Вінницький національний технічний університет
Факультет комп’ютерних систем та автоматики
Кафедра лазерної та оптикоелектронної техніки

Пояснювальна записка

до магістерської кваліфікаційної роботи

за освітньо-кваліфікаційним рівнем «магістр»

на тему:

ПАРАЛЕЛЬНИЙ ОПТИКО-ЕЛЕКТРОННИЙ СПЕЦПРОЦЕСОР ДЛЯ МАТРИЧНОЇ АЛГЕБРИ НА МОДУЛЯТОРАХ СВІТЛА

Виконав: студент 2-го курсу, групи ЛТО-18м
ОКР підготовки магістр
спеціальності 152 – метрологія та інформаційно-вимірювальна техніка
за освітньою програмою «Лазерна техніка та оптоінформатика»
Гончарук І. В. ____________________

Керівник: д.т.н., проф. каф. ЛОТ

Заболотна Н.І. _____________________

«___» ______________________ 2019 р.

Рецензент:

к.т.н. доцент Севастьянов В.М. ___________

«___» ______________________ 2019 р.

Вінниця ВНТУ - 2019 рік

ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних систем і автоматики
Кафедра лазерної та оптикоелектронної техніки
Освітньо-кваліфікаційний рівень: магістр
Спеціальність 152 «Метрологія та інформаційно-вимірювальна техніка»
Освітня програма «Лазерна техніка та оптоінформатика»

ЗАТВЕРДЖУЮ
Завідувач кафедри ЛОТ
д.т.н., проф. Заболотна Н.І.
____
«___» ___________ 2019 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Гончаруку Ігорю Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема магістерської кваліфікаційної роботи: Паралельний оптико-електронний спецпроцесор для матричної алгебри на модуляторах світла
керівник проекту (роботи) Заболотна Наталя Іванівна, д.т.н., проф.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ВНТУ від « » 2019 року № .

2. Строк подання студентом магістерської дипломної роботи:______________

3. Вихідні дані до магістерської дипломної роботи: ______________________

1 Функціональне призначення пристрою: визначення добутку матриць та обернення матриці.
2. Розмірність матриць NxN елементів, де N=320 для конкретної реалізації.
4.;Формат подання елементів матриць: з плаваючою точкою, де M – розрядність мантиси; P – розрядність порядку.
5. Спосіб оброблення – позрізовий цифровий.
5. Елементна база – модулятори світла на квантових ямах.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): Вступ. 1 Аналіз методів, структур та елементної бази для побудови спецпроцесора. 2 Паралельні базові моделі операцій матричної алгебри на основі позрізових обчислень та їх відображення на архітектуру оптико-електронного спецпроцесора. 3 Практична реалізація вузлів та блоків спецпроцесора із оцінюванням його технічних характеристик. 4 Економічна частина. Список джерел посилань. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): *1.Схема паралельного алгоритму обчислення добутку-обернення матриць на основі векторного добутку*
2. Схема структурна оптико-електронного спецпроцесора для добутку-обернення матриць на основі векторного добутку

3. Блок-схема алгоритму роботи оптоелектронного спецпроцесора для добутку-обернення матриць на основі векторного добутку
4. Схема структурна матричного зрізового накопичувального суматора з плаваючою точкою
5.Схема структурна блоку множення векторів з плаваючою точкою
6. Схема блоку множення вектора на обернений коефіцієнт

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Спеціальна частина	Заболотна Н. І. д.т.н., проф. каф. ЛОТ		
Економічна частина	Нікіфорова Л. О. к.е.н. доц. каф. ЕПВМ		

7. Дата видачі завдання «__» __________ 2019 р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Формування та затвердженя ТЗ		
2	Виконання спеціальної частини МКР. Перший рубіжний контроль виконання МКР		
3	Виконання спеціальної частини МКР. Другий рубіжний контроль МКР		
4	Виконання «Економічної частини»		
5	Попередній захист МКР		
6	Нормконтроль МКР		
7	Рецензування МКР		
8	Захист МКР		

Студент ________________ Локотей Д. Ю.
(підпис)

Керівник роботи ________________ Заболотна Н. І.
(підпис)

АНОТАЦІЯ

В даній магістерській кваліфікаційній роботі наведено вирішення наукової задачі забезпечення у комплексі високої швидкодії та багатофункціональності паралельних спецпроцесорів для матричної алгебри, побудованих на просторово-часових модуляторах світла.

Запропоновано паралельну модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність. Розроблено архітектуру оптико-електронного спецпроцесора для обчислень добутку-обернення матриць на основі векторного добутку в форматі з плаваючою точкою. Показано варіант практичної реалізації оптико-електронного спецпроцесора на оптично-керованих транспарантах із самонаведеним електрооптичним ефектом.

ADSTRACT

In this master's qualification work the solution of the scientific problem of providing in the complex of high-speed and multifunctionality of parallel special processors for matrix algebra, built on space-time modulators of light, is presented.

A parallel model of organization of calculations of the product-rotation of matrices on the basis of the vector product for the special processor of matrix operations is offered, which will allow to increase its multifunctionality. The architecture of the opto-electronic special processor for the calculation of the product-rotation of matrices based on the vector product in the floating-point format is developed. The variant of practical realization of opto-electronic special processor on optically-controlled banners with self-directed electro-optical effect is shown.

ЗМІСТ

ВСТУП

Актуальність. Значення ефективного виконання матричних операцій (додавання матриць, векторно-матричне множення та визначення добутку матриць, обернення матриць, знаходження максимального елемента вектора) сьогодні зростає для широкого кола прикладних задач. Зауважимо, що матрична алгебра слугує єдиним формалізованим інструментом для моделювання методів обробки оптичних сигналів та зображень [1-6], задачі моделювання штучних нейронних оптико-електронних мереж [7], розпізнавання образів [8]. Крім того, матричне представлення моделей відображає природний тип паралелізму в організації паралельних високопродуктивних обчислювальних системах та мережах. Тому задачу створення швидкодіючих паралельних спеціалізованих процесорів оброблення даних із розмірністю 10^6 елементів та вище пов’язують із ефективною реалізацією матричних операцій.

В той же час, матричні операції є досить зручними для виконання на двовимірних оптичних спепроцесорах. Сучасні електронні технології, що використовуються для створення кремнієвих процесорів наближаються до теоретичної межі своїх можливостей, а саме за фізичними обмеженнями швидкості розповсюдження електричного сигналу по чіпу, а також обмеженнями по мікромініатюризації. Вирішити поставлені задачі дозволяють сучасні оптико-електронні інформаційні технології, яким властивий природний паралелізм подання і оброблення інформації.

Такі переваги оптичних обчислювальних пристроїв як багатоканальна паралельна обробка великорозмірних масивів даних із організацією паралельного введення та виведення інформації, достатня розв’язка між входом та виходом, висока швидкодія, стимулюють подальший розвиток оптико-електронних спецпроцесорів для реалізації операцій матричної алгебри.

Відомі акустооптичні векторно-матричні помножувачі [9-10] долають проблему паралельного введення-виведення та організації багатовимірних оптичних зв’язків між процесорними елементами. Проте мають обмеження, пов'язані із точністю аналогового обчислення. Використання парафазного оптичного кодування [11-13] у відомих цифрових оптоелектронних спепроцесорах для визначення добутку матриць з плаваючою точкою призводить до високих апаратних витрат.

Запропоновані структури розрядно-зрізових спецпроцесорів для обернення матриць розмірності $N \times N$ елементів, орієнтовані на оптичні цифрові технології при реалізації, дозволяють отримати швидкодію на рівні 10^4 MFLOP або 10^{10} операц/с [14 – 17] при використанні в якості базових елементів просторово-часових модуляторів світла (ПЧМС) [18 – 20]. В той же час, відомі перспективні структурно-функціональні рішення оптичних розрядно-зрізових помножувачів матриць [21], орієнтованих на ПЧМС, також передбачають отримання високої продуктивності оброблення на рівні 10^{12} операцій/с.

Проте значне коло практичних задач, пов'язаних із синтезом спецпроцесорів для цифрової обробки сигналів та зображень, наприклад, вимагає реалізації операцій так званого «матричного мікроалфавіта», номенклатура якого вимагає підвищення багатофункціональності існуючих спецпроцесорів для матричної алгебри.

Отже, актуальною науково-прикладною задачею є необхідність забезпечення у комплексі високої швидкодії та багатофункціональності паралельних спецпроцесорів для матричної алгебри, побудованих на просторово-часових модуляторах світла.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі лазерної та оптикоелектроної техніки відповідно до науково-дослідної роботи 57 К7 «Оптико-електронні технології в лазерних, комунікаційних, біомедичних та інформаційних системах око-процесорного типу», що проводиться викладачами кафедри в 2018-2019 роках.

Мета і завдання дослідження. Метою роботи є підвищення багатофункціональності паралельного спецпроцесора для матричної алгебри з високою швидкодією обчислень за рахунок виконання ним додаткових матричних операцій, реалізованих на однотипних функціональних блоках, побудованих на двовимірних просторових модуляторах світла.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

– проаналізувати методи, структури та елементну базу для побудови паралельного оптико-електронного спецпроцесора для матричних операцій з високою швидкодією та багатофункціональністю;
– розробити паралельну модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність;
– розробити архітектуру оптико-електронного спецпроцесора для обчислень добутку-обернення матриць на основі векторного добутку в форматі з плаваючою точкою;
– розглянути аспекти практичної реалізації оптико-електронного спецпроцесора для паралельних обчислень добутку-обернення матриць на оптично-керованих транспарантах із самонаведеним електрооптичним ефектом;
– провести імітаційне моделювання розробленого спецпроцесора для паралельних обчислень добутку-обернення матриць для підтвердження адекватної роботи моделі;
– оцінити часові характеристики розробленого оптико-електронного спецпроцесора для матричної алгебри на модуляторах світла.

Об’єкт дослідження – процеси паралельного виконання операцій визначення добутку та обернення матриць, в паралельному оптико-електронному спецпроцесорі для матричної алгебри.

Предмет дослідження – моделі та архітектура паралельного оптико-електронного спецпроцесора для матричних операцій на основі двовимірних модуляторів світла.

Методи дослідження. При вирішенні поставлених задач застосовувались загальний теоретичний базис апарату лінійної алгебри, методи теорії синтезу паралельних обчислювальних систем; методи математичного та імітаційного моделювання; методи побудови паралельних оптоелектронних обчислювачів.

Наукова новизна одержаних реультатів:

Розвинуто підхід до побудови паралельного спецпроцесора матричних операцій з плаваючою точкою на основі оптичних позрізових обчислень, який за рахунок організації паралельного множення-обернення матриць на основі векторного добутку дозволяє підвищити багатофункціональність спецпроцесора при забезпеченні його високої швидкодії.

Практичне значення одержаних результатів:

Розроблено архітектурну організацію паралельного оптико-електронного спецпроцесора на основі векторного добутку на оптично-керованих модуляторах світла із самонаведеним електрооптичним ефектом, що дозволяє створювати спецпроцесор для матричних операцій з більш широкими функціональними можливостями та високою швидкодією на основі однотипних пристроїв.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується аргументованою постановкою мети й задач дослідження, повнотою формулювання умов, в яких вони розв’язуються та необхідними припущеннями і обмеженнями щодо застосування результатів, використанням сучасного математичного апарату та програмного забезпечення.

Особистий внесок здобувача. Усі результати отримано автором самостійно. У працях, опублікованих у співавторстві, магістранту належать: [22] концепція застосування матричної моделі при аналізі фазових розподілів лазерних зображень об’єкта.

Апробація результатів роботи. Окремі положення й результати досліджень доповідалися й обговорювалися на XLVIII науково-технічній конференції професорсько-викладацького складу та студентів факультету комп'ютерних систем і автоматики ВНТУ (м. Вінниця, березень 2019 р.).

Публікації. За тематикою дослідження, що проводилось в магістерській кваліфікаційній роботі, опубліковано 1 тези доповіді у співавторстві.

1 АНАЛІЗ МЕТОДІВ, СТРУКТУР ТА ЕЛЕМЕНТНОЇ БАЗИ ДЛЯ ПОБУДОВИ ПАРАЛЕЛЬНОГО ОПТИКО-ЕЛЕКТРОННОГО СПЕЦПРОЦЕСОРА МАТРИЧНИХ ОПЕРАЦІЙ

1.1 Місце та роль паралельних оптико-електронних спецпроцесорів матричних операцій в швидкодіючих обчислювальних системах

Досягнення сучасної елементної бази відкрило нові можливості для розвитку високопродуктивних та швидкодіючих інформаційних систем. Це забезпечується зокрема розпаралелюванням обчислювального процесу. Збільшення обчислювальної потужності мотивується в основному числовими моделюваннями складних інформаційних та інформаційно-вимірювальних систем для прикладних задач прогнозування погоди, клімату та глобальних змін в атмосфері; структурної біології та генетики людини; розпізнавання зображень [5, 6, 23 - 25].

Для розв’язання таких задач застосовуються алгоритми виконання матричних операцій: визначення добутку матриць, обернення матриць, формування матриць спеціального виду, об’єднання, переформування матриць і виконання розв’язання систем лінійних алгебраїчних рівнянь(СЛАР). Тому розробка високопродуктивних і високошвидкодіючих спеціалізованих процесорів з певною мірою високою функціональністю для розв’язання матричних задач в реальному часі є актуальною.

Моніторинг складних процесів, де є необхідність оброблення великорозмірних даних у реальному часі вимагає створення паралельних швидкодіючих обчислювальних систем, що були б орієнтовані на паралельні методи розв`язання поставлених задач.

Основними областями, де виникає така необхідність, є:

– медицина (реконструктивна томографія, мамографія та ін.);
– обробка сигналів адаптивних антенних решіток (радари в військовій

промисловості);
- прогнозування погоди, клімату і глобальних змін в атмосфері;
- побудова моделей нейронних мереж;
- розв`язання задач математичної фізики;
- економіка та економічні дослідження;
- адаптивне моделювання в геофізичних дослідженнях;
- створення образного комп`ютера:
  - розпізнавання великорозмірних зображень;
  - ідентифікація великорозмірних зображень;
  - обробка великорозмірних зображень;
  - моделювання зовнішніх середовищ;

Матричні представлення, з одного боку, є одним із основних принципів вирішення прикладних задач шляхом приведення їх формулювання до форми, придатної для вирішення на сучасних спеціалізованих обчислювачах. Тому є необхідність отримання нових паралельних інтерпретацій методів розв`язання задач лінійної алгебри, орієнтованих на матричне представлення, які б узгоджувалися зі структурою паралельної обчислювальної системи.

1.2. Аналіз обчислювальних структур для розв'язання матричних задач лінійної алгебри

Аналіз відомих оптичних паралельних процесорів для матричних обчислень зведений до таблиці 1.1.

Серед відомих оптичних паралельних процесорів для матричних обчислень найбільш функціонально-повним набором операції лінійної алгебри володіє розрядно-зрізовий спецпроцесор для обернення матриць та розв’язання систем лінійних рівнянь СЛАР, що здійснює названі операції за модифікованим методом Гаусса-Жордана [26]. За аналізом таблиці 1 в подальшому пропонується за базові обрати розглянути методи та структури для обернення матриць та визначення добутку матриць.

Таблиця 1.1 – Оптичні паралельні процесори для матричних операцій

Назва	Метод обробки інфо-рма-ції	Основний тип елементної бази	Розмір зобра-ження , піксе-лів	Характери-стики	Область застосу-вання
Цифровий оптичний пристрій фірми “Bell”	Цифро-вий	Матриці S-SEED приладів	4*8	Частота 1,1 МГц, число перемикань 40 Мб/с	Набір ма-тричн. логічних операцій над бі-нар-ними да-ними
DOC-II (digital optical computer)	Цифро-вий	Акусто-оптична брегівська комірка на основі напівп-ровідника GaP	64*128	Продуктив-ність 100 Мб/с	Векторно-матричні перетво-рення
HPOC (High performance Optoelektronic Com-munication)	Цифро-вий	Матриці переми-кання на основі дифракційних оп-тичних елементів	8*8	Продуктив-ність 4096 Тб/с	Векторно-матричні перетво-рення
Оптичний процесор Enlight256	Анало-говий	Матриця просто-рових модуляторів світла, що працю-ють на відбиття на основі GaAs/GaAlAs	256*256	Продуктив-ність $8*10^{12}$ опер/с	Векторно-матричне множення для розпі-зна-вання та обро-бле-ння зобра-жень
Розрядно-зрізовий спецпроцесор за мо-дифікованим мето-дом Гаусса-Жор-дана	Цифро-вий	S-SEED прилади	N*N	Для N=100 продуктив-ність $2*10^4$ MFLOPs	Розв`яза-ння СЛАР прямим методом

1.3 Аналіз паралельних помножувачів матриць

Класифікаційний огляд паралельних пристроїв множення матриць на період до 1995 року детально наведений в роботі [16]. Тому доречно зупинитись на нових перспективних оптико-електронних технічних вирішеннях, що з’явились пізніше.

Так, успішно розвиваються методи акустооптичного (АО) цифрового множення на основі аналогової згортки (DMAK- алгоритм) [9, 10], на основі яких було технічно просто реалізовано аналоговий акустооптичний векторно-матричний помножувач. Проте в багатьох задачах його обмежений динамічний діапазон не відповідав умовам необхідної точності, що звужувало його область застосування.

З метою підвищення точності було запропоновано при визначенні добутку чисел DMAK-алгоритм [9], який може зберігати задану точність цифрових обчислень у разі вилучення змішаних кодів, які виникають при таких операціях, із лінійного діапазону пристрою. В результаті, подаючи на входи такого помножувача компоненти векторів і матриць у вигляді двійкових кодів та використовуючи необхідні зв'язки між елемкентами, на виході помножувача отримуємо результати значної кількості попарно скалярних множень та додавань вхідних векторних і матричних компонент. Необхідні компоненти результуючого вектора подаються у з цифровою точністю, зменшено швидкості обробки, але підвищено точність обробки.

В роботі [10] розглядається схема побудови акустооптичного (АО) векторно-матричного помножувача, яка працює при умові виконання спеціального співвідношення між частотами світлових і акустичних хвиль (набір світлових частот $\nu_i = \nu_0 q^{-i}$, i=1,2,...,N, має відповідати частотам поверхневих акустичних хвиль $f_i = f_0 q^i$, i=1,2,...,N, де q – основа гармонійної послідовності). В результаті роботи такого АО-пристрою здійснюється обчислення цифрового добутку “оптичної матриці” $\tilde{a}_{mk} = a_{mk}(\nu_1, \nu_2, ..., \nu_N)$ та “звукового вектора” $\tilde{b}_k = a_k(f_1, f_2, ..., f_N)$. На виході послідовно в часі формуються компоненти результуючого вектора $\tilde{c}_{mk} = \sum_{i=1}^{N} \tilde{b}_i \tilde{a}_{ij}$ в змішаному коді [10].

Результати імітаційного моделювання робочих режимів і відповідних швидкостей обчислень з використанням експериментально отриманих величин динамічного діапазону на рівні 35 дБ і співвідношення “оптичного” і теплового шумів приблизно 10 дБ при T_0=10 мкс наведено в таблиці 1.2.

Таблиця 1.2 – Параметри робочих режимів для однорідного АО вікна [10]

$T_S \geq 0{,}13$ мкс	$T_S \geq 0{,}6$ мкс	$T_S \geq 2$ мкс
Розмірність матриці (КхМ)		
10×130	16×600	20×2000
Продуктивність		
$S_{bit} = 0{,}2 \cdot 10^{12}$	$S_{bit} = 1{,}2 \cdot 10^{12}$	$S_{bit} = 6 \cdot 10^{12}$
$S_{word} = 15 \cdot 10^{9}$	$S_{word} = 18 \cdot 10^{9}$	$S_{word} = 24 \cdot 10^{9}$

До недоліків розглянутого в [10] АО помножувача відносять те, що оцінки наведеної високої швидкодії обчислень обмежуються недостатньою швидкодією існуючих аналого-цифрових перетворювачів (АЦП), що реалізують декодування змішаного вихідного коду АО комірки в бінарний. Це обгрунтовує обмеження швидкодії АО помножувача можна на рівні 10^9 біт/с при досягненні максимально 16-розрядної точності.

В роботі [11] запропоновані оригінальні методи і високопродуктивні та точні оптичні цифрові конвеєрні спецпроцесори для виконання операцій визначення добутку матриць із використанням двійкового парафазного кодування.

Час T множення двох матриць розмірності n×m за даним методом , можна оцінити за формулою

$$T = \tau \log_2 n + T_\Sigma \log_2 n\,, \qquad (1.1)$$

де τ – час спрацьовування лазерного діода;

T_Σ – час виконання однієї операції додавання в оптичному суматорі.

Тоді продуктивність П пристрою для реалізації даного парафазного метода множення матриць можна визначити за виразом

$$\Pi = n \times m/(\tau/(\tau\ {}_2 n + T_\Sigma \log_2 n)\,. \qquad (1.2)$$

Із виразів (1.1) і (1.2) слідує, що при збільшенні розмірності матриць, час їх ноження зростає повільно за логарифмічним законом, а продуктивність при цьому зростає значно швидше.

Так,. при розмірності матриць n=m=1024 елементи час визначення добутку матриць складає близько 10^{-8} с, а продуктивність близько 10^{13} опер./с.

До недоліків розглянутих в роботах [11-13] спецпроцесорах для матричних операцій із парафазним кодуванням слід віднести надлишковість подання та обробки інформації.

В роботі [27] вищеназвані недоліки паралельних спецобчислювачів для добутку матриць усунуто шляхом застосування розрядно-зрізової обробки, паралельного введення-виведення матриць, реалізації оптичних взаємозв'язків між елементами та блоками паралельної структури. Це дозволило досягнути продуктивності цифрових обчислень на рівні10^{11}-10^{12} біт/с. Подальшого вдосконалення потребує в цьому випадку розширення діапазону подання вхідних матриць, що можна здійснити використовуючи формат даних з плаваючою точкою.

1.4 Аналіз паралельних пристроїв для обернення матриць

Аналізу методів паралельного обернення матриць присвячена робота [17]. Проаналізуємо відомі багатоканальні оптичні спецпроцесори для матричних операцій, зокрема й оптичні пристрої для обернення матриць.

Наведена на рис. 1.1 схема оптичного матричного спецпроцесора [6] реалізує в тому числі й обернення матриці за методом Гаусса-Жордана. Перетворення матриці **A** в верхню трикутну матрицю **U** здійснюють, виконуючи N помножень матриці $\mathbf{P}_k$ на розширену матрицю. Подібним чином перетворюють верхню трикутну матрицю **U** в діагональну одночасно із аналогічними перетвореннями над вектором **b** вільних членів системи алгебраїчних рівнянь на зворотному ході. В розширеній матриці результату на місці вектора **b** формується шуканий розв'язок **x** системи $\mathbf{A} \times \mathbf{x} = \mathbf{b}$.

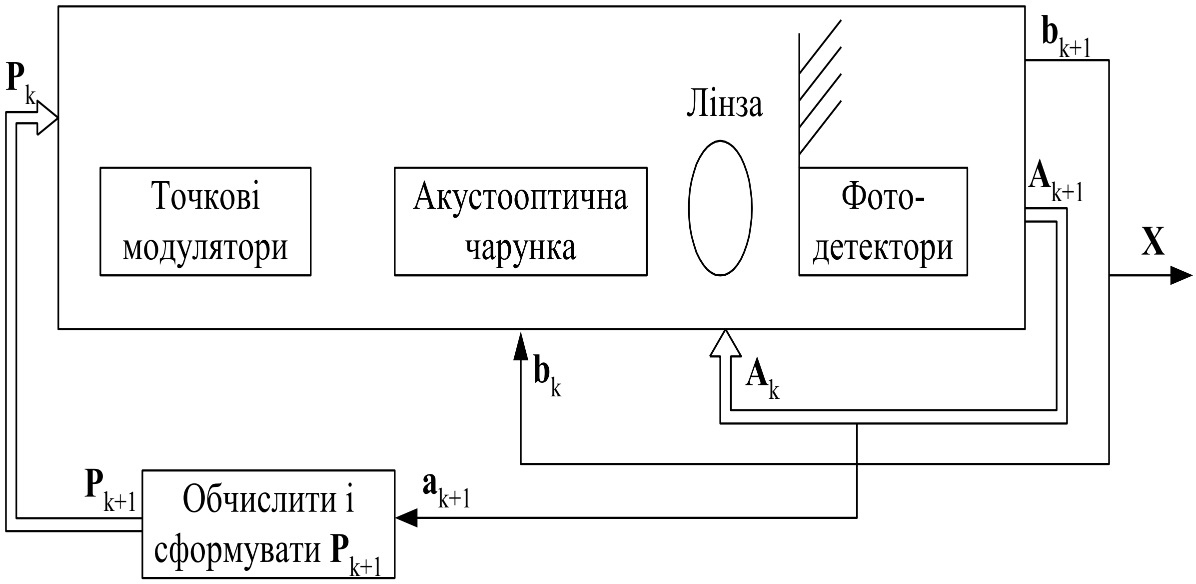


Рисунок 1.1 – Схема оптичного процесора для обернення матриць [6, 26]

Час обчислень для обернення матриць та розв'язання СЛАР оптичним процесором оцінюється за формулою:

$$T_7=N^2T_B/2, \tag{1.3}$$

де ТВ –тривалість такту множення матриці на вектор в АО спецпроцесорі із частотним розділенням каналів [6].

Отже, при наявності переваг акустооптичного подання сигналів в розглянутому спецобчислювачі для обернення матриць схема має гірші характеристики, ніж кращі систолічні варіанти схем за точністю обчислень за рахунок аналогової обробки.

Використовуючи алгоритми аналогової згортки [], досягають підвищення точності обробки безпосередньо цифрових сигналів, застосовуючи аналого-цифрове перетворення змішаного коду результату в бінарний код лінійкою із N аналого-цифрових перетворювачів (АЦП) з числом бітів $\log_2 S$. Тоді при застосуванні чотирьох лінійок АЦП з оптичними входами із 32 елементів і частотою дискретизації 5-10 МГц, швидкодія акустооптичного матричного СП оцінюється на рівні 10^9 біт.операцій/с. Зауважимо, що в цьому випадку існують обмеження на розмірність і розрядність матриць, які обробляються, обумовлених похибками АЦП.

В роботах [28, 29] наведені варіанти спецобчислювачів для розрядно-зрізового розв'язання СЛАР та обернення матриць в форматі з плаваючою точкою відповідно за методом Гаусса та модифікованим методом Гаусса-Жордана. Оцінки часових характеристик відповідно $T_Г$ та $T_{Г-Ж}$ наведених методів визначаються, виходячи із тривалостей відповідних тактів оброблення $\vartriangle T_Г$ і $\vartriangle T_{Г-Ж}$ та розмірності матриці N за формулами

$$T_Г = (2N-1)\times \vartriangle T_Г, \tag{1.4}$$

$$T_{Г-Ж} = N \times \vartriangle T_{Г-Ж}. \tag{1.5}$$

Отже, враховуючи високий рівень паралелізму розглянутих спецпроцесорів для обернення матриць та їх швидкодію, запропоновані в роботах [28, 29] моделі та структурні рішення спецпроцесорів можуть слугувати аналогами розробки.

Аналогом виступає розрядно-зрізовий СП для обернення матриць за методом Гаусса-Жордана, розроблений в дисертаційній роботі [26, 28, 29]. При високій швидкодії операцій обернення матриць (порядку 10^{10} опер. бітових/с або 10^4 MFLOP) та цифровій точності оброблення основним недоліком СП є обмежені функціональні можливості.

Оптикоелектронний СП, який розробляється, повинен забезпечувати можливість виконання крім операції обернення, ще й множення великорозмірних матриць, тобто підвищувати багатофункціональність. Орієнтація на просторово-часові модулятори світла при реалізації дозволить СП значно підвищить його швидкодію в порівнянні з окремими існуючими спецобчислювачами.

В таблиці 1.3 подані основні параметри і характеристики аналога та нової розробки.

Таблиця 1.3 – Основні параметри аналога і нової розробки

Показники	Одиниця виміру	Аналог	Нова розробка	Відношення параметрів нової розробки і аналогу
1. Кількість матричних операції	—	1	2	2/1
2.Спосіб обробки	—	Цифровий	Цифровий	—
3.Час обробки	с	Час обернення матриць $T1_{invert} = 0,016$	Час обернення матриць $T2_{invert} = 0,016$ Час визначення добутку матриць $T2_{multipl} = 0,008$	$T2_{invert} / T1_{invert} = 1$ $T2_{multipl} / T1_{invert} = 0,5$
4. Швидкодія	опер. /с	Обернення матриць $V1_{invert} = 1,38 \times 10^{10}$	Для обернення матриць $V2_{invert} = 1,38 \times 10^{10}$ Для множення матриць $V2_{multipl} = 2,2 \times 10^{7}$	$V2_{invert} / V1_{invert} = 1$ $V2_{multipl} / V1_{invert} = 1,6 \times 10^{-3}$

Основним технічним показником нової розробки є кількість матричних операцій, які можуть бути виконані даним спецпроцесором.

В порівнянні з аналогом, який виконує лише обернення матриць, СП, що розробляється, дасть можливість виконувати не лише обернення, але й множення великорозмірних матриць. Слід також зазначити, що при цьому будуть збережені досягнута висока швидкодія та точність обчислень аналога.

1.5 Аналіз елементної бази для реалізації паралельних сппроцесорів для матричних операцій

Оптико-електронні обчислювачі спеціального призначення характеризуються високою швидкістю передачі даних, високим ступенем паралелізму та низькою споживчою потужністю. Ці переваги найкраще демонструються при реалізації матричних операцій, які наділені природним паралелізмом, на просторово-часових модуляторах світла, зокрема на так званих оптично керованих транспарантах [18, 19, 30, 31].

В роботах [32 - 35] показано, що для оптоелектронних спеціалізованих обчислювальних систем найкраще застосовувати напівпровідникові транспаранти. Їм властива висока ступінь інтеграції в обчислювальні системи, враховуючи вимогу підвищеної швидкодії до останніх. Крім того, керування оптичними властивостями напівпровідника можна за допомогою різних факторів (електричної напруги, оптичного випромінювання, температури та інше).

В роботі [33] проаналізовано сучасні типи транспарантів, зокрема за швидкодією та максимальною розмірністю транспаранта, що висвітлено в табл. 1.4. Видно, що напівпровідникові транспаранти мають високу швидкодію (до 10^{-10} с), високий рівень інтеграції в систему та використання різних методів адресації. Тому можуть бути базовим елементом в структурі оптико-електронних паралельних спецпроцесорів для матричних операцій. Так, перспективні характеристики має оптоелектронний пристрій на основі транспарантів з повним набором логічних матричних операцій, описаний в роботі [36].

Даний пристрій [36] має ряд переваг над відомими транспарантами, оскільки є достатньою мірою універсальним при спрощених апаратурних витратах. При застосуванні таких пристроїв [36] в якості логічних матричних елементів І, АБО, НІ для оптоелектронного сторінкового пристрою порівняння з плаваючою комою суттєво скорочуються реалізаційні апаратні витрати.

Таблиця 1.4 – Аналіз транспарантів за швидкодією та максимальною розмірністю [33]

Тип транспаранта	Швидко-дія, с	Максимальна розмірність, пікселів
Рідкокристалічний	10^{-2}	1000 х 1000
На основі електрооптичної кераміки	10^{-7}-10^{-8}	одиниці х одиниці
На феромагнітних матеріалах	10^{-6}	десятки х десятки
На монокристалічних сегнетоелектриках	до 10^{-9}	одиниці х одиниці
Акустооптичний	10^{-6}-10^{-7}	сотні х сотні
Напівровідниковий	10^{-10}	256 х 256

Крім того, час виконання логічної матричної кон’юнкції складає $t_I = 19,73 пс;$ матричної диз’юнкції $\mathrm{t}_{АБО} = 49,75 пс;$ а операції матричного інвертування - $\mathrm{t}_{НІ} = 16,49 пс,$ що на два порядки менше в порівнянні з відомими транспарантами. Тому збільшення швидкодії сучасних оптико-електронних спецпроцесорів для матричних операцій можна однозначно пов’язувати із вищенаведеним пристроєм.

Серед вже комерційно виготовлених просторово-часових модуляторів світла найкращі результати отримано для так званих названих SEED (Self Electro-Optic Effect Device) [30, 37], на основі яких в 1990 р. фірмою AT&T Bell Laboratories був розроблений оптичний комп’ютер. Пласка фіксована система із 100 ввімкнених паралельно простих і компактних оптичних затворювачів, утворених в інтегральному виконанні чергуванням шарів GaAs і AlGaAs, покладена в основу базової структури типу SEED. Останні як фотодіоди під’єднуються до електронних логічних комірок, сформованих в підкладинці із GaAs. Світловий опромінюючий потік направлено перпендикулярно до площини кристала. Із них компонуються вузли на польових транзисторах – решітка інтелектуальних комірок. Керування здійснює зовнішній комп’ютер [20]. Час перемикання такого SEED біля 100 пс [20]. Оцінка фундаментального обмеження на швидкодію встановлена на рівні ≤1 пс, причому її можна “обмінювати” на потужність сигналу перемикання [20].

На основі цього базового елемента був розроблений тією ж компанією симетричний SEED (S–SEED), на основі якого реалізовано оптичний RS- тригер [38]. Два послідовно ввімкнених p-i-n-діодами за технологією GaAs та i-шаром в вигляді суперрешітки подають із себе S-SEED.

Відомо, що було виготовлено матриці із 32×64 пристроїв з p-i-n-діодами. Їх площа становила 100 мкм^2, енергієя спрацювання 1 пДж та час перемикання 1 нс [38]. За найближчими прогнозами для розробки матриць із 10^5 елементів час перемикання оцінювався як 1 нс. За даними, наведеними в роботі [39], зазначено, що що в структурах типу SEED час перемикання може бути знижений до 0,2 нс. Схема, реалізована на S–SEED-пристроях, що описана в роботі [40] має час перемикання, зменшений до 10 пс.

Розробки компанії LensLet Ltd (Ізраїль) демонструють прогресивний розвиток вищезазначених підходів щодо застосування ПЧМС на квантово-розмірних ямах як базових матричних елементів оптико-електронних спецпроцесорів для матричних операцій. Компанія реалізувала на SEED перший оптичний цифровий оброблювач сигналів EnLight 256 [41], щвидкодія якого склала 8-Тера (10^{12}) операцій з фіксованою точкою за секунду. Спецпроцесор EnLight 256 містить базовий блок у вигляді оптичного матричного помножувача із часовим періодом у 8нс для виконання множення вектора із 256 8-бітних елементів на матрицю 256х256 8-бітних елементів.

ПЧМС, який застосовувався в вищеназваному спецпроцесорі, має тип ABLAZE ™ 2D MQW [41] на квантових ямах. Його розміри 640х480 пікселів-модуляторів з частотою перемикання одного пікселя 20 ГГц (рис. 1.2).

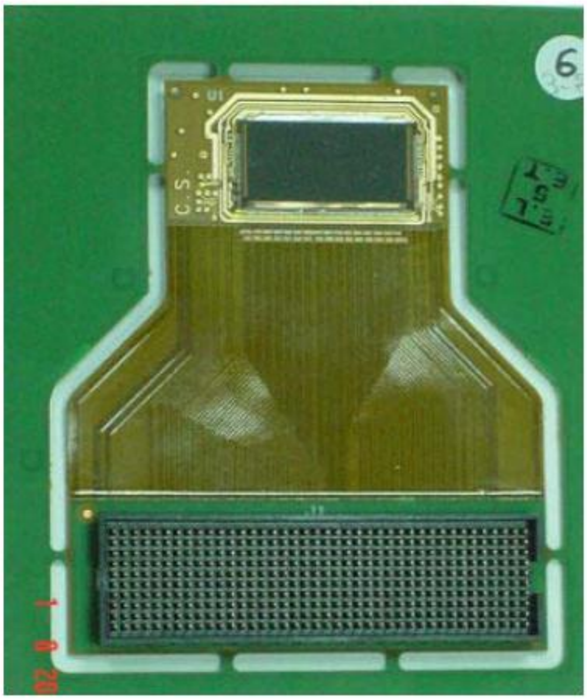


Рисунок 1.2 - Просторовий модулятор світла Ablaze

Висновки до 1 розділу

1. Показано, що існує ряд прикладних задач, ефективне розв’язання яких зводиться до розв’язання матричних задач з великорозмірними матрицями, що повинні оброблятися в реальному часі із високою точністю. Велика обчислювальна складність розв’язання цих задач обумовлює необхідність їх реалізації на спеціалізованих паралельних оптико-електронних процесорах на просторово-часових модуляторах світла .
2. Встановлено доцільність підвищення багатофункціональності паралельних спецобчислювачів для паралельного виконання матричних операцій, зокрема реалізації в одному паралельному спецпроцесорі операцій для визначення добутку матриць та обернення матриці.
3. Розглянуто структури для обернення матриць та визначення добутку матриць. Визначено розрядно-зрізові паралельні архітектури для матричних спецобчислювачів на модуляторах світла як такі, що володіють кращими характеристиками та мають розширені функціональні можливості.

2 ПАРАЛЕЛЬНІ БАЗОВІ МОДЕЛІ ОПЕРАЦІЙ МАТРИЧНОЇ АЛГЕБРИ НА ОСНОВІ ПОЗРІЗОВИХ ОБЧИСЛЕНЬ ТА ЇХ ВІДОБРАЖЕННЯ НА АРХІТЕКТУРУ ОПТИКО-ЕЛЕКТРОННОГО СПЕЦПРОЦЕСОРА

2.1 Принципи позрізового оптичного введення-виведення матриць та позрізових матричних обчислень в форматі з плаваючою точкою

Як було зазначено в роботах [14-17], для побудови паралельного швидкодіючого спецобчислювача для обернення матриць великої розмірності та розв'язання систем лінійних алгебраїчних рівнянь (СЛАР) високих порядків застосовувались позрізові паралельні обчислення [42]. Їх орієнтація на сучасні оптико-електронні інформаційні технології введення-виведення та оброблення даних, представлених у вигляді матриць, дозволила отримати перспективні часові характеристики спецобчислювачів для паралельного розв'язання СЛАР. Тому доцільно застосувати використані принципи позрізового оптичного введення-виведення та цифрового оброблення до розширення функціональних можливостей спецпроцесора для обернення матриць за рахунок реалізації ним додаткових матричних операцій, наприклад, визначення добутку матриць.

Розрядний зріз (РЗ) являє собою двійкову матрицю, утворену із значень однойменних розрядів кодів елементів матриць [14-17, 42]. При цьому оптимальним у випадку визначеного набору матричних операцій для реалізації спецпроцесором є застосування формату даних з плаваючою точкою. Надалі будемо працювати тільки з нормалізованими числами і виконувати нормалізацію всіх проміжних і кінцевих результатів.

Матричні операнди $\mathbf{X}$ і $\mathbf{Y}$ з розмірністю $N \times N$ елементів представимо в прямому коді в форматі даних з плаваючою точкою, використавши S=(M+P+2) розрядних зрізів [14-17, 42]

$$\mathbf{X} = \left\{ \mathbf{Sg}_{m_X=0}, \mathbf{X}_{m_X=1}, ..., \mathbf{X}_{m_X=M}, \mathbf{Sg}_{p_X=M+1}, \mathbf{X}_{p_X=M+2}, ..., \mathbf{X}_{p_X=S} \right\},$$
$$\mathbf{Y} = \left\{ \mathbf{Sg}_{m_Y=0}, \mathbf{Y}_{m_Y=1}, ..., \mathbf{Y}_{m_Y=M}, \mathbf{Sg}_{p_Y=M+1}, \mathbf{Y}_{p_Y=M+2}, ..., \mathbf{Y}_{p_Y=S} \right\}, \quad (2.1)$$

де $\mathbf{Sg}_{m_X=0}$ і $\mathbf{Sg}_{m_X=0}$ – зрізи знаків мантис матриць **X** і **Y**;

$\mathbf{X}_{(m_X)}$ і $\mathbf{Y}_{(m_Y)}$ – розрядні зрізи мантис матриць **X** і **Y**;

$\mathbf{Sg}_{p_X=M+1}$ і $\mathbf{Sg}_{p_Y=M+1}$ – зрізи знаків порядків матриць **X** і **Y**;

$\mathbf{X}_{(p_X)}$ і $\mathbf{Y}_{(p_Y)}$ – розрядні зрізи порядків матриць **X** і **Y**;

M і P – кількість розрядів на представлення мантис і порядків матриць **X** і **Y**.

Матриця **Z**, що є результатом обернення заданої матриці **X** або обчисленим добутком двох матриць $\mathbf{X} \times \mathbf{Y}$, подається у такому форматі:

$$\mathbf{Z} = \left\{ \mathbf{Sg}_{m_Z=0}, \mathbf{Z}_{m_Z=1}, ..., \mathbf{Z}_{m_Z=M}, \mathbf{Sg}_{p_Z=M+1}, \mathbf{Z}_{p_Z=M+2}, ..., \mathbf{Z}_{p_Z=S} \right\}. \quad (2.2)$$

Принципи паралельного оптичного введення та виведення матриць, які застосовуються в структурі спецпроцесора матричних операцій, що розробляється, говорять про те, що одночасно за один робочий такт можна подати на оптичний паралельний вхід або сформувати на паралельному оптичному виході базового вузла (елемента) спецпроцесора набір розрядних зрізів $S = M + P + 2$ поточного результату у форматі представлення з плаваючою комою.

2.2 Розробка паралельних моделей організації обчислень добутку-обернення матриць на основі векторного добутку

Вважаючи на те, що паралельна модель методу Гаусса-Жордана для обернення матриць на основі векторного добутку, яка описана в роботі [26], є ефективною з точки зору забезпечення необхідних вимог до часових характеристик обчислень, проаналізуємо її на предмет визначення можливості реалізації на основі її базових операцій також паралельної операції добутку матриць.

Спочатку розглянемо відому паралельну модель обернення матриць за методом Гаусса-Жордана на основі векторного добутку [26].

Нижче на рис. 2.1 зображено алгоритм паралельної організації обчислень добутку-обернення матриць на основі векторного добутку.

Постановка задачі передбачає здійснення процесу обернення квадратної числової матриці **X** розмірності $N \times N$ елементів**.** При цьому буде використовуватсь одинична матриця **E** такої ж розмірності.

В матрицю проміжного результату із подвійною розмірністю **R**[1:N;1:2N] вводяться початкові матриці $\mathbf{X}, \mathbf{E}$ на початку обчислень [26]:

$$\mathbf{R}^{(t=0)}[1:N;1:N]=\mathbf{X},\ \mathbf{R}^{(t=0)}[1:N;(N+1):2N]=\mathbf{E}. \qquad (2.3)$$

На поточному t-му кроці $\left(t=\overline{1,N}\right)$ обчислень обираємо елемент $r[t,t]\neq 0$ матриці **R** за ведучий елемент. Далі елементи t-го рядка матриці **R** ділимо на ведучий елемент і формуємо вектор-рядок **d**[1;1:2N].

Обчислюємо вектор-стовпець $\mathbf{v}^{(t)}[1:N;1]$, який має записану «1» у виділеному t-му стовпці матриці проміжного результату, що знаходиться в t-му рядку,

$$v^{(t)}[t;1]=-1. \qquad (2.4)$$

Обчислюємо елементи двовимірної матриці **P** як векторний добуток (⊗) вектор-рядка $\mathbf{d}^{(t)}$ і вектор-стовпця $\mathbf{v}^{(t)}[1:N;1]$ на основі співвідношення [26]

$$p^{(t)}[I;J]= v^{(t)}[I,1]\times d^{(t)}[1,J],\ (I=\overline{1,N}, J=\overline{1,2N}). \qquad (2.5)$$

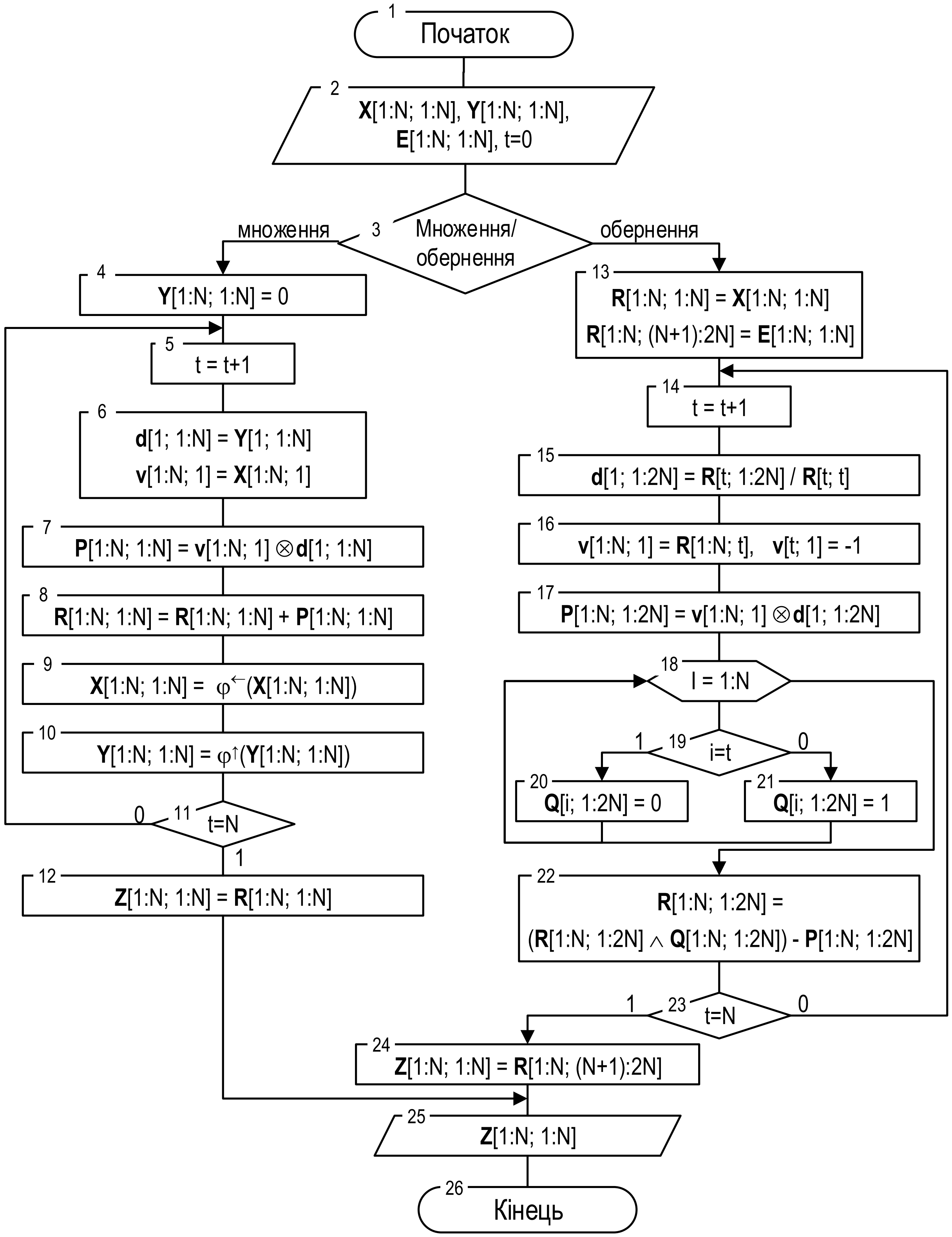


Рисунок 2.1 – Паралельний алгоритм обчислення добутку-обернення матриць на основі векторного добутку

Тоді проміжний результат **R** формується в часі $\left(t=\overline{1,N}\right)$ на основі матричного виразу :

$$\mathbf{R}^{(t)}=(\mathbf{R}^{(t-1)}\wedge\mathbf{Q}^{(t)})-\mathbf{P}^{(t)}, \tag{2.6}$$

де $\wedge$ - оператор логічного множення (кон'юнкція); $\mathbf{Q}^{(t)}$ –маска розмірності

$N\times 2N$ елементів, будь-який елемент q[i,j] якої визначається так:

$$q^{(t)}[i,j]=\begin{cases}0, & \text{якщо} \quad i=t, \quad j=\overline{1,2N};\\ 1, & \text{якщо} \quad i\neq t, \quad j=\overline{1,2N}.\end{cases} \tag{2.7}$$

Остаточний результат **Z**[1:N;1:N], що є оберненою матрицею $\mathbf{X}^{-1}$, отримаємо на N-му кроці обчислень, зчитуючи його із матриці **R**

Часова складність алгоритму Гаусса-Жордана для обернення матриць становить на основі даної моделі $T_{оберн.}$= N кроків.

Далі розглянемо модель множення матриць на основі векторного добутку.

Матриці, добуток яких визначаємо, записуються до масивів **X** і **Y** розмірністю N×N. Добуток цих матриць формується шляхом накопичення векторного добутку векторів **v** і **d**, сформованого у вигляді матриці **P** .

Вектор-стовпець **v** є першим стовпцем матриці **X**. Вектор рядок **d** є першим рядком матриці **Y**. Для наступних обчислень в на кожному кроці необхідно зсувати матрицю **X** на один рядок вправо, а матрицю **B** – на один рядок вверх.

Тоді на N–му кроці буде сформовано результат добутку вхідних матриць. який треба зчитати із відповідних елементів масиву **R**.

Часова складність паралельного визначення добутку двох квадратних матриць на основі векторного добутку складає $T_{множ.}$= N кроків.

Отже, можна визначити базові операції для розглянутої моделі паралельного множення-обернення матриць на основі векторного добутку. Це: є множення вектора на обернений коефіцієнт, обчислення векторного добутку векторів, паралельне алгебраїчне додавання матриць.

Покажемо роботу наведеної моделі на прикладах.

Приклад 1: Визначимо добуток матриць $\mathbf{X}$ та $\mathbf{Y}$

$$\mathbf{X} = \begin{pmatrix} 0,21 & -0,45 & -1,2 \\ 0,3 & 0,25 & 0,43 \\ 0,6 & -0,35 & 1,25 \end{pmatrix} \text{ і } \quad \mathbf{Y} = \begin{pmatrix} -3,2 & 0,55 & -2,3 \\ 0,5 & -6,5 & 0,5 \\ 0,6 & 0,5 & 0,45 \end{pmatrix}.$$

При t=0 маємо розширену матрицю

$$\mathbf{R}^{(t=0)} = \left(\begin{array}{ccc|ccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

Далі ($(t=1)$ формуємо вектор-рядок $\mathbf{d}^{(1)}[1;1:3]$ і вектор-стовпець $\mathbf{v}^{(1)}[1:3;1]$.

$$\mathbf{d}^{(1)} = (-3,2 \quad 0,55 \quad -2,3), \qquad \mathbf{v}^{(1)} = \begin{pmatrix} 0,21 \\ 0,3 \\ 0,6 \end{pmatrix}$$

На їх основі обчислюємо векторний добуток векторів $\mathbf{d}^{(1)}[1;1:3]$ і $\mathbf{v}^{(1)}[1:3;1]$.

$$\mathbf{P}^{(1)} = \mathbf{v}^{(1)} \otimes \mathbf{d}^{(1)} = \begin{pmatrix} -0,67 & 0,12 & -0,48 \\ -0,96 & 0,17 & -0,69 \\ -1,92 & 0,33 & -1,38 \end{pmatrix}$$

Тоді проміжний результат множення матриць $\mathbf{R}^{(1)}$ буде дорівнювати:

$$\mathbf{R}^{(1)} = \mathbf{R}^{(0)} + \mathbf{P}^{(1)} = \begin{pmatrix} -0,67 & 0,12 & -0,48 \\ -0,96 & 0,17 & -0,69 \\ -1,92 & 0,33 & -1,38 \end{pmatrix}$$

На наступному етапі (при t=2) маємо:

$$\mathbf{d}^{(2)} = (0,5 \quad -6,5 \quad 0,5), \qquad \mathbf{v}^{(2)} = \begin{pmatrix} -0,45 \\ 0,25 \\ -0,35 \end{pmatrix},$$

$$\mathbf{P}^{(2)} = \mathbf{v}^{(2)} \otimes \mathbf{d}^{(2)} = \begin{pmatrix} -0,23 & 2,93 & -0,23 \\ 0,13 & -1,63 & 0,13 \\ -0,18 & 2,28 & -0,18 \end{pmatrix},$$

$$\mathbf{R}^{(2)} = \mathbf{R}^{(1)} + \mathbf{P}^{(2)} = \begin{pmatrix} -0,9 & 3,05 & -0,71 \\ -0,83 & -1,46 & -0,56 \\ -2,1 & 2,61 & -1,56 \end{pmatrix}.$$

На етапі t=3 отримуємо такі значення обчислень:

$$\mathbf{d}^{(3)} = (0,6 \quad 0,5 \quad 0,45), \qquad \mathbf{v}^{(3)} = \begin{pmatrix} -1,2 \\ 0,43 \\ 1,25 \end{pmatrix},$$

$$\mathbf{P}^{(3)} = \mathbf{v}^{(3)} \otimes \mathbf{d}^{(3)} = \begin{pmatrix} -0,72 & -0,6 & -0,54 \\ 0,26 & 0,22 & 0,19 \\ 0,75 & 0,63 & 0,56 \end{pmatrix},$$

$$\mathbf{R}^{(3)} = \mathbf{R}^{(2)} + \mathbf{P}^{(3)} = \begin{pmatrix} -1,67 & 2,45 & -1,25 \\ -0,57 & -1,24 & -0,37 \\ -1,35 & 3,24 & -1 \end{pmatrix}.$$

Отже, результат обчислення добутку матриць **X** і **Y** сформовано у вигляді:

$$\mathbf{R} = \begin{pmatrix} -1{,}67 & 2{,}45 & -1{,}25 \\ -0{,}57 & -1{,}24 & -0{,}37 \\ -1{,}35 & 3{,}24 & -1 \end{pmatrix}.$$

Приклад 2: Визначимо обернену матрицю до матриці X.

$$\mathbf{X} = \begin{pmatrix} 0,21 & -0,45 & -1,2 \\ 0,3 & 0,25 & 0,43 \\ 0,6 & -0,35 & 1,25 \end{pmatrix}, \quad \mathbf{E} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Вигляд розширеної матриці $\mathbf{R}$ спочатку такий:

$$\mathbf{R}^{(t=0)} = \left(\begin{array}{ccc|ccc} 0,21 & -0,45 & -1,2 & 1 & 0 & 0 \\ 0,3 & 0,25 & 0,43 & 0 & 1 & 0 \\ 0,6 & -0,35 & 1,25 & 0 & 0 & 1 \end{array}\right)$$

На 1 етапі обчислень (t=1) вектор-рядок $\mathbf{d}^{(1)}[1;1:6]$ і вектор-стовпець $\mathbf{v}^{(1)}$ [1:3;1] мають вигляд:

$$\mathbf{d}^{(1)} = (1 \quad -2{,}14 \quad -5{,}71 | 4{,}76 \quad 0 \quad 0), \qquad \mathbf{v}^{(1)} = \begin{pmatrix} -1 \\ 0{,}3 \\ 0{,}6 \end{pmatrix}.$$

Обчислимо векторний добуток векторів $\mathbf{d}^{(1)}[1;1:6]$ і $\mathbf{v}^{(1)}$ [1:3;1] у вигляді

$$\mathbf{P}^{(1)} = \mathbf{v}^{(1)} \otimes \mathbf{d}^{(1)} = \left(\begin{array}{ccc|ccc} -1 & 2{,}14 & 5{,}71 & -4{,}76 & 0 & 0 \\ 0{,}3 & -0{,}64 & -1{,}71 & 1{,}43 & 0 & 0 \\ 0{,}6 & -1{,}28 & -3{,}43 & 2{,}86 & 0 & 0 \end{array}\right).$$

Маска $\mathbf{Q}$ на даному кроці виглядає так:

$$\mathbf{Q}^{(1)} = \left(\begin{array}{ccc|ccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right).$$

Проміжний результат $\mathbf{R}^{(1)}$ виглядає в цьому випадку як

$$\mathbf{R}^{(1)} = (\mathbf{R}^{(0)} \wedge \mathbf{Q}^{(1)}) - \mathbf{P}^{(1)} = \left(\begin{array}{ccc|ccc} 1 & -2{,}14 & -5{,}71 & 4{,}76 & 0 & 0 \\ 0 & 0{,}89 & 2{,}14 & -1{,}43 & 1 & 0 \\ 0 & 0{,}93 & 4{,}68 & -2{,}86 & 0 & 1 \end{array}\right).$$

На наступному етапі (t=2) маємо:

$$\mathbf{d}^{(2)} = (0 \quad 1 \quad 2{,}4 | -1{,}61 \quad 1{,}12 \quad 0), \qquad \mathbf{v}^{(2)} = \begin{pmatrix} -2{,}14 \\ -1 \\ 0{,}93 \end{pmatrix}.$$

$$\mathbf{P}^{(2)} = \mathbf{v}^{(2)} \otimes \mathbf{d}^{(2)} = \left(\begin{array}{ccc|ccc} 0 & -2{,}14 & -5{,}14 & 3{,}45 & -2{,}4 & 0 \\ 0 & -1 & -2{,}4 & 1{,}61 & -1{,}12 & 0 \\ 0 & 0{,}93 & 2{,}23 & -1{,}5 & 1{,}04 & 0 \end{array}\right).$$

$$\mathbf{Q}^{(2)} = \left(\begin{array}{ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right).$$

$$\mathbf{R}^{(2)} = (\mathbf{R}^{(1)} \wedge \mathbf{Q}^{(2)}) - \mathbf{P}^{(2)} = \left(\begin{array}{ccc|ccc} 1 & 0 & -0{,}57 & 1{,}31 & 2{,}4 & 0 \\ 0 & 1 & 2{,}4 & -1{,}61 & 1{,}12 & 0 \\ 0 & 0 & 2{,}45 & -1{,}36 & -1{,}04 & 1 \end{array}\right).$$

На етапі t=3 обчислення виглядають так:

$$\mathbf{d}^{(3)} = (0 \quad 0 \quad 1 | -0{,}56 \quad -0{,}42 \quad 0{,}41), \qquad \mathbf{v}^{(3)} = \begin{pmatrix} -0{,}57 \\ 2{,}4 \\ -1 \end{pmatrix},$$

$$\mathbf{P}^{(3)} = \mathbf{v}^{(3)} \otimes \mathbf{d}^{(3)} = \left(\begin{array}{ccc|ccc} 0 & 0 & -0,57 & 0,32 & 0,24 & -0,23 \\ 0 & 0 & 2,4 & -1,34 & -1,01 & 0,98 \\ 0 & 0 & -1 & 0,56 & 0,42 & -0,41 \end{array}\right),$$

$$\mathbf{Q}^{(3)} = \left(\begin{array}{ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right),$$

$$\mathbf{R}^{(3)} = (\mathbf{R}^{(2)} \wedge \mathbf{Q}^{(3)}) - \mathbf{P}^{(3)} = \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 2,16 & 0,23 \\ 0 & 1 & 0 & -0,27 & 2,13 & -0,98 \\ 0 & 0 & 1 & -0,56 & -0,42 & 0,41 \end{array}\right).$$

Таким чином, отримуємо обернену матрицю до матриці $\mathbf{X}$:

$$\mathbf{X}^{-1} = \mathbf{R} = \begin{pmatrix} 1 & 2,16 & 0,23 \\ -0,27 & 2,13 & -0,98 \\ -0,56 & -0,42 & 0,41 \end{pmatrix}.$$

Отримані результати підтверджують правильність роботи запропонованих моделей визначення оберненої матриці та формування добутку матриць на основі спільної для них базової операції – формування векторного добутку векторів.

2.3 Оцінювання часових характеристик позрізової моделі паралельного обчислення добутку-обернення матриць з урахуванням формату даних

Вище було визначено набір ключових арифметричних операцій для організації обчислень добутку-обернення матриць, а саме: векторний добуток векторів (ВДВ); паралельне додавання матриць; паралельне множення вектора на обернений коефіцієнт. Паралельні позрізові алгоритми виконання зазначених операцій, які виконують в форматі даних з плаваючою комою, були раніше запропоновані в роботах [26, 43, 44]. Їх особливість полягає в тому, що вони гіпотетично не залежать від розмірності даних, над якими виконуються позрізові паралельні операції в рамках концепції природного паралелізмц цифрових оптичних картинних обчислень [26].

З врахуванням їх особливостей оцінимо часові складності гілки виконання алгоритму множення на основі ВДВ та гілки виконання обернення на основі ВДВ (див. рис. 2.1). За попередніми оцінками, що зроблені вище, кожна гілка алгоритму виконується в режимі позрізових паралельних обчислень за N кроків обчислень.

Нагальною є потреба встановлення тривалості одного кроку обчислень. Його будемо визначати як оптимальний максимум із тривалостей кроків обчислення відповідно добутку матриць $\Delta T_{доб.м.}$ та обернення матриці $\Delta T_{оберн.}$, що визначені нижче за такими співвідношеннями:

$$\Delta \mathrm{T}_{доб.м.} = \mathrm{T}_{ВДВ} + \mathrm{T}_{ДОД.М.} + \mathrm{t}_{ЗАП.} + 2\mathrm{t}_{ЗС.}, \tag{2.9}$$

$$\Delta \mathrm{T}_{оберн.м.} = \mathrm{T}_{МВК} + \mathrm{T}_{ВДВ} + \mathrm{T}_{ДОД.М.} + \mathrm{t}_{Ф} + 2\mathrm{t}_{I}, \tag{2.10}$$

де $\mathrm{T}_{МВК}$ – час паралельного множення вектора на обернений коефіцієнт в форматі плаваючої точки;

$\mathrm{T}_{ВДВ}$ – час паралельного обчислення векторного добутку векторів в форматі плаваючої точки;

$\mathrm{T}_{ДОД.М.}$ – час паралельного додавання матриць в форматі плаваючої точки;

$\mathrm{t}_{ЗАП.}$ – час запису матриць в тривимірні масиви в форматі плаваючої точки;

$\mathrm{t}_{ЗС.}$ – час просторового паралельного матричного зсуву ;

t_{I} – час кон’юнкції просторового коду і маски в формі з плаваючою комою;

$\mathrm{t}_{Ф}$ – час формування вектора-стовпця.

Всі складові доданки в формулах (2.9) та (2.10) будемо оцінювати за кількістю кроків оброблення одного розрядного зрізу $\Delta T_{РЗ}$ просторового коду матриць, представлених у форматі з плаваючою точкою.

Часові характеристики (мінімальні та максимальні) відомого застосованого в даному випадку методу паралельного додавання матриць у форматі з плаваючою точкою [26, 43], що представлений M розрядними зрізами мантиси та P розрядними зрізами порядку, можемо отримати за таким співвідношенням:

$$\begin{aligned}&\min \mathrm{T}_{ДОД.М.} = \left(3\mathrm{P} + 3\mathrm{M} + 8\right) \cdot \Delta \mathrm{T}_{РЗ},\\ &\max \mathrm{T}_{ДОД.М.} = \left(4\mathrm{P} + 4\mathrm{M} + \mathrm{MP} + 11\right) \cdot \Delta \mathrm{T}_{РЗ}.\end{aligned} \tag{2.11}$$

Часові характеристики позрізового паралельного алгоритму множення вектора на обернений коефіцієнт [44], який може бути застосований в розглядуваній моделі паралельного виконання множення-обернення матриць, не залежать від розмірності вектора, який обробляється.

У випадку множення вектора розмірності N, представленого M розрядними зрізами мантиси та P розрядними зрізами порядку, на обернений коефіцієнт, представлений в такому ж форматі даних, час $\mathrm{T}_{МВК}$ можна оцінити за формулою

$$\mathrm{T}_{МВК} = \left(\mathrm{M}^2 + 7\mathrm{M} + 2\mathrm{P} + 12\right)\Delta \mathrm{T}_{РЗ}. \tag{2.12}$$

Паралельний позрізовий алгоритм визначення векторного добутку векторів [26], представлених в форматі плаваючої точки, що може бути застосований в розглядуваній моделі множення-обернення матриць, є також незалежним за часом обробки від розмірності векторів, які множаться.

Час паралельного позрізового обчислення векторного добутку векторів розмірності N, представлених M розрядними зрізами мантиси та P розрядними зрізами порядку, за вказаним алгоритмом визначається як [26]

$$\mathrm{T}_{ВДВ} = \left(\mathrm{M}^2 + 3\mathrm{M} + 2\mathrm{P} + 3\right)\Delta \mathrm{T}_{РЗ}. \tag{2.13}$$

Можна прийняти, що тривалості виконання операцій паралельного зрізового запису $t_{ЗАП.}$, зсуву $t_{ЗС}$, формування векторів $t_{Ф}$ та логічного множення t_{I} просторового коду векторів на маску відбуваються практично за крок оброблення одного розрядного зрізу, тобто

$$t_{ЗАП.} = t_{ЗС} = t_{Ф} = t_{I} = \Delta T_{РЗ}, \qquad (2.14)$$

де $\Delta T_{РЗ}$ – час оброблення одного розрядного зрізу матриці.

Підставивши в (2.9) та (2.10) вирази (2.11) – (2.14), обрахуємо час виконання одного кроку обчислення добутку матриць та відповідно обернення матриці, що становитиме відповідно:

$$\Delta T_{доб.м.} = \Delta T_{РЗ} \times (M^2 + MP + 7M + 2P + 17), \qquad (2.15)$$

$$\Delta T_{оберн.м.} = \Delta T_{РЗ} (2M^2 + MP + 14M + 8P + 28). \qquad (2.16)$$

Таким чином, оскільки тривалості кроків (2.15) та (2.16) відрізняються більше ніж в 2 рази, то приймемо їх значення як окремо тривалості кроку обернення та тривалості кроку множення.

Тоді час обчислення добутку матриць $T_{добут.м.}$ та обернення матриці $T_{оберн.м.}$ на основі векторного добутку векторів визначатимуться відповідно за формулами:

$$\mathrm{T}_{добут.м.} = \Delta \mathrm{T}_{РЗ} \times \mathrm{N} \times \left(\mathrm{M}^2 + \mathrm{M} \cdot \mathrm{P} + 7\mathrm{M} + 2\mathrm{P} + 17 \right), \qquad (2.17)$$

$$\mathrm{T}_{оберн.м.} = \Delta \mathrm{T}_{РЗ} \times \mathrm{N} \times \left(2\mathrm{M}^2 + \mathrm{MP} + 14\mathrm{M} + 8\mathrm{P} + 28 \right). \qquad (2.18)$$

Таким чином, в магістерській кваліфікаційній роботі розроблена паралельна модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність при забезпеченні високої швидкодії.

2.4 Розробка архітектурної організації оптико-електронного спецпроцесора для матричних операцій в форматі з плаваючою точкою

Для розробки архітектури паралельного спецпроцесора для матричних операцій, що виконуються за технікою позрізового оброблення, необхідно орієнтуватись на технологію оптичних цифрових картинних обчислень, описану детально в роботах [42, 26, 44]. Основною її особливістю є можливість паралельно вводити, обробляти та виводити великорозмірні розрядні зрізи матриць у паралельні оптоелектронні двовимірні або тривимірні блоки та вузли, які реалізуються, наприклад, на двовимірних просторово-часових модуляторах світла [26, 21, 32].

Враховуючи зазначену методологію відображення паралельних матричних алгоритмів на архітектуру їх спецпроцесорів та розроблений алгоритм, представлений на рисунку 2.1, розроблена структурна схема оптико-електронного спецпроцесора для паралельного добутку-обернення матриць, зображена на рисунку 2.2. Для опису її роботи застосовано розроблена блок-схему алгоритму спецпроцесора, подану на рисунку 2.3.

Схема містить оптичний картинний вхід, на який подається матриця $\mathbf{X}$, та оптичний картинний вхід, на який подається матриця $\mathbf{Y}$ при визначенні добутку двох матриць, чи одинична матриця $\mathbf{I}$ при визначенні оберненої матриці $\mathbf{X}$ (умовна вершина 3, рисунок 2.3).

Матричні операнди із названих входів записуються у тривимірні паралельні оптоелектронні регістри відповідно RGA та RGB за S зрізами просторового коду в форматі з плаваючою точкою. Названі регістри реалізують функції паралельного запису та збереження матриць, паралельного зсуву матриці ліворуч та вгору на 1 елемент. Регістр RGA містить (N×S) паралельний оптичний вихід, який формується виходами S бітів N елементів 1-го стовпчика матриці елементів RGA. Регістр RGB містить також двовимірний (N×S) паралельний оптичний вихід, який формується виходами S бітів N елементів 1-го рядка матриці елементів RGB.

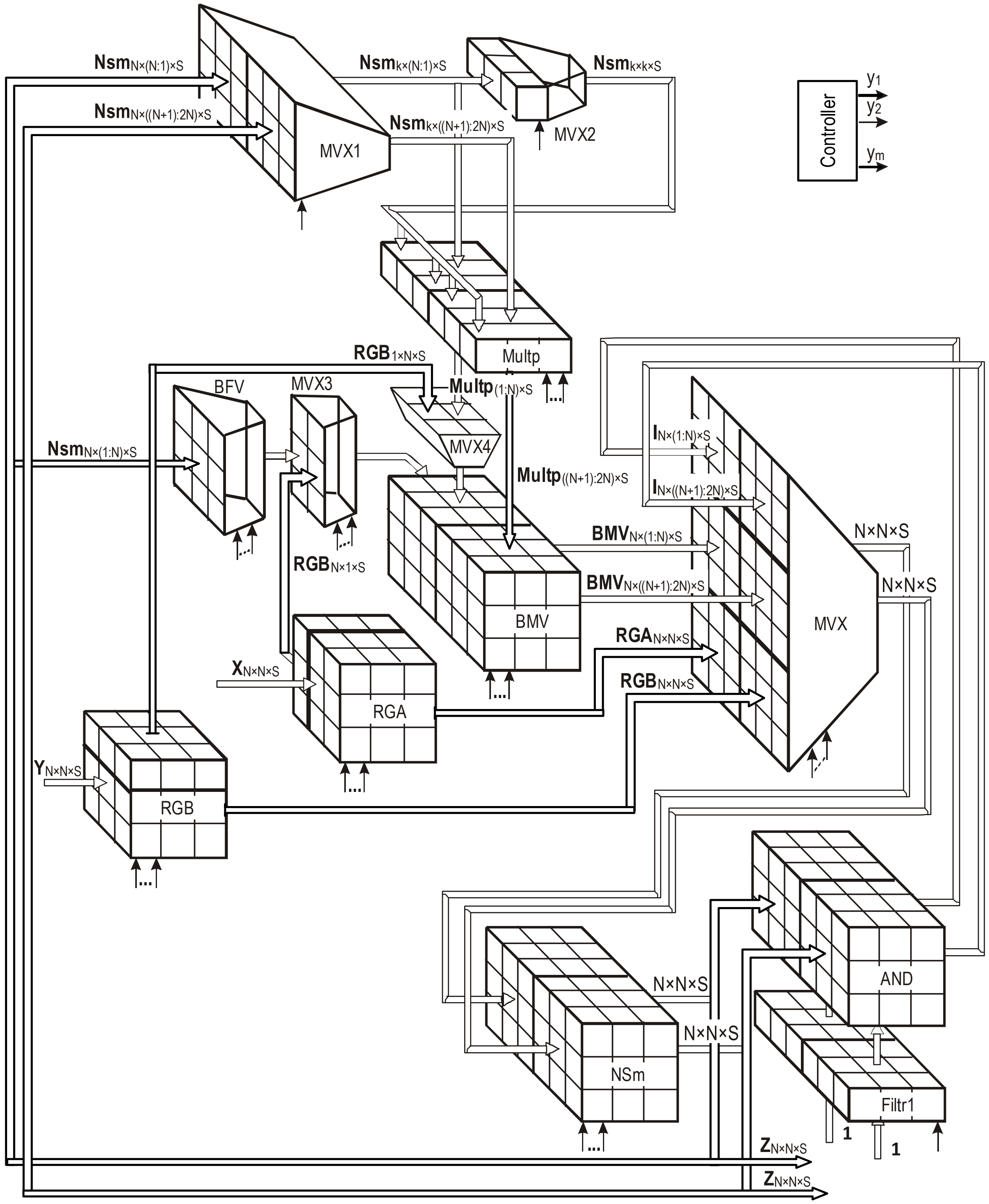


Рисунок 2.2 – Структурна схема оптоелектронного спецпроцесора для добутку-обернення матриць на основі векторного добутку

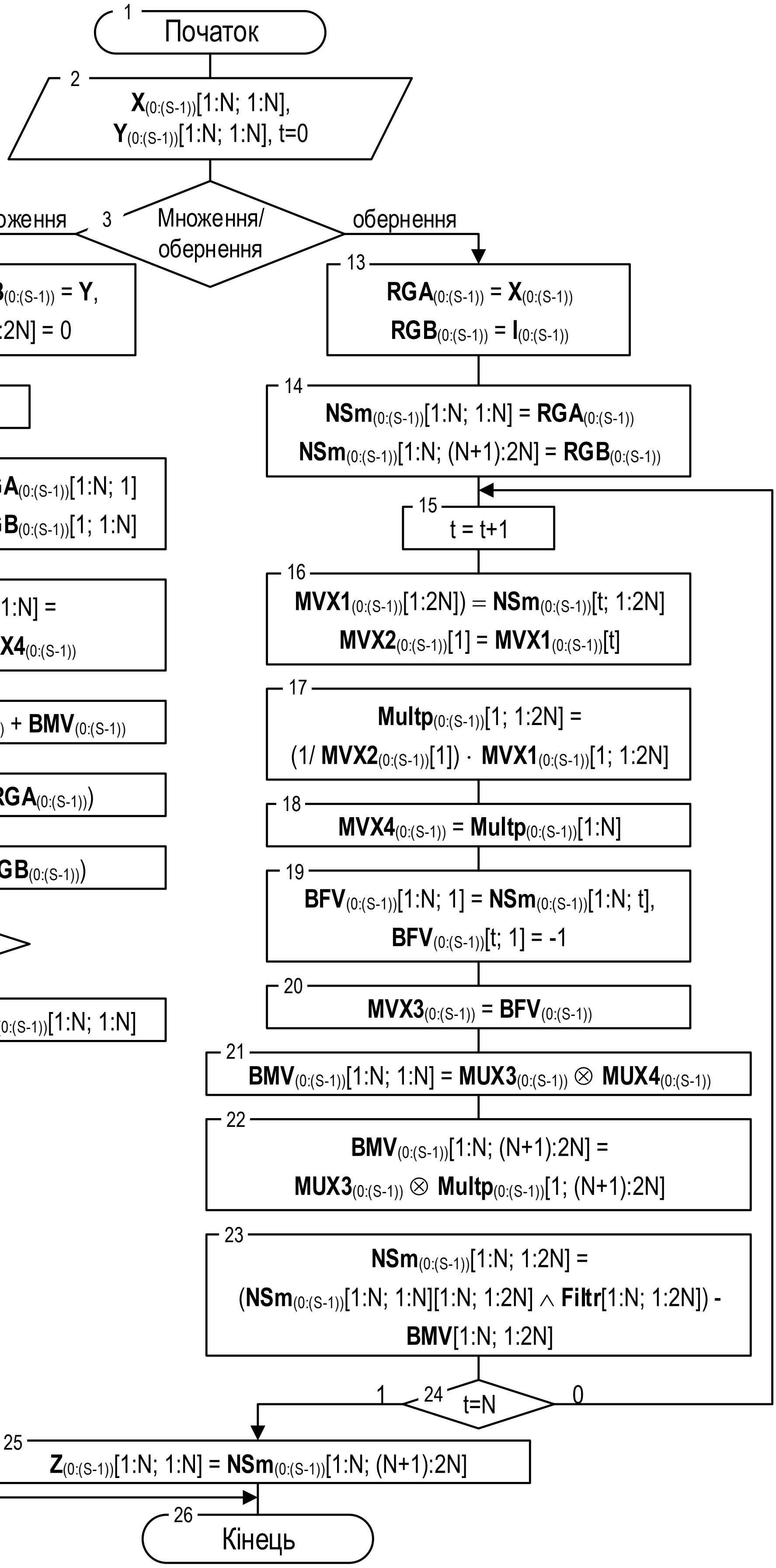


Рисунок 2.3 – Блок-схема алгоритму функціонування оптоелектронного спецпроцесора для добутку-обернення матриць на основі векторного добутку

Розглянемо режим визначення добутку матриць. Тривимірний нагромаджувальний оптоелектронний суматор NSm із розмірністю, що вдвічі перевищує розмірність матриць по площині (1:N; 1:2N), та представлення чисел в форматі із плаваючою точкою із S розрядних зрізів, обнуляється попередньо (вершина 4, рисунок 2.3).

Подальший цикл множення, представлений вершинами 5 – 9 блок-схеми (рис. 2.3) триває N циклів. За допомогою комутаторів $MUX3$ та $MUX4$ комутуємо оптичні (N×S) сигнали, що сформовані на виходах регістрів RGA і RGB, відповідно на два оптичні входи оптоелектронного блоку векторного добутку векторів BMV (вершини 6, 7, рисунок 2.3). На виході останнього формується матриця, що являє собою частковий добуток.

Накопичення часткових добутків здійснюється на паралельному оптоелектронному суматорі NSm шляхом подачі на його паралельний оптичний вхід матриці сигналів з виходу блоку BMV. Для підготовки нової інформації для обробки здійснюємо паралельний зсув ліворуч та вгору на один дискрет у регістрах відповідно RGA та RGB (вершина 8, рисунок 2.3).

Отже, відпрацювавши N тактів, маємо сформовану матрицю результату на оптичному виході суматора NSm (вершина 10, рисунок 2.3).

Розглянемо режим обернення матриці. Для цього режиму необхідно використати всю подівійну розмірність по площині нагромаджувального суматора NSm. Перші (1:N;1:N) паралельні оптичні входи суматора NSm з'єднано з паралельними оптичними виходами регістра RGA, а другі входи

(1:N; (N+1):2N) з'єднано з аналогічними виходами регістра RGB.

В нагромаджувальному суматорі NSm подвоєної розмірності попередньо запам'ятовуються матриці **X** та **I**, представлені розрядними зрізами у формі з плаваючою точкою (вершина 12, рисунок 2.3).

Алгоритм обернення матриць реалізується за N тактів.

Вважаючи, що базовою операцією для визначення добутку матриць та обернення матриць є обчислення векторного добутку векторів, структурна схема спецпроцесора містить блок векторного добутку векторів BMV, що працює в форматі з плаваючою точкою. Блок BMV містить S перших і S других N паралельних рядкових входів, S паралельних N стовпцевих входів та по S перших та других двовимірних N×N виходів.

Вектор рядок із N елементів формується на виході блоку $Multipl$ множення вектора на обернений коефіцієнт в форматі з плаваючою точкою. На його входи подається за допомогою матричного комутатора MUX1, що має по S перших та других матричних входів та по S перших та других N-канальних виходів, множник у вигляді вектора. Код значення оберненого коефіцієнта поступає на блок $Multipl$ через комутатор $MUX2$, який в кожний t-й момент часу комутує на вихід t-й елемент у рядку, який обрав комутатор $MUX1$. Паралельні входи $MUX1$ пов’язані із відповідними паралельними виходами суматора NSm.

Вектор стовпець із N елементів формується за допомогою блоку формування вектора BFV (вершина 18, рис. 2.3), на вхід якого поступає в t-й момент часу t-й ($t = \overline{1, N}$) вектор-стовпець проміжного результату із NSm. Його специфіка виражається прописуванням значення коду коефіцієнта «-1» на кожному $t-$ му такті обчислень (вершина 18, рис. 2.3).

Для трансформації записаної інформації в NSm в схемі застосовується паралельна схема матричного логічного множення AND з першими S і другими S картинними входами першої групи, зв’язаними з відповідними виходами суматора NSm, а також з аналогічними входами входами другої групи. На вхід останніх подається специфічна маска, що формується схемою $Filtr$ (вершина 22, рисунок 2.3). Перші S і другі S картинні виходи елемента AND через комутатор MUX зв’язані із відповідними входами суматора.

Матричний комутатор MUX комутує одну із трьох матриць (з виходу регістра RGB, з виходу блоку BMV чи з виходу схеми AND) на вхід NSm, на виходах якого через N тактів формується у вигляді наборів розрядних зрізів результат обернення матриці.

Таким чином, в магістерській кваліфікаційній роботі розроблено паралельну структурну схему спецпроцесора для визначення добутку матриць та обернення матриць за паралельним алгоритмом на основі обчислення векторного добутку векторів. Зазначимо, що в цьому разі виконувалась умова, що фізичний розмір структури однозначно узгоджується із розмірністю вхідних даних.

Час обчислення добутку матриць $T_{добут.м.}$ та обернення матриці $T_{оберн.м.}$ на основі векторного добутку векторів визначаються за вищенаведеними формулами відповідно (2.17). (2.18).

Аналіз розробленої структурної схеми оптоелектронного СП для паралельного множення-обернення матриць дозволив встановити базові функціональні вузли і блоки, які будуть розкриті в наступних пунктах. Це, насамперед, нижченаведені базові блоки:

1) паралельний цифровий матричний нагромаджувальний суматор NSm з плаваючою точкою;
2) паралельний цифровий помножувач вектора на обернений коефіцієнт в формі з плаваючою точкою;
3) паралельний цифровий блок обчислення векторного добутку векторів з плаваючою точкою.

3 АСПЕКТИ ПРАКТИЧНОЇ РЕАЛІЗАЦІЇ ОПТИКО-ЕЛЕКТРОННОГО СПЕЦПРОЦЕСОРА ДЛЯ ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ ДОБУТКУ-ОБЕРНЕННЯ МАТРИЦЬ

3.1 Комп’ютерне моделювання операції обернення матриці на основі векторного добутку векторів

Мета комп’ютерного моделювання – підтвердження дієздатності паралельного методу обернення матриць на основі векторної моделі визначення добутку векторів. При цьому враховується необхідність псевдопаралелізму, де паралельні процеси та операції в форматі з плаваючою комою та розрядно-зрізовим принципом представлення та оброблення матриць моделюються послідовним ходом виконання операцій, але при цьому вважається незмінним час їх виконання.

При моделюванні були розроблені послідовні алгоритми, які інтерпретують відповідні паралельні алгоритми на однопроцесорному комп’ютері. Для спрощення отриманих алгоритмів дії, які часто повторюються були винесені в окремі алгоритми. Операція, яка найчастіше використовується –запис в регістр, в ній організовано три вкладених цикли. В зовнішньому циклі послідовно організуємо роботу зі зрізами для матриць даних, що записуємо. У вкладених циклах послідовно переписуємо вміст одного РЗ в інший. Вказані цикли будуть використовувати змінні, які змінюються від 1 до N.

Для операції ініціалізації (скид) матричного регістра застосовуються дії, подібні до попередньо описаних з відмінністю лише в тому, що в якості операнду використовується скаляр, а не матриця.

Розроблено алгоритм виконання логічних операцій над РЗ. В двох вкладених циклах відбувається послідовне проходження масиву та виконання відповідної логічної операції.

Арифметичні операції, що зустрічаються в паралельному алгоритмі визначення добутку – обернення матриць, реалізуються на основі базових логічних операцій над РЗ. На такій основі розроблено об'єктно-орієнтовану програмну модель вказаного алгоритму, лістинг якої наведено у додатку И.

Застосований об'єктно-орієнтований принцип програмування є більш зручним при моделюванні завдяки можливості об'єднання даних (об'єктів) та операцій над ними (методів) в класи. Зокрема було виділено такі класи:

1. Cell – модель подання даних в системі (двовимірні просторово-часові розрядні зрізи) та інтерфейс для введення-вивдення даних.
2. Logic – модель базової комірки запропонованої архітуктури, включає в себе логічні операції над бінарними матрицями для роботи з класами Cell.
3. Adder – модель розрядно-зрізового накопичувального сумматора матриць, який заснований на логічних операціях (клас Logic) над бінарними матрицяма класу Cell.
4. Divider – модель розрядно-зрізового поділювача, який заснований на логічних операціях (клас Logic) над бінарними матрицяма класу Cell.

Програмне забезпечення розроблено на мові рівня С++. Результат виконання операції обернення матриці за модельованим алгоритмом наведено в додатку И.

3.2 Нагромаджувальний матричний оптико-електронний суматор з плаваючою точкою

В роботі [43] було розроблено метод паралельного алгебраїчного додавання матриць в форматі з плаваючою точкою.

Структура нагромаджувального матричного суматора, який реалізує цей метод, містить блок паралельного оброблення мантис матриць доданків та блок паралельного оброблення порядків матриць доданків (рисунок 3.1).

В тривимірні оптоелектронні регістри РгМ1 та РгМ2 записуємо вхідну матрицю мантис та матрицю мантис, що сформована на поточному кроці, які представлено в форматі з плаваючою точкою за бінарними зрізами.

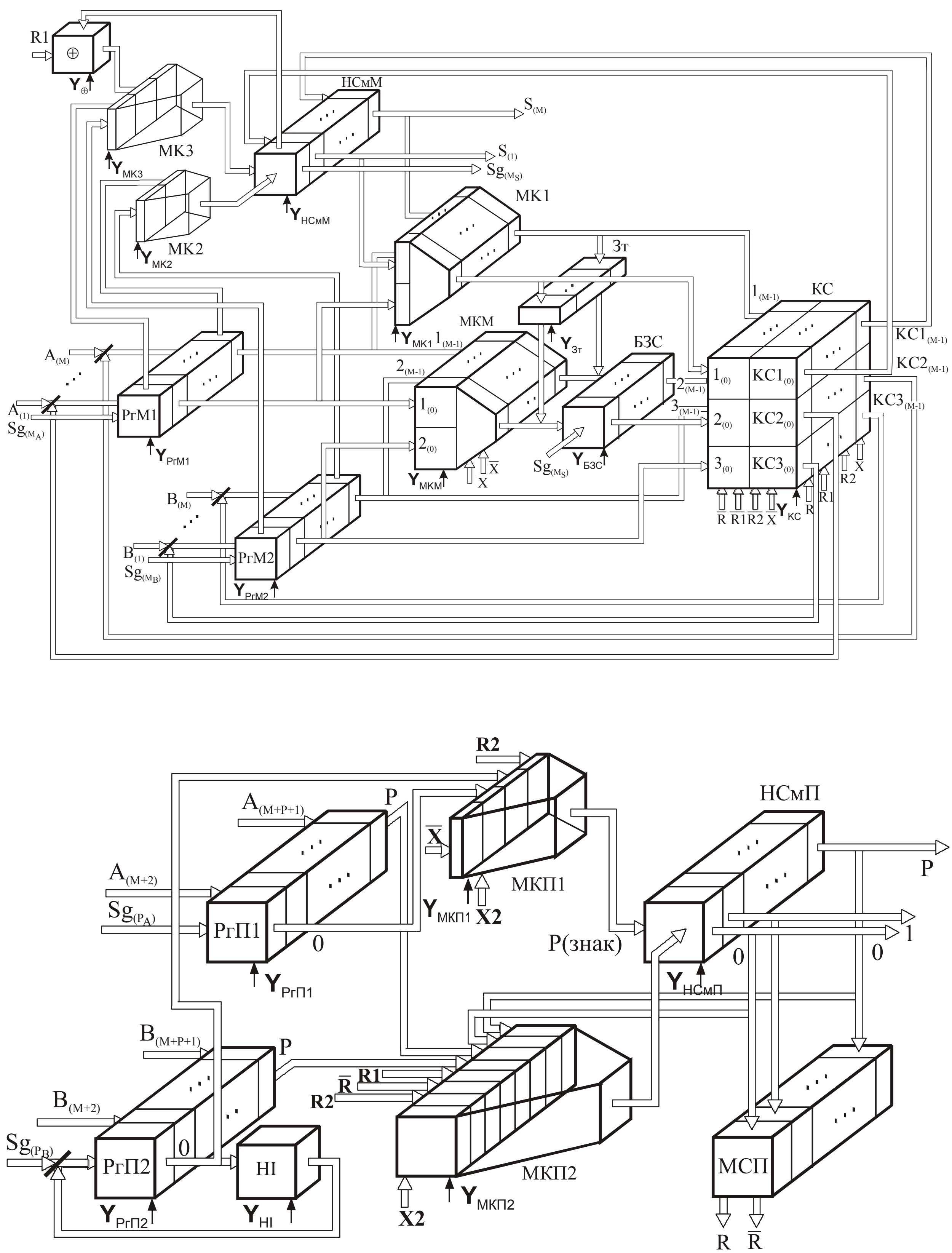


Рисунок 3.1 - Схема структурна матричного нагромаджувального суматора з плаваючою точкою

В тривимірні регістр РгП1 та РгП2 записуємо вхідну матрицю порядків та поточну матрицю порядків в аналогічному форматі, що й мантиси вхідних матричних доданків. Зазначені регістри мають відповідно (M+1) і (P+1) паралельних матричних входів-виходів та паралельний матричний знаковий вихід, а також паралельний матричний вихід молодшого зрізу.

Розглянемо роботу суматора.

Крок1. Реалізуємо віднімання матриць порядків операндів за допомогою   картинного накопичувального суматора зображень НСмП [45]. Знакові та інформаційні РЗ матриць порядків доданків подаємо на матричні входи суматора НСмП через матричні комутатори МКП1 і МКП2. Для формування від'ємної знакової матриці застосовуємо елемент НІ картинного типу. Ознака $\mathbf{X2}$ у вигляді доповняльного коду матриці знаків проміжної суми матриць порядків доданків, $\mathbf{X} = \mathbf{Sp}_{(0)}^{(t)}$, запам'ятовується в оптоелектронному матричному тригері.

Ознаку **R** формує матрична оптоелектронна схема порівняння МСП, представлена на рис. 3.1. Ознака **R** відображає результат перевірки долсіджуваної матриці на рівність матриці **E**.

В залежності від матриці сигналів, утворених на паралельних виходах **R**, **X**, **X1**, організовано наступне перетворення матриць мантис операндів, що супроводжуються відповідними змінами в матрицях порядків операндів.

Для вирівнювання матриці порядку необхідно збільшити менші елементи матриці порядку до більшого значення. Це роблять шляхом алгебраїчного додавання зрізів матриці $\left\{\overline{\mathbf{X}}, 0, 0, \ldots, \overline{\mathbf{R}}\right\}$, до проміжної суми матриці порядків доданків, записаної в НСмП. Знаковий РЗ вказаної матриці, передаємо через комутатор МКП1 на знаковий матричний вхід суматора, а інформаційні зрізи специфічної матриці через комутатор МКП2 на інформаційний матричний вхід НСмП.

Вирівнювання матриць порядків реалізуються за допомогою матричного комутатора МКМ із сигнальними та матричними керуючими входами, блоку просторового зсуву БЗС та матричної комбінаційної схеми КС.

Якщо якийсь елемент ознаки $\mathbf{X1}=0$, то при нерівних порядках вхідних матриць та при відповідних значеннях ознак **X** і **X1** будемо зсувати відповідні елементи матриці, сформованої із мантис вхідних матриць. Відповідний елемент ознаки $\mathbf{X}$ визначає операнд, з якого зчитуватиметься встановлений елемент.

Таким чином, за допомогою матричного комутатора МКМ, зв'язаного за входами-виходами із з М матричними інформаційними виходами регістрів РгМ1 і РгМ2 та керуючих сигналів $\mathbf{X}$ і $\overline{\mathbf{X}}$, формується специфічна матриця мантиси, про яку йшла вище мова. Далі вона подається на М матричних входів блоку просторового зсуву БЗС.

Блок БЗС має зробити просторовий зсув на один зріз праворуч. Так на його М матричних виходах утворюється перетворена специфічна матриця мантиси. Ознаки $\mathbf{R}, \overline{\mathbf{R}}, \mathbf{X}, \overline{\mathbf{X}}$ визначають, що буде записано на місця вивільнених елементів регістрів РгМ1 і РгМ2 за допомогою комбінаційної матричної схеми КС. Матричний комутатор МК1 комутує з виходів регістра РгМ1 або з виходів суматора НСМ сигнали на групу 1 входів схеми КС.

Група 2 та група 3 матричних входів схеми КС зв'язано відповідно з матричними виходами блоку БЗС та з матричними виходами регістра РгМ2. Групи 1-3 виходів схеми КС зв'язано із матричними входами суматора НСмМ, регістра РгМ1 і регістра РгМ2. Ці кроки повторюємо, поки не зрівняємо порядки. Зрівняну матрицю порядку записуємо в доповняльному коді в суматор в НСмП.

Крок 2. Накопичувальний матричний суматор НСмМ реалізує алгебраїчне додавання матриць мантис при зрівняних порядках. Зазначимо, що в процесі виконання алгоритму матриця результату формується на виходах суматора НСмМ.

Крок 3. Ознаки $R1$ та $R2$ сигналізують про порушення нормалізації мантиси результату відповідно ліворуч та праворуч в блоці формування ознак.

Для нормалізації мантиси в першому випадку за допомогою матричного суматора Корегування мантиси денормалізованого результату реалізують використовуючи схеми БЗС, КС, МК1 і затвору Зт із записом її в прямому коді на структурі НСмМ. При цьому є зміни в порядку результату, які реалізуються суматором НСмП із залученням в роботу комутаторів МКП1 та МКП2.

При денормалізації мантиси праворуч корекцію реалізують схеми блоку БЗС для просторового зсуву на один зріз в бік старших РЗ, комбінаційної схеми КС та суматора НСмП, який модифікує порядок денормалізованого результату.

Крок 4. Результат додавання матриць, поданий набором зрізів в форматі з плаваючою точкою, формується на виходах суматора НСмП, де попередньо відбувається переведення результуючого коду із доповняльного в прямий код.

З урахуванням реалізації базових елементів схеми, що розглядається на матричних логічних елементах, оцінимо час $\mathrm{T}_{add.matr.}^{float}$ роботи паралельного суматора з плаваючою точкою за формулою

$$T_{add.mart.}^{float} = 8(4P + 4M + MP + 11) \times \tau_{\log}, \tag{3.1}$$

Де $\tau_{\log}$ - час розповсюдження сигналу через базовий матричний логічний елемент.

В подальшому в якості базового матричного логічного елемента буде використовуватись двовимірний просторово-часовий модулятор світла. Він може мати паралельний оптичний вхід та паралельний оптичний вихід, який при наявності оптичного входу керування називається оптично керованим транспарантом, при наявності електричного входу керування називається електрично керованим транспарантом.

3.3 Оптико-електронний блок множення вектора на обернений коефіцієнт в форматі з плаваючою точкою

В роботі [46] був розроблений паралельний алгоритм множення вектора на обернений коефіцієнт в форматі з плаваючою точкою. Розробимо схему, яка

реалізує даний алгоритм, орієнтуючись на природній паралелізм оптичних цифрових технологій. Наведемо схему на рисунку 3.2.

Початкові дані у вигляді мантиси вектора **a** та мантиси оберненого до константи b коефіцієнта подаються за зрізами коду з плаваючою точкою відповідно через комутатор МКМ1 та МКМ2 до суматора НСмМ та регістра РгВ. При цьому знаковий РЗ мантиси оберненого коефіцієнта b має від'ємне значення. Зрізи порядку коефіцієнта b записуються в регістрі РгП. Якщо можливо виконати операцію множення вектора на обернений коефіцієнт $1/b$, що рівносильно діленню кожного елемента вектора безпосередньо на коефіцієнт b, то знаковий зріз порядку коефіцієнта b інвертується. Далі розглянемо дії по крокам виконання.

Крок 1. Складемо за допомогою суматора за сумою «два» знакові розрядні зрізи мантис вектора **a** та вектора, утвореного із коефіцієнта b. Знаковий РЗ, який утворився в результаті, запишемо в картинному тригері Тр.

Крок 2. Оскільки множення на обернений коефіцієнт можна замінити діленням на прямий коефіцієнт, то реалізуємо останній алгоритм на основі ділення модулів мантис векторів **a** і **b** без відновлення остачі при формуванні порядку результату ділення.

Алгоритм ділення виконуємо, застосовуючи матричний суматор обробки мантис НСмМ для того, щоб від від модуля вектора мантиси **a** відняти мантису дільника b. Аналізуючи знаковий РЗ остачі, будемо отримувати поточний зріз результату ділення. Його в процесі зсування ліворуч вмісту запишемо в молодший зріз картинного регістра PrR.

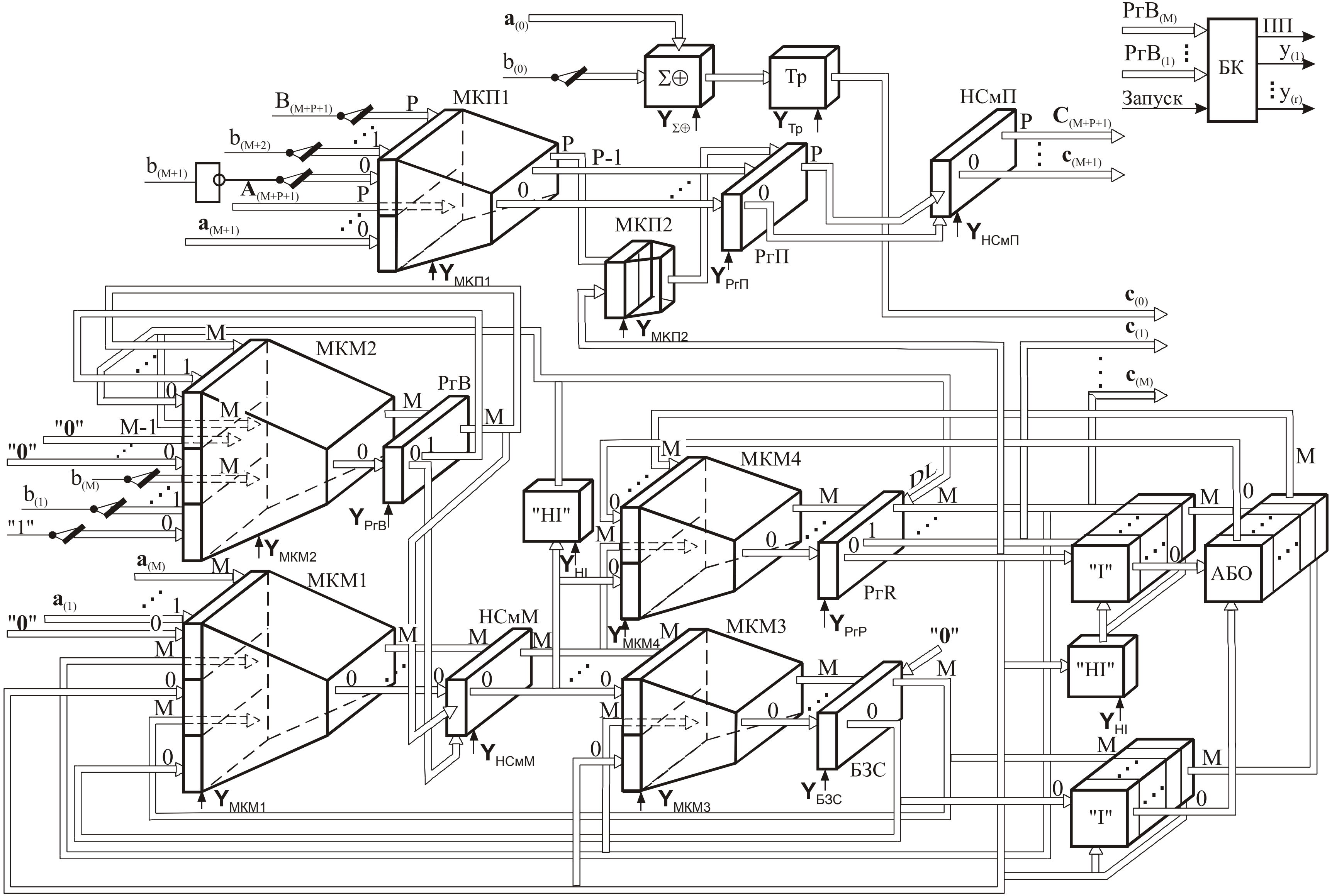


Рисунок 3.2 – Схема структурна блок множення вектора на обернений коефіцієнт з плаваючою точкою

При цьому необхідно помножити на «2» вектор матриці остачі. Це можна виконати на спеціальному картинному блоці зсуву БЗС через зсув записаної інформації вліво.

Одночасно з цим за допомогою нагромаджувального суматора НсмП алгебраїчно віднімаємо від вектора порядку **a** вектор порядку коефіцієнтів b.

Крок 3. Після отримання вектора результату може виникнути необхідність в його округленні та нормалізації. Для реалізації традиційного правила округлення необхідно запам'ятати молодший зріз вектора мантиси частки в РгВ та додати його до останнього вектора зрізу в регістрі РгR частки на структурі суматора НСмМ.

Якщо треба нормалізувати мантису вектора результату, то зробимо це, зсуваючи її на один зріз праворуч, якщо $PгR_{(0)} = 1$. А порядок при цьому маємо збільшити на «+1» . виконуючи відповідні дії на суматорі НСмП.

Крок 4. Виводимо із картинного тригера знаковий РЗ мантиси результату, а з М картинних виходів регістру РгR та з (Р+1) картинних виходів НСмП отримуємо мантису та порядок результату , поданих в прямому коді.

З урахуванням реалізації базових елементів схеми, що розглядається на матричних логічних елементах, оцінимо час роботи паралельного суматора з плаваючою точкою за формулою

$$\mathrm{T}_{multipl}^{float} = 8(M^2 + 7M + 2P + 12)\tau_{\log} \qquad (3.2)$$

де $\tau_{\log}$ - час розповсюдження сигналу через базовий матричний логічний елемент, яким слугуватиме просторовий оптично керований транспарант.

3.4 Паралельний блок визначення векторного добутку векторів з плаваючою комою

В роботі [28] був розроблений паралельний алгоритм обчислення зовнішнього добутку векторів в формі з плаваючою точкою.

Розробимо структурну схему векторного блоку для реалізації алгоритму.

Схема даного блоку містить апаратуру, яка обробляє розпаралелено в часі мантиси векторів та порядки векторів (рисунок 3.3).

Розглянемо з чого складається блок картинної обробки мантис. На паралельні входи картинних регістрів РгМА і РгМВ цього блоку паралельно подаються мантиси векторів **a** і **b** у вигляді зрізів коду, але без знаку.

Вказані регістри можуть паралельно зсувати записану в них інформацію, так РгМА зсуває вправо, а РгМВ зсуває вліво. Елементи МЗМА і МЗМВ розмножають або копіюють початкові дані до розмірності картини (N×N). Далі утворені матричні картинки логічно множаться за допомогою картинного блоку І, так формується частковий добуток. Останній через матричний комутатор МК поступає на картинний вхід нагромаджувального картинного суматора НсмМ, який виконаний з можливістю домножати вхідну картину на ваговий коефіцієнт [47]. Для нормалізації мантиси використовується блок зсуву БЗС.

Розглянемо, з чого складається блок картинної обробки порядків векторів. Так, в схемі є два регістри РгПА і РгПВ, в які вводяться та запам'ятовуються порядки векторів за розрядними зрізами. Використовуємо апаратуру, подібну до тієї, що використовується при обробці мантис. Через комутатори КЦКПА та КЦКПВ, а також розмножувачі векторів до рівня матриць МЗПА, МЗПВ та матричний комутатор МК подаємо по черзі на інформаційний картинний та знаковий картинний входи суматора НСмП вектор-стовпець, а потім вектор-рядок, векторний добуток яких визначається. З метою почергового їх пропускання до блоку НСмП використовуємо затвори З1 та З2.

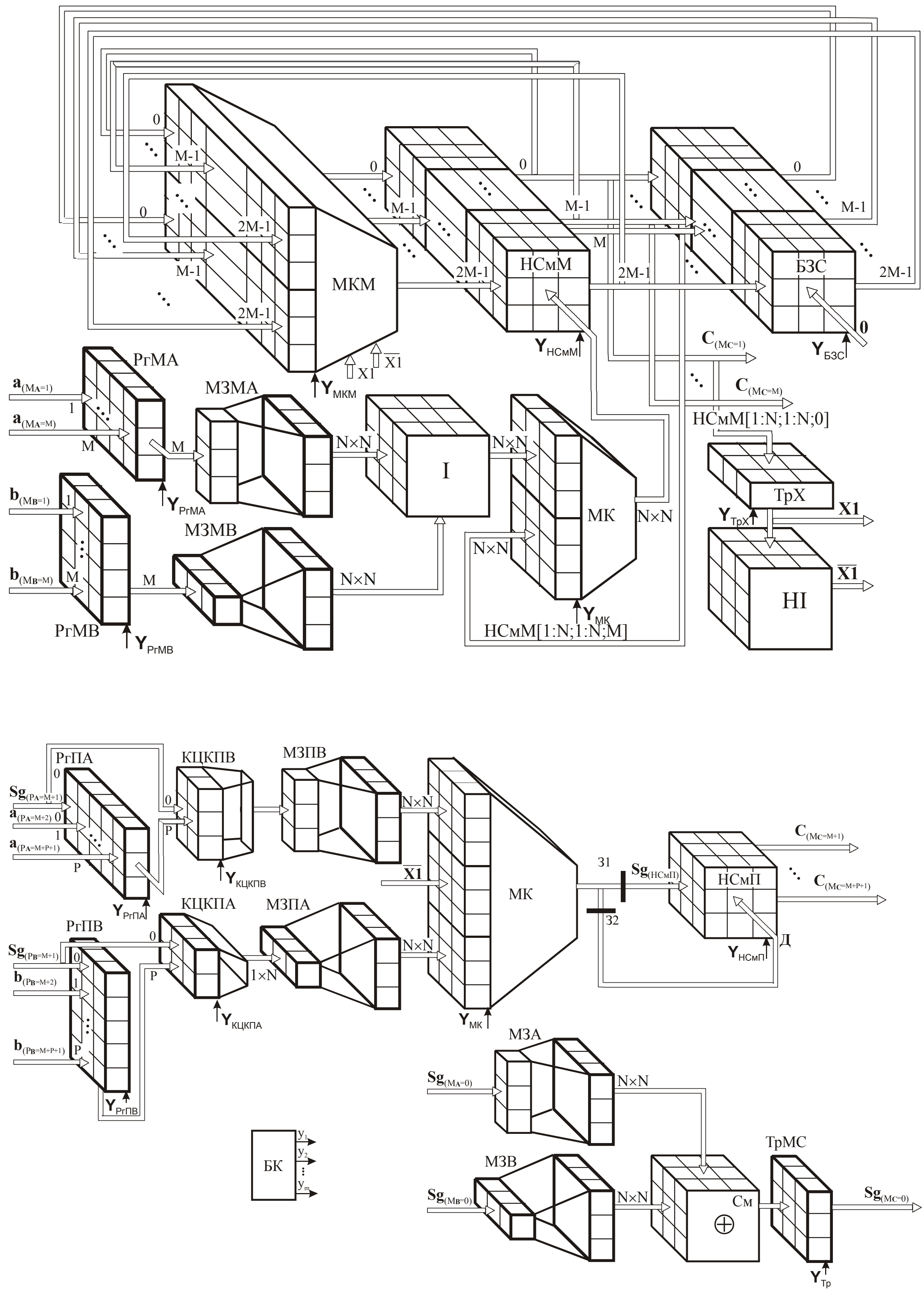


Рисунок 3.3 - Паралельний блок визначення векторного добутку векторів з плаваючою комою

Знаковий зріз результату формується за допомогою картинного суматора за модулем два, картинні входи якого зв’язані через мультиплікатори МЗА і МЗВ із знаковими зрізами порядків вхідних векторів. Отриманий знаковий зріз результату записується в картинний тригер ТрМС, вихід якого є знаковим виходом пристрою.

Слід відзначити, що суматор, який використовується в схемі блоку обробки порядків може бути виконаний як відомий суматор знакозмінних матриць, описаний в роботі [45].

При усуненні денормалізації мантиси робимо віднімання в необхідних елементах одиниці від матриці порядку, що буде зроблено за допомогою суматора НСмП подачею через комутатор специфічних операндів.

Матриця порядку результату на заключному етапі має бути переведена НСмП, вкінці повинна бути перетворена в прямий код.

Час T_{BMV}^{float} роботи блоку векторного добутку векторів визначаємо за такою формулою:

$$\mathrm{T}_{BMV}^{float} = 8\left(\mathrm{M}^2 + 3\mathrm{M} + 2\mathrm{P} + 3\right) \cdot \tau_{\text{лог}}. \tag{3.3}$$

3.5 Оцінювання апаратурних витрат на реалізацію спецпроцесора матричних операцій

Оцінювання апаратурних витрат для реалізації спецпроцесора визначається номенклатурою основних функціональних вузлів, представлених в таблиці 3.1.

Витрати в таблиці оцінено в умовних вузлах. Які є однотипними для всіх блоків спецпроцесора. Як зазначалось раніше. Таким умовним узлом слугує матричний логічний елемент І, який реалізується на світлоклапанному оптико-електронному пристрої типу «оптично керований транспарант» [18-20].

Таблиця 3.1 – Апаратурні витрати на реалізацію СП для визначення добутку- обернення матриць

Назва блоку	Кількість однотип. блоків	Число умов. вуз. для всіх однотип. бл.
Нагромаджу-валь-ний суматор NSm	1	45M+16P+156
Блок множення вектора на обернений коефіцієнт $Mult$	1	$\frac{42M+14P+144}{N}$
Блок векторного добутку векторів BMV	1	$\frac{9M+9P+21}{N}+12M+4P+110$
Регістри RGA, RGB	1	4(M+P+2)
Блок формування вектора BFV	1	$\frac{M+P+2}{2}+\frac{6(M+P+2)}{N}$
Мультиплексор MUX	1	3(M+P+2)
Мультиплексор $MUX1$	1	M+P+2
Мультиплексори $MUX3, MUX4$	1	(M+P+2)/N
Мультиплексор $MUX2$	1	(M+P+2)/2N
Схема AND	2(M+3+2)	2(M+P+2)
Фільтр $Filtr$	1	1
Разом		$\frac{60M+32P+183}{N}+67{,}5M+30{,}5P+287$

Так, наприклад, паралельні оптоелектронні регістри зсуву PrA і PrB в структурі СП забезпечують паралельний запис, збереження, паралельний зсув вліво та вгору на один елемент та паралельну видачу матричної інформації за РЗ.

Схему класичного регістру зсуву можна реалізувати у вигляді сукупності D-тригерів (рисунок 3.4), кількість яких залежить від розрядності S вхідних даних, що визначається із співвідношення S=M+P.

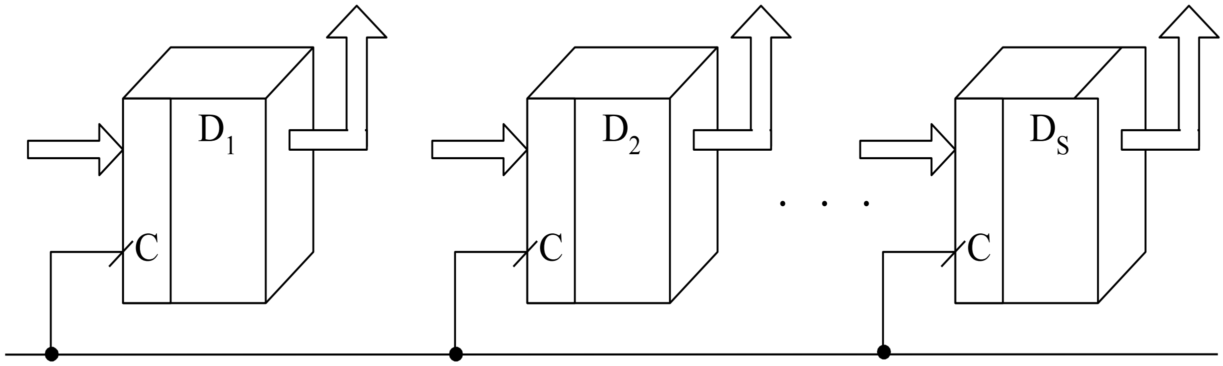


Рисунок 3.4 – Структурна схема картинного регістру зсуву, на D-тригерах

Крім того, картинні операції запису та збереження матричної інформації у розрядно-зрізовому вигляді можна виконувати на картинному двохтактному картинному тригері, реалізація якого відома із [47] (рисунок 3.5).

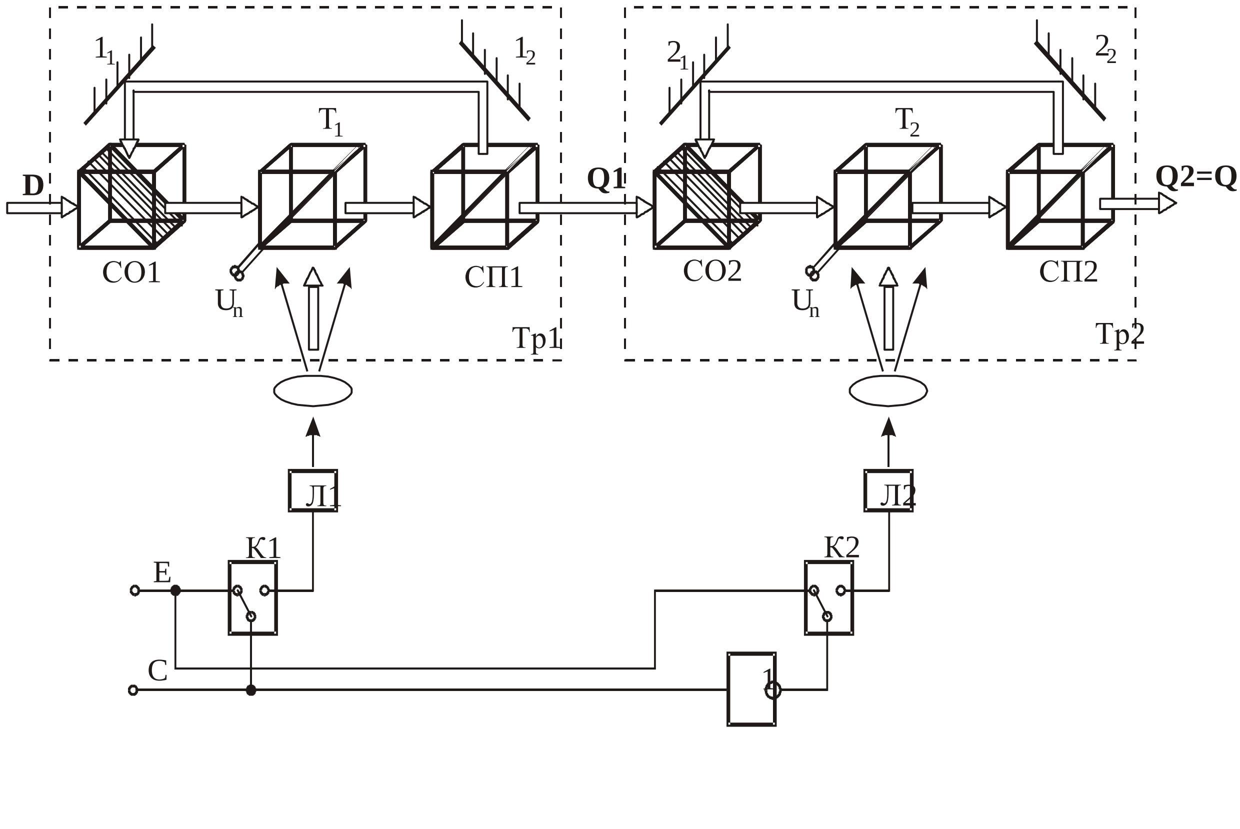


Рисунок 3.5 – Паралельний D-тригер з MS-структурою [47, 50]

Оптоелектронна схема, що наведена на рис. 3.5, містить світлооб’єднувачі СО1, СЩ2. оптично-керовані транспаранти T_1, T_2 , світлоподілювачі СП1, СП2, дзеркала 1_1 і 1_2, дзеркала 2_1 і 2_2, лазери Л1 та Л2, ключі К1, К2, які під'єднано до синхровходу С та джерела електроенергії Е.

При цьому для ОКТ, що використовуються в наведеній схемі, важливо виконання умов [50]:

$$\tau_{\text{вкл.}} < \tau_{\text{викл}}., \tag{3.4}$$

де $\tau_{вкл.}$ - час включення ОКТ; $\tau_{викл.}$ - час виключення ОКТ.

Робота картинного MS-триггера описується так [47, 50]:

$$\begin{gathered}\mathbf{Q}2^{(t+2)} = \mathbf{Q}1^{(t+1)} \wedge \overline{C}^{(t+1)}, \quad \mathbf{Q}1^{(t+1)} = \mathbf{D}^{(t)} \wedge C^{(t)}, \\ \mathbf{Q}2^{(t+2)} = \mathbf{D}^{(t)} \wedge C^{(t)} \wedge \overline{C}^{(t+1)}.\end{gathered} \tag{3.5}$$

Таким чином, для паралельного збереження числових матриць, поданих S зрізами, доцільно використовувати просторові тривимірні регістри, побудовані на основі S картинних D-тригерів, які дозволяють також паралельно записувати та зчитувати матрицю за РЗ.

Картинні цифрові комутатори, які входять до структури СП, можуть бути реалізовані різними способами. Один із традиційних варіантів реалізації КЦК, наведений на рисунку 3.6, представляє собою набір із S оптоелектронних затворів 1_1, 1_2, ..., 1_S, керуючі електроди яких являються відповідними керуючими входами КЦК, світлооб'єднувача 2 та дешифратора 3.

Даний КЦК здійснює комутацію на паралельний оптичний вихід одного із S бінарних РЗ, які подаються на відповідні S інформаційні картинні оптичні входи 4_1, 4_2, ..., 4_S. Вибір необхідного бінарного РЗ здійснюється в залежності від керуючих сигналів.

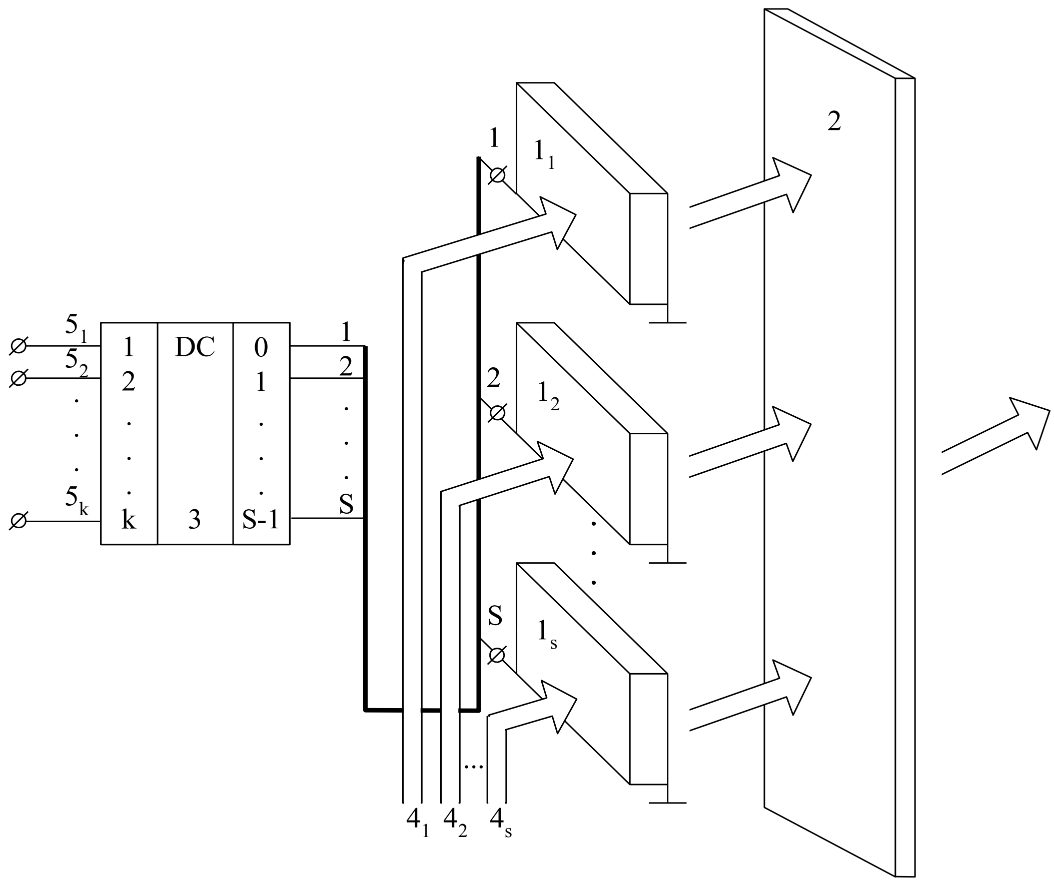


Рисунок 3.6 – Структурна схема картинного цифрового комутатора

В даному випадку реалізації комутатора позначене через k число керуючих входів КЦК обирається рівним числу S інформаційних картинних оптичних входів КЦК. А для зменшення числа керуючих входів КЦК з числа S до числа k використовується дешифратор 38 з числом адресних входів k таким, що $2^k = S$. Таке технічне рішення КЦК описане в роботі [43].

КЦК можна також реалізувати на базі структури оптоелектронного комутатора гігабітних потоків інформації, описаного в [48].

3.6 Оптико-електронні базові елементи для спецпроцесора матричних операцій

3.6.1 Джерела випромінювання

Серед джерел випромінювання для реалізації оптико-електронного спецобчислювача найкраще підходять VCSEL лазера завдяки можливості створювати малі за розмірами VCSEL матриці.

VCSEL лазери (рис. 3.7) є одним із видів напівпровідникових лазерних діодів, що випромінює лазерний промень перпендикулярно до верхньої поверхні, на відміну від звичайних напівпровідникових лазерів, які випромінюють від поверхні, яка формується сколюванням окремих чіпів.

Лазерний резонатор містить два Брегівських дзеркала (DBR) паралельні поверхні пластинам активної області, які складаються з однієї або декількох квантових ям. DBR-дзеркала складаються з чергування шарів з високими і низькими показниками заломлення. Кожен шар має товщину від чверті довжини хвилі лазера в матеріалі та має коєфіцієн відбиття вище 99% [49].

В більшості VCSELs верхнє та нижнє дзеркало легують матеріалами p та n типів, формуючи діодний перехід. В більш складних структурах шари p та n типів можуть знаходитись між дзеркалами. Така структура дозволяє зменшити електричні втрати.

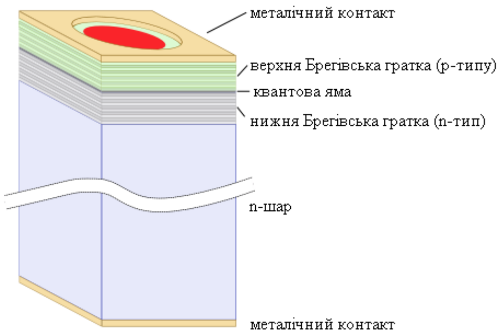


Рисунок 3.7 - Проста VCSEL структура

Накачка активного шару VCSELs може бути здійснена зовнішнім джерелом світла з меншою довжиною хвилі, зазвичай – іншим лазером (рис.3.8).

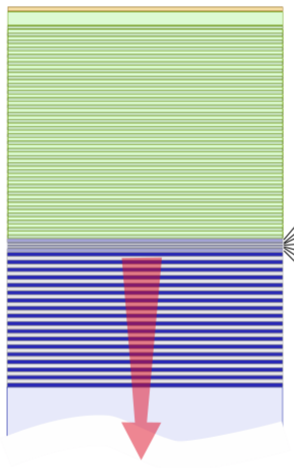


Рисунок 3.8 - Структура VCSEL лазера

VCSELs, що випромінюють на довжинах хвиль від 650 nm до 1300 nm зазвичаю формуються на GaAs підкладинці з брегівськими дзеркалами на основі AlxGa(1-x)As. Система GaAs–AlGaAs має переваги завдяки для конструювання VCSELs лазерів завдяки майже однаковій сталій гратки матеріалу. Це дозволяє уникнути зсувів шарів, які вирощені на GaAs підкладинці. До того ж коефіцієнт заломлення AlGaAs не сильно змінюється при збільшенні частини Al.

У роботі [49] наведена матриця з 10x10 лазерів з незалежною адресацією. Усі 100 елементів масиву зв’язувалия з 25 контактними площадками уздовж кожної зі сторін чіпа (розмір чіпа дорівнював 5x5 мм) за допомогою металізованих струмопровідних ліній шириною 30 мкм, що містилися на шарі Si, щоб уникнути струмів витоку. Відстань між лазерами у масиві складала 250 мкм при діаметрі лазерів 55 мкм, а розмір контактних площадок був рівним 120x120 мкм [49].

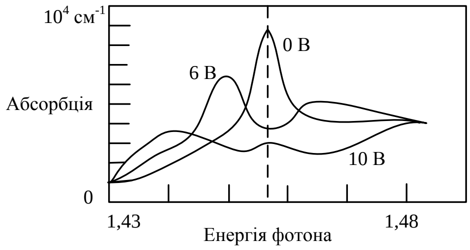
3.6.2 Оптично-керований транспарант як основний елемент оптико-електронного спецпроцесора для матричних операцій

Просторово-часові модулятори світла здійснюють модуляцію світлового потоку модулюючим середовищем у відповідності до вхідних сигналів. Якщо керують оптичним полем, то маємо оптично керовані транспаранти (ОКТ) , якщо електричним полем, то електрично керовані транспаранти (ЕКТ). Саме ОКТ використовуються як основний елемент основних блоків та вузлів розробленого СП, їх питома вага в загальному обладнанні для реалізації СП становить 98% .

Враховуючи відомі переваги використаємо в якості ОКТ SEED-пристрій, в якому застосовано ефект квантово-розмірних матеріалів при опроміненні їх світловим потоком [20].

Класична структура QCSE модуляторів, що використовується у всіх SEED, є p-i-n діодом, як показано на рис. 3.6 [51].

Просторовий модулятор світла із самонаведеним електрооптичним ефектом типу SEED побудований на тому, що багатошарова квантово-розмірна структура розміщується всередині збідненої області фотодетектора. При цьому застосовується нелінійний ефект Штарка, в результаті чого змінюються оптичні властивості шарів при дії на них перпендикулярно спрямованого електричного поля. Так виникає різкий стрибок оптичного поглинання біля енергії забороненої зони, який зміщується убік низьких енергій, не руйнуючи при цьому гострого піка екситонового поглинання. Вмикаючи SEED в зовнішній ланцюг, виникає оптоелектронний зворотній зв'язок. При додатному зворотному зв'язку зовнішній ланцюг містить навантажувальний резистор. Сильне опромінення останнього обумовлює великий фотострум в SEED, що спричиняє велике падіння напруги на навантаженні. При цьому зменшується електричне поле на SEED-структурі, що у свою чергу збільшує поглинання. Поглинання, що зросло, збільшує фотострум, до тих пір, поки пристрій не стане зовсім непрозорим. Достатньо малої енергії, щоб створити для SEED непрозорий стан.



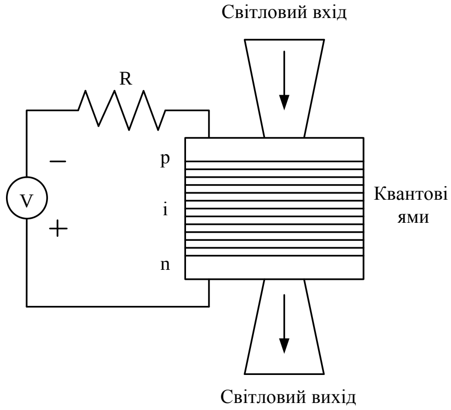


Рисунок 3.9 - Класична структура QCSE модуляторів, що використовується у всіх SEED

На сьогоднішній день існують такі види SEED [52]:

1. R-SEED – SEED на резисторах;
2. D-SEED – SEED на діодах;
3. T-SEED – SEED на транзисторах;
4. F-SEED – SEED на транзисторах з ефектом поля;
5. M-SEED – багатостанові SEED;
6. S-SEED – симетричні SEED;
7. AW-SEED – асиметричні SEED.

Базовим елементом для розробленої архітектури спецпроцесора матричних операцій пропонується використання S-SEED – симетричного бістабільного елемента.

Структурна схема S-SEED представлена на рис. 3.10.

Переваги S-SEED як бістабільного елемента, перш за все, в тому, що вони що не потребують використання інтегрованих транзисторів, як наприклад, як D-SEED та R-SEED. Зазначений елемент є бістабільним по відношенню до двох потужностей пари світлових пучків, один з сигналів відповідає високому рівню (логічна “1”), інший низькому (логічний “0”). Ця незвичайна властивість S-SEED дозволяє досягти високої точності і швидкодії.

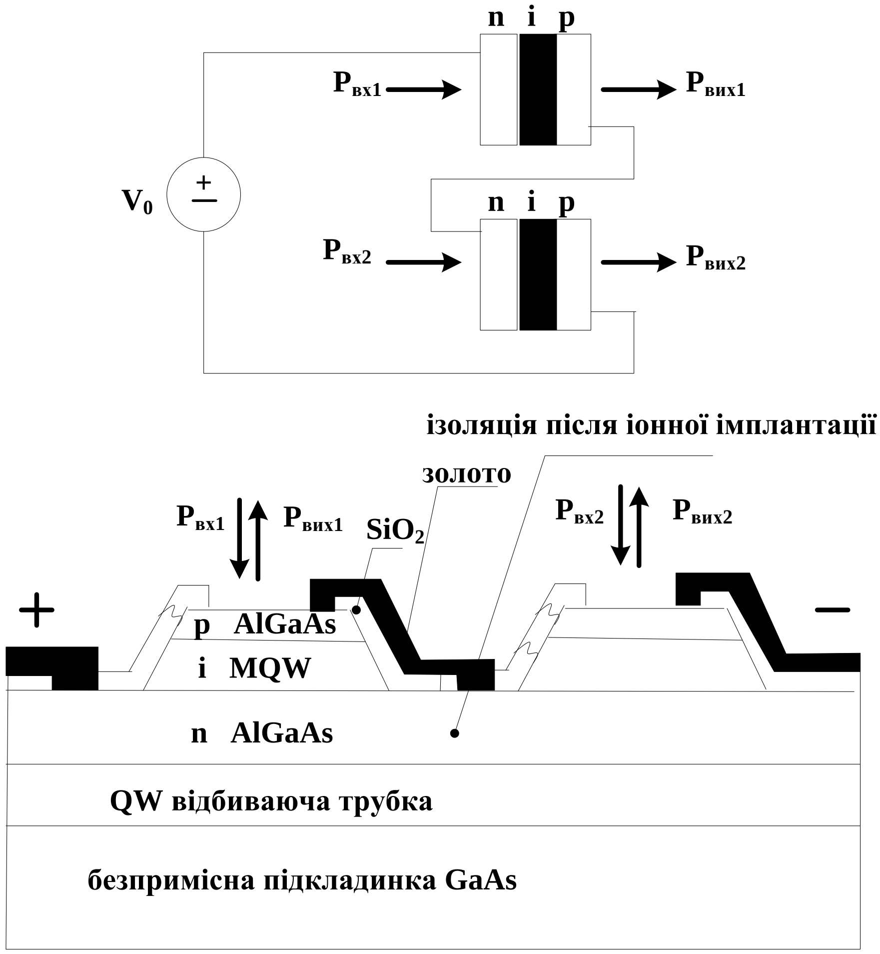


Рисунок 3.10 - Схема структурна S-SEED

Таким чином, S-SEED можна розглядати як елемент, що виконує функції простого логічного елемента, що відповідає всім вимогам базового елемента для побудови складніших цифрових комбінаційних схем. На його основі можна синтезовувати логічні операції НІ-АБО, НІ-І, та І із високою точністю та низькою споживчою енергією.

Відомі характеристики матриці з 32×64 із S-SEED у вигляді p-i-n-діодів, які мають площу 100 мкм^2, енергію спрацювання E_a=1 пДж та час перемикання $t_п$=1 нс. Створено експериментальні зразки матриць з 320х320 елементів з p-i-n-діодами площею 10 мкм^2, енергією спрацювання E_a=0,01 пДж та часом перемикання $t_п$=1 нс [51].

Як зазначено в розділі 1, компанія LensLet Ltd (Ізраїль) створила на SEED перший оптичний цифровий високошвидкісний оброблювач сигналів

EnLight 256 (рис. 1.2) [41]. Його швидкодія складає 8-Тера (10^{12}) операцій з фіксованою точкою за секунду. Основним структурним блоком EnLight 256 є: оптичний матричний помножувач, що за 8нс може помножити вектор із 256 8-бітних елементів на матрицю 256х256 8-бітних елементів [41].

Просторовий модулятор світла ABLAZE ™ 2D MQW [41] на квантових ямах, який застосовується в процесорі, має розміри 640х480 пікселів-модуляторів та швидкість перемикання одного пікселя 20 ГГц.

Отже, SEED може бути визначено в якості базового просторо-часового модулятора світла для реалізації розробленої архітектури спецпроцесора матричних операцій.

3.7 Розробка структурної схеми блоку керування для спецпроцесора для визначення добутку-обернення матриць

Блок керування для спецпроцесора, наведеного на рис. 2.2, генерує розподілену в часі послідовність керуючих сигналів, під дією якої функціонує даний СП. Так, за допомогою цих сигналів керування обираємо режим роботи СП – визначення добутку чи обернення матриць.

Відомо, що існує два принципи керування роботою обчислювальних систем: принцип “жорсткої” логіки та принцип програмованої логіки.

За першим принципом що для кожної команди СП будують набір логічних схем, які в потрібних тактах збуджують відповідні функціональні сигнали. За другим методом побудови логіки формування функціональних сигналів застосовується мікропрограмне керування.

Блоки керування з “жорсткою” логікою мають більш високу швидкодію, ніж з програмованою логікою. Універсальний характер, який дозволяє переналаштовувати пристрій на виконання іншої функції, є основною перевагою блоків керування із програмованою логікою.

Тому в магістерській роботі роботі розробляється блок керування з програмованою логікою. В автоматах, побудованих на основі ПЗП, задана мікропрограма реалізується в явній формі і зберігається в пам’яті у вигляді послідовності керуючих слів. Керуюче слово визначає порядок роботи пристрою протягом одного такту і називається мікрокомандою (МК).

Побудова керуючого автомата із програмованою логікою за існуючою граф-схемою алгоритму полягає в виборі способу адресації і формату мікрокоманди, складанні кодованої мікропрограми в вигляді таблиці і побудові структурної та принципової схеми автомату.

В порівнянні з реалізацією керуючого автомату на основі ПЗП з природною адресацією, керуючий автомат з примусовою адресацією для реалізації призводить до збільшення апаратурних витрат, але при цьому є виграш по швидкодії. Тому синтезуємо блок керування на основі автомата з примусовою адресацією.

Граф-схема алгоритму роботи блоку керування для СП паралельного множення-обернення матриць представлена на рисунку 3.11.

При цьому набір керуючих сигналів має наступний вигляд:

y_1: $\mathbf{A}_{\overline{(0:(S-1))}}[1:N;1:N]$.

y_2: $\mathbf{B}_{\overline{(0:(S-1))}}[1:N;1:N]$.

y_3: $\Pr A^{(t=0)}_{(0:(S-1))} = \mathbf{A}_{(0:(S-1))}$.

y_4: $\Pr B^{(t=0)}_{(0:(S-1))} = \mathbf{B}_{(0:(S-1))}$.

y_5: $\text{НСм}^{(t=0)}_{(0:(S-1))}[1:N;1:2N] = 0$.

y_6: t:=t+1.

y_7: $\text{КЦК3}^{(t)}_{(0:(S-1))}[1:N;1] = \Pr A^{(t)}_{(0:(S-1))}[1:N;1]$.

y_8: $\text{КЦК4}^{(t)}_{(0:(S-1))}[1;1:N] = \Pr B^{(t)}_{(0:(S-1))}[1;1:N]$.

y_9: $\text{БЗДВ}^{(t)}_{(0:(S-1))}[1:N;1:N] = \text{КЦК3}^{(t)}_{(0:(S-1))} \otimes \text{КЦК4}^{(t)}_{(0:(S-1))}$.

y_{10}: $\text{НСм}^{(t)}_{(0:(S-1))} = \text{НСм}^{(t-1)}_{(0:(S-1))} + \text{БЗДВ}^{(t)}_{(0:(S-1))}$.

у11: $\Pr A^{(t)}_{(0:(S-1))} = \varphi^{\leftarrow 1}(\Pr A^{(t\text{-}1)}_{(0:(S-1))})$.

у12: $\Pr B^{(t)}_{(0:(S-1))} = \varphi^{\uparrow 1}(\Pr B^{(t\text{-}1)}_{(0:(S-1))})$.

у13: $U_{\overline{(0:(S-1))}}[1:N;1:N] = НСм^{(t=N)}_{(0:(S-1))}[1:N;1:N]$.

у14: $\Pr B^{(t=0)}_{(0:(S-1))} = \mathbf{E}_{(0:(S-1))}$.

у15: $ЕП^{(t=0)} = 0$.

у16: $НСм^{(t=0)}_{(0:(S-1))}[1:N;1:N] = \Pr A^{(t=0)}_{(0:(S-1))}$.

у17: $НСм^{(t=0)}_{(0:(S-1))}[1:N;(N+1):2N] = \Pr B^{(t=0)}_{(0:(S-1))}$.

у18: $ЕП^{(t=0)}[1;1:(S-1)] = 1$.

у19: $ЕП^{(t=0)}[2:N;0] = 1$.

у20: $КЦК1^{(t)} = НСм^{(t\text{-}1)}_{(0:(S-1))}[t;1:2N]$.

у21: $КЦК2^{(t)} = КЦК1^{(t)}[0:S;t]$.

у22: $Под^{(t)}_{(0:(S-1))}[i] = \dfrac{КЦК1^{(t)}[0:S;i]}{КЦК2^{(t)}} КЦК1^{(t)},\ i = \overline{1:N}$.

у23: **Ф1**[1:(t-1);1:2N]:=1.

у24: **Ф2**[1:(t-1);0:(S-1)]:=1.

у25: $\mathbf{ЕП}^{(t)} := (\mathbf{ЕП}^{(t\text{-}1)})^{\downarrow 1(\mathbf{ЕП}[N;0:(S-1)])}$.

у26: **Ф1**[t;1:2N]:=0.

у27: **Ф1**[(t+1):N;1:2N]:=1.

у28: **Ф2**[t;0(S-1)]:=0.

у29: **Ф2**[(t+1):N;0:(S-1)]:=1.

у30: $БФВС[1:N;0:(S-1)] = (НСм^{(t\text{-}1)}_{(0:(S-1))}[1:N;t]) \wedge Ф2) \oplus ЕП^{(t)}$.

у31: $КЦК3^{(t)}_{(0:(S-1))}[1:N;1] = БФВС^{(t)}_{(0:(S-1))}[1:N;1]$.

у32: $КЦК4^{(t)}_{(0:(S-1))}[1;1:N] = Под^{(t)}_{(0:(S-1))}[1;1:N]$.

у33: $БЗДВ^{(t)}_{(0:(S-1))}[1:N;(N+1):2N] = КЦК3^{(t)}_{(0:(S-1))} \otimes Под^{(t)}_{(0:(S-1))}[1:(N+1):2N]$.

у34: $НСм^{(t)}_{(0:(S-1))} = НСм^{(t\text{-}1)}_{(i)} \wedge Ф1[1:N;1:2N],\ i = \overline{0:(S-1)}$.

y_{35}: $\mathbf{X}_{\overline{(0:(S-1))}}[1:N;1:N] = НСм_{(0:(S-1))}^{(t=N)}[1:N;(N+1):2N]$.

Логічні умови:

X1: Множення/обернення,

X2: t=N,

X3: t=1.

Здійснимо розбивку сигналів y_i по полям. Число полів визначається найбільшою кількістю вихідних сигналів, записаних в одній операційній вершині. В даному випадку їх чотири, тому для операційної частини буде чотири поля: Y1, Y2, Y3, Y4. В цьому випадку, мікрокоманда буде мати структуру, показану на рисунку 3.12.

В свою чергу, закодуємо умовні вершини наступним чином: X1 як 01, X2 як 10 і X3 як 11.

Формат МК при примусовій адресації буде мати два адресних поля A0 і A1 (див. рисунок 3.11). Тоді адресу наступної МК визначається в залежності від значення умови X, яка перевіряється в даному такті, наступним чином: в якості адреси використовуємо вміст поля A0, якщо X=0, і поля A1, якщо X=1. При цьому, безумовні переходи здійснюються за адресою A0.

Розбивка сигналів по полям наведена в таблиці 3.2.

Таблиця 3.2 – Розбивка сигналів по полям

Код	Y1	Y2	Y3	Y4
0001	y_1	y_2	y_3	y_4
0010	y_5	y_6	y_7	y_8
0011	y_9	y_{10}	y_{11}	y_{12}
0100	y_{13}	y_{14}	y_{16}	y_{15}
0101	y_{17}	y_{18}	y_{20}	y_{19}
0110	y_{21}	y_{22}	y_{23}	y_{24}
0111	y_{25}	y_{26}	y_{27}	y_{28}
1000	y_{29}	y_{30}	y_{31}	y_{32}
1001	y_{34}	y_{33}	y_{35}	**F**

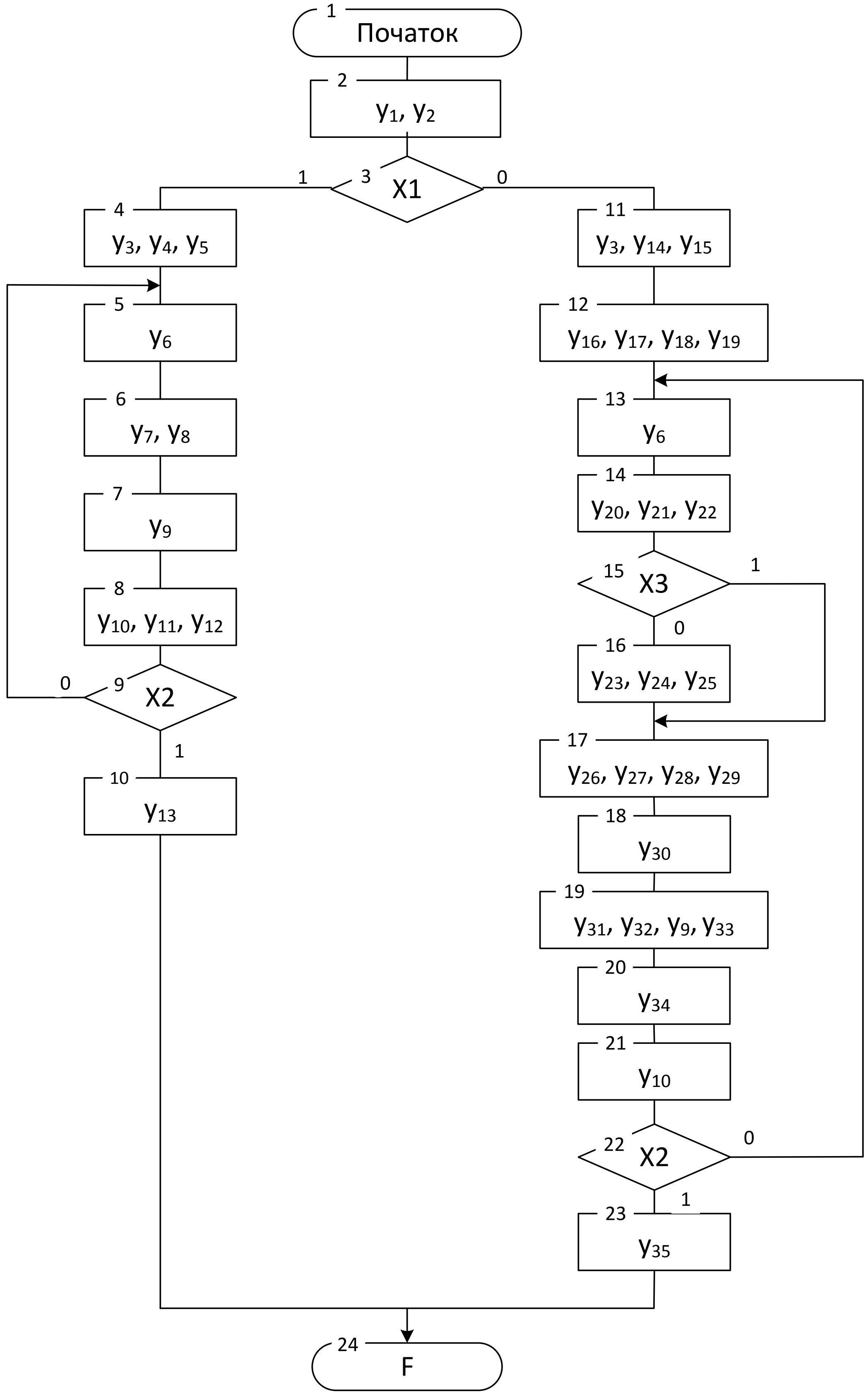


Рисунок 3.11– Граф-схема алгоритму роботи блоку керування для СП паралельного множення-обернення матриць

Y1	Y2	Y3	Y4	X	A0	A1
Операційна частина				Адресна частина		

Рисунок 3.12 – Структура керуючого слова

В цьому випадку, прошивка ПЗП буде мати вигляд, представлений в таблиці 3.3. Тоді формат МК буде мати вигляд, наведений на рисунку 3.13.

Таким чином, ємність ПЗП становить 20 слів по 28 біт, що складає 560 біт, згідно прошивці ПЗП (див. таблицю3.4).

Y1	Y2	Y3	Y4	X	A0	A1
1 4	5 8	9 12	13 16	17 18	19 23	24 28

Рисунок 3.13 – Формат мікрокоманди

Таблиця 3.3 – Прошивка ПЗП

Адреса	Y1	Y2	Y3	Y4	X	A0	A1
00000	0001	0001				00001	00001
00001					01	01000	00010
00010	0010		0001	0001		00011	00011
00011		0010				00100	00100
00100			0010	0010		00101	00101
00101	0011					00110	00110
00110		0011	0011	0011	10	00011	00111
00111	0100					10011	10011
01000		0100	0001	0100		01001	01001
01001	0101	0101	0100	0101		01010	01010
01010		0010				01011	01011
01011	0110	0110	0110		11	01100	01100
01100	0111	0111		0111		01101	01101
01101	1000	1000	0111	0111		01110	01110
01110			1000			01111	01111
01111	0011	1001	1000	1000		10000	10000
10000	1001					10001	10001
10001		0011			10	01010	10010
10010			1001			10011	10011
10011				1001(F)		00000	00000

На рисунку 3.14 показана схема блоку керування, яка містить RS тригер, дешифратори ДШY1- ДШY4, ДШX, логічні елементи І, АБО, ПЗП

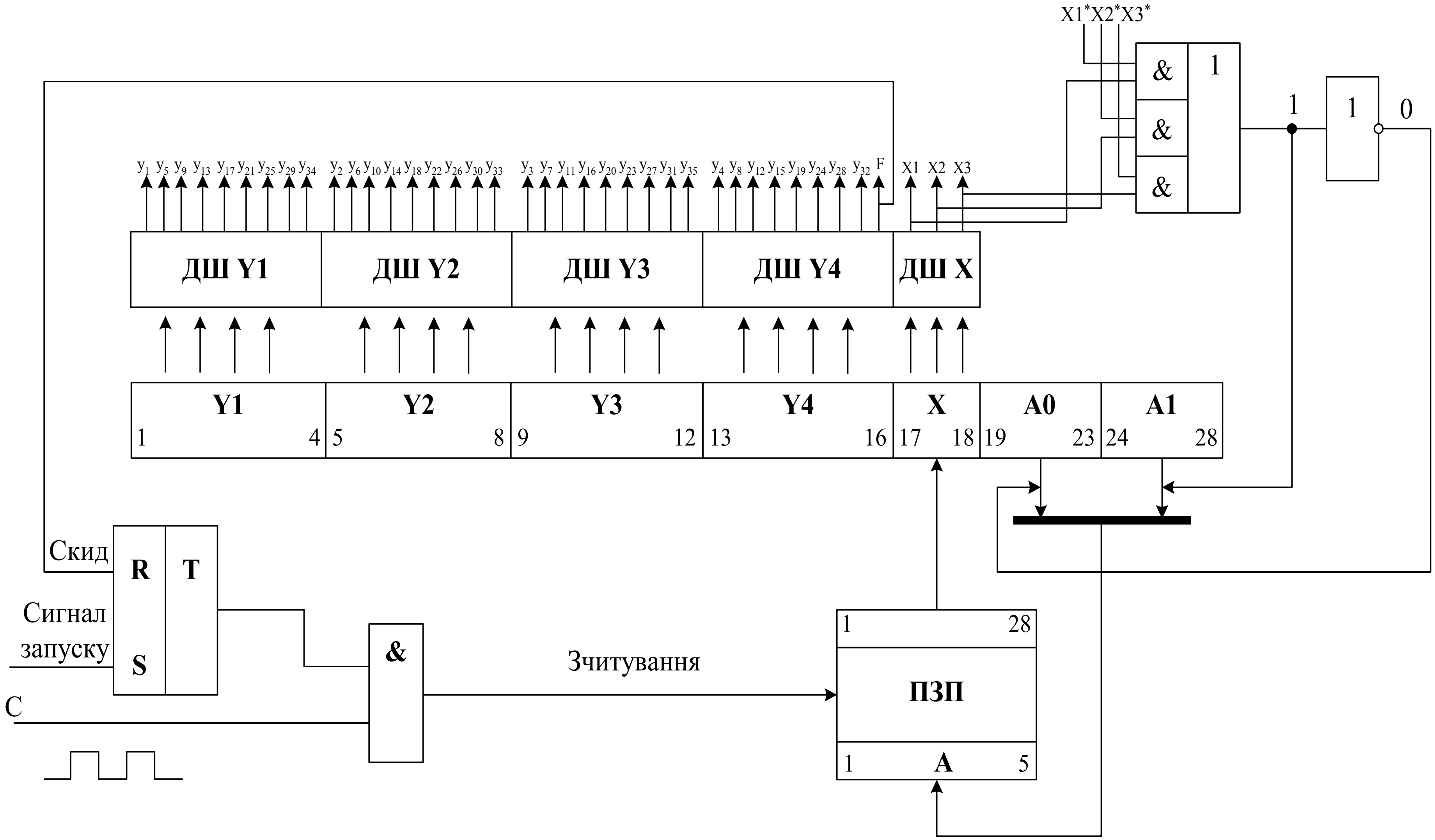


Рисунок 3.14 – Схема блоку керування

3.8 Оцінювання основних характеристик розробленого СП

Однією із основних характеристик розробленого СП є час його роботи відповідно в режимі визначення добутку матриць та в режимі обернення обернення матриць.

У вигляді формул (2.17) і (2.18) було оцінено час множення та обернення в залежності від часу обробки одного розрядного зрізу..

Оскільки в якості базових елементів при побудові СП для визначення добутку-обернення матриць використовуються оптично-керовані транспаранти то у вигляді SEED-приладів, для яких $\tau_{лог} = 10^{-9}$ с., то при параметрах розмірності матриць $N = 320$, $M = 48, P = 16$ розрахуємо час роботи СП у відповідних режимах за формулами (2.17), (2.18).

В результаті отримуємо:

$$T_{добут.м.} = \Delta T_{РЗ} \times N \times \left(M^2 + M \cdot P + 7M + 2P + 17\right) = 0,09(c)$$

$$T_{оберн.м.} = \Delta T_{РЗ} \times N \times \left(2M^2 + MP + 14M + 8P + 28\right) = 0,016(c)$$

Ще одним важливим показником для оцінки розробленого СП є його швидкодія, яка може бути оцінена за такими формулами:

$$V^{множ.} = \frac{2 \cdot N^2}{8N \cdot \left(M^2 + M \cdot P + 7M + 2P + 17\right) \cdot \tau_{лог.}}, \tag{3.6}$$

$$V^{оберн.} = \frac{\dfrac{2N \cdot \left(M^2 + 7M + 2P + 12\right) + 2N^2 \cdot \left(M^2 + 3M + 2P + 3\right)}{4M + M \cdot P + 4P + 11} + 2N^2}{8 \cdot \left(2M^2 + M \cdot P + 14M + 8P + 28\right) \cdot \tau_{лог.}}. \tag{3.7}$$

Підставивши конкретні дані, згідно технічного завдання, отримаємо наступні значення швидкодії розробленого СП для режиму множення в режиму обернення відповідно:

$$V^{\text{множ.}} = \frac{2 \cdot 320^2}{8 \cdot 320 \cdot \left(48^2 + 48 \cdot 16 + 7 \cdot 48 + 2 \cdot 16 + 17\right) \cdot 10^{-9}} = 2{,}3 \cdot 10^7 \text{ (опер./с).}$$

$$V^{\text{оберн.}} = \frac{\dfrac{2 \cdot 320 \cdot \left(48^2 + 7 \cdot 48 + 2 \cdot 16 + 12\right) + 2 \cdot 320^2 \cdot \left(48^2 + 3 \cdot 48 + 2 \cdot 16 + 3\right)}{4 \cdot 48 + 48 \cdot 16 + 4 \cdot 16 + 11} + 2 \cdot 320^{10}}{8 \cdot \left(2 \cdot 48^2 + 48 \cdot 16 + 14 \cdot 48 + 8 \cdot 16 + 28\right) \cdot 10^{-9}} =$$

$$= 1{,}4 \cdot 10^{10} \text{ (опер./с).}$$

Дослідимо як впливає зміна кожного із параметрів (N, M, P) вхідних даних на швидкодію СП в цілому.

Так, наприклад, на рисунку 3.15 показані залежності швидкодії СП при різних значеннях розмірності вхідних матриць для режиму визначення добутку матриць та режиму обернення відповідно. Аналізуючи дані залежності, можна зробити висновок про те, що швидкодія СП суттєво зростає при збільшенні розмірності вхідних матриць. Це, в свою чергу, означає, що даний СП доцільно використовувати саме для обробки великорозмірних інформаційних масивів даних.

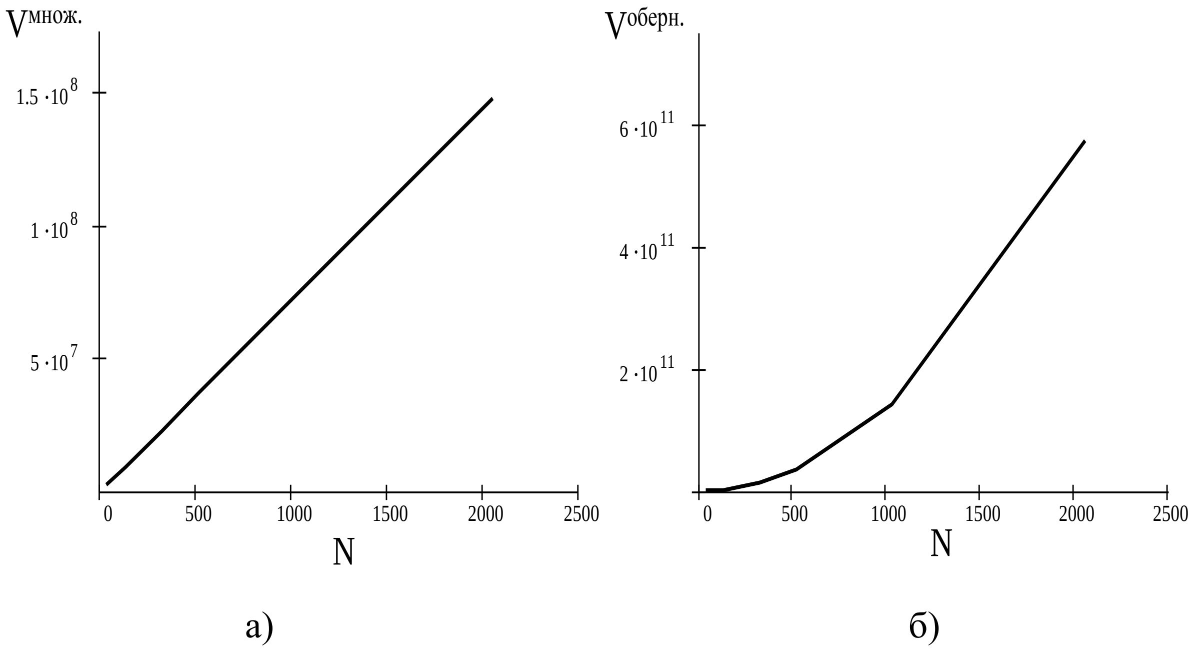


Рисунок 3.15 – Залежність швидкодії СП, що працює в режимі множення (а) та в режимі обернення (б) від розмірності вхідних матриць

На рисунку 3.16 наведені залежності швидкодії СП, що працює відповідно в режимі множення та обернення, від значення порядку вхідних даних.

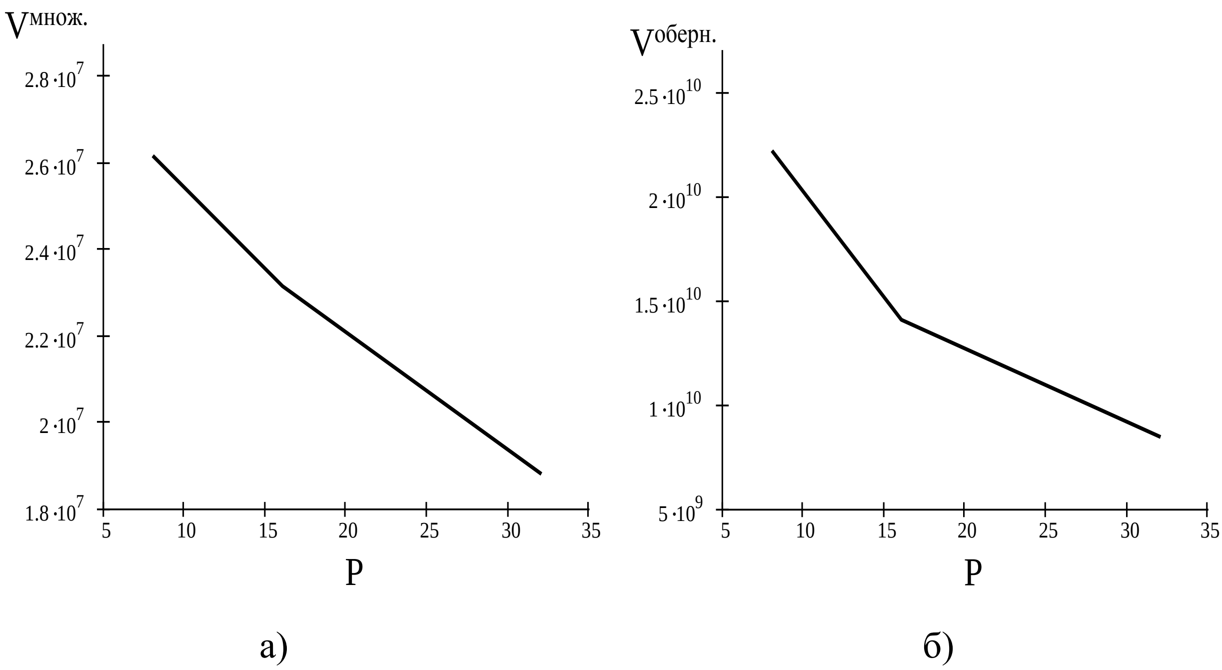
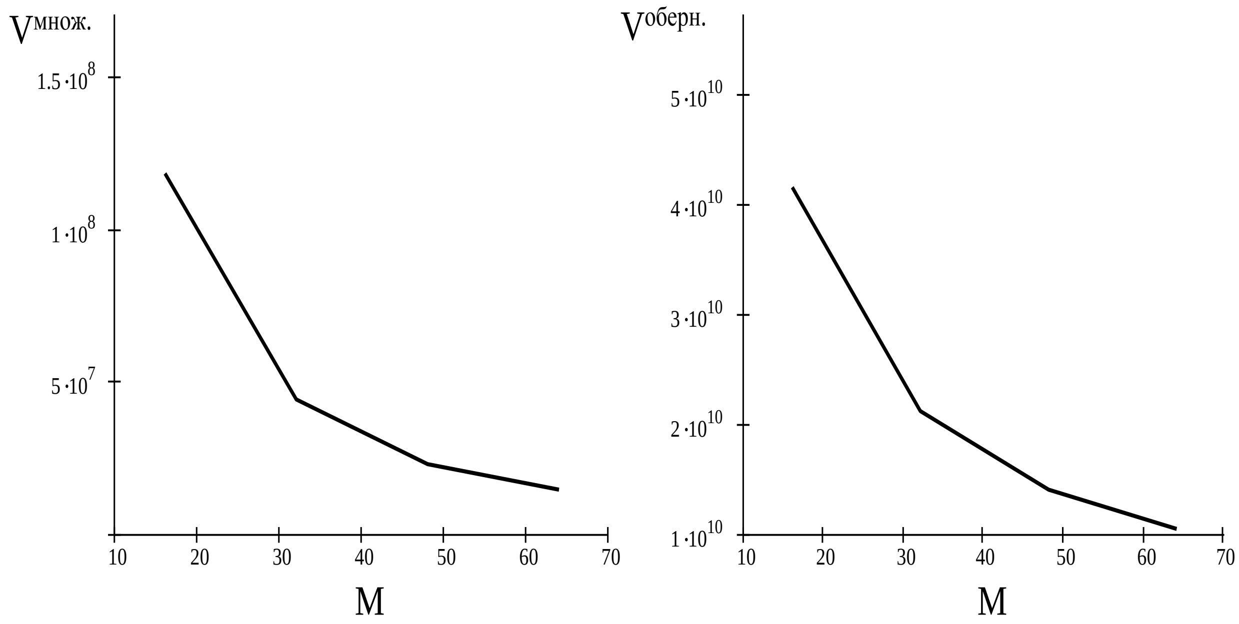


Рисунок 3.16 – Залежність швидкодії СП, що працює в режимі множення (а) та в режимі обернення (б) від порядку вхідних даних

Як видно із залежностей, представлених на рисунку 3.16, швидкодія СП, працюючого в режимі множення і в режимі обернення, дещо зменшується при значному зростанню значення порядку. Однак це зменшення швидкодії досить незначне – в межах одного порядку.

На рисунку 3.17 представлені залежності швидкодії СП від розміру мантиси вхідних даних для режиму множення та режиму обернення.

Аналізуючи графічні залежності, наведені на рисунку 3.17, можна відмітити, що значення мантиси, як значення порядку, досить мало впливає на швидкодію СП, незалежно від режиму його спрацювання (множення чи обернення матриць).



а) б)

Рисунок 3.17 – Залежність швидкодії СП,
що працює в режимі множення, від мантиси вхідних даних

Таким чином, узагальнюючи усі наведені графічні залежності, можна зробити висновок про те, що даний СП характеризується дуже високими значеннями швидкодії, яка зростає зі збільшенням розмірності матриць, і майже не залежить від величини мантиси і порядку вхідних даних.

Висновки до розділу 3

1. Розроблено новий паралельний алгоритм множення-обернення матриць на основі зовнішнього добутку векторів, який відрізняється від відомих покращенням часових характеристик, підвищеною багатофункціональністю та розширенням діапазону і точності даних, що обробляються. Це досягається за рахунок застосування прийомів оптичних цифрових обчислень і використання форми подання чисел у матрицях з плаваючою комою. Запропонований алгоритм виконує як множення, так і обернення матриць за N тактів, в той час, як найкращі окремі спецобчислювачі реалізують свої алгоритми за N^2 тактів.

2. Розроблено новий алгоритм перемноження матриць у формі з плаваючою комою на основі паралельного алгоритму множення знакозмінних цілочисельних матриць, оцінено його часові характеристики, що кращі від відомих аналогів.

3. Показано, що для адекватного відображення розроблених паралельних алгоритмів на паралельну структуру обчислювального СП необхідно враховувати такі особливості організації структури: матричний тип зв'язку між ПЕ; паралельне введення інформації за допомогою аналого-цифрового перетворювача картинного типу; наявність оптичних локальних і глобальних зв'язків.

4. Розроблено два нових варіанти структурної схеми паралельного СП для множення-обернення матриць: варіант на основі обчислення зовнішнього добутку векторів; варіант на основі перемножувача цілочисельних знакозмінних матриць, оцінено їх часові та апаратні характеристики. Обидва варіанти відрізняються від відомих покращеними часовими характеристиками та підвищеною точністю за рахунок використання форми подання чисел з плаваючою комою.

5. На основі співставлення економічних та технічних показників двох розроблених схем СП обрано варіант схеми на основі обчислення ЗДВ.

6. Розкриті основні блоки СП для паралельного множення-обернення матриць на основі обчислення ЗДВ та оцінені їх часові характеристики.

7. Оцінено основні характеристики розробленого СП. Проведено дослідження залежності швидкодії СП від параметрів вхідних даних.

4 ЕКОНОМІЧНИЙ РОЗДІЛ

4.1 Технологічний аудит розробленої архітектурної організації паралельного оптико-електронного спецпроцесора

Для оцінки комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, було здійснено незалежне експертне опитування.

Критерії оцінювання були взяті з таблиці рекомендованих критеріїв оціню-вання комерційного потенціалу розробки та їх можливої бальної оцінки.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальної шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтве-рджена експерт-ними висновками	Концепція підтверджена розрахунками	Концепція пе-ревірена на практиці	Перевірено ро-ботоздат-ність продукту в реа-льних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ри-нку	Мало аналогів на малому ринку	Кілька анало-гів на вели-кому ринку	Один аналог на великому ри-нку	Продукт не має аналогів на ве-ликому ринку
3	Ціна продукту значно вища за ціни анало-гів	Ціна продукту дещо вища за ціни аналогів	Ціна проду-кту прибли-зно дорів-нює цінам аналогів	Ціна проду-кту дещо ни-жче за ціни аналогів	Ціна проду-кту значно нижче за ціни аналогів
4	Технічні та споживчі вла-стивості про-дукту значно гірші, ніж в аналогів	Технічні та спо-живчі властиво-сті продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні анало-гів	Технічні та споживчі вла-стивості про-дукту трохи краще, ніж в аналогів	Технічні та споживчі вла-стивості про-дукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в ана-логів	Експлуата-ційні витрати дещо вищі, ніж в аналогів	Експлуа-та-ційні ви-трати на рі-вні витрат аналогів	Експлуа-та-ційні витрати трохи нижчі, ніж в аналогів	Експлуата-ційні витрати значно нижчі, ніж в аналогів

Таблиця 4.1 – Продовження

Бали (за 5-ти бальної шкалою)					
Критерій	0	1	2	3	4
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів не має
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансуван-ня ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використову-ються у виробництві

Таблиця 4.1 – Продовження

Бали (за 5-ти бальної шкалою)					
Критерій	0	1	2	3	4
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

При проведенні технологічного аудиту було залучено 3 експерти, у нашому випадку це один із керівників розробки зі створення високоефективних обчислювальних засобів, побудованих на модуляторах світла: к.т.н., професор Лисенко Г.Л.; д.т.н., професор Мартинюк Т.Б. та к.т.н., доцент кафедри лазерної та оптоелектронної техніки Тарновський М.Г.

Оцінка комерційного потенціалу системи здійснювалася за 12-ма крите-ріями з подальшим занесенням результатів до таблиці 4.2.

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	Лисенко Г.Л.,	Мартинюк Т.Б.	Тарновський М.Г..
	Бали, виставлені експертами:		
1	2	2	2
2	2	3	3
3	1	2	1
4	4	4	4
5	4	4	4
6	2	1	2
7	4	3	4
8	3	4	3
9	2	2	2
10	2	1	2
11	4	4	3
12	3	2	3
Сума балів	$СБ_1$=33	$СБ_2$=32	$СБ_3$=33
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_{1}^{3} СБ_i}{3} = 32,6.$ (4.1)		

Середньоарифметична сума балів $СБ_{сер}$:

$$\overline{СБ} = \frac{33+32+33}{3} = \frac{98}{3} = 32,67,$$

де $СБ_i$ – сума балів, виставлених кожним експертом;

n – кількість експертів.

Оцінимо рівні комерційного потенціалу розробки згідно критеріїв, наведених в таблиці 4.3.

Таблиця 4.3 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11–20	Нижче середнього
21–30	Середній
31–40	Вище середнього
41–48	Високий

Відповідно до результатів оцінювання можна зробити наступні висновки:

- розрахована на основі висновків експертів середньоарифметична сума балів $\overline{СБ}$=32,6, що відповідає рівню комерційного потенціалу розробки «вище середнього»;

- загальна якість розробки знаходиться на досить високому рівні, їй притаманна наукова та практична новизна.

Наша розробка є дорожчою у реалізації порівняно із аналогами за рахунок розширення функціональних можливостей при збереженні високої швидкодії шляхом застосування удосконаленого метода обчислень, реалізація якого вимагає залучення додаткової апаратури.

4.2 Прогнозування витрат на розробку архітектерної організації паралельного оптико-електронного спецпроцесора

Витрати на основну заробітну плату розробників (дослідників) розраховують за формулою:

$$З_о = \frac{M}{T_р} \cdot t \text{ грн} \quad (4.2)$$

де M – місячний посадовий оклад конкретного розробника, грн;

$T_р$ – середнє число робочих днів в місяці (приймемо 22 дні).

t – число днів роботи конкретного розробника (дослідника).

Розрахунок витрат на заробітну плату розробника подано в таблиці 4.4

Таблиця 4.4 – Витрати на заробітну плату розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів (годин) роботи	Витрати на заробітну плату, грн
Керівник магістерської роботи	12000	545,45	25 годин	≈ 2273
Магістрант Інженер	7000	318,18	70 днів	≈ 22273
Всього				24546

Витрати на основну заробітну плату робітників, що були задіяні у виготовленні дослідного зразка $З_{роб}$, розраховуються формулою 4.3.

$$З_{роб} = \sum_{1}^{n} ТР_i \cdot С_i \text{ грн,} \tag{4.3}$$

де n - число робіт за видами та розрядами;

$ТР_i$ – трудомісткість виконання конкретної роботи, годин;

$С_i$ – погодинна тарифна ставка робітника відповідного розряду, який виконує дану роботу, визначається за формулою 4.4:

$$С_i = \frac{М \cdot К_i}{Т_р \cdot Т_{зм} \cdot К_{вн}} \text{ грн/годин,} \tag{4.4}$$

де М – розмір мінімальної заробітної плати за місяць, грн;

В 2019 році М = 4173 грн;

$К_i$ – тарифний коефіцієнт робітника відповідного розряду;

$Т_р$ – середнє число робочих днів в місяці (приймемо 22 дні);

$Т_{зм}$ – тривалість зміни, годин; приймемо $Т_{зм}$ = 8 годин;

$К_{вн}$ –коефіцієнт виконання встановлених норм часу; приймемо, що $К_{вн}$ =1.

Для нашого випадку тарифна ставка робітника 1-го розряду становитиме:

$$С_і = \frac{4173 \cdot 1}{22 \cdot 8 \cdot 1} = 23,71 \text{ грн/годин.}$$

Витрати на основну заробітну плату робітників, що були задіяні у виготовленні дослідного зразка $З_{роб}$, зведено в таблицю 4.5.

Таблиця 4.5 – Величина витрат на основну заробітну плату робітників

Найменування технологічних операцій	Трудомісткість, год.	Розряд роботи та тарифний коефіцієнт	Погодинна тарифна ставка, грн/годину	Величина оплати, грн
Заготівельні	5	2 (1,17)	27,74	≈ 139
Слюсарно-збиральні	6	4 (1,45)	34,38	≈ 206
Виготовлення друкованої плати	7	4 (1,45)	34,38	≈ 241
Монтажні	10	4 (1,45)	34,38	≈ 349
Налагоджувальні	11	5 (1,65)	39,12	≈ 430
Всього				1365

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які брали участь у НДДКР розраховується як 10…12 % від основної заробітної плати розробників, тобто

$$З_д = (0,1...0,12) \cdot (З_о + З_{роб}). \tag{4.5}$$

Для нашого випадку отримаємо:

$$З_д = 0{,}111 \times (24546 + 1365) \approx 2876 \text{ грн.}$$

Нарахування на заробітну плату $НР_{зп}$ розробників та робітників розраховуються за формулою (4.6):

$$Н_{зп} = (З_о + З_{роб} + З_д) \cdot \frac{\beta}{100}, \qquad (4.6)$$

де β = 22% – ставка єдиного внеску на загальнообов’язкове державне соціальне страхування.

Для нашого випадку отримаємо:

$$НР_{зп} = (24546 + 1365 + 2876) \times 0{,}22 \approx 6333 \text{ грн.}$$

Амортизація А основних засобів, обладнання, комп’ютерів тощо розраховується за формулою (4.7):

$$А = \frac{Ц \cdot Н_а}{100} \cdot \frac{Т}{12} \text{ грн}, \qquad (4.7)$$

де Ц – загальна балансова вартість основних засобів, обладнання, комп’ютерів тощо, які використовувалися під час виконання роботи, грн;

$Н_а$ – річна норма амортизаційних відрахувань. Спрощено можна прийняти, що $Н_а$ = (5...25)%;

Т – термін, використання кожного виду основних засобів, місяці.

Розрахунки амортизаційних відрахувань наведено в таблиці 4.6.

Таблиця 4.6 - Амортизаційні відрахування (округлено)

Найменування обладнання	Балансова вартість, грн.	Норма аморти-зації, %	Термін використання обладнання, місяці	Величина амортизаційних відрахувань, грн.
1.Комп'ютер	32900	25	3	2056
2. Радіомонтажний стіл з оснасткою	8800	10	3	220
3. Оптичний стіл з оснасткою	11000	10	3	275
4.Пакети прикладних програм	4500	25	3	281
5. Приміщення університету, кафедри	13000	2,5	3	82
Всього				2914

Витрати на матеріали М розраховуються за формулою (4.8):

$$M = \sum_{1}^{n} H_i \cdot Ц_i \cdot K_i - \sum_{1}^{n} B_i \cdot Ц_в \text{ грн}, \quad (4.8)$$

де H_i – витрати матеріалу *i*-го найменування, кг;

$Ц_i$ – вартість матеріалу *i*-го найменування, грн/кг.;

K_i – коефіцієнт транспортних витрат, K_i = (1,1…1,15);

B_i – маса відходів матеріалу *i*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *i*-го найменування, грн/кг;

n – кількість видів матеріалів.

При виконанні роботи були використані: склотекстоліт; припій ПОС-61; полікор; клей; скло СОП1; спиртобензин СВС-50; лак УР-281; спирт гідролізний; дріт монтажний, полірит тощо. Загальна вартість всіх матеріалів становить 700 грн.

Витрати на комплектуючі К розраховуються за формулою (4.9):

$$К = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i \text{ грн,} \qquad (4.9)$$

де $Н_i$ – кількість комплектуючих *і*-го виду, шт.;

$Ц_i$ – ціна комплектуючих *і*-го виду, грн;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

n – кіль-кість видів комплектуючих.

При виконанні роботи були використані: ОКТ, світлооб'єднувальна призма, світлоподілювальна призма, дзеркало тощо. Загальна вартість комплектуючих, які були використані під час виконання даної роботи, становить приблизно 4100 грн.

Витрати на силову електроенергію $В_е$ розраховуються за формулою (4.10):

$$В_е = \frac{В \cdot П \cdot Ф \cdot К_п}{К_д}, \qquad (4.10)$$

де В – вартість 1 кВт-год. електроенергії, в 2019 р. В ≈ 2,5 грн/кВт;

П – установлена потужність обладнання, кВт; П = 2,70 кВт;

Ф – фактична кількість годин роботи обладнання, годин. Приймемо, що Ф = 158 годин;

$К_п$ – коефіцієнт використання потужності; $К_п$ < 1 = 0,85.

$К_д$ – коефіцієнт корисної дії, $К_д$ = 0,72.

Тоді витрати на силову електроенергію становитимуть:

$$В_е = \frac{В \cdot П \cdot Ф \cdot К_п}{К_д} = \frac{2,5 \cdot 2,70 \cdot 158 \cdot 0,85}{0,72} \approx 1259 \text{ грн.}$$

Інші витрати $В_{ін}$ (опалення, освітлення, ремонт, утримання приміщень тощо) розраховуються як (100…300)% від основної заробітної плати розробників, тобто:

$$В_{ін} = (1….3) \times (З_о+З_{роб}). \qquad (4.11)$$

Для нашого випадку отримаємо:

$$В_{ін} = 1,00 \times (24546+1365) = 25911 \text{ грн.}$$

Сума всіх попередніх статей дає витрати на виконання роботи безпосередньо магістрантом – $В_{заг}$.

$$В_{заг} = 24546+ 1365 + 2876+ 6333 +2914+700+4100+1259+25911=70004 \text{ грн.}$$

Загальні витрати на остаточне завершення роботи та оформлення їх результатів розраховуються за формулою (4.12):

$$ЗВ = \frac{В_{заг}}{\beta}, \qquad (4.12)$$

де β – коефіцієнт, який характеризує етап виконання даної роботи на шляху до її можливого впровадження. Для нашого випадку доцільно прийняти, що $\beta \approx 0,8$.

Тоді: $ЗВ = \frac{70004}{0,8} = 87505,00$ грн або приблизно 88 тисяч грн.

Тобто загальні витрати на остаточне завершення роботи та оформлення її результатів становлять приблизно 88 тис. грн.

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Економічний ефект від можливої комерціалізації нашої розробки – розробленої архітектерної організації паралельного оптико-електронного спецпроцесора – можливий за рахунок її значно кращих функціональних можливостей та характеристик, а також суттєвого зростання попиту на нашу розробку. Причому, якщо існуючі подібні розробки коштують на ринку в середньому приблизно 35 тис. грн, то нашу розробку (зі значно вищою конкуренто-спроможністю) можна буде реалізовувати на ринку дещо дорожче, наприклад, за 38 тис. грн, чи на 3 тис. грн дорожче.

Аналіз місткості ринку даної продукції показує, що в даний час в Україні кількість охочих придбати нашу (або аналогічну) розробку складає щороку приблизно 100 осіб, але їх кількість буде стрімко зростати. Оскільки наша розробка має кращі функціональні можливості та характеристики і є значно дешевшою за аналоги, то вона повинна користуватися попитом на ринку хоча б протягом 3-х років після впровадження.

Тобто наша розробка може бути впроваджена з 1 січня 2021 року (оскіль-ки потребує незначного доопрацювання), а її результати будуть виявлятися протягом 2021-го, 2022-го та 2023-го років.

Прогноз зростання попиту на нашу розробку складає по роках:

- 2021 р. – приблизно на Δ100 шт.;
- 2022 р. – приблизно на Δ120 шт.;
- 2023 р. – приблизно на Δ150 шт.

Розрахуємо можливе збільшення чистого прибутку $\Delta\Pi_i$, що його можна отримати потенційний інвестор від впровадження нашої розробки:

$$\Delta\Pi_i = \sum_1^n (\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\upsilon}{100}), \qquad (4.13)$$

де $\Delta Ц_o$ – зміна основного якісного показника від впровадження результатів розробки у даному році. Таким показником є зміна ціни нової розробки; для нашого випадку це буде: $\Delta Ц_o$ = (38 – 35) = +3 тис. грн;

N – основний кількісний показник, який визначає обсяг діяльності у даному році до впровадження результатів розробки; N = 100 шт.;

ΔN – покращення основного кількісного показника від впровадження результатів нашої розробки. Таке покращення відповідно по роках станови-тиме: Δ_{21} =+100, Δ_{22} = +120 та Δ_{23} = +150 шт.;

$Ц_о$ – основний якісний показник, який визначає обсяг діяльності у році після впровадження розробки; для нашого випадку $Ц_о$ = 38 тис. грн;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки; n = 3 роки;

λ – коефіцієнт, який враховує сплату податку на додану вартість;

$\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати ρ = (0,2...0,5); візьмемо ρ = 0,5;

υ – ставка податку на прибуток. У 2019 році υ = 18%.

Величина чистого прибутку $\Delta П_1$ для потенційного інвестора протягом першого року від можливого впровадження нашої розробки (2021 р.) складе:

$$\Delta П_1 = [3 \cdot 100 + 38 \cdot 100] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 1332$$ тис. грн.

Величина чистого прибутку $\Delta П_2$ для потенційного інвестора від можливого впровадження нашої розробки протягом другого (2022 р.) року складе:

$$\Delta П_2 = [3 \cdot 100 + 38 \cdot 120] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 1660$$ тис. грн.

Величина чистого прибутку $\Delta П_3$ для потенційного інвестора від можливого впровадження нашої розробки протягом третього (2023 р.) року складе:

$$\Delta П_3 = [3 \cdot 100 + 38 \cdot 150] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 2050$$ тис. грн.

Приведена вартість всіх можливих чистих прибутків ПП розраховується за формулою (в цінах на 1.12.2019 року):

$$ПП = \sum_{1}^{т} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.14)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої роботи, грн;

т – період часу, протягом якого виявляються результати впровадженої роботи, роки. Для нашого випадку т = 3 роки;

τ – ставка дисконтування; приймемо ставку дисконтування τ = 0,08 (8%);

t – період часу від моменту здійснення тих чи інших платежів (отримання прибутків та вкладення інвестицій) до моменту впровадження.

Тоді приведена вартість (в цінах на 1.12.2019 року) всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження нашої розробки, складе:

$$ПП = \frac{1332}{(1+0,08)^2} + \frac{1660}{(1+0,08)^3} + \frac{2050}{(1+0,08)^4} \approx 1142 + 1318 + 1507 = 3967 \text{ тис. грн.}$$

Далі розрахуємо початкову теперішню вартість інвестицій PV, що можуть бути вкладені інвестором у випадку реалізації нашої розробки:

$$PV = (2\ldots5) \times ЗВ, \quad (4.15)$$

де ЗВ – витрати на розробку; ЗВ = 88 тис. грн (див. формулу 4.12).

Тоді для нашого випадку отримаємо:

$$PV = (2\ldots5) \times 88 = 5 \times 88 = 440 \text{ тис. грн.}$$

Тоді абсолютний ефект від можливих вкладених інвестицій $E_{абс}$ може становити:

$$E_{абс} = ПП – PV, \quad (4.16)$$

де ПП – приведена вартість всіх можливих чистих прибутків від можливого впровадження нашої розробки, грн;

PV – теперішня вартість інвестицій PV = 440 тис. грн.

$E_{абс}$ = 3967 – 440 = 3527 тис. грн або приблизно по 1175 тис. грн щорічно протягом 3-х років.

Внутрішня норма дохідності $E_в$ інвестицій, вкладених у комерціалізацію нашої розробки, розраховується за формулою (4.17):

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.17)$$

де $E_{абс}$ – абсолютний ефект вкладених інвестицій; $E_{абс}$ = 3527 тис. грн;

PV – теперішня вартість початкових інвестицій PV = 440 тис. грн;

$T_ж$ – життєвий цикл розробки, роки. $T_ж$ = 4.

Для нашого випадку отримаємо:

$$E_в = \sqrt[4]{1 + \frac{3527}{440}} - 1 = \sqrt[4]{1 + 8{,}0159} - 1 = \sqrt[4]{9{,}0159} - 1 = 1{,}7328 – 1 \approx 0{,}7328 \approx 73{,}28\ \%.$$

Далі визначимо ту мінімальну дохідність, нижче за яку потенційний інвестор не буде зацікавлений займатися комерціалізацією нашої розробки.

Мінімальна дохідність або мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою (4.18):

$$\tau = d + f, \quad (4.18)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d = (0,10...0,19);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина

f = (0,05...0,5), але може бути і значно більше.

Для нашого випадку отримаємо:

$$\tau_{\text{мін}} = 0{,}15 + 0{,}50 = 0{,}65 \text{ або } \tau_{\text{мін}} = 65\%.$$

Оскільки величина $E_в$ = 73,28% > $\tau_{\text{мін}}$ = 65%, то потенційний інвестор може бути зацікавлений у комерційному впровадженні нашої розробки.

Далі розраховуємо термін окупності коштів, вкладених у нашу розробку. Термін окупності $T_{ок}$ можна розрахувати за формулою (4.19):

$$T_{ок} = \frac{1}{E_в}. \tag{4.19}$$

Термін окупності $T_{ок}$ коштів, вкладених у нашу розробку, становитиме:

$$T_{ок} = \frac{1}{0,7328} \approx 1,365 \text{ років,}$$

що свідчить про потенційну доцільність комерціалізації нашої розробки.

Результати виконаної економічної частини магістерської кваліфікаційної роботи зведено у таблицю 4.7:

Таблиця 4.7 Основні результати економічного розрахунку

Показники	Задані у ТЗ	Досягнуті у магістерській кваліфікаційній роботі	Висновок
1. Витрати на проведення досліджень	Не більше 100 тис. грн	88 тис. грн.	Виконано
2. Абсолютний щорічний ефект від можливого впровадження розробки, тис. грн	не менше 1000 тис. грн за рік	1175 тис. грн щорічно протягом 3-х років.	Виконано
3. Внутрішня норма дохідності вкладених інвестицій, %	не менше 65%	73,28%	Досягнуто
4. Термін окупності, роки	до 3-х років	1,365 років	Виконано

Таким чином, основні техніко-економічні показники проведених досліджень та розробленої архітектурної організації паралельного оптико-електронного спецпроцесора, визначені у технічному завданні, виконані.

ВИСНОВКИ

В магістерській кваліфікаційній роботі наведено вирішення наукової задачі забезпечення у комплексі високої швидкодії та багатофункціональності паралельних спецпроцесорів для матричної алгебри, побудованих на просторово-часових модуляторах світла.

Зокрема були отримані такі науково-дослідницькі та практичні результати.

1. В результаті проведення аналітичного огляду літературних джерел було встановлено, що на даний час існують лише окремі спецобчислювачі матричних операцій, які виконують визначення добутку або обернення матриць. Однак відсутні ефективні багатофункціональні оптоелектронні пристрої, які поєднували б можливості виконання обох цих операцій. Саме тому була визначена основна задача - розширення функціональних можливостей СП.
На основі використання властивостей природного паралелізму оптичних цифрових обчислень запропоновано паралельну модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність;

2. На основі використання властивостей природного паралелізму оптичних цифрових обчислень запропоновано паралельну модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність. Часова складність запропонованої моделі складає N тактів, в той час, як найкращі окремі спецобчилювачі реалізують свої алгоритми за N^2 тактів.

3. Розроблено архітектуру оптико-електронного спецпроцесора для обчислень добутку-обернення матриць на основі векторного добутку в форматі з плаваючою точкою шляхом її адекватного узгодження із запропонова-

ною паралельною моделлю. Архітектура СП не поступається аналогам за часовими характеристиками, швидкодією та точністю обробки даних та відрізняється підвищеною багатофункціональністю за рахунок виконання спецпроцесором додаткових матричних операцій, реалізованих на однотипних функціональних блоках, побудованих на двовимірних просторових модуляторах світла.

4. Показано варіант практичної реалізації оптико-електронного спецпроцесора для паралельних обчислень добутку-обернення матриць на оптично-керованих транспарантах із самонаведеним електрооптичним ефектом, який дозволив оцінити швидкодію спецпроцесора на рівні $1,4\times10^{10}$ *опер./с* при розширенні його функціональних можливостей при розмірності матриць $N = 320\times320$ та розрядності $P = 64$.

5. Проведено імітаційне моделювання розробленого спецпроцесора для паралельних обчислень добутку-обернення матриць для підтвердження адекватної роботи моделі. Встановлено, що швидкодія розробленого СП при оптоелектронній реалізації, де в якості базового вузла використовуються ОКТ, зростає пропорційно до збільшення розмірності вхідних матриць. Це обгрунтовує використання розробленого спецпроцесора саме для обробки великих інформаційних масивів даних, розмірність яких досягає 1000×1000 елементів, при досягненні швидкодії на рівні 10^5 *MFLOP* та часу спрацювання двовимірного просторового модулятора світла типу SEED на рівні 1 нс.

6. В результаті економічних розрахунків були визначені витрати на проведення досліджень, які склали 88 тис. грн, абсолютний щорічний ефект від можливого впровадження розробки на рівні 1175 тис. грн. щорічно протягом 3-х років, внутрішня норма дохідності вкладених інвестицій на рівні 73,28%, термін окупності до 1,365 років. Основні техніко-економічні показники проведених досліджень та розробленого паралельного оптико-електронного спецпроцесора, визначені у технічному завданні, виконані.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Дордоев С.Э. Оптическая цифровая вычислительная техника / С.Э.Дордоев // Зарубежная радиотехника.– 1989.– №10.– С.16-21.
2. Аллен Дж. Архитектура вычислительных устройств для цифровой обработки сигналов / Дж. Аллен // ТИИЭР. –1985. – Т.73 –№ 5. – С.4 –29.
3. Акаев А.А. Оптоэлектронная вычислительная система в остаточной арифметике для обработки изображений / А. А. . Акаев, С.З. Дордоев // Автометрия.– 1989.– №3.– С.48-53.
4. Кун С. Матричные процессоры на СБИС / С. Кун ; [перев. с англ].– М.: Мир, 1991.– 681 с.
5. К.Фу. СБИС для распознавания образов и обработки изображений / Фу К. ; [перев. с англ.] – М.: Мир, 1988.– 247 с.
6. Кейсесент Д. Акустооптические процессоры для операций линейной алгебры: Архитектура, алгоритмы, применение / Д. Кейсесент // ТИИЭР.– 1984.– Т.72, №7.– С.92 –113.
7. Farhat N. H. Optical implementation of the Hopfield model / Farhat N. H., Psaltis D., Prata A., Paek E. // Applied optics. – 1985. –Vol. 24. – P.1469-1475.
8. Форсайт Дэвид А. Компьютерное зрение. Современный подход = Computer Vision: A Modern Approach. / Форсайт Дэвид А., Понс Джин. — М.: Вильямс, 2004. — 928 с. — ISBN 0-13-085198-1.
9. Кулаков С.В. Акустооптические цифровые процессоры для операций матричной алгебры/ С. В. Кулаков // Зарубежная радиоэлектроника. – 1988.- №12. - С.30 – 40.
10. Проклов В.В. Акустооптические цифровые вычисления методом аналоговой свертки в спектральной плоскости / В.В. Проклов, О. А. Бышевский-Конопко, А. Л. Филатов // Радиотехника. – 2000.-№1. – С.50 – 54.
11. Вербовецкий А.А. Нетрадиционные методы построения цифрового конвейерных спецпроцессоров матричных операций / А.А. Вербовецкий // Радиотехника. – 1997.– №1.– С.89-92.

12. Вербовецкий А.А. Оптические методы построения цифровых оптических спецпроцессоров матричных операций / А. А. Вербовецкий // Радиотехника. – 1997.-№1. – С. 50-53.

13. Вербовецкий А.А. Современные методы создания оптической цифровой вычислительной техники/ А.А. Вербовецкий// Зарубежная радиоэлектроника. –1999. - №6. –С.12 –50.

14. Заболотна Н.І. Паралельна інтерпретація прямих методів розв'язання систем лінійних алгебраїчних рівнянь / Н.І. Заболотна, Шолота В.В., Веретенніков О.М. // Вимірювальна та обчислювальна техніка в технологічних процесах (Хмельницький).- №2.- 2000.- С.96-100.

15. Заболотна Н.І. Сучасні методи побудови оптико-електронних обчислювальних пристроїв для лінійно-алгебраїчних процесорів / Н.І. Заболотна, В.В. Шолота // Оптико-електронні інформаційно-енергетичні тех-нології.- №2.- 2001.- С.63-70.

16. Заболотная Н.И. Организация вычислительных структур высокопроизводительных линейно-алгебраических процессоров параллельной обработки матриц: Дис… канд. техн. наук: 05.13.08.– Винница, 1996.– 322 с.

17. Шолота В.В. Концепції та підходи до синтезу обчислювальних структур високопродуктивних процесорів для паралельного обернення матриць та розв’язання систем лінійних рівнянь / В. В. Шолота // Вимірювальна та обчислювальна техніка в технологічних процесах (Технологічний університет Поділля, м.Хмельницький). – 1998. – №2. – С.84-90.

18. Морозов В.Н. Оптоэлектронные матричные процессоры / Морозов В. Н. – М.: Радио и связь, 1986.–112 с.

19. Васильев А. А. Пространственные модуляторы света / [А.А.Васильев, Д.Касасент, И.Н.Компанец, А.В.Парфенов]. – М.: Радио и связь, 1987.– 320 с.

20. Нефф Дж.А. Двумерные пространственные модуляторы света: методический обзор / Нефф Дж.А., Атхале Р.А., Ли С.Х. // ТИИЭР.– 1990.– №5.– С.29-57.

21. Заболотная Н.И. Организация вычислительных структур высокопроизводительных линейно-алгебраических процессоров параллельной обработки матриц: дис. … кандидата техн. наук: 05.13.08 / Заболотная Наталия Ивановна. – Винница. 1995. – 227 с.

22. Заболотна Н.І., Гончарук І. Аналіз характеристичних точок фазових розподілів лазерних зображень плазми крові при діагностуванні патологій грудних залоз // XLVIII Науково-технічна конференція факультету комп'ютерних систем і автоматики , березень 2019р.: тези конференції – Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2019/paper/view/7682/6316

23. Бахрах Л.Д. Оптическая обработка сигналов приемных антенных решеток / Бахрах Л.Д. // Радиотехника.– 1990.– №5.– С.50-62.

24. Бутаков Е.А. Обработка изображений на ЭВМ / Е.А. Бутаков., В. И. Островский, И. Л.Фадеев – М.: Радио и связь.– 1987.– 235 с.

25. Головкин Б.А. Параллельные вычислительные системы / Головкин Б. А. – М.: Наука, 1980.– 243 с.

26. Шолота В.В. Структурна організація паралельних спецпроцесорів для матричних задач лінійної алгебри: дис. кандидата техн. наук: 05.13.13 / Шолота Владіслав Васильович. – Вінниця, 2000. – 205 с.

27. Заболотная Н.И., Шолота В.В. Организация параллельного перемножения знакопеременных матриц в цифровом оптоэлектронном процессоре многоуровневых изображений // Электронное моделирование. – 1997. - №3. – С.41 – 49.

28. Заболотна Н.І. Паралельний спецпроцесор для обчислення зовнішнього добутку векторів / Заболотна Н.І., Шолота В.В., Веретенніков О.М., Крук І.В. // Вимірювальна та обчислювальна техніка в технологічних процесах (Технологічний університет Поділля, м.Хмельницький).– 2000.– №3.– С.128-133.

29. Шолота В.В. Високопродуктивний процесор для паралельного розв’язання систем лінійних рівнянь та обернення матриць // Вимірювальна та обчислювальна техніка в технологічних процесах (Технологічний університет

Поділля, м.Хмельницький). – 1999. – №1. – С.86-91.

30. A.L.Lentine, H.S.Hinten. D.A.B.Miller et al. Electric field dependence of optical properties of semiconductor quantum wells: Physics and applications // Optical Engineering IEEE.– 1989.– №25.– P.1928.

31. Micah B. Yairi, Hilmi V. Demir, Chris W. Coldren, David A.B. Miller, and James S. Harris, Jr. Optically-Controlled Optical Gate Using a Double Diode Structure / Paper 12th Annual Meeting of the IEEE Lasers and Electro-Optics Society, LEOS'99.– San Francisco.– 1999.– P. 125-126.

32. Мялківська І. В. Швидкодіючі спеціалізовані обчислювачі на базі опто-електронних напівпровідникових транспарантів: автореф. дис. на здобуття наук. ступеня канд. техн. наук: 05.13.05 «Комп'ютерні системи та компоненти» / І. В. Мялківська. – Вінниця, 2009. – 20 с.

33. Мялківська І. Аналіз сучасних типів транспарантів та їх характеристик / Г. Лисенко, І. Мялківська // Оптико-електронні інформаційно-енергетичні технології. – 2007. - №2(14). – С.145-153.

34. Мялківська І. Дослідження оптичних властивостей напівпровідникових матеріалів типу $A^{III}B^{V}$ для виготовлення транспарантів / Лисенко Г.Л., Мяківська І. В. // Оптико-електронні інформаційно-енергетичні технології. – 2006. – №2(12). – С. 171 - 177.

35. Мялківська І. В. Використання оптичних транспарантів для спеціалізованих обчислювальних систем / І. В. Мялківська // Оптико-електронні інформаційно-енергетичні технології. – 2008. - №1(15). – С. 123 - 129.

36. Лисенко Г.Л. Оптоелектронний пристрій на основі транспарантів з повним набором логічних операцій для роботи з матрицями / Г. Л. Лисенко , І. В. Мялківська. О. В. Дюдюн // Оптико-електронні інформаційно-енергетичні технології. – 2009. - №1(17). – С. 71 - 76.

37. Borsook P. Alan Huang (англ.) // Network World. — 1990. — Vol. 7, no. 32. — P. 71.

38. Allman, William F. Computing's Bright Future. // U.S.News & World Report.– 1990.– Vol.108, Feb.12.– P.55-57.

39. Федоров В.Б. Оптические логические элементы для высокопроизводительных оптических процессоров // Квантовая электроника.– 1990.– №12.– С.1539-1545.

40. Micah B. Yairi, Hilmi V. Demir, Chris W. Coldren, David A.B. Miller, and James S. Harris, Jr. Optically-Controlled Optical Gate Using a Double Diode Structure / Paper 12th Annual Meeting of the IEEE Lasers and Electro-Optics Society, LEOS'99.– San Francisco.– 1999.– P. 125-126.

41. ABLAZETM 2D MQW Spatial Light Modulator Array – Режим доступу http://www.thirdwave.de/3w/tech/optical/Ablaze_Product_Brief.pdf

42. Денисов В.М. и др. Структура цифрового оптоэлектронного процессора многоуровневых изображений по пространственно-непрерывным раз¬рядным срезам // Электронное моделирование. – 1984. – №6. – С.99-106.

43. Кожем'яко В.П., Заболотна Н.І., Шолота В.В. Цифровий оптоелектронний суматор обробки матриць в формі з плаваючою комою / В.П. Кожем'яко, Н.І. Заболотна, В.В. Шолота // Вимірювальна та обчислювальна техніка в технологічних процесах (Технологічний університет Поділля, м.Хмельницький).– 1997.– №2.– С.136-142.

44. Kung-Shiuh Huang. Image algebra representation of parallel optical binary arithmetic / Kung-Shiuh Huang, B.Keith Tenkins, Alexander A.Sawchuk // Aplied Optics.– 1989.– Vol. 28, N6. – P.1263-1278.

45. Пат. 23431А Україна, МКІ G06F 15/66, G06E 1/04. Цифровий паралельний процесор багаторівневих зображень: Пат. 23431А Україна, МКІ G06F 15/66, G06E 1/04 / Кожем'яко В.П., Буда А.Г., Мартинюк Т.Б., Заболотна Н.І., Ліщинська Л.Б., Шолота В.В.– №96072786; Заявл. 11.07.96; Опубл. 31.08.98, Бюл. №4.– 6 с.

46. Кожем'яко В.П. Паралельний поділювач вектора на число в формі з рухомою комою / В.П. Кожем'яко В.П., В.В. Шолота, А. В. Зволейко// Вісник ВПІ. – 1998. - №4.- С.56-61.

47. Красиленко В.Г., Заболотная Н.И., Евтихиев Н.Н. Эффективность многоканального блока накапливающих сумматоров со взвешиванием в цифровом

векторно-матричном перемножителе // УСиМ.– 1995.– №1/2.– С.31-36.

48. Оптоэлектронное бистабильное устройство для параллельной записи, хранения и считывания изображения: А.с. 1451740 СССР, МКИ G 06 K 9/00 / Красиленко В.Г., Дубчак В.И. – №4250323/24; Заявлено 26.05.87; Опубл. 09.12.88, Бюл №2.– 4с. ил. В.Б. Котов, А.И. Микаэлян, В.К. Салахутдинов, В.А. Оптоэлектронная коммутация гигабитных потоков информации // Радиотехника. – 1990, – №2, С. 78-82.

49. Захаров С.М. Оптоэлектронные интегральные схемы с применением полупроводниковых вертикально излучающих лазеров / С.М. Захаров, В.Б. Фёдоров, В.В. Цветков // Квантовая электроника. – 1999. – №3. – С. 189-205.

50. Заболотна Н.І. Комп’ютерне моделювання задач лазерної та оптоелектронної техніки. Навчальний посібник. – Вінниця: ВНТУ, 2003. – 151 с.

51. Miller D.A. Quantum-Well-Self-Electro-Optic-Effect Devices / D.A. Miller //Optical and Quantum Electronics. –1990. –Vol.22. – P.61 – 98.

52. Faber S. The optical computer, Model T / S. Faber // Discover.– 1994.– vol.15, January.– P.95 – 96.

Додаток А

(обов’язковий)

ЗАТВЕРДЖУЮ

Завідувач кафедри ЛОТ

д.т.н., проф. Заболотна Н.І.

____

«___» ___________ 2019 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу

ПАРАЛЕЛЬНИЙ ОПТИКО-ЕЛЕКТРОННИЙ СПЕЦПРОЦЕСОР ДЛЯ МАТРИЧНОЇ АЛГЕБРИ НА МОДУЛЯТОРАХ СВІТЛА

спеціальність 152 – Метрологія та інформаційно-вимірювальна техніка

освітня програма «Лазерна техніка та оптоінформатика»

Керівник,
д.т.н., проф.

_____ Заболотна Н. І.

Виконавець,
студент гр. ЛТО-18-м

_____ Гончарук І. В.

Вінниця ВНТУ 2019

1. Підстава для виконання проекту

Робота виконується на підставі наказу ректора ВНТУ № 254 від «02» жовтня 2019р. та індивідуального завдання на магістерську кваліфікаційну роботу.

2. Мета та призначення

Метою роботи є підвищення багатофункціональності паралельного спецпроцесора для матричної алгебри з високою швидкодією обчислень за рахунок виконання ним додаткових матричних операцій, реалізованих на однотипних функціональних блоках, побудованих на двовимірних просторових модуляторах світла.

Дана система може бути застосована в інформаційно-вимірювальних системах діагностики об’єктів із формуванням лазерних зображень вимірюваних параметрів, для обробки яких застосовується розроблеий

3. Вимоги до виконання МКР

Основними вимогами є:

- розробити паралельну модель організації обчислень добутку-обернення матриць на основі векторного добутку для спецпроцесора матричних операцій, яка дозволятиме підвищити його багатофункціональність;

- розробити архітектуру оптико-електронного спецпроцесора для обчислень добутку-обернення матриць на основі векторного добутку в форматі з плаваючою точкою;

- розглянути аспекти практичної реалізації оптико-електронного спецпроцесора для паралельних обчислень добутку-обернення матриць на оптично-керованих транспарантах із самонаведеним електрооптичним ефектом;

- провести імітаційне моделювання розробленого спецпроцесора для паралельних обчислень добутку-обернення матриць для підтвердження адекватної роботи моделі;

- оцінити часові характеристики розробленого оптико-електронного спецпроцесора для матричної алгебри на модуляторах світла.

4. Джерела розробки

Список використаних джерел розробки:

1. Заболотна Н.І. Сучасні методи побудови оптико-електронних обчислювальних пристроїв для лінійно-алгебраїчних процесорів / Н.І. Заболотна, В.В. Шолота // Оптико-електронні інформаційно-енергетичні тех-нології.- №2.- 2001.- С.63-70.
2. Шолота В.В. Концепції та підходи до синтезу обчислювальних структур високопродуктивних процесорів для паралельного обернення матриць та розв’язання систем лінійних рівнянь / В. В. Шолота // Вимірювальна та обчислювальна техніка в технологічних процесах (Технологічний університет Поділля, м.Хмельницький). – 1998. – №2. – С.84-90.
3. Нефф Дж.А. Двумерные пространственные модуляторы света: методический обзор / Нефф Дж.А., Атхале Р.А., Ли С.Х. // ТИИЭР.– 1990.– №5.– С.29-57.

5. Технічні вимоги

1 Функціональне призначення пристрою: визначення добутку матриць та обернення матриці.

2. Розмірність матриць NxN елементів, де N=320 для конкретної реалізації.

4.;Формат подання елементів матриць: з плаваючою точкою, де М – розрядність мантиси; Р – розрядність порядку. Для моделювання М=48, Р=16.

5. Спосіб оброблення – позрізовий цифровий.

5. Елементна база – модулятори світла на квантових ямах.

6. Етапи БДП і терміни його виконання

	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Розробка, погодження і затвердження ТЗ			Затверджене ТЗ
2	Аналіз методів, структур та елементної бази для побудови паралельного оптико-електронного спецпроцесора матричних операцій.			Технічна частина МКР
3	Розробка паралельної моделі та архітектури спецпроцесора для матричних операцій. Практична реалізація спецпроцесора.			Технічна частина МКР
4	Розробка економічної частини			Економічна частина
5	Оформлення необхідної технічної документації, підготовка магістерської роботи до публічного захисту			МКР

7. Порядок контролю і приймання

Контроль за виконанням МКР та його етапів покладається на керівника.

Зміст питань економічної частини погоджується зі спеціалістами (консультантами з даних питань).

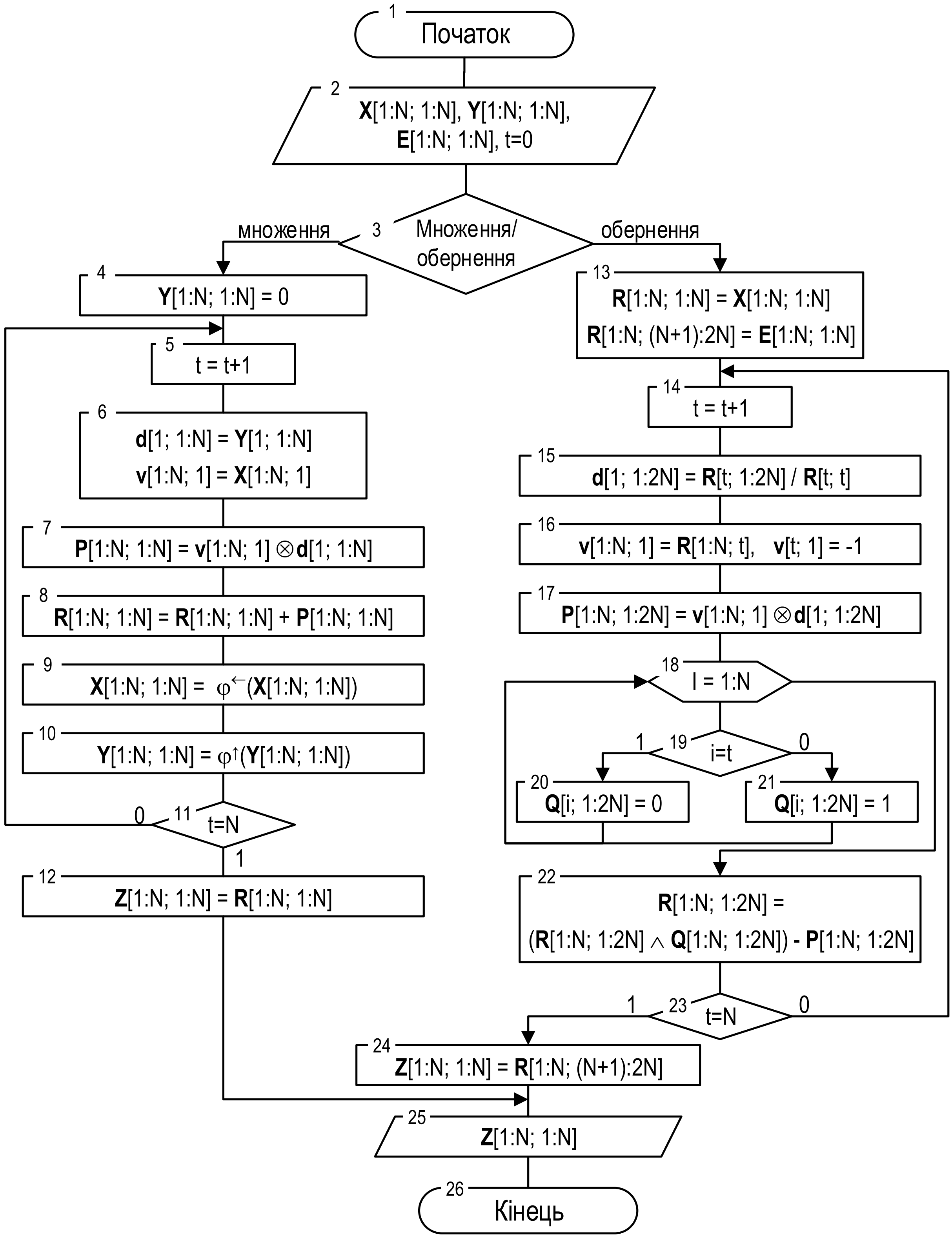
Приймання МКР здійснюється шляхом його публічного захисту перед Державною екзаменаційною комісією (ДЕК), призначеною за наказом ректора ВНТУ.

8. Вимоги щодо технічного захисту інформації

У зв’язку з тим, що інформація не є конфіденційною, заходи з технічного захисту не передбачаються.

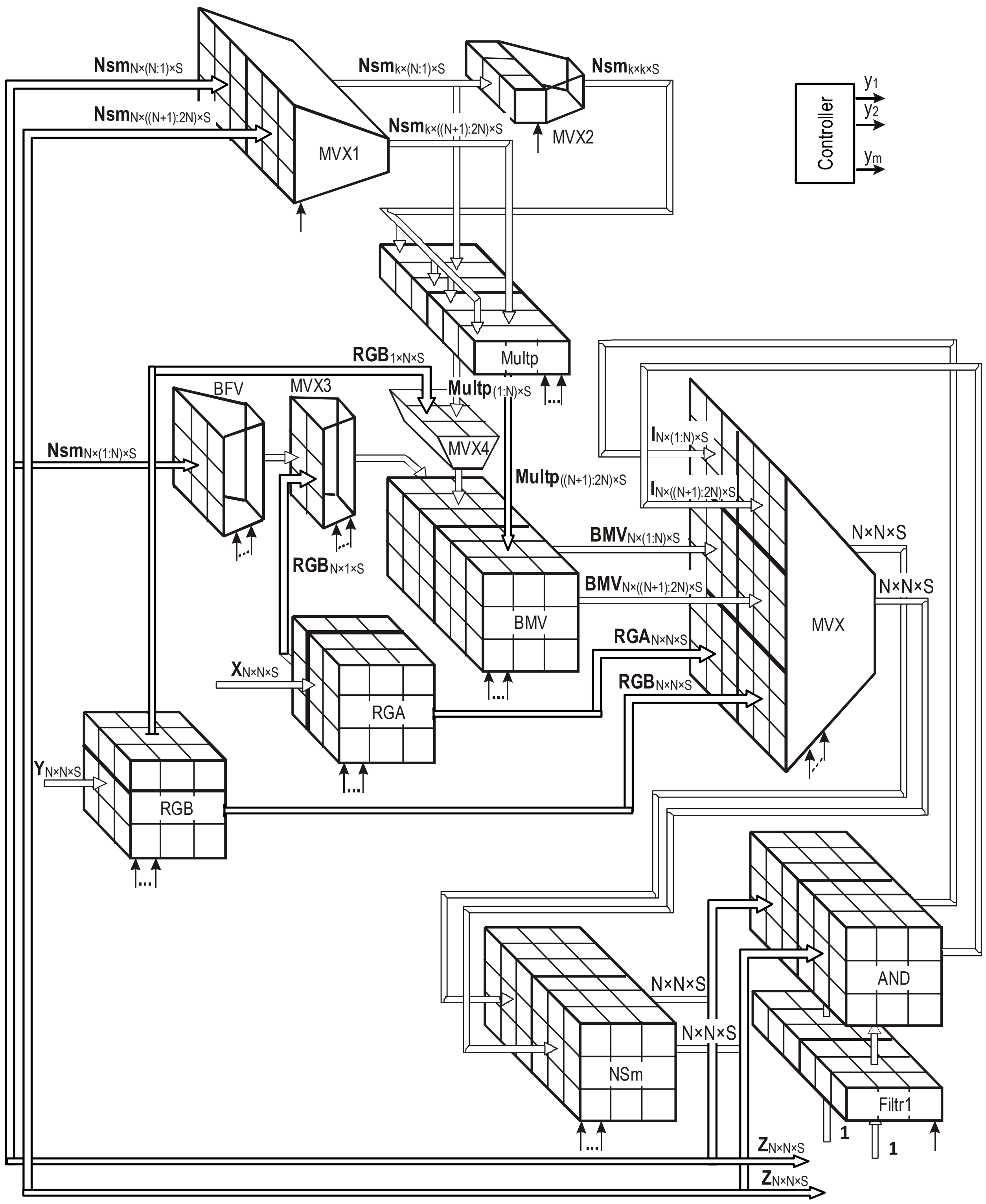
Додаток Б

Блок-схема паралельного алгоритму обчислення добутку-обернення матриць на основі векторного добутку



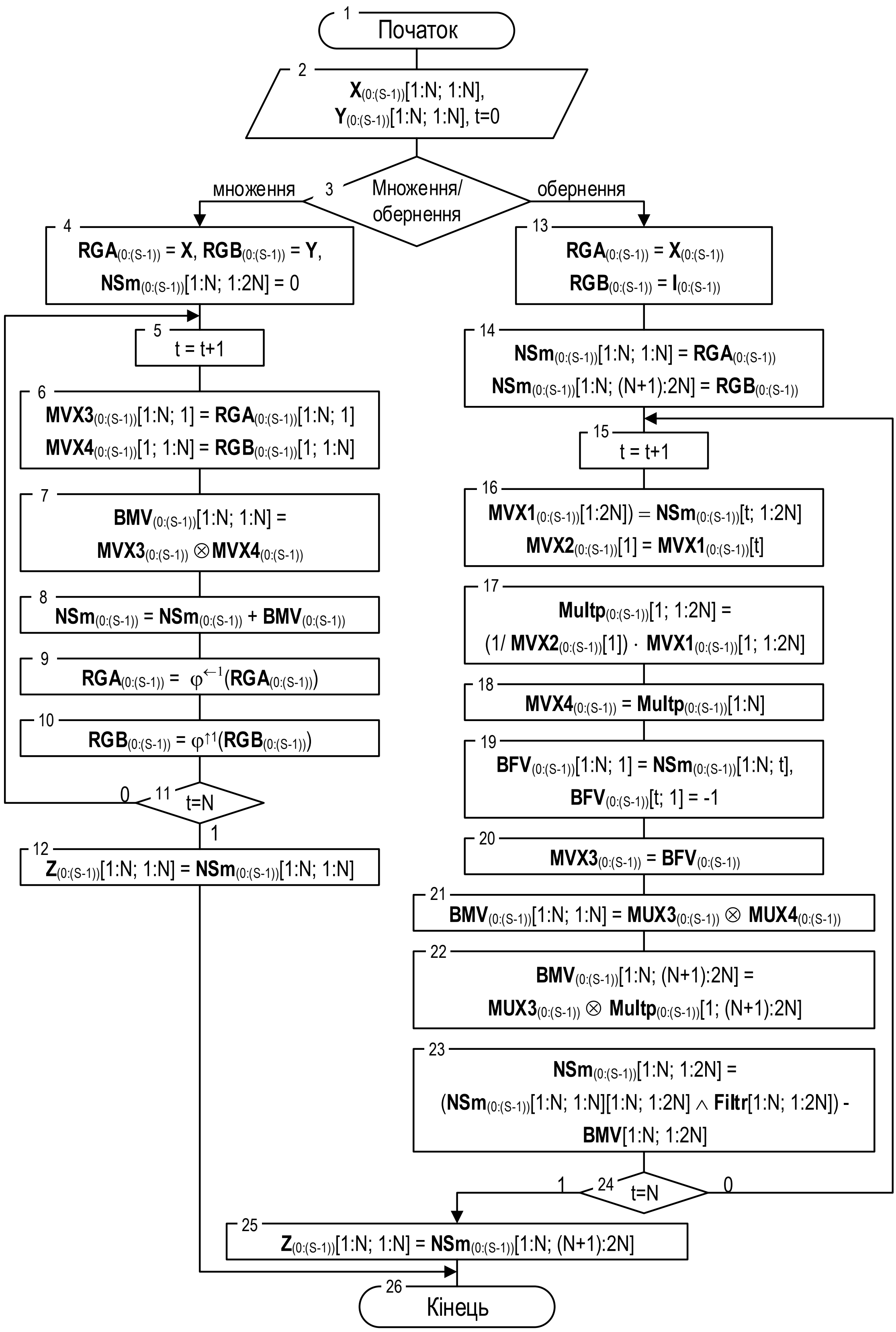
Додаток В

Схема структурна оптико-електронного спецпроцесора для добутку-обернення матриць на основі векторного добутку



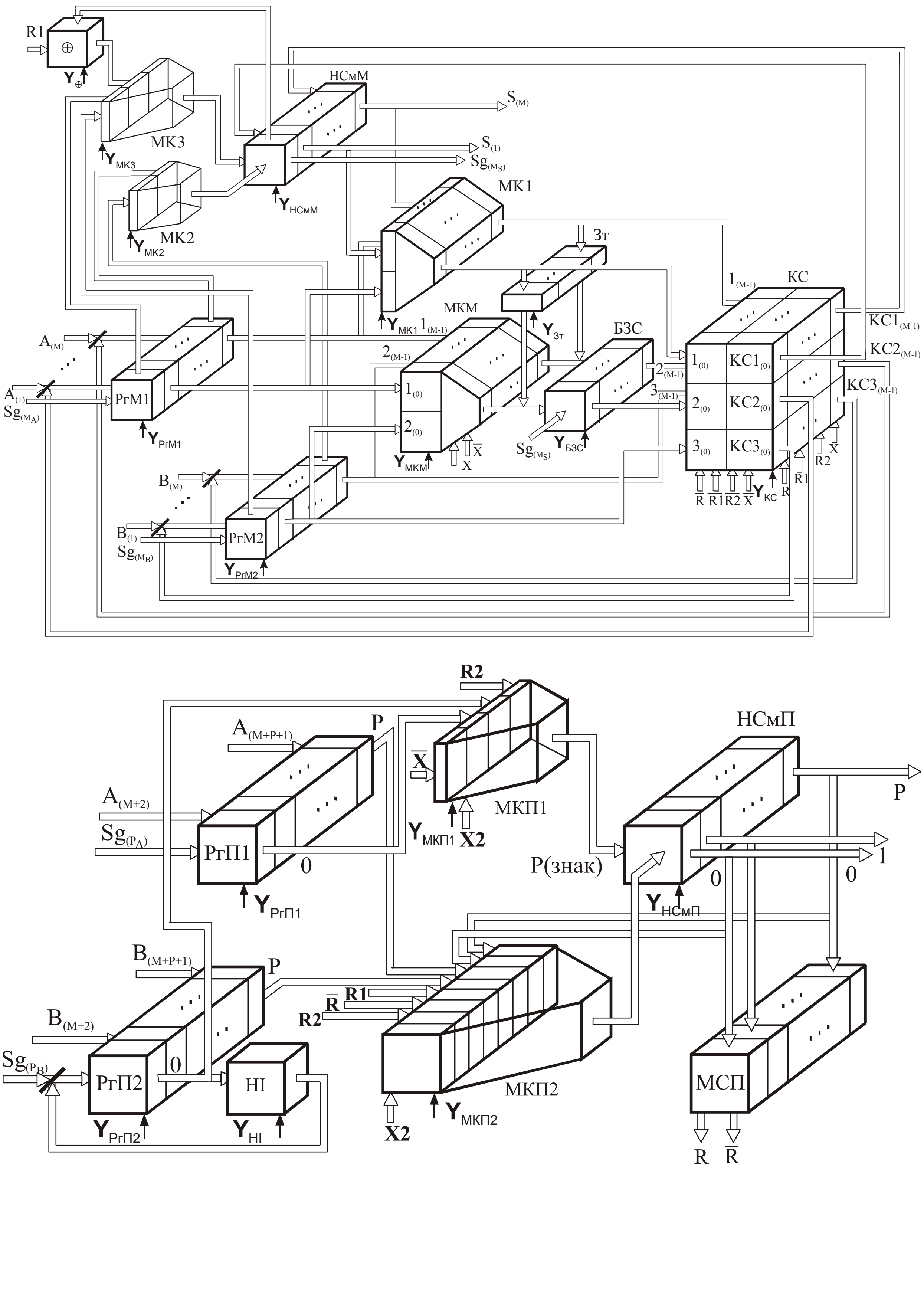
Додаток Г

Блок-схема алгоритму роботи оптоелектронного спецпроцесора для добутку-обернення матриць на основі векторного добутку



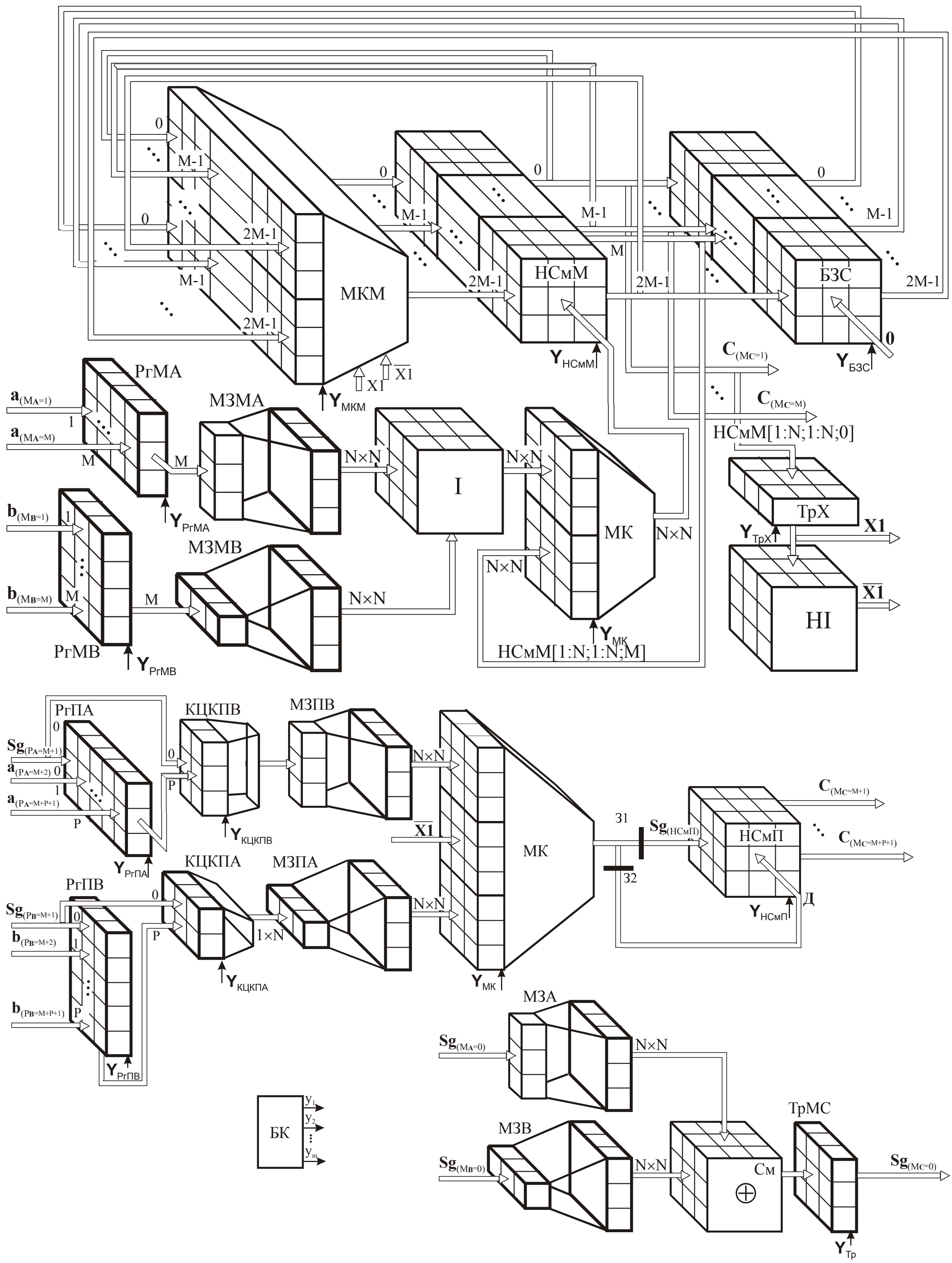
Додаток Д

схема структурна матричного накопичувального суматора з плаваючою точкою



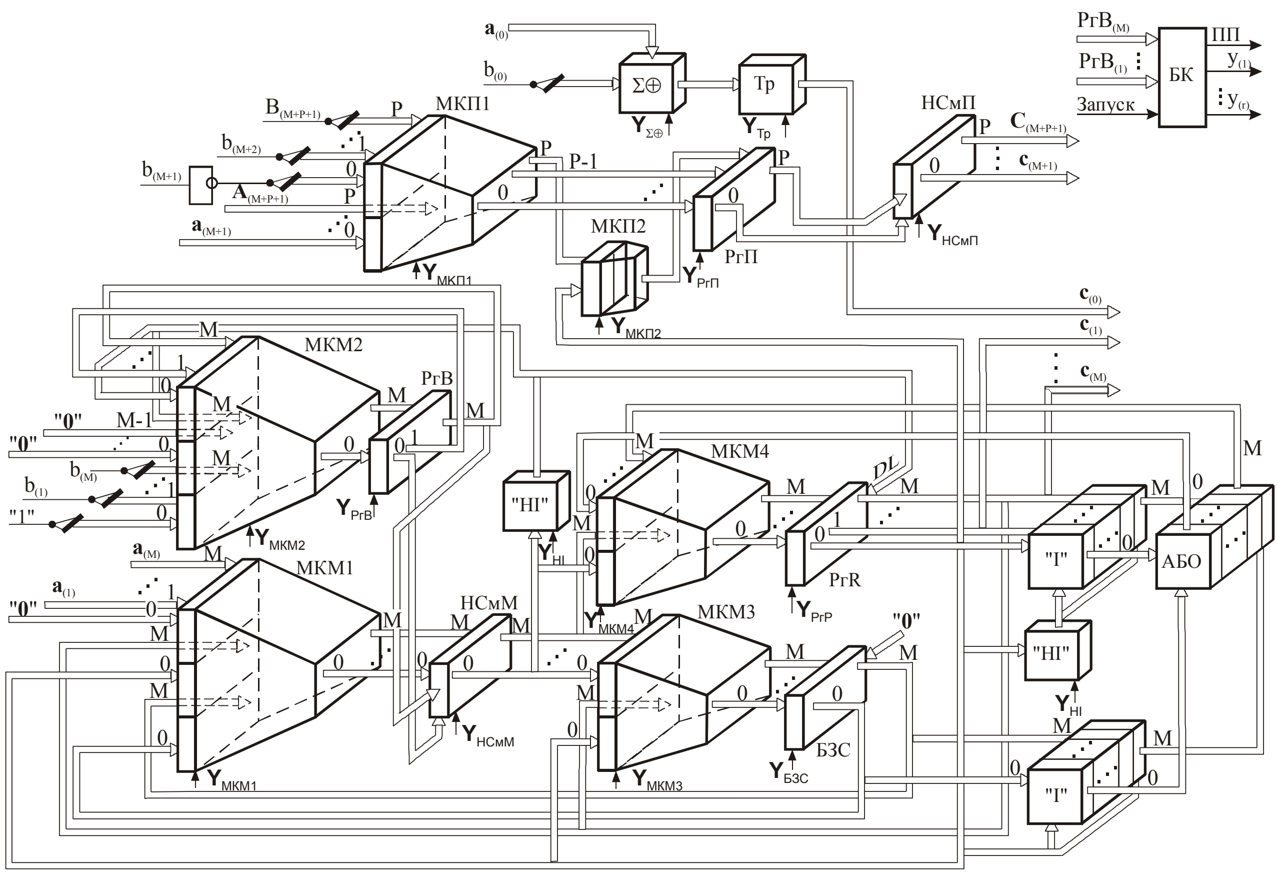
Додаток Е

Схема структурна паралельного блоку добутку векторів з плаваючою точкою



Додаток Ж

Схема структурна паралельного блоку множення вектора на обернений коефіцієнт



Додаток И

Лістинг програми

Binar.h

```
class Cell
{
public:
		CELL *Cage;
		// initialization block
   Cell(int length)
   {
				Cage=(CELL *) calloc(length,sizeof(CELL));
				/*char st = 0;
      for(int i=0; i<N1; i++)
				{
						for(int j=0; j<N2; j++)
						{
								Cage[0][i][j]=st;
						}
				}
				*/
   }

		void Dispose()
		{
				free(Cage);
		}

		void Equate(Cell Y,int height,int width,int sliseNumX,int sliseNumY)
		{
				for(int i=0; i<height; i++)
				{
						for(int j=0; j<width; j++)
						{
								Cage[sliseNumX][i][j]=Y.Cage[sliseNumY][i][j];
						}
				}
		}


		void Init(char st,int height,int width,int sliseNum)
		{
				for(int i=0; i<height; i++)
				{
						for(int j=0; j<width; j++)
						{
								Cage[sliseNum][i][j]=st;
						}
				}
		}
		// initialization block end

		// IO block
		void InputData(char *file, int height, int width)
		{
				FILE *fi;
				float inf;
				long double IN[N1][N1];
				double ri,r,b,fraction,part;
				long ord;
				if (!(fi=fopen(file,"rt")))
				{
						printf("\nNo file!");
				}
				for(int i=0; i<height; i++)
				{
						for(int j=0; j<width; j++)
						{
								fscanf(fi, "%f", &inf);
								IN[i][j]=inf;
								printf("%f ",inf);
						}
						fscanf(fi,"\n");
```

```
		printf("\n");
	}
	fclose(fi);
	for(int i=0; i<height; i++)
	{
		for(int j=0; j<width; j++)
		{
			if(IN[i][j]==0)
			{
				for(int l=0;j<l;l++)
				{
					Cage[j][i][j]=0;
				}
			}
			else
			{
				if(IN[i][j]<0)
				{
					IN[i][j]=-IN[i][j];
					Cage[0][i][j] = 1;
				}

				ord = -30;
				for(b=0.0000000009;b<1024000000;b*=2)
				{
					if (b>IN[i][j])
					{
						break;
					}
					ord++;
				}
				fraction = IN[i][j]/b;
				part = 0.5;
				for(int  f = 1;f<FractionLength+1;f++)
				{
					if(part<=fraction)
					{
						Cage[f][i][j]=1;
						fraction -= part;
					}
					else
					{
						Cage[f][i][j]=0;
					}
					part/=2;
				}
				if(ord<0)
				{
					ord = -ord;
					Cage[FractionLength+1][i][j] = 1;
					for(int o = S-1;o>FractionLength+2;o--)
					{
						if(ord == 1)
						{
							Cage[o][i][j]=1;
							break;
						}
						ri = ord;
						r = ri/2;
						ord = ord/2;
						if(r>ord)
						{
							Cage[o][i][j]=1;
						}
						else
						{
							Cage[o][i][j]=0;
						}
					}
				}
				else
				{
					for(int o = S-1;o>FractionLength+2;o--)
					{
						if(ord == 1)
						{
							Cage[o][i][j]=1;
							break;
						}
```

```
                                                ri = ord;
                                                r = ri/2;
                                                ord = ord/2;
                                                if(r>ord)
                                                {
                                                        Cage[o][i][j]=1;
                                                }
                                                else
                                                {
                                                        Cage[o][i][j]=0;
                                                }
                                        }
                                }
                        }
                }
        }
}

void InputDataFl(float fl[N1][N2], int height, int width)
{

        float inf;
        long double IN[N1][N1];
        double ri,r,b,fraction,part;
        long ord;
        //printf("Convertting \n");
        for(int i=0; i<height; i++)
        {
                for(int j=0; j<width; j++)
                {
                        inf = fl[i][j];
                        IN[i][j]=inf;
                        printf("%f ",inf);
                }
                printf("\n");
        }
        //printf("End Convertting \n");
        for(int i=0; i<height; i++)
        {
                for(int j=0; j<width; j++)
                {
                        if(IN[i][j]==0)
                        {
                                for(int l=0;j<l;l++)
                                {
                                        Cage[j][i][j]=0;
                                }
                        }
                        else
                        {
                                if(IN[i][j]<0)
                                {
                                        IN[i][j]=-IN[i][j];
                                        Cage[0][i][j] = 1;
                                }

                                ord = -30;
                                for(b=0.000000000931322574615478515625;b<1024000000;b*=2)
                                {
                                        if (b>IN[i][j])
                                        {
                                                break;
                                        }
                                        ord++;
                                }
                                fraction = IN[i][j]/b;
                                part = 0.5;
                                for(int  f = 1;f<FractionLength+1;f++)
                                {
                                        if(part<=fraction)
                                        {
                                                Cage[f][i][j]=1;
                                                fraction -= part;
                                        }
                                        else
                                        {
                                                Cage[f][i][j]=0;
                                        }
                                        part/=2;
```

```
							}
							if(ord<0)
							{
								ord = -ord;
								Cage[FractionLength+1][i][j] = 1;
								for(int o = S-1;o>FractionLength+2;o--)
								{
									if(ord == 1)
									{
										Cage[o][i][j]=1;
										break;
									}
									ri = ord;
									r = ri/2;
									ord = ord/2;
									if(r>ord)
									{
										Cage[o][i][j]=1;
									}
									else
									{
										Cage[o][i][j]=0;
									}
								}
							}
							else
							{
								for(int o = S-1;o>FractionLength+2;o--)
								{
									if(ord == 1)
									{
										Cage[o][i][j]=1;
										break;
									}
									ri = ord;
									r = ri/2;
									ord = ord/2;
									if(r>ord)
									{
										Cage[o][i][j]=1;
									}
									else
									{
										Cage[o][i][j]=0;
									}
								}
							}
						}
					}
				}
}
void OutputDataFl(float OUT[N1][N2], int height, int width)
{
	float kk;
	int  i1,k;

	for(int i=0; i<height; i++)
	{
		for(int j=0; j<width; j++)
		{
			kk=1;
			OUT[i][j]=0;
			for(int l=1; l<(NumberLength+1); l++)
			{
				kk=kk/2;
				OUT[i][j]+=Cage[l][i][j]*kk;
			}
			if(Cage[0][i][j]==1)
			{
				OUT[i][j]=OUT[i][j]*(-1);
			}
			k=1;
			i1=0;
			for(int l=NumberLength-1; l>FractionLength+1; l--)
			{
				i1+=Cage[l][i][j]*k;
				k=k*2;
			}
			kk=1;
```

```
				for(int l=0; l<i1; l++)
				{
					kk=kk*2;
				}
				if(Cage[M+1][i][j]==0)
				{
					OUT[i][j]=OUT[i][j]*kk;
				}
				else
				{
					OUT[i][j]=OUT[i][j]/kk;
				}
			}
		}

}


void OutputData(char *file, int height, int width)
{
		FILE *fo;
		float kk,OUT[N1][N2];
		int  i1,k;
		if (!(fo=fopen(file,"wt")))
		{
			printf("\nCann't open file!");
			return;
		}
		for(int i=0; i<height; i++)
		{
			for(int j=0; j<width; j++)
			{
				kk=1;
				OUT[i][j]=0;
				for(int l=1; l<(NumberLength+1); l++)
				{
					kk=kk/2;
					OUT[i][j]+=Cage[l][i][j]*kk;
				}
				if(Cage[0][i][j]==1)
				{
					OUT[i][j]=OUT[i][j]*(-1);
				}
				k=1;
				i1=0;
				for(int l=NumberLength-1; l>FractionLength+1; l--)
				{
					i1+=Cage[l][i][j]*k;
					k=k*2;
				}
				kk=1;
				for(int l=0; l<i1; l++)
				{
					kk=kk*2;
				}
				if(Cage[M+1][i][j]==0)
				{
					OUT[i][j]=OUT[i][j]*kk;
				}
				else
				{
					OUT[i][j]=OUT[i][j]/kk;
				}
			}
		}
		for(int i=0; i<height; i++)
		{
			for(int j=0; j<width; j++)
			{
				fprintf(fo, "%f ",OUT[i][j]);
				printf("%f ",OUT[i][j]);
			}
			printf("\n");
			fprintf(fo, "\n");
		}
		fclose(fo);
}
```

```
void Slise(char str[20],int height, int width, int num)
{
	printf("%s\n", str);
	for(int i=0; i<height; i++)
	{
		for(int j=0; j<width; j++)
		{
			printf("%2d",Cage[num][i][j]);
		}
		printf("\n");
	}
}

void Print(int height, int width, int length)
{
	for(int i=0;i<height;i++)
		for(int j=0;j<width;j++)
		{
			printf("\n [0:FractionLength][%d][%d]=        ",i,j);
			if(length>(FractionLength+1))
			{
				for(int l=0; l<FractionLength+1; l++)
				{
					printf("%d",Cage[l][i][j]);
				}
				printf("\n [(FractionLength+1):(NumberLength-1)][%d][%d]=",i,j);
				for(int l=FractionLength+1; l<NumberLength; l++)
				{
					printf("%d",Cage[l][i][j]);
				}
			}
			else
			{
				for(int l=0; l<length; l++)
				{
					printf("%d",Cage[l][i][j]);
				}
			}
		}
	printf("\n");
}
// IO block end

//compatibility block
CELL* GetCELL()
{
	return Cage;
}
void SetFromCELL(CELL* Y,int height, int width, int length)
{
	for(int i=0; i<height; i++)
	{
		for(int j=0; j<width; j++)
		{
			for(int l = 0;l<length;l++)
			{
				Cage[l][i][j]=Y[l][i][j];
			}
		}
	}
}
//compatibility block end

//shifts block
void LeftShift(Cell Y, int height, int width, int sliseNum)
{
	for(int i=0; i<height; i++)
	{
		for(int j=0; j<width-1; j++)
		{
			Cage[sliseNum][i][j]=Cage[sliseNum][i][j+1];
		}
	}
	for(int i=0; i<height; i++)
	{
		Cage[sliseNum][i][width-1]=Y(0,i,width-1);//[0][i][JJ-1];
	}
}
```

```
void RightShift(Cell Y, int height, int width, int sliseNum)
{
        for(int i=0; i<height; i++)
        {
                for(int j=width-1; j>0; j--)
                {
                        Cage[sliseNum][i][j]=Cage[sliseNum][i][j-1];
                }
        }
        for(int i=0; i<height; i++)
        {
                Cage[sliseNum][i][0]=Y(0,i,0);
        }
}

void UpShift(Cell Y, int height, int width, int sliseNum)
{
for(int i=0; i<height-1; i++)
        {
                for(int j=0; j<width; j++)
                {
                        Cage[sliseNum][i][j]=Cage[sliseNum][i+1][j];
                }
        }
        for(int j=0; j<width; j++)
        {
                Cage[sliseNum][height-1][j]=Y(0,height-1,j);
        }
}

void DownShift(Cell Y, int height, int width, int sliseNum)
{
        for(int i=height-1; i>0; i--)
        {
                for(int j=0; j<width; j++)
                {
                        Cage[sliseNum][i][j]=Y(sliseNum,i-1,j);
                }
        }
        for(int j=0; j<width; j++)
        {
                Cage[sliseNum][0][j]=Y(0,0,j);
        }
}

void LeftSliseShift(Cell Y, int height, int width, int sliseNum)
{
        for(int l=sliseNum-1; l>0; l--)
        {
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Cage[l][i][j]=Cage[l-1][i][j];
                        }
                }
        }
        for(int i=0; i<height; i++)
        {
                for(int j=0; j<width; j++)
                {
                        Cage[0][i][j]=Y(0,i,j);
                }
        }
}
void RightSliseShift(Cell Y, int height, int width, int sliseNum)
{
        for(int l=0; l<(sliseNum-1); l++)
        {
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Cage[l][i][j]=Cage[l+1][i][j];
                        }
                }
        }
        for(int i=0; i<height; i++)
        {
```

```
                        for(int j=0; j<width; j++)
                        {
                                Cage[sliseNum-1][i][j]=Y(0,i,j);
                        }
                }
        }

        void YoungRZShift(Cell Y,int height,int width, int sliseCount)
{
        for(int l=sliseCount-1; l>0; l--)
        {
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Cage[l][i][j]=Cage[l-1][i][j];
                        }
                }
        }
        for(int i=0; i<height; i++)
        {
                for(int j=0; j<width; j++)
                {
                        Cage[0][i][j]=Y.Cage[0][i][j];
                }
        }
}

        void OldRZShift(Cell Y,int height,int width, int sliseCount)
        {
                for(int l=0; l<(sliseCount-1); l++)
                {
                        for(int i=0; i<height; i++)
                        {
                                for(int j=0; j<width; j++)
                                {
                                        Cage[l][i][j]=Cage[l+1][i][j];
                                }
                        }
                }
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Cage[sliseCount-1][i][j]=Cage[0][i][j];
                        }
                }
        }
   int operator () (int i, int j,int l)//обращаться к элементу можно теперь так   m(1,2,1);
   {
      return Cage[i][j][l];
   }
};
BinarLogic.h

class Logic
{
public:
        Logic()
        {}
        void And(Cell X, Cell Y,Cell Z,int height, int width, int sliseNumX,int sliseNumY,int sliseNumZ)
        {
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Z.Cage[sliseNumZ][i][j]=X.Cage[sliseNumX][i][j]&Y.Cage[sliseNumY][i][j];
                        }
                }
        }
        void Or(Cell X, Cell Y, Cell Z,int height,int width,int sliseNumX,int sliseNumY,int sliseNumZ)
        {
                for(int i=0; i<height; i++)
                {
                        for(int j=0; j<width; j++)
                        {
                                Z.Cage[sliseNumZ][i][j]=X.Cage[sliseNumX][i][j]|Y.Cage[sliseNumY][i][j];
                        }
                }
```

```
		}
		void Xor(Cell X,Cell Y,Cell Z,int height,int width, int sliseNumX,int sliseNumY,int sliseNumZ)
		{
			for(int i=0; i<height; i++)
			{
				for(int j=0; j<width; j++)
				{
					Z.Cage[sliseNumZ][i][j]=X.Cage[sliseNumX][i][j]^Y.Cage[sliseNumY][i][j];
				}
			}
		}
		void Not(Cell X, Cell Y,int height,int width,int sliseNumX,int sliseNumY)
		{
			for(int i=0; i<height; i++)
			{
				for(int j=0; j<width; j++)
				{
					Y.Cage[sliseNumY][i][j]=X.Cage[sliseNumX][i][j]^1;
				}
			}
		}
		int Compare(Cell X, Cell Y,int height,int width, int sliseNumX,int sliseNumY)
		{
			int res = 1;
			for(int i=0; i<height; i++)
			{
				for(int j=0; j<width; j++)
				{
					int q1 = Y.Cage[sliseNumY][i][j];
					int q2 = X.Cage[sliseNumX][i][j];
					if(Y.Cage[sliseNumY][i][j]!=X.Cage[sliseNumX][i][j])
					{
						res=0;
						break;}}}
			return res; } };
BinarSum.h

class Adder
{
public:
		Adder()
		{
		}

		void sum(Cell X,Cell Y,Cell Z,Cell OutS,Cell OutP,int height,int width)
   {
			for(int i=0; i<height; i++)
			{
				for(int j=0; j<width; j++)
				{
					OutS.Cage[0][i][j]=X.Cage[0][i][j]^Y.Cage[0][i][j]^Z.Cage[0][i][j];

OutP.Cage[0][i][j]=((X.Cage[0][i][j]^Y.Cage[0][i][j])&Z.Cage[0][i][j])^(X.Cage[0][i][j]&Y.Cage[0][i][j]);
				}
			}
   }
		void NSm(Cell SS,Cell A,int height,int width,int sliseCount,int k_op)
   {


			 Cell X= Cell(1);
      Cell Y= Cell(1);
      Cell Z= Cell(1);
      Cell D= Cell(1);
      Cell NS = Cell(1);
      Cell NP = Cell(1);
      Cell F1 = Cell(1);
      Cell F2 = Cell(1);
      Cell Ep1 = Cell(1);
      Cell Ep2 = Cell(1);


			Ep1.Equate(A,height,width,0,0);

			Ep2.Equate(SS,height,width,0,0);

			F1.Init(1,height,width,0);

			Logic Log;
```

```
			if(k_op==0)
			{
				Ep2.Init(0,height,width,0);
			}
			for(int l = sliseCount;l>0;l--)
			{
				D.Equate(A,height,width,0,l);
				Log.Xor(D,Ep1,X,height,width,0,0,0);
				Log.Xor(SS,Ep2,Y,height,width,sliseCount,0,0);
				Log.Or(Ep2,Ep1,Z,height,width,0,0,0);
				Log.And(Z,F1,Z,height,width,0,0,0);
				Log.And(NP,F2,NP,height,width,0,0,0);
				Log.Or(NP,Z,Z,height,width,0,0,0);
				sum(Y,X,Z,NS,NP,height,width);
				SS.YoungRZShift(NS,height,width,(sliseCount+1));
				F1.Init(0,height,width,0);
				F2.Init(1,height,width,0);
			}


			Ep1.Init(0,height,width,0);
			Ep2.Init(0,height,width,0);
			D.Equate(A,height,width,0,l);
			Log.Xor(D,Ep1,X,height,width,0,0,0);//
			Log.Xor(SS,Ep2,Y,height,width,sliseCount,0,0); //
			Log.Or(Ep2,Ep1,Z,height,width,0,0,0); //
			Log.And(Z,F1,Z,height,width,0,0,0);
			Log.And(NP,F2,NP,height,width,0,0,0);
			Log.Xor(NP,Z,Z,height,width,0,0,0);
			sum(Y,X,Z,NS,NP,height,width);
			SS.YoungRZShift(NS,height,width,(sliseCount+1));
			Ep1.Dispose();
			Ep2.Dispose();
			F1.Dispose();
			F2.Dispose();
			X.Dispose();
			Y.Dispose();
			Z.Dispose();
			D.Dispose();
			NP.Dispose();
			NS.Dispose();

		}

		void NSmZv(Cell SS,Cell D,Cell PP,int height,int width,int sliseCount,char kod)
		{

			Cell NS = Cell(1);
			Cell X = Cell(1);
			Cell S1 = Cell(1);
			Logic Log;
			S1.Init(kod,height,width,0);
			Log.And(S1,SS,X,height,width,0,sliseCount,0);
			Log.And(S1,D,D,height,width,0,0,0);
			sum(D,X,PP,NS,PP,height,width);
			SS.YoungRZShift(NS,height,width,(sliseCount+1));
			NS.Dispose();
			X.Dispose();
			S1.Dispose();
}

		void Summ(Cell A,Cell B,int height,int width)
		{

			Cell Xm = Cell(M+1);
   Cell Ym = Cell(M+1);
   Cell Zm = Cell(M+1);
   Cell Dm = Cell(M+1);
   Cell Sm = Cell(2*M);
   Cell Xp = Cell(P+1);
   Cell Yp = Cell(P+1);
   Cell Zp = Cell(P+1);
   Cell Dp = Cell(P+1);
   Cell Sp = Cell(P+1);
   Cell E = Cell(1);
   Cell NUL = Cell(1);
			E.Init(1,3,3,0);
			Cell Tr = Cell(1);
   Cell R = Cell(1);
```

```
Cell R1 = Cell(1);
Cell R2 = Cell(1);
        int t1;
        Logic Log;
        for(int i=0,j=M+1;i<(P+1);i++,j++)
        {
                Xp.Equate(A,height,width,i,j);
                Yp.Equate(B,height,width,i,j);
                Sp.Init(0,height,width,i);
        }



        for(int i=0;i<(M+1);i++)
        {
                Xm.Equate(A,height,width,i,i);
                Ym.Equate(B,height,width,i,i);
                Dm.Init(0,height,width,i);
                Sm.Init(0,height,width,i);
        }

        for(int i=0;i<(P+1);i++)
        {
                Dp.Equate(Xp,height,width,i,i);
        }
        NSm(Sp,Dp,height,width,(P),0);

        for(int i=0;i<P+1;i++)
        {
                Dp.Equate(Yp,height,width,i,i);
        }
        Log.Not(Dp,Dp,height,width,0,0);
        NSm(Sp,Dp,height,width,(P),0);
        Tr.Equate(Sp,height,width,0,0);
        for(int i=0;i<P+1;i++)                {
                Log.Or(R,Sp,R,height,width,0,i,0);
        }
        Log.Not(R,R,height,width,0,0);

                while(Log.Compare(R,E,height,width,0,0)==0)
                {
                        //R.OutputData("c:\\3.txt",N1,N1);
                        for(int i=1;i<(M+1);i++)
                        {
                                Dm.Equate(Xm,height,width,i,i);
                        }
                        Dm.YoungRZShift(NUL,height,width,(M+1));
                        Log.Not(R,R2,height,width,0,0);
                        Log.Not(Sp,Dp,height,width,0,0);
                        Log.Or(Dp,R,Dp,height,width,0,0,1);
                        Log.And(Sp,R2,R1,height,width,0,0,0);
                        for(int i=1;i<(M+1);i++)
                        {
                                Log.And(Dp,Xm,Xm,height,width,1,i,i);
                                Log.And(R1,Dm,Dm,height,width,0,i,i);
                                Log.Or(Xm,Dm,Xm,height,width,i,i,i);
                        }
                        for(int i=1;i<(M+1);i++)
                        {
                                Dm.Equate(Ym,height,width,i,i);
                        }
                        Dm.YoungRZShift(NUL,height,width,(M+1));
                        Log.Or(Sp,R,Dp,height,width,0,0,1);
                        Log.And(Dp,R2,R1,height,width,0,0,0);
                        for(int i=1;i<(M+1);i++)
                        {
                                Log.And(Dp,Ym,Ym,height,width,1,i,i);
                                Log.And(R1,Dm,Dm,height,width,0,i,i);
                                Log.Or(Ym,Dm,Ym,height,width,i,i,i);
                        }
                        for(int i=1;i<P;i++)
                        {
                                Dp.Init(0,height,width,i);
                        }
                        Dp.Equate(R2,height,width,P,0);
                        NSm(Sp,Dp,height,width,P,0);
                        R.Init(0,height,width,0);
                        for(int i=0;i<(P+1);i++)
                        {
```

```
				Log.Or(R,Sp,R,height,width,0,i,0);
			}
			Log.Not(R,R,height,width,0,0);
		}
		Log.Not(Tr,R,height,width,0,0);
		for(int i=0;i<(P+1);i++)
		{
			Log.And(R,Xp,Xp,height,width,0,i,i);
			Log.And(Tr,Yp,Yp,height,width,0,i,i);
			Log.Or(Xp,Yp,Zp,height,width,i,i,i);
		}
		for(int i=0;i<(M+1);i++)
		{
			Dm.Equate(Xm,height,width,i,i);
		}
		NSm(Sm,Dm,height,width,M+1,0);
		for(int i=0;i<(M+1);i++)
		{
			Dm.Equate(Ym,height,width,i,i);
		}
		for(int i=0;i<(P+1);i++)
		{
			Dp.Equate(Zp,height,width,i,i);
			Sp.Init(0,height,width,i);
		}
		NSm(Sm,Dm,height,width,M,0);
		NSm(Sp,Dp,height,width,P,0);
		R1.Equate(Xm,height,width,0,0);
		Log.And(R1,Ym,R1,height,width,0,0,0);
		Log.Not(Xm,Xm,height,width,0,0);
		Log.Not(Ym,Ym,height,width,0,0);
		R.Equate(Xm,height,width,0,0);
		Log.And(R,Ym,R,height,width,0,0,0);
		Log.And(R,Sm,R,height,width,0,0,0);
		Log.Not(Sm,R2,height,width,0,0);
		Log.And(R1,R2,R1,height,width,0,0,0);
		Log.Or(R,R1,R1,height,width,0,0,0);
		for(int i=1;i<(M+1);i++)
		{
			Zm.Equate(Sm,height,width,i,i);
		}
		Sm.YoungRZShift(Sm,height,width,(M+1));
		Log.Not(R1,R,height,width,0,0);
		Log.Xor(R1,Sm,Sm,height,width,0,0,0);
		Zm.Equate(Sm,height,width,0,0);
		for(int i=1;i<(M+1);i++)
		{
			Log.And(Zm,R,Zm,height,width,i,0,i);
			Log.And(Sm,R1,Dm,height,width,i,0,i);
			Log.Or(Zm,Dm,Zm,height,width,i,i,i);
		}
		for(int i=0;i<(M+1);i++)
		{
			Sm.Equate(Zm,height,width,i,i);
			Dm.Init(0,height,width,i);}
		NSm(Sm,Dm,height,width,M,1);
		for(int i=0;i<(P);i++)
		{
			Dp.Init(0,height,width,i);
		}
		Dp.Equate(R1,height,width,P,0);
		NSm(Sp,Dp,height,width,P,0);
		R2.Equate(Sm,height,width,0,1);
		t1=1;
		for(int i=0;i<(M+1);i++)
	{
			Zm.Equate(Sm,height,width,i,i);
		}

		while(Log.Compare(R2,E,height,width,0,0)==0)
		{
			if(t1==(M+1))
		{
				printf("\nЗафіксовано машинний нуль");
				break;
			}
			Sm.OldRZShift(NUL,height,width,(M+1));
Log.Not(R2,R,height,width,0,0);
			for(int i=1;i<(M+1);i++)
```

```
						{
							Log.And(Zm,R2,Zm,height,width,i,0,i);
							Log.And(Sm,R,Dm,height,width,i,0,i);
							Log.Or(Zm,Dm,Zm,height,width,i,i,i);
						}
						for(int i=0;i<(M+1);i++)
						{
							Sm.Equate(Zm,height,width,i,i);
						}
						Dp.Equate(R,height,width,P,0);
						Dp.Equate(R,height,width,0,0);
						NSm(Sp,Dp,height,width,P,0);
						R2.Equate(Sm,height,width,0,1);
						t1++;
					}
					for(int i=0;i<(P+1);i++)
					{
						Dp.Init(0,height,width,i);
					}
					NSm(Sp,Dp,height,width,P,1);
					for(int i=0;i<(M+1);i++)
					{
						A.Equate(Zm,height,width,i,i);
					}
					for(int i=0;i<(P+1);i++)
					{
						A.Equate(Sp,height,width,(M+1+i),i);
					}
					Tr.Dispose();
					R.Dispose();
					R1.Dispose();
					R2.Dispose();

	}
};
```

New.cpp

```
#include "stdafx.h"
#include "OLD_libr.h"
#define N1 3
#define N2 2*N1
#define M 47
#define P 15
#define FractionLength M
#define OrderLength P
#define S M+P+2
#define NumberLength S
typedef char CELL[N1][N2];
#include "binar.h"
#include "BinarLogic.h"
#include "BinarSum.h"
#using <mscorlib.dll>
using namespace System;

int maximal(int n,float R0[]);

int _tmain()
{
	Cell a = Cell(NumberLength);
	Cell b = Cell(NumberLength);
	Cell sum = Cell(NumberLength);
	Cell sum1 = Cell(NumberLength);

	printf("A:\n");
	a.InputData("c:\\A.dat",N1,N1);
	printf("B:\n");
	b.InputData("c:\\B.dat",N1,1);
	float aa[N1][N2];
	float bb[N1][N2];
	float aa1[N1][N2];

	a.OutputDataFl(aa,3,3);

	Adder A;
	Logic l;

	printf("\n");
	a.OutputData("c:\\3.txt",N1,N1);
	b.OutputDataFl(bb,3,1);
```

```
for(int i=0;i<N1;i++)
{
		bb[i][0] = bb[i][0]/aa[i][i];
		for(int j=0;j<N1+1;j++)
		{
				if(i!=j)
				{
						aa[i][j] =  aa[i][j]/-aa[i][i];
				}
		}
		aa[i][i] = -1;
}
for(int i=0;i<N1;i++)
{
		aa[i][N1] =  bb[i][0];
}

float R0[N1][N1];
float RX[N1][N1];
float R1[N1];

for(int j=0;j<N1;j++)
{
		R0[j][0] = 0;
}
for(int u=0;u<N1;u++)
{
		R1[u] =R0[u][0];
}
double x0[N1];
for(int j=0;j<N1;j++)
{
		x0[j] = 0;
}

double x[N1];


double S1,det;

S1=0.0;
for(int i=0;i<N1;i++)
{

		for(int j=0;j<N1;j++)
		{
				S1 = S1+ aa[i][j]*x0[i];
		}
}
for(int i=0;i<N1;i++)
{
		R0[i][0]=bb[i][0]-x0[i]+S1;
		printf("R[%d]=%-5.3f\n",0,R0[i][0]);
}
for(int u=0;u<N1;u++)
{
		R1[u] =R0[u][0];
}
int f=maximal(N1,R1);
det=R0[f][0];
int iter=8;
for(int k=0;k<iter;k++)
{
		printf("iteration %d\n",k);
		printf("det[%d] %-5.3f\n",k,det);
		sum.InputDataFl(R0,N1,N1);

		for(int i=0;i<N1;i++)
		{
				for(int u=0;u<N1;u++)
				{
						for(int v=0;v<N1;v++)
						{
								RX[u][v] = 0;
						}
				}
				if(i!=f)
						RX[i][0] = aa[i][f]*det;
				else
```

```
                        RX[i][0]=-det;
                    sum1.InputDataFl(RX,N1,N1);
                    A.Summ(sum,sum1,N1,N1);
                }
                sum.OutputDataFl(R0,N1,N1);
                for(i=0;i<N1;i++)
                {
                    printf("R[%d]=%-5.3f\n",i+1,R0[i][0]);
                }
                x[f]=x[f]+det;
                for(int u=0;u<N1;u++)
                {
                    R1[u] =R0[u][0];
                }
                f=maximal(N1,R1);
                det=R0[f][0];
        }


        printf("New X:\n");
        for(int i =0;i<N1;i++)
        {
                printf("X[%d]=%-5.3f\n",i+1,x[i]);
        }
        return 0;
}
```

Додаток К

Приклад роботи програми