нові шляхи для пришвидшення дизайну.

Використання нейронних мереж є одним з основних методів для прискорення проектування інструментів пошуку мінімальних та максимальних елементів у наборі даних.

Вищезазначені методи паралельного сортування засновані на застосуванні тієї самої базової операції "порівняння та переставлення", яка полягає у порівнянні пари ключів із набору даних для сортування та перестановки цих значень, якщо їх порядок не відповідає сортуванню умови. Ці методи відрізняються лише порядком порівняння пар, а основна операція "порівняти та переставити" однакова для таких алгоритмів. Отримані PGA цих алгоритмів для сортування 16 вхідних значень показані на рисунку 1.1

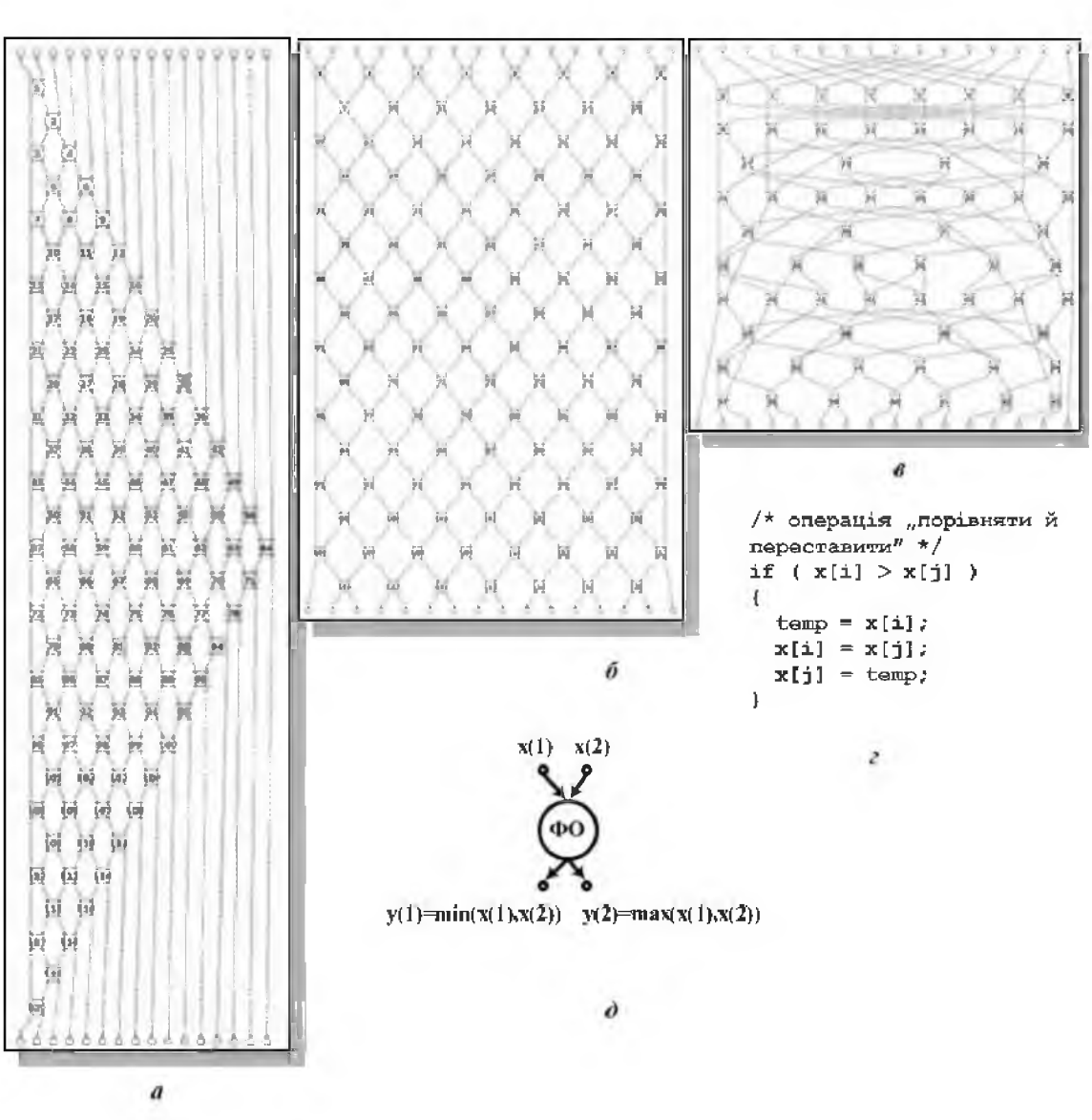

Рисунок 1.1 – Сортування шістнадцяти значень: а – ПГА сортування методом “бульбашки”; б – ПГА сортування перестановки; в – ПГА Бетчера; г, д – операція “порівняти й переставити”

Для однакової кількості вхідних значень ці алгоритми мають однакову ширину PGA, але виконують різну кількість операцій "порівняння та переставлення" та мають різну кількість рівнів.

Загальна кількість PGA дорівнює загальній кількості операцій "порівняння та перестановка". Для алгоритмів сортування методом "парно-непарної перестановки" та модифікованого методу "бульбашок" для вхідних значень кількість FD становить [2], а для алгоритму сортування методом Бетчера - що підтверджено моделюванням результати. Висота PGA для цих алгоритмів різна і є найбільшою для модифікованого алгоритму сортування методом «бульбашок» - [2]. Для алгоритму сортування „парно-непарних” перестановок висота PGA дорівнює кількості вхідних значень, а для алгоритму сортування м’ясника - найменша і дорівнює. Оскільки алгоритми виконують одну операцію "порівняння та переставлення", можна припустити, що затримка часу всіх FD однакова і дорівнює одиниці. Апаратна складність цих обчислювальних пристроїв дорівнюватиме кількості PGA FD, а затримка часу буде дорівнювати кількості рівнів PGA.

Сфери застосування нейромереж:

– Економіка та бізнес: прогнозування ринку, автоматична торгівля, оцінка ризику дефолту, прогнозування банкрутства, оцінка нерухомості, виявлення завищених та недооцінених компаній, автоматичний рейтинг, оптимізація портфеля, оптимізація товарних та грошових потоків, автоматичне зчитування чеків та форм, безпека транзакцій на пластикових картках.

– Медицина: обробка зображень медичного характеру, моніторинг стану пацієнтів, діагностика пацієнтів, діагностика ефективності лікування, очищення показань приладів від незрозумілих завад.

– Авіоніка: дослідження поведінки учні автопілот, адаптивне пілотування сильно пошкодженого літака, розпізнавання сигналів радарів.

– Зв'язок: надшвидке кодування, оптимізація і маршрутизації пакетів, стиснення інформації.

– Інтернет: пошук інформації, електронні секретарі і агенти користувача в мережі, фільтрація інформації в push-системах, Коллаборативні фільтрація, рубрикація новинних стрічок, адресна реклама, адресний маркетинг для електронної торгівлі.

Автоматизація виробництва: комплексна діагностика якості продукції (ультразвук, оптика, гамма-випромінювання, ...), оптимізація виробництва, моніторинг та візуалізація інформації, запобігання надзвичайним ситуаціям, робототехніка.

Політичні технології: аналіз та узагальнення соціологічних описів, прогнозування рейтингової динаміки, виявлення значущих факторів, об'єктивна кластеризація електорату, візуалізація соціальної динаміки населення

Systems Системи безпеки та безпеки: системи ідентифікації особи, розпізнавання голосу, розпізнавання натовпу, розпізнавання номерних знаків, аналіз аерокосмічних зображень, моніторинг потоку інформації, виявлення підробок.

Put Введення та обробка інформації: Обробка рукописних чеків, розпізнавання підписів, відбитків пальців та голосів. Введення фінансових та податкових документів у комп'ютер.

Розвідка: аналіз сейсмічних даних, асоціативні методи розвідки корисних копалин, оцінка польових ресурсів.

Існує велику кількість широко розповсюджених комерційних програм універсальних нейронних мереж (Statistica Neural Networks, NeuroShell, Matlab Neural Network Toolbox, NeuroSolutions, BrainMaker). Існує набагато більше спеціалізованих, некомерційних або дослідницьких програм, розроблених вченими для власних потреб.

Нейронні мережі широко використовуються сьогодні: нейронні мережі - це не що інше, як новий інструмент для аналізу даних. А для інших краще, щоб він міг використовувати цей фахівець у своїй предметній галузі. Основні труднощі на шляху до ще більшого розповсюдження нейротехнологій - у новому широкому колі професіоналів своєчасно формулюють свої проблеми, що дозволяє просте рішення нейронної мережі.

Основні цікаві особливості нейронних мереж на практиці такі:

- Наявність алгоритмів швидкого навчання: мережу можна швидко вивчити на звичайному комп'ютері, навіть маючи сотні вхідних сигналів і десятки тисяч опорних ситуацій. Тому нейронні мережі мають широкий спектр застосування і дозволяють вирішувати складні проблеми прогнозування, класифікації або діагностики.

Bility Можливість роботи над великою кількістю неінформативних, шумних вхідних сигналів - їх не потрібно попередньо перевіряти, нейронна мережа сама визначить їх як непридатні для вирішення проблеми і може чітко їх відхилити.

Bility Здатність працювати з корельованими незалежними змінними над різними типами інформації - дискретними, кількісними та якісними, що часто ускладнює статистичні методи

Нейронна мережа може одночасно вирішувати велику кількість проблем за допомогою одного набору вхідних сигналів - при великій кількості виходів можна передбачити значення декількох показників.

Алгоритми навчання висувають дуже мало вимог до структури нейронної мережі та властивостей її нейронів. Отже, якщо у вас є спеціальні знання або якщо у вас є особливі вимоги, ви можете цілеспрямовано вибрати тип та властивості нейронів та нейронної мережі, скласти структуру нейронної мережі вручну з окремих елементів та визначити бажані властивості для кожного з цих елементів.

1.2 Методи визначення сигмоїдальних функцій активації

Оскільки всі штучні нейронні мережі базуються на концепції нейронів, зв'язків та передавальних функцій, між різними структурами або архітектурами нейронних мереж існують подібності. Більшість змін зумовлені різними правилами навчання. Розглянемо деякі найбільш відомі штучні нейронні мережі.

Персептрон Розенбаллата

Персептрон Розенбаллата вважається першою моделлю нейронних мереж. В основі багатьох видів штучних нейронних мереж прямого поширення лежить теорія персептронів, яка є класичною для вивчення.

– Одношаровий персептрон здатний розпізнавати найпростіші зображення. Визначення зваженої суми сигналів вхідних елементів виконується окремим нейроном, який віднімає значення зсуву і передає результат через жорстку порогову функцію, вихід якої дорівнює або. Залежно від значення вихідного сигналу приймається рішення: $+1$ - вхідний сигнал належить до класу A,

– -1 - вхідний сигнал належить до класу B.

Області прийняття рішень визначають, які вхідні зображення присвоюються класу, а які - класу. Персептрон, що складається з одного нейрона, утворює дві ключові області, розділені гіперпланом.

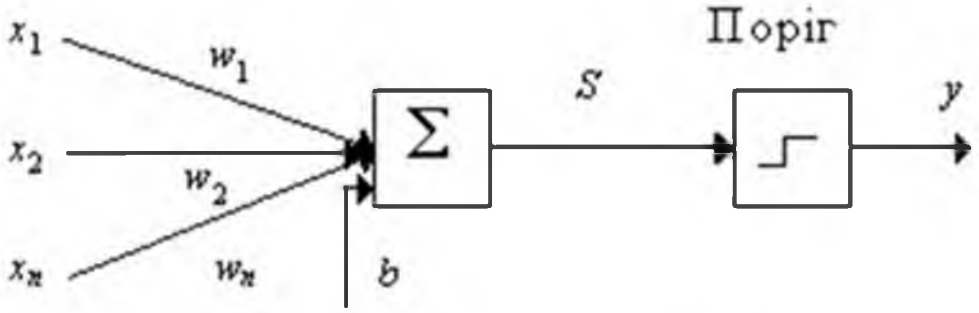

Рисунок 1.2 – Схема нейрона

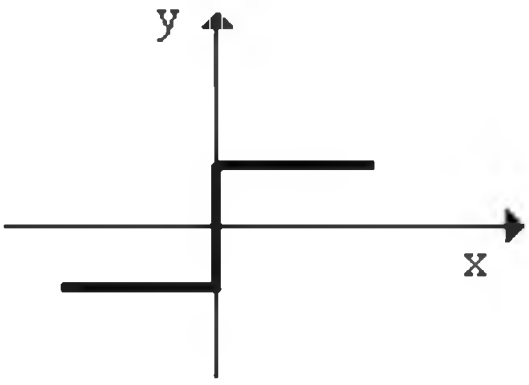

Рисунок 1.3 – Графік передатної функції

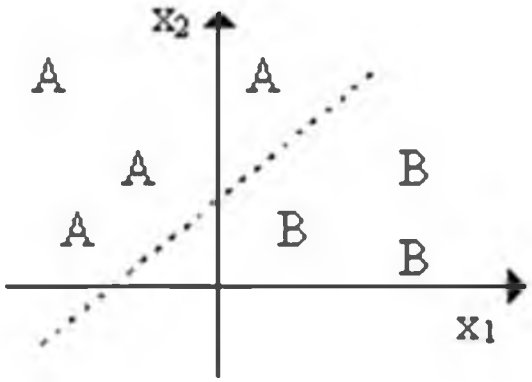

Рисунок 1.4 – Поділяюча поверхня

На рисунку показано випадок з розмірністю вихідного сигналу - 2. Розділювальна поверхня - це пряма лінія на площині. Рівняння, яке визначає лінію поділу, залежить від значень синаптичних ваг та переміщення.

Нейронна мережа зворотного поширення

Архітектура FeedForward BackPropagation була розроблена на початку 1970-х років кількома незалежними авторами: Werbor; Паркер; Румелхарт, Хінтон і Вільямс. В даний час популярною, ефективною та простою моделлю навчання для складних багатошарових мереж є парадигма BackPropagation. Його використання в різних типах додатків породило великий клас нейронних мереж з різними структурами та методами навчання.

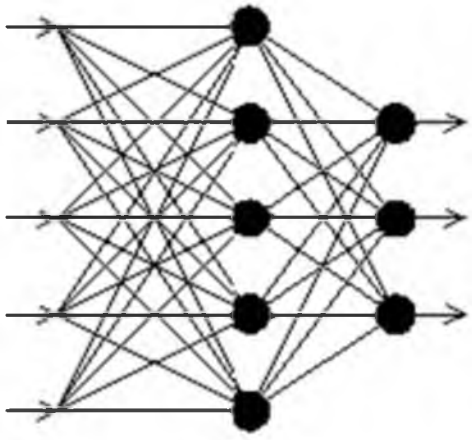

Рисунок 1.5 – Мережа зворотного поширення похибки

Типова мережа BackPropagation має вхідний рівень, вихідний рівень і принаймні один прихований рівень. Теоретично обмежень щодо кількості прихованих шарів немає, але на практиці застосовують один або два.

Нейрони організовані в шарувату структуру з прямою передачею сигналу (вперед). Кожен нейрон мережі виробляє зважену суму своїх входів, передає це значення через передавальну функцію і виводить вихідне значення. Мережа може імітувати функцію практично будь-якої складності, а кількість шарів і кількість нейронів у кожному шарі визначають складність функції.

Визначення кількості проміжних шарів та кількості нейронів у них є важливим аспектом мережевого моделювання. Більшість дослідників та інженерів використовують загальні правила, серед яких:

- Кількість входів і виходів мережі визначається кількістю вхідних і вихідних параметрів досліджуваного об’єкта, явища, процесу тощо.

Якщо складність взаємозв'язку між отриманими та бажаними вихідними даними зростає, кількість нейронів прихованого шару також повинна зростати.

- Якщо змодельований процес можна розділити на багато етапів, необхідний додатковий прихований шар (шари). Якщо процес не розділений на етапи, то додаткові шари можуть дозволити запам'ятовування і, як наслідок, неправильне загальне рішення.

Взявши до уваги всі правила, визначивши кількість шарів і кількість нейронів у кожному з них, потрібно знайти значення для синаптичних шкал і порогів мережі, які здатні мінімізувати помилку отриманого результату . Ось чому існують алгоритми навчання, де мережева модель пристосовується до наявних навчальних даних. Перебираючи мережу всіх прикладів навчання та порівнюючи згенеровані вихідні значення з бажаними значеннями, визначається помилка для конкретної моделі мережі. Набір помилок створює функцію помилок, значення якої можна розглядати як помилку мережі. Найчастіше сума квадратів помилок використовується як функція помилок.

Розглядаючи концепцію поверхні станів можна краще зрозуміти алгоритм навчання мережі зворотного поширення. Кожне значення синаптичних ваг та порогових значень мережі (вільні параметри номера моделі) відповідає одному виміру в багатовимірному просторі. Розмір відповідає помилці мережі. Для різних комбінацій ваг відповідна похибка мережі може бути представлена точкою в -вимірному просторі, всі ці точки утворюють поверхню - поверхню

станів. Знайти найнижчу точку на багатовимірній поверхні є метою навчання нейронних мереж.

Поверхня штатів має складну будову і досить багато неприємні властивості, зокрема, наявність місцевих мінімумів (точок, найнижчих у своєму конкретному середовищі, але вищих за загальний мінімум), рівнинних ділянок, сідлових точок та довгих вузьких ярів. Неможливо визначити розташування глобального мінімуму на поверхні станів аналітичними засобами, тому тренування нейронної мережі є по суті дослідженням цієї поверхні.

Враховуючи початкові конфігурації ваг та порогів (із випадково обраної точки на поверхні), алгоритм навчання поступово знаходить загальний мінімум. Розраховується вектор градієнта похибки поверхні, який вказує напрямок найкоротшого спуску на поверхню з даної точки. Щоб зменшити помилку, потрібно трохи перейти до неї. Зрештою, алгоритм зупиняється внизу, що може бути лише локальним мінімумом (в ідеалі глобальним мінімумом).

Складність полягає у виборі довжини сходинок. При великій довжині кроку зближення буде швидшим, але існує небезпека перестрибнути розчин або піти в неправильному напрямку. З невеликим кроком буде виявлено правильний напрямок, але кількість ітерацій збільшується. На практиці розмір кроку приймається пропорційно крутості схилу з деякою константою, яка є швидкістю навчання. Правильний вибір швидкості навчання залежить від конкретного завдання і проводиться експериментально. Ця константа також може залежати від часу, зменшуючись у міру просування алгоритму.

Алгоритм є ітераційним, його етапи називаються епохами. Всі приклади введення для кожної епохи по черзі подаються на вхід мережі, початкові значення мережі порівнюються з бажаними значеннями і обчислюється похибка. Значення помилок, а також градієнт поверхні станів використовуються для виправлення ваг, і дії повторюються. Коли пройдено певну кількість епох, або коли помилка досягне певного рівня незначності, або коли помилка перестане зменшуватися (користувач зазвичай вибирає бажаний критерій зупинки), процес навчання зупиняється.

Мережа Кохонена

На початку 1980-х рр. Була розроблена мережа Тойво Кохонена, яка принципово відрізняється від розглянутих вище мереж, різниця полягає в тому, що використовується неконтрольоване навчання, а навчальний набір складається лише із значень вхідних змінних.

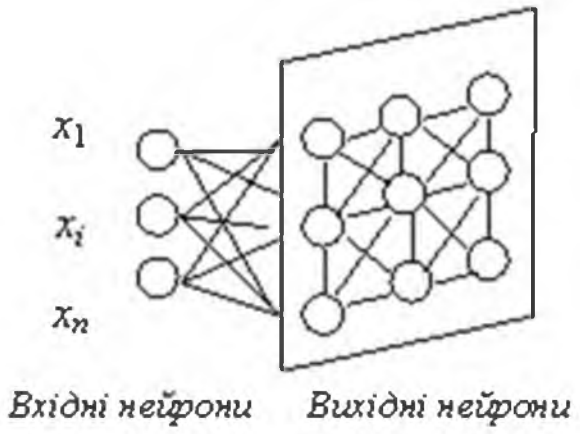

Рисунок 1.6 – Мережа Кохонена

Мережа в навчальних даних розпізнає кластери, потім дані розподіляються у відповідні кластери. Якщо наступна мережа зустріне набір даних, який не схожий на жоден із відомих зразків, вона призначає його новому кластеру. Проблеми класифікації можуть бути вирішені мережею, якщо дані містять мітки класів. Мережі Кохонена також можуть бути використані в задачах, де класи відомі - перевага буде в здатності мережі виявляти подібність між різними класами.

Мережа Кохонена має лише два шари: вхідний та вихідний. Елементи карти розташовані в якомусь просторі, як правило, двовимірному. Мережа Кохонена навчається методом послідовних наближень. Під час навчання дані надходять на входи, але мережа не коригується до еталонного значення виводу, а до закономірностей вхідних даних. З випадково вибраного початкового місця розташування центрів починається навчання.

У процесі послідовного надходження на вхід мережі навчальних прикладів визначається найбільш схожий нейрон (той, у якому скалярний добуток ваг і вхідного вектора мінімальний). Цей нейрон оголошений переможцем і є центром у регулюванні ваги сусідніх нейронів. Це правило навчання передбачає

"конкурентне" навчання з урахуванням відстані нейронів від "нейрона-переможця".

Для найбільшого збігу з вхідними даними, навчання полягає не в тому, щоб мінімізувати помилку, а в тому, щоб відрегулювати шкали (внутрішні параметри нейронної мережі).

Основний ітераційний алгоритм Кохонена послідовно проходить ряд епох, кожна з яких обробляє один приклад із навчальної вибірки. Вхідні сигнали послідовно подаються в мережу, і бажані вихідні сигнали не визначаються. Після подання достатньої кількості вхідних векторів синаптичні ваги мережі стають здатними ідентифікувати кластери. Ваги організовані таким чином, що топологічно близькі вузли реагують на подібні вхідні сигнали.

В результаті алгоритму центр кластера встановлюється в певному положенні, що задовольняє кластерним прикладам, для яких цей нейрон є "переможцем". В результаті мережевого навчання необхідно визначити ступінь сусідства нейронів, тобто середовища нейрона-переможця, який представляє велику кількість нейронів, що оточують нейрон-переможець.

Спочатку велика кількість нейронів належить до навколишнього середовища, потім його розмір поступово зменшується. Мережа утворює топологічну структуру, в якій подібні приклади утворюють групи прикладів, тісно розташованих на топологічній карті.

Мережа Хопфілда

У 1982 році Джон Хопфілд вперше представив свою асоціативну мережу в Національній академії наук. Тому мережеву парадигму називають мережею Хопфілда, на честь Хопфілда та нового підходу до моделювання. Мережа Хопфілда базується на аналогії фізики динамічних систем. Початкові додатки мережі включали асоціативну або орієнтовану на вміст пам'ять та вирішували проблеми оптимізації.

Мережа Хопфілда використовує три шари: вхідний рівень, рівень Хопфілда і вихідний рівень. Кількість нейронів у кожному шарі однакова. Виходи нейронів вхідного шару подаються на входи відповідних нейронів шару Хопфілда. Тут з'єднання мають фіксовану вагу. Виходи шару Хопфілда підключені до входів усіх нейронів шару Хопфілда, крім нього самого, а також

до відповідних елементів вихідного шару. Мережа спрямовує дані з вхідного рівня на рівень Хопфілда, направлення відбувається під час навчання. Поки не завершиться певна кількість циклів шару Хопфілда, він коливається, а поточний стан сигналів нейронів шару передається вихідному шару. Цей стан відповідає зображенню, яке буде зберігатися в Інтернеті.

Мережеве навчання вимагає, щоб навчальний образ подавався одночасно на вхідному та вихідному рівнях. Рекурсивний характер шару Хопфілда забезпечує засіб корекції всіх ваг суглобів. Відповідні пари вхід-вихід повинні бути різними для належного навчання мережі.

Якщо мережа Hopfield використовується як адресація вмісту, вона має два основних обмеження.

Суворе обмеження кількості зображень, які можна зберегти та точно відтворити. Якщо зберігається забагато зображень, мережа може або не відповідати новому неіснуючому зображенню, інакше як усі запрограмовані зображення. Приблизно 15% кількості нейронів - це обмеження ємності пам'яті для мережі в рівні Хопфілда.

Шар Хопфілда може стати нестабільним, якщо конкретні дослідження занадто подібні. Шаблон зображення вважається нестабільним, коли він застосовується за нульовий час, а мережа збігається з деяким іншим зображенням із навчального набору. Вибравши більше ортогональних прикладів навчання, цю проблему можна вирішити.

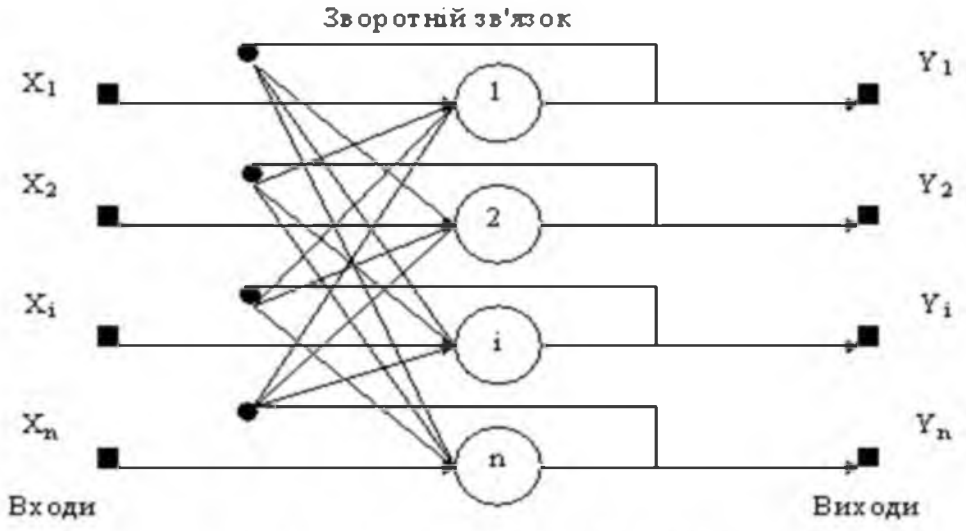

Рисунок 1.7 – Мережа Хопфілда

Існує набір двійкових сигналів (зображення, оцифровка звуку, інші дані, що описують об’єкти або характеристики процесу) для вирішення проблеми асоціативної пам’яті, яка вважається зразковою. Мережа повинна мати можливість вибрати ("запам'ятати" з часткової інформації) відповідний зразок із шумного сигналу, поданого на його вхід, або "дати висновок", що вхідні дані не відповідають жодному з вибірок.

У загальному випадку будь-який сигнал може бути описаний вектором - кількістю нейронів у мережі та величиною вхідних та вихідних векторів. Кожен елемент дорівнює або, або. Позначимо вектор, що описує -й зразок, а його компоненти відповідно - - кількістю зразків. Якщо мережа розпізнає (або «запам'ятовує») шаблон на основі представлених їй даних, його виходи будуть містити його, тобто де - вектор вихідних значень мережі :. В іншому випадку вихідний вектор не відповідає жодному зразків.

Якщо, наприклад, сигналами є певне зображення, то, графічно відображаючи дані з мережевого виходу, ви можете побачити картинку, яка повністю збігається з однією з моделей (у разі успіху) або «вільної імпровізації» мережі (у разі несправності).

Мережа Хемінга

Розширенням мережі Хопфілд є мережа Хеммінга. У середині 1980-х цю мережу розробив Річард Ліппман. Мережа Хемінга реалізує класифікатор, заснований на найменшій похибці для двійкових вхідних векторів, де похибка визначається відстанню Хемінга. Кількість бітів, які відрізняються між двома відповідними вхідними векторами фіксованої довжини, визначається як відстань Хемінга. Один вхідний вектор - беззвучний приклад зображення, інший - спотворене зображення. Вихідний вектор навчального набору - це вектор класів, до яких належать зображення. У режимі навчання вхідні вектори поділяються на категорії, для яких відстань між вибірковими вхідними векторами та поточним вхідним вектором є мінімальною.

Мережа Хемінга має три рівні: вхідний рівень із кількістю вузлів, скільки окремих двійкових функцій; рівень категорій (шар Хопфілда), з кількістю вузлів, скільки категорій або класів; вихідний рівень, що відповідає кількості вузлів на рівні категорії.

Мережа - це проста архітектура прямого розподілу з вхідним рівнем, повністю пов’язаним із рівнем категорій. Кожен нейрон у шарі категорії зворотньо зв’язаний з кожним нейроном у тому самому шарі та безпосередньо пов’язаний із вихідним нейроном. Завдяки конкуренції формується вихід із шару категорії до початкового шару.

Мережеве навчання Хемінга подібне до методології Хопфілда. Вхідний рівень отримує бажане навчальне зображення, а вихідний рівень вихідного рівня отримує значення бажаного класу, до якого належить вектор. Вихідні дані містять лише значення класу, до якого належить вхідний вектор.

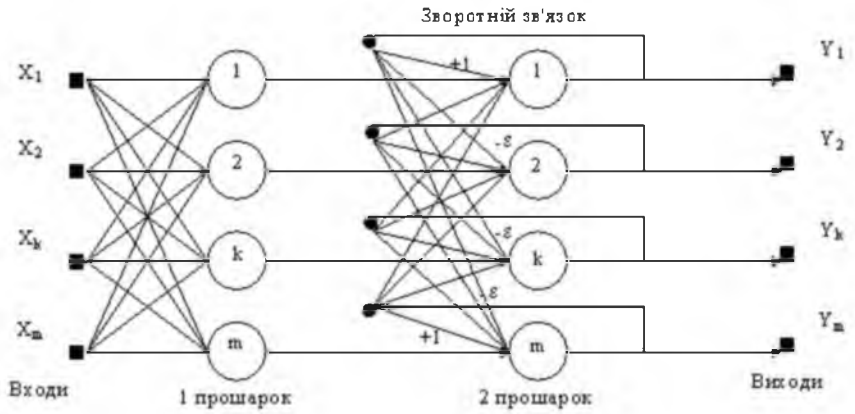

Рисунок 1.8 – Мережа Хемінга

1.3 Структури апаратних засобів сигмоїдальних функцій активації

При впровадженні замінних пристроїв необхідність вивчення принципів обміну інформацією. Обмін інформацією може бути синхронним та асинхронним. Вибір принципів обміну є важливим фактором, який впливає не лише на пропускну здатність, але і на фізичній довжині обмінного каналу та кількості пристроїв, які можна отримати підключити.Синхронний принцип обміну

За цим принципом обмін даними передається в блоках, обмежених часом. Передача даних базується на координації таймерів передавального пристрою та приймального пристрою. Тактові сигнали задають певний інтервал часу, протягом якого інформація зчитується з каналу обміну. Для початку

синхронізації використовуються спеціальні сигнали синхронізації. Розташування сигналів адреси та даних щодо сигналів синхронізації визначається фіксованим протоколом. Для того, щоб вирішити, що робити наступного разу, додаткова логіка майже не потрібна, тому синхронна передача може бути швидкою та дешевою. Канал обміну використовується більш ефективно при синхронній передачі, підтримується висока ефективність, швидкість передачі даних та вбудований надійний механізм виявлення помилок. Основним недоліком є те, що через проблему тактових спотворень шина при синхронній передачі не може бути довгою, а інтерфейсне обладнання дорожче і складніше.

Принцип асинхронного обміну

Передача обмежена часом. Додаткові затримки зв'язку можуть виникнути через відсутність зв'язку між моментами доступу до передавального пристрою та моментами видачі готових даних. Щоб вибрати блоки даних на початку та в кінці кожного з них, записується службова інформація. Цей принцип полегшує підключення до шини різноманітних пристроїв зі своїми характеристиками, і цей принцип має кращу захист від шуму. Ви можете збільшити довжину шини, не турбуючись про перекіс сигналів синхронізації. Одним з недоліків є те, що порівняно з синхронною передачею швидкість передачі даних є низькою через передачу великих обсягів службової інформації. Асинхронний принцип передачі даних повинен застосовуватися в системах, де обмін даними не є постійним і не вимагає високої швидкості передачі даних.

Аналіз способів вирішення конфліктів в пристрої обміну

Якщо велика кількість пристроїв у системі одночасно передає дані на пристрої, що замінює, можуть виникнути конфлікти. Існує декілька способів вирішити ці конфлікти, включаючи присвоєння унікального пріоритету кожному пристрою, призначення фіксованих часових інтервалів кожному пристрою та використання багатопортової пам'яті.

Вирішення конфліктів у багатопортовому сховищі

Memory Inport містить незалежні набори адрес, даних та шин управління, які забезпечують одночасний та незалежний доступ до пам'яті пристрою. Ця функція може значно спростити створення складних систем, але можуть

виникати конфлікти, і для їх вирішення потрібно використовувати методи. Конфлікти виникають, коли кілька активних пристроїв отримують доступ до однієї комірки пам'яті під час запису даних з декількох портів або при записі через один порт та читанні через інші. Існують різні способи вирішення конфліктів у синхронній та асинхронній багатопортовій пам'яті (БП). Синхронний блок живлення забезпечує взаємну синхронізацію активних пристроїв з одним системним таймером, тому використання додаткової логіки для вирішення конфліктів не передбачено. В асинхронному блоці живлення для вирішення конфліктів використовуються: арбітражна логіка, семафори, запити на переривання та система Master / Slave.

1.4 Постановка завдання кваліфікаційної роботи

Залежно від схем розподілу пам’яті існують різні способи створення пристроїв спільного використання: спільна пам’ять та спільна пам’ять. Існує дві основні моделі обміну: заснована на передачі повідомлень (для розподіленого зберігання) та використанні спільного сховища.

Розподіляючи спільний простір між усіма вхідними портами, ви можете ефективно використовувати його відповідно до типу вхідного потоку даних. У цьому випадку всі важливі обчислювальні вузли об'єднуються з пам'яттю через загальну шину. Перевага цього підходу полягає в тому, що обмін відбувається шляхом запису / зчитування інформації з комірок спільної пам'яті, доступних для всіх вузлів, і тому не вимагає часу для передачі даних. До недоліків можна віднести можливість конфліктів при одночасному доступі до комірки пам'яті, проблему повільного доступу до оперативної пам'яті та її обмеженої ємності, а також проблему масштабованості. Чим більше використовуваних вузлів, тим вищі витрати на розробку і менша ефективність.

Існують системи з однорідним і неоднорідним доступом до пам'яті. У системах з однорідним доступом всі вузли можуть одночасно отримувати доступ до сховища. Цей спосіб організації обміну найчастіше використовується для створення обмінних пристроїв. У системах з нерівним доступом до пам'яті

частина спільної пам'яті виділяється кожному вузлу. Ця пам’ять має єдиний адресний простір, тому ви можете використовувати її адресу для прямого доступу до будь-якої комірки у спільній пам’яті. Час доступу до модулів спільної пам'яті з різних вузлів різний.

У цьому випадку кожен точка має доступ до певної фіксованої кількості виділених комірок пам'яті. Для забезпечення обміну інформацією вузли з'єднуються за допомогою каналів зв'язку. Пакет, призначений певному вихідному вузлу, втрачається, якщо блок пам'яті, призначений цьому вихідному вузлу, переповнюється, хоча інші блоки можуть бути порожніми на той момент. Обмін в системі здійснюється шляхом надсилання посилань та отримання повідомлень. Ця схема розподілу пам'яті використовується для завдань, які вимагають невеликого обміну даними та великої кількості пам'яті. У цьому випадку кожен точка має доступ до певної фіксованої кількості виділених комірок пам'яті. Для забезпечення обміну інформацією вузли з'єднуються за допомогою каналів зв'язку. Пакет, призначений певному вихідному вузлу, втрачається, якщо блок пам'яті, призначений цьому вихідному вузлу, переповнюється, хоча інші блоки можуть бути порожніми на той момент. Обмін в системі здійснюється шляхом надсилання посилань та отримання повідомлень. Ця схема розподілу пам'яті використовується для завдань, які вимагають невеликого обміну даними та великої кількості пам'яті.

До переваг можна віднести масштабованість, що означає, що ви можете поєднувати велику кількість вузлів без істотного зниження ефективності їх взаємодії. Вартість системи пропорційна кількості вузлів. До недоліків можна віднести проблему обміну даними та велике споживання енергії. Обмін даними в таких системах відбувається дуже повільно порівняно зі швидкістю обчислень (і з великими затримками). Тому неможливо ефективно вирішити проблеми, що вимагають інтенсивного обміну на таких системах.

Постановка завдання.

Для апроксимації паралельного сортування масивів чисел активації використано методи кусково-лінійної апроксимації (модуль FA_Sigm_N1) та апроксимацію поліномом другого порядку (модулі FA_Sigm_N2, FA_Sigm_N3). В першому методі для перемноження використовується зсуви. В другому методі

використовується три перемножувачі, а в третьому тільки один перемножувач та зсуви. Виконано порівняння по точності реалізацій трьох сигмоїдальних функцій. Цифрова апаратна реалізація ПАРАЛЕЛЬНОГО СОРТУВАННЯ МАСИВІВ ЧИСЕЛ активації виконувалася мовою С++ в середовищі розробки Quartus II. Модулі сигмоїдальних функцій реалізовані на CUDA EP3C16F484C6 сімейства Cyclone III. Виконано моделювання їх роботи в часовій області та порівняння по необхідних апаратних ресурсах і швидкодії. Реалізовані модулі сигмоїдальних функцій будуть використані при проектуванні багатошарових нейронних мереж прямого поширення.

2 АЛГОРИТМИ І ПРОЦЕСОРНІ ЕЛЕМЕНТИ СИГМОЇДАЛЬНИХ ФУНКЦІЙ АКТИВАЦІЇ

2.1 Формування вимог для апаратної реалізації сигмоїдальних функцій активації

Алгоритми визначення сигмоїдальних функцій активації для реалізацій VLSI повинні забезпечувати детермінований рух даних, бути добре структурованими та орієнтованими на реалізацію на наборі взаємопов'язаних процесорних елементів (РЕ). Структура та операції, що виконуються ПЕ, залежать від вимог до реалізації алгоритму. Переважно, щоб основні операції алгоритмів для визначення максимальних і мінімальних чисел реалізовувались за допомогою РЕ. Багато взаємопов’язаних факторів необхідно враховувати одночасно при розробці або виборі алгоритмів для визначення максимальних і мінімальних чисел.

Алгоритми повинні бути рекурсивними та локально залежними. Усі ПЕ повинні виконувати приблизно однакові операції в рекурсивному алгоритмі. Кожен з РЕ буде повторювати виконання фіксованого набору операцій над послідовністю вхідних даних. Ефективність відображення алгоритму в РЕ безпосередньо пов’язана із методом декомпозиції розв’язку задачі та перетворенням на незалежні основні операції, що виконуються паралельно, або на залежні, що виконуються в конвеєрному режимі.

В декартовій системі координат можна сформувати точкові системи (решітки) із набору РЕ, які є моделлю паралельних апаратних структур [23]. Ця модель дозволяє оцінити часову та апаратну складність алгоритму. У решітковій моделі кожен РЕ відповідає тимчасовому та просторовому j індексам, які вказують, коли і де виконується кожна з операцій алгоритму. В алгоритмах з локальною передачею даних різниця між просторовими індексами j на кроці рекурсії обмежена деякою константою, оскільки в таких алгоритмах обмін здійснюється лише між сусідніми РЕ. Клас алгоритмів із глобальними зв’язками включає алгоритми, які мають рекурсивні просторові індекси.

Забезпечення високої швидкості є однією з основних вимог до пристроїв

для визначення максимального та мінімального числа з масиву чисел. Проблема виникає при використанні пристроїв для розв'язання проблем у режимі реального часу, що накладає певні обмеження на процес визначення максимального та мінімального чисел з масиву чисел. Ці обмеження в першу чергу пов'язані з часом вирішення завдання, який не повинен перевищувати час обміну повідомленнями. $T_{обм}$, тобто:

$$T_p \le T_{обм} \tag{2.1}$$

Час обміну залежить як від обсягу масиву N;

розрядності n і частоти надходження вхідних даних F_d, так і від кількості k каналів та їх розрядності n_k. Такий час визначається за формулою:

$$T_{обм} = \frac{Nn}{F_d k n_k} \tag{2.2}$$

Одним з основних інтегрованих параметрів для оцінки пристроїв VLSI для визначення максимальних та мінімальних чисел з масиву чисел є ефективність обладнання, яка враховує кількість виводів інтерфейсу, однорідність структури, кількість та локальність з'єднань, продуктивність зв'язків до витрат на обладнання та оцінює елементи пристрою. за виконанням [23]. Кількісне значення ККД обладнання визначається наступним чином:

$$E = \frac{R}{t_o(k_1 \sum_{i=1}^{s} W_{ПЕ_i} d_i + k_2 Q + k_3 Y)} \tag{2.3}$$

де R - складність алгоритму визначення максимального та мінімального чисел;

t_o - час визначення максимального та мінімального чисел;

$W_{ПЕ_i}$ - витрати обладнання на реалізацію i-го процесорного елемента;

d_i - кількість функціональних вузлів i-го типу;

k_1 - коефіцієнт врахування однорідності $k_1 = f(s)$;

s - кількість видів функціональних вузлів;

Q - кількість зв'язків;

k_2 - коефіцієнт врахування регулярності зв'язків $k_2 = f(\Delta j)$;

Δj - просторова зв'язкова;

Y - кількість контактів інтерфейсу;

k_3 - коефіцієнт врахування кількості виводів інтерфейсу зв'язку $k_3 = f(Y)$.

При апаратній реалізації алгоритмів визначення максимального та мінімального чисел із масиву чисел висока ефективність використання обладнання досягається узгодженням інтенсивності надходження даних $P_d = knF_d$:

де k - кількість каналів надходження даних;

n - розрядність каналів надходження даних;

F_d - частота надходження даних, із інтенсивністю обчислень (обчислювальною здатністю) апаратних засобів, яку визнають так[37]:

$$D_k = \frac{mn_m}{T_k} \tag{2.4}$$

де m - кількість каналів надходження даних у сходинках конвеєра;

n_m - розрядність каналів надходження даних у сходинках конвеєра;

T_k - такт конвеєра.

– Завданням розробки пристроїв для визначення максимальних та мінімальних чисел з масиву чисел, орієнтованих на реалізацію VLSI, з високою ефективністю обладнання є мінімізація апаратних витрат, кількості інтерфейсних штифтів, підвищення однорідності структури та регулярності зв'язку в реальний час. Розробка таких пристроїв повинна базуватися на наступних принципах для максимального використання переваг сучасної технології VLSI та забезпечення цих вимог [23-38]:

- однорідності та регулярності структури;
- локалізації та спрощення зв'язків між елементами;
- модульності побудови;
- конвеєризації та просторового паралелізму опрацювання даних;
- узгодженості інтенсивності надходження даних із інтенсивністю обчислень в пристрої;
- програмування архітектури пристрою шляхом використання програмованих логічних інтегральних схем.

2.2 Вибір принципів та елементної бази для апаратної реалізації визначення сигмоїдальних функцій активації

Пропонується розробити орієнтованих на нейронні технології компютерні системина основі інтегрованого підходу, що включає:

Сучасні комп'ютерні засоби для елементної бази, апаратного та програмного забезпечення;

Методи та алгоритми нейронних мереж;

Methods Методи визначення, алгоритми та структури VLSI для реалізації основних операцій нейро-алгоритмів.

Підставою для деталізації моделі нейронного елемента може бути, зокрема, встановлення нових фактів у галузі нейрофізіології [5]:

1. наявність безлічі ділянок синаптичного контакту;
2. дихотомічне розгалуження дендритів різного порядку, що відповідає логічним операціям "Я", "АБО", "Ексклюзивно АБО", присвоєнню максимального або мінімального сигналу в технічних аналогах;
3. Різні діаметри дендритів стовбура, гілок, що прилягають до тіла нейрона, і розмір діаметра визначають ступінь важливості інформації, що протікає через дендрити.

4. наявність "доріжок" на поверхні сома, що йдуть від основних стовбурових дендритів до аксона, викликаючи існування паралельних шляхів обробки інформації та дозволяючи використовувати логічні операції над сигналами, що надходять від різних дендритів стовбура;

5. Особливості горбка аксона; Сам аксональний горбок визначає передавальну функцію нейрона, яка набагато складніша за формою, ніж сигмоїдна або лінійна передавальні функції, що використовуються в нейромережевих технологіях.

6. наявність роздвоєного розгалуження аксона; у гілках гілки здійснюється управління сигналом, яке залежить від співвідношення діаметрів різних гілок аксона; У математичному моделюванні ці функції можуть бути реалізовані за допомогою логічних операцій.

7. Наявність аксосоматичного зворотного зв'язку, що вже було реалізовано при побудові повторюваних нейронних мереж.

Поглиблене знання будови біологічного нейрона як ефективного інструменту трансформації можна розглядати як джерело фундаментальних ідей та концепцій для створення нових парадигм нейронних мереж не лише зараз, але й у довгостроковій перспективі.

Необхідно базувати побудову орієнтованих на нейронні технології компютерні системи на принципах, що зменшують витрати та час та розширюють сфери їх застосування. Аналіз показує, що цим вимогам можна задовольнити, використовуючи такі принципи проектування:

Змінний склад обладнання, який передбачає наявність ядра комп'ютера та змінних спеціалізованих апаратних та програмних модулів, за допомогою яких ядро адаптується до вимог конкретного додатку;

Модульність, що включає розробку компонентів орієнтованих на нейронні технології компютерні системи у вигляді модулів із доступом до стандартного інтерфейсу;

Конвеєрний та просторовий паралелізм обробки даних у спеціальних апаратних та програмних модулях;

Відкритість програмного забезпечення, що дозволяє створювати та вдосконалювати, а також максимізує використання стандартних драйверів та програмного забезпечення;

Узгодженість та адаптація спеціальних апаратних та програмних модулів до інтенсивності даних та структури нейро-алгоритмів;
Програмовність архітектури завдяки використанню нещодавно запрограмованих інтегральних логічних схем.

2.3 Алгоритм та структура процесорного елементу для сигмоїдальних функцій активації

Комп'ютерна система може бути представлена у вигляді постійної частини B - універсального обчислювального ядра та змінної частини V - спеціалізованих модулів, що реалізують основні операції нейроалгоритмів. Структура нейроорієнтованої комп'ютерної системи показана на малюнку 3, де БПП - це багатопортова пам'ять.

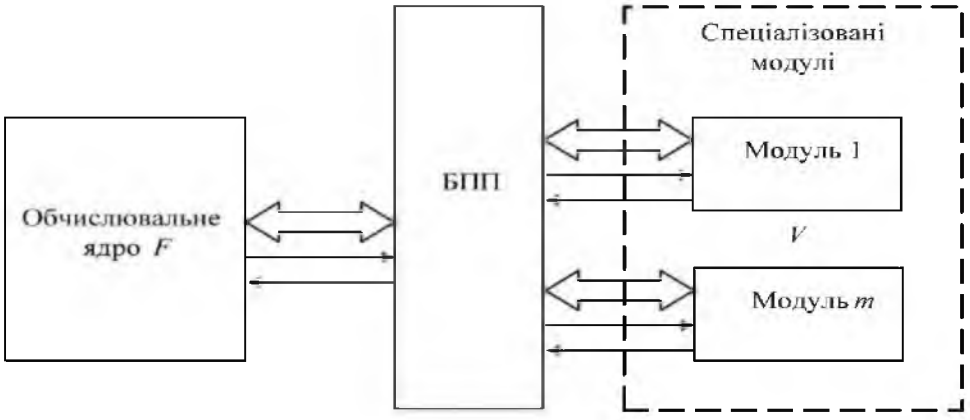

Рисунок 2.1 – Структура нейроорієнтованої комп'ютерної системи

Основними компонентами нейроорієнтованої комп'ютерної системи є ядро комп'ютера, ряд спеціалізованих модулів та ВРР. Паралельна обробка даних у орієнтованих на нейронний комп'ютерній системі вимагає організації обміну між ядром комп'ютера та низкою спеціалізованих модулів. Такий обмін у орієнтованих на нейронний комп'ютерній системі повинен здійснюватися за допомогою ВРР, що забезпечує паралельний доступ до великої кількості даних як з ядра комп'ютера, так і із спеціалізованих модулів. Нейропроцесор (нейрочіп)

є основою обчислювального ядра нейроорієнтованої комп'ютерної системи. Він містить ряд інструкцій, які добре підходять для виконання основних операцій з нейро-алгоритму, а також повний набір інструкцій загального призначення. Переважна більшість алгоритмів нейропарадигми обмежується виконанням обмеженої кількості основних операцій, таких як «додавання - множення».

Основними перевагами нейрочіпів є:

- відносно вища продуктивність (порівняно з центральним процесором);

- спрощена реалізація з'єднань "всі з усіма" (для розробника нейронних мереж);

- низьке споживання енергії;

- відносно низька ціна.

- Основні недоліки:

- висока структурна складність і низька надійність системи;

- велика складність ефективного здійснення процесу навчання, самонавчання та самоорганізації;

- значне збільшення енергоспоживання та втрата швидкості при одночасному збільшенні ступеня інтеграції нейрочіпів;

Жорстко визначена топологія.

Особливості архітектури та технічні властивості нейропроцесора визначають основні особливості обчислювального ядра нейроорієнтованої комп'ютерної системи. До таких властивостей належать: довжина інформаційного слова; кількість основних команд та час їх виконання; адресована ємність; внутрішні можливості зберігання даних та програм, а також кількість внутрішніх реєстрів.

Структура орієнтованих на нейронні технології компютерні системи залежить від конкретних вимог і множини нейроалгоритмів (N), які використовуються для розв'язання задач. При розв'язанні конкретної задачі здійснюється розподіл алгоритмів розв'язання задачі між обладнанням F і V

$$N = N_F + N_V \qquad (2.5)$$

де N_F - множина алгоритмів, які виконуються на обладнанні F;

N_V - множина алгоритмів, які виконуються на обладнанні V.

В залежності від співвідношення N_F і N_V комп'ютерні орієнтованих на нейронни системи діляться на такі типи:

з переважним використанням процесорного ядра (постійного обладнання F), коли

$$N_F \wedge N, N_V \wedge \wedge N_F >> N_V \tag{2.6}$$

з переважним використанням спеціалізованих модулів (змінної частини V), коли

$$N_F \wedge 0, N_V \wedge N, N_F << N_V \tag{2.7}$$

з рівномірним використанням постійного обладнання F і змінної частини V, коли $N_F \sim \mathrm{N_V}$.

Перший тип орієнтованих на нейронні технології компютерні системи характеризується тим, що більша частина обчислювальної потужності зосереджена в ядрі процесора.

У другому типі нейроорієнтованої комп'ютерної системи основні алгоритми обчислень реалізовані за допомогою спеціалізованих модулів, а ядро процесора використовується для виконання допоміжних службових функцій.

Третій тип орієнтованих на нейронні технології компютерні системи характеризується тим, що ядро процесора забезпечує алгоритми управління, операцій вводу-виводу та сервісні функції, а також спеціальні модулі - реалізація автоматизованих нейро-алгоритмів, які вимагають великих обчислювальних зусиль.

У більшості випадків обробка даних нейронної мережі та реалізація алгоритмів аналізу даних вимагає нормалізації вхідних даних. Нормалізація - це метод попередньої обробки вхідних даних (навчальних, тестових та робочих зразків), при якому значення ознак, що складають вхідний вектор, зменшуються до певного діапазону, зазвичай цього діапазону [0, 1] або [-1 , 1]. Існує багато

способів нормалізації вхідних значень. Найпростішою, але найбільш ефективною в більшості випадків є лінійна нормалізація. Якщо вихідні дані потрібно зменшити до діапазону [0, 1], це робиться наступним чином:

$$x_i^* = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{2.8}$$

Для приведення початкових даних до діапазону [-1, 1] лінійна нормалізація здійснюється таким чиом:

$$x_i^* = \frac{x_i}{x_{\max}} \tag{2.9}$$

Коли вхідні дані щільно заповнюють заданий інтервал, найкраще використовувати лінійну нормалізацію, оскільки вона не вимагає складних обчислень.

Основними операціями нормалізації даних є пошук максимального та мінімального чисел з масиву чисел. Для того, щоб забезпечити обробку потоків даних у реальному часі при реалізації таких операцій, потрібна висока швидкість, яка може бути досягнута паралелізацією обчислювального процесу та апаратної реалізації. Тому актуальною проблемою є розробка апаратно-орієнтованих паралельних алгоритмів для визначення максимального та мінімального числа з масиву чисел.

Відомі методи визначення максимального та мінімального чисел з масиву чисел засновані на послідовному порівнянні значень кожного числа, починаючи з другого, з поточним значенням максимального числа, яке на першому кроці приймає значення першого числа. Якщо значення числа більше поточного значення максимального числа, воно стає поточним максимальним значенням. Отже, кожна ітерація визначення в поточному значенні максимального числа містить найбільше значення із переданої частини масиву, а після визначення поточне значення є максимальним числом у всьому масиві [37-38].

Операції з визначення максимального та мінімального чисел з масиву чисел мають ряд специфічних особливостей, що не дозволяють їм використовувати відомі обчислювальні методи та алгоритми. Відомий апарат в основному орієнтований на реалізацію алгоритмів з перевагою арифметичних операцій перед логічними і не враховує особливостей визначення максимального та мінімального числа з масиву. Особливістю сучасного обладнання є неефективність використання багатобітових операційних пристроїв, що значно знижує продуктивність та ефективність використання пристроїв [39-42].

З аналізу літературних джерел можна зробити висновок, що недоліком відомих методів та алгоритмів є те, що вони не зосереджуються на апаратній реалізації, а використання існуючого обладнання для визначення максимального та мінімального числа з масиву чисел є неефективним.

Тому метою роботи є розробка алгоритмів та спеціальних структур VLSI для визначення сортування злиття в реальному часі з високою ефективністю використання пристрою.

Алгоритми з паралелізацієюта НВІС-структури пристроїв реалізація сигмоїдальних функцій активації. Аналіз методів і алгоритмів визначення сортування методом злиття у реальному часі із масиву чисел показав, що для НВІС-реалізацій найефективнішими є алгоритмами, які грунтуються на методі порозрядного порівняння [42]. Визначення максимального $A_{\max}$ і мінімального $A_{\min}$ чисел із групи чисел $A_1, A_2, ..., A_j, ...A_m,$ за таким методом виконується послідовним порівнянням розрядів всіх чисел починаючи зі старшого. При кожному порівнянні отримуємо i-і розряди максимального і мінімального чисел, визначення яких здійснюється за формулами:

$$\overline{A_{i\max}} = \bigwedge_{j=1}^{m} \overline{a_{ji} \wedge y_{ij}}, y_{1j} = 1, \tag{2.10}$$

$$A_{i\min} = \bigwedge_{j=1}^{m} \overline{\overline{a_{ji}} \wedge z_{ij}}, z_{1j} = 1, \tag{2.11}$$

де y_{ij}, z_{ij} - i-і розряди j-х слів управління;

a_{ji} - i-розряд j-о числа; m-кількість чисел у групі.

Формування $(i+1)-x$ розрядів j-х слів управління виконується за формулами (2.12) і (2.13) та виконання наступних логічних обчислень :

$$y_{(i+1)j} = (\overline{A}_{i\max} \vee x_{ji}) \wedge y_{ij}, \tag{2.12}$$

$$z_{(i+1)j} = (A_{i\min} \vee x_{ji}) \wedge z_{ij}, \tag{2.13}$$

Процес синтезу паралельних структур VLSI для визначення максимального та мінімального чисел з групи чисел зводиться до наступних етапів:

- призначення базової операції;
- просторово-часове відображення алгоритму;
- Розробка схеми процесорного елемента (PE), що реалізує Basic Operation робота алгоритму;
- Синтез структур VLSI на основі PE;
- Організація інтерфейсу VLSI.

Аналіз алгоритмів паралельного визначення сортування шляхом злиття групи чисел у реальному часі на основі побітового методу порівняння дозволив виявити основну операцію для реалізації VLSI.

Така базова операція включає формування розрядів слів управління за формулами $y_{(i+1)j} = (\overline{A}_{i\max} \vee x_{ji}) \wedge y_{ij}$ і $z_{(i+1)j} = (A_{i\min} \vee x_{ji}) \wedge z_{ij}$ та виконання наступних логічних обчислень:

$$\overline{A_{ji\max}} = \overline{a_{ji} \wedge y_{ij}}, \tag{2.14}$$

$$\overline{A_{ji\min}} = \overline{\overline{a}_{ji} \wedge z_{ij}}, \tag{2.15}$$

Виділена базова операція реалізується у вигляді ПЕ, схеми яких наведені на рисунку 2.2, де:

а - ПЕ однотактного пристрою;

б - ПЕ конвеєрного пристрою;

в - ПЕ пристрою з вертикальним опрацюванням вхідних чисел.

Особливістю розроблених ПЕ є використання спільних шин результатів, підключення до яких здійснюється за допомогою логічних елементів *2І-НЕ* з відкритим колектором. Вертикальне підключення до спільних шин результатів забезпечує збільшення розміру масиву чисел, з якого визначаються максимальне і мінімальне значення. Використання спільних шин результатів забезпечує високу швидкодії, яка не залежить від кількості чисел у масиві, а залежить тільки від часу затримки на ПЕ.

Вартість НВІС-пристроїв реалізація сигмоїдальних функцій активації в основному залежить від площі кристала, яка визначається як витратами обладнання (кількість транзисторів), так і кількістю зовнішніх виводів, число яких обмежене рівнем технології та розміром кристалу. Орієнтація структур сортування на НВІС-реалізацію вимагає зменшення числа виводів інтерфейсу та кількості з'єднань між ПЕ. Забезпечити ці вимоги можна використанням паралельно-вертикального алгоритму реалізація сигмоїдальних функцій активації, при якому надходження чисел і видача результатів здійснюється розрядними зрізами.

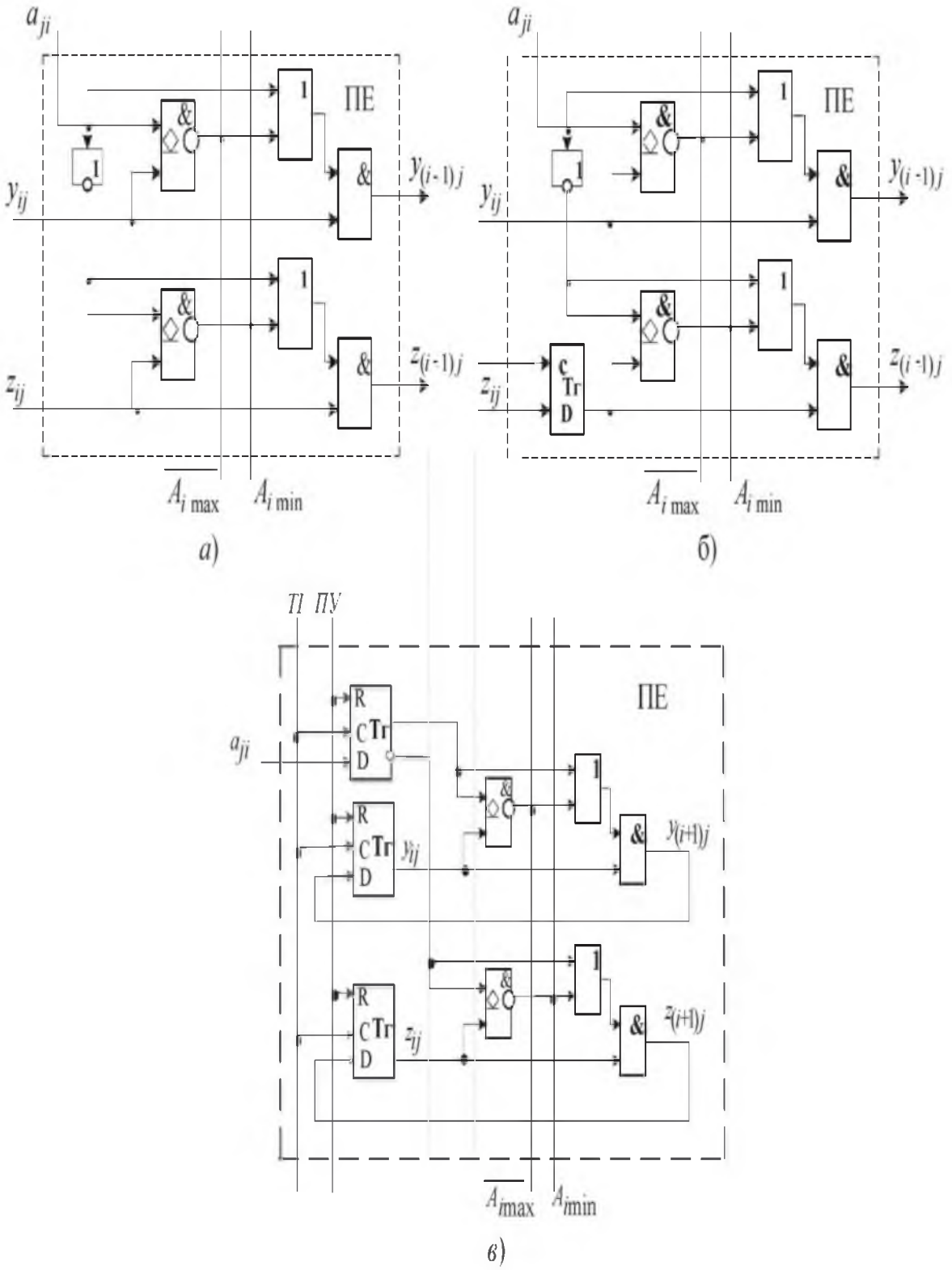

Рисунок 2.2 – де - *а* - ПЕ однотактного пристрою, б - ПЕ конвеєрного пристрою, *в* - ПЕ пристрою з вертикальним опрацюванням вхідних чисел

Структура паралельно-вертикального НВІС-пристрою реалізація сигмоїдальних функцій активації наведена на рисунку 2.3,

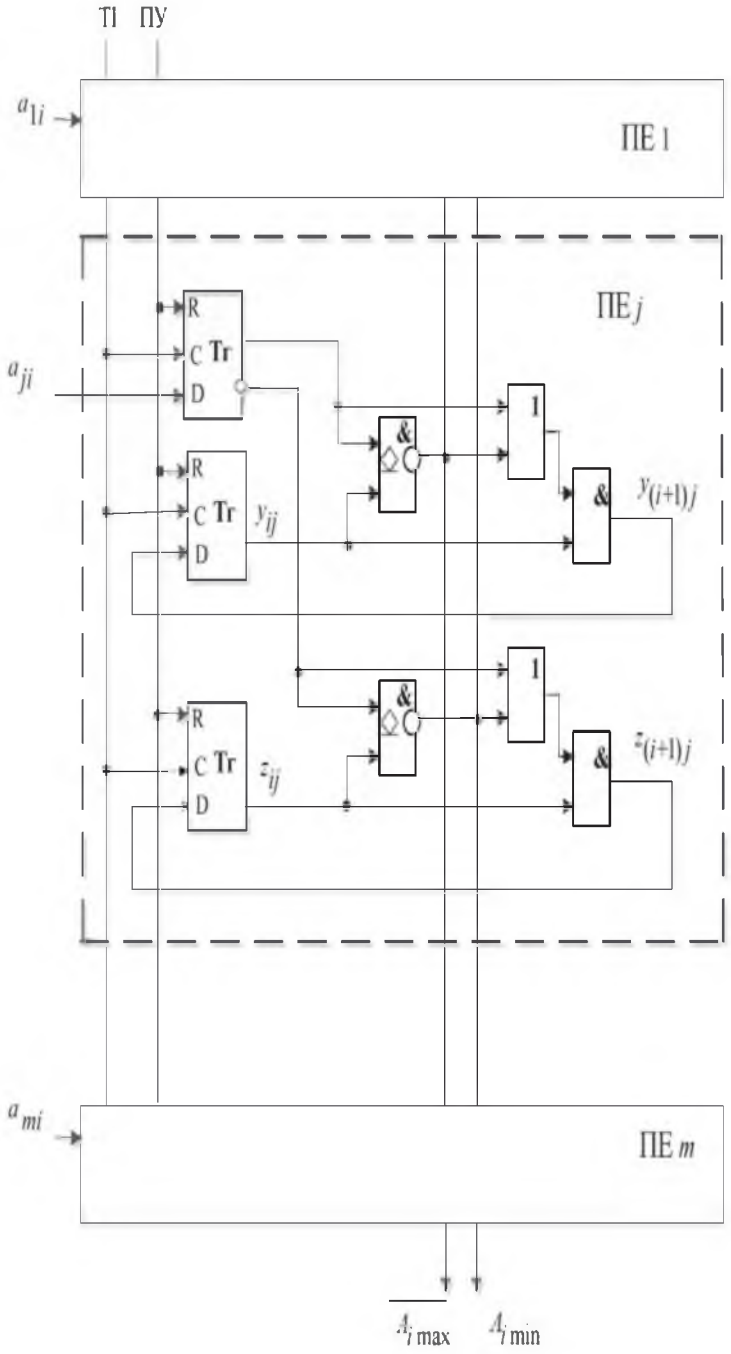

Рисунок 2.3 – Схема паралельно-вертикального НВІС-пристрою визначення максимальних і мінімаоьних значень

де TI - тактовий вхід;

$ПУ$ - вхід початкової установки тригерів;

$a_1,...,a_m$ - однорозрядні інформаційні входи, де m - кількість чисел, що порівнюються;

$ПЕ_1,...,ПЕ_m$ - $ПЕ$ пристрою з вертикальним опрацюванням вхідних чисел;

Тг1 і Тг2 - D-тригери;

$\overline{A_{\max}}$ і $A_{\min}$ - порозрядний вихід відповідно інверсного максимального та мінімального чисел.

Для збільшення ефективності визначення максимального числа з групи чисел доцільно застосувати паралельно-вертикальний підхід, при якому час визначення максимального числа з групи чисел не залежав від кількості чисел. При цьому у кожному такті роботи на інформаційні входи пристрою надходять розрядні зрізи всіх чисел починаючи зі старших розрядів.

Структуру такого пристрою наведено на Рисунок.2.4, де:

ТІ – тактовий вхід;

У – вхід початкової установки тригерів;

$x_1,\ldots,x_m$ – однорозрядні інформаційні входи, де m – кількість чисел, що порівнюються;

$БП_1,\ldots,БП_m$ – блоки порівняння;

T_1 і T_2 – D-тригери;

Вих – вихід результату.

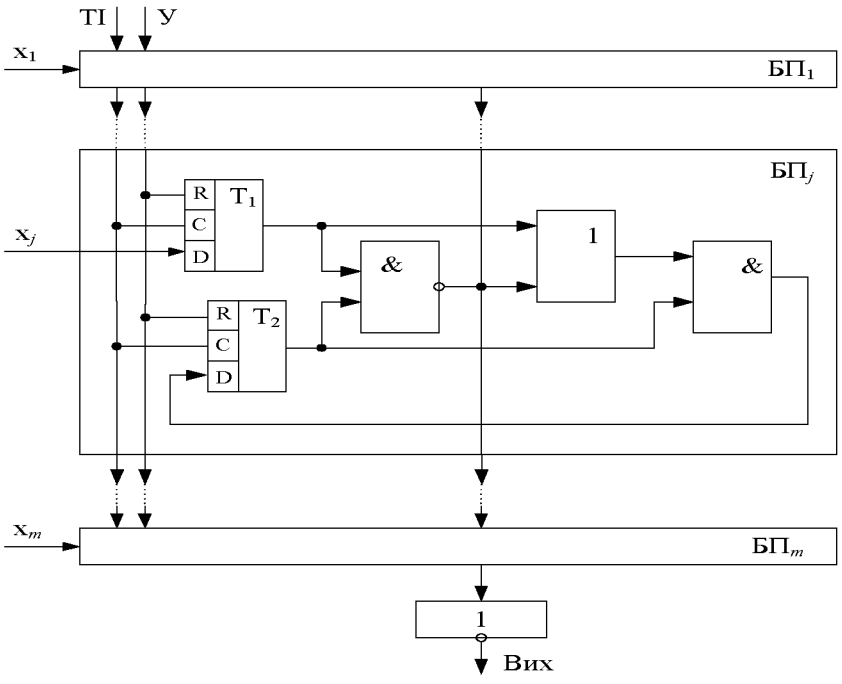

Рисунок. 2.4 – Структура пристрою визначення максимального числа з групи чисел

Пристрій для визначення максимального числа з групи чисел працює наступним чином.

Перед запуском імпульсу початкової структури, який надходить з входу початкової структури В, тригери Т1 і Т2 встановлюються в кожному блоці порівняння ВРj (j = 1, ..., m) в log.1. Інформація з виходів тригерів Т1 і Т2 (log.1) у кожному блоці порівняння ВРj встановлює виходи елементів і log log.1.

У першому циклі в кожному блоці порівняння ВРj отримує значення log.1 на вході інформації тригера Т2 від елемента І, а вхід інформації тригера Т1 з однобітового вводу інформації xj отримує значення найвищої цифри j-го числа. Перший тактовий імпульс у кожному блоці порівняння ВРj у тригері Т1 записується на найвищу цифру j-го числа, а log.1 пишеться для запуску Т2, що дозволяє j-му числу брати участь у визначенні максимального значення. У кожному блоці порівняння ВРj значення найвищої цифри j-го числа подається з виходу тригера Т1 на перший вхід елемента І-НЕ з відкритим колектором і на перший вхід елемента АБО. Log.1 з виходу тригера Т2 (керуючий сигнал) подається на другий вхід елемента AND-NOT з відкритим колектором та другий вхід елемента І. Інформація з виходів елементів І-НЕ з блоками порівняння відкритого колектора ВР1,..., ВРm поєднується складанням і надходить на другий вхід елементів АБО до цих блоків порівняння, а вхід елемента НЕ. У тому випадку, коли найвищі цифри порівняних чисел дорівнюють нулю, на вході елемента НЕ формується log.1, а в інших випадках - log.0. Інформація про вихід елемента НЕ (більша цифра максимального числа) отримується на виході результату Приклад. Якщо log.1 на других входах елементів АБО одиниці порівняння ВР1,..., ВРm на їх виходах встановлюють сигнал log.1, а з log.0 - інформацію з виходів встановлюють тригер Т1. Інформація з виходів елементів АБО на виходах елементів І генерує сигнали управління, які надходять на інформаційні входи Т2.

Інформація про тактовий імпульс з однобітового вводу інформації xj (наступна цифра) записується на тригер Т1 кожного блоку порівняння ВРj, а тригер Т2 записується на значення вихідного елемента І, що дозволяє або забороняє участь інформації з (log.1) (журнал. 0) Однорозрядний ввід інформації xj для подальшого формування максимального числа.

Формування другого і наступних бітів результатів і сигналів управління відбувається так само, як і в першому циклі.

Час визначення максимального числа з групи чисел у цьому пристрої залежить від бітового розміру чисел, а не їх кількості. Для n стовпчиків, де n - розрядність чисел, ми отримуємо максимальне число з групи m чисел.

Час визначення максимального та мінімального чисел із масиву з m рівний:

$$t_m = (t_{Tг} + 3t_I)n, \tag{2.16}$$

де $t_{Tг}$ - час запису інформації у тригер;

t_I - час затримки інформації при проходженні через логічні елементи типу *АБО, І, І-НЕ.*

Затрати обладнання на реалізацію даного пристрою рівні:

$$W_m = (3W_{Tг} + 6W_I)m \tag{2.17}$$

де $W_{Tг}$ - затрати обладнання на реалізацію D-тригера;

W_I - затрати обладнання на реалізацію логічних елементів типу *АБО, І, І-НЕ.*

Для сортування масиву з m × N чисел пропонується використовувати метод сортування злиттям. На початку сортування вхідний масив Q чисел ділиться на m масивів із N чисел. Щоб об’єднати масив із N чисел у порядок, кожне число вказується як упорядкований масив одиничного розміру. Сортування такого масиву шляхом об’єднання базується на послідовному об’єднанні впорядкованих масивів. В результаті першого етапу злиття утворюються N / 2 впорядковані масиви довжиною два. Кількість кроків, які необхідно об'єднати, щоб отримати упорядковані масиви довжиною N, визначається наступним чином:

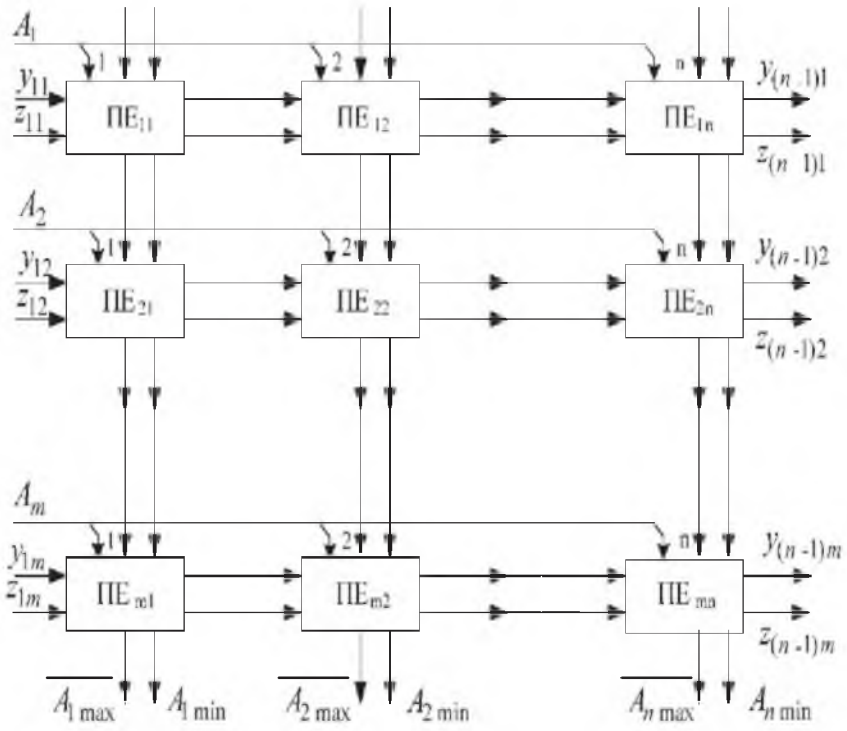

Рисунок 2.5 – Матрична НВІС-структура пристрою реалізація сигмоїдальних функцій активації

На основі однотактного і конвеєрного ПЕ синтезовано відповідно матричні однотактні (рис.2.3.4) та конвеєрні (рис.2.3.5) паралельні НВІС-структури пристрою визначення сортування методом злиття у реальному часі із групи чисел. Кожна із цих структур складається із матриці $(m \times n)$ ПЕ, в якій $ПЕ_{1i}, -ПЕ_{mi}$, - i-го стовпчика обчислюють відповідно до формул $\overline{A_{i\max}} = \bigwedge_{j=1}^{m} \overline{a_{ji} \wedge y_{ij}}, y_{1j} = 1,$ і $A_{i\min} = \bigwedge_{j=1}^{m} \overline{\overline{a_{ji}} \wedge z_{ij}}, z_{1j} = 1,$ значення i-х розрядів максимального $\overline{A_{\max}}$ і мінімального $A_{\min}$ чисел та формують у відповідності з формулами $y_{(i+1)j} = (\overline{A}_{i\max} \vee x_{ji}) \wedge y_{ij}$, і $y_{(i+1)j} = (\overline{A}_{i\max} \vee x_{ji}) \wedge y_{ij}, z_{(i+1)j} = (A_{i\min} \vee x_{ji}) \wedge z_{ij}$, значення $(i+1) - x$ слів управління $y(i+1)$ і $z(i+1)$.

Час визначення сортування методом злиття у реальному часі у однотактному пристрої визначається за формулою:

$$T_O = 4nt_i, \qquad (2.18)$$

де t_i - час спрацювання логічного елемента "I", n - розрядність чисел.

Апаратні витрати на реалізацію однотактного пристрою дорівнюють:

$$W_O = 7NnW_i, \tag{2.19}$$

де W_i - апаратні витрати на логічні елементи типу І, АБО, І-НЕ.

Особливістю конвеєрної НВІС-структури (рис.2.5) є введення у ПЕ тригерів та використання n блоків пам'яті типу FIFO. Кожний i-ий блок $FIFO_i$,- забезпечує затримку інформації на i тактів. Тривалість конвеєрного такту у такому пристрої визначається за наступною формулою:

$$T_K = t_{Tz} + 4nt_i, \tag{2.20}$$

де t_{Tz} - час спрацювання тригера.

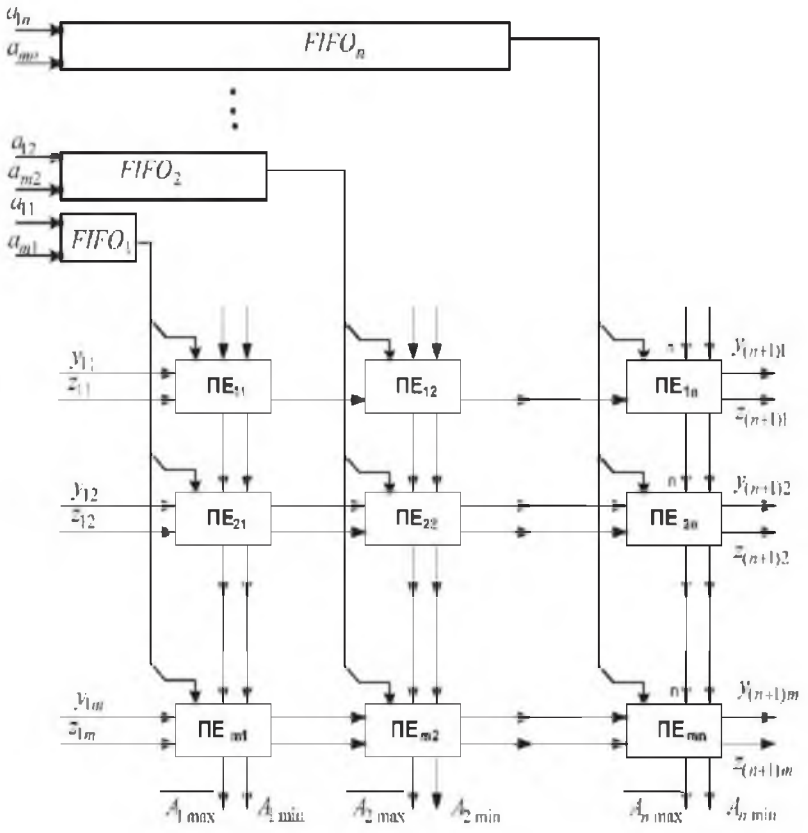

Рисунок 2.6 – Матрична конвеєрна НВІС- реалізація сигмоїдальних функцій активації

Конвеєр, стосовно процесорів, є способом, що використовується при розробці для збільшення інструкційної пропускної здатності (кількості інструкцій, які можуть бути виконані за визначений часовий проміжок). Ідея полягає в тому, щоб розділити обробку комп'ютерної інструкції на послідовність незалежних кроків, зі збереженням результатів у кінці кожного кроку. Це дозволяє управлінським ланцюгам комп'ютера отримувати інструкції зі швидкістю найповільнішого кроку обробки, але таке рішення набагато швидше, ніж виконання всіх цих кроків ексклюзивно для кожної інструкції.

При конвеєрній обробці також використовується суперскалярність – архітектура, що використовує девелику кількість декодерів команд, що можуть навантажувати роботою багато виконавчих блоків. Тобто, якщо в процесі роботи команди, що обробляються конвеєром, не суперечать одна одній і одна не залежить від результату іншої, то такий пристрій може розпаралелити виконання команд.

Апаратні затрати на реалізацію конвеєрного пристрою для визначення сортування методом злиття у реальному часі із групи m чисел дорівнюють:

$$W_K = W_{FIFO} + mn(W_{Tг} + 7w_i), \qquad (2.21)$$

де W_{FIFO} і $W_{Tг}$ - апаратні затрати відповідно на пам'ять типу FIFO і тригер.

3 АПАРАТНА РЕАЛІЗАЦІЯ ПАРАЛЕЛЬНОГО СОРТУВАННЯ МАСИВІВ ДАНИХ

3.1 Розроблення потокового графу паралельного сортування масивів даних методом злиття

Для сортування масиву з m × N чисел пропонується використовувати метод сортування злиттям. На початку сортування вхідний масив Q чисел ділиться на m масивів із N чисел. Щоб об'єднати масив із N чисел у порядок, кожне число вказується як упорядкований масив одиничного розміру. Сортування такого масиву шляхом об'єднання базується на послідовному об'єднанні впорядкованих масивів. В результаті першого етапу злиття утворюються N / 2 впорядковані масиви довжиною два. Кількість кроків, які необхідно об'єднати, щоб отримати упорядковані масиви довжиною N, визначається наступним чином:

$$p = \lceil \log_2 N \rceil \tag{3.1}$$

де ⌈ ⌉ - більше ціле число.

Сортування m упорядкованих масивів довжиною N виконується методом паралельного злиття, який заснований на основній операції злиття двох впорядкованих масивів в один упорядкований масив. На першому етапі сортування введений масив чисел ділиться на m / 2 пари впорядкованих масивів довжиною N, які об'єднуються парами. Ця комбінація реалізована на основі першого типу макрооперації (базова операція). В результаті першого етапу утворюються m / 4 упорядковані масиви довжиною 2N. На другому етапі використовується макрооперація другого типу, що поєднує два упорядковані масиви та один упорядкований масив. Ця макрооперація виконується на основі трьох макрооперацій першого типу.

Кількість кроків, необхідних для сортування масиву з m масивів чисел, визначається за формулою:

$$k = \lceil log_2 m \rceil \tag{3.2}.$$

Кожний s-й етап, де $s=1,\ldots,k$, реалізується на основі макрооперацій s-о типу, кожна з яких виконує об'єднання двох упорядкованих масивів $\{a_{1i}\}_{i=1}^{2^{s-1}N}$ та $\{a_{2i}\}_{i=1}^{2^{s-1}N}$ у один упорядкований масив $\{b_{1i}\}_{i=1}^{2^{s}N}$. Макрооперація s-о типу ґрунтується на трьох макрооперацiях (s-1)-о типу. Операційною основою всіх макрооперацій є базова операція об'єднання двох упорядкованих масивів $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$ у один упорядкований масив $\{b_{1i}\}_{i=1}^{2N}$. Кількість базових операцій необхідних для виконання макрооперації s-о типу дорівнює 3^{s-1}.

Алгоритми паралельного сортування масиву із $m\times N$ чисел методом злиття пропонується відображати у вигляді потокового графу $F=(\Phi,\Gamma)$, де $\Phi=\{\Phi_1,\Phi_2,\ldots,\Phi_n\}$ – множина функціональних операторів, які реалізують операцію сортування масивів довжиною N і базові операції об'єднання двох упорядкованих масивів $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$ у один упорядкований масив $\{b_{1i}\}_{i=1}^{2N}$, Γ-закон відображення зв'язків між операторами. Такі блок-схеми забезпечують пошук оптимальних просторово-часових рішень. Графічно блок-схема алгоритму паралельного сортування показана шляхом злиття масиву m × N чисел як кутових точок, які відповідають операторам алгоритму Phi (сортування масивів довжиною N та поєднання двох упорядкованих масивів довжиною N в один упорядкований масив) та дуг що представляють посилання. Зв'язки між операторами. Кожен оператор функції поєднує два впорядковані масиви довжиною N у впорядкований масив з двома входами та двома виходами. Кожен вхід нумерується послідовно з упорядкованими масивами N-розміру. На першому та другому виході оператора функціонального об'єднання послідовно формуються N максимальних та N мінімальних чисел об'єднаного впорядкованого масиву.

Процес розробки потокового графу паралельного сортування масиву із $m\times N$ чисел методом злиття можна розбити на такі чотири етапи:

1) декомпозиція алгоритму сортування масиву m × N чисел шляхом злиття;

2) дизайн зв'язку між операторами функцій;

3) консолідація функціональних операторів;

4) Плануйте сортувати масив із m × N чисел шляхом об'єднання.

На фазі розкладання алгоритм сортування масиву з m × N чисел паралельно поділяється методом злиття на функціональні оператори для сортування масивів довжиною N та об'єднання двох упорядкованих масивів довжиною N в один упорядкований масив. Функціональні оператори для сортування масивів довжиною N мають один вхід і один вихід, а функціональні оператори для об'єднання двох упорядкованих масивів в один мають два входи та два виходи. Для того, щоб розкласти алгоритм сортування масиву чисел шляхом злиття, доцільно використовувати метод функціональної декомпозиції, який розбиває алгоритм на функціональні оператори та показує взаємозв'язки між ними.

На фазі проектування комунікації необхідно розмістити функціональні оператори Phi та їх фіксацію на рівнях у просторі та часі. Структура з'єднань на блок-схемі між операторами функцій Phi суміжних рівнів визначається рівнем сортування. Потоковий граф алгоритму паралельного сортування масиву із $m \times N$ чисел методом злиття наведений на рис.1, де $a_{1i} - a_{mi} - m$ входи даних, $\Phi_{01} - \Phi_{0m}$ – функціональні оператори сортування масивів чисел довжиною N, $\Phi_{s1} - \Phi_{s(2m/2^s)}$ - макрооперації s-о типу, які реалізують злиття двох упорядкованих масивів довжиною $2^{s-1}N$ чисел у один довжиною 2^sN чисел.

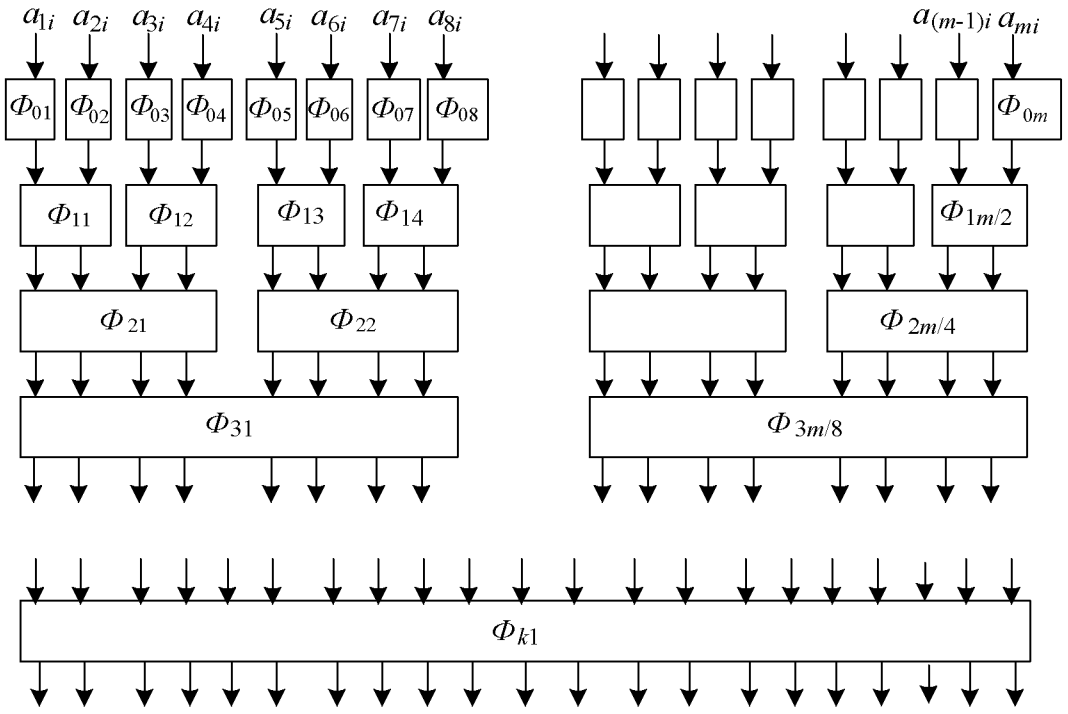

Рисунок 3.1. - Структура потокового графу алгоритму паралельного сортування масиву із $m \times N$ чисел методом злиття

Потоковий граф алгоритму паралельного сортування масиву із $m \times N$ чисел методом злиття складається із (k+1) ярусів. У нульовому ярусі виконуються оператори сортування масивів довжиною N чисел, а у s-х ярусах - макрооперація злиття двох упорядкованих масивів довжиною $2^{s-1}N$ чисел у один довжиною 2^sN чисел. У s-х ярусах виконує об'єднання двох упорядкованих масивів довжиною $2^{s-1}N$ чисел у один довжиною 2^sN чисел, яке реалізуються на основні макрооперації s-о типу. Кожна макрооперація s-о типу реалізується на трьох макрооперація (*s*-1)-о типу (рис.2).

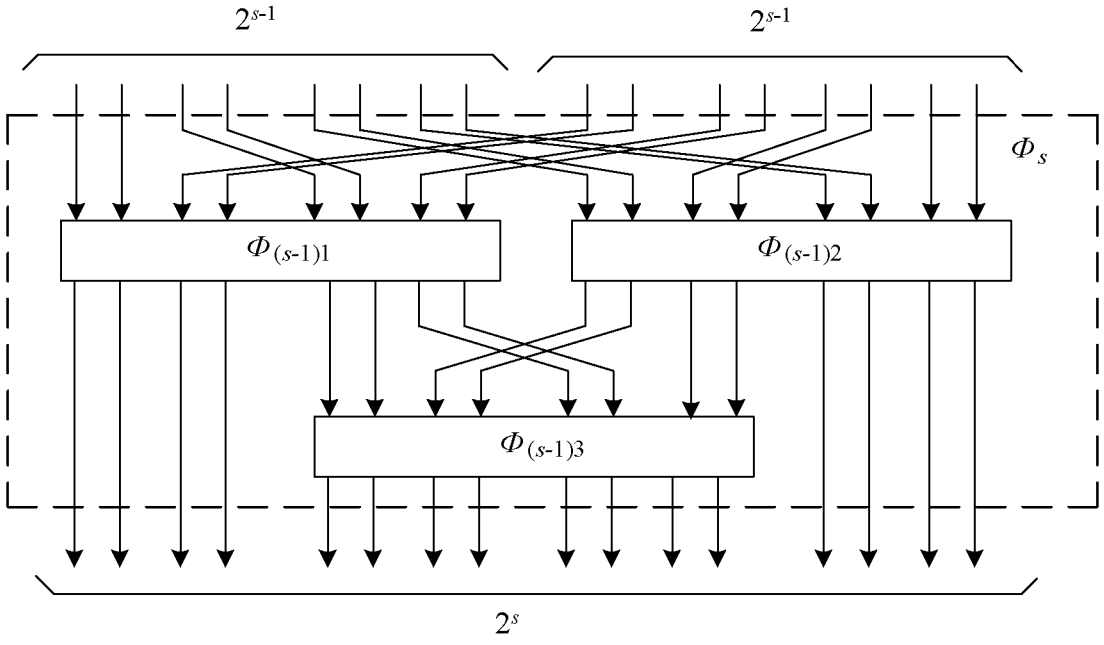

Рисунок 3.2 - Структура реалізації макрооперації s-о типу

Всі макрооперації s-о типу, які використовуються при сортуванні масиву методом злиття реалізуються на основі базової операції об'єднання двох упорядкованих масивів $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$ у один упорядкований масив $\{b_{1i}\}_{i=1}^{2N}$. Кількість базових операцій R, яка використовується для реалізації макрооперації s-о типу, визначається так:

$$R = 3^s.$$

Третій етап проектування полягає у консолідації роботи за допомогою комбінації функціональних операторів Fjk та каналів передачі даних між рівнями. Цей крок використовується для отримання конкретної блок-схеми, яка фокусується на реалізації за допомогою графічного процесора. Фаза консолідації тісно пов'язана з процесом планування процесу сортування.

Четвертий етап планування процесу сортування полягає у збереженні інформації про структуру блок-схеми алгоритму паралельного сортування масиву m × N чисел методом злиття. На цьому етапі здійснюється планування процесу сортування, визначаються значення затримок та перестановки даних. Для відтворення процесу сортування, управління, затримки та перестановки даних оператори вводяться у зазначену блок-схему. Конкретизована блок-схема, суміщена з графічним процесором, отримується шляхом лінійного проектування блок-схеми на горизонтальну вісь X. Приклад конкретної схеми алгоритму паралельного сортування масиву

із $m \times N$ чисел методом злиття наведений на рис.3, де $Ф_{cl}$ – функціональний оператор сортування масиву довжиною N чисел методом злиття (l=1,.., m), $Ф_{БОh}$ – функціональний оператор базової операції об'єднання (h=1,.., $m/2$) двох упорядкованих масивів довжиною N у один упорядкований масив $2N$, $Ф_{МЗП}$ – макрооператор затримки та перестановки, $Ф_{МУ}$ – макрооператор управління.

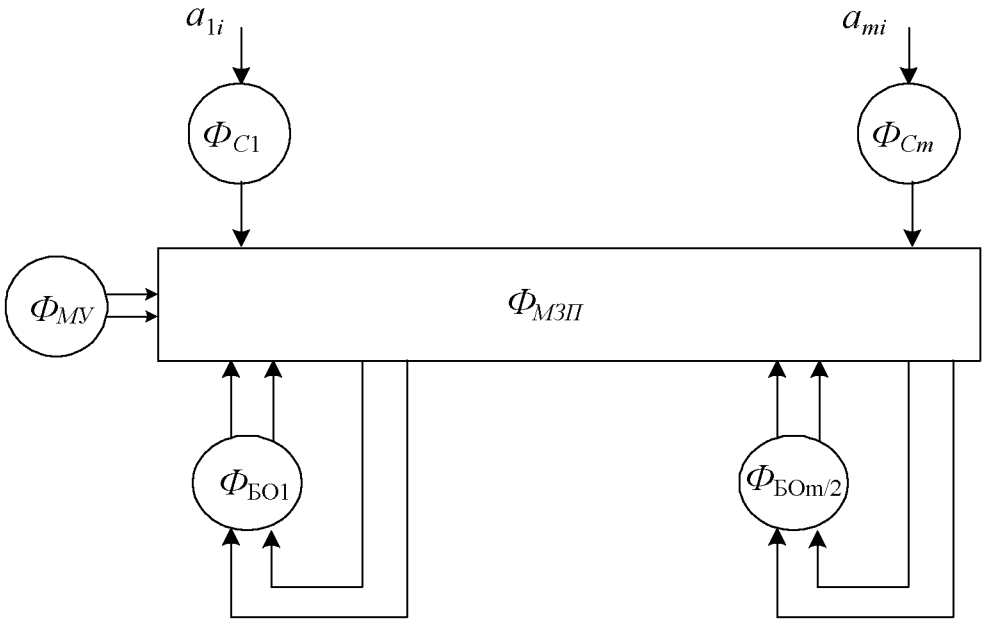

Рисунок 3.3 - Лінійна проекція потокового графу алгоритму сортування злиттям на горизонтальну вісь X

Отримана лінійна проекція потокового графу алгоритму сортування злиттям на горизонтальну вісь X орієнтована на архітектуру графічного

процесора GPU. За допомогою макрооператора затримки та перестановки $Ф_{МЗП}$ та макрооператора управління $Ф_{МУ}$, визначається величини затримок, перестановки даних і здійснюється планування процесу сортування.

3.2 Реалізація оператора послідовного сортування масиву з N чисел методом злиття

Сортування вхідного масиву з N чисел послідовно методом злиття включає поділ його на N впорядкованих масивів одиничної довжини та виконання операцій з послідовним об'єднанням цих масивів. Кількість етапів послідовного злиття впорядкованих масивів визначається за формулою (2). Операції поєднання впорядкованих масивів базуються на елементарних операціях - попарному порівнянні та перестановці елементів масиву.

Структура засобів для здійснення послідовного сортування шляхом злиття вхідного масиву з N числами показана на малюнку 3.4, на введенні даних Алі, P1k, P2k дані для встановлення першого та другого лічильників, TI1 - перші тактові імпульси, PU - початкове налаштування , ZpA1, ZpA2 - адреси сигналів запису в першому та другому лічильниках U1, U2 або U3, U4, - перемикання керуючих сигналів у першому та другому TI2, відповідно - другі тактові імпульси, Zp / Cht1, Zp / Cht2 - сигнали для вибору режиму роботи блоків пам'яті в кожному випадку першого і по-друге, VK1, VK2 - сигнал вибору кристалів першого та другого блоку пам'яті, U5, U6 - сигнали управління першого та другого драйвера шини, RP - результат порівняння, U7 - сигнали управління третього перемикача, U7, U8 - сигнали управління третього та четвертого драйвера шини, BU - блок управління, GA - генератор адрес, Rg - регістр, Sm - суматор, Lcha - лічильник адрес, Km - перемикач, BP - блок пам'яті, SHF - шинний формувач, bli - вихід бути відсортованими числами ..

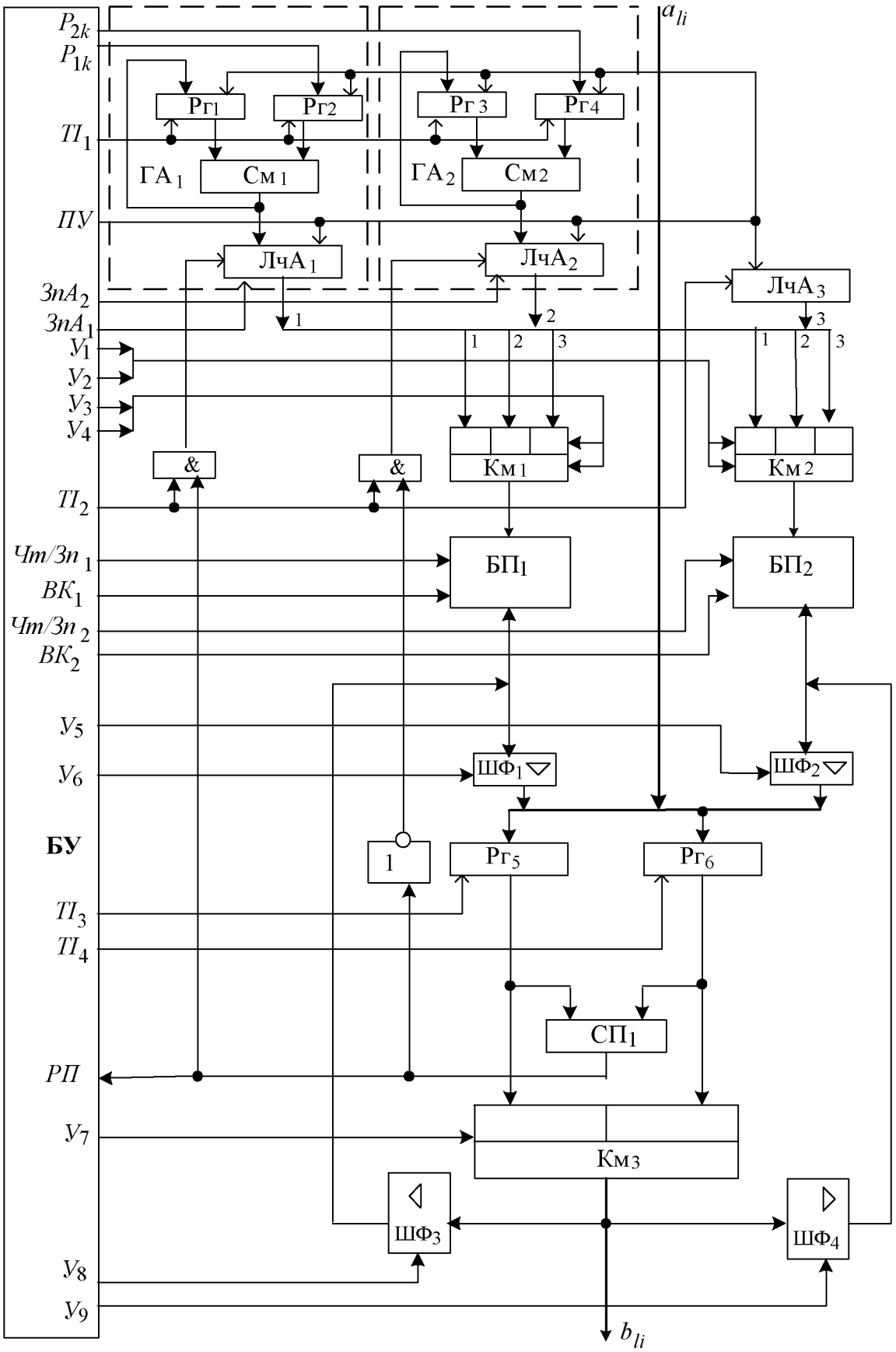

Рисунок 3.4 - Структура засобів послідовного сортування методом злиття вхідного масиву із N чисел

Перед сортуванням вхідний масив потрібно записати з N чисел у ВР1. Для цього комутатори Km1 і Km3 встановлюються в стані, в якому виходи Km1 отримують дані з виходів вимірювального пристрою LchA3, а виходи комутатора Km3 отримують дані з першого входу, драйвери шини SHF1 і SHF2 в третьому стані і SHF3 для передачі даних переходить у режим запису. Сигнал PU надходить від BU, який встановлює лічильники LchA1-LchA3 і реєструє Pg1-Pg4 на нуль. Номери масиву подаються один за одним на вхідні дані ali і записуються в блок ВР1 у кожному тактовому циклі. Після N стовпчиків у ВР1 зазначається масив із N чисел, які потрібно відсортувати.

Для сортування масиву із N чисел необхідно виконати p етапів об'єднання упорядкованих масивів. Кожний v-й етап, де $v=1,\ldots,p$, реалізується на основі макрооперації об'єднання v-о типу, яка зводиться до об'єднання двох упорядкованих масивів $\{a_{1i}\}_{i=1}^{2^{v-1}}$ та $\{a_{2i}\}_{i=1}^{2^{v-1}}$ у один упорядкований $\{b_{1i}\}_{i=1}^{2^{v}}$. Для v-о етапу об'єднання масивів необхідна кількість таких макроопераций визначається так:

$$z = \frac{N}{2^v}. \tag{3}$$

Розглянемо алгоритм покрокового виконання v-о етапу об'єднання масивів:

1 крок – встановлюємо регістри $Рг_1$-$Рг_4$, лічильники $ЛчА_1$-$ЛчА_3$ у нуль, комутатори $Км_1$ і $Км_2$ на передачу адреси відповідно з перших ($А_1$) і третіх ($А_3$) входів, шинні формувачі $ШФ_1$ і $ШФ_4$ на передачу даних, $ШФ_2$ і $ШФ_3$ у третій стан;

2 крок –записуємо число 2^v в регістри $Рг_2$ і $Рг_4$, а число 2^{v-1} в $Рг_3$ і лічильник $ЛчА_2$;

3 крок – зчитуємо з $БП_1$ найбільше число першого масиву та записуємо в $РГ_5$;

4 крок – переключаємо комутатор $Км_1$ на передачу адреси з других входів ($А_2$), зчитуємо з $БП_1$ найбільше число другого масиву та записуємо в $РГ_6$;

5 крок – порівнюємо числа з регістрів $РГ_5$ і $РГ_6$ і результати порівняння передаємо у БУ, який формує сигнал управління $Км_3$, що забезпечує передачу більшого числа та його запис у $БП_2$;

6 крок – збільшуємо на 1 значення лічильників $ЛчА_3$ та лічильника, за адресом якого було зчитано більше число;

7 крок – встановлюємо комутатор $Км_1$ на передачу адреси з виходів лічильника, за адресом якого було зчитано більше число та зчитуємо за даною адресом з $БП_1$ число, яке записується у регістр, з якого число було записано в $БП_2$;

8 і наступні кроки повторюють 5, 6 і 7 кроки до моменту коли один з лічильників $ЛчА_1$ або $ЛчА_2$ стане рівним відповідно $(2^{v-1}-1)$ або (2^v-1), тобто коли з $БП_1$ повністю зчитаний один із масивів;

Наступні кроки зводяться до читання з $БП_1$ залишку чисел іншого масиву та запис їх $БП_2$.

Дальше переходимо до об'єднання наступних двох масивів в один. Таке об'єднання передбачає запис в $ЛчА_1$ числа 2^v а в $ЛчА_2$ числа $(2^{v-1}+2^v)$ та повторення кроків починаючи 3-о до запису об'єднаного упорядкованого масиву в $БП_2$. Відсортований масив із N чисел отримуємо при виконанні p-о етапу об'єднання масивів на виході b_{li}.

3.3. Вдосконалення паралельного сортування методом злиттям

В основі алгоритмів паралельного сортування методом злиття лежить базова операція об'єднання двох упорядкованих масивів чисел $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$ у один упорядкований масив $\{a_i\}_{i=1}^{2N}$ чисел. Таке об'єднання виконується за $2N$ тактів. Метою вдосконалення паралельного сортування методом злиття є зменшення кількості тактів необхідних для отримання відсортованого масиву $\{a_i\}_{i=1}^{2N}$ чисел із двох відсортованих масивів чисел $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$. Для зменшення часу об'єднання двох упорядкованих масивів чисел $\{a_{1i}\}_{i=1}^{N}$ та $\{a_{2i}\}_{i=1}^{N}$ пропонується розпаралелити процес об'єднання масивів шляхом одночасно злиття двох масивів починаючи як з максимальних, так і мінімальних значень. Відсортований масив із $2N$ чисел отримаємо на двох виходах: на першому виході

(N_{1Cmax} найбільших чисел), на другому виході (N_{2Cmin} найменших чисел). Кількість тактів необхідних для такого об'єднання дорівнює N, що у два рази менше в порівнянні з існуючими алгоритмами об'єднання.

Базова операція об'єднання двох упорядкованих масивів чисел ґрунтуються на елементарних операціях – попарного порівняння та перестановки елементів масивів. При об'єднанні двох масивів чисел починаючи з максимальних використовуються такі операції:

$$y_{\max i} = \begin{cases} 0, \textit{коли } a_{1\max i} \le a_{2\max i} \\ 1, \textit{коли } a_{1\max i} > a_{2\max i} \end{cases}, \tag{3.4}$$

$$a_{C\max i} = \begin{cases} a_{1\max i}, \textit{коли } y_{\max i} = 1 \\ a_{2\max i}, \textit{коли } y_{\max i} = 0 \end{cases}, \tag{3.5}$$

а при об'єднанні масивів чисел починаючи з мінімальних такі:

$$y_{\min i} = \begin{cases} 0, \textit{коли } a_{1\min i} \le a_{2\min i} \\ 1, \textit{коли } a_{1\min i} > a_{2\min i} \end{cases}, \tag{3.6}$$

$$a_{C\min i} = \begin{cases} a_{1\min i}, \textit{коли } y_{\min i} = 0 \\ a_{2\min i}, \textit{коли } y_{\min i} = 1 \end{cases}, \tag{3.7}$$

де $y_{\max i}$, $y_{\min i}$ – результат попарного порівняння відповідно елементів максимальних і мінімальних масивів; $a_{1\max i}$, $a_{2\max i}$ – i-і елементи порівняння максимальних значень відповідно першого та другого масивів; $a_{1\min i}$, $a_{2\min i}$ – i-і елементи порівняння мінімальних значень відповідно першого та другого масивів; $a_{C\max i}$, $a_{C\min i}$ – i-і відсортовані елементи масивів відповідно максимальних чисел і мінімальних чисел.

Структура засобів реалізації паралельного об'єднанні двох масивів чисел наведена на рис.5, де $Вх_1$, $Вх_2$ – входи надходження відповідно першого і другого масивів чисел, Чт/Зп – вхід вибору режиму роботи пам'яті читання або запис, ТІ – вхід тактових імпульсів, ЛА – лічильник адреси, Км – комутатор, БП – блок пам'яті, СП – схема порівняння, Рг - регістр.

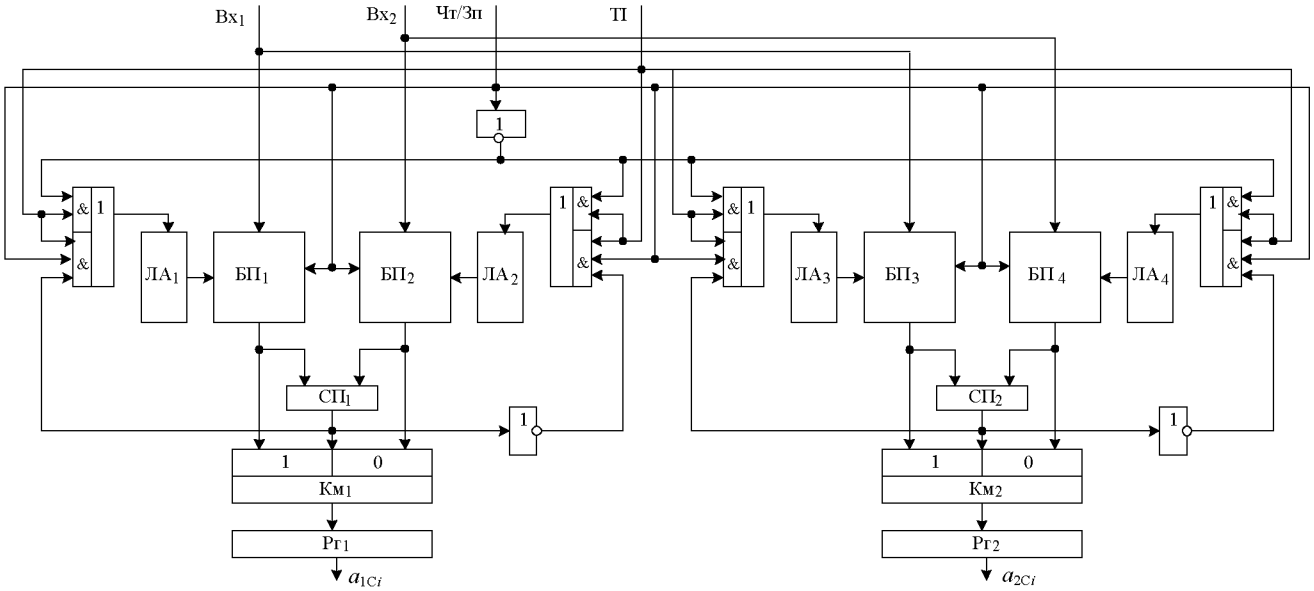

Рисунок 3.5 - Структура засобів реалізації паралельного об’єднанні двох масивів чисел

Алгоритм паралельного об'єднання двох числових полів здійснюється у два етапи. На першому етапі перший N1 і другий N2 відсортованих масивів записуються паралельно блокам пам'яті ВР1, ВР3 і ВР2, ВР4 відповідно. На другому етапі масиви, записані в ВР1 і ВР3, виконуються в порядку зменшення, а в ВР2 і ВР4 у порядку зростання. Кожен етап проводиться в N циклів. Тривалість кожної дії визначається за формулою:

$$t_{ТІ}=t_{БП}+t_{СП}+t_{Км}+t_{Рг}, \quad (5)$$

де $t_{БП}$, $t_{СП}$, $t_{Км}$ і $t_{Рг}$ – час відповідно звертання до пам’яті, порівняння двох чисел, затримки комутатора і запису в регістр.

Блок схема алгоритму паралельного об’єднанні двох масивів чисел наведена на рис.6, Лч$_ц$ - лічильник циклів, Вх$ТІ$ – вхід тактових імпульсів.

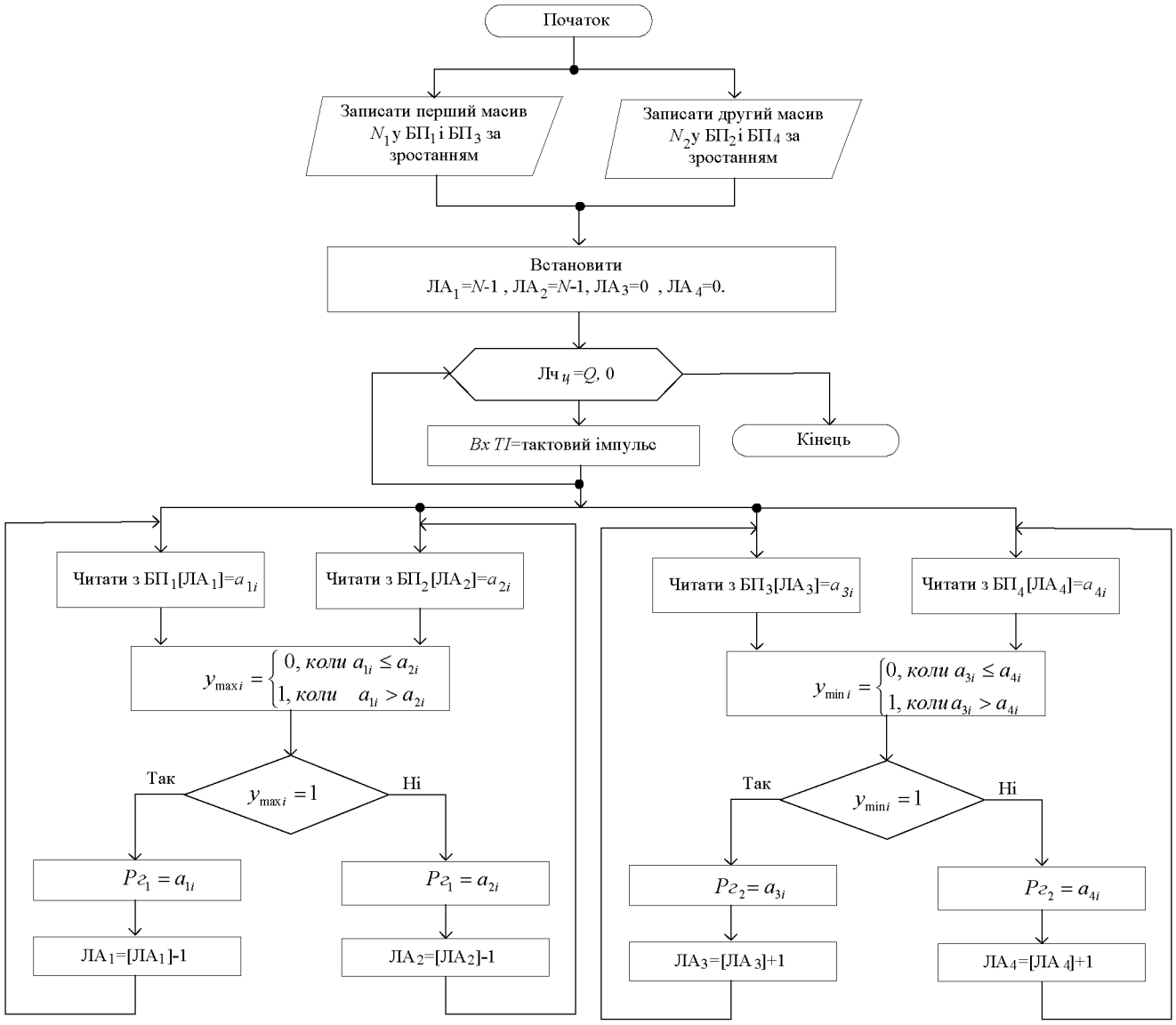

Рисунок 3.6 - Блок схема алгоритму паралельного об’єднанні двох масивів чисел

Для паралельного сортування m упорядкованих масивів чисел довжиною N чисел використаємо базову операцію паралельного об’єднанні двох масивів чисел (рис.2). Особливістю виконання такої базової операції є суміщення в часі процесу видачі відсортованих чисел з їх записом у наступні блоки пам’яті. Результатом паралельного сортування m упорядкованих масивів чисел довжиною N чисел є відсортований масив розміром $m \times N$ чисел, найбільші N числа якого послідовно отримуємо першому вході, а найменші N числа - на m виході.

3. 4 . Паралельне сортування масивів даних з використанням вдосконаленого методу злиття для графічного процесора.

Розробка програмного забезпечення для паралельного сортування наборів даних із використанням вдосконаленого методу злиття базується на використанні інтегрованого підходу, що включає: дослідження, вдосконалення та розробку методів та алгоритмів паралельного сортування наборів даних; Блок-схеми алгоритмів паралельного сортування; Архітектура графічного процесора та модель програмного забезпечення CUDA.

Багатоядерний графічний процесор nVidia GeForce RTX 2060, який містить 1920 ядер та підтримує архітектуру та модель програмного забезпечення CUDA, був використаний для реалізації паралельної сортування масивів даних за допомогою вдосконаленого методу злиття.

Оскільки CUDA - це програмна модель, орієнтована на багатоядерний архітектор, для її ефективного використання необхідно використовувати принципи та підходи паралельного програмування. Основною відмінністю між програмною реалізацією алгоритмів на CUDA (GPU) та програмуванням на центральному процесорі є використання потоків, блоків та сіток. Потік в графічному процесорі є основним елементом даних, що підлягають обробці. На відміну від процесорів, потоки CUDA надзвичайно "легкі", що, в свою чергу, забезпечує низьку вартість обміну між потоками. Блоки знову являють собою комбінацію від 64 до 512 потоків. Блоки графічного процесора складаються в сітки. Перевага цієї групування полягає в тому, що кількість блоків, які одночасно обробляються графічним процесором, тісно пов’язана з апаратними ресурсами, що в свою чергу призводить до високого масштабування програми. Оскільки ядро програми використовується для одного дзвінка для всіх мережевих потоків, не звертаючи уваги на наявні ресурси графічного процесора. Якщо у пристрою недостатньо ресурсів, блоки виконуються по одному. Якщо ресурсів достатньо, блоки обробляються паралельно. Це в свою чергу означає, що один і той же програмний код ефективно працює як на відносно простих, так і на більш складних графічних процесорах.

Базуючись на описаних принципах паралельного програмування на графічних процесорах, першим кроком реалізації програми паралельного сортування масивів даних з використанням вдосконаленого методу злиття є декомпозиція методу на рівні блоків та на рівні потоків.

Базовою операцією, що виконується на кожному потоці є об'єднання двох упорядкованих масивів чисел, яка складається з двох кроків – попарного порівняння та перестановки елементів масивів.

Загальна схема алгоритму методу сортування злиття з розподіленими між блоками і потоками базовими операція для масиву з 16-и чисел наведено на рис.7

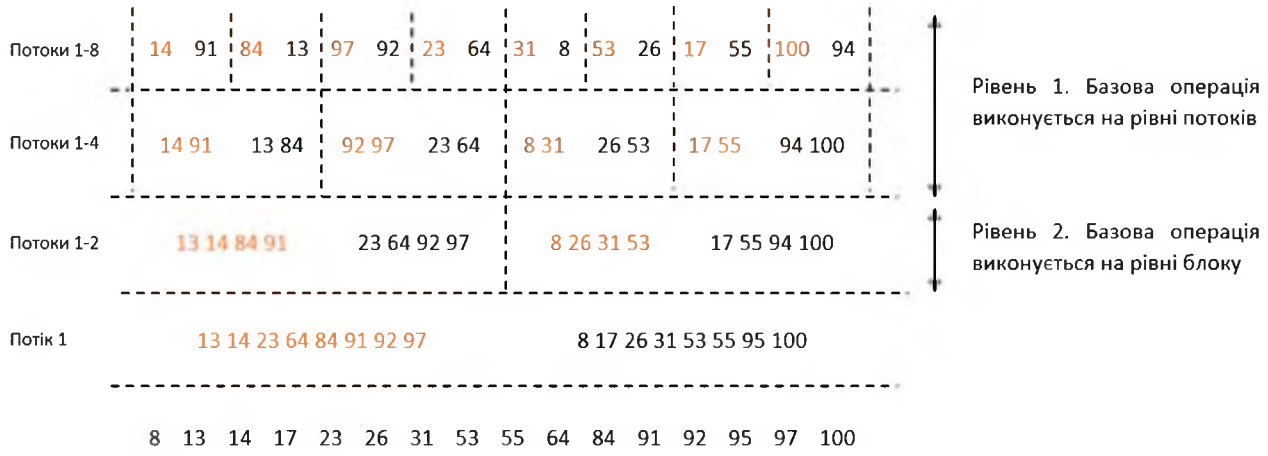

Рисунок 3.7 - Схема алгоритму методу сортування злиття на графічному процесорі

Окрім реалізації ядра програми для виконання базової операції сортування на блоках і потоках графічного процесора необхідно розроблення додаткових програм для внутрішніх операцій: створення тимчасових масивів для зберігання проміжних результатів, циклів виконання програм у потоках та блоках, бінарного пошуку, присвоєння ключів записав на фінальній стадії порівняння двох масивів і формування вихідного відсортованого масиву.

Графіки залежності часу виконання програми паралельного сортування даних вдосконаленим методом злиття для GPU та CPU у залежності від розмірів масивів наведені на рис.8.

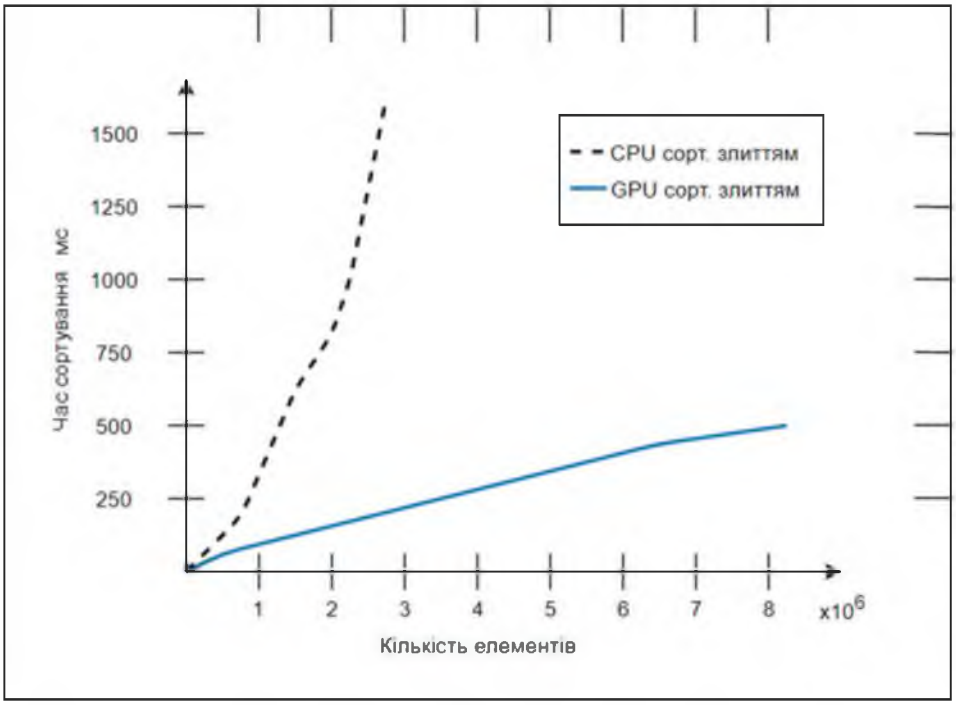

Рисунок 3.8 - Графіки часу виконання програми паралельного сортування даних вдосконаленим методом злиття для GPU та CPU у залежності від розмірів масивів

Незважаючи на велику кількість допоміжних операцій при реалізація паралельного сортування методом злиття на графічному процесорі nVidia GeForce RTX 2060 дає збільшення швидкодії приблизно в 100 разів в порівнянні з центральним процесором Inter i7.

ВИСНОВКИ

1. Удосконалено метод сортування двох масивів шляхом злиття. Поєднання двох масивів одночасно, починаючи з максимальних та мінімальних значень, перекошує процес сортування та зменшує вдвічі кількість циклів сортування порівняно з послідовним об’єднанням.

2. Визначено, що основними етапами розробки блок-схеми для сортування масивів даних паралельно з використанням вдосконаленої методології злиття, яка фокусується на архітектурі графічного процесора, є: декомпозиція алгоритму сортування масиву даних за допомогою вдосконаленого методу злиття; Дизайн зв'язку між операторами функцій; Консолідація операторів функцій; Плануйте сортувати записи, використовуючи вдосконалений метод злиття.

3. Буде розроблена паралельна блок-схема сортування масиву даних, яка ефективно просторово розпаралелює процес сортування масивів даних та зменшить час сортування за допомогою вдосконаленого методу сортування двох масивів шляхом їх об’єднання.

4. Розроблена для GPU програма GPU для паралельного сортування масивів даних з використанням вдосконаленого методу злиття, який за допомогою програмної моделі CUDA та заданої блок-схеми алгоритму скорочує час сортування приблизно в 100 разів порівняно з використанням лише центрального процесора

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гросуляк П.І, Цьома Б.І. Структура та моделі роботи системи управління мікрокліматом мінітеплиці III Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі». 26 листопада 2020, Тернопіль, Україна. Тернопіль: ЗУНУ, 2020. с.37

2. Гросуляк П.І, Цьома Б.І. Використання сучасних інформаційних технологій в сільському господарстві III Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі». 26 листопада 2020, Тернопіль, Україна. Тернопіль: ЗУНУ, 2020. с.38

3. Нейроподібні методи, алгоритми та структури обробки сигналів і зображень у реальному часі: монографія / Ю.М. Рашкевич, Р.О. Ткаченко, І.Г Цмоць, Д.Д. Пелешко. Львів: Видавництво Львівської політехніки, 2014. -256 с.

4. Проблемно-ориентированные высокопроизводительные вычислительные системы: В.Ф. Гузик, В.Е. Золотовский: Учебное пособие. Таганрог:Изд-во ТРТУ, 1998. 236 с.

5. Уоссермен Ф.Нейрокомпьютерная техника. – М.: Мир,1992. – 259с.

6. А.В. Палагин, В.Н. Опанасенко. Реконфигуриремые вычислительные системы. – К.: Просвіта, 2006.- 280с.

7. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика. – М.: Горячая Линия-Телеком, 2002. – 382 с.

8. Николаев А.Б., Фоминых И.Б. Нейросетевые методы анализа и обработки данных. Учебное пособие. - М.: МАДИ (ГТУ), 2003, 95с.

9. Цмоць І.Г. Інформаційні технології та спеціалізовані засоби обробки сигналів і зображень у реальному часі. Львів: УАД, 2005.- 227с.

10. Грибачев В. П. Элементная база аппаратных реализаций нейронных сетей // Компоненты и технологии. 2006. № 8

11. Круг П.Г. Нейронные сети и нейрокомпьютеры: Учебное пособие по курсу «Микропроцессоры». М.: Издательство МЭИ, 2002. 176 с.

12. Проблемы построения и обучения нейронных сетей / под ред. А.И.Галушкина и В.А.Шахнова. - М. Изд-во Машиностроение. Библиотечка журнала Информационные технологии №1. 1999. 105 с.

13. А.И.Галушкин Некоторые исторические аспекты развития элементной базы вычислительных систем с массовым параллелизмом (80- и 90-годы) // Нейрокомпьютер, №1. 2000. - С.68-82

14. С.И.Аряшев, С.Г.Бобков, Е.А.Сидоров Параллельный перепрограммируемый вычислитель для систем обработки информационных сигналов // "Нейроинформатика -99". - Москав, МИФИ. Часть 2. С.25-33.

15. Э.Ю. Кирсанов Цифровые нейрокомпьютеры: Архитектура и схемотехника / Под ред. А.И.Галушкина. - Казань: Казанский Гос. У-т. 1995. 131 с.

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

17. Борисов В.Л., Капитанов В.Д. Методика быстрого создания нейроускорителей // Нейрокопьютеры: разработка и приенение, №1, 2000 год.- С.12-24.

18. Роберт Хехт-Нильсен Нейрокомпьютинг: история, состояние, перспективы // Открытые системы. N4. 1998.

19. А.И. Власов Нейросетевая реализация микропроцессорных систем активной акусто- и виброзащиты// Нейрокомпьютеры:разработка и применение, №1, 2000. С.40-44.

20. Нейроприскорювачі на базі нейрочіпів - http://citforum.ru/hardware/neurocomp/neyrocomp_07.shtml

21. Елементна база нейрообчислювачів - http://opticstoday.com/katalog-statej/stati-na-ukrainskom/nejrokomputeri/elementna-baza-nejroobchislyuvachiv.html

22. Сучасні напрямки розвитку нейрокомп'ютерних технологій - http://www.victoria.lviv.ua/html/oio/html/theme9.htm

23. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы М.: Изд. Дом «Вильямс», 2001. 384с.

24. Браунси К. Основные концепции структур данных и реализация в С++. М.: Изд. Дом «Вильямс», 2002. 320с.

25. Проценко В.С. Техніка програмування мовою Сі: Навчальний посібник К.: Либідь, 1993. 224 с.

26. Шилдт Г. Теория и практика С++ СПб.: BHV Санкт-Питербург, 1996. 416 с.

27. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. М.: «Мир», 1979. - 536с.

28. Вирт Н. Алгоритмы + структуры данных = программы. М.: "Мир", 1985. 544 с.

29. Гудман С. Хидетниеми С. Введение в разработку и анализ алгоритмов. М.: "Мир", 1981. 366 с.

30. Кнут Д. Искусство программирования для ЭВМ. т.3. Сортировка и поиск. М.:Мир, 1976. 678 с.

31. Мейер Б., Бодуэн К. Методы программирования: В 2-х томах М.: Мир, 1982. 356+368с.

32. Керниган, Б.,Мова програмування Сі. Завдання по мові Сі/Б.Керниган, Д.Ритчи. М.:ФиС, 1985. 280 с.

33. Страуструп, Б. Мова програмування Сі ++ /Б.Страуструп. М.:Радіо та зв'язок, 1991.352 с.

34. Белецкий, Я. Енциклопедія мови Сі М.:Мир, 1992.687 с.

35. Белецкий, Я.ТурбоСі++: Нова розробка: навч. посібник для студентів вищих навчальних закладів / Я.Белецкий. - М.'.Машинобудівництво, 1994. -400с.

36. Пильщиков, В. Н. Збірник вправ по мові Паскаль: навч. Посібник для втузов /В. Н.Пильщиков. М.:Висш. шк.,1990. 223 с.

37. Кармен, Томас Х., Лейзерон, Чарльз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритми: построение и анализ, 2-е издание. :Пер. с англ. - М.: Издательский дом "Вильямс", 2005. 1296 с.

38. Кнут Д. Искусство программирования для ЗВМ: Сортировка и поиск. М., - 1978. - 844с.

39. *Кухарев Г.А.* и др. Техника параллельной обработки бинарних данньїх на СБИС. - М.: Виш. Шк., 1991. 226 с

40. Пат. №66138, Україна, МПК006Б7/38. Пристрій для обчислення сум парних добутків: Патент на корисну модель/ І.Г. Цмоць, О.В. Скорохода; заявник і патентовласник Національний університет«Львівська політехніка». -№ и201106811; заявл. 30.05.2011; опубл. 26.12.2011, Бюл. №24. 8 с..

41. Патент України на винахід №29700. Пристрій для визначення максимального числа з групи чисел. Бюл. №6-11. - 2000. Рашкевич Ю.М., Зербіно Д.Д, Цмоць І.Г.

42. Кун С. Матричние процессорина СБИС. - М.: Мир, 1991. 672

43. Галушкин А.И. Нейрокомпьютеры. Кн.3.-.М; ИПРЖР,2000.-528с.

44. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика. 2-е изд., стереотип. М.: Горячая линия-Телеком, 2002. 382 с.

45. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика М.: Горячая Линия-Телеком, 2002 382 с.

46. Brych V., Borysiak O., Brych B. Digital marketing of energy service companies' personnel in the context of socio-economic development // Strategies for sustainable socio-economic development and mechanisms their implementation in the global dimension : collective monograph / edited by M. Bez¬partochnyi, in 3 Vol. // VUZF University of Finance, Business and Entrepreneurship. Sofia : VUZF Publishing House «St. Grigorii Bogoslov», 2019. Vol. 3. P. 309-317.

47. с. Рассел С., Норвиг П. Искусственный интеллект: современный подход / Пер. с английского М.: Вильямс, 2007. 1408 с.

48. Рассел С., Норвиг П. Искусственный интеллект: современный подход/ Пер. с английского М.: Вильямс, 2007. 1408 с.

49. Цмоць І.Г. Принципи розробки і оцінка основних характеристик високопродуктивних процесорів на надвеликих інтегральних схемах/ Вісник ДУ "Львівська політехніка", №349, Львів, 1998 - с.5-11.

50. Батюк А.Є., Цмоць І.Г. Методи синтезу спеціалізованих обчислювальних систем для розв'язання задач у реальному часі/ Інформаційні технології і системи. Т2, №1, Львів1999 с.155-161.

51. Данілов П.О., Цмоць І.Г., Ігнатєв І.В. Апаратна реалізація обчислення максимального і мінімального чисед в масиві даних/ Сучасні комп'ютерні інформаційні технології.